In [2]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import pickle
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
from pandas.plotting import scatter_matrix
from sklearn.preprocessing import LabelEncoder
import seaborn as sns
import numpy as np
```

In [3]:
```python
data =pd.read_csv("weatherAUS.csv")
data.head()
```

Out[3]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	...	Humidity9am	Humidity3pm	Pressure9am	Pressure3pm	Cloud9am	Cloud3pm	Temp9am	Temp3pm	RainToday	Ra
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	...	71.0	22.0	1007.7	1007.1	8.0	NaN	16.9	21.8	No	
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	...	44.0	25.0	1010.6	1007.8	NaN	NaN	17.2	24.3	No	
2	2008-12-03	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	...	38.0	30.0	1007.6	1008.7	NaN	2.0	21.0	23.2	No	
3	2008-12-04	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	...	45.0	16.0	1017.6	1012.8	NaN	NaN	18.1	26.5	No	
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	...	82.0	33.0	1010.8	1006.0	7.0	8.0	17.8	29.7	No	

5 rows × 23 columns

In [4]:
```python
data.shape
```

Out[4]:
```
(8425, 23)
```

In [5]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8425 entries, 0 to 8424
Data columns (total 23 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Date           8425 non-null   object 
 1   Location       8425 non-null   object 
 2   MinTemp        8350 non-null   float64
 3   MaxTemp        8365 non-null   float64
 4   Rainfall       8185 non-null   float64
 5   Evaporation    4913 non-null   float64
 6   Sunshine       4431 non-null   float64
 7   WindGustDir    7434 non-null   object 
 8   WindGustSpeed  7434 non-null   float64
 9   WindDir9am     7596 non-null   object 
 10  WindDir3pm     8117 non-null   object 
 11  WindSpeed9am   8349 non-null   float64
 12  WindSpeed3pm   8318 non-null   float64
 13  Humidity9am    8366 non-null   float64
 14  Humidity3pm    8323 non-null   float64
 15  Pressure9am    7116 non-null   float64
 16  Pressure3pm    7113 non-null   float64
 17  Cloud9am       6004 non-null   float64
 18  Cloud3pm       5970 non-null   float64
 19  Temp9am        8369 non-null   float64
 20  Temp3pm        8329 non-null   float64
 21  RainToday      8185 non-null   object 
 22  RainTomorrow   8186 non-null   object 
dtypes: float64(16), object(7)
memory usage: 1.5+ MB
```

In [6]:
```python
data.isnull().sum()
```

Out[6]:
```
Date                0
Location            0
MinTemp            75
MaxTemp            60
Rainfall          240
Evaporation      3512
Sunshine         3994
WindGustDir       991
WindGustSpeed     991
WindDir9am        829
WindDir3pm        308
WindSpeed9am       76
WindSpeed3pm      107
Humidity9am        59
Humidity3pm       102
Pressure9am      1309
Pressure3pm      1312
Cloud9am         2421
Cloud3pm         2455
Temp9am            56
Temp3pm            96
RainToday         240
RainTomorrow      239
dtype: int64
```

In [7]:
```python
data.describe()
```

Out[7]:

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am	Pressure3pm	Cloud9am	Cloud3pm	Temp9am	Temp3pm
count	8350.000000	8365.000000	8185.000000	4913.000000	4431.000000	7434.000000	8349.000000	8318.000000	8366.000000	8323.000000	7116.000000	7113.000000	6004.000000	5970.000000	8369.000000	8329.000000
mean	13.193305	23.859976	2.805913	5.389395	7.632205	40.174469	13.847646	18.533662	67.822496	51.249790	1017.640233	1015.236075	4.566622	4.503183	17.762015	22.442934
std	5.403596	6.136408	10.459379	5.044484	3.896235	14.665721	10.174579	9.766986	16.833283	18.423774	6.828699	6.766681	2.877658	2.731659	5.627035	5.980020
min	-2.000000	8.200000	0.000000	0.000000	0.000000	7.000000	0.000000	0.000000	10.000000	6.000000	989.800000	982.900000	0.000000	0.000000	1.900000	7.300000
25%	9.200000	19.300000	0.000000	2.600000	4.750000	30.000000	6.000000	11.000000	56.000000	39.000000	1013.000000	1010.400000	1.000000	2.000000	13.800000	18.000000
50%	13.300000	23.300000	0.000000	4.600000	8.700000	39.000000	13.000000	19.000000	68.000000	51.000000	1017.700000	1015.300000	5.000000	5.000000	17.800000	21.900000
75%	17.400000	28.000000	1.000000	7.000000	10.700000	50.000000	20.000000	24.000000	80.000000	63.000000	1022.300000	1019.800000	7.000000	7.000000	21.900000	26.400000
max	28.500000	45.500000	371.000000	145.000000	13.900000	107.000000	63.000000	83.000000	100.000000	99.000000	1039.000000	1036.000000	8.000000	8.000000	39.400000	44.100000

