Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |  |
| --- | --- |
| Институт | математики и компьютерных наук |
| Кафедра | компьютерной безопасности |

ОТЧЕТ

ЛАБОРАТОРНАЯ РАБОТА 5.

РАЗРАБОТКА ЕДИНОГО ПОДХОДА К ПРЕДВАРИТЕЛЬНОЙ ОБРАБОТКИ ДАННЫХ

Выполнил: Окунев Николай Александрович,

студент 2 курса

группы КМБ-с-о-23-1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г.

Проверено с оценкой:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г.

Ставрополь, 2025

1. **Цели и задачи**

Цель лабораторной работы: изучение теоретических принципов и инструментальных редств для построения пайплайна для предварительной обработки данных. Основные задачи: – предварительная обработка данных; – изучение библиотек для предварительной обработки данных; – масштабирование признаков; – представление категориальных данных; – построение пайплайна для предварительной обработки данных.

1. **Теоретическое обоснование**

В предыдущих работах уже были представлены алгоритмы, позволяющие представить последовательность решения задач методами машинного обучения. Но для различных моделей очень часто приходится повторять одни и те же действия. Это делает возможным выработку некторого унифицированного подхода к последовательности действий, которые реализует исследователь. В данной работе рассмотрим унифицированную последовательность действий, которую обычно выполняет исследователь по предварительной обработке данных. Для решения задач машинного обучения часто приходится повторять различные блоки кода, которые являются единоообразными для разных задач, принадлежащих одному классу (регрессия, классификация, кластеризация и т.д.). Данное обстоятельство приводит к повторяющемуся шаблонному коду. Такой код называется boilerplate-код или просто boilerplate. С другой стороны, единообразная последовательность действий, которую выполняет 53 разработчик при решении задач машинного обучения часто называется пайплайном (machine learning pipeline). Рассмотрим простейший пайплан для решения задачи регрессии. Для решения задачи регрессии небходимо реализовать (в общем случае) следующие стадии: 1. Загрузка набора данных. 2. Заполнение пропусков данных в соответствии с выбранной стратегией. 3. Масштабирование признаков. 4. Обработка категоривальных признаков. 5. Разделение на тестовую и тренировочную выборку. 6. Обучение модели. 7. Интерпретация и визуализация результатов. Заполнение пропусков в данных missingvalues = SimpleImputer(missing\_values = np.nan, strategy = 'mean', verbose = 0) missingvalues = missingvalues.fit(X[:, 1:3]) X[:, 1:3]=missingvalues.transform(X[:, 1:3]) Масштабирование признаков (standartisation и normalization) from sklearn.preprocessing import StandardScaler Разделение на тестовую и обучающую выборки: from sklearn.model\_selection import train\_test\_split from sklearn.cross\_validation import train\_test\_split Перед выполнением лабораторной работы необходимо ознакомиться с базовыми принципами языка Python, используя следующие источники: [1-5]. 54

1. **Методика и порядок выполнения работы**

Для тестирования универсального пайплайна будет использоваться модель линейоной регрессии (LinearRegression из библиотеки sklearn). 3.1 Учебная задача Устовие. Построить пайплайн, реализующий первичную обработку данных. Решение. Для решения задачи необходимо написать скрипт на яыке Python (рисунок 5.1). 55

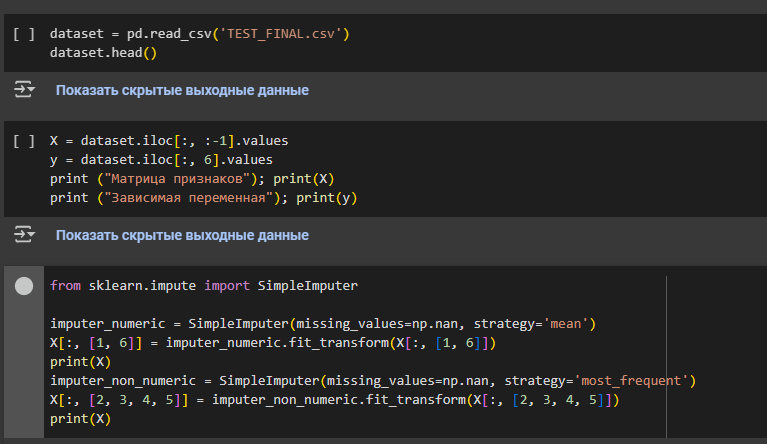


Рисунок 5.1 – Код Python, отражающий общий пайплайн для предварительной обработки данных

На рисунке 5.1 показан устаревший метод работы с трансформатором OneHotEncoder. На рисунке 5.2 показан метод. Который используется в настоящее время. Данный подход предполагает применение класса ColumnTransformer, который осуществляет преобразование значений столбцов с использованием переданных ему классов-трансформеров. В 56 качестве классов-трансформеров можно использовать классы FeatureHasher, MinMaxScaler, SimpleImputer, OneHotEncoder, Normalizer и прочие.

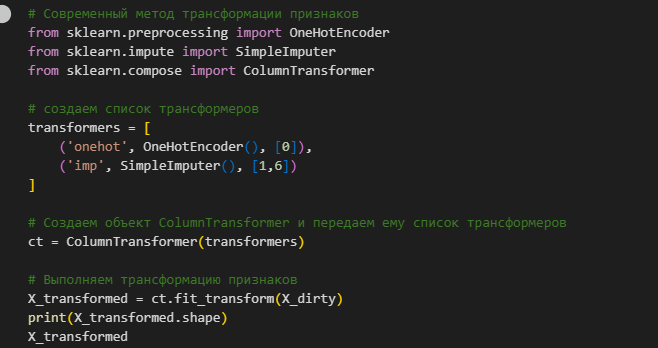


Рисунок 5.2 – Код Python, отражающий применение ColumnTransformer

Следует обратить внимание, что большинство методов трансформации приводит к получению результата в виде массива. Для конвертации массива в DataFrame можно воспользоваться следующим подходом:

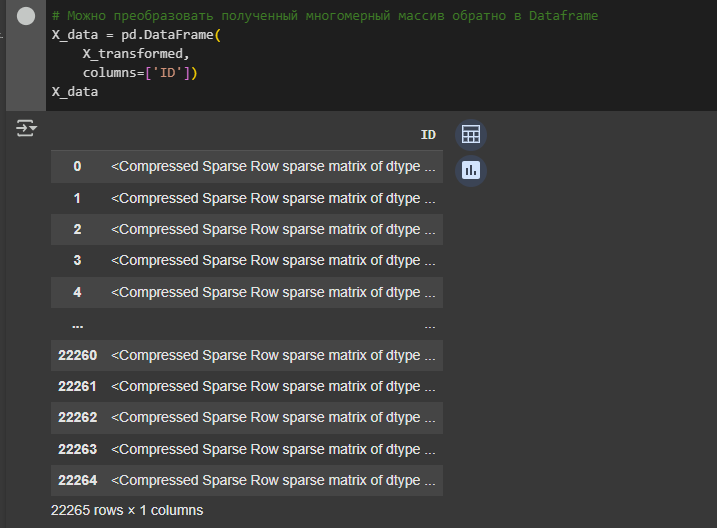


Рисунок 5.3 – Преобразование массива в DataFrame 3.2

1. **Контрольные вопросы**

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”).

**Вывод:** изучили теоретические принципы и инструментальные средств для построения пайплайна и предварительной обработки данных.