INTERNATIONAL TECHNOLOGICAL UNIVERSITY

CSC 520 PYTHON- SIGNATURE ASSIGMENT

'Analyzing and predicting stock market trend of an individual stock via constructing Python computer program'

MERVE OZMEN

December,2018

Abstract

Forecasting is the process of predicting the future values based on historical data and analyzing the trend of current data. By running simulations of future states based on present states, we can foresee the trend of stock market. Linear Regression is a powerful statistic methodology modeling the relationship between a scalar dependent variable y or more explanatory variables denoted X.

Key words: linear regression, stock market prediction

Approaches

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs. Simply put, it does some extremely complex data transformations, then figures out how to separate your data based on the labels or outputs you've defined. However, our case, SVM made us low accuracy for historical data. Making analyzing with low confidence, it does not get good prediction, that is why, this study does not apply this method.

Linear Regression is a powerful statistic methodology modeling the relationship between a scalar dependent variable y or more explanatory variables denoted X. Regression predicts a numerical value. The relations which regression establishes between predictor and target values can make a pattern. This pattern can be used on other datasets which their target values are not known. Therefore, the data needed for regression are two-part, first section for defining model and the other for testing model. In this section we choose linear regression for our analysis. First, we divide the data into two parts of training and testing. Then we use the training section for starting analysis and defining the model. 80% data used for training purpose and 20% data used for testing purpose.

Data Representation

The dataset that was used was collected from finance.yahoo.com as a collection of separated values where each row consisted of a stock on a specific day along with data on the open price, high price, low price, close price adjusts close price, volume. The Python scientific computing library numpy was used along with the data analysis library pandas in order to convert

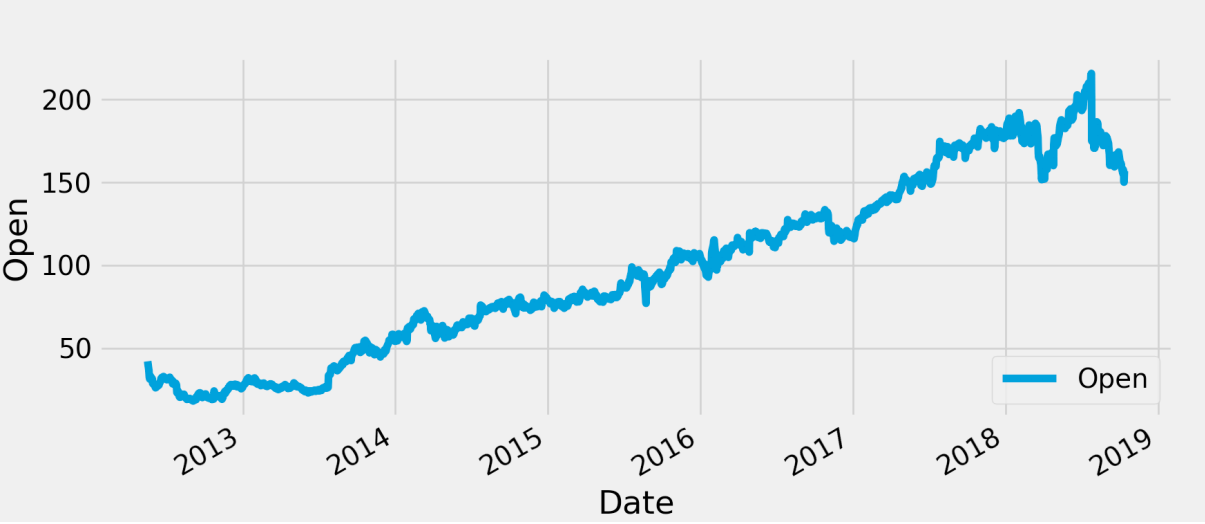
these CSV files into pandas DataFrames that were indexed by date. This allowed efficient access to stocks of interest and convenient access to date ranges.

