Student Enrollment Prediction

1) **Problem Definition**
2) **Prepare the data**
3) **Explore the data**
4) **Build models**
5) **Validate models**
6) **Results and Model Implementation**

The Director of Admissions of a mid-sized college wants a model to predict whether a student is accepted or not. The current process for evaluating applications is to convene a committee which reviews all the applications. The Director is looking to automate this process. You are provided with a data set with 37 columns and 39,441 applications covering the period 2006-2010.

1) Problem Definition

"To build a Predictive Model for Student Admissions Decision"

2) Prepare the data

Data preparation refers to manipulation of data into a form suitable for further analysis and processing.

It is the very first step of data mining and involves multiple activities. It improves the quality of data and consequently helps improving the quality of results. The well-known saying "garbage-in garbage-out" is very relevant here.

Data preparation involved multiple steps –

2.1 Importing Data

Below are the files to be used for this project along with its type -

File Name	Description	Format
DataDefinitions.csv	Explanation about fields in the given data	.csv
EnrollmentData.csv	Data to be used for modelling	.csv

The very first step is to import it (since the project is being done using R programming, hence images will be from the code wherever required) –

```
setwd("C:/R/")
getwd()

# Reading data
E_Data <- read.csv("./EnrollmentData.csv", na.strings = "", header = TRUE)
head(E_Data)
str(E_Data)
summary(E_Data)
```

2.2 Variable Identification -

Serial No.	Variable	Variable Category	D/I	Type (as per R)
1	Academic.Period	Continuous	I	Int
2	Unique.ID	Continuous	I	Int
3	State.Province	Categorical	I	Factor
4	Student.Population	Categorical	I	Logi
5	Application.Date	Categorical	I	Factor
6	Admissions.Population.Description	Categorical	I	Factor
7	Residency.Description	Categorical	I	Factor
8	College.Description	Categorical	I	Factor
9	Major.Description	Categorical	I	Factor
10	Gender	Categorical	I	Factor
11	High.School.GPA	Continuous	I	Num
12	Act.English	Continuous	I	Int
13	Act.Math	Continuous	I	Int
14	Act.Reading	Continuous	I	Int
15	Act.Science.Reasoning	Continuous	I	Int
16	Act.Composite	Continuous	I	Int
17	Sat.Verbal	Continuous	I	Int
18	Sat.Mathematics	Continuous	I	Int
19	Sat.Total.Score	Continuous	I	Int
20	Institutional.Aid.Offered	Continuous	I	Num
21	Class.Rank	Continuous	I	Int
22	Class.Size	Continuous	I	Int
23	Class.Rank.Percentile	Continuous	I	Num
24	Admissions.Athlete	Categorical (0/1)	I	Int
25	Need.Based.Financial.Aid	Categorical (0/1)	I	Int
26	Merit.Based.Financial.Aid	Categorical (0/1)	I	Int
27	Common.Application..Paper	Categorical (0/1)	I	Int
28	Common.Application	Categorical (0/1)	I	Int
29	College.Online.Application	Categorical (0/1)	I	Int
30	Common.Application.Upload	Categorical (0/1)	I	Int
31	College.Paper.Application	Categorical (0/1)	I	Int
32	Pre.Dental	Categorical (0/1)	I	Int
33	Pre.Law	Categorical (0/1)	I	Int
34	Pre.Med	Categorical (0/1)	I	Int
35	Pre.Veterinarian	Continuous	I	Int
36	Admitted	Categorical	D	Factor
37	Enrolled	Categorical	I	Factor

2.3 Missing Data

During data observation, it was found that there are a lot of missing data in 16 columns– State.Province, Admissions.Population.Description, Gender, High.School.GPA, Act.English, Act.Math, Act.Reading, Act.Science.Reasoning, Act.Composite, Sat.Verbal, Sat.Mathematics, Sat.Total.Score, Institutional.Aid.Offered, Class.Rank, Class.Size, Class.Rank.Percentile.

```
State.Province
PA      :13437
NJ      :11002
NY      : 4999
CT      : 1981
MD      : 1951
(Other): 5580
NA's    :  491
```

```
Admissions.Population.Description
Early Action    :21201
Regular         :15108
Passport        :  913
International :  688
Early Decision:  168
(Other)         :    3
NA's            : 1360
```

```
              Gender
Female        :21060
Male          :18325
Not Reported:    47
NA's          :     9
```

Similary NA's found in other above mentioned columns as well.

