2020_1127_DNN_Classification

November 27, 2020

1 Classifying Pulsars from the High Time Resolution Universe Survey (HTRU2) - Deep Neural Network (DNN) Classification

1.1 Overview & Citation

In this code notebook, we attempt to classify pulsars from the High Time Resolution Universe Survey, South (HTRU2) dataset using deep neural network (DNN) classification. The dataset was retrieved from the UC Irvine Machine Learning Repository at the following link: https://archive.ics.uci.edu/ml/datasets/HTRU2#.

The dataset was donated to the UCI Repository by Dr. Robert Lyon of The University of Manchester, United Kingdom. The two papers requested for citation in the description are listed below:

- R. J. Lyon, B. W. Stappers, S. Cooper, J. M. Brooke, J. D. Knowles, Fifty Years of Pulsar Candidate Selection: From simple filters to a new principled real-time classification approach, Monthly Notices of the Royal Astronomical Society 459 (1), 1104-1123, DOI: 10.1093/mnras/stw656
- R. J. Lyon, HTRU2, DOI: 10.6084/m9.figshare.3080389.v1.

1.2 Import the Relevant Libraries

[108]:
```
# Data Manipulation
import numpy as np
import pandas as pd

# Data Visualization
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
sns.set()

# Data Preprocessing
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler

# ANN Modeling in TensorFlow & Keras
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation
```

```
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import Dropout

# Model Evaluation
from sklearn.metrics import classification_report,confusion_matrix
```

1.3 Data Preprocessing

1.3.1 Import & Check the Data

[109]:
```
df = pd.read_csv('2020_1125_Pulsar_Data.csv')
pulsar_data = df.copy()
```

[110]:
```
pulsar_data.head()
```

[110]:

	IP_Mean	IP_StdDev	IP_Kurtosis	IP_Skewness	DM_Mean	DM_StdDev	\
0	140.562500	55.683782	-0.234571	-0.699648	3.199833	19.110426	
1	102.507812	58.882430	0.465318	-0.515088	1.677258	14.860146	
2	103.015625	39.341649	0.323328	1.051164	3.121237	21.744669	
3	136.750000	57.178449	-0.068415	-0.636238	3.642977	20.959280	
4	88.726562	40.672225	0.600866	1.123492	1.178930	11.468720	

	DM_Kurtosis	DM_Skewness	Class
0	7.975532	74.242225	0
1	10.576487	127.393580	0
2	7.735822	63.171909	0
3	6.896499	53.593661	0
4	14.269573	252.567306	0

1.3.2 Train Test Split

[111]:
```
X = pulsar_data.drop('Class',axis=1)
y = pulsar_data['Class']
```

[112]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,␣
↪random_state=42)
```

1.3.3 Scale the Data

[113]:
```
scaler = MinMaxScaler()
X_train= scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

1.4 Model 1 - DNN with 2 Hidden Layers

1.4.1 Construct the Deep Neural Network

[115]:
```
# Determine number of starting nodes by finding the shape of X_train
X_train.shape
```

[115]:
```
(13423, 8)
```

[133]:
```
model = Sequential()

# Input Layer
model.add(Dense(8,activation='relu')) # All layers utilize rectified linear␣
 ↪units (relu)

# Hidden Layers
model.add(Dense(8,activation='relu'))
model.add(Dense(8,activation='relu'))

# Output Layer (Sigmoid for Binary Classification)
model.add(Dense(1,activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam')
```

