We'll be using processed data from data prep practice assignment for this practice assignment. Our goal here is to build a predictive model for predicting our target Y with given characteristics. I have laid out steps to carry out operations. Few of these questions/steps will also include further details as and when required.

Split The Data : Train & Test

Split the data into two parts so that train contains 75% of the data and test contains rest 25%. Use seed `123`.

Note: We are not using validation in this exercise to save some effort. You should ideally use this to build a model which is better at generalising.

```
load("/Users/lalitsachan/Desktop/March onwards/CBAP with R/Data/temp.Rdata")
# Above code is to load data from the previous session. If you already have prepared
# data from data prep session , then you can ignore this safely
set.seed(123)
s=sample(1:nrow(d),0.75*nrow(d))
train=d[s,]
test=d[-s,]
```

Remove Multi-collinearity

Run a simple linear regression model. Pass the model object to function `vif` [which is found in package `car`]. Remove variables which have VIF values higher than 10. [Note : DO NOT USE P VALUES FROM THIS TO DROP VARIABLES. WE ARE ONLY CONCERNED WITH VIF VALUES FROM HERE. If in doubt, take to QA forum for further discussion]

Note : You might get error relating to aliased coefficient. This can happen for couple of reasons

1. Some variable has a constant value for entire data
2. Some categorical variable has been left in the data along with the dummy variables which you created from it. This leads to duplication of data
3. One or more variable columns are identical

