Neural Networks - intro

Part 1 - XOR

1. Using the XOR dataset below, train (400 epochs) a neural network (NN) using 2, 3, 4, and 5 hidden layers (where each layer has only 2 neurons). For each n layers, store the resulting accuracy along with n. Plot the results to find what the optimal number of layers is.
2. Repeat the above with 3 neurons in each Hidden layers. How do these results compare to the 2 neuron layers?
3. Repeat the above with 4 neurons in each Hidden layers. How do these results compare to the 2 and 3 neuron layers?
4. Using the most optimal configuraion (n-layers, k-neurons per layer), compare how `tanh`, `sigmoid`, `softplus` and `relu` effect the loss after 400 epochs. Try other Activation functions as well (https://keras.io/activations/)
5. Again with the most optimal setup, try other optimizers (instead of `SGD`) and report on the loss score. (https://keras.io/optimizers/)

Part 2 - BYOD (Bring your own Dataset)

Using your own dataset, experiment and find the best Neural Network configuration. You may use any resource to improve results, just reference it.

While you may use any dataset, I'd prefer you didn't use the diabetes dataset used in the lesson.

https://stackoverflow.com/questions/34673164/how-to-train-and-tune-an-artificial-multilayer-perceptron-neural-network-using-k

https://keras.io/

In [1]:
```
!pip3 install tensorflow keras
```

```
Requirement already satisfied: tensorflow in /Users/mhalt/opt/anaconda3/lib/py
thon3.9/site-packages (2.16.2)
Requirement already satisfied: keras in /Users/mhalt/opt/anaconda3/lib/python
3.9/site-packages (3.4.1)
Requirement already satisfied: typing-extensions>=3.6.6 in /Users/mhalt/opt/an
aconda3/lib/python3.9/site-packages (from tensorflow) (4.12.2)
Requirement already satisfied: absl-py>=1.0.0 in /Users/mhalt/opt/anaconda3/li
b/python3.9/site-packages (from tensorflow) (2.1.0)
Requirement already satisfied: setuptools in /Users/mhalt/opt/anaconda3/lib/py
thon3.9/site-packages (from tensorflow) (63.4.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /Users/
mhalt/opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (0.37.1)
Requirement already satisfied: wrapt>=1.11.0 in /Users/mhalt/opt/anaconda3/li
b/python3.9/site-packages (from tensorflow) (1.14.1)
Requirement already satisfied: six>=1.12.0 in /Users/mhalt/opt/anaconda3/lib/p
ython3.9/site-packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in /Users/mhalt/opt/anaconda3/
lib/python3.9/site-packages (from tensorflow) (2.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in /Users/mhalt/opt/anacond
a3/lib/python3.9/site-packages (from tensorflow) (0.2.0)
Requirement already satisfied: libclang>=13.0.0 in /Users/mhalt/opt/anaconda3/
lib/python3.9/site-packages (from tensorflow) (18.1.1)
Requirement already satisfied: astunparse>=1.6.0 in /Users/mhalt/opt/anaconda
3/lib/python3.9/site-packages (from tensorflow) (1.6.3)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /Users/mhalt/opt/anacond
a3/lib/python3.9/site-packages (from tensorflow) (1.65.2)
Requirement already satisfied: opt-einsum>=2.3.2 in /Users/mhalt/opt/anaconda
3/lib/python3.9/site-packages (from tensorflow) (3.3.0)
Requirement already satisfied: ml-dtypes~=0.3.1 in /Users/mhalt/opt/anaconda3/
lib/python3.9/site-packages (from tensorflow) (0.3.2)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=
4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in /Users/mhalt/opt/anaconda3/lib/python3.
9/site-packages (from tensorflow) (4.25.4)
Requirement already satisfied: flatbuffers>=23.5.26 in /Users/mhalt/opt/anacon
da3/lib/python3.9/site-packages (from tensorflow) (24.3.25)
Requirement already satisfied: requests<3,>=2.21.0 in /Users/mhalt/opt/anacond
a3/lib/python3.9/site-packages (from tensorflow) (2.28.1)
Requirement already satisfied: tensorboard<2.17,>=2.16 in /Users/mhalt/opt/ana
conda3/lib/python3.9/site-packages (from tensorflow) (2.16.2)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /Users/m
halt/opt/anaconda3/lib/python3.9/site-packages (from tensorflow) (0.6.0)
Requirement already satisfied: packaging in /Users/mhalt/opt/anaconda3/lib/pyt
hon3.9/site-packages (from tensorflow) (21.3)
Requirement already satisfied: h5py>=3.10.0 in /Users/mhalt/opt/anaconda3/lib/
python3.9/site-packages (from tensorflow) (3.11.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in /Users/mhalt/opt/anacon
da3/lib/python3.9/site-packages (from tensorflow) (1.26.4)
Requirement already satisfied: optree in /Users/mhalt/opt/anaconda3/lib/python
3.9/site-packages (from keras) (0.12.1)
Requirement already satisfied: rich in /Users/mhalt/opt/anaconda3/lib/python3.
9/site-packages (from keras) (13.7.1)
Requirement already satisfied: namex in /Users/mhalt/opt/anaconda3/lib/python
3.9/site-packages (from keras) (0.0.8)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /Users/mhalt/opt/anaconda
3/lib/python3.9/site-packages (from astunparse>=1.6.0->tensorflow) (0.37.1)
Requirement already satisfied: idna<4,>=2.5 in /Users/mhalt/opt/anaconda3/lib/
python3.9/site-packages (from requests<3,>=2.21.0->tensorflow) (3.3)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /Users/mhalt/opt/anaco
nda3/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorflow) (1.26.
11)
```

```
Requirement already satisfied: charset-normalizer<3,>=2 in /Users/mhalt/opt/an
aconda3/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorflow) (2.
0.4)
Requirement already satisfied: certifi>=2017.4.17 in /Users/mhalt/opt/anaconda
3/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorflow) (2022.9.2
4)
Requirement already satisfied: werkzeug>=1.0.1 in /Users/mhalt/opt/anaconda3/l
ib/python3.9/site-packages (from tensorboard<2.17,>=2.16->tensorflow) (2.0.3)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /User
s/mhalt/opt/anaconda3/lib/python3.9/site-packages (from tensorboard<2.17,>=2.1
6->tensorflow) (0.7.2)
Requirement already satisfied: markdown>=2.6.8 in /Users/mhalt/opt/anaconda3/l
ib/python3.9/site-packages (from tensorboard<2.17,>=2.16->tensorflow) (3.3.4)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /Users/mhalt/opt/an
aconda3/lib/python3.9/site-packages (from packaging->tensorflow) (3.0.9)
Requirement already satisfied: markdown-it-py>=2.2.0 in /Users/mhalt/opt/anaco
nda3/lib/python3.9/site-packages (from rich->keras) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /Users/mhalt/opt/ana
conda3/lib/python3.9/site-packages (from rich->keras) (2.18.0)
Requirement already satisfied: mdurl~=0.1 in /Users/mhalt/opt/anaconda3/lib/py
thon3.9/site-packages (from markdown-it-py>=2.2.0->rich->keras) (0.1.2)
```

In [ ]:
```
from keras.models import Sequential
from keras.layers import Dense
from keras.optimizers import SGD  #Stochastic Gradient Descent
```

In [ ]:
```
import numpy as np
np.random.seed(7)
import matplotlib.pyplot as plt
%matplotlib inline
```

In [ ]:
```
n = 40
xx = np.random.random((n,1))
yy = np.random.random((n,1))
```

In [ ]:
```
X = np.array([np.array([xx,-xx,-xx,xx]),np.array([yy,-yy,yy,-yy])]).reshape(2,4
y = np.array([np.ones([2*n]),np.zeros([2*n])]).reshape(4*n)
```

In [ ]:
```
plt.scatter(*zip(*X), c=y)
```

In [ ]:
```
model = Sequential()

model.add(Dense(2, input_dim=2, activation='tanh'))  #sigmoid, relu
# model.add(Dense(2, activation='tanh'))
model.add(Dense(1, activation='sigmoid'))
# model.add(Dense(1,input_dim=2, activation='sigmoid'))


sgd = SGD(lr=0.1)
model.compile(loss='binary_crossentropy', optimizer='sgd')

model.fit(X, y, batch_size=2, epochs=400) #160/4 = 40 per epoch
print(model.predict_proba(X).reshape(4*n))

# evaluate the model
scores = model.evaluate(X, y)
```

In [ ]:
```
print(model.predict_proba(X).reshape(4*n))
```

In [ ]:
```
scores = model.evaluate(X, y)
scores, model.metrics_names
```

In [ ]:
```
plt.scatter(*zip(*X), c=model.predict_classes(X))
```

In [ ]:
```
plt.scatter(*zip(*X), c=model.predict(X))
```

Using Diabetes data

http://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes/pima-indians-diabetes.data

1. Number of times pregnant
2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thickness (mm)
5. 2-Hour serum insulin (mu U/ml)
6. Body mass index (weight in kg/(height in m)^2)
7. Diabetes pedigree function
8. Age (years)
9. Class variable (0 or 1)

In [ ]:
```
from keras.models import Sequential
import pandas as pd
from keras.utils import to_categorical
coffee = pd.read_csv('~/Desktop/df_arabica_clean.csv', delimiter=",")
coffee.head

# split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]
```

In [ ]:
```
# create model
model = Sequential()
model.add(Dense(16, input_dim=8, activation='tanh'))
model.add(Dense(16, activation='tanh'))
model.add(Dense(1, activation='sigmoid'))
# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy
# Fit the model
model.fit(X, Y, epochs=1000, batch_size=10)
# evaluate the model
scores = model.evaluate(X, Y)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

In [ ]:
```
def byod2(layers, neurons, act, opt):
    model = Sequential()
    model.add(Dense(neurons, input_dim= 13, activation=act))
```

```
    for _ in range(layers - 1):
        model.add(Dense(neurons, activation=act))
    model.add(Dense(3, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer=opt, metrics=['ac
    record = model.fit(X, y, epochs=100, batch_size=10, validation_split=0.4, 
    loss, accuracy = model.evaluate(X, y, verbose=0)
    return loss, accuracy, record.history
```

In [ ]:
```
configs = [
    {'layers': 2, 'neurons': 3, 'act': 'tanh', 'opt': 'Lion'},
    {'layers': 5, 'neurons': 3, 'act': 'tanh', 'opt': 'Lion'},
    {'layers': 1, 'neurons': 4, 'act': 'tanh', 'opt': 'Lion'},
    {'layers': 2, 'neurons': 3, 'act': 'selu', 'opt': 'Lion'},
    {'layers': 5, 'neurons': 3, 'act': 'selu', 'opt': 'Lion'},
    {'layers': 1, 'neurons': 4, 'act': 'selu', 'opt': 'Lion'},
    {'layers': 2, 'neurons': 3, 'act': 'tanh', 'opt': 'Adam'},
    {'layers': 5, 'neurons': 3, 'act': 'tanh', 'opt': 'Adam'},
    {'layers': 1, 'neurons': 4, 'act': 'tanh', 'opt': 'Adam'},
    {'layers': 2, 'neurons': 3, 'act': 'selu', 'opt': 'Adam'},
    {'layers': 5, 'neurons': 2, 'act': 'selu', 'opt': 'Adam'},
    {'layers': 1, 'neurons': 4, 'act': 'selu', 'opt': 'Adam'}
]
```

In [ ]:
```
results2 = []

for config in configs:
    layers = config['layers']
    neurons = config['neurons']
    act = config['act']
    opt = config['opt']
    loss, accuracy, record = byod2(layers, neurons, act,  opt)
    results2.append({'layers': layers, 'neurons': neurons, 'activation': act,
                     'optimizater':opt, 'loss':loss, 'accuracy':accuracy, 'reco
```

In [ ]:
```

```

In [ ]:
```

```