INFO 290T – Data Mining
Final Project Report
Team: Eukwang Joo, Ryan Jung, Julia Kosheleva-Coats, Divya Menghani
May 14, 2014

SFO or OAK?
Predicting Flight Departure Delays

I. Introduction

Peter Russell, a recent Cal graduate, has started a new career in consulting with Deloitte in their San Francisco office. The job entails significant travel, and his colleagues have advised him to always schedule flights out of Oakland International Airport (OAK) instead of San Francisco International Airport (SFO) due to the higher probability of weather-related delays at SFO.

Conducting some background research, Peter found that the advice from his colleagues seems to be contradicted by data on departure delays at SFO and OAK. According to the Bureau of Transportation Statistics, in 2013 SFO had an on-time departure rate of 76.74% ranking 21st among major US airports. Conversely, according to the Air Travel Consumer Report, Oakland International Airport had an on-time departure rate of 66.0% in 2013[1].

In order to assist Peter, we set out to create a predictive model to answer the question: "Given a specific date and destination, is it better to fly out of OAK or SFO?" Our model would take a specific date, destination, and other input variables to predict the probability of all flight delays from SFO and OAK for that date.

II. Our Approach

a. Data

To conduct this analysis and build a predictive model we obtained publicly-available data on flight arrivals and departures for major U.S. airports from the American Statistical Association website[2]. The dataset contains 29 variables that describe each flight in terms of departure/arrival date and time, carrier, taxi time, time spent in the air, as well as departure and arrival delays and their causes. Although the entire dataset contains flight records from 1987 to 2008, we elected to only work with the data from 2001 through 2008. This ensured variable consistency across the entire dataset since data files for earlier years have variation in variable availability. This choice also helped us avoid El Nino/La Nina years (1988, 1997, 1999)[3] which would introduce bias into a model. The entire dataset is split into individual files – one file per year – and contains around 7 million records each. Each record represents a unique flight record as a combination of a flight number and aircraft tail number.

To reduce the amount of data, we selected to work with the most relevant variables to build our models. Out of the 29 variables, we choose nine:

- **Month, DayofMonth, DayofWeek** – key time variables;
- **DepTime** – scheduled departure time;
- **Origin** – origin airport code;
- **Destination** – destination airport code;
- **UniqueCarrier** – airline carrier code;
- **TailNum** – required unique aircraft registration number similar to a license plate number for motor vehicles;
- **DepDelay** – the length of the departure delay in minutes.

b. Assumptions

To simplify model development process, we made several assumptions:

1. **Ignore delay propagation between connecting flights**: While common in the real-world, modeling flight delay propagation would be relatively complex and might require more advanced machine learning techniques such as Markov chains or state-space modeling.
2. **Ignore effect of cancelled flights**: Since records of cancelled flight do not provide meaningful information about departure delays, they were removed from both training and testing datasets.
3. **Definition of a 'delay'**: We decided to set a departure delay definition threshold at 5 minutes. Thus, all flights with 5-minute or greater difference between scheduled and actual departure time were considered delayed.
4. **Travelers are not price or time sensitive**: We assumed that business travelers – the potential users of our model – are indifferent to price variations between SFO and OAK. Additionally, we assumed that travelers only care about the date that they fly out, and are indifferent to the time of day of their flight. This assumption is mostly focused on model application than model development. The source dataset didn't include pricing information as well.

c. Choosing Predictive Models

Since prediction of a flight delay is a classification problem we considered several algorithms including Decision Trees, Random Forests, Naïve Bayes, SVM, and logistic regression. We also thought about the problem as a Markov process due to the reliance of a current flight's departure on the airplane arriving on-time from its previous flight.

To address our project question, we decided to explore two different predictive models. First, we chose to build a Naïve Bayes model because: (1) it is a standard baseline classification model, (2) it is simple to train given our dataset, (3) its implementation exists in common Python libraries, and (4) it has been demonstrated to yield robust results that are competitive with other algorithms[4]. Secondly, we decided to build a logistic regression model because: (1) its parameter estimates are fully efficient, (2) it is suitable for predictions with relatively few variables, (3) its results are easy to interpret and draw conclusions about the relative impact of model inputs based on their coefficients, and (4) exploratory data analysis hinted at a possibility of linear relationship between the variables in our dataset and the binary outcome of a delay/no delay. Also, we supposed that these models would provide a good tradeoff between accuracy of results and difficulty of implementation.

III. Naïve Bayes Model

a. Model Inputs

To build Naïve Bayes classifier we selected the following variables as model inputs:

- Time variables (i.e., Date, Month, DayOfMonth, DayOfWeek) were selected to account for any effect that variation in weekly, monthly, and seasonal fluctuations in air travel could have on departure delay. Exploratory data analysis conducted prior to modeling revealed some variability in delays for each of time variable. For example, Figures 1 and 2 display variation in average departure delays by day of month and day of week (2008 data across all U.S. airports). While it fluctuates daily, average delay time is lower in the middle of the month and highest at the end. Similarly, there is a prominent pattern in departure delay time throughout a typical week – it is the lowest in the middle of the week when travel tends to subside and spikes on Sundays when many travelers rush to their destinations by the end of a weekend.
- Origin and Destination variables were selected to account for any airport-related inefficiencies that could be contributing to flight departure delays.
- Aircraft Tail Number was selected to capture effects of aircraft technical condition on departure delays.
- Finally, we included Airline Carrier as an input variable in case some airlines had a higher propensity for delays (for instance, due to inefficient aircraft maintenance or crew scheduling).

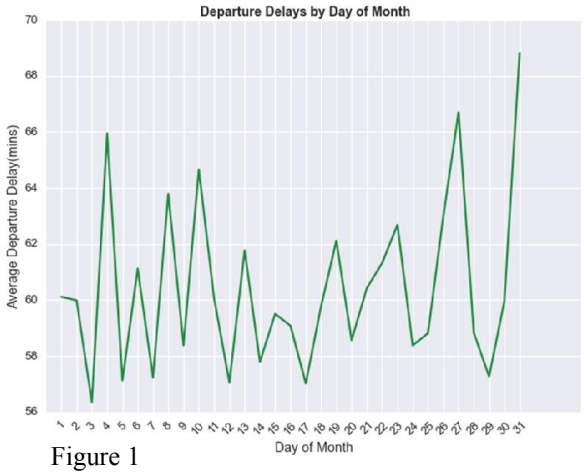

Figure 1

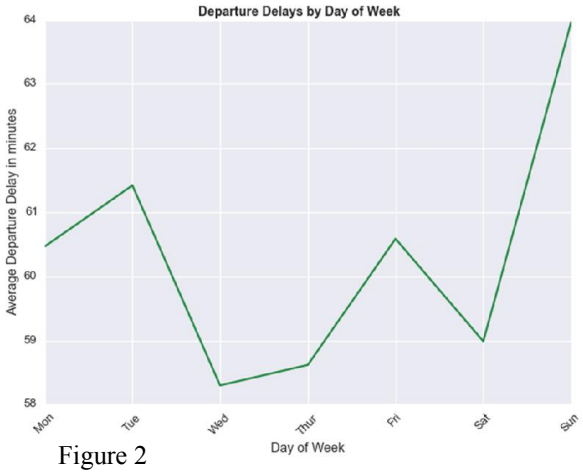

Figure 2

To implement the Naïve Bayes model, we structured the data using Python Pandas. Also, all categorical variables including Origin, Destination, and Tail Number were converted into numerical values to accommodate the requirements of the Naïve Bayes function in the scikit-learn Python library that we used for model implementation.

b. Model Implementation

The Naïve Bayes model was implemented using scikit-learn, a popular Python library for machine learning algorithms. Applied to this problem, Bayes' Theorem states:

$$P(Delay \mid attribute\ a) = [P(attribute\ a \mid Delay) * P(Delay)] / P(attribute\ a)$$

