# Лабораторная работа №3: Обработка признаков (часть 2)

# 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:

* i. масштабирование признаков (не менее чем тремя способами);
* ii. обработку выбросов для числовых признаков (по одному способу для удаления выбросов и для замены выбросов);
* iii. обработку по крайней мере одного нестандартного признака (который не является числовым или категориальным);
* iv. отбор признаков:
  + один метод из группы методов фильтрации (filter methods);
  + один метод из группы методов обертывания (wrapper methods);
  + один метод из группы методов вложений (embedded methods).

# Текстовое описание датасета

В качестве датасета будем использовать набор данных, содержащий данные с информацией об автомобиле.

Набор данных имеет следующие атрибуты:

* Name: Уникальный идентификатор для каждого автомобиля.
* MPG: Эффективность использования топлива измеряется в милях на галлон.
* Cylinders: количество цилиндров в двигателе.
* Displacement: объем двигателя с указанием его размера или мощности.
* Horsepower: Выходная мощность двигателя.
* Weight: Вес автомобиля.
* Acceleration: Возможность увеличения скорости, измеряемая в секундах.
* Model Year: год выпуска модели автомобиля.
* Origin: Страна или регион происхождения каждого автомобиля.

### Импорт библиотек

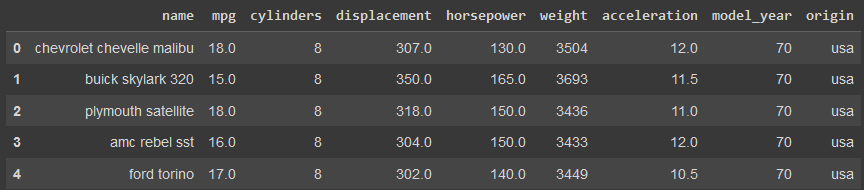
import numpy as np  
import pandas as pd  
from matplotlib import pyplot as plt  
from sklearn.preprocessing import PolynomialFeatures  
from sklearn.preprocessing import StandardScaler, MinMaxScaler, RobustScaler  
from sklearn.feature\_selection import SelectFromModel  
from sklearn.linear\_model import LinearRegression, Lasso  
import seaborn as sns  
import warnings  
import math  
warnings.simplefilter('ignore')

### Загрузка данных

data = pd.read\_csv('Automobile.csv')

Выведем первые строки

data.head()

Выведем типы данных для всех столбцов

data.dtypes

name object  
mpg float64  
cylinders int64  
displacement float64  
horsepower float64  
weight int64  
acceleration float64  
model\_year int64  
origin object  
dtype: object

## Обработка нестандартных признаков

### Признак name.

Извлечем бренд автомобиля из названия

data['brand'] = data['name'].apply(lambda x: x.split()[0])  
data.head()

## Масштабирование признаков

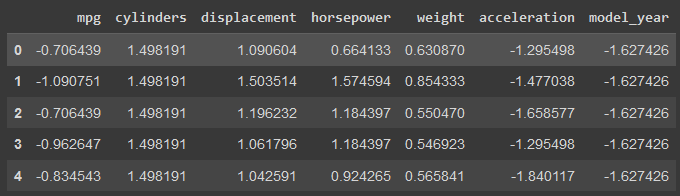
numeric\_columns = [column for column in data.columns if data.dtypes[column] != object]

numeric\_columns

['mpg',  
 'cylinders',  
 'displacement',  
 'horsepower',  
 'weight',  
 'acceleration',  
 'model\_year']

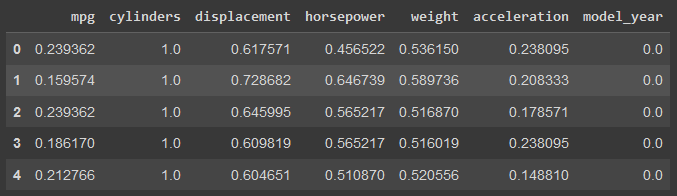
StandardScaler

data1 = pd.DataFrame(StandardScaler().fit\_transform(data[numeric\_columns]), columns=numeric\_columns);  
data1.head()



MinMaxScaler

data2 = pd.DataFrame(MinMaxScaler().fit\_transform(data[numeric\_columns]), columns=numeric\_columns);  
data2.head()

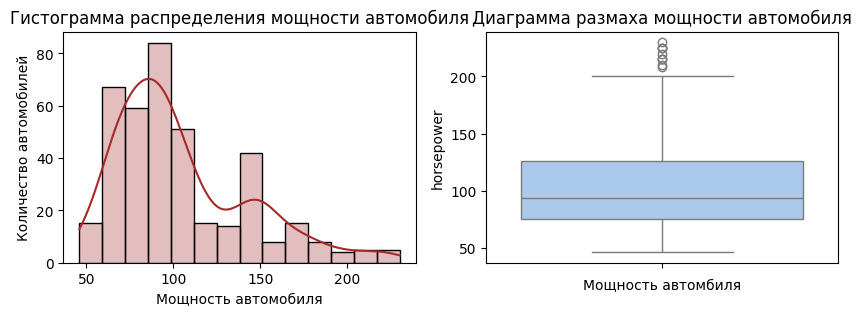
RobustScaler

data3 = pd.DataFrame(RobustScaler().fit\_transform(data[numeric\_columns]), columns=numeric\_columns);  
data3.head()

