GERMAN CREDIT SCORING DATASET

```
set.seed(13960406)
g_credit <- read.table(file = "http://archive.ics.uci.edu/ml/machine-learning-
databases/statlog/german/german.data")
g_credit <- as.data.frame(g_credit)

str(g_credit)
summary(g_credit)
head(g_credit)

##Since the column names are not understandable,the names are changed after looking at the data
description.
colnames(g_credit) <- c("status_chck_acc", "duration", "credit_history", "purpose",
                        "credit_amount", "saving_acctbonds", "present_employment", "installment_rate",
"statussex","other_debtors", "present_residence", "property", "age", "other_install_plans", "housing",
"no_credits",

                        "job", "no_people_maintenance", "telephone", "foreign_worker", "response")
##EDA
str(g_credit)
summary(g_credit)
head(g_credit)

library(purrr)
library(tidyr)
library(ggplot2)

g_credit %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
  facet_wrap(~ key, scales = "free") +
  geom_histogram()

g_credit$response <- g_credit$response - 1
g_credit$response <- as.factor(g_credit$response)

head(g_credit)

str(g_credit)

g_credit$status_chck_acc<- as.factor(g_credit$status_chck_acc)
g_credit$credit_history<- as.factor(g_credit$credit_history)
```

```
g_credit$purpose<- as.factor(g_credit$purpose)
g_credit$saving_acctbonds<- as.factor(g_credit$saving_acctbonds)
g_credit$present_employment<- as.factor(g_credit$present_employment)
g_credit$statussex <- as.factor(g_credit$statussex)
g_credit$other_debtors<- as.factor(g_credit$other_debtors)
g_credit$property <- as.factor(g_credit$property)
g_credit$other_install_plans <- as.factor(g_credit$other_install_plans)
g_credit$housing <- as.factor(g_credit$housing)
g_credit$job <- as.factor(g_credit$job)
g_credit$telephone <- as.factor(g_credit$telephone)
g_credit$foreign_worker <- as.factor(g_credit$foreign_worker)

sum(is.na(g_credit))
##Histograms of numeric variables
par(mfrow = c(2,4), oma = c(1,1,0,0) + 0.1, mar = c(3,3,1,1) + 0.1)
attach(g_credit)

boxplot(duration, col = "turquoise", pch = 19)
mtext("duration", cex = 0.8, side = 1, line = 2)

boxplot(credit_amount, col = "turquoise", pch = 19)
mtext("credit_amount", cex = 0.8, side = 1, line = 2 )

boxplot(installment_rate, col = "turquoise", pch = 19)
mtext("installment_rate", cex = 0.8, side = 1, line = 2)

boxplot(present_residence, col = "turquoise", pch = 19)
mtext("present_residence", cex = 0.8, side = 1, line = 2)

boxplot(age, col = "turquoise", pch = 19)
mtext("age", cex = 0.8, side = 1, line = 2)

boxplot(no_credits, col = "turquoise", pch = 19)
mtext("no_credits", cex = 0.8, side = 1, line = 2)

boxplot(no_people_maintenance, col = "turquoise", pch = 19)
mtext("no_people_maintenance", cex = 0.8, side = 1, line = 2)

par(mfrow = c(2,5))
attach(g_credit)

ggplot(g_credit,aes(status_chck_acc, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("Status_check_acc")
```

```
ggplot(g_credit,aes(credit_history, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("credit_history")

ggplot(g_credit,aes(purpose, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("purpose")

ggplot(g_credit,aes(saving_acctbonds, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("saving_acctbonds")

ggplot(g_credit,aes(present_employment, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("present_employment")

ggplot(g_credit,aes(statussex, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("statussex")

ggplot(g_credit,aes(other_debtors, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("other_debtors")

ggplot(g_credit,aes(property, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("property")

ggplot(g_credit,aes(other_install_plans, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("other_install_plans")

ggplot(g_credit,aes(housing, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("housing")

