ORIE 4741

Michelle Chao, Ansh Godha, Pooja Gokhale, Divya Talesra

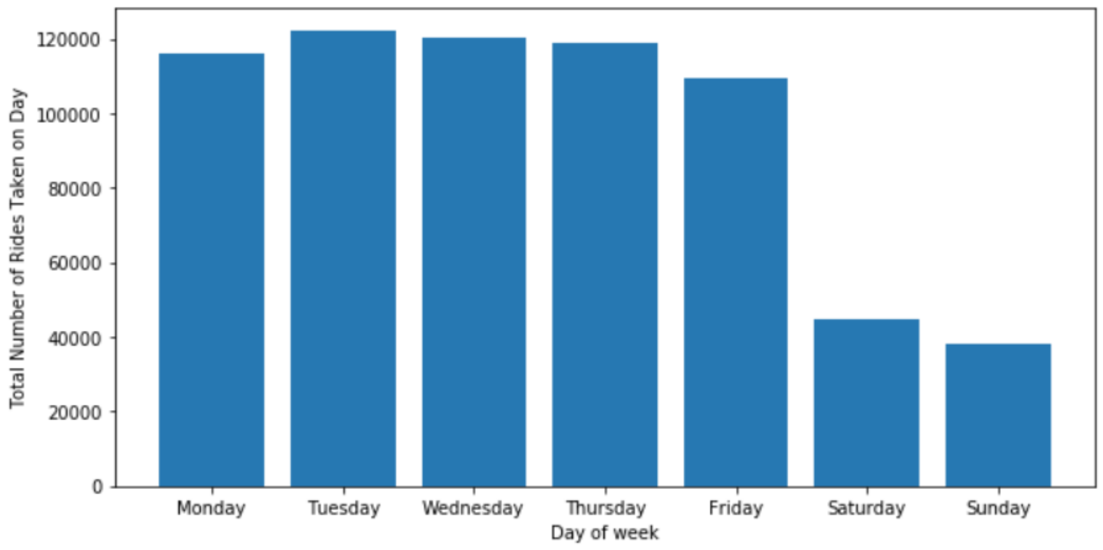
Final Report

We used the SF Bay Area Bike Share dataset. We preprocessed the data into a new dataset, where each data sample will contain the name of a station, a specific hour, whether it is rush hour, day of week, whether it is a holiday, zip code for the station, and also mean information about the weather for that day. Ultimately, we hope to predict the number of bikes that will be rented from a given station in a given hour and the other additional features in the dataset. We believe that these features are relevant to predicting the number of bikes rented from a station; when choosing these features, we looked for factors that we thought would be related to whether or not people would want to rent a bike.

