Week 2 Code Examples, CSC 400, Spring 2024

1. Logistic Regression
2. Support Vector Machines
3. K-Nearest Neighbors
4. Visualization

Logistic Regression

```
library(dplyr)
summary(mtcars)
library(caTools)
library(ROCR)

split <- sample.split(mtcars, SplitRatio = 0.8)
split

train_reg <- subset(mtcars, split == "TRUE")
test_reg <- subset(mtcars, split == "FALSE")

logistic_model <- glm(vs ~ wt + disp,
                      data = train_reg,
                      family = "binomial")
logistic_model

summary(logistic_model)
```

This example is a two independent variable model with test and train split. We can predict the outcomes or the probabilities. The example shows the probabilities.

```
predict_reg <- predict(logistic_model,
                       test_reg, type = "response")
predict_reg
```

You can change the threshold manually if you need the cut-off to be something other than 50%. Just change the 0.5 in the code.

```
predict_reg <- ifelse(predict_reg >0.5, 1, 0)
```

We can assess model performance.

```
table(test_reg$vs, predict_reg)
missing_classerr <- mean(predict_reg != test_reg$vs)
print(paste('Accuracy =', 1 - missing_classerr))

ROCPred <- prediction(predict_reg, test_reg$vs)
ROCPer <- performance(ROCPred, measure = "tpr",
                      x.measure = "fpr")

auc <- performance(ROCPred, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

We can plot the curve.

```
plot(ROCPer)
plot(ROCPer, colorize = TRUE,
     print.cutoffs.at = seq(0.1, by = 0.1),
     main = "ROC CURVE")
abline(a = 0, b = 1)

auc <- round(auc, 4)
legend(.6, .4, auc, title = "AUC", cex = 1)
```

Support Vector Machines (SVM)

You can get the dataset for the example here: https://media.geeksforgeeks.org/wp-content/uploads/social.csv. Save it somewhere you can find it again, and import it from there.

```
dataset = read_csv("R/daemen/social.csv")
dataset = dataset[3:5]
dataset$Purchased = factor(dataset$Purchased, levels = c(0, 1))

library(caTools)

#set.seed(123) #set the seed for reproducibility
split = sample.split(dataset$Purchased, SplitRatio = 0.75)

training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)
```

SVM generally requires data to be rescales in the range -1 to 1. This code is rescaling all columns but the third one.

```
training_set[-3] = scale(training_set[-3])
test_set[-3] = scale(test_set[-3])
```

Create the classifier.

```
library(e1071)
classifier = svm(formula = Purchased ~ .,
                 data = training_set,
                 type = 'C-classification',
                 kernel = 'linear')
classifier
```

Next, we make our predictions and examine the confusion matrix.

```
y_pred = predict(classifier, newdata = test_set[-3])
y_org<-test_set$Purchased
cm = table(y_org, y_pred)
cm
```

Visualize the results. First the training set, and then also the test set.

```
library('Rfast')
set  = training_set
X1 = seq(min(set[, 1]) -1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) -1, max(set[, 2]) + 1, by = 0.01)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Age', 'EstimatedSalary')
prob_set = predict(classifier, type = 'response', newdata = grid_set)
y_grid = ifelse(prob_set==0, 1, 0)
plot(set[, -3],
     main = 'SVM (Training Set)',
     xlab = 'Age',
     ylab = 'Estimated Salary',
     xlim = range(X1),
     ylim = range(X2)
)
contour(X1, X2, matrix(as.numeric(y_grid),length(X1), length(X2)), add = TRUE)
points(grid_set, pch = '.', col = ifelse(y_grid==1, 'springgreen3', 'tomato') )
points(set, pch = 21, bg = ifelse(set[, 3]== 1, 'green4', 'red3'))
```

```
library('Rfast')
set  = test_set
X1 = seq(min(set[, 1]) -1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) -1, max(set[, 2]) + 1, by = 0.01)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Age', 'EstimatedSalary')
prob_set = predict(classifier, type = 'response', newdata = grid_set)
y_grid = ifelse(prob_set == 1, 1, 0)
plot(set[, -3],
     main = 'SVM (Test Set)',
     xlab = 'Age',
     ylab = 'Estimated Salary',
     xlim = range(X1),
     ylim = range(X2)
)

contour(X1,X2, matrix(as.numeric(y_grid),length(X1), length(X2)), add = TRUE)
points(grid_set, pch = '.', col = ifelse(y_grid==1, 'springgreen3', 'tomato') )
points(set, pch = 21, bg = ifelse(set[, 3]== 1, 'green4', 'red3'))
```

K-Nearest Neighbor

We'll use the iris dataset as our example.

```
df <- data(iris)
head(iris)
ran <- sample(1:nrow(iris), 0.9 * nrow(iris))
```

We need to normalize the dataset.

```
nor <-function(x) { (x -min(x))/(max(x)-min(x))   }
iris_norm <- as.data.frame(lapply(iris[,c(1,2,3,4)], nor))
summary(iris_norm)
```

Split into test and training data.

```
iris_train <- iris_norm[ran,]
iris_test <- iris_norm[-ran,]
iris_target_category <- iris[ran,5]
iris_test_category <- iris[-ran,5]
```

Now, build the model and test the accuracy on the test set.

```
library(class)
pr <- knn(iris_train,iris_test,cl=iris_target_category,k=13)
tab <- table(pr,iris_test_category)
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(tab)
```

Depending on the number of categories, it may be useful to plot the confusion matrix as a graph, such as:

```
heatmap(tab)
```

Resources:

1. https://www.geeksforgeeks.org/logistic-regression-in-r-programming/#
2. https://www.geeksforgeeks.org/classifying-data-using-support-vector-machinessvms-in-r/#
3. https://www.kaggle.com/code/aniketvishwakarma/alternative-of-elemstatlearn-for-visualisation
4. https://towardsdatascience.com/k-nearest-neighbors-algorithm-with-examples-in-r-simply-explained-knn-1f2c88da405c