Random forests

# Introduction:

Random forest is an ensemble of many individual decision trees. It builds a large collection of de-correlated trees which are noisy but unbiased, and averages them to reduce the variance. Random forest obtains a class vote from each tree, and then classifies a sample using majority vote. Let be class prediction of the tree, then the class obtained from random forest, , is

= majority vote

# Decision trees properties

1. Low bias
2. High variance
3. Prone to overfitting

So, the idea of Random Forests is we build a bunch of decision trees by reducing its variance without affecting bias too much. We select a number of samples and then run decision trees on them.

# Requirements

1. Python2 / Python3
2. NumPy [pip install numpy]
3. SciPy [pip install scipy]
4. Scikit [pip install -U scikit-learn]

# working:

1. Assume number of cases in the training set is N. Then, sample of these N cases is taken at random but *with replacement*. This sample will be the training set for growing the tree. This method is called bagging.
2. If there are M input variables, a number m<M is specified such that at each node, m variables are selected at random out of the M. The best split on these m is used to split the node. The value of m is held constant while we grow the forest.
3. Each tree is grown to the largest extent possible and there is no pruning.
4. Predict new data by aggregating the predictions of the n-tree trees (i.e., majority votes for classification, average for regression).

Answer by Sebastian Raschka

https://www.quora.com/How-does-the-random-forest-model-work-How-is-it-different-from-bagging-and-boosting-in-ensemble-models

# Steps

1. Dealing with Missing Sample data [training data and results]
   1. Imputation (calculates median for missing feature value)

from sklearn.preprocessing import Imputer

i = Imputer(strategy=’median’)

i.fit(sample\_data)

imputed\_sample = i.transform(sample\_data)

imputed\_test = i.transform(test\_data)

* 1. If features are missing entirely.
     1. Remove that feature
     2. Add zeroes for every variable

Import numpy as np

Mask – mp.all(np.isnan(sample\_data), axis=0)

Sample\_data[:,mask] = 0.0

1. Build our training model

From sklearn.ensemble import RandomForestRegressor

Est = RandomForestRegressor(n\_estimators = 10,

max\_features = ’auto’,

max\_depth = None,

min\_samples\_split = 2,

min\_samples\_leaf = 1,

min\_weight\_fraction\_leaf = 0,

max\_leaf\_nodes = None,

n\_jobs = 1)

est.fit(training\_data, training\_results)

predicted\_results = est.predict(test\_data)

1. Evaluating the results
   1. Uses coefficient of determination ()
   2. Counter-intuitively, the result can be negative!
   3. Occurs when mean correct test result is better predictor than the model

Est.score(test\_data, correct\_results)

1. Cross Validation
   1. 80% training set 20% test set
   2. Use multiple estimators (use different parameters)
   3. Use best estimator on test set
2. Out-of-bag estimates
   1. If you got a random sample i.e. the same size or smaller than original set of samples then you are going to have some samples that miss out that are not in training data that are used for any given tree
   2. If you are training 100 trees, then chances are higher that for every single sample in your training data there will be at least one tree that sample will not be used as a training data for that tree.
   3. We use them to predict result for that sample and we do that for every sample as long as you have enough trees which gives an estimate of how well your random forest is performing
   4. Not as good as performing proper validation but faster and easier
3. Feature importance:
   1. A feature used near the top of a decision tree affects in larger proportion of samples than one used near the leaves
   2. Quantify the importance of each feature in each tree and avg them to get their importance

Est.feature\_importance\_

1. Accessing the trees:
   1. Est has attribute called “estimators\_” to get individual trees
   2. “export\_graphviz” used to visualize the trees

# Tuning parameters:

1. Number of trees
   1. More the better
   2. Diminishing results (once you go above a certain number of trees the predictive power to algorithm doesn’t improve much)
   3. Slower to construct
2. Number of features
   1. More features reduce bias
   2. But increases the correlation of trees
3. Height of the tree
   1. Too deep, risk of overfitting
   2. By making a forest we reduce the risk but we don’t eliminate it
   3. How deep the trees are going to be?

# Advantages of Random Forest

[Few tuning parameters, Good performance, Don’t need standardized training data, Built in cross validation, Quantify feature importance]

1. This algorithm can solve both type of problems i.e. *classification and regression* and does a decent estimation at both fronts.
2. The power of handle large data set with higher dimensionality. It can handle thousands of input variables and identify most significant variables so it is considered as one of the dimensionality reduction methods. Further, the model outputs Importance of variable, which can be a very handy feature (on some random data set).
3. It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.
4. It has methods for balancing errors in data sets where classes are imbalanced.
5. The capabilities of the above can be extended to unlabelled data, leading to unsupervised clustering, data views and outlier detection.
6. Random Forest involves sampling of the input data with replacement called as bootstrap sampling. Here one third of the data is not used for training and can be used to testing. These are called the out of bag samples. Error estimated on these out of bag samples is known as out of bag error. Study of error estimates by Out of bag, gives evidence to show that the out-of-bag estimate is as accurate as using a test set of the same size as the training set. Therefore, using the out-of-bag error estimate removes the need for a set aside test set.

# Disadvantages of Random Forest

1. It surely does a good job at classification but not as good as for regression problem as it does not give precise continuous nature predictions. In case of regression, it doesn’t predict beyond the range in the training data, and that they may over-fit data sets that are particularly noisy.
2. Random Forest can feel like a black box approach for statistical modelers – you have very little control on what the model does. You can at best – try different parameters and random seeds!

Random Forests applied to Airline Delays Problem

# Motivation

According to the ‘Bureau of Transportation Statistics (BTS)’, approximately **twenty percent** of the entire scheduled commercial flights are delayed costing multi-billion dollars per year. BTS has categorized airline delays into five main causes, which are ***air carrier, extreme weather, National Aviation System, late-arriving aircraft and security***. Weather ‘s percentage share accounts for about 40% of total delay minutes. Thus, study on the influence of inclement weather on airline delays is essential for efficient flight operations.

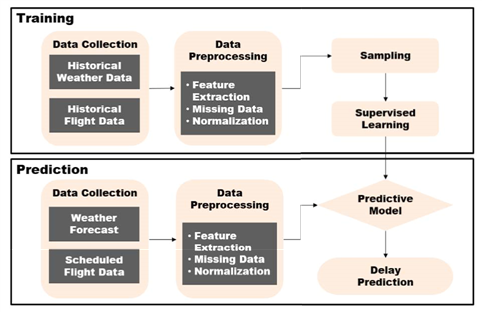
# Why Machine learning

1. Volume of historical flights and weather data are too large to analyse analytically.
2. Correlation among factors are extremely complicated and highly non-linear.
3. Machine learning is a clever method to analyse such data.

# Methodologies

1. Sampling technique: (SMOTE)
   1. The training data is imbalanced because the number of on time flight is three to four time more than delayed flights.
   2. To address issues emerged from imbalanced data, examples of minority class should be generated synthetically to adjust the distribution between majority and minority class.
   3. Synthetic Minority Over-Sampling Technique (SMOTE) is a powerful solution to imbalanced dataset addressing drawbacks of the over-sampling and under-sampling.
   4. SMOTE is an over-sampling approach that creates synthetic minority class examples.
   5. The minority class is over-sampled by introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbours.
2. Supervised learning classifiers:
   1. Random Forests
   2. Decision Trees
   3. AdaBoost
   4. kNN
3. Model Evaluation:
   1. 10-fold Cross Validation
   2. Receiver Operating Characteristics [ROC]

# Prediction Model



## Data collection:

1. Bureau of Transportation Statistics (BTS) - Airline On-time Performance dataset
2. National Oceanic and Atmospheric Administration (NOAA) – weather (wind, cloud height, visibility, temperature, pressure, precipitation, etc.)
3. FlightStats APIs
4. World Weather Online API’

## Data Pre-processing:

Following the rules of BTS, flights that arrive at the gate within 15 minutes of the scheduled time are considered as on-time. Cancelled and diverted flights in the training set are deemed as delayed. To deal with missing values in weather data, linear interpolation is used with two adjacent known values.

Interpolating , linear interpolation equation

Fields extracted from BTS dataset:

1. Quarter of Year
2. Month
3. Day of Month
4. Day of Week
5. Departure and Arrival Schedule in Local Time
6. Arrival Delay Indicator: 0 if actual arrival time minus scheduled arrival time is less than 15 minutes, 1 if actual arrival time minus scheduled arrival time is greater than or equal to 15 minutes

Fields weather fields:

1. Wind Speed Rate [*m/s*]
2. Visibility [*m*]
3. Precipitation [*mm*]
4. Snow Depth [*cm*]
5. Snow Accumulation [*cm*]
6. Weather Intensity Code
   1. Light
   2. Moderate
   3. Heavy
   4. Vicinity
7. Weather Descriptor Code
   1. Shallow
   2. Partial
   3. Patches
   4. Low Drifting
   5. Blowing
   6. Showers
   7. Thunderstorms
   8. Freezing
8. Precipitation Code
   1. Drizzle
   2. Rain
   3. Snow
   4. Snow Grains
   5. Ice Crystals
   6. Ice Pellets
   7. Hail
   8. Small Hail and/or Snow Pellets
   9. Unknown Precipitation
9. Obscuration Code
   1. Mist
   2. Fog
   3. Smoke
   4. Volcanic Ash
   5. Widespread Dust
   6. Sand
   7. Haze
   8. Spray
10. Other Weather Code
    1. Well-Developed Dust/Sand Whirls
    2. Squalls
    3. Funnel Cloud, Tornado, Waterspout
    4. Sandstorm
    5. Dust storm
    6. Combination Indicator Code

# Classification Model:

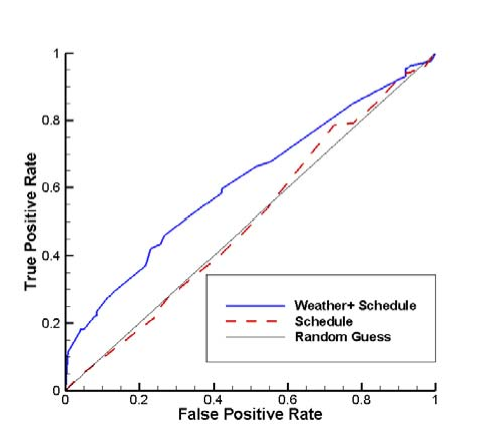
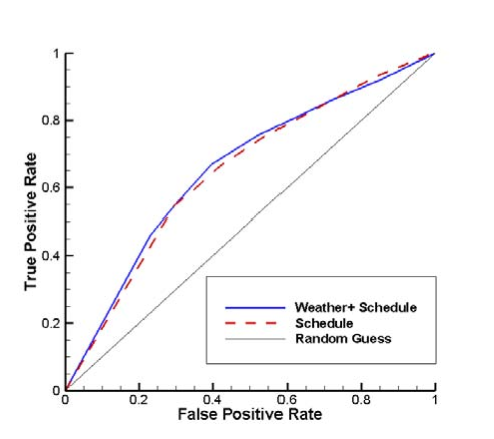
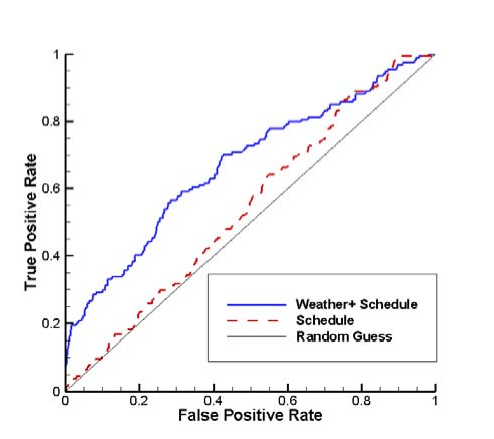
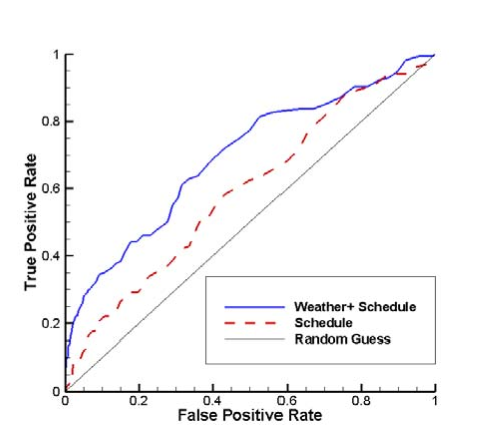
Parameters for each classifier were tuned by trial-and-error

|  |  |
| --- | --- |
| No of trees (Random Forests) | 100 |
| Learning rate (Adaboost) | 1 |
| Max number of estimators (Adabost) | 100 |
| No of neighbours (kNN) | 6 |
| Depth of tree (decision tree) | 7 |

# RESULTS AND ANALYSIS

## Impact of weather data

1. Weather data distinctly improves predictive ability of RF,AB and DT. (closer the curve to diagonal, lesser is the accuracy)
2. Without weather data AB and DT couldn’t determine whether flight will be delayed or not
3. kNN’s gap between two curves with and without weather data is quite small compared to other classifiers



## Impact of Sampling techniques

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | with Sampling Techniques | | without Sampling Techniques | |
| Classifier | Accuracy (%) | Timeelapsed(sec) | Accuracy (%) | Timeelapsed(sec) |
| Random Forest | 81.37 | 8 | 83.40 | 9 |
| AdaBoost | 78.05 | 12 | 83.21 | 12 |
| kNN | 61.69 | 2 | 82.42 | 2 |
| Decision Trees | 77.02 | 0 | 82.84 | 0 |

1. This indicates that accuracy is better in classifier without sampling. However, this doesn’t mean that sampling is a bad choice.
2. Classifiers are biased towards on-time class when trained on imbalanced data wiz easier for it to predict. Hence, more likely to classify delayed flight as on time.

Random forests predictive performance:

Confusion matrix of random forests

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | with Sampling Techniques | | without Sampling Techniques | |
|  | Predicted On-time | Predicted Delay | Predicted On-time | Predicted Delay |
| Actual On-time | 6838 | 418 | 7178 | 78 |
| Actual Delay | 1231 | 346 | 1388 | 189 |

1. Even though diagonal sum is reduced i.e. Model’s prediction accuracy, Model’s True Positive Rate (TPR) is increased, i.e. minority class recognition was Improved.
2. Comparison of FPR (on time predicted as delayed) and FNR (delayed predicted as on time) revealed that model is more likely to classify delayed flight as on time even after sampling. because we didn’t consider other causes of weather.

## Comparison between methods

1. Random forests performed best.
2. In case of imbalanced data, model can perform much better without paying special attention to minority class, hence it is more appropriate to use Receiver Operating Characteristics (ROC) curve and area under ROC curve

0.0

0.2

0.4

0.6

0.8

1.0

False Positive Rate

0.0

0.2

0.4

0.6

0.8

1.0

TruePositiveRate

Random Forests (AUC = 0.68)

AdaBoost (AUC = 0.66)

K-Neighbors (AUC = 0.66)

Decision Tree (AUC = 0.64)

1. Random forests have largest AUC also farthest from diagonal
2. kNN has poorest accuracy and lowest AUC. In confusion matrix, it had high TPR and FPR at the same time. ROC is close to diagonal for high threshold values.
3. For decision trees, TPR and FPR were low. ROC is below the diagonal line at the top-right corner. prediction performance was worse than random guess for low threshold values and sensitive to threshold values.

## Assessment of performance on test data:

|  |  |  |
| --- | --- | --- |
| Classifier | Accuracy (%) | Decrease (%) |
| Random Forest | 80.36 | 1.01 |
| AdaBoost | 71.43 | 6.62 |
| KNN | 35.71 | 25.98 |
| Decision Trees | 64.29 | 12.73 |

1. All classifiers performance degraded on unseen data, they overfitted modelling every minor variation in input.

Confusion Matrix of Random Forest with Forecast Horizons

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 5 days Forecast Horizon,  Accuracy = 26.79% | | 1 Day Forecast Horizon,  Accuracy = 30.36% | | 0 Day Forecast Horizon,  Accuracy = 80.36% | |
|  | Predicted On-time | Predicted Delay | Predicted On-time | Predicted Delay | Predicted On-time | Predicted Delay |
| Actual On-time | 7 | 39 | 7 | 39 | 44 | 2 |
| Actual Delay | 2 | 8 | 0 | 10 | 9 | 1 |

1. The model’s predictive performance is drastically lowered due to uncertainty in forecast.
2. The results with the actual weather exhibited higher accuracy as uncertainty in forecast had dropped out.
3. Nonetheless, actual weather did not provide perfect prediction. Prediction error can arise from two main sources. First one is the limitation of the current model. The other one is that delayed flights could not be captured by the model since delays are caused by non-weather-related factors.

# Conclusion:

This study proposed a prediction model enabled to classify airline delays caused by inclement weather condition. Supervised machine learning algorithms implemented in this study includes RF, AB, kNN and DT. Because the data was imbalanced, the combination of SMOTE and random under sampling were applied. The model’s prediction performance on the validation set and the test set was analysed. If the costs of false positive and false negative are taken into account, preferred performance of classifiers could be clearly determined. Then it could be a solid foundation for a decision support tool for predicting aircraft arrival.

Literature survey

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Paper no | Title | Authors | Algorithms | Pros and Cons |
| *1* | Prediction of Weather-induced Airline Delays  Based on Machine Learning Algorithms | Sun Choi, Young Jin Kim, Simon Briceno and Dimitri Mavris | SMOTE, Decision Trees, Random Forests, AdaBoost, kNN | Good selection of algorithms, dealing with oversampling, performance comparison |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |