Classification SVM

Jonathan Ho

October 20, 2022

Load packages and data

```
library(e1071)
library(MASS)
df <- read.csv("ill_dataset.csv", header=TRUE)
df <- subset(df, select=-c(1))
df$City <- factor(df$City)
df$Gender <- factor(df$Gender)
df$Illness <- factor(df$Illness)
```

Divide into train, test, and validate

Load in only 10,000 randoms rows of data due to long loading times of SVM kernels.

```
set.seed(420)
spec <- c(train = 0.6, test = 0.2, validate = 0.2)
i <- sample(cut(1:nrow(df), nrow(df)*cumsum(c(0,spec)), labels=names(spec)))
train <- df[i=="train",]
test <- df[i=="test",]
vali <- df[i=="validate",]

train <- train[sample(nrow(train), 6000),]
test <- test[sample(nrow(test), 2000),]
vali <- vali[sample(nrow(vali), 2000),]
```

Data exploration

View all columns within the dataset.

```
str(df)
```

```
## 'data.frame':    150000 obs. of  5 variables:
##  $ City   : Factor w/ 8 levels "Austin","Boston",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Gender : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 1 1 2 2 1 ...
##  $ Age    : int  41 54 42 40 46 36 32 39 51 30 ...
##  $ Income : num  40367 45084 52483 40941 50289 ...
##  $ Illness: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
```

Check for NAs.

```
sapply(df, function(y) sum(is.na(y)))
```

```
##    City  Gender     Age  Income Illness
##       0       0       0       0       0
```

Display the number of rows and columns in the dataset.

```
dim(df)
```

```
## [1] 150000      5
```

```
dim(train)
```

```
## [1] 6000    5
```

```
dim(test)
```

```
## [1] 2000    5
```

```
dim(vali)
```

```
## [1] 2000    5
```

Summary of each column.

```
summary(df)
```

```
##              City          Gender           Age            Income      
##  New York City:50307   Female:66200   Min.   :25.00   Min.   :  -654  
##  Los Angeles  :32173   Male  :83800   1st Qu.:35.00   1st Qu.: 80868  
##  Dallas       :19707                  Median :45.00   Median : 93655  
##  Mountain View:14219                  Mean   :44.95   Mean   : 91253  
##  Austin       :12292                  3rd Qu.:55.00   3rd Qu.:104519  
##  Boston       : 8301                  Max.   :65.00   Max.   :177157  
##  (Other)      :13001                                                  
##  Illness     
##  No :137861  
##  Yes: 12139  
##              
##              
##              
##              
## 
```

Logistic regression

```
glm1 <- glm(Gender~., data=train, family=binomial)
summary(glm1)
```

```
##
## Call:
## glm(formula = Gender ~ ., family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6238  -0.9971   0.5182   0.9283   2.5968
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -8.945e+00  3.260e-01 -27.436  < 2e-16 ***
## CityBoston           -1.791e-01  1.581e-01  -1.133   0.2573
## CityDallas            4.558e+00  1.951e-01  23.364  < 2e-16 ***
## CityLos Angeles      -4.948e-01  1.226e-01  -4.037 5.42e-05 ***
## CityMountain View    -4.389e+00  2.005e-01 -21.888  < 2e-16 ***
## CityNew York City    -5.500e-01  1.173e-01  -4.688 2.75e-06 ***
## CitySan Diego        -1.065e+00  1.931e-01  -5.517 3.45e-08 ***
## CityWashington D.C.   2.092e+00  1.733e-01  12.071  < 2e-16 ***
## Age                   4.651e-03  2.516e-03   1.848   0.0646 .
## Income                9.968e-05  3.188e-06  31.270  < 2e-16 ***
## IllnessYes            1.827e-01  1.081e-01   1.690   0.0910 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8217.1  on 5999  degrees of freedom
## Residual deviance: 6874.8  on 5989  degrees of freedom
## AIC: 6896.8
##
## Number of Fisher Scoring iterations: 4
```

Making base prediction (summary at end)

```
probs <- predict(glm1, newdata=test, type="response")
pred <- ifelse(probs>0.5, 2, 1)
acc1 <- mean(pred==as.integer(test$Gender))
```

Linear Kernel

Multiclass classification of Gender versus the rest of the predictors. Tuning is done to try and get the best cost. Gamma is not done since it is for non-linear kernels. A prediction is also done on the best linear svm.

```
tune_lsvm <- tune(svm, Gender~., data=vali, kernel="linear", range=list(cost=c(0.001, 0.01, 0.1,
1, 5, 10, 100)))
summary(tune_lsvm)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##    10
##
## - best performance: 0.3
##
## - Detailed performance results:
##    cost  error dispersion
## 1 1e-03 0.4575 0.03758324
## 2 1e-02 0.4200 0.04102845
## 3 1e-01 0.3045 0.02033743
## 4 1e+00 0.3025 0.02830881
## 5 5e+00 0.3020 0.03093003
## 6 1e+01 0.3000 0.03291403
## 7 1e+02 0.3005 0.03252777
```

Using the best cost

```
svm1 <- svm(Gender~., data=train, kernel="linear", cost=1, scale=TRUE)
summary(svm1)
```

```
##
## Call:
## svm(formula = Gender ~ ., data = train, kernel = "linear", cost = 1,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  4148
##
##  ( 2074 2074 )
##
##
## Number of Classes:  2
##
## Levels:
##  Female Male
```

Evaluating the prediction for a Linear Kernel

```
pred <- predict(svm1, newdata=test)
table(pred, test$Gender)
```

```
##
## pred     Female Male
##   Female    494  262
##   Male      365  879
```

```
acc2 <- mean(pred==test$Gender)
```

Polynomial Kernel

Using a Polynomial Kernel and making a prediction.

```
svm2 <- svm(Gender~., data=train, kernel="polynomial", cost=1, scale=TRUE)
summary(svm2)
```

```
##
## Call:
## svm(formula = Gender ~ ., data = train, kernel = "polynomial", cost = 1,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  1
##      degree:  3
##      coef.0:  0
##
## Number of Support Vectors:  5118
##
##  ( 2562 2556 )
##
##
## Number of Classes:  2
##
## Levels:
##  Female Male
```

```
pred <- predict(svm2, newdata=test)
acc3 <- mean(pred==test$Gender)
```

Radial Kernel

Tuning hyperparameters with different costs and gamma to find the best cost and gamma.

```
set.seed(420)
tune.out <- tune(svm, Gender~., data=vali, kernel="radial",
                 ranges=list(cost=c(0.1,1,10,100,1000),
                             gamma=c(0.5,1,2,3,4)))
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1   0.5
##
## - best performance: 0.3155
##
## - Detailed performance results:
##     cost gamma  error dispersion
## 1  1e-01   0.5 0.3225 0.02440970
## 2  1e+00   0.5 0.3155 0.02929259
## 3  1e+01   0.5 0.3170 0.02689486
## 4  1e+02   0.5 0.3230 0.02359378
## 5  1e+03   0.5 0.3290 0.02401388
## 6  1e-01   1.0 0.3265 0.02887232
## 7  1e+00   1.0 0.3175 0.02879525
## 8  1e+01   1.0 0.3220 0.02084333
## 9  1e+02   1.0 0.3285 0.01491643
## 10 1e+03   1.0 0.3410 0.02245984
## 11 1e-01   2.0 0.3455 0.02543510
## 12 1e+00   2.0 0.3170 0.02359378
## 13 1e+01   2.0 0.3235 0.01270389
## 14 1e+02   2.0 0.3320 0.01798147
## 15 1e+03   2.0 0.3495 0.01964264
## 16 1e-01   3.0 0.3610 0.03238655
## 17 1e+00   3.0 0.3170 0.02162817
## 18 1e+01   3.0 0.3330 0.01798147
## 19 1e+02   3.0 0.3515 0.01901023
## 20 1e+03   3.0 0.3695 0.02326777
## 21 1e-01   4.0 0.3775 0.02879525
## 22 1e+00   4.0 0.3180 0.01946507
## 23 1e+01   4.0 0.3365 0.01841648
## 24 1e+02   4.0 0.3585 0.01748809
## 25 1e+03   4.0 0.3735 0.01901023
```

