#What is HCC?

Hepatocellular carcinoma (HCC) is the most common type of primary liver cancer. Hepatocellular carcinoma occurs most often in people with chronic liver diseases, such as cirrhosis caused by hepatitis B or hepatitis C infection.

In this R project we will study the survival of patients with hepatocellular carcinoma with 3 different algorithms :

- ***Logistic regression***
- ***KNN***
- ***Naiive bayes***

Prerequisites:

You will need to install **mice library**

```
install.packages("mice")
```

What is Mice?

MICE (Multivariate Imputation via Chained Equations) Creating multiple imputations as compared to a single imputation (such as mean) takes care of uncertainty in missing values. MICE assume that the missing data are Missing at Random (MAR), which means that the probability that a value is missing depends only on observed value and can be predicted using them. It imputes data on a variable by variable basis by specifying an imputation model per variable.

Steps for our prediction project:

1. Make necessary imports, Get the features and lables from dataset .
2. Use Mice for imputation the missing values.
3. Training the data

   we loaded the class feature as a factor for prediction

   ```
   #splitting the data
   library(caret)
   set.seed(123)
   data_imputed[c(27:29)] <- NULL
   data_imputed$Class <- make.names(data_imputed$Class)
   data_imputed$Class <- as.factor(data_imputed$Class)
   ```

Logistic regression method

Using **GLMnet** for prediction:

We used trainControl Cross-validation function with 10 folds and 3 repeates with the calculation of class probabilites and a summary function that computes the required metrics.

What is GLMnet?

Glmnet is a package that fits a generalized linear model via penalized maximum

likelihood. The regularization path is computed for the lasso or elasticnet penalty at a grid of values for the regularization parameter lambda. The algorithm is extremely fast, and can exploit sparsity in the input matrix . It fits linear, logistic and multinomial, poisson, and Cox regression models. A variety of predictions can be made from the fitted models. It can also fit multi-response linear regression.

```
# cross validator
# cv >> 10-fold cross validation
# classProbs >> calculate the probability of levels in data (X0, X1)
ctrl <- trainControl(
  method = "cv",
  classProbs = TRUE,
  summaryFunction = twoClassSummary
)

#setting seed to prevent randomness
set.seed(123)

# the model training formula
glmFit <- train(
  Class ~ .,
  data = data_imputed,
  method = "glm",
  preProc = c("center", "scale"),
  trControl = ctrl,
  metric = "Sens" # to choose highest sensitivity
)
glmfit
```

Output:

```
Generalized Linear Model
131 samples
46 predictor
2 classes: 'X0', 'X1'
Pre-processing: centered (139), scaled (139)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 118, 117, 118, 119, 118, 118, ...
Resampling results:
ROC        Sens  Spec
0.4567361  0.4   0.5069444
```

k-Nearest Neighbors

K-nearest neighbors (KNN) algorithm uses ‘feature similarity’ to predict the values of new datapoints which further means that the new data point will be assigned a value based on how closely it matches the points in the training set.

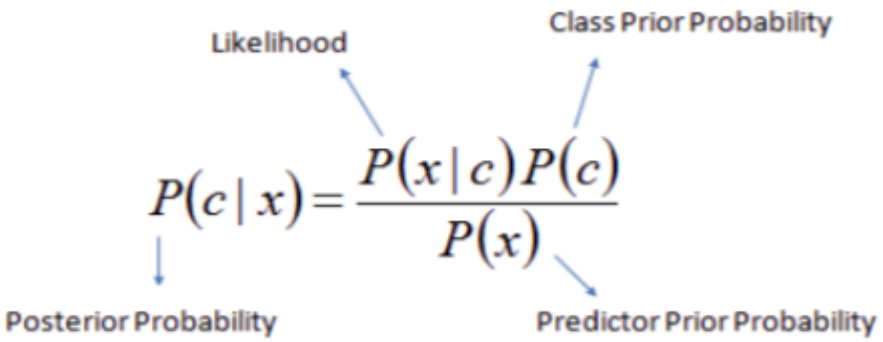

```
set.seed(123)

# Model
knnFit <- train(
  Class ~ .,
  data = data_imputed,
  method = "knn",
  preProc = c("center", "scale"),
  tuneGrid = expand.grid(k = 5),
  trControl = ctrl,
  metric = "Sens")

knnFit
```

Output:

```
k-Nearest Neighbors

164 samples
 46 predictor
  2 classes: 'X0', 'X1'

Pre-processing: centered (139), scaled (139)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 147, 148, 148, 148, 148, 148, ...
Resampling results:

  ROC        Sens       Spec
  0.6036472  0.2714286  0.8427273

Tuning parameter 'k' was held constant at a value of 5
```

Naiive Bayes method

Naive Bayes is a probabilistic machine learning algorithm that can be used in a wide variety of classification tasks.



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

```
set.seed(123)

# Model
nbFit <- train(
  Class ~ .,
  data = data_imputed,
  method = "naive_bayes",
  preProc = c("center", "scale"),
  trControl = ctrl,
  metric = "Sens",
  tuneGrid = expand.grid(laplace = 0, usekernel = FALSE, adjust = 1) )

nbFit
```

output

```
Naive Bayes

164 samples
 46 predictor
  2 classes: 'X0', 'X1'

Pre-processing: centered (139), scaled (139)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 147, 148, 148, 148, 148, 148, ...
Resampling results:

  ROC        Sens  Spec
  0.5116667  1     0.01
```