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 _____ Мололкин К.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.

Задачи

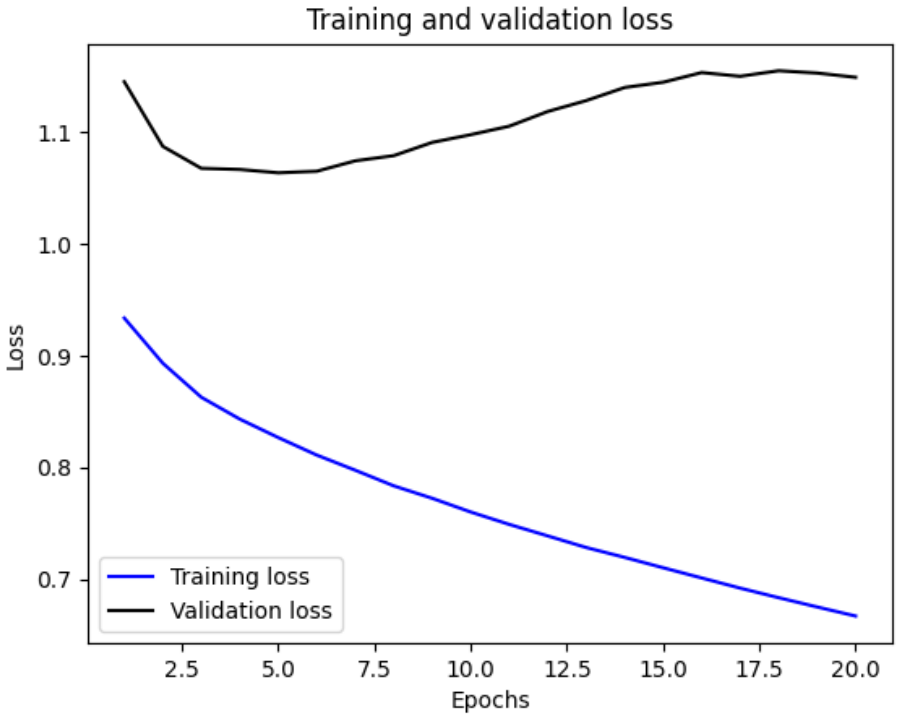
- Ознакомиться с задачей классификации
- Загрузить данные
- Создать модель ИНС в Keras
- Настроить параметры обучения
- Обучить и оценить модель

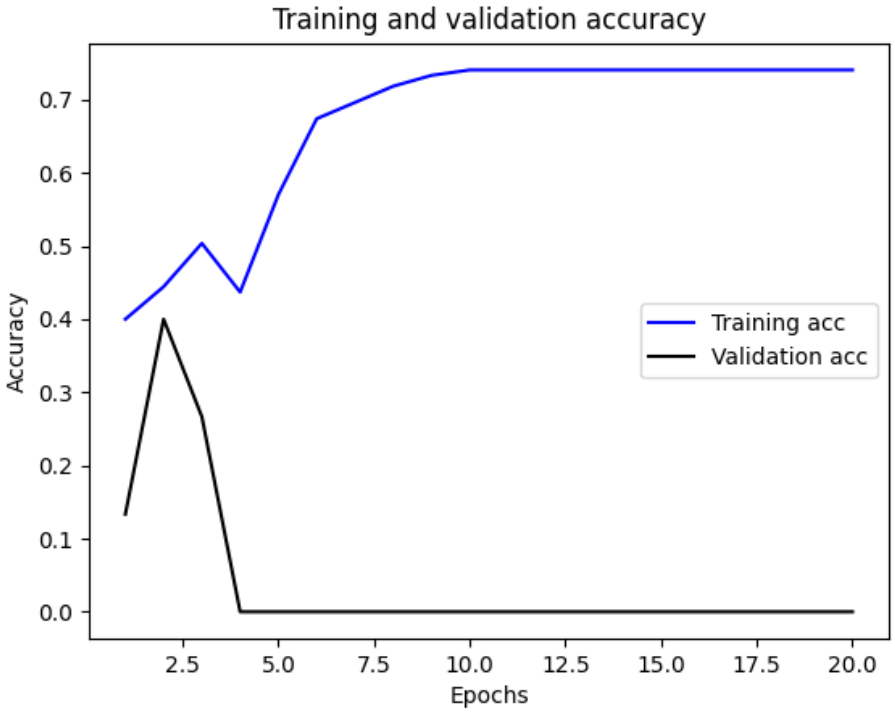
Выполнение работы

Задача многоклассовой классификации является одним из основных видов задач, для решения которых применяются нейронные сети. Для выполнения поставленной задачи был загружен файл с обучающими данными и написана программа на языке python с использованием библиотек keras для обучения ИНС, pandas для загрузки данных, scikit-learn для обучения модели и matplotlib для рисования графиков. Код программы представлен в приложении А. Затем проведен ряд тестов при разных архитектурах ИНС и параметрах обучения.

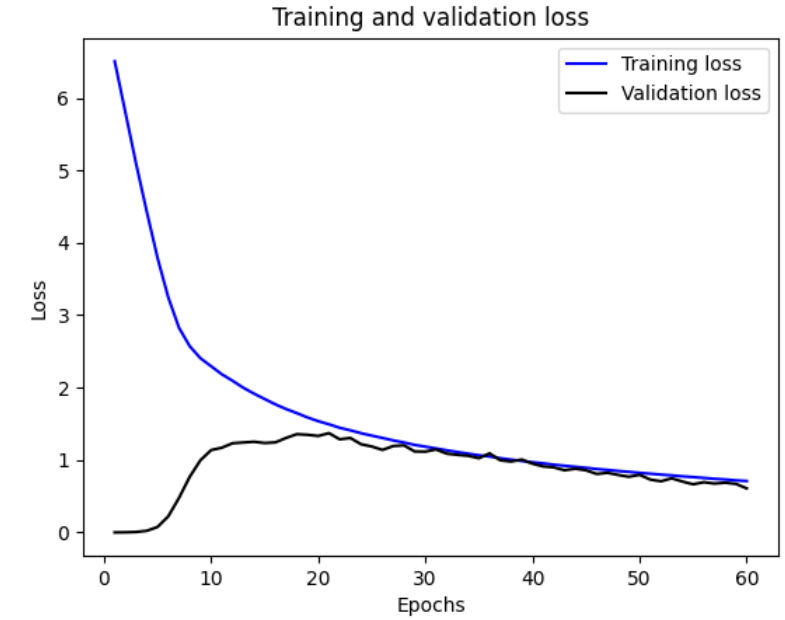
1. Обучение при двух слоях с 4 нейронами на первом и 3 на втором слоях:

a. 20 эпох, 10 выборок, 10% обучающих данных.

