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

Студент гр. 8383 _____ Ларин А.

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.

Задание

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

Требования

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

Выполнение

Необходимо решить задачу регрессии. Она отличается от задачи классификации тем, что в ней необходимо предсказать некоторый признак, представляющий из себя число, а не конечный набор значений(классов). Это напрямую отражается в конфигурации модели.
Сначала был загружен набор данных «Boston housing» из Keras.

Листинг 1:

```
      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
      import matplotlib.pyplot as plt

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

      print(test_targets)
```

Затем он был нормализован. Для этого из каждого значения вычтено среднее, а результат поделен на стандартное отклонение, что дает признаки центрированные по нулю и имеющие стандартное отклонение равное единице.
Листинг 2:

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

test_data -= mean
test_data /= std
```

Определена функция build_model(), строящая регрессионную модель.

Листинг 3:

```
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
```

В качестве функции потерь использована mse — среднеквадратичная ошибка. Также добавлен параметр mae — абсолютное отклонение предсказанного значения от целевого.

В последнем слое используется один нейрон буез функции активации, что позволяет предсказывать значения на всем вещественном интервале

В виду малого количества данных используется метод перекрестной проверки по K блокам, при котором данные разбиваются на K блоков, на K-1 блоков происходит обучения, а на 1 блоке данных — оценка

Листинг 4:

```
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()
    H = model.fit(partial_train_data, partial_train_targets,
epochs=num_epochs, batch_size=1,   validation_data=(val_data, val_targets),
verbose=0)
```

