11-5-2015

Ensemble Learning Method on Machine Maintenance Data

Xiaochuang Zhao
University of South Florida, zxc910108@gmail.com

Ensemble Learning Method on Machine Maintenance Data

by

Xiaochuang Zhao

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Arts
Department of Mathematics & Statistics
College Arts and Sciences
University of South Florida

Major Professor: Dan Shen, Ph.D.
Christos Tsokos, Ph.D.
Lu Lu, Ph.D.

Date of Approval:
November 4, 2015

Keywords: Machine Learning, Support Vector Machine, Ensemble, Penalized Logistic Regression, Predictive Maintenance,
Binary Classification

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

In the industry, a lot of companies are facing the explosion of big data. With this much information stored, companies want to make sense of the data and use it to help them for better decision making, especially for future prediction. A lot of money can be saved and huge revenue can be generated with the power of big data. When building statistical learning models for prediction, companies in the industry are aiming to build models with efficiency and high accuracy. After the learning models have been developed for production, new data will be generated. With the updated data, the models have to be updated as well. Due to this nature, the model performs best today doesn't mean it will necessarily perform the same tomorrow. Thus, it is very hard to decide which algorithm should be used to build the learning model. This paper introduces a new method that ensembles the information generated by two different **classification** statistical learning algorithms together as inputs for another learning model to increase the final prediction power.

The dataset used in this paper is NASA's Turbofan Engine Degradation data. There are 49 numeric features (X) and the response Y is binary with 0 indicating the engine is working properly and 1 indicating engine failure. The model's purpose is to predict whether the engine is going to pass or fail. The dataset is divided in training set and testing set. First, training set is used twice to build **support vector machine (SVM)** and **neural network** models. Second, it used the trained SVM and neural network model taking X of the training set as input to predict Y1 and Y2. Then, it takes Y1 and Y2 as inputs to build the **Penalized Logistic Regression** model, which is the ensemble model here. Finally, use the testing set follow the same steps to get

the final prediction result. The model accuracy is calculated using overall classification accuracy. The result shows that the ensemble model has 92% accuracy. The prediction accuracies of SVM, neural network and ensemble models are compared to prove that the ensemble model successfully captured the power of the two individual learning model.

CHAPTER 1: INTRODUCTION

With the explosion of "Big Data", statistical learning methods having been rapidly developed in the last decade. Especially in the industry, more and more companies are trying to utilize the power of statistical learning to not only gain insights from their past data sets, but more importantly to make predictions of future. There are two category of learning method; one is supervised learning and another is un-supervised learning. The difference between these two is that supervised learning models are built to predict the target Y, which is predetermined; on the other hand, un-supervised learning models don't have a target variable but are aiming at finding patterns from the existing data. Majorities of the predictive analysis are focused on supervised learning methods. In supervised learning, there are two types of problems, regression and classification. This paper is focusing on the two-class classification problem.

Two-class classification is one of the most common problems nowadays in the research field and also in the industrial world. In the industrial world, when it comes to building machine learning solutions, efficiency is one of the primary concern. It often takes 1-2 month just to fine tune the machine learning model in order to increase the accuracy, sometimes to only achieve as little as 3% improvement. So, rapid development is more important than model accuracy in most of the cases. With the help of statistical modeling software, it is fairly easy to build machine learning models without fine tuning. Depending on the data, different machine learning methods may give similar or different accuracy. However, after the models have been put in real-time production for a couple of month, when new data come in, the models' performance may vary.

Because of this nature, is it extremely difficult to choose one machine learning method for the particular problem, since the best method today might not be the best tomorrow. So, is there a way to combine the power of different two-classification machine learning methods together to give the best prediction? This question is the main focus of this paper. And one of the solution to this question is discussed in this paper.

In the industrial world, predictive maintenance is one of the most highly demanded use cases of statistical learning. In this paper, the predictive maintenance use case dataset is Turbofan Engine Degradation Data Set by NASA [1]. In this dataset the input variables are numeric variables of machine log information, the response is a categorical variable with pass/fail two classes. The goal of this paper is not to find the best model for this dataset, but using this popular use case to compare the model accuracy of traditional 2 single machine learning models with the model accuracy of the methodology of combining the two machine learning models together.

The two machine learning algorithms used here are Support Vector Machine and Neural Network. These are the two commonly used statistical learning algorithms today. The two classification models are each trained on the training data set, and prediction accuracy is accessed using prediction accuracy and sensitivity value, which is calculated separately for two models by predicting using the testing set. In order to ensemble the power of the two classification results, a penalized logistic regression is trained using the training set prediction outputs of the two classification models as two X's inputs. And the penalized logistic regression prediction is done by taking the testing data set prediction results of the two models as X's inputs. The earliest ensemble method is introduced by Dasarathy and Sheela's 1979 paper [2], which is an ensemble system using two or more classifiers to partition the feature space. Here, in this paper, the ensemble system is built using the penalized logistic regression to take the

information generated by Support Vector Machine, Neural Network as in puts in order to achieve a better result. Using the predictive maintenance dataset from NASA, the prediction power of the ensemble method is compared with prediction results of the single learning models.

CHAPTER 2: METHODOLOGY

2.1 Statistical Learning Algorithm Formulas

In this paper, the three statistical learning algorithms used are Support Vector Machine, Neural Network and Penalized Logistic Regression. This section is dedicated to introduce the math formulas for these algorithms, which are the foundation for the next section.

2.1.1 Neural Network

In machine learning, Neural Network is one of the most powerful and widespread learning methods in the artificial intelligence field. The inspiration of Neural Network is based on the biological neural network of the human brain. In human brain, the neurons are interconnected as a web to exchange electrical signals from each other. Input signals are received by dendrites, and pass on via an elongated axon, based on those inputs, an output signal is generated and passed onto other neurons.

The first computational neural network was developed in 1943 by a neuroscientist, Warren S. McCullock and a logician, Walter Pitts [3]. Taken the inspiration from the brain, they developed a mathematic neural network model called threshold logic. Followed by many fellow scientists, artificial neural network was developed over the spam of a few decades based on mathematics with the intention to help solve real life problems.

A Neural Network is a two-stage regression or classification model, which can be illustrated by a network diagram illustrated in Figure 1. The algorithm uses K-class classification, and hence there are k Y's at the bottom of the graph as output. In this paper,

however, the model was built to solve a two-class classification problem, as a result there are only two Y response variables in our intended model. The units in the middle of the network, Z's, are called hidden units. There might be more than one-layer of hidden units. Here, the more general case, one-layer hidden units are discussed. The derived features Z_m are linear combination, or linear multilogit [4] of the inputs X, which Z' are then used as inputs for modeling response Y's.

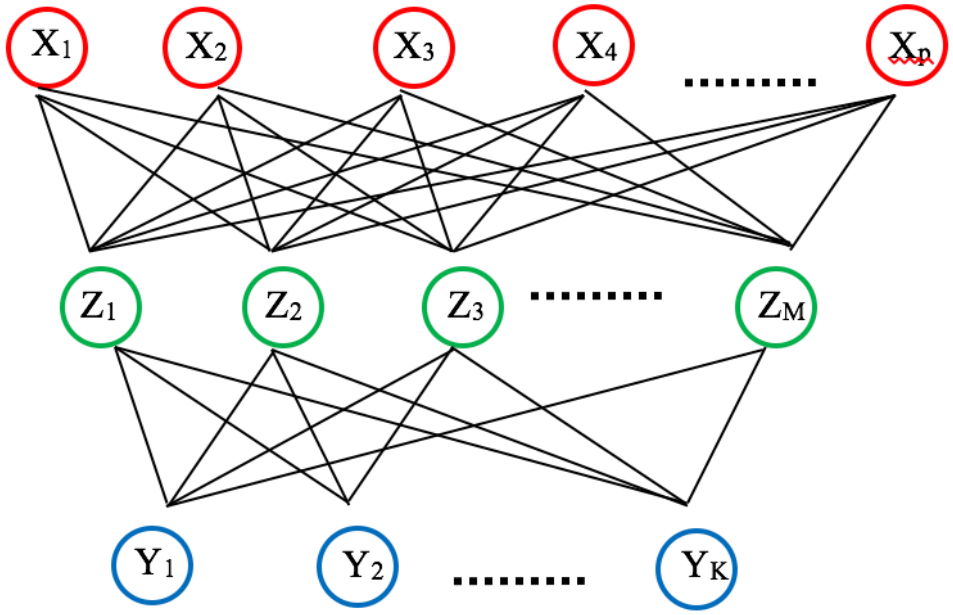

Figure 1. Single Layer Neural Network System Graph

For the most general case, Z is written as linear combinations of X. The math formula representing such a neural network is as follow:

$$\begin{aligned} Z_m &= \sigma(\alpha_{0m} + \alpha_m^T X),\ m = 1, \dots, K \\ T_k &= \beta_{0k} + \beta_k^T Z, k = 1, \dots, K \\ f_k(X) &= g_k(T), k = 1, \dots, K \end{aligned} \tag{2.1}$$

Where $Z = (Z_1, Z_2, \dots, Z_M)$, and $T_k = (T_1, T_2, \dots, T_K)$.

In the most common cases, $\sigma(v)$, the activation function is selected to be the *sigmoid* $\sigma(v) = 1/(1+e^{-v})$ [4]. In equations (2.1), the T on top of $a_m^T X$ and $\beta_k^T Z$ are stands for transpose. $a_m^T X$ is the projection of X onto the unit vector α_m, the same for $\beta_k^T Z$. For K-class classification problem, $g_k(T)$, the final transformation of the vector outputs T, is mostly in favor of the *softmax* function[4]

$$g_k(T) = \frac{e^{T_k}}{\sum_{\ell=1}^{K} e^{T_\ell}} \tag{2.2}$$

The other unknown parameters, α_{0m}, α_m^T, β_{0k} and β_k^T are called weights. It is important to choose the initial weights. Weight values, related to individual notes are also called biases. Weight values are chosen by the iterative flow of training data through the network.

Let θ denotes the full set of weights. For k-classification the errors are represented by:

$$R(\theta) = -\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik} log f_k(x_i) \tag{2.3}$$

2.1.2 Support Vector Machine (SVM)

Support Vector Machine is a method used for solving classification problem. It was first developed in 1990s in computer science and has been gaining popularity ever since. Primarily, it is designed to solve two-class classification problem. But now its extensions can solve regression problem and multi-class classification problem.

What distinguish support vector machine from other classification algorithms is the concept of hyperplane. SVM built hyperplane, or multiple hypoerplanes in a high or infinite dimensional space. What needs to be achieved is a good hyperplane that has the longest distance to the nearest training data of any class. The distance is not determined by the whole dataset of

one class, instead just by a set of support vectors. Because of such nature, outlier is not an issue when using SVM to solve either regression or classification problems. The hyperplane can be either linear form in Figure 2 or non-linear form as shown in Figure 3.

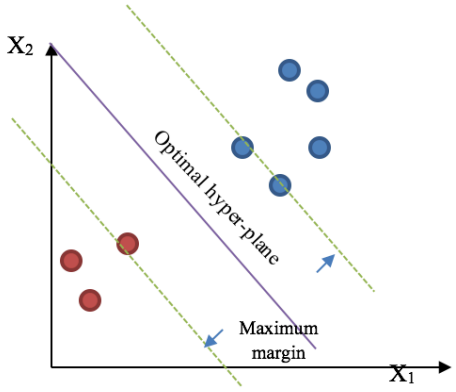

Figure 2. The Linear Maximum Margin Optimal Hyperplane Graph

There could be many hyperplanes, but which one is the best to choose? Then maximum margin comes to play. The optimal hyperplane is the separating hyperplane that has the largest margin, the furthest distance from the two different classes. The maximum margin hyperplane separates the two classes perfectly. However, this also means that the maximum margin hyperplane is extremely fitted to the training dataset, so that one addition point added can lead to drastic change in the maximum margin hyperplane. Because of such characteristics, it would be wise to choose a classifier that is based on the hyperplane which does not separate the two classes perfectly, in turn the model would be more robust. Such is called Support vector classifier. Support vector classifier allow some observations to be on the wrong side of the

margin, as well as on the wrong side of the hyperplane. The mathematic functions [5] of the support vector classifier is as follow:

$$\begin{aligned} &\underset{\beta_0,\beta_1,\ldots,\beta_p,\ \varepsilon_1,\ldots,\varepsilon_n}{\text{maximize}} \quad M \\ &\text{subject to} \quad \textstyle\sum_{i=1}^{p} \boldsymbol{\beta_j^2} = \mathbf{1}, \\ &\boldsymbol{y_i}\left(\boldsymbol{\beta_0} + \boldsymbol{\beta_1 x_{i1}} + \boldsymbol{\beta_2 x_{i2}} + \cdots + \boldsymbol{\beta_p x_{ip}}\right) \geq \boldsymbol{M}(\mathbf{1} - \boldsymbol{\epsilon_i}), \\ &\boldsymbol{\epsilon_i} \geq \mathbf{0}, \quad \textstyle\sum_{i=1}^{n} \boldsymbol{\epsilon_i} \leq \boldsymbol{C}, \end{aligned} \tag{2.4}$$

here C is a nonnegative tuning parameter. M is the width of the margin. And $\boldsymbol{\epsilon_1, \ldots, \epsilon_n}$ are the slack variable[5], which are used to permit individual observations to be on the wrong side of the hyperplane or the margin; if $\boldsymbol{\epsilon_i} = 0$ then the $\boldsymbol{i}$th observation is on the right side of the margin; if $\boldsymbol{\epsilon_i} > \mathbf{0}$ then the $\boldsymbol{i}$th observation is on the wrong side of the margin, but not on the wrong side of the hyperplane; if $\boldsymbol{\epsilon_i} > \mathbf{1}$ then the $\boldsymbol{i}$th observation is on the wrong side of the hyperplane. Solving (2.4) is an optimization problem, here details are not presented. Once (2.4) is solved, then the test dataset prediction depends on the sign of $\boldsymbol{f(x^*) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_p^*}$; if $\boldsymbol{f(x^*) > 0}$ then $\boldsymbol{y = 1}$ and if $\boldsymbol{f(x^*) < 0}$ then $\boldsymbol{y = -1}$, here y has two classes, -1 and 1.

Support vector classifier is the approach for two-class classification with linear hypserplane boundary. However, linear boundary does not cover all the practice. From the data in Figure 3, the non-linear hyperplane is a better classifier than the linear hyperplane. Support Vector Machine uses kernals to expand the feature space of support vector classifier.

The solution of solving (2.4) is in fact the inner product of the observations [5]:

$$\boldsymbol{< x_i, x_{i'} > = \textstyle\sum_{j=1}^{p} x_{ij} x_{ij'}} \tag{2.5}$$

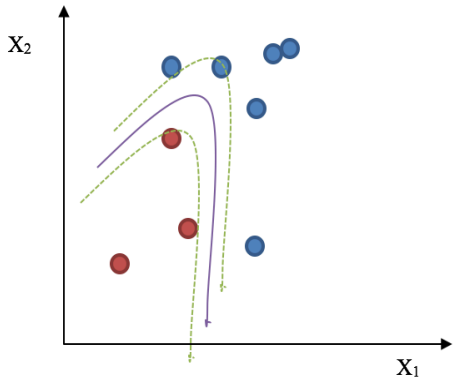

Figure 3. None-Linear SVM Hyperplane

So, the support vector classifier can be represented as:

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i < x_i, x_{i'} > , \tag{2.6}$$

where α_i are the n parameters. Solving this equation, the only thing needed is the inner product of all observations in the training set. For SVM it uses kernel, $K(x_i, x_{i'})$, instead of using the inner product. Equation (2.6) represents the linear kernel. Polynomial and radial kernels are often commonly used along with linear kernel in SVM problems.

2.1.3 Penalized Logistic Regression

Logistic Regression, developed by David Cox in 1958 [6], is a classic method for classification, which returns the probability of a class with the transforms of the linear combination of the independent variables. Nowadays with the rapid development of big data, when it comes to modeling, the biggest challenge statisticians often facing is the large set of features. A large set of features also means there is a high chance of multi-collinearity, which

logistic regression is very sensitive of. Furthermore, over fitting is also a draw back of the classic logistic regression when it comes to maximizing prediction accuracy. The solution for multi-collinearity and over-fitting problems of the logistic regression is called penalized logistic regression.

For the penalized logistic regression, there are two classic methods which are L1 absolute lasso penalty [7] and L2 quadratic ridge penalty [8]. Let $\boldsymbol{p(x_i)}$ be the probability of the combination of x of type class y=1.The logit of the classic logistic regression is:

$$\mathbf{log}\left\{\frac{\boldsymbol{p(x_i)}}{\mathbf{1}-\boldsymbol{p(x_i)}}\right\} = \boldsymbol{\beta_0} + \sum_{\boldsymbol{j=1}}^{\boldsymbol{p}} \boldsymbol{\beta_j x_j} \,, \tag{2.7}$$

the parameters in the logistic regression are best estimated by maximum likelihood. Then log-likelihood is as follow:

$$\boldsymbol{\ell(\beta)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \{\boldsymbol{y_i}\, \mathbf{log}\big(\boldsymbol{p(x_i)}\big) + (\mathbf{1}-\boldsymbol{y_i})\, \mathbf{log}\big(\mathbf{1}-\boldsymbol{p(x_i)}\big)\} \ . \tag{2.8}$$

Adding penalty to the log-likelihood gives the penalized log-likelihood:

$$\boldsymbol{\ell^*(\beta)} = \boldsymbol{\ell(\beta)} - \boldsymbol{\lambda J(\theta)} \,, \tag{2.9}$$

in (2.9) $\boldsymbol{\lambda}$ is the tuning parameter and $\boldsymbol{J(\theta)}$ is a penalty function. For the L1 lasso penalization:

$$\text{L1-penalization:} \quad \mathbf{J(\theta)} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{p}} |\boldsymbol{\beta_j}| \ , \tag{2.10}$$

for the L2 ridge penalization:

$$\text{L2-penalization:} \quad \mathbf{J(\theta)} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{p}} \boldsymbol{\beta_j}^{\mathbf{2}} \ , \tag{2.11}$$

the L1-penalization shrinks all parameters ($\boldsymbol{\beta}$) as $\boldsymbol{\lambda}$ increases and sets some of them to exactly zero. Because of this nature, L1-penalization performs parameter estimation and variable

selection at the same time. L1 normally results in some of the parameters shrinking to zero, but others have little shrinkage. Unlike L1-penalization, L2-penalization tends to result in all small but none zero parameters.

2.2 Ensemble Methodology

The three learning algorithms have been explained, and now it is time to introduce the ensemble method which takes the information generated by SVM and neural network as inputs into a penalized logistic regression to achieve better prediction result.

Building the ensemble model has two parts. The first part is training the ensemble model, the second part is testing the ensemble model. The dataset is separated in to two sets, training set and testing set. Training set is used to train the ensemble model. The purpose of testing the model is to mimic the real life prediction scenario that the model needs to make accurate prediction when it faces a set of data that it has never seen before.

The two parts of the model building process are portrayed in the pipeline form in Figure 4 and Figure 5. There are multiple steps in each part; they are distinguished in different colors and numbers.

The ensemble model training part has three steps as demonstrated in Figure 4:

Step1: Train SVM and NN models using training dataset.

Step2: Run training dataset without result(Y) trough trained SVM and NN to get prediction result Y1 and Y2.

Step3: Use Y1, Y2 as inputs and result(Y) of the training set to train Penalized Logistic Regression Model.

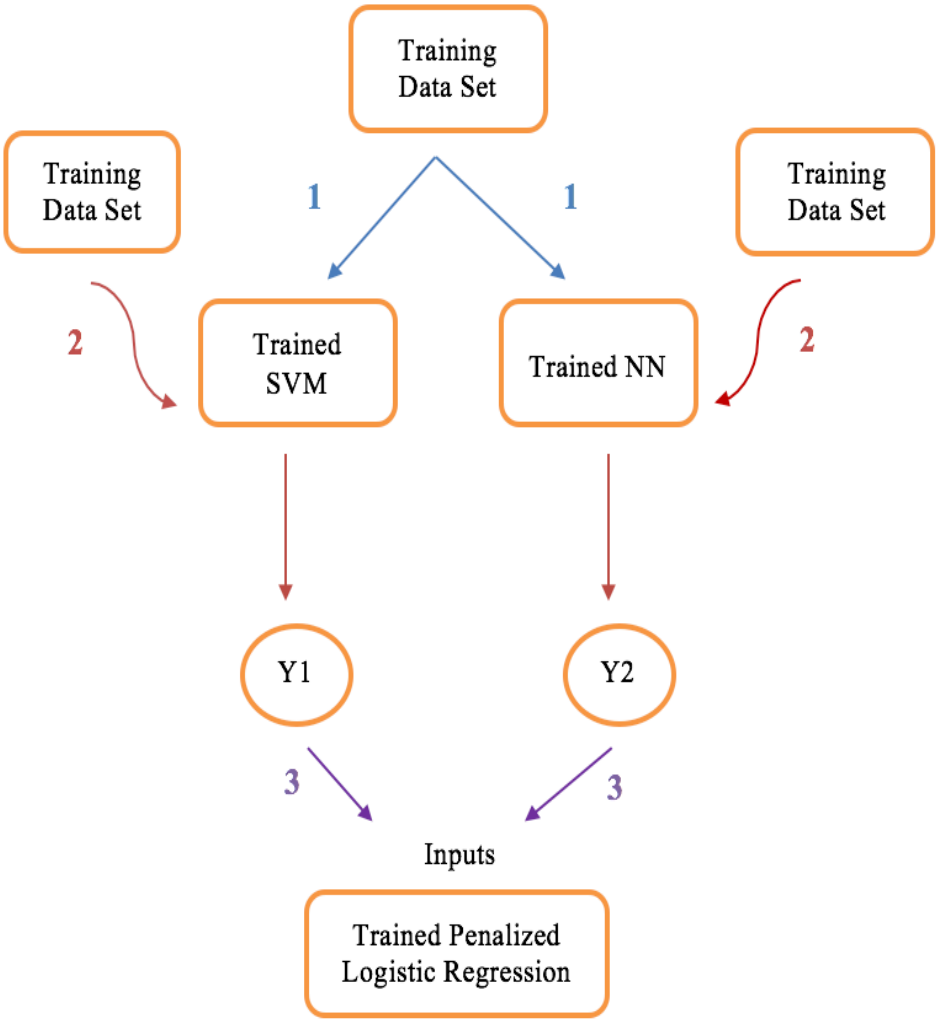

Figure 4. Ensemble Model Training Pipeline

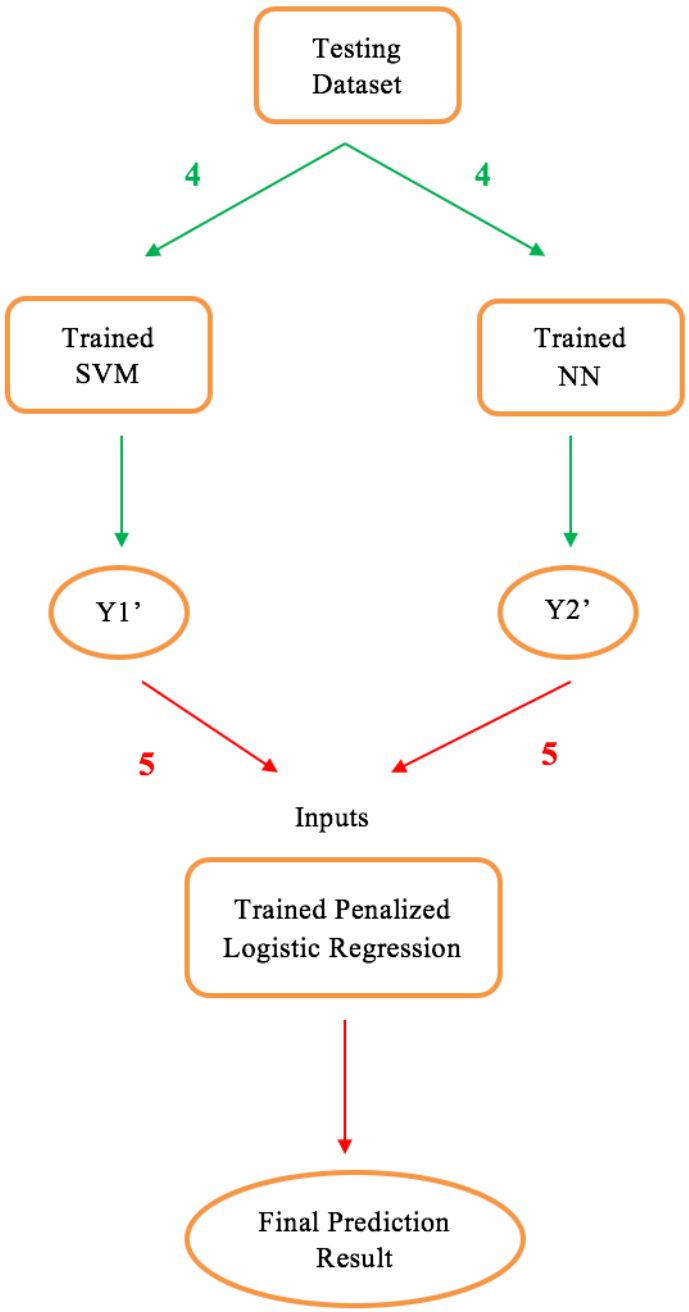

Figure 5. Ensemble Model Prediction Pipeline

The testing part of the ensemble model has 2 steps shown in Figure 5:

Step 4: Use trained SVM and trained NN taking testing set without (Y) as inputs for predictions, Y1' and Y2'.

Step 5: Use trained penalized logistic regression for final prediction use Y1' and Y2' as two inputs. The final prediction result is compared with the testing set label to calculate prediction accuracy.

CHAPTER 3 MODELING

3.1 Dataset Introduction

The machine data used in this paper is a set of turbofan engine degradation data created by NASA [1]. Data is already separated by training set and testing set. Both training set and testing set data have been normalized and feature engineered. There are 49 variables in total. In these 49 variables, 15 of them are from machine sensor raw data, 15 aggregated features calculated by using moving average of sensor values, and 15 features is the standard deviation of the sensor values. Originally there were 21 raw sensor features. However, 6 of them had constant readings, so they are not included. The rest 3 features are three raw features, cycle, setting1, setting2. Cycle is a time variable indicating the turbofan engine cycle time. Table 1 is a snapshot of a part of the data table.

All the features (X) are numeric. The response (Y) is a two-class categorical variable, where 1 indicates fail, and 0 indicates pass. Figure 6 gives the frequency of each class. 15% of the data points are fail, and 85% of the data are pass. Therefore, the frequency of pass and fail is not evenly distributed. This is a very common real life scenario. There are 20631 observations in the training dataset, and 100 observations in the testing dataset. The ensemble model will be built using training data and will be evaluated using the testing set for prediction.

Table 1.Turbofan Engine Degradation Simulation Data

cycle	setting1	setting2	s2	s3	s4	s6	s7	s8
0	0.45977012	0.16666667	0.18373494	0.40680183	0.30975692	1	0.72624799	0.24242424
0.00277008	0.6091954	0.25	0.28313253	0.4530194	0.35263336	1	0.62801932	0.21212121
0.00554017	0.25287356	0.75	0.34337349	0.36952256	0.37052667	1	0.71014493	0.27272727
0.00831025	0.54022989	0.5	0.34337349	0.25615871	0.33119514	1	0.74074074	0.31818182
0.01108033	0.3908046	0.33333333	0.34939759	0.25746675	0.40462525	1	0.66827697	0.24242424
0.01385042	0.25287356	0.41666667	0.26807229	0.29278396	0.27211344	1	0.77616747	0.18181818
0.0166205	0.55747126	0.58333333	0.38253012	0.46391977	0.26198515	1	0.72302738	0.18181818
0.01939058	0.3045977	0.75	0.40662651	0.25986484	0.3160027	1	0.64412238	0.15151515
0.02216067	0.54597701	0.58333333	0.27409639	0.43470678	0.2118501	1	0.61835749	0.22727273
0.02493075	0.31034483	0.58333333	0.15060241	0.44037497	0.30739365	1	0.60225443	0.22727273
0.02770083	0.60344828	0.25	0.32228916	0.23348594	0.31043214	1	0.75523349	0.22727273
0.03047091	0.59195402	0.66666667	0.2560241	0.26967517	0.3021607	1	0.75201288	0.28787879
0.033241	0.3908046	0.83333333	0.56024096	0.24307827	0.31363943	1	0.57809984	0.33333333
0.03601108	0.55172414	0.5	0.34337349	0.47765424	0.28544902	1	0.74557166	0.28787879
0.35180055	0.45402299	0.5	0.09939759	0.46130368	0.25472654	0	0.69404187	0.16666667
0.35457064	0.54022989	0.83333333	0.13554217	0.42620449	0.30384875	0	0.76489533	0.25757576
0.36842105	0.64942529	0.58333333	0.46686747	0.33573142	0.40428764	0	0.6489533	0.18181818
0.42382272	0.56896552	0.58333333	0.34036145	0.44168302	0.52380149	0	0.74074074	0.16666667
0.52908587	0.51724138	0.66666667	0.29216868	0.41225202	0.36833221	0	0.52173913	0.1969697
0.11357341	0.65517241	0.25	0.42168675	0.33158927	0.37542201	0	0.71819646	0.10606061
0.19944598	0.55172414	0.66666667	0.36445783	0.30455636	0.30064146	0	0.71819646	0.25757576
0.25761773	0.63218391	0.91666667	0.4126506	0.38936124	0.31785955	0	0.79066023	0.15151515
0.29916898	0.46551724	0.25	0.38253012	0.37431873	0.34638758	0	0.81481482	0.22727273
0.12465374	0.45977012	0.25	0.53614458	0.28646174	0.5624578	0	0.60225443	0.1969697

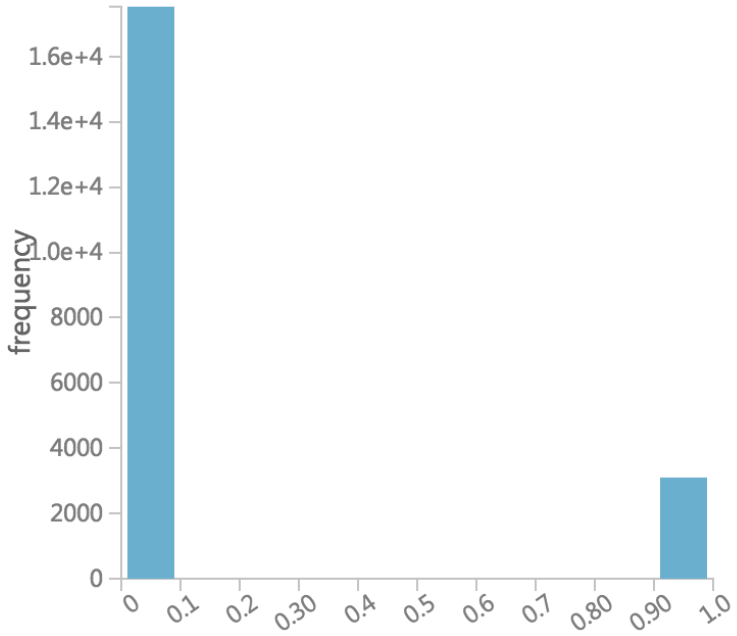

Figure 6. Frequency Histogram of Response Y

3.2 Ensemble Modeling

The statistic software used in this paper for modeling is the open source software R. The three R packages, e1071, nnet, penalized were used for building SVM, neural network, penalized logistic regression model respectively.

Follow the ensemble method presented in section 2.2 step by step, the support vector machine model and neural network model were first trained using the same training dataset, then the logistic regression model was trained by taking the training set in sample prediction results (Y1&Y2) from SVM and neural network model.

When building the SVM model, 10-fold cross validation was used to help validate the trained model accuracy. The 10-fold cross validation gave 97% accuracy for the following parameters. The tuning parameter C in formula (2.4) is chosen to be 1, the gamma parameter for the kernel is 0.02083, and the SVM kernel is radial kernel, $\boldsymbol{e}(-\boldsymbol{\gamma}|\boldsymbol{u}-\boldsymbol{v}|^{\mathbf{2}})$.

The nnet package for building neural network model has auto parameter tune functionality to pick the best model with the lowest error, so the best neural network parameters are picked by nnet R package using iteration. A 48-4-1 network is picked with 201 weights, and the weight decay parameter is 0.1.

The in training set prediction result Y1&Y2 along with the Y of the training set form the new training data for the penalized logistic regression model. Y1 and Y2 are all probability values between 0 and 1. A sample table is presented in table 2. Since SVM and neural network models were built using the same training set, then Y1 and Y2 should be very similar. The correlation coefficient of Y1 and Y2 is 0.98, which means Y1 and Y2 are very highly correlated. This finding confirms with the initial thought when constructing the methodology of ensemble model that penalized logistic regression need to be used to penalize the highly correlated inputs.

Table 2. Penalized Logistic Regression Model Training Data

	NNPred	SvmPred	Result
1	6.640414e-06	0.041007915	0
2	1.152303e-05	0.055681711	0
3	8.631169e-06	0.013780301	0
4	2.531289e-05	0.016185759	0
5	2.195143e-05	0.007383114	0
6	9.145423e-06	0.010897859	0
7	8.501707e-06	0.010319462	0
8	1.782980e-05	0.011500038	0
9	1.997463e-05	0.009523327	0
10	2.151653e-05	0.016477633	0
11	5.593279e-05	0.015256840	0

L1-penalization performs parameter estimation and variable selection at the same time. L1 normally results in some of the parameters shrunk to zero, but others have little shrinkage. Unlike L1-penalization, L2-penalization tends to results in all small but none zero parameters. Since there were only two inputs then there is no need to perform variable selection. As a result, only L2-penalization was used to build the penalized logistic regression model. The R Penalized package used to build the penalized logistic regression model chosen the best parameters with the lowest error along with the penalization. The L2 penalty is 172.5935. The trained penalized logistic regression model is as follow:

$$Y' = -4.303265 + 3.93586Y1 + 4.362072Y2 - 172.5935 \quad (3.1)$$

Y1: input from SVM.

Y2: input from Neural Network.

Using the trained models, following the two steps in figure 6 to get the final prediction result. The SVM, neural network and ensemble penalized logistic regression prediction confusion matrix are listed as Table 3.2, 3.3, 3.4 respectively.

The confusion matrix is used to check prediction accuracy. The model overall accuracy is the total corrected predicted counts divide by total counts. Here in this predictive maintenance problem, the goal is also to predict as many of true positive and as little false positive as possible.

Table 3. SVM Model Prediction Confusion Matrix

Reference **Prediction**	**0**	**1**
0	**74**	**7**
1	**1**	**18**
Prediction Accuracy	**(74+18)/100=92%**	

Table 4. Neural Network Model Prediction Confusion Matrix

Reference Prediction	**0**	**1**
0	**74**	**8**
1	**1**	**17**
Prediction Accuracy	**(74+17)/100=91%**	

Table 5. Ensemble Model Prediction Confusion Matrix

Reference Prediction	**0**	**1**
0	**74**	**7**
1	**1**	**18**
Prediction Accuracy	**(74+18)/100=92%**	

3.3 Result

Combining the prediction results from support vector machine, neutral network and ensemble penalized logistic regression gives rise to table 5. From this table, ensemble model and support vector machine model have the highest accuracy and sensitivity. And neural network model has a slightly lower accuracy and sensitivity value. The classification was done using 50% threshold.

Table 6. Model Accuracy Comparison

Model	**Classification Accuracy**
SVM	92%
Neural Network	91%
Ensemble Model	92%

CHAPTER 4: CONCLUSIONS, LIMITATIONS AND FUTURE WORK

From section 3.3, ensemble model and support vector machine model have the highest accuracy and neural network has slightly lower accuracy. The ensemble model successfully captured the highest accuracy among the two individual learning models because ensemble model has the same accuracy as SVM model, the lower accuracy of neural network model did not cause any negative effect on the ensemble model. This result shows that the ensemble method indeed achieved the goal of combining the prediction power of the two individual learning models.

Because the fact that the data has already been normalized and feature engineered, additionally with the good quality of the data, the prediction accuracy is high and similar for SVM and neural network models. Therefore, the ensemble model only successfully proved that it combined the prediction power of SVM and neural network and that it has the highest accuracy among all individual learning models, but didn't successfully show that it is able to improve accuracy more than any of the two learning models.

With more resources of machine maintenance data becoming available in the future, we can try to build more individual learning models and add them to the penalized logistic regression model for ensemble modeling. Cross-validation can also be done as an improvement instead of simply dividing the data into training and testing set. Furthermore, for most of the machine performance data, it is normally the case that the two classes in the response are unbalanced. So ideally before building the model, resampling method such as over-sampling,

SMOTE can be done to increase the number of the small class in order to achieve the optimal learning model performance.

REFERENCES

[1] A. Saxena and K. Goebel (2008). "Turbofan Engine Degradation Simulation Data Set", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository/), NASA Ames Research Center, Moffett Field, CA

[2] B. V. Dasarathy and B. V. Sheela (1979), "Composite classifier system design: concepts and me-thodology," Proceedings of the IEEE, vol. 67, no. 5, pp. 708-713.

[3] McCulloch, Warren; Walter Pitts (1943). "A Logical Calculus of Ideas Immanent in Nervous Activity". Bulletin of Mathematical Biophysics **5** (4): 115–133.

[4] Hastie, T., Tibshirani, R.,, Friedman, J. (2001). *The Elements of Statistical Learning*. New York, NY, USA: Springer New York Inc.

[5] Gareth James, Daniela Witten. *An introduction to Statistical Learning*. Springer, 2013.

[6] Cox, DR (1958). *"The regression analysis of binary sequences (with discussion)"*. J Roy Stat Soc B **20**: 215–242.

[7] Tibshirani, R. (1996). *Regression shrinkage and selection via the LASSO*. Journal of the Royal Statistical Society Series B-Methodological 58 (1), 267–288.

[8] Hoerl, A. E. and R. W. Kennard (1970). *Ridge regression: biased estimation for nonorthogonal problems.* Technometrics 12 (1), 55–67.