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

ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине «Искусственные нейронные сети»

Тема: «Регрессионная модель изменения цен на дома в Бостоне»

Студент гр. 7383 _____ Левкович Д.В.

Преподаватель _____ Жукова Н.А.

Санкт-Петербург
2020

Цели.

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 модели
- Ознакомиться с перекрестной проверкой

Выполнение работы.

В задаче классификации определяется принадлежность к одному из нескольких классов, то есть набор значений ограничен. В задаче регрессии определяется значение какой-либо характеристики объекта, значение которой может быть любое число.

1) Была создана и обучена модель искусственной нейронной сети в Keras, код представлен в приложении А.

2) При исследовании разных архитектур и обучение при различных параметрах обучения ИНС было изменено:

- Количество блоков: 2, 4, 6

Изначально была рассмотрена модель с 2 блоками и со 150 эпохами.

Были выведены графики по всем 2 блокам. Графики представлены на рис. 1-3.

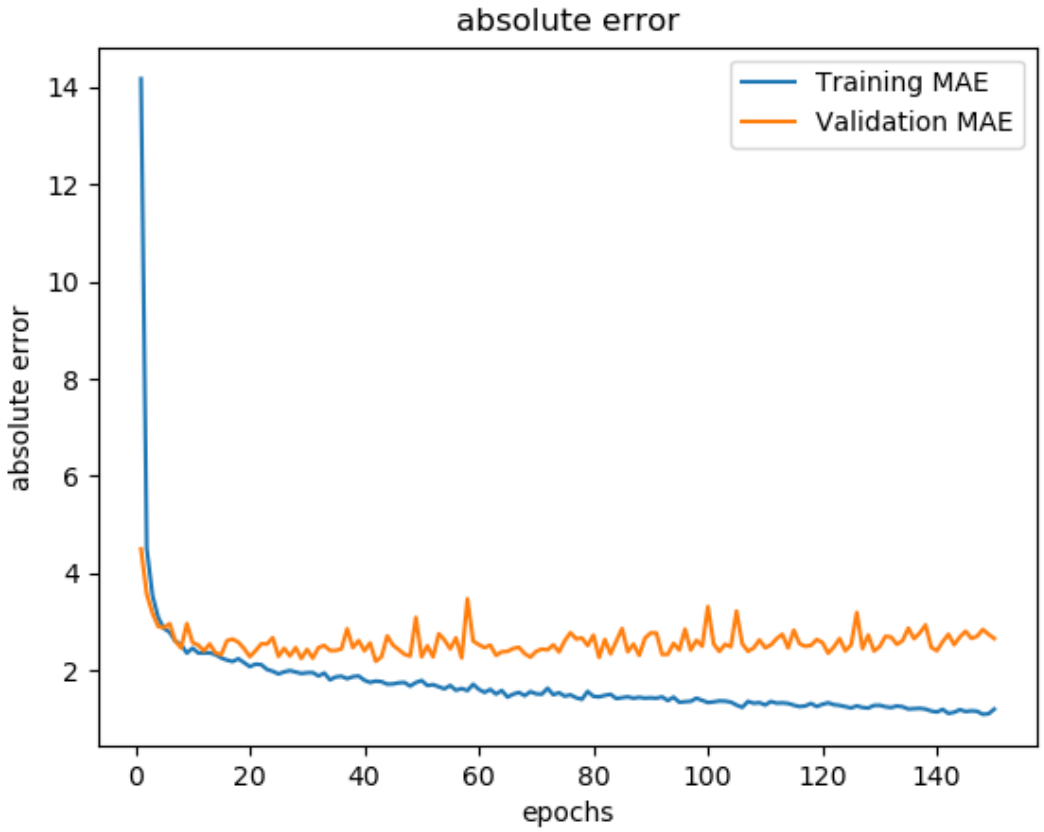


Рисунок 1 – График оценки MAE для 1 блока

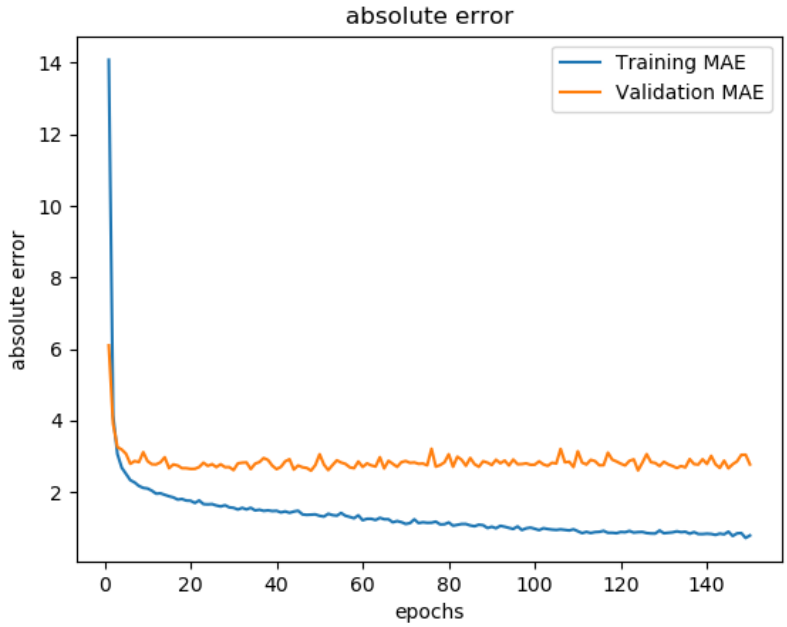


Рисунок 2 – График оценки MAE для 2 блока

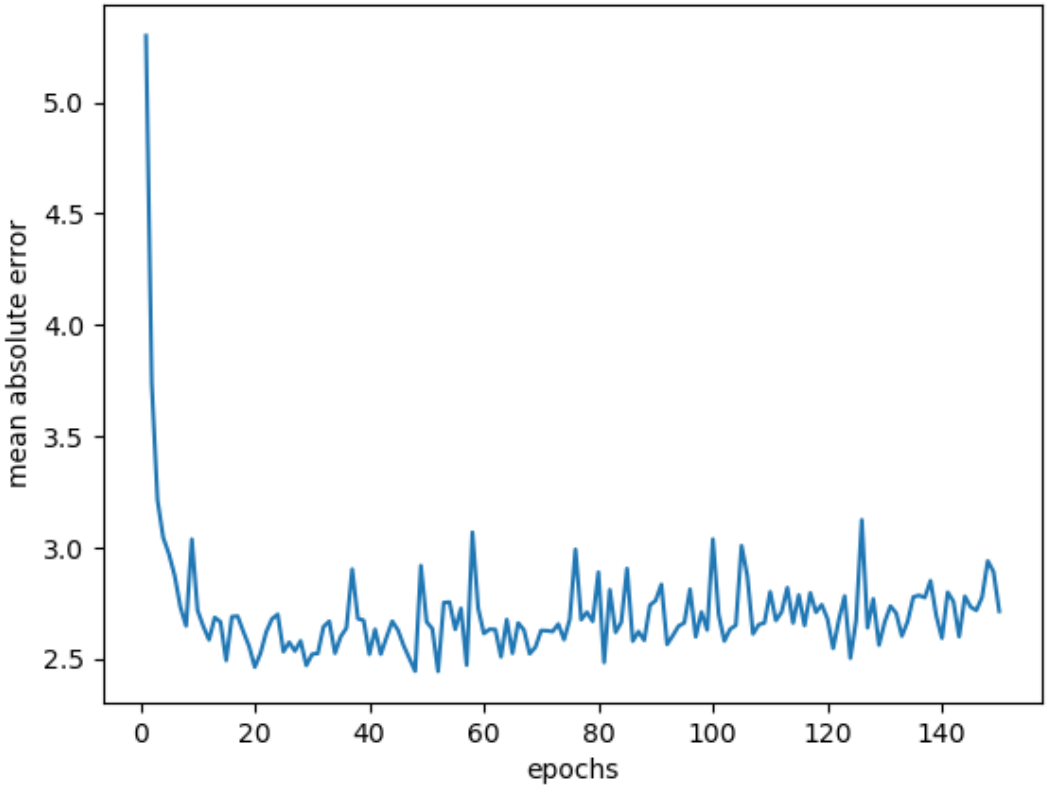


Рисунок 3 – График значения MAE по 2 блокам

2. Как видно из графика, MAE не убывает после 40-50 эпохи.

3. При уменьшении ошибки на тестовых данных, ошибки на проверочных тоже не убывают (возрастают), что говорит о переобучении. Значит точка переобучения находится на 40-50 эпохах.

Далее был рассмотрен случай с 4-мя блоками.

