Министерство образования и науки Российской Федерации

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Факультет прикладной математики и компьютерных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ

Задание 1. Полносвязные нейронные сети прямого распространения

Выполнила студентка группы 932001:

Ерыгина Виктория

Томск – 2022

**Цель** работы – получить навыки создания полносвязного бинарного и многоклассового классификаторов, а также нейросетевого регрессора с помощью библиотеки построения нейронных сетей.

**Задачи** – Построить три нейросетевые модели:

a) Бинарный Классификатор

b) Многоклассовый классификатор

c) Регрессор

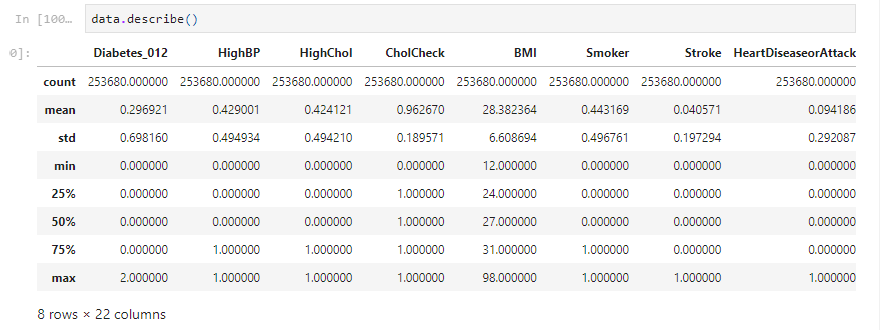
**Бинарный классификатор**

Описание данных

В ходе выполнения работы использовался датасет: Оценка вероятности диагностики диабета у человека (<https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset>). Датасет содержит 253680 записей и 22 признака.

1. Diabetes\_012 – целевая переменная. Класс отрицательный – 0 (no diabetes – нет диабета), класс положительный – 1 & 2 (prediabetes – преддиабетическое состояние & diabetes - диабет)
2. HighBP – высокое артериальное давление. 0 – невысокое, 1 – высокое.
3. HighChol – высокий уровень холестерина. 0 – невысокий, 1 – высокий.
4. CholCheck – была ли проверка холестерина в течение 5 лет. 0 – не проверялось в течение 5 лет, 1 – проверялось.
5. BMI – индекс массы тела
6. Smoker – вы курили не менее 100 сигарет за всю жизнь? 0 – нет, 1 – да.
7. Stroke – у вас был инсульт? 0 – нет, 1 - да
8. HeartDiseaseorAttack – наличие ишемической болезни сердца (ИБС) или инфаркта миокарда (ИМ). 0 – нет, 1 – да.
9. PhysActivity – наличие физической активности за последние 30 дней, не включая работу. 0 – нет, 1 – да.
10. Fruits – употребление фруктов 1 и более раз в день. 0 – нет, 1 – да.
11. Veggies – употребление овощей 1 и более раз в день. 0 – нет, 1 – да.
12. HvyAlcoholConsump – употребление более 14 алкогольных напитков в неделю для мужчин или более 7 для женщин. 0 – нет, 1 – да.
13. AnyHealthcare – наличие любого медицинского страхования. 0 – нет, 1 – да.
14. NoDocbcCost – была необходимость обратиться к врачу за последние 12 месяцев, но отказ из-за стоимости. 0 – нет, 1 – да.
15. GenHlth – оценка здоровья по шкале 1-5. 1 – отлично, 2 – очень хорошо, 3 – хорошо, 4 – нормально, 5 – плохо.
16. MentHlth – в течение скольких дней за последний месяц ваше психическое здоровье было не очень хорошим? Шкала 1 – 30 дней.
17. PhysHlth – в течение скольких дней за последний месяц ваше физическое здоровье было не очень хорошим? Шкала 1 – 30 дней.
18. DiffWalk – есть ли серьезные трудности с ходьбой или подъемом по лестнице? 0 – нет, 1 – да.
19. Sex – пол. 0 – женщина, 1 – мужчина.
20. Age – возраст. 13 категорий. 1 – от 18 до 24, 9 – от 60 до 64, 13 – от 80 и более
21. Education – образование. Шкала 1-6. 1- никогда не посещал школу ил только детский сад, 2 – классы с 1 по 8, 3 – классы с 9 по 11, 4 – 12 класс или выпускник средней школы, 5 – колледж от 1 года до 3 лет, 6 – колледж 4 года и более.
22. Income – доход. Шкала 1- 8. 1 – не менее 10000$ США, 5 – не менее 35000$ США, 8 – 75000$ США и более.

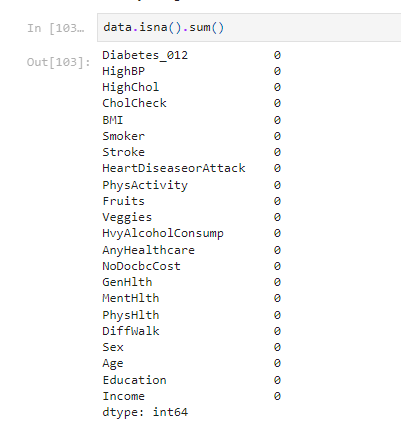
Получим описание данных с помощью функции describe():



Листинг 1. Описание данных датасета Diabetes\_012

* count (количество данных для каждого столбца).
* mean (среднее значение) показывает среднее арифметическое каждого признака.
* std (стандартное отклонение) позволяет оценить разброс данных выборки.
* min (минимальное значение).
* 25% (первый квантиль).
* 50% (второй квантиль, медиана).
* 75% (третий квантиль).
* max (максимальное значение).

Посмотрим наличие пропущенных данных:



Листинг 2. Наличие пропущенных данных датасета Diabetes\_012

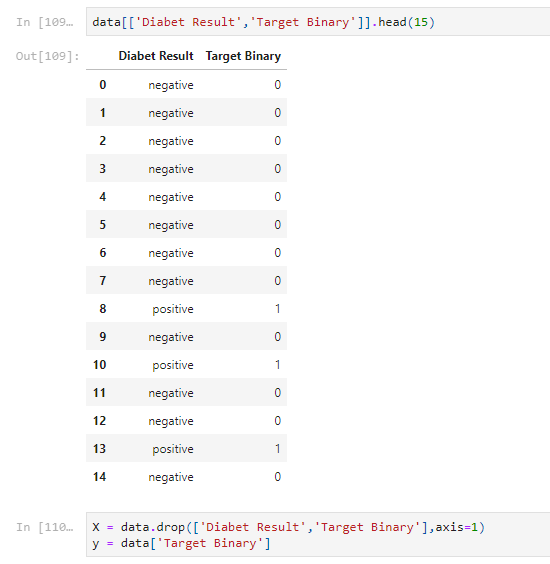
Пропущенных данных нет ни в одном из признаков.

Распространенность диабета зависит от возраста, образования, дохода, местоположения, расы и других детерминант здоровья.

Цель классификатора: оценить вероятность того, что у опрашиваемого человека есть диабет.

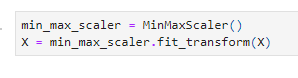
Добавление целевой переменной

Создадим новый столбец, в котором будет выводиться «negative» и «positive» в зависимости от значения, лежащего в столбце «Diabetes\_012». При этом «negative» будет относиться к 0, «positive» - к 1 и 2.



Листинг 3. Новая переменная со значениями «negative» и «positive»

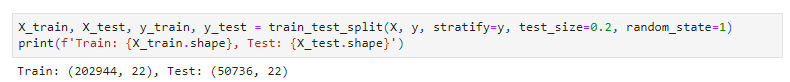
Используем инструмент для масштабирования MinMaxScaler() для дальнейшего обучения нейронной сети. Все значения будут находиться в диапазоне от 0 до 1.



Листинг 4. Использования инструмента MinMaxScaler()

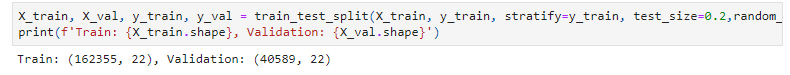
Разделение набора на обучающую, тестовую и валидационную выборки

Весь набор разбивается на 2 части, где 80% идут на обучающую, 20% на тестовую.



Листинг 5. Разбиение на обучающую и тестовую выборки

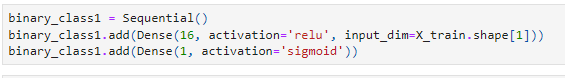
Повторно делим X\_train. Теперь 20% у нас будут использоваться для валидационного набора, 80% для обучения



Листинг 6. Разбиение на обучающую и валидационную выборки

Построение нейросети

Для построения нейронной сети используется библиотека TensorFlow. Строим экземпляр класса Sequential, то есть последовательную архитектуру нейронной сети. Добавляем полносвязный слой с 16 узлами к модели и слой sigmoid с 1 выходом.



Листинг 6. Использование функции Sequential

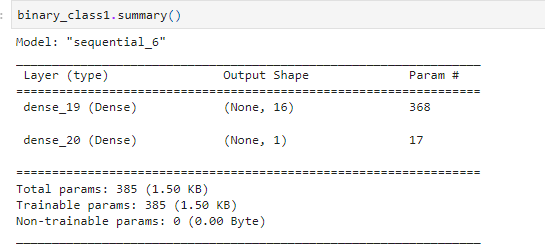
Модель сконструирована, настроим процесс ее обучения вызовом метода compile.

1. Optimizer – объект, который определяет процедуру обучения.
2. Loss – функция потерь (функция, которая минимизируется в процессе обучения).
3. Metrics используются для мониторинга обучения.



Листинг 7. Функция потерь и оптимизатор

Вызываем функцию summary, для того чтобы получить количество просуммированных весов.

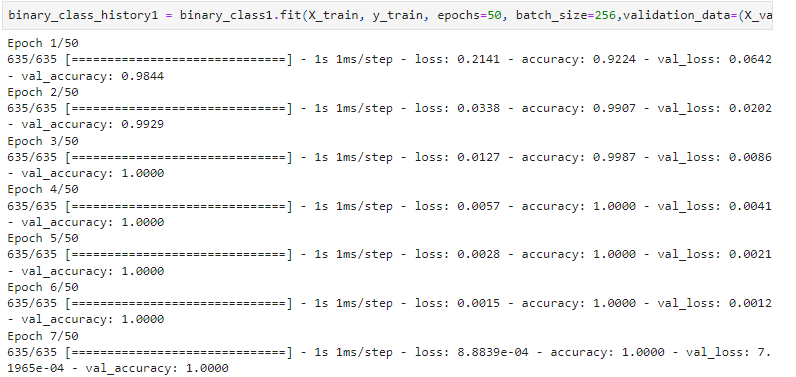


Листинг 8. Вызов функции summary

Получаем количество просуммированных весов 385, которые мы должны настроить.

