МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. Н.Э. Баумана

Факультет «Информатика и системы управления» Кафедра «Систем обработки информации и управления»

ОТЧЕТ домашнее задание №3 по дисциплине «Методы машинного обучения»	_
Тема: «Обработка признаков (часть 2).»	
ИСПОЛНИТЕЛЬ: группа ИУ5И-21М	<u>Ли Яцзинь</u> ФИО подпись"
ПРЕПОДАВАТЕЛЬ:	<u>Гапанюк Ю .E</u>
Москва - 2024	

Задание:

- 1. Выбрать один или несколько наборов данных (датасетов) для решения следующих задач. Каждая задача может быть решена на отдельном датасете, или несколько задач могут быть решены на одном датасете. Просьба не использовать датасет, на котором данная задача решалась в лекции.
- 2. Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи:
 - і. масштабирование признаков (не менее чем тремя способами);
 - ii. обработку выбросов для числовых признаков (по одному способу для удаления выбросов и для замены выбросов);
 - ііі. обработку по крайней мере одного нестандартного признака (который не является числовым или категориальным);
 - iv. отбор признаков:
 - 1. один метод из группы методов фильтрации (filter methods);
 - 2. один метод из группы методов обертывания (wrapper methods);
 - 3. один метод из группы методов вложений (embedded methods)

масштабирование признаков

Прочитайте набор данных и получите соответствующую информацию о наборе данных:

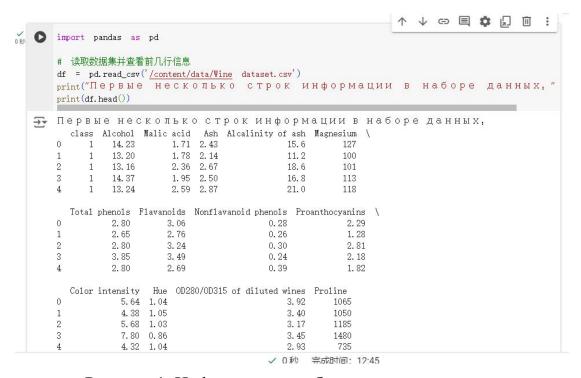


Рисунок 1. Информация с набором данных.

Как показано на рисунке 1, «Алкоголь», «Яблочная кислота», «Магний» выполняются в разных масштабах, поэтому масштабирование признаков необходимо выполнять перед каким-либо сравнением или объединением этих данных.

1. Z-оценки(Standardization)

```
    на основе Z-оценки(Standardization)

   🍙 # @title на основе Z-оценки(Standardization)
                   scaler = StandardScaler()
                   # 对特定列进行标准化
                   scaled_values = scaler.fit_transform(numeric_columns)
                   # 创建包含标准化后的值的 DataFrame
                  scaled_df = pd.DataFrame(scaled_values, columns=selected_columns)
                   # 显示标准化后的数据集
                  print("\n Стандартизированный набор данных.")
                  print(scaled_df)
 Стандартизированный набор данных.
                                    Alcohol Malic acid Magnesium
                  | National Mails and Magnesium | National Magnesium
                  ... 173 0. 876275 2. 974543 -0. 332922
                  174 0. 493343
175 0. 332758
                                                                       1. 412609 0. 158572
1. 744744 1. 422412
                   176 0. 209232
                                                                      0. 227694 1. 422412
1. 583165 -0. 262708
                  177 1.395086
                   [172 rowe v 2 columne]
```

