

|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА** - **Российский технологический университет"**  **РТУ МИРЭА** |

**Институт** Информационных Технологий

**Кафедра** Вычислительной Техники

**ПРАКТИЧЕСКИЕ РАБОТЫ № 1-10**

**по дисциплине**

**«Проектирование и обучение нейронных сетей»**

Студент группы: ИКБО-14-20 Вежновец Ф.Ю. *(Ф.И.О.студента)*

Руководитель \_\_Гуличева А.А.\_\_

*(Ф.И.О. преподавателя)*

Москва 2023

введение

В современном мире, нейронные сети играют ключевую роль во многих сферах деятельности: анализ данных, распознавание образов, компьютерная визуализация и многое другое. С помощью нейронных сетей можно обрабатывать большие объемы данных, извлекать из них полезную информацию и делать выводы.

Обучение нейронных сетей - это процесс, при котором сеть учится распознавать закономерности в данных и использовать их для принятия решений. Для этого используются различные алгоритмы обучения, такие как правило Хебба, дельта-правило и обратное распространение ошибки, и другие. В данном курсе мы рассмотрим каждый из этих алгоритмов и научимся применять их на практике.

1. ОБУЧЕНИЕ ПО ПРАВИЛАМ ХЕББА
   1. Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать Обучение по правилам Хебба.

* 1. Описание обучения по правилам ХЕББА

Первое правило Хебба – Если сигнал персептрона неверен и равен нулю, то необходимо увеличить веса тех входов, на которые была подана единица.

Вытекает первый тип ошибки: на выходе персептрона 0, правильный ответ 1. Для того, чтобы персептрон выдавал правильный ответ необходимо, чтобы скалярное произведение стало больше. Поскольку переменные принимают значения 0 или 1, увеличение суммы может быть достигнуто за счет увеличения весов. Однако нет смысла увеличивать веса при переменных, которые равны нулю. Увеличиваем веса только при тех, которые равны 1. Для закрепления единичных сигналов с весов, следует провести ту же процедуру и на всех остальных слоях.

Второе правило Хебба — Если сигнал персептрона неверен и равен единице, то необходимо уменьшить веса тех входов, на которые была подана единица.

Отсюда вытекает второй тип ошибки: на выходе персептрона 1, правильный ответ 0. Для уменьшения скалярного произведения в правой части, необходимо уменьшить веса связей при тех переменных, которые равны 1. Необходимо также провести эту процедуру для всех активных нейронов предыдущих слоев.

Алгоритм обучения по правилу Хебба сводится к следующей последовательности действий:

* 1. Инициализация весовых коэффициентов и порогов случайными значениями, близкими к нулю (чтобы сеть сразу не могла войти в насыщение).
  2. Подача на вход НС очередного входного образа.
  3. Вычисление значения выхода.
  4. Если значение выхода не совпадает с эталонным значением, то происходит модификация коэффициентов в соответствии с формулами при скорости обучения η=1. В противном случае осуществляется переход к пункту 5.
  5. Если не все вектора из обучающей выборки были поданы на вход НС, то происходит переход к пункту 2. Иначе, переход к пункту 6.
  6. Останов.

1.3 Результат работы

Создадим простейший перцептрон для распознавания единицы и нуля. Для обучения модели будем использовать алгоритм обучения Хебба.

Перцептрон реализован на языке Python (Листинг 1.1 – 1.3) результат выполнения приведен на Рисунке 1.1.

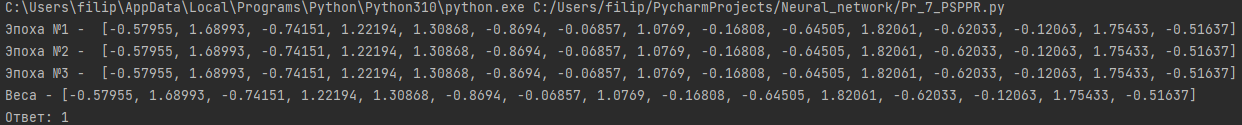


Рисунок 1.1 – Результат обучения

Заданная картинка для распознавания приведена на Рисунке 1.2.



Рисунок 1.2 – Входные данные

2. ДЕЛЬТА ПРАВИЛО

2.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать обучение по дельта правилу.

2.2 Описание обучения по дельта правилу

Алгоритм обучения Розенблатта (дельта-правило) сводится к следующей последовательности действий:

1. Инициализация весовых коэффициентов и порогов значениями, близкими к нулю.
2. Подача на вход нейронной сети очередного входного образа (входного вектора Х), взятого из обучающей выборки, и вычисление суммарного сигнала по всем входам для каждого нейрона j:

, где n – размерность входного вектора, xi – i-я компонента входного вектора, wij – весовой коэффициент связи нейрона j и входа i.

1. Вычисление значения выхода каждого нейрона:

, где bj – порог, соответствующий нейрону j

1. Вычисление значения ошибки обучения для каждого нейрона еj = dj - yj
2. Проводится модификация весового коэффициента связи по формуле
3. Повторение пунктов 2 – 5 до тех пор, пока ошибка сети не станет меньше заданной еj < езад .

