In [1]:

```python
#BCA 46 10 71
#importing of dataset
import pandas as pd
read_data = pd.read_csv('rainfall_weather_dataset.csv')
```

In [2]:

```python
#BCA 46 10 71
read_data['Date'] = pd.to_datetime(read_data['Date'])
read_data['year'] = read_data['Date'].dt.year
read_data['month'] = read_data['Date'].dt.month
read_data['day'] = read_data['Date'].dt.day
read_data.drop(['Date'], axis = 1,inplace=True)
read_data.head()
```

Out[2]:

	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	...	Cloud9am	Cloud3pm	Temp9am	Temp3pm	RainToday	RISK_MM	Rai
0	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	WNW	...	8.0	NaN	16.9	21.8	No	0.0	
1	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	WSW	...	NaN	NaN	17.2	24.3	No	0.0	
2	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	WSW	...	NaN	2.0	21.0	23.2	No	0.0	
3	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	E	...	NaN	NaN	18.1	26.5	No	1.0	
4	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	NW	...	7.0	8.0	17.8	29.7	No	0.2	

5 rows × 26 columns

In [3]:

```python
read_data.head()
```

Out[3]:

	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	...	Cloud9am	Cloud3pm	Temp9am	Temp3pm	RainToday	RISK_MM	Rai
0	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	WNW	...	8.0	NaN	16.9	21.8	No	0.0	
1	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	WSW	...	NaN	NaN	17.2	24.3	No	0.0	
2	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	WSW	...	NaN	2.0	21.0	23.2	No	0.0	
3	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	E	...	NaN	NaN	18.1	26.5	No	1.0	
4	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	NW	...	7.0	8.0	17.8	29.7	No	0.2	

5 rows × 26 columns

In [4]:

```python
read_data.shape
```

Out[4]:

```
(142193, 26)
```

In [5]:

```python
read_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 142193 entries, 0 to 142192
Data columns (total 26 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Location       142193 non-null  object
 1   MinTemp        141556 non-null  float64
 2   MaxTemp        141871 non-null  float64
 3   Rainfall       140787 non-null  float64
 4   Evaporation    81350 non-null   float64
 5   Sunshine       74377 non-null   float64
 6   WindGustDir    132863 non-null  object
 7   WindGustSpeed  132923 non-null  float64
 8   WindDir9am     132180 non-null  object
 9   WindDir3pm     138415 non-null  object
 10  WindSpeed9am   140845 non-null  float64
 11  WindSpeed3pm   139563 non-null  float64
 12  Humidity9am    140419 non-null  float64
 13  Humidity3pm    138583 non-null  float64
 14  Pressure9am    128179 non-null  float64
 15  Pressure3pm    128212 non-null  float64
 16  Cloud9am       88536 non-null   float64
 17  Cloud3pm       85099 non-null   float64
 18  Temp9am        141289 non-null  float64
 19  Temp3pm        139467 non-null  float64
 20  RainToday      140787 non-null  object
 21  RISK_MM        142193 non-null  float64
 22  RainTomorrow   142193 non-null  object
 23  year           142193 non-null  int64
 24  month          142193 non-null  int64
 25  day            142193 non-null  int64
dtypes: float64(17), int64(3), object(6)
memory usage: 28.2+ MB
```

In [6]:

```python
#BCA 46 10 71
#REPLACING VALUES
read_data['RainToday'].replace({'No': 0, 'Yes': 1},inplace = True)
read_data['RainTomorrow'].replace({'No': 0, 'Yes': 1},inplace = True)
read_data.head()
```

Out[6]:

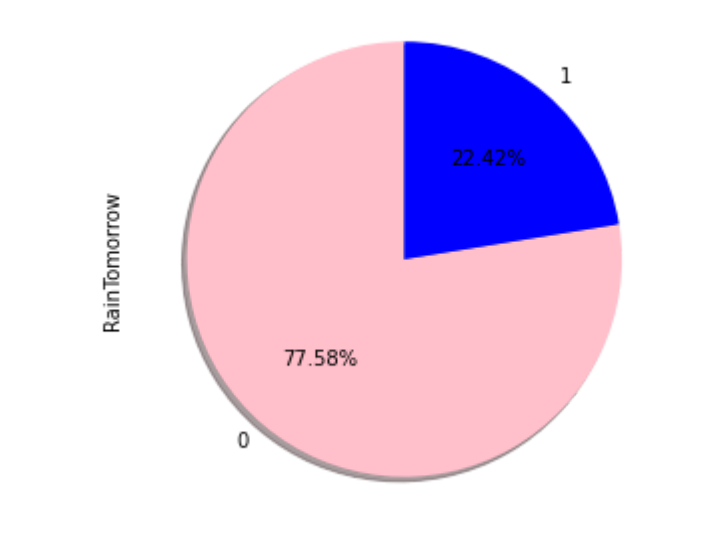
	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	...	Cloud9am	Cloud3pm	Temp9am	Temp3pm	RainToday	RISK_MM	Rai
0	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	WNW	...	8.0	NaN	16.9	21.8	0.0	0.0	
1	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	WSW	...	NaN	NaN	17.2	24.3	0.0	0.0	
2	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	WSW	...	NaN	2.0	21.0	23.2	0.0	0.0	
3	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	E	...	NaN	NaN	18.1	26.5	0.0	1.0	
4	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	NW	...	7.0	8.0	17.8	29.7	0.0	0.2	

5 rows × 26 columns

In [7]:

```python
#BCA 46 10 71
#CHECKING FOR IMBALANCED DATA
import matplotlib.pyplot as plt
plt.figure(figsize = (15,5))
read_data.RainTomorrow.value_counts(normalize = True).plot.pie(colors= ['pink','blue'],
                                                                shadow=True,autopct='%1.2f%%', startangle=90)
plt.title('Imbalanced Dataset-> Of RainTommorow Indicates ( No(0) and Yes(1) )')
plt.show()
```

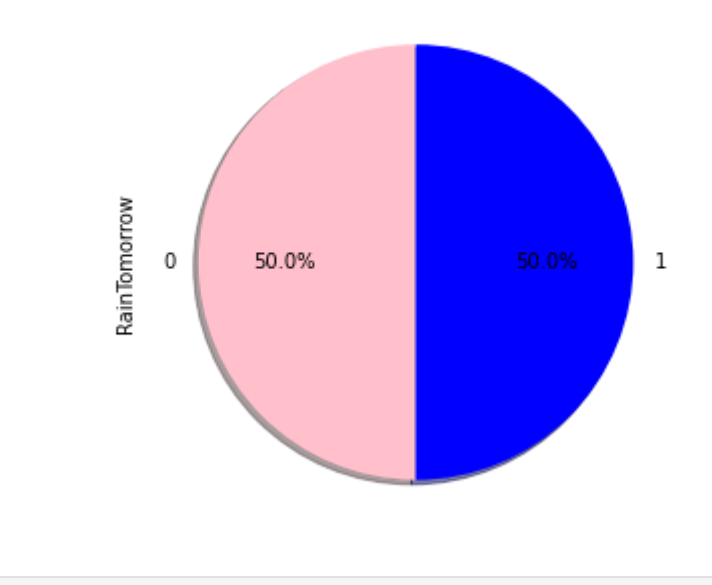
Imbalanced Dataset-> Of RainTommorow Indicates (No(0) and Yes(1))



In [8]:

