УДК 004

Расчет стоимости компьютера 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 50 компьютеров. Выходными данными будет цена компьютера по заданным характеристикам.
Ключевые слова: scikit-learn, python, нейросеть, расчет цены

Calculating the cost of a computer 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 of 50 computers has been created. The output will be the price of the computer for the given characteristics.
Keywords: scikit-learn, python, neural network, price calculation

Scikit-learn – является свободным программным обеспечением машинного обучения библиотека для Python языка программирования. Он имеет различные алгоритмы классификации, регрессии и кластеризации, включая машины опорных векторов, случайные леса, усиление градиента, k - средства и DBSCAN, и предназначен для взаимодействия с числовыми и научными библиотеками Python NumPy и SciPy.

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

К.Н. Гребнев рассмотрел в своей статье задачи распознавания сорта вина на основании результатов химического анализа, рассмотрел возможности библиотеки scikit-learn языка Python[1]. Д.С. Кокорев и Д.Б.

Степаненко в своей статье сделали глобальный обзор на библиотеку scikit-learn и возможности обучения нейросети[2]. М.С. Артамошкин привел 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]. С.В. Литвиненко разработал платформу для образования "Познавательная реальность", цель которой внедрить в образовательный процесс модели адаптивного обучения с помощью нейросети[4]. В своей статье Д.В. Климов посвятил вопросы машинного обучения в области компьютерной лингвистики, в частности классификации неструктурированных потоков текстовых сообщений [5].

Для начала необходимо будет загрузить нужные библиотеки. Они будет нужны для расчета математических функций, для работы с данными, для загрузки файлов с локального компьютера, и регрессионную модейль нейросети (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

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 (рис.2).

```
uploaded = files.upload() #Загрузка файла в историю
df = pd.read_csv('sum2.csv', delimiter=';') #чтение файла и разделение знаков

df.head()  #Вывод первых 5 строк
```

Рисунок 2 – Загрузка файлов

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» (рис.3).

```
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() # Вывод таблицы без столбца price

y.head() # Вывод столбца price
```

Рисунок 3 – Удаление цены

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 (рис.4).

```
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
```

Рисунок 4 – Разделение на тест и обучение

Следом необходимо нормализовать заданные данные, что более правильного считывания нейросетью, для это преобразуем числа, где значения будут от 0 до 1 (рис.5).

```
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 5 – Нормализация данных

Теперь напишем модель нейросети, возьмем «MLPRegressor» - эта модель оптимизирует квадратичные потери с использованием LBFGS или стохастического градиентного спуска и выглядить она будет вот так (рис.6).

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

Рисунок 6 – Модель нейросети

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.7)

```
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 7 – Обучение нейросети

Для проверки правильно ответа с выходным значением, необходимо создать базу данных, они основываются на 5 параметрах и 6 параметр это цена (рис.8).

cpu	ram	graphics	hdd	hdd_or_ssd	price
6	12	2	1000	1	40000
5	8	1	500	1	14000
7	32	6	2000	0	180000
4	4	1	250	0	4500
5	4	1	250	1	5000
6	8	2	1000	1	32000
4	2	1	500	1	7000
3	2	1	250	1	2756
2	1	0	250	0	1500
7	16	4	2000	0	108000
4	8	1	500	1	13000
2	2	0	250	0	2000
3	2	0	250	0	2500
5	8	0	500	1	13000
7	16	2	500	0	25000
7	8	4	500	0	19000
5	8	4	500	0	17000
2	2	0	500	1	4000
3	2	0	250	1	2500
5	4	2	250	0	5500
3	4	1	250	0	4000
5	4	0	500	1	9000
6	4	4	500	0	14000
5	4	1	500	0	10000

Рисунок 8 – База данных

Теперь осталось проверить правильность выходных значений с действительным (рис.9).

```
prod = np.array([[5,4,1,1,500]]) #Добавить свои параметры
prod = scaler.transform(prod) #Нормализируем
predictions = MLP.predict(prod) #Расчитываем
predictions #Выводим
```

```
array([10000.001225])
```

Рисунок 9 – Вывод

Исходя из примерных расчетов цен на компьютеры, получаем практически точно значение для данных параметров.

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

Библиографический список

1. Гребнев К.Н. Машинное обучение с помощью библиотеки scikit-learn языка python// Современные научные исследования и инновации. 2017. № 7-5 (43). С. 47-55.
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. Артамошкин М.С. Принятие решений в электронной коммерции с использованием библиотеки scikit-learn // Автоматика. Вычислительная техника. 2012. №1. С. 24-30.
4. 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.
5. Климов Д.В. Предобработка текстовых сообщений для метрического классифкатора // Труды Международного симпозиума «Надежность и качество». 2017. №2. С. 17-25.