1.4.2 Train the Model on the Test Data

[134]:
```
model.fit(x=X_train, y=y_train, epochs=200, validation_data=(X_test, y_test),␣
 ↪verbose=1)
```

```
Epoch 1/200
420/420 [==============================] - 0s 737us/step - loss: 0.3329 -
val_loss: 0.1499
Epoch 2/200
420/420 [==============================] - 0s 588us/step - loss: 0.1032 -
val_loss: 0.0866
Epoch 3/200
420/420 [==============================] - 0s 600us/step - loss: 0.0874 -
val_loss: 0.0840
Epoch 4/200
420/420 [==============================] - 0s 616us/step - loss: 0.0850 -
val_loss: 0.0838
Epoch 5/200
420/420 [==============================] - 0s 589us/step - loss: 0.0842 -
val_loss: 0.0831
Epoch 6/200
420/420 [==============================] - 0s 586us/step - loss: 0.0836 -
val_loss: 0.0833
Epoch 7/200
```

```
420/420 [==============================] - 0s 589us/step - loss: 0.0830 -
val_loss: 0.0814
Epoch 8/200
420/420 [==============================] - 0s 593us/step - loss: 0.0823 -
val_loss: 0.0814
Epoch 9/200
420/420 [==============================] - 0s 589us/step - loss: 0.0817 -
val_loss: 0.0806
Epoch 10/200
420/420 [==============================] - 0s 590us/step - loss: 0.0811 -
val_loss: 0.0803
Epoch 11/200
420/420 [==============================] - 0s 586us/step - loss: 0.0805 -
val_loss: 0.0817
Epoch 12/200
420/420 [==============================] - 0s 590us/step - loss: 0.0804 -
val_loss: 0.0797
Epoch 13/200
420/420 [==============================] - 0s 589us/step - loss: 0.0798 -
val_loss: 0.0794
Epoch 14/200
420/420 [==============================] - 0s 582us/step - loss: 0.0796 -
val_loss: 0.0808
Epoch 15/200
420/420 [==============================] - 0s 615us/step - loss: 0.0795 -
val_loss: 0.0793
Epoch 16/200
420/420 [==============================] - 0s 581us/step - loss: 0.0790 -
val_loss: 0.0786
Epoch 17/200
420/420 [==============================] - 0s 585us/step - loss: 0.0786 -
val_loss: 0.0785
Epoch 18/200
420/420 [==============================] - 0s 587us/step - loss: 0.0783 -
val_loss: 0.0783
Epoch 19/200
420/420 [==============================] - 0s 587us/step - loss: 0.0778 -
val_loss: 0.0784
Epoch 20/200
420/420 [==============================] - 0s 590us/step - loss: 0.0776 -
val_loss: 0.0784
Epoch 21/200
420/420 [==============================] - 0s 586us/step - loss: 0.0778 -
val_loss: 0.0788
Epoch 22/200
420/420 [==============================] - 0s 588us/step - loss: 0.0776 -
val_loss: 0.0805
Epoch 23/200
```

```
420/420 [==============================] - 0s 585us/step - loss: 0.0770 -
val_loss: 0.0799
Epoch 24/200
420/420 [==============================] - 0s 588us/step - loss: 0.0770 -
val_loss: 0.0775
Epoch 25/200
420/420 [==============================] - 0s 594us/step - loss: 0.0766 -
val_loss: 0.0772
Epoch 26/200
420/420 [==============================] - 0s 584us/step - loss: 0.0763 -
val_loss: 0.0774
Epoch 27/200
420/420 [==============================] - 0s 593us/step - loss: 0.0762 -
val_loss: 0.0787
Epoch 28/200
420/420 [==============================] - 0s 591us/step - loss: 0.0761 -
val_loss: 0.0771
Epoch 29/200
420/420 [==============================] - 0s 593us/step - loss: 0.0754 -
val_loss: 0.0794
Epoch 30/200
420/420 [==============================] - 0s 590us/step - loss: 0.0755 -
val_loss: 0.0767
Epoch 31/200
420/420 [==============================] - 0s 588us/step - loss: 0.0754 -
val_loss: 0.0764
Epoch 32/200
420/420 [==============================] - 0s 590us/step - loss: 0.0751 -
val_loss: 0.0775
Epoch 33/200
420/420 [==============================] - 0s 590us/step - loss: 0.0747 -
val_loss: 0.0765
Epoch 34/200
420/420 [==============================] - 0s 586us/step - loss: 0.0746 -
val_loss: 0.0761
Epoch 35/200
420/420 [==============================] - 0s 592us/step - loss: 0.0746 -
val_loss: 0.0777
Epoch 36/200
420/420 [==============================] - 0s 596us/step - loss: 0.0745 -
val_loss: 0.0760
Epoch 37/200
420/420 [==============================] - 0s 596us/step - loss: 0.0741 -
val_loss: 0.0801
Epoch 38/200
420/420 [==============================] - 0s 587us/step - loss: 0.0738 -
val_loss: 0.0770
Epoch 39/200
```

```
420/420 [==============================] - 0s 595us/step - loss: 0.0742 -
val_loss: 0.0767
Epoch 40/200
420/420 [==============================] - 0s 592us/step - loss: 0.0738 -
val_loss: 0.0752
Epoch 41/200
420/420 [==============================] - 0s 590us/step - loss: 0.0735 -
val_loss: 0.0752
Epoch 42/200
420/420 [==============================] - 0s 595us/step - loss: 0.0731 -
val_loss: 0.0753
Epoch 43/200
420/420 [==============================] - 0s 594us/step - loss: 0.0730 -
val_loss: 0.0753
Epoch 44/200
420/420 [==============================] - 0s 616us/step - loss: 0.0727 -
val_loss: 0.0782
Epoch 45/200
420/420 [==============================] - 0s 582us/step - loss: 0.0731 -
val_loss: 0.0759
Epoch 46/200
420/420 [==============================] - 0s 587us/step - loss: 0.0727 -
val_loss: 0.0755
Epoch 47/200
420/420 [==============================] - 0s 595us/step - loss: 0.0723 -
val_loss: 0.0750
Epoch 48/200
420/420 [==============================] - 0s 624us/step - loss: 0.0729 -
val_loss: 0.0753
Epoch 49/200
420/420 [==============================] - 0s 615us/step - loss: 0.0723 -
val_loss: 0.0767
Epoch 50/200
420/420 [==============================] - 0s 585us/step - loss: 0.0726 -
val_loss: 0.0747
Epoch 51/200
420/420 [==============================] - 0s 595us/step - loss: 0.0724 -
val_loss: 0.0756
Epoch 52/200
420/420 [==============================] - 0s 591us/step - loss: 0.0724 -
val_loss: 0.0741
Epoch 53/200
420/420 [==============================] - 0s 597us/step - loss: 0.0719 -
val_loss: 0.0739
Epoch 54/200
420/420 [==============================] - 0s 617us/step - loss: 0.0716 -
val_loss: 0.0748
Epoch 55/200
```

```
420/420 [==============================] - 0s 602us/step - loss: 0.0718 -
val_loss: 0.0738
Epoch 56/200
420/420 [==============================] - 0s 585us/step - loss: 0.0715 -
val_loss: 0.0740
Epoch 57/200
420/420 [==============================] - 0s 585us/step - loss: 0.0717 -
val_loss: 0.0735
Epoch 58/200
420/420 [==============================] - 0s 585us/step - loss: 0.0711 -
val_loss: 0.0738
Epoch 59/200
420/420 [==============================] - 0s 583us/step - loss: 0.0714 -
val_loss: 0.0736
Epoch 60/200
420/420 [==============================] - 0s 589us/step - loss: 0.0714 -
val_loss: 0.0745
Epoch 61/200
420/420 [==============================] - 0s 584us/step - loss: 0.0714 -
val_loss: 0.0737
Epoch 62/200
420/420 [==============================] - 0s 580us/step - loss: 0.0708 -
val_loss: 0.0732
Epoch 63/200
420/420 [==============================] - 0s 583us/step - loss: 0.0712 -
val_loss: 0.0739
Epoch 64/200
420/420 [==============================] - 0s 588us/step - loss: 0.0707 -
val_loss: 0.0732
Epoch 65/200
420/420 [==============================] - 0s 583us/step - loss: 0.0712 -
val_loss: 0.0732
Epoch 66/200
420/420 [==============================] - 0s 584us/step - loss: 0.0710 -
val_loss: 0.0753
Epoch 67/200
420/420 [==============================] - 0s 589us/step - loss: 0.0710 -
val_loss: 0.0748
Epoch 68/200
420/420 [==============================] - 0s 586us/step - loss: 0.0705 -
val_loss: 0.0735
Epoch 69/200
420/420 [==============================] - 0s 589us/step - loss: 0.0706 -
val_loss: 0.0733
Epoch 70/200
420/420 [==============================] - 0s 600us/step - loss: 0.0702 -
val_loss: 0.0738
Epoch 71/200
```

```
420/420 [==============================] - 0s 584us/step - loss: 0.0706 -
val_loss: 0.0728
Epoch 72/200
420/420 [==============================] - 0s 584us/step - loss: 0.0702 -
val_loss: 0.0730
Epoch 73/200
420/420 [==============================] - 0s 592us/step - loss: 0.0702 -
val_loss: 0.0729
Epoch 74/200
420/420 [==============================] - 0s 589us/step - loss: 0.0696 -
val_loss: 0.0728
Epoch 75/200
420/420 [==============================] - 0s 587us/step - loss: 0.0698 -
val_loss: 0.0784
Epoch 76/200
420/420 [==============================] - 0s 589us/step - loss: 0.0696 -
val_loss: 0.0750
Epoch 77/200
420/420 [==============================] - 0s 588us/step - loss: 0.0698 -
val_loss: 0.0725
Epoch 78/200
420/420 [==============================] - 0s 585us/step - loss: 0.0693 -
val_loss: 0.0779
Epoch 79/200
420/420 [==============================] - 0s 595us/step - loss: 0.0698 -
val_loss: 0.0725
Epoch 80/200
420/420 [==============================] - 0s 585us/step - loss: 0.0699 -
val_loss: 0.0738
Epoch 81/200
420/420 [==============================] - 0s 581us/step - loss: 0.0701 -
val_loss: 0.0722
Epoch 82/200
420/420 [==============================] - 0s 575us/step - loss: 0.0700 -
val_loss: 0.0723
Epoch 83/200
420/420 [==============================] - 0s 579us/step - loss: 0.0693 -
val_loss: 0.0720
Epoch 84/200
420/420 [==============================] - 0s 582us/step - loss: 0.0695 -
val_loss: 0.0727
Epoch 85/200
420/420 [==============================] - 0s 580us/step - loss: 0.0693 -
val_loss: 0.0728
Epoch 86/200
420/420 [==============================] - 0s 576us/step - loss: 0.0691 -
val_loss: 0.0719
Epoch 87/200
```

```
420/420 [==============================] - 0s 580us/step - loss: 0.0697 -
val_loss: 0.0727
Epoch 88/200
420/420 [==============================] - 0s 581us/step - loss: 0.0689 -
val_loss: 0.0759
Epoch 89/200
420/420 [==============================] - 0s 579us/step - loss: 0.0692 -
val_loss: 0.0752
Epoch 90/200
420/420 [==============================] - 0s 585us/step - loss: 0.0698 -
val_loss: 0.0719
Epoch 91/200
420/420 [==============================] - 0s 579us/step - loss: 0.0696 -
val_loss: 0.0723
Epoch 92/200
420/420 [==============================] - 0s 583us/step - loss: 0.0692 -
val_loss: 0.0735
Epoch 93/200
420/420 [==============================] - 0s 583us/step - loss: 0.0688 -
val_loss: 0.0735
Epoch 94/200
420/420 [==============================] - 0s 582us/step - loss: 0.0690 -
val_loss: 0.0721
Epoch 95/200
420/420 [==============================] - 0s 596us/step - loss: 0.0682 -
val_loss: 0.0736
Epoch 96/200
420/420 [==============================] - 0s 576us/step - loss: 0.0690 -
val_loss: 0.0719
Epoch 97/200
420/420 [==============================] - 0s 587us/step - loss: 0.0693 -
val_loss: 0.0747
Epoch 98/200
420/420 [==============================] - 0s 583us/step - loss: 0.0688 -
val_loss: 0.0716
Epoch 99/200
420/420 [==============================] - 0s 578us/step - loss: 0.0686 -
