Case Study #4 - Neural Nets

1. Upload, explore, clean, and preprocess data for neural network modeling.
   a. Create a boston_df data frame by uploading the original data set into Python. Determine and present in this report the data frame dimensions.

```
try:
    boston_df = pd.read_csv('BostonHousing.csv')
except:
    print("BostonHousing.csv is not in the present working directory.")
```

```
print(f"The dimensions of the Boston Housing dataset is {boston_df.shape}", f"where there are {bost
```

```
The dimensions of the Boston Housing dataset is (506, 14) where there are 506 rows and 14 columns.
```

   b. Display modified column names

```
boston_df.columns = boston_df.columns.str.replace(' ', '_')
boston_df.columns
```

```
Index(['CRIME', 'ZONE', 'INDUST', 'NIT_OXIDE', 'ROOMS', 'AGE', 'DISTANCE',
       'RADIAL', 'TAX', 'ST_RATIO', 'LOW_STAT', 'MVALUE', 'CHAR_RIV_Y',
       'C_MVALUE_Yes'],
      dtype='object')
```

   c. Create dummies

```
CRIME         float64
ZONE          float64
INDUST        float64
CHAR_RIV       object
NIT_OXIDE     float64
ROOMS         float64
AGE           float64
DISTANCE      float64
RADIAL          int64
TAX             int64
ST_RATIO      float64
LOW_STAT      float64
MVALUE        float64
C_MVALUE       object
dtype: object
```

	CRIME	ZONE	INDUST	NIT_OXIDE	ROOMS	AGE	DISTANCE	RADIAL	TAX	ST_RATIO	LOW_STAT	MVALUE	CHAR_RIV_Y	C_MVALUE_Yes
0	0.00632	18.0	2.31	0.538	6.575	65.2	4.0900	1	296	15.3	4.98	24.0	0	0
1	0.02731	0.0	7.07	0.469	6.421	78.9	4.9671	2	242	17.8	9.14	21.6	0	0
2	0.02729	0.0	7.07	0.469	7.185	61.1	4.9671	2	242	17.8	4.03	34.7	0	1
3	0.03237	0.0	2.18	0.458	6.998	45.8	6.0622	3	222	18.7	2.94	33.4	0	1
4	0.06905	0.0	2.18	0.458	7.147	54.2	6.0622	3	222	18.7	5.33	36.2	0	1

```
Index(['CRIME', 'ZONE', 'INDUST', 'NIT_OXIDE', 'ROOMS', 'AGE', 'DISTANCE',
       'RADIAL', 'TAX', 'ST_RATIO', 'LOW_STAT', 'MVALUE', 'CHAR_RIV_Y',
       'C_MVALUE_Yes'],
      dtype='object')
```

2. Develop a neural network model for Boston Housing and use it for predictions.
   a. First five records of the training partition

```
train_X.head(5)
```

	CRIME	ZONE	INDUST	NIT_OXIDE	ROOMS	AGE	DISTANCE	RADIAL	TAX	ST_RATIO	LOW_STAT	CHAR_RIV_Y	C_MVALUE_Yes
13	0.62976	0.0	8.14	0.538	5.949	61.8	4.7075	4	307	21.0	8.26	0	0
61	0.17171	25.0	5.13	0.453	5.966	93.4	6.8185	8	284	19.7	14.44	0	0
377	9.82349	0.0	18.10	0.671	6.794	98.8	1.3580	24	666	20.2	21.24	0	0
39	0.02763	75.0	2.95	0.428	6.595	21.8	5.4011	3	252	18.3	4.32	0	1
365	4.55587	0.0	18.10	0.718	3.561	87.9	1.6132	24	666	20.2	7.12	0	0

First five records of the training partition after scaling

```
train_X_sc_df.head()
```

	CRIME	ZONE	INDUST	NIT_OXIDE	ROOMS	AGE	DISTANCE	RADIAL	TAX	ST_RATIO	LOW_STAT	CHAR_RIV_Y	C_MVALUE_Yes
0	-0.366	-0.484	-0.462	-0.147	-0.440	-0.251	0.412	-0.646	-0.600	1.189	-0.647	-0.304	-0.452
1	-0.420	0.580	-0.902	-0.868	-0.416	0.868	1.401	-0.191	-0.736	0.582	0.203	-0.304	-0.452
2	0.714	-0.484	0.992	0.982	0.782	1.060	-1.157	1.629	1.512	0.816	1.139	-0.304	-0.452
3	-0.436	2.708	-1.220	-1.080	0.494	-1.668	0.737	-0.760	-0.924	-0.070	-1.189	-0.304	2.214
4	0.095	-0.484	0.992	1.381	-3.893	0.673	-1.037	1.629	1.512	0.816	-0.804	-0.304	-0.452

