УДК 004.855.5

РЕАЛИЗАЦИЯ КЛАССИФИКАТОРА ПРОДУКТОВ ПИТАНИЯ С ПОМОЩЬЮ МЕТОДА МАШИННОГО ОБУЧЕНИЯ

Бруевич Н.А.

Московский Технический Университет Связи и Информатики, Москва, e-mail: bruevich@mail.ru

Сложность определения требуемых для подходящей диеты продуктов питания состоит в том, что различные продукты могут иметь одинаковые показатели соотношения полезных веществ при одинаковой массе, однако оказывать на потребителя абсолютно различные эффекты – как положительные (поддержка или улучшение здоровья), так и отрицательные (ожирение, увеличенное давление и т.д.). В данной статье описана реализация инструмента классификации продуктов питания, использующего интеллектуальный анализ данных. Основной задачей является обучение алгоритма на основе входных значений продуктов питания (белки, жиры, углеводы, калорийность) методами машинного обучения. Важным этапом является проверка признаков продуктов питания на корреляцию в целью уменьшения значимых признаков, что ускорит выполнение обучения программы. Для проверки точности обучения модели использовался метод перекрестной проверки точности классификации. В качестве алгоритмов использовались: метод обратного распространения ошибки, выпрямленная линейная функция и функция обобщения логистической функции для многомерного случая. Для решения поставленной задачи был выбран язык программирования Python 3.6.4 и интерактивная оболочка Jupyter Notebook. Нейронная сеть, архитектура которой представляет собой многослойный персептрон с 2 слоями по 32 нейрона, разработана в библиотеке Skikit-learn. Данная работа выполнена в рамках курсового проекта по дисциплине «Методы интеллектуального анализа данных», научный руководитель – д.ф.-м.н., профессор Воронова Л.И.

Ключевые слова: Машинное обучение, нейронная сеть, интеллектуальные системы, классификация, skikit-learn

REALIZATION OF FOOD CLASSIFIER USING MACHINE-LEARNING METHOD

Bruevich N.A.

Moscow Technical University of Communications and Informatics, Moscow, e-mail: bruevich@mail.ru

The difficulty in determining the foods required for a fit diet is that different foods can have the same ratio of nutrients with the same mass, but have completely different effects on the consumer, both positive (supporting or improving health) and negative (obesity, increased pressure, etc.) This article describes the implementation of a food classification tool using data mining. The main task is to train the algorithm based on the input values of food (proteins, fats, carbohydrates, calories) using machine learning methods. An important step is to check the signs of food for correlation in order to reduce significant signs, which will speed up the implementation of the training program. To test the accuracy of the training model used the method of cross-validation of classification accuracy. The following algorithms were used as the algorithms: the error backpropagation method, the rectified linear function, and the generalization function of the logistic function for the multidimensional case. To solve this problem, the programming language Python 3.6.4 and the interactive shell Jupyter Notebook were chosen. The neural network, whose architecture is a multilayer perceptron with 2 layers of 32 neurons, was developed in the Skikit-learn library. This work was carried out as part of the course project on the discipline «Methods of Data Mining», the research supervisor is Dr. of Physics and Mathematics, Professor Voronova L.I.

Keywords: Machine learning, neural network, intelligent systems, classification, scikit-learn

При современном ритме жизни у малого процента населения есть время для соблюдения правильного режима питания, вследствие чего организм получает избыток вредных веществ и недостаток необходимых, что негативно влияет на здоровье. Существует немало средств, позволяющих следить за показателями здоровья и одного из основных его факторов – питания, и одним из лучших методов для данной задачи является, на мой взгляд, методы машинного обучения.

Машинное обучение – класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение модели и получение решений для вводимых условий в процессе применения решений множества сходных задач [1]. При применении данных методов система «обучается» самостоятельному вычислению требуемых выходных функций и применению новых данных к этим функциям с последующим выводом результатов вычислений**,** что позволяет снизить потребность вмешательства человека в технологические процессы.

Рассмотрим задачу классификации пищевых продуктов питания, используя для этого методы машинного обучения. Имеется набор данных, представляющий собой количество основных параметров на 100 г продукта [2].

В зависимости от значений входных признаков нейронная сеть должна относить пример к тому или иному классу из 6 продуктов, а именно колбасные изделия, крупы и каши, молочные продукты, фрукты, овощи и зелень и мясные продукты.

Также, программа должна показывать точность работы нейронной сети.

