Problem Statement:
An education department in the US needs to analyse the factors that influence the admission of a student into a college.
Analyse the historical data and determine the key drivers.

Analysis information:

Predictive

- Run logistic model to determine the factors that influence the admission process of a student (Drop insignificant variables)
- Transform variables to factors wherever required
- Calculate accuracy of the model
- Try other modeling techniques like decision tree and SVM and select a champion model
- Determine the accuracy rates for each model
- Select the most accurate model
- Identify other Machine learning or statistical techniques that can be used

Descriptive

- Categorize the grade point average into High, Medium, and Low (with admission probability percentages) and plot it on a point chart.
- Cross grid for admission variables with GRE Categorization is shown below:

GRE	Categorized
0-440	Low
440-580	Medium
580 +	High

College Admission Prediction

Exploring the dataset:

Console Terminal × Jobs ×

~/

```
> head(col_adm)
  admit gre  gpa ses Gender_Male Race rank
1     0 380 3.61   1           0    3    3
2     1 660 3.67   2           0    2    3
3     1 800 4.00   2           0    2    1
4     1 640 3.19   1           1    2    4
5     0 520 2.93   3           1    2    4
6     1 760 3.00   2           1    1    2
> # No of records
> nrow(col_adm)
[1] 400
> # Count of missing values
> sum(is.na(col_adm))
[1] 0
> # Name of the columns
> names(col_adm)
[1] "admit"       "gre"         "gpa"         "ses"
[5] "Gender_Male" "Race"        "rank"
> summary(col_adm)
     admit             gre             gpa
 Min.   :0.0000   Min.   :220.0   Min.   :2.260
 1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130
 Median :0.0000   Median :580.0   Median :3.395
 Mean   :0.3175   Mean   :587.7   Mean   :3.390
 3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670
 Max.   :1.0000   Max.   :800.0   Max.   :4.000
      ses         Gender_Male        Race
 Min.   :1.000   Min.   :0.000   Min.   :1.000
 1st Qu.:1.000   1st Qu.:0.000   1st Qu.:1.000
 Median :2.000   Median :0.000   Median :2.000
 Mean   :1.992   Mean   :0.475   Mean   :1.962
 3rd Qu.:3.000   3rd Qu.:1.000   3rd Qu.:3.000
 Max.   :3.000   Max.   :1.000   Max.   :3.000
      rank
 Min.   :1.000
 1st Qu.:2.000
 Median :2.000
 Mean   :2.485
 3rd Qu.:3.000
 Max.   :4.000
```

```
> # Data type of columns
> sapply(col_adm,class)
      admit         gre         gpa         ses Gender_Male 
  "integer"   "integer"   "numeric"   "integer"   "integer" 
       Race        rank 
  "integer"   "integer" 
> #Converting columns to factor type
> col_adm$admit=as.factor(col_adm$admit)
> col_adm$ses=as.factor(col_adm$ses)
> col_adm$Gender_Male=as.factor(col_adm$Gender_Male)
> col_adm$Race=as.factor(col_adm$Race)
> col_adm$rank=as.factor(col_adm$rank)
> # Checking the datatype after converting :
> sapply(col_adm,class)
      admit         gre         gpa         ses Gender_Male 
   "factor"   "integer"   "numeric"    "factor"    "factor" 
       Race        rank 
   "factor"    "factor" 
> 
```

Outler Detection and Removal :

```
> #outlier detection
> #for gre
> 
> iqr1=IQR(col_adm$gre)
> iqr1
[1] 140
> quantile(col_adm$gre,na.rm = T)
  0%  25%  50%  75% 100% 
 220  520  580  660  800 
> max1=660+1.5*iqr1
> max1
[1] 870
> min1=520-1.5*iqr1
> min1
[1] 310
> 
> # All the points above the upperInner fence
> print(which(col_adm$gre>max1))  #   no outlier
integer(0)
> print(which(col_adm$gre<min1))  # 4 outliers
[1]  72 180 305 316
> # for gpa variable
> iqr2=IQR(col_adm$gpa)
> iqr2
[1] 0.54
> quantile(col_adm$gpa,na.rm = T)
   0%    25%    50%    75%   100% 
2.260 3.130 3.395 3.670 4.000 
> max2=3.670+1.5*iqr2
> min2=3.130-1.5*iqr2
> 
> # All the points above the upperInner fence
> print(which(col_adm$gpa>max2))  ## No outlier
integer(0)
> 
> #all the points below the lowerInner fence
> print(which(col_adm$gpa<min2))  # 1 outlier
[1] 290
> 
```

```
> # Removal of outlier
>
> col_adm=col_adm[-c(72,180,305,316,290),]
> nrow(col_adm)   # Outlier removed
[1] 395
>
```

