Московский государственный технический университет им. Н.Э. Баумана

Отчёт по рубежному контролю №1 по курсу «Технологии машинного обучения».

«Технологии разведочного анализа и обработки данных».

Выполнил:
Анцифров Н. С.
студент группы ИУ5-61Б

Проверил:
Гапанюк Ю.Е.

Подпись и дата:

Подпись и дата:

Москва, 2022 г.

1. Задание рубежного контроля и входные данные

1.1. Вариант 4 - задача 1 - набор данных 4.

1.1.1. Задача 1.

Для заданного набора данных провести корреляционный анализ. В случае наличия пропусков в данных удалить строки или колонки, содержащие пропуски. Сделать выводы о возможности построения моделей машинного обучения и о возможном вкладе признаков в модель.

1.1.2. Дополнительное задание.

Для пары произвольных колонок данных построить график “Диаграмма рассеяния”.

1.1.3. Набор данных 4.

Доступен по адресу: https://www.kaggle.com/carlolepelaars/toy-dataset

2. Ячейки Jupyter-ноутбука

2.1. Текстовое описание датасета

В качестве набора данных используется датасет с вымышленными данными. Он имеет следующие атрибуты:

- Number - порядковый номер - индекс для каждой строки
- City - город - город проживания человека
- Gender - пол - пол человека
- Age - возраст - сколько человеку лет
- Income - доход - годовой доход человека
- Illness - болезнь - болеет ли человек

2.2. Импорт библиотек и загрузка данных

Импортируем необходимые библиотеки:

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Загрузим датасет:

[2]:
```
data = pd.read_csv('toy_dataset.csv')
```

Выведем первые 5 строк датасета:

[3]:
```
data.head()
```

[3]:

	Number	City	Gender	Age	Income	Illness
0	1	Dallas	Male	41	40367.0	No
1	2	Dallas	Male	54	45084.0	No
2	3	Dallas	Male	42	52483.0	No
3	4	Dallas	Male	40	40941.0	No
4	5	Dallas	Male	46	50289.0	No

Определим размер датасета:

[4]:
```
data.shape
```

[4]:
```
(150000, 6)
```

Определим типы столбцов:

[5]:
```
data.dtypes
```

[5]:
```
Number       int64
City        object
Gender      object
Age          int64
Income     float64
Illness     object
dtype: object
```

Часть столбцов имеют тип “Object”. Для корреляционного анализа требуется преобразование этих столбцов в числовые типы данных.

Столбец “Number” не нужен для корреляции, поэтому удалим его:

[6]:
```
data = data.drop(columns=['Number'], axis=1)
```

[7]:
```
data.head()
```

[7]:

	City	Gender	Age	Income	Illness
0	Dallas	Male	41	40367.0	No
1	Dallas	Male	54	45084.0	No
2	Dallas	Male	42	52483.0	No
3	Dallas	Male	40	40941.0	No
4	Dallas	Male	46	50289.0	No

2.3. Преобразование типов данных

Проверим уникальные значения для столбца “City”:

[8]:
```
data['City'].unique()
```

[8]:
```
array(['Dallas', 'New York City', 'Los Angeles', 'Mountain View',
       'Boston', 'Washington D.C.', 'San Diego', 'Austin'], dtype=object)
```

В качестве значений в столбце “City” могут быть следующие города: “Dallas”, “New York City”, “Los Angeles”, “Mountain View”, “Boston”, “Washington D.C.”, “San Diego” and “Austin”. Таких значений 8.

Проверим уникальные значения для столбца “Gender”:

[9]:
```
data['Gender'].unique()
```

[9]:
```
array(['Male', 'Female'], dtype=object)
```

В столбце “Gender” 2 варианта - “Male” или “Female”.

Проверим уникальные значения для столбца “Illness”:

[10]:
```
data['Illness'].unique()
```

[10]:
```
array(['No', 'Yes'], dtype=object)
```

В столбце “Illness” тоже 2 варианта - “No” или “Yes”.

Эти три столбца можно отнести к категориальным признакам.

Уникальные значения категориальных признаков можно кодировать целыми числами. Для этого можно использовать LabelEncoder из scikit-learn.

Импортируем LabelEncoder:

[11]:
```
from sklearn.preprocessing import LabelEncoder
```

Преобразуем столбец “City”:

[12]:
```
letypecity = LabelEncoder()
learrcity = letypecity.fit_transform(data["City"])
data["City"] = learrcity
data = data.astype({"City":"int64"})
```

Проверим преобразование:

[13]:
```
np.unique(learrcity)
```

[13]:
```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

Аналогично преобразуем столбцы “Gender” и “Illness”:

[14]:
```
letypegender = LabelEncoder()
learrgender = letypegender.fit_transform(data["Gender"])
data["Gender"] = learrgender
data = data.astype({"Gender":"int64"})
```

[15]:
```
np.unique(learrgender)
```

[15]:
```
array([0, 1])
```

[16]:
```
letypeill = LabelEncoder()
learrill = letypeill.fit_transform(data["Illness"])
data["Illness"] = learrill
data = data.astype({"Illness":"int64"})
```

[17]:
```
np.unique(learrill)
```

[17]:
```
array([0, 1])
```

Выведем типы столбцов после преобразования:

[18]:
```
data.dtypes
```

[18]:
```
City         int64
Gender       int64
Age          int64
Income     float64
Illness      int64
dtype: object
```

2.4. Проверка наличия пропусков

Проверим наличие пропусков:

[19]:
```
data.isnull().sum()
```

[19]:
```
City       0
Gender     0
Age        0
```

```
Income     0
Illness    0
dtype: int64
```

Видим, что пропуски не наблюдаются.

2.5. Корреляционный анализ

Корреляционный анализ помогает найти корреляции с целевом признаком, а также выявить линейно независимые нецелевые признаки:

В качестве целевого признака выберем столбец “Gender” (0 - мужчины, 1 - женщины).

Построим корреляционную матрицу:

[20]:
```
data.corr()
```

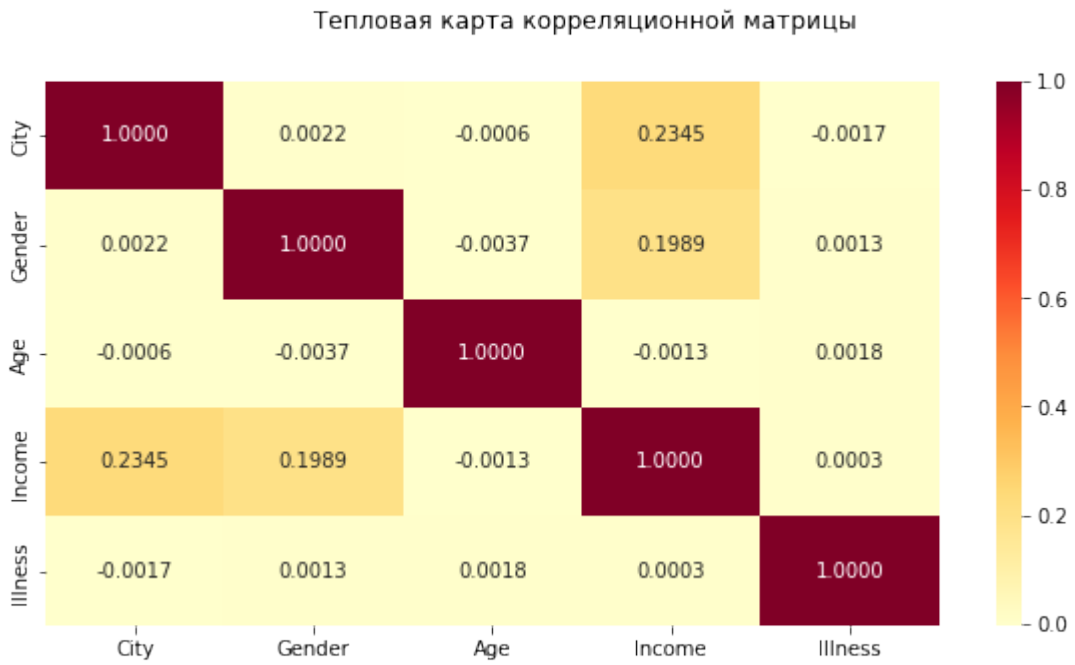
[20]:

	City	Gender	Age	Income	Illness
City	1.000000	0.002188	-0.000636	0.234541	-0.001712
Gender	0.002188	1.000000	-0.003653	0.198888	0.001297
Age	-0.000636	-0.003653	1.000000	-0.001318	0.001811
Income	0.234541	0.198888	-0.001318	1.000000	0.000298
Illness	-0.001712	0.001297	0.001811	0.000298	1.000000

Для визуализации корреляционной матрицы построим тепловую карту:

[21]:
```
fig, ax = plt.subplots(1, 1, sharex='col', sharey='row', figsize=(10,5))
fig.suptitle('Тепловая карта корреляционной матрицы')
sns.heatmap(data.corr(), ax=ax, annot=True, fmt='.4f', cmap="YlOrRd")
```

[21]:
```
<AxesSubplot:>
```



На основе корреляционной матрицы можно сделать следующие выводы: - Целевой признак наиболее сильно коррелирует с доходом (“Income”, 0.20) - Целевой признак слабо коррелирует с городом (“City”, 0.0022), возрастом (“Age”, -0.0037) и болезнью (“Illness”, 0.0013) - Наблюдается корреляция города (“City”) с доходом (“Income”) - 0.23

Сильно корреляции (>0.5) не наблюдается, значит все признаки можно оставить в модели.

2.6. Диаграмма рассеяния

Трансформируем обратно данные для столбцов “City” и “Gender”:

[22]:
```
cities = {
0: 'Austin',
1: 'Boston',
2: 'Dallas',
3: 'Los Angeles',
4: 'Mountain View',
5: 'New York City',
6: 'San Diego',
7: 'Washington D.C.',
}
data['City'] = data['City'].replace(cities)

