jupyter Lab3 (несохраненные изменения)
Logout
File Edit View Insert Cell Kernel Widgets Help
Доверенный
Запуск Markdown
Python 3

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Технологии машинного обучения

Отчет по лабораторной работе № 3

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

Студент:
группы ИУ5-64Б
Ведягин Е.А.

2021 г.
Москва

Цель лабораторной работы:

изучение способов подготовки выборок и подбора гиперпараметров на примере метода ближайших соседей.

Задание:

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

Импорт библиотек

[0]: #code_ref_1_1
```
import numpy as np
import pandas as pd
from sklearn.datasets import *
from sklearn.model_selection import train_test_split
import seaborn as sns
import matplotlib.pyplot as plt
from operator import itemgetter
import matplotlib.ticker as ticker
%matplotlib inline
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 mean_absolute_error, mean_squared_error, mean_squared_log_error, median_absolute_error, r2_score
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier
from sklearn.model_selection import cross_val_score, cross_validate
from sklearn.model_selection import KFold, RepeatedKFold, LeaveOneOut, LeavePOut, ShuffleSplit, StratifiedKFold
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.model_selection import learning_curve, validation_curve
%matplotlib inline
sns.set(style="ticks")
```

Выборка датасета и ее разделение на тестовую и обучающую

[5]: #code_ref_1_2
```
# Загрузим данные
wbc = load_breast_cancer()
for k in list:
    print(k)

data
target
frame
target_names
DESCR
feature_names
filename
```

[6]: #code_ref_1_3
```
# Сформируем датасетом
wbc = pd.DataFrame(data=wbc.np.c_[wbc['data'], wbc['target']],
                   columns=list(wbc['feature_names']) + ['target'])
wbc_df
```

Out[6]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	worst radius	worst texture	worst perimeter	worst area	worst smoothness	com
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871	17.33	184.60	2019.0	0.16220		
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667	23.41	158.80	1956.0	0.12380		
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.05999	25.53	152.50	1709.0	0.14440		
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.09744	26.50	98.87	567.7	0.20980		
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.05883	16.67	152.20	1575.0	0.13740		
...																
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.05623	26.40	166.10	2027.0	0.14100		
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.05533	28.25	155.00	1731.0	0.11660		
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.05648	34.12	126.70	1124.0	0.11390		
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.07016	39.42	184.60	1821.0	0.16500		
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.05884	30.37	59.16	268.6	0.08996		

569 rows × 31 columns

[9]: #code_ref_1_4
```
bc_df.isnull().sum()
```

Out[9]:
```
mean radius                0
mean texture               0
mean perimeter             0
mean area                  0
mean smoothness            0
mean compactness           0
mean concavity             0
mean concave points        0
mean symmetry              0
mean fractal dimension     0
radius error               0
texture error              0
perimeter error            0
area error                 0
smoothness error           0
compactness error          0
concavity error            0
concave points error       0
symmetry error             0
fractal dimension error    0
worst radius               0
worst texture              0
worst perimeter            0
worst area                 0
worst smoothness           0
worst compactness          0
worst concavity            0
worst concave points       0
worst symmetry             0
worst fractal dimension    0
target                     0
dtype: int64
```

[8]: #code_ref_1_5
```
sc = MinMaxScaler()
bc_sc = sc.fit_transform(bc_data)
X_train, X_test, Y_train, Y_test = train_test_split(
bc_sc, bc_target, test_size=0.3, random_state=1)
X_train
```

Out[8]:
```
array([[0.5527431 , 0.2505182 , 0.5363402 , ..., 0.5725773 , 0.2619516,
        0.2164179 ],
       [0.2607217 , 0.2414904 , 0.2484273 , ..., 0.0956791 , 0.0070809,
        0.0636595],
       [0.282076 , 0.2905804 , 0.2709904 , ..., 0.2270904 , 0.1326373,
        0.1143258],
       ...,
       [0.4634007 , 0.5008545 , 0.4865593 , ..., 0.6525732 , 0.3465029,
        0.5172504],
       [0.3127148 , 0.3902604 , 0.3178773 , ..., 0.2754261 , 0.1362975,
        0.1305915],
       [0.2862899 , 0.2945529 , 0.2682066 , ..., 0.1732884 , 0.0831854,
        0.0636318 ]])
```

[13]: #code_ref_1_6
```
Y_train
```

Out[13]:
```
array([0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
```