Project 8 Report:
Australian Rain Prediction

CS5830
Karl Poulson and Jensen Judkins
Slides: Google Slides
Github: Repository

Introduction

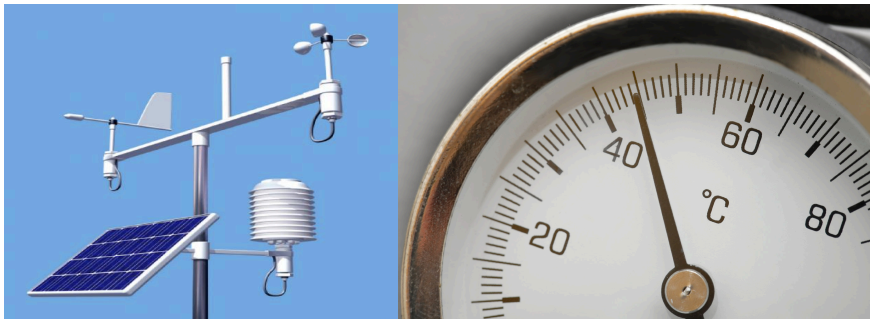
Welcome to our exploration of weather prediction! Our objective is straightforward: predict whether it will rain tomorrow (true/false). Using our data science skills, our aim is to determine if a rainstorm would ruin our beach day plans.

vs

Dataset

This dataset offers a comprehensive look at 10 years of daily weather observations from various locations across Australia, with 2 weather measurements per day. The RainTomorrow variable indicates the likelihood of rain the following day based on whether there was 1 millimeter or more of rain recorded on the current day.

The dataset contained a comprehensive list of weather features, which include: humidity, temperature, evaporation, sunshine, wind gust, wind direction, and altimeter pressure.



Analysis

While the dataset was mostly tidy, there were two significant issues that needed addressing before diving into any machine learning analysis: class imbalance and of wind direction.

Wind Direction Encoding:

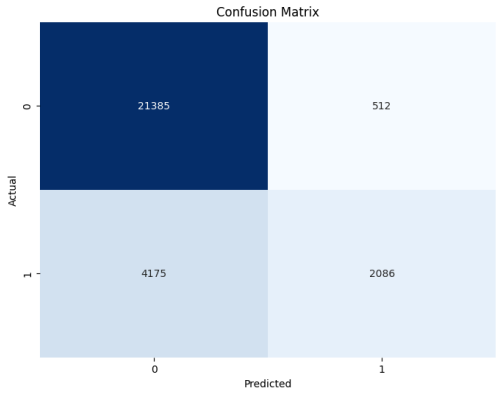
This feature indicated the wind direction in North, West, East, and South directions. Initially, we transformed these directions into angles ranging from 0 to 360 degrees. Since standardizing these values wouldn't be meaningful, so we employed a method known as circle encoding. This approach involves taking the cosine and sine of each angle, which we then standardized to fall between 0 and 1.

Class Imbalance:

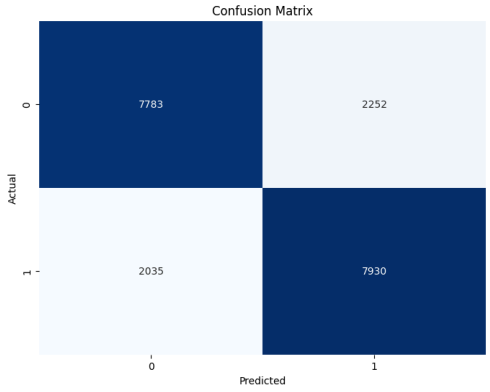
The classes were imbalanced, the feature representing RainTomorrow only composed 20% of the data. We used two techniques to solve this issue:

1. SMOTE algorithm to generate synthetic data for the underrepresented class
2. Randomly Downsampled the overrepresented class.

Confusion matrix before class balance:



Confusion matrix after class balance:

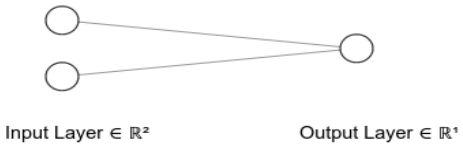


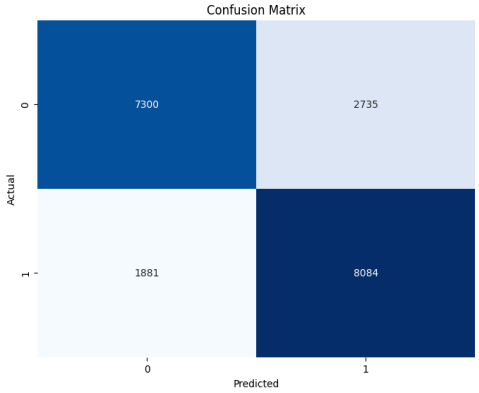
Analysis- Neural Network Architecture

We implemented a Neural Network using keras. Kernel and bias initializers were set to He Normal and 0.01 respectively. Our loss function was binary cross entropy and our last layer used a sigmoid activation function. Our dense layers used ReLU.

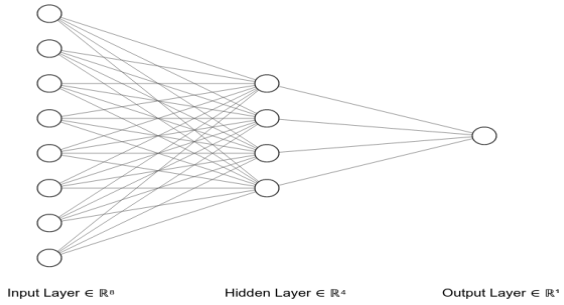
Results

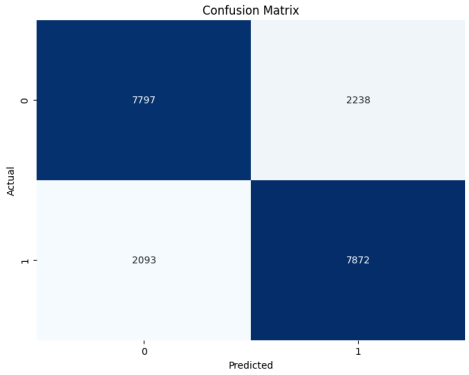
Model Architecture 0: (2, 1):
F1: 71.2, Accuracy: 71.0



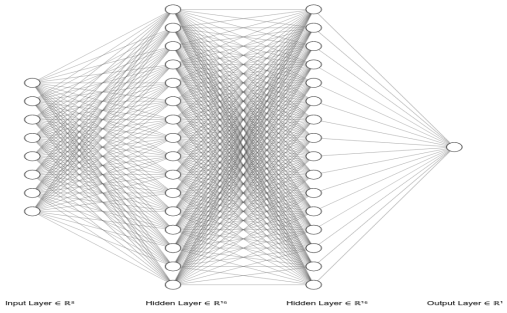


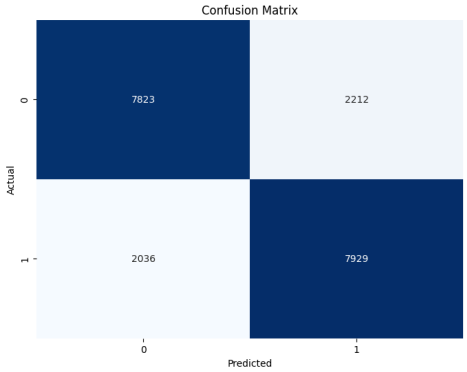
Model Architecture 1: (8, 4, 1).
F1: 78.4, Accuracy: 78.3





Model Architecture 2: (8, 16, 16, 1)
F1: 78.9 Accuracy: 78.8





Analysis - Decision Trees

We implemented a decision tree classification model using sklearn. First we started with all 24 features and a maximum depth of 4. Figuring this would be a good place to start before narrowing down features and depth. These results can be found in “Results - Decision Trees Part 1”.

