Comparing the Effectiveness of Supervised Learning Methods in Heterogenous Data

An Analysis

Phillip Igoe, Andrew Strelke, Richard de Groof, Girish Parameswarappa, Anuj Gambhir

CIS602-603

Spring 2017

***Abstract*—Machine learning is a branch of statistics that allows us to employ supervised and unsupervised learning methods to data set. The goal of our project is to compare the effectiveness of supervised learning methods, and learn more about the various methodologies along with how effective they are when compared with one another. The metric used to compare the methods is mean squared error, MSE. Four data sets were analyzed: adult data, forest fire, student performance, and wine quality. The datasets were pre-processed such that these regression methods could be employed on them. By using linear regression, lasso, decision trees, random forest, and SVM regression, we can train these models on these datasets. Each method was implemented using either Java, R, Python, or Java. Our results are displayed in both table and graphical format, with all our MSE values displayed as the square root of MSE, for clarity. Conclusively the decision trees, SVM, and random forest seem to perform a bit better, with linear regression generally performing the worst. Future work can be conducted with more data sets, or more methods to run on the data in order to broaden our results and provide more insight as to what methods work better on different data sets.**

***Keywords—machine learning, regression, R, Java, Python***

# Introduction

Machine learning is a branch of statistics which allows for the processing of data, in order to find patterns for predicting future data. One of the most used practices of machine learning are supervised learning methodologies, which are used to construct predictive models. Supervised learning methods, such as linear regression and decision trees, are generally simple to use and effective in a wide range of applications. However, not all supervised learning methods are most optimal for certain situations, and some may even require improvements to be made to the methodology in order to become practical. Linear regression models, for instance, are known to work best for classification problems using linear data and/or with monotonic constraints, whereas decision trees are not preferred unless utilizing a order-preserving tree-generation algorithm.[1] By comparing the effectiveness of supervised learning methods, we intend to learn more about various learning methodologies through implementation.

The objectives are to select the methods, select the data sets, implement the methods and run on data as well as simulated data, then discuss and investigate the results. In evaluating these methodologies, we will understand their strengths and weaknesses being applied to actual data sets. This may lead to potential improvements in the techniques, but certainly an understanding of the methodologies themselves.

# Selected Data Sets

## Adult Data

The adult data set is a collection of data which predicts if adults make more than $50,000 per year based on 14 attributes, which are both categorical and (continuous) integers.

## Forest Fire

Forest fire data contains 13 different attributes describing the area of fire burned. The response variable (area of fire burned) is continuous, and there are 517 instances total.

## Student Performance

The Student performance data is a collection of attributes detailing student performance, as well as three grades assigned.

## Wine Quality

The wine quality dataset is split into two sets: white wine, and red wine. It details the quality of wine via classifying them into categories, with 13 attributes and 178 instances total.

# Methods and Simulation

## Linear Regression

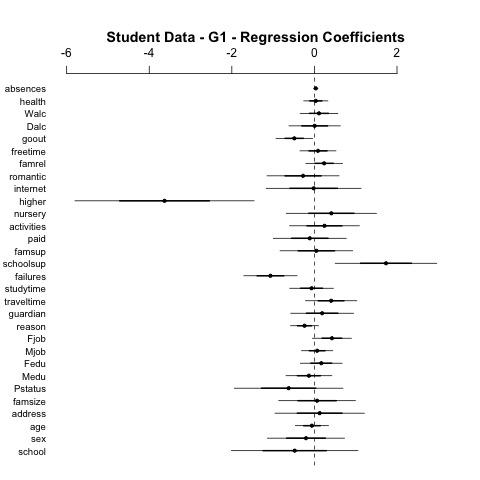
Linear regression is a method to create a model between a response variable, and multiple predictor variables. It is a technique that can used to predict data; in this way, it is a supervised learning method. Linear regression provides a ‘best fit’ line that establishes the relationship. Linear regression can also be extended to utilize multiple predictors, and still map to a response variable. This is also called multiple regression. Multiple regression fits the data to the model, with the response *Y* is a result of the weighted *X* dependent variables as regression coefficients [8].



*Figure 1 - Multiple regression as modeled by Graham (source: Graham 2003)*

Figure 1 depicts how the regression coefficients have a direct impact on the response variable *Y*. An optimal solution would strive to build the model with as little variables possible [8]. As such, we are able to apply linear regression, or multiple regression, to the data sets we have selected and attempt to get the best fitting line by calculating the regression coefficients.

There are several use cases where linear regression has been able to be applied in the past. In one case, Verwer et al. describes how it was implemented in order to determine the best order for items at an auction. As such, a regression model was used to map the variables in order to attain a solution yielding the greatest profit by predicting expected revenue [6]. In another instance, it was also applied to predict monthly rainfall using six attributes with Bagirov et al, and utilized an interesting concept of clustering linear regression in order to get the best fit. These models utilized all inputs values, much like our linear regression implementation, in order to development the rainfall prediction models [7].

Linear regression is easy to visualize when there is just one predictor and one response variable - Figure 1 visualizes a simple regression line. However, when there are three or more predictor variables, as is the case with every single one of our 

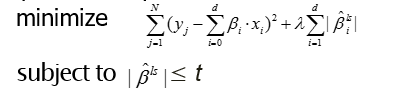
*Figure 2 - An example regression coefficient plot for the student data set*

datasets, it is not easy to visualize the corresponding regression.

