Analysis

In this section, we discuss the workflow process for executing Ordinary Least Squares (OLS), Ridge, LASSO, Principal Components, and Partial Least Squares Regressions on the `Credit-scaled.csv` dataset. (`Credit-scaled.csv` contains data from `Credit.csv` that was processed for regression analysis through factor conversion into dummy variables, mean centering, and variable standardization.)

For each regression method, we fit a model on testing data, calculate an MSE value based on the testing data, and fit the model again on the entire dataset for comparison.

All models for methods other than OLS are fit using ten-fold cross-validation. Since cross validation works through resampling data, we set a random seed before running fitting functions for reproducibility purposes. We then select the best model from a list of several produced by each fitting function using a tuning parameter. For shrinkage methods, the tuning parameter is λ; for dimension reduction methods, the parameter is the number of components.

The `R` code used for all five methods is roughly outlined in the sections below:

Ordinary Least Squares

```
# Fit ordinary least squares regression
ols_fit <- lm(y_train~x_train)
ols_fit_sum <- summary(ols_fit)

# Compute mean square error for the test set
ols_mse <- mean(summary(lm(y_test~x_test)$residuals^2))

# Refit the OLS regression on the full data set
full_data_ols_fit <- lm(y~x)
```

Shrinkage Methods

Both methods utilize the `glmnet` package.

Ridge Regression

```
# Fit the ridge model on training data
ridge_fit <- cv.glmnet(x_train,
                       y_train,
                       alpha = 0, # ridge parameter
                       lambda = grid <- 10^seq(10, -2, length = 100),
                       nfolds = 10, # performs 10-fold cross validation
                       standardize = FALSE, # we already standardized our data
                       intercept = FALSE)

# Select the best model
ridge_best_model <- ridge_fit$lambda.min

# Compute mean square error for the test set
ridge_predictions <- predict(ridge_fit, x_test, s = ridge_best_model)
```

```r
ridge_mse <- mean((y_test - ridge_predictions)^2)

# Refit the ridge model on the full data set
full_data_ridge_fit <- glmnet(x,
                              y,
                              alpha = 0, # ridge parameter
                              lambda = ridge_best_model,
                              standardize = FALSE,
                              intercept = FALSE)
```

LASSO Regression

```r
# Fit the LASSO model on the training data
lasso_fit <- cv.glmnet(x_train,
                       y_train,
                       alpha = 1, # LASSO parameter
                       lambda = grid <- 10^seq(10, -2, length = 100),
                       nfolds = 10, # performs 10-fold cross validation
                       standardize = F, # we already standardize our data
                       intercept = F)

# Select the best model
lasso_best_model <- lasso_fit$lambda.min

# Compute mean square error for the test set
lasso_predictions <- predict(lasso_fit, x_test, s = lasso_best_model)

lasso_mse <- mean((y_test - lasso_predictions)^2)

# Refit the LASSO model on the full data set
full_data_lasso_fit <- glmnet(x,
                              y,
                              alpha=1, #LASSO
                              lambda = lasso_best_model,
                              standardize = F,
                              intercept = F)
```

Dimension Reduction Methods

Both methods utilize the `pls` package.

Principal Components Regression (PCR)

```r
# Fit the PCR model on the training data
pcr_fit <- pcr(y_train~x_train,
               scale = FALSE,
               validation = "CV" # performs 10-fold cross validation
               )

# Select the best model
```

```r
pcr_best_model <- which.min(pcr_fit$validation$PRESS) #best number of components

# Compute mean square error for the test set
pcr_predictions <- predict(pcr_fit, x_test, s = pcr_best_model)

pcr_mse <- mean((y_test - pcr_predictions)^2)

# Refit the PCR model on the full data set
full_data_pcr_fit <- pcr(y~x,
                         validation = "CV")
```

Partial Least Squares Regression (PLSR)

```r
# Fit the PCR model on the training data
pls_fit <- plsr(y_train ~ x_train,
                scale = FALSE,
                validation = "CV")

# Select the best model
pls_best_model <- which.min(pls_fit$validation$PRESS) #best number of components

# Compute mean square error for the test set
pls_predictions <- predict(pls_fit, x_test, s = pls_best_model)

pls_mse <- mean((y_test - pls_predictions)^2)

# Refit the PLSR model on the full data set
full_data_pls_fit <- plsr(y ~ x,
                          scale = FALSE,
                          validation = "CV",
                          ncomp = pls_best_model)
```