2.3 Результат работы

Опишем предметную область.

Высказывания:

– сборщик установил систему охлаждения;

– сборщик одел кулер на башню;

– сборщик подключил кулер;

– сборщик прикрутил башню;

a – сборщик нанёс термопасту.

Аксиомы:

– чтобы написать экзамен, надо получить справку и найти кабинет;

Целевая логическая функция:

Для данной логической функции напишем таблицу истинности (Таблица 2.1).

Таблица 2.1 – Таблица истинности для целевой истинности

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

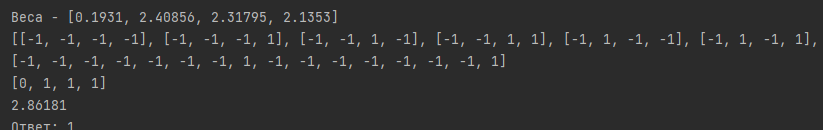


Рисунок 2.1 – Результат обучения

3. ОБРАТНОЕ РАССПРОСТАНЕНИЕ ОШИБКИ

3.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать обучение при помощи обратного распространения ошибки.

3.2 Описание алгоритма обратного распространения ошибки

Общая структура алгоритма аналогична, алгоритму обучения Розенблатта (дельта-правило) с усложнением формул подстройки весов. В качестве активационной функции в многослойных персептронах, как правило, используется сигмоидальная активационная функция, в частности логистическая:

𝒇(𝑼) = 𝟏⁄(𝟏 + 𝒆−𝒙) или 𝑶𝑼𝑻 = 𝟏⁄(𝟏 + 𝒆−𝑵𝑬𝑻)

Вспомним, что производная этой функции равна:

𝑶𝑼𝑻/ = OUT(1 – OUT)

Рассмотрим алгоритм обратного распространения ошибки, который требует выполнения следующих операций:

Шаг 1. Выбрать очередную обучающую пару из обучающего множества; подать входной вектор на вход сети. Берется пример входного сигнала с соответствующим правильным значением выхода.

Шаг 2. Вычислить выход сети. При этом нейроны последовательно от слоя к слою функционируют по следующим формуле.

Операции, выполняемые шагами 1 и 2, сходны с теми, которые выполняются при функционировании уже обученной сети, – подается входной вектор и

вычисляется получающийся выход. Вычисления выполняются послойно. На шаги 1 и 2 можно смотреть как на «движение вперед», так как сигнал распространяется по сети от входа к выходу

Шаг 3. Рассчитывается прямое распространение ошибки через сеть. Берется, функционал суммарной квадратичной ошибки сети для одного входного образа имеет вид:

𝐸 = 0,5 ∑(𝑑𝑗 − 𝑦𝑗)2

𝑛

где dj – целевое значение входного сигнала; yj – реальное значение входного сигнала.

Шаг 4. Начиная с выходов, выполняется обратное движение через ячейки выходного и промежуточного слоя, при этом программа рассчитывает значения:

Для выходной ячейки

✿𝟎 = 𝒚𝒋(𝟏 − 𝒚𝒋)(𝒅𝒋 − 𝒚𝒋) или ✿𝟎 = 𝑶𝑼𝑻(𝟏 − 𝑶𝑼𝑻)(𝑶𝑼𝑻∗ − 𝑶𝑼𝑻);

где 𝑂𝑈𝑇∗ - целевое значение.

𝑂𝑈𝑇 – реальное значение

𝑂𝑈𝑇(1 − 𝑂𝑈𝑇) – производная от сигмоида.

Выход нейрона слоя, вычитаемый из целевого значения, дает сигнал ошибки. Он умножается на производную сжимающей функции 𝑂𝑈𝑇(1 − 𝑂𝑈𝑇), вычисленную для этого нейрона, давая, таким образом, величину обратной ошибки в узле.

Для скрытых ячеек

𝜹𝒋 = 𝑶𝑼𝑻(𝟏 − 𝑶𝑼𝑻) ∑𝒌 𝑚𝒌𝒋 ✿𝒌

𝜹𝒋 – ошибка элемента с индексом j;

k – индекс, соответствующий слою, который посылает ошибку «обратно»;

∑𝒌 𝑚𝒌𝒋 ✿𝒌 – обозначает все ячейки, связанные со скрытым слоем, 𝑚𝒊𝒋 – заданный вектор веса в скрытом слое, ✿𝒌 – обратная ошибка от слоя, который ее посылает.

Шаг 5. Рассчитываем величину, на которую необходимо изменить значения весовых коэффициентов. При этом используется дельта-праило.

Тогда для весов соединений между скрытым слоем и выходом

𝒘∗ = 𝒘𝒊𝒋 + 𝑎 𝜹𝟎𝒇(𝑼𝒊)

где i – номер нейрона в выходном слое; j – номер нейрона в скрытом слое; α – коэффициент обучения; 𝒇(𝑼𝒊)– входной сигнал; 𝜹𝟎 – ошибка обучения на выходе.

