THE RISE OF THE BIG DATA: WHY SHOULD STATISTICIANS EMBRACE COLLABORATIONS WITH COMPUTER SCIENTISTS

by

XIAO CHENG

(Under the Direction of Jeongyoun Ahn)

ABSTRACT

Big Data has been the new trend in businesses. As technology advances, the ability to collect large amount of data and learn insights from them became more important. We will analyze a dataset provided by the Heritage Provider Network. The goal is to predict the number of days a patient stays in the hospital based on previous year's insurance claim records. The tools used include logistic regression, Classification and Regression Trees, and Gradient Boosting Machine (GBM). We will compare two techniques of analyzing the data. The first method will use GBM to predict the outcome variable. While in the second method, we will use a sequential modeling method, where we partition the dataset into risky and non-risky groups, and then use GBM to predict the outcome for the risky patients and assign no-stay for the non-risky group. We also discuss Big Data topics such as More Data Beats Better Algorithm.

INDEX WORDS: Big Data, Machine Learning, Decision Trees

THE RISE OF THE BIG DATA: WHY SHOULD STATISTICIANS EMBRACE COLLABORATIONS WITH COMPUTER SCIENTISTS

By

XIAO CHENG

B.S., The University of Georgia, 2013

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2013

THE RISE OF THE BIG DATA: WHY SHOULD STATISTICIANS EMBRACE COLLABORATIONS WITH COMPUTER SCIENTISTS

by

XIAO CHENG

Major Professor: Jeongyoun Ahn

Committee: Liang Liu

Pengsheng Ji

Electronic Version Approved:

Maureen Grasso

Dean of the Graduate School

The University of Georgia

December 2013

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

Technology has changed the way we collect and store data. In 2012, IBM published a study that said ninety percent of the data we currently have were created within the previous two years [1]. Big Data has been the new trend in the business world the past few years. It has been and will continue to be a key phrase that shows up in magazines and newspapers.

Every credit card transactions and every customer service conversations are being kept as data sources for businesses today. Furthermore, social networks such as Facebook, Google, and Twitter have created new platforms for companies to collect information. The amount of data will continue to grow exponentially. As the advancement of technology, more sophisticated computational tools are being developed to analyze these data and collect insights in order to make better decisions.

Traditional statistics involves designing experimental studies, collecting samples, and analyzing the dataset. This process often takes a long period of time. As technology advances, data scientists started to ask innovative questions such as how can we help customers improve their experiences, or how can we reduce the cost of providing a service? The Big Data revolution is the path that helps businesses to begin answering these problems.

The website Kaggle.com is reinventing the wheel on giving data scientists the opportunity to build algorithms and perform against each other. As the sponsors would say "We are making data science a sport". There are numerous competitions on the site ranging from job salary predictions to an All State Insurance sponsored auto-liability claims. However, the biggest of them all is the Heritage Provider Network (HPN) sponsored competition on predicting how many days a patient will stay in a hospital based on his or her insurance claims information from the previous year.

This competition started in April of 2011 and it ran for two years with a prize of $3,000,000 for the grand prize winner. In the past few years, the health care reform has become an important

topic in the United States because of the raising cost in health plans. Some of the important goals of the competition are to learn new ways of effectively managing hospital resources and providing patient care services in a timely manner.

Analysts across the world and various disciplines such as computer science, statistics, medical science, and business all came together to build and improve algorithms in order to compete for the grand prize. The goal is to have the public make informative models to predict hospital stay patterns and help HPN to allocate its resources in the future.

The focuses for Big Data are three V's, namely high velocity, high volume, and high variety. For this competition our focus is on volume and velocity. Volume indicates of the amount of data we can acquire and velocity is the speed of accessibility. High variety means different types of data other than the traditional database sources. Examples include unstructured text data, web-log files, or sensor based records. These data could be potentially very important, but in this competition, the dataset was presented by the traditional database. One hypothesis to support Big Data is that a huge data set with simple techniques beats a smaller data with more sophisticated analysis, which we will discuss further in Section 4.1.

The goal of many Big Data analysis is to detect patterns from the past dataset, and then make informed forecasting for future observations. However, regardless of the application, when building models, there will always be a tradeoff between gaining insights versus making better predictions. A simple model provides many helpful insights on a dataset, and interprets the result. However, these models will not have the best prediction error when it comes to predicting an unseen future dataset. On the other side, a very complex model will provide a better prediction error, but it comes with the cost that it is harder to explain the meaning behind the analysis. To improve our toolbox, we should keep in mind of this tradeoff. We want to improve both insights and predictions, but no model is perfect, hence our job is to build different models based on different goals, and then study the advantages and the disadvantages of each model.

Having understood the possibilities as well as the limits of building models with Big Data, it is time to test out the analysis on real world datasets. As the data became more complex, practitioners from different fields also started to work together to look for new techniques.

Machine learning is a product of such case, where computer scientists build newer algorithms based on statisticians' data analysis techniques.

With an interesting data set, there are more than a few possibilities to analyze it. The two intuitive tools are Random Forest and Gradient Boosting Machine. Both of these techniques are in the machine learning literatures that is based on tree building non-parametric models. Leo Breiman, Jerome Friedman, Charles Stone, and R.A. Olsen developed Classification and Regression Tree (CART), which is the original tree-building idea. The main point of the decision tree model is to split a dataset based on similar characteristics. Once the dataset is split, the process iterates again for both of the subgroups and continues until either there is no more data to split or that a further split will not help improve the model performances.

The Hacker Way is being utilized in the technology field, where data scientists would try numerous techniques and figure out what is the best solution while analyzing a model. We worked with a two-stage modeling procedure in this thesis. The idea behind this method is that the overwhelming majority of the data did not stay in the hospital. The regression methods of decision tree models have predictions of a continuous variable, where it will not produce a prediction of exactly zero. Hence, the idea of two-stage model is to partition a subset of patients who are unlikely to be admitted, and assign these records as no-stay. After this step, we will run Gradient Boosting Machine on the remaining subset.

The rest of the thesis will proceed as follows. Section 2 Data Description talks about the original variables and then describes the variables created to be used for the analysis. Section 3 Methodology describes the techniques used in the analysis. Section 4 Data Analysis goes in depth on the model building process. Section 5 discusses the technology trend and its effects on health care datasets. Section 6 concludes.

CHAPTER 2

DATA DESCRIPTION

The competition provided the following seven data sets listed in Table 2.1 with a brief description. Some data manipulation work was needed in order to get ready for the analysis. The following subsections will describe the work done for each data set.

Data Set Name	Description
Claims	All Claims data for Y1-Y3
DaysInHospital_y2	DIH for Y2 based on Patients with Y1 data
DaysInHospital_y3	DIH for Y3 based on Patients with Y2 data
DrugCount	Drug Count for all patients
LabCount	Lab Count for all patients
Members	Patients ID including age and gender
Target	The set of members used for the final prediction

Table 2.1: Original datasets description

2.1 CLAIMS

The claims data consists of Year 1 (Y1), and Year 2 (Y2) claims data. Y2 and Y3 Days In Hospital (DIH) records were also provided, which corresponds with Y1 claims and Y2 claims records respectively. Figure 2.1 displays the data structure and final prediction variable. To simplify the process, the claims data were partitioned by year and grouped into two sets which will be used to combine information from all other relevant data sets.

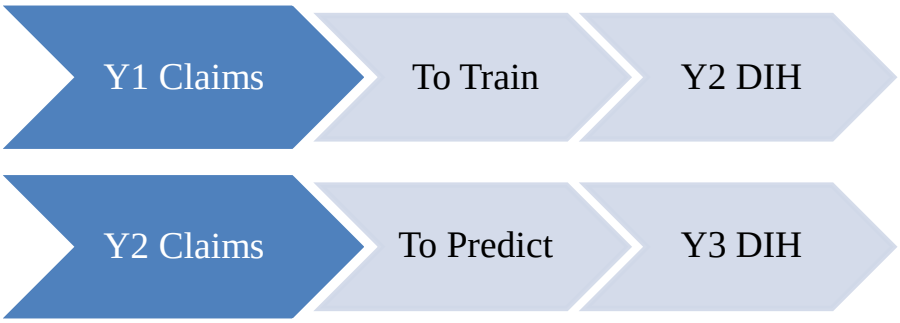

Figure 2.1: Competition data structure

The Claims data originally have the variables listed in Table 2.2. Each row of the data represents one claim record for a patient. Some members have multiple records while other members have only one record. To make the analysis practical, it was necessary to summarize the data with exactly one record per member.

Claims Variables	Description
MemberID	Member pseudonym
ProviderID	Provider pseudonym
VendorID	Vendor pseudonym
PCP	Primary Care Provider pseudonym
Year	Year of the claim
Specialty	Treatment
PlaceSvc	Care Service Location
PayDelay	Time of delayed payment after treatment
LengthOfStay	Time stayed in the care facility
DSFS	Date since first service
PrimaryConditionGroup	Condition prior to treatment
CharlsonIndex	Severity of condition
ProcedurGroup	Procedure performed during treatment
SupLOS	Whether LOS was suppressed due to privacy reasons

Table 2.2: Original variables for the Claims dataset

To illustrate this idea, for example, Charlson Index is the measure of the severity of someone's condition and it has categories from zero to five. Summaries such as average, maximum, minimum, and range were created.

2.2 DRUG AND LAB COUNT

The Drug Count and Lab Count tables were given with the following variables.

Drug/Lab Count
Member ID
Year
DSFS
DrugCount/LabCount

Table 2.3: Variables for Drug/Lab Count

Table 2.4 lists the variables derived from the original data. The drug count and lab count variables could provide many helpful insights for a patient's condition. For example, there will be a clear indication for someone's drug intake by looking at the Drug Count Range variable. The DSFS variable in these two tables is the period from when a patient had a drug or lab record since his or her first visitation.

Drug/Lab Summary
Member ID
Year
Drug/Lab Velocity
Drug/Lab Count Maximum
Drug/Lab Count Minimum
Drug/Lab Count Average
Drug/Lab Count Visits
Drug/Lab Count Range

Table 2.4: Drug/Lab count variables used in the final analysis

The traditional statistical analysis would derive variables such as the maximum, minimum, average, and range for the count of drug or lab records. However, as the technology approaching real-time analytics, another pivotal variable comes to mind is the velocity count. The velocity variable is defined as the last count subtracting the first count, and it gives insights to how a patient's condition is progressing.

If someone had six drug counts on his first visit, and two counts on his last visit, there is a good reason to believe that his condition had improved. Velocity becomes even more crucial as we get faster in obtaining data, especially with today's technology approaches real-time. The

difference in weeks is much more meaningful than a year's elapse time period. We will talk about the role of data velocity in the Big Data analysis further in Section 5.

2.3 MEMBERS AND TARGETS

The Members data provide age and gender for each patient, and it is merged with the claims table by the member ID variable. Hence, each member will have a unique identification. Both age and gender are valuable information for modeling purposes, but they are also harder to acquire because of privacy reasons outlined in the Health Insurance Portability and Accountability Act (HIPAA).

Approximately five percent of age and fifteen percent of gender were missing. New variables sexMissing and ageMissing were created. As we will see in the Analysis section, it is counterintuitive that sexMissing is often one of the first split in the decision tree models. This variable is not supposed to have the most informative characteristics for the analysis, but due to privacy reasons, we will not be able to acquire more information.

The Admission Risk (AR) Score was created based on the Milestone Winner's Paper [2]. Dr. Axelrod has over thirty years of medical experiences. He created the AR Score using the combination of PCG and age group. The Targets data has all the members who have claim records for Y3, and it is used for the final submission.

We also created a new variable based on whether a patient was admitted to hospital or not. The Admission variable is based on DIH. Table 2.5 provides the details. This variable is used for the first stage of the two-stage modeling, where we want to filter out the non-risky patients. More details on the two-stage modeling are discussed in section 4.3.

DIH = 0	Admission = No
DIH > 0	Admission = Yes

Table 2.5: Admission variable

2.4 COMPETITION MEASUREMENT

To evaluate the performance of the prediction, the DIH variable was transformed in the following formula into a logarithm scale. The objective for the competition is to minimize the final error measurement in the following formula.

$$Prediction\ error = \sqrt{\left(\frac{1}{n}\right)\sum_{i=1}^{n}[\log(pred_i + 1) - \log(act_i + 1)]^2}$$

Figure 2.2: Competition performance measurement

where:

1. *i* is the index of a member;
2. *n* is the total number of members;
3. *pred* is the predicted number of days spent in hospital for member *i* in the test period;
4. *act* is the actual number of days spent in hospital for member *i* in the test period.

The outcome variable DIH ranges from 0 day to 15 days. The health care industry has various regulations regarding disclosure of patients' information based on HIPAA, thus the competition annonymized member ID and hospital/vendor names. Another procedure to follow HIPAA was creating a new DIH category, 15+, for any patients who stayed in the hospital for over 15 days. Staying for over 15 days in the hospital is less likely to happen, thus more prone to be de-annonymized, therefore 15+ day outcome was created due to this privacy concern.

CHAPTER 3

METHODOLOGY

Decision tree methods are a new type of non-parametric modeling technique that do not make any assumptions for the underlying data distribution. These methods are computationally intensive, thus they only became popular in the last two to three decades. They are popular for data mining competitions as many winners use them to produce the best results. We will give an overview of Classification and Regression Tree (CART), Random Forest, and Boosting.

3.1 CLASSIFICATION AND REGRESSION TREE (CART)

CART is a binary recursive partitioning algorithm that can have both target variable and predictors as either continuous or categorical. The monograph in 1984 by Breiman and others [3] popularized the method. Its early uses include medical studies, which has the advantage to help doctors making decisions based on simple interpretations. This section will discuss splitting criterion, stopping rules, and tree pruning.

CART starts out with a root node that contains all the observations in the data set. The first split will partition the data into two subsets, and the criterion for the split will be in the form of "is the condition met". By convention, if the condition holds, the node splits to the left, and the rest of the data splits to the right [3]. For continuous variables, the condition would be in the form of "is X<=c". For categorical variables, the condition would be "is the member in the group".