ggplot(g_credit,aes(job, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("job")

ggplot(g_credit,aes(telephone, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("telephone")

ggplot(g_credit,aes(foreign_worker, ..count..)) +
  geom_bar(aes(fill = response), position = "dodge") + xlab("foreign_worker")

index <- sample(nrow(g_credit),nrow(g_credit)*0.80)
g_credit_train = g_credit[index,]
g_credit_test = g_credit[-index,]

model_full_logit <- glm(response ~ ., family = binomial(link = logit), g_credit_train)
model_full_probit <- glm(response ~ ., family = binomial(link = probit), g_credit_train)
model_full_cloglog <- glm(response ~ ., family = binomial(link = cloglog), g_credit_train)
summary(model_full_logit)
```

```
summary(model_full_probit)
summary(model_full_cloglog)
null_model <- glm(response ~1,family=binomial,g_credit_train)

##Stepwise elimination AIC and BIC

stepaic <- step(null_model,scope=list(lower=null_model,upper=model_full_logit),direction="both")
summary(stepaic)
stepbic <-
step(null_model,scope=list(lower=null_model,upper=model_full_logit),direction="both",k=log(nrow(g_c
redit_train)))
summary(stepbic)
# ##Forward
# foraic <- step(null_model,scope=list(lower=null_model,upper=model_full),direction="forward")
# summary(foraic)
# forbic <-
step(null_model,scope=list(lower=null_model,upper=model_full),direction="forward",k=log(nrow(g_cre
dit_train)))
# summary(forbic)

#LASSO
dummy <- model.matrix(~ ., data = g_credit)
credit_data_lasso <- data.frame(dummy[,-1])

credit_train_X = as.matrix(dplyr::select(credit_data_lasso, -response1)[index,])
credit_test_X = as.matrix(dplyr::select(credit_data_lasso, -response1)[-index,])
credit_train_Y = credit_data_lasso[index, "response1"]
credit_test_Y = credit_data_lasso[-index, "response1"]

credit_lasso <- glmnet(x=credit_train_X, y=credit_train_Y, family = "binomial")
credit_lasso_cv<- cv.glmnet(x=credit_train_X, y=credit_train_Y, family = "binomial", type.measure =
"class")
plot(credit_lasso_cv)
coef(credit_lasso, s=credit_lasso_cv$lambda.min)
coef(credit_lasso, s=credit_lasso_cv$lambda.1se)
model_lasso <- glm(response ~ status_chck_acc + duration + credit_history + purpose +
saving_acctbonds + other_debtors + present_employment +     foreign_worker + other_install_plans +
installment_rate + statussex + no_people_maintenance + credit_amount + age, family = binomial,
g_credit_train)
summary(model_lasso)
library(glmnet)
g_credit_lasso <- glmnet(x=credit_train_X, y=credit_train_Y, family = "binomial")
summary(g_credit_lasso)
```

```
##Final model
final_model_g <- glm(formula = response ~ status_chck_acc + duration + credit_history +
                purpose + saving_acctbonds + other_debtors + other_install_plans +
                installment_rate + credit_amount,
              family = binomial, data = g_credit_train)
AIC(final_model_g)

summary(final_model_g)
final_model_g$deviance
#ROC
pred_train<- predict(final_model_g, type="response")
table(g_credit_train$response,(pred_train > 0.1667)*1,dnn=c("Truth","Predicted"))
library(ROCR)
pred <- prediction(pred_train,g_credit_train$response)
perf <- performance(pred, "tpr", "fpr")
plot(perf, colorize=TRUE)
unlist(slot(performance(pred, "auc"), "y.values"))

finalmodel$deviance
summary(finalmodel)

#cost function
cost1 <- function(r, pi, pcut){
  mean(((r==0)&(pi>pcut)) | ((r==1)&(pi<pcut)))
}

#Asymmetric cost
cost2 <- function(r, pi, pcut){
  weight1 <- 5
  weight0 <- 1
  c1 <- (r==1)&(pi<pcut) #logical vector - true if actual 1 but predict 0
  c0 <-(r==0)&(pi>pcut) #logical vector - true if actual 0 but predict 1
  return(mean(weight1*c1+weight0*c0))
}
pcut <- 1/(5+1)
#Symmetric cost
cost1(r = credit_train$default, pi = pred_glm0_train, pcut)

#Asymmetric cost
cost2(r = credit_train$default, pi = pred_glm0_train, pcut)

#OUT OF SAMPLE
