ISEN 613 - Engineering Data Analysis
Spring 2017
Final Project Report
Team: Hey Prabhu!

Analysis by Classification of Radar Signals Returned from Ionosphere

Instructor: Dr. Li Zeng

Submitted by:
Aasheet Kumar (425003025)
Abhijeet Shinde (724009781)
Dwarkanath Prabhu (625008668)
Shreyas Sasle (624009630)

Contents

Abstract:

This report analyses data from a radar system to identify whether the signal is “good” or “bad”. The dataset has 34 attributes and one response. Several classification methods are applied on the dataset to predict the response. 5-fold cross-validation is then used to generate multiple test and training datasets from the original data and test all methods. The Support Vector Machine method with radial kernel achieved the best mean accuracy of 92%.

Problem Statement:

In the study of radar signals in ionospheric research it is necessary to check whether the signals returned from radar are useful for further analysis or not. Based on the available results obtained in this dataset the main objective is to find out which of these signals can be termed as “good”.

While some signals get lost through the ionosphere, some of them contain vague and irrelevant information in the form of noise. Thus, “good” radar signals are those showing evidence of some type of structure in the ionosphere, “bad” signals are those that do not. For this distinction of the bad signals from the good ones, classification methods can be applied on the radar generated signals.

Thus, the problem then becomes to identify the classification model that most accurately predicts quality of signal based on pulses received from the ionosphere by antennas in the radar system.

The Dataset:

The dataset consists of 34 attributes. These describe the pulses received by antennas in the radar system at Goose Bay, Labrador. “There were 17 pulse numbers for the Goose Bay system. Instances in this database are described by 2 attributes per pulse number”. There are 351 observations of data, with the response being either “g” or “b” standing for “good” or “bad”.

Literature Review:

The data set (Ionosphere) used has been used by various researchers as an experimental data set to train and test their methodologies. The methods used by the researchers are conventional models covered in ISEN 613 or their extensions. The data set is mainly of a signal processing unit and hence performs best when combined with statistical approach of autocorrelation function (ACF) as proposed by Sigillito, V. G., Wing, S. P., Hutton, L. V., & Baker, K. B. (1989). Hyunsoo Kim and Se Hyun Park (2004) propose an alternative approach for data reduction in SVM to save computational time. This is applicable to very large data sets having a greater count of attributes and needs to be feature extracted. The chosen “Ionosphere” data set has comparatively fewer observations to require high computational time.

Additional combinational classification techniques are proposed by Marina Skurichina and Robert P W Duin (2000 and 2002). These techniques provide good insight into different classification methods used for analysis along with their combined effects. The former gives a statistical overview and application algorithm to study the sample size effect on classifier accuracy while latter provides an approach to boosting in LDA and corresponding statistical improvements. The "Ionosphere" data set has significant number of observations as compared to predictors and can be evaluated with or without combinational techniques.

There is abundant literature available for readers for learning exploratory data analysis followed by classification models and their improvements. However, the approach followed by this project group is in accordance to learning of ISEN 613 course and pertinent to the chosen data set.

The Proposed Approach:

The first step in working with a dataset is making sure it is clean and fit to work with. All the observations from the radar signals are not useful. Hence we first make sure there weren't any missing values and deal with any which come up.

The number of attributes is relatively high 34. i.e. ~10% of the number of observations. By conducting Principal Component Analysis (PCA), we can represent the data using the principal components that explain most of the variance in the data.

Since the data is from 16 high-frequency antennas belonging to the same system, the antenna from which the data was obtained is inconsequential. Hence, we can discard the predictors in their original form and use Principal Components for all our subsequent analysis.

After we have the appropriate number of attributes we need we continue to perform various classification methods on the dataset viz.

- K Nearest Neighbors (KNN)
- Logistic Regression
- Linear Discriminant Analysis (LDA)
- Quadratic Discriminant Analysis (QDA)
- Bagging
- Random Forest
- Support Vector Classifier (SVC)
- Support Vector Machine [SVM (polynomial and radial)].

We perform these analyses by splitting the dataset into training and test data sets. Then we perform this splitting 5 times over. Thus, we find 5 different error rates for each method. Finally, we calculate the mean of this error to arrive at the best method. In other words, we perform 5-fold cross-validation to find the best method.

Note: We are not using decision trees because we are already using bagging and random forest which are aggregates of trees.

