MSCI 446 Assignment 3 - Question 1

M. Harper, H. Gomaa, K. Morris

29/03/2021

Include Packages

```
library('tidyverse')
library('caret')
library('ggplot2')
library('gridExtra')
library('AmesHousing')
library('plotly')
library('ISLR')
library('glmnet')
library('leaps')

library('tree')
library('rpart')
library('e1071')

theme_set(theme_classic())
```

Question 1: Classification

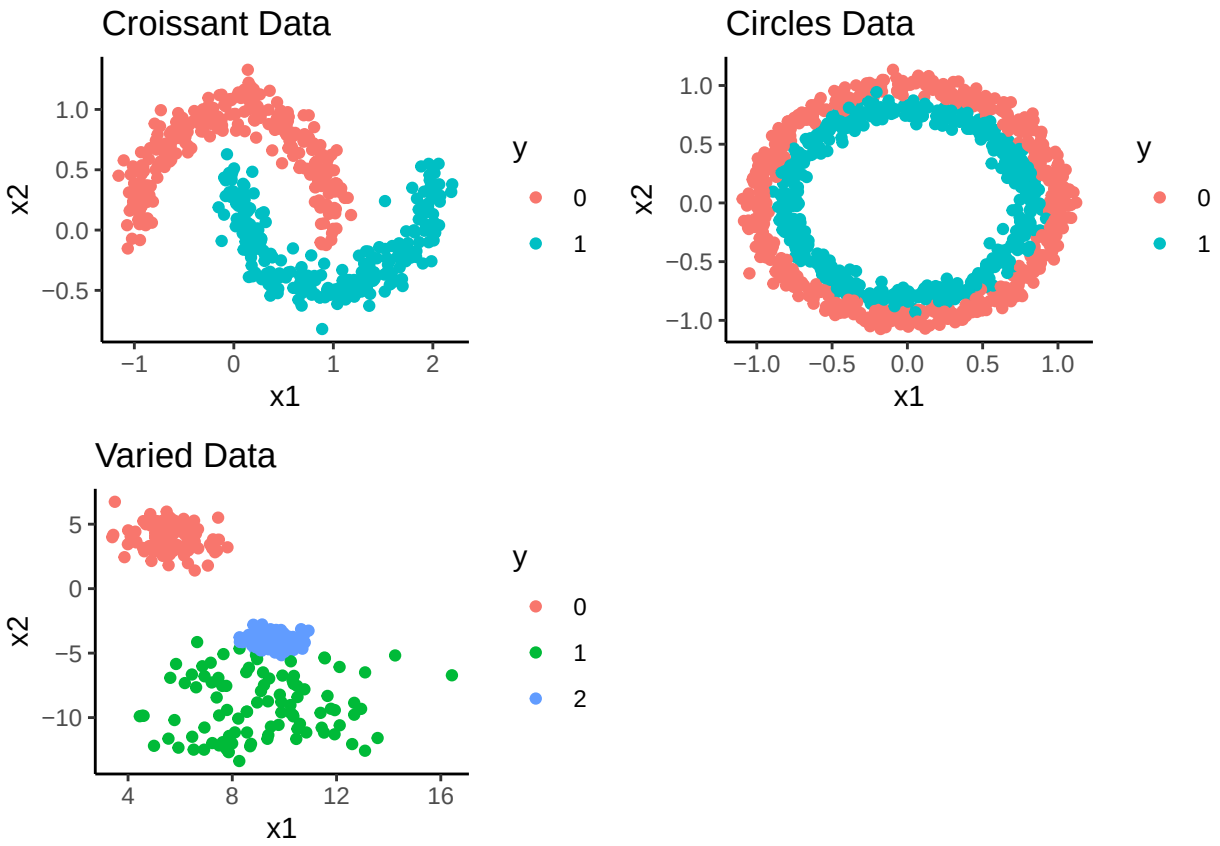
Import Data:

```
croissants <- read.csv('croissant.csv')
circles <- read.csv('circles.csv')
varied <- read.csv('varied.csv')

str(circles)
```

```
## 'data.frame':    1000 obs. of  4 variables:
##  $ X : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ x1: num  -0.394 0.975 0.359 0.762 0.512 ...
##  $ x2: num  -0.873 -0.23 -0.993 -0.794 0.749 ...
##  $ y : int  0 0 0 0 0 1 1 1 1 1 ...
```