val_loss: 0.0717
Epoch 100/200
420/420 [==============================] - 0s 578us/step - loss: 0.0687 -
val_loss: 0.0713
Epoch 101/200
420/420 [==============================] - 0s 583us/step - loss: 0.0688 -
val_loss: 0.0723
Epoch 102/200
420/420 [==============================] - 0s 581us/step - loss: 0.0686 -
val_loss: 0.0715
Epoch 103/200
```

```
420/420 [==============================] - 0s 582us/step - loss: 0.0684 -
val_loss: 0.0732
Epoch 104/200
420/420 [==============================] - 0s 587us/step - loss: 0.0685 -
val_loss: 0.0712
Epoch 105/200
420/420 [==============================] - 0s 582us/step - loss: 0.0685 -
val_loss: 0.0717
Epoch 106/200
420/420 [==============================] - 0s 580us/step - loss: 0.0682 -
val_loss: 0.0713
Epoch 107/200
420/420 [==============================] - 0s 586us/step - loss: 0.0683 -
val_loss: 0.0712
Epoch 108/200
420/420 [==============================] - 0s 576us/step - loss: 0.0688 -
val_loss: 0.0710
Epoch 109/200
420/420 [==============================] - 0s 579us/step - loss: 0.0685 -
val_loss: 0.0713
Epoch 110/200
420/420 [==============================] - 0s 581us/step - loss: 0.0687 -
val_loss: 0.0713
Epoch 111/200
420/420 [==============================] - 0s 579us/step - loss: 0.0688 -
val_loss: 0.0722
Epoch 112/200
420/420 [==============================] - 0s 598us/step - loss: 0.0682 -
val_loss: 0.0715
Epoch 113/200
420/420 [==============================] - 0s 587us/step - loss: 0.0686 -
val_loss: 0.0709
Epoch 114/200
420/420 [==============================] - 0s 583us/step - loss: 0.0682 -
val_loss: 0.0728
Epoch 115/200
420/420 [==============================] - 0s 585us/step - loss: 0.0684 -
val_loss: 0.0709
Epoch 116/200
420/420 [==============================] - 0s 586us/step - loss: 0.0684 -
val_loss: 0.0735
Epoch 117/200
420/420 [==============================] - 0s 581us/step - loss: 0.0683 -
val_loss: 0.0725
Epoch 118/200
420/420 [==============================] - 0s 579us/step - loss: 0.0688 -
val_loss: 0.0712
Epoch 119/200
```

```
420/420 [==============================] - 0s 580us/step - loss: 0.0687 -
val_loss: 0.0727
Epoch 120/200
420/420 [==============================] - 0s 595us/step - loss: 0.0678 -
val_loss: 0.0709
Epoch 121/200
420/420 [==============================] - 0s 587us/step - loss: 0.0680 -
val_loss: 0.0735
Epoch 122/200
420/420 [==============================] - 0s 580us/step - loss: 0.0685 -
val_loss: 0.0713
Epoch 123/200
420/420 [==============================] - 0s 583us/step - loss: 0.0681 -
val_loss: 0.0744
Epoch 124/200
420/420 [==============================] - 0s 583us/step - loss: 0.0680 -
val_loss: 0.0709
Epoch 125/200
420/420 [==============================] - 0s 588us/step - loss: 0.0683 -
val_loss: 0.0713
Epoch 126/200
420/420 [==============================] - 0s 582us/step - loss: 0.0682 -
val_loss: 0.0716
Epoch 127/200
420/420 [==============================] - 0s 584us/step - loss: 0.0678 -
val_loss: 0.0835
Epoch 128/200
420/420 [==============================] - 0s 586us/step - loss: 0.0682 -
val_loss: 0.0721
Epoch 129/200
420/420 [==============================] - 0s 580us/step - loss: 0.0686 -
val_loss: 0.0712
Epoch 130/200
420/420 [==============================] - 0s 581us/step - loss: 0.0679 -
val_loss: 0.0712
Epoch 131/200
420/420 [==============================] - 0s 581us/step - loss: 0.0681 -
val_loss: 0.0753
Epoch 132/200
420/420 [==============================] - 0s 586us/step - loss: 0.0681 -
val_loss: 0.0708
Epoch 133/200
420/420 [==============================] - 0s 578us/step - loss: 0.0677 -
val_loss: 0.0707
Epoch 134/200
420/420 [==============================] - 0s 587us/step - loss: 0.0683 -
val_loss: 0.0714
Epoch 135/200
```

```
420/420 [==============================] - 0s 586us/step - loss: 0.0682 -
val_loss: 0.0722
Epoch 136/200
420/420 [==============================] - 0s 594us/step - loss: 0.0683 -
val_loss: 0.0708
Epoch 137/200
420/420 [==============================] - 0s 591us/step - loss: 0.0681 -
val_loss: 0.0706
Epoch 138/200
420/420 [==============================] - 0s 578us/step - loss: 0.0681 -
val_loss: 0.0712
Epoch 139/200
420/420 [==============================] - 0s 584us/step - loss: 0.0677 -
val_loss: 0.0715
Epoch 140/200
420/420 [==============================] - 0s 586us/step - loss: 0.0681 -
val_loss: 0.0708
Epoch 141/200
420/420 [==============================] - 0s 579us/step - loss: 0.0680 -
val_loss: 0.0707
Epoch 142/200
420/420 [==============================] - 0s 583us/step - loss: 0.0676 -
val_loss: 0.0715
Epoch 143/200
420/420 [==============================] - 0s 582us/step - loss: 0.0678 -
val_loss: 0.0717
Epoch 144/200
420/420 [==============================] - 0s 580us/step - loss: 0.0681 -
val_loss: 0.0710
Epoch 145/200
420/420 [==============================] - 0s 584us/step - loss: 0.0681 -
val_loss: 0.0705
Epoch 146/200
420/420 [==============================] - 0s 581us/step - loss: 0.0679 -
val_loss: 0.0710
Epoch 147/200
420/420 [==============================] - 0s 590us/step - loss: 0.0677 -
val_loss: 0.0710
Epoch 148/200
420/420 [==============================] - 0s 588us/step - loss: 0.0679 -
val_loss: 0.0708
Epoch 149/200
420/420 [==============================] - 0s 584us/step - loss: 0.0681 -
val_loss: 0.0703
Epoch 150/200
420/420 [==============================] - 0s 581us/step - loss: 0.0676 -
val_loss: 0.0713
Epoch 151/200
```

```
420/420 [==============================] - 0s 587us/step - loss: 0.0678 -
val_loss: 0.0706
Epoch 152/200
420/420 [==============================] - 0s 591us/step - loss: 0.0677 -
val_loss: 0.0707
Epoch 153/200
420/420 [==============================] - 0s 603us/step - loss: 0.0680 -
val_loss: 0.0703
Epoch 154/200
420/420 [==============================] - 0s 592us/step - loss: 0.0678 -
val_loss: 0.0715
Epoch 155/200
420/420 [==============================] - 0s 607us/step - loss: 0.0674 -
val_loss: 0.0723
Epoch 156/200
420/420 [==============================] - 0s 600us/step - loss: 0.0677 -
val_loss: 0.0703
Epoch 157/200
420/420 [==============================] - 0s 627us/step - loss: 0.0680 -
val_loss: 0.0706
Epoch 158/200
420/420 [==============================] - 0s 619us/step - loss: 0.0677 -
val_loss: 0.0702
Epoch 159/200
420/420 [==============================] - 0s 603us/step - loss: 0.0676 -
val_loss: 0.0709
Epoch 160/200
420/420 [==============================] - 0s 591us/step - loss: 0.0677 -
val_loss: 0.0707
Epoch 161/200
420/420 [==============================] - 0s 600us/step - loss: 0.0677 -
val_loss: 0.0706
Epoch 162/200
420/420 [==============================] - 0s 584us/step - loss: 0.0678 -
val_loss: 0.0722
Epoch 163/200
420/420 [==============================] - 0s 580us/step - loss: 0.0678 -
val_loss: 0.0705
Epoch 164/200
420/420 [==============================] - 0s 578us/step - loss: 0.0680 -
val_loss: 0.0749
Epoch 165/200
420/420 [==============================] - 0s 584us/step - loss: 0.0681 -
val_loss: 0.0710
Epoch 166/200
420/420 [==============================] - 0s 581us/step - loss: 0.0677 -
val_loss: 0.0709
Epoch 167/200
```

```
420/420 [==============================] - 0s 579us/step - loss: 0.0683 -
val_loss: 0.0704
Epoch 168/200
420/420 [==============================] - 0s 586us/step - loss: 0.0674 -
val_loss: 0.0705
Epoch 169/200
420/420 [==============================] - 0s 577us/step - loss: 0.0680 -
val_loss: 0.0708
Epoch 170/200
420/420 [==============================] - 0s 583us/step - loss: 0.0675 -
val_loss: 0.0705
Epoch 171/200
420/420 [==============================] - 0s 575us/step - loss: 0.0677 -
val_loss: 0.0707
Epoch 172/200
420/420 [==============================] - 0s 579us/step - loss: 0.0677 -
val_loss: 0.0709
Epoch 173/200
420/420 [==============================] - 0s 586us/step - loss: 0.0675 -
val_loss: 0.0707
Epoch 174/200
420/420 [==============================] - 0s 585us/step - loss: 0.0676 -
val_loss: 0.0720
Epoch 175/200
420/420 [==============================] - 0s 585us/step - loss: 0.0681 -
val_loss: 0.0711
Epoch 176/200
420/420 [==============================] - 0s 588us/step - loss: 0.0672 -
val_loss: 0.0711
Epoch 177/200
420/420 [==============================] - 0s 597us/step - loss: 0.0676 -
val_loss: 0.0711
Epoch 178/200
420/420 [==============================] - 0s 599us/step - loss: 0.0682 -
val_loss: 0.0701
Epoch 179/200
420/420 [==============================] - 0s 583us/step - loss: 0.0676 -
val_loss: 0.0707
Epoch 180/200
420/420 [==============================] - 0s 581us/step - loss: 0.0679 -
val_loss: 0.0709
Epoch 181/200
420/420 [==============================] - 0s 588us/step - loss: 0.0675 -
val_loss: 0.0738
Epoch 182/200
420/420 [==============================] - 0s 597us/step - loss: 0.0671 -
val_loss: 0.0712
Epoch 183/200
```

```
420/420 [==============================] - 0s 587us/step - loss: 0.0680 -
val_loss: 0.0723
Epoch 184/200
420/420 [==============================] - 0s 588us/step - loss: 0.0675 -
val_loss: 0.0706
Epoch 185/200
420/420 [==============================] - 0s 587us/step - loss: 0.0678 -
val_loss: 0.0704
Epoch 186/200
420/420 [==============================] - 0s 584us/step - loss: 0.0680 -
val_loss: 0.0732
Epoch 187/200
420/420 [==============================] - 0s 585us/step - loss: 0.0680 -
val_loss: 0.0710
Epoch 188/200
420/420 [==============================] - 0s 583us/step - loss: 0.0679 -
val_loss: 0.0703
Epoch 189/200
420/420 [==============================] - 0s 585us/step - loss: 0.0674 -
val_loss: 0.0707
Epoch 190/200
420/420 [==============================] - 0s 583us/step - loss: 0.0677 -
val_loss: 0.0750
Epoch 191/200
420/420 [==============================] - 0s 591us/step - loss: 0.0678 -
val_loss: 0.0703
Epoch 192/200
420/420 [==============================] - 0s 581us/step - loss: 0.0676 -
val_loss: 0.0703
Epoch 193/200
420/420 [==============================] - 0s 586us/step - loss: 0.0676 -
val_loss: 0.0728
Epoch 194/200
420/420 [==============================] - 0s 588us/step - loss: 0.0676 -
val_loss: 0.0717
Epoch 195/200
420/420 [==============================] - 0s 582us/step - loss: 0.0679 -
val_loss: 0.0718
Epoch 196/200
420/420 [==============================] - 0s 591us/step - loss: 0.0675 -
val_loss: 0.0712
Epoch 197/200
420/420 [==============================] - 0s 593us/step - loss: 0.0676 -
val_loss: 0.0704
Epoch 198/200
420/420 [==============================] - 0s 588us/step - loss: 0.0684 -
val_loss: 0.0706
Epoch 199/200
```