Since the missing data is huge, simply ignoring/deleting such data will lead to data loss. Hence 'kNN-Imputation' has been used to generate data.

In **KNN imputation**, the missing values of an attribute are imputed using the given number of attributes that are most similar to the attribute whose values are missing. The similarity of two attributes is determined using a distance function.

Advantages:

- k-nearest neighbour can predict both qualitative & quantitative attributes
- Correlation structure of the data is taken into consideration

Disadvantage:

- KNN algorithm is very time-consuming in analyzing large database. It searches through all the dataset looking for the most similar instances.
- Choice of k-value is very critical. Higher value of k would include attributes which are significantly different from what we need whereas lower value of k implies missing out of significant attributes.

I have calculated k using link - https://stackoverflow.com/questions/11568897/value-of-k-in-k-nearest-neighbour-algorithm and found it to be 6. I took more than 1.5 hours to impute all the missing values.

The imputed data has been saved into another file - E_data_imputed.csv so that we need not run it again and again.

```
###################################################################################

# Using kNN imputation for missing values of the columns (or for columns having NA's)

# For finding optimal value of k
# https://stackoverflow.com/questions/11568897/value-of-k-in-k-nearest-neighbour-algorithm
# k=sqrt(ncol(TrainData))
# k=6.082763

library(VIM)
E_data_imputed <- kNN(E_Data, variable = c("State.Province",
                                            "Admissions.Population.Description",
                                            "Gender",
                                            "High.School.GPA",
                                            "Act.English",
                                            "Act.Math",
                                            "Act.Reading",
                                            "Act.Science.Reasoning",
                                            "Act.Composite",
                                            "Sat.Verbal",
                                            "Sat.Mathematics",
                                            "Sat.Total.Score",
                                            "Institutional.Aid.Offered",
                                            "Class.Rank",
                                            "Class.Size",
                                            "Class.Rank.Percentile"), k = 6)

#Imputation converted Student.Population from F/T to FALSE/TRUE, making it back to original value
E_data_imputed$Student.Population <- ifelse(E_data_imputed$Student.Population==TRUE, "T", "F")

write.csv(E_data_imputed, file = "./E_data_imputed.csv")

##################################################################################
```

```
###############################################################################

# Reading from output file as above imputation took around 1.5 hours to complete
# and my laptop was about to explode due to the heat (old configuration)

setwd("C:/R/")

NewData <- read.csv("./E_data_imputed.csv", na.strings = "", header = TRUE)

# Removing extra columns created after imputation
NewData <- subset(NewData, select = Academic.Period:Enrolled)
head(NewData)
colnames(NewData)

###############################################################################
```

2.3 Creating Training and Test Data

```
###################################################################################
# As nothing was mentioned in the problem hence using the ratio of 25:75 for Test:Train data
# Creating Training and Test Data
set.seed(1234)
sub <- sample(nrow(NewData), floor(nrow(NewData)*.75))
TrainData <- NewData[sub,]
TestData <- NewData[-sub,]
nrow(TrainData)
nrow(TestData)

###################################################################################
```

2.4 Removing identifier variable as well as dependent variable – Admitted (in this case)

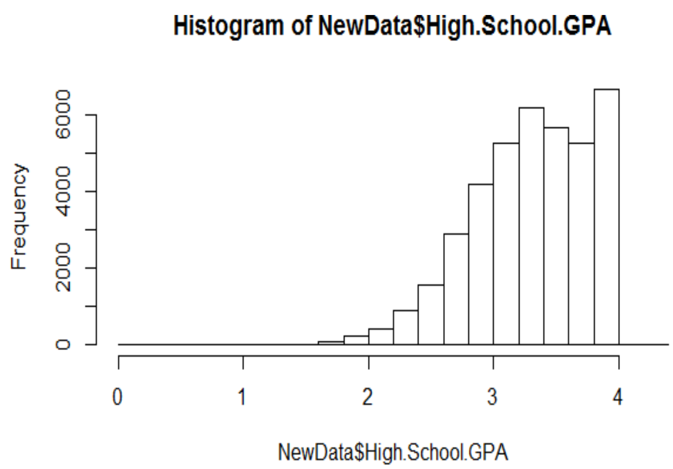
2.5 Data type conversion – In this case PCA is going to be used hence we need to convert all relevant columns into numerical data

