Homework 5: Handwritten Digit Recognition (with Lasso and Ridge)

STAT 665

due Thurs. March 2nd 5PM

For this homework assignment, we will return to the handwritten digits dataset and apply the ideas of ridge regression and lasso to classify digits via multinomial logistic regression. Review Homework 3 if you need a reminder of what the dataset looks like. We'll only be using the following two files.

- 'digits_train.csv' - a training set
- 'digits_valid.csv' - a validation set

You can use the `glmnet` package in R, or something equivalent in Python. It seems the `LogisticRegressionCV()` function in the `sklearn` library in Python may be relevant, but it is possible that parts of this problem set will be hard to accomplish using Python.

Recall that for a classification problem with K classes, multinomial logistic regression typically fits $K-1$ equations that relate log odds to a linear combination of a set of predictors. The parametrization of multinomial logistic regression that is used in `glmnet()` actually fits K equations instead of $K-1$.

Helpful Notes about `glmnet` for This Homework

Note that the `glmnet()` function and the associated `cv.glmnet()` function can be used for classification via multinomial logistic regression with the following specifications:

- Ensure that the second argument `y=` is a factor variable.
- Ensure that the optional `family=` argument is set to `multinomial`.
- For the lasso, an optional argument `type.multinomial=` can take on the values "ungrouped" (default, suggesting that each of the 10 resulting fitted equations can use different sets of non-zero predictors) or "grouped" (suggesting that the same set of predictors must be used across all 10 fitted equations). Another way to think about it is that "ungrouped" flavor of the algorithm applies the lasso/ridge constraint uniformly across all coefficients estimated. If you want the K coefficients corresponding to say, pixel 8, to either simultaneously be 0 or non-0, you would use a "grouped" lasso penalty.
- For this assignment, for cross-validation, ensure you also set the argument `type.measure="class"` so that misclassification error is used to compare models with different λ values.

When using the `predict()` function for predicting classes using a `glmnet` object, you will want to use the argument `type="class"`.

Part 1 (50%)

(1) Apply the lasso penalty to train a multinomial logistic regression classifier with `type.multinomial="grouped"`. Use 10-fold cross-validation on the training set to select your λ. Show a plot that displays the misclassification error on the y-axis for values of $\log(\lambda)$ on the x-axis. (Note: The cross-validation procedure will take some time to run, so get it started, and then go grab a cup of coffee or stretch a bit.)

(2) Report λ_{min} (the λ value that achieves the best error/deviance) and λ_{1se} (the λ value obtained using the one-standard deviation rule). Next, use both λ_{min} and λ_{1se} to generate predictions on the validation set. Report the resulting misclassification error rates.

(3) For $\lambda = \lambda_{1se}$, how many predictors are selected by `glmnet()`? How many non-zero coefficients are estimated in total for this value of λ?

(4) Repeat parts (1)-(3) with `type.multinomial="ungrouped"`.

Part 2 (20%)

(1) Apply the ridge penalty to train a multinomial logistic regression classifier. (I would not use the `type.multinomial=` argument from `cv.glmnet()` because there's no need for that here.) Again, use 10-fold cross-validation on the training set to select your λ. Show a plot that displays the misclassification error on the y-axis for values of $\log(\lambda)$ on the x-axis.

(2) Report λ_{min} and λ_{1se} and their associated misclassification errors for the validation set.

Part 3 (30%)

(1 paragraph) At this point, we have considered 3 different shrinkage approaches to classifying the handwritten digits. In your own words, how do these 3 approaches differ? Compare the performance of ridge, lasso, and your best approach from Homework 3 in prediction accuracy on the validation set. If you had to pick a favorite final model, what would you choose? Justify your answer.

What to Submit

Just your compiled PDF file, from a Python notebook or from R Markdown.