Существует две гипотезы:

1. Вычислить фракционный резерв через скорость пульсовой волны
2. Вычислить фракционный резерв через снимки КТ

На основе данных собрать модель и исследовать эффективность по отдельности и в совокупности двух моделей

Сделать вывод и предложить решение на основании исследования

Структура А) и Б):

- Исследование данных

- Визуализация

- Разработка модели

+ сравнение точности решения нейронных сетей двух моделей

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Гриф: \_\_\_\_\_\_  Экз. № \_\_\_\_\_\_\_\_\_\_   |  | | --- | |  | | ПАО «Компания «Сухой» - ОКБ Сухого |   Тема: НИР  **НАУЧНО-ТЕХНИЧЕСКИЙ ОТЧЕТ**  По графику  **« »**   |  |  | | --- | --- | | Начальник ОВВ НТЦ СКТ | А.В. Корнев | | Начальник бр. ЭММ ОВВ НТЦ СКТ | И.Д. Танненберг | | Инженер | И.А. Ниженко |     Москва 2019 г. |

**Оглавление**

[Постановка задачи 4](#_Toc37272783)

[Решение задачи 4](#_Toc37272784)

[Схема решения задачи №1 5](#_Toc37272785)

[Обработка входных данных 6](#_Toc37272786)

[Описание структуры нейронной сети 7](#_Toc37272787)

[Процедура поиска гиперпараметров модели машинного обучения 8](#_Toc37272788)

[Нейронная сеть без регуляризации 11](#_Toc37272789)

[Использование XGBoost 12](#_Toc37272790)

[Применение методик регуляризации для уменьшения дисперсии 13](#_Toc37272791)

[L1 - регуляризация 13](#_Toc37272792)

[L2 - регуляризация 13](#_Toc37272793)

[Dropout - регуляризация 14](#_Toc37272794)

[Результат сравнения применения нейронных сетей с другими методами регрессионного анализа 17](#_Toc37272795)

[Схема решения задачи №2 18](#_Toc37272796)

[Использованные источники 19](#_Toc37272797)

[Приложение А 20](#_Toc37272798)

[Основной скрипт 20](#_Toc37272799)

# Постановка задачи

Разработать модель нейронной сети с применением способов повышения точности расчета для определения фракционного резерва коронарных сосудов человека по скорости пульсовой волны и снимкам компьютерной томографии сердца человека. В ходе разработки модели решить следующие задачи:

1. Провести обработку входных данных для скорости пульсовой волны;
2. Провести обработку входных данных для снимков компьютерной томографии коронарных сосудов человека;
3. Построить нейронную сеть для определения задачи 1 и задачи 2 по оценке критериев по существующей базе данных;
4. Провести ряд мероприятий по увеличению точности решения нейронных сетей;

# Решение задачи

Решается задача определения результирующего параметра, по исходным данным, которые расположены в свободном доступе [1].

Цель текущей задачи – интерпретировать данные, решить её с помощью нейронной сети и исследовать полученные данные.

# Схема решения задачи №1

И.Д. (Исходные данные):

1. Семантические
2. Логические
3. Целые числа

Статистическая обработка данных:

1. Обучить Ω -> РЕЗ.

2. Оценить результирующий параметр S.

Рисунок 1 – Схема решения задачи

## Обработка входных данных

Обработка входных данных осуществляется в несколько этапов:

1. Формулирование вопроса
2. Сбор данных
3. Чистка данных
4. Исследование и визуализация данных
5. Тренировка алгоритма
6. Оценка результатов

Методы **сбора данных** часто плохо контролируются, что приводит к недопустимым значениям (таким как, скорость: −100), невозможным комбинациям данных (таким как, пол: Мужской, Беременность: Да), отсутствующим значениям и прочее. При анализе данных, не защищённом от такого рода проблем, можно прийти к неверным выводам. Таким образом, представление данных и их качество являются первостепенной заботой перед осуществлением анализа. Часто предварительная обработка данных является наиболее важной фазой проекта обучения машины, особенно в вычислительной биологии.

