SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Informatics

From Curious Kids to Data Wizards: An Empowerment Approach to Data Science Education

Hanya Elhashemy

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Informatics

From Curious Kids to Data Wizards: An Empowerment Approach to Data Science Education

Von neugierigen Kindern zu Daten-Zauberkünstlern: Ein Empowerment-Ansatz für die Vermittlung von Datenkompetenzen

Author:	Hanya Elhashemy
Supervisors:	Prof. Tilman Michaeli (TUM), Prof. Harold Abelson (MIT)
Submission Date:	15.09.2023

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Munich, 15.09.2023 Hanya Elhashemy

Acknowledgments

I wish to use this opportunity and pay my special regards to my supervisors, Prof. Harold Abelson and Prof. Tilman Michaeli, for allowing me to write my thesis under their guidance. I am immensely grateful to Prof. Michaeli, for his insightful feedback and constructive critiques, which greatly enriched the quality of this thesis.

I owe a debt of gratitude to Prof. Abelson, you are a true inspiration to me and millions of people. I feel incredibly fortunate to have learned from your deep knowledge. I still remember the Email I sent you, shooting my shot and asking if we could work together, although I lived on the other side of the world. This email is the best decision I made in my academic journey so far. Thank you for believing in me and giving me the chance and resources to implement my vision. Your mission to empower the next generation with technology is truly commendable, and I am deeply privileged to be a part of it.

I would like to take this opportunity to express my deepest gratitude to the App Inventor team: Evan Patton, who helped me with the software design of my data science components and the integration of the components in App Inventor; you are the reason 20 million users have access to these components today. Selim Tezel and Robert Parks, who supported me with their valuable feedback during the design of my data science curriculum. I wish to also thank David Kim, who onboarded me in the team; it was a pleasure working with you. Jeff Schiller and Susan Lane, thank you for the great discussions and explanations of App Inventor's development history. I am honored for the opportunity to have worked with all of you!

Besides, I would like to recognize the great work of the students from the Undergrad Research Opportunities Program (UROP): Jennet Zamanova, Jacky Chen, Arianna Scott, Ava Muffoletto, Gisella Kakoti, Isabela Sanchez Taipe, and Matthew Quispe. I want to extend my heartfelt thanks to you. Your flow of ideas contributed to increasing the efficacy of this thesis. Especially Jennet, thank you for helping me facilitate my workshop, I am really grateful for your time and support.

On a broader scale, I would like to sincerely thank Cynthia Rosenthal and Marisol Diaz for their support organizing my paperwork to travel to MIT as a visiting student.

In addition, I would like to thank Jeff Freilich for providing the Android tablets to my workshop participants. I'm so grateful for everyone from Responsible AI for Social Empowerment and Education (RAISE) initiative who helped make this possible.

Most importantly, I wish to express my deepest appreciation to my family. Although they were overseas, they called me every day and checked up on me during stressful times. They always had my back and supported me through my whole degree and study abroad. Without you, writing this thesis and all the other achievements in my life would not have been possible. I am forever grateful. Thank you!

Abstract

The increasing importance of data science in various fields has underscored the need to democratize data science, ensuring that a broader range of individuals can participate in this rapidly growing interdisciplinary field. The master thesis aims to address this topic by introducing (1) a data action educational framework, and (2) a data science toolkit within the widely used platform "App Inventor" at MIT.

The toolkit enables high school students with no prior data science background to build mobile applications that collect, process, and analyze real-world data. It allows them to experiment with Machine Learning (ML) models, like anomaly detection and regression, and integrate these into their projects to solve impactful real-world problems.

Recognizing the barriers that prevent students from engaging in authentic data science practices, such as cost, access, and complexity, the developed open-source toolkit provides ready-to-use data science code-blocks, conveying necessary data science capabilities with a low learning curve and no required programming knowledge.

Students can develop data thinking skills, including problem-solving and critical thinking, by engaging through the toolkit with data science practices across the entire data science lifecycle, including data collection, cleanup, visualization, and prediction.

The data action educational framework enables K-12 students to foster essential data literacy skills by empowering them to use data to solve real-world problems of high relevance to the students and their community. This shows them the potential of data science and their ability to make an impact through their data-driven solutions.

By making data science more accessible to K-12 students, this research strives to break down barriers and empower a new generation of diverse data scientists. Ultimately, it contributes to the broader goal of democratizing data science education and fostering a more inclusive and diverse data-driven future.

Contents

1. Introduction

"As data, open, big, personal or in any other guise, becomes increasingly important, power will flow to those who are able to create, control and understand data. Those who cannot, will become powerless. Further, their ability to participate in society will be severely challenged as they lack the tools to engage with an important raw material of society." [FW16].

Today, data plays a vital role in our society, and ML algorithms use data to make decisions on our behalf that impact our future. Schools have not caught up with the data revolution. Little to no schools integrated a data science curriculum covering a complete data lifecycle (see related work chapter 3), leaving the next generation with a lack of data literacy, making them vulnerable to disadvantages in this data-driven world.

Data science courses outside high schools are often not tailored for K-12 education. In addition, the high cost of courses or the advanced programming languages used in a course, assuming prior computational knowledge, could be a barrier for students, preventing them from taking the step to start learning data science.

Existing tools like Python notebooks can be overwhelming for beginners, limiting their potential to explore data science fully. Additionally, the lack of products enabling data visualization on mobile platforms has restricted students' ability to interact with data effectively.

1.1. Motivation

"Many people encounter Artificial Intelligence (AI) without being aware of it or understanding how the technology functions" [Wil21]. "Curious kids" play with an AI chatbot friend, listen to a bedtime story created by an AI tool, or draw on a digital canvas with an AI artist's support. They are amazed by the "magic" of AI, but to be true "data wizards" capable of understanding and recreating this magic, they must learn the internals of what makes AI powerful on a deeper level.

The "magic potion" of AI is made out of data. For a powerful model to produce powerful outputs, it must take the right data as input. Thus, data science practices, especially data cleanup, are crucial to creating AI. Some even state that "80% of AI is data cleanup" [1]. However, data cleanup could be challenging and time-consuming, especially when working with real-world data containing real errors [Li+21]. As for children, cleanup is usually the least exciting task.

All these reasons led to K-12 educators eliminating this step when introducing students to AI or data science in general, eliminating a tremendous learning opportunity to acquire a crucial skill. Educators provide students with preprocessed data as a black box. This stops students from reaching their potential as "data wizards".

We often observe educators focusing on teaching algorithms, statistical methods, or ML models, sometimes overlooking the application of these methods to solve real-world problems.

Some data science high school tasks prompt learners to implement a particular algorithm on a provided dataset. In such instances, educators often design an "ideal" dataset tailored exclusively to the specific algorithm to highlight the algorithm and its associated mathematical and statistical processes. This could result in students memorizing algorithms without knowing when to apply them in the real world. The lack of association between concepts and their application could decrease students' learning motivation.

Learners need to experiment with various data science techniques to decide which one is the best for the real-world problem at hand. They need to realize that the standard algorithms aren't always customized for real-world data. They often need to adjust those algorithms to match the characteristics of their specific real-world data.

Rather than instructing various data science tools, we propose initiating the learning process by introducing the diverse applications of data science. This approach aims to inspire students to recognize their own real-world challenges that they would like to address using data science techniques.

Working on real-world problems in data science also means working with real-world data. This comes with additional challenges necessary for young people to learn how to overcome.

One of the foremost challenges revolves around ensuring the quality and integrity of the data itself. Real-world data is frequently characterized by imperfections such as errors, missing values, and anomalies, which can distort the accuracy of analytical results.

Moreover, the diversity of real-world data, presented in structured, semi-structured, and unstructured formats, requires storage, processing, and analysis techniques. Data integration from diverse sources, often with varying structures and quality levels, poses another obstacle. Understanding the complexities of the domain to which the data belongs is imperative for accurate interpretation.

[1] https://www.forbes.com/sites/forbestechcouncil/2022/04/20/managing-the-data-for-the-ai-lifecycle/

These challenges offer valuable chances for students to learn, which are sometimes overlooked if educators focus solely on teaching the data science algorithms rather than teaching their application in the real world. In addition to acquiring these essential skills, students will develop computational identity and digital empowerment by working on solving their self-identified problems using data science.

We argue that working on solving real-world problems using data science would (1) increase students' motivation to learn data science, (2) enhance their data literacy and data thinking skills, (3) change their perspective about themselves and about data science, prompting them to acknowledge the impact of data science.

1.2. Objectives

This thesis aims to empower K-12 students to be data scientists. Based on the previous sections, we can derive the following main objectives for this thesis:

Goal 1: Increase Students' Motivation for Learning Data Science
Data science is not a common high school subject. Many high schoolers don't even know what data science is and why it is essential to discover this field. We aim to integrate a data science curriculum in high schools and motivate students to learn data science by enabling them to work with datasets and projects that they define and choose themselves. We plan to provide learners with tools that lower the entry barrier to data science and provide a risk-free environment to experiment with different data science algorithms.

Goal 2: Raise Students' Awareness about Data-Related Topics
By allowing students to experience the data lifecycle first-hand, we raise their awareness about relevant topics related to data science, like data privacy, the gender data gap, and biases in ML algorithms. Students experience how anomalies influence their prediction results and manipulate data-driven decisions. Students see the number of powerful insights extracted from datasets. This and similar aspects make them realize a data scientist's power and responsibility to create impactful data-driven solutions.

Goal 3: Elevate Students' Skills Set with Data Thinking
The goal is to extend students' horizons with data literacy, allowing them to observe their world through data and obtain a data-thinking mindset. The transformation from computational thinking to data thinking is especially important today, with the amount of data exponentially increasing and impacting the daily lives of all individuals. We focus primarily on data cleanup, as it is a crucial step for optimizing the results of data analysis and ML models, however often left out by educators. We elaborate more on the definition of data thinking and the required data literacy skills in section 2.1.2.

1.3. Key Contributions

This thesis contributes to the field of data science education for young people by presenting:

- A data action educational framework consisting of four topics: (1) Defining a real-world problem; (2) self-acquiring knowledge of the problem's domain; (3) working with real-world data critically; (4) understanding the impact of data.
- Curriculum for K-12 students, teaching them the data action topics.
- Data Science components in App Inventor, allowing students to apply the data action framework and experience a data lifecycle by developing mobile apps with data science features.
- Results from a research study measuring the efficacy of the data action framework on students' ability to understand the influence of data and use data science to impact their communities.

1.4. Outline

In the following, we describe the outline of this thesis. Chapter 2 provides the essential theoretical foundation needed to grasp the subject of this thesis. Chapter 3 presents related work and compares it with the approach adopted in this thesis. Chapter 4 elicits the requirements of the proposed App Inventor data science components and its software architecture. It puts the proposed components into perspective with the current App Inventor system, and presents example apps created with the proposed data science components. Chapter 5 describes the data action educational framework and how it is applied to empower students with data science. Chapter 6 describes the research study we conducted to evaluate the educational framework, and the proposed data science components we introduce to App Inventor. Additionally, we also elaborate on the study's results. Chapter 7 concludes the thesis with an overview and provides an outlook of possible future work.

2. Background

To understand the thesis approach, which centers around empowering kids with data science, it is essential to delineate a clear definition of data science, along with a thorough explanation of the fundamental skill set required for data scientists.

Furthermore, a comprehension of the technical aspects associated with a data science project and the inherent challenges encountered while working with real-life data is essential. Such insights will determine which skill sets are particularly necessary to equip kids.

In addition, it is equally important to comprehend the impact of data science in the 21st century. This knowledge will determine the importance of data science education for future generations. By thoroughly investigating these foundational aspects of data science and its relevance in the current era, the thesis aims to establish a groundwork for exploring the transformative potential of data science education in empowering young people.

This chapter presents the findings of a literature review on data science. First, we examine the definition of data science and discuss data literacy. Next, we focus on the various technical phases of a data science project solving a real-world problem. We also identify the essential skill set data scientists must have to navigate these phases effectively. Lastly, we examine the impact of data science and discuss the role of data science in computational action.

2.1. Data Science

In recent years, the significance of data has been increasingly recognized, with some likening it to the valuable resources of the 21st century, such as gold [EA16]. The exponential growth of data has necessitated the development of solutions for its storage/collection, processing, and analysis. As a response to this demand, the field of data science emerged in the early 1960s [Don17], gaining significant attention as an interdisciplinary academic field [Dha13]. Data Science leverages computer science, statistics, and domain expertise to derive insights from data [Cao17]. In this section, we define what data science is, which skills are acquired in data science, and the technicalities and phases of a data science project. In addition, we discuss the impact of data science and introduce the role of computational action in data science.

2.1.1. Data Science Definition

Koby Mike and Orit Hazzan argue that one can not describe data science in one definition as it is a multifaceted field. It can be defined as a science, a research paradigm, a research method, a discipline, a workflow, and a profession[MH23].

Data science focuses on extracting insights from data to solve a problem in a wide range of different application domains [VA16]. It combines principles from mathematics, statistics, computer science, and domain expertise to achieve that [Hay98]. As an interdisciplinary field, data science integrates domain knowledge from the application domain where it is applied [VA16].

Data Science as a Science: Empirical science is a scientific approach that relies on observation, experimentation, and evidence-based data to develop and test hypotheses, theories, and models [Hoo94]. Data science is rooted in empirical science, where data is used to understand natural phenomena and judge theories [Tuk62]. Today, Data is considered a natural resource used to make scientific discoveries.

Data Science as a Research Paradigm: Some researchers describe data science as the "fourth paradigm", building upon empirical, theoretical, and computational paradigms [Hey09; BHS09]. This paradigm deals with exploring and examining data to infer new scientific findings. Under this definition, data science is considered a new research method that transforms the research process from deductive to inductive, enabling open-ended analysis[MH23].

Data Science as a Workflow: Others describe data science as the workflow that includes all aspects of working with data, from gathering and cleaning to analyzing and visualizing [Mar23]. Data science is an iterative cycle that involves transforming raw data into meaningful insights and impactful actions [Win19]. We describe the detailed phases of the data science cycle in section 2.1.3.

Data Science as a Profession: With the exponentially increasing amount of data emerged the demand for employees capable of working on data-driven projects. In such projects, data scientists take advantage of the data resources to increase the business value of their company [Yau09]. Insights from data can be used to increase sales, enhance marketing, and offer more personalized customer service [Vou14].

Data scientists are high-ranking professionals with training to navigate the vast amount of data to uncover insights and drive decision-making. The current demand for data scientists is so high that it was declared several times as the "sexiest job of the 21st Century", showing the relevance of the skill set of a data scientist [DP12].

Data Science as a Discipline: Data science is an interdisciplinary field which requires expertise in multiple disciplines like computer science, mathematics, statistics, and an application domain [MD18]. Drew Conway visualizes the integration of different disciplines in Data science using Venn diagrams [Con10]. Following, others developed different visualizations based on Conway's vision [Tay16].

In the scope of this thesis, we examine data science as a discipline and discuss its integration in K-12 education. Tailored to high school students, we introduce our simplified version of Conway's Venn diagram (see 2.1).

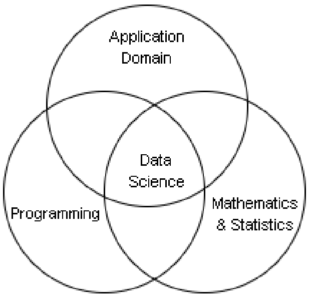


Figure 2.1.: Venn diagram visualizing the intersection of different disciplines with Data science

Data Science Vs. Statistics

In comprehending data science, a field, as described above, characterized by its strong interdisciplinary nature, it becomes essential to explore its connections with other disciplines. Notably, statistics plays a significant role, as it shares numerous synergies with data science [Has+21].

While some argue that data science is only another name for statistics [Sil20; Don17], many agree that data science must be distinguished from statistics. Data science focuses on digital data-driven solutions of any arbitrary application domain, thus widening the horizon of statistics and extending it with more data types and applications [CM18; Dha13].

While statistics mainly deals with quantitative data, data science deals with quantitative and qualitative data (e.g., data from sensors, images, text, and surveys) and focuses on extracting actionable knowledge from this data. Gelman takes it a step further and describes statistics as a "non-essential part of data science" [Gel13].

Brad Efron describes the relationship between statistics and data science as follows: "I tell my fretful friends that we have a strong positive regression coefficient with data science, as long as we remember not to let the inferential side of statistical thinking get lost in the excitement over new technology" [Has+21].

In the scope of this thesis, we see statistics as a subset of data science as displayed in 2.1. It is necessary for high schoolers to first grasp basic math and statistics before they can take the leap and understand data science and its intersection with these disciplines.

Furthermore, data science could increase their motivation for their standard statistics and math subjects as it shows them the application of these rather theoretical disciplines in the real world and the impact of these disciplines in solving problems of relevance to the students.

Thus, a data science course would build upon the knowledge and methodologies taught in their standard math and statistics classes. Its purpose, among others, is to bring mathematics and statistics to life by using them with application domain knowledge to solve a real-world problem of great relevance to the students.

Data Science Vs. AI

In addition to defining the intersections of data science with related disciplines, it is crucial to establish the relationship between data science and other computer science fields, particularly AI. The interchangeable use of data science and AI often leads to misconceptions and confusion.

As data science includes AI as a subset, exploring the relationship between the two becomes imperative. Understanding how AI fits into the broader data science landscape enhances our comprehension of its applications and potential to solve complex problems. AI encompasses complex computer algorithms that simulate human intelligence. It enables machines to “learn” and improve problem-solving as they process more data.

AI is concerned with algorithm design, development, and deployment to emulate cognition and human understanding. It involves building models that can autonomously handle tasks without human intervention [McC+07], while Data Science focuses on data analysis, finding hidden patterns and trends, prediction, and visualization. In summary, AI is a part of the more general field of data science, just like ML and DL.

2.1.2. Data Literacy

Like data science, data literacy has multiple definitions due to the complexity of the construct. It encompasses a diverse range of knowledge and skills, leading to variations in terminology across different groups based on their respective roles. Mandinach and Gummer attempt to find a common definition for data literacy [MG12]. They argue that a clear definition of data literacy helps professional development providers determine which skills and knowledge should be taught. Defining data literacy allows schools to incorporate data-driven skills into their curricula.

Data literacy is a contemporary addition to an expanding collection of literacies, including numerical literacy, statistical literacy, and digital literacy. Each of these literacies pertains to the proficiency in utilizing widely accessible mediums or technologies deemed crucial in today's context [Fra+16]. They all originate from the general concept of literacy, which refers to the ability to read and understand text. Based on that, we define data literacy, on a high level, as the skills required to understand and use data.

Through a literature review, we identified these six essential skills required for a data-literate individual:

1. **Real-World Problem-Solving:** The ability to problem-solve in data literacy involves identifying real-world problems that can be addressed using data [Dea14; MG13]. It also includes defining the problem's scope and formulating analytical approaches to solve it [Vah+06; BD15; Wil+14]. A data-literate individual should understand the role and impact of data in various societal contexts [Dea14].

2. **Collect Data:** Data literacy requires comprehending the nature of data, the diverse types of data available, and the processes involved in data generation and collection [GCB12; CM13; MG13; Wil+14]. It consists of the proficiency to gather qualitative and quantitative data through various methods, such as conducting interviews, designing surveys, making observations, and performing measurements. It is also necessary to examine and compare different data sources to check for bias [Dea14].

3. **Analyze Data:** Data literacy includes the ability to preprocess data for analysis [Wil+14; Shi05], develop hypotheses to be examined using data [MG13], critique data, and examine it to create explanations and actionable knowledge [WP99; Car+15]. It requires the ability to characterize data [Dea14] and know which tools and algorithms to use for a particular dataset [GCB12; Vah+06]. This also includes understanding the challenges attached to working with datasets with specific characteristics, e.g., big datasets, structured/ unstructured datasets, time series, cross-sectional datasets, and social network datasets.

4. **Evaluate Data:** An essential skill of a data-literate individual is evaluating the results of their data analysis and using such results as evidence to support their hypothesis [Dea14]. It's necessary to be able to assess the data quality and present the results of the data quality assessment. A data-literate individual should be able to formulate new research questions based on evaluating the validity of their data-driven insights[Wal93; Cal06].

5. **Visualize Data:** Communicating the results of data-driven projects through visualizations, such as tables, graphs, and maps, is an essential skill in data literacy [Dea14]. A data-literate individual should be able to understand visualizations and use them to present the data to different stakeholders. Depending on the

application domain of the data-driven project, different visualization techniques and graph types should be used. Visualizing the data results is essential, as it helps communicate important insights to other individuals who are not data literate. This could significantly impact society and motivate positive action based on data evidence.

6. **Ethics:** A data-literate individual must recognize the significance and influence of their skill set and data-driven projects on society. Ethical considerations are pivotal in data science, and awareness of key challenges is crucial. These include privacy and anonymity concerns, potential data misuse, the importance of data accuracy and validity, and the risks associated with model misuse or misinterpretation [Wil+14; SDH18; CJ15].

Data Thinking

Besides the data literacy skill set, defining the cognitive process or a mindset that involves approaching problems and decision-making with a data-centric perspective is necessary. Reflecting on the Venn diagram showing the different disciplines (see 2.1) that intersect with data science, Koby et al. suggest that "each discipline contributes its unique [cognitive] skills" [Mik+22], which implies that data thinking integrates computational thinking, statistical thinking, and domain thinking.

The authors argue that data thinking enhances computational thinking and statistical thinking. By integrating real-life data into the problem-solving process, data thinking enriches the understanding and relevance of domain knowledge. It also provides students with the opportunity to develop computational models and algorithms that are based on real-life data, making their solutions more meaningful and effective [Mik+22].

Besides the enhancement of computational thinking and domain thinking, we elaborate on additional cognitive skills that data thinking promotes:

1. **Problem Abstraction:** Data thinking promotes problem abstraction, which includes the ability to formulate and decompose a problem to be solved using data science[Mik+22].

2. **Pattern Recognition:** Data thinking enhances individuals' ability to recognize patterns and notice irregular phenomena in, but not limited to, data.

3. **Continuous Learning:** Data thinking increases a person's ability to continuously learn as they get used to improving models and solutions through constant and iterative monitoring and data collection.

4. **Critical Thinking:** Data thinking asks critical questions about the data, such as whether it offers a good representation of the real-life situation and how the data collection can be improved [CC18]. It is necessary to question the data, as real-life data often has errors. Data thinkers must be aware that models could produce deceiving results, so they have to critically examine their results.

5. **Decision-making:** Data thinking improves decision-making skills, as it promotes informed decisions based on a comprehensive understanding of the data and its context.

The skills mentioned above are essential not just for data science professions, but for any individual to succeed in the 21st century. In today's digital age, students must be able to effectively find, understand, evaluate, and use data [TF09]. They also need to be proficient in using technology tools. Problem abstraction, pattern recognition, continuous learning, critical thinking, and decision-making are all necessary skills in day-to-day life, not just related to the tech field.

2.1.3. Data Lifecycle

The data lifecycle offers a conceptual representation of the entire lifespan of data objects, from their creation to eventual removal or archival. In this context, data objects refer to collections of files, links, or databases [PK17]. The data lifecycle outlines a series of stages that may not apply to every object. It is a structured and systematic approach that ensures data is handled efficiently, securely, and ethically throughout its existence.

The data lifecycle enhances data quality and reliability through standardized processes, reducing errors and inaccuracies. It streamlines data-related tasks, making data management more efficient and effective [Ray13]. Access to data in a usable format allows for evidence-based decision-making and strategic planning.

For high schoolers or beginners learning data science, the data lifecycle would serve as a guideline or path they could follow. Such a guide would decrease the intimidation of starting a new data-driven project from scratch and learning a new field like data science, as it provides learners with a red path to follow.

Various perspectives exist regarding the specific stages of a data lifecycle [Bal12]. In figure 2.2 we represent our version of a data lifecycle decoupled from any specific application or technology.

The cycle starts with defining a problem to be addressed using data. In this phase, we identify questions and assumptions that we validate using specific data. We set the scope and the goals of the data-driven project, and then start the search for relevant data sources.

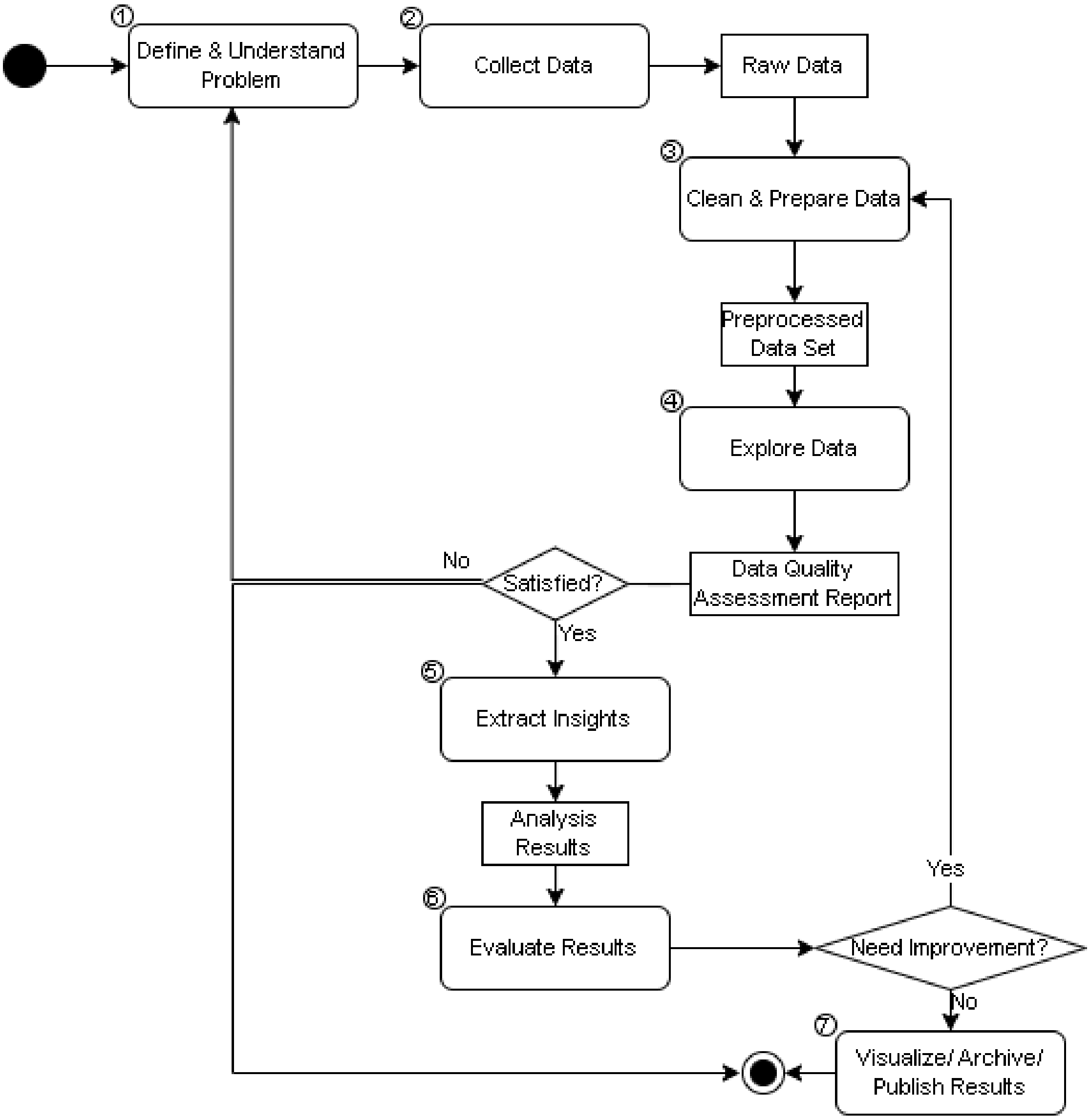


Figure 2.2.: Activity diagram representing the iterative data lifecycle process

The next phase revolves around generating new data or collecting existing data [Con06]. This phase is the beginning of a technical physical pipeline, the "data pipeline", that manifests most of the data lifecycle phases. Depending on the data type, this phase is automated in many situations through a script that extracts data from different devices and stores them somewhere, e.g., on the cloud. As a result of this step, we get raw data stored in a particular location.

The third phase deals with cleaning up and processing the raw data. This includes identifying the data features and organizing the data according to their features. As a result, we get a dataset in the desired shape and file extension (e.g., CSV). Additional data manipulation or sampling might occur in this phase, like sorting or grouping the data according to a parameter value.

Besides re-shaping and organizing the data, we start the data-cleanup process. This could include 1. removing illegal values (e.g., values outside set bounds), 2. removing misspellings, 3. removing rows with missing values, 4. removing duplicates, and other measurements specific to the dataset at hand [RD+00].

After resolving obvious data issues through cleaning, we delve deeper into data exploration during phase four to identify more comprehensive dataset characteristics. Data exploration helps in understanding the data and identifying any patterns or trends. Researchers can gain insights into the relationships between variables and identify patterns by visually exploring the data through graphs and plots.

This phase allows detecting and addressing of any violations of assumptions that may affect the validity of the analysis. By examining the data distribution, checking for anomalies, and assessing the homogeneity of variance, researchers can ensure that the statistical techniques used are appropriate and reliable [ZIE10]. This ensures that the conclusions drawn from the analysis are based on solid evidence, not just chance findings.

As a result of the data exploration phase, we create a data quality assessment report. The report discusses whether the identified problem (from phase 1) can be addressed with this data. It also describes the data, its characteristics, and identified data errors.

Based on the report, there are three possible outcomes: 1. We re-define the problem (from phase 1), 2. We determine that the data quality is sufficient to proceed and solve the problem identified in the first phase, 3. We determine that the data quality is insufficient to proceed to solve this or similar problems, and we stop the process and publish the data quality assessment report as our gained insights.

If the data is sufficient to solve the problem (identified in phase 1), we dive deeper and continue extracting more insights from the data. This phase could include creating a machine learning model and training on the preprocessed data to generate more insights. Polyzotis et al. describe a detailed ML cycle with additional phases, which include the development of the model in a ML pipeline, the training of the model on different data inputs, and the validation of the model by comparing labeled training data with input data to identify any deviations. [Pol+18b]. We will omit the specific details of a ML pipeline since it falls outside the scope of this thesis.

After generating the initial analysis results, it becomes essential to continuously evaluate them iteratively. The trained model is tested in the evaluation phase to determine its performance and accuracy. The evaluation module checks if the model has acceptable accuracy that meets the desired criteria and if any adjustments or improvements need to be made. The results are published and archived with each improvement, till the end of the project.

Application of Data Science

Data Science finds extensive applications across diverse industries and domains, making it a versatile and indispensable field in today's data-driven world [VA16; EH95]. Some prominent application domains where data science finds extensive applications include:

Business Analytics: Data Science helps businesses analyze customer behavior, market trends, and operational efficiency, leading to informed decision-making and better strategic planning [Pac22].

Healthcare and Medicine: Data Science is utilized to analyze patient data, predict disease outcomes, improve diagnostics, and enhance treatment protocols [Das+19].

Finance and Banking: Data Science is used for fraud detection, credit risk assessment, investment analysis, and customer segmentation to optimize financial operations [Azz+18].

E-commerce and Retail: Data Science enables personalized recommendations, demand forecasting, inventory optimization, and customer churn analysis [AW16].

Marketing and Advertising: Data Science is employed to target specific audiences, optimize ad campaigns, and measure marketing effectiveness [Sau21].

Social Media and Sentiment Analysis: Data Science helps in understanding customer sentiment, trends, and behaviors on social media platforms [Pra+22].

Manufacturing and Supply Chain: Data Science is used for predictive maintenance, supply chain optimization, and quality control [TWD18].

Transportation and Logistics: Data Science assists in route optimization, traffic prediction, and fleet management [Spe18].

Energy and Utilities: Data Science helps in smart grid management, and energy consumption optimization [Hea15].

Government and Public Policy: Data Science is employed for data-driven policy decisions, crime prediction, and resource allocation [KP18].

The applications of Data Science continue to grow as businesses and organizations increasingly recognize its potential to gain valuable insights from data and drive innovation in various sectors.

2.2. Advancements in Data Science

In this section, we explore the latest developments in data science, focusing on their relevance to high school data science education. We start with an in-depth examination of computational action, showcasing how this approach can be applied to enhance data science education in high schools. We then delve into strategies for making data science accessible to a broader audience, highlighting the democratization of this field. Additionally, we underscore the significance of diversity, equity, and inclusion in data science, emphasizing its role in driving progress. Finally, we introduce the open data movement, shedding light on its benefits and potential impact on the community.

