Prediction With Naive Bayes Algorithm

Importing Libraries

In [311…

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

Importing Datasets

In [312…

```
AAPL_df=pd.read_csv('Labeled_Data_modeling/AAPL_modeling_data.csv',index_col = 0)
AMZN_df=pd.read_csv('Labeled_Data_modeling/AMZN_modeling_data.csv',index_col = 0)
IBM_df=pd.read_csv('Labeled_Data_modeling/IBM_modeling_data.csv',index_col = 0)
MSFT_df=pd.read_csv('Labeled_Data_modeling/MSFT_modeling_data.csv',index_col = 0)
TSLA_df=pd.read_csv('Labeled_Data_modeling/TSLA_modeling_data.csv',index_col = 0)
```

APPLE

In [313…

```
AAPL_df.head()
```

Out[313…

	Date	Close/Last	Volume	Open	High	Low	SameDay_Binary	PreviousDay_Binary	SameDay_Percentage	PreviousDay_Percenta
0	2021-04-07	127.900	83466720.0	125.83	127.920	125.1400	1	0	1	
1	2021-04-08	130.360	88844590.0	128.95	130.390	128.5200	1	1	1	
2	2021-04-09	132.995	106686700.0	129.80	133.040	129.4700	1	1	1	
3	2021-04-10	131.635	89321380.0	133.73	133.925	131.1425	0	0	-1	
4	2021-04-11	131.635	89321380.0	133.73	133.925	131.1425	0	0	-1	

In [314…

```
AAPL_df.shape
```

Out[314…

```
(172, 10)
```

In [315…

```
AAPL_df["day"] = AAPL_df['Date'].map(lambda x: pd.to_datetime(x).day)
AAPL_df["month"] = AAPL_df['Date'].map(lambda x: pd.to_datetime(x).month)
AAPL_df["year"] = AAPL_df['Date'].map(lambda x: pd.to_datetime(x).year)
```

In [316…

```
AAPL_df.head()
```

Out[316…

	Date	Close/Last	Volume	Open	High	Low	SameDay_Binary	PreviousDay_Binary	SameDay_Percentage	PreviousDay_Percenta
0	2021-04-07	127.900	83466720.0	125.83	127.920	125.1400	1	0	1	
1	2021-04-08	130.360	88844590.0	128.95	130.390	128.5200	1	1	1	
2	2021-04-09	132.995	106686700.0	129.80	133.040	129.4700	1	1	1	
3	2021-04-10	131.635	89321380.0	133.73	133.925	131.1425	0	0	-1	
4	2021-04-11	131.635	89321380.0	133.73	133.925	131.1425	0	0	-1	

In [317…

```
AAPL_X_classification=AAPL_df[['day','Open']]
AAPL_X_regression=AAPL_df.drop(["Open","Close/Last","SameDay_Binary","PreviousDay_Binary","SameDay_Percentage","P
AAPL_Y_Binary_sameDay = AAPL_df["SameDay_Binary"]
AAPL_Y_Binary_previousDay = AAPL_df["PreviousDay_Binary"]
AAPL_Y_Percentage_sameDay = AAPL_df["SameDay_Percentage"]
AAPL_Y_Percentage_previousDay = AAPL_df["PreviousDay_Percentage"]
```

In [318…

```
print("Classification")
print(AAPL_X_classification.head(1))
print()
print("Regression")
print(AAPL_X_regression.head(1))
print()
print("Binary Same day")
print(AAPL_Y_Binary_sameDay.head(1))
print()
print("Binary Previous Day")
print(AAPL_Y_Binary_previousDay.head(1))
print()
print("Percentage Same Day")
print(AAPL_Y_Percentage_sameDay.head(1))
print()
print("Percentage Previous Day")
print(AAPL_Y_Percentage_previousDay.head(1))
```

```
Classification
   day    Open
0    7  125.83

Regression
       Volume    High     Low  day  month
0  83466720.0  127.92  125.14    7      4

Binary Same day
0    1
Name: SameDay_Binary, dtype: int64

Binary Previous Day
0    0
Name: PreviousDay_Binary, dtype: int64

Percentage Same Day
0    1
Name: SameDay_Percentage, dtype: int64

Percentage Previous Day
0   -1
Name: PreviousDay_Percentage, dtype: int64
```

