Machine Learning & Data Mining
Miniproject 1

Caltech CS/CNS/EE 155
Released January 28^{th}, 2017

---

1. Introduction

- **Group members**
  Eli Sorey
  Kun ho (John) Kim
- **Team name**
  Stochastic Gradient, Decent
- **Division of labour**
  Eli focused on dataset cleaning, feature selection, and feature transformation. (Feature Engineering) Kun ho primarily worked on the model selection, model parameter optimization, and ensemble selection methods. (Models and techniques) We delve into the specific of our work in the following sections.

2. Overview

- **Models and techniques tried**
  - ***Ensemble Selection:** Submitted model.
  - **Gradient Boosting Machines**
  - **Random Forest**
  - **Adaptive Boosting**
  - **Extra Trees**
  - **Support Vector Machines**
  - **Artificial Neural Networks**
- **Work timeline**
  - **Day1 - Day4:** Preliminary data-processing & Broad model selection.
  - **Day 4 - Day 8:** Feature engineering & Single model optimization
  - **Day 8 - Day 12:** Ensemble selection & Final model tuning

3. Approach & Results

3.1. Overview

Our team's general pipeline for this competition is outlined in the flow chart below.

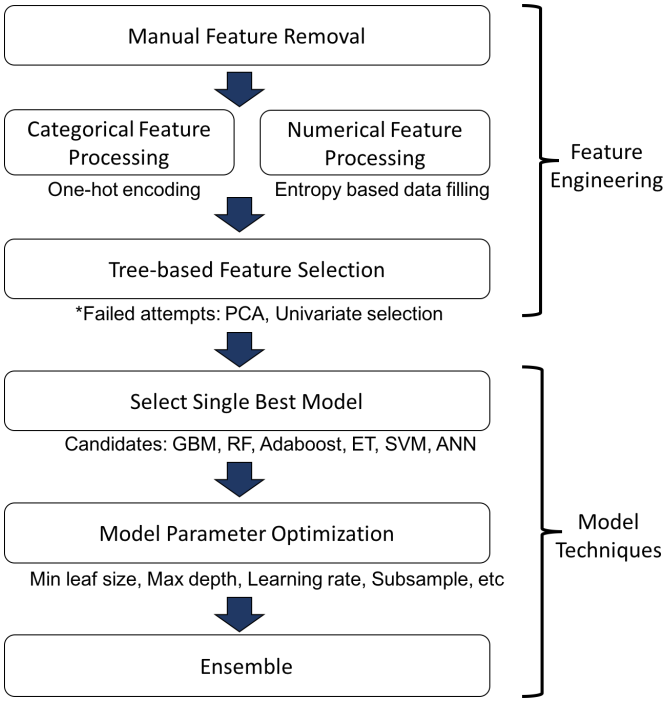

Figure 1: Overall pipeline

We first sought to transform the raw training data into a compact representation while keeping all the relevant information. The feature engineering part of the workflow covers methods of handling categorical features, numerical features, and the feature selection process. The model & techniques part covers our methods of model selection, parameter optimization, and stacking/ensemble selecting single models.

3.2. Feature engineering

3.3. Model & Techniques

[i]. Single model selection through cross validation

After preprocessing the training data, 6 broad model classes(GBM, RF, Adaboost, ET, SVM, ANN) were considered as candidates to determine which achieves the best single-model performance. 5-fold cross validation was performed and the model with the highest validation accuracy was chosen. The following bar graph demonstrates that gradient boosting machines (GBM) performed the best with a validation accuracy of 77.9% (with default parameters).

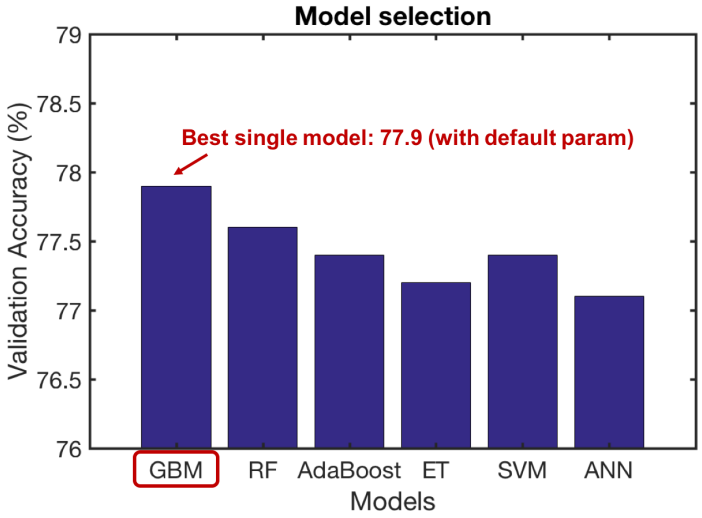

Figure 2: Cross validation for single best model selection *GBM attains the highest accuracy of* 77.9% *followed by random forests(77.6), Adaboost(77.4), Extra Trees(77.2), Support Vector Machines (77.4), and Artificial Neural Networks (77.1).*

