**Дополнительная практическая работа: «Множественная линейная регрессия»**

Оглавление

[Цель работы 1](#_Toc54468939)

[Задачи работы 1](#_Toc54468940)

[Перечень обеспечивающих средств 2](#_Toc54468941)

[Общие теоретические сведения 2](#_Toc54468942)

[**Линейная регрессия** 2](#_Toc54468943)

[**Градиентный спуск** 2](#_Toc54468944)

[**Допущения линейной регрессии** 3](#_Toc54468945)

[**Ограничения линейной регрессии** 3](#_Toc54468946)

[**Регуляризация** 3](#_Toc54468947)

[**L1-регуляризация (lasso)** 3](#_Toc54468948)

[**L2-регуляризация (ridge)** 4](#_Toc54468949)

[**Elastic Net регуляризация** 4](#_Toc54468950)

[**Множественная линейная регрессия** 4](#_Toc54468951)

[Задание 5](#_Toc54468952)

[Требования к отчету 6](#_Toc54468953)

[Литература 6](#_Toc54468954)

## Цель работы

Получить практические навыки использования линейной регрессии.

## Задачи работы

1. Научиться аналитически решать задачу линейной регрессии.
2. Научиться решать задачу линейной регрессии с помощью библиотеки sklearn.

## Перечень обеспечивающих средств

1. ПК.
2. Учебно-методическая литература.
3. Задания для самостоятельного выполнения.

## Общие теоретические сведения

### **Линейная регрессия**

Данные: пары значений , где .

называется предиктором или регрессором,

называется зависимой переменной.

Задача: Найти такие значения *a* и *b*, чтобы функция как можно точнее аппроксимировала *y*, т.е. чтобы для всех *i*.

Метрика производительности – среднеквадратичная ошибка:

где - значение из данных, - результат работы модели.

### **Градиентный спуск**

1. Случайным образом выбираем точку .
2. Вычисляем значения частных производных ошибки.
3. Изменяем координаты так, чтобы двигаться в сторону уменьшения производной:

, .

1. Если достаточно мало, то завершаем. Иначе – возвращаемся к шагу 2.

### **Допущения линейной регрессии**

Остатки: величины .

Допущения линейной регрессии

* Между *x* и *y* есть линейная зависимость.
* Остатки распределены нормальным образом.
* Среднее значение остатков равно нулю.
* Дисперсия остатков постоянна.

### **Ограничения линейной регрессии**

* Низкая точность при аппроксимации нелинейных функций.
* Нельзя использовать для вычислений вне известного интервала.
* Считаем, что предикторы не содержат ошибок измерений.
* Нет ограничений области значений.

### **Регуляризация**

Если данных мало, а модель сложная, то высока вероятность переобучения.

Регуляризация – добавление дополнительных слагаемых к метрике производительности для того, чтобы штрафовать модель за излишне сложные решения и, таким образом, препятствовать переобучению.

Смещение увеличивается, разброс уменьшается.

### **L1-регуляризация (lasso)**

L1-регуляризация обнуляет параметры , которые вносят в основном шум.

### **L2-регуляризация (ridge)**

L2-регуляризация не даёт значениям параметров бесконтрольно увеличиваться.

### **Elastic Net регуляризация**

### **Множественная линейная регрессия**

Данные: , где .

Задача: Найти такие значения , где *k* = 1, …, *K*, чтобы функция

как можно точнее аппроксимировала *y*,

т.е. чтобы для всех *i*.

, где

## Задание

**Пояснение**

Для сохранения результатов данной работы вам понадобится один файл. Назовите его «*Фамилия* – дополнительное задание.ipynb».

**Часть 1**

* Обновите свой репозиторий, созданный в практической работе №1, из оригинального репозитория:

<https://github.com/mosalov/Notebook_For_AI_Main>.

**Часть 2**

* Откройте свой репозиторий в Binder (<https://mybinder.org/>).
* Откройте файл «task5.ipynb».
* Изучите, при необходимости – выполните повторно, приведённый в файле код.
* С помощью библиотек sklearn по аналогии с имеющимся кодом решите задачу линейной регрессии, а также примените L1, L2 и ElasticNet регуляризации для случая множественной регрессии: зависимая переменная – Weight, регрессоры: Length1, Length2, Length3, Height, Width.
* Сохраните код в ipynb-файле. Необходимые пояснения опишите в своём docx/doc-файле.

**Замечания**

* 1. X\_train будет двумерным массивом – это нормально.
  2. Сразу нормируйте значения регрессоров и зависимой переменной.
  3. При нормировке вы можете передавать в MinMaxScaler сразу весь необходимый массив.
  4. Т.к. x\_train – двумерный массив, к нему не нужно применять reshape. К одномерному y\_train – нужно.
  5. Модели линейной регрессии будут возвращать массив в качестве значения coef\_.
  6. Вы не сможете без дополнительных ухищрений нарисовать графики, поэтому можете этого не делать.

## Требования к отчету

Загрузите свой файл в репозиторий, созданный в практическом задании №1 по пути: «Notebook\_For\_AI\_Main/2020 Осенний семестр/Дополнительное практическое задание/» и сделайте пул-реквест.

## Литература

* <https://habr.com/ru/post/514818/>
* <https://habr.com/ru/post/474602/>
* <http://statistica.ru/theory/osnovy-lineynoy-regressii/>
* <http://statistica.ru/theory/logisticheskaya-regressiya/>
* <https://habr.com/ru/post/485872/>
* [https://habr.com/ru/company/ods/blog/323890/#metod-maksimalnogo-pravdopodobiya](https://habr.com/ru/company/ods/blog/323890/)
* <https://dyakonov.org/2018/03/12/%D0%BB%D0%BE%D0%B3%D0%B8%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F-%D0%BE%D1%88%D0%B8%D0%B1%D0%BA%D0%B8/>