SmartInternz

www.smartinternz.com

Masterful Machines: Precise Coffee Quality Predictions

Project Hand-out, Faculty Development Program – NaanMudhalvan

Text  Description automatically generated with low confidence

Masterful Machines: Precise Coffee Quality Predictions

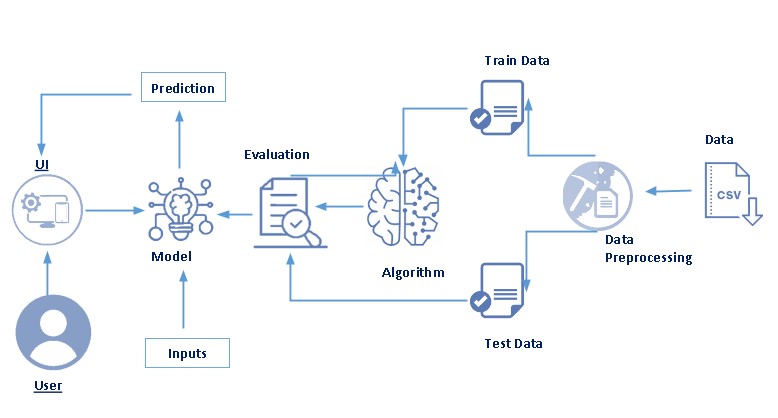
# Project Description:

**The Coffee Quality Institute (CQI) is a non-profit organization that works to improve the quality and value of coffee worldwide. It was founded in 1996 and has its headquarters in California, USA.**

**CQI's mission is to promote coffee quality through a range of activities that include research, training, and certification programs. The organization works with coffee growers, processors, roasters, and other stakeholders to improve coffee quality standards, promote sustainability, and support the development of the specialty coffee industry.**

**CQI maintains a web database that serves as a resource for coffee professionals and enthusiasts who are interested in learning about coffee quality and sustainability. The database includes a range of information on coffee production, processing, and sensory evaluation. It also contains data on coffee genetics, soil types, and other factors that can affect coffee quality.**

# Technical Architecture:



# Pre requisites:

**To complete this project, you must required following software’s, concepts and packages**

* **Anaconda navigator and pycharm:**
  + Refer the link below to download anaconda navigator
  + Link : <https://youtu.be/1ra4zH2G4o0>
* **Python packages:**
  + Open anaconda prompt as administrator
  + Type “pip install numpy” and click enter.
  + Type “pip install pandas” and click enter.
  + Type “pip install scikit-learn” and click enter.
  + Type ”pip install matplotlib” and click enter.
  + Type ”pip install scipy” and click enter.
  + Type ”pip install pickle-mixin” and click enter.
  + Type ”pip install seaborn” and click enter.
  + Type “pip install Flask” and click enter.

# Prior Knowledge:

You must have prior knowledge of following topics to complete this project.

* **ML Concepts**
  + Supervised learning: <https://www.javatpoint.com/supervised-machine-learning>
  + Unsupervised learning: <https://www.javatpoint.com/unsupervised-machine-learning>
  + Linear regression **:https://www.javatpoint.com/linear-regression-in-machine-learning**
  + Decision tree: <https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>
  + Random forest: <https://www.javatpoint.com/machine-learning-random-forest-algorithm>
  + KNN: <https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning>
  + Xgboost: <https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/>
  + Evaluation metrics: <https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/>
* **Flask Basics** : <https://www.youtube.com/watch?v=lj4I_CvBnt0>

# Project Objectives:

By the end of this project you will:

* Know fundamental concepts and techniques used for machine learning.
* Gain a broad understanding about data.
* Have knowledge on pre-processing the data/transformation techniques on outlier and some visualization concepts.

# Project Flow:

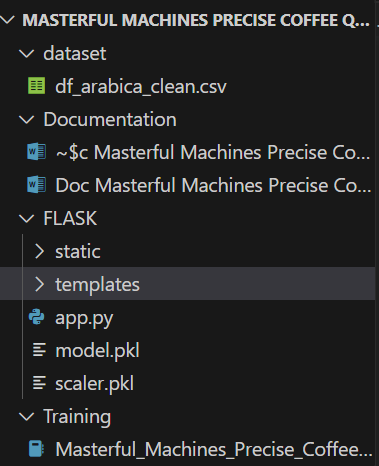
* User interacts with the UI to enter the input.
* Entered input is analyzed by the model which is integrated.
* Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

* Data collection
  + Collect the dataset or create the dataset
* Visualizing and analyzing data
  + Univariate analysis
  + Bivariate analysis
  + Multivariate analysis
  + Descriptive analysis
* Data pre-processing
  + Checking for null values
  + Handling outlier
  + Handling categorical data
  + Splitting data into train and test
* Model building
  + Import the model building libraries
  + Initializing the model
  + Training and testing the model
  + Evaluating performance of model
  + Save the model
* Application Building
  + Create an HTML file
  + Build python code

# Project Structure:

Create the Project folder which contains files as shown below



* We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
* model.pkl is our saved model. Further we will use this model for flask integration.

# Milestone 1: Define Problem / Problem Understanding

# Activity 1: Specify the business problem

## Our project aims to predict quality of coffee beans on a machine learning platform. To achieve this goal, we will analyze various features such as'Unnamed: 0', 'ID', 'Country of Origin', 'Farm Name', 'Lot Number', 'Mill', 'ICO Number', 'Company', 'Altitude', 'Region', 'Producer', 'Number of Bags', 'Bag Weight', 'In-Country Partner', 'Harvest Year', 'Grading Date', 'Owner', 'Variety', 'Status', 'Processing Method', 'Aroma', 'Flavor', 'Aftertaste', 'Acidity', 'Body', 'Balance', 'Uniformity', 'Clean Cup', 'Sweetness', 'Overall', 'Defects', 'Total Cup Points', 'Moisture Percentage', 'Category One Defects', 'Quakers', 'Color', 'Category Two Defects', 'Expiration', 'Certification Body', 'Certification Address', 'Certification Contact'. We will preprocess the dataset by handling missing values and converting categorical variables into numerical ones using techniques such as one-hot encoding or Label Encoding and data preparation according to data. Then, we will split the dataset into training and testing sets and use classification algorithms such as Linear Regression, Decision Tree, Random Forest, KNN, and XGBoost to train and test the data. Finally, we will select the best-performing model and save it in a pickle format. This model will then be deployed using Flask integration.

## Activity 2: Business requirements

## To analyze quality of beans in a machine learning platform, there are several business requirements that should be considered, depending on the specific goals and objectives of the project. Some potential requirements may include:

## ● Accuracy: The solution should be accurate in predicting quality. This is important because inaccurate predictions could lead to the spread of misinformation on the platform when estimating quality.

## ● Speed: The solution should be able to process quality of beans data quickly. This is important because timely responses to predict quality can help prevent the spread of misleading quality of beans to customers and society get some idea about coffee beans.

## Activity 3: Literature Survey (Student Will Write)

## A literature survey for a coffee beans quality analysis project would involve researching and reviewing existing studies, articles, and other publications on the topic of coffee bean quality analysis. One such study proposed an approach for identifying quality on a machine learning platform using various features in coffee beans data. The approach was evaluated on a dataset of real-world quality and achieved an accurate R2 of 0.99 and RMSE loss of 0.07. The study demonstrated the effectiveness of machine learning for quality prediction on a machine learning platform. The literature survey would also examine other studies that have explored different techniques and methods for quality analysis, as well as the challenges and limitations associated with quality estimation. By conducting a comprehensive literature survey, the coffee beans quality analysis project can gain valuable insights and knowledge that can inform the development of effective bot detection models and methods.

## Activity 4: Social or Business Impact.

**Social Impact :-** The social impact of quality analysis is significant as it can help prevent the spread of misinformation and promote transparency and accountability on machine learning platforms. By using machine learning algorithms, quality analysis can identify exact approximate qualities that may be influencing content. This can help prevent the spread of false information and ensure that quality is genuine.