Analysis

Libraries Used

```
library(class)
library(MASS)
library(e1071)
library(tree)
library(randomForest)
```

Code to Load Data

```
df <- read.csv("ionosphere.csv", header = FALSE)
colnames(df)[35] <- "SignalType"
```

Find missing values

```
MissingVals <- apply(df, 2, is.na)
which(MissingVals==TRUE)



xdf <- xdf[,-2]
```

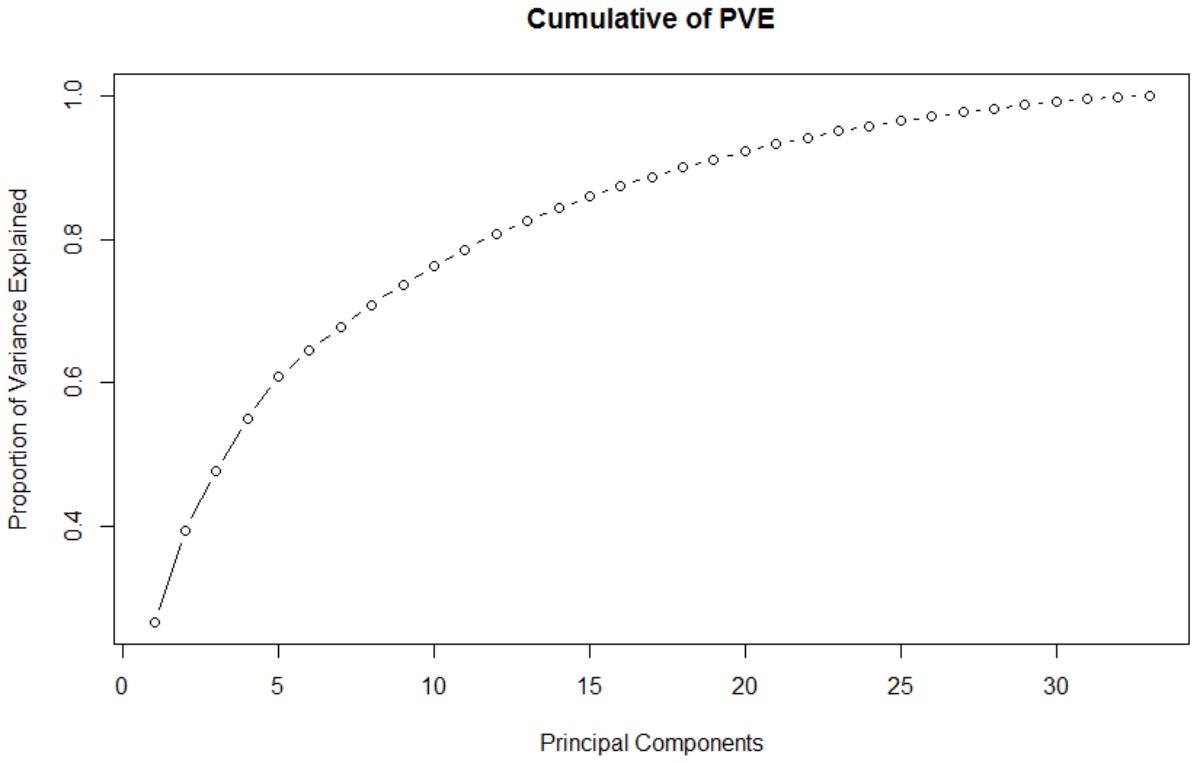
There were no missing values found in the data. However, we decided to remove the second predictor since its value was constant at 0 in all observations.

