

Rain Prediction ANN

Import Libraries

```
In [7]: ┌─▶ import numpy as np
      import pandas as pd
      import seaborn as sns
      import matplotlib.pyplot as plt
      import warnings
      warnings.filterwarnings("ignore")
```



```
In [8]: ┌─▶ from sklearn.preprocessing import LabelEncoder
      from sklearn import preprocessing
      from sklearn.preprocessing import StandardScaler
```



```
In [9]: ┌─▶ from sklearn.model_selection import train_test_split
      from sklearn.preprocessing import MinMaxScaler
      from tensorflow.keras.models import Sequential
      from tensorflow.keras.layers import Dense, Activation, Dropout
      from tensorflow.keras.callbacks import EarlyStopping
      from tensorflow.keras.layers import Dropout
      from sklearn.metrics import classification_report, confusion_matrix
      from sklearn.metrics import r2_score
      from sklearn.metrics import mean_absolute_error, mean_absolute_percentage_error
```

Loading Data

```
In [11]: ┌─▶ data = pd.read_csv(r"C:\Users\hp\OneDrive\Documents\Desktop\weatherAUS.csv")
      df = data.copy()
```



```
In [12]: ┌─▶ data.head()
```


Out[12]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	
2	2008-12-03	Albury	12.9	25.7	0.0	NaN	NaN	WSW	
3	2008-12-04	Albury	9.2	28.0	0.0	NaN	NaN	NE	
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	


```
In [15]: ┌─▶ data.info()
      510 rows × 23 columns
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 145460 entries, 0 to 145459
```



```
In [14]: ┌─▶ data.dtypes(total 23 columns):
      # Column          Non-Null Count   Dtype
      -----
```


Out[14]:

	Date	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed
0	145460 non-null object	145460 non-null float64	145460 non-null object	145460 non-null float64				
1	Location	145460 non-null object	145460 non-null float64	145460 non-null float64	145460 non-null float64	145460 non-null float64	145460 non-null object	145460 non-null float64
2	Count	145460.000000	145460.000000	145460.000000	145460.000000	145460.000000	145460.000000	145460.000000
3	MinTemp	14.0970000000	14.0970000000	14.0970000000	14.0970000000	14.0970000000	14.0970000000	14.0970000000
4	MaxTemp	32.194034	32.194034	32.194034	32.194034	32.194034	32.194034	32.194034
5	Rainfall	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
6	Evaporation	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
7	Sunshine	2.300000	2.300000	2.300000	2.300000	2.300000	2.300000	2.300000
8	WindGustDir	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
9	WindGustSpeed	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
10	WindDir9am	12.000000	12.000000	12.000000	12.000000	12.000000	12.000000	12.000000
11	WindDir3pm	12.000000	12.000000	12.000000	12.000000	12.000000	12.000000	12.000000
12	WindSpeed9am	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
13	WindSpeed3pm	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
14	Humidity9am	83.900000	83.900000	83.900000	83.900000	83.900000	83.900000	83.900000
15	Pressure9am	1012.000000	1012.000000	1012.000000	1012.000000	1012.000000	1012.000000	1012.000000
16	Pressure3pm	1012.000000	1012.000000	1012.000000	1012.000000	1012.000000	1012.000000	1012.000000
17	Cloud9am	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
18	Cloud3pm	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

In [15]: `# data.info()`

`<class 'pandas.core.frame.DataFrame'>`

RangeIndex: 145460 entries, 0 to 145459

In [14]: `# Data.describe():`

#	Column	Non-Null Count	Dtype
0	Date	145460	datetime64[ns]
1	Location	145460	non-null object
2	MinTemp	145460	float64
3	MaxTemp	144199	float64
4	Rainfall	142199	float64
5	Evaporation	82670	float64
6	Sunshine	75625	float64
7	WindGustDir	135134	object
8	WindGustSpeed	135193	float64
9	WindDir9am	134894	object
10	WindDir3pm	141232	object
11	WindSpeed9am	143892	float64
12	WindSpeed3pm	142398	float64
13	Humidity9am	142806	float64
14	Humidity3pm	140953	float64
15	Pressure9am	130395	float64
16	Pressure3pm	130432	float64
17	Cloud9am	89572	float64
18	Cloud3pm	86102	float64
19	Temp9am	143693	float64
20	Temp3pm	141851	float64
21	RainToday	142199	non-null object
22	RainTomorrow	142193	non-null object

dtypes: float64(16), object(7)

memory usage: 25.5+ MB

In [16]: `# finding null value in Data`

`data.isnull().sum()`

Out[16]:

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

In [19]: `# adding different columns for days month and year`

`data["year"] = data["Date"].dt.year`

`data["month"] = data["Date"].dt.month`

`data["day"] = data["Date"].dt.day`

In [17]: `# data is object thus changing it to datetime`

In [20]: `data[Header] = pd.to_datetime(data["Date"])`

In [18]: `# checking the data view`

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	
2	2008-12-03	Albury	13.4	25.7	0.0	NaN	NaN	WSW	
3	2008-12-04	Albury	12.9	28.0	0.0	NaN	NaN	NE	
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	
5	2008-12-06	Albury	9.2	28.0	0.0	NaN	NaN	NE	
6	2008-12-07	Albury	17.5	32.3	1.0	NaN	NaN	W	
7	2008-12-08	Albury	17.5	32.3	1.0	NaN	NaN	W	
8	2008-12-09	Albury	17.5	32.3	1.0	NaN	NaN	W	
9	2008-12-10	Albury	17.5	32.3	1.0	NaN	NaN	W	
10	2008-12-11	Albury	17.5	32.3	1.0	NaN	NaN	W	
11	2008-12-12	Albury	17.5	32.3	1.0	NaN	NaN	W	
12	2008-12-13	Albury	17.5	32.3	1.0	NaN	NaN	W	
13	2008-12-14	Albury	17.5	32.3	1.0	NaN	NaN	W	
14	2008-12-15	Albury	17.5	32.3	1.0	NaN	NaN	W	
15	2008-12-16	Albury	17.5	32.3	1.0	NaN	NaN	W	
16	2008-12-17	Albury	17.5	32.3	1.0	NaN	NaN	W	
17	2008-12-18	Albury	17.5	32.3	1.0	NaN	NaN	W	
18	2008-12-19	Albury	17.5	32.3	1.0	NaN	NaN	W	
19	2008-12-20	Albury	17.5	32.3	1.0	NaN	NaN	W	
20	2008-12-21	Albury	17.5	32.3	1.0	NaN	NaN	W	
21	2008-12-22	Albury	17.5	32.3	1.0	NaN	NaN	W	
22	2008-12-23	Albury	17.5	32.3	1.0	NaN	NaN	W	
23	2008-12-24	Albury	17.5	32.3	1.0	NaN	NaN	W	
24	2008-12-25	Albury	17.5	32.3	1.0	NaN	NaN	W	
25	2008-12-26	Albury	17.5	32.3	1.0	NaN	NaN	W	
26	2008-12-27	Albury	17.5	32.3	1.0	NaN	NaN	W	
27	2008-12-28	Albury	17.5	32.3	1.0	NaN	NaN	W	
28	2008-12-29	Albury	17.5	32.3	1.0	NaN	NaN	W	
29	2008-12-30	Albury	17.5	32.3	1.0	NaN	NaN	W	
30	2008-12-31	Albury	17.5	32.3	1.0	NaN	NaN	W	
31	2009-01-01	Albury	17.5	32.3	1.0	NaN	NaN	W	
32	2009-01-02	Albury	17.5	32.3	1.0	NaN	NaN	W	
33	2009-01-03	Albury	17.5	32.3	1.0	NaN	NaN	W	
34	2009-01-04	Albury	17.5	32.3	1.0	NaN	NaN	W	
35	2009-01-05	Albury	17.5	32.3	1.0	NaN	NaN	W	
36	2009-01-06	Albury	17.5	32.3	1.0	NaN	NaN	W	
37	2009-01-07	Albury	17.5	32.3	1.0	NaN	NaN	W	
38	2009-01-08	Albury	17.5	32.3	1.0	NaN	NaN	W	
39	2009-01-09	Albury	17.5	32.3	1.0	NaN	NaN	W	
40	2009-01-10	Albury	17.5	32.3	1.0	NaN	NaN	W	
41	2009-01-11	Albury	17.5	32.3	1.0	NaN	NaN	W	
42	2009-01-12	Albury	17.5	32.3	1.0	NaN	NaN	W	
43	2009-01-13	Albury	17.5	32.3	1.0	NaN	NaN	W	
44	2009-01-14	Albury	17.5	32.3	1.0	NaN	NaN	W	
45	2009-01-15	Albury	17.5	32.3	1.0	NaN	NaN	W	
46	2009-01-16	Albury	17.5	32.3	1.0	NaN	NaN	W	
47	2009-01-17	Albury	17.5	32.3	1.0	NaN	NaN	W	
48	2009-01-18	Albury	17.5	32.3	1.0	NaN	NaN	W	
49	2009-01-19	Albury	17.5	32.3	1.0	NaN	NaN	W	
50	2009-01-20	Albury	17.5	32.3	1.0	NaN	NaN	W	
51	2009-01-21	Albury	17.5	32.3	1.0	NaN	NaN	W	
52	2009-01-22	Albury	17.5	32.3	1.0	NaN	NaN	W	
53	2009-01-23	Albury	17.5	32.3	1.0	NaN	NaN	W	
54	2009-01-24	Albury	17.5	32.3	1.0	NaN	NaN	W	
55	2009-01-25	Albury	17.5	32.3	1.0	NaN	NaN	W	
56	2009-01-26	Albury	17.5	32.3	1.0	NaN	NaN	W	
57	2009-01-27	Albury	17.5	32.3	1.0	NaN	NaN	W	
58	2009-01-28	Albury	17.5	32.3	1.0	NaN	NaN	W	
59	2009-01-29	Albury	17.5	32.3	1.0	NaN	NaN	W	
60	2009-01-30	Albury	17.5	32.3	1.0	NaN	NaN	W	
61	2009-01-31	Albury	17.5	32.3	1.0	NaN	NaN	W	
62	2009-02-01	Albury	17.5	32.3	1.0	NaN	NaN	W	
63	2009-02-02	Albury	17.5	32.3	1.0	NaN	NaN	W	
64	2009-02-03	Albury	17.5	32.3	1.0	NaN	NaN	W	
65	2009-02-04	Albury	17.5	32.3	1.0	NaN	NaN	W	
66	2009-02-05	Albury	17.5	32.3	1.0	NaN	NaN	W	
67	2009-02-06	Albury	17.5	32.3	1.0	NaN	NaN	W	
68	2009-02-07	Albury	17.5	32.3	1.0	NaN	NaN	W	
69	2009-02-08	Albury	17.5	32.3	1.0	NaN	NaN	W	
70	2009-02-09	Albury	17.5	32.3	1.0	NaN	NaN	W	
71	2009-02-10	Albury	17.5	32.3	1.0	NaN	NaN	W	
72	2009-02-11	Albury	17.5	32.3	1.0	NaN	NaN	W	
73	2009-02-12	Albury	17.5	32.3	1.0	NaN	NaN	W	
74	2009-02-13	Albury	17.5	32.3	1.0	NaN	NaN	W	
75	2009-02-14	Albury	17.5	32.3	1.0	NaN	NaN	W	
76	2009-02-15	Albury	17.5	32.3	1.0	NaN	NaN	W	
77	2009-02-16	Albury	17.5	32.3	1.0	NaN	NaN	W	
78	2009-02-17	Albury	17.5	32.3	1.0	NaN	NaN	W	
79	2009-02-18	Albury	17.5	32.3	1.0	NaN	NaN	W	
80	2009-02-19	Albury	17.5	32.3	1.0	NaN	NaN	W	
81	2009-02-20	Albury	17.5	32.3	1.0	NaN	NaN	W	
82	2009-02-21	Albury	17.5	32.3	1.0	NaN	NaN	W	
83	2009-02-22	Albury	17.5	32.3	1.0	NaN	NaN	W	
84	2009-02-23	Albury	17.5	32.3	1.0	NaN	NaN	W	
85	2009-02-24	Albury	17.5	32.3	1.0	NaN	NaN	W	
86	2009-02-25	Albury	17.5	32.3	1.0	NaN	NaN	W	
87	2009-02-26	Albury	17.5	32.3	1.0	NaN	NaN	W	
88	2009-02-27	Albury	17.5	32.3	1.0	NaN	NaN	W	
89	2009-02-28	Albury	17.5	32.3	1.0	NaN	NaN	W	
90	2009-02-29	Albury	17.5	32.3	1.0	NaN	NaN	W	
91	2009-03-01	Albury	17.5	32.3	1.0	NaN	NaN	W	
92	2009-03-02	Albury	17.5	32.3	1.0	NaN	NaN	W	
93	2009-03-03	Albury	17.5	32.3	1.0	NaN	NaN	W	
94	2009-03-04	Albury	17.5	32.3	1.0	NaN	NaN	W	
95	2009-03-05	Albury	17.5	32.3	1.0	NaN	NaN	W	
96	2009-03-06	Albury	17.5	32.3	1.0	NaN	NaN	W	
97	2009-03-07	Albury	17.5	32.3	1.0	NaN	NaN	W	
98	2009-03-08	Albury	17.5	32.3	1.0	NaN	NaN	W	
99	2009-03-09	Albury	17.5	32.3	1.0	NaN	NaN	W	
100	2009-03-10	Albury	17.5	32.3	1.0	NaN	NaN	W	
101	2009-03-11	Albury	17.5	32.3	1.0	NaN	NaN	W	
102	2009-03-12	Albury	17.5	32.3	1.0	NaN	NaN	W	
103	2009-03-13	Albury	17.5	32.3	1.0	NaN	NaN	W	
104	2009-03-14	Albury	17.5	32.3	1.0	NaN	NaN	W	
105	2009-03-15	Albury	17.5	32.3	1.0	NaN	NaN	W	
106	2009-03-16	Albury	17.5	32.3	1.0	NaN	NaN	W	
107	2009-03-17	Albury	17.5	32.3	1.0	NaN	NaN	W	
108	2009-03-18	Albury	17.5	32.3	1.0	NaN	NaN	W	
109	2009-03-19	Albury	17.5	32.3	1.0	NaN	NaN	W	
110	2009-03-20	Albury	17.5	32.3	1.0	NaN	NaN	W	
111	2009-03-21	Albury	17.5	32.3	1.0	NaN	NaN	W	
112	2009-03-22	Albury	17.5	32.3	1.0	NaN	NaN	W</	

```
In [19]: Temp3pm      3609
In [19]: # adding different columns for days month and year
In [19]: RainToday     3261
In [19]: data["Year"] = data["Date"].dt.year
In [19]: RainTomorrow 3267
In [19]: data["month"] = data["Date"].dt.month
In [19]: data["day"] = data["Date"].dt.day

In [17]: # data is object thus changing it to datetime
In [20]: data["Date"] = pd.to_datetime(data["Date"])

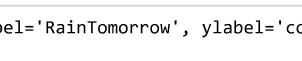
Out[20]:
In [18]: # checking the data view
          Date Location MinTemp MaxTemp Rainfall Evaporation Sunshine WindGustDir WindG
data["Date"]
0       2008-12-01    Albury   13.4    22.9     0.6        NaN      NaN        W
Out[18]: 0      2008-12-01
1      2008-12-02    Albury   7.4     25.1     0.0        NaN      NaN      WNW
2      2008-12-03    Albury   12.9    25.7     0.0        NaN      NaN      WSW
3      2008-12-04    Albury   12.9    25.7     0.0        NaN      NaN      WSW
4      2008-12-05    Albury   12.9    25.7     0.0        NaN      NaN      WSW
        ...
145455 2008-06-21    Albury   9.2     28.0     0.0        NaN      NaN      NE
145456 2017-06-22    Albury   17.5    32.3     1.0        NaN      NaN        W
145457 2017-06-23    Albury   17.5    32.3     1.0        NaN      NaN        W
145458 2017-06-24    Albury   17.5    32.3     1.0        NaN      NaN        W
145459 2017-06-25    Albury   17.5    32.3     1.0        NaN      NaN        W
In [18]: Shows 3609 rows × 9 columns
```

```
In [22]: # dropping te date column  
data.drop("Date",axis=1,inplace=True)  
  
In [24]: # Listing the column  
data.columns  
  
Out[24]: Index(['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine',  
       'WindGustDir', 'WindGustSpeed', 'WindDir9am', 'Winddir3pm',  
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',  
       'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am',  
       'Temp3pm', 'RainToday', 'RainTomorrow', 'year', 'month', 'day'],  
      dtype='object')
```