Using best cost and gamma to do a prediction.

```
svm3 <- svm(Gender~., data=train, kernel="radial", cost=1, gamma=0.5, scale=TRUE)
summary(svm3)
```

```
##
## Call:
## svm(formula = Gender ~ ., data = train, kernel = "radial", cost = 1,
##     gamma = 0.5, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  4118
##
##  ( 2068 2050 )
##
##
## Number of Classes:  2
##
## Levels:
##  Female Male
```

```
pred <- predict(svm3, newdata=test)
acc4 <- mean(pred==test$Gender)
```

Summary of Results

```
cat("Logistic Regression:\n")
```

```
## Logistic Regression:
```

```
print(paste("accuracy: ", acc1))
```

```
## [1] "accuracy:  0.688"
```

```
cat("\nLinear Kernel:\n")
```

```
##
## Linear Kernel:
```

```
print(paste("accuracy: ", acc2))
```

```
## [1] "accuracy:  0.6865"
```

```
cat("\nPolynomial Kernel:\n")
```

```
##
## Polynomial Kernel:
```

```
print(paste("accuracy: ", acc3))
```

```
## [1] "accuracy:  0.5945"
```

```
cat("\nRadial Kernel:\n")
```

```
##
## Radial Kernel:
```

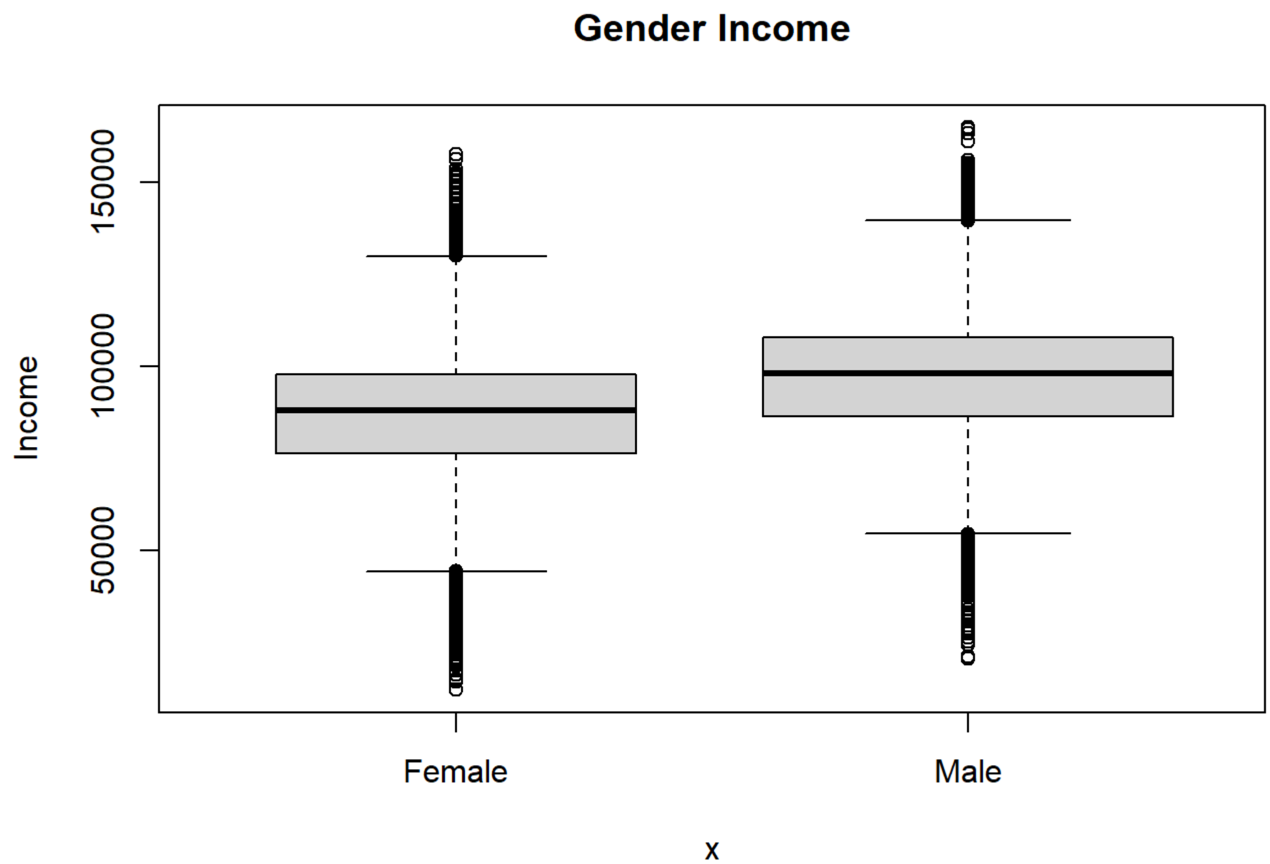
```
print(paste("accuracy: ", acc4))
```

```
## [1] "accuracy:  0.6845"
```

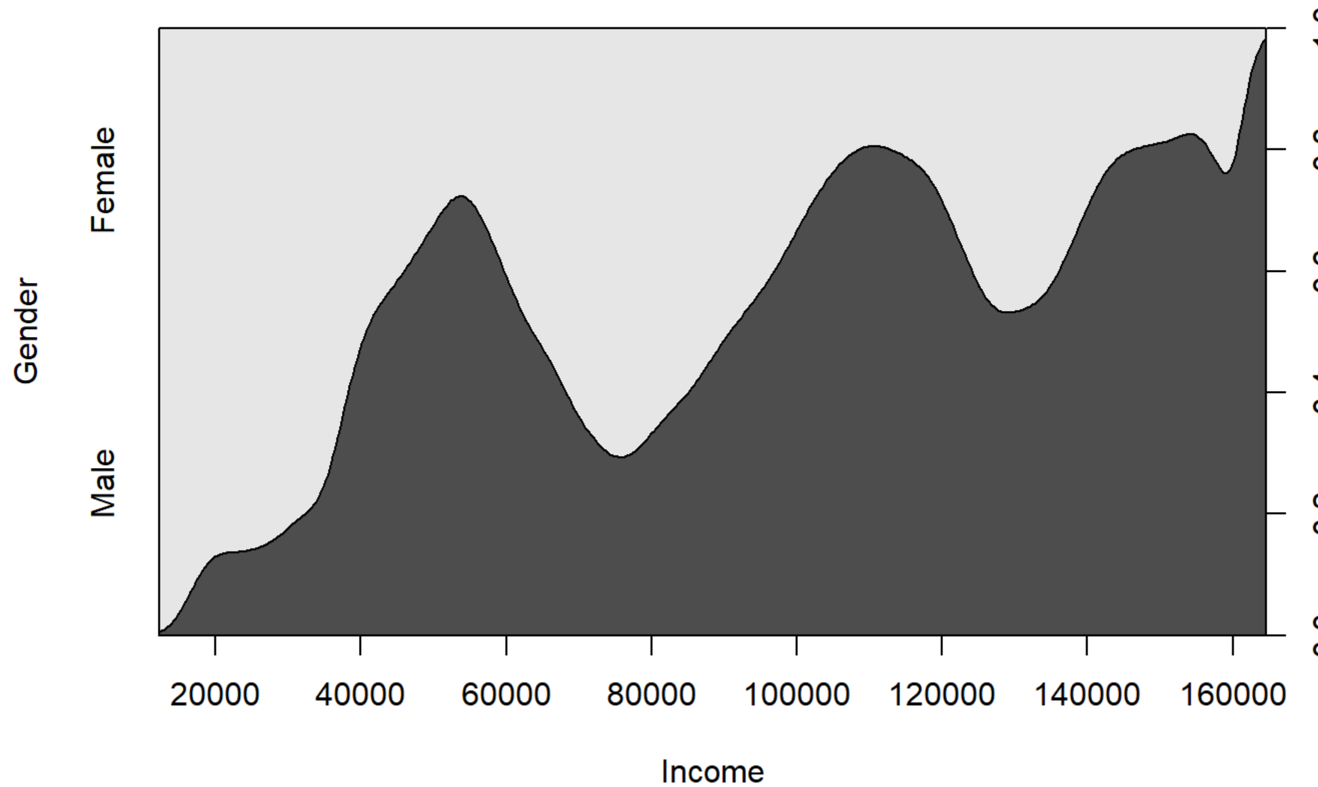
Data visualization

Box and CD plot of logistical regression on Gender and Income

```
plot(train$Gender, train$Income, main="Gender Income", ylab="Income", varwidth=TRUE)
```

