Language Classifier

Austin Chi

Purpose

- A tool to visualize and experiment with different classification methods.

Data

- Ten Thousand of the most frequently used words on Google (without swears)
- Ten Thousand of the most frequently used words on Spanish Wikipedia
- Encoded to UTF-8

App Navigation

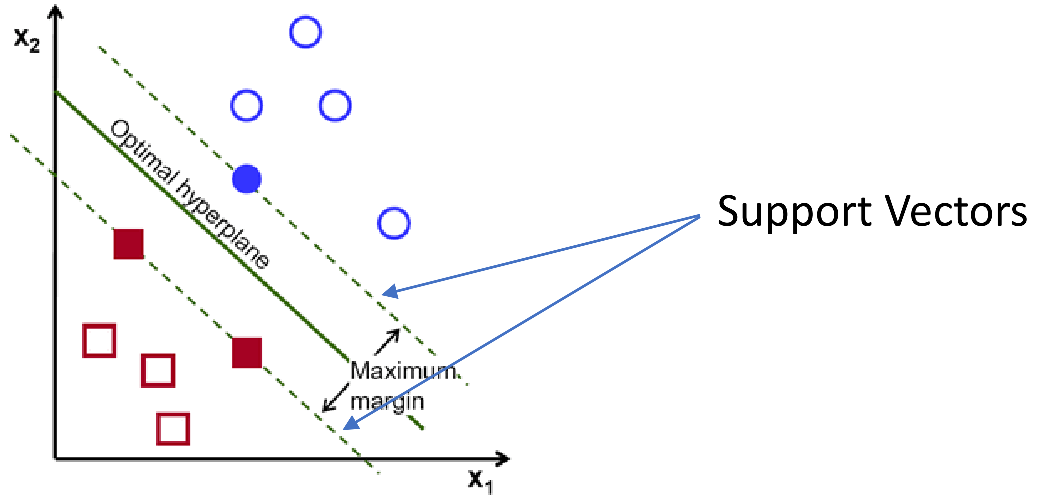
- 4 Different methods of classification
  - Neural Net
  - SVM
  - Random Forest
  - Boosted Trees

Neural Nets

- A 1 layer Neural Net with user configurable number of nodes and training iterations (steps)
- High Level Overview:
    - Inputs are first scaled. Then a weight is applied to each input and passed to the node where they are summed. A new weight is then applied and it is summed at the output. The output is compared with the true value and the error is backpropagated to the weights.
- Doesn’t work very well for this problem.

Support Vector Machines

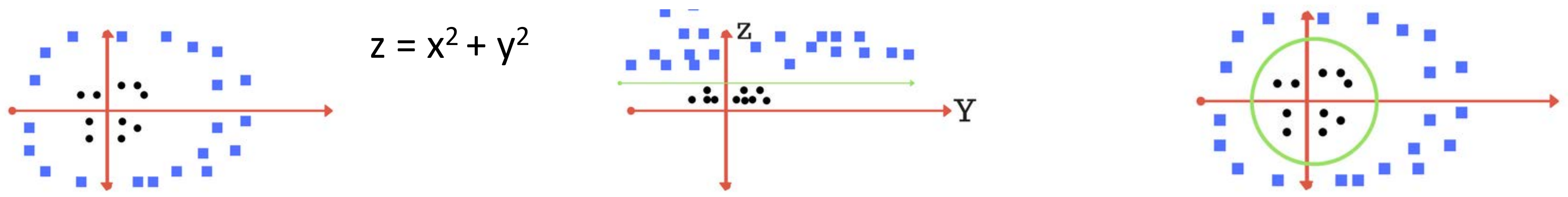
- An SVM with user configurable kernel and parameters. Each kernel has unique parameters to try.
- High Level Overview:
  - SVM finds the optimal Hyperplane defined by the maximum margin of the data (for hard margins, soft margins can be accounted for by introducing hinge loss)



Support Vector Machines

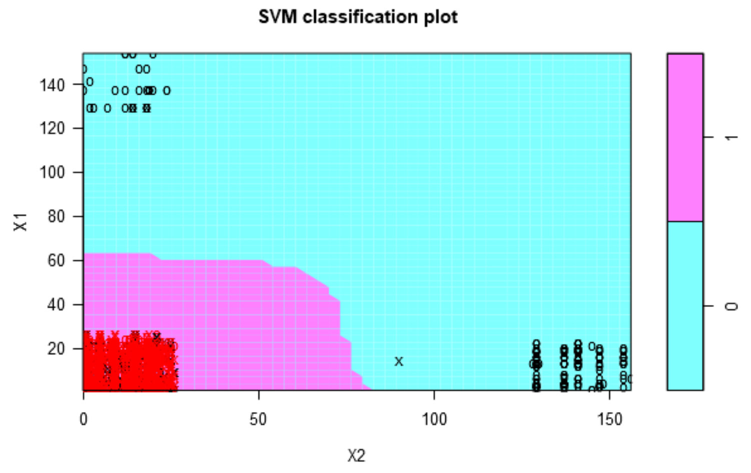
- High Level Overview Continued:
  - Many times data is not easily separable. By projecting the data into a transformed feature space through the use of a kernel. The classifier is still linear in this transformed space, but it is no longer linear in the original space.

Transform Back Into Original Space



Support Vector Machines

- Doesn’t work well for this problem, but can use diagnostics plots to understand why.
- In UTF-8 encoding, special characters (ñ, á, é ,etc…) are given high values, whereas the standard letters are given values from 1 to 26. A different encoding scheme could provide better results.



Random Forest

- A random forest with user selectable number of trees.
- High Level Overview:
    - Training Data is sampled with replacement repeatedly and used to train decision trees. The decision trees are then select from a random set of features at each candidate split. The different trees are then averaged for the final prediction
- This works much better than the previous two methods with minimal tuning.

Boosted Trees

- An AdaBoost Tree with user selectable parameters learning rate, interaction depth, number of trees, and minimum observations per node.
- High Level Overview:
  - A tree is trained with the data. The data is split into those the tree classified correctly and the those that were misclassified. Correctly classified data is weighted less and incorrectly classified data is weighted more. A new tree is then fit upon this weighted data and the process is repeated. After all the trees are built, the weighted average of the tree outcomes is used.

Boosted Trees

- This is the best method that is sensitive to tuning.
- Some tips:
  - Number of Trees: 100-1000
  - Learning Rate: [2-10]/Number of trees
  - Interaction Depth: [(4,6,8,10)]
  - Minimum Observations: 0

Tips by Owen Zhang

Packages Used

- randomForest based on Fortran code by Leo Breiman and Adele Cutler, ported to R by Andy Liaw and Matthew Wiener.
- gbm by Greg Ridgeway.
- e1071 maintained by David Meyer.
- nnet by Brian Ripley