**Northumbria University, Newcastle**

**Faculty of Engineering and Environment**

**Department of Computer and Information Sciences**

**Machine Learning on Cloud**

**Submitted By**

|  |  |
| --- | --- |
| **Name:** | **Busra Ecem Sakar** |
| **ID:** | **22065255** |
| **Programme:** | **MSc Big Data and Data Science Technology** |
| **Semester:** | **2** |
| **Tutor:** | **Dr Bilal Hassan** |
| **Date:** | **18 May 2023** |
| **Word Count:** | **4.144** |

**Table of Contents**

[1. Cloud feasibility study 4](#_Toc135333539)

[2. Data analysis and opportunity identification 5](#_Toc135333540)

[3. Data pre-processing 8](#_Toc135333541)

[3.1. Data cleaning process 9](#_Toc135333542)

[3.1.1. Identify and handle missing values 9](#_Toc135333543)

[3.1.2. Identify and handle outliers 15](#_Toc135333544)

[3.2. Transforming Data 17](#_Toc135333545)

[3.2.1. Normalization 17](#_Toc135333546)

[3.2.2. Encoding categorical variables 18](#_Toc135333547)

[3.3. Reduction 18](#_Toc135333548)

[4. Model selection and training 19](#_Toc135333549)

[5. Model evaluation and visualization 23](#_Toc135333550)

[6. Model deployment 26](#_Toc135333551)

[7. Reference 41](#_Toc135333552)

**Table of Figures**

[Figure 1 Uploading Dataset, Checking data rows & columns, and Getting first 10 row 6](#_Toc135333580)

[Figure 2 Description of Data types 7](#_Toc135333581)

[Figure 3 Data pre-processing steps (Maharana, et al., 2022) 9](#_Toc135333582)

[Figure 4 Missing values in columns, percentage in totals 10](#_Toc135333583)

[Figure 5 Box plots and Density plots for columns that have null values 11](#_Toc135333584)

[Figure 6 Box plots and Density plots for columns that have null values 12](#_Toc135333585)

[Figure 7 Filling out null values 13](#_Toc135333586)

[Figure 8 Calculation of Statistics with and without null values 14](#_Toc135333587)

[Figure 9 Percentege of null values in total after removing null values 15](#_Toc135333588)

[Figure 10 Analysing duplicate values 15](#_Toc135333589)

[Figure 11 Boxplot for total fifa points 16](#_Toc135333590)

[Figure 12 Boxplot for team scores 16](#_Toc135333591)

[Figure 13 Boxplot for goalkepeer scores 17](#_Toc135333592)

[Figure 14 Normalization (min-max normalization) 17](#_Toc135333593)

[Figure 15 Encoding Categorical Variables 18](#_Toc135333594)

[Figure 16 Checking Encoding 18](#_Toc135333595)

[Figure 17 Correlation Matrix 19](#_Toc135333596)

[Figure 18 Dropping Draw values from dataset 20](#_Toc135333597)

[Figure 19 Preparing Dataset for modelling 20](#_Toc135333598)

[Figure 20 Split train, validation, and test sets 20](#_Toc135333599)

[Figure 21 Random Forest Modelling 21](#_Toc135333600)

[Figure 22 Decision Tree Modelling 22](#_Toc135333601)

[Figure 23 AdaBoost Modelling 23](#_Toc135333602)

[Figure 24 Confusion Matrix for three model 24](#_Toc135333603)

[Figure 25 ROC Curve for three model 25](#_Toc135333604)

[Figure 26 Defining hyperparameters 26](#_Toc135333605)

[Figure 27 Random Forest Classifier Pipeline 27](#_Toc135333606)

[Figure 28 Finalized schema of Pipeline 27](#_Toc135333607)

[Figure 29 Creating ranking dataset 28](#_Toc135333608)

[Figure 30 Creating function 28](#_Toc135333609)

[Figure 31 Generating results 29](#_Toc135333610)

[Figure 32 Simulation 29](#_Toc135333611)

[Figure 33 Sign in to the Amazon SageMaker console 31](#_Toc135333612)

[Figure 34 Changing AWS Region 32](#_Toc135333613)

[Figure 35 Creating a Notebook 33](#_Toc135333614)

[Figure 36 Permissions and encryption 33](#_Toc135333615)

[Figure 37 Create notebook instance 33](#_Toc135333616)

[Figure 38 Opening Jupiter 34](#_Toc135333617)

[Figure 39 Conda\_python3 34](#_Toc135333618)

[Figure 40 Importing libraries for ML 35](#_Toc135333619)

[Figure 41 Creating S3-Bucket 35](#_Toc135333620)

[Figure 42 Dowloading the SageMaker instance and load the data into a dataframe 36](#_Toc135333621)

[Figure 43 Shuffle and split the data into training data and test data 36](#_Toc135333622)

[Figure 44 Training the ML model 36](#_Toc135333623)

[Figure 45 Deployment the model 37](#_Toc135333624)

[Figure 46 Evaluate model performance 37](#_Toc135333625)

[Figure 47 Clean up 38](#_Toc135333626)

**Table of Tables**

[Table 1 Data type and descriptions 7](#_Toc135333627)

[Table 2 Definition of Variables 7](#_Toc135333628)

# **Cloud feasibility study**

The biggest international sporting event in the world is the FIFA World Cup. For every nation, hosting the FIFA World Cup remains a desirable objective. The dataset which will be analyzed for this study includes a different list of every international football match played since the 1990s. The original data, which contains a few extra features and missing values, is available on Kaggle. The objective of this research is to analyse and preprocess the data using the appropriate machine-learning methods. Later, it will suggest and create a machine-learning model to forecast the tournament winner.

Several academic disciplines have conducted research on the FIFA World Cup due to a singular event that profoundly affects society worldwide, and academics have been interested in studying it. The literature review includes a number of scholarly studies on FIFA. Groll et al. (2018) compared the accuracy of three modelling techniques—random forests, Poisson regression models, and ranking methods—for predicting football match results. The researchers compared different prediction methods using team ability parameters as estimates of current team strengths and discovered that adding team ability parameters from ranking methods as an additional covariate to the random forest model significantly increased its predictive power, making it the best model capable of outperforming bookmakers. They came to the conclusion that the random forest approach is the most promising contender for predicting the FIFA World Cup winner when team ability parameters from the ranking systems are added as a covariate.

For the deployment of models for a specific scenario, many cloud platforms provide machine-learning solutions. Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and IBM Cloud are the best cloud systems for machine learning.

Amazon Web Services (AWS) has different services for creating and implementing machine learning models. For the development, training, and application of large-scale models, AWS SageMaker is utilised. SageMaker is made for complete end-to-end machine learning workflows and supports both supervised and unsupervised learning techniques. AWS also offers tools for cleaning and preparing data, like Amazon S3 for data storage and AWS Glue (Dutta, 2019)

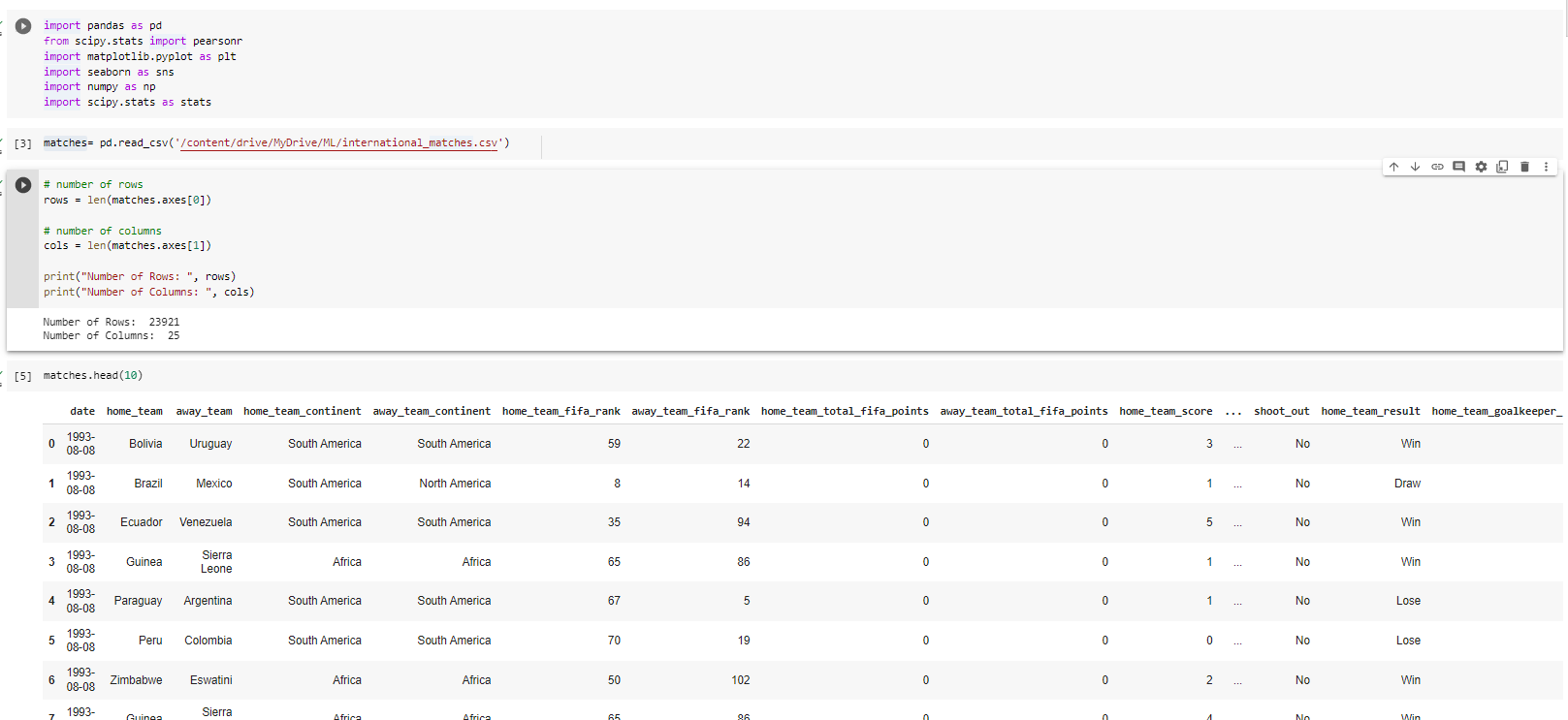
In addition to Google BigQuery for data analysis and storage, Google Cloud Platform (GCP) now offers Google Cloud Dataflow for data processing. Machine learning APIs for speech, picture, and natural language processing can also be used with GCP.

Microsoft Azure offers Azure Databricks for processing massive data and leverages Azure Machine Learning Studio to build models. For specific applications like fraud detection and predictive maintenance, Azure also offers pre-built machine learning models that can be customised (Gupta, et al., 2021).

Data scientists can rapidly and simply develop, train, and deploy machine learning models using IBM Watson Studio, one of the machine learning services that IBM Cloud offers. Big data can be stored in IBM Cloud Object Storage. The IBM Cloud is compatible with well-known machine learning frameworks including TensorFlow, Keras, and PyTorch (Gupta, et al., 2021).

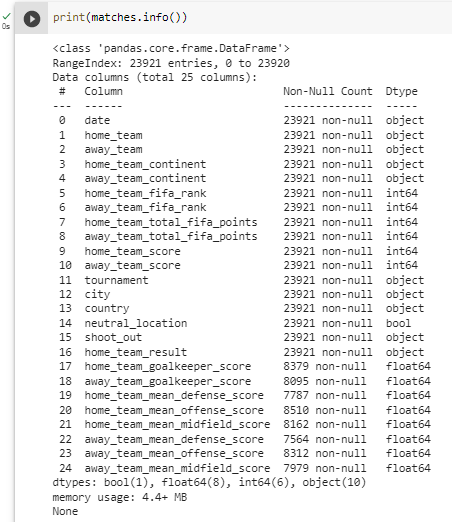
# **Data analysis and opportunity identification**

The Colaboratory (Colab) product from Google Research was used for analyses with the "Matches" dataset. The Colaboratory (Colab) product from Google Research is especially helpful for machine learning, data analysis, and education because it enables users to develop and run Python code through their browsers. (Google, n.d). The python code in figure 1 was executed after the "international\_matches.csv" dataset was uploaded to Google Drive in order to determine the total number of rows and columns in the dataset and to produce the output shown below (Figure 1).



