

Text  Description automatically generated with low confidence

Synchronized Timepiece Pricing: A Machine Learning Approach to Smart Watch Valuation

# Project Description:

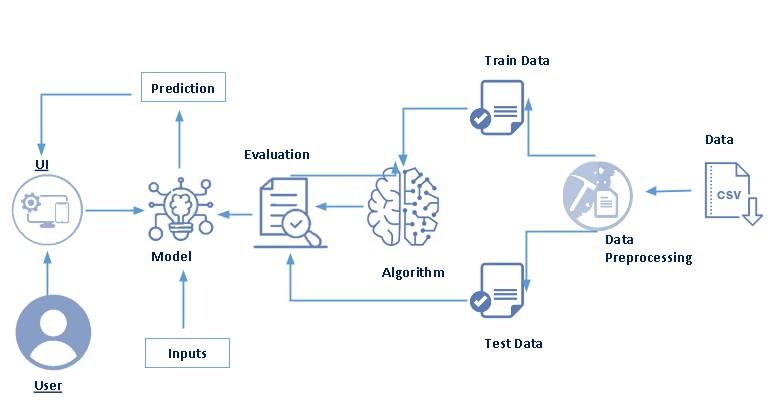
**The Smartwatch Price Dataset contains information about the features and prices of popular smartwatch models from various brands. The dataset includes columns such as Brand, Model, Operating System, Connectivity, Price (USD), Display Type, Display Size (inches), Resolution, Water Resistance (meters), Battery Life (days), Heart Rate Monitor, GPS, and NFC.  
Columns  
The dataset provides a comprehensive overview of the different smartwatches available in the market and can be used for various purposes such as price comparison, feature analysis, and market research. The data is gathered from various sources such as official brand websites, online retailers, and tech blogs. This dataset can be useful for individuals or businesses interested in the smartwatch industry, as well as researchers and data analysts**.

To identify smart watch price accurately, we will use machine learning regression algorithms such as Linear regression ,Decision tree, Random forest, and XGBoost. We will train and test the data with these algorithms and evaluate their performance using R2 score, Root Mean Square, MAE , and MSE. The best-performing model will be selected and saved in pkl format.

To make the Smart Watch price prediction model available to users, we will integrate it with a Flask web application. Users will be able to input an account's features, and the web application to predict price using specification of watch. We will also deploy the web application.

In conclusion, identifying prices of watches on machine learning platforms is crucial for preventing them from influencing content and user engagement. Our project's goal is to create an accurate prediction model and make it available to users through a Flask web application.

# 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:

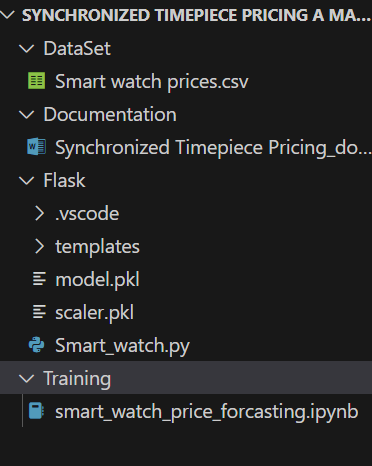
* User interacts with the UI to enter the input.
* Entered input is analyzed by the model which is integrated.
* Once model analyses the input the prediction is showcased on the UI

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 water\_quality.py for scripting.
* water\_quality.joblib 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 price of Smart Watches on a machine learning platform. To achieve this goal, we will analyze various features such as 'Brand', 'Model', 'Operating System', 'Connectivity', 'Display Type', 'Display Size (inches)', 'Resolution', 'Water Resistance (meters)', 'Battery Life (days)', 'Heart Rate Monitor', 'GPS', 'NFC', 'Price (USD)'. 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 prices of watches 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 prices. This is important because inaccurate predictions could lead to the spread of misinformation on the platform when estimating price.

## ● Speed: The solution should be able to process smart watches data quickly. This is important because timely responses to predict price can help prevent the spread of misleading prices to customers and reduce the impact of frauds in society.

## ● Flexibility: The solution should be flexible enough to adapt to changes in price of watches patterns and the platform environment. This is important because prices can evolve and change over time.

## ● Integration: The solution should be easily integrated into the existing platform infrastructure and processes. This will ensure that the solution can be used efficiently and effectively by platform administrators and stakeholders.

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

## A literature survey for a price prediction analysis project would involve researching and reviewing existing studies, articles, and other publications on the topic of smart watch analysis. One such study proposed an approach for identifying prices on a machine learning platform using various features such as 'Brand', 'Model', 'Operating System', 'Connectivity', 'Display Type', 'Display Size (inches)', 'Resolution', 'Water Resistance (meters)', 'Battery Life (days)', 'Heart Rate Monitor', 'GPS', 'NFC', 'Price (USD)'. The approach was evaluated on a dataset of real-world prices and achieved an accurate R2 of 0.8 and RMSE loss of 30. The study demonstrated the effectiveness of machine learning for price prediction on a machine learning platform. The literature survey would also examine other studies that have explored different techniques and methods for price analysis, as well as the challenges and limitations associated with price estimation. By conducting a comprehensive literature survey, the smart watch prices 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 price 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, price analysis can identify exact approximate prices that may be influencing content. This can help prevent the spread of false information and ensure that prices are genuine.

**Business Model/Impact :-** The business impact of smart watch price prediction analysis can improve the effectiveness of machine learning platforms. By analyzing various features such as 'Brand', 'Model', 'Operating System', 'Connectivity', 'Display Type', 'Display Size (inches)', 'Resolution', 'Water Resistance (meters)', 'Battery Life (days)', 'Heart Rate Monitor', 'GPS', 'NFC', 'Price (USD)', it can identify potential prices that may be misusing the platform. This can help machine learning platform providers to identify and address watch prices earlier, reducing the potential negative impact on user experience. Additionally, effective predicting prices analysis can help businesses and industries to maintain the integrity of their data and prevent fraudulent activities.

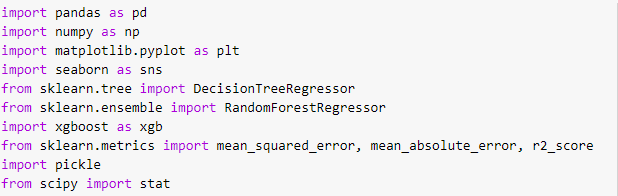
# Milestone 2: Visualizing and analysing the data

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**

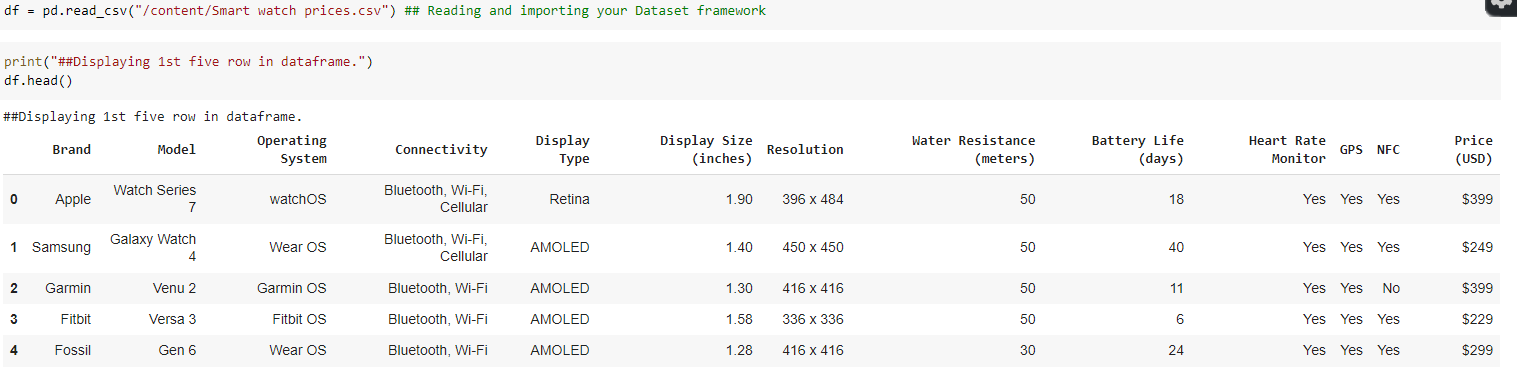
Import the necessary libraries as shown in the image. (optional) Here we have used visualization.