Для весов соединений между скрытым слоем и входом

3.3 Результат работы

Архитектура сети состоит из 15 входных нейронов, 5 нейронов в скрытом слое и 1 выходной нейрон (рисунок 3.1).

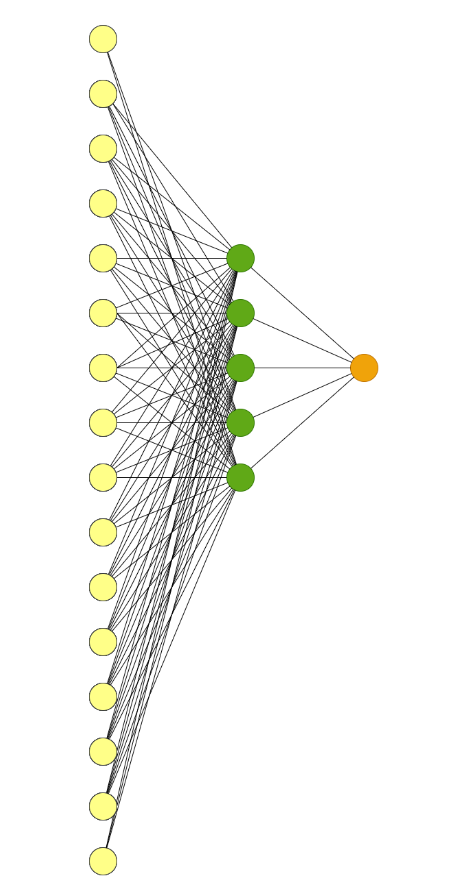


Рисунок 3.1 – Архитектура нейронной сети

Будем распознавать цифру 1 (скобочка). В качестве функции активацию на всех слоях будем использовать сигмойду.

