МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Искусственные нейронные сети»

Тема: Многоклассовая классификация цветов

Студент гр. 8383 _____ Мирсков А. А.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург

2021

Цель работы.

Реализовать классификацию сортов растения ирис (Iris Setosa - 0, Iris Versicolour - 1, Iris Virginica - 2) по четырем признакам: размерам пестиков и тычинок его цветков.

Задание

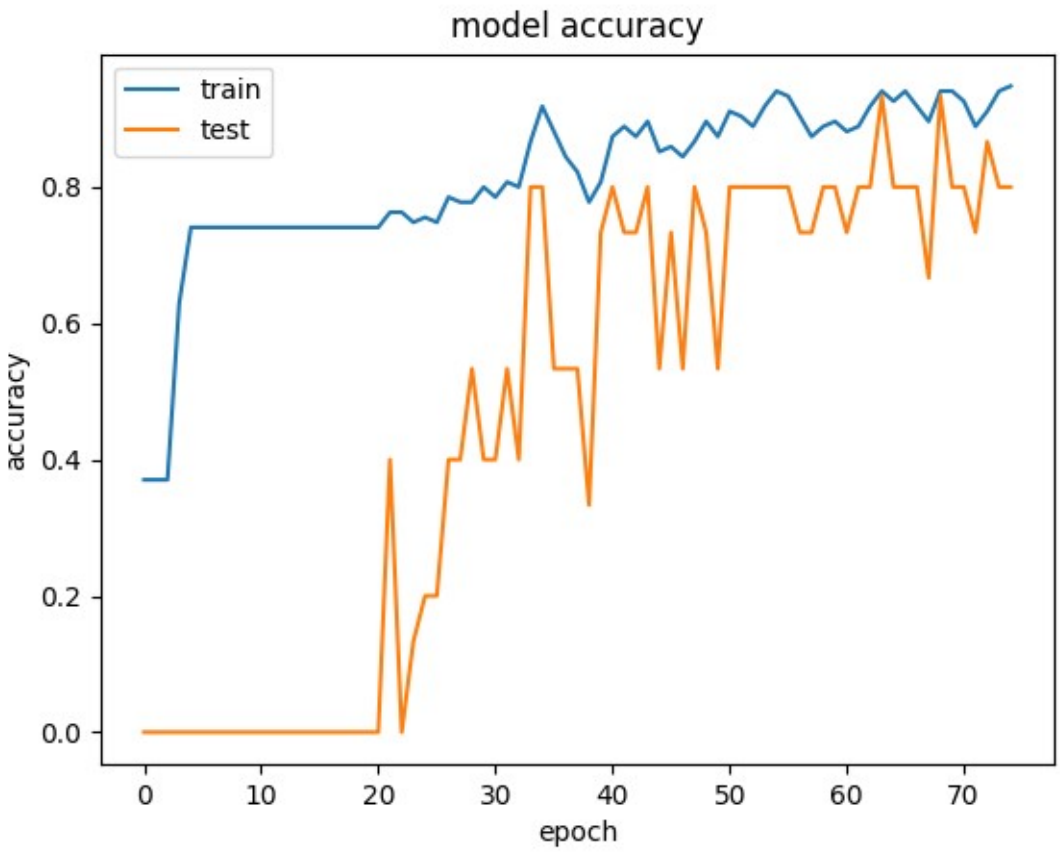
- Ознакомиться с задачей классификации
- Загрузить данные
- Создать модель ИНС в Keras
- Настроить параметры обучения
- Обучить и оценить модель

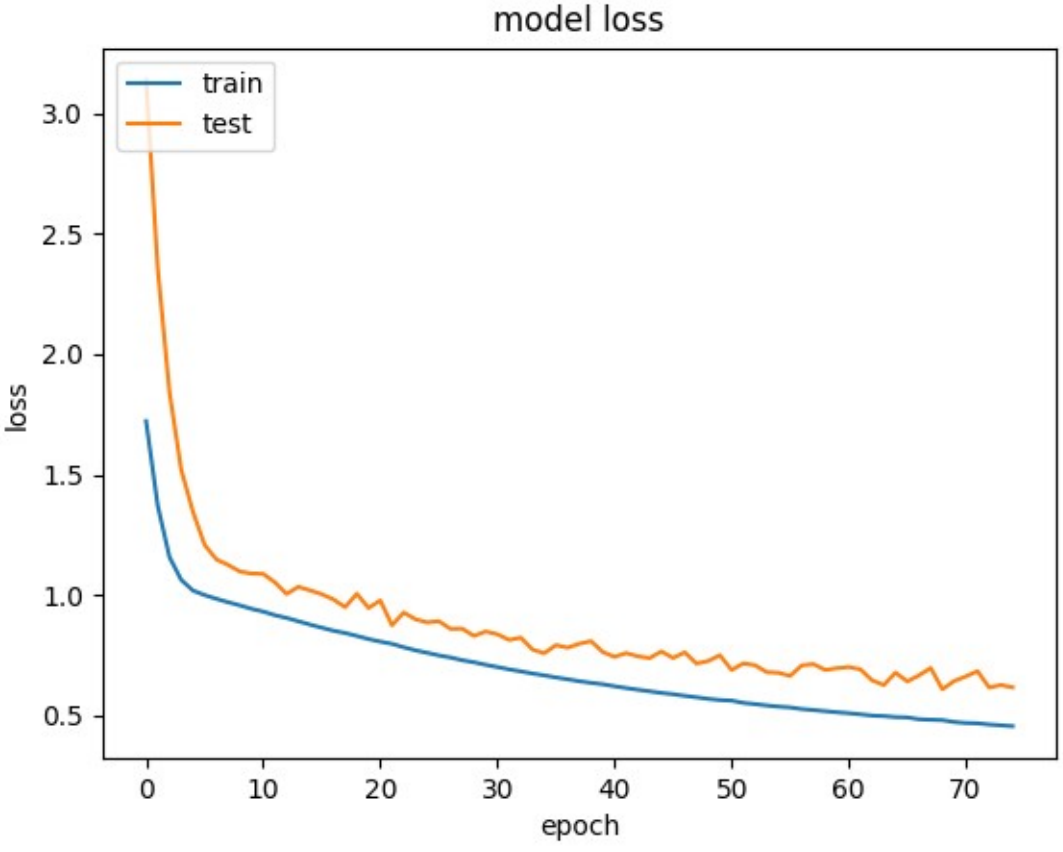
Требования

1. Изучить различные архитектуры ИНС (Разное кол-во слоев, разное кол-во нейронов на слоях)
2. Изучить обучение при различных параметрах обучения (параметры ф-ций fit)
3. Построить графики ошибок и точности в ходе обучения
4. Выбрать наилучшую модель

Выполнение работы.

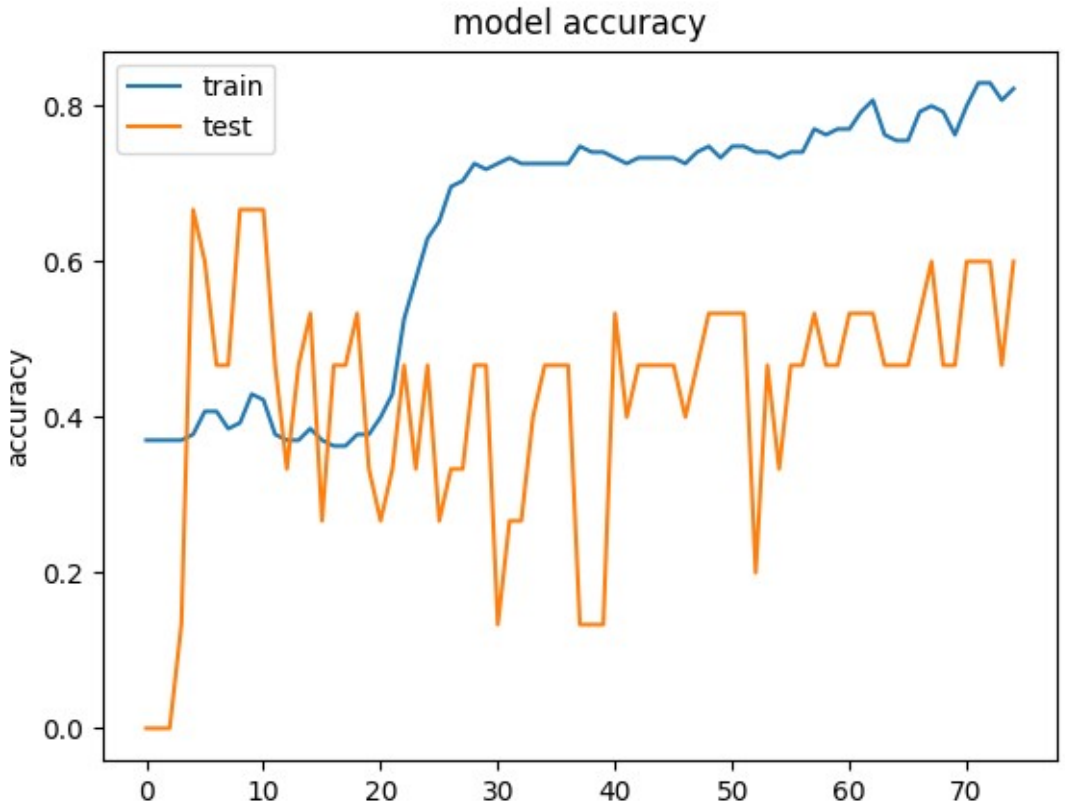
Был скачан набор данных и написана программа обучения нейронной сети и построения графиков ошибки и точности. Код программы представлен в приложении А. Графики, полученные в результате работы программы представлены ниже.

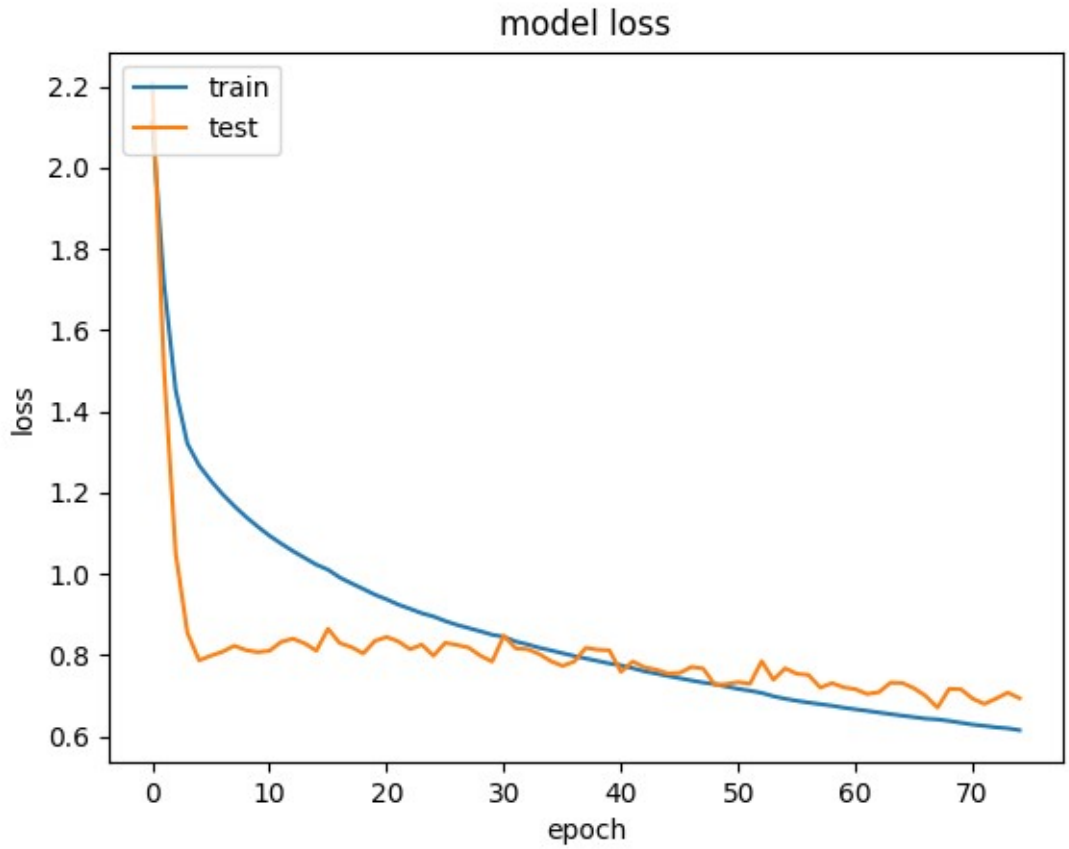

model accuracy
train
test
accuracy
epoch
0.0
0.2
0.4
0.6
0.8
0
10
20
30
40
50
60
70


model loss
train
test
loss
epoch
0.5
1.0
1.5
2.0
2.5
3.0
0
10
20
30
40
50
60
70

Изучение различных архитектур сети.

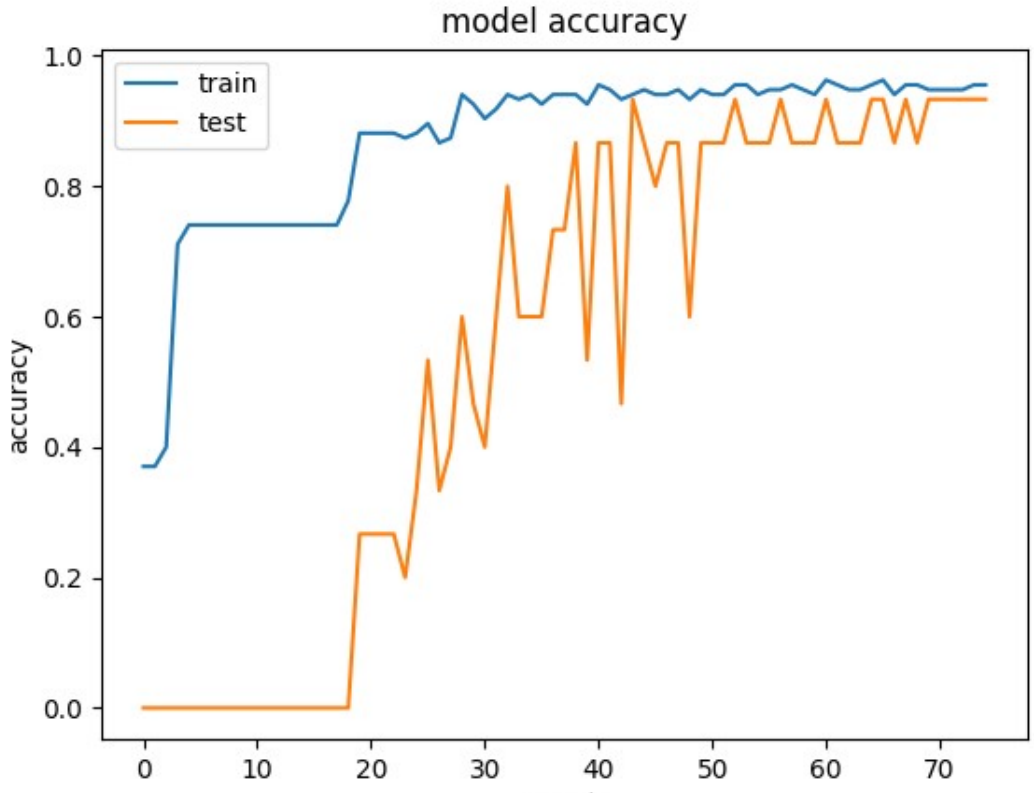
1. Был добавлен дополнительный слой с 4 нейронами. Графики, полученные в результате работы программы представлены ниже.

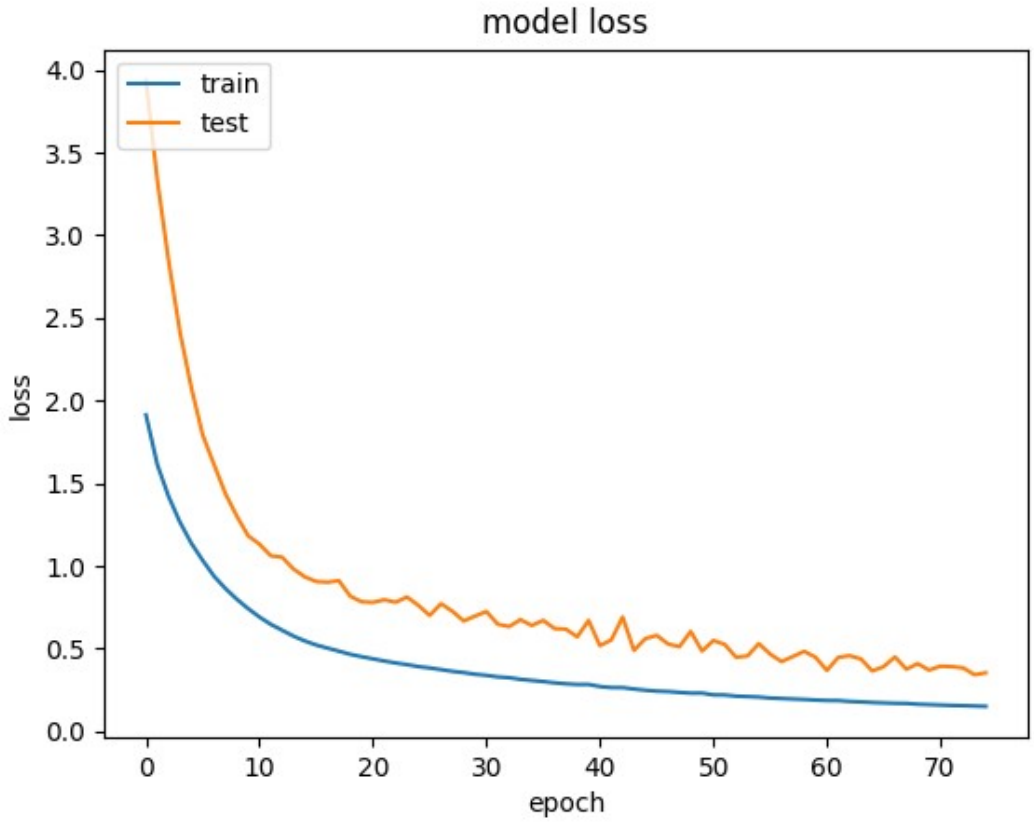




Точность уменьшилась по сравнению с первоначальной моделью, будем увеличивать кол-во нейронов.

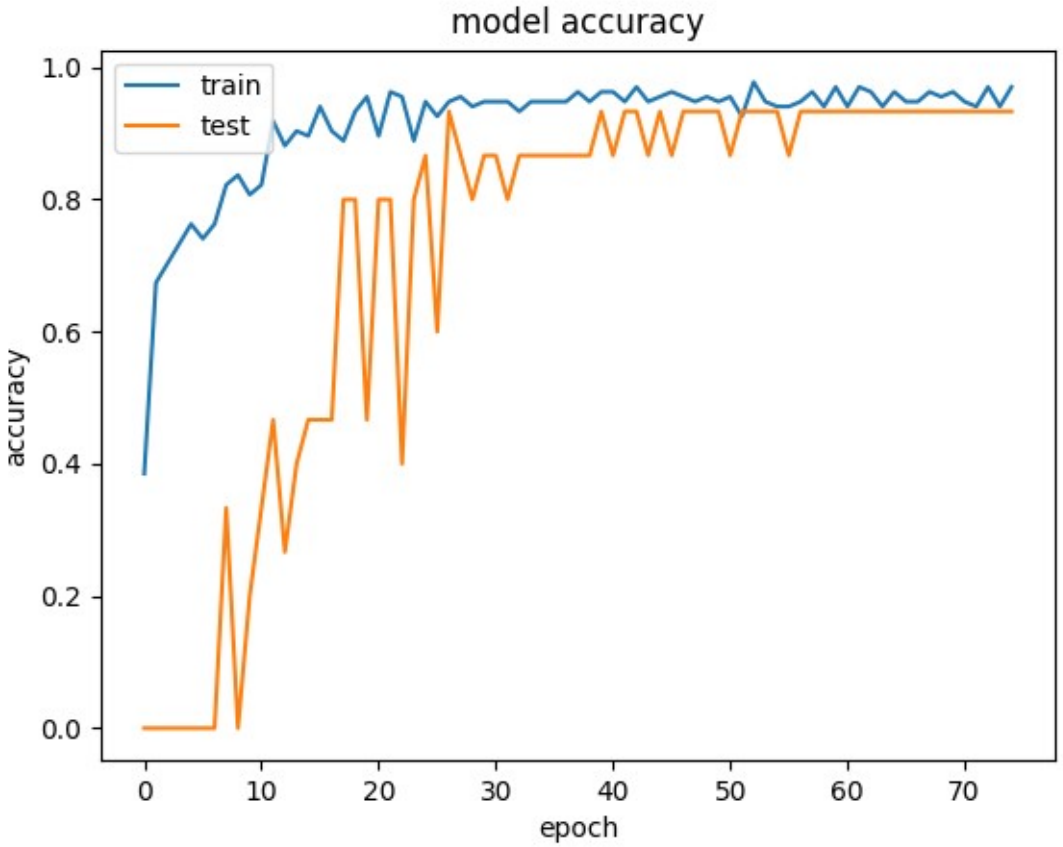
2. Количество нейронов на скрытом слое было увеличено до 16. Графики, полученные в результате работы программы представлены ниже.

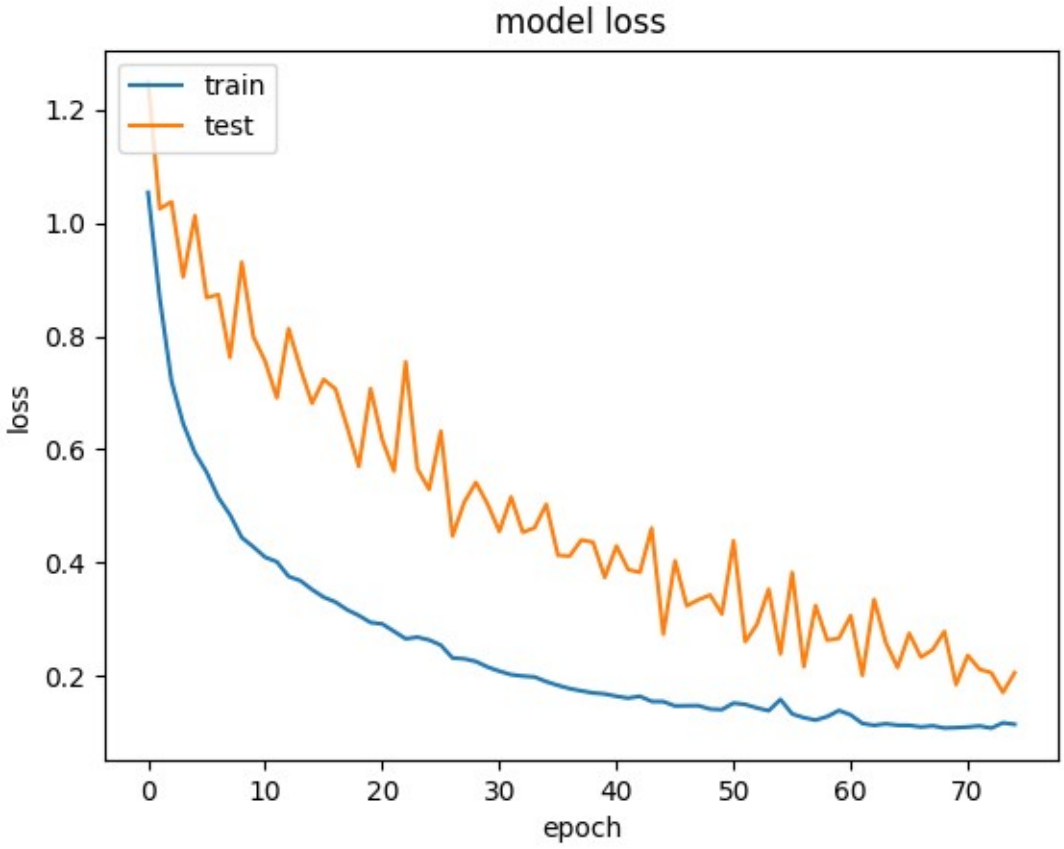




При увеличении кол-ва нейронов точность заметно возросла.

3. Увеличим кол-во нейронов на первом слое. Графики, полученные в результате работы программы представлены ниже.

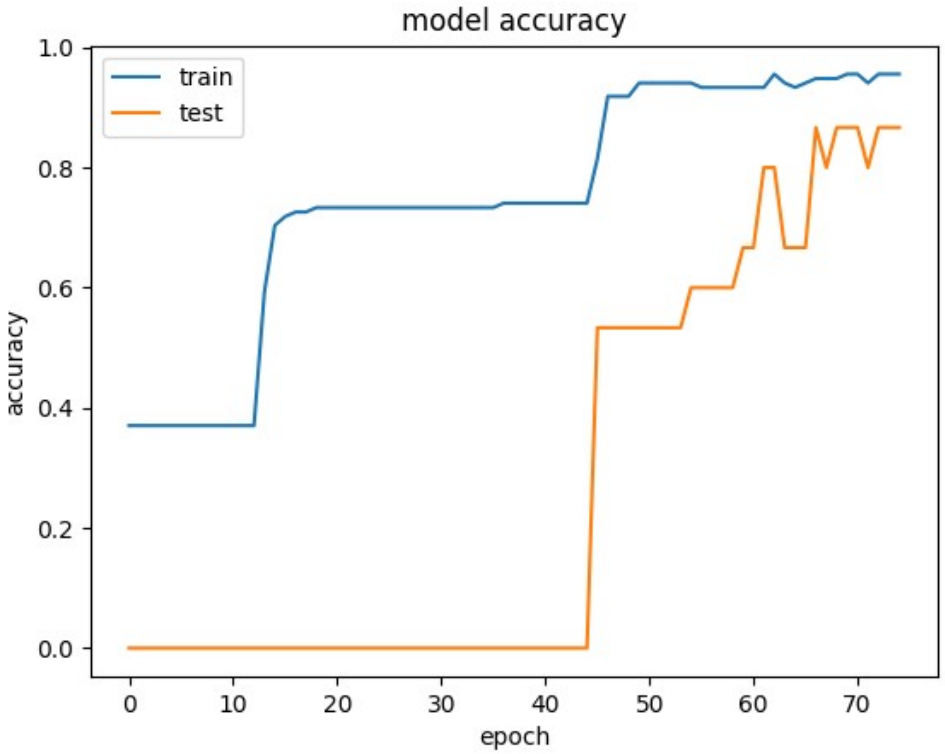


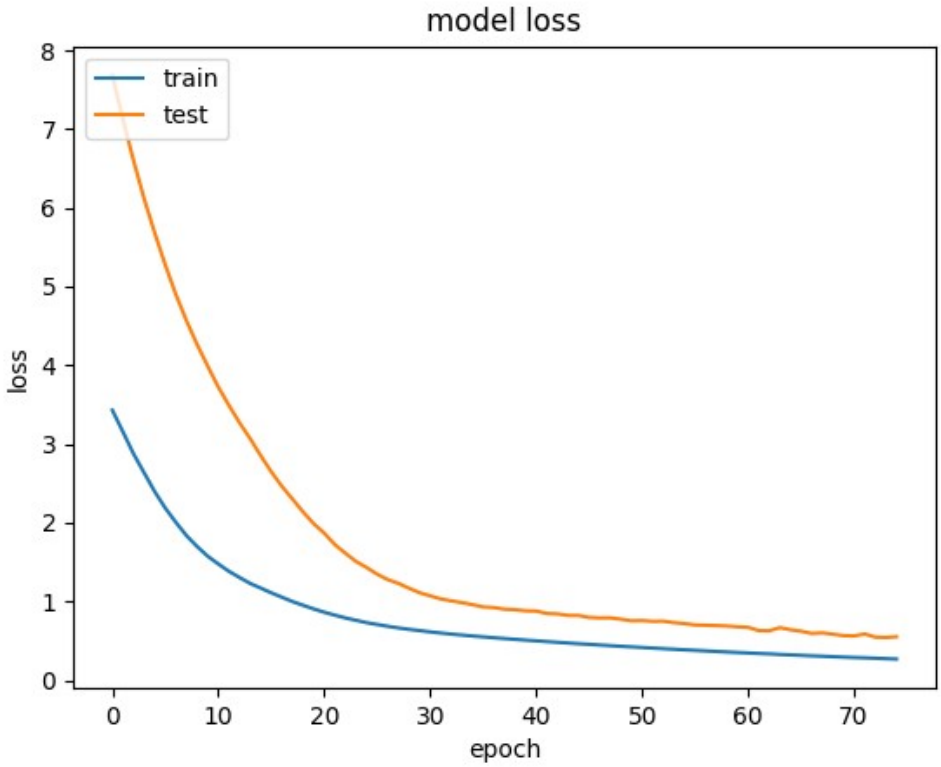


По полученным результатам можно сказать, что точность практически не изменилась, значит для дальнейших исследований можно выбрать предыдущую модель.

Изучение обучения при различных параметрах обучения:

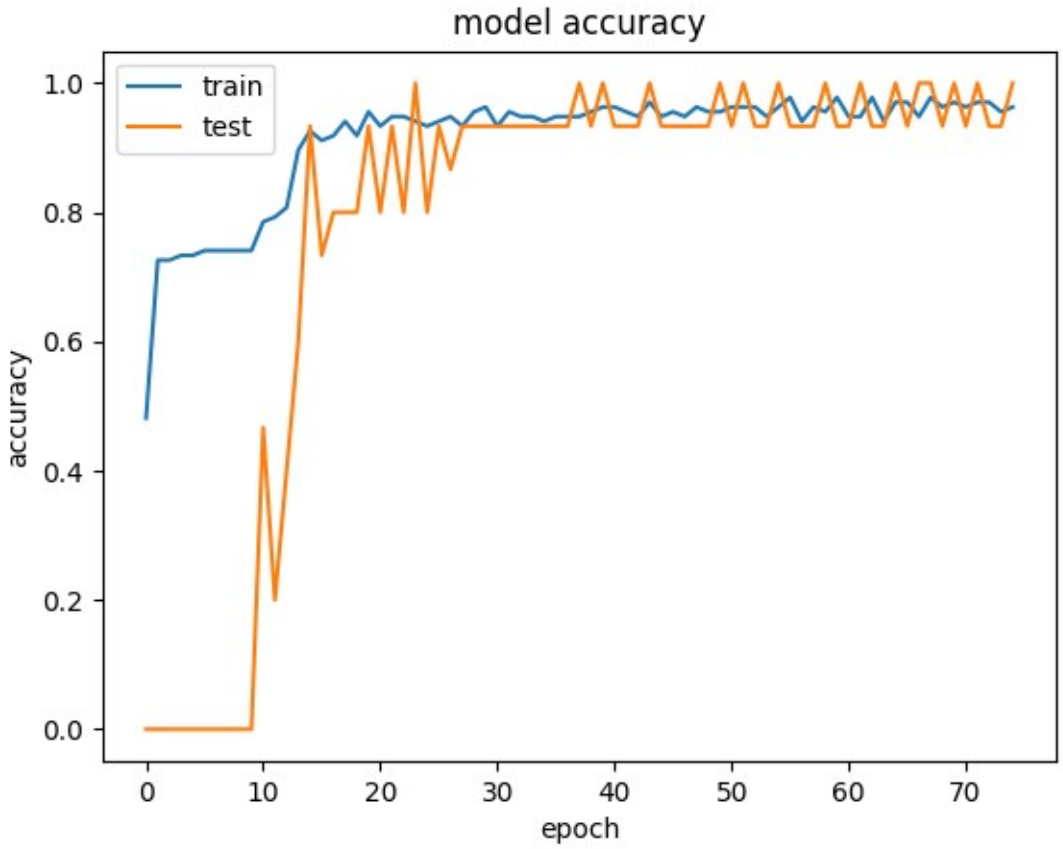
1. Увеличим batch_size до 20. Графики, полученные в результате работы программы представлены ниже.

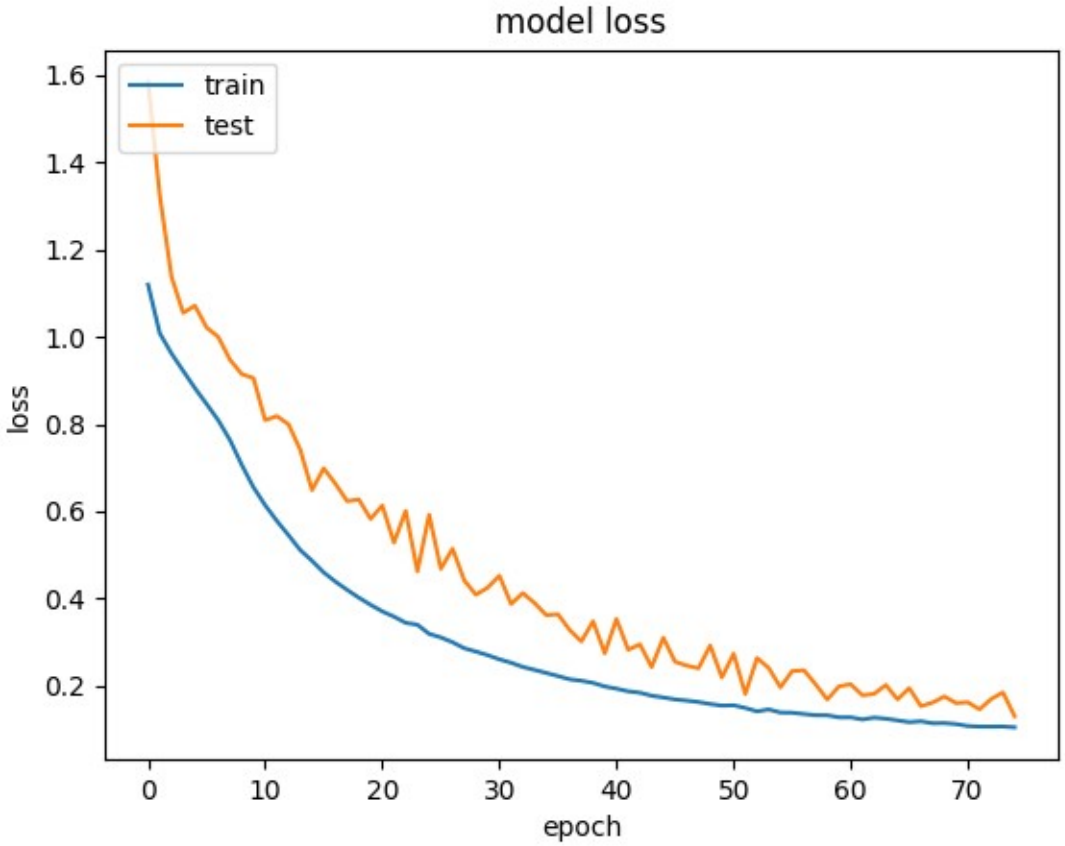




Результаты ухудшились. Попробуем уменьшить batch_size.

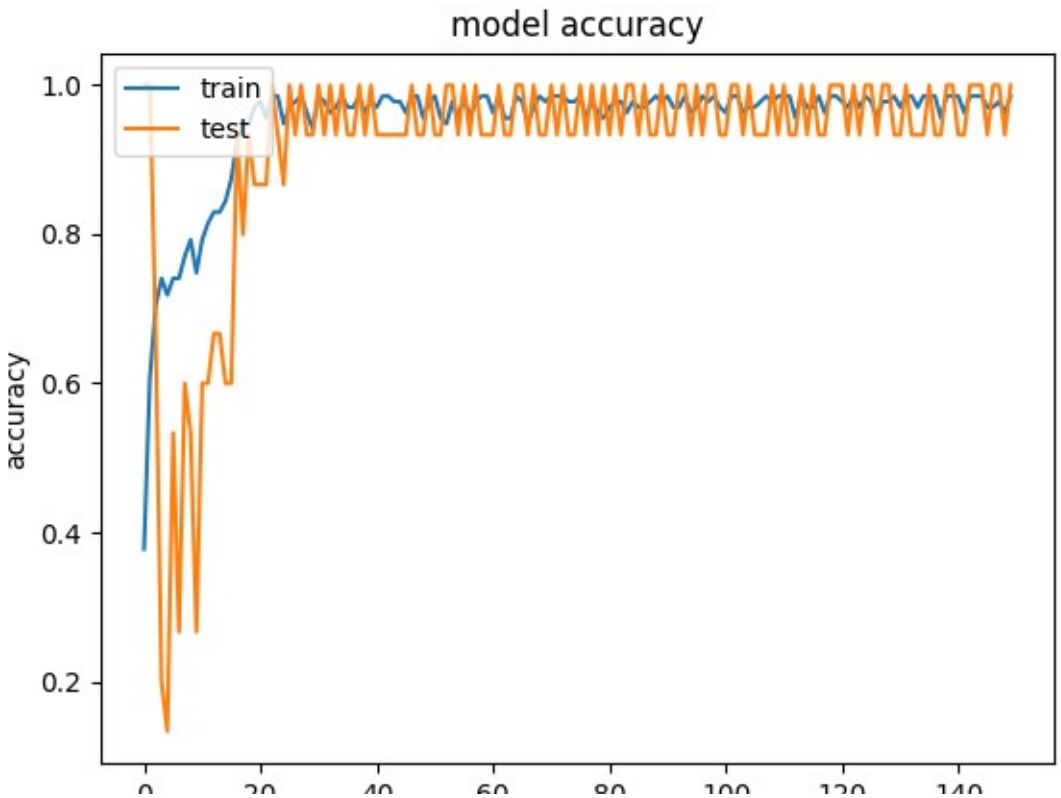
2. Уменьшим batch_size до 8. Графики, полученные в результате работы программы представлены ниже.

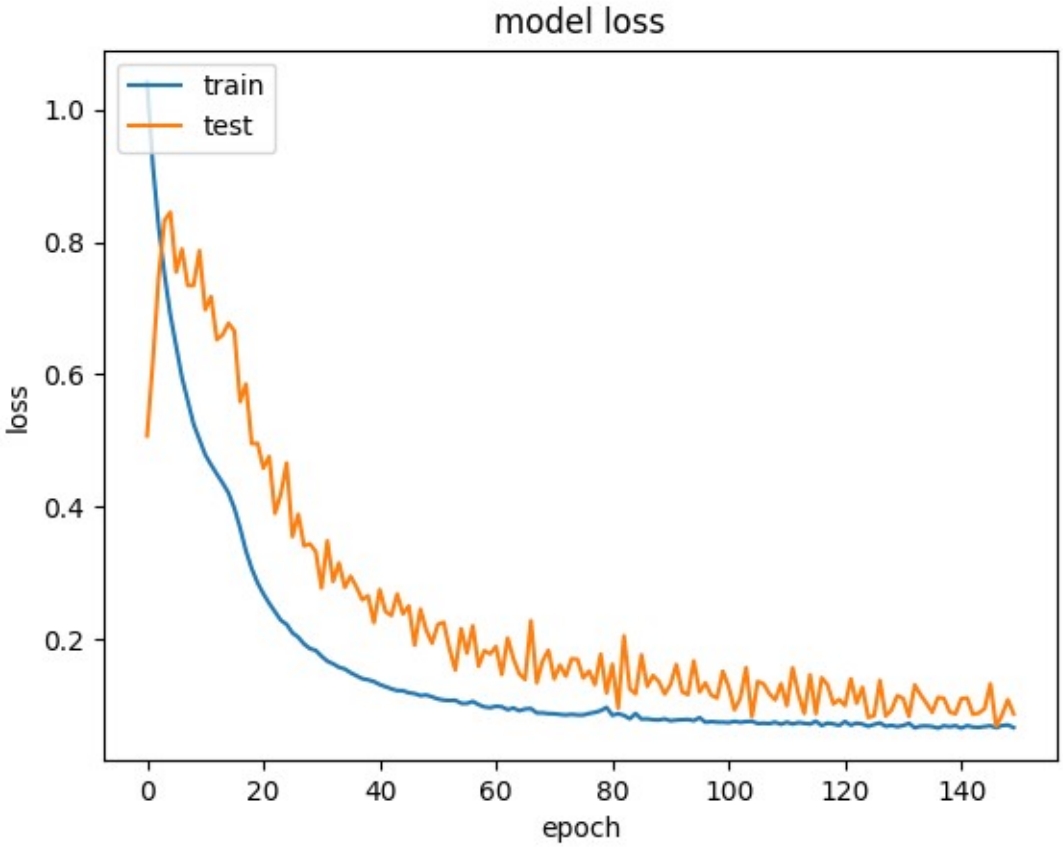




Программа показывает результаты немного лучше, чем при batch_size 10.

3. Увеличим кол-во эпох до 150. Графики, полученные в результате работы программы представлены ниже.





Последняя модель показала наилучший результат с точностью 0.985 на обучающих данных и точностью 1 на тестовых.

Выводы.

В ходе выполнения лабораторной работы была реализована классификация сортов растений по четырём признакам. Были изучены различные архитектуры ИНС и различные параметры обучения.

Приложение А

Код программы

```
import pandas
from tensorflow.keras.layers import Dense
from tensorflow.keras.models import Sequential
from tensorflow.keras.utils import to_categorical
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt

# Loading data
dataframe = pandas.read_csv("iris.csv", header=None)
dataset = dataframe.values
X = dataset[:,0:4].astype(float)
Y = dataset[:,4]

# Text labels to categorical vector
encoder = LabelEncoder()
encoder.fit(Y)
encoded_Y = encoder.transform(Y)
dummy_y = to_categorical(encoded_Y)

# Creating model
model = Sequential()
model.add(Dense(4, activation='relu'))
model.add(Dense(3, activation='softmax'))

# Initializing training parameters
model.compile(optimizer='adam',loss='categorical_crossentropy',
metrics=['accuracy'])

# Training net
```

```
history = model.fit(X, dummy_y, epochs=75, batch_size=10,
validation_split=0.1)


# Plotting accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

# Plotting loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```