2.2.1. Computational Action in Data Science

Tissenbaum, Sheldon, and Abelson introduce the computational action approach in education, which suggests that alongside acquiring computational knowledge, young individuals should learn computation by working on projects of significant relevance to them, directly influencing their lives and communities [Tis+18b]. Instead of focusing on "What should learners learn", computational action shifts the focus to "What do learners want to learn". The motivation to learn will increase if the focus is set on the individual rather than on the learning objectives [Tis+18a].

In this perspective, they delineate two fundamental facets of computational action: computational identity and digital empowerment [TSA].

Computational identity denotes an individual's acknowledgment of their computational competence in solving problems of big impact [MT10]. Computational identity encompasses perceiving oneself as a member of a wider network of digital creators. Rather than working on generic tasks with predetermined answers, learners are empowered to define their own tasks. The tasks should be relevant to the learners, and their solutions should impact the students and their community. This leads to digital empowerment [Fre93; TV90], which centers on instilling learners with the confidence that they can translate their computational identity into tangible actions with purposeful implications concerning real-world problems of significance to them.

Within the scope of this thesis, we present in chapter 5 a framework, the data action framework, that extends the computational action approach to the realm of data science, empowering students to work on solving impactful real-world problems using data science, and experiencing a complete data lifecycle while working with real-world data.

2.2.2. Democratizing Data Science

Textual literacy has been an indispensable prerequisite for active social participation. Literate individuals have consistently enjoyed more extensive life opportunities compared to the illiterate. A society's development greatly hinges on a high level of literacy. As the Internet emerged, data literacy is poised to assume a comparable level of importance, promising to shape and influence society in profound ways [Fra+16].

We contend that today, it is imperative for every individual to possess fundamental data literacy and data thinking proficiencies. Individuals lacking these skills are disadvantaged because data science finds applications across various disciplines, including non-technical domains. Hence, encountering data science is inevitable for all individuals at some point in time.

Automated resume checks, personalized shopping recommendations, or curated social media content are all instances of data science shaping our daily experiences. From determining our job prospects to guiding our everyday purchases and shaping our online content consumption, these examples vividly illustrate the pervasive reach of data science into our daily lives. Such impact underscores the need to equip individuals with data literacy and data thinking skills. Fostering these skills is crucial to preventing societal disadvantages from a lack of understanding in this data-driven era.

This underscores the significance of democratizing data science and mitigating the entry barriers into this domain. Lowering the entry barriers in data science requires a comprehensive approach that fosters accessibility and inclusivity. This can be achieved through different strategies to make the subject more approachable and attainable for a broader audience.

Firstly, offering easily accessible learning resources is crucial. Providing affordable or free courses, tutorials, and learning platforms catering to various skill levels can help learners engage with the fundamentals of data science. Flexibility in learning formats, such as part-time or online courses, accommodates individuals with diverse schedules and commitments.

Interactive learning environments play a pivotal role as well. These platforms enable hands-on experience with real-world datasets and tools, allowing learners to experiment and gain practical skills in a risk-free setting. Promoting the use of open-source tools and software reduces financial barriers and fosters a sense of community participation.

A well-defined learning pathway is essential in guiding newcomers through the complexities of data science. By breaking down intricate concepts into manageable components and providing a sense of progression, learners can confidently navigate the subject.

Demystifying the mathematical and coding aspects is another critical factor. Simplifying complex concepts through relatable explanations, hiding complex code syntax with block-based programming languages, and practical examples can alleviate intimidation and enhance comprehension.

Highlighting the practical applications of data science across various domains underscores its relevance in solving real-world challenges. This approach can spark engagement and motivation by demonstrating the tangible impact of the field.

Encouraging collaborative learning through online forums, study groups, and community events creates a supportive environment where learners can share insights, ask questions, and collaborate on projects. Representation matters, so showcasing diverse role models who have succeeded in data science can inspire more individuals to pursue the field.

By implementing these strategies collectively, the barriers preventing individuals from entering the data science field can be significantly diminished, creating a more accessible and inclusive landscape that embraces diversity and encourages skill development across various backgrounds and proficiency levels.

2.2.3. Diversity, Equity, and Inclusion in Data Science

We discussed in the sections above the positive impact of data science; however, data science could also have the potential to be employed in areas like mass surveillance, computational propaganda, and biased decision-making that can lead to discrimination [SM19].

To highlight the extent of this concern, ongoing research conducted regarding bias reduction in AI reveals that among approximately 133 biased systems spanning various industries from the year 1988 to the present day, a substantial 44.2% (59 systems) exhibit gender bias, and 25.7% (34 systems) manifest both gender and racial bias [DK20]. Additionally, this showcases the importance of data cleanup to examine biases.

To address this, there is also a common understanding that diversifying the data science workforce can help mitigate the risk of generating unfair and discriminatory data analysis results. Diverse teams enhance fairness by identifying and rectifying algorithm biases, and bring varied perspectives that foster innovation.

A comprehensive understanding of diverse populations ensures more accurate solutions, while ethical considerations are magnified, ensuring responsible technology development. Diversity, Equity, and Inclusion (DEI) fosters socially responsible practices, user-centric designs, and attracts broader talent [HBH22].

Integrating data science into K-12 education is crucial for building a diverse and skilled data science workforce. This is particularly significant in K-12 education, where STEM skills are nurtured. It's essential to counter stereotypes and misconceptions about data science careers, highlighting their broad applicability across various fields.

To achieve this, positive social messages and role models need to challenge existing stereotypes. However, this transformation requires more than just education—it demands a cultural shift. Organizations, educators, and media play pivotal roles in fostering an environment where women and underrepresented groups in tech can thrive in data science, bolstered by evidence of their potential success in the field [BB15].

2.2.4. Open Data Movement

The open data movement is a global initiative promoting transparency, collaboration, and accessibility by making various data types freely available to the public. Although the term "open data" has not yet reached two decades of existence, the concept of making scientific research universally accessible was championed by Robert King Merton as far back as the early 1940s. The principle is that research, which generates valuable data, should be freely shared for the greater benefit of society [1].

This movement seeks to remove barriers that prevent the use of data collected by governments, organizations, researchers, and individuals. The goal of the open data movement is to encourage innovation, informed decision-making, and public participation, supporting data science democratization as mentioned in 2.2.2 by providing unrestricted access to information that was traditionally locked behind bureaucratic barriers [Baa15]. By sharing data openly, proponents of the movement aim to enable individuals to utilize data for various purposes, from addressing societal challenges to creating new applications and solutions [BGV15].

In education, we also observe a shift driven by the educational data movement. This movement leverages the power of data-driven insights to inform educational decisions, identify areas for improvement, personalize learning experiences, and enhance educational policy-making. The educational data movement aims to harness the potential of data to create more tailored and impactful learning experiences for students, while also improving the overall educational system [Pie15].

[1] https://data.gov/blog/open-data-history

3. Related Work

This chapter provides an overview of different data science educational approaches. Besides, we examine existing educational tools enabling children to develop computational and data literacy skills.

3.1. Data Science Educational Approaches

This section will present the different approaches to integrating data science K-12 education in schools. We will illustrate several data science curricula that will serve as our basis for the data action framework presented in section 5. This is a reference point for teachers to integrate data science into their classes. Due to the low number of existing data science curricula for K-12, we also include a presentation of AI curricula in this section as we defined AI as a subset of data science in chapter 2.1.1 and thus some concepts of AI curricula could be reused for a data science curriculum.

Heinemann et al. present a draft data science curriculum for secondary schools (age 15-18) with four modules:
(1) "From Data to Information": This module introduces students to data and data science, emphasizing statistical thinking and data competency. It aims to enhance their understanding of how data can be transformed into information using statistical methods. They differentiate between information and ding: "Information is construed as understanding data, which happens only in a human mind. The term data is used to describe the representation of values in a machine." [Hei+18].

(2) "Big Data and Artificial Intelligence": This module mainly covers machine learning concepts, focusing on decision trees and artificial neural networks. Students will gain practical knowledge in applying these techniques. (3) "Data Projects": Students work with real-world datasets in this module to deepen their data literacy and practical skills. (4) "Data Science and Society": This module explores the societal implications of data science. Students present their projects, discussing the information they've gathered and the broader societal impact of their findings [Hei+18].

Michaeli et al. present a concept for teaching students from age 14 about Decision Tree Learning (DTL). The concept is structured into four phases: (1) Students use an unplugged activity (without using technology) to manually develop and test a decision tree for a provided data set. (2) This is followed by training a model using a visual programming tool for the same data set, allowing students to compare the manual and automated approaches. (3) Next, students apply their newly acquired knowledge to real-world datasets through a project phase. (4) Finally, the focus shifts to the social aspect, addressing questions related to transparency, fairness, and security of AI processes that arise from the application of DTL [Mic+23].

Olari and Romeike argue that existing educational approaches lack coverage of data literacy competencies. The existing approaches cover some relevant competencies within different stages of the data lifecycle, but not all. They suggest that future approaches should encompass skills such as extracting information from subject-specific data (Acquisition stage), cleansing and addressing fairness in datasets (Cleansing stage), implementing data management systems (Implementation stage), enhancing data through AI techniques (Optimization stage), visualizing data (Visualization stage), analyzing data with AI methods to generate insights (Analysis stage), evaluating and questioning results (Evaluation stage), and implementing processes for data archiving, deletion, and exchange (Sharing, Erasing, and Archiving stages) [OR21].

Based on these findings, Grillenberger and Romeike developed a "Data Literacy Competency Model" displayed in figure 3.1 as a structured approach to understanding and teaching data literacy. The competency model is divided into two parts: content areas and process areas. Content areas focus on the theoretical background and scientific concepts related to data literacy, while process areas emphasize the practical activities and practices involved in working with data. The model recognizes the interconnection between these areas and highlights the need for a comprehensive understanding of both [GR19].

Furthermore, they present a lesson sequence using the data literacy competency model. The competencies targeted in this lesson include explaining the function principle of a simple data analysis method, explaining how insights can be predicted based on existing data, interpreting predicted insights, and reflecting on the limitations and threats of data analysis [GR18]. By combining practical activities with theoretical knowledge, the lesson sequence aims to develop students' data literacy skills and empower them to apply their knowledge to solve real-world problems.

Zhou et al. present an extended framework for K-12 AI education, which aims to enhance existing AI curricula by offering additional guidelines to enrich AI learning experiences for students. The framework includes a focus on more comprehensive AI literacy objectives, strategies to engage students effectively through gamification and embodied learning, scaffolding techniques for gradual comprehension of complex AI concepts to overcome the barrier of lacking a computer science background, involvement of teachers and parents, and promoting equity, diversity, and inclusion. The framework also encourages the integration of AI education with core curricula, fostering interdisciplinary learning and the development of computational thinking

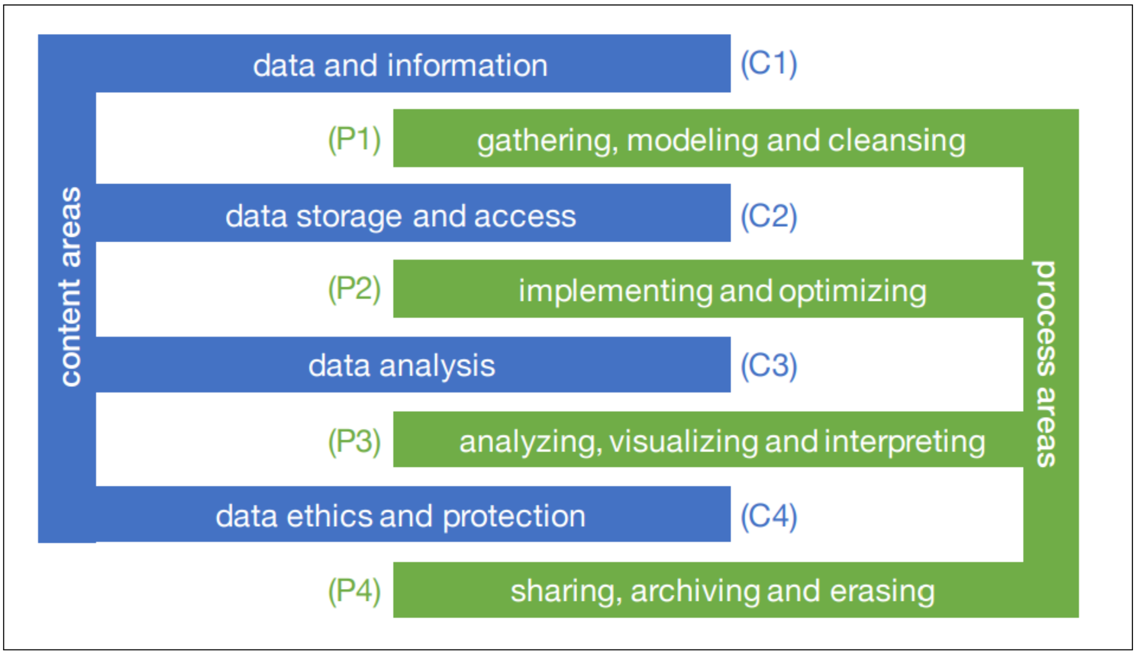

content areas
data and information
(C1)
(P1)
gathering, modeling and cleansing
data storage and access
(C2)
(P2)
implementing and optimizing
data analysis
(C3)
(P3)
analyzing, visualizing and interpreting
data ethics and protection
(C4)
(P4)
sharing, archiving and erasing
process areas

Figure 3.1.: The data literacy competency model [GR18; GR19]

skills [ZVL20].

Conclusion

Most approaches presented above focus on the content rather than the learner. Their starting point for their curriculum design is “What should K-12 students learn” instead of “What do K-12 students want to learn”. In addition, most approaches start with theoretical content before allowing students to experiment with their own data-driven projects. They explain to the students the terminology in the early stage of the curriculum, instead of letting them learn it naturally by experiencing a data lifecycle through a data-driven practical project.

So we see, for example, Heinemann et al. starting with a theoretical introduction of data science explaining the difference between data and information, Grillenberger and Romeike also starting their lesson plan with theoretical knowledge explaining the function principle of a simple data analysis method, and Michaeli et al. designing the curriculum around the ML model DTL, putting learning the algorithm in the focus.

In addition, most curricula focus on how a ML algorithm works, delving into the mathematical and statistical details of the algorithm design, neglecting the application of the algorithm in solving real-world problems, and the significant impact of data cleaning on the algorithm output. The lesson plans lack motivation for the students to critically question data points and experiment with how removing specific data points could completely manipulate the results of a ML model.

As we learned above, Olari and Romeike highlighted the lack of coverage of some data literacy competencies. What stands out the most after we examined related work is, in particular, the lack of educational material covering data cleanup. This finding is surprising, considering the importance of data cleanup. Data cleanup is the basis for the quality of data-driven insights, and numerous research confirms the importance and effect of data cleanup on accuracy [Cha20; Li+21; Zha+17; Jes+18].

In chapter 5 we present our own data science educational framework. The framework builds upon existing frameworks and aims to teach K-12 students the learning objectives discussed here while putting the learners in focus. It provides an environment where students can experiment and experience a data science lifecycle while building a project that has a significant meaning to them and their communities and uses real-world data to solve impactful real-world problems.

Our approach starts with a practical project instead of theoretical knowledge, with the goal that students naturally acquire the theory through working on the project. Another aspect that differentiates our approach from existing data science curricula is the integration of data cleanup in an interactive project-based process, empowering students to think critically about data and its impact.

3.2. Data Science Educational Tools

This section presents different educational tools that aim to introduce beginners to programming and implement the curricula shown above. We focus on tools that are suitable for data science K-12 education. The tools are divided into four categories: text-based, block-based, hybrid-based, and component-based tools.

3.2.1. Text Based

Traditional general-purpose programming languages often include a wide range of features and concepts that can overwhelm beginners. Text-based educational tools aim to simplify general-purpose programming languages by focusing on essential concepts, making it easier for learners to grasp fundamental programming ideas without unnecessary complexity.

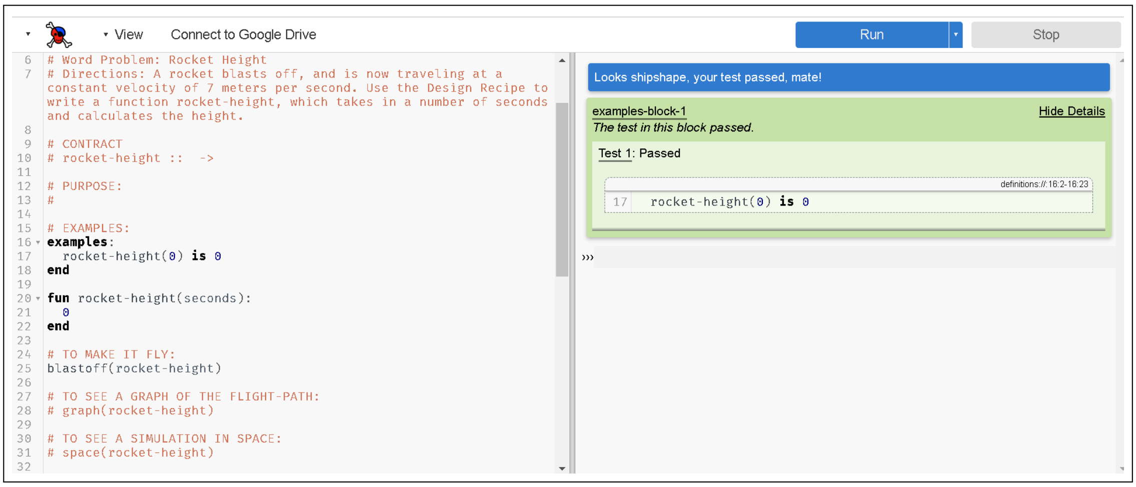
Text-based educational tools designed for beginners typically have more straightforward and readable syntax. They use common words and structures easier for learners to understand, reducing the cognitive load associated with learning programming concepts. In the following, we will discuss two examples of text-based educational tools.

Pyret

Pyret[1] is an educational text-based programming language designed to introduce beginners to the fundamentals of programming. Developed by Brown University, Pyret focuses on providing a data-centric computing environment for teaching coding and data science concepts to a wide range of learners, including those without prior programming experience [Pol+18a]. Pyret syntax is designed on purpose to be similar to Python's, a common data science language (though without semantic whitespace). This aims to provide learners an easier transition to professional data science tools [Fis22].

Key features of Pyret include its strong type system, which helps catch errors early in the development process, and its support for functional programming concepts. It offers a range of built-in data structures and functions, making it easier for beginners to work with data and perform common programming tasks.

Bootstrap [2] offers several data science workshops using Pyret. Figure 3.2 shows a starter file of one of Bootstrap's workshops. Learners can run Pyret directly in their browser or their editor of choice by installing Pyret's npm package.



```
# Word Problem: Rocket Height
# Directions: A rocket blasts off, and is now traveling at a
constant velocity of 7 meters per second. Use the Design Recipe to
write a function rocket-height, which takes in a number of seconds
and calculates the height.

# CONTRACT
# rocket-height ::  ->

# PURPOSE:
#

# EXAMPLES:
examples:
  rocket-height(0) is 0
end

fun rocket-height(seconds):
  0
end

# TO MAKE IT FLY:
blastoff(rocket-height)

# TO SEE A GRAPH OF THE FLIGHT-PATH:
# graph(rocket-height)

# TO SEE A SIMULATION IN SPACE:
# space(rocket-height)
```

Looks shipshape, your test passed, mate!

examples-block-1 Hide Details

The test in this block passed.

Test 1: Passed

definitions://:16:2-16:23

```
17    rocket-height(0) is 0
```

›››

Figure 3.2.: Web-based editor of the text-based educational programming language Pyret

[1] https://pyret.org/index.html

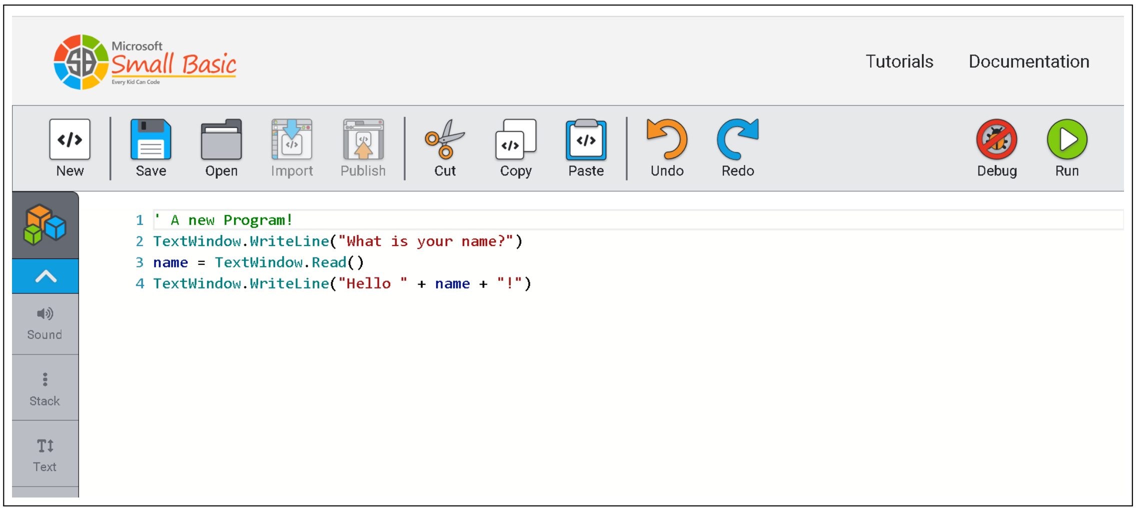
[2] https://www.bootstrapworld.org/materials/data-science/

Small Basic

Similar to Pyret, Small Basic[3] is also a text-based programming language designed by Microsoft to help students learn coding and transition from block-based coding to text-based coding. Small Basic is a simplified version of the general-purpose programming languages BASIC. It is based on .NET; therefore, any learned concepts can be easily transferred to a .NET programming language like Visual Basic.

Different from Pyret, Small Basic is not particularly designed for data-centric computations, but rather for learning general computation concepts. However, this does not exclude Small Basic text-based language from data science education, as it is a complete language that offers multiple data structures and the possibility to include third-party libraries.

Small Basic has its own interpreter and coding editor (offline as well as web-based) (see 3.3). In addition, Microsoft provides teaching material for free to learn to code using Small Basic.



```
' A new Program!
TextWindow.WriteLine("What is your name?")
name = TextWindow.Read()
TextWindow.WriteLine("Hello " + name + "!")
```

Figure 3.3.: Web-based editor of the text-based educational programming language Small Basic by Microsoft

3.2.2. Block Based

Block-based programming is a visual approach to coding where programming concepts are represented as blocks that can be dragged, dropped, and snapped together to create

[3] https://smallbasic-publicwebsite.azurewebsites.net

coding programs. Instead of typing out lines of text in a traditional programming editor, beginners use a graphical interface to assemble blocks corresponding to different programming commands and functions.

Each block typically represents a specific action or operation, such as loops, conditional, mathematical operations, and more. These blocks often have a clear visual representation and are color-coded to indicate their purpose. Users can create functional programs by arranging these blocks in a logical sequence.

Block-based programming languages are prevalent for teaching beginners and children how to code. They offer a more intuitive and visually appealing way to learn programming concepts without getting bogged down by syntax errors.

Blockly is a widely used framework for creating block-based programming languages, and various platforms like Scratch, Snap!, and MIT App Inventor (described in chapter 4) are built on top of Blockly to provide engaging learning environments for programming. In the following, we will describe Scratch and Snap! in more detail, as well as showcase another block-based editor specifically designed for learning data science: The Microsoft Data Science Editor.

Scratch

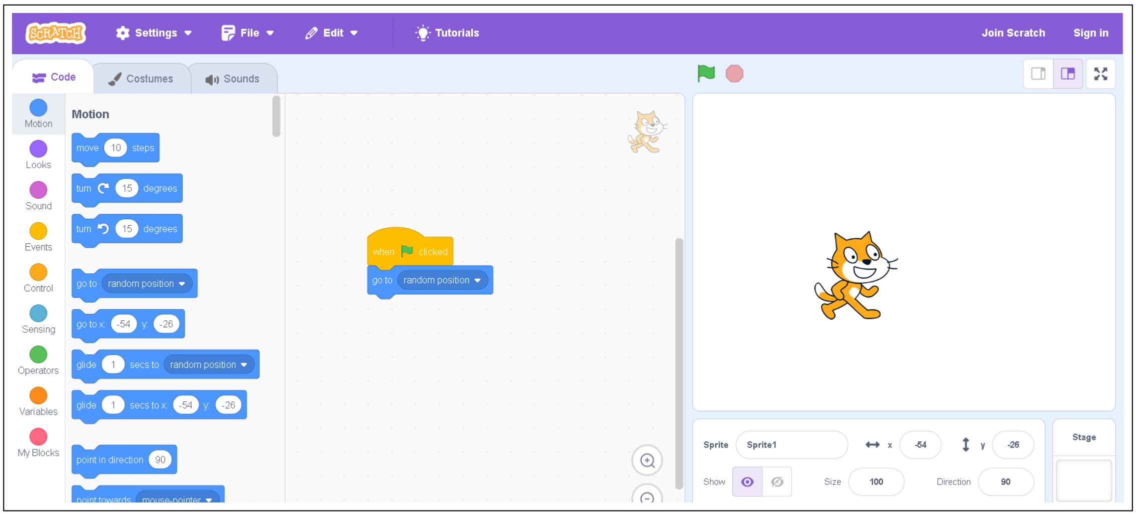
Scratch[4] is a block-based programming web platform developed by MIT's Media Lab, aimed at teaching coding to beginners, particularly children and young learners aged 8 – 16 [Res+09]. It provides an interactive and visual platform for creating animations, games, stories, and interactive projects, without requiring knowledge of traditional programming languages.

Users can arrange and connect blocks representing various coding commands. These blocks, akin to puzzle pieces, facilitate the creation of code sequences. The tool offers a range of code blocks covering movement, loops, conditions, variables, events, and more, enabling users to craft scripts governing the behavior of sprites or objects on the project stage [Mal+10].

Scratch empowers users to design and select sprites and backgrounds, then control their appearance, movement, and interactions via code blocks (see 3.4). This extends to project-sharing within an online community, fostering collaboration and feedback among users. According to the Scratch foundation website [5], Scratch is the "world's largest coding community for kids", with more than 100 million registered users.

[4] https://scratch.mit.edu

[5] https://www.scratchfoundation.org/our-story



Sprite Sprite1 x -54 y -26
Show Size 100 Direction 90

Figure 3.4.: Web-based editor of the block-based educational tool Scratch

Snap!

Snap![6], is an extension of Scratch, developed by UC Berkeley. It retains Scratch's foundational aspects but adds advanced features and capabilities. Snap! supports higher-level programming concepts like recursion, higher-order functions, and object-oriented programming. It offers greater flexibility and control, allowing users to delve into more complex programming techniques while still utilizing the visual block-based approach [FGF17].

Snap! also emphasizes modularity and abstraction, enabling users to create custom blocks and procedures (see 3.5). While Snap! may offer more powerful capabilities, it may also have a steeper learning curve compared to Scratch due to its advanced features.

Both block-based tools are designed to teach general computer science concepts, not particularly tailored for data science education. However, there exist some extensions of Scratch and Snap! that revolve around data science education.

Scratch Community Blocks, for example, is an extension of Scratch, allowing children to engage in data science. It enables children to programmatically access, analyze, and visualize data about their participation in Scratch, allowing them to explore and understand their own learning and social data [DH17].

[6] https://snap.berkeley.edu

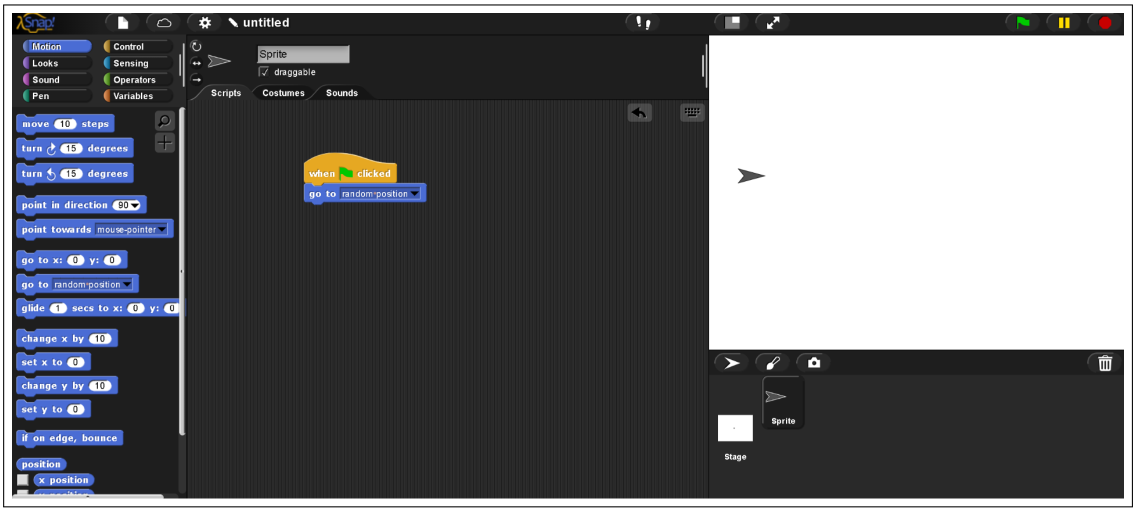


Figure 3.5.: Web-based editor of the block-based educational tool Snap!

Microsoft Data Science Editor

Microsoft created a block-based editor, particularly for data science[7]. According to their website: "The Data Science Editor is an experimental-structured editor to create data analysis programs"[8]. The editor can be used in the browser [9], in Excel Web, or in Visual Studio Code.

The editor has several datasets already integrated into the tool for the user to experiment with or upload their own dataset, either from a file or a URL. The editor supports datasets < 10 MB. The data blocks have a data preview button, as seen in 3.6. If a block manipulates the data, the preview button shows a before and after view of the dataset for the user to compare the differences. The data science blocks are divided into the following categories: Datasets, Cleanup, Organize, Compute, Visualize, Statistics, Data Variables, and Charts, covering all the data lifecycle activities.

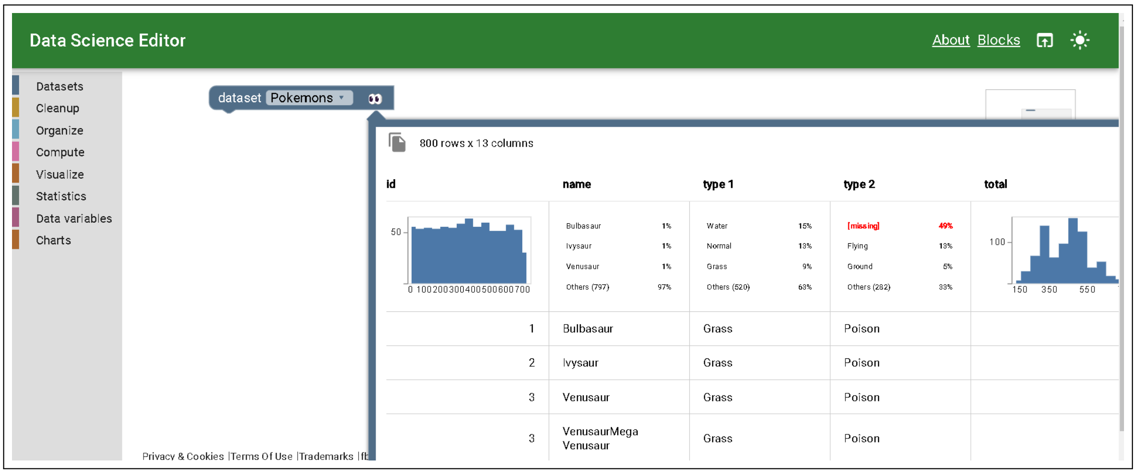
3.2.3. Hybrid Based

Hybrid-based programming tools bridge the gap between block-based and text-based programming languages, offering a transition for learners as they progress in their coding journey. These tools combine the visual simplicity of block-based programming with the more advanced capabilities of text-based languages [MLD18; LW21].

