НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет систем управления и робототехники

Лабораторная работа № 2

Прогнозирование цен на жилье с помощью нейросетевой регрессионной модели

Выполнил студент

Шамсулов Руслан Наилевич

Группа № R3235

Преподаватель: Mikhail Kakanov,

Oleg Evstafev

г. Санкт-Петербург

2021

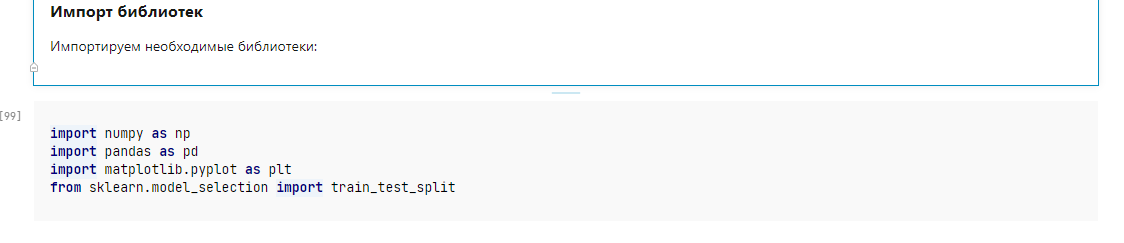
**Текст задания**

Необходимо по имеющимся данным о ценах на жильё предсказать окончательную цену каждого дома с учетом характеристик домов с использованием нейронной сети. Описание набора данных содержит 80 классов (набор переменых) классификации оценки типа жилья, и находится в файле data\_description.txt.

В работе требуется дополнить раздел «Моделирование» в подразделе «Построение и обучение модели» создать и инициализировать последовательную модель нейронной сети с помощью фрэймворков тренировки нейронных сетей как: Torch или Tensorflow. Скомпилировать нейронную сеть выбрав функцию потерь и оптимизатор соответственно. Оценить точность полученных результатов. Вывести предсказанные данные о продаже.