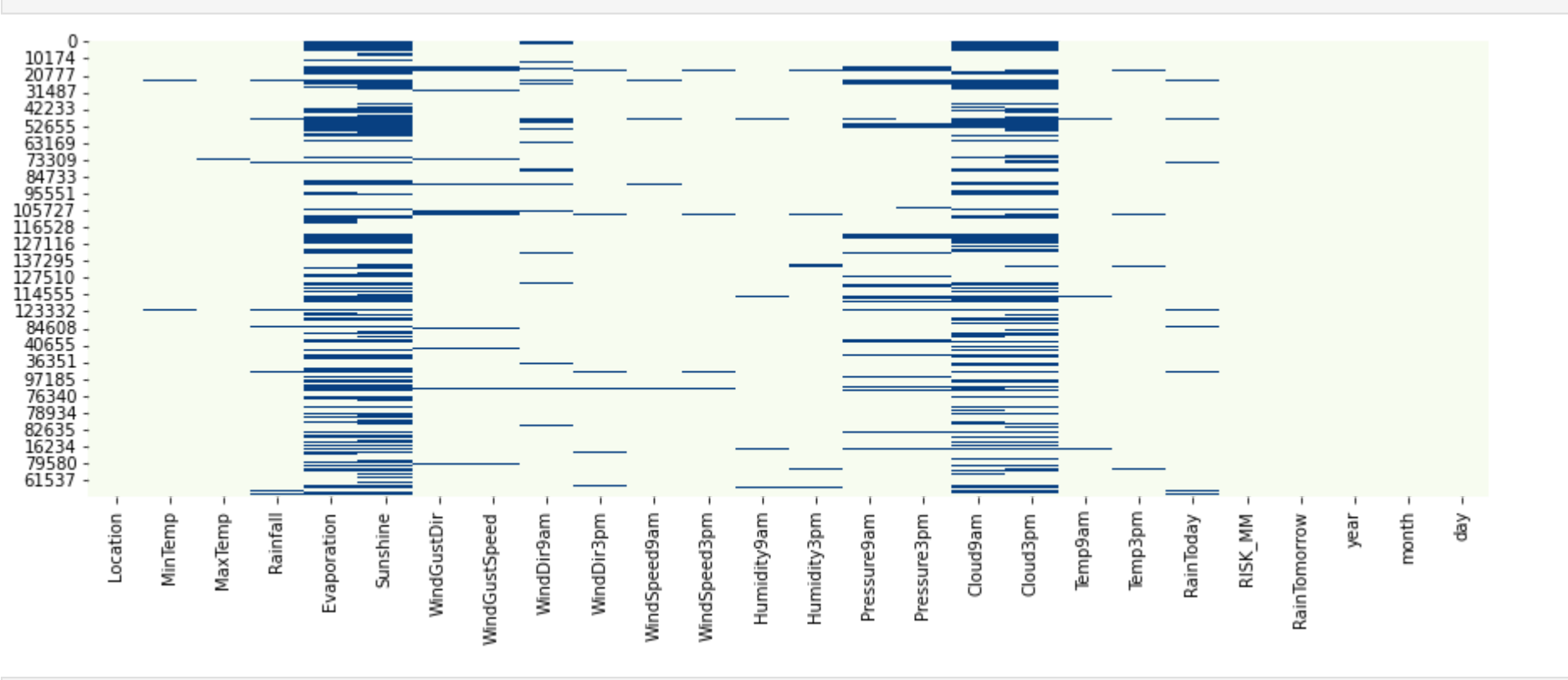
```python
#BCA 46 10 71
#HANDLING IMBALANCED DATA
from sklearn.utils import resample

no = read_data[read_data.RainTomorrow == 0]
yes = read_data[read_data.RainTomorrow == 1]
yes_os = resample(yes, replace=True, n_samples=len(no), random_state=42)
os = pd.concat([no, yes_os])

plt.figure(figsize = (15,5))
os.RainTomorrow.value_counts(normalize = True).plot.pie(colors= ['pink','blue'],shadow=True,autopct='%1.1f%%', startangle=90)
plt.title('Balanced Dataset -> After oversampling RainTommorow Indicates ( No(0) and Yes(1) )')
plt.show()
```