To check which variables are having these issues, do `summary(model_object)`. Where ever you see `NA` against a variable name [instead of coefficient values]; those are the variables with issues. Check if you made some mistake in data prep while creating those variables.

```
library(car)
lm_fit=lm(Y~.,data=train)
vif(lm_fit)
```

```
##            age          fnlwgt  education.num   capital.gain   capital.loss
##       1.631170        1.050816      12.820661       1.346642      24.373268
## hours.per.week        race_AIE       race_API     race_Black     race_White
##       1.213638        2.122246       4.881026      11.534528      15.371270
##          sex_M           rel_h        rel_nif         rel_oc         rel_um
##       1.872061       37.414331       7.734787       5.630224       4.597475
##          rel_w            wc_1           wc_2           wc_3           wc_4
##       8.076045       74.803316      64.117763     331.568042     469.803502
```

```
##           wc_5         edu_1         edu_2         edu_3         edu_4
##     120.463110     22.012516      6.806545     26.437040      4.110625
##          edu_5         edu_6          ms_1          ms_2          oc_1
##       7.531234      2.854063      2.567199     39.536462      1.528858
##           oc_2          oc_3          oc_4          oc_5          nc_1
##       1.839787      1.458810      1.540103      1.325959      1.421027
##           nc_2          nc_3          nc_4      cg_flag0      cl_flag0
##       2.295891      2.932768      1.154664      1.343211     24.307575
```

```
lm_fit=lm(Y~.-wc_5,data=train)
vif(lm_fit)
```

```
##            age        fnlwgt education.num  capital.gain  capital.loss
##       1.631169      1.050800     12.819910      1.346640     24.373267
## hours.per.week      race_AIE      race_API    race_Black    race_White
##       1.213638      2.122241      4.880917     11.534476     15.371253
##          sex_M         rel_h       rel_nif        rel_oc        rel_um
##       1.871843     37.414068      7.734757      5.629942      4.597381
##          rel_w          wc_1          wc_2          wc_3          wc_4
##       8.075388      1.782199      1.555271      4.066737      4.769444
##          edu_1         edu_2         edu_3         edu_4         edu_5
##      22.012312      6.806278     26.436759      4.110568      7.531191
##          edu_6          ms_1          ms_2          oc_1          oc_2
##       2.854061      2.567199     39.536261      1.528752      1.839629
##           oc_3          oc_4          oc_5          nc_1          nc_2
##       1.458732      1.540010      1.325863      1.421024      2.295834
##           nc_3          nc_4      cg_flag0      cl_flag0
##       2.932721      1.154664      1.343181     24.307568
```

```
lm_fit=lm(Y~.-wc_5-ms_2,data=train)
vif(lm_fit)
```

```
##            age        fnlwgt education.num  capital.gain  capital.loss
##       1.625935      1.050778     12.819910      1.346550     24.372159
## hours.per.week      race_AIE      race_API    race_Black    race_White
##       1.213459      2.121943      4.880394     11.525847     15.362434
##          sex_M         rel_h       rel_nif        rel_oc        rel_um
##       1.871373     10.300539      7.259373      5.435648      4.294404
##          rel_w          wc_1          wc_2          wc_3          wc_4
##       2.801046      1.782062      1.555246      4.066725      4.769052
##          edu_1         edu_2         edu_3         edu_4         edu_5
##      22.012275      6.805793     26.436442      4.110567      7.531093
##          edu_6          ms_1          oc_1          oc_2          oc_3
##       2.854055      2.464421      1.528716      1.839626      1.458727
##           oc_4          oc_5          nc_1          nc_2          nc_3
##       1.539936      1.325777      1.421024      2.295683      2.932670
##           nc_4      cg_flag0      cl_flag0
##       1.154644      1.342641     24.306409
```

```
lm_fit=lm(Y~.-wc_5-ms_2-edu_3,data=train)
vif(lm_fit)
```

```
##            age         fnlwgt  education.num   capital.gain   capital.loss
##       1.603582       1.050726       5.220707       1.336195      24.358937
## hours.per.week       race_AIE       race_API     race_Black     race_White
##       1.208297       2.121888       4.880375      11.524530      15.361801
##          sex_M          rel_h        rel_nif         rel_oc         rel_um
##       1.869764      10.300512       7.257899       5.432785       4.293858
##          rel_w           wc_1           wc_2           wc_3           wc_4
##       2.800924       1.781412       1.554819       4.066686       4.768936
##          edu_1          edu_2          edu_4          edu_5          edu_6
##       2.138331       1.810942       1.069162       2.010604       1.615140
##           ms_1           oc_1           oc_2           oc_3           oc_4
##       2.463098       1.528602       1.788637       1.455876       1.539805
##           oc_5           nc_1           nc_2           nc_3           nc_4
##       1.325487       1.420765       2.295581       2.930167       1.154571
##       cg_flag0       cl_flag0
##       1.342524      24.297817
```

```r
lm_fit=lm(Y~.-wc_5-ms_2-edu_3-capital.loss,data=train)
vif(lm_fit)
```

```
##            age         fnlwgt  education.num   capital.gain hours.per.week
##       1.602096       1.050693       5.214076       1.336090       1.207984
##       race_AIE       race_API     race_Black     race_White          sex_M
##       2.121888       4.880362      11.524445      15.361798       1.869625
##          rel_h        rel_nif         rel_oc         rel_um          rel_w
##      10.300366       7.257878       5.432777       4.292458       2.800913
##           wc_1           wc_2           wc_3           wc_4          edu_1
##       1.780848       1.554666       4.066542       4.768889       2.137347
##          edu_2          edu_4          edu_5          edu_6           ms_1
##       1.809512       1.069056       2.009537       1.614935       2.463069
##           oc_1           oc_2           oc_3           oc_4           oc_5
##       1.528600       1.788593       1.455811       1.539640       1.325479
##           nc_1           nc_2           nc_3           nc_4       cg_flag0
##       1.420722       2.295574       2.930155       1.154568       1.342446
##       cl_flag0
##       1.024189
```

```r
lm_fit=lm(Y~.-wc_5-ms_2-edu_3-capital.loss-race_White,data=train)
vif(lm_fit)
```

```
##            age         fnlwgt  education.num   capital.gain hours.per.week
##       1.601111       1.050574       5.213932       1.335922       1.207935
##       race_AIE       race_API     race_Black          sex_M          rel_h
##       1.008841       1.258064       1.072587       1.869613      10.296480
##        rel_nif         rel_oc         rel_um          rel_w           wc_1
##       7.255667       5.428844       4.291352       2.799959       1.780809
##           wc_2           wc_3           wc_4          edu_1          edu_2
##       1.554656       4.066387       4.768889       2.137274       1.809415
##          edu_4          edu_5          edu_6           ms_1           oc_1
##       1.069027       2.009529       1.614933       2.462898       1.528282
##           oc_2           oc_3           oc_4           oc_5           nc_1
##       1.788574       1.455734       1.539511       1.325429       1.420012
##           nc_2           nc_3           nc_4       cg_flag0       cl_flag0
##       2.283028       2.877810       1.154012       1.342258       1.024148
```

```r
lm_fit=lm(Y~.-wc_5-ms_2-edu_3-capital.loss-race_White-rel_h,data=train)
vif(lm_fit)
```

```
##           age         fnlwgt  education.num   capital.gain hours.per.week 
##      1.599440       1.050499       5.208842       1.335896       1.206457 
##      race_AIE       race_API     race_Black          sex_M        rel_nif 
##      1.008554       1.256389       1.071503       1.788981       1.927182 
##        rel_oc         rel_um          rel_w           wc_1           wc_2 
##      2.283735       1.579760       1.401878       1.780633       1.554602 
##          wc_3           wc_4          edu_1          edu_2          edu_4 
##      4.066363       4.768854       2.137273       1.809346       1.068916 
##         edu_5          edu_6           ms_1           oc_1           oc_2 
##      2.009382       1.614852       2.282765       1.527246       1.788108 
##          oc_3           oc_4           oc_5           nc_1           nc_2 
##      1.455234       1.538519       1.324800       1.418070       2.278466 
##          nc_3           nc_4       cg_flag0       cl_flag0 
##      2.862668       1.154008       1.342127       1.023902
```

