МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Искусственные нейронные сети»

Тема: Регрессионная модель изменения цен на дома в Бостоне

Студентка гр. 7382 _____ Головина Е.С.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2020

Цель работы.

Реализовать предсказание медианной цены на дома в пригороде Бостона в середине 1970-х по таким данным, как уровень преступности, ставка местного имущественного налога и т. д.

Задачи.

- Ознакомиться с задачей регрессии
- Изучить отличие задачи регрессии от задачи классификации
- Создать модель
- Настроить параметры обучения
- Обучить и оценить модель
- Ознакомиться с перекрестной проверкой

Требования.

- Объяснить различия задач классификации и регрессии
- Изучить влияние кол-ва эпох на результат обучения модели
- Выявить точку переобучения
- Применить перекрестную проверку по K блокам при различных K
- Построить графики ошибки и точности во время обучения для моделей, а также усредненные графики по всем моделям

Ход работы.

1. Знакомство с задачей регрессии

Регрессия относится к одному из основных классов машинного обучения, т.н. *обучение с учителем*. При обучении с учителем на вход подается тренировочный набор данных, на которых нейронная сеть учится и по окончании обучения способна по новым данным (похожим на тренировочные данные) получать определенный результат.

Задача регрессии заключается в том, чтобы предсказать числовое значение по входным данным. Порождаемая функция: $f: \mathbb{R}^n \to \mathbb{R}$. Различия от задачи классификации в формате выхода.

2. Создание модели

Набор данных присутствует в составе Keras. Загрузка происходит следующим образом.

```
from tensorflow.keras.datasets import boston_housing
(train_data, train_targets), (test_data, test_targets) =
boston_housing.load_data()
```

Данные – 404 обучающих и 102 контрольных образца, каждый с 13 числовыми признаками. Из-за того, что каждый признак во входных данных (например, уровень преступности) имеет свой масштаб – необходимо нормализовать данные. В противном случае сеть сможет адаптироваться к разнородным данным, но это усложнит обучение.

Процесс нормализации: для каждого признака во входных данных (столбца в матрице входных данных) из каждого значения вычитается среднее по этому признаку, и разность делится на стандартное отклонение, в результате признак центрируется по нулевому значению и имеет стандартное отклонение, равное единице.

Такое преобразование можно сделать с помощью Numpy.

```
mean = train_data.mean(axis = 0)
train_data -= mean
std = train_data.std(axis=0)
train_data /= std
test_data -=mean
test_data /= std
```

Для создания модели ИНС напишем функцию build_model():

```
def build_model():
    model = Sequential()
    model.add(Dense(64, activation='relu',
input_shape=(train_data.shape[1],)))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(1))
    model.compile(optimizer='rmsprop', loss='mse', metrics=['mae'])
    return model
```

Таким образом, построена нейронная сеть с тремя слоями. Сеть заканчивается одномерным слоем, не имеющим функции активации (это линейный слой). Применение функции активации могло бы ограничить диапазон выходных значений, а с линейным последним слоем, сеть способна предсказывать значения из любого диапазона.

Сеть компилируется с функцией потерь `mse` – mean squared error (среднеквадратичная ошибка), вычисляющей квадрат разности между предсказанными и целевыми значениями. Эта функция широко используется в задачах регрессии.

Параметр `mae` — mean absolute error (средняя абсолютная ошибка). Это абсолютное значение разности между предсказанными и целевыми значениями. Например, значение `mae`, равное 0.5, в этой задаче означает, что в среднем прогнозы отклоняются на 500 долларов США.

