Московский Государственный Технический Университет имени

Н.Э.Баумана

Факультет Информатика и системы управления

Кафедра ИУ-5

«Методы машинного обучения»

Рубежный контроль №1

По дисциплине

Выполнили студент группы ИУ-5 23М

Юй Шанчэнь

Москва 2022г

Задача №19

Для набора данных проведите масштабирование данных для одного (произвольного) числового признака с использованием метода "Mean Normalisation".

Загрузка и первичный анализ данных Используем данные из StudentsPerformance.

In [13]:

```
import seaborn as sns
import matplotlib.pyplot as plt
import os
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris
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")
df =pd.read_csv("D:/lab1/StudentsPerformance.csv")
```

In [7]:

```
df.head()
```

Out[7]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

In [9]:

```
df.isnull().sum()
```

Out[9]:

```
gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
dtype: int64
```

In [8]:

```
df.info()
```

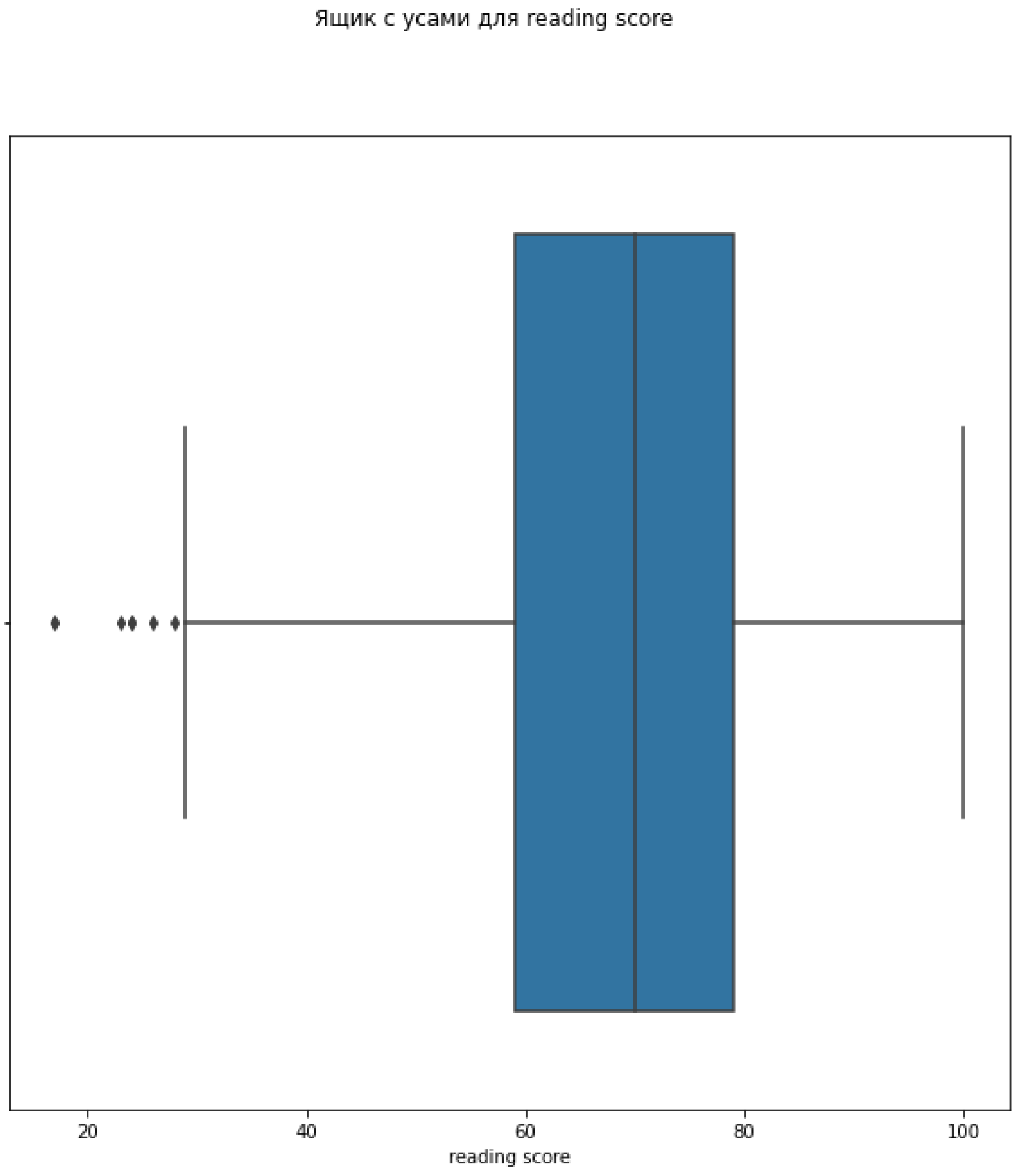
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64 
 6   reading score                1000 non-null   int64 
 7   writing score                1000 non-null   int64 
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

