Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

**ОТЧЕТ ПО САМОСТОЯТЕЛЬНОЙ РАБОТЕ**

Регрессионный анализ на многослойном персептроне

тема

Преподаватель \_\_\_\_\_\_\_\_\_\_\_ В.В. Тынченко

подпись, дата инициалы, фамилия

Студент КИ15–16Б, 031510065 \_\_\_\_\_\_\_\_\_\_\_ Т.В. Радионов

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2018

# Цель исследования

Цель данного задания провести регрессионный анализ на многослойном персептроне с одним скрытым слоем и обучающегося с учителем методом обратного распространения ошибки и его модификации путем разработки его программной модели.

# Теория

Цель задачи регрессии — предсказать значение числовой переменной (обычно называемой зависимой) на основе значений одной или более переменной-предикторов (независимых переменных), которые могут быть либо числовыми, либо категориальными. Например, возможно, вы хотите спрогнозировать годовой доход некоей персоны, исходя из ее возраста, пола (мужчина или женщина) и уровня образования.

Самая простая форма регрессии называется линейной (linear regression, LR). LR-уравнение прогнозирования может выглядеть так: доход = 17.53 + (5.11 \* возраст) + (–2.02 \* мужчина) + (–1.32 \* женщина) + (6.09 \* образование). Хотя LR полезна для некоторых задач, во многих ситуациях она не эффективна. Но существуют другие распространенные типы регрессии: полиномиальная (polynomial regression), регрессия общей линейной модели (general linear model regression) и регрессия нейронной сети (neural network regression, NNR). Пожалуй, последний тип регрессии является самой мощной формой.

Наиболее распространенный тип нейронной сети (neural network, NN) — тот, что предсказывает значение категориальной переменной. Например, спрогнозировать политические предпочтения некоей персоны (консерватор, умеренный, либерал) на основе таких факторов, как возраст, доход и пол. NN-классификатор имеет n выходных узлов, где n — количество значений, которое может принимать зависимая переменная. Значения n выходных узлов суммируются до 1.0, и их можно вольно трактовать как вероятности. Поэтому для прогноза политических предпочтений NN-классификатор должен был бы иметь три выходных узла. Если бы значения выходных узлов были (0.24, 0.61, 0.15), NN-классификатор предсказал бы «умеренный», поскольку у среднего узла оказалась самая большая вероятность.

Большинство основных различий между нейронной сетью, рассчитанной на классификацию, и той, которая предназначена для регрессии, наблюдается в методах, вычисляющих вывод и обучающих модель.

В классификаторе окончательные значения выходного узла вычислялись бы применением функции активации к каждой накопленной сумме. В случае регрессии никакая функция активации не применяется. Поэтому метод завершается простым копированием значений из сумматоров непосредственно в выходные узлы. [1]

Нейронную сеть (далее – НС) можно представить в качестве взвешенного направленного графа, узлы (нейроны) которого расположены слоями. Кроме того, узел одного слоя имеет связи со всеми узлами предыдущего слоя (рисунок 1).

Каждый узел (нейрон) скрытого слоя обладает активационной функцией — функцией, ответственной за вычисление сигнала на выходе узла. Каждый нейрон выходного слоя не использует активационную функцию. Также существует понятие смещения, представляющего из себя узел, на выходе которого всегда появляется единица.

Выбор начальных весов окажет влияние на то, сумеет ли сеть достичь глобального (или только локального) минимума ошибки, и насколько быстро этот процесс будет происходить. Изменение весов между двумя нейронами связано с производной активационной функции нейрона из последующего слоя и активационной функции нейрона слоя предыдущего. В связи с этим, важно избегать выбора таких начальных весов, которые обнулят активационную функцию или ее производную. Также начальные веса не должны быть слишком большими (или входные сигнал для каждого скрытого или выходного нейрона скорее всего попадут в регион очень малых значений сигмоида (регион насыщения)). С другой стороны, если начальные веса будут слишком маленькими, то входной сигнал на скрытые или выходные нейроны будет близок к нулю, что также приведет к очень низкой скорости обучения. Стандартная процедура инициализации весов состоит в присвоении им случайных значений в интервале (-0,5; 0,5). Значения могут быть как положительными, так и отрицательными, так как конечные веса, получающиеся после обучения сети, могут быть обоих знаков.