In [319…

```
AAPL_df.isna().sum()
```

Out[319…

```
Date                      0
Close/Last                0
Volume                    0
Open                      0
High                      0
Low                       0
SameDay_Binary            0
PreviousDay_Binary        0
SameDay_Percentage        0
PreviousDay_Percentage    0
day                       0
month                     0
year                      0
dtype: int64
```

In [320…

```
AAPL_df.isnull().sum()
```

Out[320…

```
Date                      0
Close/Last                0
Volume                    0
Open                      0
High                      0
Low                       0
SameDay_Binary            0
PreviousDay_Binary        0
SameDay_Percentage        0
PreviousDay_Percentage    0
day                       0
month                     0
year                      0
dtype: int64
```

In [321…

```
AAPL_X_classification_sameDay_train, AAPL_X_classification_sameDay_test, AAPL_Y_Binary_sameDay_train,AAPL_Y_Binar
```

In [322…

```
AAPL_X_classification_previousDay_train, AAPL_X_classification_previousDay_test, AAPL_Y_Binary_previousDay_train,
```

In [323…
```
AAPL_X_regression_sameday_train, AAPL_X_regression_sameday_test, AAPL_Y_Percentage_sameDay_train, AAPL_Y_Percenta
```

In [324…
```
AAPL_X_regression_previousday_train, AAPL_X_regression_previousday_test, AAPL_Y_Percentage_previousDay_train, AAP
```

Naive Bayes on Apple Dataset

Prediction On Same Day Approach

In [302…
```
AAPL_sameDay_model = GaussianNB()
AAPL_sameDay_model.fit(AAPL_X_classification_sameDay_train,AAPL_Y_Binary_sameDay_train)
AAPL_sameDay_pred = Aapl_sameDay_model.predict(AAPL_X_classification_sameDay_test)
```

Accuracy for same day approach

In [303…
```
print("Accuray Score: ",AAPL_sameDay_model.score(AAPL_X_classification_sameDay_test,AAPL_Y_Binary_sameDay_test))
```

```
Accuray Score:  0.6571428571428571
```

Confusion Matrix For Same Day

In [304…
```
print("Confusion Matrix: \n",confusion_matrix(AAPL_Y_Binary_sameDay_test, AAPL_smaeDay_pred))
print("\n")
# Classification Report
matrix = classification_report(AAPL_Y_Binary_sameDay_test, AAPL_sameDay_pred)
print("Classification Report For Same day: \n",matrix)
```

```
Confusion Matrix: 
 [[21  2]
 [10  2]]


Classification Report For Same day: 
               precision    recall  f1-score   support

           0       0.68      0.91      0.78        23
           1       0.50      0.17      0.25        12

    accuracy                           0.66        35
   macro avg       0.59      0.54      0.51        35
weighted avg       0.62      0.66      0.60        35
```

In [305…
```
#Extracting TN,FP,FN,TN
AAPL_sameDay_TN,AAPL_sameDay_FP,AAPL_sameDay_FN,AAPL_sameDay_TP = confusion_matrix(AAPL_Y_Binary_sameDay_test, AA
(AAPL_sameDay_TN,AAPL_sameDay_FP,AAPL_sameDay_FN,AAPL_sameDay_TP)
```

Out[305…
```
(21, 2, 10, 2)
```

Accuracy For Same Day with Confusion Matrix

In [306…
```
(AAPL_sameDay_TP + AAPL_sameDay_TN)/len(AAPL_X_classification_sameDay_test)
```

Out[306…
```
0.6571428571428571
```

Prediction on Previous Day Approach

In [307…
```
AAPL_previousDay_model = GaussianNB()
AAPL_previousDay_model.fit(AAPL_X_classification_previousDay_train,AAPL_Y_Binary_previousDay_train)
AAPL_previousDay_pred = Aapl_previousDay_model.predict(AAPL_X_classification_previousDay_test)
```

Accuracy

In [308…
```
print("The accuracy is: ",AAPL_previousDay_model.score(AAPL_X_classification_previousDay_test,AAPL_Y_Binary_previ
```

```
The accuracy is:  0.6857142857142857
```