Data Visualization And Cleaning

```
In [26]: cols = ["#C2C4E2", "#EED4E5"]
sns.countplot(x= data["RainTomorrow"], palette= "husl")
```

Out[26]: <Axes: xlabel='RainTomorrow', ylabel='count'>



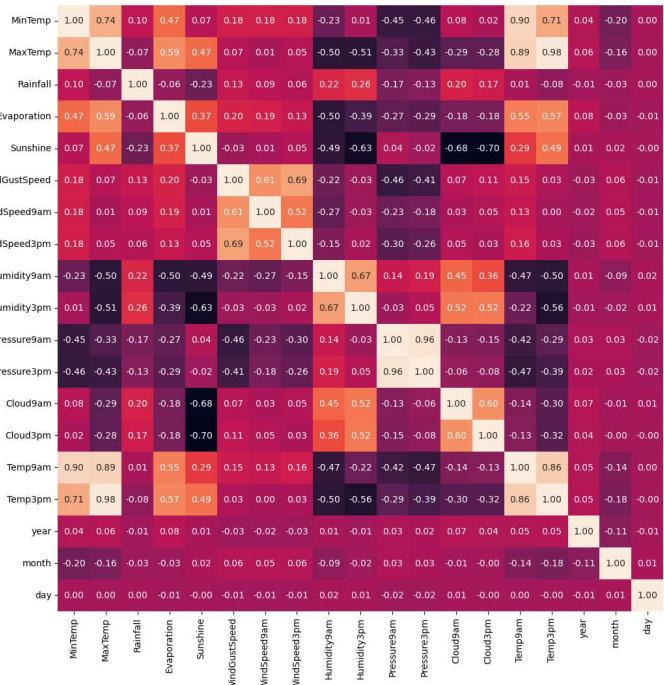
```
In [27]: plt.figure(figsize=(15,13))
ax = sns.heatmap(data.corr(), square=True, annot=True, fmt='.2f')
ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
plt.show()
```

The heatmap illustrates the correlation coefficients between various weather variables. The variables listed on the left are:

- MinTemp
- MaxTemp
- Rainfall
- Evaporation
- Sunshine
- WindGustSpeed
- WindSpeed9am
- WindSpeed3pm
- Humidity9am
- Humidity3pm
- Pressure9am
- Pressure3pm
- Cloud9am
- Cloud3pm

The correlation matrix shows strong positive correlations between variables like MaxTemp and Rainfall (~0.59), and negative correlations like MinTemp and Rainfall (~-0.10). The diagonal elements are all 1.0, representing perfect correlation with themselves.

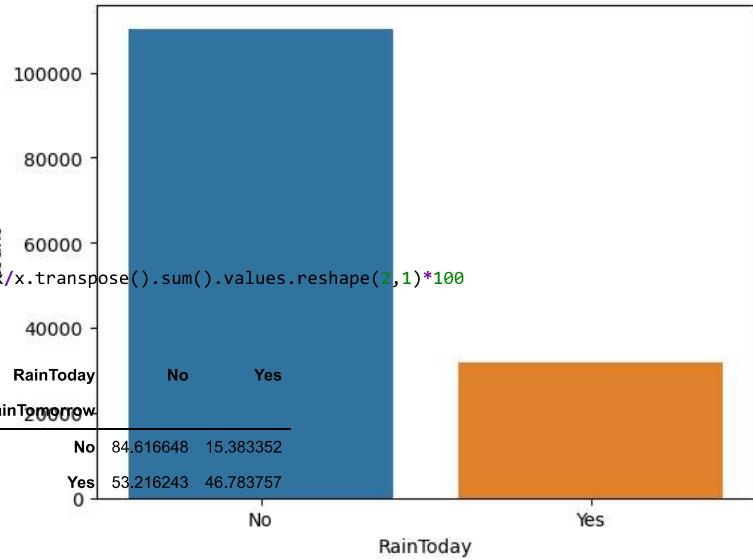
```
In [27]: plt.figure(figsize=(15,13))
ax = sns.heatmap(data.corr(), square=True, annot=True, fmt='.2f')
ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
plt.show()
```



Rain today ~ Rain tomorrow

```
In [29]: sns.countplot(x= data["RainToday"])
```

Out[29]: <Axes: xlabel='RainToday', ylabel='count'>

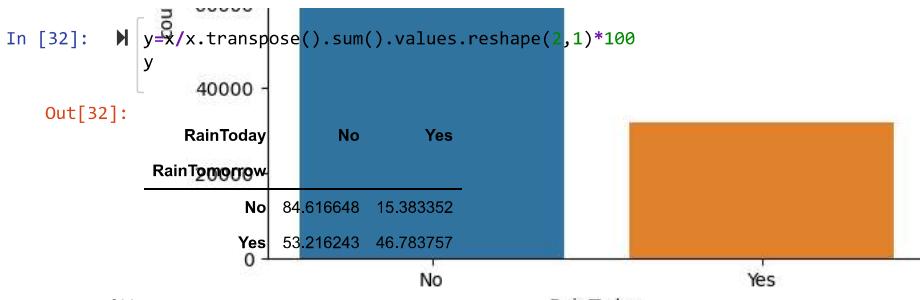


If it's
chance of raining tomorrow = 46%

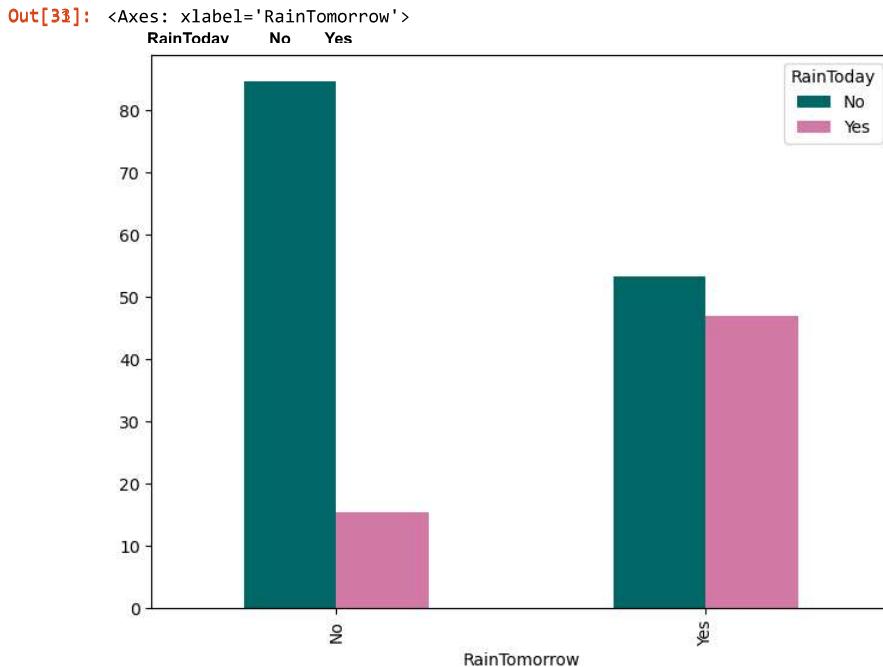
```
In [31]: x=pd.crosstab(data["RainTomorrow"],data["RainToday"])
In [33]: x.plot(kind="bar", figsize=(8,6),color=["#006666","#d279a6"])
```

