

## Activity 1.2 : Training Neural Networks

### Objective(s):

This activity aims to demonstrate how to train neural networks using keras

### Intended Learning Outcomes (ILOs):

- Demonstrate how to build and train neural networks
- Demonstrate how to evaluate and plot the model using training and validation loss

### Resources:

- Jupyter Notebook

CI Pima Diabetes Dataset

- pima-indians-diabetes.csv

### Procedures

Load the necessary libraries

```
In [4]: from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive

In [ ]: from __future__ import absolute_import, division, print_function

import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, precision_recall_curve, roc_auc_score, roc_curve, accuracy_score
from sklearn.ensemble import RandomForestClassifier

import seaborn as sns

%matplotlib inline

In [ ]: ## Import Keras objects for Deep Learning

from keras.models import Sequential
from keras.layers import Input, Dense, Flatten, Dropout, BatchNormalization
from keras.optimizers import Adam, SGD, RMSprop
```

Load the dataset

```
In [ ]: filepath = "/content/drive/MyDrive/Data folder/pima-indians-diabetes.csv"
names = ["times_pregnant", "glucose_tolerance_test", "blood_pressure", "skin_thickness", "insulin",
        "bmi", "pedigree_function", "age", "has_diabetes"]
diabetes_df = pd.read_csv(filepath, names=names)
```

Check the top 5 samples of the data

```
In [ ]: print(diabetes_df.shape)
diabetes_df.sample(5)

(768, 9)
```

	times_pregnant	glucose_tolerance_test	blood_pressure	skin_thickness	insulin	bmi	pedigree_function	age	has_diabetes
275	2	100	70	52	57	40.5	0.677	25	0
367	0	101	64	17	0	21.0	0.252	21	0
550	1	116	70	28	0	27.4	0.204	21	0
711	5	126	78	27	22	29.6	0.439	40	0
407	0	101	62	0	0	21.9	0.336	25	0

```
In [ ]: diabetes_df.dtypes

Out[ ]: times_pregnant      int64
glucose_tolerance_test  int64
blood_pressure          int64
skin_thickness          int64
insulin                 int64
bmi                     float64
pedigree_function       float64
age                     int64
has_diabetes            int64
dtype: object
```

```
In [ ]: X = diabetes_df.iloc[:, :-1].values
y = diabetes_df["has_diabetes"].values
```

Split the data to Train, and Test (75%, 25%)

```
In [ ]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=11111)
```

```
In [ ]: np.mean(y), np.mean(1-y)

Out[ ]: (0.3489583333333333, 0.6510416666666666)
```

Build a single hidden layer neural network using 12 nodes. Use the sequential model with single layer network and input shape to 8.

Normalize the data

```
In [ ]: normalizer = StandardScaler()
X_train_norm = normalizer.fit_transform(X_train)
X_test_norm = normalizer.transform(X_test)
```

Define the model:

- Input size is 8-dimensional
- 1 hidden layer, 12 hidden nodes, sigmoid activation

- Final layer with one node and sigmoid activation (standard for binary classification)

```
In [ ]: model = Sequential([
    Dense(12, input_shape=(8,)), activation="relu"),
    Dense(1, activation="sigmoid")
])
```

View the model summary

```
In [ ]: model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 12)	108
dense_1 (Dense)	(None, 1)	13

=====  
 Total params: 121 (484.00 Byte)  
 Trainable params: 121 (484.00 Byte)  
 Non-trainable params: 0 (0.00 Byte)

Train the model

- Compile the model with optimizer, loss function and metrics
- Use the fit function to return the run history.

```
In [ ]: model.compile(SGD(lr = .003), "binary_crossentropy", metrics=["accuracy"])
run_hist_1 = model.fit(X_train_norm, y_train, validation_data=(X_test_norm, y_test), epochs=200)
```

WARNING:absl:`lr` is deprecated in Keras optimizer, please use `learning\_rate` or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.SGD.

Epoch 1/200  
18/18 [=====] - 1s 14ms/step - loss: 0.8674 - accuracy: 0.4774 - val\_loss: 0.8803 - val\_accuracy: 0.4792  
Epoch 2/200  
18/18 [=====] - 0s 4ms/step - loss: 0.8252 - accuracy: 0.4965 - val\_loss: 0.8382 - val\_accuracy: 0.4896  
Epoch 3/200  
18/18 [=====] - 0s 4ms/step - loss: 0.7907 - accuracy: 0.5226 - val\_loss: 0.8034 - val\_accuracy: 0.5104  
Epoch 4/200  
18/18 [=====] - 0s 4ms/step - loss: 0.7621 - accuracy: 0.5417 - val\_loss: 0.7741 - val\_accuracy: 0.5312  
Epoch 5/200  
18/18 [=====] - 0s 4ms/step - loss: 0.7379 - accuracy: 0.5590 - val\_loss: 0.7492 - val\_accuracy: 0.5521  
Epoch 6/200  
18/18 [=====] - 0s 5ms/step - loss: 0.7173 - accuracy: 0.5955 - val\_loss: 0.7276 - val\_accuracy: 0.5885  
Epoch 7/200  
18/18 [=====] - 0s 4ms/step - loss: 0.6996 - accuracy: 0.6181 - val\_loss: 0.7088 - val\_accuracy: 0.5833  
Epoch 8/200  
18/18 [=====] - 0s 3ms/step - loss: 0.6840 - accuracy: 0.6337 - val\_loss: 0.6924 - val\_accuracy: 0.6146  
Epoch 9/200  
18/18 [=====] - 0s 4ms/step - loss: 0.6704 - accuracy: 0.6493 - val\_loss: 0.6782 - val\_accuracy: 0.6406  
Epoch 10/200  
18/18 [=====] - 0s 5ms/step - loss: 0.6581 - accuracy: 0.6528 - val\_loss: 0.6656 - val\_accuracy: 0.6354  
Epoch 11/200  
18/18 [=====] - 0s 4ms/step - loss: 0.6475 - accuracy: 0.6615 - val\_loss: 0.6544 - val\_accuracy: 0.6406  
Epoch 12/200  
18/18 [=====] - 0s 4ms/step - loss: 0.6379 - accuracy: 0.6753 - val\_loss: 0.6442 - val\_accuracy: 0.6510  
Epoch 13/200  
18/18 [=====] - 0s 5ms/step - loss: 0.6289 - accuracy: 0.6788 - val\_loss: 0.6349 - val\_accuracy: 0.6615  
Epoch 14/200  
18/18 [=====] - 0s 3ms/step - loss: 0.6207 - accuracy: 0.6892 - val\_loss: 0.6265 - val\_accuracy: 0.6667  
Epoch 15/200  
18/18 [=====] - 0s 4ms/step - loss: 0.6134 - accuracy: 0.6927 - val\_loss: 0.6187 - val\_accuracy: 0.6823  
Epoch 16/200  
18/18 [=====] - 0s 4ms/step - loss: 0.6065 - accuracy: 0.7066 - val\_loss: 0.6116 - val\_accuracy: 0.6875  
Epoch 17/200  
18/18 [=====] - 0s 4ms/step - loss: 0.6003 - accuracy: 0.7153 - val\_loss: 0.6048 - val\_accuracy: 0.6979  
Epoch 18/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5945 - accuracy: 0.7188 - val\_loss: 0.5986 - val\_accuracy: 0.6979  
Epoch 19/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5888 - accuracy: 0.7222 - val\_loss: 0.5928 - val\_accuracy: 0.7083  
Epoch 20/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5836 - accuracy: 0.7222 - val\_loss: 0.5875 - val\_accuracy: 0.7031  
Epoch 21/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5786 - accuracy: 0.7257 - val\_loss: 0.5825 - val\_accuracy: 0.7135  
Epoch 22/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5739 - accuracy: 0.7292 - val\_loss: 0.5779 - val\_accuracy: 0.7188  
Epoch 23/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5696 - accuracy: 0.7274 - val\_loss: 0.5737 - val\_accuracy: 0.7240  
Epoch 24/200  
18/18 [=====] - 0s 3ms/step - loss: 0.5653 - accuracy: 0.7274 - val\_loss: 0.5696 - val\_accuracy: 0.7240  
Epoch 25/200  
18/18 [=====] - 0s 3ms/step - loss: 0.5609 - accuracy: 0.7292 - val\_loss: 0.5658 - val\_accuracy: 0.7240  
Epoch 26/200  
18/18 [=====] - 0s 3ms/step - loss: 0.5570 - accuracy: 0.7326 - val\_loss: 0.5621 - val\_accuracy: 0.7240  
Epoch 27/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5532 - accuracy: 0.7344 - val\_loss: 0.5587 - val\_accuracy: 0.7240  
Epoch 28/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5497 - accuracy: 0.7344 - val\_loss: 0.5556 - val\_accuracy: 0.7240  
Epoch 29/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5462 - accuracy: 0.7361 - val\_loss: 0.5526 - val\_accuracy: 0.7240  
Epoch 30/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5430 - accuracy: 0.7361 - val\_loss: 0.5497 - val\_accuracy: 0.7240  
Epoch 31/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5400 - accuracy: 0.7413 - val\_loss: 0.5470 - val\_accuracy: 0.7240  
Epoch 32/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5370 - accuracy: 0.7413 - val\_loss: 0.5445 - val\_accuracy: 0.7240  
Epoch 33/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5341 - accuracy: 0.7413 - val\_loss: 0.5420 - val\_accuracy: 0.7240  
Epoch 34/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5313 - accuracy: 0.7448 - val\_loss: 0.5396 - val\_accuracy: 0.7240  
Epoch 35/200  
18/18 [=====] - 0s 3ms/step - loss: 0.5286 - accuracy: 0.7431 - val\_loss: 0.5373 - val\_accuracy: 0.7240  
Epoch 36/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5259 - accuracy: 0.7413 - val\_loss: 0.5351 - val\_accuracy: 0.7188  
Epoch 37/200  
18/18 [=====] - 0s 6ms/step - loss: 0.5235 - accuracy: 0.7448 - val\_loss: 0.5331 - val\_accuracy: 0.7188  
Epoch 38/200  
18/18 [=====] - 0s 6ms/step - loss: 0.5209 - accuracy: 0.7465 - val\_loss: 0.5310 - val\_accuracy: 0.7188  
Epoch 39/200  
18/18 [=====] - 0s 6ms/step - loss: 0.5186 - accuracy: 0.7465 - val\_loss: 0.5291 - val\_accuracy: 0.7135  
Epoch 40/200  
18/18 [=====] - 0s 6ms/step - loss: 0.5162 - accuracy: 0.7465 - val\_loss: 0.5271 - val\_accuracy: 0.7135  
Epoch 41/200  
18/18 [=====] - 0s 6ms/step - loss: 0.5139 - accuracy: 0.7431 - val\_loss: 0.5253 - val\_accuracy: 0.7188  
Epoch 42/200  
18/18 [=====] - 0s 5ms/step - loss: 0.5117 - accuracy: 0.7465 - val\_loss: 0.5237 - val\_accuracy: 0.7135  
Epoch 43/200  
18/18 [=====] - 0s 5ms/step - loss: 0.5096 - accuracy: 0.7483 - val\_loss: 0.5220 - val\_accuracy: 0.7135  
Epoch 44/200  
18/18 [=====] - 0s 5ms/step - loss: 0.5076 - accuracy: 0.7483 - val\_loss: 0.5205 - val\_accuracy: 0.7135  
Epoch 45/200  
18/18 [=====] - 0s 4ms/step - loss: 0.5056 - accuracy: 0.7448 - val\_loss: 0.5189 - val\_accuracy: 0.7135  
Epoch 46/200  
18/18 [=====] - 0s 5ms/step - loss: 0.5036 - accuracy: 0.7500 - val\_loss: 0.5176 - val\_accuracy: 0.7135  
Epoch 47/200  
18/18 [=====] - 0s 5ms/step - loss: 0.5016 - accuracy: 0.7535 - val\_loss: 0.5163 - val\_accuracy: 0.7135  
Epoch 48/200  
18/18 [=====] - 0s 4ms/step - loss: 0.4998 - accuracy: 0.7569 - val\_loss: 0.5151 - val\_accuracy: 0.7135  
Epoch 49/200  
18/18 [=====] - 0s 5ms/step - loss: 0.4981 - accuracy: 0.7587 - val\_loss: 0.5139 - val\_accuracy: 0.7188  
Epoch 50/200  
18/18 [=====] - 0s 6ms/step - loss: 0.4962 - accuracy: 0.7604 - val\_loss: 0.5128 - val\_accuracy: 0.7292  
Epoch 51/200  
18/18 [=====] - 0s 5ms/step - loss: 0.4946 - accuracy: 0.7622 - val\_loss: 0.5117 - val\_accuracy: 0.7292  
Epoch 52/200  
18/18 [=====] - 0s 5ms/step - loss: 0.4927 - accuracy: 0.7622 - val\_loss: 0.5106 - val\_accuracy: 0.7292  
Epoch 53/200  
18/18 [=====] - 0s 5ms/step - loss: 0.4910 - accuracy: 0.7587 - val\_loss: 0.5096 - val\_accuracy: 0.7344  
Epoch 54/200  
18/18 [=====] - 0s 7ms/step - loss: 0.4895 - accuracy: 0.7587 - val\_loss: 0.5086 - val\_accuracy: 0.7396  
Epoch 55/200  
18/18 [=====] - 0s 6ms/step - loss: 0.4877 - accuracy: 0.7604 - val\_loss: 0.5077 - val\_accuracy: 0.7448  
Epoch 56/200  
18/18 [=====] - 0s 5ms/step - loss: 0.4861 - accuracy: 0.7587 - val\_loss: 0.5068 - val\_accuracy: 0.7448  
Epoch 57/200  
18/18 [=====] - 0s 5ms/step - loss: 0.4845 - accuracy: 0.7622 - val\_loss: 0.5060 - val\_accuracy: 0.7448  
Epoch 58/200  
18/18 [=====] - 0s 5ms/step - loss: 0.4831 - accuracy: 0.7691 - val\_loss: 0.5052 - val\_accuracy: 0.7500  
Epoch 59/200  
18/18 [=====] - 0s 4ms/step - loss: 0.4818 - accuracy: 0.7708 - val\_loss: 0.5044 - val\_accuracy: 0.7552  
Epoch 60/200  
18/18 [=====] - 0s 5ms/step - loss: 0.4803 - accuracy: 0.7726 - val\_loss: 0.5037 - val\_accuracy: 0.7448  
Epoch 61/200  
18/18 [=====] - 0s 6ms/step - loss: 0.4788 - accuracy: 0.7743 - val\_loss: 0.5030 - val\_accuracy: 0.7448  
Epoch 62/200  
18/18 [=====] - 0s 6ms/step - loss: 0.4773 - accuracy: 0.7691 - val\_loss: 0.5023 - val\_accuracy: 0.7448  
Epoch 63/200  
18/18 [=====] - 0s 6ms/step - loss: 0.4760 - accuracy: 0.7708 - val\_loss: 0.5017 - val\_accuracy: 0.7448  
Epoch 64/200  
18/18 [=====] - 0s 6ms/step - loss: 0.4749 - accuracy: 0.7726 - val\_loss: 0.5012 - val\_accuracy: 0.7500

Epoch 65/200	18/18 [=====]	- 0s 7ms/step	- loss: 0.4736	- accuracy: 0.7778	- val_loss: 0.5006	- val_accuracy: 0.7500
Epoch 66/200	18/18 [=====]	- 0s 5ms/step	- loss: 0.4725	- accuracy: 0.7795	- val_loss: 0.5001	- val_accuracy: 0.7500
Epoch 67/200	18/18 [=====]	- 0s 5ms/step	- loss: 0.4715	- accuracy: 0.7795	- val_loss: 0.4996	- val_accuracy: 0.7552
Epoch 68/200	18/18 [=====]	- 0s 6ms/step	- loss: 0.4702	- accuracy: 0.7778	- val_loss: 0.4992	- val_accuracy: 0.7604
Epoch 69/200	18/18 [=====]	- 0s 6ms/step	- loss: 0.4692	- accuracy: 0.7795	- val_loss: 0.4987	- val_accuracy: 0.7604
Epoch 70/200	18/18 [=====]	- 0s 6ms/step	- loss: 0.4683	- accuracy: 0.7778	- val_loss: 0.4983	- val_accuracy: 0.7656
Epoch 71/200	18/18 [=====]	- 0s 5ms/step	- loss: 0.4673	- accuracy: 0.7795	- val_loss: 0.4979	- val_accuracy: 0.7656
Epoch 72/200	18/18 [=====]	- 0s 6ms/step	- loss: 0.4663	- accuracy: 0.7812	- val_loss: 0.4976	- val_accuracy: 0.7656
Epoch 73/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4654	- accuracy: 0.7778	- val_loss: 0.4972	- val_accuracy: 0.7656
Epoch 74/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4644	- accuracy: 0.7778	- val_loss: 0.4969	- val_accuracy: 0.7656
Epoch 75/200	18/18 [=====]	- 0s 3ms/step	- loss: 0.4636	- accuracy: 0.7760	- val_loss: 0.4966	- val_accuracy: 0.7656
Epoch 76/200	18/18 [=====]	- 0s 5ms/step	- loss: 0.4629	- accuracy: 0.7760	- val_loss: 0.4963	- val_accuracy: 0.7656
Epoch 77/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4621	- accuracy: 0.7726	- val_loss: 0.4961	- val_accuracy: 0.7656
Epoch 78/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4612	- accuracy: 0.7760	- val_loss: 0.4958	- val_accuracy: 0.7604
Epoch 79/200	18/18 [=====]	- 0s 3ms/step	- loss: 0.4605	- accuracy: 0.7743	- val_loss: 0.4956	- val_accuracy: 0.7604
Epoch 80/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4598	- accuracy: 0.7726	- val_loss: 0.4954	- val_accuracy: 0.7656
Epoch 81/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4590	- accuracy: 0.7743	- val_loss: 0.4952	- val_accuracy: 0.7656
Epoch 82/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4584	- accuracy: 0.7760	- val_loss: 0.4950	- val_accuracy: 0.7604
Epoch 83/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4577	- accuracy: 0.7795	- val_loss: 0.4949	- val_accuracy: 0.7604
Epoch 84/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4571	- accuracy: 0.7795	- val_loss: 0.4947	- val_accuracy: 0.7604
Epoch 85/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4565	- accuracy: 0.7795	- val_loss: 0.4946	- val_accuracy: 0.7604
Epoch 86/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4559	- accuracy: 0.7795	- val_loss: 0.4945	- val_accuracy: 0.7604
Epoch 87/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4554	- accuracy: 0.7812	- val_loss: 0.4943	- val_accuracy: 0.7604
Epoch 88/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4547	- accuracy: 0.7812	- val_loss: 0.4942	- val_accuracy: 0.7604
Epoch 89/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4542	- accuracy: 0.7830	- val_loss: 0.4941	- val_accuracy: 0.7656
Epoch 90/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4537	- accuracy: 0.7830	- val_loss: 0.4940	- val_accuracy: 0.7656
Epoch 91/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4531	- accuracy: 0.7812	- val_loss: 0.4939	- val_accuracy: 0.7656
Epoch 92/200	18/18 [=====]	- 0s 5ms/step	- loss: 0.4526	- accuracy: 0.7865	- val_loss: 0.4939	- val_accuracy: 0.7656
Epoch 93/200	18/18 [=====]	- 0s 11ms/step	- loss: 0.4520	- accuracy: 0.7865	- val_loss: 0.4938	- val_accuracy: 0.7656
Epoch 94/200	18/18 [=====]	- 0s 7ms/step	- loss: 0.4516	- accuracy: 0.7917	- val_loss: 0.4937	- val_accuracy: 0.7656
Epoch 95/200	18/18 [=====]	- 0s 10ms/step	- loss: 0.4512	- accuracy: 0.7934	- val_loss: 0.4936	- val_accuracy: 0.7656
Epoch 96/200	18/18 [=====]	- 0s 9ms/step	- loss: 0.4507	- accuracy: 0.7917	- val_loss: 0.4936	- val_accuracy: 0.7656
Epoch 97/200	18/18 [=====]	- 0s 10ms/step	- loss: 0.4503	- accuracy: 0.7934	- val_loss: 0.4935	- val_accuracy: 0.7656
Epoch 98/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4497	- accuracy: 0.7951	- val_loss: 0.4935	- val_accuracy: 0.7656
Epoch 99/200	18/18 [=====]	- 0s 3ms/step	- loss: 0.4494	- accuracy: 0.7934	- val_loss: 0.4935	- val_accuracy: 0.7656
Epoch 100/200	18/18 [=====]	- 0s 3ms/step	- loss: 0.4491	- accuracy: 0.7934	- val_loss: 0.4935	- val_accuracy: 0.7656
Epoch 101/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4487	- accuracy: 0.7917	- val_loss: 0.4934	- val_accuracy: 0.7656
Epoch 102/200	18/18 [=====]	- 0s 4ms/step	- loss: 0.4485	- accuracy: 0.7917	- val_loss: 0.4935	- val_accuracy: 0.7656
Epoch 103/200	18/18 [=====]	- 0s 5ms/step	- loss: 0.4479	- accuracy: 0.7917	- val_loss: 0.4935	- val_accuracy: 0.7656
Epoch 104/200	18/18 [=====]	- 0s 6ms/step	- loss: 0.4476	- accuracy: 0.7917	- val_loss: 0.4935	- val_accuracy: 0.7604
Epoch 105/200	18/18 [=====]	- 0s 13ms/step	- loss: 0.4473	- accuracy: 0.7917	- val_loss: 0.4935	

[illegible]

