Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №2-3
по дисциплине
«Технологии машинного обучения»
на тему

«Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных. Подготовка обучающей и тестовой выборки, кросс-валидация и подбор гиперпараметров на примере метода ближайших соседей»

Выполнила:
студент группы ИУ5-64б
Подопригорова Н. С.

Москва — 2020 г.

1. Лабораторная №2

Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных.

Задание:

1. Выбрать набор данных (датасет), содержащий категориальные признаки и пропуски в данных. Для выполнения следующих пунктов можно использовать несколько различных наборов данных (один для обработки пропусков, другой для категориальных признаков и т.д.)
2. Для выбранного датасета (датасетов) на основе материалов лекции решить следующие задачи:
   - обработку пропусков в данных;
   - кодирование категориальных признаков;
   - масштабирование данных.

2. Описание данных

- Date - Дата наблюдений
- Location - Название локации, в которой расположена метеорологическая станция
- MinTemp - Минимальная температура в градусах цельсия
- MaxTemp - Максимальная температура в градусах цельсия
- Rainfall - Количество осадков, зафиксированных за день в мм
- Evaporation - Так называемое "pan evaporation" класса A (мм) за 24 часа до 9 утра
- Sunshine - Число солнечных часов за день
- WindGustDir - направление самого сильного порыва ветра за последние 24 часа
- WindGustSpeed - скорость (км / ч) самого сильного порыва ветра за последние 24 часа
- WindDir9am - направление ветра в 9 утра

[1]:
```
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
import pandas as pd
import numpy as np
import seaborn as sns

import matplotlib.pyplot as plt
```

[2]:
```
data = pd.read_csv('weatherAUS.csv', parse_dates=['Date'])
```

[3]:
```
data.head()
```

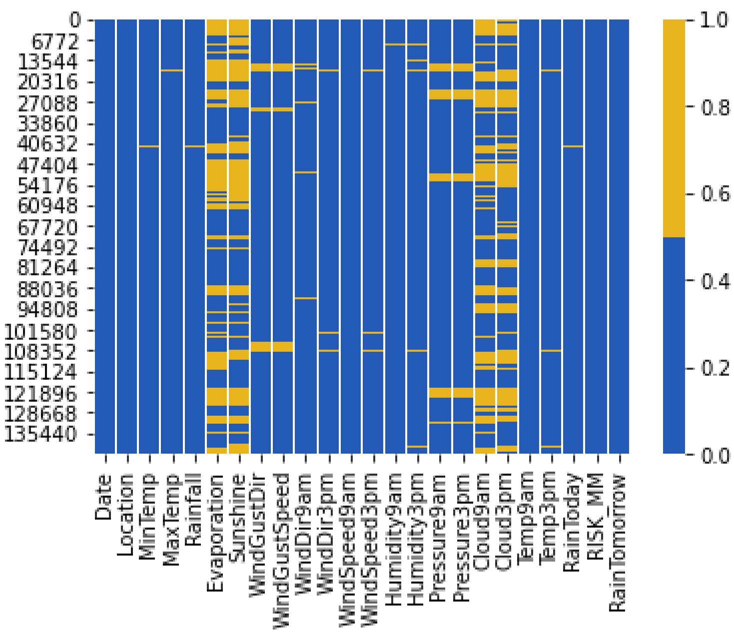
[3]:
```
        Date Location  MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine  \
0 2008-12-01   Albury     13.4     22.9       0.6          NaN       NaN
1 2008-12-02   Albury      7.4     25.1       0.0          NaN       NaN
2 2008-12-03   Albury     12.9     25.7       0.0          NaN       NaN
3 2008-12-04   Albury      9.2     28.0       0.0          NaN       NaN
4 2008-12-05   Albury     17.5     32.3       1.0          NaN       NaN
```

```
  WindGustDir  WindGustSpeed WindDir9am  ... Humidity3pm  Pressure9am  \
0           W           44.0          W  ...        22.0       1007.7
1         WNW           44.0        NNW  ...        25.0       1010.6
2         WSW           46.0          W  ...        30.0       1007.6
3          NE           24.0         SE  ...        16.0       1017.6
4           W           41.0        ENE  ...        33.0       1010.8

   Pressure3pm  Cloud9am  Cloud3pm  Temp9am  Temp3pm  RainToday  RISK_MM  \
0       1007.1       8.0       NaN     16.9     21.8         No      0.0
1       1007.8       NaN       NaN     17.2     24.3         No      0.0
2       1008.7       NaN       2.0     21.0     23.2         No      0.0
3       1012.8       NaN       NaN     18.1     26.5         No      1.0
4       1006.0       7.0       8.0     17.8     29.7         No      0.2

   RainTomorrow
0            No
1            No
2            No
3            No
4            No

[5 rows x 24 columns]
```

2.1. Обработка пропусков в данных

[4]:
```
cols = data.columns
#
#    –              ,      –
colours = ['#235AB5', '#E8B41E']
sns.heatmap(data[cols].isnull(), cmap=sns.color_palette(colours))
```

[4]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fab63b30400>
```



```
[63]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 142193 entries, 0 to 142192
Data columns (total 24 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           142193 non-null  datetime64[ns]
 1   Location       142193 non-null  object
 2   MinTemp        141556 non-null  float64
 3   MaxTemp        141871 non-null  float64
 4   Rainfall       140787 non-null  float64
 5   Evaporation    81350 non-null   float64
 6   Sunshine       74377 non-null   float64
 7   WindGustDir    132863 non-null  object
 8   WindGustSpeed  132923 non-null  float64
 9   WindDir9am     132180 non-null  object
 10  WindDir3pm     138415 non-null  object
 11  WindSpeed9am   140845 non-null  float64
 12  WindSpeed3pm   139563 non-null  float64
 13  Humidity9am    140419 non-null  float64
 14  Humidity3pm    138583 non-null  float64
 15  Pressure9am    128179 non-null  float64
 16  Pressure3pm    128212 non-null  float64
 17  Cloud9am       88536 non-null   float64
 18  Cloud3pm       85099 non-null   float64
 19  Temp9am        141289 non-null  float64
 20  Temp3pm        139467 non-null  float64
 21  RainToday      140787 non-null  object
 22  RISK_MM        142193 non-null  float64
 23  RainTomorrow   142193 non-null  object
dtypes: datetime64[ns](1), float64(17), object(6)
memory usage: 26.0+ MB
```

Рассмотрим числовые колонки с пропущенными значениями:

```
[13]: total_count = data.shape[0]
num_cols = []
for col in data.columns:
    #
    temp_null_count = data[data[col].isnull()].shape[0]
    dt = str(data[col].dtype)
    if temp_null_count>0 and (dt=='float64' or dt=='int64'):
        num_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('     {}.       {}.                    {}, {}%.'.format(col, dt,␣
 ↪temp_null_count, temp_perc))
```

```
    MinTemp.        float64.                    637, 0.45%.
    MaxTemp.        float64.                    322, 0.23%.
    Rainfall.        float64.                    1406, 0.99%.
    Evaporation.        float64.                    60843,
42.79%.
    Sunshine.        float64.                    67816, 47.69%.
    WindGustSpeed.        float64.                    9270,
6.52%.
    WindSpeed9am.        float64.                    1348,
0.95%.
    WindSpeed3pm.        float64.                    2630,
1.85%.
    Humidity9am.        float64.                    1774, 1.25%.
    Humidity3pm.        float64.                    3610, 2.54%.
    Pressure9am.        float64.                    14014,
9.86%.
    Pressure3pm.        float64.                    13981,
9.83%.
    Cloud9am.        float64.                    53657, 37.74%.
    Cloud3pm.        float64.                    57094, 40.15%.
    Temp9am.        float64.                    904, 0.64%.
    Temp3pm.        float64.                    2726, 1.92%.
```

[29]:

```
#
data_num = data[num_cols]
data_num
```

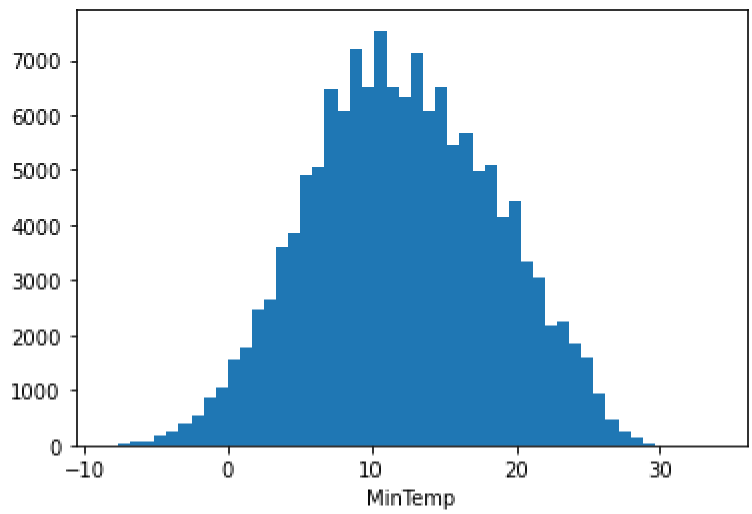
[29]:

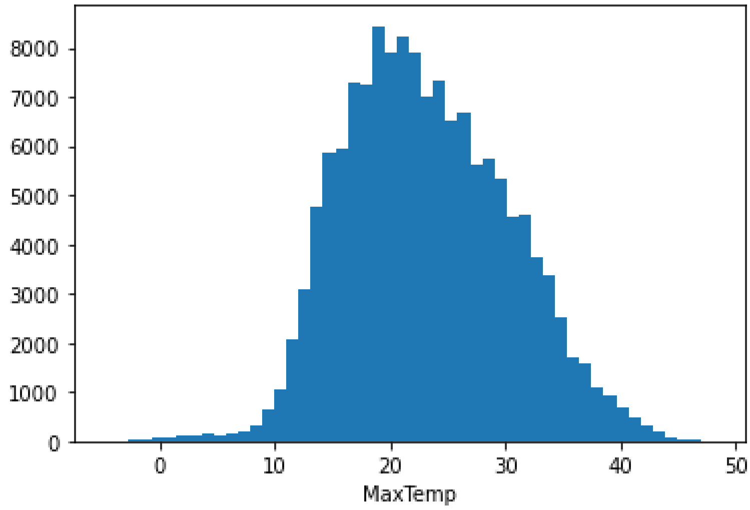
```
        MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine  WindGustSpeed  \
0          13.4     22.9       0.6          NaN       NaN           44.0
1           7.4     25.1       0.0          NaN       NaN           44.0
2          12.9     25.7       0.0          NaN       NaN           46.0
3           9.2     28.0       0.0          NaN       NaN           24.0
4          17.5     32.3       1.0          NaN       NaN           41.0
…             …        …         …            …         …              …
142188      3.5     21.8       0.0          NaN       NaN           31.0
142189      2.8     23.4       0.0          NaN       NaN           31.0
142190      3.6     25.3       0.0          NaN       NaN           22.0
142191      5.4     26.9       0.0          NaN       NaN           37.0
142192      7.8     27.0       0.0          NaN       NaN           28.0

        WindSpeed9am  WindSpeed3pm  Humidity9am  Humidity3pm  Pressure9am  \
0               20.0          24.0         71.0         22.0       1007.7
1                4.0          22.0         44.0         25.0       1010.6
2               19.0          26.0         38.0         30.0       1007.6
3               11.0           9.0         45.0         16.0       1017.6
4                7.0          20.0         82.0         33.0       1010.8
…                  …             …            …            …            …
142188          15.0          13.0         59.0         27.0       1024.7
142189          13.0          11.0         51.0         24.0       1024.6
142190          13.0           9.0         56.0         21.0       1023.5
```

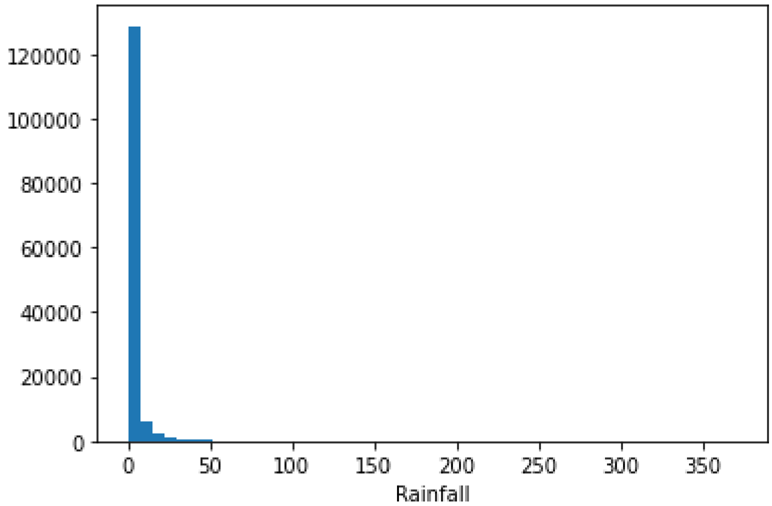
```
142191          9.0         9.0       53.0         24.0       1021.0
142192         13.0         7.0       51.0         24.0       1019.4

        Pressure3pm  Cloud9am  Cloud3pm  Temp9am  Temp3pm
0            1007.1       8.0       NaN     16.9     21.8
1            1007.8       NaN       NaN     17.2     24.3
2            1008.7       NaN       2.0     21.0     23.2
3            1012.8       NaN       NaN     18.1     26.5
4            1006.0       7.0       8.0     17.8     29.7
…               …       …       …       …       …
142188       1021.2       NaN       NaN      9.4     20.9
142189       1020.3       NaN       NaN     10.1     22.4
142190       1019.1       NaN       NaN     10.9     24.5
142191       1016.8       NaN       NaN     12.5     26.1
142192       1016.5       3.0       2.0     15.1     26.0

