THE UNIVERSITY OF NEW SOUTH WALES
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Multi-instance Learning for Prediction of Days in Hospital

By
Simon J. Keung
simon.keung@student.unsw.edu.au

Supervisor
Dr Alan Blair

Assessor
Dr Michael Bain

16 October 2012

Thesis submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering

Abstract

In days in hospital prediction, historical patient health claims data are used to predict how many days the patient will spend in hospital in the subsequent year. The is based on the data provided as part of the Heritage Health Prize Competition, a competition the rewards the top entries that produces the most accurate prediction.

The problem is most naturally represented as a multi-instance problem. However, current multi-instance learning research commonly follow the standard assumption. The standard assumption suggests that bag labels can be determined from a single within-bag instance, which is inappropriate for the particular problem domain. The thesis introduces a technique for multi-instance regression with relaxed assumptions, and demonstrate the feasibility and performance of using Gradient Boosting Machines to learn both instance level and bag level functions.

A method to transform the data to represent a single-instance learning problem is presented to compare the performance of the proposed method with common supervised learners.

Lastly, the candidate learners are used to form a committee machine that outperforms any single candidate learner and compares well with existing entrants to the competition. In all the committee machine learning techniques in the comparison, committee machines that include multi-instance learners are superior to committee machines excluding multi-instance learners.

Acknowledgements

I wish to express greatest thanks to Dr Alan Blair for supervising me over the past year. He has been patient, encouraging and he has been generous with sharing his views and insight. I would like to acknowledge Dr Tiberio Caetano, where our brief encounter may have prompted the direction of the thesis, and Dr Michael Bain, for his rigorous questioning and advice. Last but not least, I wish to thank my parents, for their continuous support and inspiration.

Table of Contents

1 Introduction

1.1 Background

1.1.1 Preventable Hospitalisation

In developed countries, healthcare costs form up to 15% of a nation's gross domestic product. Estimates have shown that these costs will rise significantly as the population ages in the upcoming years [1]. Healthcare costs are distributed amongst various parties, insurance providers, private healthcare service providers, patients, and also taxpayers. As such, there are many stakeholders that would benefit from the reduction of healthcare costs. In particular, lower healthcare costs means that healthcare is more accessible to the general population. The corollary is that high costs would have a negative impact on standard of living, by straining economic resources and reducing overall social and personal well-being. The focus of this thesis is preventable hospitalisation. Preventable hospitalisations are hospitalisations that could have been avoided through sufficient and timely provision of non-hospital healthcare. As an example, diabetic hospitalisation is nearly unnecessary with the early detection and treatment, unless there are other extenuating factors. However, over a 12 month period surveyed in Queensland, diabetic complications led to over 160000 bed days amongst the population of approximately 4.5 million [2]. Evidently, preventable hospitalisations incur a substantial cost to health systems. In the United States where the data used in this thesis is source from, nearly 4.4 million hospitalisations in the United States were deemed preventable and accounted for a over $30 billion US dollars costs in 2006 alone, not accounting for any external costs [3].

1.1.2 Heritage Health Prize Competition

As a response to the issue of preventable hospitalisations, the Heritage Provider Network ,a private health fund, has initiated the Heritage Health Prize Competition [4]. The competition is run in collaboration with Kaggle, a platform for data analysis competitions. The competition runs over two years ending in early April 2013, offering a $3 million dollar grand prize plus smaller milestone prizes for the accurate prediction of the number of days a person will spend in hospital, *DIH* in short, using data from the preceding year.

Improving DIH prediction will allow healthcare providers to develop strategies to mitigate and manage hospitalisation risk, thus reducing unnecessary hospitalisation. A patient with a high DIH prediction may be monitored more closely and may be treated as an out-patient

rather than admitted to hospital as an in-patient, with priority care given to the particular patient. Similarly, a patient with a low DIH prediction may be provided with a lower Overall, this could potentially decrease the overall cost of healthcare. At this stage, the competition is focused on accurate numerical prediction, but it is foreseeable that the results can also be used to improve medical reasoning to support the informed development of healthcare strategies.

In order the conduct the predictions, Heritage has provided two years of data from over with the DIH for each year of data as the *training set*. Competition participants may use this data to produce prediction models. A *test set* containing the third year of data is provided, in which participants produce DIH predictions using the prediction model. Entrants may submit up to one set of predictions per day, and are provided with immediate feedback based upon approximately 30% of the test set data, known as the *feedback set*. The *scoring set* consists of the remaining 70% of predictions are reserved for the final judging and awarding of prizes. The performance of the model is determined using the root mean squared (natural) log error, where a lower error denotes higher performance .

1.2 Dataset

Heritage has made over 300 MB in training set data consisting of six tables available [4] [5]. The provided data is sourced from Heritage's data collected through its healthcare provision activities, with certain fields limited or removed to maintain the privacy and anonymity of its customers. For example, some of the continuous data fields are discretised into bins.

Table 1: Summary of raw data

Table	**Number of Fields**	**Number of Entries ('000)**
Members	3	113
Claims	14	27000
Drug Count	4	818
Lab Count	4	361
DIH in Y2	3	76
DIH in Y3	3	71

The data is relational. In the sample training data, member data may only be able for Y1 or Y2 or both and every member may have multiple claims. Each entry in the member table may have up two entries in the drug count and lab count tables, one for each year of claims. Where member data exists for a particular year, the members will have an entry in the Drug, Lab, and corresponding DIH table.

1.2.1 Members Data

The members data table contains data relating to personal characteristics of each member, and it is possible to derive information such as the approximate age of a member from this table. The objective of the thesis is to predict the DIH in the following year for each member in this table.

Table 2: Fields in members data

MemberID	ID assigned to each member. The IDs are random and not ordered in any meaningful way.
AgeAtFirstClaim	Age of the member when the first claim is made, generalised into one of 10 year periods.
Sex	Sex of Member - M/F.

1.2.2 Claims Data

The claims data table contains the health claims made by each member in a given year. Each entry specifies information including the member making the claim, the healthcare provider, and medical issue. Of particular interest are the diagnosis and treatment fields of the patient, in which detailed information is not given. Instead, only a high level category is provided. The Charlson index of a claim is also included, which measures the comorbidity of a patient.

Table 3: Fields in claims data

MemberID	Foreign key to MemberID in Members Data.
ProviderID	ID of healthcare provider.
Vendor	ID assigned to each vendor.
PCP	The ID of the primary care physician, the physician responsible for the patient.
Year	Year in which the claim was made: Y1, Y2, or Y3. Where the year is Y1 or Y2, the claim should be used as training data. Otherwise, the claim is part of the test data.
Specialty	One of 12 physician specialties, defined with consultation from the American Medical Association and the Royal College of Physicians andSurgeons of Canada.
PlaceSvc	Place of service: ambulance, home, inpatient hospital, independent lab, office, outpatient hospital, urgent care, or other.

PayDelay	Number of days from the healthcare service date to the date the claim is approved and paid by the healthcare network to the service provider or patient.
LengthOfStay	Length of stay grouped into caregories: days up to six days; (1-2] weeks; (2-4] weeks; (4-8] weeks; (8-12 weeks]; (12-26]weeks; more than 26 weeks (26+ weeks).
DSFS	Number of months since the patient made the first claim.
PrimaryConditionGroup	One of 45 codes to denote the primary condition of the patient.
CharlsonIndex	A measure of a condition towards overall health. Cumulative for up to a year.
ProcedureGroup	One of 17 procedure groups defined by Current Procedural Terminology.
SupLos	Where the length of stay variable is unavailable (~95%) of claims, SupLos is 1. Otherwise, 0.

1.2.3 Drug and Lab Count

This table provides a count of unique drugs and labs claimed by a member. Counts above the 95th percentile are truncated. Heritage has withheld much of information pertaining to the specific nature of drugs and labs, for the reason of privacy concerns. In the raw data, one member may have more than one drug or lab claim. To reduce the feature space, for each Member, I only keep the last entry from the year. This is justifiable because the DrugCount and LabCount fields simply provides a running total of the number of entries in the year. The months in which the claims are made are lost, but this is a reasonable compromise.

Table 4: Fields in drug count data

MemberID	Foreign key to MemberID in Members Data.
Year	Year of data: Y1, Y2, or Y3.
DSFS	Months since first claim.
DrugCount	Count of drug claims.

Table 5: Fields in lab count data

MemberID	Foreign key to MemberID in Members Data.
Year	Year of data: Y1, Y2, or Y3.
DSFS	Months since first claim.
LabCount	Count of lab claims.

1.2.4 DIH

This table provides the target response values to be predicted by the model. The values are between 0 and 15 inclusive, where a member's DIH is coded as 15 should they spend more than 14 days in hospital in a year (99th percentile). Approximately 76,000 members have a value for year 2 and approximately 71,000 members have a value for year 3.

Table 6: Fields in days in hospital data

MemberID	ID assigned to each member
DaysInHospital_Y2	Days in hospital for Y2 (to be predicted from data in Y1)
DaysInHospital_Y3	Days in hospital for Y3 (to be predicted from data in Y2)
ClaimsTruncated	If the member has had claims truncated in the previous year, then the value is 1. Otherwise, 0.

1.3 Related Work

While it has been established that machine learning techniques are often independent from the problem domain, investigation into the application of machine learning in related problems could bring insight into intricacies of a specific problem. In this section, existing work in numerical prediction for healthcare is reviewed, and the review is used as a basis for the overall process of days in hospital prediction.

1.3.1 Bertsimas et al (2008)

Bertsimas, Bjarnadóttir, Kane, Kryder, Pandey, Vempala and Wang [6] proposed using modern machine learning techniques to learn two years of health claims data and subsequently make predictions about the third year of data. The work differs in that the response variable predicted is the amount of medical expenses rather than DIH. As DIH and medical costs have been shown to be closely related in at least some occasions [7], this is quite relevant to my proposed work.

In the work, the training data available is more comprehensive than the data provided by Heritage. Specifically, claims are categorized into over 22000 codes for diagnosis and medical procedures, over ten times than the amount provided by Heritage. For the purpose of model generation these are further grouped into 218 and 180 higher level groups for claim diagnosis and medical procedures respectively. This ensures that the data is more manageable and also reduces the effect of inconsistent interpretations between medical practitioners. For each member, the claims are aggregated over a time period where a count of each claim category is taken. Also included are variables that could indicate the health of a member, for example, cost variables such as the amount spent on medical costs, gender,

and age. Members' medical costs in the next year, the target response variable, are partitioned into five buckets. This reduces the effect of outliers and noise albeit with the trade-off of some loss of information.

The predictions are made using three methods. The first method, the baseline method, is to simply use the healthcare costs of the previous 12 months as a forecast of healthcare costs for the next 12 months. This baseline is found to be better than a random assignment as previous healthcare costs would be an indicator of a person's overall health. This results in an 80.0% overall accuracy in determining the current bucket.

The second proposed method is classification trees. The trees are generated by recursively partitioning populations into smaller groups such that uniformity is increased. It is suggested that the benefit of classification tree is that results are easily interpreted and thus medically verifiable if necessary. This results in an overall accuracy of 84.6%

The last proposed method is to use clustering techniques, specifically, using a search-and-clustering engine EigenCluster. Members are clustered using only monthly cost data with recent months given more weight. Members with similar cost characteristics are grouped together. Next, further clustering is performed within the groups with medical data included. For each cluster, a forecast is assigned based on the known target values of the cluster. Clustering results in a lower accuracy of 84.3% when compared to trees, but when greater weight is given to high-cost patients classified into the wrong bucket, this outperforms trees. This is attributed to the hierarchical application of costs before medical information.

Overall, both trees and clustering outperform the baseline method in all performance measures examined. The paper concludes that machine learning can be applied on medical claims data to make predictions about a patient and thus form a basis for patient and healthcare management.

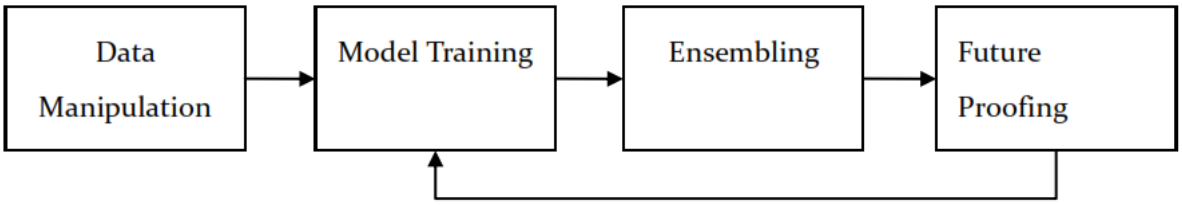
1.3.2 Axelrod, Brierley, and Vogel (2011)

Axelrod, Brierley, and Vogel [8] presented entries into the competition and have proposed solutions resulting in entries with the lowest mean square log error in the first milestone of the competition.

In their work, they describe a strategy to generate predictions consisting of four main processes. First, raw data is manipulated to follow a data schema suitable for typical supervised machine learning modelling techniques. Secondly, over 80 models are trained from the resultant data. Thirdly, ensembling is performed on the models. Lastly, they

describe *future proofing*, a process of making observations about the input data, target outputs, and models to make adjustments that could improve the generalisation accuracy of the predictive models. The previous steps are repeated as necessary, after conducting future proofing.

Figure 1: Process for days in hospital prediction



In the data manipulation process, the raw data consisted of members each with zero or more claims. As each target response variable relates to a particular member, the raw data was grouped over members with aggregate or summary values of the other data. With respect to the claims data, there are about 150 possible values amongst the medical fields associated with each claim. Each value forms a column in the manipulated data where an occurrence count over an observational period is given. In the paper, observational periods of one year and two years are used and the related entry in the manipulated table is labelled as such. Lastly, target variable is transformed to a log scale for the purpose of mathematical convenience, which allows minimisation of the root mean squared error rather than the root mean squared log error.

For the model training, over 80 models are generated using 6 methods with varying parameters. The four methods used are: Gradient Boosting Machines, Neural Networks, Bagged Trees, and Linear Models. Gradient Boosting Machine was found to be the most accurate individual algorithm, consistently achieving the lowest prediction error compared to the other algorithms. Truncation was performed on some of the predictions. This is because the predicted result may be outside the domain of DIH. Interestingly, a cap setting the lowest possible value to 0.02 rather than 0.00 showed minor improvements on the cross-validation accuracy.

The authors present a noteworthy method of ensembling. A linear combination of the model results were taken. However, of interest is the method used to find the weights (or coefficients). Firstly, out-of-sample predictions from each model are taken as the prediction set for the training data. Linear regression is performed on the candidate models to predict the out-of-sample values. To prevent overfitting, regression is conducted many times but

only 50% of candidate models are selected to participate in each regression. The mean of coefficients are taken from across each regression. Next, only the models that have one of the highest 20 weights are kept and the other models are excluded. The regression process is repeated after discarding the weights. The ensemble of models were found to achieve a higher generalisation accuracy than any individual model. Ensembling was found to be a defensive technique to prevent a single model from producing certain results that are very wrong.

Lastly, the authors perform future proofing. The predictive model uses historical data to predict future events. This is based on the assumption that the future will be the same as the past. The authors find that this is not always true for the data set. Future proofing is largely concerned with finding predictor variables that change over time. In particular, they have found that the distribution of the PayDelay variable tends towards lower values in later years compared to earlier years. Therefore, this variable was removed from the models.

1.3.3 Maelstrom (2011)

Maelstrom [9] also presented entries into the competition, achieving second place in the first milestone of the competition. However, his method of model blending was found to be superior to the ensembling technique described in [8] and was adopted by Brierley, Vogel, and Axelrod in the second milestone of the competition with improved results [10]. In order to prevent the introduction of bias in the blending, he incorporates Tikhonov regularization. Unlike the Netflix Prize, Heritage has not provided *probeset*, that is, a small labelled sample of the feedback and scoring data. The implication is that the parameter α can only be approximated using leaderboard data, which is more limited.

1.4 Thesis

The thesis is that the proposed multi-instance learning algorithm adds value to the prediction of days in hospital. This is sub-divided into the theses:

- **Chapter 2** introduces multi-instance learning and issues associated with common multi-instance learning techniques that prevent common multi-instance learning techniques to be used for days in hospital prediction. A novel solution for generalised multi-instance learning is proposed that empirically performs significantly better than benchmark algorithms.
- **Chapter 3** is concerned with transforming the data into a standard supervised learning representation so that common algorithms can be applied to show that the

performance of benchmark and comparable supervised learning algorithms do not materially outperform the proposed multi-instance learning algorithm.

- **Chapter 4** compares several methods for forming a committee machine, and the chapter shows increased committee machine performance when a multi-instance learning is introduced as a constituent model.

Overall, ordering of the theses is inspired by [8]. The work suggests that the strategy to achieve high performance solution requires using a variety of learning techniques, to create multiple candidate models that form a committee of learners. Thus, the use of multi-instance learning introduced in a similar context.

2 Multi-instance learning

2.1 Introduction

Multi-instance learning is a variation of supervised learning. In supervised learning, every instance is assigned a label. Conversely, in multi-instance learning, each instance belongs to a *bag*, where the bags are labelled. Maintaining a multi-instance representation allows more information to be captured and is also a more natural representation. Compared to supervised learning, it is also expected that there will be reduced loss of temporal knowledge. This is the motivation to conduct DIH prediction using multi-instance learning. However, current work in multi-instance learning requires some modification before this can be done.

2.2 Standard Multi-instance Assumption

I begin by introducing a trend to assume a binary classification problem where the target attribute $\Omega = \{\oplus, \ominus\}$.

Definition 1 (Multi-instance learning 1): Let dataset D be a set of elements $(B_i, F(B_i))$ where label $F(B_i) \in \Omega$. Each B_i contains a set of instances (X_{in}) from $\mathcal{X}$. Each instance has a hidden class label $G(X_{in}) \in \Omega$. Induce a learner $g: \mathbb{N}^{\mathcal{X}} \rightarrow \Omega$ for the hidden class label under the standard multi-instance learning standard assumption.

This is a common definition for multi-instance learning, as it is the definition first used when introduced by Dietterich et al [11]. The work was motivated by a problem in drug discovery. In the work, there are molecules (bags) which may be *musky* or not. Each molecule has different configurations (instances) in which not all may be musky. In the data, if at least one configuration of a molecule is musky, then the whole bag is musky. However, only bag labels are known. In the work, emphasis is placed on inducing a learner of whether a particular instance is musky, to understand the connection between molecule configurations and muskiness, rather than molecules and muskiness.

The definition has since been commonly tweaked for other areas [12]. To illustrate this, in a learner that attempts to determine whether an image is of a garden or not, a partition of the image containing grass denotes might the image is of a garden. The concept that the partition contains grass is of low importance, and the focus is on whether the image is of a garden or not. This is similar for the days in hospital prediction problem. where the aim is

not to find the label for an instance, but rather, to find the label of a bag based on its instances.

Definition 2 (Multi-instance learning 2): Let dataset D be a set of elements $(B_i, F(B_i))$ where label $F(B_i) \in \Omega$. Each B_i contains a set of instances (X_{in}) from $\mathcal{X}$. Each instance has a hidden class label $G(X_{in}) \in \Omega$. Induce a learner $f: \mathbb{N}^{\mathcal{X}} \rightarrow \Omega$ for the bag label under the standard multi-instance learning standard assumption.

In the definitions above, the relationship between the instances and bags are not clear. The standard multi instance learning assumption introduces this relationship more clearly; a bag instance is positive if and only if one more instances within the bag has a positive hidden class label .

Definition 3 (Multi-instance learning standard assumption): $F(B_i) \Leftrightarrow \exists\, X_{in} : G(X_{in})$

Notice that when learner g is available, then f can be immediately obtained. This is because $f(B_i) = g(X_{in}) \vee \ldots \vee g(X_{in})$. The converse is not true, when learner f is available g cannot be immediately obtained. As a note, the learner f is arguably more complex as it must encompass the logical relation in addition to g. This is a possible explanation for the desire for the assumption to hold and the aversion to deviate from the standard assumption.

In a study by Ray and Craven [13], it is shown that empirically, supervised learning is superior to multi-instance learning in domains where the multi-instance learning assumption is unrealistic. To revisit the earlier classification example, suppose a variation is to determine whether an image is a garden or a farm. The difference is minor and it would be difficult to for the presence of a single object to distinguish between the two. Instead, the learner should rely on all the instances, and/or the relationship within the instances, for example, the number of plants in the image and the proximity of the plants.

Similarly, consider a binary classification problem to predict that whether a patient will spend more than 14 days in hospital in the following year. There will be situations where a single instance is sufficient to assert the positive case, but in general, it is more likely that a combination of instances (claims) that collectively predict whether or not a patient will spend more than 14 days in hospital. Thus, the inductive bias of standard multiple instance learning algorithms is unlikely to be appropriate.

Under the standard assumption, for days in hospital prediction, either f or g may be learned to for successful prediction. However, as the standard assumption is inappropriate, I relax

the standard assumption. Doing so, the relation between f and g is no longer trivial, and this is an issue that will be addressed in this thesis.

2.3 Multi-instance Regression

Another consideration is how multi-instance learning can be used for numerical prediction. One method is to conduct a one-vs-many experiments for each value in the target domain. In one-vs-many experiments, a binary learner is formed for each distinct possible class. The classes are easily formed using a direct encoding. Using the direct encoding, each discrete numerical value in the target domain is a class. Alternatively, using an order encoding, where for every possible discreet numerical value i, an instance of value x is of a class C_i when $x \leq i$. An issue with using this method is that the predictions are strictly in one the discrete classes. This is an issue for DIH prediction. The performance measure penalises error exponentially, so it is wise to allow predictions over a continuous domain. A classifier would label a bag as the most likely class, but if there are two classes that are potentially the correct class, on average, predicting a value in between the two class values would increase the performance. This is the motivation for multi-instance regression. The definition of multi-instance regression assumes that the target classification attribute is real.

Definition 4 (Multi-instance Regression): Let dataset D be a set of elements $(B_i, F(B_i))$ where label $F(B_i) \in \mathbb{R}$. Each B_i contains a set of instances (X_{in}) from $\mathcal{X}$. One unknown instance in B_i is the primary instance (X_{ip}). Induce a learner $f: \mathbb{N}^{\mathcal{X}} \to \mathbb{R}$ for the bag label where $F(B_i) \Leftrightarrow \boldsymbol{w}^{\mathrm{T}} X_{ip}$.

In work by Ray [14], it is assumed that in each bag, there is one instance that is sufficient to produce the bag label, known as the primary instance X_{ip}. The other instances are thought of as *noise*. In the paper, the algorithm aims to find weight vector $\boldsymbol{w}$ for a linear multiple regression model $F(B_i) = w_1 x_{px_1} + \ldots + w_k x_{px_k}$ where x_{ix_k} denotes the kth variable, such that:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \sum_i [F(B_i) - \sum_k (w_k x_{ip_x})]^2$$

Observe that only the primary is used in the equation above, but in practice, the primary instance is not known in training time, the primary instance is approximated as the one within the bag that minimizes the total sum-squared error:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \sum_i [F(B_i) - \min_{\mathrm{n}} \sum_k (w_k x_{in_x})]^2$$

As iteration of every combination of instances is NP-hard (at least as hard as the problems in NP), the algorithm employs Expectation Maximization to search the hypothesis space greedily. Multiple linear regression is performed by initiating a hypothesis $\boldsymbol{w}$ randomly. Then, selecting instances from each bag that results the least total error according to the hypothesis (expectation), and reconstructing a new multi linear regression model based on the selected instances (maximisation). The expectation and maximisation steps are repeated until convergence.

While the work is faithful to the standard multi-instance learning assumption and contrary to my intensions, there is still relevance to days in hospital prediction as it presents insight into multi-instance learning for numerical prediction. The authors conclude that their work did not outperform ordinary regression, and suggest that the work is still significant as the method can be used as a baseline solution for future work. Thus, this is chosen as a baseline for the experiments.

2.4 Gradient Boosting Machines

In this thesis, at both levels Gradient Boosting Machines is used as a learner. While Gradient Boosting Machines can refer to a wide range of algorithms, I focus on the one introduced by Friedman [15]. The Gradient Boosting Machines use some form of regression as the basis function, while iteratively minimizing the loss function through additive basis functions to form a tree. There are several properties that make GBMs a good choice for the proof of concept. As the underlying basis learner of Gradient Boosting Machines are Linear Regression Models, this means the results are highly interpretable, and also allows for a theoretical analysis of the technique if it is required. The second property is that GBMs typically exhibit high generalisation accuracy which can be attributed to the inclusion of boosting techniques. GBMs also handle both categorical and numerical data, which avoids the requirement to transform the raw data. Lastly, feature selection naturally occurs when the selecting features to split a node on. This can be compared to a standard linear regression where the model considers all features of which some may provide no marginal benefit to performance. Overall, the properties of GBM make it a versatile learner [16].

2.5 MI Gradient Boosting Machine

I introduce a technique to conduct multi instance learning for numerical prediction, most easily referred to as *MI GBM*. The technique is explained in the context of days in hospital prediction. For days in hospital prediction, the strict assumptions of multi-instance learning

are relaxed. Rather than relying on a primary instance, the bag label learner is some unknown function f of the instances and instance labels.

Definition 5 (Generalised Multi-Instance Learning): Let dataset D be a set of elements $(B_i, F(B_i))$ where label $F(B_i) \in \mathbb{R}$. Each B_i contains a set of instances (X_{in}) from $\mathcal{X}$. Each instance has a hidden class label $G(X_{in}) \in \mathbb{R}$. Induce a learner $f: \mathbb{N}^{\mathcal{X}} \rightarrow \mathbb{R}$ for the bag label where $F(B_i) = f(X_{i1}, \dots, X_{in}, G(X_{i1}), \dots G(X_{in}))$.

In this case, there is no direct logical relationship assumed between the instance and bag labels. The relationship will be learned instead. Note that as f relies on $G(X_{in})$, a learner $g: \mathcal{X} \rightarrow \mathbb{R}$ must also be induced. This can be compared with multi-instance learning under the standard assumption, where only one of f or g needs to be induced.

To summarize the technique, there are two concepts to be learned - the instance labels, and the bag labels which are a function of the instance labels.

2.6 Attribute Representation

The instance representation is natural to the raw data, and hence the original data representation is maintained. For Gradient Boosting Machines, the algorithm is able to handle null and categorical values, there is also no transformation of null values.

For linear regression, null and categorical values are coded using indicator variables, also known as dummy variables. If a category has k possible values, then k dummy variables are created. Each dummy variables will have the value 1 to denote that the original variable belongs to the category, otherwise the value is 0.

For mathematical convenience, the natural log is applied to the response variable throughout the work, until the point where the prediction is produced where it is transformed back. The performance in the competition is measured using the root mean squared log error:

$$\varepsilon = \sqrt{\frac{1}{i}\sum_{i}\left[\ln\left(1 + F(B_i)\right) - \ln\left(1 + f(B_i)\right)\right]^2}$$

Given that most machine learning algorithm implementations typically include a setting that minimizes the squared error, the outcome is that a learner that minimizes the root mean square log error can be created fairly straightforwardly.

In the context of multi-instance learning for days in hospital prediction, a bag is a member in a given year. A member has claims, which are the instances. Drug and lab data is ignored to maintain simplicity.

2.7 Learning Instance Labels

The instances in the raw training data provided do not have instance labels. In a way, the instance label can be thought of the *contribution* it makes towards the bag label. As the notion of contribution might be difficult to grasp, this is explained by way of an analogy. Suppose that there is a bag of 5 bank cheques. The total value of cheques is $1000. The value of cheque towards the total value is the contribution. If there is no further information about the cheques, then the best guess is to assume a uniform distribution and put that each of the 5 cheques contributes $200.

Similarly, it is uncertain how each instance contributes to the bag label. I begin by making the best available, but not necessarily realistic assumption, that is, each within-bag instance makes an equal contribution to the bag label. Given that bag B_i has m instances with $F(B_i)$ days in hospital, then, put each instance label $G(X_{in})$ as $F(B_i)/m$.

At the instance level, the assumption is maintained that instances labels are independently and identically distributed $\forall\ G(X_{in})\ G(X_{im})\ : E(\ G(X_{in})\ ) = E(\ G(X_{in})\ |G(X_{im}))$. Although it is perhaps possible that each claim's contribution relies on previous claims, the data is sparse in that on average, no given sequence of instance appears more than twice. Hence, a more complex model would be subject to overfitting. The algorithm compensates for this at the bag level learner, as explained in the next section.

Next, the learner $g: \mathcal{X} \to \mathbb{R}$ for the instance label is induced using the training data. A Gaussian distribution and choose least-squared regression as the basis function is assumed.

Observe that in a Gradient Boosting Machine (or with most other learners), similar inputs are expected to somewhat converge to the same output. Without loss of generality, suppose there are exactly two equal explanatory input vectors in a dataset, with different responses n and m. Given the learner of a Gaussian linear regression model, the prediction for the input vector would be exactly $(n + m)/2$, the value that minimizes residual error. The differences between the original responses and the new response are attributed to the training data being incorrect in the first place (systematic error), or, random variation, or some combination of both. Note that as random variation decreases by increasing (and controlling) the number explanatory variables, to make accurate predictions there would

ideally be enough explanatory variables to decrease the random variation to zero. In the belief that there is more error arising from systematic error rather than the lack of explanatory variables, replacing the original responses with $(n + m)/2$ would improve the accuracy of the original dataset. Furthermore, if the explanatory variables are in fact sufficient, then given enough data the response variables are expected to converge towards their true values.

By the same reasoning, the training data contribution levels is updated with the learner's prediction as it is unlikely that the original prediction of the contribution levels are accurate. This is performed at the expense of allowing for random variation, in the view that systematic error is reduced. Adjustment of training data is necessary as the contribution levels in the training data are subsequently relied on to learn f.

Finally, the learner is used to predict contribution levels for the test set data, in preparation for the next step.

2.8 Learning Bag Labels

Recall in the previous section a bag B_i has n instances with $F(B_i)$ days in hospital, each instance label $G(X_{in})$ is originally assumed to be $F(B_i)/n$. It follows that, $G(X_{i1}) + \cdots + G(X_{in}) = F(B_i)$. Strictly following the trend of previous work, it would be arguably correct to conclude, $g(X_{i1}) + \cdots + g(X_{in}) = F(B_i)$. With respect to my hypothesis, this argument is rejected.

Earlier, instances labels are treated as independently and identically distributed. Maintaining this assumption would be contradictory to the motivation of this thesis, as this would in fact suggest that than summing the instance values is sufficient, in which case the multi-instance learning assumption is only modified, but not generalised. Considering this, I reintroduce the relationship between within-bag instances to the bag label learner f, by including the original instance vectors and the vector labels to the input. Thus, f is able to somewhat offset the error arising from originally maintaining the i.i.d. assumption.

To illustrate the two step process, $f(X_{i1}, \dots, X_{in}, G(X_{i1}), \dots G(X_{in}))$ is best decomposed into constituent functions $\{f_1, f_2\}$.

The input to f_1 may vary in length due to different number of instances in each bag. As most learners, including Gradient Boosting Machines naturally work with a fixed length input

vector, I use f_1 for the purpose of transforming the variable length vector into a fixed one. The result of f_1 forms the input for f_2 with exactly 5 items, namely:

- Sum of instances labels
- Minimum of instances labels
- Maximum of instances labels
- Standard deviation of instances labels
- Count of instances

The sum of instance labels is included customarily as it is the expected bag label when strictly following the original estimation of instance contribution. The remaining variables describe the relationships of within-bag instances. To an extent, the variables also describe the bag properties. For the purpose of days in hospital prediction, this concept is kept simple, although there is no obstacle in including extra variables that might capture more bag properties or instance relationships. As an example, the variables might include a few of the highest instance labels and smallest instance labels, rather than just one each. More ambitiously, the learning algorithm could support varying input lengths.

Having formed f_1, learner f_2 is induced with the training set. Again, the learner is a Gradient Boosting Machine that assumes a Gaussian distribution and uses least-squared regression as the basis function. The learner is then used to generate the DIH predictions for the test set.

2.9 Experiment

This sections presents an empirical comparison of the proposed technique and benchmark techniques when applied for days in hospital prediction.

2.9.1 Experimental Setup

In the experiment, I evaluate the performance of ML-GBM for days in hospital prediction, and use two benchmark multi-instance learning algorithms as a comparison. Only one learner is trained using each algorithm, as the intention of these experiments is not to optimise the predictions or determine the behavioural trend of the algorithms in changes of parameters; the intention is to provide a proof of concept of the proposed algorithm.

For the Gradient Boosting Machines, there are three parameters that require the require setting: number of iterations, the learning rate (shrinkage), and the tree depth. The number of iterations is set at 4000, the learning rate is set at 0.01 and tree depth is set at 4. Values in

of this range are found to perform well empirically [15] [17]. The *gbm* package for R is used as the implementation.

There is no freely available implementation for the MI Linear Regression. Thus, a custom implementation using a combination of T-SQL, R, and .NET is developed for the purpose of these experiments.

The results are trained using the training set data to minimise test set RMSLE. Even though predictions are made for the test set data, consisting of feedback set and scoring set, only the feedback set error is readily available. The scoring set error is held for awarding the prizes. The performance throughout the thesis is evaluated using feedback set RMSLE.

2.9.1.1 MI GBM

MI GBM is the proposed multi-instance learning algorithm. I train a learner using this algorithm, and it is compared with other multi-instance algorithms in this chapter, as well as a constituent learner in committee machines in Chapter 4.

2.9.1.2 MI GBM (Instance Only)

In this benchmark, the instances are formed using the same method as MI/GBM. However, f is not learned on training data but is instead the sum of instance labels. This benchmark attempts to test the hypothesis of whether there is a benefit from learning f rather than assuming a function.

2.9.1.3 MI Linear Regression

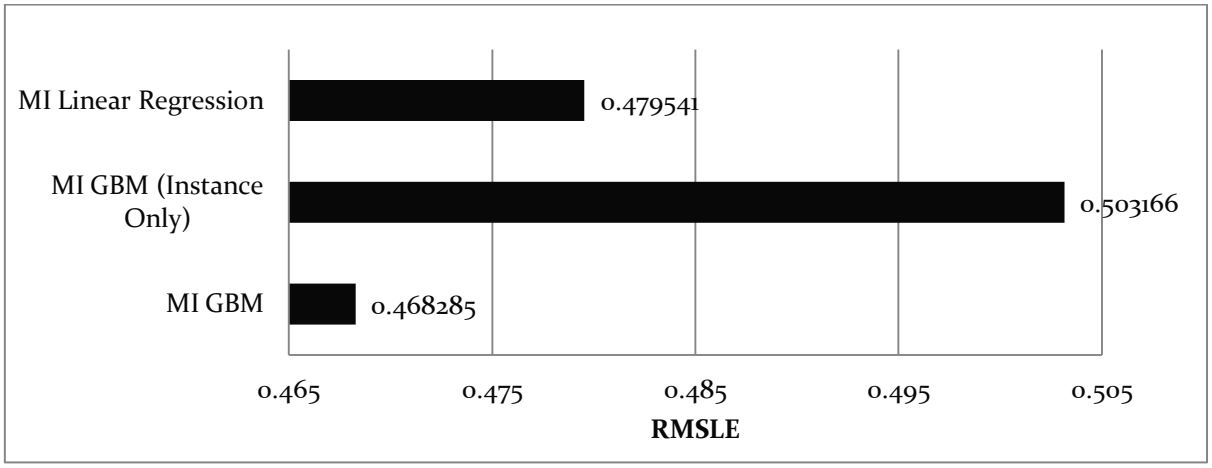
This benchmark is the same as work in [14] where the assumption is that one primary instance can predict the target. As the algorithm is non-deterministic (hypothesis are initialised randomly), this is repeated 10 times and the result with the lowest training error is retained. This benchmarks attempts to test that considering all instances will outperform selecting a single primary instance.

2.9.2 Results

It is worth noting that the MI Linear Regression in fact performs significantly better than MI GBM (Instance Only). The algorithms both incorporate a different assumption. In MI Linear Regression, it is assumed that a single primary instance label should be responsible for the bag label. In MI GBM (Instance Only), the assumption is that all instance labels contribute to the bag label, and the bag label is strictly defined as the sum of bag labels. It is not possible to state whether either assumption is stricter, because neither assumption contains

the other, thus they are not directly comparable. However, the empirical results show that for days in hospital prediction, both assumptions introduce error but the latter introduces more error. Overall, both can be compared with MI GBM where neither assumptions are present. The proposed MI GBM algorithm performs better than both of the benchmarks, as expected.

Figure 2: Comparison of multi-instance learning performance



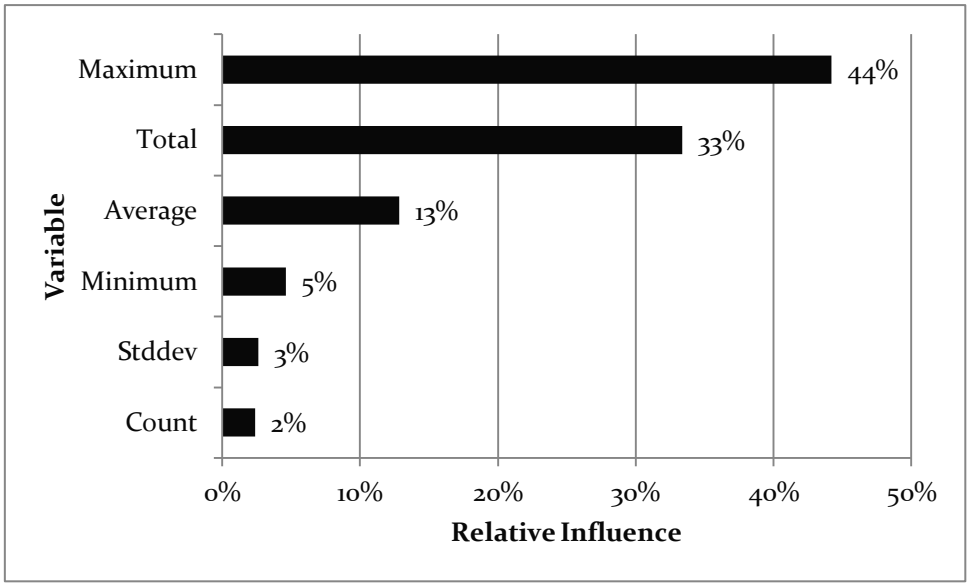
MI GBM performs better than MI Linear Regression. This is consistent with earlier beliefs that the selection of a primary instance merely relaxes, and not generalises the standard learning assumption. This comparison shows that there is, for days in hospital prediction, improvement from relaxing this assumption.

MI GBM also performs significantly better than MI GBM (Instance Only) variant where only a learner is present at the instance level. The is consistent with the hypothesis that it is beneficial to generalise the multi instance learning assumption by learning the relationship between instance and bag labels, rather than assuming one. This is illustrated by Figure 2: Comparison of multi-instance learning performance, which shows the relative influence of each variable in MI GBM. The relative influence describes the reduction in squared loss error attributed to a variable, relative to other variables.

In MI GBM (Instance Only), only the *total* variable is used as it is the only one available. This can be contrasted with MI GBM, where the *total* variable has the second highest relative influence of 33%. The bag learner exploits the availability of other variables, including the *maximum* variable which provides the highest relative influence. As approximately 67% of the relative influence is attributed to variables aside from *total* and the variables explain the relationship between within-bag instances, the result is consistent with the motivation of

learning bag-level labels as introducing the bag level learner that encapsulates the additional variables improves the accuracy for days in hospital prediction.

Figure 3: Variable relative influence in MI GBM



3 Supervised learning

3.1 Introduction

This section is concerned with representing the days in hospital prediction problem as a standard supervised learning problem.

Definition 6 (Supervised Learning): Let dataset D be a set of elements $(B_i, F(B_i))$ where label $F(B_i) \in \mathbb{R}$. Induce a learner $f: \mathbb{N}^{\mathcal{X}} \rightarrow \mathbb{R}$.

As introduced earlier, the DIH prediction problem is most easily represented as a multi-instance concept. In supervised learning the target response $F(B_i)$ is to be predicted by a single input vector. This is an issue because for every target response (DIH), there may be multiple claims. Part of the aim is to devise a method transform this multi-instance concept into a single vector. The method used to transform the data is influenced by work in ?.

The motivation for applying supervised learning algorithms on the data is that multi-instance learning research is less established than supervised learning. On a practical level, supervised learning can be applied easily and quickly as the supporting tools and literature are readily available. The results can be used for assessing the performance of the transformation method by comparing the supervised learning results with multi-instance leaning results, with focus on days in hospital prediction. Previous works provide empirical comparisons between supervised and multi-instance learners over equivalent datasets but so far the work is largely limited to multi-instance learning under the standard multi-instance learning assumption [13].

3.2 Attribute Representation

Raw data attributes are either categorical or numerical. Numerical data remain numerical. The treatment of categorical attributes depends on whether the attributes are part of the claims data or the other data. This is because there may be multiple claims for each target response.

For single-instance data, the categorical attributes are transformed into numerical representations so that each distinct category is assigned an integer. This is for convenience, due to the large number of variables it is easier to type integers compared to strings. Depending to the algorithm used, the subsequent treatment varies. For neural networks and linear regression, they are treated as factor variables and use dummy coding, whereas for

GBM and Random Forest, they are also treated as factor variables but no dummy variables are created due to the algorithms are able to handle categorical data naturally.

For the multi-instance claims data, the attributes are not directly included in the learner's input vector. Instead, summary or aggregate statistics are calculated for the input data. As the relevant aggregation functions work with categorical attributes, it is not necessary to assign an integer. This is explained in further in the section below.

As with multi-instance learning, natural log is applied to the response variable.

3.3 Instance Representation

For the heritage dataset, I form a single input vector B_i for each corresponding days in hospital using the method inspired by [8]. To do so, where the explanatory variables may span across multiple instances, aggregation functions are applied on the instances to form values that attempt to summarise the instances. Aggregation will lead to some loss of predictive ability of the variables, in this case, the temporal and ordering context is reduced. Conversely, it could be counter-argued that to safeguard against overfitting, a simpler model may be preferable. In the experimentation, I take a greedy approach to estimate a suitable level of aggregation.

3.3.1 Claims Data

In work by [8], the claims data are aggregated into observational periods. There may be legitimate motivation for this - to reduce overfitting and model training times. Claims from different months within a year are grouped together. As this may lead to loss of temporal information, their approach is tweaked. It is possible to aggregate the claims with members but also partition the claims data over smaller periods of time, for example, months or quarters.

To perform the aggregation, I introduce $P \in \{1, \dots, 12\}$ to denote the number of months that form the aggregation observational period. Recall that each claim instance includes the month at which the claim was made and that each bag contains 12 months of claims. I then partition the bag of claims by the observation period. Aggregation statistics are afterwards calculated for the partitions rather than the whole bag, as an effort to maintain part of the temporal data.

Definition (Partition of bag over observational period): Let B_{ik} be partition k of bag B_i containing instances X_{in} observed in month $m(X_{in})$. Then, $X_{in} \in B_{ik} \Leftrightarrow k(12/P) \leq m(X_{in}) < k(12/(P+1))$.

For example, if $P = 3$ then an instance from the 3rd month would fall in the first partition and an instance from the 4th month would fall in the second partition. It also follows that each bag has a equal number of partitions, as regardless the bag, $m(X_{in}) \in \{1, \dots, 12\}$. Subsequently, if a fixed number of variables for each partition is formed using aggregation statistics of the claim instances, then the total number of variables is also fixed.

3.3.1.1 Categorical Fields

Each claim contains a field for diagnosis, treatment and provider. These fields are categorical, and have a finite number of possible values. For the purpose of aggregating categorical fields, null values are considered as a category. Let $H_s(X_{ik_n}) \in S$ be a categorical field in the claim for the set of possible values in the category S and X_{ik_n} is the nth claim of the kth partition. For each $C \in \mathcal{P}_{=1}(S)$, let $s \in C$ and add a value to the input vector, using indicator notation:

$$b_{ik_s} = \sum_n I_C(H_s(X_{ik_n}))$$

This results in a value for each partition for each possible value of each category. The value is a count of instances.

3.3.1.2 Numerical Fields

For the numeric fields, the sum is taken instead. Let $H_M(X_{ik_n})$ be a numerical field in the claim for the field M and X_{ik_n} is the nth claim of the kth partition. For each $s \in S$, the input vector includes a value:

$$b_{ik_F} = \sum_n H_F(X_{ik_n})$$

In certain cases, a numerical entry has a null value. In these cases, the entry is ignored in the calculation of the sum. A value is then added to the input vector containing a count of the number of null values.

3.3.2 Other Data

The other fields have a one-to-one correspondence with the response variable (DIH). Thus, they are simply added to the input vector. This includes:

- From the drugs table
    - Count of drug claims
    - Month of the last drug claim
- From the labs table
    - Count of lab claims
    - Month of the last drug claim
- Sex of member
- Age at first claim of member

3.4 Learning Algorithms

Here, I review the supervised learning algorithms is run on the dataset. Different learners exhibit different properties. This means that certain methods will work better than the other for a particular dataset. This section introduces the learning algorithms that are used.

3.4.1 Gradient Boosting Machines

The Gradient Boosting Machine is included as a continuation from the previous chapter. The supervised learning gradient boosting machine is analogous to MI GBM.

3.4.2 Linear Regression

Linear Regression is used as a benchmark solution. The loss function is chosen to be the sum-squared error, in line with the competition's error function. Linear Regression is included as it is comparable to the baseline multi-instance learning algorithm where a primary instance is chosen. Linear Regression is conducted using R.

3.4.3 Random Forests

The Random Forest is an ensemble classifier that has characteristics of bagging and random selection of sub features [18]. The forest creates many base learners, typically decision trees. Each tree is trained on a bootstrap sample of the training data. At each node, the decision may only be split on a feature from a random subset of the features. Whilst the random forest is a classifier, it is suitable for numerical prediction. Each number is treated as a class. The base learners may predict a different value where the average is taken to be the final

answer. As the number of base learners is typically large, the output domain is, in a relaxed sense, continuous. The *randomForest* package for R is used as the implementation.

Random forests are not analogous with any of the multi-instance learning algorithms previously tested. However, as the final theses is to test the effectiveness of a committee machine, the random forest is included as a candidate learner as this helps replicate a authentic committee machine with a diverse range of candidates.

3.4.4 Multi-layer Perceptrons

Multi-layer perceptions are artificial neural networks. In this section, neural network will refer to a feedfoward multi-layer perception network.

Neural networks are typically endorsed for their ability to work with non-linearly separable problems [19]. For a binary classification problem, linear separability means that one class can be separated from the other using a single hyperplane. In a strictly linearly separable problem, logistic regression or a single perception is likely to hold the best inductive bias. In neural networks, each hidden node has some transfer function, for example, a sigmoid or a step function. This introduces non-linearity to the network.

Conversely, neural networks are relatively computationally intensive and require long training times. In general, it is not feasible to perform ensembling on neural networks on non trivial problems.

Similar to the random forests, the properties of neural networks are quite different from gradient boosting machines and linear regression, and are included in preparation for the next step to replicate an authentic committee machine with a diverse range of candidates.

3.4.4.1 Backpropagation

Backwards propagation (backpropagation) is an algorithm for training neural networks. The weights of a neural network are initialised randomly. This is followed by a number of iterations, or epochs. Each epoch has a propagation step and a weight update step. In the propagation step, the network is used to generate predictions for the training data. Afterwards, the error is propagated backwards to calculate the delta of the hidden and output units according to the delta learning rule. This credits the error or performance to the individual units. In the weight update step, each weight is updated to move in the direction opposite to the error. This has the expected effect of descending down the gradient of the error function. The epochs are repeated until the stopping criteria is met, which typically

specifies a number of iterations to perform training or the training error that the network should reduce to.

In the experiments, I select the stopping criteria so that the training occurs for 10,000 or until the training error reduces to an RMSLE of 0.01 or less, whichever occurs first. The *neuralnet* package for R is used as the implementation.

3.4.4.2 Neuroevolution of Augmenting Topologies

Neuralevolution of Augmenting Topologies (NEAT) is an algorithm that creates a neural network using a genetic algorithm [20]. A disadvantage of backpropagation is that it performs gradient descent, meaning that there is a high possibility that the hypothesis will converge to a local optimum, instead of the global one. Furthermore, backpropagation requires tweaking of parameters such as the network configuration and learning rate. NEAT seeks to solve these shortcomings by searching a larger hypothesis space with a high stochastic element.

The NEAT algorithm begins by randomly initialising a generation of simple neural networks. Each network is evaluated using a performance measure, in this case, RMSLE. A new generation of networks is created from the previous generation, where the neural networks that score better have a higher probability of being included. Furthermore, in the new generation, both the weight and topology of the neural networks may mutate. This process is repeated until the stopping criteria is met. *SharpNet*, a .NET implementation of NEAT is used.

3.5 Experiment

3.5.1 Experimental Setup

There are two parts to the experiment. The first part involves selecting the observational period. The motivation is to optimize the observation period such that it minimizes the error. This allows for a fairer comparison between the multi-instance learning algorithms and the supervised learning algorithms, given that the transformation method summarizes the data causing a loss in temporal information.

The second part of the experiment involves transforming the original data into a supervised learning data set, and using supervised learners to make predictions. The experiments lead to the training of 18 supervised learners using a variety of algorithms and parameters.

Again, the training is peformed on training set data to minimize training error. The results are evaluated using feedback set RMSLE, as with the multi-instance learning experiments.

3.5.1.1 Selecting Observational Period

Ideally, a range of observational periods would be tested for each learning algorithm. However, this is impractical as the number on experiments to conduct would scale linearly with the number of different observational periods. This may not seem like a lot, but setting $P = 2$ instead of $P = 1$ doubles the number of experiments.

The observational period is selected greedily. I create a Linear Regression model for P=3, P=6, P=12. The observational period of the model that has the least error is used for all other models. As this part of the thesis is intended to be exploratory, no ensembling or more complex modelling is performed.

3.5.1.2 Learning Algorithms

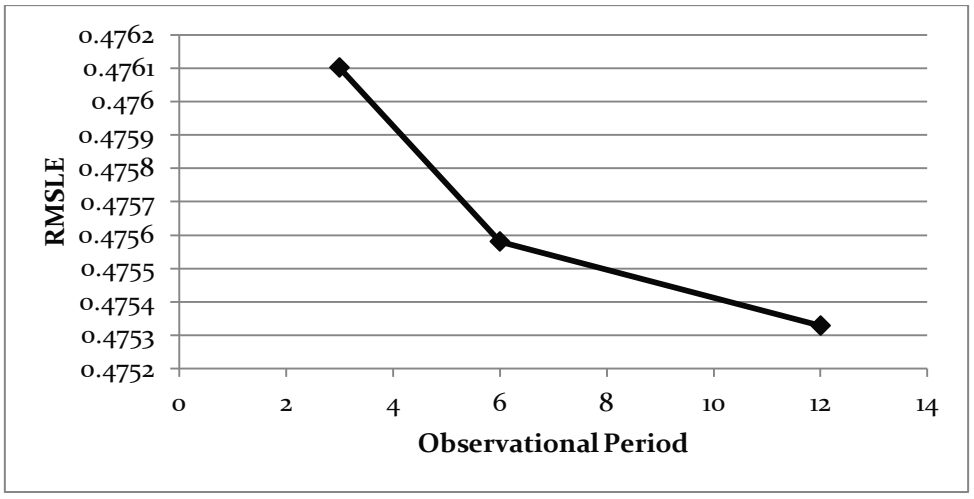
The second part of the experiment is to create the learners. Firstly, the dataset is transformed using the observational period selected in the earlier experiment. Then, I train supervised learners using the training data using the algorithms explained earlier, namely, Backpropagation, Neuroevolution of Augmenting Topologies, Random Forests, and Linear Regression.

3.5.2 Results

3.5.2.1 Observational Period

From the linear regression model, it appears that an observational period of 12 months has the least error. This value is used to transform the data for the supervised learning algorithms. This suggests that for a linear regression model, a lower number of features is preferred, because a shorter observational period increases the number of features. A typical interpretation would suggest the training set is too sparse relative to the number of features, even though a larger number of independent explanatory variables increases the potential to provide better accuracy predictions. According to the trend, the RMSLE might be reduced further with an observational period above 12. Unfortunately, this is constrained by the problem domain where a prediction must be made for each year, that is, 12 months.

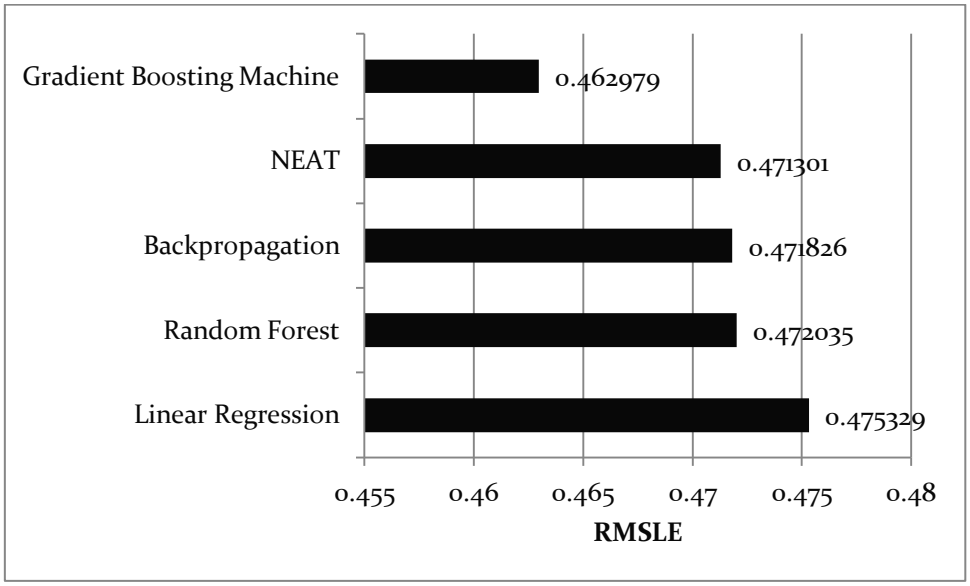
Figure 4: Performance at different observational periods



3.5.2.2 Supervised Learners

The graph shows the learners with the highest performance amongst each of the supervised learning algorithms. The parameters of the algorithms and their corresponding results are available in the appendix. The parameters for the multi-instance Gradient Boosting Machine is selected to imitate the equivalent best Gradient Boosting Machine. The choice of parameters used for Gradient Boosting Machines are also induced by trial runs of the experiment. Conversely, the settings for NEAT, Backpropagation and Random Forest learners are selected somewhat heuristically and the selection process is less informed, which may explain the poor performance. At this point, I stress that the comparison of supervised learners is only provided for curiosity, as the training of supervised learners apart from the Linear Regression and Gradient Boosting Machine is only incidental to the thesis.

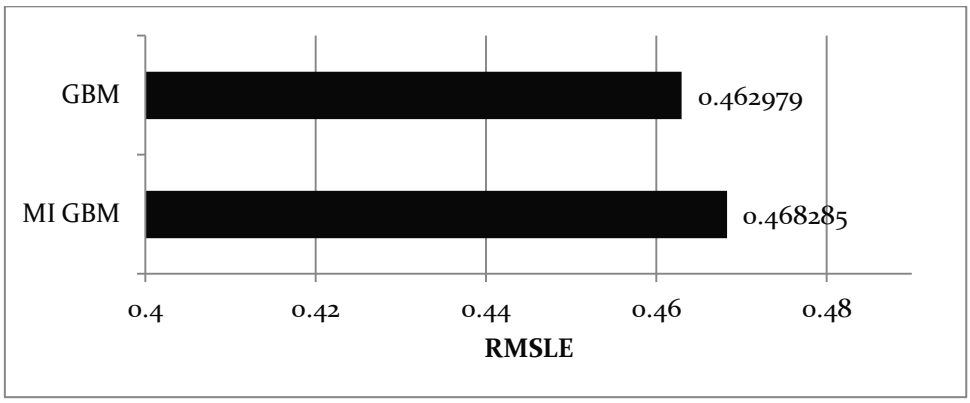
Figure 5: Comparison of supervised learning performance



3.5.2.3 Gradient Boosting Machine

MI Gradient Boosting Machine performs almost as well as the supervised Gradient Boosting Machine. This does not take into account several disadvantages of the supervised Gradient Boosting Machine. For supervised learning, it is necessary to transform the multi-instance data into a supervised representation, which is unnatural to the underlying concept. This is not difficult with days in hospital prediction, but may not be as trivial for other problems. While the transformation here does in fact enable the a supervised learner to work slightly better than MI GBM, it is quite clear that there is some loss of information. Thus, in a learning problem that is reliant on the multi-instance concept, then MI GBM may be preferred.

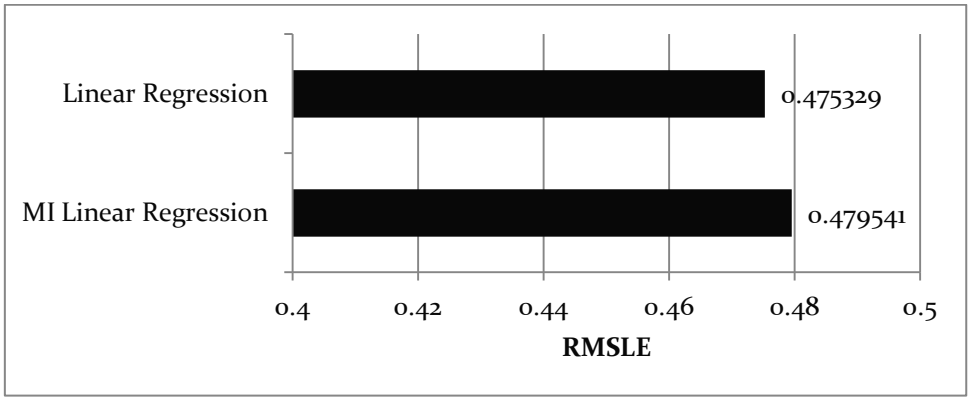
Figure 6: Comparison between GBM and MI GBM



3.5.2.4 Linear Regression

The supervised linear regression performs better than the MI Linear Regression. The result is consistent with earlier findings by X that standard linear regression performs better than multi instance linear regression where a primary instance is selected.

Figure 7: Comparison between Linear Regression and MI Linear Regression



4 Committee Machine

4.1 Introduction

A committee machine combines the multiple candidate models to form a single prediction. As an analogy, a "committee" of decision makers are on average, expected to form a better opinion that a single decision maker. A similar outcome is expected in the context of machine learning. The reasoning is that using a combination of different candidate learners increases diversity, each with their own strengths and weaknesses. By forming a committee, the learners complement each other, to provide better results on average. Undoubtedly, some of the learners already use some form of ensembling, including the gradient boosting machine and the random forest. These learners introduce diversity through selection or drawing subsets of training data, but maintain the same base learning algorithm. In this instance, I stress that this chapter is focused on forming a heterogeneous learner, using the existing learners from previous section as constituents.

Committee machines, when compared to single learners, are generally more complex, having longer training times, and restricted to batch learning. However, a committee machine almost always outperforms a single learner in accuracy. Hence in the context of competitions, where accuracy is highly regarded, the marginal effect from the addition of a constituent learner is of high relevance. This is the focus of this chapter.

4.2 Linear Combination of Models

In this chapter I limit the research to linear ensembles, or a linear combination of models. Linear ensembles are arguably the most straightforward method to form a committee. More complex ensembling methods may have dynamic structures, such using a mixture of experts. Jacobs, Jordan, Nowlan, and Hinton [21] proposed introducing a learner that allocates tasks to experts. The learner acts as a gating functions, also incorporating the explanatory variables as an input. Although these methods may improve generalisation accuracy, I choose to exclude more complex committee machines to focus on investigating the benefit of using a multi-instance learner as a constituent. With complex structures it would be less assuring that any differences in performance is attributed to the inclusion of a multi-instance learner rather than the method of forming the committee.

Definition (Linear ensemble): Let $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ be constituent predictions formed by various learners. Then, linear ensemble $\boldsymbol{y} = w_1\boldsymbol{x}_1 + \cdots + w_n\boldsymbol{x}_n$.

It follows that the hypothesis space for a linear ensemble contains all permutations of real values for $w_1, \ldots, w_n$. I include additional constraints, $\sum w_n = 1$ or $w_n \geq 0$. These constraints potentially prevent linear ensembles from overfitting. The constraints maintain the assumption that the inductive bias of the learners are already correct. Furthermore, the constraints also reduce the hypothesis space which potentially reduces the computation.

4.2.1 Baseline

In the baseline linear combination, the committee prediction is the average of all candidate model predictions. No training data is used to learn the weights. For days in hospital prediction, all weights are set as 1/18 for the committee machine excluding MI GBM and as 1/19 for the committee machine excluding MI GBM.

4.2.2 Linear Regression

Here, the weights are learned using linear regression. Using the training data, I find $\boldsymbol{y} = w_1\boldsymbol{x}_1 + \cdots + w_n\boldsymbol{x}_n$ such that the training RMSLE is minimized. To do so, I use the Lawson-Hanson algorithm for non-negative least squares regression as implemented in the *nnls* package for R. In the regression, weights will be heavily weighted towards learners that have high training accuracy, hence the learned weights are not applied directly. Instead, I use the Bayesian average with the weights from the baseline ensemble as the prior, with equal weighing between the two weight vectors. Regularisation is then performed on the weights so that they sum to 1, to maintain the earlier constraint. Lastly, I apply the weights on the test set predictions to form the committee machine prediction.

4.2.3 Bagged Generalised Linear Model

Bagging is an ensembling technique that combine weak learners to form a single hypothesis in the view that it will be a strong learner by random selections of training data [22]. In bagging, giving training set D of size m, each iteration i uses a new training set D_i by sampling m' items from D with replacement. Typically, the size of D is set to $m' = m$, in which $1/e$ samples are expected to be duplicates. For linear regression, each iteration finds the weight vector that minimizes RMSLE. Afterwards, the mean of the weights from each iteration is retained. This weights are then applied to the test set predictions to form a linear combination of predictions. Notably, this is different from work by [8]. In their work, the candidate models are sampled instead of the training data set. I do not use this method because of its perceived limitations. Bagging is more effective when the bootstrap samples are diverse. As there are a small number of learners, specifically 18 supervised learners and 1 multi-instance learner, then the bootstrap samples are likely to be similar. In contrast, there are over 100,000 instances to sample from the training set data.

For the purpose of the experiment, I follow the convention to pick N as the number of training instances. This results in the expectation that the proportion of instances duplicated will be $1 - (1/e)$. I perform 100 iterations, meaning that the probability of some instance not considered in any bag is $(1/e)^{100}$, almost a negligible figure.

4.3 Experiment

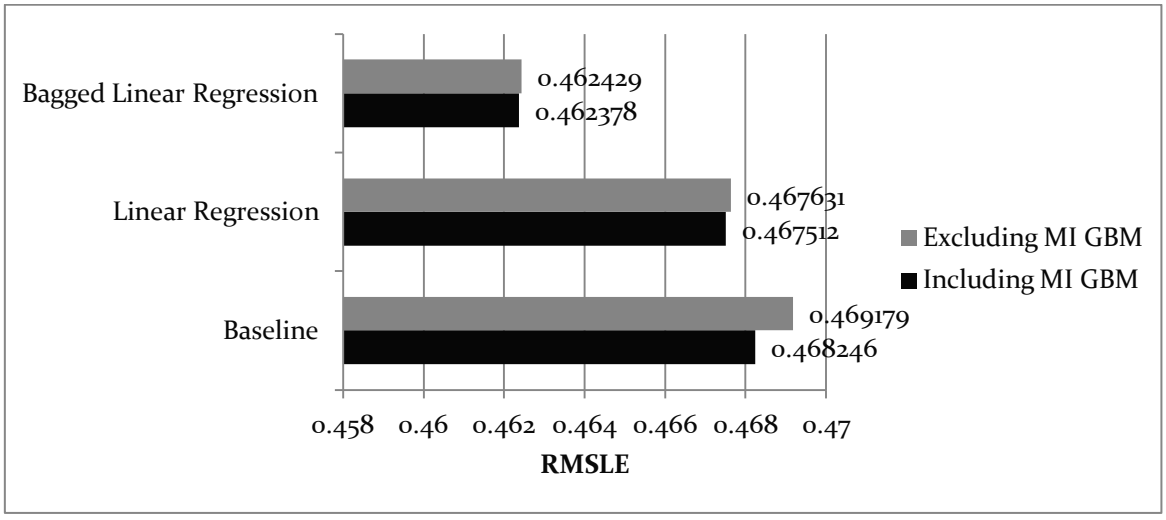
4.3.1 Experimental Setup

The purpose of the experiment is to assert that there is a marginal improvement from the inclusion of MI GBM is a committee machine. I train a total of 6 different committee machines. There are 3 methods presented earlier for learning the weights of a linear ensemble. For each method, I form one committee machine that includes the prediction from MI GBM, and one that does not. All the committee machines are also trained with predictions from the 18 supervised learners. Again, feedback set RMSLE is used as the performance measure and training is conducted to minimize error on the training data.

4.3.2 Results

The final weights for the Linear Regression and for the Bagged Linear Regression are presented in the Appendix. The weights for the baseline are obvious and hence there is no table for the weights. The results consistently show that inclusion of MI GBM in a committee machine improves the committee machine performance when compared to a committee machine that does not include MI GBM. The principle of committee machines is that increasing diversity increases performance. Thus, the increased performance from the inclusion of MI GBM suggests that the learner is uniquely different from supervised learners.

Figure 8: Performance improvements with inclusion of MI GBM in committee machines



5 Conclusions

5.1 Summary

The final predictions from the bagged linear regression committee machine with MI GBM achieves a leader board RMSLE of 0.4.2378. placing it at 160th place out of close to 1400 entrants as of the 15th October, 2012. A benchmark that predicts all members will spend 0 days in hospital places between 1155th and 1156th place. This is a reasonable attainment, given the focus of the thesis is to introduce a multi-instance learning algorithm in the context of days in hospital prediction, rather than maximising learner performance. In particular, the thesis introduces a method to induce a multi-instance learner where:

- Bags in the training data are labelled.
- Instances in the training data are not labelled.
- Learner outputs bag labels from tests samples.
- Bag labels are numeric and continuous.
- Instances are not necessarily independent from other within bag instances.

MI GBM performs better than the benchmark multi-instance learning algorithm where an instance from each bag is selected to be the primary instance, demonstrating the benefit of considering all instances. MI GBM also performs better than MI GBM (Instance Only), where the bag label concept is not learned. This verifies the rationale of learning the bag level concept rather than relying on some assumption.

In the experiments, the multi-instance learning concept is transformed into a representation suitable for supervised learning. It turns out that for days in hospital prediction, using summary statistics of the instance variables is not worse than maintaining the multi-instance representation. Even so, MI GBM maintains a natural representation which may be preferred in some situations.

The committee machines demonstrate how an ensemble of diverse learners performs better than any single learner. For days in hospital prediction, MI GBM achieves performance just under a single-instance GBM. A committee machine including MI GBM outperforms a committee machine excluding MI GBM. This suggests high distinctiveness of the MI GBM solution. In other domains where high performance is required, MI GBM may be used in the context of a committee machine to improve performance, after giving consideration for the increased complexity and reduced comprehensibility of the models.

5.2 Limitations and Future Direction

The thesis currently lacks comparability with existing research, due to the unique nature of the machine learning problem. The UCI Machine Learning Repository currently only holds two datasets for multi-instance learning, both of which follow the standard multi instance learning assumption. To improve the comparability, future work could involve creating artificial data for the problem. The same data would be made freely and publically available to improve the accessibility of multi instance learning research. Notwithstanding the experimental limitations, there are certainly many domains where the method is applicable. Earlier work in image classification focused on predicting a binary class value. However, this could be extended to predict probabilistic values instead. MI GBM could be investigated for use on other domains.

The work has been focused on learning the bag level concept by splitting it into a bag learner and an instance learner. While this is the focus for days in hospital prediction, in some cases the interest is in learning the instance level concept, that is, accurately predicting instance labels. This is the case for the drug activation problem that first motivated multi-instance learning. At present, MI GBM does not intend to make highly accuracies instance label predictions, and relies on the bag level learner to make an accurate bag label prediction. This is demonstrated by the relatively poor performance of MI GBM (Instance Only). Future work may shift the focus to learning accurate instance labels.

Works Cited

[1] World Health Organization, "World Health Statistics 2011," World Health Organization, 2011.

[2] H. Houweling, "Potentially Preventable Hospitalsations StatBite #10," Health Statistics Centre, Queensland Health, Brisbane, 2008.

[3] H. J. Jiang, C. A. Russo and M. L. Barrett, "Nationwide Frequency and Costs of Potentially Preventable Hospitalizations," U.S. Agency for Healthcare Research and Quality, Rockville, Md., 2006.

[4] Kaggle Inc., "Description - Heritage Health Prize," [Online]. Available: http://www.heritagehealthprize.com. [Accessed 15 May 2012].

[5] Kaggle Inc., "Heritage Health Prize Data Dictionary (Release 3)," April 2012. [Online]. Available: http://www.heritagehealthprize.com/c/hhp/download/Data_Dictionary_release3.pdf. [Accessed 21 May 2012].

[6] D. Bertsimas, M. Bjarnadóttir, M. Kane, J. C. Kryder, R. Pandey, S. Vempala and G. Wang, "Algorithmic Prediction of Health-Care Costs," *Operations Research,* vol. 56, no. 6, pp. 1382-1392, 2008.

[7] M. Fine, H. Pratt, S. Obrosky, J. Lave, L. McIntosh, D. Singer, C. Coley and W. Kapoor, "Relation between length of hospital stay and costs of care for patients with community-acquired pneumonia," *The American Journal of Medicine,* vol. 109, no. 5, pp. 378-385, 2000.

[8] R. Axelrod, P. Brierley and D. Vogel, "Heritage Provider Network Health Prize: Round 1 Milestone Prize," October 2011. [Online]. Available: http://www.heritagehealthprize.com/c/hhp/Leaderboard/milestone1/Market%20Makers%20-%20Milestone%201%20Description.pdf. [Accessed 20 April 2012].

[9] W. Maelstrom, "My milestone 1 solution to the Heritage Health Prize," November 2011. [Online]. Available: http://www.heritagehealthprize.com/c/hhp/Leaderboard/milestone1/Willem%20Mestrom%20-%20Milestone%201%20Description%20V2.pdf. [Accessed 22 May 2012].

[10] R. Axelrod, P. Brierley and D. Vogel, "Heritage Provider Network Health Prize: Round 2 Milestone Prize," April 2012. [Online]. Available: http://www.heritagehealthprize.com/c/hhp/Leaderboard/milestone2/Market%20Makers%20-%20Milestone%202%20Description.pdf. [Accessed 20 May 2012].

[11] T. G. Dietterich, R. H. Lathrop and T. Lozano-Perez, "Solving the Multiple-Instance Problem with Axis-Parallel Rectangles," *Artificial Intelligence Journal,* vol. 89, pp. 31-71, 1997.

[12] J. Foulds and E. Frank, "A Review of Multi-Instance Learning Assumptions," *Knowledge Engineering Review,* vol. 25, no. 1, pp. 1-25, 2010.

[13] S. Ray and M. Craven, "Supervised versus multiple instance learning: An empirical comparison," in *Proceedings of 22nd International Conference on Machine Learning*, 2005.

[14] S. Ray, "Multiple instance regression," in *In Proceedings of the 18th International Conference on Machine Learning*, 2001.

[15] J. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of Statistics ,* vol. 29, no. 5, pp. 1189-1232, 2001.

[16] F. Pan, T. Converse, D. Ahn, F. Salvetti and G. Donta, "Feature selection for ranking using boosted trees," in *Proceedings of the 18th ACM conference on Information and knowledge management*, 2009.

[17] S. Sosvilla-Rivero and P. Rodriguez, "Linkages in international stock markets: evidence from a classification procedure," *Applied Economics,* vol. 42, no. 16, pp. 2081-2089, 2010.

[18] L. Breiman, "Random Forests," *Machine Learning,* vol. 45, no. 1, pp. 5-32, 2001.

[19] A. Weigend, B. Huberman and D. Rumelhart, "Predicting the future: A connectionist approach," *International Journal of Neural Systems,* vol. 1, no. 03, pp. 193-209, 1990.

[20] K. Stanley and R. Miikkulainen, "Efficient Evolution Of Neural Network Topologies," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2002\.

[21] R. Jacobs, M. Jordan, S. Nowlan and G. Hinton, "Adaptive mixtures of local experts," *Neural Computation,* vol. 3, no. 1, pp. 79-87, 1991.

[22] L. Breiman, "Bagging Predictors," *Machine Learning,* vol. 24, pp. 123-140, 1996.

6 Appendix

6.1 Supervised Learners

Table 7: Summary of Gradient Boosting Machines

Learner	Trees	Shrinkage	Depth	Minimum Observations	RMSLE
GBM1	800	0.01	4	80	0.463664
GBM2	1200	0.01	4	80	0.463242
GBM3	1600	0.01	4	80	0.463093
GBM4	2000	0.01	4	80	0.462979
GBM5	2400	0.01	4	80	0.462992
GBM6	1600	0.01	4	60	0.463176
GBM7	1600	0.01	4	100	0.462897
GBM8	1600	0.01	4	120	0.463046

Table 8: Summary of Multi-layer Perceptrons

Learner	Hidden Units	Learning Rate	Iterations	Weight Decay	RMSLE
NN1	30	5000	5000	0.002	0.490194
NN2	25	5000	5000	0.002	0.483207
NN3	20	5000	5000	0.002	0.471826
NN4	15	5000	5000	0.002	0.480749
NN5	10	5000	5000	0.002	0.508691

Table 9: Summary of Random Forests

Learner	Number of Trees	Sample Size	RMSLE
RF1	100	10000	0.474951
RF2	500	10000	0.472226
RF3	1000	10000	0.472035

Table 10: Summary of other supervised learners

Learner	Notes	RMSLE
NEAT	Trained 10000 generations to reach 24 hidden units	0.471301
LR	Least square regression	0.475329

6.2 Linear Ensembles

Table 11: Ensemble weights from Linear Regression without MI GBM

Learner	Regression Weight	Prior Weight	Bayesian Average Weight	Regularised Weight
GBM1	0.0000	0.0556	0.0278	0.0263
GBM2	0.0000	0.0556	0.0278	0.0263
GBM3	0.0000	0.0556	0.0278	0.0263
GBM4	0.0000	0.0556	0.0278	0.0263
GBM5	0.0000	0.0556	0.0278	0.0263
GBM6	0.0000	0.0556	0.0278	0.0263
GBM7	0.0000	0.0556	0.0278	0.0263
GBM8	0.7450	0.0556	0.4003	0.3787
RF1	0.0000	0.0556	0.0278	0.0263
RF2	0.2290	0.0556	0.1423	0.1346
RF3	0.0000	0.0556	0.0278	0.0263
NN1	0.0000	0.0556	0.0278	0.0263
NN2	0.0000	0.0556	0.0278	0.0263
NN3	0.0000	0.0556	0.0278	0.0263
NN4	0.0000	0.0556	0.0278	0.0263
NN5	0.1400	0.0556	0.0978	0.0925
NEAT	0.0000	0.0556	0.0278	0.0263
LR	0.0000	0.0556	0.0278	0.0263

Table 12: Ensemble weights from Linear Regression with MI GBM

Learner	Regression Weight	Prior Weight	Bayesian Average Weight	Regularised Weight
MI GBM	0.0000	0.0526	0.0263	0.0249
GBM1	0.0000	0.0526	0.0263	0.0249
GBM2	0.0000	0.0526	0.0263	0.0249
GBM3	0.0000	0.0526	0.0263	0.0249
GBM4	0.0000	0.0526	0.0263	0.0249
GBM5	0.0000	0.0526	0.0263	0.0249
GBM6	0.0000	0.0526	0.0263	0.0249
GBM7	0.0000	0.0526	0.0263	0.0249

GBM8	0.7450	0.0526	0.3988	0.3773
RF1	0.0000	0.0526	0.0263	0.0249
RF2	0.2290	0.0526	0.1408	0.1332
RF3	0.0000	0.0526	0.0263	0.0249
NN1	0.0000	0.0526	0.0263	0.0249
NN2	0.0000	0.0526	0.0263	0.0249
NN3	0.0000	0.0526	0.0263	0.0249
NN4	0.0000	0.0526	0.0263	0.0249
NN5	0.1400	0.0526	0.0963	0.0911
NEAT	0.0000	0.0526	0.0263	0.0249
LR	0.0000	0.0526	0.0263	0.0249

Table 13: Ensemble weights from Bagged Linear Regression without MI GBM

Learner	Regression Weight	Prior Weight	Bayesian Average Weight	Regularised Weight
GBM1	0.0500	0.0556	0.0528	0.0540
GBM2	0.0120	0.0556	0.0338	0.0346
GBM3	0.0420	0.0556	0.0488	0.0499
GBM4	0.0310	0.0556	0.0433	0.0443
GBM5	0.0570	0.0556	0.0563	0.0576
GBM6	0.1160	0.0556	0.0858	0.0878
GBM7	0.0180	0.0556	0.0368	0.0376
GBM8	0.0100	0.0556	0.0328	0.0335
RF1	0.0730	0.0556	0.0643	0.0658
RF2	0.0940	0.0556	0.0748	0.0765
RF3	0.0570	0.0556	0.0563	0.0576
NN1	0.0320	0.0556	0.0438	0.0448
NN2	0.0550	0.0556	0.0553	0.0566
NN3	0.0420	0.0556	0.0488	0.0499
NN4	0.0660	0.0556	0.0608	0.0622
NN5	0.1300	0.0556	0.0928	0.0950
NEAT	0.0590	0.0556	0.0573	0.0586
LR	0.0100	0.0556	0.0328	0.0335

Table 14: Ensemble weights for Bagged Linear Regression with MI GBM

Learner	Regression Weight	Prior Weight	Bayesian Average Weight	Regularised Weight
MI GBM	0.0730	0.0526	0.0628	0.0599
GBM1	0.0490	0.0526	0.0508	0.0485
GBM2	0.0140	0.0526	0.0333	0.0318
GBM3	0.0380	0.0526	0.0453	0.0432
GBM4	0.0290	0.0526	0.0408	0.0389
GBM5	0.0580	0.0526	0.0553	0.0528
GBM6	0.1130	0.0526	0.0828	0.0790
GBM7	0.0360	0.0526	0.0443	0.0423
GBM8	0.0530	0.0526	0.0528	0.0504
RF1	0.0740	0.0526	0.0633	0.0604
RF2	0.0900	0.0526	0.0713	0.0680
RF3	0.0860	0.0526	0.0693	0.0661
NN1	0.0280	0.0526	0.0403	0.0385
NN2	0.0520	0.0526	0.0523	0.0499
NN3	0.0390	0.0526	0.0458	0.0437
NN4	0.0630	0.0526	0.0578	0.0552
NN5	0.1280	0.0526	0.0903	0.0862
NEAT	0.0620	0.0526	0.0573	0.0547
LR	0.0110	0.0526	0.0318	0.0304