Министерство науки и высшего образования РФ.

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Лабораторная работа №2

по дисциплине

«Искусственный интеллект и машинное обучение»

«Визуализация данных»

|  |
| --- |
| Выполнил:  Окунев Николай Александрович  студент 2 курса группы КМБ-с-о-23-1 специальности 10.05.01 «Компьютерная безопасность»  очной формы обучения  (подпись)  Проверил:  Свидницкий Алексей Дмитриевич ассистент ДЦРСиЭ  (подпись) |

Ставрополь, 2025 г

## 1. Цель работы

Основная цель лабораторной работы заключалась в выполнении анализа данных с использованием Python:

* Разбить исходный DataFrame на подмножества в зависимости от логического признака (например, «lowerwhisker»).
* Построить визуализации (scatter plot) для разных групп данных.
* Проанализировать распределение показателей (например, «Social support» и «Perceptions of corruption») с применением различных цветовых схем для выделения групп клиентов (например, ушедших и лояльных).

## 2. Инструменты и библиотеки

Для реализации поставленной задачи использовали следующие инструменты и библиотеки:

* **Python:** основной язык программирования для анализа данных.
* **Pandas:** библиотека для работы с табличными данными (DataFrame), позволяющая удобно фильтровать, группировать и обрабатывать данные.
* **Matplotlib:** библиотека для построения графиков, в данном случае использовалась функция plt.scatter для создания диаграммы рассеяния.
* **Seaborn:** хотя в других фрагментах кода использовалась данная библиотека для визуализации распределения данных (например, построение boxplot), в анализируемом коде основной упор делается на Matplotlib.

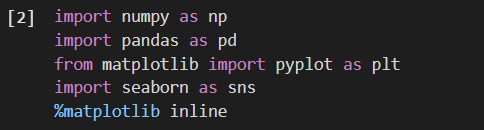


Рисунок 1 - Подключение библиотек

## 3. Описание и анализ кода

### 3.1. Подготовка данных

Обозначаем путь к датасету как data\_path и выводим информацию о типе данных в файле.

