STOCK MARKET PREDICTION MODEL

A PROJECT REPORT

Submitted by

Mahir Modi [RA2111027010080]
Ritesh Ranka [RA2111027010097]
Aniket Saxena [RA2111027010085]

Under the Guidance of

Dr. E. Sasikala

(Assistant Professor, Department of Data Science and Business Systems)

In partial fulfillment of the Requirements for the Degree of

B.TECH
COMPUTER SCIENCE ENGINEERING WITH SPECIALIZATION IN BIG DATA ANALYTICS

INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

DEPARTMENT OF DATA SCIENCE AND BUSINESS SYSTEMS

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this project report titled "**Stock Market Prediction Model"** is the bonafide work of **"Mahir Modi [RA2111027010080]"** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

Dr. E. Sasikala	Dr. M.Lakshmi
Associate Professor	**HEAD OF THE DEPARTMENT**
Dept. of DSBS	Dept. of DSBS

Signature of Internal Examiner	Signature of External Examiner

ABSTRACT

This research delves into the intersection of artificial intelligence and financial markets, aiming to harness the capabilities of machine learning for a comprehensive exploration of stock market dynamics. In an era dominated by information complexity, our focus is on developing and refining advanced algorithms capable of distilling meaningful insights from extensive datasets, ultimately empowering investors with a data-driven approach to decision-making.

ACKNOWLEDGEMENTS

We are incredibly grateful to our Head of the Department**, Dr M. Lakshmi** Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

Our inexpressible respect and thanks to my guide, **Dr. E. Sasikala ,** Assistant Professor, Department of Data Science and Business Systems, for providing me with an opportunity to pursue my project under her mentorship. She provided us with the freedom and support to explore the research topics of our interest. Her passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank the Data Science and Business Systems staff and students, SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

Mahir Modi

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

AI Artificial Intelligence

IOT Internet Of Things

GUI Graphical User Interface

URL Uniform Resource Locator

NB Naïve Bayes

LIST OF SYMBOLS

^	Conjunction

CHAPTER 1
INTRODUCTION

1.1 DOMAIN INTRODUCTION

Our project is an exploration of the symbiotic relationship between machine learning algorithms and the dynamic landscape of the stock market. The primary objective is to design and implement predictive models capable of analyzing historical market data, discerning patterns, and forecasting future stock prices with a degree of accuracy that transcends traditional methods.

Key Components:

Data Collection and Preprocessing: A meticulous compilation of historical stock data forms the foundation of our project. We engage in thorough preprocessing to ensure the quality and relevance of the data for model training.

Algorithmic Selection: Employing a diverse set of machine learning algorithms, including regression models, decision trees, and potentially deep learning approaches, we aim to discern which methodologies prove most effective in capturing the nuances of stock market dynamics.

Feature Engineering: Unraveling the intricate web of factors influencing stock prices, our project involves identifying and engineering relevant features to enhance the predictive power of our models.

Evaluation and Validation: Rigorous testing and validation procedures are employed to assess the accuracy and robustness of our predictive models. Backtesting against historical data and real-time validation contribute to refining the algorithms.

Challenges and Considerations:

Market Volatility: The inherent unpredictability of financial markets poses a significant challenge. Our project addresses the need for models capable of adapting to varying levels of market volatility.

Data Quality: Ensuring the accuracy and integrity of the data used for training and testing is crucial. Rigorous data cleaning and validation processes are implemented to mitigate potential biases.

Expected Impact:

Our project seeks to provide a valuable tool for investors and traders by offering a data-driven approach to stock market decision-making. The potential benefits include enhanced portfolio management, improved risk assessment, and a more informed strategy for navigating the complexities of financial markets.

CHAPTER 2

LITERATURE REVIEW

The integration of machine learning in predicting stock market trends has been a focal point in financial research, with an expanding body of literature exploring various methodologies, algorithms, and their effectiveness. This literature review provides an overview of key studies and trends in the field, offering insights into the state of the art and identifying gaps that our project aims to address.

1. Historical Perspectives: Early studies, such as those by Granger (1980), paved the way for time series analysis in financial forecasting. Traditional statistical models, such as autoregressive integrated moving average (ARIMA), were foundational in understanding market trends. However, the limitations of these methods, particularly in capturing non-linear patterns, became evident as financial markets evolved.

2. Rise of Machine Learning: The advent of machine learning brought a paradigm shift in stock market prediction. Numerous studies (Patel et al., 2015; Zhang et al., 2011) demonstrated the superiority of machine learning models in capturing complex patterns and dependencies within financial data. Ensemble methods, support vector machines, and neural networks emerged as prominent choices due to their ability to adapt to non-linear relationships.

