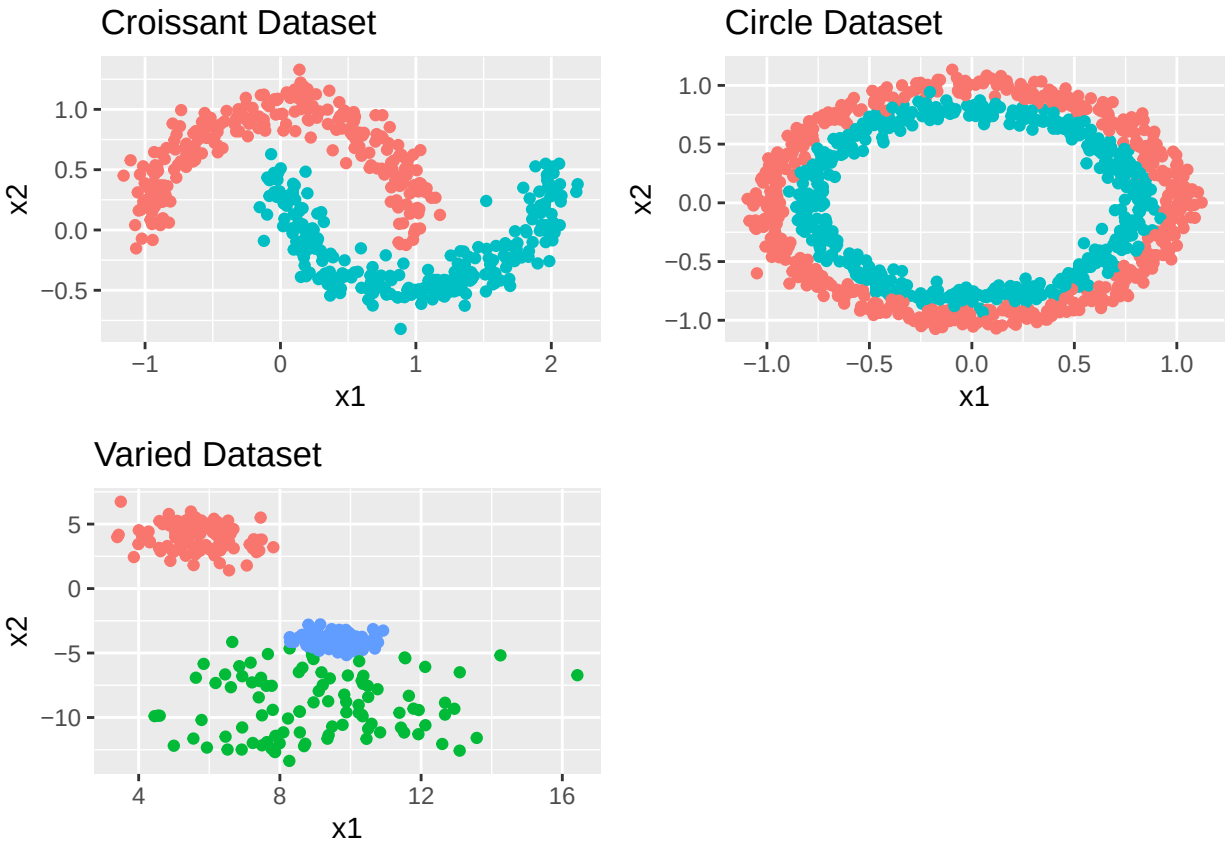
Assignment 3

Mitheysh Asokan, Eungjoo Kim

Question 1: Classification

1.1. Preprocess and Plot

```
croissant <- read.csv('croissant.csv')

circles <- read.csv('circles.csv')
varied <- read.csv('varied.csv')

croissant$y <- as.factor(croissant$y)
circles$y <- as.factor(circles$y)
varied$y <- as.factor(varied$y)

g1 <- ggplot(croissant, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("Croissant Dataset") +
  theme(legend.position = "none")

g2 <- ggplot(circles, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("Circle Dataset") +
  theme(legend.position = "none")

g3 <- ggplot(varied, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("Varied Dataset") +
  theme(legend.position = "none")

grid.arrange(g1,g2,g3,ncol=2)
```



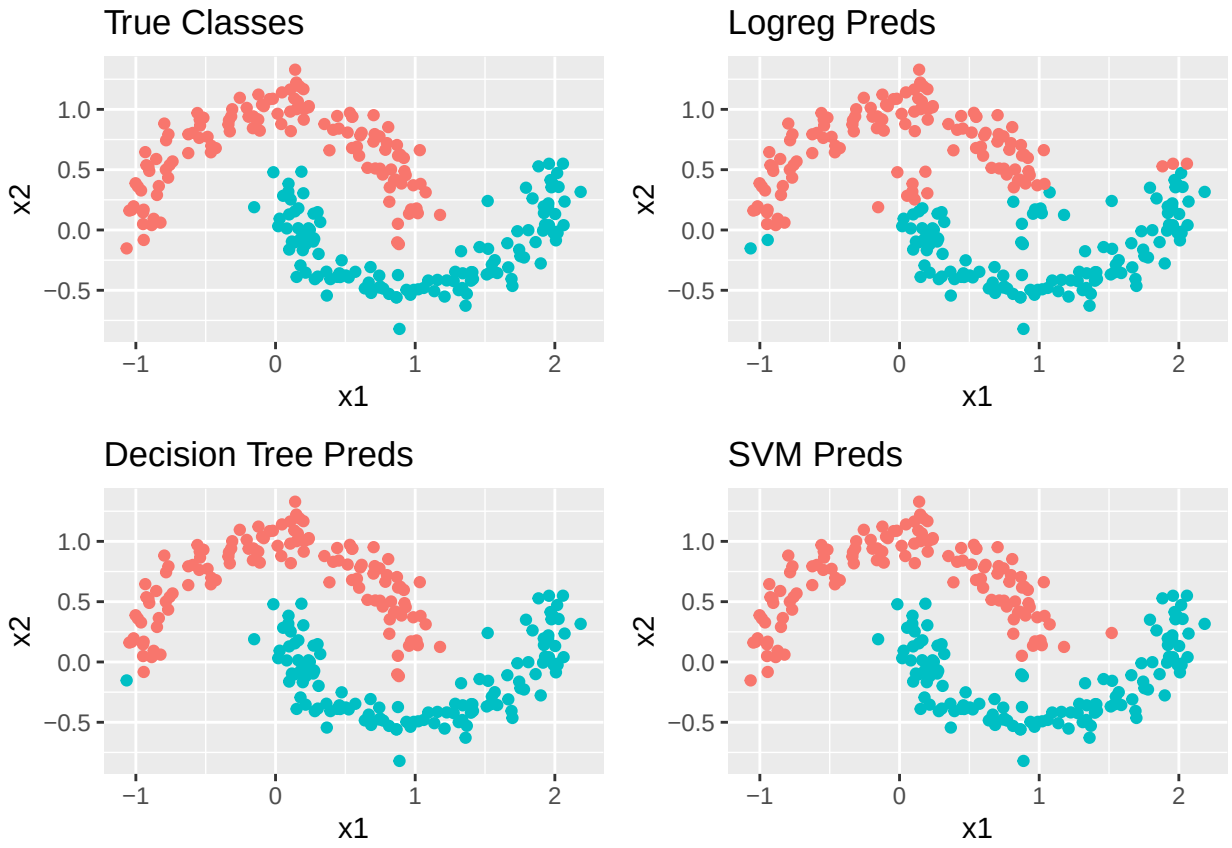
1.2. Train / Test split

```
set.seed(112)

dat <- croissant
train_ind <- sample(1:nrow(dat), floor(0.5*nrow(dat)))

train.croissant <- dat[ train_ind,]
test.croissant  <- dat[-train_ind,]

dat <- circles
train_ind <- sample(1:nrow(dat), floor(0.5*nrow(dat)))

train.circles <- dat[ train_ind,]
test.circles <- dat[-train_ind,]

dat <- varied
train_ind <- sample(1:nrow(dat), floor(0.5*nrow(dat)))

train.varied <- dat[ train_ind,]
test.varied <- dat[-train_ind,]
```

1.3. Train and Test

Croissants

```
logreg.croissant <- glm(y ~ x1+x2, data= train.croissant, family="binomial")
tree.croissant <- tree(y~x1+x2, data=train.croissant)
svmfit.croissant <-  svm(y ~ x1+x2, data=train.croissant , kernel ="radial",
                         cost =1,gamma =1,scale =FALSE)

preds.logreg <- predict(logreg.croissant,test.croissant,type = "response") > 0.5
preds.tree <- predict(tree.croissant,test.croissant, type="class")
preds.svm <- predict(svmfit.croissant,test.croissant, type="class")

g1 <- ggplot(test.croissant, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("True Classes") +
  theme(legend.position = "none")
g2 <- ggplot(test.croissant, aes(x1,x2,colour=preds.logreg)) +
  geom_point() +
  ggtitle("Logreg Preds") +
  theme(legend.position = "none")
g3 <- ggplot(test.croissant, aes(x1,x2,colour=preds.tree)) +
  geom_point() +
  ggtitle("Decision Tree Preds") +
  theme(legend.position = "none")
g4 <- ggplot(test.croissant, aes(x1,x2,colour=preds.svm)) +
  geom_point() +
  ggtitle("SVM Preds") +
  theme(legend.position = "none")

grid.arrange(g1,g2,g3,g4,ncol=2)
```




```
Accuracy(preds.logreg, test.croissant$y==1)
```

```
## [1] 0.904
```

```
ConfusionMatrix(preds.tree, test.croissant$y)
```

```
##       y_pred
## y_true   0   1
##      0 123   1
##      1   0 126
```

```
table(predict=preds.svm,actual=(test.croissant$y==1))
```

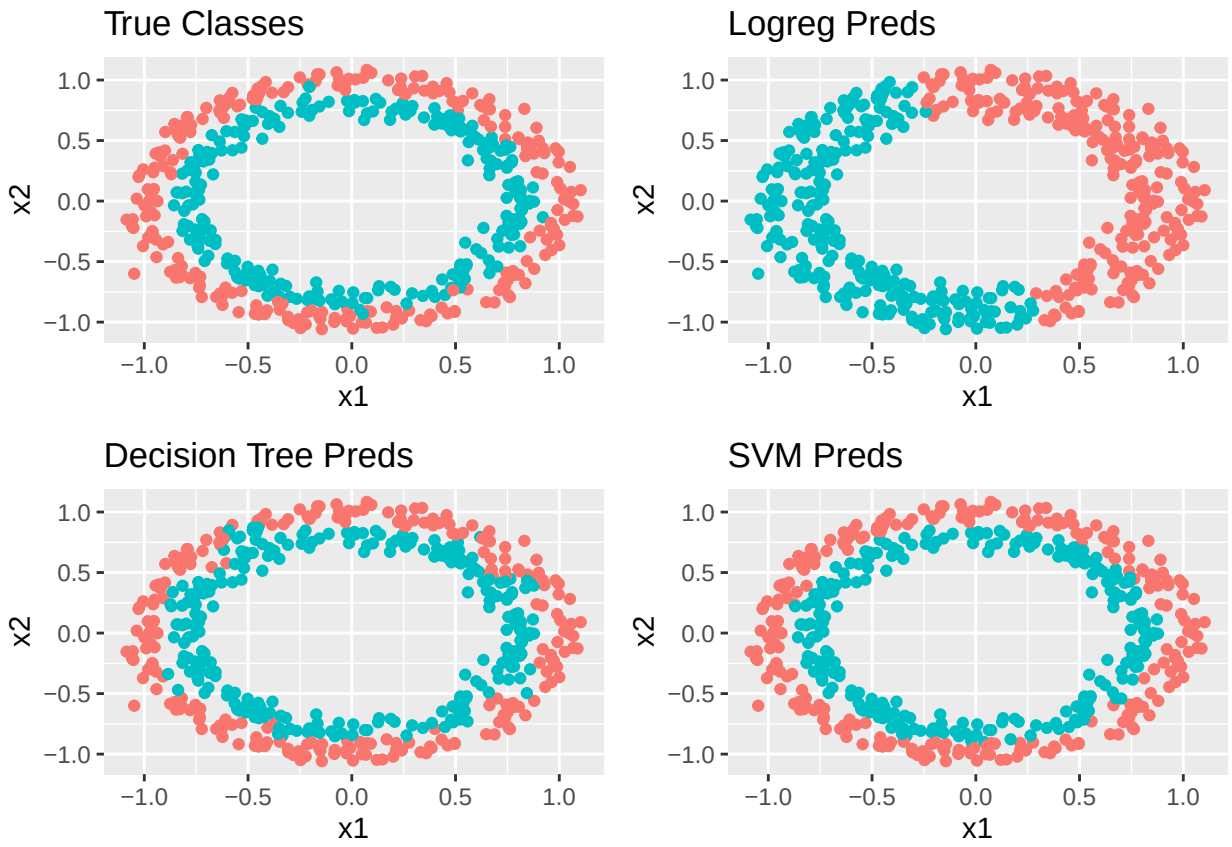
```
##         actual
## predict FALSE TRUE
##       0   122    1
##       1     2  125
```

