Springboard Data Science Intensive Capstone Project - Predicting the Likelihood of Flight Cancellations

Aashish Jain

July 31, 2017

Contents

1 Introduction

Imagine you have a trip coming up in next few days and someone tells you that "your flight has a high chance of being cancelled, so be aware of that and rethink about your travel, hotel bookings, etc..". That would be helpful for you and all other passengers traveling out there. Even though the annual flight cancellation rate is not high (about 1-2% in the US domestic market), that one rare event causes a lot of troubles to passengers in terms of rescheduling their travel plans. There are many factors such as flight date and time, origin and destination airport, airline type, weather, etc.. which might affect the cancellation rate. We use data from various sources containing these factors and build a supervised machine learning model for predicting the likelihood of flight cancellation for US domestic flights operating at selected airports.

Travel planners and booking companies such as booking.com, expedia.com, kayak.com, priceline.com, etc. can use such a model to predict the likelihood of the cancellation of a flight. They can then inform their customers well in advance, even before the airlines' management informs the passengers, about the probability of the cancellation of their upcoming flight. From the traveler's point of view, it would be very convenient for them. On the other hand, such a predictive model would enhance the product base of travel planner companies. Moreover, there is a possibility of developing an app which travelers can use to know about their flight cancellation likelihood in advance.

2 Data Acquisition and Cleaning

We acquire datasets from two different sources. The first dataset contains flight information and the second dataset has information about the weather.

The flight data is acquired from the Bureau of Transportation Statistics. This website allows downloading data for one month at a time. We downloaded data for multiple months and concatenated all the data together. More details about acquiring and concatenating the data can be found in this IPython notebook. Each row in the flight dataset corresponds to a unique flight with details such as flight date, carrier name, origin airport, destination airport, departure time, arrival time, distance, departure delay, arrival delay, cancellation status, taxi times, and many other on-performance data. We have also extracted some historical information about the flights and added new columns. The historical data contains information about flight delays, cancellation, diversions, etc.. in last "ndays" with three values of "ndays = 10, 20, 30". More details about the calculations of historical performance can be found in this IPython notebook.

The hourly weather data is downloaded using wunderground.com API in XML

format. The weather data contains information such as temperature, humidity, visibility, wind direction, weather condition etc.. One API call can be used to download data for a chosen airport and a chosen date (for all hours on that date). So, if we want to get the weather data for one airport, say LAX, for two years, we would need close to $2 \times 365 = 730$ API calls. Due to some restrictions on number of API calls per day and also on API call rate (per minute), we acquired data for only top 20 airports (in terms of observing the most traffic during 2015-2016). More details about accessing the weather data and parsing it to a proper format can be found in this IPython notebook.

Having the two datasets, we then merge them such that we get the weather information for each flight at its origin and destination locations. More details on merging these datasets can be found in this IPython notebook. Other than having missing values already in the original datasets, merging the datasets also generates some missing values. We fix all the missing values by either imputations or by filling them with zeros. We also remove some of the columns that do not provide any meaningful information. More details on the data cleaning process can be found in this IPython notebook. The cleaned dataset is then ready for explorations.

3 Data Exploration

3.1 Introduction to the cleaned data

There are 1,417,308 and 1,439,831 records for years 2015 and 2016, respectively, and 90 fields. In this project, we considered top 20 airports in the US (in terms of most traffic). These 20 airports network broadly covers the whole US as shown in Fig. 1. The justification for selecting these 20 airports is discussed in detail in this IPython notebook.

We will go through most of the fields (or columns) in the dataset to explore their relationship with flight cancellation rate. Details about each field can be found in the Bureau of Transportation Statistics and the Wunderground websites and some in this IPython notebook and this IPython notebook. The target column for this project is called "Cancelled" which contains two values: 1 for cancelled flights and 0 for not-cancelled flights.

3.2 Flight Cancellation Rate

Out of 2.8+ million flights operating at top 20 airports in 2015-2016, about 1.15% of them got cancelled. This does not seem like a large number but such rare events

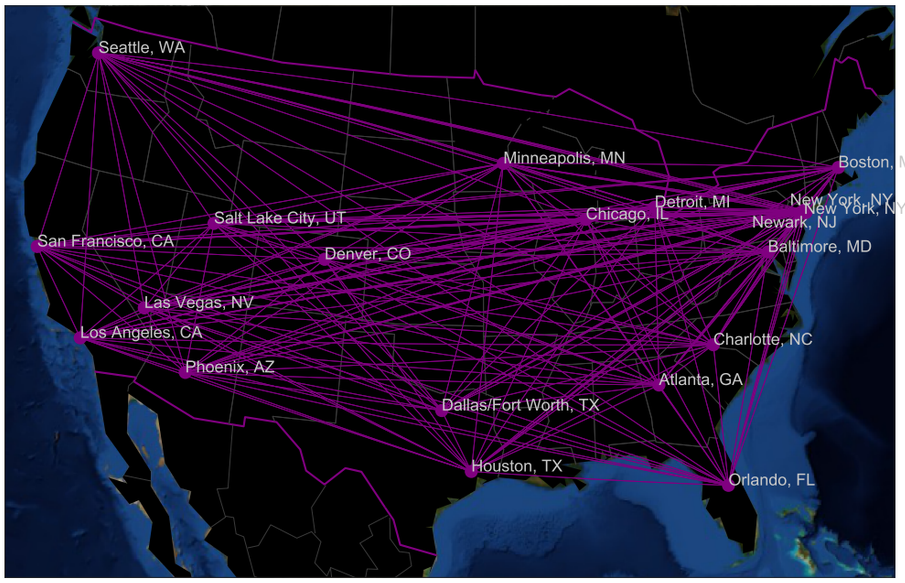

Figure 1: Network of top 20 airports in the US. Note that there are two airports in the New York City: LGA and JFK.

cause a great deal of inconveniences to passengers, and cost a lot of money to airline companies. Therefore, it is important to understand where, when and how this small events occur. To start with, we plot the total number of flights on a daily basis and see how many flights got cancelled (on a daily basis) in Fig. 2. The daily total number of flights remain almost steady with regular and periodic

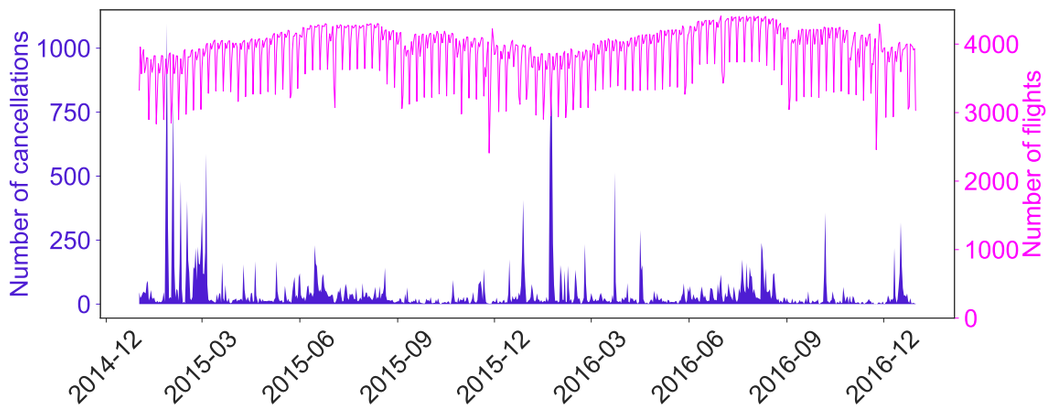

Figure 2: Total number of flights (right y-axis) and number of cancelled flights (left y-axis), on a daily basis.

troughs. The number of cancelled flights has no steady trend but has some big spikes. Knowing the number of flights and number of cancellations, we can calculate the cancellation rate for a given day. We define the cancellation rate as,

$$\text{Flight cancellation rate} = \frac{\text{Number of flights cancelled for a given scenario}}{\text{Total number of flights for a given scenario}}, \quad (1)$$

where a “scenario" can refer to a class of a field. In the plot above, a scenario would refer to a date, say June 24th 2015. Figure 3 shows the daily % cancellation

rates. Big spikes in the cancellation rates were mainly caused by bad weather as

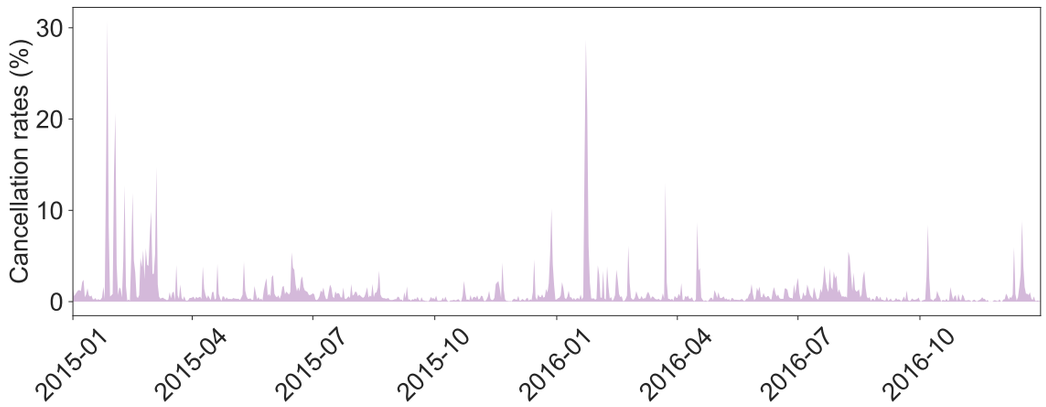

Figure 3: Daily cancellation rates.

depicted above. There were some spikes in cancellation activities in the end of June and beginning of December too.

We can discuss some other examples for cancellation rates also. For instance, if the field is weather condition which has classes such as Heavy Snow, Rain, Clear Sky etc.., a scenario can be one of these weather conditions. We then count the number of flights operating under such a scenario (say Heavy Snow) and also count the number of flights that got cancelled under the same scenario. Equation (1) can then be used to calculate the cancellation rate when the weather condition is Heavy Snow. In the following few sub-sections, we will go through many interesting fields and explore the trend for cancellation rates. There are broadly 6 categories of information that are embedded in all the fields:

1. Calendar variables
2. Airports
3. Airlines
4. Flight distance
5. Weather factors
6. Historical performances

3.3 Calendar Variables

There are many calendar variables such as quarter, month, week, day, hour, minute, etc.. Here, we explore the dependency of cancellation rates on month, day of week and scheduled hour. For some flights the origin and destination calendar variables can be different, and hence we calculate the cancellation rates for all variables at both origin and destination airports. Figure 4 shows the monthly cancellation rate. There is no difference in cancellation rates between the origin and the destination airports in any given month. February was the worst followed by January and

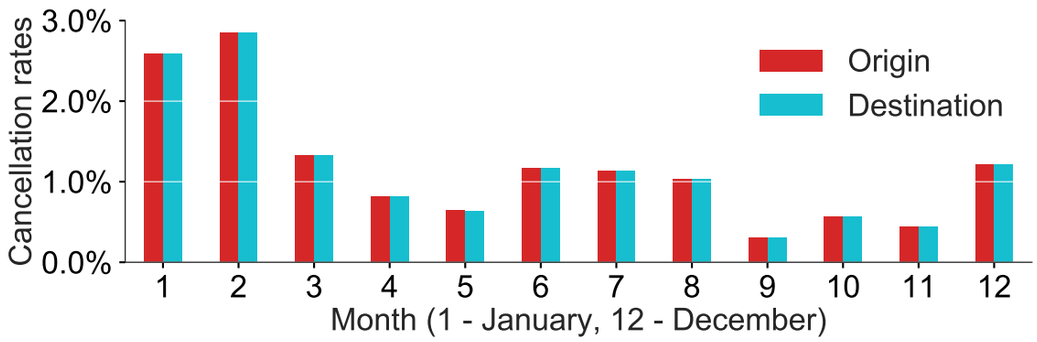

Figure 4: Monthly cancellation rates.

