STOCK PRICE PREDICTION USING APPLIED DATA SCIENCE

INTRODUCTION

The stock market is a dynamic and complex system influenced by a multitude of factors,including economic indicators, news events, investor sentiment, and market trends. Predicting its movements has been a subject of great interest for investors, traders, and researchers alike.Applied Data Science provides a powerful toolkit to analyze historical data, extract meaningful patterns, and build models to make informed predictions about future market behaviour.The goals of this project have been outlined, a data analysis plan has been developed trained tested the model and useful visualization types have been identified, and Python and data visualization libraries are being used for the analysis in “Jupyter Notebook”.

DESCRIPTION:

This stage focuses on prepping the provided CSV file in order to prepare it for other operations including Feature engineering,Model training ,Evaluation. Using the machine learning algorithm training and evaluation is part of phase 4 of the study "STOCK PRICE PREDICTION". This document includes a feature engineering using linear regression and training the model using logistic regression and evaluating the accuracy of the model trained.

Dataset

https://www.kaggle.com/datasets/prasoonkottarathil/microsoft-lifetime-stocks-dataset

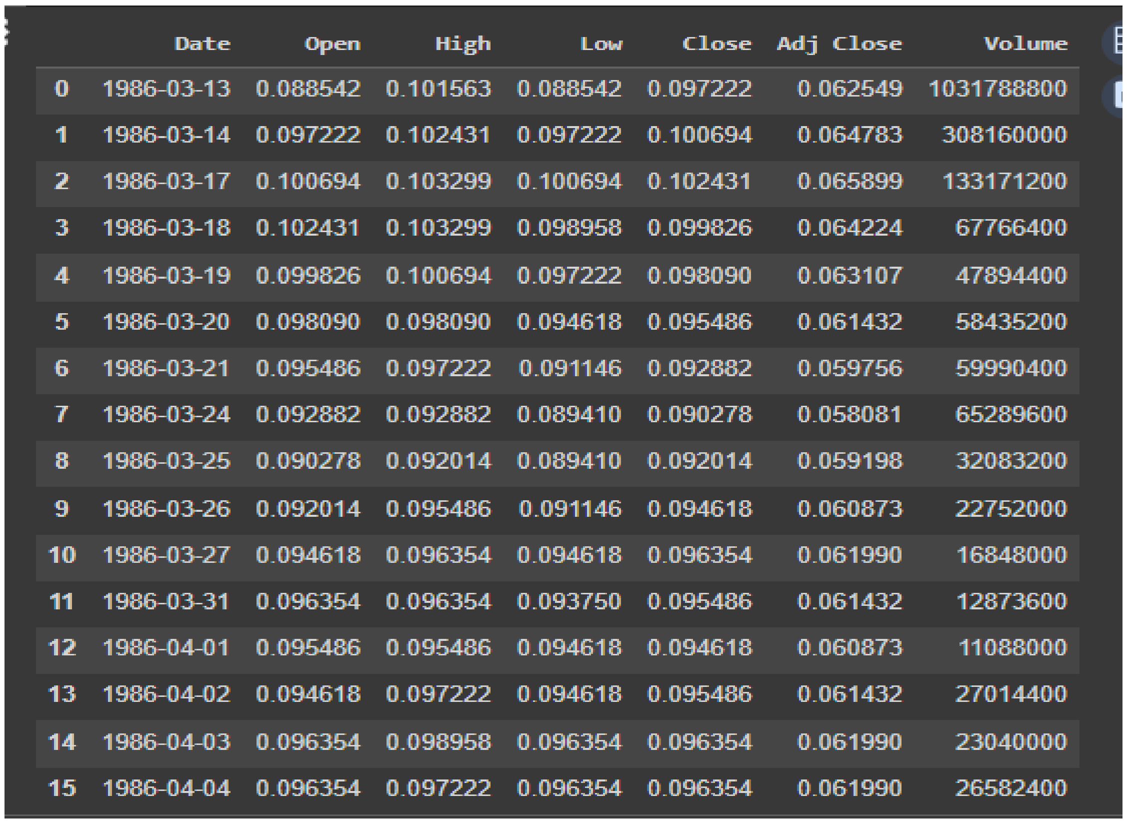
IMPLEMENTATION:

Load the dataset

Load the dataset a csv file from kaggle and using Pandas package load them

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
warnings.filterwarnings('ignore')
df = pd.read_csv('stock prediction.csv')
df.head(15)
```

OUTPUT



	Date	Open	High	Low	Close	Adj Close	Volume
0	1986-03-13	0.088542	0.101563	0.088542	0.097222	0.062549	1031788800
1	1986-03-14	0.097222	0.102431	0.097222	0.100694	0.064783	308160000
2	1986-03-17	0.100694	0.103299	0.100694	0.102431	0.065899	133171200
3	1986-03-18	0.102431	0.103299	0.098958	0.099826	0.064224	67766400
4	1986-03-19	0.099826	0.100694	0.097222	0.098090	0.063107	47894400
5	1986-03-20	0.098090	0.098090	0.094618	0.095486	0.061432	58435200
6	1986-03-21	0.095486	0.097222	0.091146	0.092882	0.059756	59990400
7	1986-03-24	0.092882	0.092882	0.089410	0.090278	0.058081	65289600
8	1986-03-25	0.090278	0.092014	0.089410	0.092014	0.059198	32083200
9	1986-03-26	0.092014	0.095486	0.091146	0.094618	0.060873	22752000
10	1986-03-27	0.094618	0.096354	0.094618	0.096354	0.061990	16848000
11	1986-03-31	0.096354	0.096354	0.093750	0.095486	0.061432	12873600
12	1986-04-01	0.095486	0.095486	0.094618	0.094618	0.060873	11088000
13	1986-04-02	0.094618	0.097222	0.094618	0.095486	0.061432	27014400
14	1986-04-03	0.096354	0.098958	0.096354	0.096354	0.061990	23040000
15	1986-04-04	0.096354	0.097222	0.096354	0.096354	0.061990	26582400

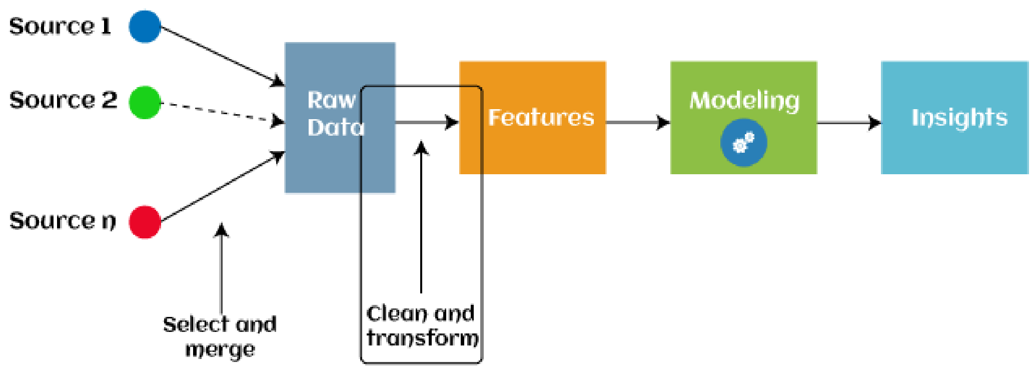
IMPORTING NECESSARY PACKAGE:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVC
from xgboost import XGBClassifier
from sklearn import metrics
```

FEATURED ENGINEERING

Feature engineering is the process of transforming raw data into features that are suitable for machine learning models. In other words, it is the process of selecting, extracting, and transforming the most relevant features from the available data to build more accurate and efficient machine learning models.Here in this stock prediction model Open,Close,High,Low column are transformed as features.Using Linear Regression and Series for visualization

