Ходырев Роман Владиславович

ИУ5-65Б

18 вариант

pandas — для работы с таблицами (DataFrame),

StandardScaler и LabelEncoder из sklearn — для масштабирования и кодирования признаков,

seaborn и matplotlib.pyplot — для визуализации данных, numpy — для работы с массивами и математикой.

import pandas as pd

from sklearn.preprocessing import StandardScaler, LabelEncoder import seaborn as sns

import matplotlib.pyplot as plt import numpy as np

Загружается датасет toy\_dataset.csv с указанием, что в нём используется разделитель ;. Результат сохраняется в переменной df (DataFrame).

file\_path = "toy\_dataset.csv"

df = pd.read\_csv(file\_path, delimiter=';')

print("Общая информация о данных:") print(df.info())

display(df.head())

Общая информация о данных:

<class 'pandas.core.frame.DataFrame'> RangeIndex: 150000 entries, 0 to 149999 Data columns (total 6 columns):

# Column Non-Null Count Dtype

1. Number 150000 non-null int64
2. City 150000 non-null object
3. Gender 150000 non-null object

3 Age 150000 non-null int64

4 Income 150000 non-null float64

5 Illness 150000 non-null object dtypes: float64(1), int64(2), object(3) memory usage: 6.9+ MB

None

|  |  |
| --- | --- |
| Number | 0 |
| City | 0 |
| Gender | 0 |
| Age | 0 |
| Income | 0 |
| Illness 0  dtype: int64 | |

Если в данных изначально нет пропусков, создаются искусственные пропуски в 1% строк столбца Income. Это нужно для демонстрации обработки отсутствующих значений.

Number City Gender

1. 1 Dallas Male
2. 2 Dallas Male
3. 3 Dallas Male
4. 4 Dallas Male
5. 5 Dallas Male

Age 41

54

42

40

46

Income Illness

40367.0

45084.0

52483.0

40941.0

50289.0

No No No No No

print("\nПроверка на пропуски:") print(df.isnull().sum())

Проверка на пропуски:

if df.isnull().sum().sum() == 0:

print("\nПропусков не найдено. Добавим искусственные пропуски.") nan\_indices = df.sample(frac=0.01, random\_state=42).index df.loc[nan\_indices, 'Income'] = np.nan

Пропусков не найдено. Добавим искусственные пропуски.

После добавления пропусков ещё раз выводится количество пропусков по столбцам — теперь они должны быть в столбце Income.

|  |  |
| --- | --- |
| Number | 0 |
| City | 0 |
| Gender | 0 |
| Age | 0 |
| Income | 1500 |
| Illness | 0 |

Пропущенные значения в столбце Income заполняются средним значением по этому столбцу (mean).

print("\nПосле добавления пропусков:") print(df.isnull().sum())

После добавления пропусков:

dtype: int64

df['Income'] = df['Income'].fillna(df['Income'].mean())

Происходит стандартизация (нормализация) столбца Income: создаётся новый столбец Income\_scaled, где значения имеют среднее 0 и стандартное отклонение 1. Это важно для модели машинного обучения.

scaler = StandardScaler()

df['Income\_scaled'] = scaler.fit\_transform(df[['Income']])

Категориальный признак Gender преобразуется в числовой формат с помощью LabelEncoder, создаётся новый столбец Gender\_LabelEncoded.

label\_encoder = LabelEncoder()

df['Gender\_LabelEncoded'] = label\_encoder.fit\_transform(df['Gender'])

Также применяется one-hot кодирование к столбцу Gender, создаются новые столбцы (например, Gender\_Female, Gender\_Male) с бинарными значениями 0 и 1. Результат сохраняется в df\_onehot.

df\_onehot = pd.get\_dummies(df, columns=['Gender'], prefix='Gender')