[142193 rows x 16 columns]
```

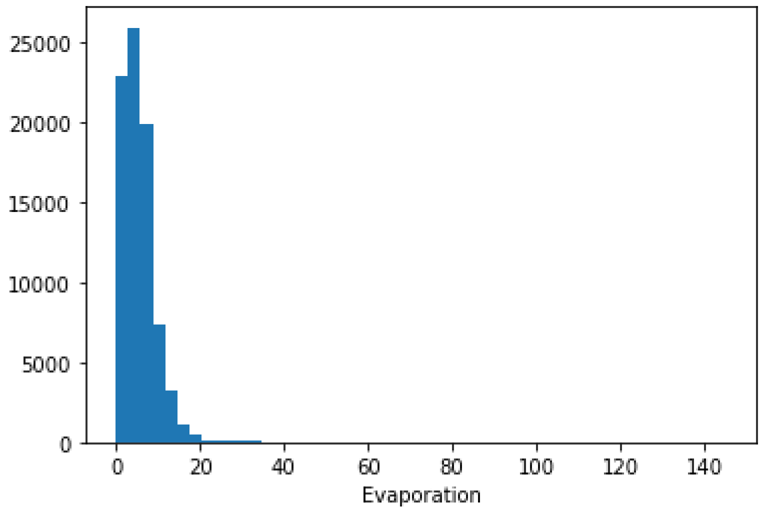
[124]:
```python
for col in data_num:
    plt.hist(data[col], 50)
    plt.xlabel(col)
    plt.show()
```

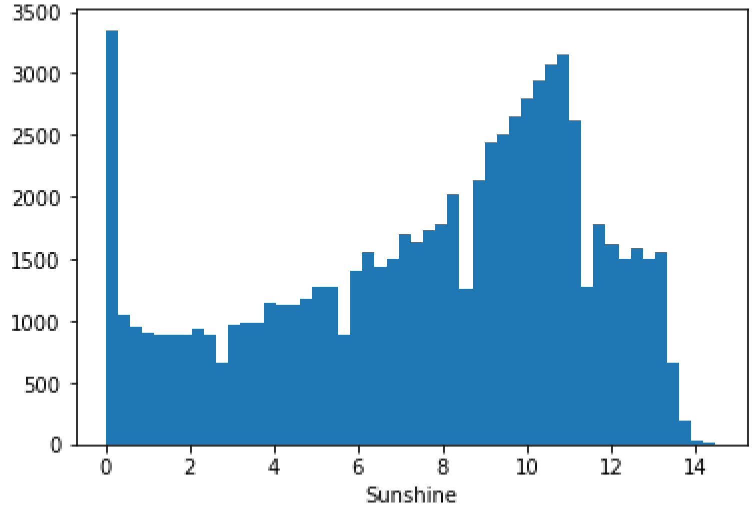
```
/Users/nonpenguin/anaconda3/lib/python3.8/site-
packages/numpy/lib/histograms.py:839: RuntimeWarning: invalid value␣
 ↪encountered
in greater_equal
  keep = (tmp_a >= first_edge)
/Users/nonpenguin/anaconda3/lib/python3.8/site-
packages/numpy/lib/histograms.py:840: RuntimeWarning: invalid value␣
 ↪encountered
in less_equal
  keep &= (tmp_a <= last_edge)
```

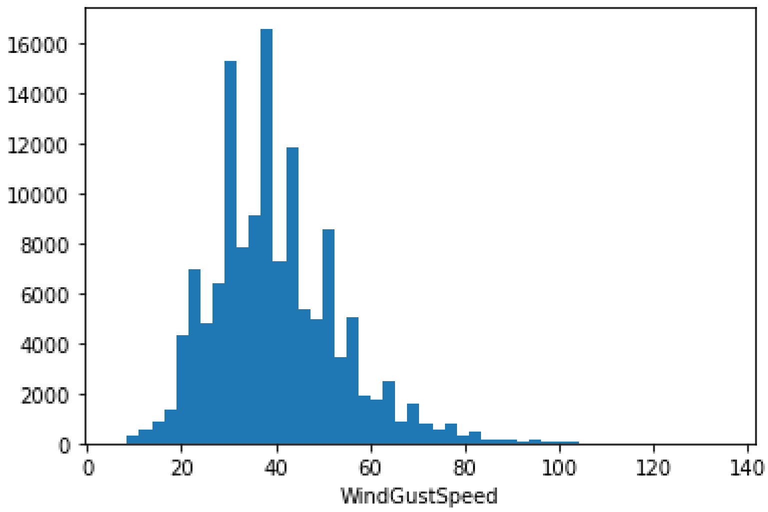


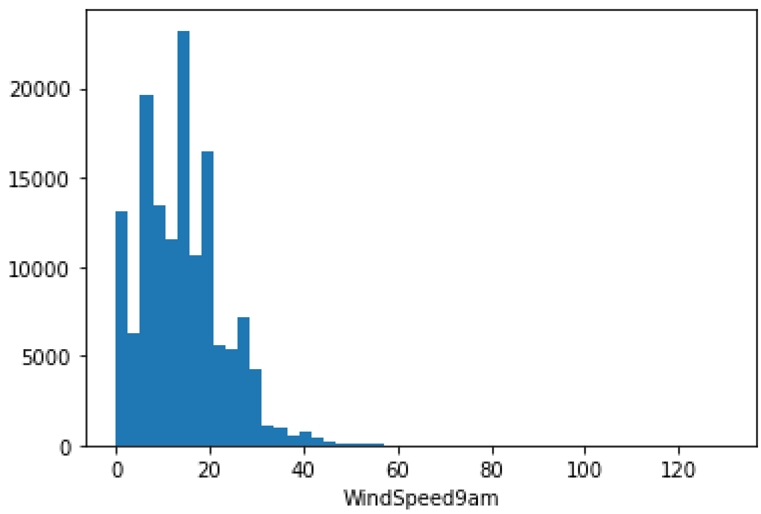

8000
7000
6000
5000
4000
3000
2000
1000
0
0
10
20
30
40
50
MaxTemp

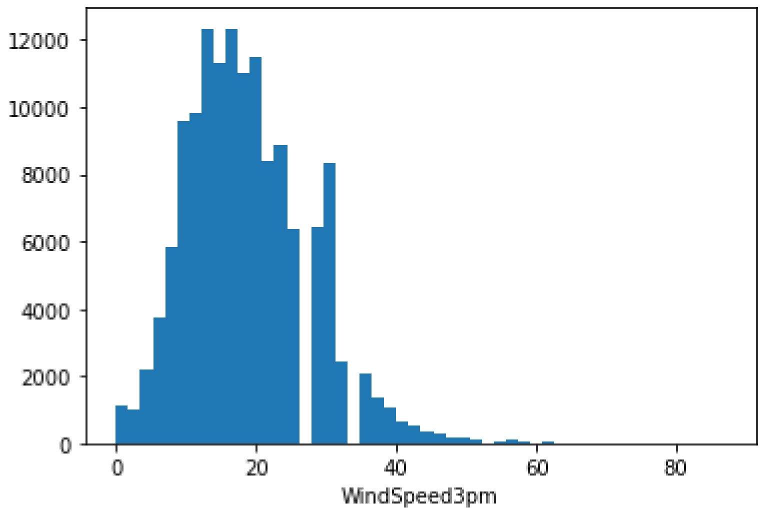

120000
100000
80000
60000
40000
20000
0
0
50
100
150
200
250
300
350
Rainfall

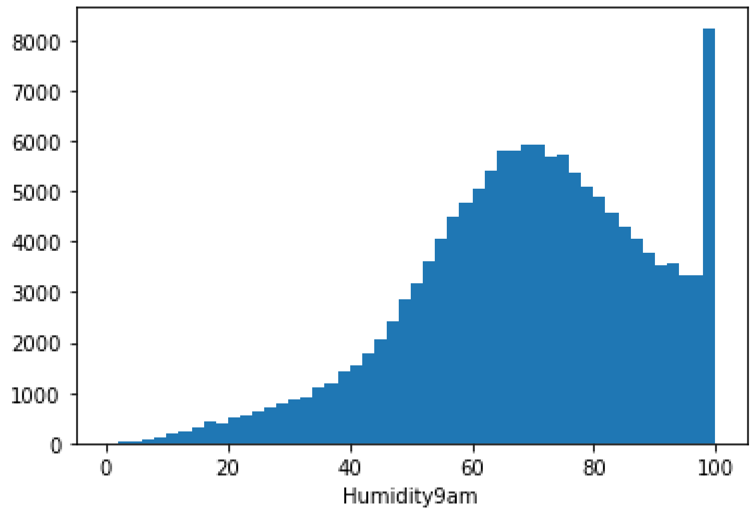

25000
20000
15000
10000
5000
0
0
20
40
60
80
100
120
140
Evaporation

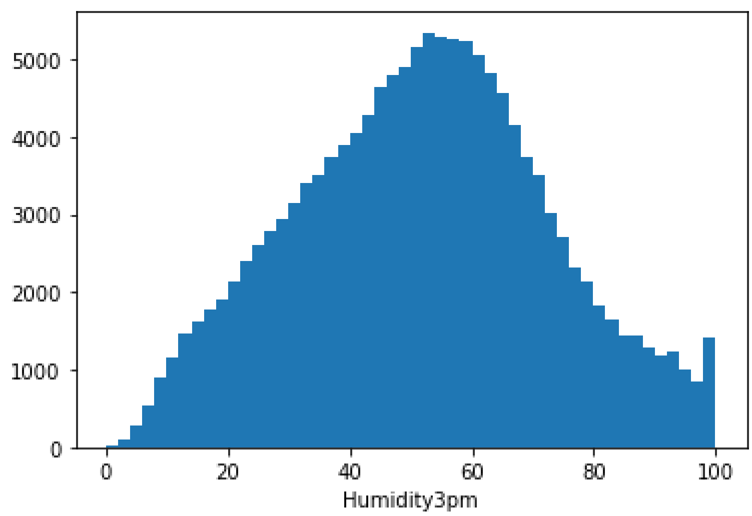

3500
3000
2500
2000
1500
1000
500
0
0
2
4
6
8
10
12
14
Sunshine

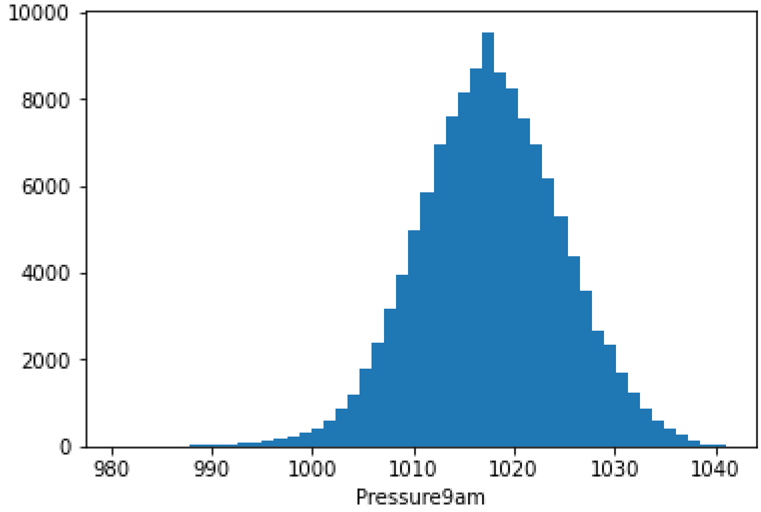

16000
14000
12000
10000
8000
6000
4000
2000
0
0
20
40
60
80
100
120
140
WindGustSpeed

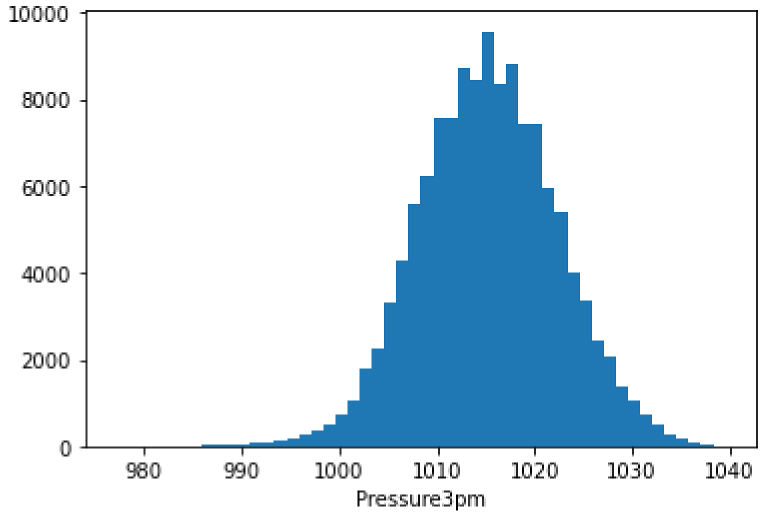

20000
15000
10000
5000
0
0
20
40
60
80
100
120
WindSpeed9am

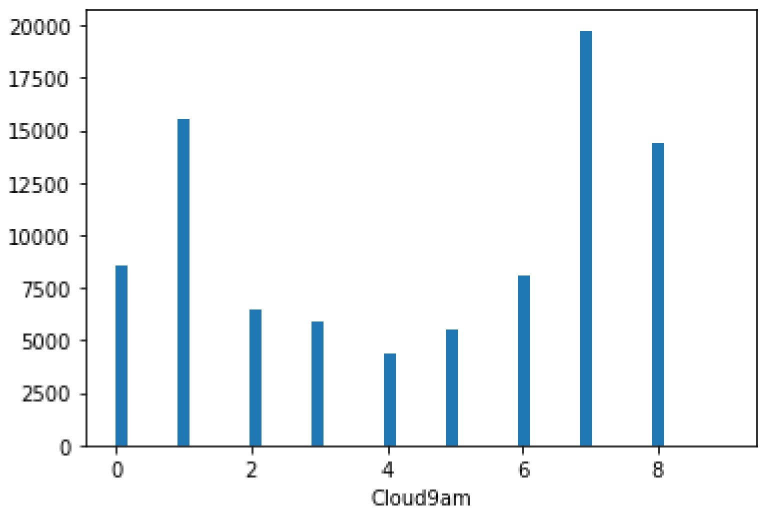

12000
10000
8000
6000
4000
2000
0
0
20
40
60
80
WindSpeed3pm

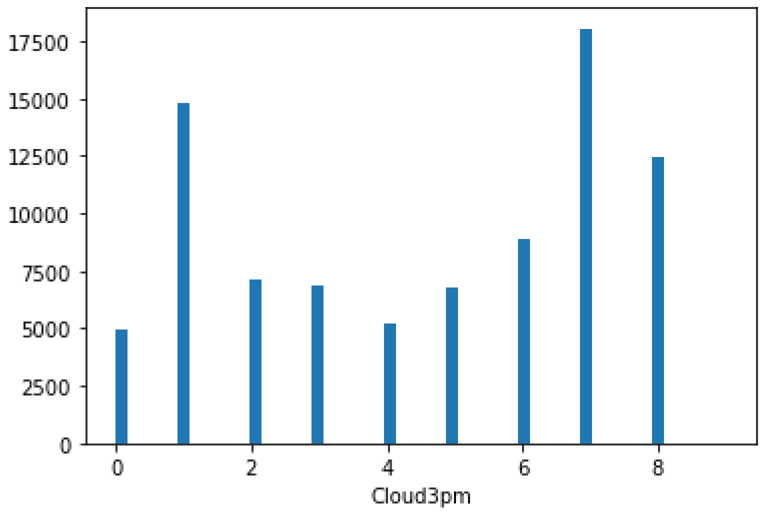

8000
7000
6000
5000
4000
3000
2000
1000
0
0
20
40
60
80
100
Humidity9am

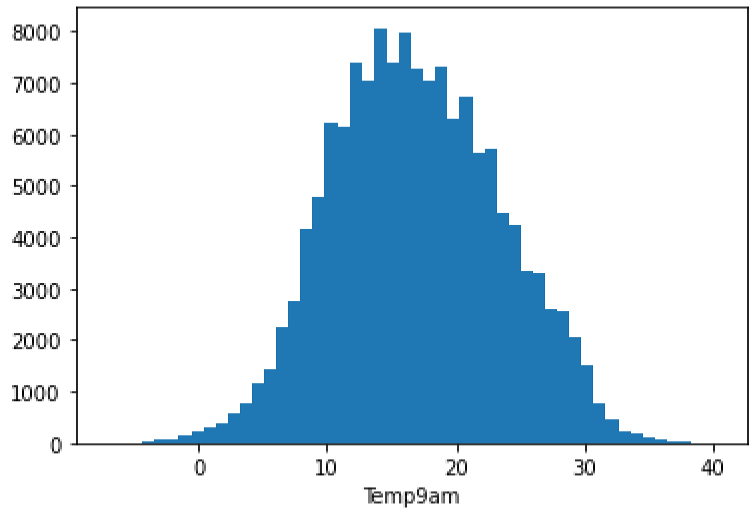

5000
4000
3000
2000
1000
0
0
20
40
60
80
100
Humidity3pm

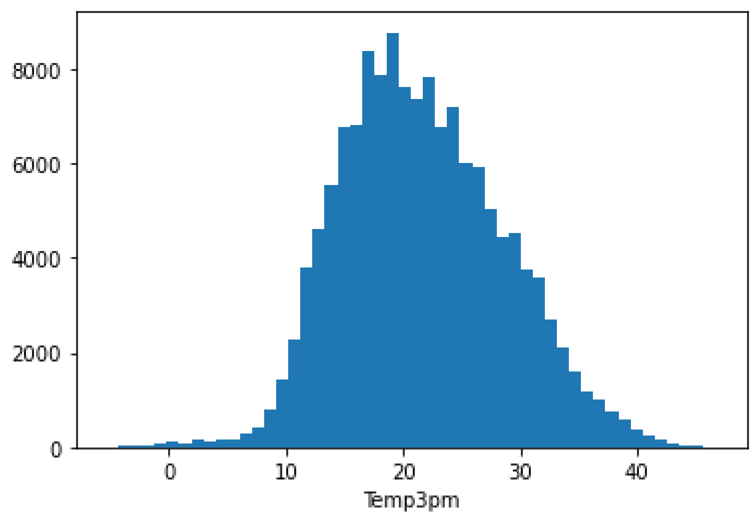

10000
8000
6000
4000
2000
0
980
990
1000
1010
1020
1030
1040
Pressure9am


10000
8000
6000
4000
2000
0
980
990
1000
1010
1020
1030
1040
Pressure3pm


20000
17500
15000
12500
10000
7500
5000
2500
0
0
2
4
6
8
Cloud9am


17500
15000
12500
10000
7500
5000
2500
0
0
2
4
6
8
Cloud3pm


8000
7000
6000
5000
4000
3000
2000
1000
0
0
10
20
30
40
Temp9am


8000
6000
4000
2000
0
0
10
20
30
40
Temp3pm

В столбце Evaporation 42.79% пропущенных данных, и его корреляция с целевым признаком низкая, так что легче всего этот столбец удалить.

[3]:
```
data = data.drop(['Evaporation'], axis = 1)
```

Все распределения, кроме Sunshine, одномодальные, так что будем использовать для заполнения пропусков моду (уточни этот моментик потом). Для Sunshine медиану.

[4]:
```
data['Sunshine'] = data['Sunshine'].fillna(data.median())
```

[5]:
```
data['Humidity9am'] = data['Humidity9am'].fillna(data['Humidity9am'].mode())
```

[6]:
```
data = data.fillna(data.mode())
```

Рассмотрим пропуски в категориальных данных

[152]:
```
#
#
cat_cols = []
for col in data.columns:
    #
    temp_null_count = data[data[col].isnull()].shape[0]
    dt = str(data[col].dtype)
    if temp_null_count>0 and (dt=='object'):
        cat_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('     {}.         {}.                    {}, {}%.'.format(col, dt,␣
 ↪temp_null_count, temp_perc))
```

```
    WindGustDir.        object.                   9330, 6.56%.
    WindDir9am.        object.                   10013, 7.04%.
    WindDir3pm.        object.                   3778, 2.66%.
    RainToday.        object.                   1406, 0.99%.
```

[153]:
```
for col in data[cat_cols]:
    print('     {}. {}'.format(col, data[col].unique()))
```

```
    WindGustDir. ['W' 'WNW' 'WSW' 'NE' 'NNW' 'N' 'NNE' 'SW' 'ENE' 'SSE' 'S'
'NW' 'SE' 'ESE'
 nan 'E' 'SSW']
    WindDir9am. ['W' 'NNW' 'SE' 'ENE' 'SW' 'SSE' 'S' 'NE' nan 'SSW' 'N'
'WSW' 'ESE' 'E'
 'NW' 'WNW' 'NNE']
    WindDir3pm. ['WNW' 'WSW' 'E' 'NW' 'W' 'SSE' 'ESE' 'ENE' 'NNW' 'SSW' 'SW'
'SE' 'N' 'S'
 'NNE' nan 'NE']
    RainToday. ['No' 'Yes' nan]
```

[7]:
```
data[:] = SimpleImputer(missing_values=np.nan, strategy='most_frequent').
 ↪fit_transform(data)
```

[110]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 142193 entries, 0 to 142192
Data columns (total 23 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           142193 non-null  datetime64[ns]
 1   Location       142193 non-null  object
 2   MinTemp        142193 non-null  float64
 3   MaxTemp        142193 non-null  float64
 4   Rainfall       142193 non-null  float64
 5   Sunshine       142193 non-null  float64
 6   WindGustDir    142193 non-null  object
 7   WindGustSpeed  142193 non-null  float64
 8   WindDir9am     142193 non-null  object
 9   WindDir3pm     142193 non-null  object
 10  WindSpeed9am   142193 non-null  float64
 11  WindSpeed3pm   142193 non-null  float64
 12  Humidity9am    142193 non-null  float64
 13  Humidity3pm    142193 non-null  float64
 14  Pressure9am    142193 non-null  float64
 15  Pressure3pm    142193 non-null  float64
 16  Cloud9am       142193 non-null  float64
 17  Cloud3pm       142193 non-null  float64
 18  Temp9am        142193 non-null  float64
 19  Temp3pm        142193 non-null  float64
 20  RainToday      142193 non-null  object
 21  RISK_MM        142193 non-null  float64
 22  RainTomorrow   142193 non-null  object
dtypes: datetime64[ns](1), float64(16), object(6)
memory usage: 25.0+ MB
```

[8]:
```
data.isnull().sum()
```

[8]:
```
Date             0
Location         0
MinTemp          0
MaxTemp          0
Rainfall         0
Sunshine         0
WindGustDir      0
WindGustSpeed    0
WindDir9am       0
WindDir3pm       0
WindSpeed9am     0
WindSpeed3pm     0
Humidity9am      0
Humidity3pm      0
Pressure9am      0
Pressure3pm      0
Cloud9am         0
```

```
Cloud3pm         0
Temp9am          0
Temp3pm          0
RainToday        0
RISK_MM          0
RainTomorrow     0
dtype: int64
```

2.2. Кодирование категориальных признаков

[8]:
```
data['RainToday'] = data['RainToday'].apply(lambda x: 1 if x == 'Yes' else␣
↪0)
data['RainTomorrow'] = data['RainTomorrow'].apply(lambda x: 1 if x == 'Yes'␣
↪else 0)
```

[136]:
```
data['Location'].unique()
```

[136]:
```
array(['Albury', 'BadgerysCreek', 'Cobar', 'CoffsHarbour', 'Moree',
       'Newcastle', 'NorahHead', 'NorfolkIsland', 'Penrith', 'Richmond',
       'Sydney', 'SydneyAirport', 'WaggaWagga', 'Williamtown',
       'Wollongong', 'Canberra', 'Tuggeranong', 'MountGinini', 'Ballarat',
       'Bendigo', 'Sale', 'MelbourneAirport', 'Melbourne', 'Mildura',
       'Nhil', 'Portland', 'Watsonia', 'Dartmoor', 'Brisbane', 'Cairns',
       'GoldCoast', 'Townsville', 'Adelaide', 'MountGambier', 'Nuriootpa',
       'Woomera', 'Albany', 'Witchcliffe', 'PearceRAAF', 'PerthAirport',
       'Perth', 'SalmonGums', 'Walpole', 'Hobart', 'Launceston',
       'AliceSprings', 'Darwin', 'Katherine', 'Uluru'], dtype=object)
```

Слишком много категорий Location для OneHotEncoder

[9]:
```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
data['Location'] = le.fit_transform(data['Location'])
```

[10]:
```
categorical = ['WindDir3pm','WindDir9am', 'WindGustDir']

data = pd.concat([data, pd.get_dummies(data[categorical],␣
↪columns=categorical, drop_first=True)],axis=1)
data.drop(categorical, axis=1, inplace=True)
```

[160]:
```
data
```

[160]:
```
          Date  Location  MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine ␣
↪\
0       733377         2     13.4     22.9       0.6          NaN       NaN
1       733378         2      7.4     25.1       0.0          NaN       NaN
2       733379         2     12.9     25.7       0.0          NaN       NaN
3       733380         2      9.2     28.0       0.0          NaN       NaN
4       733381         2     17.5     32.3       1.0          NaN       NaN
```

```
…        …        …       …       …       …          …         …
142188  736500        41     3.5    21.8       0.0           NaN        NaN
142189  736501        41     2.8    23.4       0.0           NaN        NaN
142190  736502        41     3.6    25.3       0.0           NaN        NaN
142191  736503        41     5.4    26.9       0.0           NaN        NaN
142192  736504        41     7.8    27.0       0.0           NaN        NaN

        WindGustSpeed  WindSpeed9am  WindSpeed3pm  …  WindGustDir_NNW  \
0                44.0          20.0          24.0  …                0
1                44.0           4.0          22.0  …                0
2                46.0          19.0          26.0  …                0
3                24.0          11.0           9.0  …                0
4                41.0           7.0          20.0  …                0
…                 …            …             …  …               …
142188           31.0          15.0          13.0  …                0
142189           31.0          13.0          11.0  …                0
142190           22.0          13.0           9.0  …                1
142191           37.0           9.0           9.0  …                0
142192           28.0          13.0           7.0  …                0

        WindGustDir_NW  WindGustDir_S  WindGustDir_SE  WindGustDir_SSE  \
0                    0              0               0                0
1                    0              0               0                0
2                    0              0               0                0
3                    0              0               0                0
4                    0              0               0                0
…                  …             …              …              …
142188               0              0               0                0
142189               0              0               0                0
142190               0              0               0                0
142191               0              0               0                0
142192               0              0               1                0

        WindGustDir_SSW  WindGustDir_SW  WindGustDir_W WindGustDir_WNW  \
0                     0               0              1               0
1                     0               0              0               1
2                     0               0              0               0
3                     0               0              0               0
4                     0               0              1               0
…                   …              …             …             …
142188                0               0              0               0
142189                0               0              0               0
142190                0               0              0               0
142191                0               0              0               0
142192                0               0              0               0

        WindGustDir_WSW
0                     0
1                     0
2                     1
```

```
3                      0
4                      0
...                ...
142188                 0
142189                 0
142190                 0
142191                 0
142192                 0

[142193 rows x 66 columns]
```

[11]:
```
data.shape
```

[11]:
```
(142193, 65)
```

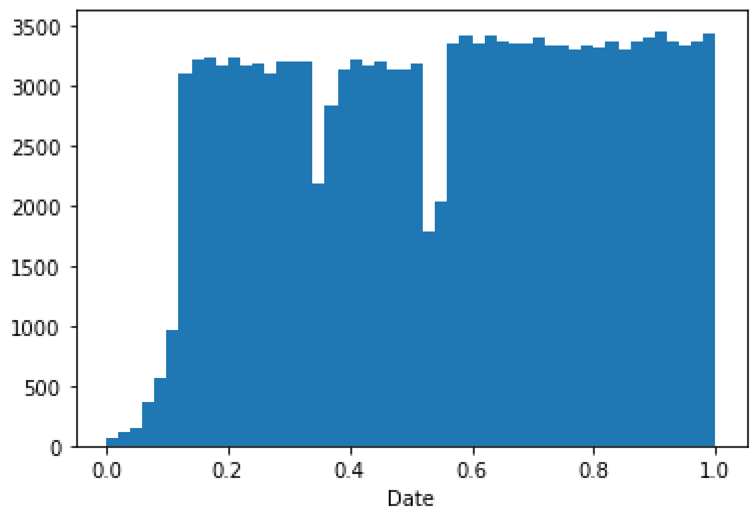
Преобразуем дату

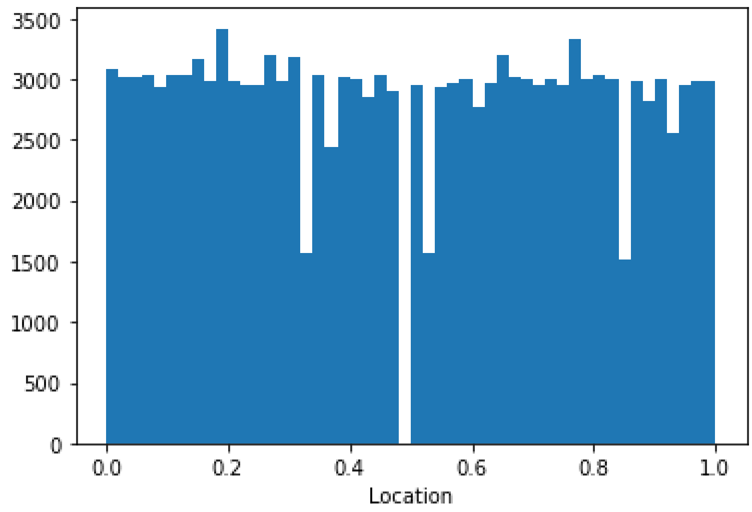
[11]:
```
import datetime as dt

data['Date'] = pd.to_datetime(data['Date'])
data['Date'] = data['Date'].map(dt.datetime.toordinal)
```

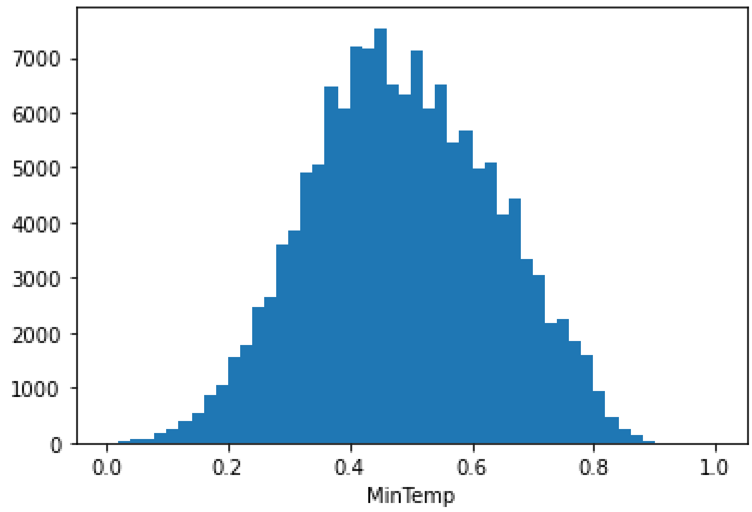
2.3. Масштабирование данных

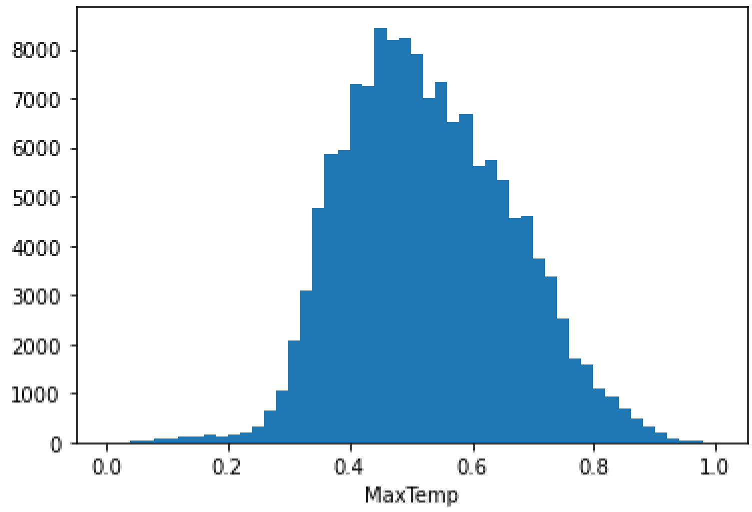
[14]:
```
from sklearn.preprocessing import MinMaxScaler

min_max_scaler = MinMaxScaler()
data[:] = min_max_scaler.fit_transform(data)
```

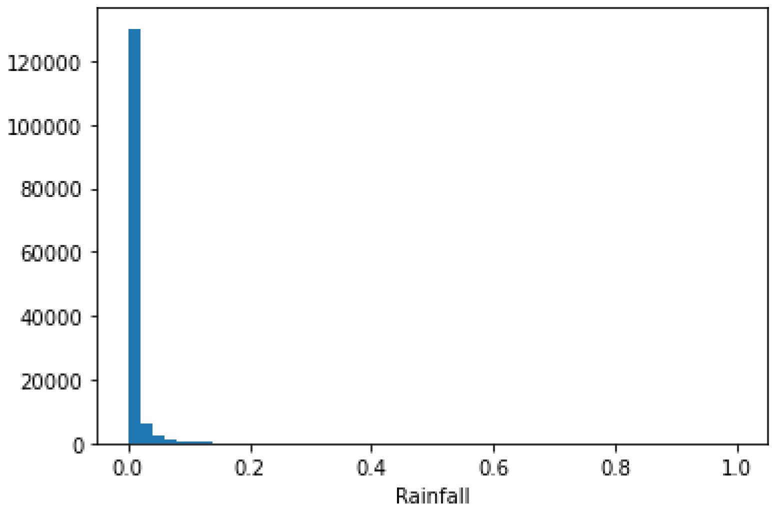
[190]:
```
for col in data.columns:
    plt.hist(data[col], 50)
    plt.xlabel(col)
    plt.show()
```

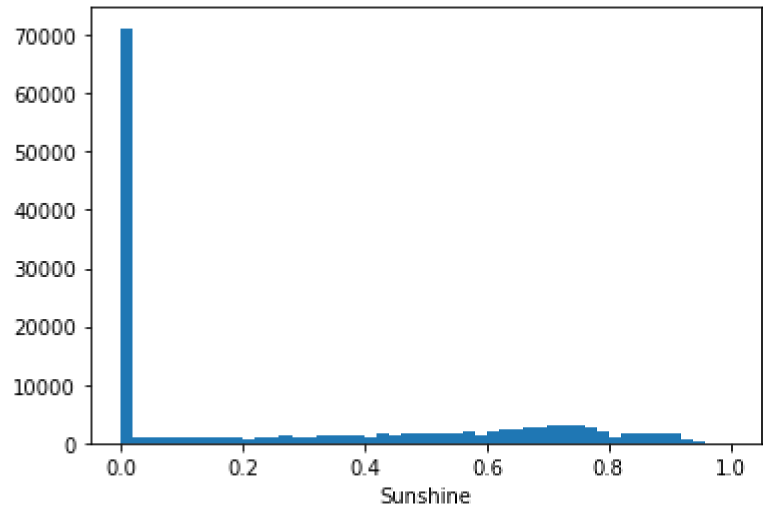


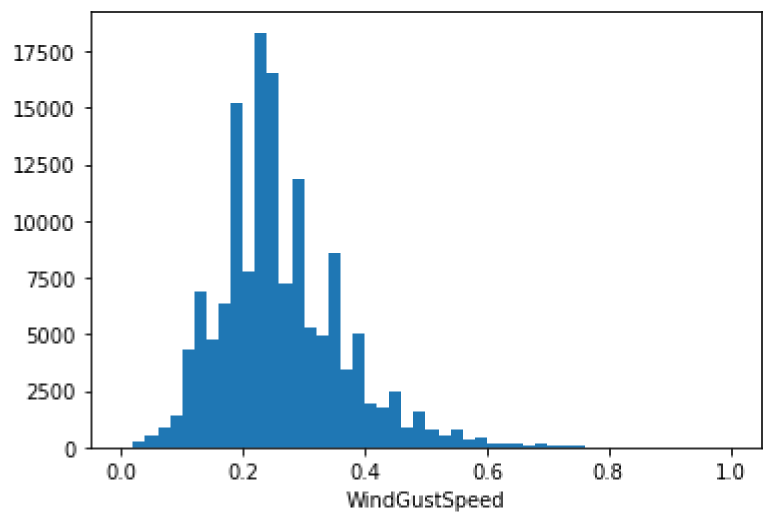

3500
3000
2500
2000
1500
1000
500
0
0.0
0.2
0.4
0.6
0.8
1.0
Location

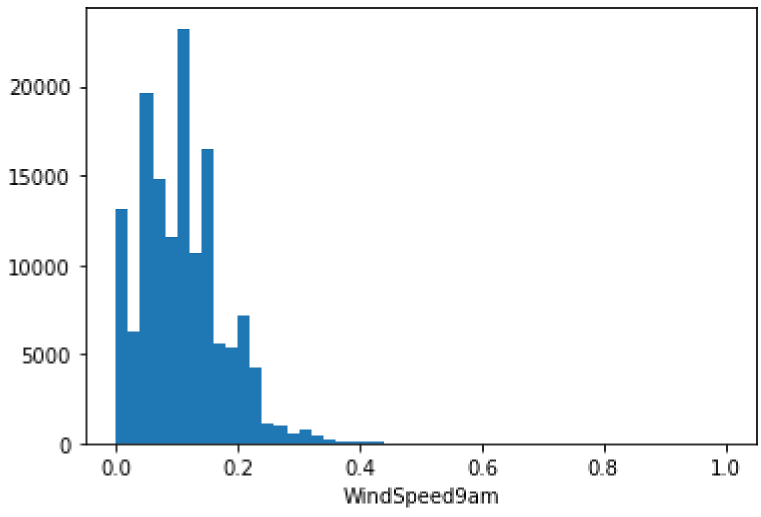

7000
6000
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0
MinTemp

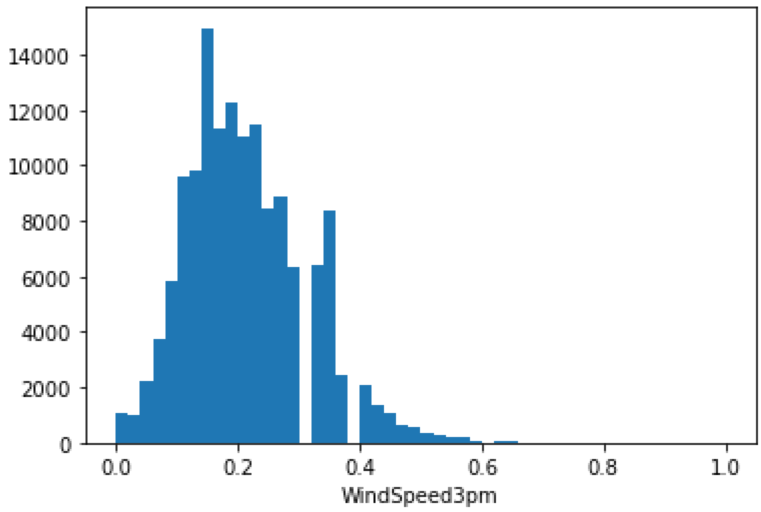

8000
7000
6000
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0
MaxTemp

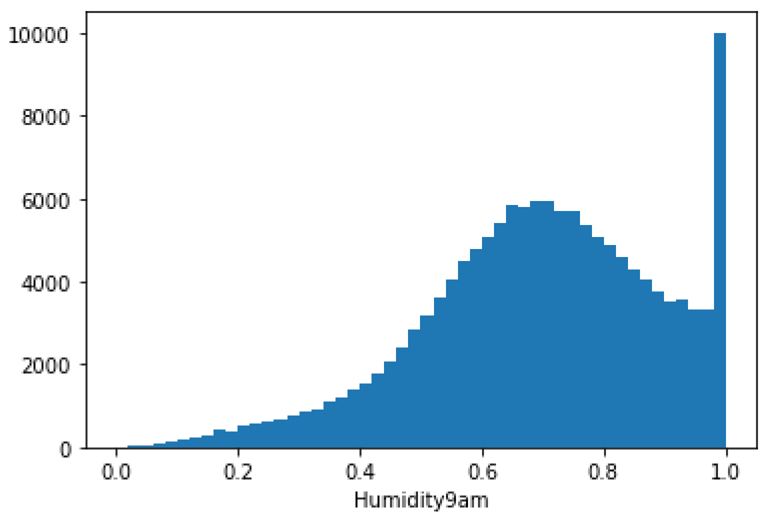

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
Rainfall

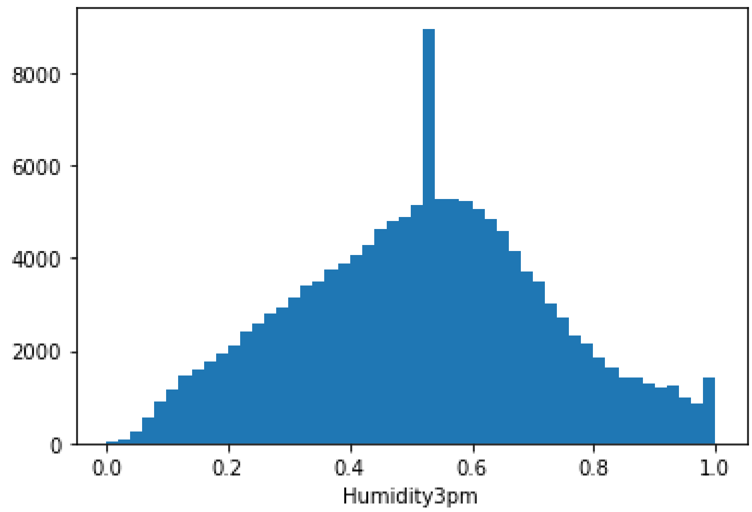

70000
60000
50000
40000
30000
20000
10000
0
0.0
0.2
0.4
0.6
0.8
1.0
Sunshine

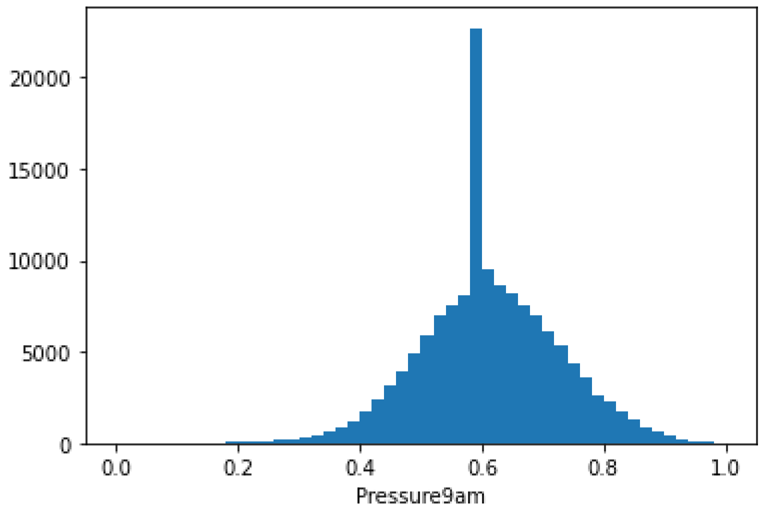

17500
15000
12500
10000
7500
5000
2500
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustSpeed

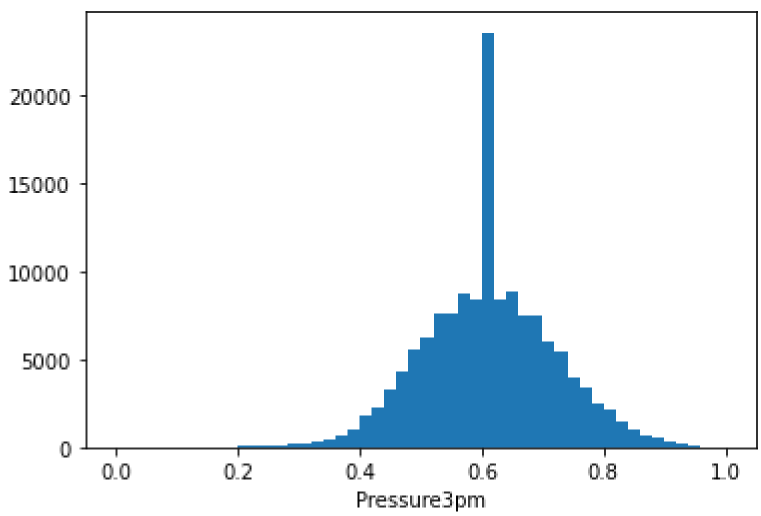

20000
15000
10000
5000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindSpeed9am

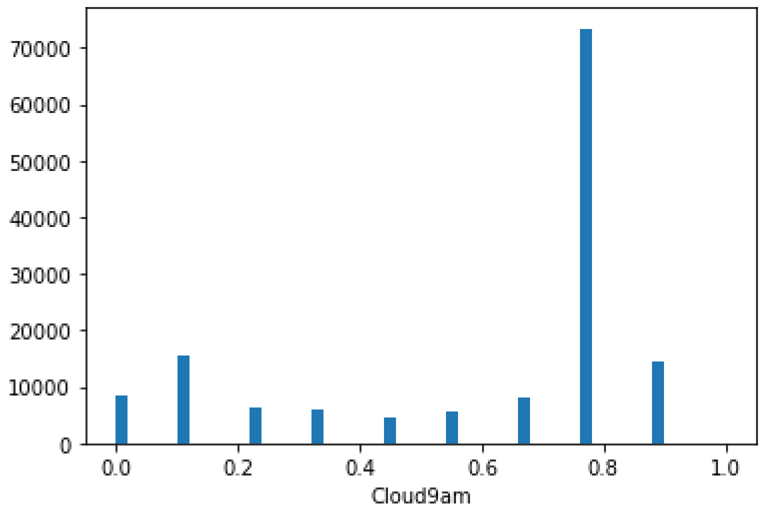

14000
12000
10000
8000
6000
4000
2000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindSpeed3pm

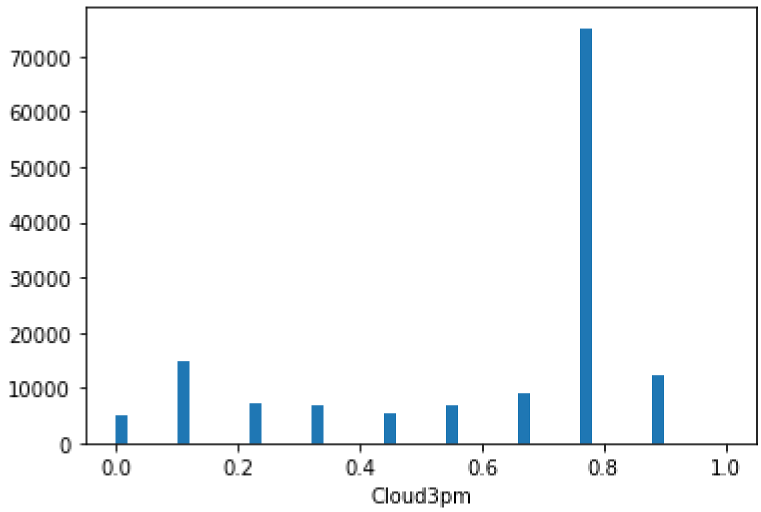

10000
8000
6000
4000
2000
0
0.0
0.2
0.4
0.6
0.8
1.0
Humidity9am

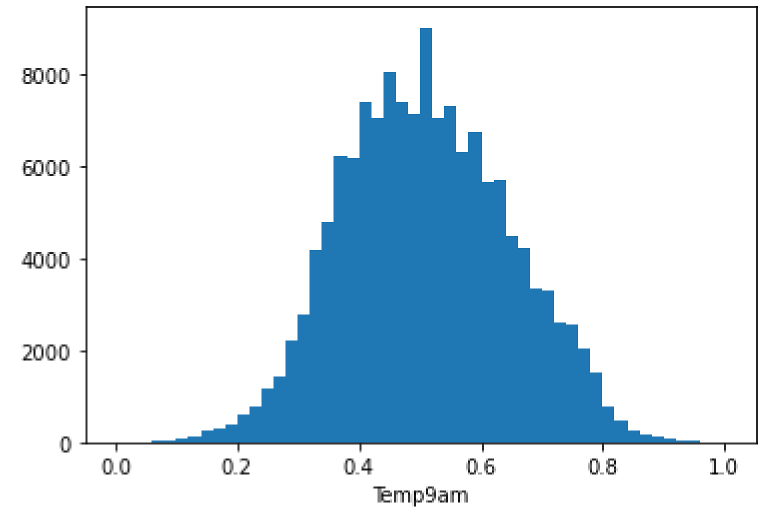

8000
6000
4000
2000
0
0.0
0.2
0.4
0.6
0.8
1.0
Humidity3pm

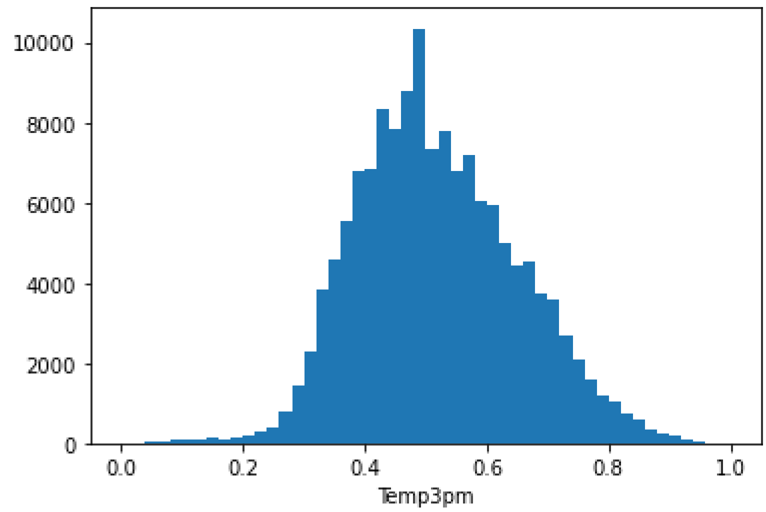

20000
15000
10000
5000
0
0.0
0.2
0.4
0.6
0.8
1.0
Pressure9am


20000
15000
10000
5000
0
0.0
0.2
0.4
0.6
0.8
1.0
Pressure3pm

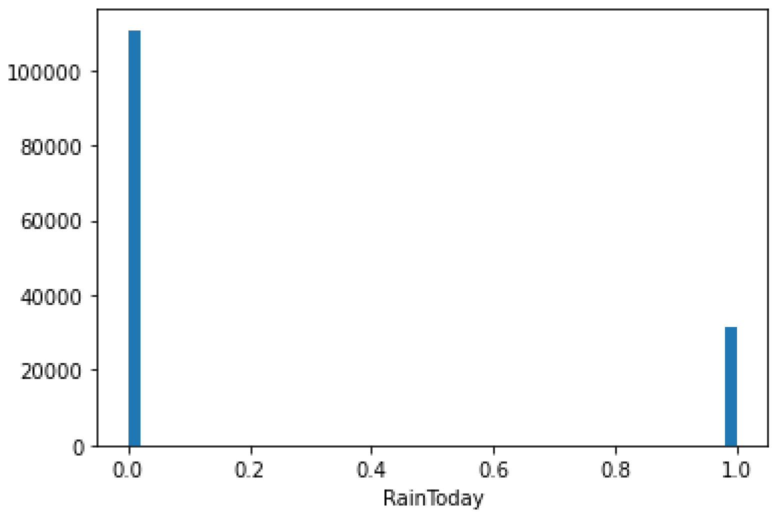

70000
60000
50000
40000
30000
20000
10000
0
0.0
0.2
0.4
0.6
0.8
1.0
Cloud9am


70000
60000
50000
40000
30000
20000
10000
0
0.0
0.2
0.4
0.6
0.8
1.0
Cloud3pm

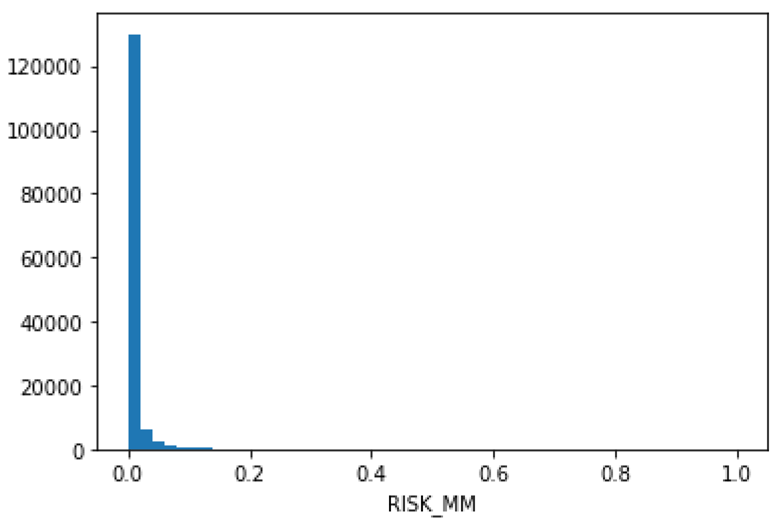

8000
6000
4000
2000
0
0.0
0.2
0.4
0.6
0.8
1.0
Temp9am



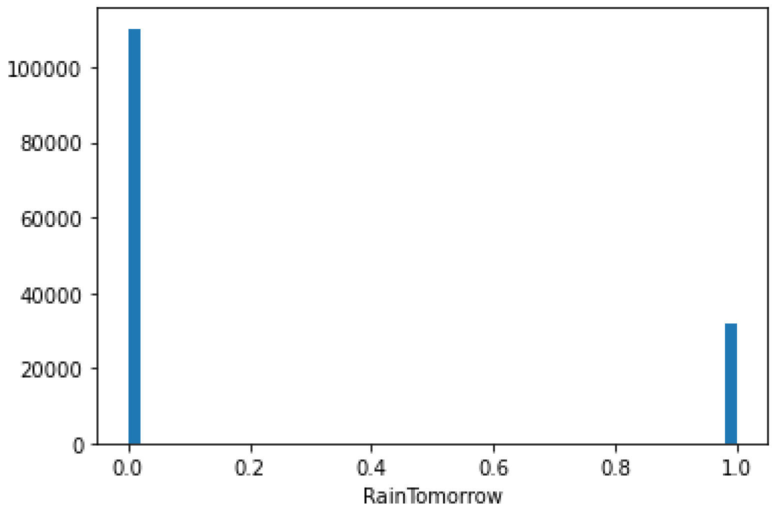
10000
8000
6000
4000
2000
0
0.0
0.2
0.4
0.6
0.8
1.0
Temp3pm

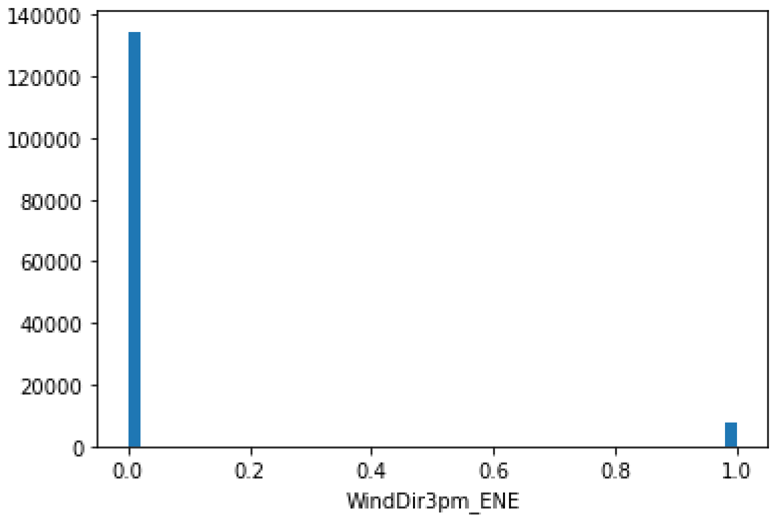


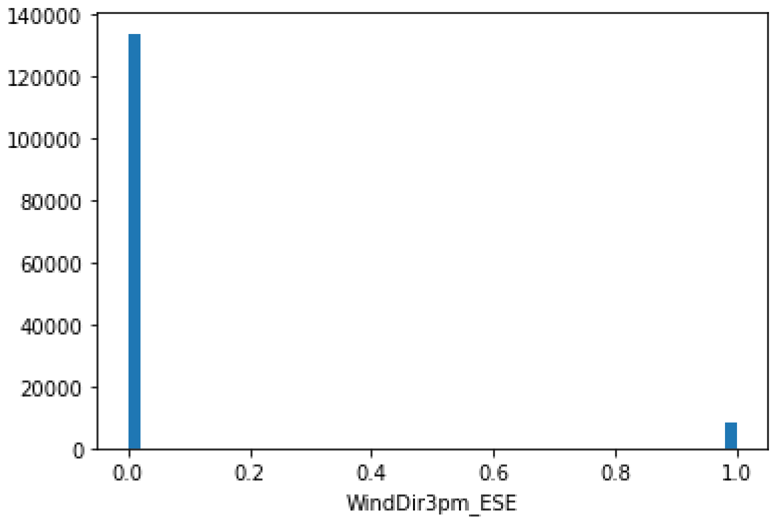
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
RainToday

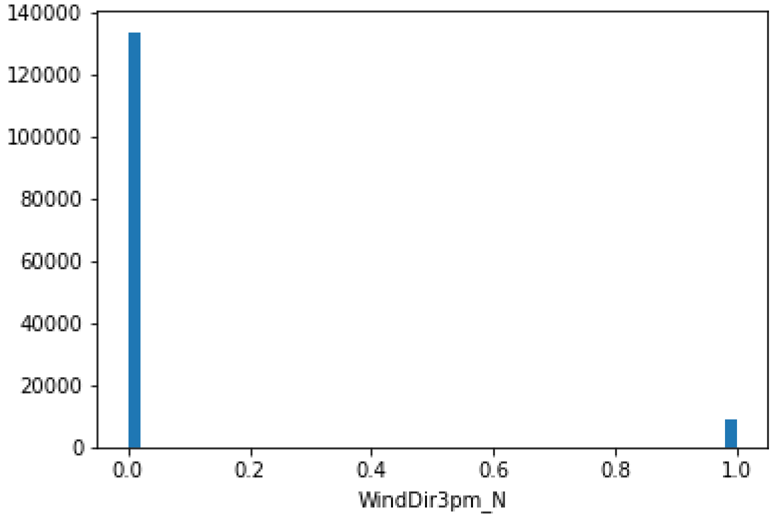


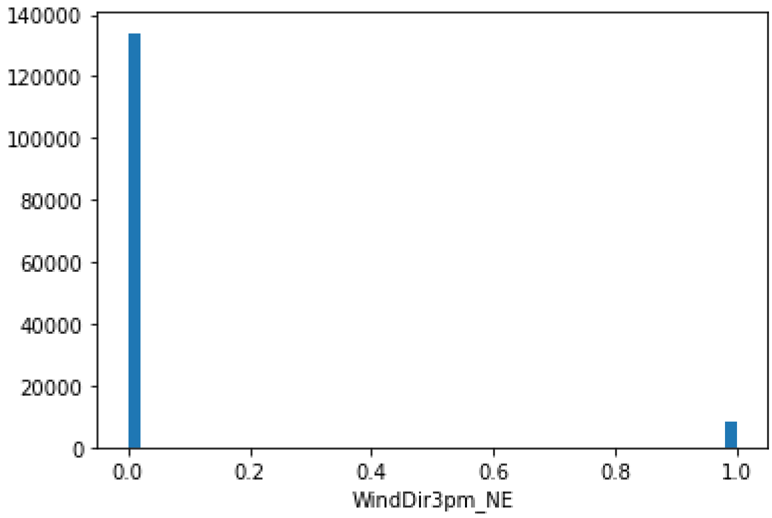
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
RISK_MM