Out[33]: <Axes: xlabel='RainTomorrow'>





In [31]: `x=pd.crosstab(data["RainTomorrow"],data["RainToday"])`
 In [33]: `x.plot(kind="bar", figsize=(8,6),color=["#006666","#d79a6"])`



Location ~ Rain Today

In [37]: `x=pd.crosstab(data["Location"],data["RainToday"])`
 x

Out[37]:

	RainToday	No	Yes
Location			
Adelaide	2402	689	
Albany	2114	902	
Albury	2394	617	
AliceSprings	2788	244	
BadgerysCreek	2345	583	
Ballarat	2247	781	
Bendigo	2472	562	
Brisbane	2452	709	
Cairns	2038	950	
Canberra	2789	629	
Cobar	2602	386	
CoffsHarbour	2084	869	
Dartmoor	2021	921	

```
In [37]: x=pd.crosstab(data["Location"],data["RainToday"])
x
```

Out[37]:

	RainToday	No	Yes
Location			
Adelaide	2402	689	
Albany	2114	902	
Albury	2394	617	
AliceSprings	2788	244	
BadgerysCreek	2345	583	
Ballarat	2247	781	
Bendigo	2472	562	
Brisbane	2452	709	
Cairns	2038	950	
Canberra	2789	629	
Cobar	2602	386	
CoffsHarbour	2084	869	
Dartmoor	2021	921	
Darwin	2341	852	
GoldCoast	2205	775	
Hobart	2426	762	
Katherine	1295	265	
Launceston	2328	700	
Melbourne	1799	636	
MelbourneAirport	2356	653	
Mildura	2680	327	
Moree	2460	394	
MountGambier	2110	921	
MountGinini	2088	819	
Newcastle	2224	731	
Nhil	1327	242	
NorahHead	2121	808	
NorfolkIsland	2045	919	
Nuriootpa	2411	592	
PearceRAAF	2257	505	
Penrith	2369	595	
Perth	2548	645	
PerthAirport	2442	567	

```
In [38]: # getting percentage of raining days and non raining days for each city
y=x/x.sum().values.reshape((-1, 1))*100
y
```

Out[38]:

	RainToday	No	Yes
Location			
Sydney	2483	472	
Sale	2357	643	
SalmonGums			
Albany	2513	520	
Townsville	70.092838	29.907162	
Albury	2430	568	
Tuggeranong	79.508469	20.491531	
Uluṟu	1406	116	
AliceSprings	91.952507	8.047493	
WaggaWagga	2440	536	
BadgerysCreek	80.088798	19.911202	
Walpole	1870	949	
Ballarat	74.207398	25.792602	
Watsonia	2261	738	
Bendigo	81.476599	18.523401	
Williamtown	1853	700	
Brisbane	77.570389	22.429611	
Witchcliffe	2073	879	
Cairns	68.206158	31.793842	
Wollongong	2269	713	
Canberra	81.597425	18.402575	
Woomera	2789	202	
Cobar	87.081660	12.918340	
CoffsHarbour	70.572299	29.427701	

```
In [38]: # getting Rain and no rain training days and non raining days for each city
y=x/x.transpose().sum().values.reshape((-1, 1))*100
y
```

Sale	2357	643
------	------	-----

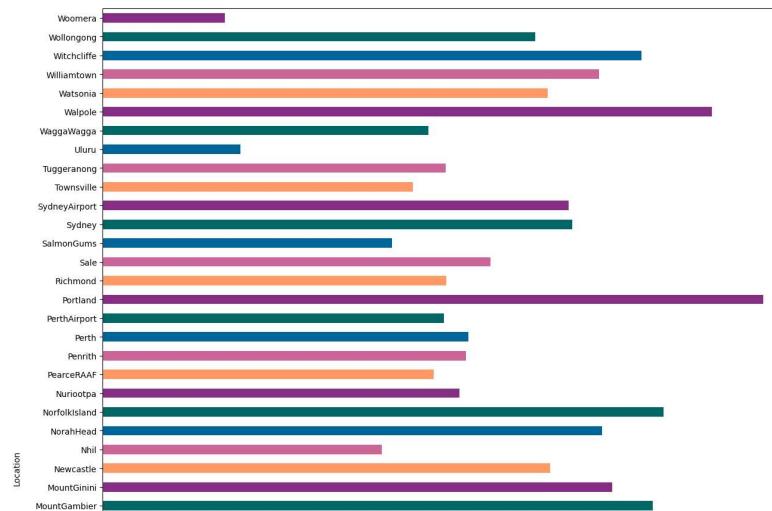
```
Out[38]:
```

SalmonGums	2483	472
RainToday	No	Yes
Sydney	2471	866

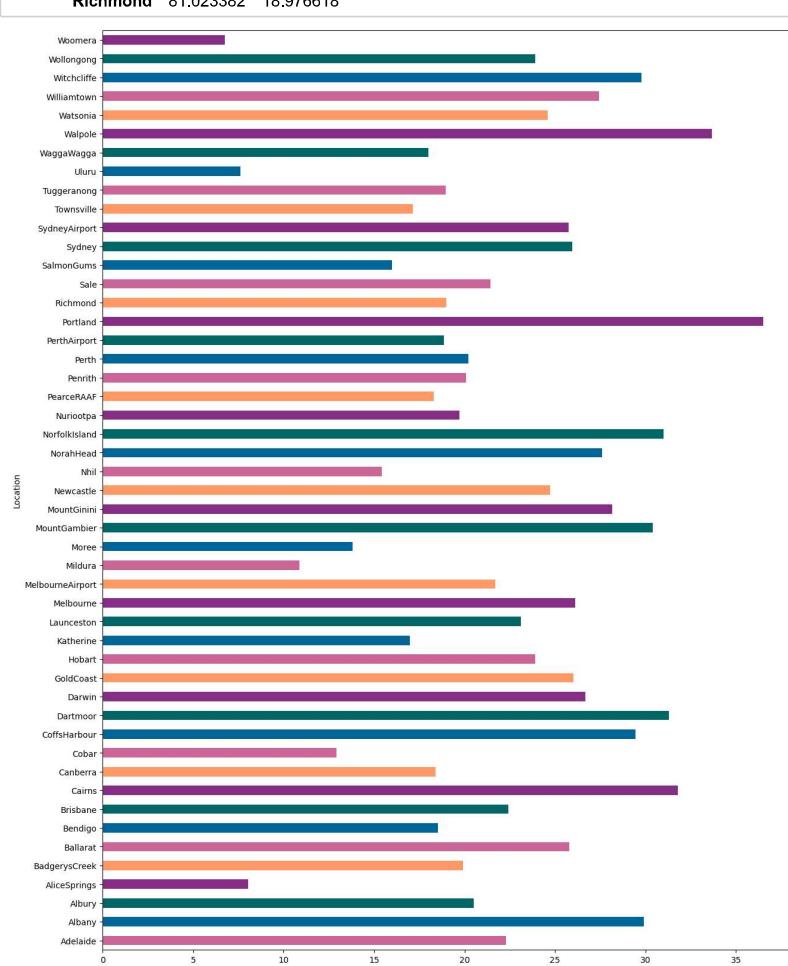
Location	2231	774
SydneyAirport	77.709479	22.290521
Adelaide	2513	520
Townsville	70.092838	29.907162
Albany	79.508469	20.491531
Tuggeranong	2430	568
Uluru	1406	116
AliceSprings	91.952507	8.047493
WaggaWagga	2440	536
BadgerysCreek	80.088798	19.911202
Walpole	1870	949
Ballarat	74.207398	25.792602
Watsonia	2261	738
Bendigo	81.476599	18.523401
Williamtown	1853	700
Brisbane	77.570389	22.429611
Witchcliffe	2073	879
Cairns	68.206158	31.793842
Wollongong	2269	713
Canberra	81.597425	18.402575
Woomera	2789	202
Cobar	87.081660	12.918340
CoffsHarbour	70.572299	29.427701
Dartmoor	68.694765	31.305235
Darwin	73.316630	26.683370
GoldCoast	73.993289	26.006711
Hobart	76.097867	23.902133
Katherine	83.012821	16.987179
Launceston	76.882431	23.117569
Melbourne	73.880903	26.119097
MelbourneAirport	78.298438	21.701562
Mildura	89.125374	10.874626
Moree	86.194814	13.805186
MountGambier	69.613989	30.386011
MountGinini	71.826625	28.173375
Newcastle	75.262267	24.737733
Nhil	84.576163	15.423837
NorahHead	72.413793	27.586207
NorfolkIsland	68.994602	31.005398
Nuriootpa	80.286380	19.713620
PearceRAAF	81.716148	18.283852
Penrith	79.925776	20.074224
Perth	79.799562	20.200438

```
In [39]: color=[ "#CC6699", "#006699", "#006666", '#862d86', '#ff9966' ]
y.Yes.plot(kind='bar', color=color);
```