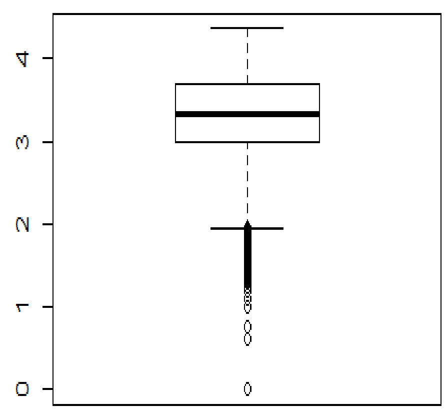
```
####################################################################################

# Removing non relevant columns (identifier variables) found in data observation
TrainData.change1 <- subset(TrainData, select = -c(Unique.ID))

# Since we are doing unsupervised learning technique, hence response variable must be removed.
TrainData.change2 <- subset(TrainData.change1, select = -c(Admitted))
colnames(TrainData.change2)

# PCA can be applied only on numerical data. Therefore, if the data has
# categorical variables they must be converted to numerical.
str(TrainData.change2)

# We have 9 variables which are either Factor or logical,
# and needs to be converted into numerical value

TrainData.change2$State.Province <- as.numeric(TrainData.change2$State.Province)
TrainData.change2$Student.Population <- as.numeric(TrainData.change2$Student.Population)
TrainData.change2$Application.Date <- as.numeric(TrainData.change2$Application.Date)

TrainData.change2$Admissions.Population.Description <- as.numeric(TrainData.change2$Admissions.Population.Description)
TrainData.change2$Residency.Description <- as.numeric(TrainData.change2$Residency.Description)
TrainData.change2$College.Description <- as.numeric(TrainData.change2$College.Description)

TrainData.change2$Major.Description <- as.numeric(TrainData.change2$Major.Description)
TrainData.change2$Gender <- as.numeric(TrainData.change2$Gender)
TrainData.change2$Enrolled <- as.numeric(TrainData.change2$Enrolled)

str(TrainData.change2) # shows all the variables have been converted into numerical value
```

3) Explore the data

3.1 Distribution

Understanding distribution for continuous variable and frequency distribution for categorical variable. Visualization methods like Histogram and box plots are being used here to understand it. E.g. High.School.GPA is highly right skewed





Similarly –

Right Skewed: High.School.GPA, Class.Rank.Percentile
Left Skewed: Institutional.Aid.Offered, Class.Rank, Class.Size

Normal: Act.English, Act.Math, Act.Reading, Act.Science.Reasoning, Act.Composite, Sat.Verbal, Sat.Mathematics, Sat.Total.Score,