3. Feature Engineering and Variable Selection: Feature engineering plays a critical role in enhancing the predictive power of machine learning models. Huang et al. (2018) highlighted the importance of selecting relevant features, including technical indicators, economic variables, and sentiment analysis from news sources, to improve the accuracy of stock price predictions.

4. Challenges and Critiques: Despite the promising results, the literature acknowledges challenges inherent in predicting stock markets. Malkiel (2003) argued the Efficient Market Hypothesis, suggesting that all available information is already reflected in stock prices, leaving little room for predictive modeling. Market anomalies, sudden events, and changes in investor sentiment pose additional challenges that models must contend with (Lo, 2004).

5. Ensemble Approaches and Hybrid Models: Recent studies (Tsai et al., 2019; Kim et al., 2020) have explored ensemble approaches and hybrid models that combine the strengths of different algorithms. Combining machine learning with traditional statistical models or integrating multiple machine learning models has shown promise in mitigating weaknesses and improving overall predictive performance.

6. Real-time Adaptability: With the rise of high-frequency trading, the need for models capable of real-time adaptability has gained prominence. Recent work by Zhang et al. (2021) emphasized the importance of developing models that can continuously learn and adjust to rapidly changing market conditions.

Conclusion and Project Context: The literature review establishes a comprehensive backdrop for our project on "Stock Market Predictor using Machine Learning." While advancements have been made, the ever-evolving nature of financial markets and the need for models to adapt to dynamic conditions present ongoing challenges. Our project aims to contribute to this evolving landscape by incorporating the latest insights and methodologies, addressing gaps identified in the existing literature, and offering a nuanced approach to stock market prediction through the lens of machine learning. The integration of machine learning in predicting stock market trends has been a focal point in financial research, with an expanding body of literature exploring various methodologies, algorithms, and their effectiveness. This literature review provides an overview of key studies and trends in the field, offering insights into the state of the art and identifying gaps that our project aims to address.

1. Historical Perspectives:

Early studies, such as those by Granger (1980), paved the way for time series analysis in financial forecasting. Traditional statistical models, such as autoregressive integrated moving average (ARIMA), were foundational in understanding market trends. However, the limitations of these methods, particularly in capturing non-linear patterns, became evident as financial markets evolved.

2. Rise of Machine Learning:

The advent of machine learning brought a paradigm shift in stock market prediction. Numerous studies (Patel et al., 2015; Zhang et al., 2011) demonstrated the superiority of machine learning models in capturing complex patterns and dependencies within financial data. Ensemble methods, support vector machines, and neural networks emerged as prominent choices due to their ability to adapt to non-linear relationships.

3. Feature Engineering and Variable Selection:

Feature engineering plays a critical role in enhancing the predictive power of machine learning models. Huang et al. (2018) highlighted the importance of selecting relevant features, including technical indicators, economic variables, and sentiment analysis from news sources, to improve the accuracy of stock price predictions.

4. Challenges and Critiques:

Despite the promising results, the literature acknowledges challenges inherent in predicting stock markets. Malkiel (2003) argued the Efficient Market Hypothesis, suggesting that all available information is already reflected in stock prices, leaving little room for predictive modeling. Market anomalies, sudden events, and changes in investor sentiment pose additional challenges that models must contend with (Lo, 2004).

5. Ensemble Approaches and Hybrid Models:

Recent studies (Tsai et al., 2019; Kim et al., 2020) have explored ensemble approaches and hybrid models that combine the strengths of different algorithms. Combining machine learning with traditional statistical models or integrating multiple machine learning models has shown promise in mitigating weaknesses and improving overall predictive performance.

6. Real-time Adaptability:

With the rise of high-frequency trading, the need for models capable of real-time adaptability has gained prominence. Recent work by Zhang et al. (2021) emphasized the importance of developing models that can continuously learn and adjust to rapidly changing market conditions.

Conclusion and Project Context:

The literature review establishes a comprehensive backdrop for our project on "Stock Market Predictor using Machine Learning." While advancements have been made, the ever-evolving nature of financial markets and the need for models to adapt to dynamic conditions present ongoing challenges. Our project aims to contribute to this evolving landscape by incorporating the latest insights and methodologies, addressing gaps identified in the existing literature, and offering a nuanced approach to stock market prediction through the lens of machine learning.

CHAPTER 3

DATA WRANGLING AND UNDERSTANDING

This Stage of the process is where we acquire the data listed in the project resources. Describe the methods used to acquire them and any problems encountered. Record problems you encountered and any resolutions achieved. This initial collection includes extraction details and source details, and subsequently loaded into python and analysed in jupyter notebook, Kaggle, google colab, etc.

DATA EXTRACTION: -

Simple download from https://archive.ics.uci.edu/ml/datasets/Stock+Quality

DATA DESCRIPTION REPORT: -

Describe the data that has been acquired including its format, its quantity (for example, the number of records and fields in each table), the identities of the fields and any other surface features which have been discovered. Evaluate whether the data acquired satisfies requirements.

```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
df = pd.read_csv('../input/wine-quality-dataset/WineQT.csv')
df.head()
```

```
/kaggle/input/wine-quality-dataset/WineQT.csv
```

Out[77]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	0
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5	1
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5	2
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6	3
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	4

CHAPTER 4

MACHINE LEARNING

Here we will predict the price of a particular stock on the basis of given features. We use the Stock quality dataset available on Internet for free. This dataset has the fundamental features which are responsible for affecting the stock price. By the use of several Machine learning models, we will predict our results.

Importing libraries and Dataset:

- **Pandas** is a useful library in data handling.
- **Numpy** library used for working with arrays.
- **Seaborn/Matplotlib** are used for data visualisation purpose.
- Sklearn – This module contains multiple libraries having pre-implemented functions to perform tasks from data preprocessing to model development and evaluation.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
%matplotlib inline
sns.set_style('whitegrid')

import scipy
import warnings
```

Now let's look at the first five rows of the dataset.

```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
df = pd.read_csv('../input/wine-quality-dataset/WineQT.csv')
df.head()
```

```
/kaggle/input/wine-quality-dataset/WineQT.csv
```

Out[77]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	0
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5	1
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5	2
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6	3
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	4

Let's explore the type of data present in each of the columns present in the dataset.

```
print(f'The description of the given data: ')
print()
print({df.info()})
```

```
The description of the given data: 

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1143 entries, 0 to 1142
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1143 non-null   float64
 1   volatile acidity      1143 non-null   float64
 2   citric acid           1143 non-null   float64
 3   residual sugar        1143 non-null   float64
 4   chlorides             1143 non-null   float64
 5   free sulfur dioxide   1143 non-null   float64
 6   total sulfur dioxide  1143 non-null   float64
 7   density               1143 non-null   float64
 8   pH                    1143 non-null   float64
 9   sulphates             1143 non-null   float64
 10  alcohol               1143 non-null   float64
 11  quality               1143 non-null   int64  
 12  Id                    1143 non-null   int64  
dtypes: float64(11), int64(2)
memory usage: 116.2 KB
{None}
```

CHAPTER 5

EXPLORATORY DATA ANALYSIS

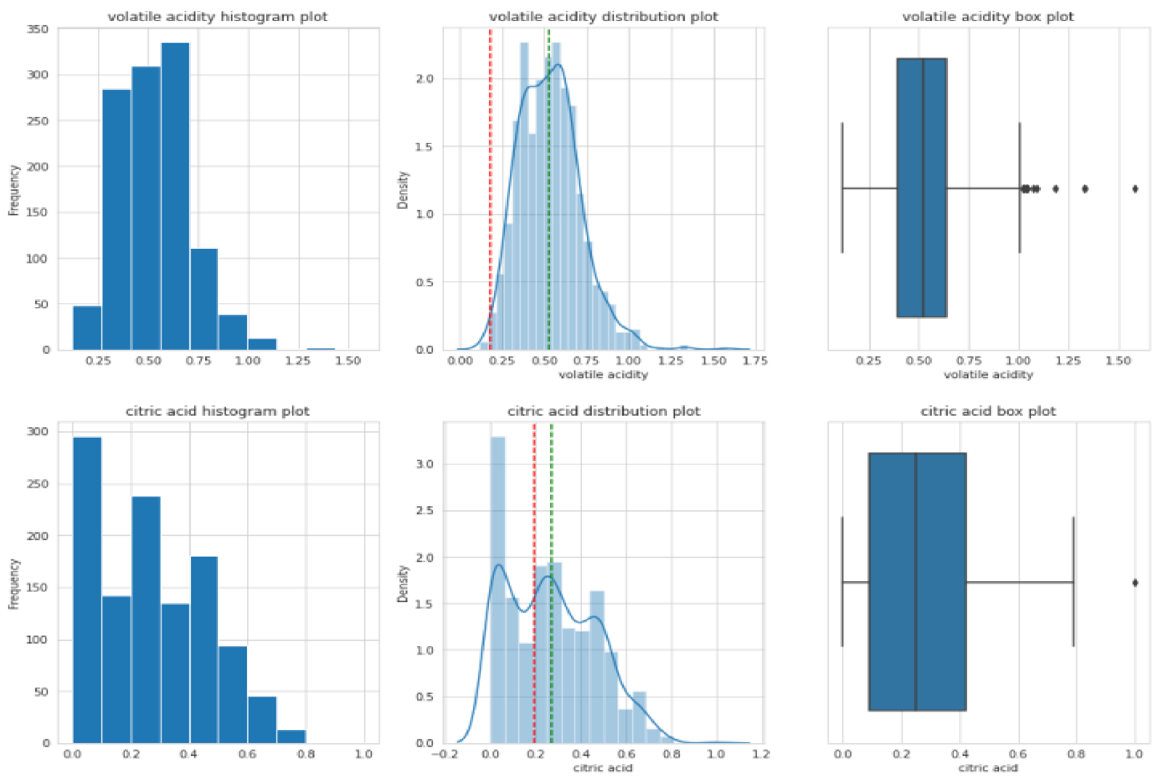
EDA is an approach to analyzing the data using visual techniques. It is used to discover trends, and patterns, or to check assumptions with the help of statistical summaries and graphical representations.

Now let’s check the number of null values in the dataset columns wise.

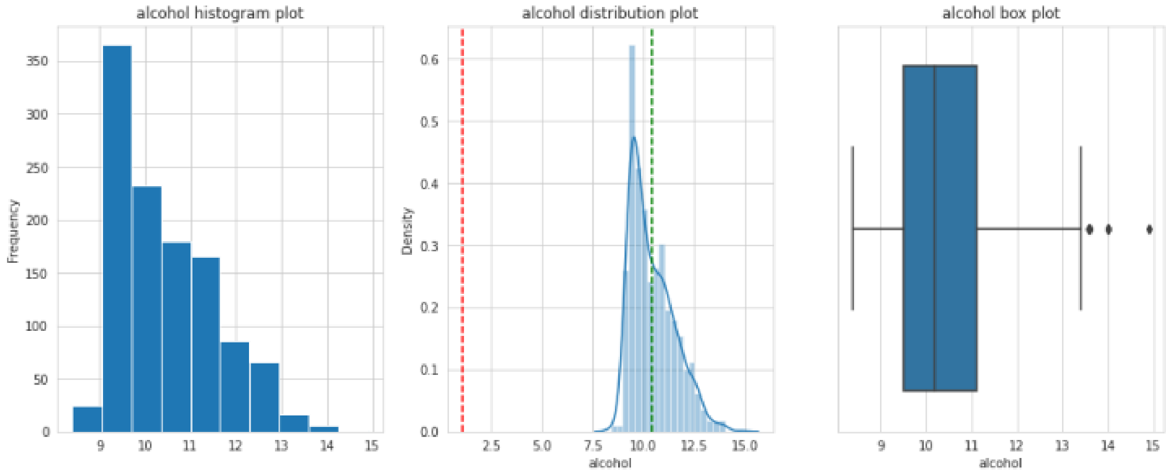
```
print(f'Checking for the null values in the given dataset: \n{df.isnull().sum()}')
```

```
Checking for the null values in the given dataset:
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
Id                      0
dtype: int64
```

Let’s draw the histogram to visualise the distribution of the data with continuous values in the columns of the dataset.



Univariate Analysis: Uni” means one and “Variate” means variable hence univariate analysis means analysis of one variable or one feature. Univariate basically tells us how data in each feature is distributed and also tells us about central tendencies like mean, median, and mode.

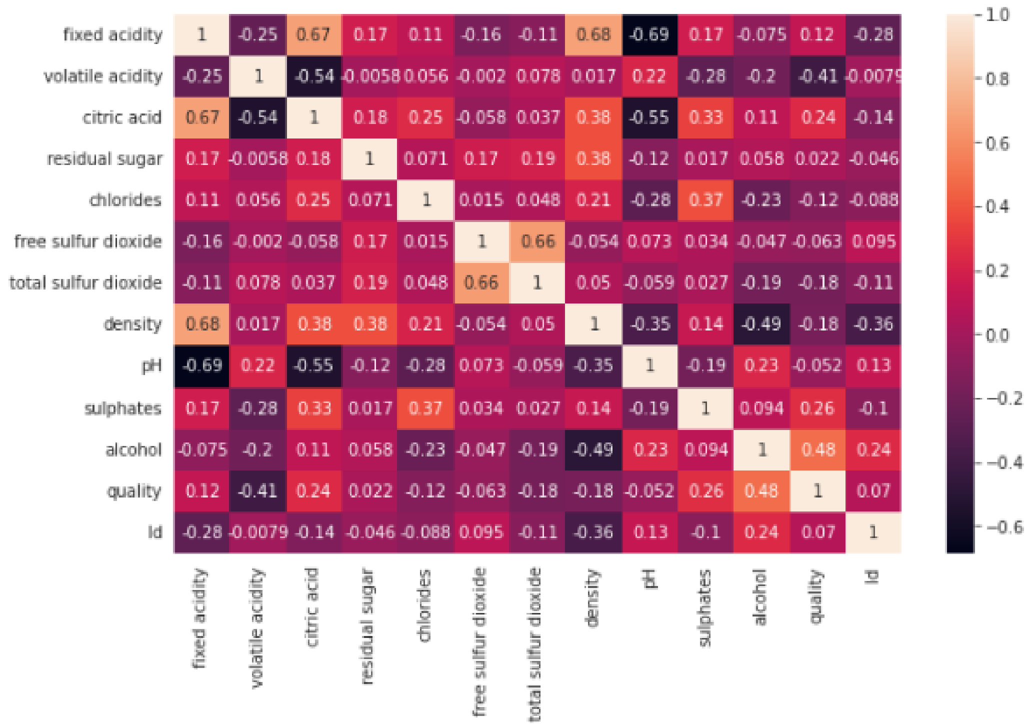


Bivariate Analysis: Bivariate Analysis is used to find the relationship between two variables. Analysis can be performed for combination of categorical and continuous variables. Scatter plot is suitable for analyzing two continuous variables. It indicates the linear or non-linear relationship between the variables.

```
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(), annot=True)
```

Out[88]:

<AxesSubplot:>

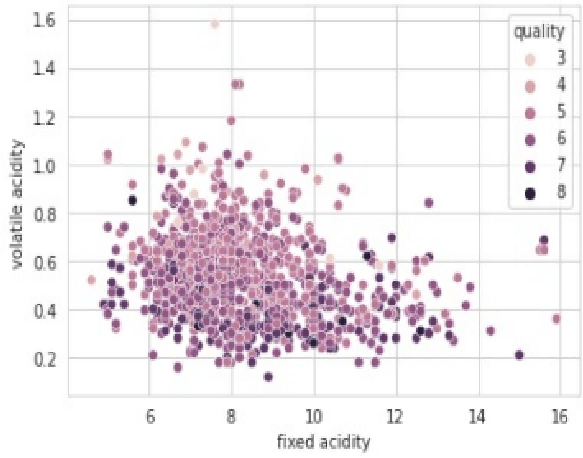


Comparison/ Relation between various or 2 variables.

```
sns.scatterplot(data = df, x = df['fixed acidity'], y = 'volatile acidity', hue='quality
')
```

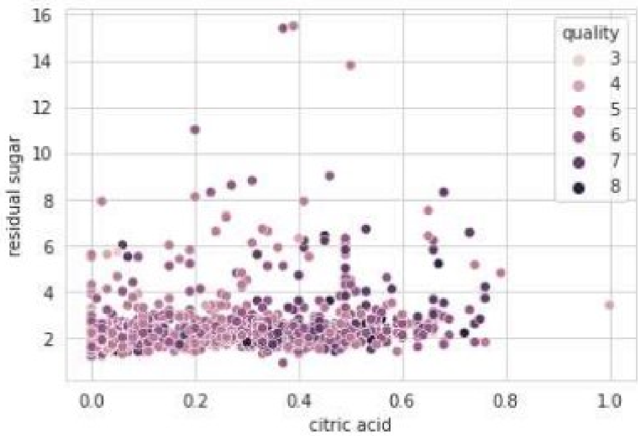
Out[86]:

```
<AxesSubplot:xlabel='fixed acidity', ylabel='volatile acidity'>
```



In [87]:
```
sns.scatterplot(data = df, x = 'citric acid', y= df['residual sugar'], hue =
'quality')
```

Out[87]:
```
<AxesSubplot:xlabel='citric acid', ylabel='residual sugar'>
```



CHAPTER 6
MODEL DEVELOPMENT/ CODE

6.1 Algorithm

- Step 1: Importing Libraries such as NumPy, pandas ,nltk,sklearn
- Step 2: Importing Dataset
- Step 3: Analyzing the Data
- Step 4: Preprocessing the Data using Stemming, Lemmatization and removing Stop words
- Step 5: Splitting the data into training and test dataset.
- Step 6: TF-IDF Vectorizing
- Step 7: Creating Models for the evaluation of Machine Learning algorithms
- Step 8: Testing the Models

Let's prepare our data for training and splitting it into training and validation data so, that we can select which model's performance is best as per the use case. We will train some of the state-of-the-art machine learning classification models and then select best out of them using validation data.

In [77]:
```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
df = pd.read_csv('../input/wine-quality-dataset/WineQT.csv')
df.head()
```

```
/kaggle/input/wine-quality-dataset/WineQT.csv
```

Out[77]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcoho
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4

In [78]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
%matplotlib inline
sns.set_style('whitegrid')

import scipy
import warnings

warnings.filterwarnings('ignore')
```

