**Project Report**

**Neural Network Model on Corrosion Resistance**

**By**

Ms Shrishti Hore

**Under the Guidance of**

Mr Akshansh Mishra

**Contents**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **Sr.**  **No.** | **Topic** | **Page No.** | | 1. | Introduction | 1 -1 | | 2. | Procedure 1, Observation1, Outputs 1 | 2-9 | | 3. | Procedure 2, Observation2, Outputs 2 | 10-18 | | 4. | Procedure 3, Observation3, Outputs 3 | 19-54 | | 5. | Conclusion | 55-55 | |  |  |

**Introduction**

The main aim of this project was to create an artificial neural network architecture to predict corrosion resistance from the dataset given to us. This is also to fine the mean squared error and mean absolute percentage error of the given model which will help to analyze the losses and efficiency of the model. The project contains various stages of execution as follow :

1. Analyzing the given datasets using various data visualization techniques.
2. Data cleaning and making specified datasets as per requirements.
3. Creating a neural network for predicting corrosion resistance.
4. Training and testing the models.
5. Checking the mean square error and mean absolute percentage error and visualizing them.
6. Plotting the 3D surface plots, 3D contour plots and tri-surface plots for the respective attributes.

**Procedure**

**1. Data Analysis and Visualization:**

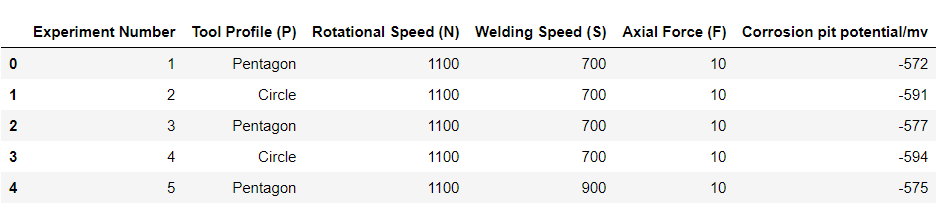
The analysis was carried out from the dataset corrosion\_data\_new.csv. This was essential to understand the parameters for plots and neural network architecture. The algorithm is as follows :

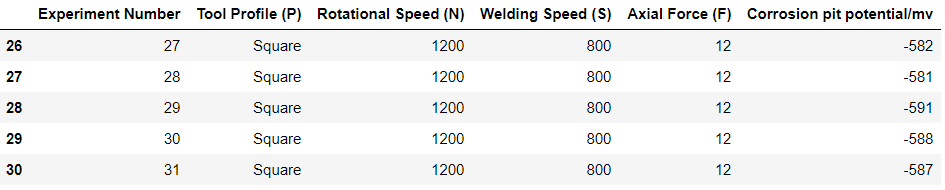
* Importing the required libraries i.e. numpy, pandas, matplotlib, os and seaborn
* Import the dataset corrosion\_data\_new.csv
* Print the head(1st 5 rows) and tail(last 5 rows) of the data
* Print the unique values of corrosion resistance
* Use seaborn to visualize a countplot for corrosion resistance setting alpha to 0.7
* Use seaborn to plot individual countplots for tool profile, rotational speed, welding speed, axial force and corrosion pit per mv
* Use seaborn to visualize a heatmap consisting of all the attributes and set annot as true
* Use seaborn to visualize barplots for every attribute to check the highest and lowest values of corrosion potential pit/mv attribute w.r.t 31 experiments

**Observation**

* There are four main attributes in the dataset Tool Profile(P), Rotational Speed(N), Welding Speed(S) and Axial force(F). Corrosion pit potential/mv (cpp) is the output or rather analyzing factor of the entire dataset for all 31 experiments
* -551 and -618 are the highest and lowest values of cpp respectively
* Similarly there are 8 pentagon, 8 circle, 1 hexagon, 1 triangle and 13 square samples
* Hexagon has the highest rotational speed count of 1200 rpm followed by circle with 1100 rpm, triangle with 1300 rpm, pentagon 1000 rpm and lowest square 1400 rpm (13,11,9,1,1)
* Welding speed count from highest to lowest is as hexagon 800mm/min, triangle 900 mm/min, circle 700 mm/min, pentagon 600mm/min and square 1000mm/min (13,8,8,1,1)
* Axial force count from highest to lowest is as hexagon 12KN, triangle 14KN, circle 10 KN, square 16KN and pentagon 8KN(13,8,8,1,1)
* -591cpp , -588 cpp, -587 cpp, -558 cpp are the highest counts at range 1.75 to 2.00

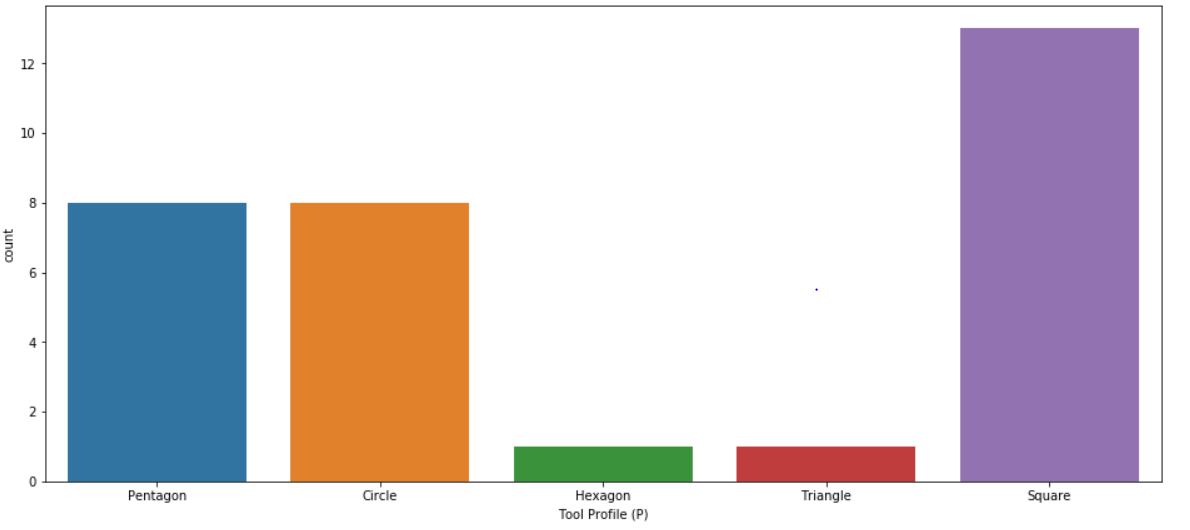
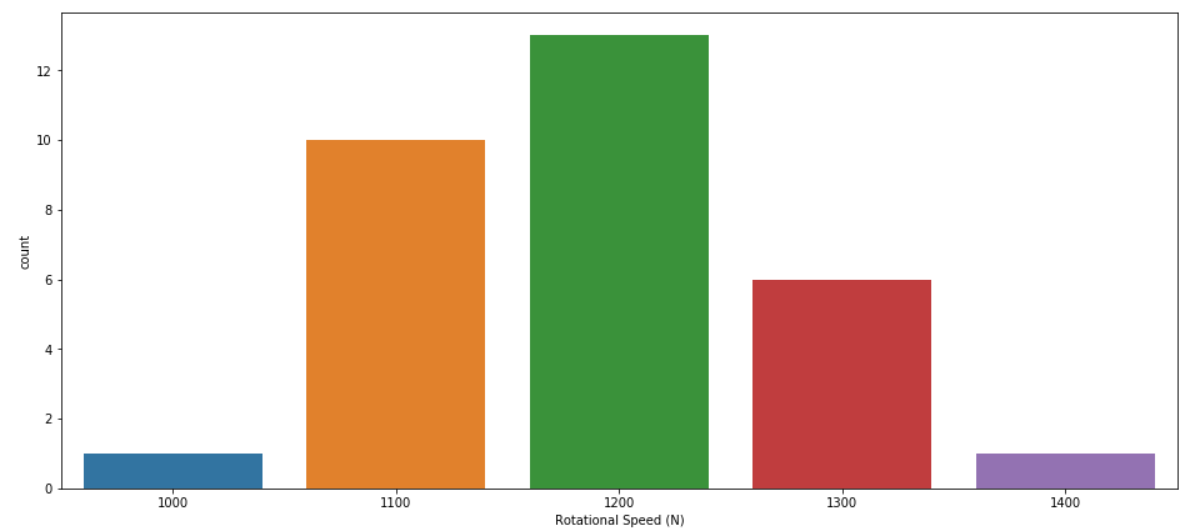
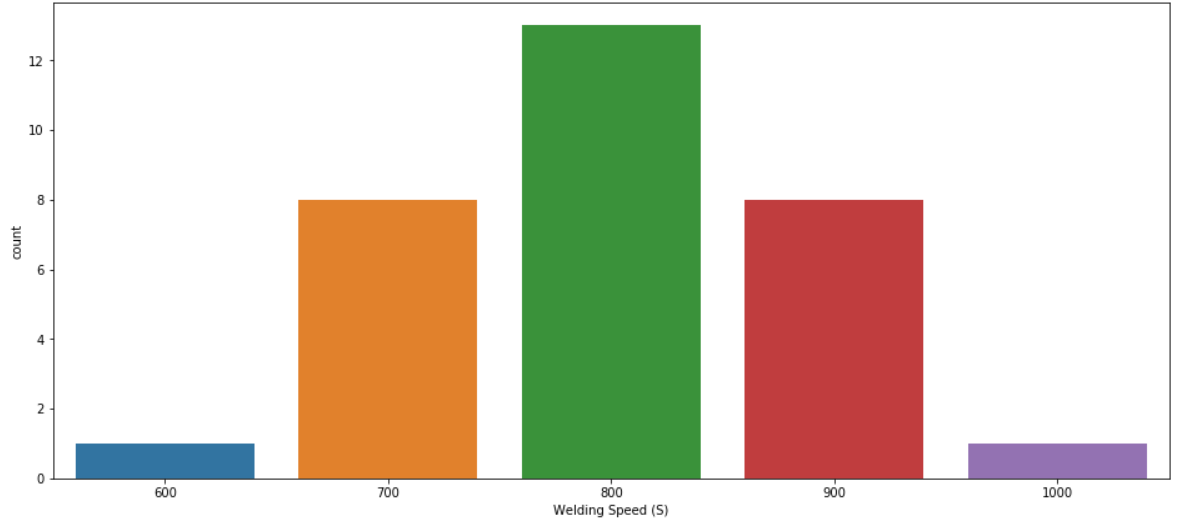
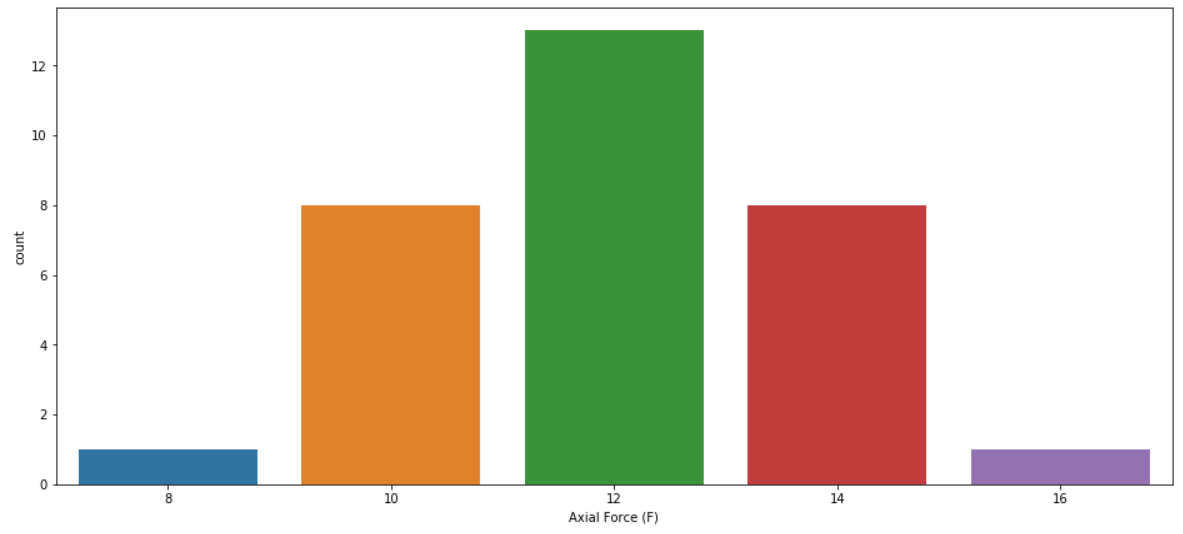
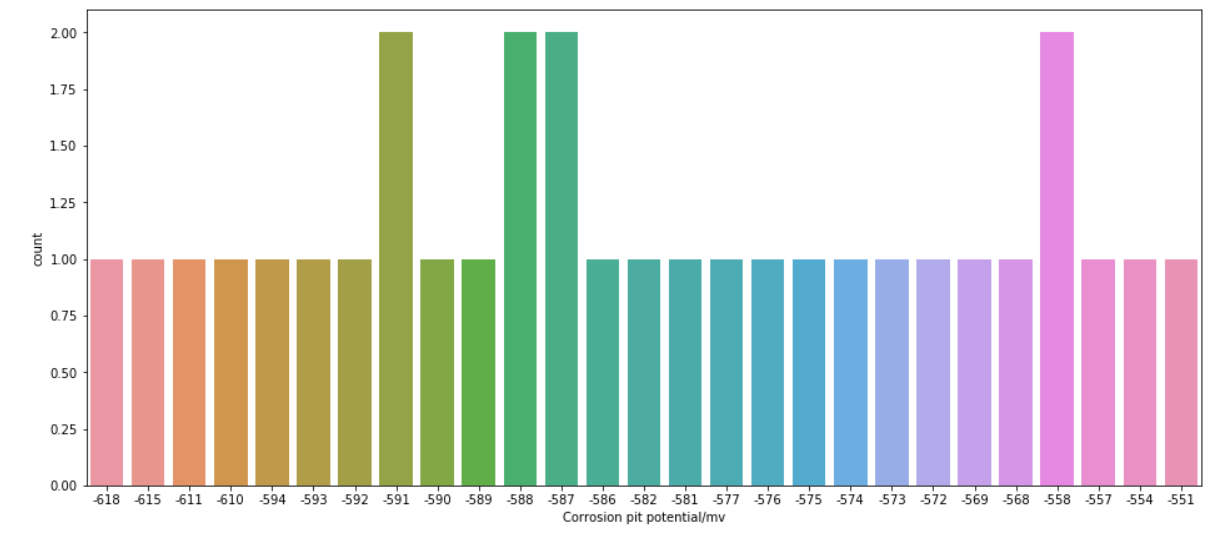
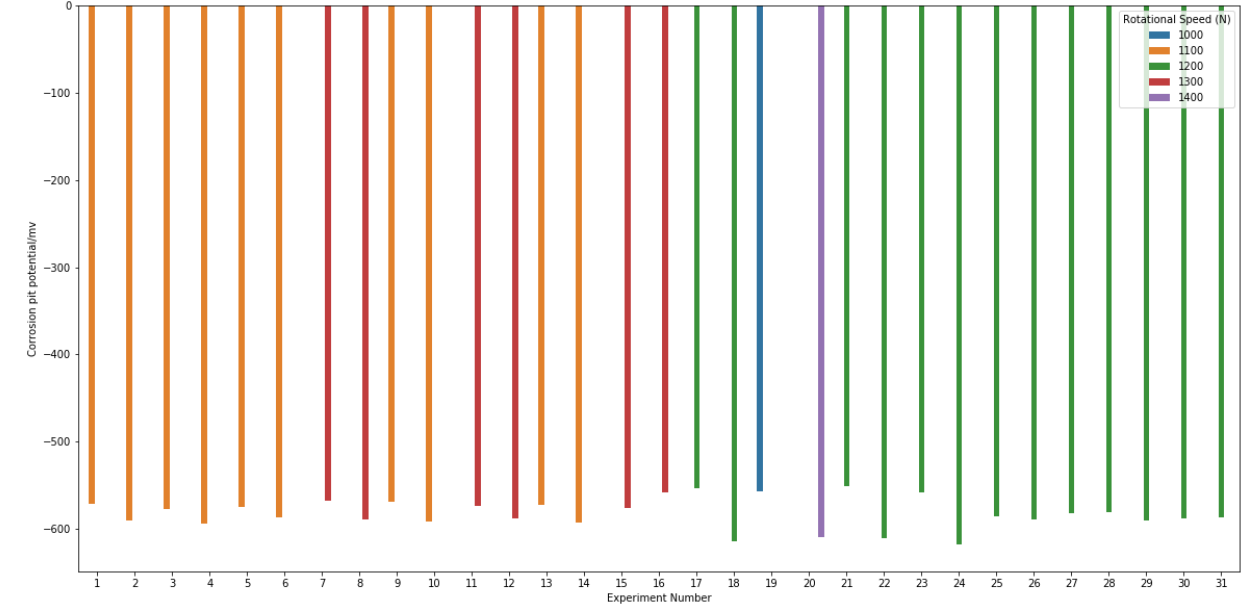
**Outputs**

