CS5805 Final Project

PREDICTING RAIN IN AUSTRALIA

Richard James Heiman

Virginia Tech

Table of Figures and Tables

Abstract

Accurate rainfall prediction is critical for agriculture, water resource management, and disaster mitigation, particularly in regions like Australia, which experiences significant climatic variability. This study explores the application of machine learning techniques to predict rainfall, leveraging historical meteorological data from Australian weather stations. Several models, including logistic regression, decision trees, random forests, and neural network approaches, were evaluated for their predictive accuracy. Key meteorological variables such as temperature, humidity, wind speed, and atmospheric pressure were used as inputs. Feature selection and preprocessing techniques were employed to address data imbalances and enhance model performance. Results indicate that ensemble methods, particularly random forests, achieved superior accuracy and reliability, outperforming traditional statistical methods. This research demonstrates the potential of machine learning to provide robust rainfall forecasts, contributing to more informed decision-making in weather-sensitive sectors. Future work includes integrating real-time data and expanding the scope to account for extreme weather events.

Introduction

This project involved four phases: starting off with an exploratory data analysis and feature engineering to determine and/or engineer the target feature for prediction, moving into a regression analysis to create an algorithm for predicting the amount of rainfall for the next day, then transitioning into the bulk of the work where a simple prediction is made whether it will rain or not, ending with an experimental phase looking at various clusters of the weather data to find emergent patterns and to see any associations between the various features in the set.

This report is outlined as such, beginning with a description of the data, then diving deeper into the four phases described above, finally ending with a summary and conclusions/recommendations derived from the results of the project.

Description of the Dataset

This proposed dataset "comprises about 10 years of daily weather observations from numerous locations across Australia" which is to be used to answer the question: Will it rain in the next day? This is publicly available data collected by the Bureau of Meterology (Australia's version of NOAA) (Joe Young, 2020). The data collected has several categorical columns including "location", "date', and "wind gust direction" and 17 numerical features.

```
Number of Categorical Features: 7
Number of Numerical Features: 16
Time Range: 2007-11-01 to 2017-06-25
Columns:
Index(['Date', 'Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation',
       'Sunshine', 'WindGustDir', 'WindGustSpeed', 'WindDir9am', 'WindDir3pm',
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
       'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am',
       'Temp3pm', 'RainToday', 'RainTomorrow'],
      dtype='object')
```

Figure 1: Summary of Dataset

Phase I

The first phase involved running an exploratory data analysis on the dataset and determining what the target variable should be for phases II and III. The first step was cleaning the data, which involved dropping any duplicates (of which there were none) and handling any nulls. The nulls were filled several different ways. The binary categorical features, ‘Rain’ and ‘RainTomorrow’, that had nulls were dropped, especially as they were logically significant data. Other categorical features were filled in by mode by location. Similarly, the numerical features were filled in by mean by location. The data was then checked for collinearity using both a correlation matrix and, later, variance inflation factor analysis.

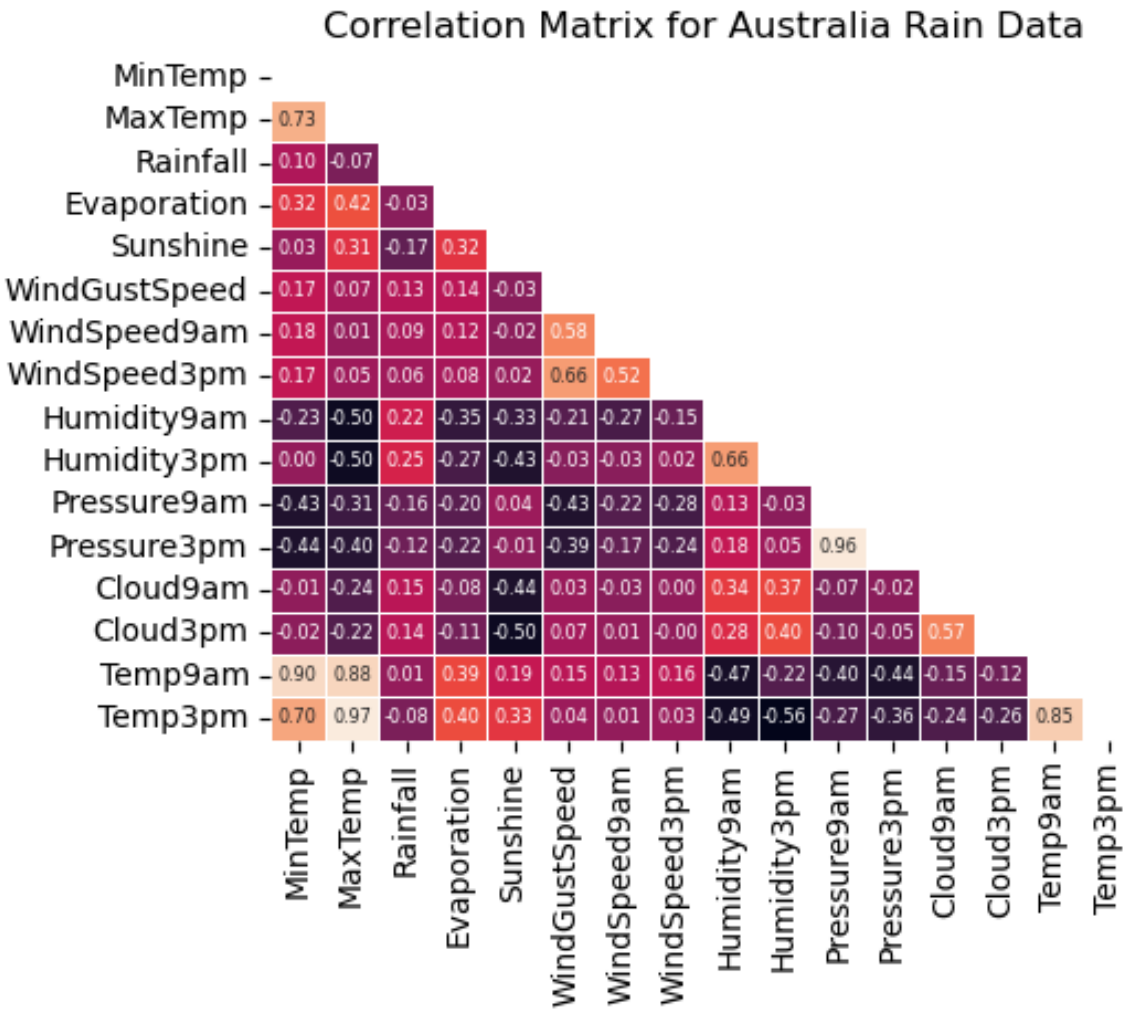

Figure 2: Correlation Matrix Data

The ‘Location’ feature was encoded using a Target Encoding approach, using the average ‘Rainfall’ value as the fill value. All other categorical features were one hot encoded.

For both the regression and the classification phase, the training and test data split was set to 80-20.

Specific to the regression phase, a new target variable was engineered, ‘Rainfall_Tomorrow’, to provide a numeric target for the regression analysis to predict. This was done by using the basic Pandas ‘shift’

function, grouping data by location.. (NumFOCUS, Inc, 2024). No data scaling was performed for the linear regression analysis.

	Rainfall	Rainfall_Tomorrow
0	0.600000	0.000000
1	0.000000	0.000000
2	0.000000	0.000000
3	0.000000	1.000000
4	1.000000	0.200000
5	0.200000	0.000000
6	0.000000	0.000000
7	0.000000	0.000000
8	0.000000	1.400000
9	1.400000	0.000000
10	0.000000	2.200000
11	2.200000	15.600000
12	15.600000	3.600000
13	3.600000	0.000000
14	0.000000	1.914115
15	1.914115	0.000000
16	0.000000	16.800000
17	16.800000	10.600000
18	10.600000	0.000000
19	0.000000	0.000000

Figure 3: Feature Engineering for Linear Regression

As stated before, variance inflation factor was used to check for collinearity, and proved to be quite useful as a method of feature reduction, taking the preprocessed data from a set of 63 features down to 18, with VIF cutoff of > 5. The exceptions to this were ‘Rainfall’ and ‘RainToday_Yes’ which were deemed too logically important and were collinear for obvious reasons.

