untitled11-1

May 6, 2024

[1]:
```
import pandas as pd
df= pd.read_csv('/content/archive ass.zip')
print(df)
```

```
            Date         Open         High          Low        Close  \
0     2010-06-29     3.800000     5.000000     3.508000     4.778000
1     2010-06-30     5.158000     6.084000     4.660000     4.766000
2     2010-07-01     5.000000     5.184000     4.054000     4.392000
3     2010-07-02     4.600000     4.620000     3.742000     3.840000
4     2010-07-06     4.000000     4.000000     3.166000     3.222000
…            …            …            …            …            …
2951  2022-03-18   874.489990   907.849976   867.390015   905.390015
2952  2022-03-21   914.979980   942.849976   907.090027   921.159973
2953  2022-03-22   930.000000   997.859985   921.750000   993.979980
2954  2022-03-23   979.940002  1040.699951   976.400024   999.109985
2955  2022-03-24  1009.729980  1024.489990   988.799988  1013.919983

        Adj Close    Volume
0        4.778000  93831500
1        4.766000  85935500
2        4.392000  41094000
3        3.840000  25699000
4        3.222000  34334500
…             …         …
2951   905.390015  33408500
2952   921.159973  27327200
2953   993.979980  35289500
2954   999.109985  40225400
2955  1013.919983  22901900

[2956 rows x 7 columns]
```

[5]:
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error ,mean_squared_error,␣
 ↪median_absolute_error,confusion_matrix,accuracy_score,r2_score
from sklearn.linear_model import LinearRegression ,ARDRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import RandomForestRegressor
```

```
from sklearn.linear_model import Ridge
from sklearn.svm import SVR
```

[12]:
```
X = df.drop(columns="Close")
y = df["Close"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,␣
 ↪random_state=42)
print("X Train : ", X_train.shape)
print("X Test  : ", X_test.shape)
print("Y Train : ", y_train.shape)
print("Y Test  : ", y_test.shape)
```

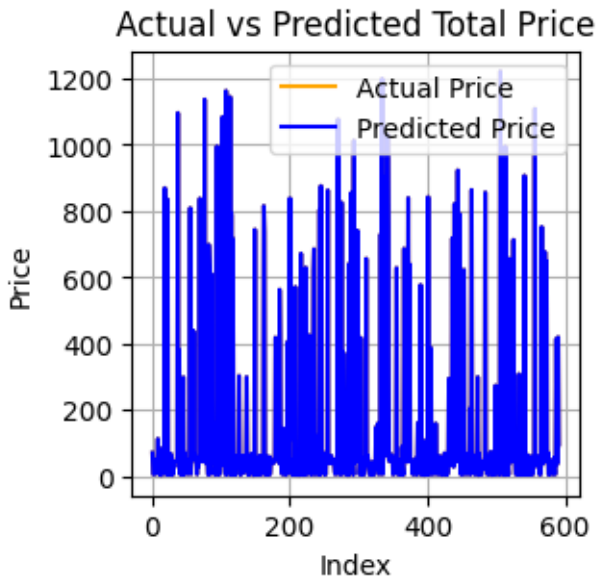
```
X Train :  (2217, 6)
X Test  :  (739, 6)
Y Train :  (2217,)
Y Test  :  (739,)
```

[28]:
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error ,mean_squared_error,␣
 ↪median_absolute_error,confusion_matrix,accuracy_score,r2_score
from sklearn.linear_model import LinearRegression ,ARDRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import Ridge
from sklearn.svm import SVR
X = df[['Open', 'High', 'Low', 'Adj Close', 'Volume']]
y = df['Close']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=0)

models = [
    LinearRegression(),
    RandomForestRegressor(),
    KNeighborsRegressor(n_neighbors=5),
    Ridge(alpha=1.0, random_state=42),
    SVR()
]
for model in models:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    mean_actual = y_test.mean()
    mse_percentage = (mse / mean_actual) * 100
    mape = mean_absolute_error(y_test, y_pred) / mean_actual * 100
    rmse=np.sqrt(mean_squared_error(y_test, y_pred))
```

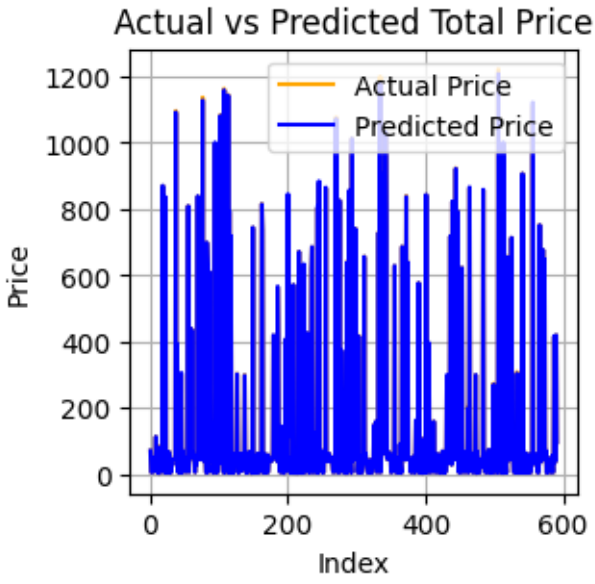
```
    print(model,f"R-squared: {r2}")
    print(model,f"Root Mean Squared Error (RMSE): {rmse}")
    print(model,f"Mean Squared Error: {mse}")
    print(model,f"Mean Absolute Error (MAE): {mae}")
    print(model,f"Mean Absolute Error Percentage (MAPE): {mape:.2f}%")
    print(model,f"Mean Squared Error Percentage: {mse_percentage:.2f}%")
    print(model,f"R-squared: {r2}")
    import matplotlib.pyplot as plt
    plt.figure(figsize=(3, 3))
    plt.plot(y_test.values, color='orange', label='Actual Price')
    plt.plot(y_pred, color='blue', label='Predicted Price')
    plt.xlabel('Index')
    plt.ylabel('Price')
    plt.title('Actual vs Predicted Total Price')
    plt.legend()
    plt.grid(True)
    plt.show()
```

```
LinearRegression() R-squared: 1.0
LinearRegression() Root Mean Squared Error (RMSE): 1.9722369296243927e-12
LinearRegression() Mean Squared Error: 3.889718506574252e-24
LinearRegression() Mean Absolute Error (MAE): 1.2785860958574846e-12
LinearRegression() Mean Absolute Error Percentage (MAPE): 0.00%
LinearRegression() Mean Squared Error Percentage: 0.00%
LinearRegression() R-squared: 1.0
```