Based on the results, the decision tree produced the highest accuracy.

Circles

```
logreg.circles <- glm(y ~ x1+x2, data= train.circles, family="binomial")
tree.circles <- tree(y~x1+x2, data=train.circles)
svmfit.circles <-  svm(y ~ x1+x2, data=train.circles , kernel ="radial",
                       cost =1, gamma=1,scale =FALSE)

preds.logreg <- predict(logreg.circles,test.circles,type = "response") > 0.5
```

```
preds.tree <- predict(tree.circles,test.circles, type="class")
preds.svm <- predict(svmfit.circles,test.circles, type="class")

g1 <- ggplot(test.circles, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("True Classes") +
  theme(legend.position = "none")
g2 <- ggplot(test.circles, aes(x1,x2,colour=preds.logreg)) +
  geom_point() +
  ggtitle("Logreg Preds") +
  theme(legend.position = "none")
g3 <- ggplot(test.circles, aes(x1,x2,colour=preds.tree)) +
  geom_point() +
  ggtitle("Decision Tree Preds") +
  theme(legend.position = "none")
g4 <- ggplot(test.circles, aes(x1,x2,colour=preds.svm)) +
  geom_point() +
  ggtitle("SVM Preds") +
  theme(legend.position = "none")

grid.arrange(g1,g2,g3,g4,ncol=2)
```




```
Accuracy(preds.logreg, test.circles$y==1)
```

```
## [1] 0.468
```

```
ConfusionMatrix(preds.tree, test.circles$y)
```

```
##       y_pred
## y_true   0   1
##      0 222  28
##      1  16 234
```

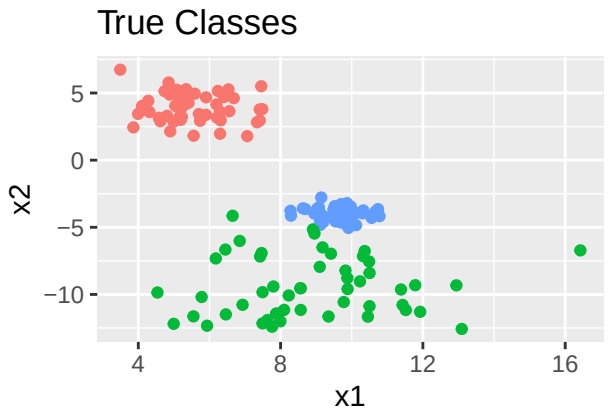
```
table(predict=preds.svm,actual=(test.circles$y==1))
```

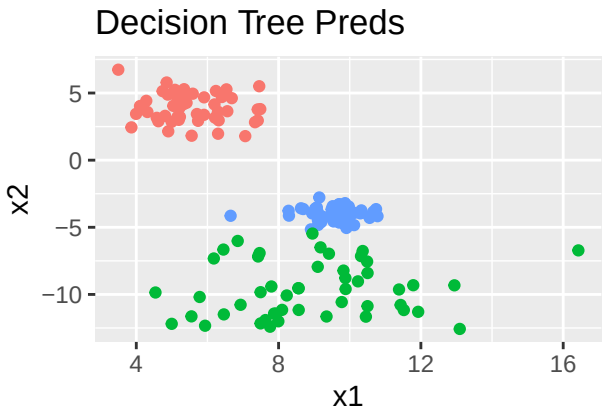
```
##        actual
## predict FALSE TRUE
##       0   244    6
##       1     6  244
```

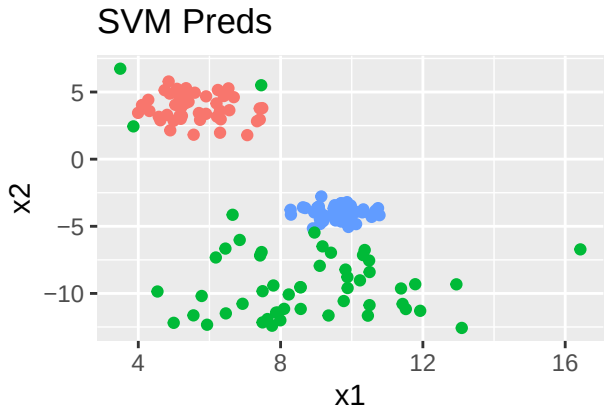
Based on the results, the SVM model produced the highest accuracy.

Varied

```
tree.varied <- tree(y~x1+x2, data=train.varied)
svmfit.varied <-  svm(y ~ x1+x2, data=train.varied , kernel ="radial",
                      cost =1, gamma=1,scale =FALSE)

preds.tree <- predict(tree.varied,test.varied, type="class")
preds.svm <- predict(svmfit.varied,test.varied, type="class")

g1 <- ggplot(test.varied, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("True Classes") +
  theme(legend.position = "none")
g2 <- ggplot(test.varied, aes(x1,x2,colour=preds.tree)) +
  geom_point() +
  ggtitle("Decision Tree Preds") +
  theme(legend.position = "none")
g3 <- ggplot(test.varied, aes(x1,x2,colour=preds.svm)) +
  geom_point() +
  ggtitle("SVM Preds") +
  theme(legend.position = "none")

grid.arrange(g1,g2,g3,ncol=2)
```








```
ConfusionMatrix(preds.tree, test.varied$y)
```

```
##        y_pred
## y_true  0  1  2
##      0 51  0  0
##      1  0 49  2
##      2  0  0 48
```

```
table(predict=preds.svm,actual=(test.varied$y==1))
```

```
##         actual
## predict FALSE TRUE
##       0    48    0
##       1     3   50
##       2    48    1
```

Based on the results, the Decision tree model produced the highest accuracy.

1.4. Cross Validation

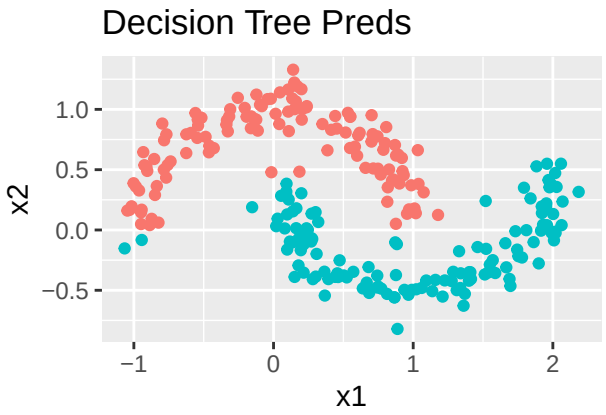
Croissant

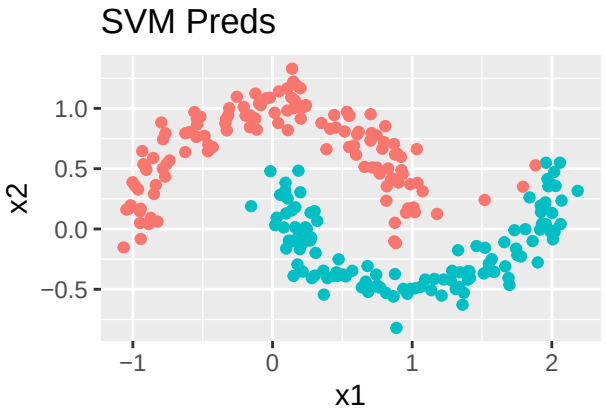
```
train_control <- trainControl(method = "cv", number = 10)
logreg.croissant <- train(y ~ x1+x2, data= train.croissant,
                          trControl = train_control,method = "glm",
                          family=binomial())
```

```
tree.croissant <- rpart(y~x1+x2, data=train.croissant)

svmfit.croissant <-  tune(svm ,y ~ x1+x2,data=train.croissant ,
                          kernel ="radial",scale =FALSE,
                          ranges =list(cost=c(0.01, 0.05, .1 ,1 ,10 ,100 ,1000),
                                       gamma=c(0.5,1,2,3,4)))

preds.logreg <- predict(logreg.croissant,test.croissant,type = "prob") > 0.5
preds.tree <- predict(tree.croissant,test.croissant, type="class")
preds.svm <- predict(svmfit.croissant$best.model,test.croissant, type="class")

g1 <- ggplot(test.croissant, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("True Classes") +
  theme(legend.position = "none")

summary(preds.logreg)
```

```
##      0               1
##  Mode :logical   Mode :logical
##  FALSE:126       FALSE:124
##  TRUE :124       TRUE :126
```

```
g3 <- ggplot(test.croissant, aes(x1,x2,colour=preds.tree)) +
  geom_point() +
  ggtitle("Decision Tree Preds") +
  theme(legend.position = "none")
g4 <- ggplot(test.croissant, aes(x1,x2,colour=preds.svm)) +
  geom_point() +
  ggtitle("SVM Preds") +
  theme(legend.position = "none")

grid.arrange(g1,g3,g4,ncol=2)
```






```
Accuracy(preds.logreg, test.croissant$y==1)
```

```
## [1] 0.5
```

```
ConfusionMatrix(preds.tree, test.croissant$y)
```

```
##        y_pred
## y_true   0   1
##      0 120   4
##      1   2 124
```

```
table(predict=preds.svm,actual=(test.croissant$y==1))
```

```
##         actual
## predict FALSE TRUE
##       0   124    3
##       1     0  123
```

Based on the results, the SVM model produced the highest accuracy. But, it is only slightly better than Decision tree.

