In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_selection import *
from sklearn.preprocessing import *
import warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import *
import tensorflow as tf
from sklearn.metrics import *
import joblib
```

```
WARNING:tensorflow:From C:\Users\amith\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\losses.py:
2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cr
oss_entropy instead.
```

In [2]:

```
df_1 = pd.read_csv("train1.csv")
df_2 = pd.read_csv("train2.csv")
df_3 = pd.read_csv("train3.csv")

df = pd.concat([df_1, df_2, df_3],axis=1)

df.head()
```

Out[2]:

	MEAN_RR	MEDIAN_RR	SDRR	RMSSD	SDSD	SDRR_RMSSD	HR	pNN25	pNN50	SD1	...	
0	885.157845	853.763730	140.972741	15.554505	15.553371	9.063146	69.499952	11.133333	0.533333	11.001565	...	15.5
1	939.425371	948.357865	81.317742	12.964439	12.964195	6.272369	64.363150	5.600000	0.000000	9.170129	...	2.1
2	898.186047	907.006860	84.497236	16.305279	16.305274	5.182201	67.450066	13.066667	0.200000	11.533417	...	13.7
3	881.757864	893.460030	90.370537	15.720468	15.720068	5.748591	68.809562	11.800000	0.133333	11.119476	...	18.1
4	809.625331	811.184865	62.766242	19.213819	19.213657	3.266724	74.565728	20.200000	0.200000	13.590641	...	48.2

5 rows × 36 columns

In [3]:

```
df_X = df.drop(columns=["condition"])
df_y = df["condition"]

df_X.columns
```

Out[3]:

```
Index(['MEAN_RR', 'MEDIAN_RR', 'SDRR', 'RMSSD', 'SDSD', 'SDRR_RMSSD', 'HR',
       'pNN25', 'pNN50', 'SD1', 'SD2', 'KURT', 'SKEW', 'MEAN_REL_RR',
       'MEDIAN_REL_RR', 'SDRR_REL_RR', 'RMSSD_REL_RR', 'SDSD_REL_RR',
       'SDRR_RMSSD_REL_RR', 'KURT_REL_RR', 'SKEW_REL_RR', 'VLF', 'VLF_PCT',
       'LF', 'LF_PCT', 'LF_NU', 'HF', 'HF_PCT', 'HF_NU', 'TP', 'LF_HF',
       'HF_LF', 'sampen', 'higuci', 'datasetId'],
      dtype='object')
```

In [4]:

```
important_features = ["MEAN_RR", "RMSSD", "pNN25", "pNN50", "LF", "HF", "LF_HF"]
df_X_1 = df_X[important_features]
df_y_1 = df_y.copy()
```

In [5]:

```
sc = StandardScaler()
X = pd.DataFrame(sc.fit_transform(df_X_1),columns=df_X_1.columns)
y = df_y_1.copy()
```

In [6]:

```
y.unique()
```

Out[6]:

```
array([0, 1, 2], dtype=int64)
```

Testing the model

In [7]:

```
X_train_val, X_test, y_train_val, y_test = train_test_split(X, y, test_size=0.2, random_state=np.random.randint(0,100
X_train, X_val, y_train, y_val = train_test_split(X_train_val, y_train_val, test_size=0.2, random_state=np.random.ra
X_train.shape
```

Out[7]:

```
(236344, 7)
```

In [8]:

```
model_1 = tf.keras.models.Sequential(layers=[
    tf.keras.layers.Dense(units=64, activation="relu", input_shape=(X_train.shape[1],)),
    tf.keras.layers.Dense(units=32, activation="relu"),
    tf.keras.layers.Dense(units=16, activation="relu"),
    tf.keras.layers.Dense(units=8, activation="relu"),
    tf.keras.layers.Dense(units=1, activation="softmax")
])
```

```
WARNING:tensorflow:From C:\Users\amith\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\backend.p
y:873: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.
```

In [9]:
```
model_1.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])
```

```
WARNING:tensorflow:From C:\Users\amith\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\optimizers
\__init__.py:309: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.
```

In [10]:
```
model_1.fit(X_train, y_train, batch_size=32, epochs=100, verbose=1, callbacks=tf.keras.callbacks.EarlyStopping(patien
```

```
Epoch 1/100
WARNING:tensorflow:From C:\Users\amith\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\utils\tf_u
tils.py:492: The name tf.ragged.RaggedTensorValue is deprecated. Please use tf.compat.v1.ragged.RaggedTensorValue ins
tead.

WARNING:tensorflow:From C:\Users\amith\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\engine\bas
e_layer_utils.py:384: The name tf.executing_eagerly_outside_functions is deprecated. Please use tf.compat.v1.executin
g_eagerly_outside_functions instead.

7386/7386 [==============================] - 26s 3ms/step - loss: 0.0000e+00 - accuracy: 0.2862 - val_loss: 0.0000e+0
0 - val_accuracy: 0.2825
Epoch 2/100
7386/7386 [==============================] - 24s 3ms/step - loss: 0.0000e+00 - accuracy: 0.2862 - val_loss: 0.0000e+0
0 - val_accuracy: 0.2825
Epoch 3/100
7386/7386 [==============================] - 25s 3ms/step - loss: 0.0000e+00 - accuracy: 0.2862 - val_loss: 0.0000e+0
0 - val_accuracy: 0.2825
```

Out[10]:
```
<keras.src.callbacks.History at 0x2a9266e2d10>
```

In [11]:
```
cat_cross_entropy_loss, accuracy_score = model_1.evaluate(X_test, y_test)
```

```
2309/2309 [==============================] - 5s 2ms/step - loss: 0.0000e+00 - accuracy: 0.2817
```

In [12]:
```
cat_cross_entropy_loss
```

Out[12]:
```
0.0
```

In [13]:
```
round(accuracy_score,2)
```

Out[13]:
```
0.28
```

In [14]:
```
y_pred = model_1.predict(X_test)
```

```
2309/2309 [==============================] - 5s 2ms/step
```

In [15]:
```
root_mean_squared_error = np.round(np.sqrt(mean_squared_error(y_test, y_pred)),2)
root_mean_squared_error
```

Out[15]:
```
0.85
```

Final Prediction of testing set

In [16]:
```
X_train_2, X_val_2, y_train_2, y_val_2 = train_test_split(X,y,test_size=0.2,random_state=np.random.randint(0,1000))
test_df = pd.read_csv("test.csv")
X_test_2 = pd.DataFrame(sc.fit_transform(test_df[important_features]),columns=important_features)
y_test_2 = test_df["condition"]
```

In [17]:
```
model_2 = tf.keras.models.Sequential(layers=[
    tf.keras.layers.Dense(units=64, activation="relu", input_shape=(X_train_2.shape[1],)),
    tf.keras.layers.Dense(units=32, activation="relu"),
    tf.keras.layers.Dense(units=16, activation="relu"),
    tf.keras.layers.Dense(units=8, activation="relu"),
    tf.keras.layers.Dense(units=1, activation="softmax")
])
```

In [18]:
```
model_2.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])
```

In [19]:
```
model_2.fit(X_train_2, y_train_2, batch_size=32, epochs=100, verbose=1, callbacks=tf.keras.callbacks.EarlyStopping(pa
Epoch 1/100
9233/9233 [==============================] - 34s 3ms/step - loss: 0.0000e+00 - accuracy: 0.2840 - val_loss: 0.0000e+0
0 - val_accuracy: 0.2876
Epoch 2/100
9233/9233 [==============================] - 29s 3ms/step - loss: 0.0000e+00 - accuracy: 0.2840 - val_loss: 0.0000e+0
0 - val_accuracy: 0.2876
Epoch 3/100
9233/9233 [==============================] - 29s 3ms/step - loss: 0.0000e+00 - accuracy: 0.2840 - val_loss: 0.0000e+0
0 - val_accuracy: 0.2876
```

Out[19]:
```
<keras.src.callbacks.History at 0x2a93f558890>
```

In [20]:
```
cat_cross_entropy_loss_2, accuracy_score_2 = model_2.evaluate(X_test_2, y_test_2)
```

```
1283/1283 [==============================] - 3s 2ms/step - loss: 0.0000e+00 - accuracy: 0.2871
```

In [21]:
```
cat_cross_entropy_loss_2
```

Out[21]:
```
0.0
```

In [22]:
```
round(accuracy_score_2,2)
```

Out[22]:
```
0.29
```

In [23]:
```
y_pred_2 = model_2.predict(X_test_2)
```

```
1283/1283 [==============================] - 3s 2ms/step
```

In [24]:
```
root_mean_squared_error_2 = np.round(np.sqrt(mean_squared_error(y_test_2, y_pred_2)),2)
root_mean_squared_error_2
```

Out[24]:
```
0.84
```

In [25]:
```
joblib.dump(value=sc,filename="scaler.joblib",compress=9)
model_2.save("model.h5")
```

In [55]:
```
```

In [55]:
```
```

In [ ]:
```
```