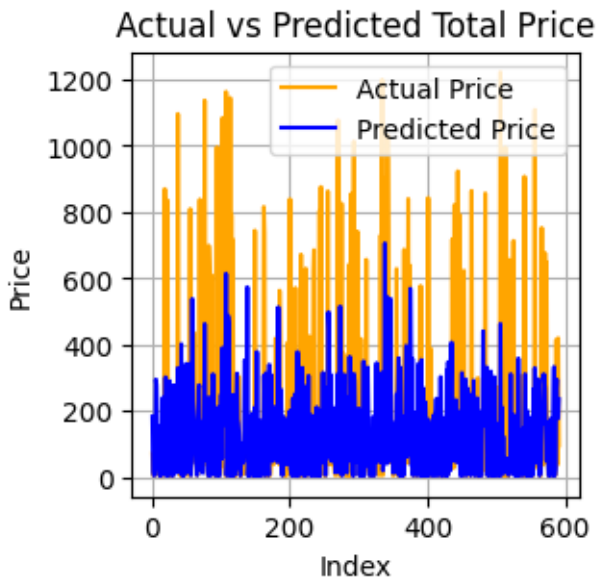



```
RandomForestRegressor() R-squared: 0.9999589567711437
RandomForestRegressor() Root Mean Squared Error (RMSE): 1.6771257192600226
```

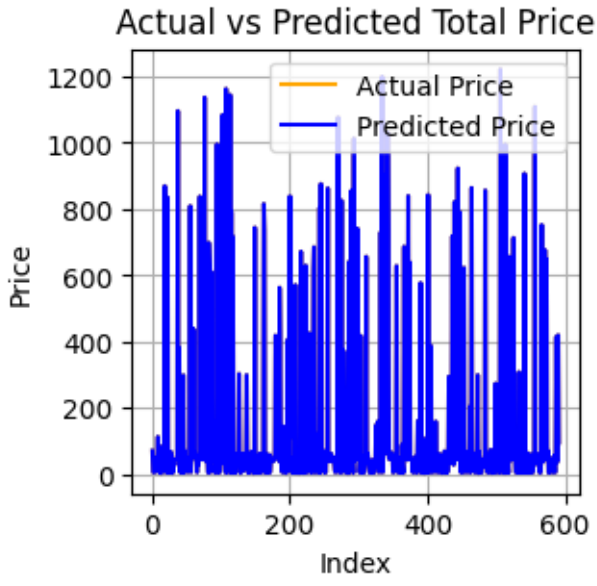
```
RandomForestRegressor() Mean Squared Error: 2.812750678203448
RandomForestRegressor() Mean Absolute Error (MAE): 0.4647565066047183
RandomForestRegressor() Mean Absolute Error Percentage (MAPE): 0.32%
RandomForestRegressor() Mean Squared Error Percentage: 1.91%
RandomForestRegressor() R-squared: 0.9999589567711437
```




```
KNeighborsRegressor() R-squared: 0.022284108680276082
KNeighborsRegressor() Root Mean Squared Error (RMSE): 258.8518042462109
KNeighborsRegressor() Mean Squared Error: 67004.25656151869
KNeighborsRegressor() Mean Absolute Error (MAE): 154.95178142263515
KNeighborsRegressor() Mean Absolute Error Percentage (MAPE): 105.45%
KNeighborsRegressor() Mean Squared Error Percentage: 45599.33%
KNeighborsRegressor() R-squared: 0.022284108680276082
```




```
Ridge(random_state=42) R-squared: 0.9999999999998596
Ridge(random_state=42) Root Mean Squared Error (RMSE): 9.810192856505214e-05
Ridge(random_state=42) Mean Squared Error: 9.623988388182592e-09
Ridge(random_state=42) Mean Absolute Error (MAE): 3.811183654131503e-05
Ridge(random_state=42) Mean Absolute Error Percentage (MAPE): 0.00%
Ridge(random_state=42) Mean Squared Error Percentage: 0.00%
Ridge(random_state=42) R-squared: 0.9999999999998596
```



```
SVR() R-squared: -0.12378511839715034
SVR() Root Mean Squared Error (RMSE): 277.51502272572236
SVR() Mean Squared Error: 77014.5878384582
SVR() Mean Absolute Error (MAE): 117.39413582086844
SVR() Mean Absolute Error Percentage (MAPE): 79.89%
SVR() Mean Squared Error Percentage: 52411.80%
SVR() R-squared: -0.12378511839715034
```