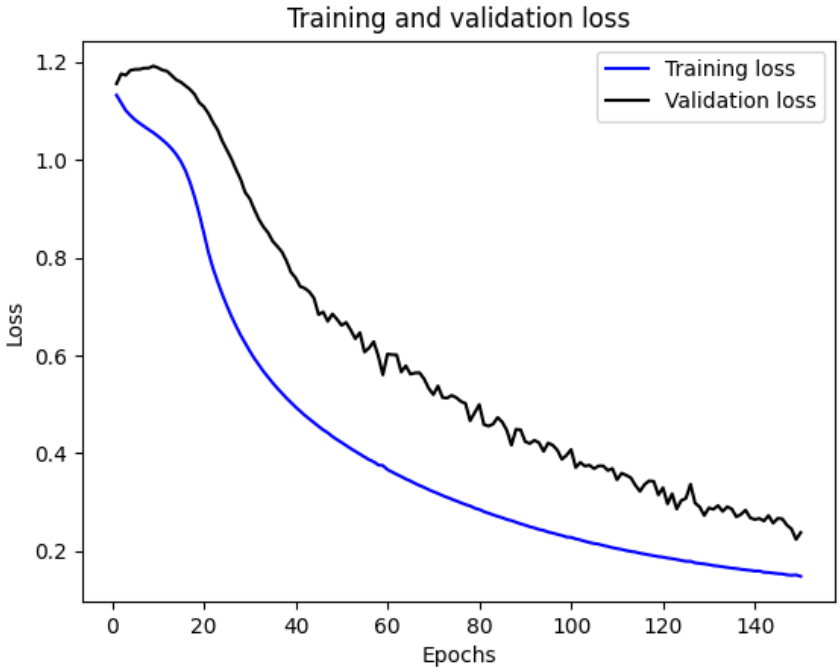


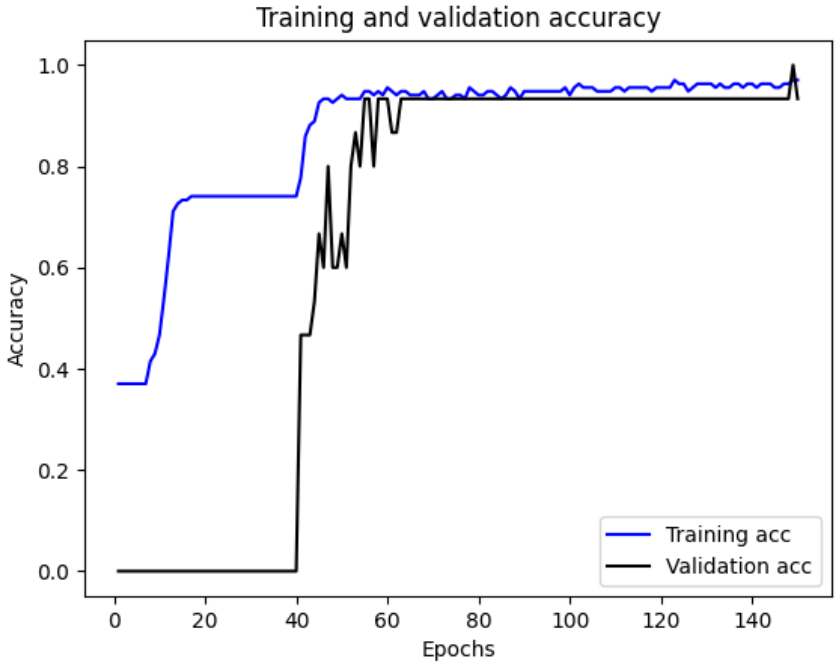


b. 60 эпох, 10 выборок, 10% обучающих данных



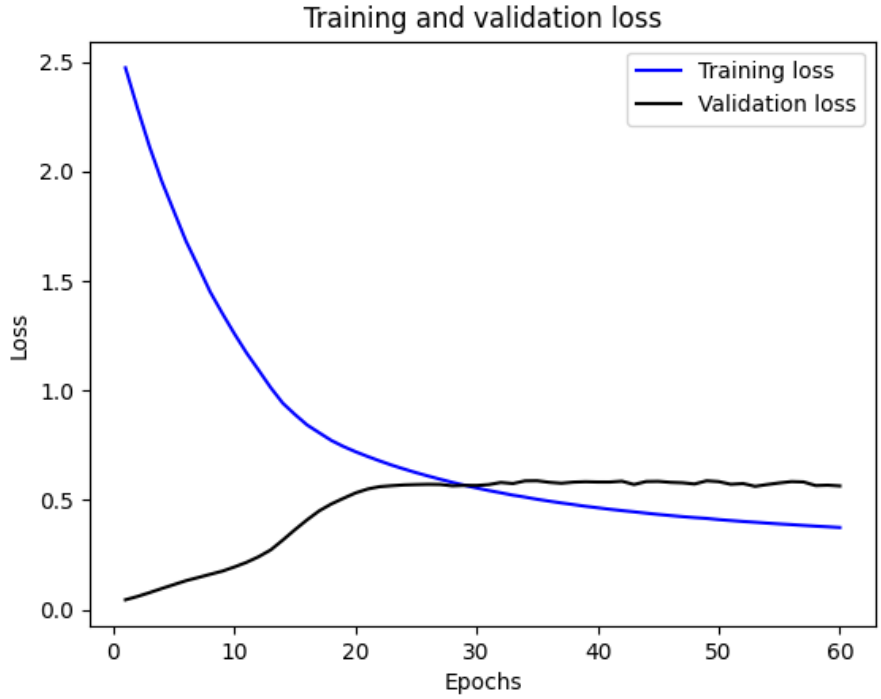
c. 150 эпох, 20 выборок, 10% обучающих данных

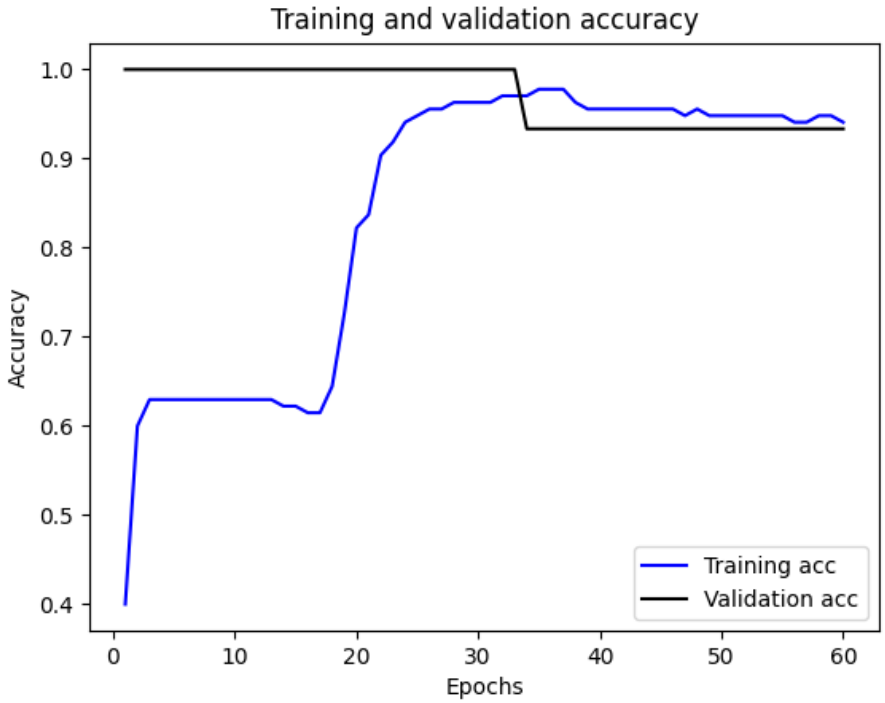




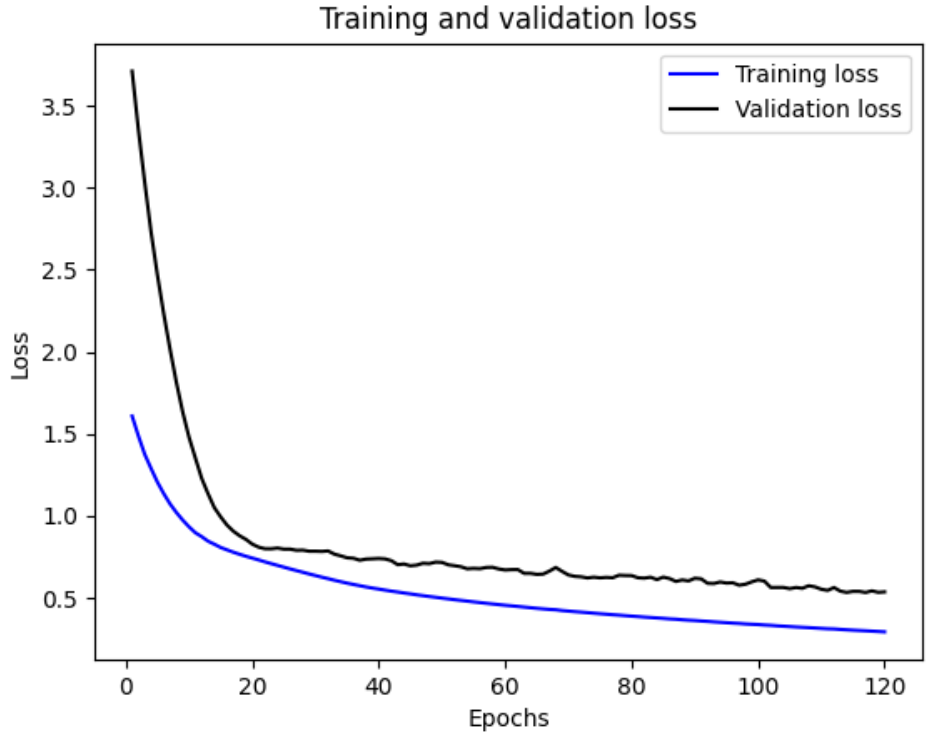
2. Обучение при двух слоях с 10 нейронами на первом и 3 на втором слоях:

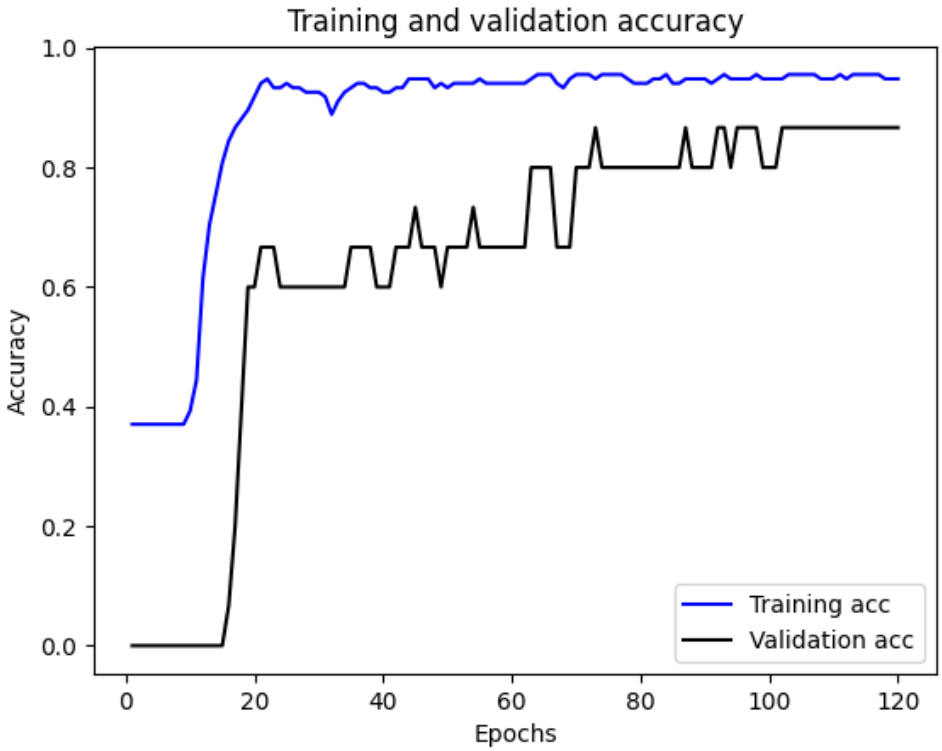
a. 60 эпох, 20 выборок, 10% обучающих данных



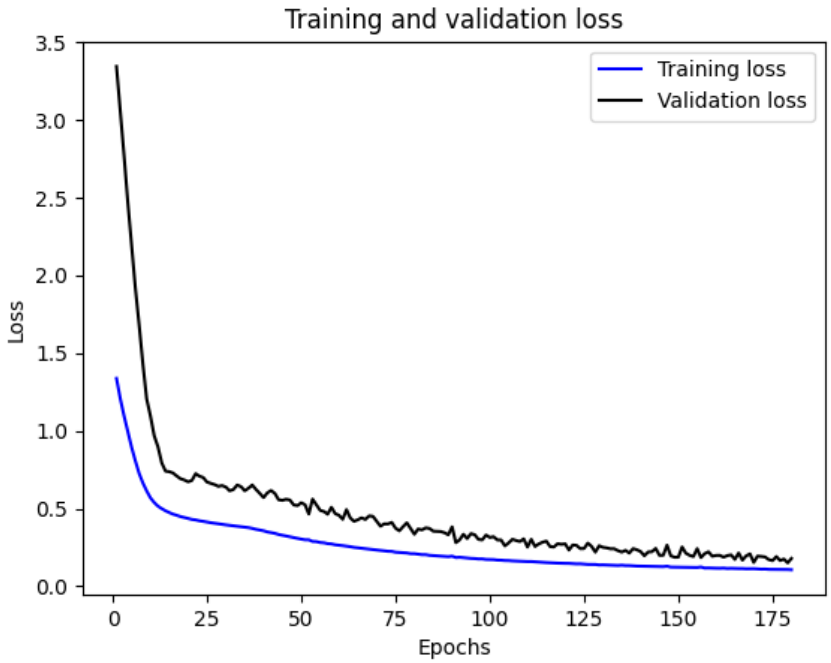


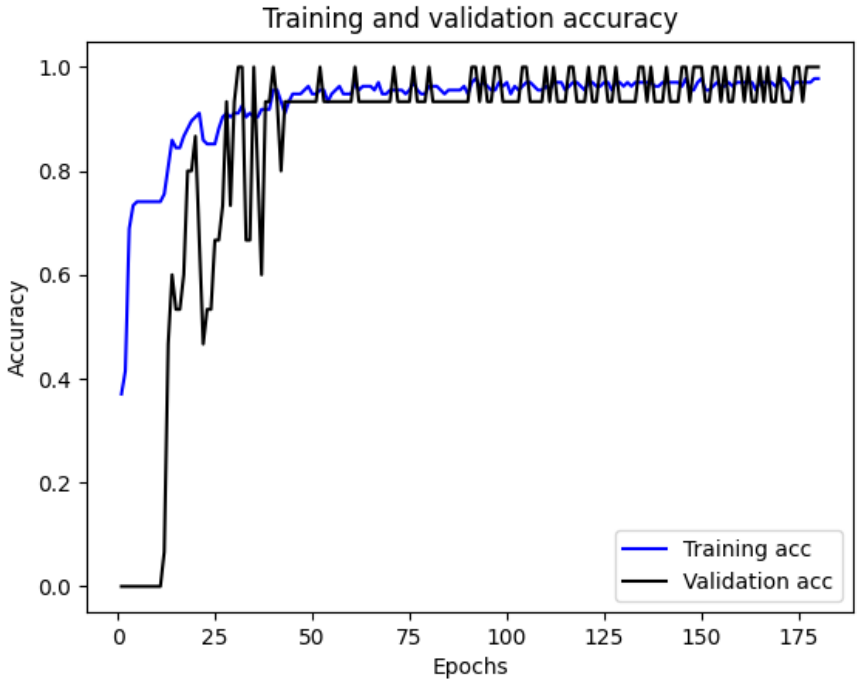
b. 120 эпох, 20 выборок, 10% обучающих данных



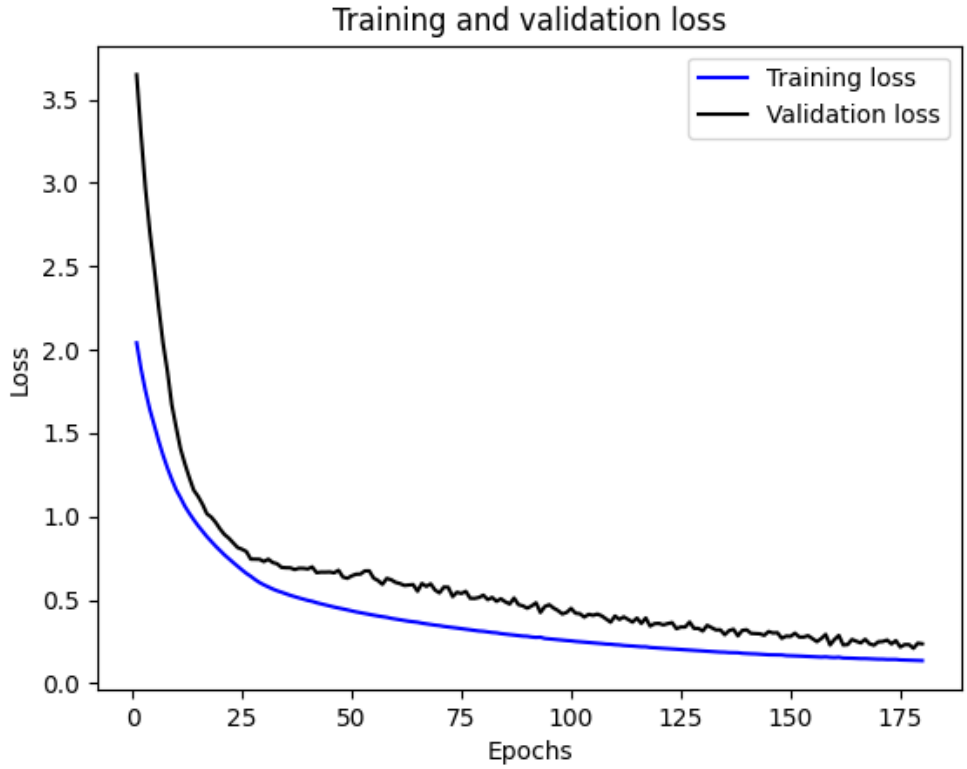


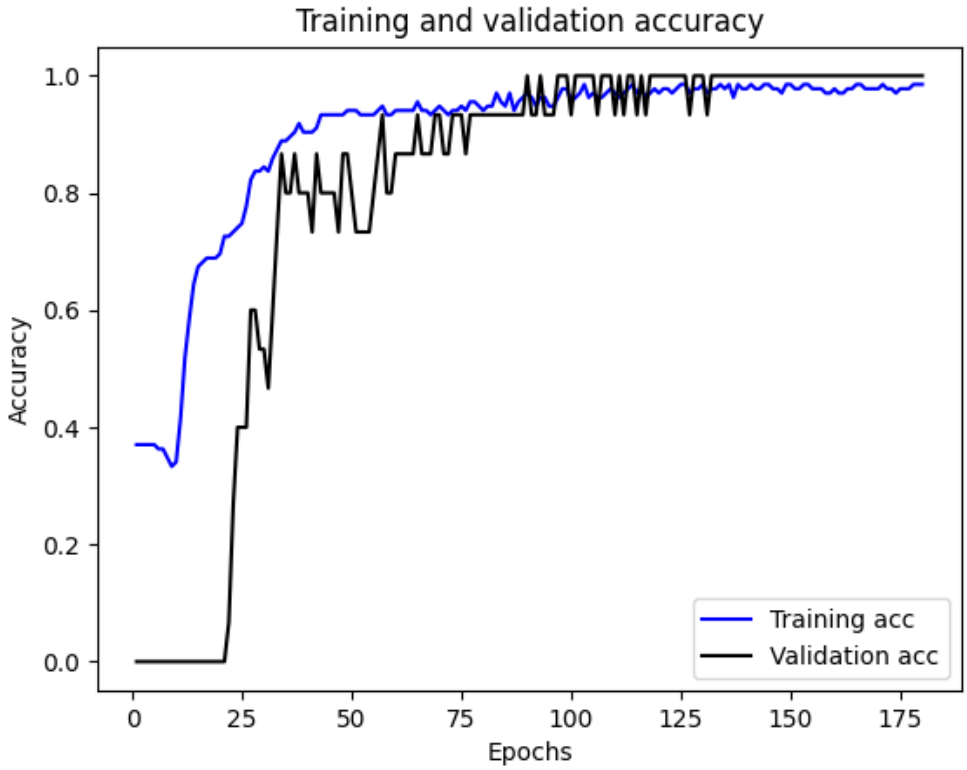
c. 180 эпох, 10 выборок, 10% обучающих данных



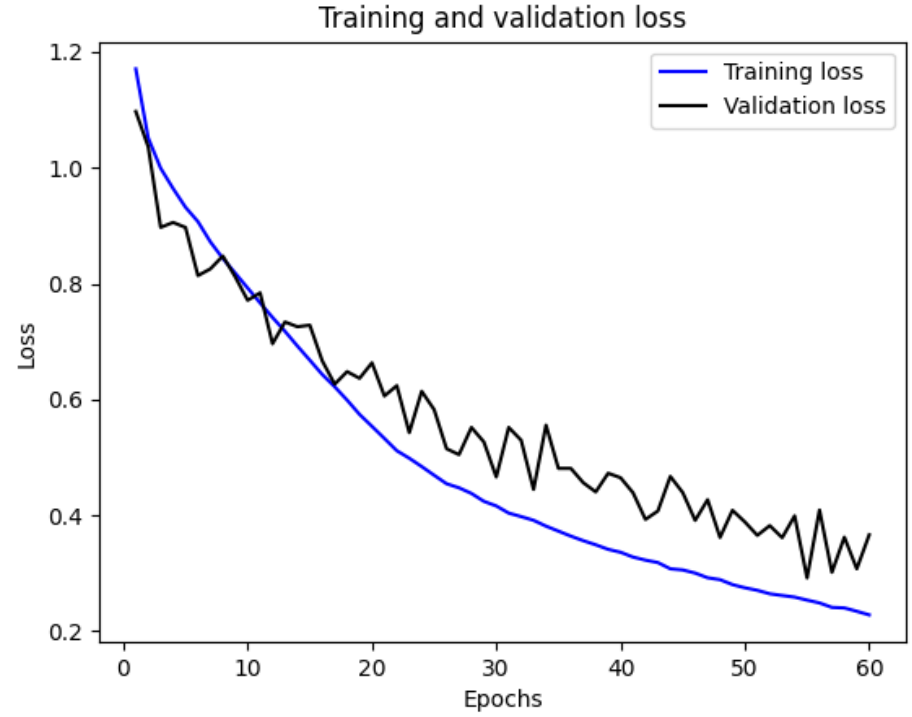


3. Обучение при двух слоях с 9 нейронами на первом и 3 на втором слоях:
   a. 180 эпох, 10 выборок, 10% обучающих данных

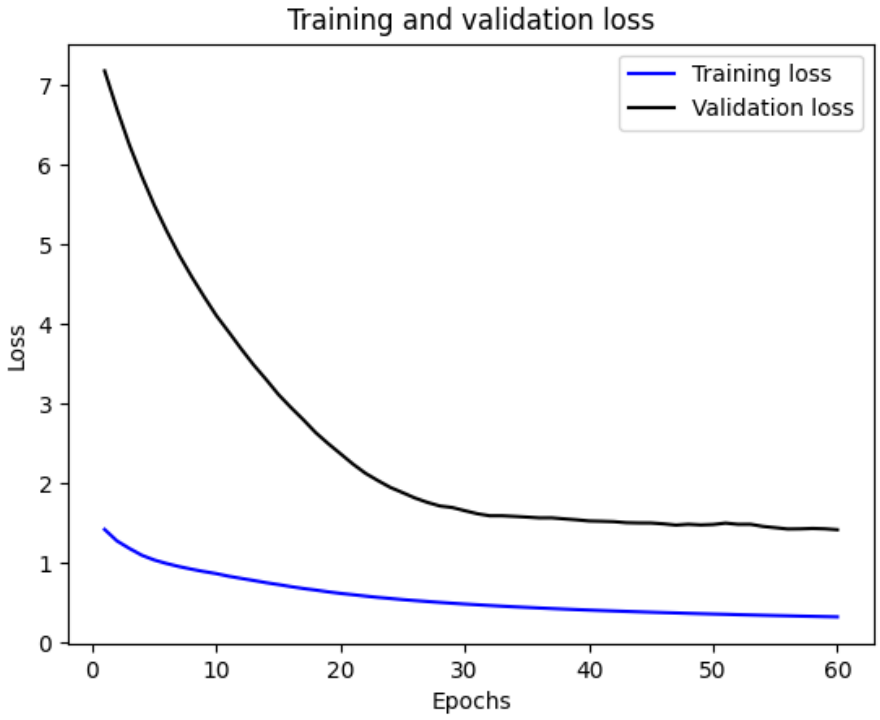


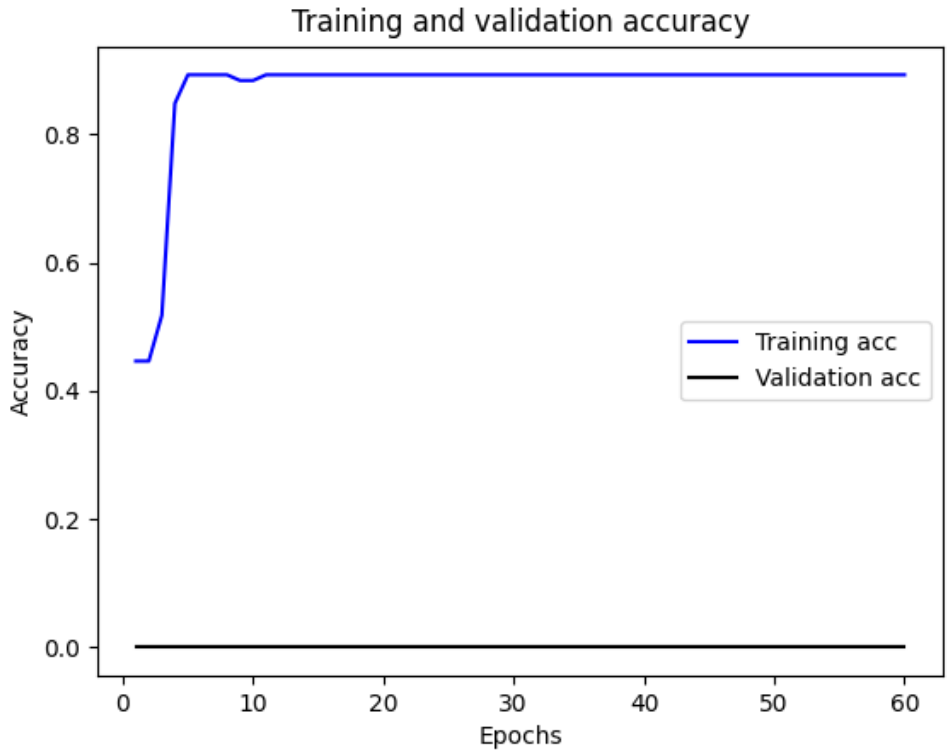


4. Обучение при трех слоях с 12 нейронами на первом, 6 на втором слоях, 3 на последнем:
   a. 60 эпох, 10 выборок, 10% обучающих данных

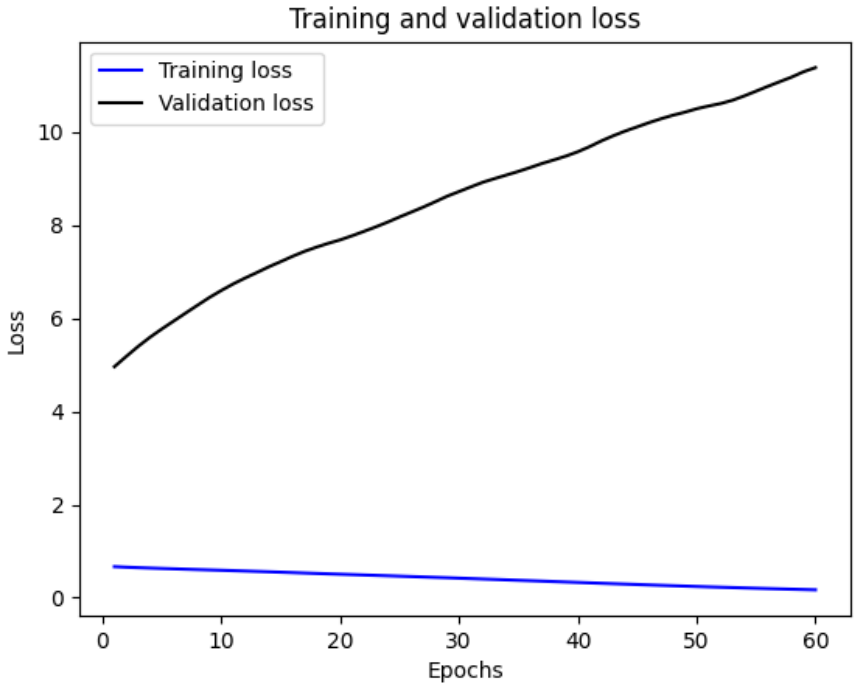


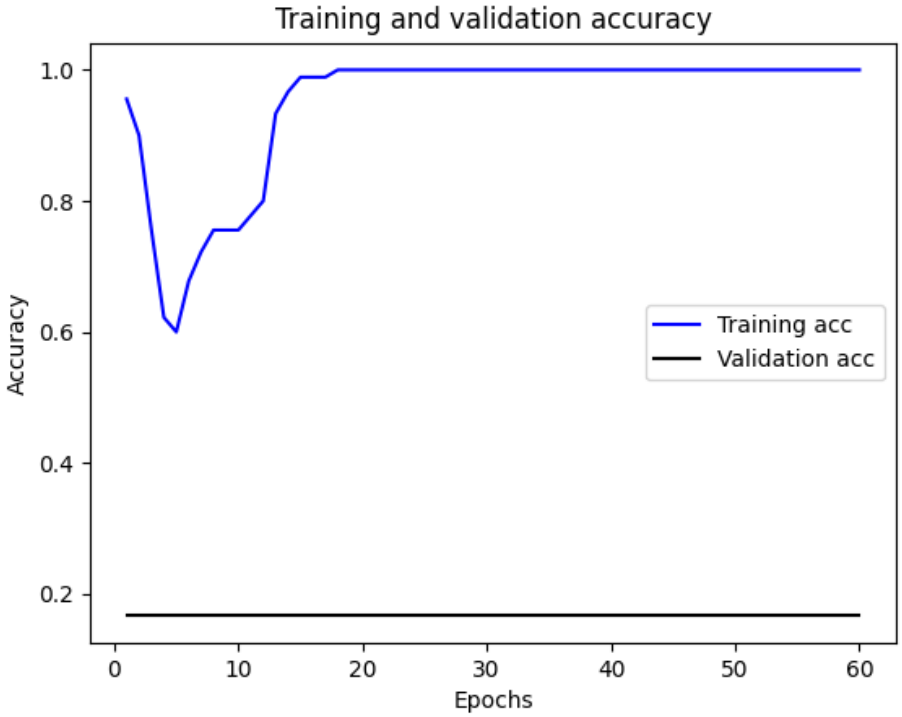
b. 60 эпох, 30 выборок, 25% обучающих данных



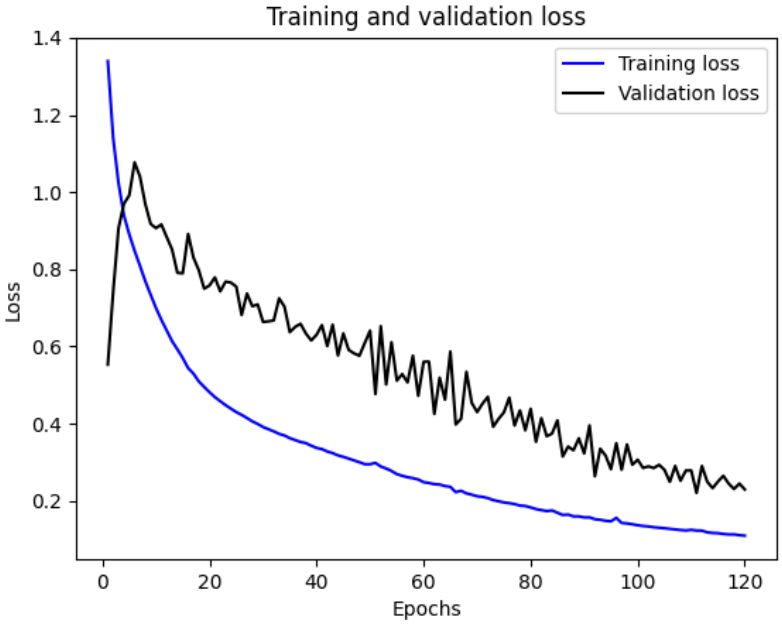


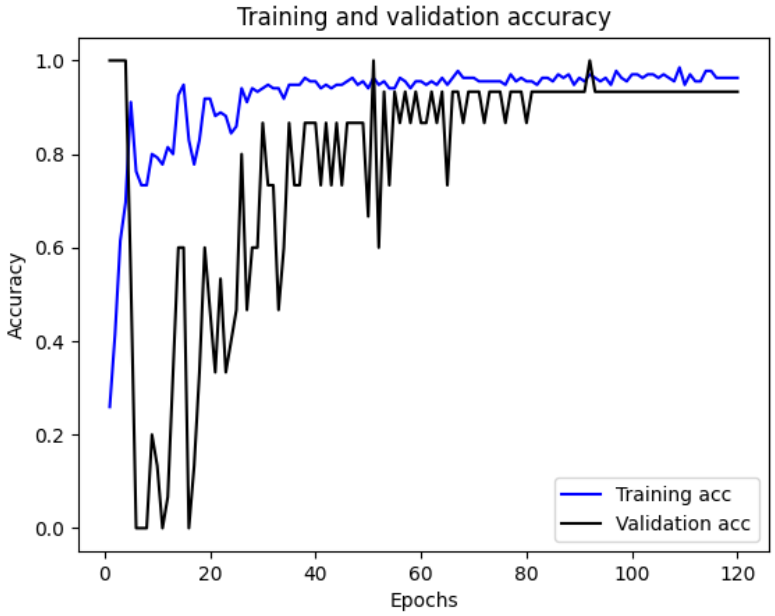
c. 60 эпох, 40 выборок, 40% обучающих данных



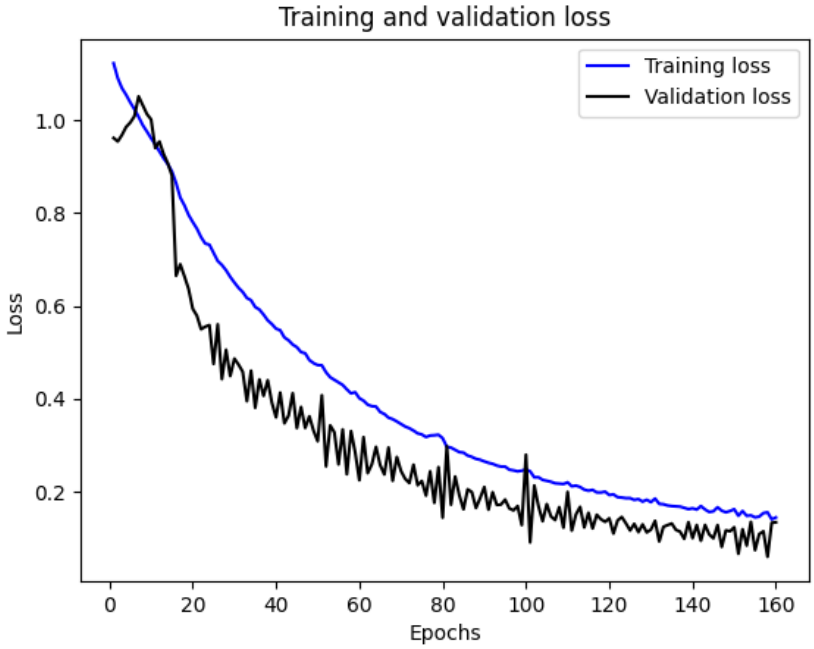


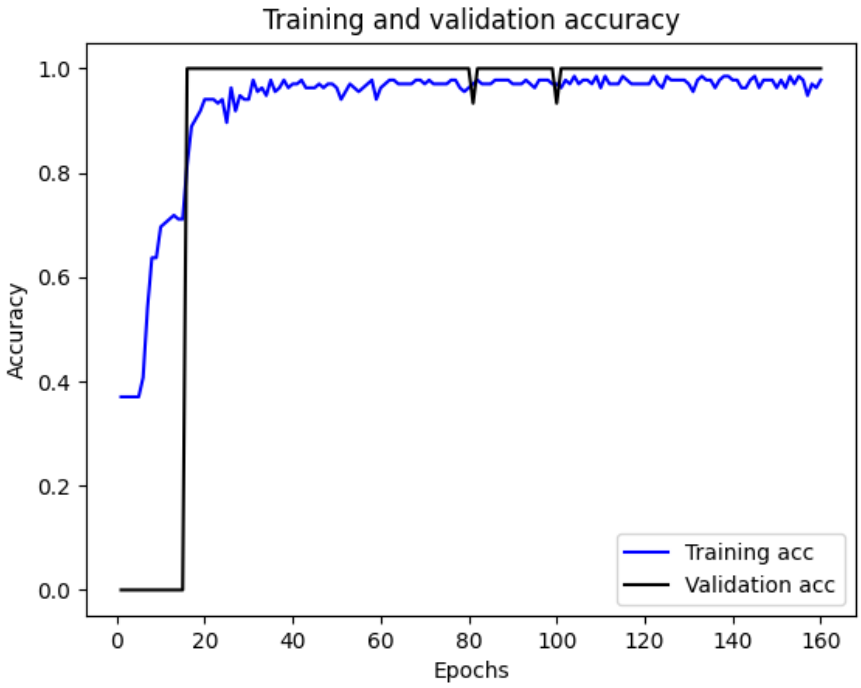
d. 120 эпох, 10 выборок, 10% обучающих данных



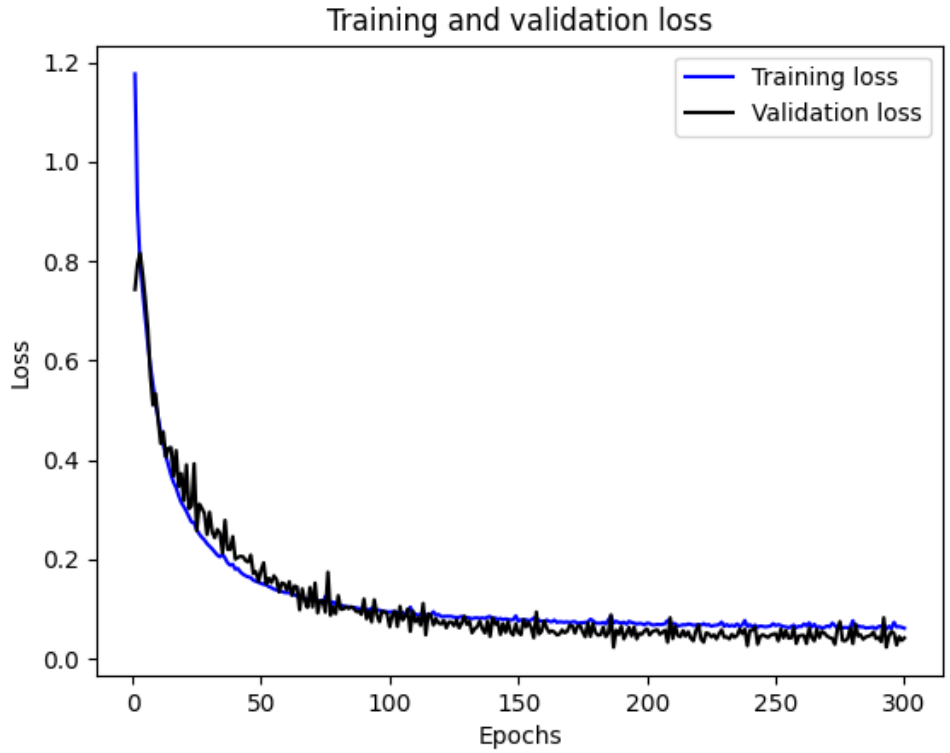


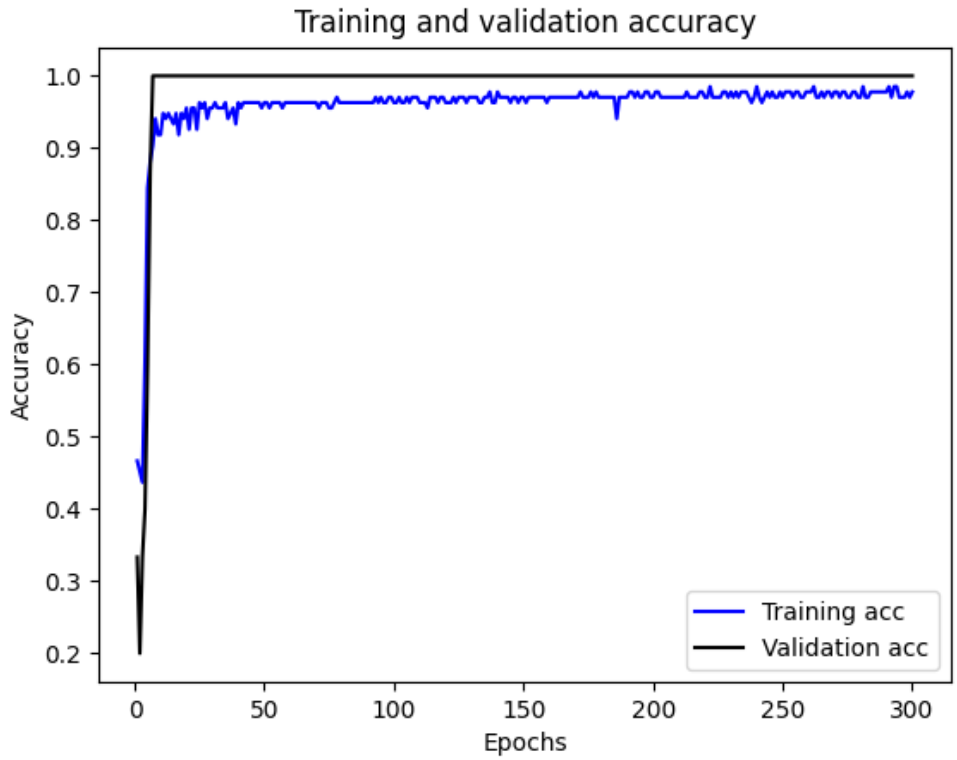
e. 120 эпох, 10 выборок, 10% обучающих данных





f. 300 эпох, 10 выборок, 10% обучающих данных





Выводы

Во время выполнения лабораторной работы была создана ИНС, которая классифицирует классифицирует сорта растения ирис по четырем признакам: размерами пестик и тычинок его цветов. Были изучены различные комбинации архитектуры ИНС меняя количество слоев и нейронов, а также различные параметры обучения модели. Из полученных результатов можно сделать вывод, что наилучшей является модель с тремя слоями, с 12 нейронами на первом слое, 6 нейрона на втором и 3 нейронами на 1, выполняя 10 выборок и используя 10% данных для обучения, при данной конфигурации, точность практически равна 100%, а ошибка меньше 0.1% уже после 100 эпох.

Приложение А

Листинг программы

```
import pandas as pd
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

dataframe = pd.read_csv("iris.csv", header=None)

dataset = dataframe.values

X = dataset[:, 0:4].astype(float)

Y = dataset[:, 4]

encoder = LabelEncoder()

encoder.fit(Y)

encoded_Y = encoder.transform(Y)

dummy_y = to_categorical(encoded_Y)

model = Sequential()

model.add(Dense(12, activation='relu'))

model.add(Dense(6, activation='relu'))

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

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

history = model.fit(X, dummy_y, epochs=300, batch_size=10, validation_split=0.1)

history_dict = history.history

loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'b', label='Training loss')
plt.plot(epochs, val_loss_values, 'k', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
```

```
plt.show()

acc_values = history_dict['accuracy']
val_acc_values = history_dict['val_accuracy']
plt.plot(epochs, acc_values, 'b', label='Training acc')
plt.plot(epochs, val_acc_values, 'k', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```