Лабораторная работа №2. Визуализация данных

Цель лабораторной работы: изучение программных средств для визуализации наборов данных.

Основные задачи:

- установка и настройка matplotlib, seaborn;
- изучение основных типов графиков библиотеки matplotlib;
- изучение основных типов графиков библиотеки seaborn;
- получение навыков анализа данных по визуальным представлениям данных.

Ход выполнения индивидуального задания:

1. Набор данных

Набор данных по сравнению с первой лабораторной работой остался прежним, а именно **Glass Identification**, который входит в репозиторий **UCI Machine Learning Repository**.

2. Первичный анализ данных

2.1 Разбор основных признаков набора данных

Для начала загрузим набор данных при помощи библиотеки **pandas** и выведем его признаки.

	Id	RI	Na	Mg	Al	Si	K	Ca	Ba	Fe	Type
0	1	1.52101	13.64	4.49	1.10	71.78	0.06	8.75	0.0	0.0	
1	2	1.51761	13.89	3.60	1.36	72.73	0.48	7.83	0.0	0.0	1
2	3	1.51618	13.53	3.55	1.54	72.99	0.39	7.78	0.0	0.0	1
3	4	1.51766	13.21	3.69	1.29	72.61	0.57	8.22	0.0	0.0	1
4	5	1.51742	13.27	3.62	1.24	73.08	0.55	8.07	0.0	0.0	1

Код вывода признаков:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

data_path = "https://archive.ics.uci.edu/ml/machine-learning-databases/gla
```

```
columns = [
    "Id", "RI", "Na", "Mg", "Al", "Si", "K", "Ca", "Ba", "Fe", "Type"
]

data = pd.read_csv(data_path, names=columns)
print(data.head())
```

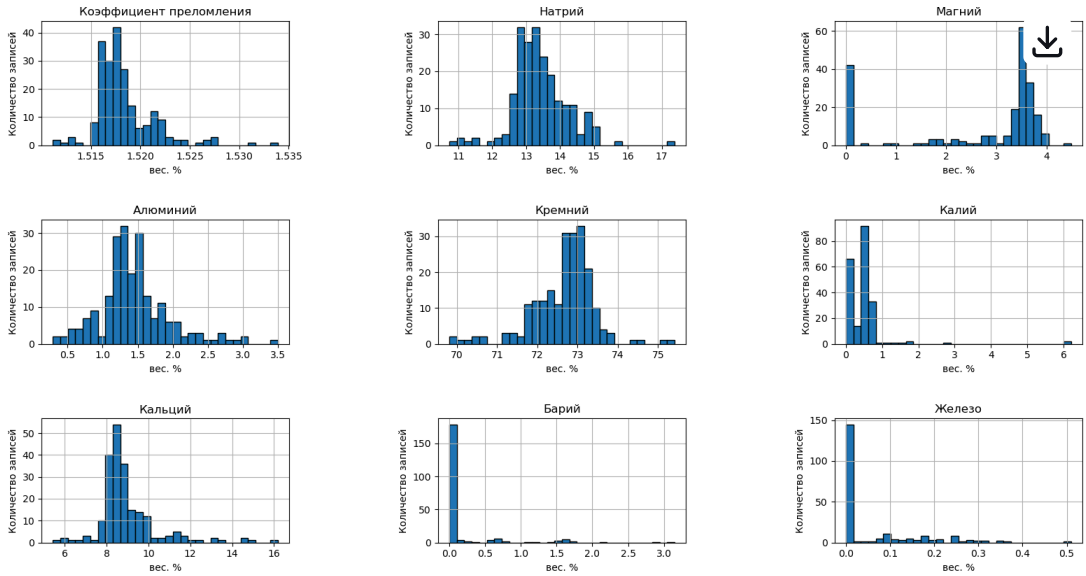
При помощи массива **columns** мы вручную указали названия для колонок в нашей таблице. В противном случае в качестве названия была бы выбрана **первая строка** из набора данных.

Для `info()` из класса **DataFrame**.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 214 entries, 0 to 213
Data columns (total 11 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Id      214 non-null    int64
 1   RI      214 non-null    float64
 2   Na      214 non-null    float64
 3   Mg      214 non-null    float64
 4   Al      214 non-null    float64
 5   Si      214 non-null    float64
 6   K       214 non-null    float64
 7   Ca      214 non-null    float64
 8   Ba      214 non-null    float64
 9   Fe      214 non-null    float64
 10  Type    214 non-null    int64
dtypes: float64(9), int64(2)
memory usage: 18.5 KB
```