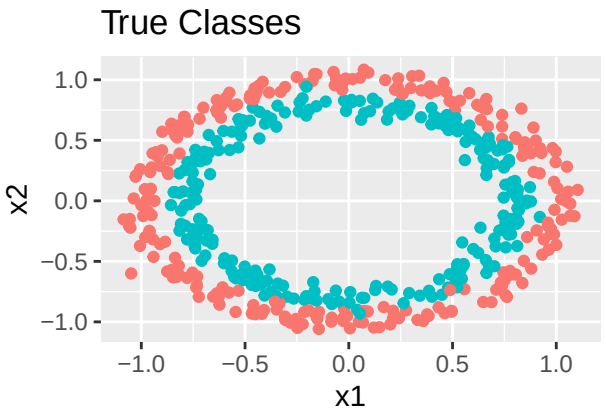
Circles

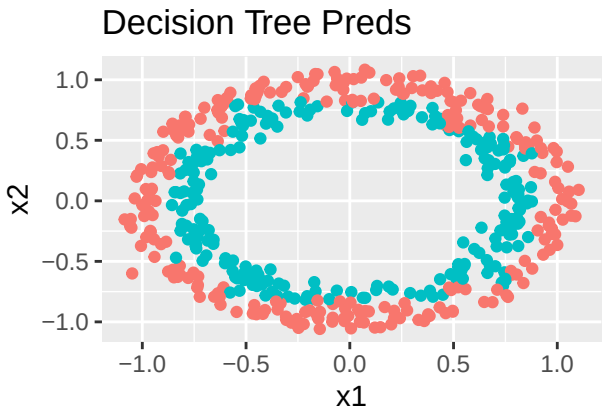
```
train_control <- trainControl(method = "cv", number = 10)
logreg.circles <- train(y ~ x1+x2, data= train.circles,
                        trControl = train_control,method = "glm",
                        family=binomial())
```

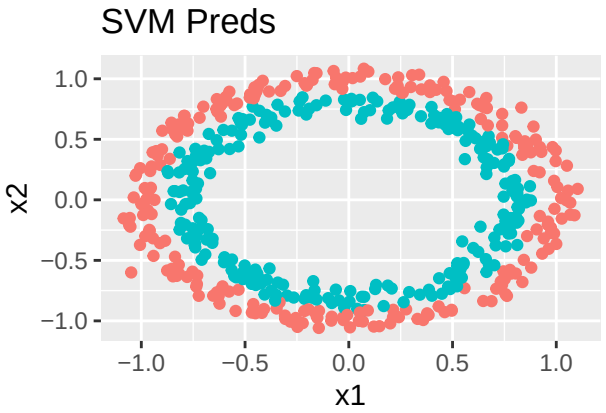
```
tree.circles <- rpart(y~x1+x2, data=train.circles)

svmfit.circles <-  tune(svm ,y ~ x1+x2,data=train.circles ,kernel ="radial",
                        scale =FALSE,
                        ranges =list(cost=c(0.01, 0.05, .1 ,1 ,10 ,100 ,1000),
                                     gamma=c(0.5,1,2,3,4)))

preds.logreg <- predict(logreg.circles,test.circles,type = "prob") > 0.5
preds.tree <- predict(tree.circles,test.circles, type="class")
preds.svm <- predict(svmfit.circles$best.model,test.circles, type="class")

g1 <- ggplot(test.circles, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("True Classes") +
  theme(legend.position = "none")

summary(preds.logreg)
```

```
##      0               1
##  Mode :logical   Mode :logical
##  FALSE:258       FALSE:242
##  TRUE :242       TRUE :258
```

```
g3 <- ggplot(test.circles, aes(x1,x2,colour=preds.tree)) +
  geom_point() +
  ggtitle("Decision Tree Preds") +
  theme(legend.position = "none")
g4 <- ggplot(test.circles, aes(x1,x2,colour=preds.svm)) +
  geom_point() +
  ggtitle("SVM Preds") +
  theme(legend.position = "none")


grid.arrange(g1,g3,g4,ncol=2)
```








```
Accuracy(preds.logreg, test.circles$y==1)
```

```
## [1] 0.5
```

```
ConfusionMatrix(preds.tree, test.circles$y)
```

```
##       y_pred
## y_true   0   1
##      0 235  15
##      1  32 218
```

```
table(predict=preds.svm,actual=(test.circles$y==1))
```

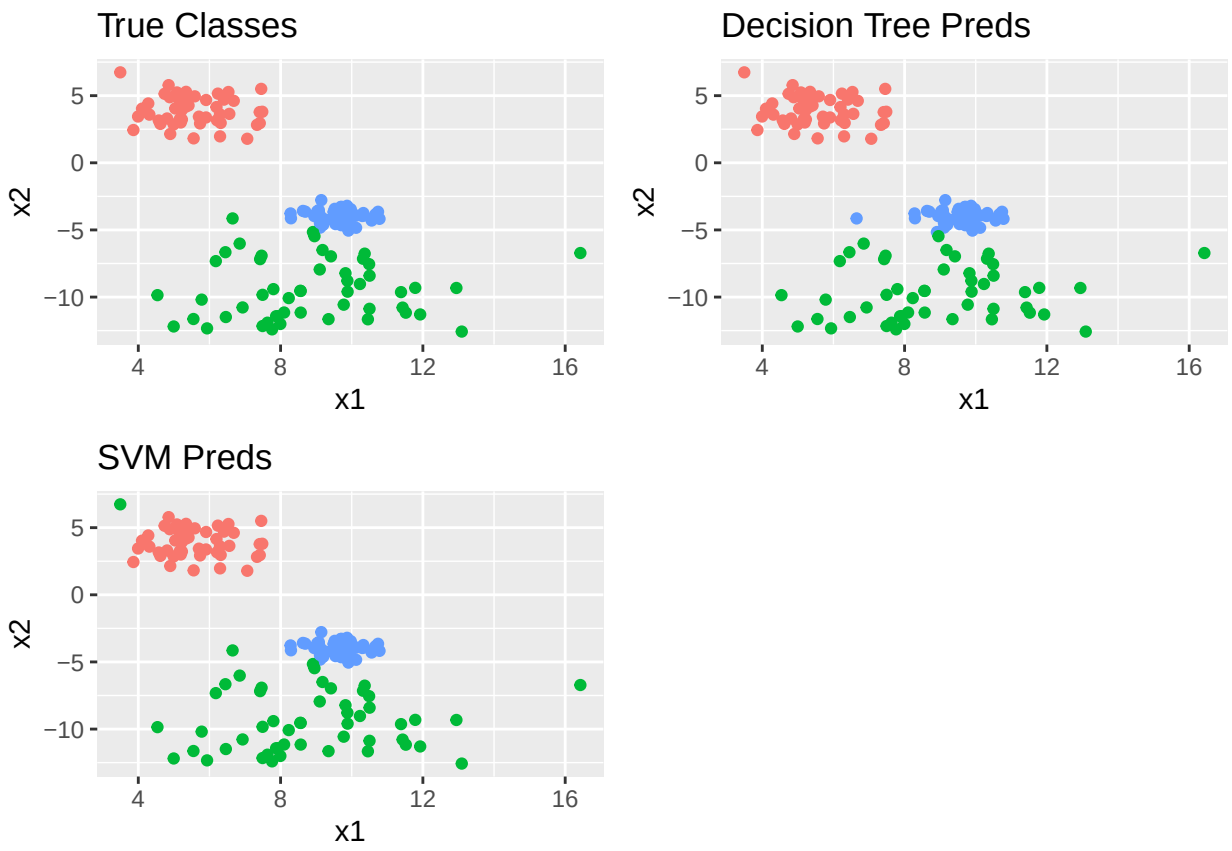
```
##         actual
## predict FALSE TRUE
##       0   242    4
##       1     8  246
```

Based on the results, the SVM model produced the highest accuracy.

Varied

```
tree.varied <- rpart(y~x1+x2, data=train.varied)

svmfit.varied <-  tune(svm ,y ~ x1+x2,data=train.varied ,kernel ="radial",
                       scale =FALSE,
                       ranges =list(cost=c(0.01, 0.05, .1 ,1 ,10 ,100 ,1000),
                                    gamma=c(0.5,1,2,3,4)))
```

```
preds.tree <- predict(tree.varied,test.varied, type="class")
preds.svm <- predict(svmfit.varied$best.model,test.varied, type="class")

g1 <- ggplot(test.varied, aes(x1,x2,colour=y)) +
  geom_point() +
  ggtitle("True Classes") +
  theme(legend.position = "none")
g2 <- ggplot(test.varied, aes(x1,x2,colour=preds.tree)) +
  geom_point() +
  ggtitle("Decision Tree Preds") +
  theme(legend.position = "none")
g3 <- ggplot(test.varied, aes(x1,x2,colour=preds.svm)) +
  geom_point() +
  ggtitle("SVM Preds") +
  theme(legend.position = "none")

grid.arrange(g1,g2,g3,ncol=2)
```




```
ConfusionMatrix(preds.tree, test.varied$y)
```

```
##        y_pred
## y_true  0  1  2
##      0 51  0  0
##      1  0 49  2
##      2  0  0 48
```

```
table(predict=preds.svm,actual=(test.varied$y==1))
```

```
##         actual
## predict FALSE TRUE
##       0    50    0
##       1     1   51
##       2    48    0
```

Based on the results, the decision tree produced the highest accuracy.

Question 2: Tree-based methods

2.1 Preprocess

This step involves splitting both the Heart and Hitters dataset into training and testing sets. Furthermore, the salaries within the Hitters dataset has been transformed to the logs of themselves. Lastly, the row identifier of the Heart dataset has been taken out for the sake of simplicity in prediction.

The Hitters dataset contains some missing data denoted by "NA". In order to compensate for such discrepancies and for simplicity's sake, the data has been manipulated to omit such missing information and only work with rows with full data columns.

```
setwd("~/446Assignments/Assignment3")
heart <- read.csv("Heart.csv")[,-1]

Hitters <- na.omit(Hitters)
Hitters$Salary <- log10(Hitters$Salary)

## 70% of the sample size
smp_size_hitters <- floor(0.70 * nrow(Hitters))
smp_size_heart <- floor(0.70 * nrow(heart))

## set the seed to make your partition reproducible
set.seed(112)

## Split the Hitters data to 70/30
train_ind_hitters <- sample(seq_len(nrow(Hitters)), size = smp_size_hitters)
train_hitters <- Hitters[train_ind_hitters, ]
test_hitters <- Hitters[-train_ind_hitters, ]
cat("hitters train size:", nrow(train_hitters), "hitters test size:", nrow(test_hitters))
```

```
## hitters train size: 184 hitters test size: 79
```

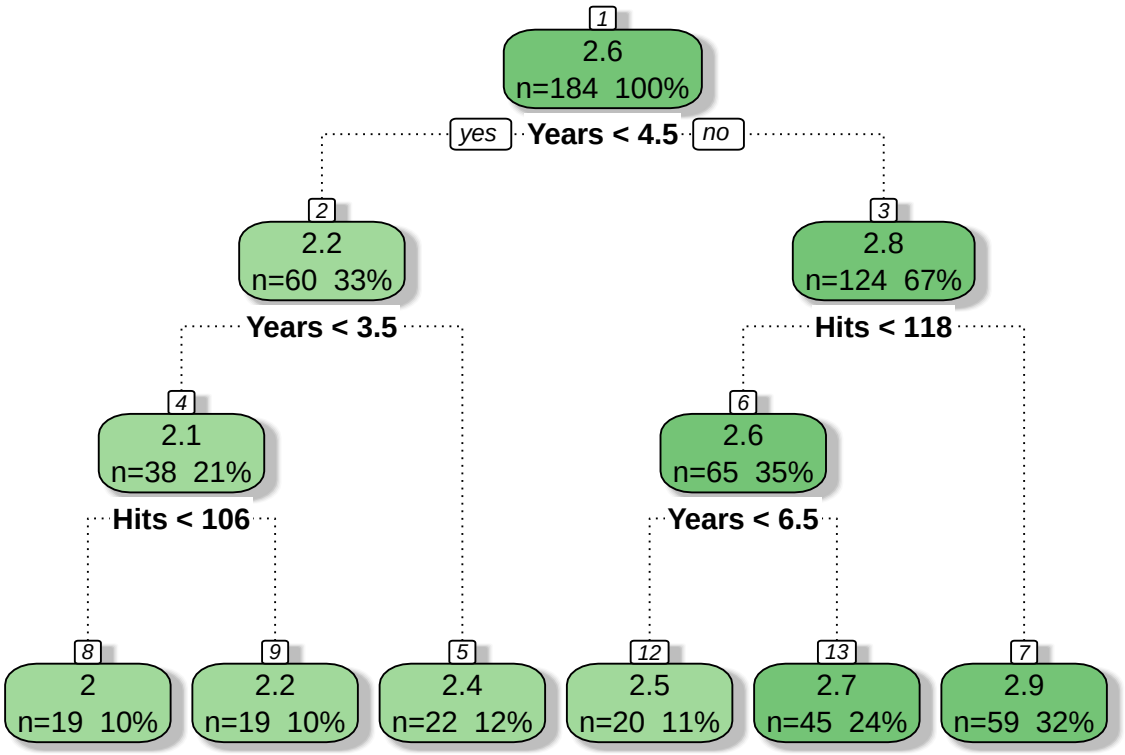
```
## Split the heart data to 70/30
train_ind_heart <- sample(seq_len(nrow(heart)), size = smp_size_heart)
train_heart <- heart[train_ind_heart, ]
test_heart <- heart[-train_ind_heart, ]
cat("hitters train size:", nrow(train_heart), "hitters test size:", nrow(test_heart))
```

```
## hitters train size: 212 hitters test size: 91
```