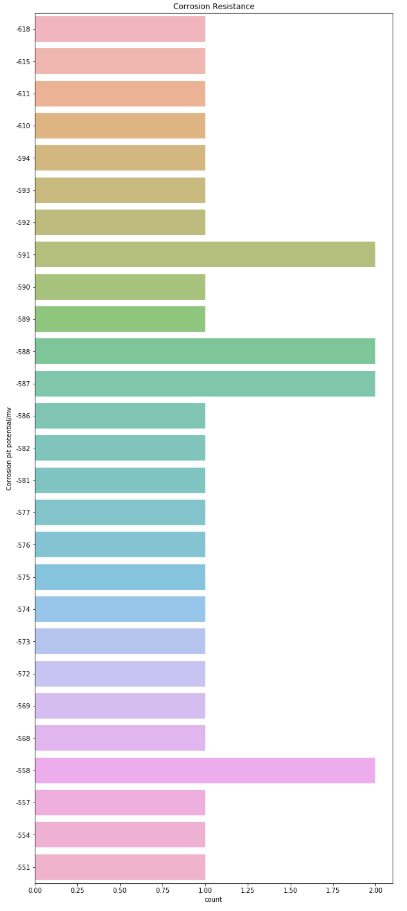




[-572 -591 -577 -594 -575 -587 -568 -589 -569 -592 -574 -588 -573 -593

-576 -558 -554 -615 -557 -610 -551 -611 -618 -586 -590 -582 -581]



**Procedure**

**2. Three dimensional Surface plots, Contour plots and Tri-surface plots**

**For Contour Plots:**

* Import libraries numpy, matplotlib, Axes3D from mpl\_toolkits and os
* Import the required dataset corrosion\_final\_data, set delimiter as comma and skip the header attributes by 1 row
* Print all the values
* Define the x and y co-ordinates using a numpy meshgrid to set x and y axis as [:,0] and [:,1] for further array representation and set z axis as the tile (or grids) with [:,2] and length as 1 for [;,2]
* Plot the 3D contour by setting the subplot x,y,z as 111 and define projection as 3D, set color from cmap as ‘rainbow’ and set the title for the respective contour plot
* Similarly follow the same steps for rotational speed (N), welding speed(S) and axial force(F) using their respective csv files
* For rotational speed set x,y,z axes as F, N and cpp respectively
* For welding speed set x,y,z axes as N, S and cpp respectively
* For rotational speed set x,y,z axes as S, F and cpp respectively

**For Surface Plots :**

* Import libraries numpy, matplotlib, Axes3D from mpl\_toolkits and os
* Import the required dataset corrosion\_final\_data, set delimiter as comma and skip the header attributes by 1 row
* Print all the values
* Define the x and y co-ordinates using a numpy meshgrid to set x and y axis as [:,0] and [:,1] for further array representation and set z axis as the tile (or grids) with [:,2] and length as 1 for [;,2]
* Plot the 3D Surface by setting the subplot x,y,z as 111 and define projection as 3D, set color from cmap as ‘viridris’ and set the title for the respective surface plot with their respective x, y, z labels
* Similarly follow the same steps for rotational speed (N), welding speed(S) and axial force(F) using their respective csv files
* For rotational speed set x,y,z axes as F, N and cpp respectively
* For welding speed set x,y,z axes as N, S and cpp respectively
* For rotational speed set x,y,z axes as S, F and cpp respectively

**Observation**

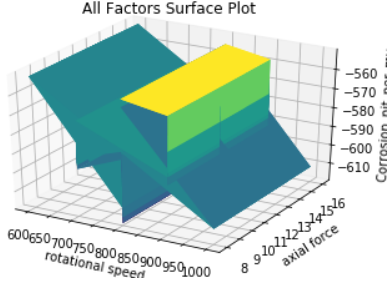
**For Contour Plots :**

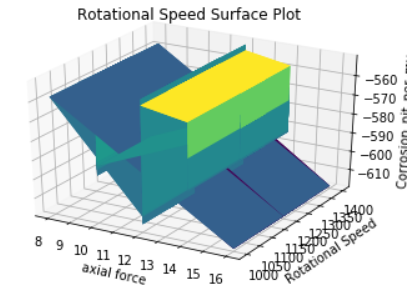
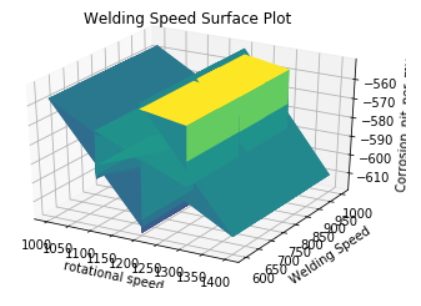
* In the first corrosion resistance contour we observe that highest cpp value (-560) is when N = 700 rpm at range of 8-16 KN axial force and lowest cpp value (-610) is when N = 1000 rpm at range of 8-16 KN axial force
* This shows that as rotational speed increases corrosion pit potential/mv also increases. Hence they are directly proportional
* This direct proportionality is observed in all the individual plots where as the welding speed increases cpp also increases
* We can also see that there is no difference in the plots of corrosion resistance and axial force separate plots
* This proves that change in axial force doesn’t directly affect corrosion resistance

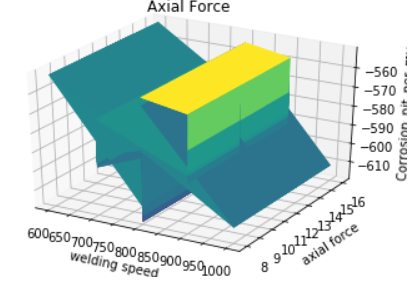
**For Surface Plots :**

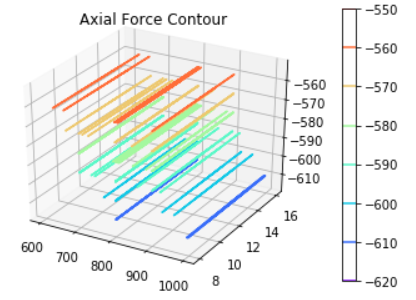
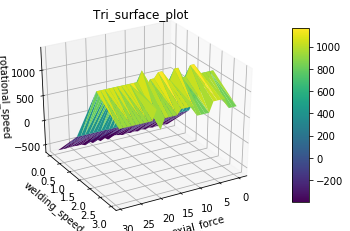
* In the Rotational speed surface plot we observe that it’s a steep gradient slope from the axial force as it proceeds from 8 till the last plot at 16
* At F = 12, cpp and rotational speed values of all the d.p range can be constant. This shows that F= 12 is a close to ideal value for almost all operating attributes
* Similarly in the Welding speed surface plot we observe cpp keeps decreasing when N ranges from 1000rpm to 1200rpm at a range of S from 600 mm/min to 1000 mm/min. And cpp and S kept N constant at 1200 rpm
* In the Axial Speed surface plot when S ranges from 600 mm/min to 800 mm/min the value of cpp goes on decreasing at F from 8KN – 16KN and same goes when S ranges from 700 mm/min to 1000 mm/min with axial force at a range of 9.5 KN to 16KN