```
Epoch 193/200
18/18 [=====] - 0s 5ms/step - loss: 0.4319 - accuracy: 0.7934 - val_loss: 0.5008 - val_accuracy: 0.7656
Epoch 194/200
18/18 [=====] - 0s 5ms/step - loss: 0.4318 - accuracy: 0.7917 - val_loss: 0.5009 - val_accuracy: 0.7656
Epoch 195/200
18/18 [=====] - 0s 5ms/step - loss: 0.4317 - accuracy: 0.7934 - val_loss: 0.5010 - val_accuracy: 0.7656
Epoch 196/200
18/18 [=====] - 0s 5ms/step - loss: 0.4316 - accuracy: 0.7917 - val_loss: 0.5010 - val_accuracy: 0.7656
Epoch 197/200
18/18 [=====] - 0s 5ms/step - loss: 0.4315 - accuracy: 0.7934 - val_loss: 0.5011 - val_accuracy: 0.7656
Epoch 198/200
18/18 [=====] - 0s 5ms/step - loss: 0.4315 - accuracy: 0.7917 - val_loss: 0.5010 - val_accuracy: 0.7708
Epoch 199/200
18/18 [=====] - 0s 6ms/step - loss: 0.4313 - accuracy: 0.7951 - val_loss: 0.5011 - val_accuracy: 0.7656
Epoch 200/200
18/18 [=====] - 0s 4ms/step - loss: 0.4312 - accuracy: 0.7951 - val_loss: 0.5012 - val_accuracy: 0.7656
```

```
In [ ]: y_pred_prob_nn_1 = model.predict(X_test_norm)
        y_pred_class_nn_1 = np.argmax(y_pred_prob_nn_1, axis=1)
```

```
6/6 [=====] - 0s 2ms/step
```

```
In [ ]: # Let's check out the outputs to get a feel for how keras apis work.
        y_pred_class_nn_1[:10]
```

```
Out[ ]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

```
In [ ]: y_pred_prob_nn_1[:10]
```

```
Out[ ]: array([[0.46349293],
               [0.6444795 ],
               [0.3868862 ],
               [0.16646963],
               [0.14696644],
               [0.5131753 ],
               [0.03617987],
               [0.29909146],
               [0.9291484 ],
               [0.13563998]], dtype=float32)
```

Create the plot\_roc function

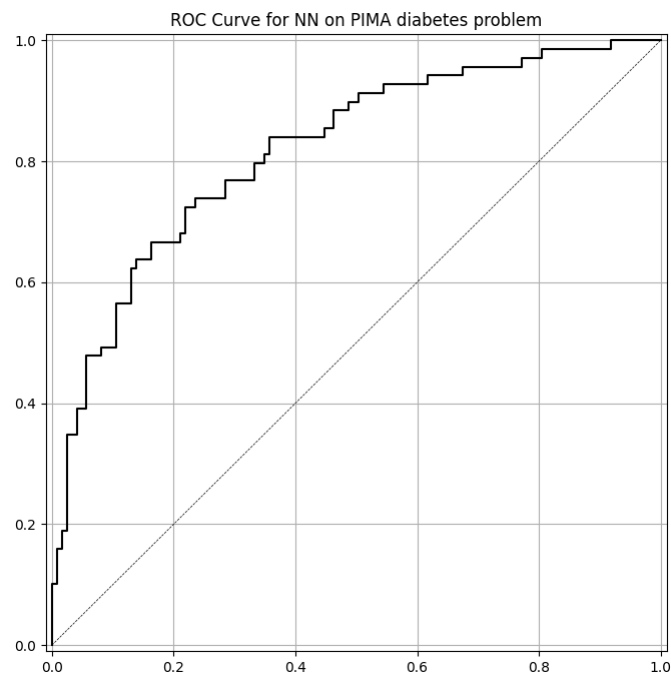
```
In [ ]: def plot_roc(y_test, y_pred, model_name):
        fpr, tpr, thr = roc_curve(y_test, y_pred)
        fig, ax = plt.subplots(figsize=(8, 8))
        ax.plot(fpr, tpr, 'k-')
        ax.plot([0, 1], [0, 1], 'k--', linewidth=.5) # roc curve for random model
        ax.grid(True)
        ax.set(title='ROC Curve for {} on PIMA diabetes problem'.format(model_name),
               xlim=[-0.01, 1.01], ylim=[-0.01, 1.01])
```

Evaluate the model performance and plot the ROC CURVE

```
In [ ]: print('accuracy is {:.3f}'.format(accuracy_score(y_test,y_pred_class_nn_1)))
        print('roc-auc is {:.3f}'.format(roc_auc_score(y_test,y_pred_prob_nn_1)))
```

```
plot_roc(y_test, y_pred_prob_nn_1, 'NN')
```

```
accuracy is 0.641
roc-auc is 0.819
```



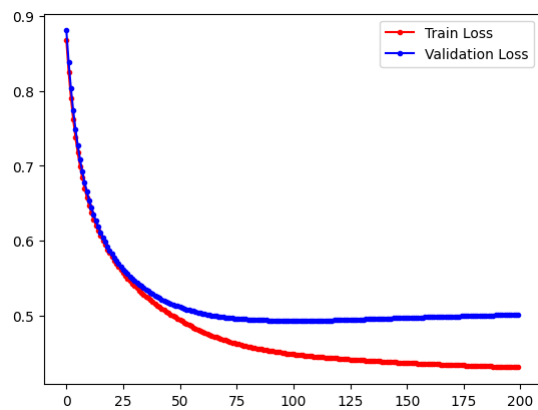
Plot the training loss and the validation loss over the different epochs and see how it looks

```
In [ ]: run_hist_1.history.keys()
```

```
Out[ ]: dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

```
In [ ]: fig, ax = plt.subplots()
        ax.plot(run_hist_1.history["loss"],'r', marker='.', label="Train Loss")
        ax.plot(run_hist_1.history["val_loss"],'b', marker='.', label="Validation Loss")
        ax.legend()
```

```
Out[ ]: <matplotlib.legend.Legend at 0x7cf7195ed4e0>
```



What is your interpretation about the result of the train and validation loss?

type your answer here

#### Supplementary Activity

- Build a model with two hidden layers, each with 6 nodes
- Use the "relu" activation function for the hidden layers, and "sigmoid" for the final layer
- Use a learning rate of .003 and train for 1500 epochs
- Graph the trajectory of the loss functions, accuracy on both train and test set
- Plot the roc curve for the predictions
- Use different learning rates, numbers of epochs, and network structures.
- Plot the results of training and validation loss using different learning rates, number of epochs and network structures
- Interpret your result

```
In [ ]: new_model = Sequential([
    Dense(6, input_shape=(8,), activation="relu"),
    Dense(6, input_shape=(8,), activation="relu"),
    Dense(1, activation="sigmoid")
])
```