After reviewing the results, we decided to check which features are providing any important information at all and if we can cut some out. The using the feature

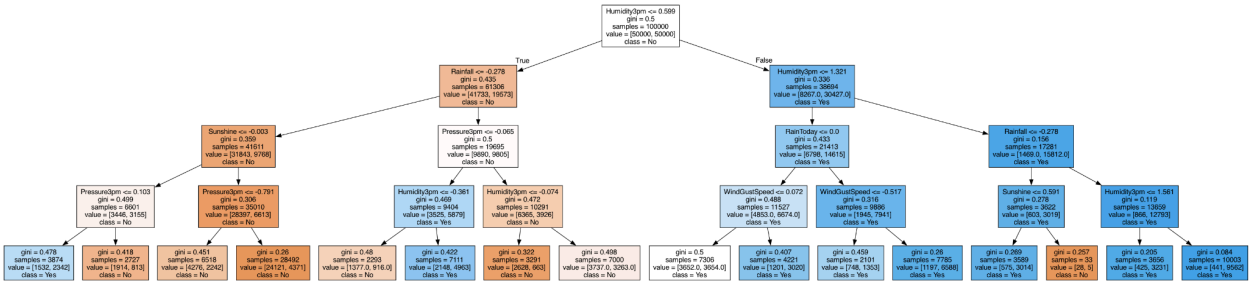
importance function from sklearn, we found that the dataset was extremely top heavy when it came to feature importance. Below are the results...

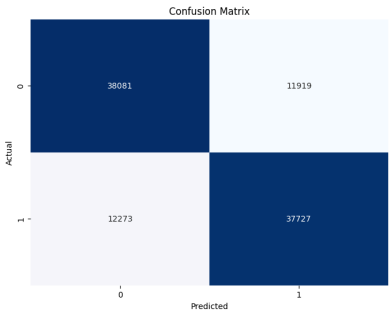
	feature	importance
10	Humidity3pm	0.736384
5	Sunshine	0.106409
3	Rainfall	0.057716
6	WindGustSpeed	0.057516
12	Pressure3pm	0.040193
1	MinTemp	0.001783
15	Temp9am	0.000000

We see that humidity, sunshine, rainfall, and wind gust speed make up over 94% of the importance among the featureset. So following this discovery we decided to make another decision tree using these top 4 features of importance above, and compare and contrast their accuracies.

Results - Decision Trees

First results including all 24 features and a max depth of 4.



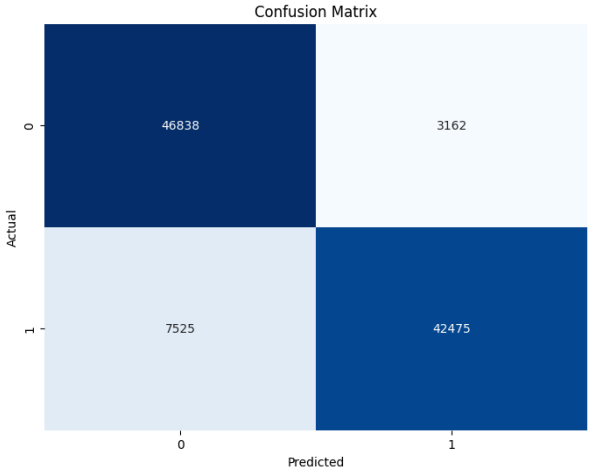


F1 score: 0.7572205607851795
Accuracy: 0.75808
Precision: 0.759920235265681

Following the first test and realizing how top heavy the results were, we decided to cut all features besides humidity, sunshine, rainfall, and wind gust speed in order to simulate resource management when it came to measurement devices and data storage. We however used a higher depth of 7 to see if we could make it more accurate which yielded better results by about 3%. We wanted to really stress the numbers however and tried for a depth of 20. This test resulted in an F1 score of 88.83%, a near 12% increase. But at the cost of its accuracy, it took several times longer (over 7 minutes in comparison to 1.5 seconds

for depth=5) to complete the model. Below is the decision tree along with the confusion matrix and results.

(The decision tree is so spread out it is actually this tiny line above this text^. A link to the photograph can also be found here.)



F1 score: 0.888254545834771
Accuracy: 0.89313
Precision: 0.9307141135482174

Conclusion

In summary, our outcomes were acceptable, achieving approximately 78 percent accuracy with our model. Predictions were determined by rounding, where values above 0.5 were classified as 1 and those below as 0. The most influential feature was whether it rained the previous day or the last measured humidity before the day changed to the next, suggesting that these factors greatly contributed to our model's performance, despite its simplicity.

Future plans involve engineering historical weather data, we hypothesize that past measurements could provide further prediction accuracy.