Confusion Matrix For Previous Day

In [309…

```
print("Confusion Matrix: \n",confusion_matrix(AAPL_Y_Binary_previousDay_test, AAPL_previousDay_pred))
print("\n")
# Classification Report
matrix = classification_report(AAPL_Y_Binary_previousDay_test, AAPL_previousDay_pred)
print("Classification Report For Previous day: \n",matrix)
```

```
Confusion Matrix:
 [[18  4]
 [ 7  6]]


Classification Report For Previous day:
               precision    recall  f1-score   support

           0       0.72      0.82      0.77        22
           1       0.60      0.46      0.52        13

    accuracy                           0.69        35
   macro avg       0.66      0.64      0.64        35
weighted avg       0.68      0.69      0.68        35
```

Conclusion

Accuracy on the previous Day approach is higher

AMAZON

In [270…

```
AMZN_df["day"] = AMZN_df['Date'].map(lambda x: pd.to_datetime(x).day)
AMZN_df["month"] = AMZN_df['Date'].map(lambda x: pd.to_datetime(x).month)
AMZN_df["year"] = AMZN_df['Date'].map(lambda x: pd.to_datetime(x).year)
```

In [271…

```
AMZN_X_classification=AMZN_df[['day','Open']]
AMZN_X_regression=AMZN_df.drop(["Open","Close/Last","SameDay_Binary","PreviousDay_Binary","SameDay_Percentage","P
AMZN_Y_Binary_sameDay = AMZN_df["SameDay_Binary"]
AMZN_Y_Binary_previousDay = AMZN_df["PreviousDay_Binary"]
AMZN_Y_Percentage_sameDay = AMZN_df["SameDay_Percentage"]
AMZN_Y_Percentage_previousDay = AMZN_df["PreviousDay_Percentage"]
```

In [276…

```
AMZN_df.isna().sum()
```

Out[276…

```
Date                      0
Close/Last                0
Volume                    0
Open                      0
High                      0
Low                       0
SameDay_Binary            0
PreviousDay_Binary        0
SameDay_Percentage        0
PreviousDay_Percentage    0
day                       0
month                     0
year                      0
dtype: int64
```

In [277…

```
AMZN_X_classification_sameDay_train, AMZN_X_classification_sameDay_test, AMZN_Y_Binary_sameDay_train, AMZN_Y_Bina
```

In [278…

```
AMZN_X_classification_previousDay_train, AMZN_X_classification_previousDay_test, AMZN_Y_Binary_previousDay_train,
```

In [279…

```
AMZN_X_regression_sameday_train, AMZN_X_regression_sameday_test, AMZN_Y_Percentage_sameDay_train, AMZN_Y_Percenta
```

In [280…

```
AMZN_X_regression_previousday_train, AMZN_X_regression_previousday_test, AMZN_Y_Percentage_previousDay_train, AMZ
```

Naive Bayes on Amazon Dataset

Prediction On Same Day Approach

In [281…

```
AMZN_sameDay_model = GaussianNB()
AMZN_sameDay_model.fit(AMZN_X_classification_sameDay_train,AMZN_Y_Binary_sameDay_train)
AMZN_sameDay_pred = AMZN_sameDay_model.predict(AMZN_X_classification_sameDay_test)
```

Accuracy for same day approach

In [282…

```
print("Accuray Score: ",AMZN_sameDay_model.score(AMZN_X_classification_sameDay_test,AMZN_Y_Binary_sameDay_test))
```

```
Accuray Score:  0.6730769230769231
```

Confusion Matrix For Same Day

In [284…

```
print("Confusion Matrix: \n",confusion_matrix(AMZN_Y_Binary_sameDay_test, AMZN_sameDay_pred))
print("\n")
# Classification Report
matrix = classification_report(AMZN_Y_Binary_sameDay_test, AMZN_sameDay_pred)
print("Classification Report For Same day: \n",matrix)
```

```
Confusion Matrix:
 [[35  1]
 [16  0]]


Classification Report For Same day:
               precision    recall  f1-score   support

           0       0.69      0.97      0.80        36
           1       0.00      0.00      0.00        16

    accuracy                           0.67        52
   macro avg       0.34      0.49      0.40        52
weighted avg       0.48      0.67      0.56        52
```