Standardization of a dataset is a common requirement for many machine learning estimators since they might behave badly if the individual features do not more or less look like standard normally distributed data.
The calculation that is made when using the standard scaler is as follows:
z = (x - u) / s
where u is the mean of the training samples and s is the standard deviation of the training samples.

b. The final values of intercepts in the first array represent the coefficients of each of the hidden layers. The final values of intercepts in the second array represent the coefficient of the output node. The values of weights in the first array represent the weights that point from each of the input nodes (13 features = 13 lists of weights) to the hidden nodes. The values in the second array represent the weights that point to the output node from each of the hidden nodes..

```
Final Intercepts for Boston Housing Neural Network Model
[array([ 0.03419315, -5.17494472, -3.19741419,  0.09979904, -1.52105762,
       -1.35938186, -0.98147659,  0.20502791,  3.90980501,  0.05306107]), array([1.89670365])]

Network Weights for Boston Housing Neural Network Model
[array([[ 6.78417419e-01,  1.40009172e+00, -2.90503954e-01,
         7.00785971e-01, -2.33128946e+00, -1.88707724e-01,
        -1.10882033e+00, -3.48212572e-01, -4.05340615e-02,
         2.81750728e-01],
       [ 9.13742181e-01,  1.17283170e-01,  4.63067759e-01,
         1.60207856e+00, -2.70619848e+00, -4.00493501e-02,
        -4.56463641e-01, -9.63440430e-01,  3.37150294e-01,
         1.85719544e+00],
       [ 1.15795214e+00,  8.58982703e-01, -1.15996602e-02,
        -4.76552881e-01,  2.67668766e-01,  8.13217035e-01,
        -4.95596540e-01, -2.31666290e-01,  1.33624317e-01,
        -1.54542128e+00],
       [ 1.32957065e+00,  6.32224008e-01, -5.34757482e-01,
        -3.11797925e+00,  1.98344263e+00,  1.98515159e-01,
         3.53150071e-01,  1.57389121e+00, -3.11587045e+00,
         8.53350866e-01],
       [ 2.59846306e+00,  7.65059220e-01, -3.42193107e-01,
        -8.88413425e-01,  2.97602894e-01,  1.09954270e+00,
        -5.67326502e-01, -3.47552978e-02, -2.18158702e-03,
         5.73247359e-01],
       [-1.02904031e+00, -1.39976705e-01, -8.34613188e-01,
        -1.04880999e+00, -4.94132403e-01, -2.94284492e-01,
         4.45657025e-01,  1.09623048e+00, -8.39853433e-01,
        -7.52867827e-01],
       [-1.43624123e+00, -3.98134531e-01, -7.63247169e-01,
         1.69432537e-01,  2.76172744e-01,  1.75199651e-01,
        -1.60496896e+00,  1.72223308e+00, -7.66136145e-01,
         8.86732031e-01],
       [ 5.16577863e-01, -1.41104072e+00, -3.68636450e-02,
        -9.27011597e-01, -1.16521603e-01,  5.18275115e-01,
         1.51939793e+00, -1.06601021e+00,  1.10589582e+00,
        -1.97841803e+00],
       [ 3.50368825e-01, -2.49796818e+00, -9.58338047e-01,
         1.18243411e+00,  1.88740007e-01,  2.28376679e+00,
         1.36159036e+00,  1.91341090e+00, -7.99699604e-01,
         2.59782916e+00],
       [ 2.87911289e-01, -5.67029528e-01, -8.23922769e-01,
        -1.71974958e-01, -3.49289893e-01, -1.79011198e-01,
         2.14643943e+00,  1.06856011e+00, -8.84082915e-01,
         1.48873655e+00],
       [-7.37997633e-01, -2.69402262e+00,  1.45608445e+00,
        -3.18516362e-01,  2.19259387e-01,  3.36744109e-01,
         8.57544062e-01, -1.00543762e+00, -3.56671373e-01,
        -3.99033793e-01],
       [-1.18412015e+00, -2.01945152e-01, -2.26204133e-01,
        -2.28686545e-01, -1.20247666e+00, -7.17183945e-01,
        -1.05780495e+00,  9.20547648e-01, -9.53629049e-02,
        -6.57145473e-01],
       [-8.53530575e-01, -1.32729949e-01,  2.19825046e+00,
         1.78294870e-01, -1.45586826e-02, -1.85926303e+00,
         1.72367387e+00,  3.31674658e+00, -1.74981202e+00,
         2.44684662e-02]]), array([[ 1.40173896],
       [ 4.93294787],
       [ 3.70829147],
       [-2.81677763],
       [ 2.01379349],
       [-2.580327  ],
       [ 1.56837554],
       [ 2.69568461],
       [ 3.65371122],
       [ 2.02015303]])]
```

