# Case study:predicting home-care duration by machine learning

Qiurui Chen

September 10, 2018

**Abstract**

Buurtzorg is a company provides home-care service. So, predicting home-care duration weakly for each patient is important for them.

In this case study, machine learning methods are applied to predict home-care duration for next week. Multiple classifiers are tried but they all failed. But Gradient-boosted trees(GBTs) and Deep neural networks(DNN) for regression perform quite well and they all reach nearly 0.99 coefficient of determination($R^2$). GBTs got better results on mean absolute error(MAE) 0.2130 and Root Mean Square Error(RMSE) 9.0385 , however, DNN only got MAE 7.0359 and RMSE 13.1125. But RMSE variance of DNN(19.9206) is much lower than GBTs(36.3063). So GBTs will predict more accurate care-duration, but it is sensitive to outliers while DNN is much more stable.

## 1 Introduction

Buurtzorg is a company that provides residential care and community care.Predicting care duration for each patient is a big concern for them, so that they can arrange medical teams beforehand effectively.

This article aims at predicting weakly home-care duration with the data provided by Buurtzorg company. Section 2 introduces some techniques applied in this experiment.Then, methodology that employed in this experiment is introduced in Section 3. After that,results of all experiments are showed in Section 4.Subsequently, Section 5 is discussion and conclusion about this case study.

The roughly steps of this experiment are that data analysis, preprocessing data, training models and picking the best one. Data is first explored by exploratory data analysis(EDA) to gain some insights.Then, after preprocessing data,multiple machine learning models are applied, including logistic regression(LR), decision tree(DT), random forest(RF), multilayer perceptron classifier (MLP), one-vs-rest classifier(OvR),boosted-gradient trees(GBTs), deep neural network(DNN).Problem is subdivided into regression and multi-classes classification two sub-problem. After testing simple models, best two models stand out and they are trained deeper with more hyper-parameters. Finally, GBTs got best performance but it is much sensitive to outliers while DNN got a slightly worse results but it is much more stable with respect to GBTs.

## 2 Background

### 2.1 Linear Regression

In statistics, linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables)[1]. Given a data set $\{y_1, x_{i_1}, ...x_{i_p}\}_{i=1}^{n}$ of n statistical units, a linear regression model assumes that the relationship between the dependent variable $y$ and the p-vector of regression $x$ is linear. This relationship is modeled through a disturbance term or error variable $\epsilon$ — an unobserved random variable that adds "noise" to the linear relationship between the dependent variable and regressors. Thus the model takes the form:

$$y_i = \beta_0 1 + \beta_1 X_{i_1} + ..... + \beta_p X_{i_p} + \epsilon_i = X_T^i \beta + \epsilon_i, i = 1, ...., n,$$

where $^T$ denotes the transpose, so that $x_i^T \beta$ is the inner product between vectors $x_i$ and $\beta$[1].

In this experiment, a combination of L1 and L2 are used as regularization. it is called elastic net[2], and its form is listed below:

$$\alpha(\lambda||w||_1) + (1-\alpha)(\frac{\lambda}{2}||w||_2{}^2), \alpha \in [0,1], \lambda \geq 0 \tag{1}$$

By setting $\alpha$ properly, elastic net contains both L1 and L2 regularization as special cases.

## 2.2 Generalized Linear Regression

Contrasted with linear regression where the output is assumed to follow a Gaussian distribution, generalized linear models (GLMs) are specifications of linear models where the response variable $Y_i$ follows some distribution from the exponential family of distributions[3]. GLMs require exponential family distributions that can be written in their "canonical" or "natural" form, aka natural exponential family distributions. The form of a natural exponential family distribution is given as:

$$f_Y(y|\theta,\tau) = h(y,\tau)\exp\left(\frac{\theta \cdot y - A(\theta)}{d(\tau)}\right)$$

where $\theta$ is the parameter of interest and $\tau$ is a dispersion parameter. In a GLM the response variable $Y_i$ is assumed to be drawn from a natural exponential family distribution:

$$Y_i \sim f\left(\cdot|\theta_i,\tau\right)$$

where the parameter of interest $\theta_i$ is related to the expected value of the response variable $\mu_i$ by

$$\mu_i = A'(\theta_i)$$

Here, $A'(\theta_i)$ is defined by the form of the distribution selected. GLMs also allow specification of a link function, which defines the relationship between the expected value of the response variable $\mu_i$ and the so called linear predictor $\eta_i$:

$$g(\mu_i) = \eta_i = \vec{x_i}^T \cdot \vec{\beta}$$

Often, the link function is chosen such that $A' = g^{-1}$, which yields a simplified relationship between the parameter of interest $\theta$ and the linear predictor $\eta$. In this case, the link function $g(\mu)$ is said to be the "canonical" link function.

$$\theta_i = A'^{-1}(\mu_i) = g(g^{-1}(\eta_i)) = \eta_i$$

A GLM finds the regression coefficients $\vec{\beta}$ which maximize the likelihood function:

$$\max_{\vec{\beta}} \mathcal{L}(\vec{\theta}|\vec{y},X) = \prod_{i=1}^{N} h(y_i,\tau)\exp\left(\frac{y_i\theta_i - A(\theta_i)}{d(\tau)}\right)$$

## 2.3 Logistic Classification

Multi-class classification is supported via multinomial logistic (softmax) regression. In multinomial logistic regression, the algorithm produces K sets of coefficients, or a matrix of dimension $K \times J$ where $K$ is the number of outcome classes and $J$ is the number of features. If the algorithm is fit with an intercept term then a length $K$ vector of intercepts is available. the confitional probabilities of the outcome classes $k \in 1, 2, \ldots, K$ are modeled using the softmax function:

$$P(Y=k|\mathbf{X},\boldsymbol{\beta}_k,\beta_{0k}) = \frac{e^{\boldsymbol{\beta}_k \cdot \mathbf{X} + \beta_{0k}}}{\sum_{k'=0}^{K-1} e^{\boldsymbol{\beta}_{k'} \cdot \mathbf{X} + \beta_{0k'}}}$$

The weighted negative log-likelihood should be minimized by a multinomial response model, with elastic-net penalty to control for overfitting.

$$\min_{\beta,\beta_0} -\left[\sum_{i=1}^{L} w_i \cdot \log P(Y=y_i|\mathbf{x}_i)\right] + \lambda\left[\frac{1}{2}\left(1-\alpha\right)||\boldsymbol{\beta}||_2^2 + \alpha||\boldsymbol{\beta}||_1\right] \tag{2}$$

## 2.4 Decision Tree

Decision trees are widely used since they are easy to interpret, handle categorical features, extend to the multi-class classification setting, do not require feature scaling, and are able to capture non-linearities are feature interactions. The decision tree is a greedy algorithm that performs a recursive binary partitioning of the feature space. The tree predicts the same label for each bottommost (leaf) partition. Each partition is chosen greedily by selecting the best split from a set of possible splits, in order to maximize the information gain at a tree node. In other words, the split chosen at each tree node is chosen from the set $\underset{s}{\text{argmax}} IG(D, s)$ where $IG(D, s)$ is the information gain when a split s is applied to a dataset $D$.

The node impurity is a measure of the homogeneity of the labels at the node. The current implementation provides two impurity measures for classification (Gini impurity and entropy) and one impurity measure for regression (variance).

Here Gini impurity is applied for classification and variance is employed for regression. Gini impurity:

$$\sum_{i=1}^{C} f_i(1 - f_i)$$

$f_i$ is the frequency of label $i$ at a node and $C$ is the number of unique labels.

variance:

$$\frac{1}{N}\sum_{i=1}^{N}(y_i - \mu)^2$$

$y_i$ is label for an instance, $N$ is the number of instances and $\mu$ is the mean given by $\frac{1}{N}\sum_{i=1}^{N} y_i$

The information gain is the difference between the parent node impurity and the weighted sum of the two child node impurities. Assuming that a split s partitions the dataset D of size N into two datasets $D_left$ and $D_right$ of sizes $N_left$ and $N_right$, respectively, the information gain is:

$$IG(D, s) = Impurity(D) - \frac{N_{left}}{N} Impurity(D_{left}) - \frac{N_{right}}{N} Impurity(D_{right})$$

In spark-ML model, there are three conditions that will lead tree stop splitting:

- the node depth is equal to the max-depth training parameter
- No split candidate leads to an information gain greater than min-infomatrion-gain parameter.
- No split candidate produces child nodes which each have at least min-Instances-for-per-Node(which is a training parameter) training instances. The algorithm injects randomness into the training process so tree reduces the variance of the predictions, improving the performance on test data.

## 2.5 Random Forest

Random forests are ensembles of decision trees. They combine many decision trees in order to reduce the risk of over-fitting.

In training phrase,the randomness injected into the training process includes:

- Sub-sampling the original dataset on each iteration to get a different training set (a.k.a. bootstrapping).
- Considering different random subsets of features to split on at each tree node.

Apart from these randomizations, decision tree training is done in the same way as for individual decision trees.

In prediction phrase, to make a prediction on a new instance, a random forest must aggregate the predictions from its set of decision trees. This aggregation is done differently for classification and regression.

- Classification: Majority vote. Each tree's prediction is counted as a vote for one class. The label is predicted to be the class which receives the most votes.
- Regression: Averaging. Each tree predicts a real value. The label is predicted to be the average of the tree predictions.

## 2.6 Gradient-Boosted Trees

Gradient-Boosted Trees(GBTs) are ensembles of decision trees. Gradient boosting iteratively trains a sequence of decision trees. On each iteration, the algorithm uses the current ensemble to predict the label of each training instance and then compares the prediction with the true label. The dataset is re-labeled to put more emphasis on training instances with poor predictions. Thus, in the next iteration, the decision tree will help correct for previous mistakes.

The specific mechanism for re-labeling instances is defined by a loss function (discussed below). With each iteration, GBTs further reduce this loss function on the training data[4]. In this experiment, Log loss are used, formula is listed blow:

$$2\sum_{i=1}^{N}\log(1+\exp(-2y_iF(x_i)))$$

where

$$\begin{aligned} N &= number\ of\ instances. \\ y_i &= label\ of\ instance\ i. \\ x_i &= features\ of\ instance\ i. \\ F(x_i) &= model's\ predicted\ label\ for\ instance\ i \end{aligned}$$

## 2.7 Neural Network

Neural networks(NN), with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques[5].

### 2.7.1 How Many Layers

How many hidden layers and how many neurons in each layer is a big concern for neural network. As for hidden layer numbers, Jeff Heaton[6] pointed out that if hidden layer is 0, then this NN is only capable of representing linear separable functions or decisions, if there is only one hidden layer, then the NN can approximate any function that contains a continuous mapping from one finite space to another. If there is more than 1 hidden layers, then NN model can represent an arbitrary decision boundary to arbitrary accuracy with rational activation functions and can approximate any smooth mapping to any accuracy. Also, but too many layers will result over-fitting and require much more data to train, which will increase computation and result more time to train.

### 2.7.2 How Many Neurons In One Layer?

Also ,for how many neurons on each layer, Howard B. Demuth et al.[7] stated a formula to help define the neurons number in each layer.

$$N_h = \frac{N_s}{(\alpha * (N_i + N_o))} \tag{3}$$

$$\begin{aligned} N_i &= number\ of\ input\ neurons. && (4) \\ N_o &= number\ of\ output\ neurons. && (5) \\ N_s &= number\ of\ samples\ in\ training\ data\ set. && (6) \\ \alpha &= an\ arbitrary\ scaling\ factor\ usually\ 2-10. && (7) \end{aligned}$$

As explained by Howard B.Demuth et al[7], the goal is to limit the number of free parameters in NN model (its degree or number of nonzero weights) to a small portion of the degrees of freedom in the data. The degrees of freedom in the data is the number samples multiple degrees of freedom (dimensions) in each sample or $N_s * (N_i + N_o)$ (assuming data are independent). So $\alpha$ is a way to indicate how general you want your model to be, or how much you want to prevent over-fitting.

### 2.7.3 Which Activation Function?

In artificial neural networks, the activation function of a node defines the output of that node given an input or set of inputs. Activation functions are really important for a Artificial Neural Network to learn and make sense of something really complicated and Non-linear complex functional mappings between the inputs and response variable.They introduce non-linear properties to our Network[8].

In this experiment, Relu activation is applied. Rectified linear unit(ReLU) is defined as:

$$f(x) = \begin{cases} 0 & x < 0 \\ x & x \geq 0 \end{cases}$$

The advantage of ReLU vanish gradient. And more it is more computationally efficient to compute than sigmoid like functions since Relu just needs to pick max(0,x) and not perform expensive exponential operations as in Sigmoids. Alex Krizhevsky et all[9] pointed out that in practice, networks with Relu tend to show better convergence performance than sigmoid. But Relu tends to blow up activation (since there is no mechanism to constrain the output of the neuron, as input itself is the output if input larger than zero).Also, if too many activations get zero then the most of the neurons in network with Relu will simply output zero, in other words, die and thereby prohibiting learning[10].

### 2.7.4 Reducing Over-fitting

Since deep neural network(DNN) is quite complex and has strong learning compacity, it can cause over-fitting easily. So, drop out regularization and early stopping methods are applied to avoid over-fitting.

In early stopping technique, data is splitted into training,validation and testing sets. When the network begins to overfit the data, the error on the validation set typically begins to rise. When the validation error increases for a specified number of iterations, the training is stopped and the weights and biases at the minimum of the validation error is the best parameter for DNN model.

