Лабораторная работа №3

Робертса Даниила Александровича

Москва 2022

1 Лабораторная работа №3

Цель лабораторной работы: изучение способов подготовки выборки и подбора гиперпараметров на примере метода ближайших соседей. Задание: * Выберите набор данных (датасет) для решения задачи классификации или регрессии. * С использованием метода train_test_split разделите выборку на обучающую и тестовую. * Обучите модель ближайших соседей для произвольно заданного гиперпараметра K. Оцените качество модели с помощью подходящих для задачи метрик. * Произведите подбор гиперпараметра K с использованием GridSearchCV и/или RandomizedSearchCV и кросс-валидации, оцените качество оптимальной модели. Желательно использование нескольких стратегий кросс-валидации. * Сравните метрики качества исходной и оптимальной моделей.

[2]:
```
#Импорт библиотек:
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
import seaborn as sns
import time
from sklearn.datasets import make_classification
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report
```

Данные доступны по следующей ссылке: https://raw.githubusercontent.com/vadim0912/MLIntro2022_Spring/n

И представляют из себя данные о погоде в некоторых локациях. Целевая переменная - RainTomorrow (будет ли дождь завтра).

[3]:
```
X = pd.read_csv('weather.csv')
```

[4]:
```
y = X.RainTomorrow.replace({'No':0, 'Yes': 1})
```

[5]:
```
del X['RainTomorrow']
```

[6]:
```
X.count()
```

[6]:
```
Unnamed: 0       142193
Date             142193
Location         142193
```

```
MinTemp          141556
MaxTemp          141871
Rainfall         140787
Evaporation       81350
Sunshine          74377
WindGustDir      132863
WindGustSpeed    132923
WindDir9am       132180
WindDir3pm       138415
WindSpeed9am     140845
WindSpeed3pm     139563
Humidity9am      140419
Humidity3pm      138583
Pressure9am      128179
Pressure3pm      128212
Cloud9am          88536
Cloud3pm          85099
Temp9am          141289
Temp3pm          139467
RainToday        140787
dtype: int64
```

[7]:
```
X.isnull().sum()
```

[7]:
```
Unnamed: 0           0
Date                 0
Location             0
MinTemp            637
MaxTemp            322
Rainfall          1406
Evaporation      60843
Sunshine         67816
WindGustDir       9330
WindGustSpeed     9270
WindDir9am       10013
WindDir3pm        3778
WindSpeed9am      1348
WindSpeed3pm      2630
Humidity9am       1774
Humidity3pm       3610
Pressure9am      14014
Pressure3pm      13981
Cloud9am         53657
Cloud3pm         57094
Temp9am            904
Temp3pm           2726
RainToday         1406
```

```
dtype: int64
```

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

[8]:
```
Unnamed: 0           0
Date                 0
Location             0
MinTemp            637
MaxTemp            322
Rainfall          1406
Evaporation      60843
Sunshine         67816
WindGustDir       9330
WindGustSpeed     9270
WindDir9am       10013
WindDir3pm        3778
WindSpeed9am      1348
WindSpeed3pm      2630
Humidity9am       1774
Humidity3pm       3610
Pressure9am      14014
Pressure3pm      13981
Cloud9am         53657
Cloud3pm         57094
Temp9am            904
Temp3pm           2726
RainToday         1406
dtype: int64
```

Немного непонятно, почему, если в этот день нет дождя(Rain today), то есть количество осадков (Rainfall)

Тут я заметил, что дождь считается, что есть(RainToday), если Rainfall>1. В принципе, переменные сильно корелируют, поэтоу, возможноб в будущем избавлюсь от одной из них

[9]:
```
X.loc[X['RainToday'] == 'Yes'].min()
```

[9]:
```
Unnamed: 0               9
Date            2007-11-02
Location          Adelaide
MinTemp               -7.8
MaxTemp               -4.8
Rainfall               1.1
Evaporation            0.0
Sunshine               0.0
WindGustSpeed          7.0
WindSpeed9am           0.0
WindSpeed3pm           0.0
```

```
Humidity9am             1.0
Humidity3pm             1.0
Pressure9am           980.5
Pressure3pm           978.2
Cloud9am                0.0
Cloud3pm                0.0
Temp9am                -7.2
Temp3pm                -5.4
RainToday               Yes
dtype: object
```

