Kapil Bastola

May 06, 2017

Final Presentation Teleprompter Script

Cover Slide

For the capstone project I worked on the traffic dataset from the City Pulse website. My goal was to be able to forecast road traffic flow based on historical traffic. I used supervised learning and time series analysis to get to the goal.

Contribution of Competitor’s Article

The competitor article I picked was “Short-Term Trafﬁc Flow Forecasting: An Experimental Comparison of Time-Series Analysis and Supervised Learning.” The authors first reviewed existing approaches to traffic flow forecasting the common view of probabilistic graphical models, presenting an extensive experimental comparison, which proposes a common baseline for their performance analysis and provides the infrastructure to operate on a publicly available data set. Then the authors provide two new support vector regression models, which are speciﬁcally devised to beneﬁt from typical trafﬁc ﬂow seasonality and are shown to represent an interesting compromise between prediction accuracy and computational efﬁciency. The competitor’s article compares the accuracy using metric called mean absolute percentage error or MAPE.

Description of Your Contribution

I follow the competitor’s footstep in doing supervised learning as well as time series analysis to forecast traffic flow. I wanted to improve on the accuracy of the results and was successful in doing so as you will see in subsequent slides. I will also compare my model by cross validating the models created using accuracy metrics and visualize results to find the model that performs the best.

Data Source and Content

As I mentioned earlier, the dataset consists of data from traffic in a city called Aarhus in Denmark. This dataset is collection of traffic data between two points for certain duration of time in CSV format for different durations. A CSV metadata file is also available that provides additional information regarding the different two points. Upon further analysis it was found that the dataset has 9 columns.

The status column has “OK” in all of the first 15 lines that we pulled. we wanted to see if there are any other values for this field. we looked at all the distinct values in the status column and saw that “OK” is the only value in the whole column.

avgMeasuredTime column has numerical values. The two sensors on two points of the road measure how long it took a vehicle in seconds to reach the second point from the first point. This field gives us the mean of total time taken in seconds by different vehicles to reach from the first point to the second point for each reading.

avgSpeed column also consists of numerical values. This column provides the average speed of vehicles between the two points in kilometer per hour (kmh).

extID column consists of 3-4 digit numerical values. Initially we was not certain what this column represented so we looked at all the distinct values in the column. we saw that they are sequential numbers and there are total of 449 distinct values in the table. Since we had 449 total files in our dataset, extID is a unique identifier for each file.

medianMeasuredTime column also consists of numerical values and has similar values as the avgMeasuredTime in the first 15 data points we looked at. This column gives us the median of total time taken in seconds by different vehicles to travel between the first and the second point for each reading.

TIMESTAMP column consists of date and time values and gives us the date and time of each reading.

vehicleCount column consists of numerical values. For each reading there are multiple vehicles passing between the two points. This column gives us the number of vehicles that travel between the two points during the readings.

\_id column consists of numerical values as well. In the first 15 rows, there are 6 digit numerical values that are all different. we wanted to count the distinct number of values in the \_id column and saw that the number of distinct values in this column is equal to the count of rows in the table. This suggests that \_id is the unique identifier for each row of data.

REPORT\_ID column consists of numerical values as well. In the first 15 rows, all the values were same in this column. Hence, we looked at the count of distinct values and found that there are 449 total unique values in the column. This suggests that this is an identifier for each file as well. In the metadata provided in the website we see that there is a column with same name and the values as the report\_id in the dataset. Hence, report\_id can be used to join the data set with the metadata file to obtain more information on each of the reading that took place.

The metadata file is a single .csv file with more information on the data streams. It has 449 rows of data implying that each row corresponds to each file in the dataset. The metadata file has information on where exactly the two points were. It contains information like street, city, latitude, longitude, postal code, and country for the two points. Apart from this it also contains ext\_id and REPORTID. These columns were present in the data set as well. Upon further look, ext\_id in the metadata file did not match the ext\_id present in the datasets but the REPORTID were same in the both file. So we can use the REPORTID in the metadata file with the REPORTID in the dataset to join the two tables if the need arises.

Your Method

For my method I first loaded my data in Zeppelin using Spark, while combining all the different CSV files. I then looked at the descriptive statistics of the datasets using SQL and R. I proceeded to split the dataset into training and test data sets for cross validation and evaluation of models to be created. Next, I build a decision tree supervised model and two different types of Auto Regressive Integrated Moving Average (ARIMA) models in R using Zeppelin. For the two ARIMA models, one was a normal ARIMA without any seasonality while the other model included seasonality on the moving average. I decided to use seasonality by looking at the line graph of the time-series, which showed seasonal patterns in the data. The first time series conducted had an auto regressive order of 4, differentiating factor of 1 and moving average order of 4. The seasonal time series model had auto regressive factor of 0, differentiating factor of 2 and moving average order of 7 along with a moving average seasonal order of 18. I looked at the Mean Squared error for my decision trees, which was very high. Since there were very few variables that could be used I decided to focus on the time-series analysis. For the two time-series analyses conducted I compared my results using mean absolute percentage error (MAPE), which my competitors use as well.

