(Group 2) Machine Learning Mini Project – Rainfall Prediction in Australia

Objectives

- To train the best model to predict whether it will rain the next day in Australia

Dataset Description

We used a dataset from Kaggle, which is originated from the Bureau of Meteorology of the Australian Government. It consists of 10 years of daily weather observations from numerous locations across Australia covering from 1 Dec 2008 to 25 Jun 2017. There are 23 columns (variables) and 145,460 rows of weather observations records. Data is stored in mixture of Numerical data, Categorical data and Date.
Attached here the link of the dataset: https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package/data

Major information is summarized as below:

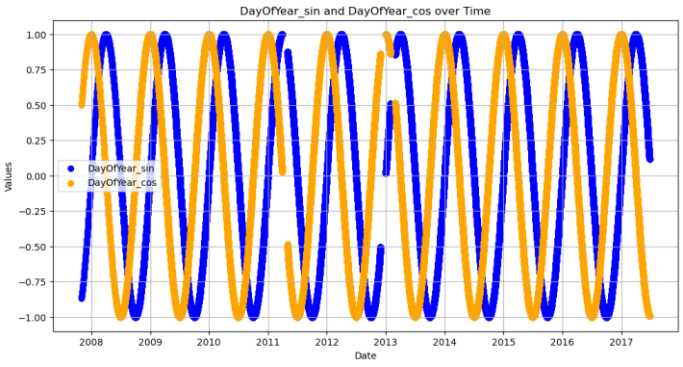
Variables	Information
Date	Day of the month
Location	Various location of Australian weather stations
Temperature	Maximum and Minimum temperature in the 24 hours to 9am
Rainfall	Rainfall in the 24 hours to 9am
Evaporation	"Class A" pan evaporation in the 24 hours to 9am
Sunshine	Bright sunshine in the 24 hours to midnight
Maximum Wind Gust	Direction, Speed, Time of strongest wind gust in the 24 hours to midnight
9am Various Measurement	Temperature, Relative humidity, Fraction of sky obscured, Wind direction averaged, Wind speed averaged, Atmospheric pressure reduced to mean sea level
3pm Various Measurement	Temperature, Relative humidity, Fraction of sky obscured, Wind direction averaged, Wind speed averaged, Atmospheric pressure reduced to mean sea level

Methodology

Before getting into the model training session, a series of feature engineering steps were performed progressively to turn raw data into effective features.

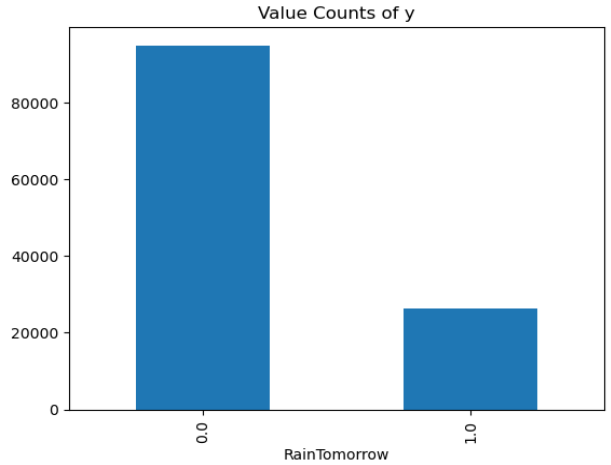
Pre-Processing

	count	count(%)	mean	min	0.25	0.50	0.75	max	std
Date	145460	100%	41,368.88	39,387.00	40,554.00	41,427.00	42,169.00	42,911.00	NaN
MinTemp	143975	99%	12.19	-8.50	7.60	12.00	16.90	33.90	6.40
MaxTemp	144199	99%	23.22	-4.80	17.90	22.60	28.20	48.10	7.12
Rainfall	142199	98%	2.36	-	-	-	0.80	371.00	8.48
Evaporation	82670	57%	5.47	-	2.60	4.80	7.40	145.00	4.19
Sunshine	75625	52%	7.61	-	4.80	8.40	10.60	14.50	3.79
WindGustSpeed	135197	93%	40.04	6.00	31.00	39.00	48.00	135.00	13.61
WindSpeed9am	143693	99%	14.04	-	7.00	13.00	19.00	130.00	8.92
WindSpeed3pm	142398	98%	18.66	-	13.00	19.00	24.00	87.00	8.81
Humidity9am	142806	98%	68.88	-	57.00	70.00	83.00	100.00	19.03
Humidity3pm	140953	97%	51.54	-	37.00	52.00	66.00	100.00	20.80
Pressure9am	130395	90%	1,017.65	980.50	1,012.90	1,017.60	1,022.40	1,041.00	7.11
Pressure3pm	130432	90%	1,015.26	977.10	1,010.40	1,015.20	1,020.00	1,039.60	7.04
Cloud9am	89572	62%	4.45	-	1.00	5.00	7.00	9.00	2.89
Cloud3pm	86102	59%	4.51	-	2.00	5.00	7.00	9.00	2.72
Temp9am	143693	99%	16.99	-7.20	12.30	16.70	21.60	40.20	6.49
Temp3pm	141851	98%	21.68	-5.40	16.60	21.10	26.40	46.70	6.94
RainToday	142199	98%	0.22	-	-	-	-	1.00	0.42
RainTomorrow	142151	98%	0.22	-	-	-	-	1.00	0.42

