Web Services Project Report on

Performance prediction of real-world web services

Bhuvan M S (12IT16)

Nitin Jamadagni (12IT47)

Deepthi M Hegde (13IT208)

Narasimha Raju (12IT65)

Under the Guidance of,

Mrs. Sowmya Kamath

Department of Information Technology, NITK Surathkal

Date of Submission: 18 April 2016

in partial fulfillment for the award of the degree

of

Bachelor of Technology In **Information Technology** At

Department of Information Technology

National Institute of Technology Karnataka, Surathkal

April 2016

Department of Information Technology, NITK Surathkal
Web Services Project
End Semester Evaluation Report (April 2016)

Course Code : IT 450

Course Title: Web Services Project

Project Title: *Performance prediction of real-world web services*

Project Group:

Name of the Student	Register No.	Signature with Date
Bhuvan M S	12IT16	
Nitin Jamadagni	12IT47	
Deepthi M Hegde	13IT208	
Narasimha Raju	12IT65	

Place:NITK, Surathkal

Date:18 April 2016 *(Name and Signature of Web Services Project Guide)*

Abstract

The demand of efficient web service quality evaluation approaches is becoming unprecedentedly strong due to the increasing presence and adoption of web services on the World Wide Web. To avoid the expensive and time-consuming web service invocations, this paper proposes a collaborative quality-of-service (QoS) prediction approach for web services by taking into account the past web service usage experiences of web service users. Based on the collected QoS data, machine learning approach is designed for personalized web service QoS value prediction. In this paper, we have employed Bayesian Network and Naive Bayes algorithms to predict the response time and throughput time of web services.

Keywords: *Web Services; Machine Learning CLassifier; Big Data; Naive Bayes; Bayes Network*

Contents

1 **Introduction** 1

2 **Literature Survey** 2

2.1 Background . . . 2

2.2 Outcome of Literature Survey . . . 3

2.3 Problem Statement . . . 3

2.4 Objectives . . . 3

3 **Methodology** 4

3.1 Data description . . . 4

3.2 Work Flow . . . 4

3.3 Data Cleaning . . . 4

3.4 Data Preprocessing . . . 4

3.5 Machine learning algorithms . . . 7

3.5.1 Naive Bayes . . . 7

3.5.2 Bayesian Network . . . 7

3.6 Evaluation metrics . . . 8

3.6.1 Accuracy . . . 8

3.6.2 F1-Measure . . . 8

3.6.3 Receiver Operating Characteristic Curve: ROC . . . 8

3.6.4 Precision and Recall Curve . . . 9

4 **Implementation** 11

4.1 Work Done . . . 11

4.1.1 Experimental setup . . . 11

4.2 **Results and Analysis** . . . 12

4.2.1 Response Time Prediction . . . 12

4.2.2 Throughput Prediction . . . 14

4.3 Innovative Work . . . 15

4.4 Individual Contribution . . . 16

5 **Conclusion & Future Work** 17

List of Figures

3.1 Flowchart representing the Methodology work flow. 5
4.1 Naive Bayes ROC and PR curves for RT model 12
4.2 Bayes Network ROC and PR curves for RT model 13
4.3 Naive Bayes ROC and PR curves for TP model 14
4.4 Bayes Network ROC and PR curves for TP model 15

List of Tables

1 Features details 6

2 System Architecture for standalone implementation 11

3 Experimental Setup 11

4 Performance comparison of different classifiers for RT model. 13

5 Performance comparison of different classifiers for TP model. 15

1 Introduction

Web Services play an important role in the Service-oriented Architecture paradigm, as they allow services to be selected on-the-fly to build applications out of existing components. A Web service is a service offered by an electronic device to another electronic device, communicating with each other via the world wide web. There has been a drastic growth in the number of web services in the recent past. These web services are hosted at defined locations and the users of the services are spread across the world. The location of the user and other such attributes determine the response time and throughput time. It is thus important to know these values while using the web service. From the users perspective, it is important to know how long the service will take to respond to a request. It might be impractical to invoke a service that takes longer than its productivity to the user. From the service providers standpoint, it is important to choose an optimal location to host the service where maximum users are benefited and to ensure there is a balance in the performance experienced by the user. An analysis of response time and throughput time is hence a useful undertaking for the user as well as the service provider.