**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.

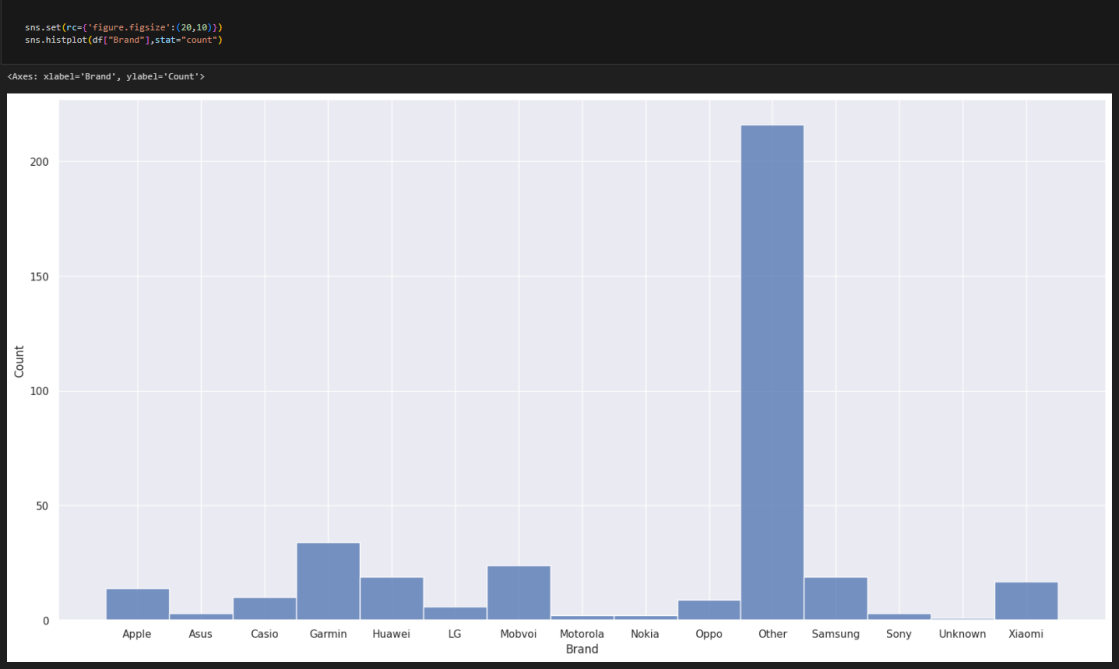
In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of csv file.



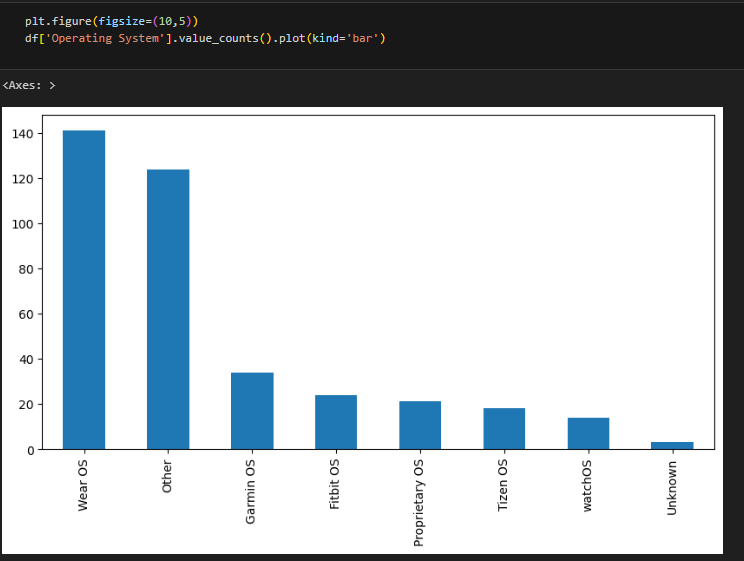
**Activity 3: Univariate analysis**

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

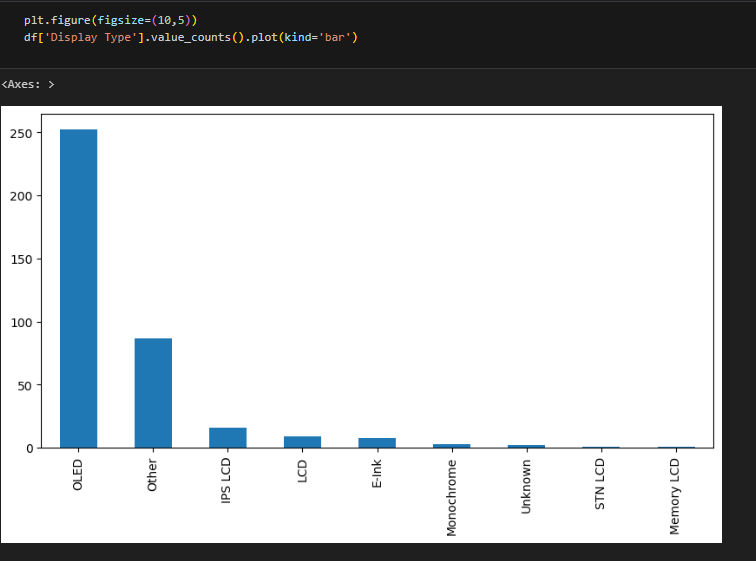
* Seaborn package provides a wonderful function hi9stplot. With the help of histplot, we can find the count of the brands from the feature.



* From the above we can understand how types of brands with count.



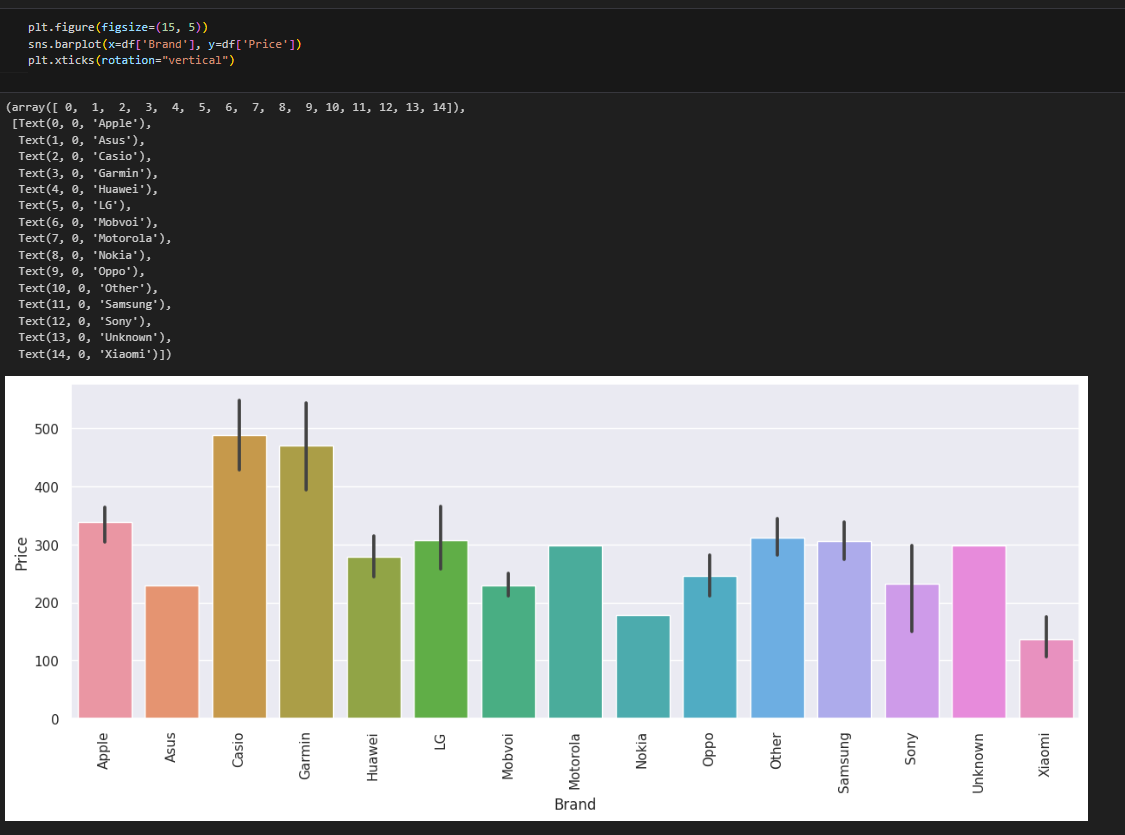
* From the above we can understand types of operating system and count.



* From the above plot we came to know how many types of display with count.

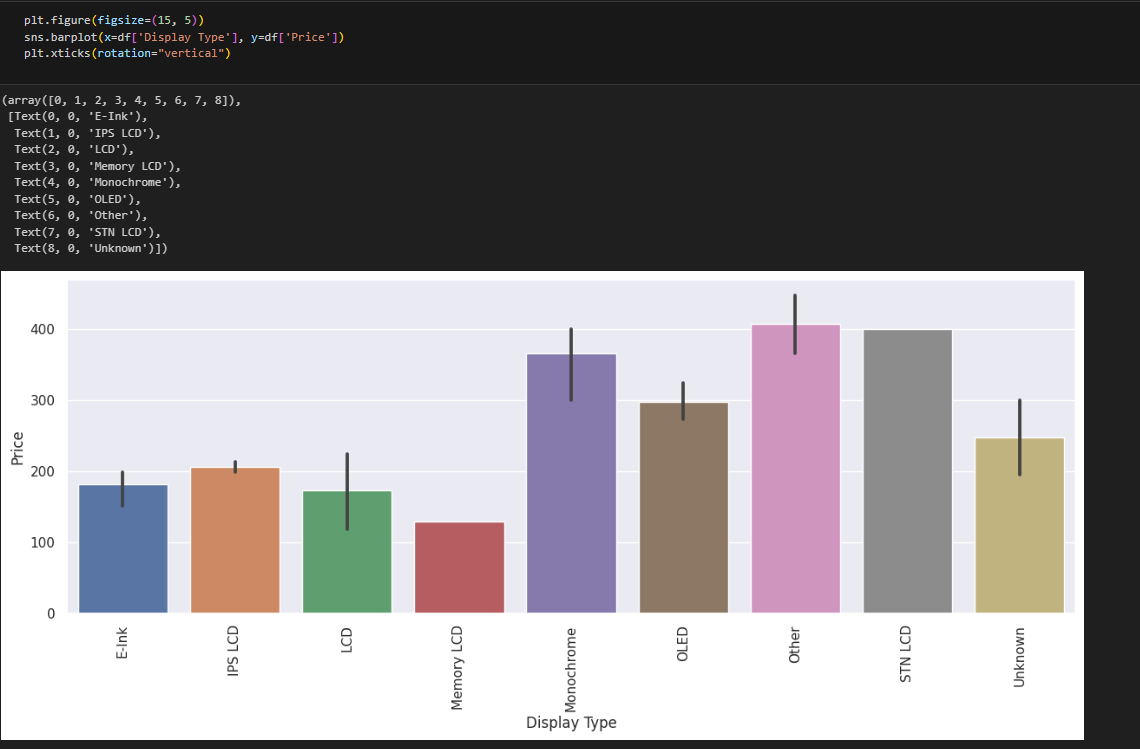
**Activity 4: Bivariate analysis**

To find the relation between two features we use bivariate analysis. Here we are visualizing the relationship between brand and price.



* This plot tells about mean price of each brand.

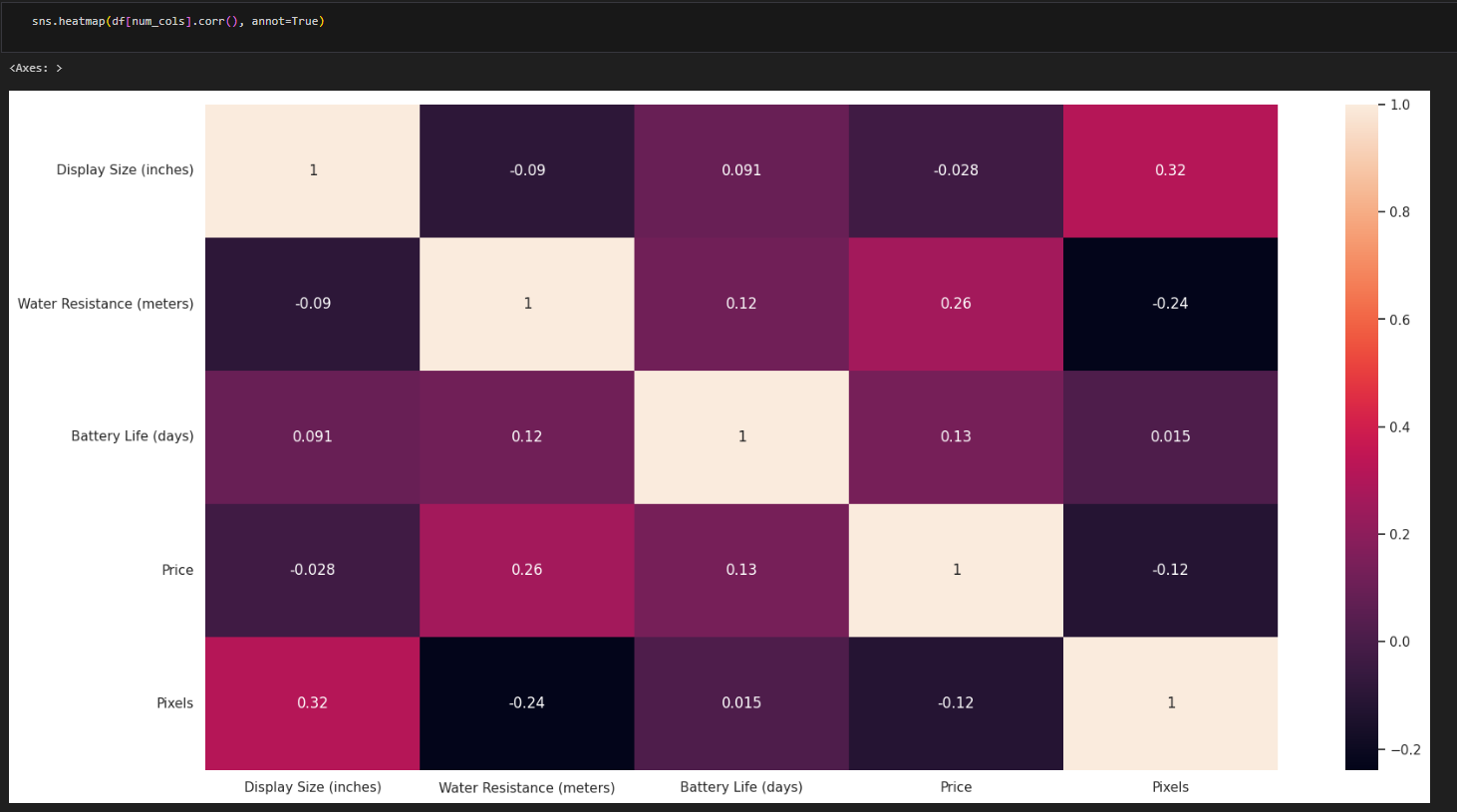
Here we are visualizing the relationship between Disply type and price.



**Activity 5: Multivariate analysis**

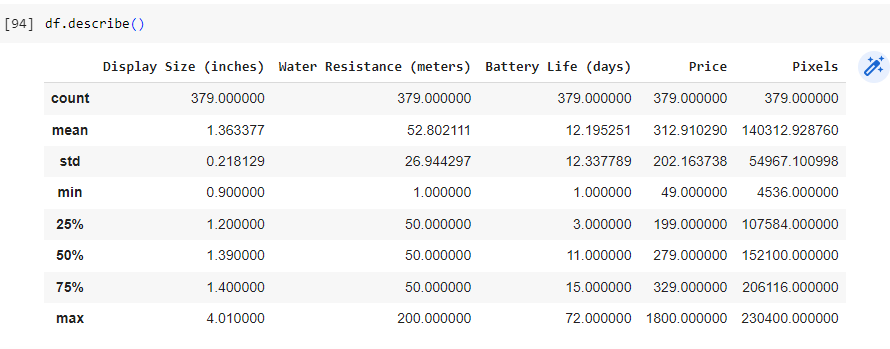
In simple words, multivariate analysis is to find the relation between multiple features. Here we have used heatmap from seaborn package.

* 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 Smart watch prices.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/rkiattisak/smart-watch-prices>

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

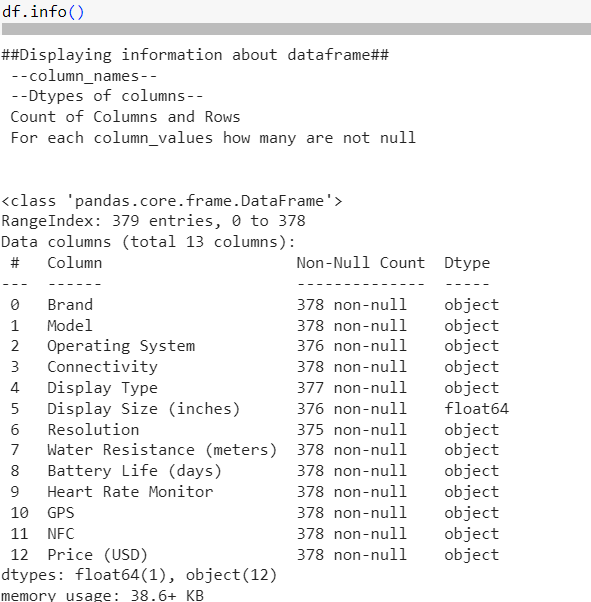
The download data set is not suitable for training the machine learning model as it might have so much of randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

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

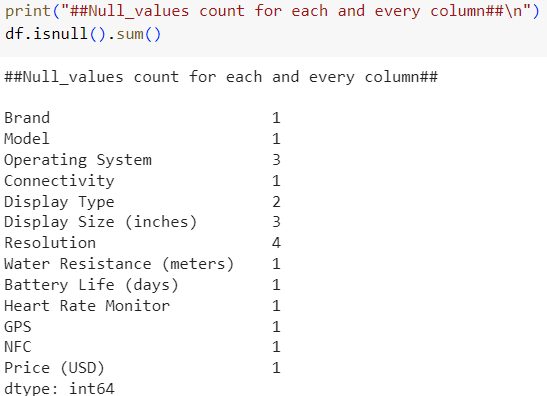
Note: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

**Activity 2: Checking for null values**

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

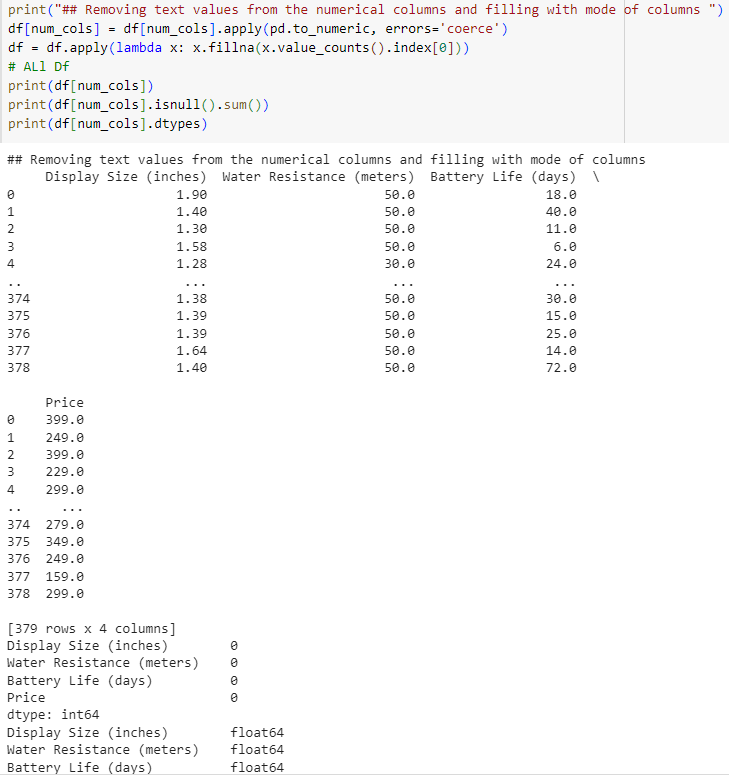
****

* 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.



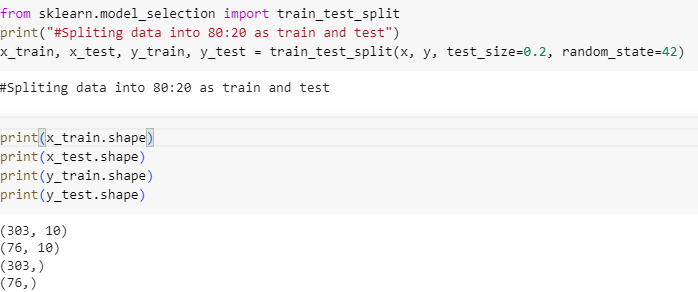
**Activity 3: Handling Missing Values**

* To handle the missing values, Text in numerical data and filled with mode.
* Now there is no any null values in our dataset.



**Activity 4: Splitting data into train and test**

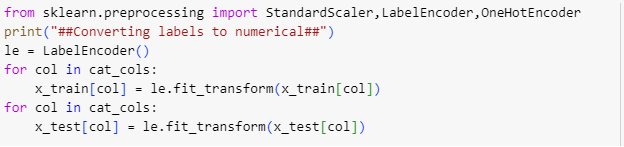
Now let’s split the Dataset into train and test sets. For splitting training and testing data we are using train\_test\_split() function from sklearn. As parameters, we are passing features, y\_classlabel, train\_size, random\_state.



**Activity 5: Handling Categorical Data**

In our data we have 3 categorical columns those are Brand, Operating System, Connectivity, Display Type, GPS, NFC now will convert them into numerical.

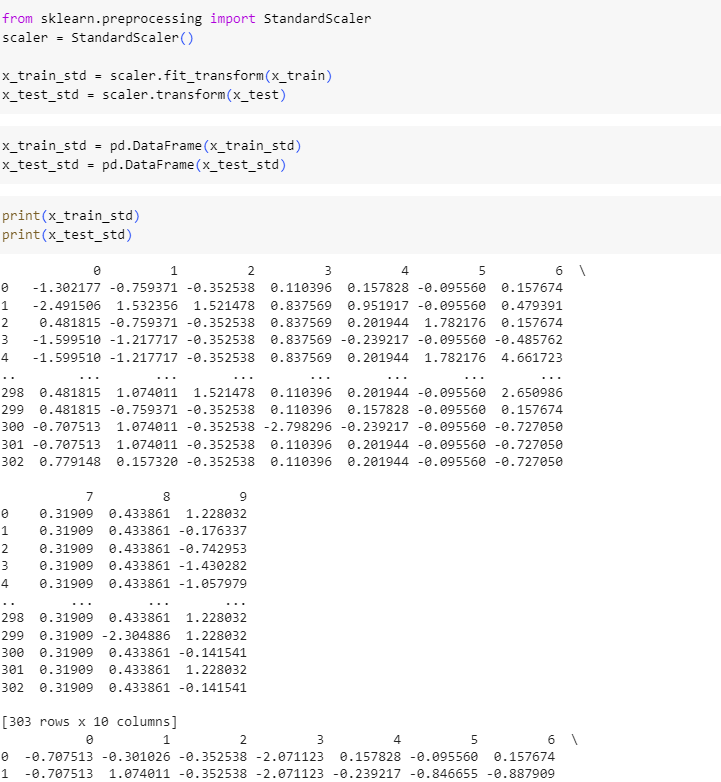
For this I imputed LabelEncoder from sklearn and applied encoding



There is no any outliers in our data so we can skip the outliers handling

**Activity 6: 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\_



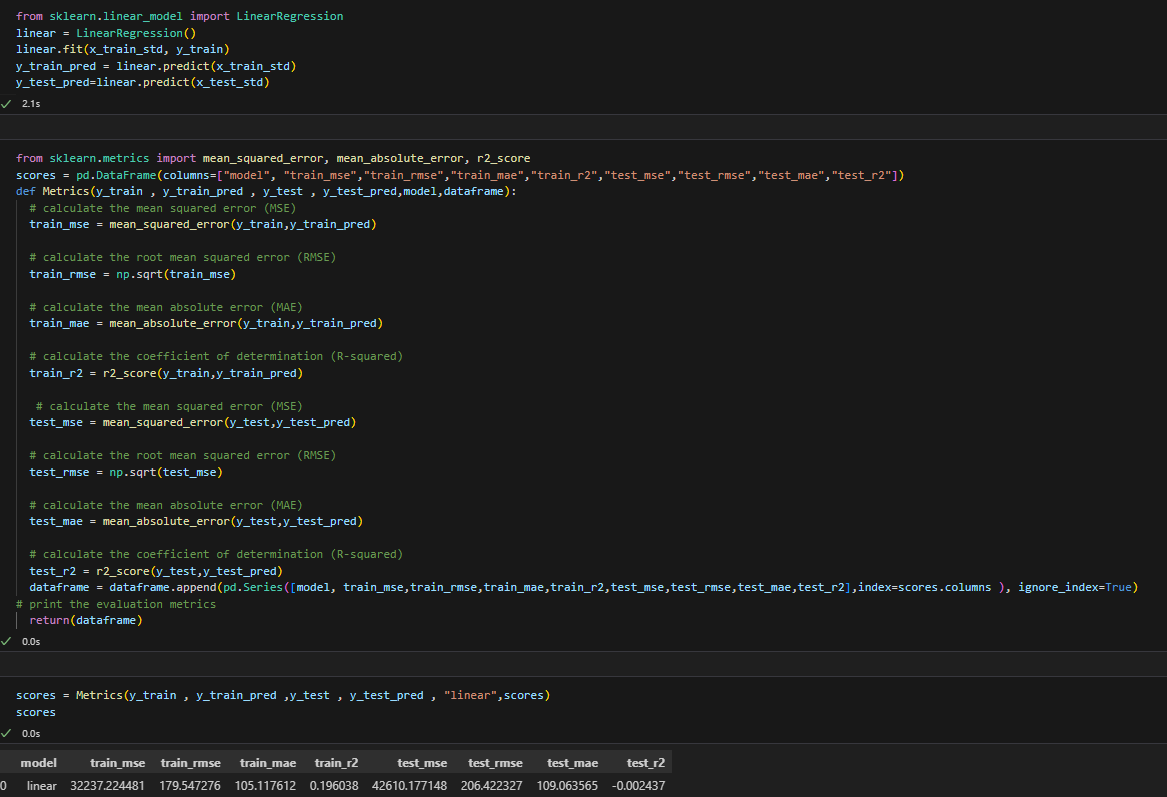
Now our data is in same scale for good model prediction.

# 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.

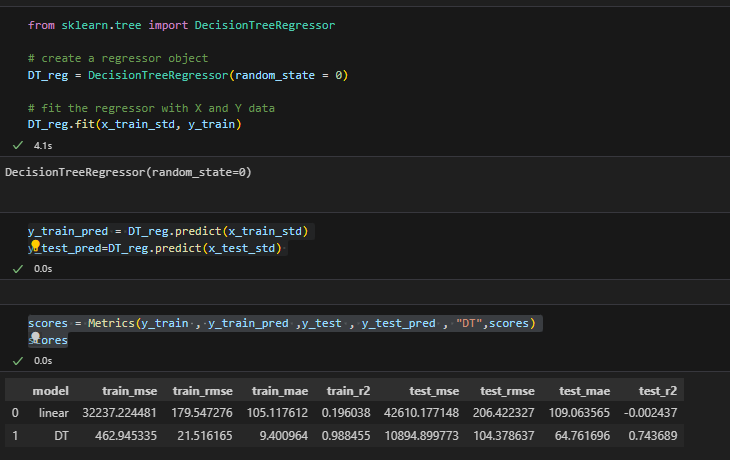
**Activity 1: Linear Regression**

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 .



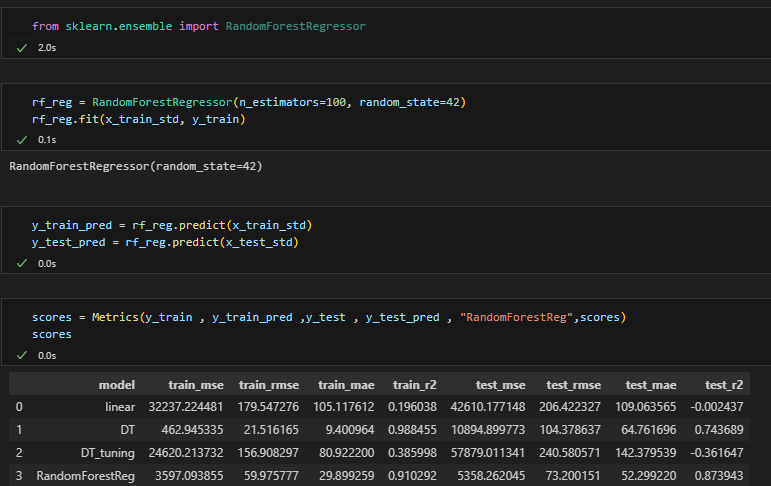
**Activity 2: Decision Tree**

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.



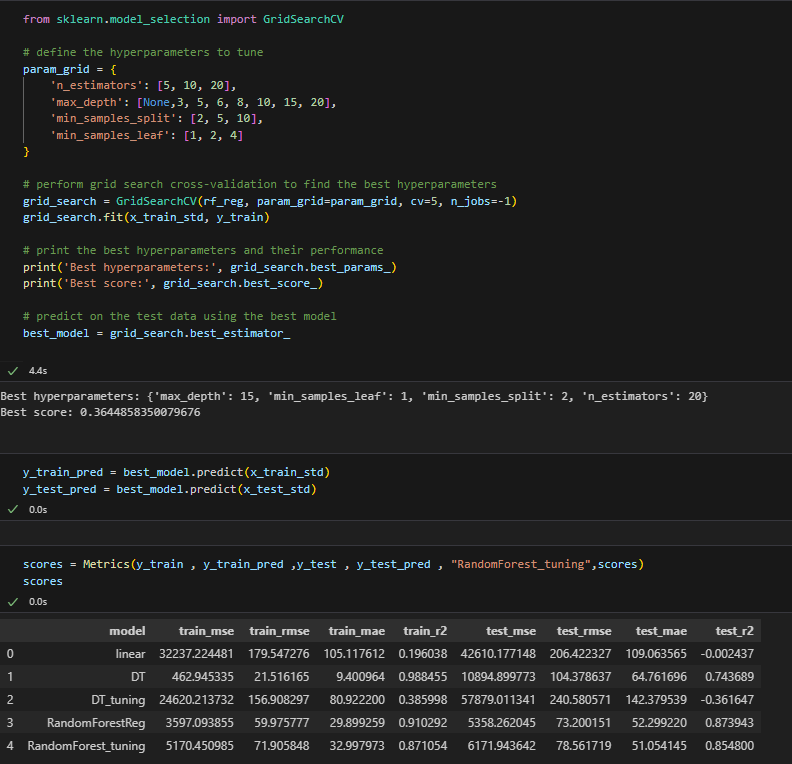
**Activity 3: Random forest model**

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.



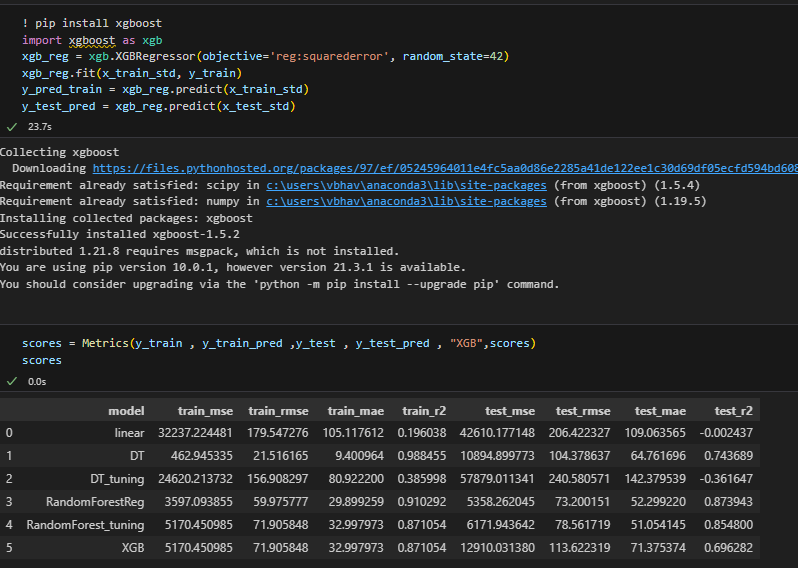
**Activity 4:Random Forest with Hyperperameter Tuning**

* Hyperperameter tuning using gridsearchcv.



**Activity 4: 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 Random Forest 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:

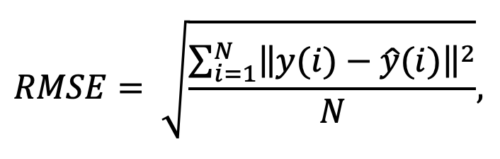
R2 = 1 - (SSR / SST)

To break down the formula further:

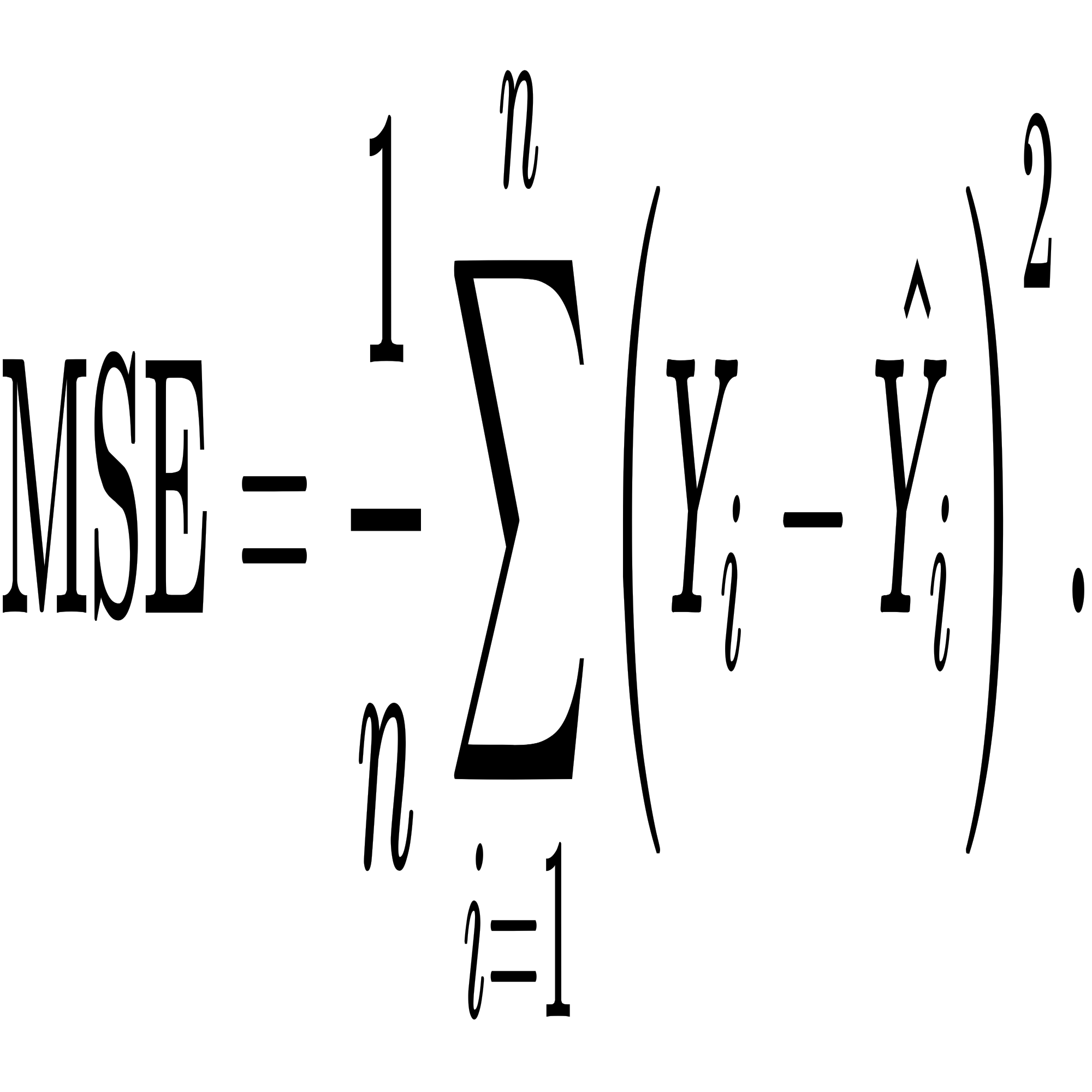
SSR = Σ(y\_actual - y\_predicted)^2, where Σ represents the sum over all data points.

SST = Σ(y\_actual - y\_mean)^2

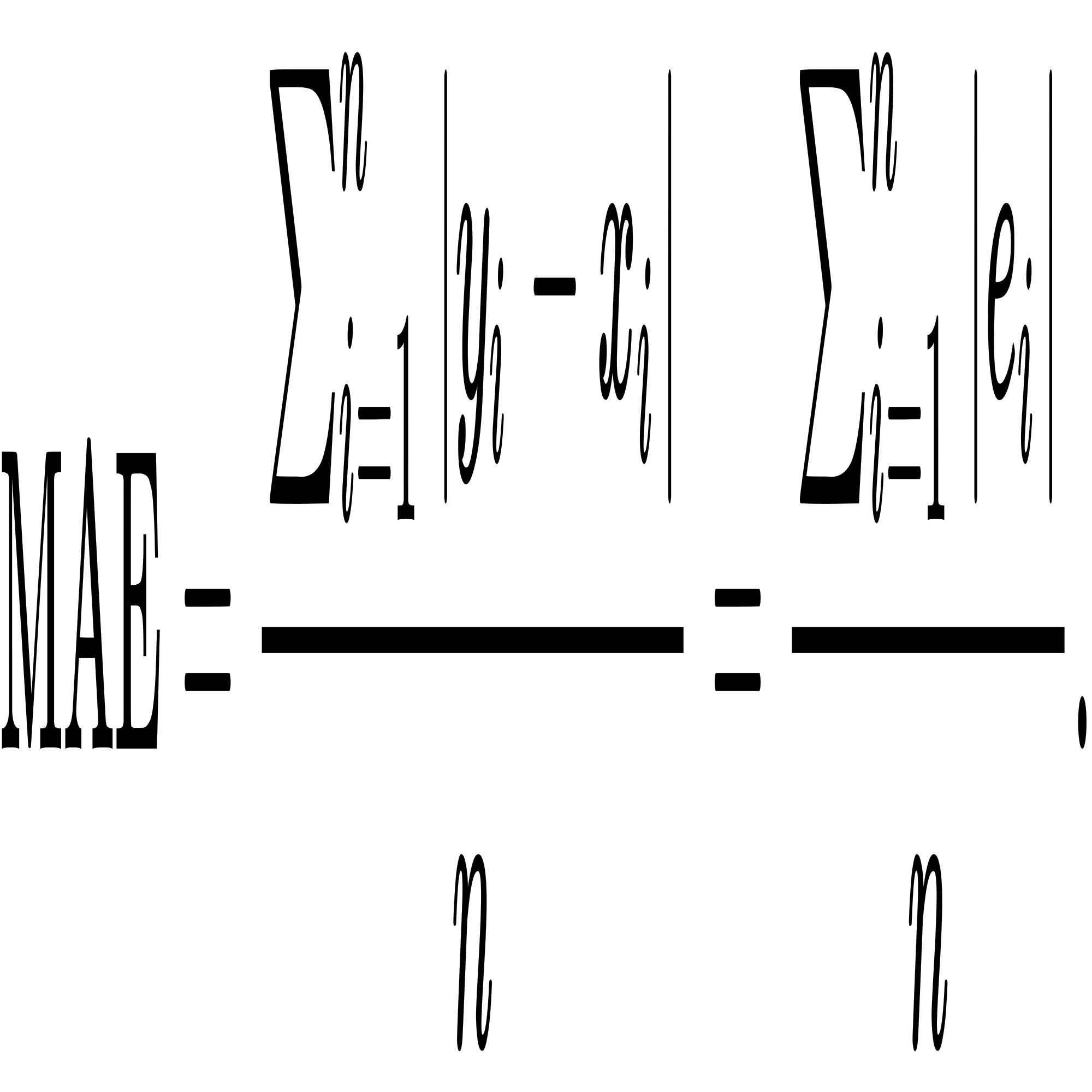
**RMSE** : Root mean square error or root mean square deviation is one of the most commonly used measures for evaluating the quality of predictions. It shows how far predictions fall from measured true values using Euclidean distance.



**MSE :** In statistics, the **mean squared error** (**MSE**) or **mean squared deviation** (**MSD**) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors that is, the average squared difference between the estimated values and the actual value.

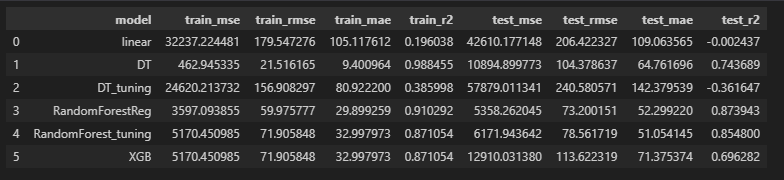


**MAE :** In statistics, **mean absolute error** (**MAE**) is a measure of errors between paired observations expressing the same phenomenon. Examples of *Y* versus *X* include comparisons of predicted versus observed, subsequent time versus initial time, and one technique of measurement versus an alternative 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.)

