```
# Importing the required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import os
import warnings
from sklearn import preprocessing
warnings.filterwarnings("ignore")

# Read the Excel file
df=pd.read_csv('/content/CC_GENERAL.csv')

# Display the  dataframe
df.head()
```

```
  CUST_ID      BALANCE  BALANCE_FREQUENCY  PURCHASES  ONEOFF_PURCHASES
\
0  C10001    40.900749           0.818182      95.40              0.00

1  C10002  3202.467416           0.909091       0.00              0.00

2  C10003  2495.148862           1.000000     773.17            773.17

3  C10004  1666.670542           0.636364    1499.00           1499.00

4  C10005   817.714335           1.000000      16.00             16.00


   INSTALLMENTS_PURCHASES  CASH_ADVANCE  PURCHASES_FREQUENCY  \
0                    95.4      0.000000             0.166667
1                     0.0   6442.945483             0.000000
2                     0.0      0.000000             1.000000
3                     0.0    205.788017             0.083333
4                     0.0      0.000000             0.083333

   ONEOFF_PURCHASES_FREQUENCY  PURCHASES_INSTALLMENTS_FREQUENCY  \
0                    0.000000                          0.083333
1                    0.000000                          0.000000
2                    1.000000                          0.000000
3                    0.083333                          0.000000
4                    0.083333                          0.000000

   CASH_ADVANCE_FREQUENCY  CASH_ADVANCE_TRX  PURCHASES_TRX
CREDIT_LIMIT  \
0                0.000000                 0              2
1000.0
1                0.250000                 4              0
7000.0
2                0.000000                 0             12
```

```
7500.0
3                0.083333                 1              1
7500.0
4                0.000000                 0              1
1200.0

      PAYMENTS  MINIMUM_PAYMENTS  PRC_FULL_PAYMENT  TENURE
0   201.802084        139.509787          0.000000      12
1  4103.032597       1072.340217          0.222222      12
2   622.066742        627.284787          0.000000      12
3     0.000000               NaN          0.000000      12
4   678.334763        244.791237          0.000000      12
```

```
# Print the few rows from the dataframe
print(df.head(5))
```

```
  CUST_ID      BALANCE  BALANCE_FREQUENCY  PURCHASES  ONEOFF_PURCHASES
\
0  C10001    40.900749           0.818182      95.40              0.00

1  C10002  3202.467416           0.909091       0.00              0.00

2  C10003  2495.148862           1.000000     773.17            773.17

3  C10004  1666.670542           0.636364    1499.00           1499.00

4  C10005   817.714335           1.000000      16.00             16.00


   INSTALLMENTS_PURCHASES  CASH_ADVANCE  PURCHASES_FREQUENCY  \
0                    95.4      0.000000             0.166667
1                     0.0   6442.945483             0.000000
2                     0.0      0.000000             1.000000
3                     0.0    205.788017             0.083333
4                     0.0      0.000000             0.083333

   ONEOFF_PURCHASES_FREQUENCY  PURCHASES_INSTALLMENTS_FREQUENCY  \
0                    0.000000                          0.083333
1                    0.000000                          0.000000
2                    1.000000                          0.000000
3                    0.083333                          0.000000
4                    0.083333                          0.000000

   CASH_ADVANCE_FREQUENCY  CASH_ADVANCE_TRX  PURCHASES_TRX
CREDIT_LIMIT  \
0                0.000000                 0              2
1000.0
1                0.250000                 4              0
7000.0
2                0.000000                 0             12
7500.0
```

```
3                0.083333                    1               1
7500.0
4                0.000000                    0               1
1200.0
```

```
      PAYMENTS  MINIMUM_PAYMENTS  PRC_FULL_PAYMENT  TENURE
0   201.802084        139.509787          0.000000      12
1  4103.032597       1072.340217          0.222222      12
2   622.066742        627.284787          0.000000      12
3     0.000000               NaN          0.000000      12
4   678.334763        244.791237          0.000000      12
```

```
df. isnull(). sum()
```

```
CUST_ID                             0
BALANCE                             0
BALANCE_FREQUENCY                   0
PURCHASES                           0
ONEOFF_PURCHASES                    0
INSTALLMENTS_PURCHASES              0
CASH_ADVANCE                        0
PURCHASES_FREQUENCY                 0
ONEOFF_PURCHASES_FREQUENCY          0
PURCHASES_INSTALLMENTS_FREQUENCY    0
CASH_ADVANCE_FREQUENCY              0
CASH_ADVANCE_TRX                    0
PURCHASES_TRX                       0
CREDIT_LIMIT                        1
PAYMENTS                            0
MINIMUM_PAYMENTS                  313
PRC_FULL_PAYMENT                    0
TENURE                              0
dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8950 entries, 0 to 8949
Data columns (total 18 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   CUST_ID                           8950 non-null   object
 1   BALANCE                           8950 non-null   float64
 2   BALANCE_FREQUENCY                 8950 non-null   float64
 3   PURCHASES                         8950 non-null   float64
 4   ONEOFF_PURCHASES                  8950 non-null   float64
 5   INSTALLMENTS_PURCHASES            8950 non-null   float64
 6   CASH_ADVANCE                      8950 non-null   float64
 7   PURCHASES_FREQUENCY               8950 non-null   float64
 8   ONEOFF_PURCHASES_FREQUENCY        8950 non-null   float64
```

```
 9   PURCHASES_INSTALLMENTS_FREQUENCY  8950 non-null   float64
 10  CASH_ADVANCE_FREQUENCY            8950 non-null   float64
 11  CASH_ADVANCE_TRX                  8950 non-null   int64
 12  PURCHASES_TRX                     8950 non-null   int64
 13  CREDIT_LIMIT                      8949 non-null   float64
 14  PAYMENTS                          8950 non-null   float64
 15  MINIMUM_PAYMENTS                  8637 non-null   float64
 16  PRC_FULL_PAYMENT                  8950 non-null   float64
 17  TENURE                            8950 non-null   int64
dtypes: float64(14), int64(3), object(1)
memory usage: 1.2+ MB
```

```
df.shape
```

```
(8950, 18)
```

```
df.describe().T
```

```
                                   count         mean           std
min  \
BALANCE                           8950.0  1564.474828  2081.531879
0.000000
BALANCE_FREQUENCY                 8950.0     0.877271     0.236904
0.000000
PURCHASES                         8950.0  1003.204834  2136.634782
0.000000
ONEOFF_PURCHASES                  8950.0   592.437371  1659.887917
0.000000
INSTALLMENTS_PURCHASES            8950.0   411.067645   904.338115
0.000000
CASH_ADVANCE                      8950.0   978.871112  2097.163877
0.000000
PURCHASES_FREQUENCY               8950.0     0.490351     0.401371
0.000000
ONEOFF_PURCHASES_FREQUENCY        8950.0     0.202458     0.298336
0.000000
PURCHASES_INSTALLMENTS_FREQUENCY  8950.0     0.364437     0.397448
0.000000
CASH_ADVANCE_FREQUENCY            8950.0     0.135144     0.200121
0.000000
CASH_ADVANCE_TRX                  8950.0     3.248827     6.824647
0.000000
PURCHASES_TRX                     8950.0    14.709832    24.857649
0.000000
CREDIT_LIMIT                      8949.0  4494.449450  3638.815725
50.000000
PAYMENTS                          8950.0  1733.143852  2895.063757
0.000000
MINIMUM_PAYMENTS                  8637.0   864.206542  2372.446607
0.019163
```

```
PRC_FULL_PAYMENT                  8950.0     0.153715     0.292499
0.000000
TENURE                            8950.0    11.517318     1.338331
6.000000

                                        25%          50%
75%  \
BALANCE                          128.281915   873.385231
2054.140036
BALANCE_FREQUENCY                  0.888889     1.000000
1.000000
PURCHASES                         39.635000   361.280000
1110.130000
ONEOFF_PURCHASES                   0.000000    38.000000
577.405000
INSTALLMENTS_PURCHASES             0.000000    89.000000
468.637500
CASH_ADVANCE                       0.000000     0.000000
1113.821139
PURCHASES_FREQUENCY                0.083333     0.500000
0.916667
ONEOFF_PURCHASES_FREQUENCY         0.000000     0.083333
0.300000
PURCHASES_INSTALLMENTS_FREQUENCY   0.000000     0.166667
0.750000
CASH_ADVANCE_FREQUENCY             0.000000     0.000000
0.222222
CASH_ADVANCE_TRX                   0.000000     0.000000
4.000000
PURCHASES_TRX                      1.000000     7.000000
17.000000
CREDIT_LIMIT                    1600.000000  3000.000000
6500.000000
PAYMENTS                         383.276166   856.901546
1901.134317
MINIMUM_PAYMENTS                 169.123707   312.343947
825.485459
PRC_FULL_PAYMENT                   0.000000     0.000000
0.142857
TENURE                            12.000000    12.000000
12.000000

                                       max
BALANCE                         19043.13856
BALANCE_FREQUENCY                   1.00000
PURCHASES                       49039.57000
ONEOFF_PURCHASES                40761.25000
INSTALLMENTS_PURCHASES          22500.00000
CASH_ADVANCE                    47137.21176
```

```
PURCHASES_FREQUENCY                     1.00000
ONEOFF_PURCHASES_FREQUENCY              1.00000
PURCHASES_INSTALLMENTS_FREQUENCY        1.00000
CASH_ADVANCE_FREQUENCY                  1.50000
CASH_ADVANCE_TRX                      123.00000
PURCHASES_TRX                         358.00000
CREDIT_LIMIT                        30000.00000
PAYMENTS                            50721.48336
MINIMUM_PAYMENTS                    76406.20752
PRC_FULL_PAYMENT                        1.00000
TENURE                                 12.00000
```

```
X = df.iloc[:, 3:7]
y = df.iloc[:, 7]
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size =
0.3)
print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(6265, 4)
(2685, 4)
(6265,)
(2685,)
```

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=.2,random
_state=0)
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
from sklearn.model_selection import cross_val_score
from sklearn import preprocessing
from sklearn.linear_model import Lasso
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
```

```
from sklearn.linear_model import LassoCV
from sklearn.metrics import r2_score

from sklearn.preprocessing import StandardScaler
sc=StandardScaler()

X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)

# importing the keras
import keras
from keras.models import Sequential
from keras.layers import Dense
# importing the required activation functions
from keras.layers import LeakyReLU,PReLU,ELU

classifier=Sequential()
 # Add the input layer and the first hidden layer
# The Units are output␣dimension(neurons), keral_intializers are
weight intilalization technqiues.
classifier.add(Dense(units = 8, kernel_initializer
='he_uniform',activation='relu',input_dim = 3))
 # Add the second input layer and the first hidden layer
 # For the relu activation function we can use he_uniform or he_normal
classifier.add(Dense(units = 6, kernel_initializer
='he_uniform',activation='relu'))
 # Adding the output layer
classifier.add(Dense(units=1,kernel_initializer='glorot_uniform',activ
ation="sigmoid"))

classifier.compile(optimizer='Adamax',loss="binary_crossentropy",metri
cs=["accuracy"])

ann = Sequential([Dense(25, activation="relu"),
                  Dense(15, activation='relu'),
                  Dense(1, activation='linear')])

ann.compile(optimizer='adam',
            loss='mean_squared_error',
            metrics=['mean_absolute_error'])


history = ann.fit(X_train, y_train, epochs=100, validation_split=0.2)

Epoch 1/100
179/179 [==============================] - 1s 3ms/step - loss:
10683.0244 - mean_absolute_error: 40.2267 - val_loss: 1412.5137 -
val_mean_absolute_error: 18.6701
Epoch 2/100
179/179 [==============================] - 0s 2ms/step - loss:
590.7090 - mean_absolute_error: 10.8253 - val_loss: 307.5636 -
```