Summary method provides us, some statistical value on database.

```
def summary(self, col='Adj Close'):
    maximum = np.max(self.df[col])
    minimum = np.min(self.df[col])

    max_day = self.df[self.df[col] == maximum].index[0].strftime("%Y-%m-%d")
    min_day = self.df[self.df[col] == minimum].index[0].strftime("%Y-%m-%d")

    return maximum, max_day, minimum, min_day
```

According to five years data, maximum closing amount is 217.5 when was 2018-07-25, minimum maximum closing amount is 17.73, the date was 2012-09-04. Our data consist of many hills and valleys.

This graph shows that opening price in our dataFrame.

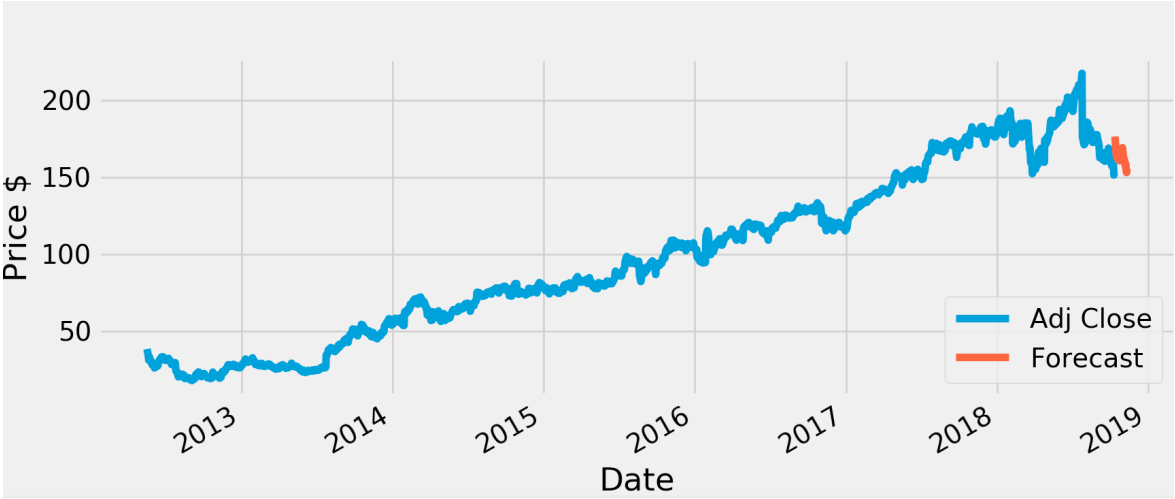


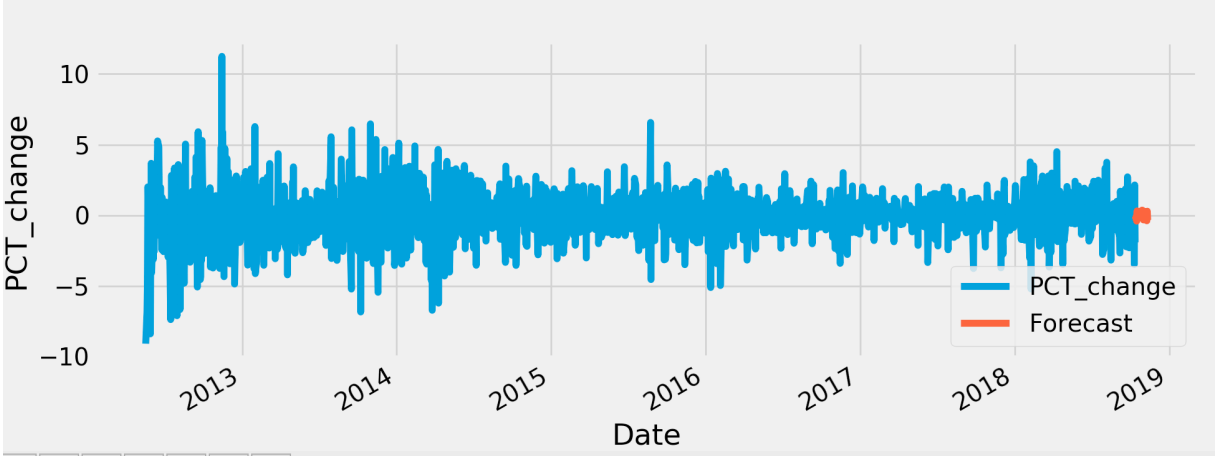
Analyzing

After organizing DataFrame, confidence of data 0.96 with linear regression analysis and 0.92 with support vector machine. So, in order to get healthier foresting, I prefer making prediction in linear regression rather than SVM. Using prediction function is below:

```
def forecasting(self, day=None, column='Adj Close'):
    # Check day value
    if day is None:
        day = self.forecast_day

    if not column in self.df.columns:
        print("This column is not exist!")
        return {}

    # Prepare Dataframe for forecasting
    # Fill holes with -9999
    self.df.fillna(-9999, inplace=True)

    # Move forecasting column's data to new column by shifting forecasting row count
    self.df['label'] = self.df[column].shift(-day)
    x = np.array(self.df.drop(['label'], 1))
    x = preprocessing.scale(x)
    x = x[:-day]
    x_lately = x[-day:]

    self.df.dropna(inplace=True)
    y = np.array(self.df['label'])
    y = np.array(self.df['label'])

    # Split dataframe as train and test data for x and y axis
    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)

    # Run Regression method over data
    clf = LinearRegression(n_jobs=-1)
    # clf = svm.SVR()
    clf.fit(x_train, y_train)
    accuracy = clf.score(x_test, y_test)

    forecast_set = clf.predict(x_lately)
    self.df['Forecast'] = np.nan

    # Get maximum date value
    last_date = self.df.iloc[-1].name
    last_unix = last_date.timestamp()
    next_unix = last_unix + Stock.one_day