2 Literature Survey

2.1 Background

Analyzing the service computing literature [13], a number of adaptive models driven approaches have been elaborated for service selection [17], optimal service composition [14], fault tolerant Web services [21] and so on. However, there is still a need of real-world Web service datasets for validating new techniques and models.

Traditional schools such as: CSR (Centre for Software Reliability) City University London, University of Calgary in Canada, University of Lbeck in Germany or University of Birmingham dedicated important resources to this subject. In [22] were provided some reusable research datasets and built several large-scale evaluations on real-world Web services for promoting the research [19]. Browsing the literature we can find many neural network based software reliability prediction models and their prediction capability is confirmed better than some statistical models.

Based on a study of 50 scientific papers published in high quality journals and 35's proceeding of international conferences, I noticed that works on software reliability prediction models, from 1991 to 2011, has undergone a spectacular evolution.

Karunanithi was among the first who presented a model of software reliability prediction based on neural networks [15]. They used the execution time as input of neural network. They have also tested their model in different network architectures such as feed-forward or recurrent neural networks. Sitte presented a method based on neural networks for prediction and compared it with the parametric recalibration models using several significant metrics on the same data sets [16]. He concluded that the neural network approach provides better predictions. Cai proposed a prediction method based on NNs using the Backpropagation algorithm for training [12]. They used data from the last 50 errors as inputs to predict failure the next time out. They concluded that the effectiveness of the method depends strongly on the nature of the dataset used. Su and his team proposed the DWCM (Dynamic Weighted Combination Model) based on neural networks to predict software reliability [18]. They used different activation functions in hidden layers depending on SRGM (Software Reliability Growth Models) and applied it on two data sets by comparing the results with statistical models. Experimental results show that DWCM gives better results than traditional models. In [10] the authors

have derived SRGM models based on NHPP heterogeneous processes (Non-homogeneous Poison Processes) using a unified theory by introducing the concept of "multiple change points" in software reliability modelling. They estimated the parameters of the proposed models using three datasets of software failures and the results were compared with those of existing SRGM models. Their model predicted cumulative number of failures at different stages of development and operation of software.

2.2 Outcome of Literature Survey

From the literature survey, we realize that no prior work has been done to model throughput and response times of web services. Various machine learning algorithms when applied to the dataset produce valuable predictions with regard to the performance of the web services. We intend to build separate models for throughput time and response time and plot the Receiver Operating Characteristics and precision recall curves in order to make further analysis of each of the algorithms. This forms the basis of performance prediction for users of the web service as well as the service providers.

2.3 Problem Statement

The aim of our project is to predict and analyse the performance of various web services to different users based on User-features and Web Service Features.

2.4 Objectives

1. To develop response time prediction model.
2. To develop throughput time prediction model.
3. Evaluate each prediction model using Receiver Operating Characteristics (ROC), Precision-Recall Curves, F1 Measure.

3 Methodology

3.1 Data description

Web service QoS dataset [1, 2] was released to offer real-world data for Web Service researchers. The dataset contains 4 files, where "userlist.txt" and "wslist.txt" introdudce 339 users and 5825 real-world Web services of the dataset, respectively; Each User and WS have unique Id's that are associated. They further have features describing each. Users have *country,longitude,latitude* attributes and the WSs have *wsdl address,web service provider,web service country* attributes. "rtmatrix.txt" and "tpmatrix.txt" contain the response-time and throughput QoS values, respectively. Each of these files have a *user id* mapped to a *ws id* and the associated *rt* or *tp* values.

3.2 Work Flow

The workflow of the methodology has been described in the Figure 3.1

3.3 Data Cleaning

The raw is in terms of matrices. It needs to be brought to a data matrix format, where each row represents an instance of web service and each column represents different features of the web service.

In addition, the data contains mixture of nominal and numeric features. Long strings in the nominal feature values need to be encoded for each unique value in order to reduce the size of the data files.

3.4 Data Preprocessing

Following preprocessing has been done:

- Some features like user-id and ws-id, have numeric values but they need to be considered as nominal since they are ids. Hence they need to be converted to nominal features.

- Throughput and Response Time are our class labels. They are numeric attributes. Since we are building a classifier model, these numeric classes need to discretized

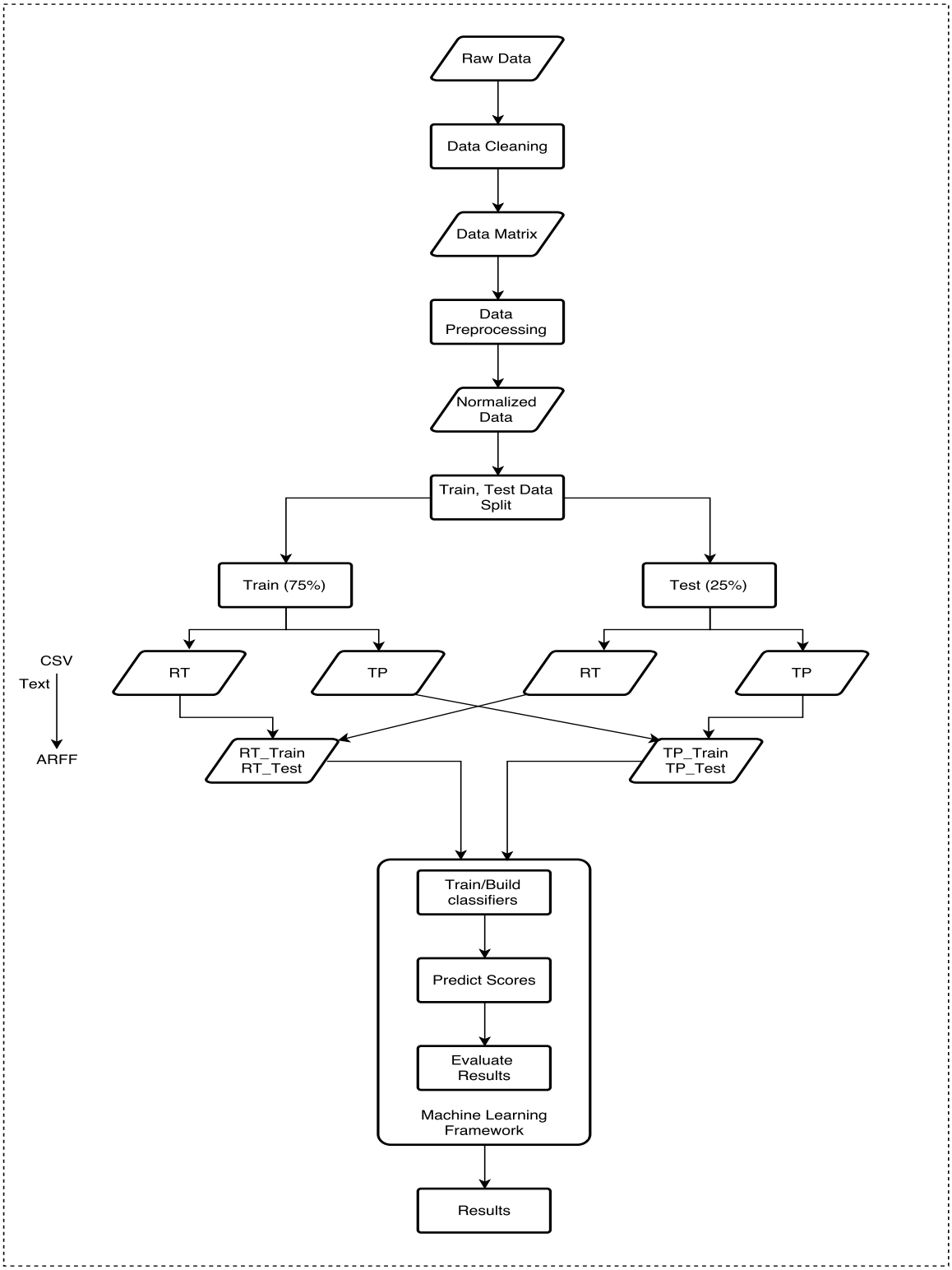


Figure 3.1: Flowchart representing the Methodology work flow.

into multiple classes. We chose 5 classes, hence response time and throughput have been mapped to 5 classes namely - very low, low, medium, high, very high (they have been labeled from 1-5 while referring in figures).

- File separation for Response Time prediction model and Throughput prediction model. In order to arrange the features and classes to correspond to standard data matrix format, class label needs to be the last column. Two separate files need to be created for each of Response Time prediction model and Throughput prediction model, which respective classes in the last.