Data Splitting:

```
> # Splitting the data into train and test
>
> set.seed(0)
> library('caTools')
> col_adm[,c(2,3)]=scale(col_adm[,c(2,3)])
> split=sample.split(col_adm$admit,SplitRatio = 0.75)
> train=subset(col_adm,split==T)
> test=subset(col_adm,split==F)
>
```

Logistic Regression:

```
> # Logistic Regression
>
> logit1=glm(admit~.,train,family ='binomial')
> summary(logit1)

Call:
glm(formula = admit ~ ., family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8458  -0.8294  -0.5794   0.9459   2.1850

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.03714    0.45482   2.280  0.02259 *
gre           0.27452    0.15262   1.799  0.07206 .
gpa           0.51793    0.16125   3.212  0.00132 **
ses2         -0.38459    0.33468  -1.149  0.25050
ses3         -0.40768    0.34356  -1.187  0.23538
Gender_Male1 -0.09887    0.27541  -0.359  0.71959
Race2        -0.34389    0.33981  -1.012  0.31153
Race3        -0.43592    0.33303  -1.309  0.19054
rank2        -1.29613    0.40854  -3.173  0.00151 **
rank3        -1.70399    0.43413  -3.925 8.67e-05 ***
rank4        -2.05159    0.51218  -4.006 6.19e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 370.01  on 295  degrees of freedom
Residual deviance: 319.50  on 285  degrees of freedom
AIC: 341.5

Number of Fisher Scoring iterations: 4

>
```

Second Logistic model by removing the insignificant variable:

```
~/
> #So here gre,ses,gender_male and race variable are not significant.
> # So building new model with gpa and rank variable.
>
> logit2=glm(admit~gpa+rank,train,family = 'binomial')
> summary(logit2)

Call:
glm(formula = admit ~ gpa + rank, family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8203  -0.8527  -0.5997   1.0019   2.2480

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.4765     0.3386   1.407  0.15931
gpa           0.6037     0.1479   4.081 4.49e-05 ***
rank2        -1.2261     0.3978  -3.082  0.00205 **
rank3        -1.7363     0.4248  -4.087 4.37e-05 ***
rank4        -2.0552     0.5038  -4.079 4.52e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 370.01  on 295  degrees of freedom
Residual deviance: 326.90  on 291  degrees of freedom
AIC: 336.9

Number of Fisher Scoring iterations: 4

>
```

Accuracy of the logistic model:

```
> #Accuracy of Logistic Model
>
> predicted_val1=predict(logit1,test,type='response')
>
> test$pred_admit1=ifelse(predicted_val1>0.5,1,0)
>
> #creating the confusion matrix
>
> conf_mat1=table(predicted=test$pred_admit1,actual=test$admit)
> conf_mat1
         actual
predicted  0  1
        0 55 26
        1 12  6
>
> # Getting the accuracy
>
> accuracy1=sum(diag(conf_mat1))/sum(conf_mat1)
>
> ## Accuracy is 0.6161616
>
```

He

SVM MODEL:

```
> #SVM Model
>
> library('e1071')
>
> svm_clf=svm(admit~.,train,type='C-classification',kernel='linear')
> summary(svm_clf)

Call:
svm(formula = admit ~ ., data = train, type = "C-classification", kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  202

 ( 94 108 )


Number of Classes:  2

Levels:
 0 1
```

Accuracy of SVM model:

```
> #accuracy of SVM model
>
> predicted_val2=predict(svm_clf,test[-1])
>
> # Creating the confusion matrix
>
> conf_mat2=table(predicted=predicted_val2,actual=test$admit)
> conf_mat1
         actual
predicted  0  1
        0 55 26
        1 12  6
>
> accuracy2=sum(diag(conf_mat2))/sum(conf_mat2)
> accuracy2
[1] 0.6161616
>
> # Accuracy is 0.61
>
```

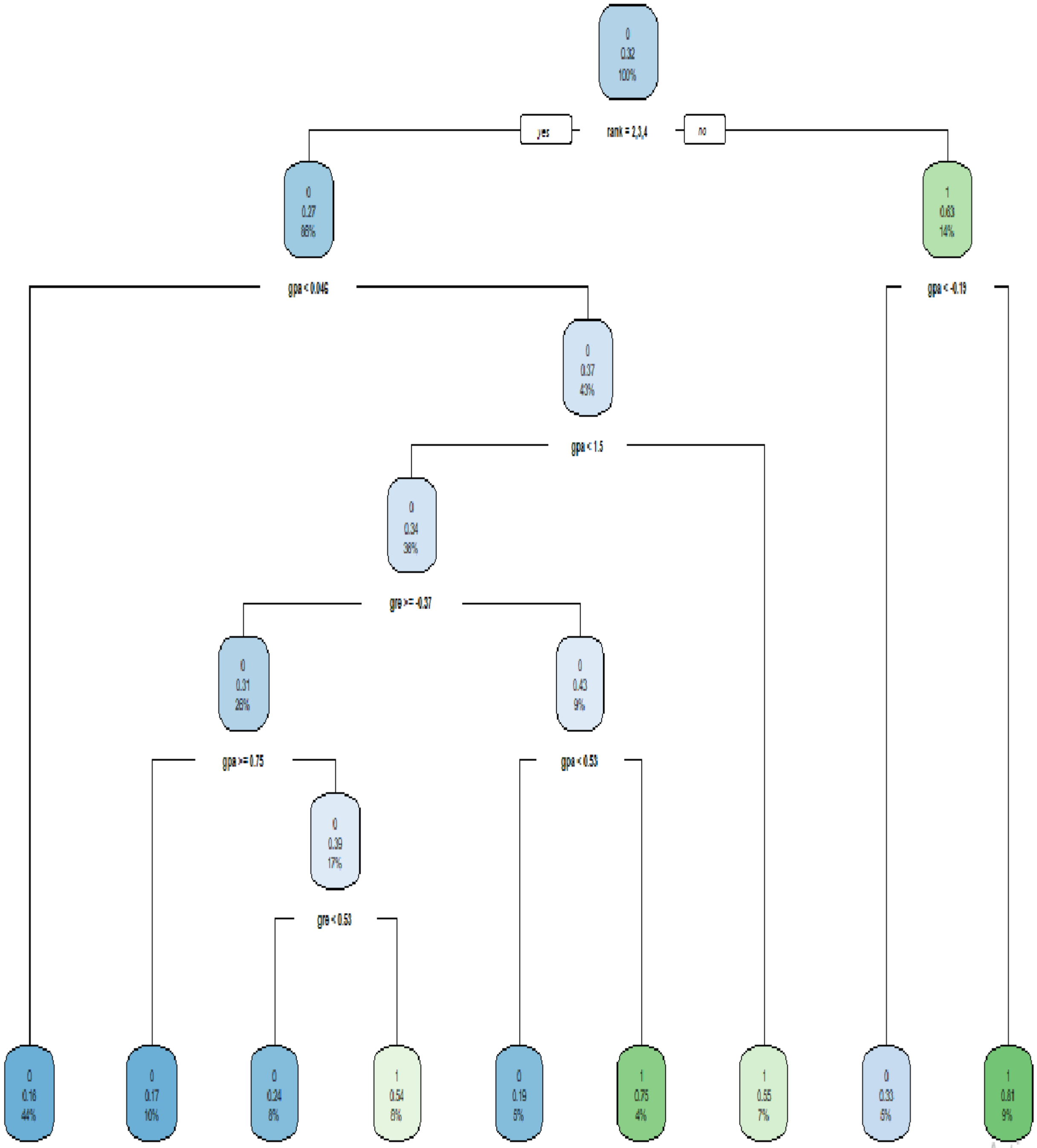
Decision tree:

```
> #Decision tree
> library('rpart')
> library('rpart.plot')
> nrow(train)
[1] 296
> nrow(test)
[1] 99
> 0.03*nrow(train)
[1] 8.88
> 0.03*nrow(test)*3
[1] 8.91
> r.cntrl=rpart.control(minsplit = 26,minbucket = 9,xval=5)
> dec_clf=rpart(admit~.,control=r.cntrl,data=train)
> summary(dec_clf)
Call:
rpart(formula = admit ~ ., data = train, control = r.cntrl)
  n= 296

          CP nsplit rel error    xerror       xstd
1 0.11702128      0 1.0000000 1.0000000 0.08520516
2 0.05319149      1 0.8829787 0.9787234 0.08471012
3 0.02127660      2 0.8297872 0.9148936 0.08309697
4 0.01063830      6 0.7446809 0.9042553 0.08280889
5 0.01000000      8 0.7234043 0.9148936 0.08309697

Variable importance
        gpa        rank         gre        Race Gender_Male
         46          28          17           4           3
        ses
          2

Node number 1: 296 observations,    complexity param=0.1170213
  predicted class=0  expected loss=0.3175676  P(node) =1
    class counts:   202    94
   probabilities: 0.682 0.318
  left son=2 (255 obs) right son=3 (41 obs)
  Primary splits:
```