pred_test<- predict(final_model_g, newdata = g_credit_test, type="response")
```

```
table(g_credit_test$response,(pred_test > 0.1667)*1,dnn=c("Truth","Predicted"))

pred1 <- prediction(pred_test,g_credit_test$response)
perf1 <- performance(pred1, "tpr", "fpr")
plot(perf1, colorize=TRUE)
auc <- unlist(slot(performance(pred1, "auc"), "y.values"))
auc
##Asymmetric misclassification rate
cost2 <- function(r, pi, pcut){
  weight1 <- 5
  weight0 <- 1
  c1 <- (r==1)&(pi<pcut) #logical vector - true if actual 1 but predict 0
  c0 <-(r==0)&(pi>pcut) #logical vector - true if actual 0 but predict 1
  return(mean(weight1*c1+weight0*c0))
}
pcut <- 1/(5+1)

cost2(r = g_credit_test$response,pi = pred_test,pcut)

#CROSS VALIDATION
pcut <- 1/6
costfunc  <- function(obs, pred.p){
  weight1 <- 5   # define the weight for "true=1 but pred=0" (FN)
  weight0 <- 1    # define the weight for "true=0 but pred=1" (FP)
  pcut <- 1/(1+weight1/weight0)
  c1 <- (obs==1)&(pred.p < pcut)    # count for "true=1 but pred=0"   (FN)
  c0 <- (obs==0)&(pred.p >= pcut)   # count for "true=0 but pred=1"   (FP)
  cost <- mean(weight1*c1 + weight0*c0)  # misclassification with weight
  return(cost) # you have to return to a value when you write R functions
}


library(boot)
credit_glm1<- glm(response~. , family=binomial, data=g_credit);
cv_result  <- cv.glm(data=g_credit, glmfit=credit_glm1, cost=costfunc, K=5)
cv_result$delta[2]

finalmodel2 <- glm(formula = response ~ status_chck_acc + duration + credit_history +
                     purpose + saving_acctbonds + other_debtors + other_install_plans +
                     installment_rate + credit_amount,
                   family = binomial, data = g_credit)

cv_result  <- cv.glm(data=g_credit, glmfit=finalmodel2, cost=costfunc, K=5)
cv_result$delta[2]
```

```
costfunc3  <- function(obs, pred.p){
  pred_test<- predict(finalmodel2, newdata = g_credit, type="response")

  pred1 <- prediction(pred_test,g_credit$response)
  perf1 <- performance(pred1, "tpr", "fpr")
  auc <- unlist(slot(performance(pred1, "auc"), "y.values"))
    # misclassification with weight
  return(auc) # you have to return to a value when you write R functions
}

cv_result  <- cv.glm(data=g_credit, glmfit=finalmodel2, cost=costfunc3, K=5)
cv_result$delta[2]

par(mfrow = c(1,1))

#CLassification Tree
credit_rpart0 <- rpart(formula = response ~ ., data = g_credit_train, method = "class")

credit_rpart <- rpart(formula = response ~ . , data = g_credit_train, method = "class", parms =
list(loss=matrix(c(0,5,1,0), nrow = 2)))

credit_rpart
prp(credit_rpart, extra = 1)

credit_train.pred.tree1<- predict(credit_rpart, g_credit_train, type="class")
table(g_credit_train$response, credit_train.pred.tree1, dnn=c("Truth","Predicted"))

#Out of sample
credit_test.pred.tree1<- predict(credit_rpart, g_credit_test, type="class")
table(g_credit_test$response, credit_test.pred.tree1, dnn=c("Truth","Predicted"))

#pruning
german_largetree <- rpart(formula = response ~ ., data = g_credit_train, cp = 0.001)
plotcp(german_largetree)




# cost <- function(r, phat){
#   weight1 <- 5
#   weight0 <- 1
#   pcut <- weight0/(weight1+weight0)
#   c1 <- (r==1)&(phat<pcut) #logical vector - true if actual 1 but predict 0
```

```
#   c0 <-(r==0)&(phat>pcut) #logical vector - true if actual 0 but predict 1
#   return(mean(weight1*c1+weight0*c0))
# }
#
# cost(credit_train$default, predict(credit_rpart, credit_train, type="prob"))
# #Predicted Class
# credit_test.pred.tree1<-
#   table()

#comparing logistic regression
#Fit logistic regression model
credit_glm_reg <- glm(response~.,
              data = g_credit_train,
              family=binomial)
#Get binary prediction
credit_test_pred_glm <- predict(final_model_g, g_credit_test, type="response")

#Confusion matrix
table(g_credit_test$response, as.numeric(credit_test_pred_glm>1/6), dnn=c("Truth","Predicted"))



###RANDOM SAMPLE 2
set.seed(139604060)
index <- sample(nrow(g_credit),nrow(g_credit)*0.90)
g_credit_train = g_credit[index,]
g_credit_test = g_credit[-index,]

model_full_logit <- glm(response ~ ., family = binomial(link = logit), g_credit_train)
model_full_probit <- glm(response ~ ., family = binomial(link = probit), g_credit_train)
model_full_cloglog <- glm(response ~ ., family = binomial(link = cloglog), g_credit_train)
summary(model_full_logit)
summary(model_full_probit)
summary(model_full_cloglog)
null_model <- glm(response ~1,family=binomial,g_credit_train)

# #Backward elimination AIC and BIC
# model_bAIC <- step(model_full,direction="backward")
# summary(model_bAIC)
# AIC(model_bAIC)
#
#
# model_bBIC <- step(model_full, k=log(nrow(g_credit_train)))
# summary(model_bBIC)
```

```
# AIC(model_bBIC)

##Stepwise elimination AIC and BIC

stepaic <- step(null_model,scope=list(lower=null_model,upper=model_full_logit),direction="both")
summary(stepaic)


stepbic <-
step(null_model,scope=list(lower=null_model,upper=model_full_logit),direction="both",k=log(nrow(g_c
redit_train)))
summary(stepbic)

# ##Forward
#
# foraic <- step(null_model,scope=list(lower=null_model,upper=model_full),direction="forward")
# summary(foraic)
#
# forbic <-
step(null_model,scope=list(lower=null_model,upper=model_full),direction="forward",k=log(nrow(g_cre
dit_train)))
# summary(forbic)

#LASSO
dummy <- model.matrix(~ ., data = g_credit)
credit_data_lasso <- data.frame(dummy[,-1])



credit_train_X = as.matrix(dplyr::select(credit_data_lasso, -response1)[index,])
credit_test_X = as.matrix(dplyr::select(credit_data_lasso, -response1)[-index,])
credit_train_Y = credit_data_lasso[index, "response1"]
credit_test_Y = credit_data_lasso[-index, "response1"]



credit_lasso <- glmnet(x=credit_train_X, y=credit_train_Y, family = "binomial")



credit_lasso_cv<- cv.glmnet(x=credit_train_X, y=credit_train_Y, family = "binomial", type.measure =
"class")
plot(credit_lasso_cv)
```

```
coef(credit_lasso, s=credit_lasso_cv$lambda.min)



coef(credit_lasso, s=credit_lasso_cv$lambda.1se)



model_lasso <- glm(response ~ status_chck_acc + duration + credit_history + purpose +
saving_acctbonds + other_debtors + present_employment +     foreign_worker + other_install_plans +
installment_rate + statussex + no_people_maintenance + credit_amount + age, family = binomial,
g_credit_train)
summary(model_lasso)

library(glmnet)
g_credit_lasso <- glmnet(x=credit_train_X, y=credit_train_Y, family = "binomial")
summary(g_credit_lasso)

##Final model
final_model_g1 <- glm(formula = response ~ status_chck_acc + duration + credit_history +
                        purpose + saving_acctbonds + other_debtors + other_install_plans +
                        installment_rate + credit_amount,
                      family = binomial, data = g_credit_train)
AIC(final_model_g1)

summary(final_model_g1)
#ROC
pred_train<- predict(final_model_g1, type="response")
table(g_credit_train$response,(pred_train > 0.1667)*1,dnn=c("Truth","Predicted"))
library(ROCR)
pred <- prediction(pred_train,g_credit_train$response)
perf <- performance(pred, "tpr", "fpr")
plot(perf, colorize=TRUE)
unlist(slot(performance(pred, "auc"), "y.values"))

final_model_g1$deviance



#cost function
cost1 <- function(r, pi, pcut){
  mean(((r==0)&(pi>pcut)) | ((r==1)&(pi<pcut)))
}
```

```
#Asymmetric cost
cost2 <- function(r, pi, pcut){
  weight1 <- 5
  weight0 <- 1
  c1 <- (r==1)&(pi<pcut) #logical vector - true if actual 1 but predict 0
  c0 <-(r==0)&(pi>pcut) #logical vector - true if actual 0 but predict 1
  return(mean(weight1*c1+weight0*c0))
}
pcut <- 1/(5+1)
#Symmetric cost
cost1(r = g_credit_train$default, pi = pred_glm0_train, pcut)

#Asymmetric cost
cost2(r = credit_train$default, pi = pred_glm0_train, pcut)

#OUT OF SAMPLE
pred_test<- predict(final_model_g1, newdata = g_credit_test, type="response")

