```python
import numpy as np
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score
from tensorflow.keras.optimizers import Adam
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras import Model, Sequential
from tensorflow.keras.layers import Dense, Dropout
from sklearn.model_selection import train_test_split
from tensorflow.keras.losses import MeanSquaredLogarithmicError

PATH_TO_DATA = 
'http://storage.googleapis.com/download.tensorflow.org/data/ecg.csv'
data = pd.read_csv(PATH_TO_DATA, header=None)
data.head()
```

```
        0         1         2         3         4         5         6  
\
0 -0.112522 -2.827204 -3.773897 -4.349751 -4.376041 -3.474986 -
2.181408
1 -1.100878 -3.996840 -4.285843 -4.506579 -4.022377 -3.234368 -
1.566126
2 -0.567088 -2.593450 -3.874230 -4.584095 -4.187449 -3.151462 -
1.742940
3  0.490473 -1.914407 -3.616364 -4.318823 -4.268016 -3.881110 -
2.993280
4  0.800232 -0.874252 -2.384761 -3.973292 -4.338224 -3.802422 -
2.534510

        7         8         9    ...       131       132       133  
134  \
0 -1.818286 -1.250522 -0.477492  ...  0.792168  0.933541  0.796958  
0.578621
1 -0.992258 -0.754680  0.042321  ...  0.538356  0.656881  0.787490  
0.724046
2 -1.490659 -1.183580 -0.394229  ...  0.886073  0.531452  0.311377 -
0.021919
3 -1.671131 -1.333884 -0.965629  ...  0.350816  0.499111  0.600345  
0.842069
4 -1.783423 -1.594450 -0.753199  ...  1.148884  0.958434  1.059025  
1.371682

        135       136       137       138       139  140
0  0.257740  0.228077  0.123431  0.925286  0.193137  1.0
1  0.555784  0.476333  0.773820  1.119621 -1.436250  1.0
2 -0.713683 -0.532197  0.321097  0.904227 -0.421797  1.0
3  0.952074  0.990133  1.086798  1.403011 -0.383564  1.0
4  1.277392  0.960304  0.971020  1.614392  1.421456  1.0
```

```
[5 rows x 141 columns]
```

```
data.shape
```

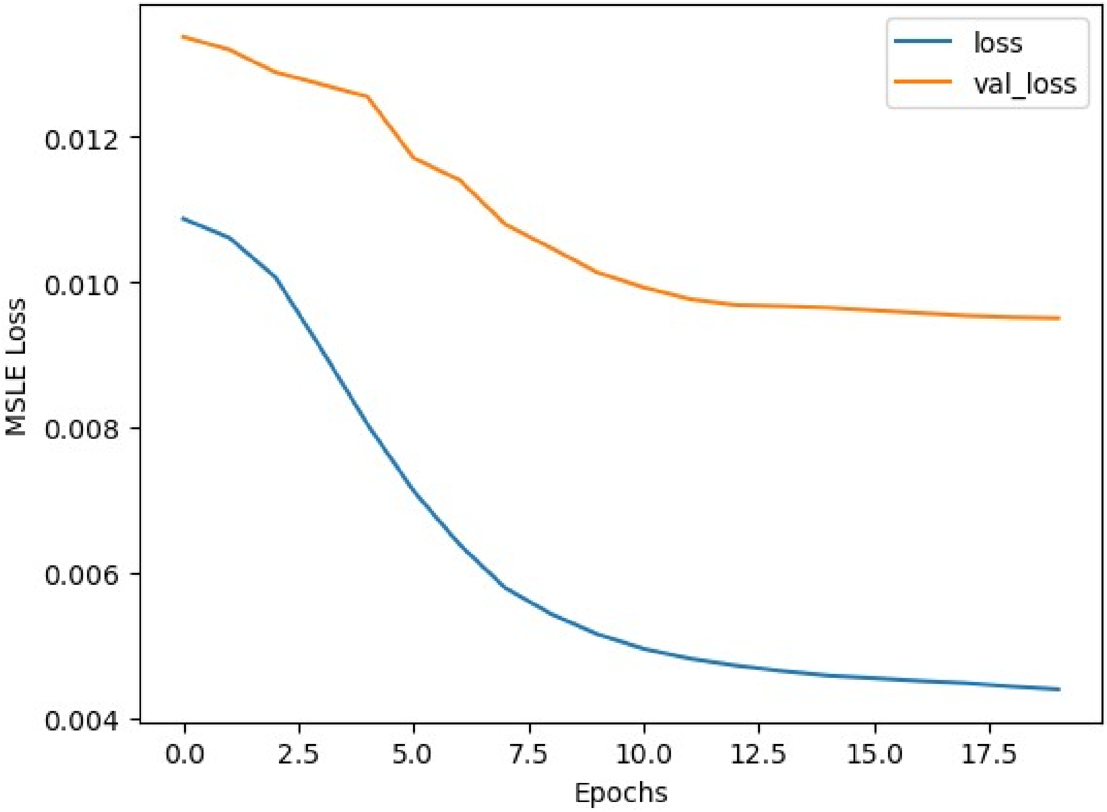
```
(4998, 141)
```

```
features = data.drop(140, axis=1)
target = data[140]
x_train, x_test, y_train, y_test = train_test_split(
 features, target, test_size=0.2, stratify=target
)
train_index = y_train[y_train == 1].index
train_data = x_train.loc[train_index]

min_max_scaler = MinMaxScaler(feature_range=(0, 1))
x_train_scaled = min_max_scaler.fit_transform(train_data.copy())
x_test_scaled = min_max_scaler.transform(x_test.copy())

class AutoEncoder(Model):
 def __init__(self, output_units, ldim=8):
   super().__init__()
   self.encoder = Sequential([
     Dense(64, activation='relu'),
     Dropout(0.1),
     Dense(32, activation='relu'),
     Dropout(0.1),
     Dense(16, activation='relu'),
     Dropout(0.1),
     Dense(ldim, activation='relu')
   ])
   self.decoder = Sequential([
      Dense(16, activation='relu'),
      Dropout(0.1),
      Dense(32, activation='relu'),
      Dropout(0.1),
      Dense(64, activation='relu'),
      Dropout(0.1),
      Dense(output_units, activation='sigmoid')
     ])

 def call(self, inputs):
   encoded = self.encoder(inputs)
   decoded = self.decoder(encoded)
   return decoded

model = AutoEncoder(output_units=x_train_scaled.shape[1])
