федеральное государственное автономное образовательное учреждение высшего образования

Санкт-Петербургский национальный исследовательский УНИВЕРСИТЕТ информационных технологий, механики и оптики

**Факультет программной инженерии и компьютерной техники**

**ЛАБОРАТОРНАЯ РАБОТА № 2**

**ПО ДИСЦИПЛИНЕ «РАЗРАБОТКА иНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ»**

Выполнил:

Чудаков М.И.

Группа: P3419

Преподаватель:

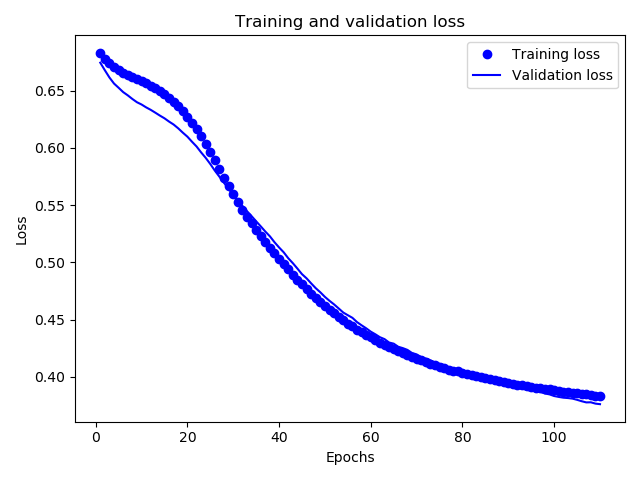
Жукова Н.А.

Санкт-Петербург

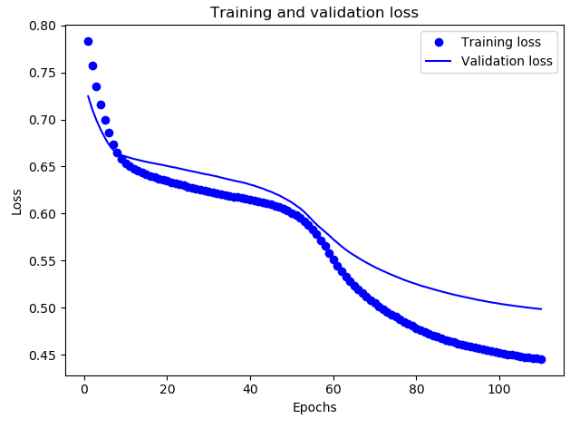
2019/2020**Ход работы**

**import** matplotlib.colors **as** mclr  
**import** matplotlib.pyplot **as** plt  
**import** numpy **as** np  
**from** tensorflow.keras.layers **import** Dense  
**from** tensorflow.keras.models **import** Sequential  
  
**def** genData(size=500):  
 data = np.random.rand(size, 2) \* 2 - 1  
 label = np.zeros([size, 1])  
 **for** i, p **in** enumerate(data):  
 **if** (p[0] + 0.2) \*\* 2 + (0.6 \* p[1]) \*\* 2 >= 0.25:  
 label[i] = 0.  
 **else**:  
 label[i] = 1.  
 div = round(size \* 0.8)  
 train\_data = data[:div, :]  
 test\_data = data[div:, :]  
 train\_label = label[:div, :]  
 test\_label = label[div:, :]  
  
 **return** (train\_data, train\_label), (test\_data, test\_label)  
  
  
**def** drawResults(data, label, prediction):  
 p\_label = np.array([round(x[0]) **for** x **in** prediction])  
 plt.scatter(data[:, 0], data[:, 1], s=30, c=label[:, 0], cmap=mclr.ListedColormap([**'red'**, **'blue'**]))  
 plt.scatter(data[:, 0], data[:, 1], s=10, c=p\_label, cmap=mclr.ListedColormap([**'red'**, **'blue'**]))  
 plt.grid()  
 plt.show()  
  
  
(train\_data, train\_label), (test\_data, test\_label) = genData()  
  
*#В данном месте необходимо создать модель и обучить ее*model = Sequential()  
model.add(Dense(2, activation=**'relu'**))  
model.add(Dense(32, activation=**'relu'**))  
model.add(Dense(1, activation=**'sigmoid'**))  
  
model.compile(optimizer=**'adam'**, loss=**'binary\_crossentropy'**, metrics=[**'accuracy'**])  
H = model.fit(train\_data, train\_label, epochs=110, batch\_size=10, validation\_data=(test\_data, test\_label))  
*# binary\_crossentropy  
# categorical\_crossentropy  
  
#Получение ошибки и точности в процессе обучения*loss = H.history[**'loss'**]  
val\_loss = H.history[**'val\_loss'**]  
acc = H.history[**'accuracy'**]  
val\_acc = H.history[**'val\_accuracy'**]  
epochs = range(1, len(loss) + 1)  
  
*#Построение графика ошибки*plt.plot(epochs, loss, **'bo'**, label=**'Training loss'**)  
plt.plot(epochs, val\_loss, **'b'**, label=**'Validation loss'**)  
plt.title(**'Training and validation loss'**)  
plt.xlabel(**'Epochs'**)  
plt.ylabel(**'Loss'**)  
plt.legend()  
plt.show()  
  
*#Построение графика точности*plt.clf()  
plt.plot(epochs, acc, **'bo'**, label=**'Training acc'**)  
plt.plot(epochs, val\_acc, **'b'**, label=**'Validation acc'**)  
plt.title(**'Training and validation accuracy'**)  
plt.xlabel(**'Epochs'**)  
plt.ylabel(**'Accuracy'**)  
plt.legend()  
plt.show()  
  
*#Получение и вывод результатов на тестовом наборе*results = model.evaluate(test\_data, test\_label)  
print(results)  
  
*#Вывод результатов бинарной классификации*all\_data = np.vstack((train\_data, test\_data))  
all\_label = np.vstack((train\_label, test\_label))  
pred = model.predict(all\_data)  
drawResults(all\_data, all\_label, pred)

При трех слоях (первый – 2 нейрона, второй – 32, третий – 1)



При двух слоях (первый – 2 нейрона, второй – 1)



**Вывод**

При добавлении промежуточного слоя с 32 нейронами показатели точности увеличились. Была выбрана бинарная модель, исходя из выходных данных.