In [8]:
```python
data =data.drop(["Date","Location","Evaporation","Sunshine","Cloud9am","Cloud3pm"],axis =1)
data.head()
```

Out[8]:

	MinTemp	MaxTemp	Rainfall	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am	Pressure3pm	Temp9am	Temp3pm	RainToday	RainTomorrow
0	13.4	22.9	0.6	W	44.0	W	WNW	20.0	24.0	71.0	22.0	1007.7	1007.1	16.9	21.8	No	No
1	7.4	25.1	0.0	WNW	44.0	NNW	WSW	4.0	22.0	44.0	25.0	1010.6	1007.8	17.2	24.3	No	No
2	12.9	25.7	0.0	WSW	46.0	W	WSW	19.0	26.0	38.0	30.0	1007.6	1008.7	21.0	23.2	No	No
3	9.2	28.0	0.0	NE	24.0	SE	E	11.0	9.0	45.0	16.0	1017.6	1012.8	18.1	26.5	No	No
4	17.5	32.3	1.0	W	41.0	ENE	NW	7.0	20.0	82.0	33.0	1010.8	1006.0	17.8	29.7	No	No

In [9]:
```python
data.columns
```

Out[9]:
```
Index(['MinTemp', 'MaxTemp', 'Rainfall', 'WindGustDir', 'WindGustSpeed',
       'WindDir9am', 'WindDir3pm', 'WindSpeed9am', 'WindSpeed3pm',
       'Humidity9am', 'Humidity3pm', 'Pressure9am', 'Pressure3pm', 'Temp9am',
       'Temp3pm', 'RainToday', 'RainTomorrow'],
      dtype='object')
```

In [10]:
```python
le =LabelEncoder()
data["WindGustDir"]=le.fit_transform(data["WindGustDir"])
data["WindDir9am"]=le.fit_transform(data["WindDir9am"])
data["WindDir3pm"]=le.fit_transform(data["WindDir3pm"])
data["RainToday"]=le.fit_transform(data["RainToday"])
data["RainTomorrow"]=le.fit_transform(data["RainTomorrow"])
data.head()
```

Out[10]:

	MinTemp	MaxTemp	Rainfall	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am	Pressure3pm	Temp9am	Temp3pm	RainToday	RainTomorrow
0	13.4	22.9	0.6	13	44.0	13	14	20.0	24.0	71.0	22.0	1007.7	1007.1	16.9	21.8	0	0
1	7.4	25.1	0.0	14	44.0	6	15	4.0	22.0	44.0	25.0	1010.6	1007.8	17.2	24.3	0	0
2	12.9	25.7	0.0	15	46.0	13	15	19.0	26.0	38.0	30.0	1007.6	1008.7	21.0	23.2	0	0
3	9.2	28.0	0.0	4	24.0	9	0	11.0	9.0	45.0	16.0	1017.6	1012.8	18.1	26.5	0	0
4	17.5	32.3	1.0	13	41.0	1	7	7.0	20.0	82.0	33.0	1010.8	1006.0	17.8	29.7	0	0