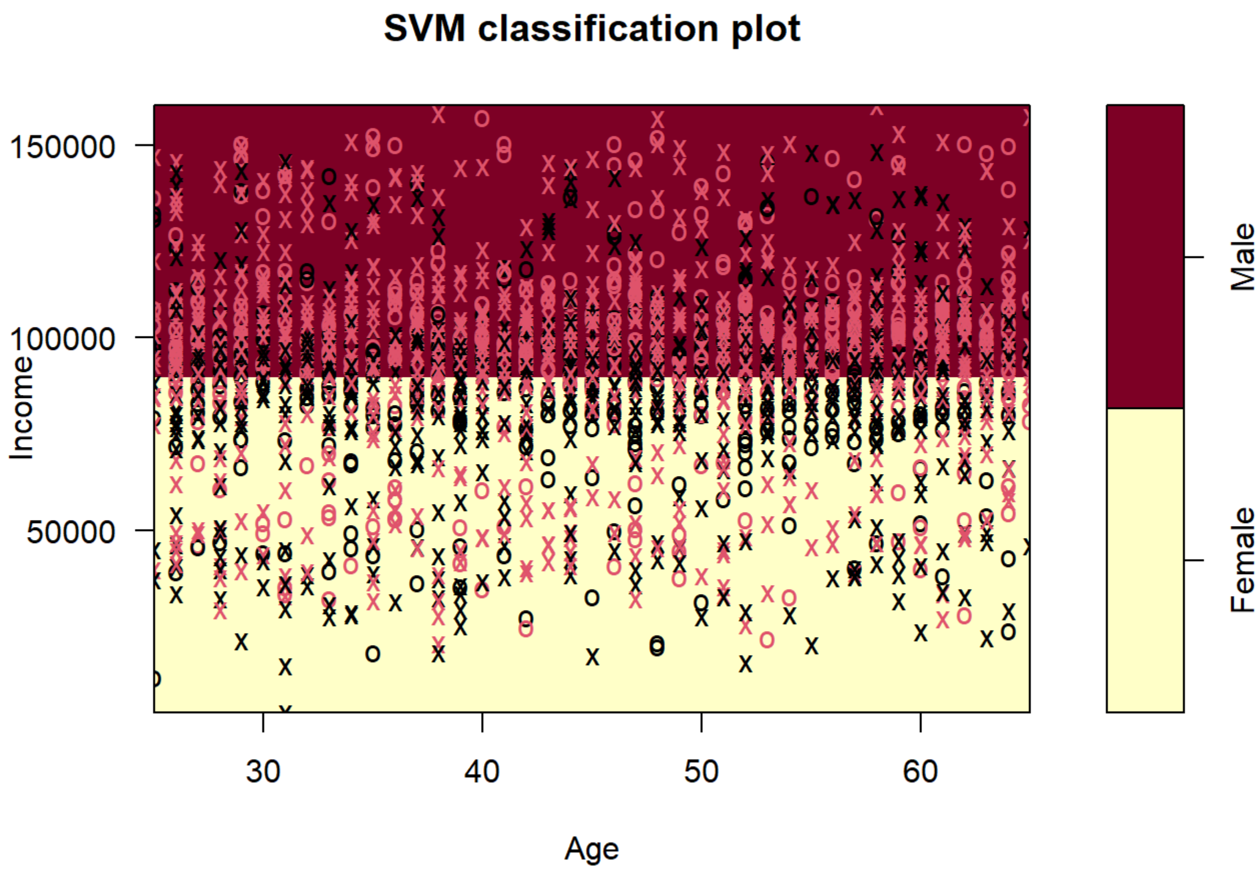



```
cdplot(train$Gender~train$Income, xlab="Income", ylab="Gender")
```