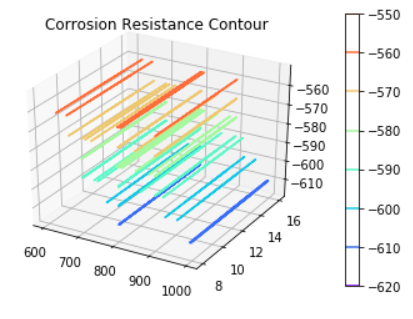
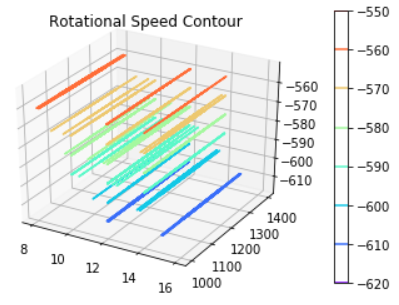
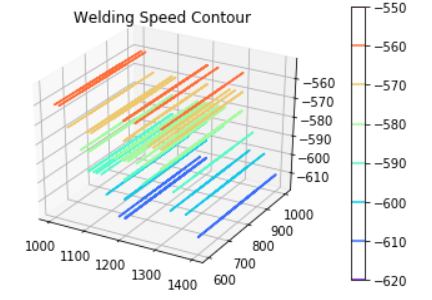
**Outputs**

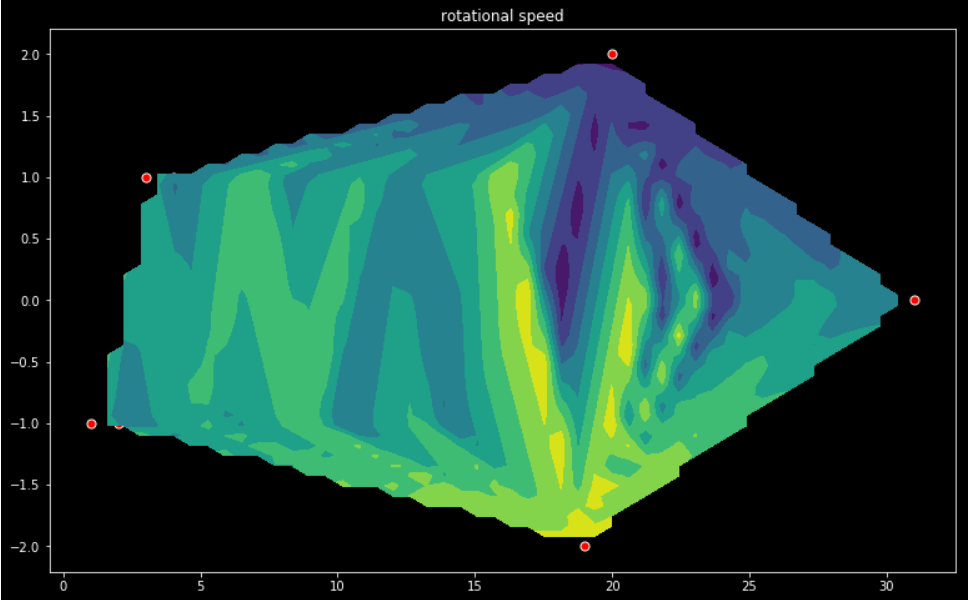
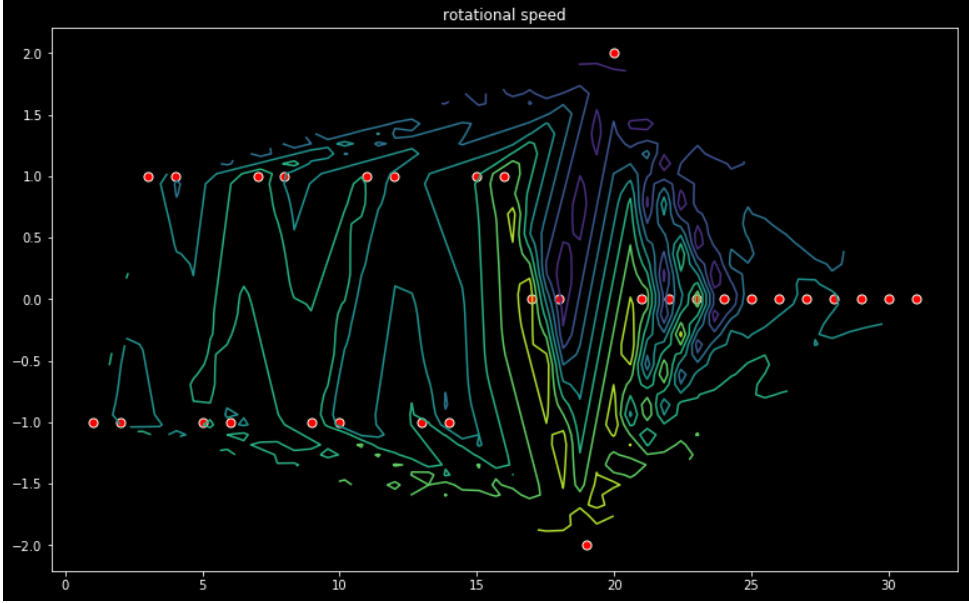
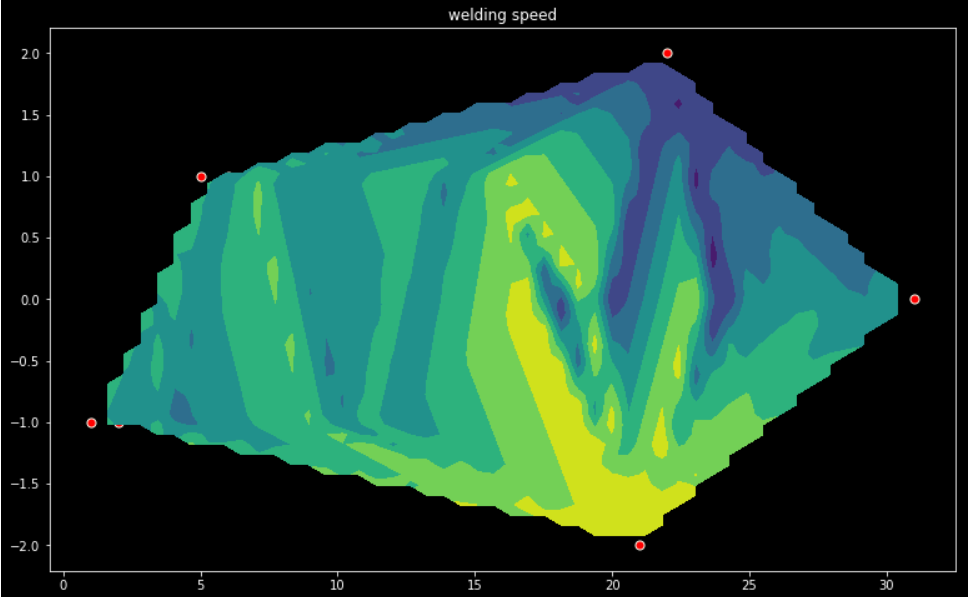
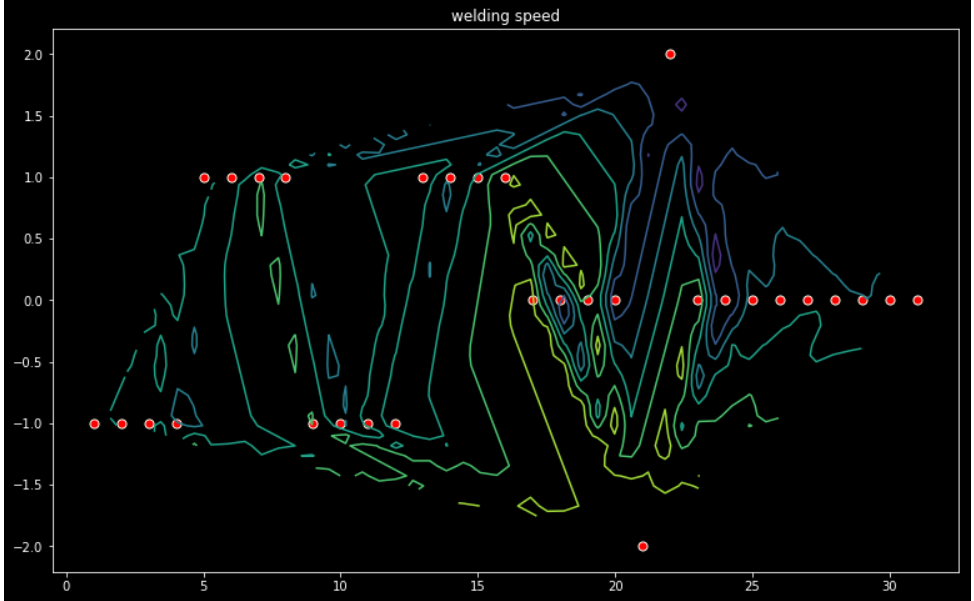
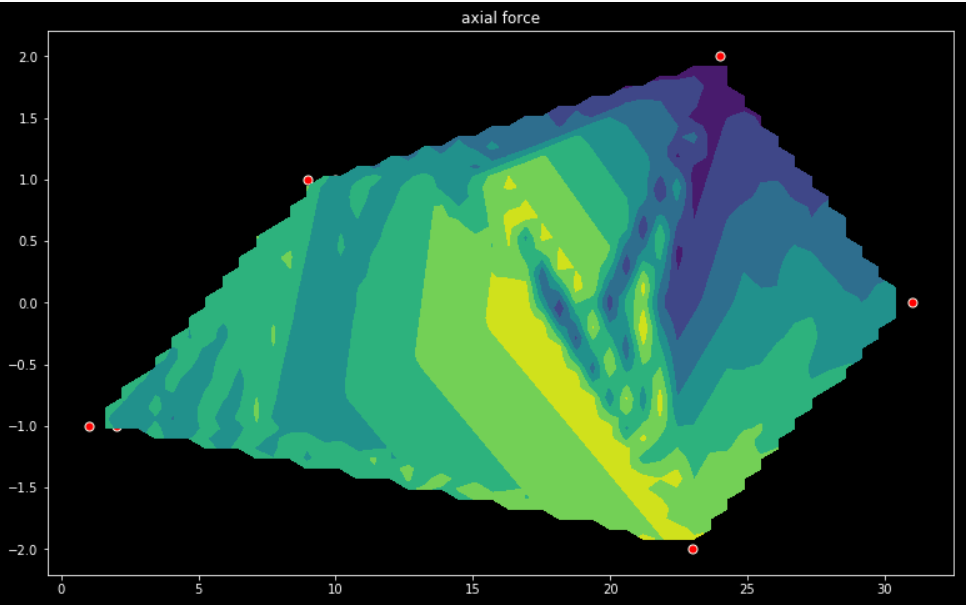
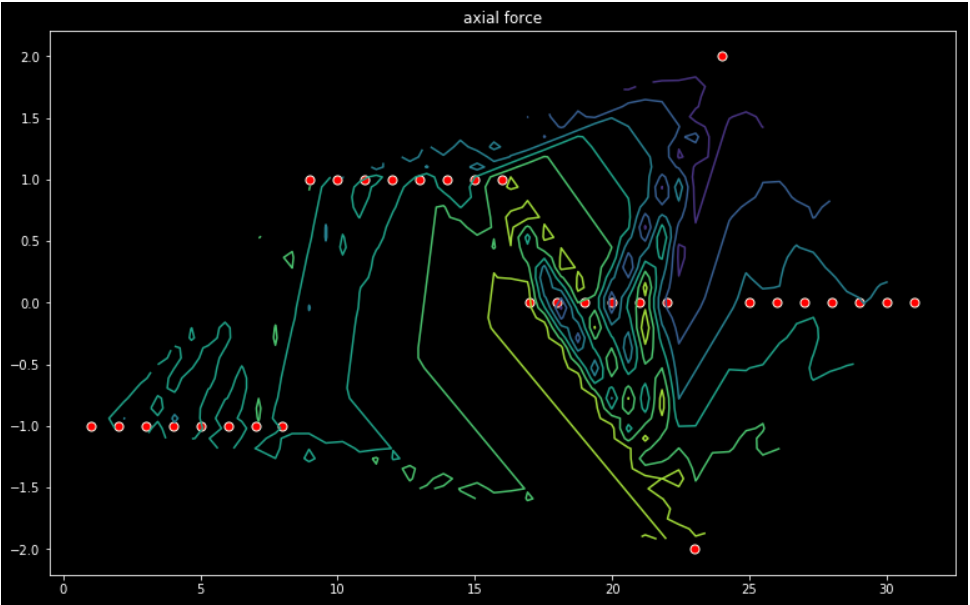
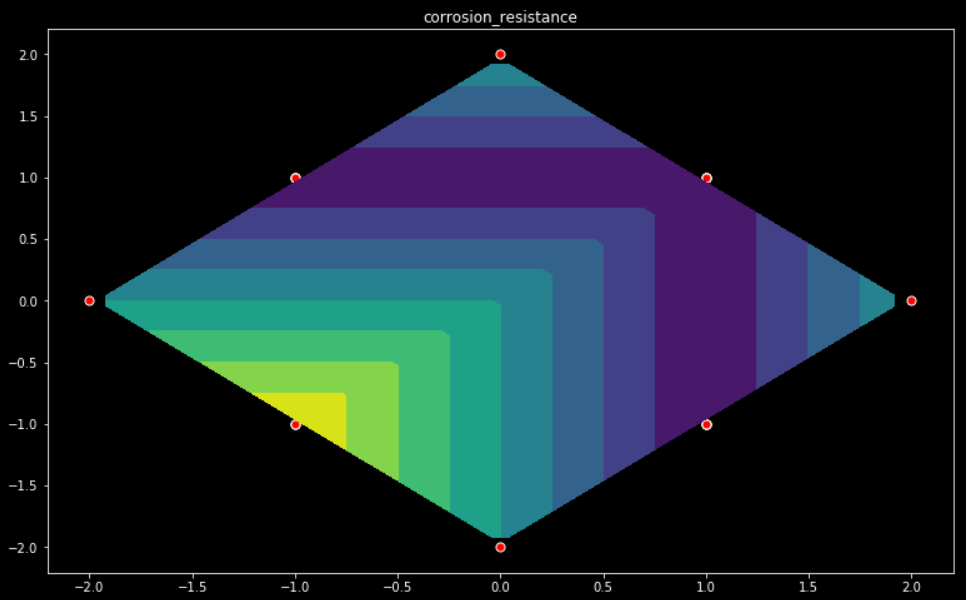
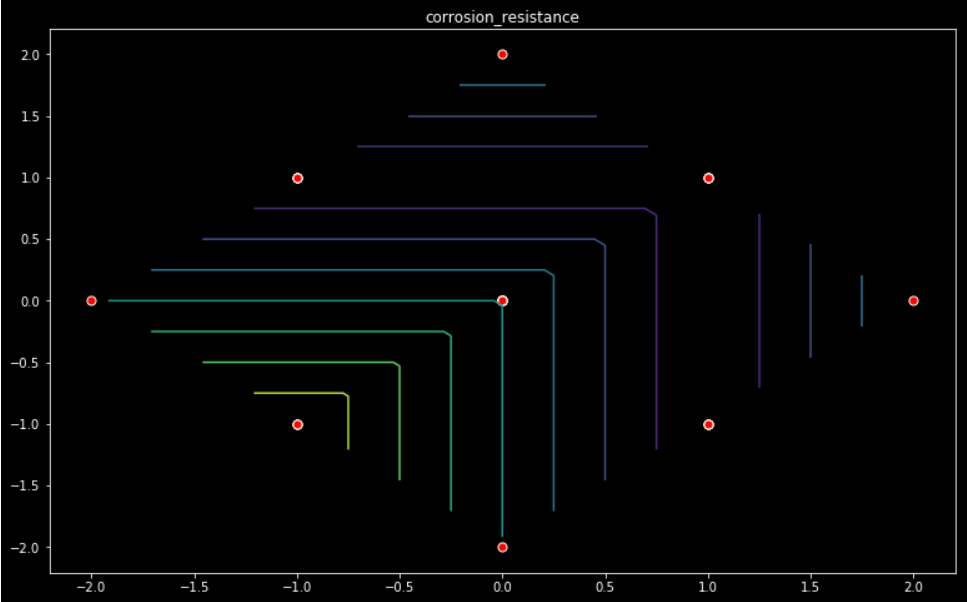




