Data Science CSE558

Predicting Rain:

A Statistical and Machine Learning Approach

Analyzing the weatherAUS Dataset

Pratham Bansal 2023392

Priyanshu Sharma 2023408

Rishabh Dwivedi 2023434

Sidharth Kumar 2023526

Umang Aggarwal 2023567

Vansh Tyagi 2023582

Dataset Description

1. Raw data [Header]

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	...	Humidity9am
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	...	71.0
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	...	44.0
2	2008-12-03	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	...	38.0
3	2008-12-04	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	...	45.0
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	...	82.0

The dataset consists of 10 years of daily weather observations from various locations across Australia.

1. Data points: 145,460
2. *Features: The dataset comprises* 23 columns.

- Numerical columns: MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm, Pressure9am, - Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm.
- Categorical columns: Date, Location, WindGustDir, WindDir9am, WindDir3pm, RainToday.

3. *Label (Target Variable):* The target variable for the model is RainTomorrow

```
raw_data.shape
✓ 0.0s
(145460, 23)
```

2. Data types

```
Data columns (total 23 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           145460 non-null  object
 1   Location       145460 non-null  object
 2   MinTemp        143975 non-null  float64
 3   MaxTemp        144199 non-null  float64
 4   Rainfall       142199 non-null  float64
 5   Evaporation    82670 non-null   float64
 6   Sunshine       75625 non-null   float64
 7   WindGustDir    135134 non-null  object
 8   WindGustSpeed  135197 non-null  float64
 9   WindDir9am     134894 non-null  object
 10  WindDir3pm     141232 non-null  object
 11  WindSpeed9am   143693 non-null  float64
 12  WindSpeed3pm   142398 non-null  float64
 13  Humidity9am    142806 non-null  float64
 14  Humidity3pm    140953 non-null  float64
 15  Pressure9am    130395 non-null  float64
 16  Pressure3pm    130432 non-null  float64
 17  Cloud9am       89572 non-null   float64
 18  Cloud3pm       86102 non-null   float64
 19  Temp9am        143693 non-null  float64
...
 21  RainToday      142199 non-null  object
 22  RainTomorrow   142193 non-null  object
dtypes: float64(16), object(7)
```

3. Statistics

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am
count	143975.000000	144199.000000	142199.000000	82670.000000	75625.000000	135197.000000	143693.000000
mean	12.194034	23.221348	2.360918	5.468232	7.611178	40.035230	14.043426
std	6.398495	7.119049	8.478060	4.193704	3.785483	13.607062	8.915375
min	-8.500000	-4.800000	0.000000	0.000000	0.000000	6.000000	0.000000
25%	7.600000	17.900000	0.000000	2.600000	4.800000	31.000000	7.000000
50%	12.000000	22.600000	0.000000	4.800000	8.400000	39.000000	13.000000
75%	16.900000	28.200000	0.800000	7.400000	10.600000	48.000000	19.000000
max	33.900000	48.100000	371.000000	145.000000	14.500000	135.000000	130.000000

Problem Statement

The objective of this project is to develop a machine learning model that predicts whether it will rain the next day ("RainTomorrow") based on today's weather data. This is framed as a binary classification problem, where the model predicts one of two outcomes: "Yes" (rain expected) or "No" (no rain).

Importance

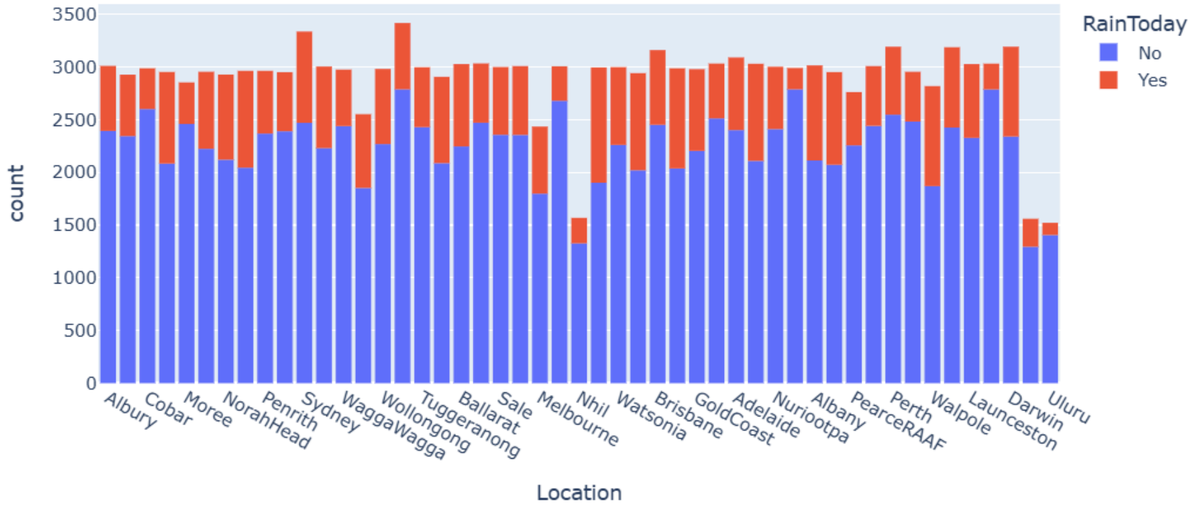
Accurate rainfall prediction is essential across multiple sectors:

1. *Agriculture:* Enables better planning of irrigation, sowing, and harvesting schedules.
2. *Transportation:* Helps anticipate and manage weather-related travel disruptions.
3. *Event Planning & Public Utility:* Assists individuals and organizations in making informed decisions, both daily and logistical.

Exploratory Data Analysis

```
px.histogram(raw_data, x='Location', title='Location vs. Rainy Days',
             color='RainToday')
```

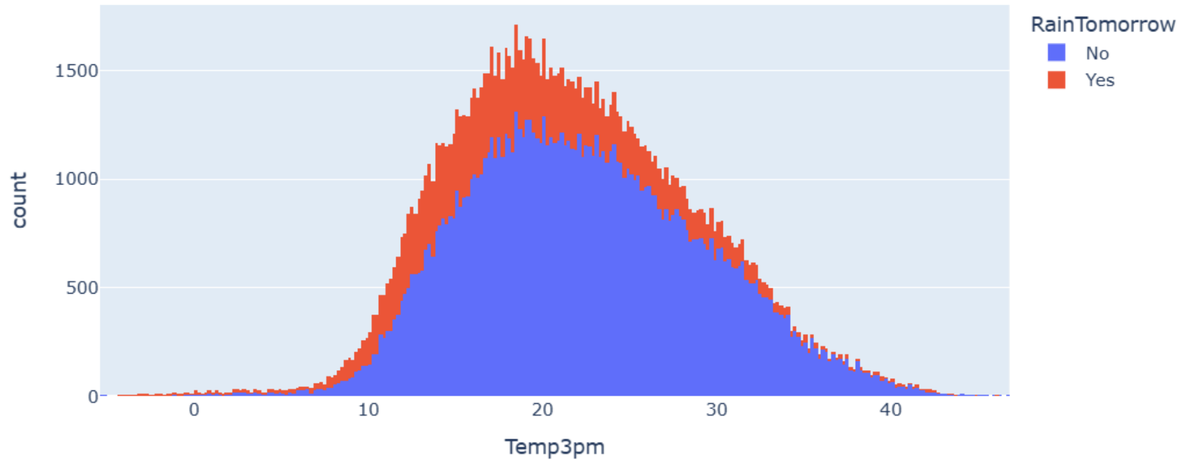
Location vs. Rainy Days



1. The datapoints are mostly uniform with around 3000 data points , with some exceptions such as Nhil
2. Different locations recieves rain on only 15-20% of the days
3. Location plays an important role in predicting