In [12]:
```python
data =data.dropna(axis =0)
```

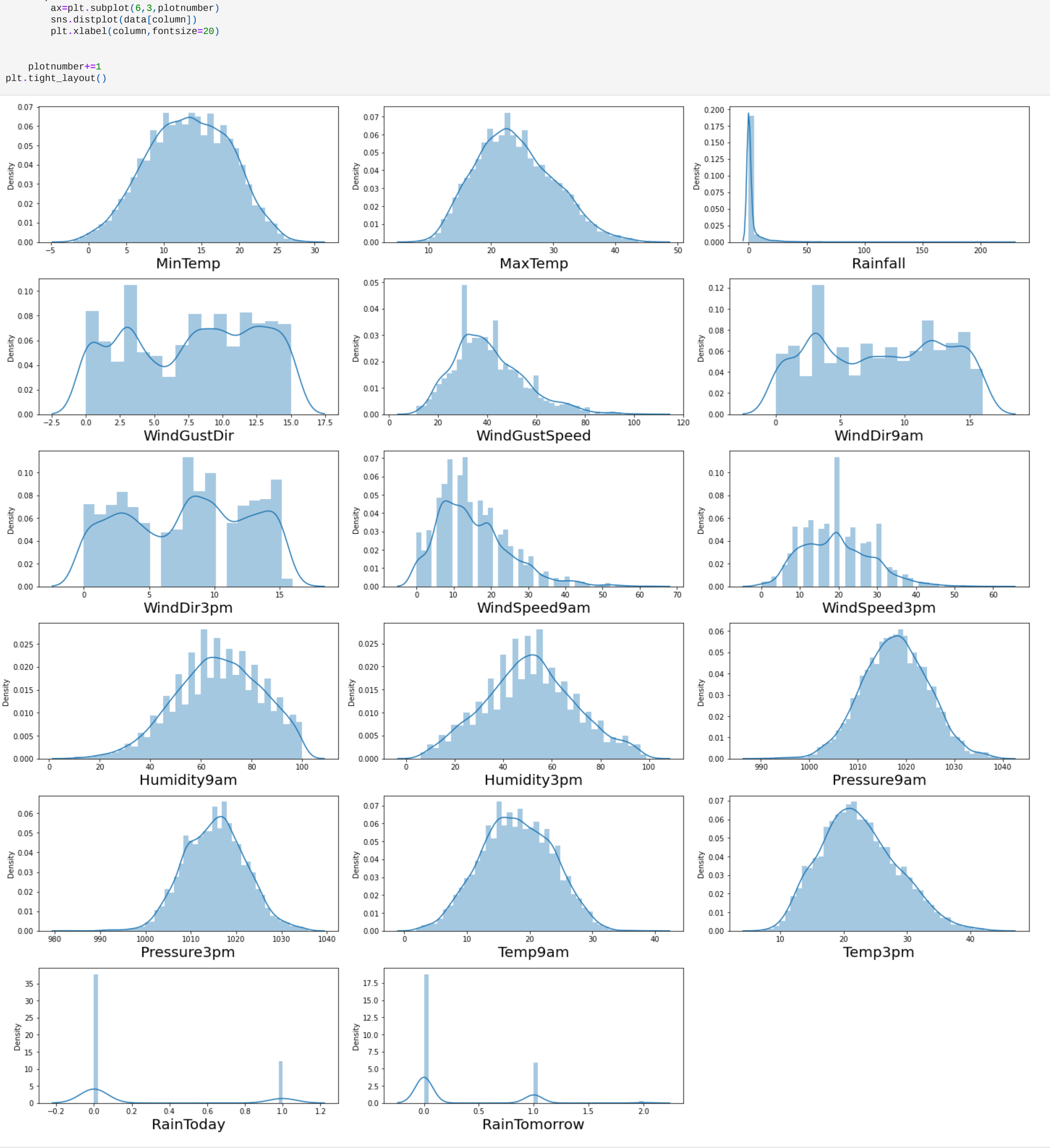
In [13]:
```python
data.isnull().sum()
```

Out[13]:
```
MinTemp          0
MaxTemp          0
Rainfall         0
WindGustDir      0
WindGustSpeed    0
WindDir9am       0
WindDir3pm       0
WindSpeed9am     0
WindSpeed3pm     0
Humidity9am      0
Humidity3pm      0
Pressure9am      0
Pressure3pm      0
Temp9am          0
Temp3pm          0
RainToday        0
RainTomorrow     0
dtype: int64
```