**Business Model/Impact :-** The business impact of quality of coffee beans prediction analysis can improve the effectiveness of machine learning platforms. By analyzing various features of beans data, it can identify potential quality that may be misusing the platform. This can help machine learning platform providers to identify and address quality of beans earlier, reducing the potential negative impact on user experience. Additionally, effective predicting quality analysis can help businesses and industries to maintain the integrity of their data and prevent fraudulent activities.

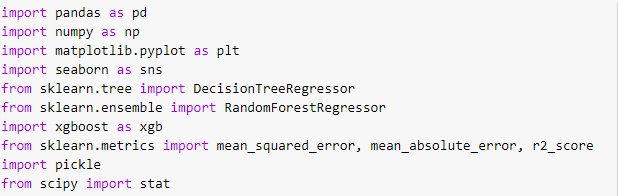
# Milestone 2: Visualizing and analysing the data

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analysing techniques.

**Note: There is n number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.**

**Activity 1: Importing the libraries**

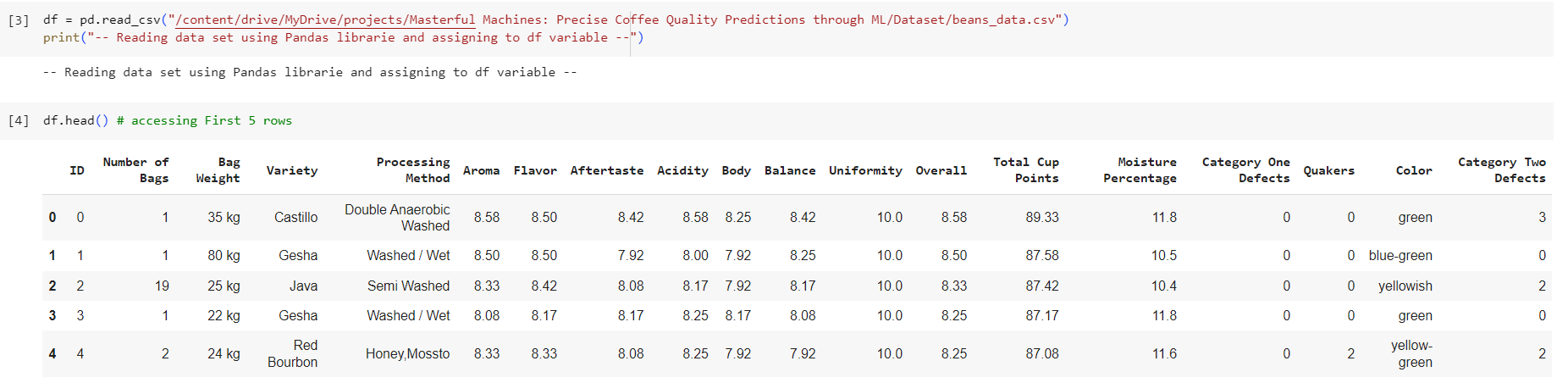
**Import the necessary libraries as shown in the image. (optional) Here we have used visualization.**



**Activity 2: Read the Dataset**

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.

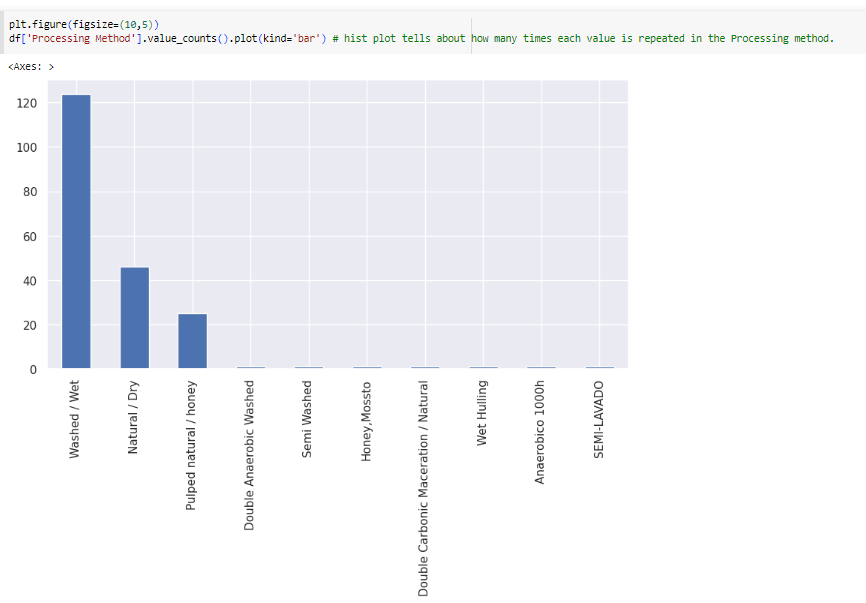
In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of csv file.



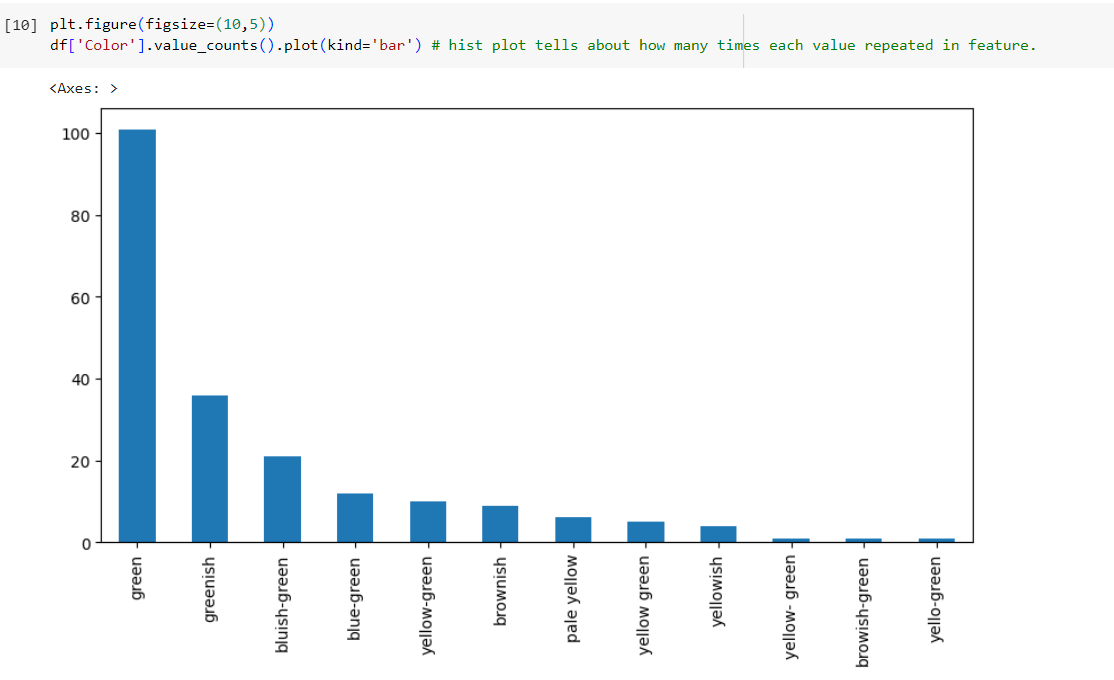
**Activity 3: Univariate analysis**

In simple words, univariate analysis is understanding the data with single feature. Here we have displayed two different graphs such as count plot, histogram and pie plot.

