# Лабораторная работа №2. Линейная регрессия.

В данной работе вам предстоит реализовать линейную регрессию от одной переменной методом градиентного спуска. Работа выполняется на языке программирования Python с использованием библиотек Numpy и Matplotlib.

Для выполнения работы запустите редактор (Visual Studio Code, PyCharm или другие) и откройте файл lab02.py

В процессе выполнения работы вам необходимо дополнить файл-заготовку lab02.py необходимыми операторами и вычислениями до работающей программы. Когда вы правильно реализуете одно из заданий работы, можете переходить к следующему. Большинство шагов работает на основе функций, сделанных на предыдущих шагах. Поэтому не стоит пропускать шаги или переходить к следующему, если в программе допущена ошибка, и она не работает как следует.

Если у вас возникли затруднения в выполнении работы, ознакомьтесь с лекционным материалом, документацией по библиотекам Numpy и Matplotlib, поищите решения в сети Интернет, проконсультируйтесь с преподавателем.

В этой лабораторной работе необходимо реализовать линейную регрессию для построения модели предсказания величины дохода сети небольших кофеен в зависимости от количества жителей города.

Файл ex1data1.txt содержит обучающий набор данных из двух столбцов:

– количество жителей города (в десятках тысяч человек);

– среднемесячная величина дохода (в десятках тысяч долларов).

В файле lab02.py написаны заготовки функций, которые вам необходимо будет модифицировать в процессе выполнения лабораторной работы. На текущий момент они не выполняют правильных действий для получения работающей линейной регрессии.

В начале файла размещена основная функция main(), в которой выполняются основные действия.

### 1. Загрузка и отображение данных

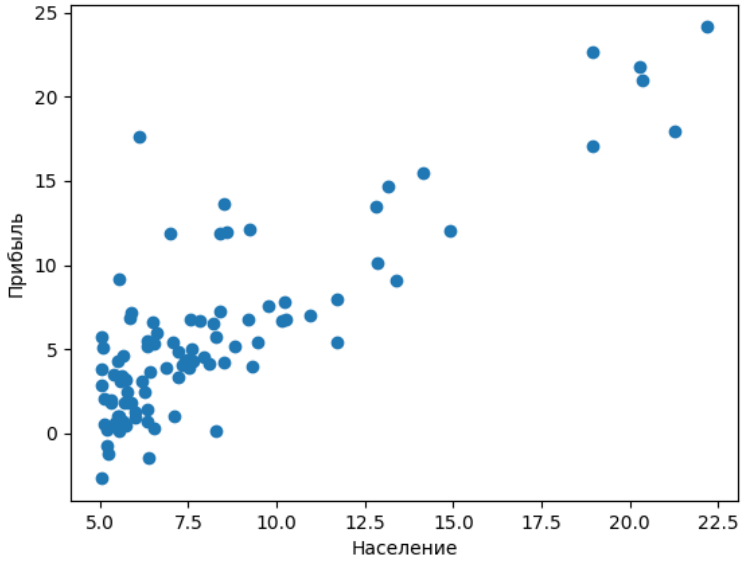
На первом шаге загружаются обучающие данные при помощи оператора loadtxt библиотеки NumPy. В результате выполнения команды текстовый файл из 97 строк и двух столбцов чисел, разделенных запятой, помещается в переменную data в виде матрицы 97×2.

Обучающие данные необходимо разбить на вектор объектов *x* (из переменной data выбирается первый столбец, получается вектор из 97 элементов) и вектор ответов *y* (второй столбец из 97 элементов).

Первым заданием является написание программы отображения обучающей выборки на графике. Вам необходимо дополнить функцию main(), чтобы она отображала график с населением города по горизонтальной оси, прибылью по вертикальной оси, на плоскости должны быть нанесены точки обучающей выборки.

Для выполнения этой задачи используйте оператор scatter библиотеки pyplot. Дополнительно можно установить подписи осей командами xlabel, ylabel, заголовок графика командой title.

Пример того, как должен выглядеть график:




После успешного отображения графика можно переходить к следующему шагу. В функции main() в нескольких местах размещены операторы выхода return, чтобы не выполнять действий, которые вы еще не запрограммировали. Операторы предваряются соответствующим комментарием:

```
# Удалите следующую строку для продолжения работы
return
```

Перед переходом к следующему шагу, удалите или закомментируйте эти строки.

**2. Функция стоимости**

На этом шаге вам необходимо наполнить необходимыми вычислениями функцию compute_cost. На вход она принимает матрицу *X* (размером 97×2, в которой первый столбец единиц, а второй равен вектору *x*), векторы *y* и *theta*. Ответом ее должно быть значение функции стоимости *J* (одно число).

Формулы для вычисления функции стоимости $J(\theta)$ для линейной регрессии одной переменной даны в разделе 1.3 учебника (лекция №2).

Когда функция compute_cost готова, запустите программу. В терминале вы увидите результаты вычисления функции стоимости для обучающей выборки от различных значений *theta* (в программу заложено два варианта), а также ожидаемые значения, которые должны быть получены, если функция написана правильно. Если ваши значения совпадают со всеми ожидаемыми, переходите к следующему шагу.

**3. Градиентный спуск**

На этом шаге нужно обучить регрессионную модель методом градиентного спуска. В программе имеется заготовка функции:

gradient_descent (X, y, theta, alpha, num_iters)

На вход она принимает *X*, *y*, *theta*, *alpha* (параметр скорости спуска), *num_iters* (количество итераций спуска). Функция должна возвращать *theta* – это найденное алгоритмом значение тета, при котором достигается минимум функции стоимости.

Функция gradient_descent является пустой, она не выполняет нужных действий. Вам необходимо наполнить ее алгоритмом градиентного спуска. Необходимый теоретический материал содержится в разделе 1.4 учебника (лекция №2).

Рекомендуется реализовать функцию в матричной форме, что облегчит выполнение последующих работ и является более правильным и простым в записи.

После того, как функция готова, запустите программу. В командном окне выводится значение найденного вашей функцией тета и сравнивается с ожидаемым (правильным) значением. Если они не совпадают, проверьте решение и попробуйте заново.

Затем программа отображает на графике обучающей выборки линию гипотезы, полученной с найденными параметрами тета. При правильно решении она должна проходить через середину множества точек обучающей выборки.

Далее мы можем воспользоваться полученной гипотезой и предсказать значения возможной прибыли для пары городов с населением, которого не было в обучающей выборке.

Гипотеза линейной регрессии в матричной форме имеет вид (раздел 2.1 учебника, лекция №3):

$h(x) = \theta x$ или $h(x) = x\theta$

(в зависимости от формы векторов может использоваться транспонирование).

Попробуйте добавить свои варианты и получить предсказание.

**4 Визуализация функции стоимости**

На данном шаге вам не нужно будет писать собственных функций. В программе строятся два графика функции стоимости: в виде поверхности и контурного графика. Изучите полученные графики, сопоставьте их с найденным градиентным спуском значением тета. Найдите оптимальное значение тета на графиках.

На этом выполнение лабораторной работы №2 завершается.