1.1 Preprocess and Plot

```
croissants$y <- as.factor(croissants$y)
circles$y <- as.factor(circles$y)
varied$y <- as.factor(varied$y)

g1 <- ggplot(data=croissants)+
  geom_point(aes(x=x1, y=x2, colour=y))+
  ggtitle("Croissant Data")

g2 <- ggplot(data=circles)+
  geom_point(aes(x=x1, y=x2, colour=y))+
  ggtitle("Circles Data")

g3 <- ggplot(data=varied)+
  geom_point(aes(x=x1, y=x2, colour=y))+
  ggtitle("Varied Data")

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



I personally love the airiness of croissants, and for that reason the *Croissant Data* looks the most eatable. Donuts are great, but one of those is enough. With Croissants, they are so light they feel healthy to eat and therefore I eat more of those.

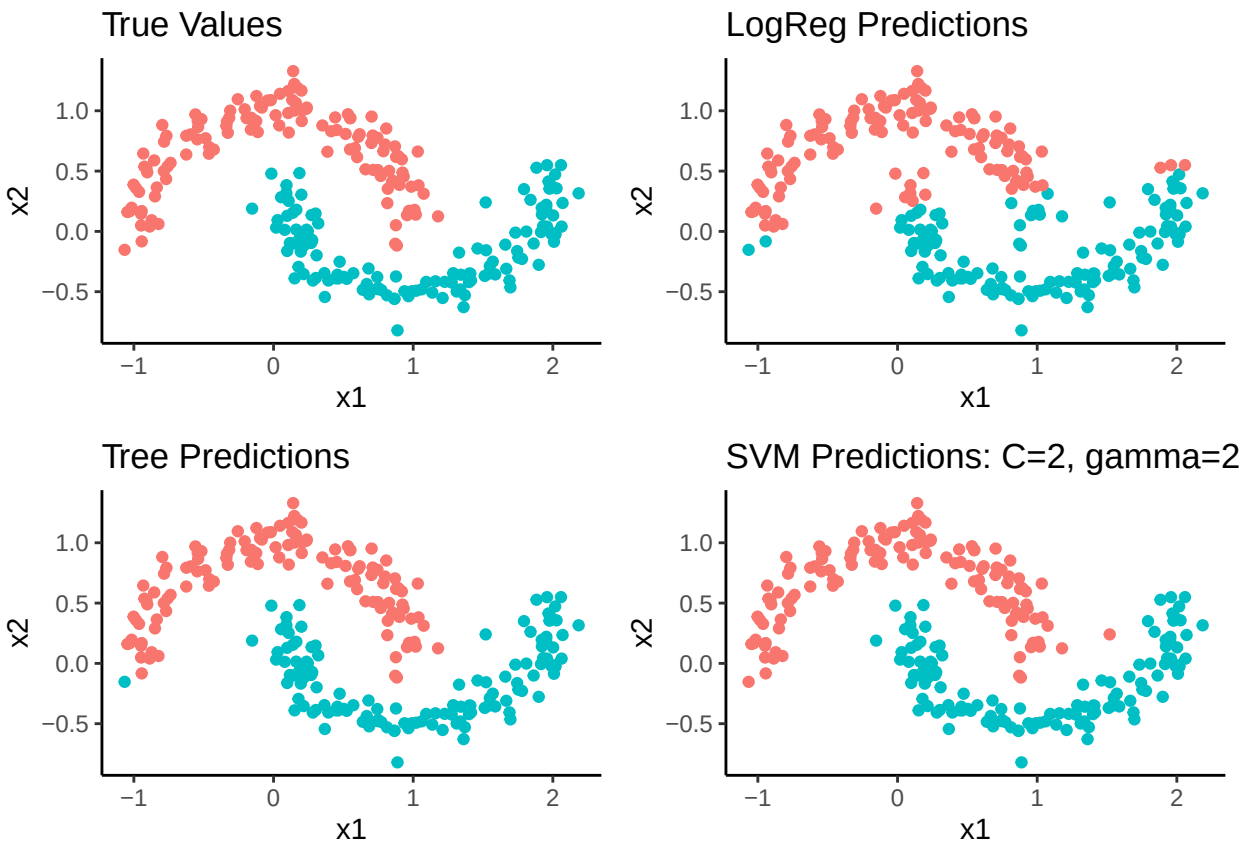
1.2 Train / Test Split

```
set.seed(112)

train_inds1 <- sample(1:nrow(croissants), floor(0.5*nrow(croissants)))
train_inds2 <- sample(1:nrow(circles), floor(0.5*nrow(circles)))
train_inds3 <- sample(1:nrow(varied), floor(0.5*nrow(varied)))

train.croissants <- croissants[train_inds1, ]
test.croissants <- croissants[-train_inds1, ]

train.circles <- circles[train_inds2, ]
test.circles <- circles[-train_inds2, ]

train.varied <- varied[train_inds3, ]
test.varied <- varied[-train_inds3, ]
```

1.3 Train and Test

```
#1. Train models on croissant data set

croissant.logreg <- glm(data=train.croissants, y ~ x1 + x2 ,family='binomial')
croissant.tree <- tree(data=train.croissants, y ~ x1 + x2)
croissant.svm <- svm(data=train.croissants, y ~ x1 + x2,
                     kernel='radial', cost=2, gamma=2, scale = FALSE)

#2. Predict the test set and plot performance

probs.logreg <- predict(croissant.logreg, newdata = test.croissants,
                        type='response')
preds.logreg <- ifelse(probs.logreg >=0.5, 1, 0)
preds.logreg <- as.factor(preds.logreg)

preds.tree <- predict(croissant.tree, newdata = test.croissants, type='class')
preds.svm <- predict(croissant.svm, newdata = test.croissants)


g4 <- ggplot(data=test.croissants)+
  geom_point(aes(x=x1, y=x2, colour=y))+
  ggtitle("True Values")+
  theme(legend.position = "none")
g5 <- ggplot(data=test.croissants)+
  geom_point(aes(x=x1, y=x2, colour=preds.logreg))+
  ggtitle("LogReg Predictions")+
  theme(legend.position = "none")
g6 <- ggplot(data=test.croissants)+
  geom_point(aes(x=x1, y=x2, colour=preds.tree))+
  ggtitle("Tree Predictions")+
  theme(legend.position = "none")
g7 <- ggplot(data=test.croissants)+
  geom_point(aes(x=x1, y=x2, colour=preds.svm))+
```

```
  ggtitle("SVM Predictions: C=2, gamma=2")+
  theme(legend.position = "none")

grid.arrange(g4, g5, g6, g7, ncol=2)
```



Reasoning for Radial kernel Selection: From observation of the croissant data set plot, it is clear that the data cannot be split linearly or with a polynomial function. For this reason, we will use a radial kernel so that the machine can have the flexibility to warp it's grouping boundaries around the data set. For example, when referencing the plots from *question 1.1*, it appears as though the classifications can be loosely represented by either a positive or negative quadratic function, which the use of a radial kernel will allow the flexibility to create such shapes.

Parameter Selection for SVM (cost and gamma): The cost and gamma parameters for the SVM model were chosen to be 2. This is done fairly arbitrarily. The code above was ran multiple times with cost and gamma parameters starting at 0.1. The plot was observed, and the model was ran again while increasing one or both of the parameters. It was found that increasing the cost and gamma parameters past a value of 2 appears to have negligible affects on the accuracy of the model predictions.

Comparing the Four Plots: Visually comparing the Four plots above conclude that Logistic Regression performed the worst when classifying the data. The plots also show that Tree based methods and SVM methods both accurately classify the data. It appears as though SVM out-performs Tree methods for this data. This is evident when comparing the classification of the data at the tips of both of the croissants, where it is evident that SVM better classifies the data.

```r
#3. Evaluate test performance of each model (Croissant Data)

croissant.logreg.acc <- mean(preds.logreg == test.croissants$y)*100
croissant.tree.acc <- mean(preds.tree == test.croissants$y)*100
croissant.svm.acc <- mean(preds.svm == test.croissants$y)*100

accuracies <- c(croissant.logreg.acc, croissant.tree.acc, croissant.svm.acc)
models <- c('Logistic Regression', 'Tree Method', 'SVM')
data.frame(Models=models, Test_Accuracy_Percent=accuracies)
```

```
##                Models Test_Accuracy_Percent
## 1 Logistic Regression                  90.4
## 2         Tree Method                  99.6
## 3                 SVM                  99.6
```

```r
croissant.logreg.cm <- confusionMatrix(preds.logreg, test.croissants$y)
croissant.tree.cm <- confusionMatrix(preds.tree, test.croissants$y)
croissant.svm.cm <- confusionMatrix(preds.svm, test.croissants$y)

croissant.logreg.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 112  12
##          1  12 114
```

```r
croissant.tree.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 123   0
##          1   1 126
```

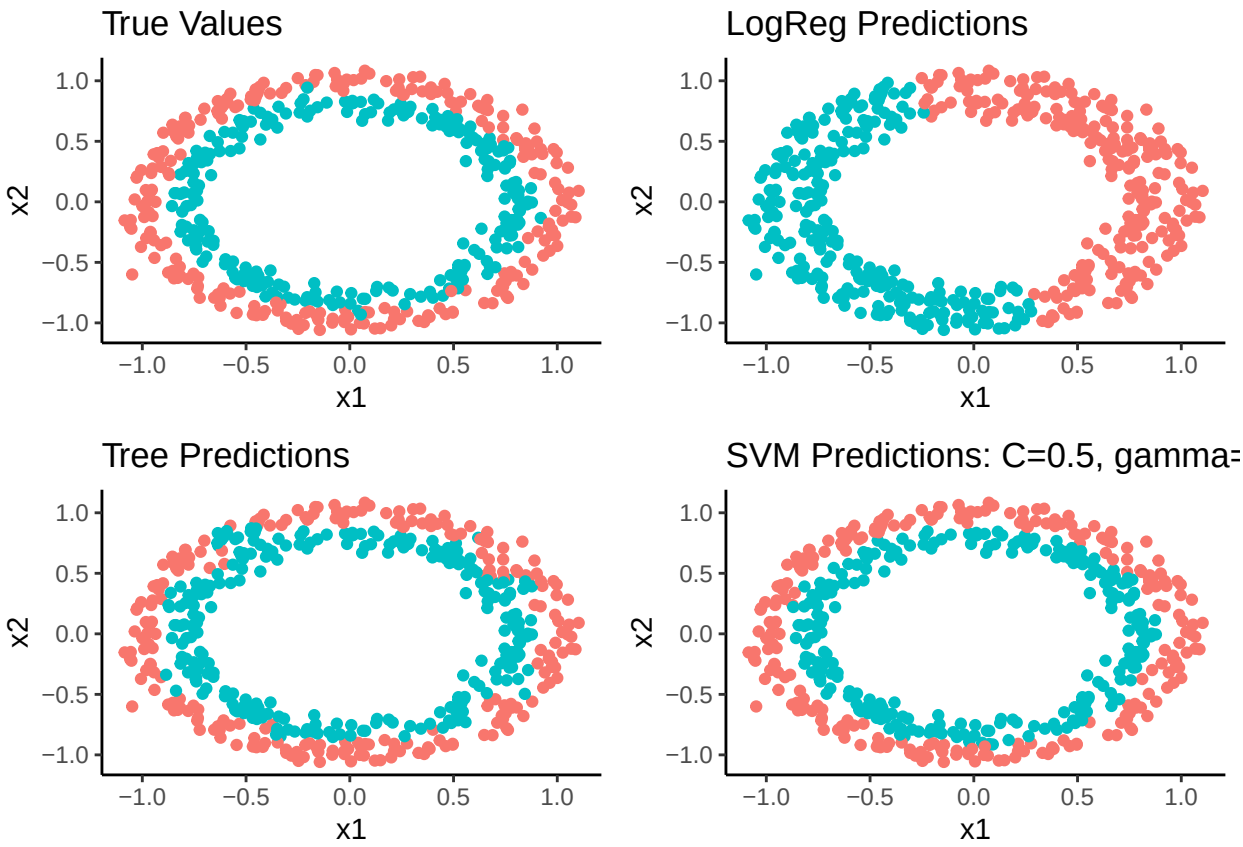
```r
croissant.svm.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 124   1
##          1   0 125
```

It appears as though the SVM and tree models had the highest prediction accuracy of 99.6%. Followed by Logistic Regression methods with accuracy's of 90.4%. Furthermore, The SVM and tree models appear to be slightly biased towards a false negative, and the tree model appears to be slightly biased towards a false positive. The logestic regression model appears to be the least biased in missclasifying the data.

Repeat All of the Above for the Circles Data Set

```
set.seed(112)
#1. Train models on Circles data set

circles.logreg <- glm(data=train.circles, y ~ x1 + x2 ,family='binomial')
circles.tree <- tree(data=train.circles, y ~ x1 + x2)
circles.svm <- svm(data=train.circles, y ~ x1 + x2, kernel='radial', cost=0.5,
                   gamma=0.5, scale = FALSE)