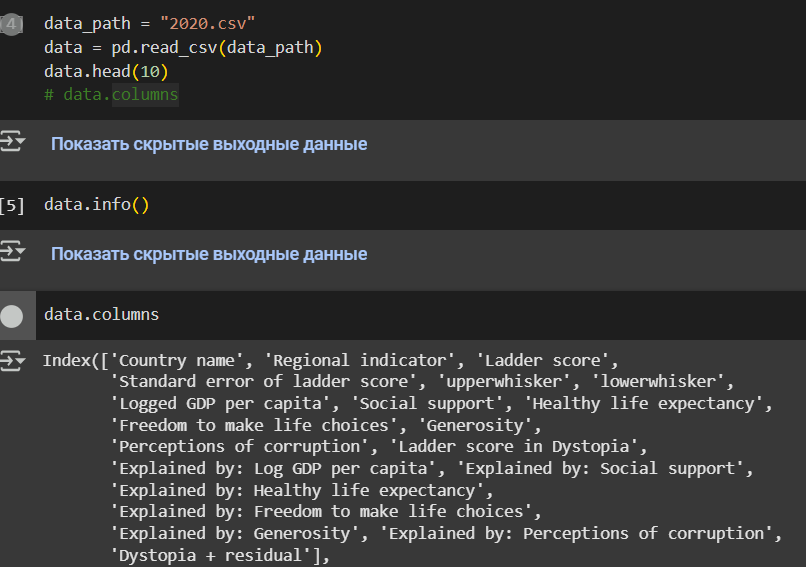


Рисунок 2 - Подготовка датасета

### 3.2. Построение визуализаций

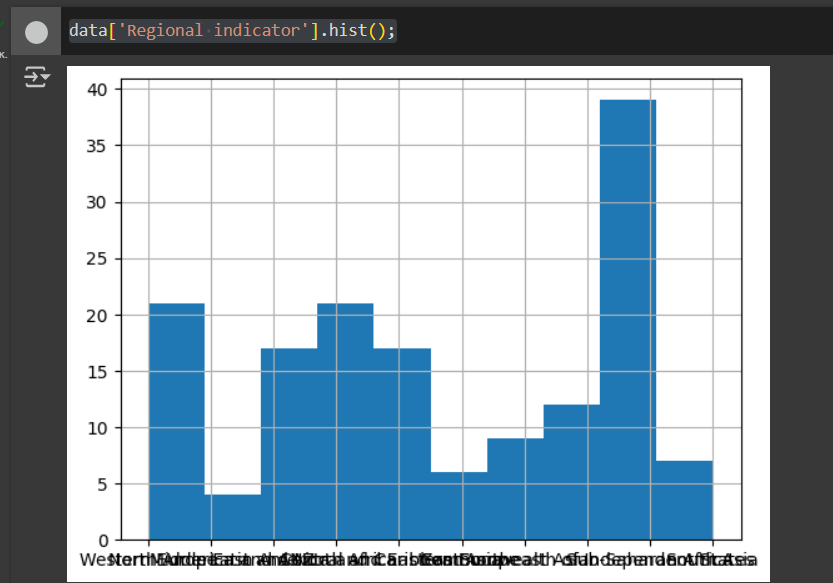


Рисунок 3 - График 1

Данный фрагмент кода строит гистограмму для столбца 'Regional indicator' из DataFrame data. Это позволяет визуально оценить распределение значений этого столбца, например, частоты различных категорий или диапазонов значений. Метод .hist() автоматически создаёт гистограмму с оптимальными настройками (количество интервалов, оси и т.д.) для данных, содержащихся в указанном столбце.

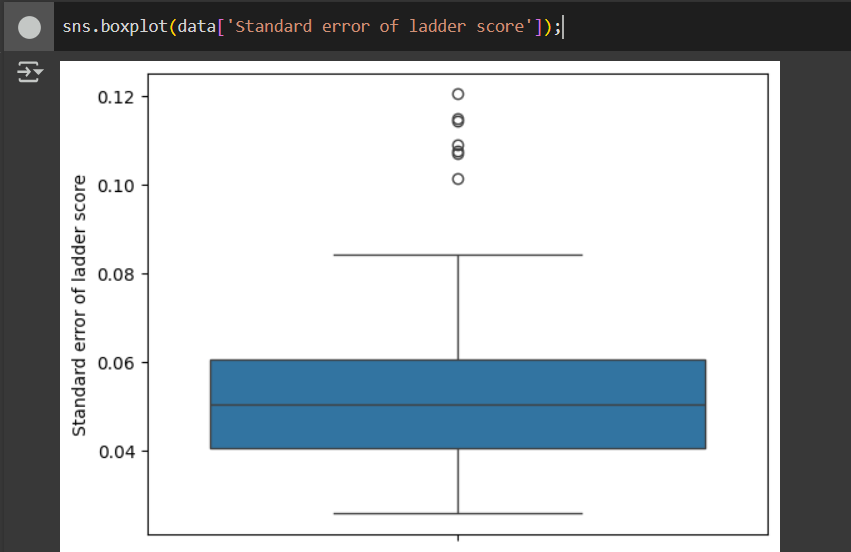


Рисунок 4 - График 2

Данный фрагмент кода использует библиотеку Seaborn для построения графика "ящик с усами" (boxplot) по данным из столбца 'Standard error of ladder score'. Рассмотрим подробнее, что происходит:

Выбор столбца:  
Из DataFrame data выбирается столбец с названием 'Standard error of ladder score'. Обычно этот столбец содержит числовые данные, характеризующие стандартную ошибку оценочного значения (например, индекса счастья или другого показателя).

* Построение boxplot:  
  Функция sns.boxplot создает график "ящик с усами", который наглядно демонстрирует:
* Медиану: центральная линия внутри ящика.
* Первый и третий квартиль: границы ящика, показывающие, где располагается 50% значений.
* "Усы": линии, выходящие из ящика, которые показывают диапазон данных, не являющихся выбросами.
* Выбросы: отдельные точки, если они присутствуют, указывающие на данные, значительно отличающиеся от остальных.