```
In [ ]: new_model.compile(SGD(lr = .003), "binary_crossentropy", metrics=["accuracy"])
run_hist_2 = new_model.fit(X_train_norm, y_train, validation_data=(X_test_norm, y_test), epochs=1500)
```

WARNING: `abs1` is deprecated in Keras optimizer, please use `learning\_rate` or use the legacy optimizer, e.g., `tf.keras.optimizers.legacy.SGD`.

Epoch 1/1500  
18/18 [=====] - 1s 19ms/step - loss: 0.7377 - accuracy: 0.3837 - val\_loss: 0.7151 - val\_accuracy: 0.4531  
Epoch 2/1500  
18/18 [=====] - 0s 6ms/step - loss: 0.7180 - accuracy: 0.4462 - val\_loss: 0.6995 - val\_accuracy: 0.4844  
Epoch 3/1500  
18/18 [=====] - 0s 5ms/step - loss: 0.7024 - accuracy: 0.5208 - val\_loss: 0.6871 - val\_accuracy: 0.5208  
Epoch 4/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.6895 - accuracy: 0.5764 - val\_loss: 0.6768 - val\_accuracy: 0.5729  
Epoch 5/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.6788 - accuracy: 0.6215 - val\_loss: 0.6682 - val\_accuracy: 0.5885  
Epoch 6/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.6698 - accuracy: 0.6354 - val\_loss: 0.6611 - val\_accuracy: 0.6510  
Epoch 7/1500  
18/18 [=====] - 0s 5ms/step - loss: 0.6620 - accuracy: 0.6580 - val\_loss: 0.6549 - val\_accuracy: 0.6771  
Epoch 8/1500  
18/18 [=====] - 0s 5ms/step - loss: 0.6553 - accuracy: 0.6771 - val\_loss: 0.6496 - val\_accuracy: 0.7031  
Epoch 9/1500  
18/18 [=====] - 0s 5ms/step - loss: 0.6492 - accuracy: 0.6753 - val\_loss: 0.6448 - val\_accuracy: 0.7188  
Epoch 10/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.6438 - accuracy: 0.6806 - val\_loss: 0.6405 - val\_accuracy: 0.7083  
Epoch 11/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.6389 - accuracy: 0.6858 - val\_loss: 0.6366 - val\_accuracy: 0.7083  
Epoch 12/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.6343 - accuracy: 0.6840 - val\_loss: 0.6330 - val\_accuracy: 0.7083  
Epoch 13/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.6301 - accuracy: 0.6875 - val\_loss: 0.6296 - val\_accuracy: 0.7188  
Epoch 14/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.6262 - accuracy: 0.6875 - val\_loss: 0.6265 - val\_accuracy: 0.7135  
Epoch 15/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.6225 - accuracy: 0.6875 - val\_loss: 0.6235 - val\_accuracy: 0.7083  
Epoch 16/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.6190 - accuracy: 0.6858 - val\_loss: 0.6206 - val\_accuracy: 0.7031  
Epoch 17/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.6155 - accuracy: 0.6858 - val\_loss: 0.6179 - val\_accuracy: 0.7083  
Epoch 18/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.6123 - accuracy: 0.6858 - val\_loss: 0.6152 - val\_accuracy: 0.7031  
Epoch 19/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.6091 - accuracy: 0.6823 - val\_loss: 0.6127 - val\_accuracy: 0.7031  
Epoch 20/1500  
18/18 [=====] - 0s 5ms/step - loss: 0.6060 - accuracy: 0.6806 - val\_loss: 0.6101 - val\_accuracy: 0.7083  
Epoch 21/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.6029 - accuracy: 0.6788 - val\_loss: 0.6076 - val\_accuracy: 0.7083  
Epoch 22/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.6000 - accuracy: 0.6753 - val\_loss: 0.6051 - val\_accuracy: 0.7031  
Epoch 23/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5971 - accuracy: 0.6771 - val\_loss: 0.6027 - val\_accuracy: 0.7083  
Epoch 24/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5942 - accuracy: 0.6753 - val\_loss: 0.6003 - val\_accuracy: 0.7083  
Epoch 25/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5914 - accuracy: 0.6719 - val\_loss: 0.5979 - val\_accuracy: 0.7083  
Epoch 26/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5886 - accuracy: 0.6753 - val\_loss: 0.5955 - val\_accuracy: 0.7083  
Epoch 27/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5859 - accuracy: 0.6736 - val\_loss: 0.5931 - val\_accuracy: 0.6979  
Epoch 28/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5831 - accuracy: 0.6771 - val\_loss: 0.5907 - val\_accuracy: 0.6979  
Epoch 29/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5804 - accuracy: 0.6771 - val\_loss: 0.5884 - val\_accuracy: 0.6979  
Epoch 30/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5777 - accuracy: 0.6788 - val\_loss: 0.5861 - val\_accuracy: 0.7031  
Epoch 31/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5749 - accuracy: 0.6806 - val\_loss: 0.5837 - val\_accuracy: 0.7031  
Epoch 32/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5722 - accuracy: 0.6858 - val\_loss: 0.5814 - val\_accuracy: 0.7031  
Epoch 33/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5694 - accuracy: 0.6875 - val\_loss: 0.5790 - val\_accuracy: 0.7031  
Epoch 34/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5667 - accuracy: 0.6927 - val\_loss: 0.5767 - val\_accuracy: 0.7031  
Epoch 35/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5639 - accuracy: 0.6962 - val\_loss: 0.5743 - val\_accuracy: 0.6979  
Epoch 36/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5611 - accuracy: 0.7014 - val\_loss: 0.5719 - val\_accuracy: 0.6979  
Epoch 37/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5584 - accuracy: 0.6979 - val\_loss: 0.5696 - val\_accuracy: 0.6979  
Epoch 38/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5556 - accuracy: 0.6997 - val\_loss: 0.5673 - val\_accuracy: 0.7031  
Epoch 39/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5528 - accuracy: 0.6997 - val\_loss: 0.5650 - val\_accuracy: 0.7083  
Epoch 40/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5500 - accuracy: 0.7014 - val\_loss: 0.5628 - val\_accuracy: 0.7083  
Epoch 41/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5473 - accuracy: 0.6997 - val\_loss: 0.5606 - val\_accuracy: 0.7135  
Epoch 42/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5445 - accuracy: 0.7049 - val\_loss: 0.5585 - val\_accuracy: 0.7083  
Epoch 43/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5418 - accuracy: 0.7049 - val\_loss: 0.5564 - val\_accuracy: 0.7240  
Epoch 44/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5391 - accuracy: 0.7083 - val\_loss: 0.5543 - val\_accuracy: 0.7240  
Epoch 45/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5365 - accuracy: 0.7083 - val\_loss: 0.5523 - val\_accuracy: 0.7135  
Epoch 46/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5339 - accuracy: 0.7101 - val\_loss: 0.5503 - val\_accuracy: 0.7135  
Epoch 47/1500  
18/18 [=====] - 0s 5ms/step - loss: 0.5314 - accuracy: 0.7118 - val\_loss: 0.5484 - val\_accuracy: 0.7135  
Epoch 48/1500  
18/18 [=====] - 0s 5ms/step - loss: 0.5290 - accuracy: 0.7135 - val\_loss: 0.5466 - val\_accuracy: 0.7135  
Epoch 49/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5265 - accuracy: 0.7153 - val\_loss: 0.5448 - val\_accuracy: 0.7083  
Epoch 50/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5241 - accuracy: 0.7205 - val\_loss: 0.5431 - val\_accuracy: 0.7135  
Epoch 51/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5218 - accuracy: 0.7222 - val\_loss: 0.5414 - val\_accuracy: 0.7135  
Epoch 52/1500  
18/18 [=====] - 0s 5ms/step - loss: 0.5193 - accuracy: 0.7240 - val\_loss: 0.5398 - val\_accuracy: 0.7135  
Epoch 53/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5171 - accuracy: 0.7222 - val\_loss: 0.5383 - val\_accuracy: 0.7135  
Epoch 54/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5148 - accuracy: 0.7257 - val\_loss: 0.5368 - val\_accuracy: 0.7083  
Epoch 55/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5127 - accuracy: 0.7240 - val\_loss: 0.5354 - val\_accuracy: 0.7083  
Epoch 56/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5106 - accuracy: 0.7240 - val\_loss: 0.5340 - val\_accuracy: 0.7083  
Epoch 57/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5084 - accuracy: 0.7292 - val\_loss: 0.5327 - val\_accuracy: 0.7031  
Epoch 58/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5063 - accuracy: 0.7326 - val\_loss: 0.5315 - val\_accuracy: 0.7083  
Epoch 59/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5043 - accuracy: 0.7326 - val\_loss: 0.5303 - val\_accuracy: 0.7083  
Epoch 60/1500  
18/18 [=====] - 0s 3ms/step - loss: 0.5021 - accuracy: 0.7344 - val\_loss: 0.5291 - val\_accuracy: 0.7135  
Epoch 61/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.5001 - accuracy: 0.7344 - val\_loss: 0.5280 - val\_accuracy: 0.7188  
Epoch 62/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.4982 - accuracy: 0.7326 - val\_loss: 0.5269 - val\_accuracy: 0.7240  
Epoch 63/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.4963 - accuracy: 0.7326 - val\_loss: 0.5258 - val\_accuracy: 0.7240  
Epoch 64/1500  
18/18 [=====] - 0s 4ms/step - loss: 0.4943 - accuracy: 0.7361 - val\_loss: 0.5248 - val\_accuracy: 0.7188



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```

Epoch 1473/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3745 - accuracy: 0.8385 - val_loss: 0.5716 - val_accuracy: 0.7240
Epoch 1474/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3744 - accuracy: 0.8403 - val_loss: 0.5715 - val_accuracy: 0.7240
Epoch 1475/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3744 - accuracy: 0.8351 - val_loss: 0.5721 - val_accuracy: 0.7292
Epoch 1476/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3744 - accuracy: 0.8420 - val_loss: 0.5718 - val_accuracy: 0.7240
Epoch 1477/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3742 - accuracy: 0.8403 - val_loss: 0.5715 - val_accuracy: 0.7240
Epoch 1478/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3743 - accuracy: 0.8420 - val_loss: 0.5725 - val_accuracy: 0.7240
Epoch 1479/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3743 - accuracy: 0.8438 - val_loss: 0.5726 - val_accuracy: 0.7240
Epoch 1480/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3741 - accuracy: 0.8420 - val_loss: 0.5730 - val_accuracy: 0.7240
Epoch 1481/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3740 - accuracy: 0.8455 - val_loss: 0.5727 - val_accuracy: 0.7240
Epoch 1482/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3739 - accuracy: 0.8420 - val_loss: 0.5727 - val_accuracy: 0.7240
Epoch 1483/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3739 - accuracy: 0.8420 - val_loss: 0.5729 - val_accuracy: 0.7240
Epoch 1484/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3743 - accuracy: 0.8455 - val_loss: 0.5723 - val_accuracy: 0.7240
Epoch 1485/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3742 - accuracy: 0.8403 - val_loss: 0.5728 - val_accuracy: 0.7240
Epoch 1486/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3739 - accuracy: 0.8420 - val_loss: 0.5734 - val_accuracy: 0.7292
Epoch 1487/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3740 - accuracy: 0.8438 - val_loss: 0.5732 - val_accuracy: 0.7240
Epoch 1488/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3742 - accuracy: 0.8438 - val_loss: 0.5730 - val_accuracy: 0.7240
Epoch 1489/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3741 - accuracy: 0.8403 - val_loss: 0.5732 - val_accuracy: 0.7240
Epoch 1490/1500
18/18 [=====] - 0s 8ms/step - loss: 0.3741 - accuracy: 0.8368 - val_loss: 0.5740 - val_accuracy: 0.7292
Epoch 1491/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3739 - accuracy: 0.8420 - val_loss: 0.5736 - val_accuracy: 0.7292
Epoch 1492/1500
18/18 [=====] - 0s 8ms/step - loss: 0.3739 - accuracy: 0.8438 - val_loss: 0.5735 - val_accuracy: 0.7292
Epoch 1493/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3737 - accuracy: 0.8438 - val_loss: 0.5739 - val_accuracy: 0.7240
Epoch 1494/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3737 - accuracy: 0.8385 - val_loss: 0.5745 - val_accuracy: 0.7292
Epoch 1495/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3737 - accuracy: 0.8438 - val_loss: 0.5740 - val_accuracy: 0.7240
Epoch 1496/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3736 - accuracy: 0.8438 - val_loss: 0.5739 - val_accuracy: 0.7292
Epoch 1497/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3738 - accuracy: 0.8438 - val_loss: 0.5740 - val_accuracy: 0.7240
Epoch 1498/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3739 - accuracy: 0.8403 - val_loss: 0.5749 - val_accuracy: 0.7292
Epoch 1499/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3737 - accuracy: 0.8420 - val_loss: 0.5745 - val_accuracy: 0.7292
Epoch 1500/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3736 - accuracy: 0.8420 - val_loss: 0.5745 - val_accuracy: 0.7292

```

```

In [ ]: #y_pred_class_nn = new_model.predict(X_test_norm)
        #y_pred_prob_nn = np.argmax(y_pred_class_nn, axis=1)

```

```

In [ ]: y_pred_prob_nnew = new_model.predict(X_test_norm)
        y_pred_class_nnew = (y_pred_prob_nnew > 0.5).astype('int32')

6/6 [=====] - 0s 2ms/step

```

```

In [ ]: y_pred_class_nnew[:10]

```

```

Out[ ]: array([[1],
               [1],
               [0],
               [0],
               [0],
               [1],
               [0],
               [0],
               [1],
               [0]], dtype=int32)

```

```

In [ ]: y_pred_prob_nnew[:10]

```

```

Out[ ]: array([[0.84498954],
               [0.54498696],
               [0.37894908],
               [0.09194799],
               [0.14860062],
               [0.7145625 ],
               [0.00742441],
               [0.4904466 ],
               [0.85841143],
               [0.19457255]], dtype=float32)

```

```

In [ ]: def plot_roc(y_test, y_pred, new_model_name):
        fpr, tpr, thr = roc_curve(y_test, y_pred)
        fig, ax = plt.subplots(figsize=(8, 8))
        ax.plot(fpr, tpr, 'k-')
        ax.plot([0, 1], [0, 1], 'k--', linewidth=.5) # roc curve for random model
        ax.grid(True)
        ax.set(title='ROC Curve for {} on PIMA diabetes problem'.format(new_model_name),
               xlim=[-0.01, 1.01], ylim=[-0.01, 1.01])

```

```

In [ ]: print('accuracy is {:.3f}'.format(accuracy_score(y_test,y_pred_class_nnew)))#y_pred_prob_nnew
        print('roc-auc is {:.3f}'.format(roc_auc_score(y_test,y_pred_prob_nnew)))#y_pred_prob_nnew

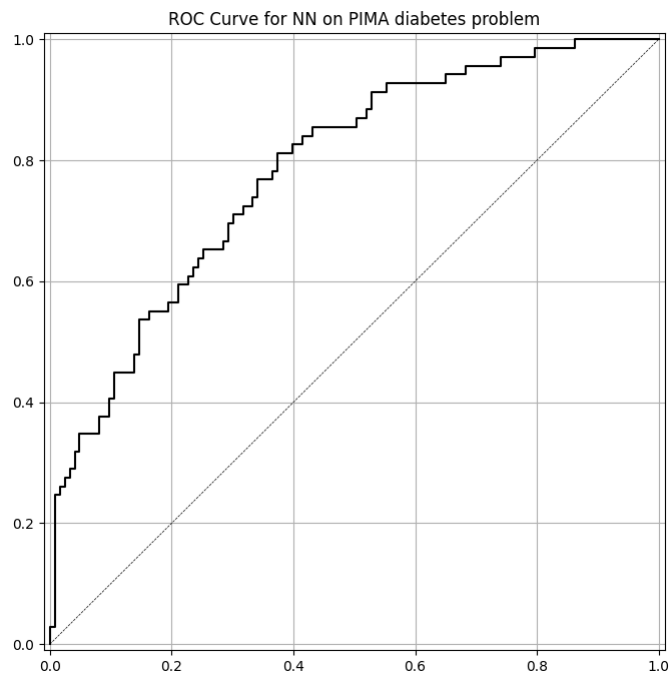
        plot_roc(y_test, y_pred_prob_nnew, 'NN')

```

```

accuracy is 0.729
roc-auc is 0.784

```



- This model shows worst result even though we've used more than 1 hidden layer, The model shows a moderate level of accuracy, and The ROC-AUC score of 0.801 suggests a satisfactory discrimination ability, indicating that the model performs reasonably well in distinguishing between positive and negative cases.

```
In [ ]: #print('accuracy is {:.3f}'.format(accuracy_score(y_test, y_pred_class_nn)))#y_pred_class_nn
        #print('roc_auc is {:.3f}'.format(roc_auc_score(y_test, y_pred_prob_nn)))#y_pred_prob_nn

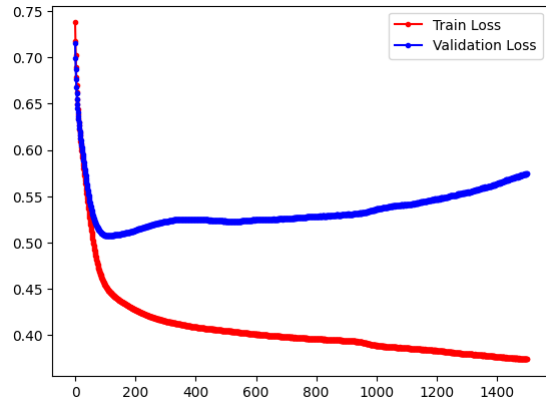
        #plot_roc(y_test, y_pred_prob_nn, 'NN')
```

```
In [ ]: run_hist_2.history.keys()
```

```
Out[ ]: dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

```
In [ ]: fig, ax = plt.subplots()
        ax.plot(run_hist_2.history["loss"], 'r', marker='.', label="Train Loss")
        ax.plot(run_hist_2.history["val_loss"], 'b', marker='.', label="Validation Loss")
        ax.legend()
```

```
Out[ ]: <matplotlib.legend.Legend at 0x7cf71b655600>
```



Learning rate = 0.001, epochs = 800,

```
In [ ]: new_model2 = Sequential([
        Dense(6, input_shape=(8,), activation="relu"),
        Dense(6, input_shape=(8,), activation="relu"),
        Dense(1, activation="sigmoid")
    ])
```

