Name: Kiran R

Location: Bengaluru, India

Email: rkirana@gmail.com

Competition:

1. Summary

- Using visualization I found out early on that the dataset prior to 2010 did not have any labels and excluded that from training. I also added variables when data was missing for certain features - to see if they could make a difference with the patterns of missing values
- The supervised learning methods I tried out for this competition were gradient boosting machines, vowpal wabbit, large scale regularized logistic regression & Support vector machines (liblinear), random forest and a bayesian regularized neural network. I did not use SVM and BRNN in my final submission though
- Tried several ensembling techniques – the best performing was the weighted average of the AUCs in the training dataset
- No external data sources were used

2. Features Selection / Extraction

Feature engineering is very important in this competition. The following were the key features by bucket:

- Text Mining – Parts of Speech
  - Percentage & Number of capitalized letters in the need, title, description and essays of the requestors [create_pct_caps.R]
  - create 'parts of speech' variables for the title, description and need_statement of the essays of the donors [create_pos_tags.R, create_parts_of_speech.R]
- Text Mining – Term-Document Matrix Approach
  - create binary TDMs for title, description and need_statement [runPyTDM.R, create_tdm.R]
- High Dimensionality Features
  - For the categorical variable, create a sparse matrix storing them as dummy variables [create_sparse.R]
- Counts for the categorical features
  - We binarize the categorical features by considering the count of the # of times it occured in train and test together [create_freq_features.R]
- Shrunken Averages for the categorical features
  - We get the shrunken averages for the categorical features. This prevents overfitting [create_shrunken_averages.R]
  - Similarly shrunken averages for each of the variables in outcomes dataset. [create_damp_alloutcomevars.R]
- General Features
  - flag_missing: Whenever “teacher_referred_count” or “great_messages_proportion” are missing, it is 1 else 0
  - date_posted_month
  - donor_charges := total_price_including_optional_support-total_price_excluding_optional_support
  - donor_charges_pct := donor_charges/total_price_including_optional_support

Feature selection was tried out using randomForest, glmnet and greedy random forest methods

3. Modeling Techniques and Training

The following models were run:

- Vowpal Wabbit
    - was run tuning the best combination of learning rate, decay learning rate with 20 passes on logistic method
    - best learning rate was 0.05 and best decay learning rate was 1
- XGBoost
    - Parameters eta and depth were tuned for binary logistic task
    - eta values of 0.05, 0.3 and 1 were tried while we tried depth of 3, 7 and 11
    - The right values were chosen via cross-validation and were different for different folds
- GBM Variants
    - Python GBMs – they all used learning rate of 0.01, interaction depth of 7 and 11 features were considered for splitting
        - Specifically one variant considered factor as integer while another considered factor as factor
    - R GBM – they all used bag.fraction = 1, learning rate/shrinkage of 0.01, 10 observations in each node, optimal number of trees tried in steps of 150 upto 15000 via early stopping
        - Specifically we tried one without factors and the factors with too many levels that were expressed as integers and the normal methods
        - These were all tried on the greedy selected variables from RF
- Undersampled random Forest
    - We tried undersampling the negative class with 20000 randomly chosen to build each tree and an equal number of positive examples.
- Weighted Random Forest
    - We tried 2 versions – a weighted random forest and an inverse weighted one – where the weights were reversed
    - This was done to provide diversity to the ensembling methods that followed

4. Code Description

The code in the files is clearly commented. A modular approach was used – every module has a separate name – the main.R file controls the flow

5. Dependencies

- Software Requirements
    - R version 3.02 or above
    - Ubuntu 14.04 Trusty Operating System
    - Python 2.7
- Data Mining Packages
    - Vowpal Wabbit (any version) must be installed and the command vw must be runnable from command line meaning vw must be in the PATH variable: https://github.com/JohnLangford/vowpal_wabbit
    - XGBoost must be installed from Tianqui Chen's repository: https://github.com/tqchen/xgboost
    - The development version of scikit Learn installed because we use GBMs with early stopping to prevent overfitting + we want to randomly choose the number of trees at each split
    - The following libraries must be installed in R data.table Matrix glmnet doMC

foreach rbenchmark Metrics gbm RRF lme4

6. How To Generate the Solution (aka README file)

- Copy the files in github to your local machine [https://github.com/rkirana/kdd2014/archive/master.zip]
- Set the variables in main.R correctly current_working_dir: Set to the current working directory where you place the above source files scikitLearnPath: Set to the installed location of the scikit-learn development version xgboost_path: Set to the XGBoost installed path
- The parts of speech features are painful to compute. So you may pick them up from here: https://www.dropbox.com/sh/fu9sx3jrdvtirlg/AAC_SrEeThn4-SxJBK2IsSzia
- Copy the files of the competition and unzip them to the same folder as the source files i.e. to the current_working_directory above
- Run source ('main.R', print.eval=T, echo=T) to run the files

7. Additional Comments and Observations

The following were key insights:

- Some of the recently posted projects did not have sufficient time to be interesting projects. So applying a penalty to recent projects in the final stage helps in improving the score by around 0.002 to 0.003
- The text features - i.e. the essay content, the title, description of the project were not very useful in prediction - but were useful for ensembling models
- Part of speech features were useful
- Time of the year is an important feature
- Some donors are likely to donate more than other donors
- The location of the school requesting donation is important as there are people who like to donate in a specific region

8. Simple Features and Methods

The factor variables and the shrunken averages worked very well in predictions. This means that the categorical features contained very powerful information in them

9. Figures

NA

10. References

- Vignettes of the following packages (data.table Matrix glmnet doMC foreach rbenchmark Metrics gbm RRF lme4)
- Wiki of vowpal wabbit
- Wiki of xgboost
- Scikit-learn Wiki
- NLTK Wiki