3. Настройка параметров, обучение и оценка модели

В связи с тем, что набор данных для обучения небольшой используем перекрестную проверку по K блокам (K-fold cross-validation). Суть ее заключается в разделении доступных данных на K блоков (обычно K = 4 или 5), создании K идентичных моделей и обучении каждой на K—1 блоках с оценкой по оставшимся блокам. По полученным K оценкам вычисляется среднее значение, которое принимается как оценка модели. В коде такая проверка реализуется следующим образом.

```
k = 4
num_val_samples = len(train_data) // k
num_epochs = 100
all_scores = []

for i in range(k):
    print('processing fold #', i)
    val_data = train_data[i * num_val_samples: (i + 1) * num_val_samples]
    val_targets = train_targets[i * num_val_samples: (i + 1) *
num_val_samples]
    partial_train_data = np.concatenate([train_data[:i *
num_val_samples], train_data[(i + 1) * num_val_samples:]], axis=0)
    partial_train_targets = np.concatenate([train_targets[:i *
num_val_samples], train_targets[(i + 1) * num_val_samples:]], axis=0)
    model = build_model()
    model.fit(partial_train_data, partial_train_targets,
epochs=num_epochs, batch_size=1, verbose=0)
    val_mse, val_mae = model.evaluate(val_data, val_targets, verbose=0)
    all_scores.append(val_mae)

print(np.mean(all_scores))