Если есть много лишней информации или зашумлённых и недостоверных данных, то извлечение знаний во время тренировки становится затруднительным. Шаг подготовки и фильтрации данных может занять существенное время. Предварительная подготовка данных включает

* очистку
* отбор экземпляров
* нормализацию
* преобразование данных
* выделение признаков
* отбор признаков

Результатом предварительной обработки данных является конечный тренировочный набор.

## Описание структуры нейронной сети

Нейронная сеть, которая используется в алгоритме кода имеет следующую структуру (таблица Таблица 2):

Таблица – Структура нейронной сети

|  |  |
| --- | --- |
| Функция активации | ReLU |
| Количество входных параметров | 7 |
| Количество выходных параметров | 1 |

## Процедура поиска гиперпараметров модели машинного обучения

Оптимизация гиперпараметров — задача машинного обучения по выбору набора оптимальных гиперпараметров для обучающего алгоритма.Одни и те же виды моделей машинного обучения могут требовать различные предположения, веса или скорости обучения для различных видов данных.

Эти параметры называются гиперпараметрами и их следует настраивать так, чтобы модель могла оптимально решить задачу обучения. Для этого находится кортеж гиперпараметров, который даёт оптимальную модель, оптимизирующую заданную функцию потерь на заданных независимых данных. Целевая функция берёт кортеж гиперпараметров и возвращает связанные с ними потери. Часто используется перекрёстная проверка для оценки этой обобщающей способности.

**Случайный поиск**

Случайный поиск заменяет полный перебор всех комбинаций на выборку их случайным образом. Это можно легко применить к дискретным установкам, приведённым выше, но метод может быть также обобщен к непрерывным и смешанным пространствам. Случайный поиск может превзойти поиск по решётке, особенно в случае, если только малое число гиперпараметров оказывает влияние на производительность алгоритма обучения машины. В этом случае говорят, что задача оптимизации имеет низкую внутреннюю размерность. Случайный поиск также легко параллелизуем и, кроме того, позволяют использовать предварительные данные путём указания распределения для выборки случайных параметров.

**Байесовская оптимизация**

Байесовская оптимизация — это метод глобальной оптимизации для неизвестной функции (чёрного ящика) с шумом. Применённая к гиперпараметрической оптимизации байесовская оптимизация строит стохастическую модель функции отображения из значений гиперпараметра в целевую функцию, применённую на множестве проверки. Путём итеративного применения перспективной конфигурации гиперпараметров, основанной на текущей модели, а затем её обновления, байесовская оптимизация стремится собрать как можно больше информации об этой функции и, в частности, место оптимума. Метод пытается сбалансировать зондирование (гиперпараметры, для которых изменение наименее достоверно известно) и использование (гиперпараметры, которые, как ожидается, наиболее близки к оптимуму). На практике байесовская оптимизация показала лучшие результаты с меньшими вычислениями по сравнению с поиском по решётке и случайным поиском ввиду возможности суждения о качестве экспериментов ещё до их выполнения.

**Оптимизация на основе градиентов**

Для конкретных алгоритмов обучения можно вычислить градиент гиперпараметров и оптимизировать их с помощью градиентного спуска. Первое использование этих техник фокусировалось на нейронных сетях. Затем эти методы были распространены на другие модели, такие как методы опорных векторов или логистическая регрессия.

Другой подход использования градиентов гиперпараметров состоит в дифференцировании шагов алгоритма итеративной оптимизации с помощью автоматического дифференцирования.