Обучение нейронной сети

Для обучения нейронной сети использовалось 50 эпох.



Листинг 8. Вызов фунции и демонстрация обучения нейронной сети

Вычисляем метрики работы для бинарного классификатора (Recall, Precision, Weighted Accuracy, AUC) и рисуем графики.

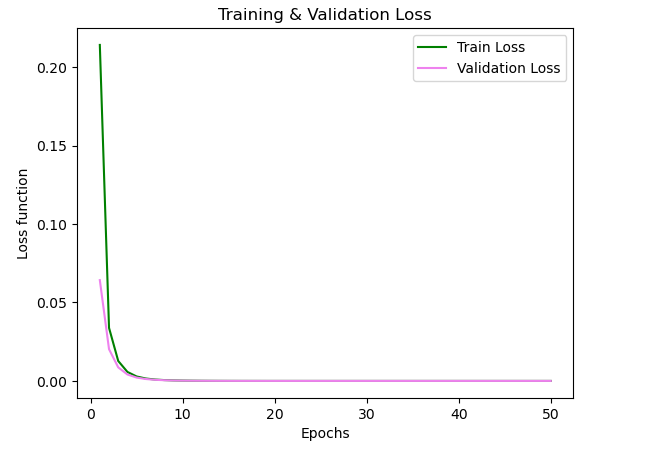


Рис 1. Training and validation loss

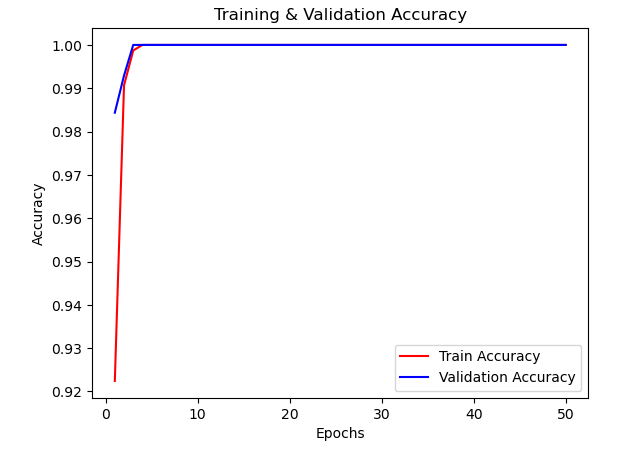
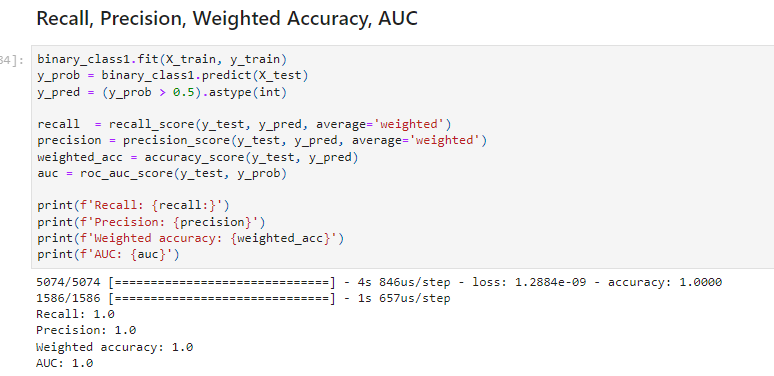


Рис 2. Training and validation accuracy



Листинг 9. Вывод значений метрик работы

Повторим то же самое с другим набором параметров. Для этого добавляем полносвязный слой с 64 узлами к модели, слой с 16 выходами и sigmoid слой с 1 выходом.

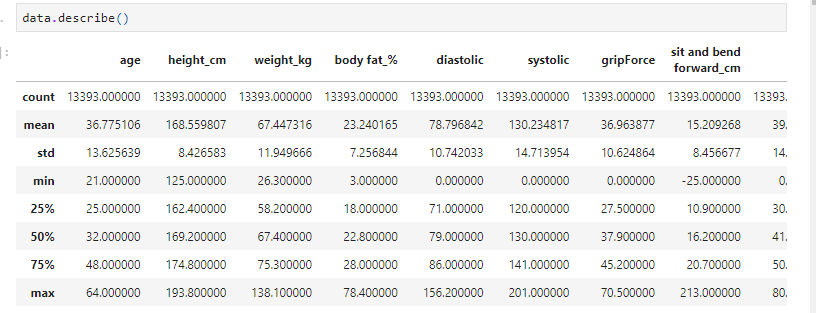
**Многоклассовый классификатор**

Описание данных

В ходе выполнения работы использовался датасет: Оценка уровня физического развития людей разного возраста (<https://www.kaggle.com/datasets/kukuroo3/body-performance-data>). Датесет содержит данные, которые подтвердили уровень работоспособности с возрастом и некоторые данные о выполнении упражнений.