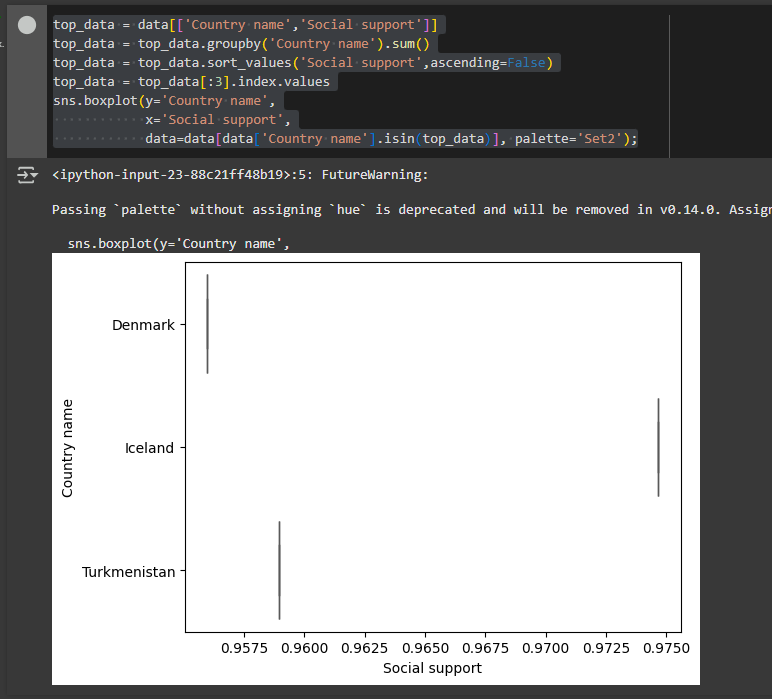


Рисунок 5 - График 3

Данный фрагмент кода выполняет следующие шаги:

Выбор и группировка данных:

* Из исходного DataFrame выбираются два столбца: 'Country name' и 'Social support'.
* Группировка выполняется по странам ('Country name'), при этом суммируются значения по столбцу 'Social support' для каждой страны.

Сортировка и выбор топ-3:

* Группированные данные сортируются по суммарным значениям 'Social support' в порядке убывания.
* Из отсортированного результата выбираются первые три страны (топ-3), и их названия сохраняются в переменной top\_data.

Построение boxplot:

* Функция sns.boxplot строит график "ящик с усами", где:
* По оси Y располагаются страны ('Country name').
* По оси X – значения 'Social support'.
* Для построения графика используется исходный DataFrame, отфильтрованный по странам, входящим в топ-3 (то есть, те, у которых суммарное значение 'Social support' наибольшее).
* Параметр palette='Set2' задаёт цветовую схему для графика.

