CITY College,
International Faculty of the University of Sheffield

Computer Science Department

FINAL YEAR INDIVIDUAL PROJECT

COVID Predictions

This report is submitted in partial fulfillment of the requirement for the degree of Bachelors in Computer Science with Honours by

Marino Osmanllari

June, 2022

Approved

Mr. Miltos Vafiadis

_____

COVID Predictions

By

Marino Osmanllari

Supervisor

Mr M. Vafiadis

Abstract

This dissertation aims to present and implement a web-based, highly accurate COVID Prediction machine. The rationale behind developing and implementing a Covid Prediction web application is that through Machine Learning technology, train algorithms to be used to analyze large amounts of data such as infection rates, population demographics and healthcare capacity, to make predictions about the future course of the disease. The report will present an overview of machine learning technology fundamentals, demonstrate cloud computing trained calculations and employ the estimated calculations in a secure web application.

DECLARATION

All sentences or passages quoted in this dissertation from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this dissertation and the degree examination as a whole.

I have completed and submitted this work by myself without assistance from or communication with another person either external or fellow student, or any AI type of content generator. I understand that not working on my own will be considered grounds for unfair means and will result in a fail mark for this work and might invoke disciplinary actions. It is also my understanding that intellectual rights of this work are shared equally between myself (the author of the dissertation), the supervisor of the project, and CITY College, International Faculty of The University of Sheffield, and that any kind of exploit of this work will need the consent of all parties involved.

☐ I agree that my dissertation can be used as a model/example by future undergraduate students, for educational purposes only.

Name:

Signed:

Date:

Acknowledgements

I would like to extend my heartfelt appreciation to my supervisor, Miltos Vafiadis. Your guidance and support have been invaluable throughout my final year project.

I would like to thank my family for their support and encouragement during my academic years. Their affection, support, and faith in my skills have all played an essential part in my achievements.

To my friends and classmates, thank you for making these 3 years way more enjoyable, and I am fortunate to have shared countless memories with all of you.

Thank you all for being a part of my journey.

Contents

Figures

Tables

Listings

Chapter 1

1. Introduction

The COVID-19 pandemic has had a significant global impact, touching almost all aspects of daily life [1]. The virus, originally discovered in late 2019, has quickly spread to almost every nation, causing widespread disease and fatalities. The response to the pandemic has required unprecedented measures, including lockdowns, social distancing, and the deployment of massive amounts of resources to healthcare systems [2].

Additionally, the epidemic has had significant economic and societal repercussions. It led to widespread unemployment and economic disruption [2]. The pandemic has also exposed and exacerbated inequalities, particularly in marginalized communities [3].

It spurred innovation and creativity in various fields, including technology and education. The widespread use of remote work, virtual meetings, and online learning has transformed the way we work and learn.

However, the pandemic revealed the vulnerabilities and gaps in our systems and structures, including healthcare, economic and social systems [1]. It has emphasized the importance of increased resilience and preparedness in the face of future pandemics and crisis [2].

Despite the difficulties posed by the pandemic, some encouraging things have happened. Hope for a future return to normalcy is given by the quick discovery of vaccinations and the implementation of immunization programs [4]. Furthermore, the pandemic has highlighted the necessity of scientific research in dealing with global health emergencies at international cooperation level.

As the globe faces the consequences of the COVID-19 pandemic, it is apparent that the COVID-19 era will be regarded as one of extraordinary struggle and change.

1.1 Project Background & Motivation

The COVID-19 pandemic has had a profound effect on populations all over the world, posing enormous difficulties for public health, economy or daily life. It is essential for successful decision-making, resource allocation and public health measures that the distribution and effects of the virus can be predicted accurately [5]. By examining numerous data sources to identify patterns and trends, machine learning can help predict COVID-19 cases. In order to build predictive models that can aid in anticipating the spread of COVID-19, Azure Machine Learning and AWS deliver strong platforms, allowing policymakers and healthcare professionals to make educated decisions and conduct targeted treatments [6]. By utilizing their capabilities, it is desired to create a reliable and accurate prediction model that can predict COVID-19 trends and help in the creation of strategies or schemas that will effectively combat the virus and lessen its effects on communities .

The pandemic has shown how critical it is to make accurate predictions about the virus's progress [7]. The issue people want to solve is the creation of a prediction model that can anticipate COVID-19 patterns at both the regional and global levels, including the number of cases, hospitalizations and possible outbreaks. The intention is to get over the constraints of conventional statistical models and improve the precision and timeliness of COVID-19 forecasts by utilizing Azure Machine Learning. In order to stop the spread of the virus this initiative will help with effective decision-making, as well as resource allocation.

1.2 Aim

The underlying aim has been developed using the challenges outlined above, as well as the rationale for this project:

"The aim of the project is to develop and build an end-to-end solution that can make predictions based on input data for estimated Covid-19 predictions for a specific future date."

1.3 Objectives

A list of objectives has been composed to work and reach the project's aim. These goals are the cornerstones of how the system will be built and the platform's design.

1. Create and train a machine learning model to forecast the likelihood of COVID-19 cases in all European Countries.
2. Evaluate the performance of the machine learning model using various metrics and identify areas for improvement.

3. Deploy the machine learning model as a web service in Azure ML, allowing users to input new data and receive real-time predictions on COVID-19 cases in a newly created web application.

The requirements will be examined, reviewed, put into practice and tested to create the system that achieves the project's goal.

1.4 Report Structure

A project that carries this complexity needs extensive documentation to back up the decisions made throughout development.

Each chapter focuses on describing a certain element of the project. There are 10 chapters that provide further information:

1. Introduction
2. Literature Review
3. System Description & Requirements
4. Project Planning
5. Azure Machine Learning
6. Azure Setup & Predictions
7. Web Application
8. Testing
9. Evaluation
10. Conclusions

Chapter 2

2. Literature Review

2.1 Artificial Intelligence

Artificial intelligence is a broad field that entails the creation of intelligent systems and software capable of doing activities that ordinarily require human intellect [8]. Numerous sectors, including healthcare, finance, transportation, education, might be drastically changed by artificial intelligence [9].

Machine Learning is a subset of the field of Artificial Intelligence that specializes in the use of algorithms and models to leverage computational systems in processing large amounts of data. Its primary applications are decision making and predictive models [10].

There are also potential ethical and societal implications of AI [9]. As AI systems become more sophisticated and interconnected into our lives, it is critical to evaluate the possible implications for employment, privacy, and power and resource distribution.

AI, encompassing machine learning as a key component, represents the grand span of intelligent technologies that continues to evolve and expand with the passage of time [9]. While machine learning has already demonstrated immense potential and impact across various domains, the broader field of AI holds even greater promise for the future.

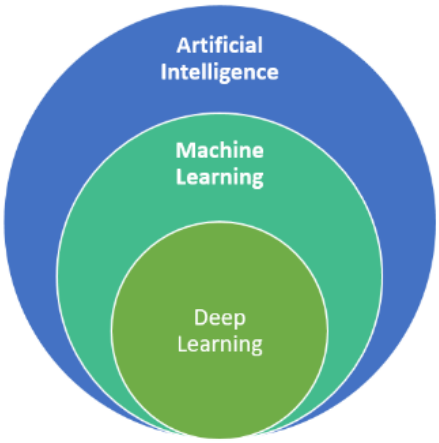

Figure 2.1: Artificial Intelligence Subsets

Machine learning, as a subset of AI, focuses on training algorithms to learn from data and make predictions or take actions based on that learning [11]. It has already revolutionized numerous industries. However, AI encompasses a wider spectrum of capabilities beyond just

learning from data. We may anticipate AI to grow increasingly sophisticated, adaptive, versatile and capable of tackling complicated tasks as it evolves. AI could evolve further, Deep Learning and Neural Networks, Natural Language Processing, Computer Vision, Reinforcement Learning, AI Ethics and Explainability, Collaboration with Human Intelligence and many more [11].

It is essential to highlight that as AI develops, there will be new challenges and considerations. Ethical dilemmas, data privacy concerns with potential disruptions in the workforce are challenges that will require a multidisciplinary approach involving not only technologists but also policymakers, ethicists and society as a whole [12].

2.2 Machine Learning

Machine learning is a subfield of artificial intelligence that makes computers or devices learn from data and make predictions or judgments based on it by using statistical models and algorithms [13]. Due to the accessibility of vast amounts of data and the advancement of powerful computer capabilities, it has grown in popularity recently.

Supervised learning and unsupervised learning are the two primary divisions of machine learning algorithms [14]. The objective is to create a model that can predict outcomes using new, unseen data [15]. A variety of problems, including picture and audio identification, natural language processing, and outcome prediction in industries including healthcare, finance, marketing, have been effectively tackled by machine learning [15].

Machine learning algorithms come in a vast variety and are utilized for a variety of applications. Common algorithms include decision trees, support vector machines, neural networks, logistic regression and linear regression [13]. The unique properties of the data and the issue at hand will determine which algorithm is used.

Machine learning is important because it supports the development of new products and provides organizations with a picture of consumer behavior trends and operational business patterns. For many businesses, machine learning has emerged as a key competitive differentiator [15].

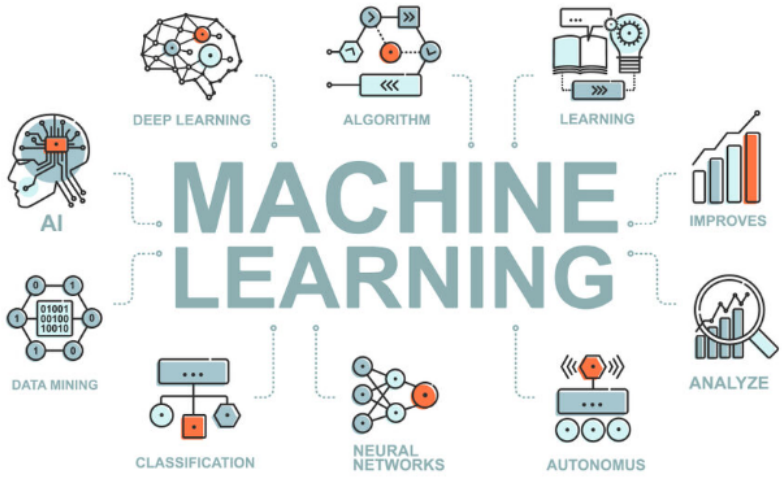

Figure 2.2: Machine Learning Importance and Types

2.2.1 Machine Learning for COVID-19

Machine learning plays a crucial role in the fight against COVID-19 by predicting new cases, developing effective strategies, and taking proactive measures to combat the spread of the virus [16]. The ability to accurately predict the trajectory of the pandemic is paramount in implementing timely interventions and resource allocation [17]. Machine learning offers a powerful tool in this regard by analyzing vast amounts of data.

By training algorithms on historical data, machine learning models can learn the underlying patterns and factors influencing the spread of the virus [18]. These models can then generate accurate predictions of future infection rates, allowing authorities to anticipate surges in cases and allocate resources accordingly. Predictive models enable decision-makers to make informed choices about implementing preventive measures such as lockdowns, testing strategies, contact tracing and vaccine distribution plans [16].

Furthermore, machine learning algorithms can leverage diverse data sources, including social media, mobility patterns, healthcare records, to enhance prediction accuracy [19]. By considering factors such as population density, demographic information, travel patterns, and the effectiveness of public health measures, machine learning models can generate localized predictions, aiding in targeted interventions at the regional level. This capability is especially valuable in combating the spread of new variants and addressing regional disparities in infection rates.

The solutions offered by machine learning extend beyond prediction alone. These algorithms can also provide insights into the effectiveness of different strategies and interventions. By analyzing large-scale data, machine learning can identify patterns in the impact of interventions like social distancing measures including masks and vaccination campaigns

[19]. This information empowers policymakers and healthcare professionals to make evidence-based decisions regarding the most effective and efficient measures to combat the virus [20].

Moreover, machine learning can aid in the development of early warning systems by integrating various data streams [21]. By monitoring key indicators such as hospital admissions, symptoms reported through digital platforms and genetic sequencing of the virus, machine learning models can detect early signs of outbreaks, allowing for swift response and containment measures [21]. This proactive approach can significantly reduce the transmission of the virus and minimize the burden on healthcare systems.

The importance of machine learning in the fight against COVID-19 cannot be overstated. Its predictive capabilities allow scientists to forecast new cases, plan interventions and allocate resources effectively. By leveraging vast amounts of data and identifying patterns, machine learning equips them with the knowledge needed to take timely action and mitigate the impact of the pandemic.

2.3 Approaches

Machine learning approaches can be categorized into two main types mentioned in section 2.2. These types are supervised learning and unsupervised learning [22]. These approaches form the foundation of many machine learning algorithms and methodologies, each serving distinct purposes in analyzing and extracting valuable insights from data [22].

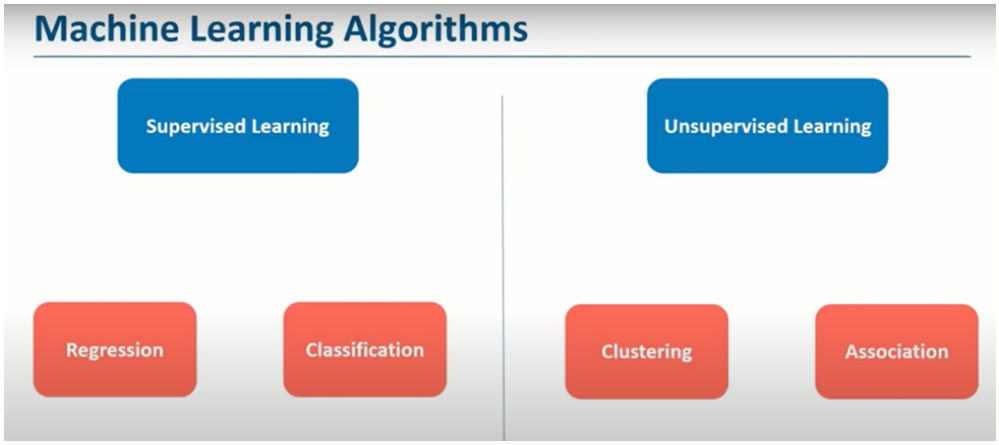

Figure 2.3: ML Algorithms - Supervised / Unsupervised Learning

2.3.1 Supervised learning

Supervised learning is a machine learning approach that involves training algorithms using labeled data [23]. In supervised learning, the algorithm learns from input-output pairs, where the input data is associated with corresponding target labels or outcomes [24]. The goal is to develop a model that can accurately predict the correct labels for unseen data.

To train a supervised learning model, the algorithm is presented with a labeled dataset that includes both input variables (features) and the corresponding correct output (target) [23]. The model learns a function that maps the inputs to the outputs. For example, in email spam classification, a supervised learning algorithm is trained on a dataset that contains a collection of emails along with their corresponding labels as spam or not spam. The algorithm analyzes the features, such as the presence of certain keywords or structural patterns, and learns to classify future emails as either spam or not spam based on these identified patterns [25].

Supervised learning can be further divided into two main tasks: classification and regression [26]. Classification algorithms are used when the target variable is categorical [24]. Some examples are classifying emails as spam or not spam. Regression algorithms, on the other hand, are employed when the target variable is continuous, like predicting home rental prices based on features like location, size, along with number of rooms [24].

Supervised learning finds applications in numerous domains, including image and speech recognition, natural language processing, and prediction in various business areas [26]. It is typically used when there is a clear relationship between the input variables and the target output and when a sufficient amount of labeled training data is available.

By training a model using supervised learning, predictions can be done on new, unseen data. A supervised learning model trained on customer data, for example, may predict whether a consumer is likely to make a purchase based on individual factors or geographical ones. The ability to make accurate predictions shows how to make informed decisions, personalized recommendations and optimize their processes.

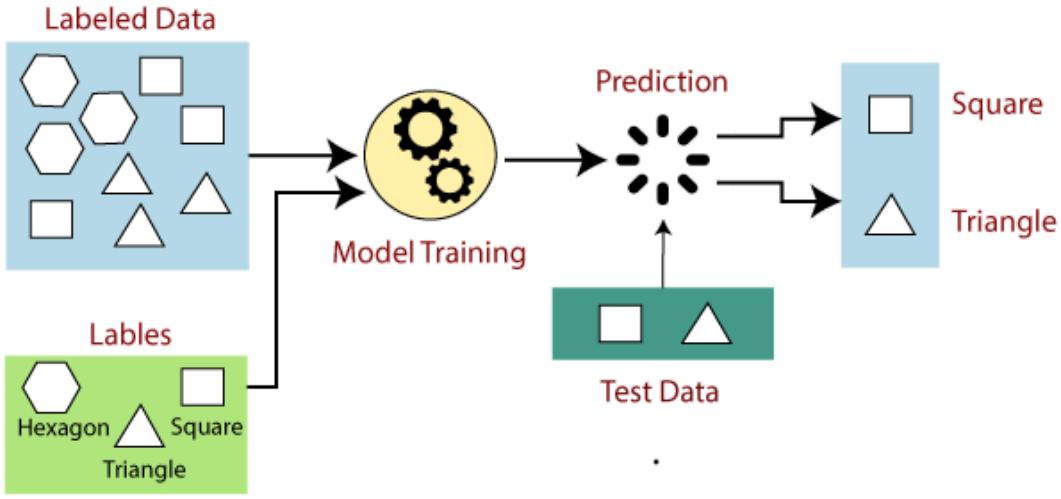

Figure 2.4: Supervised Learning Example

2.3.2 Unsupervised learning

Unsupervised learning is one of the machine learning types that involves training algorithms on unlabeled data, and aiming to discover the underlying structure, patterns, or relationships within the data [26]. It is a contrast to supervised learning, where algorithms are trained on labeled data with predefined outcomes. In unsupervised learning, the algorithm explores the data independently, without any prior knowledge or guidance [26].

A common application of unsupervised learning is customer segmentation in e-commerce [27]. By applying unsupervised learning algorithms to customer purchase histories and demographic information, it becomes possible to group customers into distinct segments based on similar purchasing behaviors or preferences [27]. Businesses may acquire insightful information from this segmentation to develop focused marketing plans and give individualized advice.

Clustering is a frequent unsupervised learning approach. Similar data points are grouped together by using clustering methods [28]. The most common ones are k-means clustering or hierarchical clustering [28]. By identifying clusters within the data, these algorithms reveal natural groupings and patterns that may not have been apparent initially [28].

Association is another fundamental technique used in unsupervised learning. Association mining aims to discover relationships and patterns in large datasets, particularly focusing on identifying associations between items or events. It involves finding frequently occurring combinations of items, also known as itemsets, and establishing associations or rules between them [28].

Unsupervised learning can also serve as a preprocessing step for supervised learning [28]. By uncovering underlying patterns in unlabeled data, unsupervised learning assists in feature extraction, anomaly detection and data preprocessing tasks [29]. By gaining insights from the unlabeled data, supervised learning algorithms can then perform better when given labeled examples. It is especially useful when there is no clear relationship between input variables and target outputs or when obtaining labeled training data is challenging [28].

One of the main challenges in unsupervised learning is the lack of a clear evaluation metric since there are no predefined labels or ground truth to compare the results to [29]. This makes it difficult to determine the effectiveness of an unsupervised learning model objectively. Researchers and practitioners often employ visualizations, statistical analyses, or domain-specific metrics to assess the quality and usefulness of the discovered patterns [29].

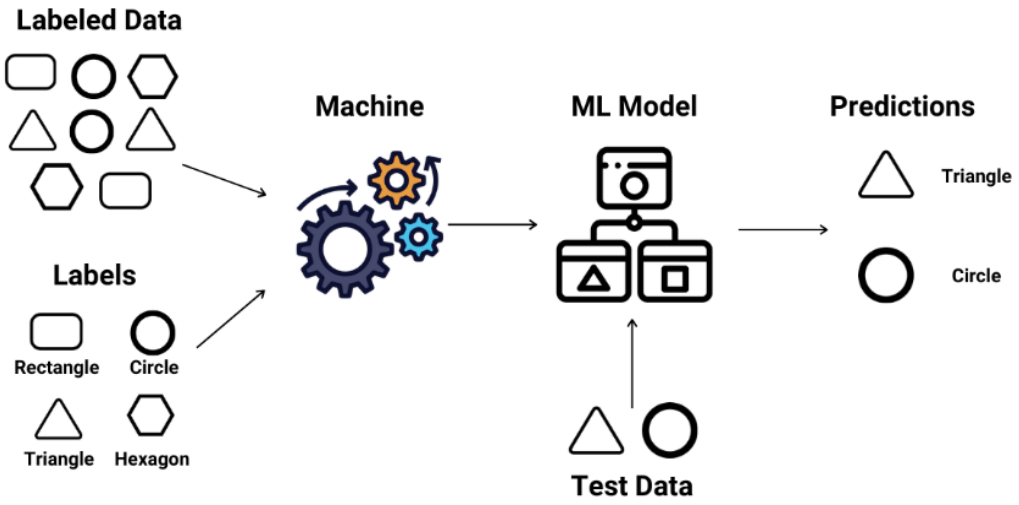

Figure 2.5: Unsupervised Learning Example

2.3.3 Other types

Other strategies have been created that don't cleanly fall into these two categories, and sometimes the same machine learning system will utilize more than one. Meta-learning and topic modeling are two examples.

Semi-supervised learning is a type of machine learning that uses partially labeled data to train models that make predictions on new, unseen data [30]. It is useful when obtaining a large amount of labeled data is difficult or expensive [30]. Neural networks and support vector machines are various algorithms that can be used [31]. Applications include image and speech recognition, natural language processing, and prediction in areas like healthcare, finance and marketing [31].

Reinforcement learning is an approach of machine learning where an “agent” learns to interact with its surroundings in order to maximize a reward [30]. It uses trial and error and a policy to choose actions that achieve the maximum expected cumulative reward [32]. Algorithms can be value-based or policy-based, and examples include Q-learning and actor-critic [32]. Applications include control systems, robotics, as well as game playing [32].

Dimensionality reduction is a machine learning technique that reduces the number of features in a dataset while retaining as much information as feasible [30]. Techniques include PCA, LDA and t-SNE, which can identify patterns or classify and visualize data [33]. It is used for preprocessing data and has applications in data visualization, feature selection and anomaly detection [33].

2.4 Models

Machine learning entails creating a model that can interpret more data to produce predictions after being trained on a set of training data [34]. Many different types of models have been used and investigated for machine learning systems.

2.4.1 Artificial Neural Networks

The biological neural networks that comprise the human brain are the basis for artificial neural networks as machine learning algorithms. They are actually made up of layers of linked neurons that process and transmit different types of information [35].

ANNs are trained to recognize patterns in data and make decisions based on that data [35]. They can be trained to recognize images, translate language or predict stock prices and many more things.

To train an ANN, a large dataset must be fed with an optimization algorithm used to adjust the weights of the connections between neurons until the network is able to accurately make predictions or classify data [35]. This exact process is known in machine learning as "training."

There are several forms of ANNs where each form is designed to do a certain task. Furthermore they are appropriate for different types of data and applications.

2.4.2 Decision Trees

Using decision trees for classification and regression problems is a form of machine learning algorithm [36]. They operate by building a model in the form of a tree structure, where each leaf node represents the ultimate prediction or conclusion, each internal node represents a characteristic or attribute, and the branches represent judgments depending on those features [36]. The method begins at the root node and divides the input into subsets according to the most crucial attribute to produce a decision tree. Until the tree is entirely developed or until a halting condition is met, this procedure is repeated at each succeeding node [37].

Decision trees handle high-dimensional data and missing values effectively, and they are simple to grasp and analyze. However, if the tree is allowed to grow too deep, they may be vulnerable to overfitting [37]. Trees can be pruned or employ strategies like cross-validation to avoid overfitting [37].

2.4.3 Support-vector Machines

A supervised machine learning technique known as a Support Vector Machine can be used for classification, regression problems, or outliers detection [38]. Finding the hyperplane in a high-dimensional space that best separates numerous classes is the objective of an SVM [38].

The way that SVMs work is to find the hyperplane that minimizes the margin or distance between the closest data points of different classes [39]. This is accomplished by figuring out the weights and bias of the hyperplane by solving a quadratic optimization problem [38]. Support vectors are the data points with the greatest influence on the location of the hyperplane that are closest to it [38].

SVMs can manage non-linear correlations between the attributes and the target variable and are successful in high-dimensional environments [40]. Additionally, they do a good job of resisting overfitting, especially when there are many more features than samples. They can be sensitive to the hyperparameters and kernel choice and do not scale well to large datasets [40].

2.4.4 Regression Analysis

Modeling the link between a dependent variable and one or more independent variables is done statistically using regression analysis [41]. Regression analysis' objective is to make educated guesses about the values of the dependent and independent variables [41].

Regression analysis comes in a variety of algorithmic forms. Linear regression, Logistic regression and Nonlinear regression are just some of the primary examples, where these algorithms use linear/nonlinear equations to data depending on the type chosen [42].

Starting with linear regression, is used to describe the connection between a continuous dependent variable and one or more independent variables [42]. The equation's formula is:

$$y = \beta 0 + \beta 1x1 + \beta 2x2 + ... + \beta nxn$$

where y is a dependent variable. Independent values are x1, x2, up to xn. Meanwhile, β0, β1, β2, up to βn are the coefficients estimated from the data. The coefficients represent how the independent variables and the dependent variable are related [43].

Logistic regression models the relationship between a binary dependent variable and one or many independent variables which is used to predict the probability of an event occurring [42].

Nonlinear regression is applied for a nonlinear relationship to represent it as a polynomial or an exponential function [42].

2.4.5 Bayesian Networks

Bayesian networks, aka Directed Graphical Models, are a form of probabilistic graphical model that demonstrates the correlations among random factors as well as the corresponding probabilities associated with them [44].

These networks are shaped like directed acyclic graphs (DAGs) with their nodes embodying the random variables, edges stand for the relationships amid those factors, and probabilities

are utilized in the form of conditional probability distributions obtained through Bayes' theorem [44]. This allows the Bayesian network to depict and coherently interpret discrepancies of events and the links between variables.

When dealing with complicated systems that have several variables and ambiguous linkages, they are very helpful [44].

2.4.6 Gaussian Processes

Gaussian processes are a kind of probabilistic model that can be employed for regression and classification purposes [45]. At their core, GPs are grounded in the concept of a Gaussian distribution, a continuous probability distribution represented by both its mean and variance [45]. When using a GP model, the outcome is assumed to come from a Gaussian distribution characterized by the mean and variance determined by the input variables [45] [46]. Additionally, it utilizes distribution's mean and variance to produce forecasts [45] [46].

GPs are attractive due to their potential to reveal complex nonlinear associations, assess the ambiguity in predictions, and conveniently incorporate prior knowledge [45]. Additionally, GPs are flexible and can be incorporated with almost any input variables.

GPs may be computationally demanding, though and they frequently perform badly when there are few training examples or numerous input dimensions [45]. To get decent results, they also need precise hyperparameter adjustment [45].

2.4.7 Genetic Algorithms

The concepts of natural evolution serve as the foundation for genetic algorithms (GAs), a category of optimization algorithm [47]. They are used to simulate the process of natural selection in order to find answers to challenging situations [47].

A population of possible solutions to a problem is randomly started in a GA. After that, the solutions are assessed using a fitness function that gauges their performance or quality [47]. Through a process known as reproduction, the solutions are chosen to create a new generation of solutions based on the fitness values [47].

Reproduction involves combining the characteristics of the selected solutions in a process called crossover, and introducing random variations through a process called mutation [48]. The new solutions are then evaluated and the process is repeated until a satisfactory solution is found or a stopping criterion is reached [48].

GAs are useful for solving problems that are difficult to solve using traditional optimization techniques, and they are particularly effective for problems with a large search space and many local optima [47]. However, they can be computationally demanding and may not scale well to very large problems.

2.5 Algorithms

Each machine learning algorithm has its own inductive bias or learning style [49]. Multiple algorithms could be suitable for a given task, and one method might fit better than others [49]. However, it is not always realistic to determine which algorithmic option would work best.

In cases like these, Microsoft's cheat sheet is the best solution where all the algorithms are listed together and show their use [49]. Another strategy would be to begin with one algorithm and then attempt the others if the results are unsatisfactory.

Although, checking the cheat sheet and trying out different algorithms are both a very effective strategy, they should not be used exclusively as the end solution. Instead, an external research and experiment is a must to guarantee a more comprehensive and effective solution.

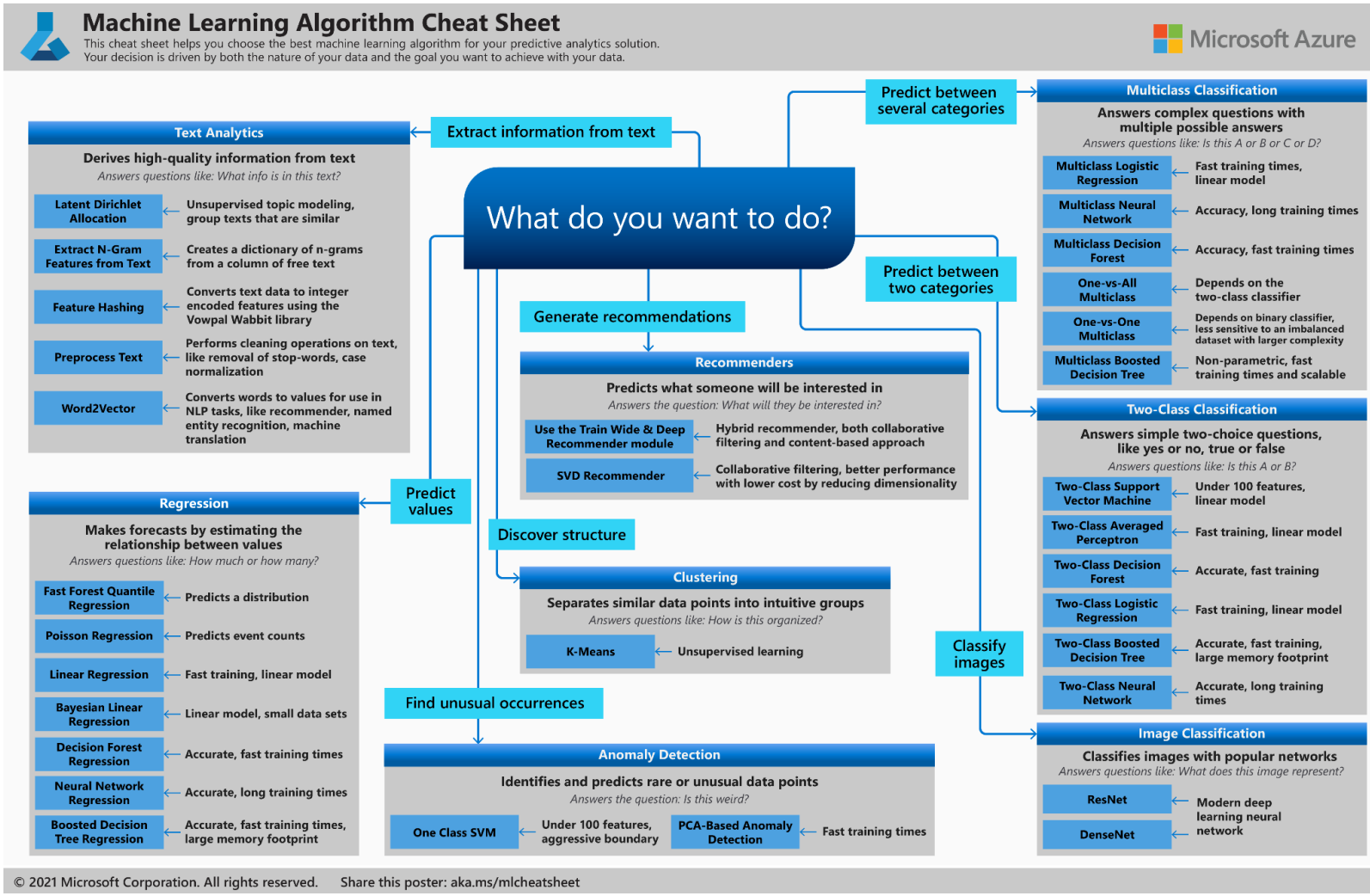

Figure 2.6: Machine Learning Algorithm Cheat Sheet

A common question is “Which machine learning algorithm should I use?” The algorithm that has to be selected depends primarily on two different aspects of your data science scenario [50]:

- “What do you want to do with your data?” Specifically, what is the business question you want to answer by learning from your past data?
- “What are the requirements of your data science scenario?” Specifically, what is the accuracy, training time, linearity, number of parameters, and number of features your solution supports?

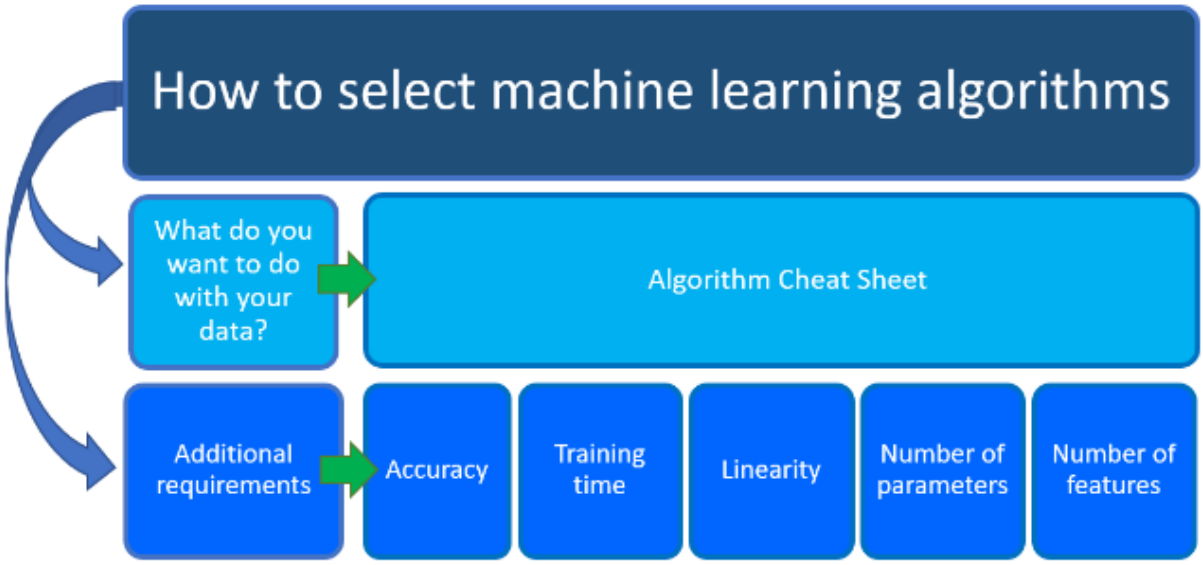

Figure 2.7: Questions regarding the selection of an Algorithm

Machine learning algorithms encompass a wide range of techniques that enable computers to learn patterns and make predictions from data [50].

Since Regression algorithms are focused on forecasts by estimating the relationship between values, they are the only algorithms that have been analyzed in this project.

Each of these methods is a different regression methodology, with unique properties and useful applications. The optimal algorithm is chosen depending on characteristics such as data type, problem complexity, desired model interpretability etc [50]. By leveraging these regression algorithms, machine learning practitioners can make accurate predictions and gain valuable insights from their data.

2.5.1 Linear Regression

Linear Regression is a fundamental approach for simulating the connection between a dependent variable and one or more independent variables [51]. It assumes a linear

relationship between the variables and estimates the coefficients to minimize the sum of squared errors [51].

2.5.2 Decision Forest Regression:

Decision forest regression is an ensemble learning method that combines multiple decision trees to make predictions [52]. It splits the data into subsets and constructs individual decision trees. The final prediction is the average or weighted average of the predictions from all trees [53].

2.5.3 Boosted Decision Tree Regression:

Boosted decision tree regression is another ensemble learning technique that combines multiple decision trees [54]. It iteratively builds weak decision trees, giving more weight to the instances that were previously misclassified. The final prediction is the weighted sum of the predictions from all trees [54].

2.5.4 Fast Forest Quantile Regression:

Fast forest quantile regression is a variant of random forest regression that focuses on estimating quantiles of the target variable [55]. It uses an ensemble of decision trees and provides a range of predictions rather than a single point estimate [55].

2.5.5 Neural Network Regression:

Neural network regression involves using artificial neural networks to model the relationship between “inputs and outputs” [56]. These networks consist of interconnected nodes or "neurons" organized in layers [56]. The network learns the relationship between the inputs and targets through training with labeled data [56].

2.5.6 Bayesian Linear Regression:

Bayesian linear regression algorithm is used to ascertain the prior probability for the model parameters rather than to identify the one "best" value of the model parameters. [57]. It incorporates prior knowledge about the parameters and updates the belief using observed data to obtain a posterior distribution [57].

2.5.7 Poisson Regression:

Poisson regression is used when the dependent variable follows a Poisson distribution [58]. It models the relationship between the independent variables and the count data, estimating the parameters that maximize the likelihood of the observed data [58].

2.6 Proprietary Software

Currently, two major cloud computing platforms stand out as widely used software for predictions are **Microsoft Azure** and **Amazon Web Services (AWS)**.

These platforms provide comprehensive suites of services, including robust machine learning capabilities, making them go-to choices for data scientists or organizations seeking to develop and deploy predictive models [59].

Azure and AWS offer a wide range of tools and services specifically designed for machine learning tasks. These platforms provide scalable computing resources, efficient data processing and extensive libraries of pre-built models and algorithms. With these features, data scientists can easily build, train and deploy predictive models for a variety of applications, including forecasting, recommendation systems, fraud detection, etc [59].

Both Azure and AWS provide a rich ecosystem of services that complement their machine learning offerings. This includes data storage and management tools, data visualization capabilities, integration with other cloud services, and robust deployment options. These platforms enable organizations to efficiently manage their data pipelines, experiment with different models and seamlessly deploy predictions into production environments.

2.6.1 Microsoft Azure

Microsoft Azure is a cloud computing platform that provides a variety of services. One of them is machine learning. It provides a range of tools and features to create, train and deploy machine learning models [60].

Utilizing the strength of the cloud to train and deploy machine learning models at scale is one of the main advantages of utilizing Azure Machine Learning [61]. Users of Azure can quickly access a variety of computational resources, such as CPUs, GPUs, and specialist machine learning gear [61]. Due to this, complicated models may be trained fast, even on huge datasets.

The vast array of ready-made models and algorithms offered by Azure Machine Learning is another benefit [62]. These models can be adjusted to fit the specifications of a given application or utilized as a starting point. Following that, it offers a variety of tools for visualizing and comprehending the outcomes of machine learning trials, which assists in making defensible conclusions regarding the next step [62].

Other services, such as tools for data storage and administration, as well as for deploying and scaling machine learning models in production contexts, are helpful for developing machine learning applications in addition to the fundamental machine learning capabilities [62].

Azure Machine Learning can be used to forecast new COVID-19 cases in addition to its general machine learning capabilities. Precise models are created for predicting the spread of the infection and responsible judgments by utilizing the platform's robust capabilities and functionalities. By allowing processing of large volumes of COVID-19 data with its scalable compute resources, the platform enables the efficient processing of this data, including feature engineering and data cleaning. A big variety of algorithms are there to be used and build predictive models and return accurate predictions.

Additionally, Azure Machine Learning offers tools for model validation and evaluation [60]. Data scientists can measure the effectiveness of their models using a variety of metrics. To increase the precision of the forecasts, the models can be modified in accordance with the findings of the evaluation and repeat the procedure.

Once the predictive model is trained and validated, Azure Machine Learning enables seamless deployment and scaling in production environments. This ensures that the model can be used to make real-time predictions on new COVID-19 data as it becomes available.

Figure 2.8: Microsoft Azure Logo

2.6.2 Amazon Web Services

Amazon Web Services (AWS) is a cloud computing platform just like Microsoft Azure which offers several possibilities for machine learning applications. AWS provides a range of tools, frameworks, models to help developers and data scientists quickly and economically create, train and deploy ML models [63].

One of the most popular ML tools offered by AWS is Amazon SageMaker, a fully managed service that develops, trains and deploys ML models [64]. SageMaker provides pre-built methods like linear regression, k-means clustering and deep learning to quickly construct ML models [63][64]. Own methods and frameworks, like TensorFlow or PyTorch, must also be supplied and these models must then be trained at scale via distributed training [64].

Amazon Comprehend is a service for natural language processing that enables extracting insights from text data, and is another important ML tool provided by AWS [65]. Text may be analyzed for sentiment, entities, key phrases and more with Comprehend [65]. Applications like customer service, social media monitoring, as well as content analysis may all benefit greatly from this.

There are also more tools such as, Amazon Rekognition that can recognize objects, scenes or people in images and videos, Amazon Personalize which is a recommendation engine service that is able to build personalized recommendations for users based on their browsing and purchase history or even Amazon Polly and Amazon Transcribe, a text-to-speech service that allows converting text into lifelike speech or transcribe audio and video files into text [66][67][68][69].

Figure 2.9: AWS Logo

2.7 Website development Software

In the realm of website development, there are several frontend and backend software options available. Some examples are JavaScript, React, Angular or Vue.js as a frontend technologies and C#, Node.js, Ruby, or Django for backend. Among these options, Blazor and the ASP.NET Framework have been selected as the preferred technologies for the frontend and backend development of the website, respectively.

These choices were made based on several factors, including their compatibility, ecosystem support and development efficiency. But other than these factors the most important one is that Blazor and ASP.NET Framework are both technologies developed by Microsoft which means there is an ease of use to connect components or other elements between them.

2.7.1 Blazor

Blazor is a free and open-source framework created by Microsoft that lets programmers to create interactive and dynamic web applications using C# [70]. Lately, the developer community has paid major attention to Blazor, a potent and adaptable web framework. The distinguishing feature of Blazor is its capacity to run Razor views directly on the client-side, doing away with the necessity for server-side rendering [71].

The name "Blazor" is derived from the combination of the words "Browser" and "Razor" [71]. Razor is a well-known .NET HTML view generating engine. By incorporating the Razor syntax within Blazor, developers can seamlessly blend C# code with HTML markup, allowing for a streamlined development experience [71].

Blazor's capability for creating Single Page Applications (SPAs) is one of its primary features. SPAs are web apps that only need to load once, but dynamically change their content in response to user interaction [70]. This makes using them more responsive and fluid. It is used to build SPAs that are fully browser-based and take advantage of cutting-edge web technologies just like WebAssembly [72].

Blazor bridges the gap between client-side and server-side programming by allowing the execution of C# code directly in the browser by utilizing the capabilities of WebAssembly [72]. This method has several advantages, suchlike decreased network latency, higher performance, and enhanced scalability [73]. Developers can design reusable UI components containing both the UI layout and related functionality using Blazor's component-based architecture. These parts are simple to put together and link, which encourages code reusability and maintainability.

Figure 2.10: Blazor Logo

2.7.2 ASP.NET Framework

A proprietary software framework created by Microsoft that predominantly runs on Microsoft Windows is known as the ASP.NET Framework [74]. It offers language interoperability, each language may utilize code written in other languages, across numerous computer languages and contains a sizable class library called Framework Class Library (FCL) [74]. The Common Language Runtime (CLR), as opposed to a hardware environment, is the environment in which programs created for the .NET Framework run [74]. Security, memory management and exception handling are just a few of the functions offered by the CLR, an application virtual machine.

As a result, "managed code" refers to computer code created using the .NET Framework [75]. Together, FCL and CLR make up the ASP.NET Framework [75]. The user interface, data access, database connectivity, cryptography, creation of web applications, numerical algorithms, and network communications are all provided by FCL [75]. By merging their source code with the ASP.NET Framework and other libraries, programmers create applications. The framework is anticipated to be used by the majority of new apps created for the Windows platform.

Microsoft also created Visual Studio, an integrated development environment for .NET apps. .NET Framework began as proprietary software, although the firm worked to standardize the software stack almost immediately, even before its first release [74]. Since then, Microsoft has changed .NET development to more closely follow a contemporary model of a community-developed software project.

Figure 2.11: ASP.NET Logo

2.7.3 Integration with Cloud Software

Tools and APIs in particular are there in each cloud provider to integrate a web application with cloud software for predictions. For instance, whereas Amazon Web Services (AWS) offers AWS SDKs and AWS API Gateway. Whereas, Microsoft Azure offers Azure Machine Learning SDK and Azure REST APIs.

HTTP queries are sent to the cloud service endpoints from the backend of any web application using these cloud services. Setting up API endpoints, establishing authentication with the cloud provider and managing data sharing securely are all part of the integration process.

Web applications with user-friendly interfaces and reliable server-side logic are built by integrating backend development tools such as C#, Node.js, Ruby, or Django together with frontend development software such as HTML/CSS, JavaScript, React, Blazor, Angular or Vue.js. Enhancing the application's capabilities through integration with cloud prediction software facilitates the use of machine learning and provides accurate predictions.

Chapter 3

3. System Description & Requirements

The system being developed is a Machine Learning model that is trained constantly by the user and a web application that allows users to predict Covid19 Cases for 44 European Countries only for the future (whatever the real current day is). Because the system will leverage machine learning technology, it will require an intermediate software to perform training and give results according to the algorithm used. The application operates by interacting with a machine learning model that has been trained to predict COVID-19 cases mainly based on 3 factors: Country, Population and future Date. Furthermore, the system shows transparency and simplicity to all users with a click of a button to display the result.

A frontend and a backend are the standard components of traditional web applications. The predictions will be rendered and shown in this scenario using Blazor as frontend and ASP.NET for the backend. The backend, which maintains the data and methods required to produce predictions, will still communicate with the machine learning model through an endpoint gateway.

Azure Machine Learning services are utilized asynchronously to generate predictions about COVID-19 cases and require an instance of the HttpClient class to send a POST request to the Azure ML service. In this circumstance, an instance of the HttpClient class is recommended since it is a quick and reliable way to send HTTP requests from ASP.NET applications. In order to process HTTP requests and responses, Azure provides a versatile API. Moreover, it has capabilities like connection pooling, which can boost efficiency by leveraging previously established connections to the same host. Using HttpClient also allows handling of errors easily that may occur during the request.

The sections below present the model theoretically and how the system will operate and additionally create a comprehensive list of the requirements that this system should satisfy will be examined.

3.1 Theoretical Model

An overview of how users will interact with the system is provided in figure 3.1 as a high level view approach. The system's business logic and how users can predict Covid Cases accurately through Azure is explained in depth in the following chapters.

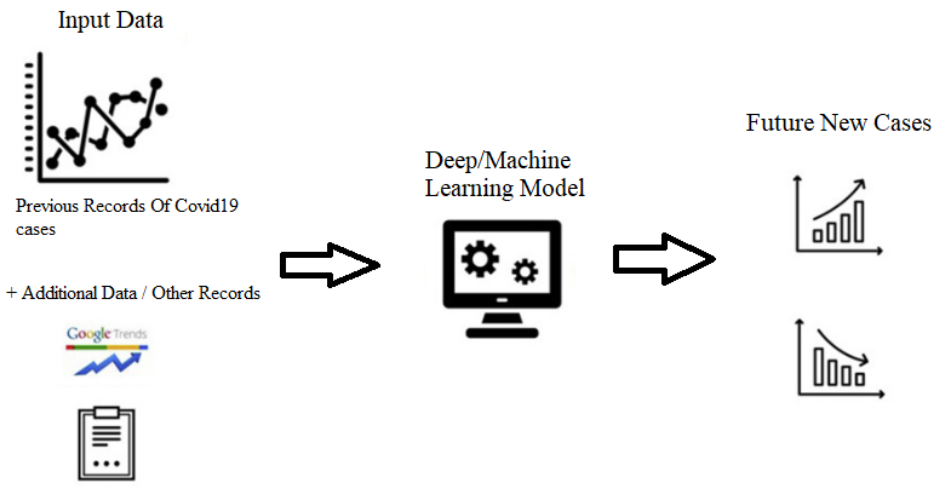

Figure 3.1: Theoretical Model of the system's business logic

- Provide to the system a data file
- Use an algorithm to train the cleaned model
- Return the estimations in Azure Machine Learning Studio
- Then "web service input" request by the user done in the Blazor application
- Return the final result (response)

3.2 Requirements & Analysis

The system specifications outline the planned characteristics and functions of the system. Any application must begin with a clear set of criteria since it defines the system's ultimate goal. The system being designed must ensure that its functionalities are appropriate, and sufficient. Thus, it should deliver a decent user experience.

Requirements are divided into two categories: functional and non-functional, and this section will provide an overview of each before going deeper in following sections [76].

3.2.1 Functional Requirements

The criteria that the end user expects the system to meet in terms of functionality are represented by functional requirements [77][78]. The services that a system delivers to its users are often specified by functional requirements [78].

Regarding the machine learning covid predictions, the system supports any type of user but is especially useful for Governments or Health sector managers so it should stay confidencial. The list below depicts these requirements:

- ***Build & Train Model:*** Use Machine learning pipelines to create a workflow, so it can clean the data, transform it, and split it into training, validation, and testing datasets.

- ***Use the "Perfect" Algorithm:*** Out of a quite big range of algorithms, one of them must be selected to return the best prediction accuracy.
- ***Evaluate Model:*** After training, the performance of the model can be evaluated on the validation and testing datasets to ensure that it is accurate and generalizable.
- ***Deploy Model as Endpoint:*** Deploy the model as an endpoint to have a connection between Azure and the app, so that the web service inputs and outputs are communicated through API calls.
- ***Select inputs:*** Users will need to provide inputs such as country and date to interact and get back results.
- ***Get Prediction:*** After a request for a prediction, the user should be able to receive it and be displayed on the UI.
- ***Save Prediction and inputs:*** Finally, after getting the prediction, the user should be able to save that prediction.

An actual web application delivered to the government or health institutions would have more functional requirements. Despite this, the project's limited scope nevertheless delivers an end-to-end solution with functional points and accurate results.

3.2.2 Non-Functional Requirements

Since Functional Requirements are visible functionalities, then Non-Functional Requirements have attributes that provide quality metrics to a system [79]. When analyzing a software program, factors like portability, security, maintainability, and other non-functional features that are crucial to its success are taken into consideration [79].

The entire system will be unable to meet user needs only if all the requirements mentioned previously fail to be implemented. Three of the most significant requirements are listed below.

-Usability: The main objective of the project is to create an accurate system and appealing application to be simple to use. The user interface design requires a lot of consideration in order to satisfy the usability standards for performance and quality. Additionally, the functionality made available should be simple to access and other situations like running into loops should be rigorously avoided. If errors occur, the system will provide users with adequate feedback to let them learn from their mistakes.

-Maintainability: The system should be simple to maintain in order to make problems and issues easier to handle. A clean, well-organized code is the best technique. Afterwards it is easier to find the relevant part of code or address errors and add new features.

-Scalability: Future implementations will be anticipated, thus the system will be constructed to allow automatic trained predictions. The necessary infrastructure should be in place to make this project expandable in a number of ways that enhance prediction accuracy.

Chapter 4

4. Project Planning

Every element will be considered throughout the project planning stage of development in order to make some essential decisions about the project's future. During the software development lifecycle, selecting the process model to be employed was perhaps the most important decision. Information regarding the complexity of the project as well as its time estimation will be provided. The risk management's final step will be to identify and assess all potential risks and problems that could occur throughout development. All of these data will be carefully taken into account while structuring a time plan to complete every requirement of the project.

4.1 Software Development Models

Software development is sometimes a complex process, it can be tricky to approach and measure the performance of any project. A software development method is necessary to tackle these challenges. SDPs offer a method for project planning and meticulous step-by-step stage scheduling [80]. Some software development methods are more convenient than others. It usually depends if these objectives are well defined and resilient [81]. This section will focus on the following development methods since they are the most influential and have each served as the foundation on software development.

The **waterfall model** is one of the most popular software development processes that uses an organized, sequential approach [82]. When project requirements are clearly specified and unlikely to alter dramatically throughout the course of development, it is frequently employed [82]. The waterfall model divides the development process into distinct phases that must be finished in a linear manner, with each phase acting as the starting point for the following [83]. Gathering requirements, analysis, system design, implementation, testing, and maintenance are common phases. Because of this, it is a very strict strategy that does not provide any flexibility or correction.

The **iterative model** uses an alternate strategy that stresses repeated cycles of development. It requires a number of iterations, each of which includes every stage of the waterfall model [84]. Each iteration involves the development team creating a functional version of the program, receiving feedback and then incorporating it into following iterations [84]. This approach offers more flexibility and can adapt to changing needs.

The **incremental model** focuses on delivering the software in increments [85]. It divides the project into smaller, manageable portions that are developed and delivered incrementally [85].

Each increment consists of a complete development cycle, including analysis, design, implementation, and testing. This model allows for early and continuous delivery of functionality, enabling stakeholders to provide feedback and make adjustments throughout the development process [85].

The **agile model** is a highly flexible and team-based process [86]. It places a strong emphasis on client participation, iterative development and adaptable planning [87]. Delivering functional software in brief cycles known as sprints is a priority according to agile approaches like Scrum and Kanban [86]. The development team continually engages in stakeholder collaboration, adjusts to shifting requirements, and concentrates on producing value for the business [87]. Transparency through continuous progress and cooperation are encouraged by agile approaches.

The **spiral model** is a combination of the iterative development and the waterfall approach [88]. It uses an iterative planning, risk analysis, development and assessment cycle method that is risk-driven [88]. A phase of the development process is represented by each iteration in the spiral model, and each cycle includes ever more intricate and polished versions of the software. The spiral approach provides for regular evaluation and adaptation during the project and facilitates risk management [88].

4.2 SDP Decision & Time Plan

For the COVID predictions project, it is decided to utilize Azure Machine Learning to develop accurate predictions. To ensure a systematic and efficient approach, the Waterfall Model is selected for the project management methodology. The Waterfall Model is a linear and sequential approach as explained above that makes it ideal for this project that has well-defined requirements and a clear understanding of the end goal, which is also worked by an individual developer.

Precise requirements and goals of the COVID predictions project are defined and the key features, data sources, and variables needed are identified for accurate predictions. A long period of research is required not only to continue to the next stage but to get familiar with the technologies being used. While following the analysis and implementation of the Azure Machine Learning model there is another workflow that has to happen simultaneously. The web application built in Blazor and ASP.NET requires functionalities to be created too so that the models can be checked whether they run successfully or not. Testing and maintenance were the last steps completed to verify their usability, configuration, efficiency, accuracy etc.

At the end of the project, the web application will be easily accessible and real-time predictions will be obtained as an end-to-end product where the estimations are done by Azure Machine Learning.

Time Plan:

October - December : The time stage where the most part of the research is conducted. A literature review on ideas pertinent to the project is concluded at these 3 months time. Each month will have a blueprint on how to proceed on a weekly basis and along with a plan for risk analysis and management. The creation of a theoretical model will be the last step to initiate the list of criterias for the other stages.

January: Configure the cloud and web application environments and provide a chance to familiarize oneself with the new technologies.

February: The cloud setup will take place to start creating a model flow that can predict the awaited covid cases and then evaluate their validity and accuracy.

March - April: Proceed and conduct a detailed analysis, design, and implementation of the web application. Most of the criterias should be satisfied by the conclusion of this period.

April - May: This is the last stage that will serve as a way to connect the cloud results to the web application. Moreover, the frontend will be finalized and accurate predictions will be possible to be displayed with whatever parameters the user decides to choose.

4.3 Risk Management

The risk management phase of the design process is critical. The complexity of the project shows that risk management is an important factor to take into account due to the probability of problems that could arise. Its main objective is to identify and evaluate any potential risks that might endanger the development of the system and application [89]. Each risk will be evaluated based on its chance of occurrence and the overall impact it would have if it actually occurs. The two taken together will give the exposure of that particular risk, which measures its overall significance. Additionally, it will be sought to present practical and reliable methods and techniques of countering particular challenges in the case of their occurrence.

Table 4.1 shows the project risks that have been identified. The development process may run into different types of risks. Additionally, each risk is described in greater detail using tables, along with a definition and a strategy for how it will be mitigated.

There are three types of metrics to assess the risk of a project, probability, impact and exposure, which is a mathematical formula that comes from the metrics aforementioned.

Probability shows the possibility of a risk occurring before or after a specific action; presented as percentage, 0 to 100%.
Impact displays the potential severity of the effects that a certain risk may have when it materializes; values are scaled from 1 to 5, with 1 denoting minimal effects and 5 denoting substantial effects.

Exposure displays the specific risks that the project is exposed to and appropriately prioritizes them. The product of probability and impact establishes the value of exposure. Threats are ranked using this number as a metric. The higher the exposure, the higher the priority of the risk is.

Risk Nr.	Description	Impact	Probability	Exposure
1	Lack of experience with new technology	5	80%	4.0
2	Change / Misunderstandment of requirements	4	25%	1.0
3	Project size underestimation	3	50%	1.5
4	Issues with Time Management	4	60%	2.4
5	Failure of final system's deployment	5	30%	1.5
6	Hardware breakdown	5	20%	1.0

Table 4.1: Risk Exposure Calculations

Risk	Mitigation Strategy
Lack of experience with new technology	Assign a specific time range to conduct research, read documentations regarding the technology being learned and practice regularly to become comfortable.
Change / Misunderstandment of requirements	A selection of a software development method should be done in the first stage of the project.
Project size underestimation	Based on the product/project being worked on there must be a good time arrangement and requirements "road" map to complete them on time.
Issues with Time Management	The most important features should be prioritized in order to put in more effort as soon as possible.
Failure of final system's deployment	Allocate time for deployment and testing of the system or application to contemplate what could or has gone wrong, so there will also be time to fix this issue.
Hardware breakdown	The best solution for this risk is having backups in the cloud or external storage such as a hard drive or USB.

Table 4.2: Mitigation Strategy regarding the Risks Identified

Chapter 5

5. Azure Machine Learning

Azure Machine Learning is a cloud-based environment used for training, deployment, automatizing, management, and tracking of machine learning models [90]. One of the biggest advantages it gives to its users is the possibility to be used for any kind of machine learning [90]. Living in this big data era, it is highly valuable to learn and be able to use machine learning for our projects. Azure provides tools that can be used not only by professional data scientists. It means that with basic knowledge and understanding of probability and data processing even a non professional data scientist can create models.

5.1 Azure Services

Machine Learning is one of the main fields that Azure Portal offers to its users. A variety of tools and services are available through the Microsoft Azure Machine Learning, including:

Azure Machine Learning Workbench - an application that manages the key functions of a machine learning project, such as data preparation and import, model development, experiment management and model deployment across several environments.

Azure Machine Learning Experimentation Service - a service that facilitates communication with Workbench, project management, access control, and version control.

Azure Machine Learning Model Management - a service that helps managing, storing, registering or even processing models.

Visual Studio Code Tools for AI - an application that allows creating scripts for machine learning tests.

Azure Machine Learning Studio - a tool that aids in the development and use of predictive analytic models. [91]

5.2 Azure Machine Learning Studio

Microsoft's Azure Machine Learning Studio is a platform used to create, train, test and deploy predictive analytics models. It aids in resolving analytical issues brought on by a piece of data. There is the option to use it to convert, analyze data, carry out various data manipulations, and produce results [92].

The collection of data is necessary in order to create a predictive analytic model. The Azure client may upload their own collection of data from the local system or choose from a huge

number of data sets provided by the program. The data can be altered as necessary to satisfy the requirements of the experiment. A number of modules of the program are used to analyze data and provide outcomes [92].

In general, Machine Learning Studio has an interactive and visible workspace that facilitates building, testing, and model manipulation. It works by connecting several components by dragging and dropping them. The number of ports can be varied depending on the module [93]. The majority of them, nevertheless, are designed to be pre-set as needed. The user can concurrently control every module. The module's settings are all programmable.

Additionally, if that particular module and stage of the experiment permits a middle result of the data, the user has access to visualization in that area [92] [93]. Figure 5.1 presents a depiction of the key ideas.

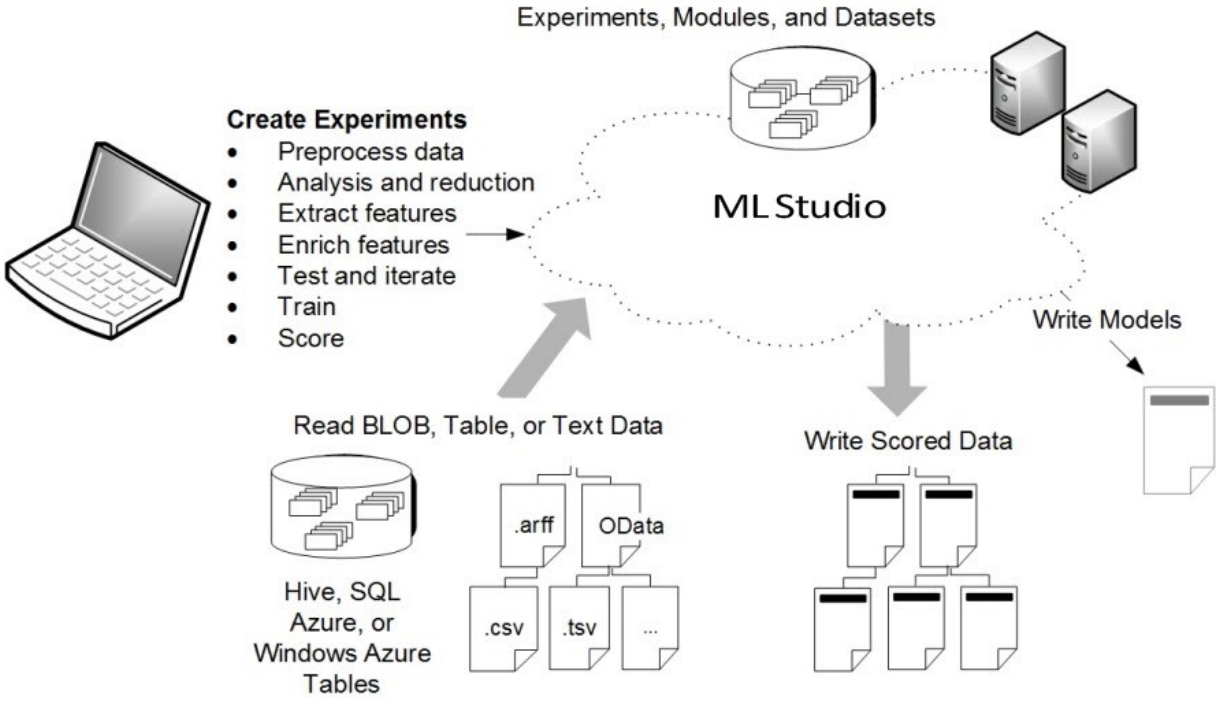

Figure 5.1: Azure Machine Learning Studio

5.2.1 ML Studio Experiment

Data sets and modules are components of an experiment. They should be connected in the most relevant way possible in order for a predictive analytic model to be built.

Dataset

A dataset is data that has been uploaded to ML Studio to be used in the modeling process. The application gives two choices: upload a data set from the local workstation or utilize data set samples, which are primarily used for training and developing software knowledge. A

dataset can be renamed, added a description, and select the format before uploading it [92] [93].

Modules

An algorithm applied to data is referred to as a module. The components of Azure Machine Learning Studio range from statistical operations to training, scoring, and validation procedures. On the left side of the canvas, there is a selection of modules from which are selected while creating the experiment [92] [93].

A module contains a set of parameters that can be used to configure and adjust the module's internal algorithm [92]. Any model can be personalized so the parameters should be modified to adapt the project's requirements.

If an experiment satisfies each of the following requirements, it is deemed legitimate and properly conducted [92]:

- ➔ Each experiment contains a minimum of one module and one data collection.
- ➔ Data sets are connected only to modules
- ➔ Each pipeline port must have a connection to data flow unless its the last one
- ➔ Modules have the required parameters sets
- ➔ No errors during the compilation

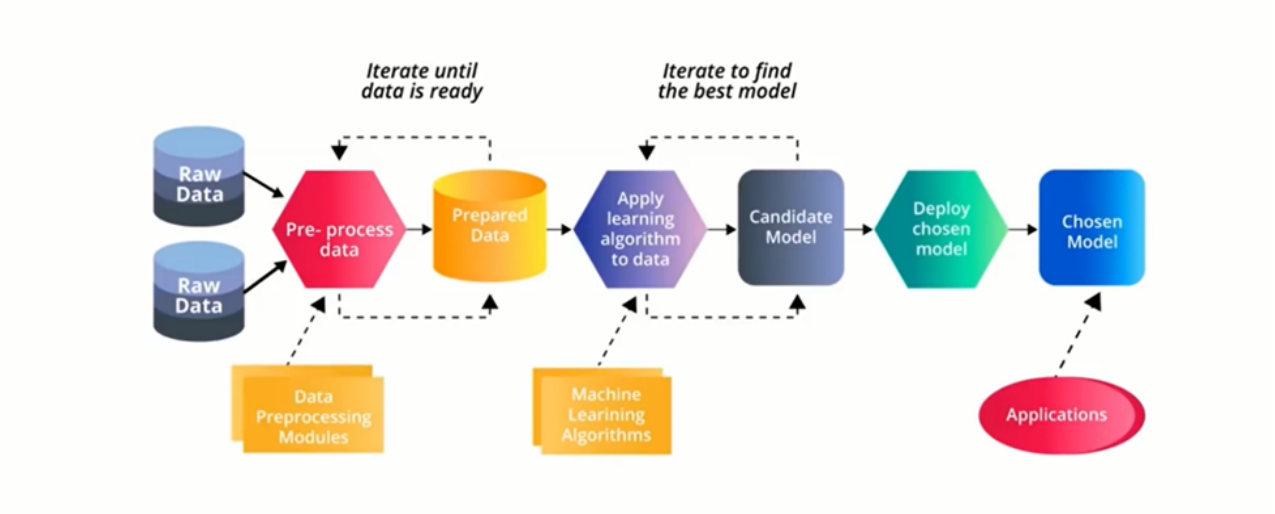

Figure 5.2: Various Processes in ML Life-Cycle

5.2.2 ML Studio web-service

Machine Learning Studio web service enables deployment of the predictive analytic solution as a web service. It establishes a real-time connection between external apps and the workflow scoring model of Machine Learning Studio. An external application receives prediction results

after making a request to a web service provided by Machine Learning Studio. An API key that was generated when the web service was launched must be supplied when making a request to a web service [92].

To deploy the model the following steps must be satisfied:

- ➔ A Training Experiment must be existent
- ➔ Create inference pipeline (real-time or batch)
- ➔ Deploy the model through AKS compute (Kubernetes)

An API command becomes available when the web service is deployed. It is a command of the post type and invoking it needs attributes. The experiment itself determines the kind and quantity of qualities. It is possible to make requests to the azure web service and receive responses after it has been deployed [92].

5.3 Machine Learning Process

Machine learning is a difficult process. It is divided into 4 main sections. A machine learning process must initially be launched after gathering the necessary data. They are usually referred to as raw data. The following stage is pre-processing. This step is essential since data isn't always clean.

It could have certain values that are missing or irrelevant to the experiment. Data is divided into two groups at this stage:

- the first random split group of data gets used to run the algorithm
- the second one is used to evaluate the model that has been trained by the algorithm.

A machine learning algorithm must be selected once the data has been filtered. Finding the ideal algorithm is not always simple. In order to find the algorithm that best achieves the objectives, a variety of algorithms is frequently evaluated on the same data and then the results can be compared. The model will next be constructed by being tested and validated. There must be further repetitions of this technique. Deploying the model is the last stage. Once this is completed, the model will be able to forecast. Figure 5.3 and 5.4 show the procedure in a step-by-step form. Whereas, figure 5.5 is the full representation of how ML Studio will help this procedure.

An essential tool for accelerating the entire machine learning process is Azure Machine Learning Studio. It offers a variety of tools that are put into use to make all the aforementioned stages more efficiently executed.

Figure 5.3: Very Simple ML Process

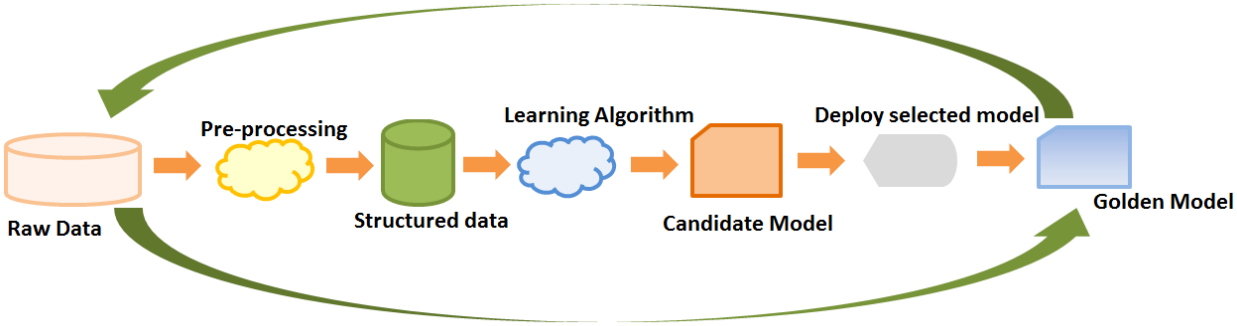

Figure 5.4: ML Process Life-Cycle

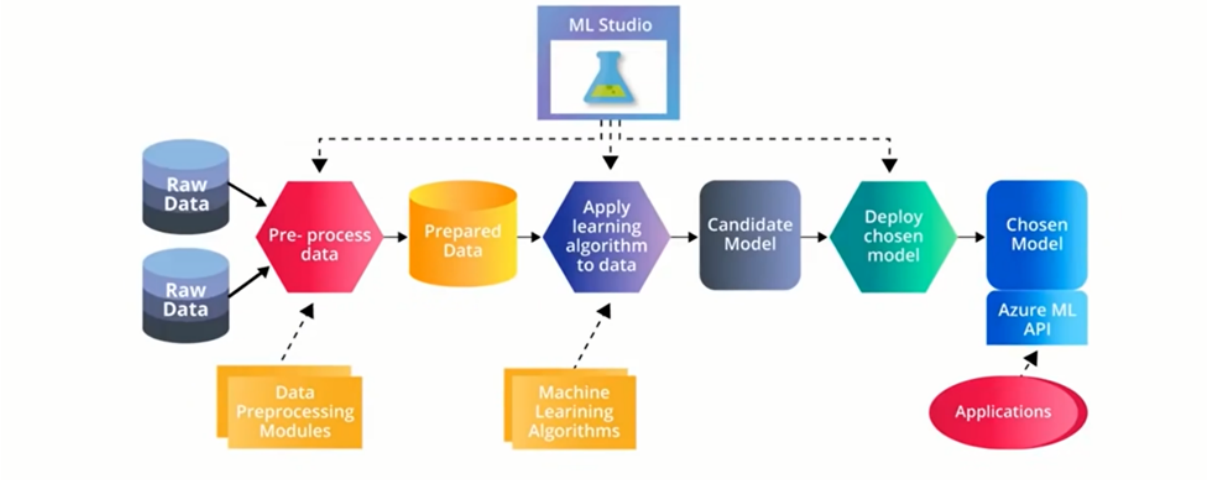

Figure 5.5: Microsoft Azure ML Studio Process

Chapter 6

6. Azure Setup & Predictions

In this chapter, the entire ML lifecycle process (MLOps) with Azure Machine Learning is covered comprehensively, providing a detailed understanding of how to manage and deploy machine learning models using the Azure Machine Learning platform. Azure Portal, ML Studio and Documentation are used to complete this project [94][95][96].

6.1 Workspace

To get started with Azure Machine Learning, users often begin by establishing a workspace on the Azure portal. This workspace offers a variety of tools for monitoring and tracking machine learning operations as well as acting as a single area for organizing data, models and experiments [97].

Users can start uploading and prepping data for analysis using tools like Azure Data Factory or Azure Databricks after a workspace has been set up. Then, they can create and train machine learning models using Azure Machine Learning's drag-and-drop interface, or they can take use of more sophisticated tools like automated machine learning to quickly iterate on various model architectures and hyperparameters [97].

For the Covid Predictions project, a workspace was created as follows with the name ‘azurecovidpredictions’. To login to an Microsoft Azure Machine Learning Studio there is a must to have a Microsoft account and start by creating an account with an Azure Subscription, with 200$ credit to use in 30 days for free and thereafter paid subscription where every tool and instance used has to be paid.

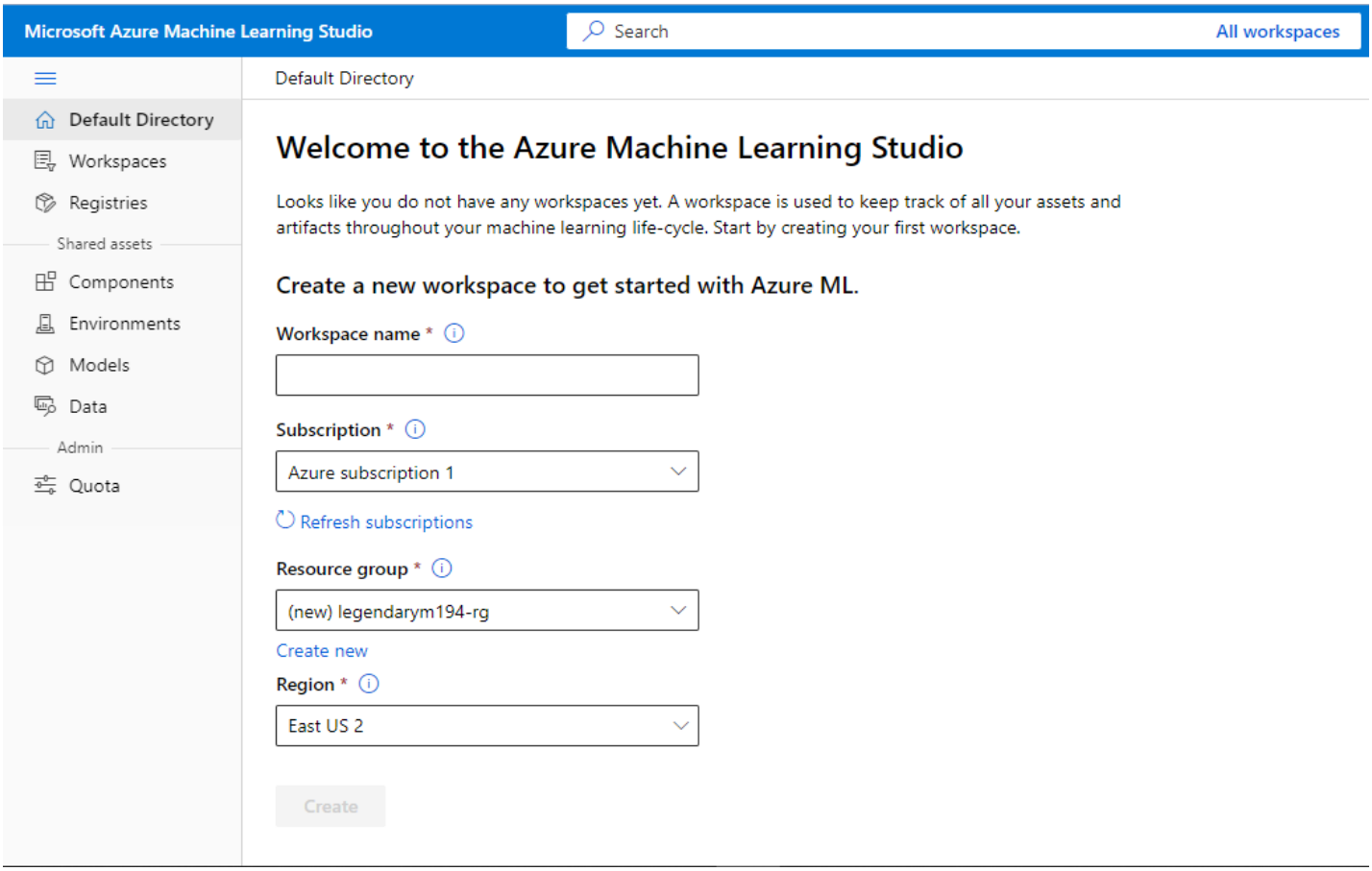

Figure 6.1: Workspace Creation

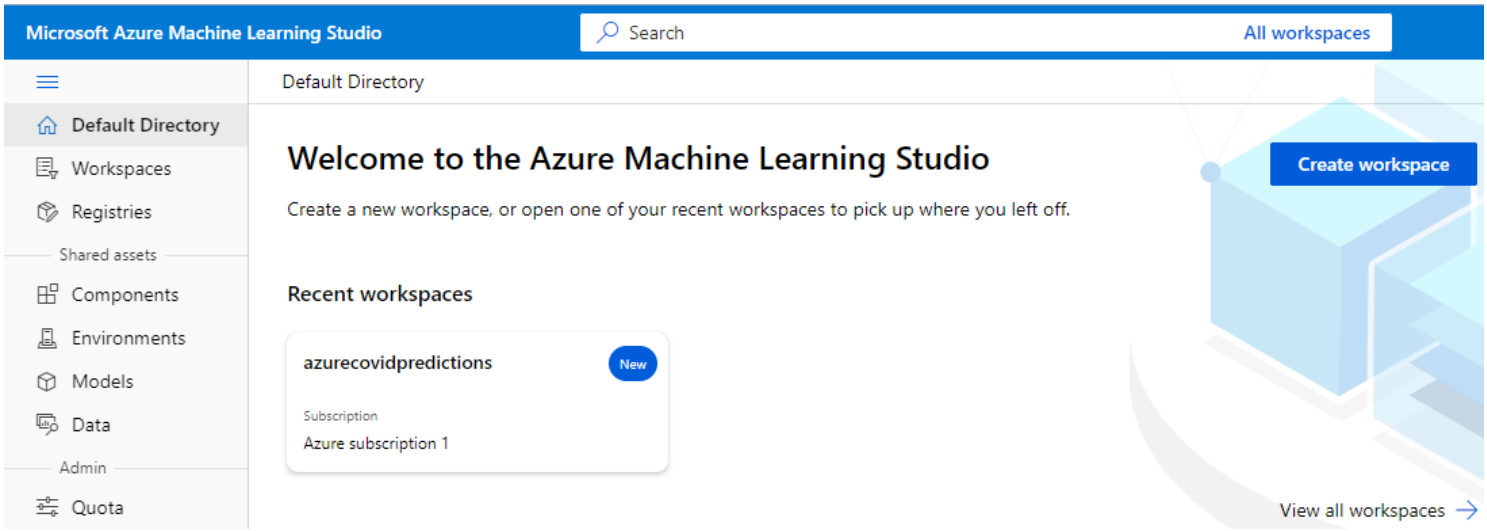

Figure 6.2: Default Directory

6.2 Compute

Compute Instance

Data exploration, data science, and other computationally demanding operations may be performed on Azure Compute Instance, a fully managed cloud-based virtual machine. The Compute Instance offers an environment ready to use with a couple of selection of data science tools and frameworks. Python, TensorFlow, R, and Jupyter notebooks are the four examples. Compute Instances are the best choice for short-term workloads since they can be easily generated and destroyed and are paid depending on use [98].