3.2 Understanding the Central Tendency

Shows the mean, median, quartile values etc –

```
 High.School.GPA  Act.English      Act.Math       Act.Reading    Act.Science.Reasoning
 Min.   :0.610   Min.   : 5.0   Min.   :13.00   Min.   : 9.00   Min.   :10.00
 1st Qu.:3.000   1st Qu.:23.0   1st Qu.:22.00   1st Qu.:23.00   1st Qu.:22.00
 Median :3.330   Median :24.0   Median :24.00   Median :25.00   Median :23.00
 Mean   :3.305   Mean   :24.4   Mean   :23.83   Mean   :25.47   Mean   :23.25
 3rd Qu.:3.700   3rd Qu.:26.0   3rd Qu.:26.00   3rd Qu.:28.00   3rd Qu.:25.00
 Max.   :4.360   Max.   :36.0   Max.   :36.00   Max.   :36.00   Max.   :36.00

 Act.Composite     Sat.Verbal     Sat.Mathematics Sat.Total.Score Institutional.Aid.Offered
 Min.   :11.00   Min.   :200.0   Min.   :200.0   Min.   : 370    Min.   :  250
 1st Qu.:23.00   1st Qu.:520.0   1st Qu.:530.0   1st Qu.:1060    1st Qu.: 8500
 Median :24.00   Median :570.0   Median :580.0   Median :1150    Median :11000
 Mean   :24.39   Mean   :568.8   Mean   :577.6   Mean   :1146    Mean   :11431
 3rd Qu.:26.00   3rd Qu.:620.0   3rd Qu.:630.0   3rd Qu.:1240    3rd Qu.:14000
 Max.   :36.00   Max.   :800.0   Max.   :800.0   Max.   :1600    Max.   :61930

   Class.Rank        Class.Size      Class.Rank.Percentile
 Min.   :  0.00   Min.   :  13.0   Min.   :  2.00
 1st Qu.: 37.00   1st Qu.: 232.0   1st Qu.: 63.50
 Median : 63.00   Median : 284.0   Median : 74.50
 Mean   : 72.64   Mean   : 302.2   Mean   : 73.12
 3rd Qu.: 95.00   3rd Qu.: 353.0   3rd Qu.: 85.00
 Max.   :780.00   Max.   :2228.0   Max.   :100.00
```

3.3 Deviation

Here we try to assess the data based on standard deviation, range, variance.

Hence, we conclude that the Data needs to be normalized.

```
# Data Normalization
#install.packages("clusterSim")
library(clusterSim)

# n1 in data.normalization means standardization ((x-mean)/sd)
TrainData.norm <- data.Normalization (TrainData.change2,type="n1",normalization="column")
str(TrainData.norm)

# shows column Pre.Veterinarian containing NaN after normalization, hence will remove it.
TrainData.norm <- subset(TrainData.norm, select = -c(Pre.Veterinarian))
str(TrainData.norm)
colnames(TrainData.norm) # returns 34 variables

#######################################################
```

3.4 Dimensionality reduction

Since the number of columns are very high – 34, hence **PCA (Principal component Analysis)** is being used to reduce the number of dimensions.

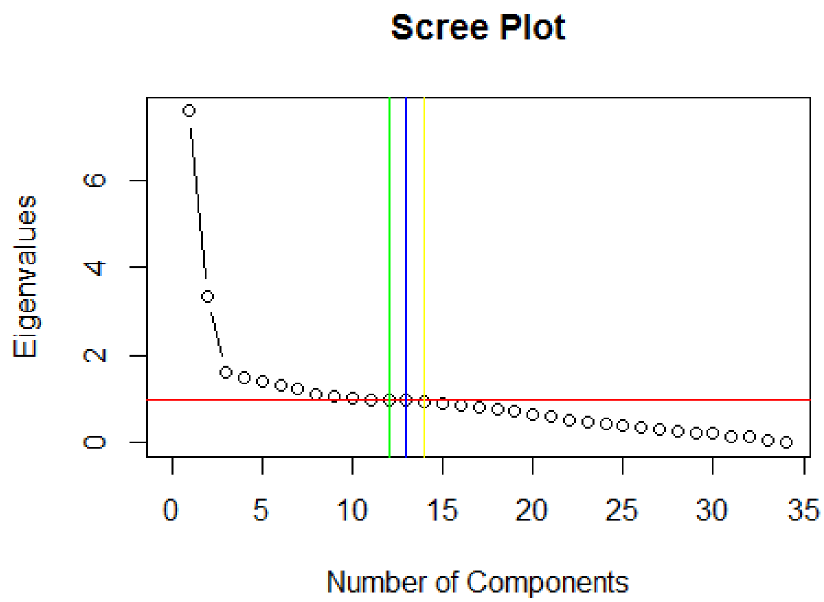
After applying PCA, we get below –

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	PC12	PC13	PC14
SS loadings	7.577	3.342	1.611	1.488	1.397	1.330	1.244	1.120	1.085	1.030	1.003	0.980	0.973	0.957
Proportion Var	0.223	0.098	0.047	0.044	0.041	0.039	0.037	0.033	0.032	0.030	0.029	0.029	0.029	0.028
Cumulative Var	0.223	0.321	0.369	0.412	0.453	0.493	0.529	0.562	0.594	0.624	0.654	0.683	0.711	0.739

