AI Cure: Where AI Meets Healing Touch

Team name: aicure_petrichorai

Team Members:

Nupur Sangwai (7620588298)

Ankur De (9674282229)

1. Introduction

Heart rate prediction from ECG signals is a critical task in healthcare, providing valuable insights into an individual's cardiovascular health. ECG recordings offer a non-invasive and continuous way to capture electrical activity within the heart. By analyzing these recordings, we can extract subtle features that provide insights into heart function and rhythm, including heart rate. This study aims to leverage the power of machine learning to develop models that can predict heart rate directly from ECG-derived data, paving the way for improved health monitoring and potential early detection of cardiac abnormalities.

2. Problem Statement:

This paper investigates the potential of machine learning models to predict individual heart rates based on data derived from Electrocardiogram (ECG) recordings. The goal is to develop a model that can accurately estimate heart rate using features extracted from ECG signals, thereby offering non-invasive and continuous monitoring capabilities.

3. Dataset Description

3.1 Dataset Overview:

The dataset consists of attributes obtained from signals measured through ECG recordings. Each entry in the dataset represents a unique measurement instance, and features include various signal-derived attributes such as MEAN_RR (Mean of RR intervals), MEDIAN_RR (Median of RR intervals), LF (Absolute power of the low-frequency band), etc.

3.2 Features:

The features of the given dataset are as follows:

['VLF', 'VLF_PCT', 'LF', 'LF_PCT', 'LF_NU', 'HF', 'HF_PCT', 'HF_NU',
'TP', 'LF_HF', 'HF_LF', 'SD1', 'SD2', 'sampen', 'higuci', 'condition',
'MEAN_RR', 'MEDIAN_RR', 'SDRR', 'RMSSD', 'SDSD', 'SDRR_RMSSD', 'pNN25',
'pNN50', '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']

3.3 Target Variable:

Heart Rate (HR): The heart rate at the respective time of measurement.

4. Data Preprocessing and EDA

4.1. Data Cleaning

Checked for missing values and outliers in the dataset. There were no features that had null values in the provided dataset.

Ensured that data types are appropriate for analysis. All features were of data type float except for datasetId which was int and condition which was ‘object’.

4.2 Exploratory Data Analysis (EDA):

Analyzed the distribution of each feature using statistical measures and visualizations. Calculated basic statistics (mean, median, standard deviation) to understand the central tendency and dispersion of numerical features. The plots showed that the heart rate for the maximum observations lie between 70-80.

From the histograms for all columns in the dataset, we observe that datasetId has the same value for all the observations in the dataset, so in the further steps we decide to drop it.

Also the ratio of LF (Absolute power of the low-frequency band (0.04 - 0.15 Hz)) to HF (high-frequency band) has most values close to zero, and a few even beyond 5000, there could be skewness due to this.

After this we explored correlations between features and the target variable (HR). We observe that there is a correlation of -1 between Heart Rate and MEAN_RR and MEDIAN_RR. Here

-1 indicates a perfect negative correlation i.e. as one variable increases, the other decreases proportionally. Features like RMSSD_REL_RR, SDSD_REL_RR, HF, HF_PCT have positive correlation, which implies that as HR variable increases, the other tends to increase as well.

5. One-Hot Encoding:

Converted categorical variable, such as 'condition,' into a numerical format using one-hot encoding and ensured the model could effectively interpret categorical information.

6. Models Used and Their Architecture:

6.1. XGBoost

This model utilizes gradient-boosting trees to combine multiple weak learners into a strong ensemble model. This specific architecture could include boosting parameters, number of trees, and feature importance analysis.

We split the data into X_train, X_test, y_train, y_test, where 20% of the data will be used as the test set, and the remaining 80% will be used as the training set.

We proceed by creating an instance of the XGBoost Regressor model, train the model on the training data and then use the predictions to evaluate the performance of the model, by comparing them with the true target values (y_test) using evaluation metric i.e. Mean Squared Error (MSE) and MAE

We get the MSE as `0.26623196850810904` and MAE comes out as `0.3210510026324215`

