Capstone Adult Data

Mark Stallworth

February 5, 2018

1. Business Understanding: Goal is to predict income greater than 50K based on attributes of the adult.data.txt dataset

Header4: Load the required packages and libraries

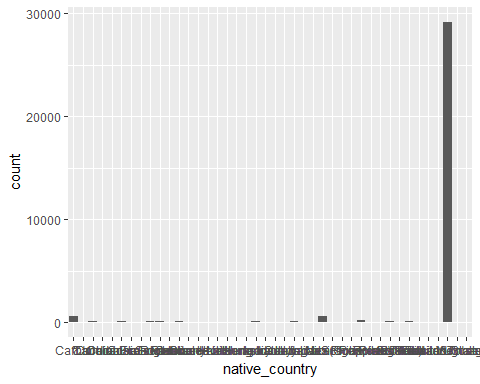
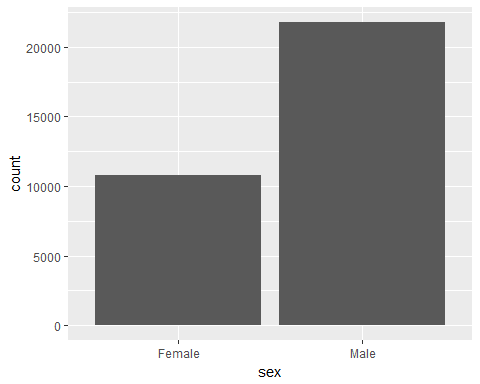
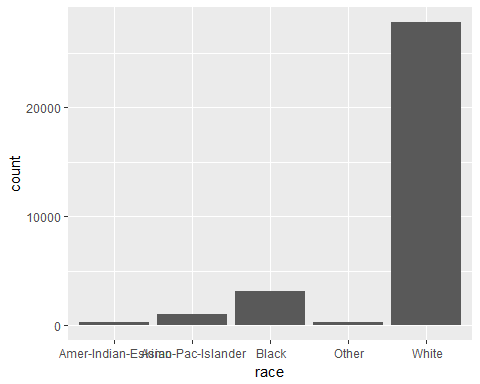
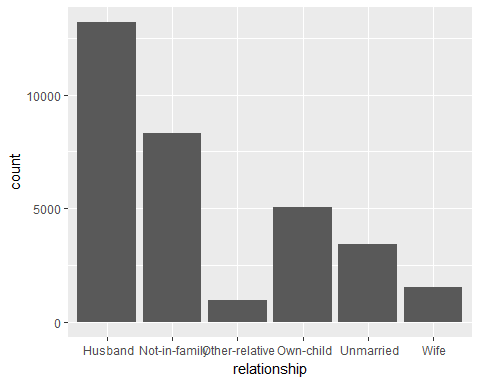
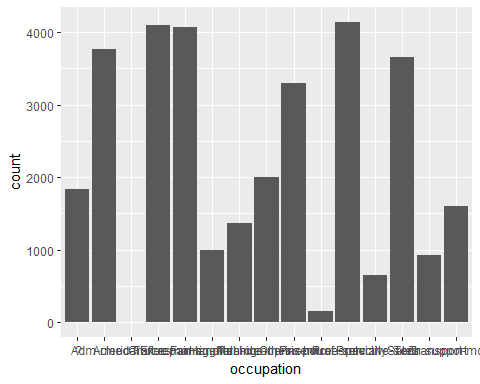
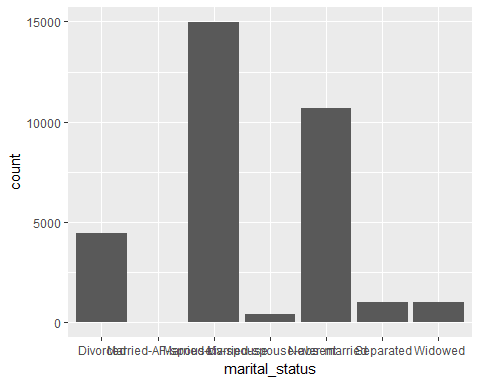
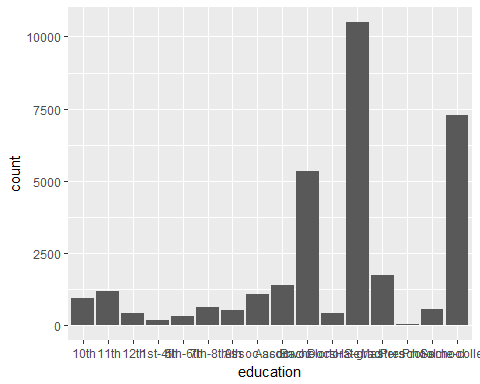
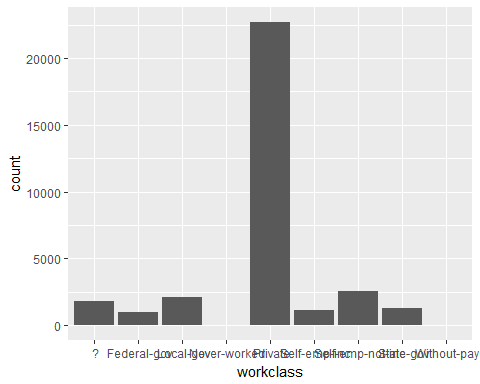
1. Data Understanding

# Review data structures and browse data

## 'data.frame': 32561 obs. of 15 variables:  
## $ age : int 39 50 38 53 28 37 49 52 31 42 ...  
## $ workclass : Factor w/ 9 levels " ?"," Federal-gov",..: 8 7 5 5 5 5 5 7 5 5 ...  
## $ fnlwwgt : int 77516 83311 215646 234721 338409 284582 160187 209642 45781 159449 ...  
## $ education : Factor w/ 16 levels " 10th"," 11th",..: 10 10 12 2 10 13 7 12 13 10 ...  
## $ education\_num : int 13 13 9 7 13 14 5 9 14 13 ...  
## $ marital\_status: Factor w/ 7 levels " Divorced"," Married-AF-spouse",..: 5 3 1 3 3 3 4 3 5 3 ...  
## $ occupation : Factor w/ 15 levels " ?"," Adm-clerical",..: 2 5 7 7 11 5 9 5 11 5 ...  
## $ relationship : Factor w/ 6 levels " Husband"," Not-in-family",..: 2 1 2 1 6 6 2 1 2 1 ...  
## $ race : Factor w/ 5 levels " Amer-Indian-Eskimo",..: 5 5 5 3 3 5 3 5 5 5 ...  
## $ sex : Factor w/ 2 levels " Female"," Male": 2 2 2 2 1 1 1 2 1 2 ...  
## $ capital\_gain : int 2174 0 0 0 0 0 0 0 14084 5178 ...  
## $ capital\_loss : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ hours\_per\_week: int 40 13 40 40 40 40 16 45 50 40 ...  
## $ native\_country: Factor w/ 42 levels " ?"," Cambodia",..: 40 40 40 40 6 40 24 40 40 40 ...  
## $ income50 : Factor w/ 2 levels " <=50K"," >50K": 1 1 1 1 1 1 1 2 2 2 ...

## [1] "data.frame"

# Data Exploration Factor Variables

 # Data Exploration Factor Variables with "?" in columns of observations select counts

## # A tibble: 9 x 2  
## `rawData$workclass` count  
## <fct> <int>  
## 1 " ?" 1836  
## 2 " Federal-gov" 960  
## 3 " Local-gov" 2093  
## 4 " Never-worked" 7  
## 5 " Private" 22696  
## 6 " Self-emp-inc" 1116  
## 7 " Self-emp-not-inc" 2541  
## 8 " State-gov" 1298  
## 9 " Without-pay" 14

## # A tibble: 15 x 2  
## `rawData$occupation` count  
## <fct> <int>  
## 1 " ?" 1843  
## 2 " Adm-clerical" 3770  
## 3 " Armed-Forces" 9  
## 4 " Craft-repair" 4099  
## 5 " Exec-managerial" 4066  
## 6 " Farming-fishing" 994  
## 7 " Handlers-cleaners" 1370  
## 8 " Machine-op-inspct" 2002  
## 9 " Other-service" 3295  
## 10 " Priv-house-serv" 149  
## 11 " Prof-specialty" 4140  
## 12 " Protective-serv" 649  
## 13 " Sales" 3650  
## 14 " Tech-support" 928  
## 15 " Transport-moving" 1597

## # A tibble: 42 x 2  
## `rawData$native\_country` count  
## <fct> <int>  
## 1 " ?" 583  
## 2 " Cambodia" 19  
## 3 " Canada" 121  
## 4 " China" 75  
## 5 " Columbia" 59  
## 6 " Cuba" 95  
## 7 " Dominican-Republic" 70  
## 8 " Ecuador" 28  
## 9 " El-Salvador" 106  
## 10 " England" 90  
## # ... with 32 more rows

# Check for missing values in integer columns:

sum(is.null(rawData$age))

## [1] 0

sum(is.null(rawData$fnlwwgt))

## [1] 0

sum(is.null(rawData$education\_num))

## [1] 0

sum(is.null(rawData$capital\_gain))

## [1] 0

sum(is.null(rawData$capital\_loss))

## [1] 0

sum(is.null(rawData$hours\_per\_week))

## [1] 0

3 Data Preparation

#identify factor variables with potential missing data and exclude label income50  
factor\_idx = which(unlist(lapply(rawData, class)) == "factor")  
factor\_idx\_depvar <- (factor\_idx[-9]) #remove column income50  
  
#Identify relative column number for factors to be used to replace "?" with "NA"  
for (i in 1:NROW(factor\_idx\_depvar)) {  
 naidx <- which(as.character(rawData[, factor\_idx\_depvar[i]]) == " ?")  
 rawData[naidx, factor\_idx\_depvar[i]] <- NA  
}  
#Drop category levels in order to remove "?" level and replace with NA; other wise imputaion substitutes "?" from original data  
rawData <- drop.levels(rawData)  
  
#replace NA values with K nearest neighbor  
?knnImputation  
rawDataClean <- knnImputation(rawData, 4)  
  
#Verify clean data for the 3 columns (workclass, occupation and native\_country) that contains missing data in source input file (adult.data.txt)  
select(rawDataClean) %>% group\_by(rawDataClean$workclass) %>% summarise(count = n())

## # A tibble: 8 x 2  
## `rawDataClean$workclass` count  
## <fct> <int>  
## 1 " Federal-gov" 2244  
## 2 " Local-gov" 2572  
## 3 " Never-worked" 75  
## 4 " Private" 22701  
## 5 " Self-emp-inc" 1116  
## 6 " Self-emp-not-inc" 2541  
## 7 " State-gov" 1298  
## 8 " Without-pay" 14

select(rawDataClean) %>% group\_by(rawDataClean$occupation) %>% summarise(count = n())

## # A tibble: 14 x 2  
## `rawDataClean$occupation` count  
## <fct> <int>  
## 1 " Adm-clerical" 4453  
## 2 " Armed-Forces" 624  
## 3 " Craft-repair" 4520  
## 4 " Exec-managerial" 4190  
## 5 " Farming-fishing" 994  
## 6 " Handlers-cleaners" 1370  
## 7 " Machine-op-inspct" 2002  
## 8 " Other-service" 3295  
## 9 " Priv-house-serv" 149  
## 10 " Prof-specialty" 4140  
## 11 " Protective-serv" 649  
## 12 " Sales" 3650  
## 13 " Tech-support" 928  
## 14 " Transport-moving" 1597

select(rawDataClean) %>% group\_by(rawDataClean$native\_country) %>% summarise(count = n())

## # A tibble: 41 x 2  
## `rawDataClean$native\_country` count  
## <fct> <int>  
## 1 " Cambodia" 420  
## 2 " Canada" 282  
## 3 " China" 94  
## 4 " Columbia" 61  
## 5 " Cuba" 95  
## 6 " Dominican-Republic" 70  
## 7 " Ecuador" 28  
## 8 " El-Salvador" 106  
## 9 " England" 90  
## 10 " France" 29  
## # ... with 31 more rows

1. Modeling

## [1] "data.frame"

## [1] 32561 13

trainRows = sample(seq(1,nrow(processedData)),0.75\*nrow(processedData))  
length(trainRows)

## [1] 24420

trainData = processedData[trainRows,]  
dim(trainData)

## [1] 24420 13

#testRows = setdiff(seq(1,nrow(inputData), trainRows)  
testRows = setdiff(seq(1, nrow(processedData)), trainRows)  
  
testData = processedData[testRows,]  
dim(testData)

## [1] 8141 13

#Process a sample of training rows due to processing constraings and educational objective per Mentor  
trainData <- trainData[sample(1:nrow(trainData), 5000, replace=FALSE),]  
nrow(trainData)

## [1] 5000

initialModel = svm(x=trainData[,1:(ncol(trainData)-1)],y=trainData[,ncol(trainData)])

## Warning in svm.default(x = trainData[, 1:(ncol(trainData) -  
## 1)], y = trainData[, : Variable(s) 'workclass..Without.pay' and  
## 'native\_country..Holand.Netherlands' constant. Cannot scale data.

summary(initialModel)

##   
## Call:  
## svm.default(x = trainData[, 1:(ncol(trainData) - 1)], y = trainData[,   
## ncol(trainData)])  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
## gamma: 0.08333333   
##   
## Number of Support Vectors: 2017  
##   
## ( 926 1091 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## <=50K >50K

initialModelPredict <- predict(initialModel,trainData[,1:(ncol(trainData)-1)])  
str(initialModelPredict)

## Factor w/ 2 levels " <=50K"," >50K": 1 2 1 1 1 1 1 1 1 1 ...  
## - attr(\*, "names")= chr [1:5000] "14001" "14321" "12456" "11553" ...

NROW(initialModelPredict)

## [1] 5000

svm.table <- table(initialModelPredict, trainData$income50)  
classAgreement(svm.table)

## $diag  
## [1] 0.8526  
##   
## $kappa  
## [1] 0.5349402  
##   
## $rand  
## [1] 0.7486032  
##   
## $crand  
## [1] 0.4194515

confusionMatrix(svm.table)

## Confusion Matrix and Statistics  
##   
##   
## initialModelPredict <=50K >50K  
## <=50K 3657 566  
## >50K 171 606  
##   
## Accuracy : 0.8526   
## 95% CI : (0.8425, 0.8623)  
## No Information Rate : 0.7656   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5349   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.9553   
## Specificity : 0.5171   
## Pos Pred Value : 0.8660   
## Neg Pred Value : 0.7799   
## Prevalence : 0.7656   
## Detection Rate : 0.7314   
## Detection Prevalence : 0.8446   
## Balanced Accuracy : 0.7362   
##   
## 'Positive' Class : <=50K   
##

**tunedInitialModel2 <-** tune**(svm, train.x = trainData[,-13], train.y=trainData$income50, kernel="radial", ranges=**list**(cost=**exp**(**c**(-1,0,3,5,7,9)), gamma=**exp**(**c**(-1,0,3,5,7,9))))**

## Warning in svm.default(structure(list(age = c(-0.629133036772184,  
## 0.323915918892115, : Variable(s) 'workclass..Without.pay' and  
## 'native\_country..Holand.Netherlands' and 'native\_country..Yugoslavia'  
## constant. Cannot scale data.

## Warning in svm.default(structure(list(age = c(0.69047320953223,  
## -1.21562470179637, : Variable(s) 'workclass..Without.pay' and  
## 'native\_country..Holand.Netherlands' constant. Cannot scale data.

erlands' constant. Cannot scale data.