Здесь можно увидеть, что набор данных содержит **214 записей** и имеет **11 колонок**, 9 из которых являются типом **float64**, а остальные 2 — **int64**.

2.2 Визуализация количественных признаков



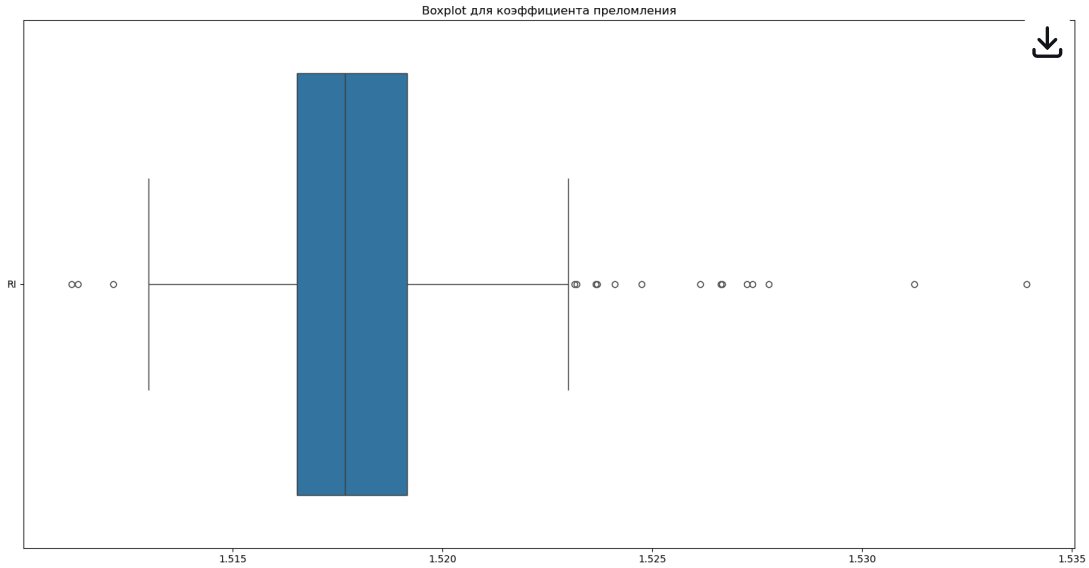
Код для визуализации количественных признаков:

```
plt.figure(figsize=(10, 6))
plt.subplots_adjust(wspace=0.6, hspace=0.6)
plt.subplot(3, 3, 1)
data['RI'].hist(bins=30, edgecolor='black')
plt.title("Коэффициент преломления")
plt.xlabel("вес. %")
plt.ylabel("Количество записей")

plt.subplot(3, 3, 2)
data['Na'].hist(bins=30, edgecolor='black')
plt.title("Натрий")
plt.xlabel("вес. %")
plt.ylabel("Количество записей")

plt.subplot(3, 3, 3)
data['Mg'].hist(bins=30, edgecolor='black')
plt.title("Магний")
plt.xlabel("вес. %")
plt.ylabel("Количество записей")

plt.subplot(3, 3, 4)
data['Al'].hist(bins=30, edgecolor='black')
plt.title("Алюминий")
plt.xlabel("вес. %")
plt.ylabel("Количество записей")

plt.subplot(3, 3, 5)
data['Si'].hist(bins=30, edgecolor='black')
plt.title("Кремний")
plt.xlabel("вес. %")
plt.ylabel("Количество записей")

plt.subplot(3, 3, 6)
data['K'].hist(bins=30, edgecolor='black')
plt.title("Калий")
plt.xlabel("вес. %")
plt.ylabel("Количество записей")

plt.subplot(3, 3, 7)
data['Ca'].hist(bins=30, edgecolor='black')
plt.title("Кальций")
plt.xlabel("вес. %")
plt.ylabel("Количество записей")

plt.subplot(3, 3, 8)
data['Ba'].hist(bins=30, edgecolor='black')
plt.title("Барий")
plt.xlabel("вес. %")
plt.ylabel("Количество записей")

plt.subplot(3, 3, 9)
data['Fe'].hist(bins=30, edgecolor='black')
plt.title("Железо")
```

```
plt.xlabel("вес. %")
plt.ylabel("Количество записей")
```