#2. Predict the test set and plot performance

probs.logreg <- predict(circles.logreg, newdata = test.circles, type='response')
preds.logreg <- ifelse(probs.logreg >=0.5, 1, 0)
preds.logreg <- as.factor(preds.logreg)

preds.tree <- predict(circles.tree, newdata = test.circles, type='class')

preds.svm <- predict(circles.svm, newdata = test.circles)


g8 <- ggplot(data=test.circles)+
  geom_point(aes(x=x1, y=x2, colour=y))+
  ggtitle("True Values")+
  theme(legend.position = "none")

g9 <- ggplot(data=test.circles)+
  geom_point(aes(x=x1, y=x2, colour=preds.logreg))+
  ggtitle("LogReg Predictions")+
  theme(legend.position = "none")

g10 <- ggplot(data=test.circles)+
  geom_point(aes(x=x1, y=x2, colour=preds.tree))+
  ggtitle("Tree Predictions")+
  theme(legend.position = "none")

g11 <- ggplot(data=test.circles)+
  geom_point(aes(x=x1, y=x2, colour=preds.svm))+
  ggtitle("SVM Predictions: C=0.5, gamma=0.5")+
  theme(legend.position = "none")


grid.arrange(g8, g9, g10, g11, ncol=2)
```



Reasoning for Radial kernel Selection: Perhaps for a more obvious reason than with the croissant data set, for the circles data set it is predicted that the use of a radial kernel will best represent the data. The data visually cannot be split with any form of polynomial, linear or sigmoidial line.

Parameter Selection for SVM (cost and gamma): A similar approach to the croissant data was used to find good performing values of cost and gamma parameters. However, it was found that a better performing model was found using considerably smaller cost and gamma parameter for the circles data than with the croissant data.

Comparing the Four Plots: Visually comparing the plots shows the poor performance of the Logistic Regression model, classifying less than half of the data points correctly. Again, Tree and SVM models are comparable, however from visual inspection it still appears as though SVM outperforms the tree model.

```
set.seed(112)
#3. Evaluate test performance of each model (Circles Data)

circles.logreg.acc <- mean(preds.logreg == test.circles$y)*100
circles.tree.acc <- mean(preds.tree == test.circles$y)*100
circles.svm.acc <- mean(preds.svm == test.circles$y)*100

accuracies <- c(circles.logreg.acc, circles.tree.acc, circles.svm.acc)
models <- c('Logistic Regression', 'Tree Method', 'SVM')
data.frame(Models=models, Test_Accuracy_Percent=accuracies)
```

```
##                Models Test_Accuracy_Percent
## 1 Logistic Regression                  46.8
## 2         Tree Method                  90.6
## 3                 SVM                  97.4
```

```
circles.logreg.cm <- confusionMatrix(preds.logreg, test.circles$y)
circles.tree.cm <- confusionMatrix(preds.tree, test.circles$y)
circles.svm.cm <- confusionMatrix(preds.svm, test.circles$y)

circles.logreg.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 113 129
##          1 137 121
```

```
circles.tree.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 219  16
##          1  31 234
```

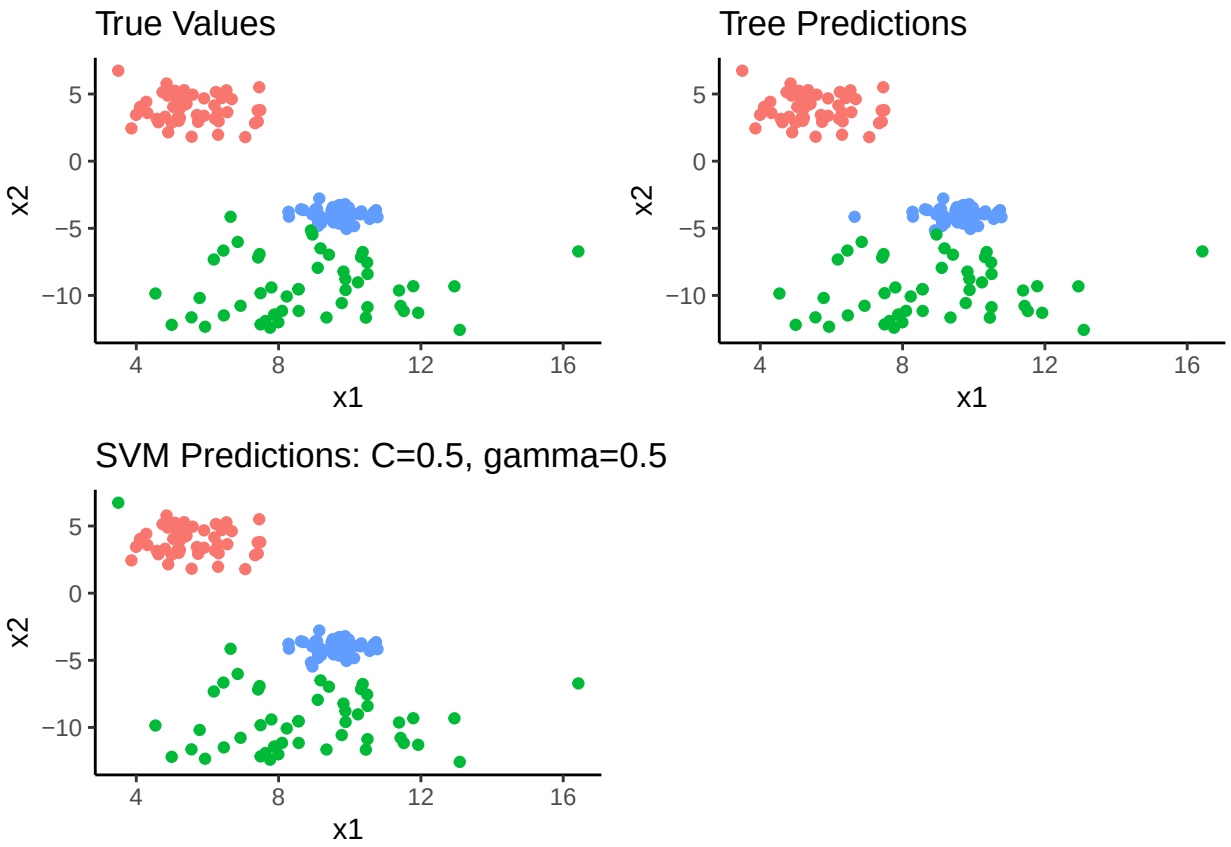
```
circles.svm.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 241   4
##          1   9 246
```

It appears as though the SVM model has the highest prediction accuracy of 97.4%. Followed by Tree and Logistic Regression methods with accuracy's of 91.2% and 46.8% respectively. Furthermore, The SVM model also appears to be the least bias of the models, with a total of 13 miss-classifications (9 false positives and 4 false negatives). The logistic regression model appears to be slightly more biased with a 8 point differential in miss-classifications (129 false negatives and 137 false positives.) Lastly, the tree model appears to be the most biased with 16 false negatives and 31 false positives

Repeat All of the Above for the Varied Data Set (Excluding Logistic Regression)