- For each of the models, the data matrix needs to be split into training set and testing set. Since we have large number of instances, we chose 75% of instances as training data and 25% of instances as testing data after the shuffling all the instances to ensure we have randomized before splitting.

- ARFF file format ensure data integrity by adding metadata to the data matrix adding details about the type of each feature. The CSV file format of the data matrix needs to be converted to ARFF file format. This can be achieved using WEKA [8].

The cleaned, preprocessed and normalized data contains 19,74,675 number of instances. The features are tabulated in the Table 1

Table 1: Features details

Feature name	**Type**	**Details**
user ID	nominal	distinct 339
web service ID	nominal	distinct 5823
user country	nominal	distinct 31
user longitude	numerical	distinct 153
user latitude	numerical	distinct 152
wsdl address	nominal	distinct 5823
web service provider	nominal	distinct 2699
web service country	nominal	distinct 74
class label = rt / tp	nominal	distinct 5

3.5 Machine learning algorithms

Machine learning is a scientific discipline that explores the construction and study of algorithms that can learn from data. Such algorithms operate by building a model from example inputs and using that to make predictions or decisions, rather than following strictly static program instructions.

Several machine learning algorithms are present. But in order to handle such huge dataset as ours and deploy in standalone architecture, we have chosen probabilistic Bayesian classifier and Naive Bayes machine learning algorithms.

3.5.1 Naive Bayes

In machine learning, naive Bayes classifiers [6] are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

In the statistics and computer science literature, Naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naive Bayes is not (necessarily) a Bayesian method; Russell and Norvig note that "[naive Bayes] is sometimes called a Bayesian classifier, a somewhat careless usage that has prompted some Bayesians to call it the idiot Bayes model."

3.5.2 Bayesian Network

A Bayesian network [7] is a probabilistic graphical model (a type of statistical model) that represents a set of random variables and their conditional dependencies via a directed acyclic graph. Formally, Bayesian networks are DAGs whose nodes represent random variables in the Bayesian sense: they may be observable quantities, latent variables, unknown parameters or hypotheses. Edges represent conditional dependencies; nodes that are not connected (there is no path from one of the variables to the other in the

bayesian network) represent variables that are conditionally independent of each other. Each node is associated with a probability function that takes, as input, a particular set of values for the node's parent variables, and gives (as output) the probability (or probability distribution, if applicable) of the variable represented by the node. For example, if m parent nodes represent m Boolean variables then the probability function could be represented by a table of $2\hat{m}$ entries, one entry for each of the $2\hat{m}$ possible combinations of its parents being true or false. Similar ideas may be applied to undirected, and possibly cyclic, graphs; such are called Markov networks.

Efficient algorithms exist that perform inference and learning in Bayesian networks. Bayesian networks that model sequences of variables (e.g. speech signals or protein sequences) are called dynamic Bayesian networks. Generalizations of Bayesian networks that can represent and solve decision problems under uncertainty are called influence diagrams.

3.6 Evaluation metrics

The following evaluation metrics are calculated for each prediction model.

3.6.1 Accuracy

Accuracy is the proportion of true results (both true positives and true negatives) among the total number of cases examined.

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{1}$$

3.6.2 F1-Measure

The traditional F-measure or balanced F-score (F1 score) is the harmonic mean of precision and recall:

$$F1score = (2 * Precision * Recall)/(Precision + Recall) \tag{2}$$

3.6.3 Receiver Operating Characteristic Curve: ROC

In statistics, a receiver operating characteristic (ROC) [3], or ROC curve, is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The curve is created by plotting the true positive rate against the

false positive rate at various threshold settings. ROC analysis provides tools to select possibly optimal models and to discard suboptimal ones independently from (and prior to specifying) the cost context or the class distribution. ROC analysis is related in a direct and natural way to cost/benefit analysis of diagnostic decision making.

The Area Under the Curve (AUC) is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. This measure is used to evaluate different binary classifiers. Higher the ROC AUC the better is the classifier.

3.6.4 Precision and Recall Curve

In pattern recognition and information retrieval with binary classification, precision (also called positive predictive value) is the fraction of retrieved instances that are relevant, while recall (also known as sensitivity) is the fraction of relevant instances that are retrieved. Both precision and recall are therefore based on an understanding and measure of relevance.