[7] https://microsoft.github.io/data-science-editor/about/

[8] https://microsoft.github.io/data-science-editor/about/

[9] https://aka.ms/ds



800 rows x 13 columns

id	name	type 1	type 2	total
	Bulbasaur 1% Ivysaur 1% Venusaur 1% Others (797) 97%	Water 15% Normal 13% Grass 9% Others (520) 63%	[missing] 49% Flying 13% Ground 5% Others (282) 33%	
1	Bulbasaur	Grass	Poison	
2	Ivysaur	Grass	Poison	
3	Venusaur	Grass	Poison	
3	VenusaurMega Venusaur	Grass	Poison	

Figure 3.6.: Microsoft's Web-based data science block-based editor

In the initial stages, learners use visual blocks to create code by dragging and connecting blocks representing programming constructs. This approach particularly benefits beginners as it provides a tangible and visual representation of code logic. As users become more comfortable with these foundational concepts, hybrid tools allow them to gradually transition to text-based coding by introducing textual elements alongside the visual blocks [ALM19].

Hybrid-based programming tools often provide features like code autocompletion, real-time error feedback, and syntax highlighting to support learners as they transition. These tools are designed to accommodate learners at various stages of their coding journey, making them suitable for both beginners and those seeking to deepen their programming skills.

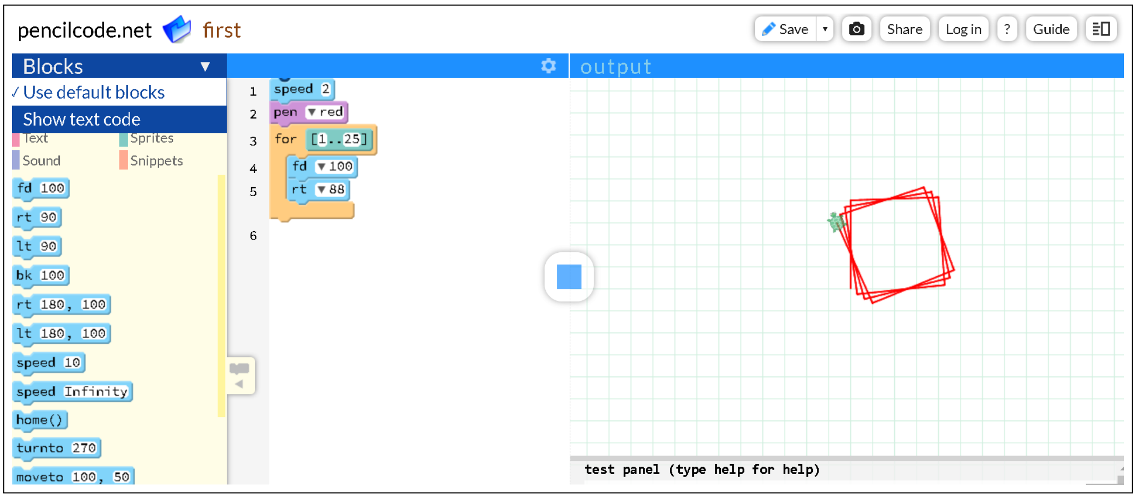
Pencil Code

Pencil Code[10] is an educational coding platform designed to help beginners learn programming. Pencil Code provides a text editor (see 3.8) where users can write code using JavaScript programming language. Alongside this text editor, it includes visual code blocks (see 3.7) that represent coding concepts [BB14]. A dropdown menu lets Learners switch between the code-blocks and the code-text view. This hybrid approach allows beginners to grasp coding logic visually while gradually transitioning to writing code directly [BDB15].

A standout feature of Pencil Code is its "Turtle Graphics," which enables users to

[10] https://pencilcode.net

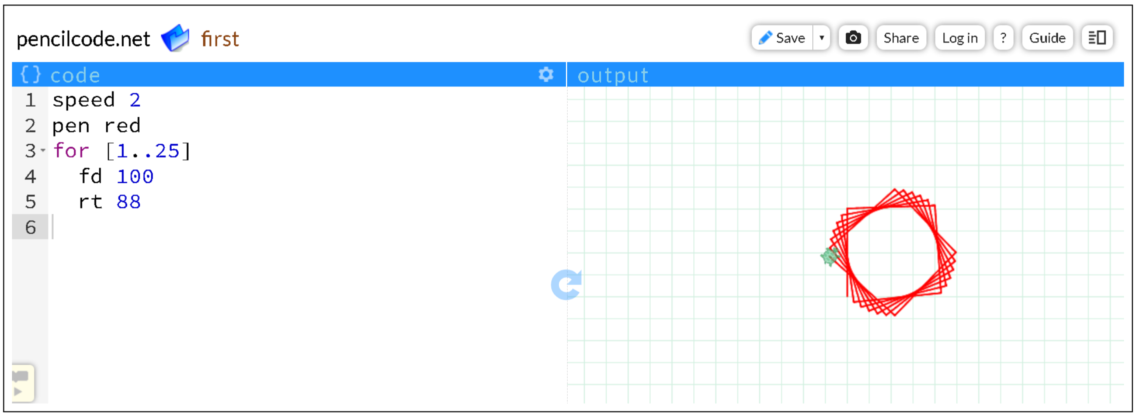
control a virtual turtle using their code. This turtle can create drawings and patterns on the screen, providing an intuitive way to comprehend programming concepts like loops, conditions, and functions [Bau+15]. Additionally, Pencil Code supports real-time collaboration, allowing users to share their projects with others for joint learning.



```
speed 2
pen red
for [1..25]
  fd 100
  rt 88
```

test panel (type help for help)

Figure 3.7.: The code-blocks view of the educational web tool Pencil Code, showing an example program



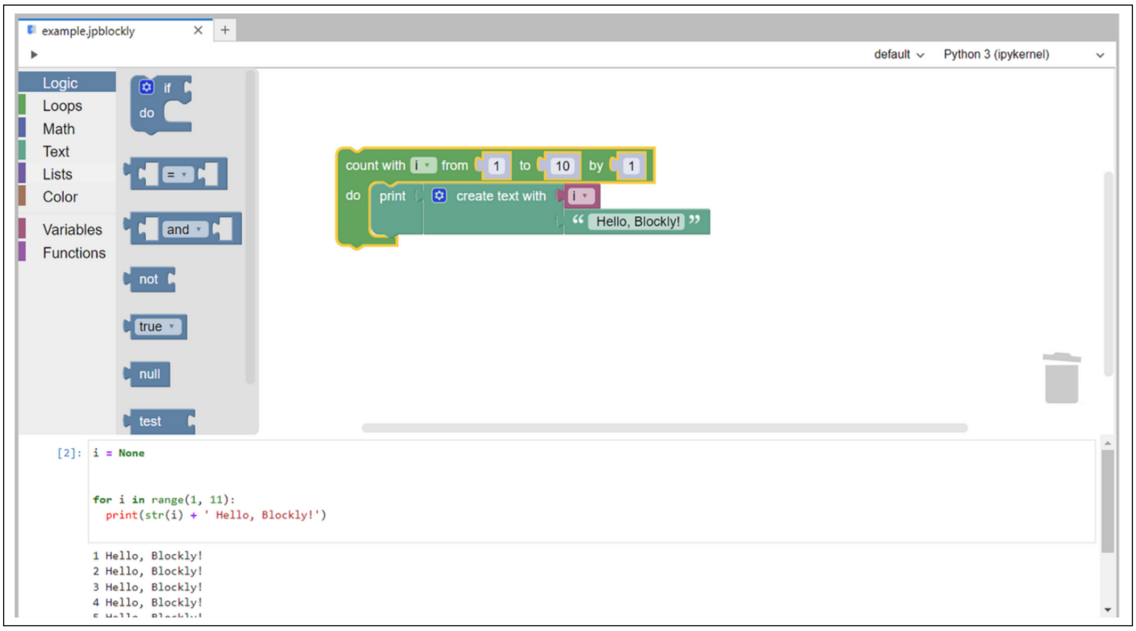
```
speed 2
pen red
for [1..25]
  fd 100
  rt 88
```

Figure 3.8.: The code-text view of the educational web tool Pencil Code, showing an example program

JupyterLab Blockly

JupyterLab Blockly[11] integrates the Blockly framework by Google into JupyterLab, a prominent environment for the development of computational notebooks in the field of data science [OFJ21]. JupyterLab's web-based notebook features cells that can hold text, code, or multimedia outputs, allowing code execution and referencing previous cell context. This enhances code organization compared to traditional line-by-line interaction.

JupyterLab employs a plugin architecture, enabling modification without altering the source code, termed JupyterLab extensions [OF21]. In this case, Blockly is a JupyterLab extension. Users can build programs (incl. data science projects) using the visual Blockly code blocks inside JupyterLab, which then get rendered in text-based code (e.g., Python) in a JupyterLab notebook cell as displayed in 3.9. Users can leverage visual blocks to create code and data manipulation workflows, enabling beginners to learn programming concepts more easily.



```
[2]: i = None


for i in range(1, 11):
  print(str(i) + ' Hello, Blockly!')
```

```
1 Hello, Blockly!
2 Hello, Blockly!
3 Hello, Blockly!
4 Hello, Blockly!
5 Hello, Blockly!
```

Figure 3.9.: The view of JupyterLab editor incorporating Blockly extension, showing both the code blocks and the corresponding code cell

[11] https://github.com/QuantStack/jupyterlab-blockly

3.2.4. Component Based

Component-based programming is a software development approach where a software application is constructed by assembling and connecting individual, self-contained units called "components". Each component encapsulates a specific functionality or behavior and can be considered a modular building block. Components are designed to interact with each other through well-defined interfaces, allowing them to communicate and collaborate.

In this approach, developers create components that are reusable, interchangeable, and loosely coupled. This means that components can be easily modified without affecting the rest of the application. Each component hides its internal implementation details, exposing only the necessary interfaces for interaction. The components are more abstract than the code blocks approach explained above because components encapsulate a complete functionality, while code-blocks encapsulate single operations used to create a functionality.

Component-based programming offers several advantages, including code reusability, maintainability, and scalability. Developers can focus on creating specialized components without having to build an entire application from scratch or deal with code syntax details. This approach simplifies software development by breaking down complex systems into manageable and reusable parts that can be reused.

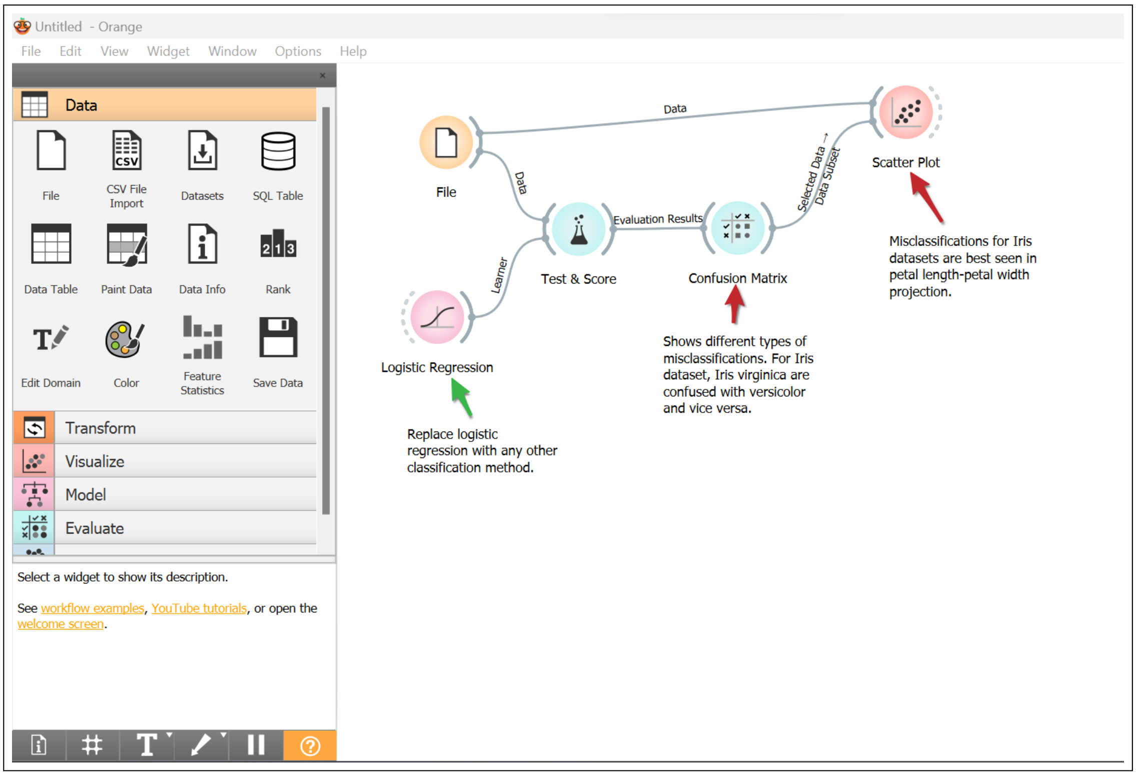
Orange

Orange[12] is a component-based tool for data mining and machine learning. It is designed for (1) experienced data scientists who want to experiment with creating new algorithms in Python while reusing existing code, and for (2) beginners who can leverage Orange's visual programming interface and avoid having to deal with complex code syntax. Users must install the tool; there is no browser version. The installation process is rather complex for beginners as it requires installing Anaconda. On the other hand, users can create a full data analysis pipeline without any Python programming or scripting.

Orange offers various data science features, from data preprocessing to modeling and evaluation. These encompass functionalities like data management and preprocessing tasks such as filtering, scaling, and attribute creation, as well as the development of classification and regression models, including decision trees, naive Bayesian classifiers, and support vector machines. The platform has evaluation and scoring methods for prediction models, featuring multiple scoring techniques and visualization tools [Dem+04].

[12] https://orangedatamining.com

Users can place pipeline components called widgets on Orange's canvas to create a data analysis pipeline and visualize the data flow (see 3.10). Each widget offers a basic functionality. The user connects these widgets through communication channels. Most functionalities are coupled with a visualization that allows exploring the results of the functionality. For example, the user can "select a node in a classification tree or rule, and explore the training instances covered by them" [DZ13; DZ12].



Select a widget to show its description.

See workflow examples, YouTube tutorials, or open the welcome screen.

Misclassifications for Iris datasets are best seen in petal length-petal width projection.

Shows different types of misclassifications. For Iris dataset, Iris virginica are confused with versicolor and vice versa.

Replace logistic regression with any other classification method.

Figure 3.10.: The editor view of the component-based tool, Orange, showing a simplified data analysis pipeline

Conclusion

From the above presented educational tools, only Pyret 3.2.1, Microsoft Data Science Editor 3.2.2, JupyterLab Blockly 3.2.3, and Orange 3.2.4 are tailored for data science education. Pyret and Orange have a higher learning curve compared to block-based approaches. Pyret is too concrete, and Orange is too abstract in comparison to block-based approaches and, thus, are more suitable for learners transitioning to large-scale production programming, not necessarily our target group of high schoolers (aged 14 to 18) with no prior programming and data science knowledge.

None of the educational tools allow students to build projects that visualize data on mobile platforms — which, after all, is where many scientists consume data visualizations today [Sil09]. In addition, the above data science tools focus on the data science functionalities rather than on the children themselves and what they would want to create with these functionalities. None of the tools take the resulting data visualizations and insights further to create a fully functional product a child would like to develop, interact with, and show off to their friends.

Therefore, we propose introducing data science blocks to App Inventor in the following chapter. This addition would enable children to design mobile apps that are not only interactive but also incorporate data analysis and leverage data collection through mobile apps. These apps would not just showcase data on graphs; they would go a step further by utilizing the data analysis outcomes to create apps that address real-world problems relevant to the children. This approach empowers children to take action based on their data analysis and solve issues relevant to them and their community.

4. Introducing Data Science to MIT App Inventor

MIT App Inventor is an open-source web platform for block-based programming, enabling everyone to build Android and iOS mobile apps to solve real-world problems in their communities [IE17]. The tool is designed to democratize computing and allow individuals to develop their computational thinking skills [WAF15]. More than a million users from over 190 countries use MIT App Inventor monthly to create mobile apps that positively affect their communities. Today, users created a total of 85.5 million mobile apps [1].

In the context of this thesis, we introduce an addition to the mentioned platform, introducing data science blocks to democratize data thinking. Our proposed system facilitates the transition of learners' computational thinking competencies to include data thinking.

In this chapter, we describe MIT App Inventor — from the tool's inception at Google in 2008, through the migration to MIT, to its current state. Then, we discuss our vision for the proposed system, introducing data science to MIT APP Inventor. We will illustrate in detail the proposed system's architecture and design goals. We conclude by showcasing example apps of students with no prior data science experience who created these apps using the proposed system to solve real-world problems relevant to them and their communities.

4.1. Existing System

In this section, we will describe the design of MIT App Inventor and the methodology behind the chosen design. We will further illustrate the tech stack used to build MIT App Inventor. Lastly, we will examine the current features that support our goal of integrating data science in App Inventor and discuss any potentially missing parts.

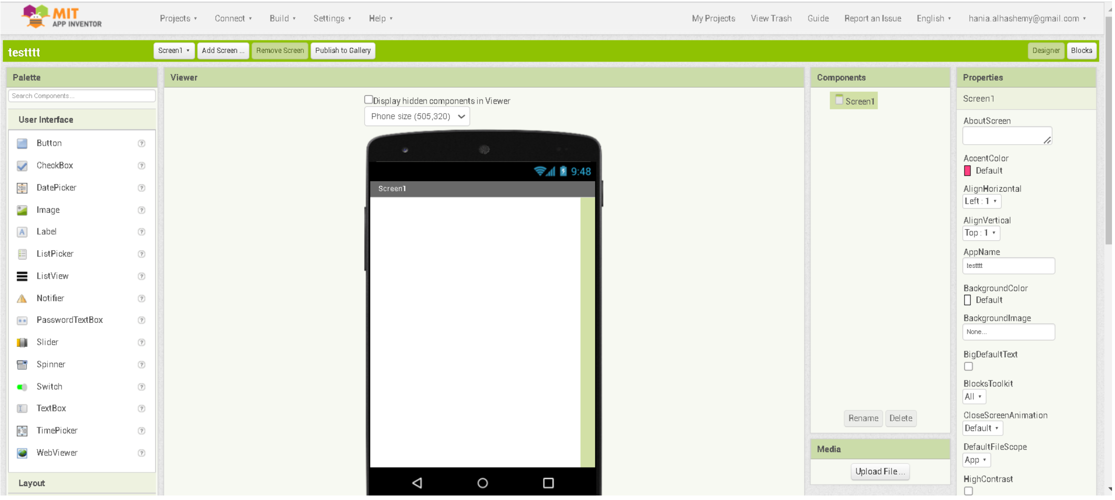
[1] `https://appinventor.mit.edu/`

4.1.1. Design Methodology

App Inventor consists of two main views: The designer editor view (see 4.1), and the block editor view (see 4.2). In the designer editor, users can drag and drop components from the left panel to the phone simulation view in the middle of the screen. It is a "What you see is what you get" (WYSIWYG) editor. There are two types of components: visible and non-visible components [XA16].

Visible components are components displayed on the phone screen, e.g., buttons, text fields, and labels, while non-visible components are hidden from the phone screen, e.g., database, and sensors. Users can customize component properties using the right panel. App Inventor's design editor enables developers to preview the app's appearance on the device screen and customize the form factor of the simulated device (e.g., phone or tablet).

In addition to the properties, components also have methods and events. Methods are a set of procedures that perform specific tasks. Events are triggered by external factors that change the app's state. The goal of components is to reduce the complexity of using platform-specific application programming interfaces (APIs). For example, App Inventor reduces the global positioning system (GPS) implementation from 629 lines of Java code to 23 simplified blocks [PTH19].



□ Display hidden components in Viewer
Phone size (505,320)

Screen1
AboutScreen
AccentColor: Default
AlignHorizontal: Left : 1
AlignVertical: Top : 1
AppName: testttt
BackgroundColor: Default
BackgroundImage: None...
BigDefaultText: □
BlocksToolkit: All
CloseScreenAnimation: Default
DefaultFileScope: App
HighContrast: □

Figure 4.1.: MIT App Inventor designer editor for adding App components

The block editor is the space to add logic and functionality to the designer components via drag and drop of code blocks, that "snap together like puzzle pieces to describe the program" [PTH19]. There are two types of code blocks: general built-in blocks and component-specific blocks. The general built-in blocks resemble the basic computational concepts: procedure, variable, logic, loop, conditional, list, and dictionary (see figure 4.2). The component-specific blocks allow the manipulation of their corresponding components. App Inventor's blocks enable developers to prioritize app logic over coding language syntax [IE17]. The code blocks are versatile to allow building apps for any purpose or goal.

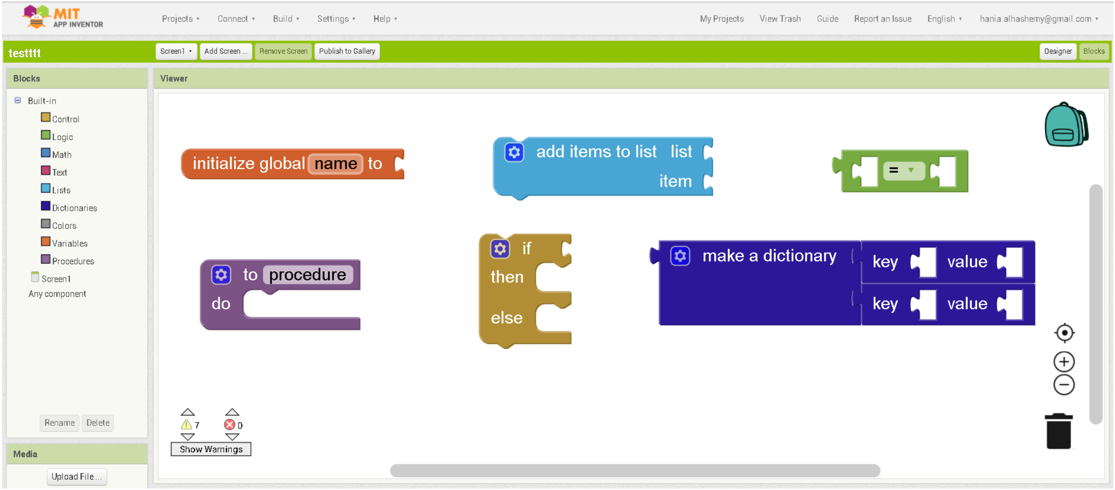


Figure 4.2.: MIT App Inventor blocks editor for adding logic and behavior to App components

App Inventor allows developers to see their creations in real-time as they build them. This incremental development approach encourages users to test their apps as they construct them. When a new component is added to the designer or new functionality is created through the code blocks, these changes are automatically reflected in the connected device or emulator in the mobile app called "Companion" used for testing. This immediate feedback enhances the development process and serves as a valuable tool for testing the app in-progress. Once the app is complete, it can be downloaded to the connected device or exported in Android Packaged (APK) format for distribution [IE17].

4.1.2. Architecture & Technology Stack

App Inventor's block-based programming language is built on Google Blockly [Fra+13]. Blockly is an open-source library enabling developers to build their own block-based programming language. It is integrated into App Inventor web view through its Javascript generator and provides Android and iOS versions [PFM17].

In the section above, we already described the web view that contains two editors: The components designer editor and the blocks' editor. We also illustrated the companion, the app on the physical device to test App Inventor's projects. In addition, App Inventor has two other vital subsystems: The App Inventor server and the build server. Figure 4.3 shows the interaction between all four subsystems.

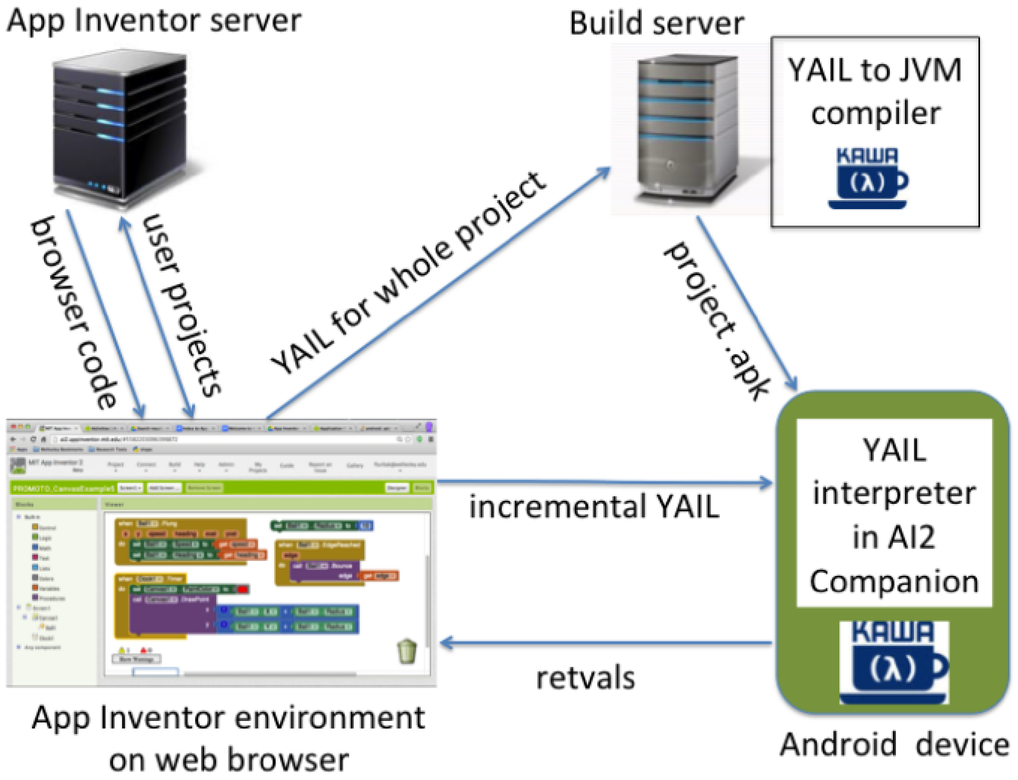


Figure 4.3.: Overview of the interactions between the different App Inventor subsystems [Sch+14].

The App Inventor server hosts cloud-based storage to store the files of users' projects. The user projects are in a zipped archive file of a format called App Inventor Android (AIA). These files contain the app's UI components in JavaScript Object Notation (JSON) format and the code blocks in Extensible Markup Language (XML) format, in addition to other project files in Scheme (SCM) and Young Android Intermediate Language (YAIL). YAIL is an intermediate language between blocks language and Java APIs for Android, created to support learners' code for Android.

The build server compiles the user's project into an APK file to be suitable for distribution (incl. Google Play App Store). To create an APK file, the project is converted entirely to YAIL. The build server then compiles YAIL code to JVM bytecode to create the APK files.

4.1.3. Existing Features

This section describes existing App Inventor components that support introducing data science to App Inventor. The components cover two phases of the data science lifecycle: Data Collection and Data Visualization. App Inventor provides multiple components to store data: CloudDB, DataFile, File, Spreadsheet, and TinyDB. Learners can upload assets to their projects with a size limit of 5 MB, and they can store public datasets in their apps using any of the above-mentioned storage components. These storage options can also be used for real-time data logging.

Another way to collect data is through the app itself, either through app user input or through the Internet of Things (IoT) component [MPS19]. The IoT component in App Inventor facilitates a link between a mobile phone application and an affordable Bluetooth sensor. This connection enables real-time data gathering from the sensor, resulting in a time series dataset. In addition, App Inventor leverages the mobile phone's built-in sensors and provides sensor components like accelerometer, location, thermometer, and light sensors. These sensor components do not require additional hardware and can log data in real-time to any used storage type inside the app.

Lastly, App Inventor enables learners to visualize any data stored in the app using the chart component. The chart component uses MPAndroidChart [2]. MPAndroidChart is an open-source Android library for creating interactive and customizable charts within Android applications. It provides various chart types, including line charts, bar charts, pie charts, area charts, and more. The library allows developers to visualize data by integrating dynamic charts into their Android apps. MPAndroidChart offers various customization options for colors, labels, data points, and animations. The chart component requires a "ChartData2D" component, which is the data to be visualized on the chart. "ChartData2D" elements can also dynamically consume data generated by other components, whether they represent local sensors, remote web services, or real-time distributed databases.

To conclude, App Inventor has the infrastructure and the potential to allow learners to work with data, especially for data collection, data storage, and data visualization. However, it is missing tools for data cleanup, data exploration, and predictions.

[2] `https://github.com/PhilJay/MPAndroidChart`

4.2. Proposed System

In this section, we propose new components to enhance App Inventor, allowing learners to experience implementing a complete data science lifecycle to solve impactful real-world problems through mobile app development. Our thorough evaluation of the existing App Inventor system in the section above and the related tools in chapter 3 leads to the proposition of these components, essential for implementing our envisioned data science educational approach. The introduced components also encompass the fundamental aspects of data literacy and data thinking, extensively addressed in the preceding chapter 2.

4.2.1. Requirements

This section presents the results of requirements engineering, initiated by learners' needed tools to develop data-driven apps with real-world impact. We divide the requirements into functional and non-functional requirements.

Functional Requirements

Functional requirements define what the proposed system should offer and how the user can interact with the newly provided components [BD09]. Our proposed functional requirements provide components to allow learners to preprocess data through cleanup and to extract predictions from the preprocessed data. For data cleanup and exploration, we present the "AnomalyDetection" component. For analysis and predictions, we propose the "Regression" component. In the following, we will elicit the functional requirements for each component.

Anomaly Detection

The Anomaly Detection component offers anomaly detection algorithms to automatically identify anomalies in a dataset incorporated inside an App Inventor project. In this context, anomalies refer to observations that deviate significantly from a dataset's expected or normative behavior [PP07]. The following are the functional requirements for the anomaly detection component:

FR1A **Detect Anomalies**: Learners should be able to integrate anomaly detection algorithms in their App Inventor apps to enable their apps to automatically detect anomalies in an arbitrary dataset.

FR2A **Customize Anomaly Detection Algorithms**: Learners should be able to interact with the anomaly detection algorithms and customize them based on their dataset and domain knowledge. For example, they should be able to set a different "normal region" or threshold that includes every possible expected behavior within their specific dataset.

FR3A **Visualize Anomalies**: Learners should be able to visually display the detected anomalies inside their apps. They should have the option to interact visually with data points and highlight anomalies using distinct colors. They should be able to visually explore the data, directly observing patterns, trends, and irregularities on their mobile devices.

FR4A **Remove Anomalies**: Learners should be able to make data-driven decisions on removing detected anomalies that do not align with the expected behavior of their data. Within the apps they create, learners should have the choice to either remove all detected anomalies collectively or select specific anomalies to retain while discarding others.

Regression

For simplicity, we will focus on linear regression in the context of this thesis. The Linear Regression component allows learners to add a line of best fit to showcase trends in the data. By fitting a straight line to the data points, they can perceive the direction and magnitude of changes, helping them predict future outcomes [Wei05]. This component empowers learners to identify the relationship between variables and make predictions based on the linear trend. The following are the functional requirements for the linear regression component:

FR1B **Calculate Linear Regression**: Learners should be able to calculate the values of the line that fits their data points and integrate this analysis in their App Inventor apps. This includes additional statistical information, like the line slope, the line intercept with the x-axis and y-axis, and the correlation coefficient to quantify the degree of linear association between variables. This information enables learners to assess the strength and significance of the linear fit and assists them in making informed decisions.

FR2B **Visualize Prediction Results**: Learners should be able to display the line of best fit inside their App Inventor apps. The component should allow them to customize the line visualization, e.g., change its color, modify the labels, change the line length (interpolate it to intercept with the axis), and interact with the visualization. The line visualization should change if the dataset changed, e.g., an anomaly was removed, reflecting that change statistically.

Non-Functional Requirements

This section enumerates the non-functional requirements, particularly the quality requirements, which we categorize using the URPS model described in [BD09].

Usability

NFR1 **Ease of Use**: Learners should easily understand how to use the data science components. The components and code blocks should have detailed descriptions that are easily understandable for children. The components' description should be accessed via one step, e.g., by hovering over the component. The code blocks should be abstract and hide most implementation details from learners. In addition, the code blocks should be intuitive for children; for example, the array index for the data science components starts at 1 instead of 0, unlike most programming languages. This decision is based on the fact that most children learn to start counting from 1 in school.

NFR2 **Efficiency**: Learners should be able to easily find the data science components inside the App Inventor components designer and blocks editor. Accessing the components should not take more than one step. The number of blocks required to achieve the above functional requirements should be as small as possible. New code blocks should offer unique functionality not available in any other existing code block.