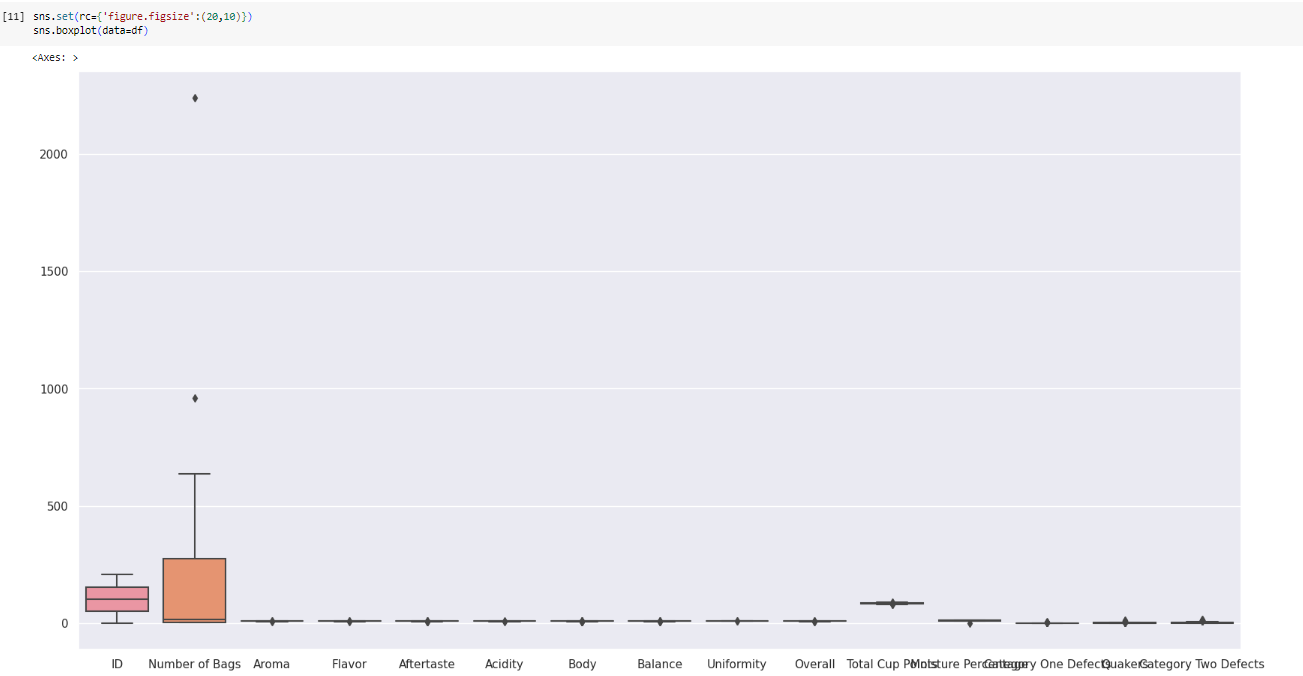
* Seaborn package provides a wonderful function bar plot. With the help of bar plot, we can find the how many times each value repeated in processing methods of the beans from the feature.



* From the above we can understand how types of bean processing methods with count.



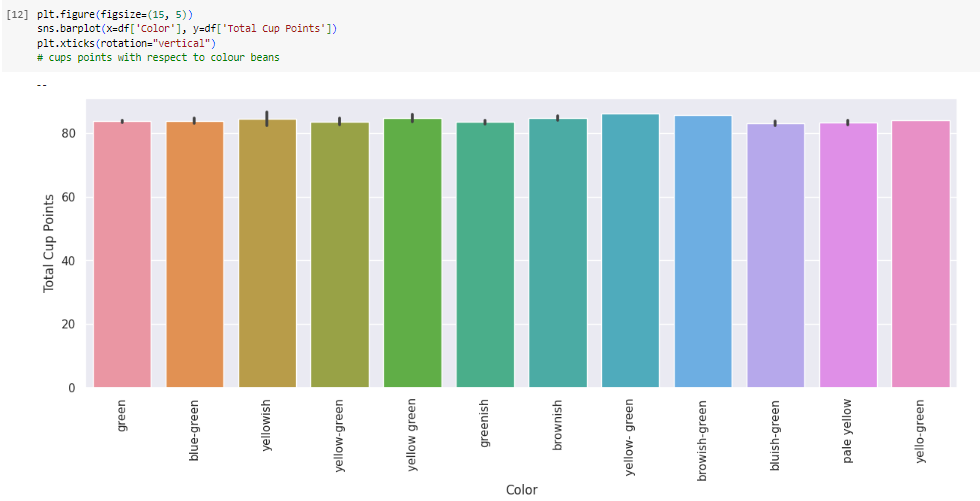
* From the above we can understand types of colors and count.



* From the above plot we came to know outliers in data.

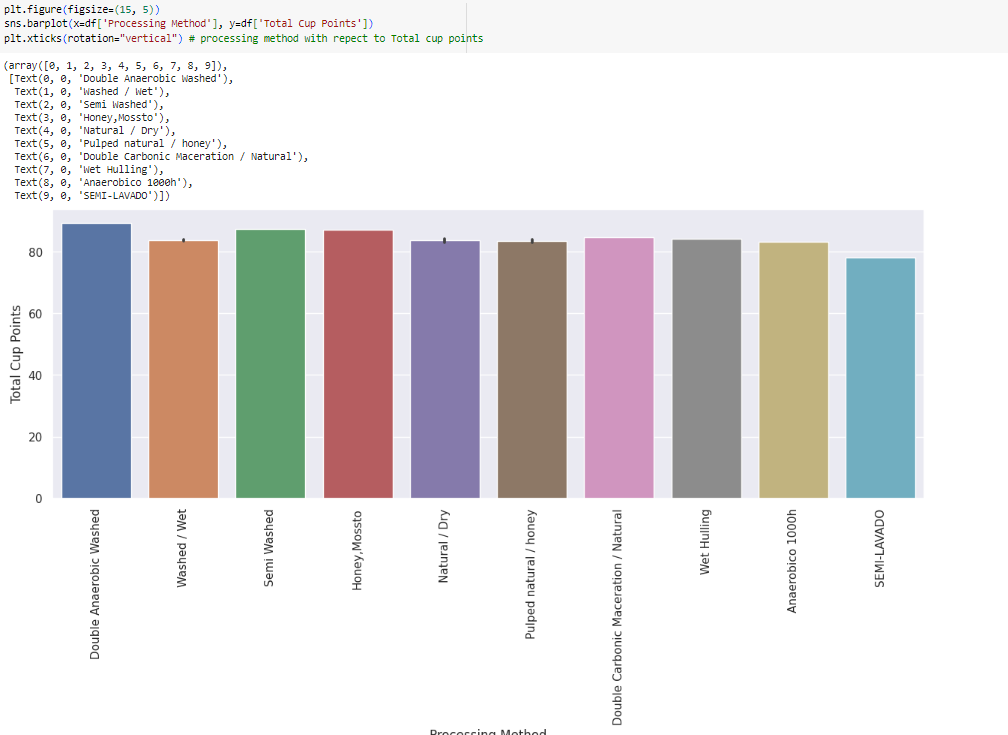
**Activity 4: Bi-variate analysis**

To find the relation between two features we use bi-variate analysis. Here we are visualizing the relationship between color and total cup points.



* This plot tells about mean cup points of each coloured bean.

Here we are visualizing the relationship between processing method and total cup points.



