Weather Classification with R

Big Data II
Final Project

Team Prof.X: Zi Long, Lin Gao, Jin Zhou, Huijue Ye, Tong Wu

1 Introduction

1.1 Background

Weather forecasting is the prediction of what the atmosphere will be like in a place by using technology and scientific knowledge to make weather observations. In other words, it is a way of predicting things like cloud cover, rain, snow, wind speed, and temperature before they happen. Weather forecasts are important since there is a vast variety of end uses. Forecasts based on temperature and precipitation are important to agriculture, and therefore to traders within commodity markets. On an everyday basis, many use weathers forecasts to determine what to wear on a given day. Since outdoor activities are severely curtailed by heavy rain, snow and wind chill, forecasts can be used to plan activities around these events and save them.

1.2 Questions and hunches

In our research, we mainly focus on how to use certain input variables we have got to forecast weather type. Before performing certain methods, we have some hypotheses on the pattern of each kind of weather. For instance, the fogging situation would have the low visibility and high humidity; rainy weather may have low dew point, low temperature, and low apparent temperature; the sunny and clear situation would have a high apparent temperature, low humidity, and high visibility; cloudy situation may have a high temperature, pretty low apparent temperature, and low air pressure, etc. Some of the variables may have more decisive power in forecasting than others. We will show these results in later parts of the reports.

2 Data sources

Our dataset is sourced from Kaggle, and the original source is darksky api. These data include hourly weather data in London from 2011-2014. The data contains 21165 observations and 12 variables.

2.1 variables list:

- visibility: numeric, the average visibility in miles, capped at 10 miles.
- windBearing: numeric, the direction that the wind is coming from in degrees, with true north at 0° and progressing clockwise. (If windSpeed is zero, then this value will not be defined.)
- temperature: numeric, the air temperature in degrees Fahrenheit.
- dewPointumeric, the dew point in degrees Fahrenheit.
- time: the UNIX time at which this data point begins.
- pressure： numeric, the sea-level air pressure in millibars, with 13 missing values (0.06%)
- apparentTemperature: numeric, the apparent (or “feels like”) temperature in degrees Fahrenheit.
- windSpeed: numeric, the wind speed in miles per hour.
- humidity: numeric, the relative humidity, between 0 and 1, inclusive.

- precipType: categorical, the type of precipitation occurring at the given time. If defined, this property will have one of the following values: “rain” and “snow”.
- icon: categorical, a machine-readable text summary of this data point, suitable for selecting an icon for display. If defined, this property will have one of the following values: clear-day, clear-night, rain, snow, sleet, wind, fog, cloudy, partly-cloudy-day, or partly-cloudy-night.
- summary: A human-readable text summary of this data point.

2.2 Data wrangling and preparation

In the data preparation process, we use Trifacta Wrangler to clean the data. The wrangling process starts with creating a new workflow and importing data into it. The dataset is of high quality from beginning with, we notice that there is a small proportion of missing data in pressure and there are no missing data in other variables. Also, many variables show reasonable distribution, which means there are no extreme values in all columns.

First, we delete the column of time since time does not provide information to predict whether there will be rain or snow. Then we delete rows with a missing value in pressure. We also generate variable “month” from variable “time”. Finally, we modify categories in the icon. Since we delete time in our dataset, we should also modify the icon column, replacing ‘clear-day’ and ‘clear-night’ with ‘clear’ and replacing ‘partly-cloudy-day’ and ‘partly-cloudy-night’ with ‘partly-cloudy’.

2.3 Descriptive Statistics and data visualizations

As we make some preparation before constructing the model, first we choose the variables that can be considered as relevant and important variables. We first narrow the variables down to nine that are important by relative research on this topic. They are visibility, windBearing, temperature, dewPoint, pressure, apparentTemperature, windSpeed, humidity, precipType, Month and icon.

Then, in order to discover the structure and possible relationship in the dataset, we use descriptive statistics, plots, and visualizations to show descriptive statistics for relevant and important variables, including: **Descriptive table**, to show the minimum, maximum, average (mean, median, mode) and standard deviation / variance of important variables; **Box-and-whisker plots**, including eight variables with x axis as precipType.

Table 2-1 Descriptive Table

	Mean	Std. Dev.	Minimum	Maximum	Median	Mode
visibility	11.27	3.00	0.18	16.09	12.30	13.07
windBearing	197.92	89.47	0.00	359.00	219.00	239.00
temperature	10.54	5.51	-3.86	30.28	10.07	9.13

dewPoint	6.63	4.90	-9.68	19.88	6.64	8.11
pressure	1014.09	11.03	975.74	1043.20	1014.73	1018.15
apparentTemperature	9.31	6.68	-8.88	30.60	10.07	10.33
windSpeed	3.95	2.02	0.06	14.80	3.73	3.25
humidity	0.78	0.14	0.25	1.00	0.81	0.90

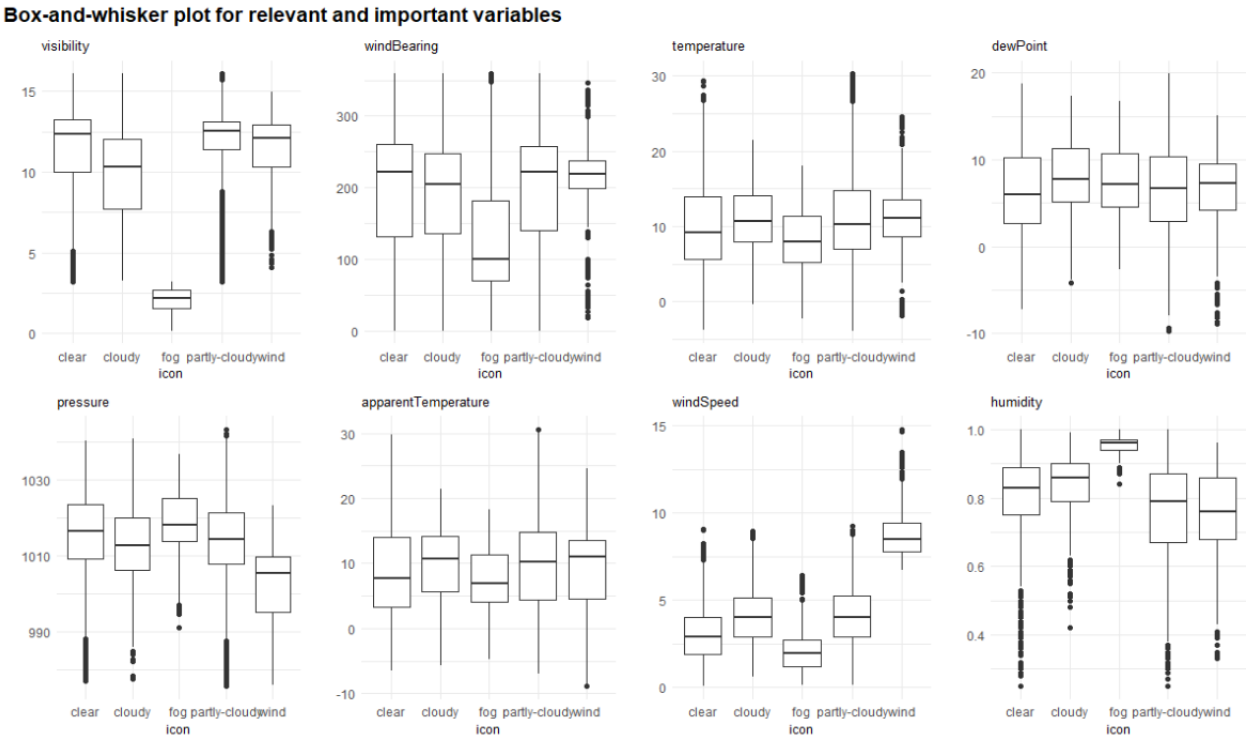


Figure 2-1 Box-and-whisker plots

From the figure 2-1, we can find that variables behave differently in different weather from the aspects of ranges and average values. As for visibility and windBearing, they have higher average visibility in rain, but the range of values become larger when precipType is snow. Variables–temperature, apparentTemperature and dewPoint–have higher mean value and larger ranges when the weather type is rain. When it comes to pressure, the range becomes wider when it is snow but with similar average

value in rain and snow. Both windSpeed and humidity have similar mean and range in rain and snow. In addition, visibility, temperature, windSpeed and humidity has many outliers.

2.4 Target variable(s) chosen and justification

The target variable we choose is icon, which is the weather type at observed time.

In this project, we want to predict weather type, whether it is clear-day, clear-night, rain, snow, sleet, wind, fog, cloudy, partly-cloudy-day, or partly-cloudy-night, by other weather data, so we choose icon as our target variable. According to common sense, temperature, wind, humidity, pressure and precipitation are usually used by people to predict weather type. So, we picked weather type, which is named as icon, as our target variable, and selected visibility, windBearing, temperature, dewPoint, pressure, apparentTemperature, windSpeed, humidity, Month and precipType, as our potential predictor variables.

2.5 Management of outliers/imbalanced classification/data leakage

Then we checked if we need to avoid outliers and target leakage.

Our dataset suffers from outliers, we use DBSCAN (eps = 10, minPts = 50) to detect outliers then we remove the whole row of outliers. 869 outliers were removed from the data, accounting for 4.1% of the data, which is reasonable.

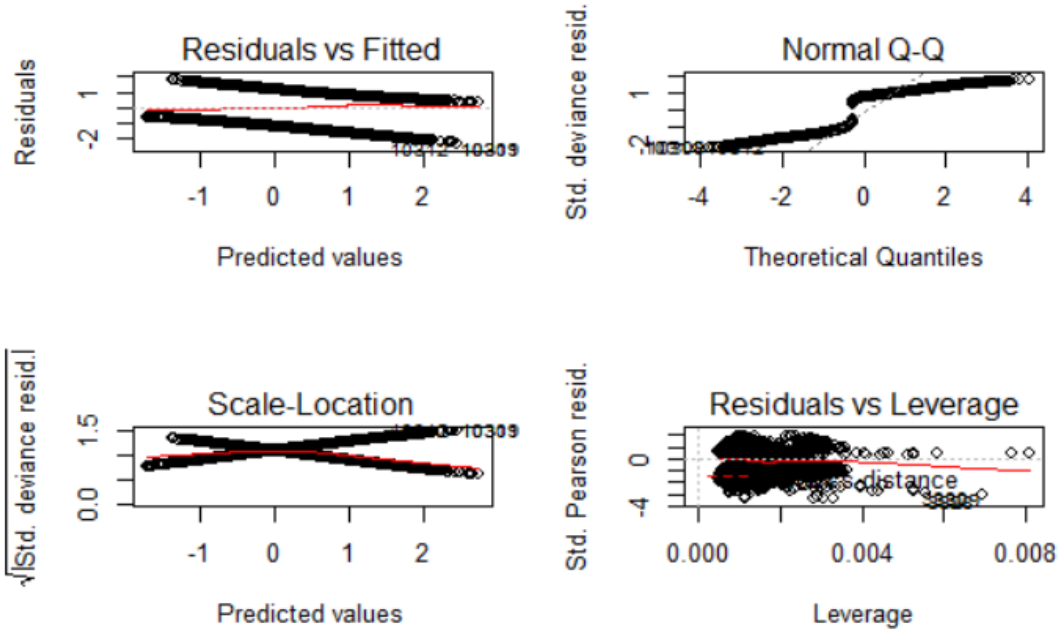
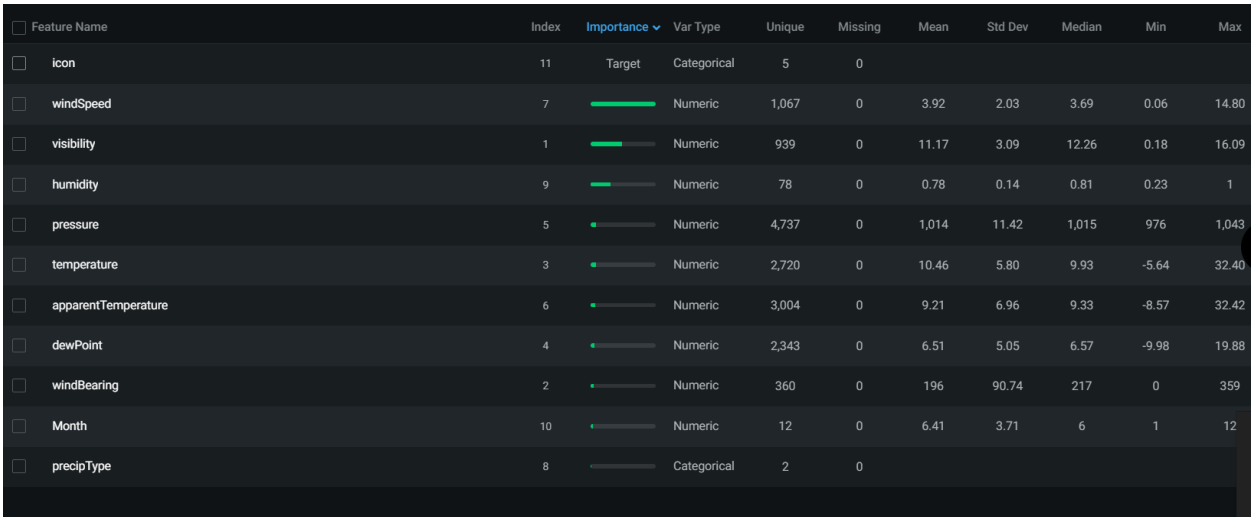


Figure2-2 Residual plots and outliers

We try several methods to detect outliers, including boxplot, cook's distance and DBscan. In the end, we use the result from DBscan to eliminate outliers because it has the best outcome. From the residual plot (Graph 2-2), since the plot looks almost diagonal, we think that most outliers have been removed from the dataset.

Our dataset also suffers from imbalanced classification to some extent. 12253 observations are classed as "partly-cloudy" while only 581 observations are classed as "fog". At first, we tried to fix the problem by deleting some rows, so every class has the same number of observations. However, we found out that the predicted accuracy did not change much when we use this technique. So, we decided not to make changes on the dataset.

We uploaded our data to Data Robot to see whether our data suffers from data leakage. From the plot below, we confirmed that our dataset does not have a data leakage problem when icon is our target variable.



Feature Name	Index	Importance	Var Type	Unique	Missing	Mean	Std Dev	Median	Min	Max
icon	11	Target	Categorical	5	0					
windSpeed	7		Numeric	1,067	0	3.92	2.03	3.69	0.06	14.80
visibility	1		Numeric	939	0	11.17	3.09	12.26	0.18	16.09
humidity	9		Numeric	78	0	0.78	0.14	0.81	0.23	1
pressure	5		Numeric	4,737	0	1,014	11.42	1,015	976	1,043
temperature	3		Numeric	2,720	0	10.46	5.80	9.93	-5.64	32.40
apparentTemperature	6		Numeric	3,004	0	9.21	6.96	9.33	-8.57	32.42
dewPoint	4		Numeric	2,343	0	6.51	5.05	6.57	-9.98	19.88
windBearing	2		Numeric	360	0	196	90.74	217	0	359
Month	10		Numeric	12	0	6.41	3.71	6	1	12
precipType	8		Categorical	2	0					

Figure 2-3 Data Leakage

3 Modeling process

We use both Hand-Crafted Models and DataRobot Models to conduct analysis on the data and we make comparison and contrast to see if there is much difference.

3.1 Data Preparation

After data wrangling, we first removed all outliers detected by DBSCAN. Then we modified the dataset to fit different models by converting categorical variables into dummy variables and normalizing the numerical variables if models needed. Finally, we split our dataset into the training set and validation set by 8:2. The purpose for training set is train the model with labeled observations. Test data set is to evaluate the model and compare the accuracy between different models.

3.2 Hand-Crafted Models

As for Hand-Crafted Models, we created five classification models: **Logistic Model, KNN**, **Decision Tree**, **Support Vector Machine, Random Forest, XGBoost,** and **Naïve Bayes**. Among all handcrafted models, RBF SVM and Random Forest are the overall two best models simulating our data.

3.2.1 Logistic Model

Logistic model is used to model the probability of a certain class. When the probability of a certain class is the largest among other classes, the model will predict the observation in that class. We run multi-class logistic regression on the target variables icon with all other variables as predictors.

Table 3-1 Coefficient of Logistic Model

	cloudy	**fog**	**partly-cloudy**	**wind**
(Intercept)	63.34	-24.90	7.60	-36.05
visibility	-0.32	-2.90	-0.06	0.04
windBearing	0.00	0.00	0.00	0.00
temperature	-2.97	2.26	-0.15	0.72
dewPoint	2.88	-2.57	0.06	-1.05
pressure	-0.01	0.00	0.00	-0.04
apparentTemperature	0.23	0.26	0.14	0.25
windSpeed	0.56	0.29	0.39	5.81
precipTypesnow	0.30	-0.46	0.20	3.13
humidity	-54.96	38.97	-3.73	31.46
Month2	0.19	0.25	0.21	-1.02

Month3	-1.12	-0.35	-0.75	1.11
Month4	-1.57	0.78	0.04	4.53
Month5	-0.20	-0.95	-0.89	7.18
Month6	-0.86	0.22	-1.07	6.95
Month7	-1.39	0.99	-0.89	4.34
Month8	-1.42	1.19	-1.00	4.18
Month9	-0.52	0.28	-0.99	4.65
Month10	-0.77	-0.05	-0.49	3.68
Month11	-0.15	0.39	0.00	1.73
Month12	-0.09	0.28	-0.14	-0.51

Table 3-2 Odds ratios of Logistic Model

	cloudy	fog	partly-cloudy	wind
(Intercept)	3.22e+27	0.00	1.00e+03	0.00
visibility	0.73	0.06	0.94	1.04
windBearing	1.00	1.00	1.00	1.00
temperature	0.05	9.58	0.86	2.05

dewPoint	17.81	0.08	1.06	0.35
pressure	0.99	1.00	1.00	0.96
apparentTemperature	1.26	1.30	1.15	1.28
windSpeed	1.75	1.34	1.48	333.62
precipTypesnow	1.35	0.63	1.22	22.87
humidity	0.00	8.40e+16	0.02	4.60e+13
Month2	1.21	1.28	1.23	0.36
Month3	0.33	0.70	0.47	3.03
Month4	0.21	2.18	1.04	92.76
Month5	0.82	0.39	0.41	1.31e+03
Month6	0.42	1.25	0.34	1.04e+03
Month7	0.25	2.69	0.41	76.71
Month8	0.24	3.29	0.37	65.37
Month9	0.59	1.32	0.37	104.58
Month10	0.46	0.95	0.61	39.65
Month11	0.86	1.48	1.00	5.64
Month12	0.91	1.32	0.87	0.60

From Table 3-1 and Table 3-2, we can know more about how each variable performs in the logistic model. Following are our findings for these two tables:

- Variables that have odds ratios with values higher than 1 suggest a unit increase in these variables would lead to an increase in probability of certain classes. For example, apparentTemperature's odds ratio for "cloudy" is 1.26, so with a unit increase in apparentTemperature, the probability for an observation to be classed as cloudy increases 26%.
- Some variables, such as dewPoint, having a higher odds ratio in one class rather than other classes suggest that these variables have strong explanations on classify certain classes.
- Some variables, such as windBearing and pressure, have the same odds ratio in each class. It means, this variable is less important in doing classification on our target variables. For example, the wind bearing, the direction of wind is not persuasive on classification.

Table 3-3 Confusion matrix for logistic model

```
Confusion Matrix and Statistics

              Reference
Prediction      clear cloudy  fog partly-cloudy wind
  clear           276     13    0           185    0
  cloudy            0      0    0             4    0
  fog               1      0  115             6    0
  partly-cloudy   800    171    2          2235   27
  wind              0      6    0            31  185

Overall Statistics

               Accuracy : 0.6929
                 95% CI : (0.6784, 0.7071)
    No Information Rate : 0.6066
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3622

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: clear Class: cloudy Class: fog Class: partly-cloudy Class: wind
Sensitivity               0.25627      0.000000    0.98291               0.9082     0.87264
Specificity               0.93356      0.998966    0.99822               0.3734     0.99038
Pos Pred Value            0.58228      0.000000    0.94262               0.6909     0.83333
Neg Pred Value            0.77644      0.953121    0.99949               0.7251     0.99296
Prevalence                0.26547      0.046833    0.02884               0.6066     0.05226
Detection Rate            0.06803      0.000000    0.02835               0.5509     0.04560
Detection Prevalence      0.11684      0.000986    0.03007               0.7974     0.05472
Balanced Accuracy         0.59491      0.499483    0.99056               0.6408     0.93151
```

The accuracy for the logistic regression model is 0.692, which sets the baseline for our models. From the confusion matrix and statistics for classification, such as sensitivity and specificity, we can see that the logistic model did a really good job on classifying "fog", "partly-cloudy" and "wind"; however, it is not good at distinguishing "clear" and "cloudy".

3.2.2 KNN

K-nearest neighbor (KNN) is a type of instance-based learning, or lazy learning, where the function is only approximated locally, and all computation is deferred until function evaluation. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors.

In the process of finding the best k-value in the KNN model, we make cross validation to calculate the accuracy of different k values. From the plot below, as k increases, accuracy of the model decreases. So, we finally choose k=5 as our best k value.

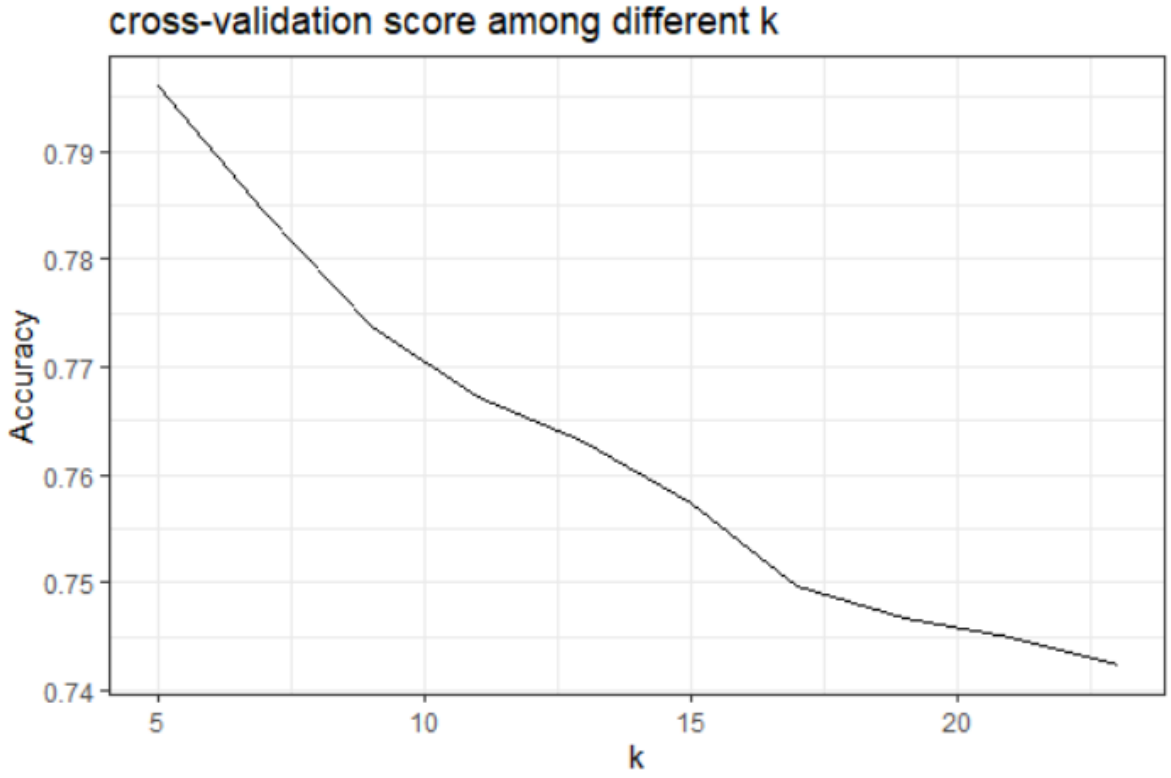


Figure 3-1 Find the best k

Table 3-4 Confusion matrix for KNN

Confusion Matrix and Statistics

Prediction	Reference clear	cloudy	fog	partly-cloudy	wind
clear	788	28	0	277	4
cloudy	18	79	0	21	5
fog	5	0	105	3	0
partly-cloudy	262	78	12	2139	31
wind	4	5	0	21	172

Overall Statistics

Accuracy	0.8092
95% CI	(0.7968, 0.8212)
No Information Rate	0.6066
P-Value [Acc > NIR]	< 2.2e-16
Kappa	0.6508
Mcnemar's Test P-Value	NA

Statistics by Class:

	Class: clear	Class: cloudy	Class: fog	Class: partly-cloudy	Class: wind
Sensitivity	0.7317	0.41579	0.89744	0.8692	0.81132
Specificity	0.8963	0.98862	0.99797	0.7600	0.99220
Pos Pred Value	0.7183	0.64228	0.92920	0.8481	0.85149
Neg Pred Value	0.9024	0.97178	0.99696	0.7902	0.98962
Prevalence	0.2655	0.04683	0.02884	0.6066	0.05226
Detection Rate	0.1942	0.01947	0.02588	0.5272	0.04240
Detection Prevalence	0.2704	0.03032	0.02785	0.6216	0.04979
Balanced Accuracy	0.8140	0.70221	0.94770	0.8146	0.90176

From the confusion Matrix, we could easily find the accuracy of the KNN classification model on predicting our valid data is 0.8092. From the confusion matrix and statistics for classification, such as sensitivity and specificity, we can see that the KNN did a satisfying job on classifying ”clear”, ”fog”, “partly-cloudy” and “wind”; however, it is not good at distinguishing “cloudy”.

3.2.3 Decision tree

Decision tree learning is a method commonly used in data mining. As for the decision tree, we first divide the whole dataset into training and validation dataset. Then we create the decision tree based on train data and plot it.

After we made the plot, we thought the tree may be too complex, which will make it hard to interpret and may overfit the data. So, we wondered if we should prune the tree back. So, we conducted cross validation to determine the size of the tree (complexity). We used plotcp() to provide a graphical representation to the cross validated error summary. The cp values are plotted against the geometric mean to depict the deviation until the minimum value is reached. From the graph we see that the value of cp is declining, and the size of the tree is good to be set to 9.

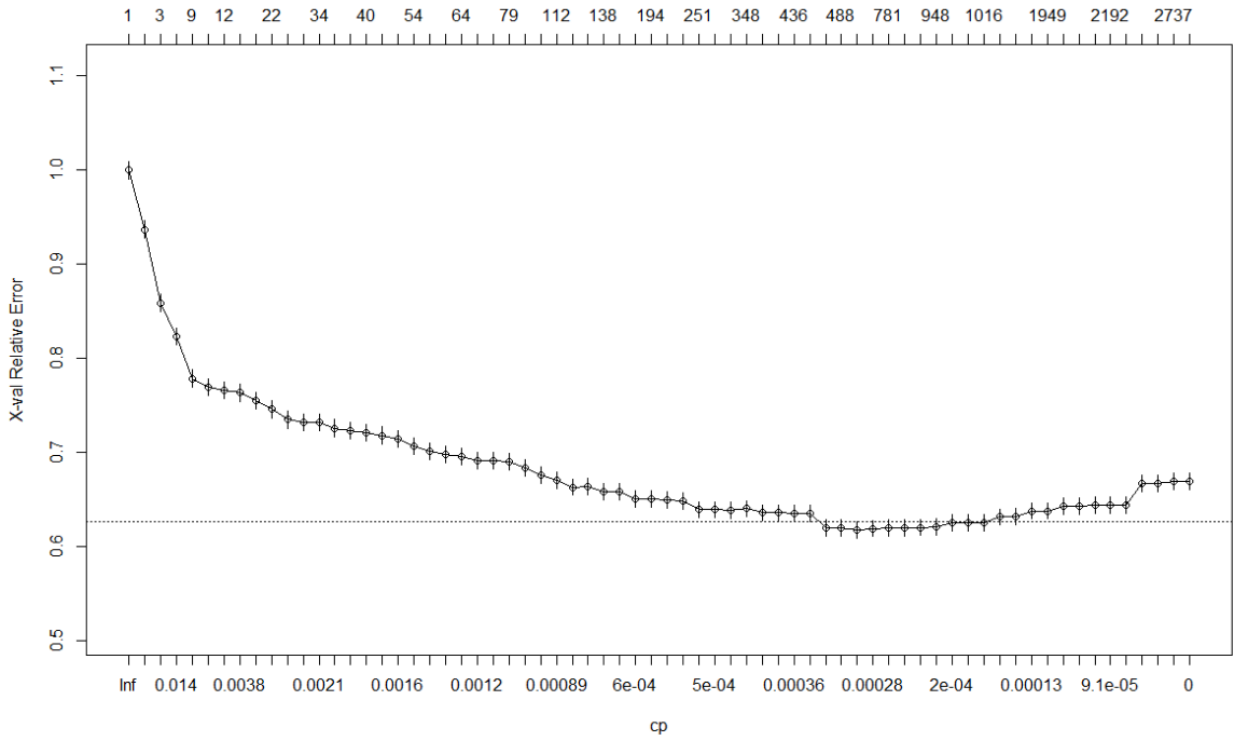


Figure 3-2 cross validated error summary

So, we pruned the tree back.

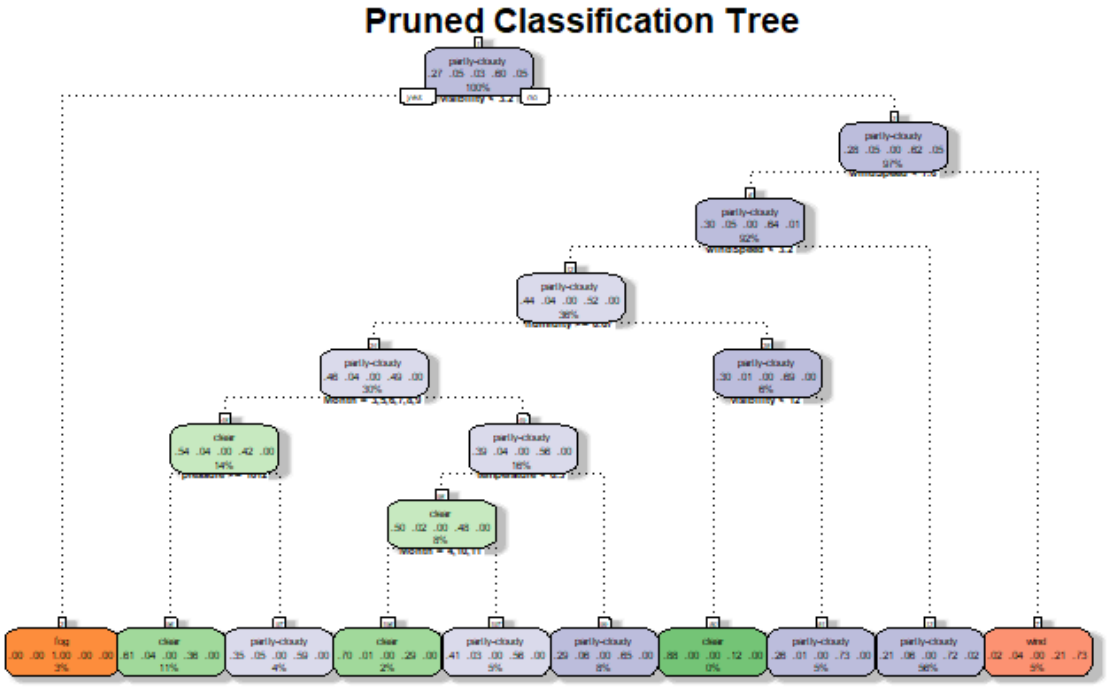


Figure 3-3 Decision tree structure

We try to interpret the tree we build now.

When visibility>=3.21 and 3.195<=windSpeed< 7.625, icon = partly-cloudy; When visibility>=3.21, windSpeed< 3.195, humidity>=0.665, Month=1,2,4,10,11,12 and temperature>=6.315, icon=partly-cloudy; When visibility>=3.21, windSpeed< 3.195, humidity>=0.665, Month=3,5,6,7,8,9 and pressure>=1012, icon=clear; When visibility>=3.21, windSpeed< 3.195, humidity< 0.665 and visibility>=11.52, icon = partly-cloudy; When visibility>=3.21, windSpeed< 3.195, humidity>=0.665, temperature< 6.315 and Month=1,2,12, icon = partly-cloudy; When visibility>=3.21, windSpeed< 3.195, humidity>=0.665, Month=3,5,6,7,8,9 and pressure< 1012, icon = partly-cloudy; When visibility>=3.21 and windSpeed>=7.625, icon = windy; When visibility>=3.21, windSpeed< 3.195, humidity>=0.665, temperature< 6.315 and Month=4,10,11, icon = clear; When 3.21<=visibility< 11.52, windSpeed< 3.195 and humidity< 0.665, icon = clear; When visibility< 3.21, icon = fog.

Table 3-5 Confusion matrix for Decision Tree

```
Confusion Matrix and Statistics

               Reference
Prediction      clear cloudy  fog partly-cloudy wind
  clear           328     18    0           180    0
  cloudy            0      0    0             0    0
  fog               0      0  117             0    0
  partly-cloudy   745    163    0          2230   35
  wind              4      9    0            51  177

Overall Statistics

               Accuracy : 0.703
                 95% CI : (0.6887, 0.717)
    No Information Rate : 0.6066
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3904

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: clear Class: cloudy Class: fog Class: partly-cloudy Class: wind
Sensitivity               0.30455       0.00000    1.00000                0.9061     0.83491
Specificity               0.93356       1.00000    1.00000                0.4091     0.98336
Pos Pred Value            0.62357           NaN    1.00000                0.7028     0.73444
Neg Pred Value            0.78788       0.95317    1.00000                0.7387     0.99083
Prevalence                0.26547       0.04683    0.02884                0.6066     0.05226
Detection Rate            0.08085       0.00000    0.02884                0.5497     0.04363
Detection Prevalence      0.12965       0.00000    0.02884                0.7821     0.05940
Balanced Accuracy         0.61905       0.50000    1.00000                0.6576     0.90913
Confusion Matrix and Statistics
```

From the confusion Matrix, we could easily find the accuracy of the decision tree model on predicting our valid data is 0.703. From the confusion matrix and statistics for classification, such as sensitivity and specificity, we can see that the classification tree did a good job on classifying ”fog”, “partly-cloudy” and “wind”; however, it is bad at distinguishing ”clear” and “cloudy”.

3.2.4 Naive Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. We use the package in R to conduct Naive Bayes methods.

Table 3-6 Confusion matrix for Naive Bayes

```
Confusion Matrix and Statistics

              Reference
Prediction      clear cloudy  fog partly-cloudy wind
  clear            12      1    3             6    0
  cloudy            0      0    0             0    0
  fog               0      0  104             0    0
  partly-cloudy  1061    188   10          2449   99
  wind              4      1    0             6  113

Overall Statistics

               Accuracy : 0.6601
                 95% CI : (0.6453, 0.6747)
    No Information Rate : 0.6066
    P-Value [Acc > NIR] : 1.105e-12

                  Kappa : 0.204

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: clear Class: cloudy Class: fog Class: partly-cloudy Class: wind
Sensitivity              0.011142       0.00000    0.88889              0.9951     0.53302
Specificity              0.996644       1.00000    1.00000              0.1491     0.99714
Pos Pred Value           0.545455           NaN    1.00000              0.6433     0.91129
Neg Pred Value           0.736059       0.95317    0.99671              0.9520     0.97483
Prevalence               0.265467       0.04683    0.02884              0.6066     0.05226
Detection Rate           0.002958       0.00000    0.02563              0.6036     0.02785
Detection Prevalence     0.005423       0.00000    0.02563              0.9384     0.03056
Balanced Accuracy        0.503893       0.50000    0.94444              0.5721     0.76508
```

From the confusion Matrix, we could easily find the accuracy of the Naive Bayes model on predicting our valid data is 0.6601. From the confusion matrix and statistics for classification, such as sensitivity and specificity, we can see that the Naive Bayes did a good job on classifying ”fog”, “partly-cloudy” and “wind”; however, it is bad at distinguishing ”clear” and “cloudy”.

3.2.5 Support Vector Machine

Support Vector Machine (SVM) is one of the most important supervised learning models. Two important parameters in RBF SVM are C and gamma. The C parameter trades off correct classification of training examples against maximization of the decision function’s margin. The gamma parameter is the inverse of the radius of influence of samples selected by the model as support vectors.

3.2.5.1 Linear SVM

Linear SVM separates groups by linear lines. We tried to tune Linear SVM with several C and gamma values, but the accuracy just changed a little bit. So, we set C equals to 1 and gamma equals to 0.1.

Table 3-6 Confusion matrix for Linear SVM

```
Confusion Matrix and Statistics

              Reference
Prediction      clear cloudy  fog partly-cloudy wind
  clear             0      0    0             0    0
  cloudy            0      0    0             0    0
  fog               1      0  116             2    0
  partly-cloudy  1075    185    1          2433   28
  wind              1      5    0            26  184

Overall Statistics

               Accuracy : 0.6737
                 95% CI : (0.659, 0.6881)
    No Information Rate : 0.6066
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2581

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: clear Class: cloudy Class: fog Class: partly-cloudy Class: wind
Sensitivity               0.0000       0.00000    0.99145                0.9886     0.86792
Specificity               1.0000       1.00000    0.99924                0.1924     0.99168
Pos Pred Value               NaN           NaN    0.97479                0.6537     0.85185
Neg Pred Value            0.7345       0.95317    0.99975                0.9164     0.99271
Prevalence                0.2655       0.04683    0.02884                0.6066     0.05226
Detection Rate            0.0000       0.00000    0.02859                0.5997     0.04535
Detection Prevalence      0.0000       0.00000    0.02933                0.9174     0.05324
Balanced Accuracy         0.5000       0.50000    0.99535                0.5905     0.92980
```

From the confusion Matrix, we could easily find the accuracy of the linear SVM model on predicting our valid data is 0.6737. From the confusion matrix and statistics for classification, such as sensitivity and specificity, we can see that the linear SVM did a good job on classifying ”fog”, “partly-cloudy” and “wind”; however, it is bad at distinguishing ”clear” and “cloudy”.

3.2.5.2 RBF SVM

Gaussian RBF (Radial Basis Function) is a popular Kernel method used in SVM models. RBF kernel is a function whose value depends on the distance from the origin or from some point. We tuned parameter C and gamma with C ranged from 0.1 to 100 and gamma ranged from 0.01 to 10. The best parameters for our data are C equals to 10 and gamma equals to 1.

Table 3-7 Confusion matrix for RBF SVM

```
Confusion Matrix and Statistics

               Reference
Prediction      clear cloudy  fog partly-cloudy wind
  clear           840     15    2           240    7
  cloudy           17    109    0            33    7
  fog               1      0  109             5    0
  partly-cloudy   216     66    6          2160   19
  wind              3      0    0            23  179

Overall Statistics

               Accuracy : 0.8373
                 95% CI : (0.8256, 0.8486)
    No Information Rate : 0.6066
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7061

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: clear Class: cloudy Class: fog Class: partly-cloudy Class: wind
Sensitivity                0.7799       0.57368    0.93162                0.8777     0.84434
Specificity                0.9114       0.98526    0.99848                0.8076     0.99324
Pos Pred Value             0.7609       0.65663    0.94783                0.8756     0.87317
Neg Pred Value             0.9197       0.97918    0.99797                0.8107     0.99143
Prevalence                 0.2655       0.04683    0.02884                0.6066     0.05226
Detection Rate             0.2070       0.02687    0.02687                0.5324     0.04412
Detection Prevalence       0.2721       0.04092    0.02835                0.6081     0.05053
Balanced Accuracy          0.8457       0.77947    0.96505                0.8427     0.91879
```

From the confusion Matrix, we could easily find the accuracy of the RBF SVM model on predicting our valid data is 0.8373. From the confusion matrix and statistics for classification, such as sensitivity and specificity, we can see that the classification tree did a good job on classifying all the categories. Even though “cloudy” is not as good as other variables, it is much better than other models do.

3.2.6 Random Forest

Random Forests are an ensemble of individual Decision Trees. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

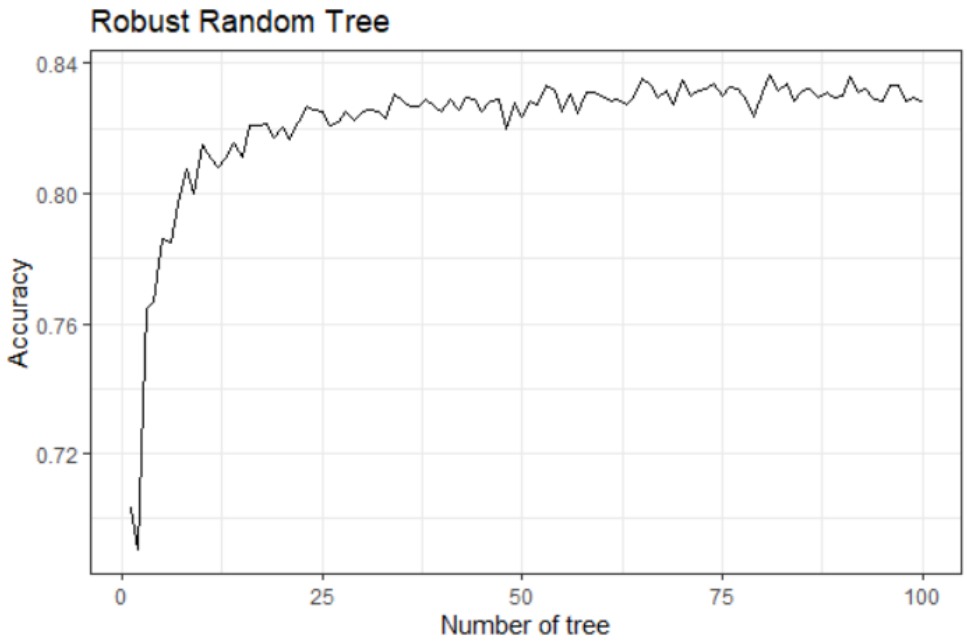


Figure 3-4 Accuracy of RF with different number of trees

From Figure 3-4, we tried different numbers of trees in the random forest model. When the number of trees is larger than 50, the random tree model becomes robust with accuracy higher than 0.80.

Table 3-8 Confusion matrix for Random Forest

```
Confusion Matrix and Statistics

                Reference
Prediction       clear cloudy  fog partly-cloudy wind
  clear            727     22    0           154    2
  cloudy             8     61    0            10    0
  fog                0      0  117             0    0
  partly-cloudy    339    101    0          2282   16
  wind               3      6    0            15  194

Overall Statistics

               Accuracy : 0.8334
                 95% CI : (0.8215, 0.8447)
    No Information Rate : 0.6066
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6837

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: clear Class: cloudy Class: fog Class: partly-cloudy Class: wind
Sensitivity                0.6750       0.32105    1.00000                0.9273     0.91509
Specificity                0.9403       0.99535    1.00000                0.7143     0.99376
Pos Pred Value             0.8033       0.77215    1.00000                0.8335     0.88991
Neg Pred Value             0.8890       0.96757    1.00000                0.8643     0.99531
Prevalence                 0.2655       0.04683    0.02884                0.6066     0.05226
Detection Rate             0.1792       0.01504    0.02884                0.5625     0.04782
Detection Prevalence       0.2231       0.01947    0.02884                0.6749     0.05373
Balanced Accuracy          0.8076       0.65820    1.00000                0.8208     0.95443
```

From the confusion Matrix, we could easily find the accuracy of the Random Forest model on predicting our valid data is 0.8334. From the confusion matrix and statistics for classification, such as sensitivity and specificity, we can see that the classification tree did an overall good job on classifying all the categories, even though “clear” and “cloudy” is not as good as other variables.

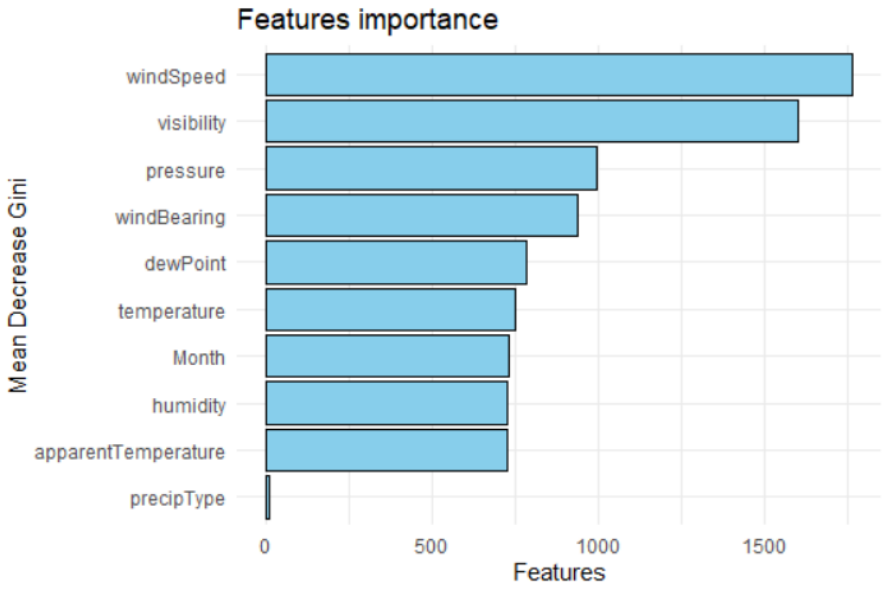


Figure 3-1 Features importance by Random Forest

The result of random forest also suggests the importance of features we used in the model. In the random forest model, each tree is developed on features randomly sampled as candidates at each split and Gini impurity is a metric used in Decision Trees to determine how, so the features that have larger mean decrease Gini, the features is more important.

From Figure 3-1, we noticed that windSpeed is the most important feature, with mean decrease Gini of approximately 1600, followed by visibility as the second important features. It makes sense since both windSpeed and visibility indicates some characteristics of the weather conditions. For example, clear suggests high visibility, while fog indicates low visibility. PrecipType has the lowest mean decrease in Gini while other features have similar moderate importance.

3.2.8 XGBoost

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. XGBoost always has good performance because

Table 3-10 Confusion matrix for XGBoost

Confusion Matrix and Statistics

Prediction	Reference clear	cloudy	fog	partly-cloudy	wind
clear	785	23	0	209	0
cloudy	9	56	0	10	2
fog	0	0	160	0	0
partly-cloudy	384	95	0	2298	5
wind	0	5	0	11	178

Overall Statistics

Accuracy	0.822
95% CI	(0.8101, 0.8334)
No Information Rate	0.5976
P-Value [Acc > NIR]	< 2.2e-16
Kappa	0.6677
Mcnemar's Test P-Value	NA

Statistics by Class:

	Class: clear	Class: cloudy	Class: fog	Class: partly-cloudy	Class: wind
Sensitivity	0.6664	0.31285	1.00000	0.9090	0.96216
Specificity	0.9240	0.99482	1.00000	0.7156	0.99604
Pos Pred Value	0.7719	0.72727	1.00000	0.8260	0.91753
Neg Pred Value	0.8777	0.97038	1.00000	0.8412	0.99827
Prevalence	0.2785	0.04232	0.03783	0.5976	0.04374
Detection Rate	0.1856	0.01324	0.03783	0.5433	0.04208
Detection Prevalence	0.2404	0.01820	0.03783	0.6577	0.04586
Balanced Accuracy	0.7952	0.65383	1.00000	0.8123	0.97910

From the confusion Matrix, we could easily find the accuracy of the XGBoost model on predicting our valid data is 0.822. From the confusion matrix and statistics for classification, such as sensitivity and specitivity, we can see that the XGBoost did an overall good job on classifying all the categories, even though “clear” and “cloudy” is not as good as other variables.

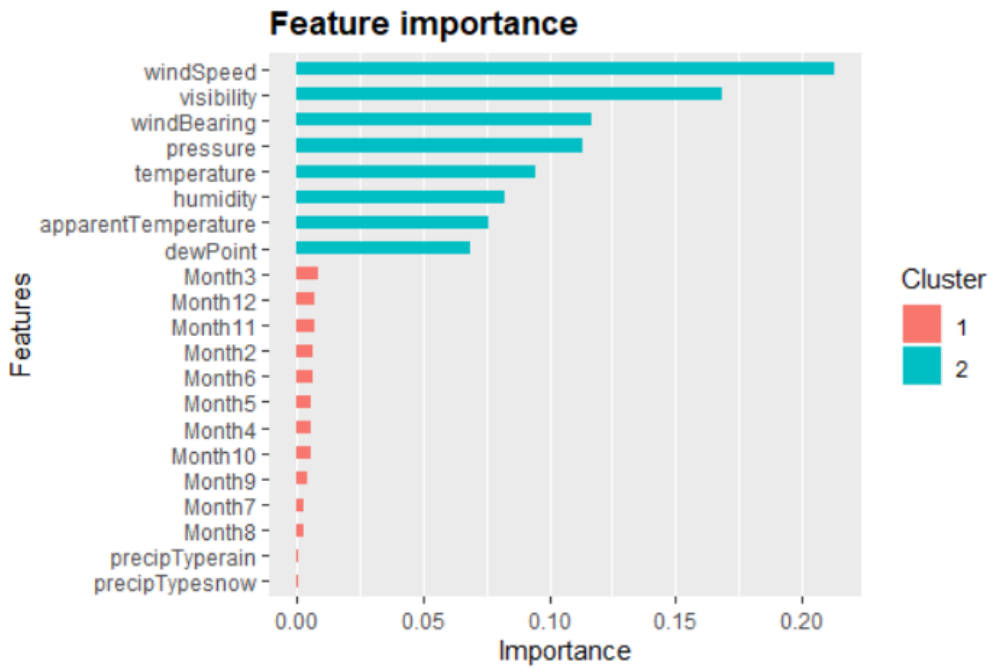


Figure 3-2 Feature importance by XGBoost

The result of XGBoost also suggests the importance of features we used in the model.

From Figure 3-2, we noticed that windSpeed is the most important feature, followed by visibility as the second important features. It makes sense since both windSpeed and visibility indicates some characteristics of the weather conditions. For example, clear suggests high visibility, while fog indicates low visibility. PrecipType has the lowest importance while other features have similar moderate importance.

3.2.9 Comparison between Handcraft Models

We compared different handcraft models in terms of test accuracy and training accuracy.

Among all handcrafted models, RBF SVM and Random Forest overall perform the best. RBF SVM has the highest score in test data accuracy among all the models (0.837), but is relatively slow considering execution time. Random Forest on the other hand, is much faster than RBF SVM. It has the highest training data accuracy (1.000) and sacrifices only a tiny amount of test data accuracy (0.833), which is the second best among all models.

KNN, Decision Tree and XGBoost on the other hand, especially XGBoost, also have good overall performance. They are close with Random Forest in execution time, which is fast, and they sacrifice only a small amount of test and training accuracy.

Compared with other models, Logistic Regression, Naive Bayes and Linear SVM are the fastest in execution time. But their flaws are also obvious that they sacrifice a lot of accuracy when modeling with our data.

Table 3-4 Comparison between Handcraft Models

Models	Test Accuracy	Training Accuracy	Time
Logistic Regression	0.693	0.688	Super-fast
KNN	0.800	0.878	Fast
Decision Tree	0.703	0.699	Fast
Naive Bayes	0.661	0.650	Super-fast
Linear SVM	0.674	0.669	Super-fast
RBF SVM	**0.837**	**0.967**	**Slow**
Random Forest	**0.833**	**1.000**	**Fast**
XGBoost	0.822	0.937	Fast

3.2.10 Management of overfitting

From the table 3-4 above, our classifiers are not overfitting since training accuracy in all models are not largely higher than test accuracy. We used different methods to avoid overfitting in different models.

In the logistic regression, in order to avoid overfitting, we control the number of predictors. As more predictors, the model becomes more complex and harder to converge. So, we only include a predictor that has a p-value less than 0.05.

In SVM models, we controlled parameter C so it won't be too large. In addition, our dataset is large, so SVM is not likely to overfit.

In the KNN model, test accuracy decreases as k increases. We try KNN on training data to get a training accuracy. Since training accuracy is just 7% higher than test accuracy, our KNN model is general to predict on new data.

In the Decision Tree, we avoid overfitting and make the model less complex by pruning the tree to a smaller tree size. Therefore, the tree will not be too deep to adopt new data.

In the Random Forest model, we controlled the number of trees and number of features randomly sampled as candidates at each split. Though the result won't be good when just considering a single tree, it ensembles all trees and has a majority vote to have a robust prediction accuracy.

In the XGBoost model, we controlled early stopping to avoid overfitting. Actually early stopping is a common method in machine learning to avoid overfitting by attempting to automatically select the inflection point where performance on the test dataset starts to decrease while performance on the training dataset continues to improve as the model starts to overfit.

3.3 Data Robot Model

3.3.1 Random Forest Classifier (Gini) (63.99% of sample size)

3.3.1.1 Introduction of model

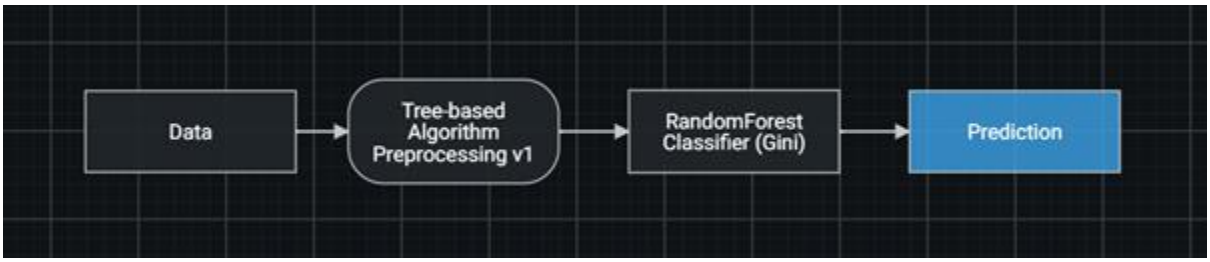


Figure 3-3 Blueprint

This model uses a tree-based algorithm preprocessing method trained on 64% of training data and then RandomForest Classifier (Gini) method to predict the target variable. Random Forests are an ensemble method where hundreds of individual decision trees are fit to bootstrap re-samples of the original dataset. This method can reduce models' variance and produce more stable estimators that generalize well out-of-sample. Thus, Random Forests are hard to over-fit and are very accurate as a prediction model.

3.3.1.2 Interpretation of model

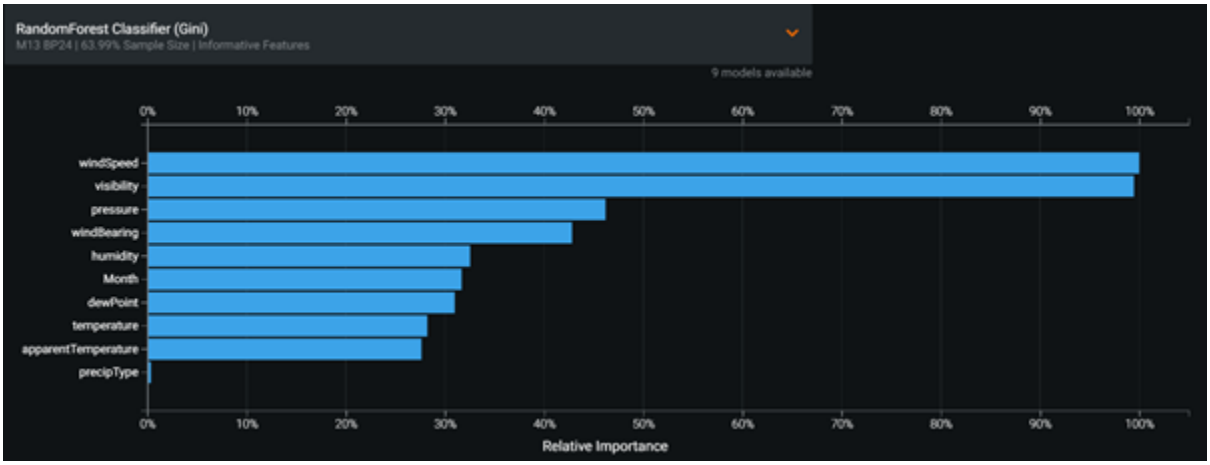


Figure 3-4 Variables relative importance

The results show the 10 variables' abilities in predicting the target variable (icon), which has five results including clear, fog, wind, cloudy and partly-cloudy. WindSpeed and visibility have the highest importance in prediction while pressure and windBearing also have relatively high importance to predict icons. The other 5 variables including humidity, Month, dewpoint, temperature and apparentTemperature have relatively lower importance in predicting the target variable.

3.3.1.3 Accuracy of model

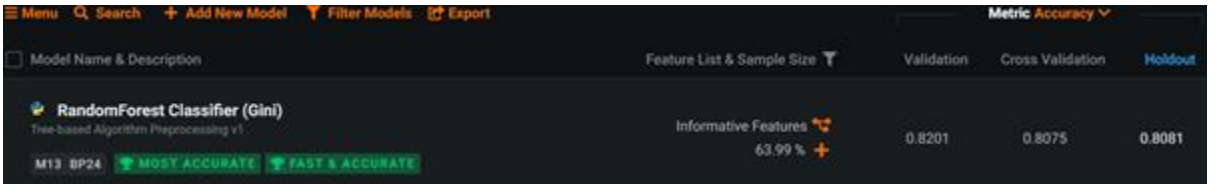


Figure 3-5 Model performance

Figure 3-6 Multiclass Confusion Matrix

In the prediction, this table indicates that our model has a high accuracy of 80.81% in stimulating our data, so Data Robot flagged it as the most accurate and fast model in the leaderboard. Following table shows the F1 score, Recall and Precision for each class.

Table 3-5 Selected Class Precision and Recall

Variable	F1 score	Recall	Precision
clear	0.7	0.63	0.78
cloudy	0.33	0.21	0.79
fog	1	1	1
partly-cloudy	0.86	0.92	0.81
wind	0.86	0.88	0.85

In a lift chart, the x-axis of the chart represents the percentage of the test dataset that is used to compare the predictions. The y-axis of the chart represents the percentage of predicted values. The

more accurate model is the one where the line has the steepest slope, and the widest vertical range, because that means the model is correctly separating high values from low values in its predictions. The model is particularly higher for the first 70% and lower for the last 30% of rows ranked by predictions.

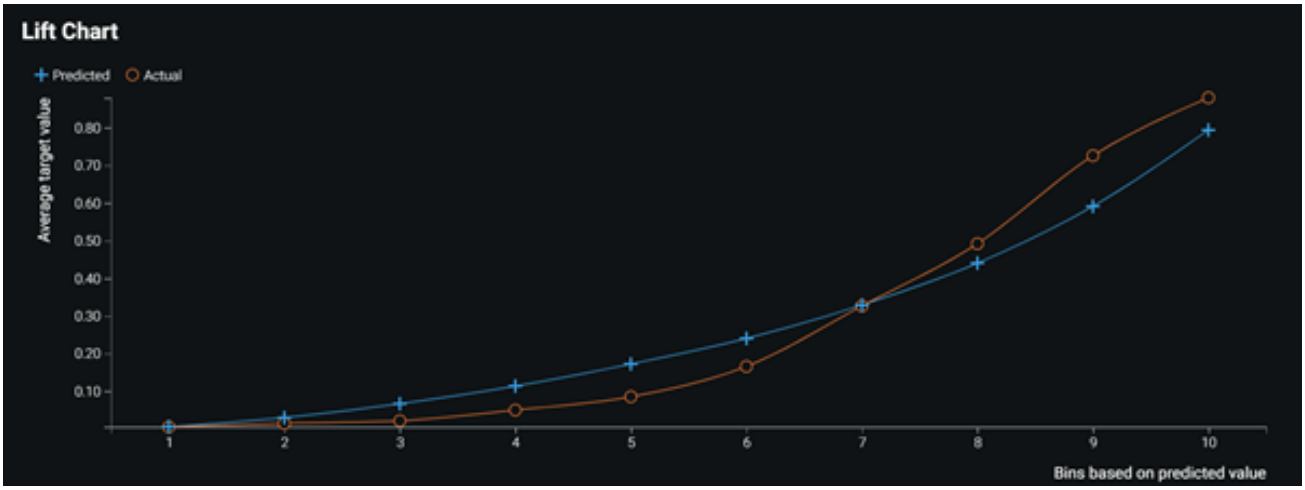


Figure 3-7 Lift Chart

3.3.2 Light Gradient Boosted Trees Classifier with Early Stopping (SoftMax Loss) (16 leaves) (64% of sample size)

3.3.2.1 Introduction

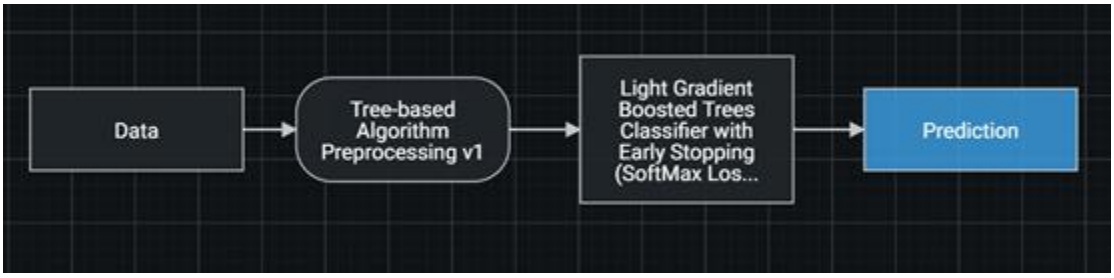


Figure 3-8 Blueprint

This model first imports data into Tree-based Algorithm Preprocessing v1 to preprocess for Tree-based Models. Then use the Gradient Boosted Trees method to develop the best models and the process is similar to random forest trees. Early stopping is used to determine the best number of trees where overfitting begins. Then we get the final prediction of the data.

3.3.2.2 Interpretation of model

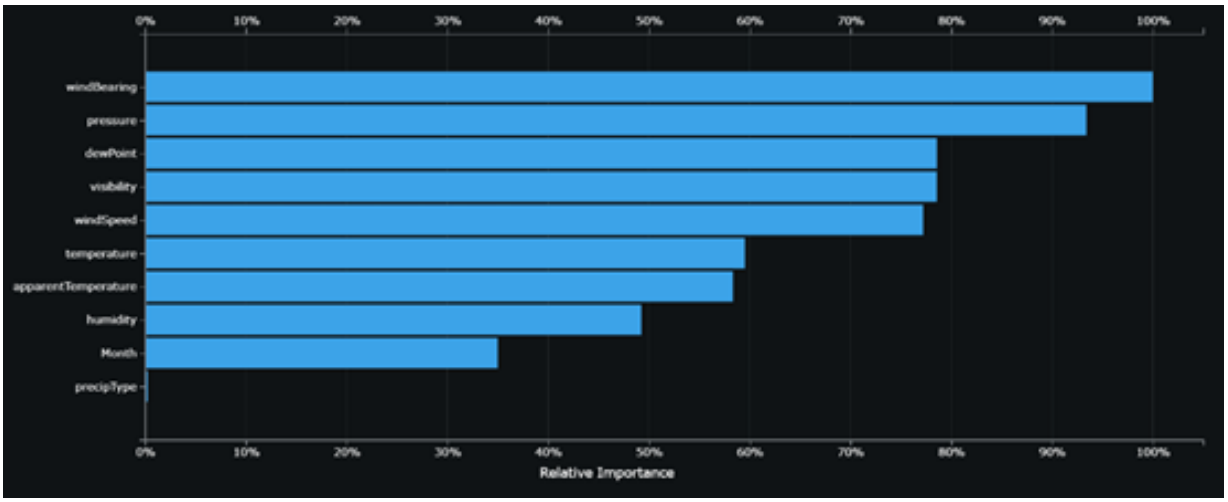


Figure 3-9 Variables relative importance

From insight of this model, we find that wind bearing is the most important factor and pressure is the second important factor. All 10 variables have influence on our targeted variable – icon. Icon represents different weather types: partly-cloudy, fog, cloudy, clear, and wind. This model explains relationships among these variables and our target value. We know different weather performance relates to wind, pressure, dew point, visibility, temperature, and humidity. Month has a great influence in affecting weather performance.

3.3.2.3 Accuracy of model

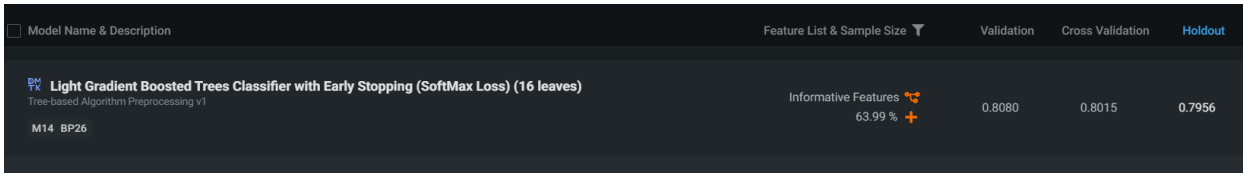


Figure 3-10 Model performance

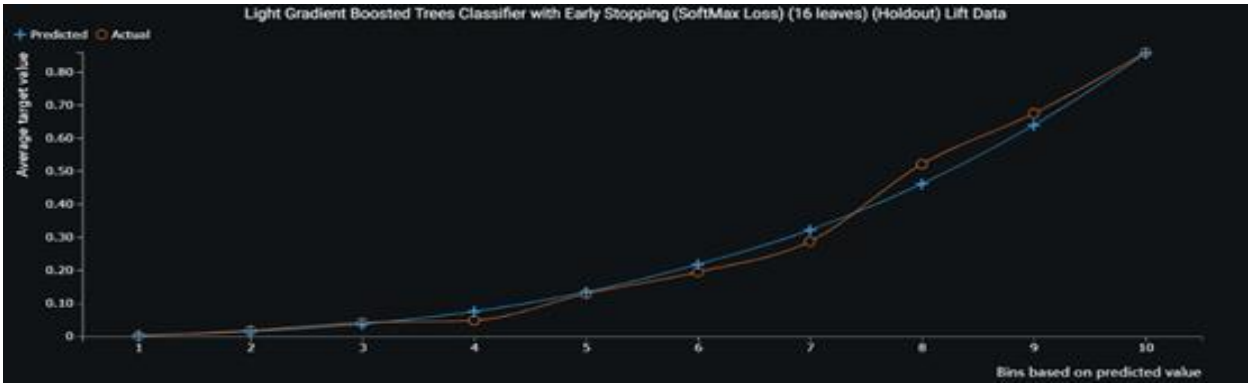


Figure 3-11 Lift Chart

Whenever the models have divergent predictions, the model represented by the blue line is typically closer to the actual results. In fact, when the model becomes more divergent, the more accurate is the model represented by the blue line. So, our Data Robot model is quite accurate.

Figure 3-12 Multiclass Confusion Matrix

Metric indicates our model has 79.56% accuracy in prediction. In the prediction, this table indicates that our model has a high accuracy in stimulating our data and does a good prediction job.

Table 3-6 Selected Class Confusion Matrix

Variable	Recall	% of wrong time
clear	0.63	37.4
cloudy	0.24	75.6
fog	1	0
partly-cloudy	0.89	10.5
wind	0.91	9.1

4 Comparison between Handcraft Models and Data Robot Models

We decided to compare our RBF SVM Classifier and Random Forest model with Data Robot ones.

4.1 Comparison on models' algorithm

Our Random Forest model is highly similar with Data Robot's random forest model. They both constructed multiple numbers of trees and trees are split based on the decrease of Gini impurity. Difference between these two models is mainly the tuned parameters.

DataRobot Light Gradient Boosted Tree Classifier is also a tree-based model; however, instead of constructing trees by randomly chosen variables, Light Gradient Boosted Tree Classifier constructs trees that learn from the previous tree. By carefully setting the early stopping point, Light Gradient Boosted Tree Classifier has faster training speed and higher efficiency.

4.2 Comparison on models' result

To have a closer look at our handcraft result and compare between models, we also estimated handcraft models' validation score according to the same standard as Data Robot's. Validation is the first cross validation result. Cross Validation is the average of all cross validation runs. Holdout is the test accuracy on test data.

Table 4-1 Comparison on models' accuracy

Model	Validation	Cross Validation	Holdout
RBF SVM	0.827	0.824	0.837
Random Forest	0.819	0.828	0.833
DataRobot Random Forest	0.820	0.808	0.808
DataRobot Light Gradient Boosted Tree Classifier	0.808	0.802	0.796

From Table 4-1, handcraft random forest has the highest cross validation score and handcraft RBF SVM has the highest holdout score.

We also noticed that all three scores of Random Forest are higher than DataRobot Light Gradient Boosted Tree Classifier, though their algorithms are basically the same. The random tree model's prediction accuracy becomes robust when the number of trees constructed is larger than 50. Both Random Tree models' number of trees is larger than 50, so we concluded that the difference between the accuracy is minor, and it is because of random state.

4.3 Comparison on models' execution time

Though we mentioned that some of our handcrafted models are more accurate than Data Robot's models, their models are much more superior in execution time. We think they are faster than ours since the models' algorithm is designed to be fast and efficient. Moreover, the server of DataRobot may be much more powerful, so we cannot beat them in execution time.

5 Conclusion and Reflection

Since weather prediction is quite important to our daily life, we decided to do research on this problem. We used a weather dataset sourced from darksky api to predict weather conditions (clear, cloudy, fog, partly-cloudy and wind) based on other relevant important environmental variables such as wind speed, temperature, dew point etc. To achieve this, we run several handcraft classification models on these variables, including logistic model, KNN, decision tree, Naive Bayes, SVM, Random Forest and XGBoost. Among all handcraft classifiers, RBF SVM and Random Forest gave the highest scores on test accuracy, 0.837 and 0.833 respectively. Noticeably, XGBoost also gave a competitive test accuracy with less execution time than RBF SVM and Random Forest.

In addition, we do the same classification on DataRobot and compare our handcraft models and Data Robot's. DataRobot suggests two models: Random Forest Classifier (Gini) and Light Gradient Trees Classifier with Early Stopping, with accuracy of around 0.8.

We are delighted to conclude that our handcraft RBF SVM and Random Forest model beat Data Robot's suggested model in terms of accuracy. Meanwhile, Data Robot's models are faster than ours since the models' algorithm is designed to be fast and efficient.

During our journey, we also encounter some problems and gain some insights. One of our insights is that we should have more consideration in selecting data and target variables in the first place. At first, we did not choose icon as our variable, but preciptype instead. However, since preciptype only has two values: rain and snow, and our data contains adequate information to do the prediction, so even the simplest model can achieve 99% accuracy on test data. To be detailed, our dataset has more than one variable indicating temperature information; preciptype is highly correlative with temperature, and it is common sense that when temperature falls near zero degree centigrade, rain turns into snow. We think that choosing preciptype as our target variable is not a good choice since the models all have high accuracy so it's hard to compare between them. Also, we think choosing preciptype as target variable may not be as significant as choosing icon in real life. Since icon represents weather type, it gives out more information than preciptype. Also, our purpose is to forecast weather, so icon, as a representative of weather type, is a more appropriate target variable.