[10]:
```
X.loc[X['RainToday'] == 'Yes'].isnull().sum()
```

[10]:
```
Unnamed: 0           0
Date                 0
Location             0
MinTemp             98
MaxTemp             65
Rainfall             0
Evaporation      13659
Sunshine         14759
WindGustDir       2380
WindGustSpeed     2356
WindDir9am        1735
WindDir3pm         942
WindSpeed9am       235
WindSpeed3pm       650
Humidity9am        445
Humidity3pm        902
Pressure9am       3047
Pressure3pm       3049
Cloud9am         10615
Cloud3pm         11377
Temp9am            208
Temp3pm            678
RainToday            0
dtype: int64
```

Удаляем айди и колонки, где данные отсутствуют в больших количествах, а также локацию

[11]:
```
del X['Unnamed: 0' ]
```

[12]:
```
del X['Cloud9am']
```

[13]:
```
del X['Evaporation']
del X['Cloud3pm']
del X['Sunshine']
```

[14]:
```
del X['Location']
```

[15]:
```
del X['RainToday']
```

[16]:
```
X.dtypes
```

[16]:
```
Date             object
MinTemp         float64
MaxTemp         float64
Rainfall        float64
WindGustDir      object
WindGustSpeed   float64
WindDir9am       object
WindDir3pm       object
WindSpeed9am    float64
WindSpeed3pm    float64
Humidity9am     float64
Humidity3pm     float64
Pressure9am     float64
Pressure3pm     float64
Temp9am         float64
Temp3pm         float64
dtype: object
```

[17]:
```
good_X=X.fillna(method='ffill')
```

[18]:
```
good_X.isnull().sum()
```

[18]:
```
Date            0
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
dtype: int64
```

[19]:
```
good_X.head()
```

[19]:
```
         Date  MinTemp  MaxTemp  Rainfall WindGustDir  WindGustSpeed  \
0  2008-12-01     13.4     22.9       0.6           W           44.0
1  2008-12-02      7.4     25.1       0.0         WNW           44.0
2  2008-12-03     12.9     25.7       0.0         WSW           46.0
3  2008-12-04      9.2     28.0       0.0          NE           24.0
4  2008-12-05     17.5     32.3       1.0           W           41.0

  WindDir9am WindDir3pm  WindSpeed9am  WindSpeed3pm  Humidity9am  Humidity3pm  \
0          W        WNW          20.0          24.0         71.0         22.0
1        NNW        WSW           4.0          22.0         44.0         25.0
2          W        WSW          19.0          26.0         38.0         30.0
3         SE          E          11.0           9.0         45.0         16.0
4        ENE         NW           7.0          20.0         82.0         33.0

   Pressure9am  Pressure3pm  Temp9am  Temp3pm
0       1007.7       1007.1     16.9     21.8
1       1010.6       1007.8     17.2     24.3
2       1007.6       1008.7     21.0     23.2
3       1017.6       1012.8     18.1     26.5
4       1010.8       1006.0     17.8     29.7
```

[20]:
```
good_X.Date=pd.to_datetime(X.Date)
```

[21]:
```
good_X.head()
```

[21]:
```
         Date  MinTemp  MaxTemp  Rainfall WindGustDir  WindGustSpeed  \
0 2008-12-01     13.4     22.9       0.6           W           44.0
1 2008-12-02      7.4     25.1       0.0         WNW           44.0
2 2008-12-03     12.9     25.7       0.0         WSW           46.0
3 2008-12-04      9.2     28.0       0.0          NE           24.0
4 2008-12-05     17.5     32.3       1.0           W           41.0

  WindDir9am WindDir3pm  WindSpeed9am  WindSpeed3pm  Humidity9am  Humidity3pm  \
0          W        WNW          20.0          24.0         71.0         22.0
1        NNW        WSW           4.0          22.0         44.0         25.0
2          W        WSW          19.0          26.0         38.0         30.0
3         SE          E          11.0           9.0         45.0         16.0
4        ENE         NW           7.0          20.0         82.0         33.0

   Pressure9am  Pressure3pm  Temp9am  Temp3pm
0       1007.7       1007.1     16.9     21.8
1       1010.6       1007.8     17.2     24.3
2       1007.6       1008.7     21.0     23.2
3       1017.6       1012.8     18.1     26.5
4       1010.8       1006.0     17.8     29.7
```