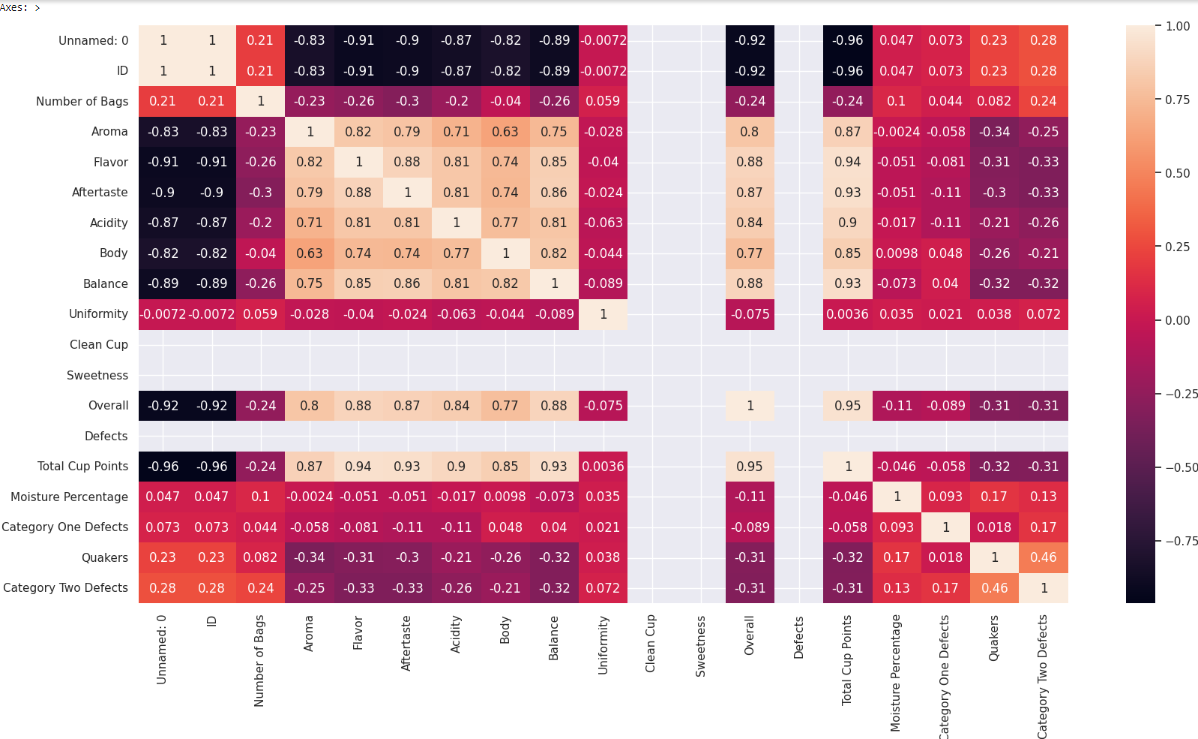
Here we analyze the relation between numerical values with respect to Total cup points



**Activity 5: Multivariate analysis**

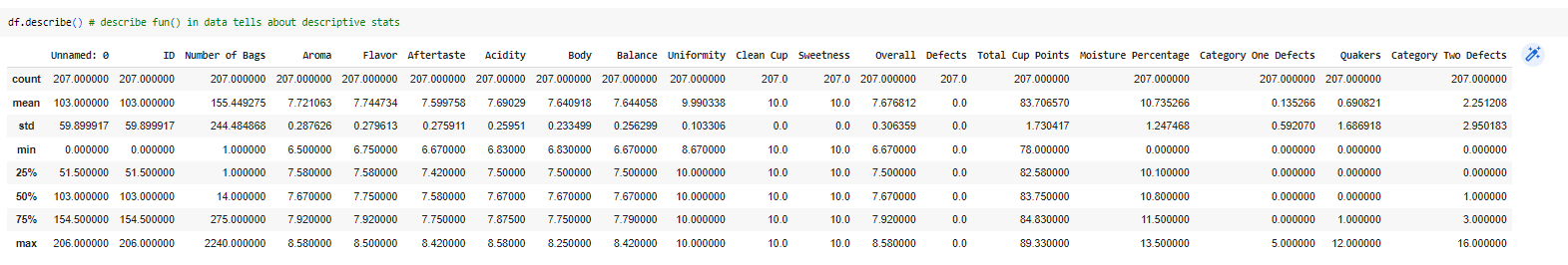
In simple words, multivariate analysis is to find the relation between multiple features. Here we have used heatmap from seaborn package.

* From the below image, we came to a conclusion that how numerical data is distributed and how they are and how much they are correlated each other.



**Activity 6: Descriptive analysis**

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.



# Milestone 3: Data Collection & Data Pre-processing

ML depends heavily on data, It is most crucial aspect that makes algorithm training possible. So this section allows you to download the required dataset.

**Activity 1: Download the dataset**

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

##### In this project we have used masterful-machines-precise-coffee-quality.csv data. This data is downloaded from kaggle.com. Please refer the link given below to download the dataset.

Link:https://www.kaggle.com/datasets/fatihb/coffee-quality-data-cqi

As we have understood how the data is collected lets pre-process the collected data.

The download data set is not suitable for training the machine learning model as it might have so much of randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

* Extracting important features from data
* Handling missing values
* Handling categorical data
* Handling outliers
* Scaling Techniques
* Splitting dataset into training and test set

Note: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

**Activity 2: Droping unimportant columns and Extracting important features.**

**droping cols**

The IDs, names of the companies, owner names, countries, certificates, and expiration date of beans may not be directly useful when predicting the quality of beans. Here's a more detailed explanation:

IDs: The unique identification numbers assigned to the beans or companies are typically arbitrary and do not contain any inherent information related to the quality of the beans. They are primarily used for identification and tracking purposes but do not provide direct insights into the quality characteristics.

Company and owner names: The names of the companies and owners associated with the beans are typically descriptive or organizational in nature. While they may provide information about the source or origin of the beans, they do not directly represent or indicate the quality attributes of the beans themselves.

Countries: The country of origin or production may be relevant in certain cases, as different countries may have specific reputations or known characteristics associated with their beans. However, the country alone is not sufficient to predict the quality of the beans accurately. Other factors, such as growing conditions, cultivation practices, and post-harvest processing, play significant roles in determining bean quality.

Certificates: Certifications obtained by the beans or companies, such as organic certifications or fair trade certifications, primarily indicate adherence to certain standards or practices. While these certifications may indirectly suggest certain quality aspects, they do not provide detailed information about the specific quality attributes of the beans themselves.

Expiration date: The expiration date indicates the recommended period during which the beans are expected to retain their optimal quality. While the expiration date is relevant for consumer safety and product freshness, it does not provide direct information about the intrinsic quality characteristics of the beans.

When predicting the quality of beans, more relevant features would include factors such as bean size, color, aroma, flavor profile, moisture content, defects, processing method, altitude of cultivation, bean variety, and cupping scores. These attributes directly relate to the sensory and physical properties of the beans, which are critical determinants of their overall quality.

Therefore, when building a predictive model for bean quality, it is advisable to focus on features that capture the intrinsic characteristics of the beans themselves, rather than factors that are indirectly or tangentially related, such as IDs, company names, owner names, countries, certificates, and expiration dates.

**drop one unique**

The columns "Status," "Clean Cup," "Sweetness," and "Defects" are being dropped from the DataFrame using the **df.drop()** function. This decision is based on the fact that these columns contain only one unique value each. Since these columns do not exhibit any variation or provide meaningful information for predicting the quality of the beans, they are deemed irrelevant for the quality prediction task.

By removing these columns from the DataFrame, we can streamline the dataset and focus on the features that are more likely to contribute to the quality prediction. It is assumed that the remaining columns in the DataFrame contain relevant information that can be utilized in the quality prediction model.

**Activity 3: Extracting important labels from Variety column and Transforming them.**

窗体顶端



窗体底端

The provided code defines a function called **segment\_var(Variety)** and applies it to the 'Variety' column of a DataFrame using the **.apply()** method. The purpose is to extract important labels from the 'Variety' column and convert the remaining values, which occur only once, into the category 'Other'.

The function takes the value of the 'Variety' column as input. It checks various conditions using **if** and **elif** statements to categorize the values into specific groups. If the value is missing (**pd.isnull(Variety)**), it is labeled as 'Other'. If a specific term is found in the value, such as 'Caturra', 'Gesha', 'Typica', etc., the function assigns the corresponding label. If none of the conditions match, the value is also labeled as 'Other'.

The updated labels are then applied to the 'Variety' column of the DataFrame using **df['Variety'].apply(segment\_var)**. This modifies the column in-place, replacing the original values with the segmented labels.

Overall, the purpose of this code snippet is to extract important labels from the 'Variety' column while converting less frequently occurring values into the category 'Other' for better categorization and analysis.

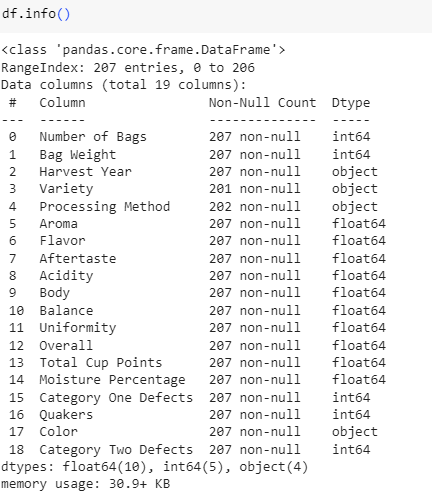
**Activity 4:Converting “Bag Weight” to numerical by using String slicing.**

****

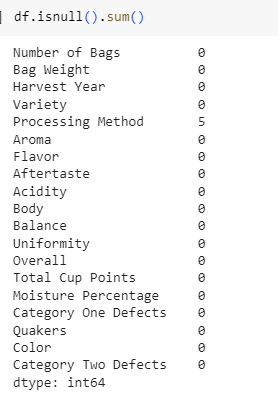
Removing kg string from the Bag of weights column and converting into numerical.

**Activity 5: Checking for null values**

* Let’s find the shape of our dataset first by df.shape method . To find the data type, df.info() function is used.

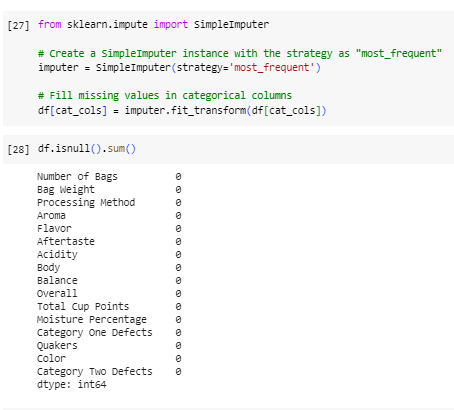
****

* For checking the null values, df.isnull() function is used. To sum those null values we use .sum() function to it. From the below image we found that there are some null values present in our dataset. So we have to handle the missing values.



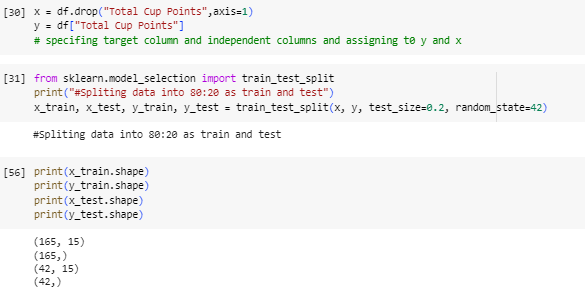
**Activity 6: Handling Missing Values**

* To handle the missing values we using simpleimputer function to fill null values by most frequent values.
* Now there is no any null values in our dataset.



**Activity 7: Splitting data into train and test**

Now let’s split the Dataset into train and test sets. For splitting training and testing data we are using train\_test\_split() function from sklearn. As parameters, we are passing features, y\_classlabel, train\_size, random\_state.



**Activity 8: Handling Categorical Data**

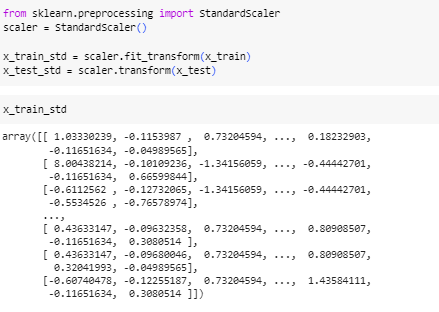
In our data we have 4 categorical columns “ Harvesting year”, “Variety”, “processing method” and “color” now will convert them into numerical.

For this imputed LabelEncoder from sklearn and applied encoding



**Activity 9: Scaling**

As we have different scales of data, in order to make good predictions using a model, we are currently in the process of scaling our data. First split the dataset into Features and y\_label. For scaling we are imported StandardScaler from sklearn then applied scaling to features\_



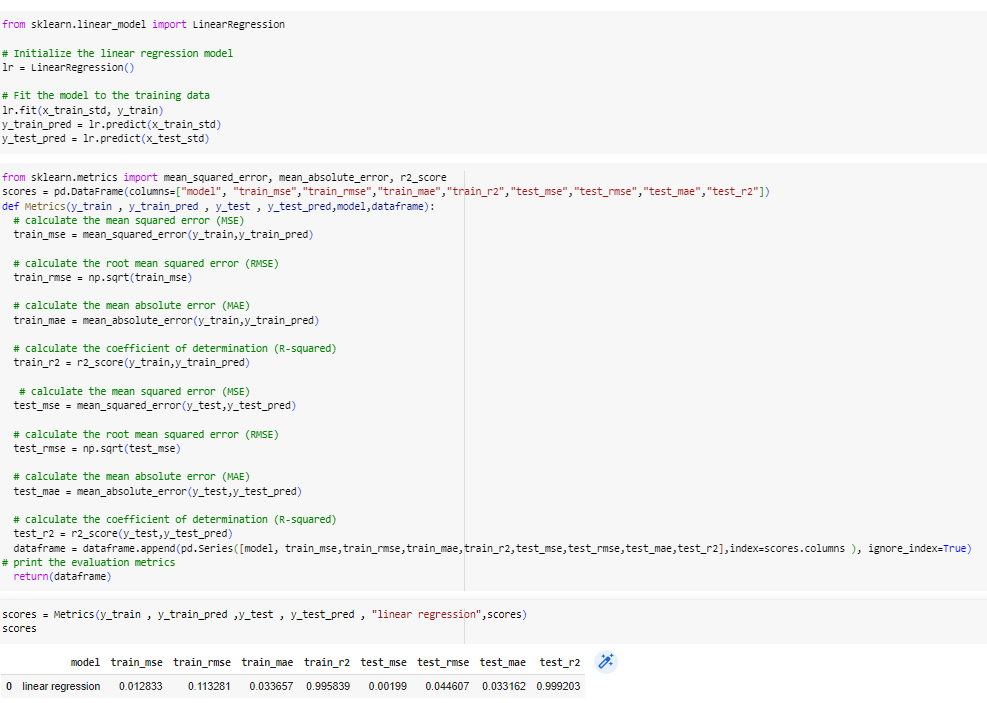
Now our data is in same scale for good model prediction.

# Milestone 4: Model Building

Now our data is cleaned and it’s time to build the model. We can train our data on different algorithms. For this project we are applying regression algorithms. The best model is saved based on its performance.

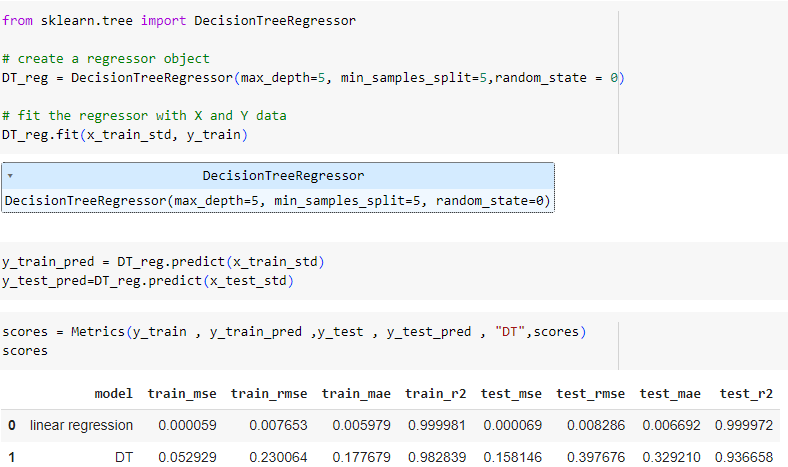
**Activity 1: Linear Regression**

A function named Linear Regression is created and train and test data are passed as the parameters. Inside the function, sklearn.linear\_model algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, wrote a function to get R2 score, RMSE, MSE, MAE to chose best accurate model .