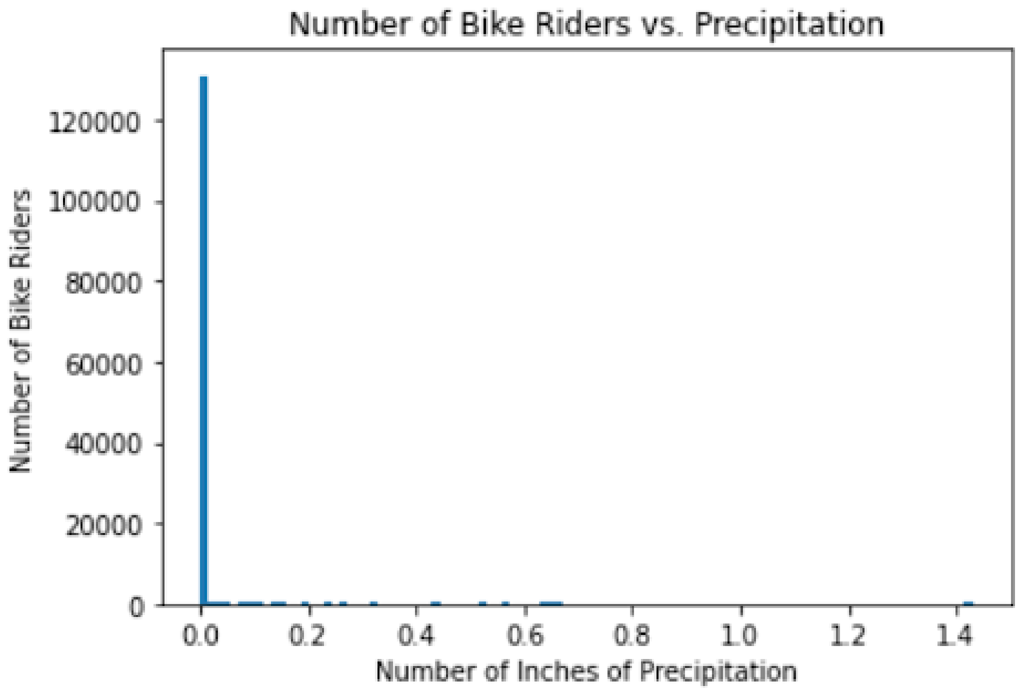
Initial Plots and Graphs of Data

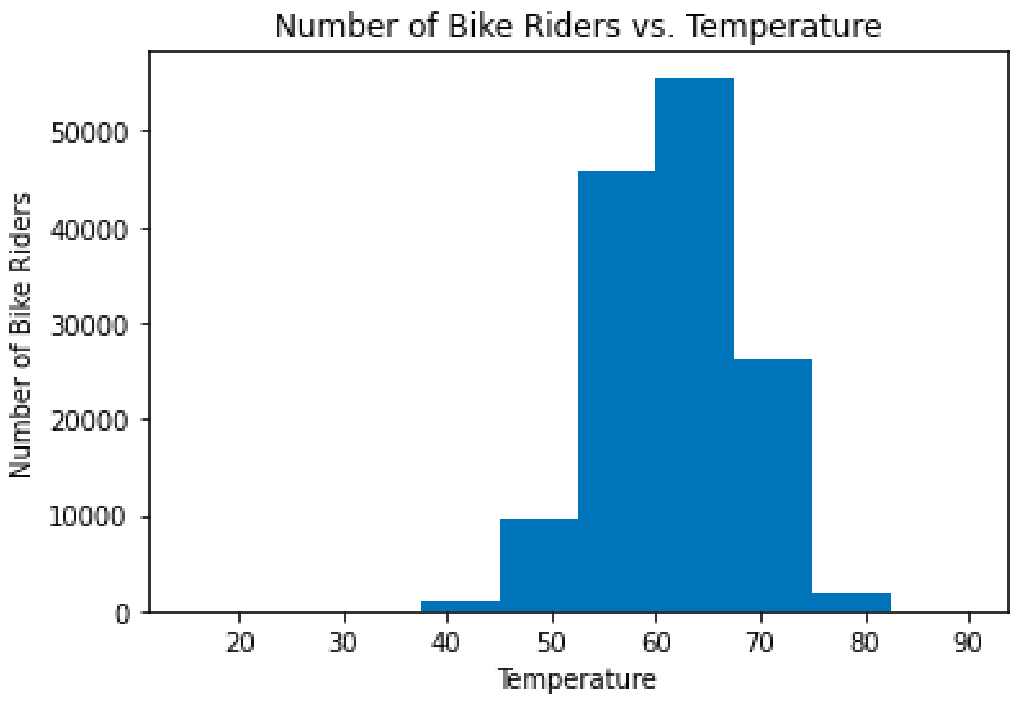
When selecting which features to incorporate into our final data set, there were obvious choices: mean_precipitation, mean_visibility, isRushHour, etc... Clearly, the average inches of precipitation on a particular day would impact the number of bike riders that day, as it would make it much more difficult to go biking. However, in order to ensure the impact of these features on the total number of bike riders, we made some plots that display this correlation.

The following plot demonstrates the correlation between the day of the week and the total number of rides that took place that day. Clearly, the most number of bike rides take place around the middle of the week (Tuesday, Wednesday, Thursday), closely followed by Monday and Friday. Then, there's an evident drop in the number of bike rides during the weekend. Thus, the day of week clearly plays an important factor in determining the number of bikes that would be rented from the station.



Similarly, the following two plots also demonstrate that there is a clear relationship between the number of inches of precipitation and the temperature on the number of bike rides. In the first plot, the number of riders drops dramatically when precipitation ¿ 0. In the second plot, we see a unimodal distribution, and the number of riders appears to be greatest at temperatures of 60-70 and slopes lower when outside of that range.





Missing/Corrupted Data

When initially going through our data, we quickly realized that there were rows where the zip code associated with a single bike ride belonged to Oregon, which clearly was incorrect since the data set was supposed to be representative of Californian bike rides. This was an instance of corrupted data in our data set. Since we discovered that there were multiple rows where the zip codes in station.csv did not match up with the actual name of the station, we used a Google Maps API to generate a correct zip code based on the latitude and longitude of the station. To add on, we also had some missing data in the weather data table. For instance, for the feature called precipitation_inches had a floating number for almost every row, except in some, it simply contained a letter T. To deal with this, we imputed the missing values.

Despite all this, we still had the trouble that our data was distributed across multiple tables. To perform regression analysis using exactly one table, we had to find a way to join everything. Pandas' join function, in addition to the presence of a zip-code column in every csv file allowed us to join all tables into one table, after which we could go forward with modelling.

Imputing Missing Values

Our weather dataset consisted of some missing values; specifically, there were missing values under the precipitation_inches column. As the last step in preprocessing our data, we imputed those missing values. There were a few different ways in which we could have predicted these missing values. To begin with, we could have simply used the other fields in the weather dataset, such as max_dew_point_f, mean_visibility_miles, mean_humidity, mean_temperature_f, etc . . . , and performed linear regression in order to simply predict the missing values in the precipitation_inches column, since our output space in this case would be real numbers. However, we went with another approach, which was the Generalized Low Rank Models (GLRM) approach. Specifically, we used the LowRankModels class in Julia. We first imported the weather data set, and removed the date column, since the data related to dates is irrelevant in determining the number of inches of precipitations. Then, we applied the GLRM library function using the quadratic loss function, the l1 loss function, and quadratic regression on the weather data. We then fit the GLRM for 1000 iterations, where the objective value decreased after each iteration. As a last step, we made the precipitation_inches values to be 0 if it was predicted to be negative.

Results from Regression Modelling

For this milestone, we chose to build various regression models that incorporated different sets of features from our input space. For each of these regression models, we split the dataset into training and testing datasets using Scikit-learn's train_test_split interface. We decided to go for a 67% / 33% train/test split. Then, using sklearn's linear regression models, we fit our dataset, and subsequently logged the error metrics derived from the regression model's predictions. These error metrics included the MSE, MAE and RMSE. As an example, for the first model, we simply included 3 weather-based features - the mean humidity, the mean visibility and the precipitation inches. Our regression model had the following coefficients: [0.02155454 0.00419423 -0.52917649]. These coefficients can be interpreted intuitively as well - the first 2 coefficients have low values, and this makes sense because California's humidity doesn't vary significantly throughout the year. This means that the humidity won't play a role in whether someone chooses to ride a bike. Similarly, visibility doesn't really affect bike-riding on a 'street-level'. Lastly, the coefficient for the 'precipitation_inches' feature is negative because a higher value for that should mean that the number of bike riders decreases (we lose approximately 1 rider for every 2 additional inches of rain).

For our last linear regression model, we chose to include every pertinent feature, with features like the day of the week and the starting station ID being represented by 1-hot vectors. Some important things to note include that the following features have negative coefficients: mean_wind_speed_mph, precipitation_inches, isHoliday, Hour 0, Hour 2, Hour 3, Hour 4, Hour 5, Hour 19, Hour 16, Hour 18, Hour 19, Hour 20, Hour 21, Hour 22, Hour 23, Station 4, Station 5, Station 7, Station 8, Station 27, Station 30, Station 32, Station 61, Station 64, Station 83, Saturday, Sunday, 94041, 94063, while the following features had positive coefficients: isRushHour, Hour 1, Hour 6, Hour 7, Hour 8, Hour 9, Hour 11, Hour 12, Hour 13, Hour 14, Hour 15, Hour 17, Station 3, Station 14, Station 22, Station 28, Station 69, Station 70, Friday, Monday, Thursday, Tuesday, Wednesday, 94107, 95113. This gave us a sense of the 'peak 'days and hours and the 'low' days and hours.

	Test
MSE	10.2
R^2	0.33

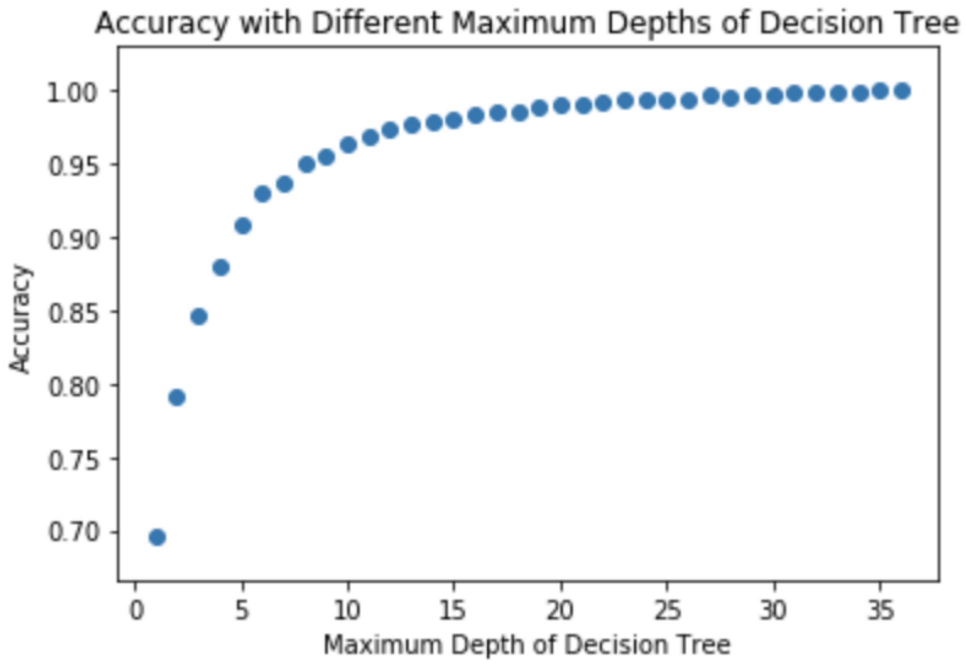
Regularizers

Even though our test error metrics were performing decently well (our test MSEs were in the 10 - 11 range), we thought there was room for improvement. This is why we attempted to use regularization techniques to make sure we get rid of any potential overfitting. Hence, for all the linear regressions we ran above, we also trained our dataset on Ridge and Lasso-regularized models as well (with different alpha coefficients). We found that there was virtually no difference when we included this, and the general performance of our model remained the same. This means that our model was not overfitting a lot to begin with. There is an intuitive explanation behind this as well - our beginning dataset was very skewed, and this means whatever coefficients the model learned led to the outputs being in a very narrow window, which is exactly what we'd expect.

Decision/Classification Trees

We found decent performance of our Linear Regression model, and we also shifted toward predicting the output, number of bikes per station, as a categorical variable. Since our response variable is the set of natural numbers and it mainly ranges from 1 to 50 in our dataset, we thought that decision trees would be an effective model. We briefly covered decision (or classification) trees in the course in section. In broad terms, a decision tree model predicts categorical outputs by splitting the data at each node, starting at the root, until all the data at each leaf node is pure ie. represents a single category. An example of a hyperparameter for decision trees is maximum depth of the tree. For more information about decision trees, see this webpage.

We used all possible columns of our final encoded features to fit the dataset. Without any restrictions on the parameters of the decision tree (ie. number of leaves, maximum depth) we were able to achieve a test accuracy of 0.999937 (one misclassified out of 15914) when we split the data into 2/3 training set, and 1/3 test set. This classifier had a depth of 37 a total of 46 leaves. Although this is high, we saw quite good accuracy when we evaluated the tree on the test set. Even when we restricted the maximum depth to 10, we were still able to see a classification accuracy of 0.964. Below, we plotted the accuracy of the decision tree with many different maximum depths. We trained the decision tree several times, with maximum depths between 1 and 37, and we then calculated the accuracy of the trained classifier. We can see that the accuracy sharply increases as depth goes from 1 to about 12 or 13, and then flattens out toward 1.



The reason for the high accuracy, even at lower maximum depths, is likely due to the concentration of the output. We observed that the majority of values in the response variable are concentrated around integers between 1-10, however the entire response variable has values ranging from 1 to 52. When we looked at the misclassified points, we found that they were most often for when the response had higher values. We would prefer trees with small maximum depth, since these trees are more likely to be able to generalize well on unseen data. Looking at the graph above, we would likely choose 15 to be the maximum depth for our model.

Weapon of Math Destruction

We do not believe that our model or findings from this project could be classified as a Weapon of Math Destruction. In lecture, a Weapon of Math Destruction was defined to be a predictive model whose outcome is not easily measurable, whose predictions can have negative consequences, or a model that creates self fulfilling or defeating feedback loops. We do not think our project fits into any of these categories. Our outcome is easily measurable: it is simply the number of bikes we predict would be rented from a specific station at the given hour and additional information. The predictions would not have negative consequences; the main consequences that we see our model having, which will be discussed later in this report, are positive and would help both the company and users of the bike sharing company.

Fairness

Fairness was not an important criterion in choosing our model. This is primarily because the features that we used did not contain any protected attributes. In other words, all of the features were valid in affecting the total number of bike riders, as there were none on which discrimination was prohibited. For example, we were given the station name, information on the time (ie day and hour), and information on the weather (ie temperature and precipitation). Inclusion of all of these features in the model does not seem to add any bias.

Production

We would not be willing to put our model into production for a few reasons. First of all, some inconsistencies we noticed in the data led us to doubt the validity of the dataset as a whole. For example, we noticed that the zip codes corresponding to stations were incorrect. Luckily, since we had the station names available, we were able to write a script ourselves to fix the zip codes. If there is other corrupt data, however, there was no way for us to notice and fix it. Along the same lines, our dataset may have been slightly skewed, which we did not realize until we were well into our analysis of it, so this could have affected our results. This is evident in the results of our analysis. Essentially, the number of bikers in a specific hour ranged from 1-10 for most of our data, so predictions for these low values were usually accurate. The issue arose when there were a large number of bikers in the hour, and as there was relatively little data on this, our models failed to predict these even remotely accurately. Given a better dataset, our models could have performed a lot better, and perhaps the results of those could be used in production.

Consequences of our Project

We tried our many different techniques learned in ORIE 4741 in this project, namely data preprocessing, linear regression, regularization, imputation, and decision (classification) trees. Our project could help people who are interested in using bike sharing; namely, it may help those who are interested in using bike sharing services to determine whether a station will be crowded, given the weather conditions, type of day, and other factors.

One notable insight from our linear regression analysis was that the coefficient for precipitation was negative; thus, there is a negative relationship between level of precipitation and the number of bikes rented in a specific station in a single hour. This information may be useful to companies as they could use it to innovate new ways to bring in riders even when the weather conditions are not necessarily favorable for biking. For example, bike sharing companies could find ways of providing rain protection or safety measures for riders.

If we had more to work on this project, we would have liked to explore other models as well, such as neural networks.