1. age - возраст (20 ~64)
2. gender - пол (женский, мужской)
3. height\_cm - рост в сантиметрах
4. weight\_kg – вес в кг
5. body fat\_% - жировые отложения в процентах
6. diastolic – диалостическое артериальное давление (мин)
7. systolic : систолическое артериальное давление (мин)
8. gripForce – сила захвата
9. sit and bend forward\_cm – глубина наклона (см)
10. sit-ups counts – количество приседаний
11. broad jump\_cm – длина прыжка (см)
12. class : A,B,C,D ( A: best) / stratified – класс (A – лучший)

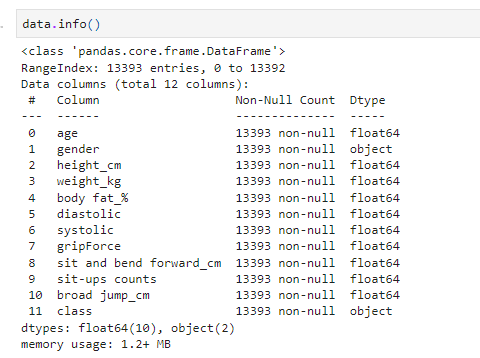
С помощью функции describe() получаем описание данных.



Листинг 10. Описание данных

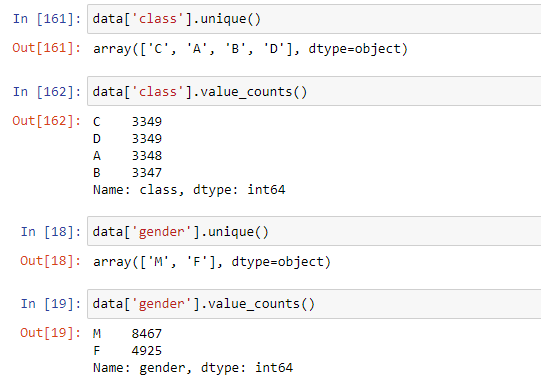
С помощью функции info() узнаем, что датасет содержит 13393 записи и 12

признаков.



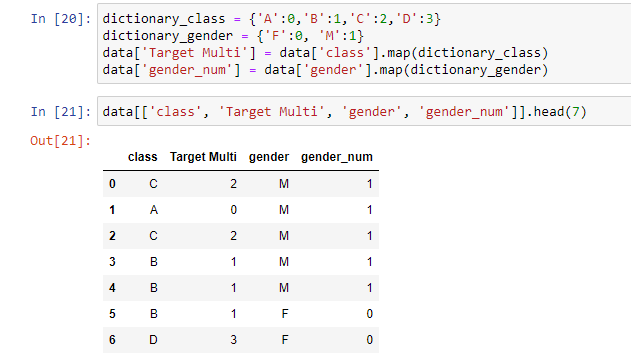
Листинг 11.

У нас есть два признака с типом object (gender и class). Так как мы не можем работать с категориальными признаками для построения модели, то преобразуем их в числовые. Выведем уникальные значения найденных категориальных признаков:



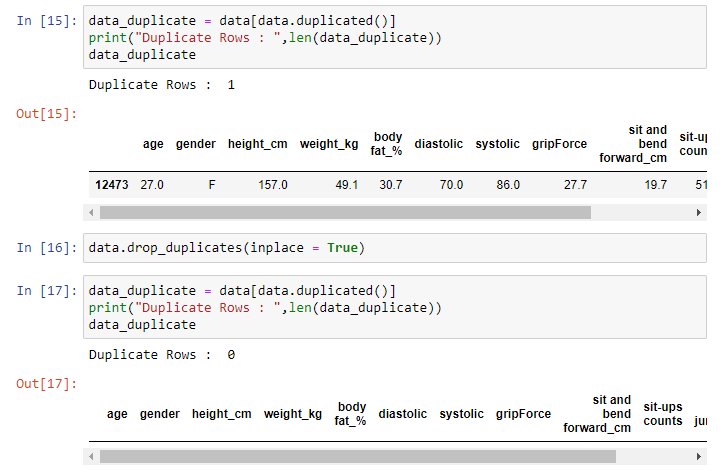
Листинг 11. Уникальные значения категориальных признаков

Видим, что признак “class” имеет 4 уникальных значения, к которому относится опрашиваемый. Признак “gender” имеет 2 уникальных значения. Чтобы работать с категориальными переменными, будем использовать свой числовой словарь и выведем полученное соответствие.



Листинг 12. Замена категориальных данных на числовые

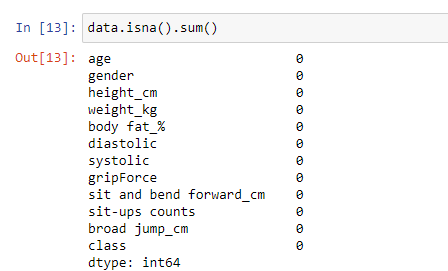
Проверяем данные на наличие дубликатов, удаляем их и делаем повторную проверку.



Листинг 13. Удаление дубликатов и проверка

Пропущенные данные

Проверим датасет на наличие пропущенных данных.

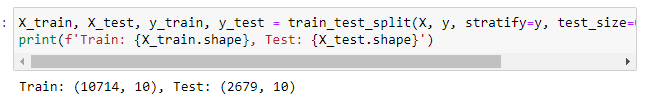


Листинг 14. Наличие пропущенных данных

Видим, что пропущенных данных нет ни по одному из признаков, поэтому можем продолжать работу дальше.

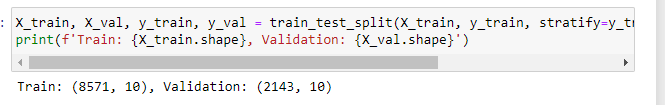
Обучающая, тестовая и валидационная выборки

Весь набор разбивается на 2 части, где 80% идут на обучающую, 20% на тестовую.

****

Листинг 15. Разделение на обучающую и тестовую

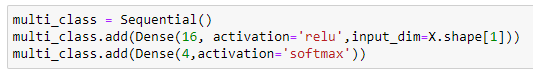
Повторно делим X\_train. Теперь 20% у нас будут использоваться для валидационного набора, 80% для обучения.



Листинг 16. Разделение на обучающую и валидационную

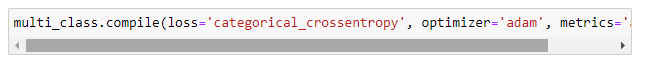
Обучение нейронной сети

Для построения нейронной сети используется библиотека TensorFlow. Строим экземпляр класса Sequential, то есть последовательную архитектуру нейронной сети. Добавляем полносвязный слой с 16 узлами и softmax слой с 4 выходами к модели.



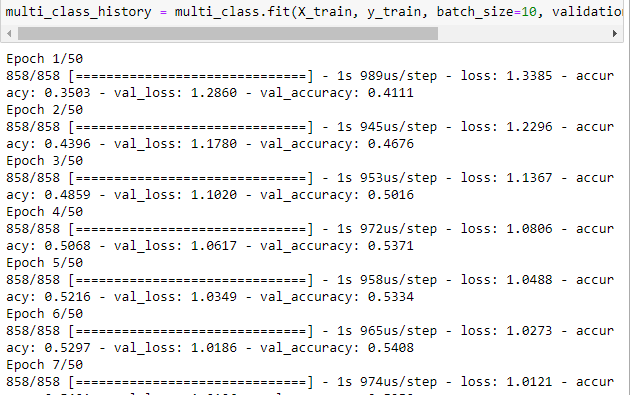
Листинг 17. Использование функции Sequential

После того как модель сконструирована, можем настроить процесс ее обучения вызовом метода compile.



Листинг 18. Использование метода compile

Обучаем нейронную сеть с помощью метода fit. Модель будет обучаться на 50 эпохах и с размером каждого блока данных 10.



Листинг 19. Демонстрация обучения нейронной сети

Вычислим следующие метрики: Recall, Precision, Weighted Accuracy, AUC. Нарисуем графики и выведем ROC – кривые для каждого класса в лучшем классификаторе.

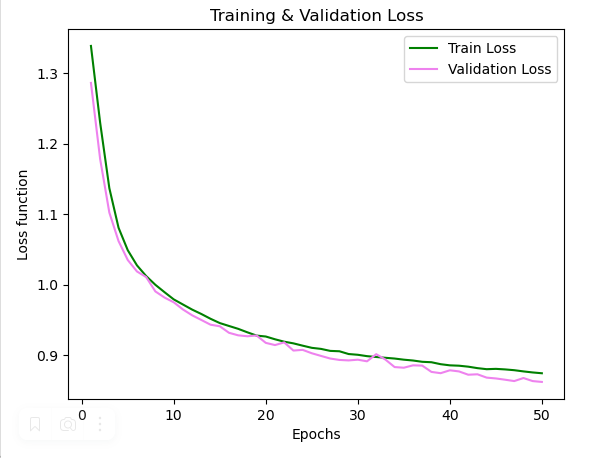


Рис 3. Training and validation loss

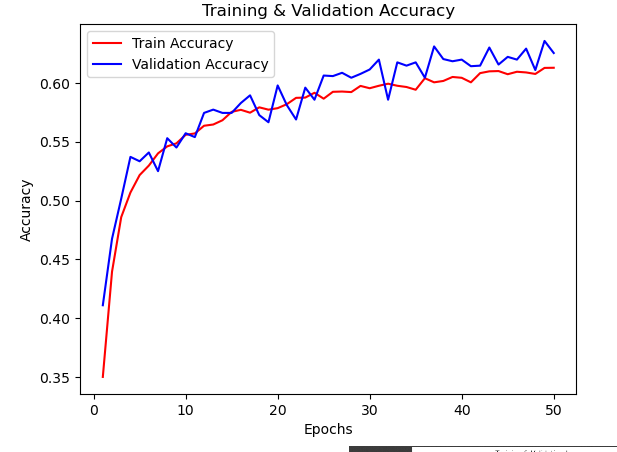
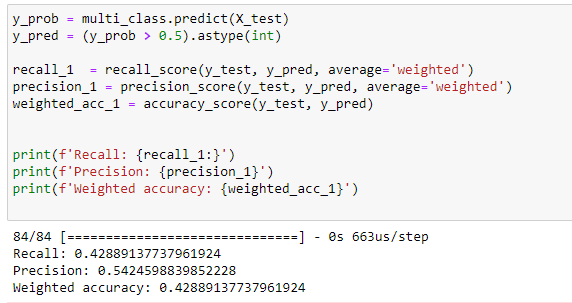