**Activity 2: Decision Tree**

A function named Decision tree is created and train and test data are passed as the parameters. Inside the function, Decision tree algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model using scores are done.



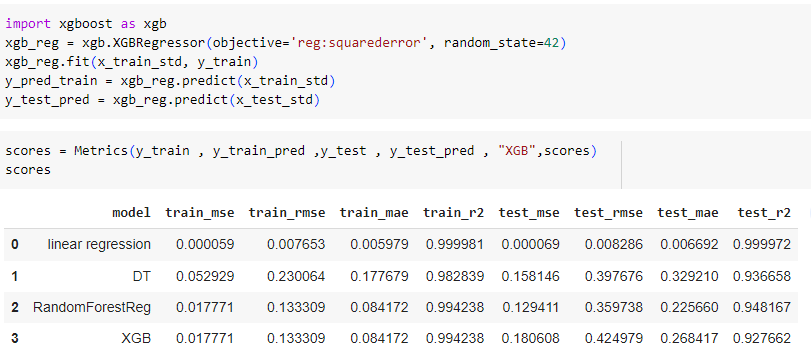
**Activity 3: Random forest model**

A function named Random\_Forest is created and train and test data are passed as the parameters. Inside the function, RandomForestRegressor algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model with scores are done.



**Activity 3: Xgboost model**

A function named xgboost is created and train and test data are passed as the parameters. Inside the function, XBregressor algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, report is done.



The Linear Regression is giving the best accuracy, best scores than other models.

# Milestone 5: Performance Testing & Hyperparameter Tuning

## Activity 1: Testing model with multiple evaluation metrics

Multiple evaluation metrics means evaluating the model's performance on a test set using different performance measures. This can provide a more comprehensive understanding of the model's strengths and weaknesses. We are using evaluation metrics for

**R2 score :** The R2 score, also known as the coefficient of determination, is a statistical measure used to evaluate the performance of a regression model. It provides an indication of how well the model fits the observed data.

Certainly! The R2 score is calculated using the following formula:

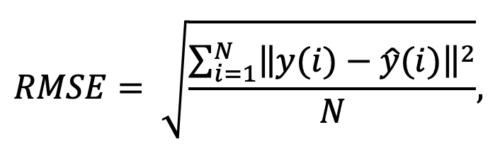
R2 = 1 - (SSR / SST)

To break down the formula further:

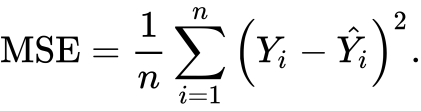
SSR = Σ(y\_actual - y\_predicted)^2, where Σ represents the sum over all data points.

SST = Σ(y\_actual - y\_mean)^2

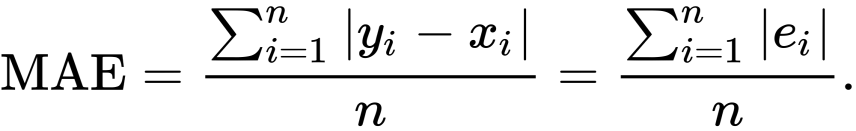
**RMSE** : Root mean square error or root mean square deviation is one of the most commonly used measures for evaluating the quality of predictions. It shows how far predictions fall from measured true values using Euclidean distance.



**MSE :** In [statistics](https://en.wikipedia.org/wiki/Statistics" \o "Statistics), the **mean squared error** (**MSE**)[[1]](https://en.wikipedia.org/wiki/Mean_squared_error" \l "cite_note-:1-1) or **mean squared deviation** (**MSD**) of an [estimator](https://en.wikipedia.org/wiki/Estimator" \o "Estimator) (of a procedure for estimating an unobserved quantity) measures the [average](https://en.wikipedia.org/wiki/Expected_value" \o "Expected value) of the squares of the [errors](https://en.wikipedia.org/wiki/Error_(statistics)" \o "Error (statistics))—that is, the average squared difference between the estimated values and the actual value.

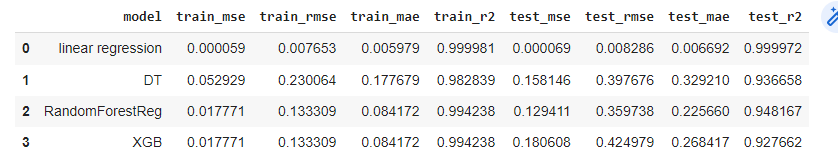


**MAE :** In [statistics](https://en.wikipedia.org/wiki/Statistics" \o "Statistics), **mean absolute error** (**MAE**) is a measure of [errors](https://en.wikipedia.org/wiki/Error_(statistics)" \o "Error (statistics)) between paired observations expressing the same phenomenon. Examples of *Y* versus *X* include comparisons of predicted versus observed, subsequent time versus initial time, and one technique of measurement versus an alternative technique of measurement. MAE is calculated as the **sum of absolute errors** divided by the sample size:



## Activity 2: Compare the model

For comparing the above five models, the comparison function is defined.



After calling the function, the results of models are displayed as output. From the four models Random Forest algorithm is performing well.

## Activity 3: Comparing model scores before & after applying hyperparameter tuning (Hyperparameter tuning is optional. For this project it is not required.)

Evaluating performance of the model From sklearn, cross\_val\_score is used to evaluate

the score of the model. On the parameters, we have given rf (model name), x, y, cv (as 5folds). Our model is performing well.

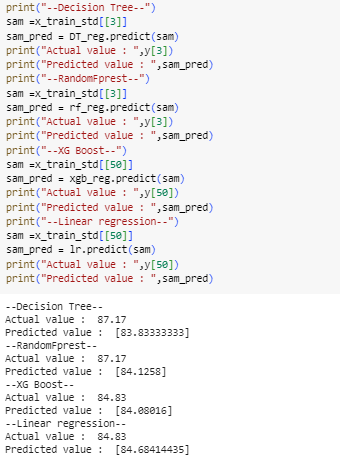
**Note:** To understand hyperperameter tuning, refer to this : <HTTP://www.javatpoint.com/hyperparameters-in-machine-learning>

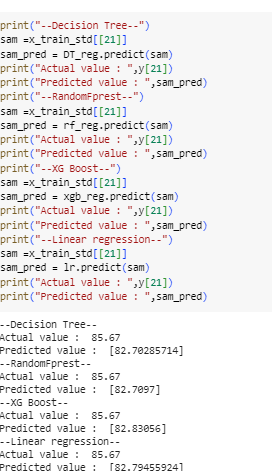
Your model performing well with out hyperperameter tuning so we skip tuning.

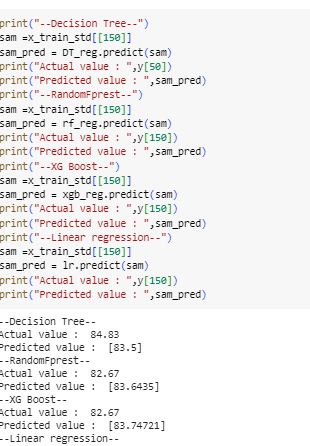
**Activity 4 : Predicting for some samples.**

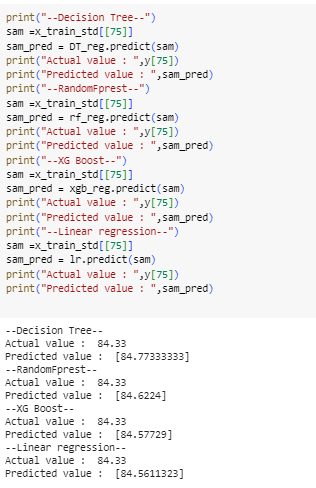
I took some samples from train and test.