```
#1. Train models on Varied data set

varied.tree <- tree(data=train.varied, y ~ x1 + x2)
varied.svm <- svm(data=train.varied, y ~ x1 + x2, kernel='radial', cost=0.2,
                  gamma=0.2, scale = FALSE)

#2. Predict the test set and plot performance

preds.tree <- predict(varied.tree, newdata = test.varied, type='class')
preds.svm <- predict(varied.svm, newdata = test.varied)


g12 <- ggplot(data=test.varied)+
  geom_point(aes(x=x1, y=x2, colour=y))+
  ggtitle("True Values")+
  theme(legend.position = "none")

g13 <- ggplot(data=test.varied)+
  geom_point(aes(x=x1, y=x2, colour=preds.tree))+
  ggtitle("Tree Predictions")+
  theme(legend.position = "none")

g14 <- ggplot(data=test.varied)+
  geom_point(aes(x=x1, y=x2, colour=preds.svm))+
  ggtitle("SVM Predictions: C=0.5, gamma=0.5")+
  theme(legend.position = "none")


grid.arrange(g12, g13, g14, ncol=2)
```



SVM Kernel and Parameter Selection: The kernel was chosen as radial because the plots visually appears as though circles could be drawn around the different clouds of data classifications. The SVM cost and gamma parameters were chosen in a manner identical to the previous two data sets.

Comparing the Three Plots: Upon inspection, It appears as though the tree and SVM models identically classify with the exception of the top left-most data point and the rightmost data point.

```
#3. Evaluate test performance of each model (Circles Data)

varied.tree.acc <- mean(preds.tree == test.varied$y)*100
varied.svm.acc <- mean(preds.svm == test.varied$y)*100

accuracies <- c(varied.tree.acc, varied.svm.acc)
models <- c('Tree Method', 'SVM')
data.frame(Models=models, Test_Accuracy_Percent=accuracies)
```

```
##        Models Test_Accuracy_Percent
## 1 Tree Method              98.66667
## 2         SVM              98.00000
```

```
varied.tree.cm <- confusionMatrix(preds.tree, test.varied$y)
varied.svm.cm <- confusionMatrix(preds.svm, test.varied$y)

varied.tree.cm$table
```

```
##           Reference
```

```
## Prediction  0  1  2
##          0 51  0  0
##          1  0 49  0
##          2  0  2 48
```

```
varied.svm.cm$table
```

```
##           Reference
## Prediction  0  1  2
##          0 50  0  0
##          1  1 49  0
##          2  0  2 48
```

For the Varied data set, it appears as though the models are equally as accurate, however the Tree method is shown to be 0.667% more accurate, and very comparably biased.

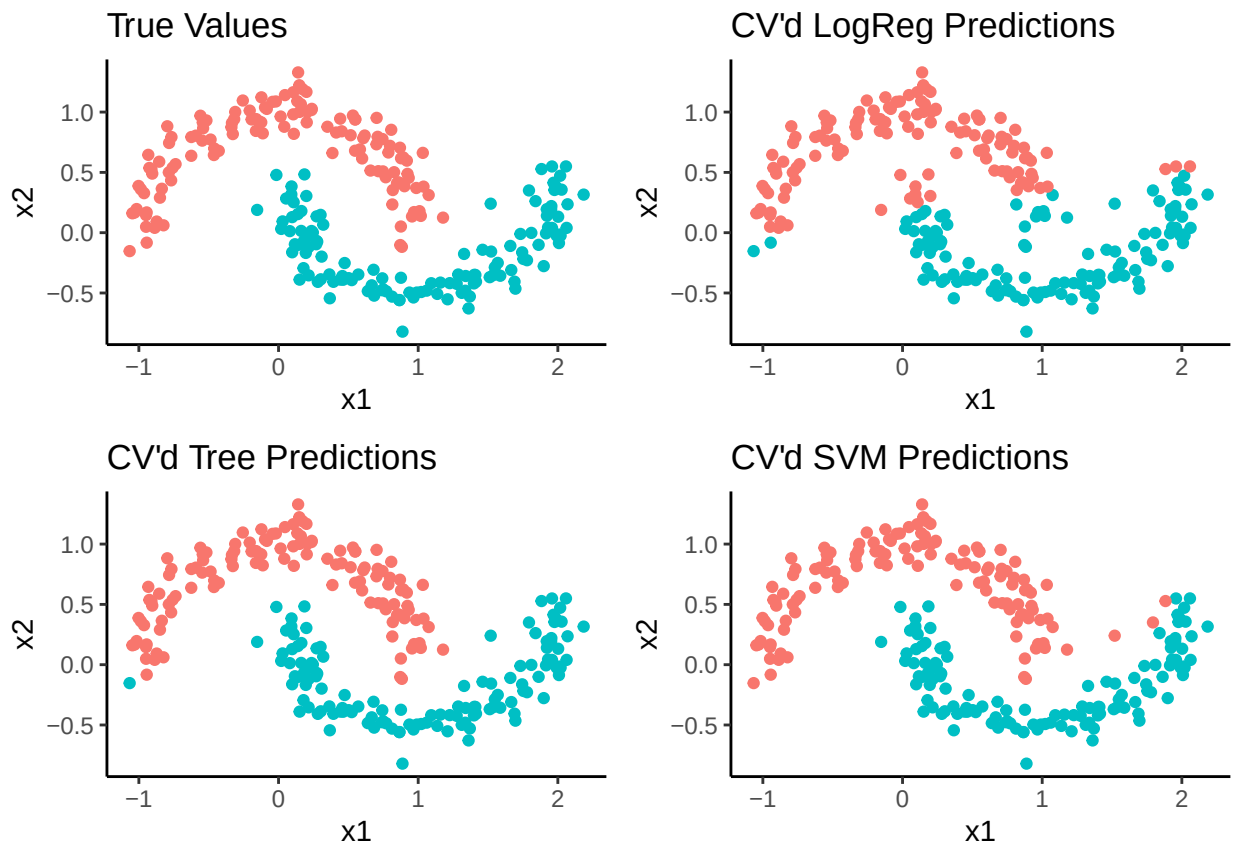
1.4 Cross Validation

The same reasoning for the selection of a kernel type for SVM from the models above also apply to the models below that are the same although have been modeled using cross-validation. However, for the cost and gamma parameters on the SVM model, they were tuned using cross-validation and not a guess and check method like previously.

Cross Validation on the Croissant Data Set

```
#Croissant Data Set
set.seed(112)
#1. Train models on croissant data set

ctrl <- trainControl(method='cv', number=10)
croissant.logreg.cv <- train(data=train.croissants, y~x1+x2, method='glm',
                             trControl=ctrl)

croissant.tree <- tree(data=train.croissants, y ~ x1 + x2)
croissant.tree.cv <- cv.tree(croissant.tree, FUN = prune.misclass)
croissant.tree.pruned <- prune.misclass(croissant.tree,
                best=croissant.tree.cv$size[which.min(croissant.tree.cv$dev)])

tune.out <- tune(svm, y~x1+x2, data=train.croissants, kernel='radial',
                 ranges = list(cost=c(0.1,0.5,1,10,100,1000),
                               gamma=c(0.1,0.3,0.5,0.7,1.0,2.0,3.0)))
croissant.svm.cv <- tune.out$best.model



#2. Predict the test set and plot performance

preds.logreg.cv <- predict(croissant.logreg.cv, newdata = test.croissants,
                           type='raw')
preds.tree.cv <- predict(croissant.tree.pruned, newdata = test.croissants,
                         type='class')
preds.svm.cv <- predict(croissant.svm.cv, newdata = test.croissants)

g15 <- ggplot(data=test.croissants)+
  geom_point(aes(x=x1, y=x2, colour=y))+
  ggtitle("True Values")+
  theme(legend.position = "none")
g16 <- ggplot(data=test.croissants)+
  geom_point(aes(x=x1, y=x2, colour=preds.logreg.cv))+
  ggtitle("CV'd LogReg Predictions")+
  theme(legend.position = "none")
g17 <- ggplot(data=test.croissants)+
  geom_point(aes(x=x1, y=x2, colour=preds.tree.cv))+
  ggtitle("CV'd Tree Predictions")+
  theme(legend.position = "none")
g18 <- ggplot(data=test.croissants)+
  geom_point(aes(x=x1, y=x2, colour=preds.svm.cv))+
  ggtitle("CV'd SVM Predictions")+
```

```
  theme(legend.position = "none")

grid.arrange(g15, g16, g17, g18, ncol=2)
```




```
#3. Evaluate test performance of each model (Croissant Data)

croissant.logreg.cv.acc <- mean(preds.logreg.cv == test.croissants$y)*100
croissant.tree.cv.acc <- mean(preds.tree.cv == test.croissants$y)*100
croissant.svm.cv.acc <- mean(preds.svm.cv == test.croissants$y)*100

accuracies <- c(croissant.logreg.cv.acc, croissant.tree.cv.acc,
                croissant.svm.cv.acc)
models <- c('Logistic Regression (CV)', 'Tree Method(CV)', 'SVM(CV)')
data.frame(Models=models, Test_Accuracy_Percent=accuracies)
```

```
##                    Models Test_Accuracy_Percent
## 1 Logistic Regression (CV)                 90.4
## 2          Tree Method(CV)                 99.6
## 3                  SVM(CV)                 98.8
```

```
croissant.logreg.cm <- confusionMatrix(preds.logreg.cv, test.croissants$y)
croissant.tree.cm <- confusionMatrix(preds.tree.cv, test.croissants$y)
croissant.svm.cm <- confusionMatrix(preds.svm.cv, test.croissants$y)

croissant.logreg.cm$table
```

```
##            Reference
## Prediction   0   1
##          0 112  12
##          1  12 114
```