Рисунок 2. Код Z-оценки и результаты

2. Масштабирование "Mean Normalisation"

```
↑ ↓ © ■ $ ♬ Ⅲ :

    Масштабирование "Mean Normalisation"

တ္တြဲ 🗘 # @title Масштабирование "Mean Normalisation"
          # 计算每列的均值和范围
         column_means = numeric_columns.mean()
column_ranges = numeric_columns.max() - numeric_columns.min()
         normalized_values = (numeric_columns - column_means) / column_ranges
         # 创建包含归一化后的值的 DataFrame
         normalized_df = pd.DataFrame(normalized_values, columns=selected_columns)
         # 显示归一化后的数据集
         print("\nНормализованный набор данных.")
         print(normalized_df)
         Нормализованный набор данных.
               Alcohol Malic acid Magnesium

    0
    0.323522
    -0.123784
    0.296287

    1
    0.052469
    -0.109950
    0.002809

    2
    0.041943
    0.004674
    0.013679

    3
    0.360364
    -0.076353
    0.144113

    4
    0.062995
    0.050129
    0.198461

                          0.654872 -0.051539
         173 0.186679
         174 0. 105101
175 0. 070890
                          0.310998 0.024548
0.384121 0.220200
         176 0.044574
                           0.050129 0.220200
         177 0.297206
                           0.348548 -0.040669
```

Рисунок 3. Код масштабирование и результаты

3.МіпМах-масштабирование

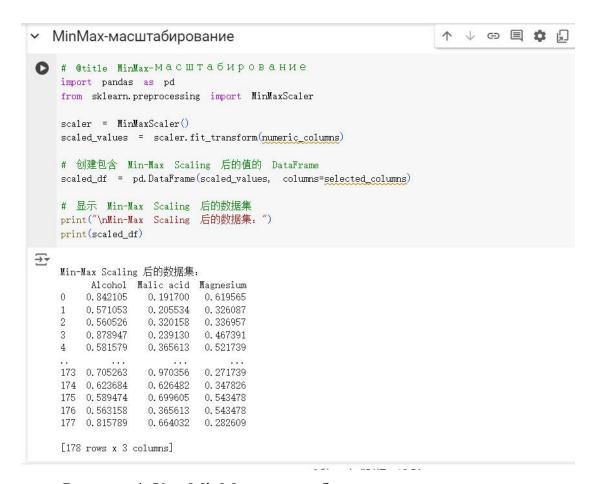


Рисунок 4. Код МіпМах-масштабированией результаты

Обработку выбросов для числовых признаков

1.Обнаружение выбросов

Обнаружение выбросов в данных с помощью метода z-оценки:

```
↑ ↓ ⊖ 팉 ◘ ఓ ॥ : |

    Z-score method

    # @title Z-score method
     import pandas as pd
     import numpy as np
     import re
     train = pd.read_csv('/content/data/houses_to_rent.csv')
     out = []
     def Zscore_outlier(df):
           # 使用正则表达式移除字符串中的货币符号和逗号,并将结果转换为浮点数
           df = df. str. replace(',', '') # 移除逗号
df = df. apply(lambda x: re. sub(r'['\d.]', '', x)) # 移除非数字字符,包括货币符号
            df = df.astype(float) # 转换为浮点数
            m = np.mean(df)
sd = np.std(df)
            for i in df:
                   z = (i - m) / sd
                   if np.abs(z) > 3:
                          out.append(i)
            print("Outliers:", out)
     Zscore_outlier(train['rent amount'])
① Outliers: [19500.0, 20000.0, 20000.0, 45000.0, 20000.0, 18000.0, 20000.0, 24000.0, 20000.0, 20000.0]
```

