Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Московский государственный технический университет имени Н.Э. Баумана» (МГТУ им. Н.Э. Баумана)

Рубежный контроль №1
по курсу «Методы машинного обучения»

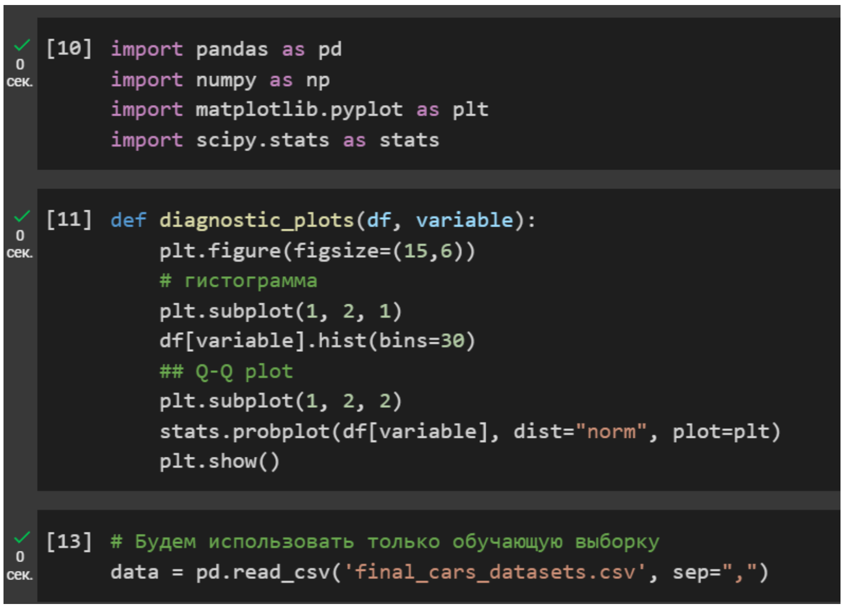
Выполнил
студент группы
ИУ5-24М
Старых Ф.А.

Москва, 2023

ИУ5-24М Вариант 13

Задача №13

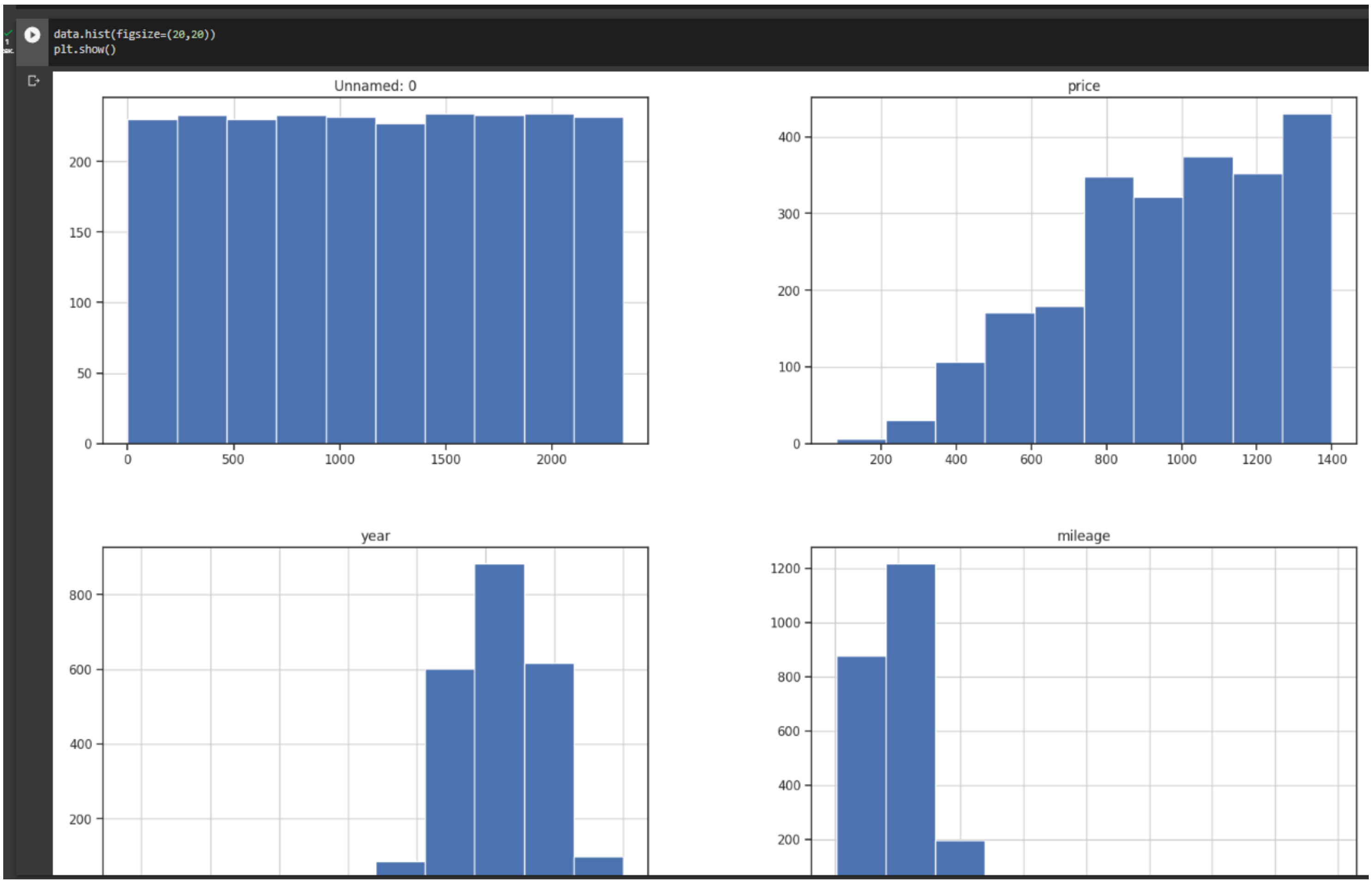
Для набора данных проведите нормализацию для одного (произвольного) числового признака с использованием функции "обратная зависимость - 1 / X".