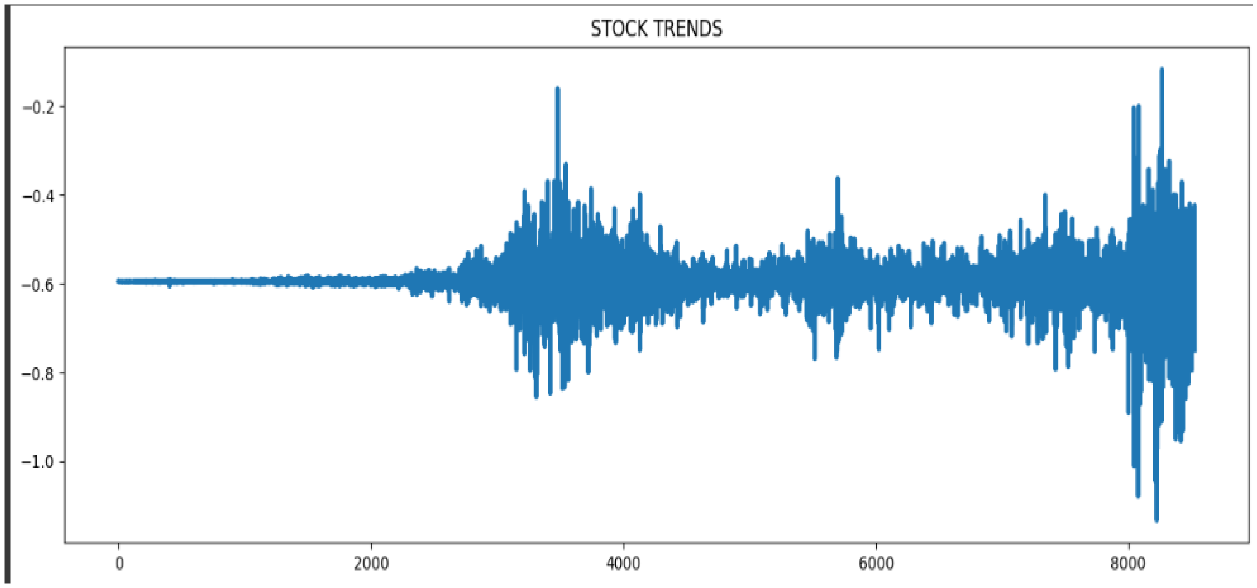


```
df['open-close']  = df['Open'] - df['Close']
df['low-high']  = df['Low'] - df['High']
# Training data
X = df.loc[:, ['open-close']]  # features
y = df.loc[:, 'low-high']  # target

# Train the model
model = LinearRegression()
model.fit(X, y)

# Store the fitted values as a time series with the same time index as
# the training data
y_pred = pd.Series(model.predict(X), index=X.index)
ax= y_pred.plot( linewidth=3)
ax.set_title('STOCK TRENDS')
```

OUTPUT



MODEL TRAINING:

Logistic regression is a supervised machine learning algorithm mainly used for classification tasks where the goal is to predict the probability that an instance of belonging to a given class or not. It is a kind of statistical algorithm, which analyze the relationship between a set of independent variables and the dependent binary variables.Here the independent and dependent variables are Open-Close and Low-High respectively.

```
df['open-close']  = df['Open'] - df['Close']
df['low-high']  = df['Low'] - df['High']
df['target'] = np.where(df['Close'].shift(-1) > df['Close'], 1, 0)

features = df[['open-close', 'low-high']]
target = df['target']

scaler = StandardScaler()
features = scaler.fit_transform(features)

X_train, X_valid, Y_train, Y_valid = train_test_split(
    features, target, test_size=0.1, random_state=2022)

print(X_train.shape, X_valid.shape)

models = [LogisticRegression(), SVC(
  kernel='poly', probability=True), XGBClassifier()]
```

EVALUATION:

In stock prediction the model is evaluated based on its accuracy in prediction of the stocks future trends and also evaluated based on validation accuracy.If the accuracy is towards close to 1 then the model is more accurate and prediction are more true and accurate.