Reliability

NFR3 **System Errors Prevention**: The code blocks should tolerate many of the learner's wrong inputs, e.g., wrong data types, to decrease the intimidation of learning data science as a beginner. The system should show the learners the mistakes in their logic and suggest ways to correct them. It should provide them with tools to test and debug their code blocks. The code blocks should not snap together like puzzle pieces if they do not fit together and would cause errors otherwise.

NFR4 **Data Loss Prevention**: No data should be lost because of the learner's inexperience. Recovering previous states of the App Inventor projects and saving checkpoints that the learners can revert to should be possible.

Performance

NFR5 **Response Time**: The system should display the data visualization on the mobile app screen in less than 1s. Changes to the data should be reflected in the visualization in real-time

Supportability

NFR6 **Extensibility**: The data science drawer containing the anomaly detection and regression component should be designed in a way that makes it easily extendible with additional data science components. On a more granular level, the anomaly detection and regression components should also be easily extendible with additional algorithms, e.g., K-means clustering and isolation forest code blocks for anomaly detection, and quadratic regression for the regression component.

NFR7 **Maintainability**: The new data science components should be integrated with the current storage, sensors, and chart components. It should not break any of the existing functionality or duplicate it.

4.2.2. Blocks UI

This section describes the UI of the proposed system extending App Inventor with data science capabilities. Figure 4.4 shows the components designer editor of App Inventor's web view. In the left panel, we introduce a data science drawer, see 4.4 (1), beside the data storage, sensors, and chart drawers so users using these related drawers can easily find the proposed data science drawer.

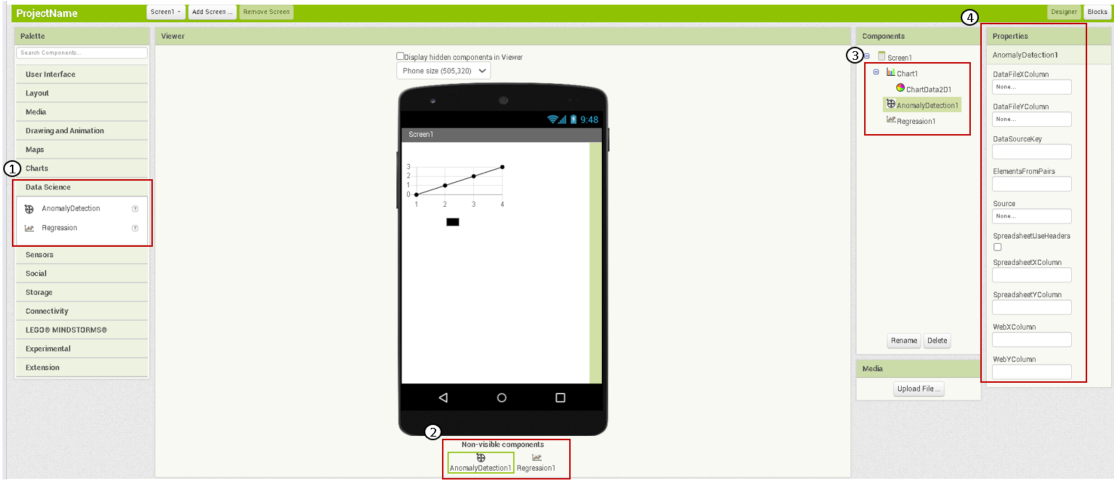
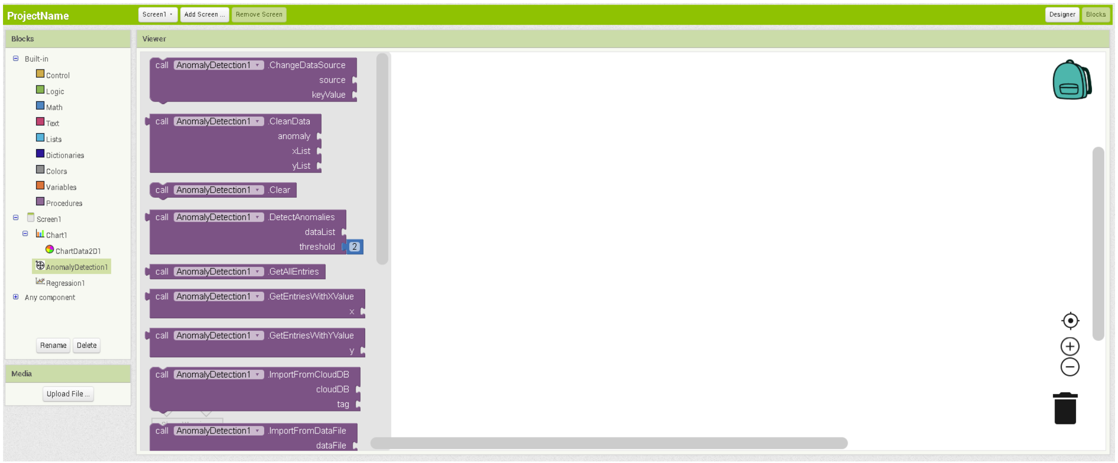


Figure 4.4.: App Inventor components designer editor showcasing the new data science components

Inside the data science drawer are the “AnomalyDetection” and “Regression” components presented in the section above. These components are non-visible components, they are not displayed on the mobile screen. They only appear at the bottom below the mobile simulation view in the middle of the screen, see 4.4 (2). This is because these components preprocess, manipulate, or analyze data in the background. The results of these components can be visualized using other components, the visible chart component, for example.

The usage of the data science components is fully independent and not bound by the chart component. This is presented in the system through the components’ hierarchy on the right side of the page, see 4.4 (3). In addition, learners can modify the components’ properties through the right-side panel, see 4.4 (4). Learners can add a data source as a property for the data science components. This could be an uploaded file containing the dataset or a spreadsheet.



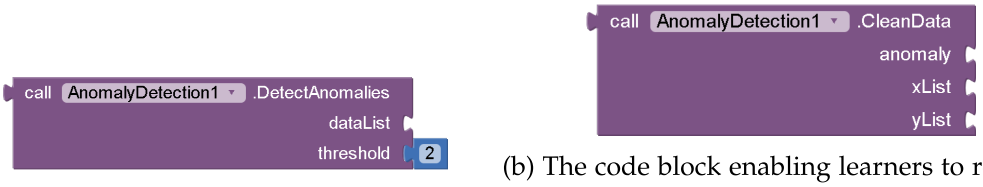
```
call AnomalyDetection1 .ChangeDataSource
                                  source
                                keyValue
call AnomalyDetection1 .CleanData
                            anomaly
                              xList
                              yList
call AnomalyDetection1 .Clear
call AnomalyDetection1 .DetectAnomalies
                               dataList
                              threshold 2
call AnomalyDetection1 .GetAllEntries
call AnomalyDetection1 .GetEntriesWithXValue
                                           x
call AnomalyDetection1 .GetEntriesWithYValue
                                           y
call AnomalyDetection1 .ImportFromCloudDB
                                 cloudDB
                                     tag
call AnomalyDetection1 .ImportFromDataFile
                                 dataFile
```

Figure 4.5.: App Inventor blocks editor showcasing the new data science components

As for the blocks editor, there are no significant changes in the UI. The left panel shows the components’ hierarchy underneath the main blocks resembling the basic computational concepts. On-click on one of the components in the hierarchy (“AnomalyDetection” in this example, see 4.5), another panel opens, the viewer panel, showing all the available code blocks for the clicked component. Learners can drag and drop any block to the white canvas to start building the logic of their mobile apps.

Anomaly Detection

The main added code blocks for the “AnomalyDetection” component are the block to detect anomalies and the block to clean the dataset and remove the anomalies. Figure



(a) The code block enabling learners to detect anomalies in their dataset integrated into their App Inventor mobile app (FR1A, FR2A)

(b) The code block enabling learners to remove anomalies of choice from their dataset integrated into their App Inventor mobile app (FR4A)

Figure 4.6.: The most important code blocks of "AnomalyDetection" components

4.6a shows the block that integrates an anomaly detection algorithm, namely the z-score anomaly detection algorithm.

The z-score measures how many standard deviations a data point is away from the mean of the dataset. Data points with z-scores that deviate significantly from the mean are flagged as potential anomalies. This method is particularly effective when the data follows a normal distribution. By setting a threshold for the z-score, anomalies can be detected and further analyzed for their significance and impact on the dataset [Kor+19].

Per default, the threshold is set to be "2". This default value is just meant to give learners an example of what could be an accepted input for the variable "threshold" of the "DetectAnomalies" block. Still, they should edit this value to a threshold of their choice. The choice of an appropriate threshold for anomaly detection depends on the specifics of the dataset and the desired level of sensitivity in detecting anomalies.

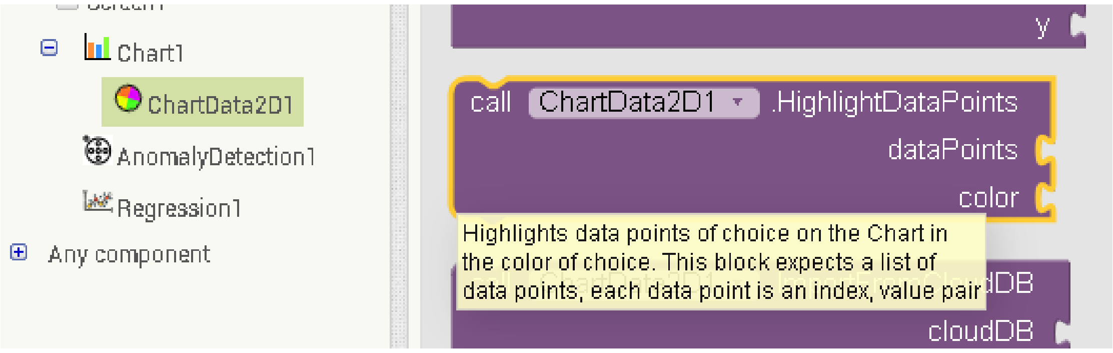
It's important to note that the choice of threshold should be guided by domain knowledge and the data context. Setting the threshold too low could result in a high number of false positives (normal data points being labeled as anomalies). Setting it too high might lead to missing actual anomalies. Experimentation and validation with the specific dataset are crucial to determine the most suitable threshold for accurate anomaly detection. Besides the threshold, this block also expects a list of the data points to be examined. In total, this single code block reduced 22 lines of code.

Figure 4.6b shows the block that cleans the dataset by removing selected anomalies. This block expects the data of the x-axis and y-axis as lists and a list of selected anomalies. Each anomaly in the list is a pair of the anomaly index (starting from 1) and the corresponding value. Based on the dataset and their domain expertise, the learners are responsible for deciding which anomalies to remove and which to keep. Not all anomalies are erroneous; some may reveal valuable insights and require additional examination.

Lastly, if learners wish to visually distinguish anomalies from the rest of the data points, they can do so by creating a chart (e.g., scatter plot) to visualize the entire dataset and highlight the anomalies with a different color using the code block from “ChartData2D” component seen in figure 4.7.

Figure 4.7 also shows a description box to explain the code block to learners. All code blocks have their corresponding description that is visible on hover. This block expects a list of data points to be highlighted, as well as the highlight color of choice.

It is important to stress that learners do not need to visualize anomalies through a chart. The anomaly detection component is independent and can be integrated into the App Inventor mobile apps in other ways. It can be combined with any other App Inventor capability; for example, the notifier can be used to inform the app user about the anomalies.



Highlights data points of choice on the Chart in the color of choice. This block expects a list of data points, each data point is an index, value pair

Figure 4.7.: The code block enabling learners to visualize anomalies (FR3A)

Regression

Figure 4.8 shows three design iterations of the code blocks to calculate linear regression values, also called “line of best fit”.

In figure 4.8 (1) we had a code block for each value separately: the y-intercept, the slope, the correlation coefficient, and the predictions. Each code block expected x-axis and y-axis values in lists as input. This approach cluttered the block editor with too many blocks.

Therefore, we decided to have just one block for the “line of best fit” values calculation, as seen in 4.8 (2). Learners provide the block with a label of the value they need to retrieve. This approach was error-prone: the block returned nothing if learners misspelled the name of the value.

To address the problems above, we changed the text label to a drop-down menu, as seen in 4.8 (3). Learners can choose from the drop-down menu which value they need to retrieve for their “line of best fit”.

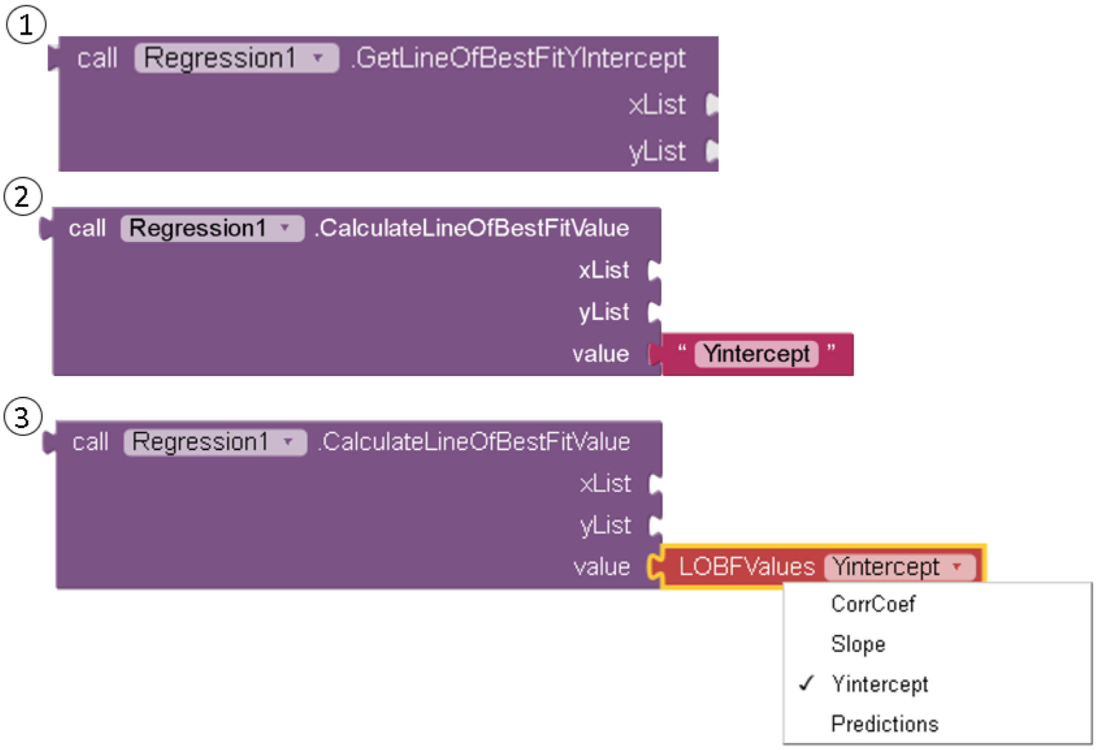

1
call Regression1 .GetLineOfBestFitYIntercept
xList
yList
2
call Regression1 .CalculateLineOfBestFitValue
xList
yList
value
" Yintercept "
3
call Regression1 .CalculateLineOfBestFitValue
xList
yList
value
LOBFValues Yintercept
CorrCoef
Slope
✓ Yintercept
Predictions

Figure 4.8.: UI design iterations for the code block responsible for calculating the values of the line of best fit (FR1B)

The last code block, shown in figure 6.3, is part of the "chartData2D" component, bound to a chart to draw the "line of best fit" on a chart. This code block only expects the x-axis and y-axis values in a list. It calculates the "line of best fit" prediction values in the background and visualizes it on the chart

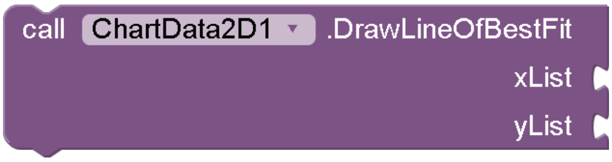


Figure 4.9.: The code block visualizing the line of best fit on the chart (FR2B)

4.2.3. Software Architecture

This section presents how we modified the existing system architecture to integrate the data science components. Figure 4.10 shows the layers of the current system: Data source and data visualization, integrated into App Inventor chart component. We extend this architecture with an additional layer: The data processing layer. The learner decides whether to visualize their data directly or process it first. After data processing, they can choose whether to proceed and visualize their processed data.

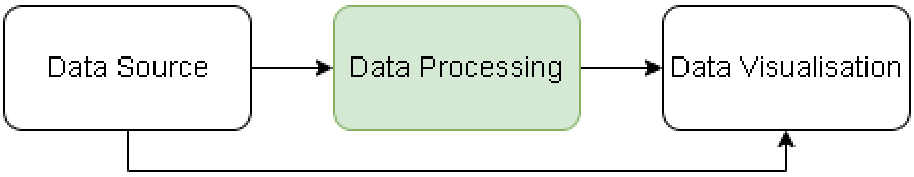


Figure 4.10.: Overview of the main layers of the data science feature in App Inventor

To abstract the data processing from the visualization, we need to modify the current classes that combined the data source and the data visualization, not accounting for an additional processing layer. Figure 4.11 shows the changes in the classes and the associations between classes to achieve this abstraction.

The software architecture follows the Model-View-Controller (MVC) architecture. The MVC architecture divides applications into three key subsystems: the model, responsible for data and logic; the View, handling user interface presentation; and the Controller, managing interactions between the Model and View. This approach enhances modularity, scalability, and maintenance by separating concerns and clarifying the roles of each subsystem [Dea09].

"ChartDataModel" is a single data source within a chart. It has a reference to instances of "ChartData", the "IDataSet", and the "ChartView", and maintains a list of entries. The first two are types provided by the MPAndroidChart library, and the latter is a view class defined by App Inventor.

To abstract the data from the visualization, we introduce a new superclass for "ChartDataModel": "DataModel" (presented in green in Figure 4.11). We move the logic of managing entries up into the newly added superclass. "DataModel" expects a generic entry, enabling data models to be used for anything, not necessarily bound to charts. To achieve this generalization, we changed some classes to be abstract. This allows subclasses to provide behaviors for their specialization of the entry.

"ChartData2D" class is a 2-dimensional implementation of the abstract "ChartDataBase" class, which holds a reference to its corresponding "ChartDataModel". In addition, it handles the connection to components that can serve data sources to the

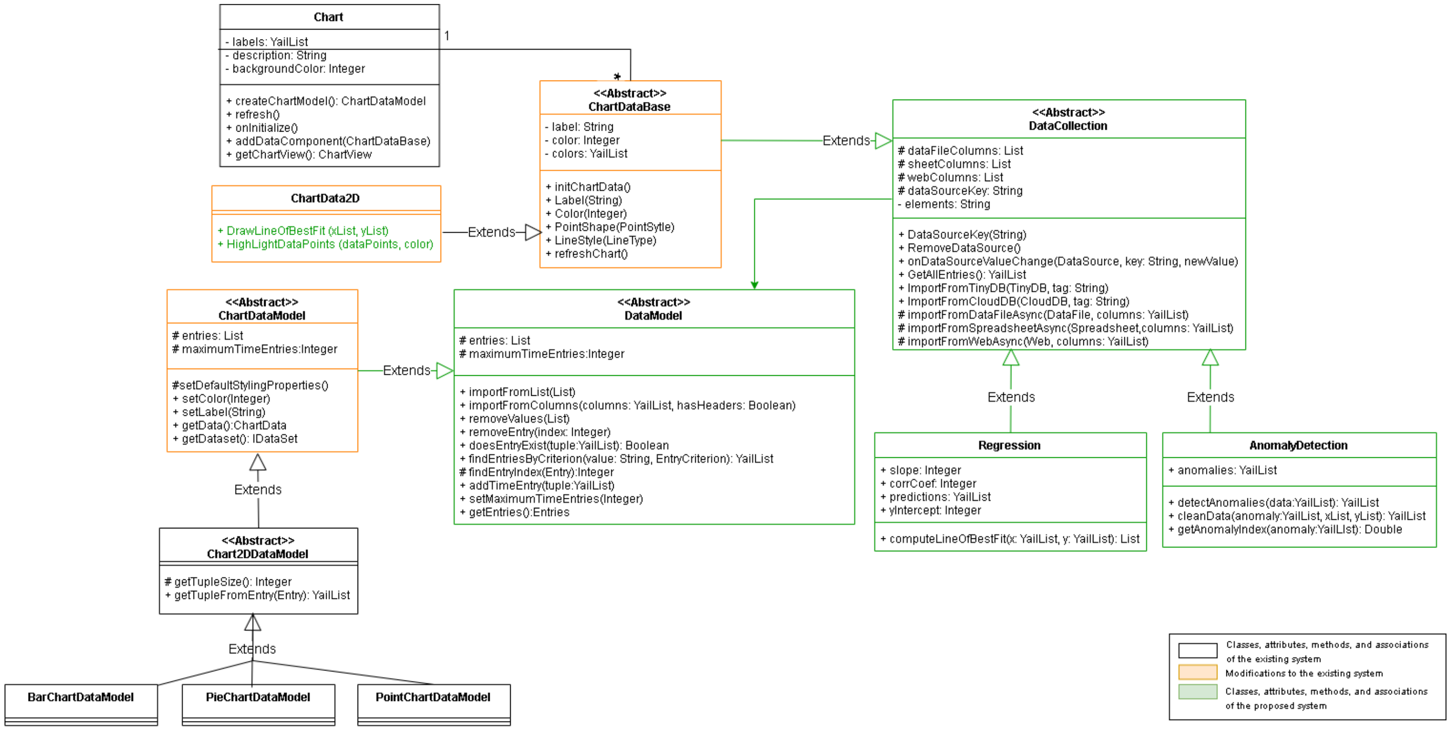


Chart
- labels: YailList
- description: String
- backgroundColor: Integer
+ createChartModel(): ChartDataModel
+ refresh()
+ onInitialize()
+ addDataComponent(ChartDataBase)
+ getChartView(): ChartView
1
*
<<Abstract>>
ChartDataBase
- label: String
- color: Integer
- colors: YailList
+ initChartData()
+ Label(String)
+ Color(Integer)
+ PointShape(PointSytle)
+ LineStyle(LineType)
+ refreshChart()
ChartData2D
+ DrawLineOfBestFit (xList, yList)
+ HighLightDataPoints (dataPoints, color)
Extends
Extends
<<Abstract>>
DataCollection
# dataFileColumns: List
# sheetColumns: List
# webColumns: List
# dataSourceKey: String
- elements: String
+ DataSourceKey(String)
+ RemoveDataSource()
+ onDataSourceValueChange(DataSource, key: String, newValue)
+ GetAllEntries(): YailList
+ ImportFromTinyDB(TinyDB, tag: String)
+ ImportFromCloudDB(CloudDB, tag: String)
# importFromDataFileAsync(DataFile, columns: YailList)
# importFromSpreadsheetAsync(Spreadsheet,columns: YailList)
# importFromWebAsync(Web, columns: YailList)
<<Abstract>>
ChartDataModel
# entries: List
# maximumTimeEntries:Integer
#setDefaultStylingProperties()
+ setColor(Integer)
+ setLabel(String)
+ getData():ChartData
+ getDataset(): IDataSet
Extends
<<Abstract>>
DataModel
# entries: List
# maximumTimeEntries:Integer
+ importFromList(List)
+ importFromColumns(columns: YailList, hasHeaders: Boolean)
+ removeValues(List)
+ removeEntry(index: Integer)
+ doesEntryExist(tuple:YailList): Boolean
+ findEntriesByCriterion(value: String, EntryCriterion): YailList
# findEntryIndex(Entry):Integer
+ addTimeEntry(tuple:YailList)
+ setMaximumTimeEntries(Integer)
+ getEntries():Entries
Extends
Extends
Regression
+ slope: Integer
+ corrCoef: Integer
+ predictions: YailList
+ yIntercept: Integer
+ computeLineOfBestFit(x: YailList, y: YailList): List
AnomalyDetection
+ anomalies: YailList
+ detectAnomalies(data:YailList): YailList
+ cleanData(anomaly:YailList, xList, yList): YailList
+ getAnomalyIndex(anomaly:YailList): Double
Extends
<<Abstract>>
Chart2DDataModel
# getTupleSize(): Integer
+ getTupleFromEntry(Entry): YailList
Extends
BarChartDataModel
PieChartDataModel
PointChartDataModel
Classes, attributes, methods, and associations of the existing system
Modifications to the existing system
Classes, attributes, methods, and associations of the proposed system

Figure 4.11.: Class Diagram showing the detailed software architecture of the proposed system and how it modifies parts of the existing system

"ChartDataModel". The primary purpose of the "ChartDataModel" is to ensure that any data is ordered and added synchronously.

For the proposed system, we introduce the "DataCollection" class as a new superclass of "ChartDataBase". We move there the logic of managing data source connections. The "DataCollection" class serves as the base class for other components taking data but are not necessarily attached to the Chart ecosystem, like the subclasses "Regression" and "AnomalyDetection" of the data processing layer.

4.2.4. Design Goals

The non-functional requirements identified in section 4.2.1 resemble most of the design goals for the proposed system. In this section, we will discuss some design goals trade-offs and justify our chosen final design for the data science components in App Inventor.

Functionality vs. Usability: The more functionality we offer in a component, the more complex it will become. Here, complexity means, the component has a lot of properties to manage, the code blocks require multiple inputs in a specific data type, etc. While this complexity makes the component more powerful and offers users more flexibility and features to integrate into their App Inventor apps, it makes it harder for learners to understand the components.

As our primary target audience is beginner high schoolers, who have never implemented a data science lifecycle before, it is more important to keep the component simple, offering just the minimum functionality that allows learners to grasp and apply the essential data science concepts.

Efficiency vs. Portability: We deliberated the utilization of java ML libraries, like Apache Spark [3], for implementing the data science code blocks. Although these libraries provide a range of ML algorithms and would alleviate the need for developing algorithms from scratch, their substantial size could potentially compromise the overall robustness of App Inventor's projects.

Consequently, we opted to develop the algorithms internally. In the future, we could reconsider using a subset of available ML libraries offering only the needed algorithms when extending App Inventor's data science drawer with more complex algorithms. However, this approach appears excessive at the current stage due to the associated size of such libraries.

Cost vs. Robustness: Datasets could allocate a large storage size. The maximum limit of App Inventor's projects' assets is 5 MB. App Inventor utilizes cloud storage as well. Extending the storage size of App Inventor's projects would enable learners to explore big data. However, it will come with an additional cost for storage.

[3] `https://spark.apache.org/mllib/`

Rapid development vs. Functionality: There is a limited period, five months, to implement and integrate the proposed system in App Inventor. Hence, we focus on the top-priority functionality. Within the scope of this thesis, some functionality with lower priority may not be fully covered.

4.3. Example Apps Produced Using the Proposed System

In this section, we showcase different apps of students from UROP. All the apps use the data science components and serve as examples showing how learners can address real-world problems using real-world data and make an impact with their created apps. It is important to note that all the apps below use real public datasets.

We present the different apps as scenarios. The scenario is a way to describe app features and bridge the conceptual gap between end-users and developers. This bridge is established by describing the scenarios in natural application domain language understandable for both the app end-user and the developer [BD09]. Each scenario will describe a sequence of interactions between a specific end-user and an app.

Diabetes Logbook App by Jennet Zamanova

Jennet describes her app as the following: "With Diabetes LogBook, you can log your glucose levels and share them with your friends and your doctor". The following scenario describes the interactions between Liz, a diabetes patient, and her diabetes logbook app displayed in figure 4.12.

Liz opens her app after having a meal with her friends (see 4.12a). She chooses the option "after meal" to log her blood glucose level after her meal (see 4.12b). She can see her past logged data on her mobile's screen, visualized on the lower graph (see 4.12c). The line of best fit in green shows her that her glucose level has continuously decreased. She was indeed on a strict diet the past few days.

The upper graph shows her the average glucose level of patients of the same age and same condition, so she can see how her level compares to other patients in her support group. She notices that some data points are below 50 mg/dL, which does not make sense. Someone must have mistyped it. The app detected it as an anomaly and highlighted it on the graph in red. She has the option to remove this wrong data by clicking on the "remove anomalies" button at the top of the screen.

Liz inserts her glucose level of today (see 4.12d). Next, she inserts the time when she measured her blood glucose level (see 4.12e). Today was her birthday; she ate out with friends and had a lot of cake for her birthday. The app recognizes her high glucose level, warns her (see 4.12f), and automatically highlights her latest entry as an anomaly. The app also shows her suggestions from trusted sources on how to self-regulate her glucose level (see 4.12g). Liz follows the suggestions and continues to monitor her glucose level. She can also contact her doctor through the app if her situation becomes critical.

EduCar App by Jacky Chen

Jacky describes his app as the following: "Educar is an app that helps adolescents understand trends in education and income, as well as explore various careers". The following scenario describes the interactions between Bob, a student in his senior year considering grad school, and his EduCar app displayed in figure 4.13.

Bob opens his app to see the latest trends in education costs in the US and explore different career options to decide on his next step. The main view (see 4.13a) shows a graph of the average tuition fees in dollars over the past years. The line of best fit shows a continuous increase. Bob realizes that doing a master's today would probably be cheaper than doing it in the future after working first. At the bottom of the screen, he can see more details about the graph, like the correlation coefficient.

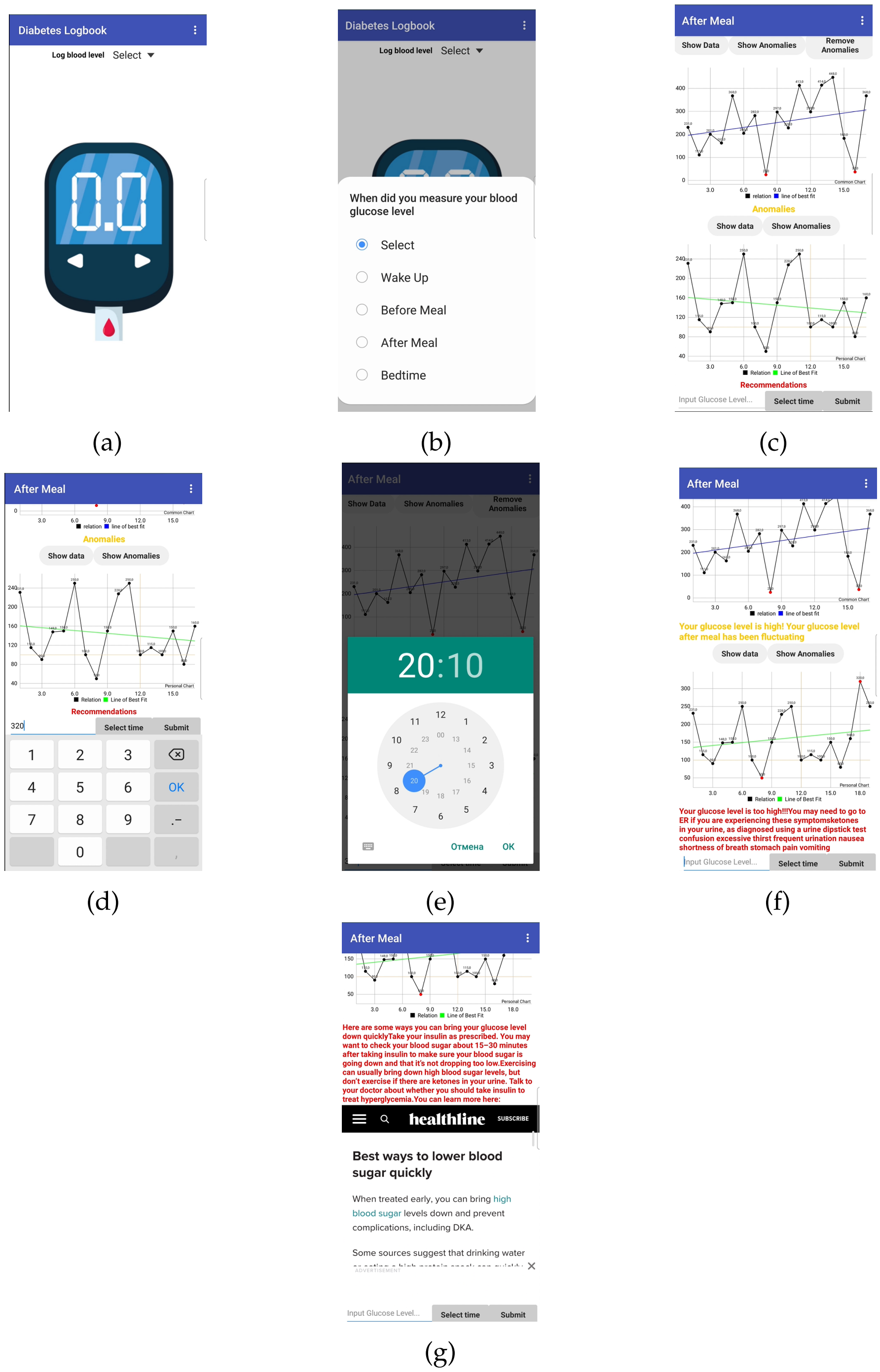
Bob is still unsure if a master's will bring him more advantages compared to just going into industry without a master's degree. He decides to explore the different careers through the explore page in his app (see 4.13b). Bob wants to earn as much money as possible, so he clicks on the $200,000 option to see which careers pay that much. A web view opens, showing him the different jobs paying that salary.

