厦門大學

本 科 毕 业 论 文

（主修专业）

机器学习在保险公司数据中的应用

A Machine Learning Application for Insurance Company Data

姓　　名：	陈轶伦
学　　号：	15220132201634
学　　院：	经济学院
专　　业：	统计学
年　　级：	2013 级
校内指导教师：	刘婧媛　副教授

二〇一七 年 四 月 九 日

Acknowledgement

The data in this article was supported by kaggle. I need to say thank you to Jingyuan Liu, Karl rohe who give me a lot useful suggestions. And Gillett John, Morvan Zhou for teaching me so many programming knowledge. I would also like to show my gratitude to Jane Wu, who always encourages me to learn what I want. Finally, I thank my parents for their selfless and love.

Abstract

In the information-explosion era, there is no doubt that big data influences our life dramatically. People in different areas want to make full use of their information to improve their goals. The life insurance application process is antiquated nowadays because of the time-consuming. In the past, customers provide extensive information to identify risk classification and eligibility including scheduling medical exams, a process that takes an average of 30 days. To make it quicker and less labor intensive for new and existing customers to get a quote, our target is to develop a predictive model that accurately classifies risk using a more automated approach by analyzing the multiple factors with a mixture of quantitative and categorical data. How many variables do we need to put on the model is a challenging problem. For this article, we only discuss three cases. First is to extract the quantitative data and analyze it directly. The second option is to use principal component analysis or locally linear embedding to reduce the data dimension and build a model based on this subset. The last one is to consider the whole database. Multiple imputation is adopted to replace the missing data for quantitative values. Machine learning methods such as decision tree and neural network are mainly used to predict the type of risk. Furthermore, the ensemble learning is adopted to improve our model accuracy and we introduce some basic idea about the categorical data analysis. By comparing the results from different models, we find that the model based on XGBoost has the best prediction rate. In addition, some important variables are found during the process.

Keywords: mixed data; decision tree; neural network; classification

摘要

在这个信息多元的时代中，大数据这个概念毫无疑问影响着我们生活的方方面面。不同领域的人们都尽可能的希望利用手头的数据分析出有用的价值为自己谋利。 目前，人生保险的申请评定过程由于需要耗费大量的时间而在大数据时代处于落后的现状。依据传统的评定过程，投保人需要提供详细的信息从而让保险公司能够为其评定相应的风险等级，这项复杂的过程往往需要花费平均一个月的时间。为了帮助保险公司更好更快的锁定新投保人的风险等级，本文力图通过构建一个有效的模型基于投保人提供的部分基本信息预测其风险。由于数据集的复杂性，模型的变量选择仍旧是一个开放式问题。在本文中，我们仅讨论三种情况。第一，我们仅使用数据库中的数值型数据进行分析；第二，我们使用不同的降维方法，提取出数据中最重要的部分，在此之上进行分析。第三，则是使用不会遭受维度灾难的模型对整个数据集直接进行分析。由于数据集中的部分属性数据含有缺失值，因此我们使用多重插补法对缺失数据进行插补。本文的预测模型主要基于决策树模型以及神经网络结构。此外，我们将集成学习纳入模型中进而提高模型的准确性，在风险等级的分类上，我们采用拓展的逻辑斯蒂模型——归一化指数模型进行区分。在使用归回分析进行模型拟合时，我们采用最小绝对值收敛法来获得较为简洁的模型。通过比较不同的模型效果，我们发现基于梯度递降法的树模型效果最佳，同时我们还找到了对风险等级起到重要作用的部分变量。

关键字：混合型数据集；决策树；神经网络；分类器

Contents

目录

1.Introduction

Insurance is the place where money collected from a group of people or organizations, to pay for the accidental losses that any of them may suffer. It contributes a lot to the growth of the society by provides stability to the unpredictable things such as nature disasters or terrorism. What is more, the insurance industries assist economy companies reduce the uncertainties through financial resources [1]. Among the hundreds of insurances, the life insurance plays an importance role. Life insurance promotes people's savings by paying the premium, that is to say, life insurance help you saving money and it will return you a decent amount of money when it expires. It can be regard as one sort of investment, which improves a habit of saving money due to the payment of premium. In addition, nobody has confidence that he or she would not get hurt by illness unexpectedly, in that case, taking an insurance will be helpful to release your financial burden.

It is wise to protect your loved ones financially by taking a life insurance, but it is indeed a major investment. Over a period of years, even a slightly lower premium can cost a lot. That's one of the reasons that some kinds of people are not willing to purchase insurance. Another reason is that the risk level for everyone is different. Someone might complain that he is healthier while he shares the same premium with others. In order to draw more people's interest, the insurance companies need to set the premium as low as possible meanwhile they do not need to worry about the risk. In general, the ultimate goals for insurance companies are to define the risk level for every customer based on their personal information.

By now, there are many theories arguing about what are the major factors that influence the customers' risk. Some factors like age, gender, smoking, lifestyle or family medical history are mentioned several times in the articles or newspapers. At the same time, with the change of our age and responsibilities, our needs for life insurance very a lot. Actually, we try to buy the insurance policy that depends on the standard of living we desire. Meanwhile, the insurance can help people reduce the risks simultaneously. For example, the cardiovascular disease, one of the leading killers in the US, differs a lot for men and women, and Jason E. Murasko [2] value the insurance status a lot as it can lower the risks for CVD.

In general, the premium that is applicable for life insurance depends on two concepts that is related to mortality and interest. While the latter factor will not be considered in this article since it is not our target. The mortality is our most interest and we will separate our applicants into different groups which represent diverse risk levels.

The rest of the paper is organized as follows. Next, we introduce multiple imputation (Section 2). After recalling some useful methods for dimensional reduction (Section 3), in Section 4 we present some methods of machine learning. Ensemble learning is proposed in Section 5. In addition, some categorical analysis in Section 6 The model result will be discussed in Section 7. Conclusions and extensive discussions are given in Section 8.

2. Multiple Imputation

Multiple imputation (MI) is so useful tool that either missing at random (MAR) assumption or missing not at random (MNAR) assumption would work [3]. MI was formally introduced by Rubin (1978), This methodology was firstly developed in the domain of sample surveys, but since then, its use has spread to other areas including observational and randomized trials, even in statistical matching [4]. In general, there are two kinds of approaches for imputing multivariate data: joint model (JM) and multivariate imputation by chained equations (MICE). Since MICE are more popular and have a lot of applications like addiction [5], cancer [6], and economy [7]. We will briefly introduce the MICE method in this section and will use it to deal with the missing value in our dataset.

Figure 1 illustrates the three main steps in multiple imputation: imputation, analyze and pooling. When facing complex missing data problem, MI will generate some complete datasets from the origin by Markov chain Monte Carlo (MCMC) techniques. Next step, the statistical methods can be adopted to the datasets, and we get the final result after pooling the analysis results under different datasets.

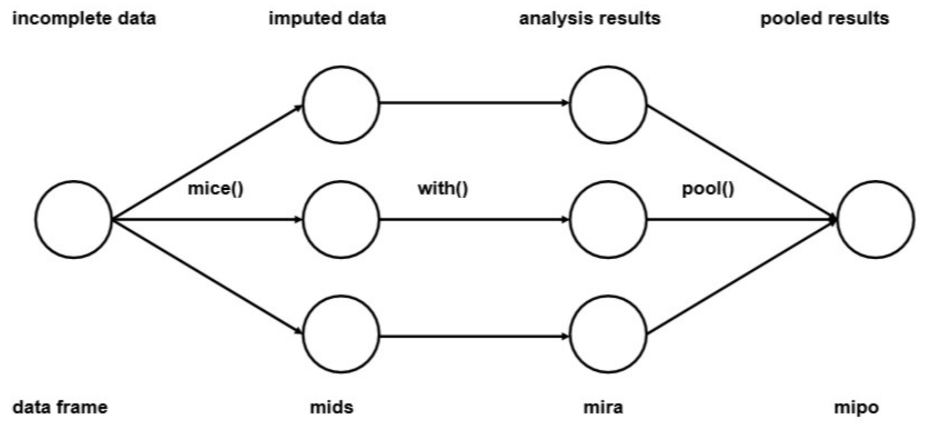


Figure 1 Main steps used in multiple imputation [8]

The chained equation process can be broken down into five general steps:

1. A simple imputation, such as imputing the mean, regression or other method is performed for every missing value in the dataset.

2. The imputations for one variable are removed to missing.
3. The observed values of the variable in Step 2 are regard as fixed and we use them to predict the missing value, which we deliberately remove in the last step. In other words, we treat missing values as dependent variable in the regression model while all the rest as independent variables in the regression model. When talking about the regression models, we mean the models that contain any methods that can help us to make a prediction such as linear, logistic, or Poisson regression models.
4. Then we replaced the missing values with predictions (imputations) from the regression model. If some missing values have been replaced, then for the next regression model for other missing values, the imputations would be used as the independent variables so the order of imputations might influent the results of the missing values.
5. We do a loop for Steps 2-4 for several times for each variable that has missing data. After each of the variables constructs one iteration or "cycle", we will do the iteration for some certain number based on research or experience. The method using regression to fill the data will make the dataset perform well for both MAR and MNAR.

Note that the regress in step 3 contains a lot of methods such as predictive mean matching, Bayesian linear regression, logistic regression, linear discriminant analysis and so on. Which regress to pick is still an open question. In our insurance model, we will use simple linear regression as our missing data are almost from quantitative data.

3. Dimensional Reduction

It is wonderful if we only have data with 2 or 3 dimension as we can see it by eyes. The dimensional reduction is one of the processes that help us reduce the number of random variables under consideration by considering a set of principal variables. In this section, we will recall the key idea about principal component analysis and locally linear embedding. Although it is better to use some supervised-based dimensional reduction methods, these methods usually require a distance matrix that measure the Euclidean distance between each sample. However, the distance matrix for 30,000 sample is so big to include in our software. We will only focus on the unsupervised case in this section.

3.1. PCA: Globality-Based

Principal component analysis (PCA) is a very classical method for dimensional reduction. PCA is very useful to project the high-dimension data onto a lower-dimension space that contains the most variance information of the original data [9]. Given the data set $X = [x_1, x_2, ..., x_n]$ in R^m, PCA try to find a projection axis $a \in R^{m*l}$, such that the mean square of the Euclidean distance between all pairs of the projected sample points $y = [y_1, y_2, \ldots, y_n]$ in R^l, i.e., $y_i = a^{\mathrm{T}}x_i$ (i = 1, …, n), is maximized as showed in the cost function:

$$J(a) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2 = \frac{1}{n}\sum_{i=1}^{n} a^{\mathrm{T}}(x_i - \bar{x})(x_i - \bar{x})^{\mathrm{T}}a = a^{\mathrm{T}}Ca$$

The basic elements of PCA are the eigenvectors of the data from the covariance matrix and its eigenvalues [10]. Denote (λ_1, e_1) ,..., (λ_p, e_p) to be the eigenvalue-eigenvector pairs for covariance matrix, where $\lambda_1 \geq ... \geq \lambda_p \geq 0$ and $e_1,..., e_p$ are standardized. Then the j th sample principal component from the i th sample is given by

$$y_{ij} = e_j^{\mathrm{T}}x_i$$

In every application, a decision must be made on how many principal components should be retained in order to effectively summarize the data. You can either retain sufficient number of principal components to account for a specified percentage of the total variance or retain the principal components whose eigenvalues are greater than the average of eigenvalues or use some plot to help. For the insurance example, we use the scree plot to detect the number of dimension. A scree plot displays the eigenvalues associated with a component or factor in descending order versus the number of the component or factor. We use scree plots in principal components analysis to visually assess which components or factors explain most of the variability in the data.

3.2. LLE: Locality-Based

Before we go to the locally linear embedding (LLE) method, it is necessary to review some basic idea of the manifold learning. Manifold learning, or nonlinear dimensionality reduction, is an alternative method to PCA which focusses on finding a low dimensional view for data sets which lie on nonlinear manifolds in a high-dimensional space. Figure 2 shows a classic example of manifold learning on an artificial dataset [11]. Figure 2(a) draw the so-called "Swiss Roll" data set which consists of 20,000 three dimensional points. Intuitively, this data set can be visualized as points in a rolled-up sheet of paper. While each of these points can be described by three coordinates, it is easier to understand these points underlying two-dimensional representation. The aim of manifold learning is to learn this representation automatically. The expected output of manifold learning algorithms is shown in Fig.2(b)

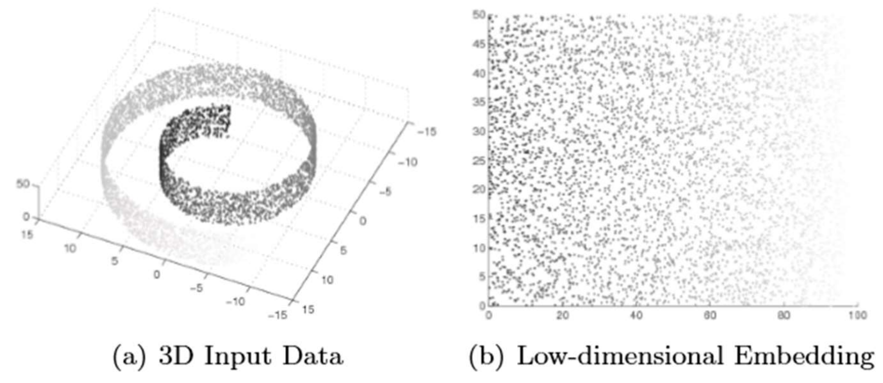


Figure2 Manifold learning example using "Swiss Roll" data set

Locally linear embedding, as one of the most popular manifold learning methods, aims to represent the manifold locally by reconstructing each point as a linear weight of its neighbors. As an input, the LLE algorithm takes n points $x_i \in R^D$, $i \in [1, n]$. As an output, it gives n points $y_i \in R^d$, $i \in [1, n]$, where d<<D. Above all, LLE tries to compute the low-dimensional embedding with the property that nearby points in a high-dimensional space remain nearby in a low-dimensional space [12]. The original LLE consists of three steps:

1. For each data point x_i, find the number of nearest neighbors based on the Euclidean distance used as a similarity measure. The optimal number of neighbors is calculated by using the algorithm proposed by Kayo [13].
2. Measure reconstruction error resulting from the approximation of each x_i by its nearest neighbors and compute reconstruction weights minimizing this error.
3. Compute the low-dimensional embedding best preserving the local geometry represented by the reconstruction weight.

In Step1 we use the Euclidean distances to determine a neighborhood around each x_i, while other definitions of "closeness" are also welcome as well. In Step2, the local geometry in the neighborhood of each data point is characterized by the linear coefficients that best reconstruct the data point from its neighbors. The reconstruction error is measured by cost function:

$$\varepsilon = \sum_{i=1}^{n} \left\| x_i - \sum_{j=1}^{n} w_{ij} x_{ij} \right\|^2$$

which satisfies two constraints: $\sum_{j=1}^{n} w_{ij} = 1$ and $w_{ij} = 0$ for the point which are not neighbors of x_i. In Step3, the low-dimensional embedding is found which best preserve the geometric properties of the original space through the minimization of the following embedding cost function:

$$\phi = \sum_{i=1}^{n} \left\| y_i - \sum_{j=1}^{n} w_{ij} y_j \right\|^2$$

subject to two constraints: $\sum_{i=1}^{n} y_i = 0$ and $\frac{1}{n}\sum_{i=1}^{n} y_i y_i^{\mathrm{T}} = I$, where I is the $d*d$ identity matrix. To calculate the matrix Y under these constraints, a new matrix M is constructed based on the matrix W: $M = (I\text{-}W)^{\mathrm{T}}(I\text{-}W)$. LLE then computes the bottom d+1 eigenvectors of M, which are the d+1 smallest eigenvalues. The first eigenvector whose eigenvalue is the closest to zero is excluded. The remaining d eigenvectors yield the final embedding Y.

As opposed to the globality-based data projection techniques like PCA, locality-based learning methods, like LLE, seek to find the nonlinear structure of the manifold existing in the given data. However, the nonlinear property makes it computationally expensive. Some enhanced models of LLE [14] can also help if computing power is good enough. For insurance model, we only use the basic LLE and choose a subset which consists of 10,000 sample since the Euclidean distance matrix needs too much space for calculating and saving. Therefore, the size of test dataset is reduced to 3,000. The number of the neighbor is calculated by computer software.

4. Machine Learning

Machine learning is the subfield of computer science and one famous people once says that the machine learning is "gives computers the ability to learn without being explicitly programmed" [15]. Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning attempts to understand the study and construction of algorithms that can learn from and make predictions on data [16]. This Section we will focus on two machine learning methods called decision tree and neural network. Our final insurance model will mainly depend on these methods.

4.1. Decision Tree

As you can guess from its name, the decision tree learning algorithm construct the model based on tree-shaped structure, which is similar to the flowchart. The decision tree itself include a lot of the logistical decision, with decision nodes that indicate a decision to be made on a feature. These splits into branches that show the decision’s choices. The tree is ended by leaf nodes that denote the result of following a combination of decisions [17]. Figure3 shows a classical example of decision tree applied in weather prediction:

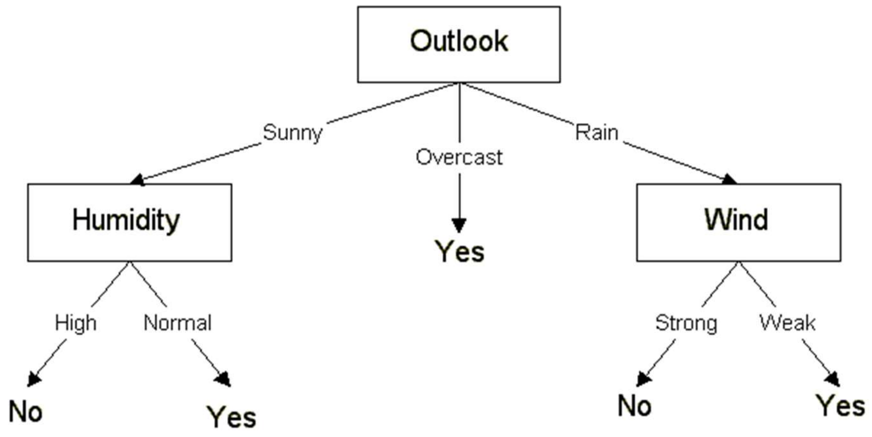


Figure3 a decision tree on predict the weather condition [18]

Now we are going into a popular decision tree algorithm called C5.0, which uses entropy for measuring purity. The entropy of a sample of data denotes the mixed level of the class values: the minimum value of 0 indicates that the sample is completely homogenous, while 1 indicates the maximum amount of disorder. The definition of entropy is specified by:

$$Entropy(S) = \sum_{i=1}^{c} -p_i \log_2(p_i)$$

In the entropy formula, for a given part of data (*S*), the term *c* implies the number of different class levels, and p_i implies the proportion of values falling into class level *i,* Given this measure of purity, the algorithm uses entropy to calculate the change in homogeneity due to a split on each possible feature. The calculation is known as information gain. The information gain for a feature *F* is calculated as the difference between the entropy in the segment before the split (S_1) and the partitions resulting from the split (S_2):

$$InfoGain(F)=Entropy(S_1)-Entropy(S_2)$$

the higher the information gain, the better a feature is at creating homogeneous groups after a split on that feature. For our insurance example, we will also use the boosting to improve the C5.0 algorithm, which consists of iteratively learning weak classifiers with respect to a distribution and adding them to a final strong classifier. That is to say, we construct several C5.0 trees and use all of them to classification.

4.2 Neural Network

The origin of Artificial Neural Networks (ANN) can be traced back to the work of Hebb (1949) [19], who proposed a learning law that became the ancestor of modern neural network training techniques. Artificial neural networks are biologically inspired devices used for mapping a set of inputs into a set of outputs. The basic properties of neural networks are the following:

1. Learning: the capability of the network to adapt its behavior to the environment, or in other words to autonomously build a representation of the map from inputs to outputs on the basis of a set of examples.
2. Generalization: the ability to react in a coherent way to imperfect inputs or to inputs not explicitly seen during learning.
3. Soft degradation: the alteration or elimination of some elements of the network does not prevent it from working but only induces a smooth degradation in the

performance.

Figure4 describes the structure of the artificial neuron. The transfer function either threshold or logistic or any other reasonable function.

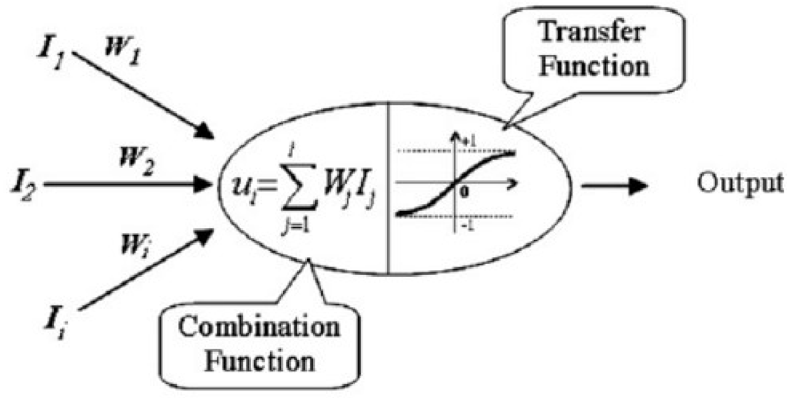


Figure4 The structure of the artificial neuron

The artificial neuron first receives some signals $I_1,\ldots,I_i$ as inputs, then computes the weighted sum of all the signals that are received as inputs to generate a global net input: $net_j = \sum I_i w_{ji}$, next, yields as output by means of an activation function. The standard method used in the ANN literature is the back-propagation algorithm (BP) [20]. Figure5 shows a simple case of a neural network.

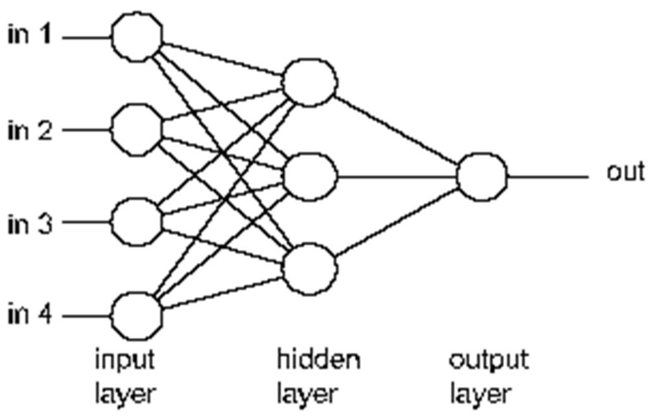


Figure5 a simple case of neural network [21]

We consider a problem with T patterns, and a network structure that includes K inputs, H hidden and J outputs. The phases that characterize BP learning are the following:

1. The vector W of weights starts with initial values drawn from a uniform distribution whose range depends on the user.
2. A pattern t is presented to the network. Given the randomly assigned weights, the pattern is propagated forward through the network.
3. A set of J errors (T_{tj}-O_{tj}) is computed from the comparison between the target and the output computed in Step2 for pattern t. The errors are used to calculate:

$$E_t(W) = \frac{1}{2}\sum_{t=1}^{T}(T_{tj} - O_{tj})^2$$

4. The general rule for modification of the weights is:

$$\Delta_t w_{ji} = -\alpha \frac{\partial E_t}{\partial w_{ji}}$$

   where α is the learning rate, a coefficient that regulates the speed of learning.

5. The derivatives can in general be expressed with:

$$\frac{\partial E_t}{\partial w_{ji}} = \frac{\partial E_t}{\partial O_{tj}}\frac{\partial O_{tj}}{\partial w_{ji}} = \frac{\partial E_t}{\partial O_{tj}}\frac{\partial O_{tj}}{\partial net_{tj}}\frac{\partial net_{tj}}{\partial w_{ji}}$$

6. The cycle is repeated from Step2 by considering a different pattern until all the patterns have been examined by the network.
7. The T squared errors are summed in order to obtain a global error over all the patterns.
8. Step2 to 7 are repeated until the global error reaches a specified value.

In general the coefficient that transforms the information in the first derivatives into a change in the estimated parameters is time-varying, however, the learning rate α used in BP is fixed. The learning rate may also be modified as learning takes place. In addition, the modification of the weights might be modified to:

$$\Delta_t w_{ji} = -\alpha \frac{\partial E_t}{\partial w_{ji}} + \beta \Delta_{t-1} w_{ji}$$

where β is the so-called momentum. The inclusion of the momentum is useful in avoiding excessive oscillations of the weights and can be justified as an approximation to a more general conjugate gradient [22]. In our insurance model, another ANN method called radial basis function (RBF) neural network is also used, the main difference between RBF and BP is that for RBF, it takes the Gaussian function as transfer function, and the output of the network is a linear combination of radial basis functions of the inputs and neuron parameters. RBF do well in the categorical classification.

Basically speaking, a RBGNN contains three layers: the input layer, the RBF layer (hidden layer) and the output layer. The inputs of hidden layer are the same as the BPNN, which is the linear combinations of scalar weights and the input vector. However, the main difference between BRGNN and BPNN is that in the hidden layer, the incoming vectors are mapping by the radial basis functions in each hidden node. The output layer yields a vector by linear combination of the outputs of the hidden nodes to produce the final output. The network output can be obtained by

$$y = f(x) = \sum_{i=1}^{k} w_i \phi_i(x)$$

Where $f(x)$ is the final output, $\phi_i(x)$ denotes the radial basis function of the hidden node, w_i denotes the hidden-to-output weight corresponding to the hidden node, and k is the total number of hidden nodes.

A radial basis function is a multidimensional function that describes the distance between a given input vector and a pre-defined center vector.

$$\phi_i(x) = \exp(-\frac{\|x - u_i\|^2}{2\sigma_i^2})$$

The advantages of RBF network is they bring much more robustness to your prediction, but as mentioned earlier they are more limited compared to commonly-used types of neural networks. However, commonly-used types of neural network models are highly vulnerable to adversarial noise and can make very wrong predictions. So it is a trade-off between higher accuracy in commonly-used types of neural networks or higher robustness in radial-basis function networks. In the insurance data example, we will use both of these methods to figure out which one is better.

5. Ensemble Learning

As an alternative to increasing the performance of a single model, it is possible to combine several models to form a powerful team. Just as the best sports teams have players with complementary rather than overlapping skillsets some of the best machine learning algorithms utilize teams of complementary models. Because a model brings a unique bias to a learning task, it may readily learn one subset of examples but have trouble with another. Therefore, by intelligently using the talents of several diverse team members, it is possible to create a strong team of multiple weak learners.

The ensemble learning methods are based on the idea that by combining multiple weaker learners, a stronger learner is created. For the rest of this section, we will introduce two powerful ensemble learning method: Extreme Gradient Boosting (XGBoost) and randomforest.

5.1 Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) [23] comprises of a tree based model, i.e. a collection of decision trees. Among the machine learning methods used in practice, gradient tree [24] is one technique that shines in many applications. The number of trees required for a particular problem can be passed as a parameter while building the model. Mathematically the model can be represented as:

$$\sum_{k=1}^{K} f_k$$

where the *f* is a decision tree or linear regression. The prediction for the *i* th data point can be written as:

$$\hat{y_i} = \sum_{k=1}^{K} f_k(x_i)$$

where x_i is the feature vector of the *i* th data point. The prediction at the *t* th step can be defined as:

$$\widehat{y_i}^t = \sum_{k=1}^{t} f_k(x_i)$$

For XGBoost, you can choose your cost function *L* depend on the specific question such as rooted mean squared error (RMSE) or logarithmic loss (LogLoss). While optimizing the objective is our primary goal, that is not the only thing that we should keep in mind while training a machine learning model. During training, it is often the case where the model learns the training data by heart and fails to predict the target of any new data point, so we need to "relax" the model a bit in each training step. In order to make the model get rid of the overfitting problem, we first define the model complexity:

$$\Omega = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

where *T* is the number of leaves and w_j is the score on the *j* th leaf. γ and λ are constants which can be passed as parameters while building the model. Then our goal become:

$$Goal = L + \Omega$$

Given an objective $Goal(y, \widehat{y_i}^t)$, for each iteration, the partial derivative $\partial_{\hat{y}} Goal(y, \hat{y})$ is calculated. Then $\hat{y}$ is improved along the direction of the gradient to minimize the objective. For the gradient descent in XGBoost, the objective function in the *t* th step is:

$$Goal^{(t)} = \sum_{i=1}^{N} L(y_i, \widehat{y_i}^{(t-1)} + f_t(x_i)) + \sum_{i=1}^{t} \Omega(f_i)$$

every step we only need to add one tree f_t into our model and another term can be regard as constant. Since derivatives of loss functions are difficult to be calculated, we use Taylor's approximation of the derivative:

$$Goal^{(t)} \approx \sum_{i=1}^{N} L[\widehat{y_i}^{(t-1)} + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \sum_{i=1}^{t} \Omega(f_i)$$

$$g_i = \partial_{\hat{y}^{(t-1)}} Goal(y, \hat{y}^{(t-1)}) \text{ and } h_i = \partial^2_{\hat{y}^{(t-1)}} Goal(y, \hat{y}^{(t-1)})$$

since $Goal(y_i, \hat{y}^{(t-1)})$ becomes a constant, our objective at the *t* th step boils down to:

$$Goal^{(t)} \approx \sum_{i=1}^{N} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t)$$

therefore, every step, we keep adding model *f* which comes from the minimization of our objective and use the whole models to predict.

5.2 Random Forest

Another ensemble-based method called random forests (or decision tree forests) focus only on ensembles of decision trees [25]. This method combines the base principles of bagging with random feature selection to add additional diversity to the decision tree models. After the ensemble of trees is generated, the model uses a vote to combine the trees' predictions [26].

Random forests combine versatility and power into a single machine learning approach. Because the ensemble uses only a small, random portion of the full feature set, random forests can handle extremely large datasets, where the so-called "curse of dimensionality" might cause other models to fail. At the same time, its error rates for most learning tasks are acceptable.

In fact, random forests provide an improvement on trees model by decorrelates the trees, which will reduce the variance when we average the trees. The process of random forests is simple. When building these decision trees, each time a split in a tree is considered. Next, a random selection of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. Unlike fitting a single large decision tree to the data, which amounts to fitting the data hard and potentially overfitting, the boosting approach instead learns slowly.

Each of these trees can be rather small, with just a few terminal nodes, determined by the parameter in the algorithm. By fitting small trees to the residuals, we slowly improve the regression function.

6. Regression Method

Generally speaking, if we can get the right feature of our model, then we only need to apply the linear regression to predict our goal. However, it is impossible to identify the true variables under hundreds of features. So, we will need to ask some general regression methods for help. In this chapter, we will introduce two special model called Softmax and lasso, which help us to handle the multi-class classification and shrink our parameters to a reasonable way. Notice that, the regression method will not only be used in the regression model, but also in a lot of situation. For example, we can use the Softmax method as our activation function in the neural network. We can use the lasso as our regression model during we build the regression trees.

6.1. Softmax

As our target is to classify the insurance application into a different class, our goal is a categorical data, which means the traditional method dealing with the quantitative data will not be suitable. For categorical data, one way to analyze is to use cumulative logistic regression for ordinal data or baseline logistic regression for nominal data [27]. In this section, we will introduce an expand model called Softmax as a more general method to classify the multiple categorical data. The Softmax function is used in various multiclass classification methods, such as multinomial logistic regression [28], multiclass linear discriminant analysis, naïve Bayes classifiers, and artificial neural networks.

Let us first come back to the logistic regression, our trainset consist *n* sample: $\{(x_1, y_1),\ldots, (x_n, y_n)\}$, the input variables $x_i \in R^{n+1}$, since the logistic regression focus on the binomial classification problem, hence the target $y \in \{0,1\}$. Th hypothesis function is :

$$h_\theta(x) = \frac{1}{1+\exp(-\theta^{\mathrm{T}}x)}$$

we will train our model to minimize the cost function:

$$J(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\{y_i \log[h_\theta(x_i)] + (1-y_i)\log[1-h_\theta(x_i)]\}$$

when it comes to the multiclass classification, a straightforward thinking is to extend the cost function into:

$$J(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} 1\{y_i = j\} log \frac{\exp(\theta_j^{\mathrm{T}} x_i)}{\sum_{l=1}^{k} \exp(\theta_l^{\mathrm{T}} x_i)}$$

in this case, the probability for x belong to the class j becomes:

$$p(y_i = j|x_i;\theta) = \frac{\exp(\theta_j^{\mathrm{T}} x_i)}{\sum_{l=1}^{k} \exp(\theta_l^{\mathrm{T}} x_i)}$$

how to solve the optimization for cost function is still an open question, if we use some iteration methods such as gradient or LBFGS, the formula will come to:

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{n}\sum_{i=1}^{n}[x_i(1\{y_i = j\} - p(y_i = j|x_i;\theta))]$$

For our insurance model, we will use Softmax in the both ANN and XGBoost method, also the traditional logistic regression will be used for comparison.

6.2. **Lasso method**

Least absolute shrinkage and selection operator (lasso) is a special linear regression method that select subset of the variables [29]. Lasso method has a lot of advantages such as it can reduce the dimension of the variables so that our model will not suffer from the curse of dimension. In addition, lasso method satisfies the practical assumption that there would be few of the variables that can explain the result. The lasso coefficients, $\hat{\beta}_\lambda^L$, minimize the quantity

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

In statistical parlance, the lasso uses an ℓ_1 penalty instead of an ℓ_2 penalty. One feature of lasso is that it can shrinks the coefficient estimates towards zero which gives us an efficient way to explain the model. Compared to the other regression method like ridge regression, the ℓ_1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large. Hence, much like best subset selection, the lasso performs variable selection. So, we can say that the lasso yields sparse models, that is, models that involve only a subset of the variables. The lasso regression has been used in many areas especially some place that need to deal with the sparse matrix [30].

7. Insurance Example

As the dimension of our dataset is so huge, we will explore our model mainly in three cases: quantitative only model, dimensional reduction model, and full data model. First of all, an introduction of our data is necessary.

The researchers at the insurance company collected data on 59,381 insurance applications. For each applicant, variables including their personal information, family background, and medical history are provided. The goal is to use these data to predict their real risk level, which is a ordinal variable containing 8 class. For the sake of their applicants' privacy, the insurance company only provide a simple description for the variable. Provided that we knew the complete information about the variable, then we could give weight for every variable for some reasons in the research. However, right now we can only treat every variable as the same value. The data contain 127 variables and 1 target. Within these 127 variables, 53 variables are discrete, 13 variables are continuous and 61 variables are categorical (nominal). Below are part of the variables:

Input:

Product_Info_1-7	A set of normalized variables relating to the product applied for
Ins Age	Normalized age of applicant
Ht	Normalized height of applicant
Wt	Normalized weight of applicant
BMI	Normalized BMI of applicant
Employment_Info_1-6	A set of normalized variables relating to the employment history of the applicant.
InsuredInfo_1-6	A set of normalized variables providing information about the applicant.
Insurance_History_1-9	A set of normalized variables relating to the insurance history of the applicant.
Family_Hist_1-5	A set of normalized variables relating to the family history of the applicant.

Medical_History_1-41	A set of normalized variables relating to the medical history of the applicant.
Medical_Keyword_1-48	A set of dummy variables relating to the presence of/absence of a medical keyword being associated with the application.

Output: The response, a target variable, an ordinal variable relating to the final decision associated with an application.

The following variables are all categorical (nominal):

Product_Info_1, Product_Info_2, Product_Info_3, Product_Info_5, Product_Info_6, Product_Info_7, Employment_Info_2, Employment_Info_3, Employment_Info_5, InsuredInfo_1, InsuredInfo_2, InsuredInfo_3, InsuredInfo_4, InsuredInfo_5, InsuredInfo_6, InsuredInfo_7, Insurance_History_1, Insurance_History_2, Insurance_History_3, Insurance_History_4, Insurance_History_7, Insurance_History_8, Insurance_History_9, Family_Hist_1, Medical_History_2, Medical_History_3, Medical_History_4, Medical_History_5, Medical_History_6, Medical_History_7, Medical_History_8, Medical_History_9, Medical_History_11, Medical_History_12, Medical_History_13, Medical_History_14, Medical_History_16, Medical_History_17, Medical_History_18, Medical_History_19, Medical_History_20, Medical_History_21, Medical_History_22, Medical_History_23, Medical_History_25, Medical_History_26, Medical_History_27, Medical_History_28, Medical_History_29, Medical_History_30, Medical_History_31, Medical_History_33, Medical_History_34, Medical_History_35, Medical_History_36, Medical_History_37, Medical_History_38, Medical_History_39, Medical_History_40, Medical_History_41

The following variables are continuous:

Ins_Age, Ht, Wt, BMI, Employment_Info_1, Employment_Info_4, Employment_Info_6, Family_Hist_2, Family_Hist_3, Family_Hist_4, Family_Hist_5

The following variables are discrete:

Medical_History_1, Medical_History_10, Medical_History_15, Medical_History_24, Medical_History_32， Medical_Keyword_1-48 are dummy variables.

The distribution of the response is extremely unbalanced, meanwhile, 8 type of category is too much for classification, so in this paper, we only analyze part of the response and delete some output with less frequency.

Table1 shows the number of every response.

Table1 The number of every response

Response	1	2	3	4	5	6	7	8
Number	6,207	6,552	1,013	1,428	5,432	11,233	8,027	19,489

In order to analyze a more balanced dataset, we merge the response into 3 class, class 1 to 5 are treat as one group, and class 6 and 7 are treat as a group. Then the response table become

Table2 The number of new responses

New	1	2	3
Number	20,632	19,260	19,489

We will build our model based on two types: tree based model and neural network based model. But before we analyze the data, there is a slight difference between these models. It is suitable to input the category type of variables into the tree model but throw them into the NN model might cause the problem, since NN will treat them as a number instead of category. So for NN case, a transformation is necessary and the all of the categorical variables are transferred into dummy variables, which means for each new variable, they can only take value 1 if the sample belongs to some certain categories and take value 0 otherwise. When we simulate the data with lasso method, we use cross validation to determine the lasso parameter.

For the limitation of the capability on computer, we would not use all of the data. The first 30,000 samples are chosen as the training set and the last 10,000 samples are chosen as predict set. For Neural Network model, we take some of the samples as validation set in case of the overfitting. Also, the multiple imputation is adopted to get rid of the missing data problem. The missing data only happen in the continuous variables, and the missing rate is acceptable so we do not need to discard some of them. For the dimensional reduction case, we use scree plot to decide how many principal variables we want to keep in the PCA and let the computer to calculate the suitable embedding dimension for LLE. Figure6 shows the scree plot for PCA:

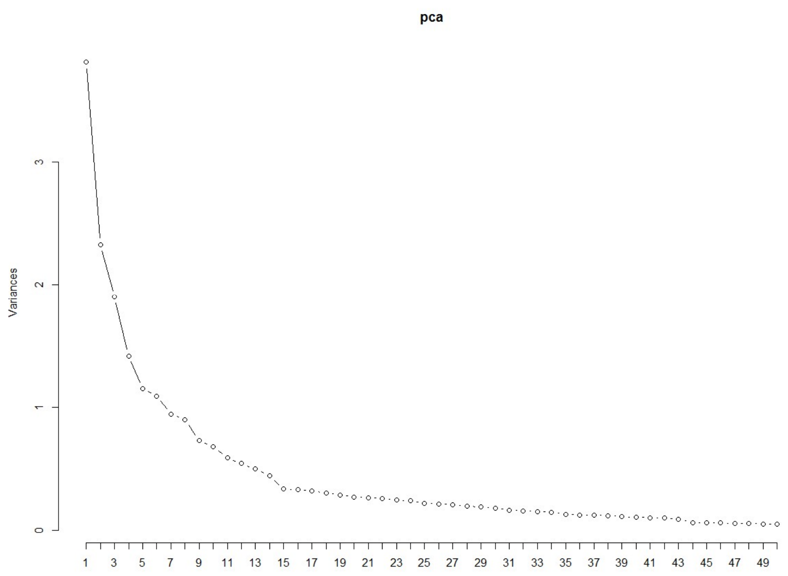


Figure6 scree plot for PCA

Table3 show the prediction rate for the 4 kind of dataset and 6 kind of classifiers:

Table3 result from different methods

Prediction rate	*Quantitative*	*PCA*(D=10)	*LLE*(D=14)	*Complete*
C5.0	61.6%	61.9%	46.7%	70.3%
XGBoost	62.5%	62.7%	50.2%	72.6%
Randomforest	62.2%	62.9%	52.0%	71.3%
Logistic-BP	62.1%	64.3%	50.1%	69.8%
Softmax-BP	61.5%	62.5%	49.3%	71.6%
RBF	63.7%	65.6%	46.8%	69.7%
Lasso	60.1%	58.2%	51.5%	66.6%

From the above table, it is surprising that the dataset using LLE method perform even worse than the dataset that only uses the quantitative variables, while the dataset PCA do improve the prediction. The reason that these two-dimensional reduction methods differ so much might be that the LLE is a nonlinear method, which needs plenty of time to find a proper kernel function and might not to be effective enough in a massive sparse matrix. We can also see that using lasso regression does not give us better result since the true model might have highly nonlinearity. As the models using complete dataset are best among 4 kinds of dataset, next we will look at the details in every classifier using complete dataset.

In the C5.0 algorithm, we use boosting method to generate 1,000 decision trees and combine them to make stronger one, table4 below is the top 15 variables that chosen from the decision trees, which indicates the importance of our variable.

Table4: top 15 variables for C5.0

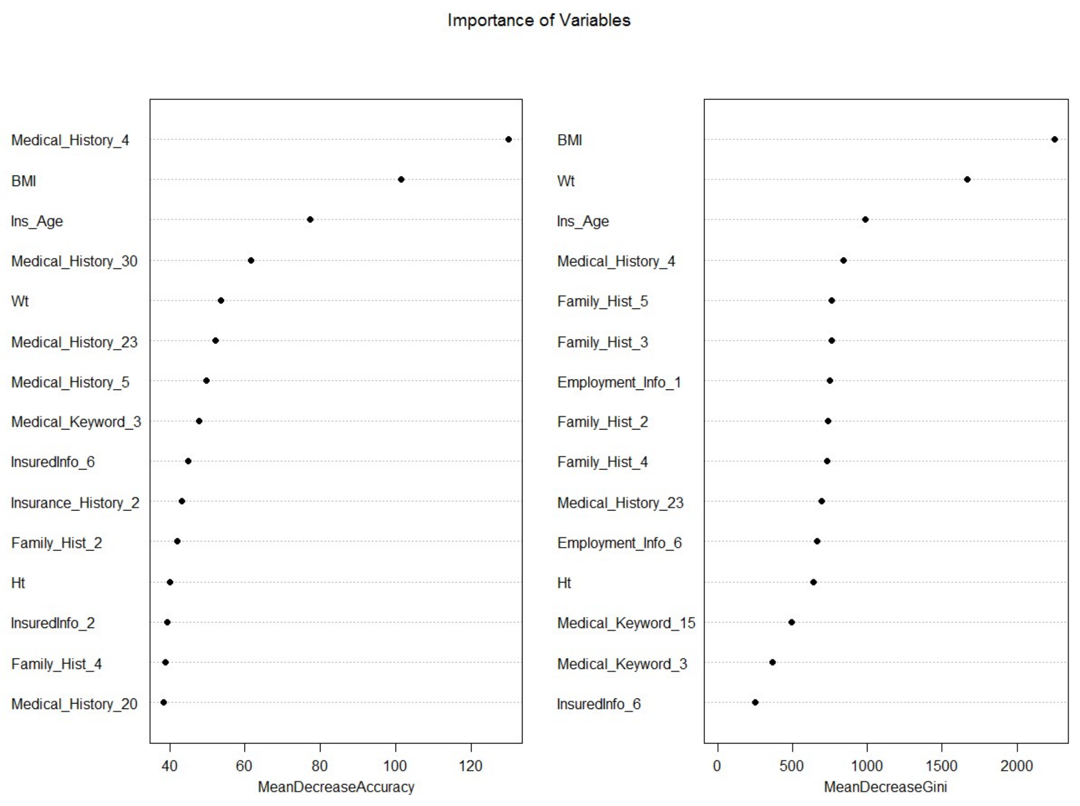
Frequency in the 1,000 trees	Variable Name
100.00%	BMI
100.00%	InsuredInfo_6
100.00%	Medical_History_11
100.00%	Medical_History_35
100.00%	Medical_Keyword_3
100.00%	Medical_Keyword_15
99.87%	Medical_Keyword_38
99.67%	Ins_Age
99.08%	Medical_History_18
99.02%	Medical_History_23
98.97%	Medical_History_40
98.95%	Wt
98.47%	Family_Hist_5
98.40%	Employment_Info_1

In the random forest, there are 500 trees in the background. We want to measure the importance of one of the covariates X_1. Choose and fix one of the trees T in the forests. Note that there is a bootstrap sample behind it. Calculate the out-of-bootstrap (OOB) error estimate, P_T(OOB). Take a random permutation of X_1 in the bootstrap sample. Construct a tree and calculate the corresponding OOB error estimate, $P_T(\mathrm{OOB}_1)$. A raw score for X_1 is calculated by

$$Raw_T(X_1) = P_T(\mathrm{OOB}_1) - P_T(\mathrm{OOB})$$

the philosophy behind what we are doing is that if X_1 is important permuting the X_1 value in the bootstrap sample will result from a high value for the OOB error estimate. The OOB error estimate would not be the same based on a different scale of the variable. Table5 shows the importance of variables calculated by accuracy number and Gini coefficient.

Table5 Importance of variables in randomforest



In the XGBoost model, the gain gives you the indication about the information of how a feature is important in making a branch of a decision tree more pure. However, with this information only, you can’t know if this feature has to be present or not to get as specific classification.

Compared these three plots, we can find that some variables play an essential role in classification, such as BMI, insurance age, weight, family history 5, employment information 1, insurance information 6, medical history 23, medical keyword 3. If we know the specified content for these variables, maybe we can understand the secret under these variables. By now we just street these items.

In the XGBoost model, the gain gives you the indication about the information of how a feature is important in making a branch of a decision tree more pure. However, with this information only, you can’t know if this feature has to be present or not to get as specific classification. Table6 give the top 15 variables that get the highest gain

Table6 Important variables in XGBoost

Feature	Gain
BMI	0.210
Medical_History_1	0.168
Ins_Age	0.063
Family_Hist_3	0.051
Wt	0.050
Family_Hist_5	0.049
Employment_Info_1	0.048
Employment_Info_6	0.040
Family_Hist_2	0.038
Family_Hist_4	0.036
Medical_History_2	0.024
Medical_Keyword_3	0.024
Ins_info_1	0.022
Ht	0.021
Medical_Keyword_15	0.020

Although we cannot make a judgment by considering the value of gain as its importance, we still could find that these variables are very similar to the table4 and table5 such as BMI, insurance age, weight, family history 5, employment information 1, and medical keyword 3. Hence at least we know they are importance at some point.

Next, we will look at more details for our models based on the top 3 methods using complete dataset. As our target is to classify the sample into three group, we will use some cross matrix to illustrate how well in each class in table7.

Table7 Prediction information for top three models

Softmax-BP

	Pred 1	Pred 2	Pred 3	Total	Accuracy
True 1	1,828	526	571	2,925	62.5%
True 2	620	1,304	674	2,598	50.2%
True 3	172	273	4,032	4,477	90.1%
Total	2,620	2,103	5,277	10,000	71.6%

XGBoost

	Pred 1	Pred 2	Pred 3	Total	Accuracy
True 1	1,768	639	518	2,925	60.4%
True 2	554	1,552	492	2,598	59.7%
True 3	161	375	3,941	4,477	88.0%
Total	2,483	2,566	4,951	10,000	72.6%

RandomForest

	Pred 1	Pred 2	Pred 3	Total	Accuracy
True 1	1,821	532	572	2,925	62.3%
True 2	654	1,330	614	2,598	51.2%
True 3	143	355	3,979	4,477	88.9%
Total	2,618	2,217	5.165	10,000	71.3%

From these three models, there is no doubt that class three is separate very well while the other two class are fuzzy. For class two, the accuracy to classify the class two type sample into the right place is only above a half. It is difficult to explain why their accuracy differ so much. Maybe it is because the features for the first two categories are not significant enough, or maybe there are some problems within our definition of class.

8.Conclusion&Discussion

In this paper, several classic machine learning methods have been used, which are based on decision tree and neural network algorithm. Our model gets benefit from the ensemble learning such as XGBoost and randomforest. Combining these kind of methods, we can extract the feature information from the observation data. In comparison with the different dataset, using the whole data is the best choice instead of reducing the data dimension by PCA or LLE. During the learning process, some variables are found as importance features to classify the sample such as BMI, Insurance information 6, and Medical history 1. After trying 7 methods to make the classification, our prediction rate is more than 70%. However, in the prediction part, the accuracy for different categories are not the same. Class three present very well while other two class require more research to improve. Compared the linear model such as lasso regression, the nonlinear models give us higher prediction rate, which means that basic linear model might not be useful and it is hard for us to interpret the logic behind every variable. In addition, the BP neural network is much better than RBF network in our example. Since our sample size is pretty large so we don’t need to put much attention on the outliers and the models we use are stable.

Our final model is based on the XGBoost algorithm and we use the Softmax for activation function. In fact, we cannot say tree model is much better than neural network or the neural network model is not suitable for this example. There is not significant evidence that the prediction rate shown in the result have huge change between these two types of model. However, for the sake of the time, the neural network model requires a huge time to train the data and we should try hundreds of different parameters to adjust the model in order to find the best one. The XGBoost method has time efficiency and it also uses the ensemble learning technique to improve the prediction rate. For the activation function, we choose the Softmax method since it highly fit our goal to classify our data into several groups.

In the data cleaning part, we don’t consider the outliers because we have a huge size of the database. And we think some outliers might have their reason to exist so we don’t delete them. Cross-validation might be useful to improve our model since it can be used to deal with the missing data when applying the multiply imputation and help the neural network to compare the different parameters during fitting the model.

Furthermore, more detail information about the variables would be helpful to make variables selection. Some interesting issues would be to develop the prediction model with adjustment for some parameters in the learning process, and supervised learning-based dimensional reduction without distance matrix. Also, another dimensional reduction methods like isomap, local fisher embedding, and autoencoder might lead to different accuracy result for our model.

9. References

[1] Anonymous, The Importance of Insurance Real Estate Finance; Dec 2005; 22, 4; Business Premium Collection

[2] Jason E. Murasko: Gender differences in the management of risk factors for cardiovascular disease: The importance of insurance status Social Science &Medicine (63) 2006 1745-1756

[3] O'Kelly, Michael, Ratitch, Bohdana Statistics in Practice : Clinical Trials with Missing Data : A Guide for Practitioners Wiley February 2014 page 185-251

[4] Alpman, Anil Implementing Rubin's Alternative Multiple Imputation Method for Statistical Matching in Stata IDEAS Working Paper Series from RePEc 2015

[5] Schnoll RA, Rukstalis M, Wileyto EP, Shields AE (2006). "Smoking Cessation Treatment by Primary Care Physicians. an Update and Call for Training." American Journal of Preventive Medicine, 31(3), 233–239.

[6] Clark TG, Stewart ME, Altman DG, Gabra H, Smyth JF (2001). "A Prognostic Model for Ovarian Cancer." British Journal of Cancer, 85(7), 944–952.

[7] Briggs A, Clark T, Wolstenholme J, Clarke P (2003). "Missing...Presumed at Random: Cost-Analysis of Incomplete Data." Health Economics, 12(5), 377–392.

[8] VanBuuren,S., Groothuis-Oudshoorn,K.(2011).mice: Multivariate Imputation by Chained Equations in R. Journal of Statistical Software,45(3), 1-67

[9] S. Wold, K. Esbensen, P. Geladi, Principal component analysis, Chemometrics and Intelligent Laboratory Systems 2 (1-3) (1987) 37-52

[10] Jolliffe, I.T. (2002). Principal Component Analysis, second edition (Springer)

[11] Robert Pless, Richard Souvenir, A Survey of Manifold Learning for Images, IPSJ Transactions on Computer Vision and Application Vol. 1 83-94 (2009)

[12] Roweis, S.T., Saul, L.K., 2000. Nonlinear dimensionality reduction by locally linear embedding. Science 290 (5500), 2323-2326

[13] Locallylinearembeddingalgorithm-extensionsandapplications/OlgaKayo/UniversitatisOuluensis, Oulu, Finland / 2006

[14] Shi-qing Zhang 2009. Enhanced supervised locally linear embedding. Pattern Recognition Letters 30

(2009) 1208-1218

[15] Phil Simon (2013). Too Big to Ignore: The Business Case for Big Data. Wiley. P.89

[16] Ron Kohavi, Foster Provost (1998). Glossary of terms. Machine learning 30: 271-274

[17] Quinlan, J. R. (1987). "Simplifying decision trees". International Journal of Man-Machine Studies. **27** (3): 221.

[18] Andrew Colin, Dr. Dobbs, Building Decision Trees with the ID3 Algorithm, Journal, June 1996

[19] Hebb, D.O. (1949) Organization of Behaviour, Science Editions, New York, NY.

[20] Rumelhart, D.E. and McClelland, J.L. (1986) Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations, MIT Press, Cambridge, MA.

[21] White, H. (1991) Learning in Artificial Neural Networks: A Statistical Perspective. Neural Computation, 1, 425-64

[22] Battiti, R. (1992) First and Second order Methods for Learning: Between Steepest Descent and Newton's Method. Neural Computation, 4, 141-66

[23] Tianqi Chen, Carlos Guestrin. (2016) XGBoost: A Scalable Tree Boosting System. Learning (cs.LG)

[24] J. Friedman. Greedy function approximation: a gradient boosting machine. Annals of Statistics, 29(5):1189–1232, 2001.

[25] Ho, Tin Kam (1995). Random Decision Forests. Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282.

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

[27] Alan Agresti, Categorical Data Analysis, 2002 John Wiley & Sons, Inc., Hoboken, New Jersey.

[28] Bishop, Christopher M. (2006). Pattern Recognition and Machine Learning. Springer.

[29] Tibshirani, Robert. 1996. "Regression Shrinkage and Selection via the lasso". Journal of the Royal Statistical Society. Series B (methodological) 58 (1). Wiley: 267–88.

[30] She, Yiyuan. "Sparse regression with exact clustering" *Electronic Journal of Statistics*.