In [327…

```
#Extracting TN,FP,FN,TN
AMZN_sameDay_TN,AMZN_sameDay_FP,AMZN_sameDay_FN,AMZN_sameDay_TP = confusion_matrix(AMZN_Y_Binary_sameDay_test, AM
(AMZN_sameDay_TN,AMZN_sameDay_FP,AMZN_sameDay_FN,AMZN_sameDay_TP)
```

Out[327…

```
(35, 1, 16, 0)
```

Prediction on Previous Day Approach

In [328…

```
AMZN_previousDay_model = GaussianNB()
AMZN_previousDay_model.fit(AMZN_X_classification_previousDay_train,AMZN_Y_Binary_previousDay_train)
AMZN_previousDay_pred = AMZN_previousDay_model.predict(AMZN_X_classification_previousDay_test)
```

Accuracy

In [329…

```
print("The accuracy is: ",AMZN_previousDay_model.score(AMZN_X_classification_previousDay_test,AMZN_Y_Binary_previ
```

```
The accuracy is:  0.4230769230769231
```

Confusion Matrix For Previous Day

In [331…

```
print("Confusion Matrix: \n",confusion_matrix(AMZN_Y_Binary_previousDay_test, AMZN_previousDay_pred))
print("\n")
# Classification Report
matrix = classification_report(AMZN_Y_Binary_previousDay_test, AMZN_previousDay_pred)
print("Classification Report For Previous day: \n",matrix)
```

```
Confusion Matrix:
 [[20 10]
 [20  2]]


Classification Report For Previous day:
               precision    recall  f1-score   support
```

```
           0       0.50      0.67      0.57        30
           1       0.17      0.09      0.12        22

    accuracy                           0.42        52
   macro avg       0.33      0.38      0.34        52
weighted avg       0.36      0.42      0.38        52
```

Conclusion

Accuracy on the Same Day approach is higher

IBM

In [404…

```
IBM_df.head()
```

Out[404…

	Date	Close/Last	Volume	Open	High	Low	SameDay_Binary	PreviousDay_Binary	SameDay_Percentage	PreviousDay_Percenta
0	2021-04-07	134.93	2976136.0	133.8400	134.9400	133.780	1	0	1	
1	2021-04-08	135.12	4087228.0	134.5700	135.6299	134.160	1	1	0	
2	2021-04-09	135.73	3023916.0	134.8700	135.7400	134.710	1	1	1	
3	2021-04-10	133.61	4811004.0	133.1625	134.0750	132.185	1	0	0	
4	2021-04-11	133.61	4811004.0	133.1625	134.0750	132.185	1	0	0	

In [405…

```
IBM_df["day"] = IBM_df['Date'].map(lambda x: pd.to_datetime(x).day)
IBM_df["month"] = IBM_df['Date'].map(lambda x: pd.to_datetime(x).month)
IBM_df["year"] = IBM_df['Date'].map(lambda x: pd.to_datetime(x).year)
```

In [406…

```
IBM_X_classification=IBM_df[['day','Open']]
IBM_X_regression=IBM_df.drop(["Open","Close/Last","SameDay_Binary","PreviousDay_Binary","SameDay_Percentage","Pre
IBM_Y_Binary_sameDay = IBM_df["SameDay_Binary"]
IBM_Y_Binary_previousDay = IBM_df["PreviousDay_Binary"]
IBM_Y_Percentage_sameDay = IBM_df["SameDay_Percentage"]
IBM_Y_Percentage_previousDay = IBM_df["PreviousDay_Percentage"]
```

In [407…

```
IBM_df.isnull().sum()
```

Out[407…

```
Date                      0
Close/Last                0
Volume                    0
Open                      0
High                      0
Low                       0
SameDay_Binary            0
PreviousDay_Binary        0
SameDay_Percentage        0
PreviousDay_Percentage    0
day                       0
month                     0
year                      0
dtype: int64
```

In [408…

```
IBM_X_classification_sameDay_train, IBM_X_classification_sameDay_test, IBM_Y_Binary_sameDay_train, IBM_Y_Binary_s
```