0
0.32
100%
yes
rank = 2,3,4
no
0
0.27
86%
1
0.63
14%
gpa < 0.046
gpa < -0.19
0
0.37
43%
gpa < 1.5
0
0.34
36%
gre >= -0.37
0
0.31
26%
0
0.43
9%
gpa >= 0.75
gpa < 0.53
0
0.39
17%
gre < 0.53
0
0.16
44%
0
0.17
10%
0
0.24
8%
1
0.54
8%
0
0.19
5%
1
0.75
4%
1
0.55
7%
0
0.33
5%
1
0.81
9%

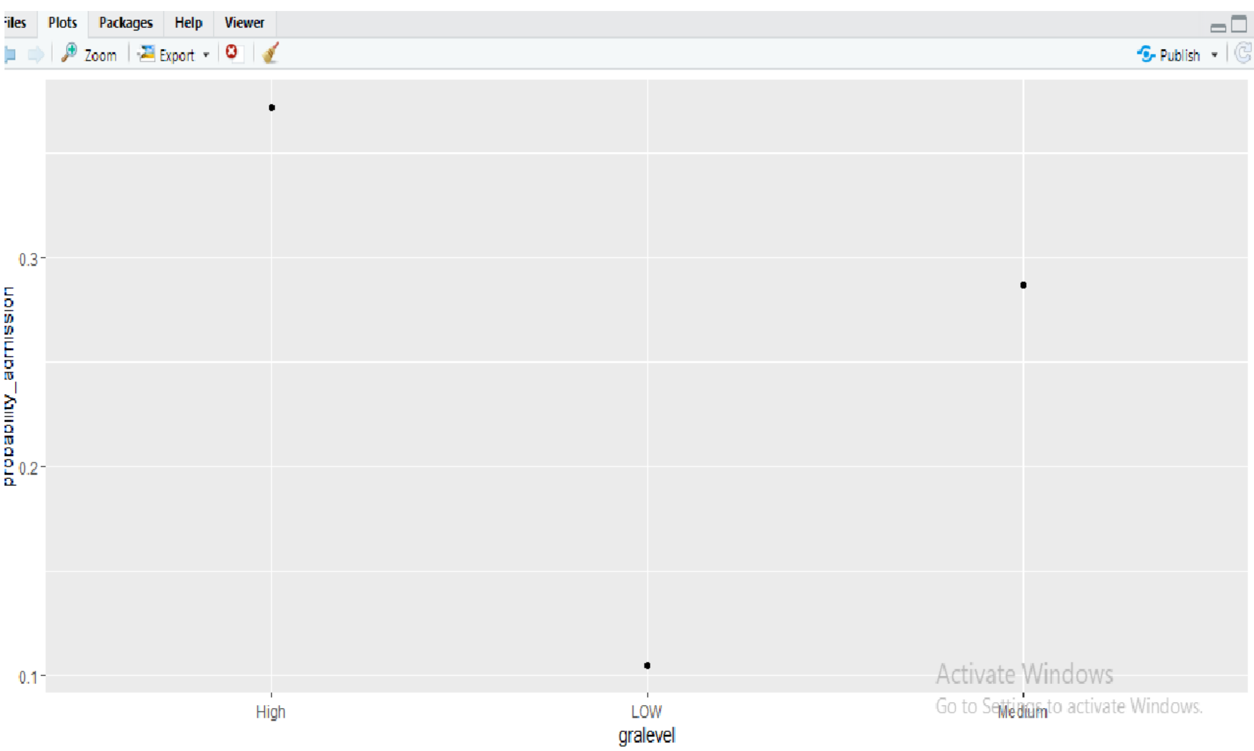
Accuracy of Decision tree:

```
~/
> #Accuracy of Decision tree
>
> predicted_val3=predict(dec_clf,test[-1],type='class')
> predicted_val3
  1  11  14  16  26  29  31  35  37  38  52  60  61  63  68  70  74  82  94  95 102 104 109 113 114 116 118
  1   1   0   0   1   0   1   0   0   0   0   0   0   0   0   1   1   0   0   1   0   1   0   0   0   1   0
121 126 127 137 138 139 144 150 151 159 161 172 176 179 192 197 198 202 203 205 206 212 214 215 216 223 233
  1   0   1   0   1   0   0   0   1   1   0   0   0   0   1   0   0   0   1   1   0   0   0   0   0   0   0
236 238 243 247 248 251 253 256 260 266 269 270 274 276 277 279 285 286 288 294 296 304 312 315 317 320 321
  0   1   0   0   0   0   1   0   0   0   1   0   0   0   1   0   0   0   0   1   0   1   1   0   0   0   0
324 325 332 339 342 358 359 369 370 373 375 382 385 386 391 394 396 400
  0   0   1   0   0   0   0   1   0   0   0   0   0   0   0   0   1   0
Levels: 0 1
>
> # Confusion matrix
>
> conf_mat3=table(predicted=predicted_val3,actual=test$admit)
> conf_mat3
         actual
predicted  0  1
        0 50 22
        1 17 10
>
> # Accuracy
>
> accuracy3=sum(diag(conf_mat3))/sum(conf_mat3)
> accuracy3
[1] 0.6060606
>
> # Accuracy is 0.6060
>
> ## It is quite evident that Regression model and SVM are the best model with
> # accuracy of 61.61%
> 
```