**Эволюционная оптимизация**

Эволюционная оптимизация — это методология для глобальной оптимизации неизвестных функций с шумом. При оптимизации гиперпараметров эволюционная оптимизация использует эволюционные алгоритмы для поиска гиперпараметров для данного алгоритма. Эволюционная оптимизация гиперпараметров следует процессу, навеянному биологической концепцией эволюции:

1. Создаём начальную популяцию случайных решений (то есть сгенерированный случайно кортеж гиперпараметров, обычно 100+);
2. Оцениваем кортежи гиперпараметров и получаем их функцию приспособленности (например, с помощью 10-кратной точности перекрёстной проверки алгоритма обучения машины с этими гиперпараметрами);
3. Ранжируем кортежи гиперпараметров по их относительной пригодности;
4. Заменяем кортежи гиперпараметров с худшей производительностью на новые кортежи гиперпараметров, образованных скрещиванием.

Повторяем шаги 2—4, пока не получим удовлетворительной производительности алгоритма или пока производительность не перестанет улучшаться

Эволюционная оптимизация используется для оптимизации гиперпараметров для статистических алгоритмов обучения машин, автоматического обучения машин, для поиска архитектуры глубоких нейронных сетей, а также для формирования весов в глубоких нейронных сетях.

## Нейронная сеть без регуляризации

Для исследования эффективности регрессии нейронной сети без применения методик регуляризации использовался Talos, встроенный в Keras. В качестве набора гиперпараметров исследования использовались параметры, приведенные в таблице Таблица 3.

Таблица - Гиперпараметры исследования эффективности нейронной сети без регуляризации

|  |  |  |  |
| --- | --- | --- | --- |
| Функция активации | Скорость обучения | Количество нейронов в слое | Количество слоев |
| ReLU | 1e-5 | 16 | 2 |
|  | 5e-5 | 28 | 3 |
| 1e-4 | 32 | 4 |
| 5e-4 | 64 | 5 |
| 1e-3 | 128 | 6 |
| 5e-3 | 256 | 7 |
| 1e-2 |  | 8 |
|  | 9 |

Благодаря случайной выборке гиперпараметров можно используя меньшее количество операций кросс-валидации выделить основные тенденции влияния различных гиперпараметров на результаты расчета.

## Использование XGBoost

## Применение методик регуляризации для уменьшения дисперсии

Переобучение – одна из проблем машинного обучения, которая заключается в чрезмерной адаптации модели к набору обучения, что приводит к плохим результатам на наборах кросс-валидации и соответственно на тестовом наборе. Далее будут приведены результаты использования методик регуляризации, которые позволяет уменьшить адаптацию модели к набору обучения (уменьшение дисперсии) не приводя к существенному увеличению ошибки на тестовом наборе (к увеличению смещения).

## L1 - регуляризация

## L2 - регуляризация

L2 - регуляризация заключается в добавлении штрафа к первоначальной функции ошибки с коэффициентом, который управляет воздействием такого типа регуляризации. Другое название - подавление весов (weight decay). Такой простой тип ограничения позволяет эффективно уменьшать дисперсию выходных результатов.

Таблица – Гиперпараметры нейронной сети

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Функция активации | Скорость обучения | Количество нейронов в слое | Количество слоев |  |
| ReLU | 1e-5 | 16 | 2 | 5e-5 |
|  | 5e-5 | 28 | 3 | 1e-4 |
|  | 1e-4 | 32 | 4 | 5e-3 |
|  | 5e-4 | 64 | 5 | 1e-3 |
|  | 1e-3 | 128 | 6 | 5e-2 |
|  | 5e-3 | 256 | 7 | 1e-2 |
|  | 1e-2 |  | 8 | 0.05 |
|  |  | 9 | 0.1 |

Таблица – Результат структуры нейронной сети

|  |  |  |
| --- | --- | --- |
| Функция активации | Параметры | Результат (величина ошибки RMSLE) |
| ReLU | Epochs:  Скорость обучения:  Количество слоев:  Количество узлов:  : |  |

Использование L2 регуляризации приводит к значительному уменьшению ошибки по сравнению с предыдущими расчетами

## Dropout - регуляризация

Одно из решений проблемы переобучения, благодаря своей простоте и практическим результатам – Dropout регуляризация («метод прореживания», «метод исключения»).