Рисунок 5- Обнаружение выбросов

2. Удалить выбросы

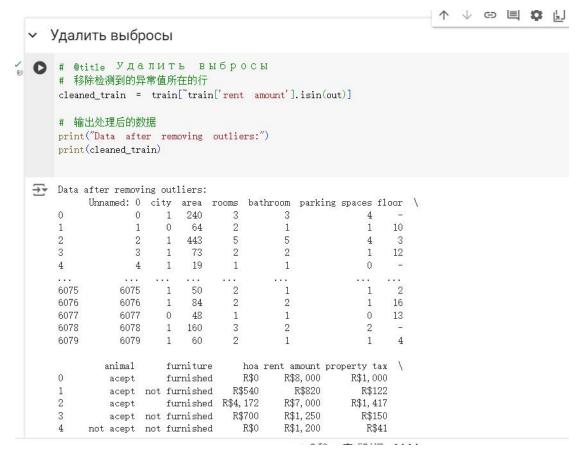


Рисунок 6-Код и результаты удаления выбросов

3. Заменить выбросы

```
# 计算中位数
           median = np.median(df)
           # Заменить выбросы медианой
           for i in range(len(df)):
                 z = np.abs((df[i] - m) / sd)
                 if z > 3:
                        df[i] = median
           # 打印处理后的数据
          print("Processed data:")
          print(df)
    Zscore_outlier(train['rent amount'])

→ Processed data:
          8000.0
           820.0
          7000.0
          1250.0
    3
          1200.0
          ...
1150.0
    6075
    6076
          2900.0
    6077
           950.0
          3500.0
    6078
    6079
          1900.0
    Name: rent amount, Length: 6080, dtype: float64
                                            / N 和 中中田河, 14.10
```

Рисунок 7-Код и результаты замены выбросов

Обработку по крайней мере нестандартного признака Загрузить набор данных

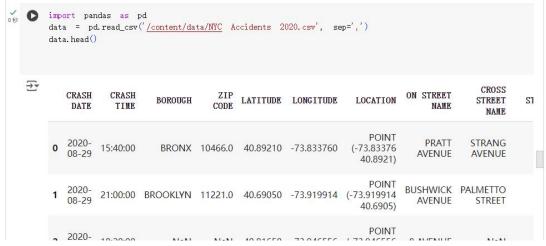


Рисунок 6-Код и результаты Загрузить набор данных

Обработка даты

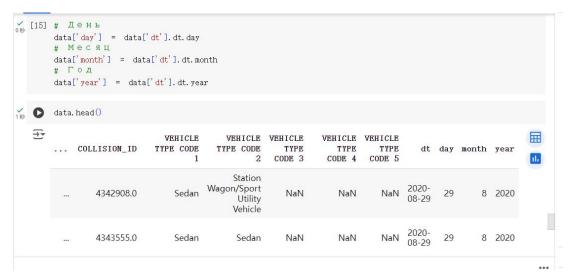


Рисунок 7-Код и результаты обработка даты

Отбор признаков

1. Методы фильтрации (filter methods).

Выбор признаков осуществлялся с использованием метода выбора порога дисперсии.

Рисунок 8-Код и результаты Методы фильтрации

2. Методы обертывания (wrapper methods).

```
data = pd_read_csy(url)

# 分割持征和目标变量
X = data_drop('NObeyesdad', axis=1) # 特征
y = data['NObeyesdad'] # 目标变量

# 初始化離机森林分类器
estimator = RandomForestClassifier()

# 初始化 FFE 特征选择器
selector = RFE(stimator, n_features_to_select=5, step=1)

# 拟合 FFE 特征选择器
selector = selector.fit(X, y)

# 获取选择的特征索引
selected_features_index = selector.get_support(indices=True)

# 根据所选特征的索引获取特征名称
selected_features = X.columns[selected_features_index]

# 打印选择的特征
print('所选特征, ")
print(selected_features)

F/选特征,
Index(['Height', 'Weight', 'FCVC_minmax', 'FAF_minmax', 'TUE_z'], dtype='object')
```

Рисунок 9-Код и результаты Методы обертывания

3. Методы вложений (embedded methods).

```
# 读取数据
ur1 = "/content/data/estimation_of_obesity_levels_based_on_eating_habits_and_physical_condition.csv"
data = pd.read_csv(ur1)

# 分劃特征和目标变量
X = data_drop('Nobeyesdad', axis=1) # 特征
y = data['Nobeyesdad'] # 目标变量

# 初始化 Lasso 模型
lasso_model = Lasso(alpha=0.1, max_iter=10000)]

# 使用 Lasso 模型进行特征选择
selector_lasso = SelectFromModel(lasso_model)

# 拟合 SelectFromModel 特征选择器
selector_lasso.fit(X, y)

# 获取选择的特征
selected_features_lasso = X.columns[selector_lasso.get_support()]

# 打印使用 Embedded Methods 选择的特征
print('使用 Embedded Methods 选择的特征: ")
print(selected_features_lasso)

**

使用 Embedded Methods 选择的特征: Index(('Weight'), dtype='object')
```

Рисунок 10-Код и результаты Методы вложений