2.2 Decision Trees for Regression

Fit decision tree of Hitters dataset, where salary is a function of Hits and Years

```
hitters.tree <- rpart(Salary ~ Hits + Years, data=train_hitters)
fancyRpartPlot(hitters.tree, caption="")
```



Based on the decision tree generated, the salary of a player who's been in the league for 6 seasons and had 125 hits can be found by looking at node 7 of the Figure above. Transforming the output back to the actual salary, 10^2.9 gives $794.33.

```
preds <- predict(hitters.tree, test_hitters)
```

Log salary of every hitter in the test set:

```
preds
```

```
##     -Andre Dawson -Andres Galarraga    -Andres Thomas     -Alex Trevino
##          2.924384          1.960581          1.960581          2.682756
##       -Bill Almon      -Bruce Bochy      -Barry Bonds       -Bob Brenly
##          2.682756          2.682756          1.960581          2.471433
##      -Bob Dernier    -Billy Hatcher     -Bill Madlock   -Bill Schroeder
##          2.682756          2.244209          2.682756          2.443473
##     -Carlton Fisk  -Candy Maldonado   -Curt Wilkerson    -Dave Anderson
##          2.682756          2.471433          2.443473          2.443473
##     -Daryl Boston    -Doug DeCinces    -Darrell Evans     -Dwight Evans
##          1.960581          2.924384          2.924384          2.924384
##    -Damaso Garcia      -Davey Lopes   -Darrell Porter   -Dick Schofield
##          2.924384          2.682756          2.682756          2.443473
##       -Dale Sveum  -Danny Tartabull     -Eddie Murray      -Frank White
##          1.960581          2.244209          2.924384          2.924384
##     -George Brett      -Glenn Davis       -Garth Iorg      -Gary Pettis
##          2.924384          2.244209          2.682756          2.924384
##        -Hal McRae  -Harold Reynolds    -Harry Spilman       -Jody Davis
```

```
##          2.682756          2.443473          2.682756          2.924384
##        -John Kruk       -Johnny Ray         -Jim Rice     -John Russell
##          1.960581          2.924384          2.924384          1.960581
##       -Kevin Bass      -Kal Daniels      -Kirk Gibson      -Ken Griffey
##          2.924384          1.960581          2.924384          2.924384
##  -Keith Hernandez    -Ken Landreaux   -Kevin Mitchell    -Ken Oberkfell
##          2.924384          2.682756          1.960581          2.924384
##    -Kirby Puckett     -Larry Sheets     -Lou Whitaker       -Mike Davis
##          2.244209          1.960581          2.924384          2.924384
##   -Mariano Duncan      -Mike Easler     -Mike Kingery       -Mark Salas
##          1.960581          2.924384          1.960581          1.960581
##     -Mike Schmidt    -Milt Thompson     -Ozzie Virgil      -Phil Garner
##          1.960581          1.960581          2.682756          2.682756
##  -Rafael Belliard       -Ron Kittle     -Rick Manning  -Rance Mulliniks
##          2.471433          2.471433          2.682756          2.682756
##     -Rey Quinones   -Rafael Ramirez     -Ron Roenicke        -Rick Schu
##          1.960581          2.924384          2.471433          1.960581
##      -Robin Yount    -Scott Bradley       -Spike Owen    -Terry Kennedy
##          2.924384          1.960581          2.443473          1.960581
##     -Tito Landrum      -Tim Laudner     -Tom Paciorek        -Vance Law
##          2.682756          2.471433          2.682756          2.682756
##     -Wally Joyner   -Wayne Tolleson    -Willie Upshaw
##          2.244209          2.924384          2.924384
```