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

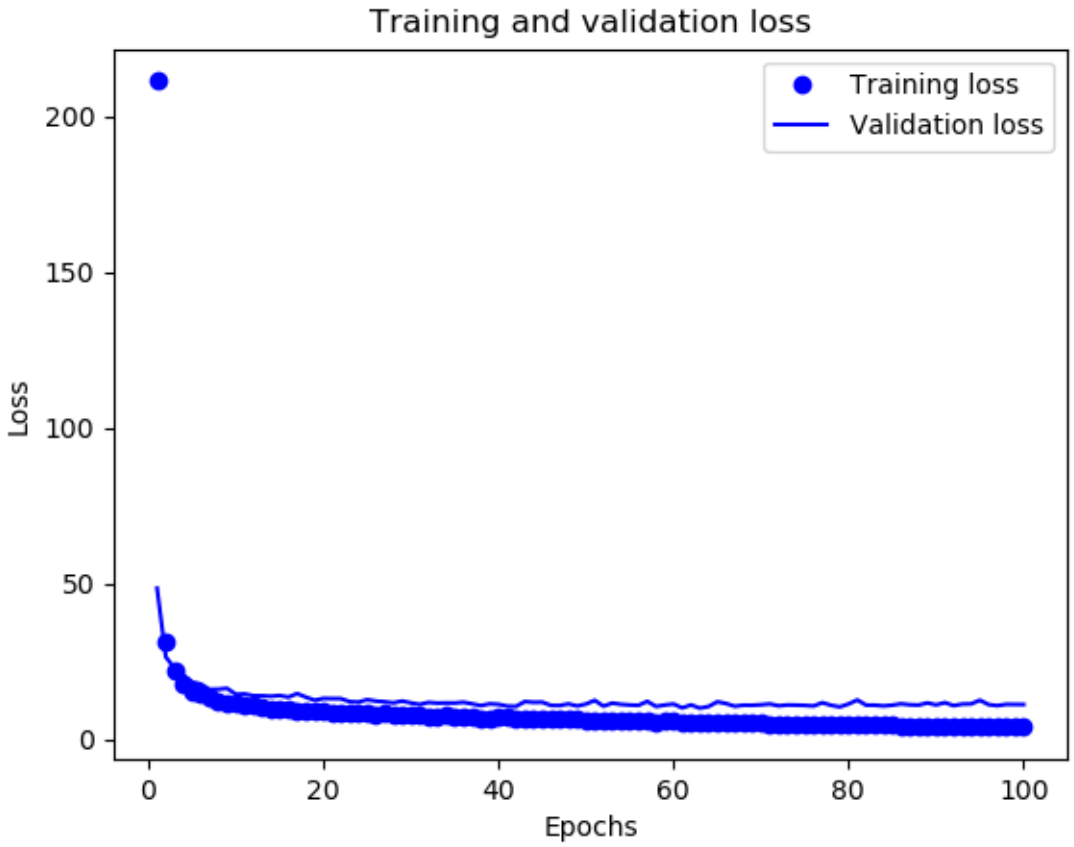
Полученные результаты:

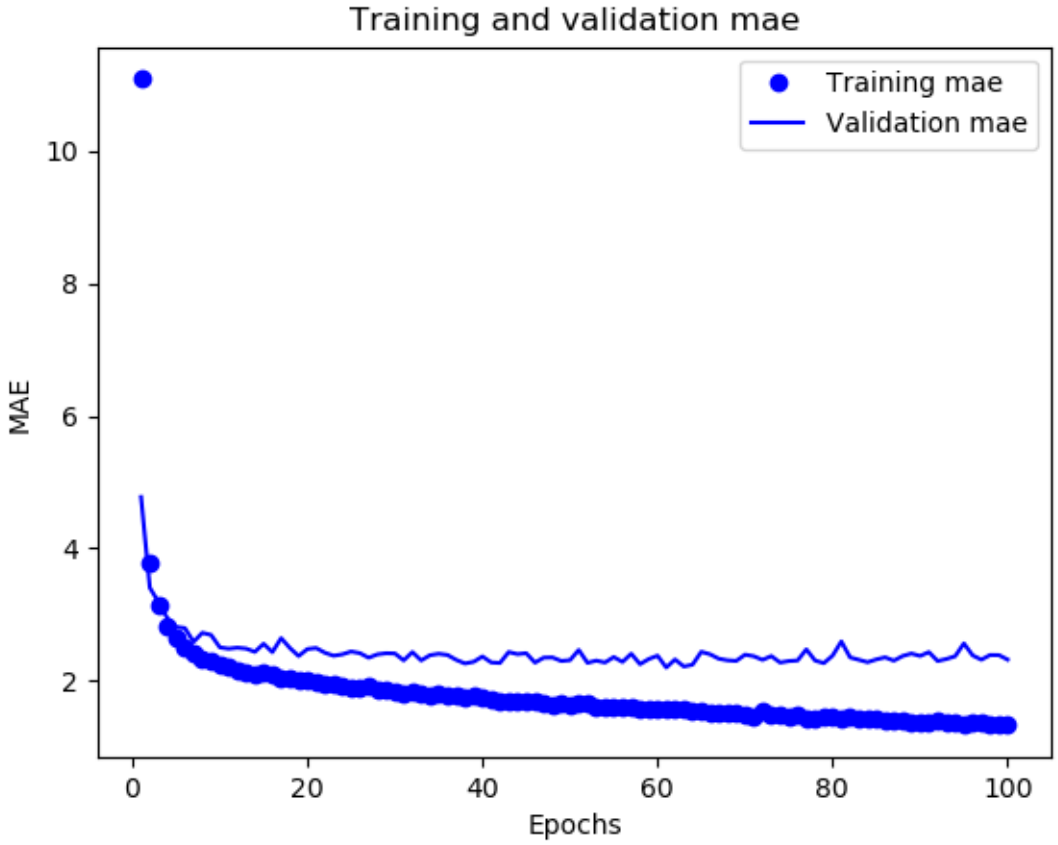
```
2.483508288860321
```

Это означает, что средняя ошибка составляет около 2400 долларов, что довольно много для цен из диапазона от 10 000 до 50 000 долларов.

4. Выбор количества эпох

Для того, чтобы решить в какую сторону двигаться: увеличивать или уменьшать число эпох выведем средние графики ошибки и точности по всем моделям.

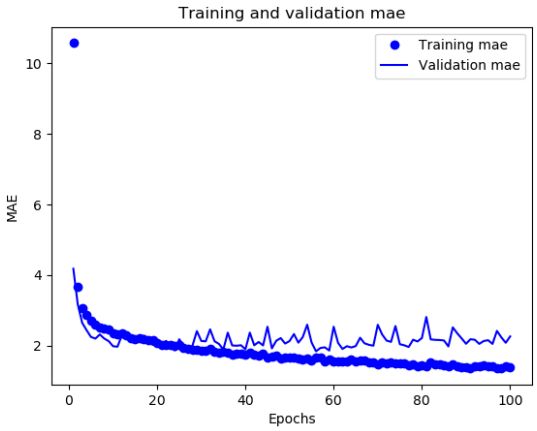

Training and validation loss
Training loss
Validation loss
Loss
200
150
100
50
0
0
20
40
60
80
100
Epochs

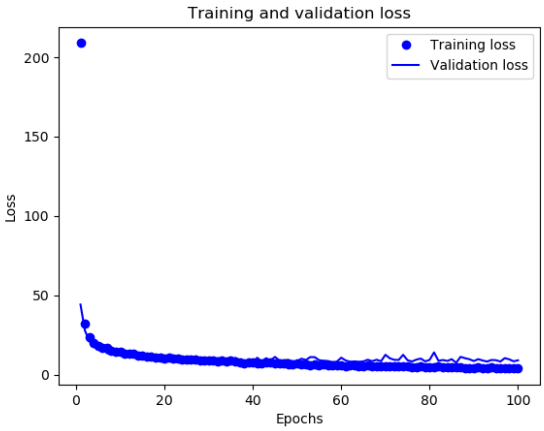