Dropout is technique where randomly drop components of neural network (outputs) from a layer of neural network. The neurons which are “dropped out” in this way do not contribute to the forward pass and do not participate in backpropagation. So every time an input is presented, the neural network samples a different architecture. This technique reduces complex co-adaptations of neurons, since a neuron cannot rely on the presence of particular other neurons. It is, therefore, forced to learn more robust features that are useful in conjunction with many different random subsets of the other neurons[11].

### 2.7.5 Other Parameters Need To Be Tuned

There are other hyper-parameters needed to be tuned. They are:

- The batch size controls how many of the training set data points are randomly exposed to the optimization process at each iteration. This has the effect of reducing potential overfitting by providing some randomness to the optimization process. Batch sizes between 10 to 40K were considered.
- The learning rate parameter controls the rate of decent during the parameter estimation iterations and these values were contrasted to be between zero and one.
- A decay rate that decreases the learning rate over time (ranging between zero and one).

## 2.8 Feature transforming

### 2.8.1 Feature scaling

Feature scaling is a method used to standardize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step[12].

Feature scaling including rescaling(min-max normalization), mean normalization, standardization and scaling to unit length. In this experiment, standardization is applied. Feature standardization makes the values of each feature in the data have zero-mean (when subtracting the mean in the numerator) and unit-variance[12]. Each feature is use the formula below to transform:

$$x' = \frac{x - \bar{x}}{\sigma}$$

where $x$ is the original feature vector $\bar{x}$ is the mean of that feature vector, and $\sigma$ is its standard deviation.

Feature scaling is general trick applied to optimization problems. With scaled features, the contour of the cost function might look like circles; then the gradient can take much more straight path and achieve the optimal point much faster, otherwise the contour of the cost function can look like very tall and skinny ovals, which will take the gradients for a long time to go back and forth to find the optimal solution[13].

scaling and centering these feature transformations are mostly innocuous and are typically needed when the model requires the features to be in common units. For example, when the distance or dot products between predictors are used (such asK-nearest neighbors or support vector machines) or when the variables are required to be a a common scale in order to apply a penalty (e.g.the lasso or ridge regression), a standardization procedure is essential[14].

### 2.8.2 Yeo-Johnson transformation

In statistics, a power transform is a family of functions that are applied to create a monotonic transformation of data using power functions. The Box-Cox transformation is a family of power transformation functions that are used to stabilize variance and make dataset look like a normal distribution,Yeo-Johnson transformation is similar with Box-cox,but it also allows for zero and negative values.

The reason to do variance-stablizing transformation is to limit/remove the relationship between mean and variance. One of very important assumptions of linear regression is the constant variance (aka homoskedasticity). Violation of the assumption will lead to "less precise parameter estimates and misleading inferential quantities such as standard errors"[15].

The Yeo-Johnson transformation law reads:

$$y_i^{(\lambda)} = \begin{cases} ((y_i+1)^{\lambda} - 1)/\lambda & if \lambda \neq 0, y \geq 0 \\ log(y_i + 1) & if \lambda = 0, y \geq 0 \\ -[(-y_i+1)^{(2-\lambda)} - 1]/(2-\lambda) & if \lambda \neq 2, y < 0 \\ -log(-y_i + 1) & if \lambda = 2, y < 0 \end{cases}$$

### 2.8.3 One-Hot-Encoding

In statistics and econometrics, particularly in regression analysis, a dummy variable (also known as an indicator variable, design variable, Boolean indicator, binary variable, or qualitative variable[1][2]) is one that takes the value 0 or 1 to indicate the absence or presence of some categorical effect that may be expected to shift the outcome.[3][4] Dummy variables are used as devices to sort data into mutually exclusive categories (such as smoker/non-smoker, etc.)[16]. One-hot-encoding is to transform categorical data into dummy variable form.

Compare to integer encoding, one-hot encoding effectively blow up the feature space, and create feature for each possible variables in categorical predictor, so that each variable will get their own weights, rather than the whole categorical predictor. So one-hot encoding performs much better than integer encoding for most machine learning algorithms ( except for decision tree, if it is deep enough, it can handle categorical variables without one-hot encoding).

### 2.8.4 Zero- and near zero-variance features

Datasets come sometimes with predictors that take an unique value across sample. This kind of predictor is non-informative, and also it can break some learning models. Constant and almost constant predictors across samples (called zero and near-zero variance predictors in [17], respectively) happens quite often. One reason is because we usually break a categorical variable with many categories into several dummy variables. Hence, when one of the categories have zero observations, it becomes a dummy variable full of zeroes[18]. The quick and brute solution is to throw these features away.

In this experiment, nearZeroVar from caret package in R language is applied. It not only removes predictors that have one unique value across samples (zero variance predictors), but also removes predictors that have both

- few unique values relative to the number of samples and
- large ratio of the frequency of the most common value to the frequency of the second most common value (near-zero variance predictors).

## 2.9 Evaluation Metrics

for classification, accuracy and F1 score, weighted precision and weighted recall. Define the class ,or label, set as:

$$L = \{\ell_0, \ell_1, \ldots, \ell_{M-1}\}$$

the true output vector $y$ consists of $N$ elements

$$\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N-1} \in L$$

A multi-class prediction algorithm generates a prediction vector $\hat{\mathbf{y}}$ of N elements

$$\hat{\mathbf{y}}_0, \hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_{N-1} \in L$$

A delta function is

$$\hat{\delta}(x) = \begin{cases} 1 & \text{if } x = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Then formula for each metrics are defined below:

$$Accuracy : ACC = \frac{TP}{TP + FP} = \frac{1}{N} \sum_{i=0}^{N-1} \hat{\delta}\left(\hat{\mathbf{y}}_i - \mathbf{y}_i\right) \tag{8}$$

$$Precisionbylabel : PPV(\ell) = \frac{TP}{TP + FP} = \frac{\sum_{i=0}^{N-1} \hat{\delta}(\hat{\mathbf{y}}_i - \ell) \cdot \hat{\delta}(\mathbf{y}_i - \ell)}{\sum_{i=0}^{N-1} \hat{\delta}(\hat{\mathbf{y}}_i - \ell)} \tag{9}$$

$$Recallbylabel : TPR(\ell) = \frac{TP}{P} = \frac{\sum_{i=0}^{N-1} \hat{\delta}(\hat{\mathbf{y}}_i - \ell) \cdot \hat{\delta}(\mathbf{y}_i - \ell)}{\sum_{i=0}^{N-1} \hat{\delta}(\mathbf{y}_i - \ell)} \tag{10}$$

$$F - measurebylabel : F(\beta, \ell) = \left(1 + \beta^2\right) \cdot \left(\frac{PPV(\ell) \cdot TPR(\ell)}{\beta^2 \cdot PPV(\ell) + TPR(\ell)}\right) \tag{11}$$

$$weightedprecision : PPV_w = \frac{1}{N} \sum_{\ell \in L} PPV(\ell) \cdot \sum_{i=0}^{N-1} \hat{\delta}(\mathbf{y}_i - \ell) \tag{12}$$

$$weightedrecall : TPR_w = \frac{1}{N} \sum_{\ell \in L} TPR(\ell) \cdot \sum_{i=0}^{N-1} \hat{\delta}(\mathbf{y}_i - \ell) \tag{13}$$

Since the data is unbalanced, so F1 and weighted precision and weighted recall are all taken into consideration.

For regression, root mean squared error(RMSE), mean absolute error (MAE), and coefficient of determination($R^2$) are applied. Formulas are listed below:

$$MAE = \frac{1}{N} \sum_{i=0}^{N-1} |\mathbf{y}_i - \hat{\mathbf{y}}_i| \tag{14}$$

$$RMSE = \sqrt{\frac{\sum_{i=0}^{N-1} (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2}{N}} \tag{15}$$

$$MSE = \frac{\sum_{i=0}^{N-1} (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2}{N} \tag{16}$$

$$R^2 = 1 - \frac{MSE}{\text{VAR}(\mathbf{y}) \cdot (N - 1)} = 1 - \frac{\sum_{i=0}^{N-1} (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2}{\sum_{i=0}^{N-1} (\mathbf{y}_i - \bar{\mathbf{y}})^2} \tag{17}$$

# 3 Methodology

## 3.1 Dataset

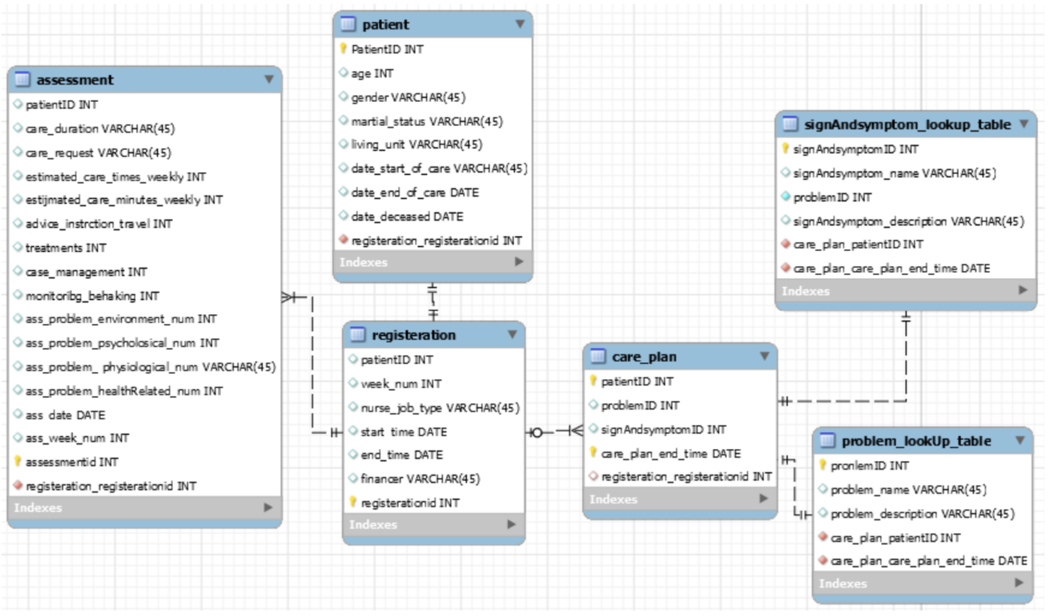
The entity relationship diagram show in 3.1. There are 4 tables and 2 lookup tables.

- Registration: This is data about each home-care that each patient takes, predictors include patientID, care start time, care end time, nurse information.
- Assessment: nurse assessments for each patient about their home care duration and their illness situation, this is done after home care. But assessment is not done after every home-care duration, this is happened when nurse think it is necessary to write a assessment based on patient's health situation changes.
- care plan: care plan information, this is about problem that each patient has.
- Patient: patient information: patient personal information, such as age, matrial status, living status etc.

two look up tables are:

- problem_lookup_table:
  This table provide illness name which is connected with care plan table by IllnessID.
- signAndsympton_lookup_table:
  this table provide symptom name which is connected with care plan table by illnessID and signAndSymptonID.

There are many different data type: categorical, date type and numerical. Detail is listed in 3.1.



assessment
patientID INT
care_duration VARCHAR(45)
care_request VARCHAR(45)
estimated_care_times_weekly INT
estijmated_care_minutes_weekly INT
advice_instrction_travel INT
treatments INT
case_management INT
monitoribg_behaking INT
ass_problem_environment_num INT
ass_problem_psycholosical_num INT
ass_problem_ physiological_num VARCHAR(45)
ass_problem_healthRelated_num INT
ass date DATE
ass_week_num INT
assessmentid INT
registeration_registerationid INT
Indexes
patient
PatientID INT
age INT
gender VARCHAR(45)
martial_status VARCHAR(45)
living_unit VARCHAR(45)
date_start_of_care VARCHAR(45)
date_end_of_care DATE
date_deceased DATE
registeration_registerationid INT
Indexes
registeration
patientID INT
week_num INT
nurse_job_type VARCHAR(45)
start time DATE
end_time DATE
financer VARCHAR(45)
registerationid INT
Indexes
care_plan
patientID INT
problemID INT
signAndsymptomID INT
care_plan_end_time DATE
registeration_registerationid INT
Indexes
signAndsymptom_lookup_table
signAndsymptomID INT
signAndsymptom_name VARCHAR(45)
problemID INT
signAndsymptom_description VARCHAR(45)
care_plan_patientID INT
care_plan_care_plan_end_time DATE
Indexes
problem_lookUp_table
pronlemID INT
problem_name VARCHAR(45)
problem_description VARCHAR(45)
care_plan_patientID INT
care_plan_care_plan_end_time DATE
Indexes

Figure 1: ERR diagram

## 3.2 Work-flow

Figure3.2 shows the work flow for this case study. There are three phrases in this workflow: data collection, preprocessing data and training models. In the first stage, empty data and outliers are first removed by Exploratory Data Analysis(EDA) since the amount of these data is quite small and there is no needs to do any data imputation. Then based on the understanding of the data, useful predictors are selected, also, some predictors are combined into a newly feature. Feature

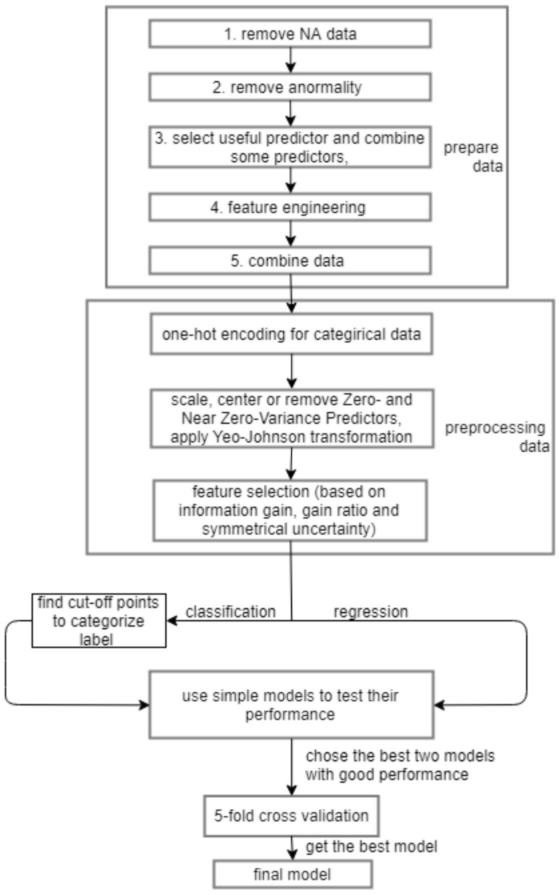

1. remove NA data
2. remove anormality
3. select useful predictor and combine some predictors,
4. feature engineering
5. combine data
prepare data
one-hot encoding for categirical data
scale, center or remove Zero- and Near Zero-Variance Predictors, apply Yeo-Johnson transformation
feature selection (based on information gain, gain ratio and symmetrical uncertainty)
preprocessing data
find cut-off points to categorize label
classification
regression
use simple models to test their performance
chose the best two models with good performance
5-fold cross validation
get the best model
final model

Figure 2: Overall work-flow

engineering is applied afterwards based on EDA and domain-knowledge. The last step of this phrase is combining all tables together and create the response for the corresponding need.

In the second preprocessing-data stage, one-hot encoding is applied to categorical data. Then scaling, centering, removing zero- and near-zero variance predictors and Yeo-Johnson transformation are employed. Last, feature are selected by entropy-based filter methods.

Then data is trained in classifiers or regressions. In classifiers, proper cut-off points to the response are needed to categorize the label. After testing simple models, best two models stand out. Then 5-fold cross validation is applied to train these two. At last, choose the best one as the final model.

In this case study, simple classifiers are LR, DT,RF and OvR while simple regressors include LR, generalized linear regression, DT, RF, GBTs and DNN(Figure3 shows the DNN structure). The evaluation metrics for classification are F1-score, accuracy, weighted precision and weighted recall, while for regression, RMSE, R2 and MAE are applied as measurements.

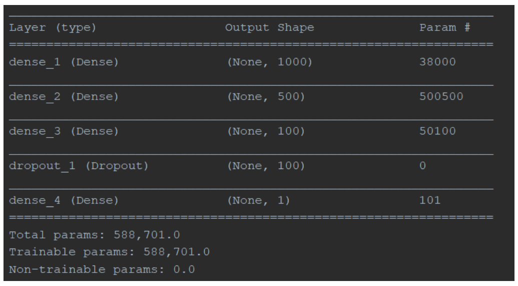
The detailed steps are listed in appendixA.

## 4 Results

### 4.1 EDA

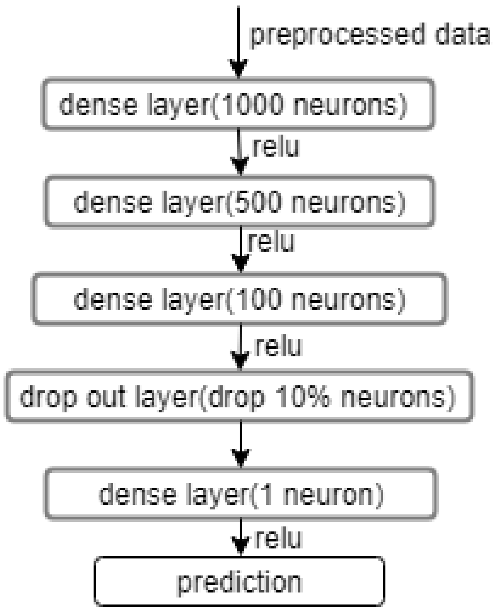
EDA is applied to gain some insights about the dataset. Also, some features are created based on these descriptive statistics analysis.

Figure4 shows the relationship between features about assessment date and the response. Figure4a shows some months contain very highly care duration data, such as January, February, July, June, August and December. Also, Figure4b indicates that long duration home-cares scatter on some particular date, like, date 5,6,8 and 15-18, 20,23,25,26,28 and 30. Since all data points are not equally distributed, all these predictors can provide some information for the response.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 1000)              38000
_________________________________________________________________
dense_2 (Dense)              (None, 500)               500500
_________________________________________________________________
dense_3 (Dense)              (None, 100)               50100
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 101
=================================================================
Total params: 588,701.0
Trainable params: 588,701.0
Non-trainable params: 0.0
```

(a) model summary



(b) model structure

Figure 3: neural network model used in case study

Figure5a shows that "less than 3 months" category accounts for the most response, including the high duration time and the middle range duration from 3000 to 6000 minutes. Figure5d indicates that "increasing care" has the strong connection with long duration cares. Also, "decreasing care" corresponds mostly to the short duration cares while "stable care demand" mostly corresponds to short duration cares but also with some specific extremes, such as care duration around 5000 minutes. Base on common sense, it is assumed that the weekly total home care duration should follow the amount of home-care, that is, they increase both or decrease together, but Figure5b tells a different story. Data are augmented in some specific care amounts, such as 7,14,21,28 and 35, especially in care amount 14, lots long duration cares also scattered here. Besides, the dots do not follow the forward-diagonal line of the plot, which is out of expectation. The same happened for nurse "estimated care duration" predictor, if it is accurate then the scatter plot of this predictor with respect to the response should looks like a diagonal line. But Figure5b shows that the nurse estimation is not so accurate, and nurses are inclined to estimate around 4800 minutes for long-duration cares.

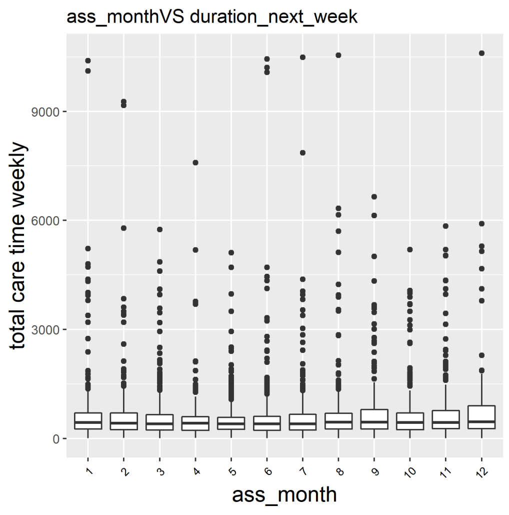
Figure6a, Figure6b, Figure6c and Figure6d shows that they are informative since the distribution for 0 and 1 are not equal.

Figure7a, Figure7b, Figure7c and Figure7d can provide some weak information about the response since the distribution for each values in each tables are quite different. Figure7e shows nurse job title can also be informative, say, most long duration cares located on "verpleegkundig specialist", "verzorgende niveau 3" and "wijkverpleegkundige/Verpleegkundig Specialist" those 3 categories. Also, "nurse team" predictor is useful since Figure7f shows some patterns, such as long duration cares are only distributed in some specific teams.

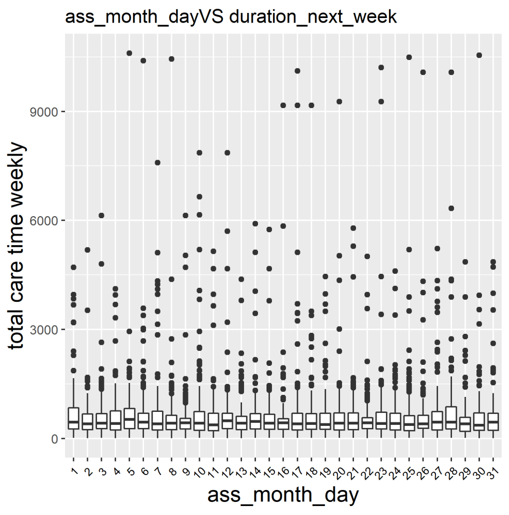
Figure8a points that most patients are old, and there is no long duration for young patient. Figure8b shows that care duration for the male and the female are slightly different. Figure8c points that some care durations are specifically fond of some particular living units, such as care duration around 8000 minutes only located on "unknown" category. The similar pattern can be detected in Figure8d

Figure9 shows the relationship between the response and the newly created features which are based on home-care happened date. Figure9a shows week_day is non-informative since each category has quite the same distribution. The multiple horizontal lines on Figure**??** show that care duration is consistent in some continuous days. The same pattern follows on Figure9d.

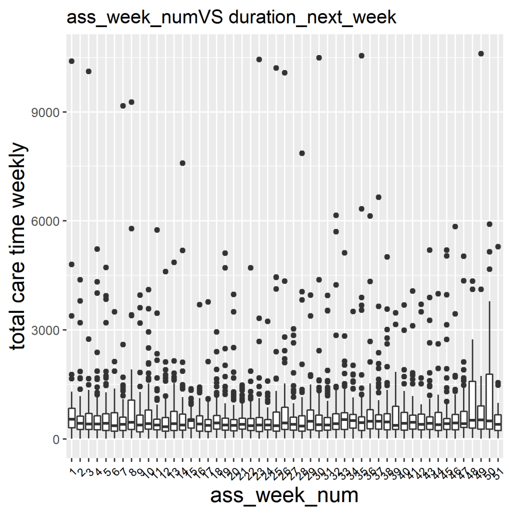
Figure10 shows that the newly created features are informative. Features about the previous one week home-care duration are created. Since Figure**??** shows some duration consistency during several dates, features about care duration in previous one week are created. The assumption is



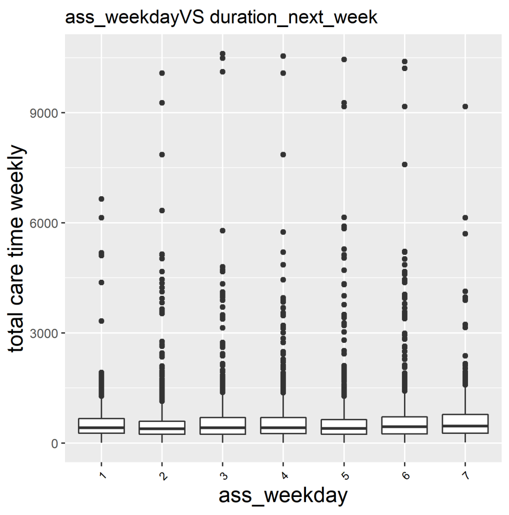
(a) month



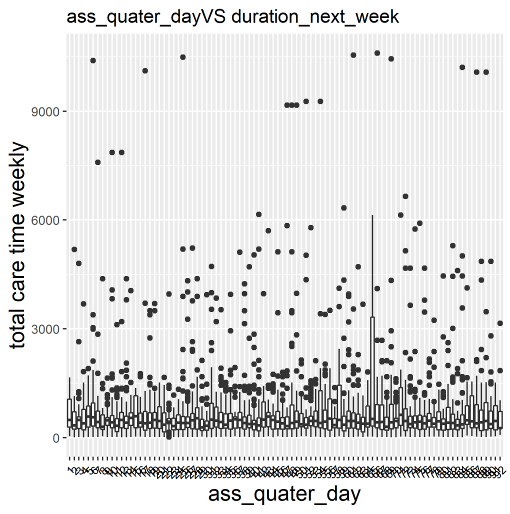
(b) date on month



(c) week number



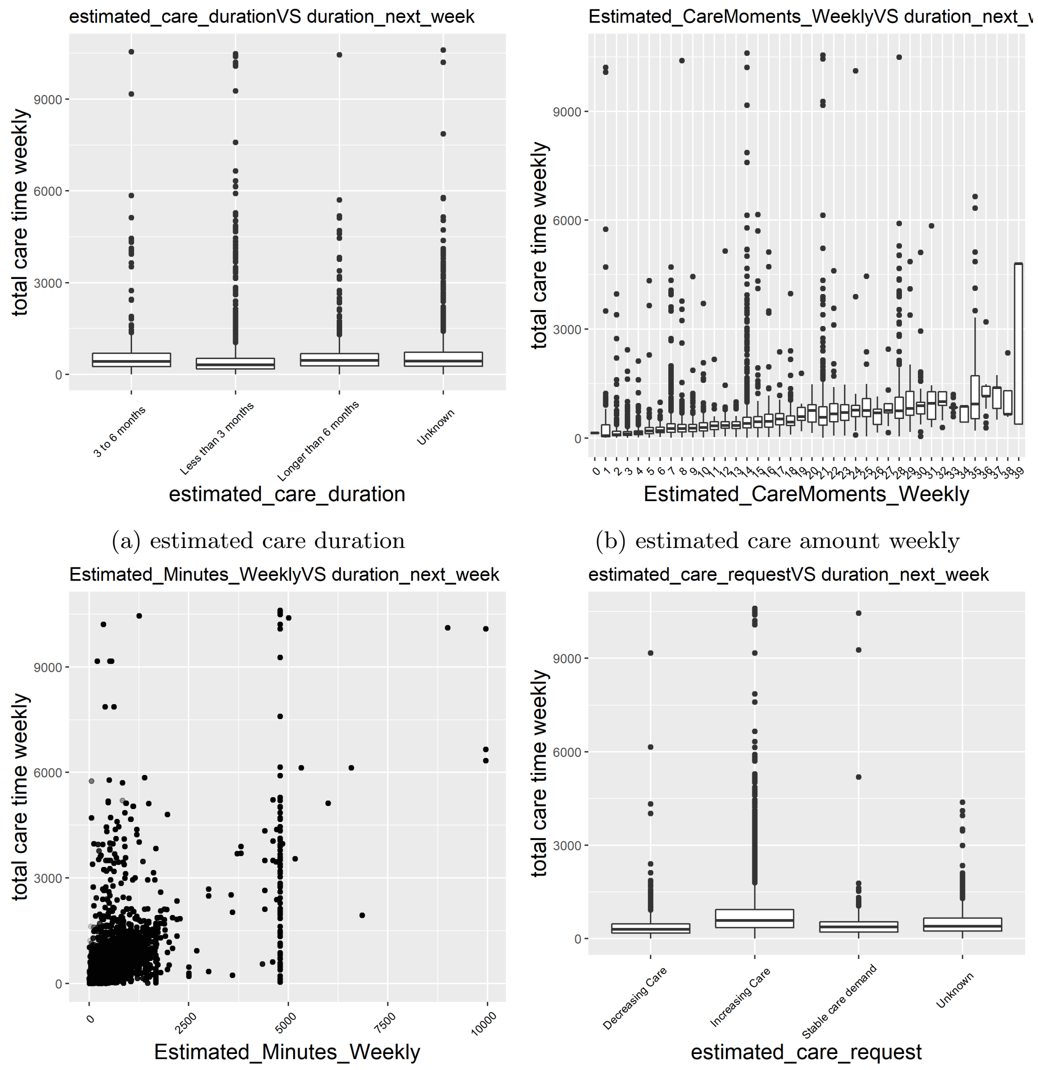
(d) week day



(e) quarter day

Figure 4: A scatter plot of predictors created with the assessment date and the response

that care-duration should be the same in continuous two weeks, that is, in scatter plots 10, data dots should fit a diagonal line. Although Figure10 does not follow this rule strictly, there still are



(a) estimated care duration

(b) estimated care amount weekly

(c) estimated care amount weekly

(d) estimated care request

Figure 5: Scatter plots of care estimation made by nurses and the response

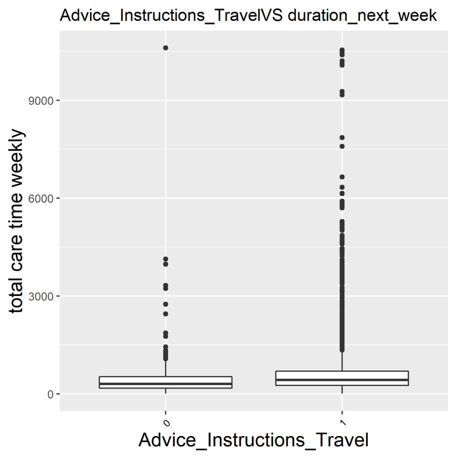
some patterns.

Figure4.1 shows the heatmap of correlation matrix. The organizational structure is visualized using a dendrogram (top and left). The dendrogram highlights at least two groups of predictors. Since the assessment date and home-care happened date are within the same week number, features extracted by these two predictors have strong correlation( red color in the the middle of ). Some corrections fit common sense, such as "decreasing care" estimated care request (left bottom) predictor has strong positive correlation with "less than 3 month" estimated care duration(left bottom). However, some findings are out of interpretation, say, "increasing care"(a estimated care request feature) that has a negative correlation with "healthRelated_num" (health related problem amount feature which is estimated by nurse).

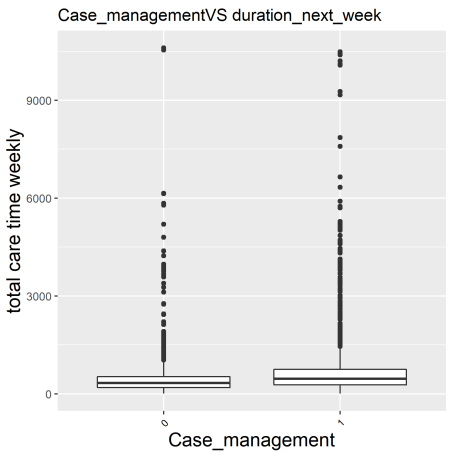
## 4.2 Classification Result

In test phrase, some simple classifiers are applied to test classification performance. Figure4.2 shows that data are unbalanced.Bar chart shows that the distribution for the response is highly skewed with a long tail.Violin plot indicates that there are three peaks in distribution: one is very huge at the beginning, another is in the middle, the last peak located at the end of the distribution. Boxplot indicates that there are lots outliers. So, it’s quite intractable to find proper cut-off-point to categorize the response.

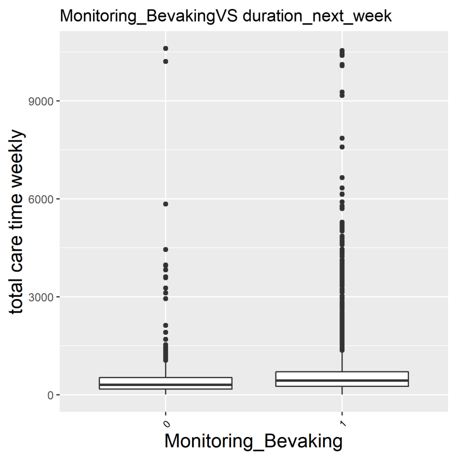
In this experiment,the label is categorized into 31 classes, 1-23 classes correspond to 1-23 hours



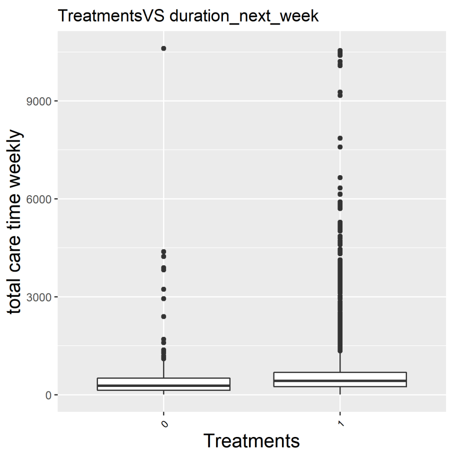
(a) Advice Instructions Travel



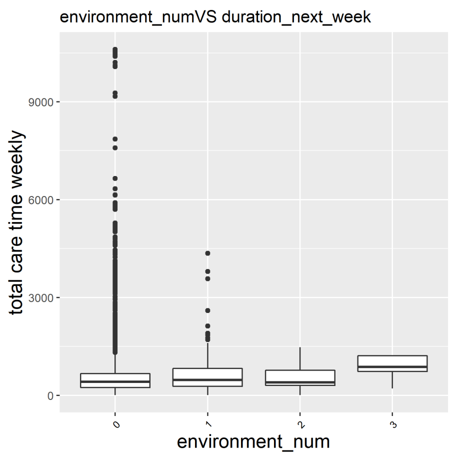
(b) Case management



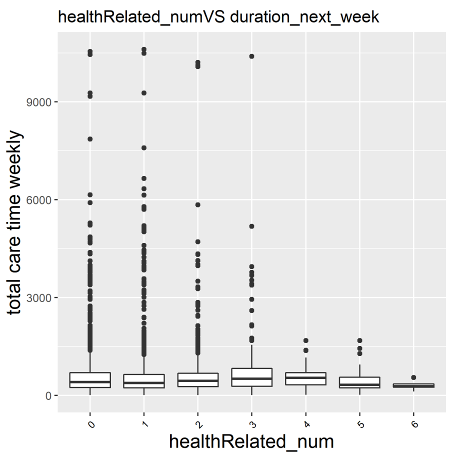
(c) monitoring



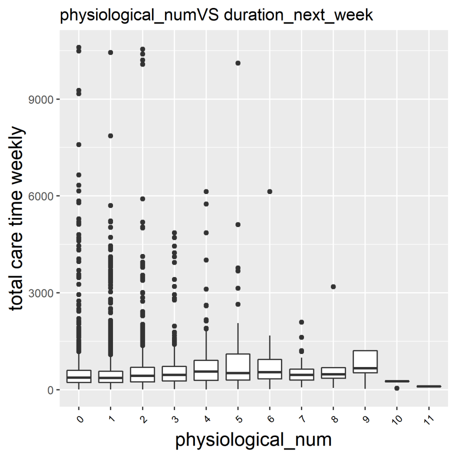
(d) treatment



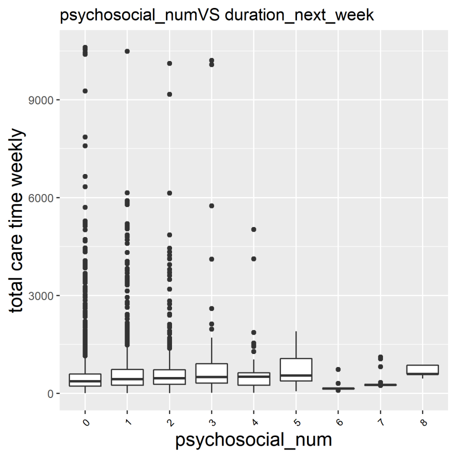
(a) environment related problems amount



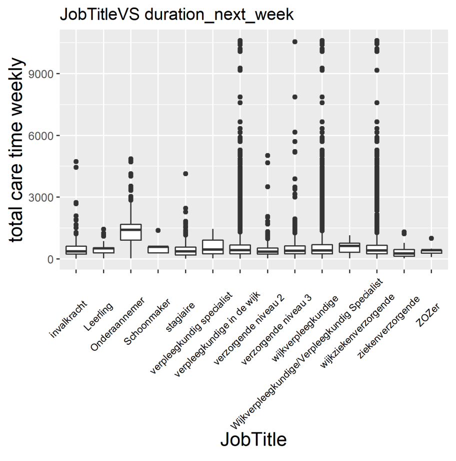
(b) health related problems amount



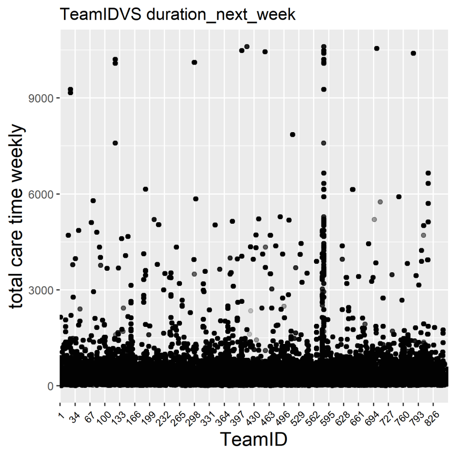
(c) physiological related problems amount



(d) psychosocial related problems amount

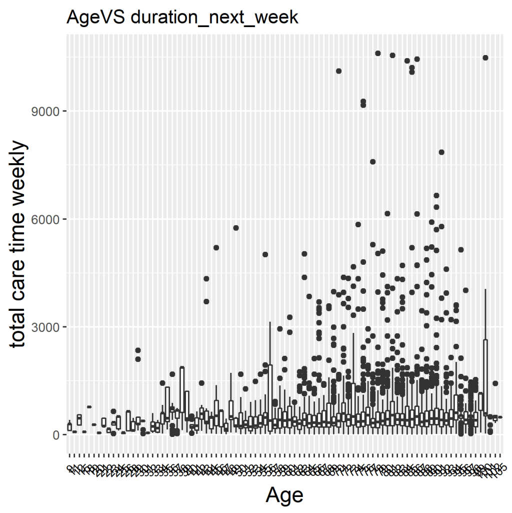


(e) nurse job title

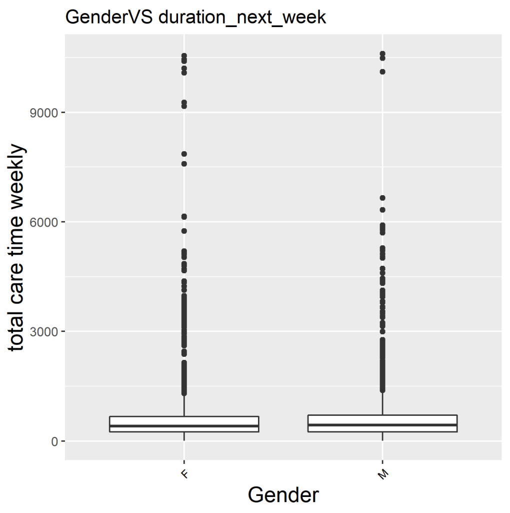


(f) nurse team

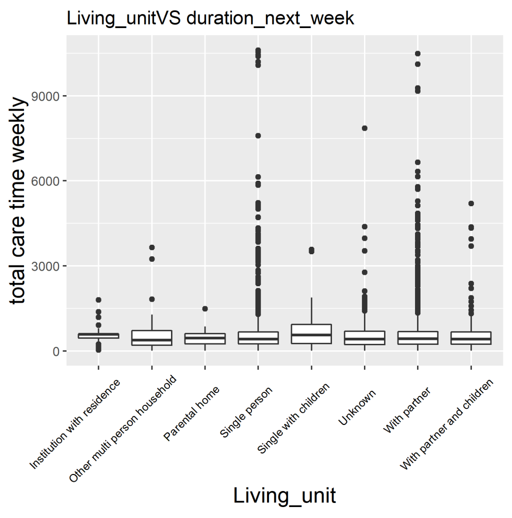
Figure 7: Scatter plots of predictors on assessment table and the response



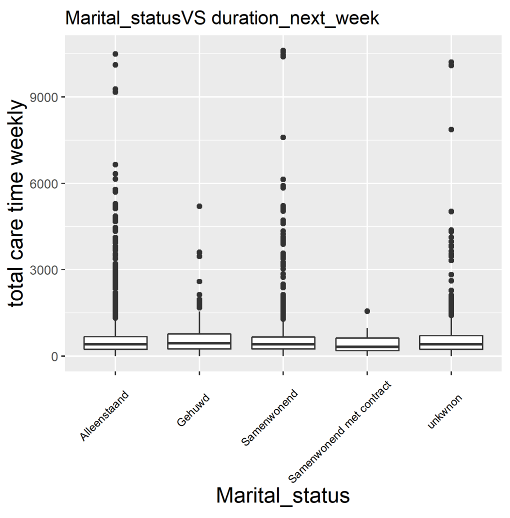
(a) patient age



(b) patient gender

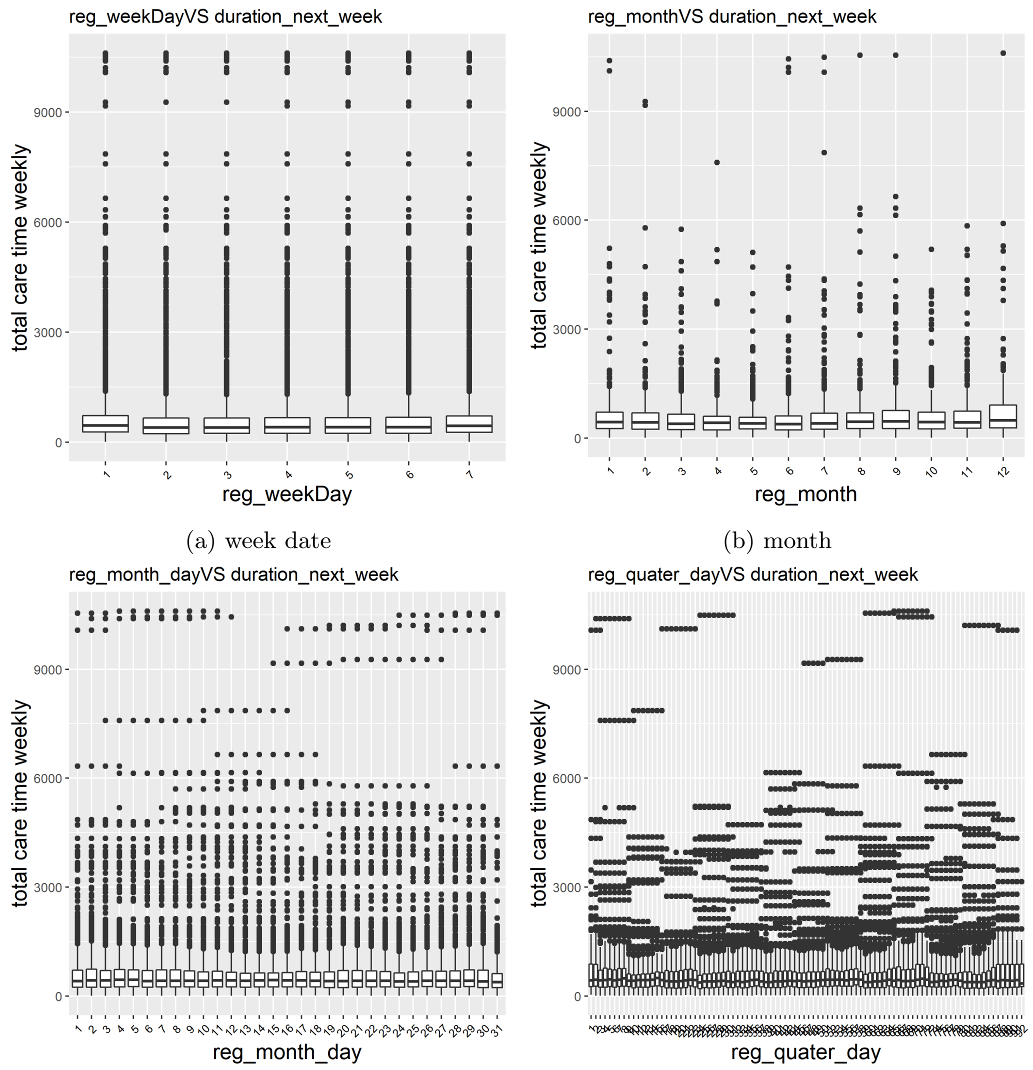


(c) patient living status



(d) patient martial status

Figure 8: Scatter plots of predictors on patient table and the response



(a) week date (b) month

(c) month date (d) quarter date

Figure 9: relationship between care end time and the label

separately, after that, data are combined and categorized into 8 groups since their frequency is quite low.

The results for classification are listed in table1 with 5 decimals. All classifiers perform bad.

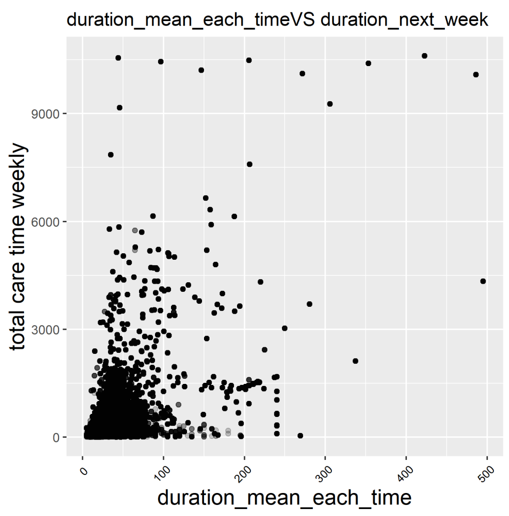
## 4.3 Regression Result

In test phrase, simple regressors are employed to test performance. The results for regression test are listed in table4.3.

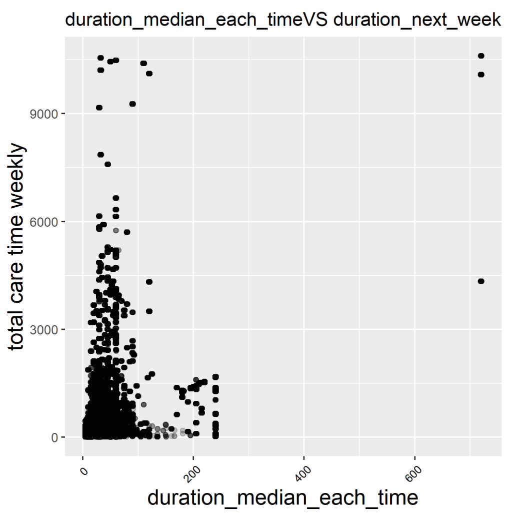
Table 2:regression performance

| regressor | RMSE | R2 | MAE |
|---|---|---|---|
| linear regression | 508.252 | 0.67969 | 228.889 |
| generalized linear regression | 504.436 | 0.684481 | 223.816 |
| decision tree regression | 444.408 | 0.755107 | 201.451 |
| random forest | 434.595 | 0.765802 | 191.337 |
| Gradient-boosted trees | 374.88 | 0.82574 | 178.186 |
| DNN(epoch=30,batch_size = 256) | 229.1842 | 0.8080 | 183.29 |

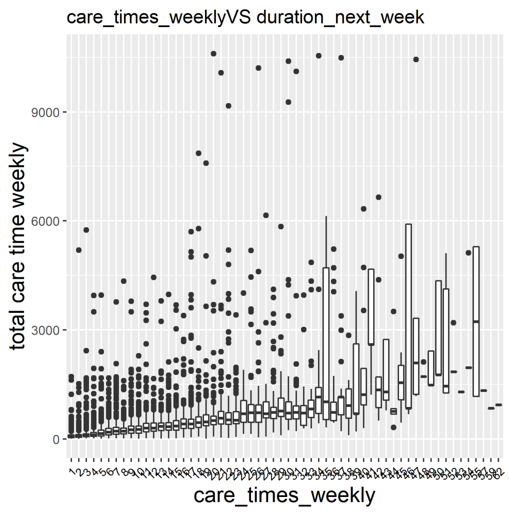
From table4.3, Gradient-boosted trees and DNN outperform the other models, so these two are explored deeper with more hyper-parameters. On facet of DNN model, epoch is tuned into 500, and batch size is 256, if mean squared error for validation set is not improved for 40 iteration, then



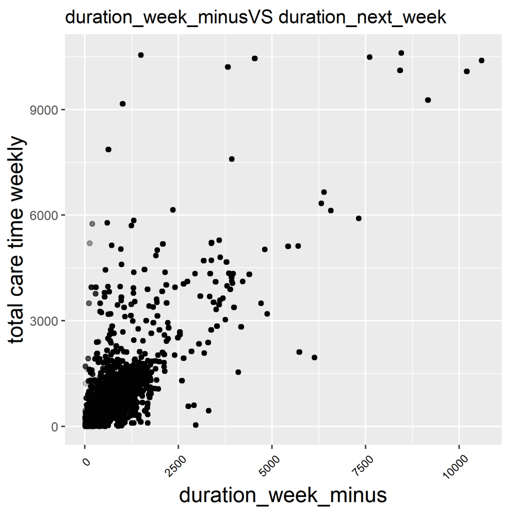
(a) mean care duration weakly



(b) median care duration weakly



(c) care amount weakly



(d) care duration weakly

Figure 10: Scatter plots of the 1-week lagged care duration and the response

training will be stopped. Adam stochastic optimization is applied with 0.001 learning rate. Also, if the performance is not imporved for the next 10 iterations, then the learning rate is reduced by 10% rate. 5-fold cross validation is employed.

Figure15, Figure14 and Figure13 show the DNN performance graphically. Figure15 shows that training error is larger than error on validation set, which is the special case for Keras(a python neural network API that applied in this experiment).

According to Keras documentation, a Keras model has two modes: training and testing. Regularization mechanisms, such as Dropout and L1/L2 weight regularization, are turned off at testing time. Besides, the training loss is the average of the losses over each batch of training data. Because the model is changing over time, the loss over the first batches of an epoch is generally higher than over the last batches. On the other hand, the testing loss for an epoch is computed using the model as it is at the end of the epoch, resulting in a lower loss[19].

Also, since these models are trained twice, and the plots depict the second training result with the pre-trained weights, so the R2 starting point is not from zero and RMSE initial point is quit small (around 35 minutes).

In Figure13, x-axes is observed data while y is predicted value, if the prediction is accurate, then this regression line should be a perfect forward diagonal line which fits prediction equals to observation equation. Figure13 shows that the results is quit good, but there are some dots that DNN failed to detect. As for these dots, one possibility is that they are outliers, which means they should be removed. But from five regression plots which correspond to different fold, these dots are

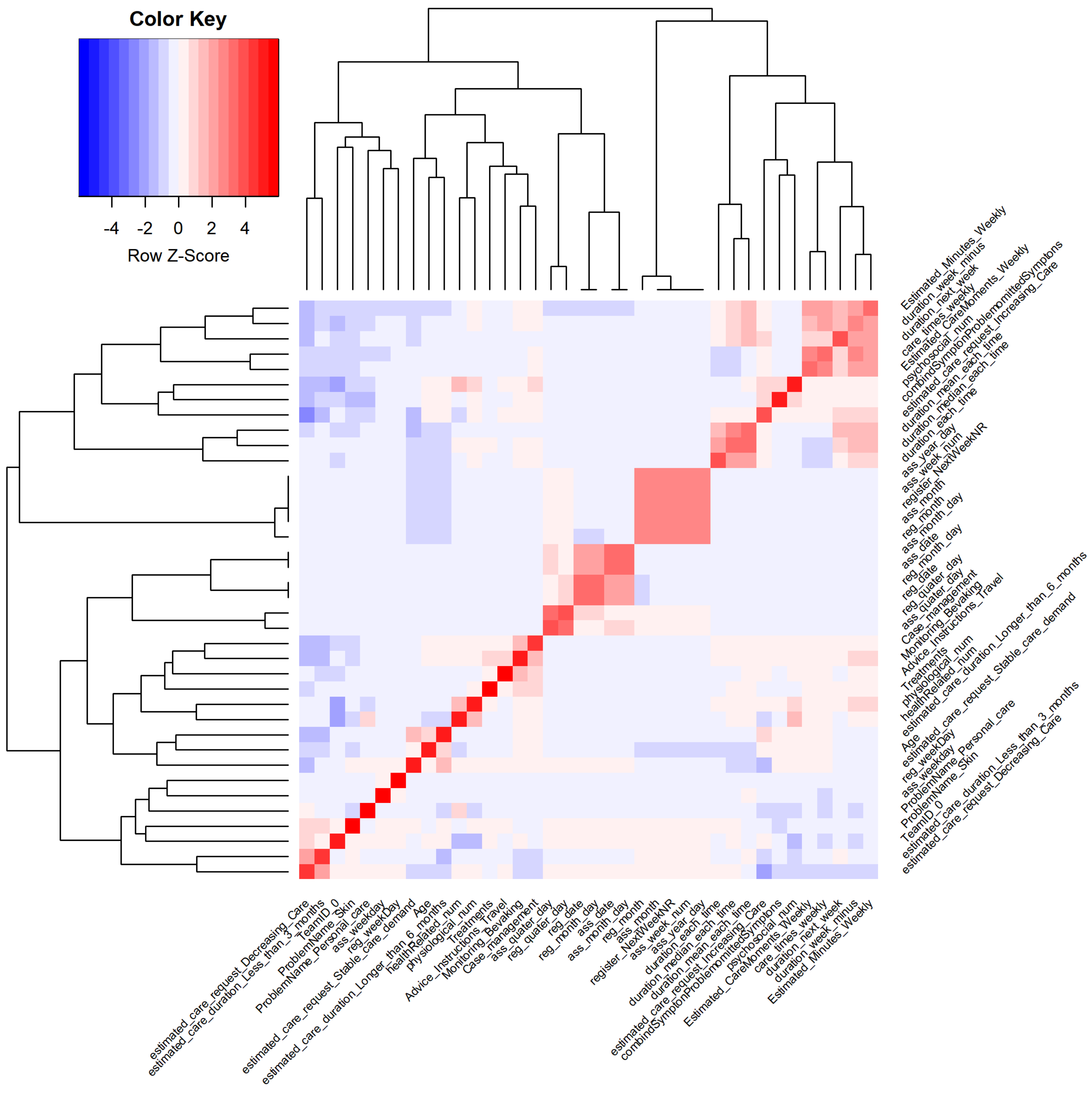


Figure 11: heatmap of correlation matrix

all different, which means that they are strongly unlikely to be abnormality. Another explanation for these dots is that the DNN learning ability is still weak and cannot explain these data since dataset is not big enough. Then, more data is needed to be collected to improve DNN learning ability.

Table4.3 shows the DNN performance on test dataset. Compared with table4.3,average MAE and RMSE is much higher than GBTs performance, but the variance of RMSE is lower, which means DNN is more robust for outliers.

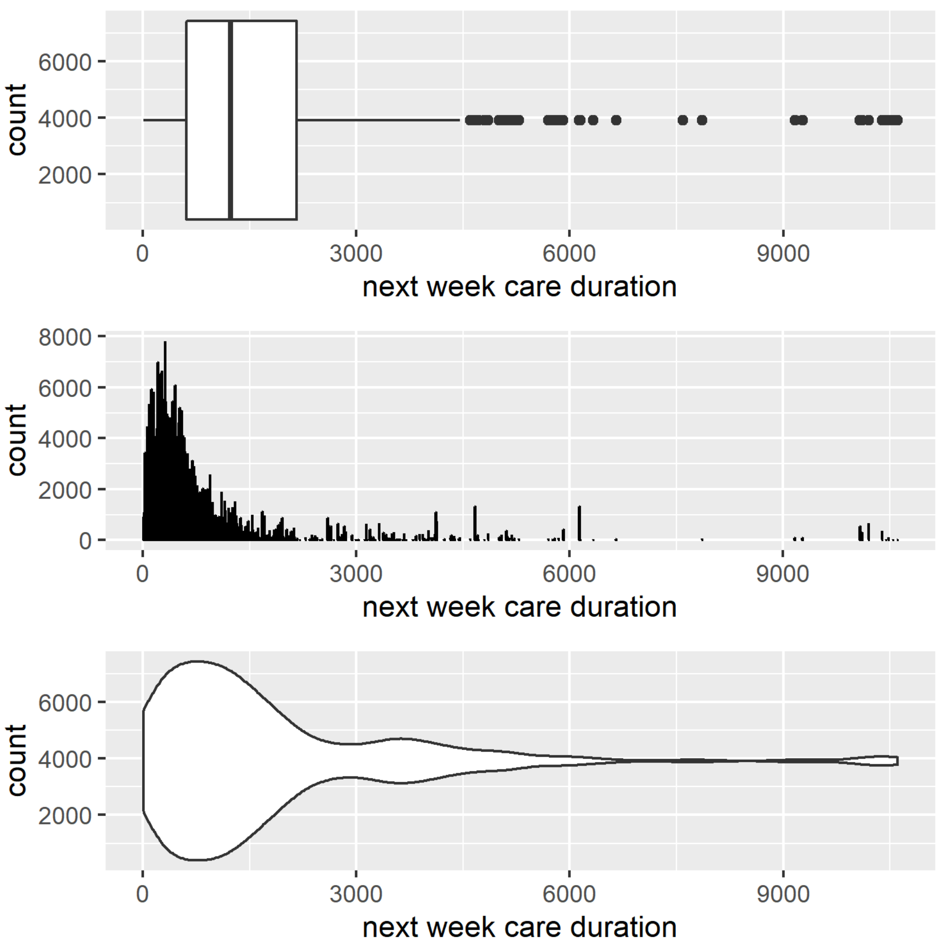


Figure 12: distribution plot of the response

Table 1: Classification Performance

| classifier | f1 | accuracy | weighted precision | weighted recall |
|---|---|---|---|---|
| logistic regression | .27657 | .28548 | .28308 | .28548 |
| decision tree | .26005 | .28496 | .26204 | .28496 |
| random forest | .29644 | .32663 | .35753 | .32663 |
| OvR | .22864 | .23880 | .24923 | .23880 |

| Table:DNN performance on test data | | | |
|---|---|---|---|
| fold | MAE | RMSE | R square |
| 1 | 5.968459 | 10.719667 | 0.999861 |
| 2 | 4.586403 | 10.350457 | 0.999870 |
| 3 | 5.627282 | 20.387276 | 0.999495 |
| 4 | 4.670589 | 9.427946 | 0.999892 |
| 5 | 6.318077 | 11.978439 | 0.999825 |
| average | 7.0359174 | 13.1125448 | 0.999788 |
| variance | 0.601453 | 19.9206 | $2.75213 \times 10^{-8}$ |

Table4.3 shows home-care duration prediction for next 1 to 15 weeks by DNN with 4-fold cross validation. Out of expectation, MAE score do not increase a lot for the latter weeks(or even decrease and get much lower MAE at the 13 week). Average RMSE and MAE are quite similar for all weeks but the RMSE variance fluctuate a lot. This model is very good for week 2,3,8,5,12,13 and 15 since they have both very low RMSE variance and MAE variance, which shows that this model is stable and accurate.

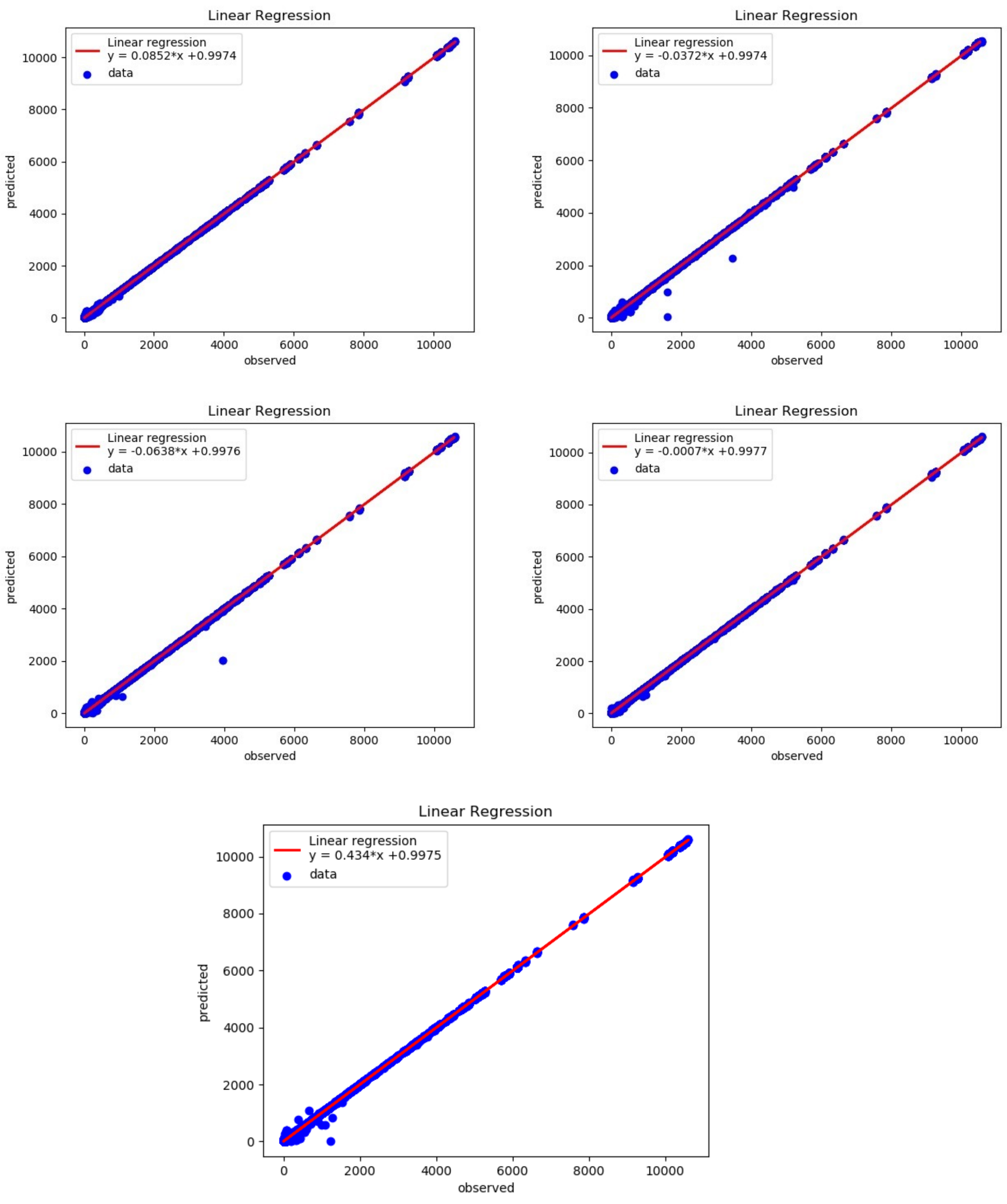


Figure 13: linear regression of prediction and observation from DNN testing results

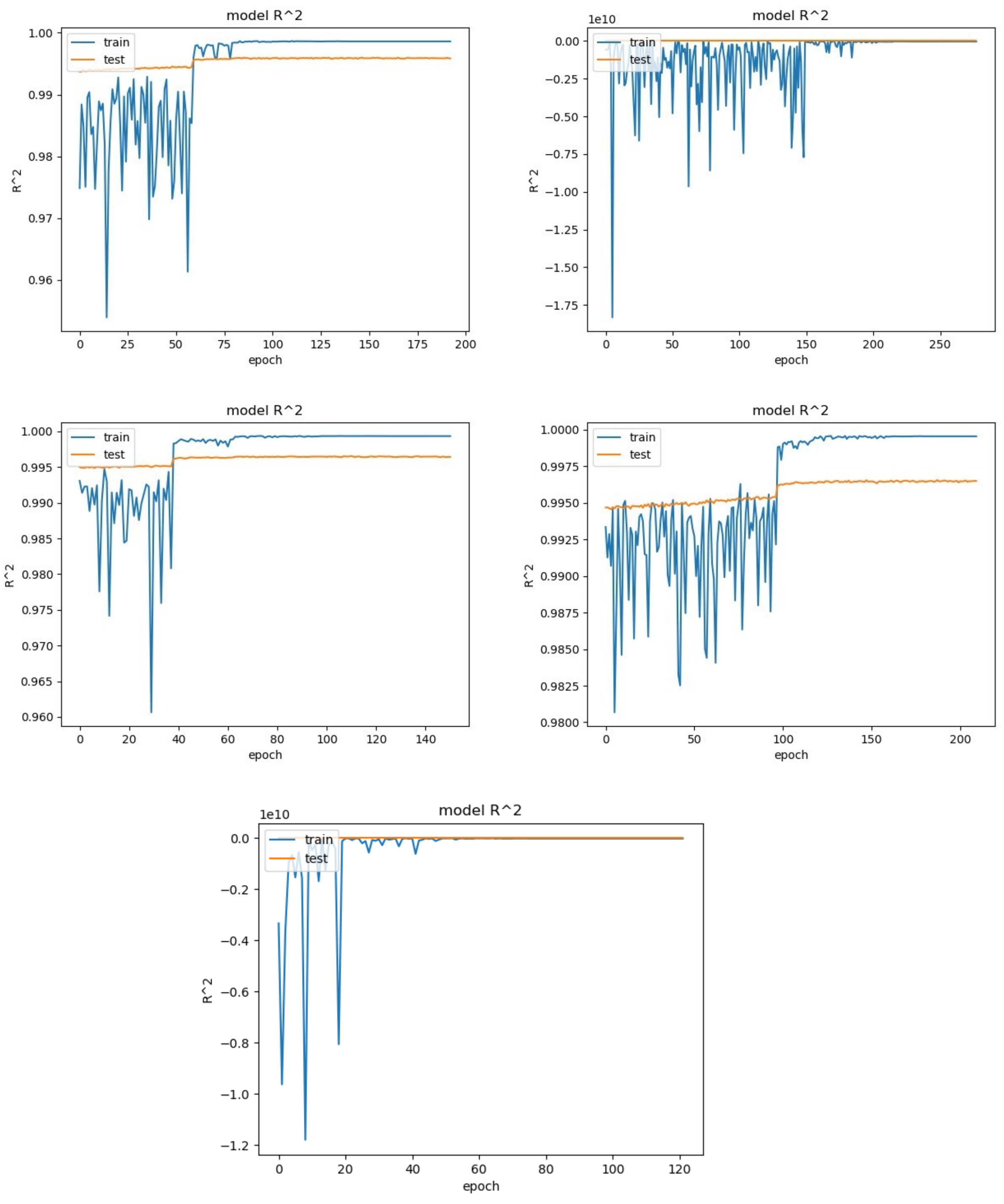


Figure 14: coefficient of determination($R^2$) for each fold of DNN testing results

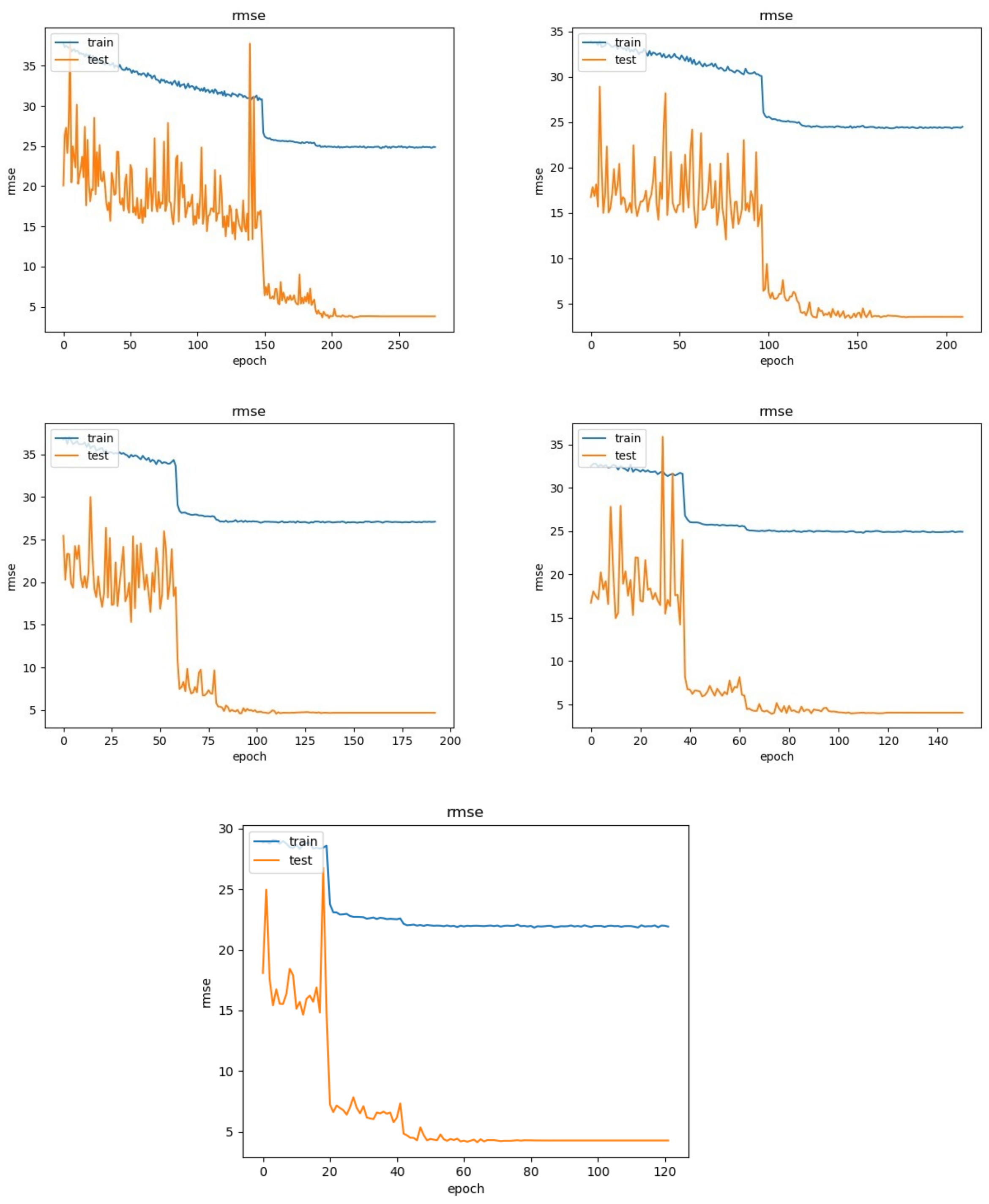


Figure 15: rooted mean squared error(RMSE) for each fold of DNN testing results

| Table:DNN performance of next 1 to 15 week care duration prediction | | | | | | | |
|---|---|---|---|---|---|---|---|
| week number | data amount | average RMSE | RMSE variance | average R square | R square variance | average MAE | MAE variance |
| 1 | 642418 | 12.7213 | 26.4136 | .99978 | $3.6143 \times 10^{-8}$ | 5.21318 | .476394 |
| 2 | 604562 | 9.65548 | .0471152 | .999883 | $3.09167 \times 10^{-11}$ | 4.73064 | .102726 |
| 3 | 569974 | 10.3735 | .987279 | .999865 | $6.96667 \times 10^{-10}$ | 4.26097 | .0352978 |
| 4 | 540212 | 16.856 | 47.6125 | .999603 | $9.85143 \times 10^{-8}$ | 6.46722 | 2.82405 |
| 5 | 511270 | 14.1787 | 5.7967 | .999749 | $7.78892 \times 10^{-9}$ | 5.20881 | .968344 |
| 6 | 485443 | 17.2281 | 33.0063 | .999604 | $6.30543 \times 10^{-8}$ | 6.08039 | .661169 |
| 7 | 462607 | 14.7498 | 21.5439 | .999706 | $3.97229 \times 10^{-8}$ | 5.30805 | 1.31465 |
| 8 | 440348 | 13.2502 | .566764 | .999833 | $3.51583 \times 10^{-10}$ | 5.42724 | .689974 |
| 9 | 419303 | 21.4657 | 113.706 | .999452 | $2.88956 \times 10^{-7}$ | 5.96737 | .442504 |
| 10 | 399372 | 16.6311 | 62.4267 | .999669 | $1.09165 \times 10^{-7}$ | 5.8688 | .304841 |
| 11 | 382167 | 15.6148 | 47.5432 | .999681 | $8.40109 \times 10^{-8}$ | 6.06584 | .910262 |
| 12 | 362664 | 13.5017 | 2.56485 | .99972 | $4.49492 \times 10^{-9}$ | 5.92112 | .552264 |
| 13 | 341653 | 12.3946 | 1.64213 | .99971 | $3.96533 \times 10^{-9}$ | 4.84936 | .0871184 |
| 14 | 326118 | 14.296 | 34.0457 | .999545 | $1.22431 \times 10^{-7}$ | 5.02931 | 1.19153 |
| 15 | 313122 | 14.0894 | 7.79763 | .999589 | $2.2394 \times 10^{-8}$ | 4.7003 | .189861 |

For gradient-boosted trees, also, 5-fold cross validation is employed.And grid search is applied in each fold. The maximum depth of tree are set to 20 and 30, which means that there are maximum 20 or 30 nodes for one simple tree. Number of iterations of boosting is 20 and 30, that is there are 20 or 30 trees built. Learning rate is 0.1 and log loss function is applied.

| Table:GBTs performance on test data | | | |
|---|---|---|---|
| fold | MAE | RMSE | R square |
| 1 | 0.1402 | 3.7108 | 0.9999 |
| 2 | 0.2045 | 15.8865 | 0.9996 |
| 3 | 0.1751 | 6.1884 | 0.9999 |
| 4 | 0.2535 | 15.2234 | 0.9997 |
| 5 | 0.2918 | 4.1836 | 0.9999 |
| average | 0.2130 | 9.0385 | 0.9998 |
| variance | 0.0036 | 36.3063 | $2.1328 \times 10^{-8}$ |

For the test data, the tree model gets result listed on table4.3 with 4 decimals remained. Also, Figure16 shows that the linear regression plot for the observation and prediction of GBTs testing results. Following above-mentioned theory, if plot fits forward-diagonal line, then the model is perfect. Figure16 shows that GBTs has a very strong learning ability, but there are some extreme outliers which shows prediction is far different from the corresponding observation. This phenomenon can also be detected by that RMSE is much more higher than MAE in GBTs performance table4.3 since RMSE is inclined to be influenced by some extremely large errors(error here refers to $|prediction - obervation|$) and MAE is more about showing the all error variance. RMSE variance in table4.3 and table4.3 shows that GBTs is more sensitive to outliers with respect to DNN.

The same GBTs model are applied to predict the next 1 to 15 weeks home-care duration. The result is listed on table4.3 with 4 decimals. Although MAE is much smaller with respect to DNN model( table4.3, RMSE variance fluctuate a lot and is a bit higher than the data in table4.3, such as for week 8 and week 12 and 13, GBTs got 120.3078,103.1694 and 161.4723 RMSE variance.

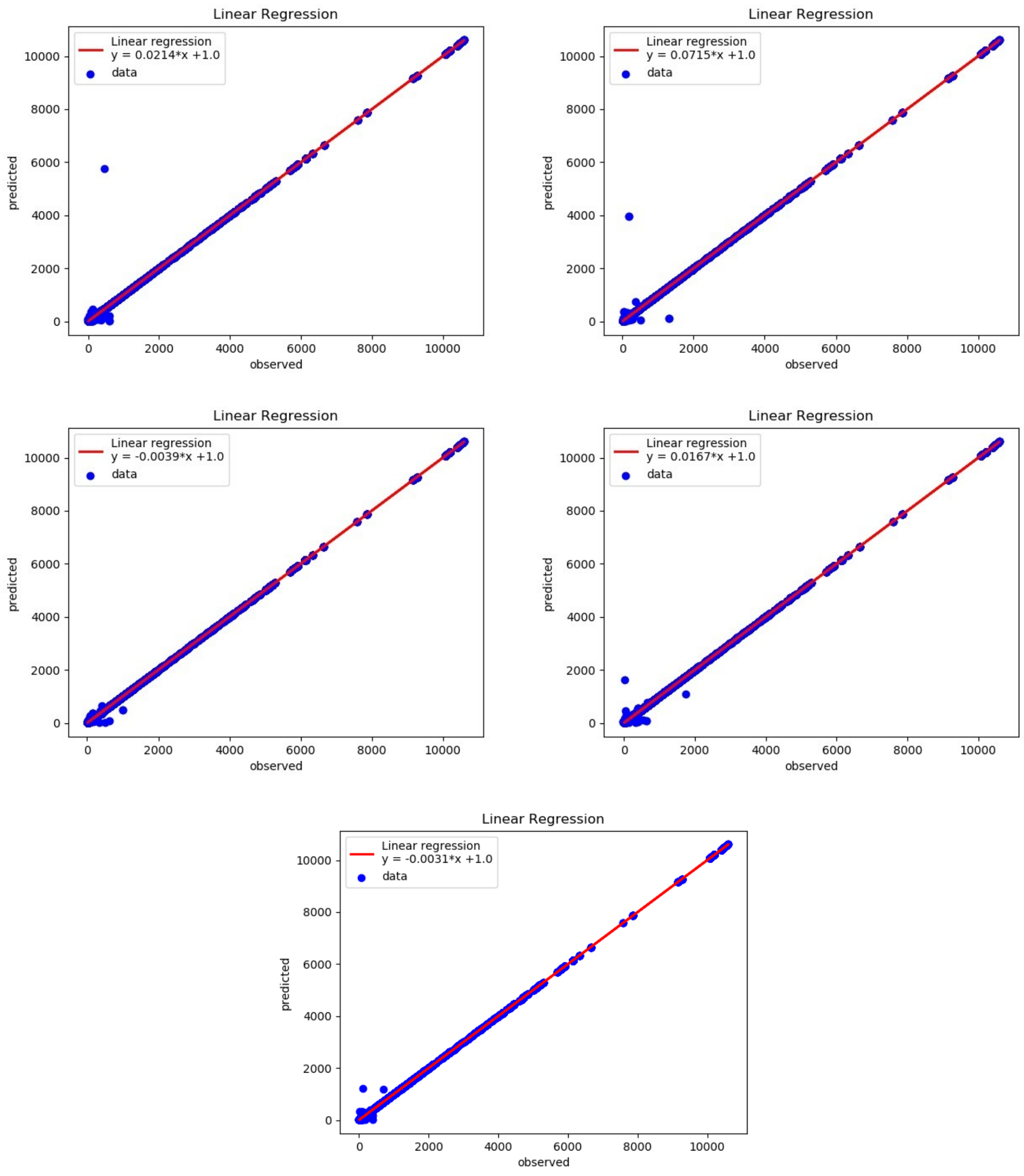


Figure 16: Linear regression of prediction and observation from GBTs testing results

| Table:GBTs performance of next 1 to 15 week care duration prediction | | | | | | |
|---|---|---|---|---|---|---|
| week number | average RMSE | RMSE variance | average R square | R square variance | average MAE | MAE variance |
| 1 | 9.0385 | 36.3063 | .9998 | $2.1328\times10^{-8}$ | .2130 | .0036 |
| 2 | 7.5515 | 17.3892 | .9999 | $6.864\times10^{-9}$ | .2075 | .0040 |
| 3 | 6.9168 | 5.2497 | .9999 | $1.7437\times10^{-9}$ | .2214 | .0003 |
| 4 | 12.0482 | 22.6318 | .9997 | $1.8419\times10^{-8}$ | .2176 | .0015 |
| 5 | 13.0081 | 58.3776 | .9997 | $7.0021\times10^{-8}$ | .2693 | .0032 |
| 6 | 18.6388 | 64.3931 | .9995 | $1.1581\times10^{-7}$ | .2643 | .0072 |
| 7 | 13.8999 | 79.1741 | .9996 | $1.4819\times10^{-7}$ | .2518 | .0024 |
| 8 | 21.6120 | 120.3078 | .9994 | $1.7869\times10^{-7}$ | .3622 | .01178 |
| 9 | 7.2598 | 13.6372 | .9999 | $4.5695\times10^{-9}$ | .19382 | .0002 |
| 10 | 10.0793 | 22.3286 | .9998 | $1.162655949\times 10^{-8}$ | .2406 | .0037 |
| 11 | 15.4323 | 68.6921 | .9996 | $1.24517047207\times 10^{-7}$ | .2329 | .0035 |
| 12 | 12.9894 | 103.1694 | .9996 | $2.72967448532\times 10^{-7}$ | .1983 | .0049 |
| 13 | 13.1867 | 161.4723 | .99947 | $6.1198\times10^{-7}$ | .21085 | .0121 |

## 5 Discussion and Conclusions

For classification problem, the quantitative response data needed to be changed into qualitative type, but it's intractable to find the proper cut-off points. There are a number of problematic issues with turning continuous data categorical. First, it is extremely unlikely that the underlying trend is consistent with the new model. Secondly, when a real trend exists, discretizing the data is most likely making it harder for the model to do an effective job since all of the nuance in the data has been removed. Third, there is probably no objective rationale for a specific cut-point. Fourth, when there is no relationship between the outcome and the predictor, there is a substantial increase in the probability that an erroneous trend will be "discovered"[14].

As for regression, gradient-boosted trees is a boosting algorithm that combined multiple week learners to create a strong one and DNN has a strong learning ability since it has multiple layer with non-linear activation. Also, Figure4.2 shows that the dependent variable distribution has high variance, which is quite suitable for high variance models, which are models that use individual data points to define their parameters, such as GBTs and neural network. Also, tree-based modeling techniques, unlike K-nearest neighbors and support vector machines, are immune to predictor distributions that are skewed or to individual samples that have unusual values (i.e.outliers)[14]. So these two regressors perform well. But the downside is that it's very time-consuming to train these model with lots of hyper-parameters. And tuning the hyper-parameter is a delicate work, such as the iteration for early stopping, if it is too small then DNN might still stuck in the local minimum of loss function, while if it is too big then it will costs lost of time to do unnecessary training.

There are some improvements for this experiment. As for DNN model, according to Max Kuhn et al.[14], correlation will deteriorate DNN performance, but the DNN results for removing predictors with more than .99 correlation score doesn't improve a lot. Maybe some other correlation threshold can be applied to test to get the better performance. Also, as mentioned above, there are some outliers in regression plots(figure16 and figure13). There are some uncertainties about why these dots are unfiited by forward-diagonal regression line. They could be abnormalities, which means they should be removed, or they could be correct data but the dataset is not big enough so that the model learning ability is weak. But nothing has been done to test those two hypotheses.

Distribution of the response (figure4.2) shows that data has a long tail, also, regression plots (figure13 and figure16) indicate that there is very small amount of data(blue dot) in the latter phase of the regression line(read line). So, the predicting ability for the long duration may be fragile since the data is not big enough. Beside, all outliers in figure13 and figure16 are located in the short home-care duration time-frame (before 2000 minutes in x-axes more specifically), which means there are some noise in these time-frame data. These two phenomena indicate that more data should be collected to make the model more robust.

For the final application, GBTs and DNNs should be both applied. And the two results are compared, if they are quite similar, then prediction from GBTs is applied as final result. But if these two predictions vary a bit, then the result from DNN is taken as final care-duration.

## References

[1] wikipedia. linear regression.

[2] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 67(2):301–320, 2005.

[3] Spark. Classification and regression.

[4] Spark. Ensembles - rdd-based api.

[5] Christos Stergiou and Dimitrios Siganos. Neural networks.

[6] Jeff Heaton. *Introduction to Neural Networks for Java, 2Nd Edition.* Heaton Research, Inc., 2nd edition, 2008.

[7] Howard B. Demuth, Mark H. Beale, Orlando De Jess, and Martin T. Hagan. *Neural Network Design.* Martin Hagan, USA, 2nd edition, 2014.

[8] Anish Singh Walia. Activation functions and it's types-which is better, 2017.

[9] Geoffrey E. Hinton Alex Krizhevsky, Ilya Sutskever. Imagenet classification with deep convolutional neural networks. 2012.

[10] Bill Ancalagon the black. What are the advantages of relu over sigmoid function in deep neural networks?, 2018.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[12] wikipedia. Feature scaling.

[13] Andrew NG. machine learning.

[14] Max Kuhn and Kjell Johnson. *Feature Engineering and Selection A Practical Approach for Predictive Models*. Chapman Hall CRC, USA, 1nd edition, 2018.

[15] Xianjun Dong. data transformation:variance-stabilizing transformation, 2012.

[16] wikipedia. dummy variable(statistics).

[17] M. Kuhn and K Johnson. *Applied Predictive Modeling*. Springer, 2013.

[18] Thiago G. Martins. Near-zero variance predictors. should we remove them?

[19] keras. Keras faq: Frequently asked keras questions, 2018.

## A Detailed Steps

The detailed steps about how to handle data are listed below.

- Preparing data
  - Removing NA:
    Since there are only a small amount of NA data, deleting them doesn't influence the classification performance.
  - Remove abnormality.
    From patient table:
    - Remove age more than 110
    - Remove date_start_of_care below than the year 2007, since Buurtzorg company started at 2007
    - Remove data that column date_end_of_care earlier than date_start_of_care

    From assessment table:
    - Remove estimated_care_minutes_weakly more than 10080 mins( 7day*24h*60mins)
    - Remove duplications

    From registration table:
    - Remove care duration less than 5 mins data (registration_end_time - registration_start_time)
  - Select useful columns and combine some columns
    From registration table:
    - patientID:
      This is used for combine data.
    - Financer:
      This is insurance company.
    - weekNR:
      This is the week number in the year.
    - JobTitle:
      This is used to identify different nurse type.

* TeamID:
  This is treated as categorical data, since different team has unique ID. And team is distributed locally, this predictor can offer some location information.
* Starttime: This is care start time, formate is alike "2017-09-08 09:23:12".
* Einddtime: This is care end time.

From assessment table:

* patientID:
  This will be used to combine data afterwards
* estimated_care_duration:
  This is categorical data done by nurse, possible value are : 3 to 6 months, Less than 3 months,Longer than 6 months and unknown
* estimated_care_request:
  This is categorical data done by nurse, possible value are: Decreasing Care,Increasing Care, Stable care demand and unknown.
* Estimated_CareMoments_weakly:
  This is how many care the patient will take in one week, this data is estaimted by nurse.
* Estimated_Minutes_weakly:
  This is numerical data estimated by nurse after finishing each care visiting. The unit is minute.
* Advice_Instructions_Travel
* Treatments
* case_management
* Monitoring_Bevaking
* problem_environment_domain_num (sum of values in columns: income, sanitation, resodence, neighborhood/workplace safety)
* problem_psychosocial_domain_num (sum of values in columns: communication with community resources, social contact, role change, interpersonal relationship, spirituality, grief, mental health, caretaking/parenting, neglect, abuse, growth and developemnet)
* problem_physoicalical_domaintextunderscore num ( sum of values in columns : Hearing, Vision, Speech and language, Oral health, Cognition, Pain, Consciousness, Skin, Neuro-musculo-skeletal function, Respiration, Circulation, Digestion-hydration, Bowel function, Urinary function, Reproductive function, Pregnancy, Postpartum, Communicable/infectious condition )
* problem_health_related_behaviors_domain_num (sum of values in columns: Nutrition, Sleep and rest patterns, Physical activity, Personal, care, Substance use, Family planning, Health care supervision,Medication regimen )

From care_plan table:

* patientID:
  This is used to combine data
* problemID: this is used to combine data in problem table
* signAndsymptomID this is used to combine data in signAndsymptom table
* care_plan_end_time

this is useful since each care should happen before this predictor time From signAndsympton_lookup_table

* signAndsymptonID
* problemID
* signAndSympton_name

From problem_lookup_table

* problemID
* problem_name

From patient table

    * PatientID
    * age
    * gender
    * martial_status:
      This is categorical data including possible values: Single,Married, Living together, Living together with contract and unknown.
    * living_unit:
      THis is categorical data including possible values: Institution with residence,Other multi person household ,Parental home,Single person,Single with children,Unknown,With partner,With partner and children.
  - Feature engineering.
    For registration table:
    * timeDiff: the duration for each care. timeDiff = endTime - startTime
    * Mean duration for each week : mean care duration weakly for each patient. This is calculated by grouping by weekNR and PatientID column
    * Median duration for each week: median care duration weakly for each patient.
    * Total duration for each week: sum care duration weakly for each patient.
    * reg_weekDay: care are happend in which day on one week, 0 refers to sunday
    * reg_month: care happened on which month
    * reg_month_day:care happnend on which day in a month
    * reg_quater_day: care happended on which quarter

    For assessment table:
    * ass_month: assement made on which month
    * ass_weekday: assement made on which day in one week ( 0 corresponding to Sunday)
    * ass_month_day: assessment made on which day in month
    * ass_quarter_day: assessment made on which quarter
  - Combine data:
    Combined by PatientID, care_end_time in care_plan table must longer than register WeekNR. assessment_date_week_num must be the same with weekNR in registration table. Create a new column named next_week_duration which week_num is one more than weekNR in registratin table. Steps are listed below:
    1. Copy and rename registration table into next_registration, delete columns except 3 columns: PatientID, total_duration_for_each_week and weekNR
    2. Rename column in next_registration from total_duration_for_each_week into duration_for_next_week
    3. Rename column in next_registration from weekNR into next_weekNR
    4. Combined next_registration and registration table by PatientID and condition: next_weekNR = weekNR +1
    5. Categorize duration_for_next_week into hours and calculate frequency for each hour categorization, combined less frequent categorizes into one. ( in this experiment, we get 177 categories corresponding to 177 hours, and we categorize hours after 15 into 5 groups since they have low frequency)

- Preprocessing data
  - Change categorical data into one-hot-encoding
  - scale, center and remove Zero- and Near Zero-Variance Predictors,Yeo-Johnson transformation.
  - Feature selection (based on information gain, gain ratio and symmetrical uncertainty)
- Use simple classifiers to test performance
  Randomly split data into training data and test data with .7 and .3 percentage. Then use some simple classifiers to test data. For classification:

- Logistic regression : L2 penalty is applied, the regularization parameter is 0, and max iteration is 50
- Decision tree: Gini is applied for information gain calculation, max depth is 5 (means 5 internal node )
- Random forest: Gini is applied, max depth is 4, number of features to consider for splits at each node is sqrt of all features.
- Multilayer perceptron classifier: a simple version of Neural Network with 3 layers, input (feature) layer, middle layer (set 30 here) and output (classes) layer(20 classes here). Nodes in intermediate layers use sigmoid function and nodes in the output layer use softmax function. The logistic loss function for optimization and L-BFGS as an optimization routine is applied here.
- One-vs-rest classifier: it cares a binary classification problem for each of the k classes.

For regression:

- linear regression:
  loss function is squared error, corresponding to formula2.1, $\alpha$ is 0.3, $\lambda$ is 0.8.
- Generalized linear regression
- regression decision tree
- gradient boosted tree
- neural network: neural network structure is shown in figure3. Since data are not linearly separable, which means more than one layer need to be applied. According to formula shown in 2.7.2, there are around 600,000 data, so the first layer is 1000 neurons. And output layer is one neuron with Relu activation function, because prediction cannot be negative, so Relu is applicable.

• choose the best two models to train

  - DNN: 5 cross-validation is applied. Figure3 shows the neural network result. Adam optmizer is applied. Loss function is mean squared error. Iteration for early stopping is 40, that is, if mean squared error does not decrease in next 40 iterations, then training will be stopped. Also, learning rate is reduced by 10% if the performance does not improved in 10 iterations. Initial learning rate is $10^{-4}$.
  - GBTs:.And grid search is applied in each fold. The maximum depth of tree are set to 20 and 30, which means that there are maximum 20 or 30 nodes for one simple tree. Number of iterations of boosting is 20 and 30, that is there are 20 or 30 trees built. Learning rate is 0.1 and log loss function is applied.

## B Code

Github address for code is: `https://github.com/RachelChen3/ecare`