In [409…

```
IBM_X_classification_previousDay_train, IBM_X_classification_previousDay_test, IBM_Y_Binary_previousDay_train,IBM
```

In [410…

```
IBM_X_regression_sameday_train, IBM_X_regression_sameday_test, IBM_Y_Percentage_sameDay_train, IBM_Y_Percentage_s
```

In [411…

```
IBM_X_regression_previousday_train, IBM_X_regression_previousday_test, IBM_Y_Percentage_previousDay_train, IBM_Y_
```

Naive Bayes on IBM Dataset

Prediction On Same Day Approach

In [412…

```
IBM_sameDay_model = GaussianNB()
IBM_sameDay_model.fit(IBM_X_classification_sameDay_train,IBM_Y_Binary_sameDay_train)
IBM_sameDay_pred = IBM_sameDay_model.predict(IBM_X_classification_sameDay_test)
```

Accuracy for same day approach

In [413…

```
print("Accuray Score: ",IBM_sameDay_model.score(IBM_X_classification_sameDay_test,IBM_Y_Binary_sameDay_test))
```

```
Accuray Score:  0.6346153846153846
```

Confusion Matrix For Same Day

In [414…

```
print("Confusion Matrix: \n",confusion_matrix(IBM_Y_Binary_sameDay_test, IBM_sameDay_pred))
print("\n")
# Classification Report
matrix = classification_report(IBM_Y_Binary_sameDay_test, IBM_sameDay_pred)
print("Classification Report For Same day: \n",matrix)
```

```
Confusion Matrix:
 [[10 11]
 [ 8 23]]


Classification Report For Same day:
               precision    recall  f1-score   support

           0       0.56      0.48      0.51        21
           1       0.68      0.74      0.71        31

    accuracy                           0.63        52
   macro avg       0.62      0.61      0.61        52
weighted avg       0.63      0.63      0.63        52
```

In [415…

```
#Extracting TN,FP,FN,TN
IBM_sameDay_TN,IBM_sameDay_FP,IBM_sameDay_FN,IBM_sameDay_TP = confusion_matrix(IBM_Y_Binary_sameDay_test, IBM_sam
(IBM_sameDay_TN,IBM_sameDay_FP,IBM_sameDay_FN,IBM_sameDay_TP)
```

Out[415…

```
(10, 11, 8, 23)
```

Accuracy For Same Day with Confusion Matrix

In [416…

```
(IBM_sameDay_TP + IBM_sameDay_TN)/len(IBM_X_classification_sameDay_test)
```

Out[416…

```
0.6346153846153846
```

Prediction on Previous Day Approach

In [417…

```
IBM_previousDay_model = GaussianNB()
IBM_previousDay_model.fit(IBM_X_classification_previousDay_train,IBM_Y_Binary_previousDay_train)
IBM_previousDay_pred = IBM_previousDay_model.predict(IBM_X_classification_previousDay_test)
```

Accuracy

In [418…

```
print("The accuracy is: ",IBM_previousDay_model.score(IBM_X_classification_previousDay_test,IBM_Y_Binary_previous
```

```
The accuracy is:  0.6346153846153846
```

Confusion Matrix For Previous Day

In [419…

```
print("Confusion Matrix: \n",confusion_matrix(IBM_Y_Binary_previousDay_test, IBM_previousDay_pred))
print("\n")
# Classification Report
```

```
matrix = classification_report(IBM_Y_Binary_previousDay_test, IBM_previousDay_pred)
print("Classification Report For Previous day: \n",matrix)
```

```
Confusion Matrix:
 [[19 13]
 [ 6 14]]


Classification Report For Previous day:
               precision    recall  f1-score   support

           0       0.76      0.59      0.67        32
           1       0.52      0.70      0.60        20

    accuracy                           0.63        52
   macro avg       0.64      0.65      0.63        52
weighted avg       0.67      0.63      0.64        52
```

