Matthew Voynovich
Data Analytics
Professor Eleish
Lab 5

SVM with linear kernel

```
> # Test svm model 1
> test.pred <- predict(svm.mod1, test)
> cm = as.matrix(table(Actual = test$Type, Predicted = test.pred))
> cm
      Predicted
Actual  1  2  3
     1  8  1  0
     2  3 14  1
     3  0  0  9
> n = sum(cm) # number of instances
> nc = nrow(cm) # number of classes
> diag = diag(cm) # number of correctly classified instances per class
> rowsums = apply(cm, 1, sum) # number of instances per class
> colsums = apply(cm, 2, sum) # number of predictions per class
> p = rowsums / n # distribution of instances over the actual classes
> q = colsums / n # distribution of instances over the predicted
> accuracy <- sum(diag)/n
> accuracy
[1] 0.8611111
> recall = diag / rowsums
> precision = diag / colsums
> f1 = 2 * precision * recall / (precision + recall)
> data.frame(precision, recall, f1)
  precision    recall        f1
1 0.7272727 0.8888889 0.8000000
2 0.9333333 0.7777778 0.8484848
3 0.9000000 1.0000000 0.9473684
```

SVM with polynomial kernel

```
> #Test svm model 2
> test.pred <- predict(svm.mod2, test)
> cm = as.matrix(table(Actual = test$Type, Predicted = test.pred))
> cm
      Predicted
Actual  1  2  3
     1  8  1  0
     2  1 16  1
     3  0  0  9
> n = sum(cm) # number of instances
> nc = nrow(cm) # number of classes
> diag = diag(cm) # number of correctly classified instances per class
> rowsums = apply(cm, 1, sum) # number of instances per class
> colsums = apply(cm, 2, sum) # number of predictions per class
> p = rowsums / n # distribution of instances over the actual classes
> q = colsums / n # distribution of instances over the predicted
> accuracy <- sum(diag)/n
> accuracy
[1] 0.9166667
> recall = diag / rowsums
> precision = diag / colsums
> f1 = 2 * precision * recall / (precision + recall)
> data.frame(precision, recall, f1)
  precision    recall        f1
1 0.8888889 0.8888889 0.8888889
2 0.9411765 0.8888889 0.9142857
3 0.9000000 1.0000000 0.9473684
```

KNN with tuned k (best = 7)

```
> # Test knn
> test.pred <- predict(knn.mod, test)
> cm = as.matrix(table(Actual = test$Type, Predicted = test.pred))
> cm
      Predicted
Actual  1  2  3
     1  9  0  0
     2  1 16  1
     3  0  0  9
> n = sum(cm) # number of instances
> nc = nrow(cm) # number of classes
> diag = diag(cm) # number of correctly classified instances per class
> rowsums = apply(cm, 1, sum) # number of instances per class
> colsums = apply(cm, 2, sum) # number of predictions per class
> p = rowsums / n # distribution of instances over the actual classes
> q = colsums / n # distribution of instances over the predicted
> accuracy <- sum(diag)/n
> accuracy
[1] 0.9444444
> recall = diag / rowsums
> precision = diag / colsums
> f1 = 2 * precision * recall / (precision + recall)
> data.frame(precision, recall, f1)
  precision    recall        f1
1       0.9 1.0000000 0.9473684
2       1.0 0.8888889 0.9411765
3       0.9 1.0000000 0.9473684
```

Comparing models

The best model was KNN with a k = 7 as it has really good precision, recall, and f1 score across all classes. The worst model was the tuned SVM with linear kernel and was bad with predicting type 1. The tuned SVM with a polynomial kernel did very well despite having lower precision, recall, and f1 than the KNN with k = 7. The polynomial kernel SVM also still failed to predict type 1 as well as KNN. Thus the polynomial kernel performs better than the linear kernel but the best model out of svm and KNN for predicting wine type is KNN with a k=7.