# Final Report

Samuel Burge, Chad Austgen, Skylar Liu

April 19, 2022

## Classification Task

### Methodology

For the classification task we approached the problem with a spread of well-known classification algorithms. This included the Naive Bayes classifier, regularized logistic regression (specifically ridge, lasso and elastic net regularization), gradient boosted trees, random forests and support vector machines (using radial and polynomial kernels). For model selection and assessment, we utilized 10-fold cross-validation and, with some algorithms, another nested cross-validation to select tuning hyper-parameter using a grid search. Nested cross-validation is a well-known approach to handling both tuning and assessment of models with hyper-parameters, and we opted to employ this approach to avoid potential over-fitting and bias introduced when the same data used to validate the models is also included in hyper-parameter optimization (Cawley and Talbot 2010).

The only classifier we used that was in covered in the course was the regularized version of the logistic regression model. Before the analysis we created large plot matrices and verified across all predictors that $X$ and $Y$ were not well separated suggesting the best linear method might be logistic regression. However, we were not able to fit a logistic

regression model outright since $p > n$. We utilized regularization methods to reduce the features in our analysis using lasso, ridge, and elastic net regularization methods (Zou and Hastie 2005). We also opted to use a different boosting algorithm for the classification trees, XGBoost, compared to the algorithms found in R's gbm package. This was primarily due to the computational resources necessary to perform nested cross-validations, as XGBoost is known for it's impressive predictive performance and computational efficiency (Chen and Guestrin 2016).

## Results

The results of our analysis are depicted in the table below. Note the degree to which some classifiers, such as ridge logistic regression, under-performed during the assessment despite a low training error, on average. Overall, the tree-based methods had the best performance of all the classifiers and the boosted classification trees had the best generalization performance with a CV error rate of 31%. After re-fitting the final model using the same grid search procedure, our model's tuning parameters were an 80% subsample ratio the training instances, a learning rate (shrinkage penalty) of 0.001 ,and a max interaction depth for each tree of 4. These tuning parameters were selected primarily to help us avoid over-fitting the model given the large amount of flexibility these algorithms can provide if left unchecked.

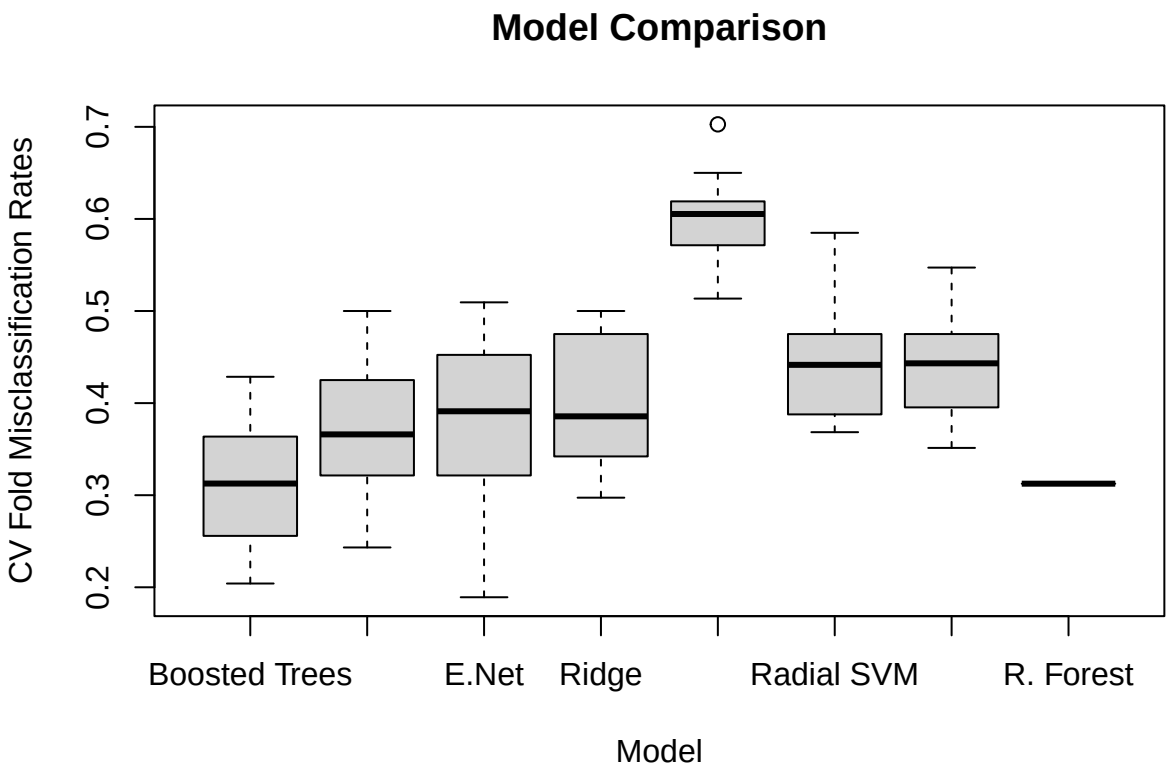


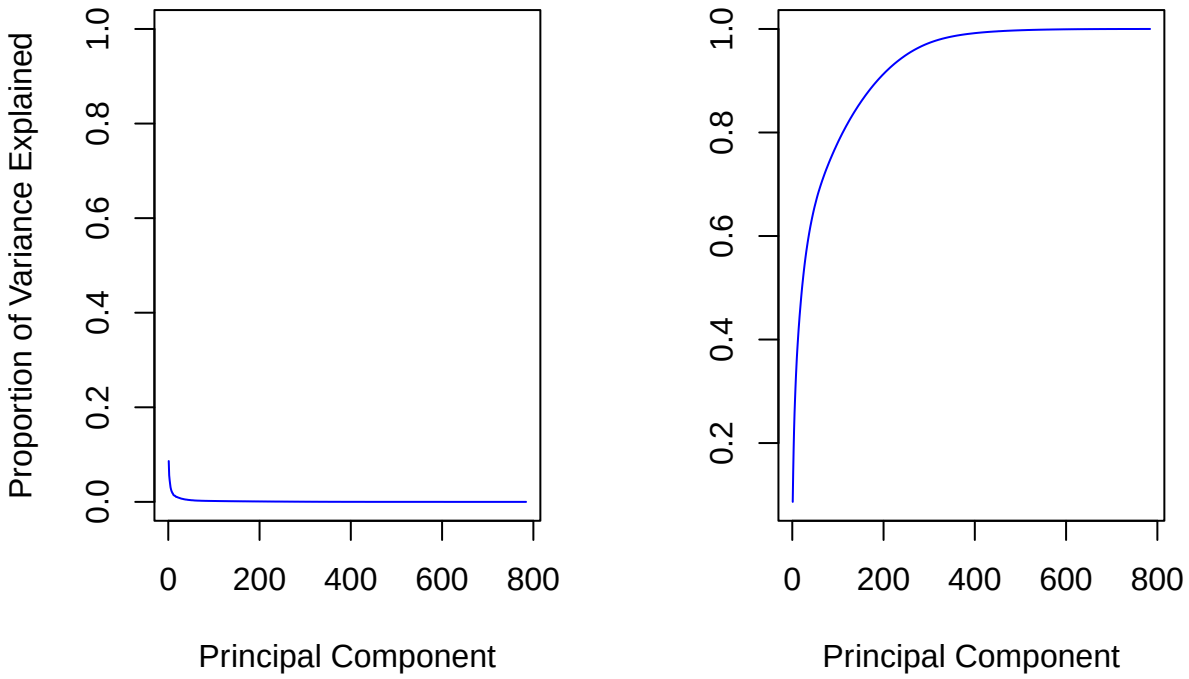

Table 1: Results from 10-fold cross-validation to assess model generalization performance.

| | Training Error | Est. Test Error |
|---|---|---|
| **Lasso** | 0.3060377 | 0.3727149 |
| **Net** | 0.3060377 | 0.3814926 |
| **Ridge** | 0.1786087 | 0.3972233 |
| **Naive Bayes** | 1.0000000 | 0.6019532 |
| **Radial SVM** | 0.4468479 | 0.4460467 |
| **Poly. SVM** | 0.4147253 | 0.4384205 |
| **Random Forest** | NA | 0.3125000 |
| **Boosted Trees** | 0.0401765 | 0.3081585 |

# Clustering Task

## Pre-processing

For the clustering task, we initially looked at performing principal components analysis (PCA) to reduce the number of dimensions in the data set. The two scree plots below show that the number of principal components necessary to capture at least 90% of the variation in the data set was about 187, which did not seem beneficial enough to consider for the analysis. Therefore, we decided to retain all the original features in the data set.




## Methodology

We opted to use k-means and hierarchical clustering, in part because the high-dimensional data is intractable with DBSCAN and due to computing restraints. Since the given data

does not have any contextual basis for selecting the number of clusters $K$, we need one or more criteria to determine the number of clusters. Several well-known and widely used approaches include the elbow method, the silhouette method (Rousseeuw 1987), and more recently the use of the gap statistic (Tibshirani, Walther, and Hastie 2001).

# References


Cawley, Gavin C., and Nicola L. C. Talbot. 2010. "On over-Fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation." *Journal of Machine Learning Research* 11 (70): 2079–2107. http://jmlr.org/papers/v11/cawley10a.html.

Chen, Tianqi, and Carlos Guestrin. 2016. "XGBoost: A Scalable Tree Boosting System." In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–94. KDD '16. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2939672.2939785.

Rousseeuw, Peter J. 1987. "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis." *Journal of Computational and Applied Mathematics* Volume 20 (https://doi.org/10.1016/0377-0427(87)90125-7): 53–65.

Tibshirani, Robert, Guenther Walther, and Trevor Hastie. 2001. "Estimating the Number of Clusters in a Data Set via the Gap Statistic." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63 (2): 411–23. https://doi.org/https://doi.org/10.1111/1467-9868.00293.

Zou, Hui, and Trevor Hastie. 2005. "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 67 (2): 301–20. http://www.jstor.org/stable/3647580.