March. We see some mild spikes for June and December too. The high cancellation rate in January and February was mainly due to the snow storms in the east coast. We can also look at the day of the week and understand its influence on the cancellation rate in Fig. 5. End of the weekend and beginning of the week ob-

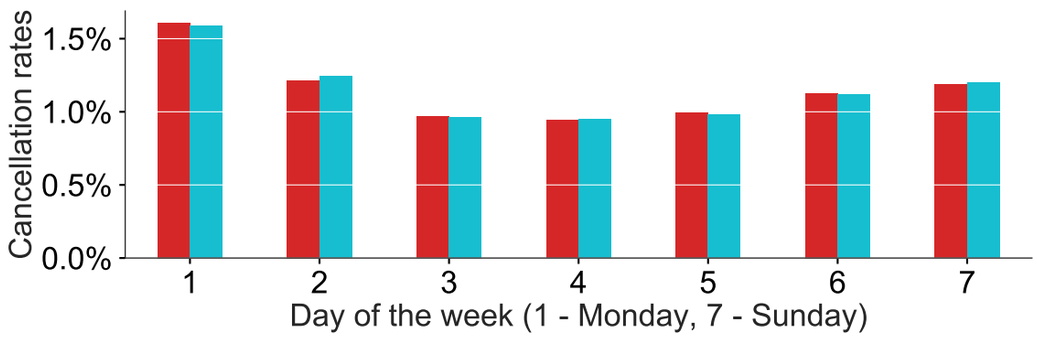

Figure 5: Cancellation rates depend on the day of the week. Note the colors correspond to the same legend as in Fig.4.

served higher cancellation rates as compared to the middle week days. There are very slight differences in cancellation rates between the origin and the destination airports for any day of the week. We can go down one more level in the calendar variable space and explore the hours of the flights throughout the day in Fig. 6. For all scheduled departure hours, except between 2 - 3 AM, the cancellation rates

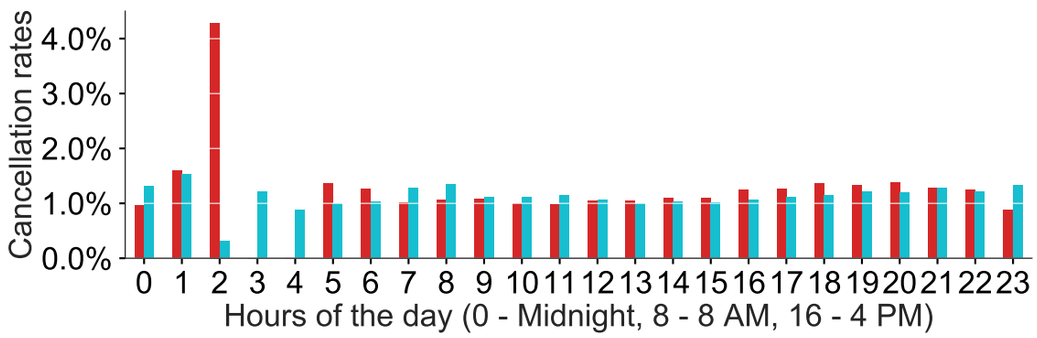

Figure 6: Hourly cancellation rates throughout the day. Note the colors correspond to the same legend as in Fig.4.

are below 2%. We do not see any big spike in the case of scheduled arrival hours. Out of 210 flights scheduled to depart between 2-3 AM , 9 were cancelled, leading to spike at 2-3 AM red bar in the figure. This completes our brief discussion on calendar variables.

3.4 Airports

We calculate the cancellation rates for flights departing from and arriving at all top 20 airports and display the results in Fig. 7. The IATA code can be found in

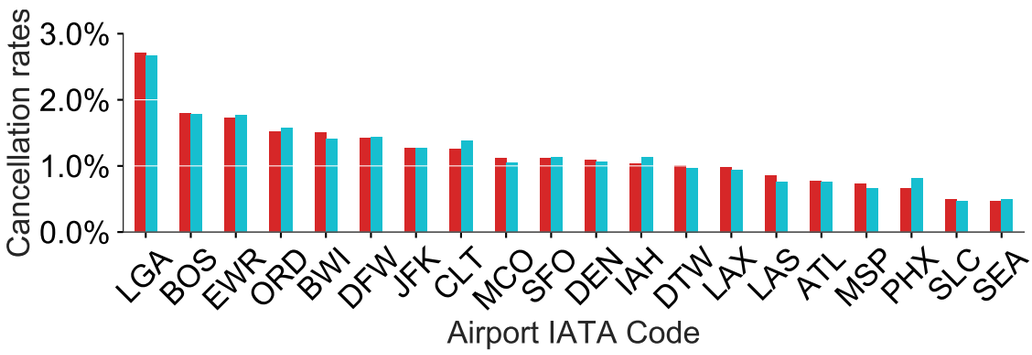

Figure 7: Cancellation rates dependency on the airport. Note the colors correspond to the same legend as in Fig.4.

this link. The flights departing from LaGuardia Airport (LGA) have the highest cancellation rate whereas the flights departing from Seattle - Tacoma International Airport (SEA) have lowest rate. The top 2 and the bottom 2 airports remain the same whether we are looking at origin or destination airport.

3.5 Airlines

For airlines, it does not make sense to distinguish between the origin and destination. Figure 8 shows the cancellation rates for 13 airlines. The airline IATA code

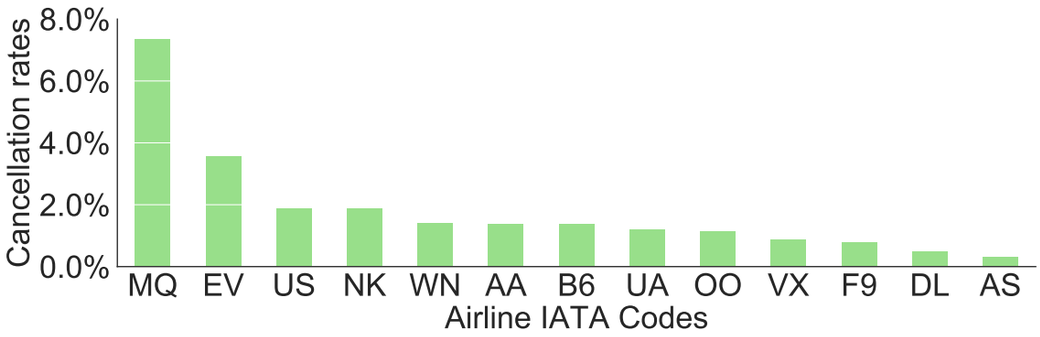

Figure 8: Cancellation rates dependency on the airline.

can be found in this link. The highest cancellation rate is seen for the Envoy Air (MQ), and lowest is seen for the Alaska Airlines (AS). Two airlines with highest cancellation rates (MQ and EV) are not mainline airlines but rather regional ones. OO (SkyWest Airlines) is also regional airline but has relatively lower cancellation rates.

3.6 Flight Distance

Flight distance is recorded in miles and is a continuous variable. For every numerical value of this variable, we calculate the cancellation rate, which is shown in Fig. 9. There is a data point for Distance = 21 miles for which the cancellation

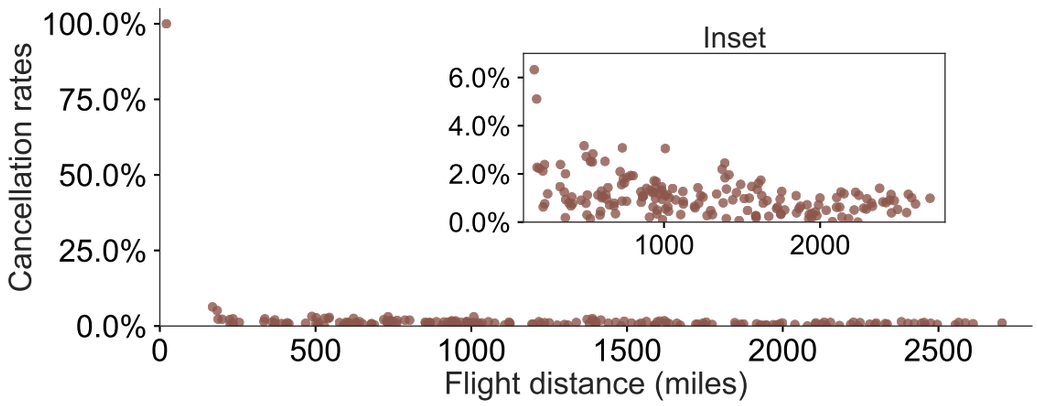

Figure 9: Cancellation rates as a function of flight distance.

rate was 100%. In order to see the clear trend for all other data points, we omitted the 21 miles point and replotted the data in the inset figure. We can see that the cancellation rates are higher for shorter distance flights and lower for longer distance flights. We performed a hypothesis test to test the null that there is no relationship between the distance and the cancellation rate. Using Spearman's ρ, we found a weak correlation of -0.39 which was statistically significant.

3.7 Weather Factors

There are many weather factors such as temperature, dew point, pressure, wind speed, wind direction, humidity etc.. but we will focus on only some factors here to keep the discussion short. A detailed data exploration can be found in this IPython notebook.

Fig. 10 displays the cancellation rate asa function of temperature at both origin and destination (at the time of departure and arrival, respectively). There appears to be two broad temperature regimes here in terms of cancellation rates. The cancellation rates are four times higher when the temperatures are below 40°F as compared to the situations when temperatures above 40°F. which can be clearly seen in the inset figure. Humidity and wind speed have similar effects on cancellation rates as shown in Fig. 11 for both origin and destination locations. For very dry weather conditions (less than 10%), there is a decreasing trend for cancellation rate. The rate then monotonically increase for humidity more than 20%. For the monotonic part, we found statistically significant values of Spearman's correlation ρ to be around 0.79 for origin and 0.82 for destination. Also, we see a monotonically increasing trend for cancellation rates as a function of wind speed until around 25 mph. For wind speed greater than that, the cancellation rates start

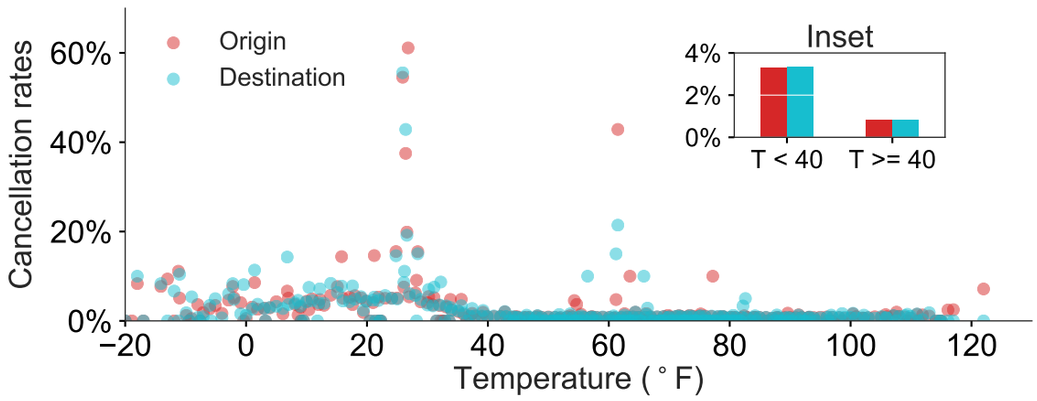

Figure 10: Cancellation rate as a function of temperature. In the inset bar chart, y-axis is for the cancellation rate and T represents temperature.

