202480-Fall 2024-ITCS-3162-051-Introduction to Data Mining: Project 3

Sean Kiser
10/18/2024

Kaggle Dataset Link:
https://www.kaggle.com/datasets/zongaobian/microsoft-stock-data-and-key-affiliated-companies

Introduction to Microsoft Stock Dataset:
I am using the Microsoft Stock Data Set. This data set has 9714 rows and 7 columns. These columns include date, open, high, low, close, adjacent close, and volume. Just to give a little more insight into stocks I will describe what each of these columns mean. Date is the year, month and date of a particular stock's price. Open is the price of the stock when the market opens. High is the highest price of the stock on the given day. Low is the lowest price of the stock on a given day. Close is the price of the stock when the market closes. Adjacent close is the price of the stock after the market is closed based on adjustments. Volume is the number of shares traded on a given day. This data set will be used in order to create a linear regression model in order to predict the price of Microsoft Stock.

What is Linear Regression?
Linear regression is a type of model that is used to fit a line to a set of data points. The method used to fit a line to the data is called the least squares method. The line is fit in order to minimize the distance from the points to the line. Then after a line is fit to the data, the p value is calculated. The p value is how well the line fits the data. An example of a linear relationship is miles on a car and price. This has a negative linear relationship because as the car gets more miles the price of the car goes down.

Preprocessing the Data:
The data from this data set is already pretty clean. There are no duplicate values and there are also no null values. For my purpose of using this data I will need to create new columns with previous days data in order to train the linear regression model. For example I will need to make columns for previous days close, previous days open, previous days volume and so on. I will also

need to decide what I want to use as my target for training the model. In order to do what I described this is the code that I did. I also decided to do a whole month shift and year shift.

```
df['Prev_Open'] = df['Open'].shift(1)
df['Prev_High'] = df['High'].shift(1)
df['Prev_Low'] = df['Low'].shift(1)
df['Prev_Close'] = df['Close'].shift(1)
df['Prev_Adj_Close'] = df['Adj Close'].shift(1)
df['Prev_Volume'] = df['Volume'].shift(1)
```

Because of this it created one null value for each of the new columns. In order to clean this I chose to just drop those columns since it will not affect the model that much.

Understanding the Data:

Here are the correlations for two data frames. One is the data frame that uses previous days data and the other uses previous years data. The one thing that I would like to highlight is the difference in correlation between the two dataframes

Previous days data:

	Open	High	Low	Close	Adj Close	Volume	Prev_Open	Prev_High	Prev_Low	Prev_Close	Prev_Adj_Close	Prev_Volume
Open	1.000000	0.999948	0.999941	0.999875	0.998902	-0.359792	0.999812	0.999877	0.999885	0.999929	0.998956	-0.349091
High	0.999948	1.000000	0.999925	0.999937	0.998929	-0.358818	0.999792	0.999865	0.999837	0.999888	0.998883	-0.348473
Low	0.999941	0.999925	1.000000	0.999946	0.999002	-0.360943	0.999754	0.999802	0.999844	0.999873	0.998934	-0.349790
Close	0.999875	0.999937	0.999946	1.000000	0.999023	-0.359971	0.999709	0.999767	0.999773	0.999808	0.998841	-0.349175
Adj Close	0.998902	0.998929	0.999002	0.999023	1.000000	-0.359871	0.998739	0.998761	0.998832	0.998836	0.999822	-0.348935
Volume	-0.359792	-0.358818	-0.360943	-0.359971	-0.359871	1.000000	-0.359468	-0.358827	-0.360314	-0.359689	-0.359613	0.598823
Prev_Open	0.999812	0.999792	0.999754	0.999709	0.998739	-0.359468	1.000000	0.999948	0.999941	0.999875	0.998900	-0.349017
Prev_High	0.999877	0.999865	0.999802	0.999767	0.998761	-0.358827	0.999948	1.000000	0.999925	0.999937	0.998927	-0.348076
Prev_Low	0.999885	0.999837	0.999844	0.999773	0.998832	-0.360314	0.999941	0.999925	1.000000	0.999946	0.999000	-0.350128
Prev_Close	0.999929	0.999888	0.999873	0.999808	0.998836	-0.359689	0.999875	0.999937	0.999946	1.000000	0.999022	-0.349188
Prev_Adj_Close	0.998956	0.998883	0.998934	0.998841	0.999822	-0.359613	0.998900	0.998927	0.999000	0.999022	1.000000	-0.348935
Prev_Volume	-0.349091	-0.348473	-0.349790	-0.349175	-0.348935	0.598823	-0.349017	-0.348076	-0.350128	-0.349188	-0.348935	1.000000

Previous years data:

	Open	High	Low	Close	Adj Close	Volume	PrevYear_Open	PrevYear_High	PrevYear_Low	PrevYear_Close	PrevYear_Adj_Close
Open	1.000000	0.999947	0.999940	0.999873	0.998958	-0.378257	0.954151	0.954623	0.953869	0.954260	0.957068
High	0.999947	1.000000	0.999924	0.999936	0.998987	-0.377237	0.954597	0.955059	0.954316	0.954698	0.957464
Low	0.999940	0.999924	1.000000	0.999945	0.999059	-0.379466	0.953942	0.954413	0.953660	0.954048	0.956888
Close	0.999873	0.999936	0.999945	1.000000	0.999081	-0.378446	0.954207	0.954674	0.953929	0.954313	0.957122
Adj Close	0.998958	0.998987	0.999059	0.999081	1.000000	-0.377869	0.953410	0.953855	0.953153	0.953517	0.958354
Volume	-0.378257	-0.377237	-0.379466	-0.378446	-0.377869	1.000000	-0.349722	-0.349524	-0.350012	-0.349776	-0.353136
PrevYear_Open	0.954151	0.954597	0.953942	0.954207	0.953410	-0.349722	1.000000	0.999931	0.999924	0.999838	0.998105
PrevYear_High	0.954623	0.955059	0.954413	0.954674	0.953855	-0.349524	0.999931	1.000000	0.999898	0.999919	0.998152
PrevYear_Low	0.953869	0.954316	0.953660	0.953929	0.953153	-0.350012	0.999924	0.999898	1.000000	0.999923	0.998220
PrevYear_Close	0.954260	0.954698	0.954048	0.954313	0.953517	-0.349776	0.999838	0.999919	0.999923	1.000000	0.998267
PrevYear_Adj_Close	0.957068	0.957464	0.956888	0.957122	0.958354	-0.353136	0.998105	0.998152	0.998220	0.998267	1.000000
PrevYear_Volume	-0.331272	-0.331413	-0.331279	-0.331407	-0.331297	0.188919	-0.321469	-0.320042	-0.323045	-0.321628	-0.322695

Experiment 1 Without PCA and Standardization:

I first use the data that uses the previous days data in order to create my model. This is the code that I used for this:

```
X = df.drop(columns=['Close' , 'Open', 'High', 'Adj Close', 'Low', 'Volume', 'Date'], axis=1)
y = df['Close']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=42)
```

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((5827, 6), (3886, 6), (5827,), (3886,))
```

```
day_model = SGDRegressor()
```

```
day_model.fit(X_train, y_train)
```

```
SGDRegressor
SGDRegressor()
```

I then used mean squared error, root mean squared error, and r-squared to evaluate my model and here are the results:

```
mse
```

```
3.355053165558419e+47
```

```
r2
```

```
-3.7618600057065626e+43
```

```
rmse
```

```
5.792282076658922e+23
```

All of these numbers are very large and show that the model is not performing well at all. The model will have to be adjusted so that it will perform better.

Experiment 2 With Standardization:
This time when creating the model we made sure to standardize the data. Now after standardizing the data and training the model here are the results:

```
mse
```

```
4.309250453807612
```

```
r2
```

```
0.99951682443953
```

```
rmse
```

```
2.0758734195050557
```

As you can see this is a much better result. Let's go a little more in depth and explain what each of these numbers mean. The value 4.31 for mean squared error means that the average squared difference between predicted stock prices and the actual value is 4.31. The value 0.9995 for R-Squared means that this model does a very good job at explaining variability in stock prices. The value 2.08 for root mean squared error means that on average the model deviates 2.08 dollars from the actual stock price. Now lets do it with PCA to see if it can improve the model.

Experiment 3 With Standardization and PCA:
The final experiment will be using PCA on top of this to see if we can make the model more accurate. Here are the results:

```
mse
```

```
4.271231679648142
```

```
r2
```

```
0.9995210873021113
```

```
rmse
```

```
2.0666958362681584
```

As you can see we get very similar results so PCA didn't change too much this time. This likely makes a lot of sense as there are not that many columns used in this model. We would see a lot more of a difference if we were using more columns.

Impact:
This project shows that machine learning models can be used in order to predict stock prices. In fact in the real world people do use models to do so. Something to note though is that it is impossible to fully predict the stock market. Factors are always changing so one can never fully predict it for 100%. Another thing to note is that this is not by any means the best model for predicting the stock market. Currently the most prevalent model that is used for predicting stock prices is LSTM models (Long Short Term Memory). Also another thing to consider is that models for predicting the stock markets constantly have to be tweaked. As factors change it is super important that the model also changes.

Conclusion:
In this project I learned how to create a basic linear model. I also learned how to standardize my model as well as use PCA to reduce the dimensions. I also learned how to shift data in order to create columns with previous days data. Standardization definitely helped my models performance but PCA did not have a very noticeable impact. This likely has to do with the fact that my data didn't have that many columns to begin with.