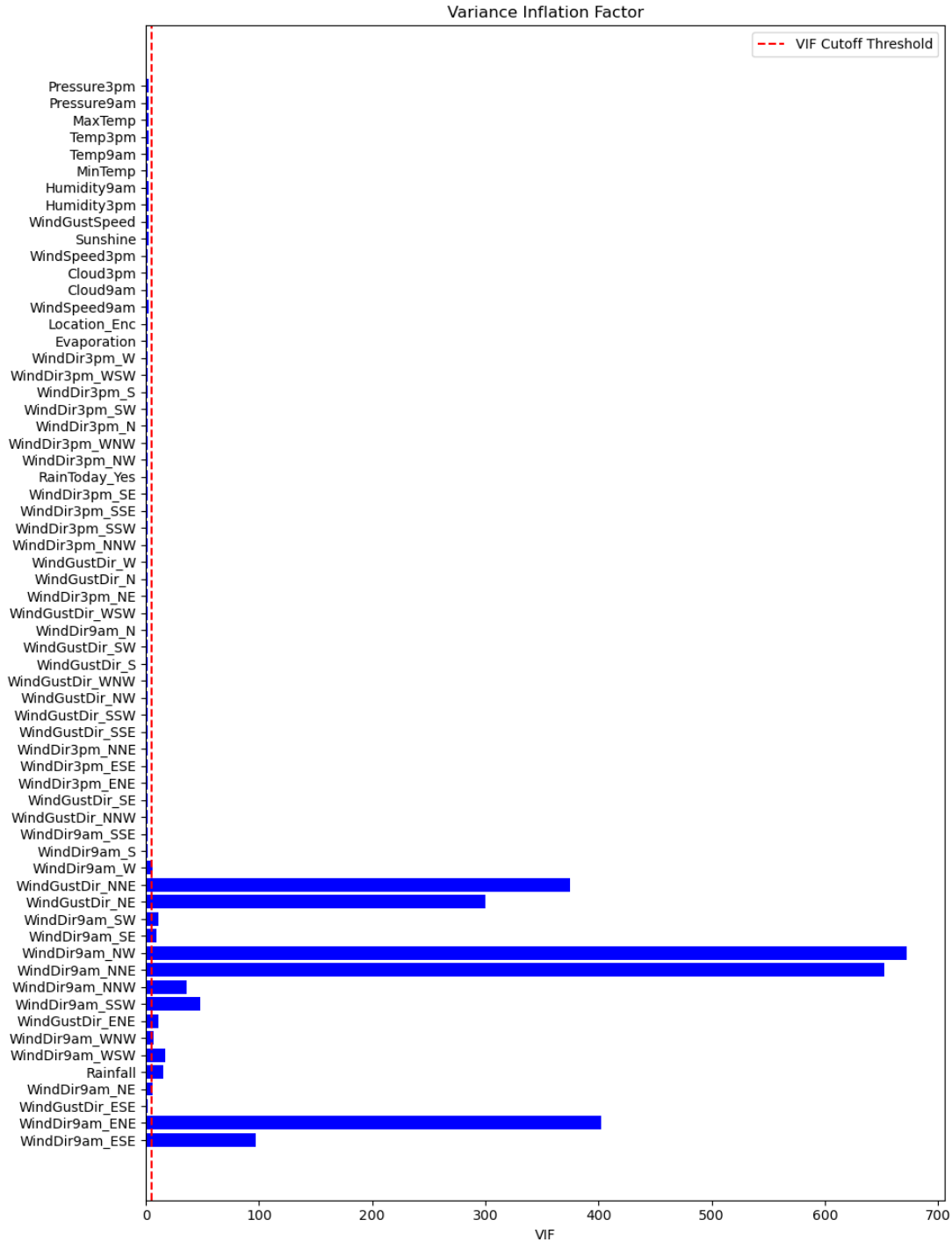

Figure 4: VIF of Preprocessed Data

For the classification phase, the target variable was set to ‘RainTomorrow_Yes’, a boolean classifier. The SMOTE method was used to balance the data according to the target variable. After splitting the data into test and train, PCA was applied for any additional feature reduction, bringing the final tally down to 17 features.

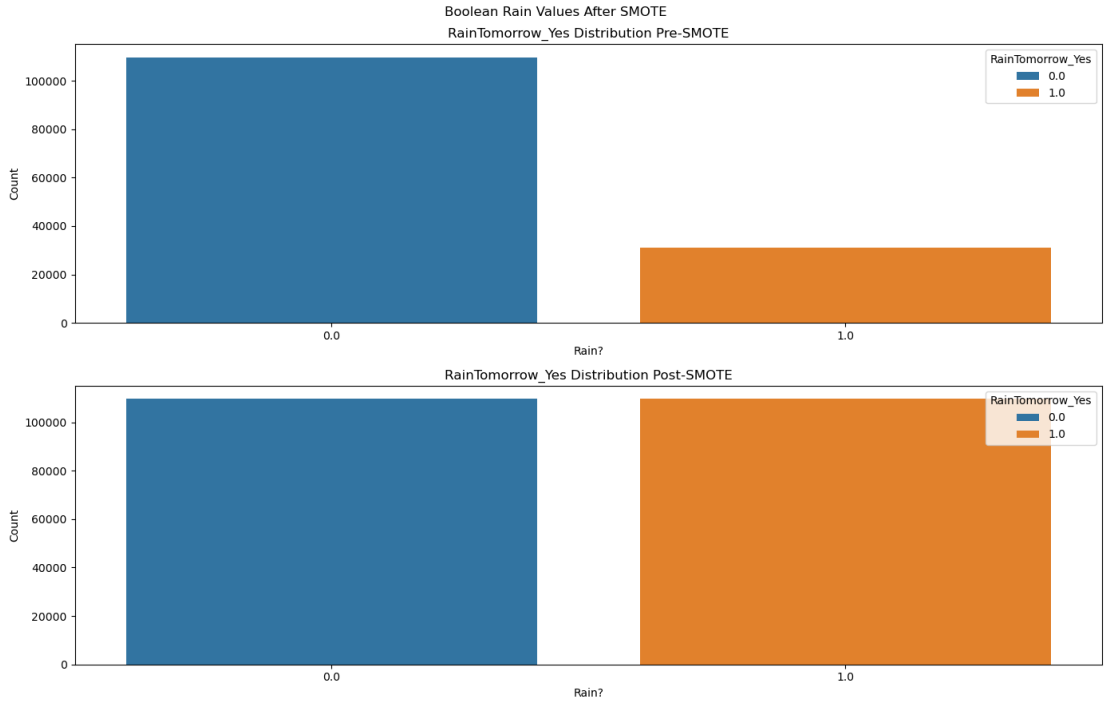

Figure 5: SMOTE Rebalancing

```
Old dimensionality:  (175337, 18) (43835, 18) New dimensionality:  (175337, 17) (43835, 17)
```

Figure 6: Classification Analysis Feature Set Dimensionality

Phase II

Phase II of this project was dedicated to using linear regression in order to predict a continuous target variable, ‘Rainfall_Tomorrow’. I.e. asking the question: “How much will it rain tomorrow?”.

For all modeling in this phase, the statsmodels linear regression OLS (Ordinary Least Squares) API was used (Perktold, Seabold, Taylor, & statsmodels-developers, n.d.). The first step in this process was evaluating the base model.

```
Base Model Score:  0.26128431894532356
Base Model Regression Summary
                                 OLS Regression Results
================================================================================
Dep. Variable:      Rainfall_Tomorrow   R-squared (uncentered):                   0.261
Model:                            OLS   Adj. R-squared (uncentered):              0.261
Method:                 Least Squares   F-statistic:                              2214.
Date:                Fri, 06 Dec 2024   Prob (F-statistic):                        0.00
Time:                        09:26:38   Log-Likelihood:                     -3.8454e+05
No. Observations:              112629   AIC:                                  7.691e+05
Df Residuals:                  112611   BIC:                                  7.693e+05
Df Model:                          18
Covariance Type:            nonrobust
```

Figure 7: Base Model OLS Summary

Tuning of the model involved using the feature selection techniques Random Forest Analysis and Backwards Stepwise Regression.