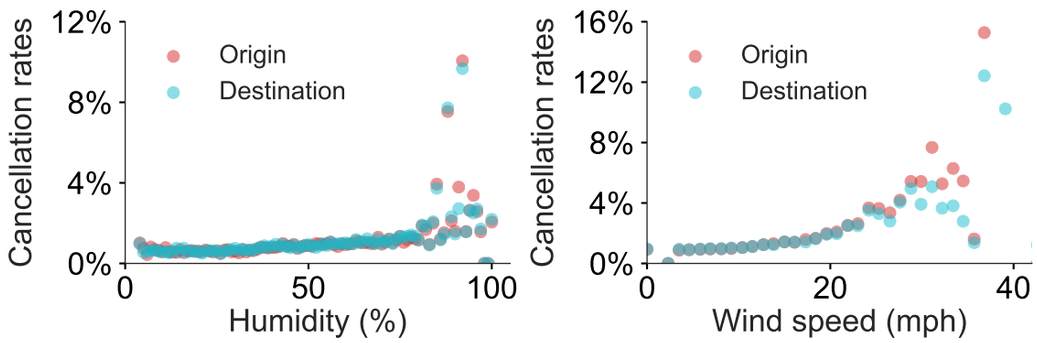

Figure 11: Cancellation rate as a function of % humidity and wind speed (mph).

to decrease. The maximum point for cancellation rates are different for origin and destination airports. Overall, the trends are pretty much like parabolic curves for wind speed.

We now look at the wind direction in FIg. 12. It turns out that from 90 to 270 degrees, i.e. from East to South to West, the cancellation rates are around 1%. When the wind direction goes from West to close to North, the cancellation rates increased. The rates fluctuate a lot when the wind direction is close to North. From North to Wast wind direction, the cancellation rate start to decrease. Finally,

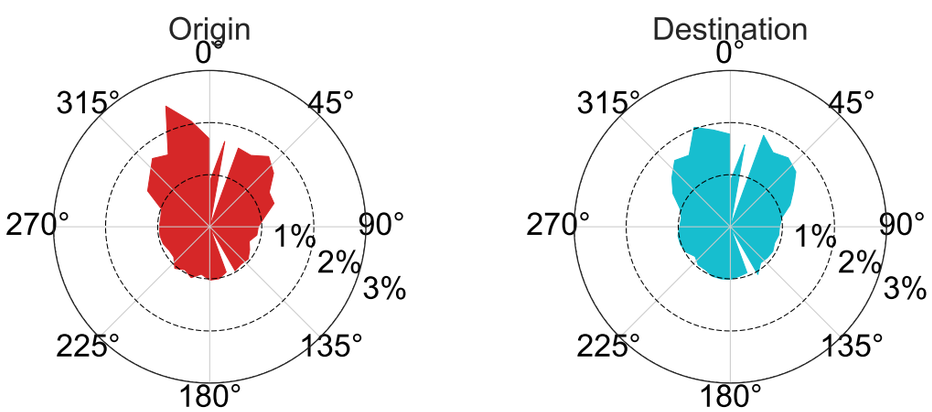

Figure 12: Cancellation rate as a function of wind direction. 0° corresponds to North, 90°% to East, 180° to South and 270° to West.

we look at various weather conditions and find out the cancellation rates under

all conditions, as displayed in Fig. 13. Cancellation rates are pretty high when

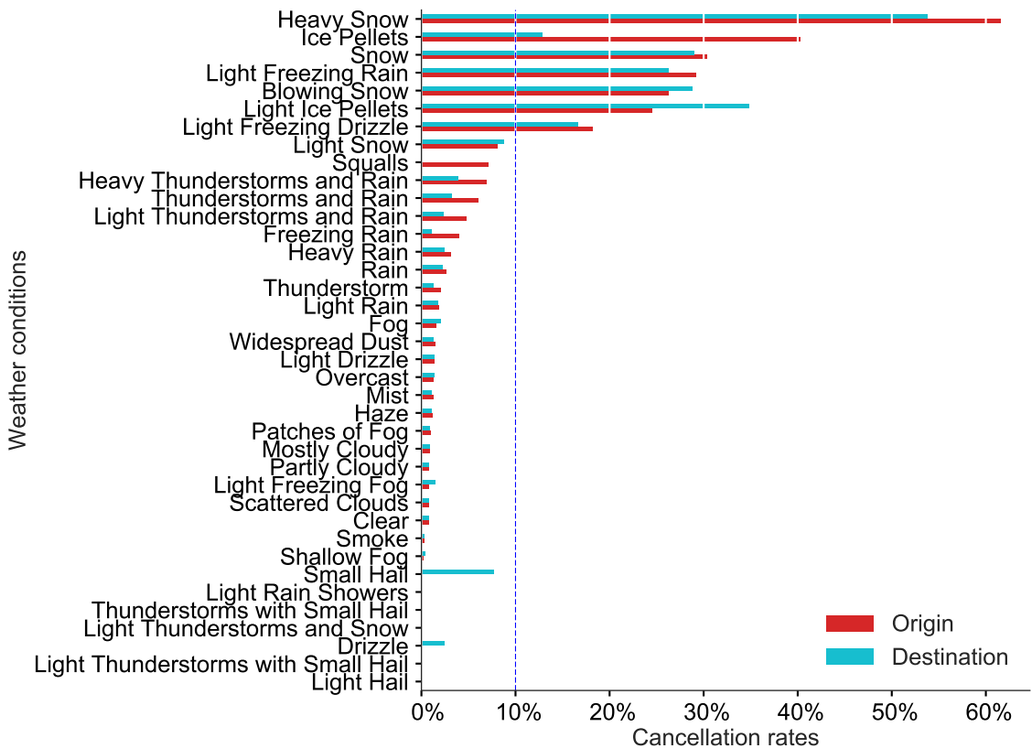

Figure 13: Cancellation rates under different weather conditions at origin and destination airports. The dashed vertical blue line indicates 10% cancellation rate line.

there are snow related activities at both origin and destination airports. Rain with some thunderstorms also has high cancellation rates. There are similar top weather factors for both locations for cancellation rates greater than 10% (indicated by blue dashed line), with an exception such as “Squalls" which has very low cancellation rate at destination and high rates at the origin airports. However, this observation is not significant as there are only 28 records for Squalls at origin. Similarly, the cancellation rates are almost 0 when there are any type of hail conditions. Again, the number of records containing this condition is 69 at origin, so this observation is also not significant.

3.8 Historical Performances

For historical performances of a given flight, we have the number of cancellations, number of diversions, departure delays, arrival delays and taxi times for last “ndays", where we have three values of ndays = 10, 20 and 30. Figure 14 shows the influence of the number of cancellations and diversions in the last ndays on the cancellation rate of the flight in question. Each data point in this figure is the result of some calculations. For example, to calculate the cancellation rate for a given number of cancellations (N_c), we pick all the flights for which the number of cancellations were N_c in the last ndays. We then count the number of such flights, and also the number of cancelled flights. The ratio then gives us the cancellation rate for N_c. We can perform the similar steps for all other historical performance variables but let us first discuss Fig. 14. For the left panel figure, we observe a

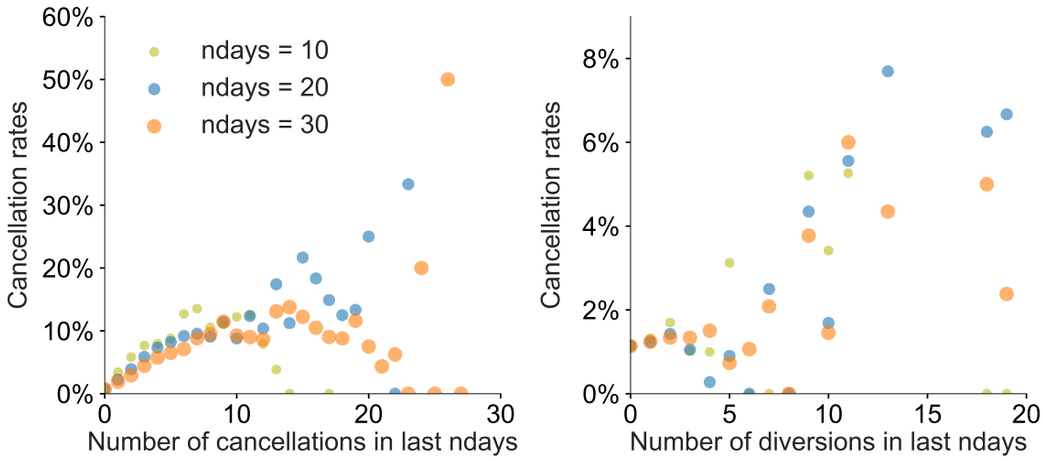

Figure 14: Cancellation rates as a function of the number of cancellations and the number of diversions in the last ndays.

parabolic pattern with maximum in cancellation rates occurring at different values of the number of cancellations for different values of ndays. There also appears to be some outliers in all three cases, usually at higher values of the number of cancellations in last ndays. For the right panel figure, generally, the cancellation rates for a flight is higher when the number of diversions of that flight in the last ndays are higher.

Sometimes, we come across a flight for which we do not find any history in the last ndays. We call such flights as temporary flights. The term “temporary" could be different for different ndays. For example, a flight can be temporary if looked at 10 days history but “routine" if looked at 30 days history. Figure 15 (a) compares the cancellation rates for temporary (Yes) and routine (No) flights for all three ndays. Usually, the cancellation rates are higher for temporary flights. There

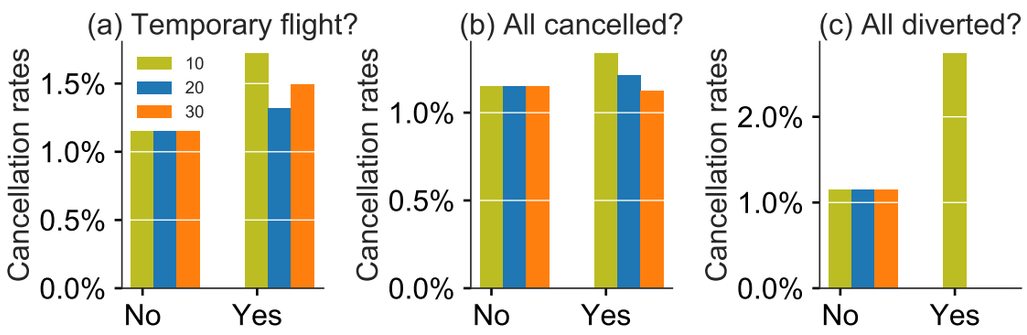

Figure 15: Cancellation rate comparisons for three different history indicators, and for all three ndays. The legend 10, 20 and 30 indicates different ndays.

are also some flights for which the history tells that all the flights in last ndays got cancelled. The cancellation rates are slightly higher when all flights got cancelled in the last ndays, as seen in Fig. 15 (b). Similarly, there are flights for which there were 100% diverted flights in the last ndays. We found such cases only for ndays=10 and the cancellation rates are about 3 times higher as compared to the flights for which the history had no 100% diversions, as shown in Fig. 15 (c).

Now, we look at the influence of departure delays statistics on cancellation

rates. Figure 16 presents the cancellation rates for given median values of the departure delays in the last ndays.