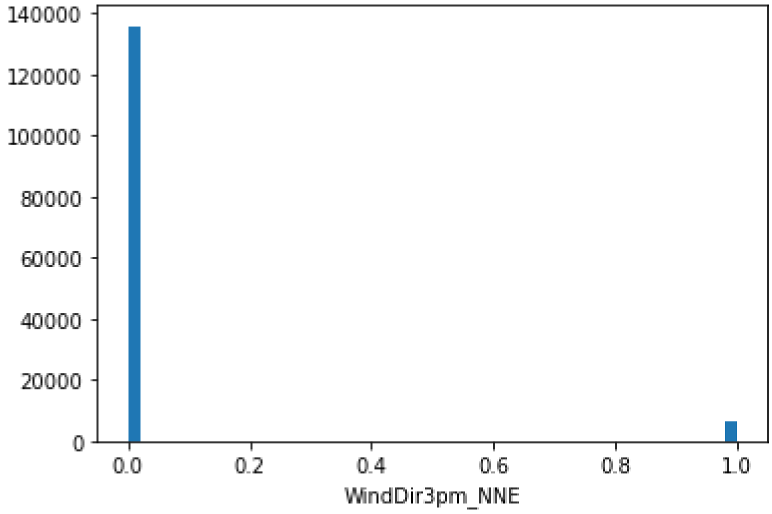

100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
RainTomorrow

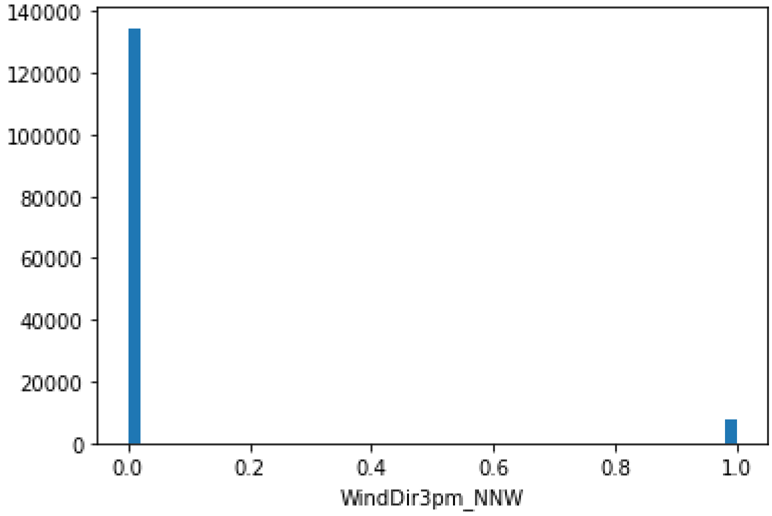

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_ENE

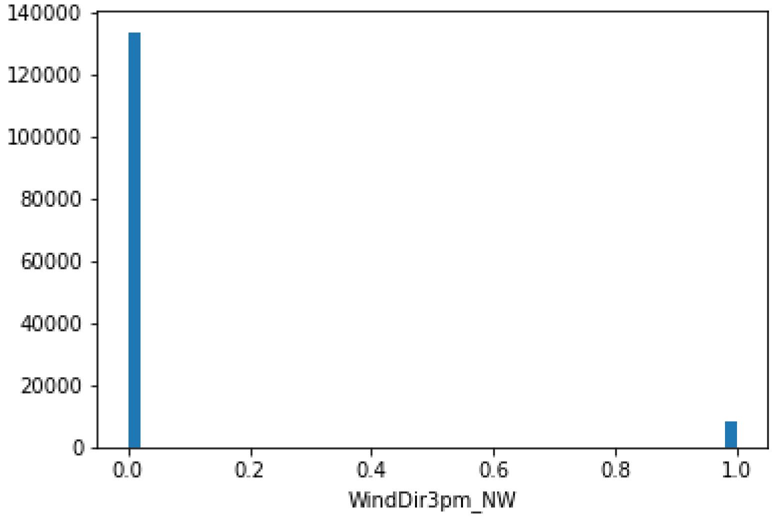

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_ESE

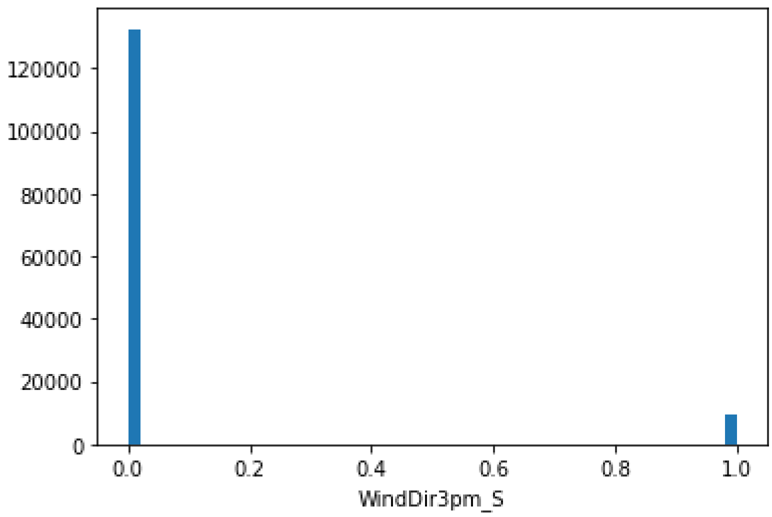

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_N

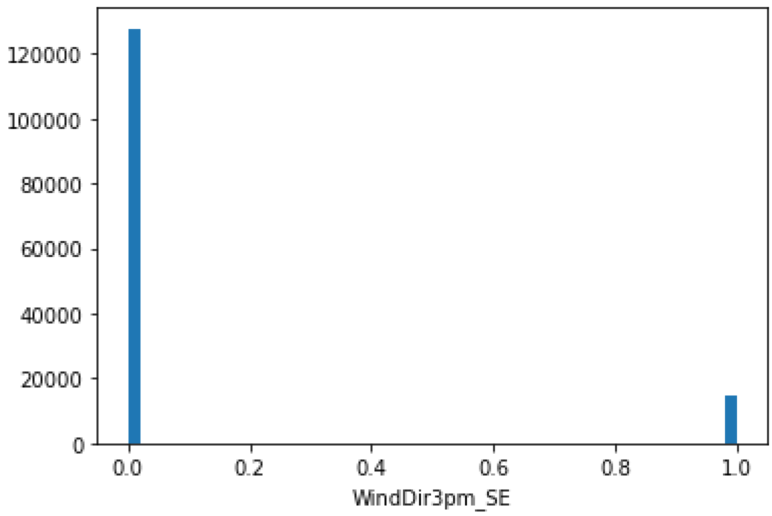

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_NE

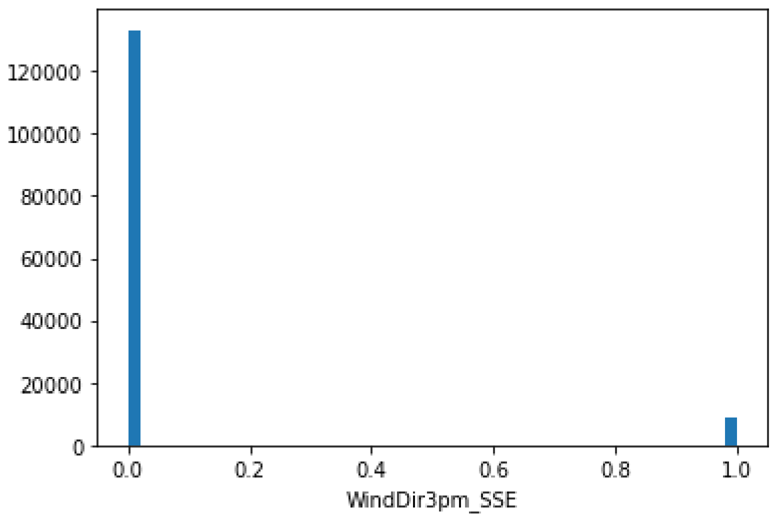

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_NNE

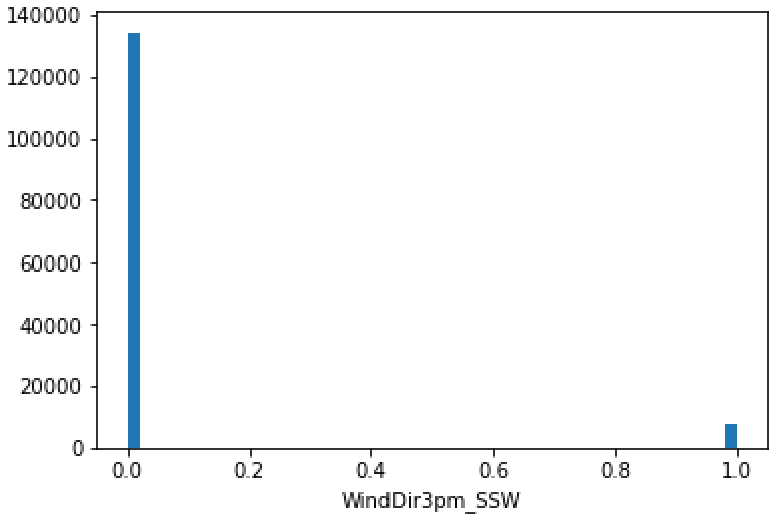

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_NNW

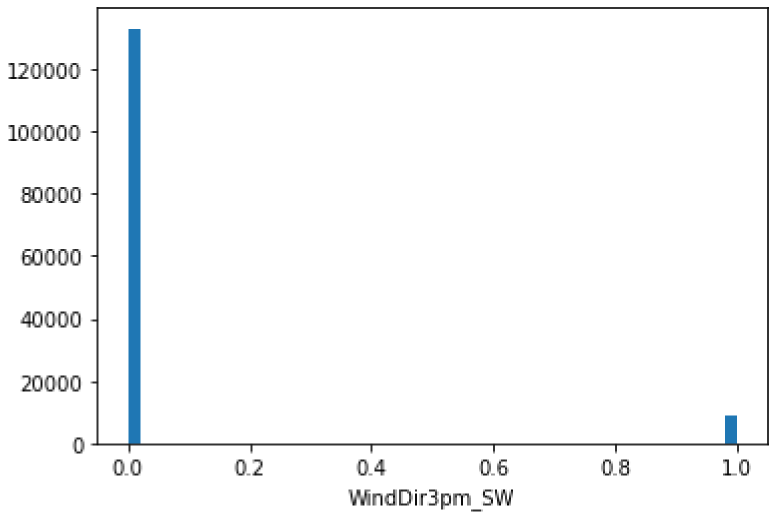

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_NW

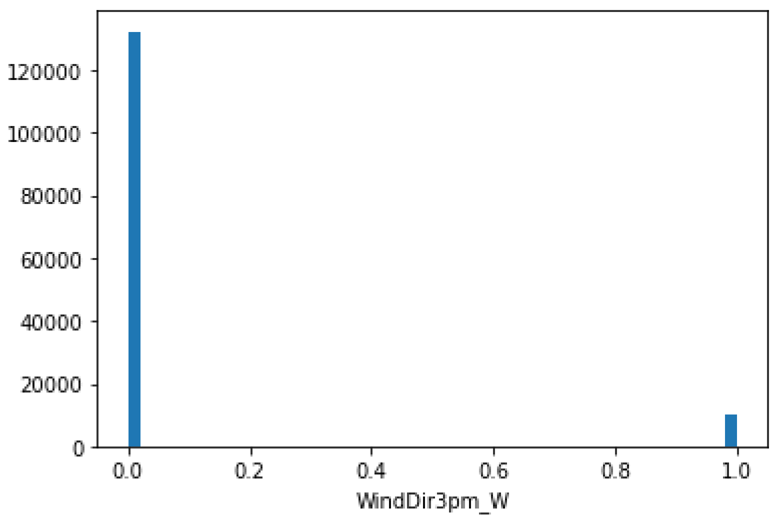

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_S

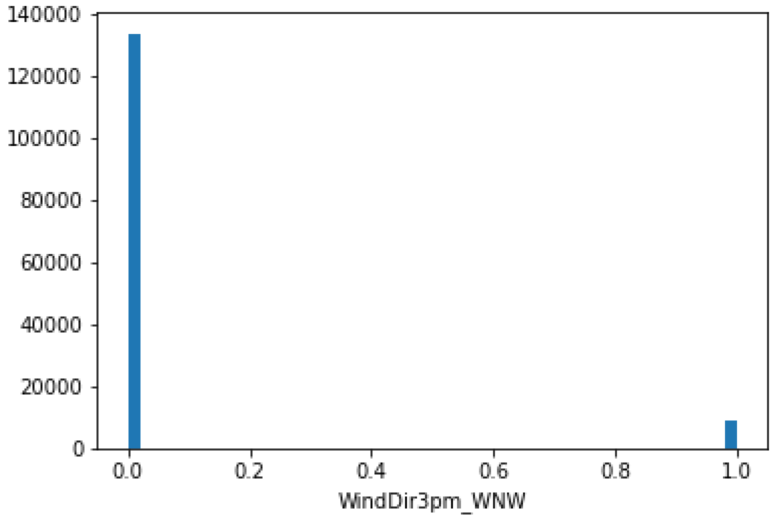

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_SE

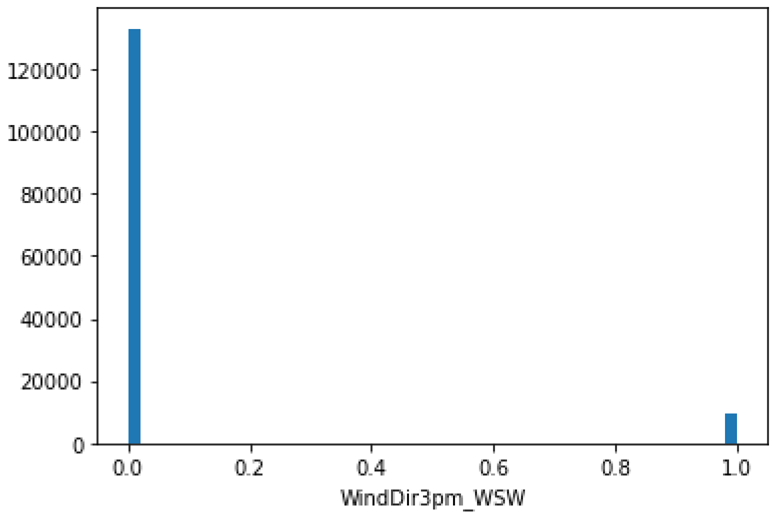

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_SSE

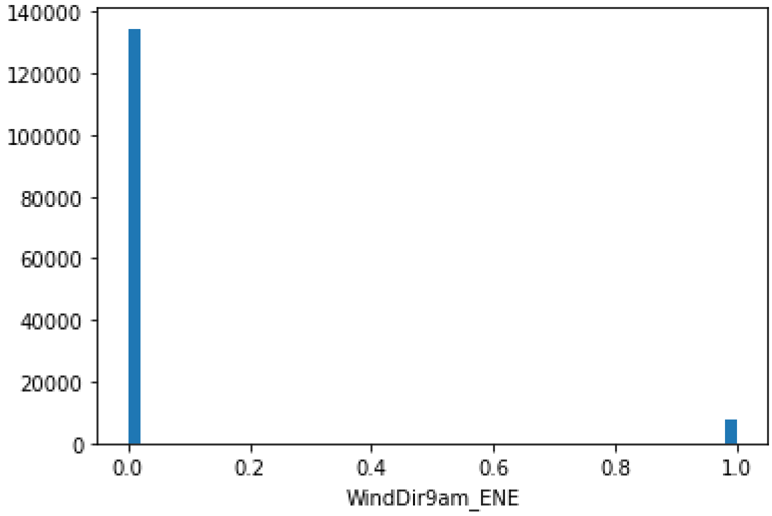

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_SSW

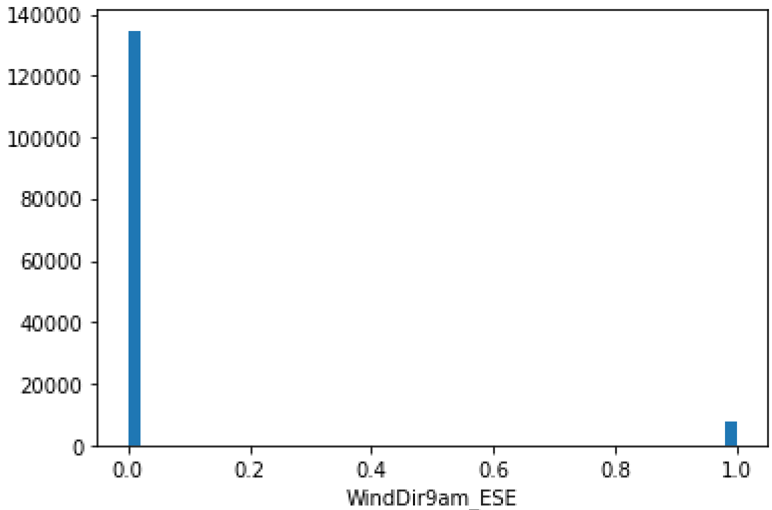

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_SW

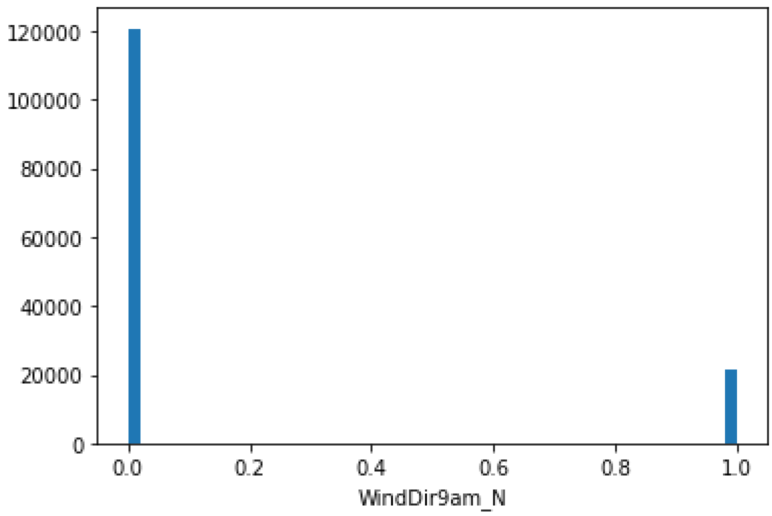

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_W

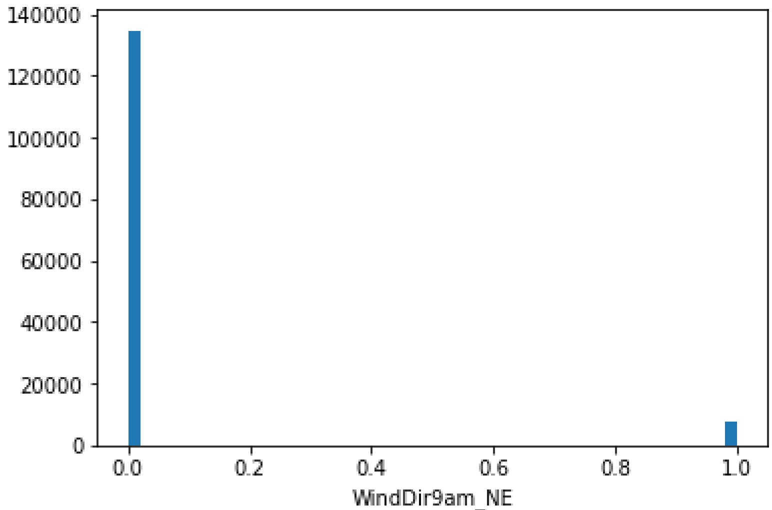

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_WNW

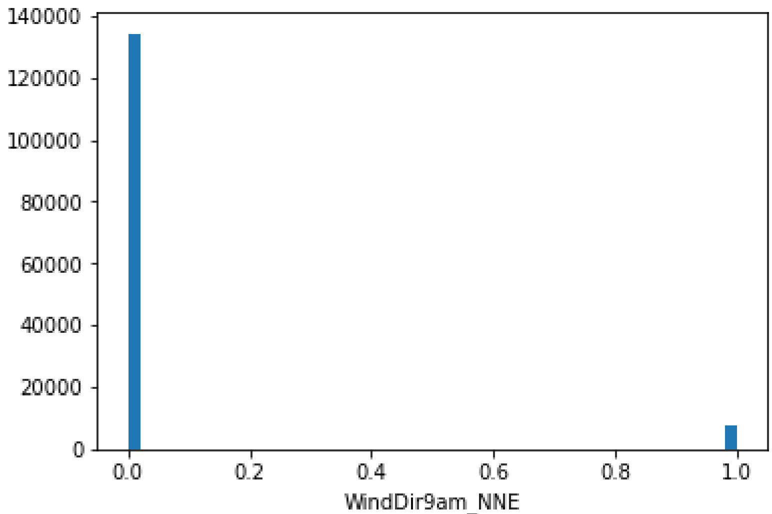

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir3pm_WSW

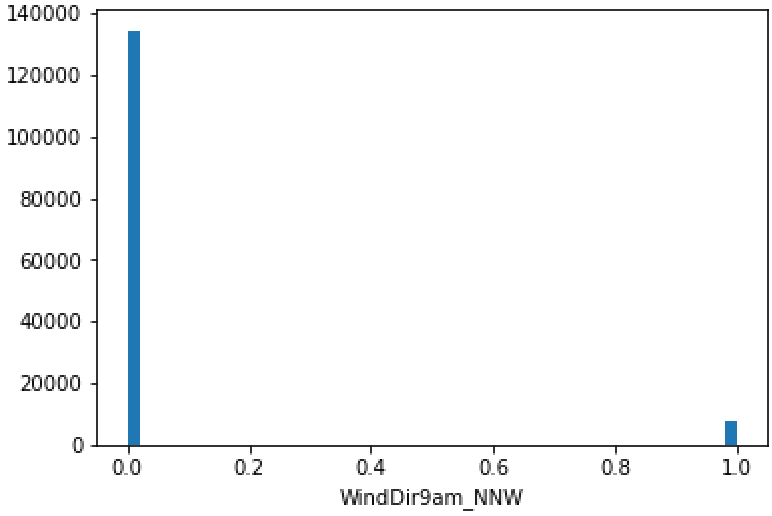

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_ENE

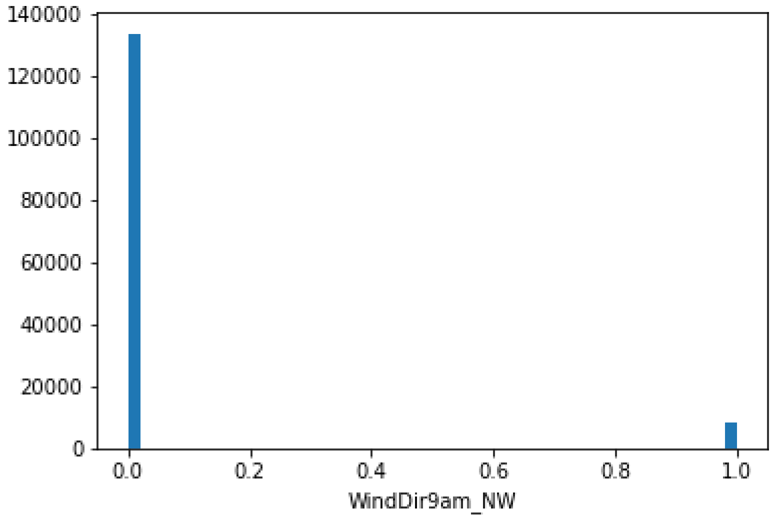

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_ESE

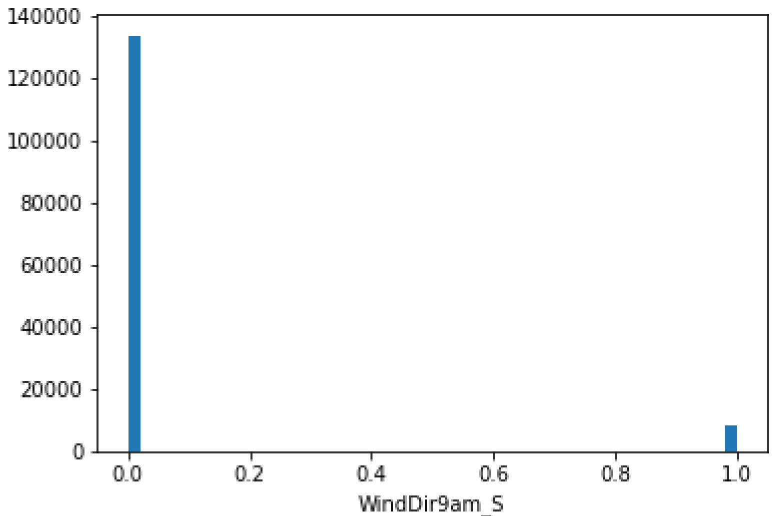

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_N

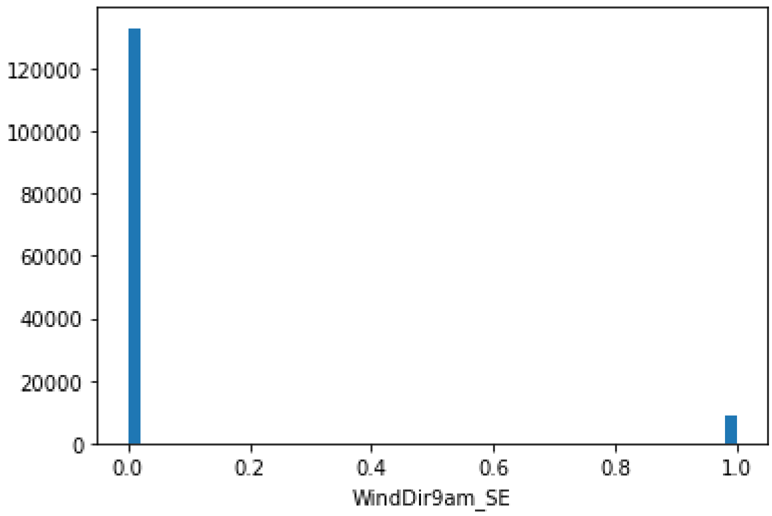

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_NE

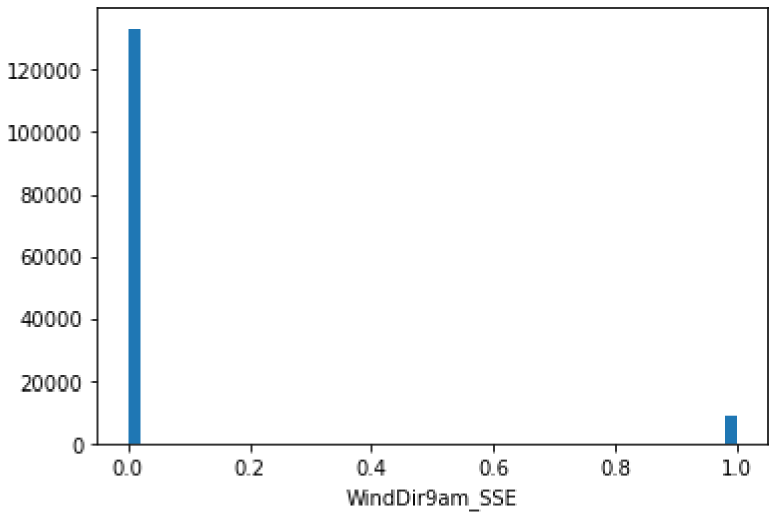

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_NNE

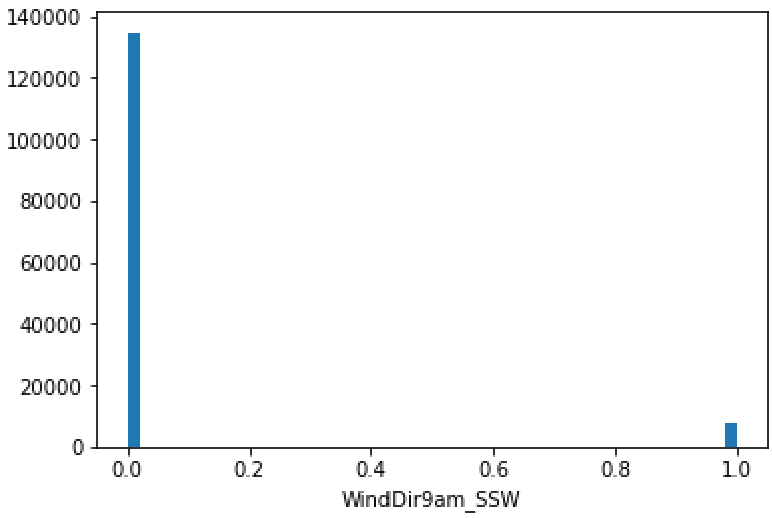

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_NNW

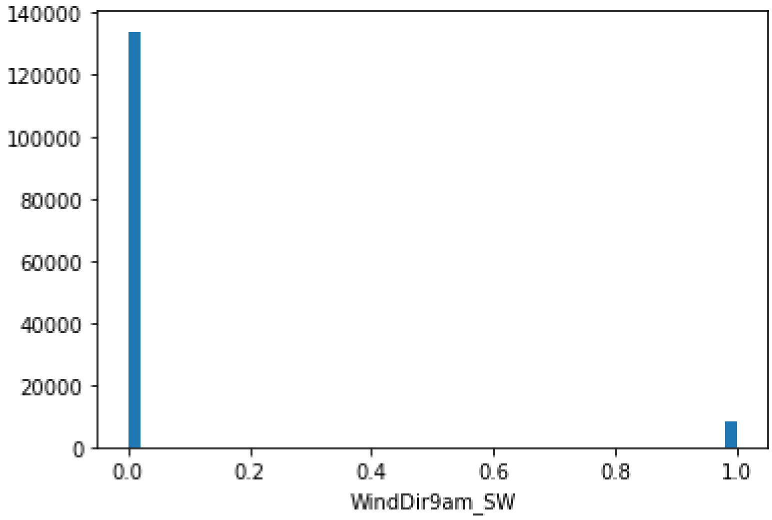

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_NW

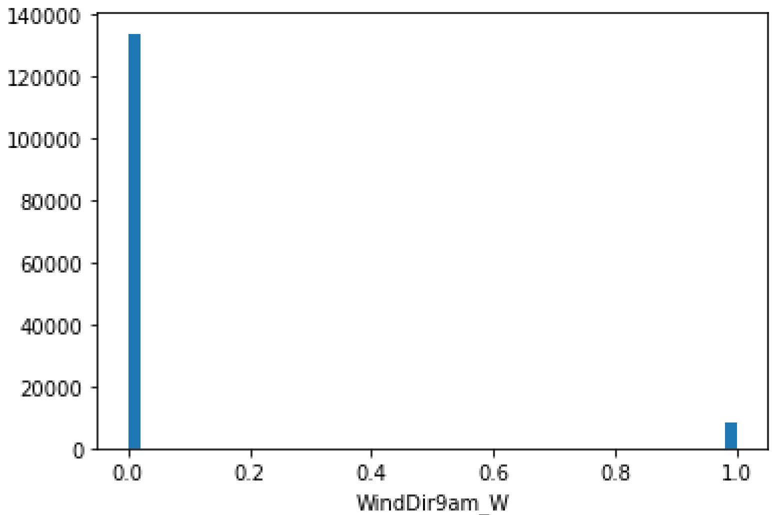

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_S

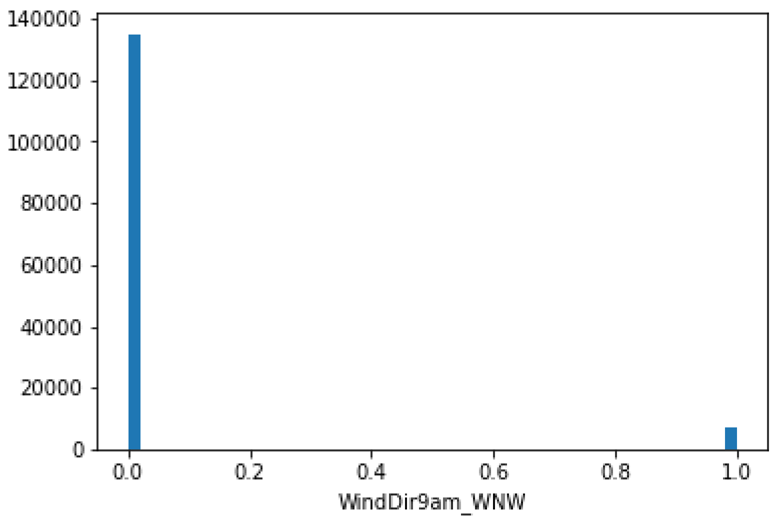

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_SE

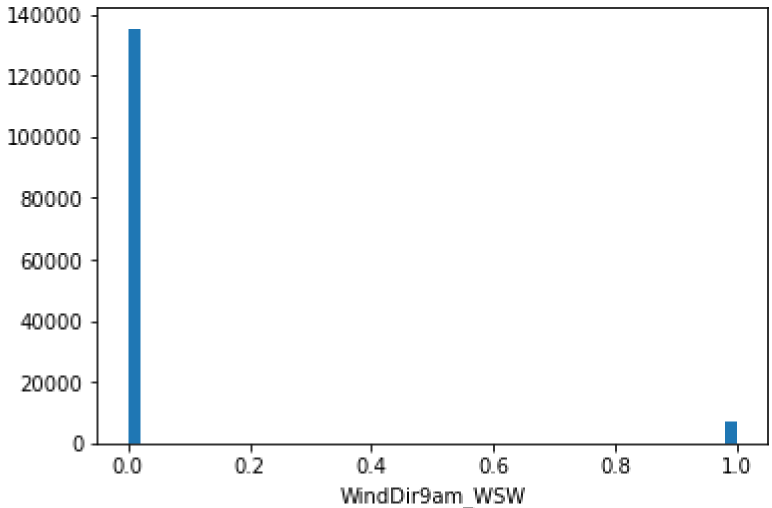

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_SSE

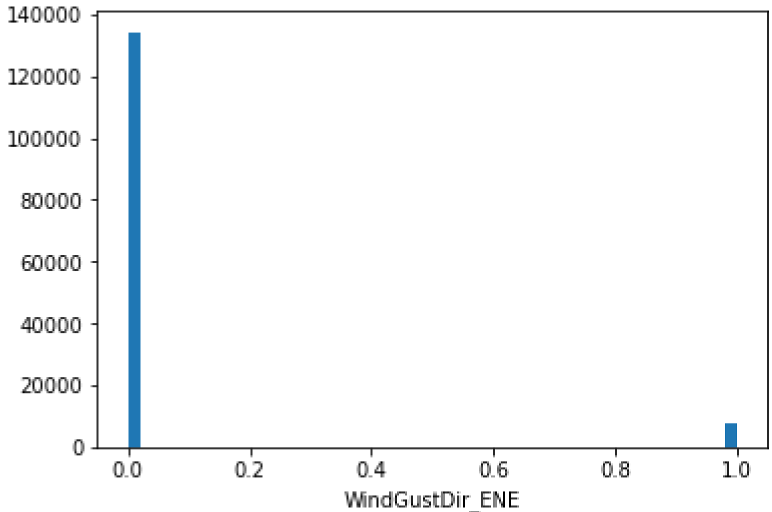

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_SSW

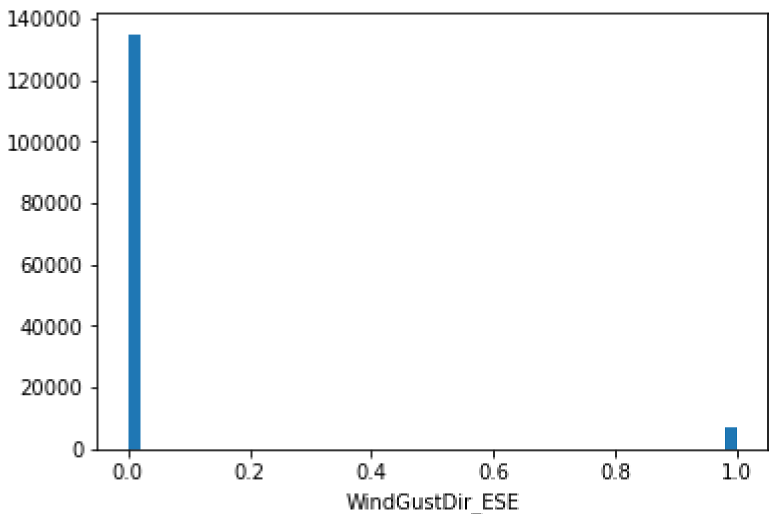

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_SW

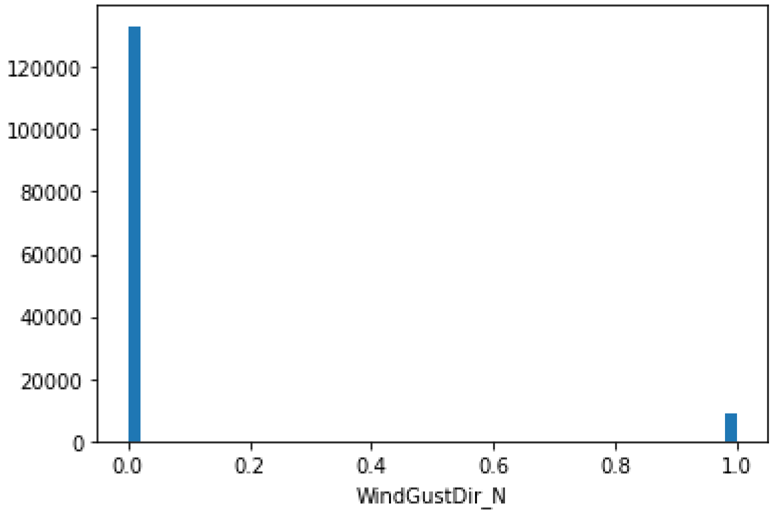

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_W

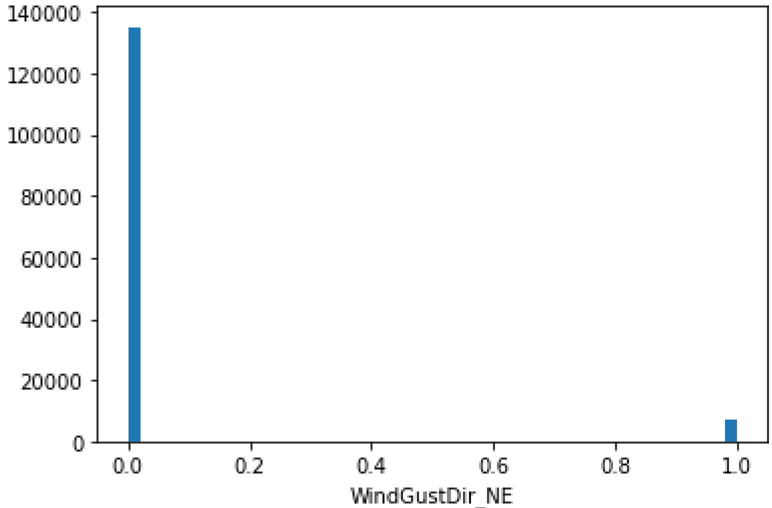

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_WNW

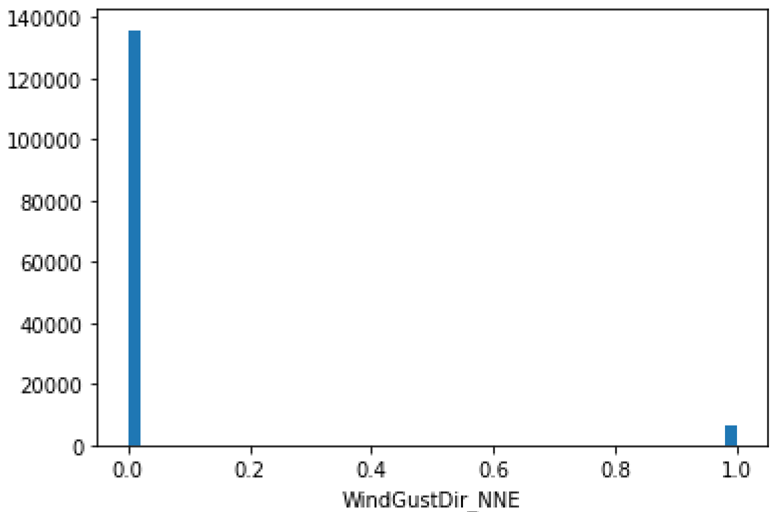

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindDir9am_WSW

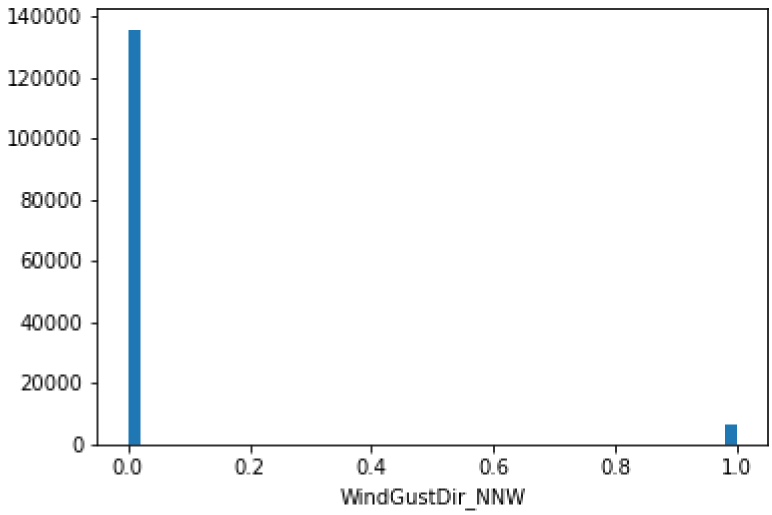

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_ENE

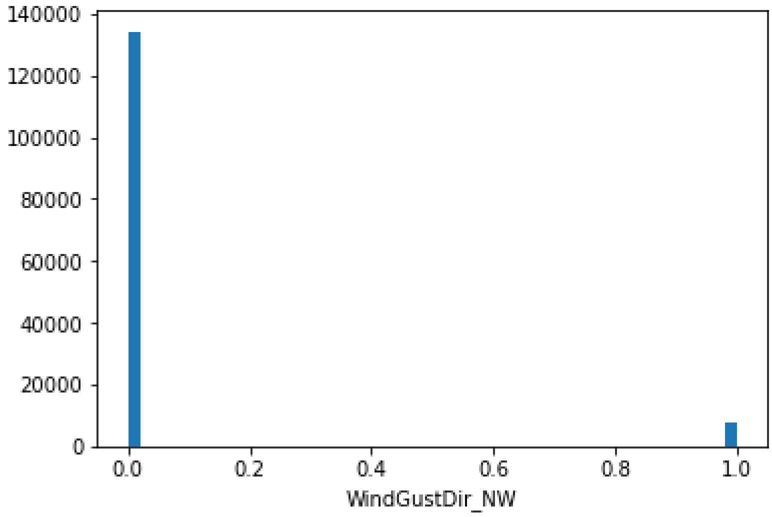

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_ESE

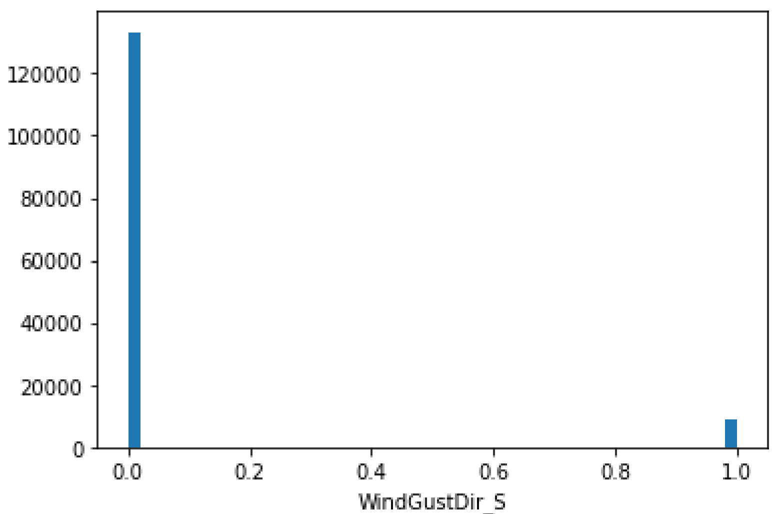

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_N

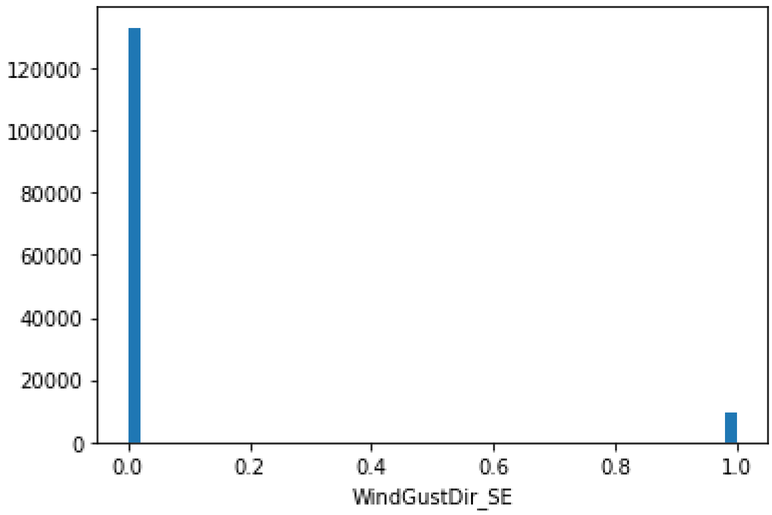

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_NE

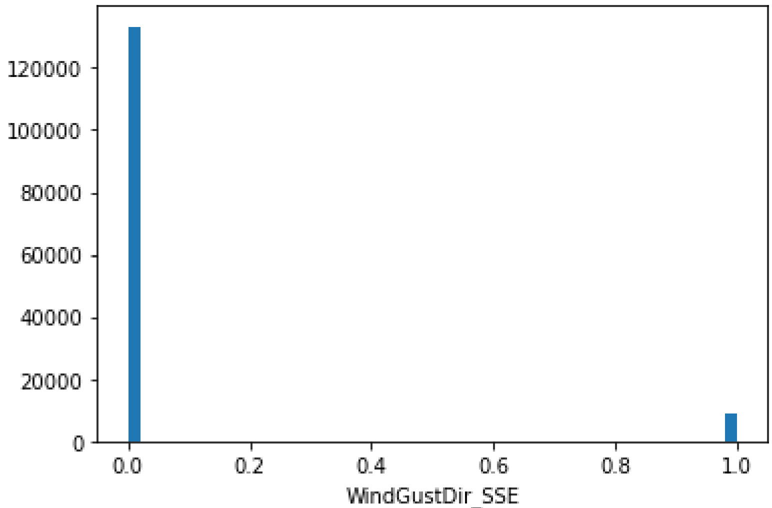

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_NNE

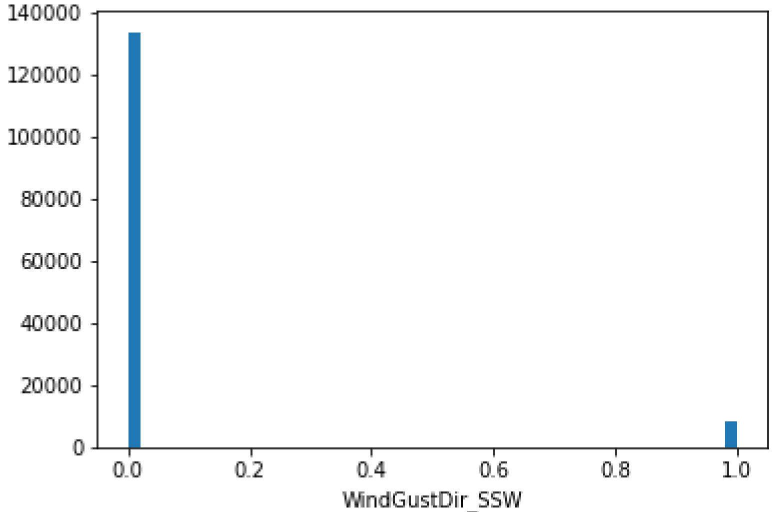

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_NNW

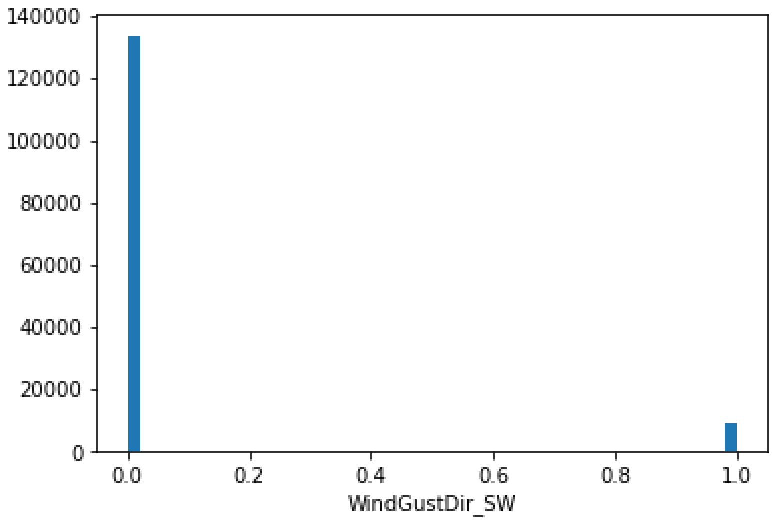

140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_NW

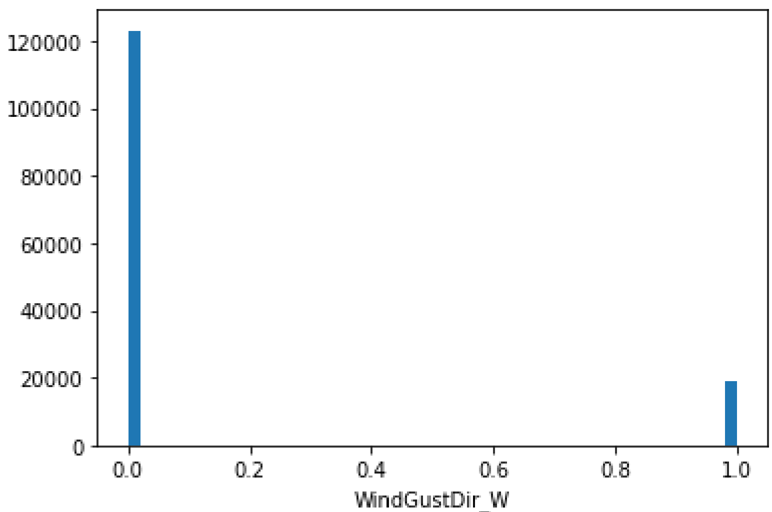

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_S

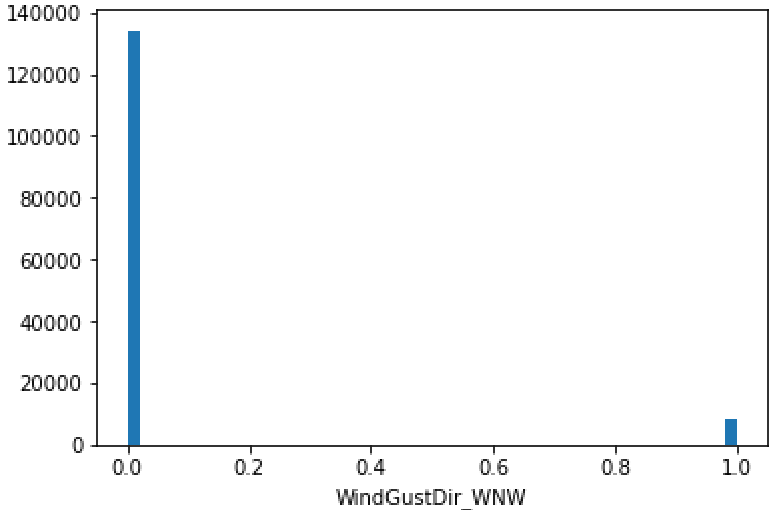

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_SE

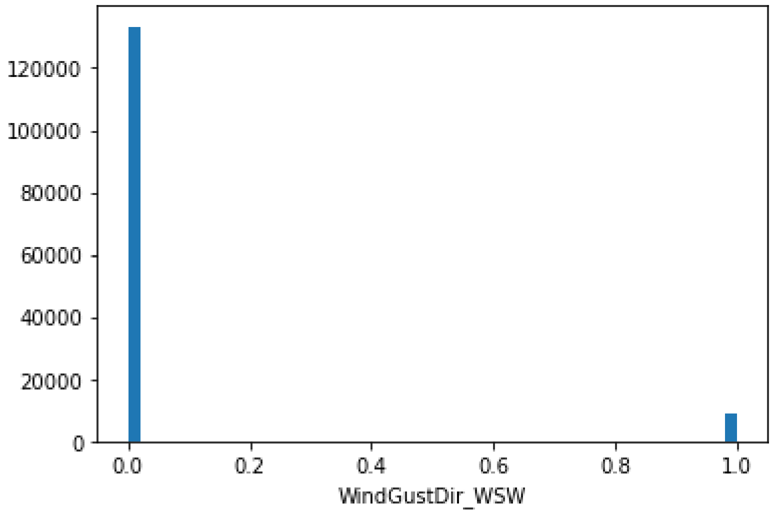

120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_SSE


140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_SSW


140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_SW


120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_W


140000
120000
100000
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
WindGustDir_WNW



3. Лабораторная №3

Подготовка обучающей и тестовой выборки, кросс-валидация и подбор гиперпараметров на примере метода ближайших соседей.

Задание:

1. Выберите набор данных (датасет) для решения задачи классификации или регрессии.
2. С использованием метода train_test_split разделите выборку на обучающую и тестовую.
3. Обучите модель ближайших соседей для произвольно заданного гиперпараметра K. Оцените качество модели с помощью подходящих для задачи метрик.
4. Произведите подбор гиперпараметра K с использованием GridSearchCV и/или RandomizedSearchCV и кросс-валидации, оцените качество оптимальной модели. Желательно использование нескольких стратегий кросс-валидации.
5. Сравните метрики качества исходной и оптимальной моделей.

3.1. Разделение выборки на обучающую и тестовую

[12]:
```
data = data.drop(['RISK_MM'], axis = 1)
```

[13]:
```
y = data['RainTomorrow']
data.drop(['RainTomorrow'], axis = 1)
```

[13]:
```
          Date  Location  MinTemp  MaxTemp  Rainfall  Sunshine ␣
↪WindGustSpeed  \
0       733377         2     13.4     22.9       0.6       0.0            44.0
1       733378         2      7.4     25.1       0.0       0.0            44.0
2       733379         2     12.9     25.7       0.0       0.0            46.0
3       733380         2      9.2     28.0       0.0       0.0            24.0
```

```
4       733381        2     17.5     32.3      1.0      0.0            41.0
…        …      …     …      …       …      …          …
142188  736500       41      3.5     21.8      0.0      0.0            31.0
142189  736501       41      2.8     23.4      0.0      0.0            31.0
142190  736502       41      3.6     25.3      0.0      0.0            22.0
142191  736503       41      5.4     26.9      0.0      0.0            37.0
142192  736504       41      7.8     27.0      0.0      0.0            28.0

        WindSpeed9am  WindSpeed3pm  Humidity9am  …  WindGustDir_NNW  \
0               20.0          24.0         71.0  …                0
1                4.0          22.0         44.0  …                0
2               19.0          26.0         38.0  …                0
3               11.0           9.0         45.0  …                0
4                7.0          20.0         82.0  …                0
…                …            …           …   …              …
142188          15.0          13.0         59.0  …                0
142189          13.0          11.0         51.0  …                0
142190          13.0           9.0         56.0  …                1
142191           9.0           9.0         53.0  …                0
142192          13.0           7.0         51.0  …                0

        WindGustDir_NW  WindGustDir_S  WindGustDir_SE  WindGustDir_SSE  \
0                    0              0               0                0
1                    0              0               0                0
2                    0              0               0                0
3                    0              0               0                0
4                    0              0               0                0