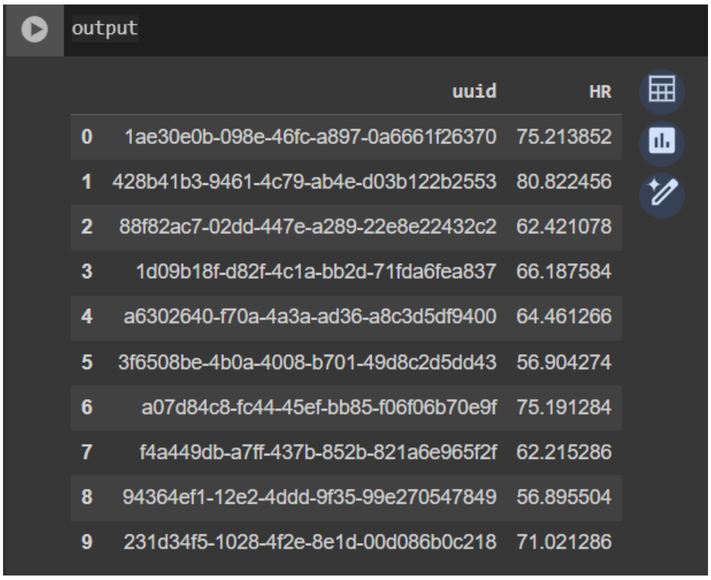
6.2. Random Forest

This model builds a collection of decision trees on random subsets of the features and data, aggregating their predictions for improved accuracy.

We have used the Random Forest model from Tensorflow Decision Forests (TF-DF), RandomForestModel (rf) is trained and then visualized a decision tree of depth=3, the values of MSE and MAE on the validation data come out as 0.3135 and 0.2676. Then the features were sorted from the most important to least important features. The larger the importance score for NUM_AS_ROOT, the more impact it has on the outcome of the model.

Finally the model is evaluated on test data using the evaluation metric i.e. Mean Squared Error (MSE) and Mean Absolute Error (MAE).

It was evaluated on the **sample_test_data.csv** that was provided in the github repository of the main problem statement of aicure. The output values are as follows



```
output
```

	uuid	HR
0	1ae30e0b-098e-46fc-a897-0a6661f26370	75.213852
1	428b41b3-9461-4c79-ab4e-d03b122b2553	80.822456
2	88f82ac7-02dd-447e-a289-22e8e22432c2	62.421078
3	1d09b18f-d82f-4c1a-bb2d-71fda6fea837	66.187584
4	a6302640-f70a-4a3a-ad36-a8c3d5df9400	64.461266
5	3f6508be-4b0a-4008-b701-49d8c2d5dd43	56.904274
6	a07d84c8-fc44-45ef-bb85-f06f06b70e9f	75.191284
7	f4a449db-a7ff-437b-852b-821a6e965f2f	62.215286
8	94364ef1-12e2-4ddd-9f35-99e270547849	56.895504
9	231d34f5-1028-4f2e-8e1d-00d086b0c218	71.021286

The values of MSE for Random Forest is `0.13533647036685723` and MAE is `0.25939088017335477`

6.3. Neural Network

The neural network architecture comprises input, hidden, and output layers. The input layer consists of nodes representing the input features, which in this case is 36, followed by hidden layers with activation functions.

The first hidden layer (Dense(64, activation='relu', input_shape=(36,))) has 64 neurons, uses the Rectified Linear Unit (ReLU) activation function, and expects input data with a shape of (36,). The second hidden layer (Dense(32, activation='relu')) has 32 neurons and uses the ReLU activation function. The output layer (Dense(1)) has a single neuron, since it is a regression task. The output layer produces the predicted heart rate.

The model ran for 150 epochs, with a validation split of 0.1, and the Mean Squared Error (MSE) on the validation set comes out to be 0.16146883368492126.

After testing the model on **sample_test_data.csv** the MSE comes out to be `286.35394287109375`, which clearly states high variance and low bias, we observe overfitting, and the predictions aren't close to the actual values, and the MAE comes out to be `12.51825489054735`

7. Evaluation and Metrics

Mean Squared Error (MSE): Quantifies the average squared difference between predicted and true heart rates.
Mean Absolute Error(MAE) measures the average absolute difference between the predicted values and the actual (ground truth) values.

	MSE	MAE
XGBoost	0.16860733056816837	0.3210510026324215
Random Forest	0.13533647036685723	0.25939088017335477
Neural Network	286.35394287109375	12.51825489054735

8. Results and Discussions

In conclusion, the Random Forest model demonstrated the lowest Mean Squared Error, indicating superior performance in predicting heart rates from ECG signals.
When it comes to the observed MSE values for Neural Networks since the MSE on test data is huge but on validaion data it is low, from this we can say there is high variance and low bias, we observe overfitting, and the predictions aren't close to the actual values, thus we do not go ahead with this approach. The MSE of XGBoost wasn’t promising in the first place so we didn’t move forward with that approach.
Finally we choose the Random Forest as the best approach.

Future Work:

Future work could involve:
Fine-tuning hyperparameters to optimize model performance, especially for XGBoost
We could reduce the overfitting of the Neural Network, by using regularization techniques, or doing feature selection since this dataset has a wide variety of features.

We could also go ahead with investigating the impact of feature scaling and normalization on model outcomes.