УДК 004

Расчет стоимости кондиционера с помощью библиотеки scikit-learn

Семченко Регина Викторовна
Приамурский государственный университет имени Шолом-Алейхема
студент

Еровлев Павел Андреевич
Приамурский государственный университет имени Шолом-Алейхема
студент

Аннотация
В данной статье описан процесс создания нейросети с помощью библиотеки scikit-learn. Создана база данных для обучения из характеристик и цен 50 кондиционеров. Выходными данными будет цена кондиционера по заданным характеристикам.
Ключевые слова: scikit-learn, python, нейросеть, расчет цены

Calculating the cost of an air conditioner using the scikit-learn library

Semchenko Regina Viktorovna
Sholom-Aleichem Priamursky State University
student

Erovlev Pavel Andreevich
Sholom-Aleichem Priamursky State University
student

Abstract
This article describes the process of creating a neural network using the scikit-learn library. A database for training was created from the characteristics and prices of 50 air conditioners. The output will be the price of the air conditioner according to the specified characteristics.
Keywords: scikit-learn, python, neural network, price calculation

Scikit-learn – это библиотека для языка Python, для машинного обучения. Она имеет различные алгоритмы классификации, регрессии и кластеризации.

Цель данной статьи разработать нейросеть с использованием библиотеки scikit-learn и создать базу данных для обучения.

М.С. Артамошкин приведел краткий обзор Python-библиотеки Scikit-learn, с примерами ее использования. Также рассмотрел ее применение на реальном примере, где проводится анализ клиентов интернет-магазина для оптимизации продаж[3]. Д.С.Кокорев и Д.Б.Степаненко в своей статье

сделали глобальный обзор на библиотеку scikit-learn и возможности обучения нейросети [2]. В своей статье Д.В. Климов посвятил вопросы машинного обучения в области компьютерной лингвистики, в частности классификации неструктурированных потоков текстовых сообщений[5]. С.В. Литвиненко разработал платформу для образования "Познавательная реальность", цель которой внедрить в образовательный процесс модели адаптивного обучения с помощью нейросети[4]. К.Н. Гребнев рассмотрел в своей статье задачи распознавания сорта вина на основании результатов химического анализа, рассмотрел возможности библиотеки scikit-learn языка Python [1].

Сначала загружаем нужные библиотеки. Они будет нужны для расчета математических функций, для работы с данными, для загрузки файлов с локального компьютера, и регрессионную модель нейросети. Далее пишем функцию загрузки файла с локального компьютера в систему и считываем его. Делителем будет служить «;», поэтому укажем это в параметре, для точного построения таблицы.

Следующим шагом будет нахождение имени «price» и при условии, что «axis» равен единице, то строка превращается в столбец и удаляется столбец. Следом выводим значения без столбца «price» и отдельно выводим сам столбец «price». Далее разделяем данные на данные для обучения нейросети и данные для теста. Данные для теста нейросеть не будет знать и поэтому есть возможность сравнить ее с правильным ответом.

Так как данные разделяются по умолчанию, то происходит это в соотношении 75% на 25%, меньшее для проведения тестов. Следом необходимо нормализовать заданные данные, что более правильного считывания нейросетью, для этого преобразуем числа, где значения будут от 0 до 1.

Теперь напишем модель нейросети, возьмем «MLPRegressor» - эта модель оптимизирует квадратичные потери с использованием LBFGS или стохастического градиентного спуска, выставим значение нейронов на 10 в первом слое, 100 во втором слое, и 10 в третьем слое. Следующим шагом будет обучение нейросети, нужно будет прогнать через нее все эти данные. Потом возьмем тестовое значение и предскажем его ответ, а далее найдем квадратичную и абсолютную ошибку (рис.1-2).

```
# 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

uploaded = files.upload() #Загрузка файла в историю
df = pd.read_csv('sum1.csv', delimiter=';') #чтение файла и разделение знаков

df.head()  #чтение первых строк

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

X.head()

y.head()

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. Значения разделения по умолчанию


from sklearn.preprocessing import StandardScaler

# для масштабирования, т.к. нейросеть чувствительна
# Стандартная оценка образца x рассчитывается как: z = (x - u) / s
# где u - среднее значение обучающих выборок или ноль, если with_mean = False,
# и s - стандартное отклонение обучающих выборок или единица, если with_std = False.

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

X_train
```

Рисунок 1 – Код нейросети

```
MLP=MLPRegressor(solver='lbfgs', random_state=42, max_iter=10000, hidden_layer_sizes=(10,100,10) )
# 'lbfgs' - оптимизатор в семействе квазиньютоновских методов.
# Устанавливаем случайное состояние на 42
# Кол-во итераций 10 000
# 3 слоя по 10,100 и 10 нейронов

MLP.fit(X_train,y_train) # обучение нейросети

predictions = MLP.predict(X_test) # предсказание

predictions

y_test # известные значения

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)

mean_absolute_error(y_test, predictions)
```

Рисунок 2 - Код нейросети

Для проверки правильно ответа с выходным значением, необходимо создать базу данных, они основываются на 4 параметрах и 5 параметр это цена, данные составлены таким образом, что значения, которые формулируются как «да» или «нет», заменены за «1» и «0» соответственно (Рис.3).

Heat	energy	night_mode	timer	price
1	500	0	1	10000
1	700	0	1	14000
0	600	0	1	6000
0	650	1	1	13000
1	750	0	1	15000
0	550	1	1	11000
1	500	0	1	10000
1	400	1	1	12000
1	800	0	0	8000
1	900	0	0	9000
1	850	1	0	17000
0	850	0	0	5000
0	800	1	0	8000
1	500	0	0	5000
1	700	1	1	21000
1	600	0	1	12000
0	650	1	0	6500
1	750	0	1	15000
1	550	1	1	16500
0	500	1	1	10000
1	400	1	1	12000
0	800	1	0	8000
1	900	1	1	27000

Рисунок 3 – База данных

Теперь осталось проверить правильность выходных значений с действительным.

```
prod = np.array([[1,1000,0,1]]) #Добавить свои параметры
prod = scaler.transform(prod) #Нормализируем
predictions = MLP.predict(prod) #Расчитываем
predictions #Выводим
```

```
array([17808.77612804])
```

Рисунок 4 – Вывод данных

Исходя из примерных расчетов цен на кондиционеры, получаем примерное значение для данных параметров.

В данной статье была реализована нейросеть и база данных для ее обучения, а так же произведен тест и получен результат.

Библиографический список

1. Артамошкин М.С. Принятие решений в электронной коммерции с использованием библиотеки scikit-learn // Автоматика. Вычислительная техника. 2012. №1. С. 24-30.
2. Кокорев Д.С., Степаненко Д.Б. Scikit-learn: машинное обучение в python //

Труды Международного симпозиума «Надежность и качество». 2014. №5. С. 14-20.
3. Литвиненко С.В. Образовательная платформа "познавательная реальность" для адаптивного обучения средствами виртуальной и дополненной реальности дисциплинам естественно-научного цикла и возможность интеграции с продуктами фирмы "1С" // Современные научные исследования и инновации. 2017. № 1 . С. 75-79.
4. Климов Д.В. Предобработка текстовых сообщений для метрического классифкатора // Труды Международного симпозиума «Надежность и качество». 2017. №2. С. 17-25.
5. Гребнев К.Н. Машинное обучение с помощью библиотеки scikit-learn языка python // Современные научные исследования и инновации. 2017. № 7-5 (43). С. 47-55.