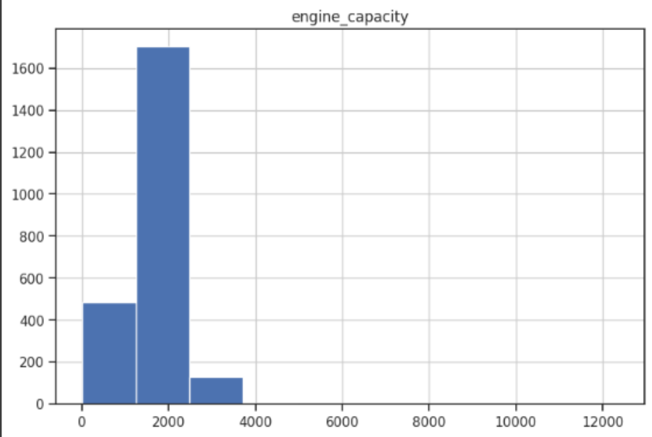
```
[10] import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     import scipy.stats as stats
```

```
[11] 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()
```

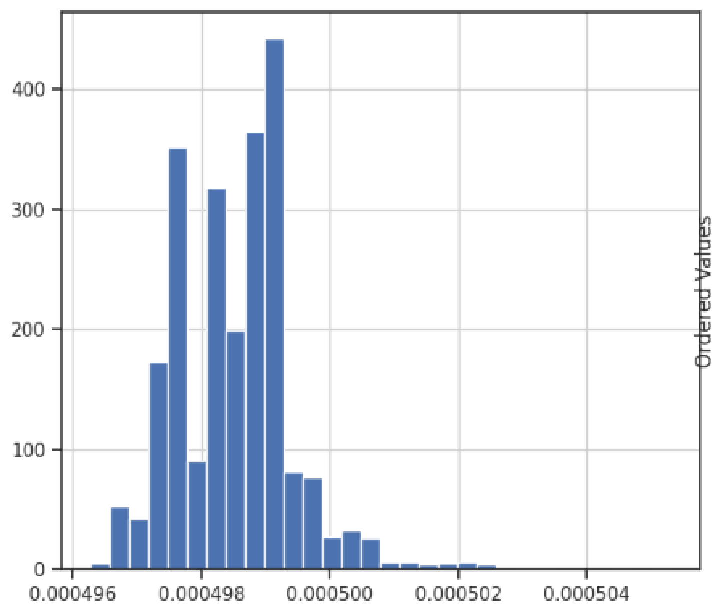
```
[13] # Будем использовать только обучающую выборку
     data = pd.read_csv('final_cars_datasets.csv', sep=",")
```