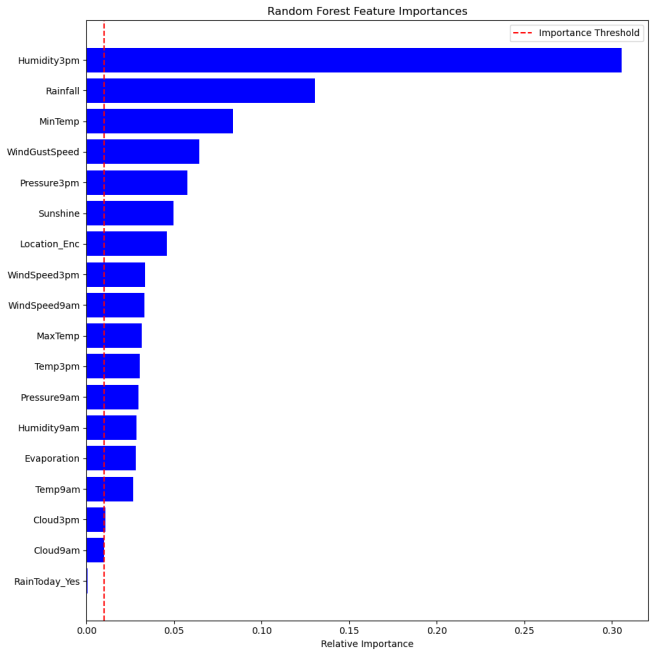

Figure 8: Random Forest Feature Importances

```
Random Forest Model Score:  0.26110447739518017
Random Forest Model Regression Summary
                            OLS Regression Results
==============================================================================
Dep. Variable:      Rainfall_Tomorrow   R-squared (uncentered):                   0.261
Model:                            OLS   Adj. R-squared (uncentered):              0.261
Method:                 Least Squares   F-statistic:                              2342.
Date:                Sat, 07 Dec 2024   Prob (F-statistic):                        0.00
Time:                        13:22:40   Log-Likelihood:                     -3.8455e+05
No. Observations:              112629   AIC:                                  7.691e+05
Df Residuals:                  112612   BIC:                                  7.693e+05
Df Model:                          17
Covariance Type:            nonrobust
```

Figure 9: RFA Linear Model Metrics

Backwards stepwise regression did not provide any suggestions for features to be dropped, with any accuracy being lost if any features were not kept, which is evident in the RFA above. Therefore, the model ended up being the same as the base model.

```
Backwards Stepwise Regression Model Score:  0.26128431894532356
Backwards Stepwise Regression Summary
                            OLS Regression Results
==============================================================================================
Dep. Variable:      Rainfall_Tomorrow   R-squared (uncentered):                   0.261
Model:                            OLS   Adj. R-squared (uncentered):              0.261
Method:                 Least Squares   F-statistic:                              2214.
Date:                Sat, 07 Dec 2024   Prob (F-statistic):                        0.00
Time:                        13:22:44   Log-Likelihood:                     -3.8454e+05
No. Observations:              112629   AIC:                                  7.691e+05
Df Residuals:                  112611   BIC:                                  7.693e+05
Df Model:                          18
Covariance Type:            nonrobust
```

Figure 10: BSR Linear Model Metrics

Collecting metrics for the above three methods produced the following table.

Final Regression Model Evaluation Table

Model Iteration	AIC	BIC	R-Squared Adj
Base	769115	769288	0.261284
Random Forest Model	769142	769305	0.261104
Backwards Stepwise Regression Model	769115	769288	0.261284

Figure 11: Linear Regression Metrics Table

Looking at the above metrics, it was a clear choice as to which iteration should be used for final evaluation: the base and/or BSR model, which were identical.

The final model was evaluated using the same metrics as above, with the addition of a confidence interval analysis for each feature in the final set, the Mean Squared Error of the prediction vs the actual ‘Rainfall’, and then with a visual comparison of the predicted vs actual ‘Rainfall’.

```
Mean Squared Error of Final Linear Regression Model:  70.52657273221448
                                OLS Regression Results
==============================================================================================
Dep. Variable:      Rainfall_Tomorrow   R-squared (uncentered):                   0.261
Model:                            OLS   Adj. R-squared (uncentered):              0.261
Method:                 Least Squares   F-statistic:                              2214.
Date:                Fri, 06 Dec 2024   Prob (F-statistic):                        0.00
Time:                        09:33:25   Log-Likelihood:                     -3.8454e+05
No. Observations:              112629   AIC:                                  7.691e+05
Df Residuals:                  112611   BIC:                                  7.693e+05
Df Model:                          18
Covariance Type:            nonrobust
```

Figure 12: Final Linear Model Metrics

```
Confidence Interval For Final Model:
MinTemp         -1.304950   0.317743
MaxTemp         -2.120794   1.020492
Rainfall        75.296001  79.808347
Evaporation      3.983982   8.569772
Sunshine        -4.925162  -4.358903
WindGustSpeed   11.788231  13.032103
WindSpeed9am    -1.062581   0.631582
WindSpeed3pm    -7.844640  -6.643740
Humidity9am     -1.601533  -0.776336
Humidity3pm     10.954499  11.877264
Pressure9am     27.404432  30.636036
Pressure3pm    -37.767927 -34.508224
Cloud9am        -0.982753  -0.580522
Cloud3pm         0.606332   1.079768
Temp9am         -0.374800   2.413979
Temp3pm          1.759733   4.851817
Location_Enc    -0.431822   0.065650
RainToday_Yes    0.227632   0.492362
```