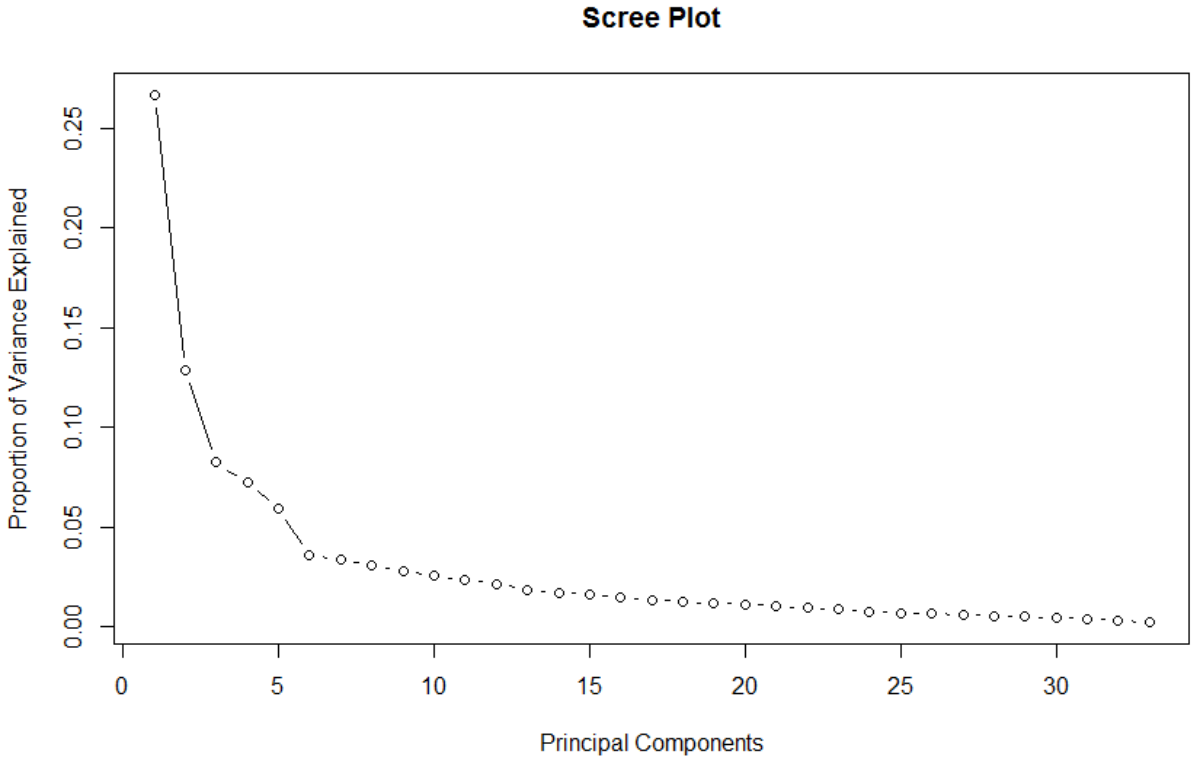
Principal Component Analysis (PCA)

```
prOut <- prcomp(xdf, scale = TRUE)


# Scree plot

prvar <- prOut$sdev^2
pve <- prvar/sum(prvar)
plot(pve, type = "b", ylab = "Proportion of Variance Explained", xlab =
"Principal Components", main = "Scree Plot")
plot(cumsum(pve), type = "b", ylab = "Proportion of Variance Explained", xlab
= "Principal Components", main = "Cumulative of PVE")
```


Cumulative of PVE
Proportion of Variance Explained
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
Principal Components


Scree Plot
Proportion of Variance Explained
0.00
0.05
0.10
0.15
0.20
0.25
0
5
10
15
20
25
30
Principal Components

By studying the scree plot we can see the proportion of variance explained. Looking at the cumulative plot, we can see that 5 principal components explain most of the variability (~60%) In the scree plot, the line starts to straighten after the 5th principal component. We can conclude that the 6th principal components onwards, explain a very small proportion and are likely not important. Hence, we decided to use 5 principal components for our analysis.

Creating Test and Training Dataset

In order to test the methods that we will employ, we split the data into training and test datasets by cross validation method explained in the later part of report. We choose the first 20% of the data as the test set and the rest as the training set.

```
z <- prOut$x[,1:5]

k = 5
testSample <- 1:as.integer(nrow(z)/k)
ztrain <- z[-testSample,]
ztest <- z[testSample,]
signalTrain <- df$SignalType[-testSample]
signalTest <- as.character(df$SignalType[testSample])
```

Classifications Techniques

We will attempt to apply several classification techniques to the data in order to predict the response i.e. the nature of signal (“good” or “bad”). We will test these methods for accuracy using cross-validation.

1.KNN

KNN is a non-parametric method for classification where the input consists of k closest data points or “neighbors” where the response is assigned to the class most common amongst its k nearest neighbors. In our case, we are using 3 nearest neighbors.

```
knnPreds <- as.character(knn(ztrain,ztest,signalTrain,k=3))
predCheck <- cbind(knnPreds,signalTest)
colnames(predCheck)[1] <- "knn"
knnAccuracy <- mean(predCheck[,"knn"] == predCheck[,"signalTest"])
```

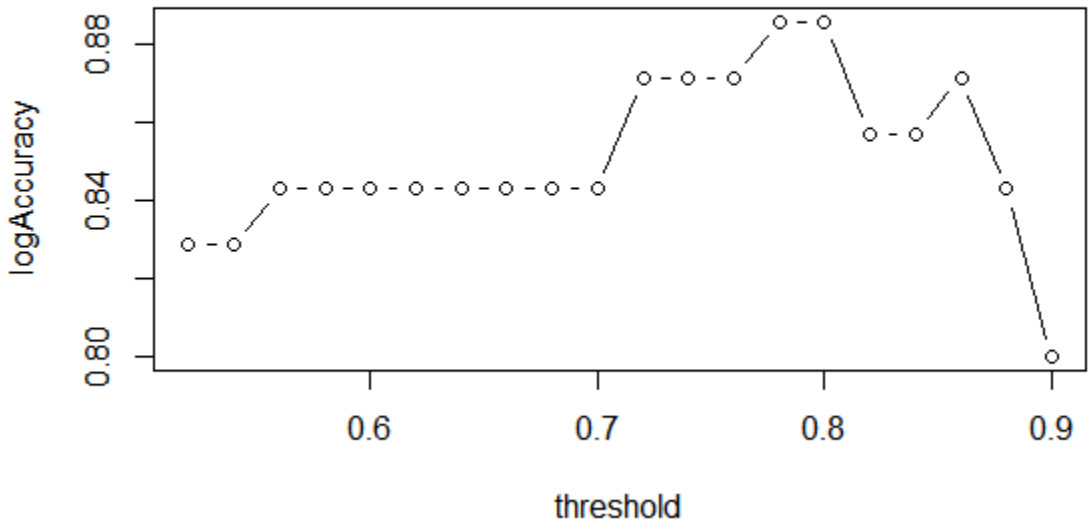
Test error: 0.1286
Accuracy: 0.8714

2.Logistic Regression

Logistic regression calculates the probability of the response belonging to one class or the other, in our case good or bad, based on a logarithmic transformation of regression on the predictors.

```
ztrain <- data.frame(ztrain,signalTrain)
ztest <- as.data.frame(ztest)
logAccuracy <- rep(0,20)
threshold <- rep(0,20)
logPredCheck <- as.character(signalTest)

for(i in 1:20){
  threshold[i] <- 0.5 +i*0.02
  logFit <- glm(signalTrain~.,data = ztrain, family = "binomial")
  logProbs <- predict(logFit,ztest,type = "response")
  logPreds <- rep("b",nrow(ztest))
  logPreds[logProbs>threshold[i]] = "g"
  logPredCheck <- cbind(logPredCheck, logPreds)
  colnames(logPredCheck)[1] <- "signalTest"
  colnames(logPredCheck)[length(colnames(logPredCheck))] <-
paste0("Logistic",i)
  logAccuracy[i] <- mean(logPredCheck[,paste0("Logistic",i)] ==
logPredCheck[,"signalTest"])
  }
plot(threshold, logAccuracy, type = 'b')
predCheck <-
cbind(as.character(logPredCheck[,paste0("Logistic",which.max(logAccuracy))]),
predCheck)
colnames(predCheck)[1] <- "Logistic"
logAccuracy <- logAccuracy[which.max(logAccuracy)]
```



Test error: 0.1142
Accuracy: 0.8858

In the above code, we looped through threshold values i.e. the value of probability above which a signal can be classified as “good”. The values of threshold ranged between 0.5

and 0.9 with increments of 0.02 to find threshold that gives the lowest test error. This value was then selected to predict the response using logistic regression.

3. Linear Discriminant Analysis (LDA)

Linear discriminant analysis (LDA) is a method used in statistics to find linear combination of features that characterizes two or more classes of objects.

```
ldaFit <- lda(signalTrain~.,data=ztrain)
ldaPreds <- predict(ldaFit,ztest)

predCheck <- cbind(as.character(ldaPreds$class),predCheck)
colnames(predCheck)[1] <- "lda"
ldaAccuracy <- mean(predCheck[,"lda"] == predCheck[,"signalTest"])
```

Test error: 0.2000
Accuracy: 0.8000

4. Quadratic Discriminant Analysis (QDA)

Quadratic discriminant analysis (QDA) is a method in statistics to separate two or more classes by a quadratic surface.

```
qdaFit <- qda(signalTrain~.,data=ztrain)
qdaPreds <- predict(qdaFit,ztest)
predCheck <- cbind(as.character(qdaPreds$class),predCheck)
colnames(predCheck)[1] <- "qda"
qdaAccuracy <- mean(predCheck[,"qda"] == predCheck[,"signalTest"])
```

