Лабораторная работа №1. Первичный анализ данных

Цель лабораторной работы: изучение программных средств для организации рабочего места специалиста по анализу данных и машинному обучению.

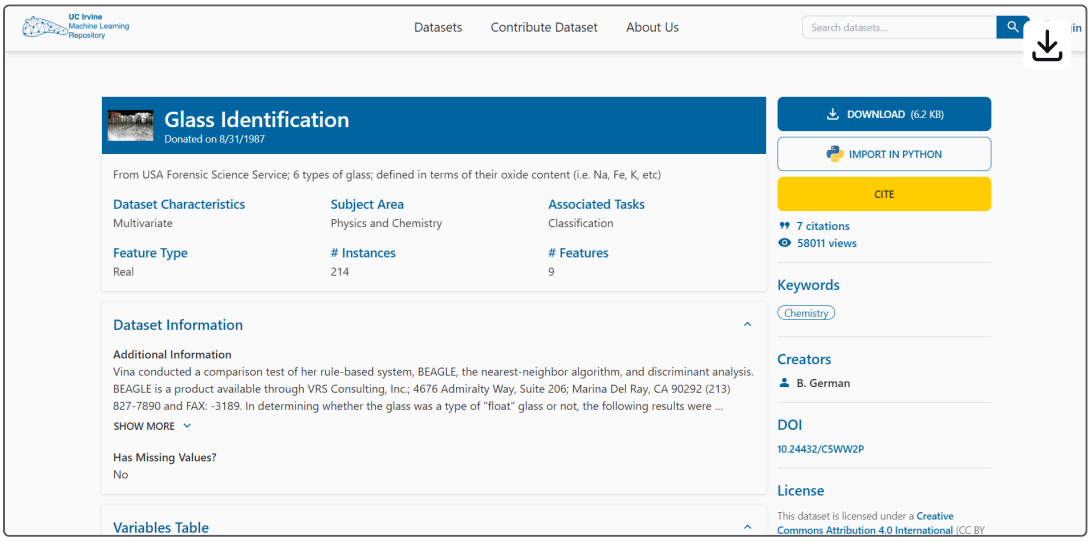
Основные задачи:

- получение программного доступа к данным, содержащимся в источниках различного типа;
- выполнение предварительного анализа данных и получение обобщенных характеристик наборов данных;
- исследование простых методов визуализации данных;
- изучение основных библиотек Python для работы с данными

Ход выполнения индивидуального задания:

1. Выбор набора данных

В качестве данных для лабораторной работы был выбран набор под названием **Glass Identification**, который входит в репозиторий **UCI Machine Learning Repository**.



2. Первичный анализ данных

2.1 Описание набора данных

Набор данных содержит результаты химического анализа различных типов стекла. Каждый образец описывается рядом **физических** и **химических** характеристик, включая **показатель преломления** и **процентное содержание различных оксидов**.

Цель набора данных — классифицировать образцы стекла по их типам на основе указанных характеристик.

Задачи, которые можно решить с использованием данного обучающего набора данных:

- **Криминалистика и судебная экспертиза**
- **Контроль качества и производство стекла**
- **Экологические и геологические исследования**

Описание признаков:

Признак	Описание	Тип
Id_number	Номер ID	Integer
RI	Коэффициент преломления	Continuous
Na	Натрий (вес. %)	Continuous
Mg	Магний (вес. %)	Continuous
Al	Алюминий (вес. %)	Continuous
Si	Кремний (вес. %)	Continuous
K	Калий (вес. %)	Continuous
Ca	Кальций (вес. %)	Continuous
Ba	Барий (вес. %)	Continuous
Fe	Железо (вес. %)	Continuous

2.2 Форма набора данных

Количество элементов набора: 214

Количество признаков: 10

Количество пропущенных значений: 0

Статистические показатели отдельных признаков:

Поиск среднего значения:

```
Среднее значение Rl: 1.5183654205607477
Среднее значение Na: 13.407850467289
Среднее значение Mg: 2.684532710280374
Среднее значение Al: 1.444906542056075
Среднее значение Si: 72.65093457943925
Среднее значение K: 0.4970560747663551
Среднее значение Ca: 8.95696261682243
Среднее значение Ba: 0.17504672897196263
Среднее значение Fe: 0.05700934579439253
```