В случае с большинством нейронных сетей цель состоит в обучении сети таким образом, чтобы достичь баланса между способностью сети давать верный отклик на входные данные, использовавшиеся в процессе обучения (запоминания), и способностью выдавать правильные результаты в ответ на входные данные, схожие, но неидентичные тем, что были использованы при обучении (принцип обобщения). Обучение сети методом обратного распространения ошибки включает в себя три этапа: подачу на вход данных, с последующим распространением данных в направлении выходов, вычисление и обратное распространение соответствующей ошибки, и корректировку весов. После обучения предполагается лишь подача на вход сети данных и распространение их в направлении выходов. При этом, если обучение сети может являться довольно длительным процессом, то непосредственное вычисление результатов обученной сетью происходит очень быстро. Кроме того, существуют многочисленные вариации метода обратного распространения ошибки, разработанные с целью увеличения скорости протекания процесса обучения.

Также стоит отметить, что однослойная нейронная сеть существенно ограничена в том, обучению каким шаблонам входных данных она подлежит в то время, как многослойная сеть (с одним или более скрытым слоем) не имеет такого недостатка [2].

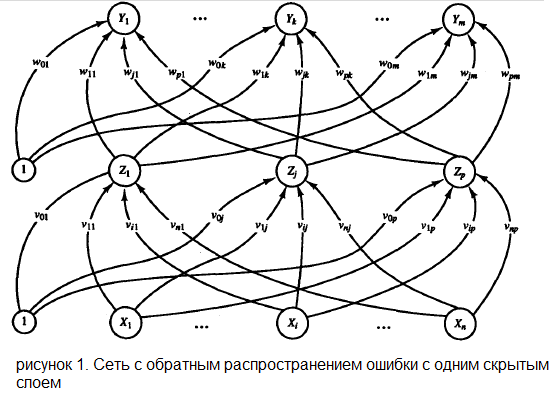


Рисунок 1 – Сеть с обратным распространением ошибки с одним скрытым слоем

Нормализация входных данных – это процесс, при котором все входные данные проходят процесс "выравнивания", т.е. приведения к интервалу [0,1] или [-1,1]. Если не провести нормализацию, то входные данные будут оказывать дополнительное влияние на нейрон, что приведет к неверным решениям.

В общем виде формула нормализации выглядит так:

|  |  |
| --- | --- |
|  | (1) |

где:

* – значение, подлежащее нормализации;
* – интервал значений х;
* – интервал, к которому будет приведено значение. [3]

В качестве активационной функции в многослойных персептронах, как правило, используется сигмоидальная активационная функция, в частности логистическая:

|  |  |
| --- | --- |
|  | (2) |

Алгоритм обратного распространения ошибки следующий:

1. Инициализировать синаптические веса маленькими случайными значениями.
2. Выбрать очередную обучающую пару из обучающего множества; подать входной вектор на вход сети.
3. Вычислить выход сети.
4. Вычислить разность между выходом сети и требуемым выходом (целевым вектором обучающей пары).
5. Подкорректировать веса сети для минимизации ошибки (как см. ниже).
6. Повторять шаги с 2 по 5 для каждого вектора обучающего множества до тех пор, пока ошибка на всем множестве не достигнет приемлемого уровня.

Рассмотрим подробней 5 шаг – корректировка весов сети. Здесь следует выделить два нижеописанных случая.

*Случай 1. Корректировка синаптических весов выходного слоя*

Например, для модели нейронной сети на рис. 1, это будут веса имеющие следующие обозначения:  и . Определимся, что индексом  будем обозначать нейрон, из которого выходит синаптический вес, а  – нейрон в который входит (рисунок 2):

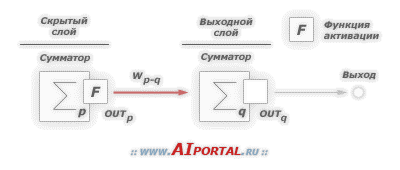


Рисунок 2 – Простая модель многослойной нейронной сети

Введем величину , которая равна разности между требуемым  и реальным  выходами, умноженной на 1, поскольку в NN-регрессии функция активации для выходных узлов не применяется:

|  |  |
| --- | --- |
|  | (3) |

Тогда, веса выходного слоя после коррекции будут равны:

|  |  |
| --- | --- |
|  | (4) |

где:

* – номер текущей итерации обучения;
* – величина синаптического веса, соединяющего нейрон  с нейроном ;
* – коэффициент «скорости обучения», позволяет управлять средней величиной изменения весов;
* – выход нейрона .