**Ход работы**

1) Общий код для всех моделей независимо от гиперпараметров

****

**Изображение выглядит как текст

Автоматически созданное описание**

**Изображение выглядит как стол

Автоматически созданное описание**

**Изображение выглядит как стол

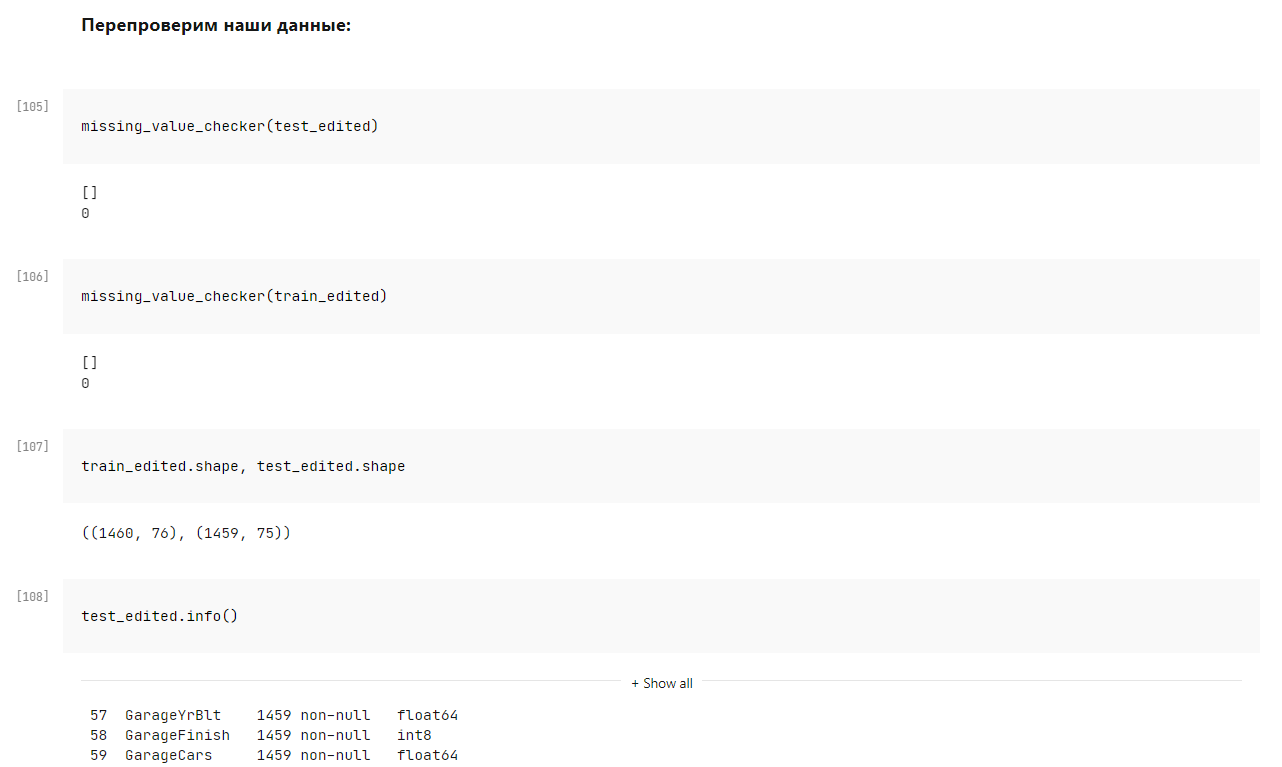
Автоматически созданное описание**

**Изображение выглядит как текст

Автоматически созданное описание**

**Изображение выглядит как текст

Автоматически созданное описание**

****

**Изображение выглядит как текст

Автоматически созданное описание**

**Изображение выглядит как текст

Автоматически созданное описание**

**2) Подбор различных гипермараметров NN**

В ходе работы необходимо перебробовать различные гиперпараметры сети в соответсвии с заданием :

**При выполнении:**

* Попробуйте использовать разное количество нейронов на входном слое, например 100, 150, 200 300.
* Добавьте в нейронную сеть скрытый слой с разным количеством нейронов.
* Используйте разное количество эпох: 10, 15, 20, 25, 30.
* Используйте разные размеры мини-выборки (batch\_size): 10, 50, 100, 200.
* Попробуйте использовать разные значения оптимизатора optimizers и функции потерь loss. Сравните полученые результаты.

Проверим роботоспособность кода с такими значениями гипермараметров:

* 200 нейронов на входном слое
* 2 скрытых слоя с 200 нейронами
* 30 эпох
* Размер мини-выборки 100
* Loss = msle
* Optimizer = adam