Conclusion

Accuracy on the previous Day and same day approach is same

MICROSOFT

In [420…

```
MSFT_df.head()
```

Out[420…

	Date	Close/Last	Volume	Open	High	Low	SameDay_Binary	PreviousDay_Binary	SameDay_Percentage	PreviousDay_Percentag
0	2021-04-07	249.90	22719840.0	247.8100	250.930	247.19	1	0	1	
1	2021-04-08	253.25	23625200.0	252.7700	254.139	252.00	1	1	0	
2	2021-04-09	255.85	24326830.0	252.8700	255.990	252.44	1	1	1	
3	2021-04-10	255.75	25109805.0	256.0925	258.250	254.89	0	0	0	
4	2021-04-11	255.75	25109805.0	256.0925	258.250	254.89	0	0	0	

In [421…

```
MSFT_df["day"] = MSFT_df['Date'].map(lambda x: pd.to_datetime(x).day)
MSFT_df["month"] = MSFT_df['Date'].map(lambda x: pd.to_datetime(x).month)
MSFT_df["year"] = MSFT_df['Date'].map(lambda x: pd.to_datetime(x).year)
```

In [422…

```
MSFT_X_classification=MSFT_df[['day','Open']]
MSFT_X_regression=MSFT_df.drop(["Open","Close/Last","SameDay_Binary","PreviousDay_Binary","SameDay_Percentage","P
MSFT_Y_Binary_sameDay = MSFT_df["SameDay_Binary"]
MSFT_Y_Binary_previousDay = MSFT_df["PreviousDay_Binary"]
MSFT_Y_Percentage_sameDay = MSFT_df["SameDay_Percentage"]
MSFT_Y_Percentage_previousDay = MSFT_df["PreviousDay_Percentage"]
```

In [423…

```
MSFT_df.isna().sum()
```

Out[423…

```
Date                      0
Close/Last                0
Volume                    0
Open                      0
High                      0
Low                       0
SameDay_Binary            0
PreviousDay_Binary        0
SameDay_Percentage        0
PreviousDay_Percentage    0
day                       0
month                     0
year                      0
dtype: int64
```

In [371…

```
MSFT_X_classification_sameDay_train, MSFT_X_classification_sameDay_test, MSFT_Y_Binary_sameDay_train, MSFT_Y_Bina
```

In [372…

```
MSFT_X_classification_previousDay_train, MSFT_X_classification_previousDay_test, MSFT_Y_Binary_previousDay_train,
```

In [373…

```
MSFT_X_regression_sameday_train,MSFT_X_regression_sameday_test, MSFT_Y_Percentage_sameDay_train, MSFT_Y_Percentag
```

In [374…

```
MSFT_X_regression_previousday_train, MSFT_X_regression_previousday_test, MSFT_Y_Percentage_previousDay_train, MSF
```

Naive Bayes on Microsoft Dataset

Prediction On Same Day Approach

In [375…

```
MSFT_sameDay_model = GaussianNB()
MSFT_sameDay_model.fit(MSFT_X_classification_sameDay_train,MSFT_Y_Binary_sameDay_train)
MSFT_sameDay_pred = MSFT_sameDay_model.predict(MSFT_X_classification_sameDay_test)
```

Accuracy for same day approach

In [376…

```
print("Accuray Score: ",MSFT_sameDay_model.score(MSFT_X_classification_sameDay_test,MSFT_Y_Binary_sameDay_test))
```

```
Accuray Score:  0.5576923076923077
```

Confusion Matrix For Same Day

In [378…

```
print("Confusion Matrix: \n",confusion_matrix(MSFT_Y_Binary_sameDay_test, MSFT_sameDay_pred))
print("\n")
# Classification Report
matrix = classification_report(MSFT_Y_Binary_sameDay_test, MSFT_sameDay_pred)
print("Classification Report For Same day: \n",matrix)
```

```
Confusion Matrix:
 [[24  5]
 [18  5]]


Classification Report For Same day:
               precision    recall  f1-score   support

           0       0.57      0.83      0.68        29
           1       0.50      0.22      0.30        23

    accuracy                           0.56        52
   macro avg       0.54      0.52      0.49        52
weighted avg       0.54      0.56      0.51        52
```