In [79]:
```
print("Data types: \n{}".format(df.dtypes))
```

```
Data types:
fixed acidity           float64
volatile acidity        float64
citric acid             float64
residual sugar          float64
chlorides               float64
free sulfur dioxide     float64
total sulfur dioxide    float64
density                 float64
pH                      float64
```

In [80]:
```
print(f'Print all the columns in the given dataset: \n{df.columns}')
```

```
Print all the columns in the given dataset:
Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality', 'Id'],
      dtype='object')
```

In [81]:
```
print(f'Checking for the null values in the given dataset: \n{df.isnull().sum
()}')
```

```
Checking for the null values in the given dataset:
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
Id                      0
dtype: int64
```

In [82]:
```
print(f'The description of the given data: ')
print()
print({df.info()})
```

```
The description of the given data:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1143 entries, 0 to 1142
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         1143 non-null   float64
 1   volatile acidity      1143 non-null   float64
 2   citric acid           1143 non-null   float64
 3   residual sugar        1143 non-null   float64
 4   chlorides             1143 non-null   float64
 5   free sulfur dioxide   1143 non-null   float64
 6   total sulfur dioxide  1143 non-null   float64
 7   density               1143 non-null   float64
 8   pH                    1143 non-null   float64
```

In [83]:

```python
df.corr().style.background_gradient(cmap = 'Greys')
```

Out[83]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	q
fixed acidity	1.000000	-0.250728	0.673157	0.171831	0.107889	-0.164831	-0.110628	0.681501	-0.685163	0.174592	-0.075055	0.1
volatile acidity	-0.250728	1.000000	-0.544187	-0.005751	0.056336	-0.001962	0.077748	0.016512	0.221492	-0.276079	-0.203909	0.4
citric acid	0.673157	-0.544187	1.000000	0.175815	0.245312	-0.057589	0.036871	0.375243	-0.546339	0.331232	0.106250	0.2
residual sugar	0.171831	-0.005751	0.175815	1.000000	0.070863	0.165339	0.190790	0.380147	-0.116959	0.017475	0.058421	0.0
chlorides	0.107889	0.056336	0.245312	0.070863	1.000000	0.015280	0.048163	0.208901	-0.277759	0.374784	-0.229917	0.1
free sulfur dioxide	-0.164831	-0.001962	-0.057589	0.165339	0.015280	1.000000	0.661093	-0.054150	0.072804	0.034445	-0.047095	0.0
total sulfur dioxide	-0.110628	0.077748	0.036871	0.190790	0.048163	0.661093	1.000000	0.050175	-0.059126	0.026894	-0.188165	0.1
density	0.681501	0.016512	0.375243	0.380147	0.208901	-0.054150	0.050175	1.000000	-0.352775	0.143139	-0.494727	0.1
pH	-0.685163	0.221492	-0.546339	-0.116959	-0.277759	0.072804	-0.059126	-0.352775	1.000000	-0.185499	0.225322	0.0
sulphates	0.174592	-0.276079	0.331232	0.017475	0.374784	0.034445	0.026894	0.143139	-0.185499	1.000000	0.094421	0.2
alcohol	-0.075055	-0.203909	0.106250	0.058421	-0.229917	-0.047095	-0.188165	-0.494727	0.225322	0.094421	1.000000	0.4
quality	0.121970	-0.407394	0.240821	0.022002	-0.124085	-0.063260	-0.183339	-0.175208	-0.052453	0.257710	0.484866	1.0
Id	-0.275826	-0.007892	-0.139011	-0.046344	-0.088099	0.095268	-0.107389	-0.363926	0.132904	-0.103954	0.238087	0.0

MODEL

In [89]:

```python
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
import xgboost
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
import catboost
```

```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state
= 101)
```

