IEOR 242 Final Project Report

Team members: Siyi Weng, Jing Yu, Tianyi Xu, Chengjie Xia, Zhibo Tang

1. Motivation:

We want to use climatic factors such as wind direction and speed, pressure, and humidity, etc. on the previous day to predict if a location (in Australia) will rain the second day. For daily use, our model can work as a reference for people to dress tomorrow and help people to prepare for the need of bringing umbrellas, raincoats, etc. For business, we can help businesses to make better decisions, such as the airline company, so they can schedule their plan in advance. For farming, we can help farmers to get prepared for weather hazards, intense fall/rainstorms to avoid the risk of loss.

2.Data Collection and Processing:

2.1.Data Collection

The data we collected is from http://www.bom.gov.au/climate/data which contains daily weather observations from numerous weather stations across Australia in 10 years.

Since there are about 145k records in the original dataset, we randomly sampled 5000 weather records which is a reasonable size for us to do further analysis.

Then we did some sanity checks on the data:

- Check If there are missing values in the label
- Check the labels column are all yes or no

We randomly split the data such that 70% are training data and 30% are testing data.

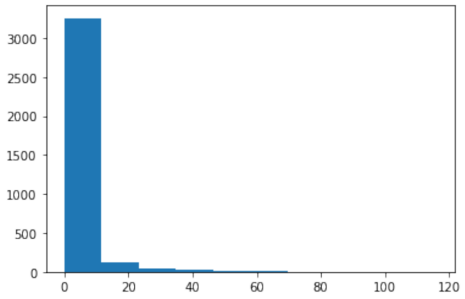
We also encode the label to be 0 and 1, 0 for “not raining tomorrow ” and 1 for “raining tomorrow”.

2.2.Examine Outliers

We then check if the data contains any abnormal data by examining the difference between the minimum and 1 percentile of the data and the difference between the maximum and 99 percentile of the data. Since all the gaps are not extremely broad, we then conclude that there are no outliers in the data.

2.3.Feature Engineering 1: Create Binary Indicator “rainToday”

There’s a column “rainfall” in the data describing the amount of rainfall recorded for the day in mm. After looking at the histogram of this column, we can see the histogram skewed right and about 60% of the data are in the range 0-10mm. So we convert the continuous numerical variable into a binary categorical variable by thresholding at 1mm. In other words, If the rainfall is larger than 1mm then we regard today as a rainy day and vice versa.



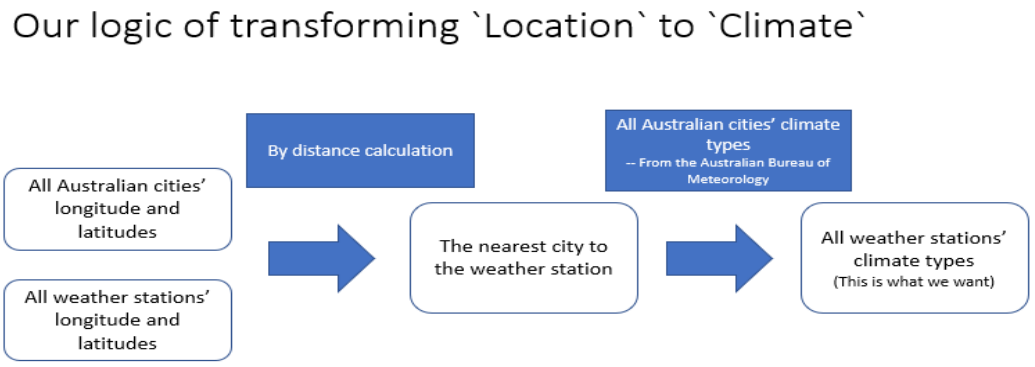
2.4.Feature Engineering 2: Extract Column “Month” for timestamp

As it is common sense that the rainfall is seasonal, we want to take care of the relationship between time and rainfall in our modeling process. However, there are 2141 unique dates in our “Date” column. Since it is far too large to create categorical variables for a date directly, we extract the month from it without losing much information. In that case, the granularity of the data has reduced from daily data to monthly data.

2.5.Feature Engineering 3: Transform feature `Location` to `Climate`:

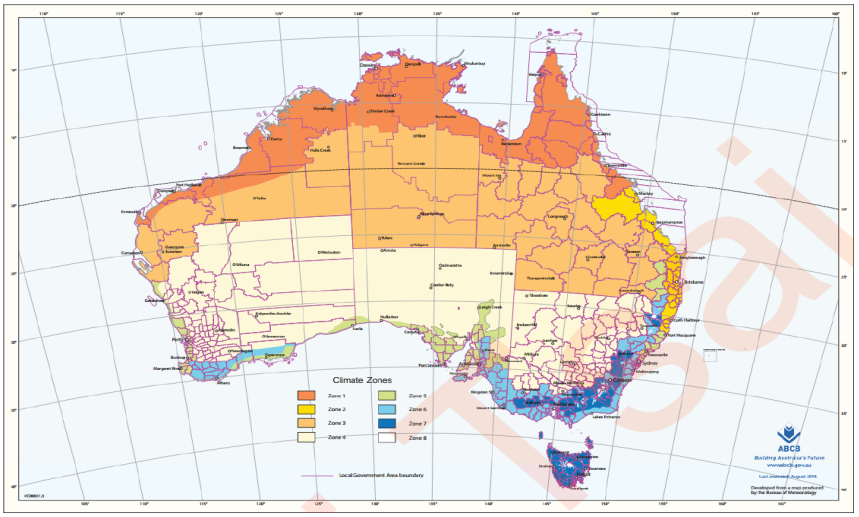
The `Location` variable represents the common name of the location of the weather station and there are 49 different locations in our dataset. We obviously cannot simply use the get dummy function to include 49 various locations categories in our dataset. This method could make the data very sparse and lead to poor model

performance. Thus, we considered deploying cities' location information we can find online, getting the corresponding climate types for the various cities, and finally finding the nearest city with the locations of the weather stations to match up the corresponding climate types. In this way, we not only effectively reduce the categories number, but also provide interpretable information in the further model analysis process.



Our solution is:

First of all, given all the cities in Australia, we used a web crawler to find out the longitude and latitude of these cities from Google. We stored this position information into the CSV file "cityll.csv". Secondly, we found out an official document named "Cityclimate.csv" from the Australian Bureau of Meteorology. (http://www.bom.gov.au/jsp/ncc/climate_averages/climate-classifications/index.jsp) This document records the climate types of different cities in Australia. There is a total of 7 types of climate. (Please check the picture below for full details of climate types in Australia) We combined the "cityll.csv" and "Cityclimate.csv" into one data frame called "citylld" and this data set contained the position information of all Australian cities and their climate types.



Then, we use the web crawler to again access the position information of all the weather stations' locations in our training data set. Next, we used the angle to radian and np.argsort() function to find the closest city near the weather stations and stored the information in CSV file "samplelocation.csv".

Now we successfully realize the transformation from `Location` to `Climate`.

2.6.Handle the missing values:

We displayed the percentage of every features' missing rate and we found out most of the features such as `MaxTemp`, `WindDir9am` etc. are less than 10%. However, the features `Evaporation`, `Sunshine`, `Cloud9am` and `Cloud3pm` have approximately 40% of the missing rate.

To deal with the categorical variables, we deployed SimpleImputer to find the most frequent category and filled in every missing value with the most frequent value. For the numerical variables, we used SimpleImputer to get the means and filled the missing value with the mean values.

2.7.Encoding and normalization:

Last but not the least, we encoded every categorical variable using OrdinalEncoder. Also, we normalized the numerical variables using StandardScaler. This step is necessary because we will use unit-sensitive models such as SVM in the model building process. This action can effectively accelerate the convergence rate for our models.

Now we have obtained our clean, standardized, encoded dataset with no missing values.

3. Analytics models:

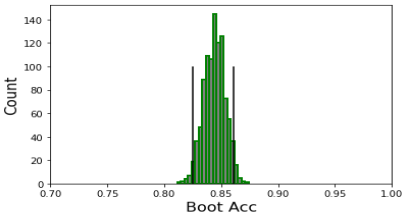
We divided our dataset into 30% test set and 70% training set. Then, we applied nine different machine learning models to make predictions on whether or not it rains tomorrow.

(1) Baseline model:
Predicting tomorrow will not rain with accuracy 0.77133.

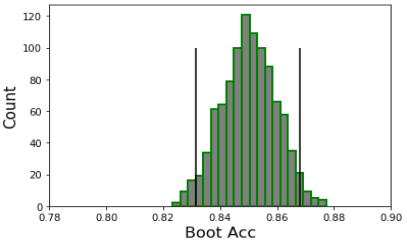
(2) SVM models:
Select kernels [‘Linear’,’Poly’,’RBF’,’Sigmoid’] based on training set data.
‘Linear’: accuracy:0.844 recall:0.469388 Auc:0.869029
‘Poly’: accuracy: 0.840667 recall: 0.306122 Auc:0.868157
‘RBF’: accuracy: 0.8133 recall: 0.306122 Auc:0.814873
‘Sigmoid’: accuracy: 0.655333 recall:0.154519 Auc: 0.437308

Since the sigmoid kernel has too low accuracy, we excluded Sigmoid. Then, we used the training set to build our model and test the performance of models with test set data.

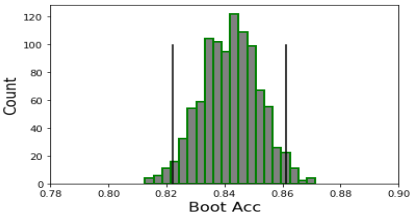
(2.1) SVM-Linear Kernel
Accuracy: 0.844
95% confidence interval [0.82533333 , 0.86133333]
confusion matrix: [1105, 52]
[182,161] TPR=0.4694



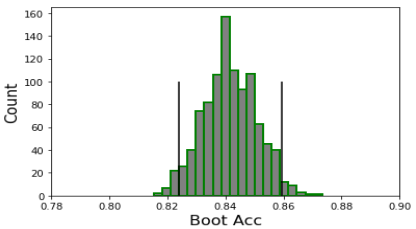
(2.2) SVM-Poly Kernel
Accuracy:0.8506
95% confidence interval [0.83133333 , 0.868]
Confusion matrix: [1113,44]
[180,163] TPR=0.4752



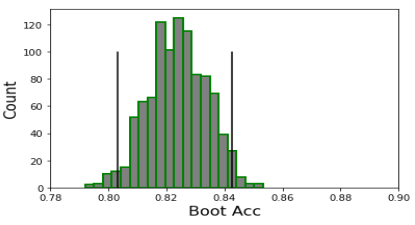
(2.3) SVM-RBF Kernel:
Accuracy: 0.842
95% confidence interval [0.822 , 0.86133]
Confusion Matrix: [1115, 42]
[195,148] TPR=0.4314



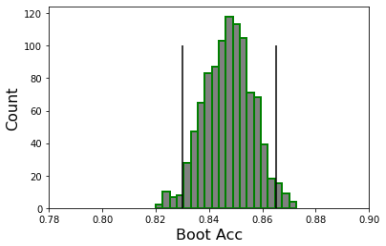
(3) Logistic Regression:
Accuracy: 0.842
95% confidence interval [0.824 , 0.85933]
Confusion Matrix: [1114 43]
[194 149] TPR=0.434



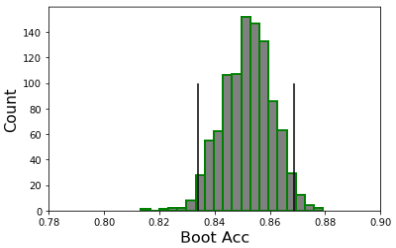
(4) CART:
Accuracy: 0.824
95% confidence interval [0.803 , 0.8426]
Confusion Matrix: [1122 35]
[229 114] TPR=0.3324



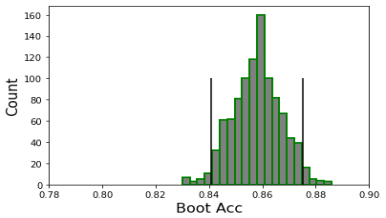
(5) Random Forest:
Accuracy: 0.847
95% confidence interval [0.83 , 0.865333]
Confusion Matrix: [1107 50]
[179 164] TPR=0.478



(6) Gradient Boosting Tree
Accuracy:0.852
95% confidence interval [0.834 , 0.868667]
Confusion Matrix: [1103 50]
[168 175] TPR=0.5102



(7) XGBoost
Accuracy: 0.858
95% confidence interval [0.8406667 , 0.8753337]
Confusion Matrix: [1111 46]
[167 176] TPR=0.513



The following table shows the accuracy of each model on the test set and run time.

Model	Accuracy	Time(s)
Base Model	0.771	0
SVM-Linear Kernel	0.844	480
SVM-Poly Kernel	0.851	25
SVM-RBF Kernel	0.842	25
Logistic Regression	0.842	22
CART	0.824	22
Random Forest	0.847	182
Gradient Boosting Tree	0.852	284
XGBoost	0.858	681

Evaluation:

- In general, all of our models demonstrated higher accuracy than the baseline model (0.771).
- If we only consider accuracy as the metrics, XGBoost is the most powerful model with 0.858 accuracy.
- Taking running time into account, 'SVM-Poly Kernel might be the best model with balanced running time and accuracy.

4.Impact:

（1）People:
Prediction on whether or not it will rain tomorrow helps people adjust their ways to work and trip plans.

（2）Industries:
Agriculture:
Farm operation is highly dependent on weather conditions. Continuous rainy days may lead to crop harvest and livestock's living standards. The prediction helps farm owners make necessary adjustments to the schedule to avoid economic loss.

Insurance:
Insurance companies care about if it rains tomorrow because many natural hazards such as floods may damage the construction, increase the possibility of car incidents, etc. The prediction helps insurance companies avoid economic loss.

Commercial Fishing:
Commercial fishing companies depend on weather prediction to decide whether or not they will go fishing on that day. The accurate prediction could potentially avoid economic and livelihood loss.

Tourism:
Prediction helps tourism companies to better plan the trip and serve customers.