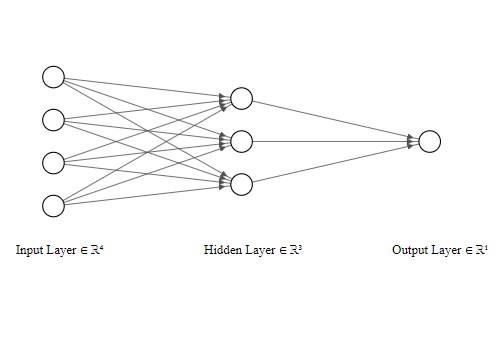
  

**Procedure**

**3. Neural Network Architecture for corrosion resistance and mean square error & mean absolute percentage error**

* Import all required libraries i.e. numpy (loadtext)), pandas, matplotlib, scikitlearn (standardscalar, model\_selection, regression), keras (sequential, dense, SGD), tensorflow, os, csv and time
* Import required dataset i.e. corrosion\_final\_data.csv
* Print the dataset values
* Split datasets in input values(x) and output values(y) as iloc values [:, :-1] and [:, -1] respectively for data pre-processing for regression
* Standardize dataset and reshape the dataset using standardscalar and fit\_transform x and y. Use reshape on y w.r.t. its length at 1
* Split into train and test using model\_selection and set the train\_size and test\_size as 0.75 and 0.25 respectively. This will not overfit or underfit the data
* Using formula Nh=Ns/(α∗(Ni+No)) ie; 30/(2(3+1)) = 3.75 approx 4. This formula is used to fix the number of nodes in the hidden layers.
* Select model as Sequential and set the layers of the neural network architecture as follows:
* The first hidden layer has 4 nodes, input are three so input\_dim=3, activation is set as relu (rectified linear unit) and kernel initializer as uniform
* The second hidden layer has 3 nodes and uses relu activation function
* The output layer had one node and uses sigmoid activation function
* Setting input layer with right no. of input features by setting it to 3 for 3 input features
* Set opt using SGD from keras at set linear at 0.01 and momentum to 0.9
* Use model.compile to set mean squared error, optimizer to adam and metrics to mean\_squared\_error same goes for mean absolute percentage error
* This model will be fit with stochastic gradient descent with a learning rate of 0.01 and momentum of 0.9
* Training will be performed for 100 epochs and test set will be evaluated at the end of each epoch by fitting the model to history variable
* Now evaluate the model training and testing for mean squared error and training and testing for mean absolute percentage error
* Finally plot the loss during training and testing separately using history while label, legend, title is being set as per need



**Observation**

**For Neural Network Architecture, Mean Square Error and Mean Absolute Percentage Error**

* Firstly since we are using Keras setting up the sequence of our model , defining the thickness of the layers and optimizing the NN , Tensorflow is used in the backend
* Since we separate the dataset as inputs and outputs we are taking cpp as the outputs and N,S,F as the inputs
* And the x and y are split and reshaped successfully with the train and test size defined and the hidden layers are also defined successfully to relu and sigmoid
* Since stochastic gradient descent is set with a learning rate of 0.01 and momentum of 0.9 the model is steady and not overfitted
* When the epochs of 100 and batch size of 10 set , we observe that we get four main values as loss, val\_loss, mean\_square\_loss , val\_mean\_square\_loss
* We train on 22 samples that validate on 8 samples with each epoch being tested and trained at a speed of 0 secs 505 usecs to 35 m secs 1ms range
* After training and testing the model over 60 times the mean\_square\_error decreases from 1.4450 to 1.2842
* Similarly val\_mean\_squared\_error decreases from 0.4210 to 0.4109 and val\_loss decreases from 0.4208 to 0.4109
* Loss of the train and test NN is from 1.4450 to 1.2842
* While evaluating the training and testing model we observe that train = 1.283 and test = 0.411 with samples 22 and 8 respectively
* Since we have plot different line plots for loss and mean square error the train plot loss decreases as epoch is increased while for test the relation between loss and epoch remains constant at 0.4109
* Same goes for mean\_squared\_error

**Outputs**

**For NN and Mean square error**

WARNING:tensorflow:From C:\Users\Shrishti D Hore\Miniconda3\envs\shripy\lib\site-packages\keras\backend\tensorflow\_backend.py:422: The name tf.global\_variables is deprecated. Please use tf.compat.v1.global\_variables instead.

From C:\Users\Shrishti D Hore\Miniconda3\envs\shripy\lib\site-packages\keras\backend\tensorflow\_backend.py:422: The name tf.global\_variables is deprecated. Please use tf.compat.v1.global\_variables instead.

Train on 22 samples, validate on 8 samples

Epoch 1/100

22/22 [==============================] - 1s 35ms/step - loss: 1.4450 - mean\_squared\_error: 1.4450 - val\_loss: 0.4210 - val\_mean\_squared\_error: 0.4210

Epoch 2/100

22/22 [==============================] - 0s 592us/step - loss: 1.4419 - mean\_squared\_error: 1.4419 - val\_loss: 0.4208 - val\_mean\_squared\_error: 0.4208

Epoch 3/100

22/22 [==============================] - 0s 789us/step - loss: 1.4401 - mean\_squared\_error: 1.4401 - val\_loss: 0.4206 - val\_mean\_squared\_error: 0.4206

Epoch 4/100

22/22 [==============================] - 0s 1ms/step - loss: 1.4376 - mean\_squared\_error: 1.4376 - val\_loss: 0.4203 - val\_mean\_squared\_error: 0.4203

Epoch 5/100

22/22 [==============================] - 0s 733us/step - loss: 1.4356 - mean\_squared\_error: 1.4356 - val\_loss: 0.4201 - val\_mean\_squared\_error: 0.4201

Epoch 6/100

22/22 [==============================] - 0s 830us/step - loss: 1.4330 - mean\_squared\_error: 1.4330 - val\_loss: 0.4199 - val\_mean\_squared\_error: 0.4199

Epoch 7/100

22/22 [==============================] - 0s 894us/step - loss: 1.4310 - mean\_squared\_error: 1.4310 - val\_loss: 0.4197 - val\_mean\_squared\_error: 0.4197

Epoch 8/100

22/22 [==============================] - 0s 696us/step - loss: 1.4290 - mean\_squared\_error: 1.4290 - val\_loss: 0.4194 - val\_mean\_squared\_error: 0.4194

Epoch 9/100

22/22 [==============================] - 0s 710us/step - loss: 1.4272 - mean\_squared\_error: 1.4272 - val\_loss: 0.4191 - val\_mean\_squared\_error: 0.4191