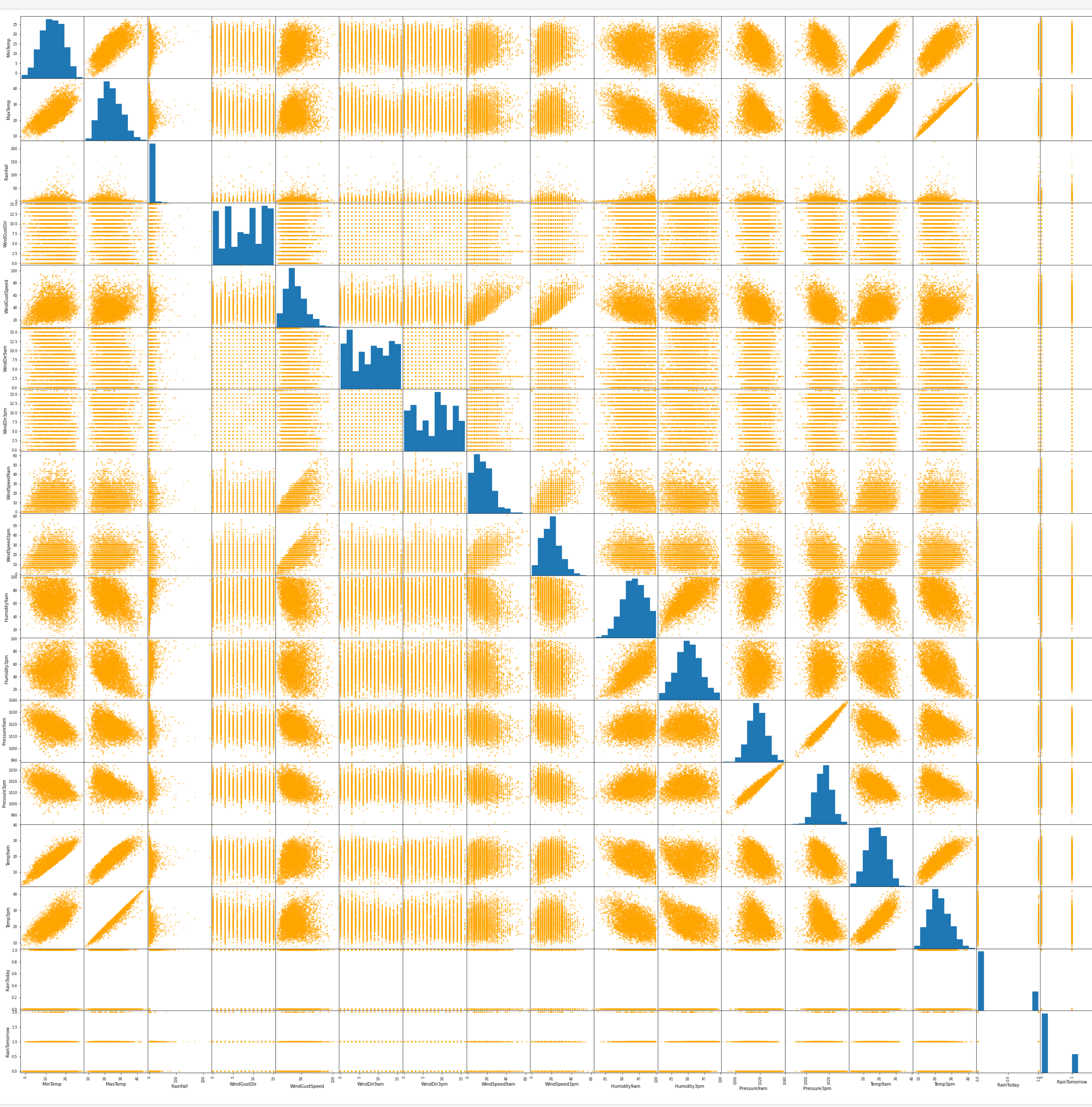
In [15]:
```python
plt.figure(figsize=(20,20))
plotnumber=1

for column in data:
    if plotnumber<=17:
        ax=plt.subplot(6,3,plotnumber)
        sns.distplot(data[column])
        plt.xlabel(column,fontsize=20)


    plotnumber+=1
plt.tight_layout()
```