*Случай 2. Корректировка синаптических весов скрытого слоя*

Определимся, что индексом  будем обозначать нейрон из которого выходит синаптический вес, а  – нейрон в который входит (обратите внимание на появление новой переменной ) (рисунок 3):

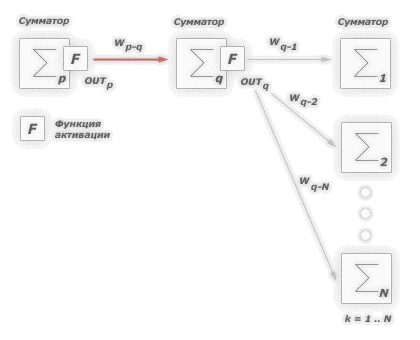


Рисунок 3 - Простая модель многослойной нейронной сети

Введем величину , которая равна:

|  |  |
| --- | --- |
|  | (5) |

где:

* – сумма от 1 по M.

Тогда, веса скрытых слоев после коррекции будут равны:

|  |  |
| --- | --- |
|  | (6) |

Для каждого нейрона в скрытом слое должно быть вычислено delta и подстроены все веса, ассоциированные с этим слоем. Этот процесс повторяется слой за слоем по направлению к входу, пока все веса не будут подкорректированы [4].

Нейрон смещения или bias нейрон — это третий вид нейронов, используемый в большинстве нейросетей (рисунок 4). Особенность этого типа нейронов заключается в том, что его вход и выход всегда равняются 1 и они никогда не имеют входных синапсов. Нейроны смещения могут, либо присутствовать в нейронной сети по одному на слое, либо полностью отсутствовать, 50/50 быть не может (красным на схеме обозначены веса и нейроны которые размещать нельзя). Соединения у нейронов смещения такие же, как у обычных нейронов — со всеми нейронами следующего уровня, за исключением того, что синапсов между двумя bias нейронами быть не может. Следовательно, их можно размещать на входном слое и всех скрытых слоях, но никак не на выходном слое, так как им попросту не с чем будет формировать связь.