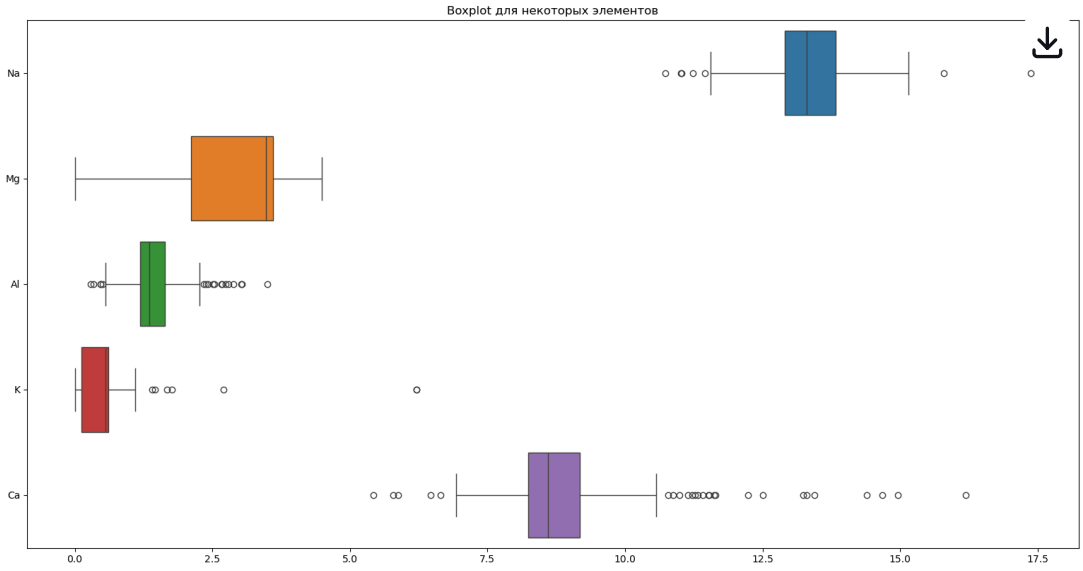
Для построения гистограмм вызывается метод `hist()` класса **DataFtrame**. В качестве параметров метода используются `bins` (Количество интервалов, на которые разбивается диапазон данных), а также `edgecolor` (выделение границ интервалов).

2.3 Визуализация при помощи “Ящика с усами”



Код для boxplot’а коэффициента преломления:

```
plt.figure(figsize=(10, 6))
plt.title("Boxplot для коэффициента преломления")
sns.boxplot(data=data[['RI']], orient='h')
```