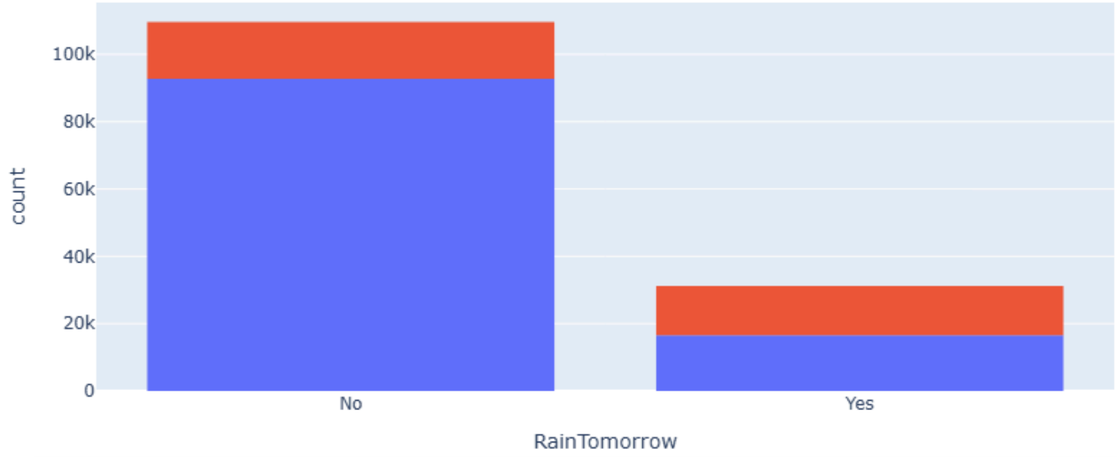
```
px.histogram(raw_data,
             x='Temp3pm',
             title='Temperature at 3 pm vs. Rain Tomorrow',
             color='RainTomorrow')
```

Temperature at 3 pm vs. Rain Tomorrow



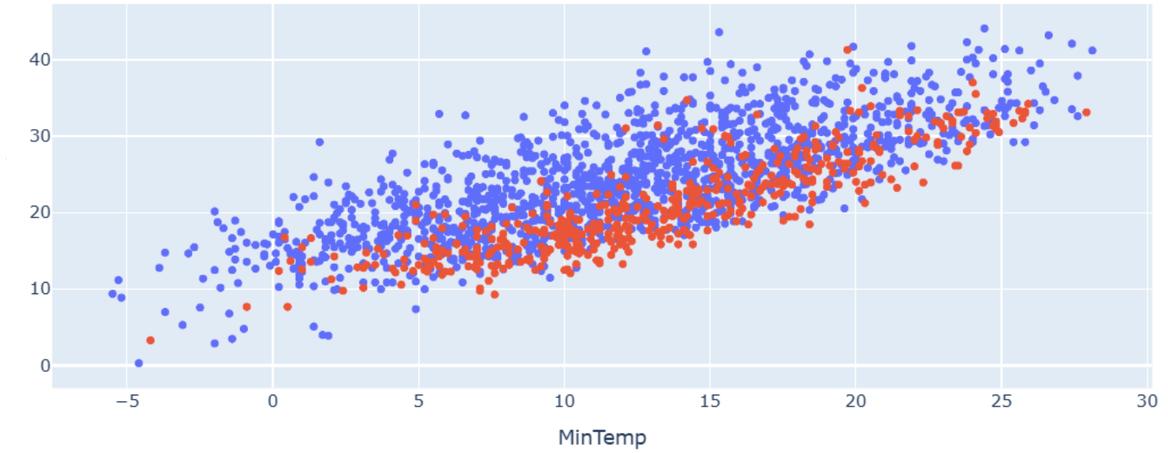
From the graph we can observe that when the temperature is lower , it is more likely to rain tomorrow but when the temp is higher its less likely , but we still see some rain , hence temperature at 3pm is a strong indicator whether it wil rain tomorrow

Rain Tomorrow vs. Rain Today



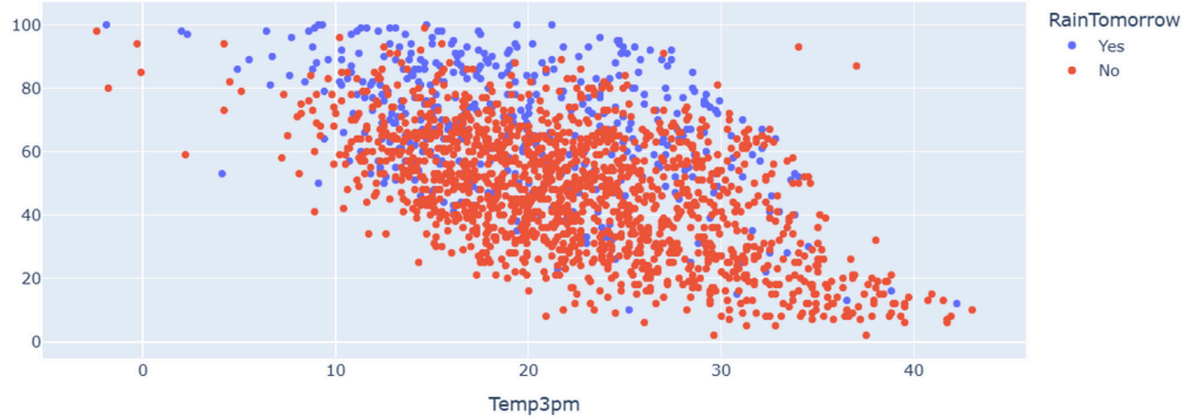
From the above analysis we can say that , if it didnt rain today then there's a pretty good chance that it will not rain tomorrow, but if it did rain today then its equilikely that it may or may not rain tomorrow

Min Temp. vs Max Temp.



