Presentation Notes

Safiuddin Mohammed
Brandon Pover
Trevor Moos
Vivek M
Shawn K

KNN

Briefly, these were the steps followed in implementing the KNN algorithm.

1. **Data Imported and Cleaned.**
2. **Label Encoding**. In the dataset we find 5 categorical variables. Since we need to calculate Euclidean distance between variables, we need to encode the values for each of the 5 categorical variables. Label Encoding is a process where each unique item in the categorical variable column is assigned an integer value. This allows us to include the categorical variables in our model and analysis.
3. **Scaling**. To avoid the magnitude of one variable diminishing the value of another variable due to units used, it becomes important to scale the data. For Example, Distance is in 100s of miles whereas delay is measured in minutes. We use the normalization technique to do this as shown below.

$$z = \frac{x - \mu}{\sigma}$$

4. **Data sampling and split.** The Data is split into 75: 25 Train : Test ratio. The final 25% of records are used to calculate the final RMSE value.
5. **Feature Selection**. There are a total of 21 features in the data set, which imputes a total combination of 2^{21} combinations. Since this is exponential, it is critical to rationally choose the features. We use the correlation of the features with the taxi-out variable to shortlist a combination of features, by adding one feature at a time to test the RMSE.
6. **Cross-validate for each combination of features** - Note minimum *RMSE* and tuning parameter *k*. Finalize feature combination with minimum RMSE and finalize k.
7. **Run against test data, with finalised tuning parameter and feature combination**. Note the final OOS RMSE.

Boosting Regression

The first step of the data prep was to convert some of the columns to the appropriate types, specifically the columns that should be factors, such as wind, destination, etc. Next was to drop the tail number column as it had too many unique values and the boosting model would not be able to handle it. Once the data was cleaned and able to be handled by the boosting model, the data was split into a training, validation, and test set, with a 50/25/25 split respectively. This is to ensure that the model is generalizing the best with new data. With the split in place the boosting model was run on a combination of tuning parameters which were chosen mostly arbitrarily, but gave the feel of a least complex, middle, and most complex value. Once the model was finished running, the best parameters were chosen and then again ran against the test set to get the final RMSE of 5.98. With this, the important variables in the boosting model could also be graphed and considered, where it found that the destination was the most important variable.

Bagging Regression

Began by removing Tail_Num and DEST variables from data. Random forest function cannot calculate factors with over 53 levels (Did not have statistical significance as well). Create a rough model to see what is the optimal range of trees when plotted against MSE. Found that there is minimal difference between the number of trees after 100. This was confirmed in the model testing as seen on slide 14 where the model with only 100 trees had the lowest test RMSE. Then, the variable importance was created and evaluated. As seen on slide 15, the airline carrier variable (OP_UNIQUE_CARRIER), the weather conditions, and the scheduled departure time had the greatest impact when predicting taxi out time. We found that airline carriers can vary greatly in their efficiency as seen on slide 16. Correspondingly, when using all the variables as required by bagging, the lowest test MSE found was 6.279551.

Random Forest Regression

- Started with cleaning Dew.Point variable
- Encoded factor variables using Label and One-hot encoding, put every variable in the model and looked for important variables
- Continued the process by eliminating TAIL_NUM and DEST as they didn't show much significance. Eliminated MONTH, DAY_OF MONTH, DAY_OF_WEEK as it didn't make much sense in regression
- Train test split : 75/25
- Parameter Tuning : `maxnodes`, `ntrees`, `mtry` using 5-fold cross validation on training data
- OOB sample testing and results
- Important variables : OP_UNIQUE_CARRIER, Condition, sch_dep, wind

Multiple Linear Regression

Began by looking for meaningful predictors. After reading in the file and sifting through some of the data, a simple regression was executed for every quantitative column and the coefficients and F-scores were noted. The two variables furthest from zero were sch_dep and Pressure. These points were plotted against TAXI_OUT in separate plots and abline was used to visualize the slope. Next, mixed selection was performed using these two variables. The highest F score was attained when running multiple regression with sch_dep and sch_arr. Using these two variables gave the best predictions for Y, with a standard error of about 6.7.