Рис 4. Training and validation accuracy



Листинг 20. Вывод метрик работы

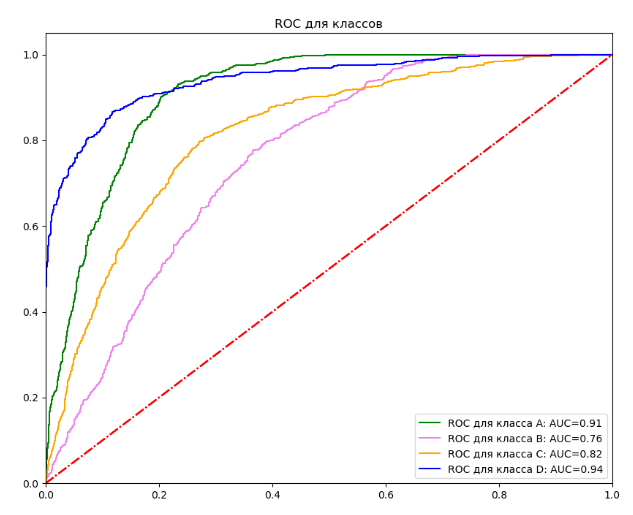
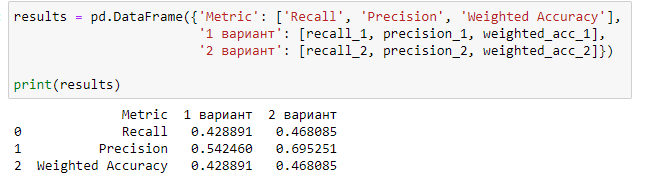


Рис 5. ROC кривые для каждого класса

Проделаем то же самое, но с другим набором параметров. Для этого добавим полносвязный слой с 32 узлами, слой с 16 выходами и softmax слой с 4 выходами.

Вывод

Чем больше изгиб кривой, тем лучше модель классифицирует данные по категориям. Чем ближе AUC находится к 1, тем лучше модель. Заметим, что во втором варианте класс D имеет самую высокую AUC, следовательно, это лучшая модель для классификации данных по категориям.



Листинг 21. Вывод таблицы метрик двух случаев

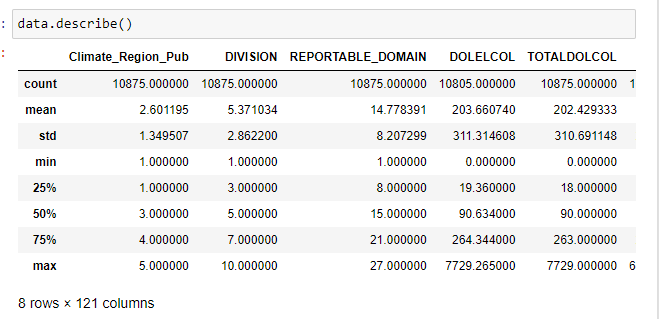
По метрике **Recall** 2 вариант превосходит, следовательно, вторая модель способна лучше обнаруживать положительные (истинные) примеры в данных.

Так как **Precision** второй модель больше, то она имеет более высокую точность.

**Регрессор**

Для построения модели использовался датасет: оценка расходов домохозяйства на один из видов потребляемых ресурсов (DS\_2019\_public.csv). Описание атрибутов расположено в файле recs2009\_public\_codebook.xlsx.

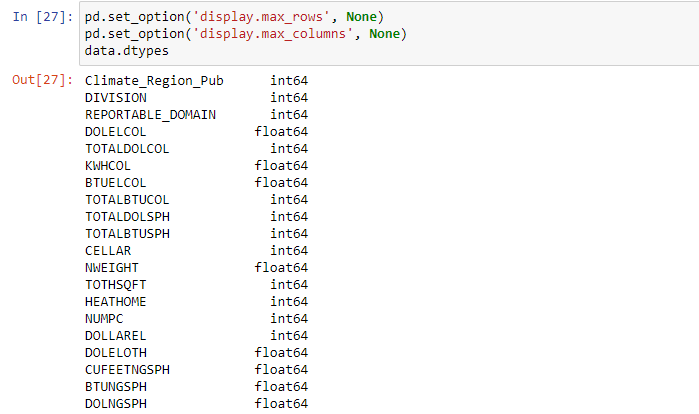
Получаем описание датасета по количеству данных, среднему значению, стандартному отклонению, минимальному значению, квантилям и максимальному значению с помощью метода describe():



Листинг 22. Использование метода describe

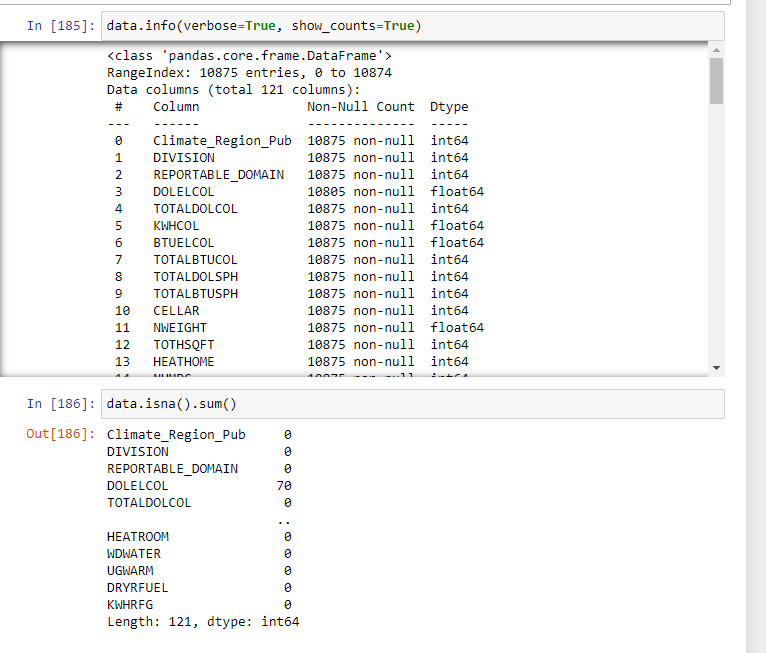
Датасет содержит 10875 записей, 121 признак, в которых находится одна целевая переменная target.

Выведем все типы признаков, чтобы удостовериться, что среди них нет категориальных, которые нам нужно будет обработать для дальнейшего обучения.



Листинг 23. Вывод всех типов признаков

Проверим датасет на пропущенные данные с помощью применения методов info() и isna().



Листинг 24. Поиск пропущенных данных

Так как у меня не выводились значения, представленные в файле, я поменяла тип файла на xlsx и конвертировала в csv. Из-за этого мы получили пропущенные данные, которые нам необходимо обработать для дальнейшей работы. Последовательно выведем процент пропущенных данных в некоторых признаках и удалим строки, которые их содержат.



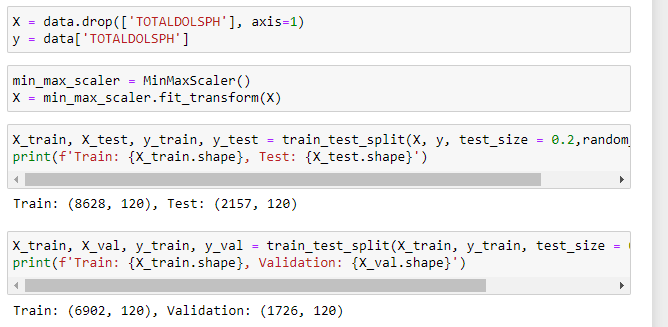
Листинг 25. Обработка пропущенных данных

Обучающая, тестовая и валидационная выборки

В качестве предсказываемого признака выбран **TOTALDOLSPH**.

Используем инструмент для масштабирования MinMaxScaler() для дальнейшего обучения нейронной сети. Все значения будут находиться в диапазоне от 0 до 1.

Разделим набор на обучающую, тестовую и валидационную выборки.



Листинг 26. Разделение набора на обучающую, тестовую и валидационную выборки

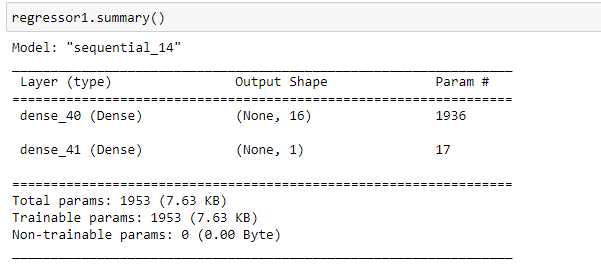
Обучение нейронной сети

Для построения нейронной сети используется библиотека TensorFlow. Строим экземпляр класса Sequential, то есть последовательную архитектуру нейронной сети. Добавим полносвязный слой с 16 узлами и слой с 1 выходом.



Листинг 27. Использование метода Sequential

Вызываем функцию summary, для того чтобы получить количество просуммированных весов.



Листинг 28. Использование метода summary

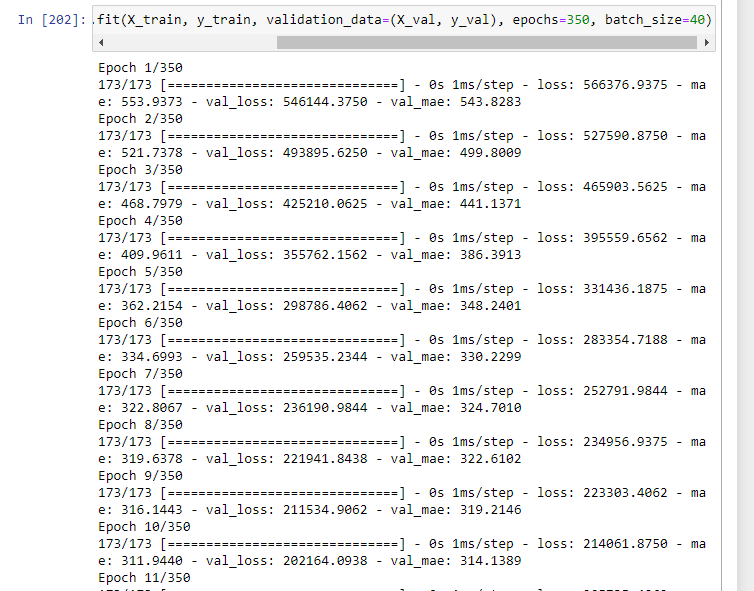
Получаем количество просуммированных весов 1953, которые мы должны настроить.

Модель сконструирована, настроим процесс ее обучения вызовом метода compile.



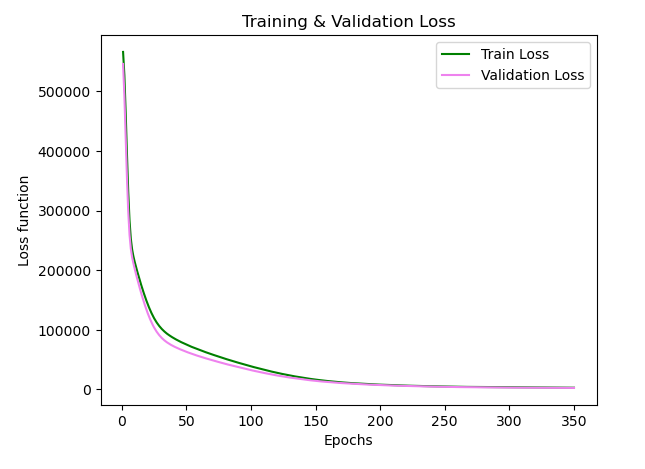
Листинг 29. Использование метода compile

Обучаем нейросетевую модель на 350 эпохах и с размером каждого блока данных 40.

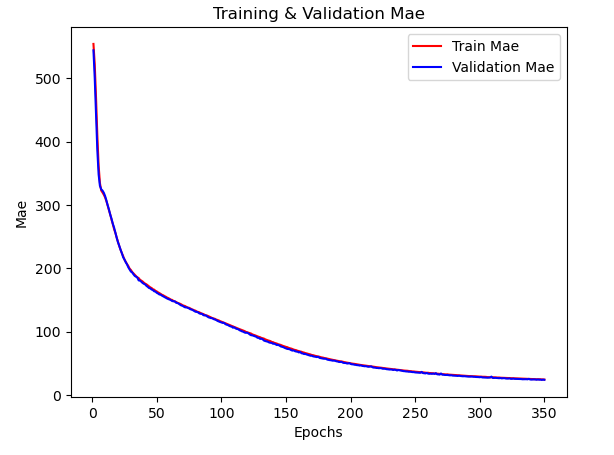


Листинг 30. Обучение нейронной сети

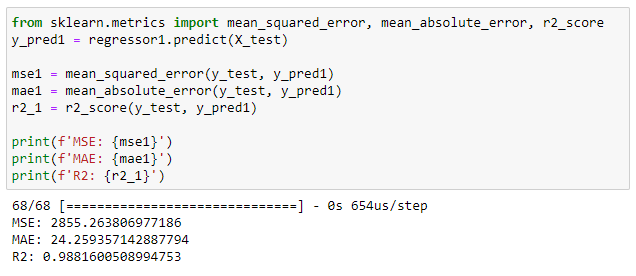
Вычислим метрики работы: MSE, MAE, R2, и нарисуем графики.



Листинг 6. Training and validation loss



Листинг 7. Training and validation MAE

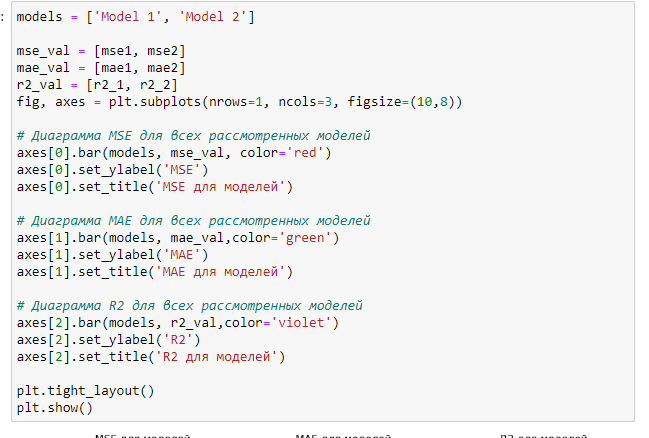


Листинг 31. Вывод метрик работы

Проделаем то же самое, но с другим набором параметров. Для этого выберем полносвязный слой с 32 узлами и слой с 1 выходом.

Вывод

Выведем графики метрик работы для двух построенных моделей:



Листинг 32. Вывод метрик работы для двух моделей

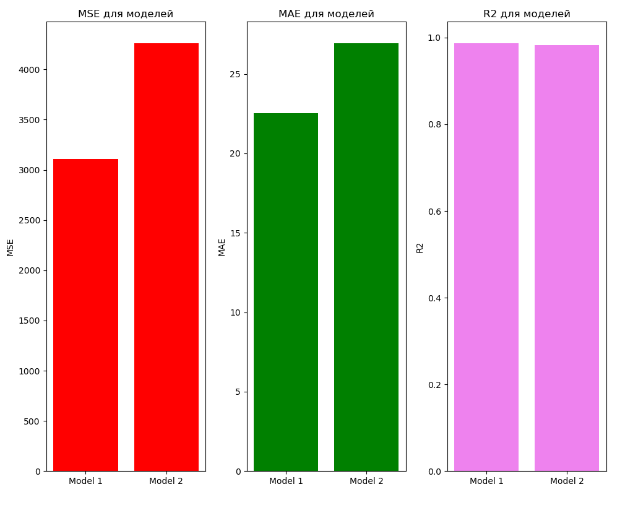
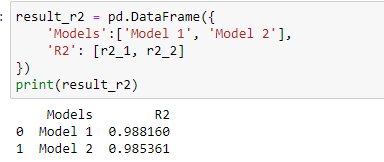


Рис 8. Сравнительные графики для метрик работы двух моделей

Чем ниже **MSE**, тем лучше модель соответствует набору данных. Следовательно, точность прогнозирования первой модели лучше.

Чем ниже **MAE**, тем лучше модель соответствует набору данных, поэтому Model 1 лучше прогнозирует.

На диаграмме видно, что значения R2 обеих моделей высокие.



Листинг 33. Сравнение метрики R2

Значение **R2** первой модели превосходит, поэтому Model 1 более эффективно использовать для прогнозирования данных.

**Программный код**

Программный код выложен на GitHub: <https://github.com/Erygina/Neural-networks.git>