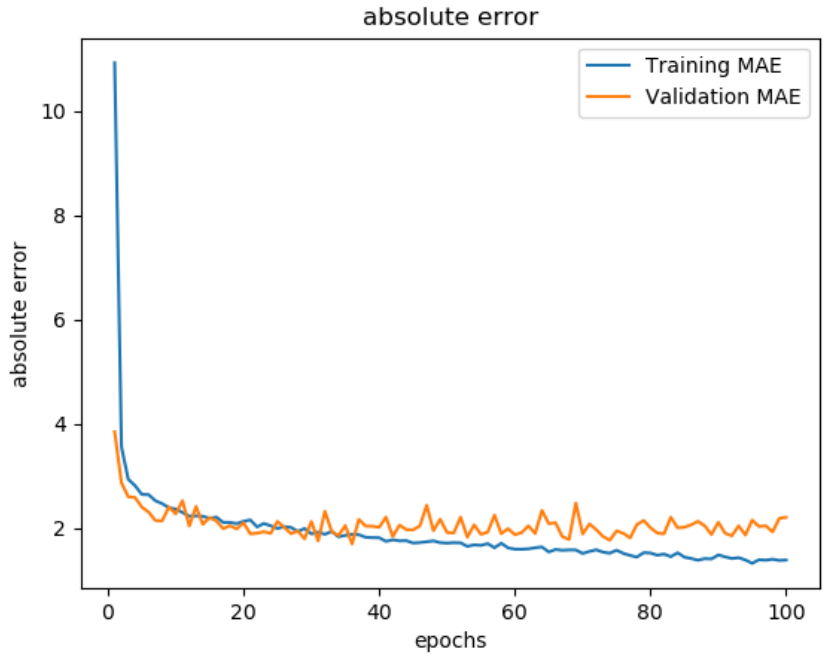


Рисунок 4 – График оценки MAE для 1 блока

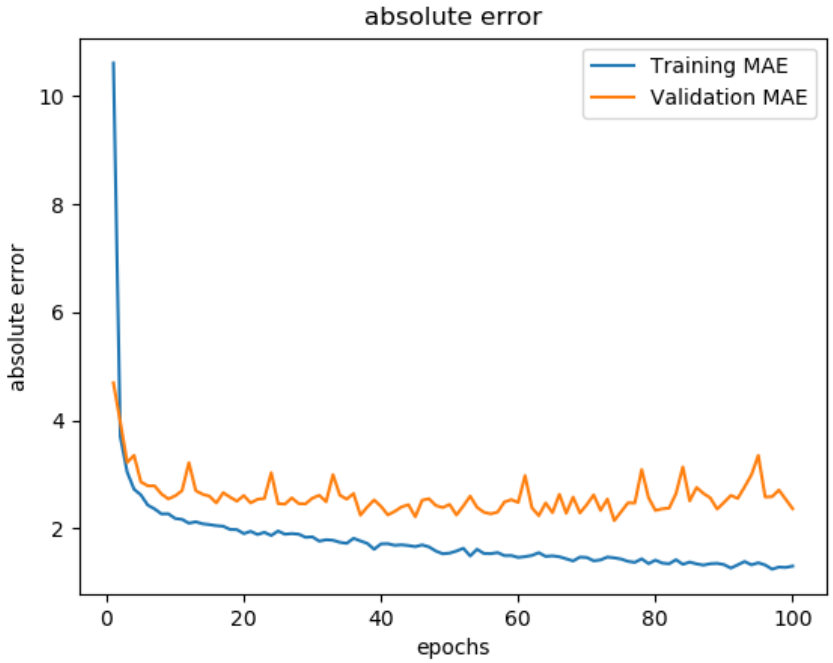


Рисунок 5 – График оценки MAE для 2 блока

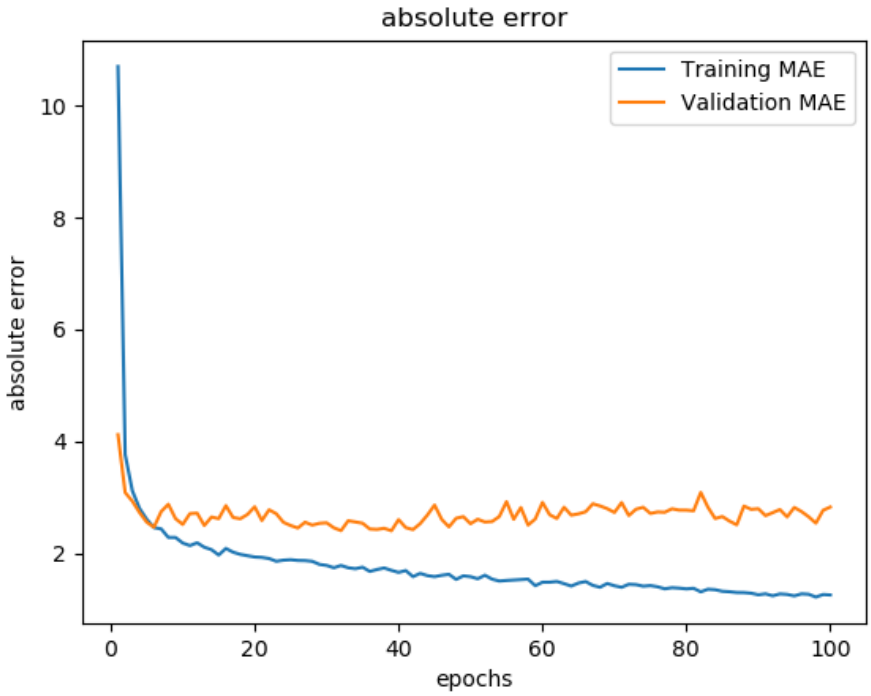


