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

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Искусственные нейронные сети»

Тема: Регрессионная модель изменения цен на дома в Бостоне

Студентка гр. 8383 ___ Аверина О.С.

Преподаватель ___ Жангиров Т.Р.

Санкт-Петербург

2021

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.

Данный набор содержит относительно немного образцов данных: всего 506, разбитых на 404 обучающих и 102 контрольных образца. И каждый признак во входных данных (например, уровень преступности) имеет свой масштаб. Например, некоторые признаки являются пропорциями и имеют значения между 0 и 1, другие — между 1 и 12 и т. д.

Постановка 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.

Были загружены данные для обучения из библиотеки Keras:

```
import numpy as np
from tensorflow.keras.layers  import  Dense
from tensorflow.keras.models import Sequential
from  tensorflow.keras.utils  import  to_categorical
from tensorflow.keras.datasets import boston_housing

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

Затем проведена нормализация данных:

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

Была определена функция создания модели сети 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
```

Выходной слой сети не имеет функции активации – для данной задачи регрессии нет смысла ограничивать диапазон выходных значений. В качестве функции потерь используется $MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$, где все разности между наблюдаемыми значениями и значениями, предсказанными изучаемой регрессионной моделью возводятся в квадрат, суммируются, сумма делится на общее число ошибок.

В качестве метрики используется mae — mean absolute error (средняя абсолютная ошибка). Это абсолютное значение разности между предсказанными и целевыми значениями.

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}.$$

Так как набор данных небольшой, будет применена перекрестная проверка 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.

```
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))
```

Обучена модель из четырех подмоделей на 100 эпохах. Графики точности и потерь представлены на рис. 1-5

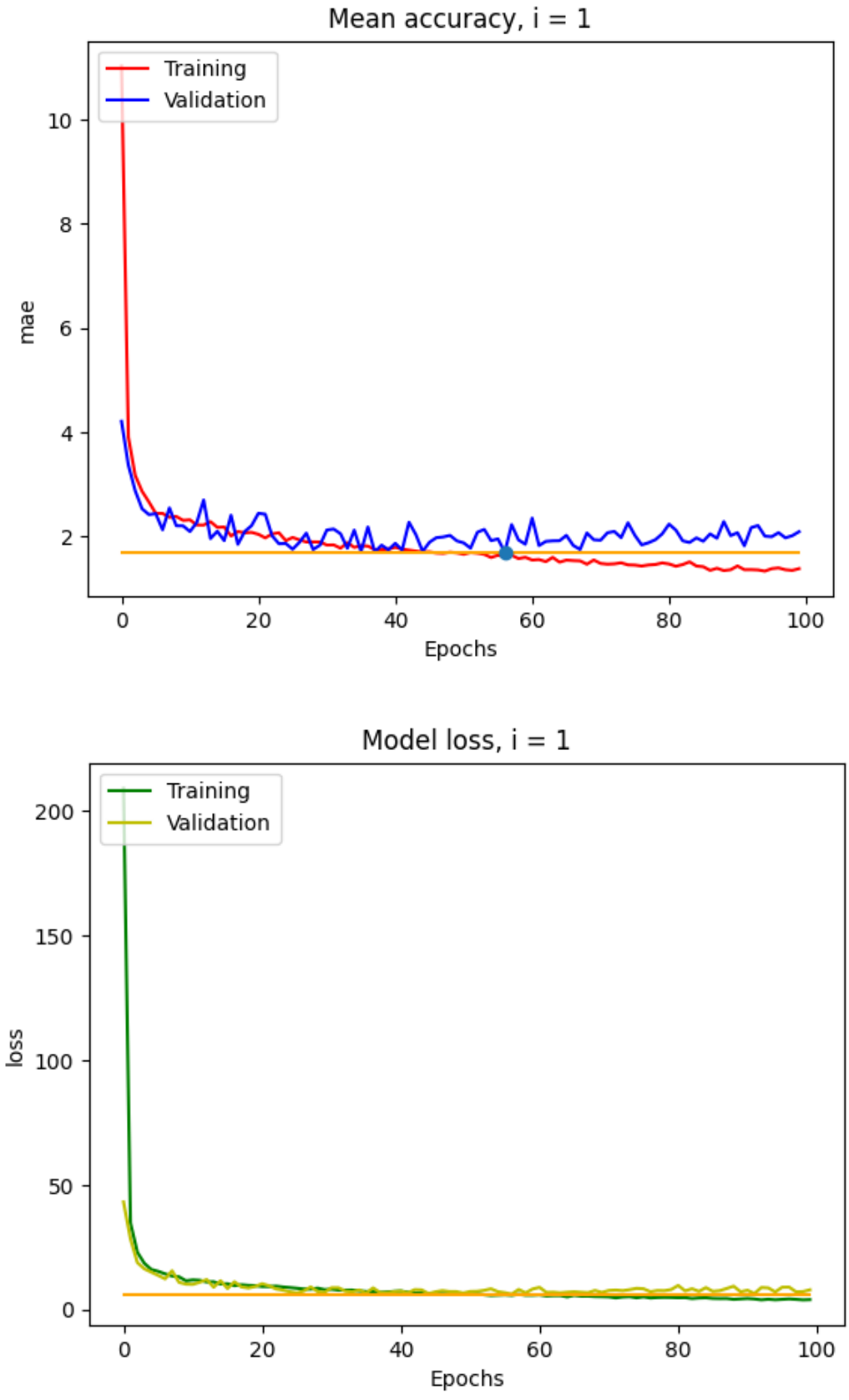
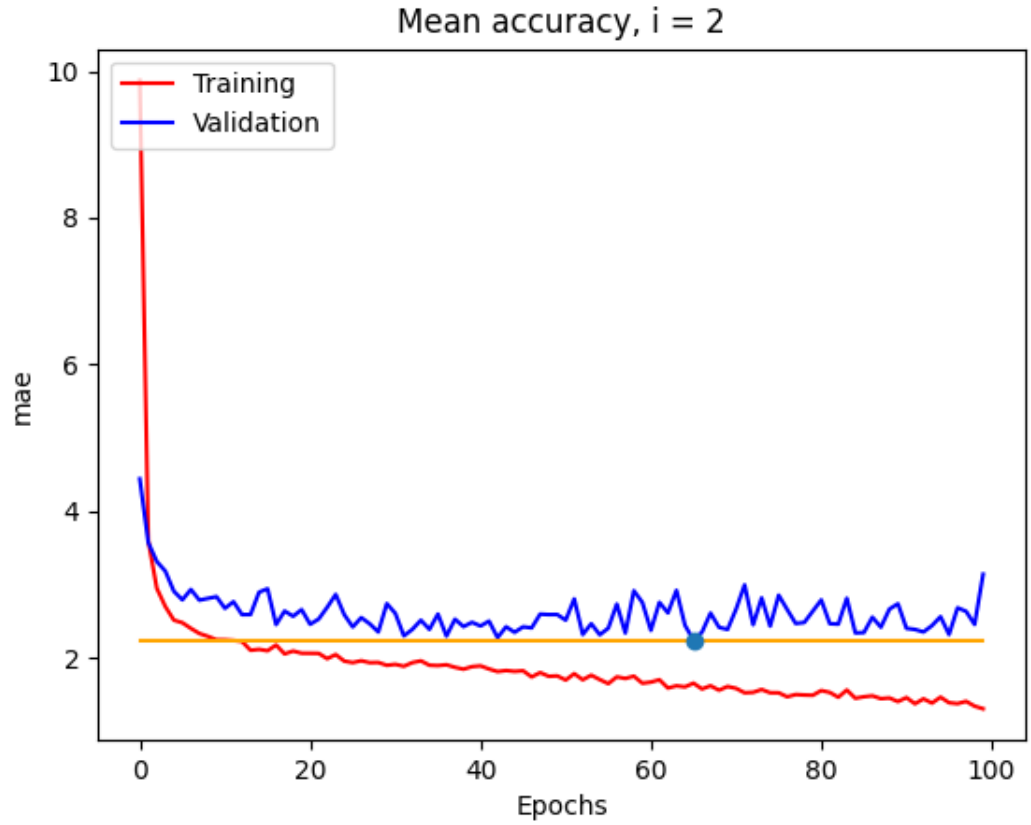


