Московский государственный технический университет
им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»

Рубежный контроль № 1

По курсу «методы машинного обучения в АСОИУ»

Выполнил:

студентка ИУ5-23М
Семенов И.А.

Проверил:

Гапанюк Ю.Е.

Подпись:

30.03.2024

Москва, 2024

Задание

Рубежный контроль №1

Выполнил: Семенов Илья, студент ИУ5-23М

Вариант 14, согласно ему номера задач: 14 и 34 для первой и второй соответственно. Для моей группы доп. требование: для произвольной колонки данных построить график "Ящик с усами (boxplot)".

14 - Для набора данных проведите нормализацию для одного (произвольного) числового признака с использованием функции "квадратный корень".

34 - Для набора данных проведите процедуру отбора признаков (feature selection). Используйте метод вложений (embedded method). Используйте подход на основе линейной или логистической регрессии (в зависимости от того, на решение какой задачи ориентирован выбранный Вами набор данных - задачи регрессии или задачи классификации).

Решение

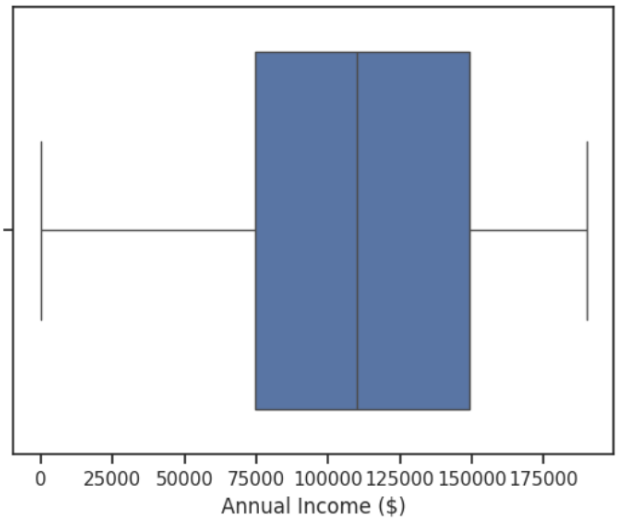
Подготовка данных

```
import pandas as pd
data = pd.read_csv('Customers.csv', sep=',', encoding='windows-1251')
df_1 = data.drop('Gender', axis=1)
df = df_1.drop('Profession', axis=1)
df.head()
```

	CustomerID	Age	Annual Income ($)	Spending Score (1-100)	Work Experience	Family Size
0	1	19	15000	39	1	4
1	2	21	35000	81	3	3
2	3	20	86000	6	1	1
3	4	23	59000	77	0	2
4	5	31	38000	40	2	6

```
import seaborn as sns

sns.set(style="ticks")
sns.boxplot(x=df['Annual Income ($)'])
```

```
<Axes: xlabel='Annual Income ($)'>
```



Задача №1 (14)

14 - Для набора данных проведите нормализацию для одного (произвольного) числового признака с использованием функции "квадратный корень".