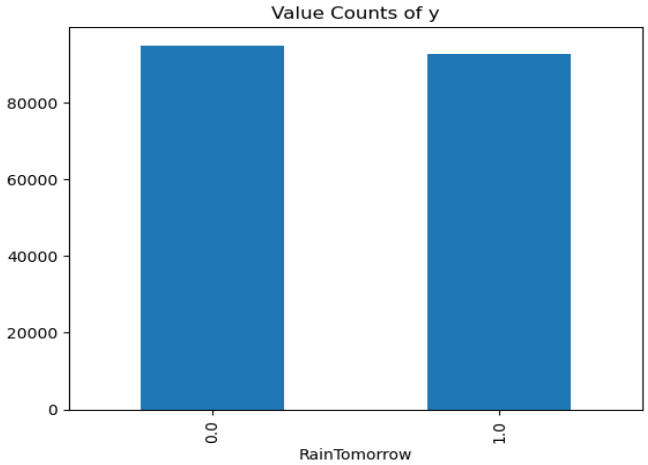


Missing Values was filled by propagating the last valid observation to next valid observation or by mode.
One-Hot Encoding was performed for categorical data to prepare for machine learning process.

Re-sampling



(before ADASYN)



(after ADASYN)

Since there is class imbalance for the target variable “RainTomorrow”, it is essential to take measure to avoid it from leading to biases towards the majority class in modeling. Adaptive Synthetic Sampling (ADASYN) is considered a more appropriate oversampling technique in this case as it has taken into consideration the distribution of the original sample in the synthetic process. Therefore, ADASYN was adopted.

Train-Test Split

In order to avoid data leakage that the machine might possibly learn about the future in case of a random train-test time series data split, the dataset was thus segregated into train and test set using the date 1 January 2016 as a dividing point. The resulting training set and testing set occupied around 82% and 18% of the whole dataset respectively.

Feature Scaling

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

X_train = scaler.fit_transform(x_train)