The Compute Instance is created for the short-term workload that this project requires. The virtual machine type selected is CPU for fast calculations and the specific VM is Standard_D2_V3 which costs $0.10/h. Performance and reliability may be enhanced as a result of ensuring that the computing resources required for COVID-19 forecasts are kept separate from other workloads operating on the same system.

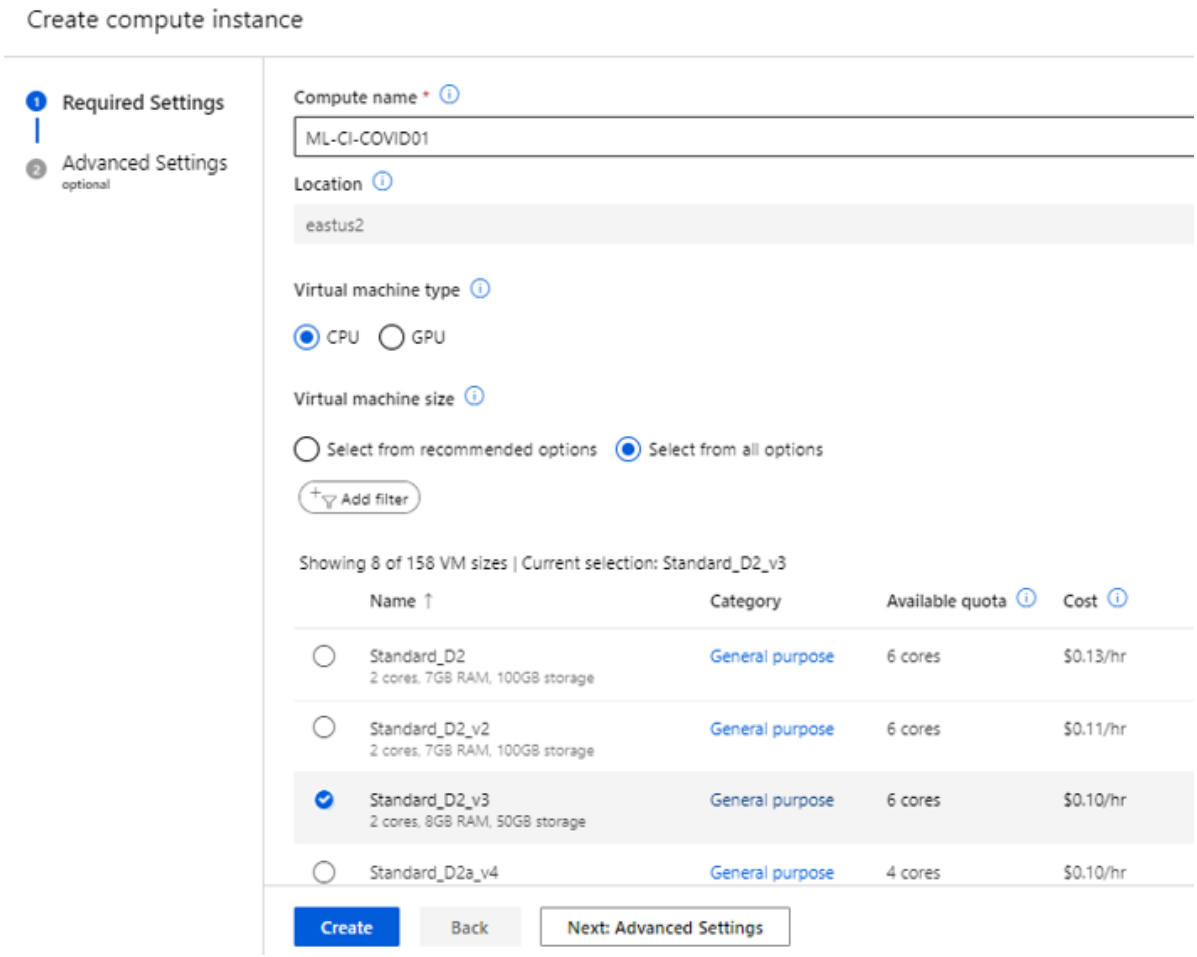

Figure 6.3: Compute Instance Creation Settings

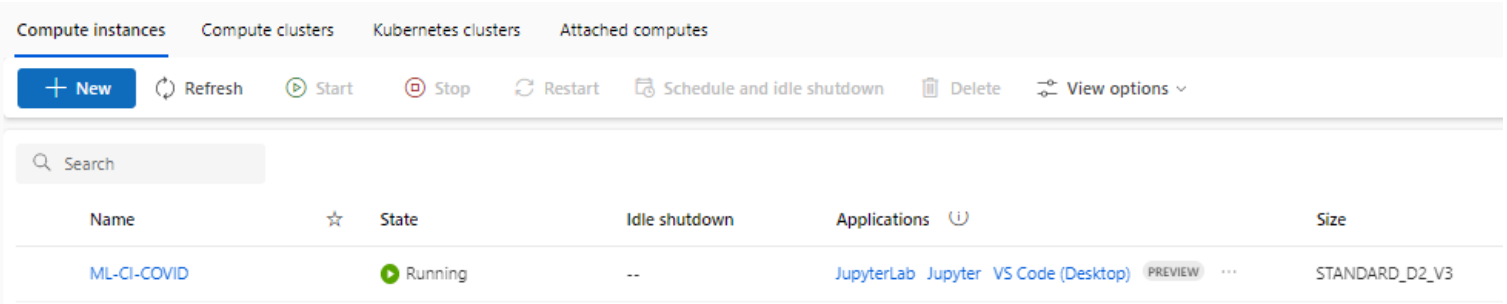

Figure 6.4: Compute Instance in Running State

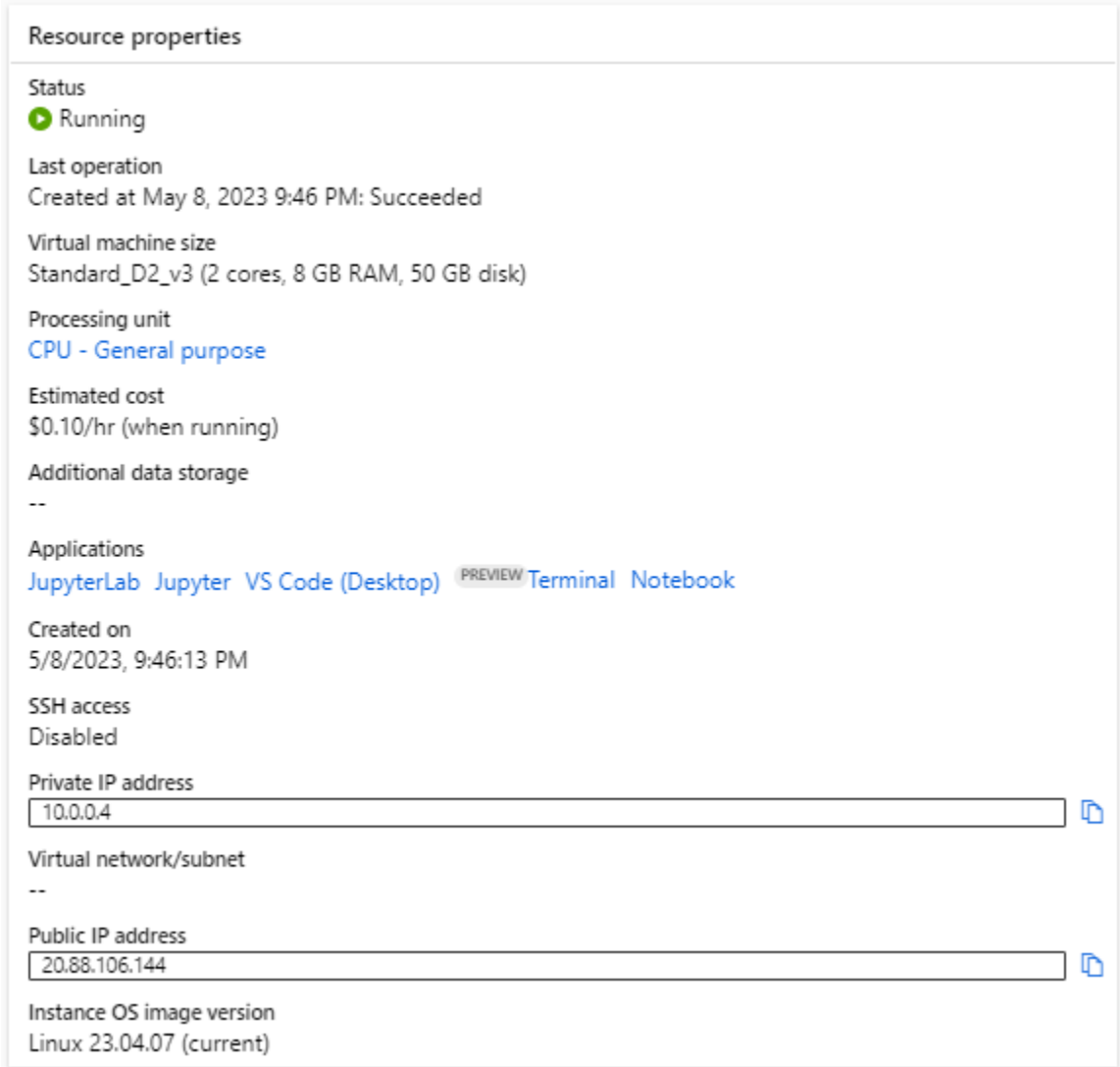

Figure 6.5: Compute Instance Resource Property Details

Compute Cluster

Workloads for parallel and distributed computing can be run on the controlled cluster of virtual machines known as Azure Compute Cluster. It offers a scalable infrastructure for conducting big data processing, machine learning models or even other compute-intensive operations. A Compute Cluster can be configured to automatically scale up or down in response to demand, always with enough resources available. The cost of Compute Cluster is determined by the quantity and duration of virtual machines used [98].

The Compute Cluster is created for scalability and high availability. The virtual machine type selected is CPU again for fast calculations and the specific VM is Standard_D11_V2 which is memory optimized and costs $0.15/h. By dividing the task among several nodes, a compute cluster may maximize resource use, enhancing performance and minimizing resource waste. This is especially helpful for projects involving COVID-19 forecasts that include huge datasets or intricate machine learning models that demand a lot of processing power. In this case there is only 1 node, since it is for personal use.

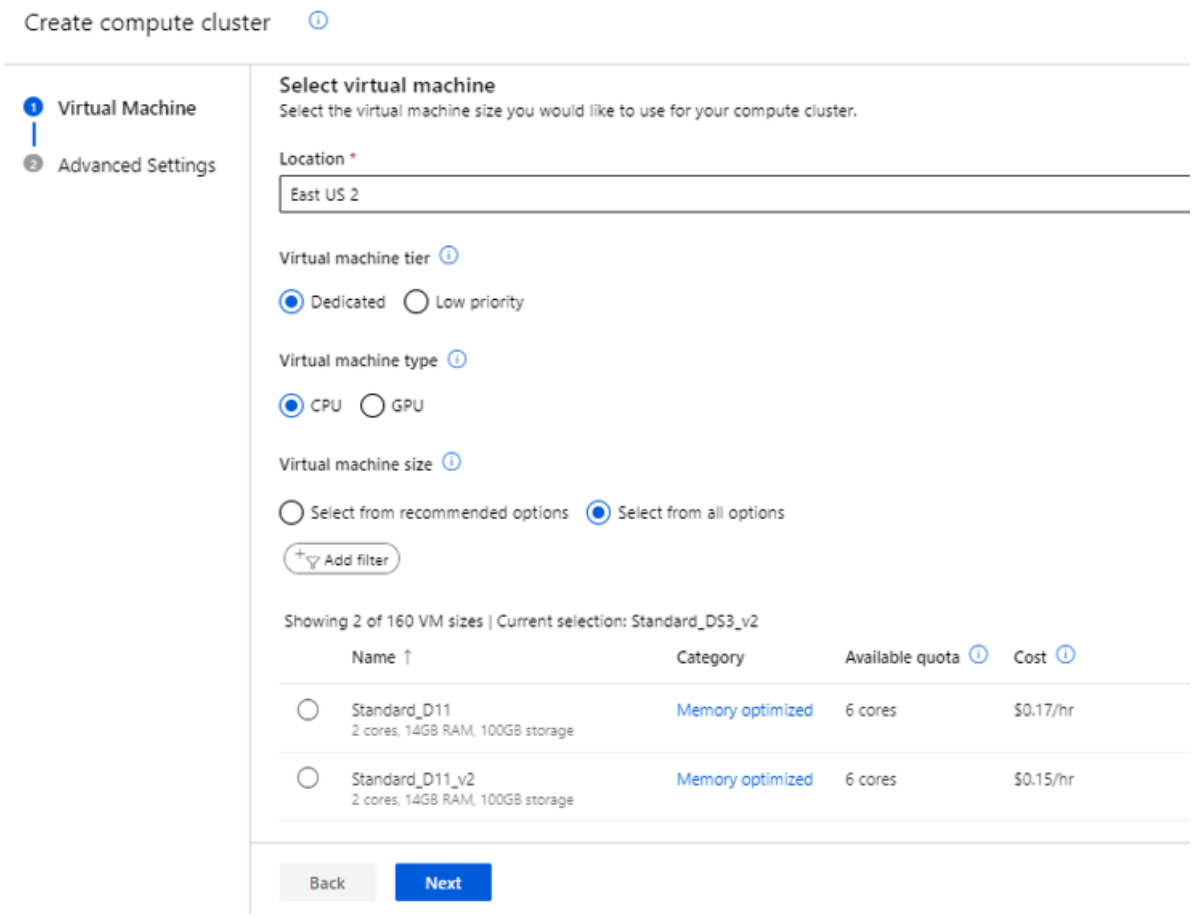

Figure 6.6: Compute Cluster Creation Settings

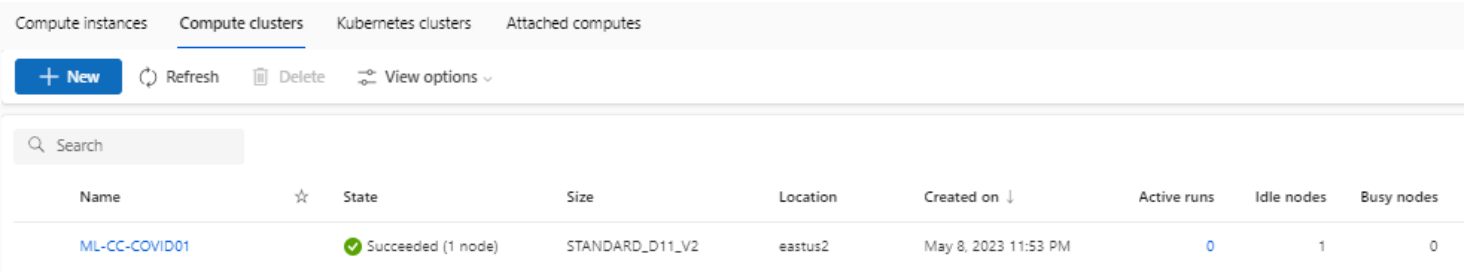

Figure 6.7: Compute Cluster Node Successfully Created

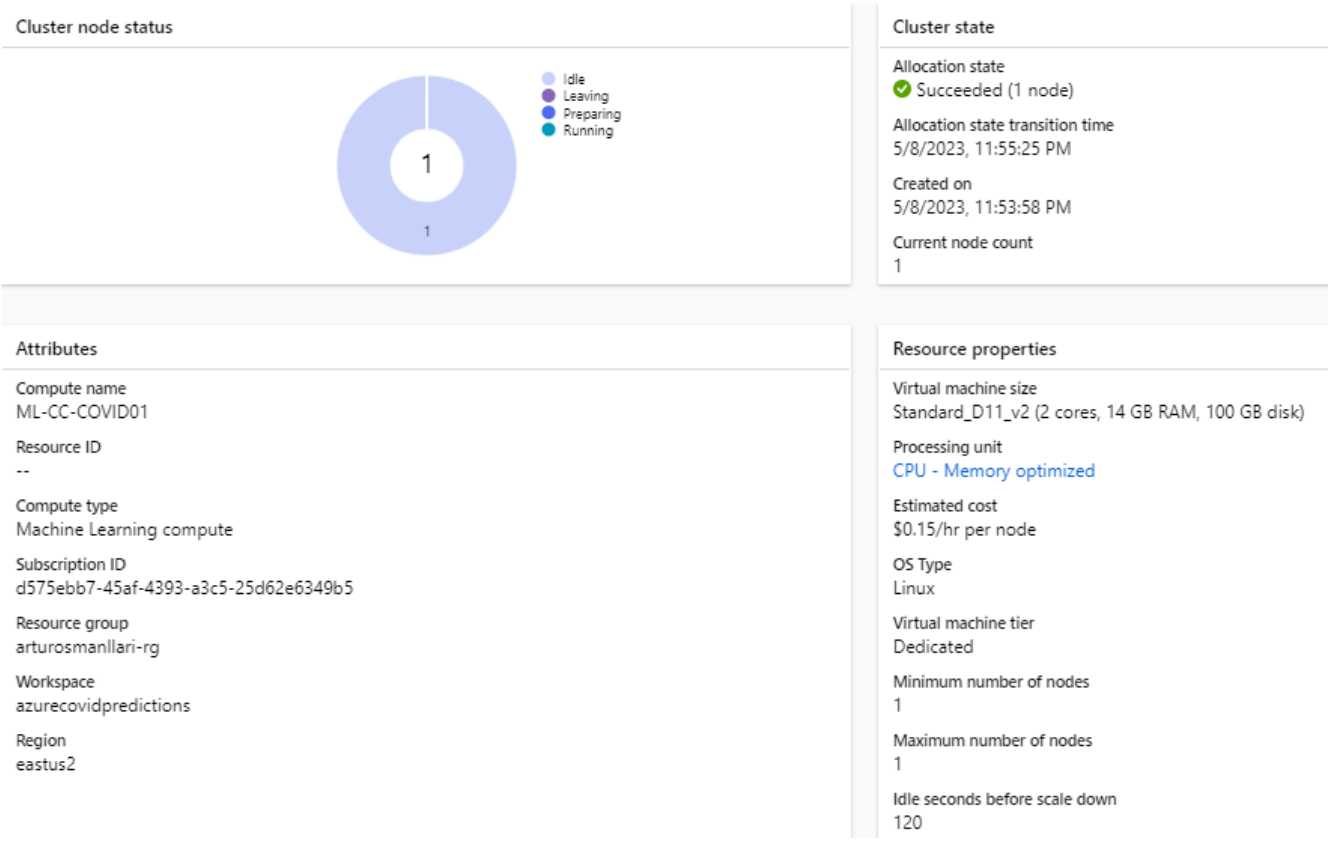

Figure 6.8: Compute Cluster Node Status and Details

AKS Compute

A managed container orchestration solution called Azure Kubernetes Solution (AKS) Compute is used to deploy and scale containerized applications. For executing machine learning models, data processing, and other compute-intensive activities in containerized environments, AKS offers a flexible and scalable architecture. AKS Compute is a great choice for creating end-to-end machine learning pipelines since it is simple to combine with other Azure services like Azure Machine Learning, Azure Cosmos DB, and Azure Storage. The cost of using AKS Compute is calculated according to the cluster's node count and use duration [98].

The only task that the AKS Compute is created for is to deploy the covid predictions real-time inference model to an endpoint that can be exposed later on to the web application. The virtual machine type selected is CPU again for fast calculations and the specific VM is Standard_D13_V2 which has 8 cores, 56GB RAM, 400GB storage in order to compute the endpoint as soon as possible.

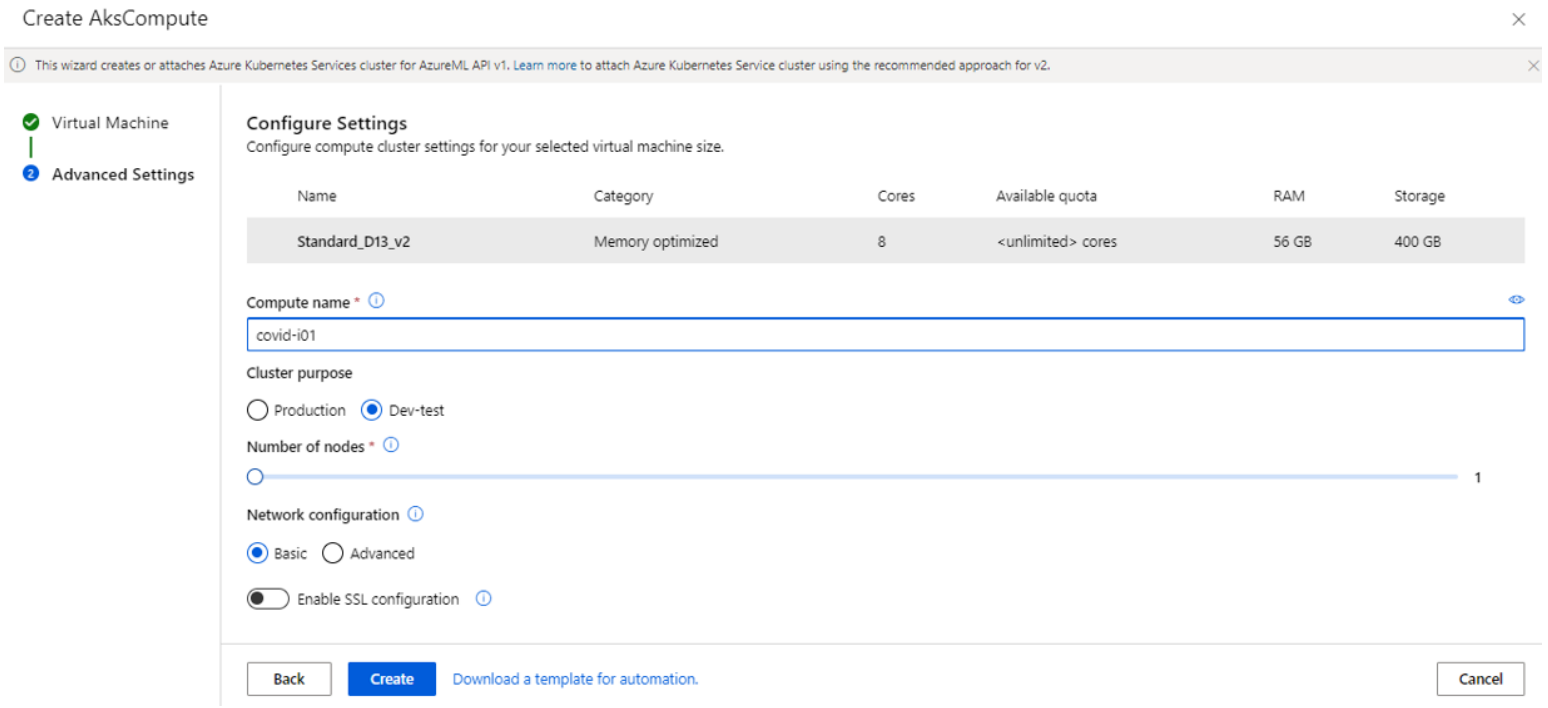

Figure 6.9: AKS Compute Creation Settings

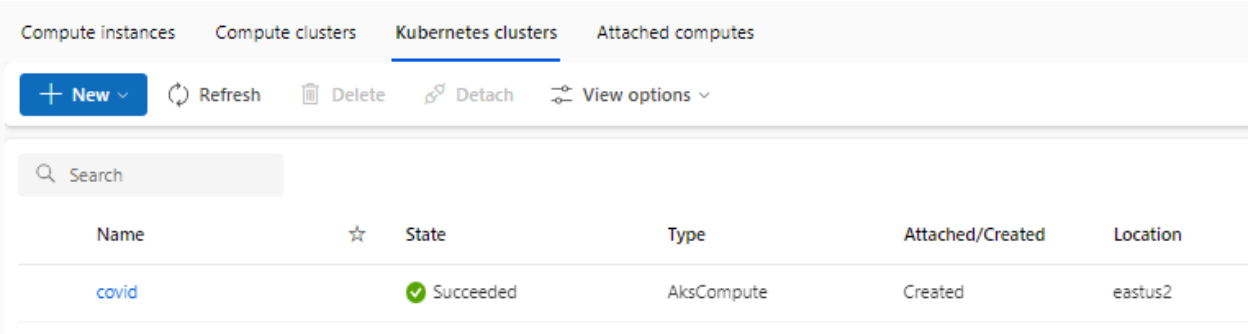

Figure 6.10: AKS Compute Successfully Created

6.3 Storages & File Uploads

6.3.1 Blob Storage

Scalable storage and access for unstructured data.

Data lakes are constructed with Azure Blob Storage to support analytics needs, also it also offers storage to design robust mobile and cloud-native applications. Utilize tiered storage to save expenses on long-term data storage while readily scaling up for applications that require high-performance computation and machine learning [99].

- Scalable, durable, available
- Secured
- Optimized for data lakes
- Comprehensive data management

Blob storages are created only in Azure Portal and not Azure Machine Learning as it is mentioned a lot as the main site of this project. Initially, a blob storage is created with the name ‘azurestorage01’. In order to upload a blob there is required to have a container.

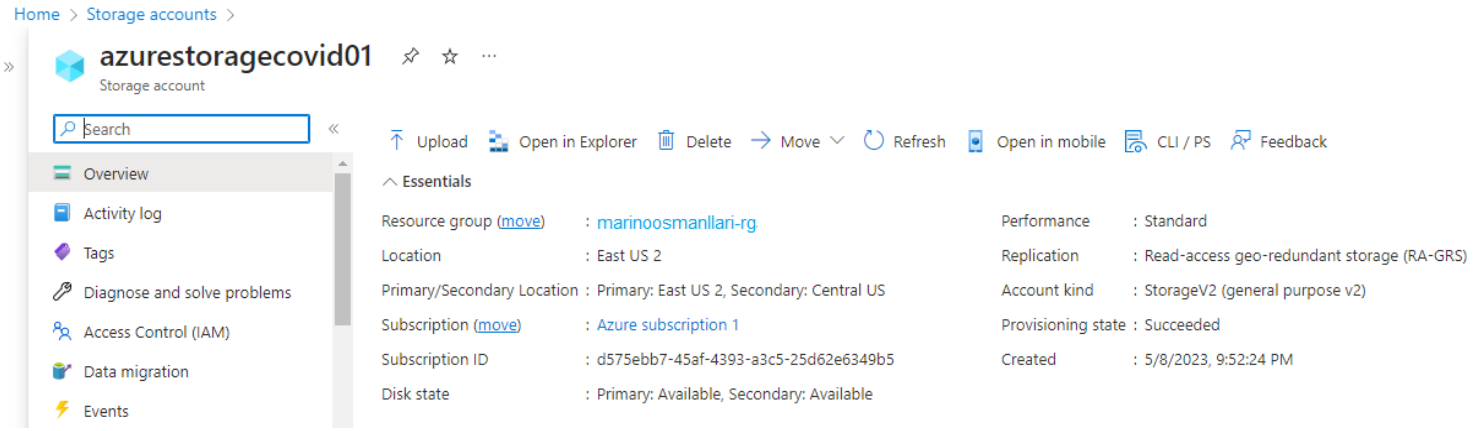

Figure 6.11: Azure Storage Overview

6.3.2 Containers

A container in Azure is a compact, independent executable package that contains all of the components required to execute an application, including as code, runtime, system tools, libraries and settings. Containers are a type of separated computing environment that may be set up and operated reliably in a variety of computer settings, including on-premises, on the public cloud, and in hybrid environments [99].

This container that is created is essential to store the blob file that should be uploaded. In figure 6.12 the uploaded file in the ‘covidcontainer’ is called covid-data-europe.csv which is a downloaded CSV file from the internet. It contains all the covid data of the world in a single file which is daily updated by WHO (World Health Organisation) and used partially for the covid predictions project.

Figure 6.13 is the site where the csv file has been downloaded from and is open source [100].

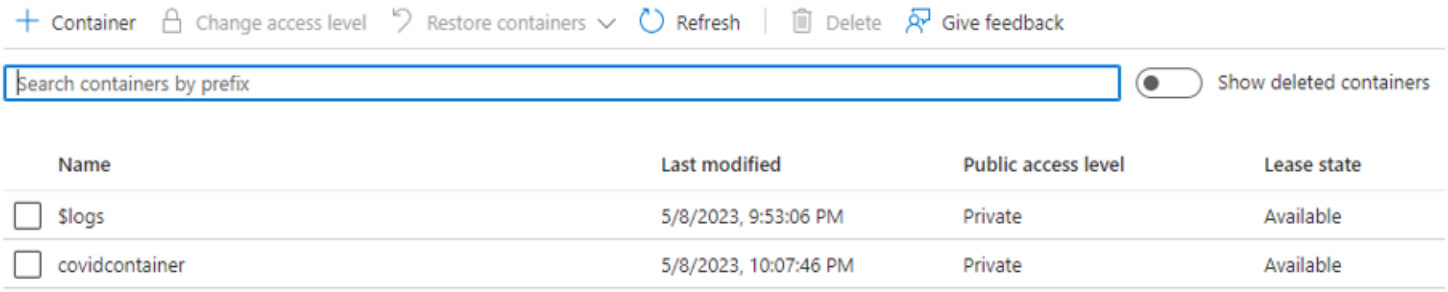

Figure 6.12: Azure Storage Containers

Figure 6.13: 'Our World in Data' COVID-19 Dataset Online

6.3.3 Blob File Upload & Cleanup

Azure allows blob files up to 2MB so the data coming from 'Our World in Data' was partially used as mentioned above. The reason the csv file is 1.72MB is because the data has over 100 columns and 40000+ rows of confirmed cases and much more information where most of the columns got deleted to be allowed to upload the csv file. Moreover, a python script is applied to take out all the countries of the world except the European ones. This is a normal but very important cleanup to get back the data required for the project.

```
import csv

input_file = 'owid-covid-data.csv'
output_file = 'cleaned-covid-data.csv'

with open(input_file, 'r') as csvfile:
    reader = csv.DictReader(csvfile)
    fieldnames = reader.fieldnames
    data = [row for row in reader if row['continent'] == 'Europe']
```

```
with open(output_file, 'w', newline='') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    for row in data:
        writer.writerow(row)

print(f"Data has been cleaned and saved to {output_file}.")
```

Listing 6.1: European Countries Clean-up Python Code

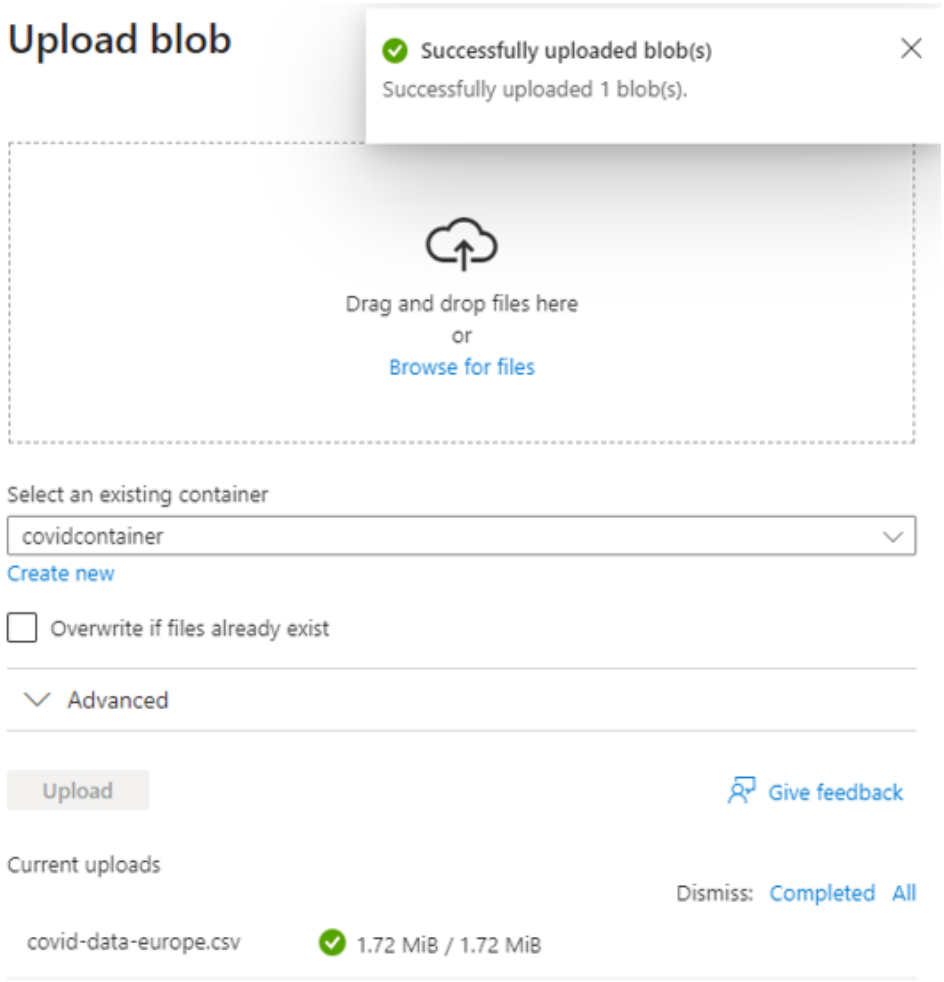

Figure 6.14: Uploaded Blob ‘covid-data-europe.csv’

6.3.4 Datastore

Now, back in Azure Machine Learning Studio, a datastore is produced to retrieve back the uploaded file in the Azure Portal and use it as a pipeline later down the line to create the predictions. To make a datastore, a couple of sections are filled such as the datastore type, subscription ID, storage account, blob container and authentication type and key.