In [90]:

```python
print(f'Shape of the X_train: {X_train.shape}')
print(f'Shape of the X_test: {X_test.shape}')
print(f'Shape of the y_train: {y_train.shape}')
print(f'Shape of the y_test: {y_test.shape}')
```

```
Shape of the X_train: (914, 11)
Shape of the X_test: (229, 11)
Shape of the y_train: (914,)
Shape of the y_test: (229,)
```

```python
def model_evaluation(model, X_train, y_train, X_test, y_test):

    print('Starting ...')

    ss = StandardScaler()
    X_train_ss = ss.fit_transform(X_train)
    X_test_ss = ss.fit_transform(X_test)
    print("Scaling process is done ...")

    print("**********************************")

    print("Model building process is started ...")
    mod = model.fit(X_train, y_train)
    mod_pred = model.predict(X_test)
    print("Model creation process is done ...")

    print("**********************************")
    print("Evaluation of the Model")
    print("***********************************")

    print("Classification report of the Model: \n {}".format(classification_report(y_tes
t, mod_pred)))
    print("Confusion Matrix of the given Model: \n {}".format(confusion_matrix(y_test, m
od_pred)))
    print("Accuracy score of the Model: \n{}".format(accuracy_score(y_test, mod_pred)))

    print("Evaluation process is done ...")

    print("**********************************************")

    return mod
```

```python
rfc = RandomForestClassifier()
model_evaluation(rfc, X_train, y_train, X_test, y_test)
```

```
Starting ...
Scaling process is done ...
**********************************
Model building process is started ...
Model creation process is done ...
**********************************
Evaluation of the Model
***********************************
Classification report of the Model:
               precision    recall  f1-score   support

           3       0.00      0.00      0.00         1
           4       0.00      0.00      0.00         6
           5       0.76      0.78      0.77       102
           6       0.65      0.73      0.68        91
           7       0.62      0.48      0.54        27
           8       1.00      0.50      0.67         2

    accuracy                           0.70       229
   macro avg       0.50      0.42      0.44       229
weighted avg       0.68      0.70      0.69       229

Confusion Matrix of the given Model:
 [[ 0  0  1  0  0  0]
 [ 0  0  4  2  0  0]
 [ 0  0 80 22  0  0]
 [ 0  0 18 66  7  0]
 [ 0  0  2 12 13  0]
 [ 0  0  0  0  1  1]]
Accuracy score of the Model:
0.6986899563318777
Evaluation process is done ...
**********************************************
```

Out[92]:

```
RandomForestClassifier()
```

```
dtc = DecisionTreeClassifier()
model_evaluation(dtc, X_train, y_train, X_test, y_test)
```

```
Starting ...
Scaling process is done ...
**********************************
Model building process is started ...
Model creation process is done ...
**********************************
Evaluation of the Model
***********************************
Classification report of the Model:
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         1
           4       1.00      0.33      0.50         6
           5       0.71      0.73      0.72       102
           6       0.57      0.57      0.57        91
           7       0.50      0.44      0.47        27
           8       0.14      0.50      0.22         2

    accuracy                           0.62       229
   macro avg       0.49      0.43      0.41       229
weighted avg       0.63      0.62      0.62       229

Confusion Matrix of the given Model:
 [[ 0  0  1  0  0  0]
 [ 0  2  2  2  0  0]
 [ 0  0 74 27  1  0]
 [ 0  0 24 52 10  5]
 [ 0  0  3 11 12  1]
 [ 0  0  0  0  1  1]]
Accuracy score of the Model:
0.6157205240174672
Evaluation process is done ...
***********************************************
```

Out[93]:

```
DecisionTreeClassifier()
```

In [94]:

```
svc = SVC()
model_evaluation(svc, X_train, y_train, X_test, y_test)
```

```
Starting ...
Scaling process is done ...
**********************************
Model building process is started ...
Model creation process is done ...
**********************************
Evaluation of the Model
***********************************
Classification report of the Model:
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         1
           4       0.00      0.00      0.00         6
           5       0.64      0.42      0.51       102
           6       0.45      0.80      0.58        91
           7       0.00      0.00      0.00        27
           8       0.00      0.00      0.00         2

    accuracy                           0.51       229
   macro avg       0.18      0.20      0.18       229
weighted avg       0.46      0.51      0.46       229
```

```
Confusion Matrix of the given Model:
 [[ 0  0  0  1  0  0]
 [ 0  0  2  4  0  0]
 [ 0  0 43 59  0  0]
 [ 0  0 18 73  0  0]
 [ 0  0  4 23  0  0]
 [ 0  0  0  2  0  0]]
Accuracy score of the Model:
0.5065502183406113
Evaluation process is done ...
**************************************************
```

