IST 387 HW 10

```
# Enter your name here: Ezra Cohen
```

Attribution statement: (choose only one and delete the rest)

```
# 1. I did this homework by myself, with help from the book and the professor.
```

Support vector machines (SVM) are a highly flexible and powerful method of doing **supervised machine learning**. Supervised learning means that there is a **criterion one is trying to predict**. The typical strategy is to **divide data** into a **training set** and a **test set** (for example, **two-thirds training** and **one-third test**), train the model on the training set, and then see how well the model does on the test set.

In this homework, we will use another banking dataset to train an SVM model to **classify potential borrowers into 2 groups of credit risk** – **reliable borrowers** and **borrowers posing a risk**. You can learn more about the variables in the dataset here: https://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29

This kind of classification algorithms is used in many aspects of our lives – from credit card approvals to stock market predictions, and even some medical diagnoses.

Part 1: Load and condition the data

A. Copy the contents of the following .csv file into a dataframe called **credit**:

https://ist387.s3.us-east-2.amazonaws.com/data/GermanCredit.csv

You will also need to install( ) and library( ) **kernlab** and **caret**.

```
library(kernlab)
library(caret)
credit<-read_csv("https://ist387.s3.us-east-2.amazonaws.com/data/GermanCredit.csv")
```

B. Which variable contains the outcome we are trying to predict, **credit risk**? For the purposes of this analysis, we will focus only on the numeric variables and save them in a new dataframe called **cred**:

```
cred <- data.frame(duration=credit$duration,
                   amount=credit$amount,
                   installment_rate=credit$installment_rate,
                   present_residence=credit$present_residence,
                   age=credit$age,
                   credit_history=credit$number_credits,
                   people_liable=credit$people_liable,
                   credit_risk=as.factor(credit$credit_risk))
```

```
Error in data.frame(duration = credit$duration, amount = credit$amount, : object 'credit' not found
Traceback:


1. data.frame(duration = credit$duration, amount = credit$amount,
 .     installment_rate = credit$installment_rate, present_residence = credit$present_residence,
 .     age = credit$age, credit_history = credit$number_credits,
 .     people_liable = credit$people_liable, credit_risk = credit$credit_risk)
```

C. Although all variables in **cred** except **credit_risk** are coded as numeric, the values of one of them are also **ordered factors** rather than actual numbers. In consultation with the **data description link** from the intro, write a comment identifying the **factor variable** and briefly **describe** each variable in the dataframe.

```
View(cred)
#The factor variable is credit risk with a one being if there is credit risk and a zero being if there aren't and
it is if they are likely to pay back laons, duration is how long it was, amount is credit score, I'm not able to
 figure out what installment rate is, present residance is how many places they have to live in, age is how old t
hey are, credit history I'm not able figure out, and people liable is how many people they are liable for
```

Part 2: Create training and test data sets

D. Using techniques discussed in class, create **two datasets** – one for **training** and one for **testing**.

```
trainlist<-createDataPartition(y=cred$credit_risk, p=.70,list=FALSE)
trainSet<-cred[trainlist,]
testSet<-cred[-trainlist,]
```

E. Use the dim( ) function to demonstrate that the resulting training data set and test data set contain the appropriate number of cases.

```
dim(trainSet)
dim(testSet)
```

Part 3: Build a Model using ksvm( )

F. Build a support vector model using the **ksvm( )** function using all of the variables to predict **credit_risk**. Once you have specified the model statement and the name of the training data set, you can use the same parameters as shown on page 237 of the textbook:

```
#kernel= "rbfdot", kpar = "automatic", C = 5, cross = 3, prob.model = TRUE
ksvm(credit_risk~., data=trainSet,kernel= "rbfdot", kpar = "automatic", C = 5, cross = 3, prob.model = TRUE )
```

G. Write a block comment that summarizes what you learned from the book about those parameters. The two parameters of greatest interest are **C=5** and **cross=3**.

```
#kernel= "rbfdot" weighs the differant variables in order to get most seperation between 0 and 1 the two possible
values of credit risk, kpar being automatic means we are trusting parametars set by the creaters of the dataset,
 c is the cost of constraints making it low means our model makes more mistakes but is more general, cross says h
ow many times it will be valisated, in this case 3, and allows us to prevent overfitting
```

H. Store the output of the kvsm( ) run in a variable called **svmOut** and then **echo** that variable to the console. You will know that you are on the right track if your cross-validation error is reported in the neighborhood of **0.3**. The other output information is mainly diagnostic and is not of great concern at this time.

```
svmOut<-ksvm(credit_risk~., data=trainSet,kernel= "rbfdot", kpar = "automatic", C = 5, cross = 3, prob.model = TR
UE )
svmOut
```

Part 4: Predict Values in the Test Data and Create a Confusion Matrix

I. Use the predict( ) function to validate the model against test data. Store the predictions in a variable named **svmPred**.

```
svmPred<-predict(svmOut, testSet, type="response")
```

J. The **svmPred** object contains a list of classifications for reliable (=0) or risky (=1) borrowers. Review the contents of **svmPred** using View(), str( ), or head( ).

```
View(svmPred)
str(svmPred)
head(svmPred)
```

K. Create a **confusion matrix** (a 2 x 2 table) that compares **svmPred** to the contents of **testSet$credit_risk**.

```
table(testSet$credit_risk,svmPred)
```

L. Calculate an **error rate** based on what you see in the confusion matrix. See pages 243-244 of the textbook for more information.

```
(18+71)/300
```

M. Compare your calculations with the **confusionMatrix()** function from the **caret** package.

```
confusionMatrix(testSet$credit_risk, svmPred)
#it says it is about 70% accurate and the error rate said it was about 30% inaccurate so the calculations are cor
rect
```

N. Explain, in a block comment:
1) why it is valuable to have a “test” dataset that is separate from a “training” dataset, and
2) what potential ethical challenges this type of automated classification may pose.

```
# It is important to have a test and train sets so that you can create a model and then see how good it is with c
ompleatly new data, some ethical challenges might be that the data doesn't give any information about the people
 it is gathering data on so there would be no way to know if it has any inherrant bias
```