MiniProject 3:

STAT 6340

Raheel Ahmed

Rsa170130

Question 1

a. **Perform an exploratory analysis of data.**
In commented portion of the code. We can see that there are 1000 subjects in the database and 20 features, with the one binary response being Default. Checking status, history, savings, and other plans were found to have a noticeable negative correlation with Default, whereas duration and amount had a strong positive correlation.

b. **Build a “reasonably good” logistic regression model for these data. There is no need to explore interactions. Carefully justify all the choices you make in building the model.**
I checked to see correlations between all the variables and default and the ones that I chose in my logistic regression model had both a high magnitude of correlation and a lower p-value in our full and partial logistic regression model

c. **Write the final model in equation form. Interpret the estimated coefficients of at least two predictors. Provide training error rate for the model.**
   a. Score = 1.60477871 - 0.58096911*checkingstatus1 - 0.37347583*history - 0.22999377 *savings + 0.17344665*installment + 0.03794233*duration - 0.26281146*otherplans

   Final Equation: 1/(1 + exp(-Score))

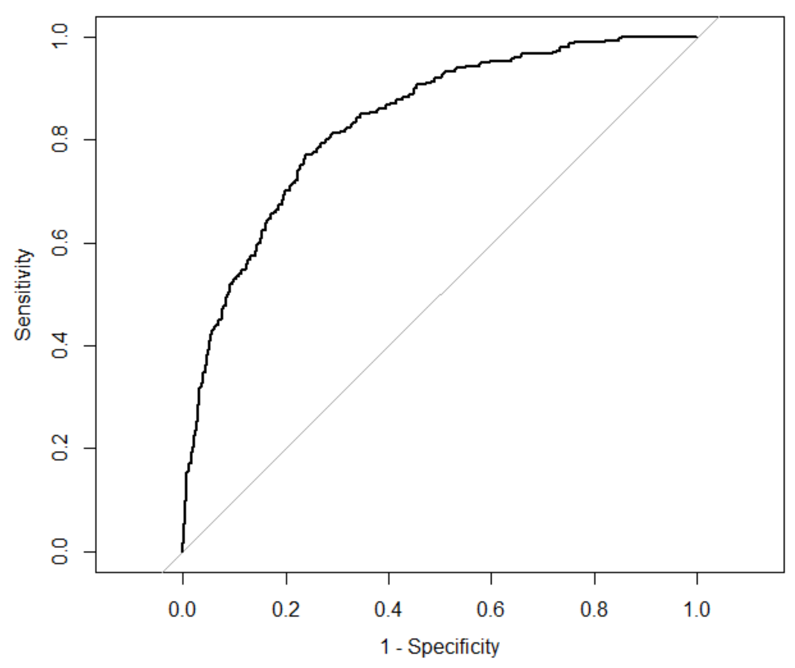
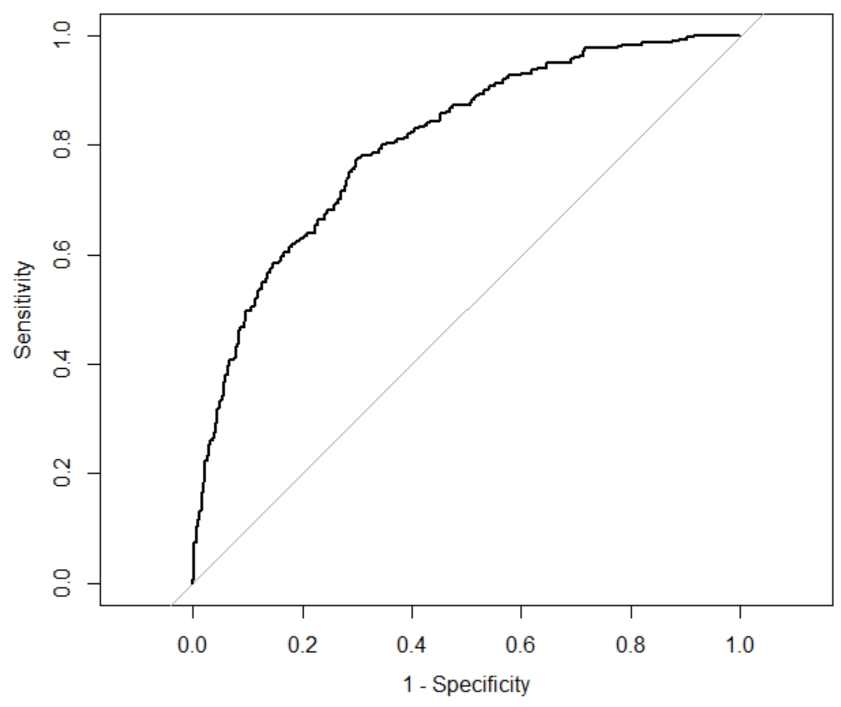
   b. For every 1 unit increase in installment the log odds of Defaulting increase by 0.173. For every 1 unit of increase in savings the log odds of Defaulting decrease by 0.22999.
   c. Training Error Rate: 0.245

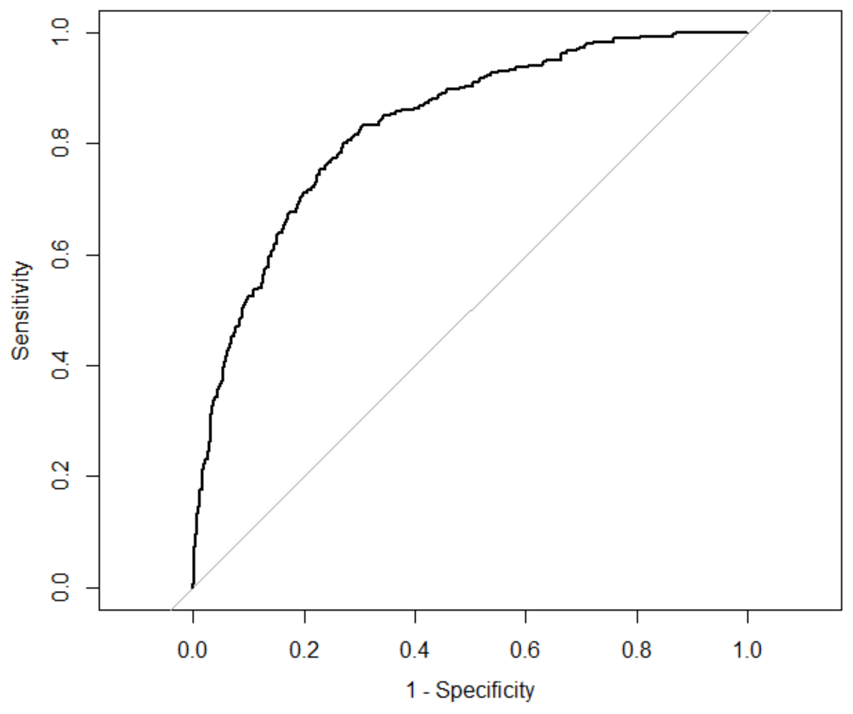
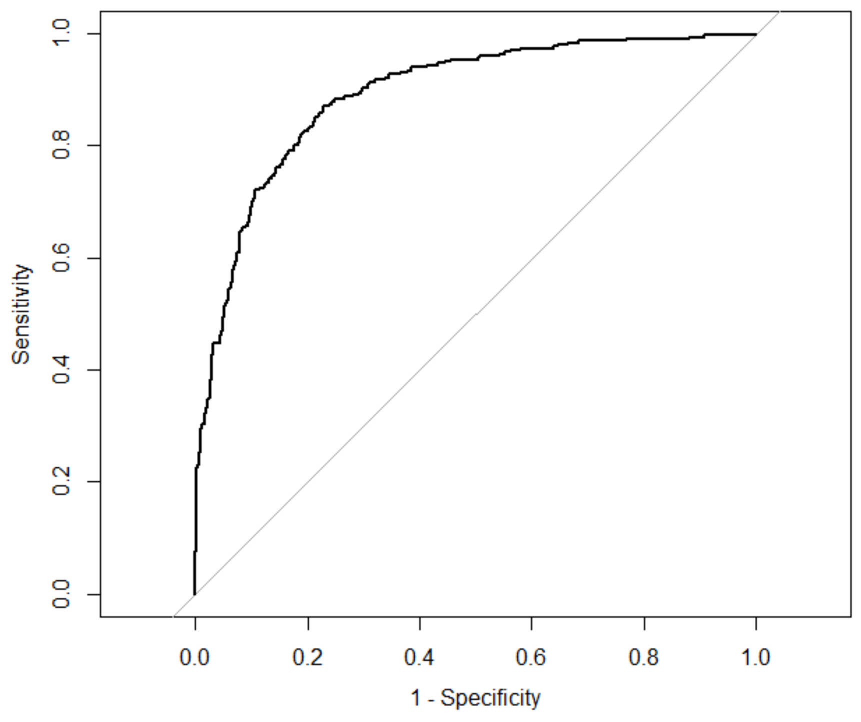
Question 2

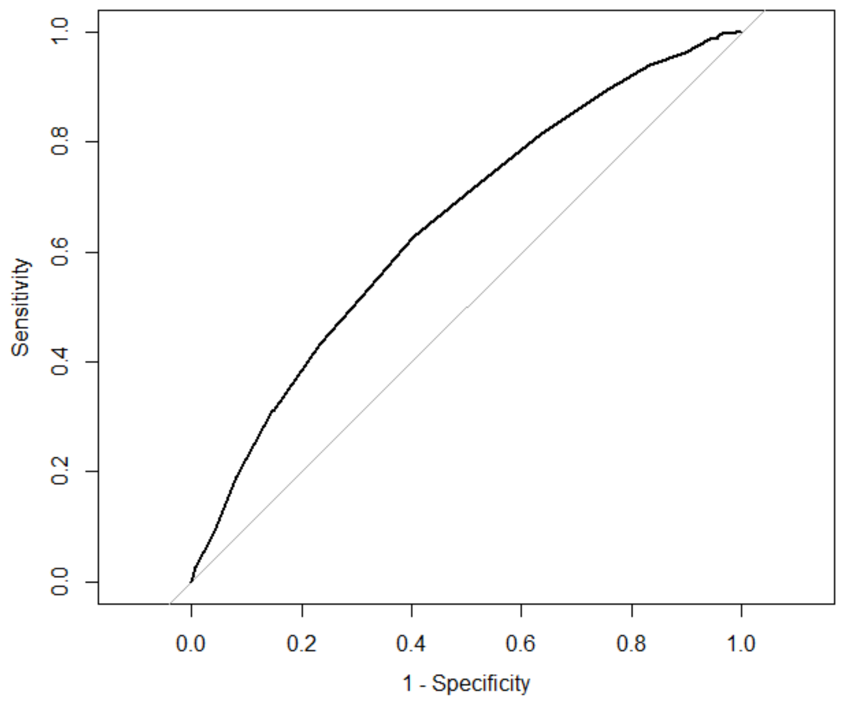
Note: Major points excluding evaluations of performance have been commented in the code below, such as implementations of LOOCV, KNN Tuning, etc.

Model Type	Error Rate	Sensitivity	Specificity	LOOCV
Logistic Regression with all predictors	0.214	0.5333333	0.8942857	0.249000 0.248446(bias adjusted)
Custom Logistic Regression Model	0.245	0.4333333	0.8928571	0.24900 0.248907(bias adjusted)
LDA Model	0.223	0.5400000	0.8785714	0.239
QDA Model	0.177	0.6966667	0.8257143	0.278
KNN	0.288	0.08333333	0.98428571	0.288

Model Type	ROC Curve	AUC

Logistic Regression with all predictors		0.8338
Custom Logistic Regression Model		0.8031

LDA Model		0.8322
QDA Model		0.8907

KNN		0.65

h. **Compare the results from various classifiers. Which classifier would you recommend? Justify your answer.**

It seems in our case that QDA seemed to perform best when considering most of our evaluation methods. In terms of the ROC curves, KNN performed the worst, with the lowest AUC. Logistic regression of both the custom model and the full model were fairly similar to LDA and QDA, but QDA did have the highest AUC.

Similarly, in terms of error rate, KNN performed least well. Both models of logistic regression were close to LDA and QDA, but QDA did have the lowest error rate. Additionally, for all these models, specificity was noticeably higher than sensitivity, most likely because these models were optimizing for accuracy, but the underlying pattern of QDA outperforming remained, although the other models excluding KNN were close in performance, even while considering these metrics of evaluation.

Code

```r
library(boot)
library(e1071)
library(pROC)
library(class)
library(MASS)

# Read the data into a dataframe variable
german_data <- read.csv("germancredit.csv", stringsAsFactors = T)
# Print number of observations in the german data and provide info on its
features / response
nrow(german_data)
str(german_data)
# Proportion of observations that are defaulting
pie(table(german_data$Default))
# Check to see which variables tend to have a strong correlation with Default
cor(german_data)[,1]

# Produce a logistic regression model using all the predictors
all_lr <- glm(Default ~ ., data = german_data, family = "binomial")
summary(all_lr)

# Produce a logistic regression model that uses predictors that appeared to
have a very low p-value in our previous model
partial_lr <- glm(Default ~ checkingstatus1 + history + savings + installment
+ duration + otherplans, data = german_data, family = "binomial")
summary(partial_lr)
coef(partial_lr)

# Dropped variables are still significant, but model still seems to be fair
in its interpretation
anova(partial_lr, all_lr, test = "Chisq")

# Calculate probabilities and predictions for our partial logistic regression
model
partial_lr.prob <- predict(partial_lr, type="response")
partial_lr.pred <- ifelse(partial_lr.prob >= 0.5, 1, 0)
# Error rate of partial logistic regression model
1 - mean(partial_lr.pred == german_data$Default)
# Confusion matrix and sensitivity, specificity provided
table(partial_lr.pred, german_data$Default)
c(130/(130+170), (625/(625+75)))
# Create roc variable, print to obtain AUC and plot to show curve
roc.part_lr <- roc(german_data$Default, partial_lr.prob)
roc.part_lr
plot(roc.part_lr, legacy.axes=T)
# Calculate LOOCV estimate of test error for the partial log-reg model
cost <- function(r, pi = 0){mean(abs(r - pi) > 0.5)}
part_lr.err <- cv.glm(german_data, partial_lr, cost, K=nrow(german_data))
part_lr.err$delta


# Calculate probabilities and predictions for our full logistic regression
model
all_lr.prob <- predict(all_lr, type="response")
all_lr.pred <- ifelse(all_lr.prob >= 0.5, 1, 0)
# Error rate of full logistic regression model
1 - mean(all_lr.pred == german_data$Default)
# Confusion matrix and sensitivity, specificity provided
table(all_lr.pred, german_data$Default)
c(160/(160+140), (626/(626+74)))
# Create roc variable, print to obtain AUC and plot to show curve
roc.all_lr <- roc(german_data$Default, all_lr.prob)
roc.all_lr
plot(roc.all_lr, legacy.axes=T)

#2b. Our own estimate of LOOCV estimation for full logistic regression model
sum_misclass <- 0
for(i in 1:nrow(german_data)){
  # Fit to training excluding one subject, predict the subject, then find
error
  excluded_subject <- german_data[i,]
  remaining_train <- german_data[-i,]
  log_fit <- glm(Default ~ ., data = remaining_train, family="binomial")
  prob <- predict(log_fit, excluded_subject, type="response")
  if(prob >= 0.5){
    pred = T
  }
  else{
    pred = F
  }
  err <- 1 - (pred == excluded_subject$Default)
  # Sum error
  sum_misclass <- sum_misclass + err
}
# Find average of summed error to obtain LOOCV estimate
(sum_misclass / nrow(german_data))