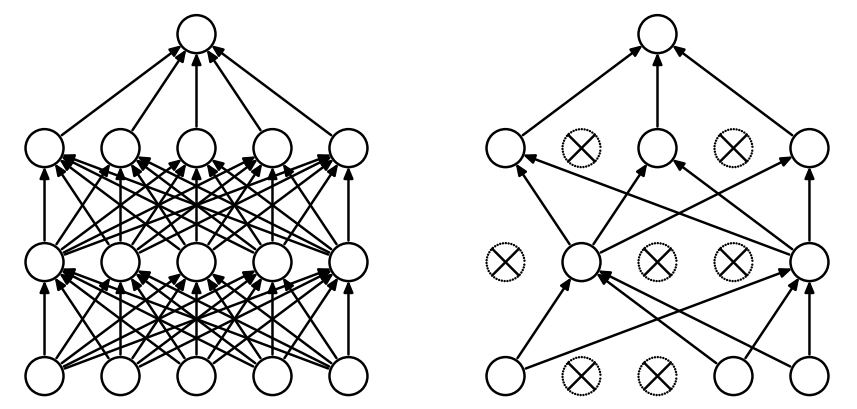


Рисунок 13 – Визуализация стандартной нейронной сети и нейронной сети с применением регуляризации Dropout. Графическое представление метода Dropout

Главная задача применения Dropout – вместо обучения одной нейронной сети обучить ансамбль нескольких нейронных сетей, а затем усреднить полученные результаты.

Сети для обучения получаются с помощью исключения из сети (dropping out) нейронов с вероятностью *p* (коэффициент Dropout), таким образом, вероятность того, что нейрон останется в сети, составляет *q = 1 – p*. «Исключение» нейрона означает, что при любых входных данных или параметрах он возвращает 0.

Исключенные нейроны не вносят свой вклад в процесс обучения ни на одном из этапов алгоритма обратного распространения ошибки (backpropagation), поэтому исключение хотя бы одного из нейронов равносильно обучению новой нейронной сети.

Таблица – Гиперпараметры нейронной сети. Фреймворк Keras

|  |  |  |  |
| --- | --- | --- | --- |
| Функция активации | Скорость обучения | Количество нейронов в слое | Количество слоев |
| ReLU | 1e-1 | 16 | 4 |
|  | 1e-2 | 32 |  |
| 1e-3 | 64 |
| 1e-4 | 128 |
| 1e-5 | 256 |
| 1e-6 |  |
| 5e-3 |
| 5e-4 |
| 5e-5 |

Таблица – Результат структуры нейронной сети

|  |  |  |
| --- | --- | --- |
| Функция активации | Параметры | Результат (величина ошибки RMSLE) |
| ReLU | Epochs:  Скорость обучения:  Количество слоев:  Количество узлов:  Dropout:  P1 =  P2 =  P3 = |  |

Выводы после применения Dropout регуляризации:

# Результат сравнения применения нейронных сетей с другими методами регрессионного анализа

# Схема решения задачи №2

# Использованные источники

x

|  |  |
| --- | --- |
| 1. | Serigne. Stacked Regressions to predict House Prices [Электронный ресурс] // Kaggle: [сайт]. [2017]. URL: https:/​/​www.kaggle.com/​serigne/​stacked-regressions-top-4-on-leaderboard |
| 2. | Sakia R.M. The Box-Cox transformation techinque: a review // The Statisticain, No. 41, 1992. pp. 169-178. |
| 3. | LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-Based Learning applied to Document Recognition, 1998. |
| 4. | Goodfellow I., Bengio Y., Courville A. Deep Learning. 2nd ed. Cambridge, Massachusets: MIT Press, 2017. 201-204 pp. |
| 5. | Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting // Journal of Machine Learning Research, Июнь 2014. pp. 1929-1958. |

x

# Приложение А

## Основной скрипт