…                  …            …              …               …
142188               0              0               0                0
142189               0              0               0                0
142190               0              0               0                0
142191               0              0               0                0
142192               0              0               1                0

        WindGustDir_SSW  WindGustDir_SW  WindGustDir_W  WindGustDir_WNW  \
0                     0               0              1                0
1                     0               0              0                1
2                     0               0              0                0
3                     0               0              0                0
4                     0               0              1                0
…                   …             …            …              …
142188                0               0              0                0
142189                0               0              0                0
142190                0               0              0                0
142191                0               0              0                0
142192                0               0              0                0

        WindGustDir_WSW
0                     0
1                     0
```

```
2                      1
3                      0
4                      0
...                  ...
142188                 0
142189                 0
142190                 0
142191                 0
142192                 0

[142193 rows x 63 columns]
```

[14]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(data, y, test_size=0.
 ↪25, random_state=23)
```

3.2. Модель ближайших соседей для произвольно заданного гиперпараметра K

Выберем K=5

[15]:
```
from sklearn.neighbors import KNeighborsClassifier
```

[32]:
```
%%time
neigh = KNeighborsClassifier(n_neighbors=5)
neigh.fit(X_train, y_train)

prediction = neigh.predict(X_test)
```

```
CPU times: user 11.6 s, sys: 55 ms, total: 11.7 s
Wall time: 11.7 s
```

[16]:
```
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import precision_score, recall_score, f1_score,␣
 ↪classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_curve, roc_auc_score
```

[32]:
```
print ("balanced_accuracy_score = {}".
 ↪format(balanced_accuracy_score(y_test, prediction)))  #
print ("precision_score = {}".format(precision_score(y_test, prediction)))
print ("recall_score = {}".format(recall_score(y_test, prediction)))
print ("f1_score = {}".format(f1_score(y_test, prediction)))
```

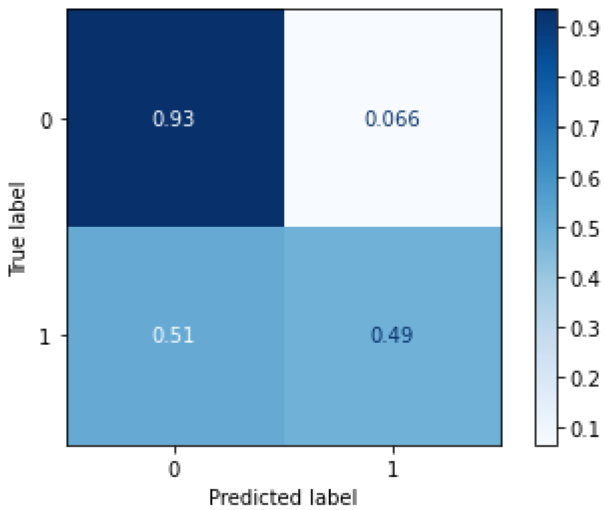
```
0.8261273172241131
balanced_accuracy_score = 0.6546730748378109
precision_score = 0.7208135235076598
recall_score = 0.34751050553928436
f1_score = 0.46894063063837105
```

[36]:
```
classification_report(y_test, prediction, target_names=['rain','no rain'], output_dict=True)
```

[36]:
```
{'rain': {'precision': 0.8386802254195133,
  'recall': 0.9618356441363374,
  'f1-score': 0.8960460149010242,
  'support': 27696},
 'no rain': {'precision': 0.7208135235076598,
  'recall': 0.34751050553928436,
  'f1-score': 0.46894063063837105,
  'support': 7853},
 'accuracy': 0.8261273172241131,
 'macro avg': {'precision': 0.7797468744635865,
  'recall': 0.6546730748378109,
  'f1-score': 0.6824933227696977,
  'support': 35549},
 'weighted avg': {'precision': 0.8126427219703647,
  'recall': 0.8261273172241131,
  'f1-score': 0.8016957214296294,
  'support': 35549}}
```

[40]:
```
plot_confusion_matrix(neigh, X_test, y_test, cmap=plt.cm.Blues, normalize='true')
```

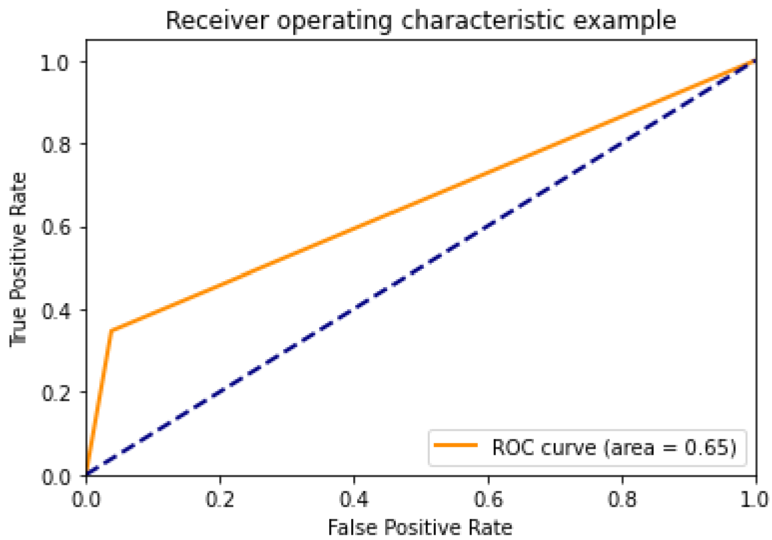
[40]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x7fad98c6eb20>
```