```
val_mean_absolute_error: 6.9332
Epoch 3/100
179/179 [==============================] - 0s 2ms/step - loss:
442.2806 - mean_absolute_error: 8.7208 - val_loss: 181.3344 -
val_mean_absolute_error: 6.8326
Epoch 4/100
179/179 [==============================] - 0s 3ms/step - loss:
367.9572 - mean_absolute_error: 8.5399 - val_loss: 141.9806 -
val_mean_absolute_error: 5.4607
Epoch 5/100
179/179 [==============================] - 0s 3ms/step - loss:
275.2189 - mean_absolute_error: 7.1626 - val_loss: 95.3480 -
val_mean_absolute_error: 4.2944
Epoch 6/100
179/179 [==============================] - 0s 3ms/step - loss: 99.1643
- mean_absolute_error: 4.2673 - val_loss: 50.1666 -
val_mean_absolute_error: 3.0250
Epoch 7/100
179/179 [==============================] - 0s 2ms/step - loss:
224.6519 - mean_absolute_error: 5.4786 - val_loss: 113.3728 -
val_mean_absolute_error: 5.1210
Epoch 8/100
179/179 [==============================] - 0s 3ms/step - loss: 93.3821
- mean_absolute_error: 4.1918 - val_loss: 66.9402 -
val_mean_absolute_error: 3.7248
Epoch 9/100
179/179 [==============================] - 1s 4ms/step - loss:
233.2860 - mean_absolute_error: 6.2765 - val_loss: 19.5429 -
val_mean_absolute_error: 1.8852
Epoch 10/100
179/179 [==============================] - 1s 4ms/step - loss: 21.9522
- mean_absolute_error: 1.9194 - val_loss: 50.8204 -
val_mean_absolute_error: 3.2031
Epoch 11/100
179/179 [==============================] - 1s 3ms/step - loss: 57.4103
- mean_absolute_error: 2.9428 - val_loss: 13.4273 -
val_mean_absolute_error: 1.5308
Epoch 12/100
179/179 [==============================] - 1s 4ms/step - loss: 14.6849
- mean_absolute_error: 1.6963 - val_loss: 27.2659 -
val_mean_absolute_error: 2.5682
Epoch 13/100
179/179 [==============================] - 0s 2ms/step - loss: 64.3696
- mean_absolute_error: 3.1564 - val_loss: 14.0962 -
val_mean_absolute_error: 1.4170
Epoch 14/100
179/179 [==============================] - 0s 2ms/step - loss:
2512.9985 - mean_absolute_error: 14.7521 - val_loss: 36483.4258 -
val_mean_absolute_error: 93.3086
```

```
Epoch 15/100
179/179 [==============================] - 0s 2ms/step - loss:
1337.8169 - mean_absolute_error: 10.5635 - val_loss: 42.1939 -
val_mean_absolute_error: 3.3940
Epoch 16/100
179/179 [==============================] - 0s 2ms/step - loss: 24.5449
- mean_absolute_error: 2.2450 - val_loss: 16.6926 -
val_mean_absolute_error: 2.1897
Epoch 17/100
179/179 [==============================] - 0s 2ms/step - loss: 22.3985
- mean_absolute_error: 1.9889 - val_loss: 12.5305 -
val_mean_absolute_error: 1.4068
Epoch 18/100
179/179 [==============================] - 1s 3ms/step - loss: 17.0427
- mean_absolute_error: 1.7158 - val_loss: 11.1651 -
val_mean_absolute_error: 1.3311
Epoch 19/100
179/179 [==============================] - 0s 2ms/step - loss: 12.0577
- mean_absolute_error: 1.4583 - val_loss: 7.9046 -
val_mean_absolute_error: 1.2848
Epoch 20/100
179/179 [==============================] - 0s 2ms/step - loss: 11.0720
- mean_absolute_error: 1.4067 - val_loss: 12.9346 -
val_mean_absolute_error: 1.8762
Epoch 21/100
179/179 [==============================] - 0s 2ms/step - loss: 74.6066
- mean_absolute_error: 3.2099 - val_loss: 12.4497 -
val_mean_absolute_error: 1.7023
Epoch 22/100
179/179 [==============================] - 0s 2ms/step - loss: 14.6323
- mean_absolute_error: 1.7133 - val_loss: 21.4090 -
val_mean_absolute_error: 2.4417
Epoch 23/100
179/179 [==============================] - 0s 2ms/step - loss: 13.0816
- mean_absolute_error: 1.6486 - val_loss: 9.5239 -
val_mean_absolute_error: 1.5131
Epoch 24/100
179/179 [==============================] - 0s 2ms/step - loss:
416.7830 - mean_absolute_error: 6.4315 - val_loss: 66.8472 -
val_mean_absolute_error: 4.4371
Epoch 25/100
179/179 [==============================] - 0s 2ms/step - loss:
185.0755 - mean_absolute_error: 4.9563 - val_loss: 11.8160 -
val_mean_absolute_error: 1.7918
Epoch 26/100
179/179 [==============================] - 0s 2ms/step - loss: 39.9436
- mean_absolute_error: 2.5972 - val_loss: 3.7689 -
val_mean_absolute_error: 0.8816
Epoch 27/100
```