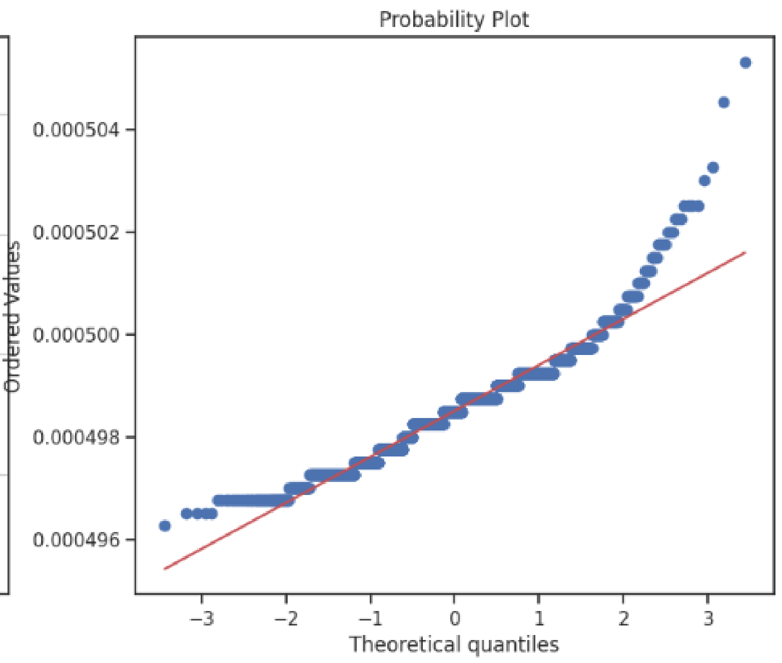


```
data.hist(figsize=(20,20))
plt.show()
```



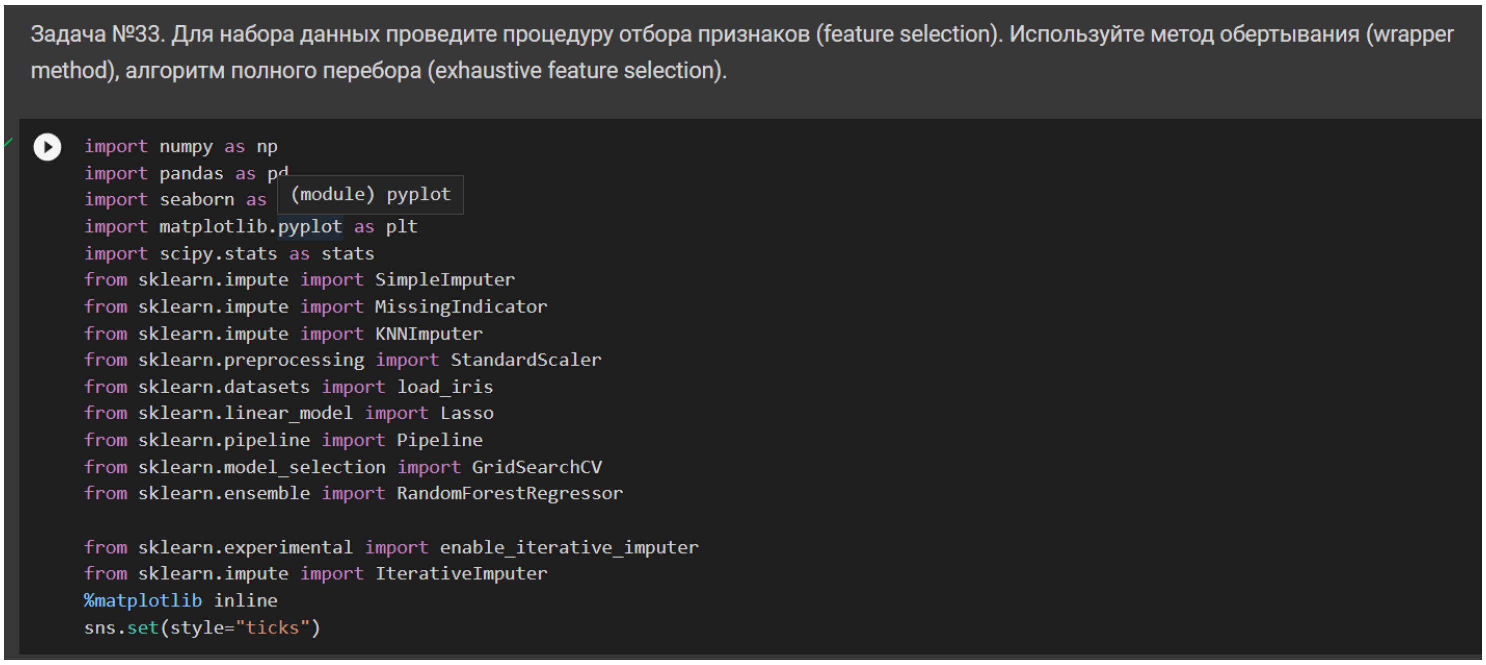
```
[15] data['price'] = 1 / (data['year'])
     diagnostic_plots(data, 'price')
```





