A Comparison of Statistical Learning Methods for Regression

OLS, LASSO, bagging: A Comparison

Andrew Bates

December 20, 2018

Executive Summary

This is the executive summary.

1 Introduction

what is the problem? why do we care about it? reference Moneyball somehow. if a franchise can accurately predict an appropriate salary for a player they wish to acquire (accurate estimate a players 'worth'), they will have an advantage over teams that make an offer (is this how it works? or do teams just pick players and players don't have a choice?) based on more traditional methods. Or, if an MLB franchise can accurately estimate a given players 'worth', that player will likely be more satisfied with their position and renew their contract with the team, a situation that benefits both parties. In this paper we analyze data on players from the 1986 and 1987 Major League Basebal seasons with the aim of building a model that provides accurate predictions of a given players salary. To that end (?), we examine three statistical learning methods to determine which is more suitable for the problem at hand. Or, in order to achieve this goal we investigate three clases of models and provide a comparison of each models performance.

2 Methods

a paragraph about the data. where does it come from, how was it collected. this data set contains 322 observations. describe a few of the variables: There are 20 variables included in this data set and can mostly be broken down into two categories: performance metrics for the 1986 season (number of at bats, number of home runs, etc.), and performance metrics based on a given players career (career RBIs, career hits, etc.).

a paragraph describing the methods? for example, a brief overview of lasso an bagging for those less familiar with them? maybe put this in appendix and just reference it? then i could use equations! what exactly am i going to use with bagging. the obvious candidate is random forests. and in reality, this would be the easiest to do. it might be nice to add in an additional method or two. e.g. boosting, rule fit. i don't think this will be much more work. the real work is going to be eda and feature engineering. once the actual model fitting setup is done, the rest will be fairly automatic.

a paragraph about feature engineering? this would probably be good to have. but keep it generic. for skewed variables, we performed log transformations (or something else like box cox). we also derived several variables. for example, hit percentage (hits / at bats), home run percentage (home runs / at bat and home runs / runs). don't list them all but just give an example or two. have a table in the appendix which list all the variables used for model fitting. don't forget to mention handling of missing values. should they be imputed first, and then do eda? or eda, impute, eda?

a paragraph on model fitting procedure. i want to put each algorithm on the same playing field as much as possible. should i do a train/test split? the data set isn't very big so is it worth it? and 80/20 split is 257/64 and a 90/10 split is 289/32. it would be nice to do this b/c then we can get an idea of generalization error and have a final method to compare the methods. we could compare by cross validation error but it would be nicer to compare on out of sample performance.

i think i probably will do a train/test split, maybe 90/10 or even 85/15. for each model class, do cross validation (repeated?) to come up with a final model for comparison with the other methods. then the 'winner' will be the one with the lowest test set error. what to use for loss metric? i'm thinking rmsle b/c i think with the log in there it penalizes small and large errors more evenly (reference). salary is likely skewed so this would be a better metric than rmse. don't forget a final software summary b/c Levine will probably want that.

for regression, should we worry about linearity? if this was a 'data modeling' approach then definitely. but if we worry about linearity, then we are probably limiting the performance potential of the other methods. also, there are 20 variables in the data set, and i will probably make at least 5-10 more. this will be too many to look at to try to force linearity. and forcing linerity will likely mean subsetting the data set. this reduces the performance potential even more. i think the best bet is not to worry about regression assumptions. just treat it as a learning method vs. a 'data modeling' method.

3 Analysis

3.1 Exploratory Analysis

first talk about missing value imputation. explain what method and why. also explain why even impute. if the data set was bigger, might consider just removing them but in this case i think we need to impute them.

3.1.1 Feature Engineering

look at plots of all the variables. for skewed ones (salary for sure), do transformations. log would be simplest but also consider box cox and look into other transformation methods. should we look at correlations? we don't really need to be concerned with collinearity here do we? again, don't need to go into a ton of detail here but highlight a few variables, especially salary and any extreme cases. put a table in the appendix listing out all the final features and mention it here.

3.2 Modeling Fitting

explain model fitting method. for each method, k-fold cross validation was performed with the final model being the one that minimized the mean cross validation error. We used rmsle as a performance metric because it penalizes large and small errors more evenly compared to rmse. don't forget to discuss hyperparameter tuning. for liner regression, the only real hyperparameter is the number of variables. within each cross validation run, maybe we can do some sort of stepwise regression variable selection. how would this work in code? can this be done with recipes? can this be done with mlr? will i have to implement a custom method via MASS? for the others, does recipes handle random search or only grid search? can it even do hyperparameter tuning? (i assume yes). maybe list out the the range of hyperparameters considered for random forest. with lasso, would it be ok to extend this to elastic net or does it have to be strictly lasso?

3.3 Prediction Results

4 Conclusion

limitations: the most obvious is the timeline. the model constructed here should clearly (?) not be used to estimate player salaries for the next season (2019). besides the obvious issue of inflation, player salaries have experienced an inflation rate beyond that of the overall inflation rate (reference). Additionaly, elite level athletes have generally seen a tremendous growth in performance (skills, ability) over the last 32 years (reference). The game itself has also experienced an evolution (changes) that may have an impact on model performance. From rule changes, increases in overall althletic performance, and changes in player population, among others, the game of baseball has surely changed since 1986. That is not to say that the model developed here is useless however. We just want to caution the reader that one should not expect exceptional performance if this model is applied to a current MLB season (although that might be an interesting experiment). The usefulness in this analysis comes from the overall model construction procedure. This gives future analysts a starting point to work with.

Another limiting factor of this analysis is in the available variables in the data set. Baseball statistics has improved vastly over the last 32 years (reference) and there are substantially more metrics available currently (reference). This will no doubt improve the likelihood of providing the model with the ‘right’ features; features that could have a major impact on model performance. Often times predictive modeling is more about coming up with (obtaining) appropriate features than a particular model (reference). another thing that might help is player position and stats associated with various positions (i’m thinking pitchers here)

If future analysts deisire to construct a model for estimating contemporary MLB player salaries, we recommend the following approach.

A Supplementary Material

B R Code