МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Кафедра «Систем обработки информации и управления»

ОТЧЕТ

Рубежный контроль № 1

Выполнил:

студент группы ИУ5И-21М

Мьоу Зо У

Москва - 2021

Рубежный контроль (Вариант 16)

Задача №16.

Для набора данных проведите нормализацию для одного (произвольного) числового признака с использованием преобразования Бокса-Кокса (Box-Cox transformation).

Решение

Загрузка и предобработка данных

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import load_wine
from sklearn.datasets import load_boston
import scipy.stats as stats
from sklearn.svm import SVR
from sklearn.svm import LinearSVC
from sklearn.feature_selection import SelectFromModel
from sklearn.linear_model import Lasso
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import VarianceThreshold
from sklearn.feature_selection import mutual_info_classif, mutual_info_regression
from sklearn.feature_selection import SelectKBest, SelectPercentile
from IPython.display import Image
%matplotlib inline
sns.set(style="ticks")
```

```
def diagnostic_plots(df, variable):
    plt.figure(figsize=(15,6))
    # гистограмма
    plt.subplot(1, 2, 1)
    df[variable].hist(bins=30)
    ## Q-Q plot
    plt.subplot(1, 2, 2)
    stats.probplot(df[variable], dist="norm", plot=plt)
    plt.show()
```

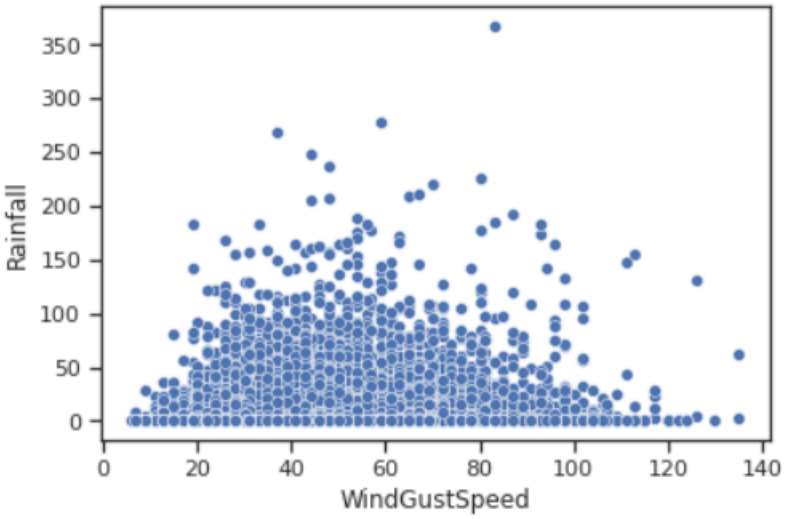
```
# Будем использовать только обучающую выборку
dataset = pd.read_csv('./weatherAUS.csv', sep=",")
```

```
dataset.head(10)
```

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindSpe
0	2008-12-01	Albury	13.400	22.900	0.600	nan	nan	W	44.000	W	WNW	20.000	24.000
1	2008-12-02	Albury	7.400	25.100	0.000	nan	nan	WNW	44.000	NNW	WSW	4.000	22.000
2	2008-12-03	Albury	12.900	25.700	0.000	nan	nan	WSW	46.000	W	WSW	19.000	26.000
3	2008-12-04	Albury	9.200	28.000	0.000	nan	nan	NE	24.000	SE	E	11.000	9.000
4	2008-12-05	Albury	17.500	32.300	1.000	nan	nan	W	41.000	ENE	NW	7.000	20.000
5	2008-12-06	Albury	14.600	29.700	0.200	nan	nan	WNW	56.000	W	W	19.000	24.000
6	2008-12-07	Albury	14.300	25.000	0.000	nan	nan	W	50.000	SW	W	20.000	24.000
7	2008-12-08	Albury	7.700	26.700	0.000	nan	nan	W	35.000	SSE	W	6.000	17.000
8	2008-12-09	Albury	9.700	31.900	0.000	nan	nan	NNW	80.000	SE	NW	7.000	28.000
9	2008-12-10	Albury	13.100	30.100	1.400	nan	nan	W	28.000	S	SSE	15.000	11.000

```
data['WindGustSpeed_boxcox'], param = stats.boxcox(data['WindGustSpeed'])
print('Оптимальное значение λ = {}'.format(param))
sns.scatterplot(x='WindGustSpeed',y='Rainfall',data=dataset)
```

```
Оптимальное значение λ = 8.472135811722177
<matplotlib.axes._subplots.AxesSubplot at 0x7faf57f4db90>
```



Задача №36.

Для набора данных проведите процедуру отбора признаков (feature selection). Используйте класс SelectKBest для 5 лучших признаков, и метод, основанный на взаимной информации.

```
wine = load_wine()
wine_X = wine.data
wine_y = wine.target
wine_feature_names = wine['feature_names']
wine_x_df = pd.DataFrame(data=wine['data'], columns=wine['feature_names'])
```

```
boston = load_boston()
boston_X = boston.data
boston_y = boston.target
boston_feature_names = boston['feature_names']
boston_x_df = pd.DataFrame(data=boston['data'], columns=boston['feature_names'])
```

```
sel_mi = SelectKBest(mutual_info_regression, k=5).fit(wine_X, wine_y)

list(zip(boston_feature_names, sel_mi.get_support()))
```

```
[('CRIM', True),
 ('ZN', False),
 ('INDUS', False),
 ('CHAS', False),
 ('NOX', False),
 ('RM', False),
 ('AGE', True),
 ('DIS', False),
 ('RAD', False),
 ('TAX', True),
 ('PTRATIO', False),
 ('B', True),
 ('LSTAT', True)]
```

```
boston_feature_names[sel_mi.get_support()]
```

```
array(['CRIM', 'AGE', 'TAX', 'B', 'LSTAT'], dtype='<U7')
```