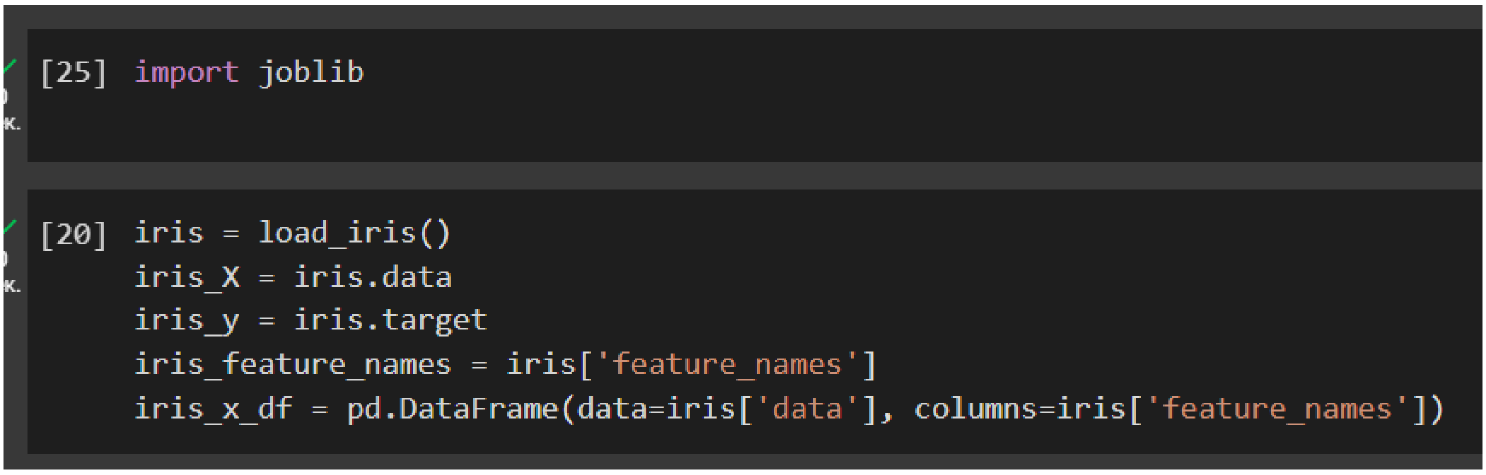
Задача №33

Для набора данных проведите процедуру отбора признаков (feature selection). Используйте метод обертывания (wrapper method), алгоритм полного перебора(exhaustive feature selection).