(1) SS loadings represents Eigenvalues (amount of variance accounted for by each Principal Component)

- Will consider till PC13 (as they are almost near 1)

(2) As per breaks in scree plot as well as abline at 1 cutting it, shows 13 components can be retained

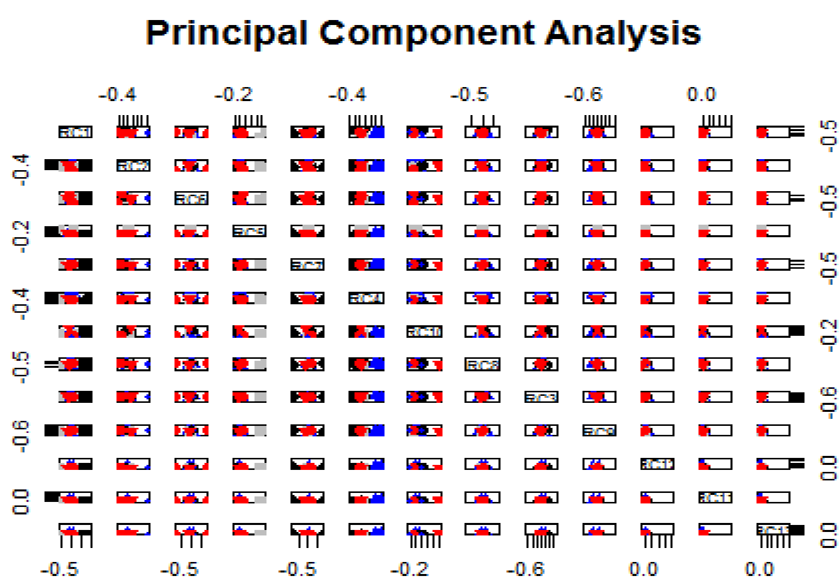


(3) number of components to retain Proportion of variance -

Retain components that account for at least x% of the total variance 5% or 10%, etc.

Retain components that combinedaccount for x% of the cumulative variance Usually at least 70% -

Again, we arrive at PC13. Hence finally decide to use 13 principal components. Below is the plot after pca rotation –



4) Build models

Decision Tree: Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.

```
# Modelling using decision tree

# install.packages("rpart")
library(rpart)
dtree.model <- rpart(Admitted ~ .,data = TrainData.new, method = "class", control = rpart.control(cp = -1
summary(dtree.model)
```

Naïve Bayes: The Naïve Bayes algorithm is a simple probabilistic classifier that calculates a set of probabilities by counting the frequency and combinations of values in a given dataset. The algorithm uses Bayes theorem and assumes all attributes to be independent given the value of class variable. Naïve Bayes classifier is based on Bayes theorem and the theorem of total probability.

In this classifier we compute the conditional probability P(Cj/X) and assign X to those class Ci having large probability i.e. Xϵ Cj if P(Cj/X)>P(Ci/X) for all i≠j=1,2,….,m.

```
# Naive-bayes

library(e1071)
classifier_NB <- naiveBayes(Admitted ~ RC1 + RC2 + RC3 + RC4 + RC5 + RC6 + RC7 + RC8 + RC9 +
                              RC10 + RC11 + RC12 + RC13, TrainData.new)

classifier_NB
```

Logistic Regression

```
# Logistic Regression

classifier_logit <- glm(Admitted ~ RC1 + RC2 + RC3 + RC4 + RC5 + RC6 + RC7 + RC8 + RC9 +
                          RC10 + RC11 + RC12 + RC13, family="binomial",data=TrainData.new)

summary(classifier_logit)
```

Neural Network

```
library(neuralnet)
classifier_NN <- neuralnet(f,data=TrainData.new1,hidden=c(5,3),linear.output=F)
summary(classifier_NN)
```

5) Validate models

Based on our analysis ‘Decision Tree’ and ‘Naïve-Bayes’ are the best models for the prediction. Please find below the results –

Decision Tree

```
> confusionMatrix(rpart.prediction, t)
Confusion Matrix and Statistics

          Reference
Prediction Admitted Denied
  Admitted     5725   1285
  Denied       1333   1518

               Accuracy : 0.7345
                 95% CI : (0.7257, 0.7432)
    No Information Rate : 0.7157
    P-Value [Acc > NIR] : 1.693e-05

                  Kappa : 0.3509
 Mcnemar's Test P-Value : 0.3583

            Sensitivity : 0.8111
            Specificity : 0.5416
         Pos Pred Value : 0.8167
         Neg Pred Value : 0.5324
             Prevalence : 0.7157
         Detection Rate : 0.5806
   Detection Prevalence : 0.7109
      Balanced Accuracy : 0.6763

       'Positive' Class : Admitted
```

Naïve Bayes

```
> confusionMatrix(t, prediction_NB)
Confusion Matrix and Statistics

          Reference
Prediction Admitted Denied
  Admitted     5312   1746
  Denied        962   1841

               Accuracy : 0.7254
                 95% CI : (0.7165, 0.7342)
    No Information Rate : 0.6362
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3776
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8467
            Specificity : 0.5132
         Pos Pred Value : 0.7526
         Neg Pred Value : 0.6568
             Prevalence : 0.6362
         Detection Rate : 0.5387
   Detection Prevalence : 0.7157
      Balanced Accuracy : 0.6800

       'Positive' Class : Admitted
```

In both the case, we get below almost same accuracy, while accuracy reduces drastically in logistic regression -

Logistic Regression

```
> confusionMatrix(t, prediction_logit_new)
Confusion Matrix and Statistics

          Reference
Prediction Admitted Denied
  Admitted      933   6125
  Denied       1314   1489

               Accuracy : 0.2456
                 95% CI : (0.2371, 0.2542)
    No Information Rate : 0.7721
    P-Value [Acc > NIR] : 1

                  Kappa : -0.2218
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.41522
            Specificity : 0.19556
         Pos Pred Value : 0.13219
         Neg Pred Value : 0.53122
             Prevalence : 0.22787
         Detection Rate : 0.09462
   Detection Prevalence : 0.71575
      Balanced Accuracy : 0.30539

       'Positive' Class : Admitted
```

Neural Network

6) Results and Model Implementation

The aim is to develop the application which will be useful in admission system. As this admission management system project includes the admission process of student, starting from when the student takes admission in college in first year till that student completes his course and collect leaving certificate from the college.

Planning to develop an application that is robust, secure and scalable enough to meet the expectations of client. Using technologies like .NET, HTML5, JQUERY, SQL SERVER, SSRS, TABLEAU etc will helps in building and deploying the application easily. Below mentioned are the USP's of application:

- Secure, scalable, and on-demand access to admission data with cloud and mobile based online student admission management system.
- Generate powerful reports with charts that provide insights on the number of prospective students applying for programs, seat allocation and utilization using student recruitment software.
- Track full status of student applications throughout the admission process, from inquiry through application, admission and enrollment.
- Access student data from anywhere and streamline evaluation process which allows you to automatically approve or reject applications for specific courses, or assign to staff.
- •Admission software allows you to manage fee payments with options for invoicing, installment plans, discounts and enable students to pay online. Automatically send fee due alerts, payment reminders & generate reports on dues.
- Manage student's enrollment and registration for courses by configuring rules and conditions. Automatically capture student data and register all types of students including new, transfer, continuing, credit and non-credit students
- Track students enrolled for courses, view grades and monitor performance through student surveys and assessments based on various metrics using with online assessment software for higher education.
- The simple, lightning fast data migration completely transforms your institution. Import large amount of academic data including student and staff profiles, documents, images, and videos that sync with web and mobile applications. Your data can be exported anytime in CSV format so that you can view and analyze the data according to your needs.
- Create and manage a large number of user accounts for students, faculty, staff and parents with the ability to create profiles that include biographic and demographic information, contact details, etc. Map users with any combination of groups and directories to provide a customized user experience.
- •Create custom forms and fields with great features, including uploads that will allow students to load images, documents and save files automatically to Dropbox, where they can share them with faculty and other users.