Рисунок 6 – График оценки MAE для 3 блока

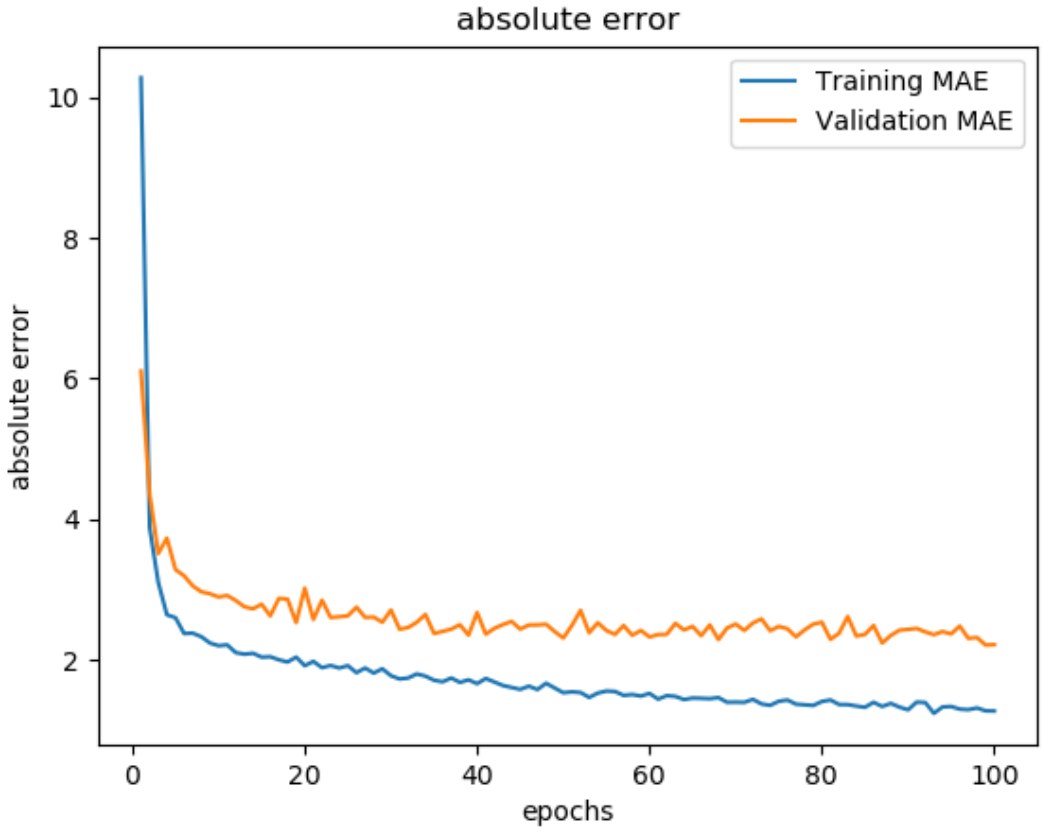


Рисунок 7 – График оценки MAE для 4 блока

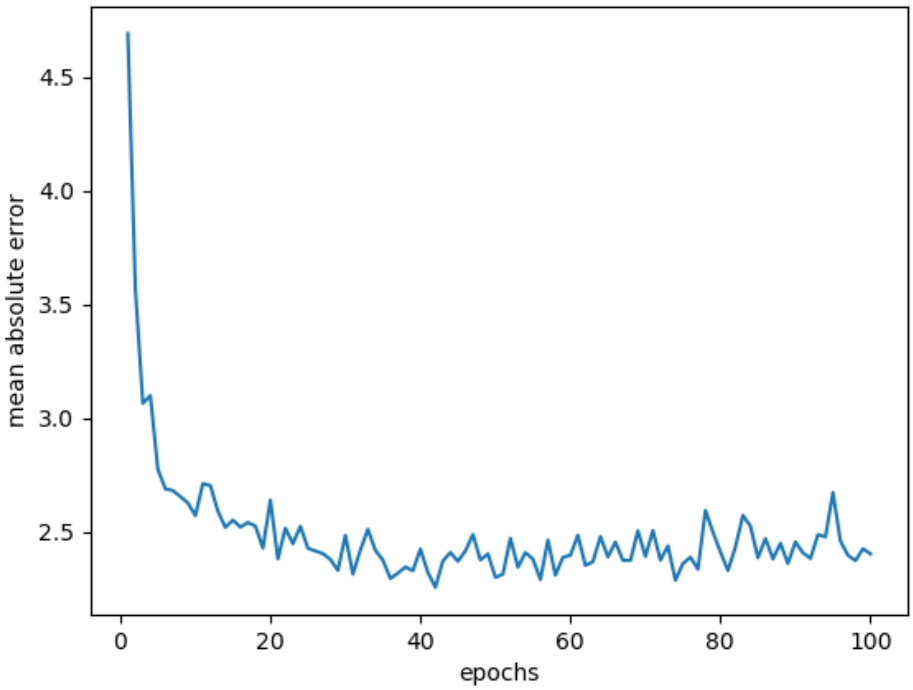


Рисунок 8 – График значения MAE по 4 блокам

Для 6 блоков:

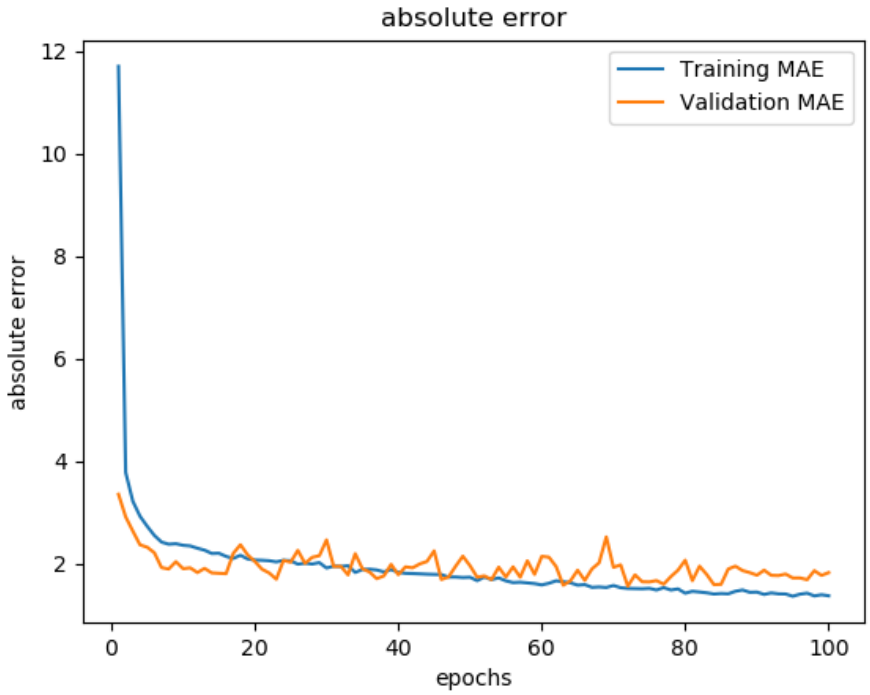


Рисунок 9 – График оценки MAE для 1 блока

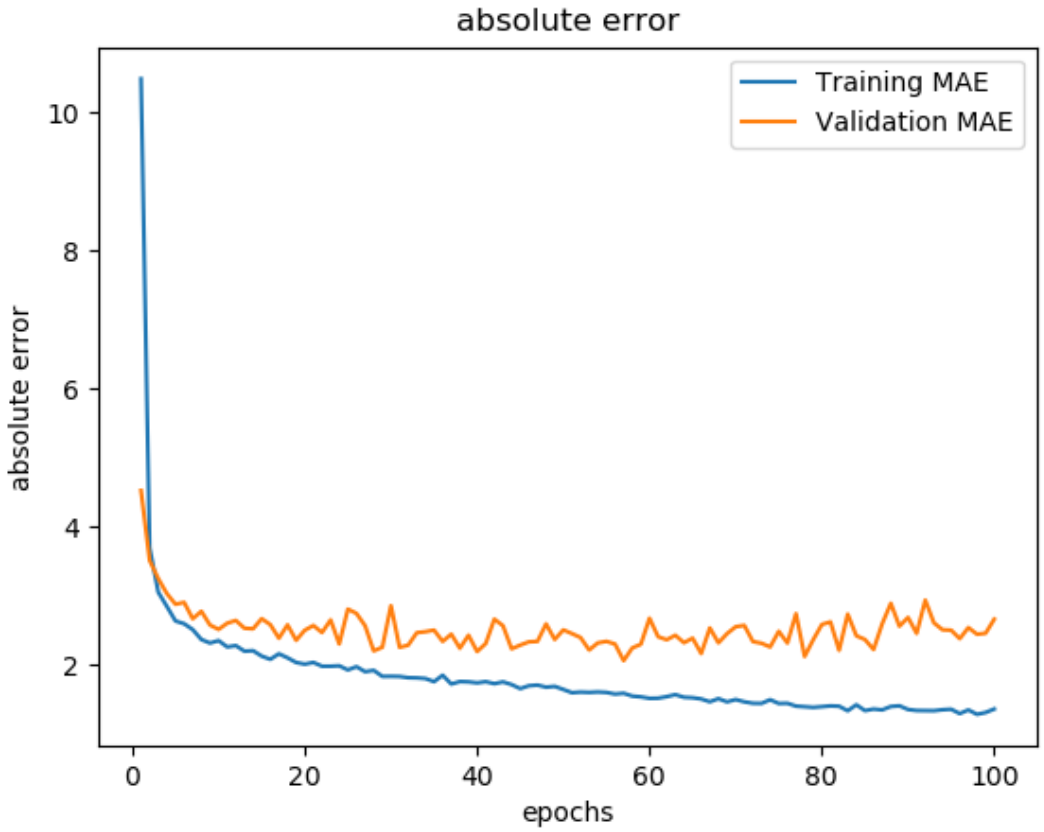


Рисунок 10 – График оценки MAE для 2 блока

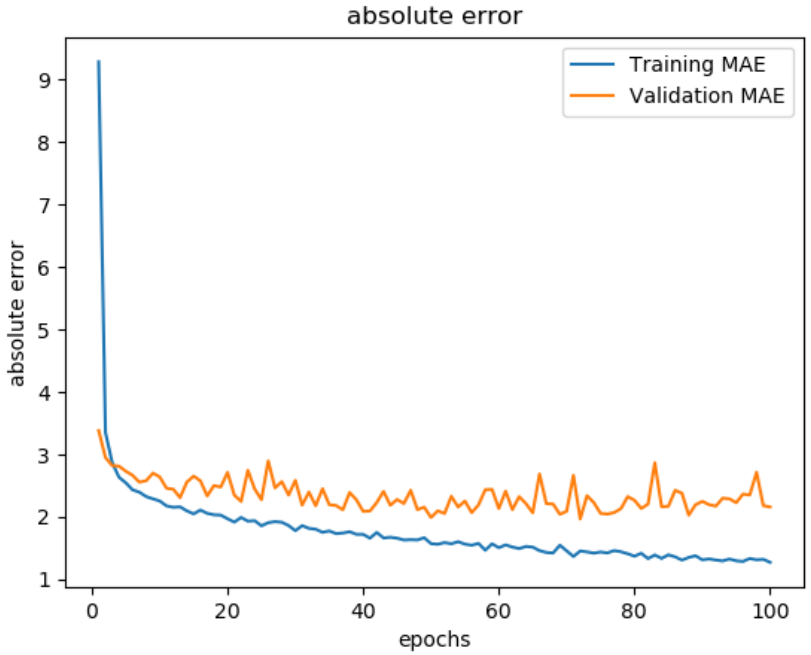


Рисунок 11 – График оценки MAE для 3 блока

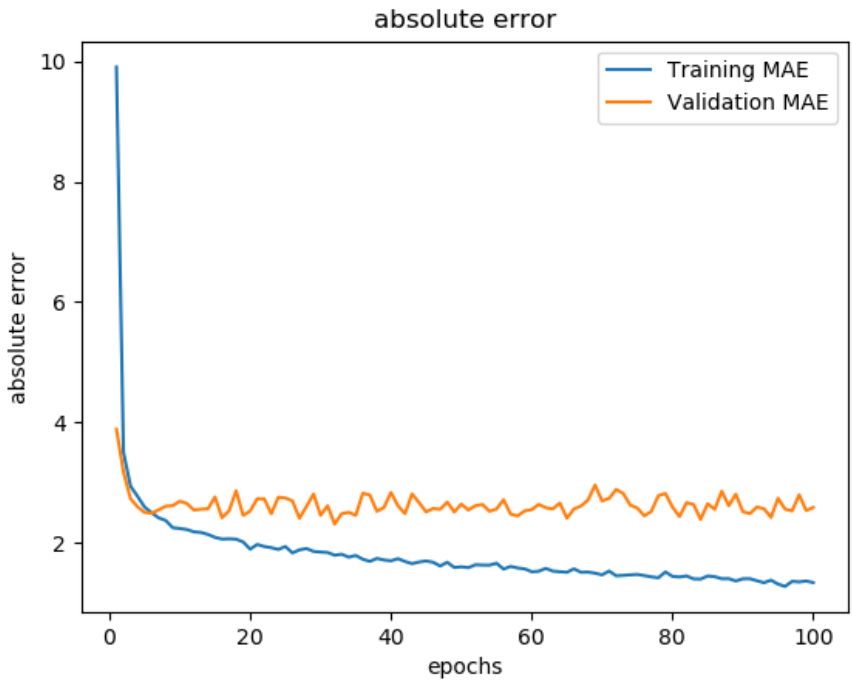


Рисунок 12 – График оценки MAE для 4 блока

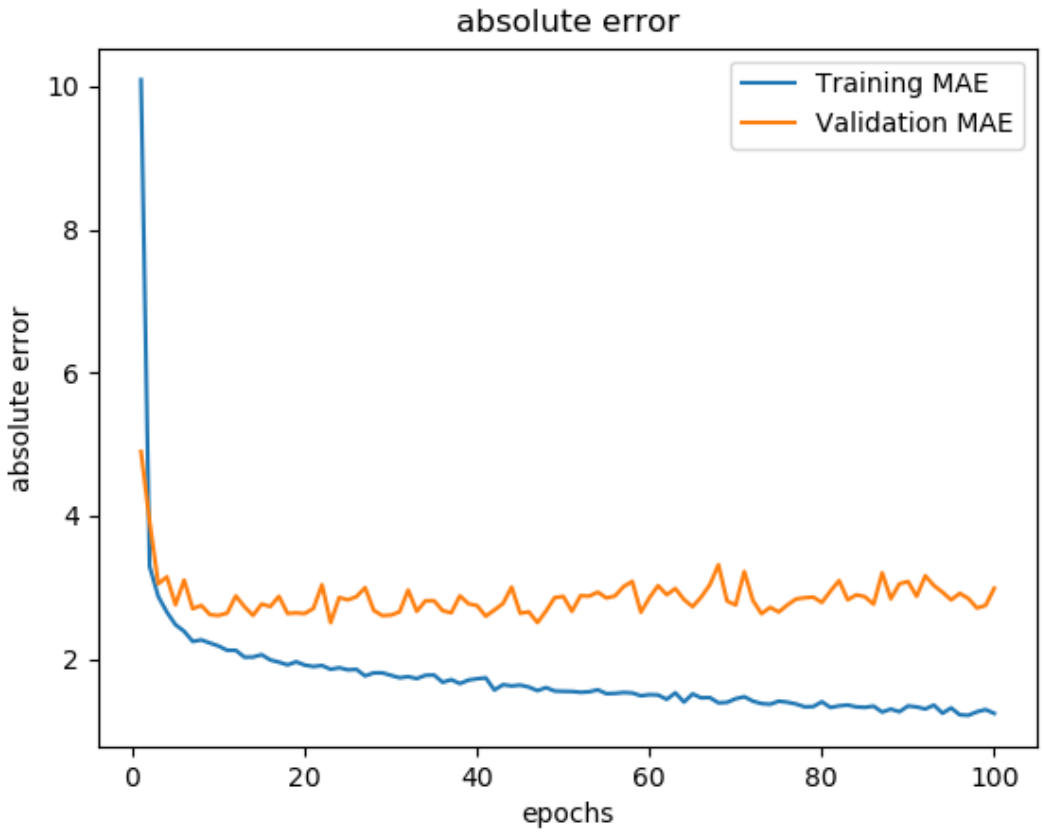


Рисунок 13 – График оценки MAE для 5 блока

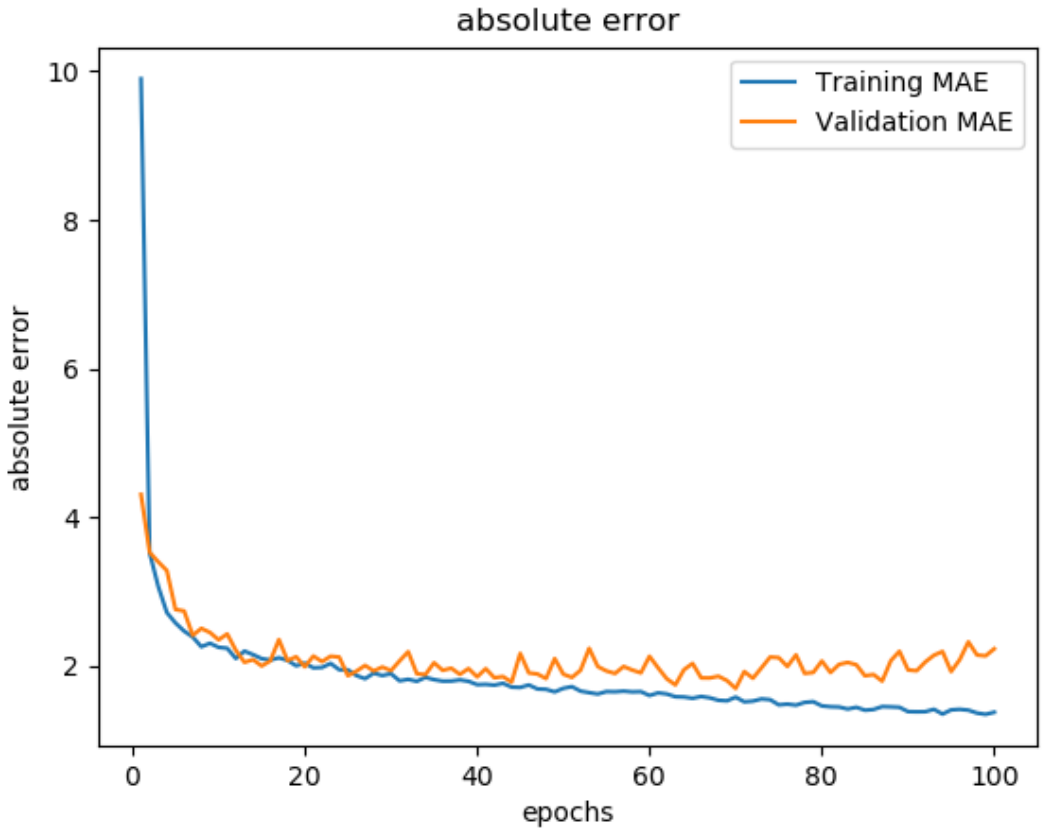
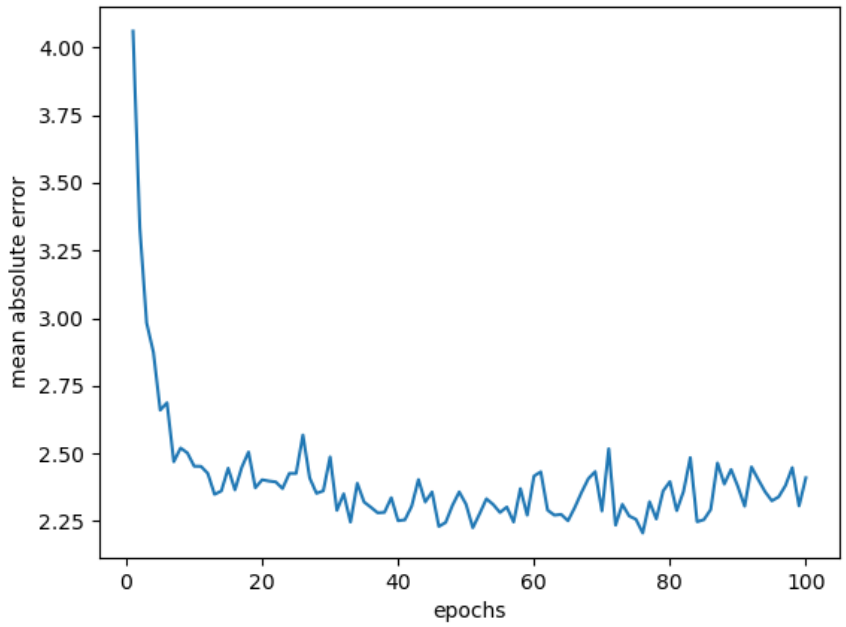


Рисунок 14 – График оценки MAE для 6 блока



Вывод.