**Figure 1 Uploading Dataset, Checking data rows & columns, and Getting first 10 row**

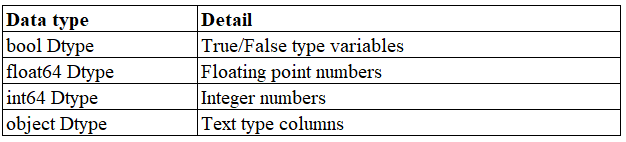
After loading the data, the data types are checked (Figure 2).



**Figure 2 Description of Data types**

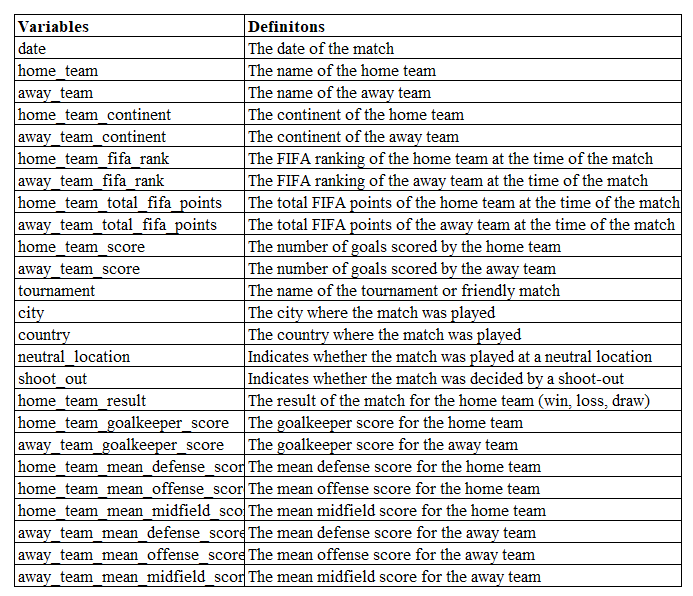
The Python code in Figure 2 was run to examine the variables and their types in the dataframe. Dataframe has object, float64, int64, bool type column which is; (Harrison and Petrou, 2020). Based on Table 1, bool Dtype is used for True/False type variables. Float64 Dtye is used for Floating point numbers. Int64 Dtype is used for integer numbers.

**Table 1 Data type and descriptions**



The variables in the data are explained in Table 2 of the report. The match's date is therefore included in the data. Additionally, the teams that host the match and those that do not have their score, point, and rank values calculated individually. The information also contains the city where the game was played, the type of competition, if it took place at a neutral site, whether a penalty shootout was used to decide the game, and the outcome of the game (win, defeat, or tie) for the host club.

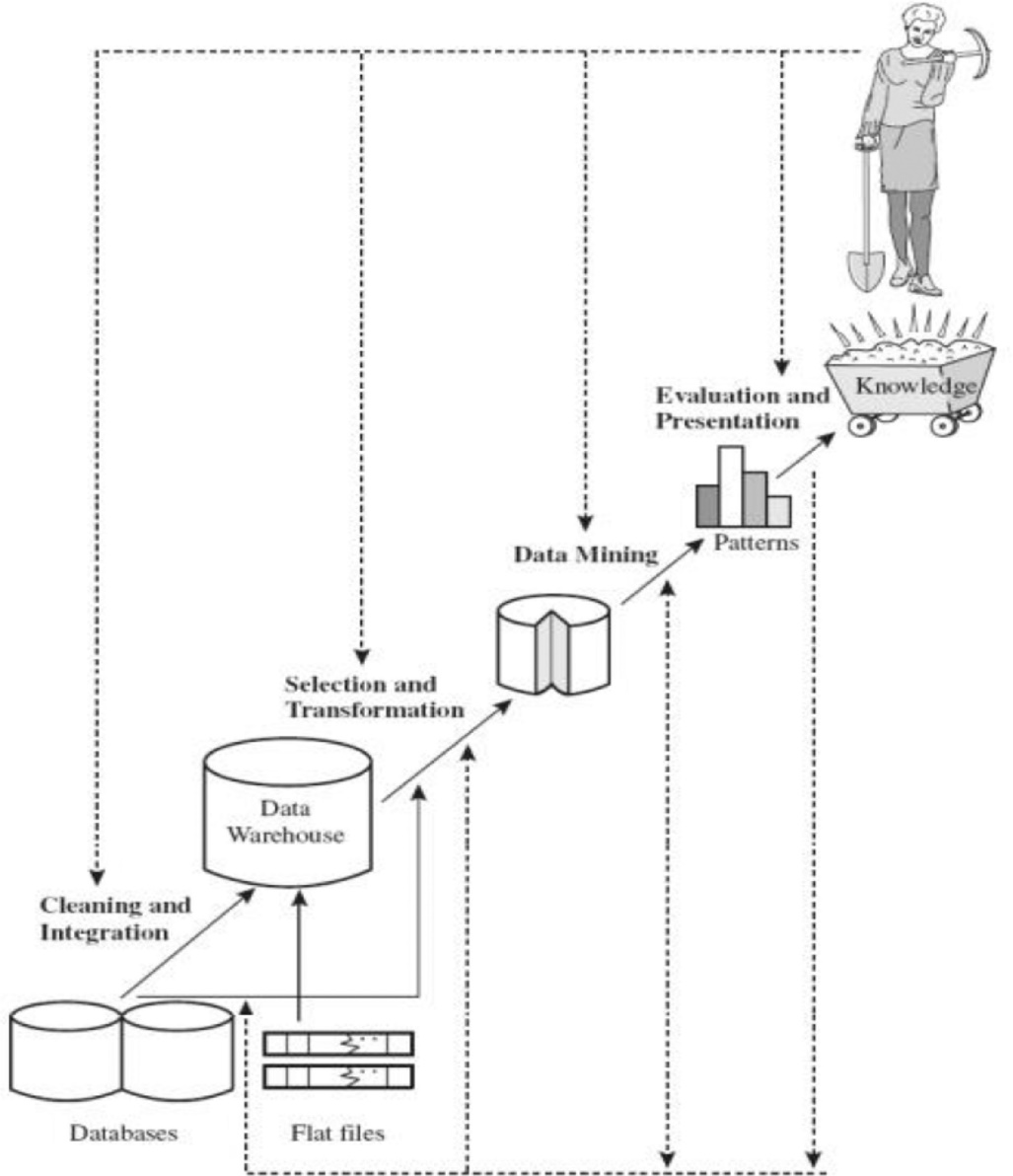
**Table 2 Definition of Variables**



# **Data pre-processing**

Pre-processing the data is a crucial step in getting it ready for machine learning. Maharana, et al, 2022 mentioned in their studies that, as shown in Figure 2, the data pre-proessing steps are as follows;

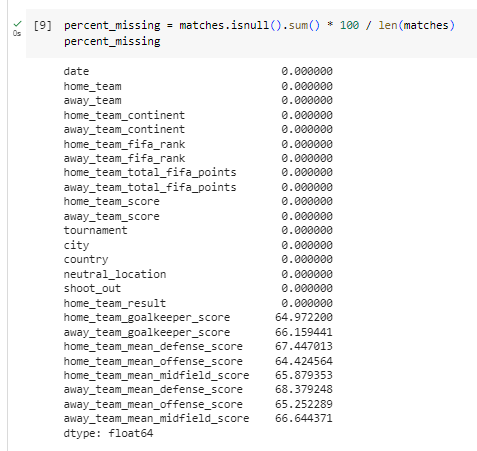
1. Cleaning
2. Transforming
3. Reduction



**Figure 3 Data pre-processing steps (Maharana, et al., 2022)**

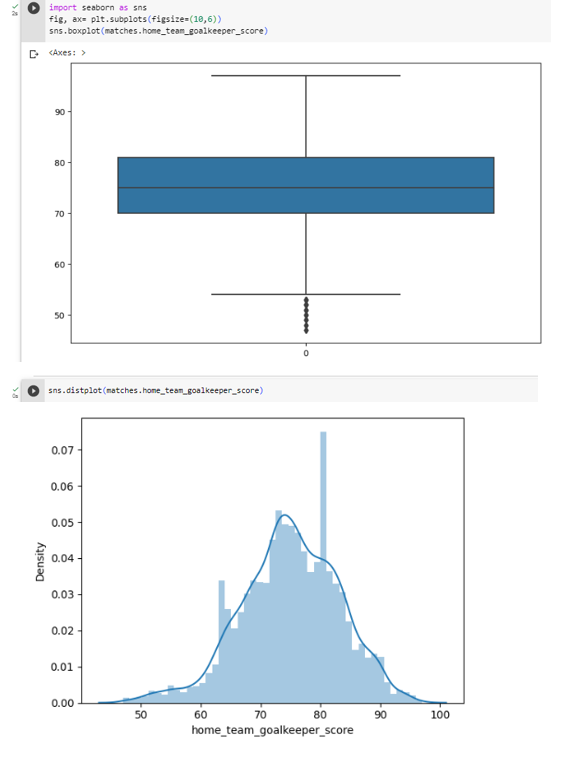
## **3.1. Data cleaning process**

### **3.1.1. Identify and handle missing values**

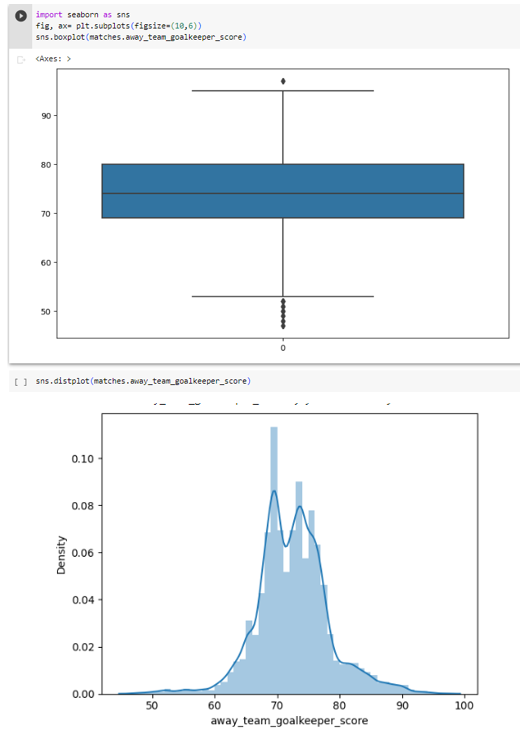


**Figure 4 Missing values in columns, percentage in totals**

Plots such as box plots and density plots come in very handy in deciding which techniques to use for filling out null values.

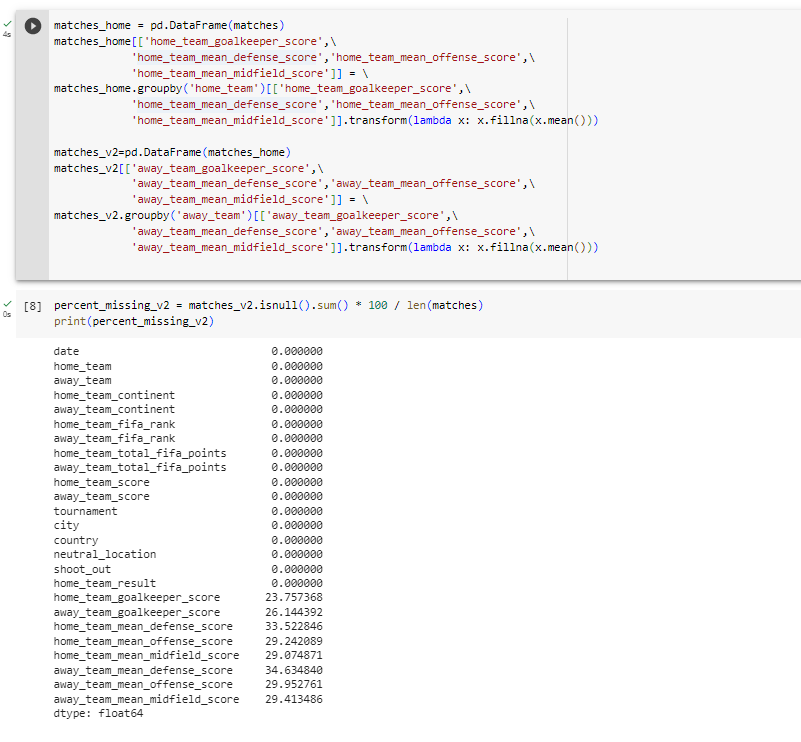


**Figure 5 Box plots and Density plots for columns that have null values**

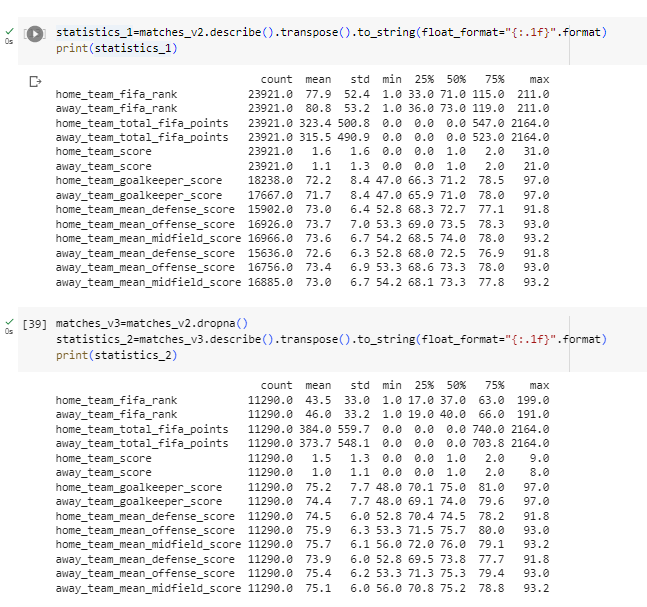


**Figure 6 Box plots and Density plots for columns that have null values**

Mean imputation is often used when the missing values are numerical and the distribution of the variable is approximately normal. As shown in Figures 5 and 6, the distribution of the variable is approximately normal. So, mean imputation can be used for filling out missing values.