Код для поиска среднего значения:

```python
import numpy as np
import matplotlib.pyplot as plt

data_path = "glass.data"
data = np.genfromtxt(data_path, delimiter=",")

Rl = []
Na = []
Mg = []
Al = []
Si = []
K = []
Ca = []
Ba = []
Fe = []

for dot in data:
    Rl.append(dot[1])
    Na.append(dot[2])
    Mg.append(dot[3])
    Al.append(dot[4])
    Si.append(dot[5])
    K.append(dot[6])
    Ca.append(dot[7])
    Ba.append(dot[8])
    Fe.append(dot[9])

print("Среднее значение Rl:", np.mean(Rl))
print("Среднее значение Na:", np.mean(Na))
print("Среднее значение Mg:", np.mean(Mg))
print("Среднее значение Al:", np.mean(Al))
print("Среднее значение Si:", np.mean(Si))
print("Среднее значение K:", np.mean(K))
print("Среднее значение Ca:", np.mean(Ca))
print("Среднее значение Ba:", np.mean(Ba))
print("Среднее значение Fe:", np.mean(Fe))
```

Поиск максимального значения:

```
Максимальное значение Rl: 1.53393
Максимальное значение Na: 17.3
Максимальное значение Mg: 4.49
Максимальное значение Al: 3.5
Максимальное значение Si: 75.41
Максимальное значение K: 6.21
Максимальное значение Ca: 16.19
Максимальное значение Ba: 3.15
Максимальное значение Fe: 0.51
```

Код для поиска максимального значения:

```
print("Максимальное значение Rl:", np.max(Rl))
print("Максимальное значение Na:", np.max(Na))
print("Максимальное значение Mg:", np.max(Mg))
print("Максимальное значение Al:", np.max(Al))
print("Максимальное значение Si:", np.max(Si))
print("Максимальное значение K:", np.max(K))
print("Максимальное значение Ca:", np.max(Ca))
print("Максимальное значение Ba:", np.max(Ba))
print("Максимальное значение Fe:", np.max(Fe))
```

Поиск минимального значения:

```
Минимальное значение Rl: 1.51115
Минимальное значение Na: 10.73
Минимальное значение Mg: 0.0
Минимальное значение Al: 0.29
Минимальное значение Si: 69.81
Минимальное значение K: 0.0
Минимальное значение Ca: 5.43
Минимальное значение Ba: 0.0
Минимальное значение Fe: 0.0
```

Код для поиска минимального значения:

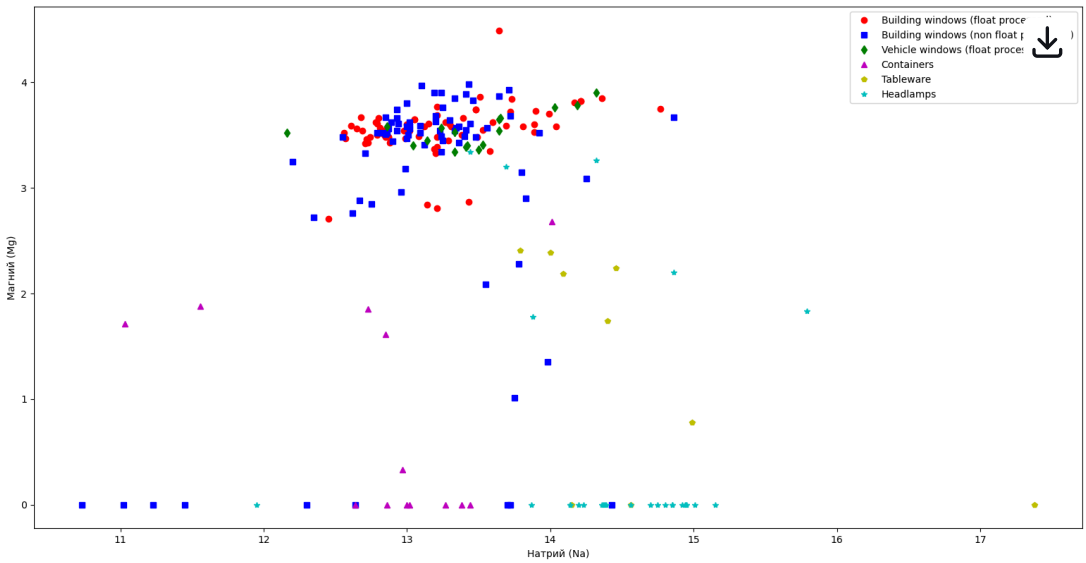
```
print("Минимальное значение Rl:", np.min(Rl))
print("Минимальное значение Na:", np.min(Na))
print("Минимальное значение Mg:", np.min(Mg))
print("Минимальное значение Al:", np.min(Al))
print("Минимальное значение Si:", np.min(Si))
print("Минимальное значение K:", np.min(K))
print("Минимальное значение Ca:", np.min(Ca))
print("Минимальное значение Ba:", np.min(Ba))
print("Минимальное значение Fe:", np.min(Fe))
```

