Milestone 3: Feature Engineering and Hyperparameter Tuning

1. ***Perform necessary feature engineering on your dataset, describe any:***
   1. ***New features introduced since the last milestone***
   2. ***Updated features and specific transformations performed since the last milestone***

Since our last milestone we identified that our model was still not performing accurately despite using the entirety of our dataset. Although this is more than expected given the spontaneous nature of flight prices, we still believe there is room for improvement.

Therefore, we further engineer our input variables by extending the *preprocess\_variables.ipynb* notebook. This updated feature set will ideally provide us improved performance. Please refer to the notebook for further details.

* **Flight Distance**: Instead or in addition to encoding the starting and ending airport as categorical variables, we introduce a new variable as the straight line distance between the two airports. This variable is a critical addition as
* **Day of Week and Time of Day for Departures and Arrivals**: We add variables to include the day of the week and the time of the day for both departures and arrivals into our model. These could prove essential because of patterns such as flights being more expensive towards the end of the week (as people leave for weekend trips) and towards later in the day (after people get off work).
* **Including a Proxy for Demand (Population)**: The price of a ticket is heavily impacted by the total air traffic passing through a city. For example, a small airport may charge a lot more for a comparable flight because there are very few flights departing. We attempt to proxy this factor using the population of the departure and arrival cities.

1. ***Perform hyperparameter tuning on your dataset with the newly engineered features through your base pipeline from milestone #2***
   1. ***Remember to start small, i.e., on a small subset first then submit it as a .py job in your cloud service provider***
   2. ***Record the hyperparameters and values you experimented with***

We attempt to tune the hyperparameters for the random forest model we created in the last milestone. To this end we utilize K-Fold (specifically, 5-Fold) cross validation and a grid search across the essential parameters of numTrees and maxDepth. *(please see hyperparameter\_tuning.ipynb)*

* For **numTrees**, we started with 50 and incremented by 50 until reaching 200
* For **maxDepth**, we started with 5 and incremented by 5 until reaching 20

We found a new optimal hyperparameter combination by using maxDepth of 15 and numTrees of 50,

This indicates that perhaps our particular dataset benefits from a simpler model. This may make sense because the combination variables to a particular flight are quite limited, and any error may come out more out of the variance of the underlying target, rather than a lack of predictive power of the model. In other words, an over-complex model is likely to overfit the data.

Given sufficient time, we will attempt even more hyperparameters, especially on the lower end, variate the other hyperparameters of the randomForest,

1. ***Summarize and report key findings from this milestone in a word/text file***
   1. ***Insights learned***
   2. ***Analysis of your baseline model performance***

Some of the major insights we have learnt are:

* **Variable Tailoring**: variables that we have used may not represent our data well, but more importantly, we may still not have extracted the “essence” of a variable which will make it most suitable to predicting prices. A good example of this is the date column, where we need to extract the day of the week from the date to make it particularly impactful to the price prediction.
* **Hyperparameters can make a big difference**: We were surprised to see a substantial decrease in RMSE just from optimizing our grid search, we will keep this in mind during our final development.

**DELIVERABLES:**

1. Jupyter notebook and all necessary Python scripts needed to replicate your work
2. Word/text file summarizing your key findings

**Notes**

* Instead of using one hot encoding for starting and destination, use distance, this should simplify the problem whilst giving us a ordinal variable
* Day of the week at which the flight departs
* Day of the week for which the flight lands
* Time of the day for which the flight departs
* Time of the day for which the flight lands
* Size of arrival and departure cities (population). This is a proxy for demand