Рисунок 1. Первая модель, первая подмодель



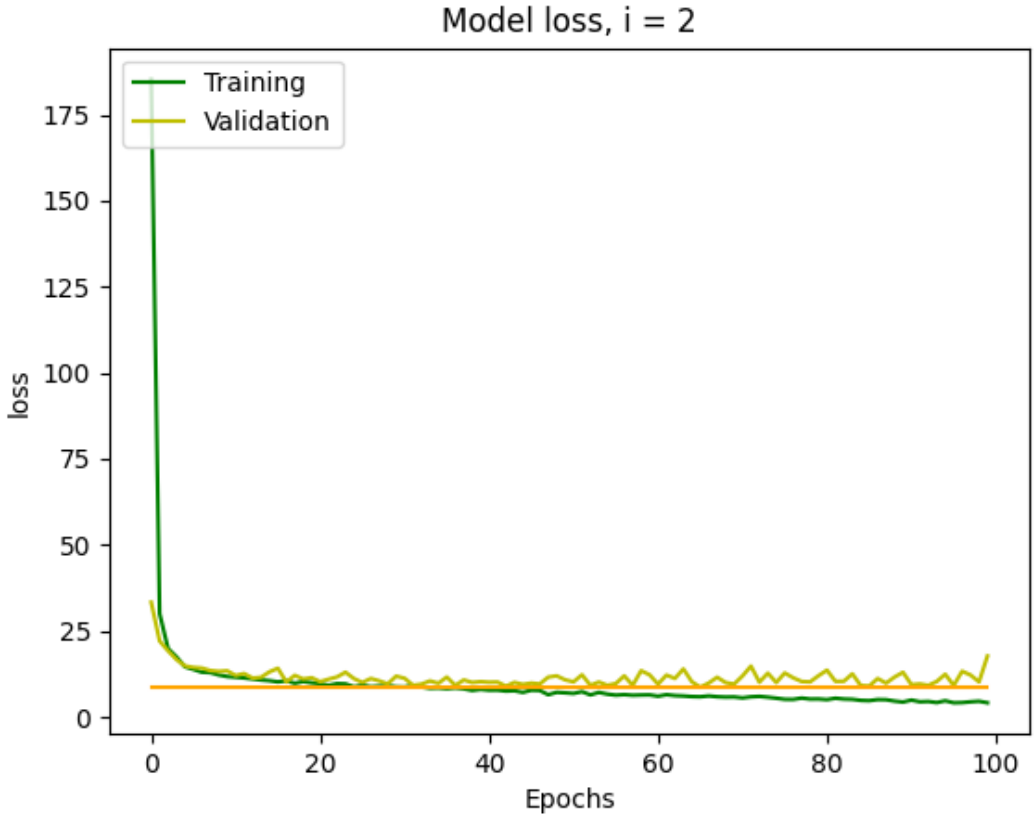
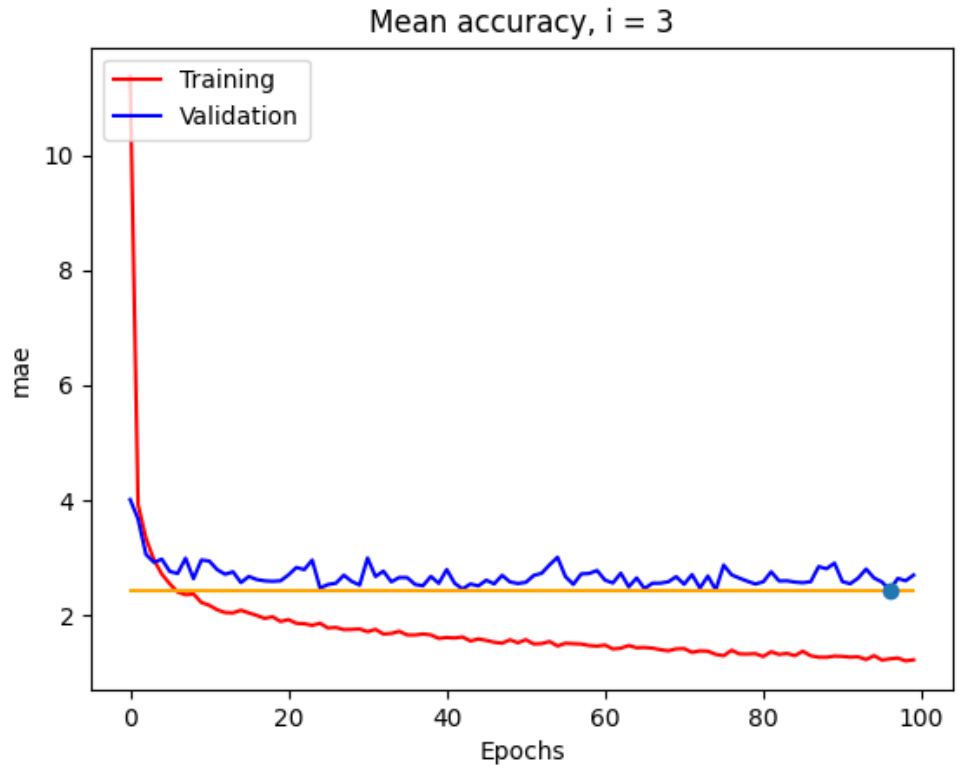


Рисунок 2. Первая модель, вторая подмодель



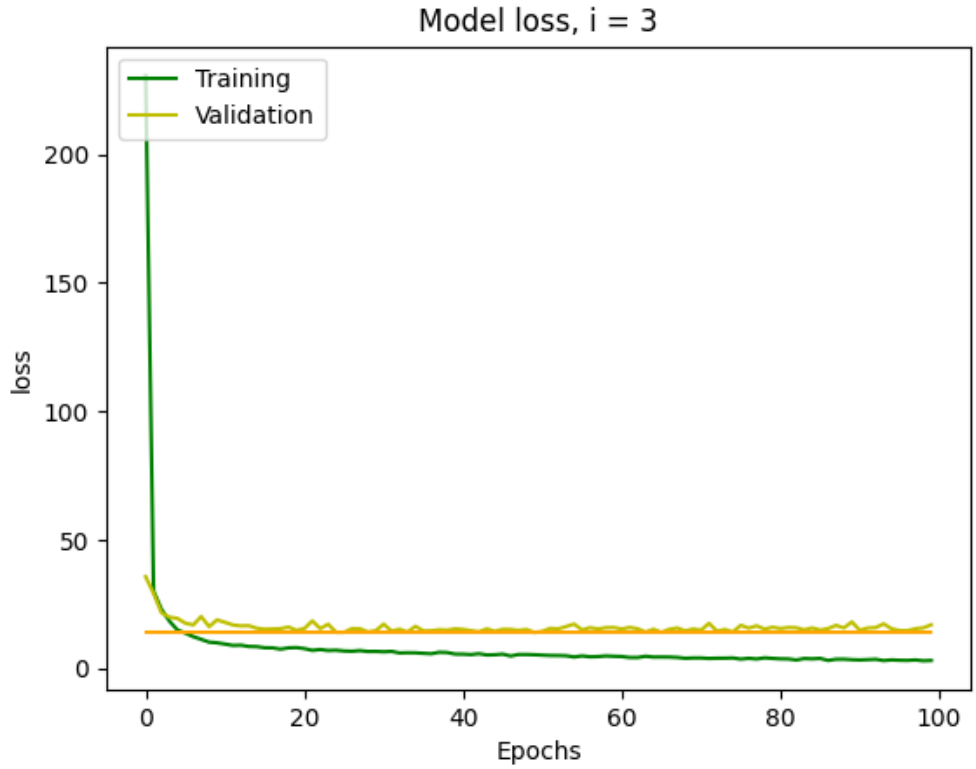
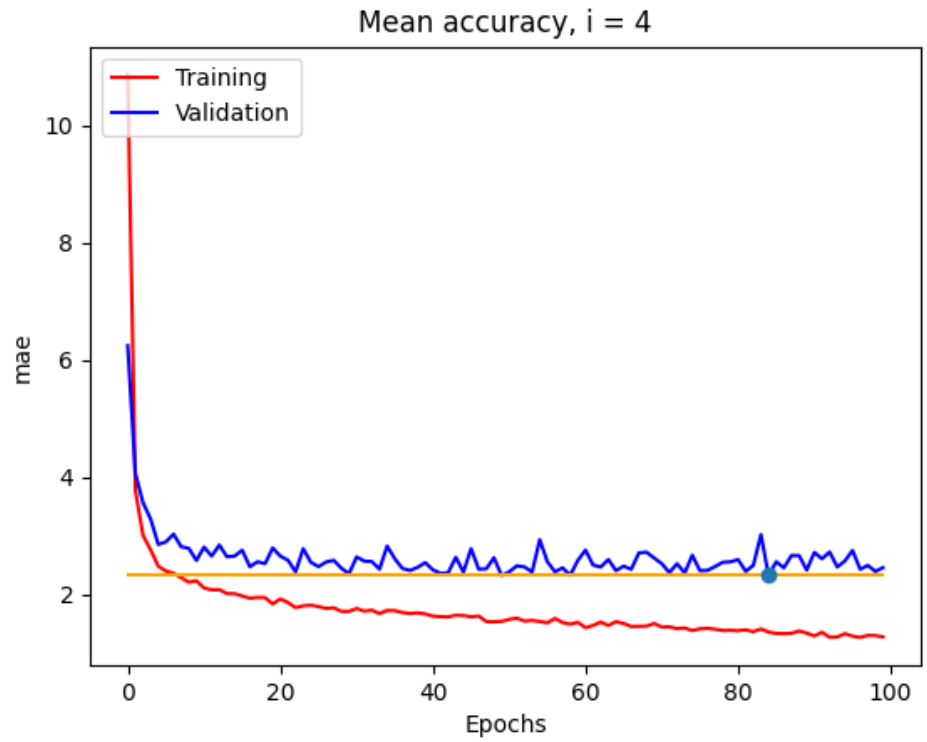