```
In [ ]: new_model2.compile(SGD(lr = .001), "binary_crossentropy", metrics=["accuracy"])
        run_hist_3 = new_model2.fit(X_train_norm, y_train, validation_data=(X_test_norm, y_test), epochs=800)
```

WARNING:absl:'lr' is deprecated in Keras optimizer, please use 'learning\_rate' or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.SGD.



Epoch 1/800  
18/18 [=====] - 1s 14ms/step - loss: 0.7897 - accuracy: 0.3715 - val\_loss: 0.7313 - val\_accuracy: 0.4323  
Epoch 2/800  
18/18 [=====] - 0s 4ms/step - loss: 0.7428 - accuracy: 0.4115 - val\_loss: 0.7004 - val\_accuracy: 0.5104  
Epoch 3/800  
18/18 [=====] - 0s 4ms/step - loss: 0.7134 - accuracy: 0.5208 - val\_loss: 0.6812 - val\_accuracy: 0.5833  
Epoch 4/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6950 - accuracy: 0.5938 - val\_loss: 0.6704 - val\_accuracy: 0.6250  
Epoch 5/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6829 - accuracy: 0.6372 - val\_loss: 0.6627 - val\_accuracy: 0.6510  
Epoch 6/800  
18/18 [=====] - 0s 5ms/step - loss: 0.6738 - accuracy: 0.6562 - val\_loss: 0.6565 - val\_accuracy: 0.6510  
Epoch 7/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6662 - accuracy: 0.6753 - val\_loss: 0.6510 - val\_accuracy: 0.6562  
Epoch 8/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6596 - accuracy: 0.6771 - val\_loss: 0.6458 - val\_accuracy: 0.6615  
Epoch 9/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6538 - accuracy: 0.6875 - val\_loss: 0.6411 - val\_accuracy: 0.6562  
Epoch 10/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6486 - accuracy: 0.6979 - val\_loss: 0.6367 - val\_accuracy: 0.6667  
Epoch 11/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6437 - accuracy: 0.7049 - val\_loss: 0.6324 - val\_accuracy: 0.6719  
Epoch 12/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6389 - accuracy: 0.7066 - val\_loss: 0.6284 - val\_accuracy: 0.6719  
Epoch 13/800  
18/18 [=====] - 0s 3ms/step - loss: 0.6343 - accuracy: 0.7101 - val\_loss: 0.6245 - val\_accuracy: 0.6823  
Epoch 14/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6301 - accuracy: 0.7118 - val\_loss: 0.6208 - val\_accuracy: 0.6927  
Epoch 15/800  
18/18 [=====] - 0s 3ms/step - loss: 0.6258 - accuracy: 0.7135 - val\_loss: 0.6171 - val\_accuracy: 0.6979  
Epoch 16/800  
18/18 [=====] - 0s 5ms/step - loss: 0.6216 - accuracy: 0.7153 - val\_loss: 0.6135 - val\_accuracy: 0.7031  
Epoch 17/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6176 - accuracy: 0.7188 - val\_loss: 0.6098 - val\_accuracy: 0.7031  
Epoch 18/800  
18/18 [=====] - 0s 3ms/step - loss: 0.6136 - accuracy: 0.7205 - val\_loss: 0.6062 - val\_accuracy: 0.7083  
Epoch 19/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6096 - accuracy: 0.7188 - val\_loss: 0.6026 - val\_accuracy: 0.7083  
Epoch 20/800  
18/18 [=====] - 0s 4ms/step - loss: 0.6057 - accuracy: 0.7170 - val\_loss: 0.5989 - val\_accuracy: 0.7188  
Epoch 21/800  
18/18 [=====] - 0s 3ms/step - loss: 0.6019 - accuracy: 0.7135 - val\_loss: 0.5952 - val\_accuracy: 0.7188  
Epoch 22/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5981 - accuracy: 0.7135 - val\_loss: 0.5915 - val\_accuracy: 0.7188  
Epoch 23/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5943 - accuracy: 0.7170 - val\_loss: 0.5878 - val\_accuracy: 0.7240  
Epoch 24/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5904 - accuracy: 0.7240 - val\_loss: 0.5841 - val\_accuracy: 0.7292  
Epoch 25/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5864 - accuracy: 0.7240 - val\_loss: 0.5803 - val\_accuracy: 0.7292  
Epoch 26/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5825 - accuracy: 0.7292 - val\_loss: 0.5765 - val\_accuracy: 0.7292  
Epoch 27/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5783 - accuracy: 0.7257 - val\_loss: 0.5727 - val\_accuracy: 0.7240  
Epoch 28/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5742 - accuracy: 0.7257 - val\_loss: 0.5689 - val\_accuracy: 0.7188  
Epoch 29/800  
18/18 [=====] - 0s 5ms/step - loss: 0.5699 - accuracy: 0.7326 - val\_loss: 0.5651 - val\_accuracy: 0.7240  
Epoch 30/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5659 - accuracy: 0.7326 - val\_loss: 0.5613 - val\_accuracy: 0.7292  
Epoch 31/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5617 - accuracy: 0.7396 - val\_loss: 0.5574 - val\_accuracy: 0.7344  
Epoch 32/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5576 - accuracy: 0.7396 - val\_loss: 0.5536 - val\_accuracy: 0.7344  
Epoch 33/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5535 - accuracy: 0.7448 - val\_loss: 0.5497 - val\_accuracy: 0.7396  
Epoch 34/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5494 - accuracy: 0.7465 - val\_loss: 0.5461 - val\_accuracy: 0.7396  
Epoch 35/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5457 - accuracy: 0.7483 - val\_loss: 0.5427 - val\_accuracy: 0.7448  
Epoch 36/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5419 - accuracy: 0.7465 - val\_loss: 0.5396 - val\_accuracy: 0.7448  
Epoch 37/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5383 - accuracy: 0.7483 - val\_loss: 0.5366 - val\_accuracy: 0.7448  
Epoch 38/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5350 - accuracy: 0.7483 - val\_loss: 0.5336 - val\_accuracy: 0.7448  
Epoch 39/800  
18/18 [=====] - 0s 3ms/step - loss: 0.5318 - accuracy: 0.7483 - val\_loss: 0.5309 - val\_accuracy: 0.7552  
Epoch 40/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5286 - accuracy: 0.7483 - val\_loss: 0.5283 - val\_accuracy: 0.7500  
Epoch 41/800  
18/18 [=====] - 0s 3ms/step - loss: 0.5257 - accuracy: 0.7483 - val\_loss: 0.5259 - val\_accuracy: 0.7500  
Epoch 42/800  
18/18 [=====] - 0s 5ms/step - loss: 0.5229 - accuracy: 0.7517 - val\_loss: 0.5237 - val\_accuracy: 0.7552  
Epoch 43/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5202 - accuracy: 0.7517 - val\_loss: 0.5216 - val\_accuracy: 0.7552  
Epoch 44/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5178 - accuracy: 0.7517 - val\_loss: 0.5196 - val\_accuracy: 0.7552  
Epoch 45/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5154 - accuracy: 0.7535 - val\_loss: 0.5177 - val\_accuracy: 0.7552  
Epoch 46/800  
18/18 [=====] - 0s 3ms/step - loss: 0.5130 - accuracy: 0.7535 - val\_loss: 0.5158 - val\_accuracy: 0.7552  
Epoch 47/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5109 - accuracy: 0.7535 - val\_loss: 0.5140 - val\_accuracy: 0.7604  
Epoch 48/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5087 - accuracy: 0.7500 - val\_loss: 0.5123 - val\_accuracy: 0.7604  
Epoch 49/800  
18/18 [=====] - 0s 4ms/step - loss: 0.5067 - accuracy: 0.7517 - val\_loss: 0.5106 - val\_accuracy: 0.7656  
Epoch 50/800  
18/18 [=====] - 0s 5ms/step - loss: 0.5047 - accuracy: 0.7535 - val\_loss: 0.5090 - val\_accuracy: 0.7656  
Epoch 51/800  
18/18 [=====] - 0s 3ms/step - loss: 0.5028 - accuracy: 0.7587 - val\_loss: 0.5074 - val\_accuracy: 0.7656  
Epoch 52/800  
18/18 [=====] - 0s 3ms/step - loss: 0.5008 - accuracy: 0.7587 - val\_loss: 0.5059 - val\_accuracy: 0.7708  
Epoch 53/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4990 - accuracy: 0.7622 - val\_loss: 0.5044 - val\_accuracy: 0.7708  
Epoch 54/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4973 - accuracy: 0.7622 - val\_loss: 0.5028 - val\_accuracy: 0.7604  
Epoch 55/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4957 - accuracy: 0.7622 - val\_loss: 0.5014 - val\_accuracy: 0.7604  
Epoch 56/800  
18/18 [=====] - 0s 3ms/step - loss: 0.4940 - accuracy: 0.7639 - val\_loss: 0.5000 - val\_accuracy: 0.7656  
Epoch 57/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4923 - accuracy: 0.7656 - val\_loss: 0.4986 - val\_accuracy: 0.7760  
Epoch 58/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4908 - accuracy: 0.7622 - val\_loss: 0.4973 - val\_accuracy: 0.7760  
Epoch 59/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4893 - accuracy: 0.7639 - val\_loss: 0.4960 - val\_accuracy: 0.7812  
Epoch 60/800  
18/18 [=====] - 0s 5ms/step - loss: 0.4879 - accuracy: 0.7604 - val\_loss: 0.4948 - val\_accuracy: 0.7760  
Epoch 61/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4865 - accuracy: 0.7639 - val\_loss: 0.4937 - val\_accuracy: 0.7812  
Epoch 62/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4852 - accuracy: 0.7674 - val\_loss: 0.4926 - val\_accuracy: 0.7812  
Epoch 63/800  
18/18 [=====] - 0s 3ms/step - loss: 0.4838 - accuracy: 0.7674 - val\_loss: 0.4916 - val\_accuracy: 0.7812  
Epoch 64/800  
18/18 [=====] - 0s 4ms/step - loss: 0.4826 - accuracy: 0.7674 - val\_loss: 0.4905 - val\_accuracy: 0.7865

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

```
Epoch 769/800
18/18 [=====] - 0s 4ms/step - loss: 0.4132 - accuracy: 0.7882 - val_loss: 0.5055 - val_accuracy: 0.7760
Epoch 770/800
18/18 [=====] - 0s 5ms/step - loss: 0.4130 - accuracy: 0.7934 - val_loss: 0.5054 - val_accuracy: 0.7760
Epoch 771/800
18/18 [=====] - 0s 4ms/step - loss: 0.4130 - accuracy: 0.7899 - val_loss: 0.5054 - val_accuracy: 0.7760
Epoch 772/800
18/18 [=====] - 0s 5ms/step - loss: 0.4129 - accuracy: 0.7934 - val_loss: 0.5055 - val_accuracy: 0.7760
Epoch 773/800
18/18 [=====] - 0s 4ms/step - loss: 0.4129 - accuracy: 0.7934 - val_loss: 0.5056 - val_accuracy: 0.7760
Epoch 774/800
18/18 [=====] - 0s 5ms/step - loss: 0.4130 - accuracy: 0.7917 - val_loss: 0.5056 - val_accuracy: 0.7760
Epoch 775/800
18/18 [=====] - 0s 4ms/step - loss: 0.4128 - accuracy: 0.7934 - val_loss: 0.5057 - val_accuracy: 0.7760
Epoch 776/800
18/18 [=====] - 0s 4ms/step - loss: 0.4128 - accuracy: 0.7917 - val_loss: 0.5058 - val_accuracy: 0.7760
Epoch 777/800
18/18 [=====] - 0s 4ms/step - loss: 0.4127 - accuracy: 0.7917 - val_loss: 0.5058 - val_accuracy: 0.7760
Epoch 778/800
18/18 [=====] - 0s 4ms/step - loss: 0.4126 - accuracy: 0.7951 - val_loss: 0.5059 - val_accuracy: 0.7760
Epoch 779/800
18/18 [=====] - 0s 5ms/step - loss: 0.4125 - accuracy: 0.7951 - val_loss: 0.5061 - val_accuracy: 0.7760
Epoch 780/800
18/18 [=====] - 0s 4ms/step - loss: 0.4126 - accuracy: 0.7917 - val_loss: 0.5062 - val_accuracy: 0.7760
Epoch 781/800
18/18 [=====] - 0s 5ms/step - loss: 0.4125 - accuracy: 0.7917 - val_loss: 0.5062 - val_accuracy: 0.7760
Epoch 782/800
18/18 [=====] - 0s 4ms/step - loss: 0.4125 - accuracy: 0.7917 - val_loss: 0.5062 - val_accuracy: 0.7760
Epoch 783/800
18/18 [=====] - 0s 5ms/step - loss: 0.4124 - accuracy: 0.7934 - val_loss: 0.5063 - val_accuracy: 0.7760
Epoch 784/800
18/18 [=====] - 0s 6ms/step - loss: 0.4123 - accuracy: 0.7934 - val_loss: 0.5063 - val_accuracy: 0.7760
Epoch 785/800
18/18 [=====] - 0s 4ms/step - loss: 0.4123 - accuracy: 0.7917 - val_loss: 0.5065 - val_accuracy: 0.7760
Epoch 786/800
18/18 [=====] - 0s 5ms/step - loss: 0.4123 - accuracy: 0.7899 - val_loss: 0.5065 - val_accuracy: 0.7760
Epoch 787/800
18/18 [=====] - 0s 5ms/step - loss: 0.4123 - accuracy: 0.7917 - val_loss: 0.5065 - val_accuracy: 0.7760
Epoch 788/800
18/18 [=====] - 0s 4ms/step - loss: 0.4122 - accuracy: 0.7917 - val_loss: 0.5064 - val_accuracy: 0.7760
Epoch 789/800
18/18 [=====] - 0s 5ms/step - loss: 0.4122 - accuracy: 0.7917 - val_loss: 0.5065 - val_accuracy: 0.7760
Epoch 790/800
18/18 [=====] - 0s 4ms/step - loss: 0.4120 - accuracy: 0.7934 - val_loss: 0.5068 - val_accuracy: 0.7760
Epoch 791/800
18/18 [=====] - 0s 5ms/step - loss: 0.4120 - accuracy: 0.7917 - val_loss: 0.5066 - val_accuracy: 0.7760
Epoch 792/800
18/18 [=====] - 0s 4ms/step - loss: 0.4119 - accuracy: 0.7934 - val_loss: 0.5067 - val_accuracy: 0.7708
Epoch 793/800
18/18 [=====] - 0s 4ms/step - loss: 0.4120 - accuracy: 0.7917 - val_loss: 0.5068 - val_accuracy: 0.7708
Epoch 794/800
18/18 [=====] - 0s 5ms/step - loss: 0.4120 - accuracy: 0.7917 - val_loss: 0.5070 - val_accuracy: 0.7708
Epoch 795/800
18/18 [=====] - 0s 5ms/step - loss: 0.4119 - accuracy: 0.7934 - val_loss: 0.5069 - val_accuracy: 0.7708
Epoch 796/800
18/18 [=====] - 0s 5ms/step - loss: 0.4118 - accuracy: 0.7917 - val_loss: 0.5071 - val_accuracy: 0.7708
Epoch 797/800
18/18 [=====] - 0s 4ms/step - loss: 0.4117 - accuracy: 0.7934 - val_loss: 0.5070 - val_accuracy: 0.7708
Epoch 798/800
18/18 [=====] - 0s 5ms/step - loss: 0.4118 - accuracy: 0.7899 - val_loss: 0.5071 - val_accuracy: 0.7708
Epoch 799/800
18/18 [=====] - 0s 4ms/step - loss: 0.4118 - accuracy: 0.7899 - val_loss: 0.5072 - val_accuracy: 0.7708
Epoch 800/800
18/18 [=====] - 0s 5ms/step - loss: 0.4117 - accuracy: 0.7917 - val_loss: 0.5073 - val_accuracy: 0.7708
```

```
In [ ]: y_pred_prob_nnew1 = new_model2.predict(X_test_norm)
        y_pred_class_nnew1 = (y_pred_prob_nnew1 > 0.5).astype('int32')
        6/6 [=====] - 0s 2ms/step
```

```
In [ ]: y_pred_class_nnew1[:10]
```

```
Out [ ]: array([[1],
              [1],
              [0],
              [0],
              [0],
              [1],
              [0],
              [0],
              [1],
              [0]], dtype=int32)
```

```
In [ ]: y_pred_prob_nnew1[:10]
```

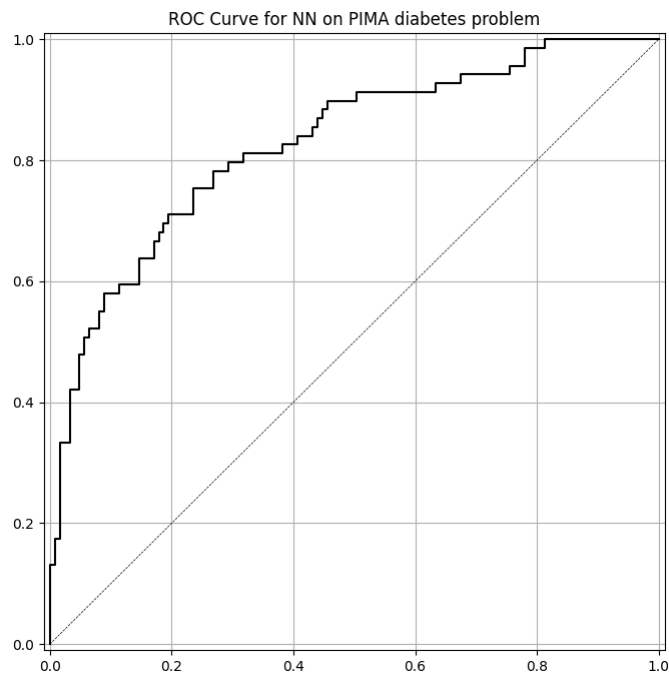
```
Out [ ]: array([[0.6447143 ],
                [0.762542  ],
                [0.3831105 ],
                [0.11248234],
                [0.24771403],
                [0.53396714],
                [0.01348655],
                [0.45988557],
                [0.887487  ],
                [0.1467733 ]], dtype=float32)
```

```
In [ ]: def plot_roc(y_test, y_pred, new_model_name1):
        fpr, tpr, thr = roc_curve(y_test, y_pred)
        fig, ax = plt.subplots(figsize=(8, 8))
        ax.plot(fpr, tpr, 'k-')
        ax.plot([0, 1], [0, 1], 'k--', linewidth=.5) # roc curve for random model
        ax.grid(True)
        ax.set(title='ROC Curve for {} on PIMA diabetes problem'.format(new_model_name1),
              xlim=[-0.01, 1.01], ylim=[-0.01, 1.01])
```

```
In [ ]: print('accuracy is {:.3f}'.format(accuracy_score(y_test, y_pred_class_nnew1)))#y_pred_prob_nnew
        print('roc_auc is {:.3f}'.format(roc_auc_score(y_test, y_pred_prob_nnew1)))#y_pred_prob_nnew

        plot_roc(y_test, y_pred_prob_nnew1, 'NN')

accuracy is 0.771
roc_auc is 0.827
```



- The accuracy of this model, which stands at 74.5%, indicates that the predictions it made were partially accurate. It received an ROC-AUC score of 0.803 in this regard. Higher scores indicate a better separation across classes. This evaluates the model's ability to distinguish between positive and negative examples. Overall, though, this model could be better.

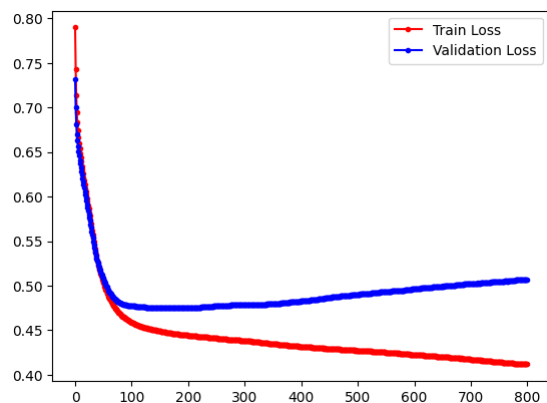
```
In [ ]: run_hist_3.history.keys()
```

```
Out[ ]: dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

```
In [ ]: fig, ax = plt.subplots()
```

```
ax.plot(run_hist_3.history["loss"], 'r', marker='.', label="Train Loss")
ax.plot(run_hist_3.history["val_loss"], 'b', marker='.', label="Validation Loss")
ax.legend()
```

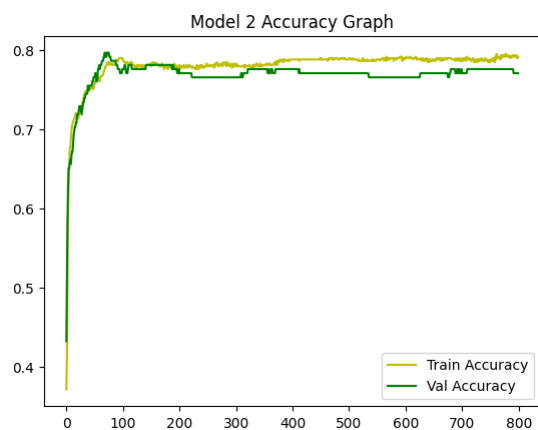
```
Out[ ]: <matplotlib.legend.Legend at 0x7cf71b44f550>
```



```
In [ ]: fig, ax = plt.subplots()
```

```
ax.plot(run_hist_3.history['accuracy'], 'y', label = 'Train Accuracy')
ax.plot(run_hist_3.history['val_accuracy'], 'g', label = 'Val Accuracy')
plt.title("Model 2 Accuracy Graph")
ax.legend()
```

```
Out[ ]: <matplotlib.legend.Legend at 0x7cf71b361450>
```



```
In [ ]: #y_pred_prob_nn_1 = model.predict(X_test_norm)
#y_pred_class_nn_1 = np.argmax(y_pred_prob_nn_1, axis=1)
```

model 3. 1 hidden layer, lr = 002, epoch = 1500

```
In [ ]: new_model3 = Sequential([
        Dense(6, input_shape=(8,)), activation="relu"),
        Dense(1, activation="sigmoid")
    ])
```

```
In [ ]: new_model3.compile(SGD(lr = .002), "binary_crossentropy", metrics=["accuracy"])
run_hist_4 = new_model3.fit(X_train_norm, y_train, validation_data=(X_test_norm, y_test), epochs=1500)
```

WARNING:absl:`lr` is deprecated in Keras optimizer, please use `learning\_rate` or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.SGD.

Epoch 1/1500	18/18 [=====]	- 1s 12ms/step - loss: 0.6218 - accuracy: 0.6962 - val_loss: 0.5952 - val_accuracy: 0.7135
Epoch 2/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.6130 - accuracy: 0.7066 - val_loss: 0.5894 - val_accuracy: 0.7188
Epoch 3/1500	18/18 [=====]	- 0s 5ms/step - loss: 0.6051 - accuracy: 0.7066 - val_loss: 0.5842 - val_accuracy: 0.7292
Epoch 4/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5977 - accuracy: 0.7135 - val_loss: 0.5796 - val_accuracy: 0.7344
Epoch 5/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5910 - accuracy: 0.7118 - val_loss: 0.5754 - val_accuracy: 0.7604
Epoch 6/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5849 - accuracy: 0.7118 - val_loss: 0.5716 - val_accuracy: 0.7604
Epoch 7/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5792 - accuracy: 0.7135 - val_loss: 0.5682 - val_accuracy: 0.7604
Epoch 8/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5742 - accuracy: 0.7188 - val_loss: 0.5652 - val_accuracy: 0.7656
Epoch 9/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5695 - accuracy: 0.7222 - val_loss: 0.5623 - val_accuracy: 0.7656
Epoch 10/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5651 - accuracy: 0.7240 - val_loss: 0.5597 - val_accuracy: 0.7604
Epoch 11/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5610 - accuracy: 0.7257 - val_loss: 0.5572 - val_accuracy: 0.7656
Epoch 12/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5571 - accuracy: 0.7274 - val_loss: 0.5548 - val_accuracy: 0.7708
Epoch 13/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5533 - accuracy: 0.7292 - val_loss: 0.5525 - val_accuracy: 0.7656
Epoch 14/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5498 - accuracy: 0.7309 - val_loss: 0.5504 - val_accuracy: 0.7656
Epoch 15/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5466 - accuracy: 0.7326 - val_loss: 0.5485 - val_accuracy: 0.7656
Epoch 16/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5436 - accuracy: 0.7361 - val_loss: 0.5467 - val_accuracy: 0.7604
Epoch 17/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5407 - accuracy: 0.7344 - val_loss: 0.5451 - val_accuracy: 0.7656
Epoch 18/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5379 - accuracy: 0.7378 - val_loss: 0.5435 - val_accuracy: 0.7656
Epoch 19/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5353 - accuracy: 0.7413 - val_loss: 0.5421 - val_accuracy: 0.7656
Epoch 20/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5329 - accuracy: 0.7396 - val_loss: 0.5407 - val_accuracy: 0.7656
Epoch 21/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5305 - accuracy: 0.7378 - val_loss: 0.5394 - val_accuracy: 0.7604
Epoch 22/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5283 - accuracy: 0.7396 - val_loss: 0.5381 - val_accuracy: 0.7656
Epoch 23/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5262 - accuracy: 0.7361 - val_loss: 0.5370 - val_accuracy: 0.7656
Epoch 24/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5243 - accuracy: 0.7378 - val_loss: 0.5359 - val_accuracy: 0.7708
Epoch 25/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5224 - accuracy: 0.7361 - val_loss: 0.5348 - val_accuracy: 0.7760
Epoch 26/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5204 - accuracy: 0.7396 - val_loss: 0.5339 - val_accuracy: 0.7760
Epoch 27/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5186 - accuracy: 0.7431 - val_loss: 0.5329 - val_accuracy: 0.7760
Epoch 28/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5168 - accuracy: 0.7431 - val_loss: 0.5320 - val_accuracy: 0.7708
Epoch 29/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5151 - accuracy: 0.7483 - val_loss: 0.5311 - val_accuracy: 0.7708
Epoch 30/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5135 - accuracy: 0.7517 - val_loss: 0.5302 - val_accuracy: 0.7708
Epoch 31/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5120 - accuracy: 0.7500 - val_loss: 0.5294 - val_accuracy: 0.7708
Epoch 32/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5104 - accuracy: 0.7552 - val_loss: 0.5287 - val_accuracy: 0.7708
Epoch 33/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5090 - accuracy: 0.7552 - val_loss: 0.5279 - val_accuracy: 0.7656
Epoch 34/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5077 - accuracy: 0.7569 - val_loss: 0.5272 - val_accuracy: 0.7604
Epoch 35/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5063 - accuracy: 0.7604 - val_loss: 0.5264 - val_accuracy: 0.7656
Epoch 36/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5050 - accuracy: 0.7674 - val_loss: 0.5257 - val_accuracy: 0.7656
Epoch 37/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5037 - accuracy: 0.7656 - val_loss: 0.5249 - val_accuracy: 0.7656
Epoch 38/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5025 - accuracy: 0.7674 - val_loss: 0.5242 - val_accuracy: 0.7656
Epoch 39/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.5012 - accuracy: 0.7691 - val_loss: 0.5235 - val_accuracy: 0.7656
Epoch 40/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.5000 - accuracy: 0.7708 - val_loss: 0.5228 - val_accuracy: 0.7708
Epoch 41/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.4988 - accuracy: 0.7726 - val_loss: 0.5221 - val_accuracy: 0.7708
Epoch 42/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.4976 - accuracy: 0.7760 - val_loss: 0.5214 - val_accuracy: 0.7708
Epoch 43/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.4965 - accuracy: 0.7760 - val_loss: 0.5208 - val_accuracy: 0.7708
Epoch 44/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.4955 - accuracy: 0.7778 - val_loss: 0.5201 - val_accuracy: 0.7708
Epoch 45/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.4944 - accuracy: 0.7760 - val_loss: 0.5195 - val_accuracy: 0.7708
Epoch 46/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.4934 - accuracy: 0.7778 - val_loss: 0.5188 - val_accuracy: 0.7708
Epoch 47/1500	18/18 [=====]	- 0s 4ms/step - loss: 0.4924 - accuracy: 0.7760 - val_loss: 0.5182 - val_accuracy: 0.7708
Epoch 48/1500	18/18 [=====]	- 0s 3ms/step - loss: 0.4913 - accuracy: 0.7760 - val_loss: 0.5176 - val_accuracy: 0.7708
Epoch 49/1500	18/1	



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```

Epoch 1473/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3932 - accuracy: 0.8229 - val_loss: 0.5383 - val_accuracy: 0.7604
Epoch 1474/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3932 - accuracy: 0.8229 - val_loss: 0.5384 - val_accuracy: 0.7604
Epoch 1475/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3931 - accuracy: 0.8212 - val_loss: 0.5382 - val_accuracy: 0.7604
Epoch 1476/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5383 - val_accuracy: 0.7604
Epoch 1477/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5383 - val_accuracy: 0.7604
Epoch 1478/1500
18/18 [=====] - 0s 8ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5383 - val_accuracy: 0.7604
Epoch 1479/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3931 - accuracy: 0.8212 - val_loss: 0.5383 - val_accuracy: 0.7604
Epoch 1480/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5384 - val_accuracy: 0.7604
Epoch 1481/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3931 - accuracy: 0.8212 - val_loss: 0.5383 - val_accuracy: 0.7604
Epoch 1482/1500
18/18 [=====] - 0s 8ms/step - loss: 0.3930 - accuracy: 0.8229 - val_loss: 0.5384 - val_accuracy: 0.7604
Epoch 1483/1500
18/18 [=====] - 0s 8ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5385 - val_accuracy: 0.7604
Epoch 1484/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5383 - val_accuracy: 0.7604
Epoch 1485/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3930 - accuracy: 0.8229 - val_loss: 0.5384 - val_accuracy: 0.7604
Epoch 1486/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5385 - val_accuracy: 0.7604
Epoch 1487/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3930 - accuracy: 0.8229 - val_loss: 0.5384 - val_accuracy: 0.7604
Epoch 1488/1500
18/18 [=====] - 0s 5ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5385 - val_accuracy: 0.7604
Epoch 1489/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3931 - accuracy: 0.8247 - val_loss: 0.5385 - val_accuracy: 0.7604
Epoch 1490/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3930 - accuracy: 0.8247 - val_loss: 0.5386 - val_accuracy: 0.7604
Epoch 1491/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3931 - accuracy: 0.8229 - val_loss: 0.5387 - val_accuracy: 0.7604
Epoch 1492/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3931 - accuracy: 0.8264 - val_loss: 0.5387 - val_accuracy: 0.7604
Epoch 1493/1500
18/18 [=====] - 0s 8ms/step - loss: 0.3932 - accuracy: 0.8247 - val_loss: 0.5387 - val_accuracy: 0.7604
Epoch 1494/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3930 - accuracy: 0.8229 - val_loss: 0.5385 - val_accuracy: 0.7604
Epoch 1495/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3930 - accuracy: 0.8264 - val_loss: 0.5386 - val_accuracy: 0.7604
Epoch 1496/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3931 - accuracy: 0.8247 - val_loss: 0.5387 - val_accuracy: 0.7604
Epoch 1497/1500
18/18 [=====] - 0s 9ms/step - loss: 0.3930 - accuracy: 0.8229 - val_loss: 0.5387 - val_accuracy: 0.7604
Epoch 1498/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3930 - accuracy: 0.8247 - val_loss: 0.5386 - val_accuracy: 0.7604
Epoch 1499/1500
18/18 [=====] - 0s 7ms/step - loss: 0.3930 - accuracy: 0.8264 - val_loss: 0.5386 - val_accuracy: 0.7604
Epoch 1500/1500
18/18 [=====] - 0s 6ms/step - loss: 0.3929 - accuracy: 0.8264 - val_loss: 0.5387 - val_accuracy: 0.7604

```

```

In [ ]: y_pred_prob_nnew2 = new_model2.predict(X_test_norm)
        y_pred_class_nnew2 = (y_pred_prob_nnew2 > 0.5).astype('int32')

6/6 [=====] - 0s 2ms/step

```

```

In [ ]: y_pred_class_nnew2[:10]

```

```

Out[ ]: array([[1],
              [1],
              [0],
              [0],
              [0],
              [1],
              [0],
              [0],
              [1],
              [0]], dtype=int32)

```

```

In [ ]: y_pred_prob_nnew2[:10]

```

```

Out[ ]: array([[0.6447143 ],
              [0.762542  ],
              [0.3831105 ],
              [0.11248234],
              [0.24771403],
              [0.53396714],
              [0.01348655],
              [0.45988557],
              [0.887487  ],
              [0.1467733 ]], dtype=float32)

```

```

In [ ]: def plot_roc(y_test, y_pred, new_model_name):
        fpr, tpr, thr = roc_curve(y_test, y_pred)
        fig, ax = plt.subplots(figsize=(8, 8))
        ax.plot(fpr, tpr, 'k-')
        ax.plot([0, 1], [0, 1], 'k--', linewidth=.5) # roc curve for random model
        ax.grid(True)
        ax.set(title='ROC Curve for {} on PIMA diabetes problem'.format(new_model_name),
              xlim=[-0.01, 1.01], ylim=[-0.01, 1.01])

```

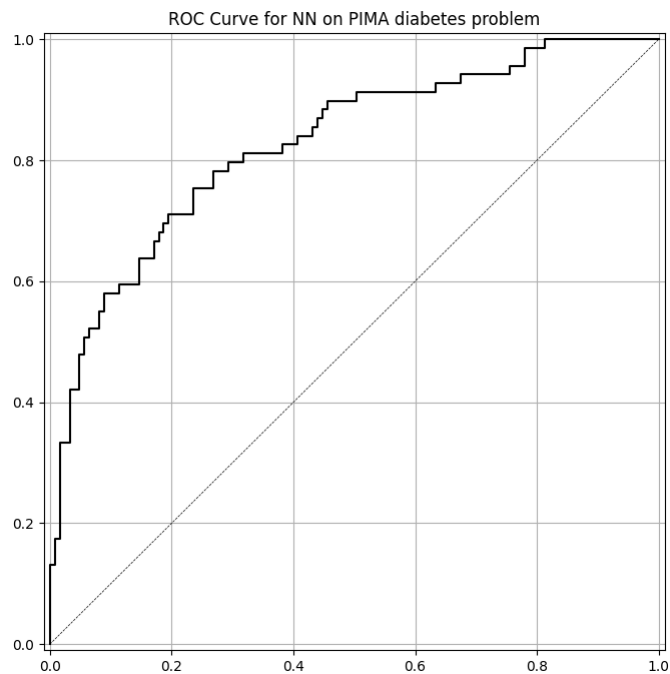
```

In [ ]: print('accuracy is {:.3f}'.format(accuracy_score(y_test,y_pred_class_nnew2)))#y_pred_prob_nnew
        print('roc_auc is {:.3f}'.format(roc_auc_score(y_test,y_pred_prob_nnew2)))#y_pred_prob_nnew

        plot_roc(y_test, y_pred_prob_nnew2, 'NN')

accuracy is 0.771
roc_auc is 0.827

```

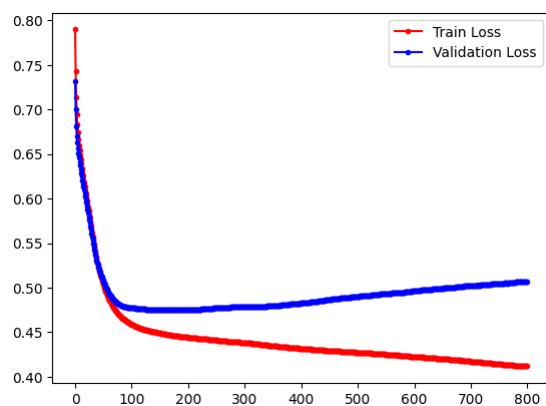


- This model shows the accuracy, standing at 74.5%, signifies that the model's predictions were correct at some point. In terms of ROC-AUC, it achieved a score of 0.803. This assesses the model's capability to distinguish between positive and negative instances, with higher values suggesting a superior separation between classes. But overall, not a good model.

```
In [ ]: run_hist_3.history.keys()
Out[ ]: dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

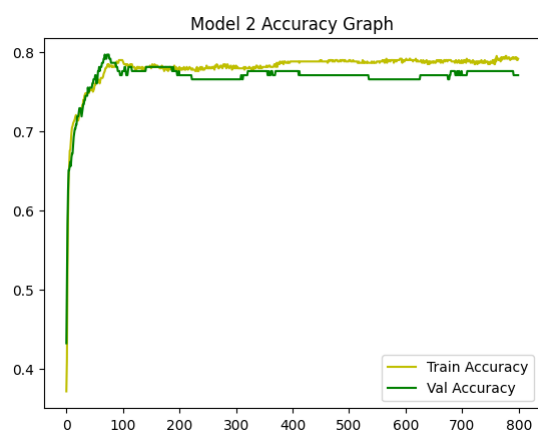
In [ ]: fig, ax = plt.subplots()
ax.plot(run_hist_3.history["loss"], 'r', marker='.', label="Train Loss")
ax.plot(run_hist_3.history["val_loss"], 'b', marker='.', label="Validation Loss")
ax.legend()
```

Out[ ]: <matplotlib.legend.Legend at 0x7cf71b2d7310>



```
In [ ]: fig, ax = plt.subplots()
ax.plot(run_hist_3.history['accuracy'], 'y', label = 'Train Accuracy')
ax.plot(run_hist_3.history['val_accuracy'], 'g', label = 'Val Accuracy')
plt.title("Model 2 Accuracy Graph")
ax.legend()
```

Out[ ]: <matplotlib.legend.Legend at 0x7cf71b02ded0>



Using Byukilmaz.csv



```
In [74]: from __future__ import absolute_import, division, print_function

import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, precision_recall_curve, roc_auc_score, roc_curve, accuracy_score
from sklearn.ensemble import RandomForestClassifier

import seaborn as sns
%matplotlib inline
```

```
In [75]: ## Import Keras objects for Deep Learning

from keras.models import Sequential
from keras.layers import Input, Dense, Flatten, Dropout, BatchNormalization
from keras.optimizers import Adam, SGD, RMSprop
```

```
In [76]: byu_df = pd.read_csv('/content/drive/SharedDrives/CPE312/HOAS/buyukyilmaz.csv')
byu_df
```

```
Out[76]:
```

	meanfreq	sd	median	Q25	Q75	IQR	skew	kurt	sp.ent	sfm	...	centroid	meanfun	minfun	maxfun	meandom	mindom	maxdom	dfrange	modindx	gender
0	0.059781	0.064241	0.032027	0.015071	0.090193	0.075122	12.863462	274.402906	0.893369	0.491918	...	0.059781	0.084279	0.015702	0.275862	0.007812	0.007812	0.007812	0.000000	0.000000	male
1	0.066009	0.067310	0.040229	0.019414	0.092666	0.073252	22.423285	634.613855	0.892193	0.513724	...	0.066009	0.107937	0.015826	0.250000	0.009014	0.007812	0.054688	0.046875	0.052632	male
2	0.077316	0.083829	0.036718	0.008701	0.131908	0.123207	30.757155	1024.927705	0.846389	0.478905	...	0.077316	0.098706	0.015656	0.271186	0.007990	0.007812	0.015625	0.007812	0.046512	male
3	0.151228	0.072111	0.158011	0.096582	0.207955	0.111374	1.232831	4.177296	0.963322	0.727232	...	0.151228	0.088965	0.017798	0.250000	0.201497	0.007812	0.562500	0.554688	0.247119	male
4	0.135120	0.079146	0.124656	0.078720	0.206045	0.127325	1.101174	4.333713	0.971955	0.783568	...	0.135120	0.106398	0.016931	0.266667	0.712812	0.007812	5.484375	5.476562	0.208274	male
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3163	0.131884	0.084734	0.153707	0.049285	0.201144	0.151859	1.762129	6.630383	0.962934	0.763182	...	0.131884	0.182790	0.083770	0.262295	0.832899	0.007812	4.210938	4.203125	0.161929	female
3164	0.116221	0.089221	0.076758	0.042718	0.204911	0.162193	0.693730	2.503954	0.960716	0.709570	...	0.116221	0.188980	0.034409	0.275862	0.909856	0.039062	3.679688	3.640625	0.277897	female
3165	0.142056	0.095798	0.183731	0.033424	0.224360	0.190936	1.876502	6.604509	0.946854	0.654196	...	0.142056	0.209918	0.039506	0.275862	0.494271	0.007812	2.937500	2.929688	0.194759	female
3166	0.143659	0.090628	0.184976	0.043508	0.219943	0.176435	1.591065	5.388298	0.950436	0.675470	...	0.143659	0.172375	0.034483	0.250000	0.791360	0.007812	3.593750	3.585938	0.311002	female
3167	0.165509	0.092884	0.183044	0.070072	0.250827	0.180756	1.705029	5.769115	0.938829	0.601529	...	0.165509	0.185607	0.062257	0.271186	0.227022	0.007812	0.554688	0.546875	0.350000	female

3168 rows x 21 columns

```
In [77]: print('Data Shape: ', byu_df.shape, '\n\n')
print('Data Types:\n', byu_df.dtypes)

Data Shape: (3168, 21)

Data Types:
meanfreq    float64
sd          float64
median      float64
Q25         float64
Q75         float64
IQR         float64
skew        float64
kurt        float64
sp.ent      float64
sfm         float64
mode        float64
centroid    float64
meanfun     float64
minfun      float64
maxfun      float64
meandom     float64
mindom      float64
maxdom      float64
dfrange     float64
modindx     float64
gender      object
dtype: object
```

```
In [78]: from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
byu_df.gender = le.fit_transform(byu_df.gender)
```

```
In [79]: X = byu_df.drop('gender', axis = 1).values
y = byu_df.gender.values
```

```
In [80]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=11111)
```

```
In [81]: sc = StandardScaler()
X_train_norm = sc.fit_transform(X_train)
X_test_norm = sc.fit_transform(X_test)
```

```
In [84]: model_supp = Sequential([
    Dense(6, input_shape = (20, ), activation = 'relu'),
    Dense(1, activation = 'sigmoid')
])
```

```
In [105]: model_supp.compile(SGD(lr = 0.003), "binary_crossentropy", metrics = ['accuracy'])

run_hist_supp = model_supp.fit(X_train_norm, y_train, validation_data = (X_test_norm, y_test), epochs = 1500)

WARNING:absl:lr is deprecated in Keras optimizer, please use `learning_rate` or use the legacy optimizer, e.g., tf.keras.optimizers.legacy.SGD.
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



```

Epoch 1473/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0384 - accuracy: 0.9874 - val_loss: 0.0703 - val_accuracy: 0.9798
Epoch 1474/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0385 - accuracy: 0.9874 - val_loss: 0.0700 - val_accuracy: 0.9798
Epoch 1475/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0384 - accuracy: 0.9870 - val_loss: 0.0700 - val_accuracy: 0.9798
Epoch 1476/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0384 - accuracy: 0.9874 - val_loss: 0.0700 - val_accuracy: 0.9798
Epoch 1477/1500
75/75 [=====] - 0s 3ms/step - loss: 0.0384 - accuracy: 0.9874 - val_loss: 0.0699 - val_accuracy: 0.9798
Epoch 1478/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0384 - accuracy: 0.9874 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1479/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0384 - accuracy: 0.9874 - val_loss: 0.0700 - val_accuracy: 0.9798
Epoch 1480/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0384 - accuracy: 0.9874 - val_loss: 0.0699 - val_accuracy: 0.9798
Epoch 1481/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0703 - val_accuracy: 0.9798
Epoch 1482/1500
75/75 [=====] - 0s 5ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1483/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0698 - val_accuracy: 0.9798
Epoch 1484/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0385 - accuracy: 0.9874 - val_loss: 0.0699 - val_accuracy: 0.9798
Epoch 1485/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1486/1500
75/75 [=====] - 0s 3ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0703 - val_accuracy: 0.9798
Epoch 1487/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1488/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0703 - val_accuracy: 0.9798
Epoch 1489/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1490/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0700 - val_accuracy: 0.9798
Epoch 1491/1500
75/75 [=====] - 0s 5ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1492/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0382 - accuracy: 0.9874 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1493/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1494/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0700 - val_accuracy: 0.9798
Epoch 1495/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0382 - accuracy: 0.9874 - val_loss: 0.0706 - val_accuracy: 0.9798
Epoch 1496/1500
75/75 [=====] - 0s 3ms/step - loss: 0.0384 - accuracy: 0.9870 - val_loss: 0.0701 - val_accuracy: 0.9798
Epoch 1497/1500
75/75 [=====] - 0s 3ms/step - loss: 0.0383 - accuracy: 0.9870 - val_loss: 0.0700 - val_accuracy: 0.9811
Epoch 1498/1500
75/75 [=====] - 0s 3ms/step - loss: 0.0383 - accuracy: 0.9874 - val_loss: 0.0703 - val_accuracy: 0.9798
Epoch 1499/1500
75/75 [=====] - 0s 4ms/step - loss: 0.0382 - accuracy: 0.9874 - val_loss: 0.0702 - val_accuracy: 0.9798
Epoch 1500/1500
75/75 [=====] - 0s 3ms/step - loss: 0.0382 - accuracy: 0.9874 - val_loss: 0.0702 - val_accuracy: 0.9798

```

```

In [112]: y_pred_prob_nn_supp = model_supp.predict(X_test_norm)
y_pred_class_nn_supp = (y_pred_prob_nn_supp > 0.5).astype('int32')
25/25 [=====] - 0s 1ms/step

```

```

In [113]: y_pred_class_nn_supp[:10]

```

```

Out[113]: array([[1],
 [1],
 [1],
 [1],
 [1],
 [1],
 [1],
 [1],
 [0],
 [0]], dtype=int32)

```

```

In [114]: y_pred_prob_nn_supp[:10]

```

```

Out[114]: array([[9.9989748e-01],
 [9.9985604e-01],
 [9.9817389e-01],
 [9.9999958e-01],
 [9.9989897e-01],
 [9.9819648e-01],
 [9.9983877e-01],
 [9.8757589e-01],
 [5.3232161e-05],
 [3.3626668e-05]], dtype=float32)

```

```

In [115]: run_hist_supp.history.keys()

```

```

Out[115]: dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

```

```

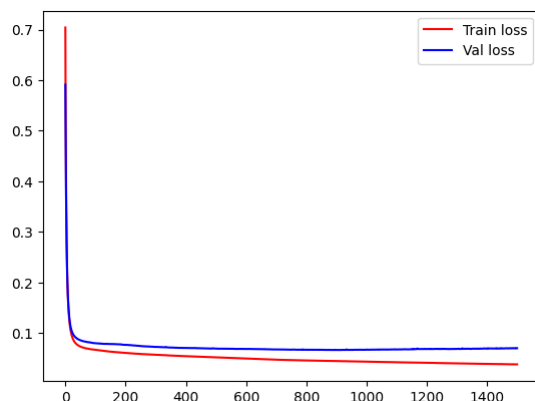
In [116]: fig, ax = plt.subplots()
ax.plot(run_hist_supp.history['loss'], 'r', label = 'Train loss')
ax.plot(run_hist_supp.history['val_loss'], 'b', label = "Val loss")
ax.legend()

```

```

Out[116]: <matplotlib.legend.Legend at 0x7b29bddc6a10>

```



```

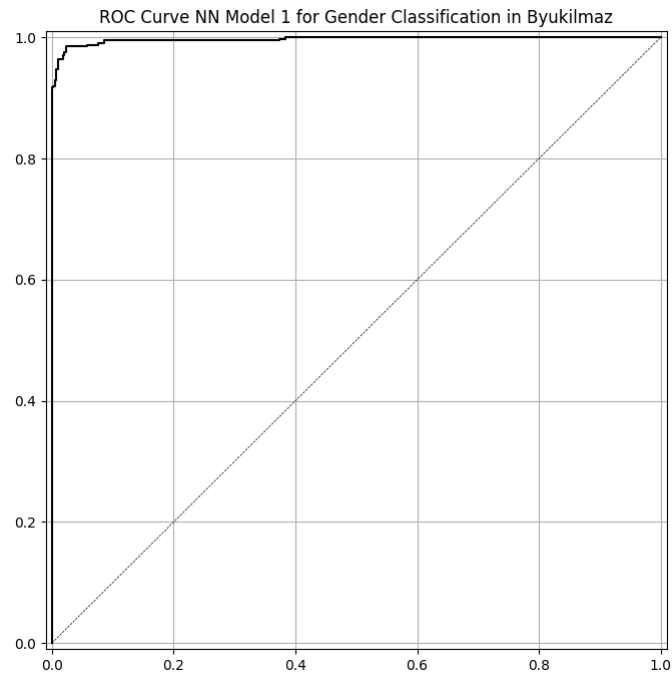
In [117]: def plot_roc(y_test, y_pred, model_name):
fpr, tpr, thr = roc_curve(y_test, y_pred)
fig, ax = plt.subplots(figsize=(8, 8))
ax.plot(fpr, tpr, 'k-')

```

```
ax.plot([0, 1], [0, 1], 'k--', linewidth=.5) # roc curve for random model
ax.grid(True)
ax.set(title='ROC Curve {} for Gender Classification in Byukilmaz'.format(model_name),
       xlim=[-0.01, 1.01], ylim=[-0.01, 1.01])
```

```
In [118.: print('accuracy is {:.3f}'.format(accuracy_score(y_test,y_pred_class_nn_supp)))
print('roc-auc is {:.3f}'.format(roc_auc_score(y_test,y_pred_prob_nn_supp)))
plot_roc(y_test, y_pred_prob_nn_supp, "NN Model 1")
```

```
accuracy is 0.980
roc-auc is 0.997
```



- The model shows above has a great result by having a high accuracy and roc-auc result. After testing it out with 1 hidden layer and 1500 epoch and .003 learning rate.

## 2hidden layer, 200 epoch, .003 learning rate

```
In [103.: model_sup2 = Sequential([
Dense(6, input_shape = (20, ), activation = 'relu'),
Dense(6, input_shape = (20, ), activation = 'relu'),
Dense(1, activation = 'sigmoid')
])
```

```
In [104.: model_sup2.compile(SGD(lr = 0.003), "binary_crossentropy", metrics = ['accuracy'])
run_hist_sup2 = model_sup2.fit(X_train_norm, y_train, validation_data = (X_test_norm, y_test), epochs = 200)
```

WARNING:absl:lr is deprecated in Keras optimizer, please use 'learning\_rate' or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.SGD.

Epoch 1/200  
75/75 [=====] - 1s 5ms/step - loss: 0.6613 - accuracy: 0.7189 - val\_loss: 0.6347 - val\_accuracy: 0.7386  
Epoch 2/200  
75/75 [=====] - 0s 2ms/step - loss: 0.6072 - accuracy: 0.7795 - val\_loss: 0.5861 - val\_accuracy: 0.8005  
Epoch 3/200  
75/75 [=====] - 0s 2ms/step - loss: 0.5591 - accuracy: 0.8215 - val\_loss: 0.5387 - val\_accuracy: 0.8295  
Epoch 4/200  
75/75 [=====] - 0s 2ms/step - loss: 0.5108 - accuracy: 0.8413 - val\_loss: 0.4904 - val\_accuracy: 0.8535  
Epoch 5/200  
75/75 [=====] - 0s 3ms/step - loss: 0.4629 - accuracy: 0.8565 - val\_loss: 0.4428 - val\_accuracy: 0.8674  
Epoch 6/200  
75/75 [=====] - 0s 3ms/step - loss: 0.4172 - accuracy: 0.8708 - val\_loss: 0.3970 - val\_accuracy: 0.8813  
Epoch 7/200  
75/75 [=====] - 0s 3ms/step - loss: 0.3743 - accuracy: 0.8796 - val\_loss: 0.3544 - val\_accuracy: 0.8965  
Epoch 8/200  
75/75 [=====] - 0s 3ms/step - loss: 0.3352 - accuracy: 0.8893 - val\_loss: 0.3143 - val\_accuracy: 0.9028  
Epoch 9/200  
75/75 [=====] - 0s 3ms/step - loss: 0.2988 - accuracy: 0.9011 - val\_loss: 0.2782 - val\_accuracy: 0.9129  
Epoch 10/200  
75/75 [=====] - 0s 2ms/step - loss: 0.2657 - accuracy: 0.9137 - val\_loss: 0.2464 - val\_accuracy: 0.9179  
Epoch 11/200  
75/75 [=====] - 0s 3ms/step - loss: 0.2360 - accuracy: 0.9263 - val\_loss: 0.2186 - val\_accuracy: 0.9293  
Epoch 12/200  
75/75 [=====] - 0s 4ms/step - loss: 0.2103 - accuracy: 0.9377 - val\_loss: 0.1954 - val\_accuracy: 0.9457  
Epoch 13/200  
75/75 [=====] - 0s 4ms/step - loss: 0.1883 - accuracy: 0.9461 - val\_loss: 0.1763 - val\_accuracy: 0.9482  
Epoch 14/200  
75/75 [=====] - 0s 5ms/step - loss: 0.1700 - accuracy: 0.9541 - val\_loss: 0.1599 - val\_accuracy: 0.9520  
Epoch 15/200  
75/75 [=====] - 0s 4ms/step - loss: 0.1545 - accuracy: 0.9588 - val\_loss: 0.1466 - val\_accuracy: 0.9520  
Epoch 16/200  
75/75 [=====] - 0s 4ms/step - loss: 0.1416 - accuracy: 0.9613 - val\_loss: 0.1357 - val\_accuracy: 0.9533  
Epoch 17/200  
75/75 [=====] - 0s 4ms/step - loss: 0.1308 - accuracy: 0.9634 - val\_loss: 0.1269 - val\_accuracy: 0.9520  
Epoch 18/200  
75/75 [=====] - 0s 4ms/step - loss: 0.1221 - accuracy: 0.9642 - val\_loss: 0.1200 - val\_accuracy: 0.9533  
Epoch 19/200  
75/75 [=====] - 0s 5ms/step - loss: 0.1149 - accuracy: 0.9672 - val\_loss: 0.1141 - val\_accuracy: 0.9558  
Epoch 20/200  
75/75 [=====] - 0s 4ms/step - loss: 0.1088 - accuracy: 0.9668 - val\_loss: 0.1097 - val\_accuracy: 0.9558  
Epoch 21/200  
75/75 [=====] - 0s 6ms/step - loss: 0.1037 - accuracy: 0.9697 - val\_loss: 0.1059 - val\_accuracy: 0.9571  
Epoch 22/200  
75/75 [=====] - 1s 8ms/step - loss: 0.0996 - accuracy: 0.9697 - val\_loss: 0.1030 - val\_accuracy: 0.9583  
Epoch 23/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0959 - accuracy: 0.9701 - val\_loss: 0.1004 - val\_accuracy: 0.9596  
Epoch 24/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0929 - accuracy: 0.9710 - val\_loss: 0.0987 - val\_accuracy: 0.9583  
Epoch 25/200  
75/75 [=====] - 0s 2ms/step - loss: 0.0902 - accuracy: 0.9693 - val\_loss: 0.0969 - val\_accuracy: 0.9583  
Epoch 26/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0879 - accuracy: 0.9710 - val\_loss: 0.0954 - val\_accuracy: 0.9596  
Epoch 27/200  
75/75 [=====] - 0s 2ms/step - loss: 0.0861 - accuracy: 0.9718 - val\_loss: 0.0939 - val\_accuracy: 0.9596  
Epoch 28/200  
75/75 [=====] - 0s 2ms/step - loss: 0.0844 - accuracy: 0.9718 - val\_loss: 0.0929 - val\_accuracy: 0.9609  
Epoch 29/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0827 - accuracy: 0.9726 - val\_loss: 0.0918 - val\_accuracy: 0.9609  
Epoch 30/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0813 - accuracy: 0.9722 - val\_loss: 0.0910 - val\_accuracy: 0.9609  
Epoch 31/200  
75/75 [=====] - 0s 2ms/step - loss: 0.0801 - accuracy: 0.9726 - val\_loss: 0.0903 - val\_accuracy: 0.9609  
Epoch 32/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0791 - accuracy: 0.9731 - val\_loss: 0.0894 - val\_accuracy: 0.9609  
Epoch 33/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0779 - accuracy: 0.9739 - val\_loss: 0.0889 - val\_accuracy: 0.9596  
Epoch 34/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0769 - accuracy: 0.9743 - val\_loss: 0.0883 - val\_accuracy: 0.9596  
Epoch 35/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0761 - accuracy: 0.9743 - val\_loss: 0.0878 - val\_accuracy: 0.9609  
Epoch 36/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0754 - accuracy: 0.9743 - val\_loss: 0.0871 - val\_accuracy: 0.9634  
Epoch 37/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0748 - accuracy: 0.9739 - val\_loss: 0.0868 - val\_accuracy: 0.9634  
Epoch 38/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0741 - accuracy: 0.9739 - val\_loss: 0.0865 - val\_accuracy: 0.9634  
Epoch 39/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0735 - accuracy: 0.9769 - val\_loss: 0.0863 - val\_accuracy: 0.9697  
Epoch 40/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0729 - accuracy: 0.9760 - val\_loss: 0.0859 - val\_accuracy: 0.9710  
Epoch 41/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0724 - accuracy: 0.9781 - val\_loss: 0.0862 - val\_accuracy: 0.9710  
Epoch 42/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0721 - accuracy: 0.9785 - val\_loss: 0.0858 - val\_accuracy: 0.9710  
Epoch 43/200  
75/75 [=====] - 0s 5ms/step - loss: 0.0716 - accuracy: 0.9785 - val\_loss: 0.0855 - val\_accuracy: 0.9710  
Epoch 44/200  
75/75 [=====] - 1s 8ms/step - loss: 0.0712 - accuracy: 0.9794 - val\_loss: 0.0856 - val\_accuracy: 0.9710  
Epoch 45/200  
75/75 [=====] - 0s 5ms/step - loss: 0.0709 - accuracy: 0.9785 - val\_loss: 0.0852 - val\_accuracy: 0.9710  
Epoch 46/200  
75/75 [=====] - 0s 2ms/step - loss: 0.0705 - accuracy: 0.9798 - val\_loss: 0.0850 - val\_accuracy: 0.9710  
Epoch 47/200  
75/75 [=====] - 0s 2ms/step - loss: 0.0700 - accuracy: 0.9798 - val\_loss: 0.0847 - val\_accuracy: 0.9710  
Epoch 48/200  
75/75 [=====] - 0s 2ms/step - loss: 0.0697 - accuracy: 0.9811 - val\_loss: 0.0846 - val\_accuracy: 0.9722  
Epoch 49/200  
75/75 [=====] - 0s 2ms/step - loss: 0.0694 - accuracy: 0.9806 - val\_loss: 0.0843 - val\_accuracy: 0.9710  
Epoch 50/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0690 - accuracy: 0.9811 - val\_loss: 0.0841 - val\_accuracy: 0.9710  
Epoch 51/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0687 - accuracy: 0.9815 - val\_loss: 0.0841 - val\_accuracy: 0.9710  
Epoch 52/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0685 - accuracy: 0.9811 - val\_loss: 0.0837 - val\_accuracy: 0.9722  
Epoch 53/200  
75/75 [=====] - 0s 6ms/step - loss: 0.0681 - accuracy: 0.9811 - val\_loss: 0.0834 - val\_accuracy: 0.9710  
Epoch 54/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0678 - accuracy: 0.9811 - val\_loss: 0.0832 - val\_accuracy: 0.9710  
Epoch 55/200  
75/75 [=====] - 0s 6ms/step - loss: 0.0675 - accuracy: 0.9811 - val\_loss: 0.0829 - val\_accuracy: 0.9710  
Epoch 56/200  
75/75 [=====] - 1s 7ms/step - loss: 0.0674 - accuracy: 0.9811 - val\_loss: 0.0827 - val\_accuracy: 0.9722  
Epoch 57/200  
75/75 [=====] - 1s 8ms/step - loss: 0.0670 - accuracy: 0.9811 - val\_loss: 0.0828 - val\_accuracy: 0.9735  
Epoch 58/200  
75/75 [=====] - 0s 7ms/step - loss: 0.0669 - accuracy: 0.9815 - val\_loss: 0.0825 - val\_accuracy: 0.9710  
Epoch 59/200  
75/75 [=====] - 1s 8ms/step - loss: 0.0666 - accuracy: 0.9815 - val\_loss: 0.0824 - val\_accuracy: 0.9722  
Epoch 60/200  
75/75 [=====] - 0s 5ms/step - loss: 0.0665 - accuracy: 0.9811 - val\_loss: 0.0823 - val\_accuracy: 0.9722  
Epoch 61/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0660 - accuracy: 0.9823 - val\_loss: 0.0820 - val\_accuracy: 0.9722  
Epoch 62/200  
75/75 [=====] - 0s 3ms/step - loss: 0.0661 - accuracy: 0.9819 - val\_loss: 0.0820 - val\_accuracy: 0.9722  
Epoch 63/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0657 - accuracy: 0.9815 - val\_loss: 0.0818 - val\_accuracy: 0.9722  
Epoch 64/200  
75/75 [=====] - 0s 4ms/step - loss: 0.0656 - accuracy: 0.9819 - val\_loss: 0.0818 - val\_accuracy: 0.9735

[illegible]

[illegible]

```
Epoch 193/200
75/75 [=====] - 0s 2ms/step - loss: 0.0520 - accuracy: 0.9857 - val_loss: 0.0689 - val_accuracy: 0.9760
Epoch 194/200
75/75 [=====] - 0s 2ms/step - loss: 0.0523 - accuracy: 0.9857 - val_loss: 0.0686 - val_accuracy: 0.9747
Epoch 195/200
75/75 [=====] - 0s 3ms/step - loss: 0.0522 - accuracy: 0.9861 - val_loss: 0.0687 - val_accuracy: 0.9735
Epoch 196/200
75/75 [=====] - 0s 2ms/step - loss: 0.0521 - accuracy: 0.9853 - val_loss: 0.0687 - val_accuracy: 0.9747
Epoch 197/200
75/75 [=====] - 0s 2ms/step - loss: 0.0519 - accuracy: 0.9840 - val_loss: 0.0691 - val_accuracy: 0.9747
Epoch 198/200
75/75 [=====] - 0s 3ms/step - loss: 0.0519 - accuracy: 0.9857 - val_loss: 0.0688 - val_accuracy: 0.9747
Epoch 199/200
75/75 [=====] - 0s 2ms/step - loss: 0.0520 - accuracy: 0.9853 - val_loss: 0.0688 - val_accuracy: 0.9747
Epoch 200/200
75/75 [=====] - 0s 2ms/step - loss: 0.0518 - accuracy: 0.9861 - val_loss: 0.0685 - val_accuracy: 0.9747
```

```
In [106]: y_pred_prob_nn_supp2 = model_supp2.predict(X_test_norm)
y_pred_class_nn_supp2 = (y_pred_prob_nn_supp2 > 0.5).astype('int32')

25/25 [=====] - 0s 1ms/step
```

```
In [107]: y_pred_class_nn_supp2[:10]
```

```
Out[107]: array([[1],
 [1],
 [1],
 [1],
 [1],
 [1],
 [1],
 [1],
 [0],
 [0]], dtype=int32)
```

```
In [108]: y_pred_prob_nn_supp2[:10]
```

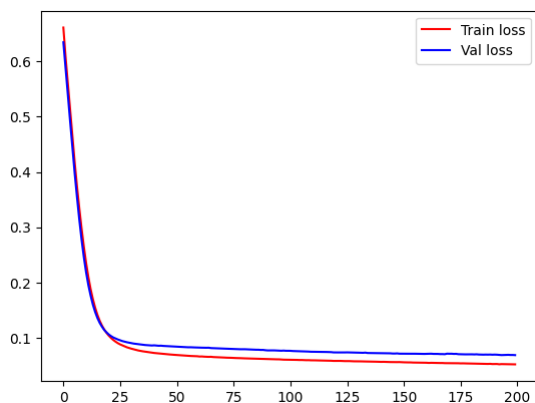
```
Out[108]: array([[9.9758953e-01],
 [9.9632859e-01],
 [9.9464029e-01],
 [9.9986553e-01],
 [9.9732304e-01],
 [9.5251268e-01],
 [9.9892521e-01],
 [5.4142284e-01],
 [2.0988989e-01],
 [3.9202740e-04]], dtype=float32)
```

```
In [109]: run_hist_supp2.history.keys()
```

```
Out[109]: dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

```
In [110]: fig, ax = plt.subplots()
ax.plot(run_hist_supp2.history['loss'], 'r', label = 'Train loss')
ax.plot(run_hist_supp2.history['val_loss'], 'b', label = "Val loss")
ax.legend()
```

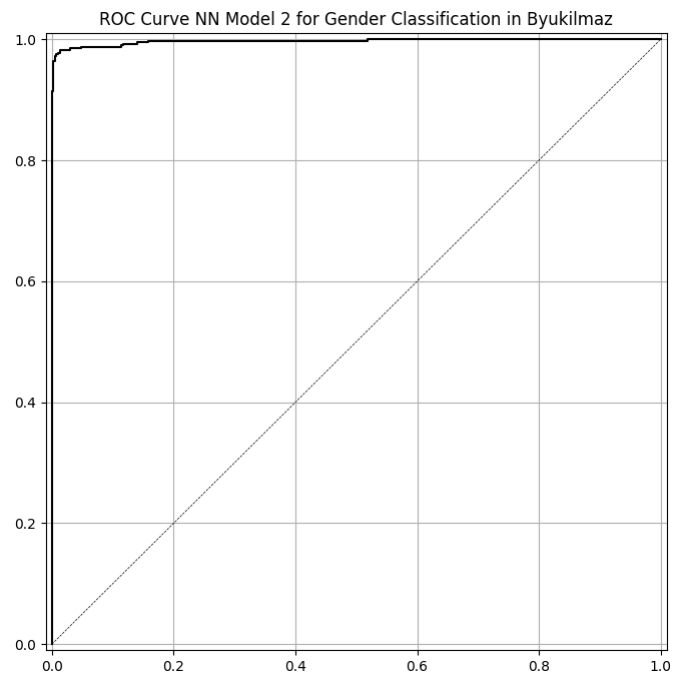
```
Out[110]: <matplotlib.legend.Legend at 0x7b29b595dd50>
```



```
In [119]: def plot_roc(y_test, y_pred, model_name):
fpr, tpr, thr = roc_curve(y_test, y_pred)
fig, ax = plt.subplots(figsize=(8, 8))
ax.plot(fpr, tpr, 'k-')
ax.plot([0, 1], [0, 1], 'k--', linewidth=.5) # roc curve for random model
ax.grid(True)
ax.set(title='ROC Curve {} for Gender Classification in Byukilmaz'.format(model_name),
xlim=[-0.01, 1.01], ylim=[-0.01, 1.01])
```

```
In [162]: print('accuracy is {:.3f}'.format(accuracy_score(y_test,y_pred_class_nn_supp2)))
print('roc_auc is {:.3f}'.format(roc_auc_score(y_test,y_pred_prob_nn_supp2)))
plot_roc(y_test, y_pred_prob_nn_supp, "NN Model 2")
```

```
accuracy is 0.975
roc_auc is 0.997
```



-With 2 hidden layers and 800 epoch with the same learning rate, this model shows a impressive result.

## 2 hidden layer, learning rate of .001, epoch of 800

```
In [151... model_supp3 = Sequential([
    Dense(6, input_shape = (20, ), activation = 'relu'),
    Dense(6, input_shape = (20, ), activation = 'relu'),
    Dense(1, activation = 'sigmoid')
])
```