Using each attribute of the data, the algorithm calculates the right hand side probabilities and then multiplies the probabilities together to get a probability for a delay given a tuple of attributes (i.e., the nine variables that we have selected). Python's scikit-learn provides a fairly simple function for training, testing, and assessing the results of the model.

c. Model Training

Because of the large size of our dataset, we used a sample of data (33%) for training. Moreover, the training dataset was selected to represent a 50%-50% split between delayed and on-time flights. This was done to ensure that the model does not over-predict on-time flights, which are more common. Instead, by oversampling delayed flights during training, the model is able to more accurately assess the probabilities that contribute to delays. Finally, we trained the model on all available airports (without limiting to SFO and OAK). This allowed the model to train on and understand factors for delays across many airports making the model more generally applicable and not over-fit to the patterns peculiar to SFO and OAK airports.

d. Model Results and Testing

We employed a 8-fold cross-validation approach for training and testing the Naïve Bayes model whereby we used 7 years of data to train (sampled at 33% of available data) and the 8th year to test the model (sampled at 50%). This procedure was repeated eight times so that every year served once as the testing year. Using this procedure allowed us to test for over-fitting. Significant variation in the test results would indicate over-fitting. Based on the accuracy, precision, and recall measures, we calculated the F1-score as *2 * (precision * recall) / (precision + recall)*. This measure factors in the trade-off between precision and recall.

The results of the 8-fold cross-validation are as follows:

Test Year	Precision	Recall	Accuracy	F1-score
2001	0.58	0.62	0.61	0.60
2002	0.60	0.65	0.65	0.63
2003	0.65	0.70	0.70	0.67
2004	0.63	0.66	0.66	0.64
2005	0.62	0.64	0.64	0.63
2006	0.60	0.63	0.62	0.61
2007	0.60	0.61	0.60	0.61
2008	0.59	0.61	0.61	0.60
Average	0.61	0.64	0.64	0.63

Table 1

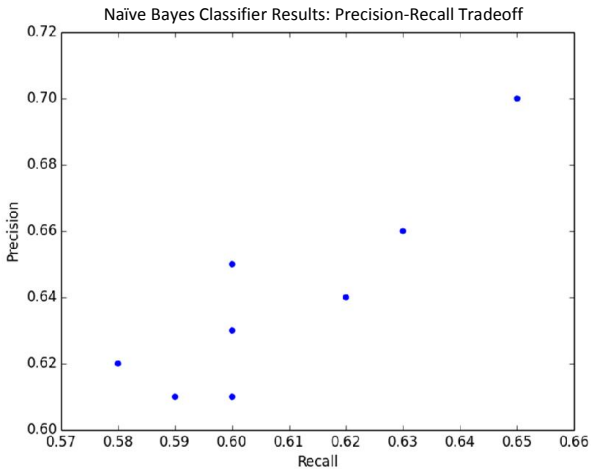

Figure 3

Given these consistent results, we believe that the model is robust for predicting flight delays. Figure 3 visually depicts precision-recall tradeoff for each validation test.

IV. Logistic Regression Model

a. Model Inputs

To construct the logistic regression we selected the following set of variables as the model inputs and prepared a dataset for modeling using Python Pandas:

- Similar to the Naïve Bayes model, day of week variable was selected to account for its effect on departure delay.
- Departure time was also selected to account for factors such as busiest time of the day and time-of-day specific weather conditions such as fog. To provide a proper logistic model input, this variable was transformed into categorical variable. Based on our research, we found that fog-related delays tend to occur before 1pm[5]. Therefore, we discretized time of day into categories from 07:00 to 12:59 (peak fog time at SFO), 13:00 to 17:59 (daylight hours), and 18:00 to 06:59 (night time).
- Origin and Destination variables were selected to account for any airport-related inefficiencies that could be contributing to flight departure delays.
- Finally, Airline Carrier was selected as an input variable to account for possibility of some airlines having a higher propensity for delays.