[38]:
```
#       ROC-
def draw_roc_curve(y_true, y_score, pos_label, average):
    fpr, tpr, thresholds = roc_curve(y_true, y_score,
                                     pos_label=pos_label)
    roc_auc_value = roc_auc_score(y_true, y_score, average=average)
    plt.figure()
```

```
    lw = 2
    plt.plot(fpr, tpr, color='darkorange',
             lw=lw, label='ROC curve (area = %0.2f)' % roc_auc_value)
    plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic')
    plt.legend(loc="lower right")
    plt.show()
```

[40]:
```
draw_roc_curve(y_test, prediction, pos_label=1, average='micro')
```



3.3. Подбор гиперпараметра K

[17]:
```
from sklearn.model_selection import cross_val_score, cross_validate
from sklearn.model_selection import KFold, RepeatedKFold, LeaveOneOut,␣
↪LeavePOut, ShuffleSplit, StratifiedKFold
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.model_selection import learning_curve, validation_curve
```

Оценка качества модели с использованием кросс-валидации

[23]:
```
scores = cross_val_score(KNeighborsClassifier(n_neighbors=50),
                         data, y,
                         cv=5)
scores, np.mean(scores)
```

[23]:
```
(array([0.8317803 , 0.82295439, 0.83353845, 0.83300513, 0.82962937]),
 0.8301815298322947)
```

Подбор гиперпараметров на основе решетчатого поиска и кросс-валидации

[22]:
```
n_range = np.array(range(5,45,5))
tuned_parameters = [{'n_neighbors': n_range}]
tuned_parameters
```

[22]:
```
[{'n_neighbors': array([ 5, 10, 15, 20, 25, 30, 35, 40])}]
```

[23]:
```
%%time
clf_gs2 = GridSearchCV(KNeighborsClassifier(), tuned_parameters, cv=5)
clf_gs2.fit(data, y)
```

```
CPU times: user 5min 8s, sys: 1.18 s, total: 5min 9s
Wall time: 5min 9s
```

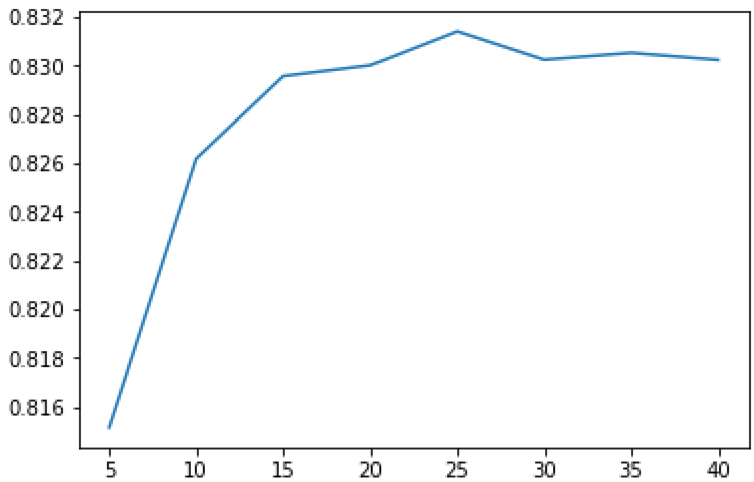
[23]:
```
GridSearchCV(cv=5, estimator=KNeighborsClassifier(),
             param_grid=[{'n_neighbors': array([ 5, 10, 15, 20, 25, 30, 35,
40])}])
```

[24]:
```
clf_gs2.best_params_
```

[24]:
```
{'n_neighbors': 25}
```

[25]:
```
plt.plot(n_range, clf_gs2.cv_results_['mean_test_score'])
```

[25]:
```
[<matplotlib.lines.Line2D at 0x7fad9a126f40>]
```



[26]:
```
clf_gs2.best_estimator_.fit(X_train, y_train)
best_prefiction1 = clf_gs2.best_estimator_.predict(X_train)
best_prefiction2 = clf_gs2.best_estimator_.predict(X_test)
```

3.4. Сравнение метрик качества исходной и оптимальной моделей

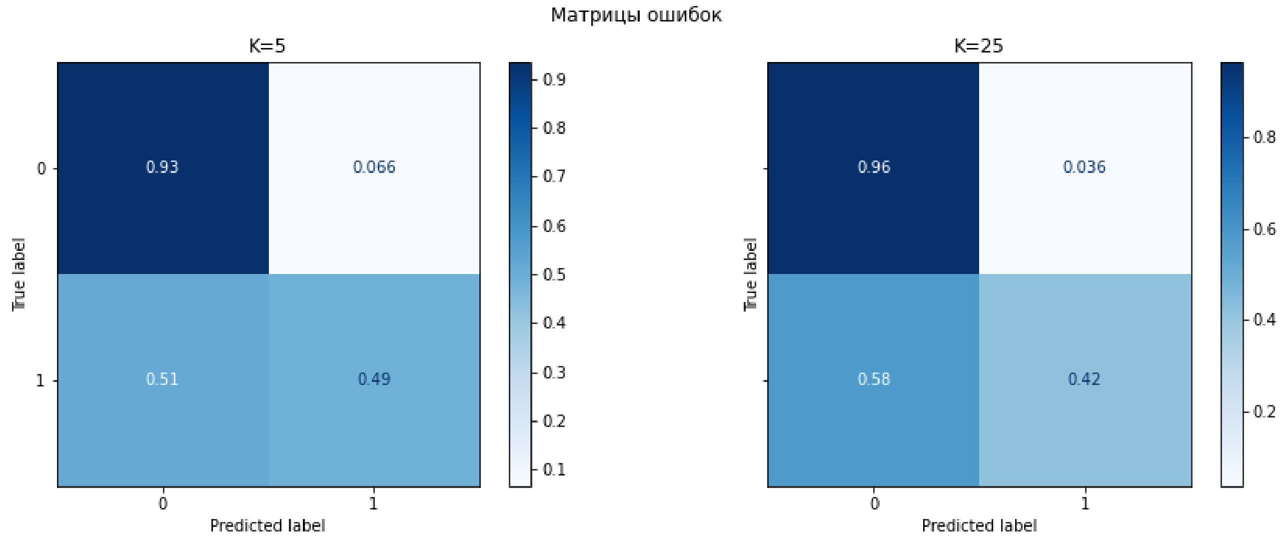
[50]:
```
print ("balanced_accuracy_score = {}".
 ↪format(balanced_accuracy_score(y_train, best_prefiction1)))  #
print ("precision_score = {}".format(precision_score(y_train,␣
 ↪best_prefiction1)))
print ("recall_score = {}".format(recall_score(y_train, best_prefiction1)))
print ("f1_score = {}".format(f1_score(y_train, best_prefiction1)))
```

```
balanced_accuracy_score = 0.7045669472447322
precision_score = 0.745130890052356
recall_score = 0.453075257863237
f1_score = 0.5635096610706366
```

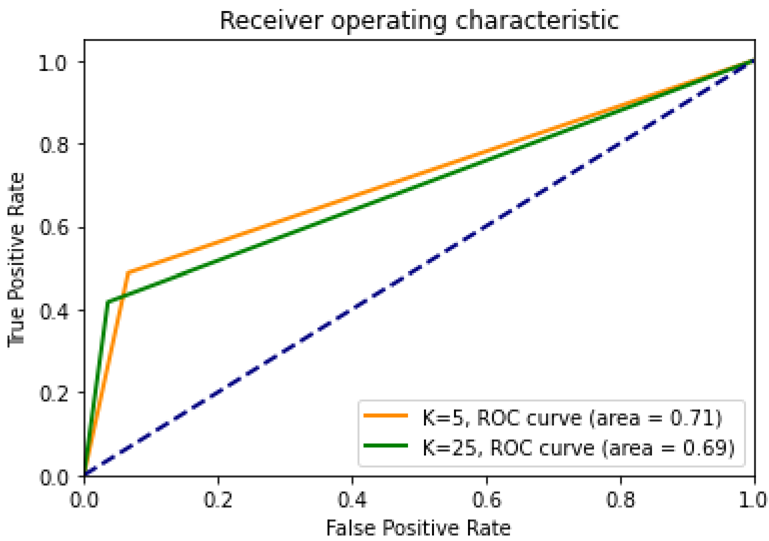
[39]:
```
print ("accuracy_score = {}".format(accuracy_score(y_test,␣
 ↪best_prefiction2)))
print ("balanced_accuracy_score = {}".
 ↪format(balanced_accuracy_score(y_test, best_prefiction2)))  #
print ("precision_score = {}".format(precision_score(y_test,␣
 ↪best_prefiction2)))
print ("recall_score = {}".format(recall_score(y_test, best_prefiction2)))
print ("f1_score = {}".format(f1_score(y_test, best_prefiction2)))
```

```
accuracy_score = 0.8434273819235422
balanced_accuracy_score = 0.690910610556056
precision_score = 0.7676105780482096
recall_score = 0.41767477397173053
f1_score = 0.5409863104073891
```

[33]:
```
fig, ax = plt.subplots(1, 2, sharex='col', sharey='row', figsize=(15,5))
plot_confusion_matrix(neigh, X_test, y_test, cmap=plt.cm.Blues,␣
 ↪normalize='true',  ax=ax[0])
plot_confusion_matrix(clf_gs2.best_estimator_, X_test, y_test, cmap=plt.cm.
 ↪Blues, normalize='true',  ax=ax[1])

fig.suptitle('         ')
ax[0].title.set_text('K=5')
ax[1].title.set_text('K=25')
```



[35]:
```
fpr5, tpr5, thresholds5 = roc_curve(y_test, prediction, pos_label=1)
roc_auc_value5 = roc_auc_score(y_test, prediction, average='micro')

fpr25, tpr25, thresholds25 = roc_curve(y_test, best_prefiction2,␣
↪pos_label=1)
roc_auc_value25 = roc_auc_score(y_test, best_prefiction2, average='micro')
plt.figure()
lw = 2
plt.plot(fpr5, tpr5, color='darkorange', lw=lw, label='K=5, ROC curve (area␣
↪= %0.2f)' % roc_auc_value5)
plt.plot(fpr25, tpr25, color='green', lw=lw, label='K=25, ROC curve (area =␣
↪%0.2f)' % roc_auc_value25)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```



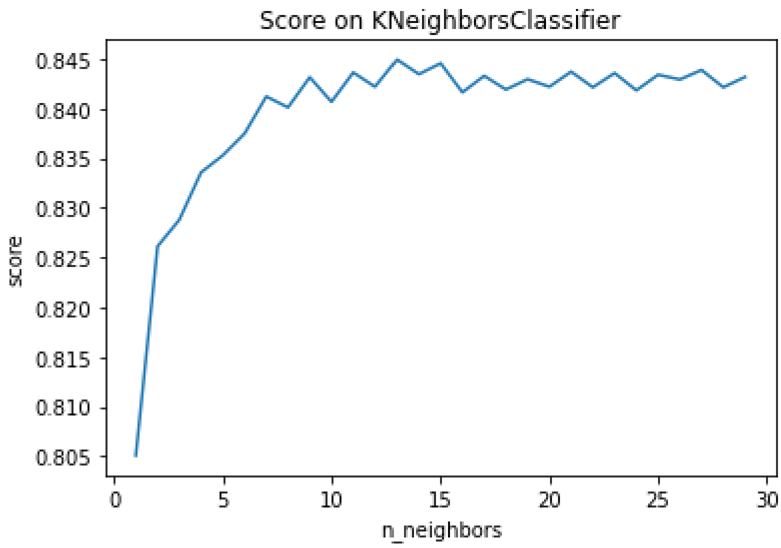
[17]:

```
%%time

n_nb = range(1, 30)
res = []

for i in n_nb:
  neigh = KNeighborsClassifier(n_neighbors=i)
  neigh.fit(X_train, y_train)

  prediction = neigh.predict(X_test)

  res.append((np.mean(prediction == y_test)))
```

```
CPU times: user 6min 20s, sys: 867 ms, total: 6min 21s
Wall time: 6min 21s
```

[18]:

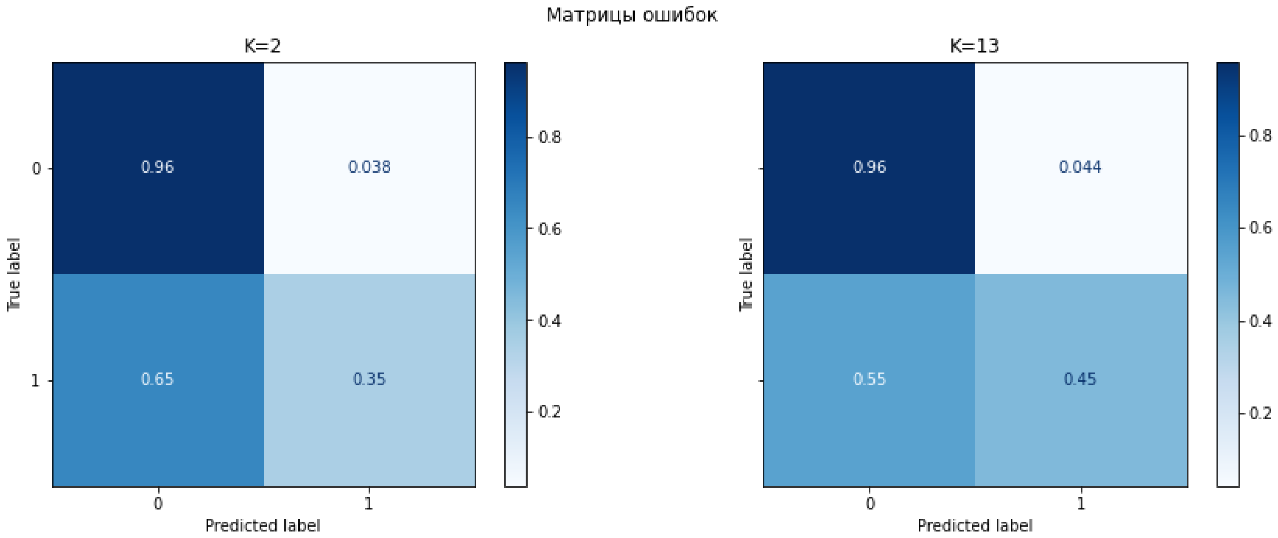
```
plt.plot(n_nb, res)
plt.xlabel("n_neighbors")
plt.ylabel("score")
plt.title("Score on KNeighborsClassifier")
plt.show()
```



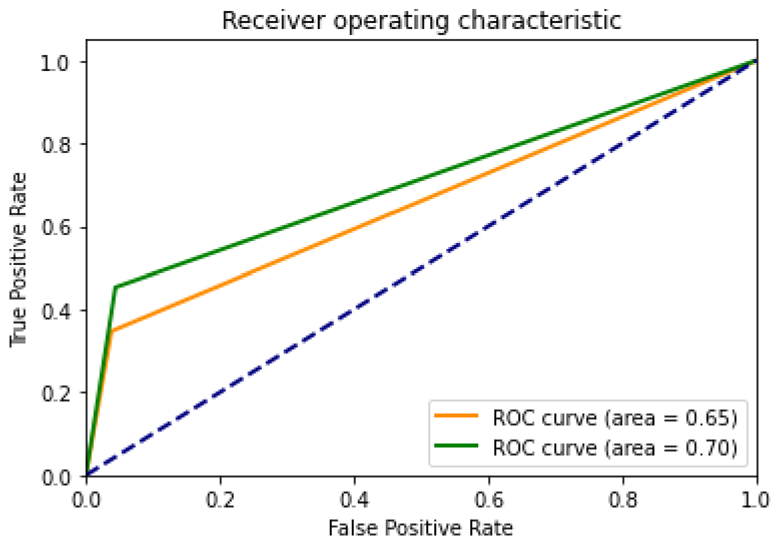
[29]:
```
%%time
neigh13 = KNeighborsClassifier(n_neighbors=13)
neigh13.fit(X_train, y_train)

prediction13 = neigh13.predict(X_test)
```

```
0.8449464119947115
CPU times: user 13.8 s, sys: 56.3 ms, total: 13.9 s
Wall time: 13.9 s
```

[45]:
```
fig, ax = plt.subplots(1, 2, sharex='col', sharey='row', figsize=(15,5))
plot_confusion_matrix(neigh, X_test, y_test, cmap=plt.cm.Blues,␣
 ↪normalize='true',  ax=ax[0])
plot_confusion_matrix(neigh13, X_test, y_test, cmap=plt.cm.Blues,␣
 ↪normalize='true',  ax=ax[1])

fig.suptitle('        ')
ax[0].title.set_text('K=5')
ax[1].title.set_text('K=13')
```



[49]:
```
fpr1, tpr1, thresholds1 = roc_curve(y_test, prediction, pos_label=1)
roc_auc_value1 = roc_auc_score(y_test, prediction, average='micro')

fpr2, tpr2, thresholds2 = roc_curve(y_test, prediction13, pos_label=1)
roc_auc_value2 = roc_auc_score(y_test, prediction13, average='micro')
plt.figure()
lw = 2
plt.plot(fpr1, tpr1, color='darkorange', lw=lw, label='K=5, ROC curve (area␣
↪= %0.2f)' % roc_auc_value1)
plt.plot(fpr2, tpr2, color='green', lw=lw, label='K=13, ROC curve (area =␣
↪%0.2f)' % roc_auc_value2)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```


Receiver operating characteristic
1.0
0.8
0.6
0.4
0.2
0.0
True Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate
ROC curve (area = 0.65)
ROC curve (area = 0.70)