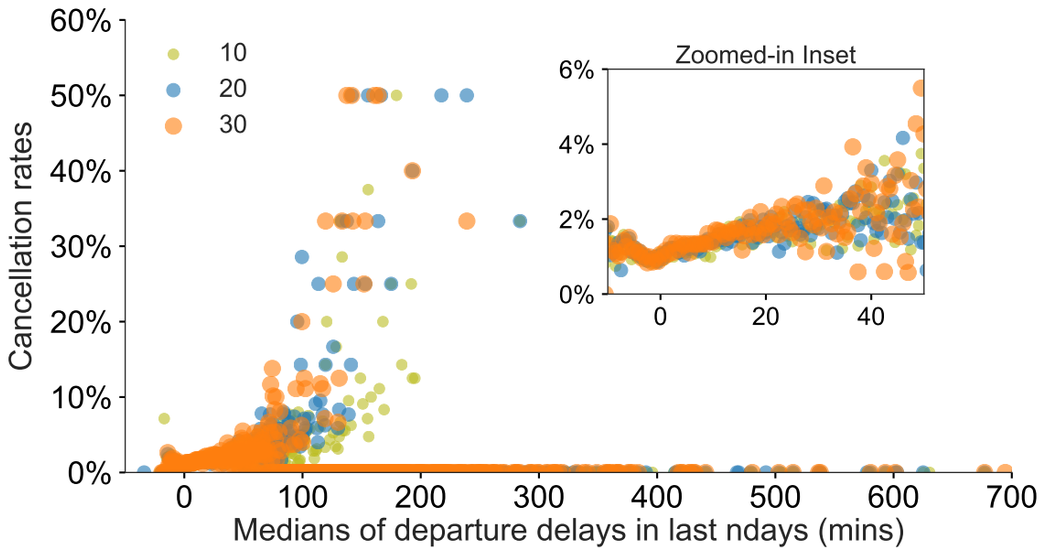

Figure 16: Cancellation rates as a function of the medians of departure delays in the last ndays. The legend 10, 20 and 30 indicates different ndays. The inset figure has the same x and y labels as the main figure.

Cancellation rates are 0 (except a couple of instances) when the medians of departure delays in last ndays days were greater than 200 minutes. For less than 200 mins, and more specifically between 0 and 70 mins, there is a somewhat increasing cancellation rate as a function of median values, as displayed in the inset figure. Statistically significant correlations (Spearman's $\rho = 0.77, 0.69, 0.55$, for ndays = 10, 20, 30, respectively) are found for all three ndays data in 0-70 mins range.

In a similar fashion, we can also look at the effect of medians of arrival delays in Fig. 17. This scatter plot looks very similar to the one that we got for medians of departure delays. For more than about 200 mins arrival delays, the cancellation rates are 0 for any ndays history. However, when we zoom-in the plot between -50 to 50 mins, we observe a non-monotonic behavior.

4 Modeling

Knowing the labels for flight cancellations, i.e. 0 for not cancelled and 1 for cancelled, we use supervised machine learning algorithms to build a predictive model. Furthermore, since there are only two outcomes (or classes) in the data (0 and 1), we use binary classification algorithms. The models are trained using the 50% of the data and the remaining 50% is used to evaluate the performance of the models. Out of about 2.85 millions flights in 2015-2016, only about 1.15% flights got cancelled, hence we have a highly imbalanced data. Mostly, all standard algorithms are not well suited for learning with highly imbalanced data. The

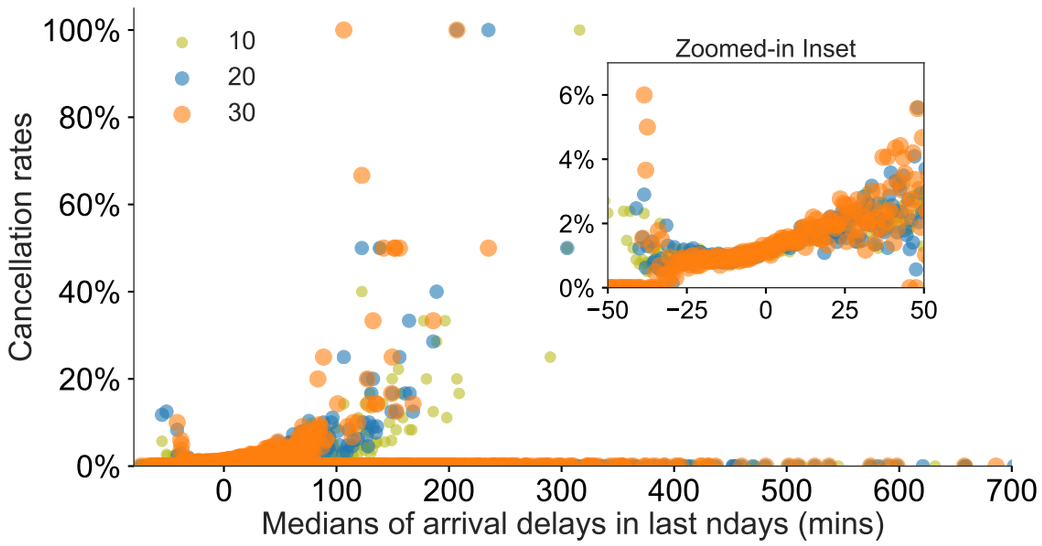

Figure 17: Cancellation rates as a function of the medians of arrival delays in the last ndays. The legend 10, 20 and 30 indicates different ndays. The inset figure has the same x and y labels as the main figure.

reason for this difficulty is that the learning in most algorithms is biased towards the majority class. In other words, for highly imbalanced dataset, the classification algorithms fail to predict positive class in reasonable numbers. To address this issue, either training data is resampled so that the learning algorithm gets balanced data and learns to predict both classes reasonably well, or the training data is weighted more for minority class and less for majority class. Not only this resampling matters but also the evaluation metric to measure the performance of the model plays a vital role. For example, the accuracy can be really high even if none of the minority class examples are predicted. There are many different types of evaluation metrics that are appropriate for imbalanced classes. Shortly, we will go through some classification algorithms and build predictive models by keeping the imbalanced data challenge issue in our mind.

4.1 Data Pre-processing

Before feeding the data into any machine learning algorithms, there are some pre-processing steps that must be performed on the data. We outline these steps below. Note that some of the steps are not required (or not good for the best results) in some algorithms, but we list below all the pre-processing steps (in order that they are performed) that we used across all classification algorithms in this work:

1. **Label encoding:** In the dataset, there are some variables with numerical values, some variables with categories and some variables with binary values (0 and 1). For numerical and binary variables, we do not worry about labeling. However, we perform label encoding for the categorical variables. This step is carried out on the whole dataset.

2. **Data splitting:** The second step involves splitting the label encoded dataset

into train and test datasets. In this project we split them equally with 50%-50% ratio. Also, we split them in such a manner that the fractions of both classes remain almost same in train and test datasets.

3. **Resampling or weighting:** In the third step, we take care of the imbalanced data issue by addressing it at the data level by either resampling or weighting the training dataset. Resampling may involve under-sampling the majority class examples or oversampling the minority class examples. One can also use the combinations of under and over sampling techniques. There also exists Synthetic Minority Oversampling Technique (SMOTE) which generates synthetic training data for minority class examples by using interpolations. Sometimes, rather than resampling, weighting the training samples works best. In the weighting technique, more weights are given to minority class examples. Different resampling techniques and weighting techniques work best for different algorithms, so we try few different techniques with all the classification algorithms in this project and pick the best ones.

4. **Scaling:** For some algorithms, it is necessary that we scale the values of all features to lie within a fixed range. We scale features such that all features have values between 0 and 1.

After scaling the features, we can also reduce the feature space by selecting features that have most predictive power. Different techniques such that χ^2 test or even principal component analysis (PCA) can be used to do so. However, in this project we have not performed any feature selection at this stage.

4.2 Modeling Pipeline and Evaluation Metric

Once the data is pre-processed, we feed them to classification algorithm to build the model. In order to evaluate the performance of the model, we test the model on the test dataset. Before making predictions on test dataset, we use the exact same pre-processing steps that we used for training dataset and apply them on the test dataset. Python's scikit learn library has a very useful class called "pipeline" which we use to combine all the steps, i.e. pre-processing and classifier learning steps into one. This pipeline is then applied directly on the test dataset. Note that for some algorithms, training the 50% of the data, which is about 1.4 million examples, is very slow. For such algorithms, we first consider only the 10% of the whole data. We then follow the pre-processing steps on the 10% of the data. Once we know the optimal hyperparameters and the best resampling/weighting technique, we then use the whole data to build the model. For tuning hyper-parameters we use 5-fold cross validation with grid search method in scikit learn. We run the cross validation to obtain the optimal hyperparameters for all considered resampling/weighting techniques.

One more important consideration while performing cross validation is the selection of a proper evaluation metric. Especially, for imbalanced data, it is impor-

tant to be careful about the choice of the evaluation metric. Our aim is to predict the flight cancellation likelihood with data containing only about 1.14% of the cancelled flights in 2015-2016. Accuracy is not a good metric for such datasets. We definitely want to have high true positive rate (or recall) with cancelled flight tagged as positive class. At the same time, we do not want lots of false positives or less precision. Most of the time, the choice of a good metric depends on business needs. In this project, we keep in mind all elements of a confusion matrix. Also, we want a metric which is threshold invariant, so F1-score is also not a great choice. There are two popular metrics such as area under the curve (AUC) of receiver operating characteristic (ROC) curve and AUC of precision recall (PR) curve. Theoretically, ROC curves are useful in an algorithm that optimizes PR AUC. Also, T. Saito and M. Rehmsmeier found that for imbalanced data, precision recall curve is more informative than ROC curve. So, we will stick with PR AUC as our scoring parameter for tuning hyperparameters.

4.3 Logistic Regression

For logistic regression algorithm, we use all 4 pre-processing steps mentioned in Sec. 4.1. Training is pretty slow for the 50% of the whole data, so we consider only the 10% of the whole data to tune hyperparameters and find best resampling/weighting technique. Table 1 shows the results for various metrics for all different resampling/weighting techniques that we tried. We found, using 10%

Table 1: Logistic regression classifier - choosing the best resampling/weighting technique

Resampling/weighting technique	**Class**	**Values**			**PR AUC**	**ROC AUC**	**CPU time (min)**
		Precision	Recall	F1-score			
RandomUnderSampler (10%)	0	0.99	1.00	0.99	0.51	0.50	0.01
	1	0.00	0.00	0.00			
SMOTE (10%)	0	0.99	1.00	0.99	0.51	0.50	0.1
	1	0.00	0.00	0.00			
No resampling, using weighting (10%)	0	0.99	1.00	0.99	0.51	0.50	0.03
	1	0.00	0.00	0.00			
Final - RandomUnderSampler (100%)	0	0.99	1.00	0.99	0.51	0.50	0.1
	1	0.00	0.00	0.00			

of the whole data, that all techniques led to very poor model. However, the random under sampling technique (RandomUnderSampler in scikit learn) spent least computational time. So, we used random under-sampling technique and build the classifier using the whole data (1.4+ million examples in training dataset). This model is represented as final model and colored in green in Tab. 1, and the ROC and PR curves are shown in Fig. 18 . Testing on the holdout data (50% of the whole data) gave us the same poor results. In conclusion, by maximizing PR AUC, the logistic regression algorithm produces predictions that are absolutely by "chance" or "luck" by penalizing the coefficients very strongly (very high regularization). This also means that we get high bias model, i.e. under-fit model, which explains the term "luck". Also, the model does not predict even a single flight cancellation, which leads to zero recall, precision and hence F1-score. In other words, the model works really well but only for negative class, i.e. non-cancelled flights. Upon trying a different metric (such as F1-score or recall) for optimization

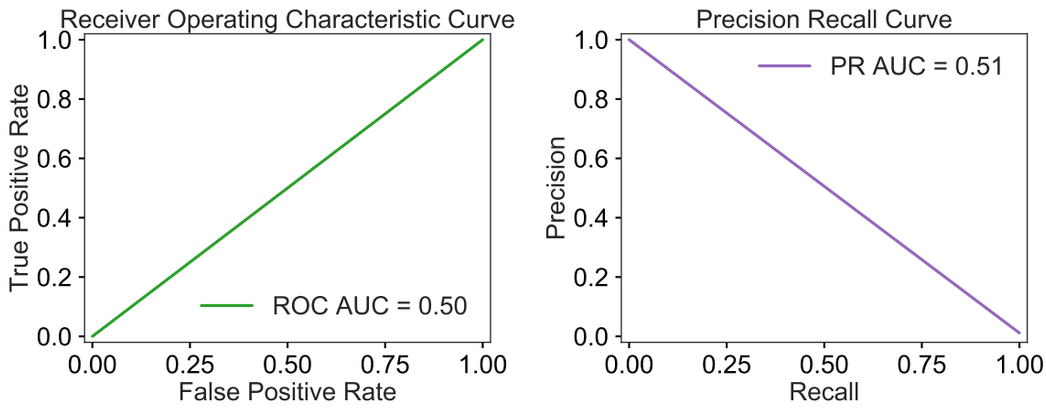

Figure 18: ROC and PR curves for the logistic regression classifier.

in cross validation step, we obtained non-zero values for precision, recall and F1-score but the PR AUC was too low. Again, choosing a metric depends on business requirements. In this project, we fix the scoring parameter to be PR AUC across all classifiers, so we accept the poor results that we get from logistic regression.

4.4 Gaussian Naive Bayes

We use Gaussian Naive Bayes classifier and the first three steps of the pre-processing steps mentioned in Sec. 4.1. This algorithm is not that slow so we use the whole dataset and split that into train and test (holdout) datasets with 50%-50% ratio. Table 2 shows the results for various metrics for all different resampling techniques that we tried. Note that the Naive Bayes classification implementation in scikit learn does not have any option for weighting the classes. So, here we try only two resampling techniques. Both resampling techniques give PR AUC to be

Table 2: Gaussian Naive Bayes classifier - choosing the best resampling technique

Resampling/weighting technique	**Class**	**Values**			**PR AUC**	**ROC AUC**	**CPU time (min)**
		Precision	Recall	F1-score			
RandomUnderSampler	0	0.99	0.87	0.93	0.08	0.76	0.03
	1	0.04	0.48	0.08			
SMOTE	0	0.99	0.86	0.92	0.08	0.75	1
	1	0.04	0.53	0.08			

0.08, which is very poor. However, the random under sampling is faster and also the ROC AUC is slightly better than that obtained using SMOTE. So, we choose random under sampling. The chosen model is highlighted in green in Tab. 2. Any Naive Bayes algorithm assumes that the features are independent for a given class. One of the reasons for a very poor PR AUC is probably due to violating this assumption. Therefore, we can try to look at correlations amongst features and remove the ones that are highly correlated. We found many pairs of features that have correlation coefficient values closer to 1. We removed the top 10 features with high correlations and trained the model using the best chosen resampling technique. Removing features did not help in improving the results but also it did not decrease the PR AUC. Finally, we consider the reduced feature set and the resulting ROC and PR curves (obtained by running model on 50% holdout dataset)

are shown in Fig. 19. As far as ROC curves are concerned, Gaussian Naive Bayes

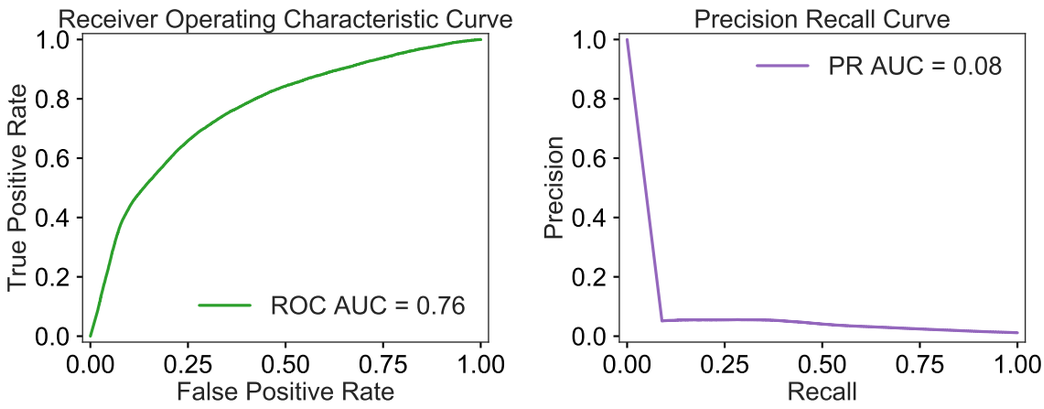

Figure 19: ROC and PR curves for the Gaussian Naive Bayes classifier.

is better than logistic regression. The reason PR AUC is so much smaller than ROC AUC is because the ROC curve has information about true negatives which dominate the confusion matrix in the current problem, especially in the bottom left of ROC curve or for very high values of thresholds.

4.5 Random Forest

For the random forest model, we do not have to scale the features and so we skip the fourth step in the pre-processing steps mentioned in Sec. 4.1. This algorithm is also not that slow so we use the whole dataset and split that into train and test (holdout) datasets with 50%-50% ratio. Table 3 shows the results for various metrics for all different resampling/weighting techniques that we tried. The random

Table 3: Random forest classifier - choosing the best resampling/weighting technique

Resampling/weighting technique	**Class**	**Values**			**PR AUC**	**ROC AUC**	**CPU time (min)**
		Precision	Recall	F1-score			
RandomUnderSampler	0	1.00	0.81	0.90	0.24	0.85	0.17
	1	0.04	0.73	0.08			
SMOTE	0	0.99	1.00	0.99	0.28	0.82	30
	1	0.70	0.15	0.25			
No resampling, using weighting	0	0.99	1.00	0.99	0.33	0.84	10
	1	0.67	0.21	0.32			

forest model has many hyperparameters to be optimized. Using 5-fold cross validation, we optimized all hyperparameters except the number of trees. For Tab. 3, we used 50 trees. In order to improve further, we can increase the number of trees in the random forest. However, the computational cost will go up by doing so. We can play with the trade-off and pick the number of trees such that beyond that number the PR AUC does not increase much. Figure 20 shows that the PR AUC improves by increasing the number of trees but at the cost of computational time. The PR AUC seems to be leveling off asymptotically after around 40-50 trees. In other words, for more than 50 trees, the gain in PR AUC is not as significant as the increase in computational cost is. So, we stick with 50 trees. With 50 trees, and

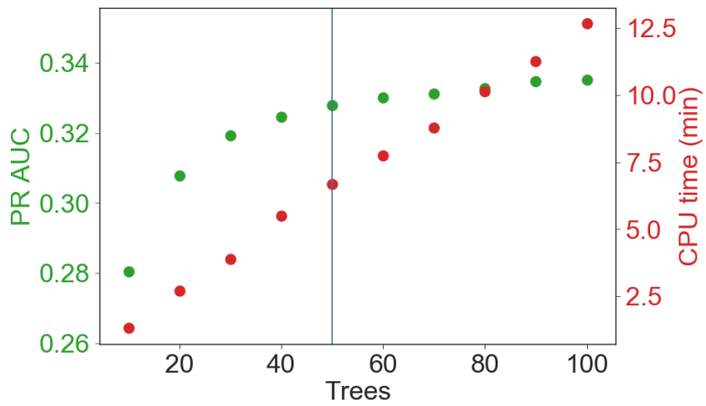

Figure 20: PR AUC (green) and CPU time (red) as a function of the number of trees. The vertical blue line indicates 50 trees.

all other optimized hyper-parameters, we can now try to play with feature selections which is one of a by products of the random forest model. Figure 21 shows

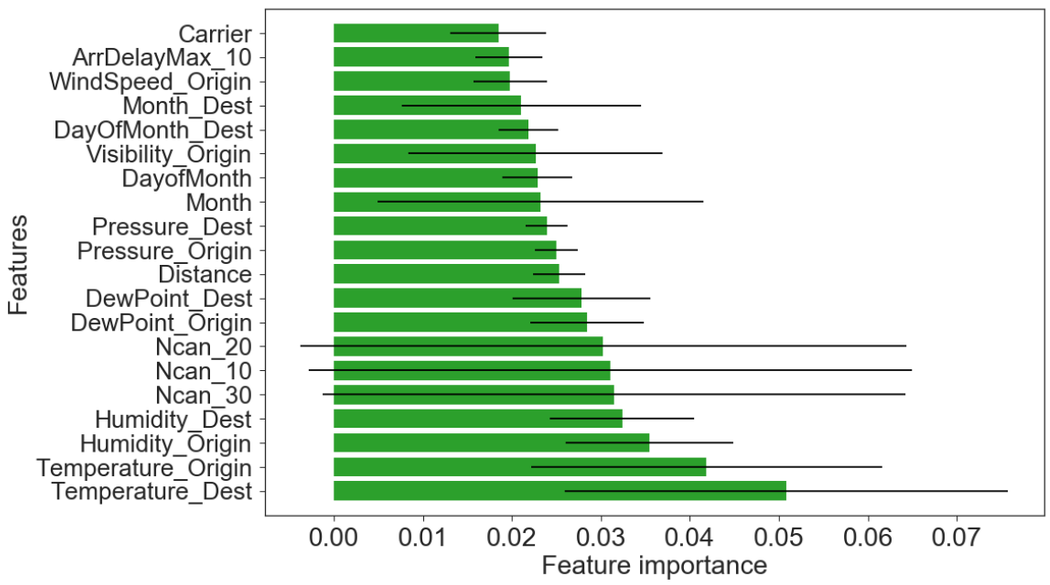

Figure 21: Top 20 features (out of 67) as given by feature importance from the random forest model. Most important features are at the bottom. The horizontal line indicates standard deviation on feature importance.

the top 20 features in terms of their feature importances. Out of these top 20 features, 10 have information about the weather, 4 are historical performance features (which we engineered), 4 are calendar variables and remaining two have information about the flight (Carrier and Distance). The calendar variables do not make much sense here because it is quite possible that Month and Month_Dest have quite the same information. Similarly, DayofMonth and DayOfMonth_Dest are very much the same in terms of information. Moreover, there are other calendar variables such as DayOfWeek which seemed a better predictor than DayOfMonth in Sec. 3.3. In order to find optimal choice of features that maximize PR AUC, we use different values of cutoffs on feature importance and obtain PR AUC. Figure 22 informs us that the PR AUC is almost constant when the cutoff is less than 0.02. However, there seems like a maximum when we remove features whose feature importance values are less than 0.009. We can use the reduced set of features (using cutoff of 0.009) and train the model again. However, before that, there is one

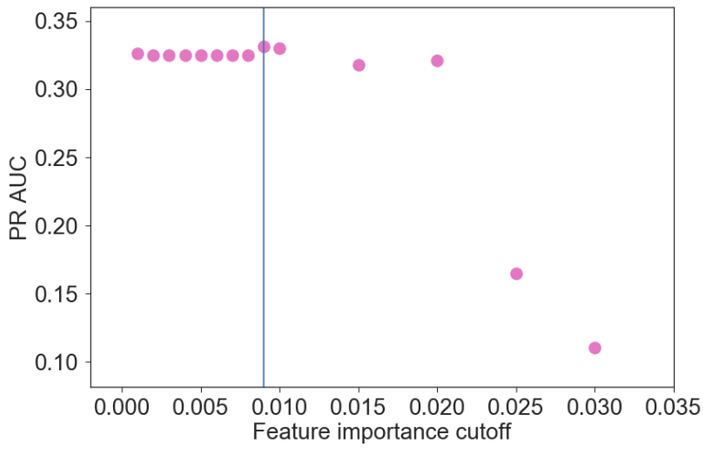

Figure 22: PR AUC as a function of feature importance cutoff.

more item that we have not yet tried, and that is one hot encoding. We should have done it in the beginning but luckily there is no problem because all the features that we are going to remove are either binary or numerical. In one-hot encoding (OHE) we need to worry about only categorical variables. After doing OHE, we create additional features containing 0s and 1s. We find that performing OHE does not help in improving PR AUC. Therefore, there is no advantage of doing OHE. It in fact slows down the computation due to increased number of features (dummy ones). So, finally we only use the reduced set of features (with feature importance greater than 0.009), with weighting training data (no resampling), and use 50 trees, and re-train the model. The PR AUC on the test dataset is found to be 0.33, which is same as what we found with all features. Since we get the same results with less number of features, we stick with less number of features. Figure 23 displays the ROC and PR curves based on the test dataset. The random forest

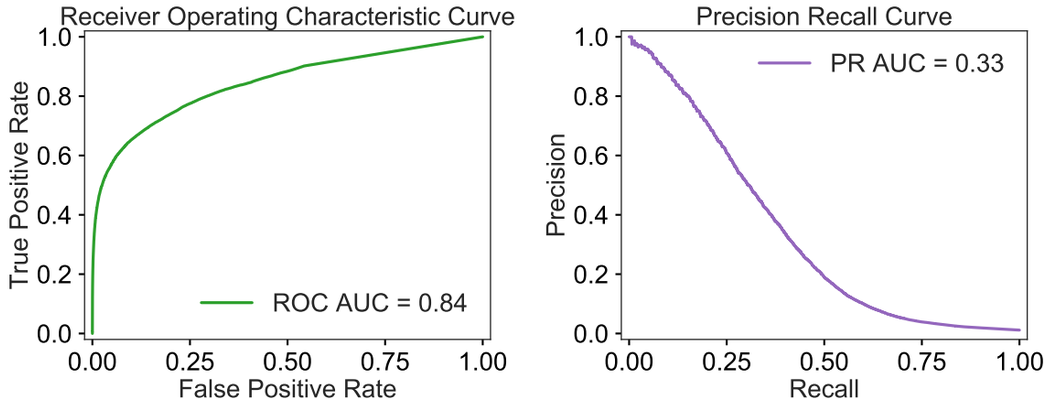

Figure 23: ROC and PR curves for the random forest classifier.

classifier gives us the best model so far. We now go through some more ensemble based approaches to see if we get any improvements.

4.6 Gradient Boosting

For this algorithm also, we do not need to use any feature scaling. However, due to high computational cost we first work with 10% of the data to find optimum hyperparameters and best resampling/weighting technique. Table 4 shows the results for various metrics for all different resampling/weighting techniques that we tried. For gradient boosting, we find the SMOTE to be the best resampling technique. The value of PR AUC is not as good as we got for from the random forest model. However, this low value is based on training the model using only 10% of the data. Before considering the whole data for training, we look at the effect of the number of estimators on PR AUC using the subset of the data. We also study the feature importance, similar to random forest, to select top features. Once we have the optimum model using the subset of the data, we will then use the whole data to train the classifier. We first study the effect of the number of estimators in Fig. 24. Clearly, the PR AUC improves by increasing the number of estimators and it seems to be leveling off at around 60 estimators. So, we will choose 60 estimators for the gradient boosting algorithm. Note that, unlike random forest, the CPU time levels off for more than 50 estimators. With 60 estimators, and all other optimized hyper-parameters, we can now study the feature importance. Figure 25 shows the top 20 features in terms of their feature importances. Out of these top 20 features, 9 have information about the weather, 3 are historical performance features (which we engineered), 4 are calendar variables and remaining four have information about the flight (Carrier, Distance, Origin and Destination). The calendar variables do not make much sense here because it is quite possible that DayOfWeek and DayOfWeek_Dest have quite the same information. Similarly, Month and Month_Dest are very much the same in terms of information. In

Table 4: Gradient boosting classifier - choosing the best resampling/weighting technique

Resampling/weighting technique	Class	Values			PR AUC	ROC AUC	CPU time (min)
		Precision	Recall	F1-score			
RandomUnderSampler	0	1.00	0.77	0.87	0.14	0.83	0.01
	1	0.04	0.73	0.07			
SMOTE	0	0.99	1.00	0.99	0.18	0.80	3
	1	0.64	0.09	0.15			

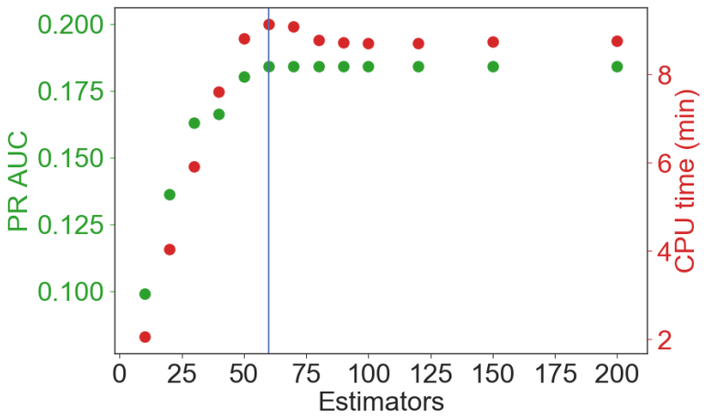

Figure 24: PR AUC (green) and CPU time (red) as a function of the number of estimators. The vertical blue line indicates 60 estimators.

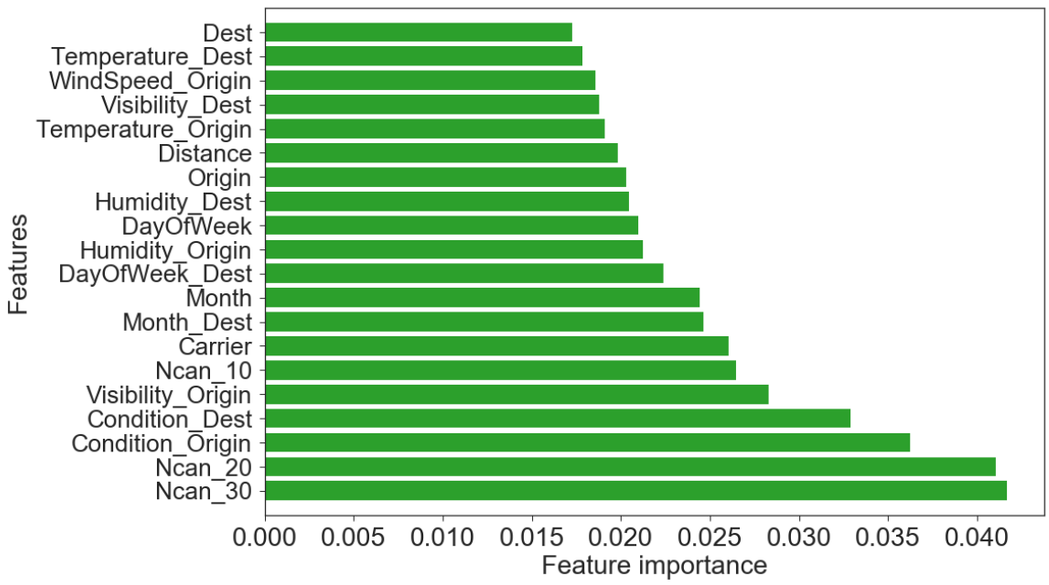

Figure 25: Top 20 features (out of 67) as given by feature importance from the gradient boosting model. Most important features are at the bottom.

order to find optimal choice of features that maximize PR AUC, we use different values of cutoffs on feature importance and obtain PR AUC. Figure 26 suggests that the PR AUC is maximum when we remove features whose feature importance values are less than 0.015. The gradient boosting classification algorithm

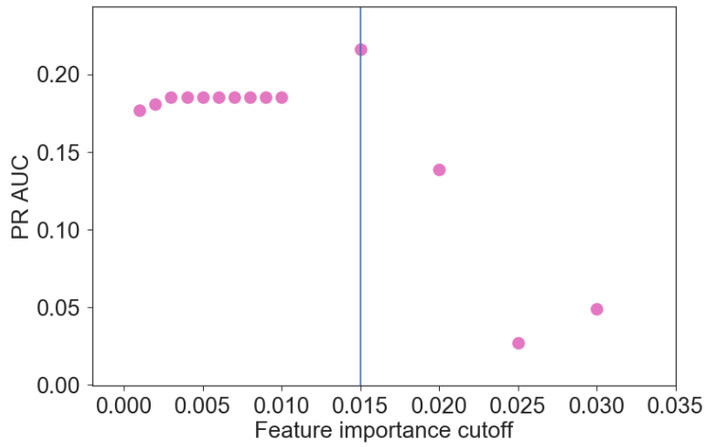

Figure 26: PR AUC as a function of feature importance cutoff.

is computationally very expensive, so we did not try OHE here. Therefore, as a final case for the gradient boosting model, we use selected features whose feature importances are greater than 0.015, and use SMOTE for oversampling, and use 60 estimators. We now use the whole dataset and the resultant model gives PR AUC to be 0.35 based on the holdout dataset. Figure 27 shows the ROC and PR curves based on the test (holdout) dataset. The gradient boosting gives us PR AUC better than that obtained through random forest, but with cost of significant CPU time. It took about 27 hours to train the gradient boosting classifier using the 50% of the whole dataset on an i-Mac with 4 GHz Intel Core i7 processor.

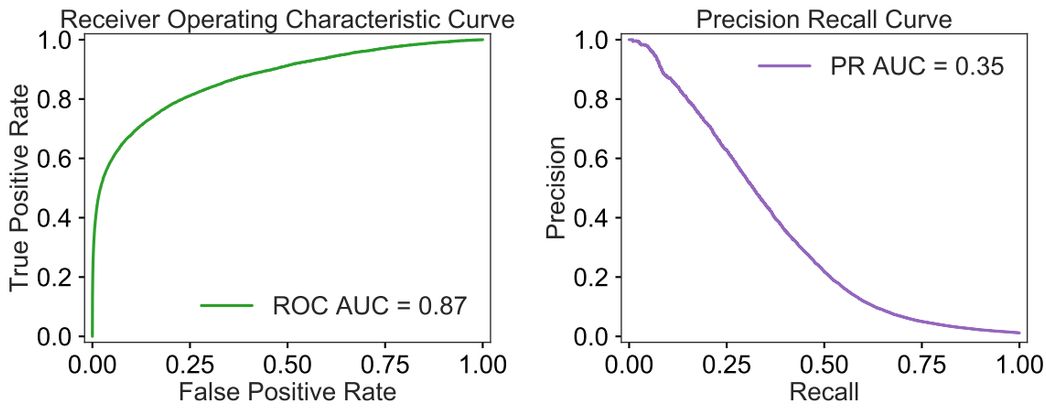

Figure 27: ROC and PR curves for the gradient boosting classifier.

4.7 Extremely Randomized Trees

Compared to random forest, in extremely randomized trees (ET), there is additional level of randomness while the splits are performed. In random forest model, a random subset of features is used and then a feature is decided deterministically using a threshold. In ET, however, thresholds are selected at random for each feature. In ET, we do not to use any feature scaling. This algorithm is not that slow so we work with the whole data. Table 5 shows the results for various metrics for all different resampling/weighting techniques that we tried. For ET, we find that

Table 5: ET classifier - choosing the best resampling/weighting technique

Resampling/weighting technique	**Class**	**Values**			**PR AUC**	**ROC AUC**	**CPU time (min)**
		Precision	Recall	F1-score			
RandomUnderSampler	0	1.00	0.83	0.91	0.26	0.86	0.25
	1	0.05	0.73	0.09			
SMOTE	0	0.99	0.99	0.99	0.30	0.85	18
	1	0.40	0.32	0.35			
No resampling, using weighting	0	0.99	1.00	0.99	0.33	0.86	10
	1	0.59	0.26	0.36			

the weighting is the best approach to handle imbalanced data problem. The value of PR AUC is similar to what we found using the random forest model. We now try to study the effect of the number of estimators in Fig. 28 to see if 50 trees is a good enough number. Clearly, the PR AUC improves by increasing the number of trees but at the cost of computational time. The PR AUC seems to be leveling off asymptotically after around 40-50 trees. In other words, for more than 50 trees, the gain in PR AUC is not as significant as the increase in computational cost is. So, we stick with 50 trees. With 50 trees, and all other optimized hyper-parameters, we can now study the feature importance. Figure 29 shows the top 20 features in terms of their feature importances. Out of these top 20 features, 8 have information about the weather, 3 are historical performance features (which we engineered), 6 are calendar variables and remaining three have information about the flight (Carrier, Distance and Origin). The calendar variables do not make much sense here because it is quite possible that DayOfWeek and DayOfWeek_Dest have quite the same information. Similarly, DayOfMonth and DayOfMonth_Dest are very much the same in terms of information. In order to find optimal choice of features that maximize PR AUC, we use different values of cutoffs on feature importance and

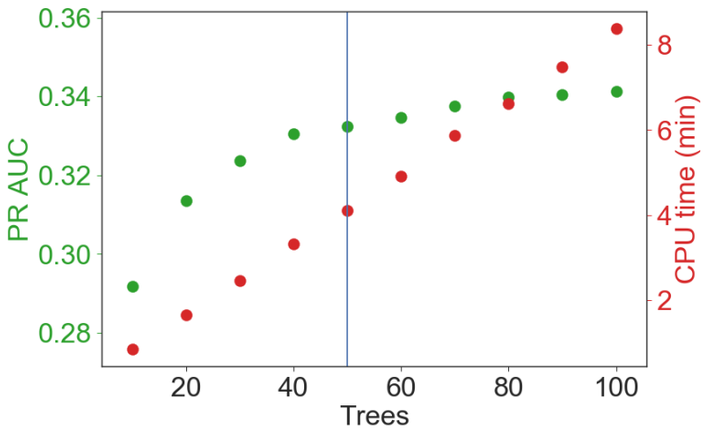

Figure 28: PR AUC (green) and CPU time (red) as a function of the number of trees. The vertical blue line indicates 50 trees.

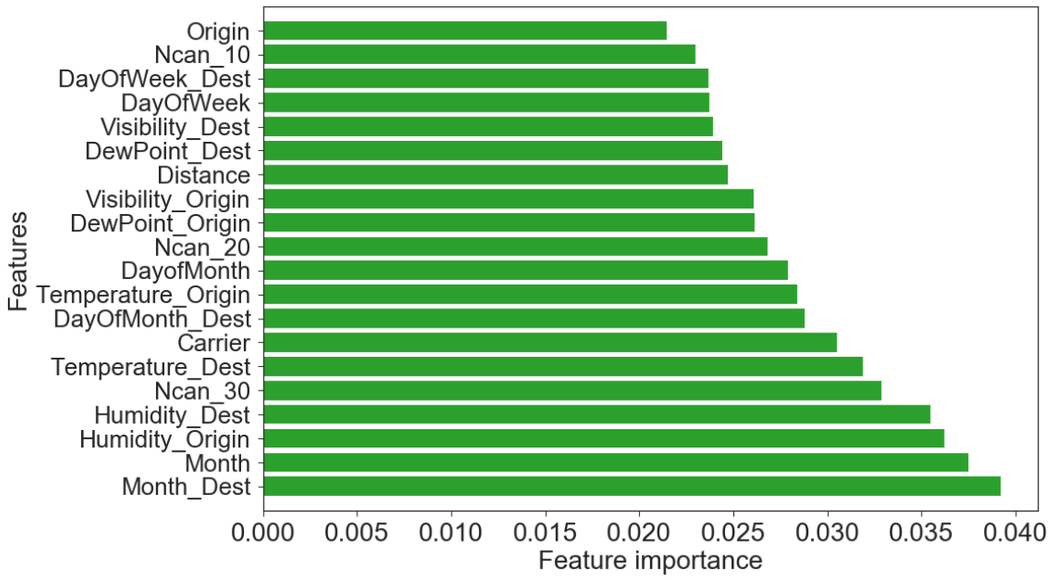

Figure 29: Top 20 features (out of 67) as given by feature importance from the ET model. Most important features are at the bottom.

obtain PR AUC. Figure 30 suggests that the PR AUC is almost constant when the cutoff is less than 0.02. To minimize computational cost by maintaining the PR AUC, we choose the cutoff to be 0.015. We can use the reduced set of features

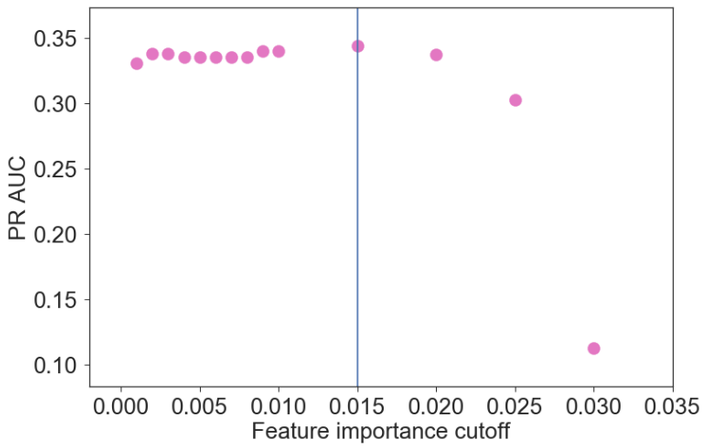

Figure 30: PR AUC as a function of feature importance cutoff.

(using cutoff of 0.015) and train the model again. However, similar to random for-

est, we can also try doing one-hot encoding (OHE). We should have done it in the beginning but luckily there is no problem because all the features that we are going to remove are either binary or numerical. In one-hot encoding (OHE) we need to worry about only categorical variables. After doing OHE, we create additional features containing 0s and 1s. We find that performing OHE improves the value of PR AUC to 0.38, which is the best so far. So, finally for ET, we used the reduced set of features (with feature importance greater than 0.015) and applied OHE to all categorical variables, with weighting training data (no resampling), and used 50 trees. Figure 31 displays the ROC and PR curves based on the test dataset. The

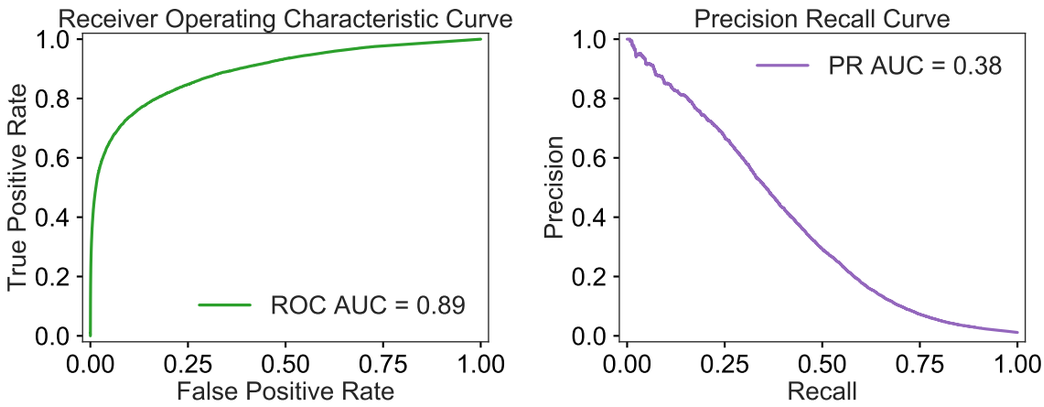

Figure 31: ROC and PR curves for the extremely randomized trees classifier.

ET classifier gives us the best model so far. In the next section, we compare all the models that we have tried in this work and choose the best one.

4.8 Model Comparisons

We have used Logistic Regression, Gaussian Naive Bayes, Random Forest, Gradient Boosting and Extra Randomized Trees classifiers to build a model to predict flight cancellation likelihood. Based on testing the models on the holdout dataset (50% of the whole data), we found different performance of all models. The results of various evaluation metrics scores are shown in Tab. 6 for all models. Other than

Table 6: Comparing all models: Red for the worst and green for the best model.

Model	PR AUC	ROC AUC	Brier Score	Log Loss
Logistic Regression	0.51	0.50	0.01	0.69
Gaussian Naive Bayes	0.08	0.76	0.14	1.55
Random Forest	0.33	0.84	0.01	0.08
Gradient Boosting	0.35	0.87	0.01	0.14
Extremely Randomized Trees	0.38	0.89	0.01	0.08

PR AUC and ROC AUC, we also calculated Brier's score (similar to mean square error), and log loss (cost function in logistic regression). These additional metrics

are suitable for comparing models when we are interested in predicting probabilities or likelihood. For good models, the values of these two metrics should be close to zero. Table 6 shows that logistic regression has the highest value for PR AUC, however the ROC AUC is worst. For the Extremely Randomized Trees (ET) model, we get the best results for both PR AUC and ROC AUC. Moreover, both the Brier score and log loss are minimum for ET model. Therefore, the ET model is the best one in this study. We can also look at the ROC and PR curves for all models in Fig. 32 which corroborates the fact that the ET model performs the best. Though the best model i this work has ROC AUC of 0.89, the PR AUC is pretty

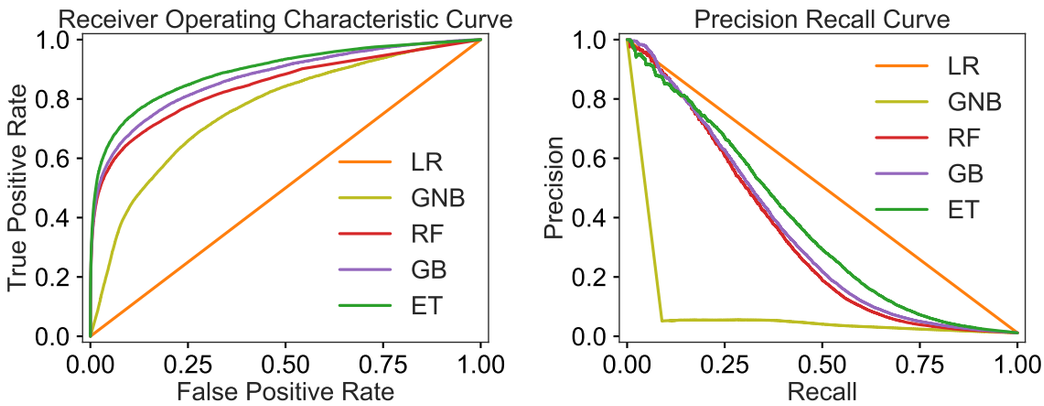

Figure 32: Comparing ROC and PR curves for all models. LR: Logistic Regression, GNB: Gaussian Naive Bayes, RF: Random Forest, GB: Gradient Boosting, ET: Extremely Randomized Trees.

low. As discussed before, the reason for high ROC AUC is due to dominant nature of true negatives (which is absent in case of PR curves).

As an alternative, we also tried to calculate different metrics on a “balanced" test data. In order to get the balanced test data, we picked only the positive examples from the test dataset (holdout dataset) and and equal number of negative examples. This under-sampled test dataset is significantly smaller subset. Calculating various performance metrics on this under-sampled data may not seem a correct approach but we want to see if doing so makes the results different. We compare the results for all models, based on under-sampled test data, in Tab. 7. AUC for both ROC and PR are much better for all models, however, the ET model

Table 7: Comparing all models when test data is under-sampled: Red for the worst and green for the best model.