Epoch 10/100

22/22 [==============================] - 0s 2ms/step - loss: 1.4252 - mean\_squared\_error: 1.4252 - val\_loss: 0.4189 - val\_mean\_squared\_error: 0.4189

Epoch 11/100

22/22 [==============================] - 0s 757us/step - loss: 1.4232 - mean\_squared\_error: 1.4232 - val\_loss: 0.4187 - val\_mean\_squared\_error: 0.4187

Epoch 12/100

22/22 [==============================] - 0s 689us/step - loss: 1.4215 - mean\_squared\_error: 1.4215 - val\_loss: 0.4186 - val\_mean\_squared\_error: 0.4186

Epoch 13/100

22/22 [==============================] - 0s 620us/step - loss: 1.4202 - mean\_squared\_error: 1.4202 - val\_loss: 0.4184 - val\_mean\_squared\_error: 0.4184

Epoch 14/100

22/22 [==============================] - 0s 719us/step - loss: 1.4181 - mean\_squared\_error: 1.4181 - val\_loss: 0.4182 - val\_mean\_squared\_error: 0.4182

Epoch 15/100

22/22 [==============================] - 0s 579us/step - loss: 1.4166 - mean\_squared\_error: 1.4166 - val\_loss: 0.4179 - val\_mean\_squared\_error: 0.4179

Epoch 16/100

22/22 [==============================] - 0s 2ms/step - loss: 1.4153 - mean\_squared\_error: 1.4153 - val\_loss: 0.4176 - val\_mean\_squared\_error: 0.4176

Epoch 17/100

22/22 [==============================] - 0s 645us/step - loss: 1.4135 - mean\_squared\_error: 1.4135 - val\_loss: 0.4175 - val\_mean\_squared\_error: 0.4175

Epoch 18/100

22/22 [==============================] - 0s 1ms/step - loss: 1.4122 - mean\_squared\_error: 1.4122 - val\_loss: 0.4173 - val\_mean\_squared\_error: 0.4173

Epoch 19/100

22/22 [==============================] - 0s 841us/step - loss: 1.4109 - mean\_squared\_error: 1.4109 - val\_loss: 0.4171 - val\_mean\_squared\_error: 0.4171

Epoch 20/100

22/22 [==============================] - 0s 690us/step - loss: 1.4096 - mean\_squared\_error: 1.4096 - val\_loss: 0.4169 - val\_mean\_squared\_error: 0.4169

Epoch 21/100

22/22 [==============================] - 0s 534us/step - loss: 1.4080 - mean\_squared\_error: 1.4080 - val\_loss: 0.4168 - val\_mean\_squared\_error: 0.4168

Epoch 22/100

22/22 [==============================] - 0s 571us/step - loss: 1.4065 - mean\_squared\_error: 1.4065 - val\_loss: 0.4167 - val\_mean\_squared\_error: 0.4167

Epoch 23/100

22/22 [==============================] - 0s 547us/step - loss: 1.4042 - mean\_squared\_error: 1.4042 - val\_loss: 0.4166 - val\_mean\_squared\_error: 0.4166

Epoch 24/100

22/22 [==============================] - 0s 543us/step - loss: 1.4026 - mean\_squared\_error: 1.4026 - val\_loss: 0.4165 - val\_mean\_squared\_error: 0.4165

Epoch 25/100

22/22 [==============================] - 0s 545us/step - loss: 1.4009 - mean\_squared\_error: 1.4009 - val\_loss: 0.4163 - val\_mean\_squared\_error: 0.4163

Epoch 26/100

22/22 [==============================] - 0s 603us/step - loss: 1.3994 - mean\_squared\_error: 1.3994 - val\_loss: 0.4161 - val\_mean\_squared\_error: 0.4161

Epoch 27/100

22/22 [==============================] - 0s 656us/step - loss: 1.3973 - mean\_squared\_error: 1.3973 - val\_loss: 0.4159 - val\_mean\_squared\_error: 0.4159

Epoch 28/100

22/22 [==============================] - 0s 800us/step - loss: 1.3957 - mean\_squared\_error: 1.3957 - val\_loss: 0.4158 - val\_mean\_squared\_error: 0.4158

Epoch 29/100

22/22 [==============================] - 0s 772us/step - loss: 1.3941 - mean\_squared\_error: 1.3941 - val\_loss: 0.4157 - val\_mean\_squared\_error: 0.4157

Epoch 30/100

22/22 [==============================] - 0s 648us/step - loss: 1.3924 - mean\_squared\_error: 1.3924 - val\_loss: 0.4155 - val\_mean\_squared\_error: 0.4155

Epoch 31/100

22/22 [==============================] - 0s 557us/step - loss: 1.3909 - mean\_squared\_error: 1.3909 - val\_loss: 0.4154 - val\_mean\_squared\_error: 0.4154

Epoch 32/100

22/22 [==============================] - 0s 708us/step - loss: 1.3895 - mean\_squared\_error: 1.3895 - val\_loss: 0.4153 - val\_mean\_squared\_error: 0.4153

Epoch 33/100

22/22 [==============================] - 0s 590us/step - loss: 1.3880 - mean\_squared\_error: 1.3880 - val\_loss: 0.4153 - val\_mean\_squared\_error: 0.4153

Epoch 34/100

22/22 [==============================] - 0s 589us/step - loss: 1.3865 - mean\_squared\_error: 1.3865 - val\_loss: 0.4151 - val\_mean\_squared\_error: 0.4151

Epoch 35/100

22/22 [==============================] - 0s 1ms/step - loss: 1.3850 - mean\_squared\_error: 1.3850 - val\_loss: 0.4150 - val\_mean\_squared\_error: 0.4150

Epoch 36/100

22/22 [==============================] - 0s 685us/step - loss: 1.3834 - mean\_squared\_error: 1.3834 - val\_loss: 0.4149 - val\_mean\_squared\_error: 0.4149

Epoch 37/100

22/22 [==============================] - 0s 907us/step - loss: 1.3816 - mean\_squared\_error: 1.3816 - val\_loss: 0.4148 - val\_mean\_squared\_error: 0.4148

Epoch 38/100

22/22 [==============================] - 0s 664us/step - loss: 1.3798 - mean\_squared\_error: 1.3798 - val\_loss: 0.4147 - val\_mean\_squared\_error: 0.4147

Epoch 39/100

22/22 [==============================] - 0s 639us/step - loss: 1.3786 - mean\_squared\_error: 1.3786 - val\_loss: 0.4147 - val\_mean\_squared\_error: 0.4147

Epoch 40/100

22/22 [==============================] - 0s 546us/step - loss: 1.3765 - mean\_squared\_error: 1.3765 - val\_loss: 0.4147 - val\_mean\_squared\_error: 0.4147

Epoch 41/100

22/22 [==============================] - 0s 617us/step - loss: 1.3751 - mean\_squared\_error: 1.3751 - val\_loss: 0.4146 - val\_mean\_squared\_error: 0.4146

Epoch 42/100

22/22 [==============================] - 0s 897us/step - loss: 1.3735 - mean\_squared\_error: 1.3735 - val\_loss: 0.4146 - val\_mean\_squared\_error: 0.4146

Epoch 43/100

22/22 [==============================] - 0s 645us/step - loss: 1.3723 - mean\_squared\_error: 1.3723 - val\_loss: 0.4145 - val\_mean\_squared\_error: 0.4145

Epoch 44/100

22/22 [==============================] - 0s 837us/step - loss: 1.3709 - mean\_squared\_error: 1.3709 - val\_loss: 0.4144 - val\_mean\_squared\_error: 0.4144

Epoch 45/100

22/22 [==============================] - 0s 884us/step - loss: 1.3693 - mean\_squared\_error: 1.3693 - val\_loss: 0.4143 - val\_mean\_squared\_error: 0.4143

Epoch 46/100

22/22 [==============================] - 0s 1ms/step - loss: 1.3677 - mean\_squared\_error: 1.3677 - val\_loss: 0.4142 - val\_mean\_squared\_error: 0.4142

Epoch 47/100

22/22 [==============================] - 0s 953us/step - loss: 1.3663 - mean\_squared\_error: 1.3663 - val\_loss: 0.4141 - val\_mean\_squared\_error: 0.4141

Epoch 48/100

22/22 [==============================] - 0s 746us/step - loss: 1.3648 - mean\_squared\_error: 1.3648 - val\_loss: 0.4139 - val\_mean\_squared\_error: 0.4139

Epoch 49/100

22/22 [==============================] - 0s 850us/step - loss: 1.3636 - mean\_squared\_error: 1.3636 - val\_loss: 0.4138 - val\_mean\_squared\_error: 0.4138

Epoch 50/100

22/22 [==============================] - 0s 538us/step - loss: 1.3617 - mean\_squared\_error: 1.3617 - val\_loss: 0.4136 - val\_mean\_squared\_error: 0.4136

Epoch 51/100

22/22 [==============================] - 0s 612us/step - loss: 1.3598 - mean\_squared\_error: 1.3598 - val\_loss: 0.4134 - val\_mean\_squared\_error: 0.4134

Epoch 52/100

22/22 [==============================] - 0s 533us/step - loss: 1.3586 - mean\_squared\_error: 1.3586 - val\_loss: 0.4132 - val\_mean\_squared\_error: 0.4132

Epoch 53/100

22/22 [==============================] - 0s 742us/step - loss: 1.3570 - mean\_squared\_error: 1.3570 - val\_loss: 0.4131 - val\_mean\_squared\_error: 0.4131

Epoch 54/100

22/22 [==============================] - 0s 1ms/step - loss: 1.3555 - mean\_squared\_error: 1.3555 - val\_loss: 0.4129 - val\_mean\_squared\_error: 0.4129

Epoch 55/100

22/22 [==============================] - 0s 557us/step - loss: 1.3541 - mean\_squared\_error: 1.3541 - val\_loss: 0.4128 - val\_mean\_squared\_error: 0.4128

Epoch 56/100

22/22 [==============================] - 0s 621us/step - loss: 1.3526 - mean\_squared\_error: 1.3526 - val\_loss: 0.4127 - val\_mean\_squared\_error: 0.4127

Epoch 57/100

22/22 [==============================] - 0s 662us/step - loss: 1.3505 - mean\_squared\_error: 1.3505 - val\_loss: 0.4125 - val\_mean\_squared\_error: 0.4125

Epoch 58/100

22/22 [==============================] - 0s 505us/step - loss: 1.3488 - mean\_squared\_error: 1.3488 - val\_loss: 0.4124 - val\_mean\_squared\_error: 0.4124

Epoch 59/100

22/22 [==============================] - 0s 695us/step - loss: 1.3474 - mean\_squared\_error: 1.3474 - val\_loss: 0.4123 - val\_mean\_squared\_error: 0.4123

Epoch 60/100

22/22 [==============================] - 0s 761us/step - loss: 1.3456 - mean\_squared\_error: 1.3456 - val\_loss: 0.4122 - val\_mean\_squared\_error: 0.4122

Epoch 61/100

22/22 [==============================] - 0s 713us/step - loss: 1.3441 - mean\_squared\_error: 1.3441 - val\_loss: 0.4121 - val\_mean\_squared\_error: 0.4121

Epoch 62/100

22/22 [==============================] - 0s 658us/step - loss: 1.3428 - mean\_squared\_error: 1.3428 - val\_loss: 0.4120 - val\_mean\_squared\_error: 0.4120

Epoch 63/100

22/22 [==============================] - 0s 584us/step - loss: 1.3414 - mean\_squared\_error: 1.3414 - val\_loss: 0.4119 - val\_mean\_squared\_error: 0.4119

Epoch 64/100

22/22 [==============================] - 0s 683us/step - loss: 1.3398 - mean\_squared\_error: 1.3398 - val\_loss: 0.4118 - val\_mean\_squared\_error: 0.4118