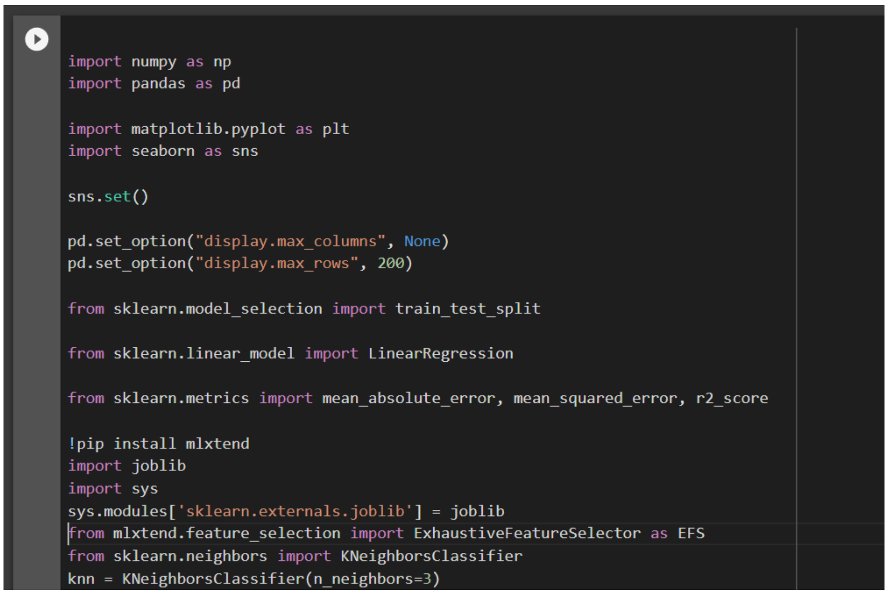
Задача №33. Для набора данных проведите процедуру отбора признаков (feature selection). Используйте метод обертывания (wrapper method), алгоритм полного перебора (exhaustive feature selection).

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats as stats
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
from sklearn.impute import KNNImputer
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import load_iris
from sklearn.linear_model import Lasso
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestRegressor

from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
%matplotlib inline
sns.set(style="ticks")
```

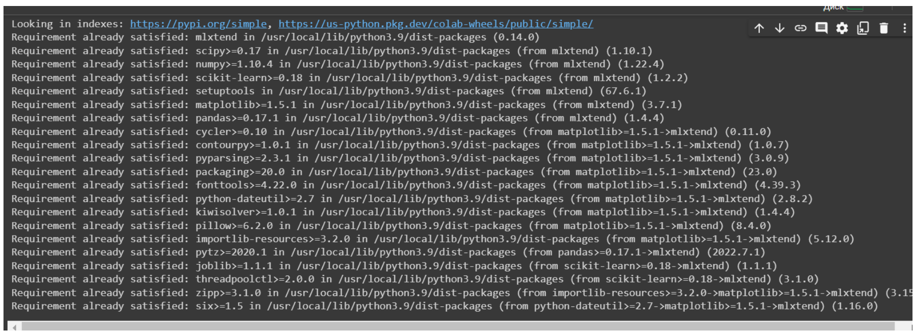