Для решения данной задачи воспользуемся подвидом нейронной сети – персептроном.

Персептрон, – нейронная сеть прямого распространения сигнала (без обратных связей), в которой входной сигнал преобразуется в выходной, проходя последовательно через один/несколько слоев. Присутствие нескольких скрытых слоев оправдано лишь в случае использования нелинейных функций активации.

Перед управлением входными данными требуется подключить библиотеки для работы с машинным обучением на языке Python, а именно Tensorflow, Scikit-learn, Itertools, Numpy, Pandas и Matplotlib [3].

Сохраняем файл data.xlsx в формате .csv. В результате получили датасет из 542 примеров, которые необходимо классифицировать по классу продукта, к которому он относится.

Импортируем файл data.csv, предоставив ему имя в коде filename, по которому идёт обращение и считывание данных файла [4].

Для проверки верного считывания данных из файла выводим список, содержащий признаки и выходные значения для обработки.

В качестве оптимального эмпирическим путем был выбрана архитектура модели, – персептрон с 2 скрытыми сломи по 32 нейрона в каждом – 4 входных и 6 выходных.

Выведем графики попарного отношения входных признаков между собой, что покажет взаимное отношение признаков друг к другу в различных классах. Данный метод может быть полезен при выборке наиболее важных отношений признаков для обнаружения главных признаков и/или их корреляций. В данном случае, наблюдается несколько графиков, где большая часть классов идентична на основе двух признаков (жиры-калории, углеводы-калории), однако поскольку признаки взаимозависимы (калорийность зависит от количества белков, жиров, углеводов) сжатие данных ухудшит показатели точности обучения.

Пример входных данных

Продукт	Белки, г	Жиры, г	Углеводы, г	Калории, ккал
Колбаса вареная диетическая	12.10	13.50	0.00	170.00
Колбаса вареная докторская	12.80	22.20	1.50	257.00
Колбаса вареная куриная	15.50	16.20	2.30	223.00

```
fileName = "data.csv"  # файл с данными или путь к нему
# считываем данные из файла
df = pd.read_csv(fileName,
                 header=0,
                 delimiter=";")
```

Рис. 1. Код импорта датасета

```
features = list(df.columns[:-1])   # ['proteins', 'fats',
label = df.columns[-1]
print(df.head())
```

```
   proteins  fats  carbohydrates  calories  label
0      12.1  13.5            0.0     170.0      0
1      12.8  22.2            1.5     257.0      0
2      15.5  16.2            2.3     223.0      0
3      12.2  28.0            0.1     301.0      0
4      11.7  22.8            0.2     252.0      0
```

Рис. 2. Код вывода данных, считываемых из файла data.csv

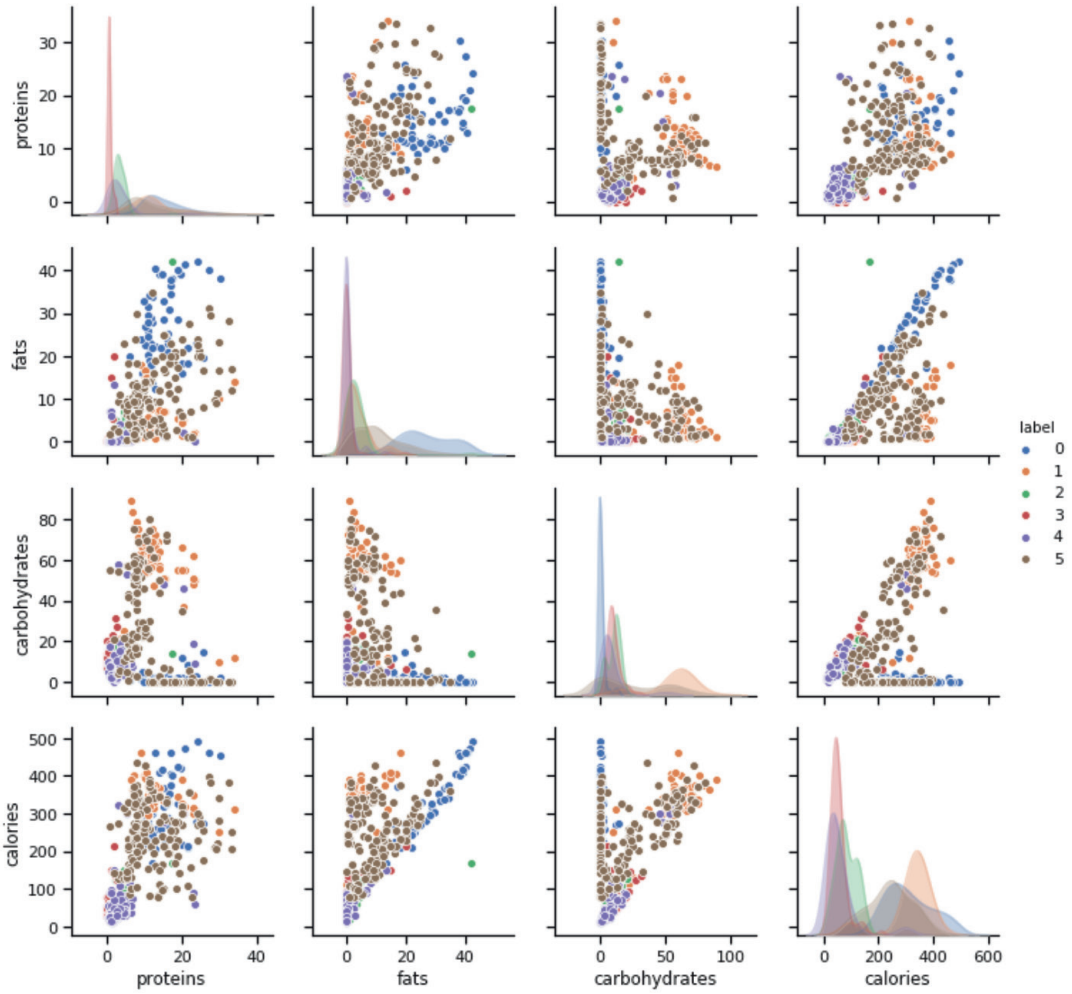

Рис. 3. Отображение классов в графах при попарной зависимости графиков от признаков

```
def create_model():
    # создание модели нейронной сети
    global count_neurons
    input_neurons = len(features)
    output_neurons = count_class
    model = Sequential()
    model.add(Dense(count_neurons, input_dim=input_neurons, activation='relu'))  # первый скрытый слой
    model.add(Dropout(0.1))  # отбрасываем часть связей между слоями для избежания переобучения модели
    model.add(Dense(count_neurons, activation='relu'))   # второй скрытый слой
    model.add(Dropout(0.1))
    model.add(Dense(output_neurons, activation='softmax')) #  выходной слой
    # компилируем модель
    model.compile(loss='categorical_crossentropy', optimizer="adam", metrics=['accuracy'])
    return model
```

Рис. 4. Код создания модели персептрона

В качестве первого входных параметров на слой поступают данные с предыдущего слоя, которые получают собственный вес функцией активации ReLU (функция «выпрямителя», образует переход выхода функции в единицу при аргументе =<0) [5], и затем случайно исключаются методом DropOut (с вероятностью, вводимой разработчиком, исключает каждый нейрон на слое, что позволяет избавиться от проблемы переобучения сети), что позволяет избежать проблемы переобучения сети.

Для обучения и проверки обучаемости изменим представление созданного датасета: используя команду «train_test_split», разделяем изначальный набор данных на 2 набора – тренировочный (X_train, y_train) и тестовый (X_text, y_text) в соотношении 2/1 [6]. После чего инициируем ранее созданную модель персептрона в качестве обработчика, и используя команду «.fit()», применяем в каждому примеру созданную модель в течение заданного заранее количества итераций (эпох) в процессе выполнения команды.

На основе обучающего датасета применим модель классификатора, описанную ранее, и проверим точность предсказаний классов на тренировочном наборе; результат выполнения программы можно увидеть на рисунке [7].

```
# Обучение нейронной сети
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)
model = create_model()
model.fit(X_train, y_train, epochs=epochs, batch_size=batch_size)
```

```
Epoch 1/100
334/334 [==============================] - 1s 2ms/step - loss: 1.5867 - acc: 0.3623
Epoch 2/100
334/334 [==============================] - 0s 578us/step - loss: 1.2120 - acc: 0.5090
Epoch 3/100
334/334 [==============================] - 0s 580us/step - loss: 1.0523 - acc: 0.5329
Epoch 4/100
334/334 [==============================] - 0s 550us/step - loss: 1.0131 - acc: 0.5509
Epoch 5/100
334/334 [==============================] - 0s 558us/step - loss: 0.9320 - acc: 0.6377
Epoch 6/100
334/334 [==============================] - 0s 566us/step - loss: 0.9264 - acc: 0.5868
Epoch 7/100
334/334 [==============================] - 0s 562us/step - loss: 0.8816 - acc: 0.6347
Epoch 8/100
334/334 [==============================] - 0s 550us/step - loss: 0.8347 - acc: 0.6497
Epoch 9/100
334/334 [==============================] - 0s 551us/step - loss: 0.8249 - acc: 0.6437
Epoch 10/100
```

Рис. 5. Код обучения нейронной сети и результаты первых итераций

```
# предсказываем класс объекта
y_pred = model.predict_classes(X_test)
y_inv = lb.inverse_transform(y_test)

print("Точность классификации составила: {:.2%}".format(accuracy_score(y_inv, y_pred)))
```

```
Точность классификации составила: 78.18%
```

Рис. 6. Код предсказывания классов тестового набора, вывод точности классификации

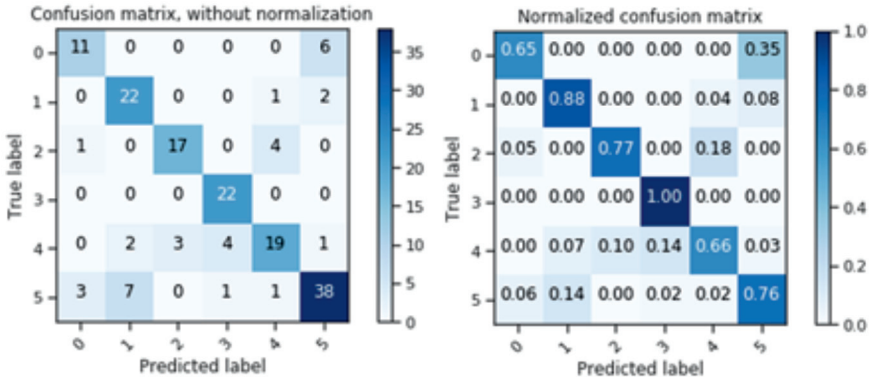

Рис. 7. Матрицы отношения предсказанных значений классов к истинным значениям

После выполнения данных операций строим матрицы точности, которые показывают отношение (нормализованное или первоначальное число) предсказанных классов к истинным значениям в тренировочном наборе, которые можно увидеть на рисунке ниже: данные матрицы дают представление о том, насколько велика доля примеров из тестового набора, классифицированных верно (по-вертикали) к общему числу тестовых примеров, представленных в датасете (по-горизонтали).

Данные матрицы показывают, что лучше всего обученная сеть предсказывает 4-й класс (фрукты), хуже всего предсказание сказывается на 1 м классе (колбасные изделия) из-за схожих показателей входных данных между 1 м классом и классом № 6 (мясные изделия).

Выводы

В данной статье была проанализирована проблема классификации продуктов питания методами машинного обучения; обозначены основные понятия; описан метод обучения на основе многослойного персептрона. Экспериментальной частью являлось написание программы для получения точности классификации на основе тестового набора данных. Полученная программа показывает 80 % точности классификации и позволяет визуализировать данные для удобной работы с ними.

Список литературы

1. Воронова Л.И., Воронов В.И. Machine Learning: регрессионные методы интеллектуального анализа данных: учебное пособие. М.: МТУСИ, 2018. 83 с.

2. Калорийность продуктов [Электронный ресурс]. Режим доступа: http://www.calorizator.ru/product (дата обращения: 01.02.2019).

2. Top 20 Python Libraries for Data Science in 2018 [Электронный ресурс]. Режим доступа: https://www.kdnuggets.com/2018/06/top-20-python-libraries-data-science-2018.html (дата обращения: 01.02.2019).

3. Python Software Foundation [Электронный ресурс]. Режим доступа: https://docs.python.org/2/library/csv.html (дата обращения: 01.02.2019).

4. Курс машинного обучения Andrew Ng Machine Learning [Электронный ресурс]. Режим доступа: https://www.coursera.org/learn/machinelearning (дата обращения: 01.02.2019).

5. Доусон М. Программируем на Python. СПб.: Питер, 2014. 416 с.

6. Рашка С. P28 Python и машинное обучение / пер. с англ. А. В. Логунова. М.: ДМК Пресс, 2017. 418 с.