Department of Computer Science
American International University-Bangladesh

Course Name: DATA WAREHOUSING AND DATA MINING

"Project on Classification of Weather Condition of Australia using NAÏVE BAYES ALGORITHM"

Section: B

Supervised By:

Dr. Akinul Islam Jony

Associate Professor & Head-In-Charge [Undergraduate Program], Computer Science

Submitted By:

Md. Abdullah Al Nasir (19-41052-2)

Nazmus Sakib (20-44113-2)

Tonoy Chandra Sarker (20-43804-2)

Submission Date: November 16, 2023.

Dataset Link: https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package

1. Introduction:

This project revolves around a decade-spanning Australian weather dataset, capturing comprehensive meteorological attributes. With a focus on predicting next-day rain, the binary classification task centers on the "Rain Tomorrow" variable, signaling precipitation of 1mm or more. Leveraging Python, the code explores, cleans, and transforms the dataset for robust modeling. It addresses missing values, maps categorical variables, and splits data for training and testing. A Naive Bayes classifier is implemented for rain prediction, offering valuable insights into Australian weather patterns. This versatile dataset encourages research, analysis, and machine learning applications, fostering a deeper understanding of Australia's dynamic weather trends.

2.Dataset description:

The dataset encompasses a decade of daily weather records from diverse locations throughout Australia. Comprising 23 attributes, it offers comprehensive insights into meteorological conditions, including rainfall, temperature, humidity, pressure, and wind speed. The primary focus lies in predicting next-day rain, with the target variable "Rain Tomorrow" indicating whether precipitation of 1mm or more occurred. This binary classification task, answering Yes or No to the question of rain the following day, forms the crux of potential machine learning applications. Researchers and data enthusiasts can harness this rich dataset to develop predictive models, enhancing their ability to forecast rain patterns. With its user-friendly interface and wealth of information, the dataset serves as a valuable resource for those keen on exploring and understanding the intricacies of Australian weather patterns over the past decade.

Details:

- **The dataset comprises 61,893 instances.**
- **It consists of 23 attributes.**
- **There are missing values in the dataset, as identified and addressed in the code.**
- **The class values are binary, representing "Rain Tomorrow" with precipitation of 1mm or more as either Yes (1) or No (0).**

- **The attributes include meteorological parameters such as rainfall, temperature (MinTemp and MaxTemp), humidity, pressure, wind speed (WindSpeed9am and WindSpeed3pm), and cloud cover (Cloud9am and Cloud3pm), among others.**

The Real Dataset:

Date	Location	MinTemp	MaxTemp	Rainfall	Evaporatic	Sunshine	WindGust[	WindGustS	WindDir9a	WindDir3p	WindSpee	WindSpee	Humidity9	Humidity3	Pressure9a	Pressure3p	Cloud9am	Cloud3pm	Temp9am	Temp3pm	RainToday	RainTomorro
########	Albury	13.4	22.9	0.6	NA	NA	W	44	W	WNW	20	24	71	22	1007.7	1007.1	8	NA	16.9	21.8	No	No
########	Albury	7.4	25.1	0	NA	NA	WNW	44	NNW	WSW	4	22	44	25	1010.6	1007.8	NA	NA	17.2	24.3	No	No
########	Albury	12.9	25.7	0	NA	NA	WSW	46	W	WSW	19	26	38	30	1007.6	1008.7	NA	2	21	23.2	No	No
########	Albury	9.2	28	0	NA	NA	NE	24	SE	E	11	9	45	16	1017.6	1012.8	NA	NA	18.1	26.5	No	No
########	Albury	17.5	32.3	1	NA	NA	W	41	ENE	NW	7	20	82	33	1010.8	1006	7	8	17.8	29.7	No	No
########	Albury	14.6	29.7	0.2	NA	NA	WNW	56	W	W	19	24	55	23	1009.2	1005.4	NA	NA	20.6	28.9	No	No
########	Albury	14.3	25	0	NA	NA	W	50	SW	W	20	24	49	19	1009.6	1008.2	1	NA	18.1	24.6	No	No
########	Albury	7.7	26.7	0	NA	NA	W	35	SSE	W	6	17	48	19	1013.4	1010.1	NA	NA	16.3	25.5	No	No
########	Albury	9.7	31.9	0	NA	NA	NNW	80	SE	NW	7	28	42	9	1008.9	1003.6	NA	NA	18.3	30.2	No	Yes
########	Albury	13.1	30.1	1.4	NA	NA	W	28	S	SSE	15	11	58	27	1007	1005.7	NA	NA	20.1	28.2	Yes	No
########	Albury	13.4	30.4	0	NA	NA	N	30	SSE	ESE	17	6	48	22	1011.8	1008.7	NA	NA	20.4	28.8	No	Yes
########	Albury	15.9	21.7	2.2	NA	NA	NNE	31	NE	ENE	15	13	89	91	1010.5	1004.2	8	8	15.9	17	Yes	Yes
########	Albury	15.9	18.6	15.6	NA	NA	W	61	NNW	NNW	28	28	76	93	994.3	993	8	8	17.4	15.8	Yes	Yes
########	Albury	12.6	21	3.6	NA	NA	SW	44	W	SSW	24	20	65	43	1001.2	1001.8	NA	7	15.8	19.8	Yes	No
########	Albury	8.4	24.6	0	NA	NA	NA	NA	S	WNW	4	30	57	32	1009.7	1008.7	NA	NA	15.9	23.5	No	NA
########	Albury	9.8	27.7	NA	NA	NA	WNW	50	NA	WNW	NA	22	50	28	1013.4	1010.3	0	NA	17.3	26.2	NA	No
########	Albury	14.1	20.9	0	NA	NA	ENE	22	SSW	E	11	9	69	82	1012.2	1010.4	8	1	17.2	18.1	No	Yes
########	Albury	13.5	22.9	16.8	NA	NA	W	63	N	WNW	6	20	80	65	1005.8	1002.2	8	1	18	21.5	Yes	Yes
########	Albury	11.2	22.5	10.6	NA	NA	SSE	43	WSW	SW	24	17	47	32	1009.4	1009.7	NA	2	15.5	21	Yes	No
########	Albury	9.8	25.6	0	NA	NA	SSE	26	SE	NNW	17	6	45	26	1019.2	1017.1	NA	NA	15.8	23.2	No	No
########	Albury	11.5	29.3	0	NA	NA	S	24	SE	SE	9	9	56	28	1019.3	1014.8	NA	NA	19.1	27.3	No	No
########	Albury	17.1	33	0	NA	NA	NE	43	NE	N	17	22	38	28	1013.6	1008.1	NA	1	24.5	31.6	No	No
########	Albury	20.5	31.8	0	NA	NA	WNW	41	W	W	19	20	54	24	1007.8	1005.7	NA	NA	23.8	30.8	No	No
########	Albury	15.3	30.9	0	NA	NA	N	33	ESE	NW	6	13	55	23	1011	1008.2	5	NA	20.9	29	No	No
########	Albury	12.6	32.4	0	NA	NA	W	43	E	W	4	19	49	17	1012.9	1010.1	NA	NA	21.5	31.2	No	No
########	Albury	16.2	33.9	0	NA	NA	WSW	35	SE	WSW	9	13	45	19	1010.9	1007.6	NA	1	23.2	33	No	No

Figure: The CSV file of the dataset of weather condition of Australia.

3.

```
import pandas as pd
data = pd.read_csv("E:/weatherAUS.csv")
data
```

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	...	Humidity9am	Humidity3pm	Pres
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	...	71.0	22.0	
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	...	44.0	25.0	
2	2008-12-03	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	...	38.0	30.0	
3	2008-12-04	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	...	45.0	16.0	
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	...	82.0	33.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
145455	2017-06-21	Uluru	2.8	23.4	0.0	NaN	NaN	E	31.0	SE	...	51.0	24.0	
145456	2017-06-22	Uluru	3.6	25.3	0.0	NaN	NaN	NNW	22.0	SE	...	56.0	21.0	
145457	2017-06-23	Uluru	5.4	26.9	0.0	NaN	NaN	N	37.0	SE	...	53.0	24.0	
145458	2017-06-24	Uluru	7.8	27.0	0.0	NaN	NaN	SE	28.0	SSE	...	51.0	24.0	
145459	2017-06-25	Uluru	14.9	NaN	0.0	NaN	NaN	NaN	NaN	ESE	...	62.0	36.0	

Short description:
This Python code loads an Australian weather dataset from "weatherAUS.csv," enabling data exploration and analysis for a concise and insightful project.