## Обработка выбросов для числовых признаков

Замена выбросов

fig = plt.figure(figsize=(10, 3))  
axes = fig.subplots(1, 2)  
sns.histplot(data['horsepower'], kde=True, color='brown', alpha=0.3, ax=axes[0])  
axes[0].title.set\_text(f"Гистограмма распределения мощности автомобиля")  
axes[0].set\_xlabel('Мощность автомобиля')  
axes[0].set\_ylabel('Количество автомобилей')  
sns.boxplot(data['horsepower'], palette='pastel', ax=axes[1])  
axes[1].title.set\_text(f"Диаграмма размаха мощности автомобиля")  
axes[1].set\_xlabel('Мощность автомбиля')  
plt.show();

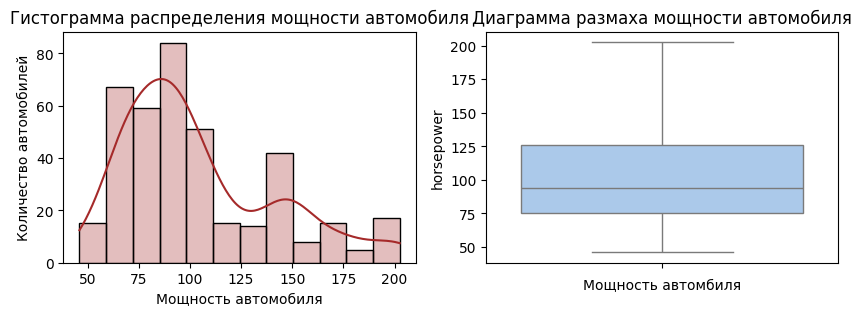


K = 1.5  
col = 'horsepower'  
IQR = data[col].quantile(0.75) - data[col].quantile(0.25)  
lower\_boundary = data[col].quantile(0.25) - (K \* IQR)  
upper\_boundary = data[col].quantile(0.75) + (K \* IQR)  
round(lower\_boundary, 2), round(upper\_boundary, 2)

(-1.5, 202.5)

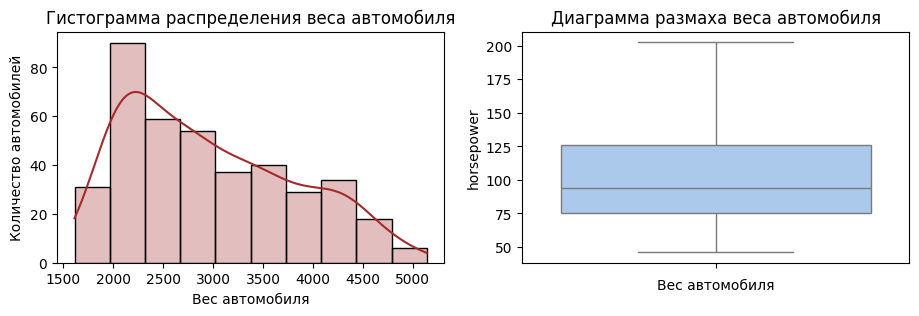
data[col] = np.where(data[col] > upper\_boundary, upper\_boundary, np.where(data[col] < lower\_boundary, lower\_boundary, data[col]))

fig = plt.figure(figsize=(10, 3))  
axes = fig.subplots(1, 2)  
sns.histplot(data['horsepower'], kde=True, color='brown', alpha=0.3, ax=axes[0])  
axes[0].title.set\_text(f"Гистограмма распределения мощности автомобиля")  
axes[0].set\_xlabel('Мощность автомобиля')  
axes[0].set\_ylabel('Количество автомобилей')  
sns.boxplot(data['horsepower'], palette='pastel', ax=axes[1])  
axes[1].title.set\_text(f"Диаграмма размаха мощности автомобиля")  
axes[1].set\_xlabel('Мощность автомбиля')  
plt.show();



Удаление выбросов

fig = plt.figure(figsize=(11, 3))  
axes = fig.subplots(1, 2)  
sns.histplot(data['weight'], kde=True, color='brown', alpha=0.3, ax=axes[0])  
axes[0].title.set\_text(f"Гистограмма распределения веса автомобиля")  
axes[0].set\_xlabel('Вес автомобиля')  
axes[0].set\_ylabel('Количество автомобилей')  
sns.boxplot(data['horsepower'], palette='pastel', ax=axes[1])  
axes[1].title.set\_text(f"Диаграмма размаха веса автомобиля")  
axes[1].set\_xlabel('Вес автомобиля')  
plt.show();