table(g_credit_test$response,(pred_test > 0.1667)*1,dnn=c("Truth","Predicted"))

pred1 <- prediction(pred_test,g_credit_test$response)
perf1 <- performance(pred1, "tpr", "fpr")
plot(perf1, colorize=TRUE)
auc <- unlist(slot(performance(pred1, "auc"), "y.values"))
auc
##Asymmetric misclassification rate

cost2 <- function(r, pi, pcut){
  weight1 <- 5
  weight0 <- 1
  c1 <- (r==1)&(pi<pcut) #logical vector - true if actual 1 but predict 0
  c0 <-(r==0)&(pi>pcut) #logical vector - true if actual 0 but predict 1
  return(mean(weight1*c1+weight0*c0))
}

pcut <- 1/(5+1)

cost2(r = g_credit_test$response,pi = pred_test,pcut)

#CROSS VALIDATION
pcut <- 1/6
costfunc  <- function(obs, pred.p){
  weight1 <- 5   # define the weight for "true=1 but pred=0" (FN)
```

```
  weight0 <- 1    # define the weight for "true=0 but pred=1" (FP)
  pcut <- 1/(1+weight1/weight0)
  c1 <- (obs==1)&(pred.p < pcut)    # count for "true=1 but pred=0"   (FN)
  c0 <- (obs==0)&(pred.p >= pcut)   # count for "true=0 but pred=1"   (FP)
  cost <- mean(weight1*c1 + weight0*c0)  # misclassification with weight
  return(cost) # you have to return to a value when you write R functions
}


library(boot)
credit_glm1<- glm(response~. , family=binomial, data=g_credit);
cv_result  <- cv.glm(data=g_credit, glmfit=credit_glm1, cost=costfunc, K=5)
cv_result$delta[2]

finalmodel2 <- glm(formula = response ~ status_chck_acc + duration + credit_history +
                     purpose + saving_acctbonds + other_debtors + other_install_plans +
                     installment_rate + credit_amount,
                   family = binomial, data = g_credit)

cv_result  <- cv.glm(data=g_credit, glmfit=finalmodel2, cost=costfunc, K=5)
cv_result$delta[2]

costfunc3  <- function(obs, pred.p){
  pred_test<- predict(finalmodel2, newdata = g_credit, type="response")

  pred1 <- prediction(pred_test,g_credit$response)
  perf1 <- performance(pred1, "tpr", "fpr")
  auc <- unlist(slot(performance(pred1, "auc"), "y.values"))
  # misclassification with weight
  return(auc) # you have to return to a value when you write R functions
}

cv_result  <- cv.glm(data=g_credit, glmfit=finalmodel2, cost=costfunc3, K=5)
cv_result$delta[2]

par(mfrow = c(1,1))

#CLassification Tree
credit_rpart0 <- rpart(formula = response ~ ., data = g_credit_train, method = "class")

credit_rpart <- rpart(formula = response ~ . , data = g_credit_train, method = "class", parms =
list(loss=matrix(c(0,5,1,0), nrow = 2)))

credit_rpart
```

```
prp(credit_rpart, extra = 1)

credit_train.pred.tree1<- predict(credit_rpart, g_credit_train, type="class")
table(g_credit_train$response, credit_train.pred.tree1, dnn=c("Truth","Predicted"))

#Out of sample
credit_test.pred.tree1<- predict(credit_rpart, g_credit_test, type="class")
table(g_credit_test$response, credit_test.pred.tree1, dnn=c("Truth","Predicted"))

#pruning
german_largetree <- rpart(formula = response ~ ., data = g_credit_train, cp = 0.001)
plotcp(german_largetree)




# cost <- function(r, phat){
#   weight1 <- 5
#   weight0 <- 1
#   pcut <- weight0/(weight1+weight0)
#   c1 <- (r==1)&(phat<pcut) #logical vector - true if actual 1 but predict 0
#   c0 <-(r==0)&(phat>pcut) #logical vector - true if actual 0 but predict 1
#   return(mean(weight1*c1+weight0*c0))
# }
#
# cost(credit_train$default, predict(credit_rpart, credit_train, type="prob"))
# #Predicted Class
# credit_test.pred.tree1<-
#   table()

#comparing logistic regression
#Fit logistic regression model
credit_glm_reg <- glm(response~.,
                      data = g_credit_train,
                      family=binomial)
#Get binary prediction
credit_test_pred_glm <- predict(credit_glm_reg, g_credit_test, type="response")

#Confusion matrix
table(g_credit_test$response, as.numeric(credit_test_pred_glm>1/6), dnn=c("Truth","Predicted"))
```