```
croissant.tree.cm$table
```

```
##            Reference
## Prediction   0   1
##          0 123   0
##          1   1 126
```

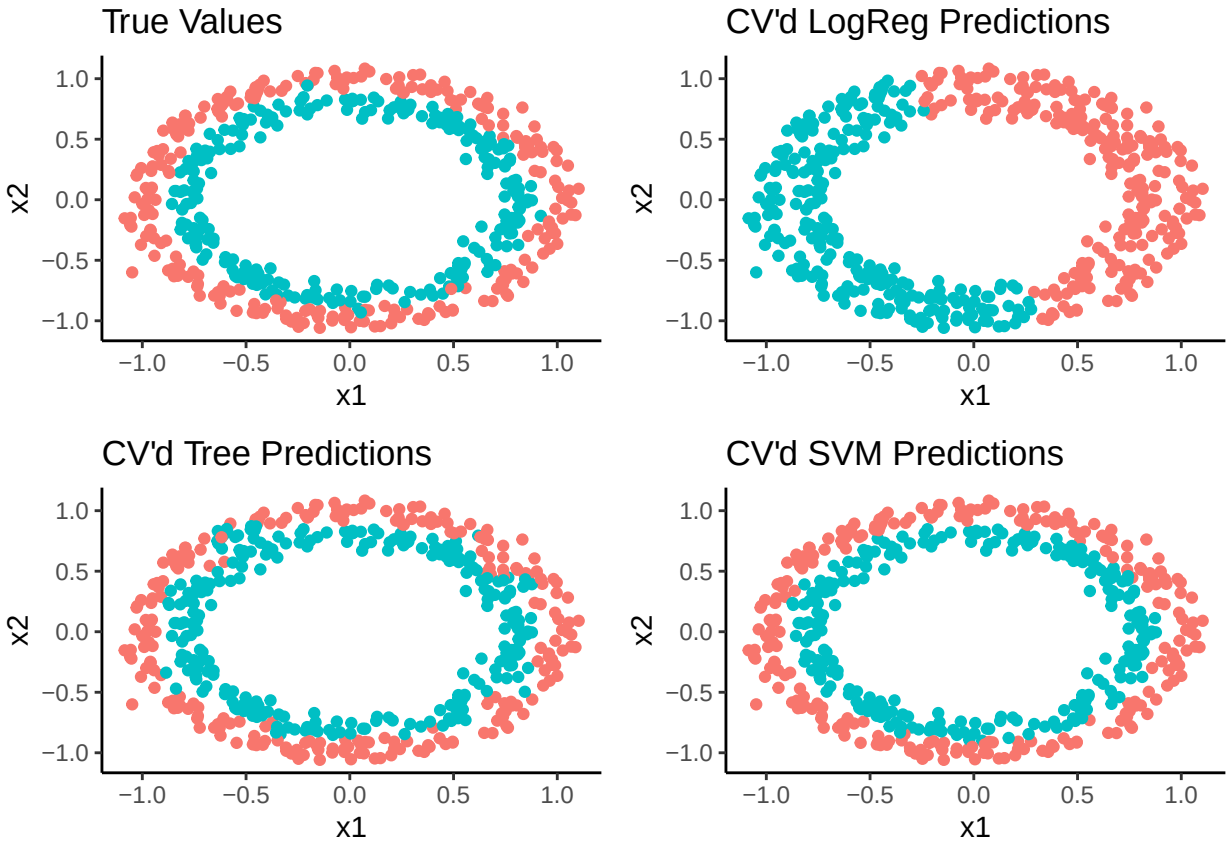
```
croissant.svm.cm$table
```

```
##            Reference
## Prediction   0   1
##          0 124   3
##          1   0 123
```

Cross Validation of the Circles Data set

```
#Circles Data Set
set.seed(112)
#1. Train models on croissant data set

ctrl <- trainControl(method='cv', number=10)
circles.logreg.cv <- train(data=train.circles, y~x1+x2, method='glm',
                           trControl=ctrl)

circles.tree <- tree(data=train.circles, y ~ x1 + x2)
circles.tree.cv <- cv.tree(circles.tree, FUN = prune.misclass)
circles.tree.pruned <- prune.misclass(circles.tree,
                    best=circles.tree.cv$size[which.min(circles.tree.cv$dev)])

tune.out <- tune(svm, y~x1+x2, data=train.circles, kernel='radial',
                 ranges = list(cost=c(0.1,0.5,1,10,100,1000),
                               gamma=c(0.1,0.3,0.5,0.7,1.0,2.0,3.0)))
circles.svm.cv <- tune.out$best.model


#2. Predict the test set and plot performance

preds.logreg.cv <- predict(circles.logreg.cv, newdata = test.circles,
                           type='raw')
preds.tree.cv <- predict(circles.tree.pruned, newdata = test.circles,
                         type='class')
preds.svm.cv <- predict(circles.svm.cv, newdata = test.circles)


g19 <- ggplot(data=test.circles)+
  geom_point(aes(x=x1, y=x2, colour=y))+
```

```
  ggtitle("True Values")+
  theme(legend.position = "none")
g20 <- ggplot(data=test.circles)+
  geom_point(aes(x=x1, y=x2, colour=preds.logreg.cv))+
  ggtitle("CV'd LogReg Predictions")+
  theme(legend.position = "none")
g21 <- ggplot(data=test.circles)+
  geom_point(aes(x=x1, y=x2, colour=preds.tree.cv))+
  ggtitle("CV'd Tree Predictions")+
  theme(legend.position = "none")
g22 <- ggplot(data=test.circles)+
  geom_point(aes(x=x1, y=x2, colour=preds.svm.cv))+
  ggtitle("CV'd SVM Predictions")+
  theme(legend.position = "none")

grid.arrange(g19, g20, g21, g22, ncol=2)
```




```
#3. Evaluate test performance of each model (Croissant Data)

circles.logreg.cv.acc <- mean(preds.logreg.cv == test.circles$y)*100
circles.tree.cv.acc <- mean(preds.tree.cv == test.circles$y)*100
circles.svm.cv.acc <- mean(preds.svm.cv == test.circles$y)*100

accuracies <- c(circles.logreg.cv.acc, circles.tree.cv.acc,
                circles.svm.cv.acc)
```

```
models <- c('Logistic Regression (CV)', 'Tree Method(CV)', 'SVM(CV)')
data.frame(Models=models, Test_Accuracy_Percent=accuracies)
```

```
##                    Models Test_Accuracy_Percent
## 1 Logistic Regression (CV)                 46.8
## 2          Tree Method(CV)                 90.6
## 3                  SVM(CV)                 97.8
```

```
circles.logreg.cm <- confusionMatrix(preds.logreg.cv, test.circles$y)
circles.tree.cm <- confusionMatrix(preds.tree.cv, test.circles$y)
circles.svm.cm <- confusionMatrix(preds.svm.cm, test.circles$y)

circles.logreg.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 113 129
##          1 137 121
```