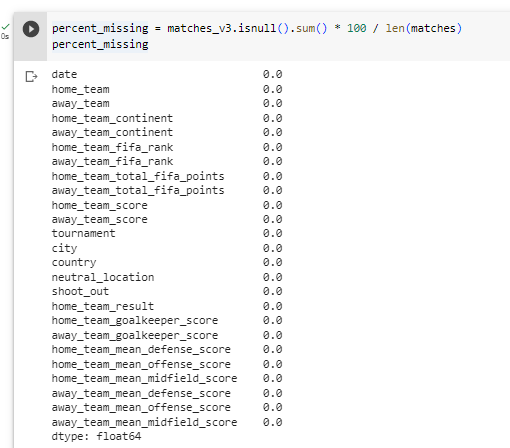


**Figure 7 Filling out null values**

Figure 7 illustrates how group by home team averages are used to fill in the null data for all variables in home team scores and group by away team averages for all variables in away team scores. Following the calculation, it was discovered that null data made up 23.8% of the home team's custodian score and 26.1% of the away team's custodian score.

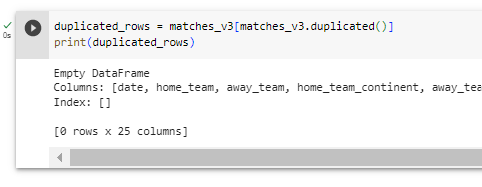
**Figure 8 Calculation of Statistics with and without null values**

Figure 8 illustrates a comparison of statistics between home\_team\_goalkeeper\_score and away\_team\_goalkeeper\_score with and without null data. As a result, even though the average of home\_team\_total\_fifa\_points was 323.4±500.8, it was determined to be 384.0±559.7 when null data were disregarded. Given that null data had little impact on the mean change, it was chosen to omit them. Similar to away\_team\_total\_fifa\_points, which had a mean of 315.5±490.9 before null data were removed, it was projected to be 373.7±548.1. Given that null data had little impact on the mean change, it was chosen to omit them.

When the statistical numbers in Figure 8 are reviewed, it is seen that the values for "home\_team\_total\_fifa\_points," "away\_team\_total\_fifa\_points," "home\_team\_score," and "away\_team\_score" all include the value "0." The value of "0" in these scores, however, was not disregarded because it also had significance, therefore it was included in the analysis.

**Figure 9 Percentege of null values in total after removing null values**

After removing null values, as shown in Figure 9, there are no null values in data.



**Figure 10 Analysing duplicate values**

After removing null values, as shown in Figure 10, there were no duplicate values.

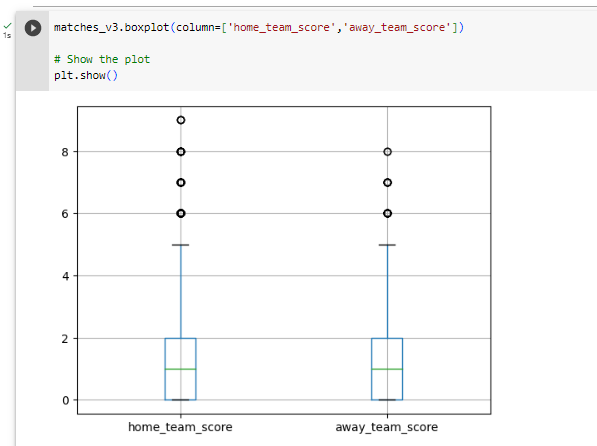
### **3.1.2. Identify and handle outliers**

One method for finding outliers is the interquartile range method. A statistical measure called the IQR tells us how evenly distributed and variable the data are. A box plot is used to display the distribution of the data, with whiskers extending from the box to the minimum and highest values. The whiskers reflect the interquartile range, or IQR, which represents the middle 50% of the data. Outliers are data points that are above or below the whiskers and are typically denoted by dots or asterisks (Andrea, et al., 2013).

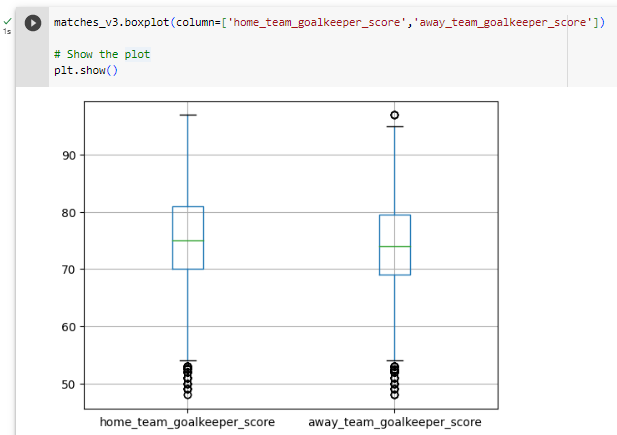
For total fifa points, team scores, and goal keeper scores, the box plot visialeted to detect outliers. As shown Figure 11, 12, and 13, the data included in the outlier data. However, these outlier data are thought to have meaning in the data. Therefore, it was not excluded from the analysis, normalization is used for reduce outliers.



**Figure 11 Boxplot for total fifa points**



**Figure 12 Boxplot for team scores**



**Figure 13 Boxplot for goalkepeer scores**

## **Transforming Data**

### **Normalization**

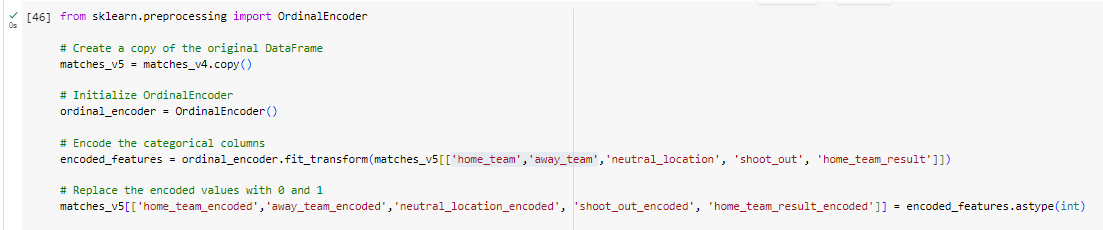
The performance of some machine learning algorithms can be enhanced by first normalising the numerical features such that they are all on a comparable scale. As shown in Figure 14, numerical values normamized.



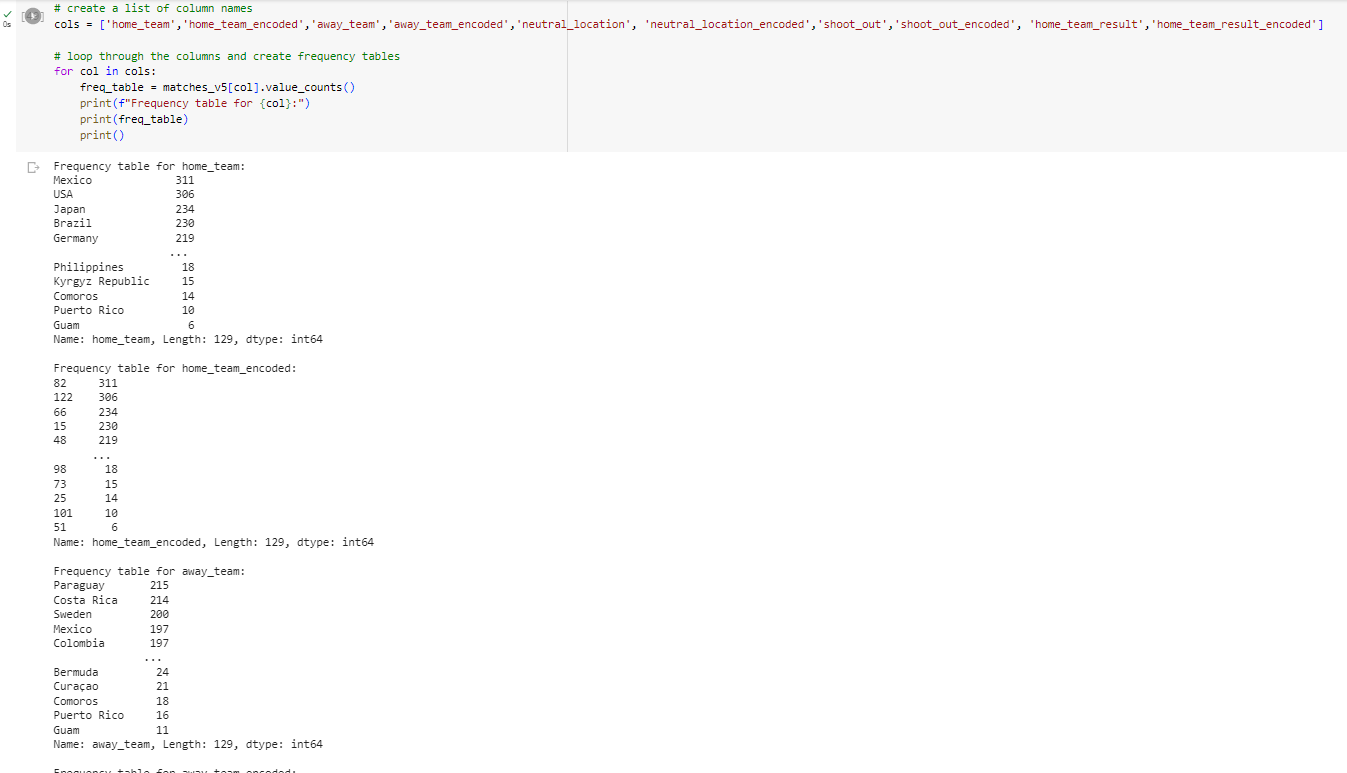
**Figure 14 Normalization (min-max normalization)**

### **Encoding categorical variables**

After normalisation, as shown in Figure 15, Ordinal encoding is applied and categorical variables encoding to give them a numerical representation. The encoding arrangement of the categorical data was checked by frequency analysis. As indicated in Figure 16, there is no error in encoding and it is coded correctly.



**Figure 15 Encoding Categorical Variables**

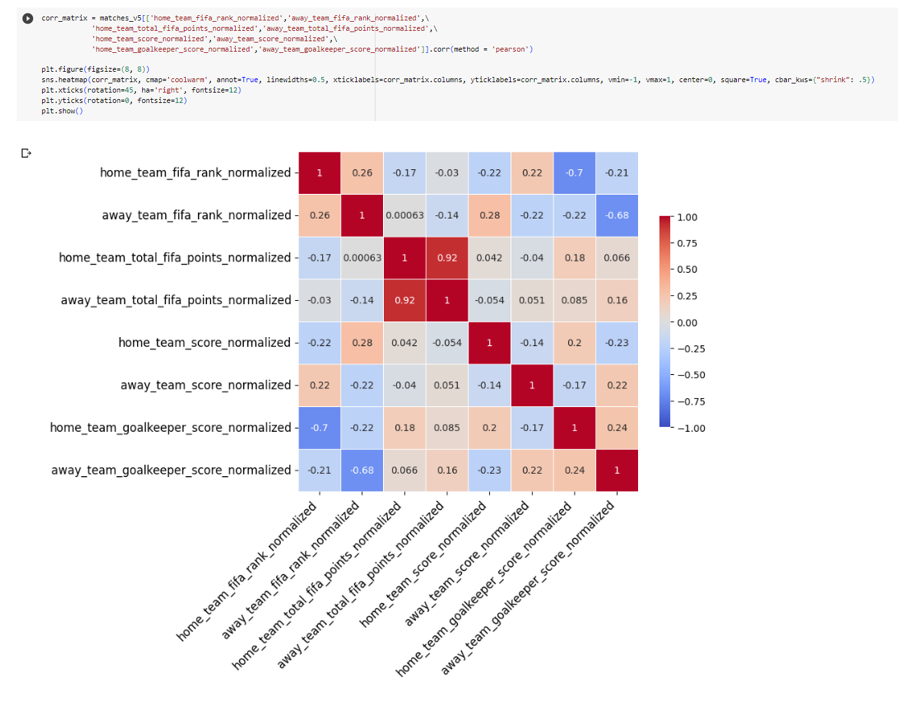


**Figure 16 Checking Encoding**

## **3.3. Reduction**

Correlation analysis was performed to determine the relationship between the normalized data and the data thought to be in the model (Figure 17). According to the analysis, as the goal keeeper score increases, the rank decreases, that is, the team rises to the front in the ranking. Likewise, as the goal keeper score decreases, the rank increases, so the team drops down in the ranking. In this case, it is very important that these two data are in the model.