The plot shows that on days when it rains (RainToday = 'Yes'), the daily temperature range is small. The red dots are tightly clustered, indicating the minimum and maximum temperatures are close to each other.

Temp (3 pm) vs. Humidity (3 pm)



High humidity and cool temperatures are key indicators for rain.

Data Preprocessing

1. Splitting into Training, Validation, and Test Sets:

```
year = pd.to_datetime(raw_data.Date).dt.year

train_df = raw_data[year < 2015]
val_df = raw_data[year == 2015]
test_df = raw_data[year > 2015]
print('train_df.shape :', train_df.shape)
print('val_df.shape :', val_df.shape)
print('test_df.shape :', test_df.shape)
```

✓ 0.1s

```
train_df.shape : (101018, 23)
val_df.shape : (17885, 23)
test_df.shape : (26557, 23)
```

Why: To prevent data leakage by training only on past data and testing on future data.

Effect: Ensures a realistic evaluation the model predicts future weather based on past patterns.

```
print(raw_data.isnull().sum())
```

✓ 0.0s

```
Date                0
Location            0
MinTemp          1485
MaxTemp          1261
Rainfall         3261
Evaporation     62790
Sunshine        69835
WindGustDir     10326
WindGustSpeed   10263
WindDir9am      10566
WindDir3pm       4228
WindSpeed9am     1767
WindSpeed3pm     3062
Humidity9am      2654
Humidity3pm      4507
Pressure9am     15065
Pressure3pm     15028
Cloud9am        55888
Cloud3pm        59358
Temp9am          1767
Temp3pm          3609
RainToday        3261
RainTomorrow     3267
dtype: int64
```

2. Handling missing values

- Deleting rows

```
raw_data.dropna(subset=['RainToday', 'RainTomorrow'], inplace=True)
raw_data.shape
```

✓ 0.0s

```
(140787, 23)
```

- Imputing

```
from sklearn.impute import SimpleImputer
```

✓ 0.0s

```
imputer = SimpleImputer(strategy = 'mean')
imputer.fit(raw_data[numeric_cols])
train_inputs[numeric_cols] = imputer.transform(train_inputs[numeric_cols])
val_inputs[numeric_cols] = imputer.transform(val_inputs[numeric_cols])
test_inputs[numeric_cols] = imputer.transform(test_inputs[numeric_cols])
```

✓ 0.1s

```
MinTemp          0
MaxTemp          0
Rainfall         0
Evaporation      0
Sunshine         0
WindGustSpeed    0
WindSpeed9am     0
WindSpeed3pm     0
Humidity9am      0
Humidity3pm      0
Pressure9am      0
Pressure3pm      0
Cloud9am         0
Cloud3pm         0
Temp9am          0
Temp3pm          0
```

Why: To ensure the dataset is complete and suitable for model training.

Effect: Deleting reduces data size but removes errors; imputing keeps all data but may introduce minor bias.

3. Scaling Numeric Features

```
raw_data[numeric_cols].describe()
```
✓ 0.2s

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am
count	140319.000000	140480.00000	140787.000000	81093.000000	73982.000000	131682.000000	139732.000000
mean	12.184824	23.23512	2.349974	5.472516	7.630540	39.970520	13.990496
std	6.403879	7.11450	8.465173	4.189132	3.781729	13.578201	8.886210
min	-8.500000	-4.80000	0.000000	0.000000	0.000000	6.000000	0.000000
25%	7.600000	17.90000	0.000000	2.600000	4.900000	31.000000	7.000000
50%	12.000000	22.60000	0.000000	4.800000	8.500000	39.000000	13.000000
75%	16.800000	28.30000	0.800000	7.400000	10.700000	48.000000	19.000000
max	33.900000	48.10000	371.000000	145.000000	14.500000	135.000000	130.000000