```
circles.tree.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 219  16
##          1  31 234
```

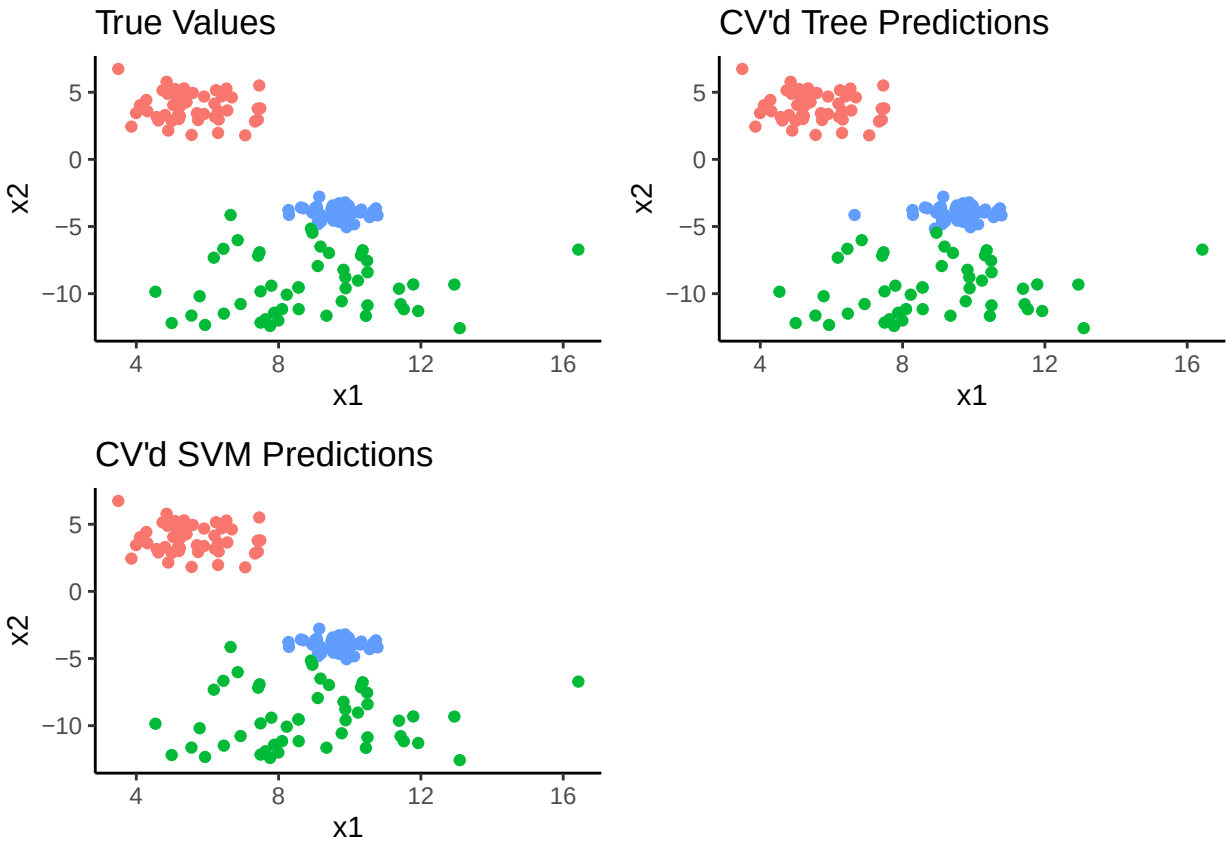
```
circles.svm.cm$table
```

```
##           Reference
## Prediction   0   1
##          0 243   4
##          1   7 246
```

Cross Validation on the Varied Data Set (Excluding Logistic Regression)

```
#Varied Data Set
set.seed(112)
#1. Train models on Varied data set

varied.tree <- tree(data=train.varied, y ~ x1 + x2)
varied.tree.cv <- cv.tree(varied.tree, FUN = prune.misclass)
varied.tree.pruned <- prune.misclass(varied.tree,
                     best=varied.tree.cv$size[which.min(varied.tree.cv$dev)])

tune.out <- tune(svm, y~x1+x2, data=train.varied, kernel='radial',
                 ranges = list(cost=c(0.1,0.5,1,10,100,1000),
                               gamma=c(0.1,0.3,0.5,0.7,1.0,2.0,3.0)))
varied.svm.cv <- tune.out$best.model
```

```
#2. Predict the test set and plot performance

preds.tree.cv <- predict(varied.tree.pruned, newdata = test.varied,
                         type='class')
preds.svm.cv <- predict(varied.svm.cv, newdata = test.varied)


g23 <- ggplot(data=test.varied)+
  geom_point(aes(x=x1, y=x2, colour=y))+
  ggtitle("True Values")+
  theme(legend.position = "none")
g24 <- ggplot(data=test.varied)+
  geom_point(aes(x=x1, y=x2, colour=preds.tree.cv))+
  ggtitle("CV'd Tree Predictions")+
  theme(legend.position = "none")
g25 <- ggplot(data=test.varied)+
  geom_point(aes(x=x1, y=x2, colour=preds.svm.cv))+
  ggtitle("CV'd SVM Predictions")+
  theme(legend.position = "none")

grid.arrange(g23, g24, g25, ncol=2)
```




```
#3. Evaluate test performance of each model (Croissant Data)

varied.tree.cv.acc <- mean(preds.tree.cv == test.varied$y)*100
```

```
varied.svm.cv.acc <- mean(preds.svm.cv == test.varied$y)*100

accuracies <- c(varied.tree.cv.acc, varied.svm.cv.acc)
models <- c('Tree Method(CV)', 'SVM(CV)')
data.frame(Models=models, Test_Accuracy_Percent=accuracies)
```

```
##            Models Test_Accuracy_Percent
## 1 Tree Method(CV)              98.66667
## 2         SVM(CV)             100.00000
```

```
varied.tree.cm <- confusionMatrix(preds.tree.cv, test.varied$y)
varied.svm.cm <- confusionMatrix(preds.svm.cv, test.varied$y)

varied.tree.cm$table
```

```
##           Reference
## Prediction  0  1  2
##          0 51  0  0
##          1  0 49  0
##          2  0  2 48
```

```
varied.svm.cm$table
```

```
##           Reference
## Prediction  0  1  2
##          0 51  0  0
##          1  0 51  0
##          2  0  0 48
```