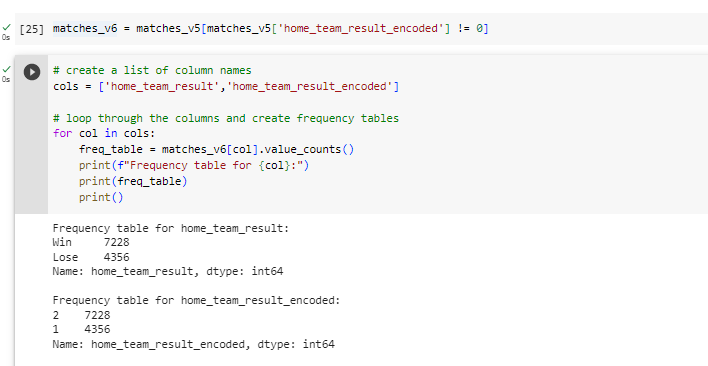
There is a high degree of positive correlation between the total fifa point home team and the away team. Accordingly, as the team's total fifa point increases, its total fifa point also increases. Due to high correlation, these two variables did not used in the models.



**Figure 17 Correlation Matrix**

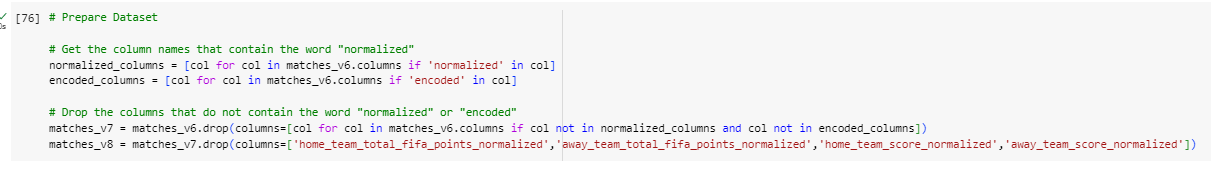
# **Model selection and training**

It was decided that it would be more meaningful for the model to remove the home team result draw data in the data. And the rows that draw from the dataframe are removed (Figure 18).



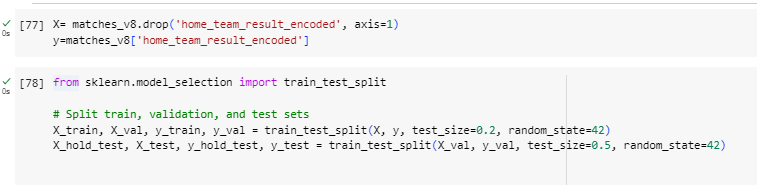
**Figure 18 Dropping Draw values from dataset**

After extracting the draw values from the dataset, a new dataset was prepared with encoding and normalized columns to make it ready for modelling (Figure 19).



**Figure 19 Preparing Dataset for modelling**

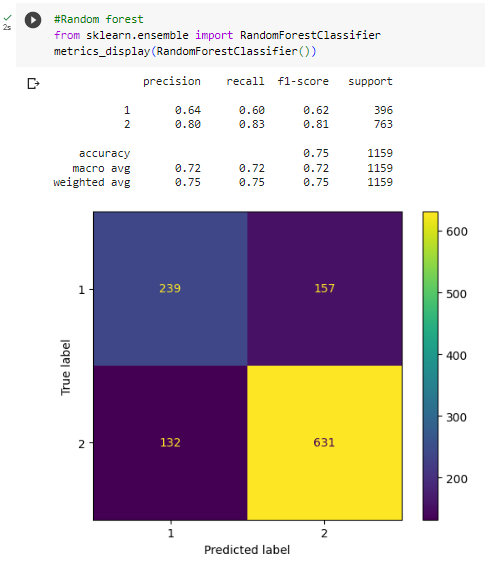
To prevent overfitting, the train, validate, and test split technique was used (Figure 20).



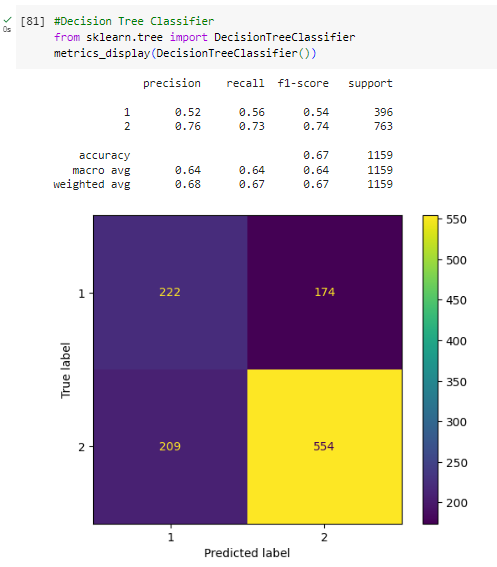
**Figure 20 Split train, validation, and test sets**

***MODELLING***

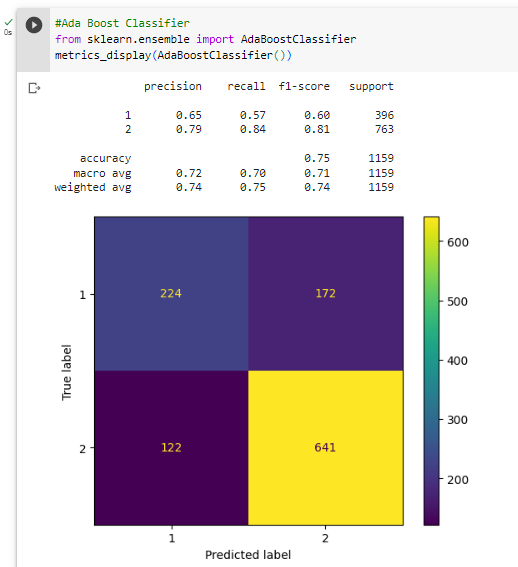
Popular machine learning algorithms used for classification and regression applications include Random Forest, Decision Tree, and AdaBoost. Powerful machine learning algorithms like Random Forest, Decision Tree, and AdaBoost provide many methods for enhancing prediction accuracy and managing complicated datasets. AdaBoost modifies its learning process to focus on challenging samples, while Random Forest makes use of the strengths of many Decision Trees. A decision tree is a straightforward but effective algorithm that uses feature values to guide a series of decisions that lead to a final prediction. It creates a structure resembling a tree where each leaf node represents a class or a regression value and each inside node reflects a judgement based on a particular feature.



**Figure 21 Random Forest Modelling**



**Figure 22 Decision Tree Modelling**

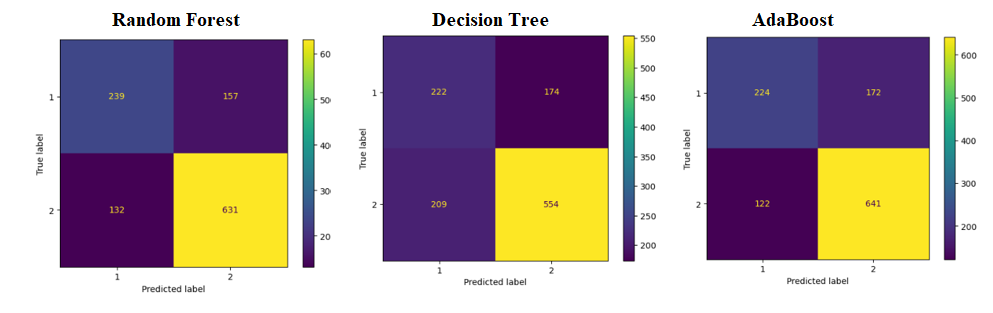


**Figure 23 AdaBoost Modelling**

# **Model evaluation and visualization**

***Confusion Matrix for Models***

A table that summarises a classification model's performance on a set of test data for which the true values are known is called a confusion matrix. It displays the number of samples that the model properly or erroneously predicts for each class. Additionally, it can be used to determine other metrics that assess the model's quality, including accuracy, precision, recall, and F1-score (Luque, et al., 2019). In academic research, confusion matrices are frequently employed to contrast various classifiers and to pinpoint their advantages and disadvantages. Based on Figure 26, we can said that, Random Forest will be best algorithm for this research.

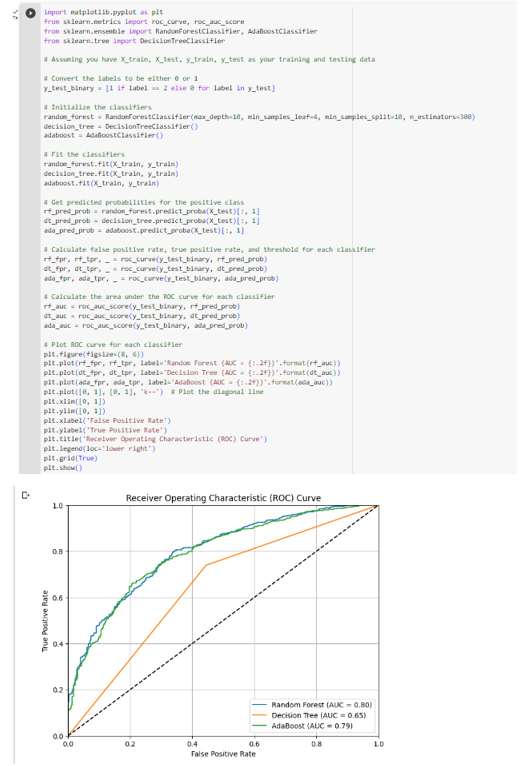


**Figure 24 Confusion Matrix for three model**

***ROC Curve***

The receiver operating characteristic curve (ROC curve) is a graph that displays how well a classification model performs across all categorization levels. The True Positive Rate (TPR) and False Positive Rate (FPR) are plotted on this curve (Khalilia, et al., 2011). To plot the Receiver Operating Characteristic (ROC) curve in Python, the “roc\_curve” function from the “sklearn.metrics module” is used.

As shown in Figure 27, when the Roc curve graphs are analyzed for all 3 models, the Random Forest method has the highest AUC value. Model selection will continue with the Ramdom Forest method.



**Figure 25 ROC Curve for three model**

# **Model deployment**

Based on accuracy calculation, It was decided to use Random Forest Classifier in the study. Before saving models, hyperparameters applied to find the best hyperparameters.

As shown in Figure 24, Best Hyperparameters:

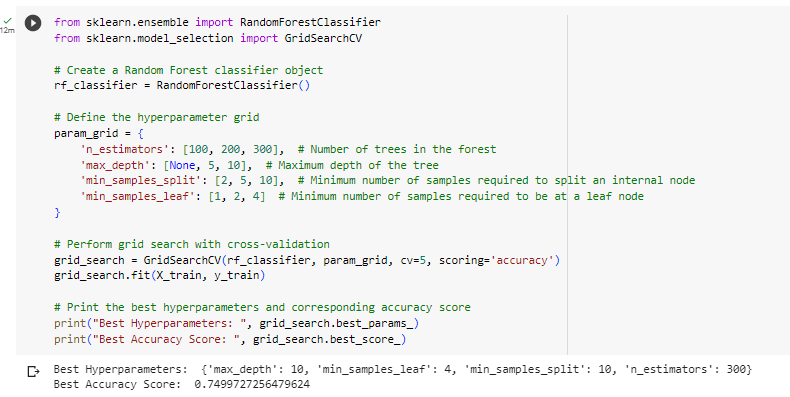
The minimum number of samples required to be at a leaf node, 'max\_depth': 10

The minimum number of samples required to be at a leaf node, 'min\_samples\_leaf': 4

The minimum number of samples required to split an internal node, 'min\_samples\_split': 10

The number of trees in the random forest, 'n\_estimators': 300

Best Accuracy Score: This score is an indication of how accurately the model performed when the best hyperparameters were used. The best accuracy score in this situation is 0.7499, meaning that the model, trained with the best hyperparameters, can accurately predict the target variable with a precision of roughly 74.99%.