Figure 13: Confidence Interval Analysis for Final Model

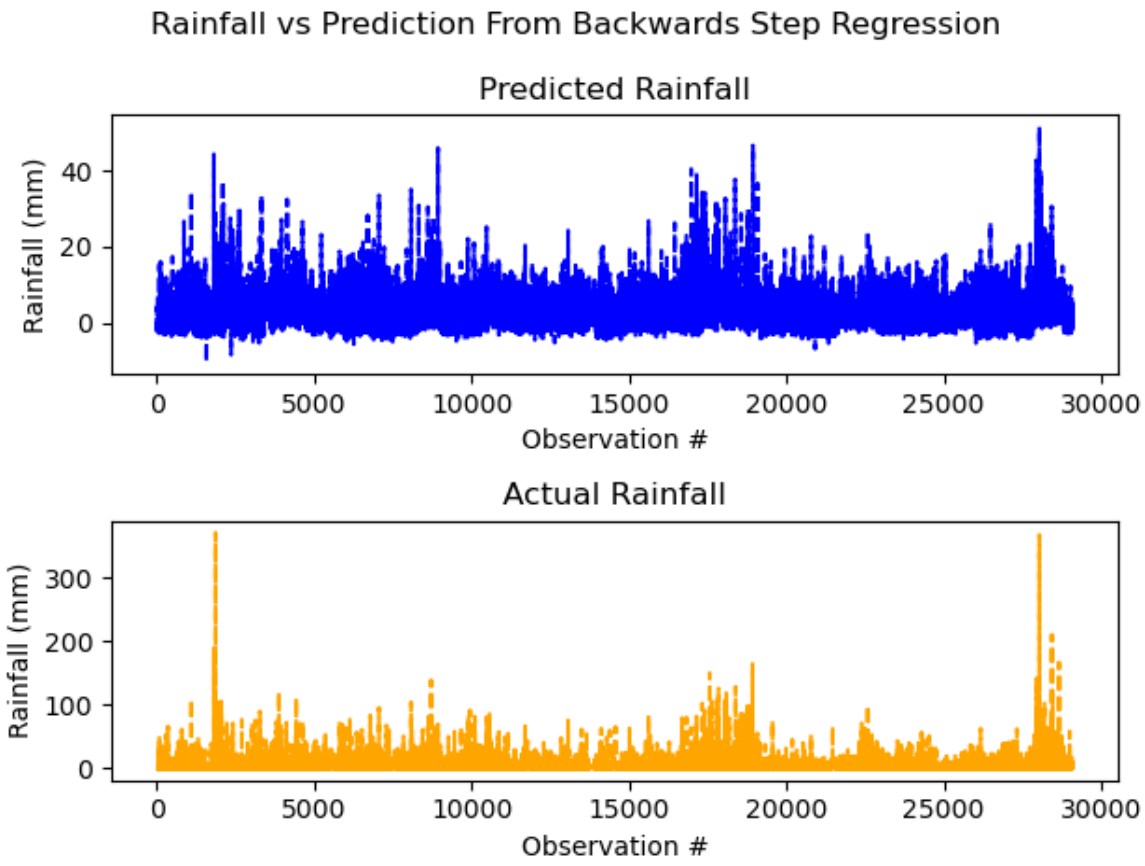

Figure 14: Predicted vs Actual Rainfall from Linear Regression

Phase III

Phase III of the project focused on using various classification models to predict a target 'RainTomorrow_Yes' feature. I.e. answering the question, "Will it rain tomorrow?".

Evaluating the models for classification was done in a programmatic manner, with a grid search used to fine tune each tested classifier (with one exception) and then uniform metrics gathered, including plotting the ROC curve and graphing the confusion matrix, for each one for final selection. The data was 8-fold shuffle stratified for the grid search using the basic sklearn KFold library (developers, 2024). The classifiers tested, in order, were Decision Tree, Logistic Regression, K-Nearest Neighbors, Support Vector Machine, Naïve Bayes, Random Forest, and Multi-Layer Perceptron (Neural Network).

The exception to the tuning method used was with the Decision Tree classifier. The grid search was used for pre-pruning, while an additional optimization was run post-pruning using Cross-Complexity Analysis for determining ccp_alpha. Notably, the final model utilized the Entropy method as opposed to the Gini index.

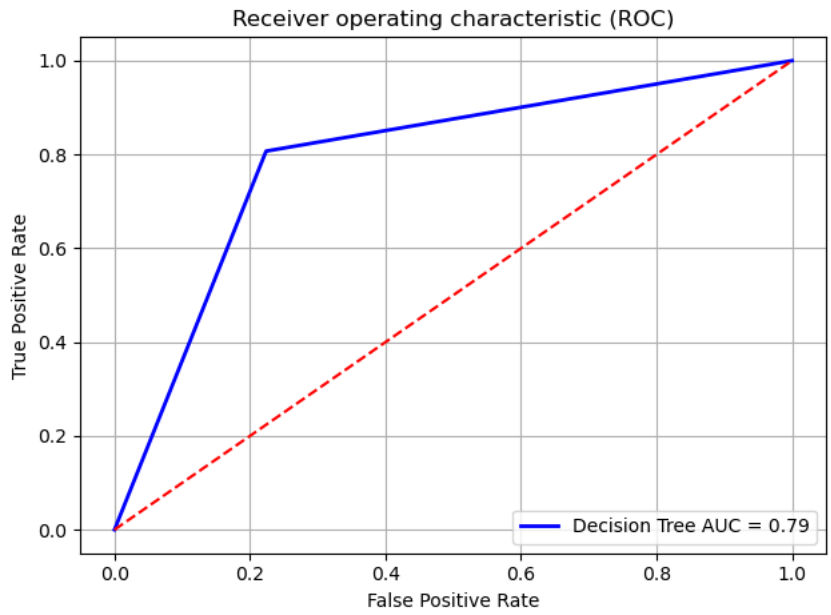

Figure 15: Decision Tree ROC

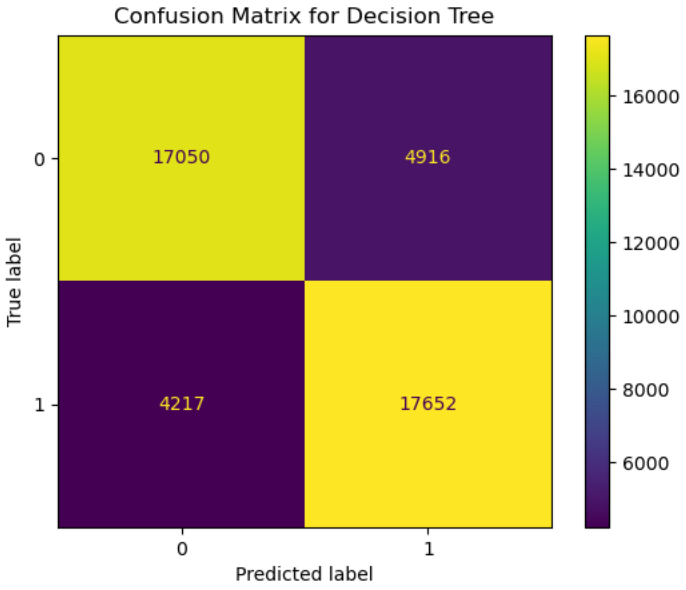

Figure 16: Decision Tree Confusion Matrix

For logisitic regression, Elasticnet was chosen as the regularization penalty, to allow for greatest breadth of tuning search, which forced the solver used to be Stochastic Average Gradient Accelerated, as per the model docs (developers, 2024).

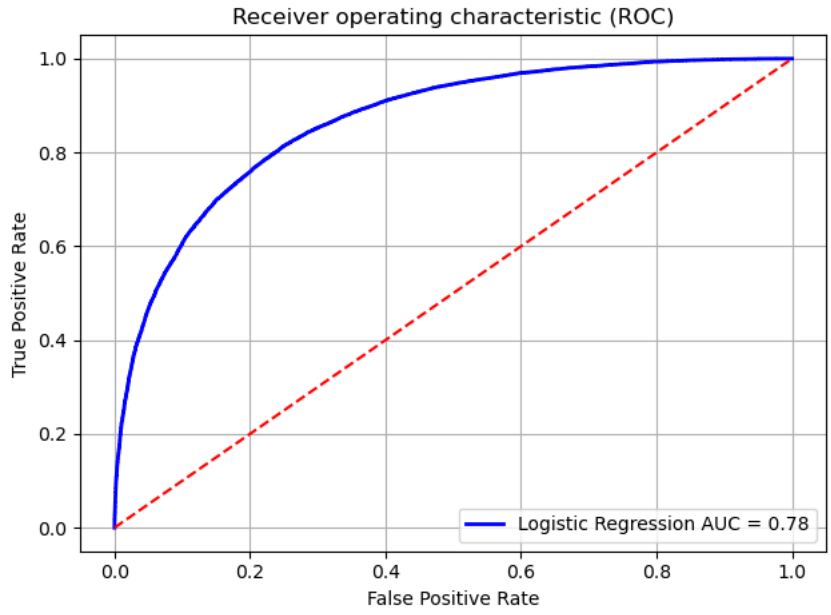

Figure 17: Logistic Regression ROC Curve

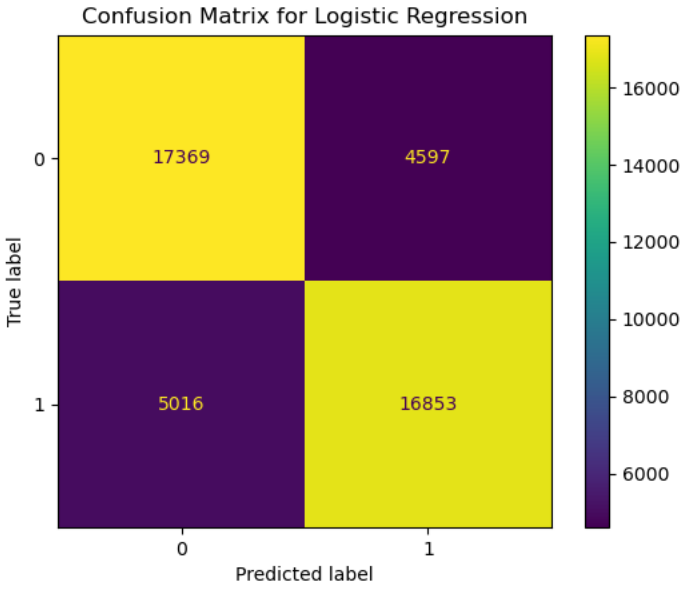

Figure 18: Logistic Regression Confusion Matrix

The parameter selection for K-Nearest Neighbors was arbitrary, choosing random values from the list provided in the source documentation, including defaults. This classifier proved to be effective given its computational requirement.