Training and validation mae
Training mae
Validation mae
MAE
10
8
6
4
2
0
20
40
60
80
100
Epochs

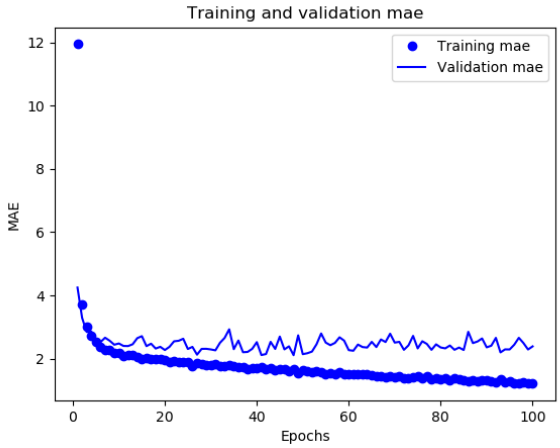
По графику ошибок можно заметить, что примерно на 40 эпохе происходит переобучение, т.к. между потерями при обучении и оценке появляется явный разрыв. Значит нужно двигаться в сторону уменьшения эпох.

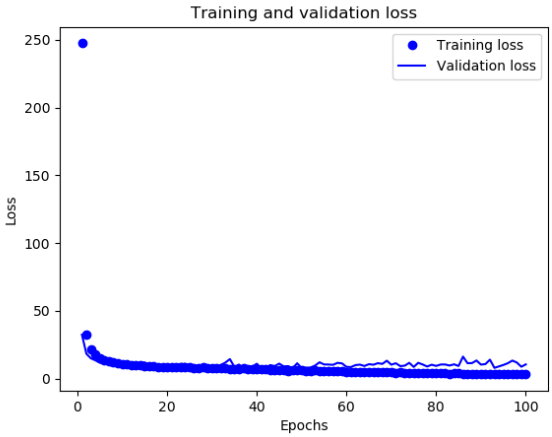
Проверив значения для эпох от 10 до 100 с шагом 10, выявили, что наименьшее значение в результате дают 50 и 60 эпоха.

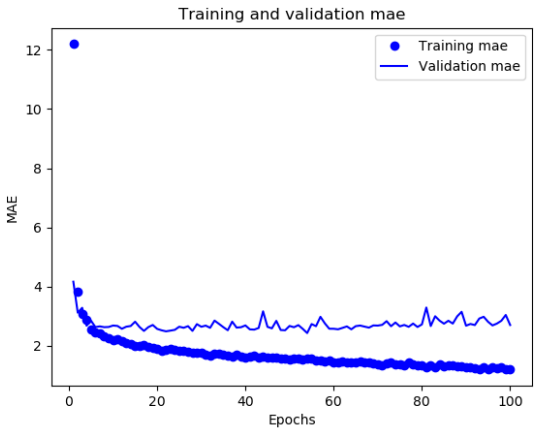
Проверив значения между ними – наименьшего значение достигли на 54 эпохах. Ставим 54 эпохи и смотрим графики результатов.

