Indian Institute of Information Technology (IIIT), Kota

Group Presentation, Pattern Recognition (CST403)

# Assessment of Heart Rate and Risk Analysis of Cardiac Disorders using **Supervised Learning**

**Presented by:**
**Sanatan Shrivastava : 2018KUCP1096**
**Yash Singh : 2018KUCP1088**
**Shaktiraj Daudra : 2018KUCP1092**
**Ajay Sharma : 2018KUCP1087**
**Rahul Singh Tanwar : 2018KUCP1043**

**Presented to:**
**Dr. Ankit Sharma**
**Faculty Coordinator,**
**Pattern Recognition (CST403)**
**Indian Institute of Information Technology (IIIT), Kota.**

# Table of contents

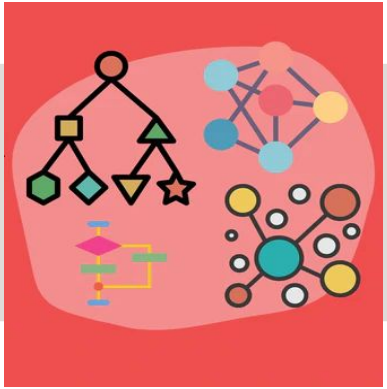
# Project goals

## Goal 1

Picked Features like sex, age etc from the Dataset.

## Goal 2

Pre-Process Features values.



## Goal 3

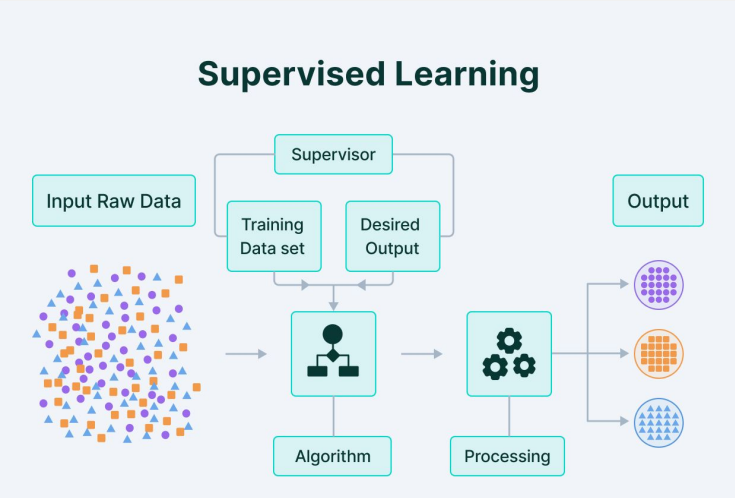
Perform algorithmic analysis using few selected algorithms


## Goal 4

Predict values on the basis of algorithm with maximum accuracy; draw conclusion

# Introduction

- ★ We have conducted an extensive study to find the solution to a classification problem of heart risk analysis and whether there is a probability of cardiac disorder.
- ★ We have used a variety of Machine Learning algorithms.
- ★ This is a classification problem, with input features as a variety of parameters, and the target variable as a binary variable, predicting whether heart disease is present or not.
- ★ The dataset used here is originally available on the University of California, Irvine (UCI) website.
- ★ Dataset used (Originally, at UCI) : https://www.kaggle.com/ronitf/heart-disease-uci



# Dataset

dataset.shape ⇒ (303, 14)

**Features:**

1. **age (AGE):** age
2. **sex (SEX):** 1: male, 0: female
3. **cp:** chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic
4. **trestbps:** resting blood pressure
5. **chol:** serum cholesterol in mg/dl
6. **fbs:** fasting blood sugar > 120 mg/dl
7. **restecg:** resting electrocardiographic results (values 0,1,2)
8. **thalach:** maximum heart rate achieved
9. **exang:** exercise induced angina
10. **oldpeak:** oldpeak = ST depression induced by exercise relative to rest
11. **slope:** the slope of the peak exercise ST segment
12. **ca:** number of major vessels (0-3) colored by fluoroscopy
13. **thal:** thal: 3 = normal; 6 = fixed defect; 7 = reversible defect

# Some snapshot of Dataset-

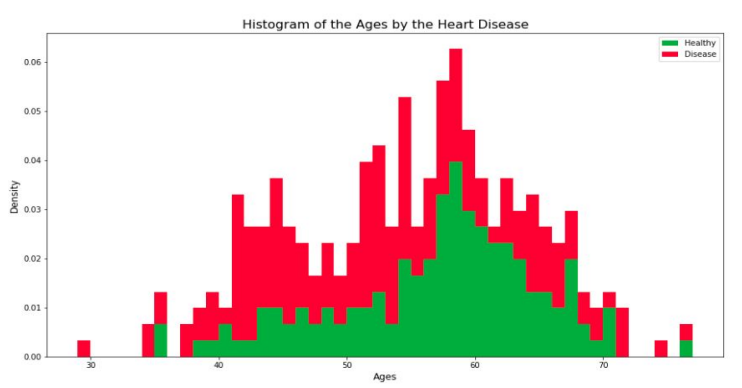
```
dataset.head(5)
```

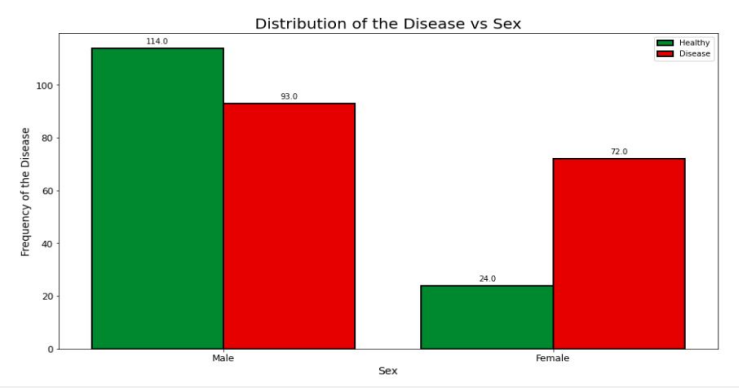
| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63 | 1 | 3 | 145 | 233 | 1 | 0 | 150 | 0 | 2.3 | 0 | 0 | 1 | 1 |
| **1** | 37 | 1 | 2 | 130 | 250 | 0 | 1 | 187 | 0 | 3.5 | 0 | 0 | 2 | 1 |
| **2** | 41 | 0 | 1 | 130 | 204 | 0 | 0 | 172 | 0 | 1.4 | 2 | 0 | 2 | 1 |
| **3** | 56 | 1 | 1 | 120 | 236 | 0 | 1 | 178 | 0 | 0.8 | 2 | 0 | 2 | 1 |
| **4** | 57 | 0 | 0 | 120 | 354 | 0 | 1 | 163 | 1 | 0.6 | 2 | 0 | 2 | 1 |

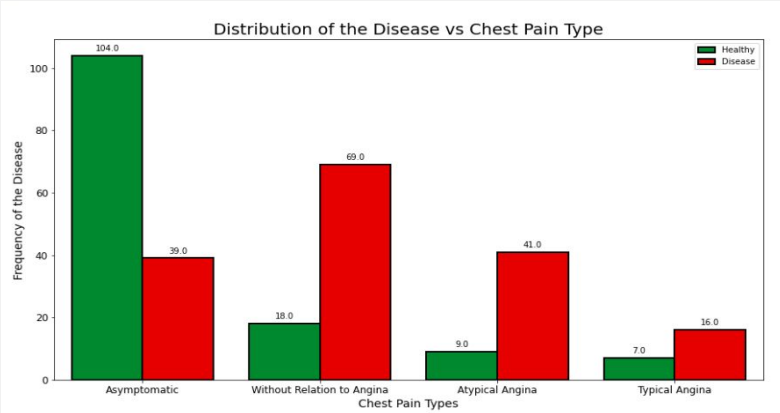
**`dataset["target"].unique()` ⇒ `array([1, 0])`**

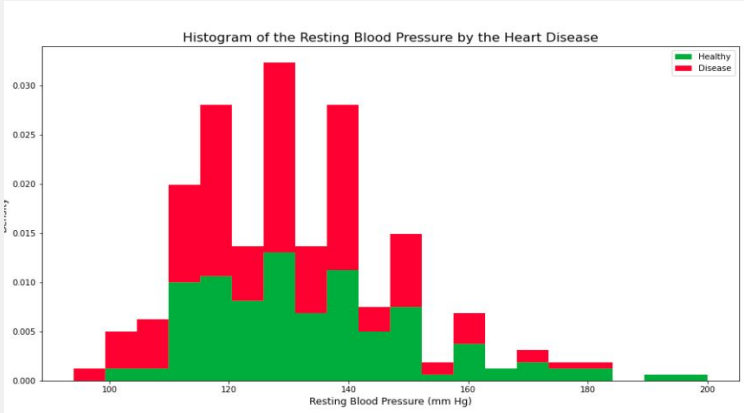
Clearly, this is a classification problem, with the target variable having values '0' and '1'

# Parametric Analysis:

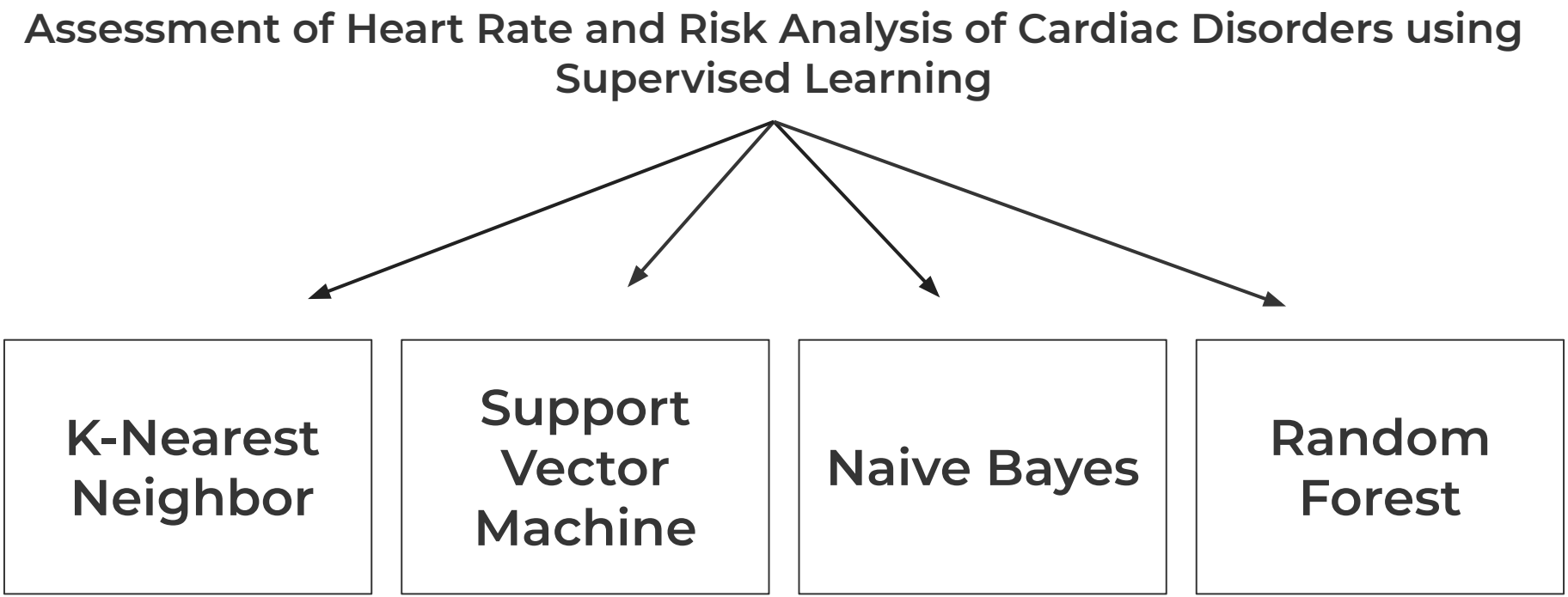


# Algorithms used to process features:



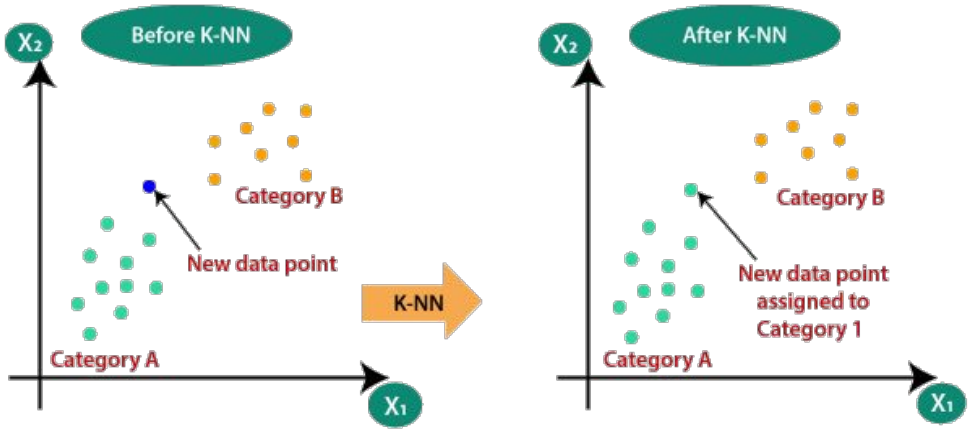
# K Nearest Neighbor (KNN)



K-Nearest Neighbour is Machine Learning algorithm based on Supervised Learning technique.

**Why to use it ?**

1. K-NN algorithm assumes the similarity between the new data and available data and put the new data point into the category that is most similar to the available categories.
2. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.



# 67.21%

Accuracy achieved by K-Nearest Neighbor (KNN) Algorithm.

## Model Fitting:

```
from sklearn.neighbors import
KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=7)

knn.fit(X_train,Y_train)

Y_pred_knn=knn.predict(X_test)
score_knn = round(accuracy_score(Y_pred_knn,Y_test)*100,2)


print("The accuracy score achieved using KNN is:
"+str(score_knn)+" %")
```

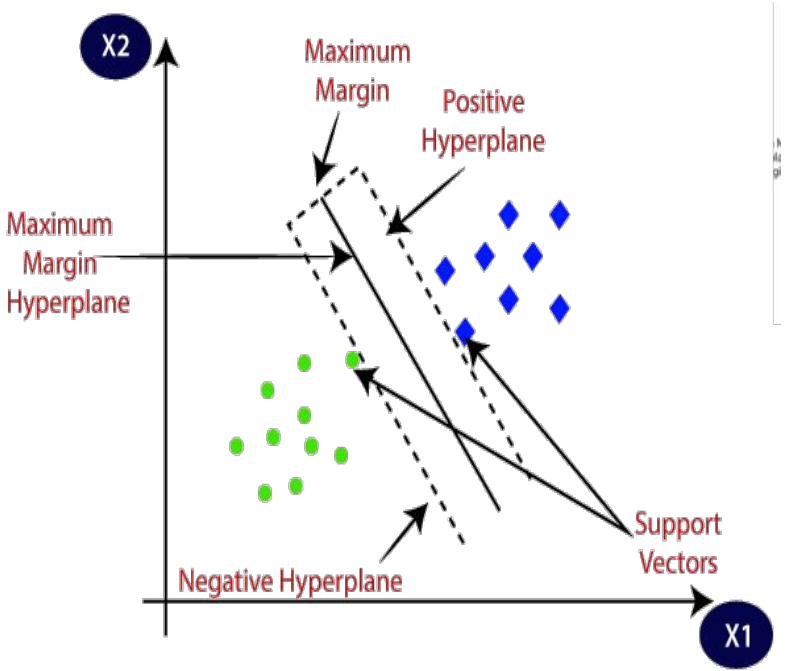
## Splitting of data:

```
from sklearn.model_selection import
train_test_split

predictors = dataset.drop("target",axis=1)
target = dataset["target"]

X_train,X_test,Y_train,Y_test =
train_test_split(predictors,target,test_si
ze=0.20,random_state=0)
```

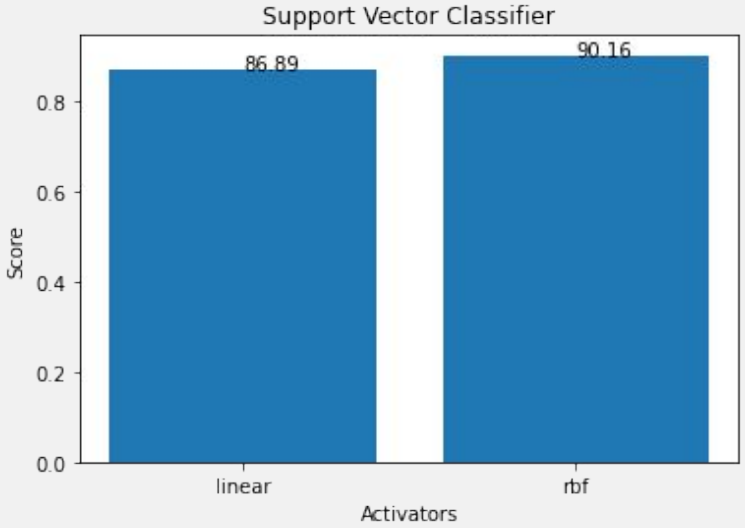
# Support Vector Machine (SVM)



- Used primarily for classification problems.
- Creates a decision boundary that can segregate n-dimensional space into classes.
- Power to reduce the *curse of dimensionality*.
- Hyperplane is chosen in a way to maximize the margin.
- The data points or vectors that are the closest to the hyperplane are known as Support Vectors.



# SVM Results




```
kernels = ['linear', 'rbf']

svc = []

for i in range(len(kernels)):

    SVclassifier = SVC(kernel = kernels[i])

    SVclassifier.fit(X_train, Y_train)

    svc.append(SVclassifier.score(X_test, Y_test))
```

**86.89%**

Accuracy achieved by Linear Function

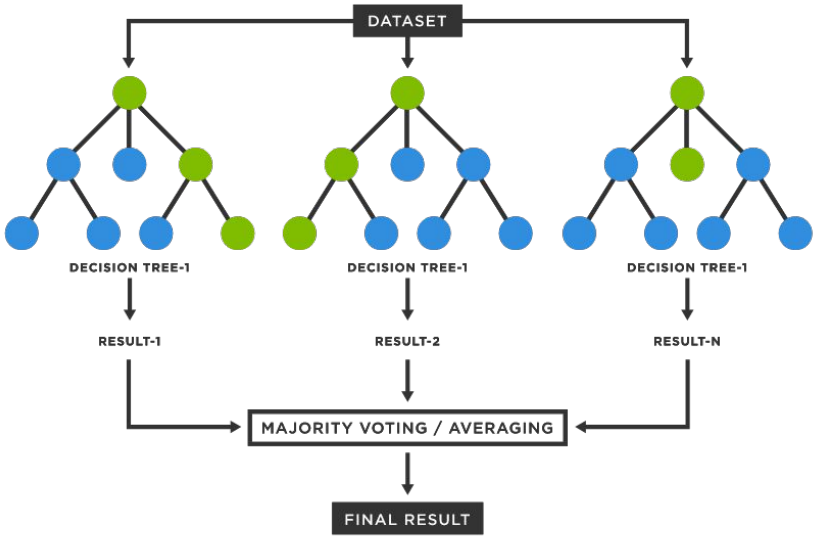
**90.16%**

Accuracy achieved by ‘rbf’ - Radial Basis Function

# Random Forest Algorithm



- In terms of accuracy, it is unrivalled among contemporary algorithms.
- On huge databases, it performs well. Handling 10,000 inputs at a time.
- As the forest grows, it generates an internal unbiased estimate of the generalisation error.
- It offers a method for guessing missing data that works well and retains accuracy even when a high percentage of the data is missing.
- It includes techniques for balancing error in uneven data sets with a class population.
- The forests that are created can be preserved and used on other data in the future.
- Unlabeled data can be used to develop unsupervised clustering, data views, and outlier identification skills.



# Code for
# Random Forest Algorithm

```
max_accuracy = 0

for x in range(2000):
    rf =
RandomForestClassifier(random_state=x)
    rf.fit(X_train,Y_train)
    Y_pred_rf = rf.predict(X_test)
    current_accuracy =
round(accuracy_score(Y_pred_rf,Y_test)*100,2)
    if(current_accuracy>max_accuracy):
        max_accuracy = current_accuracy
        best_x = x

rf =
RandomForestClassifier(random_state=best_x)
rf.fit(X_train,Y_train)
Y_pred_rf = rf.predict(X_test)
```

# 95.08%

Accuracy achieved by Random Forest.

# Naive Bayes Algorithm

- Naive Bayes is a method to predict the probability of output based on various attributes.
- This algorithm is mostly used in classification and with problems having multiple classes.

## Why to use it ? Advantages are :

- It is easy and fast to predict the class of the test data set.
- It also performs well in multi-class prediction.
- When assumption of independence holds, a Naive Bayes classifier performs better compared to other models like logistic regression and less training data is needed.

# 85.25%

Accuracy achieved by Naives Bayes Algorithm.

$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$$

## Model fitting

```
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(X_train,Y_train)
Y_pred_nb = nb.predict(X_test)
```

# Conclusion

| Algorithm | Accuracy |
| --- | --- |
| K-Nearest Neighbor | 67.21% |
| SVM | 90.16% |
| Naive Bayes | 85.25% |
| Random Forest | 95.08% |

# Thanks!

We’ll be glad to answer any questions.

We are thankful for your guidance and your understanding.