Quantitative Results 1

For the results, we got a decision tree as shown in the picture above. First node of the decision tree is whether the vehicle count is higher than 0.5 or not. If it is higher than 0.5, average speed is 47. If the vehicle count is lower than 0.5 we encounter another node, which examines whether time of day is greater than 4 am or not. If the time of day is less than 4 am, average speed is 46. If the time of day is greater than 5 am we encounter another node that examines whether time of day is after 6 pm or not. If time of day is after 6 pm the predicted average speed is 42 kph and if it is before 6 pm and after 5 am, the predicted average speed is 33 kph. We looked at the error of this model and saw that the average MSE is 478.

Quantitative Results 2

Next, we wanted to do time series analysis on the average speed. We first aggregated the 20 million rows data into hourly data for the time series analysis. After sorting the data by day and hour, we have an ordered tie series data that can be used for time series analysis. Using R and forecast library, we use auto.arima function to run an auto regressive integrated moving average (ARIMA) model to create a fit that can be used to predict future data. In order to use the ARIMA model we need to provide which level we want to set the model at. We can utilize just the auto-regression (AR) part of the model by setting to a numerical value or we can just set the moving average (MA) portion of the model. Auto.arima function in the forecast library selected an order of 4 for AR, 1 for I, and an order of 4 for MA. We got a Mean Absolute Percentage Error (MAPE) of 2.52. We also predicted the last 12 values of the time series. When comparing the actual versus the predicted and the variation in the actual speed is not as prevalent in the predicted speed. Also looking at the Ljung-Box test for significance, we see that there are first few lags that are not significant.

We wanted to pick our own orders for AR and MA by looking at the ACF and PACF. Looking at the ACF and PACF we see that there is some seasonality in the data and decided to add a seasonality component to the ARIMA and performed an ARIMA order of (0,2,7) with a seasonality of (0,0,18). Using these orders, we reran the ARIMA fit and predicted the last 12 hours of average speed of the time series again. For this model we got an MAPE of 2.19, which is an improvement from the previous time series model. For the 12 hours of predicted average speed versus the actual average speed we see that the variation in the actual values is represented in the predicted values as well as seen in the figure below. When looking at the Ljung-Box test for significance, we see that all the lags has p-values less than or close to 0.05 meaning a statistically significant result. This tells us that the seasonal ARIMA is a better model compared to the non-seasonal one.

Discussion: Comparison with your competitor

While comparing with the competitor article, the table from the competitor article shows that the lowest mean absolute percentage error they got was for a Seasonal ARIMA or SARIMA model with MAPE of 5.0. Both of our time series models performed better than the best Seasonal ARIMA from the competitor. For our first model we got an MAPE of 2.52 and the next model we got an MAPE of 2.19. Both MAPE of models that we created were better than the best MAPE of the competitor’s model.

Performance on Big Data: Time measurement

Loading raw csv data files in spark took 52 seconds and there were 20 million records. Loading dataset in R took 1 minutes and 31 seconds. I wanted to aggregate 5-minute data to an hourly data and this aggregation in R took 4 mins and 18 seconds to complete. Running descriptive statistics on the data only took 4 minutes while running the decision tree with all 20 million records took me about 3 mins and 5 secs. The non-seasonal time series took me about 32 seconds while the seasonal time series model took 4 mins and 1 sec to run. Getting the count of all rows in the data files took 18 secs to run. Plotting the result from non-seasonal time series model took less than a sec while the one for the seasonal time series model took 2 secs to run.

Conclusion

As the competitor we also found that Seasonal Auto Regressive Integrated Moving Average (SARIMA) model is best model to predict traffic flow as we observed the least amount of mean absolute percentage error using this model. Using supervised learning models like the decision trees was not the best way to approach the dataset we had because of the lack of variables and we observed a high error in the decision tree’s result. The seasonal ARIMA and ARIMA model that we built had a much lower MAPE compared to the competitor. Since the dataset the competitor’s article used and we used are different with different variables and attributes, it may be difficult to compare the two results but only looking at how the model we built predicted the results compared to the actual values, we can say that seasonal auto regressive integrated moving average (SARIMA) is the best predictor of traffic flow.