Comparing Statistical Classification Techiques: An Example with US Census Data

David Wigglesworth

December 18, 2018

1 Questions / Introduction

With the expansion of computational power and data availability over the last few decades, it has become much easier for businesses, governments, and other economic agents to make anticipate future outcomes with predictive modeling. Classification is a particular form of predictive modeling wherein analysts attempt to assign individuals to one of several outcome categories. For example, banks are naturally interested in predicting whether a potential borrower may default on her loans. From a statistical perspective, we suppose there is a discrete random variable Y whose data generating process we estimate using a joint distribution of predictors X. This paper compares popular classification techniques in modern supervised machine learning with a focus on nonstructural prediction. We are particularly interested in discussing the motivations and merits from the perspective of an undergraduate or young industry practitioner.

We evaluate the performance of ten classification techniques on a testing set (or hold-out set) of a relatively clean and interesting dataset from the US Census. Our objects of interest include the accuracy, sensitivity, and specificity rates computed on the testing set. How well do these techniques predict whether a given individual makes above or below $50,000? How do their performances compare to one another? How do their performances compare? How does our prior knowledge of the data and models help us make sense of the results?

2 Literature Review

Through their association with terms such as "machine learning", statistical classifiers may be percieved as irreducibly complex computer programs by new students. However, the earlier statistical classifiers far predate the luxuries of modern computation. One of the earliest statistical classifiers, Fisher's Discriminant Analysis, rely largely on matrix multiplication that could be completed by hand (Fisher 1936). Alternative formulations only require computing sample means and variances (James et al. 2013). Two decades later, techniques such as Logistic Regression could be estimated using Newton-Raphson (Cox 1958) (Friedman, Hastie, and Tibshirani 2001). With the advent of modern computing, it became easier to implement decades-old algorithms that were once infeasible. For example, the support vector machines algorithm was originally conceived in the 1960s. However, the development of kernel estimation methods allowed SVMs to be implemented as a classifier (Boser, Guyon, and Vapnik 1992). Therefore, the arsenal of classification techniques that data analysts enjoy today is the result of a decades-long collaboration between statisticians and computer scientists.

There are generally two types of classification models: those that only consider an all-or-none exclusive distinction between categories of interest and those that estimate a probability of membership to each category (Dreiseitl and Ohno-Machado 2002). However, neither type of model outperforms the other under general conditions (Lim, Loh, and Shih 2000) (Dreiseitl and Ohno-Machado 2002). Some methods that ostensibly fall into use all-or-none assignment can be reframed as probability models. For example, data analysts may assign the probability of an observation belonging to a category as the number of nearest neighbors of that category as a fraction of the K-nearest neighbors. Suppose that of five nearest neighbors, three are red and two are blue; one can estimate the probability of our given observation being red as 60% and blue as 40%. Ultimately, however, there is no difference in outcome between picking the most likely category (treating the model as a Bayesian classifier) and all-or-none assigning the observation to the modal category; each frame of reference results in a maximum error of twice the Bayesian probability of

error (Cover and Hart 1967). On the other hand, the interpretation; a data point that falls on the "red" side of the fitted hyperplane cannot also fall on the "blue" side. This, however, is equivalent to estimating $P(Y = red|X) = 1$ and $P(Y = blue|X) = 0$. Thus, one can argue that the ultimate goal of classification is to construct probability models. Whereas regression attempts to estimate the realizations of a random variable Y with predictors X, classification attempts to estimate the probability mass function of Y as a function of predictors X.

Since classification techniques are governed by wide-ranging assumptions and computational frameworks, it is not surprising that data analysts' choice of models often depends on the structure and scope of a given project. For example, Decision Trees are useful when projects demand model interpretability, and K-Nearest Neighbors are useful when projects demand relatively quick runtimes (Kotsiantis, Zaharakis, and Pintelas 2007). However, there is no singular "go-to" model for projects that are concerned solely with predictive accuracy. There are few significant differences in average accuracy amongst popular classification techniques when trained and tested on a wide variety of datasets (Lim, Loh, and Shih 2000). As such, when time and interpretability are of little concern, it is often not obvious which classification model analysts should estimate on a given project without a detailed knowledge of both the relevant algorithms and the data itself. Consequently, one can train a variety of models on the data and pick the best performing algorithm by accuracy rate, sensitivity, specificity, area under the receiver operating characteristic curve, or some generalized loss function (Lim, Loh, and Shih 2000).