Рисунок 3. Первая модель, третья подмодель



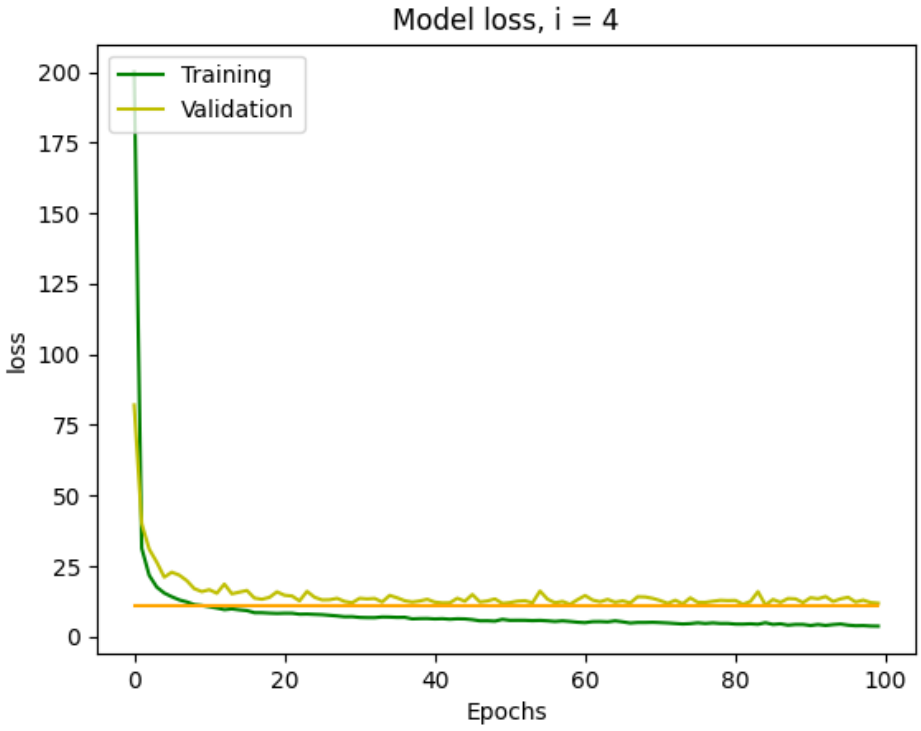
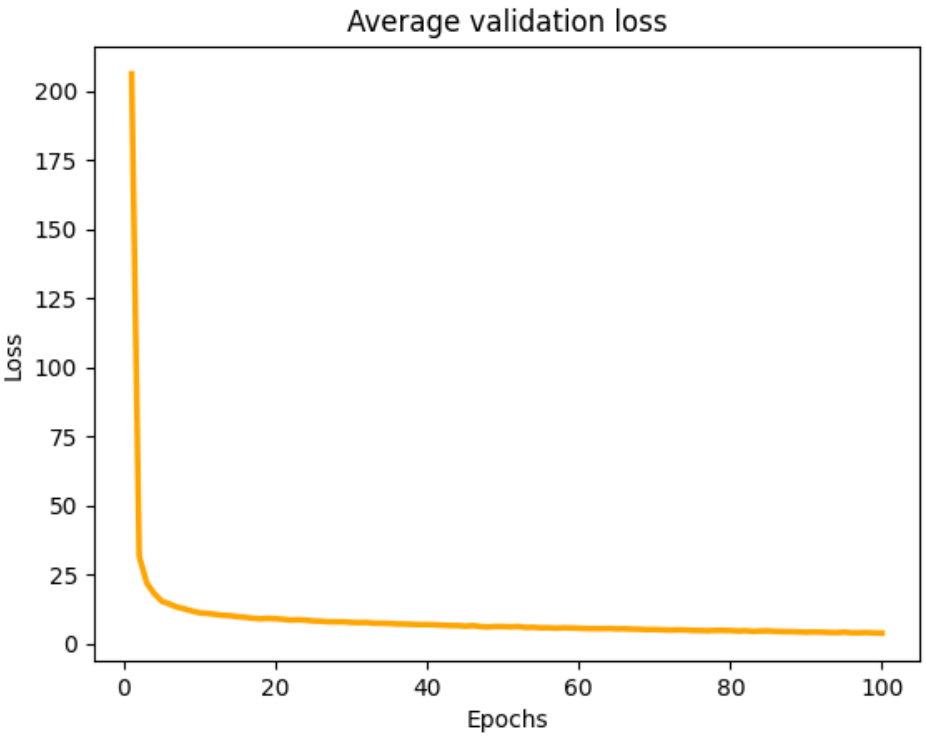


Рисунок 4. Первая модель, четвертая подмодель



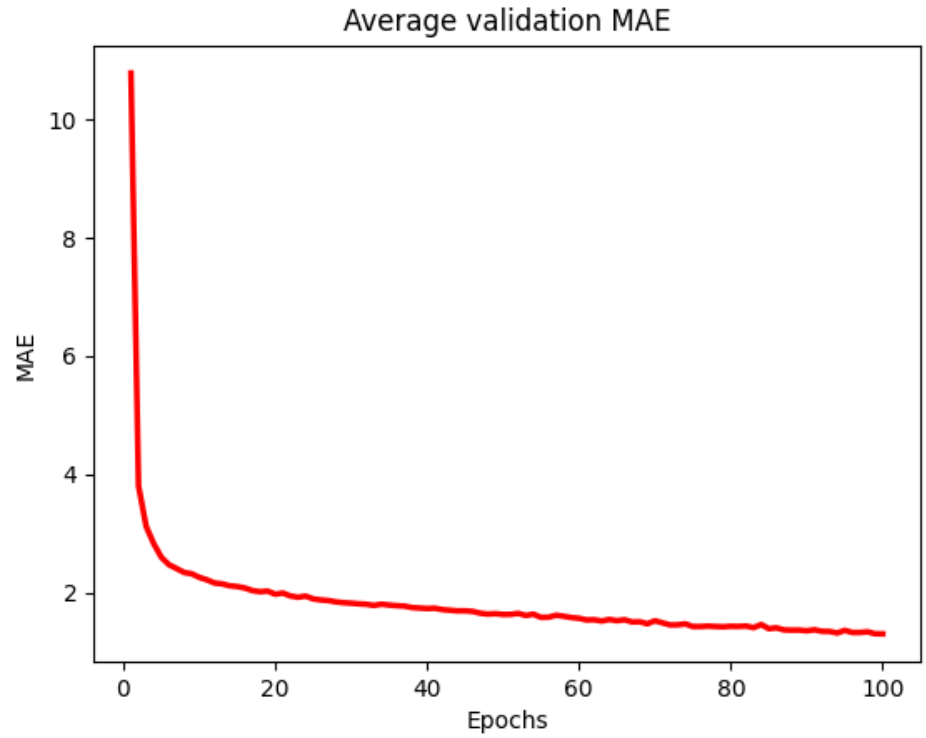
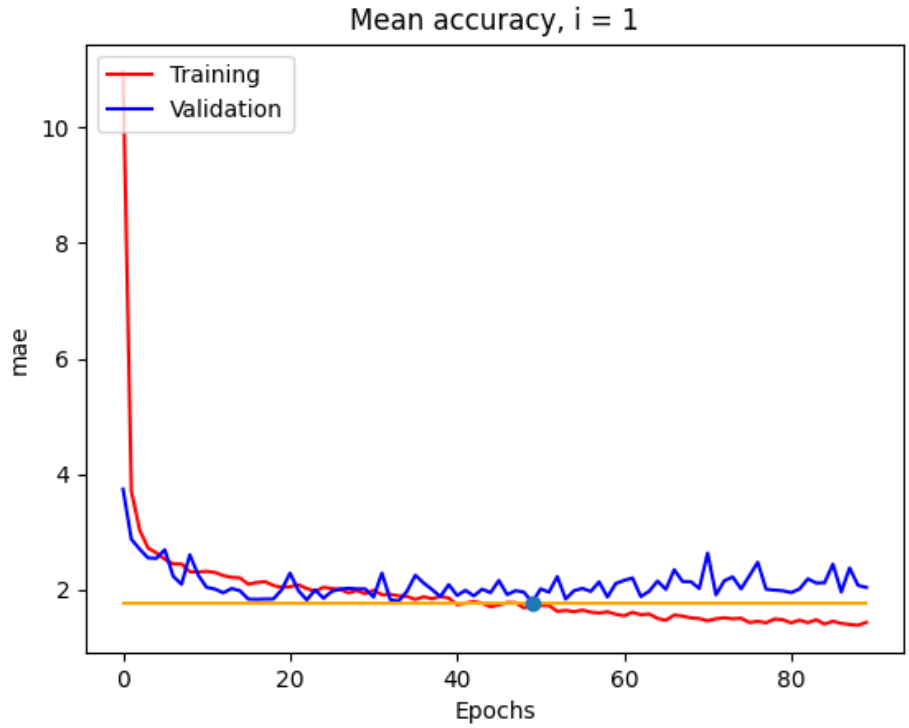


Рисунок 5. Показатели первой модели

Получена оценка модели 1.820, то есть результат модели отличается от истинного на 1820 долларов. Показатели сети перестают улучшаться в среднем после 80

эпох, можно предположить, что точка переобучения находится примерно там же, поэтому в следующей модели число эпох будет уменьшено.

Проведем второе исследование, изменив число эпох на 80.



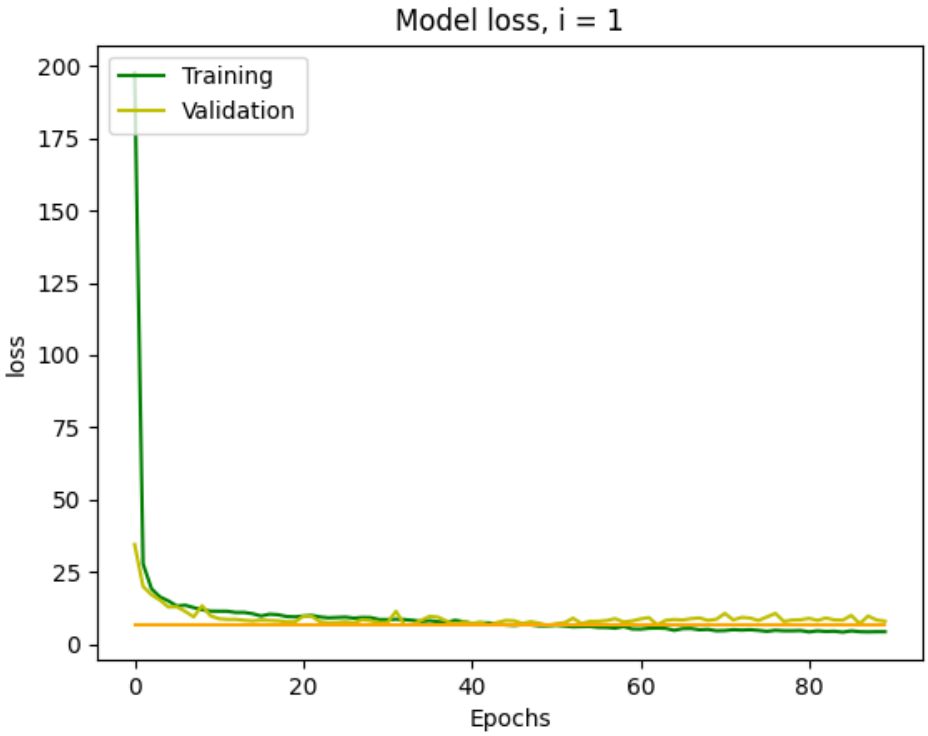
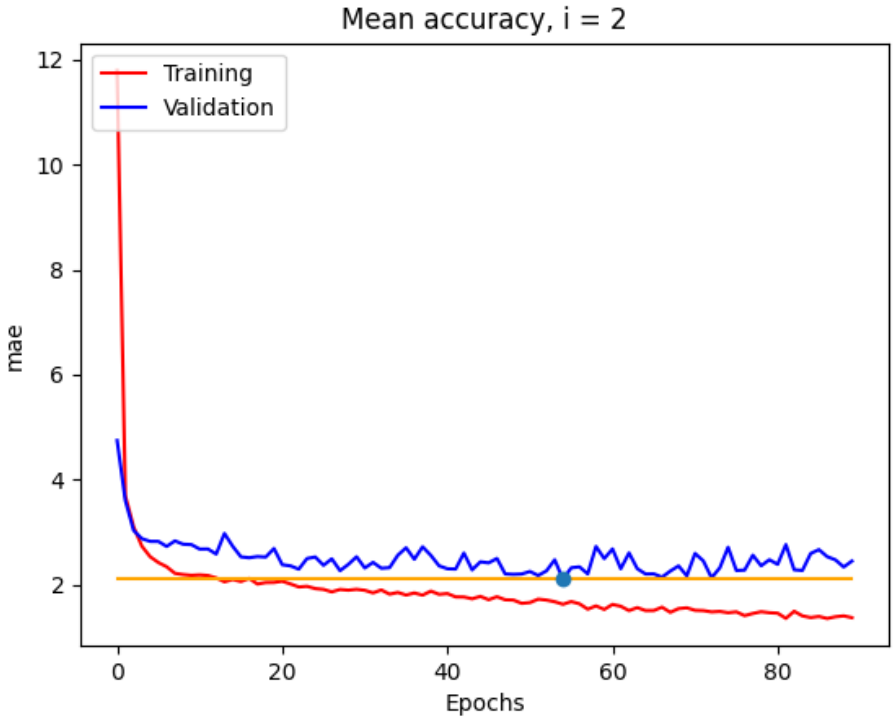


Рисунок 6. Вторая модель, первая подмодель



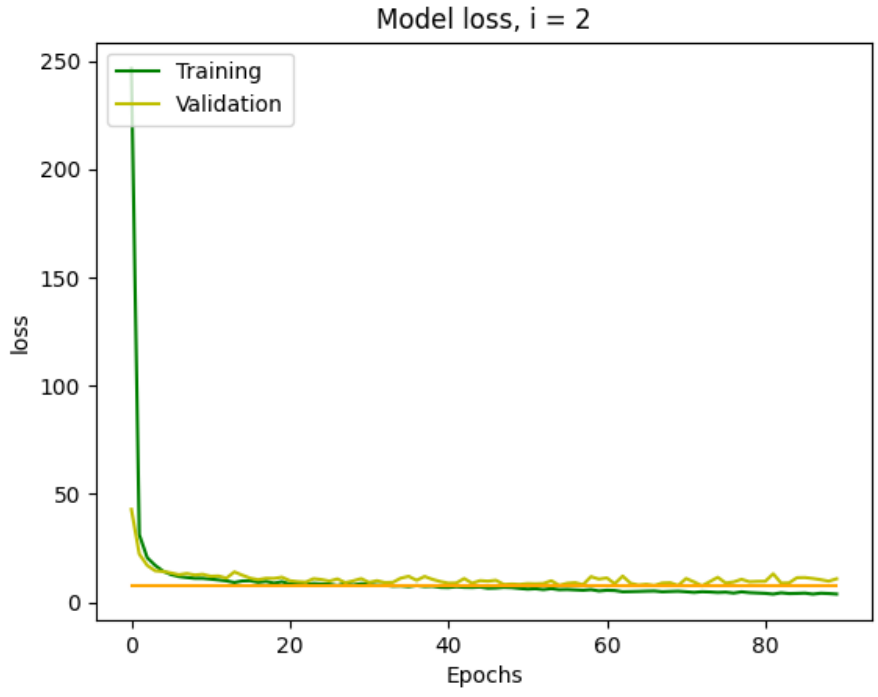


Рисунок 7. Вторая модель, вторая подмодель

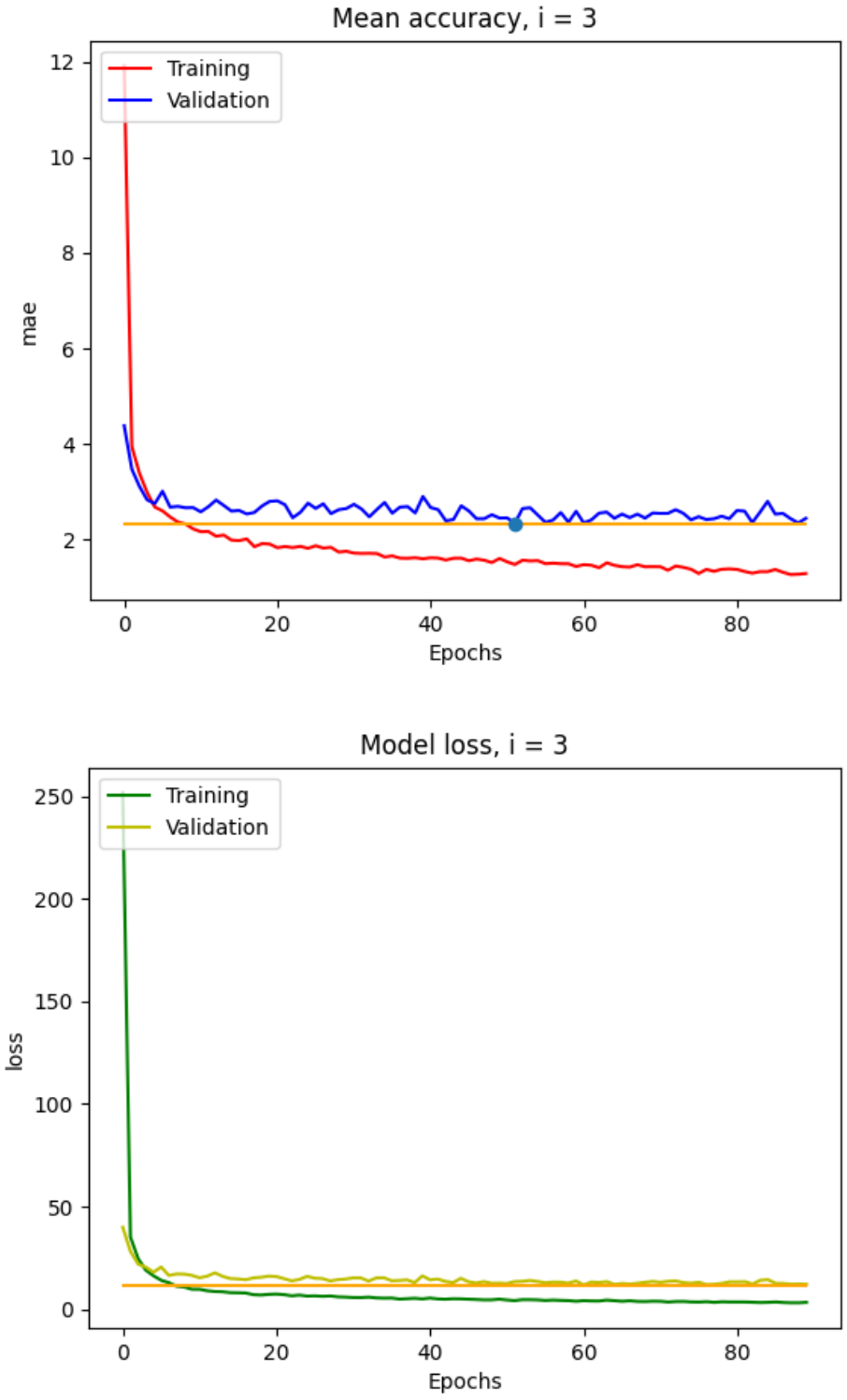
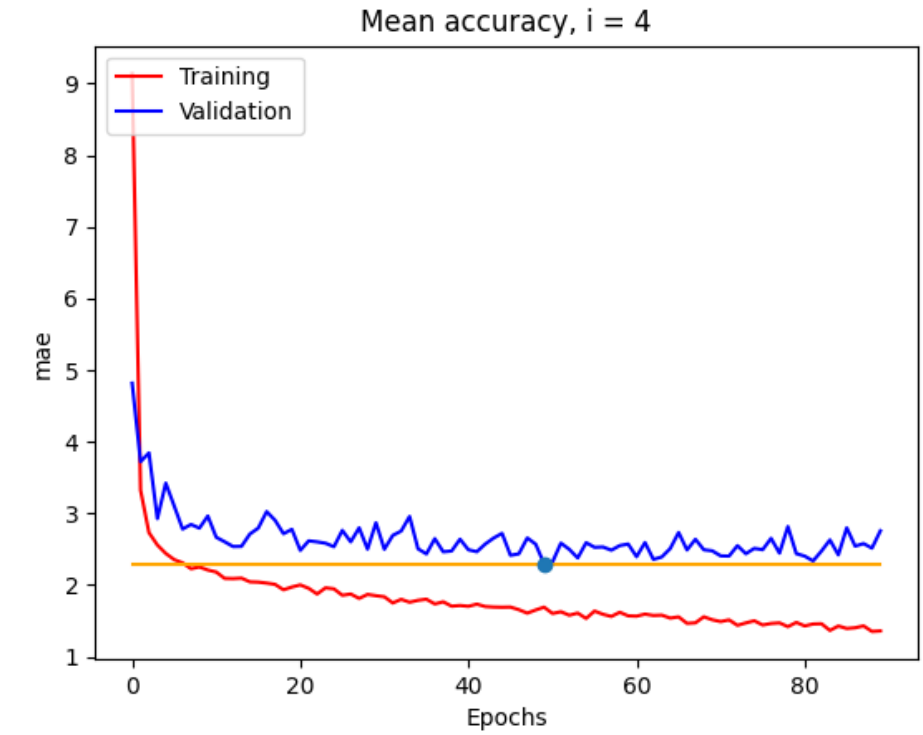


Рисунок 8. Вторая модель, третья подмодель



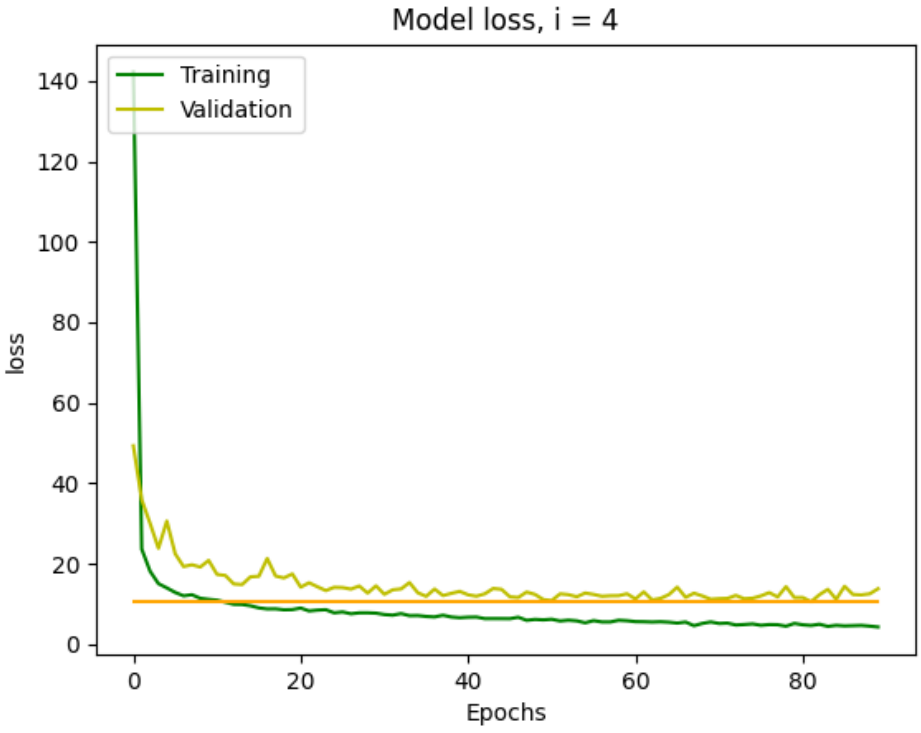


Рисунок 9. Вторая модель, четвертая подмодель

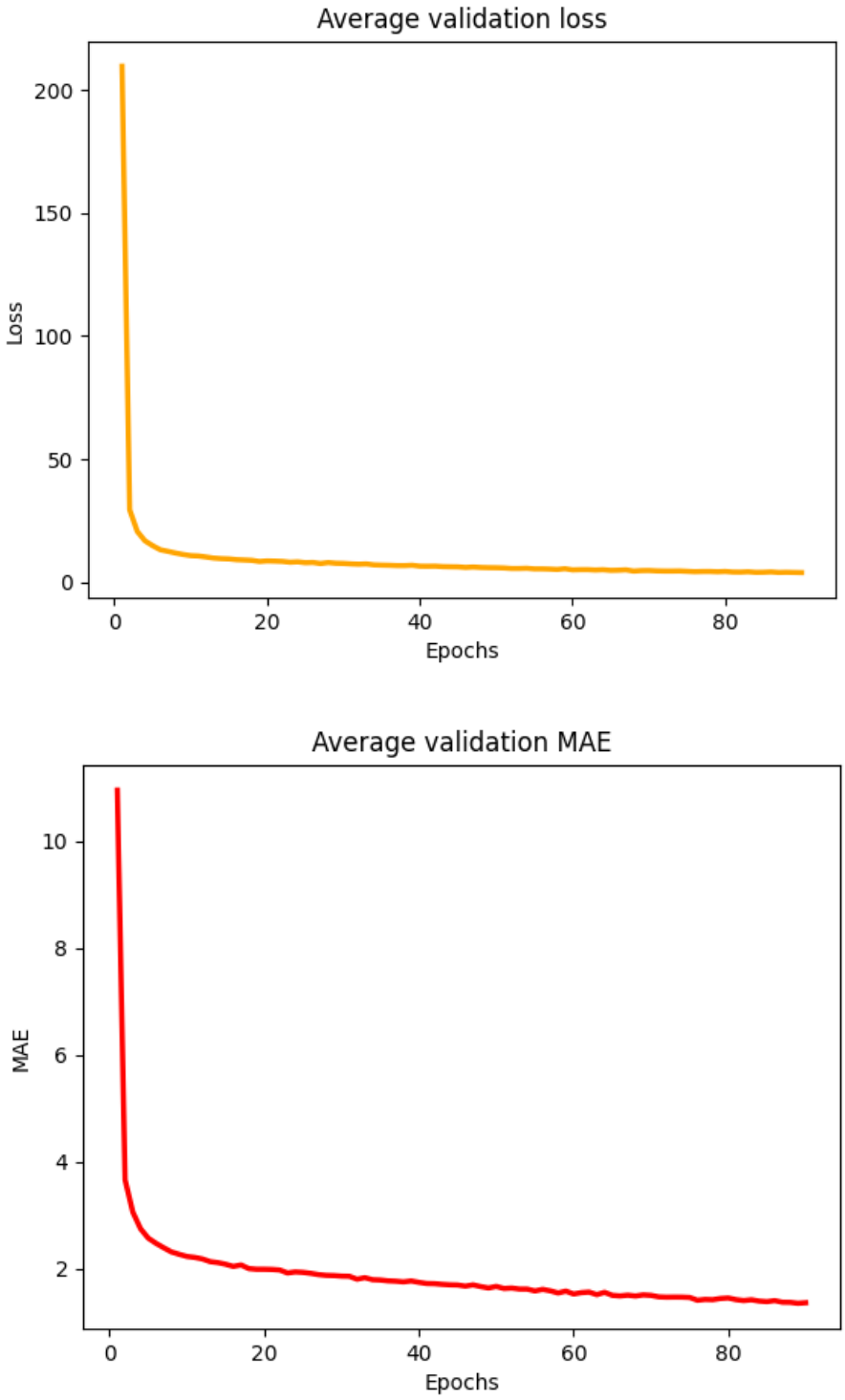


Рисунок 10. Показатели второй модели

При уменьшении числа эпох до 80 получена оценка 1.876, то есть результат стал хуже на 0.05. Также теперь минимальные показатели нейросети были получены за 50 эпох, значит примерно тогда произошло переобучение модели

Далее исследуется влияние числа блоков. Количество подмоделей увеличено до 6.

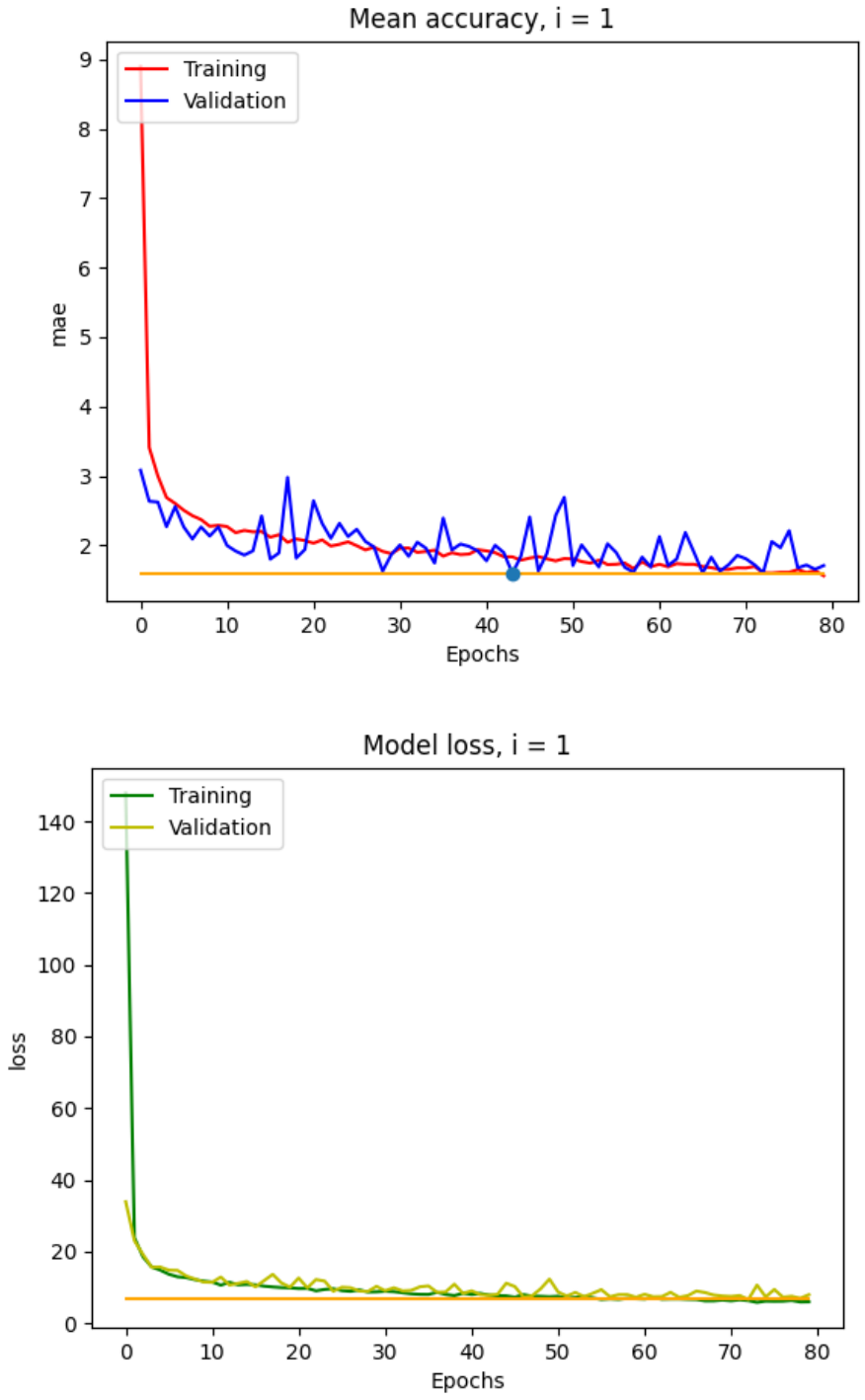


Рисунок 11. Третья модель, первая подмодель

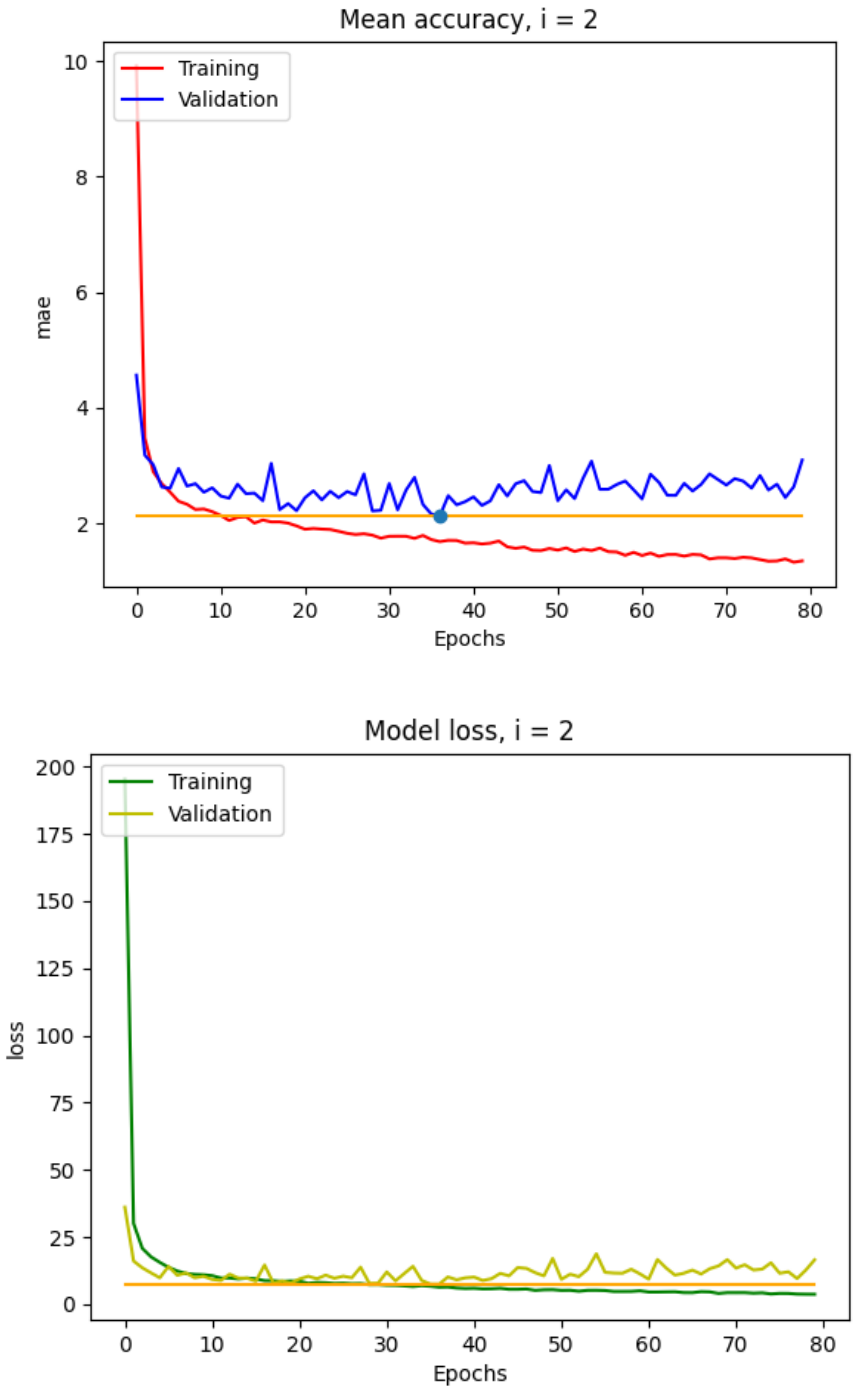


Рисунок 12. Третья модель, вторая подмодель

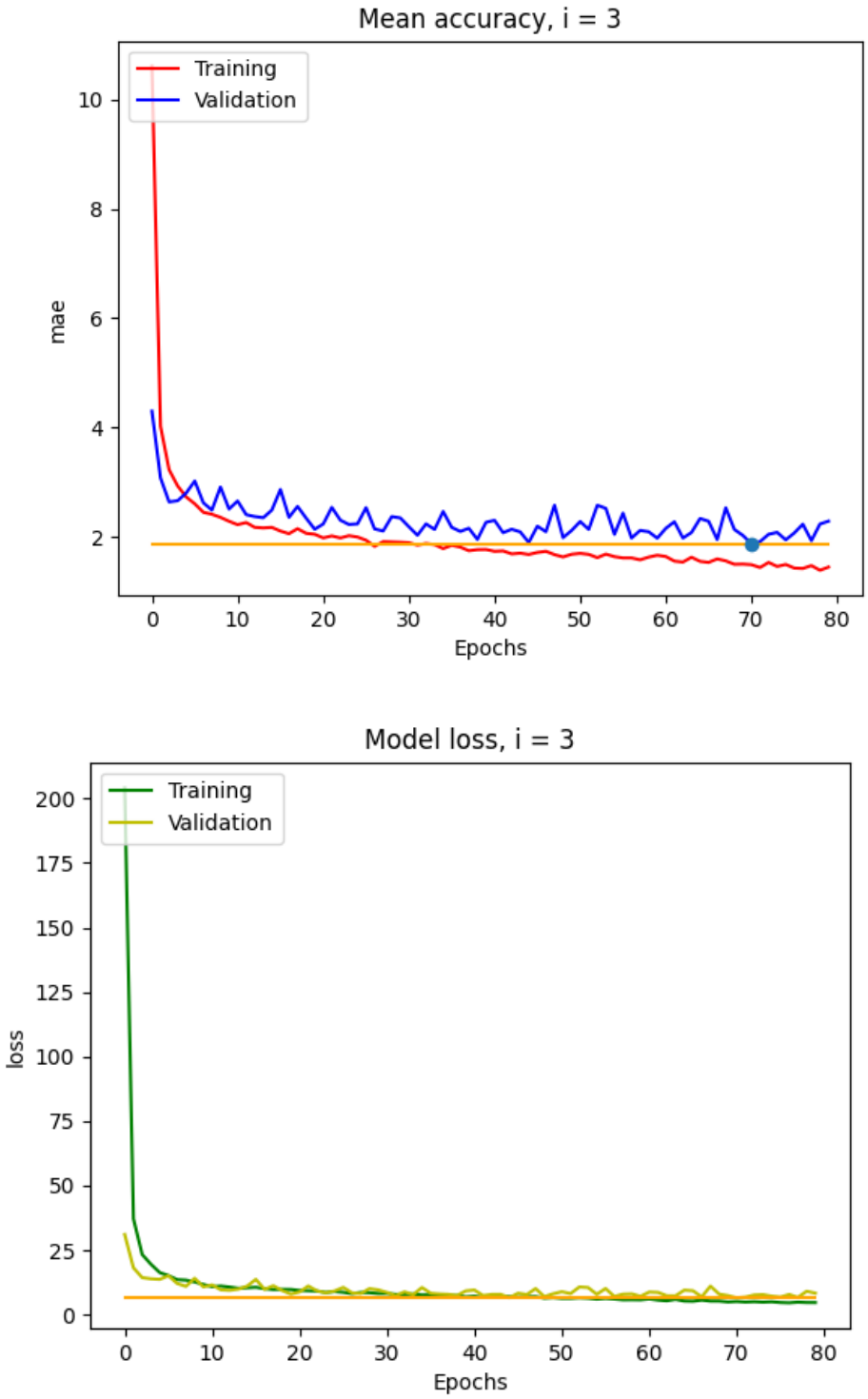


Рисунок 13. Третья модель, третья подмодель

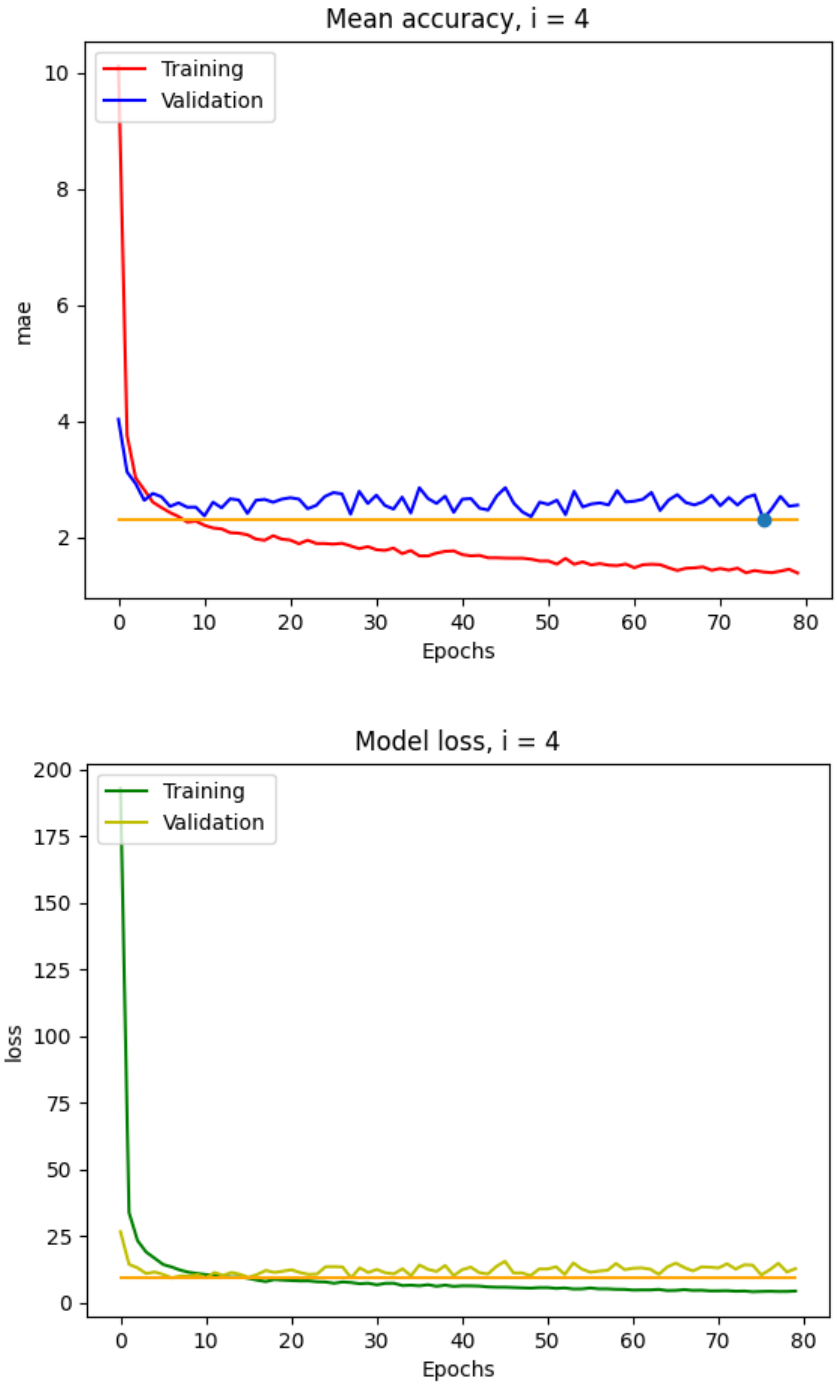


Рисунок 14. Третья модель, четвертая подмодель

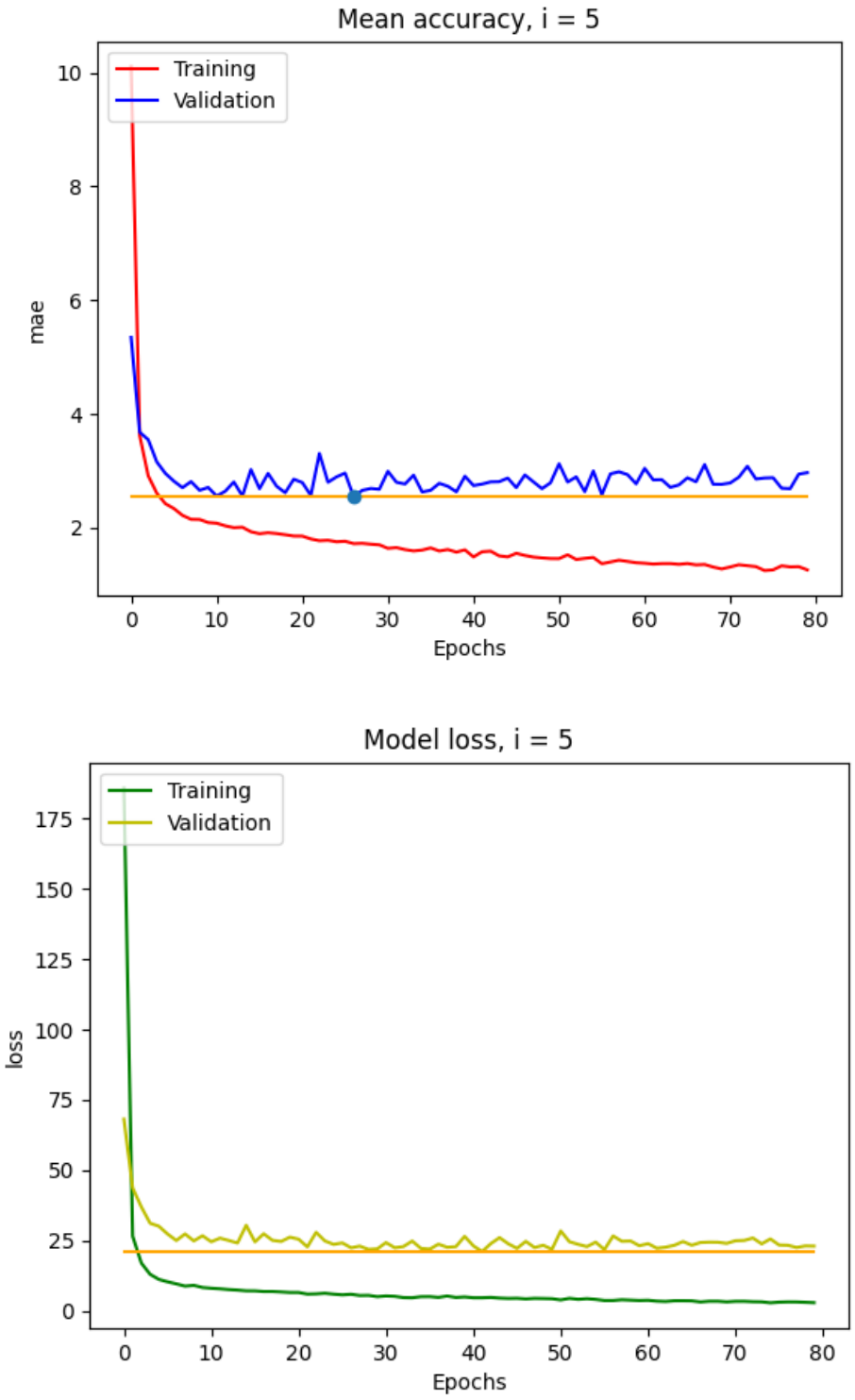


Рисунок 15. Третья модель, пятая подмодель

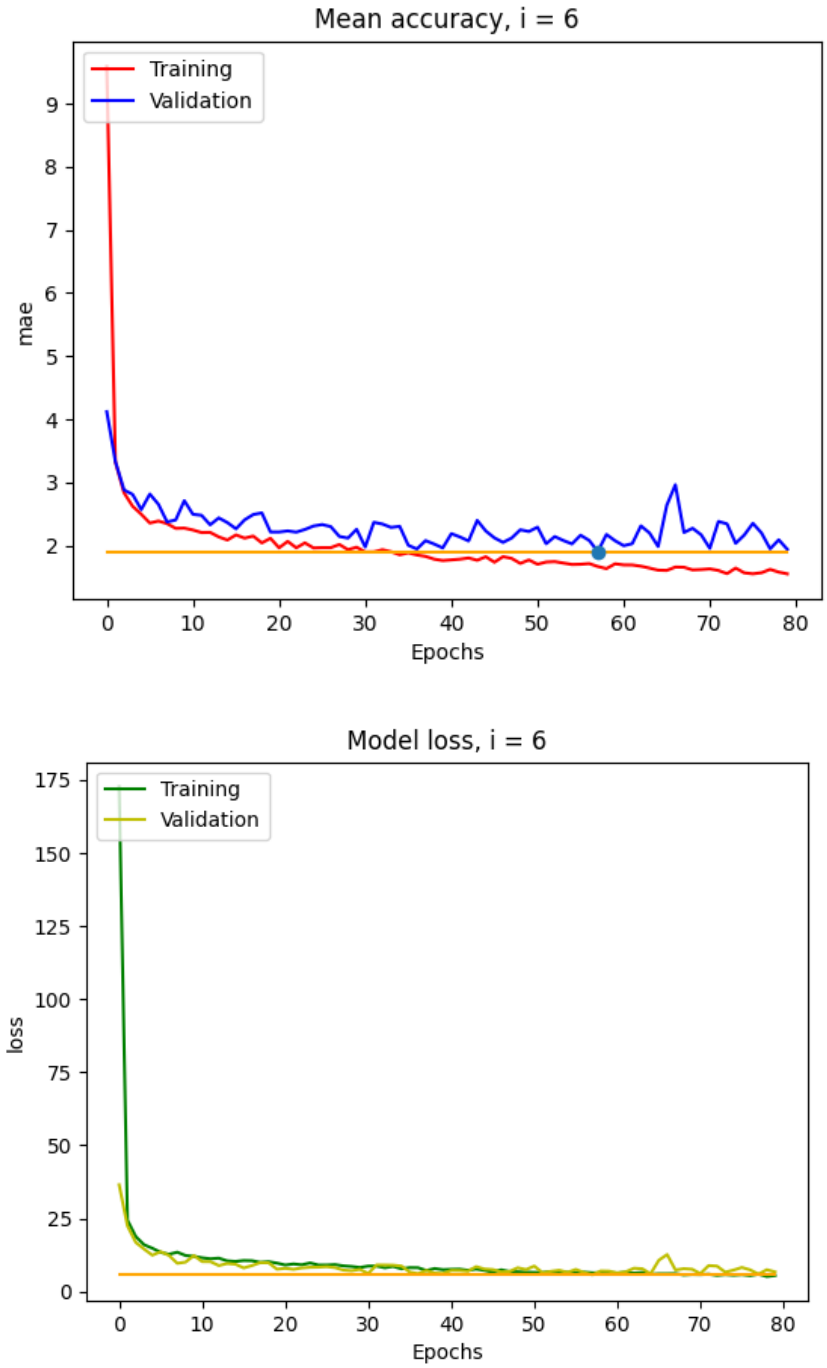


Рисунок 16. Третья модель, шестая подмодель

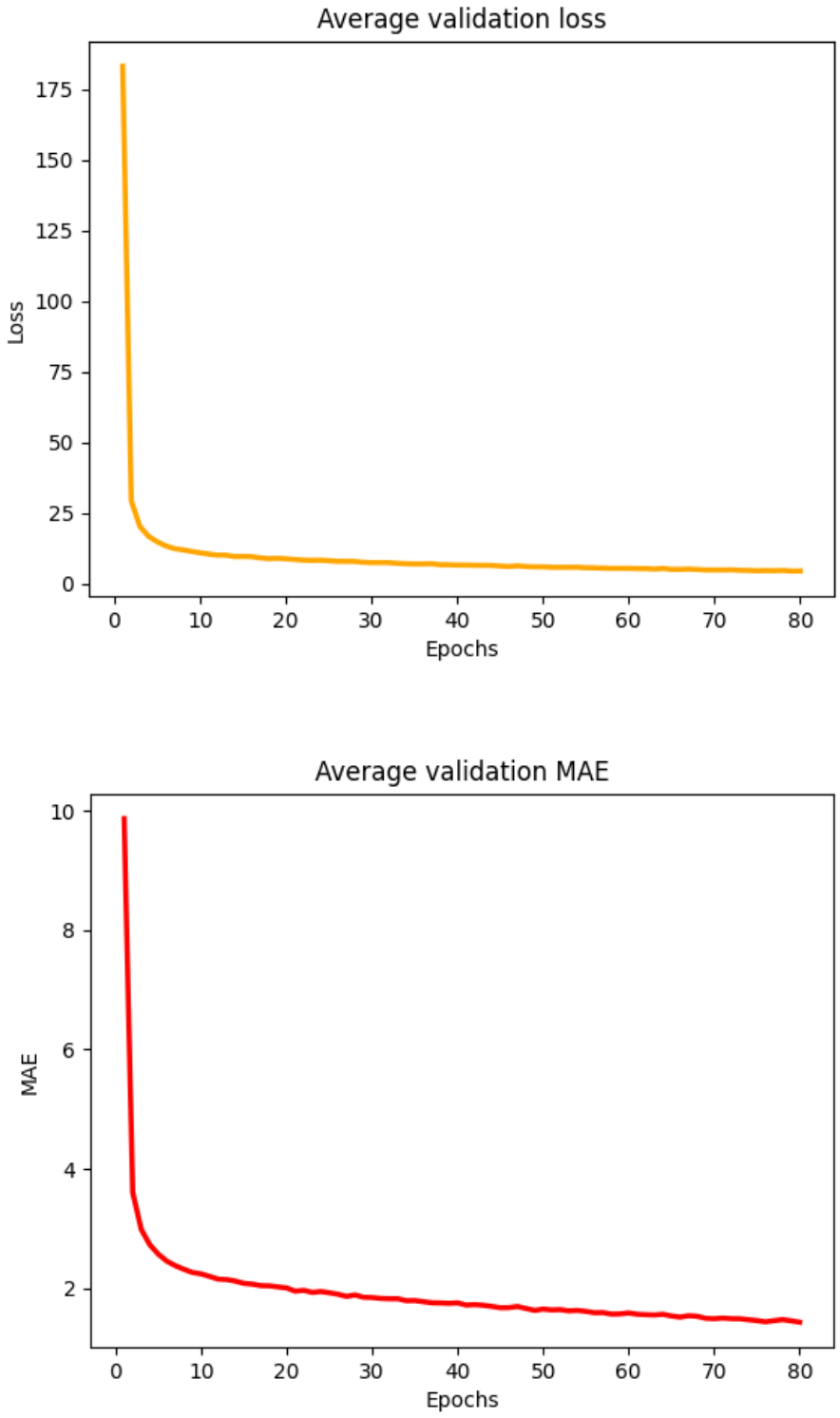


Рисунок 17. Показатели третьей модели

Для данной модели получена оценка 1.921, результат улучшился. Увеличение количества блоков ухудшило результат 1 модели на 0.1.

Таким образом, в ходе лабораторной работы были исследованы три модели: 100

эпох и 4 блока, 80 эпох и 4 блока, 80 эпох и 6 блоков. Лучшей оценкой является значение 1.820, полученное первой моделью.

Выводы.

В ходе лабораторной работы была построена модель нейронной сети для решения задачи регрессии: предсказания стоимости домов на основании некоторых параметров. Для этого был использован метод перекрестной проверки на K блоков, исследовано влияние количества блоков, эпох обучения, найдена точка переобучения.