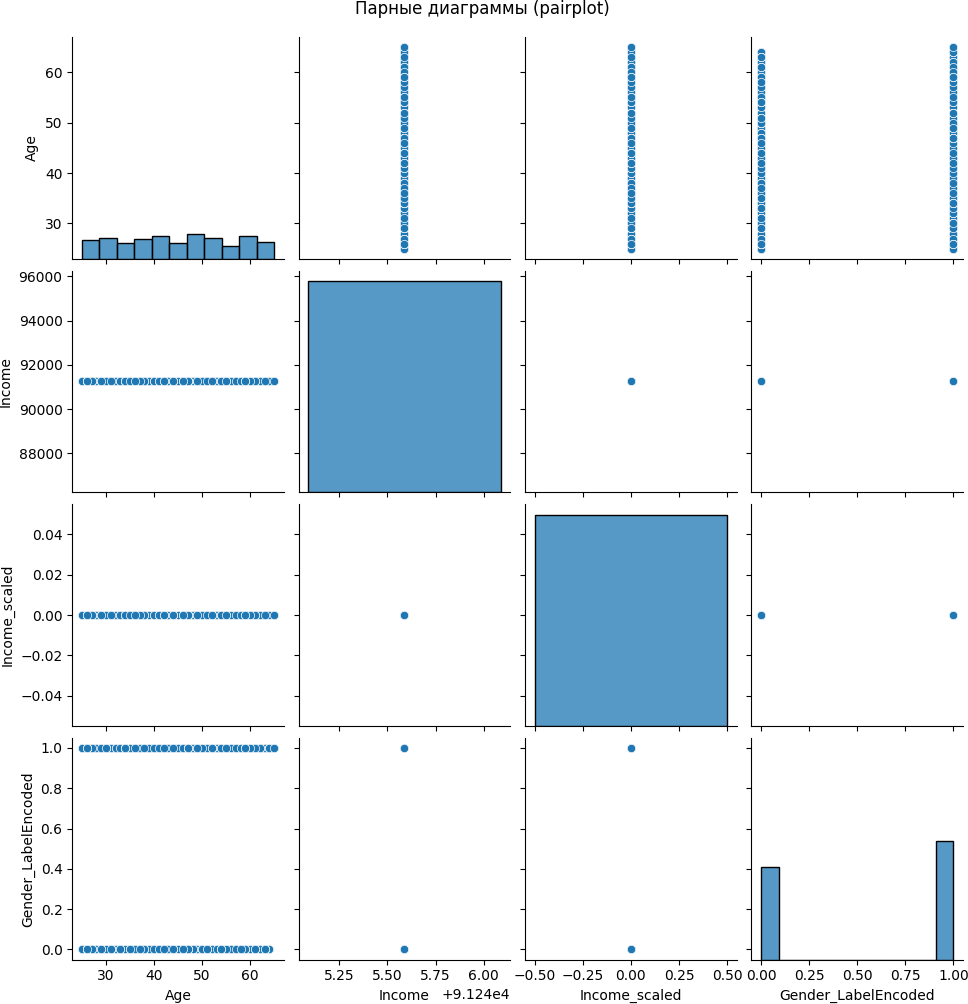
Для визуализации случайным образом выбирается 1000 строк из исходного датафрейма. Это позволяет ускорить визуализацию и сделать графики более читаемыми.

sample\_df = df.sample(1000, random\_state=42)

Создаётся парная диаграмма (pairplot) для переменных Age, Income, Income\_scaled и Gender\_LabelEncoded. Это помогает визуально оценить распределение и взаимосвязи между переменными.

sns.pairplot(sample\_df[['Age', 'Income', 'Income\_scaled', 'Gender\_LabelEncoded']])

plt.suptitle("Парные диаграммы (pairplot)", y=1.02) plt.show()



В процессе выполнения задания были использованы следующие методы:

1. Предобработка данных (data preprocessing):
   * Проверка на пропуски с помощью df.isnull().sum().
   * Искусственное добавление пропусков, чтобы отработать процедуру их обработки.
   * Заполнение пропущенных значений средним (mean)

значением. Это простой и часто используемый способ,

особенно если распределение признака близко к нормальному.

1. Масштабирование признаков (feature scaling):
   * Использован StandardScaler, чтобы стандартизировать признак Income (преобразовать его к нулевому среднему и единичному стандартному отклонению). Это важно для моделей, чувствительных к масштабу данных (например, линейная регрессия, метод ближайших соседей и др.).
2. Кодирование категориальных признаков:
   * Применён LabelEncoder для преобразования категориального признака Gender в числовой формат. Это удобно, когда категорий всего две.
   * Также использовано one-hot кодирование (pd.get\_dummies), которое даёт бинарные столбцы на каждую категорию. Это универсальный метод кодирования категориальных признаков, который позволяет избежать ложной порядковости, присущей LabelEncoder.
3. Визуализация данных:
   * Использован pairplot из библиотеки seaborn для анализа взаимосвязей между переменными. Это позволяет наглядно увидеть зависимости, кластеры и выбросы в данных.