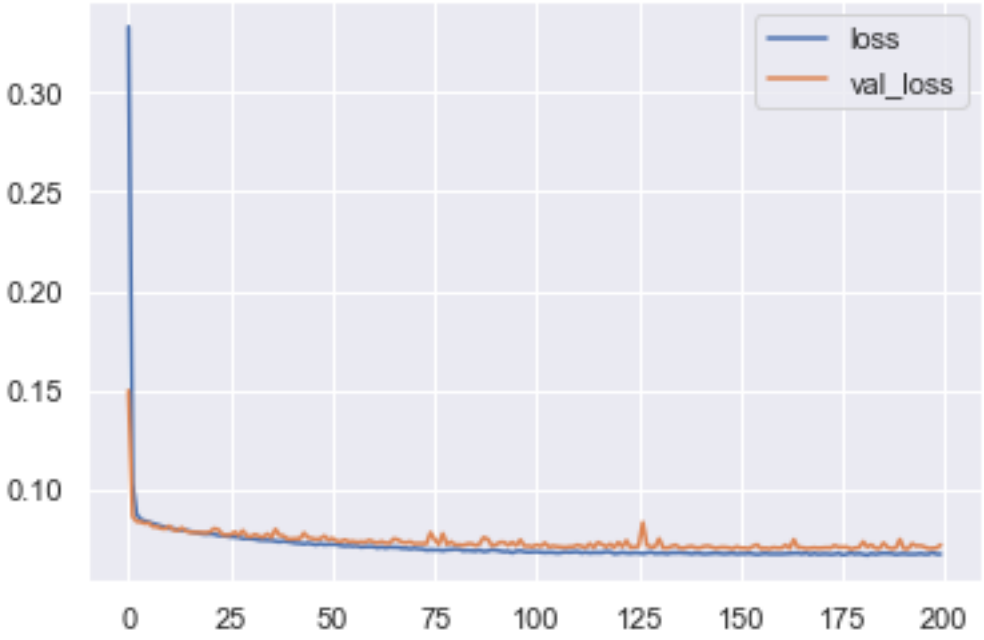
```
420/420 [==============================] - 0s 582us/step - loss: 0.0676 -
val_loss: 0.0707
Epoch 200/200
420/420 [==============================] - 0s 588us/step - loss: 0.0676 -
val_loss: 0.0722
```