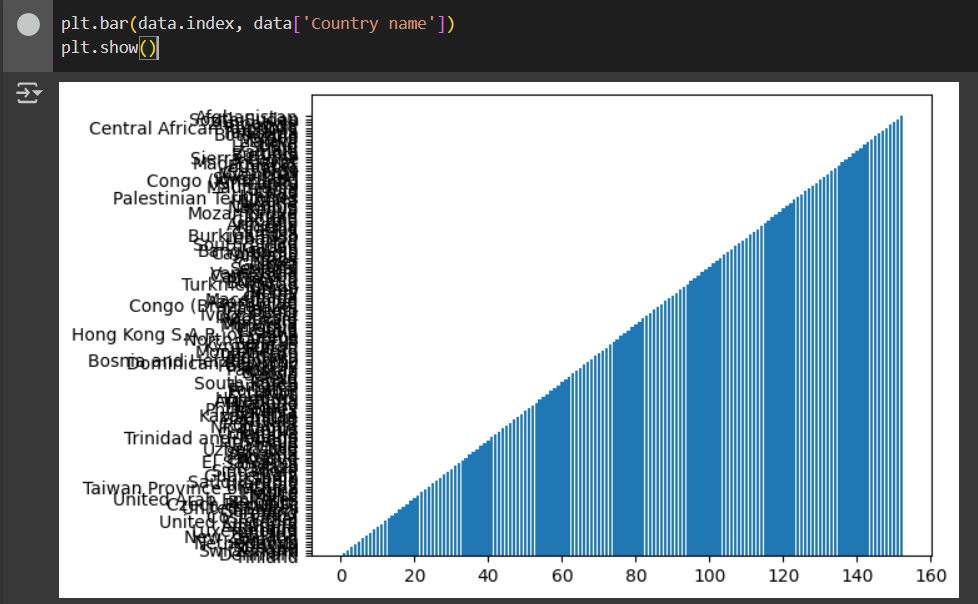


Рисунок 6 - График 4

Код выполняет построение столбчатой диаграммы с использованием библиотеки Matplotlib:

* plt.bar(data.index, data['Country name']):
* По оси X используются индексы DataFrame (data.index).
* По оси Y берутся значения из столбца 'Country name'.  
  Однако, поскольку 'Country name' представляет собой строковые значения (имена стран), их использование в качестве значений для оси Y может привести к тому, что график не будет отображаться корректно, так как столбчатая диаграмма ожидает числовые данные для высоты столбцов.

plt.show():  
Отображает созданный график.

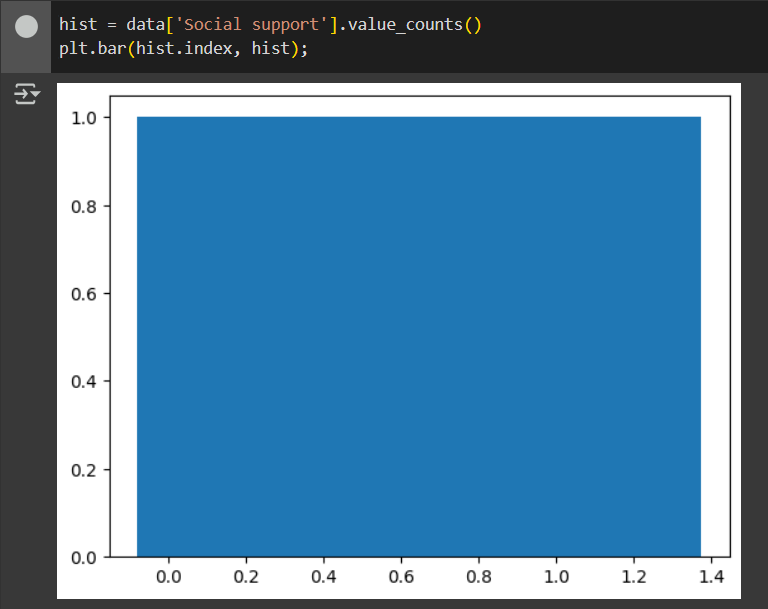


Рисунок 7 - График 5

Функция value\_counts() подсчитывает, сколько раз встречается каждое уникальное значение в столбце 'Social support'. Результатом является объект Series, где индексами являются уникальные значения, а значениями – частоты их появления.

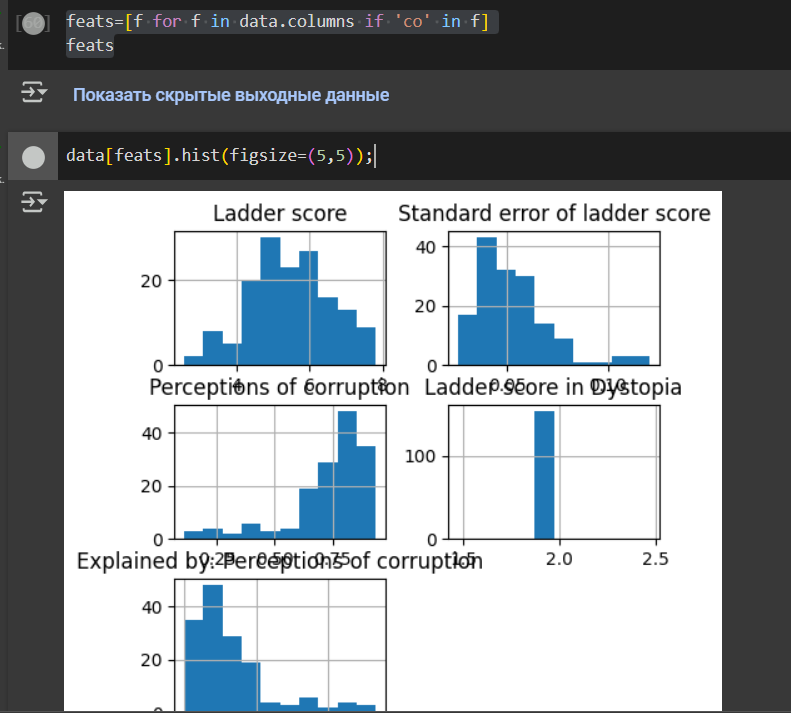


Рисунок 8 - График 5

Ищет в колонках совпадения со строкой “co” и строит их диаграммы

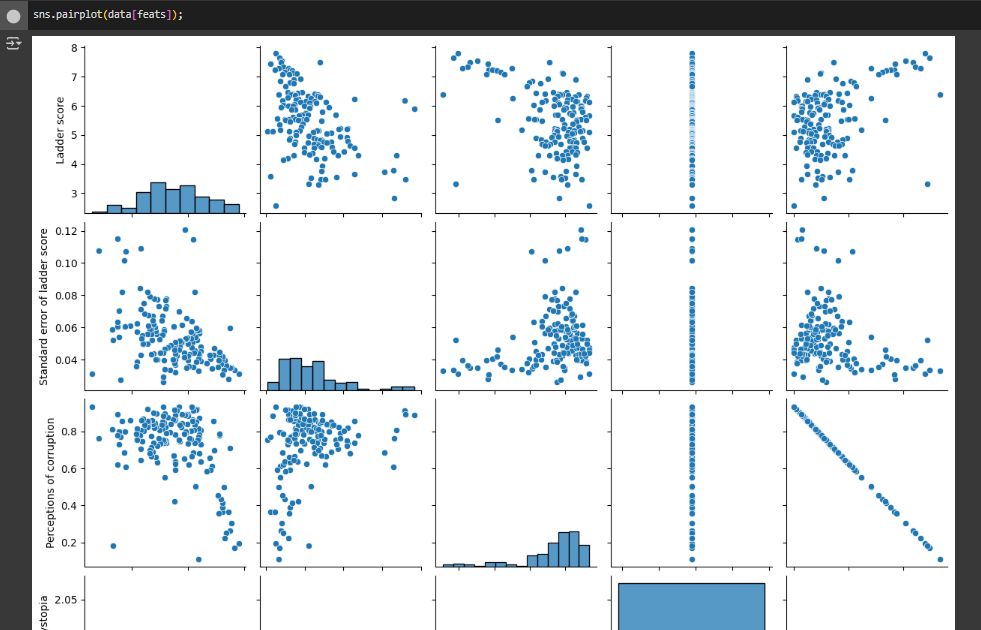


Рисунок 9 - График 6

sns.pairplot() из библиотеки Seaborn строит диаграммы рассеяния (scatter plot) для каждой пары числовых столбцов из data[feats].

На диагонали выводятся гистограммы (если данные числовые).

Графики позволяют визуально выявить взаимосвязи между переменными.



Рисунок 10 - График 7

Этот код строит тепловую карту (heatmap) корреляций между числовыми столбцами в DataFrame data с использованием библиотеки Seaborn.