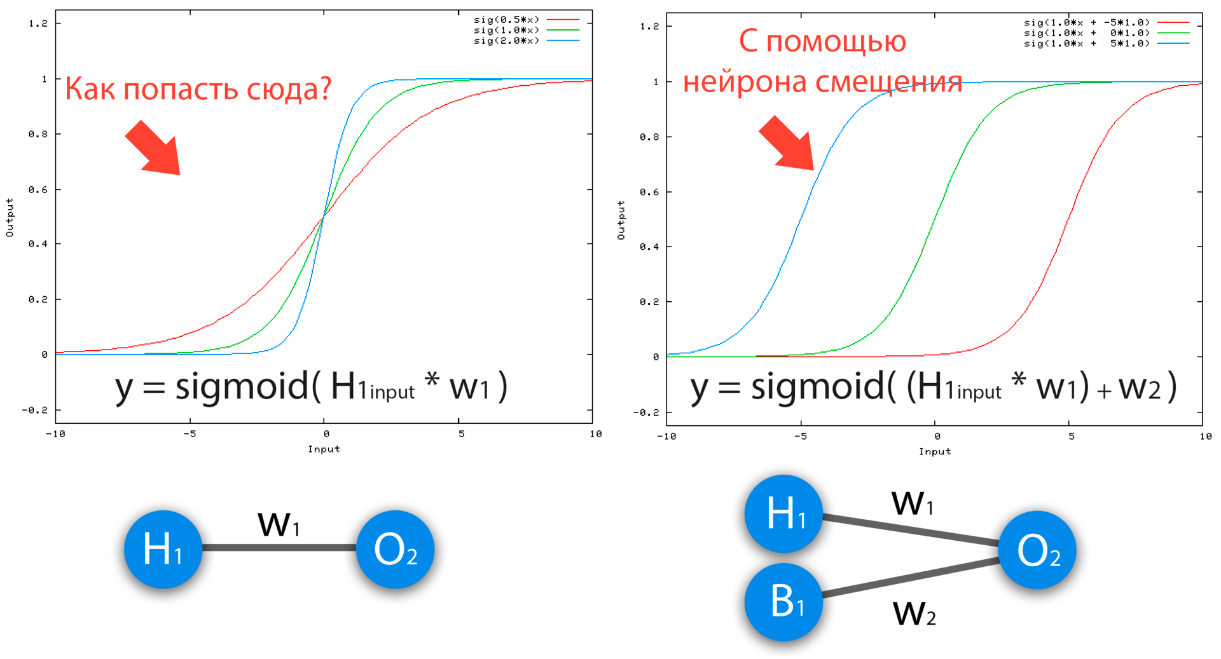


Рисунок 4 – Нейрон смещения

Существует метод ускорения обучения для алгоритма обратного распространения, увеличивающий также устойчивостьпроцесса. Этот метод, названный импульсом, заключается в добавлении к коррекции веса члена, пропорционального величине предыдущего изменения веса. Как только происходит *коррекция*, она "запоминается" и служит для модификации всех последующих коррекций. Уравнения коррекции модифицируются следующим образом:

|  |  |
| --- | --- |
|  | (7) |

где  — коэффициент импульса, который обычно устанавливается около 0,9.

Используя метод импульса, сеть стремится идти по дну "узких оврагов" поверхности ошибки (если таковые имеются), а не двигаться "от склона к склону". Этот метод, по-видимому, хорошо работает на некоторых задачах, но дает слабый или даже отрицательный эффект на других.

Если  равен 1,0, то новая коррекция игнорируется и повторяется предыдущая. В области между 0 и 1 коррекция веса сглаживается величиной, пропорциональной  *По*-прежнему,  является коэффициентом скорости обучения, служащим для управления средней величиной изменения веса.

Сходимость говорит о том, правильная ли архитектура НС и правильно ли были подобраны гиперпараметры в соответствии с поставленной задачей (рисунок 5). Допустим, программа выводит ошибку НС на каждой итерации в лог. Если с каждой итерацией ошибка будет уменьшаться, то мы на верном пути и наша НС сходится. Если же ошибка будет прыгать вверх — вниз или застынет на определенном уровне, то НС не сходится. В 99% случаев это решается изменением гиперпараметров. Оставшийся 1% будет означать, что ошибка в архитектуре НС. Также бывает, что на сходимость влияет переобучение НС [5].



Рисунок 5 – Сходимость НС

# Описание задачи

Для выполнения данного задания взята задача регрессии SGEMM GPU kernel performance [6] из базы данных сайта Machine Learning Repository [7].

Данные представляют из себя следующие входные и выходные параметры:

*вход*

* 1-2. MWG, NWG: per-matrix 2D tiling at workgroup level: {16, 32, 64, 128} (integer);
* 3. KWG: inner dimension of 2D tiling at workgroup level: {16, 32} (integer);
* 4-5. MDIMC, NDIMC: local workgroup size: {8, 16, 32} (integer);
* 6-7. MDIMA, NDIMB: local memory shape: {8, 16, 32} (integer);
* 8. KWI: kernel loop unrolling factor: {2, 8} (integer);
* 9-10. VWM, VWN: per-matrix vector widths for loading and storing: {1, 2, 4, 8} (integer);
* 11-12. STRM, STRN: enable stride for accessing off-chip memory within a single thread: {0, 1} (categorical);
* 13-14. SA, SB: per-matrix manual caching of the 2D workgroup tile: {0, 1} (categorical).

*выход*

* 15-18. Run1, Run2, Run3, Run4: performance times in milliseconds for 4 independent runs using the same parameters. They range between 13.25 and 3397.08.

Данные состоят из 241600 записей, из которых взяты 192001 в качестве обучающей выборки.