In order to implement the logistic regression model, categorical variables (i.e., Origin, Destination, and Airline Carrier) had to be transformed into binary since categorical or non-ordinal values are not meaningful inputs for a logistic regression. This was accomplished through a data pivoting procedure. For example, a variable for each carrier (over 300 in the entire dataset) was created and value of '1' was assigned to a single variable that represents a carrier for a specific flight record. All other carrier variables were filled with zero. Thus, each carrier was treated as an independent feature (since carriers are not correlated). Origin and Destination variables were transformed in the similar manner.

This data transformation was very computationally intensive and it resulted in a high dimensionality dataset. We got multiple memory errors when we tried building our dataset using Pandas. Thus, we decided to exclude some variables that were used in Naïve Bayes classifier, although we thought that they might have some predictive power. Thus, we decided to drop month, day of month, and aircraft

tail number as input variables. Otherwise, we would have to create dummy variables for every unique tail number (more than 7000 in 2008 dataset alone) which would have further increased computational complexity.

b. Model Implementation

We built our own logistic regression model for this project instead of using any Python library. We modeled it as:

$$P(delay) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m$$

This model uses a gradient descent algorithm to estimate coefficients for each variable. The learning model is designed to iterate over the training data and converge to a global solution. Once the coefficient values are determined, the model is then tested against a different sample of the data.

c. Model Training

Instead of using the k-fold validation by year utilized for the Naïve Bayes classifier, we used a random sampling of data from the 2001 to 2008 data. First, SFO and OAK flights were separated out of the sample set. Then, a fixed number of delays from the SFO/OAK data were randomly selected. Next, a random sample of on-time flights from SFO and OAK was selected such that the number of on-time flights matched the number of delayed flights. Again, by ensuring an even mix of delayed and on-time flights, we avoided model over-fitting to only recognize on-time flights. The sample size was progressively increased, although the results showed little variation across sample sizes.

d. Model Results and Testing

This model used maximum likelihood estimates (MLE) to perform fitting, yet it did not converge to a global solution. We suspect that this result is due to lack of computation capability of used machines. Figures 4 and 5 present the results of the model.

Logit Regression Results

Dep. Variable:	delay	No. Observations:	552264
Model:	Logit	Df Residuals:	552148
Method:	MLE	Df Model:	115
Date:	Mon, 05 May 2014	Pseudo R-squ.:	0.03886
Time:	23:38:55	Log-Likelihood:	-3.6792e+05
converged:	False	LL-Null:	-3.8280e+05
		LLR p-value:	0.000

	coef	std err	z	P>|z|	[95.0% Conf.	Int.]
dayOfWeek	0.0395	0.001	28.450	0.000	0.037	0.042
depTime	0.4724	0.003	135.999	0.000	0.466	0.479
carrier_3.0	0.4974	0.026	19.487	0.000	0.447	0.547
carrier_4.0	0.4629	0.081	5.707	0.000	0.304	0.622
carrier_5.0	-0.0812	0.083	-0.982	0.326	-0.243	0.081
carrier_6.0	-0.0024	0.024	-0.097	0.923	-0.050	0.045
carrier_7.0	-0.4049	0.034	-11.958	0.000	-0.471	-0.339
carrier_8.0	-0.2134	0.030	-7.229	0.000	-0.271	-0.156
carrier_9.0	0.3108	0.040	7.782	0.000	0.233	0.389
carrier_10.0	0.0158	0.028	0.561	0.575	-0.039	0.071
carrier_11.0	0.0547	0.069	0.791	0.429	-0.081	0.190
carrier_12.0	0.2303	0.029	8.080	0.000	0.174	0.286
carrier_13.0	-0.1475	0.044	-3.341	0.001	-0.234	-0.061
carrier_14.0	0.3010	0.028	10.763	0.000	0.246	0.356
carrier_15.0	0.4777	0.064	7.412	0.000	0.351	0.604
carrier_16.0	-0.3320	0.051	-6.528	0.000	-0.432	-0.232
carrier_17.0	0.2951	0.069	4.290	0.000	0.160	0.430
carrier_18.0	-0.1773	0.069	-2.570	0.010	-0.313	-0.042
carrier_19.0	-0.0355	0.037	-0.956	0.339	-0.108	0.037
carrier_20.0	-0.2080	0.167	-1.243	0.214	-0.536	0.120
carrier_21.0	0.2063	0.087	2.360	0.018	0.035	0.378

Figure 4

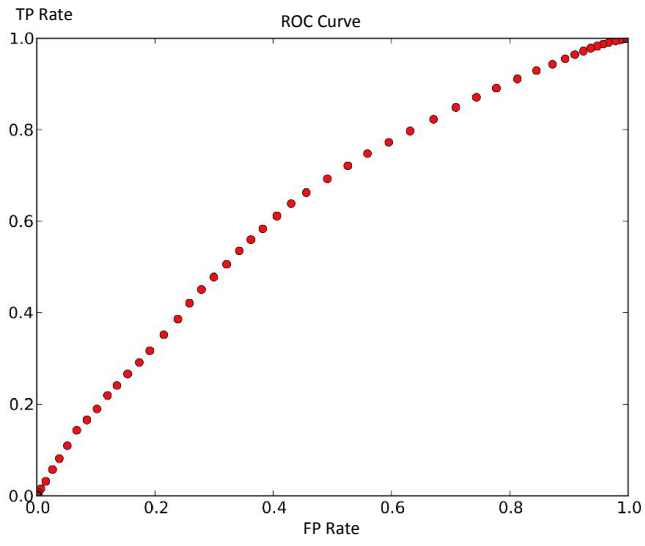

Figure 5

Because the model could not converge and because we had a reduced set of explanatory variables, the pseudo R^2 for the model was very low at 3.9%. This means that the model does not have much explanatory power. Put another way, it is likely that the variables we are using are not the strongest linear predictors of flight delays. The ROC curve shows that around point 0.4~0.5 brings out better performance than at any other point. In the real-world, we would likely augment this dataset with another dataset such as weather or aircraft maintenance records in order to get a more powerful model. Obviously, this would come at the cost of higher computational complexity, so tradeoffs would have to be considered.

We encountered several problems during testing of this model. This suggests limitations of the model and our data. The number of delays in the sample was less than 25%. In order to maintain a 50%-50% mix in the training data, we were limited to using at most 60% of the data. However, a more significant challenge was that the training data size was limited by the data structure and memory of the hardware. Given the 300+ variables that we were attempting to train, a fairly large dataset is required to ensure there are ample cases of every variable and combination of variables in order to properly train the model. Because of the limitations on the size of the dataset that we could use to train the model, the model was not able to converge to produce true values for all coefficients. Retrospectively, it may have made sense for us to restrict the number of carriers to the major carriers (~30-50). This may have made convergence for the model more likely.

e. Conclusions about Logistic Regression Model

Although our model shows good accuracy and a similar F1-score to the Naïve Bayes model (Table 2), the computational complexity of handling the dummy variables is significant. Logistic regression can be a very powerful model for predicting binary outcomes, but there is a high cost to computing this model. The model must iteratively converge to a solution – a process that depends on the quality of the training data. A best practice for implementing logistic regression models is being thoughtful about the quality and structure of the data, as well as the predictive power of variables before implementing the model. We couldn't use all the attributes available in the data because of the variable conversion. As mentioned earlier, converting Tail Number alone would have resulted in over 7000 additional dummy variables.

Precision	Recall	Accuracy	F1-score
0.61	0.59	0.61	0.60

Table 2

V. Model Selection

In choosing our final model, it is difficult to make an objective comparison given the differences in training/validation methods used by models and model results. Given the limitations of the logistic regression model encountered during its implementation and unreliability of its results, we elected to move forward with recommending Naïve Bayes classifier for prediction and application.

VI. Model Application

There are several uses of the Naïve Bayes classifier model that are possible, but we thought that the following application would be the most practical and interesting.

When to Leave and from which airport?

Peter has been assigned to his first engagement in Los Angeles and needs to be on-site by Monday, September 8th. He can arrive a few days early and work out of the local office to ensure that he is well-prepared for the project. Using our model, we predict September 8th and the six days before to see which day has the lowest probability of flight departure delays (Figure 6). First, we see that for every day in the week leading up to September 8th, SFO is a better option than OAK. The model shows that Saturday, September 6th is the best date for Peter to depart from the Bay Area to LAX. This gives him at least one day to explore the city and settle in.

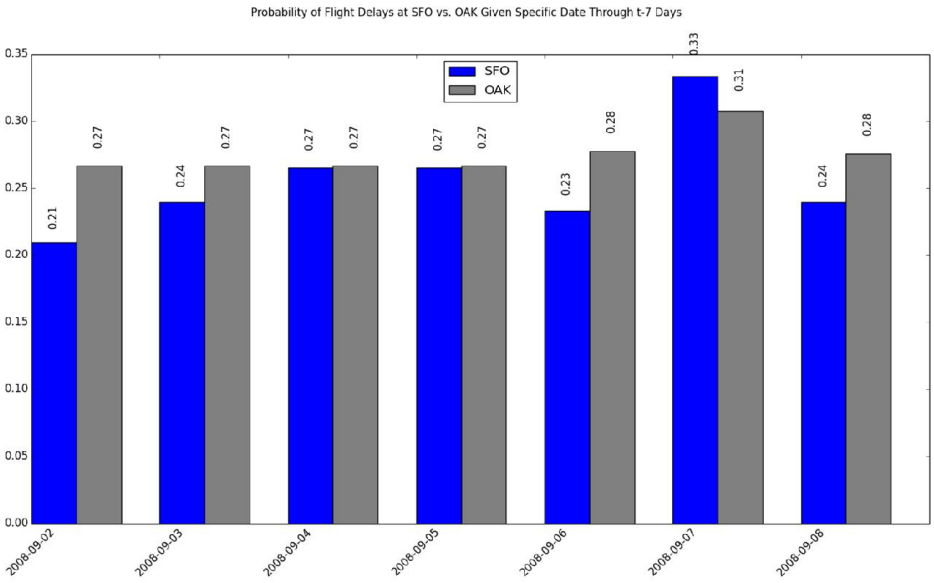

Figure 6

VII. Conclusions

We explored two models in attempt to predict departure delays at SFO and OAK. Our Naïve Bayes classification model performed relatively well and we believe it is a robust enough to be applied in predicting delays at all major U.S. airports in addition to SFO and OAK. We encountered multiple problems when developing the logistic regression model, mostly around complex data transformations necessary to ensure meaningful inputs and computational complexity encountered during training. Despite multiple attempts to improve the model via variation in input variables and sample sizes, the model did not converge to a global solution. Thus, it cannot be used for departure delay prediction with a given dataset.

The key takeaways and learnings from this project include: (1) the importance of conducting thorough explanatory data analysis, asking the right question from the dataset, and foreseeing data limitations, (2) data structure and computational complexity have to be taken into consideration when selecting predictor variables, (3) the process of modeling is an iterative process requiring multiple adjustments such as ensuring that training data is well balanced to produce non-biased outcomes.

VIII. Future Work

Our successful Naïve Bayes model could be further improved by augmenting the flights dataset with additional data sources such weather data since weather patterns are likely important factors influencing departure delays. In addition, other classification models like Decision Trees and SVM can be explored for this dataset. More complex models that can incorporate delay propagation (e.g., Markov chain) can also be considered.

References:

1. http://www.rita.dot.gov/bts/subject_areas/airline_information/airline_ontime_tables/2013_12/table_06
2. http://stat-computing.org/dataexpo/2009/the-data.html
3. http://ggweather.com/enso/oni.htm
4. Han, J., Kamber, M., Pei, J. (2012). Data Mining: Concepts and Techniques. *Morgan Kaufmann.*
5. http://crankyflier.com/2010/10/14/san-franciscos-fog-and-runway-problems-give-the-airport-a-dubious-honor/