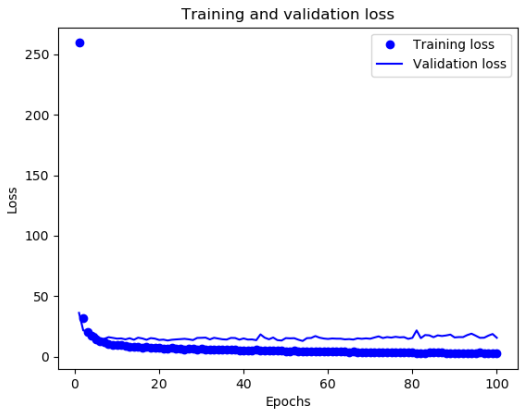


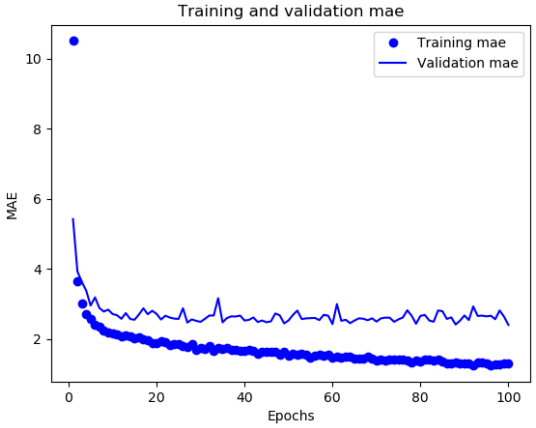


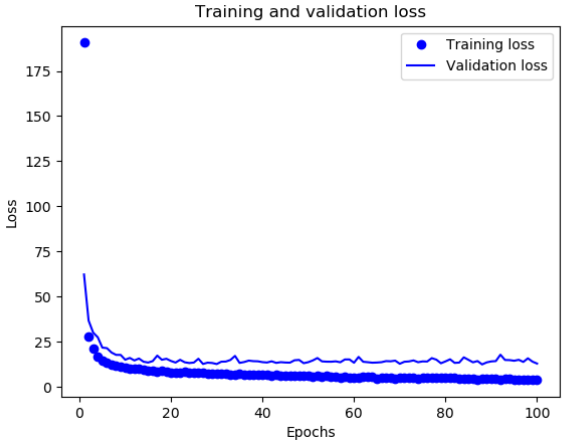




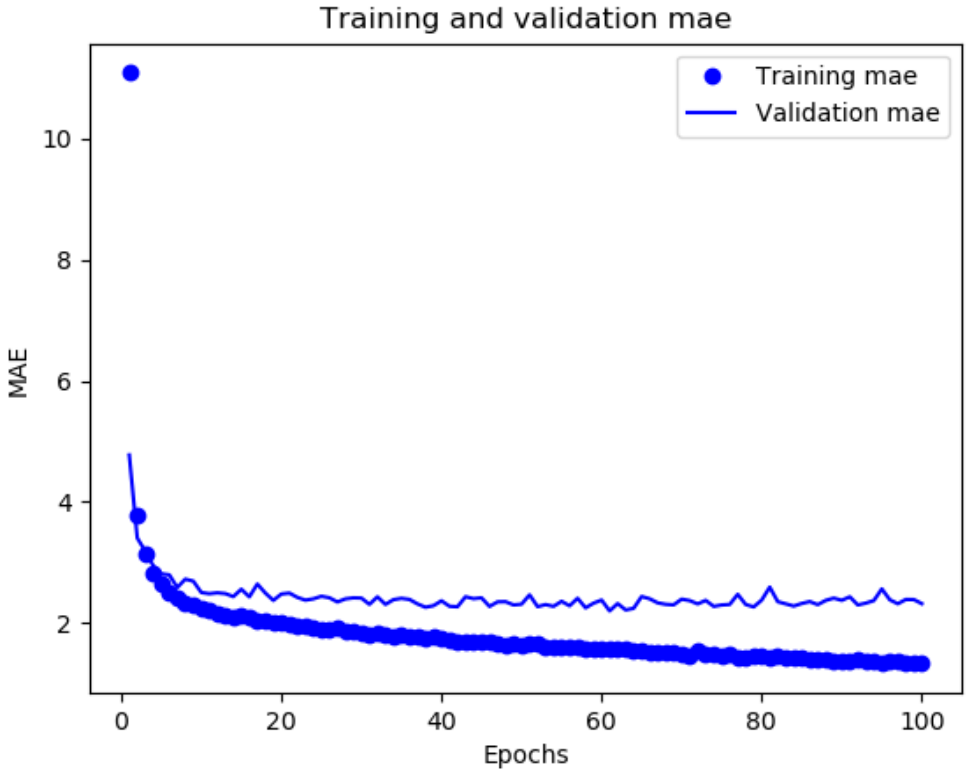


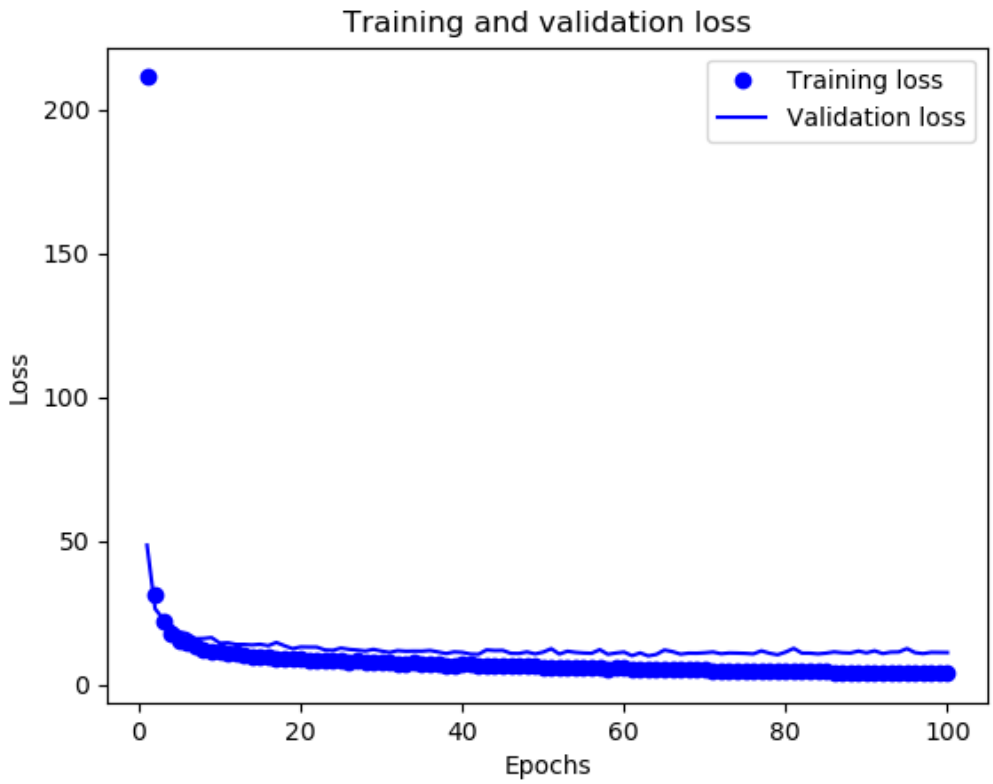






Графики средних значений:





Вывод.

В ходе выполнения работы реализовали решение задачи регрессии: предсказание медианной цены на дома в пригороде Бостона в середине 1970-х по набору разных, влияющих на цену данных. Для этого создали искусственную нейронную сеть с помощью библиотеки keras в python, протестировали получившуюся модель, затем, проведя ряд экспериментов, выбрали наилучшую конфигурацию модели.