Model	**PR AUC**	**ROC AUC**	**Brier Score**	**Log Loss**
Logistic Regression	0.75	0.50	0.5	0.69
Gaussian Naive Bayes	0.75	0.76	0.32	2.42
Random Forest	0.88	0.84	0.39	2.56
Gradient Boosting	0.89	0.87	0.39	5.69
Extremely Randomized Trees	0.91	0.89	0.30	1.09

again performs the best with PR AUC = 0.91 and ROC AUC = 0.89. Correspond-

ing ROC and PR curves are shown in Fig. 33. Using under-sampled test data

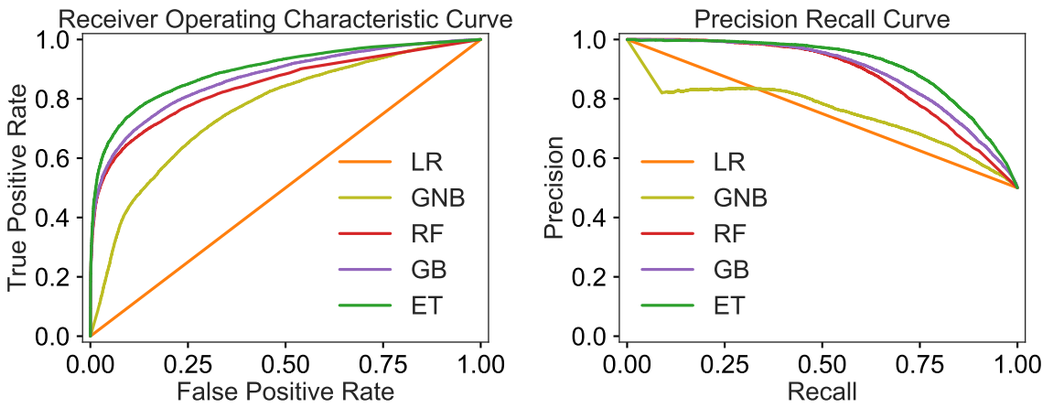

Figure 33: Comparing ROC and PR curves for all models when test data is under-sampled. LR: Logistic Regression, GNB: Gaussian Naive Bayes, RF: Random Forest, GB: Gradient Boosting, ET: Extremely Randomized Trees.

to calculate all performance metrics, we obtained much better values for PR and ROC AUC but the best model is ET model only.

5 Using Model and Recommendations

We now briefly discuss about using the best model to predict flight cancellation likelihood. To make predictions on new data, one would need to filter out some features and then perform OHE on categorical features. Other than these feature processing steps, all other pre-processing steps are stored in the model pipeline. Table 8 contains the list of all 31 features that we need to feed to the ET classification model. The table is sorted by the feature importance values obtained from the ET model, with highest on the top. There are some features such as Ncan_10, Ncan_20 and Ncan_30 that are originally not present in either the flight dataset or weather dataset. We engineered these features by accounting for historical performances of flights. So, one would need to calculate these features beforehand. Some features are categorical, so one needs to perform OHE on them. After OHE, the number of features are 295. Once the features are ready including OHE, the ET model can be run on the new dataset and predictions can be made.

We now discuss little bit about recommending some ideas to potential clients. Since we are only interested in predicting the probabilities of both negative and positive classes, we do not bother about the choice of thresholds or operating point in Figs 32 and 33. The ET model would give us the probabilities of both classes. These probability values can then be used to make categories to indicate different levels of warning or alert messages. For example, if p(cancelled) is the probability of a flight being cancelled, then we can have a categorization like the one shown in Tab. 9. We can recommend such a categorization to different clients, whether it is a booking website or for an app, depending on their business model and requirements.

Table 8: Final set of features for the ET model

Features	Source type	Data type	OHE required?	Description
Month_Dest	Flight data	Categorical	Yes	Month at destination
Month	Flight data	Categorical	Yes	Month at origin
Humidity_Origin	Weather data	Numerical	No	Humidity at origin (%)
Humidity_Dest	Weather data	Numerical	No	Humidity at destination (%)
Ncan_30	Flight history data	Numerical	No	Number of cancellations in last 30 days
Temperature_Dest	Weather data	Numerical	No	Temperature at destination (°F)
Carrier	Flight data	Categorical	Yes	Airline carrier
DayOfMonth_Dest	Flight data	Categorical	Yes	Day of month at destination
Temperature_Origin	Weather data	Numerical	No	Temperature at origin (°F)
DayOfMonth	Flight data	Categorical	Yes	Day of month at origin
Ncan_20	Flight history data	Numerical	No	Number of cancellations in last 20 days
DewPoint_Origin	Weather data	Numerical	No	Dew point at origin (°F)
Visibility_Origin	Weather data	Numerical	No	Visibility at origin (miles)
Distance	Flight data	Numerical	No	Flight distance (miles)
DewPoint_Dest	Weather data	Numerical	No	Dew point at destination (°F)
Visibility_Dest	Weather data	Numerical	No	Visibility at destination (miles)
DayOfWeek	Flight data	Categorical	Yes	Day of week at origin
DayOfWeek_Dest	Flight data	Categorical	Yes	Day of week at destination
Ncan_10	Flight history data	Numerical	No	Number of cancellations in last 10 days
Origin	Flight data	Categorical	Yes	Origin airport
Dest	Flight data	Categorical	Yes	Destination airport
WindDirection_Origin	Weather data	Numerical	No	Wind direction at origin (degrees)
WindSpeed_Origin	Weather data	Numerical	No	Wind speed at origin (mph)
WindDirection_Dest	Weather data	Numerical	No	Wind direction at destination (degrees)
Condition_Origin	Weather data	Categorical	Yes	Weather condition at origin
WindSpeed_Dest	Weather data	Numerical	No	Wind speed at destination (mph)
Condition_Dest	Weather data	Categorical	Yes	Weather condition at destination
Pressure_Origin	Weather data	Numerical	No	Pressure at origin (inHg)
CRSDepHr	Flight data	Categorical	Yes	Scheduled departure hour
Pressure_Dest	Weather data	Numerical	No	Pressure at destination (inHg)
CRSArrHr	Flight data	Categorical	Yes	Scheduled arrival hour

Table 9: An example of a possible categorization of alerts messages to indicate flight cancellation chances

Probability ranges	Alert messages
$p(\text{cancelled}) > 0.9$	Extremely high chance
$0.75 < p(\text{cancelled}) \leq 0.9$	Very high chance
$0.5 < p(\text{cancelled}) \leq 0.75$	High chance
$0.25 < p(\text{cancelled}) \leq 0.5$	Moderate chance
$0.1 < p(\text{cancelled}) \leq 0.25$	Low chance
$p(\text{cancelled}) \leq 0.1$	Very low chance

6 Assumptions and Limitations

The flight and weather data are in time series format. It is likely that there are some correlations for a given flight between two times. In this project, we assume that all flights are independent, i.e. there are no correlations. This assumption makes the problem little bit easier to implement various models that we used. Other than this assumption, we have several limitations in the data which might have reduced the robustness of the machine learning model that we developed.

1. We have considered only 20 airports in this project due to weather data provider's API restrictions. Though 20 airports broadly cover the whole US, a lot of information will not be learned by the model due to the absence of all airports in the dataset.
2. For new data, the prediction of the model will depend on how good the prediction of the weather is, say after 3 days. In other words. if we want to predict the likelihood of the flight cancellation for a flight which is scheduled after 3 days, we would need to know the weather after 3 days (which we do not know "accurately" today). This means that the model will predict better if the weather prediction is better or if we are trying to predict the flights not far ago than the scheduled departure.
3. Apart from knowing whether a flight was cancelled, we also have information about the cancellation codes such as A, B, C, D. Most probably these codes correspond to different reasons or factors for cancellation. However, other than just knowing these codes, we do not know the exact meanings of these codes. If the exact meanings were known, we would have built a multi-class classification model. Lack of this information restricted us to develop a binary classifier only.
4. We found that SunCountry Airline (IATA code: SY) data is missing in the original flight dataset which we acquired from the Bureau of Transportation Statistics. We checked for the absence of only this airline from a personal travel experience. It is possible that some other data might also be missing in the original dataset. Therefore, we emphasize that the analysis carried out in this project is only based on the data source that we mentioned here.

7 Other Data and Future Work

Other than the original flight data, weather data and historical performance data, we can also acquire or feature engineer more datasets which might enhance the predictive power of the machine learning model. Following is a list of some possibilities:

1. Knowing the airport name, we can extract informations such as number of

runways, runway length (and width), airport type, airport infrastructure, airport capacity etc.., and create new features.

2. Similarly, knowing the airlines names, we can extract airlines ratings, their stock market performance, their revenue and assets, reputations etc.., and create new fields.

3. In terms of historical performances, we only engineered some features for airline historical performances. We can also engineer some features to get airport historical performance data.

4. We can also get data from social networking sites and news media about the sentiment for each airlines and airports. The sentiment analysis can then be used to create more features.

5. Datasets containing world events such as catastrophic natural destruction, terror attacks, political movement, sport tournaments, etc.. can also be acquired and used to merge with our original datasets.

Each one of these ideas can be difficult if the data is not easily accessible. However, we believe that the predictive power of the model will be improved by incorporating these factors.

8 Conclusions

In this project, we first explored the original flight dataset along with the merged weather datasets and engineered historical performance variables, to understand their influence on the flight cancellation rates. Top 20 airports (in terms of the number of flights operating) were considered in this work for the years 2015 - 2016. The overall cancellation rate was about 1.28%, which is not a large number but that is what makes the project more interesting.

8.1 Data Exploration Conclusions

We explored 6 types of informations available in the datasets and found that most of them influence cancellation rates. Following is a quick summary of the exploratory data analysis:

1. **Calendar variables:** Most flights were cancelled in the winter months and least in the falls months. We also found that the cancellations are worst in the end and beginning of a week.
2. **Airports:** Usually the airports in the east coast had higher cancellation rates as compared to the rest of the nation. LaGuardia and Boston airports topped

the list and Salt Lake City and Seattle were bottom in the list of cancellation rates.

3. **Airlines:** Regional airlines like Envoy Air and ExpressJet Airlines had higher cancellation rates whereas Delta Airlines, Frontier Airlines and Alaska Airlines performed the best in terms of flight cancellations.
4. **Flight distance:** The cancellation rates were higher for shorter distance flights and lower for longer distance flights.
5. **Weather factors:** Snowy weather is worst and clear sky or even cloudy weather conditions are best in terms of flight cancellation. We also found an increasing trend for cancellation rates as humidity and wind speed increased at both origin and destination airports. Cancellation rates were higher when the temperatures were less than 40°F, as compared to temperatures greater than 40°F. Finally, we discovered an interesting trend with respect to the wind direction. The cancellation rates were close to 1% when the wind direction at both origin and destination airports were from East - to - South - to - West. However, we observed a jump in the cancellation rate as the wind direction approached close to North bound.
6. **Historical performances:** In general, we found smaller cancellation rates for flights that got cancelled or diverted less in the last 10-30 days. However, when all the flights got cancelled in the last 10-30 days, the cancellation rate was slightly higher as compared to the cases when all flights were not cancelled. We investigated the cancellation rates for temporary flights (flights that ran only once in last 10-30 days) and found that temporary flights were more likely to cancelled than routine flights. Finally, we also looked at the effect of the history of departure and arrival delays on the cancellation rate of the flight in question and found some interesting trends for shorter range of delays.

8.2 Modeling Conclusions

After exploring all datasets, we used five different supervised classification algorithms (Logistic Regression, Gaussian Naive Bayes, Random Forest, Gradient Boosting and Extremely Randomized Trees) to train the predictive model by using 50% of the whole data. The remaining 50% was used to evaluate the model.

1. Using various performance evaluation metrics, we found that the Extremely Randomized Trees classifier gives the the best model performance.
2. We achieved the ROC AUC to be about 0.89. The AUC for PR was not great (about 0.38) but when we ran the model on a balanced test dataset (which we obtained by under- sampling the test data), we got PR AUC to be about 0.91.
3. In terms of features, we found that we do not need all the features that we collected from different data sources. In fact, we needed only 31 features, of

which 12 were from flight data, 16 were from weather data and 3 were from flight historical performances data (which we engineered).

The model is decent enough to be used for predicting the likelihood of flight cancellations. As mentioned in Sec. 6, there are some limitations in the current model such as lack of complete data, poor predictions for flights that are not scheduled to depart very soon in future, and also assuming no time-correlation between flights. By overcoming these limitations and incorporating some of the ideas mentioned in Sec. 7, the model can further be improved and we leave it for the future work.