Активационная функция представляет из себя сигмоидальную функцию, нейрон смещения отсутствует

Инициализация начальных весов происходит случайным образом в диапазоне [-0.5; 0.5].

Обучаться НС будет c заданным в программе количеством шагов и скоростью обучения по алгоритму обратного распространения ошибки, описанного в теоретической части. Затем результат обучения проверяется полной выборке реальных данных. Основным результатом регрессии является текстовое сообщение с погрешностью, которая отражает среднее расхождение полученных данных от реальных (отдельно взятые погрешности данных просуммированы и поделены на количество данных). Стоит отметить, что это лишь способ оценки качества обучения нейросети. Отдельно взятые погрешности могут значительно отличаться от общей средней.

Также можно будет выполнить переобучение программы.

С целью исследования различных характеристик будет встроен в программу график с изменяющимися параметрами: средняя погрешность на общей выборке – ось Y, скорость обучения (eta) – ломаная на графике, количество шагов обучения (параметр N) – ось X. погрешность рассматривается в диапазоне [0; 1], для получения точного результата следует умножить на 100 и рассматривать результат в %; графиков eta будет 10 (от 0.1 до 1.0 с шагом 0.1), ось X будет в диапазоне от начального значения N до конечного (который можно изменить в программе). Также будет введен дополнительно параметр количества опытов M, который отвечает за количество повторений обучения с текущими значениями eta и N и выводом среднего значения средней погрешности (данный параметр позволит «сгладить» график eta – чем больше M, тем меньше вероятность ошибки при вычислении параметра error).

В результате можно будет проводить исследование НС путем изучения графиков и определения наиболее оптимальных параметров для программы с целью улучшить настройку НС.

# Ход работы

Необходимо установить оптимальные параметры для eta и количества шагов N. Для этого проведем ряд исследований НС и найдем параметры, при которых наблюдется минимальная средняя погрешность.

Стоит отметить, что при eta > 0.6 наступает паралич сети, т.к. значения весов становятся очень большими и активационная функция выдает «1» вне зависимости от того, что поступающие значения различны - этот диапазон не следует использовать при конфигурации НС.

Для первого опыта был взят диапазон N [10; 100] c шагом 10 при M = 10. Как видно из рисунка 6, сеть имела погрешность менее 12.2% при eta = 0.4

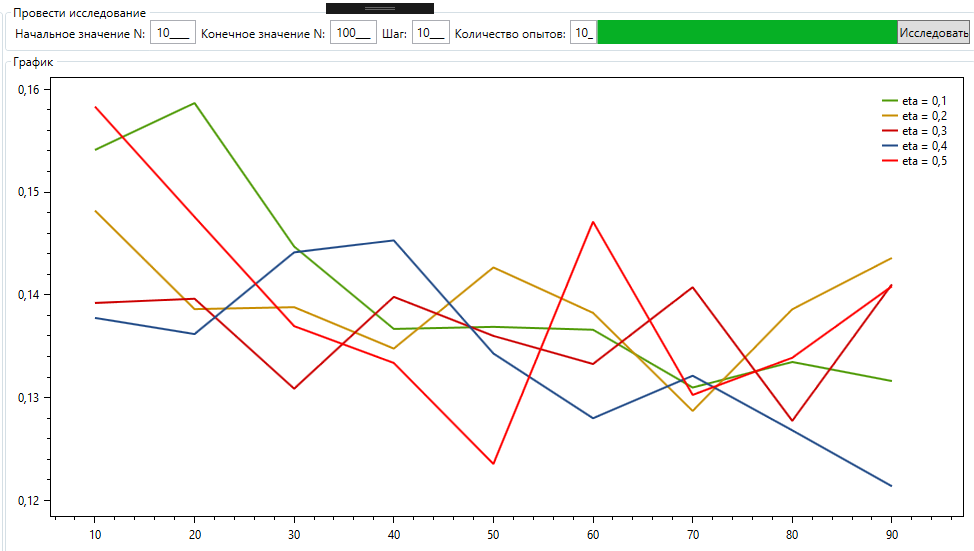


Рисунок 6 – Опыт 1

Для второго опыта был взят диапазон N [100; 500] c шагом 50 при M = 10. Как видно из рисунка 6, сеть имела погрешность менее 11.5% при всех eta кроме 0.1.

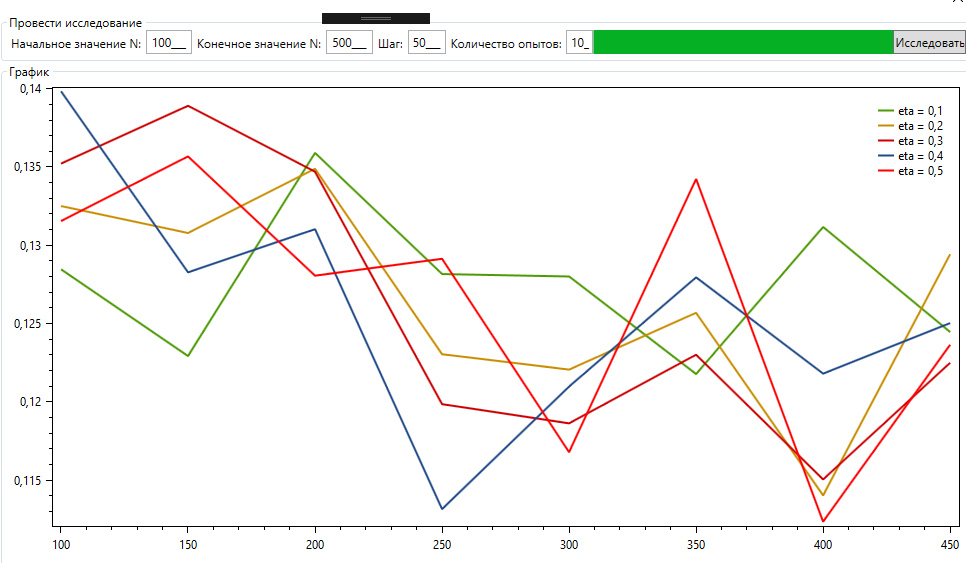


Рисунок 7 – Опыт 2

Для третьего опыта был взят диапазон N [500; 5000] c шагом 500 при M = 10. Как видно из рисунка 8, сеть имела погрешность менее 3% при всех eta = 0.4. Примечательно, что наблюдается сходимость сети.

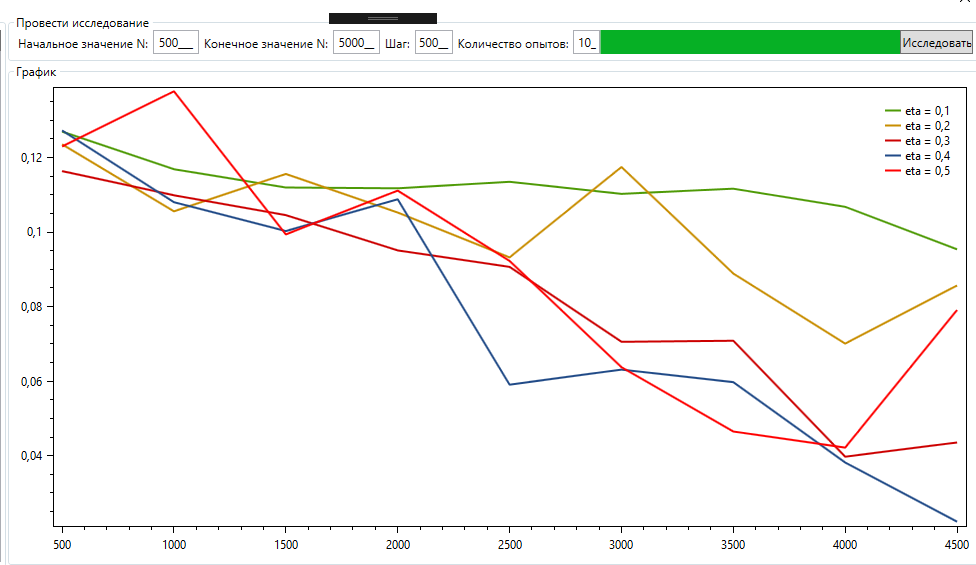


Рисунок 8 – Опыт 3

Исходя из предыдущих опытов делаем предположение, что оптимальным является eta = 0.4. Проверим это предположение на диапазоне N [0; 10000] c шагом 500 при M = 10. Предположение подтверждается частично на рисунке 9, где оптимальным eta оказался диапазон [0.2; 0.4] с погрешностью менее 2%. Примечательно, что при eta = 0.5 начиная с N = 7000 мы наблюдаем переобучение НС.