Next, he clicks on the "show data" button to see if a master's degree would add to that base salary (see 4.13c). He sees that there is more than $30,000 increase between an associate degree and a master's degree. He continues exploring the right degree for him by taking a quiz (see 4.13d). The app predicts the right career for him based on his answers to the quiz questions (see 4.13e). Bob almost made up his mind about his next career step. Now, he wants to connect with experienced people in the same career field to validate his choice. He connects with people through the network option inside the app (see 4.13f).

Disney Data Diary App by Arianna Scott

Arianna describes her app as the following: "Disney Data Diary is an app for kids to maximize their fun at Disney World through the power of data science!". The following scenario describes the interactions between Ana, a teenager planning her first trip to Disney, and her Disney Data Diary app displayed in figure 4.14.

Ana logs in to her app by inserting her private password (see 4.14a). After unlocking the app, she sees the main menu with the options: Wait times, recommendations, todo



When did you measure your blood glucose level

● Select

○ Wake Up

○ Before Meal

○ After Meal

○ Bedtime

Your glucose level is high! Your glucose level after meal has been fluctuating

Your glucose level is too high!!!You may need to go to ER if you are experiencing these symptomsketones in your urine, as diagnosed using a urine dipstick test confusion excessive thirst frequent urination nausea shortness of breath stomach pain vomiting

Here are some ways you can bring your glucose level down quicklyTake your insulin as prescribed. You may want to check your blood sugar about 15–30 minutes after taking insulin to make sure your blood sugar is going down and that it's not dropping too low.Exercising can usually bring down high blood sugar levels, but don't exercise if there are ketones in your urine. Talk to your doctor about whether you should take insulin to treat hyperglycemia.You can learn more here:

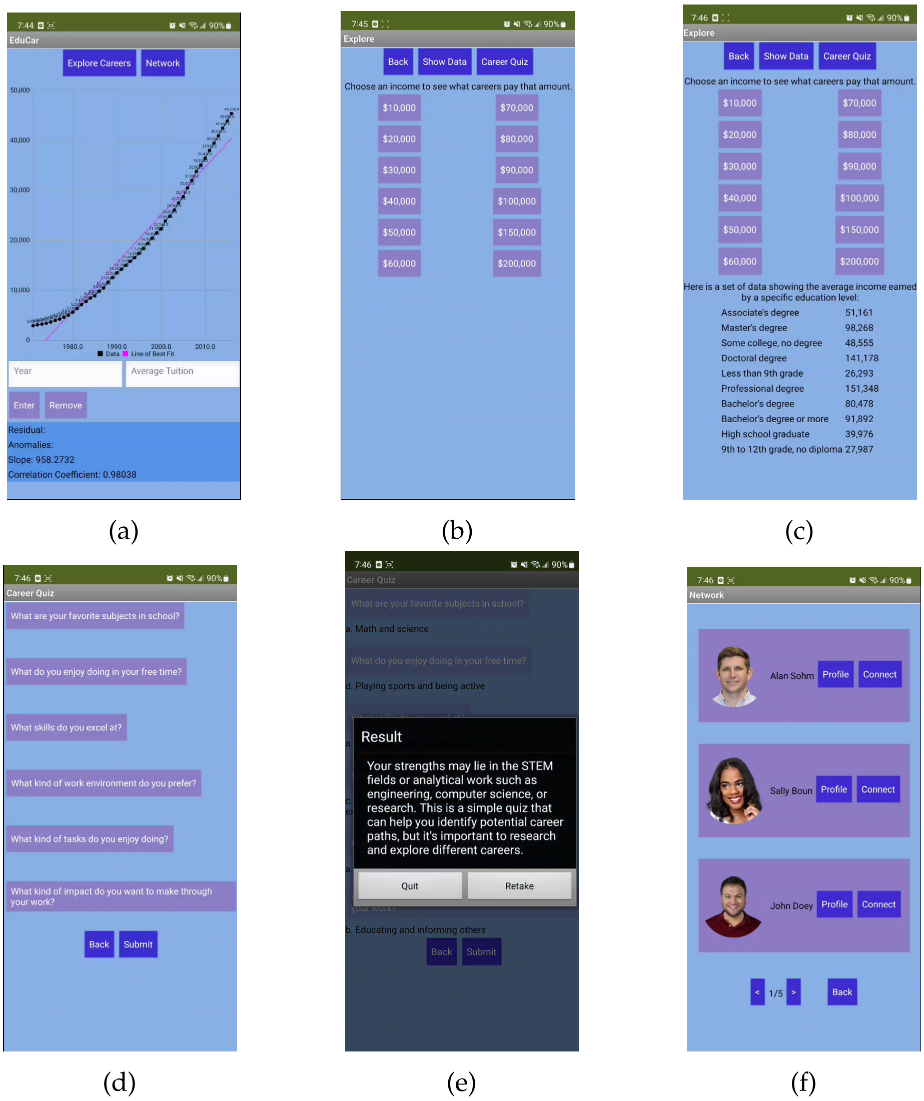
Best ways to lower blood sugar quickly

When treated early, you can bring high blood sugar levels down and prevent complications, including DKA.

Some sources suggest that drinking water

(a) (b) (c) (d) (e) (f) (g)

Figure 4.12.: The different views of the Diabetes Logbook app, showcasing the app's features



Residual:
Anomalies:
Slope: 958.2732
Correlation Coefficient: 0.98038

(a)

Choose an income to see what careers pay that amount.

(b)

Choose an income to see what careers pay that amount.

Here is a set of data showing the average income earned by a specific education level:

Associate's degree	51,161
Master's degree	98,268
Some college, no degree	48,555
Doctoral degree	141,178
Less than 9th grade	26,293
Professional degree	151,348
Bachelor's degree	80,478
Bachelor's degree or more	91,892
High school graduate	39,976
9th to 12th grade, no diploma	27,987

(c)

What are your favorite subjects in school?

What do you enjoy doing in your free time?

What skills do you excel at?

What kind of work environment do you prefer?

What kind of tasks do you enjoy doing?

What kind of impact do you want to make through your work?

(d)

What are your favorite subjects in school?

a. Math and science

What do you enjoy doing in your free time?

d. Playing sports and being active

Result

Your strengths may lie in the STEM fields or analytical work such as engineering, computer science, or research. This is a simple quiz that can help you identify potential career paths, but it's important to research and explore different careers.

b. Educating and informing others

(e)

(f)

Figure 4.13.: The different views of the EduCar app, showcasing the app's features

list, and photos (see 4.14b). She clicks on wait times to see the best time to go to Disney and which rides to take, depending on the ride queue data (see 4.14c).

From the rides list, Ana chooses to look at the data from the "Splash Mountain" ride. The upper graph shows her the average daily wait times for a year. She notices many spikes in the data, so she activates the anomaly detection feature by clicking the "highlight anomalies" button. Now, many data points are highlighted in red; these are influencing her prediction results, so she removes them by clicking on the button "clean data".

After she cleaned the data, the app predicted the following in yellow at the top of the screen: "The best time to go is in February. The average wait time is 7.5 minutes" (see 4.14d). It makes sense that this ride is less crowded in February (during winter) as there is a chance to get wet from the water splash, and maybe fewer people would want to get wet during cold weather. Thus, the app seems to be giving her reasonable predictions.

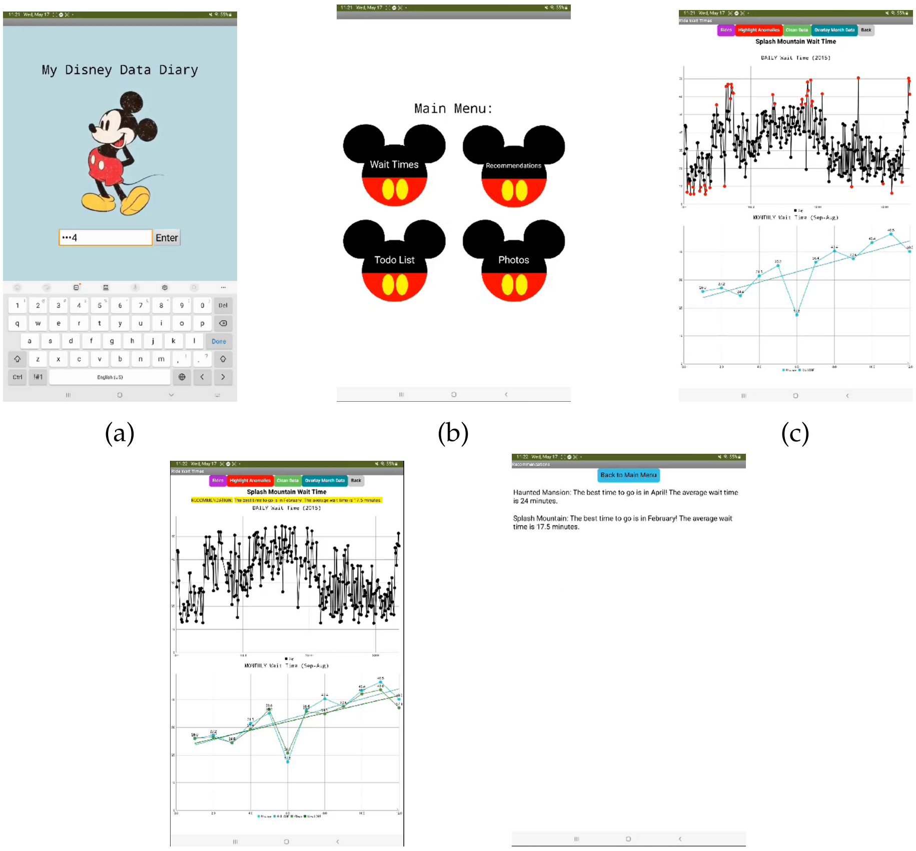
The lower graph shows her the average wait times per month so she can validate again the prediction results. She decides to overlay the prediction results before the data clean up and after the data clean up to see the differences. Through the line of best fit displayed on the lower graph, she can see that this ride will get more crowded in the summer due to the increasing slope.

Ana wants to discover more recommendations to help her better plan her Disney visit, so she navigates to the recommendation page (see 4.14e). On this page, the app lists all the rides of Disney and when it's best to visit them. Based on the recommendations, she postponed her visit until after summer so the park would be less crowded. She writes these notes in her todo list inside the app, beside the other things she needs to organize for her visit. She can't wait to take photos at the park when she visits using the photo feature inside the app.

My Tennis Coach App by Ava Muffoletto

Ava describes her app as the following: "My Tennis Coach uses the power of data science and anomaly detection to perform as your virtual tennis coach, giving you feedback through the screen!". The following scenario describes the interactions between Luc, a beginner tennis player, and his "My Tennis Coach" app displayed in figure 4.15.

Luc opens his app after recording his tennis session today. The recording is saved in the app as a film. He can either watch the film or analyze it (see 4.15a). He decides to look at the analysis to learn from his mistakes. He first sets the coaching style to "tough love", and then chooses to analyze the hit speed from the menu (see 4.15b). The data of the speed per hit extracted from the film is now displayed in his app (see 4.15c).



RECOMMENDATION: The best time to go is in February. The average wait time is 17.5 minutes.

Haunted Mansion: The best time to go is in April! The average wait time is 24 minutes.

Splash Mountain: The best time to go is in February! The average wait time is 17.5 minutes.

(a) (b) (c)

(d) (e)

Figure 4.14.: The different views of the Disney Data Diary app, showcasing the app's features

Luc clicks the “coach me” button to get more insights from the data (see 4.15d). He now sees anomalies highlighted in red and some description for a chosen hit indicating whether it was a successful hit.

Moneyball App by Matthew Quispe

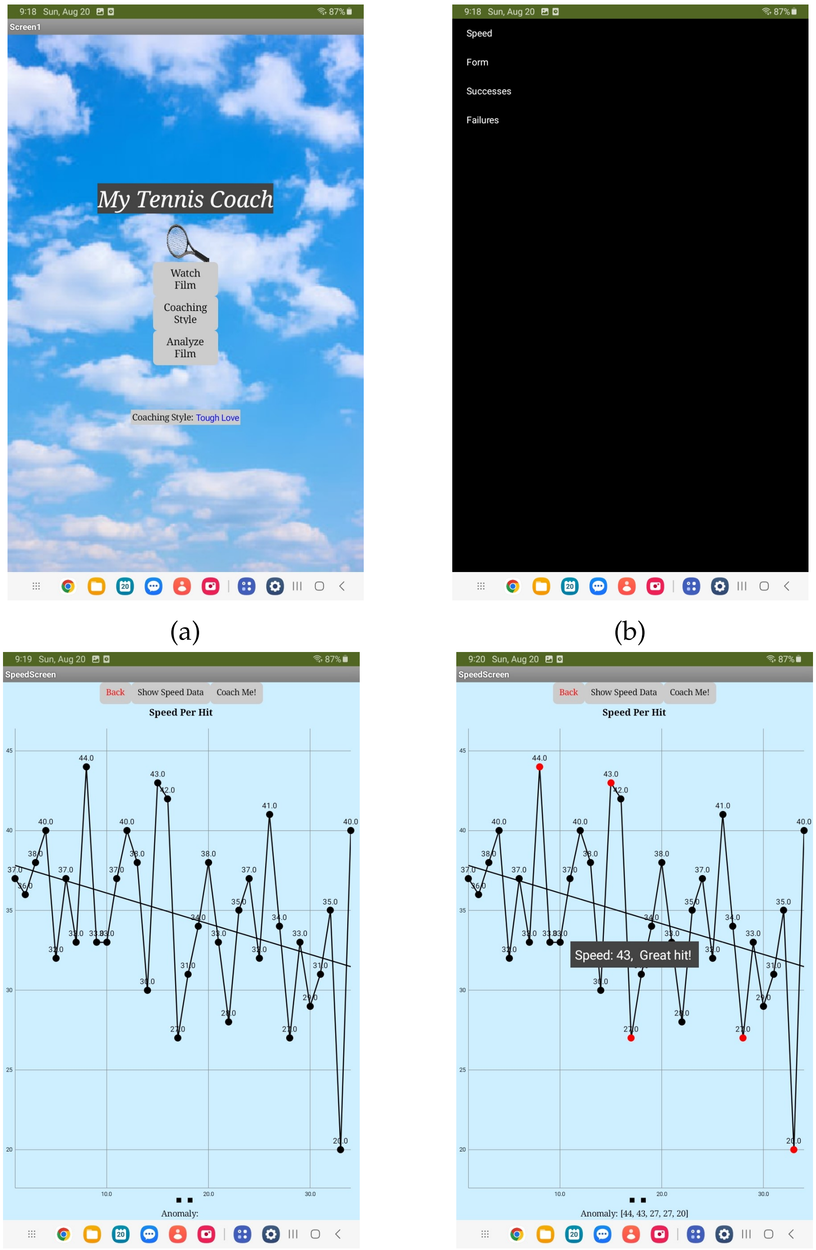
Matthew describes his app as the following: “With Moneyball, kids can look at baseball stats and understand the relationship between team payroll and performance”. The following scenario describes the interactions between Max, a big fan of Major League Baseball (MLB), and his Moneyball app displayed in figure 4.16.

Max uses his app to get more insights about MLB. He opens his app and chooses between analyzing the seasons’ data or the teams’ data (see 4.16a). He chooses the teams’ data and picks season 2019 (see 4.16b). The app shows him a graph (see 4.16c) displaying the players’ payroll in millions of dollars on the y-axis and the corresponding team win rate on the x-axis.

The line of best fit shows that the higher the payroll, the more the team wins. The app analyses the correlation and displays the analysis results underneath the graph: “In 2019, each extra $50 million in salaries earned you 7.8891 more wins in a 162-game season. The correlation is weak, with a coefficient of 0.4255”. The app also highlights the anomalies in red on the graph.

Max clicks on the “remove outliers” button at the top of the screen (see 4.16d). In this context, anomaly means, the teams vastly overperformed and underperformed their salary. Now, the app shows the following analysis after removing the anomalies from the data: “In 2019, each extra $50 million in salaries earned you 7.36889 more wins in a 162-game season. The correlation is weak, with a coefficient of 0.43398”.

There is still a weak correlation, so before making any conclusions, Max decides to look at the other seasons to know if he can confidently say there is a relation between a team’s wins and how much the team is compensated for their hard work.



Coaching Style: Tough Love

Anomaly:

Speed: 43, Great hit!

Anomaly: [44, 43, 27, 27, 20]

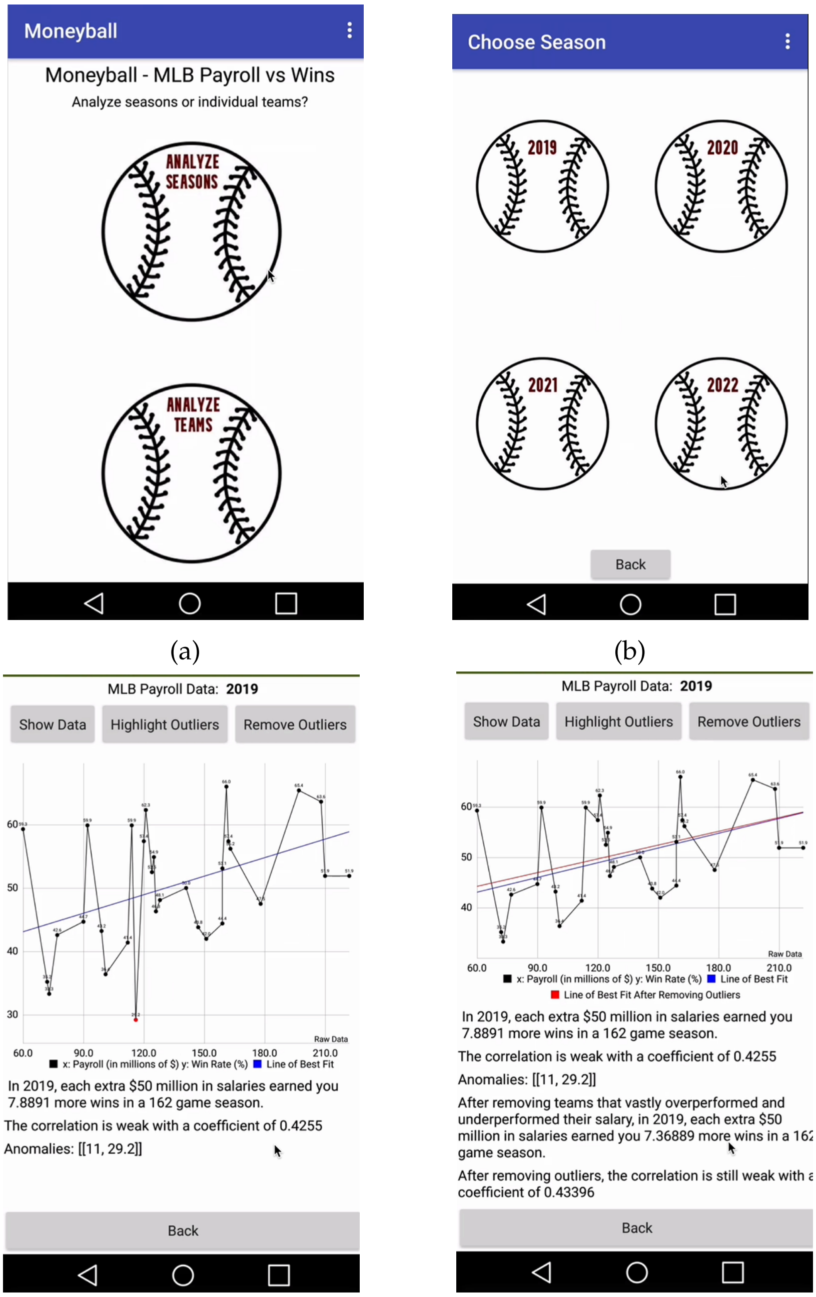
(a)

(b)

(c)

(d)

Figure 4.15.: The different views of the My Tennis Coach app, showcasing the app's features



(a)

(b)

In 2019, each extra $50 million in salaries earned you 7.8891 more wins in a 162 game season.

The correlation is weak with a coefficient of 0.4255

Anomalies: [[11, 29.2]]

(c)

In 2019, each extra $50 million in salaries earned you 7.8891 more wins in a 162 game season.

The correlation is weak with a coefficient of 0.4255

Anomalies: [[11, 29.2]]

After removing teams that vastly overperformed and underperformed their salary, in 2019, each extra $50 million in salaries earned you 7.36889 more wins in a 162 game season.

After removing outliers, the correlation is still weak with a coefficient of 0.43396

(d)

Figure 4.16.: The different views of the Moneyball app, showcasing the app's features

5. Data Action Educational Framework

This chapter outlines the Data Action Framework, which builds upon the Computational Action Framework by Tissenbaum, Sheldon, and Abelson described in section 2.2.1, but with a specific focus on data-related computations. The framework enables widespread adoption and easy access to data science education, thereby catering to a diverse demographic of learners. The framework allows teachers to embed a long-term data science curriculum in their institution at no cost. This particularly supports high schools with limited resources and promotes the democratization of data science education.

The framework empowers young people to learn data science by working on their chosen data-driven projects of significant relevance to them, directly influencing their lives and communities. This allows students to develop their identity as data scientists who could translate their skills into impactful projects promoting their digital empowerment.

The framework targets high school students aged 14 to 18 who already have familiarity with App Inventor's interface and capabilities. They can experimentally learn data science concepts while building meaningful apps that solve real-world problems without delving into underlying code syntax.

We first illustrate the concepts taught through the framework for K-12 students. Then, we describe how students can practice and apply these concepts to solve real-world problems. We present an example of using the framework with students from UROP. Lastly, we describe how the framework changes the students' perspective of their world after learning and applying its concepts.

5.1. Concepts

Influenced by the computational action framework and the educational approaches discussed in related work chapter 3, we created the data action framework covering the following topics:

1. **Defining a real-world Problem:** Students learn to identify impactful real-world problems that are personally relevant to them. In addition, they learn how to define and abstract a problem to attempt to solve it using data science.

2. **Self-acquiring knowledge of problem's domain:** For their defined problem, students research and acquire knowledge about the specific domain where their data-driven solution is applied. They also need to gather more data related to that domain.

3. **Working with real-world data critically:** Students work with authentic, real-world data, which is characterized by being "messy" and might contain errors. Students learn to apply different data science algorithms to process the data. They understand that it is required to question the meaning behind the data and examine it critically.

4. **Understanding the impact of data:** Through their developed data-driven solutions, students see the impact of data and the meaningful insights they can extract from data. Students recognize the impact of data and understand the importance of using data ethically.

5.2. Practices

We leverage the data science components of the educational open-source platform, App Inventor, described in the proposed system section 4.2 to apply the above concepts. Students practice the above concepts through experimental project-based learning by creating personalized apps to solve real-world problems and applying the data science lifecycle explained in section 2.1.3. The resulting apps are presented in the previous chapter 4.3.

We apply the framework with seven computer science first-year students from MIT's UROP. These students did not study any data science related subjects. However, they have experience with App Inventor. During the first week, we asked UROP students to identify real-world challenges that held personal significance to them and that they wished to solve. The challenges identified included diabetes, student debt, theme park wait times, lack of motivation for sports, and climate change.

They were then tasked with researching and finding relevant data about their identified problems. While some students relied on online data sets, others combined public data with their own experiences, and one student solely used personal and friends' data. Weekly presentations and feedback sessions were conducted, with us acting as the moderator.

In the third week, we introduced the UROP students to the data science components in App Inventor, along with data storage and visualization capabilities. The goal was to build mobile apps to help solve their identified challenges. They were initially instructed to create a simple one-screen app displaying a chart visualizing their data.

They had to determine the appropriate chart type for their specific dataset and desired goal.

The following week, they were expected to demo their apps. Students faced common considerations, such as incorporating data into their apps, selecting relevant parameters (feature selection), and choosing appropriate chart types. Students noticed unusual spikes in each other's app graphs during the demos. While the student using personal data could explain the anomalies and elicit the reason behind the spikes, others using anonymous public datasets were puzzled.

This led to a discussion on the presence of errors in real-world data and the need for data cleaning. They learned about the different types of anomalies and that, depending on the domain, these anomalies could hold significant information relevant to solving their previously identified challenges.

To further progress, students were asked to process their data and conduct detailed domain research to explain the anomalies. They utilized the data science components in App Inventor, focusing on anomaly detection and linear regression. App Inventor code blocks for these algorithms are illustrated in section 4.2.2.

The underlying code of these algorithms is intentionally concealed to prevent overwhelming beginners. This said, the blocks still require some input from the learners. Learners can acquire the input knowledge through "trial and error", for example, seeing how changing the threshold value would influence the tolerance of the anomaly detection algorithm.

Students started identifying patterns and extracting insights from their data. They iteratively acquired more in-depth knowledge of their datasets and the domains of their identified challenges. In addition, they continued building more features in their apps that used the data insights as input to provide a solution to the challenge they identified in the first week.

Lastly, after several iterations of their apps, the students presented their final apps to a larger audience, practicing their presentation skills and seeing the interest of a broader audience in their work. After their final presentation, in a reflection session, students learned that they experienced a simplified data science lifecycle throughout their projects — collecting data, data cleanup, data visualization, and prediction.

They started learning about the different terminology of the data science concepts they applied while building their apps. Furthermore, we introduced them to more complex machine learning concepts beyond anomaly detection and linear regression that they can further experiment with and compare their results with their current app.

5.3. Perspectives

In this section, we describe how the framework changes the students' perspective in terms of (1) how they view the world and (2) how they view themselves. The students first learned to identify real-world challenges that held personal significance to them and that they were eager to tackle. This process allowed them to establish a meaningful connection between their learning and real-world issues.

In their pursuit of solutions, the students conducted thorough research to locate relevant datasets. They gained proficiency in collecting and analyzing data from various sources, including public datasets and personal experiences. Furthermore, they acquired the skills to develop mobile apps that collect, visualize, and extract insights from data using App Inventor.

Most importantly, the students gained an understanding of anomalies and the importance of data cleaning. They recognized that real-world data could contain errors, underscoring the significance of data cleaning practices. They further realized that anomalies could hold crucial information pertinent to solving their identified challenges. By observing the direct impact of data cleaning on their predictive outcomes and "line of best fit", they gleaned insights into optimizing and improving the results of ML models.

By identifying patterns and extracting knowledge from their data, the students gained valuable insights and deepened their understanding of the domains related to their specific challenges. Engaging with their chosen real-world challenges was a powerful motivator for the students, igniting their enthusiasm for acquiring new skills. The tangible outcome of having a product in their hands as a result of their work amplified the relevance of their efforts and instilled a strong sense of accomplishment.

Lastly, witnessing the real-world impact of their apps in solving significant challenges within their society profoundly changed the students' perspectives and empowered them to embrace the role of data scientists.

After experiencing the data action framework, students gain awareness of data science's influence on their daily lives. They know how insights are extracted from data and will be more careful when giving out their personal data. They also realized that they must question any data insights presented to them, as these could be highly manipulated through errors in the data. They grasped the impact of data, seeing the results of their projects, and would aim to make ethical data-driven products. In the next chapter, we present a research study that evaluates the effects of the data action framework on changing the students' perspective.

6. Research Study

In this chapter, we describe the research study we conducted to evaluate the effectiveness of our data action educational framework and the new data science code blocks in App Inventor. MIT Committee on the Use of Humans as Experimental Subjects (COUHES), which serves as MIT's Institute Review Board (IRB), approved the research study.

The research study consists of a three-hour workshop, teaching high school students data science by applying a short version of our data action framework, focusing on the two learning objectives: (1) Working with real-world data critically and (2) Understanding the impact of data, described in chapter 5.

We first list the research questions we examined in the context of the research study. Then, we describe the study participants and the criteria to accept them in the study. Following that, we illustrate the workshop outline and content. To conclude, we describe the data collected during the research study and present the analysis results of the collected data.

6.1. Research Questions

The research study aims to understand how students see data science and their roles as data scientists. It will also explore how educational activities can change their thoughts about data science. By examining the evolution of students' views before and after educational activities, the study will provide valuable insights into the effectiveness of the data action framework in shaping their understanding of the field.

The study predicts that students who participate in the workshop will develop skills in data processing, visualization, and interpretation, and will be able to use these skills to create solutions for real-world problems. The study also expects that participation in the program will foster the development of data thinking skills, such as recognizing patterns and relationships in complex data sets, applying critical thinking to evaluate data quality, and effectively communicating data insights.

The research questions we investigated were: (1) Is the data action framework effective in enabling students to recognize the impact of data science and utilize it to build impactful solutions for real-world problems? (digital empowerment) and (2)

Is the data action framework effective in empowering students to be data scientists (self-efficacy)?

6.2. Participants

For the research study, it is required that the participants are high school students between the ages of 14 and 18 who have prior experience with App Inventor, so they are familiar with its UI. To recruit the participants, we contacted the computer science teacher, Lisa Miller of Medford Vocational Technical High School, requesting to apply our data action framework in her class. The teacher has been teaching her 10th-grade students computer science using App Inventor for the complete school year and offered us a slot to apply our data action framework in a one-day workshop towards the end of the school year 2023.

Due to logistical issues outside our control, we could not go to the school. However, the teacher and her class visited our facility to participate in the workshop. Her class and students that participated in our workshop consisted of 14 students; four identified as female students and ten as male students. Apart from two male students aged 16, the rest of the participants are 17 years old. All the participants have comparable skill levels, as they are in the same class and receive the same education throughout the school year. Although all the participants have prior programming knowledge and used App Inventor before, none have previous data science experience.

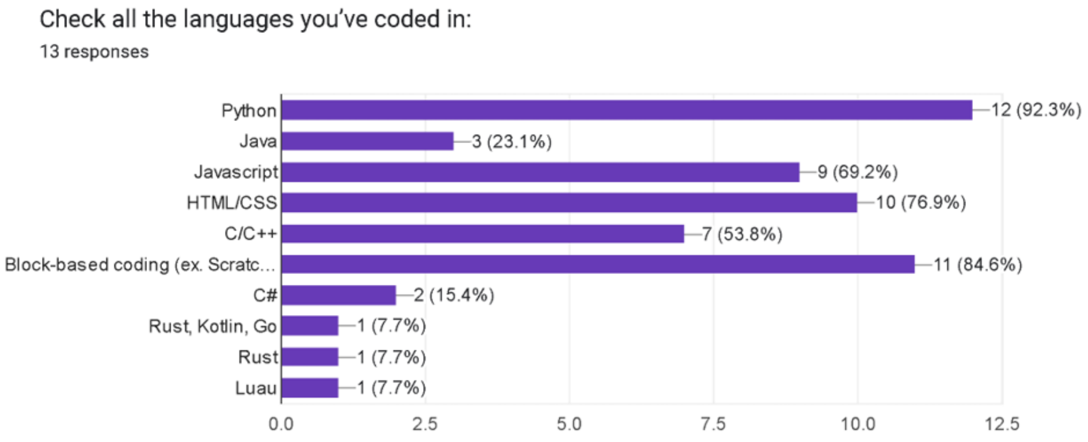


Figure 6.1.: Results from the pre-survey about participants' past experience with various programming languages

We asked the participants to check all the programming languages they used before during the pre-survey to understand their skill level. Figure 6.1 shows the results of this question. We were surprised that almost all participants (apart from one) had used Python before. Python is widely used in data science. Although their teacher confirmed that they are all familiar with App Inventor, only 11 out of 13 participants have actually "coded in" a block-based environment like App Inventor, according to the participants' responses.

6.3. Workshop Design

Table 6.1 displays the rough schedule of the 3-hour workshop we conducted for the research study. In the following, we will describe the workshop activities in more detail and explain the reasoning behind the choice of these activities and their sequence.