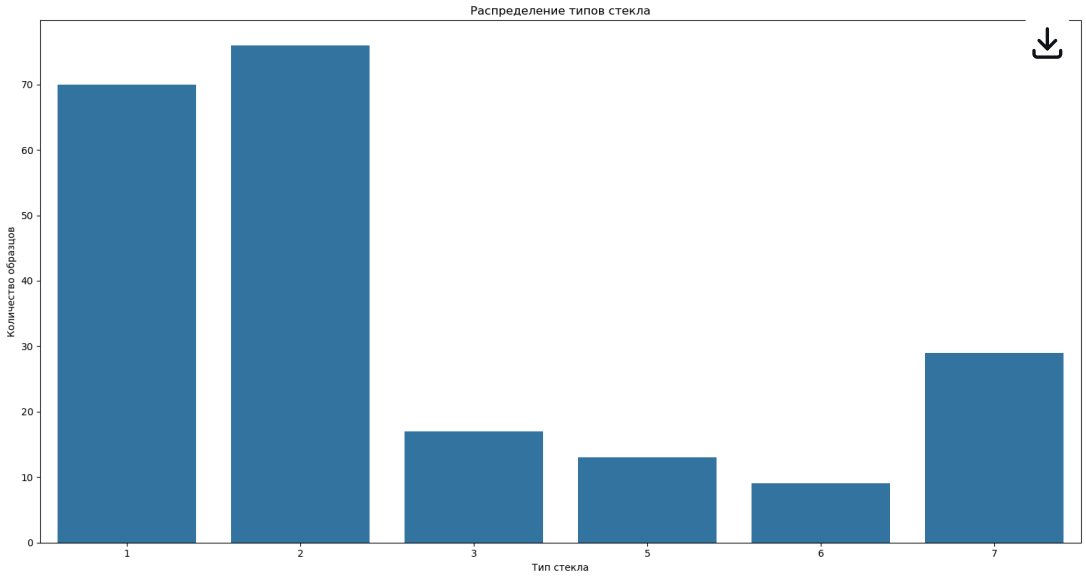


Код для boxplot’а некоторых элементов:

```
plt.figure(figsize=(10, 6))
plt.title("Boxplot для некоторых элементов")
```

```
sns.boxplot(data=data[['Na', 'Mg', 'Al', 'K', 'Ca']], orient='h')
```

2.4 Визуализация категориальных признаков



Код для визуализации категориальных признаков:

```
plt.figure(figsize=(10, 6))
sns.countplot(data=data['Type'])
plt.title("Распределение типов стекла")
plt.xlabel("Тип стекла")
plt.ylabel("Количество образцов")
```

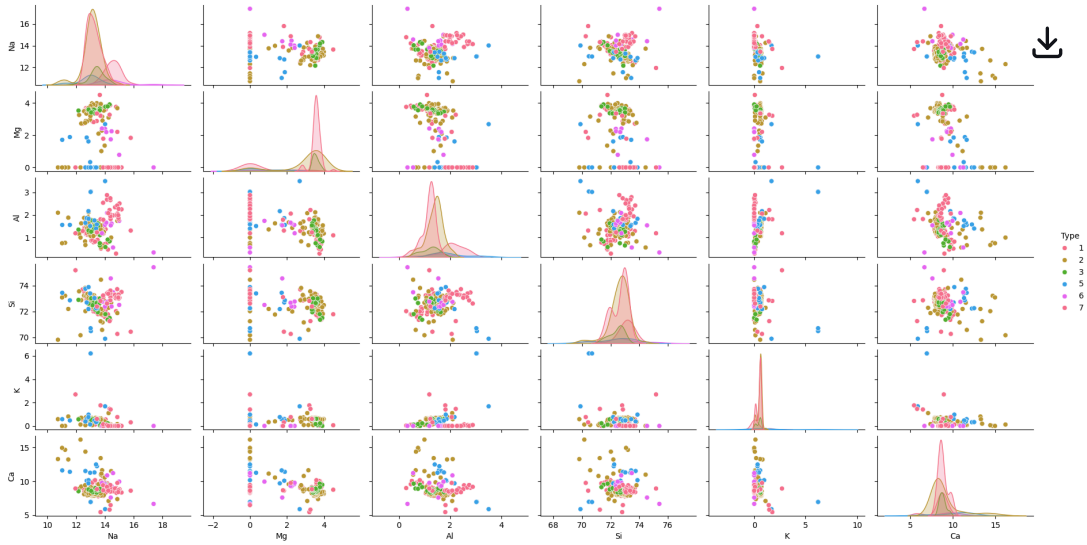
При помощи метода `countplot()` из библиотеки **seaborn** можно вывести количество образцов для каждого типа стекла в наборе данных. Как можно заметить, в данном наборе **наибольшее** количество записей со **2-м типом стекла (Building windows (non float processed))**, а также с **1-м типом (Building windows (float processed))**. **Наименьшее** же количество записей с типом стекла под номером **6 (Tableware)**.