Test error: 0.1142
Accuracy: 0.8858

5. Bagging

Bagging is a method of generating several decision trees by applying bootstrap on the training dataset. These trees, then aggregated, predict the class of the response with less variance. We will aggregate 100 decision trees in our analysis

```
p <- ncol(z)
set.seed(1)
bagMod <-
randomForest(signalTrain~.,data=ztrain,mtry=p,ntree=100,importance=TRUE)
bagPreds <- predict(bagMod, ztest)
predCheck <- cbind(as.character(bagPreds),predCheck)
colnames(predCheck)[1] <- "Bagging"
bagAccuracy <- mean(predCheck[,"Bagging"] == predCheck[,"signalTest"])
```

Test error: 0.0857
Accuracy: 0.9143

6. Random Forest

Random Forests is the same as bagging except that the number of predictors can be specified. In bagging all predictors are used. If number of predictors is p, the convention is to use $\sqrt{p}$ predictors. In our case, p = 5, Hence we will use 2 predictors.

```
set.seed(1)
rfMod <-
randomForest(signalTrain~.,data=ztrain,mtry=2,ntree=100,importance=TRUE)
rfPreds <- predict(rfMod, ztest)
predCheck <- cbind(as.character(rfPreds),predCheck)
colnames(predCheck)[1] <- "Random Forest"
rfAccuracy <- mean(predCheck[,"Random Forest"] == predCheck[,"signalTest"])
```

Test error: 0.0571
Accuracy: 0.9429

7. Support Vector Classifier

Support vector machining (SVM) is a discriminative classifier formally defined by a separating hyperplane i.e. using training data it outputs an optimal hyperplane which categorizes new examples.

The support vector classifier uses a linear boundary to separate classes. In this method, we will use multiple values of “cost” or a representation of the amount by which the responses from training data are allowed to violate the linear boundary. We will select the best model and apply it to the test data.

```
i <- -3:2
costs <- 10^i
gammas <- seq(0.5,5,by = 0.5)
degrees <- i[5:6]

set.seed(1)
tuneOutSVC = tune(svm,signalTrain~.,data=ztrain, kernel="linear",
ranges=list(cost=costs))
bestModSVC <- tuneOutSVC$best.model
svcPreds <- predict(bestModSVC, ztest)
predCheck <- cbind(as.character(svcPreds),predCheck)
colnames(predCheck)[1] <- "SVC"
svcAccuracy <- mean(predCheck[,"SVC"] == predCheck[,"signalTest"])
```

Test error: 0.2000
Accuracy: 0.8000

8. Support Vector Machine

The radial and polynomial support vector classifiers use a radial and polynomial kernel respectively to separate classes. In this method, we will use multiple values of “cost” or a representation of the amount by which the responses from training data are allowed to violate the

kernel boundary. We will also check for the best value of gamma for the radial kernel and degree of the polynomial kernel. We will select the best model and apply it to the test data.

```
set.seed(1)
tuneOutRadial = tune(svm,signalTrain~.,data=ztrain,
kernel="radial",ranges=list(cost=costs,gamma=gammas))
bestModRadial <- tuneOutRadial$best.model
svmRadialPreds <- predict(bestModRadial, ztest)
predCheck <- cbind(as.character(svmRadialPreds),predCheck)
colnames(predCheck)[1] <- "SVM Radial"
svmRadialAccuracy <- mean(predCheck[,"SVM Radial"] ==
predCheck[,"signalTest"])
```

Test error: 0.0428
Accuracy: 0.9572

```
set.seed(1)
tuneOutPoly = tune(svm,signalTrain~.,data=ztrain,
kernel="polynomial",ranges=list(cost=costs,degree=degrees))
bestModPoly <- tuneOutPoly$best.model
svmPolyPreds <- predict(bestModPoly, ztest)
predCheck <- cbind(as.character(svmPolyPreds),predCheck)
colnames(predCheck)[1] <- "SVM Poly"
svmPolyAccuracy <- mean(predCheck[,"SVM Poly"] == predCheck[,"signalTest"])
```

Test error: 0.2000
Accuracy: 0.8000

Cross-Validation

Cross-validation is a method of estimating error from the same data set by creating multiple test and training data sets. In our case, we will divide the dataset into 5 non-overlapping test data sets and in each case use the remaining data as the training data set. This way we will calculate 5 different error rates which we will eventually average to find the method with lowest mean error rate.

```
cvAll <- function(cvCount) {

  # Creating training and test data sets
  k = 5
  testFactor <- as.integer(nrow(z)/k)
  if(cvCount==k){
    testSample <- ((cvCount-1)*testFactor+1):nrow(z)
    } else {
  testSample <- ((cvCount-1)*testFactor+1):((cvCount)*testFactor)
  }


  # The comments below are only placeholders for actual code written above
  # for each method. The code is identical. Not reproduced to save space.

  # KNN
```

```
  # Logistic Regression
  # LDA
  # QDA
  # Random Forest
  # SVC
  # SVM

  accuracyTable <- c(svmPolyAccuracy, svmRadialAccuracy, svcAccuracy,
rfAccuracy, bagAccuracy, qdaAccuracy, ldaAccuracy, logAccuracy, knnAccuracy)
  errorTable <- 1 - accuracyTable
  names(errorTable) <- colnames(predCheck)[-length(colnames(predCheck))]

  return(errorTable)
}

cvCount <- 1:k
cvErrors <- lapply(cvCount, cvAll)
cvErrorTable <- data.frame()

for(j in 1:k){
  cvErrorsUnlisted <- unlist(cvErrors[j])
  cvErrorTable <- rbind(cvErrorTable,cvErrorsUnlisted)
}

names(cvErrorTable) <- names(unlist(cvErrors[1]))
meancverrors <- apply(cvErrorTable,2, mean)
```

Error table:

Runs	SVM poly	SVM Radial	SVC	Random Forest	Bagging	QDA	LDA	Logistic	KNN
1	0.2000	0.0428	0.2000	0.0571	0.0857	0.1142	0.2000	0.1142	0.1286
2	0.2285	0.1571	0.2428	0.1714	0.2000	0.1142	0.2285	0.2142	0.2000
3	0.1885	0.0857	0.1428	0.0857	0.1571	0.1428	0.1428	0.1285	0.1143
4	0.1285	0.0571	0.1142	0.0571	0.0857	0.0428	0.0857	0.1428	0.1000
5	0.0422	0.0281	0.0422	0.0140	0.0281	0.05633	0.0563	0.0845	0.0423
MEAN	0.1570	0.0742	0.1485	0.0771	0.1113	0.0941	0.1427	0.1369	0.1170

Result

After using various classification techniques like SVM, SVC, random forest, Bagging, LDA, QDA, KNN and log regression and running 5 different iterations to get different test errors we see that radial SVM gives us the lowest test error at 7.42% (accuracy: 92.58%), next was random forest with error rate of: 7.71% (accuracy: 92.29%).

Conclusion and Discussions

After looking at the results we can safely state that the random forest and radial SVM models are best suited for our dataset. While the methods discussed in this report are rather primitive, they provide good results with about 92% mean test accuracy for the best model while the maximum test accuracy is over 98%. The selected methods i.e. SVM and random forest can now be employed on any subsequent data along with principal component analysis to predict the nature of the signal received by the radar system

Further improvements can be by made by observing more data and training the models further. The best models from this report can also be stacked to generate more robust models. Sigillito et al suggest using more sophisticated methods such as feedforward neural network in multilayer and single layer networks to achieve around 98% test accuracy.

References

1. Sigillito, V. G., Wing, S. P., Hutton, L. V., & Baker, K. B. (1989). Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, *10*(3), 262-266
2. Kim, H., & Park, H. (2004, April). Data reduction in support vector machines by a kernelized ionic interaction model. In *Proceedings of the 2004 SIAM International Conference on Data Mining* (pp. 507-511). Society for Industrial and Applied Mathematics.
3. Skurichina, M., Kuncheva, L. I., & Duin, R. P. (2002, June). Bagging and boosting for the nearest mean classifier: Effects of sample size on diversity and accuracy. In *International Workshop on Multiple Classifier Systems* (pp. 62-71). Springer Berlin Heidelberg.
4. Skurichina, M., & Duin, R. P. (2000, June). Boosting in linear discriminant analysis. In *International Workshop on Multiple Classifier Systems* (pp. 190-199). Springer Berlin Heidelberg.
5. Image source: https://www.linkedin.com/pulse/your-data-analysis-future-khaled-bahaa