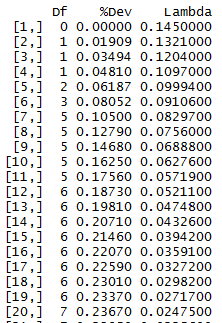
The solution we devised was to visualize the regression coefficients on a plot. Figure 2 depicts the regression coefficient plot for the student data set, with each of the dependent variables being plotted to show the impact they have (as coefficients) on the response variable.

The pseudocode for linear regression implemented with R is included in the appendix.

## Lasso

Lasso was initially designed for the least square model in linear regression analysis. The application of Lasso in linear regression has to do with minimizing the Sum of Squared Error (SSE), also known as Sum of Squared Residuals (SSR), This is achieved by using the L1 regularization:

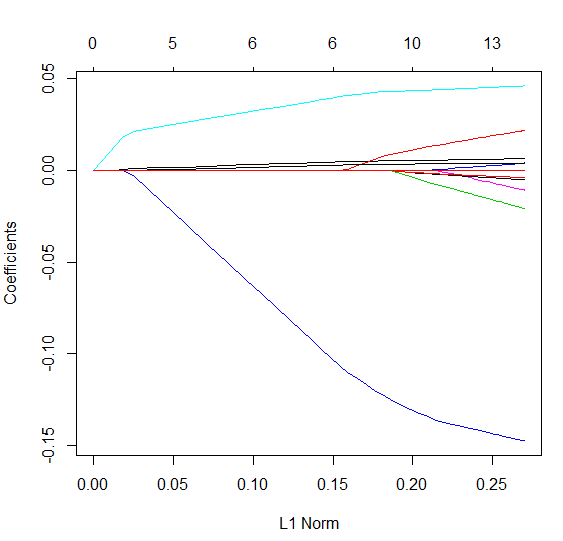
where lambda (λ) is the tuning parameter that controls the amount of regularization, and Bi are the d Lasso coefficient estimates.

Once the implementation for the standard Lasso model had been completed, test were ran for the actual and simulated data for the selected datasets mentioned previously in section III. The model used was the Gaussian linear model, which is the same as the least squares model mentioned earlier in this section. The actual datasets, as well as the simulated dataset generated beforehand, were loaded directly to a linear model with 75% of the data used for training and the other 25% for testing. In order to accomplish this, input matrices for the predictor variables x were constructed, whereas the response variables y simply contained the targeted attribute of the given dataset. The regularization path are computed by a fitting function as a table of data containing the values for the 

*Figure 3 - A summary of the first twenty steps of the regularization path for the adult dataset*

fitted models’ degrees of freedom, percent of deviance explained, and the values of the tuning parameters lambda.[10]

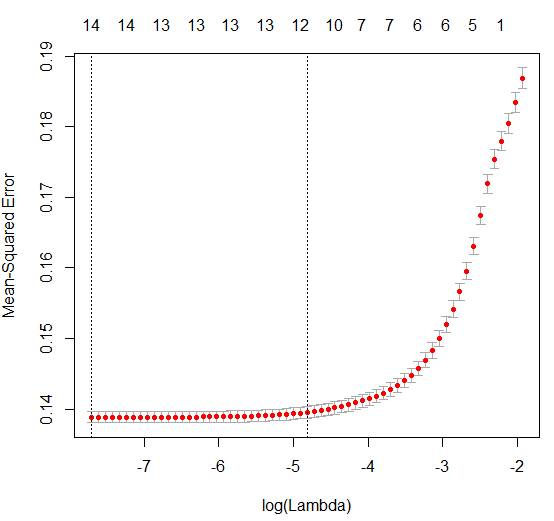
Although a table is useful for viewing the regularization path, we can gain a much better understanding of the lasso coefficients by plotting them into a graph as seen in Figure 4 below.



*Figure 4 - L1 Coefficient Trace Plot*

Each colored curve represents a different predictor variable, whereas its path correspond to the value of its coefficients against the regularization parameter lambda (L1 Norm). The axis above the graph demonstrates the degrees of freedom, or nonzero coefficients, at the given L1 Norm parallel to it. If the L1 Norm equals zero, we obtain no shrinkage of coefficients, and hence obtain the least squares solution as our solution. As it increases, however, the regularization parameter sets the linear coefficients closer to zero. This visualization clearly depicts the variable selection portion of the Lasso method, as it demonstrates how the tuning parameter affect the values of coefficients in the model.

Based on resulting data for the fitted models, we must then select the most optimal one. Cross-validation is known to be the most simple and effective method for estimating the most optimal value for the regularization parameter; particularly k-fold cross-validation. The generated plot, shown in Figure 5 below, extracts the mean squared errors of the fitted models and highlights the optimal lambda values selected by the k-fold validation.



*Figure 5- K-Fold Validation Plot*

The line to the left specifies the tuning parameter that results in the minimum MSE, whereas the tuning parameter shown by the line to the right defines as the maximum regularized model. In order to satisfy the objective specification of this research, the lambda value producing the minimum MSE was selected to generate the predictions. Once the predicted values have been generated, we can finally calculate the simulated MSE. This process was conducted on each of the actual and simulated datasets to evaluate the performance of the Lasso methodology across various applications.

## Decision Trees

Decision trees are non-linear learners that work well when there are complex interactions between predictor variables.