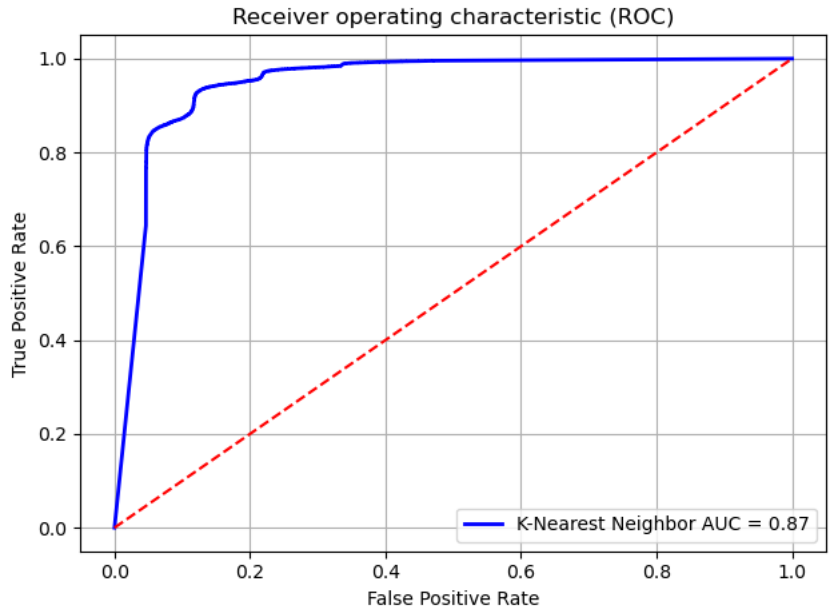

Figure 19: KNN ROC Curve

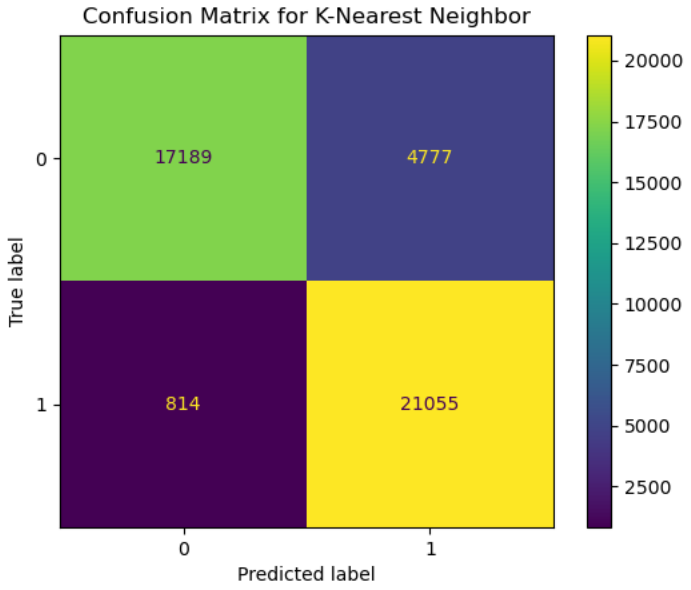

Figure 20: KNN Confusion Matrix

Support Vector Machine was by and large a difficult classifier to work with, as computationally it is extremely expensive, with the grid search running for more than 24 hours and the final selected model taking 1+ hours to fit with the data. For this, reason, the model was written statically to a file to be loaded with submission, we recommend anyone testing this code to use the statically written file. Given its computational requirements and the outcome, this classifier did not seem worthwhile.

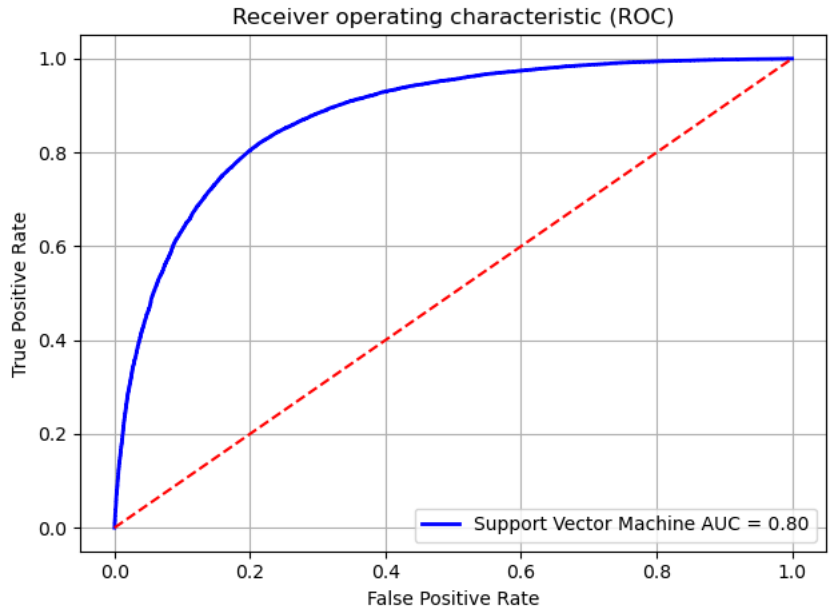

Figure 21: SVM ROC Curve

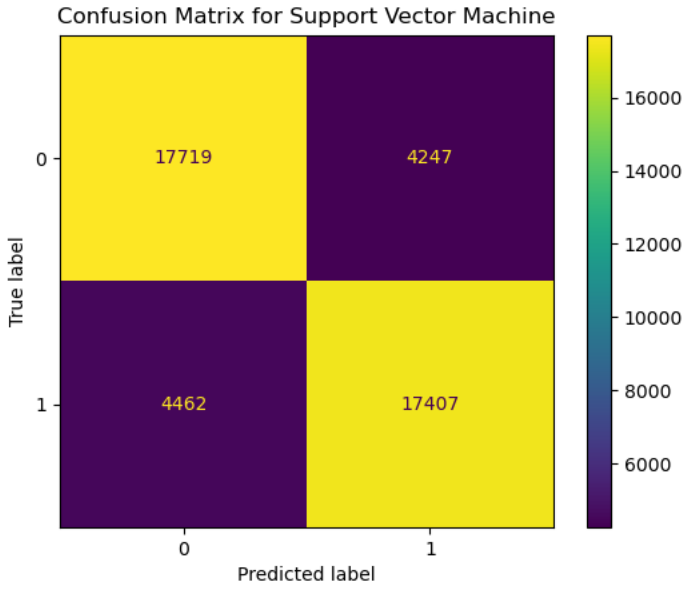

Figure 22: SVM Confusion Matrix

For Naïve Bayes, the specific Gaussian flavor of the model was chosen, as the data was both quantitative and standardized. The parameters available for selection were scarce. This classifier notably runs incredibly quickly and is just as accurate as the much more expensive SVM.