Time	Activity
9:00 – 9:15 EST	Fill pre-Survey
9:15 – 9:30 EST	Data science unplugged activity
9:30 – 9:35 EST	Introduction of a real-world problem to be solved
9:35 – 10:00 EST	App Inventor setup and example project walkthrough
10:00 – 10:15 EST	Break
10:15 – 10:20 EST	Find a correlation
10:20 – 10:35 EST	Implement "line of best fit"
10:35 – 10:40 EST	Insights from "line of best fit"
10:40 – 10:55 EST	Implement "line of best fit" values calculation
10:55 – 11:10 EST	Break
11:10 – 11:15 EST	Detect anomalies
11:15 – 11:30 EST	Implement anomaly detection algorithm
11:30 – 11:40 EST	Removing anomalies
11:40 – 11:45 EST	Discussing results
11:45 – 11:50 EST	Reflection
11:50 – 12:00 EST	Fill post-survey

Table 6.1.: Outline of the workshop designed for the research study participants

We started the workshop by greeting the students and giving them time to settle in and get comfortable with the workshop environment. Initially, they were tasked to scan the QR code displayed on the workshop slides and fill in a pre-workshop survey.

After all students completed the pre-survey, we kicked off the workshop with a data science unplugged activity [LSR19]: the guessing game. We introduced the game rules to the students:

1. Find a partner to guess your favorite food.
2. Reveal three pieces of information about you that help your partner identify your favorite food.
3. Make it challenging for your partner. You get a point for every wrong guess they make.
4. With every wrong guess, you have to reveal one additional piece of information about you that would help them guess your favorite food.
5. Your partner has a maximum of 3 trials to guess your favorite food.

Besides breaking the ice and activating the students, the purpose of this unplugged activity is also to show the students how they can predict insights based on data their game partner reveals to them. Later, during the workshop, they experienced how the mental model to predict insights during the guessing game is connected to an algorithmic machine-learning model to predict insights for real-world datasets.

The connection is only revealed at the end of the workshop as a reflection. The reason for this is to give the students a chance to experience implementing the machine learning models themselves first and the time to be able to make the connection themselves instead of pre-feeding them this as theoretical knowledge.

After the game, we explained to them that they had solved the problem of guessing their partner's favorite food, but following that, we aim to solve more impactful problems. We introduced them to the United Nations' global goals for inspiration of topics they could work on in the future. We tackled goal 13, climate action, for the workshop and introduced the topic to the students.

After we presented to them the problem we will try to solve in the workshop context, we explained that the next step is to collect data. Due to the time constraint, we did this step before the workshop without the students. We showed them the collected real-world data in an Excel sheet of Spirit Lake. The data included the lake's name, the year the data was recorded, the average temperature during winter in Celsius, and the number of ice days during that year.

We asked the students to give some insights about the data by looking at the Excel sheet and to predict in which year the lake will have no ice days. The students were unable to answer these questions. One student suggested visualizing the data to answer these questions. We agreed with the student and decided collectively to visually explore the data to answer the question about ice days.

We asked the students to download the App Inventor starter AIA file we provided them, which visualizes the lake data in our climate action mobile app. Together with the students, we imported the provided starter file into App Inventor. We then reviewed the existing code blocks that read data from the spreadsheet and visualize them on the app. We tried to connect the companion app with App Inventor to run the climate action app on the students' mobile devices so that they could see the data visualizations on their mobile screens.

Figure 6.2 shows the climate action app 6.2a and its corresponding code blocks 6.2b. This is the starter app the students built on during the workshop and added more functionalities to.

The reason we provide them with some code blocks in the starter file is to avoid the intimidation that comes with starting an app from scratch on a blank canvas and also because of the workshop's limited time duration. The app consists of two graphs: The upper scatter plot visualizes the number of ice days during winter for Spirit lake, and the bottom scatter plot visualizes the average temperature during winter in Celsius for the same lake. The x-axis is the year the data was collected for both graphs.

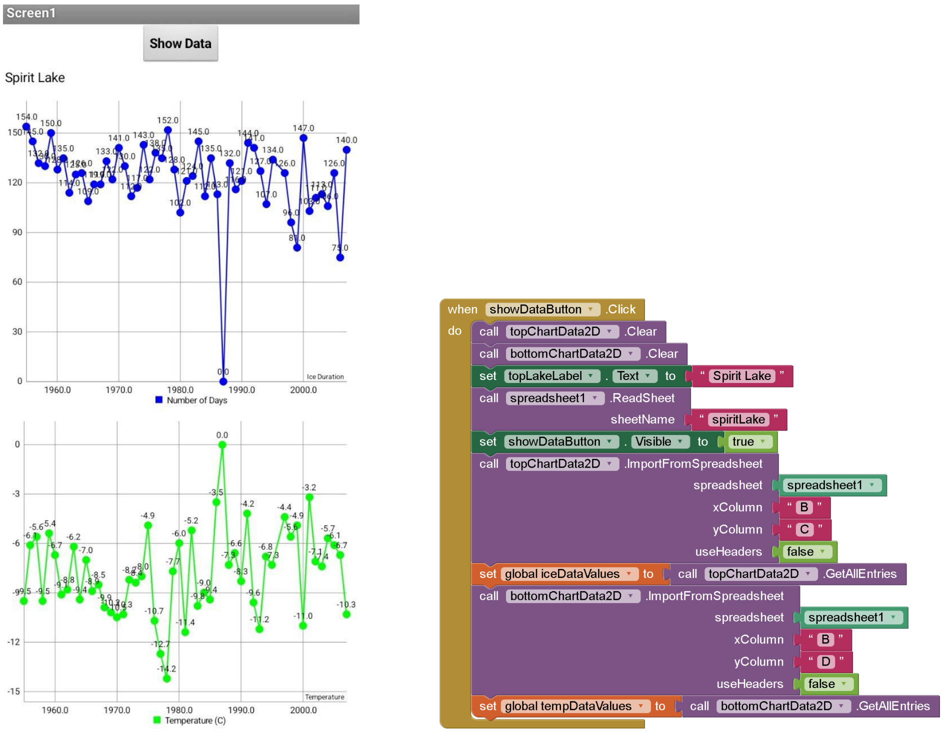
We repeated the same questions and asked the students if, now looking at the visualizations on their mobile devices, they could make any conclusions about the data and identify correlations. This time, the students could tell more about the data compared to just looking at the spreadsheet. They were able to visually identify one anomaly, for example. However, they were still unable to identify a trend in the data.

We showed them multiple example graphs of perfect, strong, weak, and no correlation. Students noticed that for a strong correlation between the x-axis and the y-axis, the data should take the shape of a line. They were able to recall the formula for a line graph from their linear algebra math class: $y = mx + b$, while m is the slope and b is the y-intercept.

After that, the students suggested fitting the lake data in a line graph to examine if there is a correlation between the x-axis and the y-axis. They started implementing that for their mobile apps, and we revealed the linear regression App Inventor blocks they could use to create the line of best fit for their data.

Figure 6.3 shows the app after they added the line of best fit for both graphs 6.3a and the corresponding code blocks they added to achieve that 6.3b. The students could now tell that the ice duration graph has a negative correlation while the temperature graph has a positive correlation: Each year, it gets warmer and the warmer it gets, the fewer ice days we will get.

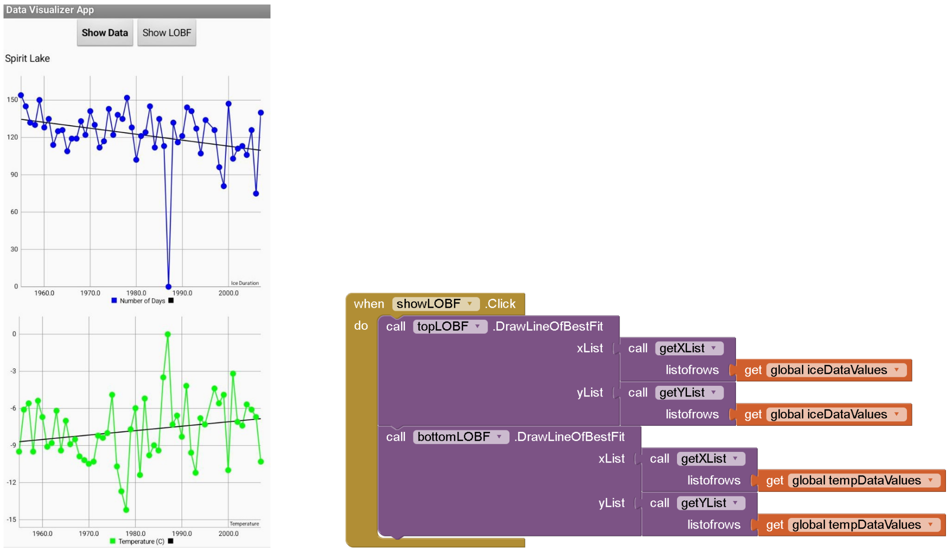
Students were still unable to answer the question: When (in which year) will we have no more ice in the lake during winter? They were also unsure how strong or



```
when showDataButton .Click
do  call topChartData2D .Clear
    call bottomChartData2D .Clear
    set topLakeLabel . Text to " Spirit Lake "
    call spreadsheet1 .ReadSheet
                        sheetName " spiritLake "
    set showDataButton . Visible to true
    call topChartData2D .ImportFromSpreadsheet
                        spreadsheet spreadsheet1
                            xColumn " B "
                            yColumn " C "
                         useHeaders false
    set global iceDataValues to call topChartData2D .GetAllEntries
    call bottomChartData2D .ImportFromSpreadsheet
                        spreadsheet spreadsheet1
                            xColumn " B "
                            yColumn " D "
                         useHeaders false
    set global tempDataValues to call bottomChartData2D .GetAllEntries
```

(a) (b)

Figure 6.2.: The climate action starter app provided to the students at the beginning of the worshop



```
when showLOBF .Click
do  call topLOBF .DrawLineOfBestFit
                        xList   call getXList
                                     listofrows   get global iceDataValues
                        yList   call getYList
                                     listofrows   get global iceDataValues
    call bottomLOBF .DrawLineOfBestFit
                        xList   call getXList
                                     listofrows   get global tempDataValues
                        yList   call getYList
                                     listofrows   get global tempDataValues
```

(a) (b)

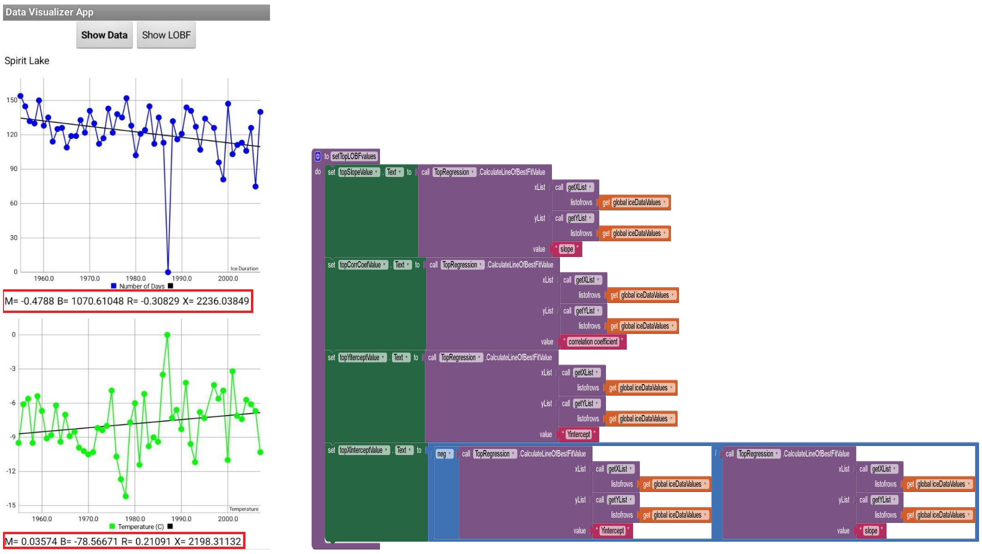
Figure 6.3.: Line of best fit, the first feature implemented by the students for their climate action app during the workshop

weak their identified correlation was. We discussed the line graph formula again, and the students correctly identified that we needed to calculate the x-intercept and the correlation coefficient to answer these questions.

After writing the x-intercept equation with the students on the whiteboard, they implemented it in App Inventor using the previously introduced linear regression code blocks. Figure 6.4 shows their app 6.4a after they included the calculation of these values and the corresponding code blocks that they added for this feature 6.4b.

Based on the calculation results displayed on their apps, students could predict that from the year 2236, lake Spirit will no longer witness any ice days. We asked the students if something could have influenced this result. The students pointed out the anomaly in the year 1987, where there were zero ice days, and the average temperature during winter was zero degrees Celsius. This does not make sense, as water freezes at zero degrees Celsius. Here, students relied on their domain knowledge to identify the erroneous data point. They could visually identify this anomaly and collectively decided to remove this data point as it influenced their linear regression results.

We asked them if they could visually identify other anomalies. Some students pointed out other spikes in the data, but were unsure if these spikes were anomalies that need to be removed. We introduced them to the anomaly detection component in App Inventor and explained to them that it is an algorithm using statistical methods to



```
M= -0.4788 B= 1070.61048 R= -0.30829 X= 2236.03849
M= 0.03574 B= -78.56671 R= 0.21091 X= 2198.31132
```

```
to setTopLOBFvalues
do  set topSlopeValue . Text to  call TopRegression .CalculateLineOfBestFitValue
                                                               xList  call getXList
                                                                          listofrows  get global iceDataValues
                                                               yList  call getYList
                                                                          listofrows  get global iceDataValues
                                                               value  " slope "
    set topCorrCoefValue . Text to  call TopRegression .CalculateLineOfBestFitValue
                                                               xList  call getXList
                                                                          listofrows  get global iceDataValues
                                                               yList  call getYList
                                                                          listofrows  get global iceDataValues
                                                               value  " correlation coefficient "
    set topYiterceptValue . Text to  call TopRegression .CalculateLineOfBestFitValue
                                                               xList  call getXList
                                                                          listofrows  get global iceDataValues
                                                               yList  call getYList
                                                                          listofrows  get global iceDataValues
                                                               value  " Yintercept "
    set topXInterceptValue . Text to  neg  call TopRegression .CalculateLineOfBestFitValue
                                                               xList  call getXList
                                                                          listofrows  get global iceDataValues
                                                               yList  call getYList
                                                                          listofrows  get global iceDataValues
                                                               value  " Yintercept "
                                      /    call TopRegression .CalculateLineOfBestFitValue
                                                               xList  call getXList
                                                                          listofrows  get global iceDataValues
                                                               yList  call getYList
                                                                          listofrows  get global iceDataValues
                                                               value  " slope "
```

(a) (b)

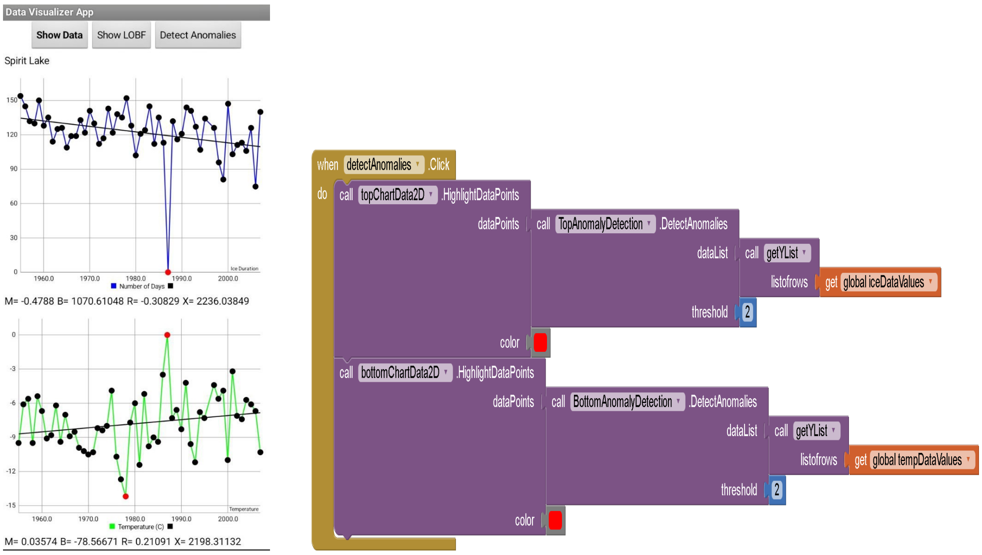
Figure 6.4.: Line of best fit values (highlighted with a red box), the second feature implemented by the students for their climate action app during the workshop

automatically detect anomalies in a dataset.

Students incorporated the anomaly detection code blocks in their app, as seen in figure 6.5. They could now see the anomalies highlighted in red on their mobile screen 6.5a. They experimented with the threshold attribute of the corresponding anomaly detection code blocks, adjusting the sensitivity of the anomaly detector and observing the results 6.5b.

Afterward, we engaged in a discussion to determine which anomalies contain meaningful information and should be retained, and which anomalies are erroneous data points that can disrupt the accuracy of our linear regression results. After the discussion, the students implemented the feature to remove chosen anomalies, as seen in figure 6.6. When clicking the remove anomalies button for the top graph, the erroneous data point is removed from the chart, and the linear regression values and visualization change accordingly (see 6.6a). Based on the new calculation results, after the anomalies' removal, students could predict that from the year 2280, lake Spirit will no longer witness any ice days, giving them an extra 44 years compared to the first results before data cleanup.

We concluded the workshop with a discussion about the impact of data and the influence of data cleanup on our prediction results, pointing out the importance of critically questioning data analysis results. We proceeded with a reflection session



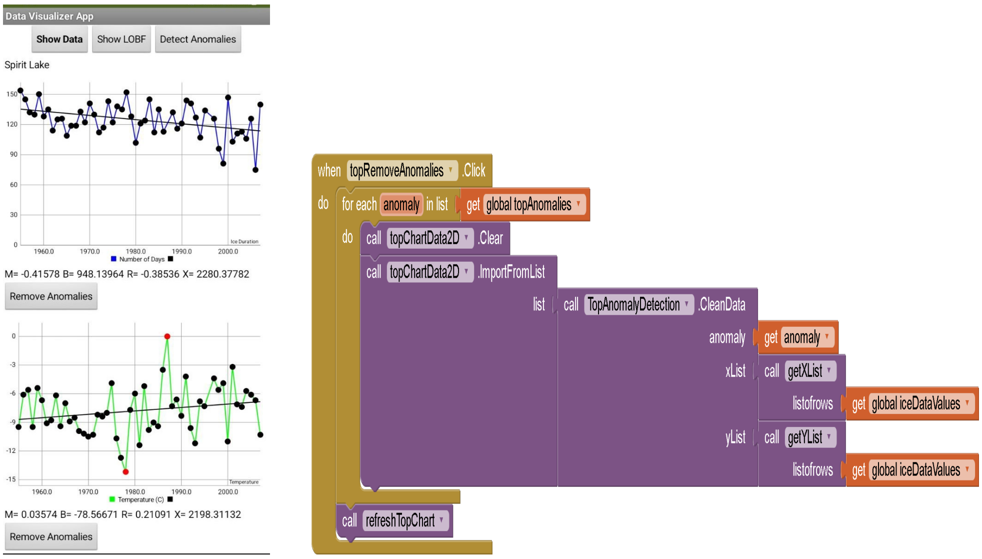
M= -0.4788 B= 1070.61048 R= -0.30829 X= 2236.03849

M= 0.03574 B= -78.56671 R= 0.21091 X= 2198.31132

```
when detectAnomalies .Click
do  call topChartData2D .HighlightDataPoints
        dataPoints  call TopAnomalyDetection .DetectAnomalies
                        dataList  call getYList
                                      listofrows  get global iceDataValues
                        threshold  2
        color
    call bottomChartData2D .HighlightDataPoints
        dataPoints  call BottomAnomalyDetection .DetectAnomalies
                        dataList  call getYList
                                      listofrows  get global tempDataValues
                        threshold  2
        color
```

(a) (b)

Figure 6.5.: Anomaly detection, the third feature implemented by the students for their climate action app during the workshop



M= -0.41578 B= 948.13964 R= -0.38536 X= 2280.37782

M= 0.03574 B= -78.56671 R= 0.21091 X= 2198.31132

```
when topRemoveAnomalies .Click
do  for each anomaly in list  get global topAnomalies
    do  call topChartData2D .Clear
        call topChartData2D .ImportFromList
            list  call TopAnomalyDetection .CleanData
                      anomaly  get anomaly
                      xList  call getXList
                                 listofrows  get global iceDataValues
                      yList  call getYList
                                 listofrows  get global iceDataValues
    call refreshTopChart
```

(a) (b)

Figure 6.6.: Anomaly removal, the fourth and last feature implemented by the students for their climate action app during the workshop

where we introduced the students to the data lifecycle and its terminology.

We explained how we experienced an entire data lifecycle during this workshop, from identifying the climate action problem to extracting insights about the effect of climate change on Spirit lake. We also reflected on the unplugged activity, and students discussed its connection to the data lifecycle. The workshop slides are attached in the appendix (see A) for more details.

6.4. Data Collection

The study anticipates that the workshop's effectiveness will depend on several aspects, such as the quality and relevance of the data science curriculum, the usability of the tools, and the level of engagement and motivation of the students. The research study will use pre- and post-surveys, to test these hypotheses and assess the program's impact on students' data science skills and attitudes. In addition, we recorded the workshop to examine the students' engagement post-workshop.

One participant did not fill out the post-survey due to time constraints, thus their answer was removed from the pre-survey. Besides, all other 13 participants completed the pre- and the post-survey. Table 6.2 presents the questions of the pre- and post-surveys. Participants answered these questions on a Likert scale of 1 (strongly disagree) to 5 (strongly agree), except for question Q11, where they chose from the options: Yes, No, Maybe, or "I don't know".

The post-survey has additional questions to evaluate their workshop experience and the used App Inventor code blocks:

Q15 Would you recommend this class to a classmate or friend? — Answer on the Likert scale 1 (Not at all) 5 (Yes definitely)

Q16 How (if at all) did the class and activities change how you think about data science? — Long answer text

Q17 After this class, what does data science now mean to you? — Long answer text

Q18 What was the easiest part of working with data science components in app inventor? — Long answer text

Q19 What was the hardest part of working with data science components in app inventor? — Long answer text

Q20 How would you change things to improve the experience of using data science when making apps in App Inventor? — Long answer text

Q21 If you had more time, what changes would you make to improve on the last app you made in the workshop? — Long answer text

Q22 Would you use app inventor data science components again in the future? — Choice: Yes, no, maybe, "I don't know"

Question Type	Question Number	Question
Motivation	Q1	How much does math or stats excite you?
Motivation	Q2	How interested would you be in learning how to code?
Motivation	Q3	How interested would you be in a career that involves coding?
Knowledge	Q4	I am confident I know what data science is and can describe it.
Self-efficacy	Q5	I see myself as a data scientist.
Data literacy	Q6	I feel confident working with real-life data.
Data literacy	Q7	I feel confident in visualizing data.
Data literacy	Q8	I feel confident in my ability to extract insights from data (identify patterns, identify anomalies, calculate predictions).
Digital empowerment	Q9	I feel confident in my ability to use data to solve a real-life challenge.
Self-efficacy, Digital empowerment	Q10	I am confident I can design and create my own technology project, not just something someone tells me to create.
Self-efficacy, Digital empowerment	Q11	Given the right tools, could you see yourself making your own interesting app without a teacher's help?
Digital empowerment	Q12	I am confident I can make an impact in my own community or in the world using technology.
Perception of data science	Ql3	I want to include data science in technology projects that I create.
Perception of data science	Q14	I believe using data science would enhance my apps and bring a good beneficial impact to my community or the world.

Table 6.2.: Survey instrument used in the research study.

6.5. Results

This section will present some notable results of the pre- and post-survey answers. We will divide the results into five sections: (1) Motivation: evaluating students' motivation in learning data science, (2) Data Literacy: evaluating students' acquired skills through the workshop, (3) Digital Empowerment & Self-Efficacy: evaluating students' view of themselves and their impact after acquiring data literacy skills, (4) Data Science in App Inventor: evaluating the tool, (5) Perception of Data Science: evaluating the students' view of the impact of data science.

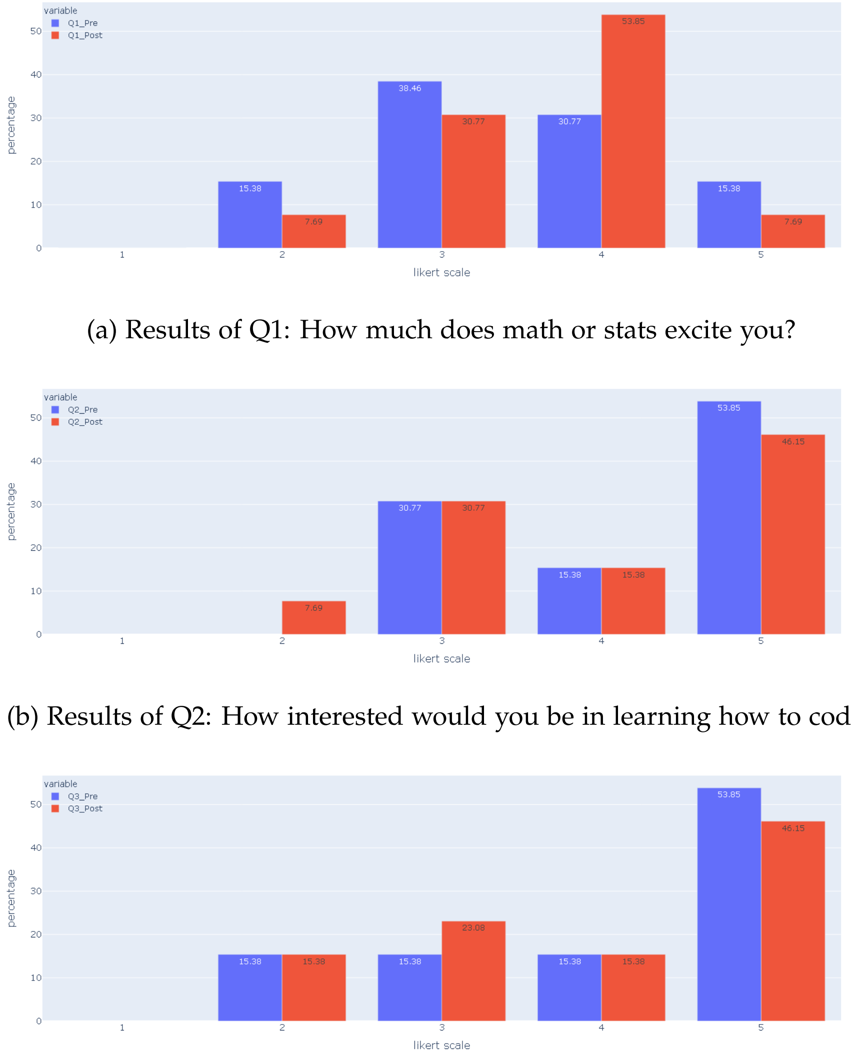
Motivation

> The data action framework increased students' excitement for their mathematics and statistics school subjects.

To assess students' motivation for learning data science, we inquired about their enthusiasm for subjects they are already familiar with that also intersect with data science. In particular, we asked them about their excitement for Mathematics, Statistics, and Coding. In addition, we asked them about their interest in a future tech career. Figure 6.7 shows their pre- and post-survey answers to the three questions about their motivation for learning data science. The pre-survey results are in blue, and the post-survey results are in red. On the x-axis is a Likert scale of 1 (strongly disagree) to 5 (strongly agree), and on the y-axis is the percentage of participants voting for that choice. This axis description applies to all red/blue bar charts in this chapter.

We see an increase in students' excitement for mathematics and statistics (see 6.7a) after the workshop in comparison to their excitement for these subjects before the workshop. On the other hand, their interest in coding stayed the same or slightly decreased after the workshop (see 6.7b). This could have two explanations: 1. They have experience with general-purpose programming languages like Python, and therefore block-based programming did not challenge them or excite them enough, as block-based programming is less powerful than languages like Python; 2. The data science code-blocks are abstract and designed to hide underlying complex code, so students assume they do not need programming knowledge to be data scientists.

Their interest in a career that involves coding also slightly decreased after the workshop (see 6.7c). However, their interest in a data scientist role increased by around 7.7 percent after the workshop (see 6.8).

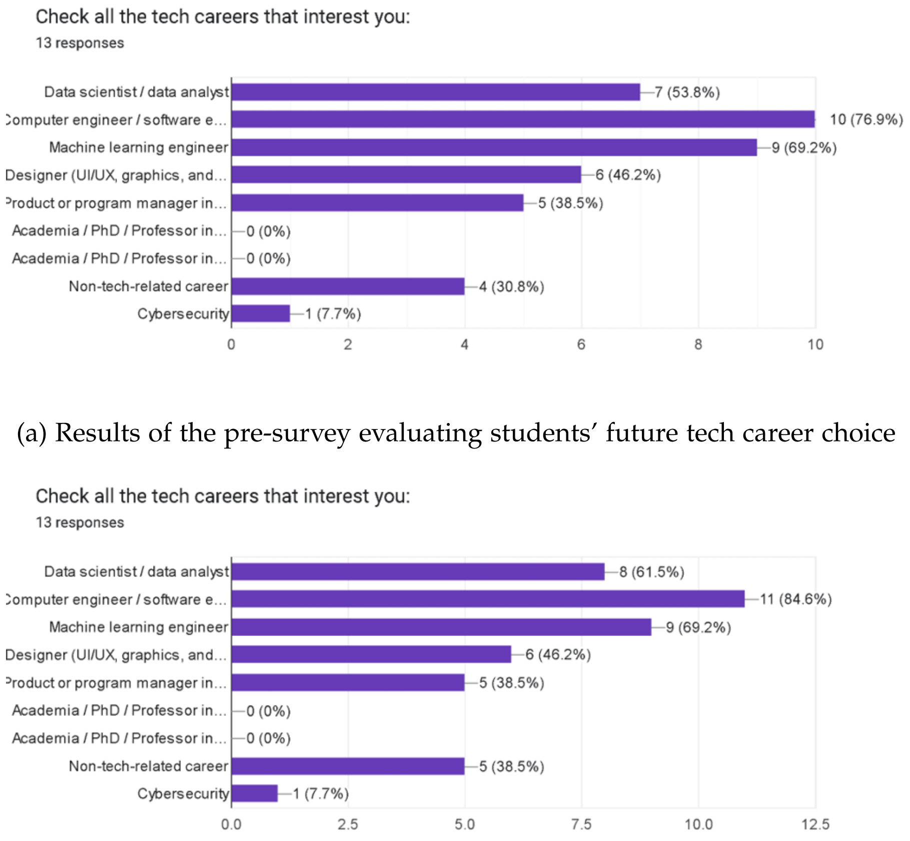


(a) Results of Q1: How much does math or stats excite you?

(b) Results of Q2: How interested would you be in learning how to code?

(c) Results of Q3: How interested would you be in a career that involves coding?

Figure 6.7.: Results of the pre- and post-survey questions evaluating students' motivation on a Likert scale



(a) Results of the pre-survey evaluating students' future tech career choice

(b) Results of the post-survey evaluating students' future tech career choice

Figure 6.8.: Results of the pre- and post-survey questions evaluating students' future tech career choice

Data Literacy

The data action framework increased students' confidence in their data literacy skills

We asked the participants to answer four questions before and after the workshop to evaluate the participants' acquired data literacy skills and, with that, the effectiveness of the data action framework. The questions assessed students' sense of confidence in (1) data science terminology, (2) working with real-life data, (3) data visualization, and (4) data analysis.