genders = {
0: 'Male',
1: 'Female',
}
data['Gender'] = data['Gender'].replace(genders)
```

[23]:
```
data.head()
```

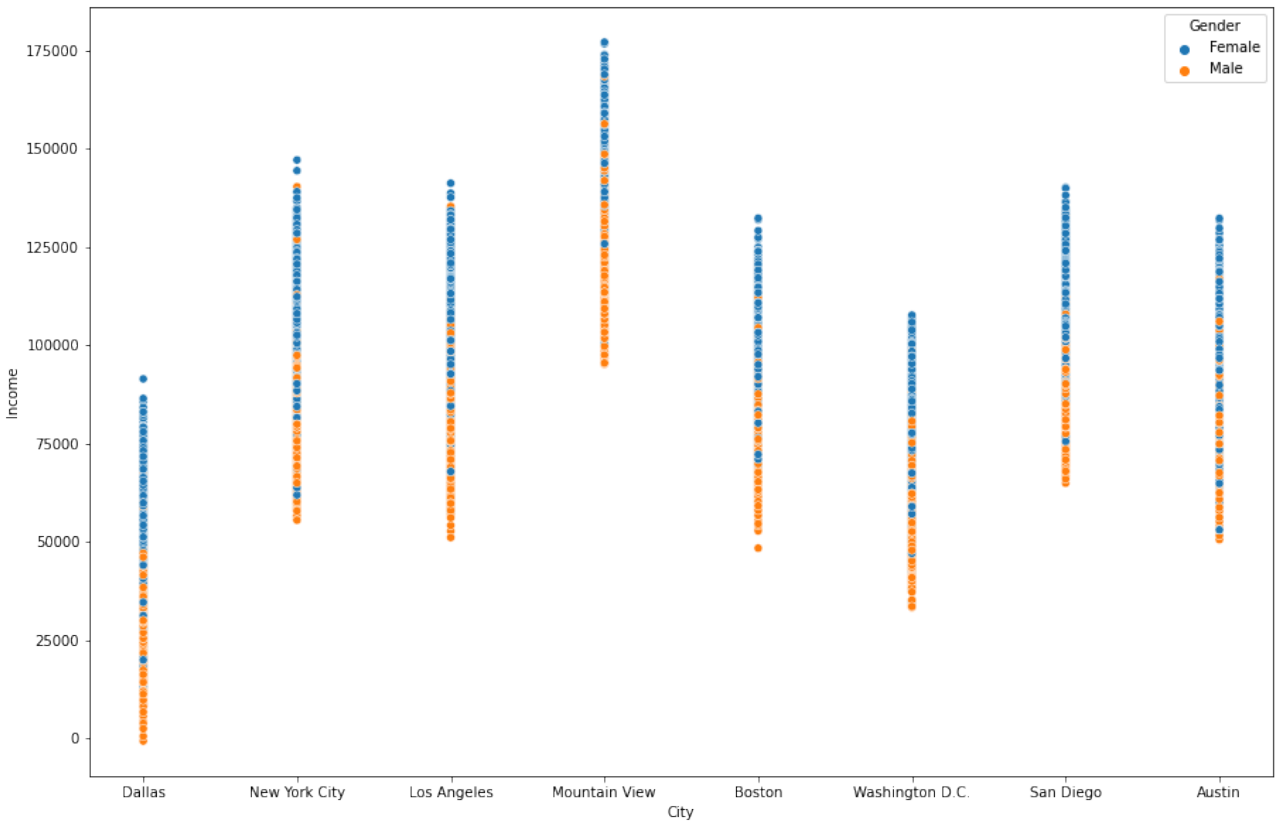
[23]:

	City	Gender	Age	Income	Illness
0	Dallas	Female	41	40367.0	0
1	Dallas	Female	54	45084.0	0
2	Dallas	Female	42	52483.0	0
3	Dallas	Female	40	40941.0	0
4	Dallas	Female	46	50289.0	0

Построим диаграмму рассеяния для столбцов “City” и “Income”, покрасим относительно “Gender”:

[24]:
```
fig, ax = plt.subplots(figsize=(15,10))
sns.scatterplot(ax=ax, x='City', y='Income', hue='Gender', data=data)
```

[24]:
```
<AxesSubplot:xlabel='City', ylabel='Income'>
```



Из диаграммы видно следующее: - Доход женщин выше, чем доход мужчин во всех городах
- Городами с самыми высокими доходами являются Mountain View, New York City и Los Angeles
- Городами с самыми низкими доходами являются Dallas, Washington D.C., Boston