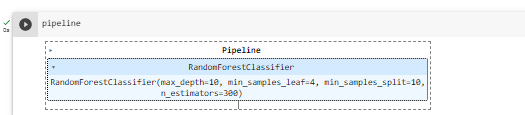
**Figure 26 Defining hyperparameters**

With these hyoerparameters, Random Forest Model was created and saved, as shown in Figure 25.



**Figure 27 Random Forest Classifier Pipeline**

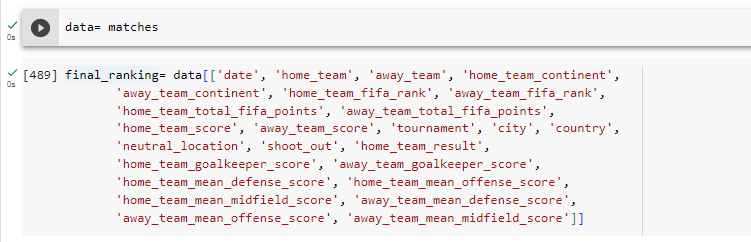
The finalized schema of pipeline is shown at Figure 28.

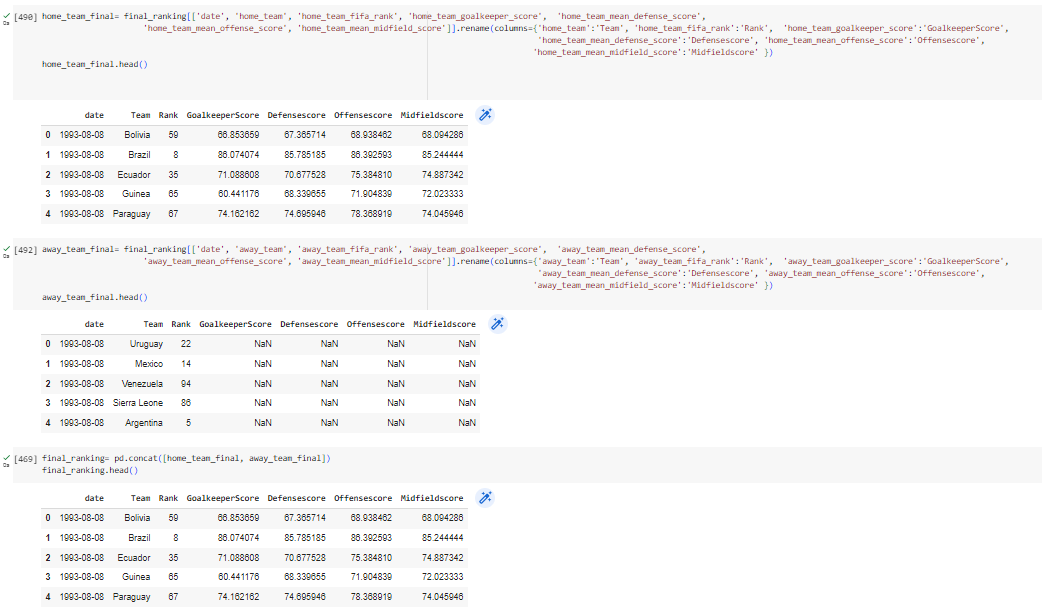


**Figure 28 Finalized schema of Pipeline**

**Simulation**

As shown in Figure 29, final\_ranking dataset is created to predict winners.

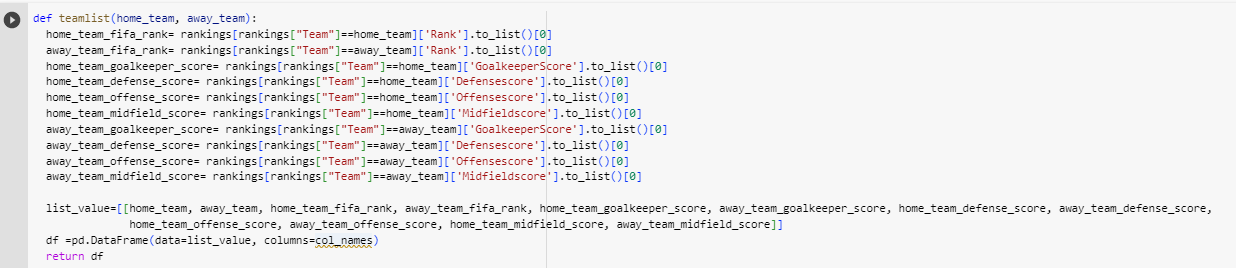
****

****



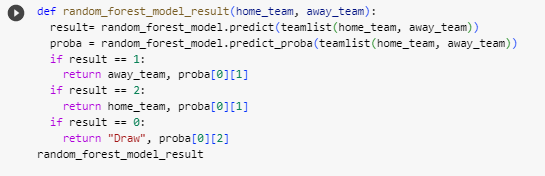
**Figure 29 Creating ranking dataset**

As shown in Figure 30, created a function that, when home\_team and away\_team values are imposed, returns a dataframe of values.



**Figure 30 Creating function**

For generating the result, as shown at Figure 31, the function is created.



**Figure 31 Generating results**

With random\_forest\_model, Simulation is created for predicting the winners. (Figure 32)



**Figure 32 Simulation**



|  | **Team** | **points** |
| --- | --- | --- |
| **3** | Netherlands | 8 |



|  | **Team** | **points** |
| --- | --- | --- |
| **0** | England | 10 |



|  | **Team** | **points** |
| --- | --- | --- |
| **0** | Argentina | 8 |



|  | **Team** | **points** |
| --- | --- | --- |
| **0** | France | 9 |



|  | **Team** | **points** |
| --- | --- | --- |
| **0** | Spain | 9 |



|  | **Team** | **points** |
| --- | --- | --- |
| **0** | Belgium | 10 |



|  | **Team** | **points** |
| --- | --- | --- |
| **0** | Brazil | 9 |

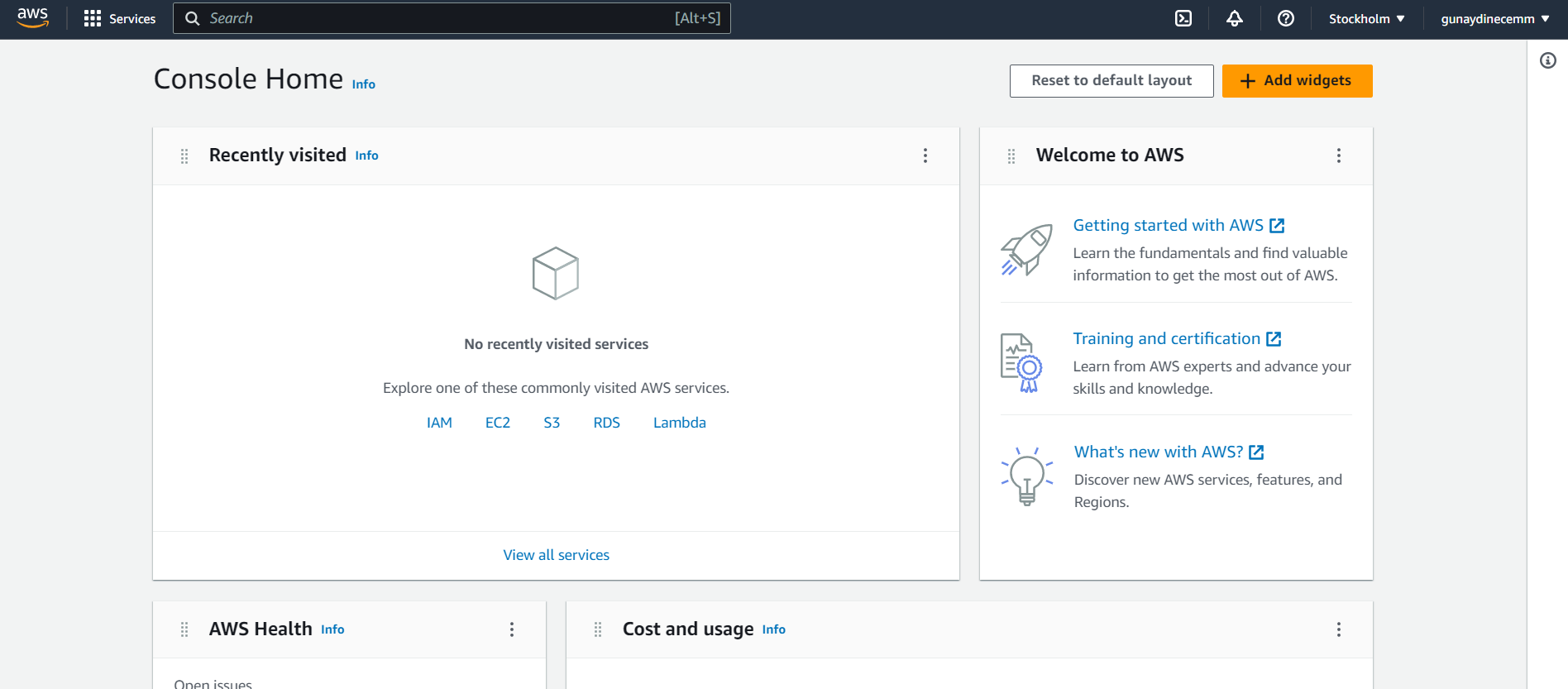


|  | **Team** | **points** |
| --- | --- | --- |
| **0** | Portugal | 8 |

**SageMaker**

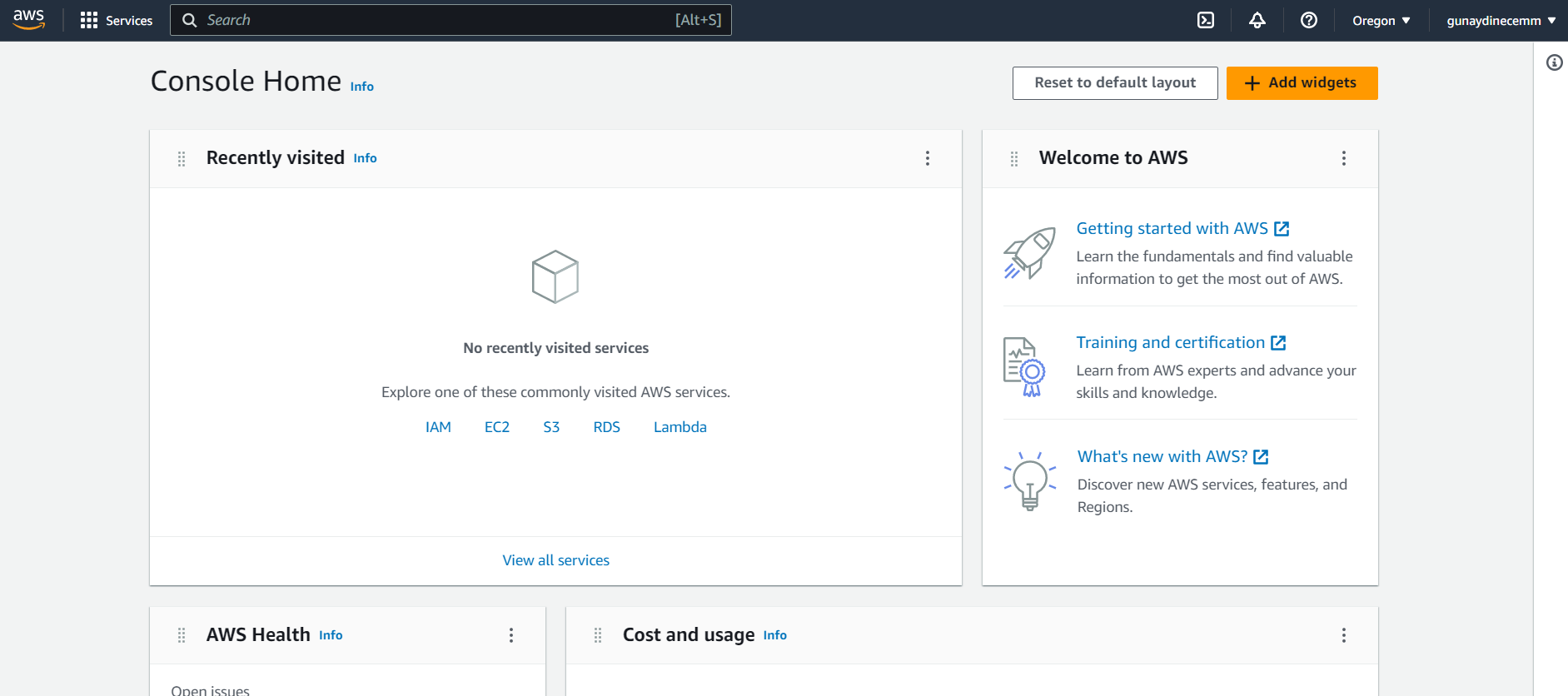
With Amazon SageMaker, the complexity is less and it makes it much simpler to develop and deploy ML models. SageMaker maintains all of the underlying infrastructure to train your model at petabyte size and deploy it to production after you select the appropriate algorithms and frameworks from the large range of options available (Amazon SageMaker ,2022)

First, the user should sign in to Amazon SageMaker console, if user has not a Amazon account, they have to create an account first. (Figure 29)



**Figure 33 Sign in to the Amazon SageMaker console**

Due to This tutorial using the US West (Oregon) Region, the user should change the region to Oregon. (Figure 30).