I predicted for that samples.

****

****

****

****

**Activity 5: Saving the model**

Finally we chosen the best model now saving that model and standard scaler to transform values for predictions.



# Milestone 6: Application Building

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

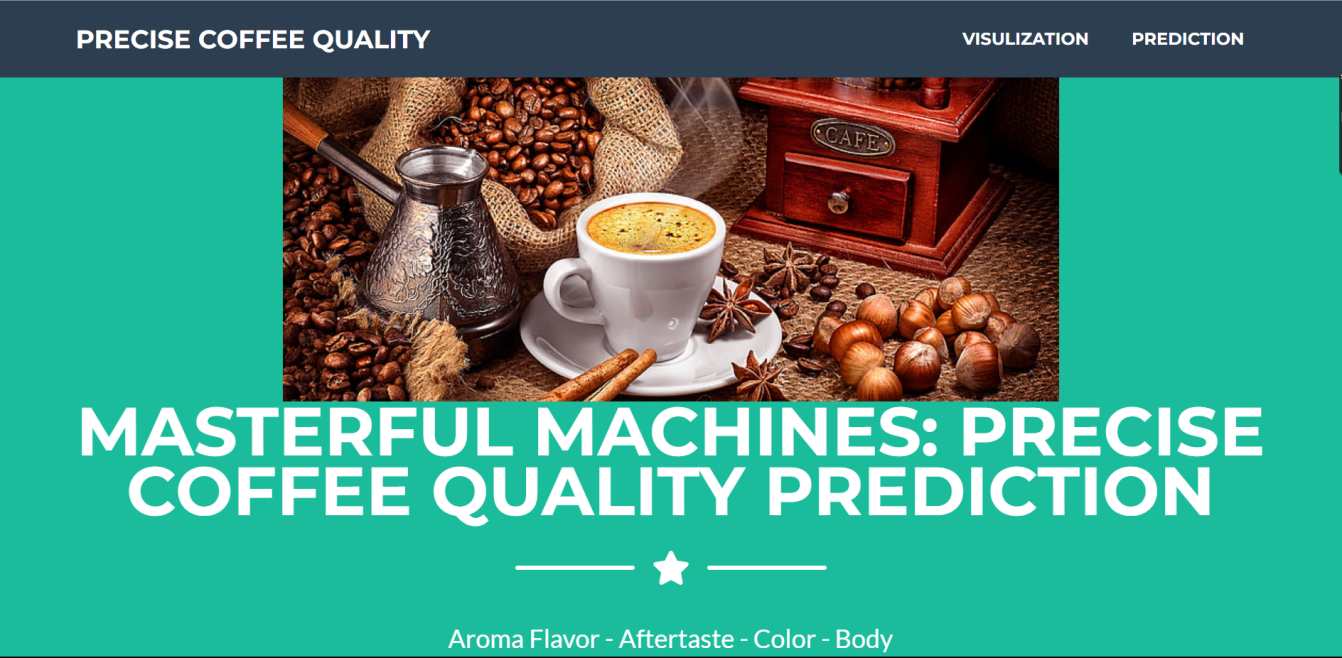
* Building HTML Pages
* Building serverside script

**Activity1: Building Html Pages:**

For this project create three HTML files namely

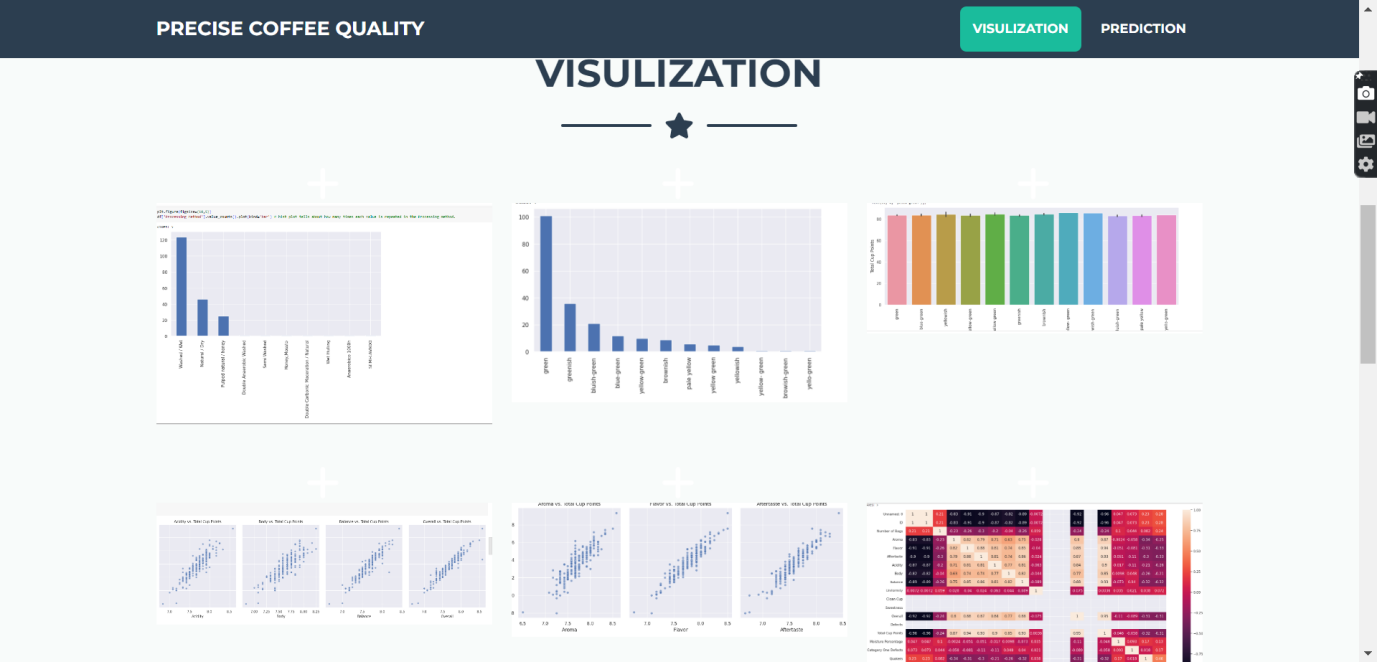
* index.html and save them in templates folder.

Let’s see how our index.html page looks like:



Now when you click on visualization button from top right corner you will get redirected to Visualizations.

Lets look how our file looks like:

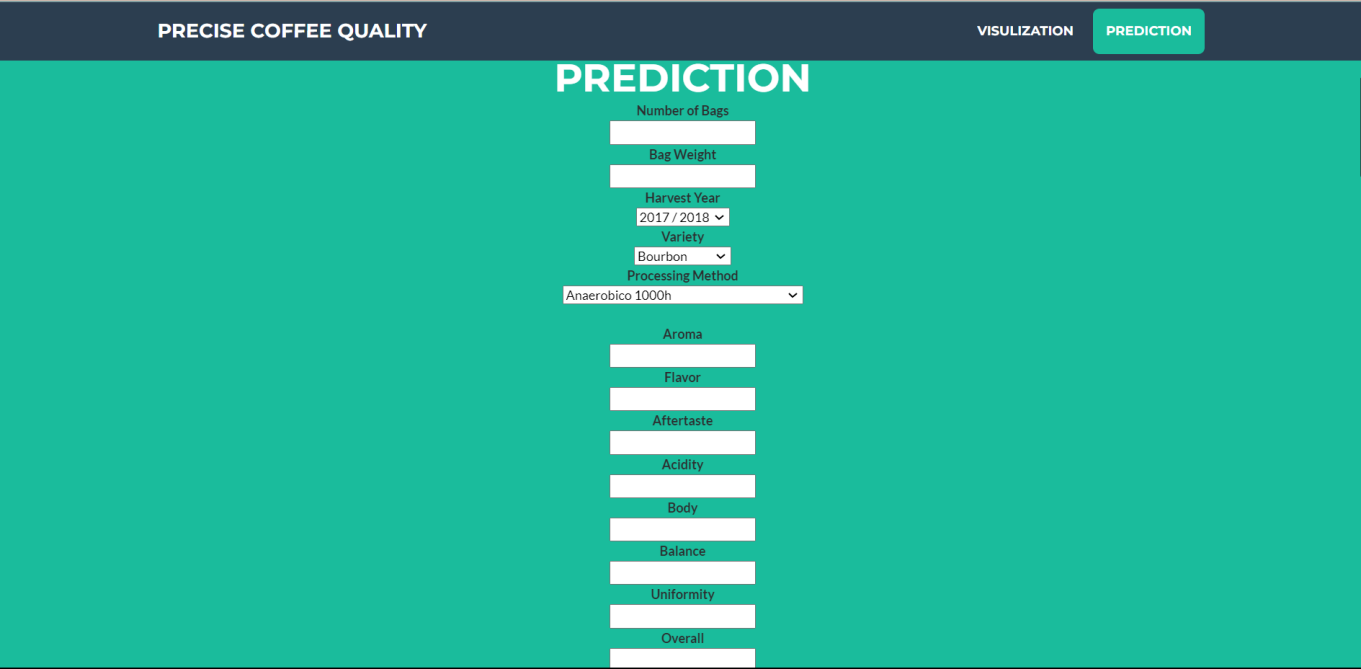


When we click on Graph and opens details of graph.

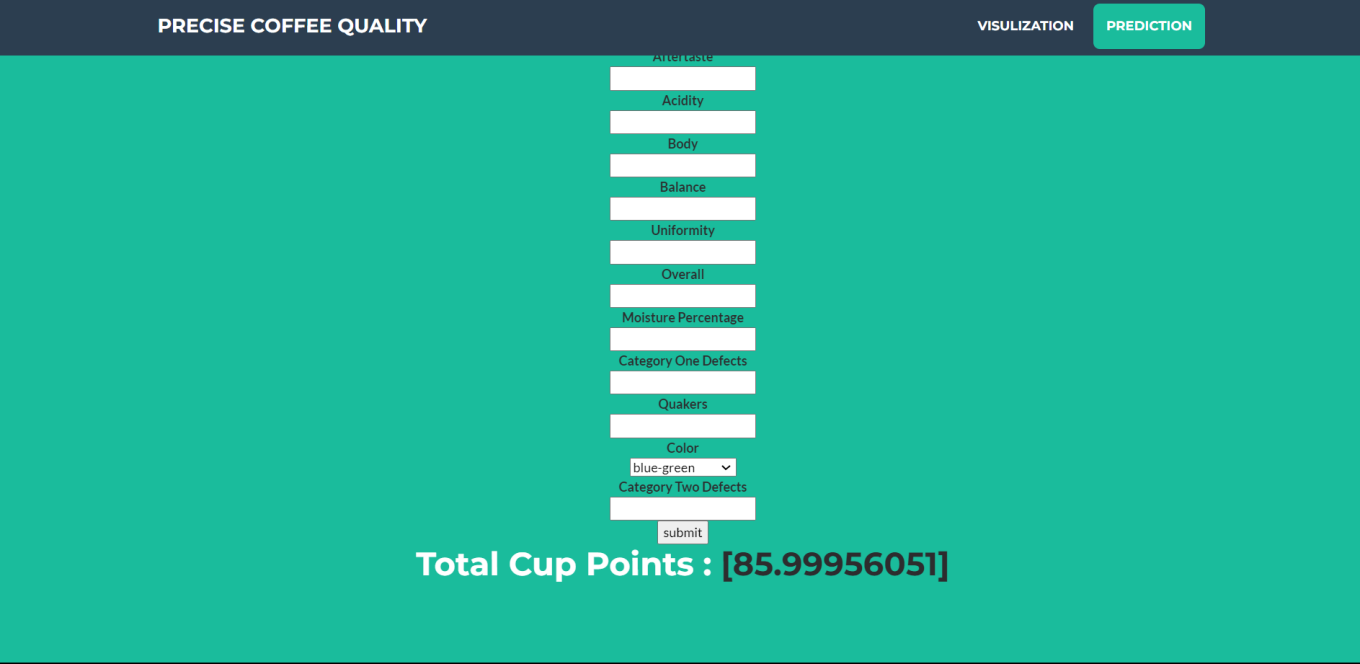


Now when you click on predict button you will redirect to predictions.

Lets look how our file looks like:

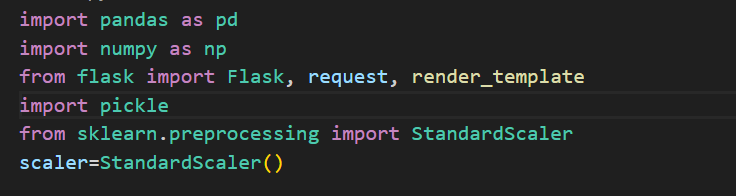


Will try with different numbers and then click on predict button and number must be in integer .

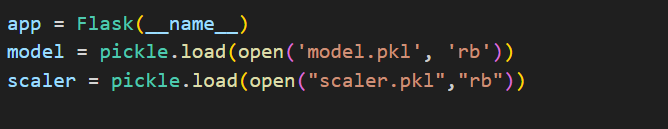


**Activity 2: Build Python code:**

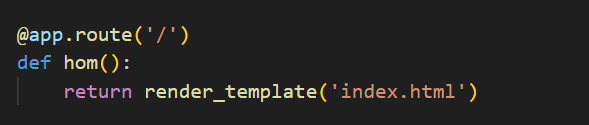
Import the libraries



Load the saved model. Importing flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (\_\_name\_\_) as argument.



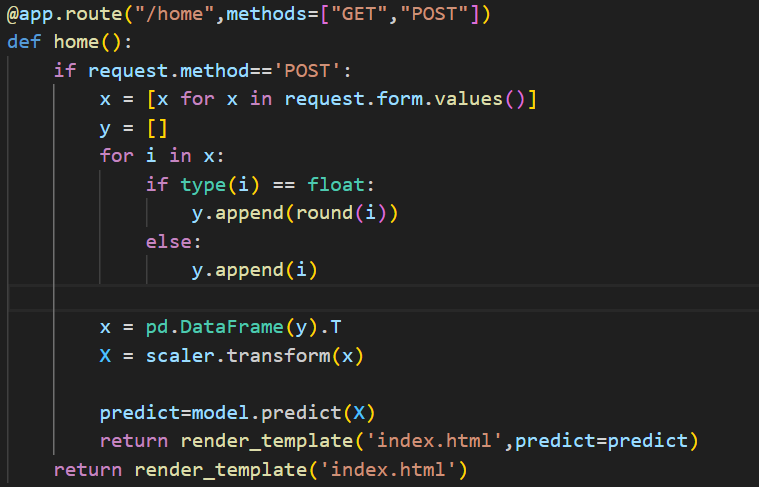
Render HTML page:



Here we will be using declared constructor to route to the HTML page which we have created earlier.

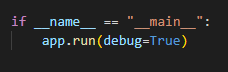
In the above example, ‘/’ URL is bound with index.html function. Hence, when the index page of the web server is opened in browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:



Here we are routing our app to output() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will rendered to the text that we have mentioned in the output.html page earlier.

Main Function:



**Activity 3: Run the application**

* Open anaconda prompt from the start menu
* Navigate to the folder where your python script is.
* Now type “python app.py” command
* Navigate to the localhost where you can view your web page.
* Click on the inspect button from the top right corner, enter the inputs, click on the predict button, and see the result/prediction on the web.