    # Append predicted values to  DataFrame
    for i in forecast_set:
        next_date = datetime.datetime.fromtimestamp(next_unix)
        next_unix += Stock.one_day
        self.df.loc[next_date] = [np.nan for _ in range(len(self.df.columns) - 1)] + [i]

    return {'values': forecast_set, 'accuracy': accuracy, 'days': day}
```

This analyze is very flexible, you can change your any input to get this input's result and also you can change your prediction day. This flexibility comes from OOP, there is some results about prediction.





These graphs show that, price and change of percentage after prediction. The orange color shows us prediction values in 30.

```
def plot(self, label='Price', col='Adj Close', forecast=False):
    style.use('fivethirtyeight')
    self.df[col].plot()
    if forecast:
        self.df['Forecast'].plot()

    plt.legend(loc=4)
    plt.xlabel('Date')
    plt.ylabel(label)
    plt.show()
```

Plot method provides us to show plot. Variables are defined as a soft code so you can draw any plot for any variable. Also ‘forecast=False’ or ‘forecast=True’ provides us to get plot before forecast or after forecast.

```
if __name__ == "__main__":
    stock = Stock("FB", "/Users/merveozmen/PycharmProjects/untitled5/FB.csv")
    print(stock.get_columns()[0])
    print(stock.forecasting(30,'PCT_change'))
    stock.plot(label='Forecast', col='Forecast', forecast=False)
    stock.plot(label='Open', col=stock.get_columns()[0], forecast=False)
    stock.plot(label='PCT_change', col='PCT_change', forecast=True)
    print(stock.summary())
```

The main function also makes us job easier. We can see how this project is flexible.

Conclusion

The aim of our research study is to help understanding how to use some python libraries, packages and how to implement an algorithm. As per the discussed works above our system, predicts the stock prices based on Linear regression, and Support Vector Machine regression approach using variables. We compare these methods on bases of confidence value and analyzed that linear regression provide best result compare another method.

Preferences

- Introduction to linear regression downloaded pdf file from http://google.com.
- Data sources from http://yahoofinance.com