Übungsblatt 7

1. XOR Problem erstellen

Schreiben Sie eine Funktion `XORData` in Python welche eine Trainings- und Testmenge zurückliefert. Dabei werden zwei Objekte (NumPy-Arrays) zurückgeliefert. Input-Parameter der Funktion ist eine Kantenlänge d. Dabei werden die Daten erzeugt durch vier Quadrate mit dem Kantenlänge $0.1 \leq d \leq 0.8$. Die Mittelpunkte sind (0,0) und (1,0) (0,1) und (1,1). Visualisieren Sie diese Menge durch einen Scatter-Plot für $d = 0.5$.

2. MLP Trainieren

Wir nutzen hier die Menge aus Aufgabe 4 mit $d = 0.5$. Trainieren Sie mittels Keras jeweils eine neuronales Netz mit möglichst wenigen Neuronen, welches die Testmenge trotzdem fehlerfrei klassifizieren könnte. Nutzen hierzu den `adam`-Optimierer mit seinen Default-Werten. Des weiteren nutzen Sie dazu

1. die `binary_crossentropy` als Loss-Function zusammen mit einem Neuron mit Sigmoid-Aktivierung im Outputlayer
2. den `mse` als Loss-Function zusammen mit einem Neuron mit Sigmoid-Aktivierung im Outputlayer
3. die `categorical_crossentropy` als Loss-Function zusammen mit zwei Neuronen mit Softmax-Aktivierung im Outputlayer

Dabei wenden Sie als Optimierer `adam` mit den Default-Werten und ein Early-Stopping an. Sorgen Sie dafür, dass die Genauigkeit auf der Testmenge (als Validierungsmenge übergeben) mitprotokolliert wird. Manchmal konvergiert das Training und manchmal nicht. Trainieren Sie jedes Netz mindestens 10 Mal und mitteln Sie den Verlauf von `val_acc` in der History.

3. MLP Auswerten

Plotten Sie die Genauigkeit Ihrer Netze, die Sie gemittelt und in History gespeichert haben, und vergleichen Sie die Werte. Welche Architektur liefert die beste Klassifikation?
Stellen Sie mit Hilfe von drei **pcolormesh**-Plots übersichtlich dar, wie die Netze die Bereiche von $x \in [-0.5, 1.5]$ und $y \in [-0.5, 1.5]$ klassifiziert haben. Überlegen Sie, ob diese Ergebnisse dem **"XOR-Problem"** aus Kapitel 7.2 entsprechen.