```
In [152... model_supp3.compile(SGD(lr = 0.001), "binary_crossentropy", metrics = ['accuracy'])
run_hist_supp3 = model_supp3.fit(X_train_norm, y_train, validation_data = (X_test_norm, y_test), epochs = 800)
```

WARNING:absl:'lr' is deprecated in Keras optimizer, please use 'learning\_rate' or use the legacy optimizer, e.g., tf.keras.optimizers.legacy.SGD.

Epoch 1/800  
75/75 [=====] - 2s 11ms/step - loss: 0.6292 - accuracy: 0.5997 - val\_loss: 0.5769 - val\_accuracy: 0.6843  
Epoch 2/800  
75/75 [=====] - 0s 3ms/step - loss: 0.5529 - accuracy: 0.7210 - val\_loss: 0.5185 - val\_accuracy: 0.7866  
Epoch 3/800  
75/75 [=====] - 0s 3ms/step - loss: 0.5069 - accuracy: 0.7976 - val\_loss: 0.4776 - val\_accuracy: 0.8295  
Epoch 4/800  
75/75 [=====] - 0s 5ms/step - loss: 0.4734 - accuracy: 0.8312 - val\_loss: 0.4454 - val\_accuracy: 0.8687  
Epoch 5/800  
75/75 [=====] - 1s 7ms/step - loss: 0.4460 - accuracy: 0.8497 - val\_loss: 0.4178 - val\_accuracy: 0.8826  
Epoch 6/800  
75/75 [=====] - 0s 3ms/step - loss: 0.4217 - accuracy: 0.8670 - val\_loss: 0.3933 - val\_accuracy: 0.8965  
Epoch 7/800  
75/75 [=====] - 0s 2ms/step - loss: 0.3999 - accuracy: 0.8767 - val\_loss: 0.3711 - val\_accuracy: 0.9003  
Epoch 8/800  
75/75 [=====] - 0s 3ms/step - loss: 0.3798 - accuracy: 0.8809 - val\_loss: 0.3502 - val\_accuracy: 0.9091  
Epoch 9/800  
75/75 [=====] - 0s 2ms/step - loss: 0.3607 - accuracy: 0.8897 - val\_loss: 0.3303 - val\_accuracy: 0.9129  
Epoch 10/800  
75/75 [=====] - 0s 3ms/step - loss: 0.3407 - accuracy: 0.8956 - val\_loss: 0.3099 - val\_accuracy: 0.9217  
Epoch 11/800  
75/75 [=====] - 0s 3ms/step - loss: 0.3187 - accuracy: 0.9053 - val\_loss: 0.2887 - val\_accuracy: 0.9318  
Epoch 12/800  
75/75 [=====] - 0s 2ms/step - loss: 0.2950 - accuracy: 0.9184 - val\_loss: 0.2653 - val\_accuracy: 0.9407  
Epoch 13/800  
75/75 [=====] - 0s 2ms/step - loss: 0.2696 - accuracy: 0.9285 - val\_loss: 0.2417 - val\_accuracy: 0.9495  
Epoch 14/800  
75/75 [=====] - 0s 2ms/step - loss: 0.2450 - accuracy: 0.9377 - val\_loss: 0.2190 - val\_accuracy: 0.9545  
Epoch 15/800  
75/75 [=====] - 0s 2ms/step - loss: 0.2223 - accuracy: 0.9440 - val\_loss: 0.1988 - val\_accuracy: 0.9609  
Epoch 16/800  
75/75 [=====] - 0s 3ms/step - loss: 0.2022 - accuracy: 0.9478 - val\_loss: 0.1800 - val\_accuracy: 0.9634  
Epoch 17/800  
75/75 [=====] - 0s 2ms/step - loss: 0.1843 - accuracy: 0.9524 - val\_loss: 0.1642 - val\_accuracy: 0.9659  
Epoch 18/800  
75/75 [=====] - 0s 4ms/step - loss: 0.1689 - accuracy: 0.9537 - val\_loss: 0.1505 - val\_accuracy: 0.9672  
Epoch 19/800  
75/75 [=====] - 0s 3ms/step - loss: 0.1556 - accuracy: 0.9566 - val\_loss: 0.1390 - val\_accuracy: 0.9722  
Epoch 20/800  
75/75 [=====] - 0s 4ms/step - loss: 0.1446 - accuracy: 0.9588 - val\_loss: 0.1294 - val\_accuracy: 0.9735  
Epoch 21/800  
75/75 [=====] - 0s 3ms/step - loss: 0.1352 - accuracy: 0.9617 - val\_loss: 0.1210 - val\_accuracy: 0.9760  
Epoch 22/800  
75/75 [=====] - 0s 3ms/step - loss: 0.1273 - accuracy: 0.9630 - val\_loss: 0.1146 - val\_accuracy: 0.9747  
Epoch 23/800  
75/75 [=====] - 0s 4ms/step - loss: 0.1204 - accuracy: 0.9638 - val\_loss: 0.1094 - val\_accuracy: 0.9735  
Epoch 24/800  
75/75 [=====] - 0s 4ms/step - loss: 0.1146 - accuracy: 0.9642 - val\_loss: 0.1048 - val\_accuracy: 0.9722  
Epoch 25/800  
75/75 [=====] - 0s 3ms/step - loss: 0.1098 - accuracy: 0.9651 - val\_loss: 0.1014 - val\_accuracy: 0.9710  
Epoch 26/800  
75/75 [=====] - 0s 3ms/step - loss: 0.1059 - accuracy: 0.9668 - val\_loss: 0.0980 - val\_accuracy: 0.9710  
Epoch 27/800  
75/75 [=====] - 0s 4ms/step - loss: 0.1022 - accuracy: 0.9680 - val\_loss: 0.0949 - val\_accuracy: 0.9710  
Epoch 28/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0992 - accuracy: 0.9684 - val\_loss: 0.0929 - val\_accuracy: 0.9710  
Epoch 29/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0966 - accuracy: 0.9697 - val\_loss: 0.0906 - val\_accuracy: 0.9710  
Epoch 30/800  
75/75 [=====] - 0s 4ms/step - loss: 0.0944 - accuracy: 0.9693 - val\_loss: 0.0892 - val\_accuracy: 0.9684  
Epoch 31/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0925 - accuracy: 0.9710 - val\_loss: 0.0877 - val\_accuracy: 0.9697  
Epoch 32/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0906 - accuracy: 0.9718 - val\_loss: 0.0865 - val\_accuracy: 0.9697  
Epoch 33/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0891 - accuracy: 0.9726 - val\_loss: 0.0852 - val\_accuracy: 0.9710  
Epoch 34/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0876 - accuracy: 0.9726 - val\_loss: 0.0841 - val\_accuracy: 0.9710  
Epoch 35/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0865 - accuracy: 0.9722 - val\_loss: 0.0830 - val\_accuracy: 0.9710  
Epoch 36/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0852 - accuracy: 0.9739 - val\_loss: 0.0817 - val\_accuracy: 0.9710  
Epoch 37/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0844 - accuracy: 0.9722 - val\_loss: 0.0814 - val\_accuracy: 0.9710  
Epoch 38/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0834 - accuracy: 0.9731 - val\_loss: 0.0805 - val\_accuracy: 0.9710  
Epoch 39/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0825 - accuracy: 0.9739 - val\_loss: 0.0803 - val\_accuracy: 0.9710  
Epoch 40/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0816 - accuracy: 0.9739 - val\_loss: 0.0792 - val\_accuracy: 0.9710  
Epoch 41/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0810 - accuracy: 0.9739 - val\_loss: 0.0790 - val\_accuracy: 0.9697  
Epoch 42/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0800 - accuracy: 0.9743 - val\_loss: 0.0790 - val\_accuracy: 0.9697  
Epoch 43/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0795 - accuracy: 0.9764 - val\_loss: 0.0782 - val\_accuracy: 0.9697  
Epoch 44/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0788 - accuracy: 0.9760 - val\_loss: 0.0780 - val\_accuracy: 0.9697  
Epoch 45/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0784 - accuracy: 0.9769 - val\_loss: 0.0766 - val\_accuracy: 0.9722  
Epoch 46/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0778 - accuracy: 0.9760 - val\_loss: 0.0757 - val\_accuracy: 0.9722  
Epoch 47/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0772 - accuracy: 0.9764 - val\_loss: 0.0752 - val\_accuracy: 0.9735  
Epoch 48/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0763 - accuracy: 0.9773 - val\_loss: 0.0762 - val\_accuracy: 0.9722  
Epoch 49/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0761 - accuracy: 0.9777 - val\_loss: 0.0748 - val\_accuracy: 0.9722  
Epoch 50/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0756 - accuracy: 0.9785 - val\_loss: 0.0746 - val\_accuracy: 0.9722  
Epoch 51/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0753 - accuracy: 0.9777 - val\_loss: 0.0744 - val\_accuracy: 0.9722  
Epoch 52/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0748 - accuracy: 0.9794 - val\_loss: 0.0737 - val\_accuracy: 0.9710  
Epoch 53/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0744 - accuracy: 0.9777 - val\_loss: 0.0733 - val\_accuracy: 0.9722  
Epoch 54/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0741 - accuracy: 0.9773 - val\_loss: 0.0736 - val\_accuracy: 0.9722  
Epoch 55/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0737 - accuracy: 0.9794 - val\_loss: 0.0733 - val\_accuracy: 0.9722  
Epoch 56/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0732 - accuracy: 0.9790 - val\_loss: 0.0737 - val\_accuracy: 0.9735  
Epoch 57/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0729 - accuracy: 0.9811 - val\_loss: 0.0728 - val\_accuracy: 0.9722  
Epoch 58/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0725 - accuracy: 0.9802 - val\_loss: 0.0724 - val\_accuracy: 0.9735  
Epoch 59/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0722 - accuracy: 0.9798 - val\_loss: 0.0727 - val\_accuracy: 0.9722  
Epoch 60/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0720 - accuracy: 0.9798 - val\_loss: 0.0725 - val\_accuracy: 0.9722  
Epoch 61/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0716 - accuracy: 0.9802 - val\_loss: 0.0724 - val\_accuracy: 0.9722  
Epoch 62/800  
75/75 [=====] - 0s 2ms/step - loss: 0.0714 - accuracy: 0.9802 - val\_loss: 0.0722 - val\_accuracy: 0.9722  
Epoch 63/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0712 - accuracy: 0.9798 - val\_loss: 0.0721 - val\_accuracy: 0.9722  
Epoch 64/800  
75/75 [=====] - 0s 3ms/step - loss: 0.0708 - accuracy: 0.9806 - val\_loss: 0.0714 - val\_accuracy: 0.9760



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

```

Epoch 769/800
75/75 [=====] - 0s 3ms/step - loss: 0.0332 - accuracy: 0.9891 - val_loss: 0.0653 - val_accuracy: 0.9811
Epoch 770/800
75/75 [=====] - 0s 4ms/step - loss: 0.0328 - accuracy: 0.9886 - val_loss: 0.0665 - val_accuracy: 0.9785
Epoch 771/800
75/75 [=====] - 0s 3ms/step - loss: 0.0331 - accuracy: 0.9891 - val_loss: 0.0661 - val_accuracy: 0.9773
Epoch 772/800
75/75 [=====] - 0s 3ms/step - loss: 0.0327 - accuracy: 0.9891 - val_loss: 0.0659 - val_accuracy: 0.9773
Epoch 773/800
75/75 [=====] - 0s 3ms/step - loss: 0.0330 - accuracy: 0.9878 - val_loss: 0.0659 - val_accuracy: 0.9798
Epoch 774/800
75/75 [=====] - 0s 3ms/step - loss: 0.0326 - accuracy: 0.9886 - val_loss: 0.0657 - val_accuracy: 0.9798
Epoch 775/800
75/75 [=====] - 0s 3ms/step - loss: 0.0328 - accuracy: 0.9891 - val_loss: 0.0658 - val_accuracy: 0.9811
Epoch 776/800
75/75 [=====] - 0s 3ms/step - loss: 0.0328 - accuracy: 0.9882 - val_loss: 0.0662 - val_accuracy: 0.9785
Epoch 777/800
75/75 [=====] - 0s 3ms/step - loss: 0.0328 - accuracy: 0.9886 - val_loss: 0.0663 - val_accuracy: 0.9798
Epoch 778/800
75/75 [=====] - 0s 3ms/step - loss: 0.0325 - accuracy: 0.9895 - val_loss: 0.0658 - val_accuracy: 0.9798
Epoch 779/800
75/75 [=====] - 0s 3ms/step - loss: 0.0328 - accuracy: 0.9886 - val_loss: 0.0661 - val_accuracy: 0.9811
Epoch 780/800
75/75 [=====] - 0s 3ms/step - loss: 0.0326 - accuracy: 0.9886 - val_loss: 0.0665 - val_accuracy: 0.9785
Epoch 781/800
75/75 [=====] - 0s 3ms/step - loss: 0.0319 - accuracy: 0.9891 - val_loss: 0.0662 - val_accuracy: 0.9798
Epoch 782/800
75/75 [=====] - 0s 3ms/step - loss: 0.0328 - accuracy: 0.9886 - val_loss: 0.0658 - val_accuracy: 0.9798
Epoch 783/800
75/75 [=====] - 0s 3ms/step - loss: 0.0328 - accuracy: 0.9882 - val_loss: 0.0661 - val_accuracy: 0.9798
Epoch 784/800
75/75 [=====] - 0s 3ms/step - loss: 0.0326 - accuracy: 0.9891 - val_loss: 0.0665 - val_accuracy: 0.9785
Epoch 785/800
75/75 [=====] - 0s 3ms/step - loss: 0.0326 - accuracy: 0.9895 - val_loss: 0.0663 - val_accuracy: 0.9798
Epoch 786/800
75/75 [=====] - 0s 3ms/step - loss: 0.0325 - accuracy: 0.9891 - val_loss: 0.0665 - val_accuracy: 0.9798
Epoch 787/800
75/75 [=====] - 0s 3ms/step - loss: 0.0327 - accuracy: 0.9891 - val_loss: 0.0657 - val_accuracy: 0.9811
Epoch 788/800
75/75 [=====] - 0s 3ms/step - loss: 0.0325 - accuracy: 0.9886 - val_loss: 0.0659 - val_accuracy: 0.9811
Epoch 789/800
75/75 [=====] - 0s 3ms/step - loss: 0.0325 - accuracy: 0.9886 - val_loss: 0.0656 - val_accuracy: 0.9798
Epoch 790/800
75/75 [=====] - 0s 3ms/step - loss: 0.0326 - accuracy: 0.9886 - val_loss: 0.0658 - val_accuracy: 0.9798
Epoch 791/800
75/75 [=====] - 0s 3ms/step - loss: 0.0322 - accuracy: 0.9886 - val_loss: 0.0672 - val_accuracy: 0.9773
Epoch 792/800
75/75 [=====] - 0s 3ms/step - loss: 0.0324 - accuracy: 0.9886 - val_loss: 0.0670 - val_accuracy: 0.9773
Epoch 793/800
75/75 [=====] - 0s 3ms/step - loss: 0.0322 - accuracy: 0.9874 - val_loss: 0.0667 - val_accuracy: 0.9811
Epoch 794/800
75/75 [=====] - 0s 3ms/step - loss: 0.0325 - accuracy: 0.9891 - val_loss: 0.0666 - val_accuracy: 0.9798
Epoch 795/800
75/75 [=====] - 0s 3ms/step - loss: 0.0325 - accuracy: 0.9886 - val_loss: 0.0665 - val_accuracy: 0.9811
Epoch 796/800
75/75 [=====] - 0s 3ms/step - loss: 0.0320 - accuracy: 0.9895 - val_loss: 0.0678 - val_accuracy: 0.9785
Epoch 797/800
75/75 [=====] - 0s 3ms/step - loss: 0.0325 - accuracy: 0.9882 - val_loss: 0.0673 - val_accuracy: 0.9798
Epoch 798/800
75/75 [=====] - 0s 3ms/step - loss: 0.0323 - accuracy: 0.9882 - val_loss: 0.0670 - val_accuracy: 0.9811
Epoch 799/800
75/75 [=====] - 0s 3ms/step - loss: 0.0323 - accuracy: 0.9891 - val_loss: 0.0666 - val_accuracy: 0.9798
Epoch 800/800
75/75 [=====] - 0s 3ms/step - loss: 0.0324 - accuracy: 0.9886 - val_loss: 0.0667 - val_accuracy: 0.9798

```

```

In [153.. y_pred_prob_nn_supp3 = model_supp3.predict(X_test_norm)
y_pred_class_nn_supp3 = (y_pred_prob_nn_supp3 > 0.5).astype('int32')
25/25 [=====] - 0s 1ms/step

```

```

In [154.. y_pred_class_nn_supp3[:10]

```

```

Out[154]: array([[1],
          [1],
          [1],
          [1],
          [1],
          [1],
          [1],
          [0],
          [0]], dtype=int32)

```

```

In [155.. y_pred_prob_nn_supp3[:10]

```

```

Out[155]: array([[9.9999905e-01],
          [9.9985705e-01],
          [9.9691033e-01],
          [9.9999899e-01],
          [9.9986291e-01],
          [9.9948049e-01],
          [9.9999982e-01],
          [9.9930364e-01],
          [2.1352344e-03],
          [3.7764621e-06]], dtype=float32)

```

```

In [156.. run_hist_supp3.history.keys()

```

```

Out[156]: dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

```

```

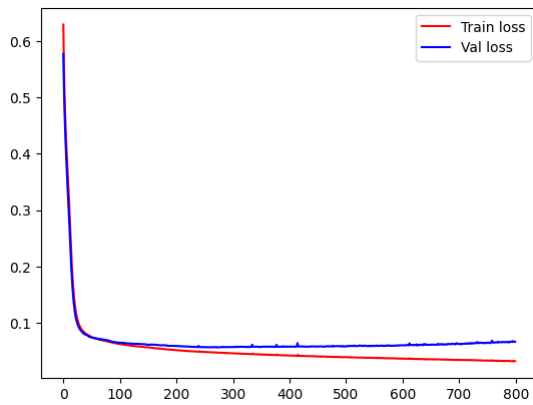
In [157.. fig, ax = plt.subplots()
ax.plot(run_hist_supp3.history['loss'], 'r', label = 'Train loss')
ax.plot(run_hist_supp3.history['val_loss'], 'b', label = "Val loss")
ax.legend()

```

```

Out[157]: <matplotlib.legend.Legend at 0x7b29a2ee680>

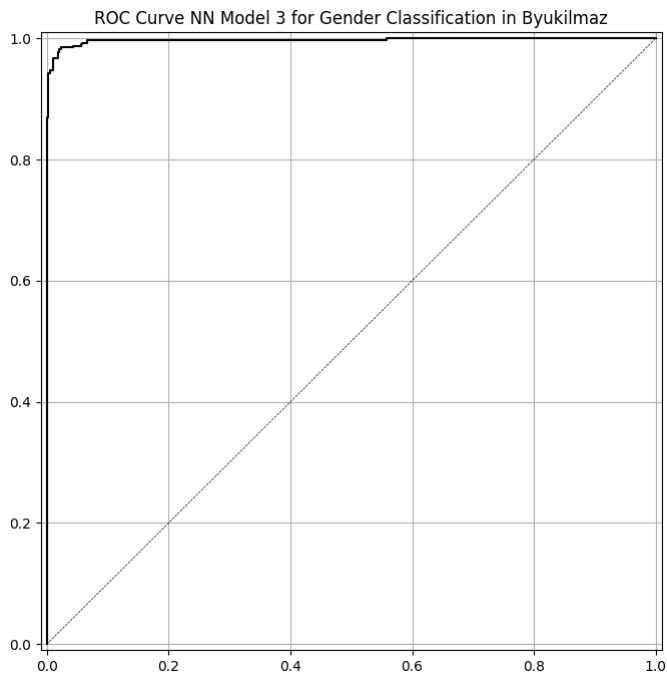
```



```
In [158]: def plot_roc(y_test, y_pred, model_name):
fpr, tpr, thr = roc_curve(y_test, y_pred)
fig, ax = plt.subplots(figsize=(8, 8))
ax.plot(fpr, tpr, 'k-')
ax.plot([0, 1], [0, 1], 'k--', linewidth=.5)
ax.grid(True)
ax.set(title='ROC Curve {} for Gender Classification in Byukilmaz'.format(model_name),
      xlim=[-0.01, 1.01], ylim=[-0.01, 1.01])
```

```
In [161]: print('accuracy is {:.3f}'.format(accuracy_score(y_test,y_pred_class_nn_supp3)))
print('roc-auc is {:.3f}'.format(roc_auc_score(y_test,y_pred_prob_nn_supp3)))
plot_roc(y_test, y_pred_prob_nn_supp3, "NN Model 3")
```

accuracy is 0.980  
roc-auc is 0.997



- This model shows a

## Conclusion

This activity demonstrates how does building and training the neural network underscore the importance of meticulous data preprocessing. And, Understanding the nature of the data, handling non-numeric values, and adapting code to different data structures in ensuring a smooth and error-free machine learning workflow.

## Google Colab link

### Colab Link:

<https://colab.research.google.com/drive/1Sg9sGnBQyJAroSFE4ev3ObrURD91yTRZ?usp=sharing>

```
In [160]: #!jupyter-nbconvert --to html '/content/drive/MyDrive/CoLab Notebooks/Hands-on Activity 1.2 - Training Neural Networks (Figueroa).ipynb'
```