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
scaler.fit(raw_data[numeric_cols])
```

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am
count	101018.000000	101018.000000	101018.000000	101018.000000	101018.000000	101018.000000	101018.000000
mean	0.483341	0.525191	0.006418	0.036929	0.524770	0.265454	0.108661
std	0.148901	0.131864	0.022729	0.021438	0.199800	0.102340	0.068795
min	0.000000	0.013233	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.377358	0.429112	0.000000	0.026207	0.517241	0.193798	0.053846
50%	0.478774	0.514178	0.000000	0.037741	0.526244	0.255814	0.100000
75%	0.589623	0.616257	0.002695	0.037741	0.634483	0.310078	0.146154
max	1.000000	1.000000	1.000000	0.568276	0.986207	1.000000	0.669231

Why: To ensure that features with large ranges (e.g., Pressure9am) do not disproportionately influence the model's learning process compared to features with small ranges (e.g., Rainfall).

Effect: Scaling numeric features makes all values comparable in range, improving model stability, faster convergence, and preventing features with larger scales from dominating the learning process.

4. Encoding categorical columns

```
raw_data[categorical_cols].nunique()
```
✓ 0.0s

```
Location       49
WindGustDir    16
WindDir9am     16
WindDir3pm     16
RainToday       2
dtype: int64
```

```
from sklearn.preprocessing import OneHotEncoder
encoder = OneHotEncoder(sparse_output=False, handle_unknown='ignore')
encoder.fit(raw_data[categorical_cols])
encoded_cols = list(encoder.get_feature_names_out(categorical_cols))
train_inputs[encoded_cols] = encoder.transform(train_inputs[categorical_cols])
val_inputs[encoded_cols] = encoder.transform(val_inputs[categorical_cols])
test_inputs[encoded_cols] = encoder.transform(test_inputs[categorical_cols])
```

WindDir3pm_SE	WindDir3pm_SSE	WindDir3pm_SSW	WindDir3pm_SW	WindDir3pm_W	WindDir3pm_WNW	WindDir3pm_WSW	WindDir3pm_nan
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

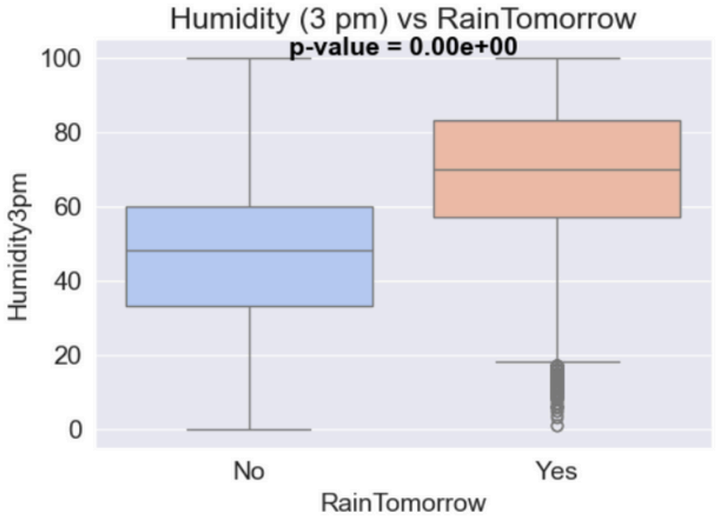
Why: Machine learning models can only work with numerical data. Categorical (text) data like 'Location' or 'WindDir9am' must be converted into numbers.

Effect: Encoding categorical columns converts text labels into numeric form, allowing machine learning models to interpret and learn from categorical data effectively.

Hypothesis Testing

HYPOTHESIS-1 : Humidity3pm vs RainTomorrow

- H_0: The mean Humidity3pm is the same for days it rains tomorrow and days it doesn't.
- H_a: The mean Humidity3pm is different

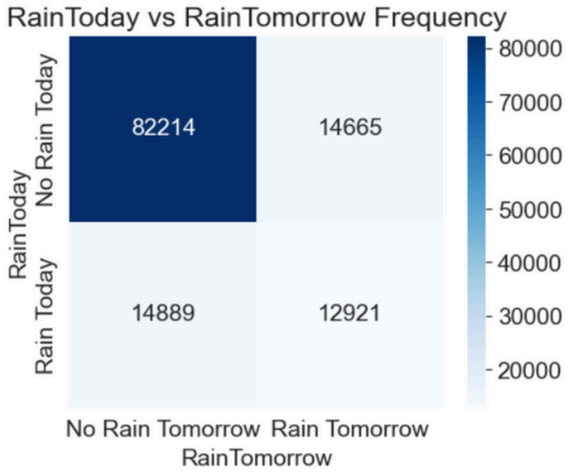