[22]:
```
good_X.dtypes
```

[22]:
```
Date             datetime64[ns]
MinTemp                 float64
MaxTemp                 float64
Rainfall                float64
WindGustDir              object
WindGustSpeed           float64
WindDir9am               object
WindDir3pm               object
WindSpeed9am            float64
WindSpeed3pm            float64
Humidity9am             float64
Humidity3pm             float64
Pressure9am             float64
Pressure3pm             float64
Temp9am                 float64
Temp3pm                 float64
dtype: object
```

[23]:
```
good_X=pd.get_dummies(good_X)
```

[24]:
```
good_X.head()
```

[24]:

	Date	MinTemp	MaxTemp	Rainfall	WindGustSpeed	WindSpeed9am	\
0	2008-12-01	13.4	22.9	0.6	44.0	20.0	
1	2008-12-02	7.4	25.1	0.0	44.0	4.0	
2	2008-12-03	12.9	25.7	0.0	46.0	19.0	
3	2008-12-04	9.2	28.0	0.0	24.0	11.0	
4	2008-12-05	17.5	32.3	1.0	41.0	7.0	

	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am	...	WindDir3pm_NNW	\
0	24.0	71.0	22.0	1007.7	...	0	
1	22.0	44.0	25.0	1010.6	...	0	
2	26.0	38.0	30.0	1007.6	...	0	
3	9.0	45.0	16.0	1017.6	...	0	
4	20.0	82.0	33.0	1010.8	...	0	

	WindDir3pm_NW	WindDir3pm_S	WindDir3pm_SE	WindDir3pm_SSE	WindDir3pm_SSW	\
0	0	0	0	0	0	
1	0	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	1	0	0	0	0	

	WindDir3pm_SW	WindDir3pm_W	WindDir3pm_WNW	WindDir3pm_WSW
0	0	0	1	0
1	0	0	0	1
2	0	0	0	1

```
3              0              0              0              0
4              0              0              0              0

[5 rows x 61 columns]
```

[25]:
```
good_X.dtypes
```

[25]:
```
Date              datetime64[ns]
MinTemp                  float64
MaxTemp                  float64
Rainfall                 float64
WindGustSpeed            float64
                          ...
WindDir3pm_SSW             uint8
WindDir3pm_SW              uint8
WindDir3pm_W               uint8
WindDir3pm_WNW             uint8
WindDir3pm_WSW             uint8
Length: 61, dtype: object
```

[26]:
```
good_X['dayofyear'] = good_X['Date'].dt.dayofyear
```

[27]:
```
del good_X['Date']
```

[28]:
```
from sklearn.preprocessing import StandardScaler,MinMaxScaler
sc2 = MinMaxScaler()
good_X = sc2.fit_transform(good_X)
```

[29]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(good_X, y, test_size=0.25,␣
 ↪random_state=10, shuffle = False)
```

Данные теперь выглядят лучше, и имеют формат либо флоат, либо uint8

[32]:
```
from sklearn.model_selection import GridSearchCV
```

KNN

[37]:
```
from sklearn.neighbors import KNeighborsClassifier
```

[62]:
```
neigh = KNeighborsClassifier(n_neighbors=13)
neigh.fit(X_train, y_train)
print(classification_report(y_test, neigh.predict(X_test)))
```

```
              precision    recall  f1-score   support

           0       0.82      0.98      0.89     27882
           1       0.70      0.20      0.32      7667
```

```
    accuracy                           0.81     35549
   macro avg       0.76      0.59      0.60     35549
weighted avg       0.79      0.81      0.77     35549
```

[39]:
```
from sklearn.feature_selection import chi2, SelectKBest, f_classif
```

[40]:
```
# будем использовать 5 лучших параметров
ft = SelectKBest(chi2, k = 5).fit(X_train, y_train)
print('Score: ', ft.scores_)
```