→ From the above 3 models it is quite evident that Regression and SVM are thebest two models with accuracy of 61%.

Categorize the grade points average into High, Medium, Low(with admissionprobability percentage) and plot it into a point chart.

```
> #Categorize the grade point average into High, Medium and Low
>
> categorize=transform(col_adm,gralevel=ifelse(col_adm$gre<440,'LOW',ifelse(col_adm$gre<580,'Medium','High')))
> View(categorize)
>
> sum_descp=aggregate(admit~gralevel,categorize,FUN=sum)
> length_descp=aggregate(admit~gralevel,categorize,FUN=length)
> length_descp
  gralevel admit
1     High   226
2      LOW    38
3   Medium   136
> probability_table=cbind(sum_descp,Recs=length_descp[,2])
> probability_table
  gralevel admit Recs
1     High    84  226
2      LOW     4   38
3   Medium    39  136
> probability_table_final=transform(probability_table,probability_admission=admit/Recs)
> probability_table_final
  gralevel admit Recs probability_admission
1     High    84  226             0.3716814
2      LOW     4   38             0.1052632
3   Medium    39  136             0.2867647
>
> #Plotting the grade point in chart
>
> library(ggplot2)
> ggplot(probability_table_final,aes(x=gralevel,y=probability_admission))+geom_point()
> |
```



	admit	gre	gpa	ses	Gender_Male	Race	rank	gralevel
1	0	380	3.61	1	0	3	3	LOW
2	1	660	3.67	2	0	2	3	High
3	1	800	4.00	2	0	2	1	High
4	1	640	3.19	1	1	2	4	High
5	0	520	2.93	3	1	2	4	Medium
6	1	760	3.00	2	1	1	2	High
7	1	560	2.98	2	1	2	1	Medium
8	0	400	3.08	2	0	2	2	LOW
9	1	540	3.39	1	1	1	3	Medium
10	0	700	3.92	1	0	2	2	High
11	0	800	4.00	1	1	1	4	High
12	0	440	3.22	3	0	2	1	Medium
13	1	760	4.00	3	1	2	1	High
14	0	700	3.08	2	0	2	2	High
15	1	700	4.00	2	1	1	1	High
16	0	480	3.44	3	0	1	3	Medium
17	0	780	3.87	2	0	3	4	High
18	0	360	2.56	3	1	3	3	LOW
19	0	800	3.75	1	1	3	2	High
20	1	540	3.81	1	0	3	1	Medium
21	0	500	3.17	3	0	2	3	Medium