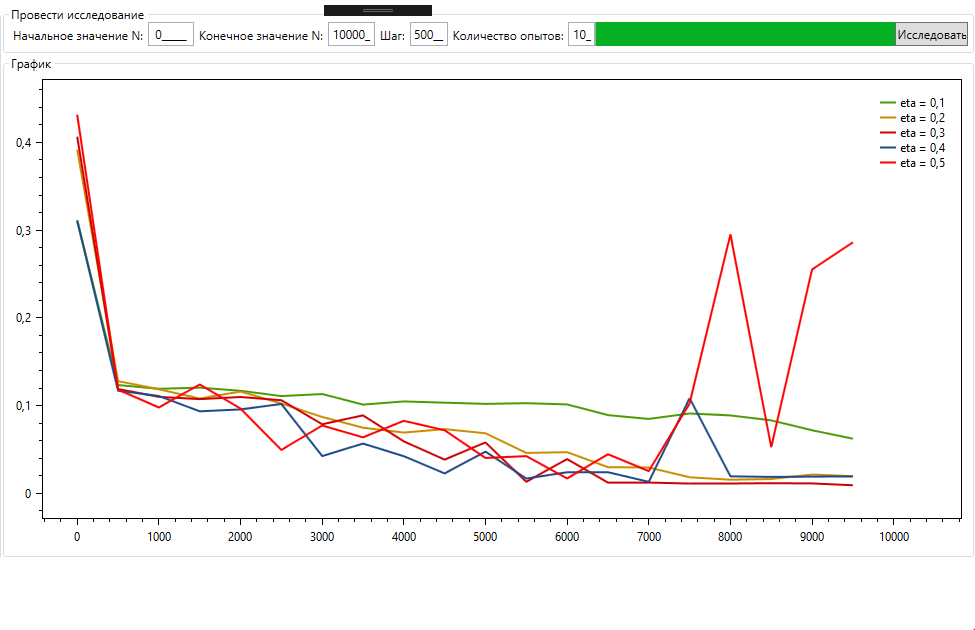


Рисунок 9 – Опыт 4

Теперь в пятом опыте сравним результаты обучения алгоритма обратного распространения ошибки и его модификации (рисунок 10) при найденном оптимальном eta = 0.3 и рекомендуемом значении (указанном в теоретической части) импульса mu = 0.9. Как видно, модификация в данной задаче не принесла лучших результатов, даже наоборот.