Figure 6 shows the general process. Regions of interest are divided according to the splits of the decision tree. Figure 9 shows the decision tree trained on the same data set (iris data set). Triangular entities depict nodes with children (vertically arranged connections), with the circles representing leaf nodes or entities without children. Each data sample follows the tree according to its value at each parent node in the tree. For example, at the start of the tree, a data sample with attribute 2 (zero indexed) having a value less than or equal to 2.35 will be classified as zero.

Otherwise, the right child of the first entity is compared and the classification follows the tree until terminating at a leaf node, the final classification for that sample.

This model may be applied to discrete (classification trees) or continuous (regression trees) data. We found, however, that on our data sets, the best results were found by assigning numeric values to each nominal or ordinal category.

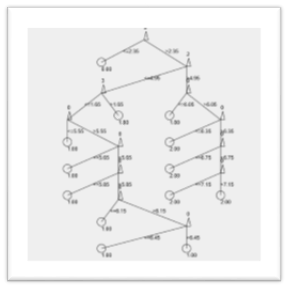
Decision and regression trees have been studied for their ability to produce accurate determinations of response variable given predictor variables. David Lohrmann et al. applied regression trees to longitudinal data encoding ordinal and categorical data into discrete variables with

Fig. 7. Decision Tree Depicting parent/child relationships

some success [2]. John W.T. Lee also studied decision trees for the interpretation of ordinal data [1].

In addition to providing a continuous output, regression trees may be used to indicate the relative importance of predictor variables. Jeffrey Kiline et al. used regression trees

.

to narrow down deciding factors in the assessment of heart disease risk [3].

Indeed, the decision tree is built using a measure of information gain at split, with the highest information gain occurring at the uppermost splits. This may result in insight into the data.

For example, in figure 7, we see that the value 0.0 is assigned at the first split and, for the most part, additional classifications are assigned according to the next split only. The computation is very fast, comparing just two out of four attributes and is fairly robust.

Variations of the standard regression tree model have been explored to boost accuracy. Such examples include boosting [4], bagging and random forests [5]. These techniques essentially grow multiple trees, either sequentially as in boosting where trees are combined on a subset of the predictors, or entire trees trained on a subset of the data.

Some of our best results were obtained using regression trees. Figure 8 shows the adult data set classified on a regression tree with at most 100 nodes. The first split on attribute 8 (zero indexed) is very accurate at making predictions. With little loss of precision, the tree may assign <= 50k (approx. 0) to the left branch and > 50k to the right branch (approx. 1).

These data were obtained using a GMM simulation method. Analysis of this attribute in the original data set is not as revealing, with less of a sharp contrast between instances. Figure 9 shows the distribution of classes across this particular attribute value. There is less of a clear division, with both classes shown throughout the plot.

Indeed, the fitted tree across the non-gmm simulated data has a much more complex tree (see figure 10) when compared with the tree for the gmm simulated data (see figure 11). The non-gmm simulated fitted tree has a greater unbalancing of classes between splits, while the majority of the splits in the gmm simulated tree classify as 1.0.

Why does the gmm-simulated data produce so much more accurate results? In figure 12 there is depicted a two-dimensional binning process. The data are separated according to being above or below the average of each attribute. This is, essentially, a clustering process. Data are well separated as is visible in the screenshots. We interpret this to mean that the data is bi-modal in two dimensions and, therefore, the gmm process produces well-defined clusters. The decision tree takes advantage of this bimodality, as the basis of each split is in two directions.

## Random Forest

Random forest or random decision forests are an ensemble learning method for classification and regression and other tasks, it classifies decision trees on various sub-samples of the dataset and for regression it uses averaging to improve the predictive accuracy and control over-fitting.

Random forest also provides feature importance scores for each predictor variables and also allows us to evaluate any chances in correct classification or regression with the growth of small and large number of trees.

The **Student performance dataset** provides a great example in implementing the Random forest along with the measure of importance of predictor variables and measure of various internal structures of data i.e the different data points nearer to one another.

Measuring variable importance: **Dn = {( Xi Yi)}ni=1**

When we create a model using the **ANOVA**(Analysis of variance) which is a collection of statistical model used to test differences between two or more means basically analyzing variance and OLS(Ordinary least squares) function for predicting using the response variable. From the **Figure 13** we can see that when we run the Student-performance dataset on the random forest model we get the importance of the predictor variables and how it affects the overall performance and accuracy of the model.

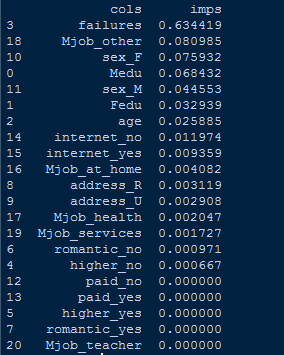
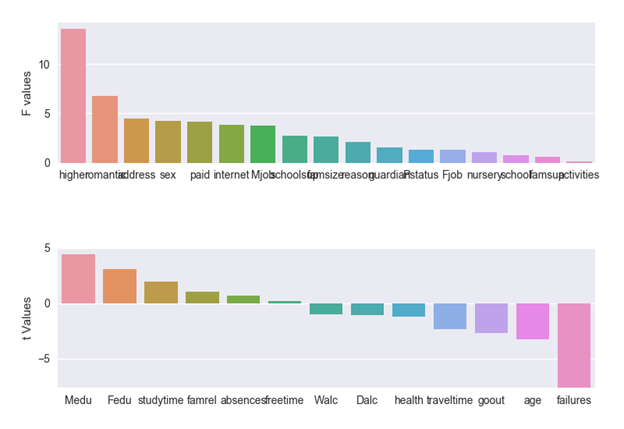


Figure 13

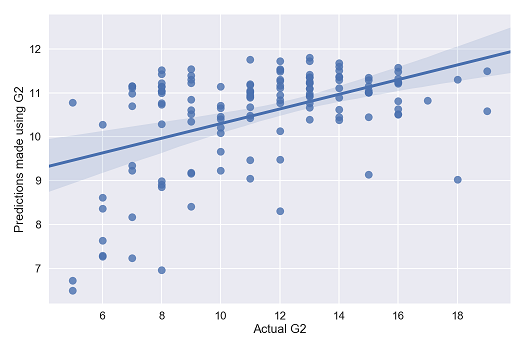
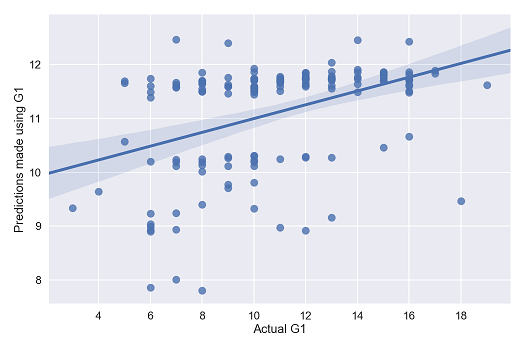
When we plot the graphs to check the Root Mean squared error(RMSE) on all the response variables using the feature importance we got different variations in the graph.

From the above graphs we can see the different predictions using different response variables(G1,G2,G3) we can see good prediction values using response variable G3 and when run on simulated sample data sets generated(GMM x 10

Please refer the RESULTS section for all the prediction values and method comparisons. All the implementations were done in python using sklearn, sea born, matplotlib libraries.

Figure 14

**Figure 14** describes how the predictors are divided iteratively into two types of list i.e one by F values means the values with data type which are qualitative whereas the values with t values contain the quantitative values from the regression. In the above graph we can see that failures goes to the -ve values of regression whereas the Mother’s education level has the highest +ve value of importance.



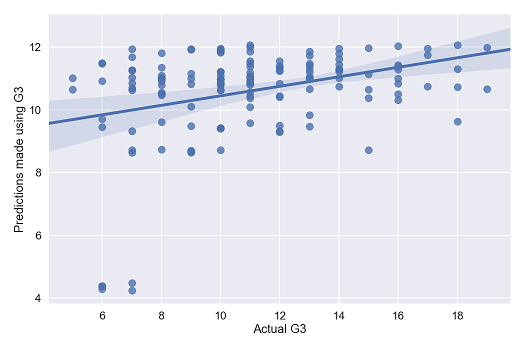


Figure 15

## SVM Regression

Support Vector Machines are very specific class of algorithms, which is characterized by the usage kernels, absence of local minima, sparseness of the solution and capacity control obtained by acting on the margin, or on number of support vectors.

Support Vector Machine can be applied not only to classification but also to the case of regression. Still it contains all the main features that characterize maximum margin algorithm, a nonlinear function is learned by linear learning machine mapping into high dimensional kernel induced feature space. The capacity of the system is controlled by parameters that do not depend on the dimensionality of the feature space.

The quality estimation is measured by the loss function ***L(y,f(x,w))****.* SVM regression uses a new type of loss function called *e*- insensitive loss function proposed by Vapnik:



The empirical risk is:



The measure of quality that we used in order to compare our methods on different datasets is by calculating the sum of root mean square errors (RMSE) and we’ve got some good and bad results for different datasets when applied to different types of regression techniques

The following Figure 16 shows a plot between the predicted values of student G1 results and the actual values of student G1 results. The SVM function models/trains the values based the actual values in Student$G1, and predicts the values.

model <- svm(G1~.,data=student)

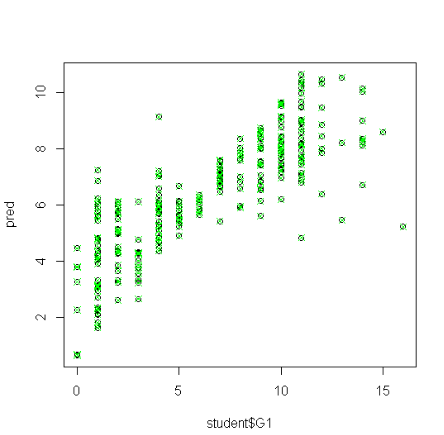
pred <- predict(model,student)

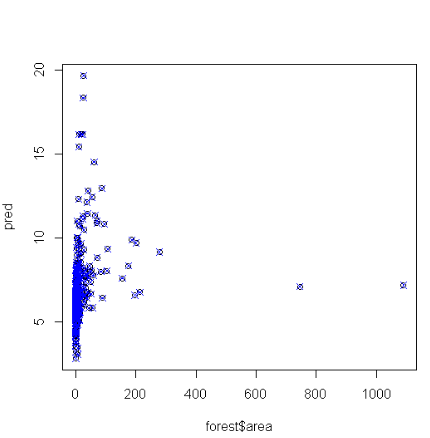
Figure 16 shows the graph between the predicted results and actual results for G1 in student dataset.

Figure 17 shows the predicted area in case of forest fire

Here we can see on the x-axis when the actual value of Student$G1 is 10 , the predicted values shown in the green dots are predicting the values to be somewhere between 7 to 11, which is quite accurate considering the amount of errors.

The following graph is for the forest fire dataset, where x-axis has the predicted area affected under forest fire V/S the actual area affected in the forest fire on the y-axis.

Here, consider a small blue dot at the extreme right of the graph, which is beyond the 1000 unit square mark, this means that the SVM is predicting that in case of forest fire it is less likely to have fire in more than 1000 unit square and more likely to have forest fire to occur between 0 to 200 unit square of the area.

Most of the fires were actually very small or negligible as compared to one or two big fires, thus here we get a proper prediction of the area affected by forest fire.

## Simulation

1. *The GMM Methodology:*

Data were sampled using the Gaussian Mixture Model methodology. A Gaussian mixture model assumes multiple separate Gaussian distributions occurring within the same dataset. Figure 18 below shows how the technique is applied to clustering.

According to the Gaussian model, some probability can be associated with data points lying within the “Bell curve” (see figure 19).

Parameters for the normal distribution are learnt according to the Expectation Maximization algorithm (see figure 20). This process tunes the values for the mixture model according to the data set. As part of the process, the mixture model contains mixing proportions, which specify the proportion of each class in the representation. To simulate the data, we use these training proportions to produce data sets composed of each class by their respective amounts.

Then, learned parameters mu and sigma can be used to generate data from a normal distribution.

### b) Preprocessing

For the assignment, we used the GMM methodology to generate data according to the mixing proportions, variances and means of the learnt multi-modal distribution. Generally, the data had to be in continuous form as input for the different methods and for the GMM simulation.

In order to translate categorical to continuous data, we used binning. Data were assigned the respective index of each category of each feature.

For some features, this made sense as they were ordinal variables and the differences between categories could be more accurately assigned an index. Understanding the relative importance of the variables, the bins could be adjusted to encompass more than one value. This could prove to be an enhancement in future work on these datasets and methodologies.

# Results

After implementing the methods on the described data sets, the following results were computed:

1. Results of Simulation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Data Set** | **Method Results (Square root of MSE)** | | | | |
| ***Lin. Reg.*** | ***Lasso*** | ***Decision Trees*** | ***Rand. Forest*** | ***SVM*** |
| ***Red Wine*** | 0.6433392573 | 0.6550395 | 0.883 | 0.5074613182 | 0.55455 |
| ***Red Wine (SIM.)*** | 1.663245021 | 1.6762813 | 1.172 | 0.5083753014 | 1.17472 |
| ***White Wine*** | 0.7600101973 | 0.5765167 | 1.09 | 0.5069965457 | 0.6327606 |
| ***White Wine***  ***(SIM.)*** | 1.853837372 | 1.856057 | 1.35 | 0.5045478751 | 1.525 |
| ***Student Data G1*** | 3.867275527 | 3.52204 | 1.703 | 3.0778172 | 2.177 |
| ***Student Data G1***  ***(SIM.)*** | 6.032048574 | 4.252725 | 5.22 | 0.5154828843 | 1.887 |
| ***Student Data G2*** | 4.124821208 | 3.945564 | 1.51 | 3.5790346 | 2.5 |
| ***Student Data G2***  ***(SIM.)*** | 6.080853559 | 4.178557 | 4.78 | 0.5063497866 | 1.919 |
| ***Student Data G3*** | 5.412973305 | 4.391984 | 5.95 | 3.884055 | 2.52 |
| ***Student Data G3***  ***(SIM.)*** | 6.216009974 | 4.26506 | 5.05 | 0.5084449701 | 1.879 |
| ***Adult Data*** | 0.4868583983 | 0.373415 | 0.325 | 0.505645248 | 0.3437 |
| ***Adult Data***  ***(SIM.)*** | 0.5345519619 | 0.331471 | 0.072 | 0.5243167622 | 0.1066 |
| ***Forest Fire*** | 63.92403304 | 1.400801 | 3.57 | 0.4949989303 | 63.63385 |
| ***Forest Fire***  ***(SIM.)*** | 3.728667322 | 3.535251 | 3.13 | 0.5000214671 | 2.37 |

Table 1 showcases our results. All values are reported as the square root of the MSE. Graphical representations of these results in the form of bar charts can be found in the appendix.

# Discussion / Conclusion

The given results demonstrate how these supervised learning methodologies work given their generalized form, as well as an overview of their predictive performance based on the data at hand. For each of the wine datasets, linear regression seemed to perform just as well as when the L1 regularization penalty is applied to it. Looking into the results for the Lasso simulations show that the optimal lambda selected was very close to zero, meaning that applying this method might not suite well given this dataset. However, the more advanced methods, such as decision trees, SVM regression, and random forest demonstrated much better predictive performance; especially random forest whose MSE for the simulated data nearly match that of the real data.

The results for the three student performance datasets also seemed to exhibit a consist pattern to one another. This time, however, Lasso performed much better than linear regression with values further away from zero being selected for the optimal lambda. Again, SVM regression and random forest produced to most desirable results for the student datasets, but this time it was SVM regression that proved as the most optimal solution. It should be noted, however, that random forest was able to perform much better than any of the other supervised learning methods in terms of the simulated data, but the results for the real datasets scored much higher MSE when testing. Therefore, what is believed to be occurring is an inherent multi modal distribution to the data and method, due to the fact that the Hidden Markov Model (HMM) used in the random forest approach assumes two or more gaussian distributions in the data. Bi-modality, however, is implied on the real student performance dataset. As a result, the HMM method might cluster and generate the simulated data well compared to that of the real data. Decision trees, on the other hand, resulted in worse predictive performance than that of the other advanced methods, and even Lasso. In addition, the average MSE calculated for the simulated data typically scored much lower than when fitting the real data to the model.

For the adult dataset, Lasso again scored much better predictive performance than that of linear regression; proving itself as a practical improvement to the linear regression methodology. Decision trees seemed to work best using the adult data, with SVM regression slightly tailing behind. Random forest, however, surprisingly did worse than any of the other methodologies, which further proves its effectiveness when bi-modality is implied to data. More specifically, random forest seems not to work well when values for the response variable are sparse, such as in binomial response variables.

Finally are the results for the forest fire dataset, which demonstrated some shocking results. For one, all methods seemed to score very low MSE when utilizing the simulated data. When using the real data, however, linear regression and SVM regression exhibited predictive performances that were much larger than what resulted from using the simulated data. The most efficient method was random forest, which resulted in MSE scores close to zero for both the real and simulated data. Lasso and decision trees seemed to work just as well, with Lasso only obtaining slightly higher predictive performance when using the real dataset.

In conclusion, the more advanced supervised learning methodologies such as decision trees, SVM regression, and random forest look as if they are the most suitable approaches for predicting outcomes of heterogenous data. Linear regression scored as the least optimal method overall, whereas Lasso fulfilled its purpose of minimizing errors in the least squares model. Although the random forest method is typically capable of producing desired predictive performance, modality of the data must be considered before deciding to commit this approach.

##### **Appendix**

Pseudocode for implementing linear regression in R:

*Import the data set*

*Set seed*

*Partition data set - 75% for training, 25% for testing*

*Using lm, train a linear regression model with the training data*

*Using ARM package, export a coefplot for regression model*

*Using the predict function, make predictions using the linear regression model and the test data*

*Compute MSE*

*Repeat above process for each of the 10 simulated data sets, computed average MSE by keeping a summation for each run, then dividing it by 10*

Pseudocode for implementing lasso regression in R:

*Import glmnet library*

*Import the data set*

*store filenames and number of files in arrays*

*for each file*

*Read file and store contents in table*

*Parition training and test data*

*Store predictor variables in matrix x*

*Store response variables in list y*

*Use cv.glmnet() to fit the model and perform cross validation to estimate optimal lambda*

*Plot the mean squared errors across each lambda value.*

*Use predict.glmnet() to calculate predicted values based on the selected fitted model, and store values in list*

*Calculate the mean squared error of the fitted model as: MSE = mean(y-)*

*Calculate average mean squared error across all simulated dataset as: mean(MSE)*

Pseudocode for implementing decision tree analysis in java:

*Import the non-gmm data set.*

*Partition data set - 75% for training, 25% for*

*testing.*

*Using smile machine learning library in java, train*

*a regression tree on the training data.*

*Using the same library, input testing values for*

*prediction.*

*Gather predictions and subtract from target value.*

*sum the error and output the MSE*

*repeat for each of the 10 simulated data sets and*

*output the average.*

Here are our results in bar chart format:

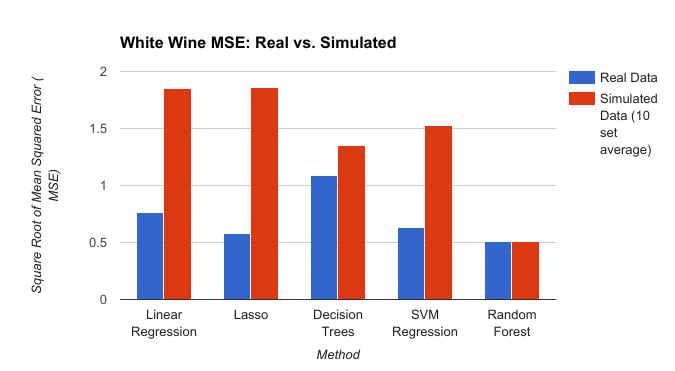
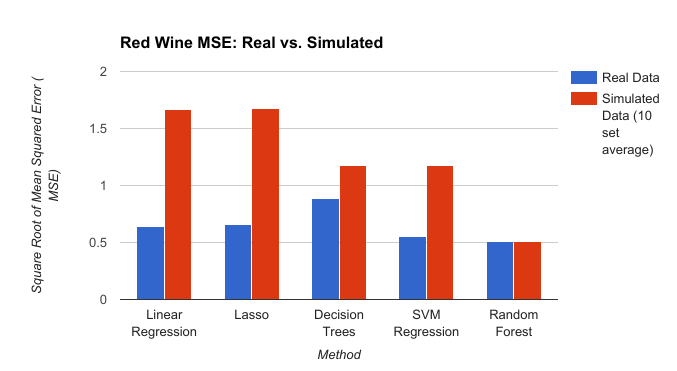
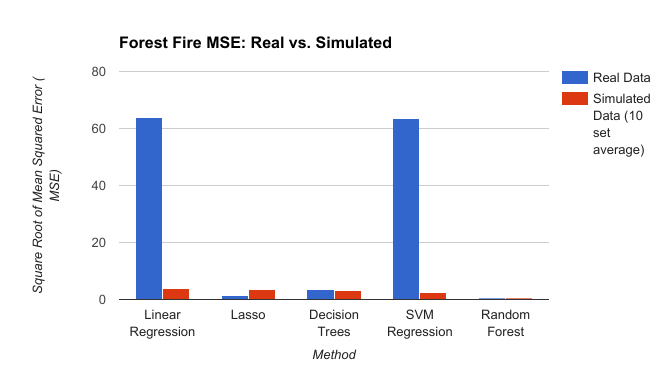
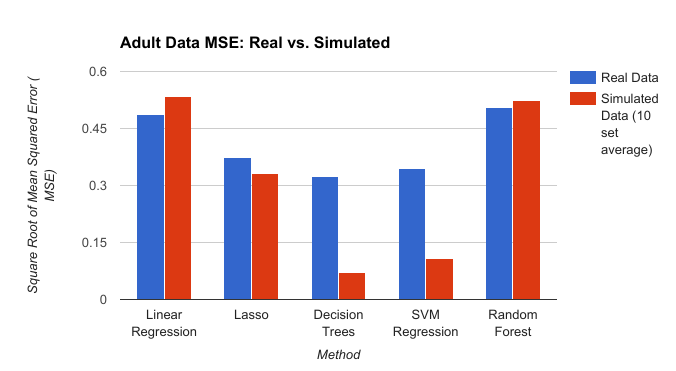
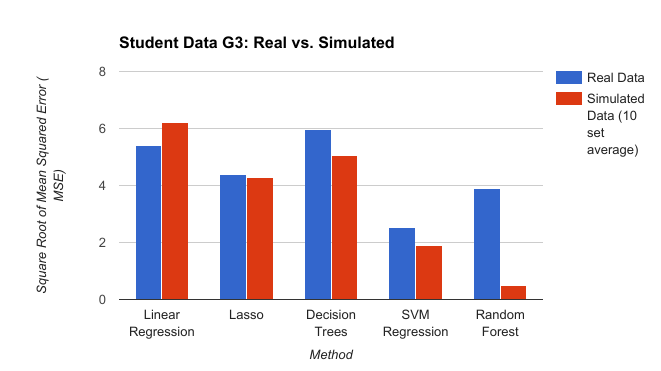
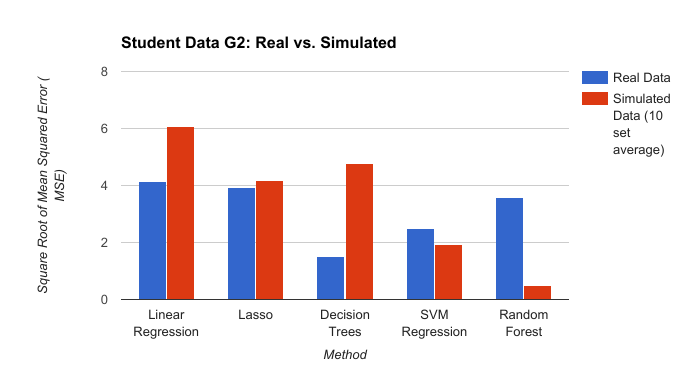
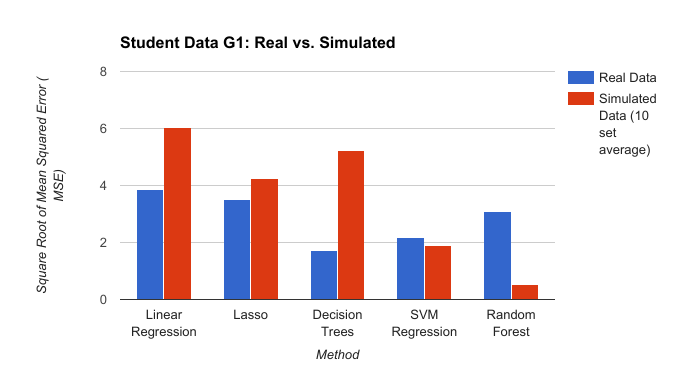
 

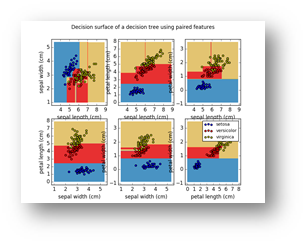
Fig. 6. Decision Tree Process (http://scikit-learn.org/stable/modules/tree.html).

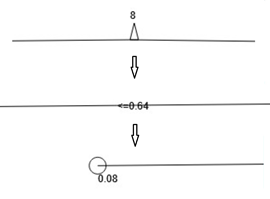
Fig. 8. Highly revealing single split

Fig. 9. Distribution of classes across one attribute value.



Fig. 10. Tree fitted on non-gmm simulated data.



Fig. 11. Tree fitted on gmm-simulated data.

Fig. 12. The data is well-represented by two-dimensional splits.

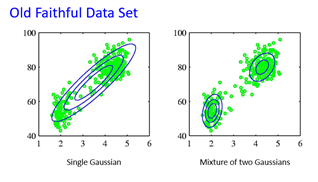
Fig. 18. The Gaussian mixture model.

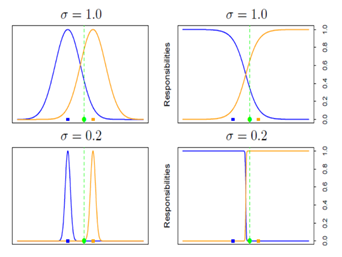
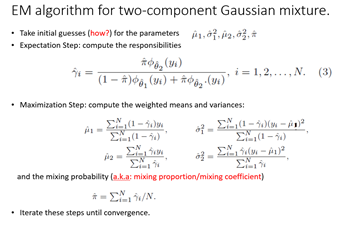
Fig. 19. Gaussian mixture model may assign data point a probability as being in one of two clusters.

Fig. 20. The expectation maximization method.

##### **Contribution**

In general, all members implemented their own method on the data sets. Below is a breakdown of the contributions:

**Phil**: linear regression, general group organization, project proposal

Paper: linear regression section, abstract, data sets, results

**Richard**: decision trees, project proposal, simulated data

Paper: decision tree section, preprocessing and simulated data

**Andrew**: lasso, project proposal

Paper: introduction, conclusion

**Girish**: random forest

Paper: random forest

**Anuj**: SVM

Paper: SVM

##### **Meeting Notes**

All meetings below were physical / google hangout meetings that were planned - toward the end, we touched base after class regularly. Lots of information exchange occurred over the group chat. Group chat transcript is available upon request, and can be provided by group leader.  
  
Feb 21, 1pm  
Richard, Phil  
- Met in TEX 101D, went over initial proposal, filled it out after picking out data sets + methods, discussed simulation data  
  
March 9, 3:30  
- Compared and consolidated hw #2 everyone attended (but not everyone attempted all hw problems)  
  
March 15, 4pm  
Richard, Girish, Andrew, Phil  
- Discussed project feedback, decided to stick to methods that are regression, decided to eliminate datasets to now include only adult data set, student performance data, wine quality. - Decided to split the work into one google drive document, with one student per method  
- Created google doc for splitting project work into methods, so everyone has one method  
  
March 22:  
- Announcement of official work split, updated revised proposal.  
  
April 5  
- Brief meeting after class, everyone attended  
- Decided to use a calendar to pace ourselves with milestones, so project can be completed. Calendar was a shared calendar with the different milestones  
  
April 6th and on, touched base after class (when people were present)  
- Brief meeting after class, everyone attended  
  
April 20  
- Meeting to consolidate hw #3 answers - Phil, Richard, Andrew had answers that were consolidated for hw #3.

##### **References**

1. Lee, John WT, Daniel S. Yeung, and Xizhao Wang. "Monotonic decision tree for ordinal classification." *Systems, Man and Cybernetics, 2003. IEEE International Conference on*. Vol. 3. IEEE, 2003.
2. Lohrmann, David, Wasantha Jayawardene, and Mohammad Torabi. "Factors Associated With Changes in Fruit Intake During Young Adulthood: A Classification and Regression Tree Analysis of Longitudinal Data." *Journal of Nutrition Education and Behavior* (2017).
3. Kline, Jeffrey A., et al. "Pretest probability assessment derived from attribute matching." *BMC medical informatics and decision making* 5.1 (2005): 26.
4. Robinson, John W. "Regression tree boosting to adjust health care cost predictions for diagnostic mix." *Health services research* 43.2 (2008): 755-772.
5. González, Camino, José Mira-McWilliams, and Isabel Juárez. "Important variable assessment and electricity price forecasting based on regression tree models: classification and regression trees, Bagging and Random Forests." *IET Generation, Transmission & Distribution* 9.11 (2015): 1120-1128.
6. Sicco Verwer, Yingqian Zhang, Qing Chuan Ye, Auction optimization using regression trees and linear models as integer programs, Artificial Intelligence, Volume 244, March 2017, Pages 368-395.
7. Adil M. Bagirov, Arshad Mahmood, Andrew Barton, Prediction of monthly rainfall in Victoria, Australia: Clusterwise linear regression approach, Atmospheric Research, Volume 188, 15 May 2017, Pages 20-29.
8. Graham, M. H. (2003), CONFRONTING MULTICOLLINEARITY IN ECOLOGICAL MULTIPLE REGRESSION. Ecology, 84: 2809–2815.
9. Fan. Jianging, Liu, Runze, *Variable Selection via Penalized Likelihood*, Department of Statistics Papers, 14 June 1999, Pages 1, <https://escholarship.org/uc/item/8j29393d> (Ill source this later)
10. [https://www.rdocumentation.org/packages/glmnet/versions/2.0-5/topics](https://www.rdocumentation.org/packages/glmnet/versions/2.0-5/topics/glmnet)
11. <http://www.bios.unc.edu/~dzeng/BIOS740/randomforest.pdf>
12. <http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>
13. <http://stackoverflow.com/questions/19991445/run-an-ols-regression-with-pandas-data-frame>
14. <http://seaborn.pydata.org/generated/seaborn.lmplot.html?highlight=lmplot#seaborn.lmplot>
15. <http://kernelsvm.tripod.com/>
16. <https://www.mathworks.com/help/stats/understanding-support-vector-machine-regression.html?requestedDomain=www.mathworks.com>
17. https://en.wikipedia.org/wiki/Support\_vector\_machine