Balanced Dataset -> After oversampling RainTommorow Indicates (No(0) and Yes(1))



In [9]:

```python
#BCA 46 10 71
#MISSING DATA PATTERN
import seaborn as sns
plt.figure(figsize = (15,5))
sns.heatmap(os.isnull(), cbar=False, cmap='GnBu')
plt.show()
```



In [10]:

```python
#BCA 46 10 71
total = os.isnull().sum().sort_values(ascending=False)
percent = (os.isnull().sum()/os.isnull().count()).sort_values(ascending=False)
missing = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
missing.head()
```

Out[10]:

	Total	Percent
Sunshine	104711	0.474596
Evaporation	95325	0.432054
Cloud3pm	85658	0.388239
Cloud9am	81388	0.368886
Pressure9am	21620	0.097991

In [11]:

```python
#BCA 46 10 71
os.select_dtypes(include=['object']).columns
```

Out[11]:

```
Index(['Location', 'WindGustDir', 'WindDir9am', 'WindDir3pm'], dtype='object')
```

In [12]:

```python
#BCA 46 10 71
#IMPORT CATEGORICAL VALUE WITH MODE
os['Location'] = os['Location'].fillna(os['Location'].mode()[0])
os['WindGustDir'] = os['WindGustDir'].fillna(os['WindGustDir'].mode()[0])
os['WindDir9am'] = os['WindDir9am'].fillna(os['WindDir9am'].mode()[0])
os['WindDir3pm'] = os['WindDir3pm'].fillna(os['WindDir3pm'].mode()[0])
```

In [13]:

```python
#BCA 46 10 71
#CONVERT CATEGORICAL TO CONTINUOUS FEATURES
from sklearn.preprocessing import LabelEncoder
lencoders = {}
for col in os.select_dtypes(include=['object']).columns:
    lencoders[col] = LabelEncoder()
    os[col] = lencoders[col].fit_transform(os[col])
```

In [14]:

```python
#BCA 46 10 71
import warnings
warnings.filterwarnings("ignore")
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
MiceImputed = os.copy(deep=True)
mice_imputer = IterativeImputer()
MiceImputed.iloc[:, :] = mice_imputer.fit_transform(os)
```

In [15]:

```python
MiceImputed.head()
```

Out[15]:

	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	...	Cloud9am	Cloud3pm	Temp9am	Temp3pm	RainToday	RISK_MM	Rai
0	2.0	13.4	22.9	0.6	6.083728	7.574125	13.0	44.0	13.0	14.0	...	8.000000	4.784585	16.9	21.8	0.0	0.0	
1	2.0	7.4	25.1	0.0	5.976502	11.590325	14.0	44.0	6.0	15.0	...	1.886788	2.618932	17.2	24.3	0.0	0.0	
2	2.0	12.9	25.7	0.0	8.071945	12.385405	15.0	46.0	13.0	15.0	...	1.970936	2.000000	21.0	23.2	0.0	0.0	
3	2.0	9.2	28.0	0.0	6.298094	11.862159	4.0	24.0	9.0	0.0	...	1.371664	2.121637	18.1	26.5	0.0	1.0	
4	2.0	17.5	32.3	1.0	7.390434	6.157304	13.0	41.0	1.0	7.0	...	7.000000	8.000000	17.8	29.7	0.0	0.2	

