Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»

Отчет
Лабораторная работа № 1
По курсу «Методы машинного обучения»

Истории о данных

ИСПОЛНИТЕЛЬ:
Попов Илья Андреевич
Группа ИУ5-23М

_____

"__"_____2022 г.

ПРЕПОДАВАТЕЛЬ:
Гапанюк Ю.Е.

_____

"__"_____2022 г.

Москва 2022

Задание

- Выбрать набор данных (датасет).

- Создать "историю о данных" в виде юпитер-ноутбука, с учетом следующих требований:

1. История должна содержать не менее 5 шагов (где 5 - рекомендуемое количество шагов). Каждый шаг содержит график и его текстовую интерпретацию.
2. На каждом шаге наряду с удачным итоговым графиком рекомендуется в юпитер-ноутбуке оставлять результаты предварительных "неудачных" графиков.
3. Не рекомендуется повторять виды графиков, желательно создать 5 графиков различных видов.
4. Выбор графиков должен быть обоснован использованием методологии data-to-viz. Рекомендуется учитывать типичные ошибки построения выбранного вида графика по методологии data-to-viz. Если методология Вами отвергается, то просьба обосновать Ваше решение по выбору графика.
5. История должна содержать итоговые выводы. В реальных "историях о данных" именно эти выводы представляют собой основную ценность для предприятия.

- Сформировать отчет и разместить его в своем репозитории на github.

Выполнение в среде Jupyter Notebook:

Popov I.A. lab1

In [105…
```
import numpy as np
import pandas as pd
import seaborn as sns;
import matplotlib.pyplot as plt
%matplotlib inline
```

In [4]:
```
raw_data = pd.read_csv('../RK_1/StudentsPerformance.csv', sep=',')
```

In [5]:
```
raw_data.head()
```

Out[5]:

	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 [6]:
```
raw_data.nunique()
```

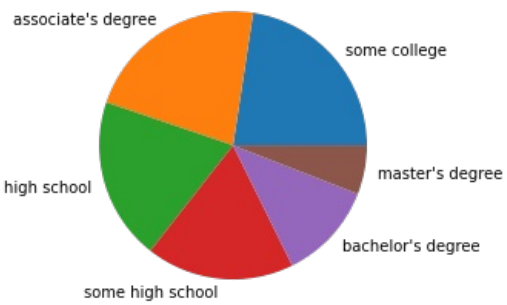
Out[6]:
```
gender                         2
race/ethnicity                 5
parental level of education    6
lunch                          2
test preparation course        2
math score                    81
reading score                 72
writing score                 77
dtype: int64
```

In [21]:
```
raw_data.keys()
```

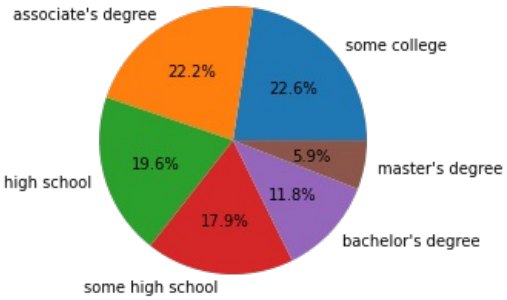
Out[21]:
```
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'test preparation course', 'math score', 'reading score',
       'writing score'],
      dtype='object')
```

In [85]:
```
#готовим данные

genders = raw_data.get('gender').value_counts()
races = raw_data.get('race/ethnicity').value_counts(sort=False)
parents_edu_s = raw_data.get('parental level of education').value_counts()
lunches = raw_data.get('lunch').drop_duplicates().value_counts()
prep_cources = raw_data.get('test preparation course').value_counts()

maths = raw_data.get('math score').to_numpy()
reads = raw_data.get('reading score').to_numpy()
writes = raw_data.get('writing score').to_numpy()
```

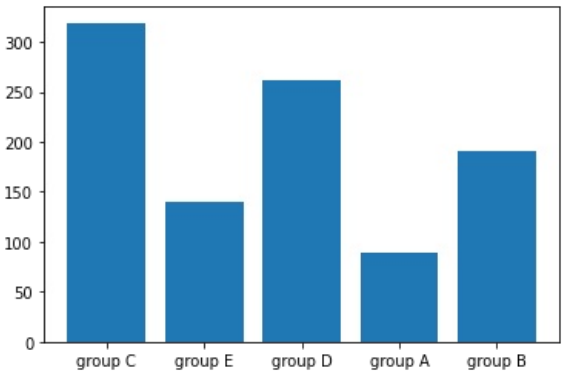
In [50]:
```
#pie процентное соотношение степеней образования родителей студентов

values1 = parents_edu_s.to_numpy()
plt.pie(values1, labels=parents_edu_s.keys())
plt.show()
```