**Figure 34 Changing AWS Region**

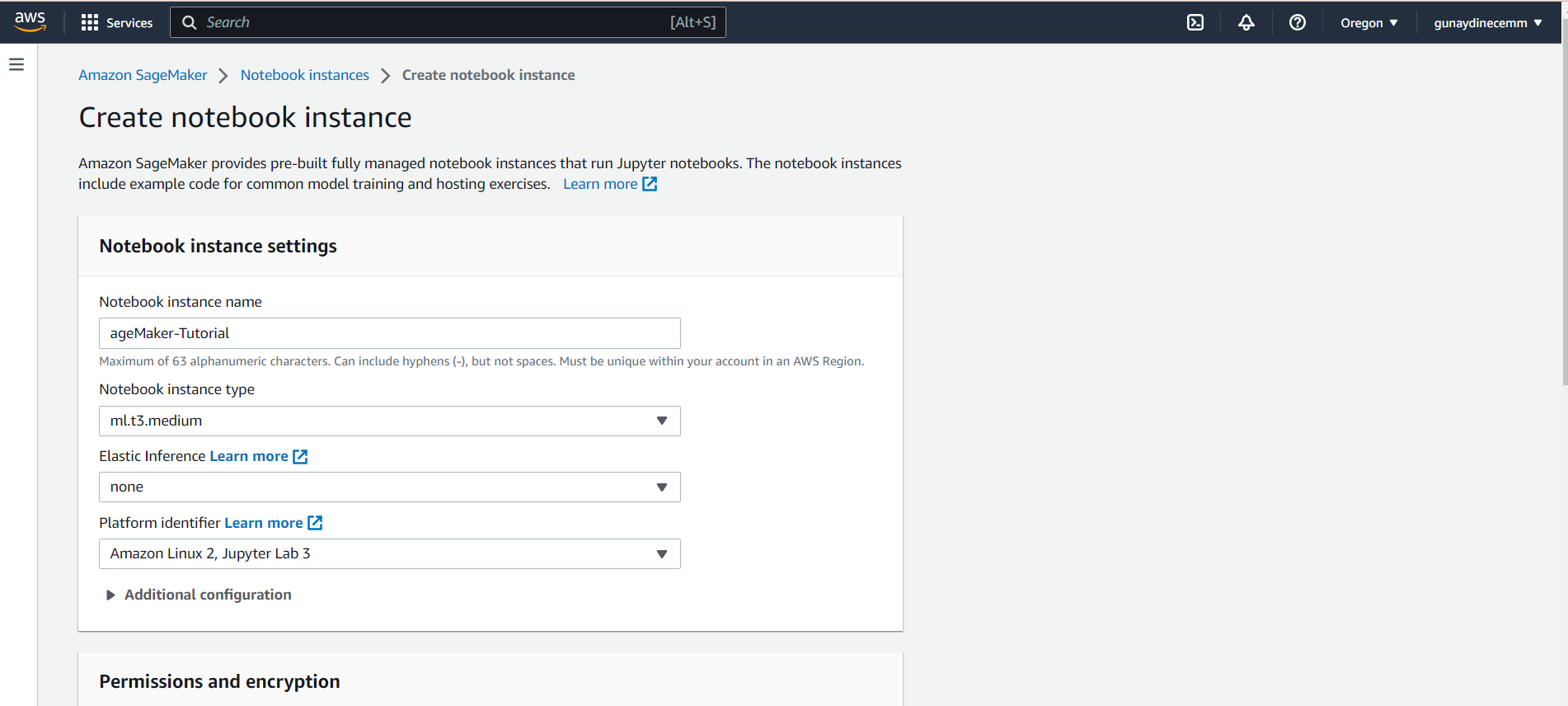
On the Create notebook instance page, in the Notebook instance setting box, filled the following fields (Figure 31).

For Notebook instance name, typed SageMaker-Tutorial.

For Notebook instance type, choosed ml.t2.medium.

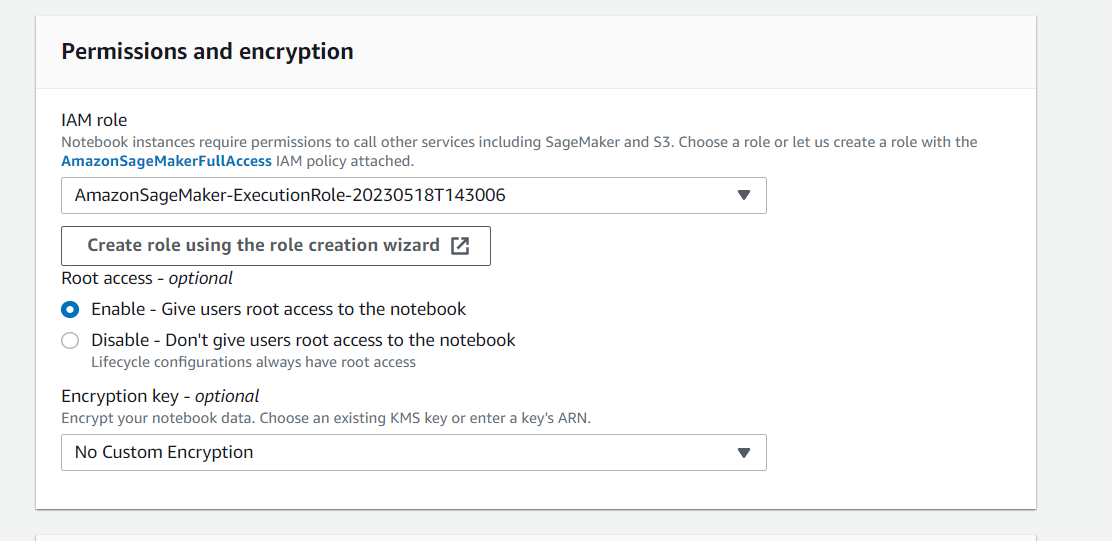
For Elastic inference, keep the default selection of none.

For Platform identifier, keep the default selection.



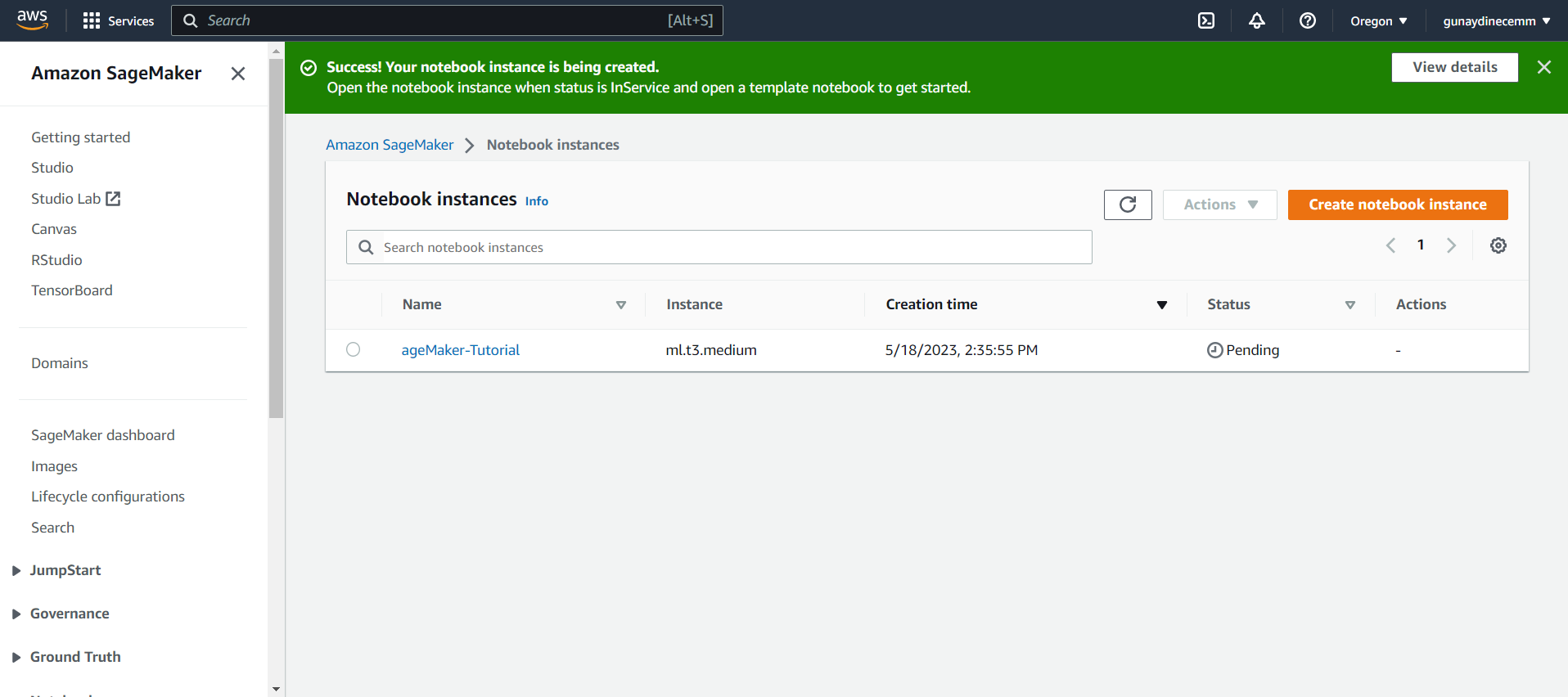
**Figure 35 Creating a Notebook**

I had a role so I just choosed the role that I created before (Figure 32)

