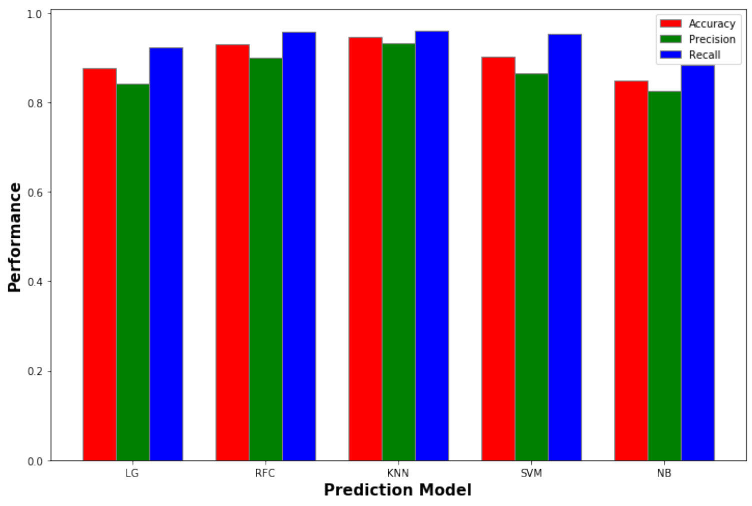
Heart Disease Prediction

Mane Minasyan

August 14, 2022

The purpose of the program is to decide if the patient has heart disease based on the given parameters. To specify the percentage of potential heart disease several models have been evaluated to select the one having the best performance. Since the requirement is to select 5 features for building the model, firstly feature selection has been implemented. The ExtraTreesClassifier has been built for feature selection. Since this method builds random decision trees the algorithm has been run 100 times to calculate the average importance score for each feature. The top 5 features with the highest feature importance are selected for further experiments. The selected features are: cp, ca, exang, thal and oldpeak.

To select the best model Logistic Regression, KNN, SVM, Random Forest and Naive Bayes are compared. The histogram below shows the accuracy, precision and recall for the given models.



Looking at the results the best performing model is KNN. However, since for the given task, it is important to predict the percentage of possible heart disease and the results for the KNN are mostly 0 percent or 100 percent it is a better idea to select the second best model which is Random Forest. The reason for not diverse probabilities is that it estimates is simply a fraction of votes among nearest neighbours. Increasing the number of neighbours will enlarge the variability, but will also decrease the performance of the model.

The function predict-for-instance takes the specified 5 features as an input and outputs the target in JSON format. The output is a predicted probability for the binary targets 0 and 1. The function and its corresponding output are shown below.

In [44]:
```
#The model
def predict_for_instance(js):
    df = pd.DataFrame.from_dict(js,orient='index')
    df=pd.DataFrame(df.T).reindex()
    rand_for = RandomForestClassifier()
    rand_for.fit(X_train,y_train)
    y_pred_rand_for=rand_for.predict_proba(df)
    df['prediction_prob_0']=y_pred_rand_for[0][0]
    df['prediction_prob_1']=y_pred_rand_for[0][1]
    df_1=df.to_json()
    df_json=json.loads(df_1)
    return df_json
```

In [45]:
```
l=predict_for_instance(p)
print(l)
```

```
{'cp': {'0': 2.0}, 'ca': {'0': 1.0}, 'exang': {'0': 0.0}, 'thal': {'0': 3.0}, 'oldpeak': {'0': 0.2}, 'prediction_prob
_0': {'0': 0.2}, 'prediction_prob_1': {'0': 0.8}}
```

The REST API is built using a flask. The system is built on the user input of the features and outputs the probability of having heart disease given the set of parameters. The demo of how to run the application is attached as a video clip.

The biggest limitation of this assignment is time. In case of continuing working on the project, multiple feature selection methods will be evaluated to select the one having the best performance. Feature selection techniques which can be applied include recursive feature elimination(RFE), recursive feature selection algorithms, feature selection based on the Shapley value(cooperative Game Theory) and a few others. Furthermore, further changes can be done based on the discussions with the clients to specify what they want to see.