#2c. Use package to estimate LOOCV for full log-reg model
cost <- function(r, pi = 0){mean(abs(r - pi) > 0.5)}
all_lr.err <- cv.glm(german_data, all_lr, cost, K=nrow(german_data))
all_lr.err$delta

############################################################################
###
############################################################################
###
# Fit an LDA model with all predictors
lda.fit <- lda(Default ~ ., data=german_data)
lda.pred <- predict(lda.fit, german_data)
# Find error rate for LDA model
1 - mean(lda.pred$class == german_data$Default)
# Print confusion matrix and sensitivity, specificity
table(lda.pred$class, german_data$Default)
c(162/(162+138), 615/(615+85))
# Calculate roc variable for LDA then print it to find AUC, then plot to show
curve
roc.lda <- roc(german_data$Default, lda.pred$posterior[,1])
roc.lda
plot(roc.lda, legacy.axes=T)

# Calculate LOOCV for LDA using same principles in 2b.
```

```r
lda_misclass <- 0
for(i in 1:nrow(german_data)){
  l_fit <- lda(Default ~ ., data=german_data[-i, ])
  l_pred <- predict(l_fit, german_data[i,])
  err <- 1 - (l_pred$class == german_data[i,]$Default)
  lda_misclass <- lda_misclass + err
}
(lda_misclass/nrow(german_data))


############################################################################
###
############################################################################
###
# Fit a QDA model on all predictors in the german data
qda.fit <- qda(Default ~ ., data = german_data)
qda.pred <- predict(qda.fit, german_data)
# Calculate error rate for QDA model on training data
1 - mean(qda.pred$class == german_data$Default)
# Print confusion matrix and sensitivity, specificity
table(qda.pred$class, german_data$Default)
c(209/(209+91), (578/(578+122)))
# Make roc variable for QDA model and print to show AUC, then plot to show
curve
roc.qda <- roc(german_data$Default, qda.pred$posterior[,1])
roc.qda
plot(roc.qda, legacy.axes=T)

# Find LOOCV for QDA model using same principles as in 2b.
qda_misclass <- 0
for(i in 1:nrow(german_data)){
  q_fit <- qda(Default ~ ., data=german_data[-i, ])
  q_pred <- predict(q_fit, german_data[i,])
  err <- 1 - (q_pred$class == german_data[i,]$Default)
  qda_misclass <- qda_misclass + err
}
(qda_misclass/nrow(german_data))

############################################################################
##
############################################################################
##
# Turn factors into numerical data for each qualitative feature
german_data[,c(2,4,5,7,8,10,11,13,15,16,18,20,21)] <-
  sapply(german_data[,c(2,4,5,7,8,10,11,13,15,16,18,20,21)], as.numeric)
# Tune knn with cross validation for n folds
knn.cross <- tune.knn(x = german_data[,-1],
                      y = as.factor(german_data[,1]),
                      k = 1:200,
                      tunecontrol=tune.control(sampling = "cross"),
                      cross=nrow(german_data))
# Print to find optimal K
summary(knn.cross)
optim_K <- 38
# Make a KNN classifier using the optimal K
classifier_knn <- knn(train = german_data[,-1],
                      test = german_data[,-1],
                      cl = german_data$Default,
                      k = optim_K, prob=T)
# Store probabilities for ROC curve
knn_probs <- attr(classifier_knn, "prob")
# Print error rate for KNN
1 - mean(classifier_knn == german_data$Default)
# Print confusion matrix and sensitivity, specificity for KNN model
table(classifier_knn, german_data$Default)
c(25/(25+275), (689/(689+11)))
# Produce ROC curve for KNN model of german dataset
roc.knn <- roc(german_data$Default, knn_probs)
# Print variable to find AUC then plot the curve
roc.knn
plot(roc.knn, legacy.axes=T)
```