В ходе выполнения данной работы была изучена задача регрессии с помощью библиотеки Keras и ее отличие от задачи классификации. Была изучена перекрестная проверка.

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А: ИСХОДНЫЙ КОД

```
from keras.layers import Dense
from keras.models import Sequential
from keras.datasets import boston_housing
import numpy as np
import matplotlib.pyplot as plt


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


(train_data, train_targets), (test_data, test_targets) =
boston_housing.load_data()

mean = train_data.mean(axis=0)
std = train_data.std(axis=0)

train_data -= mean
train_data /= std

test_data -= mean
test_data /= std

k = 6
num_val_samples = len(train_data) // k
num_epochs = 100
mae_histories = []
for i in range(k):
    print(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_target = np.concatenate([train_targets[: i *
num_val_samples],
```

```
                                                train_targets[(i +
1) * num_val_samples:]], axis=0)
    model = build_model()
    history = model.fit(partial_train_data,
partial_train_target, epochs=num_epochs, batch_size=1,
                        validation_data=(val_data,
val_targets))
    mae = history.history['mae']
    v_mae = history.history['val_mae']
    x = range(1, num_epochs + 1)
    mae_histories.append(v_mae)
    plt.figure(i + 1)
    plt.plot(x, mae, label='Training MAE')
    plt.plot(x, v_mae, label='Validation MAE')
    plt.title('absolute error')
    plt.ylabel('absolute error')
    plt.xlabel('epochs')
    plt.legend()

average_mae_history = [np.mean([x[i] for x in mae_histories])
for i in range(num_epochs)]
plt.figure(0)
plt.plot(range(1, num_epochs + 1), average_mae_history)
plt.xlabel('epochs')
plt.ylabel("mean absolute error")
```