4.

```
data.drop(['Date','Location','Pressure9am', 'Pressure3pm', 'WindGustDir', 'WindDir9am', 'WindDir3pm', 'WindGustSpeed', 'Temp9am',
```

5.

```
data
```

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Cloud9am	Cloud3pm	RainToday	R
0	13.4	22.9	0.6	NaN	NaN	20.0	24.0	71.0	22.0	8.0	NaN	No	
1	7.4	25.1	0.0	NaN	NaN	4.0	22.0	44.0	25.0	NaN	NaN	No	
2	12.9	25.7	0.0	NaN	NaN	19.0	26.0	38.0	30.0	NaN	2.0	No	
3	9.2	28.0	0.0	NaN	NaN	11.0	9.0	45.0	16.0	NaN	NaN	No	
4	17.5	32.3	1.0	NaN	NaN	7.0	20.0	82.0	33.0	7.0	8.0	No	
...	...	...	...	...	...	...	...	...	...	...	...	...	
145455	2.8	23.4	0.0	NaN	NaN	13.0	11.0	51.0	24.0	NaN	NaN	No	
145456	3.6	25.3	0.0	NaN	NaN	13.0	9.0	56.0	21.0	NaN	NaN	No	
145457	5.4	26.9	0.0	NaN	NaN	9.0	9.0	53.0	24.0	NaN	NaN	No	
145458	7.8	27.0	0.0	NaN	NaN	13.0	7.0	51.0	24.0	3.0	2.0	No	
145459	14.9	NaN	0.0	NaN	NaN	17.0	17.0	62.0	36.0	8.0	8.0	No	

145460 rows × 13 columns

Short description:
This code loads an Australian weather dataset, featuring meteorological parameters. Potential project goals include analyzing patterns, predicting rain, and deriving insights for informed decision-making in various applications.

6.

```
data.columns[data.isna().any()]
```

```
Index(['MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine',
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
       'Cloud9am', 'Cloud3pm', 'RainToday', 'RainTomorrow'],
      dtype='object')
```

Short description:
This code identifies columns with missing values in an Australian weather dataset, informing data cleaning efforts for a comprehensive analysis and predictive modeling in this project.

7.

```
data.dropna(subset=['MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine',
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
       'Cloud9am', 'Cloud3pm', 'RainToday', 'RainTomorrow'], inplace=True)
print(data)
```

```
print(data)
```

```
        MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine  WindSpeed9am  \
6049       17.9     35.2       0.0         12.0      12.3           6.0
6050       18.4     28.9       0.0         14.8      13.0          19.0
6052       19.4     37.6       0.0         10.8      10.6          30.0
6053       21.9     38.4       0.0         11.4      12.2           6.0
6054       24.2     41.0       0.0         11.2       8.4          17.0
...         ...      ...       ...          ...       ...           ...
142298     19.3     33.4       0.0          6.0      11.0           9.0
142299     21.2     32.6       0.0          7.6       8.6          13.0
142300     20.7     32.8       0.0          5.6      11.0          17.0
142301     19.5     31.8       0.0          6.2      10.6           9.0
142302     20.2     31.7       0.0          5.6      10.7          15.0

        WindSpeed3pm  Humidity9am  Humidity3pm  Cloud9am  Cloud3pm RainToday  \
6049            20.0         20.0         13.0       2.0       5.0        No
6050            19.0         30.0          8.0       1.0       1.0        No
6052            15.0         42.0         22.0       1.0       6.0        No
6053             6.0         37.0         22.0       1.0       5.0        No
6054            13.0         19.0         15.0       1.0       6.0        No
...              ...          ...          ...       ...       ...       ...
142298          20.0         63.0         32.0       0.0       1.0        No
142299          11.0         56.0         28.0       7.0       0.0        No
142300          11.0         46.0         23.0       0.0       0.0        No
142301          17.0         62.0         58.0       1.0       1.0        No
142302           7.0         73.0         32.0       6.0       5.0        No
```

Short description:
This code drops rows with missing values in essential weather parameters, producing a refined Australian weather dataset with 61,893 entries. This preprocessing step is crucial for robust analysis and modeling in a focused in this project, enhancing the dataset's quality and reliability.

8.

```
inputs = data.drop(['RainToday','RainTomorrow'], axis='columns')
inputs
```

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Cloud9am	Cloud3pm
6049	17.9	35.2	0.0	12.0	12.3	6.0	20.0	20.0	13.0	2.0	5.0
6050	18.4	28.9	0.0	14.8	13.0	19.0	19.0	30.0	8.0	1.0	1.0
6052	19.4	37.6	0.0	10.8	10.6	30.0	15.0	42.0	22.0	1.0	6.0
6053	21.9	38.4	0.0	11.4	12.2	6.0	6.0	37.0	22.0	1.0	5.0
6054	24.2	41.0	0.0	11.2	8.4	17.0	13.0	19.0	15.0	1.0	6.0
...	...	...	...	...	...	...	...	...	...	...	...
142298	19.3	33.4	0.0	6.0	11.0	9.0	20.0	63.0	32.0	0.0	1.0
142299	21.2	32.6	0.0	7.6	8.6	13.0	11.0	56.0	28.0	7.0	0.0
142300	20.7	32.8	0.0	5.6	11.0	17.0	11.0	46.0	23.0	0.0	0.0
142301	19.5	31.8	0.0	6.2	10.6	9.0	17.0	62.0	58.0	1.0	1.0
142302	20.2	31.7	0.0	5.6	10.7	15.0	7.0	73.0	32.0	6.0	5.0

61893 rows × 11 columns

Short description:
This code creates a new Data Frame, "inputs," by excluding the target variables ('Rain Today' and 'Rain Tomorrow'). The resulting dataset contains 61,893 rows and 11 essential weather features, setting the stage for feature analysis in a focused-on project.