Изображение выглядит как текст

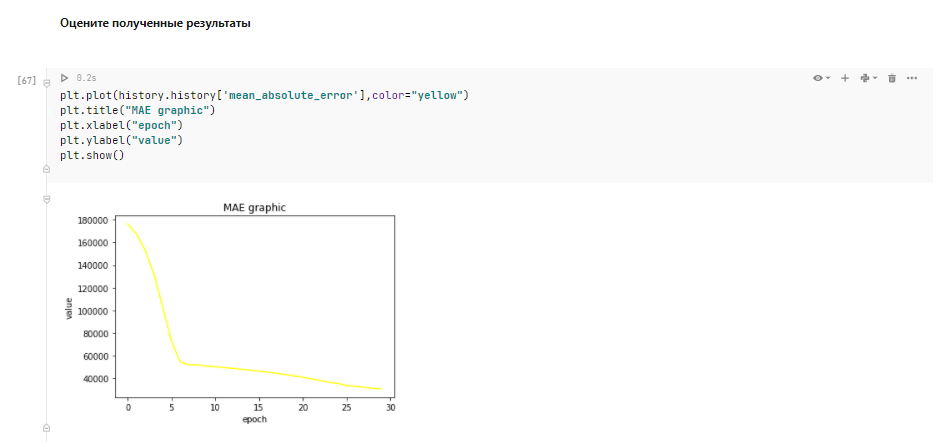
Автоматически созданное описание

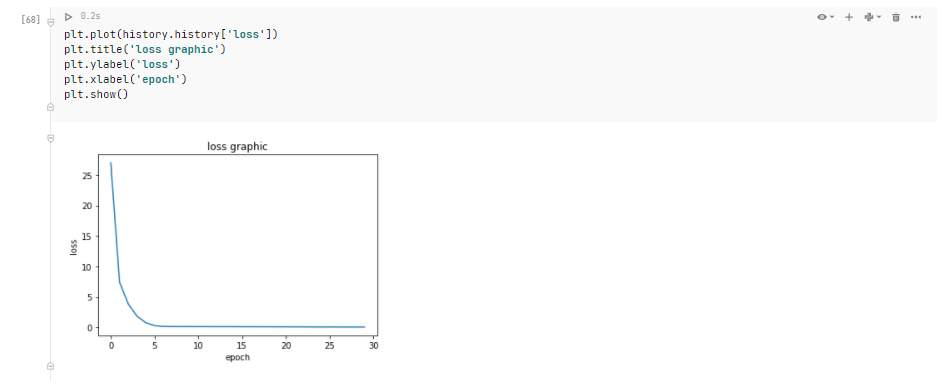
Изображение выглядит как текст

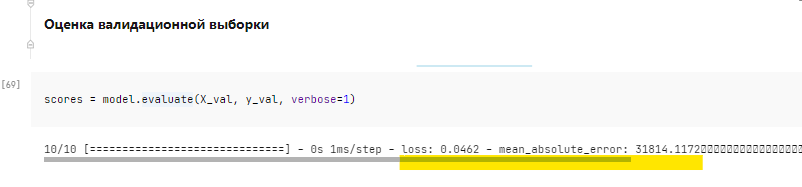
Автоматически созданное описание

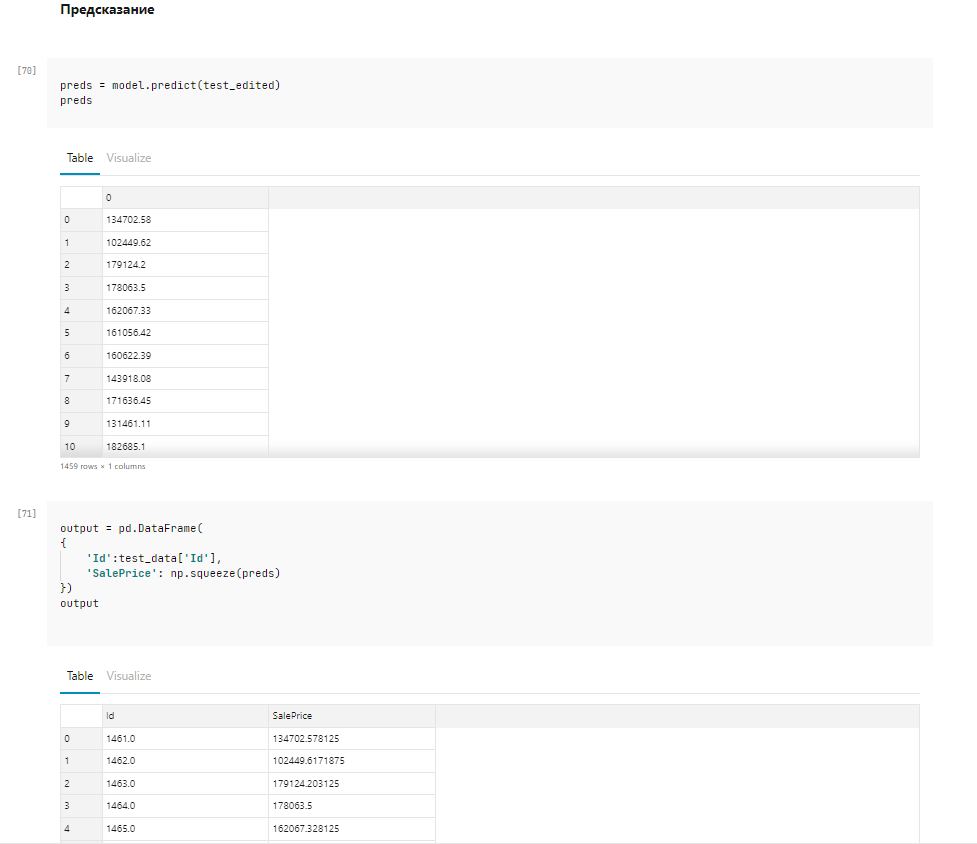
Изображение выглядит как текст

Автоматически созданное описание





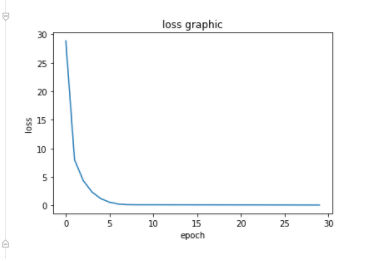
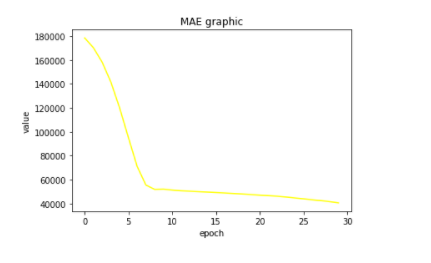




Далее будем менять гипермараметры и оценивать точность нашей нейронной сети

**Для начала будем варьировать количество нейронов на входном слое:**

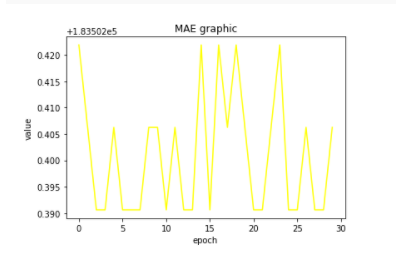
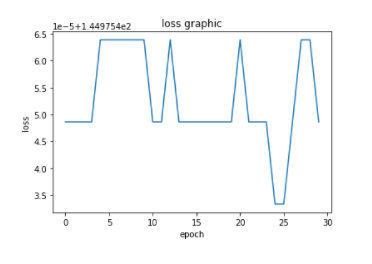
1) in\_layers = 100