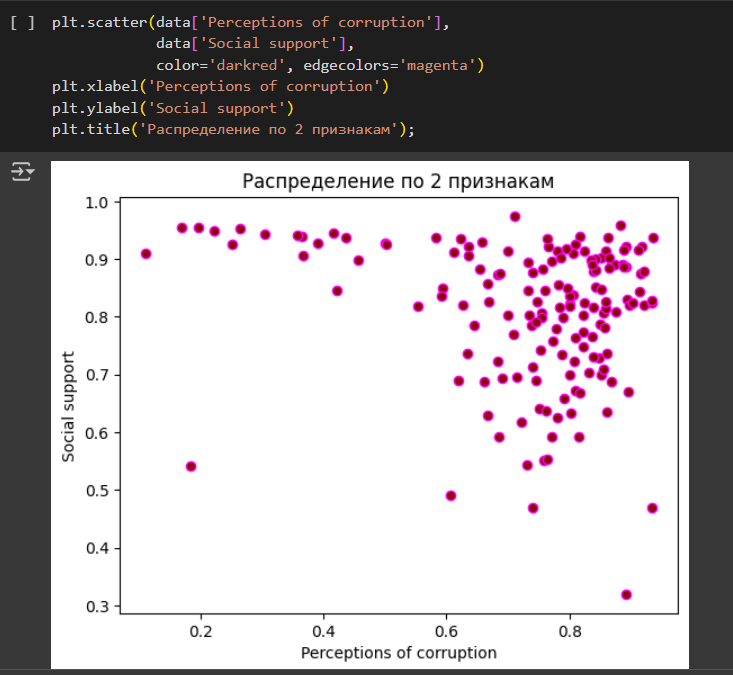


Рисунок 11 - График 8

цвет точки зависит от целевого значения признака

data\_uncorr = data.drop(feats, axis=1)

data\_uncorr.columns

Рисунок 11 - Удаление столбцов

Этот код выполняет удаление из DataFrame data всех столбцов, содержащих подстроку 'co' в названии, и выводит оставшиеся столбцы.

3.3 Контрольные вопросы

1)Какие инструментальные средства используются для организации рабочего места специалиста Data Science?

Языки программирования:

* Python: Самый популярный язык для Data Science. У него много библиотек для анализа данных, машинного обучения и визуализации.
* R: Язык для статистических вычислений и графики. Тоже очень популярен в Data Science.
* SQL: Для работы с базами данных и извлечения данных.
* Scala: Для работы с большими данными и Spark.
* IDE (интегрированные среды разработки):
* Jupyter Notebook: Очень удобная среда для интерактивной работы с кодом, визуализацией и документацией.
* Visual Studio Code (VS Code): Мощный редактор кода с поддержкой Python, R и других языков. У него много расширений для Data Science.
* PyCharm: IDE от JetBrains специально для Python.
* RStudio: IDE для R.

Библиотеки и пакеты:

* NumPy: Для работы с массивами и матрицами.
* Pandas: Для анализа и манипуляции данными.
* Scikit-learn: Для машинного обучения.
* Matplotlib и Seaborn: Для визуализации данных.
* TensorFlow и PyTorch: Для глубокого обучения.
* Statsmodels: Для статистического моделирования.

Инструменты для работы с большими данными:

* Apache Spark: Для обработки больших данных.
* Hadoop: Для хранения и обработки больших данных.
* Dask: Для параллельных вычислений в Python.

Системы контроля версий:

* Git: Для управления версиями кода и совместной работы.
* GitHub, GitLab, Bitbucket: Платформы для хостинга Git-репозиториев.

Инструменты для развертывания моделей:

* Docker: Для создания контейнеров с моделями.
* Kubernetes: Для управления контейнерами.
* Cloud platforms (AWS, Azure, GCP): Для развертывания моделей в облаке.

Инструменты для визуализации данных:

* Tableau: Для создания интерактивных дашбордов.
* Power BI: Еще один инструмент для дашбордов от Microsoft.

Другие инструменты:

* Anaconda: Платформа для управления пакетами и средами в Python и R.
* Slack, Microsoft Teams: Для общения и совместной работы.
* Trello, Jira: Для управления проектами и задачами.

Это только основные инструменты. Data Scientist может использовать и другие инструменты в зависимости от проекта и задач. Главное чтобы инструменты помогали ему анализировать данные, строить модели и получать полезные результаты.

2)Какие библиотеки Python используются для работы в области машинного обучения? Дайте краткую характеристику каждой библиотеке.

Scikit-learn:

Описание: Самая популярная библиотека для машинного обучения. В ней есть много алгоритмов для классификации, регрессии, кластеризации, уменьшения размерности и много чего еще. Еще в ней есть инструменты для оценки моделей, выбора параметров и предварительной обработки данных.