Generally, we saw that tree-based non-linear classifiers out-performed linear models such as support vector machines. We attribute this to the fact that demographic features are often only separable by highly non-linear discriminant boundaries. This is in agreement with the fact that the training data projected onto the first two principal components showed no notable separating boundaries. (Hence, our attempt to preprocess the data using principal component eigenvector bases transformation was unsuccessful) This influenced our decision of including more tree-based non-linear classifiers into the pool of models to perform cross validation on. GBMs proved to be a low variance model as it achieved repeatable validation accuracy. (Random forests had relatively higher variance) However, the low variance characteristic made it difficult to enhance the performance of GBMs by only including GBMs in the pool of trained models to ensemble select on. Section 3.3-[iii] will elaborate more on this issue. We now move on to show the results of optimizing GBM model parameters.

[ii]. Parameter optimization for GBM

After selecting Gradient Boosting Machines(GBM) as the single best model, we sought to optimize the boosting base model (decision tree) parameters as well as the boosting specific parameters. We considered 5 main parameters for optimization: minimum leaf size, maximum height, learning rate, subsample, and number of estimators. Minimum leaf size and maximum height are both decision tree parameters which control the degree of regularization. Moreover, learning rate, subsample, and number of estimators alters the degree of stochasticity in training each base model, the constants involved in the boosting equation, and the number of boosted trees to train. The number of estimators demonstrated no significant impact on the cross validation accuracy beyond 500 trees and thus we omit the plot regarding this parameter. In fig. 4. we show the validation accuracy of the GBM model plotted against the remaining 4 parameters subject to optimization. The optimum parameters were discovered through a grid search function provided by sklearn, which performs cross validation on the full space of possible parameter quadruples (min leaf size, max height, learning rate, sub sample). Here we show the cross validation performance of each parameter separately, while fixing all other parameters to be the optimum values. Due to the slight randomness in the model that the learning algorithm converges to, the maximum cross validation accuracy differs slightly from plot to plot. Since we seek to show quantitative rationale for our choice of the optimum parameters, we deem these plots as sufficient. The overall best parameters chosen for the GBM model were: (min leaf size, max height, learning rate, sub sample) = (15, 3, 0.1, 1).

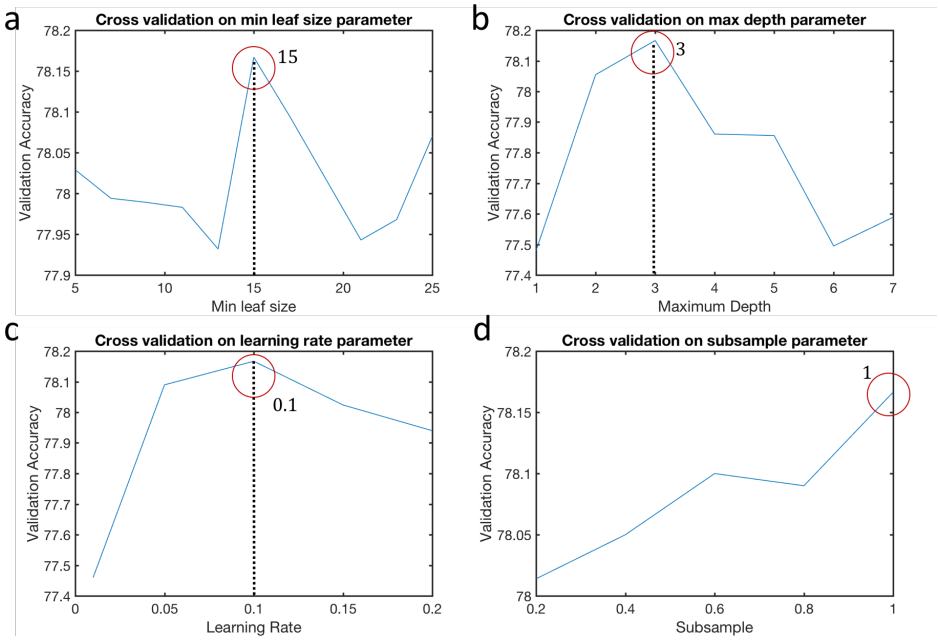

Figure 3: Cross validation for model parameter selection *Through an exhaustive grid search we determined the quadruple of parameters which gave the best validation accuracy around* 78.2%. *(a) Validation on min leaf size shows a peak at 15 (b) Validation on max depth shows peak at 3 (c) Validation on learning rate shows a peak at 0.1 (d) Validation on subsample shows a peak at 1*