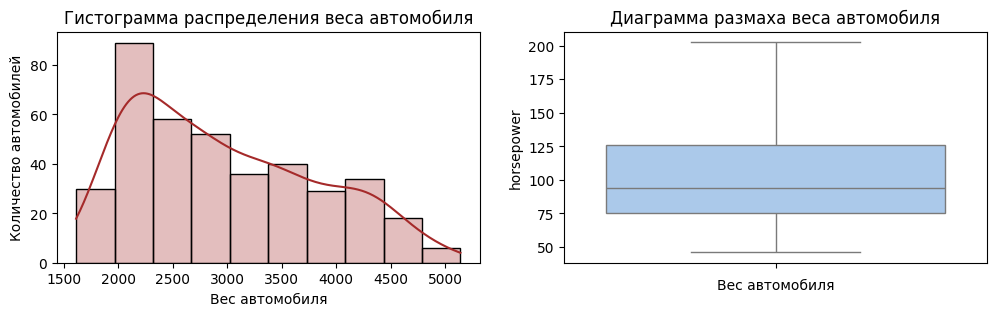


K = 1.5  
col = 'weight'  
IQR = data[col].quantile(0.75) - data[col].quantile(0.25)  
lower\_boundary = data[col].quantile(0.25) - (K \* IQR)  
upper\_boundary = data[col].quantile(0.75) + (K \* IQR)  
round(lower\_boundary, 2), round(upper\_boundary, 2)

(147.38, 5684.38)

data = data[data['horsepower'] < 5684.38]

fig = plt.figure(figsize=(12, 3))  
axes = fig.subplots(1, 2)  
sns.histplot(data['weight'], kde=True, color='brown', alpha=0.3, ax=axes[0])  
axes[0].title.set\_text(f"Гистограмма распределения веса автомобиля")  
axes[0].set\_xlabel('Вес автомобиля')  
axes[0].set\_ylabel('Количество автомобилей')  
sns.boxplot(data['horsepower'], palette='pastel', ax=axes[1])  
axes[1].title.set\_text(f"Диаграмма размаха веса автомобиля")  
axes[1].set\_xlabel('Вес автомобиля')  
plt.show();



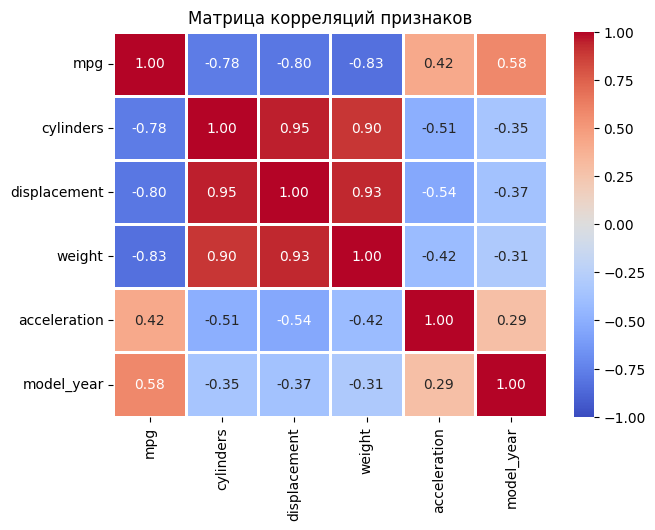
## Отбор признаков

Метод фильтрации

print(f'Всего записей: {data.shape[0]}')  
print('--------------------')  
for column in data.columns:  
 print(f'{column}: {data[column].value\_counts().count()} уникальных значений', end='\n\n')

Всего записей: 392  
--------------------  
name: 301 уникальных значений  
  
mpg: 127 уникальных значений  
  
cylinders: 5 уникальных значений  
  
displacement: 81 уникальных значений  
  
horsepower: 88 уникальных значений  
  
weight: 346 уникальных значений  
  
acceleration: 95 уникальных значений  
  
model\_year: 13 уникальных значений  
  
origin: 3 уникальных значений  
  
brand: 37 уникальных значений

plt.figure(figsize=(7, 5))  
sns.heatmap(data1.drop('horsepower', axis=1).corr(), vmin=-1, vmax=1, annot=True, cmap='coolwarm', fmt='.2f', linewidth=1)  
plt.title('Матрица корреляций признаков');



plt.figure(figsize=(7, 4))  
sns.heatmap(pd.DataFrame(data.corr()['horsepower'].sort\_values(ascending=False)[1:]), vmin=-1, vmax=1, annot=True, fmt='.2f', cmap='coolwarm')  
plt.title('Корреляция признаков с мощностью автомобиля');