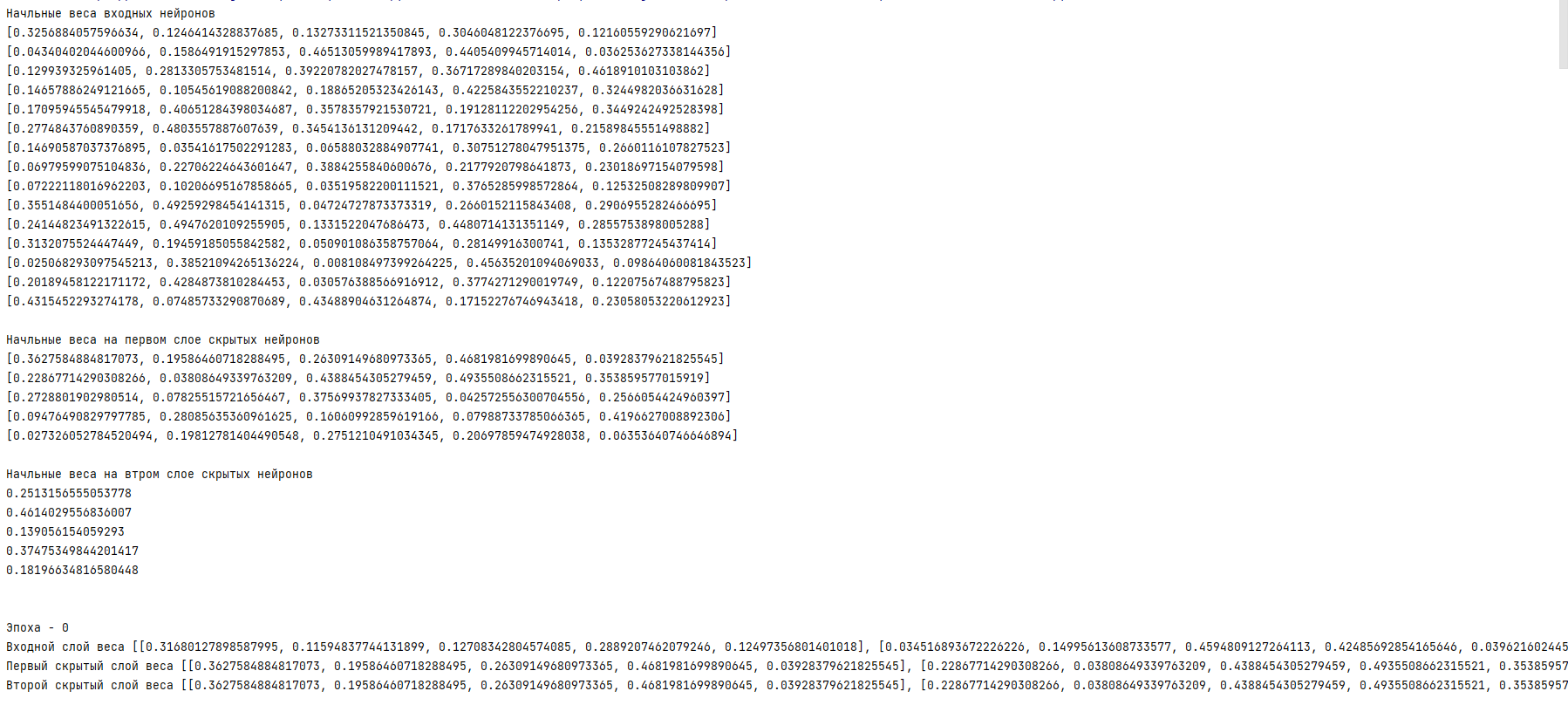


Рисунок 3.2 – Результат работы программы

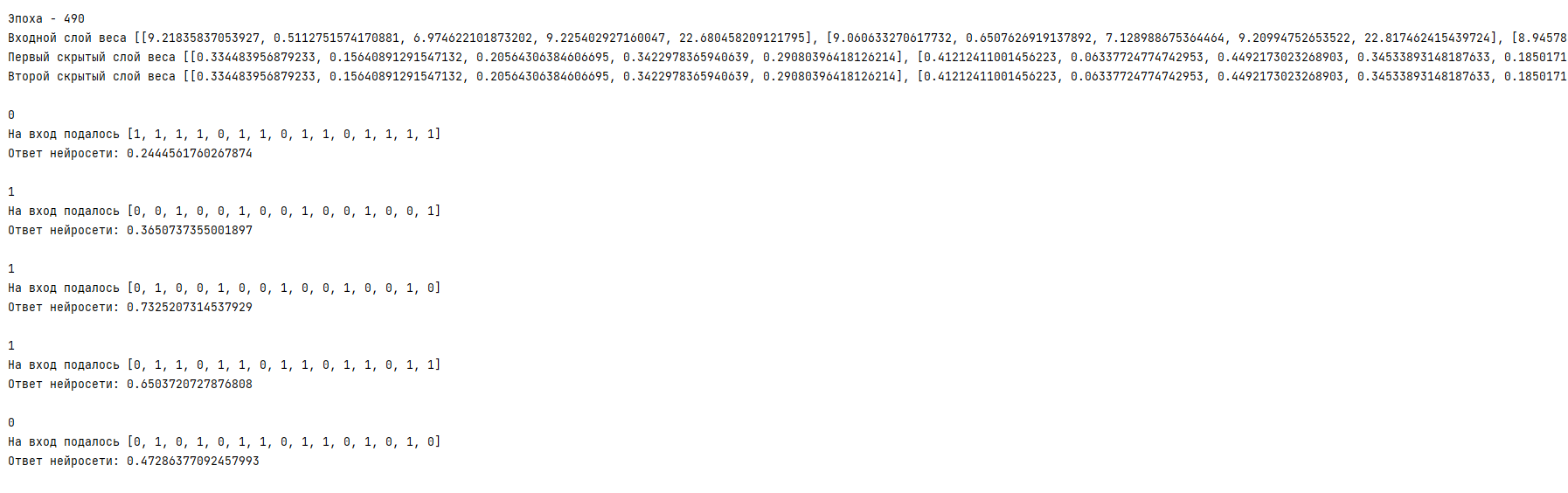


Рисунок 3.3 – Результат работы программы

Нейронная сеть правильно классифицирует единицу и обучается при помощи обратного распространения ошибки.

4. РАДИАЛЬНО БАЗИСНЫЕ ФУНКЦИИ

4.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать нейронную сеть с радиально базисными функциями.

4.2 Описание нейросетей с радиально базисными функциями

Нейронная сеть с радиально базисными функциями (RBF) - это тип искусственной нейронной сети, которая использует радиальные базисные функции (RBF) как функции активации. RBF - это нелинейные функции, которые имеют локальный характер, то есть они имеют большие значения вблизи своего центра и малые значения вдали от него.

Архитектура нейросети с RBF обычно состоит из двух слоев:

* Входной слой: Этот слой получает входные данные.
* Скрытый слой: Этот слой состоит из RBF.
* Выходной слой: Этот слой производит выходной сигнал.

Обучение нейросети с RBF заключается в настройке параметров RBF, чтобы минимизировать ошибку между выходами сети и целевыми значениями.

RBF-сети обладают рядом преимуществ по сравнению с другими типами нейронных сетей. Они являются универсальными аппроксиматорами, то есть они могут аппроксимировать любую непрерывную функцию с произвольной точностью. Кроме того, они устойчивы к шуму в данных.

RBF-сети находят широкое применение в различных областях, включая:

* Функция приближения: RBF-сети могут использоваться для аппроксимации сложных функций, таких как функции распределения вероятностей.
* Классификация: RBF-сети могут использоваться для классификации объектов по их характеристикам.
* Регрессия: RBF-сети могут использоваться для прогнозирования значений зависимой переменной по значениям независимых переменных.
* Системы управления: RBF-сети могут использоваться для управления сложными системами.

Описание обучения нейросети с радиально базисными функциями (RBF):

Этап 1: Выбор центроидов

Центры RBF выбираются из обучающих данных таким образом, чтобы они равномерно покрывали область ввода.

Этап 2: Оптимизация параметров RBF

Параметры RBF, такие как ширина RBF и веса выходного слоя, оптимизируются с помощью методов оптимизации, таких как градиентный спуск или генетические алгоритмы.

