# Intro to Data Science HW 9

##### Copyright Jeffrey Stanton, Jeffrey Saltz, and Jasmina Tacheva

# Rutwik Ghag

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

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

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.

**Support vector machines (SVM)** are a highly flexible and powerful method of doing **supervised machine learning**.

Another approach is to use **partition trees (rpart)**

In this homework, we will use another banking dataset to train an SVM model, as well as an rpart 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

1. Read the contents of the following .csv file into a dataframe called **credit**:

<https://intro-datascience.s3.us-east-2.amazonaws.com/GermanCredit.csv>

You will also need to install( ) and library( ) several other libraries, such as **kernlab** and **caret**.

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──

## ✓ ggplot2 3.3.5 ✓ purrr 0.3.4  
## ✓ tibble 3.1.4 ✓ dplyr 1.0.7  
## ✓ tidyr 1.1.3 ✓ stringr 1.4.0  
## ✓ readr 2.0.1 ✓ forcats 0.5.1

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(kernlab)

##   
## Attaching package: 'kernlab'

## The following object is masked from 'package:purrr':  
##   
## cross

## The following object is masked from 'package:ggplot2':  
##   
## alpha

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

credit <- read.csv("https://intro-datascience.s3.us-east-2.amazonaws.com/GermanCredit.csv")

1. 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$people\_liable, credit\_risk = as.factor(credit$credit\_risk))

1. 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.

#credit risk is the factor variable.  
#duration - duration of the loan in months  
#amount - credit amount  
#installment\_rate - Installment rate in percentage of disposable income  
#present\_residence - present residence since (# no. of months)  
#age - age of the applicant  
#credit\_history - number of existing credits at this bank  
#people\_liable - Number of people being liable to provide maintenance for  
#credit\_risk - binary code, dividing the applicant as reliable borrower or non-reliable borrower

## Part 2: Create training and test data sets

1. 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,]

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

dim(trainSet)

## [1] 700 8

dim(testSet)

## [1] 300 8

## Part 3: Build a Model using SVM

1. Using the caret package, build a support vector model using all of the variables to predict **credit\_risk**

svm.Model <- train(credit\_risk ~., data = trainSet, method = "svmRadial", preProcess = c("center", "scale"), tuneLength = 10)  
#svm.Model <- ksvm(credit\_risk ~ ., data = trainSet, C= 5, cross = 3, prob.model = TRUE)

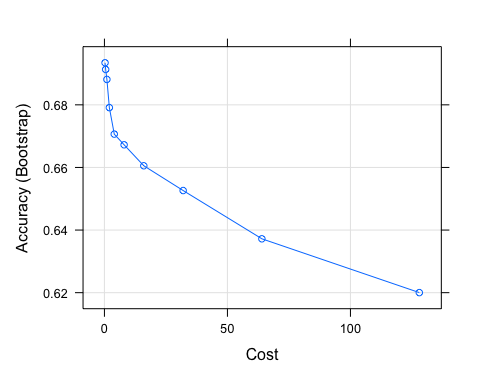
B. output the model

Hint: explore finalModel in the model that would created in F.

svm.Model

## Support Vector Machines with Radial Basis Function Kernel   
##   
## 700 samples  
## 7 predictor  
## 2 classes: '0', '1'   
##   
## Pre-processing: centered (7), scaled (7)   
## Resampling: Bootstrapped (25 reps)   
## Summary of sample sizes: 700, 700, 700, 700, 700, 700, ...   
## Resampling results across tuning parameters:  
##   
## C Accuracy Kappa   
## 0.25 0.6934366 0.01207023  
## 0.50 0.6913010 0.03992506  
## 1.00 0.6881184 0.06886762  
## 2.00 0.6791090 0.08142897  
## 4.00 0.6706491 0.08465021  
## 8.00 0.6672347 0.10038766  
## 16.00 0.6605388 0.10372714  
## 32.00 0.6526243 0.10457852  
## 64.00 0.6372063 0.09485261  
## 128.00 0.6200183 0.07764459  
##   
## Tuning parameter 'sigma' was held constant at a value of 0.1465513  
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were sigma = 0.1465513 and C = 0.25.

plot(svm.Model)



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

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

svmPred <- predict(svm.Model, newdata = testSet, type = "raw")

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

head(svmPred)

## [1] 1 1 1 1 1 1  
## Levels: 0 1

#Our model has predicted a few reliable and risky borrowers

1. Explore the **confusion matrix**, using the caret package

confusionMatrix(svmPred, testSet$credit\_risk)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 0 0  
## 1 90 210  
##   
## Accuracy : 0.7   
## 95% CI : (0.6447, 0.7513)  
## No Information Rate : 0.7   
## P-Value [Acc > NIR] : 0.5284   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.0   
## Specificity : 1.0   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.7   
## Prevalence : 0.3   
## Detection Rate : 0.0   
## Detection Prevalence : 0.0   
## Balanced Accuracy : 0.5   
##   
## 'Positive' Class : 0   
##

1. What is the **accuracy** based on what you see in the confusion matrix.

#The accuracy is 70%

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

str(svmPred)

## Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...

table(svmPred,testSet$credit\_risk)

##   
## svmPred 0 1  
## 0 0 0  
## 1 90 210

1. 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.

#1) We would want to know how well our model might perform on unseen data, hence we divide our datasets to see how well our model performs i.e is it generalizing things well or overfitting it  
#2) Our model performs based on the data/parameters made available to it, so we can see bias , unfair discrimination and lack of transparency at times with regards to how the model comes with conclusions.

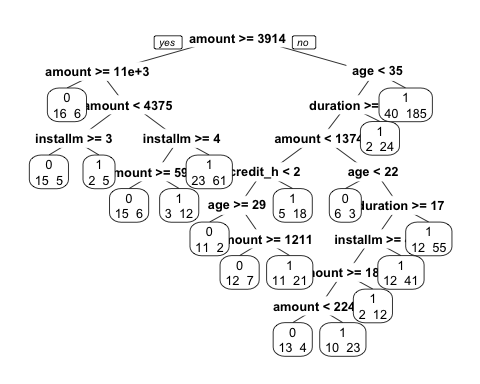
## Part 5: Now build a tree model (with rpart)

A. Build a model with rpart Note: you might need to install the e1071 package

library(rpart)  
cartTree1 <- rpart(credit\_risk ~ ., data = trainSet)

B. Visualize the results using rpart.plot()

library('rpart.plot')  
prp(cartTree1, faclen = 0, cex = 0.8, extra = 1)



C. Use the **predict()** function to predict the testData, and then generate a confusion matrix to explore the results

predictValues <- predict(cartTree1, newdata=testSet,   
type = "class")  
  
  
credited\_risk <- as.factor(testSet$credit\_risk == 0)  
confMatrix <- table(predictValues,credited\_risk)  
confMatrix

## credited\_risk  
## predictValues FALSE TRUE  
## 0 21 19  
## 1 189 71

D. Review the attributes being used for this credit decision. Are there any that might not be appropriate, with respect to fairness? If so, which attribute, and how would you address this fairness situation. Answer in a comment block below

#The personal status such as sex and marital status is not required