Epoch 65/100

22/22 [==============================] - 0s 698us/step - loss: 1.3384 - mean\_squared\_error: 1.3384 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 66/100

22/22 [==============================] - 0s 815us/step - loss: 1.3369 - mean\_squared\_error: 1.3369 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 67/100

22/22 [==============================] - 0s 467us/step - loss: 1.3356 - mean\_squared\_error: 1.3356 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 68/100

22/22 [==============================] - 0s 760us/step - loss: 1.3341 - mean\_squared\_error: 1.3341 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 69/100

22/22 [==============================] - 0s 746us/step - loss: 1.3325 - mean\_squared\_error: 1.3325 - val\_loss: 0.4118 - val\_mean\_squared\_error: 0.4118

Epoch 70/100

22/22 [==============================] - 0s 751us/step - loss: 1.3309 - mean\_squared\_error: 1.3309 - val\_loss: 0.4119 - val\_mean\_squared\_error: 0.4119

Epoch 71/100

22/22 [==============================] - 0s 607us/step - loss: 1.3294 - mean\_squared\_error: 1.3294 - val\_loss: 0.4119 - val\_mean\_squared\_error: 0.4119

Epoch 72/100

22/22 [==============================] - 0s 612us/step - loss: 1.3278 - mean\_squared\_error: 1.3278 - val\_loss: 0.4120 - val\_mean\_squared\_error: 0.4120

Epoch 73/100

22/22 [==============================] - 0s 612us/step - loss: 1.3263 - mean\_squared\_error: 1.3263 - val\_loss: 0.4119 - val\_mean\_squared\_error: 0.4119

Epoch 74/100

22/22 [==============================] - 0s 522us/step - loss: 1.3248 - mean\_squared\_error: 1.3248 - val\_loss: 0.4119 - val\_mean\_squared\_error: 0.4119

Epoch 75/100

22/22 [==============================] - 0s 733us/step - loss: 1.3235 - mean\_squared\_error: 1.3235 - val\_loss: 0.4118 - val\_mean\_squared\_error: 0.4118

Epoch 76/100

22/22 [==============================] - 0s 629us/step - loss: 1.3217 - mean\_squared\_error: 1.3217 - val\_loss: 0.4118 - val\_mean\_squared\_error: 0.4118

Epoch 77/100

22/22 [==============================] - ETA: 0s - loss: 0.9926 - mean\_squared\_error: 0.99 - 0s 573us/step - loss: 1.3204 - mean\_squared\_error: 1.3204 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 78/100

22/22 [==============================] - 0s 836us/step - loss: 1.3189 - mean\_squared\_error: 1.3189 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 79/100

22/22 [==============================] - 0s 733us/step - loss: 1.3176 - mean\_squared\_error: 1.3176 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 80/100

22/22 [==============================] - 0s 1ms/step - loss: 1.3160 - mean\_squared\_error: 1.3160 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 81/100

22/22 [==============================] - 0s 1ms/step - loss: 1.3144 - mean\_squared\_error: 1.3144 - val\_loss: 0.4118 - val\_mean\_squared\_error: 0.4118

Epoch 82/100

22/22 [==============================] - 0s 619us/step - loss: 1.3130 - mean\_squared\_error: 1.3130 - val\_loss: 0.4118 - val\_mean\_squared\_error: 0.4118

Epoch 83/100

22/22 [==============================] - 0s 619us/step - loss: 1.3111 - mean\_squared\_error: 1.3111 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 84/100

22/22 [==============================] - 0s 1ms/step - loss: 1.3095 - mean\_squared\_error: 1.3095 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 85/100

22/22 [==============================] - 0s 715us/step - loss: 1.3070 - mean\_squared\_error: 1.3070 - val\_loss: 0.4117 - val\_mean\_squared\_error: 0.4117

Epoch 86/100

22/22 [==============================] - 0s 796us/step - loss: 1.3053 - mean\_squared\_error: 1.3053 - val\_loss: 0.4116 - val\_mean\_squared\_error: 0.4116

Epoch 87/100

22/22 [==============================] - 0s 590us/step - loss: 1.3038 - mean\_squared\_error: 1.3038 - val\_loss: 0.4116 - val\_mean\_squared\_error: 0.4116

Epoch 88/100

22/22 [==============================] - 0s 867us/step - loss: 1.3019 - mean\_squared\_error: 1.3019 - val\_loss: 0.4115 - val\_mean\_squared\_error: 0.4115

Epoch 89/100

22/22 [==============================] - 0s 659us/step - loss: 1.3007 - mean\_squared\_error: 1.3007 - val\_loss: 0.4114 - val\_mean\_squared\_error: 0.4114

Epoch 90/100

22/22 [==============================] - 0s 710us/step - loss: 1.2990 - mean\_squared\_error: 1.2990 - val\_loss: 0.4114 - val\_mean\_squared\_error: 0.4114

Epoch 91/100

22/22 [==============================] - 0s 2ms/step - loss: 1.2974 - mean\_squared\_error: 1.2974 - val\_loss: 0.4113 - val\_mean\_squared\_error: 0.4113

Epoch 92/100

22/22 [==============================] - 0s 782us/step - loss: 1.2959 - mean\_squared\_error: 1.2959 - val\_loss: 0.4112 - val\_mean\_squared\_error: 0.4112

Epoch 93/100

22/22 [==============================] - 0s 655us/step - loss: 1.2946 - mean\_squared\_error: 1.2946 - val\_loss: 0.4111 - val\_mean\_squared\_error: 0.4111

Epoch 94/100

22/22 [==============================] - 0s 626us/step - loss: 1.2927 - mean\_squared\_error: 1.2927 - val\_loss: 0.4111 - val\_mean\_squared\_error: 0.4111

Epoch 95/100

22/22 [==============================] - 0s 686us/step - loss: 1.2911 - mean\_squared\_error: 1.2911 - val\_loss: 0.4110 - val\_mean\_squared\_error: 0.4110

Epoch 96/100

22/22 [==============================] - 0s 1ms/step - loss: 1.2899 - mean\_squared\_error: 1.2899 - val\_loss: 0.4110 - val\_mean\_squared\_error: 0.4110

Epoch 97/100

22/22 [==============================] - 0s 678us/step - loss: 1.2881 - mean\_squared\_error: 1.2881 - val\_loss: 0.4109 - val\_mean\_squared\_error: 0.4109

Epoch 98/100

22/22 [==============================] - 0s 542us/step - loss: 1.2866 - mean\_squared\_error: 1.2866 - val\_loss: 0.4109 - val\_mean\_squared\_error: 0.4109

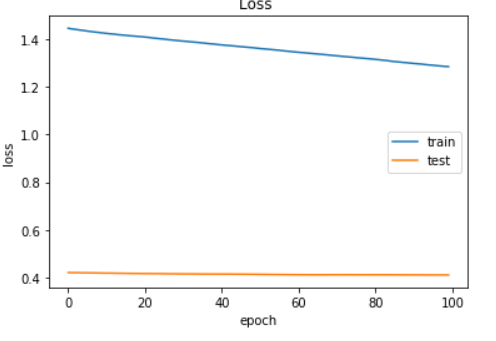
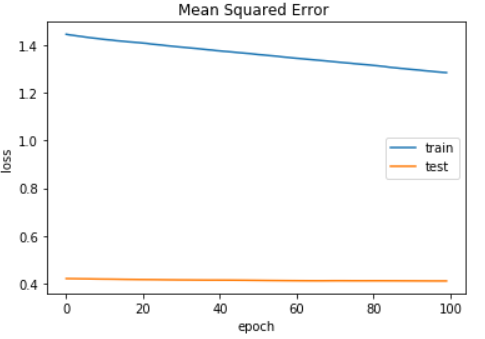
Epoch 99/100

22/22 [==============================] - 0s 753us/step - loss: 1.2852 - mean\_squared\_error: 1.2852 - val\_loss: 0.4109 - val\_mean\_squared\_error: 0.4109

Epoch 100/100

22/22 [==============================] - 0s 531us/step - loss: 1.2842 - mean\_squared\_error: 1.2842 - val\_loss: 0.4109 - val\_mean\_squared\_error: 0.4109

fitted model sucessfully !!!

**For Mean Absolute Percentage Error :**

runfile('C:/Users/ShrishtiDHore/OneDrive/Documents/Stir\_Research/Folder/Stir\_Research\_Corrosion\_Resistance\_NN\_and\_mean\_absolute\_percentage\_error.py', wdir='C:/Users/Shrishti D Hore/OneDrive/Documents/Stir\_Research/Folder')

C:\Users\Shrishti D Hore\Miniconda3\envs\shripy\lib\site-packages\sklearn\utils\validation.py:595: DataConversionWarning: Data with input dtype int64 was converted to float64 by StandardScaler.

warnings.warn(msg, DataConversionWarning)

C:\Users\Shrishti D Hore\Miniconda3\envs\shripy\lib\site-packages\sklearn\utils\validation.py:595: DataConversionWarning: Data with input dtype int64 was converted to float64 by StandardScaler.

warnings.warn(msg, DataConversionWarning)

C:\Users\Shrishti D Hore\Miniconda3\envs\shripy\lib\site-packages\sklearn\utils\validation.py:595: DataConversionWarning: Data with input dtype int64 was converted to float64 by StandardScaler.

warnings.warn(msg, DataConversionWarning)

C:\Users\Shrishti D Hore\Miniconda3\envs\shripy\lib\site-packages\sklearn\utils\validation.py:595: DataConversionWarning: Data with input dtype int64 was converted to float64 by StandardScaler.

warnings.warn(msg, DataConversionWarning)

Train on 22 samples, validate on 8 samples

Epoch 1/100

22/22 [==============================] - 3s 130ms/step - loss: 187.0062 - mean\_absolute\_percentage\_error: 187.0063 - val\_loss: 194.0537 - val\_mean\_absolute\_percentage\_error: 194.0537

Epoch 2/100

22/22 [==============================] - 0s 5ms/step - loss: 186.4747 - mean\_absolute\_percentage\_error: 186.4747 - val\_loss: 193.6945 - val\_mean\_absolute\_percentage\_error: 193.6945

Epoch 3/100

22/22 [==============================] - 0s 820us/step - loss: 185.8751 - mean\_absolute\_percentage\_error: 185.8751 - val\_loss: 193.3386 - val\_mean\_absolute\_percentage\_error: 193.3386

Epoch 4/100

22/22 [==============================] - 0s 5ms/step - loss: 185.4050 - mean\_absolute\_percentage\_error: 185.4050 - val\_loss: 192.9883 - val\_mean\_absolute\_percentage\_error: 192.9883

Epoch 5/100

22/22 [==============================] - 0s 3ms/step - loss: 184.8652 - mean\_absolute\_percentage\_error: 184.8652 - val\_loss: 192.6509 - val\_mean\_absolute\_percentage\_error: 192.6509

Epoch 6/100

22/22 [==============================] - 0s 3ms/step - loss: 184.3589 - mean\_absolute\_percentage\_error: 184.3589 - val\_loss: 192.3195 - val\_mean\_absolute\_percentage\_error: 192.3195

Epoch 7/100

22/22 [==============================] - 0s 3ms/step - loss: 183.8652 - mean\_absolute\_percentage\_error: 183.8652 - val\_loss: 191.9897 - val\_mean\_absolute\_percentage\_error: 191.9897

Epoch 8/100

22/22 [==============================] - 0s 5ms/step - loss: 183.3993 - mean\_absolute\_percentage\_error: 183.3994 - val\_loss: 191.6617 - val\_mean\_absolute\_percentage\_error: 191.6617

Epoch 9/100

22/22 [==============================] - 0s 3ms/step - loss: 182.9079 - mean\_absolute\_percentage\_error: 182.9079 - val\_loss: 191.3528 - val\_mean\_absolute\_percentage\_error: 191.3528