Taking the reviewed literature together, statistical classification as an economic problem: how do we estimate the optimal conditional probability model subject to the constraints of time, compute power, etc. To achieve a desired level of performance, a data analyst must either exercise a deep understanding of the subject area or consider a variety of models. However, it is impractical to consider every possible permutation of every known classifier. Thus, a data analyst must also have some prior knowledge to inform her choice of models. This paper is intended to illustrate a multiplie-model approach.

3 Data Description

The `adult` dataset is accessible through the University of California Irvine's Machine Learning Repository. A sub-sample of individuals was selected from the 1994 US Census Survey to include only those who are at least 16 years old, have an adjusted gross income of more than $100, work more than zero hours per week, and have sample weights greater than 1. The dataset contains information primarily concerning individuals' socioeconomic status and job information. The stated purpose of the `adult` dataset is "to determine whether a person makes over 50K a year." ("UCI Machine Learning Repository: Adult Data Set" 1996). For our purposes, let us say that the wealthier category belongs to the "positive" outcome and the poorer category belongs to the "negative" outcome.

Table 1 summarizes the column variables included in the dataset post-cleaning. The major steps in the data cleaning process are detailed and justified below.

First, we choose to discard the survey weights `fnlwgt` for two reasons. One, we are more concerned with prediction than obtaining interpretable model parameters. Two, the survey weights were constructed based off of the Census' entire survey sample, not our sub-sample. Even if we had reason to believe weights would augment our prediction, the provided survey weights would likely not be reflective of the holistic US population in 1994. For the sake of argument, if we assume that our dataset reflects a random sample of the population meeting the inclusion criteria outlined in the documentation, then the inclusion of survey weights is unnecessary altogether. After all, this paper is only concerned with comparing testing set performance of various classification methods, not studying the dynamics of income in the labor market.

Two, since `education_num` is just a numeric re-coding of `education`, we should remove one column to prevent any of our models from using redundant information. Including `education_num` is advantageous for some models since it uses spends only one degree of freedom as opposed to several degrees of freedom in the form of dummy variables. On the other hand, including `education` is likely to give an advantage to models that can cleverly split and navigate nonlinear numeric variables, e.g. random forests, over linear

models, e.g. logistic regression. We choose to advance with `education` and discard `education_num` while acknowledging that this decision could have a substantial impact on our results.

Three, we choose to discard the `country`. Since there are many unique values that appear very few times, not every country is guaranteed to be present in both the training and testing data partitions. Also, the inclusion of another several-level factor variable will also exacerbate the previously mentioned degrees of freedom problem. Our ad hoc solution is to replace `country` with a factor variable that indicates whether one's home country is the United States or else; we call it`immigrant`.

4 Methods

4.1 Data Partitioning

We use random sampling to split the data into two partitions: the training and testing sets. The training set, which we use to fit our models, contains 80% of the total observations in `adult`. The testing set, which we use to evaluate model performance, contains 20% of the total observation in `adult`.

4.2 Models

This paper considers ten classification techniques: the Null model, Logistic Regression (Logit), Regularized Logit (Regularized Logit), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Naive Bayes, K-Nearest Neighbors (KNN), Random Forest, Support Vector Machines (SVM), and Neural Networks. These models were chosen because of their effectiveness and popularity; all of them merit discussion in trusted resources such as Friedman et al.'s *The Elements of Statistical Learning*. Absent any relevant knowledge of the determinants of income in the United States, these models constitute a good starting point.

Each model is fit using R's `caret` package with the help of dependencies when needed; e.g., `caret` requires that the `kernlab` package be loaded fit kernel density estimates. Of these ten models, three were not thoroughly discussed in ECON224. Below, we briefly discuss these three models.

4.2.1 Naive Bayes

As its name might suggest, Naive Bayes is motivated by Bayes' Rule. Let k be a category of interest out of the set of categories $l = \{1, \dots, L\}$. Let X_i be a vector of m predictors for individual $i \in \{1, \dots, n\}$. Let the X_j be a vector of values corresponding to predictor $j \in \{1, \dots, m\}$:

$$P(Y_i = k|X_1, ..., X_n) = \frac{P(Y_i = k) \cdot P(X_1, \dots, X_n|Y_i = k)}{\sum_l P(Y_i = l) \cdot P(X_1, \dots, X_n|Y_i = l)}$$

The model's naivety comes from the assumption that $\{X_1, ..., X_n\}$ are conditionally independent, allowing us to replace the likelihood terms with the product the probabilities of X_i given Y_i.

$$P(Y_i = k|X_1, ..., X_n) = \frac{P(Y_i = k) \cdot \prod_i P(X_i|Y_i = k)}{\sum_l P(Y_i = l) \cdot \prod_i P(X_i|Y_i = l)}$$

We can substitute estimates of elements on the right-hand side of the above equation to estimate the probability that i belongs to category k given $\{X_1, ..., X_n\}$.

The prior probabilities can be estimated by substituting the proportion of training data that belongs to category k. That is:

$$\hat{P}(Y_i = k) = \frac{1}{n} \sum_i I(Y_i = k)$$

The likelihood terms can be estimated a few different ways. In this paper, our final Naive Bayes model uses Nadaraya-Watson to estimate joint density functions for every unique combination of X_j and Y_i, which we call $f_{j|l}(\cdot)$:

$$\hat{P}(X_i|Y_i = k) = \{\hat{f}_{j|k}(X_{i1}), \dots, \hat{f}_{m|k}(X_{im})\}$$

Once we have the computed estimates, we use the following assignment rule. Notice that we do not need to consider the denominator of Bayes Rule since only the numerator changes across categories.

$$\hat{Y}_i = \underset{k}{\text{argmax}} \left[\hat{P}(Y_i = k) \cdot \prod_i \hat{P}(X_i | Y_i = k) \right]$$

4.2.2 Support Vector Machines

Suppose that each observation in a dataset belongs to one of two categories. In which case, the goal of the Support Vector Machine classifier is to find a hyperplane that separates the categories when the predictors of the training set are plotted in space. Specifically, we wish to find the "optimal" hyperplane: one that maximizes the sum of the orthogonal distances, or margin, between the hyperplane itself the nearest training observation belonging to each category. However, it is not always possible to fit a hyperplane that perfectly separates the training observations in k-dimensional space. In which case, we apply a kernel - typically a linear, polynomial, or radial bias function kernel - to transform the training data into higher dimensions where an acceptable hyperplane can be plotted.

In principle, it is possible to iteratively transform the data into higher dimensions until a hyperplane can perfectly split the training observations. However, as with most statistical learning techniques, fixation on fitting the training data perfectly tends to produce overfitting, which results in poor performance on the testing set. To prevent overfitting, we can impose a "budget": an allowance of training points that can fall on the wrong side of the optimal hyperplane. We thus arrive at the following optimization problem:

$$\begin{aligned}
&\max_{\beta_j, \epsilon_i} \; M \\
&\text{subject to} \;\; \sum_j \beta_j^2 = 1, \;\; y_i \left(\sum_j \beta_j X_{ij} \right) \geq M(1 - \epsilon_i) \\
&\text{subject to} \;\; \epsilon_i \geq 0, \;\; \sum_i \epsilon_i \leq C
\end{aligned}$$

Here, M is the margin. Also, ϵ_i is the unit-distance (relative to the hyperplane) between the margin and a point given that it falls on the wrong side of the margin; ϵ_i is 0 if the training observation falls on the right side of the margin. The first set of constraints ensures that we are indeed fitting a hyperplane; i.e., a linear combination of the predictor variables. The second constraint imposes a limit on how much ϵ_i we are willing to tolerate. Given some observation from the testing set, we assign it according to which side of the fitted hyperplane it lies when plotted in space. SVMs can be generalized to situations with more categories by fitting more hyperplanes.