2.5 Визуализация соотношения количественных признаков

Код для визуализации pairplot:

```
sns.pairplot(data=data[['Na', 'Mg', 'Al', 'Si', 'K', 'Ca']])
```

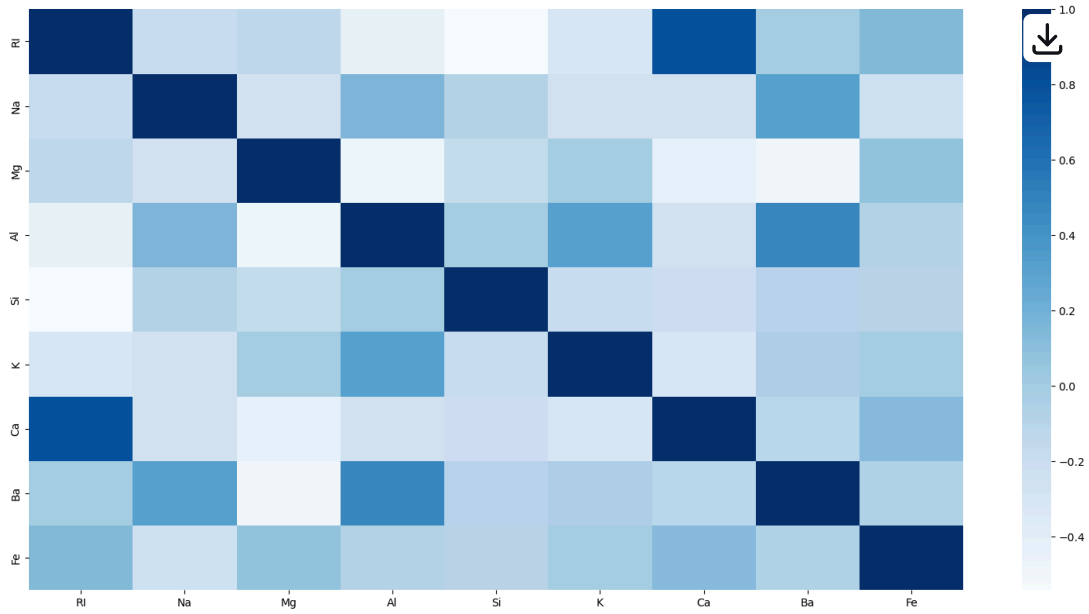
Здесь с использованием метода `pairplot()` из библиотеки **seaborn** представлены диаграммы попарного сравнения признаков для некоторых элементов. На диагональных графиках представлены **гистограммы распределения отдельного признака**, на внедиагональных позициях – **попарные распределения**.



Код для визуализации pairplot с использованием параметра типа:

```
sns.pairplot(data=data[['Na', 'Mg', 'Al', 'Si', 'K', 'Ca', 'Type']], hue='
```

2.6 Визуализация матрицы корреляции



Код для визуализации матрицы корреляции:

```
data.drop(columns=["Id"], inplace=True)

plt.figure(figsize=(10, 6))
sns.heatmap(data.corr(), cmap=plt.cm.Blues)
```

При помощи метода `drop()` мы удаляем ненужный параметр **Id** и уже используя функцию `heatmap()` строим матрицу.

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.

4. Были изучены такие функции, как **hist()**, **boxplot()**, **countplot()**, **pairplot()** и **heatmap()**.
5. Для управления наборами данных чаще всего используется **pandas**.
6. Удаление строк с пропущенными значениями может привести к значительной потере информации, особенно если пропуски встречаются часто. Это может исказить выборку и повлиять на точность модели или дальнейший анализ данных.
7. Не следует
8. Разбиение данных на **обучающую (train)** и **тестовую (test)** выборку необходимо для оценки качества модели. Вариант 20:80 наиболее оптимальный.
9. a) dataset = read_csv(“data.csv”)