```
[25] import joblib
```

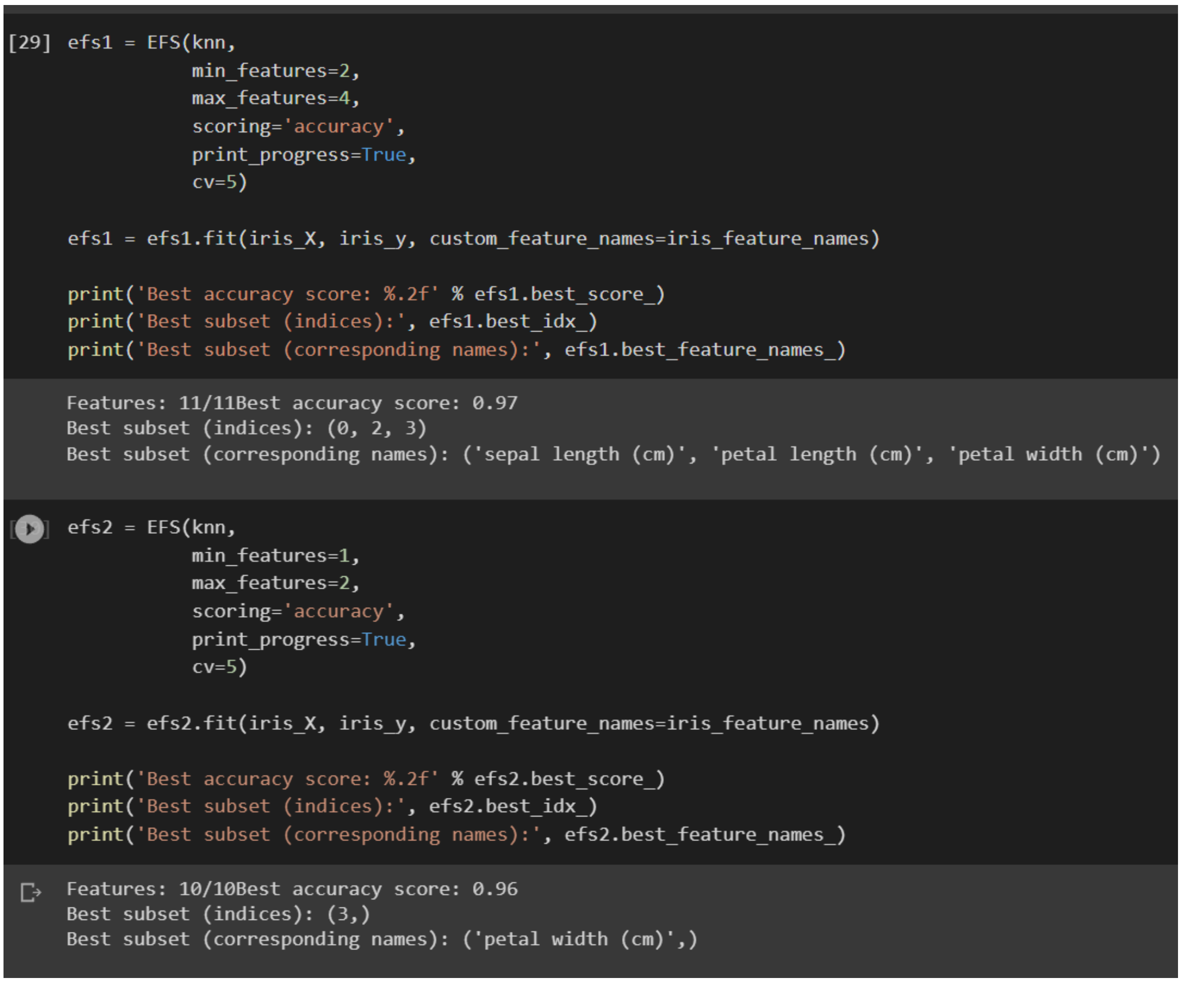
```
[20] iris = load_iris()
     iris_X = iris.data
     iris_y = iris.target
     iris_feature_names = iris['feature_names']
     iris_x_df = pd.DataFrame(data=iris['data'], columns=iris['feature_names'])
```



```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns

sns.set()

pd.set_option("display.max_columns", None)
pd.set_option("display.max_rows", 200)

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

!pip install mlxtend
import joblib
import sys
sys.modules['sklearn.externals.joblib'] = joblib
from mlxtend.feature_selection import ExhaustiveFeatureSelector as EFS
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
```