Epoch 10/100

22/22 [==============================] - 0s 947us/step - loss: 182.3647 - mean\_absolute\_percentage\_error: 182.3647 - val\_loss: 191.0504 - val\_mean\_absolute\_percentage\_error: 191.0504

Epoch 11/100

22/22 [==============================] - 0s 2ms/step - loss: 181.9306 - mean\_absolute\_percentage\_error: 181.9306 - val\_loss: 190.7373 - val\_mean\_absolute\_percentage\_error: 190.7373

Epoch 12/100

22/22 [==============================] - 0s 4ms/step - loss: 181.4755 - mean\_absolute\_percentage\_error: 181.4755 - val\_loss: 190.4290 - val\_mean\_absolute\_percentage\_error: 190.4290

Epoch 13/100

22/22 [==============================] - 0s 4ms/step - loss: 181.0210 - mean\_absolute\_percentage\_error: 181.0210 - val\_loss: 190.1255 - val\_mean\_absolute\_percentage\_error: 190.1255

Epoch 14/100

22/22 [==============================] - 0s 908us/step - loss: 180.5844 - mean\_absolute\_percentage\_error: 180.5844 - val\_loss: 189.8250 - val\_mean\_absolute\_percentage\_error: 189.8250

Epoch 15/100

22/22 [==============================] - 0s 1ms/step - loss: 180.1646 - mean\_absolute\_percentage\_error: 180.1646 - val\_loss: 189.5235 - val\_mean\_absolute\_percentage\_error: 189.5235

Epoch 16/100

22/22 [==============================] - 0s 2ms/step - loss: 179.7471 - mean\_absolute\_percentage\_error: 179.7471 - val\_loss: 189.2336 - val\_mean\_absolute\_percentage\_error: 189.2336

Epoch 17/100

22/22 [==============================] - 0s 4ms/step - loss: 179.3336 - mean\_absolute\_percentage\_error: 179.3336 - val\_loss: 188.9499 - val\_mean\_absolute\_percentage\_error: 188.9499

Epoch 18/100

22/22 [==============================] - 0s 2ms/step - loss: 178.9553 - mean\_absolute\_percentage\_error: 178.9553 - val\_loss: 188.6781 - val\_mean\_absolute\_percentage\_error: 188.6781

Epoch 19/100

22/22 [==============================] - 0s 7ms/step - loss: 178.6039 - mean\_absolute\_percentage\_error: 178.6039 - val\_loss: 188.3970 - val\_mean\_absolute\_percentage\_error: 188.3970

Epoch 20/100

22/22 [==============================] - 0s 5ms/step - loss: 178.3440 - mean\_absolute\_percentage\_error: 178.3440 - val\_loss: 188.1007 - val\_mean\_absolute\_percentage\_error: 188.1007

Epoch 21/100

22/22 [==============================] - 0s 1ms/step - loss: 178.0773 - mean\_absolute\_percentage\_error: 178.0773 - val\_loss: 187.8244 - val\_mean\_absolute\_percentage\_error: 187.8244

Epoch 22/100

22/22 [==============================] - 0s 679us/step - loss: 177.8228 - mean\_absolute\_percentage\_error: 177.8228 - val\_loss: 187.5443 - val\_mean\_absolute\_percentage\_error: 187.5443

Epoch 23/100

22/22 [==============================] - 0s 1ms/step - loss: 177.5501 - mean\_absolute\_percentage\_error: 177.5501 - val\_loss: 187.2961 - val\_mean\_absolute\_percentage\_error: 187.2961

Epoch 24/100

22/22 [==============================] - 0s 9ms/step - loss: 177.3065 - mean\_absolute\_percentage\_error: 177.3065 - val\_loss: 187.0528 - val\_mean\_absolute\_percentage\_error: 187.0528

Epoch 25/100

22/22 [==============================] - 0s 2ms/step - loss: 177.0547 - mean\_absolute\_percentage\_error: 177.0547 - val\_loss: 186.8138 - val\_mean\_absolute\_percentage\_error: 186.8138

Epoch 26/100

22/22 [==============================] - 0s 505us/step - loss: 176.7875 - mean\_absolute\_percentage\_error: 176.7875 - val\_loss: 186.5731 - val\_mean\_absolute\_percentage\_error: 186.5731

Epoch 27/100

22/22 [==============================] - 0s 4ms/step - loss: 176.5654 - mean\_absolute\_percentage\_error: 176.5654 - val\_loss: 186.3344 - val\_mean\_absolute\_percentage\_error: 186.3344

Epoch 28/100

22/22 [==============================] - 0s 5ms/step - loss: 176.3222 - mean\_absolute\_percentage\_error: 176.3222 - val\_loss: 186.1600 - val\_mean\_absolute\_percentage\_error: 186.1600

Epoch 29/100

22/22 [==============================] - 0s 2ms/step - loss: 176.1134 - mean\_absolute\_percentage\_error: 176.1134 - val\_loss: 186.0350 - val\_mean\_absolute\_percentage\_error: 186.0350

Epoch 30/100

22/22 [==============================] - 0s 5ms/step - loss: 175.9629 - mean\_absolute\_percentage\_error: 175.9629 - val\_loss: 185.8981 - val\_mean\_absolute\_percentage\_error: 185.8981

Epoch 31/100

22/22 [==============================] - 0s 1ms/step - loss: 175.8021 - mean\_absolute\_percentage\_error: 175.8021 - val\_loss: 185.7462 - val\_mean\_absolute\_percentage\_error: 185.7462

Epoch 32/100

22/22 [==============================] - 0s 754us/step - loss: 175.6690 - mean\_absolute\_percentage\_error: 175.6690 - val\_loss: 185.5931 - val\_mean\_absolute\_percentage\_error: 185.5931

Epoch 33/100

22/22 [==============================] - 0s 4ms/step - loss: 175.4769 - mean\_absolute\_percentage\_error: 175.4769 - val\_loss: 185.4133 - val\_mean\_absolute\_percentage\_error: 185.4133

Epoch 34/100

22/22 [==============================] - 0s 4ms/step - loss: 175.2995 - mean\_absolute\_percentage\_error: 175.2995 - val\_loss: 185.1738 - val\_mean\_absolute\_percentage\_error: 185.1738

Epoch 35/100

22/22 [==============================] - 0s 2ms/step - loss: 175.1325 - mean\_absolute\_percentage\_error: 175.1325 - val\_loss: 184.9347 - val\_mean\_absolute\_percentage\_error: 184.9347

Epoch 36/100

22/22 [==============================] - 0s 3ms/step - loss: 174.9417 - mean\_absolute\_percentage\_error: 174.9417 - val\_loss: 184.7048 - val\_mean\_absolute\_percentage\_error: 184.7048

Epoch 37/100

22/22 [==============================] - 0s 2ms/step - loss: 174.7045 - mean\_absolute\_percentage\_error: 174.7045 - val\_loss: 184.5104 - val\_mean\_absolute\_percentage\_error: 184.5104

Epoch 38/100

22/22 [==============================] - 0s 703us/step - loss: 174.5545 - mean\_absolute\_percentage\_error: 174.5545 - val\_loss: 184.3293 - val\_mean\_absolute\_percentage\_error: 184.3293

Epoch 39/100

22/22 [==============================] - 0s 1ms/step - loss: 174.3888 - mean\_absolute\_percentage\_error: 174.3888 - val\_loss: 184.1553 - val\_mean\_absolute\_percentage\_error: 184.1553

Epoch 40/100

22/22 [==============================] - 0s 2ms/step - loss: 174.2570 - mean\_absolute\_percentage\_error: 174.2569 - val\_loss: 183.9698 - val\_mean\_absolute\_percentage\_error: 183.9698

Epoch 41/100

22/22 [==============================] - 0s 5ms/step - loss: 174.1017 - mean\_absolute\_percentage\_error: 174.1017 - val\_loss: 183.7781 - val\_mean\_absolute\_percentage\_error: 183.7781

Epoch 42/100

22/22 [==============================] - 0s 818us/step - loss: 173.9244 - mean\_absolute\_percentage\_error: 173.9244 - val\_loss: 183.5839 - val\_mean\_absolute\_percentage\_error: 183.5839

Epoch 43/100

22/22 [==============================] - 0s 1ms/step - loss: 173.7570 - mean\_absolute\_percentage\_error: 173.7570 - val\_loss: 183.3837 - val\_mean\_absolute\_percentage\_error: 183.3837

Epoch 44/100

22/22 [==============================] - 0s 3ms/step - loss: 173.6186 - mean\_absolute\_percentage\_error: 173.6186 - val\_loss: 183.1453 - val\_mean\_absolute\_percentage\_error: 183.1453

Epoch 45/100

22/22 [==============================] - 0s 507us/step - loss: 173.4061 - mean\_absolute\_percentage\_error: 173.4061 - val\_loss: 182.9088 - val\_mean\_absolute\_percentage\_error: 182.9088

Epoch 46/100

22/22 [==============================] - 0s 3ms/step - loss: 173.2502 - mean\_absolute\_percentage\_error: 173.2502 - val\_loss: 182.6394 - val\_mean\_absolute\_percentage\_error: 182.6394

Epoch 47/100

22/22 [==============================] - 0s 925us/step - loss: 173.0102 - mean\_absolute\_percentage\_error: 173.0102 - val\_loss: 182.3919 - val\_mean\_absolute\_percentage\_error: 182.3919

Epoch 48/100

22/22 [==============================] - 0s 2ms/step - loss: 172.8559 - mean\_absolute\_percentage\_error: 172.8559 - val\_loss: 182.1002 - val\_mean\_absolute\_percentage\_error: 182.1002

Epoch 49/100

22/22 [==============================] - 0s 6ms/step - loss: 172.6654 - mean\_absolute\_percentage\_error: 172.6654 - val\_loss: 181.7935 - val\_mean\_absolute\_percentage\_error: 181.7935

Epoch 50/100

22/22 [==============================] - 0s 1ms/step - loss: 172.4415 - mean\_absolute\_percentage\_error: 172.4415 - val\_loss: 181.5168 - val\_mean\_absolute\_percentage\_error: 181.5168

Epoch 51/100

22/22 [==============================] - 0s 3ms/step - loss: 172.2582 - mean\_absolute\_percentage\_error: 172.2582 - val\_loss: 181.2548 - val\_mean\_absolute\_percentage\_error: 181.2548

Epoch 52/100

22/22 [==============================] - 0s 4ms/step - loss: 172.0816 - mean\_absolute\_percentage\_error: 172.0816 - val\_loss: 180.9956 - val\_mean\_absolute\_percentage\_error: 180.9956

Epoch 53/100

22/22 [==============================] - 0s 935us/step - loss: 171.9128 - mean\_absolute\_percentage\_error: 171.9128 - val\_loss: 180.7562 - val\_mean\_absolute\_percentage\_error: 180.7562

Epoch 54/100

22/22 [==============================] - 0s 10ms/step - loss: 171.6989 - mean\_absolute\_percentage\_error: 171.6990 - val\_loss: 180.5475 - val\_mean\_absolute\_percentage\_error: 180.5475

Epoch 55/100

22/22 [==============================] - 0s 6ms/step - loss: 171.5401 - mean\_absolute\_percentage\_error: 171.5401 - val\_loss: 180.3324 - val\_mean\_absolute\_percentage\_error: 180.3324

Epoch 56/100

22/22 [==============================] - 0s 4ms/step - loss: 171.3717 - mean\_absolute\_percentage\_error: 171.3717 - val\_loss: 180.1043 - val\_mean\_absolute\_percentage\_error: 180.1043

Epoch 57/100

22/22 [==============================] - 0s 10ms/step - loss: 171.1908 - mean\_absolute\_percentage\_error: 171.1908 - val\_loss: 179.8431 - val\_mean\_absolute\_percentage\_error: 179.8431