Особенности: Простая в использовании, хорошо документирована, много примеров.

TensorFlow:

Описание: Библиотека от Google для глубокого обучения. В основном используется для создания и обучения нейронных сетей.

Особенности: Очень мощная, гибкая, можно использовать для решения разных задач.

Keras:

Описание: Высокоуровневый API для работы с TensorFlow (и другими бэкендами). Упрощает создание и обучение нейронных сетей.

Особенности: Простая в использовании, позволяет быстро создавать сложные модели.

PyTorch:

Описание: Еще одна популярная библиотека для глубокого обучения. В основном используется в научных исследованиях.

Особенности: Гибкая, динамическая, хорошо подходит для экспериментов.

XGBoost:

Описание: Библиотека для градиентного бустинга. Очень хорошо работает на разных задачах.

Особенности: Быстрая, точная, много настроек.

LightGBM:

Описание: Еще одна библиотека для градиентного бустинга от Microsoft.

Особенности: Быстрая, использует меньше памяти, чем XGBoost.

CatBoost:

Описание: Библиотека для градиентного бустинга от Yandex. Хорошо работает с категориальными данными.

Особенности: Не требует предварительной обработки категориальных данных, устойчива к переобучению.

Statsmodels:

Описание: Библиотека для статистического моделирования. В ней есть много моделей для регрессии, анализа временных рядов и других задач.

Особенности: Подробные результаты, позволяет проводить статистический анализ.

NLTK:

Описание: Библиотека для обработки естественного языка. В ней есть инструменты для токенизации, стемминга, лемматизации, POS-теггинга и других задач.

Особенности: Много инструментов для работы с текстом.

SpaCy:

Описание: Еще одна библиотека для обработки естественного языка. Она более быстрая и эффективная, чем NLTK.

Особенности: Быстрая, хорошо подходит для больших объемов текста.

3)Почему при реализации систем машинного обучения широкое распространение получили библиотеки Python?

* Простота и читаемость: Python легко учить и читать. У него понятный синтаксис и много примеров. Это делает его доступным для людей с разным опытом.
* Большое количество библиотек: В Python есть много библиотек для машинного обучения, анализа данных, визуализации и других задач. Это позволяет Data Science решать разные задачи, не изобретая велосипед.
* Сообщество: У Python большое и активное сообщество. Это значит что всегда можно найти помощь, задать вопрос или поделиться своим опытом.
* Кроссплатформенность: Python работает на разных операционных системах (Windows, macOS, Linux). Это удобно для разработки и развертывания моделей.
* Гибкость: Python можно использовать для разных задач, от простых скриптов до сложных систем машинного обучения.
* Интеграция с другими языками: Python можно интегрировать с другими языками, например C++ и Java. Это позволяет использовать существующий код и библиотеки.
* Открытый исходный код: Python - это язык с открытым исходным кодом. Это значит что его можно использовать бесплатно и изменять под свои нужды.
* Поддержка со стороны крупных компаний: Google, Facebook и другие крупные компании используют Python для машинного обучения. Они разрабатывают и поддерживают библиотеки, что делает Python еще более популярным.

5)Библиотека для управления наборами данных: Pandas.

6)Нежелательная стратегия при обработке пропусков: б) удаление строк, содержащих пропуски в данных (потому что можно потерять много важной информации).

7)OneHotEncoder для категориальной переменной: Да, нужно применять OneHotEncoder. Потому что модели машинного обучения обычно работают с числовыми данными, а OneHotEncoder преобразует категориальные признаки в числовые.

8)Разбиение на обучающую и тестовую выборку:

Обучающая выборка нужна для обучения модели.

Тестовая выборка нужна для оценки качества обученной модели на новых данных.

Оптимальное соотношение: 20:80 или 25:75 (тестовая:обучающая). Это позволяет модели хорошо обучиться и при этом получить адекватную оценку качества.

10)Код для загрузки данных из CSV-файла: а) dataset = read\_csv(“data.csv”).

## 4. Вывод

Работа демонстрирует навыки работы с библиотекой Pandas для фильтрации и разбиения данных, а также использование Matplotlib для построения визуализаций.