Изображение выглядит как текст

Автоматически созданное описание

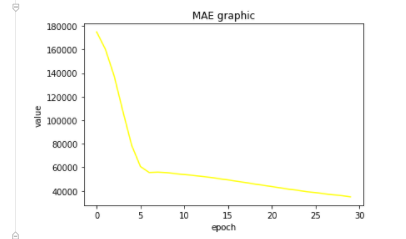
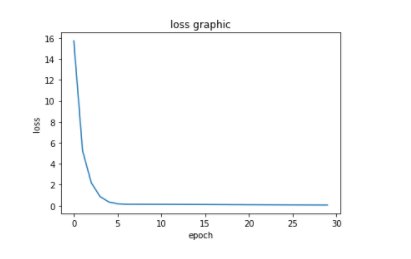
2) in\_layers = 150

Изображение выглядит как текст

Автоматически созданное описание

3) in\_layers = 300

Изображение выглядит как текст

Автоматически созданное описание

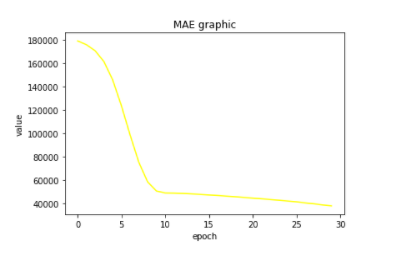
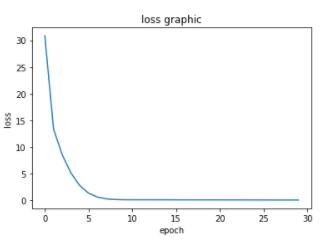
**Наилучшие показатели на валидационной выборке:**

**In\_layers = 200**: loss: 0.0462 - mean\_absolute\_error: 31814.11

**In\_layers = 300**: loss: 0.0484 - mean\_absolute\_error: 31486.28

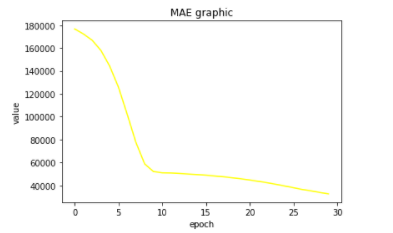
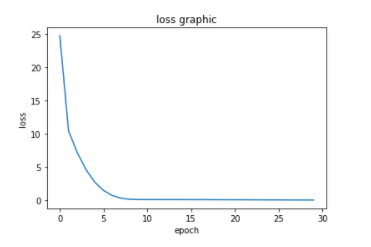
**Теперь будем изменять количество нейронов в скрытом слое (оставим in\_layers = 200, 2 скрытых слоя)**

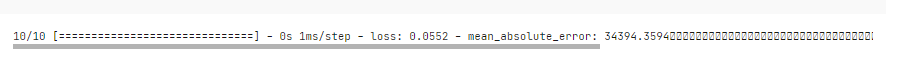
1) Hided = 100

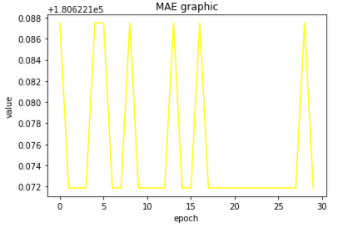
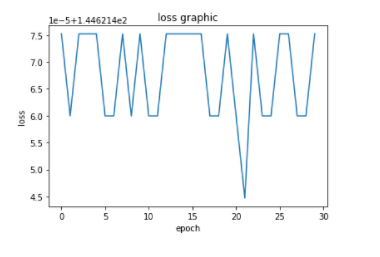


2) Hided = 150



3) Hided = 300

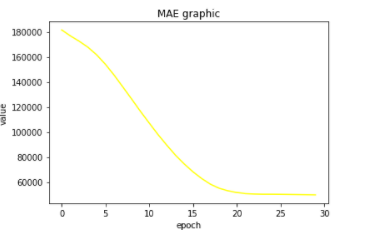
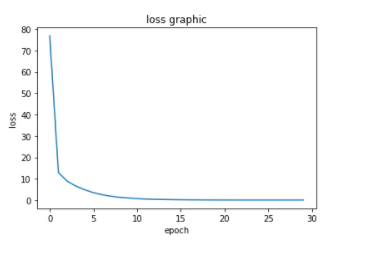
 



**Наилучшие показатели на валидационной выборке при hided = 200 (loss = 0.05, mae = 31815)**

**Попробуем добавить/ убрать один скрытый слой (hided = 200)**

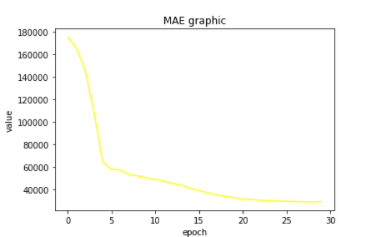
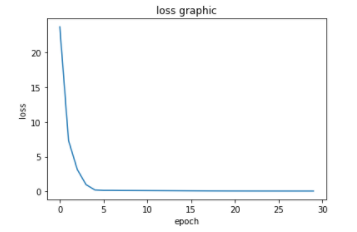
**1) Убираем один слой (теперь всего 1 скрытый слой)**