4.2.3 Neural Networks

Neural Networks can operate as two-stage classifiers. Suppose we have a network diagram consisting of three layers. The first layer contains a node for every predictor variable X. The second layer consists of M nodes that represent "hidden" features Z derived from the first layer. The third layer contains one 0-1 node for every outcome category Y. Essentially, the hidden features are a linear combination of the predictors, and the outcome is modeled as a linear combination of derived features.

$$\begin{aligned}
Z_m &= f\left(\sum_j \alpha_j X_j \right), \;\; m = 1, \ldots, M \\
T_l &= \sum_m \beta_m Z_m, \;\; l = 1, \ldots, L \\
P(Y = k | X) &= g(T_k) \;\; \forall \;\; k \in l
\end{aligned}$$

With estimates of the conditional probability that observation i takes on category k, we can assign i to the most likely k. Above, $f(\cdot)$ is an "activation function" – commonly the logistic function. Also, $g(\cdot)$ denotes

a function that could be used to transform the linear combinations of Z. In classification settings, we often use the softmax function. Note that these are the functions that R's `nnet` package uses by default:

$$f(\alpha, X) = \frac{1}{1 + \exp(-\sum_j \alpha_j X_j)}$$

$$g(T) = \frac{\exp(T_k)}{\sum_l \exp(T_l)}$$

Neural Networks implicitly contain parameters called "weights", which are used to help form estimates of α and β. Optimal weights can be chosen using a process (not to be discussed here) called "backward propagation." These weights are not to be confused with "case weights", which give varying degrees of consideration to each observation (like survey weights). It is also common for neural networks to impose some amount of "decay" to prevent overfitting. Like the λ parameter in ridge regression, decay is an L2 Regularization problem that causes the weights to shrink towards zero as they pass through layers of the network (Friedman, Hastie, and Tibshirani 2001).

4.3 Tuning Methodology

Table 2 summarizes the tuning methodology for each model. Though we try to train each model as similarly as possible, we make three major ad hoc decisions to limit runtime:

- The Logit is tuned using stepwise AIC, not 10-fold cross-validation. The `caret` package has no pre-built way of tuning general linear models with cross-validation.
- The SVM is fit with a linear kernel. We do not consider other kernels.
- The Neural Network fixes case weights at one and uses the functions outlined in Section 4.2.3.

5 Results

Each model's testing set accuracy, sensitivity, and specificity rates are displayed in Table 3. We also construct confidence intervals around each computed performance indicator using the Clopper-Pearson procedure. The intervals are plotted in Figures 1, 2, and 3. These graphs enable us to easily test the hypothesis of equal performance across pairwise models and establish a ranking of models for each performance indicator.

Recall that we defined the "positive" outcome to be those who make above $50,000. Therefore, the sensitivity rate is the proportion of those making above $50,000 who are identified as such, and the specificity rate is the proportion of those making below $50,000 who are identified as such.

5.1 Accuracy

From an accuracy perspective, the clear winner is the Random Forest model, whose lower accuracy bound of 0.974 far exceeds the penultimate KNN model's upper accuracy bound of 0.875. As we expect, the worst performing model on-average is the Null model, but we cannot say with statistical certainty that LDA performed significantly better. We also fail to reject the hypothesis that the logistic regression, regularized logistic regression, and SVM models perform equally well.

5.2 Sensitivity

When we consider sensitivity as our performance indicator, the clear winner is again the Random Forest model. Other the high-performers include the Neural Network and KNN models, which perform similarly to one another. The Null model, by definition, predicts that each observation should make below $50,000, meaning that it is impossible for the Null model to correctly identify an individual making above $50,000; it follows that its sensitivity is zero. This time, however, low-performing models such as LDA and Naive Bayes perform significantly better than the Null Model.

5.3 Specificity

The Null and Naive Bayes model both correctly identify all of those making below $50,000. While this is definitionally the case for the Null model, it is surprising to see Naive Bayes perform well in this regard. It may also be surprising to see that LDA also performed well, managing to perform significantly better than the Random Forest. Here, the worst performer is the Neural Network.