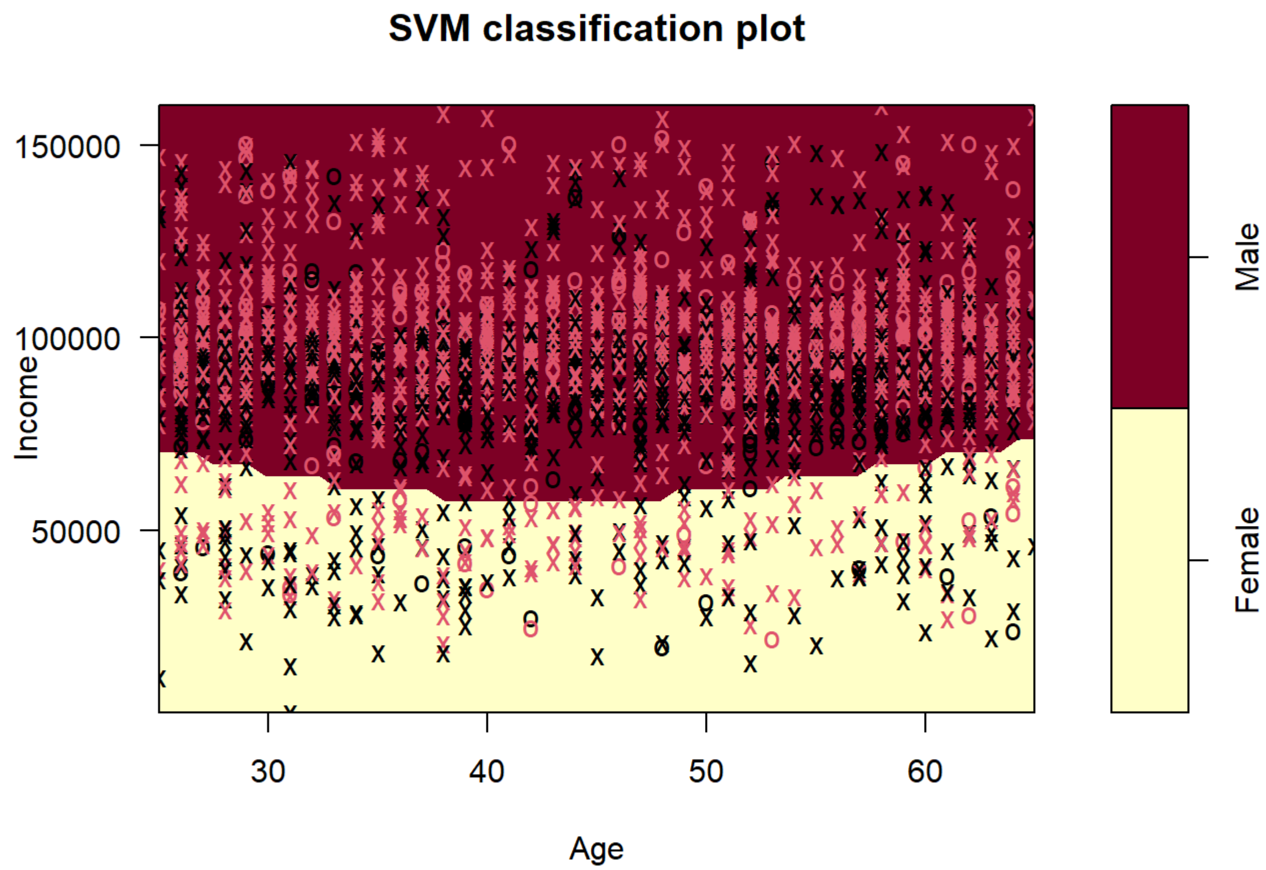
Plot of Linear Kernel with Gender, Income, and Age.

```
plot(svm1, test, Income~Age)
```



