**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ   
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра АСУ

Отчет

о лабораторной работе № 3

по дисциплине «Интеллектуальный анализ данных»

# на тему: «нейронные сети для решения задач классификации и регрессии»

Выполнил:

студент группы ИСТ-19б

Карпович В.Д.

Проверили:

Шуватова Е.А.

Васяева Т.А.

Донецк – 2022

**Цель работы:** познакомиться с понятием полносвязной нейронной сети, с понятием метрик качества моделей обучения, функциями потерь и оптимизаторами обучения, с понятием «переобучение» модели нейронной сети; научиться строить с нуля и обучать нейронную сеть с помощью Keras и TensorFlow.

**Вариант**

|  |  |
| --- | --- |
| Вариант | Метод |
| 4 | Классификация многоклассовая |

**Датасет**

База данных оценки автомобилей

1. buying - цена покупки
2. maint - цена обслуживания
3. doors - количество дверей
4. persons - вместимость с точки зрения людей, чтобы нести
5. lug\_boot - размер багажника
6. safety - предполагаемая безопасность автомобиля

Источник:

https://archive.ics.uci.edu/ml/datasets/car+evaluation

**Исходный код**

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder

import tensorflow as tf

import matplotlib.pyplot as plt

df = pd.read\_csv("car.data")

X = df.values[:, :-1]

Y = df.values[:, -1]

print(df)

X[:, 0] = LabelEncoder().fit\_transform(X[:, 0])

X[:, 1] = LabelEncoder().fit\_transform(X[:, 1])

X[:, 2] = LabelEncoder().fit\_transform(X[:, 2])

X[:, 3] = LabelEncoder().fit\_transform(X[:, 3])

X[:, 4] = LabelEncoder().fit\_transform(X[:, 4])

X[:, 5] = LabelEncoder().fit\_transform(X[:, 5])

Y = LabelEncoder().fit\_transform(Y)

X = X.astype('float32')

print(X)

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size = 0.33)

Y\_train = tf.keras.utils.to\_categorical(Y\_train)

Y\_test = tf.keras.utils.to\_categorical(Y\_test)

model = tf.keras.Sequential([

    tf.keras.layers.Dense(256, activation='relu'),

    tf.keras.layers.Dense(4, activation='softmax')

])

model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

epochs = 30

history = model.fit(X\_train, Y\_train, epochs = epochs, batch\_size = 5)

loss, accuracy = model.evaluate(X\_test, Y\_test, verbose=0)

print("Точность на тестовой выборке: {:.2f}%".format(accuracy \* 100))

plt.plot(

    np.arange(1, epochs + 1),

    history.history['accuracy'], label='Точность'

)

plt.plot(

    np.arange(1, epochs + 1),

    history.history['loss'], label='Функция потерь'

)

plt.xlabel('Эпохи', size=14)

plt.legend()

plt.show()

**Результат работы**

buying maint doors persons lug\_boot safety class

0 vhigh vhigh 2 2 small low unacc

1 vhigh vhigh 2 2 small med unacc

2 vhigh vhigh 2 2 small high unacc

3 vhigh vhigh 2 2 med low unacc

4 vhigh vhigh 2 2 med med unacc

... ... ... ... ... ... ... ...

1723 low low 5more more med med good

1724 low low 5more more med high vgood

1725 low low 5more more big low unacc

1726 low low 5more more big med good

1727 low low 5more more big high vgood

[1728 rows x 7 columns]

[[3. 3. 0. 0. 2. 1.]

[3. 3. 0. 0. 2. 2.]

[3. 3. 0. 0. 2. 0.]

...

[1. 1. 3. 2. 0. 1.]

[1. 1. 3. 2. 0. 2.]

[1. 1. 3. 2. 0. 0.]]

Epoch 1/30

232/232 [==============================] - 8s 17ms/step - loss: 0.7908 - accuracy: 0.6863

Epoch 2/30

232/232 [==============================] - 5s 19ms/step - loss: 0.6696 - accuracy: 0.7035

Epoch 3/30

232/232 [==============================] - 4s 17ms/step - loss: 0.5945 - accuracy: 0.7502

Epoch 4/30

232/232 [==============================] - 4s 19ms/step - loss: 0.5266 - accuracy: 0.7632

Epoch 5/30

232/232 [==============================] - 4s 16ms/step - loss: 0.4684 - accuracy: 0.7969

Epoch 6/30

232/232 [==============================] - 4s 16ms/step - loss: 0.4271 - accuracy: 0.8194

Epoch 7/30

232/232 [==============================] - 3s 13ms/step - loss: 0.3861 - accuracy: 0.8349

Epoch 8/30

232/232 [==============================] - 3s 14ms/step - loss: 0.3551 - accuracy: 0.8548

Epoch 9/30

232/232 [==============================] - 4s 18ms/step - loss: 0.3285 - accuracy: 0.8608

Epoch 10/30

232/232 [==============================] - 5s 23ms/step - loss: 0.3078 - accuracy: 0.8764

Epoch 11/30

232/232 [==============================] - 5s 20ms/step - loss: 0.2807 - accuracy: 0.8920

Epoch 12/30

232/232 [==============================] - 4s 17ms/step - loss: 0.2693 - accuracy: 0.8946

Epoch 13/30

232/232 [==============================] - 3s 15ms/step - loss: 0.2491 - accuracy: 0.9110

Epoch 14/30

232/232 [==============================] - 3s 14ms/step - loss: 0.2414 - accuracy: 0.9153

Epoch 15/30

232/232 [==============================] - 3s 15ms/step - loss: 0.2280 - accuracy: 0.9196

Epoch 16/30

232/232 [==============================] - 4s 15ms/step - loss: 0.2158 - accuracy: 0.9274

Epoch 17/30

232/232 [==============================] - 5s 20ms/step - loss: 0.2050 - accuracy: 0.9343

Epoch 18/30

232/232 [==============================] - 3s 14ms/step - loss: 0.1959 - accuracy: 0.9360

Epoch 19/30

232/232 [==============================] - 5s 20ms/step - loss: 0.1888 - accuracy: 0.9300

Epoch 20/30

232/232 [==============================] - 5s 20ms/step - loss: 0.1776 - accuracy: 0.9430

Epoch 21/30

232/232 [==============================] - 7s 29ms/step - loss: 0.1713 - accuracy: 0.9499

Epoch 22/30

232/232 [==============================] - 4s 17ms/step - loss: 0.1614 - accuracy: 0.9559

Epoch 23/30

232/232 [==============================] - 4s 15ms/step - loss: 0.1525 - accuracy: 0.9559

Epoch 24/30

232/232 [==============================] - 4s 16ms/step - loss: 0.1529 - accuracy: 0.9568

Epoch 25/30

232/232 [==============================] - 3s 15ms/step - loss: 0.1444 - accuracy: 0.9680

Epoch 26/30

232/232 [==============================] - 4s 15ms/step - loss: 0.1365 - accuracy: 0.9611

Epoch 27/30

232/232 [==============================] - 3s 15ms/step - loss: 0.1322 - accuracy: 0.9654

Epoch 28/30

232/232 [==============================] - 3s 15ms/step - loss: 0.1254 - accuracy: 0.9706

Epoch 29/30

232/232 [==============================] - 4s 18ms/step - loss: 0.1214 - accuracy: 0.9706

Epoch 30/30

232/232 [==============================] - 3s 15ms/step - loss: 0.1181 - accuracy: 0.9654

Точность на тестовой выборке: 95.27%