c. Five validation records that contain actual and predicted median prices (MVALUE), and their residuals.

```
Predictions for House Price for Validation Partition
     Actual  Prediction  Residual
307    28.2       29.63     -1.43
343    23.9       25.81     -1.91
47     16.6       20.65     -4.05
67     22.0       20.63      1.37
362    20.8       24.47     -3.67
```

d. The significant difference between the RMSE and MAPE values for the training and validation partitions suggests that the model is likely overfitting to the training data. Being that this is the case, I would not recommend using this neural network for predictions.

```
Accuracy Measures for Training Partition for Neural Network

Regression statistics

                      Mean Error (ME) : -0.0033
       Root Mean Squared Error (RMSE) : 1.5851
            Mean Absolute Error (MAE) : 1.1342
          Mean Percentage Error (MPE) : -0.9031
Mean Absolute Percentage Error (MAPE) : 6.1132
```

```
Accuracy Measures for Validation Partition for Neural Network

Regression statistics

                      Mean Error (ME) : -0.5680
       Root Mean Squared Error (RMSE) : 3.9407
            Mean Absolute Error (MAE) : 2.7470
          Mean Percentage Error (MPE) : -5.4903
Mean Absolute Percentage Error (MAPE) : 14.5074
```

3. Develop an improved neural network model with grid search.

a. Best score and best parameter value from GridSearchCV

```
Best score:0.8877
Best parameter:  {'hidden_layer_sizes': 2}
```

b. The final intercepts and network weights of the improved neural network model.

```
Final Intercepts for Boston Housing Neural Network Model
[array([-5.8049252 ,  9.24593197]), array([6.40051439])]

Network Weights for Boston Housing Neural Network Model
[array([[-0.30389718, -1.13481572],
       [-0.82423068,  0.01522733],
       [ 3.41753368, -0.19618274],
       [-0.8772186 , -0.35023142],
       [-1.38317186,  2.43543214],
       [ 0.02280384, -0.99890554],
       [-0.33525388, -1.02588688],
       [ 3.41626164, -0.42845199],
       [ 1.70241347, -1.51513226],
       [-1.32750925, -0.71321435],
       [-1.27314847, -0.4747823 ],
       [ 0.23793571,  0.12117713],
       [ 3.18966496,  1.48465918]]), array([[2.28575643],
       [1.50263471]])]
```

c. The RMSE and MAPE values for the training and validation partitions are relatively close, indicating that the model is not severely overfitting to the training data compared to the previous model. I would recommend the use of this neural network.

```
Accuracy Measures for Training Partition for Neural Network

Regression statistics

                      Mean Error (ME) : -0.0001
       Root Mean Squared Error (RMSE) : 2.6987
            Mean Absolute Error (MAE) : 2.0674
          Mean Percentage Error (MPE) : -1.8526
Mean Absolute Percentage Error (MAPE) : 10.6337
```

```
Accuracy Measures for Validation Partition for Neural Network

Regression statistics

                      Mean Error (ME) : 0.1367
       Root Mean Squared Error (RMSE) : 3.0185
            Mean Absolute Error (MAE) : 2.2846
          Mean Percentage Error (MPE) : -2.7484
Mean Absolute Percentage Error (MAPE) : 12.1011
```

d. When comparing the optimized neural net to the multiple linear regression models that use predictors derived from backward elimination, the neural network outperforms the backwards elimination model in both RMSE and MAPE accuracy scores. Because of this I would prefer to use the neural network for predictions.

```
Accuracy Measures for Validation Set - Backward Elimination

Regression statistics

                    Mean Error (ME) : 0.3854
       Root Mean Squared Error (RMSE) : 3.7318
           Mean Absolute Error (MAE) : 2.7591
         Mean Percentage Error (MPE) : -2.8698
Mean Absolute Percentage Error (MAPE) : 13.9371
```

```
Accuracy Measures for Validation Partition for Neural Network

Regression statistics

                    Mean Error (ME) : 0.1367
       Root Mean Squared Error (RMSE) : 3.0185
           Mean Absolute Error (MAE) : 2.2846
         Mean Percentage Error (MPE) : -2.7484
Mean Absolute Percentage Error (MAPE) : 12.1011
```