Предположения на основе первичного анализа:

- Можно заметить, что у некоторых признаков, а именно у **магния (Mg)**, **калия (K)**, **бария (Ba)** и **железа (Fe)** минимальное значение равно **нулю**. Это свидетельствует об **отсутствии** данных материалов в некоторых образцах стекла;

- **Большое** значение у **кремния (Si)** довольно ожидаемо, так как он является **основным** компонентом стекла.

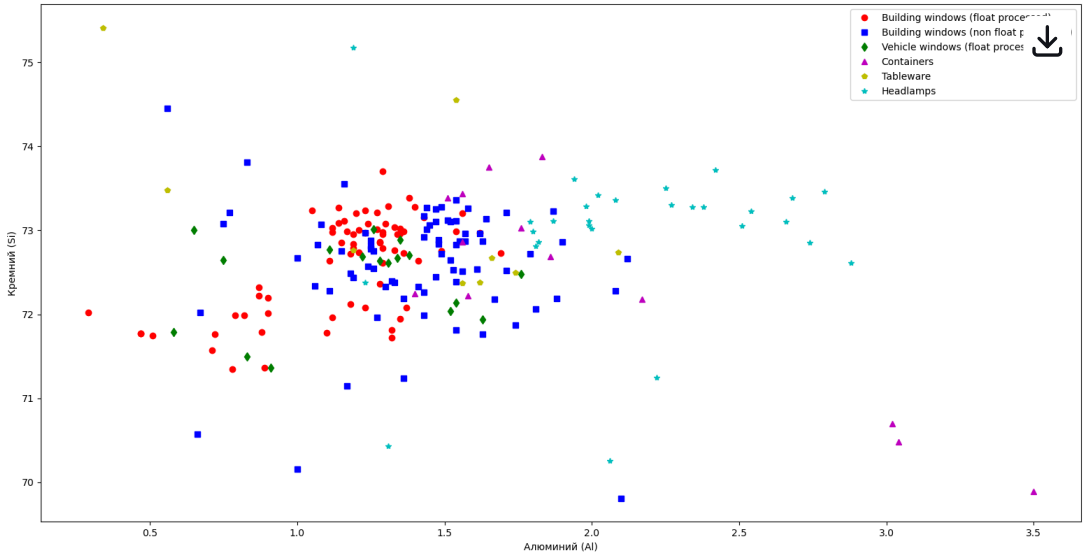
3. Графические представления набора данных



Проекция №1. Натрий и магний

Код для визуализации проекции:

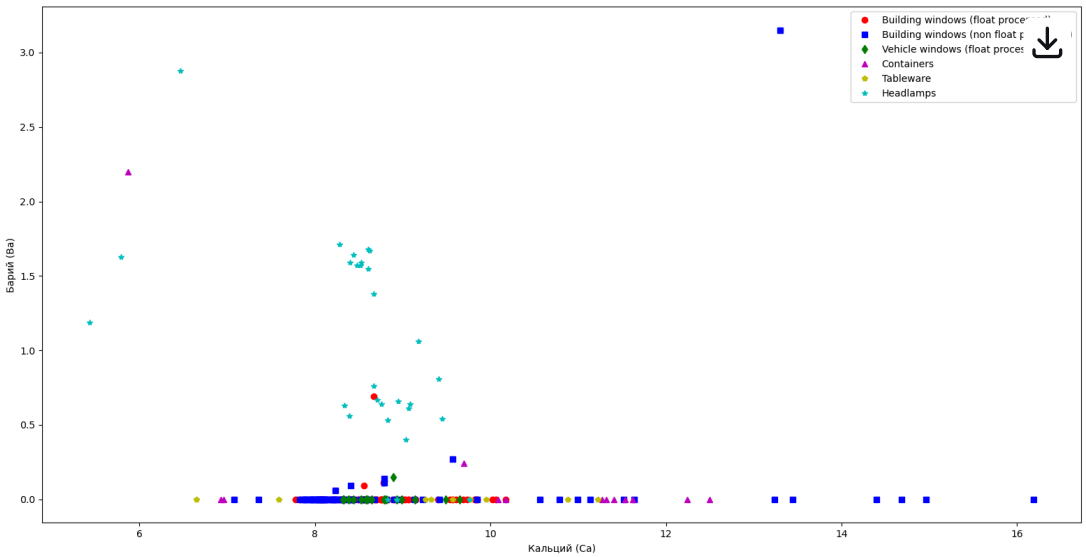
```
plt.figure(1)
plt.plot(Na[:70], Mg[:70], 'ro', label='Building windows (float processed)
plt.plot(Na[70:146], Mg[70:146], 'bs', label='Building windows (non float
plt.plot(Na[146:163], Mg[146:163], 'gd', label='Vehicle windows (float prc
plt.plot(Na[163:176], Mg[163:176], 'm^', label='Containers')
plt.plot(Na[176:185], Mg[176:185], 'yp', label='Tableware')
plt.plot(Na[185:214], Mg[185:214], 'c*', label='Headlamps')
plt.legend()
plt.xlabel('Натрий (Na)')
plt.ylabel('Магний (Mg)')
```