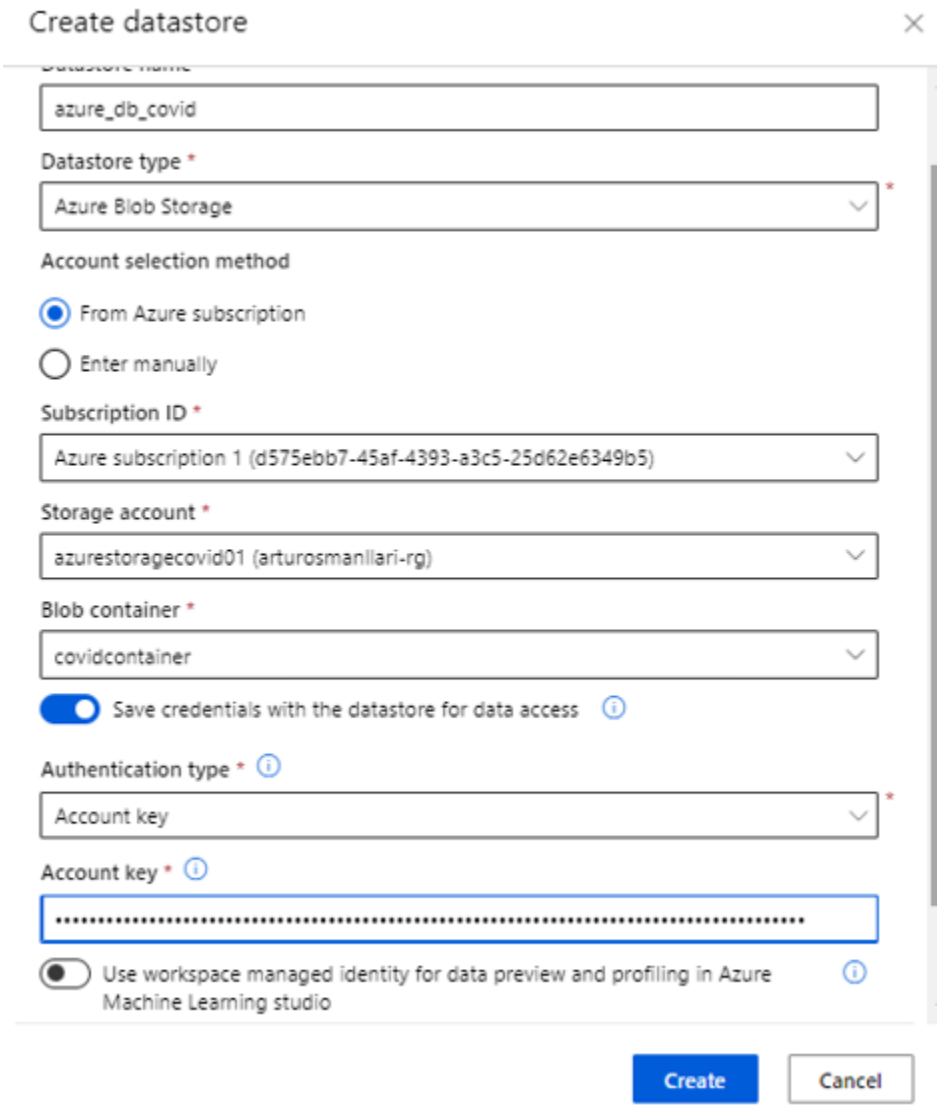

Figure 6.15: Datastore Creation

The authentication key is located in the Azure Portal inside the storage account 'azurestoragecovid01' and this key is copied and pasted back into the input box in the figure above.

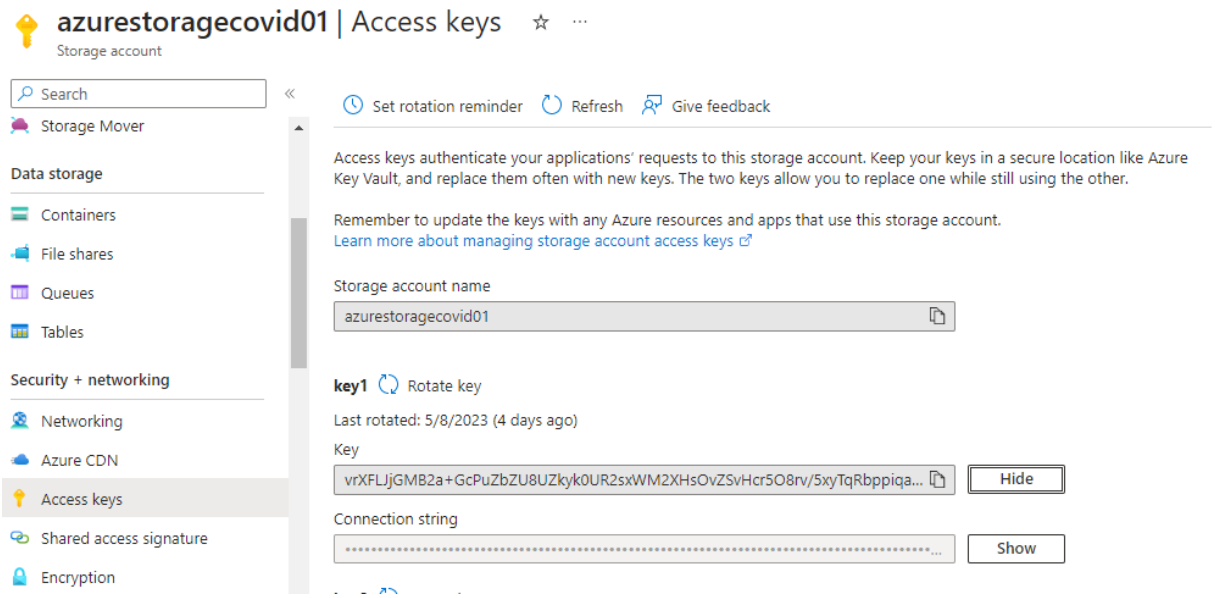

Figure 6.16: Storage Access Keys

6.3.5 Data Asset

The datastore is completed so the only part left is to create the data asset. Named 'covid_data_eu' and the type is Tabular. A sort of structured data that is arranged into rows and columns is referred to as a "tabular data model" and is represented by the data asset type "tabular" in Azure so that is the exact reason why this asset is selected as Tabular.

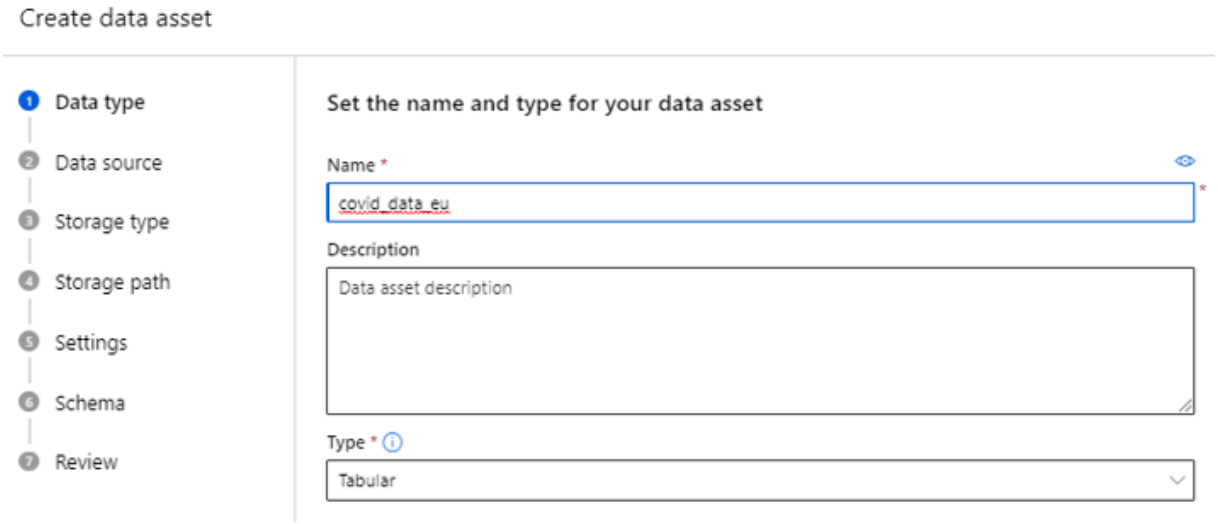

Figure 6.17: Data Asset Name and Type

Uploading the covid data blob is a great strategy due to the fact that this file is now always found in the Azure Storage and can be retrieved easily rather than retrieving it from local/web files where they could be erased and never be found again.

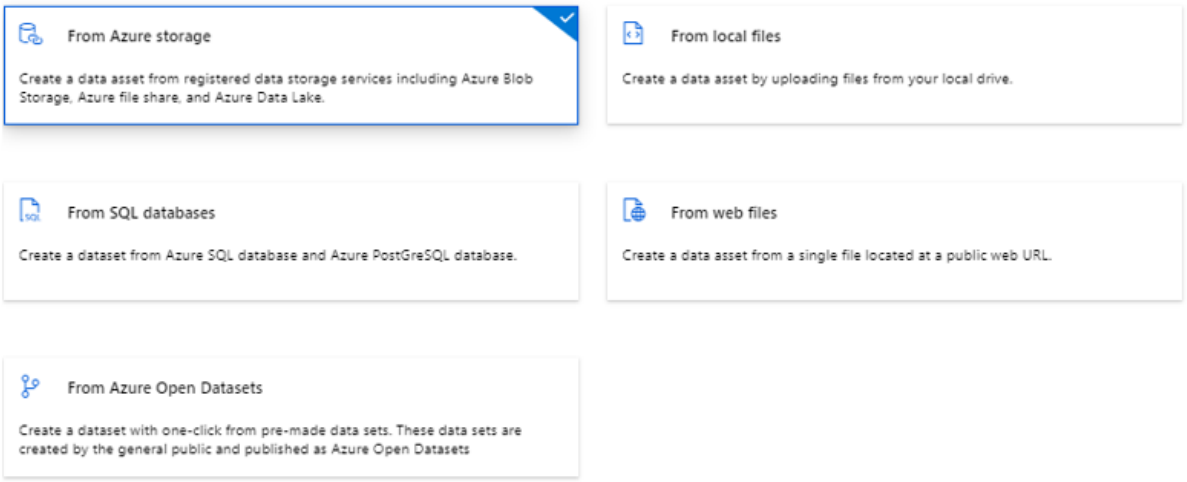

Figure 6.18: Selection for storage

Thereafter, the datastore type is Azure Blob storage and azure_db_covid is selected from the storage 'azurestoragecovid01' out of the other 2 datastores auto-created by Azure.

Following that the uploaded file appears and gets selected to be reviewed.

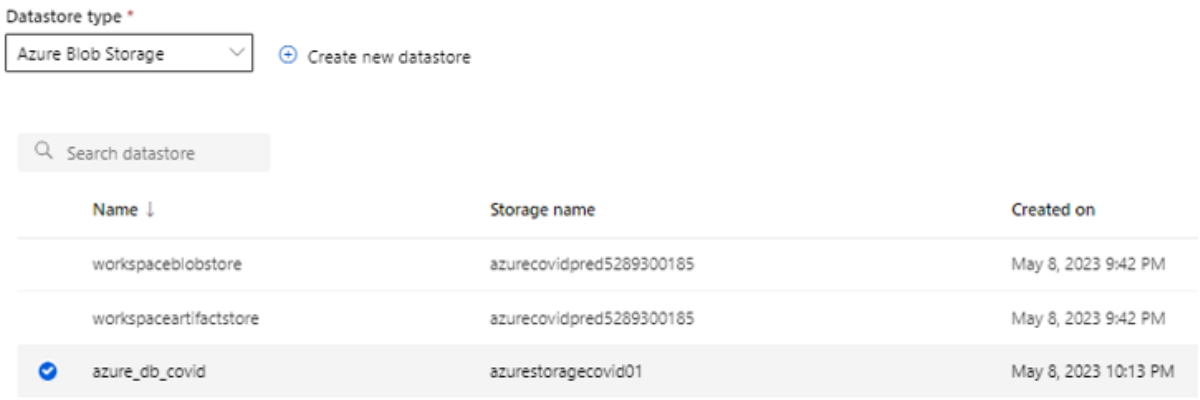

Figure 6.19: Azure Blob Storage Selected

Figure 6.20: Dataset 'covid-data-europe' Selected

The settings determine how data is parsed. The initial settings are automatically detected and can be changed if needed, but after watching the data review with the 5 columns, everything seems perfect and is ready to be passed to the Schema section.

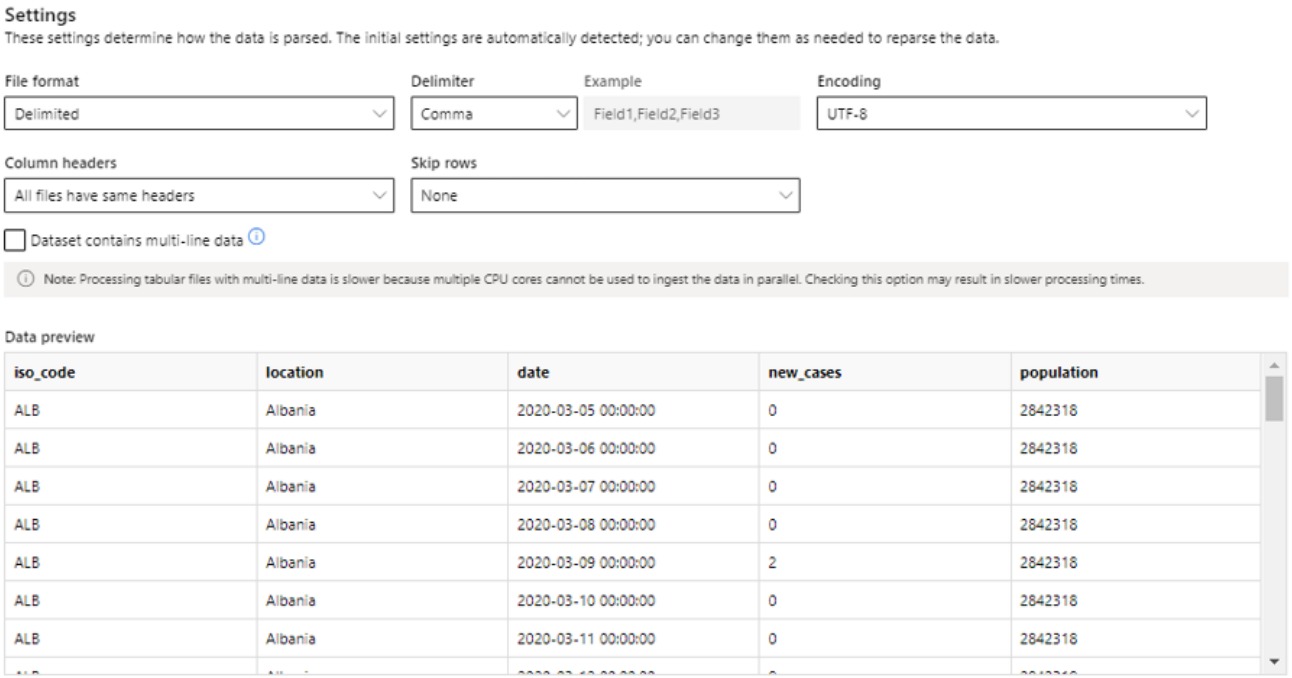

Figure 6.21: Settings Overview

In Schema the column types that are auto-detected based on the initial subset of the data can be included or change type, date format and properties.

Iso_code and location remain as strings, new_cases and population as integers and date in date format.

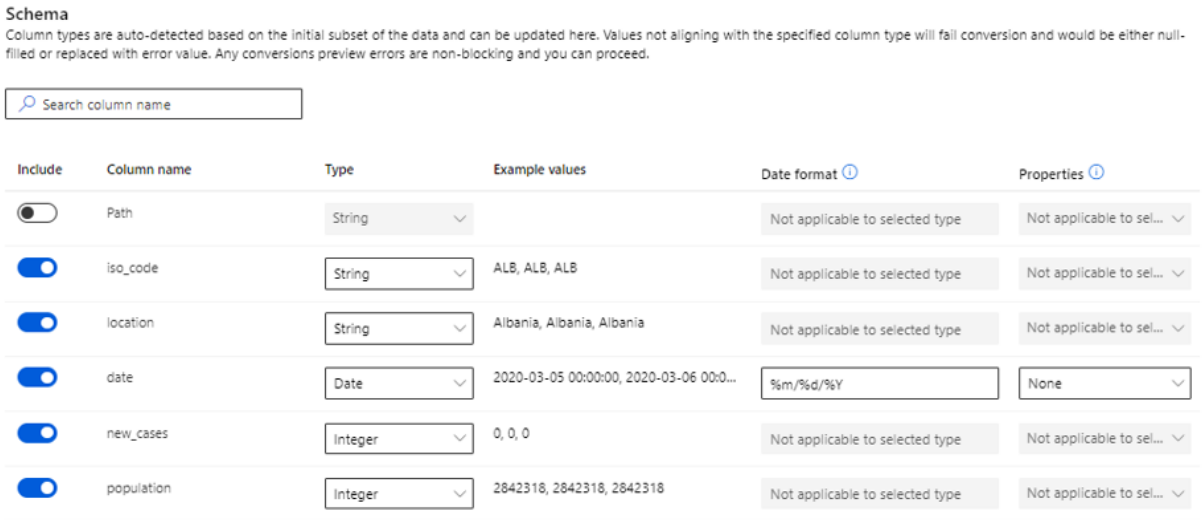

Figure 6.22: Dataset Schema

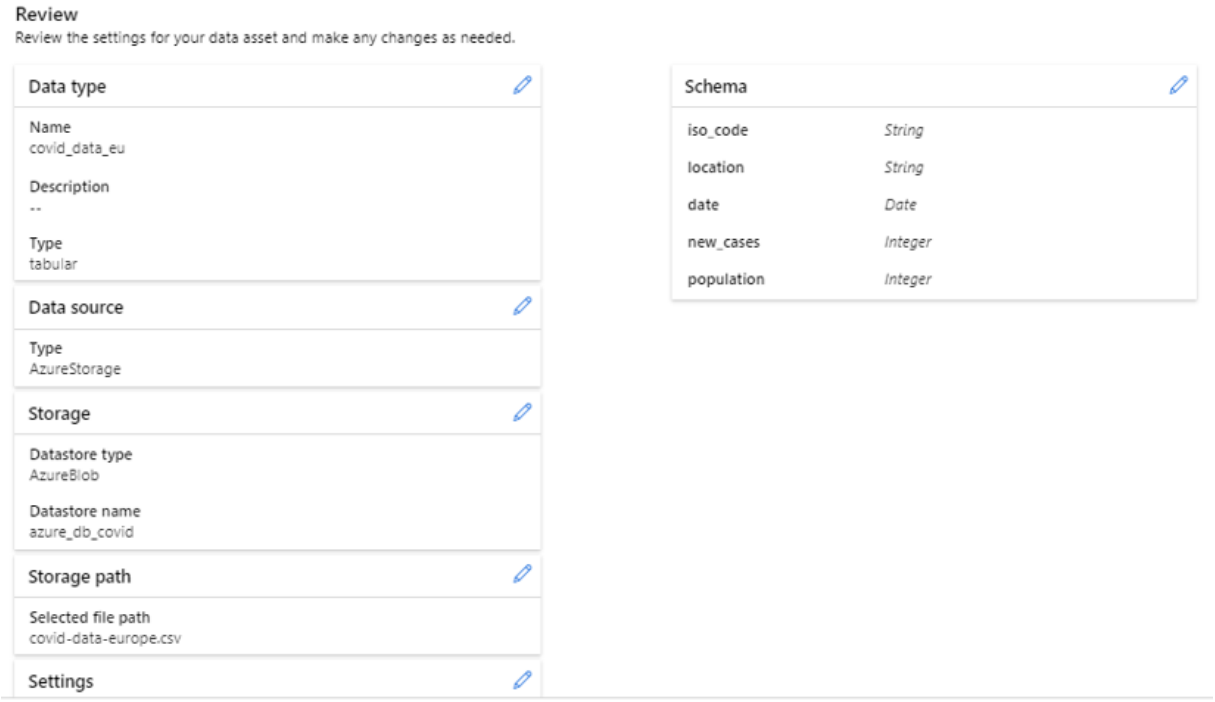

Figure 6.23: Data Asset Review

6.4 Designer

Workflows for data processing, integration and automation are created and configured using the visual interface known as the Designer in Azure. Utilizing pre-built activities and connections that can be quickly set to connect to a variety of data sources and services, the Designer offers a drag-and-drop interface for creating workflows [101].

Users can create workflows using the Designer for a range of situations, including data integration, data transformation, data analysis, and application integration. Workflows can be programmed to run repeatedly or sparked by certain occasions, such as the entry of new data [102].

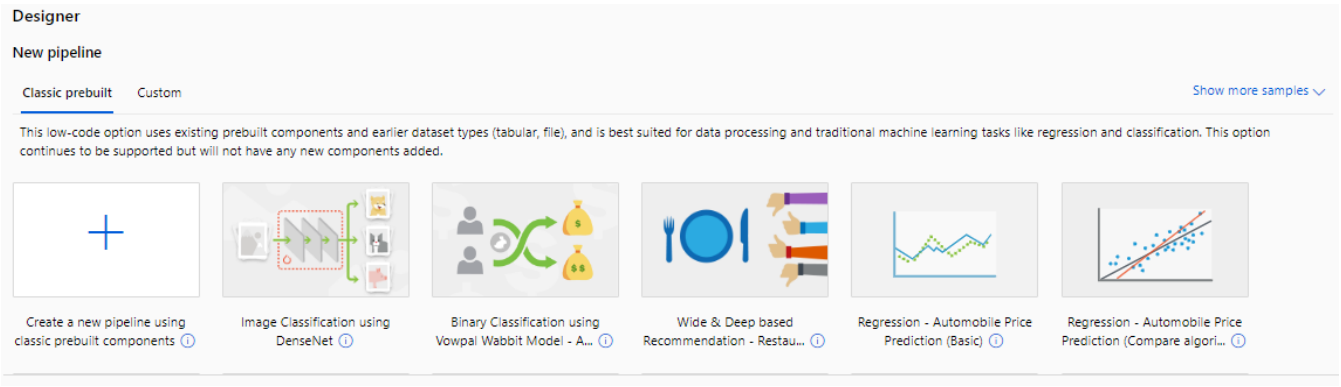

Figure 6.24: Prebuilt Pipelines collection

The Designer is a part of Azure Data Factory, a fully managed data integration service that enables customers to design, plan and control data pipelines that transfer and convert data from different sources to different destinations [102]. Initially, when it is opened there are ready made pipelines to show the data scientist how to start in this environment.

The Designer in Azure has the following salient characteristics and advantages:

- Drag and drop interface that is simple to use for setting up processes
- Easy connection to a variety of data sources and services thanks to pre-built activities and connections
- Integration with a number of Azure services (e.g. Azure SQL Database, Azure Blob Storage, Azure Data Lake Storage)
- Options for triggering and scheduling workflows for effective automation
- Integration for visibility and troubleshooting with monitoring and logging services

After selecting to create a new pipeline using classic pre-built components, the canvas is empty while in the left side of the window there are 2 parts of the asset library which are Data and Component.

In Data, we can find all the datasets uploaded by the data scientist, so in this case it can be seen that ‘covid_data_eu’ is already there and can be used to initiate the model.

The other one is Component, where all components are there to be used and categorized to be found easily. Each of these components has a distinct role, and it will be discussed briefly which ones are utilized and for what reason.

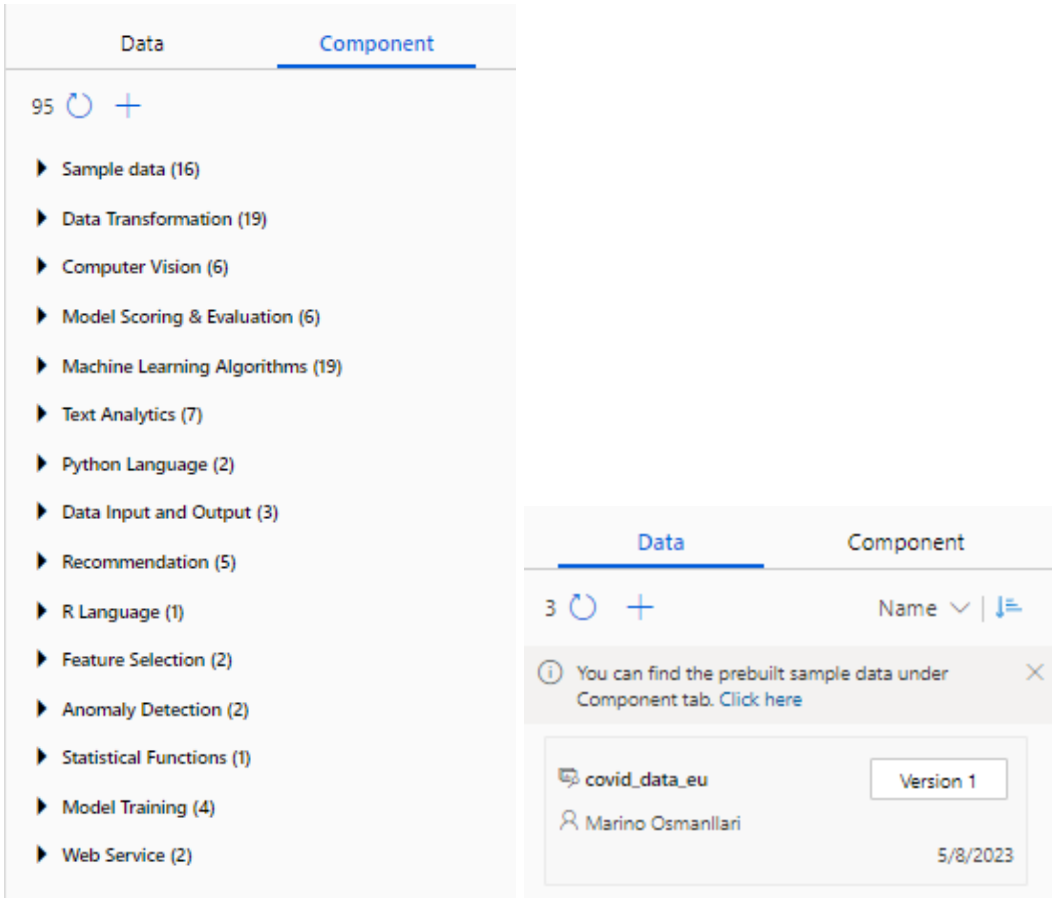

Figure 6.25: Data and Components in Designer Page

6.4.1 Covid19 Predictions Model

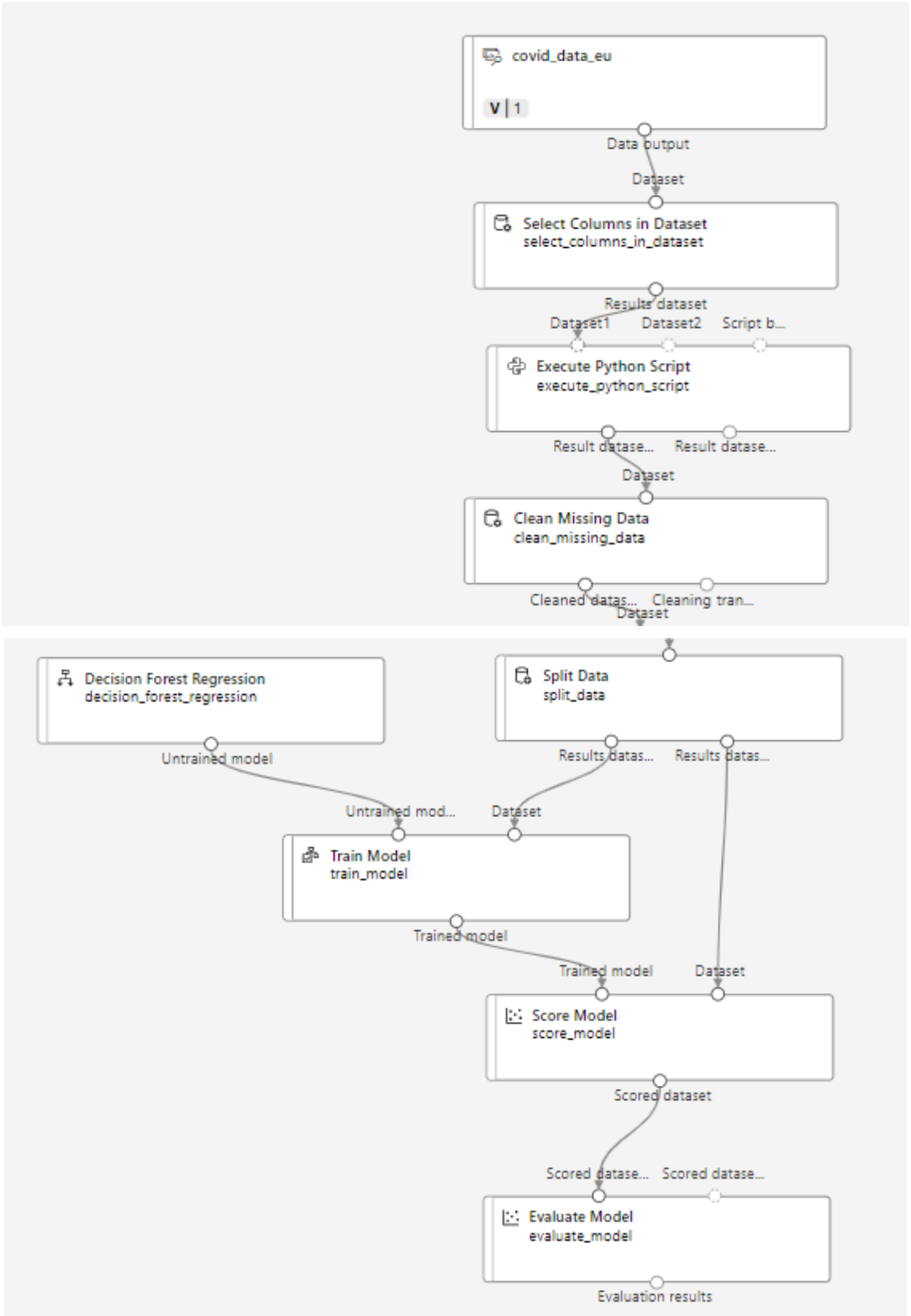

Figure 6.26: Custom Covid19 Predictions Model

Pipelines are used in Azure Machine Learning to build a series of connected stages that may be utilized for model training, data preparation and lastly, deployment. A trained model, a data transformation, or any other process produced during the pipeline execution can be the output of a pipeline.

This workflow can be divided into 2 parts: Data Collection & Transformation and Model Training/Scoring.

More specifically:

- Data Collection (eu dataset)
- Data Transformation (select columns, clean missing data, split data)
- Execution of Python Script
- Feed the data to train the model through an ML Algorithm
- Test with validation
- Score and Evaluate the Model

And afterwards, a new model has to be created to do the following:

- Implement and expose a web service
- Retrain as new data comes in again

6.4.1.1 Select Columns in Dataset

The first pipeline is the dataset provided previously in figure 6.26. Then, the desired columns are selected out of it. The selected columns in the dataset are iso_code, location, date, new_cases and population.

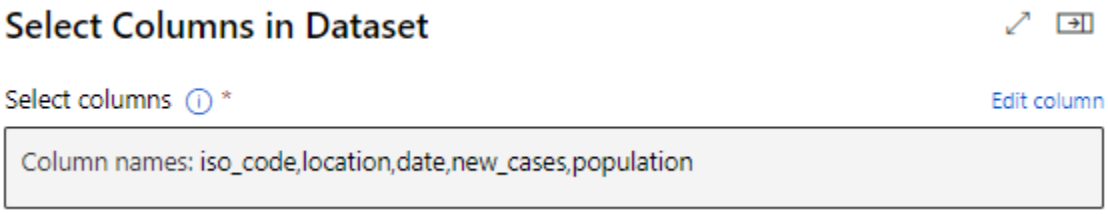

Figure 6.27: Selected Columns

6.4.1.2 Execute Python Script

The dataset provided had some problems like most of the data found online. The data looked fine but when the workflow was executed to find out the results of the predictions, they were awfully wrong. The metrics showed a big inconsistency, however the algorithm used played a part in it. After checking the csv file manually, it was found that in the later years' data started

to be recorded once every few days or weekly which was a great challenge for the ML's model. A few solutions are proposed:

- Gather the data into months
- Gather the data into weeks
- Transform the data and split it into days

Out of them the last one was chosen with the reason that this project has to do with covid cases so specific days play a huge role into the difference of covid recorded cases or predicted ones. For example, a day before Christmas, Easter or any other feast and celebration could have few cases and then spike upwards in the next few days to indicate that gatherings have happened and the covid cases have increased massively.

49058	UKR	Ukraine	2/10/2023	0	39701744
49059	UKR	Ukraine	2/11/2023	0	39701744
49060	UKR	Ukraine	2/12/2023	0	39701744
49061	UKR	Ukraine	2/13/2023	0	39701744
49062	UKR	Ukraine	2/14/2023	5283	39701744
49063	UKR	Ukraine	2/15/2023	0	39701744
49064	UKR	Ukraine	2/16/2023	0	39701744
49065	UKR	Ukraine	2/17/2023	0	39701744
49066	UKR	Ukraine	2/18/2023	0	39701744
49067	UKR	Ukraine	2/19/2023	0	39701744
49068	UKR	Ukraine	2/20/2023	0	39701744
49069	UKR	Ukraine	2/21/2023	7344	39701744
49070	UKR	Ukraine	2/22/2023	0	39701744
49071	UKR	Ukraine	2/23/2023	0	39701744
49072	UKR	Ukraine	2/24/2023	0	39701744
49073	UKR	Ukraine	2/25/2023	0	39701744
49074	UKR	Ukraine	2/26/2023	0	39701744
49075	UKR	Ukraine	2/27/2023	0	39701744
49076	UKR	Ukraine	2/28/2023	9792	39701744
49077	UKR	Ukraine	3/1/2023	0	39701744
49078	UKR	Ukraine	3/2/2023	0	39701744
49079	UKR	Ukraine	3/3/2023	0	39701744
49080	UKR	Ukraine	3/4/2023	0	39701744

Figure 6.28: Error regarding Data Recorded (weekly)

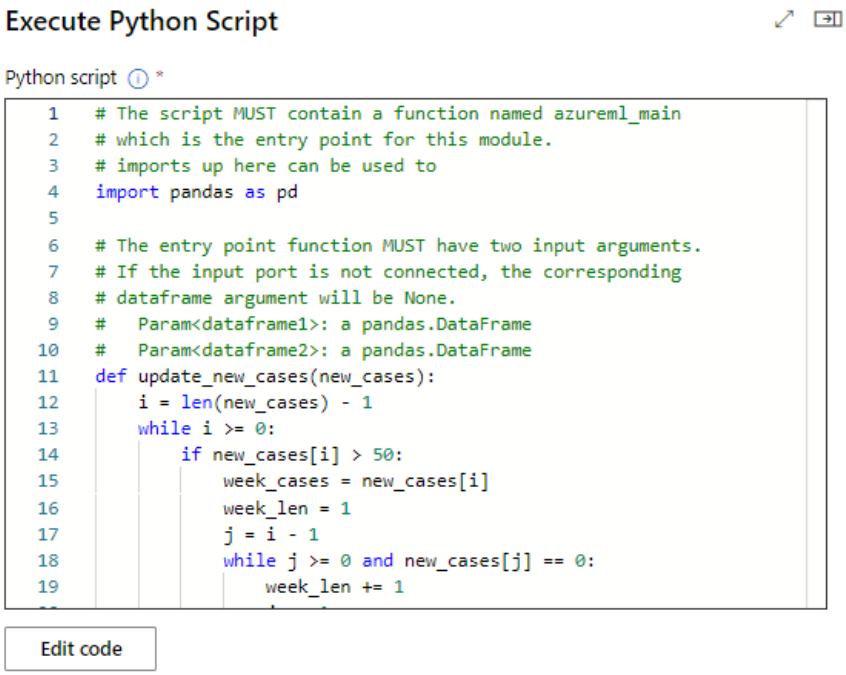

Figure 6.29: Execute Python Script Pipeline

This python code is 'covid-split.py' which is executed locally to check manually if the cases have successfully been split. There were many attempts to fix it and this is the last version that confirmed cases were split equally from the last date among the 0's above it. Otherwise, if the cases were split randomly it would have been almost the same as unedited with random spikes upwards and downwards.

```
import csv

input_file = 'covid-data-europe.csv'
output_file = 'cleaned-covid-data-europe.csv'

# This function takes a list of new cases and updates it by splitting any week
with more than 50 cases into daily cases for each day in that week.
def update_new_cases(new_cases):
    i = len(new_cases) - 1
    while i >= 0:
        # Check if the current week has more than 50 cases
        if new_cases[i] > 50:
            week_cases = new_cases[i]
            week_len = 1
            j = i - 1
            # Count the number of consecutive weeks with 0 cases
            while j >= 0 and new_cases[j] == 0:
                week_len += 1
                j -= 1
```

```
            # Calculate the daily cases for the week
            daily_cases = round(week_cases / week_len)
        # Update the new_cases list with the daily cases for each day in the week
            for k in range(j + 1, i + 1):
                new_cases[k] = daily_cases
            # Move the index back to the end of the previous week (or the
beginning of the list)
            i = j
        else:
            i -= 1
    # Return the updated new_cases list
    return new_cases

with open(input_file, 'r') as csvfile:
    reader = csv.DictReader(csvfile)
    fieldnames = reader.fieldnames
    data = [row for row in reader]

new_cases = [int(row['new_cases']) if row['new_cases'].isdigit() else 0 for row
in data]
updated_new_cases = update_new_cases(new_cases)

for i, row in enumerate(data):
    row['new_cases'] = updated_new_cases[i]

with open(output_file, 'w', newline='') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    for row in data:
        writer.writerow(row)

print(f"Data has been updated and saved to {output_file}.")
```

Listing 6.2: Local Test to Split Gathered Covid Cases into Days

Meanwhile, listing 6.3 is the python code executed in the pipeline in the workflow which uses two methods. One to execute the code and split the cases and the other one that updates the new_cases data column. The ‘update_new_cases’ method that executes the code is provided in the listing 6.2 so there is no need for duplicates.

```
import pandas as pd

# This is the main function that will be called by Azure ML Studio.
def azureml_main(dataframe1: pd.DataFrame, dataframe2: pd.DataFrame = None) ->
pd.DataFrame:
    # Copy the input dataframe to avoid modifying the original data.
    data = dataframe1.copy()
```

```
    # Convert the 'new_cases' column to numeric data type with errors coerced to
NaN.
    data['new_cases'] = pd.to_numeric(data['new_cases'], errors='coerce')

    # Convert the 'new_cases' column to a list and update it using the
'update_new_cases' function.
    new_cases = data['new_cases'].tolist()
    updated_new_cases = update_new_cases(new_cases)

    # Update the 'new_cases' column in the dataframe with the updated values.
    data['new_cases'] = updated_new_cases

    # Return the updated dataframe to Azure ML Studio.
    return data
```

Listing 6.3: Split Gathered Covid Cases in Azure ML Studio to the next Pipeline

6.4.1.3 Clean Missing Data

Missing data is cleaned from the model where all the columns are selected with a single exclusion. The new_cases are the ones required to be trained. If any cell of the dataset is found to be empty other than new_cases, the whole row is removed. This way it confirms the integrity and accuracy of the ML model by preventing biased or incorrect predictions caused by missing data.

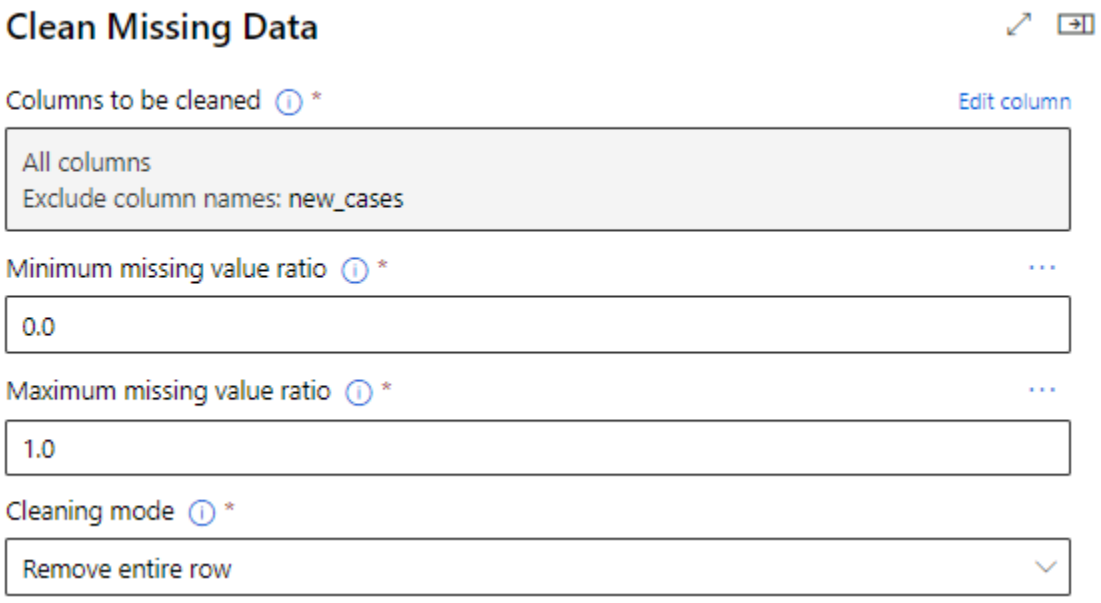

Figure 6.30: Clean Missing Data

6.4.1.4 Split Data

Next step is to split the data. In order to produce training and testing datasets, it is standard practice in machine learning to split the data into 0.7 and 0.3 (or any other equivalent ratio, such as 0.8 and 0.2). This is known as the train-test split.

The reason for splitting the data is to assess how well the ML model is performing on unseen data. The model is trained using the splitted training dataset (70%), and its performance is assessed using the testing dataset (30%).

By dividing the data in this manner, overfitting may be prevented by enabling the machine learning model to be trained on just a part of the data and assessed on a distinct subset. When a model is too complicated, it tends to learn the intricacies and noise in the training data instead of the underlying patterns. The model will perform well on the training data but badly on the testing data if it is overfitted.

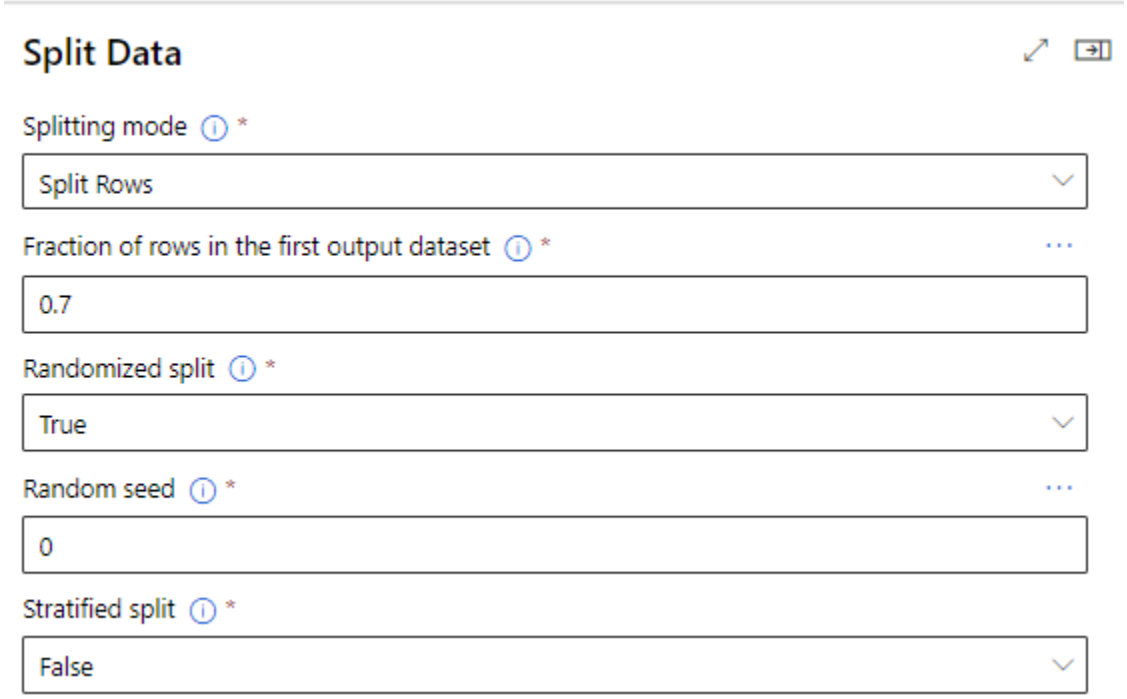

Figure 6.31: Split Data

6.4.1.5 Algorithm Selection

Algorithm selection is key for accurate and effective machine learning models. The algorithm selected is Decision Forest Regression. Regression makes forecasts by estimating the relationship between values. Out of this sentence the key word is “forecasts” which is essential for this project.

There were 7 alternatives to choose from:

1. Linear Regression
2. Decision Forest Regression
3. Boosted Decision Tree Regression
4. Fast Forest Quantile Regression
5. Neural Network Regression
6. Bayesian Linear Regression
7. Poisson Regression

The best performance for covid predictions was brought by Decision Forest Regression.

Decision Forest Regression consists of an ensemble of decision trees. Each tree outputs a Gaussian distribution as a prediction. Then an aggregation is performed to find a Gaussian distribution closest to the combined distribution of all trees in the model [52].

The advantages of this algorithm are:

- Efficient computation & memory usage during testing and prediction
- Can represent non-linear decision boundaries
- Integrated feature selection/classification and resilient to noisy data

In Azure there are possibilities that the algorithm can be modified. Trainer mode can be selected between SingleParameter and ParameterRange. Moreover, the number of decision trees, maximum depth and minimum number of samples per leaf node can be changed manually.

Explained previously, the set is divided into 70%-Training and 30%-Testing. But this 70% of training will be split again into 70%-Training and 30%-Validation to create a better model. There are a lot of options where all the results afterwards are tested to see how and which modification comes closest to the best model. Three alternatives were tested where one was to change a single option, two options or more. Out of all of them nevertheless, the default one appeared to train the best model and that is why it was the one selected.

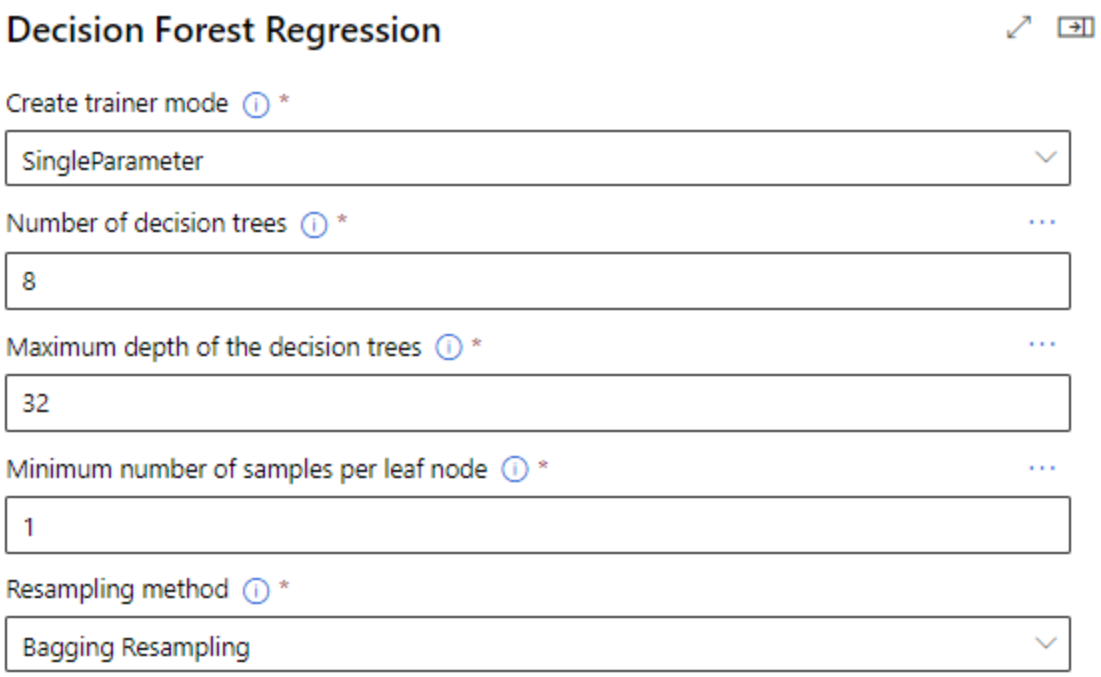

Figure 6.32: Decision Forest Regression Algorithm

6.4.1.6 Model Training & Scoring

Finally, the model is trained for the only column required 'new_cases', scored by the machine and evaluated by 5 metrics which will be explained in a bit.

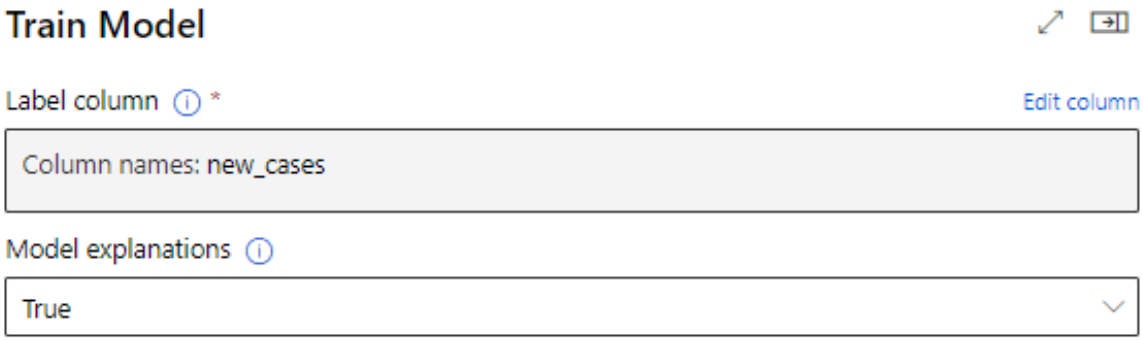

Figure 6.33: Train Model

Figure 6.34: Score Model

To run this model, in settings some small configurations are needed to be filled. The default compute is selected is Compute cluster and the default datastore is azure_db_covid.

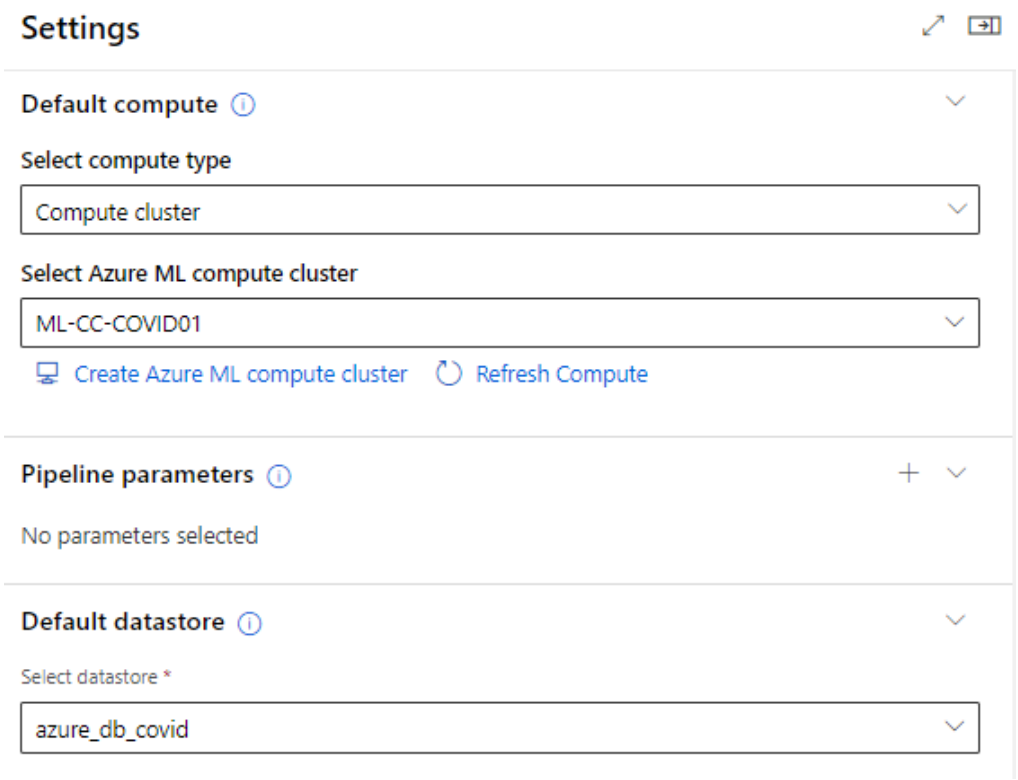

Figure 6.35: Settings Section to Run Workflow Pipeline Model

The model gets submitted and starts running. Every pipeline should show on the left side a green vertical rectangular and a check mark to indicate that they have been completed successfully and the results are ready, figure 6.38 shows it clearly in the next section. In case that any of them is red and shows an 'X', then the model has failed and a log is displayed to understand what happened wrong.

By opening the scored model (Scored_Dataset) and clicking on the scored labels all the statistics are shown in the small window on the right side. These statistics are calculated by Azure's VMs and produce exact values of each one. It can be seen clearly that the min and max values are completely fine due to the fact that there cannot be negative values or more

than huge countries' populations such as (Russia, Germany, France, UK, Spain). Other than that, the rest are not possible to be predicted and calculated by a human.

Scored Labels

Statistics

Mean	4885.2922
Median	517.5625
Min	0
Max	410575.375
Standard deviation	17301.4713
Unique values	9473
Missing values	0
Feature type	Numeric Score

Figure 6.36: Scored Labels Statistics

Scored_dataset

Rows 15,082 Columns 6

iso_code	location	date	new_cases	population	Scored Labels
BLR	Belarus	2023-03-31 00:00:00	0	9534956	0
AUT	Austria	2023-01-03 00:00:00	1317	8939617	3241.125
DEU	Germany	2020-10-05 00:00:00	1156	83369840	2675.375
SWE	Sweden	2021-10-06 00:00:00	597	10549349	588.625
NLD	Netherlands	2021-05-25 00:00:00	2727	17564020	3187.25
LVA	Latvia	2022-04-06 00:00:00	1223	1850654	1024.25
AND	Andorra	2022-06-17 00:00:00	41	79843	34.25
AUT	Austria	2022-12-20 00:00:00	2842	8939617	5624.375
BIH	Bosnia and Herzegovina	2021-01-19 00:00:00	110	3233530	270
NOR	Norway	2023-03-23 00:00:00	90	5434324	79.625
SWE	Sweden	2020-08-06 00:00:00	333	10549349	383.75
LVA	Latvia	2020-06-11 00:00:00	2	1850654	3.125
CYP	Cyprus	2023-02-06 00:00:00	194	896007	196.5
LVA	Latvia	2022-05-04 00:00:00	424	1850654	291
BLR	Belarus	2020-12-20 00:00:00	1931	9534956	1923.875

Figure 6.37: Scored Dataset Visualization

6.4.1.7 Model Evaluation

Now the model is ready to be checked and evaluated. Going into the metrics is the only way to understand if the model is accurate or not.

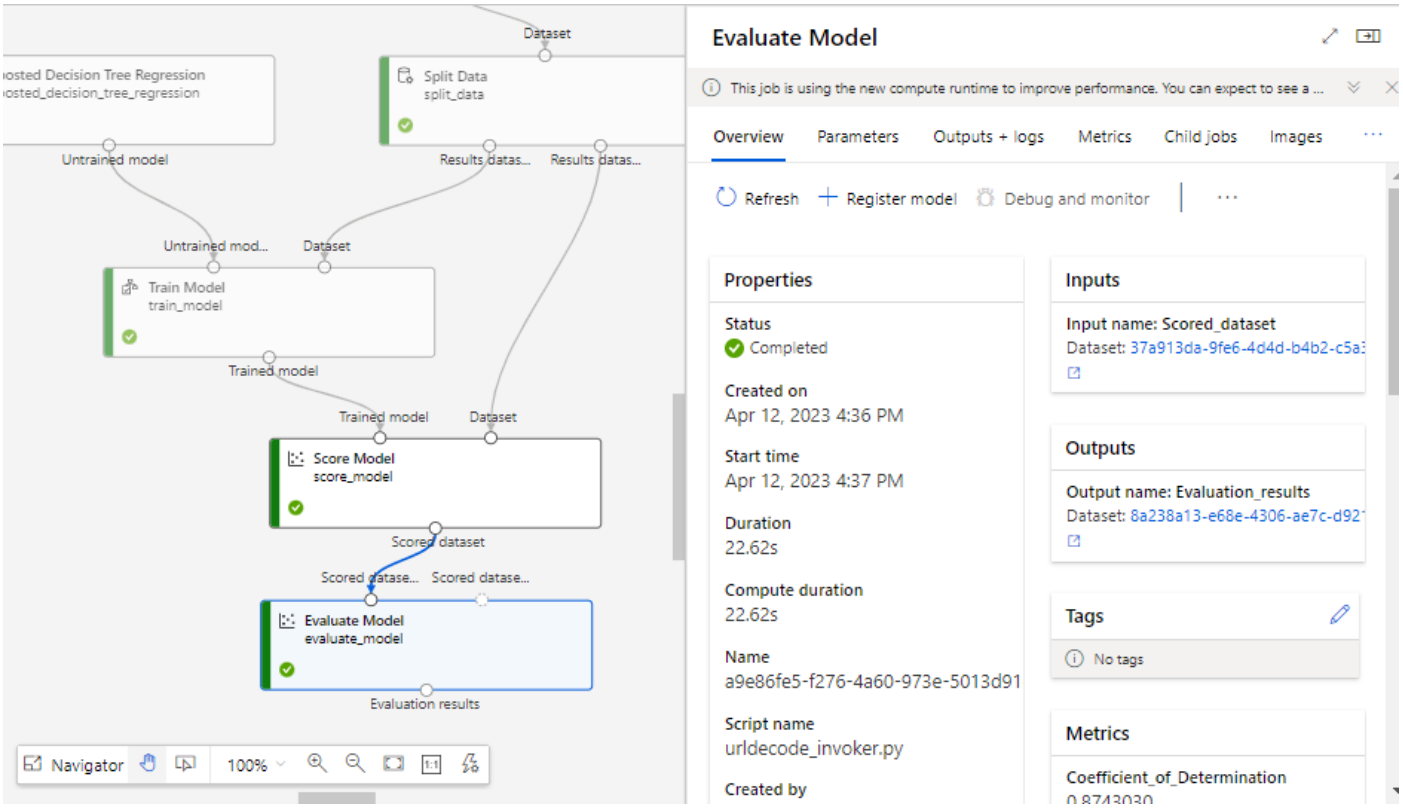

Figure 6.38: Evaluated Model Pipeline

The regression model metrics provided are intended to assess the level of error. If the disparity between observed and anticipated values is minimal, a model is said to fit the data well. However, examining the pattern of the residuals (the difference between any one projected point and its associated actual value) might reveal a lot about the model's possible bias [103].

Mean Absolute Error (MAE):

The Mean Absolute Error measures the average absolute difference between the actual and anticipated values. It is derived by averaging the absolute disparities between the values that were anticipated and those that actually occurred. The MAE is frequently used to gauge how effectively a regression model can forecast the desired variable and how close the predictions are to the actual outcomes [103].

Root Mean Squared Error (RMSE):

The Root Mean Squared Error measures the average deviation of the predicted values from the actual values. The difference between the anticipated and actual numbers is squared, and this difference is used to produce the indicator that disregards the difference between over-prediction and under-prediction [103].

Relative Absolute Error (RAE):

The Relative Absolute Error is the relative absolute difference between actual and predicted values, relative because the mean difference is divided by the arithmetic mean. The average of the absolute differences between the predicted and actual values, divided by the average of the actual values gives back the RAE [103].

Relative Squared Error (RSE):

The Relative Squared Error is a decimal error number like RAE that normalizes the total squared difference between the actual and predicted values. The estimation calculated by dividing the average of the actual values by the average of the squared differences between the predicted and actual values. [103].

Coefficient of Determination:

The Coefficient of Determination, also referred as R^2 is a measure of how well the regression model fits the data where 0 is not fit at all and 1 as the optimal one. It is determined as the percentage of the dependent variable's overall variance that is explained by the independent variables [103].

All five metrics are explained in detail and the idea is for the errors to have very low values (Relative errors close to 0) and the Coefficient of Determination to be as close as possible to 1.

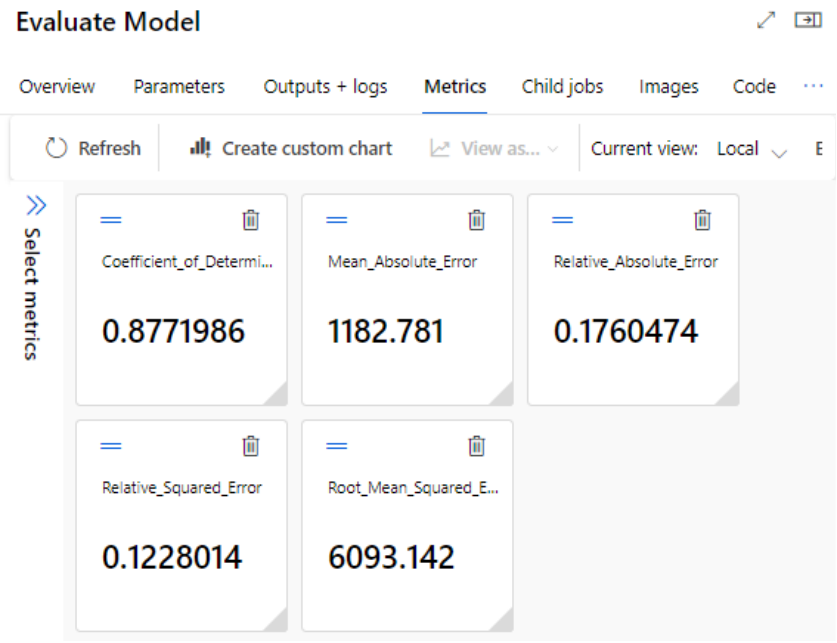

Figure 6.39: Evaluate Model Metrics

6.4.2 Covid19 Predictions real-time inference

The model is quite accurate so it is ready to create a Real-time inference pipeline and allow inputs and outputs to be taken out of the model in a custom application. In the navigation part in the canvas there can be seen a button to create an inference pipeline.

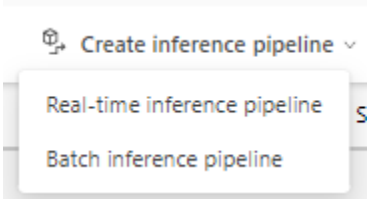

Figure 6.40: Inference Pipeline Creation

In order to allow machine learning models to generate predictions on new data in real-time, real-time inference was developed in Azure. Applying a trained machine learning model to new data in order to provide predictions or conclusions as soon as feasible is known as real-time inference [104].

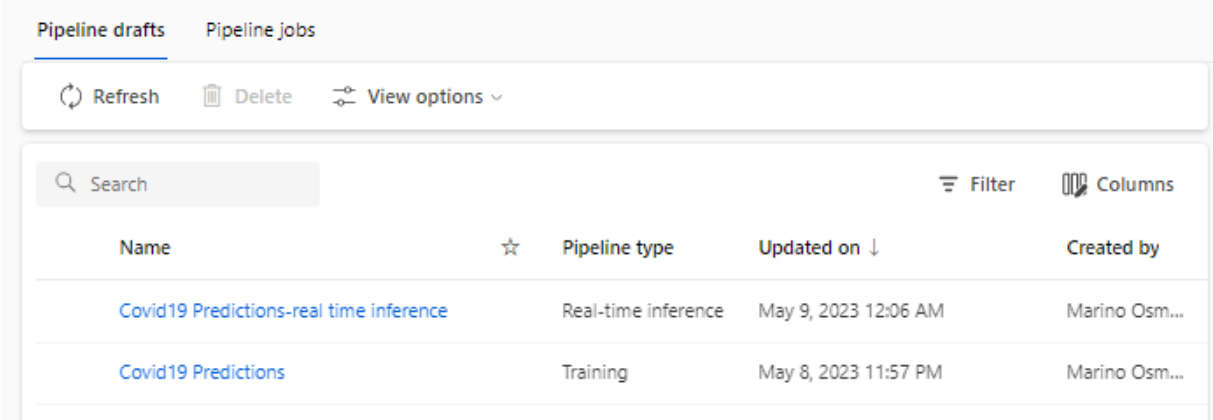

Figure 6.41: Pipeline Drafts and Types

Azure constructs a real-time inference model from the one before but with a few changes. This is the final model that is edited by taking some of the pipelines that are not required anymore such as executing the python script, cleaning data after data has been updated from the python script etc.

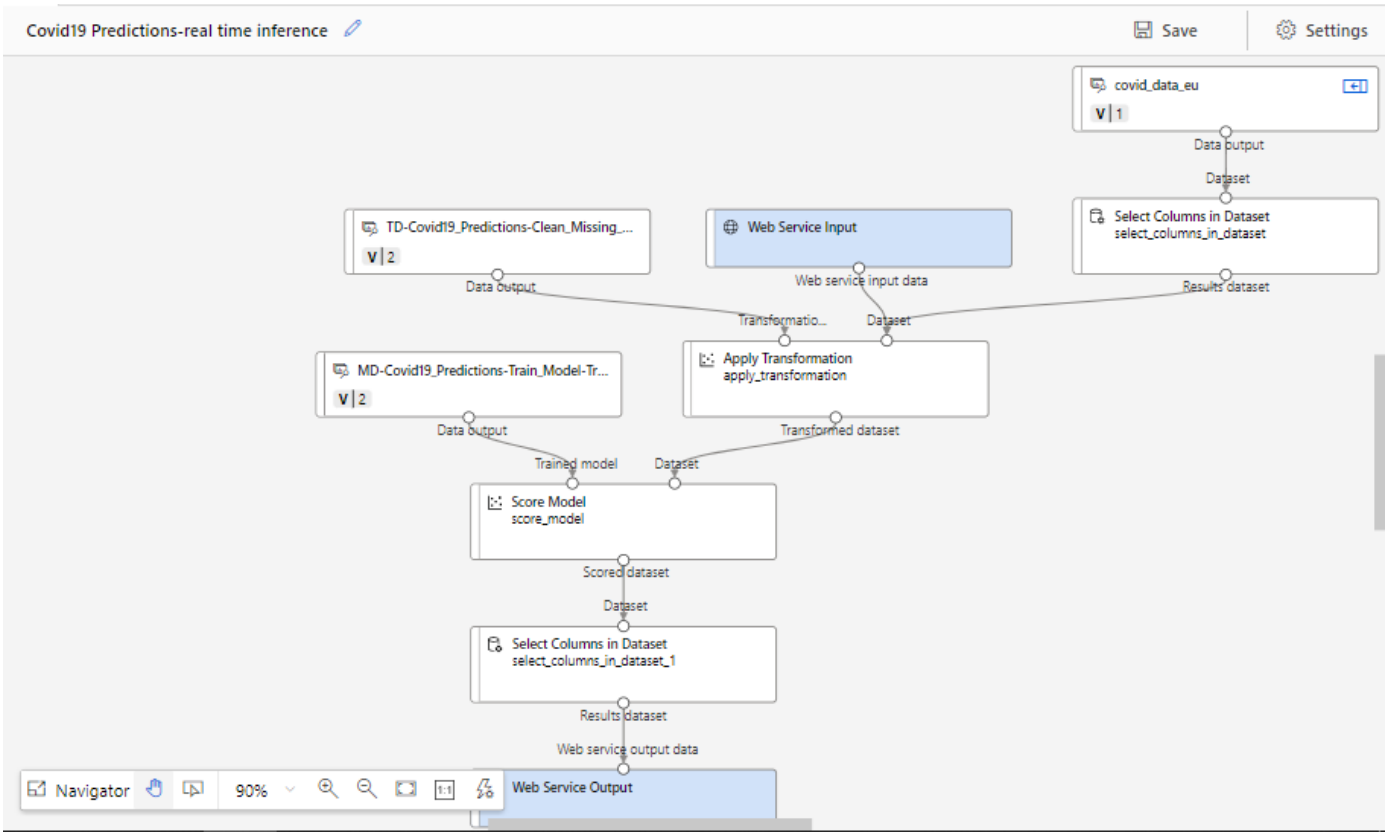

Figure 6.42: Real Time Inference Model

Covid dataset is given again but this time the columns selected are all except the new cases which are not required to be as inputs anymore to retrieve predictive results. The selected columns, a web service input and the cleaned missing data pipeline provided by the previous model all are connected to the apply transformation pipeline. When Azure finishes transforming this dataset it goes alongside with the trained model from the previous workflow to be scored and compared with the prior scored model. The last steps are to reselect the columns in the Dataset which is only one, the scored labels, and then it gets passed as a web service output. Both web service input and output are required so when it is exposed to an endpoint the user can pass an input and expect an output by the system.

Figure 6.43: Select Columns (Scored Labels only)

The model gets run by submitting it at the top and also can be evaluated if a new pipeline gets introduced by connecting the selected scored labels pipeline. The metrics everytime are quite similar with the previous model so it shows that the model is correct.

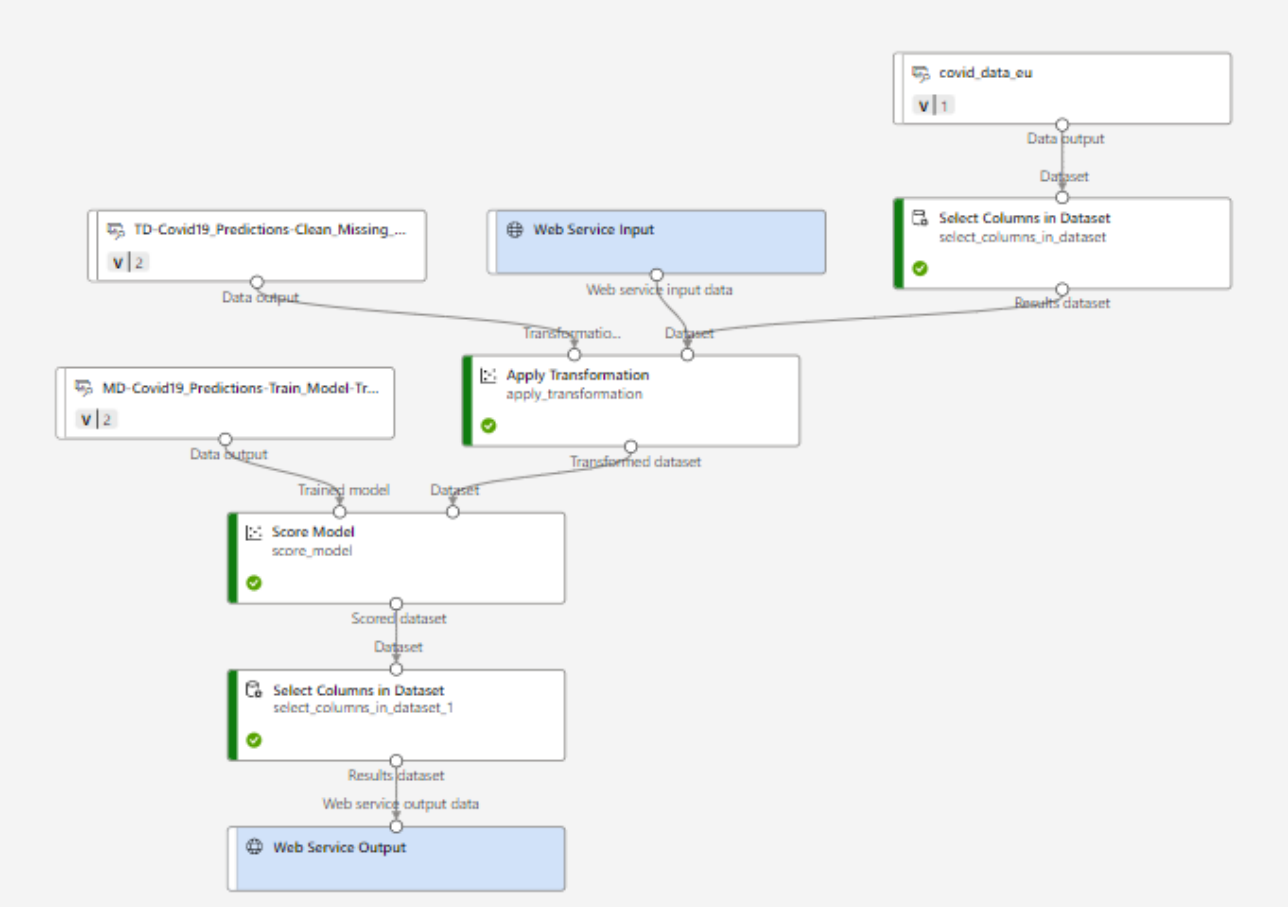

Figure 6.44: Successful Run of the Real-Time Inference

6.5 Endpoint

It is time to set up the real-time endpoint. The real-time endpoint requires a name but most importantly, the AKS compute created initially in order to be able to be deployed.

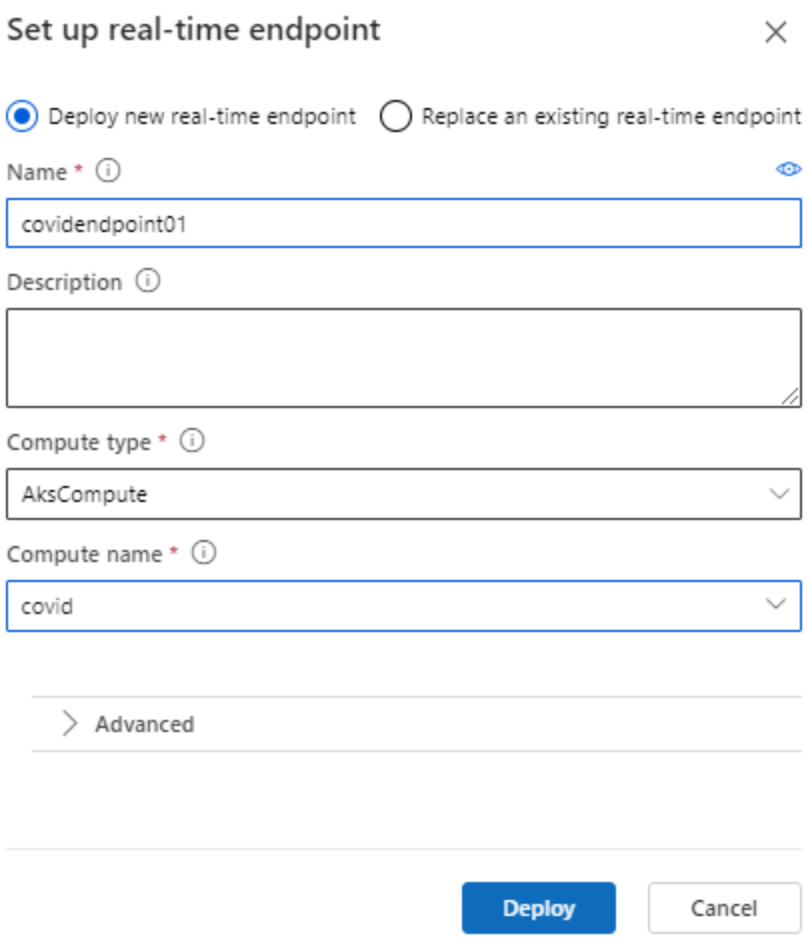

Figure 6.45: Real-Time Endpoint Deployment

The real time endpoint is in Transitioning Deployment state and then after waiting a few minutes to hours, the endpoint must be ready. It goes to Healthy Deployment state and has two more parts in the navigation: Test and Consume. Test allows the user of Azure to test their model manually by inputting values and getting back results. Consume is where all the details about the endpoint are such as, the REST endpoint, the primary/secondary API key and a consumption option for the developer to get back code that can be used in their web application in 3 different languages: Python, C# and R.

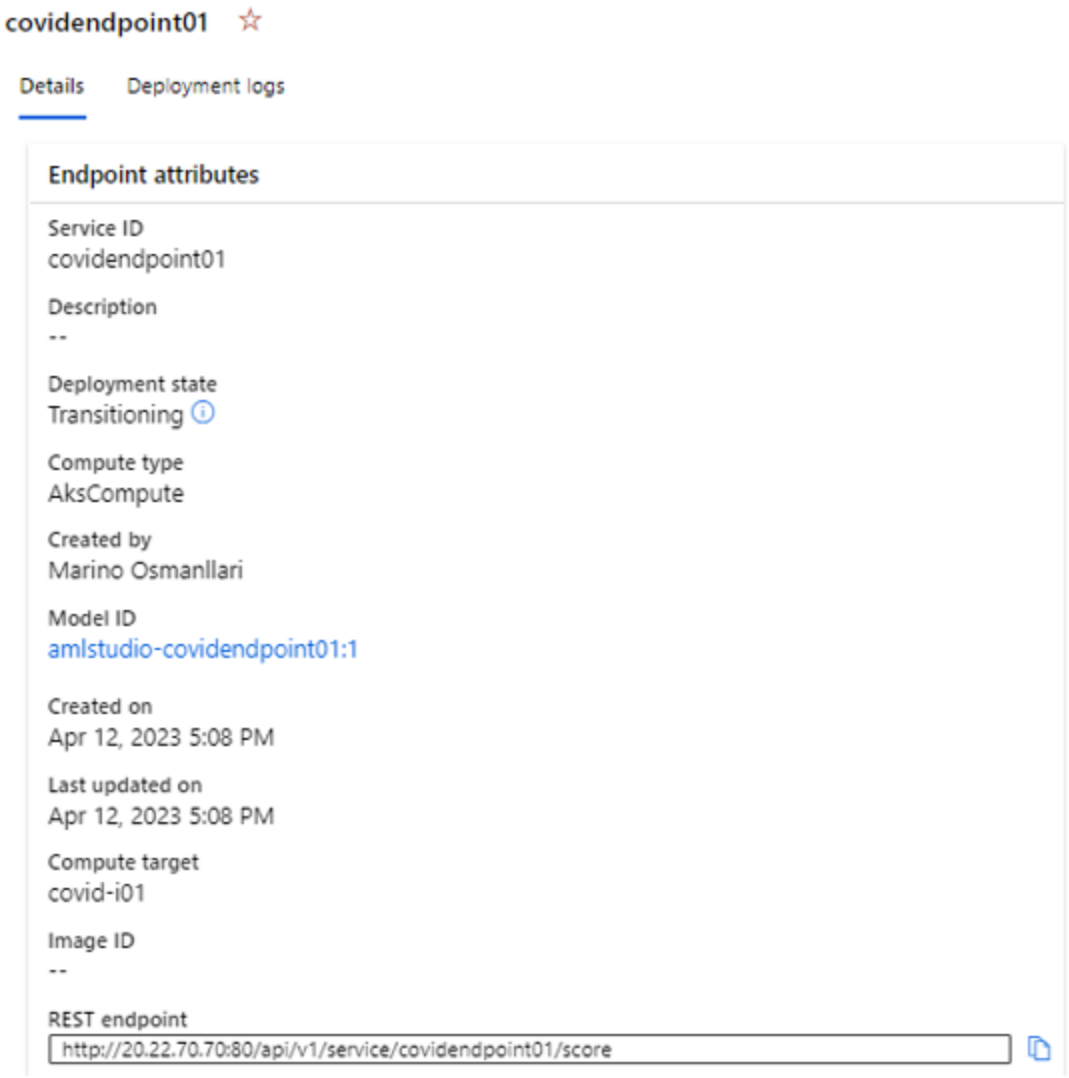

Figure 6.46: Endpoint Attributes

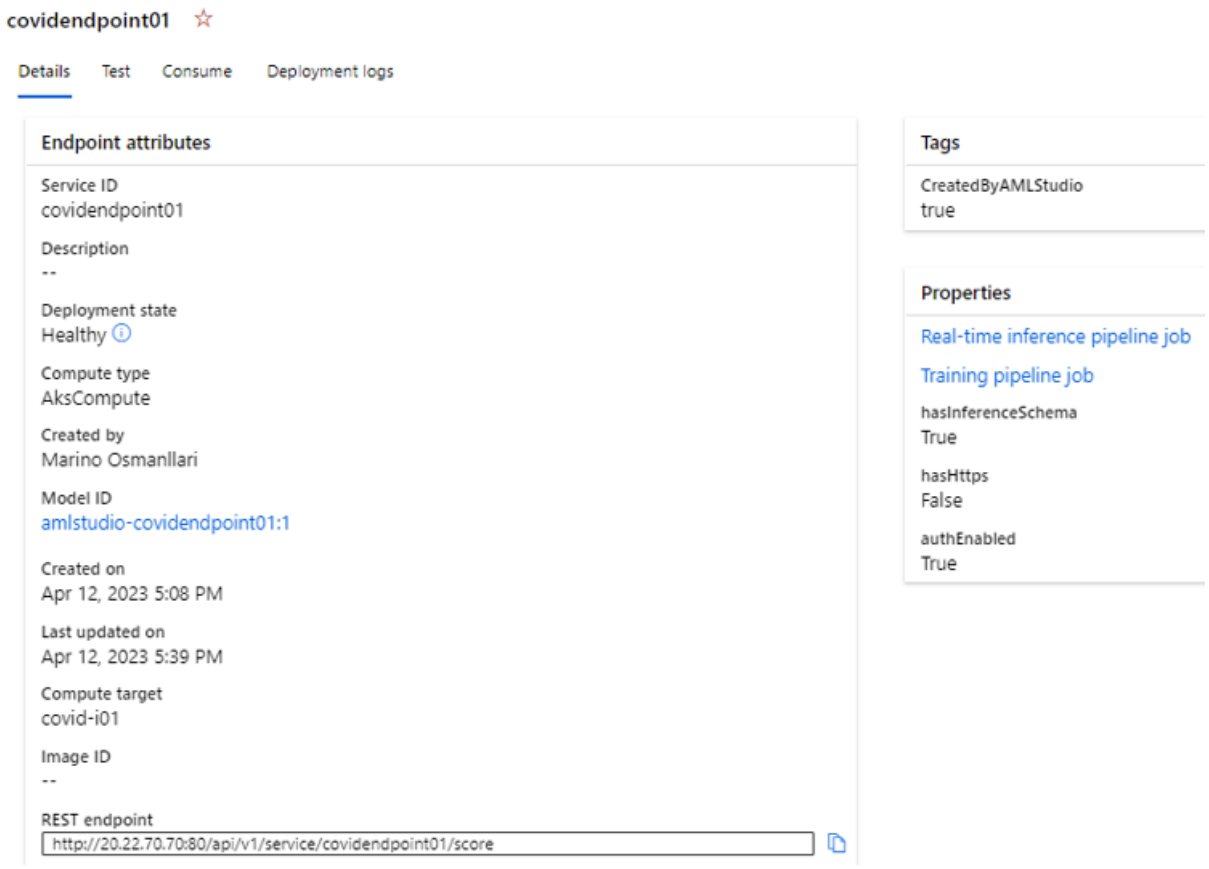

Figure 6.47: Healthy Deployment State

6.5.1 Endpoint Test

After deploying an endpoint in Azure ML, the Endpoint Test section can test the endpoint's functionality before making it publicly available. It can be tested if the endpoint is operating as intended by sending requests to it and seeing the results in this section.

It has a user-friendly interface for submitting test queries to the endpoint, including the ability to specify input data and examine the resulting predictions. For debugging and troubleshooting, the unprocessed HTTP requests and answers are easily inspected in the deployment logs.

In the beginning, the top 3 rows of the dataset are ready there to show the json format that should be followed to get back results. The Azure user can manually put new values and click test to get back fast results for the covid predictions of the model. The format in the figure 6.48 is always the same, JSON. In Test, we can put as many inputs with the only requirement which is to have the correct format, iso code, location, date and population.

A test is found in the figure 6.49.

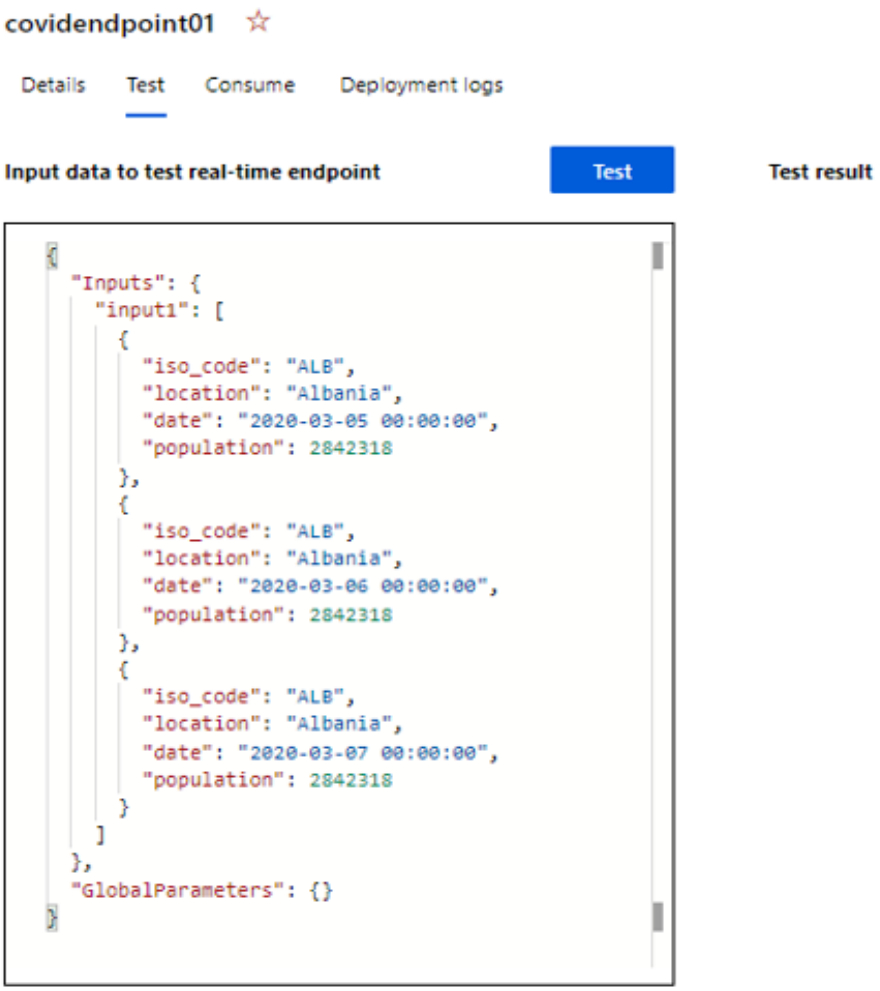

Figure 6.48: Endpoint Testing (Default inputs)

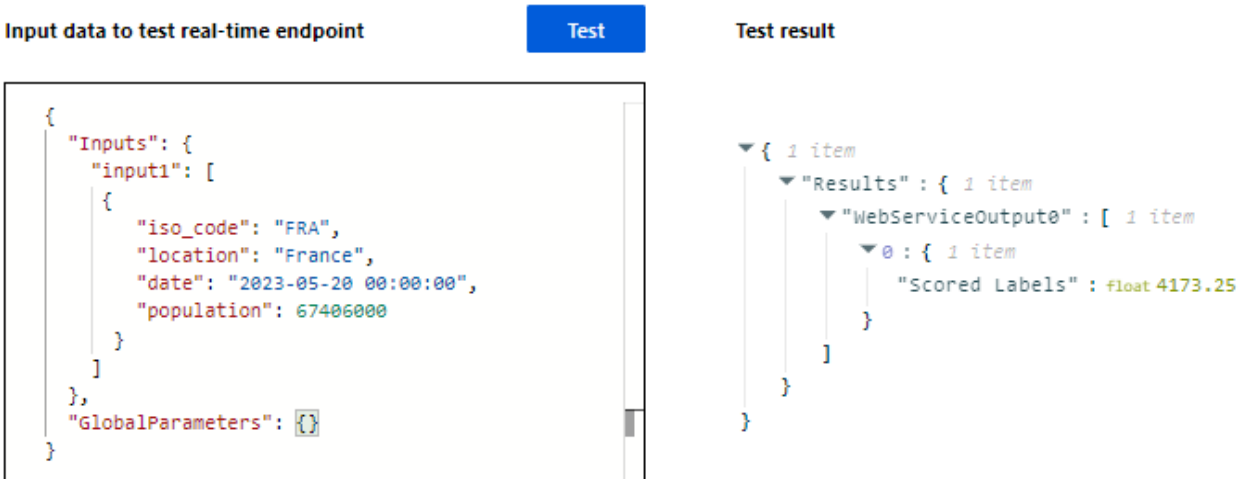

Figure 6.49: Endpoint Testing Result

6.5.2 Endpoint Consume

When implementing a machine learning model on Azure, an endpoint is required most of the time. By making the model accessible as a web service that can be utilized by devices or applications to provide predictions in real-time, it serves as a starting point for engaging with the model.

Depending on the particular needs and use case, endpoints may be generated in Azure using either Azure Machine Learning or Azure Functions. Highly performant endpoints that can handle a high amount of incoming requests may be deployed using Azure Machine Learning.

For increased speed or to handle fluctuating numbers of incoming requests, Azure endpoints may be expanded either horizontally or vertically. In order to manage greater traffic, the endpoint can be scaled vertically or horizontally. Vertical scaling entails increasing the resources allotted to each instance, such as CPU or memory.

The cost of Azure endpoints is determined by a variety of variables, including the volume of requests, the size of the deployed model or even the degree of performance moderated by the Virtual Machines. It is crucial to thoroughly weigh the pricing alternatives and select the strategy that is best for any particular use case.

It is crucial to properly test and validate an Azure endpoint before releasing it to production to make sure it is operating as intended and to find any problems or issues.

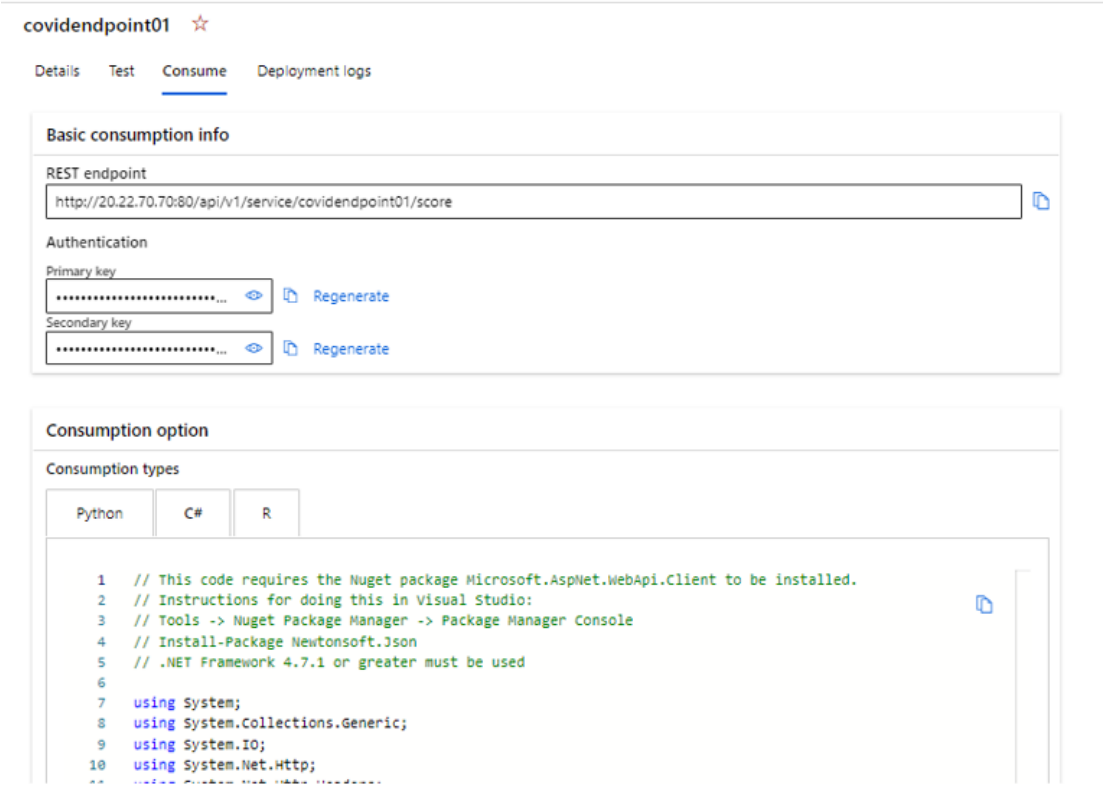

Figure 6.50: Endpoint Consummation info and option

From the Endpoint Consume, it is required to retrieve the whole C# code produced by Azure and pasted to an Invoker folder in the application's code that will be discussed in the later chapters.

Chapter 7

7. Web Application

A web application was created utilizing Blazor and ASP.NET in addition to working on Azure Machine Learning Studio to build models and make predictions. As the application abstracts away the challenges of Azure technologies, the integration of these technologies, allows users to be able to estimate new future COVID-19 cases that may affect European nations without having the need to be familiar with Azure technology or run any tests directly on the Azure platform to leverage the prediction capabilities.

The web application provides a user-friendly interface that makes the prediction process simpler by utilizing the capabilities of Blazor and .NET. Users can choose the country and future date and then the system receives predictions for new COVID-19 cases using the pre-trained models developed on Azure Machine Learning Studio.

7.1 Reason of the Technologies Selected

Blazor and .NET Framework offer several advantages for obtaining predictions from an Azure Machine Learning model connected to an endpoint.

Both Blazor and .NET Framework utilize C# as their primary language. C# is a very sophisticated and expressive programming language that possesses a diverse set of tools/features and capabilities for developing robust applications. By leveraging C# throughout the application stack, from the frontend to the backend, a language consistency is maintained to minimize the learning curve, allow faster development and reduce complexity.

Blazor and .NET Framework have excellent integration with Azure services, including Azure Machine Learning. This integration enables connection to the Azure Machine Learning model's endpoint and efficiently consumes its predictions within this Blazor application. With built-in Azure SDKs and libraries, authentication is easily completed, communicating with the endpoint and handling the prediction responses [105].

Reusable UI components and services were other reasons why the technologies used reduced development time and effort. Additionally, the rich ecosystem of libraries and tools available for .NET Framework provides extensive support for various development tasks, such as data manipulation, serialization, and API integrations, further enhancing productivity [105].

The .NET Framework is known for its scalability, allowing applications to scale as the user base grows but in this case it was not a priority because the product is confidential and used by a very small number of people that work for Health Organizations or Governments.

However, one significant drawback of Blazor is its relative newness as a programming language, which results in a less established community support compared to more established languages such as React or Angular. This limited community support can be attributed to the fact that Blazor, being a newer language, still has unresolved errors or issues that have yet to be addressed or fixed. Consequently, developers using Blazor face challenges in finding comprehensive documentation and a large community to rely on for assistance and guidance.

7.2 UI Analysis

7.2.1 Home Page

The home page of our web application serves as the central hub. It gives users an overview of the application's features, capabilities and also offers easy navigation to relevant sections. The page has a simple layout and informative cards, each of which explains a different component of the application.

Upon landing on the home page, users are greeted with a brief introductory paragraph highlighting the application's purpose and benefits for data scientists and health organizations. This paragraph emphasizes the ability to obtain predictions from machine learning models connected to Azure and the positive impact it can have on decision-making and patient care.

Beneath the introductory paragraph, users will find a set of interactive cards, each representing a navigation link to different sections of the application. These cards are designed to guide users and provide a clear understanding of the available functionalities. Clicking on a card button will direct users to the respective section or external link to the global COVID-19 data provided by Institute for Health Metrics and Evaluation.

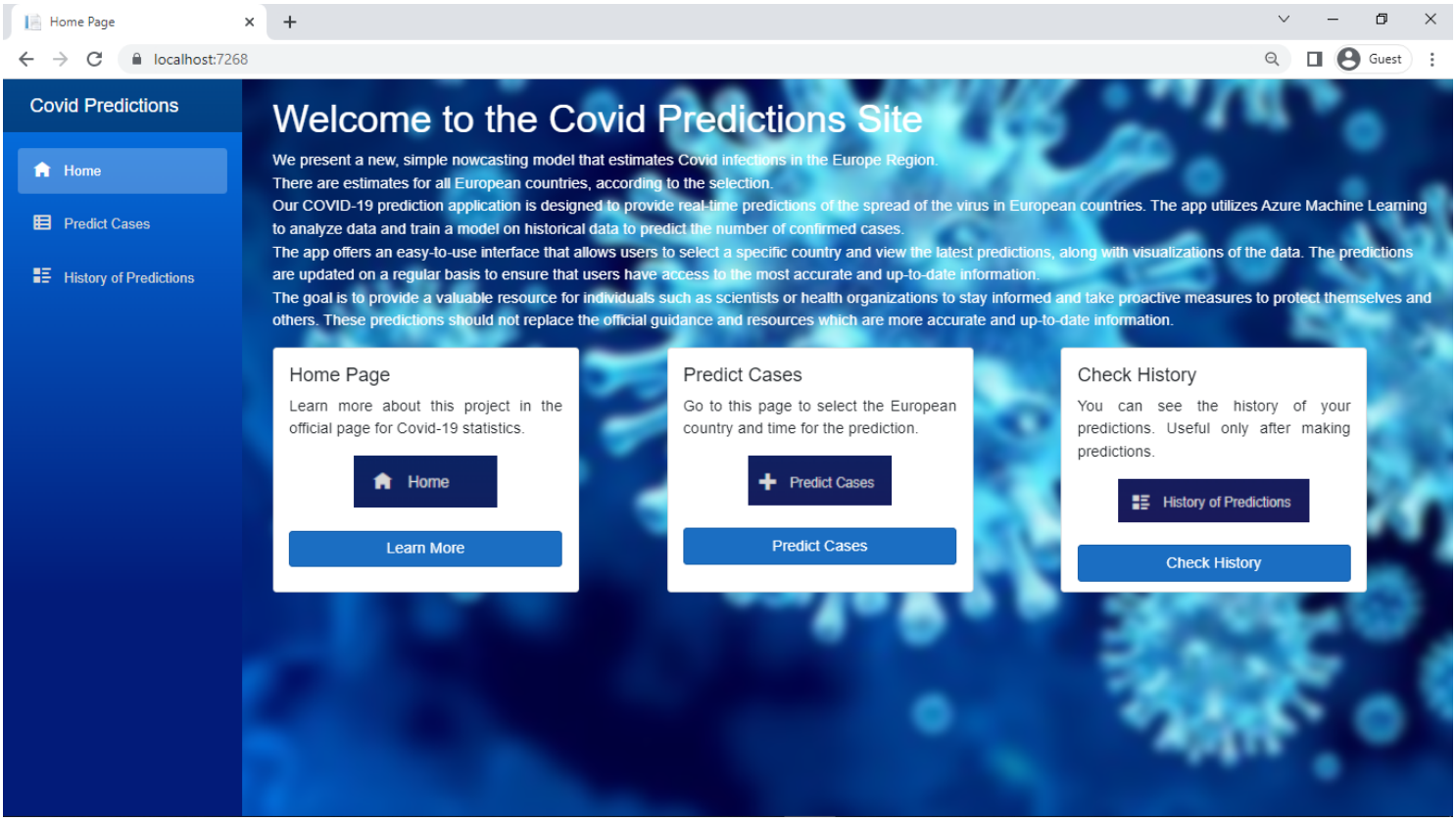

Figure 7.1: Home Page

7.2.2 Predictions Page

The Prediction page is the most important part of the web application which provides a straightforward interface to generate COVID-19 case predictions using Azure Machine Learning. This page features four input boxes: Country, ISO code, Population and Date. By inputting the relevant information into these fields, it can derive accurate predictions for new COVID-19 cases for all the European countries obtained by the initial dataset of Azure.

In the "Country" input box there are 44 countries ready to be selected. By selecting the desired country from a dropdown menu the Iso Code and Population is filled in automatically by the system so that the user does not have to or unintentionally provide false information which does not correspond to the data run in Azure ML Studio. Population helps to contextualize the prediction in relation to the population size. So mostly, large countries will probably estimate a higher amount of new cases.

Lastly, the "Date" input box allows users to specify the future date for which they want to generate predictions for the country selected. There are restrictions for this input. There cannot be “predictions” or “forecasts” if the date is in the past of the real current day. Hence, the current day is the minimum and maximum is the end of the year 2030 to keep the product relevant and the model validity.

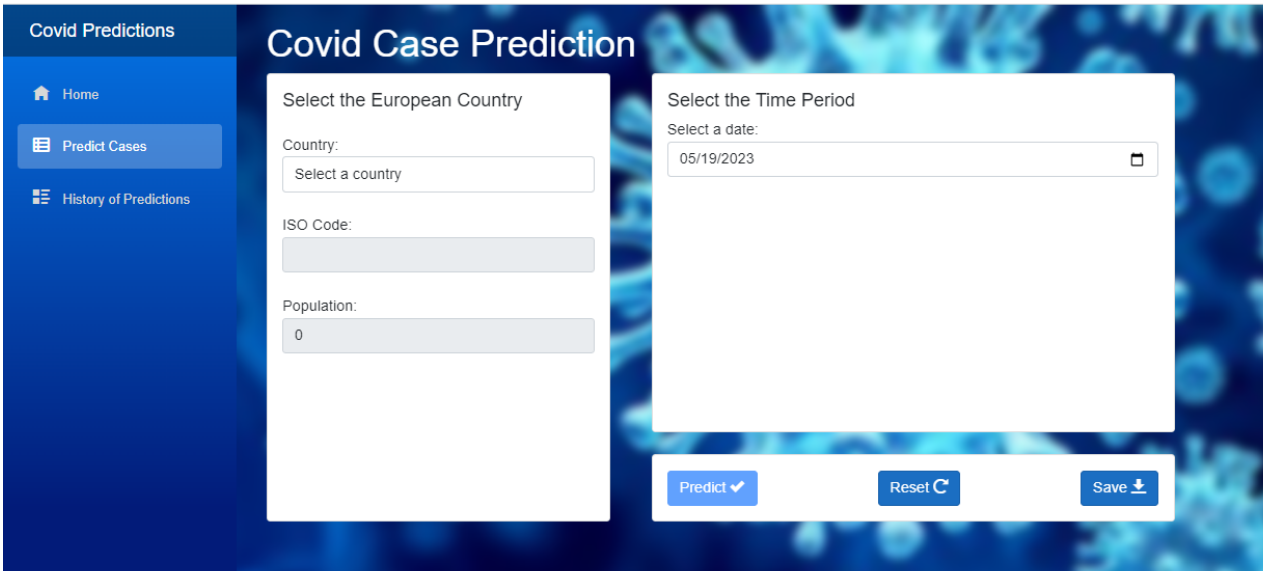

Figure 7.2: Predictions Page

Once the necessary inputs have been provided, the system can initiate the prediction process once the "Predict" button is clicked. The application then leverages the pre-trained Azure Machine Learning model to compute and derive accurate predictions for COVID-19 cases based on the provided inputs. An instant result is displayed to the user who provided the inputs.

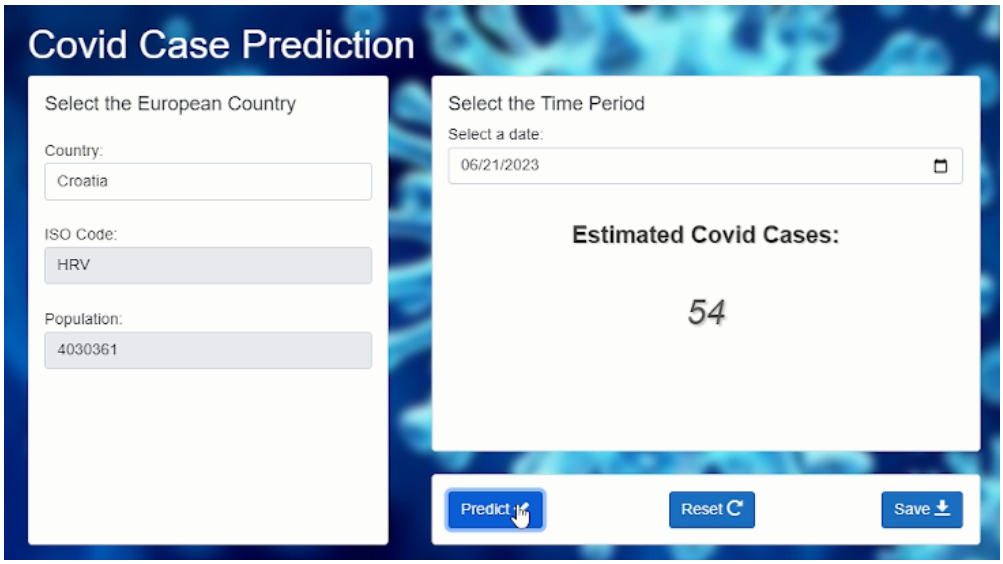

Figure 7.3: Estimation of Covid Cases

The Prediction page empowers users to obtain real-time predictions conveniently, without requiring in-depth knowledge of Azure technologies or complex model deployment processes. By simplifying the input process and leveraging the power of Azure Machine Learning, it can give valuable insights into the potential trajectory of COVID-19 cases, aiding in decision-making, resource allocation, along with proactive planning.

7.2.3 History Page

The History page of the web application provides a comprehensive record of predicted COVID-19 cases for different countries. This page displays five columns of data derived from the inputs provided on the Predictions page: country, ISO code, population, date and predicted cases. The History page serves as a valuable reference and allows users to save the predictions.

Each column indicates the data provided by the previous page. They all display what the user selected and predicted from the Azure response. They all serve as unique identifiers.

Users can refer to the History page to review and analyze the trends and patterns in the predicted cases. This saved data can be valuable for monitoring the accuracy of predictions over time, identifying any significant changes or deviations, and finding out trends. It additionally serves as a starting point for comparisons and analysis in the future.

The History page gives users an overview of their prior forecasts and helps them to make decisions based on the recorded data by organizing and making this information easily accessible.

Figure 7.4: History of Predictions Page

Unfortunately, the History page is not functional. The predictions and all the other information is not recorded and saved successfully, due to time constraints in the development process.

The focus may have been primarily on implementing core functionalities as a prime example the prediction generation from Azure models and user interface design.

7.3 Technical Details

Initially, the provided code from listing 7.1 below should be included in a separate folder to make the connection between the app and the endpoint.

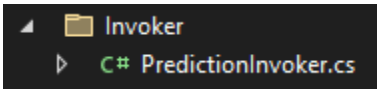

Figure 7.5: Microsoft Visual Studio Invoker Folder

Since it has a C# code, it makes the job way easier. It is a POST request to a web service endpoint and response handling C# console application. The HTTP request is sent using the HttpClient class from the System.Net.Http namespace. Some requirements are, having installed NuGet package Microsoft.AspNet.WebApi.Client, Newtonsoft.Json, and have 4.7.1 .Net Framework version or higher, as the top of the code comments mention.

```
// This code requires the Nuget package Microsoft.AspNet.WebApi.Client to be
installed.
// Install-Package Newtonsoft.Json
// .NET Framework 4.7.1 or greater must be used
```

Listing 7.1: NuGet Instructions

The asynchronous function called ‘InvokeRequestResponseService’ is in charge of calling the web service, found in the Appendix. To avoid certificate validation, it creates an instance of the HTTP client and configures it with a special HTTP client handler. The JSON payload that will be submitted as the request body is included in the requestBody variable [105].

The API key or token needed to authenticate the request is stored in the ‘apiKey’ variable and must be replaced with the primary/secondary key or AMLToken for the endpoint. It is set as the HTTP request's Authorization header. The content type is set to "application/json" and the request body is serialized into a ‘StringContent’ object.

The client.PostAsync sends the POST request and serialized JSON payload is sent to the given endpoint using the PostAsync function. The server's answer is represented as a ‘HttpResponseMessage’ object that is returned. If the response is successful, a string representation of the response's content is read and written to the console through ‘response.IsSuccessStatusCode’ otherwise an error will be displayed.

When namespaces are imported into C# code, the @using directives are used to make the types and members specified in the imported namespaces accessible inside the current code file. Every import for the whole covid predictions project goes to _Imports.razor. For instance, here are the imports for the project:

```
@using System.Net.Http
@using Microsoft.AspNetCore.Authorization
@using Microsoft.AspNetCore.Components.Authorization
@using Microsoft.AspNetCore.Components.Forms
@using Microsoft.AspNetCore.Components.Routing
@using Microsoft.AspNetCore.Components.Web
@using Microsoft.AspNetCore.Components.Web.Virtualization
@using Microsoft.JSInterop
@using Covid_Predictions
@using Covid_Predictions.Shared
```

Listing 7.2: _Imports.razor File

Judging by the namespace names they provide classes for Http requests, authorization, routing etc.

Now the most crucial part of the whole application has to be the GetPredictionsAsync() method located in the Predictions.razor file. The code snippets provided, retrieve predictions from the Azure ML service. The whole method is located in the Appendix.

- It sets up an HTTP client and adds the API key to the request headers for authentication.

```
const string apiKey = "s5GxetTYeKJeNDeR0H4lAf6yXac0ldlg";
var client = new HttpClient();
client.DefaultRequestHeaders.Authorization = new
AuthenticationHeaderValue("Bearer", apiKey);
```

Listing 7.3: Set Up of HTTP client

- It constructs the input data payload, which includes an array of input objects. Each input object contains the properties: `iso_code`, `location`, `date`, and `population`, which are used as input features for the prediction.

```
var inputObjects = new[]{
   new{
      iso_code = IsoCode,
      location = SelectedCountry,
      date = SelectedDate.ToString("yyyy-MM-dd HH:mm:ss"),
      population = Population
    }
};
```

Listing 7.4: Construct of the input data payloads

- The payload is serialized into a JSON string using `JsonSerializer.Serialize()`. Then an instance of `StringContent` is created, which represents the HTTP request body with the JSON payload.

```
var payload = JsonSerializer.Serialize(new
{
   Inputs = new{
       input1 = inputObjects
   },
   GlobalParameters = new { }
});
```

Listing 7.5: Payload Serialization

- The HTTP request is sent to the Azure ML service using `client.PostAsync()`, including the endpoint URL and the content. The response from the Azure ML service is obtained as a string using `response.Content.ReadAsStringAsync()`.

```
var response = await client.PostAsync
("http://20.22.70.70:80/api/v1/service/covidendpoint02/score", content);
var result = await response.Content.ReadAsStringAsync();
```

Listing 7.6: Sending HTTP Request and get back Response

- The response string is parsed as JSON using `JsonDocument.Parse()`. The parsed JSON is traversed to extract the prediction result. It looks for specific properties and arrays within the JSON structure to retrieve the desired prediction information.

```
try{
   JsonDocument jsonDoc = JsonDocument.Parse(result);
   JsonElement root = jsonDoc.RootElement;
   JsonElement resultsElement;
    if (root.TryGetProperty("Results", out resultsElement)){
        JsonElement webServiceOutput0Element;
        if (resultsElement.TryGetProperty("WebServiceOutput0", out
webServiceOutput0Element) && webServiceOutput0Element.ValueKind ==
JsonValueKind.Array){
                        JsonElement scoredLabelsElement;
                        // … …
                    }
     }
```

Listing 7.7: Checking For the result inside every Array

- If the prediction result is successfully extracted, it is assigned to the `predictionResult` variable otherwise exceptions are handled and printed to the console.

```
private async Task GetPredictionsAsync(){
   try{ … …
      try{ … …

            if (webServiceOutput0Element[0].TryGetProperty("Scored Labels", out
      scoredLabelsElement)){
          predictionResult = scoredLabelsElement.GetRawText();
          Console.WriteLine(predictionResult);
          }
           catch (JsonException ex){
               Console.WriteLine($"Error deserializing response:{ex.Message}");
           }

       }
      catch (Exception ex){
          Console.WriteLine($"Error executing Azure ML service:{ex.Message}");
          }
      }
```

Listing 7.8: PredictionResult and exceptions handled

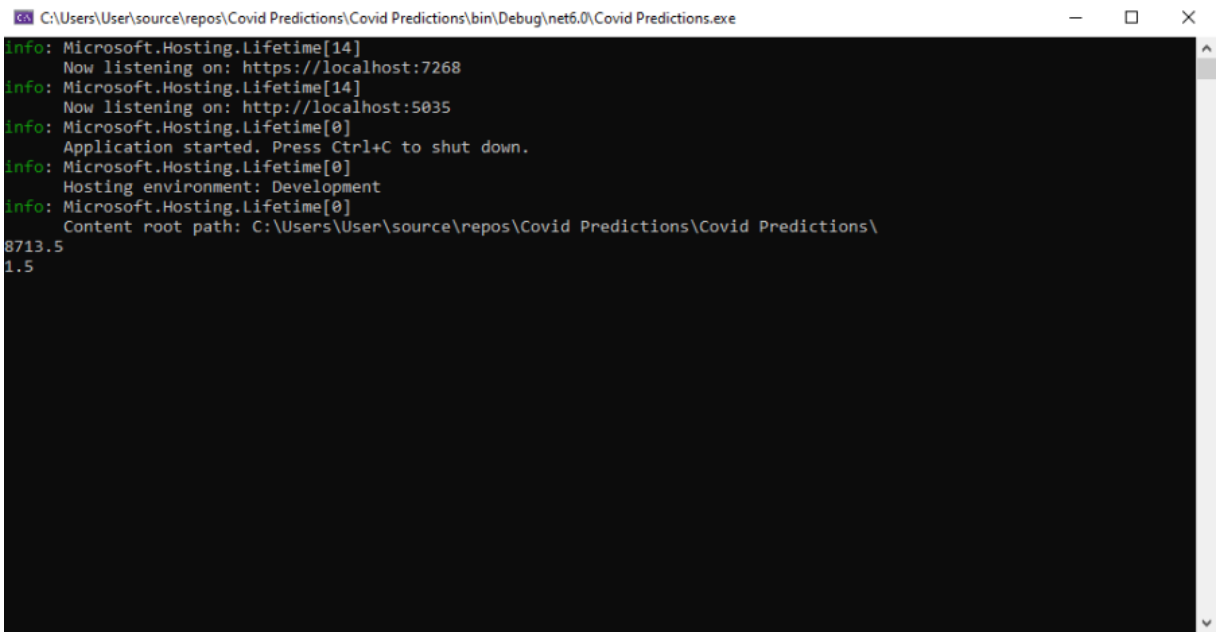

Figure 7.6: Terminal/Console Deserialized Response

Chapter 8

8. Testing

An Azure Machine Learning project's success and efficacy depend heavily on testing. It aids in validating the performance, accuracy, efficiency, and dependability of the machine learning model, ensuring that it generates reliable results.

The final testing split into 2 parts, Azure Testing and Web Application Testing, and was performed mainly manually.

8.1 Test Plan

Before diving into the testing specifications, a testing plan is constructed to show some of the test types with description and additionally the actual and expected results.

A well-structured testing strategy assists in methodically detecting possible issues, assuring extensive test coverage, and allowing for unambiguous comparison of actual and expected results.

Below is a test plan created by the developer:

Nr.	Test Type	Description	Expected Result	Actual Result	Comments
1	ML-Evaluation Coefficient of Determination	Measure CoD to show how fit is the model	Close to 1	0.8762	Pass
2	ML-Evaluation of 5 Metrics	Measure the 5 Metrics between different parameters of the same Algorithm	Default as the best option	Default is the best option	Pass
3	Endpoint-Input Schema	Use the most important inputs to predict a model	Figure 8.1 inputs	Removed total cases and new cases	Proven to be more accurate with those details
4	Job Experiments -	Train Job Experiments to lower	Less than 10s after	4-6 seconds	Pass

	Duration	their duration	initial train		
5	Job Experiment-Min-Max	Use statistics to achieve generalization and accuracy	Min: 0 Max: < 4% of EU countries avg.	Min: 0 Max: ≈ 410000	Pass
6	ML- Changing Data	Update EU covid data to achieve flexibility	Very low deviations with previous predictions	Low Deviations	Pass
7	APP- HTTP request validity	Integrity between the web application and Azure ML through endpoints	Code number 200 to indicate that the request was successful	Code 400 fixed to Code 200	Pass
8	APP- Input Consistency with Azure	Input schema should be applied and executed by the web service inputs	HTTP input request to be consistent with Azure model	Consistent values	Pass
9	APP- NUnit Testing	Unit Testing Framework for C#	Pass All Checks	1 Failed Test	Pass, with an exception explained later

Table 8.1: Test Plan

8.2 Azure Testing

For an Azure Machine Learning project, testing is crucial for the following reasons:

Accuracy

To generate predictions on completely new, unforeseen data, machine learning models are trained on past data. By contrasting the predictions with actual results, testing evaluates the model's precision and validates its performance. It enables the developer to fine-tune and enhance the model's accuracy by assisting in the identification of any inconsistencies, biases, or restrictions. Validating the model endpoint's input and output is the first step. The input data that is submitted to the endpoint is checked to be in the right format, right input scheme and

correspond to the desired data types. The output from the endpoint inevitably matches the required format.

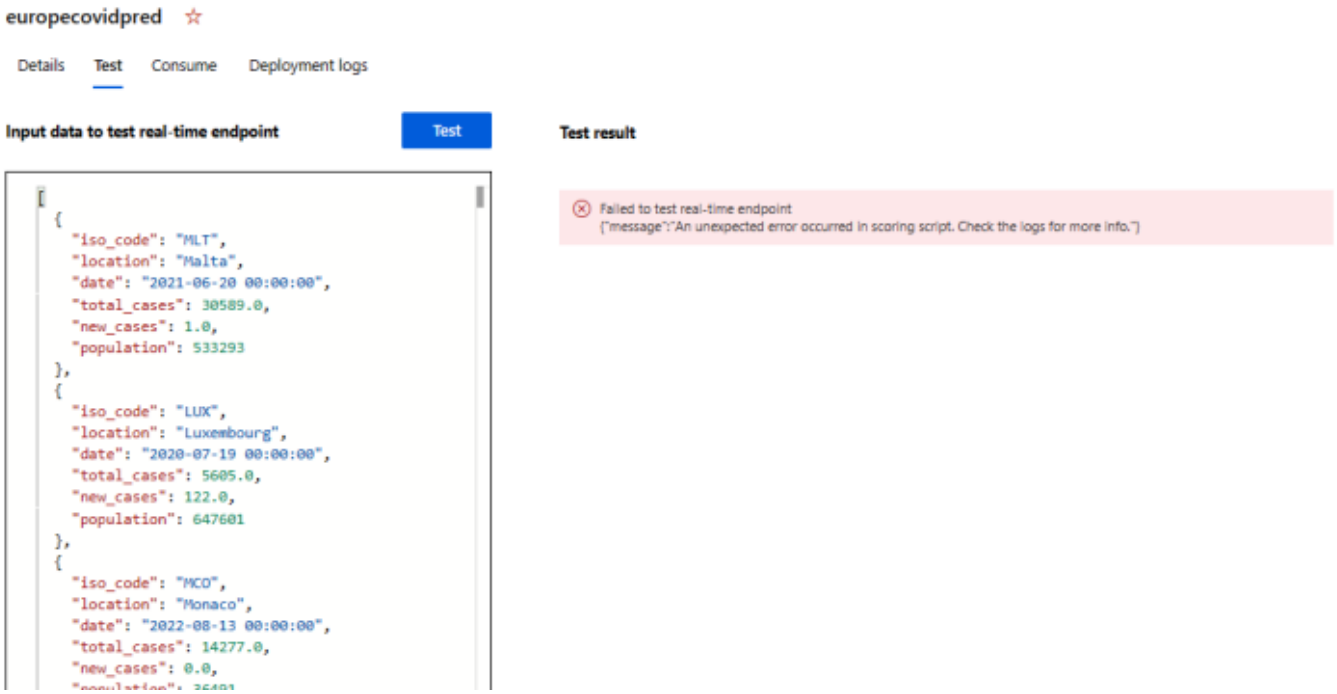

Figure 8.1: Test Result Fail

Discoverability

Testing aids in the discovery and correction of mistakes, flaws, or unexpected behaviors in the deployment process or the model. Identified problems can impact the model's functionality or the accuracy of the findings by executing a range of test scenarios. Early error identification gives the chance to make the required corrections, raising the project's overall quality. Functional testing is necessary to ensure that the model and endpoint acts as anticipated. To guarantee that the model generates reliable predictions, several situations and edge cases were tested such as completely different countries population-wise or dates. Then, the output is matched with the estimated cases by testing with various input data.

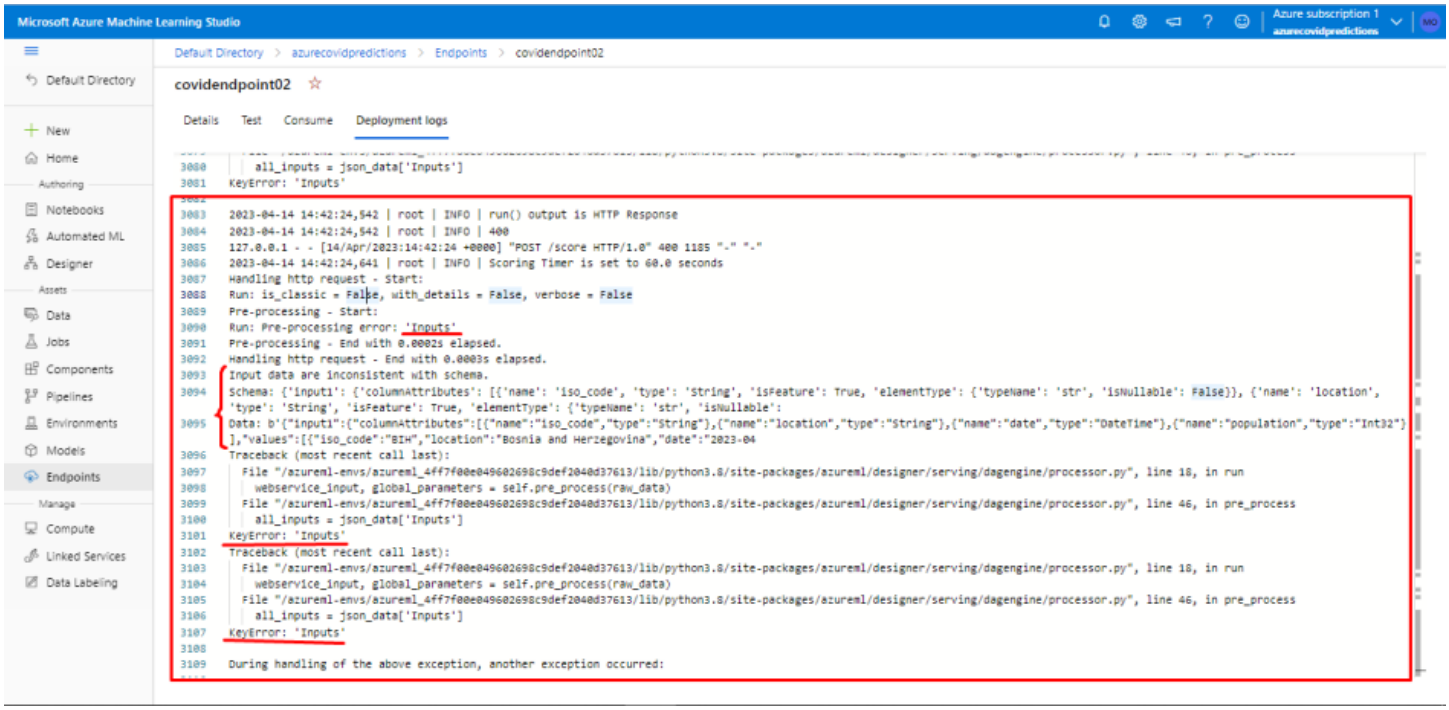

Figure 8.2: Pre-processing Input Schema Error

Configuration

Covid Predictions project integrates a number of components, including data pretreatment, model training, and deployment infrastructure. Through testing it ensured the appropriate interaction and communication between these components. By detecting any configuration issues, data inconsistencies, or compatibility issues, integration testing assures the smooth running of the whole project. Usually the configuration issue would come from the connection of the selected columns of the dataset with the python script that crashed or made the whole model fail.

Efficiency

Efficiency is a strong aspect that has been assessed in this Azure Machine Learning project. This involves evaluating the model endpoint's scalability, throughput and response time under various workloads. Possible bottlenecks were found all the time, but with allocated resources working more efficiently, it was made sure that the model satisfies the necessary performance standards by doing performance testing.

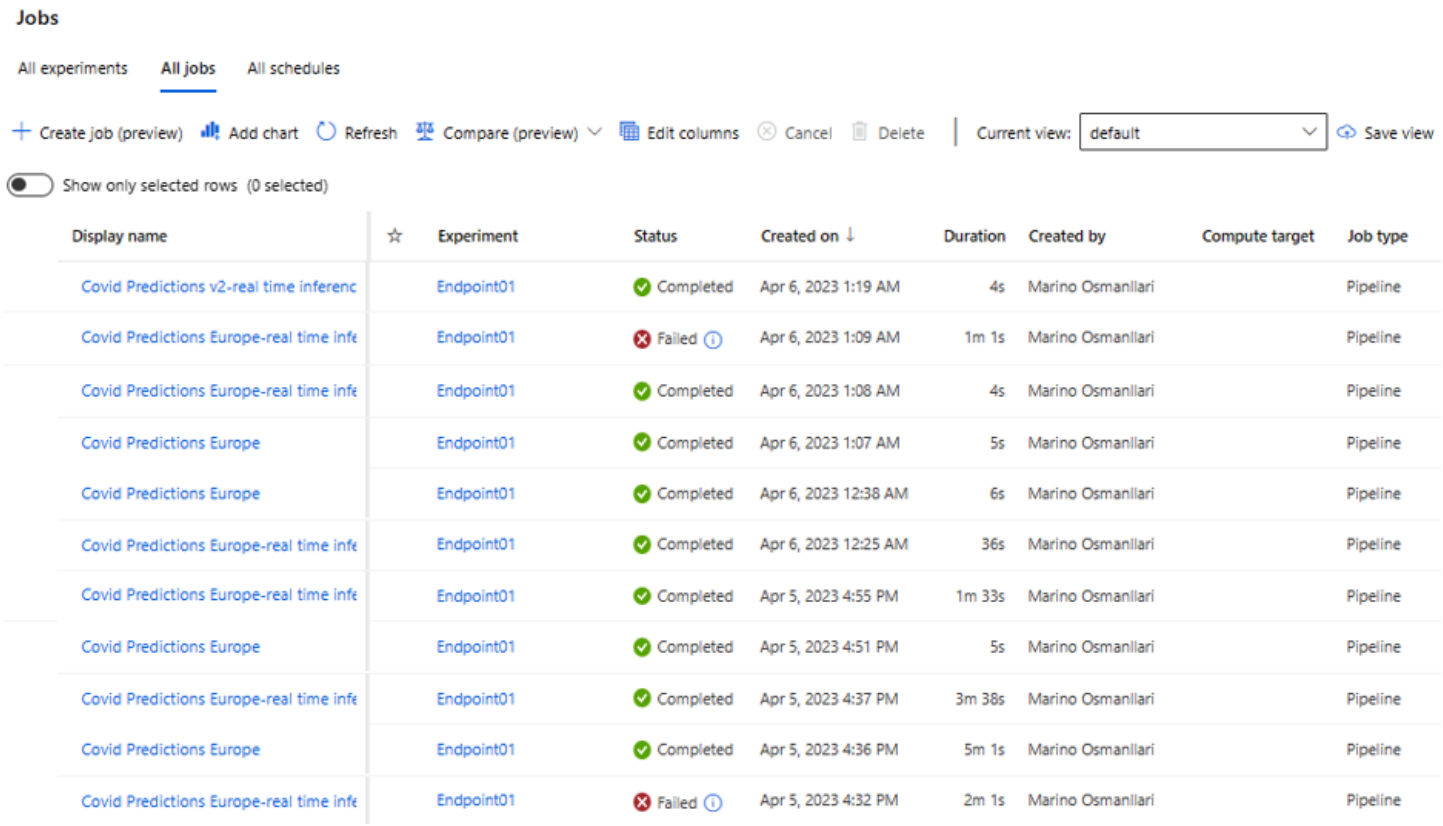

Figure 8.3: Completed and Failed Pipeline Jobs

Generalization

For machine learning models to be deemed robust and dependable, they must perform well on newly introduced data. The generalization and scenario handling skills of the model are evaluated by testing on a wide range of sample datasets. It helps to eliminate the possibility of overfitting or underfitting and guarantees that the model works consistently across various data distributions. So, since there are more than 40 european countries in the dataset, this was an easy task which was solved by itself running the model in Azure to check for problems such as bad metrics or wrong values. For example, wrong values for minimum cases, maximum cases, median and mode.

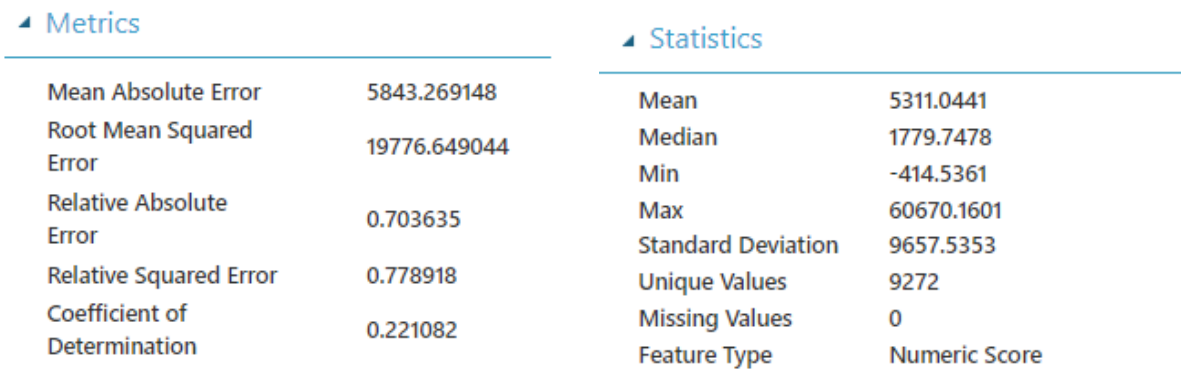

Figure 8.4: Metrics and Statistics Mistakes

Flexibility

Moreover the ability to adjust to changing data was extremely easy for this cloud provider. Real-world data is dynamic and subject to modification. They evaluate the model's ability to accommodate new or changing data through testing which is quite simple because even when new data comes the model still operates the same way and also improves the performance by validating the predicted cases for the future dates are utterly close. The model's applicability and relevance as fresh data become available are provided by its flexibility.

8.3 Application Testing

Testing is required as well for the web application connected to Azure Machine Learning.

Data is verified that is being delivered in the POST request follows the specified formats and data types by performing input data validation. Before processing it with Azure Machine Learning, the validation of the data was done on the server side.

The endpoint that receives the POST request is checked to verify if it responds successfully and delivers the appropriate HTTP status codes, for instance, 200 for success, 400 for a faulty request, etc.

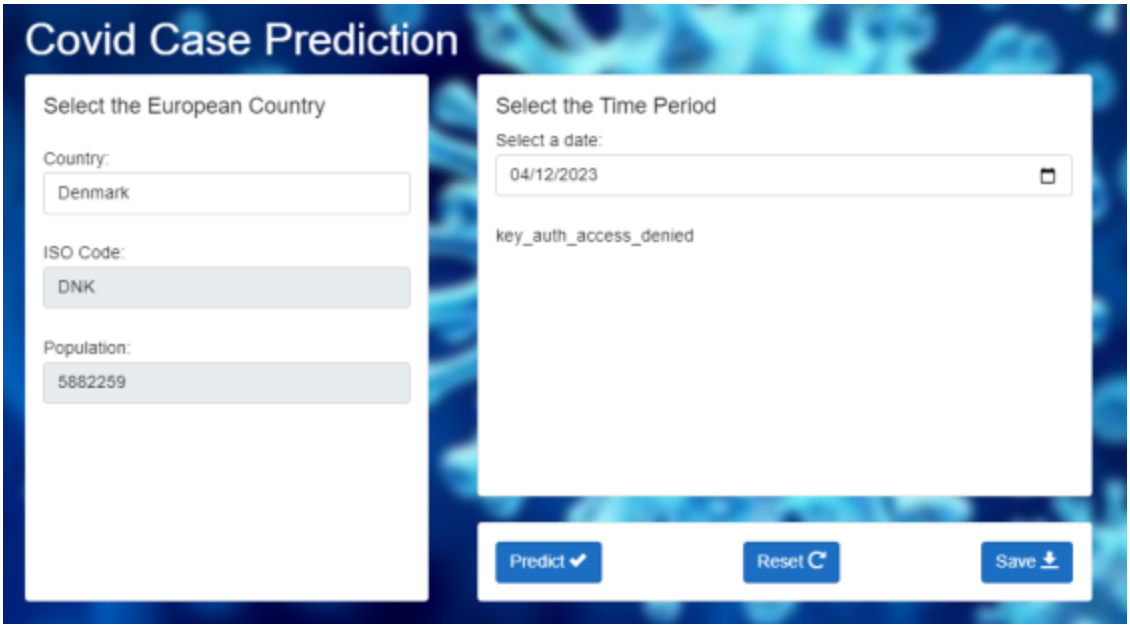

Figure 8.5: Key_auth_access_denied Error

This is one of the errors with code 400 which showed inconsistency with the input data sent to Azure with the POST request.

```
{"error": {"code": 400, "message":
"Input Data Error. Input data are inconsistent with schema.\nSchema:
{'input1': {'columnAttributes': [{'name': 'iso_code', 'type': 'String', 'isFeature': True,
 'elementType': {'typeName': 'str', 'isNullable': False}},
  {'name': 'location', 'type': 'String', 'isFeature': True,
 'elementType': {'typeName': 'str', 'isNullable':
 \nData: b'{\"iso_code\":\"HUN\",\"location\":\"Hungary\",\"date\":\"2023-4-8\",
\"population\":9967304}
'\nTraceback (most recent call last):\n File
\"/azureml-envs/azureml_4ff7f00e049602698c9def2040d37613/lib/python3.8/site-packages/azureml
/designer/serving/dagengine/processor.py\",
 line 18, in run\n webservice_input, global_parameters = self.pre_process(raw_data)\n File
\"/azureml-envs/azureml_4ff7f00e049602698c9def2040d37613/lib/python3.8/site-packages/azureml
/designer/serving/dagengine/processor.py\",
line 46, in pre_process\n all_inputs = json_data['Inputs']\nKeyError: 'Inputs'\n", "details": ""}}
```

Listing 8.1: Error code 400

Most of the tests with the focus on integrity or performance are done in the application however they are checked on the endpoint deployment logs provided by Azure ML Studio.

Integrity checking had to do with tests run to indemnify that the app and Azure Machine Learning are integrated. Sending test data to Azure and ensuring that the predictions made and delivered are accurate were much necessary for this.

```
4980  Predicting regression value - End with 0.0049s elapsed.
4981  BaseLearner._predict - End with 0.0050s elapsed.
4982  Successfully predicted.
4983  Regression Model Scored Columns:
4984  There are 1 score columns: "Regression Assigned Labels"
4985  ScoreModelModule.run - End with 0.0141s elapsed.
4986  Executing node 2: Score Model - End with 0.0156s elapsed.
4987  Executing node 3: Select Columns in Dataset - Start:
4988  Return without parsing
4989  Parse ColumnSelection parameter
4990  SelectColumnsModule.run - Start:
4991  Select column indexes from Dataset
4992  SelectColumnsModule.run - End with 0.0010s elapsed.
4993  Executing node 3: Select Columns in Dataset - End with 0.0027s elapsed.
4994  Processing - End with 0.0332s elapsed.
4995  Post-processing - Start:
4996  Post-processing - End with 0.0000s elapsed.
4997  Handling http request - End with 0.0336s elapsed.
4998  2023-04-17 14:33:20,581 | root | INFO | run() output is HTTP Response
4999  2023-04-17 14:33:20,581 | root | INFO | 200
5000  127.0.0.1 - - [17/Apr/2023:14:33:20 +0000] "POST /score HTTP/1.0" 200 60 "-" "-"
```

Figure 8.6: Successful Prediction and post-processing HTTP request

The prediction method performance is evaluated by ensuring the POST request's response time satisfies the requirements of the application. To assess logs, developers compare the time required to make one or more requests using a form of load testing.

Figure 8.7: 8 Cores Successfully Finished

- NUnit Testing

NUnit is a widely used unit testing framework in the .NET ecosystem. It allows developers to write automated tests that can be executed repeatedly, ensuring consistent and reliable results. This saves time and effort compared to manual testing. Another advantage is verifying the behavior of isolated units of code, functions or methods, which allows focused testing and helps identify and fix issues at an early stage.

Initially, when creating the NUnit Test project Visual Studio provides a default test which passes 100%. After creating two test cases, one of them for the history page to assess the prediction criticality and another one for the prediction itself coming from Azure. In figure 8.8, GetPredictionsAsync has failed the test for the reason that it is impossible to “predict" future cases for any country at any time. The only testing that can be done for this method is to test if it gives positive results or not, which passes the test, because new covid cases can only be 0 or more.

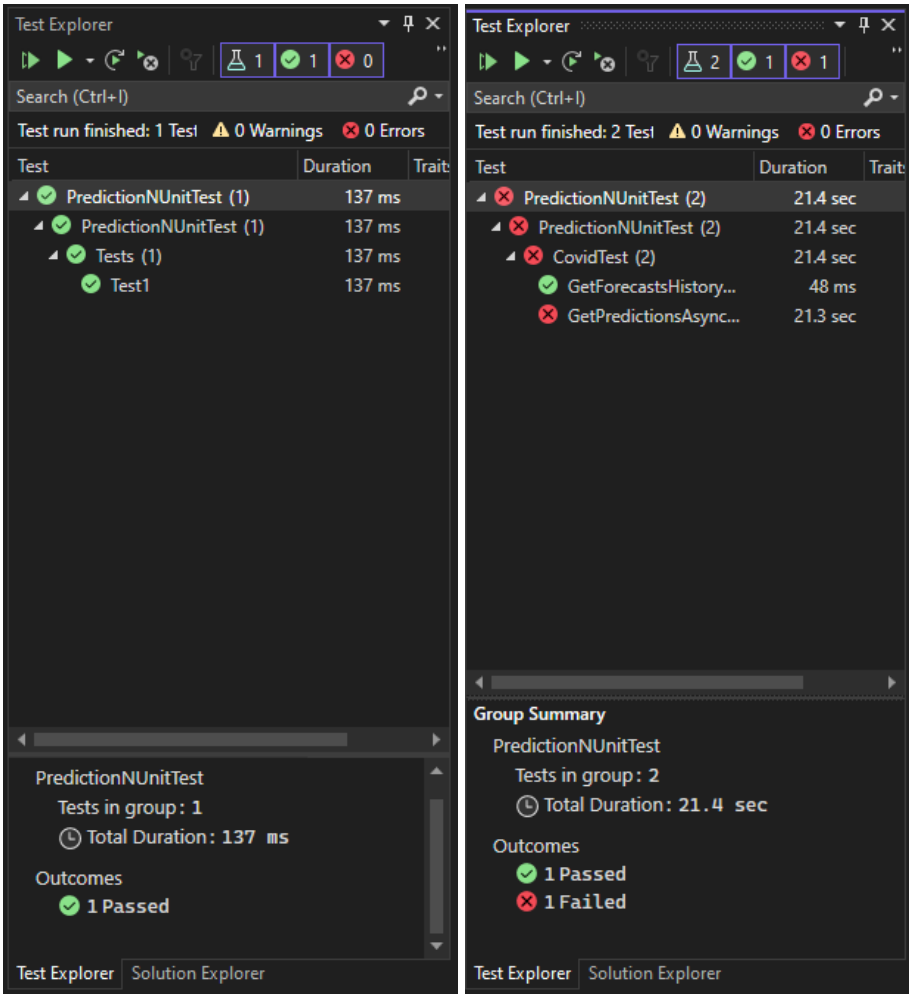

Figure 8.8: NUnit Testing Methods

8.4 Accuracy & Maintenance

This section will go into detail about how a month-long COVID prediction model will be tested and evaluated for the country of Greece in April 2023. Predictions will be examined how close they came to the confirmed ones and graphically display the results to compare the real and predicted case counts.

The anticipated case counts were compared with the actual confirmed cases published by authoritative sources, including the Greek Ministry of Health and the World Health Organization (WHO), in order to evaluate the model's accuracy. By taking both the daily and cumulative case counts into account, a thorough analysis was completed.

It was found that the model's performance fluctuated during the month when comparing the predicted and actual confirmed COVID cases. The model demonstrated good accuracy in the first few days of April, with the estimated and actual cases closely aligned. But over the

course of the month, it became clear that the predictions and the actual cases did not get very close results even though they lined up correctly.

As time goes by, the ML evaluation will be repeated to detect drifting and take any necessary actions such as data cleaning, transformations or any other model modifications. The reason behind this is that ML models tend to “drift” after some time by losing their accuracy.

Graphical Representation:

A graph was created to graphically represent the comparison study, showing the trend of actual confirmed cases next to expected cases. X-axis shows the dates in April 2023, while the y-axis represents the number of covid cases. Two lines are displayed on the graph: one shows the actual confirmed cases, and the other shows the expected case numbers.

The level of alignment or segregation between the predicted and actual examples is shown on the graph demonstrating the model's performance. It offers a precise illustration of any differences, trends, or patterns found during the testing period.

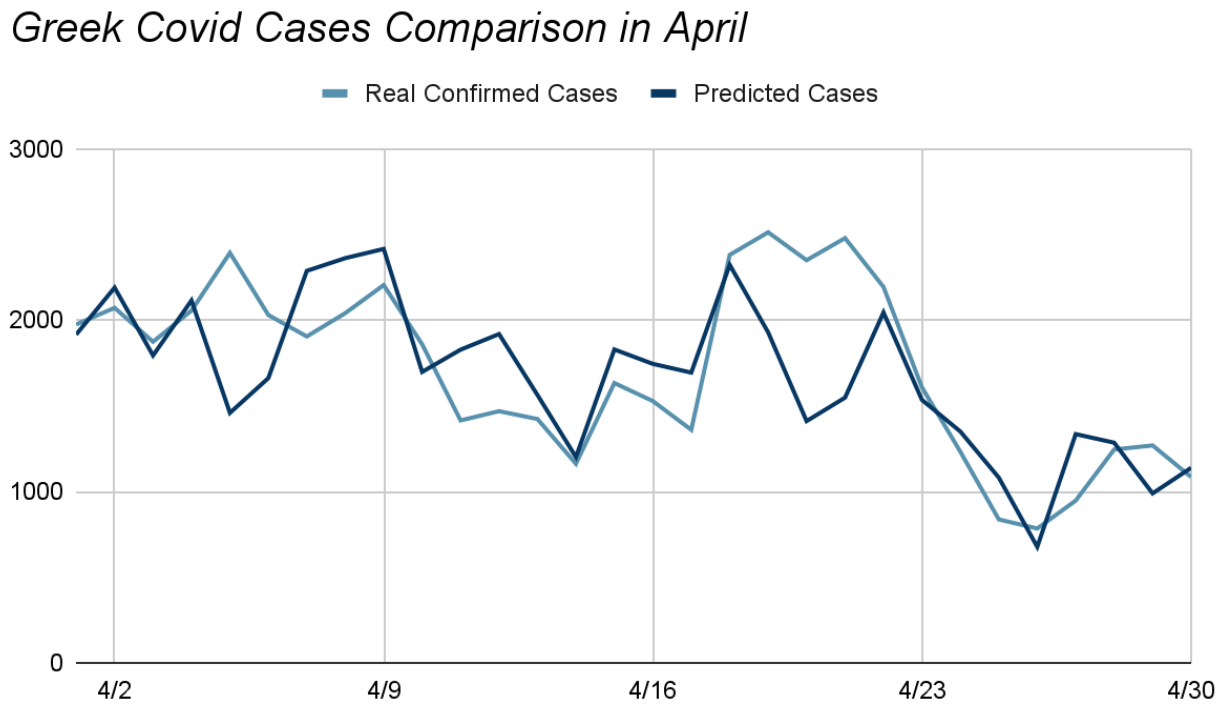

Figure / Graph 8.9: April Comparison of Real - Predicted Cases Graph

Chapter 9

9. Evaluation

Despite not being a part of the development process, the evaluation section will act as an analysis of the project deliverables, focusing on the various problems that potentially affected the completion of the system objectives. Moreover, the problems encountered and future improvements are stated in the next subsections to note the complexity or improve the system's operation in the near future.

9.1 Evaluation of the system objectives

The evaluation of the system objectives is covered in this section. The objectives served as the foundation for developing the system and were first outlined in section 1.3 of the project's introduction. The platform was designed with an emphasis on these objectives.

- ☑ Create and train a machine learning model to forecast the likelihood of COVID-19 cases in all European Countries.

The initial aim was to develop and train a machine learning model that could predict the probability of COVID-19 cases throughout all of Europe. For this, it was necessary to collect pertinent data, transform it, and select the appropriate machine learning algorithm. The model's goal was to estimate the likelihood of new cases based on multiple variables including country location and population density. It was trained using historical data on COVID-19 cases.

- ☑ Evaluate the performance of the machine learning model using various metrics and identify areas for improvement.

The second objective was to assess the machine learning model's performance using several metrics. In order to evaluate the model's efficacy, this phase involves assessing its accuracy, average absolute error difference and other performance metrics. Areas were pinpointed where the model performed well and where needed work by examining these metrics. This review stage was crucial for making sure the model's dependability and optimizing its parameters for more accurate forecasts.

- ☑ Deploy the machine learning model as a web service in Azure ML, allowing users to input new data and receive real-time predictions on COVID-19 cases in a newly created web application.

The final objective was to use Azure ML to deploy the machine learning model as a web service. Through a newly developed online application, users were able to enter data, such as the most current COVID-19 numbers and get real-time forecasts on the number of cases. A

user-friendly interface was offered for obtaining and using the predictions from the machine learning model by deploying Azure ML. This goal was to make the system usable and accessible for governments or health organizations who seek to predict COVID-19 cases in European nations.

The section ends by noting that all of these goals were achieved and that the overall outcomes were excellent. The machine learning model was effectively trained and developed, and the performance of the model was assessed using the right metrics. Real-time predictions may be accessed by users through a web application thanks to the implementation on Azure ML. The accomplishment of these goals demonstrated that the initiative met its aims and generated a beneficial result.

9.2 Problems Encountered

Numerous difficulties were encountered when the COVID predictions project was developed. The project's development was impacted by many difficulties, but finally attempts were made to get through them and provide a high-quality product that was adequate as well as the necessary process documentation. The following are some major difficulties encountered throughout the project:

Azure Accounts
Managing Azure accounts during the project presented challenges. To guarantee an effortless deployment and integration of the machine learning model as a web service, it needed setting up and configuring the accounts. There may have been problems with account setup, access restrictions, or subscription administration that needed more time and work to fix. Not only that but the insufficient funds to run such an expensive project resulted in signing up from different accounts to receive a “free subscription” even though it had already been worked on other accounts and then re-set up all the data, models, endpoints etc.

Not good metrics
Metrics evaluation of the machine learning model's performance caused difficulties. It was possible that the chosen measures did not adequately reflect how well the model predicted COVID-19 instances. But after a more thorough examination, this necessitated reevaluating the metrics by going through everything step by step. Starting with the dataset that was supposed to be excellent, provided by ‘Our World In Data, Org’, had some inconsistencies.

Negative values
Dealing with negative values derived in the scored model complicated the model training process. Negative values resulted in inaccurate predictions or have an impact on the model's performance as a whole. The data had to be handled and preprocessed appropriately, and negative values had to be either corrected or treated in a way that did not impair the model's predictions.

Wrong scored labels because of total cases trained not new cases

The target variable used to train the model was new cases rather than total cases. This resulted in misleading forecasts and evaluations of the probability of new COVID-19 cases at the start of the project. To guarantee the model was trained using the proper labels and increase prediction accuracy, modifications had to be performed. So, after realizing that all the values obtained by training the model were extremely high, it was understood that the wrong data was being trained.

Python Script making all cases 0 or not working at all

There were issues with the Python script used for data transformation. The script was not working properly initially, giving erroneous results by transforming the column into 0's making it theoretically the perfect model, or even not running at all. To find and fix the script-related problem, debugging and troubleshooting were required. To understand what the problem was, the python script was tested locally explained at section 6.4.1.2.

Real-time endpoint fail, by putting all inputs to predict cases

Failures occurred during the deployment of the machine learning model as a real-time web service. They were the consequence of problems with endpoint configuration, input data processing, or online application interaction. To guarantee that the real-time predictions could be accessed, troubleshooting was needed to determine the primary reason of the failure and put the essential solutions in place such as deploying the model as real-time inference rather than batch inference, exclude new_cases cleaning data etc.

Nu-Get Packages in Blazor and failed imports

Working with NuGet packages led to compatibility problems or import errors, especially when using the Blazor framework. The creation of the web application and its connection with the machine learning model was impacted by this and many packages were avoided to keep this project simple and working perfectly. One of these error prone packages was BlazorStrap. To guarantee a smooth integration, efforts have to be made to address package-related issues, update dependencies, or discover alternate solutions.

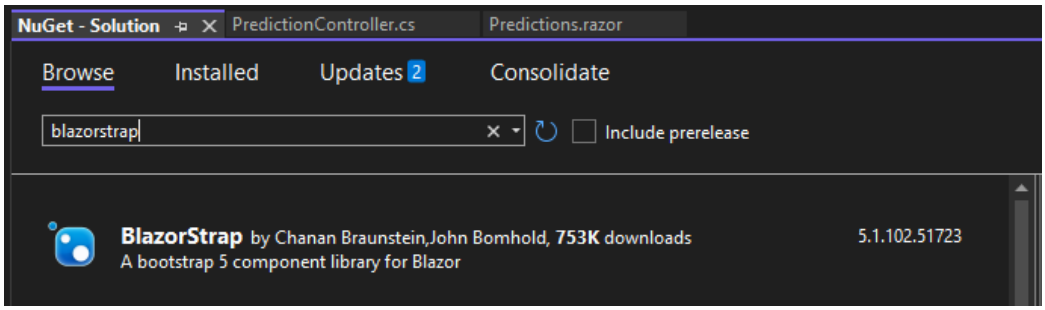

Figure 9.1: BlazorStrap Package

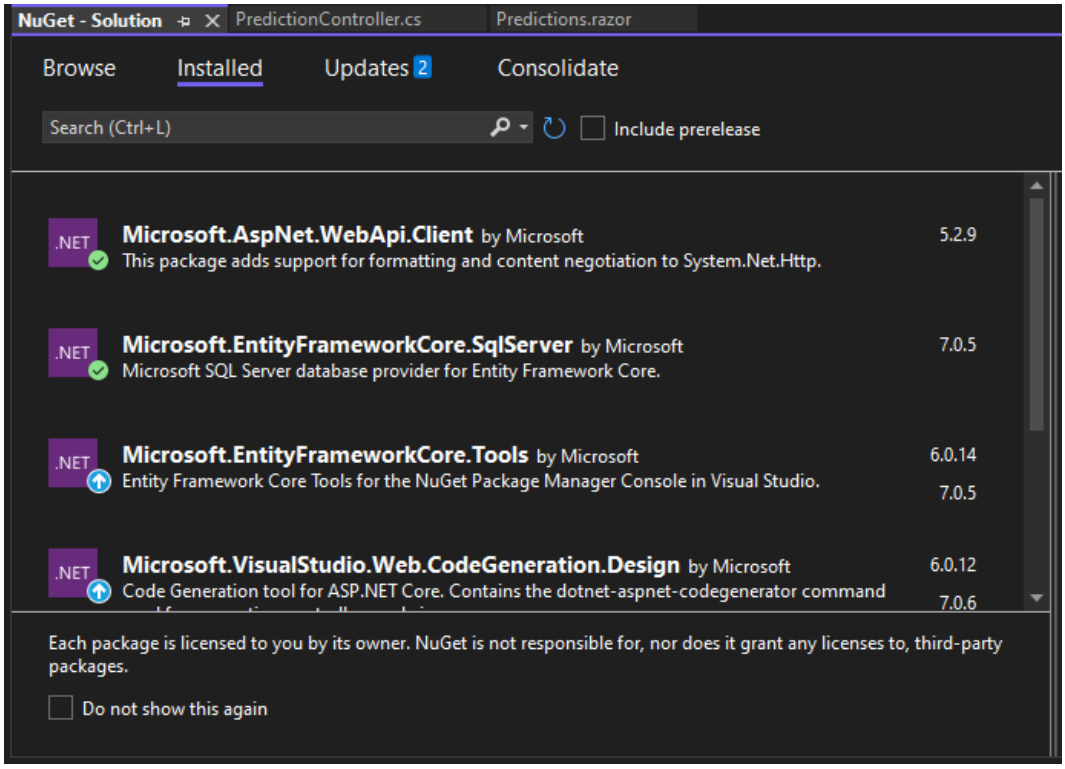

Figure 9.2: Installed Nu-Get Packages

Key auth access denied

Figure 8.5 shows the Azure environment's access prohibition for key authentication that prevented the machine learning model from being accessed. This was a mistake which was found out when the free subscription of the Azure account had either expired or the 200$ free credit was totally used. For flawless interaction with the deployed web service, it was essential to handle the authentication problems, making sure the access rights and authentication procedures were in place.

Http request fail

HTTP requests to get data from the web service failed countless times. These errors were the result of network difficulties, bad request setups, or even compatibility concerns. The online application's flawless operation and the retrieval of real-time forecasts depended on locating and fixing these problems. One relief was having the C# code provided by the Endpoint Consume which required only the manual edit of the API key.

Incorrect input schema

The input schema of the machine learning model presented one of the difficulties shown in figure 8.2 back in the Azure Testing section. The model's anticipated input schema was not compatible with the input data's actual structure or data types. The difference in the input schema resulted in errors and unexpected behavior when making predictions. Everytime the

schema displayed failed execution due to the fact that the inputs were passed immediately without creating a serialized payload to be accepted by Azure's endpoint input schema.

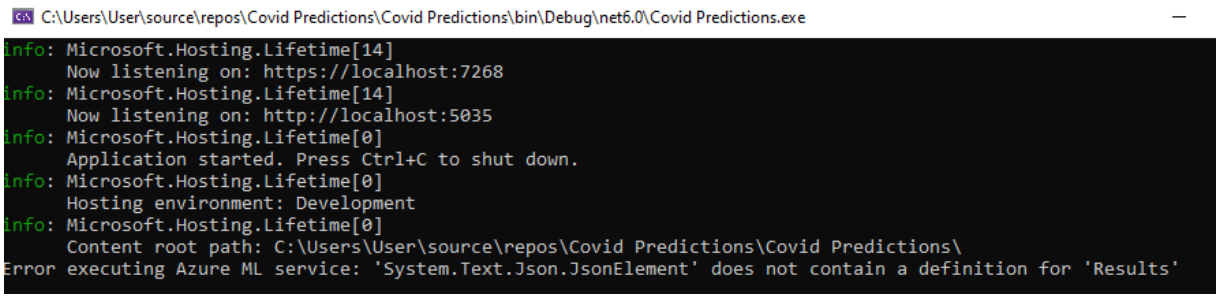

Figure 9.3: Error executing 'System.Text.Json.JsonElement'

Difficulty deserializing JSON output to get the results

The project also ran into trouble trying to obtain the prediction results from the JSON output by deserializing it. The deployed web service's server sent a response in JSON format after receiving a successful request. However, it was difficult to extract the necessary data from the JSON answer and transform it into a prediction. The efficient extraction and usage of the prediction findings were hampered by the challenge of deserializing the JSON output. To effectively get the relevant information, it was necessary to carefully analyze the JSON structure, comprehend the hierarchical data parts from 'results' to 'WebServiceOutput0' to 'scoredLabels', and use the proper parsing.

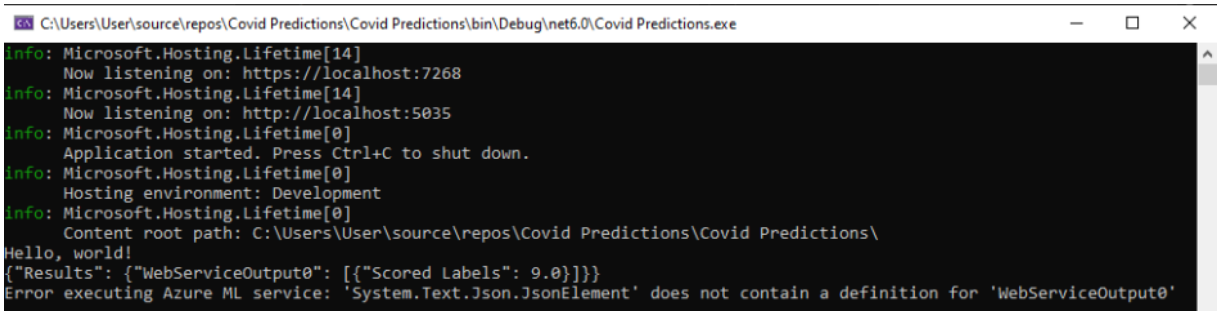

Figure 9.4: 'WebServiceOutput0' Deserialization Failure

Problems with DbContext to save predictions

The SQL database was created in Microsoft SQL Server Management Studio and the table with all the columns. The Nu-Get packages from Microsoft.EntityFrameworkCore.SqlServer created an issue within the DbContext class. The defined DbSet properties for the table and each column, which represent the corresponding entities in the application, could not get configured to the database and interact with it.

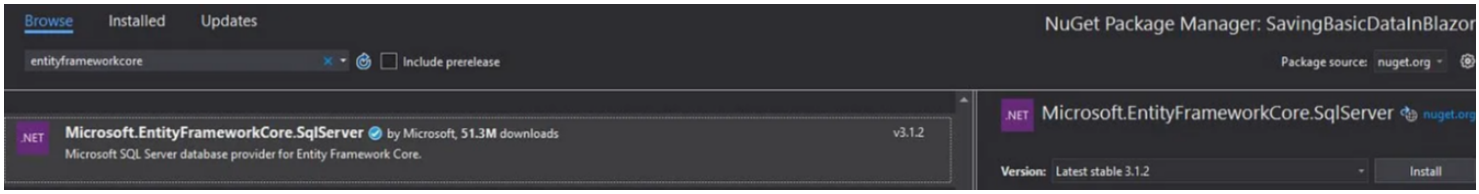

Figure 9.5: Nu-Get Package by Microsoft for SqlServer

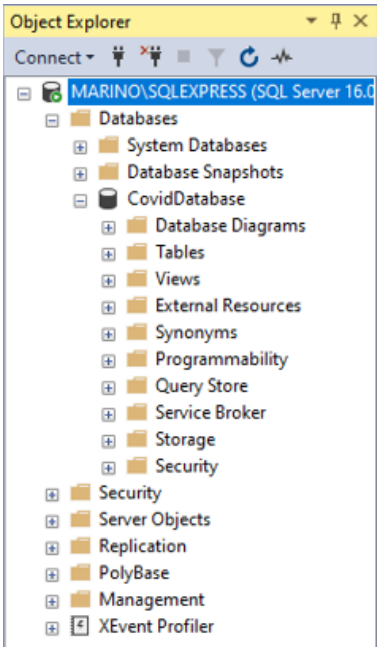

Figure 9.6: Microsoft SQL Server Management Studio

9.3 Future Improvements

Although the project accomplished its objectives and produced excellent outcomes, there are still a number of areas where improvements for both the ML model and the web application. These modifications are designed to improve the whole system's operation and correct any flaws. Future improvements of the project may take into account the following recommendations:

- Enhancing the Machine Learning Model:

Explore and incorporate additional features: To increase the predicted accuracy of the machine learning model, adding more features and variables are a possibility. Insights on COVID-19 cases, for instance, may be gained from variables like vaccination rates, public health measurements and socioeconomic indices.

Update the model frequently: Trends and patterns in COVID-19 vary over time, therefore it's critical to constantly update the model with the most recent data to make sure its forecasts are accurate.

Additional predictions not related to new_cases: Beyond new examples, broaden the prediction ability to incorporate more COVID-19-related indicators. Forecasting statistics like hospitalizations or death rates may be included in this. Users may better grasp the entire impact of COVID-19 and make informed plans thanks to the extensive set of projections that are provided.

- User Interface and Experience:

Weekly/monthly case predictions added to the user interface to improve it: Users may choose to view weekly or monthly forecasts of COVID-19 cases. Users may gain a wider perspective on the trends and patterns of the illness as a result.

Visualization tools, graphs, statistics: Interactive visualizations that show the anticipated instances over time, such as statistics, line charts or heatmaps, make it simpler for users to understand and evaluate the data. More examples are to include the option to create custom prediction parameters, compare regions, or visualize historical data.

Designing a better user interface: The user interface should constantly be enhanced to make it more intuitive to use and visually appealing by taking into account user feedback to pinpoint areas that require improvement and deal with any usability problems.

- Automated weekly predictions without user interaction:

Implement an automated system to produce weekly predictions without requiring human input: This can entail planning frequent updates and retraining models using the most recent data. Users can rely on the system to give up-to-date forecasts without manual involvement thanks to the automation of the prediction process, ensuring they have access to the most recent information for decision-making.

- Deployment and Infrastructure:

Cloud provider optimization: To maintain effective resource use and cost-effectiveness, the Azure ML implementation has to be continuously analyzed and improved.

Chapter 10

10. Conclusion and Final Words

To conclude, the final-year project offered the chance to create a system developed through a machine learning model coupled with a web application to interact with the model and obtain accurate predictions which required so many new technologies, tools and research to complete it.

This project was kept on track by the supervisor's constant guidance, which made the outcome to be considered very successful. The journey of the past nine months has been filled with challenges, obstacles, and moments of self-doubt, the requirements appeared to be quite impossible to be finished but with persistence, willingness, grit, consistency and supervisor’s motivation and feedback this project’s outcome was the the complete opposite of what I thought it would be. Continuous learning, modeling and experimenting brought out the best version of myself.

This system’s aim is achieved and now provides accurate COVID-19 predictions while offering transparency and simplicity to users. By combining machine learning technology, frontend and backend frameworks, scripts, and other Azure services, the system delivers a reliable and user-friendly solution for predicting COVID-19 cases, catering to the needs of governments, health sector managers and other users seeking valuable insights for decision-making.

References

[1] "Impact of COVID-19 on people's livelihoods, their health and our food systems," *World Health Organization*, 13 Oct. 2020. [Online]. Available: https://www.who.int/news/item/13-10-2020-impact-of-covid-19-on-people%27s-livelihoods-their-health-and-our-food-systems [Accessed: 29-May-2023].

[2] A. V. Raveendran, R. Jayadevan, and S. Sashidharan, "Long COVID: An overview," *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, vol. 15, no. 3, pp. 869–875, Apr 2021. doi:10.1016/j.dsx.2021.04.007

[3] Y. Shi et al., "An overview of COVID-19," *Journal of Zhejiang University-Science B*, vol. 21, no. 5, pp. 343–360, May 2020, doi: https://doi.org/10.1631/jzus.b2000083.

[4] T. J. Witek, "How the Global COVID-19 Pandemic Brought Drug and Vaccine Development into the Public Mainstream," *Pharmaceutical Medicine*, Sep. 2021, doi: https://doi.org/10.1007/s40290-021-00402-y.

[5] A. S. Kwekha-Rashid, H. N. Abduljabbar, and B. Alhayani, "Coronavirus disease (COVID-19) cases analysis using machine-learning applications," *Applied Nanoscience*, vol. 13, no. 3, pp. 2013–2025, 2021. doi:10.1007/s13204-021-01868-7

[6] B. Nithya and V. Ilango, "Predictive analytics in healthcare using machine learning tools and techniques," *International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 492-499, 2017, doi: 10.1109/ICCONS.2017.8250771.

[7] S. Dananjayan and G. M. Raj, "Artificial Intelligence during a pandemic: The COVID‐19 example," *The International Journal of Health Planning and Management*, vol. 35, no. 5, May 2020, doi: https://doi.org/10.1002/hpm.2987.

[8] B. J. Copeland, "Artificial intelligence | Definition, Examples, and Applications," *Encyclopedia Britannica,* Sep. 11, 2022. [Online]. Available: https://www.britannica.com/technology/artificial-intelligence [Accessed: 29-May-2023].

[9] E. Burns, N. Laskowski, and L. Tucci, "What is Artificial Intelligence (AI)? - AI definition and how it works," *TechTarget,* Feb. 2022. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence [Accessed: 29-May-2023].

[10] "What is Artificial Intelligence (AI) ?," *IBM*, 2023, [Online]. Available: https://www.ibm.com/topics/artificial-intelligence [Accessed: 29-May-2023].

[11] A. Schroer, "Artificial Intelligence.," *BuiltIn*, Mar. 03, 2023. [Online]. Available: https://builtin.com/artificial-intelligence [Accessed: 31-May-2023].

[12] J. Frankenfield, “Artificial Intelligence: What it is and how it is used,” *Investopedia*, Jul. 06, 2022. [Online]. Available: https://www.investopedia.com/terms/a/artificial-intelligence-ai.asp [Accessed: 31-May-2023].

[13] “What is machine learning?,” IBM, 2023, [Online]. Available: https://www.ibm.com/topics/machine-learning [Accessed: 31-May-2023].

[14] “Machine learning: What it is, tutorial, definition, types,” *Javatpoint*, [Online]. Available: https://www.javatpoint.com/machine-learning [Accessed: 31-May-2023].

[15] E. Burns, “What is machine learning and why is it important?,” *SearchEnterpriseAI*, Mar. 2021. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML [Accessed: 31-May-2023].

[16] S. G. Paul *et al.*, “Combating covid-19 using machine learning and Deep Learning: Applications, challenges, and future perspectives,” *Array*, vol. 17, p. 100271, Mar 2023. doi:10.1016/j.array.2022.100271

[17] S. H. Bareen et al. “Role of Machine Learning Techniques to Tackle the COVID-19 Crisis: Systematic Review.” *JMIR medical informatics,* vol. 9, 11 Jan 2021, doi:10.2196/23811

[18] M. Kumar, S. Atalla, N. Almuraqab, and I. A. Moonesar, “Detection of covid-19 using Deep Learning techniques and cost effectiveness evaluation: A survey,” *Frontiers in Artificial Intelligence*, vol. 5, May 2022. doi:10.3389/frai.2022.912022

[19] K. Moulaei, M. Shanbehzadeh, Z. Mohammadi-Taghiabad, and H. Kazemi-Arpanahi, “Comparing machine learning algorithms for predicting COVID-19 mortality,” *BMC Medical Informatics and Decision Making*, vol. 22, no. 1, Jan. 2022. doi:10.1186/s12911-021-01742-0

[20] D. N. Vinod and S. R. Prabaharan, “Covid-19-the role of Artificial Intelligence, Machine Learning, and Deep Learning: A newfangled,” *Archives of Computational Methods in Engineering*, vol. 30, no. 4, pp. 2667–2682, Jan 2023. doi:10.1007/s11831-023-09882-4

[21] Y. Gao, G.-Y. Cai, W. Fang, H.-Y. Li, and S.-Y. Wang et al., “Machine Learning Based Early Warning System enables accurate mortality risk prediction for COVID-19,” *Nature Communications*, vol. 11, no. 1, pp. 5033–5082, Oct 2020. doi:10.1038/s41467-020-18684-2

[22] “What is machine learning?: How it works, tutorials, and examples,” *Mathworks.com*, 2019. [Online]. Available: https://www.mathworks.com/discovery/machine-learning.html [Accessed: 31-May-2023].

[23] “What is supervised learning?,” *IBM*, [Online]. Available: https://www.ibm.com/topics/supervised-learning [Accessed: 31-May-2023].

[24] “Supervised machine learning - Javatpoint,” *Javatpoint*, 2022. [Online]. Available: https://www.javatpoint.com/supervised-machine-learning [Accessed: 31-May-2023].

[25] R. Raj, “Email spam and non-spam filtering using machine learning,” *Enjoyalgorithms*, [Online]. Available: https://www.enjoyalgorithms.com/blog/email-spam-and-non-spam-filtering-using-machine-learning [Accessed: 31-May-2023].

[26] S. Bansal, “Supervised and Unsupervised learning - GeeksforGeeks,” *GeeksforGeeks*, Apr. 17, 2019. [Online]. Available: https://www.geeksforgeeks.org/supervised-unsupervised-learning [Accessed: 31-May-2023].

[27] “Ecommerce Machine Learning: Business Benefits + Use Cases,” *BigCommerce*. [Online]. Available: https://www.bigcommerce.com/articles/ecommerce/machine-learning/ [Accessed: 31-May-2023].

[28] D. Johnson, “Unsupervised Machine Learning: What is, Algorithms, Example,” *Guru99.com*, Sep. 21, 2019. [Online]. Available: https://www.guru99.com/unsupervised-machine-learning.html [Accessed: 31-May-2023].

[29] “Unsupervised Machine learning - Javatpoint,” *Javatpoint*. [Online]. Available: https://www.javatpoint.com/unsupervised-machine-learning [Accessed: 31-May-2023].

[30] O. G. Yalçın, “4 Machine Learning Approaches that Every Data Scientist Should Know,” *Medium*, Feb. 02, 2021. [Online]. Available: https://towardsdatascience.com/4-machine-learning-approaches-that-every-data-scientist-should-know-e3a9350ec0b9 [Accessed: 31-May-2023].

[31] “Semi-Supervised Learning, Explained with Examples,” *AltexSoft*, Mar. 18, 2022. [Online]. Available: https://www.altexsoft.com/blog/semi-supervised-learning/ [Accessed: 31-May-2023].

[32] S. Bhatt, “Reinforcement Learning 101,” *Medium*, Apr. 19, 2019. [Online]. Available: https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292 [Accessed: 31-May-2023].

[33] “Introduction to Dimensionality Reduction - GeeksforGeeks,” *GeeksforGeeks*, Jun. 2017. [Online]. Available: https://www.geeksforgeeks.org/dimensionality-reduction/ [Accessed: 31-May-2023].

[34] T. Shin, “All Machine Learning Models Explained in 6 Minutes,” *Medium*, Oct. 17, 2020. [Online]. Available: https://towardsdatascience.com/all-machine-learning-models-explained-in-6-minutes-9fe30ff6776a [Accessed: 31-May-2023].

[35] D. Kalita, “An Overview and Applications of Artificial Neural Networks,” Analytics Vidhya, Mar. 30, 2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2022/03/an-overview-and-applications-of-artificial-neural-networks-ann/ [Accessed: 31-May-2023].

[36] “1.10. Decision Trees — scikit-learn 0.22 documentation,” *Scikit-learn.org*, 2009. [Online]. Available: https://scikit-learn.org/stable/modules/tree.html [Accessed: 31-May-2023].

[37] A. Saini, “Decision Tree Algorithm - A Complete Guide,” *Analytics Vidhya*, Aug. 29, 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/ [Accessed: 31-May-2023].

[38] “1.4. Support Vector Machines — scikit-learn 0.20.3 documentation,” *Scikit-learn.org*, 2018. [Online]. Available: https://scikit-learn.org/stable/modules/svm.html [Accessed: 31-May-2023].

[39] “Support Vector Machine (SVM) Algorithm - Javatpoint,” *Javatpoint,* [Online]. Available: https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm [Accessed: 31-May-2023].

[40] R. Gandhi, “Support Vector Machine — Introduction to Machine Learning Algorithms,” Towards Data Science, Jun. 07, 2018. [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 [Accessed: 31-May-2023].

[41] “What is Regression Analysis?,” TIBCO Software. [Online]. Available: https://www.tibco.com/reference-center/what-is-regression-analysis [Accessed: 31-May-2023].

[42] S. Taylor, “Regression Analysis,” Corporate Finance Institute, Nov. 24, 2022. [Online]. Available: https://corporatefinanceinstitute.com/resources/data-science/regression-analysis/ [Accessed: 31-May-2023].

[43] “Chapter 2 Multiple Regression I (Part 1) 1 Regression several predictor variables.” [Online]. Available: https://web.njit.edu/~wguo/Math644_2012/Math644_Chapter%202_part1.pdf [Accessed: 31-May-2023].

[44] X.S. Yang, “Mathematical foundations,” *Introduction to Algorithms for Data Mining and Machine Learning*, pp. 19–43, 2019, doi: https://doi.org/10.1016/b978-0-12-817216-2.00009-0.

[45] J. Görtler, R. Kehlbeck, and O. Deussen, “A Visual Exploration of Gaussian Processes,” *Distill*, vol. 4, no. 4, p. 17, Apr. 2019, doi: https://doi.org/10.23915/distill.00017.

[46] “Gaussian Process,” *www.cs.cornell.edu*. [Online]. Available: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote15.html [Accessed: 31-May-2023].

[47] GeeksforGeeks, “Genetic Algorithms,” GeeksforGeeks, Jun. 29, 2017. [Online]. Available: https://www.geeksforgeeks.org/genetic-algorithms/ [Accessed: 31-May-2023].

[48] X.-S. Yang, “Genetic Algorithms,” *Nature-Inspired Optimization Algorithms*, pp. 91–100, 2021, doi: https://doi.org/10.1016/b978-0-12-821986-7.00013-5.

[49] “Machine Learning Algorithm Cheat Sheet - designer - Azure Machine Learning,” *learn.microsoft.com*, May 29, 2023. [Online]. Available:

https://learn.microsoft.com/en-us/azure/machine-learning/v1/algorithm-cheat-sheet [Accessed: 31-May-2023].

[50] "How to select a machine learning algorithm - Azure Machine Learning," *learn.microsoft.com*, May 29, 2023. [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/v1/how-to-select-algorithms [Accessed: 31-May-2023].

[51] "Microsoft Linear Regression Algorithm," *learn.microsoft.com*, Dec. 09, 2022. [Online]. Available: https://learn.microsoft.com/en-us/analysis-services/data-mining/microsoft-linear-regression-algorithm [Accessed: 31-May-2023].

[52] "Decision Forest Regression: Component Reference - Azure Machine Learning," *learn.microsoft.com*, Nov. 04, 2021. [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/decision-forest-regression [Accessed: 31-May-2023].

[53] "Random Forest Algorithm for Absolute Beginners in Data Science," *Analytics Vidhya*, Oct. 19, 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/10/an-introduction-to-random-forest-algorithm-for-beginners/ [Accessed: 31-May-2023].

[54] "Boosted Decision Tree Regression: Component Reference - Azure Machine Learning," *learn.microsoft.com*, Jul. 06, 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/boosted-decision-tree-regression [Accessed: 31-May-2023].

[55] "Fast Forest Quantile Regression: Module reference - Azure Machine Learning," *learn.microsoft.com*, Nov. 04, 2021. [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/fast-forest-quantile-regression [Accessed: 31-May-2023].

[56] "Neural Network Regression: Component Reference - Azure Machine Learning," *learn.microsoft.com*, Nov. 04, 2021. [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/neural-network-regression [Accessed: 31-May-2023].

[57] "Introduction To Bayesian Linear Regression | Simplilearn," *Simplilearn.com*, Oct. 31, 2022. [Online]. Available: https://www.simplilearn.com/tutorials/data-science-tutorial/bayesian-linear-regression [Accessed: 31-May-2023].

[58] "Poisson Regression: Component reference - Azure Machine Learning," *learn.microsoft.com*, Nov. 10, 2021. [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/poisson-regression [Accessed: 31-May-2023].

[59] C. BasuMallick, “AWS vs. Azure: Understanding the Key Differences,” *Spiceworks*, Apr. 28, 2022. [Online]. Available: https://www.spiceworks.com/tech/cloud/articles/aws-vs-azure/ [Accessed: 31-May-2023].

[60] S. Bigelow, “What is Microsoft Azure and How Does It Work?,” *SearchCloudComputing*, 2022. [Online]. Available: https://www.techtarget.com/searchcloudcomputing/definition/Windows-Azure [Accessed: 31-May-2023].

[61] “Azure Benefits and incentives | Microsoft Azure,” *azure.microsoft.com*. [Online]. Available: https://azure.microsoft.com/en-us/pricing/offers/#cloud [Accessed: 31-May-2023].

[62] M. Collier and R. Shahan, Microsoft Azure Essentials - Fundamentals of Azure. Microsoft Press, 2015.

[63] “What Is AWS? - Amazon Web Services,” *Amazon Web Services*, Inc., 2019. [Online]. Available: https://aws.amazon.com/what-is-aws/ [Accessed: 31-May-2023].

[64] “Build, Train, and Deploy Machine Learning Models | Amazon SageMaker,” *Amazon Web Services*, Inc., 2019. [Online]. Available: https://aws.amazon.com/sagemaker/ [Accessed: 31-May-2023].

[65] “Amazon Comprehend - Natural Language Processing (NLP) and Machine Learning (ML),” *Amazon Web Services*, Inc., 2019. [Online]. Available: https://aws.amazon.com/comprehend/ [Accessed: 31-May-2023].

[66] “Amazon Rekognition – Video and Image - AWS,” *Amazon Web Services*, Inc., 2017. [Online]. Available: https://aws.amazon.com/rekognition/ [Accessed: 31-May-2023].

[67] “Real-time personalization and recommendation | Amazon Personalize | AWS,” *Amazon Web Services*, Inc., 2019. [Online]. Available: https://aws.amazon.com/personalize/ [Accessed: 31-May-2023].

[68] “Amazon Polly,” *Amazon Web Services*, Inc., 2019. [Online]. Available: https://aws.amazon.com/polly/ [Accessed: 31-May-2023].

[69] “Amazon Transcribe – Speech to Text - AWS,” *Amazon Web Services*, Inc., 2019. [Online]. Available: https://aws.amazon.com/transcribe/ [Accessed: 31-May-2023].

[70] “What is Blazor | Blazor tutorial for beginners,” *Pragimtech*. [Online]. Available: https://www.pragimtech.com/blog/blazor/what-is-blazor/ [Accessed: 31-May-2023].

[71] “What is Blazor? – Blazor University.” *Blazor University,* [Online]. Available: https://blazor-university.com/overview/what-is-blazor/ [Accessed: 31-May-2023].

[72] “Blazor University - What is WebAssembly?,” *Blazor University,* [Online]. Available: https://blazor-university.com/overview/what-is-webassembly/ [Accessed: 31-May-2023].

[73] “What’s behind the hype about Blazor?,” *Stack Overflow Blog*, Feb. 26, 2020. [Online]. Available: https://stackoverflow.blog/2020/02/26/whats-behind-the-hype-about-blazor/ [Accessed: 31-May-2023].

[74] “What is ASP.NET? | The Open Source Web Framework,” *umbraco.com*. [Online]. Available: https://umbraco.com/knowledge-base/aspnet/ [Accessed: 31-May-2023].

[75] “ASP.NET - Introduction,” *Tutorialspoint.com*, 2019. [Online]. Available: https://www.tutorialspoint.com/asp.net/asp.net_introduction.htm [Accessed: 31-May-2023].

[76] R. S. Pressman, *Software engineering : a practitioner’s approach*. Boston: Mcgraw-Hill, 2009.

[77] Ruth Malan et al. *Functional requirements and use cases.* Bredemeyer Consulting, 2001.

[78] R. Malan, H.-P. Company, and D. Bredemeyer, “Functional Requirements and Use Cases Functional Requirements.” [Online]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=aceaa41855c38aebe7c823e60e94b39506a92b99 [Accessed: 31-May-2023].

[79] L. Chung, *Non-functional requirements in software engineering*. New York: Springer, 2012.

[80] Walt Scacchi. Process models in software engineering. Encyclopedia of software engineering, 2002.

[81] K. Sharma, “Top 12 Software Development Methodologies & its Advantages / Disadvantages | TatvaSoft,” *Tatvasoft.com*, Dec. 25, 2020. [Online]. Available: https://www.tatvasoft.com/blog/top-12-software-development-methodologies-and-its-advantages-disadvantages/ [Accessed: 31-May-2023].

[82] K. Petersen, C. Wohlin, and D. Baca, “The Waterfall Model in Large-Scale Development”, *Blekinge Institute of Technology*, pp. 386–400, 2009. doi: https://doi.org/10.1007/978-3-642-02152-7_29.

[83] Kramer, Mitch, “Best Practices in Systems Development Lifecycle: An Analysis Based on the Waterfall Model,” *Review of Business & Finance Studies*, v. 9, pp. 77-84, 2018. SSRN: https://ssrn.com/abstract=3131958

[84] “SDLC Iterative Model,” *Tutorialspoint*, 2019. [Online]. Available: https://www.tutorialspoint.com/sdlc/sdlc_iterative_model.htm [Accessed: 31-May-2023].

[85] W. Pedrycz and K. -C. Kwak, "The Development of Incremental Models," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 3, pp. 507-518, June 2007, doi: 10.1109/TFUZZ.2006.889967.

[86] R. Matinnejad, "Agile Model Driven Development: An Intelligent Compromise," *2011 Ninth International Conference on Software Engineering Research, Management and Applications,* Baltimore, MD, USA, pp. 197-202, 2011. doi: 10.1109/SERA.2011.17.

[87] J. Erickson, K. Lyytinen, and K. Siau, “Agile Modeling, Agile Software Development, and Extreme Programming,” *Journal of Database Management*, vol. 16, no. 4, pp. 88–100, Oct 2005, doi: https://doi.org/10.4018/jdm.2005100105.

[88] B. W. Boehm, "A spiral model of software development and enhancement," vol. 21, no. 5, pp. 61-72, May 1988, doi: 10.1109/2.59.

[89] M. Power, "The risk management of everything", *Journal of Risk Finance*, vol. 5, no. 3, pp. 58-65, 2011. https://doi.org/10.1108/eb023001

[90] "Azure Machine Learning - ML as a Service," *Microsoft Azure*, [Online]. Available: https://azure.microsoft.com/en-au/products/machine-learning [Accessed: 31-May-2023].

[91] A. Khemiri, "Microsoft Azure Machine Learning," *Medium*, Jul. 12, 2019. [Online]. Available: https://medium.com/@ahmedkhemiri24/microsoft-azure-machine-learning-d148478e867c [Accessed: 31-May-2023].

[92] H. Mao, "Azure Technical Documentation Contributor Guide," *GitHub*, May 16, 2023. [Online]. Available: https://github.com/Huachao/azure-content/blob/master/articles/machine-learning/machine-learning-what-is-ml-studio.md [Accessed: 31-May-2023].

[93] H. Marius, "A Brief Introduction to Azure Machine Learning Studio," *Medium*, Dec. 10, 2021. [Online]. Available: https://towardsdatascience.com/a-brief-introduction-to-azure-machine-learning-studio-9bbf41800a60 [Accessed: 31-May-2023].

[94] Microsoft Azure Machine Learning Studio, 2023, [Online]. Available: https://ml.azure.com/

[95] Microsoft Azure Portal, 2023, [Online]. Available: https://portal.azure.com/

[96] "Azure Machine Learning documentation," *learn.microsoft.com*, 2023, [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning [Accessed: 31-May-2023].

[97] "Create Your Azure Free Account Today | Microsoft Azure," *Microsoft Azure*. [Online]. Available: https://azure.microsoft.com/en-us/free/virtual-machines/ [Accessed: 31-May-2023].

[98] V. Alto, "Introduction to Azure Machine Learning," *Medium*, Nov. 28, 2022. [Online]. Available: https://medium.com/microsoftazure/introduction-to-azure-machine-learning-13143ccd19b2 [Accessed: 31-May-2023].

[99] "Introduction to Blob (object) storage - Azure Storage," *Microsoft Azure*, Aug. 11, 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction [Accessed: 31-May-2023].

[100] "Coronavirus (COVID-19) Cases - Statistics and Research," *Our World in Data,* 2022. [Online]. Available: https://ourworldindata.org/covid-cases [Accessed: 31-May-2023].

[101] "Azure Machine Learning designer | Microsoft Azure," *Microsoft Azure*, [Online]. Available: https://azure.microsoft.com/en-us/products/machine-learning/designer/ [Accessed: 31-May-2023].

[102] “What is the Azure Machine Learning designer? - Azure Machine Learning,” *Microsoft Azure*, [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/concept-designer [Accessed: 31-May-2023].

[103] “Evaluate Model: Component Reference - Azure Machine Learning,” *Microsoft Azure*, Nov. 10, 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/evaluate-model [Accessed: 31-May-2023].

[104] “Machine learning inference during deployment - Cloud Adoption Framework,” *Microsoft Azure*, Dec. 01, 2022. [Online]. Available: https://learn.microsoft.com/en-us/azure/cloud-adoption-framework/innovate/best-practices/ml-deployment-inference [Accessed: 31-May-2023].

[105] “Consuming Azure Machine Learning in ASP.NET Core,” *Telerik Blogs*, Jul. 10, 2017. [Online]. Available: https://www.telerik.com/blogs/consuming-azure-machine-learning-asp-net-core [Accessed: 31-May-2023].

Appendix

- PredictionInvoker.cs

```
// This code requires the Nuget package Microsoft.AspNet.WebApi.Client to be
installed.
// Install-Package Newtonsoft.Json
// .NET Framework 4.7.1 or greater must be used
namespace CallRequestResponseService
{
    class Program
    {
        static void Main(string[] args)
        {
            InvokeRequestResponseService().Wait();
        }

        static async Task InvokeRequestResponseService()
        {
            var handler = new HttpClientHandler()
            {
                ClientCertificateOptions = ClientCertificateOption.Manual,
                ServerCertificateCustomValidationCallback =
                        (httpRequestMessage, cert, cetChain, policyErrors) => {
return true; }
            };
            using (var client = new HttpClient(handler))
            {
                var requestBody = @"{
                  ""Inputs"": {
                    ""input1"": [
                      {
                        ""iso_code"": ""ALB"",
                        ""location"": ""Albania"",
                        ""date"": ""2020-03-05 00:00:00"",
                        ""population"": 2842318
                      },
                      {
                        ""iso_code"": ""ALB"",
                        ""location"": ""Albania"",
                        ""date"": ""2020-03-06 00:00:00"",
                        ""population"": 2842318
                      },
                      {
                        ""iso_code"": ""ALB"",
                        ""location"": ""Albania"",
```

```
                        ""date"": ""2020-03-07 00:00:00"",
                        ""population"": 2842318
                      }
                    ]
                  },
                  ""GlobalParameters"": {}
                }";

                const string apiKey = "s5GxetTYeKJeNDeR0H4lAf6yXac0ldlg";
                if (string.IsNullOrEmpty(apiKey)){
                    throw new Exception("A key should be provided to invoke the
endpoint");
                }
                client.DefaultRequestHeaders.Authorization = new
AuthenticationHeaderValue("Bearer", apiKey);
                client.BaseAddress = new
Uri("http://20.22.70.70:80/api/v1/service/covidendpoint02/score");

                var content = new StringContent(requestBody);
                content.Headers.ContentType = new
MediaTypeHeaderValue("application/json");

                HttpResponseMessage response = await client.PostAsync("",
content);

                if (response.IsSuccessStatusCode){
                    string result = await response.Content.ReadAsStringAsync();
                    Console.WriteLine("Result: {0}", result);
                }
                else {
                    Console.WriteLine(string.Format("The request failed with
status code: {0}", response.StatusCode));

Console.WriteLine(response.Headers.ToString());

                    string responseContent = await
response.Content.ReadAsStringAsync();
                    Console.WriteLine(responseContent);
                }
            }
        }
    }
}
```

Listing A.1: ‘CallRequestResponseService’ provided by Azure Endpoint Section

- Predictions.razor, GetPredictionsAsync() method

```
  private async Task GetPredictionsAsync()
    {
        try
        {
            // Set up the HTTP client
            const string apiKey = "s5GxetTYeKJeNDeR0H4lAf6yXac0ldlg";
            var client = new HttpClient();
            client.DefaultRequestHeaders.Authorization = new
AuthenticationHeaderValue("Bearer", apiKey);

            // Construct the input data payloads
            var inputObjects = new[]
            {
            new
                {
                    iso_code = IsoCode,
                    location = SelectedCountry,
                    date = SelectedDate.ToString("yyyy-MM-dd HH:mm:ss"),
                    population = Population
                }
            };

            var payload = JsonSerializer.Serialize(new
            {
                Inputs = new
                {
                    input1 = inputObjects
                },
                GlobalParameters = new { }
            });

            var content = new StringContent(payload, Encoding.UTF8,
"application/json");

            // Send the HTTP request and collect the prediction result
            var response = await
client.PostAsync("http://20.22.70.70:80/api/v1/service/covidendpoint02/score",
content);
            var result = await response.Content.ReadAsStringAsync();

            try
            {
                JsonDocument jsonDoc = JsonDocument.Parse(result);
                JsonElement root = jsonDoc.RootElement;
```

```
                JsonElement resultsElement;

                if (root.TryGetProperty("Results", out resultsElement))
                {
                    JsonElement webServiceOutput0Element;
                    if (resultsElement.TryGetProperty("WebServiceOutput0", out
webServiceOutput0Element) && webServiceOutput0Element.ValueKind ==
JsonValueKind.Array)
                    {
                        JsonElement scoredLabelsElement;
                        if (webServiceOutput0Element[0].TryGetProperty("Scored
Labels", out scoredLabelsElement))
                        {
                            predictionResult = scoredLabelsElement.GetRawText();
                            Console.WriteLine(predictionResult);
                        }
                    }
                }
            }
            catch (JsonException ex)
            {
                Console.WriteLine($"Error deserializing response:{ex.Message}");
            }

        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error executing Azure ML service:{ex.Message}");
        }
    }
```

Listing A.2: GetPredictionsAsync Method to send HTTP request and receive response