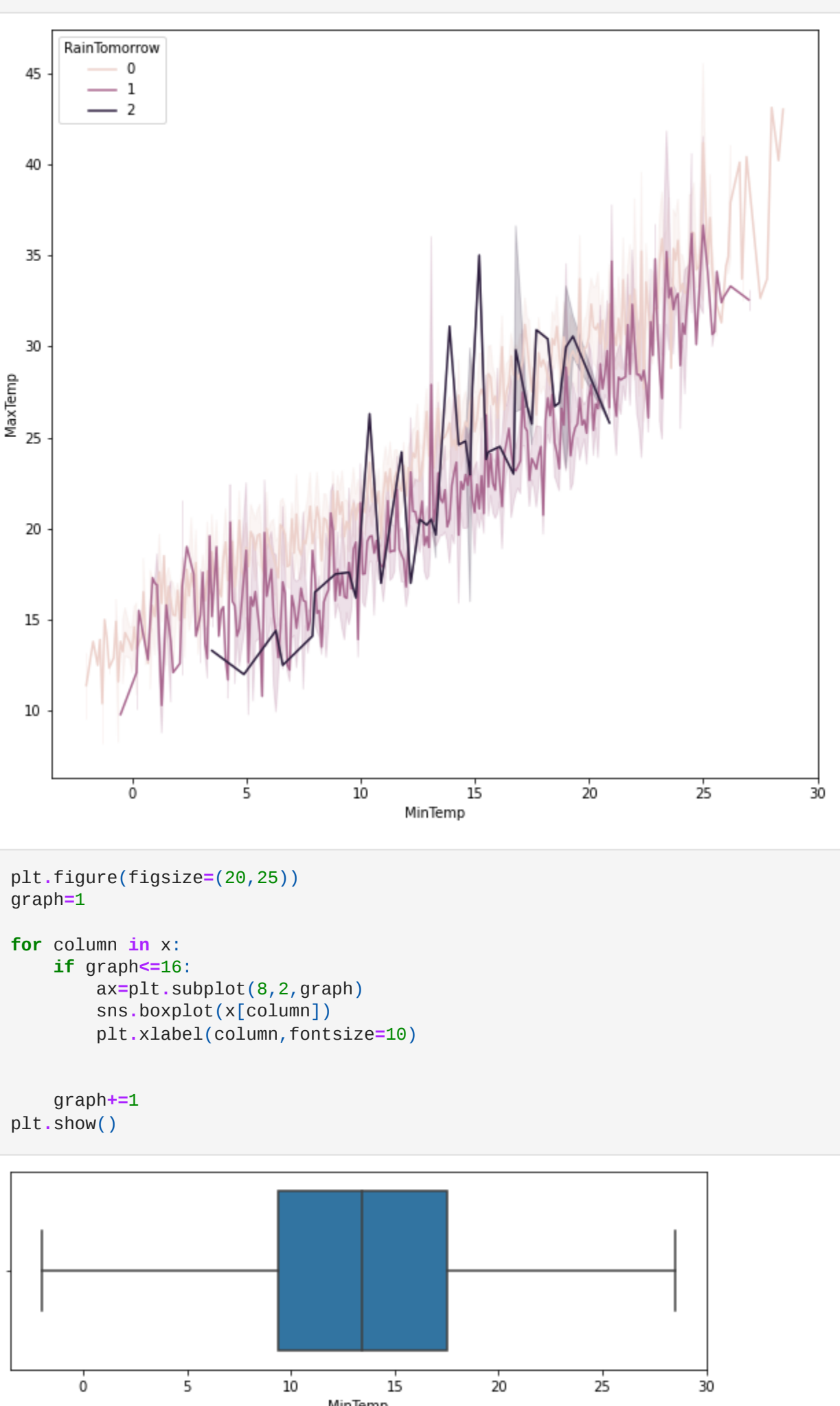
In [18]:
```python
scatter_matrix(data , figsize=(45,45) ,diagonal="hist" ,color="orange")
plt.show()
```



In [34]:
```python
from sklearn.metrics import classification_report
import statsmodels.formula.api as smf
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,confusion_matrix,roc_curve,roc_auc_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
```

In [20]:
```python
x =data.drop(['RainTomorrow'], axis =1)
y =data['RainTomorrow']
```

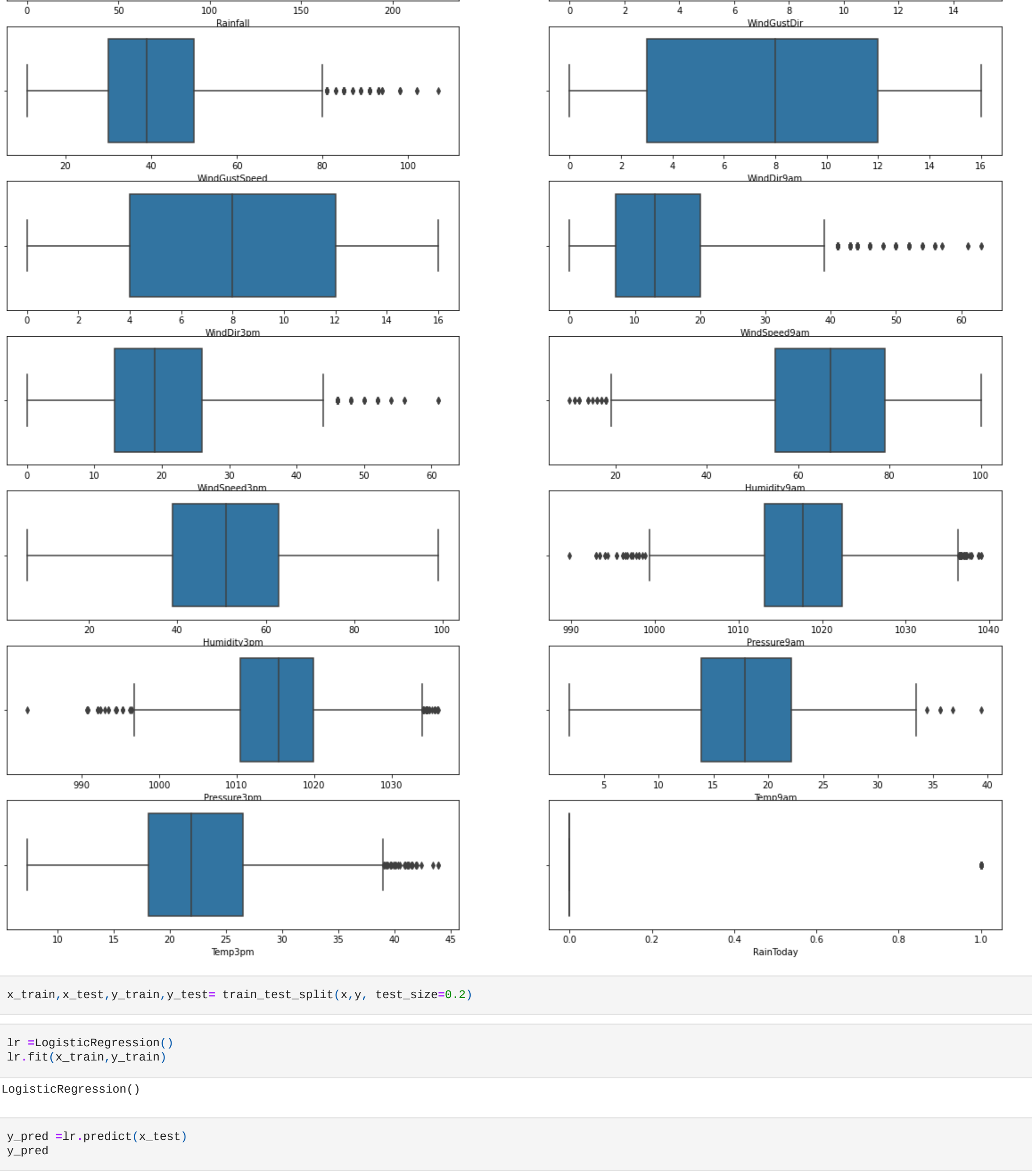
In [22]:
```python
plt.figure(figsize=(10,10))
sns.lineplot( x ="MinTemp" , y="MaxTemp" , hue ="RainTomorrow" , data =data)
plt.show()
```



In [23]:
```python
plt.figure(figsize=(20,25))
graph=1

for column in x:
    if graph<=16:
        ax=plt.subplot(8,2,graph)
        sns.boxplot(x[column])
        plt.xlabel(column,fontsize=10)


    graph+=1
plt.show()
```



In [24]:
```python
x_train,x_test,y_train,y_test= train_test_split(x,y, test_size=0.2)
```

In [25]:
```python
lr =LogisticRegression()
lr.fit(x_train,y_train)
```

Out[25]:
```
LogisticRegression()
```

In [26]:
```python
y_pred =lr.predict(x_test)
y_pred
```

Out[26]:
```
array([1, 0, 0, ..., 0, 0, 1])
```

In [29]:
```python
print("Accuracy =",accuracy_score(y_test,y_pred))
print("Confusion matrix =",confusion_matrix(y_test,y_pred))
```

```
Accuracy = 0.8303571428571429
Confusion matrix = [[946  65   0]
 [141 170   0]
 [ 19   3   0]]
```

In [30]:
```python
# classifier report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.86      0.94      0.89      1011
           1       0.71      0.55      0.62       311
           2       0.00      0.00      0.00        22

    accuracy                           0.83      1344
   macro avg       0.52      0.49      0.50      1344
weighted avg       0.81      0.83      0.82      1344
```

In [38]:
```python
# Decision Tree Classifier
dtc =DecisionTreeClassifier()
dtc.fit(x_train,y_train)
```

Out[38]:
```
DecisionTreeClassifier()
```

In [39]:
```python
y_predi =dtc.predict(x_test)
y_predi
```

Out[39]:
```
array([1, 0, 0, ..., 0, 0, 0])
```

In [40]:
```python
print("Accuracy =",accuracy_score(y_test,y_predi))
print("Confusion matrix =",confusion_matrix(y_test,y_predi))
```

```
Accuracy = 0.8683035714285714
Confusion matrix = [[916  89   6]
 [ 74 236   1]
 [  3   4  15]]
```

In [41]:
```python
print(classification_report(y_test,y_predi))
```

```
              precision    recall  f1-score   support

           0       0.92      0.91      0.91      1011
           1       0.72      0.76      0.74       311
           2       0.68      0.68      0.68        22

    accuracy                           0.87      1344
   macro avg       0.77      0.78      0.78      1344
weighted avg       0.87      0.87      0.87      1344
```

In [ ]:

In [42]:
```python
# Random Forest Classifier
rf =RandomForestClassifier()
rf.fit(x_train,y_train)
```

Out[42]:
```
RandomForestClassifier()
```

In [43]:
```python
y_predict =rf.predict(x_test)
y_predict
```

Out[43]:
```
array([1, 0, 0, ..., 0, 0, 1])
```

In [45]:
```python
print("Accuracy =",accuracy_score(y_test,y_predict))
print("Confusion matrix =",confusion_matrix(y_test,y_predict))
```

```
Accuracy = 0.9055059523809523
Confusion matrix = [[978  33   0]
 [ 87 224   0]
 [  5   2  15]]
```

In [46]:
```python
print(classification_report(y_test,y_predict))
```

```
              precision    recall  f1-score   support

           0       0.91      0.97      0.94      1011
           1       0.86      0.72      0.79       311
           2       1.00      0.68      0.81        22

    accuracy                           0.91      1344
   macro avg       0.93      0.79      0.85      1344
weighted avg       0.90      0.91      0.90      1344
```

In [ ]: