УДК 004

Решение задачи на нахождение стоимости с помощью нейронной сети

Стрельцова Марина Николаевна
Приамурский государственный университет им. Шолом-Алейхема
Студент

Научный руководитель:
Баженов Руслан Иванович
Приамурский государственный университет им. Шолом-Алейхема
к.п.н., доцент, зав. кафедрой информационных систем, математики и правовой информатики

Аннотация
В последние десятилетия в мире бурно развивается новая прикладная область математики, специализирующаяся на искусственных нейронных сетях. В данной статье решена задача о прогнозировании цены холодильника с помощью ресурса Google Colaboratory и библиотеки scikit-learn с применением модуля sklearn.neural_network на языке программирования python.
Ключевые слова: нейронная сеть, scikit-learn, параметры, предсказания, Google Colaboratory.

Solving the problem of finding the cost using a neural network

Streltsova Marina Nikolaevna
Sholom-Aleichem Priamursky State University
Student

Scientific adviser:
Bazhenov Ruslan Ivanovich
Sholom-Aleichem Priamursky State University
Candidate of pedagogical sciences, associate professor, Head of the Department of Information Systems, Mathematics and Legal Informatics

Abstract
In recent decades, a new applied field of mathematics specializing in artificial neural networks has been rapidly developing in the world. This article solves the problem of forecasting the price of a refrigerator using the Google Colaboratory resource and the scikit-learn library using the sklearn.neural_network module in the python programming language.
Keywords: neural network, scikit-learn, parameters, predictions, Google Colaboratory.

В последние десятилетия в мире бурно развивается новая прикладная область математики, специализирующаяся на искусственных нейронных сетях. Актуальность исследований в этом направлении подтверждается массой различных применений нейросетей. Это автоматизация процессов распознавания образов, адаптивное управление, аппроксимация функционалов, прогнозирование, создание экспертных систем, организация ассоциативной памяти и многие другие приложения. С помощью нейросетей можно, например, предсказывать стоимость холодильника, основываясь на имеющихся данных.

Н. И. Червяков и Э. Е. Тихонов показывают результаты анализа и сравнения методов прогнозирования с методами прогнозирования на нейронных сетях. Рассматривают вопросы определения структуры и выбора типа нейронной сети для задач прогнозирования [1]. Д. В. Голов, Л. В. Красовская в статье рассматривают область применения нейронных сетей и приводят концепцию глубокого обучения [2]. А. А. Григорова, М. Б. Стоянова в своей работе исследовали применение искусственных нейронных сетей для решения задач прогнозирования, охарактеризовали нейронные сети, применяемые в современных информационных технологиях, эффективные алгоритмы их тренировки [3]. Д. В. Плотников, Е. А. Сопов в своей статье превели экспериментальное исследование эффективности работы сверточных нейронных сетей при решении задач распознавания лица и мимики человека [4]. И. А. Дохтаева, А. А. Суконщиков свою статью посвящают рассмотрению программы Scilab и её возможностей для решения задач по нечёткой логике и нейронным сетям. Разрабатывают комплекс лабораторных работ по предмету «Математические основы искусственного интеллекта» [5]. C. Seibold и другие в своей работе предлагают нейросетевые обучающие схемы, которые основаны на различных чередованиях обучающих данных, чтобы повысить работоспособность [6]. X. Xu, D. Cao, Y. Zhou, J. Gao в статье кратко представляют исследования и применение интеллектуальных технологий в диагностике неисправностей оборудования, а также описывается преимущества применения нечеткой нейросетевой технологии в диагностике неисправностей оборудования и излагаются основы нечеткой теории и нейросетевой технологии [7].

Целью данного исследования является решение задачи на нахождение стоимости с помощью нейронных сетей.

Для решения задачи будет использоваться библиотека scikit-learn - бесплатная библиотека машинного обучения для языка программирования Python [9].

Рассмотрим задачу для решения: имеются некоторые параметры холодильника: энергопотребление (e), объем(v), уровень шума(t) и цена(p). Необходимо на основание данных из таблицы 1, спрогнозировать цену холодильника при объеме 177, энергопотребление 202, уровне шума 39.

Таблица 1 – Данные для задачи

№	v	e	t	p
1	429	347.0	42	34999
2	93	109.0	42	8999
3	260	259.0	39	23499
4	207	217.0	40	18999
5	298	363.5	43	29999
6	269	442.0	39	31999
7	276	281.0	40	24999
8	45	105.0	42	6499
9	302	277.0	39	25999
10	364	417.0	42	34999
11	156	347.0	43	22999
12	239	267.0	40	22999
13	510	405.0	43	40999
14	341	309.0	36	28999
15	167	206.0	41	16999
16	310	467.0	39	34499
17	65	109.0	42	7999
18	311	272.0	40	25999
19	384	325.0	36	31999
20	93	109.0	42	8999
21	222	245.0	42	19999
22	240	301.0	40	20999
23	180	215.0	43	17999
24	308	286.0	40	24999
25	331	294.0	40	26999
26	301	326.0	40	27999
27	351	292.0	38	28999
28	256	342.0	43	26499
29	318	343.0	39	29999
30	223	284.0	40	22499

Решим следующую задачу с помощью нейронных сетей и онлайн ресурса Google Colaboratory [8]. Google Colaboratory — это бесплатный облачный сервис на основе Jupyter Notebook, который предоставляет всё необходимое для машинного обучения прямо в браузере.

В начале работы с Google Colab необходимо подключить библиотеки: работа с математическими функциями, массивами; работа с данными; нейросеть, модель-регрессия; загрузка файла с локального компьютера (Рис.1).

```
[ ]  # импорт библиотек
     import numpy as np    # работа с мат.функциями, массивами
     import pandas as pd   # работа с данными
     from sklearn.neural_network import MLPRegressor  # нейросеть, модель-регрессия
     from google.colab import files # загрузка файла с локального компьютера
```

Рисунок 1 – Импорт библиотек

Следующим шагом будет подключение функции выбора файлов для загрузки данных, после прописываем функцию чтения Excel файла в переменную df. Так же в этой части кода прописываем переменную Z с параметрами, которые нужно найти (Рис.2). Для того чтобы данные правильно загрузить их необходимо преобразовать в специальный вид (в Excel записать данные через запятую) и сохранить в .csv формате.

```
[ ]   uploaded = files.upload()
      df = pd.read_csv('holodos123.csv')
      Z=np.array([[177,202,39]])
```

Рисунок 2 – Загрузка данных

После загрузки файла с данными можно просмотреть, что загрузилось. Для этого прописываем df.head() в коде и далее будут отображены первичные пять строчек данных, если же вести просто df то покажется таблица со всеми загруженными данными (Рис.3).

```
df.head()
```

	v	e	t	p
0	429	347.0	42	34999
1	93	109.0	42	8999
2	260	259.0	39	23499
3	207	217.0	40	18999
4	298	363.5	43	29999

Рисунок 3 – Просмотр загруженных данных

Теперь необходимо разделить таблицу с данными на две таблицы. В первой таблице “X” будут отображаться входные данные, а это параметры холодильника: энергопотребление, объем, уровень шума. А во второй “y” выходные данные, здесь будет отображаться только цена. Чтобы это сделать необходимо прописать код представленный на рисунке 4. С помощью

X.head() и y.head() можно посмотреть первые строчки вывода данных без цены и только цены.

```
[5] X = df.drop('p',axis=1) # входные - удаление столбца p, так как p - целевой
    y = df['p'] # выходной - только столбец p, так как p - целевой
```

```
[6] X.head()
```

	v	e	t
0	429	347.0	42
1	93	109.0	42
2	260	259.0	39
3	207	217.0	40
4	298	363.5	43

```
y.head()
```

```
0    34999
1     8999
2    23499
3    18999
4    29999
Name: p, dtype: int64
```

Рисунок 4 – Разделение на входные и выходные данные

Далее подключаем класс для разделения на обучающую и тестовую выборки. X_test показывает случайную выборку для обучения. Код показан на рисунке 5.

```
[ ] from sklearn.model_selection import train_test_split # для разделения на обучающую и тестовую выборки
    X_train, X_test, y_train, y_test = train_test_split(X, y) # разделение на обучающую и тестовую выборки.
    X_test
```

	v	e	t
6	276	281.0	40
3	207	217.0	40
15	310	467.0	39
1	93	109.0	42
5	269	442.0	39
13	341	309.0	36
20	222	245.0	42
19	93	109.0	42

Рисунок 5 – Разделение на обучающую и тестовую выборки

Следующим шагом будет импортирование класса для масштабирования, так как нейронная сеть является чувствительной. Распишем подробнее каждую строчку кода:

- scaler = StandardScaler() вызов класса для нормализации
- scaler.fit(X_train) применение метода fit для вычисления среднего значения и стандартного отклонения для последующего масштабирования
- X_train = scaler.transform(X_train) нормализация массива тренировочной выборки
- X_test = scaler.transform(X_test) нормализация массива тестирующей выборки.

Расписанный код изображен на рисунке 6.

```
[10]
    из  sklearn.preprocessing  import  StandardScaler  # для масштабирования, т.к. нейросеть  чувствительна
    # Стандартная оценка выборки x вычисляется a s: z = (x - u) / s
    # где u - среднее значение обучающих выборок или z ero, если with_mean = False,
    #and s - стандартное отклонение амплитуд обучения или единица, если with_std = False.
    scaler = StandardScaler ()  # начало нормализации,

    scaler.fit (X_train)
    X_train = scaler.transform (X_train)
    X_test = scaler.transform (X_test)
    Z = scaler.transform (Z)
    Z

    array([[-0.83546938, -0.8395359 , -0.71340474]])
```

Рисунок 6 – Масштабирование и нормализация

Далее объявляем параметры для обучения нейронной сети. Каждый параметр необходимо настраивать под данные из задачи. Рассмотрим подробнее значение каждого задаваемого параметра:

- Activation – это функция активация для скрытого слоя. В данном случае используется функция 'relu'
- Solver – это решатель для оптимизации веса. В данном случае используется 'lbfgs', так как данный параметр лучше подходит для небольших наборов данных.
- Random-state – определяет генерацию случайных чисел для весов. В данном случае стоит число 42, так оно является особенным и подходит для большинства случаев.
- Max_iter – это максимальное число итераций, по умолчанию равно 200, в данном случае используется 10000 итераций.
- Hidden_layer_sizes – количество нейронов в одном элементе скрытого слоя (у нейронной сети три слоя). Для данного случая используются два слоя со значениями 4, 4.

Остальные параметры MLPRegressor задаются по умолчанию. Описания параметров взято с официального сайта библиотеки scikit-learn.org [9].

Строчка кода MLP.fit(X_train,y_train) обозначает обучение нейросети на обучающих данных. Обучение нейронной сети представленно на рисунке 7.

```
MLP=MLPRegressor(activation='relu', solver ='lbfgs', random_state=42, max_iter=10000, hidden_layer_sizes=(4,4) )
MLP.fit(X_train,y_train) # обучение нейросети
```

```
MLPRegressor(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
             beta_2=0.999, early_stopping=False, epsilon=1e-08,
             hidden_layer_sizes=(4, 4), learning_rate='constant',
             learning_rate_init=0.001, max_fun=15000, max_iter=10000,
             momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True,
             power_t=0.5, random_state=42, shuffle=True, solver='lbfgs',
             tol=0.0001, validation_fraction=0.1, verbose=False,
             warm_start=False)
```

Рисунок 7 – Обучение нейросети

На рисунке 8 показан код для предсказания цен из тестовой выборки, и показаны реальные цены из выборки.

```
[ ] predictions = MLP.predict(X_test) # предсказание
```

```
[ ] predictions
```

```
array([24304.69329234, 19017.62035019, 35341.68985734,  9488.2596701 ,
       32028.96893307, 29738.93780168, 19876.71238631,  9488.2596701 ])
```

```
[ ] y_test # известные значения
```

```
6     24999
3     18999
15    34499
1      8999
5     31999
13    28999
20    19999
19     8999
Name: p, dtype: int64
```

Рисунок 8 – Предсказание цен

Проверим предсказания на ошибки. Для этого подключаем два класса: средняя абсолютная ошибка и средняя квадратичная ошибка. После вводим код для выведения данных ошибок (Рис.9).

```
[ ] from sklearn.metrics import mean_absolute_error # средняя абсолютная ошибка
    from sklearn.metrics import mean_squared_error # средняя квадратичная ошибка

[ ] mean_squared_error(y_test, predictions)

    279330.63934277097

[ ] mean_absolute_error(y_test, predictions)

    428.2913254779428
```

Рисунок 9 – Проверка на ошибки

На рисунке 9 видно, что нейронная сеть практически точно предсказывает стоимость. Так данные большие, минимальное значение абсолютной ошибки, которое удалось подобрать после обучения равно 428.

В задаче необходимо было спрогнозировать стоимость холодильника при объеме 177, энергопотребление 202, уровне шума 39. Данные были введены в начале кода и показаны на рисунке 2. А код для предсказания цены для заданных параметров холодильника представлен на рисунке 10.

```
predictions = MLP.predict(Z)
predictions

array([17259.06202719])
```

Рисунок 10 – Код предсказания стоимости

Стоимость холодильника при объеме 177, энергопотребление 202, уровне шума 39 примерно равна 17259р.

В настоящее время нейронные сети становятся всё популярнее, а область их применения растет с огромной скоростью. Сложно представить решение трудных задач, требующих огромных расчетов, без помощи нейронных сетей. Знания и умения работать с нейронными сетями будут полезны для людей, которые хотят связать свой трудовой путь с программированием.

В данной статье было рассмотрено решение задачи о нахождение стоимости холодильника с помощью ресурса Google Colaboratory и библиотеки scikit-learn с применением модуля sklearn.neural_network на языке программирования python.

Библиографический список

1. Червяков Н. И., Тихонов Э. Е. Применение нейронных сетей для задач прогнозирования и проблемы идентификации моделей прогнозирования на нейронных сетях //Нейрокомпьютеры: разработка, применение. 2003.

№. 10-11. С. 25-31.
2. Голов Д. В., Красовская Л. В. Нейронные сети и распознание рукописных цифр на основе искусственных нейронных сетей //Исследования технических наук. 2014. №. 4. С. 18-20.
3. Григорова А. А., Стоянова М. Б. Искусственные нейронные сети и особенности их построения для решения задач прогнозирования //Вестник Херсонского национального технического университета. 2017. №. 4 (63) С. 160-166.
4. Плотников Д. В., Сопов Е. А. Решение задач распознавания лиц и мимики с помощью сверточных нейронных сетей //Решетневские чтения. 2017. №. 21-2. С. 234-236.
5. Дохтаева И. А., Суконщиков А. А. Прогаммная система Scilab для решения задач по нечёткой логике и нейронным сетям //Информатизация инженерного образования. 2016. С. 114-117.
6. Seibold C. et al. Accurate and robust neural networks for face morphing attack detection //Journal of Information Security and Applications. 2020. Т. 53. С. 102526.
7. Xu X. et al. Application of neural network algorithm in fault diagnosis of mechanical intelligence //Mechanical Systems and Signal Processing. 2020. С. 106625.
8. Google Colaboratory URL: https://colab.research.google.com/ (дата обращения: 04.06.2020).
9. Scikit-learn.org URL: https://scikit-learn.org/ (дата обращения: 04.06.2020).