Лабораторная работа №1: «Линейная регрессия»

Задание:

Набор данных ex1data1.txt представляет собой текстовый файл, содержащий информацию о населении городов (первое число в строке) и прибыли ресторана, достигнутой в этом городе (второе число в строке). Отрицательное значение прибыли означает, что в данном городе ресторан терпит убытки.

Набор данных ex1data2.txt представляет собой текстовый файл, содержащий информацию о площади дома в квадратных футах (первое число в строке), количестве комнат в доме (второе число в строке) и стоимости дома (третье число).

Задание:

1. Загрузите набор данных ex1data1.txt из текстового файла.
2. Постройте график зависимости прибыли ресторана от населения города, в котором он расположен.
3. Реализуйте функцию потерь J(θ) для набора данных ex1data1.txt.
4. Реализуйте функцию градиентного спуска для выбора параметров модели. Постройте полученную модель (функцию) совместно с графиком из пункта 2.
5. Постройте трехмерный график зависимости функции потерь от параметров модели (θ0 и θ1) как в виде поверхности, так и в виде изолиний (contour plot).
6. Загрузите набор данных ex1data2.txt из текстового файла.
7. Произведите нормализацию признаков. Повлияло ли это на скорость сходимости градиентного спуска? Ответ дайте в виде графика.
8. Реализуйте функции потерь J(θ) и градиентного спуска для случая многомерной линейной регрессии с использованием векторизации.
9. Покажите, что векторизация дает прирост производительности.
10. Попробуйте изменить параметр α (коэффициент обучения). Как при этом изменяется график функции потерь в зависимости от числа итераций градиентного спуск? Результат изобразите в качестве графика.
11. Постройте модель, используя аналитическое решение, которое может быть получено методом наименьших квадратов. Сравните результаты данной модели с моделью, полученной с помощью градиентного спуска.

Реализация:

In[1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pnd
```

1.1 Загрузите набор данных ex1data1.txt из текстового файла.

In[2]:

```
ex1data1 = np.loadtxt('ex1data1.txt', delimiter=',')
x, y = np.expand_dims(ex1data1[:, 0], axis=1), ex1data1[:, 1]
```

1.2 Постройте график зависимости прибыли ресторана от населения города, в котором он расположен (рисунок 1).

In[3]:

```
plt.scatter(x, y)
plt.xlabel('Population', size=16)
plt.ylabel('Income', size=16)
plt.show()
```

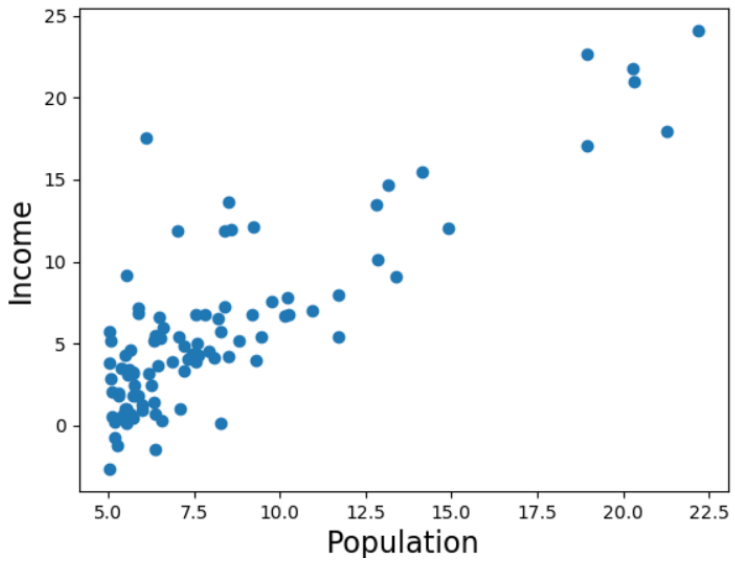


Рисунок 1 – иллюстрация зависимости прибыли ресторана от населения

1.3 Реализуйте функцию потерь J(0) для набора данных ex1data1.txt

In[4]:

```
def MSECost(y, theta):
    def predictedValue(x):
        return theta[0]*x + theta[1]

    def lossValue(existent, predicted):
        return (existent - predicted)

    hypothesis = map(predictedValue, y)
    lossValue = map(lossValue, y, hypothesis)
    costValue = sum(map(lambda x: x**2, lossValue)) / (2*len(y))
    return hypothesis, lossValue, costValue

theta = [0.9089, 0.9089]
_, _, cost = MSECost(x, theta)
print(cost)
```

Out[4]: 0.07520593

1.4 Реализуйте функцию градиентного спуска для выбора параметров модели. Постройте полученную модель (функцию) совместно с графиком из пункта 2.

In[5]:

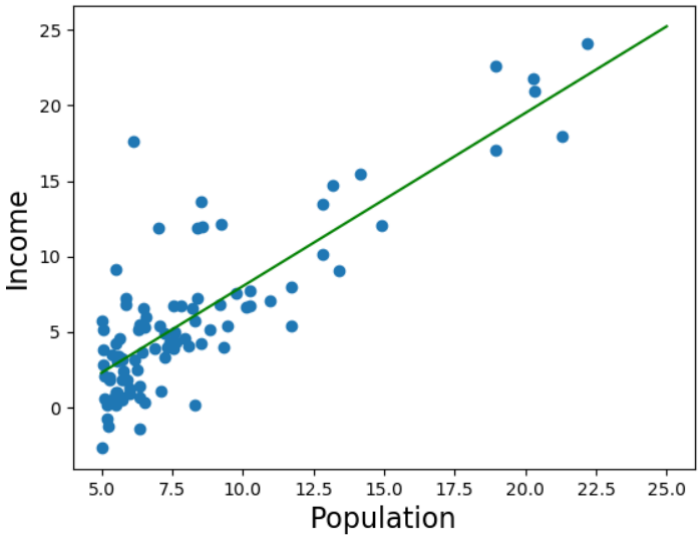
```
def f(x, y, theta):
    return theta[0] + x * theta[1] - y

def gradient(x, y, theta, a, iter_count):
    training_logs = []
    for i in range(iter_count):
        dj_dt0, dj_dt1 =0, 0
        count = len(x)
        for j in range(len(x)):
            h = f(x[j], y[j], theta)
            dj_dt0 += h
            dj_dt1 += x[j] * h
        dj_dt0 /= count
        dj_dt1 /= count
        theta[0] -= a* dj_dt0
        theta[1] -=a * dj_dt1

        curr_cost = cost_func(x, y, theta)
        training_logs.append([i, curr_cost, theta[0]], theta[1])
    return theta, training_logs
```

In[6]:

```
theta, logs = gradient(
    x, y, theta = [0, 0], a = 0.012, iter_count = 1000
)
```

In[7]:

```
plt.scatter(x, y)
plt.xlabel('Population', size=16)
plt.ylabel('Income', size=16)
x = np.linspace(5, 25, 3)
plt.plot(x, theta[0] + theta[1] * x, 'g')
plt.show()
```



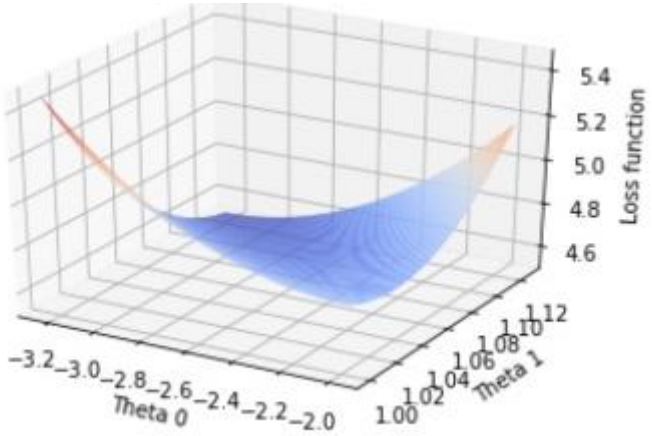
1.5 Постройте трехмерный график зависимости функции потерь от параметров модели (θ0 и θ1) как в виде поверхности, так и в виде изолиний (contour plot).

In[8]:

```
logs_df = pd.DataFrame(logs, columns=['iter', 'loss', 'theta0', 'theta1'])
data = logs_df[logs_df['theta1'] > 1]
X, Y = np.meshgrid(data['theta0'], data['theta1'])
Z = np.zeros((data['theta0'].size, data['theta1'].size))

for i, theta0 in enumerate(data['theta0']):
    for j, theta1 in enumerate(data['theta1']):
        Z[i, j] = cost_func(x, y, [theta0, theta1])

from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
```

In[9]:

```
fig = plt.figure()
ax = fig.gca(projection = '3d')
ax.plot_surface(X, Y, Z, cmap=cm.coolwarm, linewidth=0)
ax.set_xlabel('Theta 0')
ax.set_ylabel('Theta 1')
ax.set_zlabel('Loss function')
plt.show()
```




```
fig, ax = plt.subplots()
CS = ax.contour(X, Y, Z, cmap='viridis')
ax.clabel(CS, inline=1, fontsize=10)
ax.set_xlabel('Theta 0')
ax.set_ylabel('Theta 1')
plt.show()
```