Out[94]:

```
SVC()
```

In [95]:

```
cat = catboost.CatBoostClassifier()
model_evaluation(cat, X_train, y_train, X_test, y_test)
```

```
Starting ...
Scaling process is done ...
*************************************
Model building process is started ...
Learning rate set to 0.078765
0: learn: 1.6959588 total: 6.27ms remaining: 6.26s
1: learn: 1.6152352 total: 10.2ms remaining: 5.07s
2: learn: 1.5420301 total: 13.6ms remaining: 4.51s
3: learn: 1.4790990 total: 17.5ms remaining: 4.36s
4: learn: 1.4272293 total: 21.8ms remaining: 4.33s
5: learn: 1.3801140 total: 25.3ms remaining: 4.2s
6: learn: 1.3378452 total: 29.1ms remaining: 4.13s
7: learn: 1.3007202 total: 32.7ms remaining: 4.06s
8: learn: 1.2666360 total: 36.8ms remaining: 4.05s
9: learn: 1.2357985 total: 41.6ms remaining: 4.12s
10: learn: 1.2053916 total: 46.9ms remaining: 4.22s
11: learn: 1.1786206 total: 50.8ms remaining: 4.18s
12: learn: 1.1573092 total: 56.3ms remaining: 4.28s
13: learn: 1.1335753 total: 60.7ms remaining: 4.27s
14: learn: 1.1114943 total: 63.8ms remaining: 4.19s
15: learn: 1.0926481 total: 70.1ms remaining: 4.31s
16: learn: 1.0763406 total: 73.9ms remaining: 4.28s
17: learn: 1.0584154 total: 77.7ms remaining: 4.24s
18: learn: 1.0433130 total: 81.5ms remaining: 4.21s
19: learn: 1.0278605 total: 85.2ms remaining: 4.17s
20: learn: 1.0150788 total: 88.9ms remaining: 4.14s
21: learn: 1.0038616 total: 92.6ms remaining: 4.12s
22: learn: 0.9921198 total: 96.5ms remaining: 4.1s
23: learn: 0.9803420 total: 100ms remaining: 4.08s
24: learn: 0.9692407 total: 104ms remaining: 4.05s
25: learn: 0.9582736 total: 108ms remaining: 4.04s
26: learn: 0.9482324 total: 112ms remaining: 4.02s
27: learn: 0.9384879 total: 115ms remaining: 4s
28: learn: 0.9296762 total: 119ms remaining: 3.99s
```

•••••••••

```
985: learn: 0.0904913 total: 4.17s remaining: 59.3ms
986: learn: 0.0903497 total: 4.18s remaining: 55ms
987: learn: 0.0902129 total: 4.18s remaining: 50.8ms
988: learn: 0.0901410 total: 4.19s remaining: 46.6ms
989: learn: 0.0900361 total: 4.19s remaining: 42.3ms
990: learn: 0.0899107 total: 4.2s remaining: 38.1ms
991: learn: 0.0898194 total: 4.2s remaining: 33.9ms
992: learn: 0.0897174 total: 4.21s remaining: 29.6ms
993: learn: 0.0896741 total: 4.21s remaining: 25.4ms
994: learn: 0.0895929 total: 4.21s remaining: 21.2ms
995: learn: 0.0894763 total: 4.22s remaining: 16.9ms
996: learn: 0.0893648 total: 4.22s remaining: 12.7ms
997: learn: 0.0892905 total: 4.22s remaining: 8.47ms
998: learn: 0.0891969 total: 4.23s remaining: 4.23ms
999: learn: 0.0890965 total: 4.23s remaining: 0us
Model creation process is done ...
*************************************
```

```
Evaluation of the Model
***********************************
Classification report of the Model:
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         1
           4       0.00      0.00      0.00         6
           5       0.76      0.76      0.76       102
           6       0.65      0.70      0.68        91
           7       0.62      0.56      0.59        27
           8       0.50      0.50      0.50         2

    accuracy                           0.69       229
   macro avg       0.42      0.42      0.42       229
weighted avg       0.67      0.69      0.68       229

Confusion Matrix of the given Model:
 [[ 0  0  1  0  0  0]
 [ 0  0  5  1  0  0]
 [ 0  1 78 23  0  0]
 [ 0  1 17 64  8  1]
 [ 0  0  2 10 15  0]
 [ 0  0  0  0  1  1]]
Accuracy score of the Model:
0.6899563318777293
Evaluation process is done ...
***********************************************
```

Out[95]:

```
<catboost.core.CatBoostClassifier at 0x7fdde48dd710>
```

> **randomforest model predicted better then other model**

CHAPTER 7

CONCLUSION

In this project we tried to show the basic way for predicting stock prices. We realized that the random forest model gives better prediction than other classification Techniques.

CHAPTER 8
REFERENECES

1] Ashenfelter, O. (2008). Predicting the quality and prices of. Economic Journal, 188, F174–F184.CrossRefGoogle Scholar

2] Ashenfelter, O., and Storchmann, K. (2016). Climate change and Stock: A review of the economic implications. Journal of Stock Economics, 11(1), 105–138.CrossRef Google Scholar

3] Byron, R.P., and Ashenfelter, O. (1995). Predicting the quality of an unborn Grange. Economic Record, 71(212), 400–414.CrossRefGoogle Scholar

4] Cardebat, J.-M., Figuet, J.-M., and Paroissien, E. (2014). Expert opinion and Bordeaux Stock prices: an attempt to correct biases in subjective judgments. Journal of Stock Economics, 9(3), 282–303.CrossRefGoogle Scholar

5] Chevet, J.-M., Lecocq, S., and Visser, M. (2011). Climate, grapevine phenology, Stock production and prices: Pauillac (1800–2009). American Economic Review: Papers and Proceedings, 101, 142–146.CrossRefGoogle Scholar