**** ****

****

**Точность упала**

**2) Добавляем один слой (теперь всего 3 скрытых слоя)**

**** ****

****

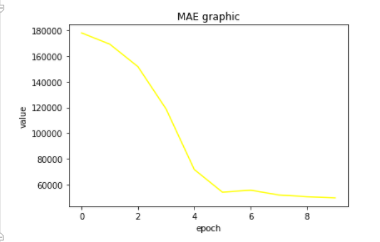
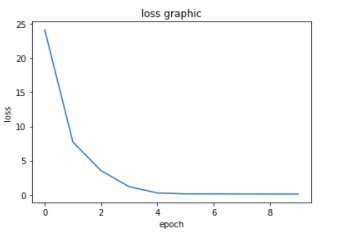
**Показатели loss и mae улучшились, следовательно оставим 3 скрытых слоя**

**При добавлении еще одного слоя (4 скрытых слоя показатели ухудшаются)**

**Вариация количества эпох для обучения (epochs)**

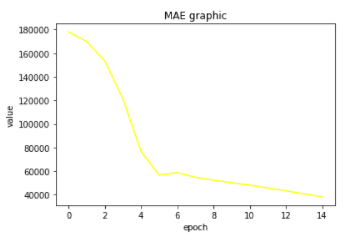
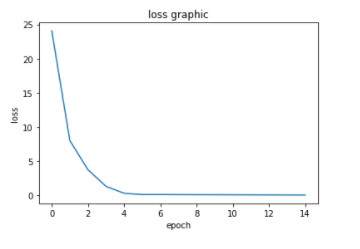
**Прежние наиболее оптимальные гипераметры: in\_layers = 200, hided = 200 (3 слоя)**

**1) epochs = 10**

**** ****

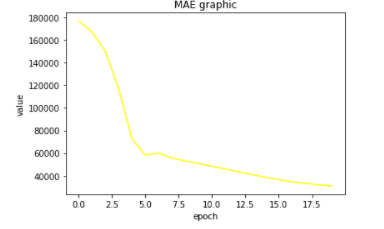
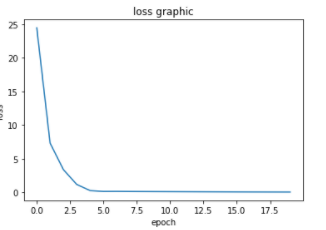
****

**2) epochs = 15**

**** ****

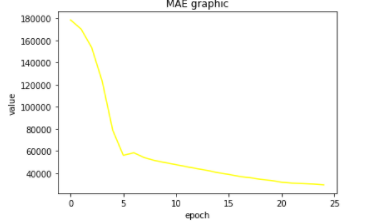
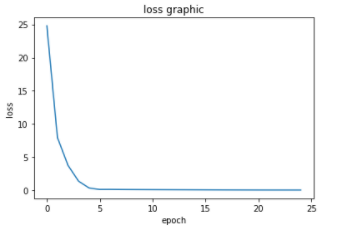
****

**2) epochs = 20**

**** ****

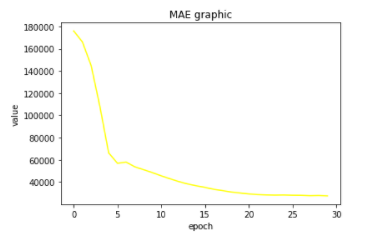
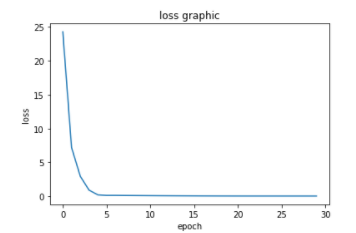
****

**2) epochs = 25**

**** ****

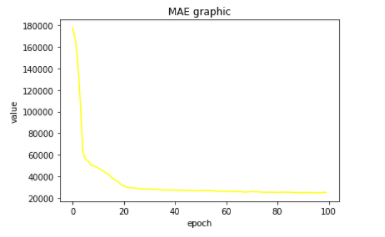
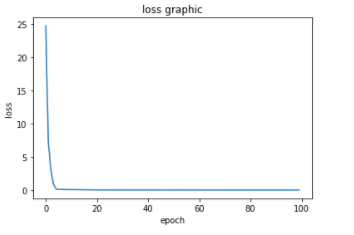
****