```
179/179 [==============================] - 0s 2ms/step - loss: 16.3358
- mean_absolute_error: 1.7123 - val_loss: 14.4238 -
val_mean_absolute_error: 1.9116
Epoch 28/100
179/179 [==============================] - 0s 2ms/step - loss: 14.0058
- mean_absolute_error: 1.7396 - val_loss: 10.3126 -
val_mean_absolute_error: 1.5156
Epoch 29/100
179/179 [==============================] - 0s 2ms/step - loss: 27.8712
- mean_absolute_error: 2.4478 - val_loss: 4.3477 -
val_mean_absolute_error: 1.1219
Epoch 30/100
179/179 [==============================] - 0s 2ms/step - loss: 11.5133
- mean_absolute_error: 1.5955 - val_loss: 3.3303 -
val_mean_absolute_error: 0.9358
Epoch 31/100
179/179 [==============================] - 0s 2ms/step - loss: 11.2406
- mean_absolute_error: 1.5827 - val_loss: 6.7149 -
val_mean_absolute_error: 1.2091
Epoch 32/100
179/179 [==============================] - 0s 2ms/step - loss: 15.4504
- mean_absolute_error: 1.3529 - val_loss: 47.4667 -
val_mean_absolute_error: 3.1983
Epoch 33/100
179/179 [==============================] - 0s 2ms/step - loss:
243.0065 - mean_absolute_error: 5.8470 - val_loss: 16.0667 -
val_mean_absolute_error: 2.1377
Epoch 34/100
179/179 [==============================] - 0s 2ms/step - loss: 58.5057
- mean_absolute_error: 3.1442 - val_loss: 8.0621 -
val_mean_absolute_error: 1.5651
Epoch 35/100
179/179 [==============================] - 0s 2ms/step - loss: 46.1626
- mean_absolute_error: 2.9221 - val_loss: 32.8565 -
val_mean_absolute_error: 2.8622
Epoch 36/100
179/179 [==============================] - 1s 3ms/step - loss: 32.5732
- mean_absolute_error: 2.4070 - val_loss: 16.8094 -
val_mean_absolute_error: 2.0145
Epoch 37/100
179/179 [==============================] - 1s 4ms/step - loss: 71.7386
- mean_absolute_error: 3.1229 - val_loss: 89.3355 -
val_mean_absolute_error: 4.8731
Epoch 38/100
179/179 [==============================] - 1s 4ms/step - loss:
441.3957 - mean_absolute_error: 7.6154 - val_loss: 11.7962 -
val_mean_absolute_error: 1.7744
Epoch 39/100
179/179 [==============================] - 1s 4ms/step - loss:
```

```
127.9106 - mean_absolute_error: 4.3824 - val_loss: 6.4472 -
val_mean_absolute_error: 1.2726
Epoch 40/100
179/179 [==============================] - 1s 3ms/step - loss: 42.1532
- mean_absolute_error: 2.9365 - val_loss: 7.2375 -
val_mean_absolute_error: 1.1381
Epoch 41/100
179/179 [==============================] - 0s 2ms/step - loss: 18.9670
- mean_absolute_error: 1.8068 - val_loss: 22.3887 -
val_mean_absolute_error: 2.1220
Epoch 42/100
179/179 [==============================] - 0s 2ms/step - loss: 8.9254
- mean_absolute_error: 1.3846 - val_loss: 4.1762 -
val_mean_absolute_error: 0.9141
Epoch 43/100
179/179 [==============================] - 0s 2ms/step - loss: 7.1895
- mean_absolute_error: 1.2299 - val_loss: 4.0704 -
val_mean_absolute_error: 0.8605
Epoch 44/100
179/179 [==============================] - 0s 3ms/step - loss: 47.4338
- mean_absolute_error: 2.0772 - val_loss: 261.8691 -
val_mean_absolute_error: 8.0752
Epoch 45/100
179/179 [==============================] - 0s 2ms/step - loss:
136.2192 - mean_absolute_error: 4.7935 - val_loss: 26.2176 -
val_mean_absolute_error: 2.6275
Epoch 46/100
179/179 [==============================] - 0s 2ms/step - loss: 10.4769
- mean_absolute_error: 1.4339 - val_loss: 3.0139 -
val_mean_absolute_error: 0.7362
Epoch 47/100
179/179 [==============================] - 0s 2ms/step - loss: 6.7888
- mean_absolute_error: 1.1020 - val_loss: 34.0492 -
val_mean_absolute_error: 3.0488
Epoch 48/100
179/179 [==============================] - 0s 2ms/step - loss: 16.3018
- mean_absolute_error: 1.8830 - val_loss: 6.0569 -
val_mean_absolute_error: 1.2016
Epoch 49/100
179/179 [==============================] - 0s 2ms/step - loss:
108.0194 - mean_absolute_error: 4.2468 - val_loss: 29.5113 -
val_mean_absolute_error: 2.7773
Epoch 50/100
179/179 [==============================] - 1s 3ms/step - loss:
671.3815 - mean_absolute_error: 8.7067 - val_loss: 32.8994 -
val_mean_absolute_error: 2.5922
Epoch 51/100
179/179 [==============================] - 0s 2ms/step - loss: 32.6548
- mean_absolute_error: 2.4948 - val_loss: 5.3507 -
```

```
val_mean_absolute_error: 1.0711
Epoch 52/100
179/179 [==============================] - 0s 2ms/step - loss: 28.3187
- mean_absolute_error: 1.8558 - val_loss: 4.6174 -
val_mean_absolute_error: 0.9541
Epoch 53/100
179/179 [==============================] - 0s 2ms/step - loss: 6.6170
- mean_absolute_error: 1.2087 - val_loss: 6.1178 -
val_mean_absolute_error: 1.2640
Epoch 54/100
179/179 [==============================] - 0s 2ms/step - loss: 10.2256
- mean_absolute_error: 1.5351 - val_loss: 8.6056 -
val_mean_absolute_error: 1.5439
Epoch 55/100
179/179 [==============================] - 0s 3ms/step - loss: 3.7929
- mean_absolute_error: 0.9831 - val_loss: 7.7344 -
val_mean_absolute_error: 1.2296
Epoch 56/100
179/179 [==============================] - 0s 2ms/step - loss: 27.7788
- mean_absolute_error: 2.1240 - val_loss: 7.1517 -
val_mean_absolute_error: 1.1146
Epoch 57/100
179/179 [==============================] - 0s 2ms/step - loss: 6.2934
- mean_absolute_error: 1.1731 - val_loss: 2.7351 -
val_mean_absolute_error: 0.6840
Epoch 58/100
179/179 [==============================] - 0s 2ms/step - loss: 93.6469
- mean_absolute_error: 3.4465 - val_loss: 4.5641 -
val_mean_absolute_error: 1.0143
Epoch 59/100
179/179 [==============================] - 0s 2ms/step - loss: 9.0187
- mean_absolute_error: 1.3915 - val_loss: 6.2775 -
val_mean_absolute_error: 1.2428
Epoch 60/100
179/179 [==============================] - 0s 2ms/step - loss: 6.7204
- mean_absolute_error: 1.1960 - val_loss: 45.5158 -
val_mean_absolute_error: 3.0790
Epoch 61/100
179/179 [==============================] - 0s 2ms/step - loss: 20.2282
- mean_absolute_error: 2.1292 - val_loss: 39.9500 -
val_mean_absolute_error: 3.0411
Epoch 62/100
179/179 [==============================] - 0s 3ms/step - loss:
166.4083 - mean_absolute_error: 5.2546 - val_loss: 152.7737 -
val_mean_absolute_error: 6.3579
Epoch 63/100
179/179 [==============================] - 1s 4ms/step - loss: 41.3072
- mean_absolute_error: 2.5989 - val_loss: 4.3659 -
val_mean_absolute_error: 0.9697
```

```
Epoch 64/100
179/179 [==============================] - 1s 4ms/step - loss: 4.4804
- mean_absolute_error: 1.0162 - val_loss: 8.9315 -
val_mean_absolute_error: 1.5883
Epoch 65/100
179/179 [==============================] - 1s 3ms/step - loss: 13.4855
- mean_absolute_error: 1.5417 - val_loss: 3.6854 -
val_mean_absolute_error: 0.9256
Epoch 66/100
179/179 [==============================] - 1s 4ms/step - loss: 5.1524
- mean_absolute_error: 1.0532 - val_loss: 3.8969 -
val_mean_absolute_error: 1.0035
Epoch 67/100
179/179 [==============================] - 0s 2ms/step - loss: 21.5848
- mean_absolute_error: 2.0810 - val_loss: 20.0567 -
val_mean_absolute_error: 2.1371
Epoch 68/100
179/179 [==============================] - 0s 3ms/step - loss:
188.3291 - mean_absolute_error: 4.2918 - val_loss: 410.8587 -
val_mean_absolute_error: 10.0101
Epoch 69/100
179/179 [==============================] - 0s 3ms/step - loss:
108.7716 - mean_absolute_error: 3.4063 - val_loss: 7.2518 -
val_mean_absolute_error: 1.3897
Epoch 70/100
179/179 [==============================] - 0s 2ms/step - loss: 4.9356
- mean_absolute_error: 1.0211 - val_loss: 6.9975 -
val_mean_absolute_error: 1.2402
Epoch 71/100
179/179 [==============================] - 0s 3ms/step - loss: 3.6501
- mean_absolute_error: 0.9139 - val_loss: 4.5494 -
val_mean_absolute_error: 1.2219
Epoch 72/100
179/179 [==============================] - 0s 2ms/step - loss: 4.2180
- mean_absolute_error: 1.0071 - val_loss: 1.7416 -
val_mean_absolute_error: 0.5814
Epoch 73/100
179/179 [==============================] - 0s 2ms/step - loss: 2.5056
- mean_absolute_error: 0.7972 - val_loss: 2.1289 -
val_mean_absolute_error: 0.6345
Epoch 74/100
179/179 [==============================] - 0s 2ms/step - loss: 6.5822
- mean_absolute_error: 1.2341 - val_loss: 5.5211 -
val_mean_absolute_error: 1.1569
Epoch 75/100
179/179 [==============================] - 1s 3ms/step - loss: 3.7029
- mean_absolute_error: 0.9021 - val_loss: 1.4654 -
val_mean_absolute_error: 0.5929
Epoch 76/100
```