5 rows × 26 columns

In [16]:

```python
MiceImputed.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 220632 entries, 0 to 23751
Data columns (total 26 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Location       220632 non-null  float64
 1   MinTemp        220632 non-null  float64
 2   MaxTemp        220632 non-null  float64
 3   Rainfall       220632 non-null  float64
 4   Evaporation    220632 non-null  float64
 5   Sunshine       220632 non-null  float64
 6   WindGustDir    220632 non-null  float64
 7   WindGustSpeed  220632 non-null  float64
 8   WindDir9am     220632 non-null  float64
 9   WindDir3pm     220632 non-null  float64
 10  WindSpeed9am   220632 non-null  float64
 11  WindSpeed3pm   220632 non-null  float64
 12  Humidity9am    220632 non-null  float64
 13  Humidity3pm    220632 non-null  float64
 14  Pressure9am    220632 non-null  float64
 15  Pressure3pm    220632 non-null  float64
 16  Cloud9am       220632 non-null  float64
 17  Cloud3pm       220632 non-null  float64
 18  Temp9am        220632 non-null  float64
 19  Temp3pm        220632 non-null  float64
 20  RainToday      220632 non-null  float64
 21  RISK_MM        220632 non-null  float64
 22  RainTomorrow   220632 non-null  float64
 23  year           220632 non-null  float64
 24  month          220632 non-null  float64
 25  day            220632 non-null  float64
dtypes: float64(26)
memory usage: 45.4 MB
```

In [17]:

```python
MiceImputed.isna()
```

Out[17]:

	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	...	Cloud9am	Cloud3pm	Temp9am	Temp3pm	RainToday	RISK_MM
0	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16163	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
44739	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
10169	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
6662	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False
23751	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False

220632 rows × 26 columns

In [18]:

```python
#BCA 46 10 71
# DETECTING OUTLIERS WITH IQR
Q1 = MiceImputed.quantile(0.25)
Q3 = MiceImputed.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

```
Location         25.000000
MinTemp           9.200000
MaxTemp          10.200000
Rainfall          2.400000
Evaporation       4.117708
Sunshine          5.914120
WindGustDir       9.000000
WindGustSpeed    19.000000
WindDir9am        8.000000
WindDir3pm        8.000000
WindSpeed9am     13.000000
WindSpeed3pm     11.000000
Humidity9am      26.000000
Humidity3pm      30.000000
Pressure9am       8.700000
Pressure3pm       8.841193
Cloud9am          4.000000
Cloud3pm          3.692199
Temp9am           9.300000
Temp3pm           9.800000
RainToday         1.000000
RISK_MM           5.200000
RainTomorrow      1.000000
year              5.000000
month             5.000000
day              15.000000
dtype: float64
```

In [19]:

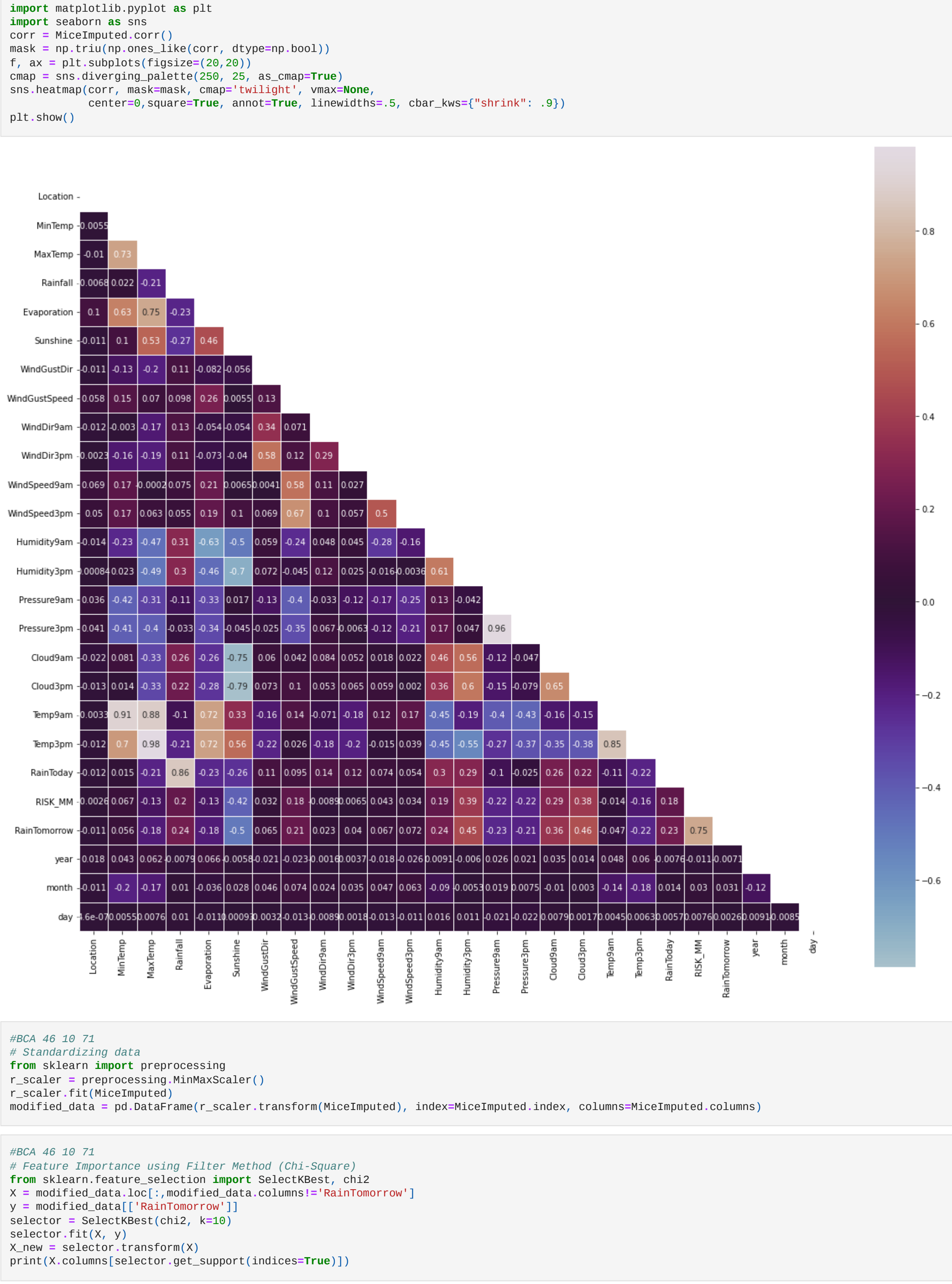
```python
#BCA 46 10 71
# REMOVING OUTLIERS FROM THE DATASET
MiceImputed = MiceImputed[~((MiceImputed < (Q1 - 1.5 * IQR)) |(MiceImputed > (Q3 + 1.5 * IQR))).any(axis=1)]
MiceImputed.shape
```

Out[19]:

```
(156799, 26)
```

In [20]:

```python
#BCA 46 10 71
#CORRELATION HEATMAP
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
corr = MiceImputed.corr()
mask = np.triu(np.ones_like(corr, dtype=np.bool))
f, ax = plt.subplots(figsize=(20,20))
cmap = sns.diverging_palette(250, 25, as_cmap=True)
sns.heatmap(corr, mask=mask, cmap='twilight', vmax=None,
            center=0,square=True, annot=True, linewidths=.5, cbar_kws={"shrink": .9})
plt.show()
```



In [21]:

```python
#BCA 46 10 71
# Standardizing data
from sklearn import preprocessing
r_scaler = preprocessing.MinMaxScaler()
r_scaler.fit(MiceImputed)
modified_data = pd.DataFrame(r_scaler.transform(MiceImputed), index=MiceImputed.index, columns=MiceImputed.columns)
```

In [22]:

```python
#BCA 46 10 71
# Feature Importance using Filter Method (Chi-Square)
from sklearn.feature_selection import SelectKBest, chi2
X = modified_data.loc[:,modified_data.columns!='RainTomorrow']
y = modified_data[['RainTomorrow']]
selector = SelectKBest(chi2, k=10)
selector.fit(X, y)
X_new = selector.transform(X)
print(X.columns[selector.get_support(indices=True)])
```

