CHAPTER III
EXPERIMENTAL APPLICATION

3.1 Data Physical Preprocessing

Before performing the following variable selection routines in order to develop a well-designed predictive model, it is advisable to perform the various preprocessing routines such as filtering the extreme values from the training data set, transforming all the variables that are not normally distributed, and imputing missing values and replacing incorrect non-missing value.

Pre-processing of the data is one of the most complicated steps to a well-designed neural network model. That is, limiting the number of input variables in the model in order to reduce the effect in the "curse of dimensionality, and yet, retaining as much of the relevant information as possible with respect to the input variables included in the model that best explains the output responses. Also, it is important to exclude outliners or extreme values from the analysis and perform transformations to achieve normality in the input and target variables.

Therefore, in the experiment, the imperfect data were preprocessed. After wiping off the noise data created in the some short periods of paper factory shutdown or paper breaking interval, we still deleted 3 kinds of variables whose source data were seriously damaged. The 3 kinds of variables are ASA flow, 1^{st} cationic starch flow and 2^{nd} cationic starch flow. Furthermore, 8 kinds of variables were combined in couples into 4 different variables because they had obviously linear relations with each other, such as the speed of GCC flow and PCC flow. When the factory uses more GCC in the manufacturing process, the value of PCC will be absolutely reduced, vice versa. However, if they are not combined, the experiment data will include large amounts of "0" which will affect the accuracy of forecasting. The 4 groups are: the 4^{th} drying section pressure and the 10^{th} drying section pressure, jet size to S/P top flow and jet size to S/P bottom flow, starch to S/P top flow and starch to S/P bottom flow, GCC flow and PCC flow. Here the time interval between the inputs data collected is 10 minutes, but the time interval of output paper properties which should be the same as the production time of

one jumbo reel is about one hour. Therefore, the source data were transposed by every hour into six different columns.

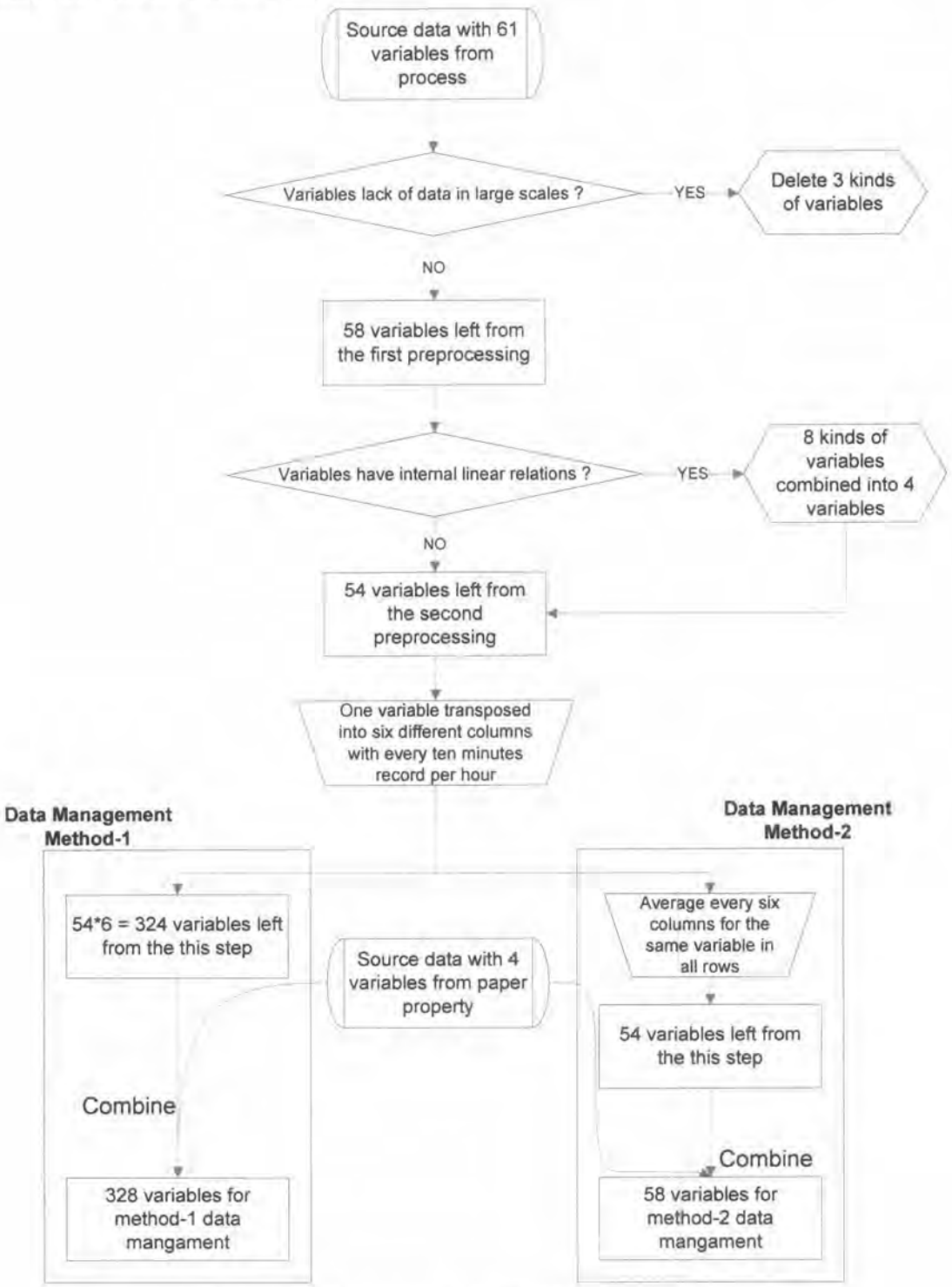

Figure 3.1: Source data preprocessing analysis.

In this way, the number of final inputs should be $[(61-3-4)\cdot 6+4]=328$. In this formula, 61 stands for the original number of

variables, 3 means the number of variables that deleted, the first 4 means the 8 variables that grouped into 4, and the second 4 means the 4 basic paper properties. .That means every 6 different records for 54 production parameters and 4 basic paper properties (Basis WT, Ash, Moisture and Thickness).

Compared to this data design method, another data set in which six different columns of the same parameter in every ten minutes of one hour were averaged in one column was prepared. Therefore, there are 58 variables for the second method of data set. The detailed data preprocessing management flow is show in Figure 3.1. The related operating parameters description for forecasting is shown in Table 3.1.

Description	Unit	Amount
Refining Energy	Kwh/t	3
Fan Pump Rotating Speed	Rpm	2
Wire silo Temperature	$^{\circ}C$	1
Steam Temperature	$^{\circ}C$	1
Freeness	Ml	3
Perform flow	L/h	1
Stock to PM flow	L/s	1
BMA flow	L/h	1
Jet size flow	L/h	2
Retention aid to F/P 2 flow	L/h	1
Cationic starch flow	L/h	1
Starch to S/P flow	L/h	2
OBA to mixing chest flow	L/h	1
Head box re-circle flow	L/m	1
Dye flow	Ml/h	1
GCC PCC flow	Kg/min	1
PM speed	M/min	1
Forming section vacuum	kpa	5

Head box pressure	kpa	1
Steam pressure	kpa	1
Drying section pressure	kpa	2
Couch roll vacuum	kpa	2
Taper header pressure	Mbar	1
Attenuator pressure	kpa	1
Nip pressure	N/cm	7
Chamber pressure	Mm H_2O	2
Head box Change length	Mm	1
Retention	%	2
Fiber Ratio	%	2
Valve open ratio of TS	%	1
Jet wire ratio	%	1
Head box consistency	%	1
Basic Paper Properties	/	4
ALL	/	58

Table 3.1: Related parameters description.

3.2 Input Pruning

3.2.1 Data Normalization

It is important to standardize the input variables in the model to assure convergence in the optimization process and if the model is unsatisfactory then a transformation might be advised. If one input has a range of 0 to 1, while another input has a range of 0 to 1,000,000, then the contribution of the first input to the distance will be swamped by the second input. So it is essential to rescale the inputs so that their variability reflects their importance, or at least is not in inverse relation to their importance. For lack of better prior information, it is common to standardize each input to the same range or the same standard deviation.

Standardizing either input or target variables tends to make the training process better behaved by improving the numerical condition of the optimization problem and ensuring the various default values involved in initialization and termination are appropriate.

Min-max normalization performs a linear transformation on the original data [14]. Suppose that $\min_A$ and $\max_A$ are the minimum and maximum values of an attribute A. Min-max normalization maps a value named as v of A to v' in the range [new_$\min_A$, new_$\max_A$] by computing

$$v' = \frac{v - \min_A}{\max_A - \min_A}(new\_\max_A - new\_\min_A) + new\_\min_A$$

Min-max normalization preserves the relationships among the original data values. It will encounter an "out-of-bounds" error if a future input case for normalization falls outside of the original data range for A.

In this research, two data sets are firstly normalized by min-max normalization into the range of [0, 1] that will benefit for the process weight balance and the steepest descent step of Quasi-Newton algorithm.

3.2.2 Input Reduction

According to the numbers of the two data sets samples respectively are 328 and 58. It is very necessary to use some technique to reduce the dimensions of data sets and make the predictive model simpler. In this study, PCA (Principal Components Analysis) and Standard Step-wise regression will be adopted to optimize the source data sets. Because of the large number of the first data set columns and many variables in the same parameter, Step-wise regression will physically cut so much information that it may cause the inaccuracy of final consequence. Compared with Step-wise regression method, Principal Components Analysis is based on constructing an independent linear combination of input variables in which the eigenvectors capture the maximum amount of variability in the original data set. Therefore, the first data set will be transacted by PCA. The second data set, the dimensions of which are reduced by averaging the 54 different production-operating parameters, will be transacted by Step-wise regression. So by using standard step-wise regression, the input variables that are not related to the curl forecasting were skipped.

After PCA, 41 principal components were selected from the first data set for the next training models; the detailed information is shown in the Figure 3.2. In the step-wise regression part, the significant variables are evaluated by R-Square statistic, which measures the proportion of the variability, observed in the data set (the minimum value of R-Square is defined as 0.0005). After step-wise regression, 23 variables were selected from the second data set sample for the next training model. Table 3.2 shows the details of chosen variables.

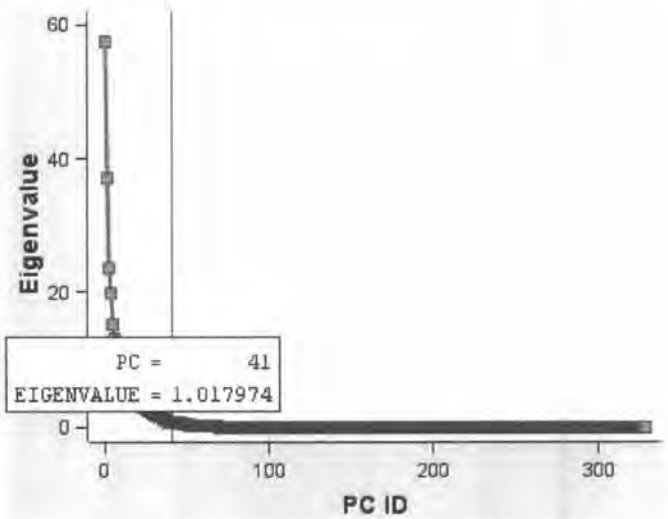

Figure 3.2: Principal component analysis result.

Selected Variables	R-Square
Couch roll low vacuum pressure	0.114439
The nip pressure of the 4th press stage	0.060797
Jet size to S/P top flow velocity	0.053224
Jet wire ratio of pulp flow and fabrics	0.034517
Cationic starch to mixing chest flow	0.015932
Nip pressure of TS size press part	0.009194
The 4th drying section pressure	0.006082
The 10th drying section pressure	0.009308
Starch to S/P bottom flow	0.019739

Trans suction box vacuum of forming section	0.008264
The freeness of short fiber for assembly line L2	0.008582
The freeness of long fiber	0.003981
The ratio of short fiber in the mixed of long fiber, short fiber and paper recycling	0.002531
Horizontal distance	0.002248
The freeness of short fiber for assembly line L1	0.003236
Top chamber pressure	0.002834
GCC_PCC flow	0.001556
LP steam temperature	0.001329
Nip pressure of DS size press part	0.000871
LP steam pressure	0.000795
The nip pressure of the 1st press stage	0.000694
Pick up pressure	0.000587
The consuming energy of refining the long fiber	0.000581

Table 3.2: Selected variables for second data set.

3.3 Tuning Neural Networks Models

3.3.1 Data Partition

The validation data set is used to reduce the bias and obtain unbiased prediction estimates in validating the accuracy of the fitted model by fine-tuning the model and comparing the accuracy between separate fitted models of different types and complexity. Fine-tuning the model is performed to avoid both under-fitting and over-fitting. Therefore, validation data set is applied in order to revert to a simpler predictive model as opposed to fitting the model to the training data set. That is, by fitting the model several times in selecting the appropriate complexity of the model while at the same time avoiding over-fitting that leads to poor generalization. This is because the

training error can be very low even when the generalization error is very high. In other words, choosing a model based on the training error will result in the most complex models being selected even if the same model generalizes poorly. Therefore, the selected model will result in poor generalization. Again, the validation data set is used in selecting the final predictive model. The test data set is used at the end of the model fitting in order to obtain a final honest and unbiased assessment of how well the predictive model generalizes by fitting the same model to the new data. The test data set should be applied to reduce the bias and obtain unbiased prediction estimates that are entirely separate from the data that generated the prediction estimates in evaluating the performance of the modeling fit. The reason for creating the test data set is that at times the validation data set might generate inaccurate results. Therefore, a test data set might be created in providing an unbiased assessment of the accuracy of the statistical results. The purpose of the validation and test data sets is to fit the model to new data in order to assess the generalization performance of the model.

In summaries, training set is a set of examples used for learning, which is to fit the parameters of the classifier. Validation set is a set of examples used to tune the parameters of a classifier, for example to choose the number of hidden units in a neural network. Test set is a set of examples used only to assess the performance of a fully-specified classifier.

In this MLP modeling, the preprocessed data sets are partitioned into 3 parts, 60% for training set, 10% for validation set, the rest for testing set. There are totally 2288 observations for each preprocessed data sets, suitable to adopt Newton-based algorithm to optimize the learning process.

3.3.2 Preliminary Training

In neural network modeling, there is not well known standard method for computing the initial weight estimates. Therefore, preliminary training is performed before network training, which is designed to determine the most appropriate starting values to be used as the initial weight estimates for the subsequent network training run that is critical to the iterative convergence procedure. Furthermore, preliminary training

is used to accelerate convergence in the iteration process, with the idea of avoiding bad local minimums in the error function.

Preliminary training uses a small number of random starting values, but takes a few iterations (20 by default) from each, and chooses the best of the final values as the starting value. The error surface of a linear function is a simple parabola. However, the error surfaces of neural network models are plagued with multiple minima, local and global, as shown in Figure 3.3. The solution avoids inferior local minima. Preliminary training is one strategy from initialization. The usual steps of initialization are shown as follows:

1. Standardize the input variables by subtracting the mean and dividing by the standard deviation.
2. Set the input-to-hidden weights to a small random number drawn from the normal distribution. The input-to-hidden weights should not be exactly set to zero, because singularities in the error function may cause lack of progress in the iterative optimization algorithm.
3. Set the output bias equal to the mean target, transformed by the link function.
4. Use the hyperbolic tangent activation function (tanh), so that the inflection point and, hence, the output from the hidden units, is close to zero. The arctangent or Elliott activation functions are acceptable alternatives because they are also centered at zero and range between negative and positive one. In contrast, the logistic function is centered at one-half.
5. Set the hidden-to-output connection weight to zero. On the first iteration, therefore, the output activation is solely given by the output bias, that is, the mean of the target (on the link scale). The hidden-to-output weights are poised to move positively or negatively.

Figure 3.3: Multiple minima in error function.

The reason that preliminary training is so critical is that many of the error depend on the initial weight estimates being a close appropriation to the final weight estimates, which will result in the acceleration of the iterative grid search procedure and avoid bad local minimums.

Therefore, preliminary training has been used in the Quasi-Newton sub-model to improve the health of initial weights estimates and biases. 20 observations from training data were randomly selected to do the preliminary training process.

3.3.3 Early Stop

Early stopping is designed to improve generalization in controlling network training by terminating network training once the validation error begins to increase in order to prevent over-fitting to the network model. Early stopping requires an enormous number of hidden layer units in order to avoid bad local minimums.

It penalizes large weights or bumps to the iterative process in the interpolation of the curve. The basic idea in early stopping is to stop the neural network iterative process when the validation error reaches a desirable minimum and avoiding a global minimum usually due to over-fitting in the neural network model.

The procedure of early stopping is as follows:

1. Divide the available data into two separate training and validation sets.
2. Use a large number of HUs.

3. Use small random initial values.
4. Use a slow learning rate.
5. Compute the validation error periodically during training.
6. Stop training when the validation error "starts to go up".

Early stopping is closely related to ridge regression [15]. If the learning rate is sufficiently small, the sequence of weight vectors on the iteration will approximate the path of continuous steepest descent down the error function. Early stopping chooses a point along this path that optimizes an estimate of the generalization error computed from the validation set. Ridge regression also defines a path of weight vectors by varying the ridge value. The ridge value is often chosen by optimizing an estimate of the generalization error computed by cross-validation, generalized cross-validation, or boot-strapping.

Considered that the pruning technology of early stopping and preliminary training can not be used in the same sub-model, we utilize early stopping in Double Dogleg sub-model to decrease the time consuming and avoid local minimum.

Early stopping has several advantages:

- It is fast.
- It can be applied successfully to networks in which the number of weights far exceeds the sample size.
- It requires only one major decision by the user: what proportion of validation cases to use.

But statisticians keep skeptical attitude on early stopping for the reason that it appears to be statistically inefficient due to the use of the split-sample technique.

3.3.4 Direct Connection

Direct connection is a special type of network architecture in which additional connections are added to the neural network design where each input unit bypasses the input-to-hidden layer and connects directly to each output unit. Direct connection is also called a skip-layer design. Bypassing the hidden layer and making a direct connection from the input layer to the output layer is called a skip layer design. A skip-layer design is essentially least-squares regression modeling in a neural network

design. Thus, an MLP with a skip-layer explicitly contains a linear model as a special case. Figure 3.4 shows a paradigm that combines the linear and nonlinear model.

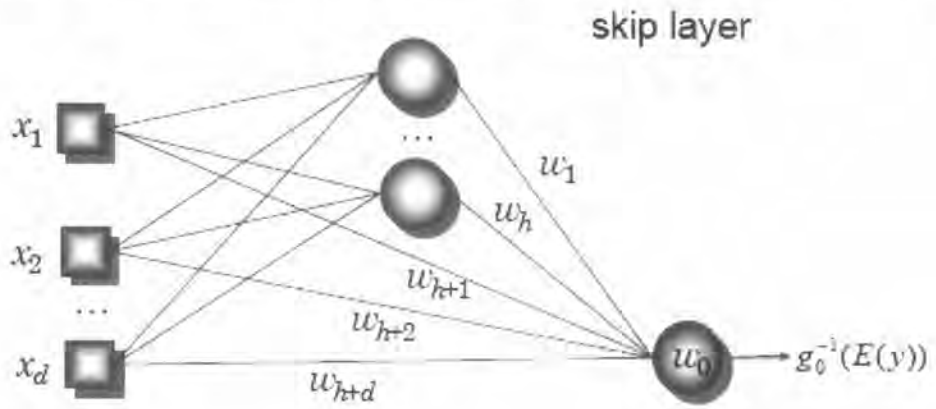

Figure 3.4: A skip-layer network.

The relations between the input and target are very complicated in this research. Since that a regression model is not pre-designed to explore the probably linear relations among them, so a skip-layer is designed to add in the neural network model. Therefore, the final neural network model is the same as a combination of non-linear MLP model and linear regression model, which avoids the possibility of losing linear information among input data sets and the target value. The result of comparison of whether using direct connection or not in the models is shown in the training results summaries of the next chapter.

3.3.5 Number of Hidden Units

Usually a single hidden layer unit is applied in the neural network model. But by adding more hidden units to the model, it will increase the complexity to the network design and can approximate any relatively smooth nonlinear function to any degree of accuracy. One of the most important decisions in network designing is the number of units in the hidden layer. Selecting the correct number of hidden units is an important aspect in producing good generalization performance, which is the main goal in network training. The best number of hidden units to apply to the neural network design depends on the number of input and output variables to the network model, the number of observations in the training data and the noise level in the underlying distribution of the training data, the number of training cases, the amount of noise in the targets, the complexity of the function or classification to be learned, the architecture,

the type of hidden unit activation function, the training algorithm and regularization. In most situations, there is no way to determine the best number of hidden units without training several networks and estimating the generalization error of each. Therefore, it is recommended to fit the network model numerous times with a different number of hidden units and analyzing the various modeling assessment statistics and stop the iterative process when the goodness-of-fit statistics begins to increase.

Generally, there is no standard method in selecting the appropriate number of hidden units. Selecting too many hidden units will lead to over-fitting; otherwise, too little hidden units will lead to under-fitting. In order to select the proper number of hidden units for each sub neural networks model, we decided to try some approximate number of hidden units and measured the performance. Considering about the observations and variables number of training data, in the experiment phase, we set the number of hidden unit firstly to 10, and vary this number up and down subtly to get the finally suitable number for each sub-model.