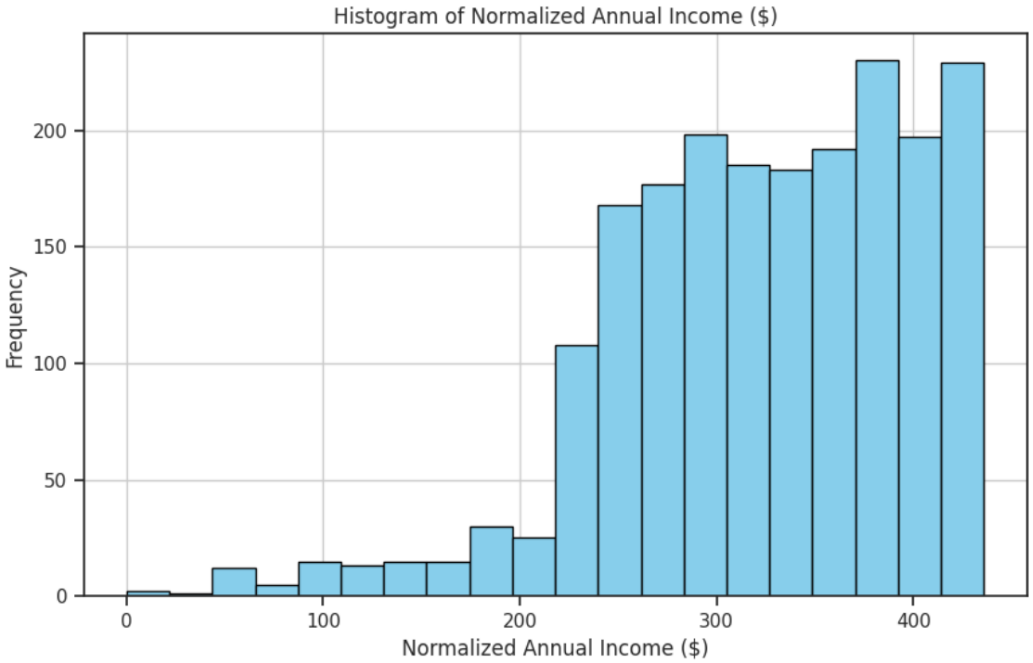
```
[8] import matplotlib.pyplot as plt  # noqa
    import numpy as np  # noqa
    import pandas as pd  # noqa
    import scipy.stats as stats  # noqa
```

```
# Выберите числовой признак для нормализации, например, 'Annual Income ($)'
numeric_feature = 'Annual Income ($)'

# Нормализация с использованием квадратного корня
data['Normalized_' + numeric_feature] = data[numeric_feature].apply(lambda x: x**0.5)

# Просмотр результата
print(data[['Annual Income ($)', 'Normalized_' + numeric_feature]].head())
```

```
   Annual Income ($)  Normalized_Annual Income ($)
0              15000                    122.474487
1              35000                    187.082869
2              86000                    293.257566
3              59000                    242.899156
4              38000                    194.935887
```

```
[10] plt.figure(figsize=(10, 6))
     plt.hist(data['Normalized_' + numeric_feature], bins=20, color='skyblue', edgecolor='black')
     plt.title('Histogram of Normalized Annual Income ($)')
     plt.xlabel('Normalized Annual Income ($)')
     plt.ylabel('Frequency')
     plt.grid(True)
     plt.show()
```

[10]



Задача №2 (34)

Для набора данных проведите процедуру отбора признаков (feature selection). Используйте метод вложений (embedded method). Используйте подход на основе линейной или логистической регрессии (в зависимости от того, на решение какой задачи ориентирован выбранный Вами набор данных - задачи регрессии или задачи классификации).

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.feature_selection import SelectFromModel

# Подготовка данных
X = df.drop('Spending Score (1-100)', axis=1)
y = df['Spending Score (1-100)']

# Разделение данных на обучающий и тестовый наборы
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

0 сек.

```
# Обучение модели линейной регрессии
model = LinearRegression()
model.fit(X_train, y_train)

# Отбор признаков
sfm = SelectFromModel(model)
sfm.fit(X_train, y_train)

# Важность признаков
selected_features = X_train.columns[sfm.get_support()]
print("Selected features:", selected_features)

# Построение модели с отобранными признаками
X_train_selected = sfm.transform(X_train)
X_test_selected = sfm.transform(X_test)

# Повторное обучение модели с отобранными признаками
model_selected = LinearRegression()
model_selected.fit(X_train_selected, y_train)

# Оценка производительности модели
train_score = model_selected.score(X_train_selected, y_train)
test_score = model_selected.score(X_test_selected, y_test)

print("Train R^2 score with selected features:", train_score)
print("Test R^2 score with selected features:", test_score)
```

```
Selected features: Index(['Work Experience', 'Family Size'], dtype='object')
Train R^2 score with selected features: 0.002389411232827876
Test R^2 score with selected features: -0.008291352952053188
```