1. Precision (P): is defined as the fraction of ground truth positives among all the examples that are predicted to be positive.

$$Precision = TP/(TP + FP) \tag{3}$$

   Where, TP, FP stand for True Positives and False Positives respectively

2. Recall (R): is defined as the fraction of ground truth positives that are predicted to be positives.

$$Recall = TP/(TP + FN) \tag{4}$$

   Where, FN denotes False Negatives.

3. Precision-Recall Curve [4]: The tradeoff between the precision and recall can be studied by looking at the plot of how precision changes as a function of recall. This plot is known as the Precision-Recall (PR) curve. The precision is plotted on the y-axis and recall on x-axis. The area under precision recall curve is a cost-effective metric for evaluation of classifiers. High value of area under the PR curve indicates better performance. Area under the PR curve is preferred way of studying the

performance of classifiers in skewed (i.e. imbalance between positives and negatives) datasets such as the one considered in this work [4].

4 Implementation

4.1 Work Done

The above specified workflow has been completely implemented in python using python-weka-wrapper for machine learning related tasks. The system configurations and tools used has been specified in the following subsections.

4.1.0.1 System Architecture The standalone implementation of the methodology was carried out with system configurations as shown in Table 2.

Table 2: System Architecture for standalone implementation

Operating System	Microsoft Windows 8 Pro and Ubuntu 14.04 LTS
Processor	Intel(R) Core(TM) i7-3610QM CPU @2.30GHz
RAM	8.00 GB (7.89 GB usuable)
System Type	64-bit OS, x64-based processor

4.1.1 Experimental setup

The tools used in the implementation along with their version number has been shown in Table 3.

Table 3: Experimental Setup

Tool	**Version**
python	2.7.9
pandas	0.16.0
scikit-learn	0.16.0
matplotlib	1.4.3
NumPy	1.9.2
Vowpal Wabbit	7.10

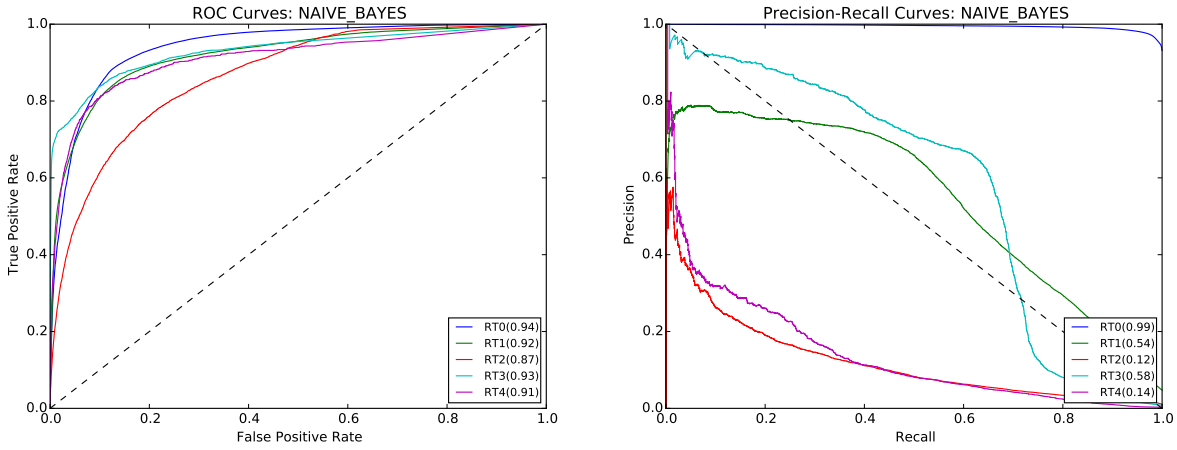
4.2 Results and Analysis

The classifier algorithm learns on the training data and saves a trained model, which is hence used to predict the probabilities of the occurrence of each class for each test instance. These scores are used to calculate the weighted area under the ROC curve and the weighted area under the PR curve, F1 score and plot the ROC and PR curves.

The above is carried out for Response time and Throughput model separately. The results of these models for different algorithms has been presented in the following sections.