6 Discussion

Under some circumstances, certain types of misclassifications are costlier than others. For example, a bank would much rather misclassify a solvent borrower as a potential defaulter than vice versa. In the former case, no loan would be issued, but in the latter case, the bank would lose out on its lent money. In the case of the `adult` dataset, it is not obvious whether we would prefer one type of misclassification over the other. As such, we suppose that each misclassification has equal weight and refrain from imposing a cost function to evaluate overall model performance. Instead, we discuss some of the more interesting results from a holistic perspective.

The Random Forest's high overall accuracy rate can be attributed to its ability to reliably identify wealthier individuals. This fact is particularly interesting when we consider that wealthier individuals are the less common case; they only constitute about one-quarter of the sample. At the same time, the models that perform well when correctly identify poorer individuals, namely LDA and Naive Bayes, tend to perform poorly when identifying wealthier individuals. Thus, we can say that the Random Forest is the most balanced model in that it does not perform poorly on either income category.

The KNN, Logit, Regularized Logit, SVM, and Neural Network models represent the "middle of the pack" with respect to accuracy. While we only fail to reject the hypothesis Logit, Regularized Logit, and SVM have the same accuracy rate, we might argue that there is little practical significance; they only differ by about five accuracy points. The fact that the Logit and Regularized Logit models perform similarly in all respects is interesting – shrinkage does not seem to substantially improve upon a well-specified model estimated with the typical maximum likelihood procedure.

Interestingly, LDA and Naive Bayes – classifiers both motivated by Bayes' Rule – perform similarly to the Null model in both sensitivity and specificity. This suggests that these methods were not able to acquire substantial information that goes beyond simple prior probability estimates. The poor performance is likely due to the strong assumptions of each model. That is, LDA assumes the predictors are jointly normally distributed with homogeneous variance and Naive Bayes assumes that the predictors are generated independently. QDA, a version of LDA that allows for heterogeneous variance, has significantly higher accuracy, but its overall performance is not stellar. The poor performance of these models could be easily anticipated since there is no reason to assume that the predictors are independent or Normally distributed. After all, predictors such as education level and occupation type are both related and generally asymmetric; i.e., not normally distributed.

We are careful to not generalize these results. For example, we cannot always guarantee that Random Forests will perform very well and LDA models will perform very poorly. In particular, this dataset features many predictor variables with several levels, forcing linear and parametric models, e.g. the Logit, to spend many degrees of freedom with the potential for little information gain. It would be interesting to see how the results of this paper's procedure would change across different datasets, especially those featuring more outcome categories and numeric predictors.

7 Appendix

Table 1: Column Variables of `adult` Dataset

Variable	Description
`age`	Continuous variable indicating age.
`type_employer`	Factor variable with 9 levels indicating job sector.
`fnlwgt`	Continuous variable indicating survey weight. (Discarded)
`education`	Factor variable with 16 levels indicating education level.
`education_num`	Numerically re-coded education variable: 1 through 16. (Discarded)
`marital`	Factor variable with 7 levels indicating marital status.
`occupation`	Factor variable with 15 levels indicating job type.
`relationship`	Factor variable with 6 levels indicating role in family unit.
`race`	Factor variable with 5 levels indicating race.
`sex`	Factor variable with 2 levels indicating sex.
`capital_gain`	Continuous variable indicating gains from non-wage sources.
`capital_loss`	Continuous variable indicating losses from non-wage sources.
`hr_per_week`	Continuous variable indicating hours worked per week.
`country`	Factor variable with 42 levels indicating native country. (Discarded)
`immigrant`	Factor variable with 2 levels indicating US is not native country. (Generated)
`income`	Factor variable with 42 levels indicating native country.

Table 2: Tuning Methodology for Each Model

Model	Parameters	Method	Object
Null	N/A	N/A	N/A
Logit	Stepwise Feature Selection	AIC	AIC
Regularized Logit	Mixing Parameter, Penalization Parameter	10-Fold CV	Accuracy
LDA	Stepwise Feature Selection, Number of Predictors	10-Fold CV	Accuracy
QDA	Stepwise Feature Selection, Number of Predictors	10-Fold CV	Accuracy
Naive Bayes	Laplace Correction, Kernel Type, Bandwidth	10-Fold CV	Accuracy
KNN	Number of Neighbors	10-Fold CV	Accuracy
Random Forest	MTRY, # Trees (Fixed at 500)	10-Fold CV	Accuracy
SVM	Cost, Kernel (Fixed at Linear)	10-Fold CV	Accuracy
Neural Network	Hidden Units, Decay, Case Weights (Fixed at 1)	10-Fold CV	Accuracy

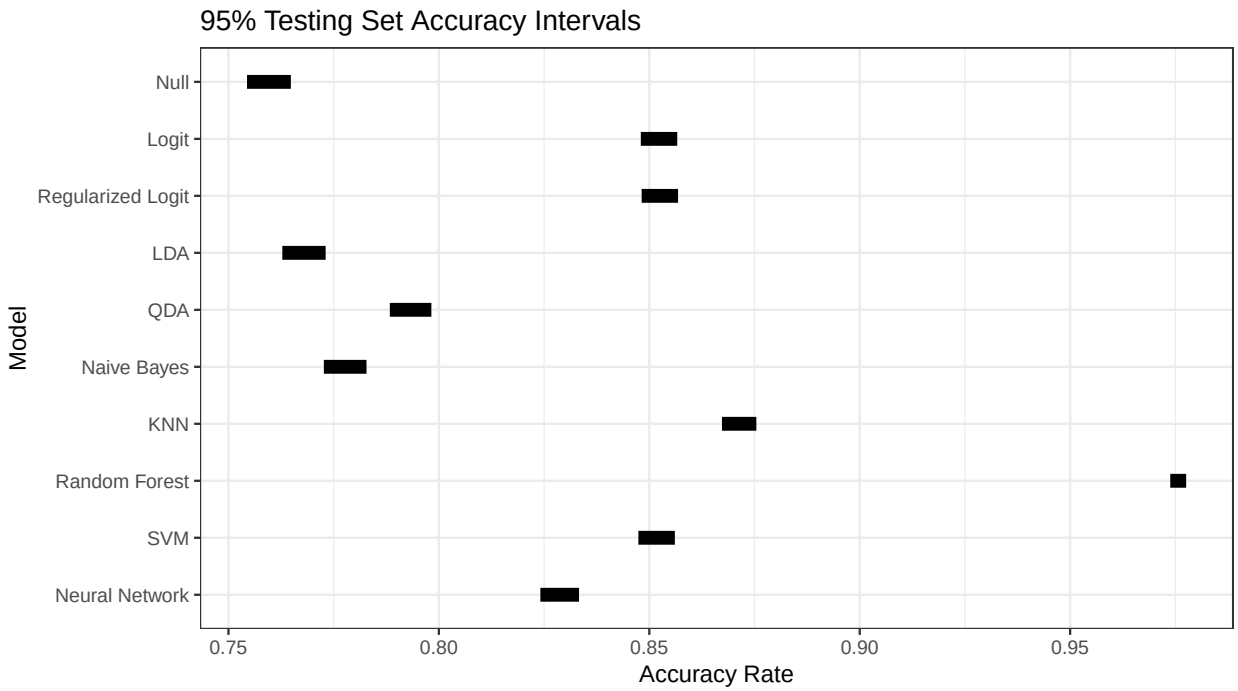

Figure 1: 95% Accuracy Intervals

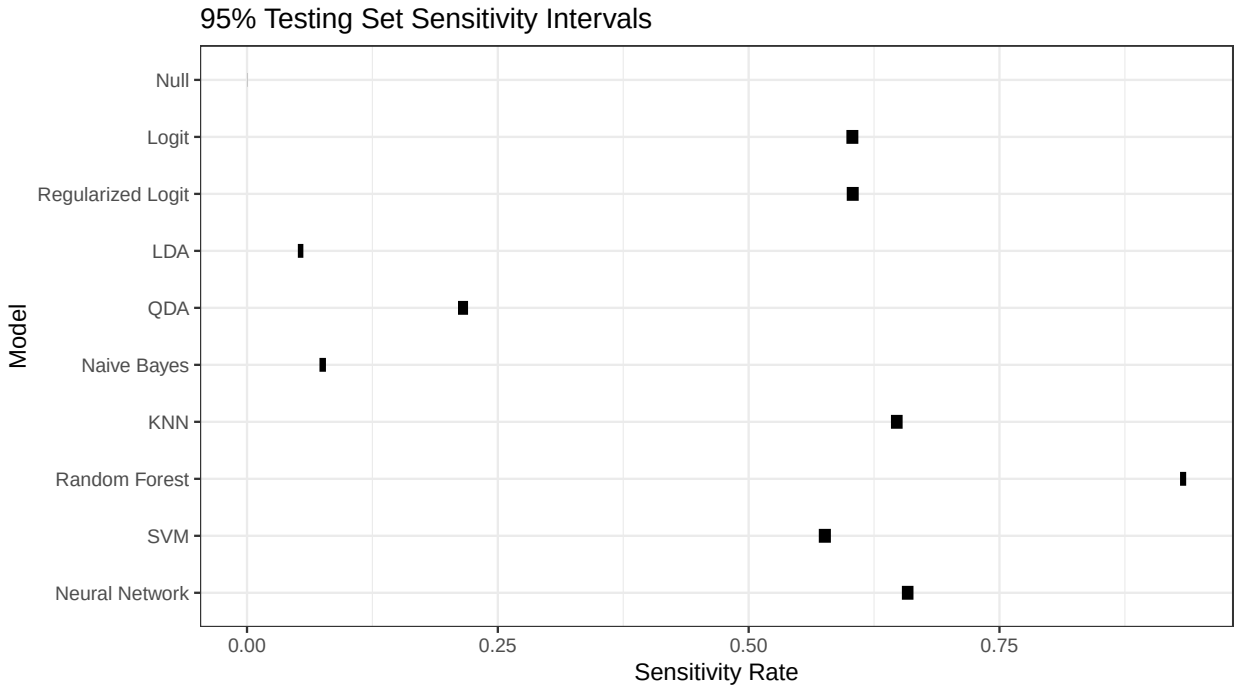

Figure 2: 95% Sensitivity Intervals

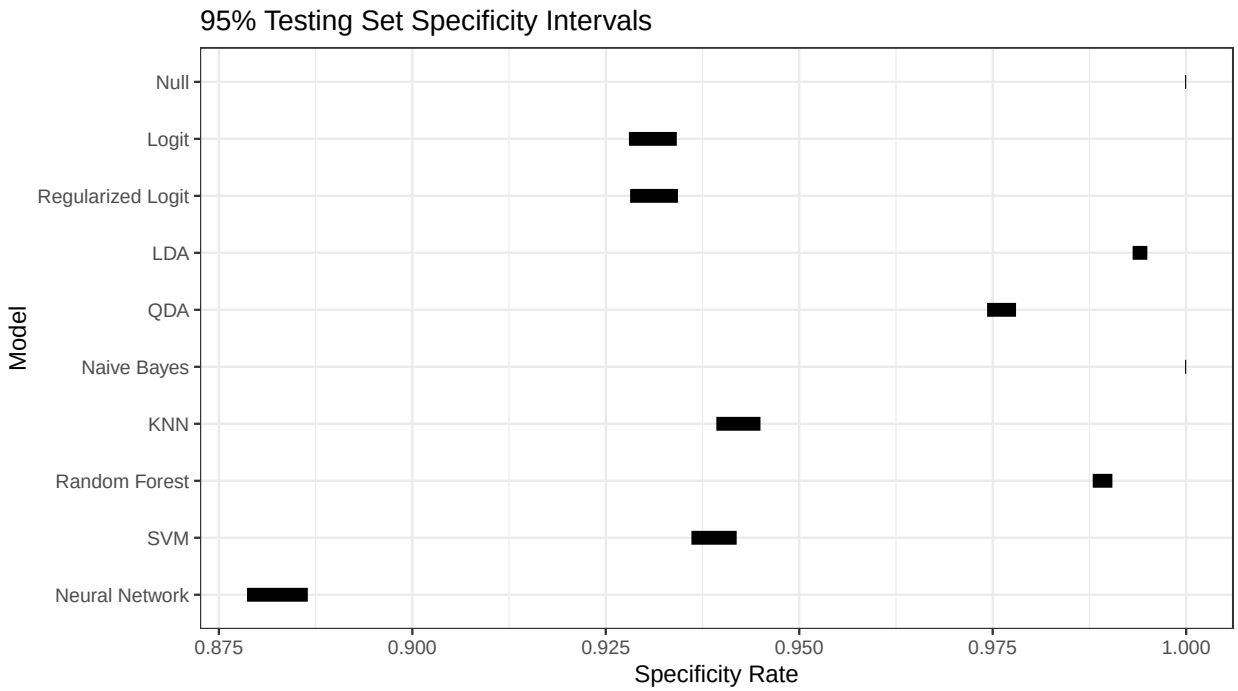

Figure 3: 95% Specificity Intervals

Table 3: Summary of Testing Set Performance Indicators

	Accuracy	Sensitivity	Specificity
Null	0.760	0.000	1.000
Logit	0.852	0.603	0.931
Regularized Logit	0.853	0.604	0.931
LDA	0.768	0.053	0.994
QDA	0.793	0.215	0.976
Naive Bayes	0.778	0.075	1.000