Build a logistic regression model

Once you are done with removing variables based on vif , build a logistic regression model. Pass it through step function to drop variables which do not meaningfully contribute to your model.Post that , save the probability score to train.

plot probability scores with outcome to see if score has been able to bring some differentiation. Your plot would be similar to this.

```r
fit=glm(Y~.-wc_5-ms_2-edu_3-capital.loss-race_White-rel_h,data=train,family = "binomial")
fit=step(fit)
```

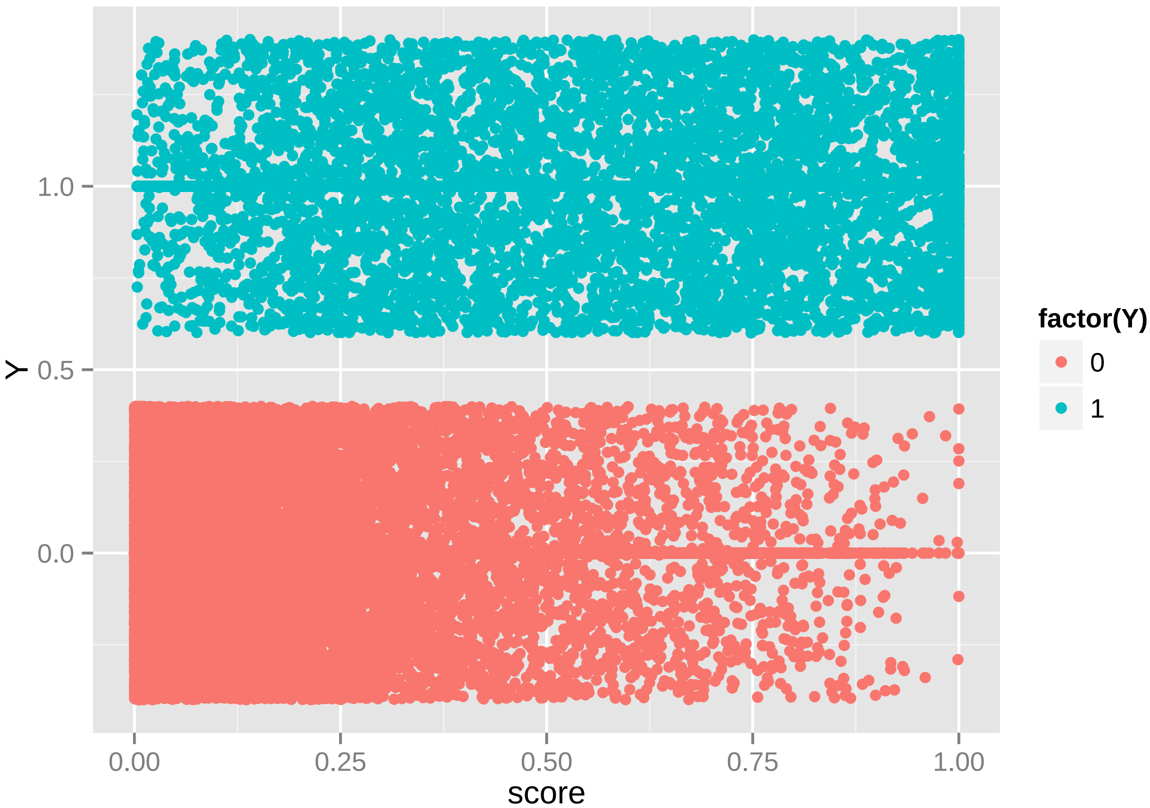
```r
summary(fit)
```

```
## 
## Call:
## glm(formula = Y ~ age + fnlwgt + education.num + capital.gain + 
##     hours.per.week + race_AIE + race_Black + sex_M + rel_nif + 
##     rel_oc + rel_um + rel_w + wc_1 + wc_2 + wc_4 + edu_2 + edu_5 + 
##     ms_1 + oc_1 + oc_2 + oc_3 + oc_4 + oc_5 + nc_1 + nc_2 + nc_3 + 
##     cg_flag0 + cl_flag0, family = "binomial", data = train)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -6.5194  -0.5240  -0.1890  -0.0317   3.3963  
## 
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)    
## (Intercept)    -1.036e+01  3.501e-01 -29.595  < 2e-16 ***
## age             2.580e-02  1.829e-03  14.104  < 2e-16 ***
## fnlwgt          7.422e-07  1.963e-07   3.781 0.000156 ***
## education.num   3.086e-01  1.230e-02  25.077  < 2e-16 ***
## capital.gain    7.343e-04  4.084e-05  17.978  < 2e-16 ***
```

```
## hours.per.week  2.886e-02  1.800e-03  16.039  < 2e-16 ***
## race_AIE       -4.223e-01  2.543e-01  -1.660 0.096843 .
## race_Black     -1.750e-01  8.596e-02  -2.036 0.041707 *
## sex_M           1.064e+00  8.866e-02  12.000  < 2e-16 ***
## rel_nif        -1.337e+00  7.381e-02 -18.111  < 2e-16 ***
## rel_oc         -2.419e+00  1.769e-01 -13.674  < 2e-16 ***
## rel_um         -1.582e+00  1.155e-01 -13.698  < 2e-16 ***
## rel_w           1.695e+00  1.153e-01  14.703  < 2e-16 ***
## wc_1            4.641e-01  1.014e-01   4.575 4.77e-06 ***
## wc_2            9.487e-01  1.110e-01   8.543  < 2e-16 ***
## wc_4            3.732e-01  4.918e-02   7.590 3.21e-14 ***
## edu_2           5.631e-01  3.130e-01   1.799 0.072029 .
## edu_5          -1.176e-01  5.785e-02  -2.033 0.042016 *
## ms_1           -7.591e-01  8.783e-02  -8.642  < 2e-16 ***
## oc_1            1.077e+00  6.782e-02  15.887  < 2e-16 ***
## oc_2            8.232e-01  7.369e-02  11.171  < 2e-16 ***
## oc_3            6.798e-01  6.477e-02  10.496  < 2e-16 ***
## oc_4            3.261e-01  6.336e-02   5.147 2.65e-07 ***
## oc_5           -5.631e-01  1.208e-01  -4.660 3.16e-06 ***
## nc_1            8.329e-01  2.384e-01   3.494 0.000476 ***
## nc_2            7.829e-01  1.772e-01   4.418 9.96e-06 ***
## nc_3            8.493e-01  1.538e-01   5.523 3.33e-08 ***
## cg_flag0        2.585e+00  2.200e-01  11.748  < 2e-16 ***
## cl_flag0       -1.114e+00  8.210e-02 -13.567  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27002  on 24419  degrees of freedom
## Residual deviance: 15545  on 24391  degrees of freedom
## AIC: 15603
##
## Number of Fisher Scoring iterations: 8
```

```
train$score=predict(fit,train,type="response")
```

```
library(ggplot2)
ggplot(train,aes(x=score,y=Y,color=factor(Y)))+geom_point()+geom_jitter()
```