Дополнительное требование по группам

Для студентов групп ИУ5-23М, ИУ5И-23М - для произвольной колонки данных построить график "Ящик с усами (boxplot)".

In [23]:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.boxplot(x=df['reading score'])
fig.suptitle('Ящик с усами для reading score')
plt.show()
```



Выбрать числовые признаки "math score" и "reading score"

In [9]:

```
col = pd.DataFrame(df, columns=["math score","reading score"])
col.head()
```

Out[9]:

	math score	reading score
0	72	72
1	69	90
2	90	95
3	47	57
4	76	78

In [ ]:

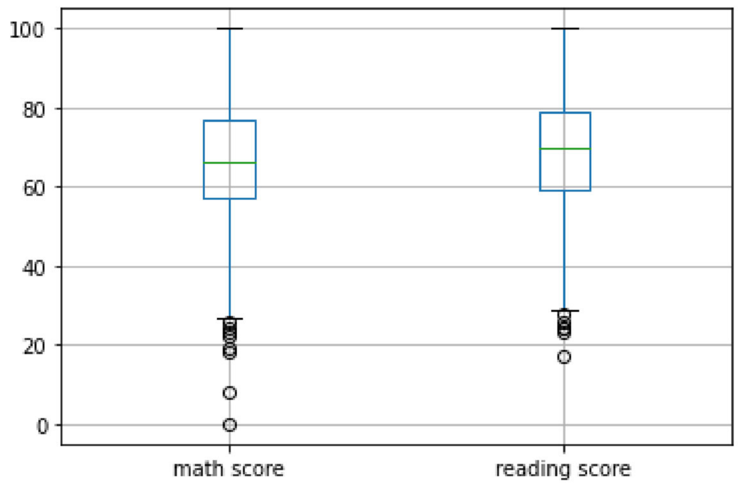
```

```

In [29]:

```
df.boxplot(column=['math score','reading score'])
```

Out[29]:

```
<AxesSubplot:>
```



In [ ]:

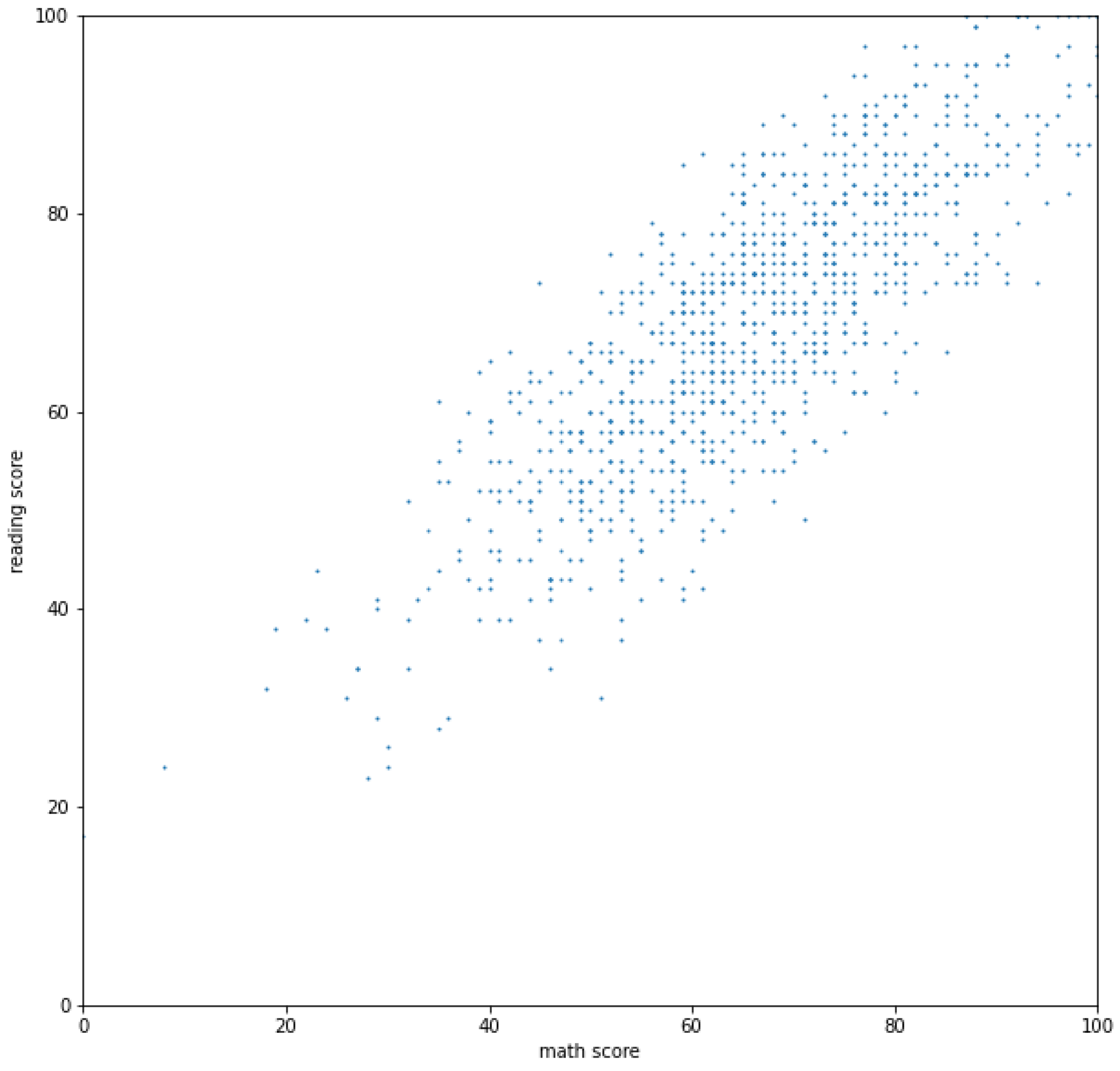
```

```

In [27]:

```
ax = df.plot.scatter(x='math score',y='reading score',s=1,figsize=(10,10))
ax.set_xlim(0, 100)
ax.set_ylim(0, 100)
```

Out[27]:

```
(0.0, 100.0)
```



Задача №19 Для набора данных проведите масштабирование данных для одного (произвольного) числового признака с использованием метода "Mean Normalisation".

Масштабирование данных для признаков "math score" и "reading score"с использованием метода "Mean Normalisation"

In [34]:

```
hdata = df.loc[:,['math score','reading score']]
print('min(math score) = ' + str(np.min(hdata['math score'])))
print('max(math score) = ' + str(np.max(hdata['math score'])))
print('min(reading score) = ' + str(np.min(hdata['reading score'])))
print('max(reading score) = ' + str(np.max(hdata['reading score'])))
```

```
min(math score) = 0
max(math score) = 100
min(reading score) = 17
max(reading score) = 100
```

In [ ]:

In [35]:

```
hdata = hdata.apply(lambda x: (x - np.mean(x))/(np.max(x)-np.min(x)))
hdata.columns = ['Normalized math score','Normalized reading score']
hdata.head()
```

Out[35]:

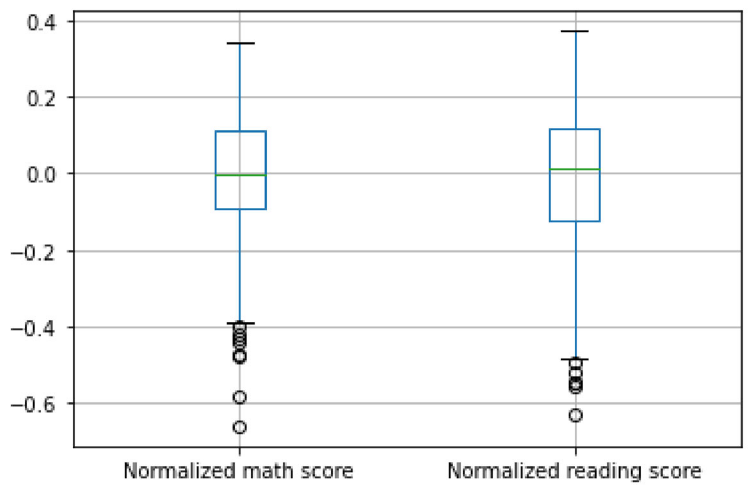
	Normalized math score	Normalized reading score
0	0.05911	0.034108
1	0.02911	0.250976
2	0.23911	0.311217
3	-0.19089	-0.146614
4	0.09911	0.106398

In [38]:

```
hdata.boxplot(column=['Normalized math score','Normalized reading score'])
```

Out[38]:

```
<AxesSubplot:>
```



Задача №39.

Для набора данных проведите процедуру отбора признаков (feature selection). Используйте класс SelectPercentile для 10% лучших признаков, и метод, основанный на взаимной информации.

In [16]:

```
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'])
```

In [17]:

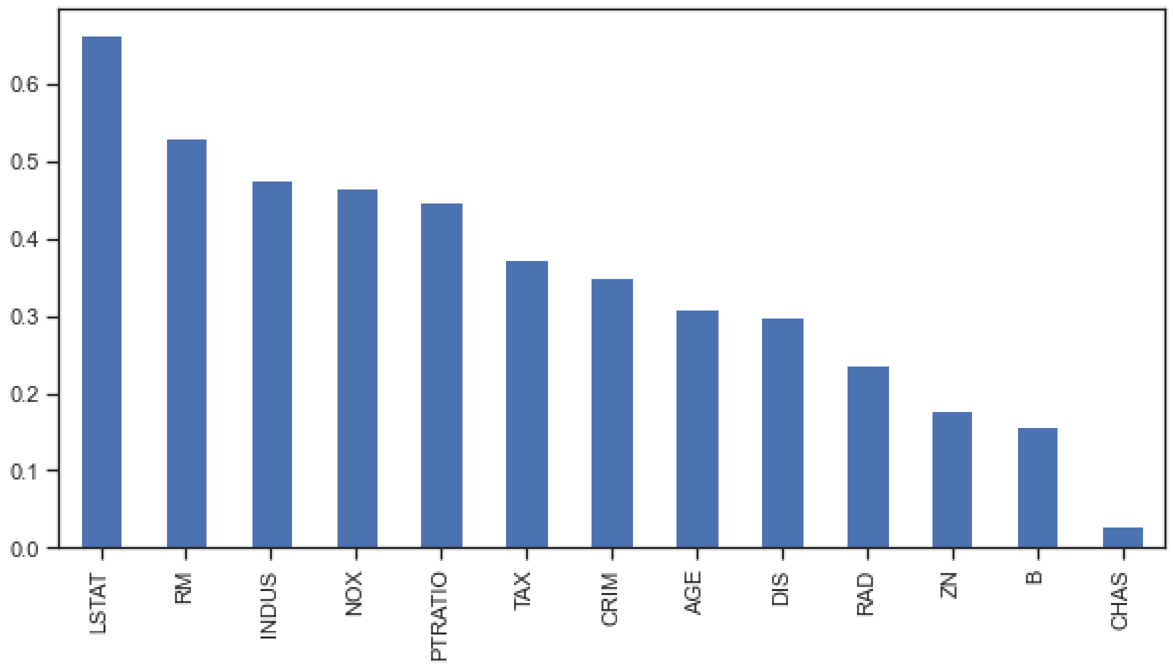
```
boston = load_boston()
boston_X = boston.data
boston_y = boston.target
boston_feature_names = boston['feature_names']
boston_x_df = pd.DataFrame(data=boston['data'], columns=boston['feature_names'])
```

In [18]:

```
mi = mutual_info_regression(boston_X, boston_y)
mi = pd.Series(mi)
mi.index = boston_feature_names
mi.sort_values(ascending=False).plot.bar(figsize=(10,5))
```

Out[18]:

```
<AxesSubplot:>
```



In [19]:

```
sel_mi = SelectPercentile(mutual_info_regression, percentile=10).fit(boston_X, boston_y)

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

Out[19]:

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

In [ ]:

```

```