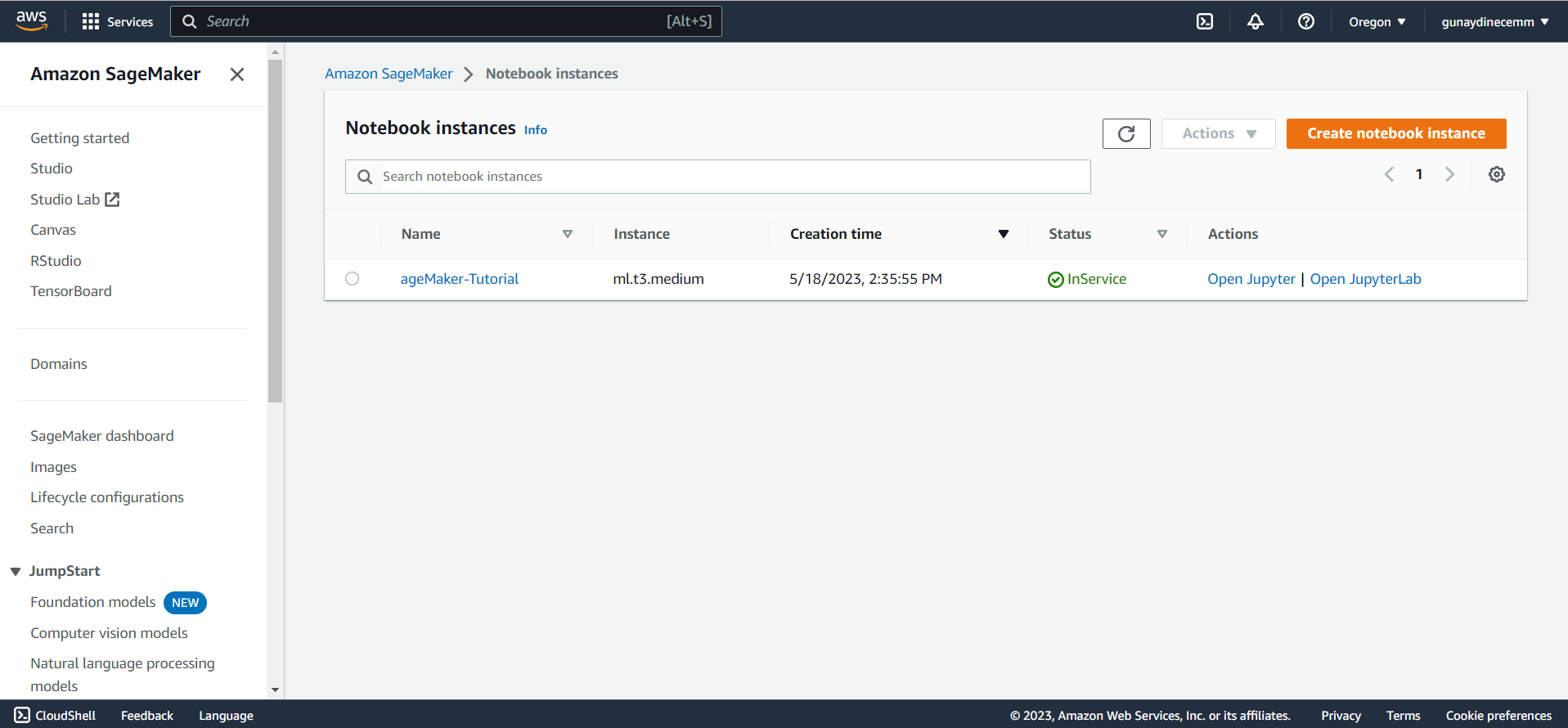


**Figure 36 Permissions and encryption**

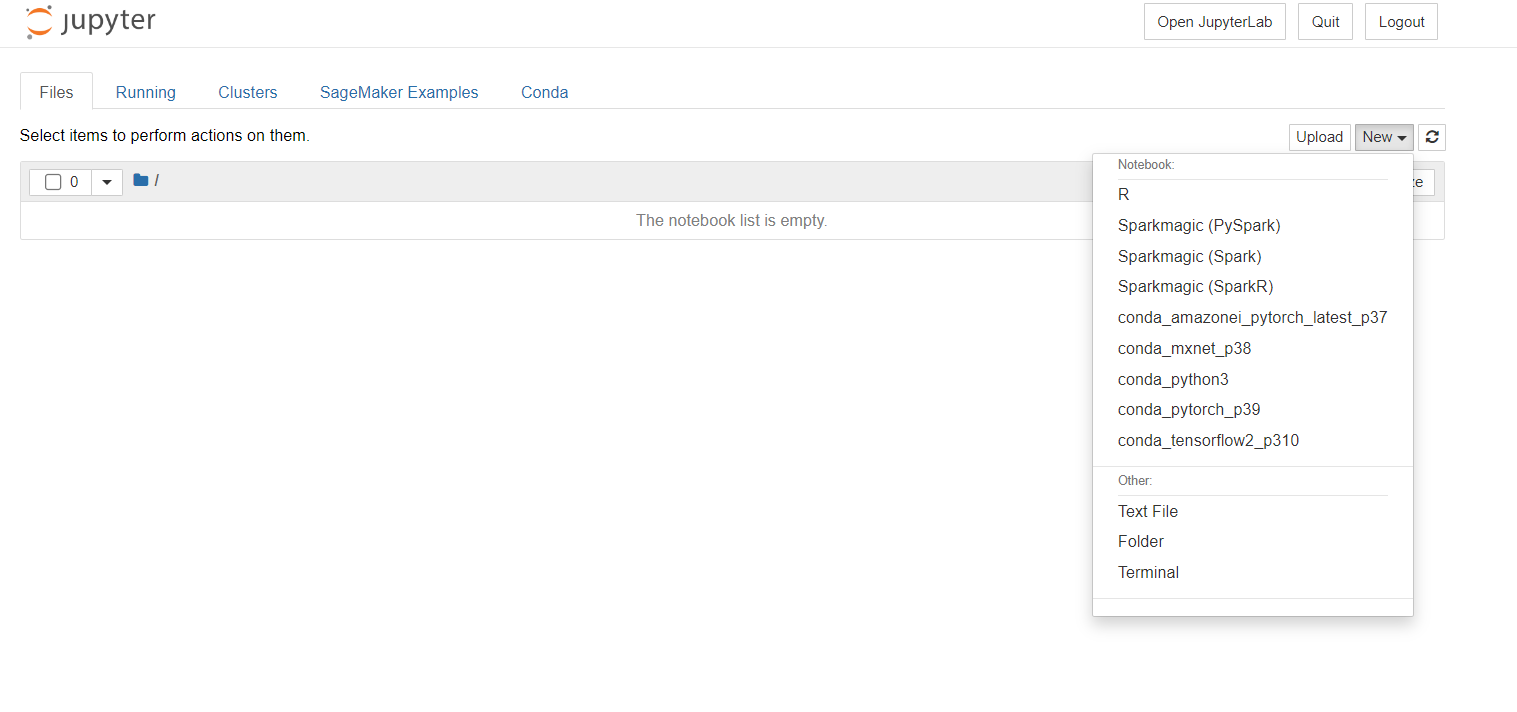
The new SageMaker-Tutorial notebook instance is shown in the section for notebook instances with a Status of Pending. When the Status changes to InService, the notebook is prepared (Figure 33).



**Figure 37 Create notebook instance**

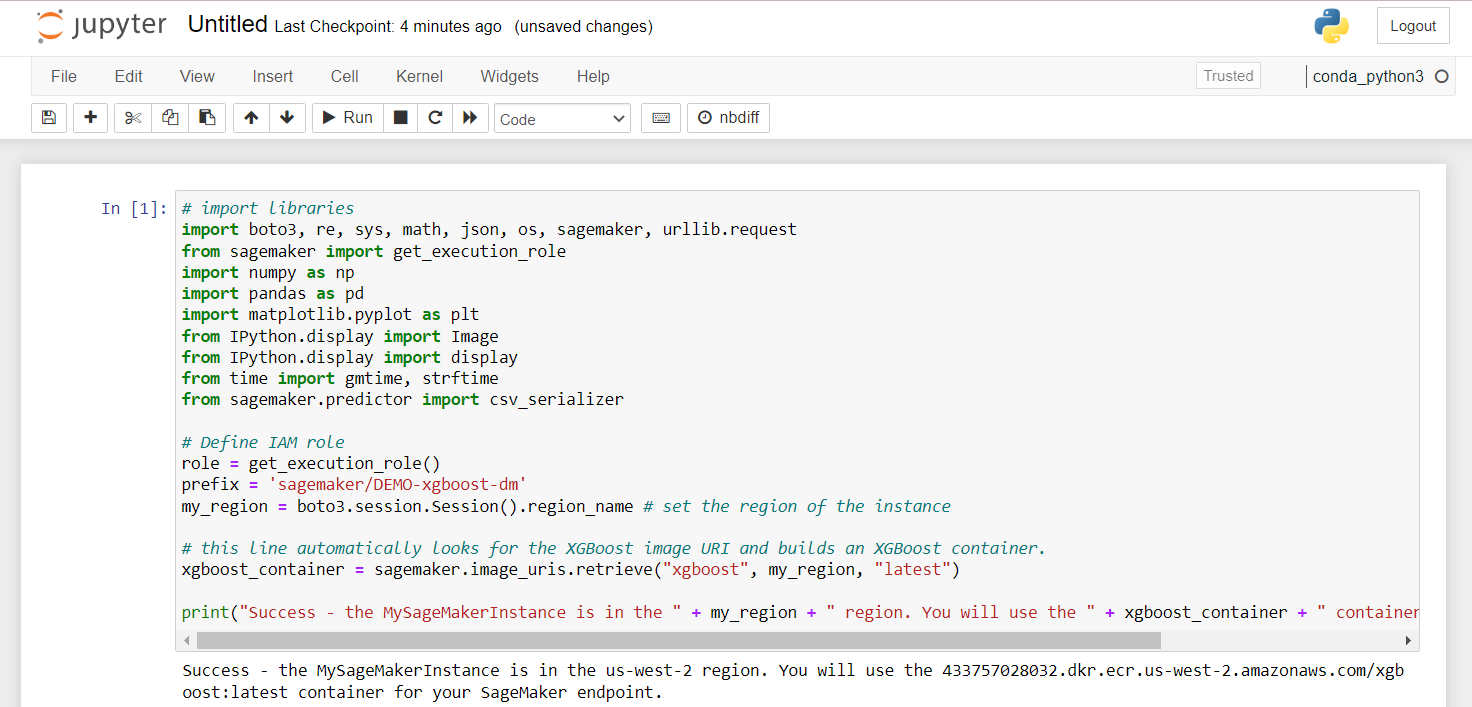


**Figure 38 Opening Jupiter**

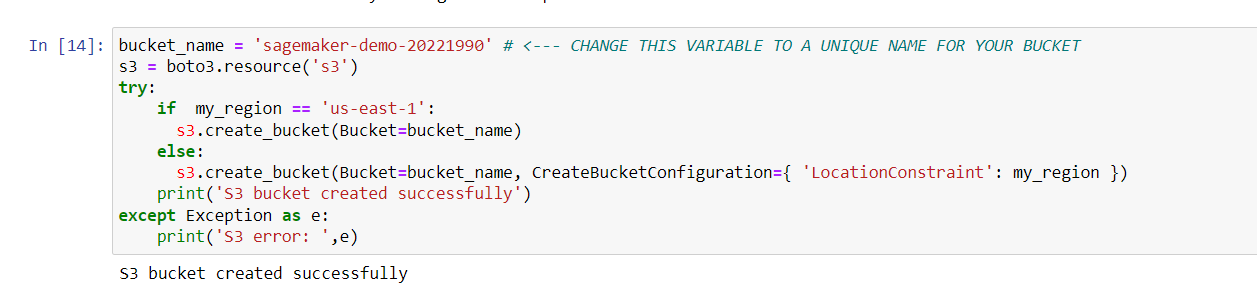


**Figure 39 Conda\_python3**

In order to prepare the data, train the ML model, and deploy it, this code imports the necessary libraries and defines the environment variables (Figure 40)



**Figure 40 Importing libraries for ML**

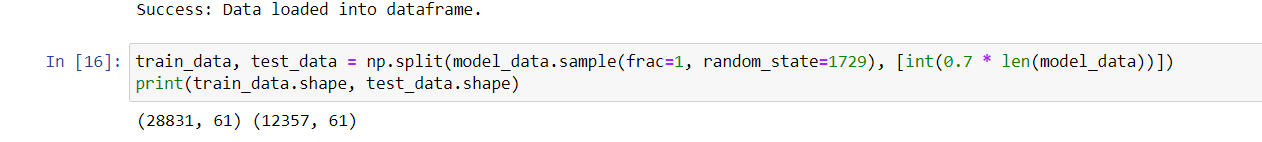


**Figure 41 Creating S3-Bucket**



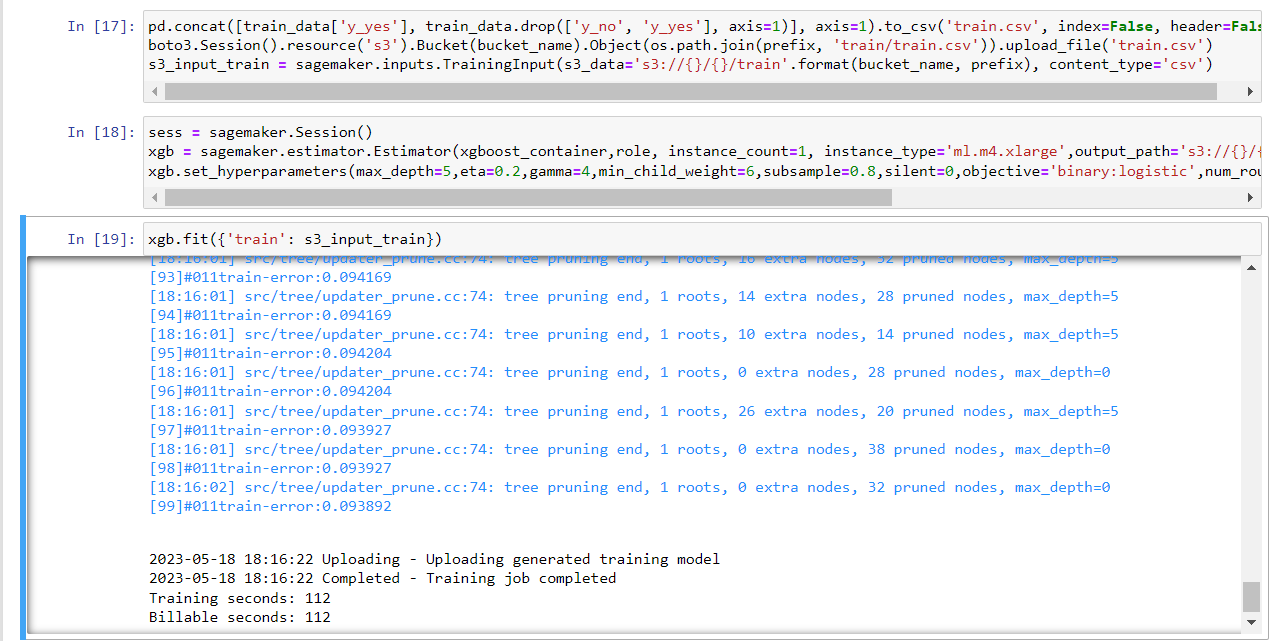
**Figure 42 Dowloading the SageMaker instance and load the data into a dataframe**

As shown at Figure 43, The training data (70% of customers) is used during the model training loop while The test data (the remaining 30% of customers) is used to evaluate the performance of the model and measure how well the trained model generalizes to unseen data.



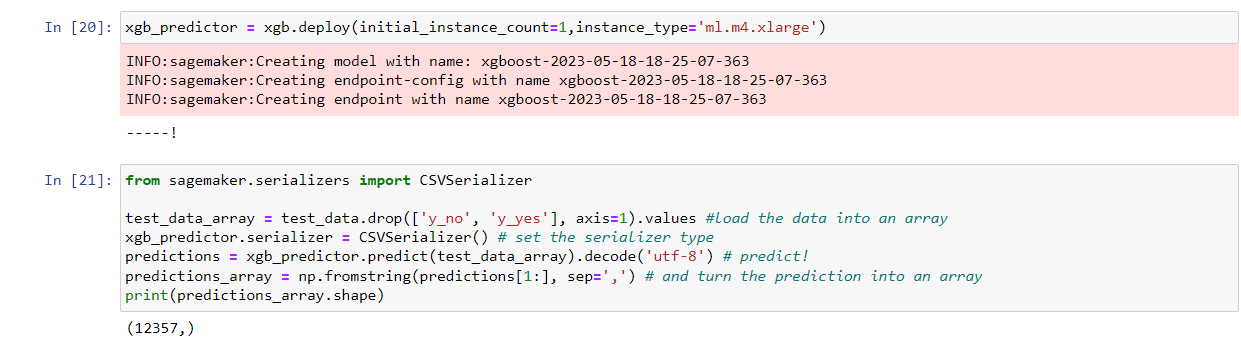
**Figure 43 Shuffle and split the data into training data and test data**

As shown at Figure 44 training dataset is used to train machine learning model. The training data's header and first column are reformatted by the 17th function, which then loads the data from the S3 bucket. To use the pre-built XGBoost algorithm from Amazon SageMaker, this step is necessary. The XGBoost model, an estimator, was created on 18th, along with the Amazon SageMaker session and its hyperparameters. The training task has begun with the 19th code (Amazon SageMaker, 2022).



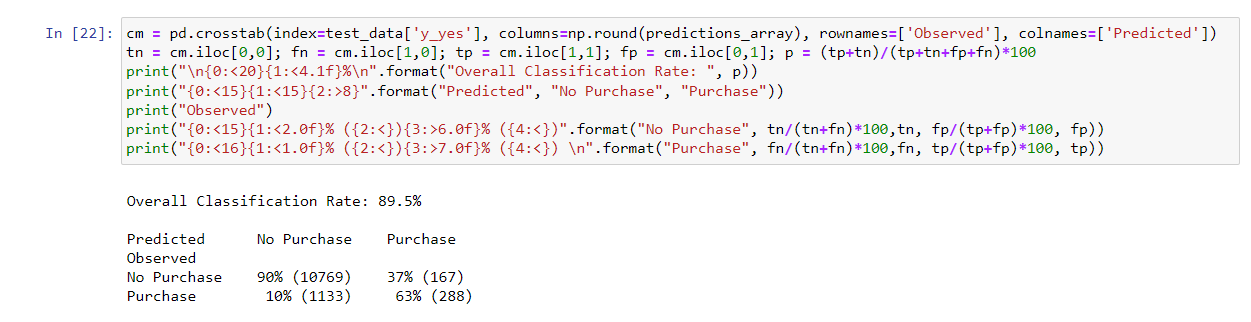
**Figure 44 Training the ML model**

Figure 45 shows the deployment of the trained model to an endpoint along with the reformatting, loading, and running of the model to produce predictions. The model is uploaded to a server by the 20th code, which also establishes a reachable SageMaker endpoint. The test data's client enrollment in the bank product was anticipated by the 21st code.



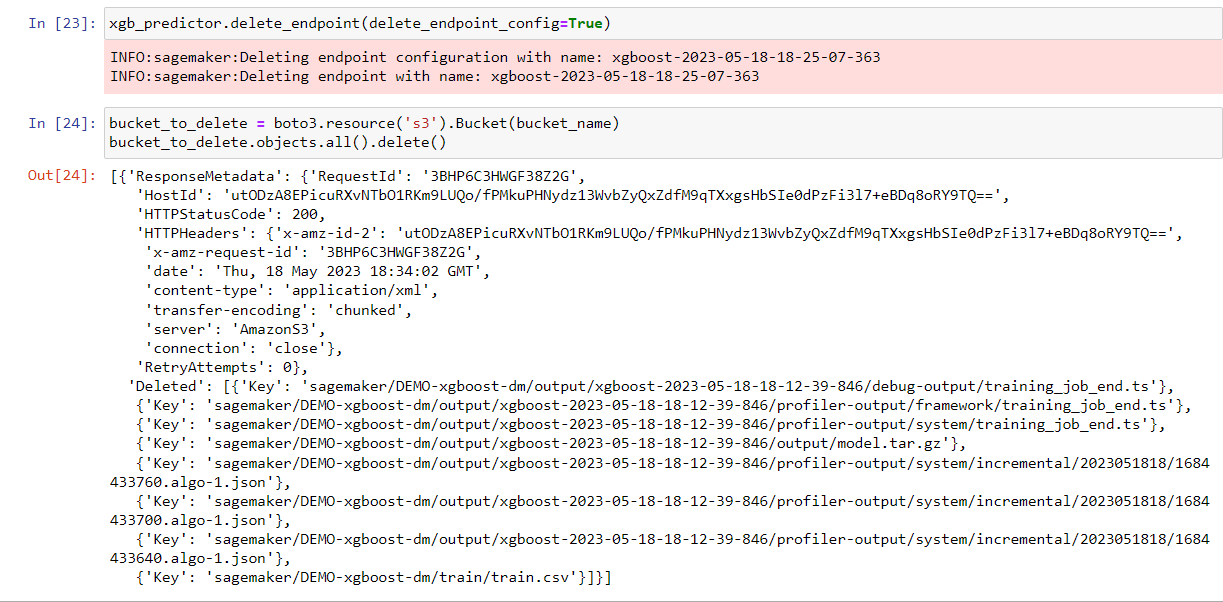
**Figure 45 Deployment the model**

Figure 46 shows that evaluate the performance and accuracy of the machine learning model. Based on the forecast, we can say that you correctly predicted that 90% of the test subjects would register in a certificate of deposit, with a precision of 65% (278/429) for enrolled and 90% (10,785/11,928) for didn't enrol.



**Figure 46 Evaluate model performance**

As shown in Figure 45, terminated the resources used in this lab. Terminating resources is important because it reduces costs and is a best practice.



**Figure 47 Clean up**

**Without using SageMaker**

There are several hosting platforms available, such as AWS EC2, Google Cloud Platform, Microsoft Azure, or Heroku that provision a virtual machine instance on your chosen hosting platform. The researchers need installation the necessary dependencies and libraries required for model deployment. Same as Sagemaker, the researchers create a deployment script that loads the trained model, sets up any preprocessing steps, and defines the inference logic. For deploying the model, the hosting platform's tools or APIs are used to deploy the model. After deploying the model, the researcher can test it by sending sample data and verifying the predictions.

**Ethical and Privacy Concerns**

*Data Governance and Security:*

SageMaker: For hosting machine learning models, Amazon SageMaker offers a scalable, secure infrastructure. It comes with built-in security features including access controls, encryption both in transit and at rest, and interaction with AWS Identity and Access Management (IAM) for precise access control.

Alternative Deployment: The researcher must make sure that a secure infrastructure is in place before deploying models without SageMaker. Utilising secure servers or virtual machines, encrypting data both in transit and at rest, establishing robust access restrictions and authentication procedures, and routinely monitoring and updating security configurations are all examples of how to do this.

*Data Anonymization and Privacy Protection:*

SageMaker: The researcher can incorporate privacy protection procedures and data anonymization strategies using SageMaker as part of the data pretreatment pipeline. Before training models, SageMaker's data transformation features enable data cleaning, transformation, and anonymization.

Alternative Deployment: When using data for training or inference without SageMaker, you must make sure that any sensitive or personally identifiable information (PII) has been appropriately anonymized or removed to preserve privacy.

*Model Transparency and Explainability:*

SageMaker: SageMaker offers tools and features to improve model explainability and transparency. SageMaker Clarify can identify and correct model biases, while SageMaker Debugger can analyse and visualise the training process.

Alternative Deployment: The researcher should aim for model openness and explainability while deploying without SageMaker. To offer justifications for the model's predictions, think about employing interpretable models or methods, such as rule-based models or methods like LIME or SHAP.

*Ethical Review and Oversight:*

SageMaker: SageMaker offers a platform where the researcher can incorporate various stakeholders in the model creation process but does not explicitly provide ethical review processes.

Alternative Deployment: To evaluate the potential societal impact, biases, fairness, and ethical ramifications of deploying the model, it is crucial to set up an ethical review mechanism within the organisation or include external experts.

*User Rights and Consent*

SageMaker: SageMaker doesn't offer any particular tools for controlling user permissions or consent.

Alternative Deployment: When deploying without SageMaker, it is important to make sure that users or data subjects have given their consent before any data is collected and used.

Overall, although there are no inherent ethical or privacy issues with Amazon SageMaker, it is crucial to take these issues into account before utilising SageMaker or any other machine learning platform. The platform's use, the data it processes, and the models that are implemented on it are essentially the sources of the worries.

# **Reference**

Amazon SageMaker (2022) *Build, Train, and Deploy a Machine Learning Model with Amazon SageMaker*, *Amazon*. Available at: https://aws.amazon.com/getting-started/hands-on/build-train-deploy-machine-learning-model-sagemaker/ (Accessed: 01 May 2023).

Andrea, K., Shevlyakov, G.L. and Smirnov, P.O., 2013. Detection of outliers with boxplots.

Dutta, P., 2019. Comparative study of cloud services offered by Amazon, Microsoft & Google. *International Journal of Trend in Scientific Research and Development*, *3*(3), pp.981-985.

Google (no date). *Google colab*. Google. Available at: https://research.google.com/colaboratory/faq.html (Accessed: April 25, 2023).

Groll, A., Ley, C., Schauberger, G. and Van Eetvelde, H., 2018. Prediction of the fifa world cup 2018-a random forest approach with an emphasis on estimated team ability parameters. *arXiv preprint arXiv:1806.03208*.

Gupta, B., Mittal, P. and Mufti, T., 2021, March. A review on Amazon web service (AWS), Microsoft azure & Google cloud platform (GCP) services. In *Proceedings of the 2nd International Conference on ICT for Digital, Smart, and Sustainable Development, ICIDSSD 2020, 27-28 February 2020, Jamia Hamdard, New Delhi, India*.

Harrison, M. and Petrou, T. (2020) “1,” in *Pandas 1.x cookbook: Practical recipes for scientific computing, Time Series analysis and exploratory data analysis using Python*. Birmingham, UK: Packt Publishing, pp. 6–9.

Khalilia, M., Chakraborty, S. and Popescu, M., 2011. Predicting disease risks from highly imbalanced data using random forest. *BMC medical informatics and decision making*, *11*, pp.1-13.

Luque, A., Carrasco, A., Martín, A. and de Las Heras, A., 2019. The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, *91*, pp.216-231.

Maharana, K., Mondal, S. and Nemade, B., 2022. A review: Data pre-processing and data augmentation techniques. *Global Transitions Proceedings*.