Постановка задачи.

Необходимо дополнить фргамент кода моделью ИНС для решения задачи бинарной классификации по сгенерированным данным.

Вариант 5.

Функция генерации данных:

```
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] + .5 >= p[1]) and (p[0] - 0.5 <= p[1]):
            label[i] = 1.
        else:
            label[i] = 0.
    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)
```

Выполнение работы.

В данном случае принимаются значения x, y и была выбрана модель с двумя слоями с функцией активации Relu по 32 нейрона соответственно.

```
model = models.Sequential()
model.add(layers.Dense(32, activation='relu', input_shape=(2,)))
model.add(layers.Dense(32, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

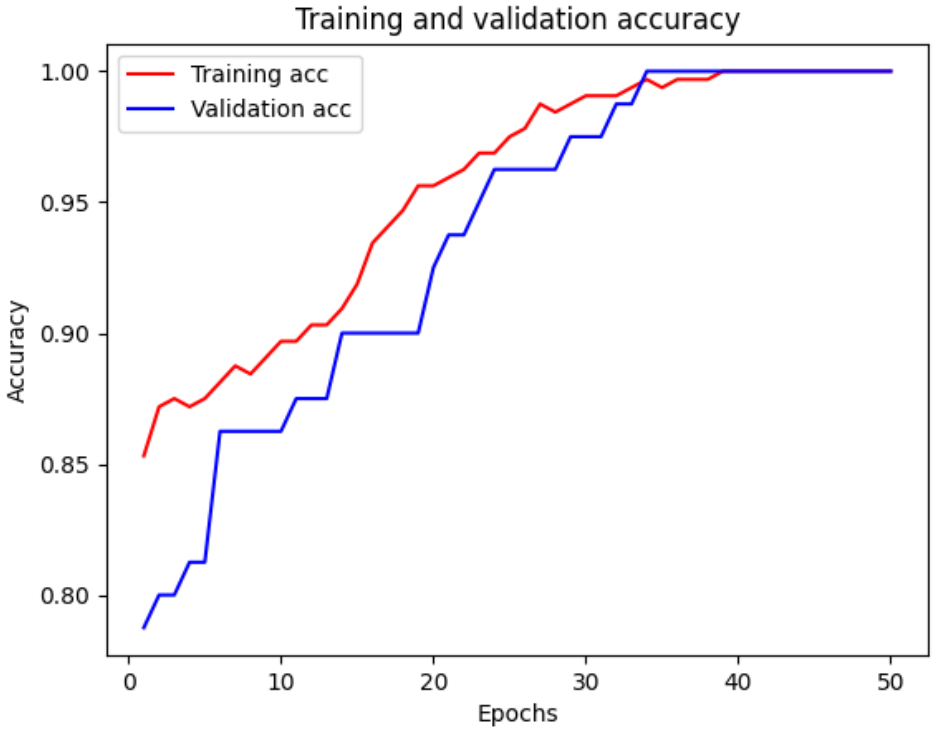
В качестве функции оптимизации выбрана RMSProp, функции потерь – бинарная кроссэнтропия, метрика – точность.

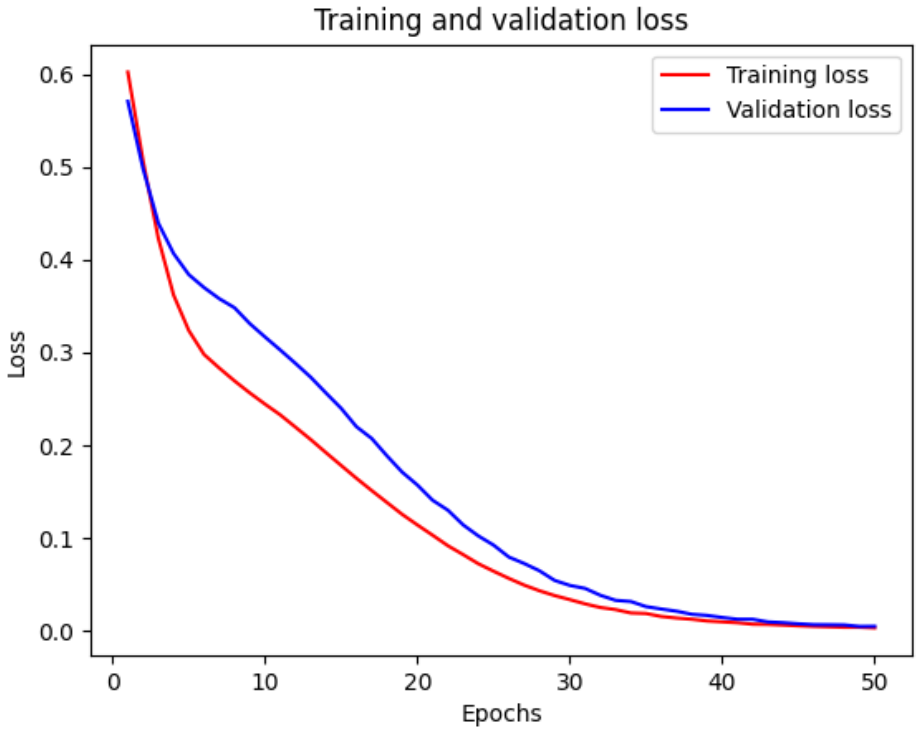
```
model.compile(optimizer='rmsprop', loss='binary_crossentropy',
metrics=['accuracy'])
```

Обучение проводится в течение 50 эпох, с размером выборки 10.

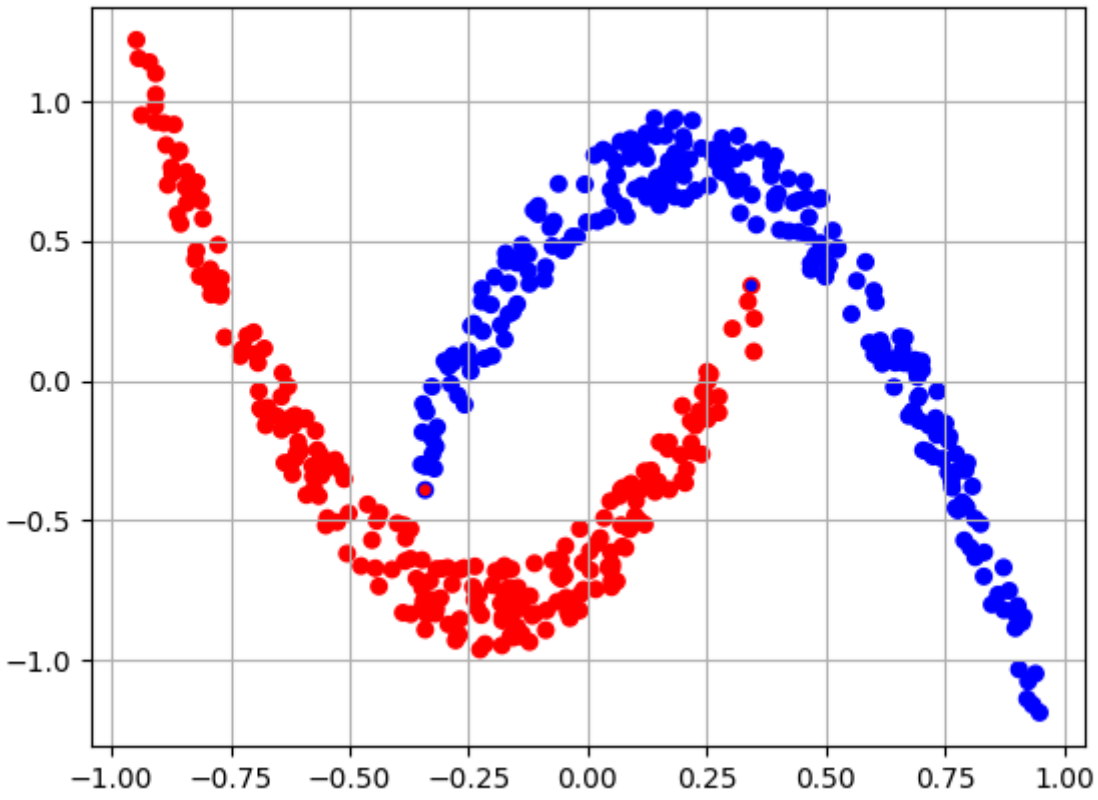
```
x_val = train_data[:len(train_data) * 2 // 10]
partial_x_train = train_data[len(train_data) * 2 // 10:]
y_val = train_label[:len(train_label) * 2 // 10]
partial_y_train = train_label[len(train_label) * 2 // 10:]
H = model.fit(partial_x_train, partial_y_train, epochs=50, batch_size=10,
validation_data=(x_val,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

Точность и ошибки созданной ИНС:





На визуальном распределении видно, что есть 2 ошибки в том случае, когда функции макисмально приближены друг к другу.


1.0
0.5
0.0
−0.5
−1.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00