```
            val_mse, val_mae = model.evaluate(val_data, val_targets,
verbose=0)
            all_scores.append(val_mae)
```

График точности и ошибки для 100 эпох, K= 4 для каждой модели представлен на рис.1.1, 1.2 соответственно.

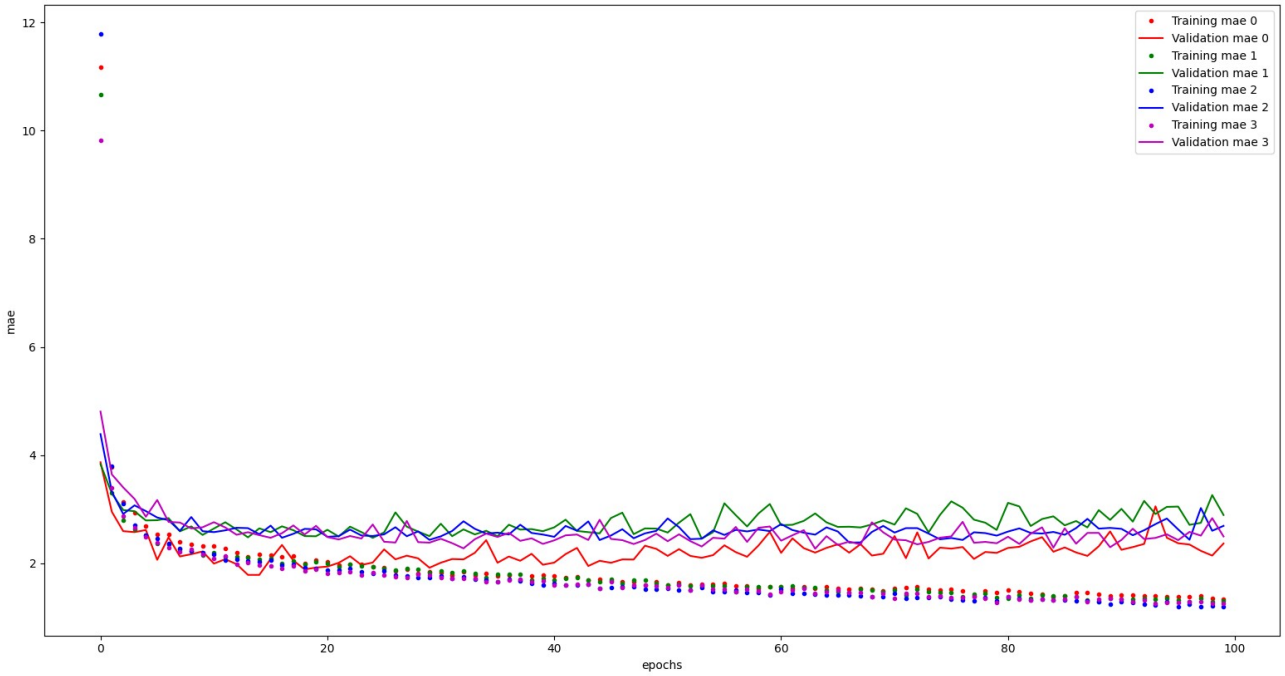

Рисунок 1.1 — График точности при 100 эпохах, K=4 для каждой модели

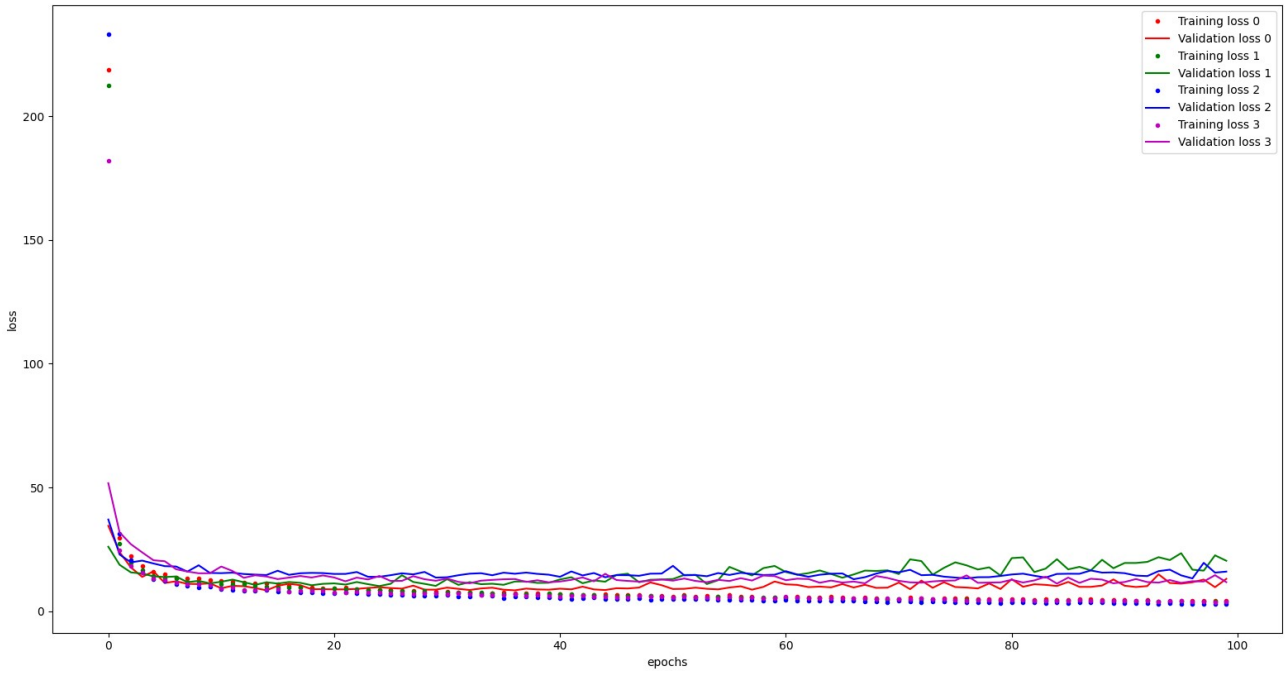

Рисунок 1.2 — График ошибки при 100 эпохах, K=4 для каждой модели

Также построен усредненный график по всем моделям. Он приведен на рис. 2.1, 2.2 для точности и ошибки соответственно.

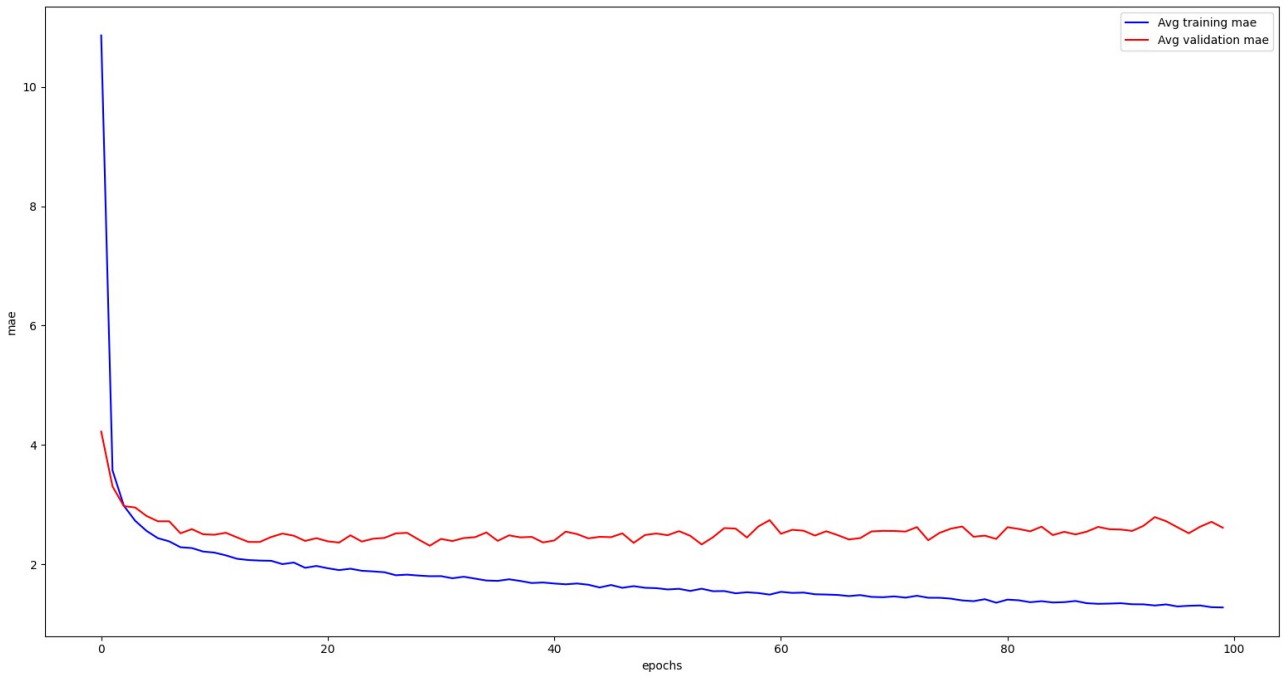

Рисунок 2.1 — График точности при 100 эпохах, K=4 усредненный по всем моделям

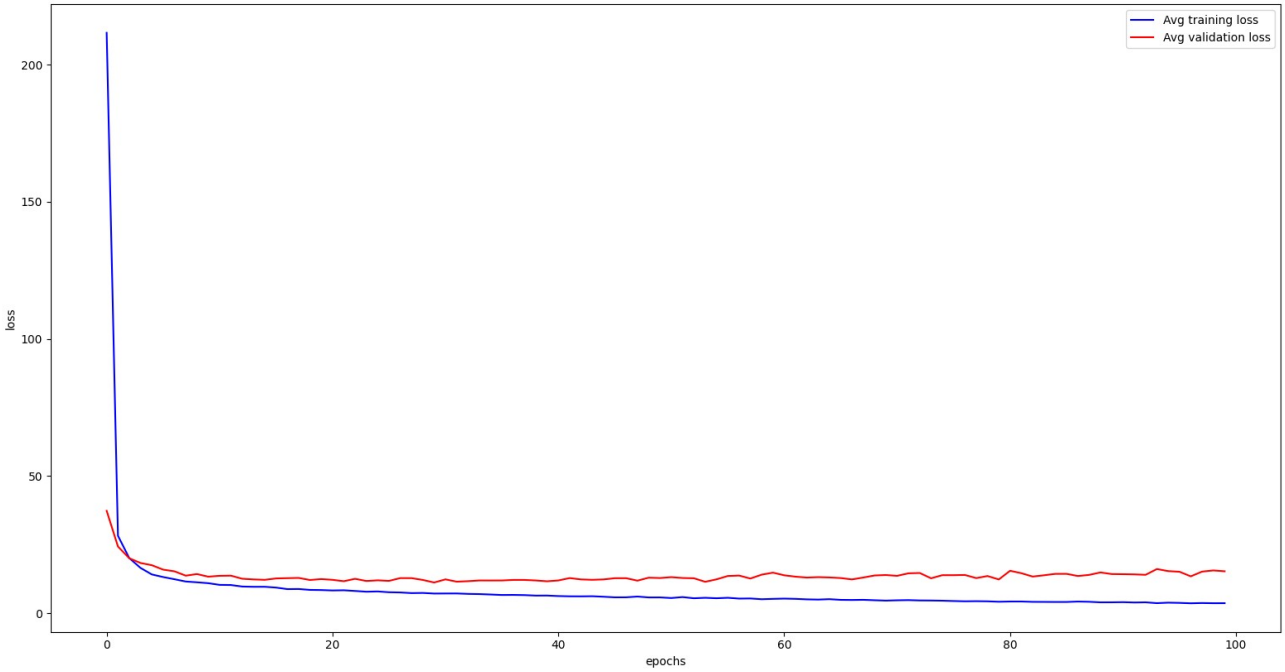

Рисунок 2.2 — График ошибки при 100 эпохах, K=4 усредненный по всем моделям

Среднее значение оценок модели получилось равным 2.6.

Количество эпох увеличено со 100 до 150. Графики точности и ошибки представлены на рис.3.1, 3.2. Точность по прежнему составила 2.6.

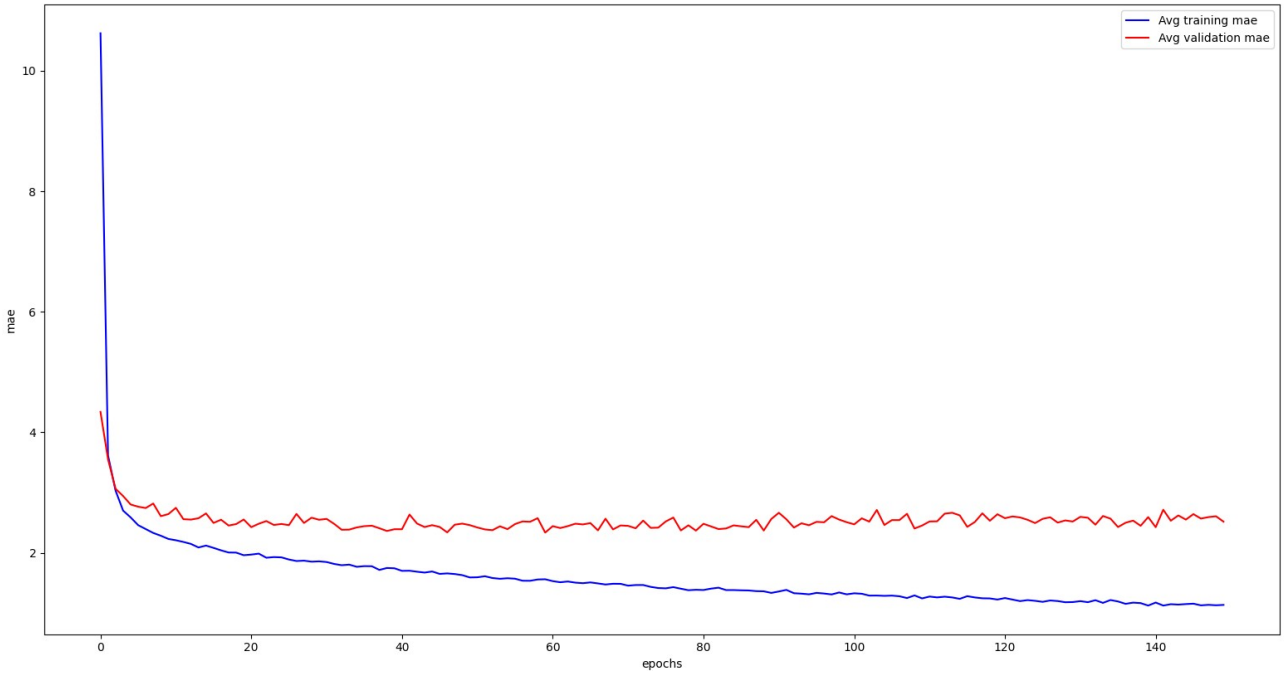

Рисунок 3.1 — График точности при 150 эпохах, K=4 усредненный по всем моделям

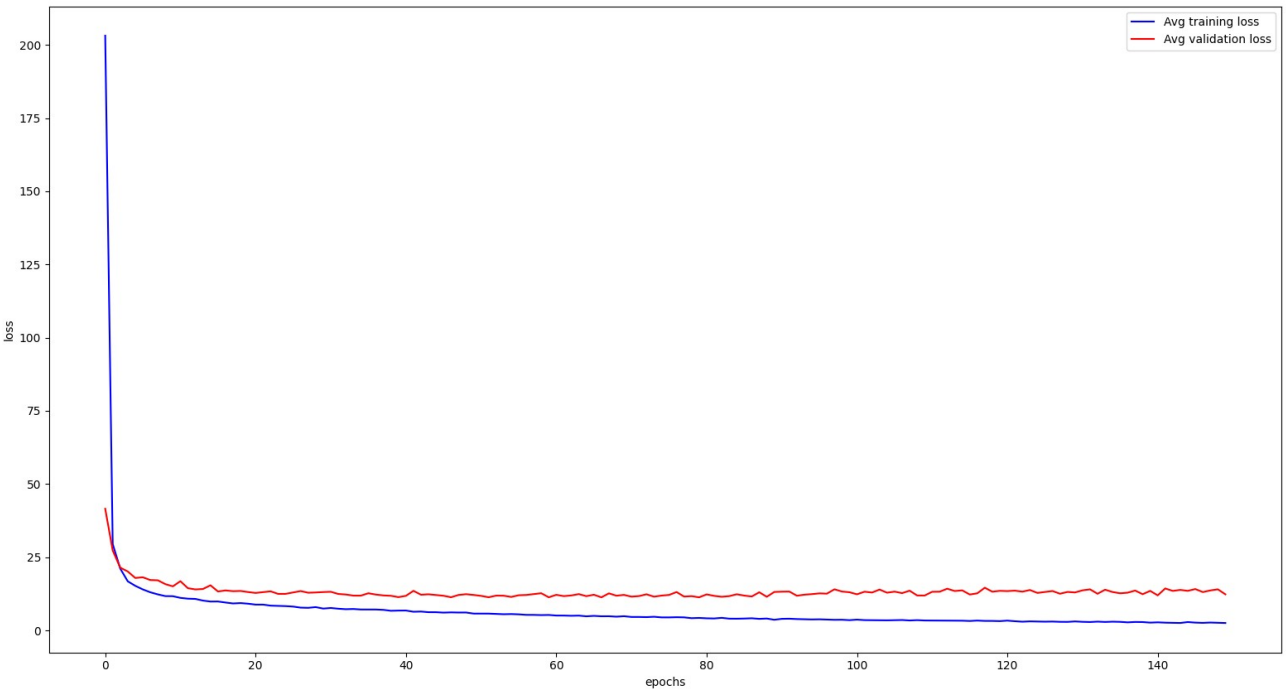

Рисунок 3.2 — График ошибки при 150 эпохах, K=4 усредненный по всем

Заметно что с 20 по последнюю эпоху модель почти не меняет своих характеристик и даже ухудшает свои показатели, что говорит о переобучении.
Судя по графику переобучение началось в области 30-50 эпохи. Модель повторно обучена на 40-ка эпохах. Результаты на графиках на рис.4.1, 4.2. Точность модели составила 2.4, что лучше предыдущих показателей.

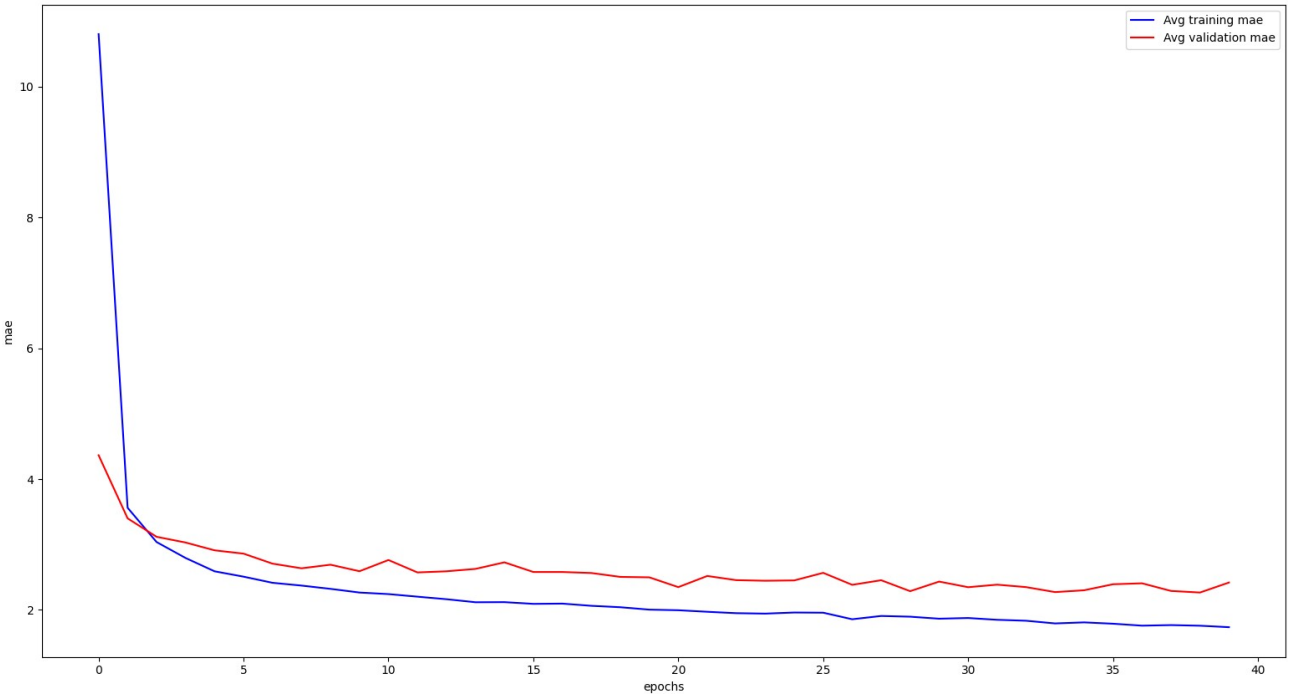

Рисунок 4.1 — График точности при 40 эпохах, K=4 усредненный по всем моделям

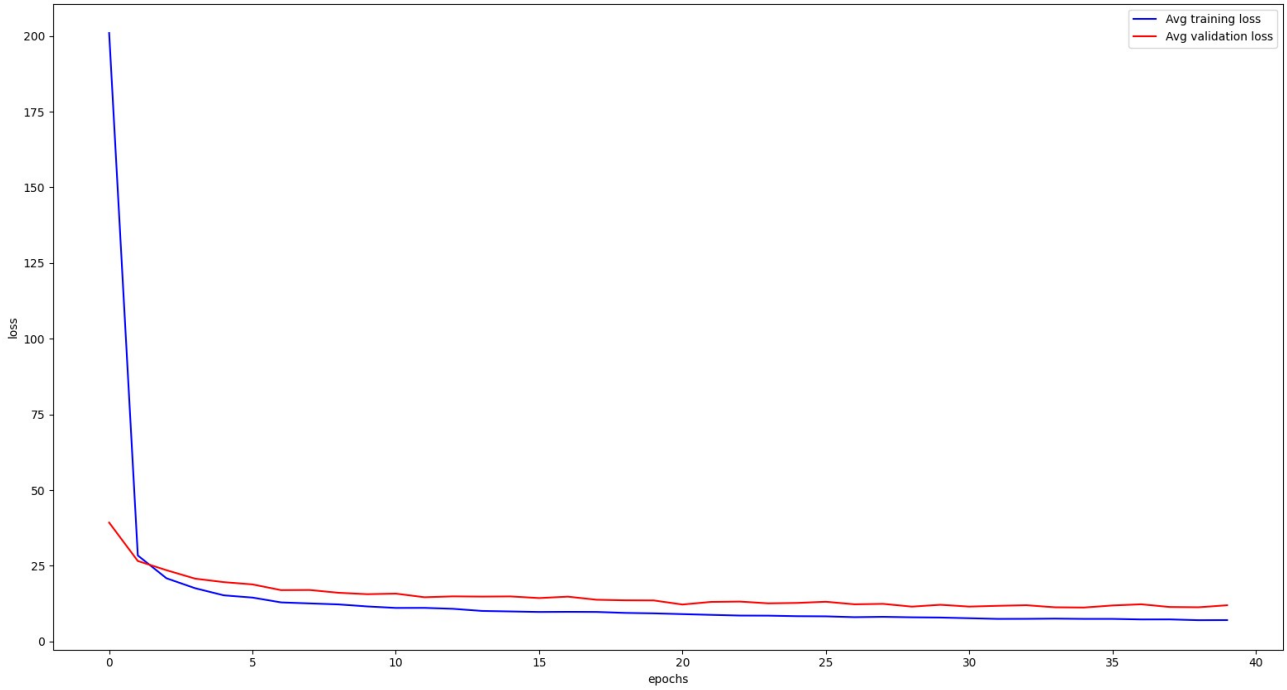

Рисунок 4.2 — График ошибки при 40 эпохах, K=4 усредненный по всем моделям

Далее проведен эксперимент по варьированию K.
Точность модели при K = 2 и 40 эпохах составила 2.42. График по всем моделям представлен на рис.5.1, 5.2. Усредненный график на рис.6.1, 6.2

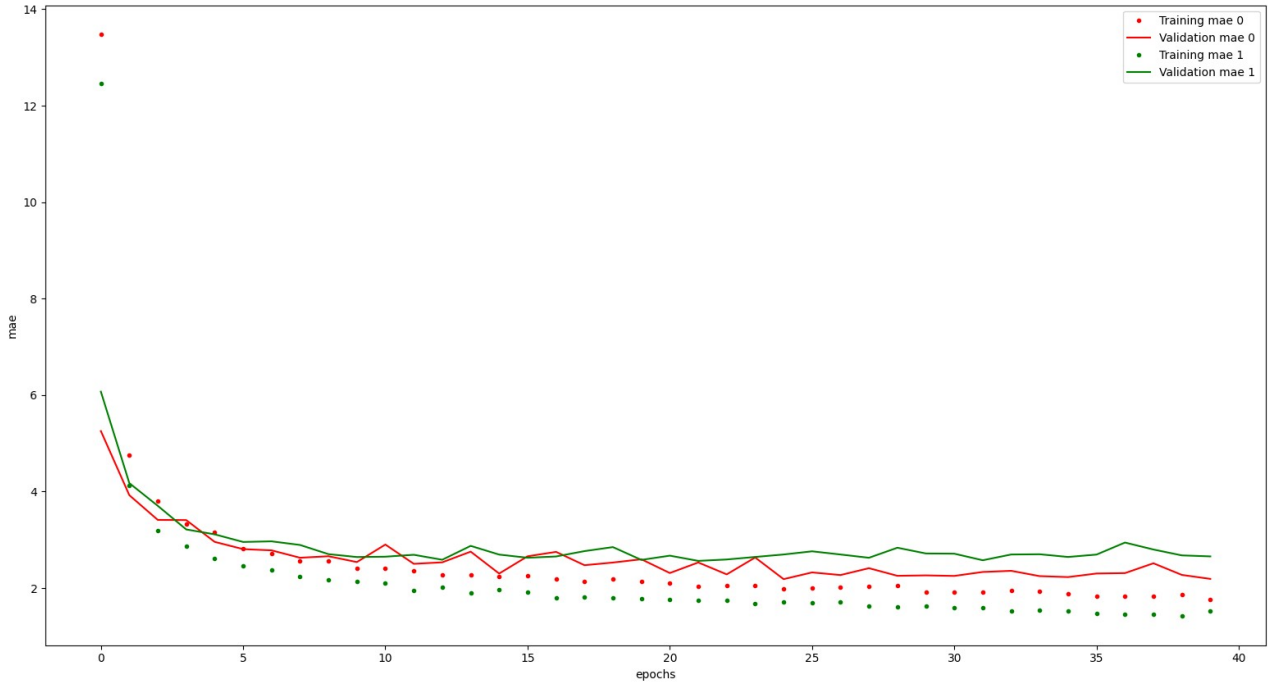

Рисунок 5.1 — График точности при 40 эпохах, K=2 для каждой модели

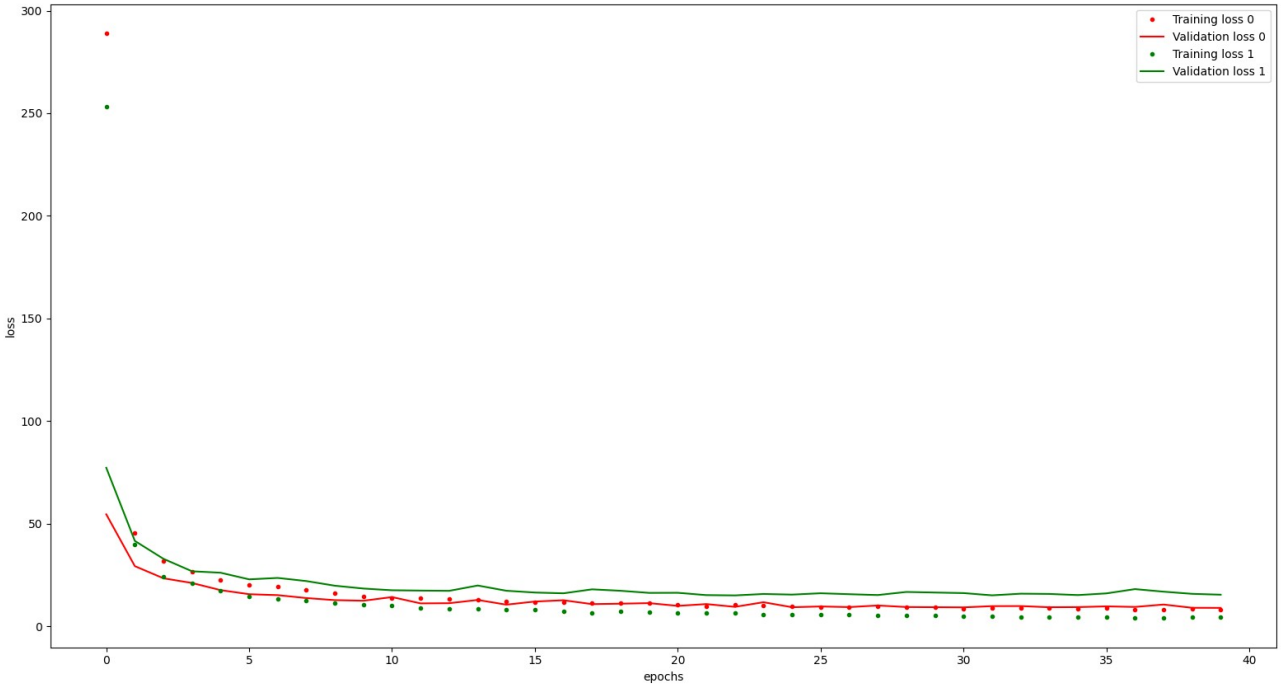

Рисунок 5.2 — График ошибки при 40 эпохах, K=2 для каждой модели

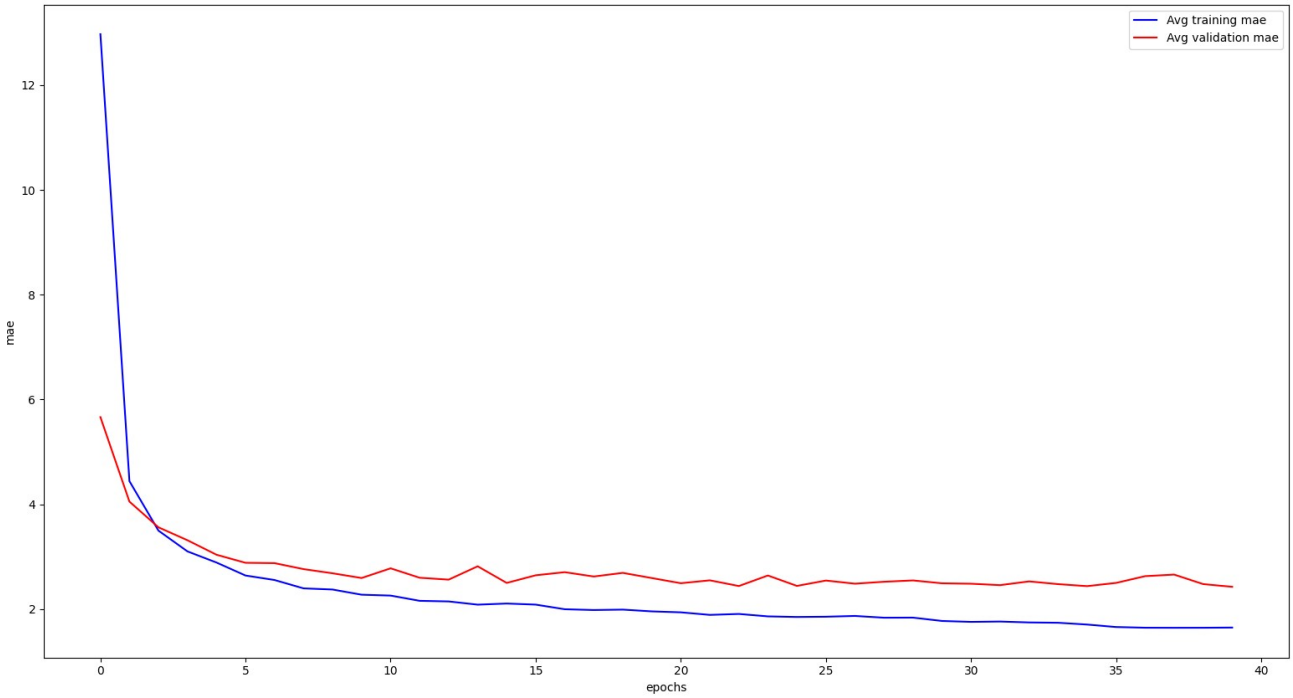

Рисунок 6.1 — График точности при 40 эпохах, K=2 усредненный по всем моделям

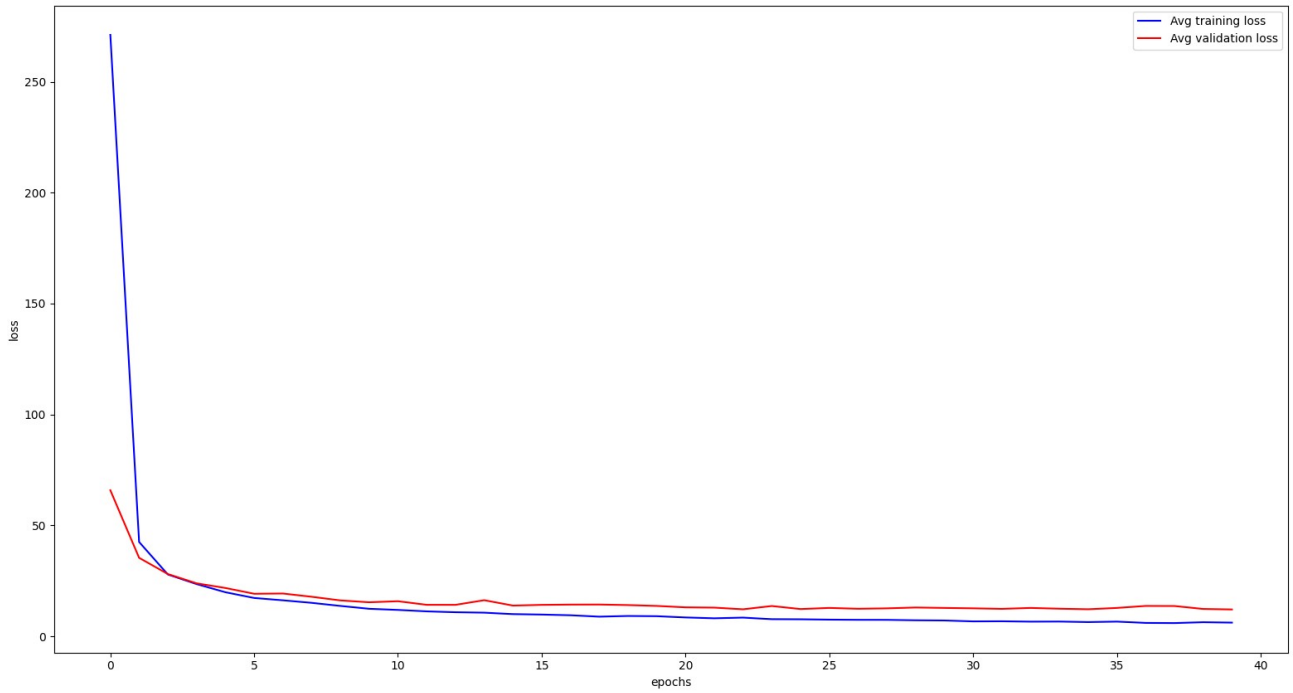

Рисунок 6.2 — График ошибки при 40 эпохах, K=2 усредненный по всем моделям

Для модели при K = 6 и 40 эпохах точность составила 2.34. Усредненный график представлен на рис.7.1, 7.2.

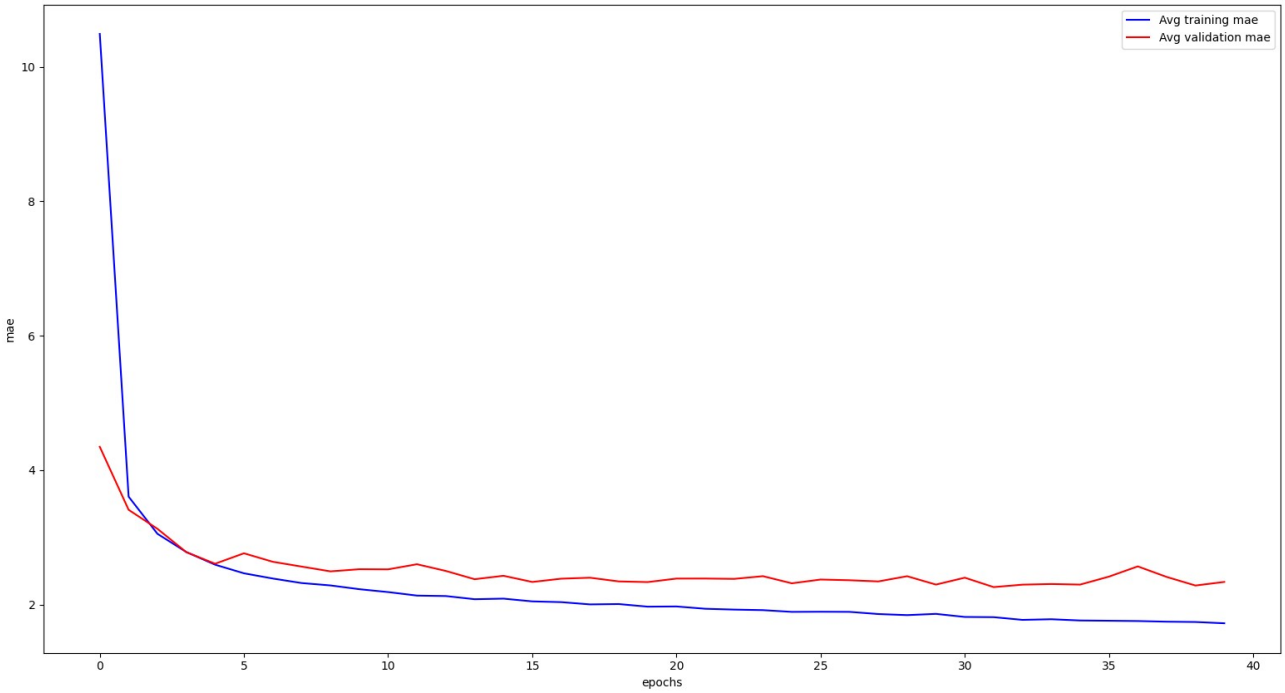

Рисунок 7.1 — График точности при 40 эпохах, K=6 усредненный по всем моделям

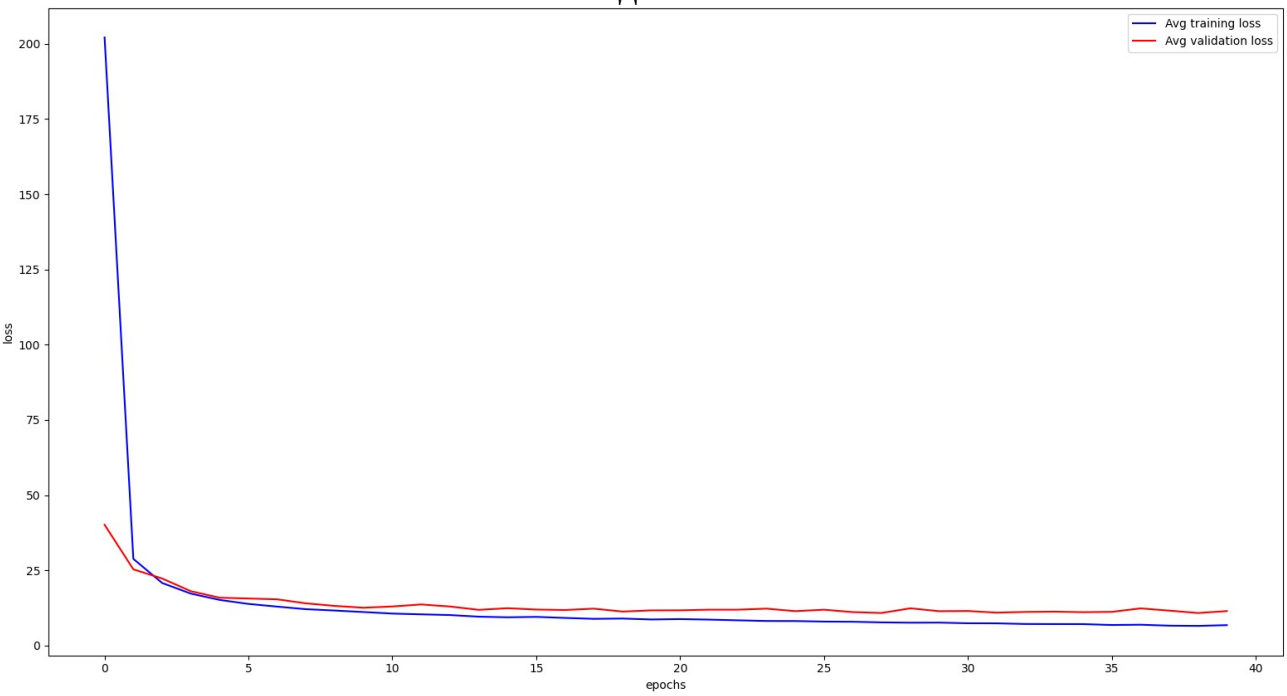

Рисунок 7.2 — График ошибки при 40 эпохах, K=6 усредненный по всем моделям

Для модели при K = 8 и 40 эпохах точность составила 2.4. Усредненный график представлен на рис.8.1, 8.2.

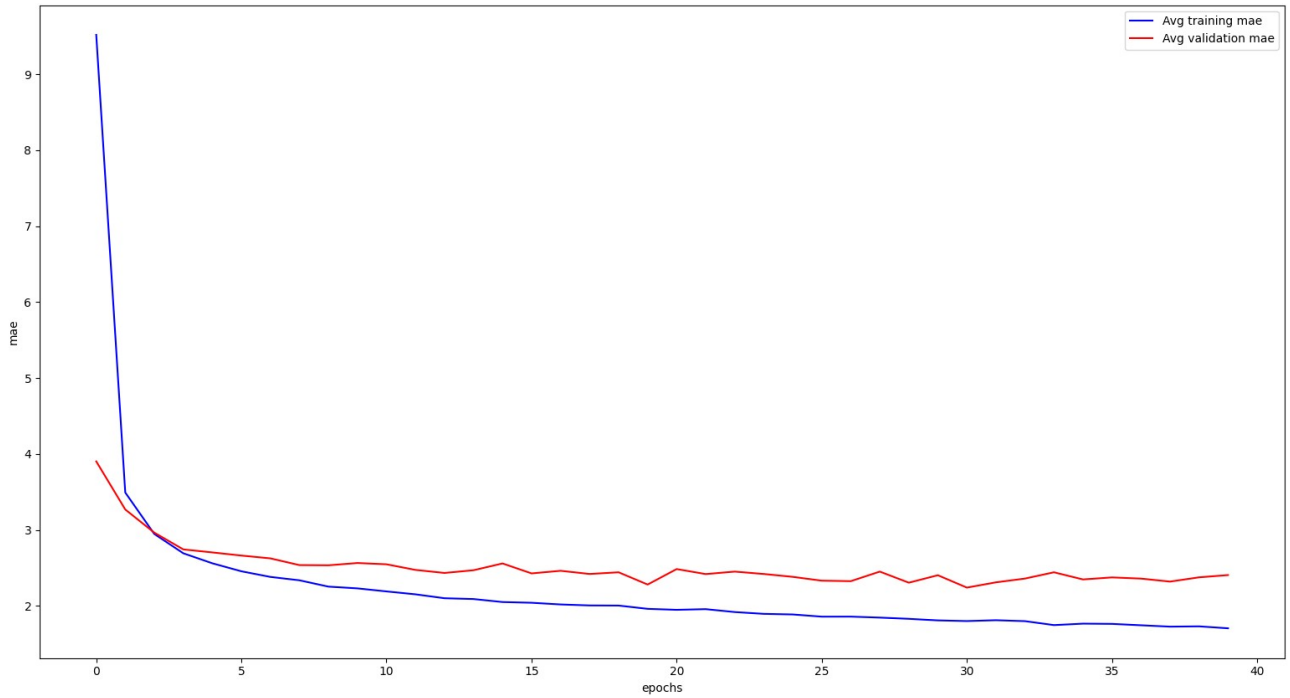

Рисунок 8.1 — График точности при 40 эпохах, K=8 усредненный по всем моделям

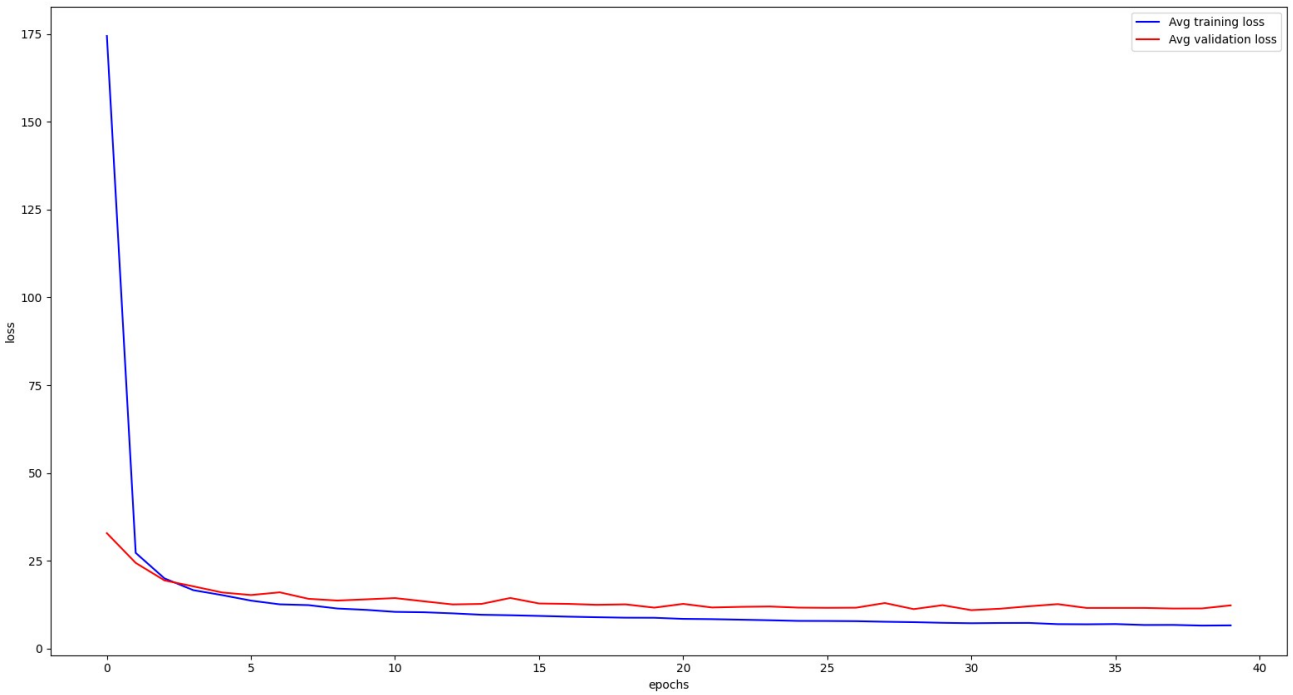

Рисунок 8.2 — График ошибки при 40 эпохах, K=8 усредненный по всем моделям

Для модели при K = 10 и 40 эпохах точность составила 2.43. При дальнейшем повышении числа блоков точность падает. Самое низкое значение mae соответствует K=6. Дополнительно проверены его окрестности: при K=5 mae~2.35, при K=7 mae~2.4. Т.о. K=6 дает лучшие результаты mae~2.3

Выводы.

Проведены эксперименты по построению регрессионных моделей для предсказания цен на дома в пригороде бостона по датасету «Boston Housing Dataset». Было проварьировано число эпохи выявлено, что повышение их числа не несет пользы в связи с переобучением. Приблизительно установлена точка переобучения в окрестности 40 эпох.

Проварьировано число блоков K для кросс-валидации. При K=2,4,8 точность практически неизменна и составляет ~2.4, а при более высоких значениях начинает снижаться, однако при K=6 mae составляет 2.3.