Shrinkage

The following code imports the pre-processed data model matrices and responses.

```
load("train.mat.JV.RData"); load("test.mat.JV.RData")
load("train.JV.RData"); load("test.JV.RData")
```

The `glmnet` library has `glmnet` and `cvglmnet` functions that can be used to fit lasso and ridge models. The `Metrics` library has a `rmsle` function that can be used to evaluate error, because this is the same metric that Kaggle uses.

```
library(glmnet)
library(Metrics)
```

Ridge Regression

Ridge regression uses least squares estimation, but with the constraint that the sum of the squared coefficient estimates must be less than a value s (a penalty tuning parameter).

The following code fits a ridge method to the training data. It then uses 5-fold cross validation to select the best penalty tuning parameter λ using the built-in function `cv.glmnet`. Lastly, it makes a prediction using that tuning parameter and writes the results to a .csv file.

```
rdg.reg <- glmnet(train.mat, train$SalePrice, alpha=0)  # fit ridge regression

rdg.s <- cv.glmnet(train.mat, train$SalePrice, alpha=0, nfolds=5)$lambda.min  # extract best pen
alty tuning parameter

rdg.pred <- predict(rdg.reg, s=rdg.s, newx=test.mat)  # make prediction
rdg.pred.df <- data.frame(Id=test$Id, SalePrice=rdg.pred); write.csv(rdg.pred.df, "rdg.pred.csv"
, row.names=F)  # write to CSV
```

The following code estimates the test error using 5-fold cross-validation from the results above.

```
set.seed(1)  # consistency of k-fold validation breaks
fold.index <- cut(sample(1:nrow(train.mat)), breaks=5, labels=FALSE)  # split data into 5 folds

rdg.rmslek <- c()  # initialize storage of the k RMSLE's
for (k in 1:5) {
    train.x <- train.mat[fold.index != k,]  # fold training set
    train.y <- train$SalePrice[fold.index != k]  # fold training response
    test.x <- train.mat[fold.index == k,]  # fold test set
    true.y <- train$SalePrice[fold.index == k]  # fold test response

    rdg.regk <- glmnet(train.x, train.y, alpha=0)  # fit ridge regression using training data
    rdg.predk <- predict(rdg.regk, newx=test.x, s=rdg.s, type="response")  # predict response fo
r test data
    rdg.rmslek <- c(rdg.rmslek, rmsle(actual=true.y, predicted=rdg.predk))  # store the RMSLE me
tric for this test fold
}

rdg.rmsle <- mean(rdg.rmslek)  # calculate the average RMSLE
```

Summary of Results

Method	Lambda	Estimated RMSLE	Actual RMSLE
Ridge Regression	15921.10313	0.13922	0.18548

The Lasso

The lasso uses least squares estimation, but with the constraint that the sum of the absolute value of the coefficient estimates must be less than a value s (a penalty tuning parameter).

The following code fits a lasso method to the training data. It then uses 5-fold cross validation to select the best penalty tuning parameter λ using the built-in function `cv.glmnet`. Lastly, it makes a prediction using that tuning parameter and writes the results to a .csv file.

```
las.reg <- glmnet(train.mat, train$SalePrice, alpha=1)  # fit lasso regression

las.s <- cv.glmnet(train.mat, train$SalePrice, alpha=1, nfolds=5)$lambda.min  # extract best pen
alty tuning parameter

las.pred <- predict(las.reg, s=las.s, newx=test.mat)  # make prediction
las.pred.df <- data.frame(Id=test$Id, SalePrice=las.pred); write.csv(las.pred.df, "las.pred.csv"
, row.names=F)  # write to CSV
```

The following code estimates the test error using 5-fold cross-validation from the results above.

```
set.seed(1)  # consistency of k-fold validation breaks
fold.index <- cut(sample(1:nrow(train.mat)), breaks=5, labels=FALSE)  # split data into 5 folds

las.rmslek <- c()  # initialize storage of the k RMSLE's
for (k in 1:5) {
    train.x <- train.mat[fold.index != k,]  # fold training set
    train.y <- train$SalePrice[fold.index != k]  # fold training response
    test.x <- train.mat[fold.index == k,]  # fold test set
    true.y <- train$SalePrice[fold.index == k]  # fold test response

    las.regk <- glmnet(train.x, train.y, alpha=1)  # fit lasso regression using training data
    las.predk <- predict(las.regk, newx=test.x, s=las.s, type="response")  # predict response fo
r test data
    las.rmslek <- c(las.rmslek, rmsle(actual=true.y, predicted=las.predk))  # store the RMSLE me
tric for this test fold
}

las.rmsle <- mean(las.rmslek)  # calculate the average RMSLE
```

Summary of Results

Method	Lambda	Estimated RMSLE	Actual RMSLE
Lasso Regression	376.45280	0.13168	0.19808