```
t-stat = 176.1791, p-value = 0.0000e+00
Decision: Reject H₀
```

I rejected the null hypothesis (p-value was extremely small). The boxplot clearly shows that the median humidity at 3 pm is significantly higher on days that are followed by rain. This strongly suggests Humidity3pm is a key predictor.

HYPOTHESIS-2: RainToday vs RainTomorrow (Proportion Test & Chi-square)

- H_0: The proportion of rain tomorrow is the same whether it rained today or not. (They are independent).
- H_a: The proportion is different. (They are dependent).

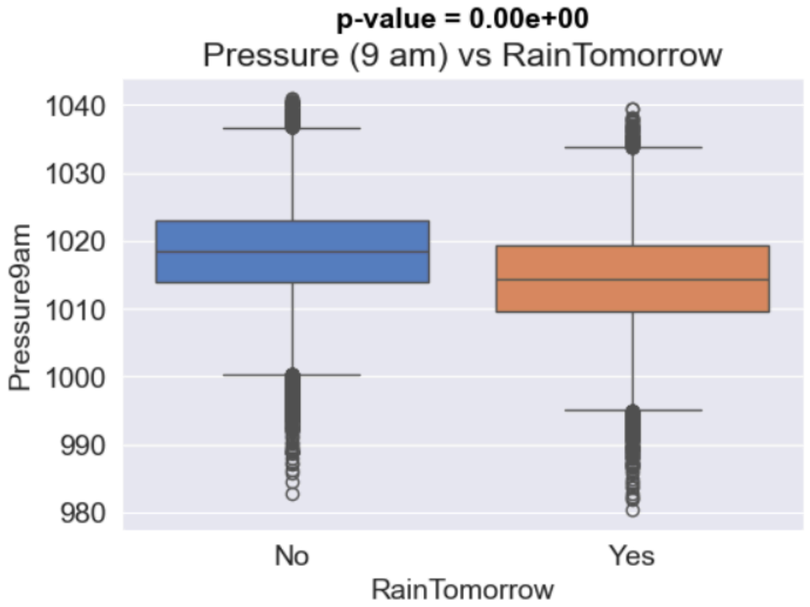



```
Z-test:
Z-stat = -110.9301, p-value = 0.0000e+00
Decision: Reject H₀

Chi-square test:
Chi-square = 12303.6794, p-value = 0.0000e+00, dof = 1
Decision: Reject H₀
```

I rejected the null hypothesis with both the z-test and the chi-square test. The p-values were near zero, indicating a strong dependency between rain today and rain tomorrow. The heatmap shows that if it rains today, it's far more likely to rain tomorrow (and vice-versa).

HYPOTHESIS-3: Pressure9am vs RainTomorrow

- H_0: The mean Pressure9am is the same for days it rains tomorrow and days it doesn't.
- H_a: The mean Pressure9am is different.

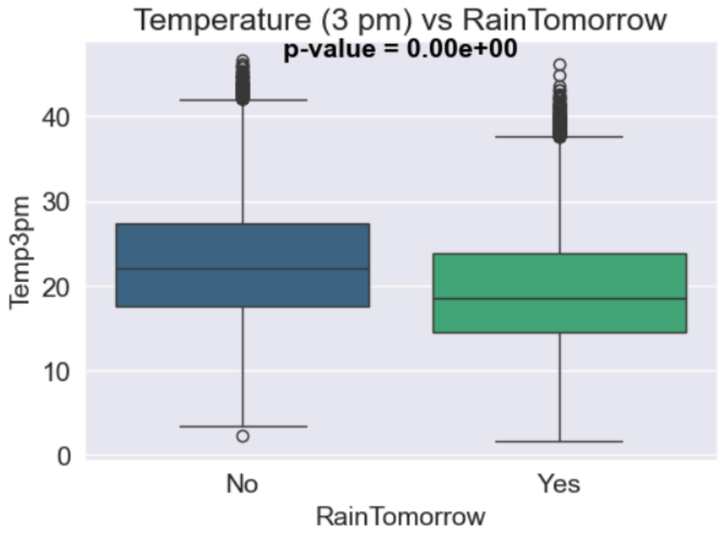