Performance Metrics for A particular Cutoff

A cutoff can be decided with multiple considerations or business requirement. All of these requirements are generally based on confusion matrix. Confusion matrix is nothing but cross table of real 1/0[response] against predicted 1/0 [predicted response].

Consider an arbitrary cutoff 0.3. Get predicted response for this cutoff. Using the predicted response column, calculate Following metrics . [Consider 1= Positive , 0=Negative]

- TP (True Positive), FP (False Positive), TN (True Negative), FN(False Negative)
- Accuracy { Defined as (TP+TN)/(P+N), where P= TP+FN and N=TN+FP}
- S_n (Sensitivity) { Defined as TP/P}
- S_p (Sepecificity) {Defined as TN/N}
- Dist { Defined as $\sqrt{(1-S_n)^2+(1-S_p)^2}$}
- KS { Defined as $\frac{TP}{P}-\frac{FP}{N}$}
- M { A hypothetical metric defined as $\frac{9*FN+0.6*FP}{1.9*(P+N)}$}

```
cutoff=0.3
predicted=as.numeric(train$score>cutoff)
TP=sum(train$Y==predicted & predicted==1)
FP=sum(train$Y!=predicted & predicted==1)
TN=sum(train$Y==predicted & predicted==0)
FN=sum(train$Y!=predicted & predicted==0)

P=TP+FN
N=TN+FP
```

```
Sn=TP/P
Sp=TN/N

Dist=sqrt((1-Sn)**2+(1-Sp)**2)
KS=Sn - (FP/N)
M=(9*FN+0.6*FP)/(1.9*(P+N))
```

Getting optimal cutoff for each metric

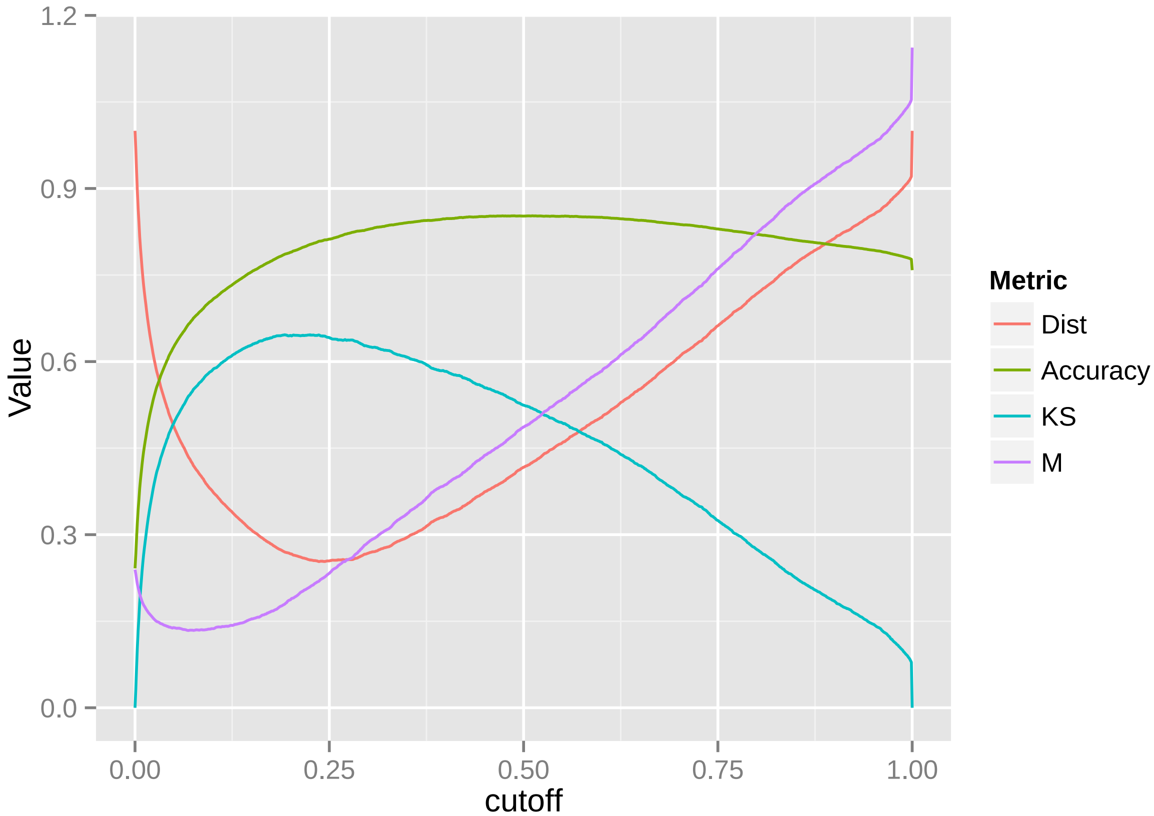
Consider 1000 cutoffs between 0 to 1 [Equally spaced , not random]. { Hint : Use function seq to generate these}. Using a for loop calculate Dist, Accuracy, KS and M for all these cutoffs. Find cutoffs for which

