Практическая работа № 6 от 03.04.2020

Нейронная сеть

Цель работы

Использование нейронной сети для решения задачи регрессии.

Задачи работы

1. Изучить основы библиотеки keras.
2. Научиться создавать и использовать нейронную сеть.
3. Научиться анализировать и оценивать результаты работы нейронной сети.

Перечень обеспечивающих средств

1. ПК.
2. Учебно-методическая литература.
3. Задания для самостоятельного выполнения.

Общие теоретические сведения

В данном задании мы работаем с набором данных, описывающим параметры центральных процессоров и их производительность. Набор данных имеет следующую структуру (https://github.com/mosalov/DataPreparation):

1. Vendor name – название производителя процессора.
2. Model Name – названи модели процессора.
3. MYCT – машинный цикл в наносекундах.
4. MMIN – минимальный размер памяти в килобайтах.
5. MMAX – максимальный размер памяти в килобайтах.
6. CACH – размер кэша в килобайтах.
7. CHMIN – минимальное количество каналов.
8. CHMAX – максимальное количество каналов.
9. PRP – опубликованная относительная производительность.
10. ERP – относительная производительность, рассчитанная в статье по ссылке.

В предыдущем задании мы обработали этот набор данных, проведя очистку и преобразование каждого столбца данных, за исключением «Model Name» и «ERP».

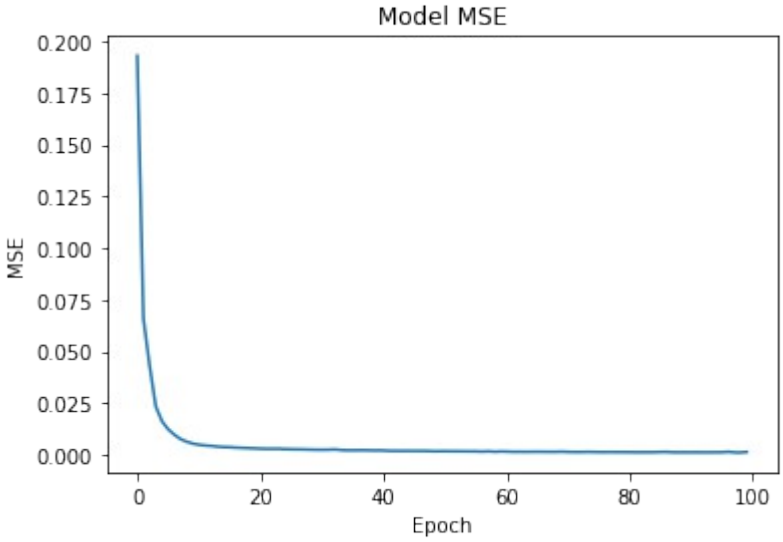
Мы предполагаем, что производительность процессора (значение параметра «PRP») представляет собой функцию от остальных параметров (1 и 3-7). Однако аналитический вид данной функции нам не известен.

Для того, чтобы апроксимировать эту функцию, можно использовать простейшую нейронную сеть. На вход этой сети будем подавать значения параметров 1 и 3-7, а на выходе получать значение параметра 9. Это задача регрессии, т. к. мы вычисляем непосредственно значение, а не принадлежность к некоторому классу.

Для оценки качества работы сети мы используем изначально выделенную тестовую выборку. Правильно обученная нейронная сеть показывает «разумную» ошибку на тестовой выборке. Критерий разумности тут, к сожалению, только эмпирический, т.е. определяется экспериментально, и, в общем случае, зависит от задачи.

Тем не менее, можно сформулировать примерный порядок проверки обученной нейронной сети и рекомендации по ее дальнейшей настройке.

Проверка №1
Строим графики средней абсолютной и среднеквадратической ошибки для обучающей выборки. Они должны сначала убывать с ростом номера эпохи обучения, а затем выходить на постоянный, невысокий уровень. Если этого не происходит, значит структура сети или параметры обучения не позволяют процессу сойтись.
Ожидаемый вид графика:



Если ошибка убывает, но не выходит на стабильный уровень – можно увеличить количество эпох обучения (чтобы обучение происходило дольше) или уменьшить размер батча (чтобы обчение происходило чаще).

Если ошибка убывает быстро, график идет практически вертикально, то имеет смысл уменьшить количество эпох обучения (чтобы обучение заканчивалось быстрее) или увеличить размер батча (чтобы обучение происходило реже).

Если ошибка не убывает либо не стабилизируется можно попробовать увеличить время обучения (число эпох) или усложнить структуру сети (количество скрытых слоев и нейронов в них). В крайнем случае можно заменть алгоритм оптимизации функции ошибки.

Проверка №2

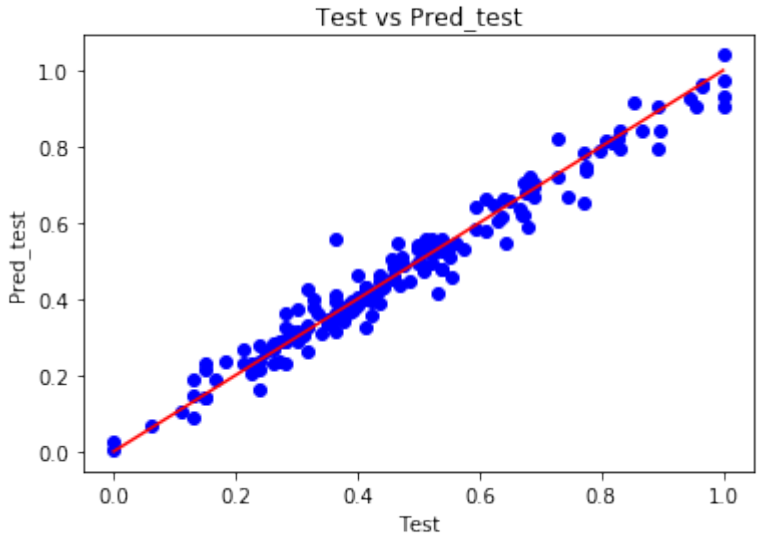
Строим графики средней абсолютной и среднеквадратической ошибки для тестовой выборки. Они должны иметь ту же форму, что и для обучающей выборки, но постоянный уровень ошибки на тестовой выборке должен быть хотя бы немного, но выше постоянного уровня ошибки на обучающей выборке. Это логично, ведь для обучения сеть не видела тестовую выборку и должна ошибать на ней хоть не намного, но чаще.