Location	81.023382	18.976618
----------	-----------	-----------



```
In [39]: color=[ "#CC6699", "#006699", "#006666", '#862d86', '#ff9966' ]
y.Yes.plotPortland("hbar",figsize=(15,20),color=color);
```

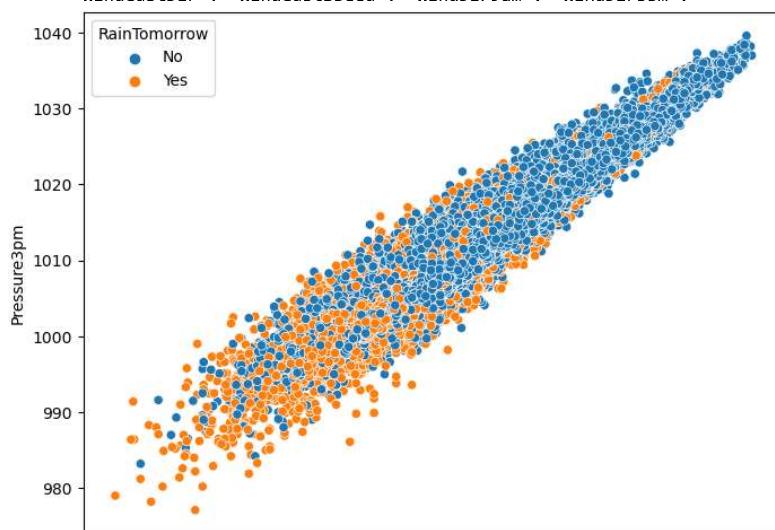


location affects the raining as for Portland, it's raining 36% of days and for Woomers it's raining only 6% of the days

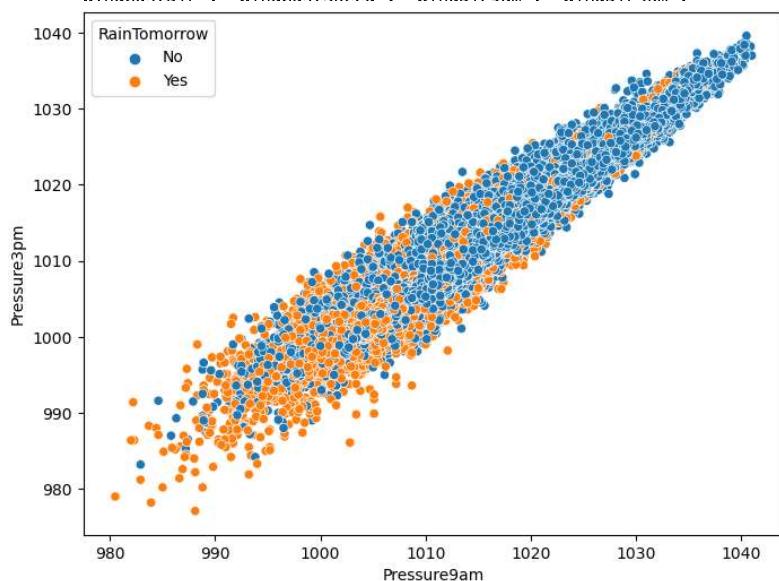
Effects of humidity and pressure on raining tomorrow

```
In [40]: data.columns
```

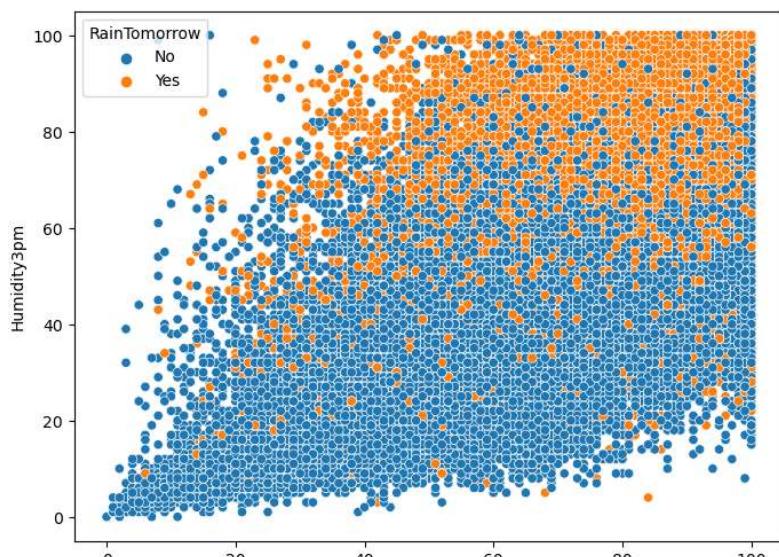
```
In [42]: plt.figure(figsize=(8,6))
sns.scatterplot(data=data,x='Pressure9am',y='Pressure3pm',hue='RainTomorrow')
    'WindGustDir'. 'WindGustSpeed'. 'WindDir9am'. 'WindDir3pm'.
```



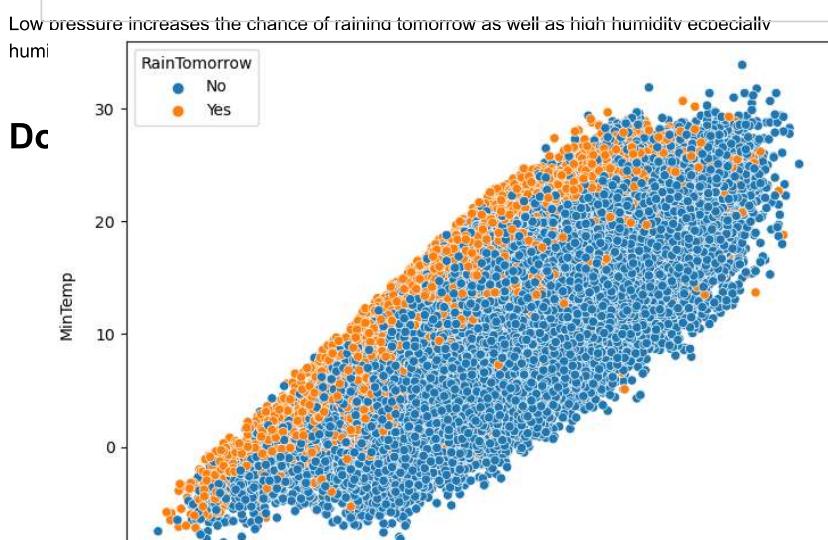
```
In [42]: plt.figure(figsize=(8,6))
sns.scatterplot(data=data,x='Pressure9am',y='Pressure3pm',hue='RainTomorrow')
plt.show()
```



```
In [44]: plt.figure(figsize=(8,6))
sns.scatterplot(data=data,x='Humidity9am',y='Humidity3pm',hue='RainTomorrow')
plt.show()
```



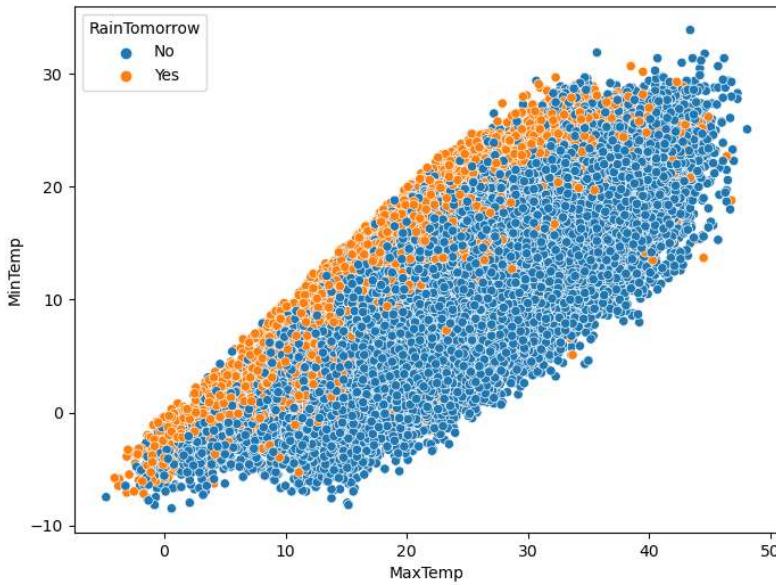
```
In [45]: plt.figure(figsize=(8,6))
sns.scatterplot(x='MaxTemp', y='MinTemp', data=data, hue='RainTomorrow');
plt.show()
```



```
In [45]: plt.figure(figsize=(8,6))
sns.scatterplot(x='MaxTemp', y='MinTemp', data=data, hue='RainTomorrow');
```

Low pressure increases the chance of raining tomorrow as well as high humidity especially
humid

Dc



Chance of raining tomorrow increases when the maximum temperature of the day and minimum one are close to each other

It's a rainy season in June, July and August as these are the coldest months of the year in Australia

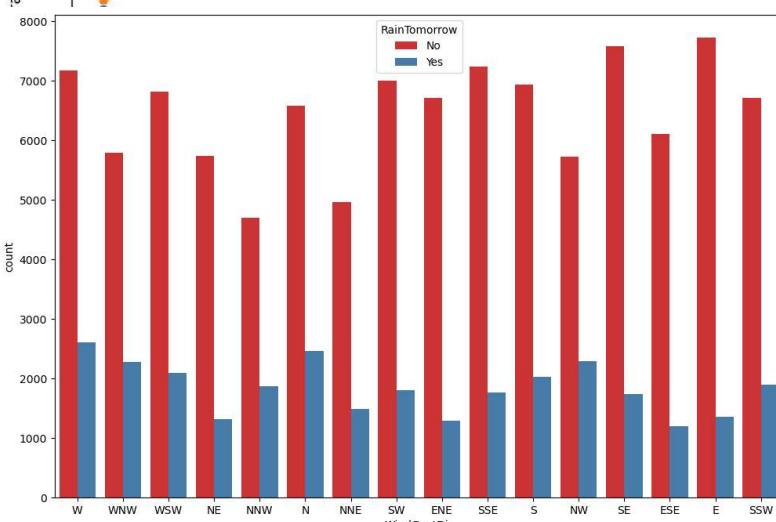
Effect of rain fall and evaporation on raining tomorrow

```
In [51]: plt.figure(figsize=(8,6))
sns.scatterplot(data=data,x='Rainfall',y='Evaporation',hue='RainTomorrow');
```



```
In [52]: plt.figure(figsize=(12,8))
sns.countplot(data=data,x='WindGustDir',hue='RainTomorrow',palette='Set1');
```

Highest to



In [52]:

plt.figure(figsize=(12,8))
sns.countplot(data=data,x='WindGustDir',hue='RainTomorrow',palette='Set1');

WindGustDir	No	Yes
W	7200	2600
NW	5800	2300
WSW	6900	2100
NE	5700	1300
NNW	4700	1900
N	6600	2400
NNE	5000	1400
SW	7000	1800
ENE	6700	1200
SSE	7200	1800
S	6900	2100
NW	5800	2300
SE	7600	1700
ESE	6100	1100
E	7700	1300
SSW	6700	1900

Handling Missing Values

```
In [54]: # percentage of missing data in each column  
data.isnull().sum()/data.shape[0]*100
```

```
Out[54]: Location          0.000000
          MinTemp         1.020899
          MaxTemp         0.866905
          Rainfall        2.241853
          Evaporation    43.166506
          Sunshine        48.009762
          WindGustDir     7.098859
          WindGustSpeed   7.055548
          WindDir9am      7.263853
          WindDir3pm       2.906641
          WindSpeed9am    1.214767
          WindSpeed3pm    2.105046
          Humidity9am     1.824557
          Humidity3pm      3.098446
          Pressure9am     10.356799
          Pressure3pm     10.331363
          Cloud9am         38.421559
          Cloud3pm         40.807095
          Temp9am          1.214767
          Temp3pm          2.481094
          RainToday        2.241853
```

```
In [57]: RainTomorrow    2.245978
      # numeric features 0.000000
      Month(data.dtypes = 0.000000)
      obj_cols = list(t[0].dtypes)
      dtype: float64
```

```
In [58]: for i in num_cols:  
In [55]:     # filling feature with target percentage of missing data using random choice  
     data[i].fillna(data[i].median(), inplace=True)  
     1st=['Evaporation', 'Sunshine', 'Cloud9am', 'Cloud3pm']  
     for col in 1st:
```

```
In [59]: datafish = list.sum()
          data[col] = data[col].fillna(pd.Series(
              np.random.choice(fill_list, size =
```

```
Out[59]: #categorical features
Mintemp
MaxTemp
Objectcols = list(s[s].index)
Rainfall
```

```
In [56]: > top_5_in_object_cols:
    WindGustDir.fillna(data[i].mode()[0], inplace=True)
    WindGustSpeed
    WindDir9am
    WindDir3pm
    WindSpeed9am
    WindSpeed3pm
    Humidity9am
    Humidity3pm
    Pressure9am
```

```
In [57]: RainTomorrow    2.245978
In [57]: # generic features 0.000000
In [57]: Monthdata.dtypes = .00000064)
In [57]: num_cols = list(t[0]000000)
In [57]: dtype: float64

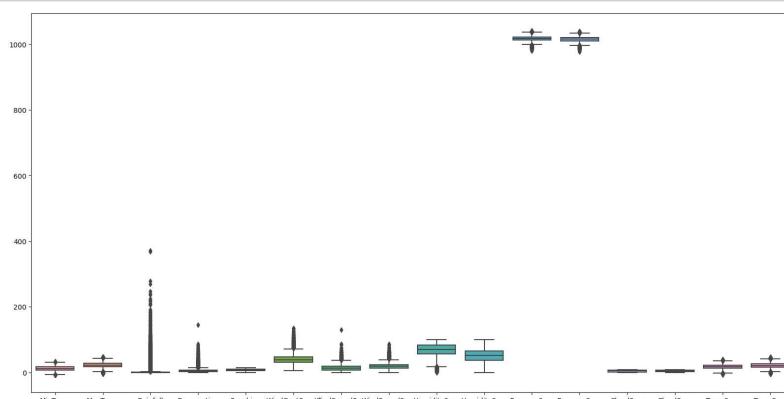
In [58]: for i in num_cols:
In [55]:     # filling missing values with large percentage of missing data using random choice
In [55]:     lst=[ 'Evaporation', 'Sunshine', 'Cloud9am', 'Cloud3pm' ]
In [55]:     for col in lst:
In [59]:         data.fillna(fill_list).sum()
In [59]:         data[col] = data[col].fillna(pd.Series(
In [59]:             np.random.choice(fill_list , size = len(data.index))))
Out[59]: Location
MinTemp
MaxTemp
WindGustDir
WindGustSpeed
Rainfall
Evaporation
Sunshine
WindGustDir
WindGustSpeed
WindDir9am
WindDir3pm
WindSpeed9am
WindSpeed3pm
Humidity9am
Humidity3pm
Pressure9am
Pressure3pm
Cloud9am
Cloud3pm
Temp9am
Temp3pm
RainToday
RainTomorrow
year
month
day
dtype: int64

In [56]: for i in object_cols:
In [56]:     data[i].fillna(data[i].mode()[0], inplace=True)
In [56]: WindGustDir
In [56]: WindGustSpeed
In [56]: WindDir9am
In [56]: WindDir3pm
In [56]: WindSpeed9am
In [56]: WindSpeed3pm
In [56]: Humidity9am
In [56]: Humidity3pm
In [56]: Pressure9am
In [56]: Pressure3pm
In [56]: Cloud9am
In [56]: Cloud3pm
In [56]: Temp9am
In [56]: Temp3pm
In [56]: RainToday
In [56]: RainTomorrow
In [56]: year
In [56]: month
In [56]: day
In [56]: dtype: int64
```

Headline Outliers

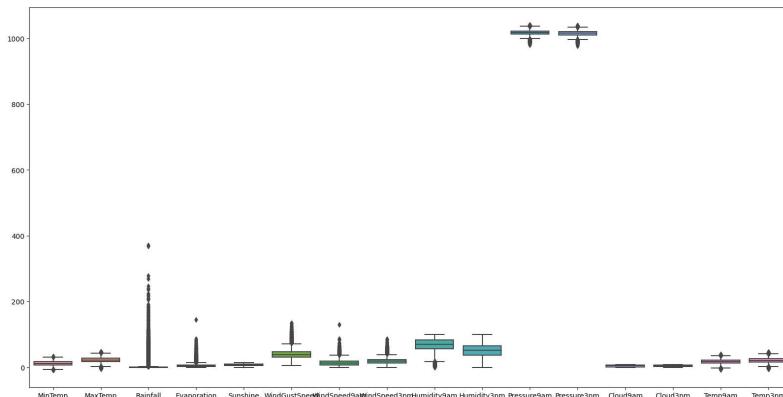
```
In [60]: num_cols
Out[60]: ['MinTemp',
'MaxTemp',
'Rainfall',
'Evaporation',
'Sunshine',
'WindGustSpeed',
'WindSpeed9am',
'WindSpeed3pm',
'Humidity9am',
'Humidity3pm',
'Pressure9am',
'Pressure3pm',
'Cloud9am',
'Cloud3pm']

In [61]: # Show all the outlier as a graph using boxplot for each columns
plt.figure(figsize=(20,10))
sns.boxplot(data=df,orient="v")
plt.show()
```



```
In [62]: # MinTemp
x=data.loc[(data['MinTemp']>30)|(data['MinTemp']<-6)]
print('No. of outliers =',x.shape[0],'Innerpercentage of the outliers =',str(rn)
```

```
In [61]: # Cloud9am
# Show 3pm, the outlier as a graph using boxplot for each columns
plt.figure(figsize=(20,10))
sns.boxplot(data=df,orient="v")
plt.show()
```



```
In [62]: # MinTemp
x=data.loc[(data['MinTemp']>30)|(data['MinTemp']<-6)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
data=data.loc[(data['MinTemp']<=30)&(data['MinTemp']>=-6)]
```

NO. of outliers = 76
percentage of the outliers = 0.05%

```
In [63]: # MaxTemp
x=data.loc[(data['MaxTemp']>45)|(data['MaxTemp']<1)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
data=data.loc[(data['MaxTemp']<=45)&(data['MaxTemp']>=1)]
```

NO. of outliers = 215
percentage of the outliers = 0.15%

```
In [64]: # Rainfall
x=data.loc[data['Rainfall']>55]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
data=data.loc[data['Rainfall']<=55]
```

NO. of outliers = 647
percentage of the outliers = 0.45%

```
In [66]: # Evaporation
x=data.loc[(data['Evaporation']>30)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
data=data.loc[(data['Evaporation']<=30)]
```

NO. of outliers = 378
percentage of the outliers = 0.26%

```
In [69]: # WindSpeed3pm
```

```
In [67]: # WindGustSpeed
x=data.loc[data['WindSpeed3pm']>45]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
x=data.loc[(data['WindGustSpeed']>75)&(data['WindGustSpeed']<45)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
data=data.loc[data['WindGustSpeed']<80]
No. of outliers = 445
percentage of the outliers = 0.31%
No. of outliers = 0
percentage of the outliers = 0.0%
```

```
In [70]: # Humidity9am
```

```
In [68]: # WindSpeed9am
x=data.loc[data['Humidity9am']<19]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
x=data.loc[(data['WindSpeed9am']>520)&(data['WindSpeed9am']<19)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
data=data.loc[data['WindSpeed9am']<40]
No. of outliers = 1703
percentage of the outliers = 1.21%
No. of outliers = 872
percentage of the outliers = 0.68%
```

```
In [71]: # Pressure9am
```

```
x=data.loc[(data['Pressure9am']>1035) | (data['Pressure9am'] <1000)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/len(data))*100,2))+'%')
data=data.loc[(data['Pressure9am']<=1035) & (data['Pressure9am'] >=1000)]
```

NO. of outliers = 1659
percentage of the outliers = 1.19%

```
In [69]: # WindSpeed3pm
x=data.loc[data['WindSpeed3pm']>45]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/80)*100,2))+'%')
data=data.loc[data['WindSpeed3pm']<=45]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/80)*100,2))+'%')
data=data.loc[data['WindGustSpeed']<80]
No. of outliers = 445
percentage of the outliers = 0.31%
In [70]: # Humidity9am
x=data.loc[data['Humidity9am']<=19]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/80)*100,2))+'%')
data=data.loc[data['Humidity9am']>19]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/80)*100,2))+'%')
data=data.loc[data['WindSpeed9am']<=40]
No. of outliers = 1703
percentage of the outliers = 1.21%
No. of outliers = 972
percentage of the outliers = 0.68%
In [71]: # Pressure9am
x=data.loc[(data['Pressure9am']>1035) | (data['Pressure9am'] <1000)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/80)*100,2))+'%')
data=data.loc[(data['Pressure9am']<=1035) & (data['Pressure9am'] >=1000)]
No. of outliers = 1659
percentage of the outliers = 1.19%
In [72]: # Pressure3pm
x=data.loc[(data['Pressure3pm']>1032) | (data['Pressure3pm'] <1000)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/80)*100,2))+'%')
data=data.loc[(data['Pressure3pm']<=1032) & (data['Pressure3pm'] >=1000)]
No. of outliers = 1147
percentage of the outliers = 0.83%
In [73]: # Temp9am
x=data.loc[(data['Temp9am']>=35) | (data['Temp9am']<=0)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/80)*100,2))+'%')
data=data.loc[(data['Temp9am']<35) & (data['Temp9am']>0)]
No. of outliers = 334
percentage of the outliers = 0.24%
In [74]: # Temp3pm
x=data.loc[(data['Temp3pm']>40) | (data['Temp3pm']<=0)]
print('NO. of outliers =',x.shape[0],'\npercentage of the outliers =',str(round((x.shape[0]/80)*100,2))+'%')
data=data.loc[(data['Temp3pm']<=40) & (data['Temp3pm']>0)]
No. of outliers = 364
percentage of the outliers = 0.27%
In [75]: # show all the outliers as a graph using boxplot for each column in the dataset
plt.figure(figsize=(20,10))
sns.boxplot(data=df,orient='v')
plt.show()#
In [77]: data.columns
Out[77]: Index(['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustDir', 'WindGustSpeed', 'WindDir9am', 'WindDir3pm', 'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm', 'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am', 'Temp3pm', 'RainToday', 'RainTomorrow', 'year', 'month', 'day'], dtype='object')

In [81]: # Encoding
from sklearn.preprocessing import LabelEncoder
label_encoder=LabelEncoder()
for i in object_cols:
    data[i] = label_encoder.fit_transform(data[i])
In [76]: data.shape
Out[76]: (135877, 25)
In [82]: x.drop(["RainTomorrow", "day"],axis=1).values
y = data["RainTomorrow"].values
```

```
In [77]: In [77]: data.columns
```

```
Out[77]: Index(['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine',
   'WindGustDir', 'WindGustSpeed', 'WindDir9am', 'WindDir3pm',
   'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
   'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am',
   'Temp3pm', 'RainToday', 'RainTomorrow', 'year', 'month', 'day'],
  dtype='object')
```

En

```
In [81]: # Encoding
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
for i in object_cols:
    data[i] = label_encoder.fit_transform(data[i])
```

```
In [76]: In [76]: data.shape
```

```
Out[76]: (135877, 25)
```

```
y = data["RainTomorrow"].values
```

Modeling

```
In [87]: In [87]: x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.25,random_state=42)
```

```
In [88]: In [88]: scaler = MinMaxScaler()
scaler.fit(x_train)
x_train = scaler.transform(x_train)
x_test = scaler.transform(x_test)
```

```
In [89]: In [89]: from tensorflow.keras.optimizers import Adam
from keras import regularizers
model = Sequential()

model.add(Dense(units=24,activation='tanh',))

#model.add(Dropout(0.2))

model.add(Dense(units=18,activation='tanh',kernel_regularizer=regularizers.l2(0.01)))
#model.add(Dropout(0.2))

model.add(Dense(units=23,activation='tanh',kernel_regularizer=regularizers.l2(0.01)))
model.add(Dropout(0.5))

model.add(Dense(units=12,activation='tanh'))
```

```
In [93]: In [93]: model.add(Dropout(0.2))
model.fit(x=x_train,
          y=y_train,
          epochs=100,
          validation_data=(x_test, y_test),
          verbose=1,
          callbacks=[early_stop],
          # For a binary classification problem
          opt = Adam(learning_rate=0.001))
model.compile(loss='binary_crossentropy', optimizer=opt)
```

```
In [91]: In [91]: Epoch 1/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3734
al_loss: 0.3734
Epoch 2/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3817 - v
al_loss: 0.3716
Epoch 3/150
3185/3185 [=====] - 8s 2ms/step - loss: 0.3784 - v
al_loss: 0.3685
Epoch 4/150
3185/3185 [=====] - 8s 3ms/step - loss: 0.3757 - v
al_loss: 0.3648
Epoch 5/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3739 - v
al_loss: 0.3708
Epoch 6/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3724 - v
al_loss: 0.3613
```

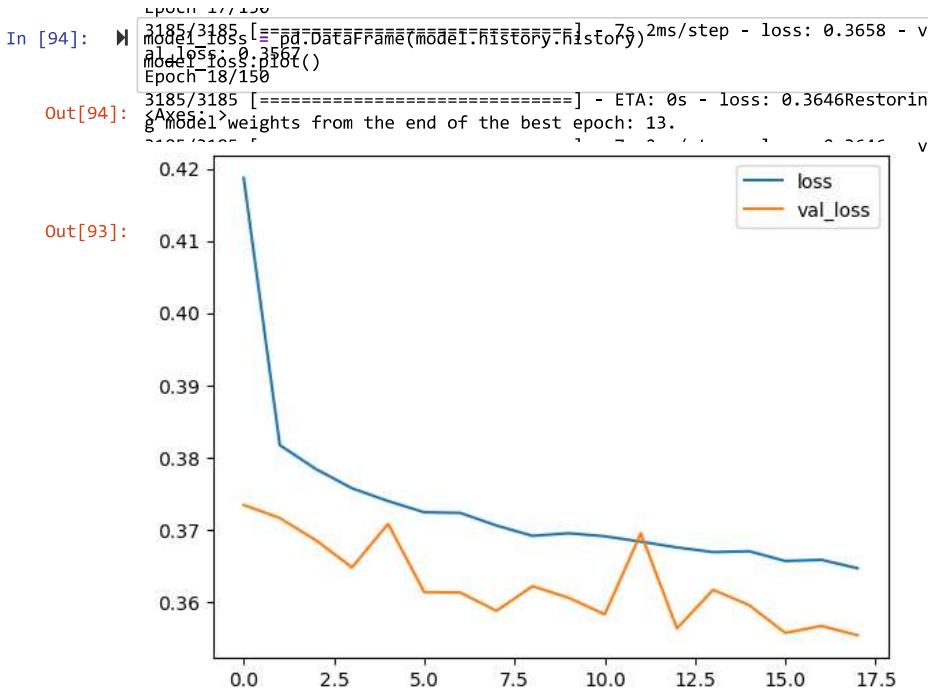
```
In [93]: model.add(Dropout(0.2))
model.fit(x=x_train,
          y=y_train, epochs=1, activation='sigmoid')
          validation_data=(x_test, y_test), verbose=1,
          callbacks=[early_stop],
# For a binary classification problem
opt = Adam(learning_rate=0.001)
model.compile(loss='binary_crossentropy', optimizer=opt)
```

```
In [91]: Epoch 1/150
3185/3185 [=====] - EarlyStopping(monitor='val_loss', mode='min', min_delta=0.001, v
al_loss: 0.3734
Epoch 2/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3817 - v
al_loss: 0.3716
Epoch 3/150
3185/3185 [=====] - 8s 2ms/step - loss: 0.3784 - v
al_loss: 0.3685
Epoch 4/150
3185/3185 [=====] - 8s 3ms/step - loss: 0.3757 - v
al_loss: 0.3648
Epoch 5/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3739 - v
al_loss: 0.3708
Epoch 6/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3724 - v
al_loss: 0.3613
Epoch 7/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3723 - v
al_loss: 0.3613
Epoch 8/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3706 - v
al_loss: 0.3587
Epoch 9/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3691 - v
al_loss: 0.3622
Epoch 10/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3695 - v
al_loss: 0.3605
Epoch 11/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3691 - v
al_loss: 0.3583
Epoch 12/150
3185/3185 [=====] - 8s 2ms/step - loss: 0.3683 - v
al_loss: 0.3695
Epoch 13/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3675 - v
al_loss: 0.3563
Epoch 14/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3669 - v
al_loss: 0.3617
Epoch 15/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3670 - v
al_loss: 0.3595
Epoch 16/150
3185/3185 [=====] - 7s 2ms/step - loss: 0.3657 - v
al_loss: 0.3557
Epoch 17/150
```

```
In [94]: model.fit(x=x_train, y=y_train, epochs=1, validation_data=(x_val, y_val), v
al_loss: 0.3552
Epoch 18/150
```

Out[94]: 3185/3185 [=====] - ETA: 0s - loss: 0.3646Restoring model's weights from the end of the best epoch: 13.





In [95]:

```
prediction = model.predict(x_test)
prediction = (prediction>0.5)
```

1062/1062 [=====] - 1s 1ms/step

In [98]:

```
confusion_matrix(y_test,prediction)
```

Out[98]:

```
array([[25359, 1478],
       [3626, 3507]], dtype=int64)
```

In [99]:

```
print(classification_report(y_test,prediction))
```

	precision	recall	f1-score	support
0	0.87	0.94	0.91	26837
1	0.70	0.49	0.58	7133
accuracy			0.85	33970
macro avg	0.79	0.72	0.74	33970
weighted avg	0.84	0.85	0.84	33970

In [101]:

```
mean_squared_error(y_test ,prediction)
```

Out[101]:

```
0.15025022078304387
```

In [102]:

```
mean_absolute_error(y_test , prediction)
```

Out[102]:

```
0.15025022078304387
```

In [103]:

```
r2_score(y_test , prediction)
```

Out[103]:

```
0.09426735242587214
```

In []: