Predicting Length of Future Hospitalization Period: A Comparative Study

Anonymous Author(s)
Affiliation
Address
email

Abstract

Our goal in this report is to predict number of days that a patient is going to stay in hospital in future using the claims that the subject has submitted in past. We attack the problem using three different techniques, and we compare their performance on the large scale Heritage Health Prize (HHP) dataset. We here focus on Linear Regression, Neural Network, and Random Forrest for regression. We also examine a kmeans based feature extractor [1] on the dataset, and we show how this feature extractor can improve the performance of simple techniques like linear regression. Our results on HHP dataset are promissing, and we achieve comparable results to the current leading teams in the competition.

1 Introduction

Over last decades, with advances in computational resources, machine learning has been deployed in many research areas. One of these areas is health care systems, where the goal is to use the computational power to help experts in diagnosis or treatments. Machine learning can even enable us to extract patterns that are not visible or trivial to human.

Heritage Health Prize (HHP) is a competition which is held by Heritage Provider Network in order to use machine learning to reduce length of future hospitalization of a patient. The goal in this competition is to predict the number of days that a patients will spend in a hospital based on their past medical history. For developing such an algorithm a large dataset of medical information for about 100,000 patients over two years has been provided, along with number of days that they spent in hospital for two the successive years. Then the task in competition is to predict the number of hospitalization days for the third year.

There has been tremendous research on regression problem which makes reviewing all those proposed techniques impossible in this report. For a brief overview if techniques one can check [2], [3]. Here we review techniques proposed for the HHP by the best two teams of the first milestone of competition which is available publicly. Brierley et al. [4] won the first milestone by ensembling four different techniques. They use Random Forrest [5], Neural Networks, Linear model, and Gradient Boosting trees [6]. Their final model is an ensemble all those models. Mestrom [7], the second winner of the milestone developed 21 different models and formed a final model by linear combination all those model. Typically their models is a linear model fed to a nonlinear function such as sigmoid or log. Different models differ based on the features that have been used, the basis function applied to linear combination, or the subset of data used for training.

Similar experience on Netflix prize shows that the final solution to the problem may not be a single method, but an ensemble of different techniques. For this purpose in this project we consider a few simple techniques proposed for regression. We compare the performance of these methods on this dataset, and we form our final predictor by combining all simple methods. Here we use linear least square model, Neural Networks, and Bagged Trees. The first three techniques has been shown to be effective by the winners of the first milestone. In addition to regression, we examined a simple feature learning technique and we show how a better feature can improve the performance of a simple technique such a linear regression model.

The organization of paper is as following. In Sec. 2 we define the problem, and the data provided by the competition. In Sec. 3 we review the models we used for the problem, and in Sec. 4 we examine

Table 1: Features representing a claim. We extract following fields for a claim, and we represent each one of these fields with a vector of zeros with a one-entry at the corresponding index.

Fields	Length
Specialty	13
Place of Service	9
Primary Condition	45
Procedure Group	18
Charles Index	4
Length of Stay	12
Total	101

them on the data and we show effect different parameters on the performance of our system. Finally, we discuss the drawbacks of our system, and its possible solutions in Sec. 5

2 Heritage Health Competition

In HHP we are provided with 2.6 million claims submitted by about 113000 members over 3 years in addition to the number of days they spent on the successive years. For each claim, general information is stored, including: the specialty of the case (such as surgery, laboratory, internal, and etc.), primary condition (a broad grouping of diagnostic categories such as infection, cancer, appendicitis, and etc.), procedure group (such as radiology, pathology, medicine, evaluation and etc), Charles Index (A measure of affect of disease between 0-4), and the number of hospitalization days. In addition to the claims, we are also given with number of drugs and lab test has been used by a patient over a year, as well as the members sex gender and age. Claims are available for 3 years, and the number of days that the member has spent in hospital in the second and third year is provided. The problem is to predict the number of days that members are going to hospitalize in the fourth year given the claims they submitted over first three years.

The competitions final evaluation is based on Eq. 1

$$E = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(log(y_i+1) - log(t_i+1))^2}, \tag{1}$$

where y_i and t_i are the prediction and ground truth for the ith subject, and N is the total number of subjects.

The evaluation criteria in Eq. 1 can be considered as the squared log ratio of prediction and actual target value. First insight into the problem reveals that fitting a model to the training target values by minimizing the least square error between prediction and ground truth will not result in a good performance on the test data. A better model can be achieved by optimizing the same criteria as in Eq. 1 on the training data. However, due the log function in the evaluation measure the optimization in many learning techniques such as linear model will result in a non-convex optimization problem. However, instead of fitting a model to the actual target values one can fit the model to the logarithm of target values. This technique has two advantages: First the actual evaluation criterion is minimized over training which may result in a better model. Second, the fitting problem will not become non-convex because of the log function.

2.1 Features

In order to be able to compare different regression techniques on HHP dataset, we need to fix the set of features used for those techniques. In this section we briefly describe the features we extracted from claims, and in the next section we will describe the methods we implemented.

A patient may have different number of claims over a year. So, the claims should be represented in a way that it can be easily integrated to a representation for all claims over a year. Fortunately, most of the fields of claims are discrete, and one of K representation can be used to represent those field. For example, length of stay for a claim is discretized to 1 day, 2 day, ..., 6 days, [1-2) week, [2-4) weeks, [4-8) weeks, [8-12) weeks, [12-26) and +26 weeks. So, the length of stay field can be represented with a binary features of the length 12 with only one non-zero component representing the case. In Table.1 the features that we used to represent a claim and their length has been shown.

we represent a claim by a feature of the length 101. All claims of a patient are then summed up together to represent the claims submitted by patient over a year. The age of a patient is also discritized in bins of 10-year length, and can be represented in the same 1 of K representation using

$K = 10$. Similarly we represent the gender of patient using a vector of $K = 3$ for Male, Female, and Not Available. We also extract the total number of drugs and lab tests issued for the patient over a year. We form our final discriptor by concatenating claim features (101), age(10), sex(3), drug count(1) and lab count(1) which results in a sparse feature of length 116.

3 Methods

In this section we review the models we used to predict the number of days that a patient is going to spend in hospital. All the techniques has been used for regression problem before. Our goal in this report is to compare their effectiveness for HHP competition. We mainly focus on different models for regression, but we also examined a feature learning technique to show how a different feature could improve the accuracy of a regression model. In following section we start with simple Linear Model, and we continue with Neural Netwok for regression. Later we continue with a non-parametric, tree based method for regression problem. Finally, we conclude the section with a simple but effective feature learning algorithm.

We assume we are given a set of training data in the form $\{x_i, y_i\}$ where $x_i \in \mathcal{R}^D$ is the feature vector describing all claims submitted by a patient, y_i is the number of days that the patient has spent on the following year, and D is the length of feature vector. The goal of training algorithm is to learn a function such as $\mathcal{F} : \mathcal{X} \rightarrow \mathcal{R}$ that can be used to predict length of hospitalization period for a new data. Note that as we discussed in Sec. 2 we will replace y_i with $log(y_i)$ so that minimizing the Mean Square Error results in minimizing the evaluation criterion in Eq. 1. For simplicity after this we represent $log(y_i)$ using y_i, and therefore instead of calculating error expression in Eq. 1 we will use simple MSE to evaluate different techniques.

3.1 Linear Regression

The simplest approach to train a model for our problem is the linear regression. In this approach we assume the function $\mathcal{Y}$ is in the form of:

$$y(x, w) = w^T x \tag{2}$$

The minimization of MSE on the training data has analytical solution. If we assume X is the design matrix where the ith row has the sample x_i^T then:

$$w^* = (X^T X)^{-1} X^T y \tag{3}$$

However, in order to control overfitting typically a regularization term $E(w)$ is added to MSE objective function as well. Here, we use a L2-norm regularizer known as ridge regression which can be solved by:

$$w^* = (X^T X + \lambda \mathcal{I}_D)^{-1} X^T y \tag{4}$$

where $\mathcal{I}_D$ is identity matrix of size D and λ is the regularization parameter that will be chosen by cross-validation.

3.2 Neural Network

There is an assumption behind linear regression model that may not hold for our problem. In this model, it is assumed that the target value can be modeled as linear combination of different measurements (features here). One may add some non-linearity to the model by designing nonlinear basis functions, however, the problem will be designing good basis functions. Nueral Network are general class of non-linear parametric functions that maps input x to a target value y. The advantage of neural networks is their flexibility to tune different parameters of a non-linear model. However, adding this option makes parameter learning a non-convex problem. Therefore instead of a global optimum, a local optimum is achieved by iterative techniques such a gradient decent.

We use a two-layer network to model the mapping function where there is a hidden layer between input and output nodes. The hidden layer can be considered as feature functions applied to the data. The output of hidden layer is passed to a linear model to form the final output of function. Typically in regression problem, an identity function is used in output node however, the transfer function in hidden layers can be chosen among a few options. Besides these functions, number of hidden layers should be also set. In Sec. 4 we will discuss the efficiency of different parameter settings.

Parameter learning in a feed forward Nueral network can be done by many methods in gradient decent family such as steepest decent and stochastic gradient decent. In this method, parameters are set randomly in the first iteration, and in each iteration they are updated in direction of negative gradient of error function:

$$w^{t+1} = w^t - \eta \nabla E(w^t) \tag{5}$$

where w^t is the current parameters of the network, $E(w^t)$ is the error function (MSE in our case) and η is the step size for update.

The gradient in Eq.5 can be calculated by Error Backpropagation technique. In this technique, the data is fed forward in the network and outputs of neurons are calculated. Then gradient of parameters are calculated by traversing the network backword. We use MATLAB neural network toolbox for learning network parameters. This software uses Levenberg–Marquardt [8] algorithm for optimizing the parameters which can be considered as an extension of gradient decent technique.

3.3 Random Forest

Tree-based techniques are simple but effective non-parametric techniques that have been widely used for regression and classification. They can be viewed as a combinations of many models that each makes a decision at a region. In a very simple form a CART, classification and regression tree [9], is a binary tree whose model is selected by traversing in tree from root to a leaf. In each node a feature of input is compared with a threshhold and the successive node is chosen based on the output of comparison. Therefore, each node will correspond to a region in input space which has a constant value for output variable.

Learning a CART, corresponds to constructing a tree that has the lowest MSE on the training data. Constructing a tree is consist of choosing splitting variable for each node, and assigning an output value to the regions at the leaves. Given the regions, it can be shown that the output of the tree for the region at a leaf can be calculated by:

$$\hat{y}_m = avg(y_i | x_i \in R_m) \tag{6}$$

where R_m is a region segmented by the mth leaf and $\hat{y}_m$ is the output of tree for the region.

Finding the regions that minimized the MSE on training data globally is computationally infeasible. Therefore, typically a greedy algorithm is used for finding the splitting variable and the threshold for each node. In this technique, for a node on the tree, all splitting variables are examined and the one that minimize the MSE for two resulting region is chosen. i.e. threshhold θ_j can divide the variable j into two regions $R_1(j, \theta_j)$ and $R_2(j, \theta_j)$ where:

$$R_1(j, \theta_j) = \{x | x^{(j)} \leq \theta_j\}, \qquad R_2(j, \theta_j) = \{x | x^{(j)} > \theta_j\} \tag{7}$$

So, we need to find the splitting variable and the threshold that minimizes:

$$\min_{j, \theta_j} \Big[\min_{\hat{y}_1} \sum_{x_i \in R_1(j, \theta_j)} (y_i - \hat{y}_1)^2 + \min_{\hat{y}_1} \sum_{x_i \in R_2(j, \theta_j)} (y_i - \hat{y}_2)^2 \Big] \tag{8}$$

$\hat{y}_1$ and $\hat{y}_2$ can be found by Eq.6 for a j and θ_j. The threshold in the outer minimization in Eq.8 can be done quickly by scanning the all input data.

Learning a tree for large dataset such as HHP data can be done very quickly. However, the output a CART will suffer from overfitting and noise. Bagging is a technique for reducing the variance of output of different low biased models which is achieved by averaging their output. Random Forest is a bagging technique proposed for CART that average the output of many noisy trees in order to reduce the variance of output. Creating a Random Forest can be done simply in an iterative algorithm where in each iteration a CART is created from a sample subset of training data and a sample subset of input features. The final output of the forest will be the average of output of each tree created in the learning.

3.4 Feature Learning by K–means

In all the methods, discussed above we use a hand designed feature extracted from the claims. Even though these feature are interpretable by human, but they are not necessarily good predictors. Many feature learning has been proposed in machine learning community for learning features from the data i.e. RBM [10] and sparse auto-encoders [11]. In this section we review a simple feature proposed in [1].

In this technique the raw feature, X is first whitened by projecting the data to its principal components and diving each variable in the projection space by its standard deviation. Then, K–mean clustering technique is run on the data to extract K center of clusters, c^k. The final feature is formed by:

$$f^k(x) = \max \{0, \mu(z) - z_k\} \tag{9}$$

where $z_k = \|x - c^k\|_2$ and $\mu(z)$ is the mean of elements of z. In this case, the feature f^k will be zero where the distance of the x and the center c^k is above the average. The main advantage of this

Table 2: Performance of different techniques on validation dataset, and their corresponding coefficient in final model.

Method	Linear Regression	Neural Network	Random Forest	Kmeans Feature
MSE	0.4524	0.4518	0.4510	0.4496
Coefficient	0.0014	0.1416	0.2918	0.2707

technique is its simplicity. Unlike other techniques that has many parameters to tune, in this method we only need to set the number of clusters. Next, we will train a linear regression model on top of this feature and we show how new feature performs with respect to the other models.

4 Experiments

In this section we examine the techniques we reviewed in previous section on HHP data. In HHP competition the data for first two years is available for training. The third year is separated for test purpose. Therefore, the groundtruth for the third year is not available. Here, we use first year's data for the training, and the second year's data for validation in order to tune the parameters of different techniques. Finally, we will form our final classifier by combining all methods, we will submit a score file to the competitions website to evaluate our system with other team members.

Each year's data consists of about 70,000 samples. Therefore, training our models with all possible parameters for some methods especially neural network may take a very long time. For this purpose we only tune the main parameters of methods, and we limit the domain of parameters to small number of choices.

In Fig. 1 we visualized performance all techniques with using different parameters. On all these figures the y axis is the Mean Square Error(MSE) between output of a model and the groundtruth. Since we fit our models to the logarithm of ground truth, this measure is equal to the performance measure used in the competition.

Linear Regression: This model has only one parameter to tune which is the regularizer trade off coefficient, λ. We changed λ from 10^{-4} to 10^{2}, and the best result was achieved at $\lambda = 10^{-2}$.

Neural Network: This model has many parameters to tune, such as transfer function in hidden node, number of hidden nodes, step size and batch size in training. Unfortunately,the method is also very slow on large scale data. Hens, we set all parameter to their default setting in the implementation except for the transfer function type and number of hidden neurons. We examined two transfer functions: radial basis function (radial) $u(x) = e^{x^2}$, and hyperbolic tangent sigmoid function (tansig) $u(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$. We can see in Fig. 1 that the latter function outperforms other choices on the validation set using four hidden neurons.

Random Forest: For random forest there are a few parameters to tune. We set sampling size of training data to 10% of whole training data to prevent training from becoming very slow. We also set feature sample size to 33% of all feature so that we can have a proper set of all good predictors. In Fig. 1 we visualize the performance this model versus different number of trees in model. Suprisingly, as we increase the number of trees in the model the error decreases, but the model never overfits into our problem. This shows the robustness of random forest against overfitting.

K–means Feature: We only need to set the number of clusters for this model. In Fig. 1 we show how inceasing the number of clusters improves the performance. Note that we used regularized linear regression on top of this feature, so we also need to cross validate on different values of this parameter. Here, we changed λ from 10^{-4} to 10^{2} but we show the best λ for each K, number of clusters.

In Table. 2 we compared the best performance of different techniques on the validation set. We can see neural netwok works slightly better than linear regression model, and random forest model outperforms both those methods. Finally, we can see a simple linear model trained on k–means features has the best accuracy.

Now that we have the output of different methods on the validation dataset, we can train a linear model that combines all these models together and fuse them into one model. We train this model, and we obtained the coefficients in the seconds row of Table. 2. In order to create our submission for the competition, we applied all the models to the test data and we combined their output using the coefficients trained from the validation dataset. Our submission had 0.4641 error on the test dataset, and was ranked in the top 25% of all 700 participants of the competition while the current leading entry has 0.4542 error.

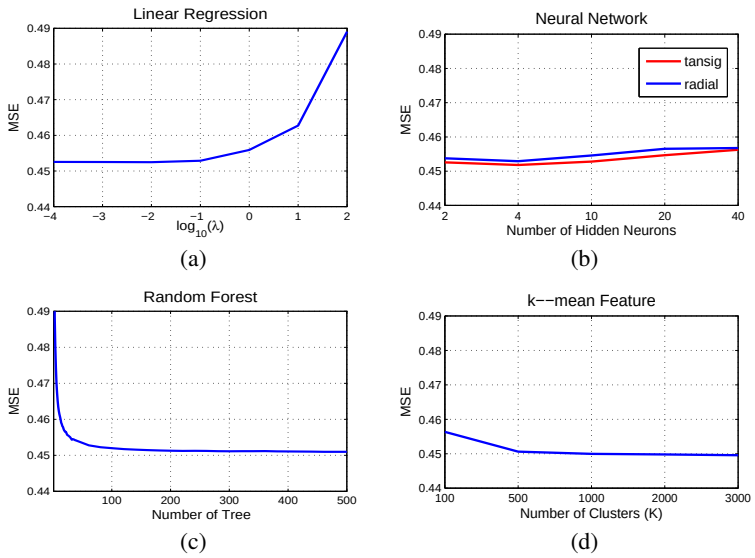

Figure 1: Performance of different methods on validation dataset using different parameters. a) linear regression model using different λ parameter. b) Neural Netwoks with radial basis function (radial) or hyperbolic tangent sigmoid function (tansig) and different number of hidden neurons. c) Different number of trees in Random Forest Model. d) different number of clusters for Kmeans feature.

5 Conclusions

In this report, we compared performance of different regression methods on the HHP dataset. We also showed how k–means features outperforms all other models on the validation set. We finally formed a model by linearly combining all the models we reviewed. We also showed that our ensambled model achieved comparable result in HHP competition leaderboard.

Howevere, due to the limited time, a few unexplored paths remained for future work. In learning process we did not tune all parameters for all the models. For example, larger sample size for Random Forest method may produce better model. We also used kmeans feature in a linear model. We could feed this feature to other models. In feature extraction, we ignored the temporal aspect of data. A subset of patients have their claims available for two years. A model that considers first two years for predicting the third year may have better result. But it might be problematic for the patients with claims only for a year.

References

[1] Adam Coates, Honglak Lee, and Andrew Y. Ng, "An analysis of single-layer networks in unsupervised feature learning", 2011.

[2] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[3] Trevor Hastie, Robert Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*, Springer, July 2001.

[4] R. Axelrod p. Brierley, D. Vogel, "Heritage provider network health prize, round 1 milestone prize, how we did it team market makers", 2011.

[5] Leo Breiman, "Random forests", *Machine Learning*, vol. 45, no. 1, pp. 5, 2001.

[6] Jerome H. Friedman, "Greedy function approximation: A gradient boosting machine.", Oct. 2001.

[7] W. Mestrom, "My milestone 1 solution to the heritage health prize", 2011.

[8] K. Levenberg, "A method for the solution of certain non-linear problems in least squares", *Quarterly Applied Mathematics*, vol. II(2), pp. 164–168, 1944.

[9] L. Brieman, J. H. Friedman, R. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth International, 1984.

[10] Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh, "A fast learning algorithm for deep belief nets", *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[11] Ian J. Goodfellow, Quoc V. Le, Andrew M. Saxe, Honglak Lee, and Andrew Y. Ng, "Measuring invariances in deep networks", in *NIPS*, Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, Eds. 2009, pp. 646–654, Curran Associates, Inc.