In [380…

```
#Extracting TN,FP,FN,TN
MSFT_sameDay_TN,MSFT_sameDay_FP,MSFT_sameDay_FN,MSFT_sameDay_TP = confusion_matrix(MSFT_Y_Binary_sameDay_test, MS
(MSFT_sameDay_TN,MSFT_sameDay_FP,MSFT_sameDay_FN,MSFT_sameDay_TP)
```

Out[380…

```
(24, 5, 18, 5)
```

Accuracy For Same Day with Confusion Matrix

In [381…

```
(MSFT_sameDay_TP + MSFT_sameDay_TN)/len(MSFT_X_classification_sameDay_test)
```

Out[381…

```
0.5576923076923077
```

Prediction on Previous Day Approach

In [382…

```
MSFT_previousDay_model = GaussianNB()
MSFT_previousDay_model.fit(MSFT_X_classification_previousDay_train,MSFT_Y_Binary_previousDay_train)
MSFT_previousDay_pred = MSFT_previousDay_model.predict(MSFT_X_classification_previousDay_test)
```

Accuracy

In [383…

```
print("The accuracy is: ",MSFT_previousDay_model.score(MSFT_X_classification_previousDay_test,MSFT_Y_Binary_previ
```

```
The accuracy is:  0.5576923076923077
```

Confusion Matrix For Previous Day

In [384…

```
print("Confusion Matrix: \n",confusion_matrix(MSFT_Y_Binary_previousDay_test, MSFT_previousDay_pred))
print("\n")
# Classification Report
matrix = classification_report(MSFT_Y_Binary_previousDay_test, MSFT_previousDay_pred)
print("Classification Report For Previous day: \n",matrix)
```

```
Confusion Matrix:
 [[20  7]
 [16  9]]


Classification Report For Previous day:
               precision    recall  f1-score   support

           0       0.56      0.74      0.63        27
           1       0.56      0.36      0.44        25

    accuracy                           0.56        52
   macro avg       0.56      0.55      0.54        52
weighted avg       0.56      0.56      0.54        52
```

Conclusion

Accuracy on the previous Day approach and Same day is same

TESLA

In [386…

```
TSLA_df.head()
```

Out[386…

	Date	Close/Last	Volume	Open	High	Low	SameDay_Binary	PreviousDay_Binary	SameDay_Percentage	PreviousDay_Percentag
0	2021-04-07	670.970	26309430.0	687.00	691.3800	667.840	0	0	-1	
1	2021-04-08	683.800	23924330.0	677.38	689.5499	671.645	1	1	1	
2	2021-04-09	677.020	21437090.0	677.77	680.9700	669.430	0	0	0	
3	2021-04-10	717.105	39076550.0	728.20	742.7950	705.060	0	1	-1	
4	2021-04-11	717.105	39076550.0	728.20	742.7950	705.060	0	0	-1	

In [388…

```
TSLA_df["day"] = TSLA_df['Date'].map(lambda x: pd.to_datetime(x).day)
TSLA_df["month"] = TSLA_df['Date'].map(lambda x: pd.to_datetime(x).month)
TSLA_df["year"] = TSLA_df['Date'].map(lambda x: pd.to_datetime(x).year)
```

In [389…

```
TSLA_X_classification=TSLA_df[['day','Open']]
TSLA_X_regression=TSLA_df.drop(["Open","Close/Last","SameDay_Binary","PreviousDay_Binary","SameDay_Percentage","P
TSLA_Y_Binary_sameDay = TSLA_df["SameDay_Binary"]
TSLA_Y_Binary_previousDay = TSLA_df["PreviousDay_Binary"]
TSLA_Y_Percentage_sameDay = TSLA_df["SameDay_Percentage"]
TSLA_Y_Percentage_previousDay = TSLA_df["PreviousDay_Percentage"]
```

In [390…

```
TSLA_df.isna().sum()
```

Out[390…

```
Date                      0
Close/Last                0
Volume                    0
Open                      0
High                      0
Low                       0
SameDay_Binary            0
PreviousDay_Binary        0
SameDay_Percentage        0
PreviousDay_Percentage    0
day                       0
```

```
month                        0
year                         0
dtype: int64
```

In [391…
```
TSLA_X_classification_sameDay_train, TSLA_X_classification_sameDay_test, TSLA_Y_Binary_sameDay_train, TSLA_Y_Bina
```