- Dist is minimum
- Accuracy is maximum
- KS is maximum
- M is minimum

Note: optimal cutoffs for all four condition will not be same [if they are , it will be a coincidence]

Also plot these values in a single plot to show how these metrics vary across cutoff range. Your plot will look like these

```
cutoff_data=data.frame(cutoff=99,Dist=99,Accuracy=99, KS=99,M=99)
cutoffs=seq(0,1,length=1000)
for( cutoff in cutoffs){
predicted=as.numeric(train$score>cutoff)
TP=sum(train$Y==predicted & predicted==1)
FP=sum(train$Y!=predicted & predicted==1)
TN=sum(train$Y==predicted & predicted==0)
FN=sum(train$Y!=predicted & predicted==0)

P=TP+FN
N=TN+FP

Sn=TP/P
Sp=TN/N

Dist=sqrt((1-Sn)**2+(1-Sp)**2)
Accuracy= (TP+TN)/(P+N)
KS=Sn - (FP/N)
M=(9*FN+0.6*FP)/(1.9*(P+N))

cutoff_data=rbind(cutoff_data,c(cutoff,Dist,Accuracy,KS,M))
}
cutoff_data=cutoff_data[-1,]
library(tidyr)

cutoff_data %>%
  gather(Metric,Value,Dist:M) %>%
  ggplot(aes(x=cutoff,y=Value,color=Metric))+geom_line()
```




```
cutoff_dist=cutoff_data$cutoff[which.min(cutoff_data$Dist)][1]
cutoff_Accuracy=cutoff_data$cutoff[which.max(cutoff_data$Accuracy)][1]
cutoff_KS=cutoff_data$cutoff[which.max(cutoff_data$KS)][1]
cutoff_M=cutoff_data$cutoff[which.min(cutoff_data$M)][1]
```

Performance on Test Data

Get scores for test data also . Apply respective cutoffs and evaluate the related metrics as well for each of those cutoffs.

```
test$score=predict(fit,test,type="response")

# Confusion Matrix for dist cutoff
table(test$Y,as.numeric(test$score>cutoff_dist))
```

```
##
##         0    1
##   0 4931 1268
##   1  293 1649
```

```
# Confusion Matrix for Accuracy cutoff
table(test$Y,as.numeric(test$score>cutoff_Accuracy))
```

```
##
```

```
##         0    1
##    0 5797  402
##    1  769 1173
```

```r
# Confusion Matrix for KS cutoff
table(test$Y,as.numeric(test$score>cutoff_KS))
```

```
##
##         0    1
##    0 4869 1330
##    1  259 1683
```

```r
#Confusion Matrix for M cutoff
table(test$Y,as.numeric(test$score>cutoff_M))
```

```
##
##         0    1
##    0 3539 2660
##    1   64 1878
```