[134]:
```
<tensorflow.python.keras.callbacks.History at 0x7fe8f7f80460>
```

1.4.3 Visualize the Loss Function

[135]:
```
losses = pd.DataFrame(model.history.history)
losses.plot()
```

[135]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8fc57ea00>
```



Our validation loss remains relatively stable with the actual loss, so overfitting is minimal.

1.4.4 Test the Model

[136]:
```
#y_pred = np.argmax(model.predict(X_test),axis=-1)
y_pred = model.predict_classes(X_test)
```

1.4.5 Model Evaluation

[137]:
```
confusion = confusion_matrix(y_test,y_pred)
print(f'CONFUSION MATRIX:
 ↪\n\n{confusion[0][0]}\t{confusion[0][1]}\n{confusion[1][0]}\t{confusion[1][1]}')
```

```
CONFUSION MATRIX:

4034    36
56      349
```

[139]:
```
print(f"CLASSIFICATION REPORT:\n\n{classification_report(y_test,y_pred)}")
```

```
CLASSIFICATION REPORT:

              precision    recall  f1-score   support

           0       0.99      0.99      0.99      4070
           1       0.91      0.86      0.88       405

    accuracy                           0.98      4475
   macro avg       0.95      0.93      0.94      4475
weighted avg       0.98      0.98      0.98      4475
```

1.5 Model Optimization

1.5.1 Iterating Through the Models

Let's experiment with the number of hidden layers in our network. We'll iterate from 2 to 50 hidden layers, each containing 8 units

[145]:
```
results = []
for i in range(2,51):
    model_loop = Sequential()

    # Input and hidden layers
    for j in range(0,(i+1)):
        model_loop.add(Dense(8,activation='relu'))

    # Output layer
    model_loop.add(Dense(1,activation='sigmoid'))

    # Compile layers
    model_loop.compile(loss='binary_crossentropy', optimizer='adam')

    # We will reduce epochs to 100 to reduce run time.
    # 100 was chosen based on previous loss function visualization.
    model_loop.fit(x=X_train,y=y_train.values,
```

```
        validation_data=(X_test,y_test.values),
        batch_size=128,epochs=100,verbose=0)

    # Model evaluation
    predictions_loop = model_loop.predict_classes(X_test)

    # Calculate statistics for each iteration
    confusion = confusion_matrix(y_test,predictions_loop)

    tp = confusion[1][1]
    tn = confusion[0][0]
    fp = confusion[0][1]
    fn = confusion[1][0]

    total = tp+tn+fp+fn
    accuracy = (tp+tn)/total
    precision = tp/(tp+fp)
    recall = tp/(tp+fn)
    f1_score = 2*precision*recall/(precision+recall)

    # Append results to the list for future reference
    results.append([i,accuracy,precision,recall,f1_score])

    # Print classification report after each iteration
    print(f"CLASSIFICATION REPORT FOR {i} HIDDEN LAYERS:
↪\n\n{classification_report(y_test,predictions_loop)}")
```

```
CLASSIFICATION REPORT FOR 2 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.93      0.83      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.93      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 3 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.94      0.83      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.93      4475
```

```
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 4 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.99      0.99      0.99      4070
           1       0.91      0.86      0.89       405

    accuracy                           0.98      4475
   macro avg       0.95      0.93      0.94      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 5 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.92      0.84      0.88       405

    accuracy                           0.98      4475
   macro avg       0.95      0.92      0.93      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 6 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.93      0.84      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.92      0.93      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 7 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      1.00      0.99      4070
           1       0.94      0.82      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.93      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 8 HIDDEN LAYERS:
```

	precision	recall	f1-score	support
0	0.98	0.99	0.99	4070
1	0.92	0.84	0.88	405
accuracy			0.98	4475
macro avg	0.95	0.92	0.93	4475
weighted avg	0.98	0.98	0.98	4475

CLASSIFICATION REPORT FOR 9 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.99	0.99	0.99	4070
1	0.91	0.86	0.89	405
accuracy			0.98	4475
macro avg	0.95	0.93	0.94	4475
weighted avg	0.98	0.98	0.98	4475

CLASSIFICATION REPORT FOR 10 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.98	0.99	0.99	4070
1	0.93	0.84	0.88	405
accuracy			0.98	4475
macro avg	0.96	0.92	0.94	4475
weighted avg	0.98	0.98	0.98	4475

CLASSIFICATION REPORT FOR 11 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.99	0.99	0.99	4070
1	0.91	0.86	0.88	405
accuracy			0.98	4475
macro avg	0.95	0.93	0.94	4475
weighted avg	0.98	0.98	0.98	4475

CLASSIFICATION REPORT FOR 12 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.98	0.99	0.99	4070
1	0.94	0.82	0.88	405

```
    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.93      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 13 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.94      0.83      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.94      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 14 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      1.00      0.99      4070
           1       0.95      0.82      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.94      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 15 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.93      0.82      0.87       405

    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.93      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 16 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.99      0.99      0.99      4070
           1       0.91      0.85      0.88       405

    accuracy                           0.98      4475
   macro avg       0.95      0.92      0.93      4475
weighted avg       0.98      0.98      0.98      4475
```

```
CLASSIFICATION REPORT FOR 17 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.93      0.83      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.93      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 18 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      1.00      0.99      4070
           1       0.95      0.82      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.91      0.93      4475
weighted avg       0.98      0.98      0.98      4475


<ipython-input-145-31c4091dd8c8>:34: RuntimeWarning: invalid value encountered
in long_scalars
  precision = tp/(tp+fp)
/opt/anaconda3/lib/python3.8/site-
packages/sklearn/metrics/_classification.py:1221: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
CLASSIFICATION REPORT FOR 19 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475

CLASSIFICATION REPORT FOR 20 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
```

```
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 21 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 22 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 23 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 24 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      4070
           1       0.92      0.86      0.89       405

    accuracy                           0.98      4475
   macro avg       0.95      0.92      0.94      4475
```

```
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 25 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.92      0.85      0.88       405

    accuracy                           0.98      4475
   macro avg       0.95      0.92      0.94      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 26 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.93      0.84      0.88       405

    accuracy                           0.98      4475
   macro avg       0.96      0.92      0.94      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 27 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.99      0.99      0.99      4070
           1       0.91      0.85      0.88       405

    accuracy                           0.98      4475
   macro avg       0.95      0.92      0.93      4475
weighted avg       0.98      0.98      0.98      4475

CLASSIFICATION REPORT FOR 28 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475

CLASSIFICATION REPORT FOR 29 HIDDEN LAYERS:
```

	precision	recall	f1-score	support
0	0.91	1.00	0.95	4070
1	0.00	0.00	0.00	405
accuracy			0.91	4475
macro avg	0.45	0.50	0.48	4475
weighted avg	0.83	0.91	0.87	4475

CLASSIFICATION REPORT FOR 30 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.99	0.99	0.99	4070
1	0.92	0.85	0.88	405
accuracy			0.98	4475
macro avg	0.95	0.92	0.94	4475
weighted avg	0.98	0.98	0.98	4475

CLASSIFICATION REPORT FOR 31 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.99	0.99	0.99	4070
1	0.90	0.87	0.88	405
accuracy			0.98	4475
macro avg	0.94	0.93	0.94	4475
weighted avg	0.98	0.98	0.98	4475

CLASSIFICATION REPORT FOR 32 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.98	1.00	0.99	4070
1	0.95	0.83	0.88	405
accuracy			0.98	4475
macro avg	0.96	0.91	0.94	4475
weighted avg	0.98	0.98	0.98	4475

CLASSIFICATION REPORT FOR 33 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.98	0.99	0.99	4070
1	0.94	0.81	0.87	405

```
    accuracy                           0.98      4475
   macro avg       0.96      0.90      0.93      4475
weighted avg       0.98      0.98      0.98      4475
```

CLASSIFICATION REPORT FOR 34 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 35 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      4070
           1       0.96      0.80      0.88       405

    accuracy                           0.98      4475
   macro avg       0.97      0.90      0.93      4475
weighted avg       0.98      0.98      0.98      4475
```

CLASSIFICATION REPORT FOR 36 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 37 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.92      0.85      0.88       405

    accuracy                           0.98      4475
   macro avg       0.95      0.92      0.94      4475
weighted avg       0.98      0.98      0.98      4475
```

CLASSIFICATION REPORT FOR 38 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.99	0.99	0.99	4070
1	0.93	0.85	0.89	405
accuracy			0.98	4475
macro avg	0.96	0.92	0.94	4475
weighted avg	0.98	0.98	0.98	4475

CLASSIFICATION REPORT FOR 39 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.91	1.00	0.95	4070
1	0.00	0.00	0.00	405
accuracy			0.91	4475
macro avg	0.45	0.50	0.48	4475
weighted avg	0.83	0.91	0.87	4475

CLASSIFICATION REPORT FOR 40 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.91	1.00	0.95	4070
1	0.00	0.00	0.00	405
accuracy			0.91	4475
macro avg	0.45	0.50	0.48	4475
weighted avg	0.83	0.91	0.87	4475

CLASSIFICATION REPORT FOR 41 HIDDEN LAYERS:

	precision	recall	f1-score	support
0	0.91	1.00	0.95	4070
1	0.00	0.00	0.00	405
accuracy			0.91	4475
macro avg	0.45	0.50	0.48	4475
weighted avg	0.83	0.91	0.87	4475

CLASSIFICATION REPORT FOR 42 HIDDEN LAYERS:

	precision	recall	f1-score	support

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4070
           1       0.92      0.85      0.88       405

    accuracy                           0.98      4475
   macro avg       0.95      0.92      0.94      4475
weighted avg       0.98      0.98      0.98      4475
```

CLASSIFICATION REPORT FOR 43 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 44 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 45 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

CLASSIFICATION REPORT FOR 46 HIDDEN LAYERS:

```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405
```

```
    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475

CLASSIFICATION REPORT FOR 47 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475

CLASSIFICATION REPORT FOR 48 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475

CLASSIFICATION REPORT FOR 49 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475

CLASSIFICATION REPORT FOR 50 HIDDEN LAYERS:

              precision    recall  f1-score   support

           0       0.91      1.00      0.95      4070
           1       0.00      0.00      0.00       405

    accuracy                           0.91      4475
   macro avg       0.45      0.50      0.48      4475
weighted avg       0.83      0.91      0.87      4475
```

1.5.2 Cleaning the Results

Display the results in a pandas dataframe

[152]:
```
results_df = pd.
↪DataFrame(columns=['Hidden_Layers','Accuracy','Precision','Recall','F1-Score'],data=results
results_df
```

[152]:

	Hidden_Layers	Accuracy	Precision	Recall	F1-Score
0	2	0.978771	0.928177	0.829630	0.876141
1	3	0.979665	0.936111	0.832099	0.881046
2	4	0.979888	0.913386	0.859259	0.885496
3	5	0.979218	0.921622	0.841975	0.880000
4	6	0.979218	0.926230	0.837037	0.879377
5	7	0.979218	0.943182	0.819753	0.877147
6	8	0.978994	0.921409	0.839506	0.878553
7	9	0.980112	0.911458	0.864198	0.887199
8	10	0.979888	0.929155	0.841975	0.883420
9	11	0.979218	0.906250	0.859259	0.882129
10	12	0.979218	0.940678	0.822222	0.877470
11	13	0.979888	0.936288	0.834568	0.882507
12	14	0.979888	0.946176	0.824691	0.881266
13	15	0.978547	0.930362	0.824691	0.874346
14	16	0.978994	0.912467	0.849383	0.879795
15	17	0.978771	0.928177	0.829630	0.876141
16	18	0.979665	0.946023	0.822222	0.879789
17	19	0.909497	NaN	0.000000	NaN
18	20	0.909497	NaN	0.000000	NaN
19	21	0.909497	NaN	0.000000	NaN
20	22	0.909497	NaN	0.000000	NaN
21	23	0.909497	NaN	0.000000	NaN
22	24	0.980112	0.917989	0.856790	0.886335
23	25	0.979441	0.919571	0.846914	0.881748
24	26	0.979665	0.931319	0.837037	0.881664
25	27	0.979218	0.914894	0.849383	0.880922
26	28	0.909497	NaN	0.000000	NaN
27	29	0.909497	NaN	0.000000	NaN
28	30	0.979665	0.917553	0.851852	0.883483
29	31	0.979441	0.900256	0.869136	0.884422
30	32	0.980335	0.946479	0.829630	0.884211
31	33	0.978324	0.937500	0.814815	0.871863
32	34	0.909497	NaN	0.000000	NaN
33	35	0.979218	0.958824	0.804938	0.875168
34	36	0.909497	NaN	0.000000	NaN
35	37	0.979888	0.924528	0.846914	0.884021
36	38	0.981006	0.932432	0.851852	0.890323
37	39	0.909497	NaN	0.000000	NaN
38	40	0.909497	NaN	0.000000	NaN

39	41	0.909497	NaN	0.000000	NaN
40	42	0.979441	0.919571	0.846914	0.881748
41	43	0.909497	NaN	0.000000	NaN
42	44	0.909497	NaN	0.000000	NaN
43	45	0.909497	NaN	0.000000	NaN
44	46	0.909497	NaN	0.000000	NaN
45	47	0.909497	NaN	0.000000	NaN
46	48	0.909497	NaN	0.000000	NaN
47	49	0.909497	NaN	0.000000	NaN
48	50	0.909497	NaN	0.000000	NaN

There appear to be many null values. Let's remove them.

[155]:
```
cleaned_results = results_df.dropna()
cleaned_results
```

[155]:

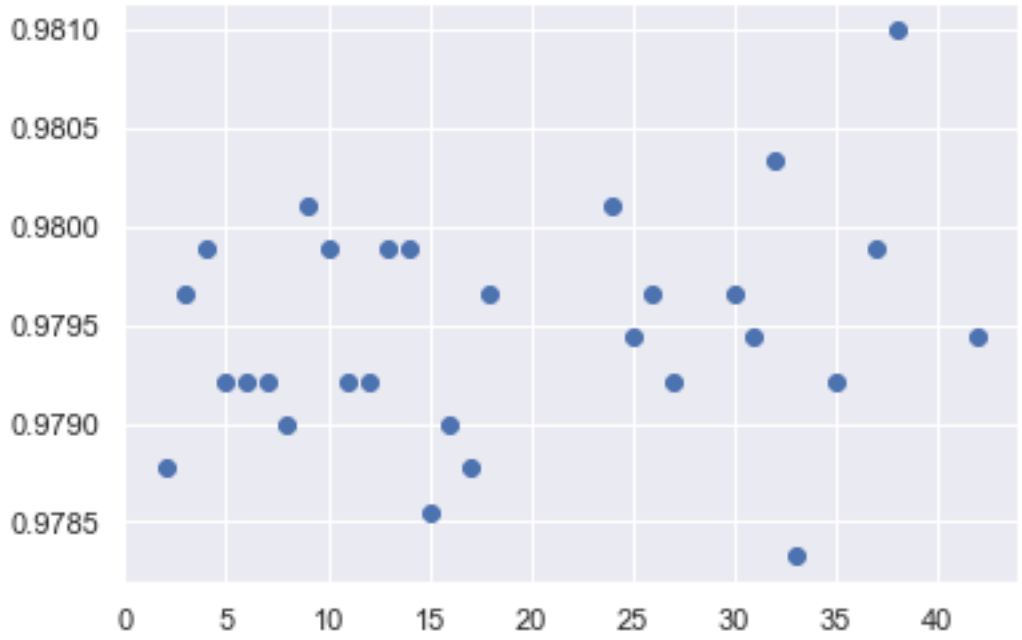
	Hidden_Layers	Accuracy	Precision	Recall	F1-Score
0	2	0.978771	0.928177	0.829630	0.876141
1	3	0.979665	0.936111	0.832099	0.881046
2	4	0.979888	0.913386	0.859259	0.885496
3	5	0.979218	0.921622	0.841975	0.880000
4	6	0.979218	0.926230	0.837037	0.879377
5	7	0.979218	0.943182	0.819753	0.877147
6	8	0.978994	0.921409	0.839506	0.878553
7	9	0.980112	0.911458	0.864198	0.887199
8	10	0.979888	0.929155	0.841975	0.883420
9	11	0.979218	0.906250	0.859259	0.882129
10	12	0.979218	0.940678	0.822222	0.877470
11	13	0.979888	0.936288	0.834568	0.882507
12	14	0.979888	0.946176	0.824691	0.881266
13	15	0.978547	0.930362	0.824691	0.874346
14	16	0.978994	0.912467	0.849383	0.879795
15	17	0.978771	0.928177	0.829630	0.876141
16	18	0.979665	0.946023	0.822222	0.879789
22	24	0.980112	0.917989	0.856790	0.886335
23	25	0.979441	0.919571	0.846914	0.881748
24	26	0.979665	0.931319	0.837037	0.881664
25	27	0.979218	0.914894	0.849383	0.880922
28	30	0.979665	0.917553	0.851852	0.883483
29	31	0.979441	0.900256	0.869136	0.884422
30	32	0.980335	0.946479	0.829630	0.884211
31	33	0.978324	0.937500	0.814815	0.871863
33	35	0.979218	0.958824	0.804938	0.875168
35	37	0.979888	0.924528	0.846914	0.884021
36	38	0.981006	0.932432	0.851852	0.890323
40	42	0.979441	0.919571	0.846914	0.881748

1.5.3 Visualizing the Iterations

Let's visualize the performance by number of hidden layers.

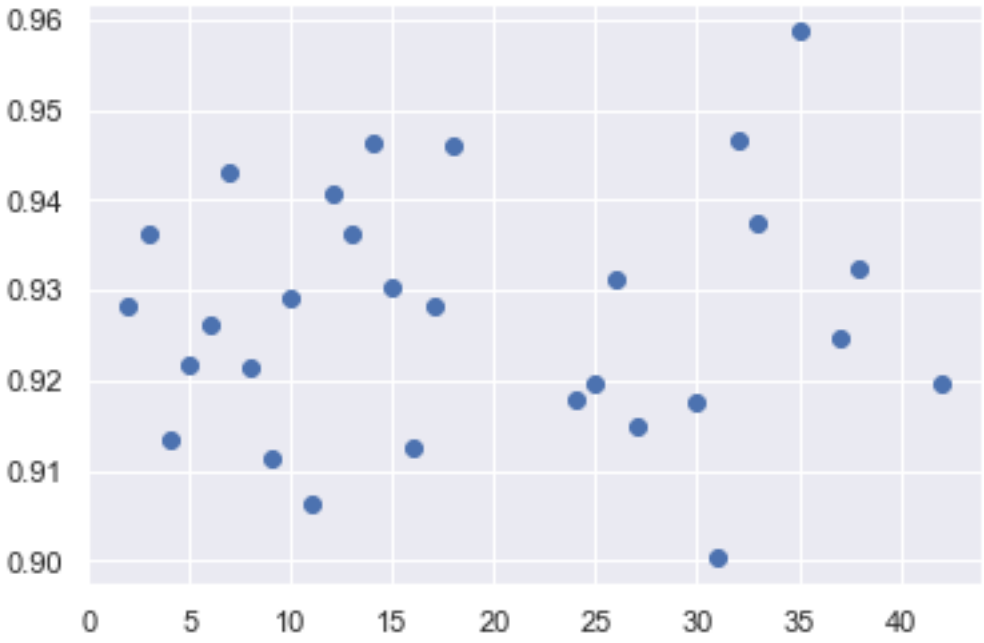
[161]:
```
# Visualizing Accuracy
plt.scatter(x=cleaned_results['Hidden_Layers'],y=cleaned_results['Accuracy'])
```

