APM4990: Final Project Grading Outline

Below is the outline for the grading of the final project. Your main goal is to build a model which estimates travel time between two points in New York City.

You will be working with the **New York City Taxi data**:

- **NYC Government page:** https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
- **Big Query:** https://console.cloud.google.com/marketplace/details/city-of-new-york/nyc-tlc-trips?filter=solution-type:dataset&filter=category:encyclopedic

Please only use the data from **2016**.

I highly encourage using Big Query, as it allows you to easily join to other interesting data sets such as weather. Click here for some public data sets available: https://cloud.google.com/bigquery/public-data/.

Broad requirements:

- **Your goal is to build a model which will accurately predict the travel time of a taxi trip in New York City.**
- Each team member has pushed their contributions to Github with their contributions clear in Github. No more than 4 members per team.

Specific Requirements:

- Your model should take as input the `lat,lng` of the `pickup` location and `dropoff` location, in addition to whichever other features you wish to use. It should output an expected trip duration in `seconds`.
- *I will provide you with a final test data set which I will hide the actual travel times from*. You will need to submit predictions on this test set, and we will then compare who has the best performance in the class!
- An iPython notebook which shows your analysis/work.
- The full code base in the same Github repo.
- Provide a link in the projects worksheet to your completed project.
- **Bonus:** Build a basic website that allows you to enter in trip data, and return the estimate (please sign up for a free website here: https://www.pythonanywhere.com/ if interested).

Below is a breakdown of the grading scheme.

Data Gathering and Preparation (35%):

Data gathering/preprocessing:

Data pipeline (5%):

- Did you query the data from Big Query?
- How easy is it to fetch new data and retrain your model?

Data integrity checks (5%):

- Did you account for missing values and outliers?
- Is there information leakage? ie. a variable which is actually inferred by the outcome (eg. predicting a user likes a movie using the fact that they've liked that movie before).
- Are some variables non-sensical or redundant? (ie. if you see "Male" sometimes and "M" other times, or numerical values in the gender column).

Feature Engineering (25%):

- Did you convert categorical features into one hot encoded dummy variables?
- Did you properly transform variables appropriate for the model type? (eg. how do you deal with lat/lngs in a linear model?)
- Was there an opportunity to make a new variable from the old ones that has more predictive power?
- Do you join the data to any other interesting data sets that have useful predictive power?
- Do you standardize variables when necessary?

Model Selection, Comparison and Cross Validation (60%):

Predicting Travel Time

Exploratory Analysis (10%):

- Did you analyze the features and how they are related to the outcome variable? (eg. scatter plots, histograms).
- Did you look at correlations or chi-squared if the variables are categorical? (https://en.wikipedia.org/wiki/Chi-squared_test. But feel free to find a package that does this automatically).

Model Selection (50%):

- Did you randomly split your data into training and testing data (20%, 80%) using k-fold cross validation?

- Did you perform regularization Linear model: Why did you use L^1 or L^2? Decision Tree Models: Which parameters did you tune and why? I expect to see use of GridSearchCV for this with at least 3 fold cross validation.
- Did you try out various models and see which one performed best? Did you plot the comparisons of the models in a way which is digestable to the reader?
- Did you make feature importance plots and discussed their meaning or lack thereof?

Code Quality (5%):

- Is the code well written with comments and descriptions?
- Is the code modular? Have you avoided repeating code and defined helpful functions or classes?

Extra interesting ideas (BONUS 10-20%):

This isn't necessary, but I'm leaving this here to allow for interesting and novel modeling/strategy approaches that I may not have thought of.

- Did you use a novel modelling approach for your problem that required coding something by hand or using a novel approach from a paper?