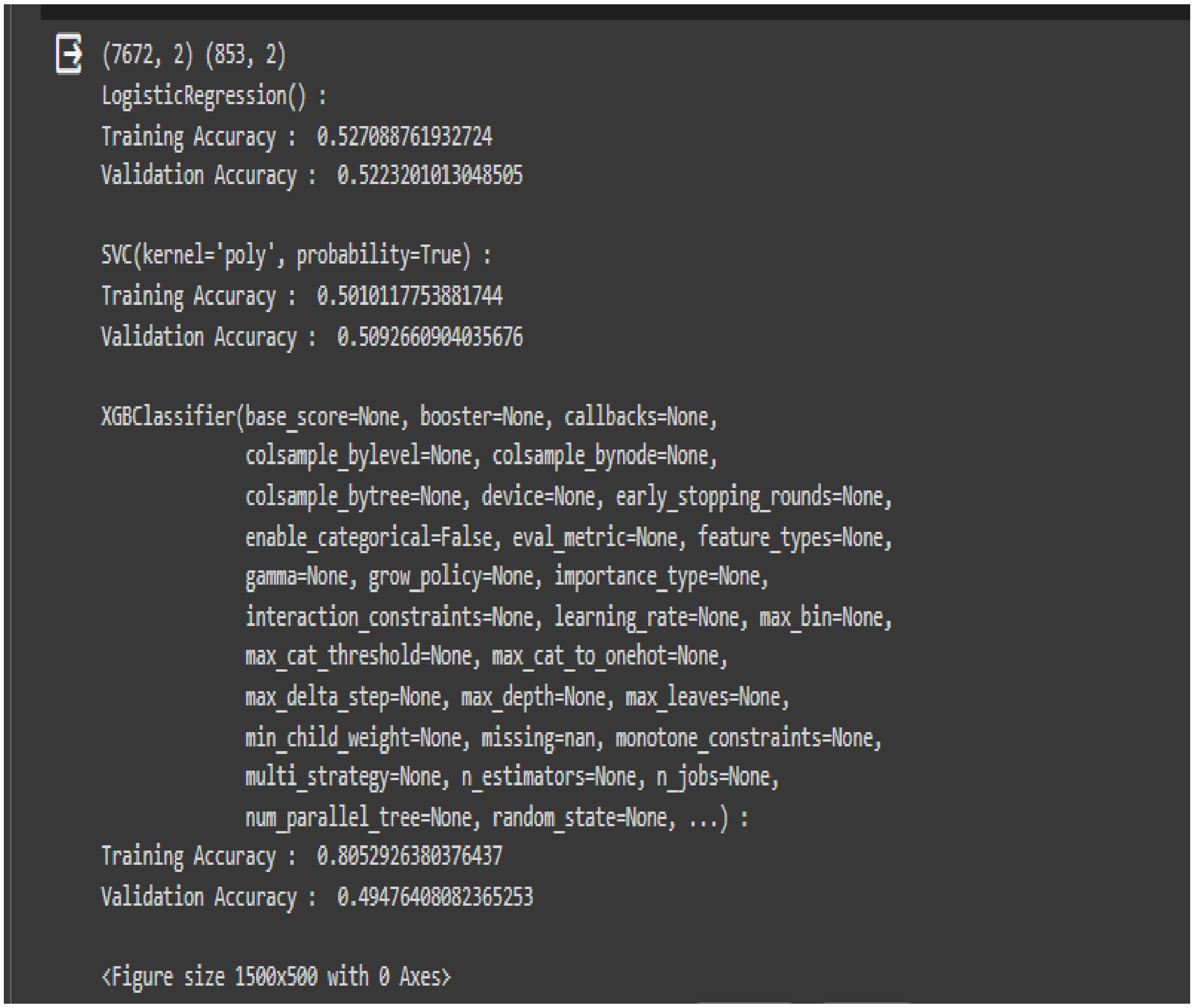
```python
for i in range(3):
  models[i].fit(X_train, Y_train)

  print(f'{models[i]} : ')

  print('Training Accuracy : ', metrics.roc_auc_score(
    Y_train, models[i].predict_proba(X_train)[:,1]))

  print('Validation Accuracy : ', metrics.roc_auc_score(
    Y_valid, models[i].predict_proba(X_valid)[:,1]))
  print()
```

OUTPUT



```
(7672, 2) (853, 2)
LogisticRegression() :
Training Accuracy :  0.527088761932724
Validation Accuracy :  0.5223201013048505

SVC(kernel='poly', probability=True) :
Training Accuracy :  0.5010117753881744
Validation Accuracy :  0.5092660904035676

XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=None, n_jobs=None,
              num_parallel_tree=None, random_state=None, ...) :
Training Accuracy :  0.8052926380376437
Validation Accuracy :  0.49476408082365253

<Figure size 1500x500 with 0 Axes>
```