[161]: <matplotlib.collections.PathCollection at 0x7fe8e9781340>



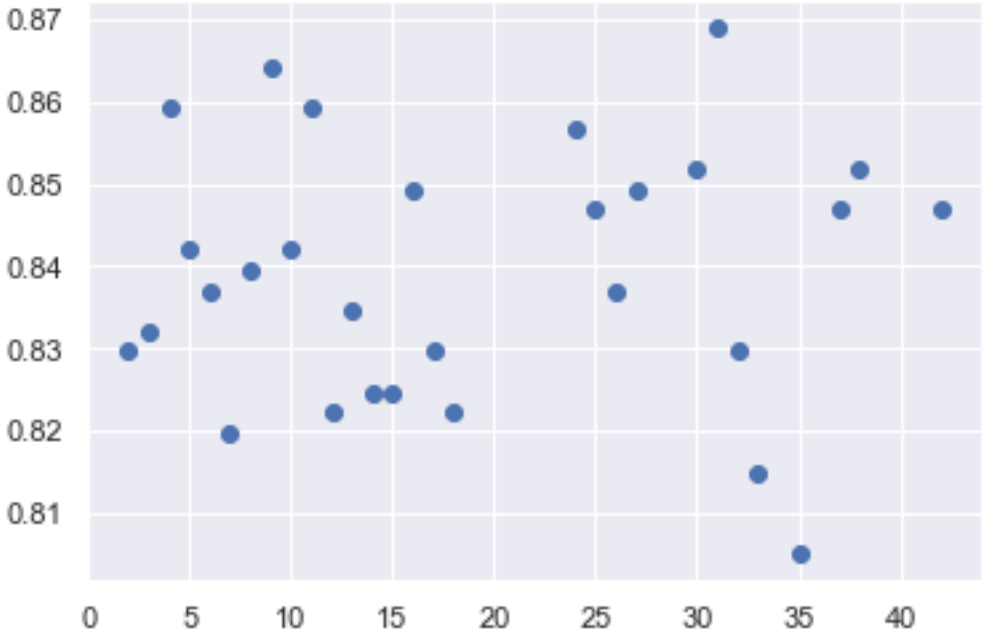
[163]:
```
# Visualizing Precision
plt.scatter(x=cleaned_results['Hidden_Layers'],y=cleaned_results['Precision'])
```

[163]: <matplotlib.collections.PathCollection at 0x7fe8e0483760>



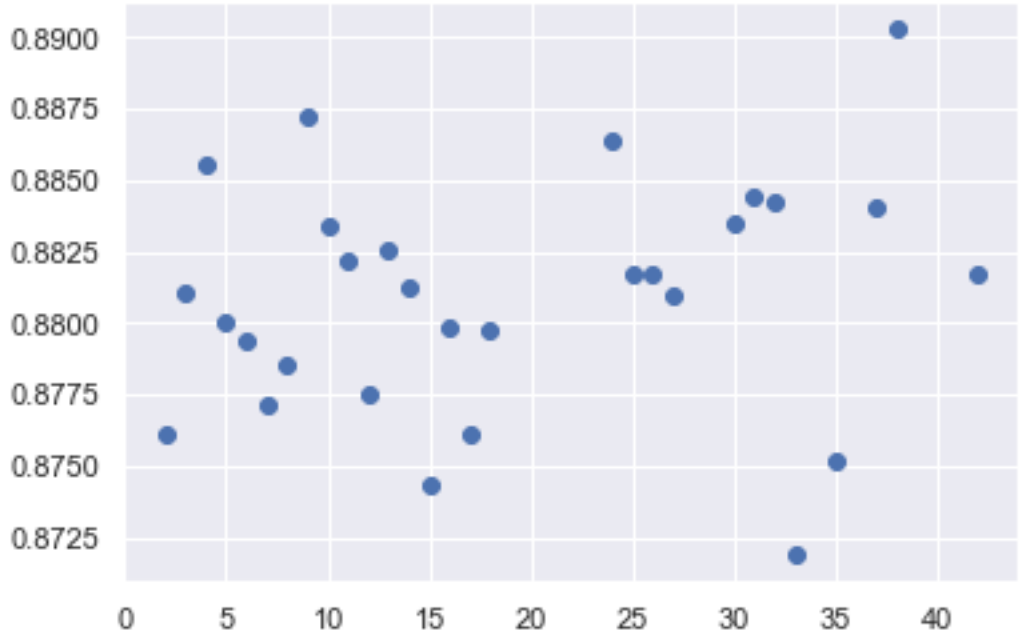
[164]:
```
# Visualizing Recall
plt.scatter(x=cleaned_results['Hidden_Layers'],y=cleaned_results['Recall'])
```

[164]: <matplotlib.collections.PathCollection at 0x7fe8fe99a130>



[165]:
```
# Visualizing F1-Score
plt.scatter(x=cleaned_results['Hidden_Layers'],y=cleaned_results['F1-Score'])
```

[165]:
```
<matplotlib.collections.PathCollection at 0x7fe8fe936850>
```



1.5.4 Analyzing the Results

[162]:
```
cleaned_results.describe()
```

[162]:

	Hidden_Layers	Accuracy	Precision	Recall	F1-Score
count	29.000000	29.000000	29.000000	29.000000	29.000000
mean	18.965517	0.979480	0.927520	0.839251	0.880956
std	12.040060	0.000567	0.013611	0.015649	0.004102
min	2.000000	0.978324	0.900256	0.804938	0.871863
25%	9.000000	0.979218	0.917989	0.829630	0.878553
50%	16.000000	0.979441	0.928177	0.839506	0.881266
75%	30.000000	0.979888	0.936288	0.849383	0.883483
max	42.000000	0.981006	0.958824	0.869136	0.890323

[181]:
```
# Display Top 5 Models by Accuracy
cleaned_results.sort_values(by='Accuracy',ascending=False).
↪drop(['Precision','Recall','F1-Score'],axis=1)[:5]
```

[181]:

	Hidden_Layers	Accuracy
36	38	0.981006

```
30              32  0.980335
7                9  0.980112
22              24  0.980112
11              13  0.979888
```

[178]:
```
# Display Top 5 Models by Precision
cleaned_results.sort_values(by='Precision',ascending=False).
↪drop(['Accuracy','Recall','F1-Score'],axis=1)[:5]
```

[178]:
```
    Hidden_Layers  Precision
33              35   0.958824
30              32   0.946479
12              14   0.946176
16              18   0.946023
5                7   0.943182
```

[183]:
```
# Display Top 5 Models by Recall
cleaned_results.sort_values(by='Recall',ascending=False).
↪drop(['Accuracy','Precision','F1-Score'],axis=1)[:5]
```

[183]:
```
    Hidden_Layers    Recall
29              31  0.869136
7                9  0.864198
2                4  0.859259
9               11  0.859259
22              24  0.856790
```

[182]:
```
# Display Top 5 Models by F1-Score
cleaned_results.sort_values(by='F1-Score',ascending=False).
↪drop(['Accuracy','Precision','Recall'],axis=1)[:5]
```

[182]:
```
    Hidden_Layers  F1-Score
36              38  0.890323
7                9  0.887199
22              24  0.886335
2                4  0.885496
29              31  0.884422
```

1.6 Conclusions

The best performing models for each category below (number of hidden layers in parentheses) were: * Accuracy = 0.981 from Model(38) * Precision = 0.959 from Model(35) * Recall = 0.869 from Model(31) * F1-Score = 0.890 from Model(38)

The objective of this project is to find a model that can correctly identify pulsars, which account for only approximately 10% of the dataset. Therefore, the most important metric is recall for this exercise. The model with the best recall had 31 hidden layers. Its evaluation metrics are saved below for future comparison with the other models:

[1]:
```
with open("2020_1127_DNN_Results.csv","w") as file:
    file.write('Model,Accuracy,Precision,Recall,F1-Score\n')
    file.write('DNN,0.98,0.90,0.87,0.88\n')
```