KNN	0.871	0.648	0.942
Random Forest	0.976	0.933	0.989
SVM	0.852	0.576	0.939
Neural Network	0.829	0.659	0.883

References

Boser, Bernhard E, Isabelle M Guyon, and Vladimir N Vapnik. 1992. "A Training Algorithm for Optimal Margin Classifiers." In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 144–52. ACM.

Cover, Thomas, and Peter Hart. 1967. "Nearest Neighbor Pattern Classification." *IEEE Transactions on Information Theory* 13 (1). IEEE: 21–27.

Cox, David R. 1958. "The Regression Analysis of Binary Sequences." *Journal of the Royal Statistical Society. Series B (Methodological)*. JSTOR, 215–42.

Dreiseitl, Stephan, and Lucila Ohno-Machado. 2002. "Logistic Regression and Artificial Neural Network Classification Models: A Methodology Review." *Journal of Biomedical Informatics* 35 (5-6). Elsevier: 352–59.

Fisher, Ronald A. 1936. "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7 (2). Wiley Online Library: 179–88.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. 2001. *The Elements of Statistical Learning*. Vol. 1. 10. Springer series in statistics New York, NY, USA:

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning*. Vol. 112. Springer.

Jed Wing, Max Kuhn. Contributions from, Steve Weston, Andre Williams, Chris Keefer, Allan Engelhardt, Tony Cooper, Zachary Mayer, et al. 2018. *Caret: Classification and Regression Training*. https://CRAN.R-project.org/package=caret.

Karatzoglou, Alexandros, Alex Smola, Kurt Hornik, and Achim Zeileis. 2004. "Kernlab – an S4 Package for Kernel Methods in R." *Journal of Statistical Software* 11 (9): 1–20. http://www.jstatsoft.org/v11/i09/.

Kotsiantis, Sotiris B, I Zaharakis, and P Pintelas. 2007. "Supervised Machine Learning: A Review of Classification Techniques." *Emerging Artificial Intelligence Applications in Computer Engineering* 160: 3–24.

Lim, Tjen-Sien, Wei-Yin Loh, and Yu-Shan Shih. 2000. "A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms." *Machine Learning* 40 (3). Springer: 203–28.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

"UCI Machine Learning Repository: Adult Data Set." 1996. https://archive.ics.uci.edu/ml/datasets/adult.

Zhu, Hao. 2018. *KableExtra: Construct Complex Table with 'Kable' and Pipe Syntax*. https://CRAN.R-project.org/package=kableExtra.