[iii]. Handling class-imbalance

Upon analyzing the label distribution of the training data, we found that over 73% were of class 1, while only 27% were class 2. Such class-imbalance can often lead to the tree predicting in favor of the majority class, which generates a low-precision, high-recall classifier. (In terms of the majority class) Our solution to tackling this class-imbalance problem was to set a prediction bias. To elaborate, first recall that a decision tree makes predictions by taking the argmax of the probability distribution over the class-labels at each leaf node. For a binary classification task, this means that the class with probability higher than 0.5 becomes the prediction. We instead modify the threshold with a prediction bias p such that in order for class 1 to be the prediction at a leaf node, it must have a probability $0.5 + p$. Fig. 4. shows cross validation accuracy while varying p from -0.02 to 0.02. Other bias values (> 0.2 or < -0.2) led to worse model performance than the optimum value $p = 0.013$.

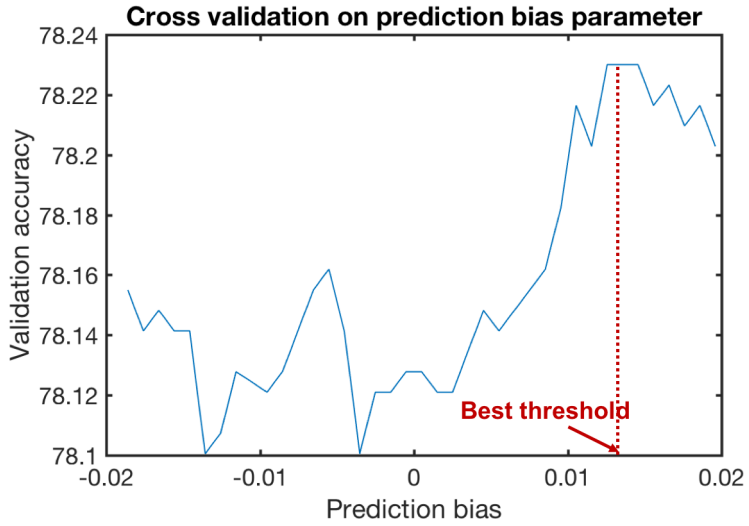

Figure 4: Cross validation accuracy vs prediction bias *Validation accuracy is highest with a prediction bias of* 0.013. *Qualitatively, a positive prediction bias means to predict in favor of class 2 (minority class).*

[iv]. Ensemble selection

85% of the total training data was partitioned towards training a model library and 15% was reserved as a validation set to ensemble select on. We trained a model library consisting of 20 GBMs, 20 ANNs, and 20 Random Forests. Upon an initial attempt of building a library with only GBMs, we realized that boosting methods have very low variance. Hence, most models generated similar predictions and ensemble selection did not lead to an improvement in the validation accuracy. Hence, we decided to add more diverse models into the pool in order to take the full benefit of both bagging and boosting. For succinctness of the report, we omit the parameter optimization process for ANNs and RFs. The final accuracy of the ensemble selected model was approximately 0.1 better than a single GBM attaining the highest validation accuracy ≈ 78.3. Our submission of this model received a **final score of 78.7 on the 2008 dataset and 76.17 on the 2012 dataset**.

4. Conclusion

[i]. Overall rank: 9^{th} for 2008 dataset, 26^{th} for 2012 dataset

[ii]. Discoveries

[iii]. Challenges

Dealing with real-word data was more difficult than training models to fit low-noise data which are often given out for homework assignments. We believe that learning how to engineer the features effectively was the most challenging aspect of this project. This required us to understand the features beyond their numeric values, which required many hours because there were 380 of them. We learned that many feature engineering techniques are task specific which led to many of our initial attempts (such as PCA, Univariate selection) failing. As a more concrete example, it was difficult to decide what to do with the various non-response values (negative values), as described in Section 3.2. Ensuring that the transformed training data and test data were compatible (ie, had the same shape) was an issue because our handling of categorical features sometimes created discrepancies in the number of features between the training and test sets. For instance, that a categorical feature attains values 1 and 2 in the training set and 3 in the test set. Then the transformed training set will have two features for the original categorical feature, whereas the transformed test set will have only one. This was remedied by adding additional feature columns to both data sets to ensure that they had the same features. Through overcoming these challenges, the most important lesson that we learned was that it is crucial to fully understand the data before attempting any feature selection or model optimization.

[iv]. Concluding Remarks

This competition gave us a sense of the difficulties machine learning practitioners face when modeling noisy real-word data. Lastly and most importantly, we would like to thank the TAs and the instructor for organizing such a great learning experience. We hope our report demonstrates that your efforts to find creative ways in teaching were not in vein.