X_test = scaler.transform(x_test)
```

(Min-Max Scaling)

It was observed that features fell into distribution of different scale. To make a fair calculation in the model training process where some algorithms with distance metrics are highly sensitive to scale difference, feature scale was thus required in this case. Min-Max Scaling technique was preferred and adopted as it avoided any possible negative value from happening in the input data where a negative value does not make sense to be existed in variables like rainfall, evaporation, sunshine, etc.

After feature engineering steps were finished, features were ready for machine learning.

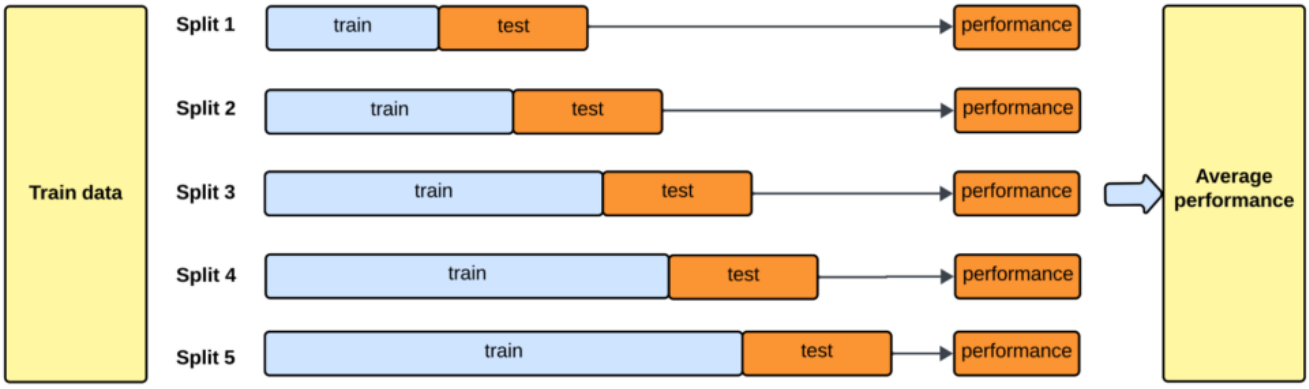
Machine Learning Approach

Supervised Learning is destined for this study as there is resulting target variable “RainTomorrow” for each daily observation.

Model Selection and Hyperparameter Tuning

Model	Hyperparameter Tuned
Logistic Regression	Solver , Penalty
K Nearest Neighbor (KNN)	N neighbors , Metric
Decision Tree	Criterion , Maximum of Depth
Random Forest	Criterion , Maximum of Depth

Several models were picked for training and compared for the best prediction of whether it will rain tomorrow in Australia. It includes Logistic Regression, K Nearest Neighbor (KNN), Decision Tree and Random Forest.