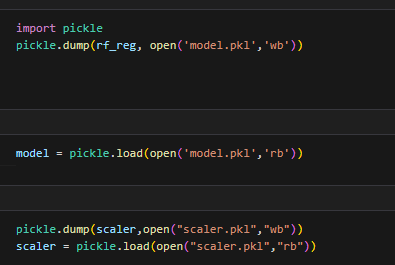
Evaluating performance of the model From sklearn, cross\_val\_score is used to evaluate

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

**Activity 4: 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

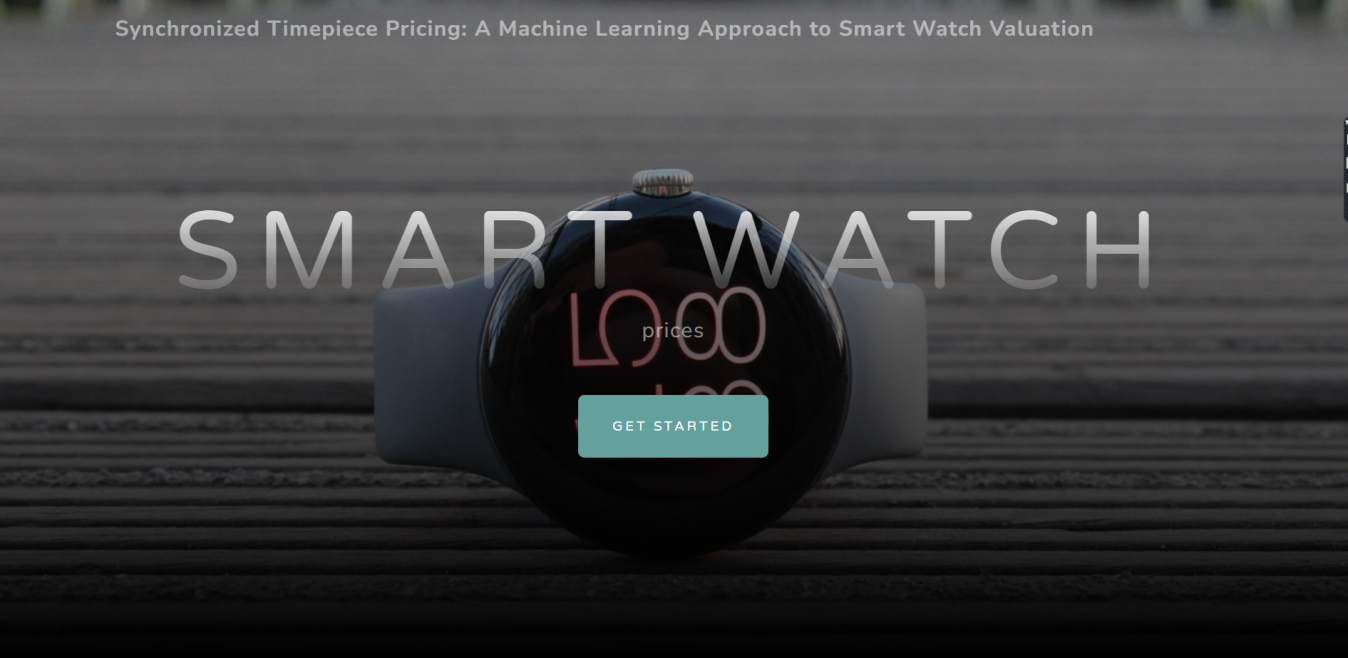
* Building HTML Pages
* Building serverside script

**Activity1: Building Html Pages:**

For this project create three HTML files namely

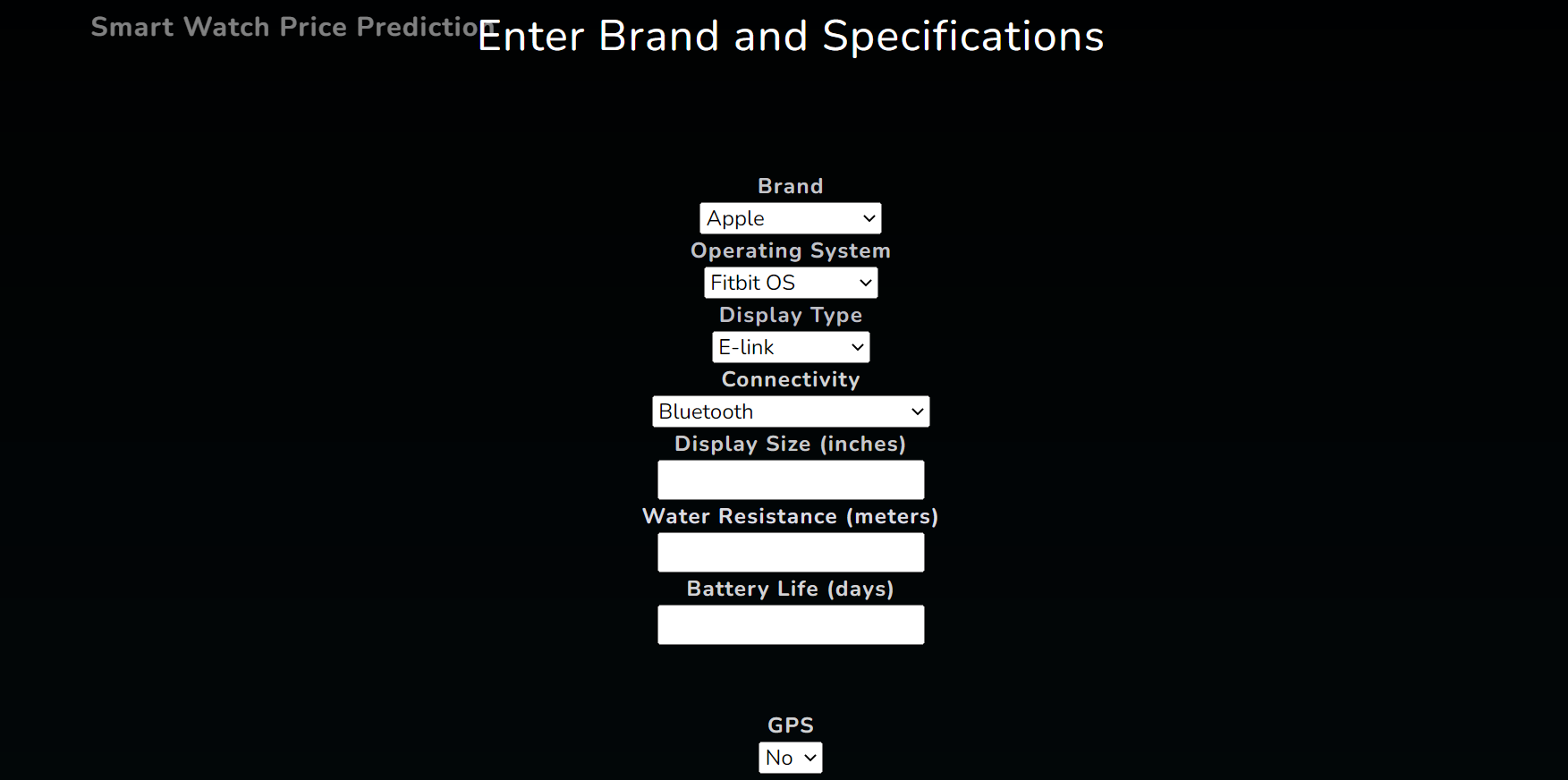
* index.html and save them in templates folder.

