Bruce Campbell Final Project

Fri Jul 29 10:40:04 2016

```r
rm(list = ls())
set.seed(7)

setwd("C:/st-617/")

## use read.csv2 since data fields are delimited by semicolons (;)

wine.data <- read.csv2("FinalProject/winequality-red.csv")

# Specifying the data type via colClasses did not work - so we sapply
# as.numeric to convert the factor data to numeric.  We may want to convert
# some of the variables to factors later - but for visualization of the raw
# features we should use numeric type.
wine.data[, c(1:12)] <- sapply(wine.data[, c(1:12)], as.numeric)

DF <- wine.data
DF <- DF[order(DF$quality), ]

train <- as.integer(unlist(read.csv("train.sample.csv", header = TRUE)))
```

Training with balanced test set

Multiclass Polynomial Subset Balanced Training

```r
DFCoarse <- DF[, c("fixed.acidity", "volatile.acidity", "chlorides", "total.sulfur.dioxide",
    "density", "sulphates", "alcohol")]

DFCoarse <- data.frame(scale(DF, center = TRUE, scale = TRUE))
DFCoarse$quality_cat <- factor(ifelse(wine.data$quality < 5, "LOW", ifelse(wine.data$quality >
    6, "HIGH", "MED")))
DFCoarse$quality <- NULL
DFCoarseTrain <- DFCoarse[train, ]
DFM <- DFCoarseTrain[DFCoarseTrain$quality_cat == "MED", ]
DFM <- DFM[1:200, ]
DFL <- DFCoarseTrain[DFCoarseTrain$quality_cat == "LOW", ]
DFH <- DFCoarseTrain[DFCoarseTrain$quality_cat == "HIGH", ]
DFCoarseTrain <- rbind(DFL, DFM, DFH)
DFCoarseTest <- DFCoarse[-train, ]

library(e1071)
library(pander)
gamma_default <- 1/(ncol(DFCoarseTrain) - 1)

gamma_list <- ((1:10)/5) * gamma_default
cost_list = c(0.001, 0.01, 0.1, 1, 5, 10, 100)
```

```r
tune.svm.quality_cat = tune(svm, quality_cat ~ ., data = DFCoarseTrain, kernel = "polynomial",
    ranges = list(gamma = gamma_list, cost = cost_list))
svm.fit.quality_cat <- tune.svm.quality_cat$best.model

svm.pred.quality_cat <- predict(svm.fit.quality_cat, DFCoarseTest)
TB <- table(svm.pred.quality_cat, DFCoarseTest$quality_cat)
pander(TB)
```

	HIGH	LOW	MED
HIGH	21	8	125
LOW	2	0	10
MED	39	12	316

```r
ACC_Multiclass_Polynomial_subset_balanced = (sum(diag(TB)))/length(DFCoarseTest$quality_cat)

ACC_Class_HIGH_Polynomial_subset_balanced = diag(TB)[1]/sum(DFCoarseTest$quality_cat ==
    "HIGH")

ACC_Class_LOW_Polynomial_subset_balanced = diag(TB)[2]/sum(DFCoarseTest$quality_cat ==
    "LOW")

ACC_Class_MED_Polynomial_subset_balanced = diag(TB)[3]/sum(DFCoarseTest$quality_cat ==
    "MED")

pander(data.frame(class = c("LOW", "MED", "HIGH"), accuracy = c(ACC_Class_LOW_Polynomial_subset_balance
    ACC_Class_MED_Polynomial_subset_balanced, ACC_Class_HIGH_Polynomial_subset_balanced)),
    caption = "Multiclass Polynomial Subset Balanced SVM accuracy by class")
```

Table 2: Multiclass Polynomial Subset Balanced SVM accuracy by class

	class	accuracy
LOW	LOW	0
MED	MED	0.7007
HIGH	HIGH	0.3387

Multiclass RBF Subset Balanced Training

```r
tune.svm.quality_cat = tune(svm, quality_cat ~ ., data = DFCoarseTrain, kernel = "polynomial",
    ranges = list(gamma = gamma_list, cost = cost_list))
svm.fit.quality_cat <- tune.svm.quality_cat$best.model

svm.pred.quality_cat <- predict(svm.fit.quality_cat, DFCoarseTest)
TB <- table(svm.pred.quality_cat, DFCoarseTest$quality_cat)
pander(TB)
```

	HIGH	LOW	MED
HIGH	20	7	117
LOW	4	1	14
MED	38	12	320

```r
ACC_Multiclass_RBF_subset_balanced = (sum(diag(TB)))/length(DFCoarseTest$quality_cat)

ACC_Class_HIGH_RBF_subset_balanced = diag(TB)[1]/sum(DFCoarseTest$quality_cat ==
    "HIGH")

ACC_Class_LOW_RBF_subset_balanced = diag(TB)[2]/sum(DFCoarseTest$quality_cat ==
    "LOW")

ACC_Class_MED_RBF_subset_balanced = diag(TB)[3]/sum(DFCoarseTest$quality_cat ==
    "MED")

pander(data.frame(class = c("LOW", "MED", "HIGH"), accuracy = c(ACC_Class_LOW_RBF_subset_balanced,
    ACC_Class_MED_RBF_subset_balanced, ACC_Class_HIGH_RBF_subset_balanced)),
    caption = "Multiclass RBF Subset Balanced SVM accuracy by class")
```

Table 4: Multiclass RBF Subset Balanced SVM accuracy by class

	class	accuracy
LOW	LOW	0.05
MED	MED	0.7095
HIGH	HIGH	0.3226

```r
method.accuracy <- data.frame(method = c("Multiclass RBF Subset Balanced SVM",
    "Multiclass Polynomial Subset Balanced SVM"), accuracy = as.numeric(c(ACC_Multiclass_RBF_subset_bala
    ACC_Multiclass_Polynomial_subset_balanced)))

method.accuracy <- method.accuracy[order(method.accuracy$accuracy, decreasing = FALSE),
    ]

DFA = method.accuracy
pander(DFA, caption = "Accuracy by Method")
```

Table 5: Accuracy by Method

	method	accuracy
2	Multiclass Polynomial Subset Balanced SVM	0.6323
1	Multiclass RBF Subset Balanced SVM	0.6398

```r
# pander(method.accuracy, caption = 'Accuracy by Method')
```