В целом, обучение RBF-сетей заключается в настройке параметров RBF, чтобы минимизировать ошибку между выходами сети и целевыми значениями.

4.3 Результат работы

Будем обучать нейронную сеть на датасете ирисов (Рисунок 4.1).

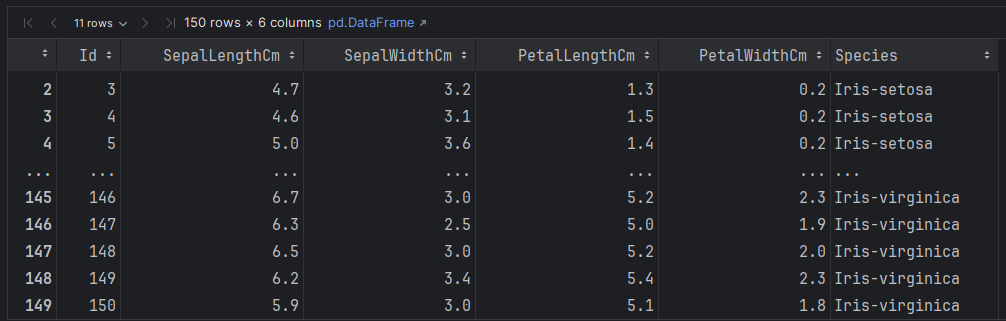


Рисунок 4.1 – Датасет

Обучаться будем 500 эпох, с шагом обучения 0.01 (Рисунок 4.2).

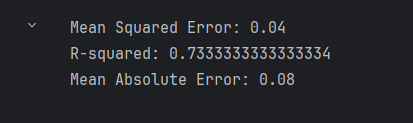


Рисунок 4.2 – Полученная точность на тестовой выборке

5. КАРТЫ КОХОНЕНА

5.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать карты Кохонена.

5.2 Описание карт Кохонена

Карты Кохонена, также известные как самоорганизующиеся карты (SOM), представляют собой тип нейронной сети с обучением без учителя, который используется для задач кластеризации и визуализации.

Карта Кохонена состоит из двух слоев нейронов:

* Входной слой: Этот слой получает входные данные.
* Скрытый слой: Этот слой состоит из нейронов, которые называются нейронами Кохонена.

Нейронные клетки Кохонена имеют нелинейные функции активации, которые обычно принимают форму гауссиана.

**Обучение.**

Обучение карты Кохонена происходит следующим образом:

* Каждый нейрон скрытого слоя назначается одному из кластеров.
* Для каждого входного вектора определяется нейрон Кохонена, который имеет минимальное расстояние до входного вектора. Этот нейрон называется победителем.
* Веса победителя и его соседей корректируются таким образом, чтобы уменьшить расстояние между ними и входным вектором.

Процесс обучения повторяется для всех входных векторов обучающей выборки.

**Принцип работы.**

При обучении карты Кохонена происходит постепенное формирование кластеров в пространстве входных данных. Каждый нейрон Кохонена отвечает за один из кластеров. Веса нейронов Кохонена отражают центр кластера, за который они отвечают. В результате обучения карта Кохонена становится двумерной картой, на которой кластеры представлены отдельными областями.

5.3 Результат работы

Задача будет нетривиальной, будем кластеризовать данные на 2 класса, на вход подаются 2 переменные от 0 до 1 (Рисунок 5.1). Шаг обучения = 0.1, количество эпох 10.

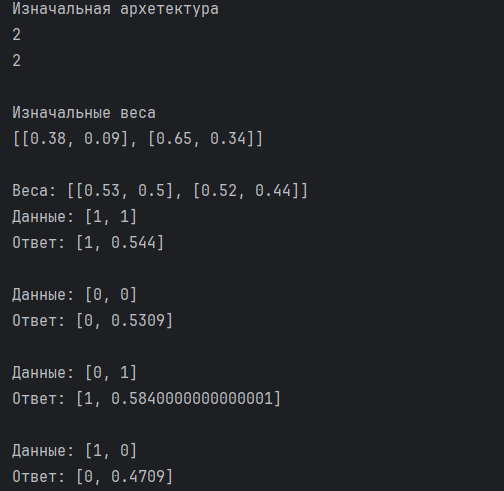


Рисунок 5.1 – Результат работы карт Кохонена.

6. ВСТРЕЧНОЕ РАСПРОСТРАНЕНИЕ

6.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать нейронную сеть встречного распространения.

6.2 Описание нейронной сети встречного распространения

Нейронная сеть встречного распространения — это тип нейронной сети с обучением с учителем, которая используется для решения задач классификации и регрессии.

Сеть встречного распространения состоит из трех слоев нейронов:

* Входной слой: Этот слой получает входные данные.
* Скрытый слой: Этот слой состоит из нейронов, которые обрабатывают входные данные.
* Выходной слой: Этот слой производит выходной сигнал.

Нейронные клетки входного слоя имеют линейные функции активации. Нейронные клетки скрытого слоя имеют нелинейные функции активации, такие как сигмоида или гиперболический тангенс. Нейронные клетки выходного слоя имеют линейные функции активации.

**Обучение.**

Обучение сети ВР происходит следующим образом:

* На вход сети подается обучающий вектор.
* Сеть производит выходной сигнал.
* Сравнивается выходной сигнал с целевым значением.
* Ошибка сети рассчитывается как разница между выходным сигналом и целевым значением.
* Веса сети обновляются таким образом, чтобы уменьшить ошибку сети.
* Процесс обучения повторяется для всех обучающих векторов.

**Принцип работы.**

При обучении сети ВР веса сети настраиваются таким образом, чтобы минимизировать ошибку сети. Ошибка сети рассчитывается как разница между выходным сигналом сети и целевым значением.

Веса сети обновляются с помощью метода градиентного спуска. Градиентный спуск — это метод оптимизации, который использует градиент ошибки для обновления параметров сети.

6.3 Результат работы

Будем обучать нейронную сеть на датасете ирисов (Рисунок 6.1). Количество эпох = 100, изначальные шаги обучения для а и б = 0.1, коэффициент а уменьшается на 20% после каждого примера. Точность на обучающих и тестовых данных представлена на рисунке 6.2 – 6.3.

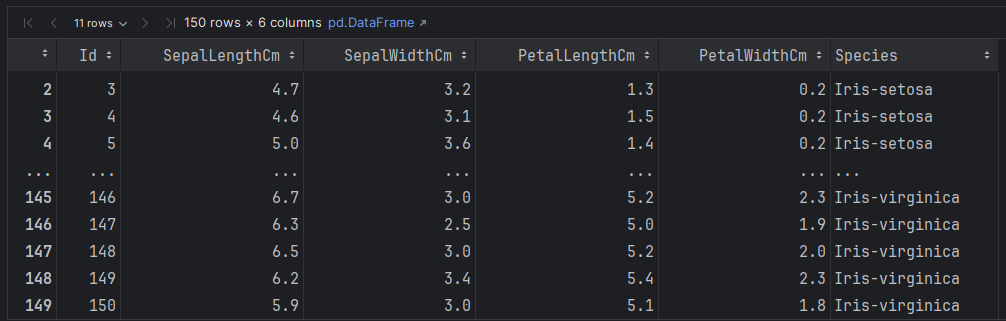


Рисунок 6.1 – Датасет

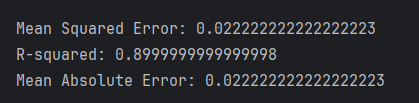


Рисунок 6.2 – Точность на обучающих данных

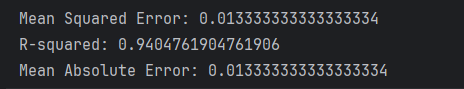


Рисунок 6.3 – Точность на тестовых данных

7. РЕКУРЕНТНЫЕ НЕЙРОННЫЕ СЕТИ

7.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать рекуррентную нейронную сеть.

7.2 Описание рекуррентной нейронной сети

Рекуррентная нейронная сеть — это тип нейронной сети, в которой связи между нейронами образуют направленные петли. Благодаря этому рекуррентная нейронная сеть может обрабатывать последовательные данные, такие как временные ряды или речевые сигналы.

**Структура.**

Рекуррентная нейронная сеть обычно состоит из следующих слоев:

* Входной слой: Этот слой получает последовательные данные.
* Скрытый слой: Этот слой состоит из нейронов, которые обрабатывают последовательные данные.
* Выходной слой: Этот слой производит выходной сигнал.

Нейронные клетки скрытого слоя имеют нелинейные функции активации, такие как сигмоида или гиперболический тангенс.

**Обучение.**

Обучение РНС происходит следующим образом:

* На вход сети подается последовательность обучающих данных.
* Сеть производит выходной сигнал.
* Сравнивается выходной сигнал с целевым значением.
* Ошибка сети рассчитывается как разница между выходным сигналом и целевым значением.
* Веса сети обновляются таким образом, чтобы уменьшить ошибку сети.

Процесс обучения повторяется для всех обучающих последовательностей.

**Принцип работы.**

При обучении РНС веса сети настраиваются таким образом, чтобы минимизировать ошибку сети. Ошибка сети рассчитывается как разница между выходным сигналом сети и целевым значением.

Веса сети обновляются с помощью метода градиентного спуска. Градиентный спуск — это метод оптимизации, который использует градиент ошибки для обновления параметров сети.

7.3 Результат работы

Будем обучать нейронную сеть на своём датасете цены биткоина. Количество эпох = 50, изначальные шаги обучения = 0.01. Результат работы программы представлен на рисунке 7.1.

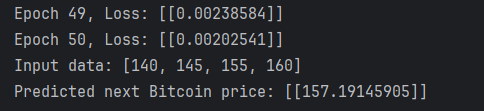


Рисунок 7.1 – Результат обучения рекуррентной нейронной сети

8. СВЁРТОЧНАЯ НЕЙРОННАЯ СЕТЬ

8.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать свёрточную нейронную сеть нейронную сеть.

8.2 Описание свёрточой нейронной сети

Свёрточная нейронная сеть (CNN) - это тип нейронной сети, которая использует сверточные операции для обработки данных. Свёрточные операции позволяют сети обнаруживать пространственные закономерности в данных, что делает их особенно эффективными для задач, связанных с изображениями и видео.

**Структура.**

CNN обычно состоит из следующих слоев:

* Входной слой: Этот слой получает изображения или видео.
* Свёрточный слой: Этот слой выполняет свёрточные операции над входными данными.
* Постсвёрточный слой: Этот слой выполняет операции над выходом свёрточного слоя.
* Выходной слой: Этот слой производит выходной сигнал.

Свёрточный слой состоит из нейронов, которые имеют локальную область ввода. Свёрточная операция выполняется следующим образом:

* Каждый нейрон свёрточного слоя принимает на вход область ввода.
* Веса каждого нейрона свёрточного слоя умножают соответствующие элементы области ввода.
* Результаты умножения суммируются.
* Суммирование производится с использованием функции активации.

**Обучение.**

Обучение CNN происходит следующим образом:

* На вход сети подается обучающее изображение или видео.
* Сеть производит выходной сигнал.
* Сравнивается выходной сигнал с целевым значением.
* Ошибка сети рассчитывается как разница между выходным сигналом и целевым значением.
* Веса сети обновляются таким образом, чтобы уменьшить ошибку сети.

Процесс обучения повторяется для всех обучающих изображений или видео.

**Принцип работы.**

При обучении CNN веса сети настраиваются таким образом, чтобы минимизировать ошибку сети. Ошибка сети рассчитывается как разница между выходным сигналом сети и целевым значением.

Веса сети обновляются с помощью метода градиентного спуска. Градиентный спуск — это метод оптимизации, который использует градиент ошибки для обновления параметров сети.

8.3 Результат работы

Будем обучать нейронную сеть на датасете Fashion-MNIST (Рисунок 8.1).

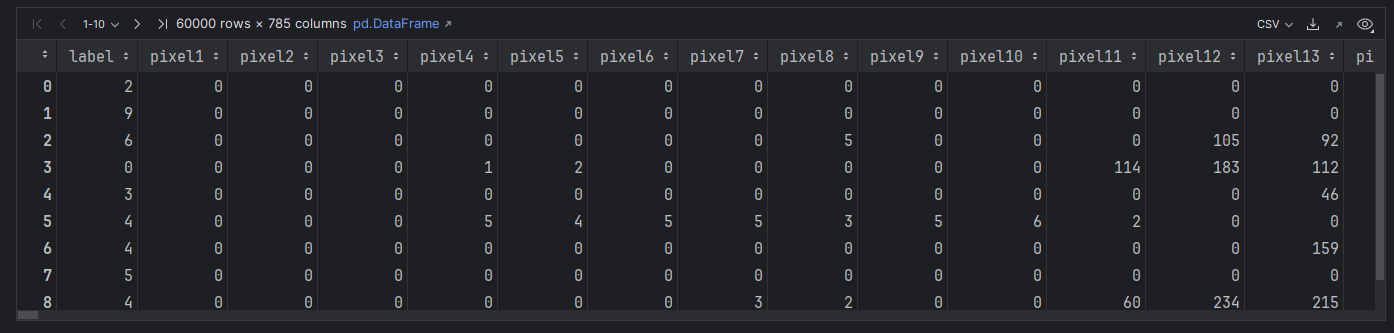


Рисунок 8.1 – Датасет

Обучаться будем всего 1 эпоху, с шагом обучения 0.1 (Рисунок 8.2).

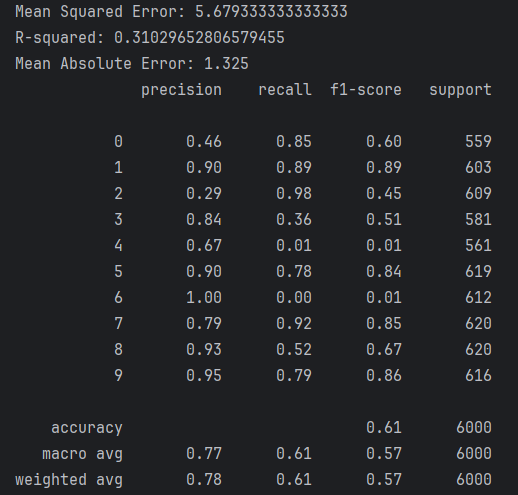


Рисунок 8.2 – Точность предсказаний нейросети



Рисунок 8.3 – Confusion Матрица

9. ТРАНСФОРМЕР

9.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать трансформер.

9.2 Описание трансформера

Трансформер — это тип нейронной сети, который использует механизм внимания для обработки последовательностей данных. Механизм внимания позволяет сети учитывать все элементы последовательности одновременно, что делает его особенно эффективным для задач, связанных с естественным языком.

**Структура.**

Трансформер обычно состоит из следующих слоев:

* Входной слой: Этот слой получает последовательность данных.
* Механизм внимания: Этот слой выполняет операцию внимания над входными данными.
* Выходной слой: Этот слой производит выходной сигнал.

Механизм внимания работает следующим образом:

* Для каждого элемента последовательности рассчитывается внимание к другим элементам последовательности.
* Внимание рассчитывается с использованием функции внимания.
* Отчеты внимания используются для обновления выходов механизма внимания.

**Обучение.**

Обучение трансформера происходит следующим образом:

* На вход сети подается обучающая последовательность данных.
* Сеть производит выходной сигнал.
* Сравнивается выходной сигнал с целевым значением.
* Ошибка сети рассчитывается как разница между выходным сигналом и целевым значением.
* Веса сети обновляются таким образом, чтобы уменьшить ошибку сети.

Процесс обучения повторяется для всех обучающих последовательностей.

**Принцип работы.**

При обучении трансформера веса сети настраиваются таким образом, чтобы минимизировать ошибку сети. Ошибка сети рассчитывается как разница между выходным сигналом сети и целевым значением.

Веса сети обновляются с помощью метода градиентного спуска. Градиентный спуск — это метод оптимизации, который использует градиент ошибки для обновления параметров сети.

9.3 Результат работы

Будем обучать трансформер на данных о температуре (Рисунок 9.1).

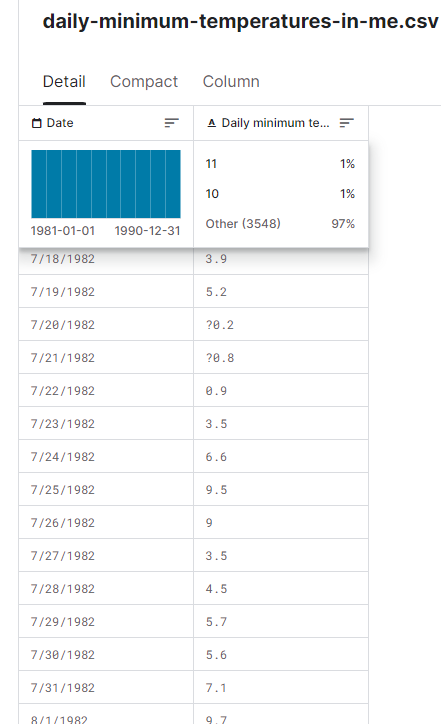


Рисунок 9.1 – Датасет

Обучать датасет будем 100 эпох, результат обучения Рисунок 9.2.

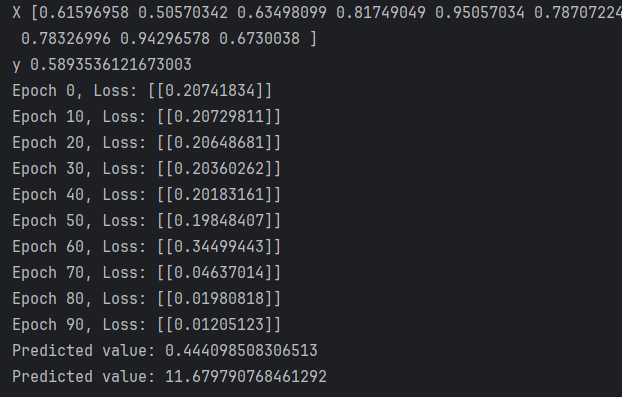


Рисунок 9.2 – Результат работы программы

10. ГРАФОВЫЕ НЕЙРОННЫЕ СЕТИ

10.1 Постановка задачи

Пользуясь лекционными материалами и методическими указаниями реализовать графовую нейронную сеть.

10.2 Описание графовую нейронную сеть

Графовая нейронная сеть — это тип нейронной сети, которая может обрабатывать данные, представленные в виде графа. Граф представляет собой набор узлов, связанных между собой ребрами. Узлы могут представлять объекты, а ребра - отношения между объектами.

**Структура.**

Графовая нейронная сеть обычно состоит из следующих слоев:

* Входной слой: Этот слой получает данные, представленные в виде графа.
* Слой распространения: Этот слой распространяет информацию по графу.
* Выходной слой: Этот слой производит выходной сигнал.

Слой распространения может быть реализован с использованием различных методов, таких как распространение с весами, распространение с использованием градиента и распространение с использованием внимания.

**Обучение.**

Обучение графовая нейронная сеть происходит следующим образом:

* На вход сети подается обучающий граф.
* Сеть производит выходной сигнал.
* Сравнивается выходной сигнал с целевым значением.
* Ошибка сети рассчитывается как разница между выходным сигналом и целевым значением.
* Веса сети обновляются таким образом, чтобы уменьшить ошибку сети.

Процесс обучения повторяется для всех обучающих графов.

**Принцип работы.**

При обучении трансформера веса сети настраиваются таким образом, чтобы минимизировать ошибку сети. Ошибка сети рассчитывается как разница между выходным сигналом сети и целевым значением.

Веса сети обновляются с помощью метода градиентного спуска. Градиентный спуск — это метод оптимизации, который использует градиент ошибки для обновления параметров сети.

10.3 Результат работы