In order to optimize the performance of each model, Grid Search with certain hyperparameters tuned were done. The list of hyperparameters tuned was listed out on the above table. To strike a balance between the performance of models and computational efficiency of the training process, not all hyperparameters were tuned and only certain impactful ones were picked.



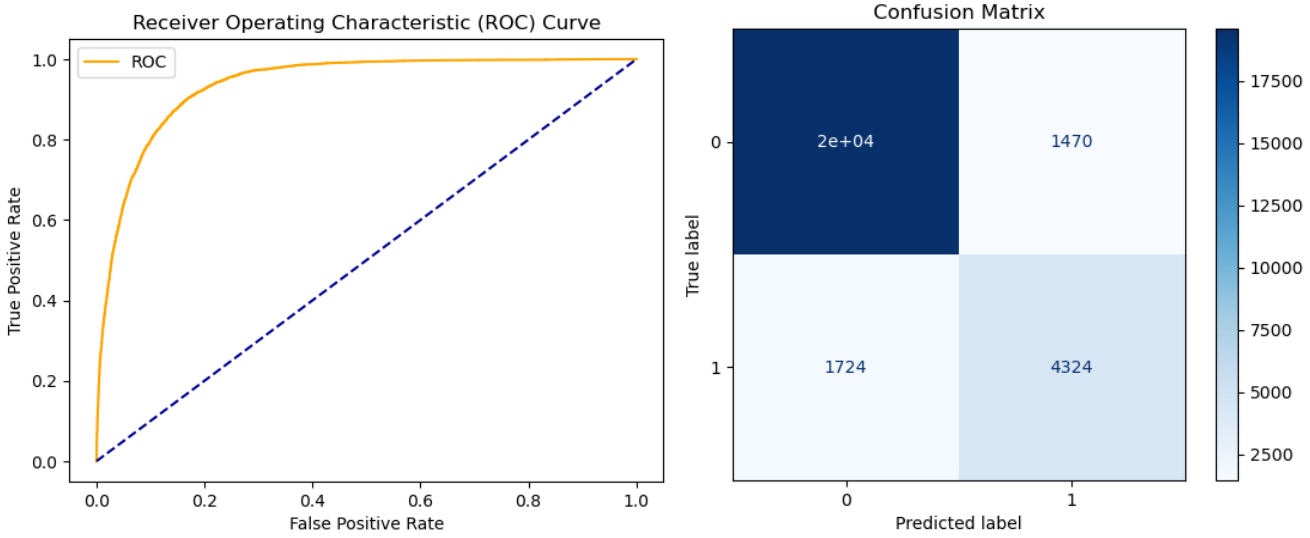
In addition, Time Series Split Cross Validation was adopted for Grid Search. It features that the first chunk of train data will never be included in the test set and is perfect to avoid any potential data leakage in the training process.

Analysis

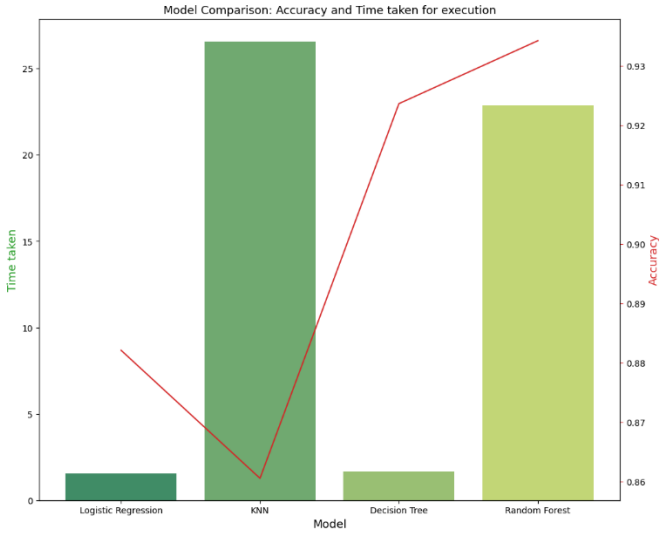
Our analysis started with certain set of hyperparameters at the beginning for each of the model. It is hereby illustrated using Logistic Regression. From that, major outcomes like Accuracy, ROC Area Under Curve, Precision, Recall, etc are found for each model. This step is repetitive across all other models.

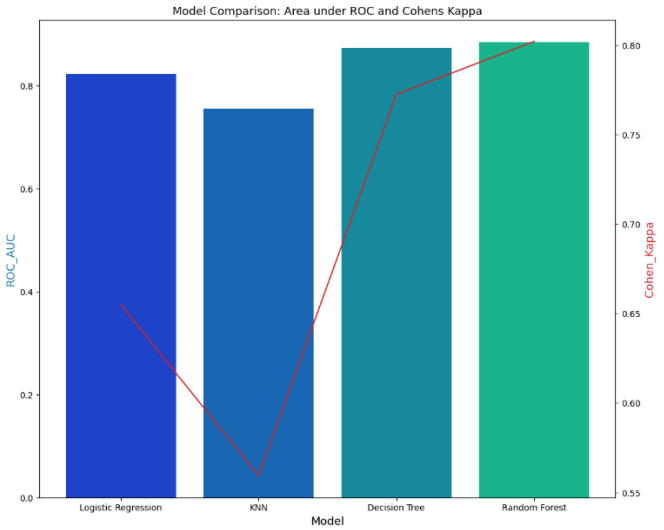
```
params_lr2 = {'penalty': 'l2', 'solver': 'newton-cg'}

model_lr2 = LogisticRegression(**params_lr2)
model_lr2, accuracy_lr2, roc_auc_lr2, coh_kap_lr2, tt_lr2 = run_model(model_lr2,
                                                                       columnsWithoutList(X_train,col_set_feature_engineering),
                                                                       y_train,
                                                                       columnsWithoutList(X_test,col_set_feature_engineering),
                                                                       y_test)
```

```
Accuracy = 0.8821358721724049
ROC Area under Curve = 0.8225583390450857
Cohen's Kappa = 0.6549179906108287
Time taken = 1.5566654205322266
              precision    recall  f1-score   support

         0.0    0.91908   0.93017   0.92459     21051
         1.0    0.74629   0.71495   0.73028      6048

    accuracy                        0.88214     27099
   macro avg    0.83268   0.82256   0.82744     27099
weighted avg    0.88052   0.88214   0.88123     27099
```

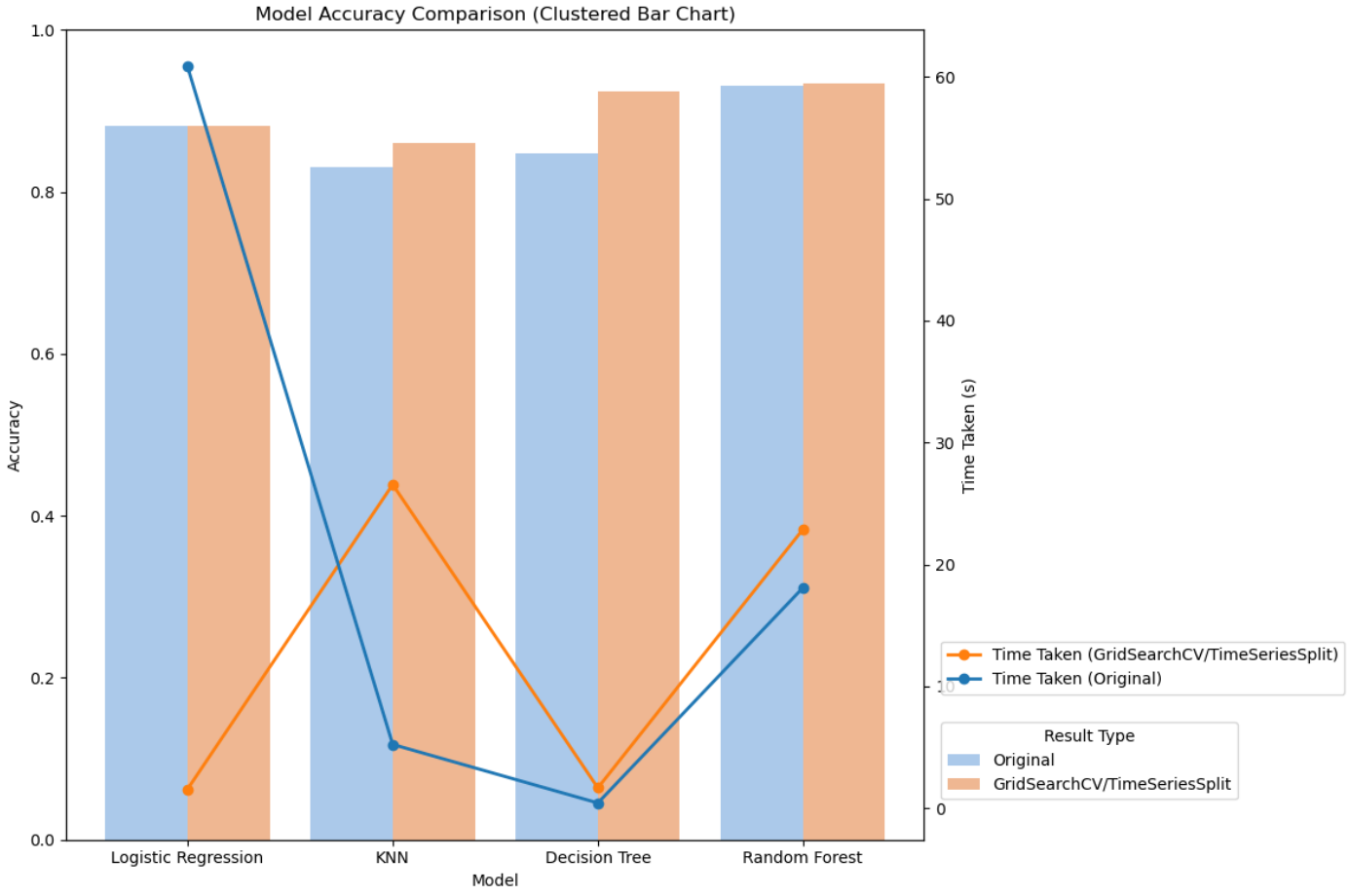


Before grid search, all the model training outcomes are compared and represented by the graphs below. It is found that Random Forest performed the best among all with the highest accuracy, decent time taken, highest ROC_AUC and Cohen's Kappa.





After grid search with certain hyperparameters tuned, it is still found that Random Forest performed the best among all with the highest accuracy and decent time taken. It is also observed that the increment of accuracy differed across model in the grid search procedure and time taken towards reaching the best hyperparameters combination in each model is generally longer.



Challenges and Limitations

Major challenges and limitations encountered in this study are listed in the below table.

Data Leakage	Train-Test Split by randomness is not appropriate.
Class Imbalance	The most appropriate method for balancing class had to be found.
Other Factors	The dataset might not have incorporated all factors that lead to rainfall.

Future Work

Areas for improvement considered for better future work are listed in the below table.

Feature Selection	Selecting features that are the most decisive in predicting rainfall is currently limited by domain expertise.
Incorporation of more Models	It is currently limited by time and knowledge. There might be other models that better fit the dataset which could help generate a more accurate prediction.
Incorporation of more Factors	For integrity of the model, expertise in rainfall causality and comprehensive observations of all factors are required but are constraints in this study.

Conclusion

Our team has successfully trained several models, one of which at best achieved 92.4% accuracy in the testing session. Although there is always no perfection and there is room for improvement, we believe the model trained is acceptable considered the knowledge we possess at present. We also will not hold back in consistently evolving all the aspects for future modeling.

Work Distribution

Even distribution among team members