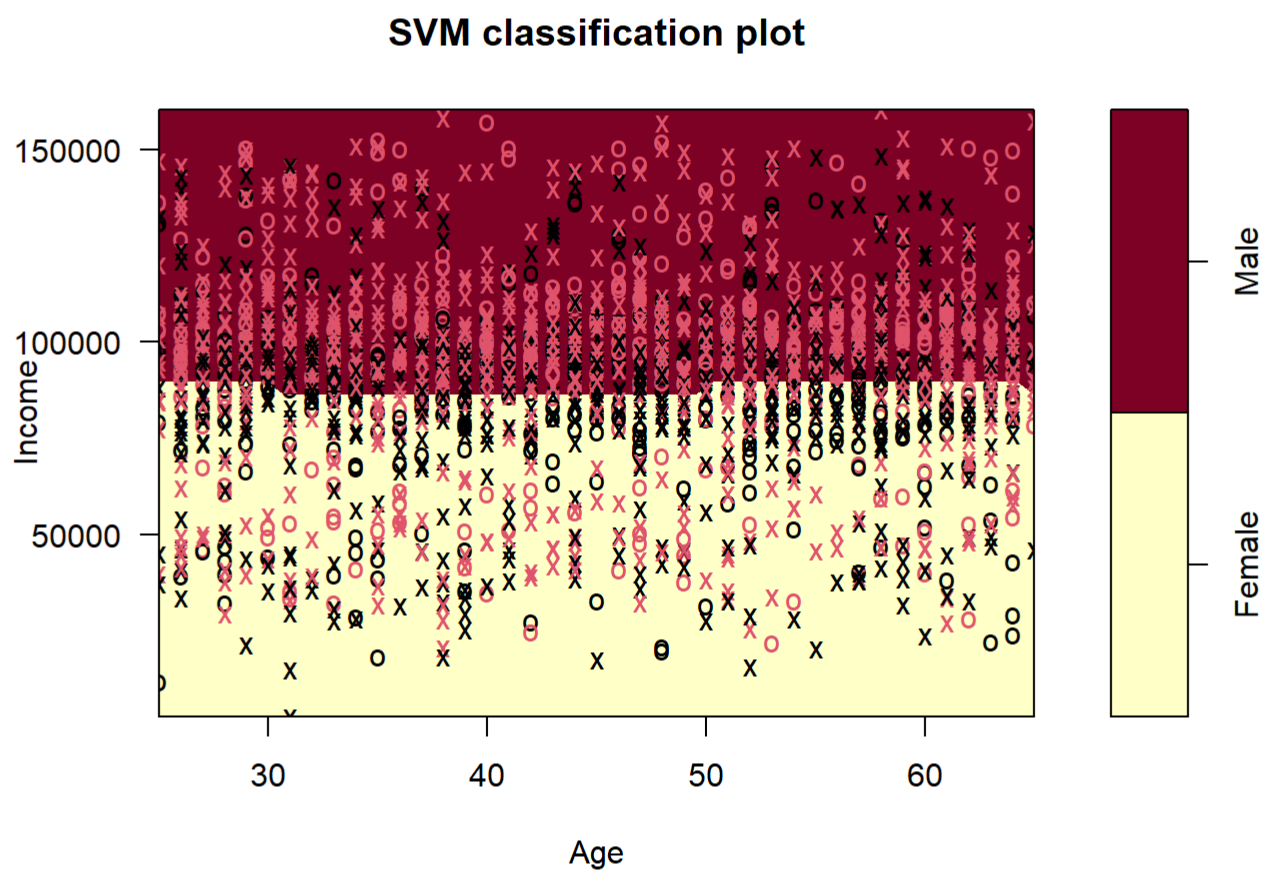
Plot of Polynomial Kernel on Gender, Income, and Age.

```
plot(svm2, test, Income~Age)
```



Plot of Radial Kernel on Gender, Income, and Age.

```
plot(svm3, test, Income~Age)
```



Results Discussion

Looking at the given metrics, the best SVM for classification accuracy is a Linear Kernel. However, the Linear Kernel still is slightly worse than straight Logistic Regression. Looking at the plots of each kernel, a polynomial kernel would not fit since it has more SVMs in the male than female. The Radial Kernel plot seems to try its best in encapsulating the SVMs, however, because the data is so spread out, it might be having trouble doing so. Comparing Linear and Radial Kernel plots, it can be seen why both are really close in accuracy since both still separate the data decently well.