In [392…
```
TSLA_X_classification_previousDay_train, TSLA_X_classification_previousDay_test, TSLA_Y_Binary_previousDay_train,
```

In [393…
```
TSLA_X_regression_sameday_train, TSLA_X_regression_sameday_test, TSLA_Y_Percentage_sameDay_train, TSLA_Y_Percenta
```

In [394…
```
TSLA_X_regression_previousday_train, TSLA_X_regression_previousday_test, TSLA_Y_Percentage_previousDay_train, TSL
```

Naive Bayes on Tesla Dataset

Prediction On Same Day Approach

In [395…
```
TSLA_sameDay_model = GaussianNB()
TSLA_sameDay_model.fit(TSLA_X_classification_sameDay_train,TSLA_Y_Binary_sameDay_train)
TSLA_sameDay_pred = TSLA_sameDay_model.predict(TSLA_X_classification_sameDay_test)
```

Accuracy for same day approach

In [396…
```
print("Accuray Score: ",TSLA_sameDay_model.score(TSLA_X_classification_sameDay_test,TSLA_Y_Binary_sameDay_test))
```

```
Accuray Score:  0.6538461538461539
```

Confusion Matrix For Same Day

In [398…
```
print("Confusion Matrix: \n",confusion_matrix(TSLA_Y_Binary_sameDay_test, TSLA_sameDay_pred))
print("\n")
# Classification Report
matrix = classification_report(TSLA_Y_Binary_sameDay_test, TSLA_sameDay_pred)
print("Classification Report For Same day: \n",matrix)
```

```
Confusion Matrix: 
 [[28  4]
 [14  6]]


Classification Report For Same day: 
               precision    recall  f1-score   support

           0       0.67      0.88      0.76        32
           1       0.60      0.30      0.40        20

    accuracy                           0.65        52
   macro avg       0.63      0.59      0.58        52
weighted avg       0.64      0.65      0.62        52
```

In [399…
```
#Extracting TN,FP,FN,TN
TSLA_sameDay_TN,TSLA_sameDay_FP,TSLA_sameDay_FN,TSLA_sameDay_TP = confusion_matrix(TSLA_Y_Binary_sameDay_test, TS
(TSLA_sameDay_TN,TSLA_sameDay_FP,TSLA_sameDay_FN,TSLA_sameDay_TP)
```

Out[399…
```
(28, 4, 14, 6)
```

Accuracy For Same Day with Confusion Matrix

In [400…
```
(TSLA_sameDay_TP + TSLA_sameDay_TN)/len(TSLA_X_classification_sameDay_test)
```

Out[400…
```
0.6538461538461539
```

Prediction on Previous Day Approach

In [401…
```
TSLA_previousDay_model = GaussianNB()
TSLA_previousDay_model.fit(TSLA_X_classification_previousDay_train,TSLA_Y_Binary_previousDay_train)
```

```
TSLA_previousDay_pred = TSLA_previousDay_model.predict(TSLA_X_classification_previousDay_test)
```

Accuracy

In [402…

```
print("The accuracy is: ",TSLA_previousDay_model.score(TSLA_X_classification_previousDay_test,TSLA_Y_Binary_previ
```

```
The accuracy is:  0.5
```

Confusion Matrix For Previous Day

In [403…

```
print("Confusion Matrix: \n",confusion_matrix(TSLA_Y_Binary_previousDay_test, TSLA_previousDay_pred))
print("\n")
# Classification Report
matrix = classification_report(TSLA_Y_Binary_previousDay_test, TSLA_previousDay_pred)
print("Classification Report For Previous day: \n",matrix)
```

```
Confusion Matrix:
 [[26  0]
 [26  0]]


Classification Report For Previous day:
               precision    recall  f1-score   support

           0       0.50      1.00      0.67        26
           1       0.00      0.00      0.00        26

    accuracy                           0.50        52
   macro avg       0.25      0.50      0.33        52
weighted avg       0.25      0.50      0.33        52
```

```
C:\Users\husey\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1221: UndefinedMetricWarning: Preci
sion and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` pa
rameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

Conclusion

Accuracy on the Same Day approach is higher

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js