```
Score:  [4.83320880e+01 7.08541369e+01 4.72149631e+02 2.24661534e+02
 5.73909296e+01 4.54716015e+01 2.87953707e+02 1.63400966e+03
 1.00834332e+02 8.14512865e+01 8.73072541e-02 1.06231114e+02
 5.44671268e+01 8.01561050e+01 5.38149205e+00 7.28413457e+01
 5.61480088e+01 2.34671816e-01 5.12120136e+01 3.25997297e+01
 8.34456465e+00 1.80557277e+01 2.70782688e+00 4.98812288e-01
 6.13982423e+00 2.82950582e+01 1.30716095e+01 2.73446975e+00
 5.08153398e+01 1.11328889e+01 3.94658635e+01 1.88381683e+02
 1.66571620e+00 4.11383674e+01 1.99203810e+02 5.76046876e+00
 2.86669509e+01 5.95186373e+01 6.59942190e+01 4.01574504e-01
 3.27905507e-03 3.37259215e-01 1.26690617e+00 1.31084816e-01
 1.77465032e+01 4.94579694e+01 1.03046030e+01 1.00045929e+02
 4.67576039e+01 1.59476891e+01 7.06292529e+01 1.76152892e+01
 1.37516288e+00 6.67882046e-02 1.05529738e+00 1.23724282e+01
 7.91862669e+00 1.70570815e+00 4.67233604e-01 1.31577295e+01
 1.59172178e-01]
```

[41]:
```
X_train_2 = ft.transform(X_train)
X_test_2 = ft.transform(X_test)
```

[42]:
```
from sklearn import preprocessing
X_train_3 = preprocessing.StandardScaler().fit(X_train_2).transform(X_train_2.
 ↪astype(float))
X_test_3 = preprocessing.StandardScaler().fit(X_test_2).transform(X_test_2.
 ↪astype(float))
```

[43]:
```
grid_params = { 'n_neighbors' : [5,7,9,11,13,15, 17],
                'weights' : ['uniform','distance'],
                'metric' : ['minkowski','euclidean','manhattan']}
```

[44]:
```
gs = GridSearchCV(KNeighborsClassifier(), grid_params, verbose = 1, cv=3, n_jobs␣
 ↪= -1)
```

[45]:
```
g_res = gs.fit(X_train_3, y_train)
```

```
Fitting 3 folds for each of 42 candidates, totalling 126 fits
```

[46]:
```
g_res.best_params_
```

[46]:
```
{'metric': 'manhattan', 'n_neighbors': 17, 'weights': 'uniform'}
```

[47]:
```
knn = g_res.best_estimator_
```

[48]:
```
knn.fit(X_train_3, y_train)
```

[48]:
```
KNeighborsClassifier(metric='manhattan', n_neighbors=17)
```

[66]:
```
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
def print_metrics(y_test, y_pred):
    print(f"R^2: {r2_score(y_test, y_pred)}")
    print(f"MSE: {mean_squared_error(y_test, y_pred)}")
    print(f"MAE: {mean_absolute_error(y_test, y_pred)}")
```

[49]:
```
print(classification_report(y_test, knn.predict(X_test_3)))
```

```
              precision    recall  f1-score   support

           0       0.86      0.95      0.90     27882
           1       0.70      0.46      0.55      7667

    accuracy                           0.84     35549
   macro avg       0.78      0.70      0.73     35549
weighted avg       0.83      0.84      0.83     35549
```

До оптимизации:

[68]:
```
print_metrics(y_test, neigh.predict(X_test))
```

```
R^2: -0.12714442379714463
MSE: 0.1906664041182593
MAE: 0.1906664041182593
```

После:

[67]:
```
print_metrics(y_test, knn.predict(X_test_3))
```

```
R^2: 0.053118862628955266
MSE: 0.16017328194885933
MAE: 0.16017328194885933
```

2 Вывод:

При подборе параметров при помощи grid search предсказание модели стало гораздо точнее, хотя и не идеально.