Let T be the root node, $T_R \; and \; T_L$ be the subsequent child nodes. Define $I(T), I(T_R), and \; I(T_L)$ as the impurity measure of the root, right split, and left split nodes respectively. Let $p_R \; and \; p_L$ be the proportions of observations that goes into the right and left splits. The impurity function is defined as

$$impurity \; function = I(T) - [p_R I(T_R) + \; p_L I(T_L)] \tag{1}$$

CART will look through all the possible splits from the root node. The algorithm is computationally intensive, but the advantage is that the number of possible splits is finite. There

are many forms of impurity measures. Two well-known measures used for CART are Gini Index and Entropy, and the mathematical definition is given in (2) and (3) respectively. We will analyze a simple example to illustrate each method.

Suppose there are K classifications, let $p_1, p_2, \ldots\ldots, p_K$ be probabilities for any class in the present node, with $\sum_j p_j = 1$, and $p_j \geq 0$ for all j. Define

$$Gini\ Index = 1 - \sum_j p_j^2 \tag{2}$$

$$Entropy = \sum_j -p_j \log p_j \tag{3}$$

CART first calculates the impurity measure at the parent node, $I(T)$ in (1). Then it searches through all the possible splits for that node. Calculate the weighted average of the child nodes, $p_R I(T_R) + p_L I(T_L)$ in (1). Find the split that minimizes (1) and choose that split. Then CART recursively start the algorithm again. This result makes intuitive sense. We want to have the purest class at each terminal nodes, thus the p_j's should be large, which means that category is the dominate class in that node. Hence the impurity function will be small.

The following simple example will be used to demonstrate the impurity function calculation.

Gender	Age	Class
Male	Greater than 70	No
Male	Less than 70	Yes
Male	Less than 70	Yes
Female	Greater than 70	No
Female	Less than 70	No
Female	Less than 70	Yes

Table 3.1: Impurity function and splitting criterion illustration example

The Gini index for the root node is 0.5. If a split is made by gender, the weighted average is 4/9, and for a split made by age, the weighted average is 5/16. Similarly for Entropy, the root node is 0.301. If a split is made by gender, the weighted average is .27643, and for a split made by age, the weighted average is .2442. Hence age is the better split for this example.

There is a subtle difference between Gini and Entropy. For a simple and small data set such as the example provided above, either Gini or Entropy will give the same result. For a model built on a large and complex data set, that would often not be the case. An analogy would be

throwing dart from three feet away versus throwing from ten feet away. The technique one uses from the shorter distance is not as important as from the longer range [4].

Gini looks in the data and tries to isolate the largest class from the others. At each node, this method is performed recursively. If weights are used for different classifications, Gini will favor the most "important" class. In a perfect world, the final tree would have pure class at each terminal node. In other situations, it will try to get to the purity as much as possible. In the mathematical language, the best model will have a Gini Index of 1. Gini has performed well in many experiments and it is set as the default for CART.

CART uses an approach called greedy algorithm, which means once a variable is split, the algorithm will stop looking at this variable in the future. The goal for this approach is to increase the efficiency of computing time, a serious issue for large datasets. The greedy algorithm's weakness is that it will find the locally best model based on the subset of predictors that are left, which could potentially overlook the globally best model.

An innovative feature of the CART algorithm is that the tree always splits to the maximum size and only then pruned back to the optimal size. The authors of the CART monograph argued that it is important to capture all the variable contributions to the data, and forcing a stopping rule earlier would cause the data to miss certain characteristics [3]. The maximal tree will have one of the following three possibilities:

1) The terminal roots have one observation, or
2) The terminal roots have only one class.
3) A predefined number of minimum numbers of observations in each terminal root.

There are two ways to prune back the tree: data partitioning and cross-validation. Cross validation was used because often there are not enough data. It has been used since multiple regression model testing, and its special case is the jack-knife analysis, which leaves one observation out each time. In today's technology, with more available data and more efficient computation tools, data partitioning is the preferred option.

Data partitioning starts by splitting the sample into two sets: the training data and the testing data. Then it builds the tree with the training data, and prune back using the testing data. The

cross validation method will divide the original data into V equal sub-samples. Then use the other $V-1$ portions as the training data, and the last set as the testing data. It will run across V permutations, where V different testing sets will be used, and the result of all the samples will be combined to calculate the error rate, which in turn will be applied to all the training samples in the end.

Once a tree is pruned, a classification will be assigned to all the data in that node. Suppose there are K classes, $j = \{1,2, \dots, K\}$. Define $j^* = \arg max_j\, p(j|t)$, if $j^* =$ "Admitted", then every data that ends up in this node will be assigned "Admitted". Let $r(t) = 1 - p(j^*|t)$ be the re-substitution estimate, and $R(T) = r(t) * p(t)$ be the misclassification cost function, where $p(t)$ is the proportion of the sample data that goes to node t out of the whole training data set. $R(T)$ always improves as more splits are made. Thus, it cannot be the most effective measurement cost.

Since that $R(T)$ increases monotonically as more splits are made, the pruning algorithm is based on the cost function in (4). An intuitive meaning of the formula will be discussed below.

$$R_\alpha(T) = R(T) + \alpha|T|\,, \tag{4}$$

Where:

1) $R(T)$ is the misclassification cost from the splitting rules;
2) α is the complex parameter, which is selected by the modeler;
3) T is the number of nodes.

More splits will always have a lower $R(T)$ as shown in the proof, and the optimal tree will then be the maximal sized. Thus, the complex parameter α will be needed to minimize the adjusted cost function $R_\alpha(T)$. The value α is a positive number, if α = 0, then $R_\alpha(T) = R(T)$, which indicates the tree is split to the maximal. As α approaches to infinity, the tree of size 1 will be selected, or just the root node, because $R_\alpha(T)$ will be minimized when T is the smallest.

While pruning the maximal sized tree, there are finite numbers of possible optimal sub-trees. The goal is to minimize the function in (4). CART will prune the tree iteratively to find the best tree based on the user provided value α. The default choice of α for many packages is 0.01, but users should always experiment other possibilities to find the best value for a given data set.

3.2 RANDOM FOREST

Random Forest is an extension of CART popularized by [5]. Its features include bootstrap aggregation, ensemble methods of trees, committees of experts, and combining models. Random Forest was an innovative method at the time of the publication where it consistently beat other top algorithms. Currently, it is still a powerful method used for model buildings as well as data cleansing and preliminary exploratory analysis.

Random Forest builds multiple CART-like trees. The new idea is providing two random components to the analysis. One component is the bootstrapping samples for each tree. The other one is choosing the best variable split based on a subset of the original predictors, where the number of variables to choose from the subset is defined by the modeler.

The bootstrap aggregation is a sampling with replacement method. At the n^{th} tree, samples from the training data will be drawn, some observations will be drawn once, and others will be drawn more than once or none at all. Empirical evidences have shown about 36% of the samples will not be drawn, these are called the out of bag (OOB) samples. Random Forest repeatedly performs bootstrap across all the trees, but each tree will have a different set of OOB samples [5].

Suppose there are M predictors in the data set. The user will input a number K, which is less than M. At each node, Random Forest will only look at K out of M predictors to choose the best split. The splitting criterion by default uses the Gini index. Once the split is made, at the next node, it will look at a different set of K predictors to choose the best split. Experiments have shown with a large numbers of predictors at each node, the set of eligible variables will be quite different from each other [5].

The advantage of using only a subset of predictors improves computation efficiency tremendously. Another advantage of Random Forest is scalability. Its algorithm allows for parallel computing, where in today's big data era, a cluster of multiple computers can build different trees at the same time. This is one important distinction from Gradient Boosting Machine, which builds trees sequentially, thus cannot be performed in parallel time [6].

The goal of Random Forest is to produce weak learners and then combine the result. The algorithm will aim to have all the terminal nodes with only one data point. All the trees are split to the maximal without any pruning. Eventually, the important variables will make into the tree.

There are two competing forces that affect the model result. If the user let K be large, then the trees will be correlated to each other, which produces weak model. On the other hand, as K increases, each tree will provide stronger strength. Thus, increasing K will introduce a tradeoff. Breiman suggested trying out $\sqrt{M}, \frac{1}{2}\sqrt{M}, and, 2\sqrt{M}$ first, and then find the best choice somewhere in between [5].

The model strength is measured by the OOB error estimate. A data point will often appear multiple times as the OOB sample. This observation will have a prediction for each tree that it acted as OOB, and the result will be scaled to get a measurement result for that observation. The sum of all OOB samples will be used in the final model measurement.

The final prediction will have every data points in the data set go through each tree. Every observation gets assigned a class by voting for categorical variable or averaging of all the prediction values for continuous variable.

3.3 BOOSTING

The original boosting method, AdaBoost, was developed by Freud and Schapire (1995) [7]. The main idea behind boosting is to fit a large number of small trees or weak learners, and then ensemble them to get the prediction.

Each tree is designed to improve only slightly compared to random guessing, thus it is called a weak learner. The goal of boosting is to capture a little more information at a time until no more "useful" improvements can be gained. Boosting methods has been further developed after AdaBoost, and it has been used by a variety of applications and industries such as telecommunication, banking, and search engine optimizations.

The initial step of the analysis assigns equal weights to all the data samples. If there are N observations, the first tree will assign 1/N to each data point. After the first tree is built, the second tree will be build based on the first tree's error terms. While building the next tree, an

observation that was predicted inaccurately from the previous tree will get an increased weight. In the same manner, an observation predicted correctly will get a decreased weight. Sequential trees will be built in a similar fashion until it reaches the maximum number of trees, which is defined by the modeler.

If a data point is continuously being inaccurately predicted, its weight will keep increasing, thus becoming even more influential. This idea is used by all boosting methods, where more attention will be given to inaccurate data points. Similar to the Random Forest, the final prediction for Boosting is based on majority vote for classification, and averaging for regression. Although Boosting often beats random forest, the drawback for sequentially built trees is that parallel computing implementation is not very practical with today's technology.

Jerome Friedman developed GBM in 1999. It is based on the original boosting trees but Friedman implemented finer details into the algorithm. The method has become one of the most powerful data mining techniques in many applications, and many data competition's winners used GBM in some ways. GBM builds similar sequential trees as other boosting methods. It uses CART trees as the weak learners, and generally starts with only five or six splits initially [8].

Programming languages such as R and Python have GBM packages [9]. We used R's GBM package for our analysis. User-defined parameters in GBM include the following:

- Loss function, examples include least square and absolute deviation
- Number of training data points N
- Number of trees M
- Number of terminal nodes K for each tree
- Shrinkage rate λ, and
- Subsampling rate p

Over-fitting is a common issue when building complex models. Friedman introduced shrinkage rate λ and optimal number of trees M as regularization measures. Both of these parameters are set by the user. λ is between 0 and 1. $\lambda = 1$ indicates the model uses no shrinking. The two parameters λ and M are related in such a way that as the user decreases λ, M needs to be increased in order to accommodate the smaller improvement requirement. The best way to select

M is to set a given λ, and then use a validation data set to select the optimal M based on prediction errors.

In the follow-up paper, Friedman also added a sub-sampling procedure [10]. The idea came from Breiman's Bagging method used in Random Forest. At each iterative step of building a new tree, a subset of the training data is selected without replacement. The subsampling rate p is between 0 and 1. Friedman has found $0.5 < p < 0.8$ performed best in GBM. In R, $p = 0.5$ is set as the default, users should experiment different values while building models in different applications. This implementation improves the computation efficiency tremendously as each tree only work with half of the original sample size.

A summary of the improved Stochastic Gradient Boosting algorithm described by Friedman [8] is listed in figure 3.1.

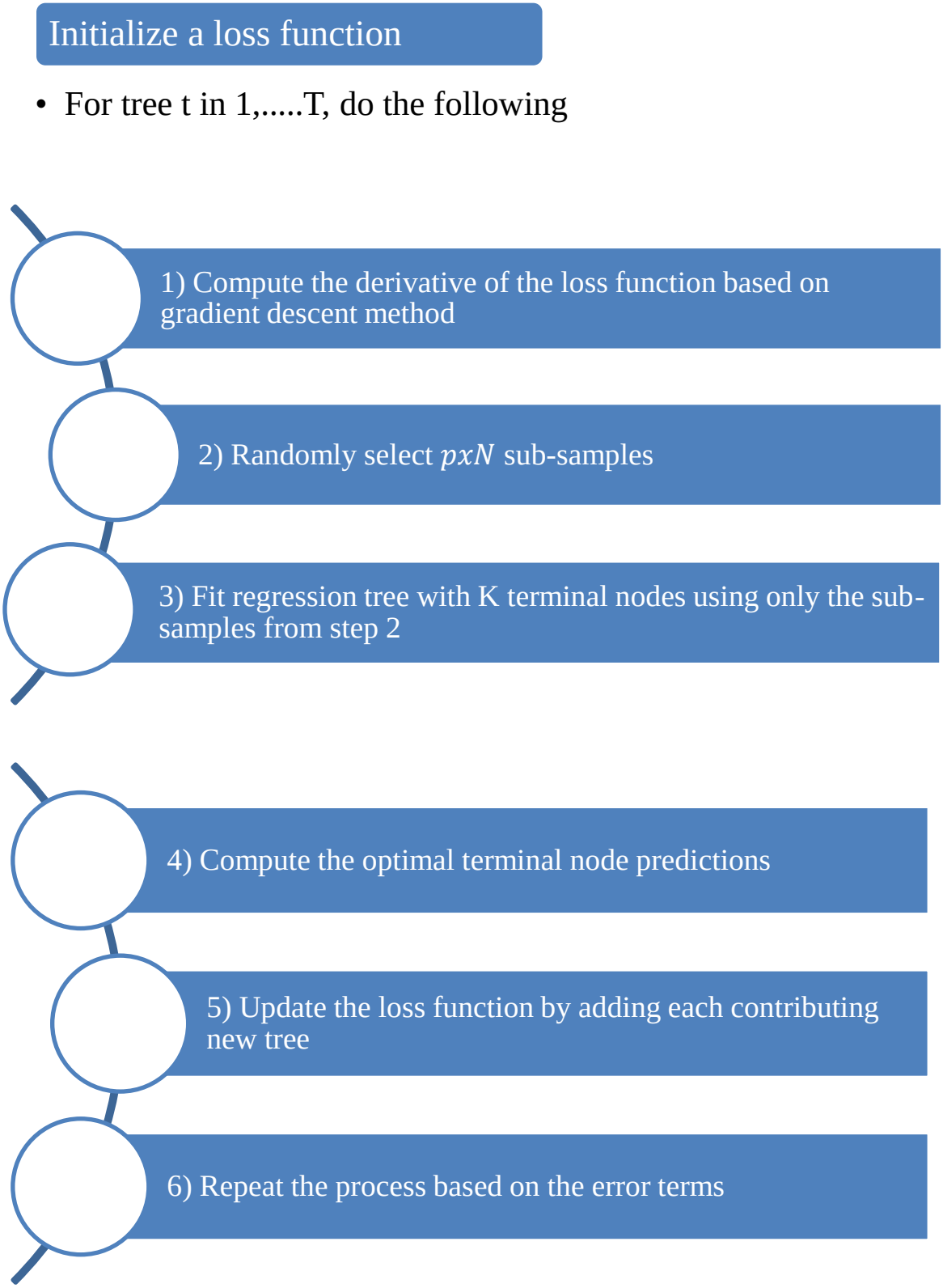

Figure 3.1: Stochastic Gradient Boosting algorithm

The GBM has been a popular algorithm with good results since Friedman's papers. Top firms such as Google and Amazon are devoting resources to further develop and implement it use. The downside of GBM includes intensive computation cost and difficulty of understanding the model for typical business users. As technology continues to advance and more resources explaining GBM, more people will understand its benefit and use it in different applications.

3.4 LOGISTIC REGRESSION

Logistic regression studies the same problem as CART's classification method, which predicts whether a patient is admitted to the hospital in our context. Similar to CART, Logistic regression can have both continuous and categorical predictors, and the outcome is a probability value for a class. In this subsection, we will briefly discuss the model fitting procedure and its connection to CART.

Logistic regression uses Maximum Likelihood Estimator to fit the model. It is in the family of Generalized Linear Model. Suppose $\pi(x)$ is the probability that a patient will be admitted to the hospital, and $1 - \pi(x)$ is the probability that a patient will not be admitted. The logit link function is displayed in (5).

$$logit\ link: \ln\left(\frac{\pi(x)}{1-\pi(x)}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_i x_i, \quad (5)$$

where each $\beta_i's$ is the coefficients fitted to the model based on the Maximum Likelihood Estimator method.

The left hand side of (5) is a probability between 0 and 1. Usually we will assign no-stay if $\pi(x)$ is less than 0.5, however, given the context of our study with many more no-stays, we will assign no-stay if it is less than 0.8. The reason here is to reduce false negatives in the analysis, which we will talk about with more details in section 4.3.

Logistic regression is a good substitute for CART's classification problem. It has been the top modeling choice for binary outcome variable in many fields. CART and other decision tree methods have provided modelers new options, but logistic regression continues to be a good comparison.

CHAPTER 4

DATA ANALYSIS

Rattle is a user-friendly package in R that simplifies the modeling building process. It is built in a point-and-click manner that the users can import the data and then choose the appropriate parameters simply. We used Rattle for various Generalized Linear Models, CART, and Random Forest analyses. Currently, Rattle only has AdaBoost algorithm built in, but as time advances, more complicated algorithms will be implemented.

Figure 4.1 below is a screenshot of the package Rattle in R. This is a Graphic User Interface (GUI) tool used for various decision tree models.

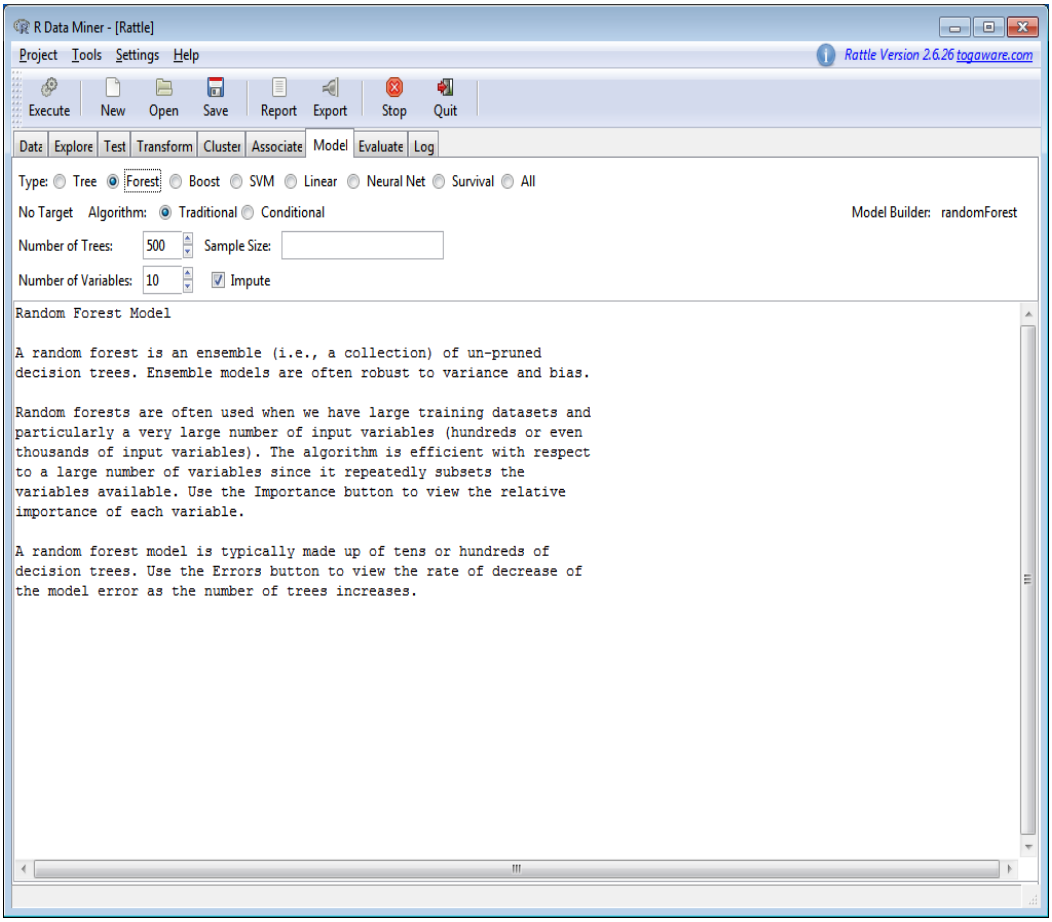

Figure 4.1: Screenshot of Rattle Package built in R

4.1 MORE DATA BEATS BETTER ALGORITHM

This is a topic that will not have a consensus conclusion among the data science communities. Technology enables us to store more available data and build better computation tools for modeling. The classic tradeoff for any business stakeholders is whether to put more resources for data storage capacity or build better programming tools?

The current state of technology can handle over 70,000 observations without any problems. Hence for this thesis, using the more complex decision tree methods is preferred. However, to understand and test the argument presented, we would like to study this particular scenario further.

We split the training set into ten randomly selected subsets. We then train the smaller subsets using Gradient Boosting Machine and the original set using the more straightforward General Linearized Models. The parameter used for Gradient Boosting Machine is provided in Table 4.1. We would like to compare the final model performances between the two approaches.

# of Trees	Shrinkage	Inter. Depth	Bagging %	Training %
1500	0.001	5	0.7	0.9

Table 4.1: GBM parameters used for smaller subsets

Table 4.2 gives the performances of the ten subsets using the Gradient Boosting Machine technique. The results in this table as well as the rest of the thesis were calculated based on the prediction error formula provided in Figure 2.3.

Model Subset	Performance
1	0.277598
2	0.274341
3	0.273786
4	0.277085
5	0.284742
6	0.273745
7	0.280607
8	0.283868
9	0.282588
10	0.277598

Table 4.2: Performances for the subsets

The overall average of these ten subsets combined is 0.278202. When we run the original set with over 70,000 observations, we got the results of 0.217234 for Poisson Regression and 0.220663 for Generalized Linear Model with the least square link function. We can see clearly that bigger data beat better algorithm in this scenario.

4.2 MODELING

Data Mining Competitions gave data scientists a new way to analyze big data problems. A few years ago, data analytics were used as a business intelligence tool to improve business decision makings. However, Kaggle and other data competitions gave data scientists a new type of platform to analyze data. Teams from across the world compete against each other to build better models. The measurement used for the competition is to minimizing the error term. Thus, in many cases, the business context is often non-essential. Rather building the model based on data is the only important factor.

One of the winners of the Kaggle competition said in an interview that variable selection, model calibration, and model blending are the three keys to building the best models. Variable selection is described in the Data Descriptive section. We will discuss the remaining components and then focus on them for the model building.

The goal of any statistical analysis is to perform better in future predictions. For many models, there could be useful improvements made to decrease the error term. For a multiple regression analysis, variable selection methods and choosing interaction terms are two possibilities to calibrate a model. With non-parametric models, often there are no rigid ways of choosing one model over another, but it will take trial and error to find the best model.

A different way to ask the same question for our analysis is what parameter values for each model will produce the best result. For example, in CART, we can set the complexity parameter in a few different ways. After each model run, we will adjust these values and compare the results. This process is more of a guess work and making adjustments to parameter values accordingly rather than having a rigid scientific procedure.

Once we build all the models which are calibrated to its best outcome, we can combine some of these models in a certain way to improve the predictions. The simplest form of blending is

averaging the results. Nate Silver used multiple polls to make his best prediction on elections. More complicated blending used by data mining competitors includes assigned weights for each model based on submission scores.

We would start by averaging Random Forest with Gradient Boosting Machine (GBM) results in our data analysis. In the next step, if we believe GBM has a better result, we would give it more weight in the averaging process. After the basic blending analysis, we will also focus on the two-stage modeling, which is discussed in more details in section 4.3.

One aspect of decision tree models is the partition of validation data sets. Three groups are needed in each analysis: they are the training set, validation set, and testing set. The training set is used to build the model, the validation set will be used to pruning the trees, or improve the newly built trees, and the testing set is a hold-out sample used to measure the performance. The partitioning size is denoted by training set/validation set/testing set. For example, 70/15/15 is the default setting in Rattle, which means seventy-five percent of the original data is used as the training set, fifteen percent as validation set and fifteen percent as testing set.

As we mentioned in the Random Forest section, it is better to use partition rather than cross-validation method when building models for large data sets, because a seventy percent subset out of 70,000 observations is still relatively big. Thus the concern of insufficient sample size is not a problem. We also checked 60/20/20 and 80/10/10 partitions. These results ended up to be very close to the default 70/15/15. Hence the results followed are all based on 70/15/15.

Throughout the analysis, we have been emphasizing that the parameters of the decision tree based models were chosen based on trial and error. Hence, there are no fixed ways of selecting them. We have used the default measures first, and then made adjustments accordingly.

There are 142 variables in the data, thus we start with twelve variables, which is approximately the square root of 142 as the predictor subset in the Random Forest analysis. To start the analysis, we build models using Y1 claims and Y2 DIH, and then check the performance on Y2 claims and Y3 DIH.

Table 4.3 below shows the parameters used when building the Random Forest model.

Model	# of Trees	# of Predictors	Run. Time (min)	Performance
1	50	12	12.84	0.230096
2	70	6	12.91	0.225893
3	100	12	26.07	0.231661
4	100	6	17.89	0.225421
5	100	24	39.53	0.228222

Table 4.3: Random Forest performances

We run five different Random Forest models to get a better idea of which parameters worked the best. We started with fifty trees and twelve bagging predictors. As we increase either the number of trees or the number of predictors, the running time will increase. After trying a few combinations, we have found model 4 with 100 trees and six predictors performed the best for this particular data set.

While both Random Forest and GBM are recently developed modeling techniques, GBM improves the speed of computation much faster. On the same processor, a Random Forest model with 100 trees will run approximately twenty minutes, while a GBM runs in less than two minutes. The reason behind this is Random Forest has to run each tree to the maximal leaf, which takes a tremendous amount of memory space and processing time. On the other hand, GBM only runs four to six nodes on each tree. Thus it improves model computation speed.

Besides the speed factor, GBM also has better performance. This is expected as GBM brings more ingredient and new nuances such as sequential selection and correcting the error terms into the model building process. Because of these factors, we run more GBM analysis than Random Forest throughout our data modeling process.

Without any particular routine selection method, we run a few GBM models to get an intuitive feel of the performances. The goal here is to try different options for the parameters and then fine tune it in order to find the best combination.

Table 4.4 has the GBM performances that we run. We will talk about the ideas behind selecting some of the parameters below.

Model #	# of Trees	Shrinkage	Int. Depth	Bagging %	Training %	Performance
1	500	0.001	1	0.5	0.5	0.221517
2	1000	0.001	2	0.5	0.5	0.221265
3	10000	0.001	2	0.5	0.5	0.217164
4	1500	0.001	4	0.5	0.5	0.216236
5	1500	0.001	4	0.7	0.5	0.216024
6	1500	0.001	4	0.9	0.7	0.215867
7	2000	0.0001	6	0.9	0.7	0.221265
8	1500	0.001	5	0.7	0.9	0.214563
9	1500	0.005	5	0.5	0.9	0.216063
10	1500	0.001	5	0.9	0.9	0.214640

Table 4.4: Gradient Boosting Machine performances

We begin with the number of trees, 500 and 1000 were a decent starting point. We got good results and relatively fast running time. Hence, we increased it to 10,000 trees to see what insights we can gain from it. Two factors came in as the results. The first one was that the running time slowed down tremendously, which is a down side of running such a huge tree. Second, as we continue to have the shrinkage rate at 0.001, we realized the improvements were mostly zero after around 1500 trees.

Having too many trees after that threshold became essentially useless. Hence, we were comfortable to run 1500 trees afterward. The shrinkage rate goes along with the number of trees as we discussed in section 3.3a. Smaller shrinkage rate require more trees. The rate 0.001 was a good starting point, we tried with other numbers a few times, but still found 0.001 was the best choice selection.

The interaction depth is the number of interaction variables for a given model. One indicates no interaction terms, or just an additive model. Two means interaction of up to two terms, and three means any interaction terms can have three terms and so on. After trying a few different options, we increased this parameter to six and the running time slowed down. Also as we used six, the performance actually decreased. Thus, we found the interaction depth of five was the optimal choice.

The Bagging and Training percentages indicates the proportion of predictors and the training sample size of the data, respectively. Bagging selection is similar to the number of predictors' choice in Random Forest. There are no suggestions on a good value for GBM as Breiman has made for Random Forest. The size of training sample is also a guessing work. Based on Table 4.3, model eight with seventy percent bagging and ninety percent training yielded the best result.

Random forest and GBM are good models for general problems, as they provide a vast number of selection parameters for modelers to try and learn insights behind these models. This part of the analysis will be used as the base line when we start working on the two-stage modeling analysis. We will test to see how effective the new two-stage modeling will measure against the current techniques.

4.3 TWO-STAGE MODELING

At the beginning part of the analysis, we noticed the data was extremely unbalanced toward no-stay patients. The Y2 data had over eighty-five percent of the patients not admitted to hospital, while the rest were admitted from one day to over fifteen days. Based on this observation, we decided to use a two-stage modeling method. The idea is to run a classification model on the data based only on the Admitted variable. Then run the regression methods on the subset of the data without patients who were both predicted and actually no-stay.

	Predicted Admitted	Predicted No-Stay
Actual Admitted	Okay	False Negative
Actual No-Stay	False Positive	Okay

Table 4.5: Two-stage modeling groups

Table 4.5 provides more information about the two-stage modeling. In the first stage of the analysis, we will build Logistics Regression models for the classification variable Admission. Define the fourth quadrant to be non-risky, those patients that were both predicted and actually

not admitted to the hospital. We can assume that their claim records indicate of routine check-ups such as annual physicals or lab testing. The first quadrant has patients we need to build models as they were clearly more likely to be admitted to the hospital based on claim records.

We should also focus on the remaining two groups. We need to pay close attention on both the classification assignments and the length of stay for both groups. Due to the unbalanced nature of the data set, the first question should be which mistake is more costly, False Positive or False Negative. Similar to other health studies, it is more important to focus on False Negatives. This group is predicted to be healthy but actually was admitted to the hospital.

The reasoning behind this is that we want to be conservative for this group, making sure if a patient who has certain health characteristics more likely to be admitted will indeed predicted that way. The class weight in Logistic regression is set to be 0.5 as the default. However, to give more weights to the more costly class, we will use 0.8 instead.

Figure 4.2 provides the algorithm.

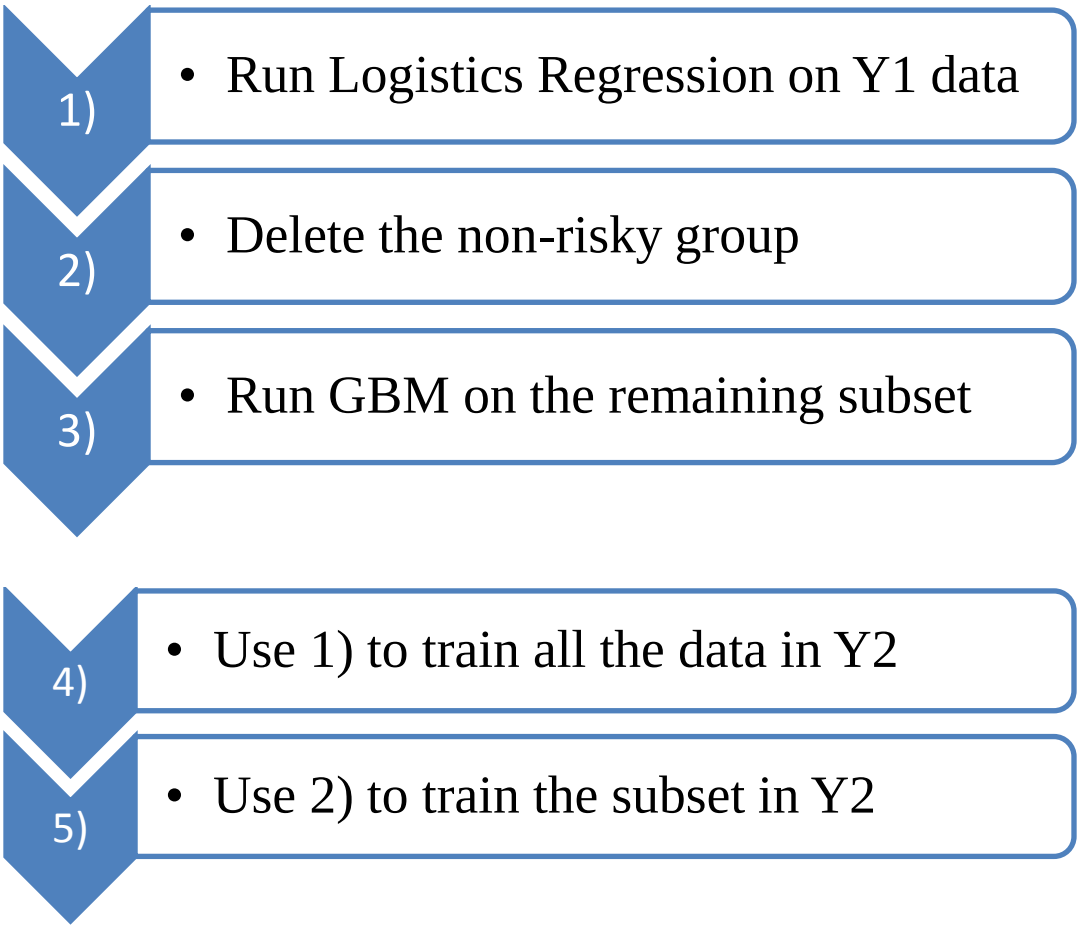

Figure 4.2: Two-stage modeling algorithm

The logic of the two-stage analysis is to separate out the classes of admitted and non-admitted in the training set, this is the first model stage. We then take out the non-risky group from the training set, and we ran Gradient Boosting Machine on the remaining data to predict the number of days stayed in the hospital, this is the second model stage. This subset data are patients with risky characteristics. Hence the second stage model will put more focus only them, and will be applied to Y2 testing data.

Once the model is built for the training data, we will apply this model to the testing set, using the same two-stage procedure. We will apply the classification model to the testing Y2 data. This model will eliminate a subset of the patients. Then we use the second stage model which was built with risky patients to predict the subset in Y2. The best result we got was 0.215485. Table

4.6 provides the summary of the analysis. This result is comparable to the top performance from the Gradient Boosting Machine models.

Logistic Regression	GBM	Result
First Stage	Second Stage	0.215485

Table 4.6: Two-stage modeling result

4.4 DISCUSSION OF TWO-STAGE MODELING

The advantage of the two-stage analysis is that we can eliminate many of non-risky patients and thus focus only on risky patients. Non-risky patients are less likely to have serious health conditions, thus we can put them aside and assume they will not be admitted. Due the size of the data, the variability for the non-risky group would be large, which is counter-productive because we want to have this group predicted zero days. Having the two stage analysis will allow us to assign more zero-day predictions.

On the flip side of the coin, we will make two kinds of errors. Both contribute to the error term significantly. The false negatives will have zero days as the prediction but in fact they did stay in the hospital for a number of days. Similarly problem occurs for the false positives. We can try to control one of the false predictions at stage one, but there will be approximately ten percent of observations assigned to the wrong group.

The idea of two-stage modeling for Big Data is a novel idea. For any new ideas, it has both positives and negatives. However, the applications it can be used are numerous. Examples include insurance claims, where the first stage analyze the indicator variable claim, and the second stage will look at how much claims occur given there is a claim. Other examples include credit card default and fraud detection.

CHAPTER 5

HEALTH DATA AND TECHNOLOGY

Health care is one of the most vigorously debated topics in the United States in the past few years. In this section, we will discuss the role of technology and the health care data management.

The health care reform has been in the media over many years, and there does not seem to be any progress in improving the current status. There are few reasons for this scenario which include but are not limited to the rising cost, the patient care system, and the privacy protection. As the population age continues to increase, and medical technology keeps improving, it is not a surprise that the cost of health care is increasing.

For many Americans, health care is also an emotional issue. We want to have the best patient care system in the world but also keeping it affordable. Another issue that does not come up until problems arise is the privacy for patient's information. Doctors and providers have sensitive information such as social security number or patient's conditions, this information often are the target of malicious hacking schemes, so protecting it today is even more important.

Data privacy laws are not robust in America. Online transactions can be used by the company without any consultation with the customers. However, due to the sensitive nature of medical related information, the Congress passed Health Insurance Portability and Accountability Act (HIPPA) in 1996 and continues to update the current bill to protect the privacy of health care consumers.

Besides the privacy issues, technology has been helping professionals in gaining more insights in the health care data. Real-time analytics is the goal for all data scientists. Its advantages are more volume of data, faster speed of accessibility and up-to-date information that comes into the data warehouse in real-time. The technology is already in place, and the health care industry is taking advantages of the new innovations in this space.

In today's technology, we can capture these data. However, there are challenges to maintain it. For example, how can we define the time frame of 'now'? Is there a stopping time point for us to freeze and access the data? These questions will be especially crucial for data scientists in the near-future as the technology continues to advance.

The amounts of data information are vast in the health care industry. Hospital records along with insurance information for the United States population will amounts to hundreds of millions data points, and that is just the tipping point. Much more complex records and data storage are in the recording system, but we need better ways to storing it.

The Hadoop ecosystems along with the cloud computing methods were popularized by Google in their papers on Big Table [11]. The idea is to storage data more efficiently and builds parallel computing in the process. We will briefly discuss some of the ideas behind Map Reduce such as parallel computing and data replication.

Data replication is a data protection method. Suppose we have six data points and three computers. Figure 5.1 is a simple example.

Figure 5.1: Map Reduce data replication

We have six letters in the file. We will partition three different files to store each letter. The first goal is make the files smaller, and the second goal is to make a replication of the data, therefore, if one server is broken, the data will not be lost. Larger data sets will increasingly need to have security protections.

More data and better technology implicates new options to analyze our data. The analysis that we have talked about were built-in packages by statistical software such as R and Python. We only had to change the parameters to try different models. We had also worked with two-stage models and made some comparisons between the methods. However, this is only scratching the surface of a Data Scientist. The top prize winners in the competition used many varieties of decision tree methods. Instead of using a built-in package, one can write the algorithm for any rule-based model.

Machine learning is a field that continues to popularize today, and its applications will broaden to different fields. The rise of Big Data will bring more interests for the field. Both Big Data in general as well as the Health Care sectors will require better storage and new analyzing technical tools. Computers have been increasing in both speed and storage spaces to accommodate.

In this thesis, we advocated the area of being more experimental in analyses. In a Facebook press release, Mark Zuckerberg mentioned the Hacker Way, which means trying to use various unknown methods and then look at the results to make decisions. The new era of statistics is also experimental, instead of starting with a hypothesis, and then uses data to acquire evidences. We will let data detect the insights, only asking questions after what we find from the underlying data.

Traditional Statistics make hypothesis, or have assumptions regarding the underlying distribution of the data. For example, the hypothesis in our data would be patients stayed in the hospital because he or she visited the emergency room more than three times. An assumption would be that patients stay in the hospital follow a Poisson distribution. We would analyze the data, and decide whether this is the case.

Data scientists today use the Hacker Way. They will organize the data, and then build different models solely to minimizing the error terms. Only after models are built, will they ask meaningful insights from the analysis, and discuss further actions to proceed. In Kaggle and other software companies such as Google, and Facebook, the Hacker Way has become the new trend.

CHAPTER 6

CONCLUSION

CART, Random Forest, Gradient Boosting Machine, and Logistic Regression are all important analytical techniques for Big Data problems. We run different methods to get the best results for the prediction. Among these popular methods, GBM had the best results, which confirmed its powerful development designed by Friedman.

To think about the problem in a new perspective, as we found the unbalance-ness of the data, we analyzed the data using the two-stage modeling process. This strategy provided a new lens of looking at the problem. Separating the data into different characteristics of risky and non-risky makes the analysis more of a personalization problem. As Big Data continues to grow, it is especially important to focus on fields such as personalized advertisements when businesses providing a service.

Another usefulness of the two-stage model is to assign zeroes to the patients not staying in the hospital the following year. Both Random Forest and Gradient Boosting Machine study a continuous outcome variable, thus it is impossible to have an exact prediction of zero days. By working with the two-stage model, we can partition a big part of the non-risky group and assign them zero days, which help to reduce the prediction error term.

The Hacker way gave us the opportunity to try non-traditional analysis and it provided new insights for the problem. We should broaden the view of new possibilities and being open to try different ways of solving the same problem. On the subject of the hacker way, open competition such as the Kaggle website gave data scientists a new way to improve the field. Open sourced software or operating systems such as R or Linux has been around for many years. The business model is to have users around the world contribute more advancement in the technology space. At the same time, it offers a free tool for analysts to use. Kaggle provided a similar platform for data scientists. We can compete in data analysis, and provide more insights.

Because of these new developments, there are even more collaborations between different fields. Computer scientists work on improving algorithms, and manage data storages in a more efficient

way. For statisticians, our task is to design experiments, build better models, avoid over-fitting and provide interesting business insights.

There are important collaborations needed for both fields. As statisticians try to build better models, we need better software or algorithms from computer scientists. The Big Data field will require even more sophisticated tools in both acquiring and analyzing the data. More work from both disciplines together will help improve the area.

REFERENCES

[1] IBM. (2011), From the Stretched to the Strengthened. http://public.dhe.ibm.com/common/ssi/ecm/en/gbe03433usen/GBE03433USEN.PDF

[2] Brierley, P. Vogel, D. Axelrod, R. (2011), Heritage Provider Network Health Prize Round 1 Milestone Prize: How We Did It – Team 'Market Makers'. Kaggle Competition Winner's Paper.

[3] Breiman, L. Friedman, J. Olshen, R. and Stone, C. (1984), Classification and Regression Trees. Chapman and Hall.

[4] Breiman, L. (1996), Some Properties of Splitting Criteria. Machine Learning, 24(1): 41-47.

[5] Breiman, L. (2001), Random Forest. Machine Learning, 45(1): 5-32.

[6] Geurts, P. Ernst, D. and Wehenkel, L. (2006), Extremely Randomized Trees. Machine Learning, 63(1): 3-42.

[7] Freud, Y. and Schapire, R. (1999), A Short Introduction to Boosting. Journal of Japanese Society of Artificial Intelligence, 14(5): 771-780.

[8] Friedman, J. Greedy Function Approximation: (1999), A Gradient Boosting Machine. Annals of Statistics, 41(4): 1189-1232.

[9] Ridgeway, G. (2005), Generalized Boosting Models: A guide to the gbm package. http://cran.r-project.org/web/packages/gbm/index.html

[10] Friedman, J. (1999), Stochastic Gradient Boosting. Computational Statistics & Data Analysis, 38(4): 367-378.

[11] Ghemawat, S. Gobioff, H. Leung, S. (2003), The Google Filesystem. Google Research Publications.