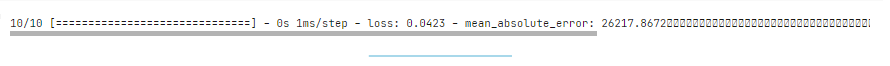
**3) epochs =30**

**** ****

****

**4 \*) epochs = 100**

**** ****

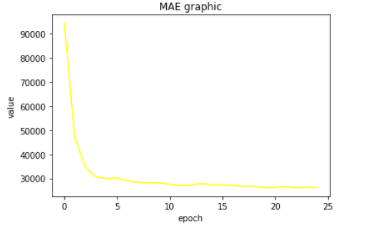
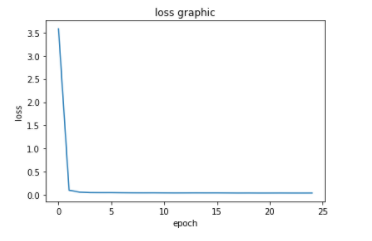
****

**Наилучшим показателем из выборки 10, 15, 20, 25, 30 оказалось epochs = 25**

**(loss = 0.039, mae = 27516,06)**

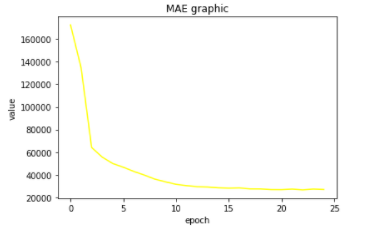
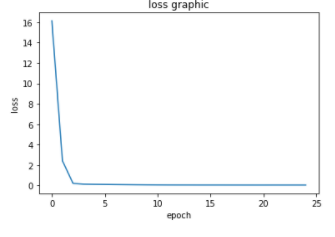
**Проверим разные размеры мини-выборки (batch\_size): 10, 50, 100, 200.**

1) batch\_size = 10

** **

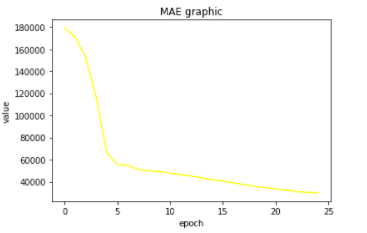
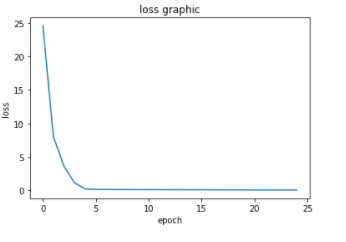
****

**2)** batch\_size = 50

** **

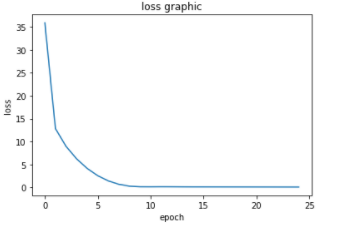
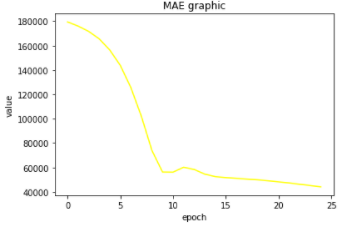
****

**3)** batch\_size = 100

**** ****

****

**4)** batch\_size = 200

****

****

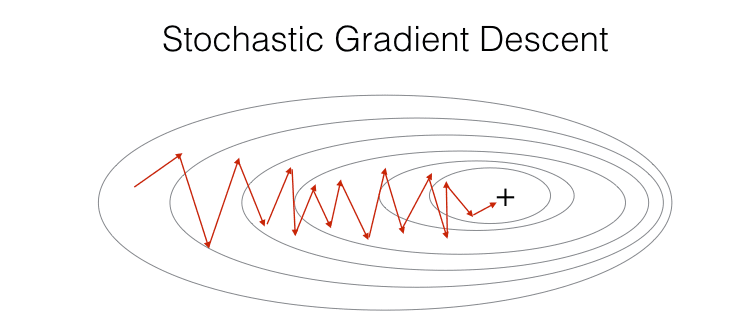
**Наилучшие показатели при**  **batch\_size = 10**

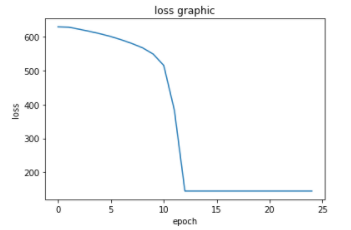
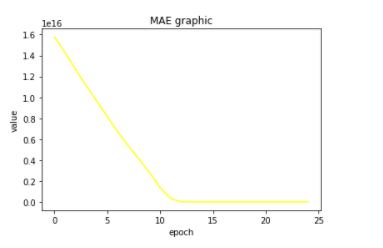
loss: 0.0428 mae: 24743.36

**Теперь попробуем поменять optimizer**

**1) SGD (стохастический градиентный спуск)**

Стохастический градиентный спуск похож на градиентный спуск(алгоритм оптимизации, используемый для минимизации некоторой функции путем итеративного движения в направлении самого крутого спуска, определяемого отрицательным значением градиента. ), только SGD стремится сбалансировать устойчивость стохастического градиентного спуска и эффективность пакетного градиентного спуска. Это наиболее распространенная реализация градиентного спуска, используемая в области глубокого обучения.



