Выполнен 5 вариант:

Задача:

Необходимо дополнить следующий фрагмент кода моделью ИНС, которая способна провести бинарную классификацию по сгенерированным данным:

Была выбрана функция 5 варианта:

```
# 5 вариант
def genData(size=500):
    size1 = size//2
    size2 = size - size1
    x1 = np.random.rand(size1, 1)*1.3 - 0.95
    y1 = np.asarray([3.5*(i+0.2)**2 - 0.8 + (np.random.rand(1)-0.5)/3 for i in x1])
    data1 = np.hstack((x1, y1))
    label1 = np.zeros([size1, 1])
    div1 = round(size1*0.8)
    x2 = np.random.rand(size2, 1)*1.3 - 0.35
    y2 = np.asarray([-3.5*(i-0.2)**2 + 0.8 + (np.random.rand(1)-0.5)/3 for i in x2])
    data2 = np.hstack((x2, y2))
    label2 = np.ones([size2, 1])
    div2 = round(size2*0.8)
    div = div1 + div2
    order = np.random.permutation(div)
    train_data = np.vstack((data1[:div1], data2[:div2]))
    test_data = np.vstack((data1[div1:], data2[div2:]))
    train_label = np.vstack((label1[:div1], label2[:div2]))
    test_label = np.vstack((label1[div1:], label2[div2:]))
    return (train_data[order, :], train_label[order, :]), (test_data, test_label)
```

Была создана модель:

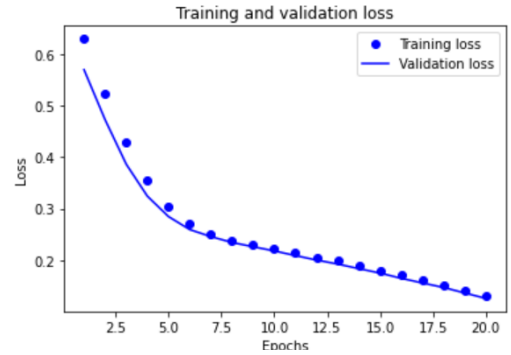
```
# В данном месте необходимо создать модель и обучить ее
model = models.Sequential()
model.add(layers.Dense(64, activation='relu', input_shape=(2,)))
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics='accuracy')
```

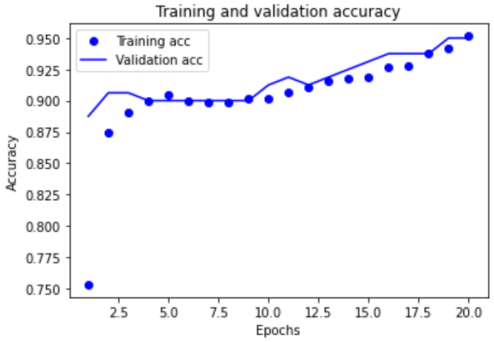
Принимаются значения x, y. Первые два слоя состоят из 64 нейронов. В качестве оптимизатора используется ‘rmsprop’, а в качестве функции потерь – ‘binary_crossentropy’

Данные для обучения: обучение происходит на протяжении 20 эпох.

```
x_val = train_data[:len(train_data) * 2 // 10]
partial_x_train = train_data[len(train_data) * 2 // 10:]
y_val = train_label[:len(train_label) * 2 // 10]
partial_y_train = train_label[len(train_label) * 2 // 10:]
H = model.fit(partial_x_train, partial_y_train, epochs=20, batch_size=50, validation_data=(x_val, y_val))
```

Результат:





Ошибки происходят в тех местах, где обе функции практически касаются друг друга, а также там, где происходит возрастание 1 функции.