```
179/179 [==============================] - 0s 3ms/step - loss: 7.9483
- mean_absolute_error: 1.3807 - val_loss: 2.6793 -
val_mean_absolute_error: 0.8067
Epoch 77/100
179/179 [==============================] - 1s 3ms/step - loss: 2.8138
- mean_absolute_error: 0.8636 - val_loss: 1.9303 -
val_mean_absolute_error: 0.6652
Epoch 78/100
179/179 [==============================] - 0s 2ms/step - loss: 2.6822
- mean_absolute_error: 0.8208 - val_loss: 1.6862 -
val_mean_absolute_error: 0.6957
Epoch 79/100
179/179 [==============================] - 0s 3ms/step - loss: 4.9873
- mean_absolute_error: 1.0548 - val_loss: 1.5840 -
val_mean_absolute_error: 0.5533
Epoch 80/100
179/179 [==============================] - 0s 2ms/step - loss:
267.7043 - mean_absolute_error: 5.3128 - val_loss: 180.9967 -
val_mean_absolute_error: 6.5651
Epoch 81/100
179/179 [==============================] - 0s 2ms/step - loss: 20.2974
- mean_absolute_error: 1.8493 - val_loss: 3.0808 -
val_mean_absolute_error: 0.9951
Epoch 82/100
179/179 [==============================] - 0s 2ms/step - loss: 27.8220
- mean_absolute_error: 2.1023 - val_loss: 3.9098 -
val_mean_absolute_error: 1.0925
Epoch 83/100
179/179 [==============================] - 0s 2ms/step - loss: 5.5615
- mean_absolute_error: 1.0945 - val_loss: 4.0087 -
val_mean_absolute_error: 0.8878
Epoch 84/100
179/179 [==============================] - 1s 3ms/step - loss: 14.0887
- mean_absolute_error: 1.7332 - val_loss: 18.1754 -
val_mean_absolute_error: 1.7366
Epoch 85/100
179/179 [==============================] - 0s 2ms/step - loss: 12.3751
- mean_absolute_error: 1.5621 - val_loss: 1.7252 -
val_mean_absolute_error: 0.7311
Epoch 86/100
179/179 [==============================] - 0s 2ms/step - loss: 2.0380
- mean_absolute_error: 0.6522 - val_loss: 0.9219 -
val_mean_absolute_error: 0.4644
Epoch 87/100
179/179 [==============================] - 0s 2ms/step - loss: 0.8431
- mean_absolute_error: 0.5016 - val_loss: 4.0015 -
val_mean_absolute_error: 1.0114
Epoch 88/100
179/179 [==============================] - 0s 2ms/step - loss: 1.6365
```

```
- mean_absolute_error: 0.6835 - val_loss: 1.1309 -
val_mean_absolute_error: 0.5514
Epoch 89/100
179/179 [==============================] - 1s 3ms/step - loss: 1.8422
- mean_absolute_error: 0.5885 - val_loss: 1.7393 -
val_mean_absolute_error: 0.7187
Epoch 90/100
179/179 [==============================] - 1s 3ms/step - loss:
135.7390 - mean_absolute_error: 4.6588 - val_loss: 15.5592 -
val_mean_absolute_error: 1.8011
Epoch 91/100
179/179 [==============================] - 1s 3ms/step - loss: 6.8968
- mean_absolute_error: 1.2415 - val_loss: 2.8007 -
val_mean_absolute_error: 0.9123
Epoch 92/100
179/179 [==============================] - 1s 4ms/step - loss: 26.8753
- mean_absolute_error: 1.3682 - val_loss: 211.9412 -
val_mean_absolute_error: 6.6281
Epoch 93/100
179/179 [==============================] - 1s 3ms/step - loss: 21.4068
- mean_absolute_error: 1.7086 - val_loss: 1.5433 -
val_mean_absolute_error: 0.6022
Epoch 94/100
179/179 [==============================] - 0s 2ms/step - loss: 4.4831
- mean_absolute_error: 0.8963 - val_loss: 1.8388 -
val_mean_absolute_error: 0.7111
Epoch 95/100
179/179 [==============================] - 0s 2ms/step - loss: 1.4177
- mean_absolute_error: 0.6453 - val_loss: 1.5164 -
val_mean_absolute_error: 0.6838
Epoch 96/100
179/179 [==============================] - 0s 2ms/step - loss: 1.7745
- mean_absolute_error: 0.6872 - val_loss: 5.4907 -
val_mean_absolute_error: 1.1700
Epoch 97/100
179/179 [==============================] - 0s 3ms/step - loss:
252.0938 - mean_absolute_error: 4.2420 - val_loss: 15.7147 -
val_mean_absolute_error: 1.9938
Epoch 98/100
179/179 [==============================] - 1s 3ms/step - loss: 9.8874
- mean_absolute_error: 1.3442 - val_loss: 3.4503 -
val_mean_absolute_error: 0.8949
Epoch 99/100
179/179 [==============================] - 0s 2ms/step - loss: 4.5343
- mean_absolute_error: 0.9222 - val_loss: 2.5472 -
val_mean_absolute_error: 0.7789
Epoch 100/100
179/179 [==============================] - 0s 2ms/step - loss: 3.5107
```

```
- mean_absolute_error: 0.8058 - val_loss: 3.0279 -
val_mean_absolute_error: 0.7781
```