Проекция №2. Алюминий и кремний

Код для визуализации проекции:

```
plt.figure(2)
plt.plot(Al[:70], Si[:70], 'ro', label='Building windows (float processed)
plt.plot(Al[70:146], Si[70:146], 'bs', label='Building windows (non float
plt.plot(Al[146:163], Si[146:163], 'gd', label='Vehicle windows (float pro
plt.plot(Al[163:176], Si[163:176], 'm^', label='Containers')
plt.plot(Al[176:185], Si[176:185], 'yp', label='Tableware')
plt.plot(Al[185:214], Si[185:214], 'c*', label='Headlamps')
plt.legend()
plt.xlabel('Алюминий (Al)')
plt.ylabel('Кремний (Si)')
```

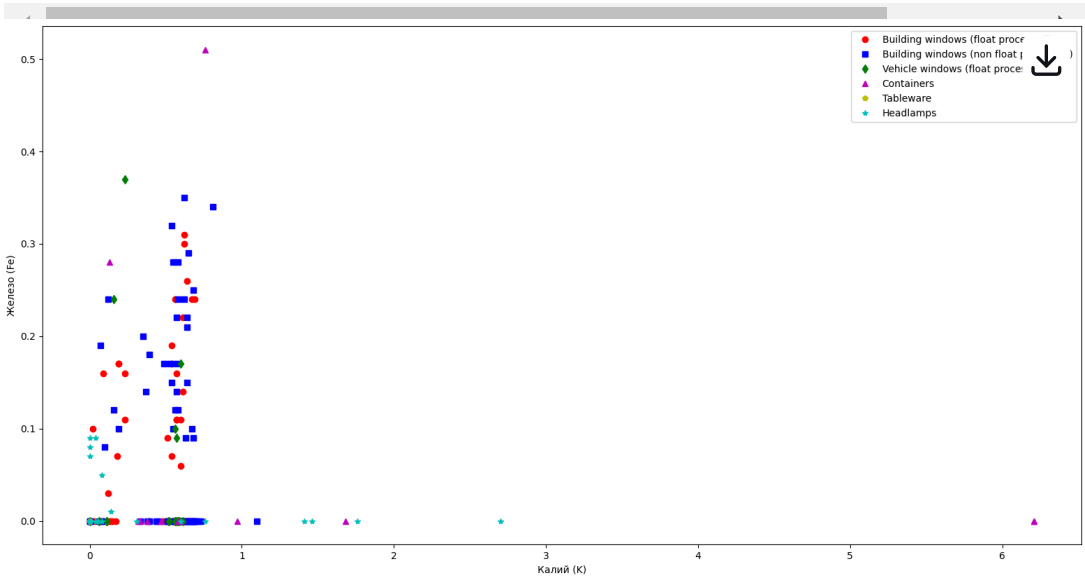


Проекция №3. Кальций и барий

Код для визуализации проекции:

```
plt.figure(3)
plt.plot(Ca[:70], Ba[:70], 'ro', label='Building windows (float processed)
```