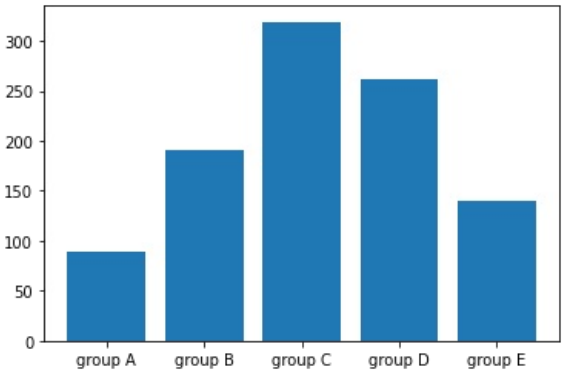
In [51]:
```
#pie (более удачный)

values1 = parents_edu_s.to_numpy()
plt.pie(values1, labels=parents_edu_s.keys(), autopct='%1.1f%%')
plt.show()
```



In [56]:
```
#bars количественое отображение студентов по этническим группам

values2 = races.to_numpy()
x = np.arange(len(races))
plt.bar(x, values2)
plt.xticks(x, races.keys())
plt.show()
```

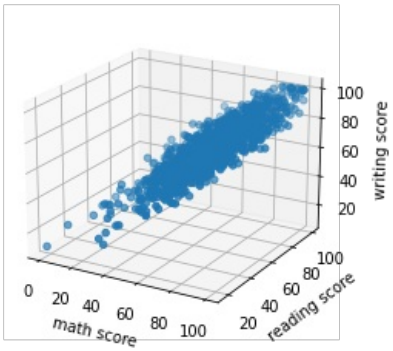


In [81]:
```
#bars (колонки отсортированы по алфавиту)

races_data = np.unique(np.sort(raw_data.get('race/ethnicity').to_numpy()), return_counts=True)

values2 = races_data[1] #counted unique values
races_labels = races_data[0] #labels
x = np.arange(len(races))
plt.bar(x, values2)
plt.xticks(x, races_labels)
plt.show()
```



In [104…
```
#scatter баллы по всем предметам на трёхмерном графике


score_data = zip(maths, reads, writes)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

#for x, y, z in score_data:
```

```
ax.scatter(maths, reads, writes)

ax.set_xlabel('math score')
ax.set_ylabel('reading score')
ax.set_zlabel('writing score')

ax.view_init(20, 300)

plt.show()
```

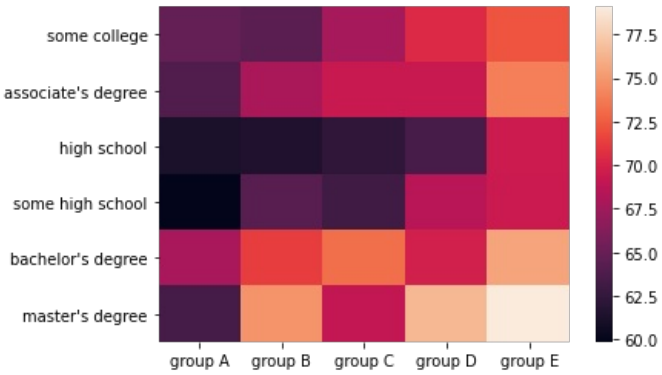


In [165…

```
#heatmap средний балл по всем предмеам в зависимости от этнической пренадлежности и образования родителей

hm_df = pd.DataFrame(columns=parents_edu_s.keys())

temp_row = []
for race in races_data[0]:
    for parent_edu in parents_edu_s.keys():
        filter_race = raw_data.where(raw_data['race/ethnicity'] == race).dropna()
        filter_prnt_edu = filter_race.where(filter_race['parental level of education'] == parent_edu).dropna()
        temp_row.append(np.average(filter_prnt_edu.get(['math score','reading score','writing score']).values))
    hm_df = pd.concat([hm_df, pd.DataFrame(data=[temp_row], index=[race], columns=parents_edu_s.keys())])
    temp_row = []

sns.heatmap(hm_df.T)
```

Out[165…

```
<AxesSubplot:>
```



In [172…

```
#boxplot "ящик с усами" для оценок по всем предметам

sns.boxplot(data=raw_data.get(['math score','reading score','writing score']))
plt.show()
```