```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: mlxtend in /usr/local/lib/python3.9/dist-packages (0.14.0)
Requirement already satisfied: scipy>=0.17 in /usr/local/lib/python3.9/dist-packages (from mlxtend) (1.10.1)
Requirement already satisfied: numpy>=1.10.4 in /usr/local/lib/python3.9/dist-packages (from mlxtend) (1.22.4)
Requirement already satisfied: scikit-learn>=0.18 in /usr/local/lib/python3.9/dist-packages (from mlxtend) (1.2.2)
Requirement already satisfied: setuptools in /usr/local/lib/python3.9/dist-packages (from mlxtend) (67.6.1)
Requirement already satisfied: matplotlib>=1.5.1 in /usr/local/lib/python3.9/dist-packages (from mlxtend) (3.7.1)
Requirement already satisfied: pandas>=0.17.1 in /usr/local/lib/python3.9/dist-packages (from mlxtend) (1.4.4)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (0.11.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (1.0.7)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (3.0.9)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (23.0)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (4.39.3)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (1.4.4)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (8.4.0)
Requirement already satisfied: importlib-resources>=3.2.0 in /usr/local/lib/python3.9/dist-packages (from matplotlib>=1.5.1->mlxtend) (5.12.0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.9/dist-packages (from pandas>=0.17.1->mlxtend) (2022.7.1)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.9/dist-packages (from scikit-learn>=0.18->mlxtend) (1.1.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.9/dist-packages (from scikit-learn>=0.18->mlxtend) (3.1.0)
Requirement already satisfied: zipp>=3.1.0 in /usr/local/lib/python3.9/dist-packages (from importlib-resources>=3.2.0->matplotlib>=1.5.1->mlxtend) (3.15
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages (from python-dateutil>=2.7->matplotlib>=1.5.1->mlxtend) (1.16.0)
```



```
[29] efs1 = EFS(knn,
               min_features=2,
               max_features=4,
               scoring='accuracy',
               print_progress=True,
               cv=5)

     efs1 = efs1.fit(iris_X, iris_y, custom_feature_names=iris_feature_names)

     print('Best accuracy score: %.2f' % efs1.best_score_)
     print('Best subset (indices):', efs1.best_idx_)
     print('Best subset (corresponding names):', efs1.best_feature_names_)
```

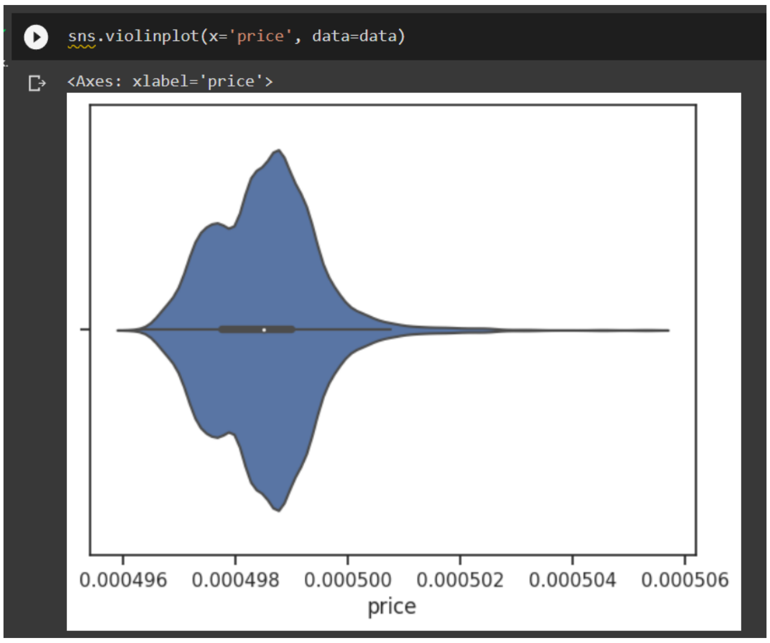
```
Features: 11/11Best accuracy score: 0.97
Best subset (indices): (0, 2, 3)
Best subset (corresponding names): ('sepal length (cm)', 'petal length (cm)', 'petal width (cm)')
```

```
efs2 = EFS(knn,
           min_features=1,
           max_features=2,
           scoring='accuracy',
           print_progress=True,
           cv=5)

efs2 = efs2.fit(iris_X, iris_y, custom_feature_names=iris_feature_names)

print('Best accuracy score: %.2f' % efs2.best_score_)
print('Best subset (indices):', efs2.best_idx_)
print('Best subset (corresponding names):', efs2.best_feature_names_)
```

```
Features: 10/10Best accuracy score: 0.96
Best subset (indices): (3,)
Best subset (corresponding names): ('petal width (cm)',)
```

Доп задание

Для студентов группы ИУ5-24М
- для произвольной колонки данных построить график
 "Скрипичная диаграмма (violin plot)"



```
sns.violinplot(x='price', data=data)
```

```
<Axes: xlabel='price'>
```