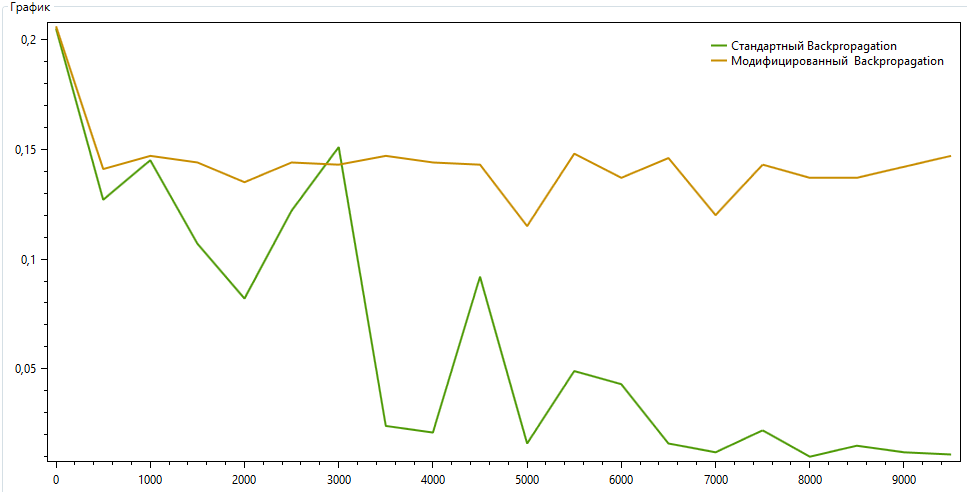


Рисунок 10 – Опыт 5

Подведя итоги исследования, можно предположить, что наименьшая погрешность в НС будет наблюдаться при eta в диапазоне [0.2; 0.4] и N в диапазоне [8000; 9000]. Также применение модификаций не обязательно сделают результаты лучше, как вышло это в опыте 5.

Теперь убедимся в том, что найденные параметры и выбор алгоритма в ходе исследования принесут наилучшие результаты: проведем исследование НС при оптимальных параметрах: N = 8500 и eta = 0.3 на стандартном алгоритме обратного распространения ошибки. Проведем 10 опытов и занесем в таблицу 1:

Таблица 1 – Исследование НС при оптимальных параметрах

|  |  |
| --- | --- |
| № опыта | Погрешность, % |
| 1 | 1.2 |
| 2 | 1 |
| 3 | 0.9 |
| 4 | 0.9 |
| 5 | 0.8 |
| 6 | 1.1 |
| 7 | 1 |
| 8 | 1.7 |
| 9 | 0.9 |
| 10 | 1.3 |

# Выводы

В ходе исследования были установлены оптимальные параметры для НС с поставленной задачей, на которых была достигнута минимальная погрешность в 0.8% предсказания результата. Также наблюдалась сходимость НС при увеличении количества опытов, а также и переобучение НС при дальнейшем увеличении.

Отдельный выход в НС и его сравнение с ожиданием и вычисление погрешности не проводилось в связи с их большим количеством, для этого была вычислена общая средняя погрешность для всех данных (что означает, что у отдельно взятого случая может быть погрешность как больше, так и меньше), однако это позволило оценить эффективность обучения и прогнозирования разработанной НС.

Разработанная НС довольно эффективно выполняет регрессионный анализ и делает точные предсказания результатов.

# Список использованных источников

x

|  |  |
| --- | --- |
| 1. | Тесты - Регрессия нейронной сети [Электронный ресурс] // Microsoft magazine: [сайт]. [2016]. URL: https:/​/​msdn.microsoft.com/​ru-ru/​magazine/​mt683800.aspx (дата обращения: 10.05.2018). |
| 2. | Алгоритм обучения многослойной нейронной сети методом обратного распространения ошибки (Backpropagation) [Электронный ресурс] // habr: [сайт]. [2013]. URL: https:/​/​habr.com/​post/​198268/ (дата обращения: 25.04.2018). |
| 3. | НЕЙРОННЫЕ СЕТИ - ОТ ТЕОРИИ К ПРАКТИКЕ [Электронный ресурс] // MQL5.community: [сайт]. [2012]. URL: https:/​/​www.mql5.com/​ru/​articles/​497 (дата обращения: 10.05.2018). |
| 4. | Алгоритм обратного распространения ошибки [Электронный ресурс] // Портал искусственного интеллекта: [сайт]. URL: http:/​/​www.aiportal.ru/​articles/​neural-networks/​back-propagation.html (дата обращения: 25.04.2018). |
| 5. | Нейронные сети для начинающих. Часть 2 [Электронный ресурс] // habr: [сайт]. [2017]. URL: https:/​/​habr.com/​post/​313216/ (дата обращения: 25.04.2018). |
| 6. | Cedric Nugteren and Valeriu Codreanu. CLTune: A Generic Auto-Tuner for OpenCL Kernels. In: MCSoC: 9th International Symposium on Embedded Multicore/Many-core Systems-on-Chip. IEEE, 2015 URL: http://ieeexplore.ieee.org/document/7328205/. |
| 7. | SGEMM GPU kernel perfomance Data Set [Электронный ресурс] // Machine Learning Repository: [сайт]. [2018]. URL: https:/​/​archive.ics.uci.edu/​ml/​datasets/​SGEMM+GPU+kernel+performance (дата обращения: 10.05.2018). |

x