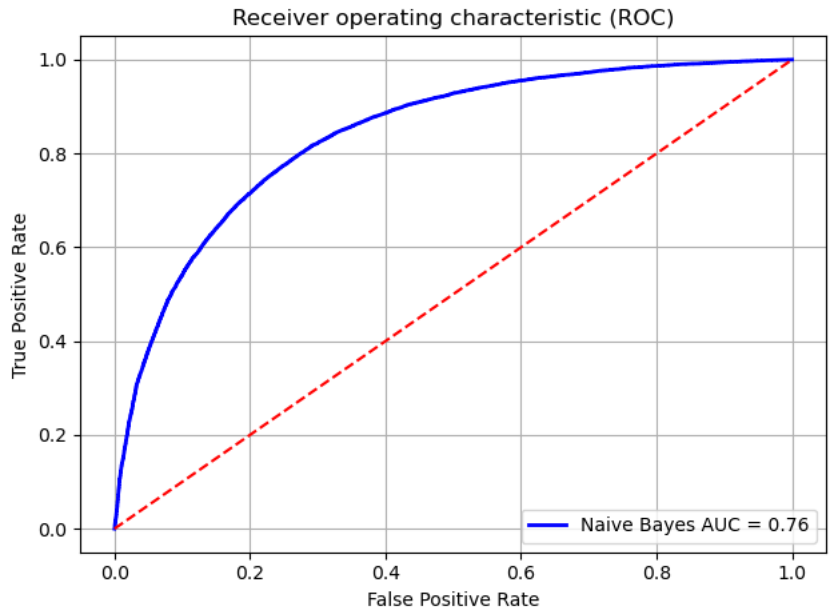

Figure 23: Naive Bayes ROC Curve

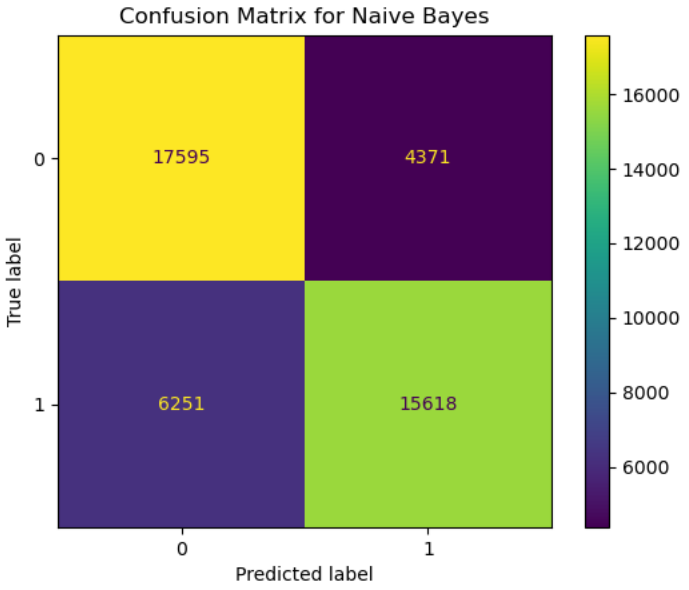

Figure 24: Naive Bayes Confusion Matrix

Random Forest was an optional classifier, as it was not covered in the course; however, it proved to be the most accurate of the list. Parameters selected were chosen based off the above Decision Tree classifier, knowing that grid search for this classifier was computationally expensive.

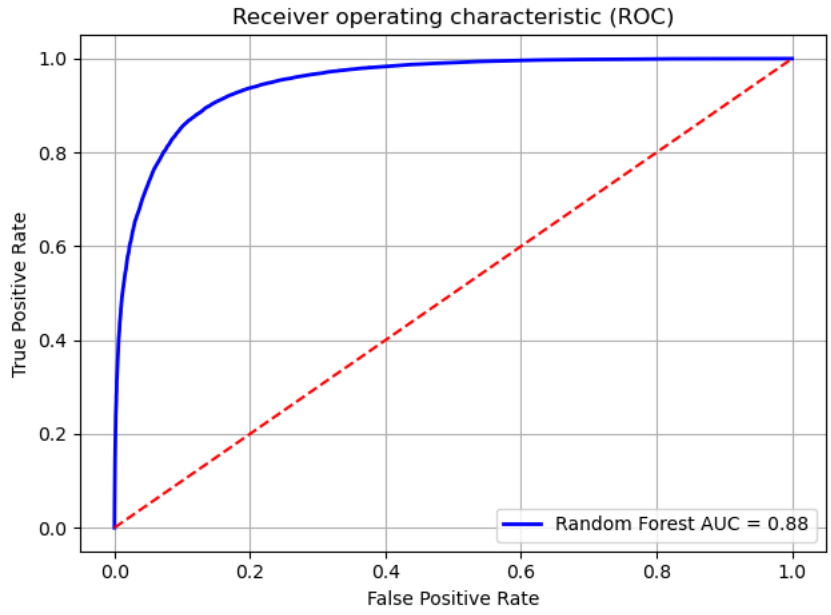

Figure 25: Random Forest ROC Curve

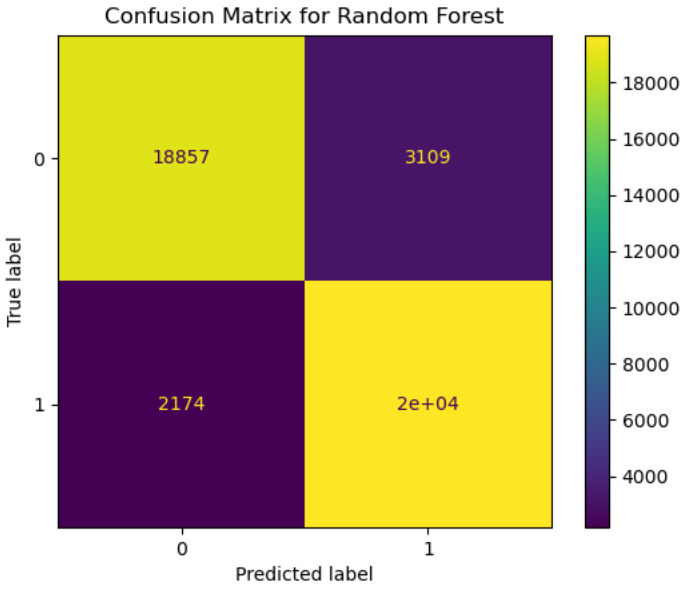

Figure 26: Random Forest Confusion Matrix

The Neural Network classifier, in the form of a Multi-Layer Perceptron, was another computationally expensive classifier. The parameters were again chosen arbitrarily based off the documentation.

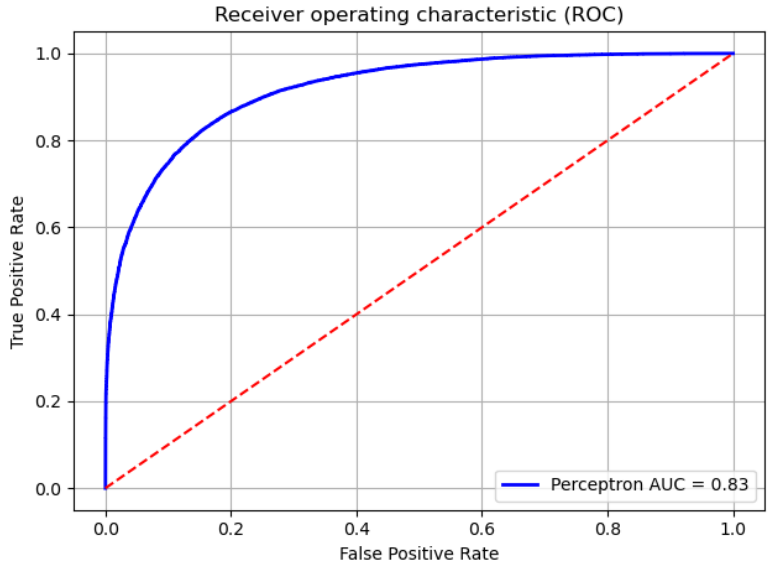

Figure 27: MLP ROC Curve

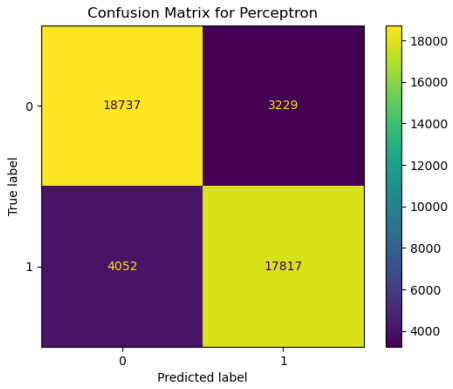

Figure 28: MLP Confusion Matrix

Metrics for each classifier were compiled into a table to then choose which one to use for final evaluation.

