**Постановка задачи.**  
<https://www.kaggle.com/hugoncosta/price-of-flats-in-moscow>  
Данные о стоимости квартир в Москве в зависимости от параметров квартиры и ее расположения.

Структура:  
price - Цена квартиры в $1000  
totsp - Общая площадь квартиры, кв.м.  
livesp - Жилая площадь квартиры, кв.м.  
kitsp - Площадь кухни, кв.м.  
dist - Расстояние от центра в км.  
metrdist - Расстояние до метро в минутах  
walk - 1 – пешком от метро, 0 – на транспорте  
brick - 1 – кирпичный, монолит ж/б, 0 – другой  
floor - 1 – этаж кроме первого и последнего, 0 – иначе  
code - Число от 1 до 8, при помощи которого мы группируем наблюдения по подвыборкам:  
1. Наблюдения сгруппированы на севере, вокруг Калужско-Рижской линии метрополитена  
2. Север, вокруг Серпуховско-Тимирязевской линии метрополитена  
3. Северо-запад, вокруг Замоскворецкой линии метрополитена  
4. Северо-запад, вокруг Таганско-Краснопресненской линии метрополитена  
5. Юго-восток, вокруг Люблинской линии метрополитена  
6. Юго-восток, вокруг Таганско-Краснопресненской линии метрополитена  
7. Восток, вокруг Калиниской линии метрополитена  
8. Восток, вокруг Арбатско-Покровской линии метрополитена

Задача: вычисление значения параметра price на основании остальных параметров.  
Опциональный вариант: вычисление значения параметра code на основании остальных параметров.

**Подход к обработке данных.**  
price – был выброс в районе (500-750) и не очень равномерное распределение . Были применены метод numpy.clip() , логарифм и MinMaxScaler() fit\_transform(data).flatten() для приведения значений к инвервалу (0,1)

totsp – выброс в районе 190 и не очень равномерное распределние . Были применены метод numpy.clip() , квадратный корень и MinMaxScaler() fit\_transform(data).flatten() для приведения значений к инвервалу (0,1)

livesp – выброс (90-100) и неравномерное распределение . Были применены метод numpy.clip() , логарифм и MinMaxScaler() fit\_transform(data).flatten()

kitsp , dist - были применены только MinMaxScaler() fit\_transform(data).flatten() для приведения значений к инвервалу (0,1)

mertdist – выброс в районе 20 и и не очень равномерное распределение . Были применены метод numpy.clip() , квадратный корень и MinMaxScaler() fit\_transform(data).flatten() для приведения значений к инвервалу (0,1)

walk , brick , floor – принимают только 0 и 1 поэтому были оставлены без изменений

code – было применено унитарное кодирование методом pandas.get\_dummies() т.к. стоблец код это не содержит в себе количественную меру , а атрибутивное свойство в данном случае местоположение квартиры .

**Выбранные базовые параметры нейронной сети и алгоритма обучения.**функция активации - ReLU

Количество нейронов входного слоя – 16

Первый скрытый слой – 50

Второй скрытый слой – 30

Третий скрытый слой - 18

Выходной слой – 1

Количество эпох – 100

Батчей – 32

Выборка 0.2

**Ссылка на репозиторий с кодом проекта с базовыми параметрами.**  
<https://github.com/broker99/cursach>

**Описание проведенных экспериментов.**Для начала эмпирическим путем было выявлено оптимальное количество скрытых слоев , количества нейронов в слоях , количество батчей и эпох при выборке 0.1 . После чего я стал увеличивать выборку на 0.2 и 0.3 и наблюдать за изменениями .

С увеличением выборки графики становятся более плавными и сходятся лучше , модель почти не изменилась. С уменьшением выборки графики чуть-чуть изменился в лучшую сторону , сходятся лучше , модель стала не изменилась . Отношение тренировочной выборки к тестовой 0.2 оказалось самым оптимальным .

**Итоговый вывод.**Выборка данных нужна для обучения модели на части данных. Выборка очень важный этап в машинном обучении т.к. если она будет слишком маленькая , то мы можем плохо обучить нашу модель , если большая то наоборот переобучить . В подавляющем большинстве тренировочная выборка меньше тестовой .