```
Index(['Sunshine', 'WindGustSpeed', 'Humidity9am', 'Humidity3pm',
       'Pressure9am', 'Cloud9am', 'Cloud3pm', 'Temp3pm', 'RainToday',
       'RISK_MM'],
      dtype='object')
```

In [23]:

```python
#BCA 46 10 71
from sklearn.feature_selection import SelectFromModel
from sklearn.ensemble import RandomForestClassifier as rf

X = MiceImputed.drop('RainTomorrow', axis=1)
y = MiceImputed['RainTomorrow']
selector = SelectFromModel(rf(n_estimators=100, random_state=0))
selector.fit(X, y)
support = selector.get_support()
features = X.loc[:,support].columns.tolist()
print(features)
print(rf(n_estimators=100, random_state=0).fit(X,y).feature_importances_)
```

```
['Sunshine', 'Humidity3pm', 'Cloud3pm', 'RISK_MM']
[0.00163243 0.00198108 0.00213486 0.00664109 0.00194937 0.04161249
 0.00125107 0.00681081 0.00120607 0.00134744 0.00094832 0.0013446
 0.00245146 0.04764479 0.00589437 0.0103183  0.0212991  0.06114829
 0.0019734  0.00343862 0.00333322 0.77088339 0.00078559 0.00092265
 0.00104717]
```

In [24]:

```python
#BCA 46 10 71
features = MiceImputed[['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustDir',
                       'WindGustSpeed', 'WindDir9am', 'WindDir3pm', 'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am',
                       'Humidity3pm', 'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am', 'Temp3pm',
                       'RainToday']]
target = MiceImputed['RainTomorrow']

# Split into test and train
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42,
                                                    shuffle=True, stratify=target)

# Normalize Features
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.fit_transform(X_test)
```

In [25]:

```python
#BCA 46 10 71
import time
from sklearn.metrics import accuracy_score, roc_auc_score, cohen_kappa_score, plot_confusion_matrix, roc_curve, classification_report
def run_model(model, X_train, y_train, X_test, y_test, verbose=True):
    t0=time.time()
    if verbose == False:
        model.fit(X_train,y_train, verbose=0)
    else:
        model.fit(X_train,y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    roc_auc = roc_auc_score(y_test, y_pred)
    coh_kap = cohen_kappa_score(y_test, y_pred)
    time_taken = time.time()-t0
    print("Accuracy = {}".format(accuracy))
    print("ROC Area under Curve = {}".format(roc_auc))
    print("Cohen's Kappa = {}".format(coh_kap))
    print("Time taken = {}".format(time_taken))
    print(classification_report(y_test,y_pred,digits=5))

    probs = model.predict_proba(X_test)
    probs = probs[:, 1]
    fper, tper, thresholds = roc_curve(y_test, probs)
     # plot_roc_cur(fper, tper)

     # plot_confusion_matrix(model, X_test, y_test,cmap=plt.cm.Blues, normalize = 'all')

    return model, accuracy, roc_auc, coh_kap, time_taken
```

In [26]:

```python
#BCA 46 10 71
#LOGISTIC REGRESSION
from sklearn.linear_model import LogisticRegression

params_lr = {'penalty': 'l1', 'solver':'liblinear'}

model_lr = LogisticRegression(**params_lr)
model_lr, accuracy_lr, roc_auc_lr, coh_kap_lr, tt_lr = run_model(model_lr, X_train, y_train, X_test, y_test)
```

```
Accuracy = 0.7888073979591836
ROC Area under Curve = 0.7690488393150476
Cohen's Kappa = 0.5480134594084004
Time taken = 7.134801387786865
              precision    recall  f1-score   support

         0.0    0.80711   0.85898   0.83224     19125
         1.0    0.75495   0.67912   0.71503     12235

    accuracy                        0.78881     31360
   macro avg    0.78103   0.76905   0.77363     31360
weighted avg    0.78676   0.78881   0.78651     31360
```

In [ ]: