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

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Искусственные нейронные сети»

Тема: «Многоклассовая классификация цветов»

Студентка гр. 7381 _____ Давкаева В.С.

Преподаватель _____ Жукова Н.А.

Санкт-Петербург
2020

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.

Порядок выполнения работы.

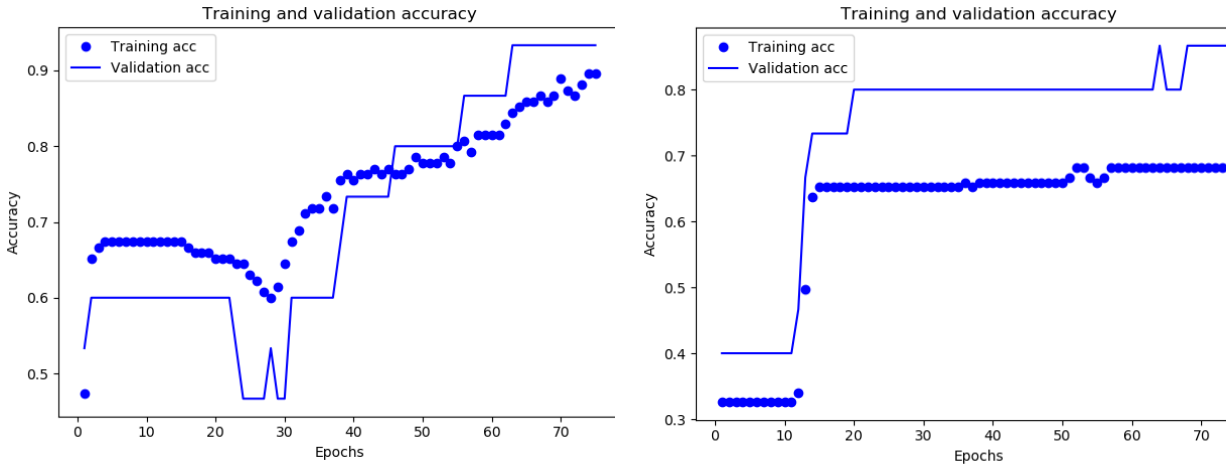
1. Ознакомиться с задачей классификации;
2. Загрузить данные;
3. Создать модель ИНС в Keras;
4. Настроить параметры обучения;
5. Обучить и оценить модель;

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;

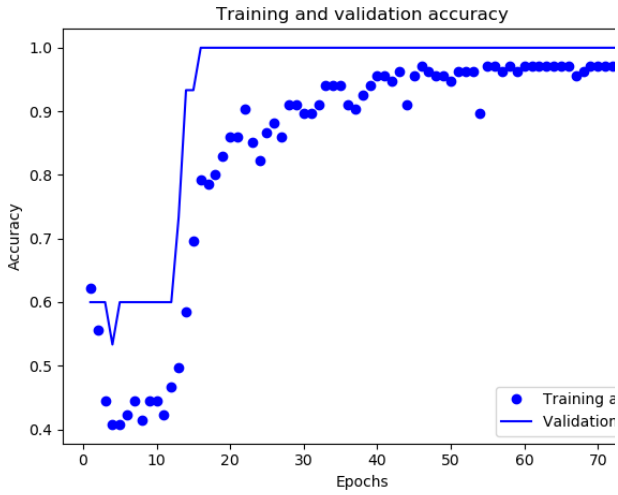
Ход работы.

Попробуем на единственном нейронном слое поменять количество нейронов. Полученные графики точности и потерь представлены на рис. 1, 2 соответственно.

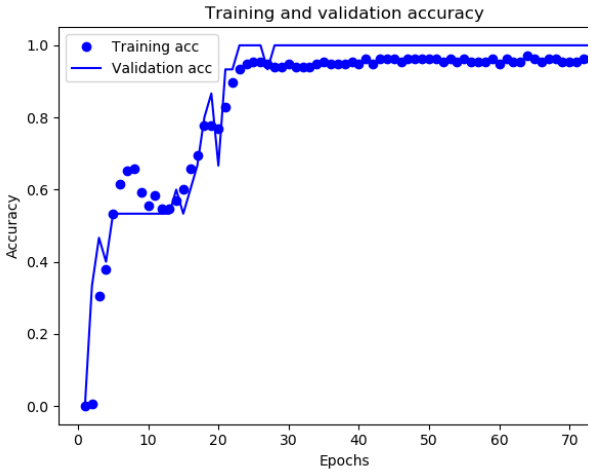


а

б

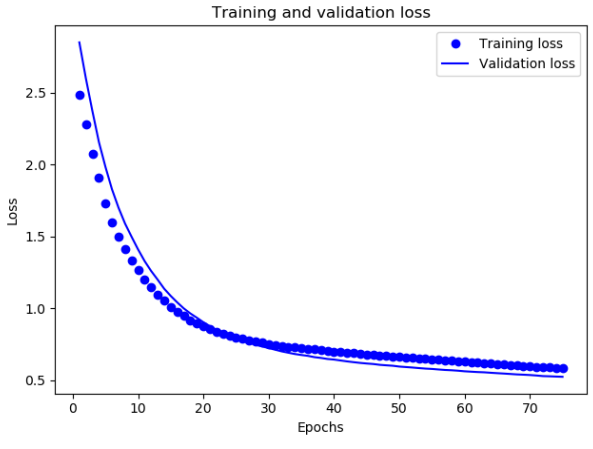


в

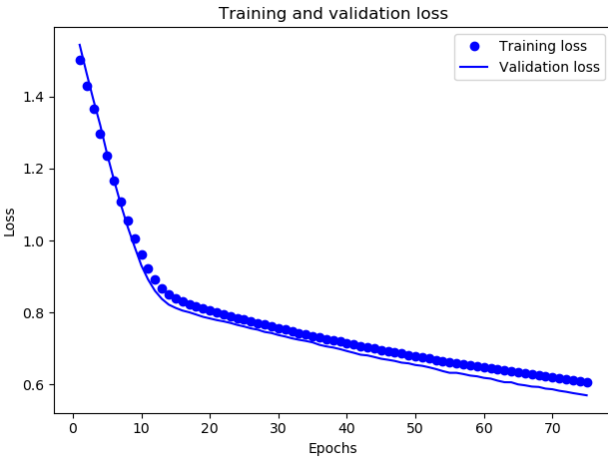


г

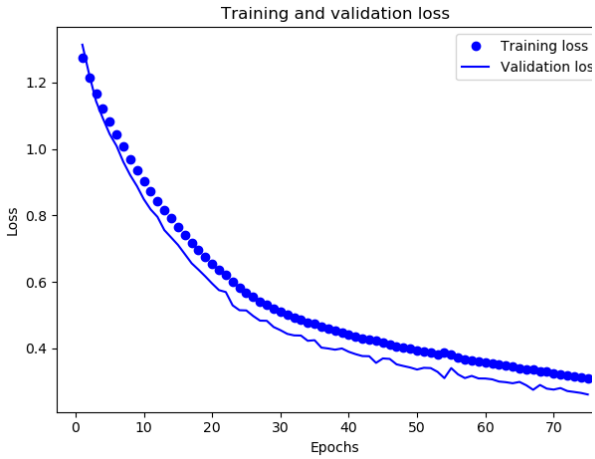
Рисунок 1 – Точность модели при 2 (а), 4(б), 8 (в) и 16 (г) нейронах на единственном нейронном слое.



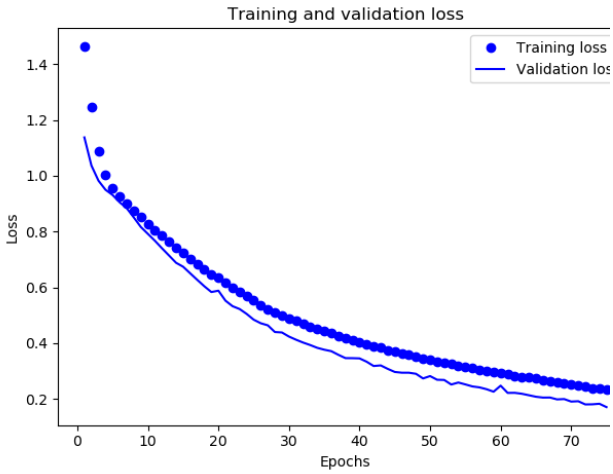
а



б



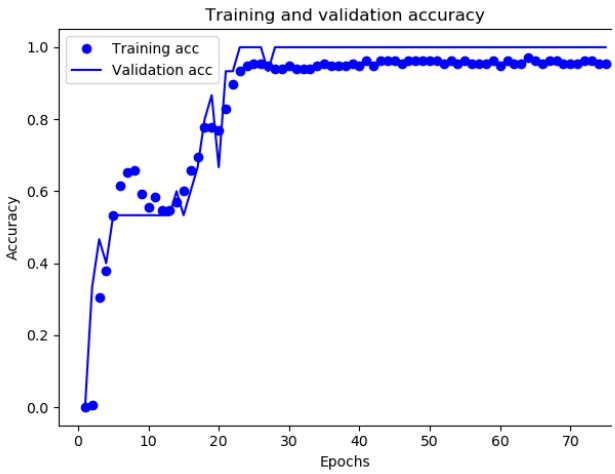
в



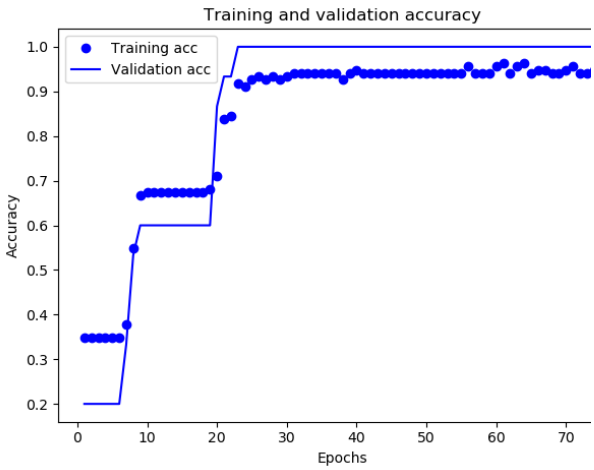
г

Рисунок 2 – График потерь при 2 (а), 4(б), 8 (в) и 16 (г) нейронах на единственном нейронном слое.

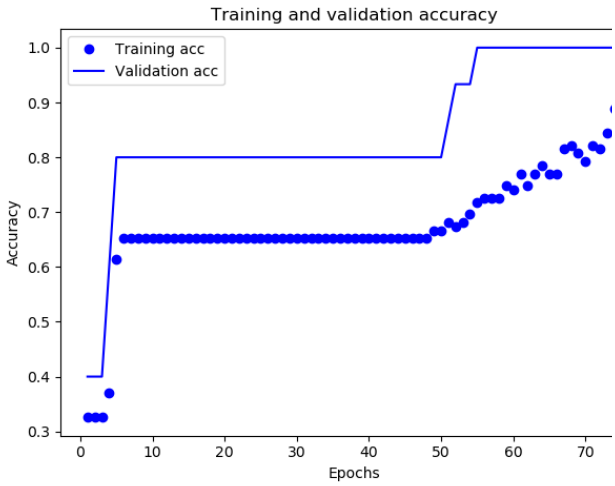
Как видим, при увеличении числа нейронов на единственном слое повышается точность модели раньше происходит переобучение нейронной сети. Теперь при фиксированном числе нейронов (4) на каждом слое построим графики при увеличении числа слоёв. Графики точности модели и потерь в данных условиях представлены на рис. 3, 4.



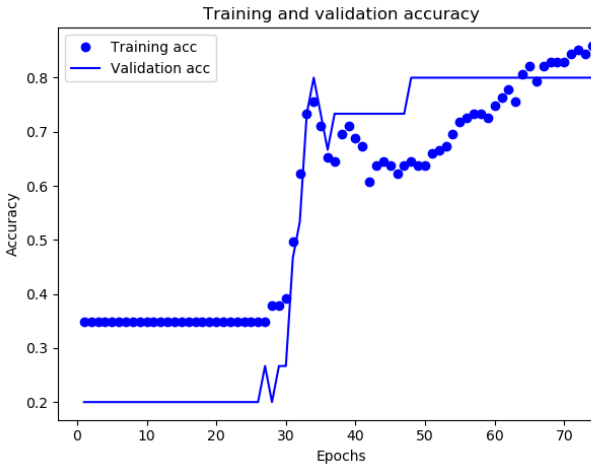
а



б



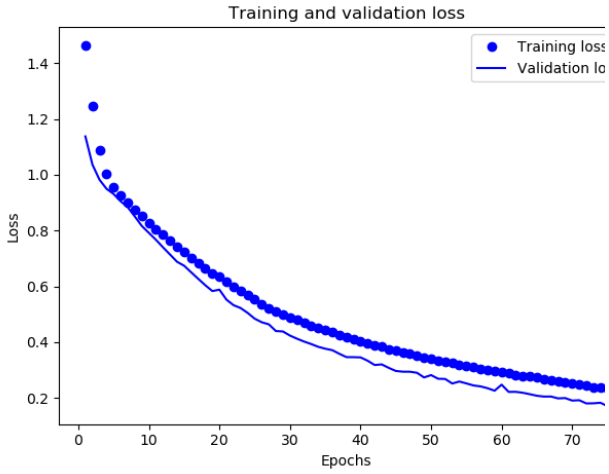
в



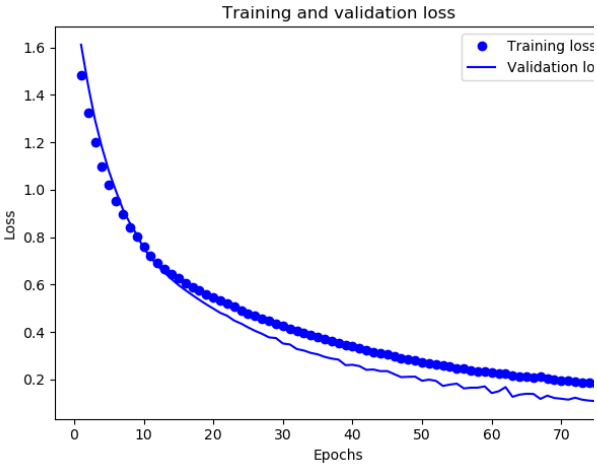
г

Рисунок 3- Точность модели при 4 нейронах на 1 (а), 2 (б), 3(в) и 4(г) нейронных слоях.

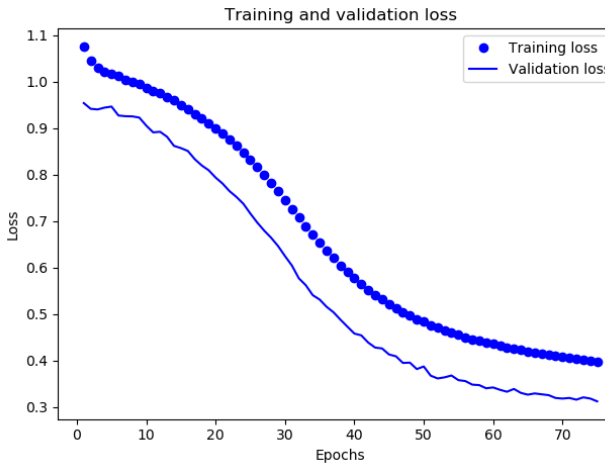
Как видно, количество слоёв положительно сказывается на точности модели и скорости её обучения.



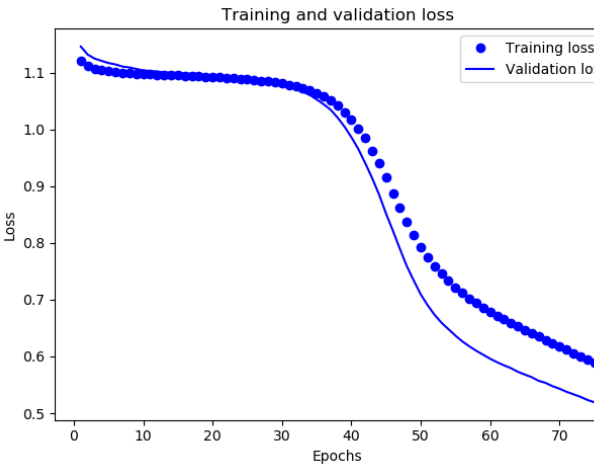
а



б



в

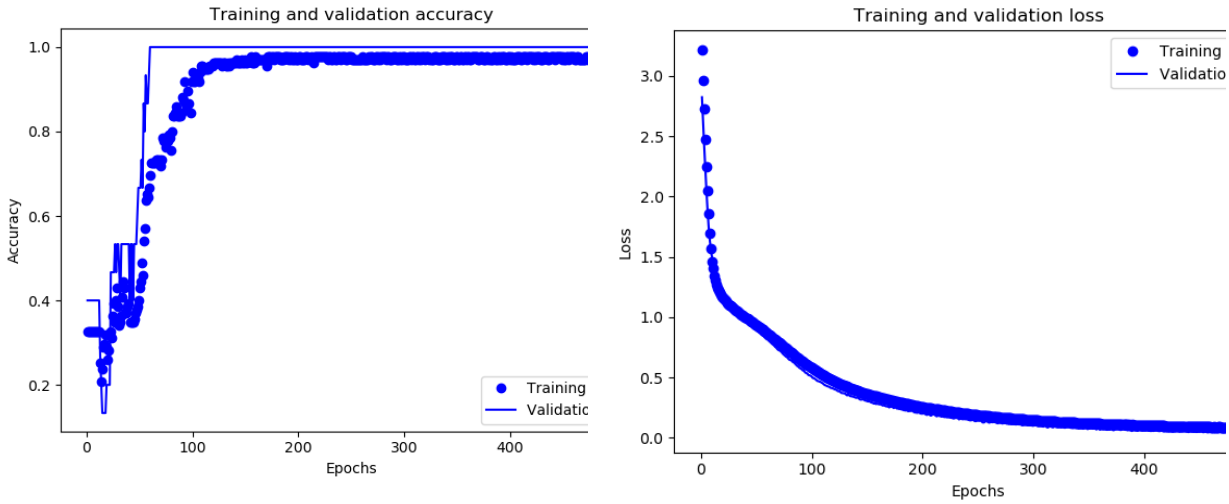


г

Рисунок 4 – График потерь при 4 нейронах на 1 (а), 2 (б), 3(в) и 4 (г) нейронных слоях.

Оптимальной архитектурой для данной модели я выбираю 1 нейронный слоя с 4 нейронами.

Теперь посмотрим, как будет везти себя модель при большом количестве эпох (например, 500). Полученные графики представлены на рис. 3.



а б

Рисунок 5 – Точность и потери модели из 1 нейронного слоя с 4 нейронами(500 эпох).

Как видим по графикам, данная модель достигла максимальной точности примерно за 70 поколений, дальнейшее обучение приведет к переобучению.

Выводы.

В ходе выполнения лабораторной работы были изучены различные архитектуры искусственных нейронных сетей, изучили обучение при различных параметрах обучения, построили графики ошибок и точности в ходе обучения, выбрали наилучшую модель.

Наилучшая модель в рамках данной работы состоит из 1 нейронного слоя с 4 нейронами.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД

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

dataframe = pandas.read_csv("iris.csv", header=None)
dataset = dataframe.values
rand = list(range(len(dataset)))
random.seed(999)
random.shuffle(rand)
dataset = dataset[rand]
X = dataset[:,0:4].astype(float)
Y = dataset[:,4]

encoder = LabelEncoder()
encoder.fit(Y) #выделение классов
encoded_Y = encoder.transform(Y) #трансформация в числовой аналог
dummy_y = to_categorical(encoded_Y) #Преобразует в двоичную матрицу классов


model = Sequential()

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

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

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

history_dict = history.history
#график ошибки
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
#график точности
plt.clf()
acc_values = history_dict['acc']
val_acc_values = history_dict['val_acc']
plt.plot(epochs, acc_values, 'bo', label='Training acc')
plt.plot(epochs, val_acc_values, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```