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



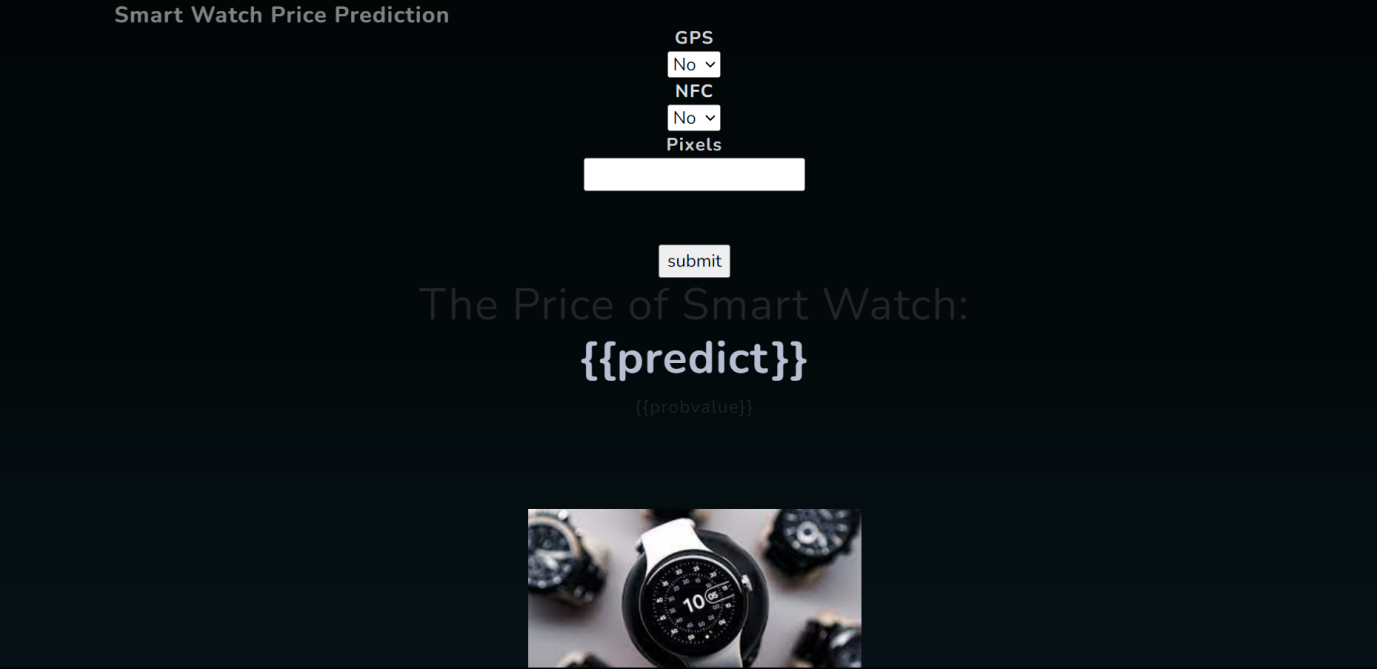
Now when you click on get started button from top right corner you will get redirected to enter values of specifications.

Lets look how our file looks like:

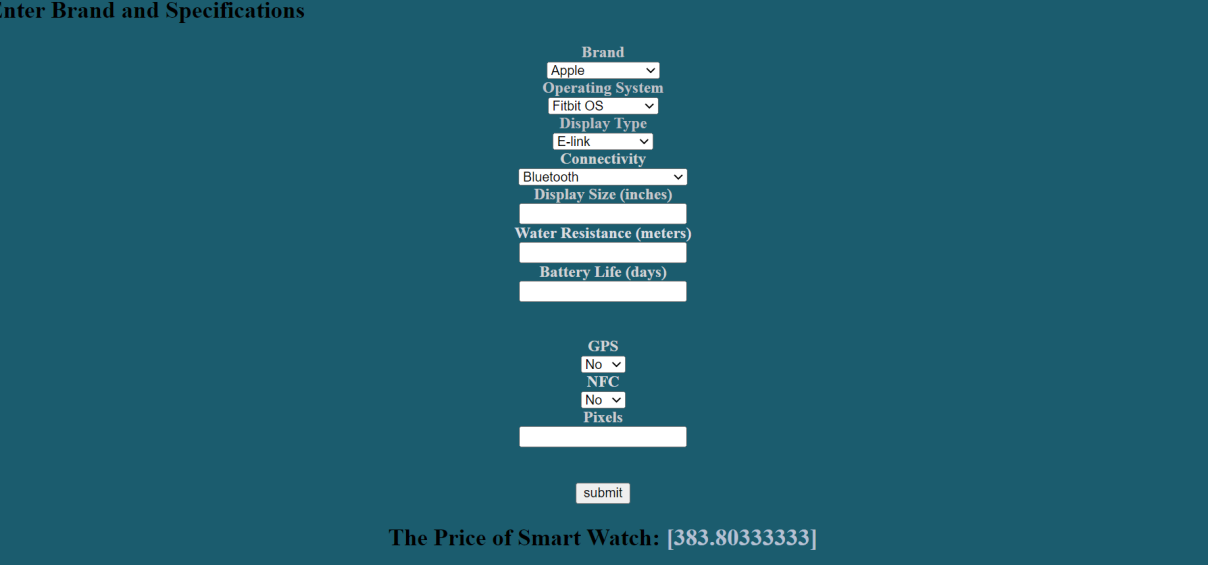


Now when you click on predict button you will get output

Lets look how our file looks like:

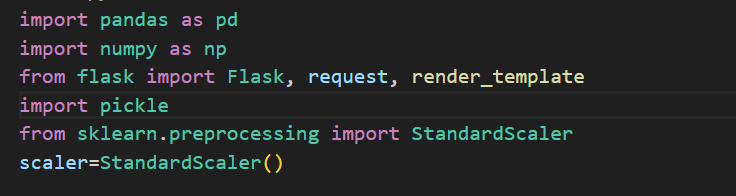


Will try with different numbers and then click on predict button and number must be in integer form only and pixels must to be enter inform of numerical like 340\*340 = 115600 .

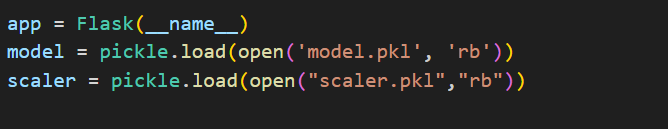


**Activity 2: Build Python code:**

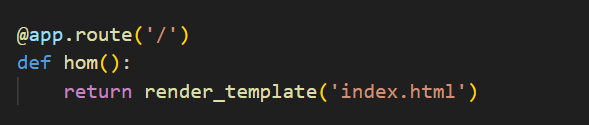
Import the libraries



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.



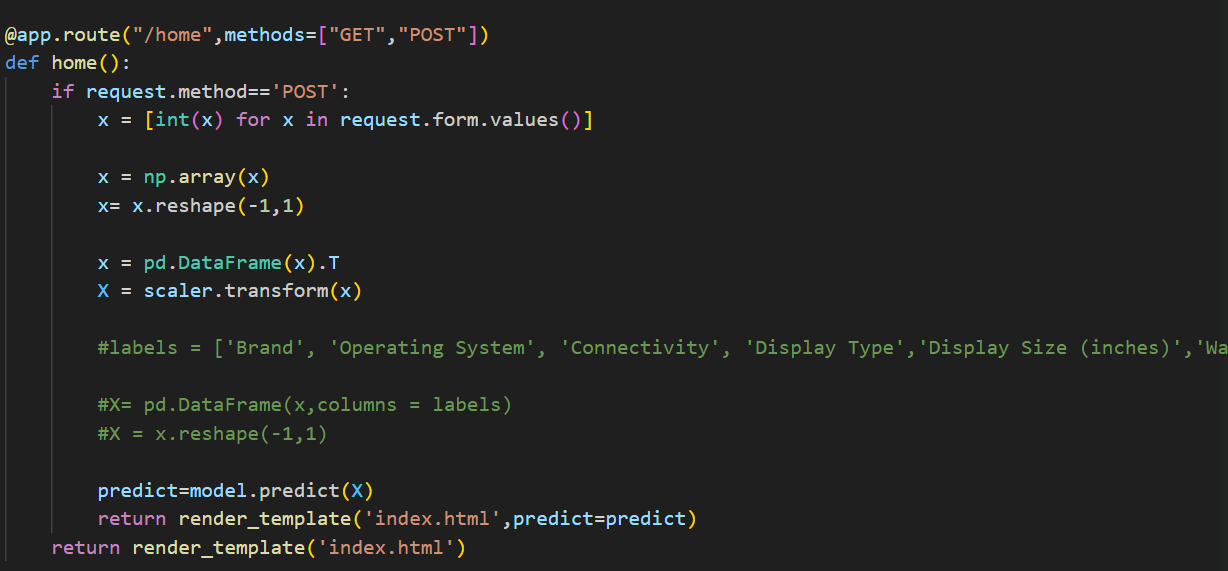
Render HTML page:



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

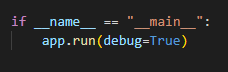
In the above example, ‘/’ URL is bound with index.html function. Hence, when the index page of the web server is opened in browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

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 Smart\_watch.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.