model.compile(loss='msle', metrics=['mse'], optimizer='adam')
epochs = 20
```

```
history = model.fit(
 x_train_scaled,
 x_train_scaled,
 epochs=epochs,
 batch_size=512,
 validation_data=(x_test_scaled, x_test_scaled)
)

Epoch 1/20
5/5 [==============================] - 4s 149ms/step - loss: 0.0109 -
mse: 0.0244 - val_loss: 0.0134 - val_mse: 0.0311
Epoch 2/20
5/5 [==============================] - 0s 35ms/step - loss: 0.0106 -
mse: 0.0238 - val_loss: 0.0132 - val_mse: 0.0307
Epoch 3/20
5/5 [==============================] - 0s 33ms/step - loss: 0.0101 -
mse: 0.0226 - val_loss: 0.0129 - val_mse: 0.0300
Epoch 4/20
5/5 [==============================] - 0s 34ms/step - loss: 0.0091 -
mse: 0.0203 - val_loss: 0.0127 - val_mse: 0.0295
Epoch 5/20
5/5 [==============================] - 0s 34ms/step - loss: 0.0080 -
mse: 0.0180 - val_loss: 0.0125 - val_mse: 0.0291
Epoch 6/20
5/5 [==============================] - 0s 38ms/step - loss: 0.0071 -
mse: 0.0159 - val_loss: 0.0117 - val_mse: 0.0272
Epoch 7/20
5/5 [==============================] - 0s 34ms/step - loss: 0.0064 -
mse: 0.0143 - val_loss: 0.0114 - val_mse: 0.0265
Epoch 8/20
5/5 [==============================] - 0s 36ms/step - loss: 0.0058 -
mse: 0.0129 - val_loss: 0.0108 - val_mse: 0.0252
Epoch 9/20
5/5 [==============================] - 0s 34ms/step - loss: 0.0054 -
mse: 0.0121 - val_loss: 0.0105 - val_mse: 0.0244
Epoch 10/20
5/5 [==============================] - 0s 35ms/step - loss: 0.0052 -
mse: 0.0115 - val_loss: 0.0101 - val_mse: 0.0237
Epoch 11/20
5/5 [==============================] - 0s 34ms/step - loss: 0.0050 -
mse: 0.0111 - val_loss: 0.0099 - val_mse: 0.0232
Epoch 12/20
5/5 [==============================] - 0s 35ms/step - loss: 0.0048 -
mse: 0.0108 - val_loss: 0.0098 - val_mse: 0.0229
Epoch 13/20
5/5 [==============================] - 0s 36ms/step - loss: 0.0047 -
mse: 0.0106 - val_loss: 0.0097 - val_mse: 0.0227
Epoch 14/20
5/5 [==============================] - 0s 30ms/step - loss: 0.0047 -
mse: 0.0104 - val_loss: 0.0097 - val_mse: 0.0226
```

```
Epoch 15/20
5/5 [==============================] - 0s 35ms/step - loss: 0.0046 -
mse: 0.0102 - val_loss: 0.0096 - val_mse: 0.0226
Epoch 16/20
5/5 [==============================] - 0s 32ms/step - loss: 0.0046 -
mse: 0.0102 - val_loss: 0.0096 - val_mse: 0.0225
Epoch 17/20
5/5 [==============================] - 0s 34ms/step - loss: 0.0045 -
mse: 0.0101 - val_loss: 0.0096 - val_mse: 0.0224
Epoch 18/20
5/5 [==============================] - 0s 36ms/step - loss: 0.0045 -
mse: 0.0100 - val_loss: 0.0095 - val_mse: 0.0224
Epoch 19/20
5/5 [==============================] - 0s 33ms/step - loss: 0.0044 -
mse: 0.0099 - val_loss: 0.0095 - val_mse: 0.0223
Epoch 20/20
5/5 [==============================] - 0s 37ms/step - loss: 0.0044 -
mse: 0.0098 - val_loss: 0.0095 - val_mse: 0.0223
```

```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.xlabel('Epochs')
plt.ylabel('MSLE Loss')
plt.legend(['loss', 'val_loss'])
plt.show()
```


loss
val_loss
MSLE Loss
0.012
0.010
0.008
0.006
0.004
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epochs