Final Regression Model Evaluation Table

Model	Precision	Recall	Specificity	F1-Score	AUC ROC
Decision Tree	0.782169	0.80717	[0.80171157 0.21783056]	0.794473	0.791685
Logistic Regression	0.785688	0.770634	[0.77592138 0.21431235]	0.778088	0.780678
K-Nearest Neighbor	0.815074	0.962778	[0.95478531 0.18492567]	0.882791	0.872653
Support Vector Machine	0.80387	0.795967	[0.79883684 0.19613005]	0.799899	0.801311
Naive Bayes	0.78133	0.714162	[0.7378596 0.21867027]	0.746237	0.757586
Random Forest	0.863664	0.90059	[0.89662879 0.13633573]	0.881741	0.879526
Perceptron	0.846574	0.814715	[0.82219492 0.15342583]	0.830339	0.833857

Figure 29: Classifier Evaluation Table

Based on the above table, both Random Forest and KNN proved to be the most effective, producing similar metrics across the board, neither one winning out fully. Random Forest was chosen as it had the higher AUC.

Using the final model, a 2D projection using the first two columns in the training set to visualize the classification or misclassification of the data. Unfortunately, because any decision boundary visualization libraries require 2 or 3 features per target, it was unable to be graphed.

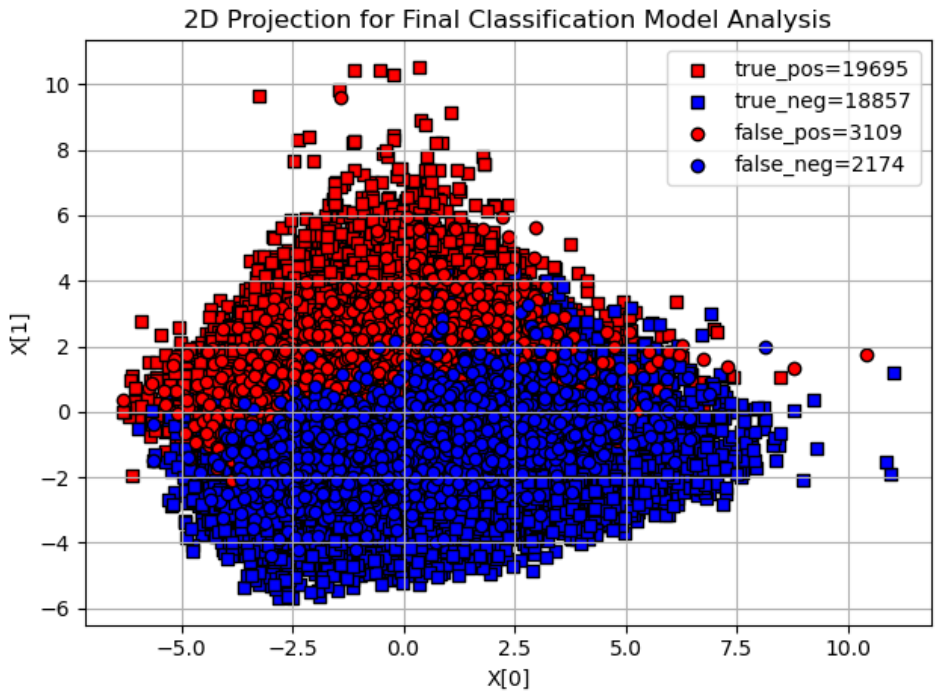

Figure 30: Final Model Classification Visualization

Phase IV

This phase of the project was experimental and could have gone several routes. For clustering, the initial route chosen was to try and identify any particular ranges of dates in which it rains more for each location. Unfortunately, the clustering algorithm used, K-Means, was not sufficient in parsing the date ranges on a year-by-year basis, due to its distance-calculation formula. Instead, the route chosen was to determine any clusters based on ‘Location’, ‘Rainfall’, and ‘Sunshine’; i.e. answering the question: “Which locations are sunny vs rainy?”.

Optimizing the number of clusters in K-Means algorithm utilized both the Elbow Method and the Silhouette Method.

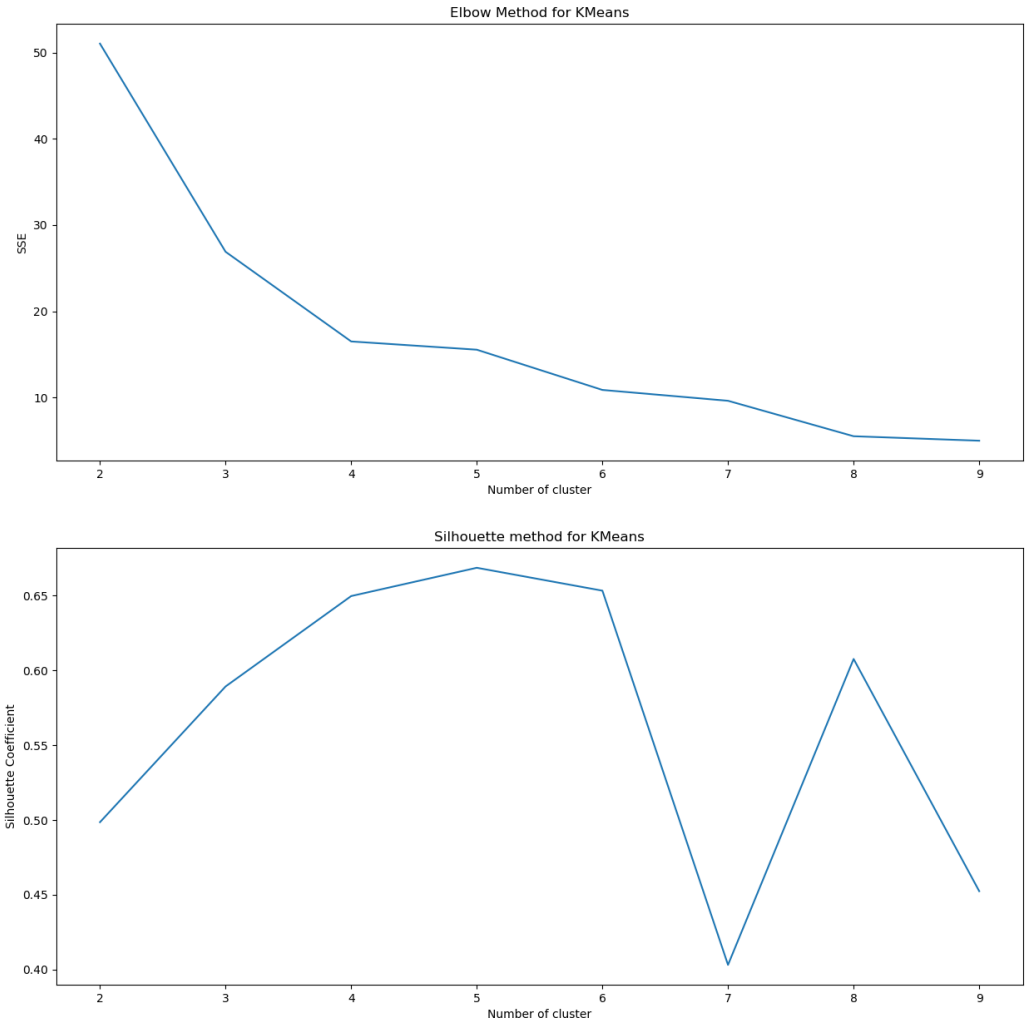

Figure 31: Elbow Method and Silhouette Method Optimization for K-Means Clustering