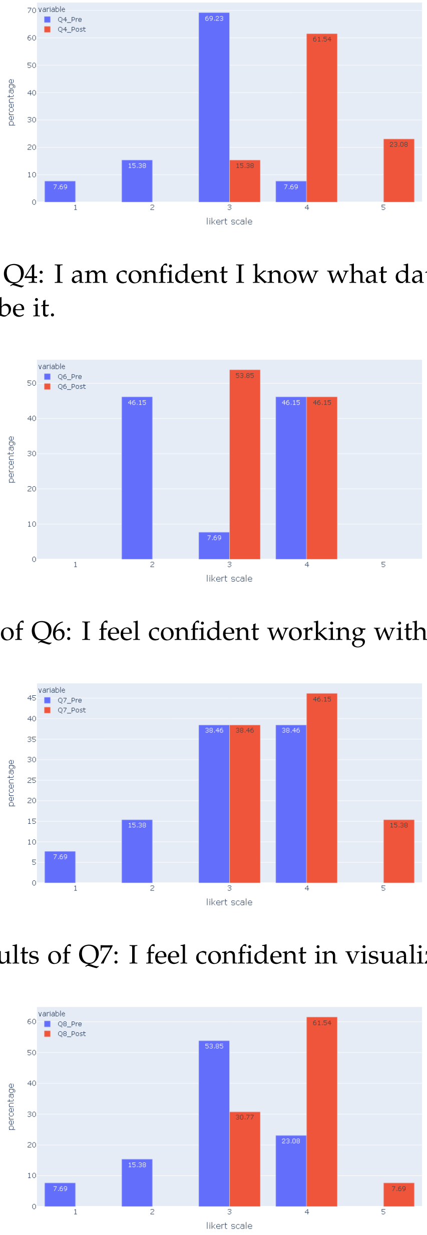
For all four categories, we see in figure 6.9 an apparent increase in participants' confidence in their data literacy skills compared to before the workshop. The absolute majority agree or strongly agree about their data literacy abilities, with 84.62% for Q4 (see 6.9a), 46.15% for Q6 (see 6.9b), 61.53% for Q7 (see 6.9c), and 69.23% for Q8 (see 6.9d).

After the workshop, in the post-survey, we asked the participants to define what data science means to them (Q17). Some described it in the context of problem-solving: "It means how you can extract information and use math to solve certain problems", "it means problem-solving", "data science means to have a challenge you can solve by analyzing and exploring data". Others defined it by describing the data lifecycle: "I guess data science is the process of identifying problems, collecting, processing and using data to inform decisions", "it means the process of studying, validating and interpreting data".

Digital Empowerment & Self-Efficacy

The data action framework empowered students to see themselves as data scientists who could translate their acquired data literacy skills to solve impactful real-world problems using real-world data.

The main goal of the research study is to evaluate how the students' views of themselves and their impact changed after participating in the workshop. This section presents the results of pre- and post-survey questions that attempt to evaluate that. Figure 6.10 shows the pre- and post-responses to question Q5: "I see myself as a data scientist". We see a clear increase in students' view of themselves as data scientists after the workshop. Compared to the pre-results, 15% more agree with the statement "I see myself as a data scientist" after participating in the workshop activities.



(a) Results of Q4: I am confident I know what data science is and can describe it.

(b) Results of Q6: I feel confident working with real-life data.

(c) Results of Q7: I feel confident in visualizing data.

(d) Results of Q8: I feel confident in my ability to extract insights from data (identify patterns, identify anomalies, calculate predictions).

Figure 6.9.: Results of the pre- and post-survey questions evaluating students' acquired data literacy skills on a Likert scale

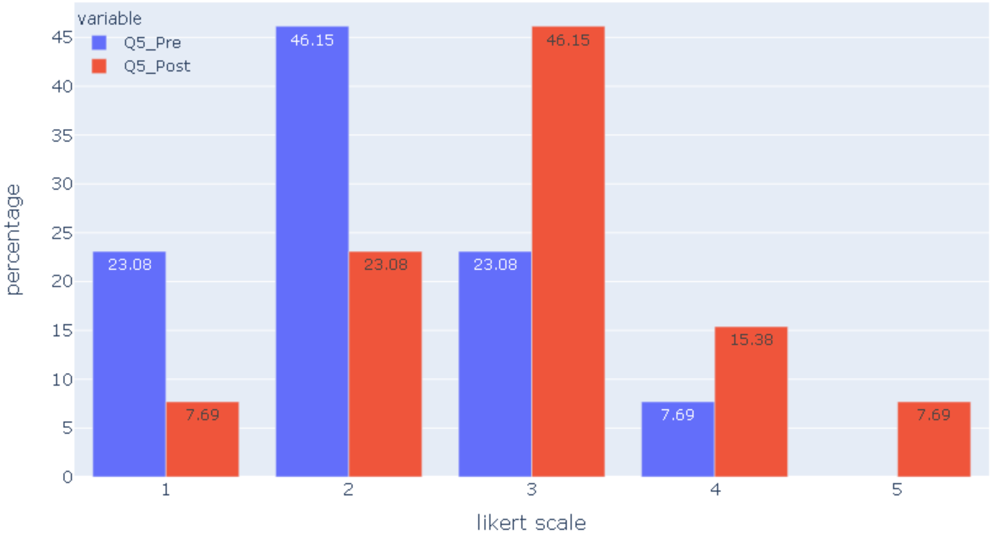


Figure 6.10.: Results of the pre- and post-survey question evaluating participants' view of themselves as data scientists on a Likert scale

In addition, to measure how the workshop empowered the students, we evaluated their confidence in their ability to solve real-world problems and impact their community. Figure 6.11 shows the pre- and post-survey results of the questions evaluating students' digital empowerment. Before the workshop, only around 38% of the participants were confident in their ability to solve real-world problems; after the workshop, this increased to around 77% (see 6.11a), showing an apparent increase in their self-efficacy.

As for their ability to make a positive impact using technology, around 47% agreed after the workshop that they can make a positive change in their world using their acquired digital skills. Also, here we see a noticeable positive increase compared to the pre-survey results (see 6.11b).

Furthermore, we examined if students feel capable of creating their own interesting apps without teachers' help (Q10, Q11). Before the workshop, 53.8% answered "maybe", 7.7% "I don't know", and 38.5% answered yes to the question "given the right tools, could you see yourself making your own interesting app without a teacher's help?". After the workshop, the percentage of participants answering "yes" positively increased to 69.2% while the rest, 30.8%, answered "maybe". This shows an apparent increase in students' confidence in their digital literacy post-workshop.

Data Science in App Inventor

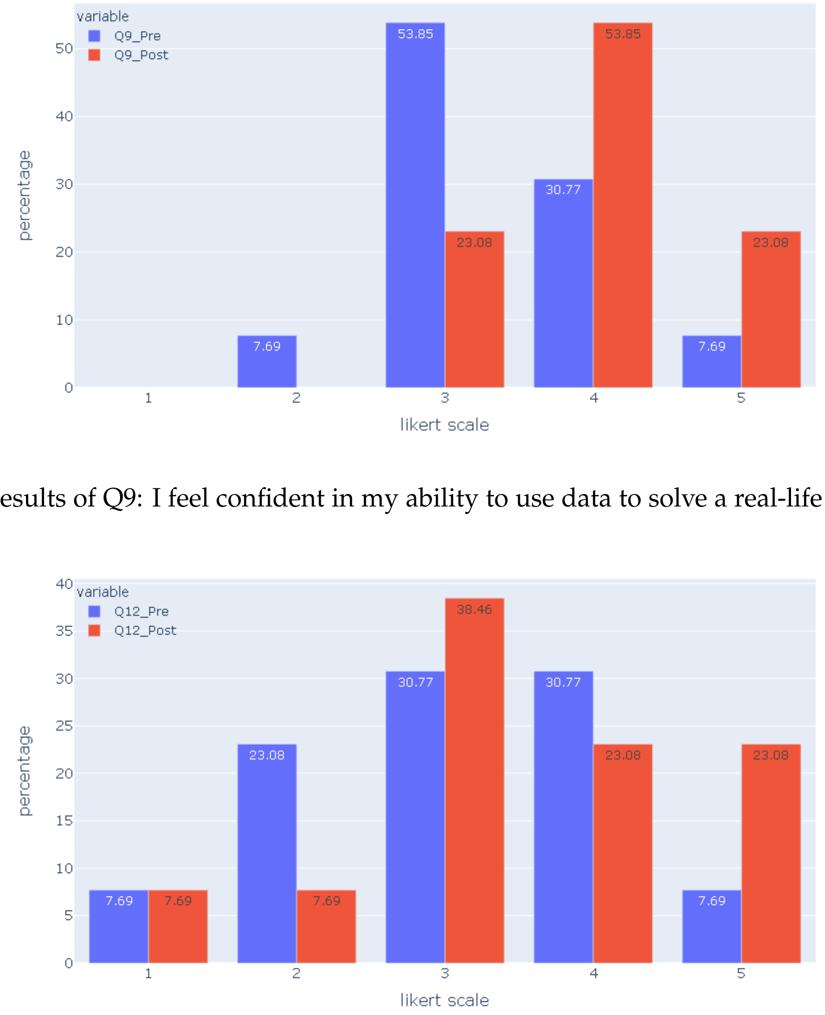
> Students appreciate the simplicity of integrating data science in their mobile apps through drag and drop of ready-to-use code blocks.

In the post-survey, we asked the participants a couple of open-ended questions (Q18 – Q22) to examine their overall experience with the used educational tool, particularly the data science code-blocks in App Inventor. Responses to the post-survey question: "What was the **easiest** part of working with data science components in app inventor?" (Q18) included: "I liked the coding blocks. It was fairly easy", "the blocks were easy to read", "adding to the display as it's just drag and drop", "the block coding is the easiest part", "the logic is pretty simple to understand", and "not having to type everything". To summarize, students liked the ease of integrating complex data science algorithms in mobile apps through a simple drag-and-drop of blocks.

For comparison, we asked them: "What was the **hardest** part of working with data science components in app inventor?" (Q19). We divide their responses into three categories: (1) The connection between App Inventor and the mobile device (Companion App), (2) error handling, and (3) App Inventor UI.

Category (1) includes the following responses: "Was hard at the start to connect to the companion", "getting the data to show up on the app proved a bit tricky", "the tablets were not working for a while", "the connections".
The first setup step during the workshop to connect the starter file with the companion app did not work for most students: They could successfully import the project but not run it on their mobile devices and, thus, could not see the data visualizations at first. This was due to a mistake on our side in the setup of the code blocks responsible for reading data from a spreadsheet. The spreadsheet ID property was not appropriately filled for the component. This mistake occurred through the export and distribution of the starter file.

We identified this issue during the scheduled short break and provided the students with the correct spreadsheet ID to connect their apps. After that, all the students were able to run the app on their mobile devices. This issue is not directly related to our new data science blocks. However, as we see from the participants' responses, it might have slightly affected our research study results. We explained to the students that unexpected errors like these happen in programming all the time and are part of the learning process.



(a) Results of Q9: I feel confident in my ability to use data to solve a real-life challenge.

(b) Results of Q12: I am confident I can make an impact in my own community or in the world using technology.

Figure 6.11.: Results of the pre- and post-survey questions evaluating students' digital empowerment on a Likert scale

This leads to the response we categorized as (2) error handling: “It is hard to pinpoint an error”. While it could be related to the issue mentioned above and how long it took to handle it, we take this as a general criticism regardless. We will address it in the open requirements section 7.1.2.

Because the code blocks are abstract and hide coding details for simplicity, finding bugs in the app is harder. Programming languages, like Python, offer more powerful debug tools than block-based programming languages, but on the other hand, are more complex and require a higher learning curve.

The remaining responses fall under (3) App Inventor UI: “The framework can be hard to navigate”, “changing the structure of the designer app and how it looks”, “there are lots of different functions in app inventor that you need to study, so you know which one to use”, “the hardest part was listening while trying to find the right blocks to place”, and “many components look very similar”. The comments mainly focused on how hard it is to find the right block due to App Inventor’s high number of existing blocks. In addition, some students commented on the design (“look and feel”) of the App UI components. We will address improvement suggestions for these in the future work section 7.3.

For Q20, we asked the participants: “How would you change things to improve the experience of using data science when making apps in app inventor?”. Around 23% stated that they are satisfied with the status quo and would not change anything, 30.7% referenced the issue above connecting App Inventor to the mobile devices and suggested improving that to make the connection faster and “seamless”.

The rest of the responses varied, ranging from: “Maybe a definition of the functions under them, so we know what they do”, “I would try to collect more data and use it to further the app development.”, “Mit app inventor’s overall UI is a bit outdated and clunky but other than that it was good”, and “more integration with graphs”.

Furthermore, we asked the participants: “If you had more time, what changes would you make to improve on the last app you made in the workshop?” (Q21). Around 30.7% said they wouldn’t change anything about the app. The rest submitted the following responses (not including similar responses): “Possible add more buttons to isolate similarities between the graph”, “maybe I’d clean up the interface to make it a little more visually appealing”, “I would make more designs”, “I would have the option to extend the graph to see the x-intercept”, “add changes to the graph maybe to show in the coming years what it might look like”, “make another prediction”, and “I would probably get the Anomaly pruning completed”. The responses were divided between improving the app’s UI and improving the implemented data science features.

Figure 6.12 shows the participants’ responses to Q22: “Would you use app inventor data science components again in the future?”. The majority answered “Maybe” (53.8%), while 38.5% said they would use the data science components in App Inventor again in the future. We consider this a success given that the participants have previous experience with Python, a more powerful programming language for data science, compared to block-based programming languages.

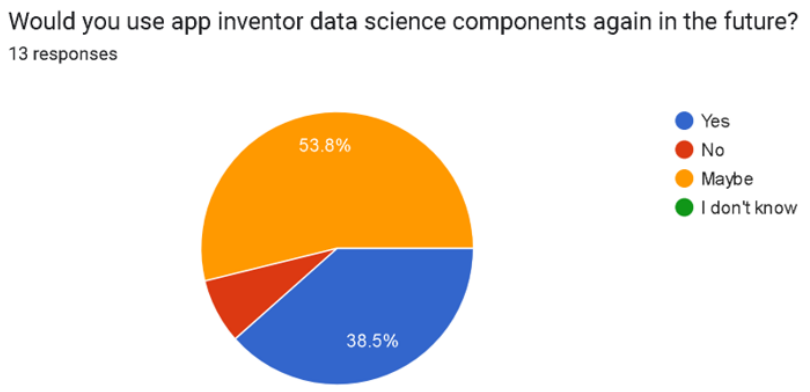
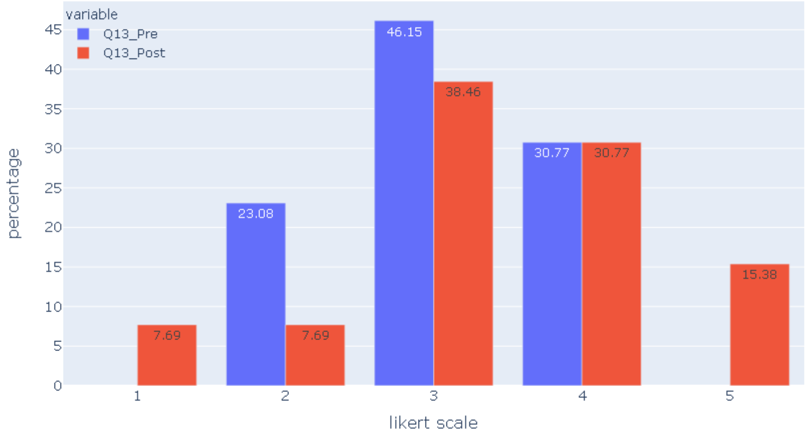


Figure 6.12.: Results of Q22 from the post-survey: Would you use app inventor data science components again in the future?

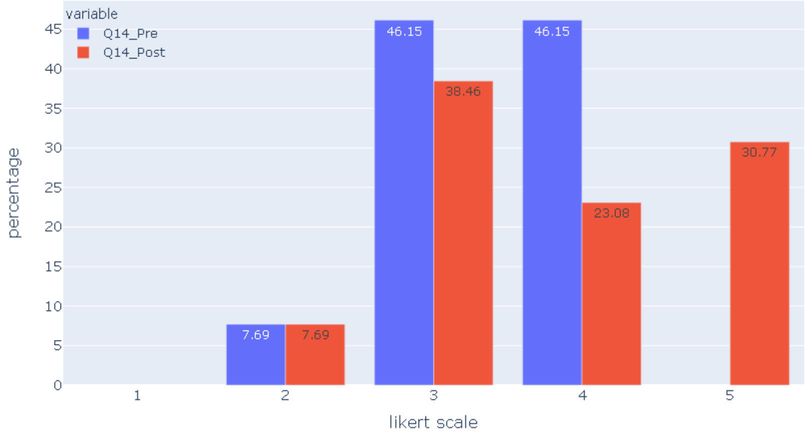
Perception of Data Science

> The data action framework changed students' views on the importance of data cleanup and the effect of anomalies on their linear regression results. The majority believe that including data science would enhance their mobile apps.

In this section, we evaluate how the workshop changed the students' view of data science and its impact. Figure 6.13 shows the pre- and post-survey results of Q13 and Q14. The students were divided for Q13; around 15% did not agree that they would include data science in tech projects they will create in the future, while more than 45% agreed they would, and less than 40% were unsure or did not have a strong opinion regarding that question (see 6.13a). However, the majority believe using data science will enhance their apps and benefit their community and the world, with a noticeable increase in agreement compared to the pre-survey results for the same question (see 6.13b).



(a) Results of Q13: I want to include data science in technology projects that I create



(b) Results of Q14: I believe using data science would enhance my apps and bring a good beneficial impact on my community or the world.

Figure 6.13.: Results of the pre- and post-survey questions evaluating students' perception of data science on a Likert scale

Some responses for Q17: “After this class, what does data science now mean to you”, showed a change in students’ perception of data science. They recognized the interdisciplinary nature of data science, responding to Q17 with: “It means to me that data science can be used in a lot of ways”. They also recognized the value of data and the importance of collecting it: “It shows how useful and how important collecting data is to scientists”.

In addition, we asked them a direct, open-ended question to evaluate their view about data science: “How (if at all) did the class and activities change how you think about data science?” (Q16). Three participants stated that the workshop did not change their views, but rather “reinstated” them: “It didn’t change much because I knew already how beneficial data science was to solving problems. It further reinstated that understanding”.

However, the rest of the participants explained how this workshop changed their view on data science: “I thought data science was something you only observe rather than change and interact with”, “it made me understand the importance of regression models more”, “coming into the class, I thought data science would be a lot harder to understand, but it wasn’t”, “this class made me think about context and anomalies in data”, “it made me think more about how anomalies can make noticeable impacts on your data sets”, “I understand what data science is more and the thought process that goes into it. Going, I knew data science obviously worked with data but not exactly how”, “there’s more math than I expected”, “it changed the way I thought about the effects of outliers when applying data analysis”, “yes, now I can understand data science in a more critical way”, and “data science seems fun. Has a lot of patterns to different solutions.”.

7. Summary

This chapter concludes the thesis by presenting the current status of the educational framework and its related proposed educational tool. We illustrate how we achieved the goals in the objectives section 1.2. Furthermore, we will discuss open requirements and how they could be completed in the future (see 7.1.2) as well as future work that could be built upon this proposed framework (see 7.3).

7.1. Status

This section discusses the achievements realized by the data action educational framework. In addition, we provide an overview of the current status of the proposed educational tool. We compare the system's status with the requirements defined in subsection 4.2.1. We divide the requirements into three categories:
● **Implemented Requirements:** The objective of the requirement has been achieved, and no additional changes are necessary.
◐ **Partially Implemented Requirements:** The requirement's objective is only partially met, necessitating further changes to bring it to a state that meets the client's acceptance criteria.
○ **Not implemented Requirement:** The objective of the requirement is not fulfilled as the requirement is not implemented yet.

7.1.1. Realized Requirements

In this section, we outline the requirements achieved within the context of the 5-months duration of the thesis implementation period. We divide the requirements into two sections: requirements related to the educational framework and requirements related to developing the proposed system in App Inventor.

Proposed Educational Tool

Because we relied on the proposed tool in the research study to realize the educational framework, it was crucial to implement all its functional requirements before the study occurred. And have them tested and ready to be used during the study. Therefore, we prioritized completing a minimal viable product providing the main features needed for the study (functional and non-functional requirements). After the workshop, we further improved the system based on the participants' responses. Table 7.1 shows the status of the functional requirements. We fully implemented and integrated the functional requirements for the anomaly detection and the linear regression components of the data science drawer in App Inventor.

Functional Requirements		Status
FR1A	Detect Anomalies	●
FR2A	Customize Anomaly Detection Algorithms	●
FR3A	Visualize Anomalies	●
FR4A	Remove Anomalies	●
FR1B	Calculate Linear Regression	●
FR2B	Visualize Prediction Results	◐

Table 7.1.: Implementation status of functional requirements
● completed - ◐ partially completed - ○ incomplete

Table 7.2 shows the status of the non-functional requirements. After observing the interaction between the research study participants and the tool during the workshop, we added more code-block descriptions explaining how to use them. In addition, we removed code-blocks clutter by removing duplicate code blocks and adjusting the components' hierarchy to make it easier for the learners to find the code-blocks they need (NFR1, NFR2).

We added more error handling, for example, warning the user if the detected anomalies list is empty or if the x-axis and y-axis lists are of different lengths or empty (NFR3). We optimized the linear regression code-blocks, so the calculation operations are only executed once, increasing the performance of the code-blocks (NFR5). We extensively tested the data science components and integrated them into App Inventor's main source code, used by more than 18 million users (NFR6, NFR7).

Proposed Educational Framework

We created the data action educational framework and tested it with several groups. We applied the framework in full length (six weeks duration) with UROP students, where they experienced an entire data science lifecycle: Identified real-world problems and solved them by building mobile apps with data science capabilities presented in

Non-Functional Requirements		Status
Usability		
NFR1	Ease of Use	●
NFR2	Efficiency	●
Reliability		
NFR3	System Errors Prevention	◐
NFR4	Data Loss Prevention	●
Performance		
NFR5	Response Time	●
Supportability		
NFR6	Extensibility	●
NFR7	Maintainability	●

Table 7.2.: Implementation status of non-functional requirements
● completed - ◐ partially completed - ○ incomplete

section 4.3.

In addition, we implemented a short version of the framework in a 3-hour workshop with high schoolers (ages 16 and 17), where they worked with real-world data to address climate change. They learned about the importance of data cleanup and the effect of anomalies on their prediction results.

7.1.2. Open Requirements

Although we could fulfill all main requirements, there were improvement suggestions and findings resulting from our research study that we did not complete due to the time constraint of this thesis. In the following, we will elaborate on them.

Currently, the "line of best fit" visualization range is as big as the range of the dataset. However, users want to have control over this range and define it themselves, so they can visually see the x-intercept of the line (FR2B).

While we prevent many user errors by, for example, not allowing the code-blocks to snap, we could improve the error handling of the component properties that users need to fill manually. This is not directly related to the data science components. Still, it affects them, which we saw during the workshop when the spreadsheet ID property was not appropriately filled for the spreadsheet component, and the user did not receive any error message (NFR3).

In addition, it would be nice to apply the research study workshop again with more high schoolers, especially those who do not have previous experience with a general-purpose programming language (e.g., Python), to verify the assumptions we made in the results section.

7.2. Conclusion

In this thesis, we democratized data science by creating a low-barrier open-source data science toolkit for anyone to learn and experience a data science lifecycle. We integrated the toolkit in App Inventor, enabling millions of users to experiment with data science and integrate data science in mobile apps to solve real-world problems and make an impact.

We empowered learners in many ways through our proposed data action framework. The research study results show that students feel more confident in their data literacy skills after participating in the framework's activities. They especially learned valuable data cleanup techniques. Students also recognize the power of their newly acquired skills and feel capable of impacting their world through these acquired skills (Goal 3 – see 1.2).

Our work also increased students' awareness of how data is used in their world and influences their daily lives. It increased students' critical thinking ability, as they are now aware of how single erroneous data points (anomalies) could spoil analysis results and manipulate citizens to take specific (in some cases, wrong) actions. On the other hand, they realized data science's power when used for good to make an impact and solve challenging real-world problems (like climate action) based on data evidence (Goal 2 – see 1.2).

We claim that democratizing data science also increased the diversity in this field, motivating more high schoolers of different backgrounds to learn data science and possibly consider data science as their future career path (Goal 1 – see 1.2).

In conclusion, this thesis has significantly contributed to young learners' data science education (see 1.3). It has introduced a comprehensive data action educational framework encompassing four key areas: defining real-world problems, self-acquiring domain knowledge, critical data analysis, and understanding data's societal impact.

Additionally, this work has produced a tailored curriculum designed for K-12 students, effectively teaching them these essential data action principles, especially data cleanup, which no related work has covered before. Furthermore, integrating data science components into App Inventor has enabled students to practically apply the data action framework, allowing them to develop mobile applications with embedded data science features.

Lastly, the results from the research study have provided valuable insights into the efficacy of this framework, demonstrating its ability to empower students with a profound understanding of data's influence and the practical application of data science to effect positive change within their communities. These key contributions collectively represent a significant step forward in fostering data literacy and data-driven problem-solving skills among the next generation of learners.

7.3. Future Work

The work described in this thesis is a promising start for the data action educational framework. We could further develop the work proposed in this thesis to reach and empower even more students with data science. This section illustrates visionary additions to App Inventor and outlines potential areas of research or development for the introduced data action framework that extends beyond the scope of the thesis.

Data Science Algorithms Variety

The anomaly detection component currently integrates one type of algorithm, namely the statistical z-score anomaly detection technique. This method assumes that normal data points follow a known statistical distribution, such as a Gaussian (normal) distribution.

It would be beneficial for learners to experiment with different types of anomaly detection algorithms and compare the results of each to know which is more suitable for their dataset. This would allow them to learn when to use which type of algorithms. In the future, we could, for example, add code-blocks for distance-based (neighbor-based, density-based, and clustering), classification, and angle-based anomaly detection techniques in App Inventor.

The same applies to the regression component, which currently supports linear regression only. We could extend this component with further regression types like logistic regression, polynomial regression, decision tree regression, etc.

We have already started researching how to include classification models in App Inventor. We are working on extending the data science framework to include Naive Bayes. We supervise a UROP student, who started brainstorming and sketching mobile app ideas integrating the Naive Bayes algorithm.

We could also explore other machine-learning directions like reinforcement learning or federated learning and add components in App Inventor that would allow beginners to explore and get in contact with these techniques in a simplified way.

Explore the Integration of Big Data

We currently have the limitation that App Inventor only supports assets of a maximum of 5 MB large. While this dataset size is enough to allow learners to experience a data lifecycle, it would be a great addition to enable learners to experiment with big data. This would require significant changes in App Inventor's infrastructure to support big data. In addition, the storage of big data would introduce more costs. This might be a challenge for App Inventor as it is a non-profit open-source platform.

Generative AI Assistant

With the rise of generative AI tools like ChatGPT [1], people started to question whether the skill to write code will be replaced by generative AI in the future. While we believe that generative AI will reshape the way people learn computing, putting the focus on understanding code rather than on writing code, it is still necessary for individuals to have enough knowledge to assess whether the produced code by generative AI is correct or not. Researching the effect of generative AI on our proposed data action framework is an exciting field to delve into as future work.

Tools working with data, like Microsoft Excel, already integrated a generative AI assistant. We could see the same vision for App Inventor data science components. An AI assistant that suggests which type of plots and which type of algorithms to use, depending on the dataset and the identified problem to be solved using the data. This assistant would also help the learner find the blocks they need to implement a specific data science feature, making it easier to navigate App Inventor's UI.

Data Action Framework for Adults

We have discussed in the context of this thesis the interdisciplinary inherent in data science and the importance of data literacy in almost all fields, even non-technical ones. This opens the opportunity to discover how to empower non-tech-related employees with data literacy skills. The corporate training and development market is constantly growing, and there is a need for data-literate employees. It would be interesting to research how App Inventor and the data action framework could be modified or extended to cater to this target group.

[1] https://chat.openai.com/

A. Appendix

Research Study — Workshop Slides

Data Science
Workshop

06/05/2023
Hanya Elhashemy

Survey

https://bit.ly/datasci-mit

Let‘s Warm Up with the Guessing Game

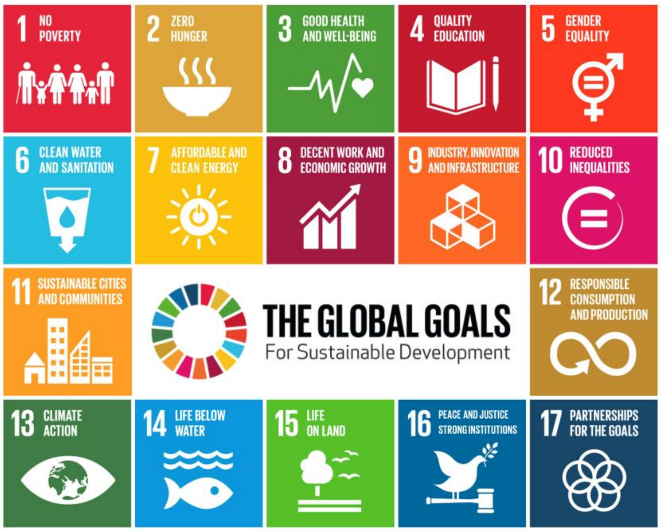
1. Find a partner to guess your favourite food
2. Reveal 3 pieces of information about you that helps your partner identify your favourite food
3. Make it challenging for your partner. You get a point for every wrong guess they make
4. With every wrong guess, you have to reveal 1 additional piece of information about you that would help them guess your favourite food
5. Your partner has a maximum of 3 trials to guess your favourite food

I love travelling to Italy, they have great food there!

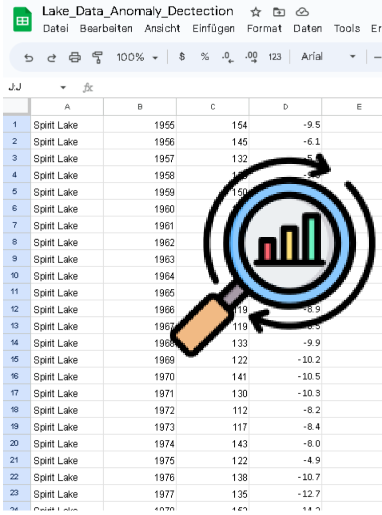
I might be lactose intollerant but I can‘t live without cheese. Cheese is the perfect topping on almost everything.

I love meals that are easy to eat. I can just eat it on the go with my hands

Next Challenge: Climate Change

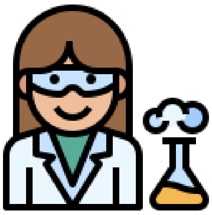

1 NO POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
5 GENDER EQUALITY
6 CLEAN WATER AND SANITATION
7 AFFORDABLE AND CLEAN ENERGY
8 DECENT WORK AND ECONOMIC GROWTH
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE
10 REDUCED INEQUALITIES
11 SUSTAINABLE CITIES AND COMMUNITIES
THE GLOBAL GOALS
For Sustainable Development
12 RESPONSIBLE CONSUMPTION AND PRODUCTION
13 CLIMATE ACTION
14 LIFE BELOW WATER
15 LIFE ON LAND
16 PEACE AND JUSTICE STRONG INSTITUTIONS
17 PARTNERSHIPS FOR THE GOALS

Next Challenge: Climate Change



Lake_Data_Anomaly_Dectection
Datei Bearbeiten Ansicht Einfügen Format Daten Tools Erweiterungen Hilfe
100%
$ % .0 .00 123
Arial
10
B I A
J:J
fx

	A	B	C	D	E	F	G
1	Spirit Lake	1955	154	-9.5			
2	Spirit Lake	1956	145	-6.1			
3	Spirit Lake	1957	132	[illegible]			
4	Spirit Lake	1958	[illegible]	[illegible]			
5	Spirit Lake	1959	150	[illegible]			
6	Spirit Lake	1960	[illegible]	[illegible]			
7	Spirit Lake	1961	[illegible]	[illegible]			
8	Spirit Lake	1962	[illegible]	[illegible]			
9	Spirit Lake	1963	[illegible]	[illegible]			
10	Spirit Lake	1964	[illegible]	[illegible]			
11	Spirit Lake	1965	[illegible]	[illegible]			
12	Spirit Lake	1966	119	-8.9			
13	Spirit Lake	1967	119	[illegible]			
14	Spirit Lake	1968	133	-9.9			
15	Spirit Lake	1969	122	-10.2			
16	Spirit Lake	1970	141	-10.5			
17	Spirit Lake	1971	130	-10.3			
18	Spirit Lake	1972	112	-8.2			
19	Spirit Lake	1973	117	-8.4			
20	Spirit Lake	1974	143	-8.0			
21	Spirit Lake	1975	122	-4.9			
22	Spirit Lake	1976	138	-10.7			
23	Spirit Lake	1977	135	-12.7			



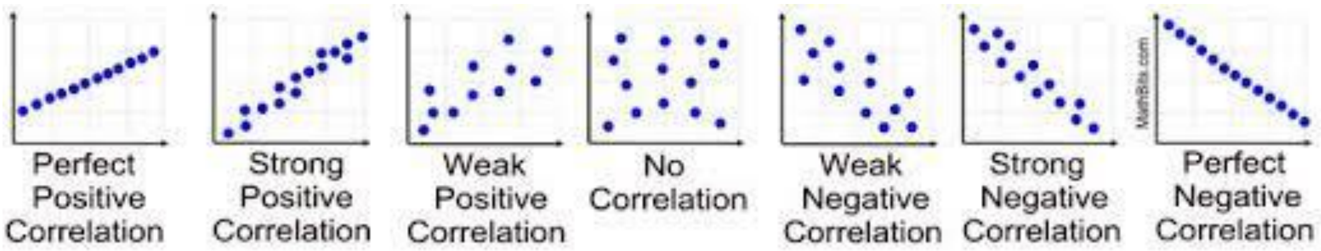
Can you see any insights in this data that would help scientists understand climate change?

Links

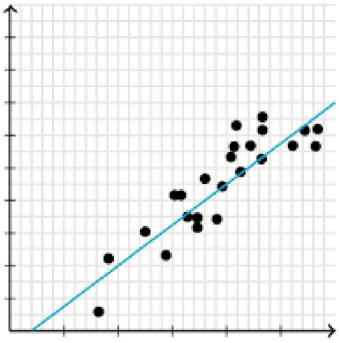
App Inventor: https://datasci.appinventor.mit.edu/

Starter File: https://bit.ly/datasci-mit

Let‘s Try to Find a Correlation

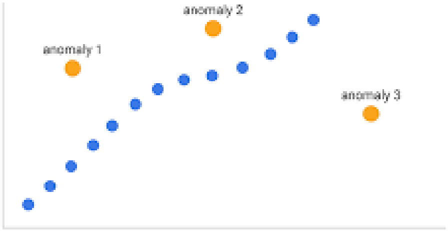


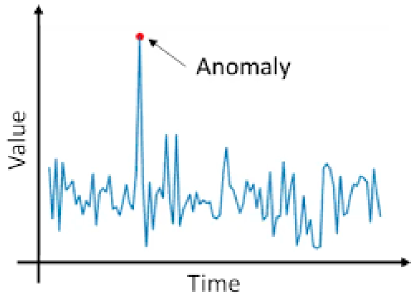
Let‘s Try to Find a Correlation



$y = mx + b$

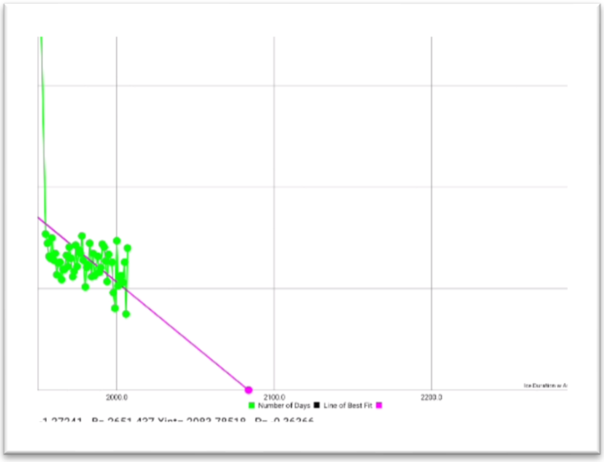
Is Something Messing Up Our Predictions?



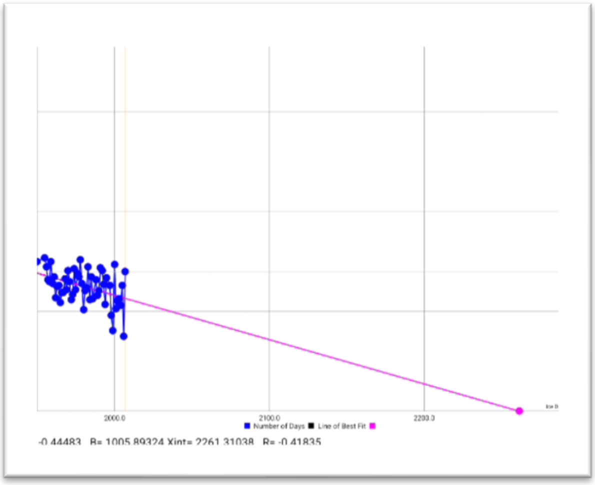


Anomalies!!!

Is Something Messing Up Our Predictions?



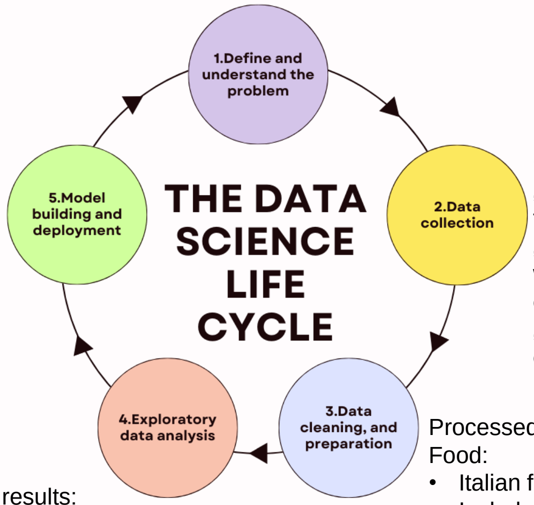
-1.27241 B= 2651.427 Xint= 2082.78518 R= -0.36266



-0.44483 B= 1005.89324 Xint= 2261.31038 R= -0.41835

The Guessing Game as a Data Scientist

Guess my partner‘s favourite food



THE DATA SCIENCE LIFE CYCLE

1.Define and understand the problem

2.Data collection

3.Data cleaning, and preparation

4.Exploratory data analysis

5.Model building and deployment

Collected Data:
„I love travelling to Italy, they have great food there!“
„I might be lactose intollerant but I can‘t live without cheese. Cheese is the perfect topping on almost everything.“
„I love meals that are easy to eat. I can just eat it on the go with my hands“

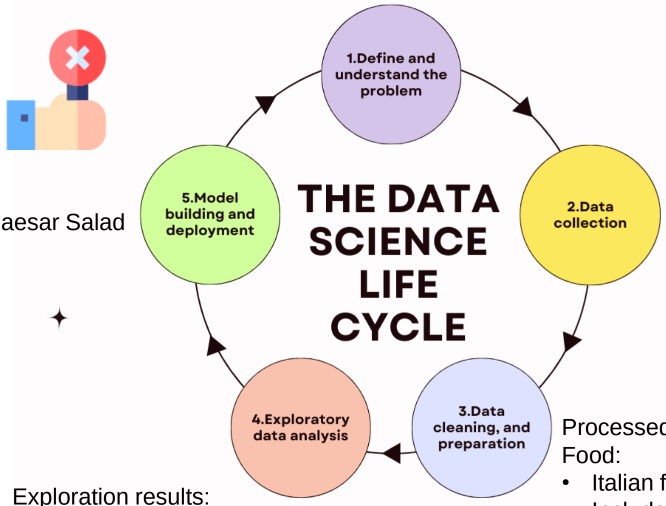
Processed Data:
Food:
- Italian food
- Includes cheese

Eating Habbit:
- Maybe lactose intollerant
- Easy to eat

Exploration results:
The top famous italian food: Pasta, Gnocchi, Risotto, Pizza, Lasagne, Caesar Salad, Gelato, Polenta, Ravioli

The Guessing Game as a Data Scientist

Guess my partner‘s favourite food


1.Define and understand the problem
2.Data collection
3.Data cleaning, and preparation
4.Exploratory data analysis
5.Model building and deployment
THE DATA SCIENCE LIFE CYCLE

Caesar Salad

Collected Data:
„I love travelling to Italy, they have great food there!“
„I might be lactose intollerant but I can‘t live without cheese. Cheese is the perfect topping on almost everything.“
„I love meals that are easy to eat. I can just eat it on the go with my hands“

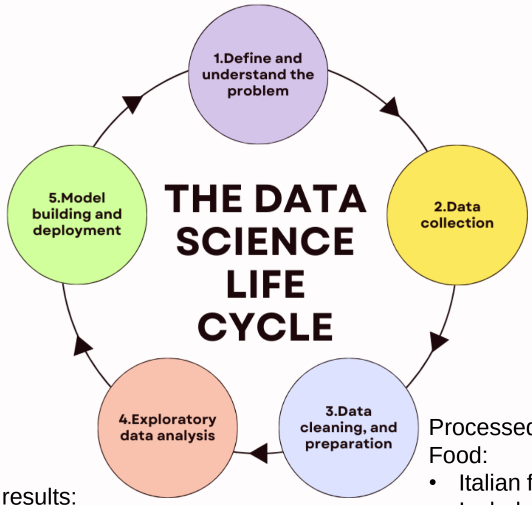
Processed Data:
Food:
- Italian food
- Includes cheese

Eating Habbit:
- Maybe lactose intollerant
- Easy to eat

Exploration results:
The top famous italian food: ~~Pasta, Gnocchi, Risotto~~, Pizza, ~~Lasagne~~, Caesar Salad, ~~Gelato, Polenta, Ravioli~~

The Guessing Game as a Data Scientist

Guess my partner‘s favourite food



THE DATA SCIENCE LIFE CYCLE

1.Define and understand the problem

2.Data collection

3.Data cleaning, and preparation

4.Exploratory data analysis

5.Model building and deployment

Pizza

Collected Data:
„I love travelling to Italy, they have great food there!“
„I might be lactose intollerant but I can‘t live without cheese. Cheese is the perfect topping on almost everything.“
„I love meals that are easy to eat. I can just eat it on the go with my hands“

Processed Data:
Food:

- Italian food
- Includes cheese

Eating Habbit:

- Maybe lactose intollerant
- Easy to eat

Exploration results:
The top famous italian food: ~~Pasta, Gnocchi, Risotto~~, Pizza, ~~Lasagne~~, Caesar Salad, ~~Gelato, Polenta, Ravioli~~

Survey – Last Activity

https://bit.ly/datasci-mit

Abbreviations

UI User Interface

ML Machine Learning

AI Artificial Intelligence

UROP Undergrad Research Opportunities Program

YAIL Young Android Intermediate Language

AIA App Inventor Android

JSON JavaScript Object Notation

XML Extensible Markup Language

SCM Scheme

APK Android Packaged

DTL Decision Tree Learning

RAISE Responsible AI for Social Empowerment and Education

List of Figures

List of Tables

Bibliography

[ALM19] H. Alrubaye, S. Ludi, and M. W. Mkaouer. “Comparison of block-based and hybrid-based environments in transferring programming skills to text-based environments.” In: *arXiv preprint arXiv:1906.03060* (2019).

[AW16] S. Akter and S. F. Wamba. “Big data analytics in E-commerce: a systematic review and agenda for future research.” In: *Electronic Markets* 26 (2016), pp. 173–194.

[Azz+18] A. Azzini, S. Marrara, A. Topalović, M. P. Bach, and M. Rattigan. “Opportunities and Risks for Data Science in Organizations: Banking, Finance, and Policy-Special Session Overview.” In: *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE. 2018, pp. 612–613.

[Baa15] S. Baack. “Datafication and empowerment: How the open data movement re-articulates notions of democracy, participation, and journalism.” In: *Big Data & Society* 2.2 (2015), p. 2053951715594634.

[Bal12] A. Ball. “Review of data management lifecycle models.” In: (2012).

[Bau+15] D. Bau, D. A. Bau, M. Dawson, and C. S. Pickens. “Pencil code: block code for a text world.” In: *Proceedings of the 14th international conference on interaction design and children*. 2015, pp. 445–448.

[BB14] D. Bau and D. A. Bau. “A preview of Pencil Code: A tool for developing mastery of programming.” In: *Proceedings of the 2nd Workshop on Programming for Mobile & Touch*. 2014, pp. 21–24.

[BB15] F. D. Berman and P. E. Bourne. “Let’s make gender diversity in data science a priority right from the start.” In: *PLoS biology* 13.7 (2015), e1002206.

[BD09] B. Bruegge and A. H. Dutoit. “Object–oriented software engineering. using uml, patterns, and java.” In: *Learning* 5.6 (2009), p. 7.

[BD15] R. Bhargava and C. D’Ignazio. “Designing tools and activities for data literacy learners.” In: *Workshop on data literacy, Webscience*. 2015.

[BDB15] D. Bau, M. Dawson, and A. Bau. “Using pencil code to bridge the gap between visual and text-based coding.” In: *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*. 2015, pp. 706–706.

[BGV15] F. Z. Borgesius, J. Gray, and M. Van Eechoud. “Open data, privacy, and fair information principles: Towards a balancing framework.” In: *Berkeley Technology Law Journal* 30.3 (2015), pp. 2073–2131.

[BHS09] G. Bell, T. Hey, and A. Szalay. “Beyond the data deluge.” In: *Science* 323.5919 (2009), pp. 1297–1298.

[Cal06] R. Callingham. “Assessing statistical literacy: A question of interpretation.” In: *International Conference on Teaching Statistics*. 2006.

[Cao17] L. Cao. “Data science: a comprehensive overview.” In: *ACM Computing Surveys (CSUR)* 50.3 (2017), pp. 1–42.

[Car+15] J. Carlson, M. Fosmire, C. C. Miller, and M. S. Nelson. “Determining data information literacy needs.” In: *Data information literacy: Librarians, data, and the education of a new generation of researchers* (2015), pp. 11–33.

[CC18] L. Cao and L. Cao. *Data science thinking*. Springer, 2018.

[Cha20] C. P. Chai. “The importance of data cleaning: Three visualization examples.” In: *Chance* 33.1 (2020), pp. 4–9.

[CJ15] J. Carlson and L. Johnston. *Data information literacy: Librarians, data, and the education of a new generation of researchers*. Purdue University Press, 2015.

[CM13] J. Calzada Prado and M. Á. Marzal. “Incorporating data literacy into information literacy programs: Core competencies and contents.” In: *Libri* 63.2 (2013), pp. 123–134.

[CM18] I. Carmichael and J. Marron. “Data science vs. statistics: two cultures?” In: *Japanese Journal of Statistics and Data Science* 1 (2018), pp. 117–138.

[Con06] M. J. Conway. *How to collect data: Measurement & Evaluation*. American Society for Training and Development, 2006.

[Con10] D. Conway. “The data science Venn diagram.” In: *Recuperado de http://www. dataists. com/2010/09/the-data-science-venn-diagram* (2010).

[Das+19] S. Dash, S. K. Shakyawar, M. Sharma, and S. Kaushik. “Big data in healthcare: management, analysis and future prospects.” In: *Journal of big data* 6.1 (2019), pp. 1–25.

[Dea09] J. Deacon. “Model-view-controller (mvc) architecture.” In: *Online][Citado em: 10 de março de 2006.] http://www. jdl. co. uk/briefings/MVC. pdf* 28 (2009).

[Dea14] E. Deahl. “Better the data you know: Developing youth data literacy in schools and informal learning environments.” In: *Available at SSRN 2445621* (2014).

[Dem+04] J. Demšar, B. Zupan, G. Leban, and T. Curk. "Orange: From experimental machine learning to interactive data mining." In: *Knowledge Discovery in Databases: PKDD 2004: 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa, Italy, September 20-24, 2004. Proceedings 8*. Springer. 2004, pp. 537–539.

[DH17] S. Dasgupta and B. M. Hill. "Scratch community blocks: Supporting children as data scientists." In: *Proceedings of the 2017 CHI conference on human factors in computing systems*. 2017, pp. 3620–3631.

[Dha13] V. Dhar. "Data science and prediction." In: *Communications of the ACM* 56.12 (2013), pp. 64–73.

[DK20] C. D'ignazio and L. F. Klein. *Data feminism*. MIT press, 2020.

[Don17] D. Donoho. "50 years of data science." In: *Journal of Computational and Graphical Statistics* 26.4 (2017), pp. 745–766.

[DP12] T. H. Davenport and D. Patil. "Data scientist." In: *Harvard business review* 90.5 (2012), pp. 70–76.

[DZ12] J. Demšar and B. Zupan. "Orange: Data mining fruitful and fun." In: *Inf. Družba IS* 6 (2012), pp. 1–486.

[DZ13] J. Demšar and B. Zupan. "Orange: Data mining fruitful and fun-a historical perspective." In: *Informatica* 37.1 (2013).

[EA16] J. A. Espinosa and F. Armour. "The big data analytics gold rush: a research framework for coordination and governance." In: *2016 49th Hawaii International Conference on System Sciences (HICSS)*. IEEE. 2016, pp. 1112–1121.

[EH95] Y. Escoufier and C. Hayashi. *Data science and its applications*. Academic Press/Harcourt Brace, 1995.

[FGF17] A. Feng, M. Gardner, and W.-c. Feng. "Parallel programming with pictures is a Snap!" In: *Journal of Parallel and Distributed Computing* 105 (2017), pp. 150–162.

[Fis22] K. Fisler. "Data-Centricity: Rethinking Introductory Computing to Support Data Science." In: *1st International Workshop on Data Systems Education*. 2022, pp. 1–3.

[Fra+13] N. Fraser et al. "Blockly: A visual programming editor." In: *URL: https://code. google. com/p/blockly* 42 (2013).

[Fra+16] M. Frank, J. Walker, J. Attard, and A. Tygel. "Data Literacy-What is it and how can we make it happen?" In: *The Journal of Community Informatics* 12.3 (2016).

[Fre93] P. Freire. "Pedagogy of the oppressed (20th-Anniversary ed.)" In: *New York Continuum* (1993).

[FW16] M. Frank and J. Walker. "Some key challenges for data literacy." In: *The Journal of Community Informatics* 12.3 (2016).

[GCB12] J. Gray, L. Chambers, and L. Bounegru. *The data journalism handbook: How journalists can use data to improve the news.* " O'Reilly Media, Inc.", 2012.

[Gel13] A. Gelman. "Statistics is the least important part of data science." In: *Statistical Modeling, Causal Inference, and Data Science blog* (2013).

[GR18] A. Grillenberger and R. Romeike. "Developing a theoretically founded data literacy competency model." In: *Proceedings of the 13th Workshop in Primary and Secondary Computing Education*. 2018, pp. 1–10.

[GR19] A. Grillenberger and R. Romeike. "About classes and trees: Introducing secondary school students to aspects of data mining." In: *Informatics in Schools. New Ideas in School Informatics: 12th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2019, Larnaca, Cyprus, November 18–20, 2019, Proceedings 12*. Springer. 2019, pp. 147–158.

[Has+21] H. Hassani, C. Beneki, E. S. Silva, N. Vandeput, and D. Ø. Madsen. "The science of statistics versus data science: What is the future?" In: *Technological Forecasting and Social Change* 173 (2021), p. 121111.

[Hay98] C. Hayashi. "What is data science? Fundamental concepts and a heuristic example." In: *Data Science, Classification, and Related Methods: Proceedings of the Fifth Conference of the International Federation of Classification Societies (IFCS-96), Kobe, Japan, March 27–30, 1996*. Springer. 1998, pp. 40–51.

[HBH22] A. A. de Hond, M. M. van Buchem, and T. Hernandez-Boussard. "Picture a data scientist: a call to action for increasing diversity, equity, and inclusion in the age of AI." In: *Journal of the American Medical Informatics Association* 29.12 (2022), pp. 2178–2181.

[Hea15] D. Headrick. "Electric utilities and energy market innovations." In: *Research Technology Management* 58.3 (2015), p. 2.

[Hei+18] B. Heinemann, S. Opel, L. Budde, C. Schulte, D. Frischemeier, R. Biehler, S. Podworny, and T. Wassong. "Drafting a data science curriculum for secondary schools." In: *Proceedings of the 18th Koli calling international conference on computing education research*. 2018, pp. 1–5.

[Hey09] T. Hey. *The fourth paradigm*. United States of America., 2009.

[Hoo94] J. N. Hooker. “Needed: An empirical science of algorithms.” In: *Operations research* 42.2 (1994), pp. 201–212.

[IE17] M. A. Inventor and M. Explore. “App inventor.” In: *lınea]. Disponible en: http://appinventor. mit. edu/explore/.[Accedido: 26-may-2015]* (2017).

[Jes+18] M. Jesmeen, J. Hossen, S. Sayeed, C. Ho, K. Tawsif, A. Rahman, and E. Arif. “A survey on cleaning dirty data using machine learning paradigm for big data analytics.” In: *Indonesian Journal of Electrical Engineering and Computer Science* 10.3 (2018), pp. 1234–1243.

[Kor+19] J. Kornak, J. Fields, W. Kremers, S. Farmer, H. W. Heuer, L. Forsberg, D. Brushaber, A. Rindels, H. Dodge, S. Weintraub, et al. “Nonlinear Z-score modeling for improved detection of cognitive abnormality.” In: *Alzheimer's & Dementia: Diagnosis, Assessment & Disease Monitoring* 11.1 (2019), pp. 797–808.

[KP18] C. Kadar and I. Pletikosa. “Mining large-scale human mobility data for long-term crime prediction.” In: *EPJ Data Science* 7.1 (2018), pp. 1–27.

[Li+21] P. Li, X. Rao, J. Blase, Y. Zhang, X. Chu, and C. Zhang. “CleanML: A study for evaluating the impact of data cleaning on ml classification tasks.” In: *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE. 2021, pp. 13–24.

[LSR19] A. Lindner, S. Seegerer, and R. Romeike. “Unplugged Activities in the Context of AI.” In: *Informatics in Schools. New Ideas in School Informatics: 12th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2019, Larnaca, Cyprus, November 18–20, 2019, Proceedings 12*. Springer. 2019, pp. 123–135.

[LW21] Y. Lin and D. Weintrop. “The landscape of Block-based programming: Characteristics of block-based environments and how they support the transition to text-based programming.” In: *Journal of Computer Languages* 67 (2021), p. 101075.

[Mal+10] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond. “The scratch programming language and environment.” In: *ACM Transactions on Computing Education (TOCE)* 10.4 (2010), pp. 1–15.

[Mar23] S. Marjona. “DATA SCIENCE.” In: *Journal of new century innovations* 32.2 (2023), pp. 69–72.

[McC+07] J. McCarthy et al. “What is artificial intelligence.” In: (2007).

[MD18] F. Murtagh and K. Devlin. "The development of data science: implications for education, employment, research, and the data revolution for sustainable development." In: *Big Data and Cognitive Computing* 2.2 (2018), p. 14.

[MG12] E. B. Mandinach and E. S. Gummer. "Navigating the Landscape of Data Literacy: It IS Complex." In: *WestEd* (2012).

[MG13] E. B. Mandinach and E. S. Gummer. "A systemic view of implementing data literacy in educator preparation." In: *Educational Researcher* 42.1 (2013), pp. 30–37.

[MH23] K. Mike and O. Hazzan. "What is Data Science?" In: *Communications of the ACM* 66.2 (2023), pp. 12–13.

[Mic+23] T. Michaeli, S. Seegerer, L. Kerber, and R. Romeike. "Data, Trees, and Forests–Decision Tree Learning in K-12 Education." In: *arXiv preprint arXiv:2305.06442* (2023).

[Mik+22] K. Mike, N. Ragonis, R. B. Rosenberg-Kima, and O. Hazzan. "Computational thinking in the era of data science." In: *Communications of the ACM* 65.8 (2022), pp. 33–35.

[MLD18] L. Moors, A. Luxton-Reilly, and P. Denny. "Transitioning from block-based to text-based programming languages." In: *2018 International Conference on Learning and Teaching in Computing and Engineering (LaTICE)*. IEEE. 2018, pp. 57–64.

[MPS19] T. Munasinghe, E. W. Patton, and O. Seneviratne. "Iot application development using mit app inventor to collect and analyze sensor data." In: *2019 IEEE International Conference on Big Data (Big Data)*. IEEE. 2019, pp. 6157–6159.

[MT10] A. V. Maltese and R. H. Tai. "Eyeballs in the fridge: Sources of early interest in science." In: *International Journal of Science Education* 32.5 (2010), pp. 669–685.

[OF21] A. M. Olney and S. D. Fleming. "JupyterLab Extensions for Blocks Programming, Self-Explanations, and HTML Injection." In: *Joint Proceedings of the Workshops at the 14th International Conference on Educational Data Mining*. Vol. 3051. 2021.

[OFJ21] A. M. Olney, S. D. Fleming, and J. C. Johnson. "Learning data science with Blockly in JupyterLab." In: *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. 2021, pp. 1373–1373.

[OR21] V. Olari and R. Romeike. “Addressing ai and data literacy in teacher education: A review of existing educational frameworks.” In: *The 16th Workshop in Primary and Secondary Computing Education*. 2021, pp. 1–2.

[Pac22] W. R. Paczkowski. *Business Analytics: Data Science for Business Problems*. Springer Nature, 2022.

[PFM17] E. Pasternak, R. Fenichel, and A. N. Marshall. “Tips for creating a block language with blockly.” In: *2017 IEEE blocks and beyond workshop (B&B)*. IEEE. 2017, pp. 21–24.

[Pie15] P. J. Piety. *Assessing the educational data movement*. Teachers College Press, 2015.

[PK17] B. Plale and I. Kouper. “The centrality of data: data lifecycle and data pipelines.” In: *Data analytics for intelligent transportation systems*. Elsevier, 2017, pp. 91–111.

[Pol+18a] J. G. Politz, K. Fisler, S. Krishnamurthi, and B. S. Lerner. “From Spreadsheets to Programs: Data Science and CS1 in Pyret.” In: *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. 2018, pp. 1058–1058.

[Pol+18b] N. Polyzotis, S. Roy, S. E. Whang, and M. Zinkevich. “Data lifecycle challenges in production machine learning: a survey.” In: *ACM SIGMOD Record* 47.2 (2018), pp. 17–28.

[PP07] A. Patcha and J.-M. Park. “An overview of anomaly detection techniques: Existing solutions and latest technological trends.” In: *Computer networks* 51.12 (2007), pp. 3448–3470.

[Pra+22] Y. N. Prajapati, U. Sesadri, T. Mahesh, S. Shreyanth, A. Oberoi, and K. P. Jayant. “Machine Learning Algorithms in Big Data Analytics for Social Media Data Based Sentimental Analysis.” In: *International Journal of Intelligent Systems and Applications in Engineering* 10.2s (2022), pp. 264–267.

[PTH19] E. W. Patton, M. Tissenbaum, and F. Harunani. “MIT app inventor: Objectives, design, and development.” In: *Computational thinking education* (2019), pp. 31–49.

[Ray13] J. M. Ray. *Research data management: Practical strategies for information professionals*. Purdue University Press, 2013.

[RD+00] E. Rahm, H. H. Do, et al. “Data cleaning: Problems and current approaches.” In: *IEEE Data Eng. Bull.* 23.4 (2000), pp. 3–13.

[Res+09] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, et al. "Scratch: programming for all." In: *Communications of the ACM* 52.11 (2009), pp. 60–67.

[Sau21] J. R. Saura. "Using data sciences in digital marketing: Framework, methods, and performance metrics." In: *Journal of Innovation & Knowledge* 6.2 (2021), pp. 92–102.

[Sch+14] J. Schiller, F. Turbak, H. Abelson, J. Dominguez, A. McKinney, J. Okerlund, and M. Friedman. "Live programming of mobile apps in App Inventor." In: *Proceedings of the 2nd Workshop on Programming for Mobile & Touch*. 2014, pp. 1–8.

[SDH18] J. S. Saltz, N. I. Dewar, and R. Heckman. "Key concepts for a data science ethics curriculum." In: *Proceedings of the 49th ACM technical symposium on computer science education*. 2018, pp. 952–957.

[Shi05] M. Shields. "Information literacy, statistical literacy, data literacy." In: *IASSIST quarterly* 28.2-3 (2005), pp. 6–6.

[Sil09] J. Silvertown. "A new dawn for citizen science." In: *Trends in ecology & evolution* 24.9 (2009), pp. 467–471.

[Sil20] N. Silver. *What I need from statisticians-Statistics views*. 2020.

[SM19] K. Stathoulopoulos and J. C. Mateos-Garcia. "Gender diversity in AI research." In: *Available at SSRN 3428240* (2019).

[Spe18] M. G. Speranza. "Trends in transportation and logistics." In: *European Journal of Operational Research* 264.3 (2018), pp. 830–836.

[Tay16] D. Taylor. "Battle of the data science Venn diagrams." In: *KDNuggets News* (2016).

[TF09] B. Trilling and C. Fadel. *21st century skills: Learning for life in our times*. John Wiley & Sons, 2009.

[Tis+18a] M. Tissenbaum, J. Sheldon, H. Abelson, and M. Sherman. "Examining a Secondary School Computational Action Curriculum Using App Inventor and the Internet of Things." In: *Copyright 2018 The Hong Kong Jockey Club All rights reserved. ISBN: 978-988-77034-5-7* (2018), p. 104.

[Tis+18b] M. Tissenbaum, M. A. Sherman, J. Sheldon, and H. Abelson. "From computational thinking to computational action: Understanding changes in computational identity through app inventor and the internet of things." In: International Society of the Learning Sciences, Inc.[ISLS]., 2018.

[TSA] M. Tissenbaum, J. Sheldon, and H. Abelson. “Reducing the Barriers for Computational Action.” In: ().

[Tuk62] J. W. Tukey. “The future of data analysis.” In: *The annals of mathematical statistics* 33.1 (1962), pp. 1–67.

[TV90] K. W. Thomas and B. A. Velthouse. “Cognitive elements of empowerment: An “interpretive” model of intrinsic task motivation.” In: *Academy of management review* 15.4 (1990), pp. 666–681.

[TWD18] S. Tiwari, H.-M. Wee, and Y. Daryanto. “Big data analytics in supply chain management between 2010 and 2016: Insights to industries.” In: *Computers & Industrial Engineering* 115 (2018), pp. 319–330.

[VA16] W. Van Der Aalst and W. van der Aalst. *Data science in action*. Springer, 2016.

[Vah+06] P. Vahey, L. Yarnall, C. Patton, D. Zalles, and K. Swan. “Mathematizing middle school: Results from a cross-disciplinary study of data literacy.” In: *Annual Meeting of the American Educational Research Association, San Francisco, CA*. 2006, pp. 1–15.

[Vou14] Z. Voulgaris. *Data scientist: the definitive guide to becoming a data scientist*. Technics Publications, 2014.

[WAF15] D. Wolber, H. Abelson, and M. Friedman. “Democratizing computing with app inventor.” In: *GetMobile: Mobile Computing and Communications* 18.4 (2015), pp. 53–58.

[Wal93] K. K. Wallman. “Enhancing statistical literacy: Enriching our society.” In: *Journal of the American Statistical Association* 88.421 (1993), pp. 1–8.

[Wei05] S. Weisberg. *Applied linear regression*. Vol. 528. John Wiley & Sons, 2005.

[Wil+14] S. Williams, E. Deahl, L. Rubel, and V. Lim. “City digits: Local lotto: Developing youth data literacy by investigating the lottery.” In: *Journal of Digital Media Literacy* 2.2 (2014), pp. 1–16.

[Wil21] R. Williams. “How to train your robot: project-based ai and ethics education for middle school classrooms.” In: *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. 2021, pp. 1382–1382.

[Win19] J. M. Wing. “The data life cycle.” In: *Harvard Data Science Review* 1.1 (2019), p. 6.

[WP99] C. J. Wild and M. Pfannkuch. “Statistical thinking in empirical enquiry.” In: *International statistical review* 67.3 (1999), pp. 223–248.

[XA16] B. Xie and H. Abelson. "Skill progression in MIT app inventor." In: *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE. 2016, pp. 213–217.

[Yau09] N. Yau. "Rise of the data scientist." In: *Retrieved from* (2009).

[Zha+17] H. Zhang, J. Li, K. Kara, D. Alistarh, J. Liu, and C. Zhang. "ZipML: Training linear models with end-to-end low precision, and a little bit of deep learning." In: *International Conference on Machine Learning*. PMLR. 2017, pp. 4035–4043.

[ZIE10] A. F. Zuur, E. N. Ieno, and C. S. Elphick. "A protocol for data exploration to avoid common statistical problems." In: *Methods in ecology and evolution* 1.1 (2010), pp. 3–14.

[ZVL20] X. Zhou, J. Van Brummelen, and P. Lin. "Designing AI learning experiences for K-12: Emerging works, future opportunities and a design framework." In: *arXiv preprint arXiv:2009.10228* (2020).