Если форма графика ошибки на тестовой выборке не совпадает с формой графика на обучающей, значит, произошло так называемое «переобучение», т.е. сеть просто запомнила все варианты из обучающей выборки, а не извлекал из них некие зависимости. В таком случае можно попробовать уменьшить время обучения (число эпох) и упростить структуру сети (уменьшить количество скрытых слоев и нейронов в них). Если это не приводит к результату нужно пересмотреть разбивку на обучающую и тестовую выборки. Увеличение обучающей выборки дает сети больше возможности обучиться, увеличение тестовой выборки приводит к более надежным результатам проверки.

Проверка №3

Строим график отношения предсказанного и целевого значений для обучающей выборки. В предельном случае все точки будут лежать на прямой y = x, однако это почти наверняка будет означать, что сеть переобучилась. В идеале точки должны лежать «вдоль» этой прямой, но с небольшими отклонениями.

Ожидаемый вид графика:



Если точки лежат слишком близко к прямой – «переобучение», действия аналогичны описанным в проверке №2.

Если точки лежат слишком «в разброс», значит сеть недостаточно обучена, нужно либо упростить ее структуру, либо увеличить время обучения.

Проверка №4
Строим график отношения предсказанного и целевого значений для тестовой выборки. Аналогично, в идеале точки должны лежать «вдоль» этой прямой, но с небольшими отклонениями.

Если точки не лежат «вдоль» прямой, значит сеть «переобучена» на обучающей выборке, действия аналогичны описанным в проверке №2.

Проверка №5
Цель – убедиться, что отклонения предсказанных значений от целевых распределены нормально.
Что это значит – если отклонения распределены нормально, значит полученная апроксимация функции (наша нейронная сеть) одинаково применима ко всему множеству значений и может быть использована без дополнительных ограничений. Наличие «выбросов» говорит о том, что для некоторых интервалов значений сеть допускает более крупные ошибки, чем для остальных. Это нужно учитывать при применении сети в качестве апроксимации функции.
Как проверить распределение отклонений на нормальность – рассмотрим три варианта (в коде показаны примеры реализации для всех трех вариантов):

1. Построить гистрограмму отклонений и проверить, что она напоминает «колокол».
2. Построить квантиль-квантиль график. Для нормального распределения точки должны лежать на прямой.
3. Применить один из статистических тестов и оценить полученные p-значение. Оно должно быть заметно выше нуля (не будем здесь погружаться в математику, объясняющую что это и почему так работает).

Если распределение далекот от нормального – полученая нейронная сеть не справляется со своей задачей. «Простое» решение – увеличить размер выборки, но это не всегда возможно. Более сложный вариант – провести анализ предварительной обработки данных и постараться по-максимуму убрать из данных выбросы и отклонения от норамального и равномерно распределений.

Если распределение похоже на нормальное визуально, но статистические тесты его отвергают, можно проигнорировать это. Так довольно часто делаеют в реальных исследованиях, хотя это и остается на совести исследователя.

Дополнительные теоретические сведения и пояснения по используемым программным инструментам для данного задания даны в лекции №2 «Нейронные сети» и в комментариях к коду в репозитории: https://github.com/mosalov/SimpleNeuralNetworkWithKeras.

Задание

- Сделайте форк репозитория https://github.com/mosalov/SimpleNeuralNetworkWithKeras.
- Добавьте в полученный репозиторий файл с данными, обработанными в предыдущем задании.
- Откройте репозиторий в Binder(https://mybinder.org/) и запустите файл «machine_data_analysis.ipynb».
- Следуйте представленному в файле коду, изменяя его при необходимости.
- Проанализируйте полученную нейронную сеть, сохраните ее и загрузите файл в репозиторий.
- Создайте файл отчёта и внесите внего результаты анализа в виде результатов проверок, перечисленных в теоретической части.
- Внесите изменения в структуру нейронной сети (количество скрытых слоев и количество нейронов в них), проанализируйте результат, внесите результаты проверок в файл отчёта. Сохраните сеть, загрузите файл в репозиторий.
- Исключите 2-3 входных параметра по своему выбору. Создайте и обучите нейронную сеть. Проанализируйте результат, внесите результаты проверок в файл отчёта. Сохраните сеть, загрузите файл в репозиторий.
- Сохраните файл Jupyter notebook с названием «Задание 6.ipynb» и загрузите его в репозиторий.

Требования к предоставлению результатов

Требуется представить результаты в виде письма на адрес mosalov.op@ut-mo.ru с указанием ФИО и ссылки на репозиторий с сохраненным файлом Jupyter notebook, файлами нейронных сетей и файлом отчёта.

Литература

1. https://developers.google.com/machine-learning/data-prep/transform/introduction
2. http://blog.datalytica.ru/2018/04/blog-post.html