Based the above methods, the final number of clusters chosen was 5. Using the K-Means algorithm the following 'Location' based clusters were identified:

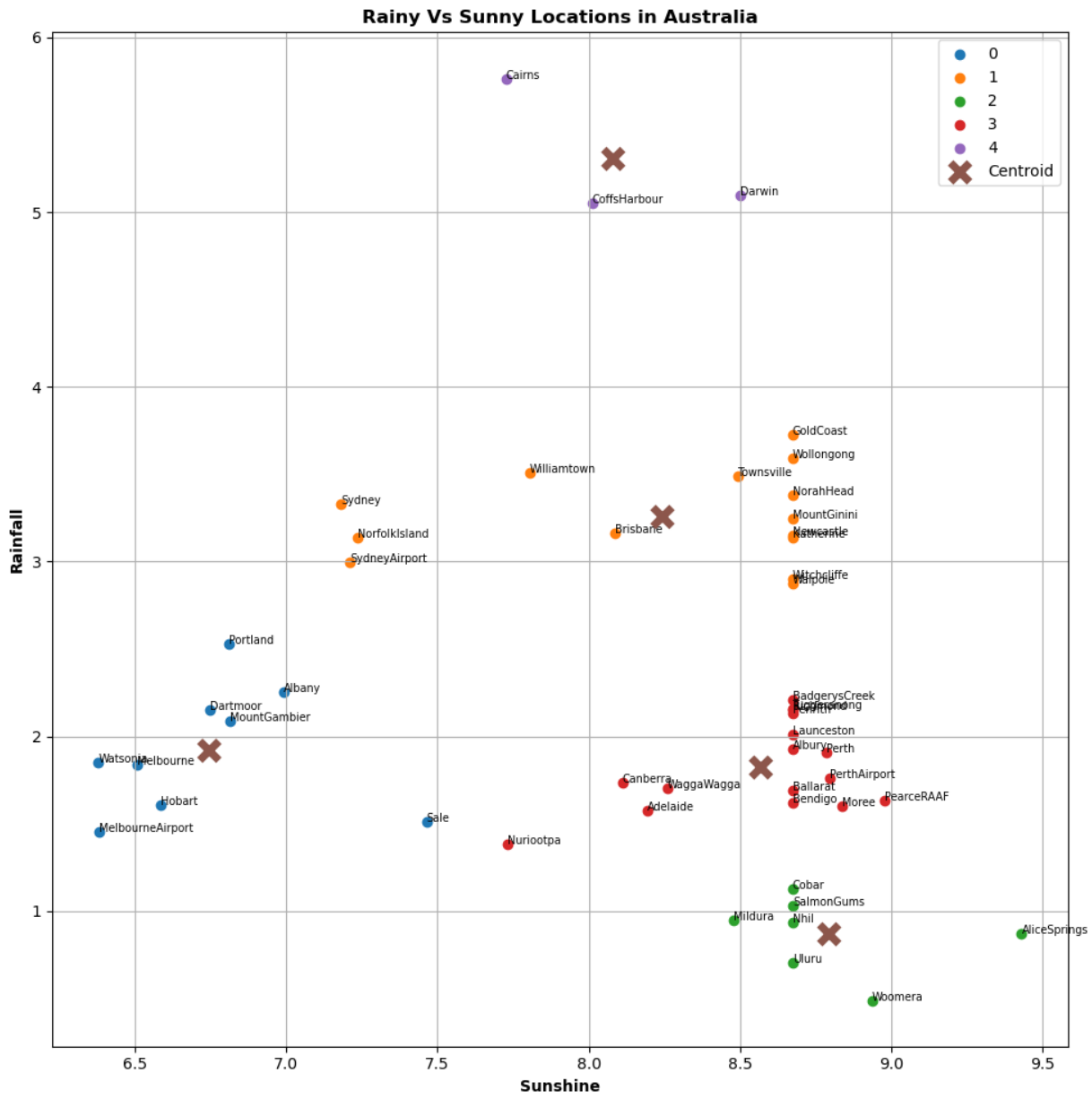

Figure 32: Rainy Vs Sunny Clustering Visualization

Post-clustering, this phase concluded with an exploration into the various associations between the features in the dataset. This was done using the Apriori Algorithm. The dataset was engineered so that all features were binarized based on a quantile cut.

	MinTemp	MaxTemp	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am
0	True	True	False	False	True	True
1	False	True	False	False	True	False
2	True	True	False	False	True	True
3	False	True	False	False	False	False
4	True	True	False	False	True	False

	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am	Pressure3pm	Cloud9am
0	True	True	False	False	False	True
1	True	False	False	False	False	False
2	True	False	False	False	False	False
3	False	False	False	False	False	False
4	True	True	False	False	False	True

	Cloud3pm	Temp9am	Temp3pm	RainToday	RainTomorrow
0	False	True	True	False	False
1	False	True	True	False	False
2	False	True	True	False	False
3	False	True	True	False	False
4	True	True	True	False	False

Figure 33: Association Rule Mining - Feature Engineering

Using the above features, the Apriori algorithm produced the following associations:

Association Rule Mining Table (Apriori Algorithm)

	support	itemsets
11	0.428676	frozenset({'MinTemp', 'Temp9am'})
12	0.427156	frozenset({'MaxTemp', 'Temp9am'})
13	0.473907	frozenset({'MaxTemp', 'Temp3pm'})
14	0.416352	frozenset({'Temp3pm', 'Temp9am'})
15	0.408077	frozenset({'MaxTemp', 'Temp9am', 'Temp3pm'})

Figure 34: Associations for the Dataset

Unfortunately, the algorithm could not identify any associations with a support factor above 0.5. Additionally, the associations that were found could be easily logically concluded.

Recommendations

Recommendations from this report follow a number of lessons learned, conclusions reached, and ideas gained from working with this dataset.

Most important of these is that certain datasets cannot conform to every type of machine learning prediction. In this case, the dataset did not seem to produce great results for regression, or at the least, linear regression. If given more time, non-linear regression would be a worthwhile route to explore for better results.

Notably, as discussed in Phase III, more computationally expensive classifiers do not always produce better results. While Random Forest was chosen as the final model for classification, KNN might be a better fit for scaling due to its lower computational requirement. Naïve Bayes may even be a good consideration if it were more accurate.

Association Rule mining seems to be a useful tool for certain datasets, not so much this one. In the case here, it only pointed out what was obvious.

Overall, the different seemed out of order. Clustering and Association Rule mining would have been more useful for this dataset from the onset, given that they provide more insights *about* the data rather than *from* the data. If done again, this project should go in the order of EDA, Clustering and Association Rule Mining, Classification, and then Regression. Doing classification before regression would better logically follow the questions they ask/the features they are trying to predict.

If further time was given, future work on this project would include: exploring a non-linear model for regression, better feature engineering for more accurate results, better visualizations for classification results, and more exploration into other clusters found in the dataset in order to gain more insights.

References

WORKS CITED

developers, s.-l. (2024). *scikit-learn.linear_model.LogisticRegression*. Retrieved from scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

developers, s.-l. (2024). *sklearn.model_selection.KFold*. Retrieved from scikit-learn: https://scikit-learn.org/dev/modules/generated/sklearn.model_selection.KFold.html

Joe Young, A. Y. (2020). *Rain in Australia*. Retrieved from Kaggle: https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package

NumFOCUS, Inc. (2024). *pandas.DataFrame.shift*. Retrieved from pandas: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.shift.html

Perktold, J., Seabold, S., Taylor, J., & statsmodels-developers. (n.d.). *statsmodels.regression.linear_model.OLS*. Retrieved from statsmodels: https://www.statsmodels.org/dev/generated/statsmodels.regression.linear_model.OLS.html#statsmodels.regression.linear_model.OLS