Epoch 58/100

22/22 [==============================] - 0s 2ms/step - loss: 171.0092 - mean\_absolute\_percentage\_error: 171.0092 - val\_loss: 179.5952 - val\_mean\_absolute\_percentage\_error: 179.5952

Epoch 59/100

22/22 [==============================] - 0s 1ms/step - loss: 170.7915 - mean\_absolute\_percentage\_error: 170.7915 - val\_loss: 179.3444 - val\_mean\_absolute\_percentage\_error: 179.3444

Epoch 60/100

22/22 [==============================] - 0s 3ms/step - loss: 170.6052 - mean\_absolute\_percentage\_error: 170.6052 - val\_loss: 179.0958 - val\_mean\_absolute\_percentage\_error: 179.0958

Epoch 61/100

22/22 [==============================] - 0s 791us/step - loss: 170.4008 - mean\_absolute\_percentage\_error: 170.4008 - val\_loss: 178.8410 - val\_mean\_absolute\_percentage\_error: 178.8410

Epoch 62/100

22/22 [==============================] - 0s 716us/step - loss: 170.1643 - mean\_absolute\_percentage\_error: 170.1643 - val\_loss: 178.5821 - val\_mean\_absolute\_percentage\_error: 178.5821

Epoch 63/100

22/22 [==============================] - 0s 2ms/step - loss: 169.9403 - mean\_absolute\_percentage\_error: 169.9403 - val\_loss: 178.3250 - val\_mean\_absolute\_percentage\_error: 178.3250

Epoch 64/100

22/22 [==============================] - 0s 5ms/step - loss: 169.7299 - mean\_absolute\_percentage\_error: 169.7299 - val\_loss: 178.0648 - val\_mean\_absolute\_percentage\_error: 178.0648

Epoch 65/100

22/22 [==============================] - 0s 3ms/step - loss: 169.5011 - mean\_absolute\_percentage\_error: 169.5011 - val\_loss: 177.8103 - val\_mean\_absolute\_percentage\_error: 177.8103

Epoch 66/100

22/22 [==============================] - 0s 838us/step - loss: 169.2620 - mean\_absolute\_percentage\_error: 169.2620 - val\_loss: 177.5540 - val\_mean\_absolute\_percentage\_error: 177.5540

Epoch 67/100

22/22 [==============================] - 0s 3ms/step - loss: 169.0495 - mean\_absolute\_percentage\_error: 169.0495 - val\_loss: 177.3001 - val\_mean\_absolute\_percentage\_error: 177.3001

Epoch 68/100

22/22 [==============================] - 0s 2ms/step - loss: 168.8201 - mean\_absolute\_percentage\_error: 168.8201 - val\_loss: 177.0509 - val\_mean\_absolute\_percentage\_error: 177.0509

Epoch 69/100

22/22 [==============================] - 0s 5ms/step - loss: 168.6070 - mean\_absolute\_percentage\_error: 168.6070 - val\_loss: 176.7875 - val\_mean\_absolute\_percentage\_error: 176.7875

Epoch 70/100

22/22 [==============================] - 0s 1ms/step - loss: 168.3592 - mean\_absolute\_percentage\_error: 168.3592 - val\_loss: 176.5056 - val\_mean\_absolute\_percentage\_error: 176.5056

Epoch 71/100

22/22 [==============================] - 0s 838us/step - loss: 168.1092 - mean\_absolute\_percentage\_error: 168.1092 - val\_loss: 176.2180 - val\_mean\_absolute\_percentage\_error: 176.2180

Epoch 72/100

22/22 [==============================] - 0s 4ms/step - loss: 167.8437 - mean\_absolute\_percentage\_error: 167.8437 - val\_loss: 175.9706 - val\_mean\_absolute\_percentage\_error: 175.9706

Epoch 73/100

22/22 [==============================] - 0s 2ms/step - loss: 167.7013 - mean\_absolute\_percentage\_error: 167.7013 - val\_loss: 175.7846 - val\_mean\_absolute\_percentage\_error: 175.7846

Epoch 74/100

22/22 [==============================] - 0s 3ms/step - loss: 167.5415 - mean\_absolute\_percentage\_error: 167.5415 - val\_loss: 175.6645 - val\_mean\_absolute\_percentage\_error: 175.6645

Epoch 75/100

22/22 [==============================] - 0s 3ms/step - loss: 167.3878 - mean\_absolute\_percentage\_error: 167.3878 - val\_loss: 175.5576 - val\_mean\_absolute\_percentage\_error: 175.5576

Epoch 76/100

22/22 [==============================] - 0s 1ms/step - loss: 167.2329 - mean\_absolute\_percentage\_error: 167.2329 - val\_loss: 175.4481 - val\_mean\_absolute\_percentage\_error: 175.4481

Epoch 77/100

22/22 [==============================] - 0s 3ms/step - loss: 167.1107 - mean\_absolute\_percentage\_error: 167.1107 - val\_loss: 175.3364 - val\_mean\_absolute\_percentage\_error: 175.3364

Epoch 78/100

22/22 [==============================] - 0s 5ms/step - loss: 166.9424 - mean\_absolute\_percentage\_error: 166.9424 - val\_loss: 175.2484 - val\_mean\_absolute\_percentage\_error: 175.2484

Epoch 79/100

22/22 [==============================] - 0s 2ms/step - loss: 166.8162 - mean\_absolute\_percentage\_error: 166.8162 - val\_loss: 175.1311 - val\_mean\_absolute\_percentage\_error: 175.1311

Epoch 80/100

22/22 [==============================] - 0s 2ms/step - loss: 166.6659 - mean\_absolute\_percentage\_error: 166.6659 - val\_loss: 175.0051 - val\_mean\_absolute\_percentage\_error: 175.0051

Epoch 81/100

22/22 [==============================] - 0s 4ms/step - loss: 166.4984 - mean\_absolute\_percentage\_error: 166.4984 - val\_loss: 174.9110 - val\_mean\_absolute\_percentage\_error: 174.9110

Epoch 82/100

22/22 [==============================] - 0s 3ms/step - loss: 166.3575 - mean\_absolute\_percentage\_error: 166.3575 - val\_loss: 174.8305 - val\_mean\_absolute\_percentage\_error: 174.8305

Epoch 83/100

22/22 [==============================] - 0s 1ms/step - loss: 166.2053 - mean\_absolute\_percentage\_error: 166.2054 - val\_loss: 174.7408 - val\_mean\_absolute\_percentage\_error: 174.7408

Epoch 84/100

22/22 [==============================] - 0s 6ms/step - loss: 166.0520 - mean\_absolute\_percentage\_error: 166.0520 - val\_loss: 174.6643 - val\_mean\_absolute\_percentage\_error: 174.6643

Epoch 85/100

22/22 [==============================] - 0s 3ms/step - loss: 165.9261 - mean\_absolute\_percentage\_error: 165.9261 - val\_loss: 174.5774 - val\_mean\_absolute\_percentage\_error: 174.5774

Epoch 86/100

22/22 [==============================] - 0s 4ms/step - loss: 165.7523 - mean\_absolute\_percentage\_error: 165.7523 - val\_loss: 174.4951 - val\_mean\_absolute\_percentage\_error: 174.4951

Epoch 87/100

22/22 [==============================] - 0s 3ms/step - loss: 165.6117 - mean\_absolute\_percentage\_error: 165.6117 - val\_loss: 174.3775 - val\_mean\_absolute\_percentage\_error: 174.3775

Epoch 88/100

22/22 [==============================] - 0s 5ms/step - loss: 165.4746 - mean\_absolute\_percentage\_error: 165.4746 - val\_loss: 174.2242 - val\_mean\_absolute\_percentage\_error: 174.2242

Epoch 89/100

22/22 [==============================] - 0s 3ms/step - loss: 165.2764 - mean\_absolute\_percentage\_error: 165.2764 - val\_loss: 174.0912 - val\_mean\_absolute\_percentage\_error: 174.0912

Epoch 90/100

22/22 [==============================] - 0s 8ms/step - loss: 165.0702 - mean\_absolute\_percentage\_error: 165.0702 - val\_loss: 173.9679 - val\_mean\_absolute\_percentage\_error: 173.9679

Epoch 91/100

22/22 [==============================] - 0s 806us/step - loss: 164.8980 - mean\_absolute\_percentage\_error: 164.8980 - val\_loss: 173.8046 - val\_mean\_absolute\_percentage\_error: 173.8046

Epoch 92/100

22/22 [==============================] - 0s 2ms/step - loss: 164.6949 - mean\_absolute\_percentage\_error: 164.6949 - val\_loss: 173.6451 - val\_mean\_absolute\_percentage\_error: 173.6451

Epoch 93/100

22/22 [==============================] - 0s 4ms/step - loss: 164.5004 - mean\_absolute\_percentage\_error: 164.5004 - val\_loss: 173.5011 - val\_mean\_absolute\_percentage\_error: 173.5011

Epoch 94/100

22/22 [==============================] - 0s 460us/step - loss: 164.3378 - mean\_absolute\_percentage\_error: 164.3378 - val\_loss: 173.3816 - val\_mean\_absolute\_percentage\_error: 173.3816

Epoch 95/100

22/22 [==============================] - 0s 5ms/step - loss: 164.0807 - mean\_absolute\_percentage\_error: 164.0807 - val\_loss: 173.3374 - val\_mean\_absolute\_percentage\_error: 173.3374

Epoch 96/100

22/22 [==============================] - 0s 9ms/step - loss: 163.9238 - mean\_absolute\_percentage\_error: 163.9238 - val\_loss: 173.2563 - val\_mean\_absolute\_percentage\_error: 173.2563

Epoch 97/100

22/22 [==============================] - 0s 4ms/step - loss: 163.7277 - mean\_absolute\_percentage\_error: 163.7277 - val\_loss: 173.1539 - val\_mean\_absolute\_percentage\_error: 173.1539

Epoch 98/100

22/22 [==============================] - 0s 11ms/step - loss: 163.5421 - mean\_absolute\_percentage\_error: 163.5421 - val\_loss: 173.0358 - val\_mean\_absolute\_percentage\_error: 173.0358

Epoch 99/100

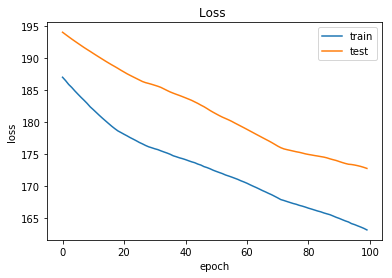
22/22 [==============================] - 0s 5ms/step - loss: 163.3370 - mean\_absolute\_percentage\_error: 163.3370 - val\_loss: 172.8905 - val\_mean\_absolute\_percentage\_error: 172.8905

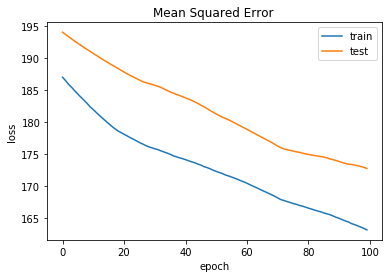
Epoch 100/100

22/22 [==============================] - 0s 4ms/step - loss: 163.0935 - mean\_absolute\_percentage\_error: 163.0935 - val\_loss: 172.7256 - val\_mean\_absolute\_percentage\_error: 172.7256

22/22 [==============================] - 0s 486us/step

8/8 [==============================] - 0s 5ms/step

****

****

**Conclusion**

Thus the following things were executed :

* Plotting a data visualization on the given dataset
* Plotting Surface Plots
* Plotting TriSurface Plots
* Plotting Contour Plots
* A successful Working Neural Network Architecture For Corrosion Resistance
* Training and Testing the NN model
* Minimization of Mean Square Error
* Minimization of Mean Absolute Percentage Error

The above mentioned tasks were executed and tested and the observations were noted down as described above.