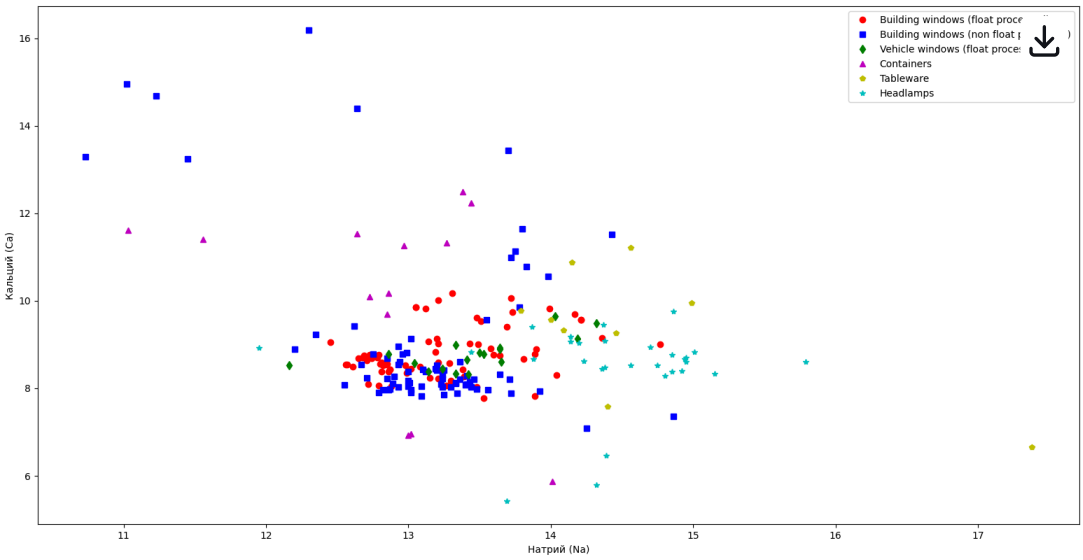
```
plt.plot(Ca[70:146], Ba[70:146], 'bs', label='Building windows (non float
plt.plot(Ca[146:163], Ba[146:163], 'gd', label='Vehicle windows (float pro
plt.plot(Ca[163:176], Ba[163:176], 'm^', label='Containers')
plt.plot(Ca[176:185], Ba[176:185], 'yp', label='Tableware')
plt.plot(Ca[185:214], Ba[185:214], 'c*', label='Headlamps')
plt.legend()
plt.xlabel('Кальций (Ca)')
plt.ylabel('Барий (Ba)')
```



Проекция №4. Калий и железо

Код для визуализации проекции:

```
plt.figure(4)
plt.plot(K[:70], Fe[:70], 'ro', label='Building windows (float processed)'
plt.plot(K[70:146], Fe[70:146], 'bs', label='Building windows (non float p
plt.plot(K[146:163], Fe[146:163], 'gd', label='Vehicle windows (float proc
plt.plot(K[163:176], Fe[163:176], 'm^', label='Containers')
plt.plot(K[176:185], Fe[176:185], 'yp', label='Tableware')
plt.plot(K[185:214], Fe[185:214], 'c*', label='Headlamps')
plt.legend()
plt.xlabel('Калий (K)')
plt.ylabel('Железо (Fe)')
```



Проекция №5. Натрий и кальций

Код для визуализации проекции:

```
plt.figure(5)
plt.plot(Na[:70], Ca[:70], 'ro', label='Building windows (float processed)
plt.plot(Na[70:146], Ca[70:146], 'bs', label='Building windows (non float
plt.plot(Na[146:163], Ca[146:163], 'gd', label='Vehicle windows (float pro
plt.plot(Na[163:176], Ca[163:176], 'm^', label='Containers')
plt.plot(Na[176:185], Ca[176:185], 'yp', label='Tableware')
plt.plot(Na[185:214], Ca[185:214], 'c*', label='Headlamps')
plt.legend()
plt.xlabel('Натрий (Na)')
plt.ylabel('Кальций (Ca)')
```

Контрольные вопросы

1. Для организации рабочего места специалиста Data Science используются такие средства как: язык программирования — **Python**; среды разработки — **Jupyter Notebook, Spyder, PyCharm, VS Code**; визуализация — **Matplotlib**, библиотеки для анализа данных — **Pandas, NumPy, SciPy.**
2. **Scikit-learn** – библиотека для классического машинного обучения (регрессия, классификация, кластеризация). Обладает удобным API и включает множество алгоритмов.

   TensorFlow – мощный инструмент для глубокого обучения и нейросетей, разработанный Google. Позволяет работать как с CPU, так и с GPU.

   PyTorch – альтернатива TensorFlow, разработанная Facebook. Отличается гибкостью и удобством в исследовательских проектах.

3. Простота языка, большое количество библиотек, интеграция с другими языками и системами, поддержка GPU и облачных вычислений, кроссплатформенность.