9.

```
target = data[['RainToday', 'RainTomorrow']]
```

10.

```
: target
```

]:

	RainToday	RainTomorrow
6049	No	No
6050	No	No
6052	No	No
6053	No	No
6054	No	No
...	...	...
142298	No	No
142299	No	No
142300	No	No
142301	No	No
142302	No	No

61893 rows × 2 columns

Short description:
This code creates a target Data Frame, "target," containing the 'Rain Today' and 'Rain Tomorrow' columns from the Australian weather dataset. These variables will be utilized for classification in a predictive modeling project, contributing to informed decision-making based on rain predictions.
11.

```
: target['RainToday'] = target['RainToday'].map({'No': 0, 'Yes': 1})
target['RainTomorrow'] = target['RainTomorrow'].map({'No': 0, 'Yes': 1})
```

```
C:\Users\NAZMUS SAKIB\AppData\Local\Temp\ipykernel_18884\1921879921.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
rsus-a-copy
  target['RainToday'] = target['RainToday'].map({'No': 0, 'Yes': 1})
C:\Users\NAZMUS SAKIB\AppData\Local\Temp\ipykernel_18884\1921879921.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
rsus-a-copy
  target['RainTomorrow'] = target['RainTomorrow'].map({'No': 0, 'Yes': 1})
```

Short description:
This code maps categorical values in the 'Rain Today' and 'Rain Tomorrow' columns to numerical equivalents (0 and 1). While it achieves the desired transformation, it generates SettingWithCopy Warnings, suggesting a potential improvement in Data Frame indexing for enhanced clarity and efficiency in this project.

12.

```
target
```

]:

	RainToday	RainTomorrow
6049	0	0
6050	0	0
6052	0	0
6053	0	0
6054	0	0
...	...	...
142298	0	0
142299	0	0
142300	0	0
142301	0	0
142302	0	0

61893 rows × 2 columns

Short description:
This code successfully transforms the categorical values in the 'RainToday' and 'RainTomorrow' columns into numerical equivalents (0 and 1), enabling efficient classification for predictive modeling in a data science project. The resulting 'target' DataFrame is now ready for analysis.

13.

```
2]: from sklearn.model_selection import train_test_split

# Split the data into training and testing sets with a test size of 30%
x_train, x_test, y_train, y_test = train_test_split(inputs, target, test_size=0.3)

# Access the x_test variable to see the resulting test set
print(x_test)
print(x_train)
```

```
        MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine  WindSpeed9am  \
118415     21.3     37.6       0.0          8.8      11.9          11.0
104502      4.5     10.3      10.0          2.9       8.4          28.0
46076      16.3     22.8       0.0          4.6       9.3          30.0
36851      21.4     40.8       0.0         11.2       4.9          11.0
67235       5.6     15.0       0.2          2.0       7.7           6.0
...         ...      ...       ...          ...       ...           ...
32031      11.3     21.1       0.2          2.0       4.9          15.0
80097      18.2     38.5       0.0          8.0      11.8           7.0
105746     10.0     15.2       0.0          3.4       2.0          15.0
88052      19.6     29.9       0.0          6.4       9.9          15.0
141415     25.4     33.0       0.0          7.0      10.7           9.0

        WindSpeed3pm  Humidity9am  Humidity3pm  Cloud9am  Cloud3pm
118415          22.0         54.0         41.0       0.0       1.0
104502          30.0         67.0         58.0       4.0       6.0
46076           28.0         43.0         14.0       6.0       1.0
36851           35.0         19.0          7.0       7.0       5.0
67235           20.0         81.0         55.0       7.0       3.0
...              ...          ...          ...       ...       ...
32031           15.0         94.0         60.0       8.0       3.0
80097           19.0         74.0         56.0       1.0       6.0
```

Short description:
This code employs the train_test_split function from scikit-learn to split the dataset into training and testing sets, with 30% of the data reserved for testing. The resulting x_train, x_test, y_train, and y_test DataFrames are ready for training and evaluating models in this project.

14.

```
In [33]: print(inputs.shape)
         print(target.shape)
```

print(target.shape)

```
(61893, 11)
(61893, 2)
```

Short description:
This code snippet prints the dimensions of the inputs and target DataFrames, indicating that there are 61,893 samples with 11 features in the input data and 61,893 samples with 2 target variables in the project.

15.

```
In [34]: y_train
```

Out[34]:

	RainToday	RainTomorrow
12183	0	0
70699	0	0
86115	0	0
6207	0	0
123588	0	0
...	...	...
6307	0	0
88934	0	1
95202	0	0
67872	0	1
34970	1	0

43325 rows × 2 columns

Short description:

This snippet shows a portion of the training set (y_train), displaying RainToday and RainTomorrow labels for various samples. Each row represents a sample, and the columns represent binary labels (0 or 1) for rain occurrence on the specified days.

16.

```
# Naive Bayes implementation
# Count the occurrences of each class in the target
total_samples = len(y_train)
total_rain_today = y_train['RainToday'].sum()
total_no_rain_today = total_samples - total_rain_today

total_rain_tomorrow = y_train['RainTomorrow'].sum()
total_no_rain_tomorrow = total_samples - total_rain_tomorrow

# Calculate class probabilities
prob_rain_today = total_rain_today / total_samples
prob_no_rain_today = total_no_rain_today / total_samples

prob_rain_tomorrow = total_rain_tomorrow / total_samples
prob_no_rain_tomorrow = total_no_rain_tomorrow / total_samples

# Separate the training data based on the target classes
rain_today_data = x_train[y_train['RainToday'] == 1]
no_rain_today_data = x_train[y_train['RainToday'] == 0]

rain_tomorrow_data = x_train[y_train['RainTomorrow'] == 1]
no_rain_tomorrow_data = x_train[y_train['RainTomorrow'] == 0]

# Calculate conditional probabilities for RainToday
prob_rain_today_given_data = len(rain_today_data) / total_samples
prob_no_rain_today_given_data = len(no_rain_today_data) / total_samples

# Calculate conditional probabilities for RainTomorrow
prob_rain_tomorrow_given_data = len(rain_tomorrow_data) / total_samples
prob_no_rain_tomorrow_given_data = len(no_rain_tomorrow_data) / total_samples
```

```
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 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, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0,
0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1,
0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0,
0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0,
1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0,
1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1,
1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1,
0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0,
```

Short description:
This code implements a basic Naive Bayes classifier for predicting rain occurrence (RainToday and RainTomorrow) based on weather features. It calculates class and conditional probabilities and uses them to make predictions on a test set, producing a list of predicted outcomes.

17.

In [38]:
```
# Example of a new data point for testing
new_data_point = {
    'MinTemp': 18.3,
    'MaxTemp': 27.1,
    'Rainfall':1.4,
    'Evaporation': 2.4,
    'Sunshine': 5.3,
    'WindSpeed9am': 6,
    'WindSpeed3pm': 22,
    'Humidity9am': 76,
    'Humidity3pm': 73,
    'Cloud9am': 6.0,
    'Cloud3pm': 6.0,
}

# Use the predict_naive_bayes function to make a prediction
prediction = predict_naive_bayes(new_data_point)
print(prediction)
```

```
1
```

Short description:
This code provides an example of using the Naive Bayes classifier to predict rain occurrence for a new weather data point. The predict_naive_bayes function is applied to the given features, and the output indicates the predicted outcome (1 for rain, 0 for no rain).

18.

```python
from sklearn.metrics import accuracy_score

# Convert the list of predictions to a pandas Series for easier comparison

predicted_series = pd.Series(predictions, index=y_test.index)

# Calculate accuracy

accuracy = accuracy_score(y_test['RainTomorrow'], predicted_series)

print(f'Accuracy: {accuracy * 100:.2f}%')
```

```
Accuracy: 76.28%
```

Short description:
This code evaluates the accuracy of a Naive Bayes classifier in predicting rain occurrence on the test set. Achieving 76.28% accuracy, it assesses the model's performance for project validation.

Conclusion:

In conclusion, this project unveils the wealth of insights within a decade-long Australian weather dataset. From data loading to preprocessing and predictive modeling, the code facilitates a journey through meteorological intricacies. The refined dataset, enriched with numerical mappings and split for training and testing, sets the stage for impactful analysis. The Naive Bayes classifier, a key feature, offers predictions for rain occurrences, showcasing the project's potential for informed decision-making and a deeper understanding of Australia's dynamic weather patterns. This venture stands as a testament to the power of data science in unraveling nature's mysteries.