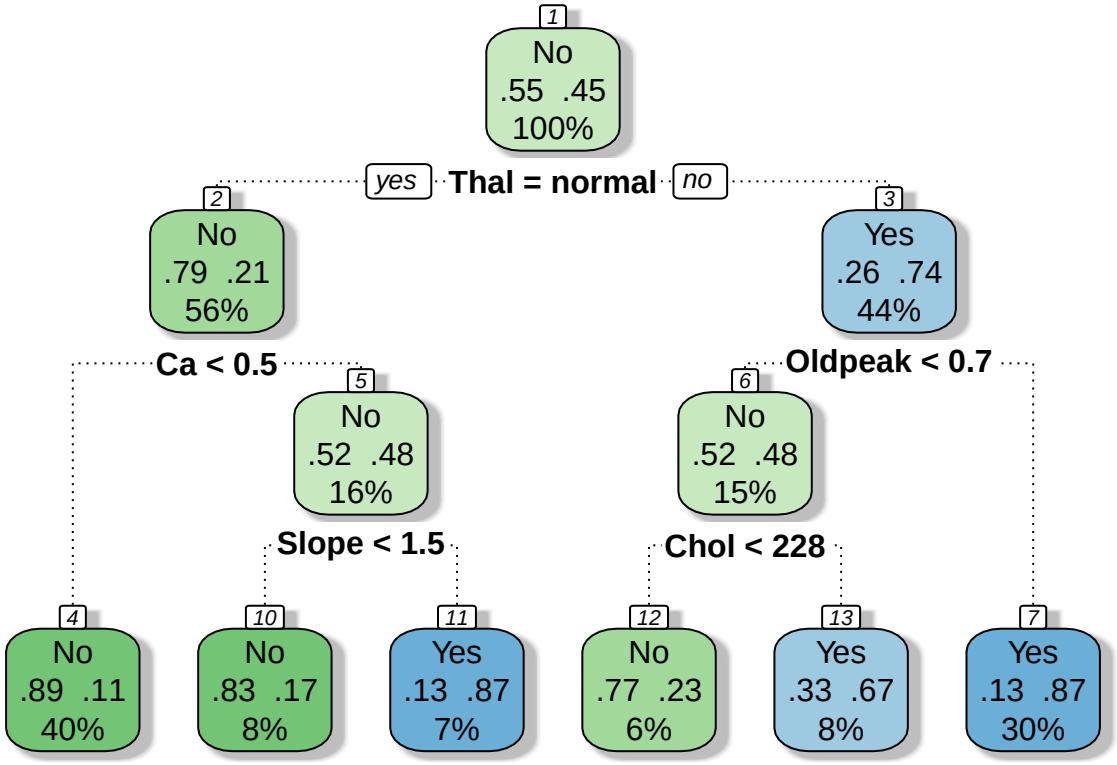
```
RMSE <- RMSE(preds,test_hitters$Salary)
SSE <- (RMSE^2)*nrow(test_hitters)
cat("Total Sum Squared Error (SSE) for the regression model:", SSE)
```

```
## Total Sum Squared Error (SSE) for the regression model: 6.216793
```

2.3. Decision Trees for Classification

Classification decision tree is generated on the Heart training set, by mapping AHD to all predictors as follows:

```
heart.tree <- rpart(AHD ~ ., data=train_heart)
fancyRpartPlot(heart.tree, caption="")
```




```
preds_heart <- predict(heart.tree, test_heart, type = "class")
```

The prediction accuracy for the first 5 patients in the test set using "predict" can be found as:

```
Accuracy(preds_heart[1:5],test_heart$AHD[1:5])
```

```
## [1] 0.8
```

The confusion matrix for the resulted classification tree:

```
ConfusionMatrix(preds_heart, test_heart$AHD)
```

```
##        y_pred
## y_true No Yes
##    No  38   9
##    Yes 12  32
```

2.4. Bagging: Regression

Fitting bagging model to Hitters training dataset, the 4 of the player salaries are predicted:

```
## Only include data that's all filled
train_hitters.bag <- train_hitters[complete.cases(train_hitters), ]
hitters.bag <- randomForest(Salary ~ . , data = train_hitters.bag, mtry = ncol(train_hitters.bag)-
preds.bag <- predict(hitters.bag, test_hitters)
cat("The predicted log salaries of the first 4 players are: ", head(preds.bag, 4))
```

```
## The predicted log salaries of the first 4 players are:  2.904626 2.010606 2.025359 2.61838
```

```
RMSE.bag <- RMSE(preds.bag, test_hitters$Salary)
SSE.bag <- (RMSE.bag^2)*nrow(test_hitters)
cat("Total Sum Squared Error (SSE) for the bagging model:", SSE.bag)
```

```
## Total Sum Squared Error (SSE) for the bagging model: 4.789401
```

The SSE of the bagging model is lower than that of the regression tree.

2.5. Bagging: Classification

Fitting the bagging model to Heart training dataset by mapping AHD to all predictors

```
train_heart.bag <- train_heart[complete.cases(train_heart), ]
train_heart.bag$AHD <- factor(train_heart.bag$AHD)
heart.bag <- randomForest(AHD ~ . , data = train_heart.bag)
preds_heart.bag <- predict(heart.bag, test_heart, type = "class")
cat("The predicted results for the first 4: ", head(preds_heart.bag, 4))
```

```
## The predicted results for the first 4:  1 1 1 2
```

The confusion matrix for the resulted classification bagging model:

```
ConfusionMatrix(preds_heart.bag, test_heart$AHD)
```

```
##        y_pred
## y_true No Yes
##    No  41   5
##    Yes 11  33
```

The prediction accuracy for the first 5 patients in the test set using "predict" can be found as:

```
Accuracy(preds_heart.bag[1:5],test_heart$AHD[1:5])
```

```
## [1] 1
```

The accuracy seems to be much better for the bagging model. This is due to the fact that bagging model uses cross validation on decision trees by fitting different trees and averaging the predictions.

2.6. Random Forest: Regression

Fitting random forest model to Hitters data by mapping salary to all predictors:

```
train_hitters_2.6 <- na.omit(train_hitters)
hitters.forest <- randomForest(Salary ~ . , data = train_hitters_2.6,mtry = 6, importance=T)
preds.forest <- predict(hitters.forest, test_hitters)
cat("The predicted log salaries of the first 4 players are: ", head(preds.bag, 4))
```

```
## The predicted log salaries of the first 4 players are:  2.904626 2.010606 2.025359 2.61838
```

```
RMSE.forest <- RMSE(preds.forest, test_hitters$Salary)
SSE.forest <- (RMSE.forest^2)*nrow(test_hitters)
cat("Total Sum Squared Error (SSE) for the random forest model:", SSE.forest)
```

```
## Total Sum Squared Error (SSE) for the random forest model: 4.484519
```

The SSE for the random forest model is slightly lower than the bagging model.