****

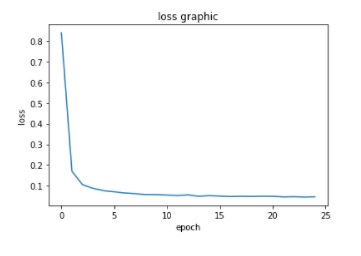
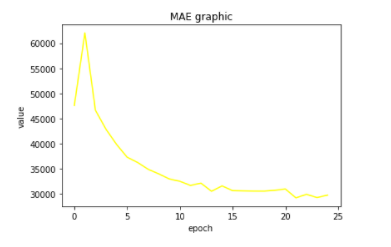
****

**2) RMSprop – среднеквадратичное распространение (должен быть вид затухающего колебания).**

Среднеквадратичное распространение корня (RMSprop) — это экспоненциально затухающее среднее значение. Существенным свойством RMSprop является то, что вы не ограничены только суммой прошлых градиентов, но вы более ограничены градиентами последних временных шагов. RMSprop вносит свой вклад в экспоненциально затухающее среднее значение прошлых «квадратичных градиентов». В RMSProp мы пытаемся уменьшить вертикальное движение, используя среднее значение, потому что они суммируются приблизительно до 0, принимая среднее значение. RMSprop предоставляет среднее значение для обновления.

**Изображение выглядит как текст

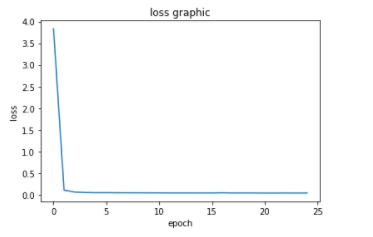
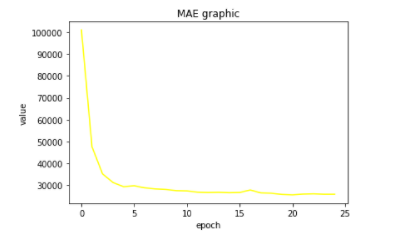
Автоматически созданное описание**

****

****

**3) Adam**

Adam — adaptive moment estimation.(Сочетание RMSprop и оптимизатора импульса) Он сочетает в себе и идею накопления движения и идею более слабого обновления весов для типичных признаков.

****

****

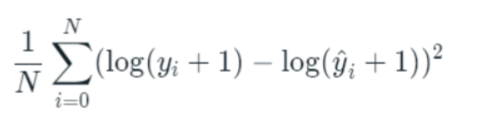
**Наилучший результат – optimizer Adam (**loss: 0.037 MAE: 24075.757**)**

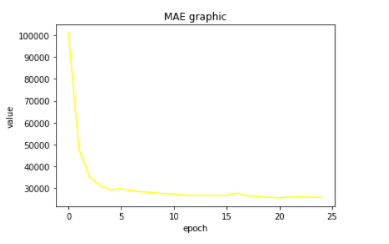
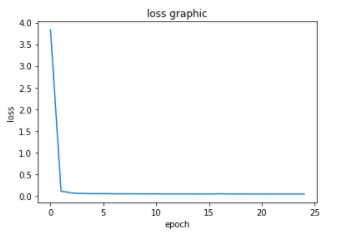
**Теперь попробуем поменять loss – функцию ошибок**

loss используется для обновления модели во время обратного распространения (нужно для оптимизатора),

в то время как metrics необходимо для оценки модели (насколько хорошо отработала).

**1) MSLE** – Средняя квадратичная логарифмическая ошибка

****

**** ****

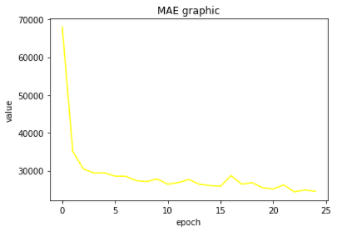
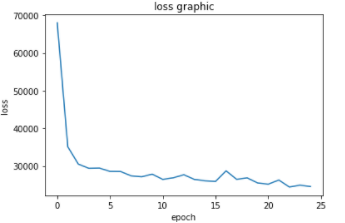
****

**2) MAE**

**Изображение выглядит как текст

Автоматически созданное описание**

Средняя абсолютная ошибка (MAE) – это усреднённая сумма модулей разницы между реальным и предсказанным значениями. MAE во многом похожа на MSE, но она отличается меньшей чувствительностью к выбросам значений (так как не берётся квадрат отклонения).

**** ****

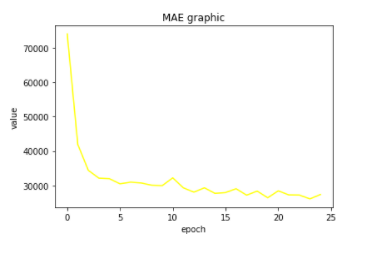
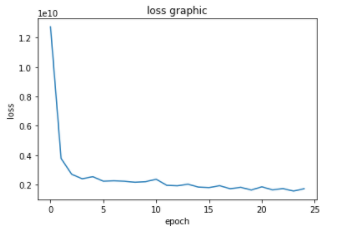
****

**3) MSE**

**Изображение выглядит как текст

Автоматически созданное описание**

Для каждой точки вычисляется квадрат отклонения, после чего полученные значения суммируются и делятся на общее количество точек. Чем ближе полученное значение к нулю, тем точнее наша модель. Данный метод расчёта в значительной мере чувствителен к выбросам в выборке, или к выборкам где разброс значений очень большой. В основном, данная функция применяется для переменных, распределение которых близко к распределению Гаусса.

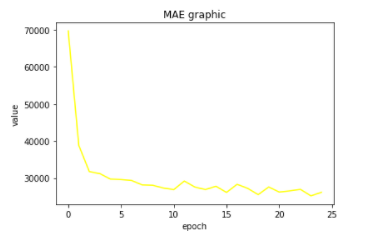
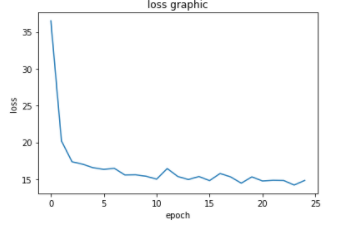
**** ****

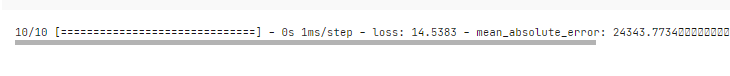
****

**4)MAPE** (MeanAbsolutePercentageError)

**Изображение выглядит как текст, часы

Автоматически созданное описание**

**** ****

****

**Наилучшая функция ошибок – MSLE**

Loss = 0.037, MAE = 24075,76

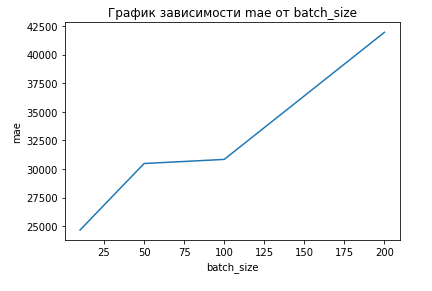
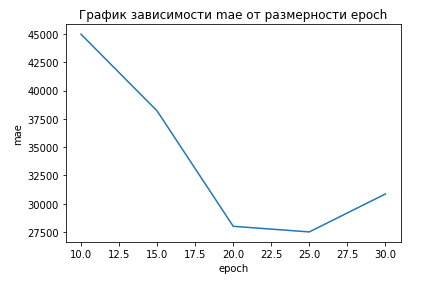
**Выводы и ответы на вопросы:**

В ходе подбора гипермараметров нейронной сети мне удалось достичь следующих показателей mae и loss:

**Loss = 0.037, MAE = 24075,76**

Этот результат достигается при следующих значениях параметров сети :

* Кол-во нейронов на входном слое: 200
* Кол-во нейронов в скрытом слое: 200
* Кол- во скрытых слоев: 3
* Функции активации: ReLu (кроме выходного слоя, где использовано ‘linear’)
* Epoch: 25
* Batch\_size: 10
* Optimizer : adam
* Loss: MSLE
* **Как выше перечисленные параметры влияют на полученный вами результат?**



* **Что такое эпоха (Epoch)? В чем отличие от итерации (Iteration)?**

Итерация - это своеобразный счетчик, который увеличивается каждый раз, когда нейронная сеть проходит один тренировочный сет. Эпоха увеличивается каждый раз, когда мы проходим весь набор тренировочных сетов. Сначала n раз увеличивается итерация, а потом уже эпоха.

* **Что такое функция активации? Какие вам известны?**

**Функция активации – это такая функция, которая**  определяет выходное значение нейрона в зависимости от результата взвешенной суммы входов и порогового значения. (Своебразная надстройка над весами, чтобы вместить значение в дипазон , обычно от 0 до 1)

Мне известны linear function, sigmoid, tanh, ReLU, Leaky ReLu, MaxOut

* **Что такое MSE(Mean Squared Error) - Средняя квадратичная ошибка? Что такое MAE(Mean Absolute Error)? Для чего используются.**

MAE во многом похожа на MSE, но она отличается меньшей чувствительностью к выбросам значений (так как не берётся квадрат отклонения).

MSE в значительной мере чувствителен к выбросам в выборке, или к выборкам где разброс значений очень большой. В основном, данная функция применяется для переменных, распределение которых близко к распределению Гаусса.