Please complete the assigned problems to the best of your abilities. Ensure that the work you do is entirely your own, external resources are only used as permitted by the instructor, and all allowed sources are given proper credit for non-original content.

1 Recitation Exercises

These exercises are to be found in: **Introduction to Statistical Learning, 7th Printing (Online Edition)** by *Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani.*

1.1 Chapter 5

Exercises: 2,3

1.2 Chapter 6

Exercises: 1,2,3,4,5

2 Practicum Problems

These problems will primarily reference the *lecture materials and the examples given in class* using **R** and **CRAN**. It is suggested that a *RStudio* session be used for the programmatic components.

2.1 Problem 1

Load the *Yacht Hydrodynamics* sample dataset from the UCI Machine Learning Repository (**yacht_hydrodynamics.data**) into **R** using a dataframe (**Note**: The feature labels need to be manually specified). Use the **caret** package to perform a 80/20 test-train split (via the **createDataPartition** function), and obtain a training fit for a linear model. (**Hint**: The model fit should use all available features with the *residuary resistance* as the target.). What are the training MSE/RMSE and R^2 results? Next, use the **caret** package to perform a bootstrap from the full sample dataset with N=1000 samples for fitting a linear model (via the **trainControl** method), resulting in a training MSE/RMSE and R^2 for each resample. Plot a histogram of the RMSE values, and provide a mean RMSE and R^2 for the fit. How do these values compare to the basic model? How does the performance on the test set for the original and boostrap model compare?

2.2 Problem 2

Load the *German Credit Data* sample dataset from the UCI Machine Learning Repository (**german.data-numeric**) into **R** using a dataframe (**Note**: The

final column is the class variable coded as 1 or 2). Use the **caret** package to perform a 80/20 test-train split (via the **createDataPartition** function), and obtain a training fit for a logistic model via the **glm** package. (**Hint**: You may select a subset of the predictors based on exploratory analysis, or use all predictors for simplicity.). What are the training Precision/Recall and F_1 results? Next, use the **trainControl** and **train** functions to perform a k=10 fold cross-validation fit of the same model, and obtain cross-validated training Precision/Recall and F_1 values. How do these values compare to the original fit? How does the performance on the test set for the original and cross-validated model compare?

2.3 Problem 3

Load the *mtcars* sample dataset from the built-in datasets (**data(mtcars)**) into **R** using a dataframe. Perform a basic 80/20 test-train split on the data (you may use **caret**, the sample method, or manually) and fit a linear model with *mpg* as the target response, and all other variables as predictors/features (you will need to set up a dummy variable for *am*). What features are selected as relevant based on resulting t-statistics? What are the associated coefficient values for relevant features? Perform a *ridge* regression using the **glmnet** package from CRAN, specifying a vector of 100 values of λ for tuning. Use cross-validation (via **cv.glmnet**) to determine the minimum value for λ - what do you obtain? (**Hint**: You can use **doMC** in order to speed-up your cross-validation by specifying **parallel=TRUE** in your **glmnet** calls.). Plot training *MSE* as a function of λ (you may also use $\log \lambda$). What is out-of-sample test set performance (using **predict**), and how do the coefficients differ versus the regular linear model? Has ridge regression performed shrinkage, variable selection, or both?

2.4 Problem 4

Load the *swiss* sample dataset from the built-in datasets (**data(swiss)**) into **R** using a dataframe. Perform a basic 80/20 test-train split on the data (you may use **caret**, the sample method, or manually) and fit a linear model with *Fertility* as the target response, and all other variables as predictors/features. What features are selected as relevant based on resulting t-statistics? What are the associated coefficient values for relevant features? Perform a *lasso* regression using the **glmnet** package from CRAN, specifying a vector of 100 values of λ for tuning. Use cross-validation (via **cv.glmnet**) to determine the minimum value for λ - what do you obtain? (**Hint**: You can use **doMC** in order to speed-up your cross-validation by specifying **parallel=TRUE** in your **glmnet** calls.). Plot training *MSE* as a function of λ (you may also use $\log \lambda$). What is out-of-sample test set performance (using **predict**), and how do the coefficients differ versus the regular linear model? Has lasso regression performed shrinkage, variable selection, or both?