:

: 20+ pages warning message native\_country Cannot scale data.

:

## Warning in svm.default(structure(list(age = c(0.69047320953223,  
## -1.21562470179637, : Variable(s) 'workclass..Without.pay' and  
## 'native\_country..Holand.Netherlands' and 'native\_country..Scotland'  
## constant. Cannot scale data.

## Warning in svm.default(structure(list(age = c(0.69047320953223,  
## -1.21562470179637, : Variable(s) 'workclass..Without.pay' and  
## 'native\_country..Holand.Netherlands' constant. Cannot scale data.

## Warning in svm.default(structure(list(age = c(0.69047320953223,  
## -1.21562470179637, : Variable(s) 'workclass..Without.pay' and  
## 'native\_country..Holand.Netherlands' constant. Cannot scale data.

## Warning in svm.default(x = structure(list(age = c(1.13034195830037,  
## 0.323915918892115, : Variable(s) 'workclass..Without.pay' and  
## 'native\_country..Holand.Netherlands' constant. Cannot scale data.

summary(tunedInitialModel2)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 1 0.3678794  
##   
## - best performance: 0.1678   
##   
## - Detailed performance results:  
## cost gamma error dispersion  
## 1 0.3678794 0.3678794 0.1776 0.01602221  
## 2 1.0000000 0.3678794 0.1678 0.01595689  
## 3 20.0855369 0.3678794 0.1978 0.01620562  
## 4 148.4131591 0.3678794 0.2082 0.01695615  
## 5 1096.6331584 0.3678794 0.2154 0.01756385  
## 6 8103.0839276 0.3678794 0.2230 0.01458310  
## 7 0.3678794 1.0000000 0.2110 0.02138016  
## 8 1.0000000 1.0000000 0.2020 0.01469694  
## 9 20.0855369 1.0000000 0.2132 0.01987069  
## 10 148.4131591 1.0000000 0.2140 0.01918333  
## 11 1096.6331584 1.0000000 0.2116 0.02043526  
## 12 8103.0839276 1.0000000 0.2146 0.01580577  
## 13 0.3678794 20.0855369 0.2344 0.02301304  
## 14 1.0000000 20.0855369 0.2326 0.02268725  
## 15 20.0855369 20.0855369 0.2332 0.02257235  
## 16 148.4131591 20.0855369 0.2332 0.02257235  
## 17 1096.6331584 20.0855369 0.2332 0.02257235  
## 18 8103.0839276 20.0855369 0.2332 0.02257235  
## 19 0.3678794 148.4131591 0.2344 0.02301304  
## 20 1.0000000 148.4131591 0.2346 0.02303717  
## 21 20.0855369 148.4131591 0.2344 0.02283856  
## 22 148.4131591 148.4131591 0.2344 0.02283856  
## 23 1096.6331584 148.4131591 0.2344 0.02283856  
## 24 8103.0839276 148.4131591 0.2344 0.02283856  
## 25 0.3678794 1096.6331584 0.2344 0.02301304  
## 26 1.0000000 1096.6331584 0.2344 0.02301304  
## 27 20.0855369 1096.6331584 0.2344 0.02301304  
## 28 148.4131591 1096.6331584 0.2344 0.02301304  
## 29 1096.6331584 1096.6331584 0.2344 0.02301304  
## 30 8103.0839276 1096.6331584 0.2344 0.02301304  
## 31 0.3678794 8103.0839276 0.2344 0.02301304  
## 32 1.0000000 8103.0839276 0.2344 0.02301304  
## 33 20.0855369 8103.0839276 0.2344 0.02301304  
## 34 148.4131591 8103.0839276 0.2344 0.02301304  
## 35 1096.6331584 8103.0839276 0.2344 0.02301304  
## 36 8103.0839276 8103.0839276 0.2344 0.02301304

print(tunedInitialModel2)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 1 0.3678794  
##   
## - best performance: 0.1678