```
t-stat = -86.4506, p-value = 0.0000e+00
Decision: Reject H₀
```

I rejected the null hypothesis. The t-statistic was negative, and the boxplot shows that the average Pressure9am is significantly lower on days preceding rain. This aligns with meteorological principles.

HYPOTHESIS-4: Temp3pm vs RainTomorrow

- H_0: The mean Temp3pm is the same for days it rains tomorrow and days it doesn't.
- H_a: The mean Temp3pm is different.

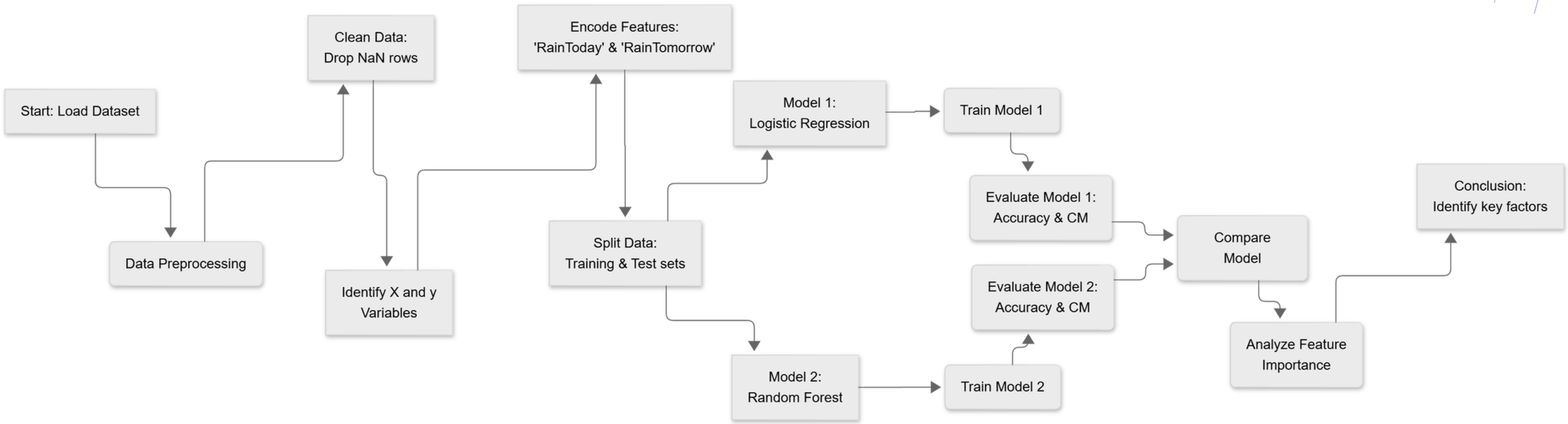



```
t-stat = -71.8972, p-value = 0.0000e+00
Decision: Reject H₀
```

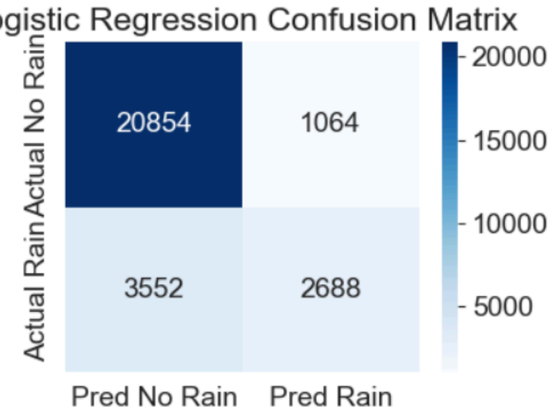
I rejected the null hypothesis. The negative t-statistic indicates that the average Temp3pm is significantly lower on days before it rains.

Conclusion: All four tested variables (Humidity3pm, RainToday, Pressure9am, and Temp3pm) show a statistically significant relationship with RainTomorrow. This validates my initial observations from the EDA and confirms they are excellent candidates for inclusion in my predictive model.

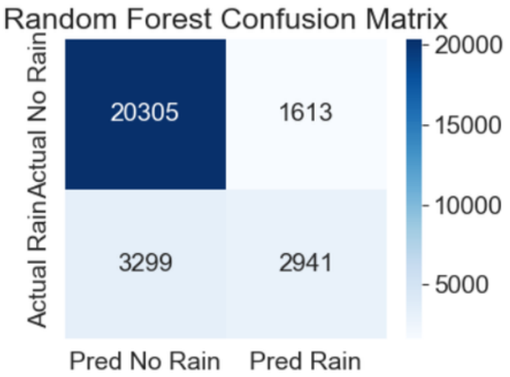
Experiments




```
--- Logistic Regression ---
Classification Report:
              precision    recall  f1-score   support

     No Rain     0.8545    0.9515    0.9004     21918
        Rain     0.7164    0.4308    0.5380      6240

    accuracy                         0.8361     28158
   macro avg     0.7854    0.6911    0.7192     28158
weighted avg     0.8239    0.8361    0.8201     28158
```




```
--- Random Forest ---
Classification Report:
              precision    recall  f1-score   support

     No Rain     0.8602    0.9264    0.8921     21918
        Rain     0.6458    0.4713    0.5449      6240

    accuracy                         0.8256     28158
   macro avg     0.7530    0.6989    0.7185     28158
weighted avg     0.8127    0.8256    0.8152     28158
```

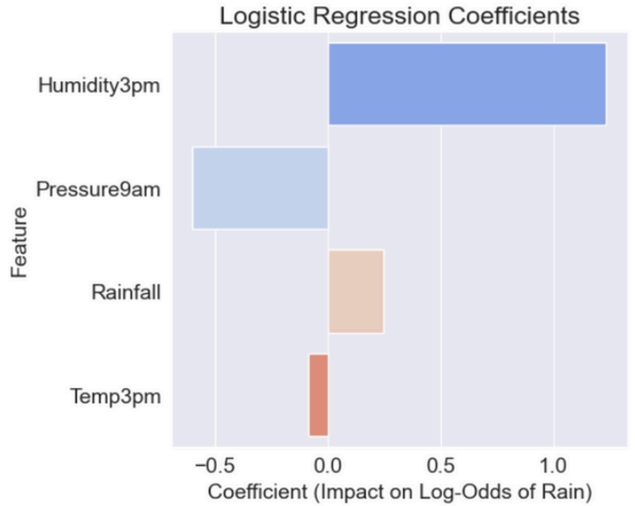


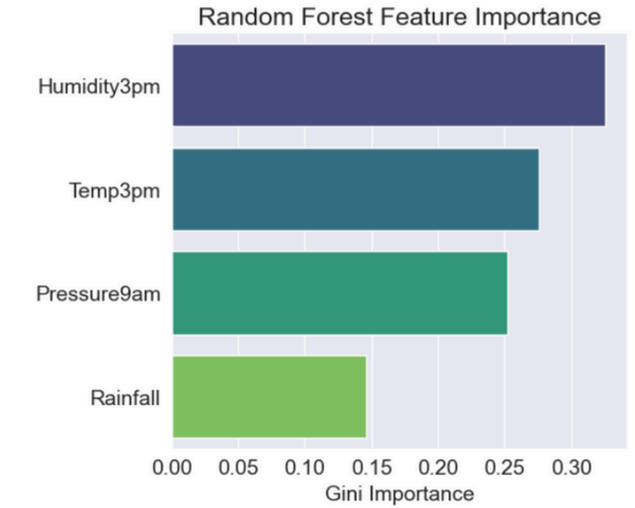
Conclusion

The machine learning experiments successfully validated the findings from my hypothesis tests.

1. **Model Performance**: Both models performed well, achieving ROC AUC scores of over 0.83 (my actual scores were 0.8308 for LogReg and 0.8114 for RandomForest). This is a strong result, especially since I only used the four features I tested. This confirms that these features have significant predictive power.

2. **Feature Importance**: The feature importance and coefficient plots confirmed which variables were most important and validated my hypothesis test results.For both models, Humidity3pm was the most dominant predictor by a large margin. This directly supports the hypothesis test (Test 1), which had an extremely small p-value.The Logistic Regression coefficients were particularly insightful:Humidity3pm and Rainfall have positive coefficients, meaning as they increase, the probability of rain increases.Pressure9am and Temp3pm have negative coefficients, meaning as they increase, the probability of rain decreases. This perfectly matches the findings from my hypothesis tests (Tests 3 and 4).





The statistical tests (Part 5) correctly identified key predictive features. The machine learning models (Part 6) confirmed these findings in a practical experiment, showing that these four variables alone can build a robust model for predicting rain. The statistical significance translated directly into predictive power.

Thank you