4.2.1 Response Time Prediction

4.2.1.1 Naive Bayes The ROC curve and PR curve for Naive Bayes algorithm for RT model have been shown in Figure 4.1.



(a) ROC curve

(b) PR Curve

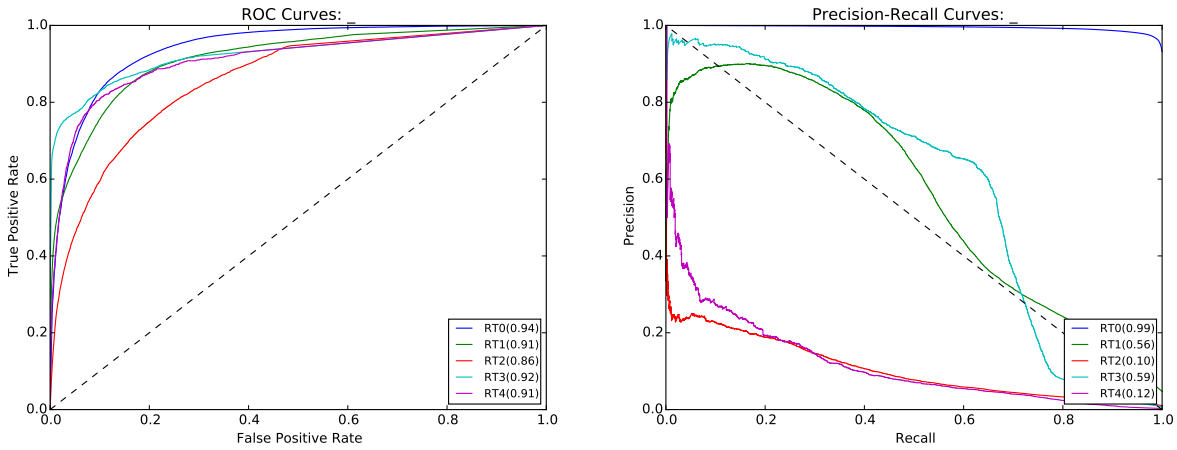
Figure 4.1: Naive Bayes ROC and PR curves for RT model

We can clearly observe from the ROC curves that the classifier is performing significantly well for most of the classes. Whereas, from PR curves we can see that the classifier is performing well in distinguishing some classes but not well for other classes.

In detail, we can observe, that Area under ROC for classes, RT0, RT1, RT3 and RT4 are 0.94, 0.92, 0.93 and 0.91 respectively are above 0.9, which is significantly good result. But for RT2 it is 0.87 which is comparably less, but good enough. Area under PR curves for RT0 is 0.99 which is very good, but for other classes - RT1 and RT3 is just above 0.5 which is good, but for Rt2 and RT4 the PR curve doesn't show promising results.

4.2.1.2 Bayesian Networks

The ROC curve and PR curve for Bayesian Networks algorithm for RT model have been shown in Figure 4.2.



(a) ROC curve

(b) PR Curve

Figure 4.2: Bayes Network ROC and PR curves for RT model

We can clearly obverse from the ROC curves that the classifier is performing significantly well for most of the classes. Whereas, from PR curves we can see that the classifier is performing well in distinguishing some classes but not well for other classes.

In detail we can observe that, the area under ROC curves for RT0, RT1, RT3 and RT4 are above 0.9 hence significantly good, whereas for RT2 it is 0.86 which is relatively less, but considerably good. The area under PR curves, for RT0 is 0.99 which is very good. For RT1 and RT3 it is above 0.5, hence considerable, but for RT2 and RT4 it is not promising.

4.2.1.3 Comparison of Results

Summary of the results containing the scores from various evaluation metrics as explained in the Methodology section is shown in Table 4.

Table 4: Performance comparison of different classifiers for RT model.

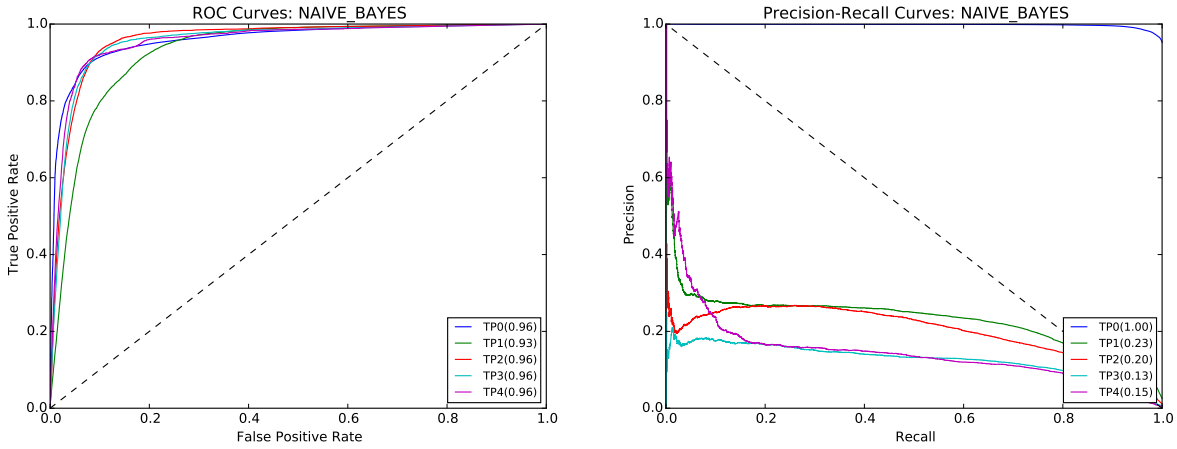
Algorithm	**ROC-AUC**	**PR-AUC**	**F1-Score**	**Accuracy (%)**
Naive Bayes	**0.9396**	**0.9585**	**0.9305**	**92.1209**
Bayes Network	**0.9390**	**0.9593**	**0.9170**	**89.9573**

Considering area under ROC as primary metric, we can say that both the classifiers are performing equally well, while Naive Bayes has better F1-score, also in showing better

accuracy.

4.2.2 Throughput Prediction

4.2.2.1 Naive Bayes The ROC curve and PR curve for Naive Bayes algorithm for TP model have been shown in Figure 4.3.



(a) ROC curve (b) PR Curve

Figure 4.3: Naive Bayes ROC and PR curves for TP model

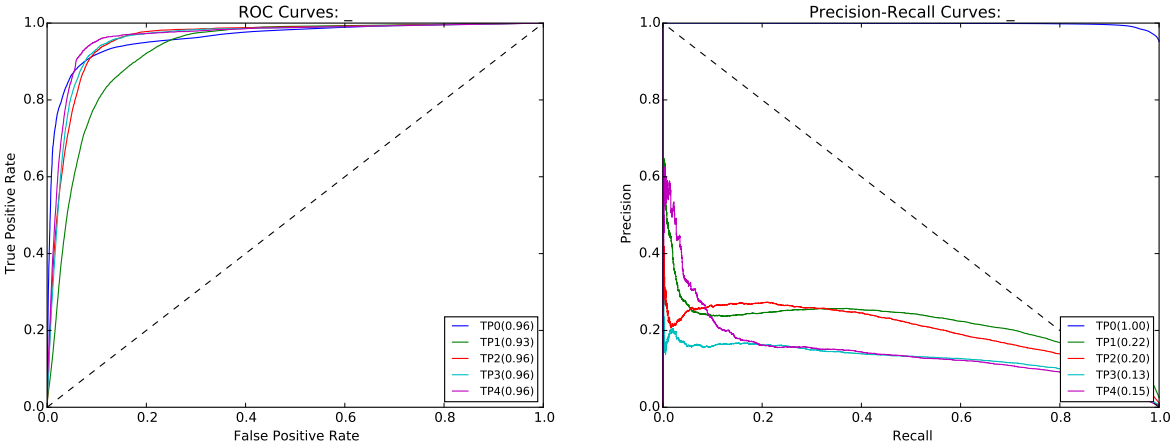
We can clearly obverse from the ROC curves that the classifier is performing significantly well for most of the classes. Whereas, from PR curves we can see that the classifier is performing well in distinguishing one classes but not well for other classes.

In detail, we can observe that, the area under ROC curve for all TP classes is above 0.9 and in fact for TP0, TP2, TP3 and TP4, it is above 0.95 which is very promising result. Area under PR curve for TP0 is 1.0 which is perfect but for other classes is below 0.25 which is not a prosiming result.

4.2.2.2 Bayesian Networks The ROC curve and PR curve for Bayesian Networks algorithm for TP model have been shown in Figure 4.4.

We can clearly obverse from the ROC curves that the classifier is performing significantly well for most of the classes. Whereas, from PR curves we can see that the classifier is performing well in distinguishing one classes but not well for other classes.

In detail, we can observe that, the area under ROC curve for all TP classes is above 0.9 and in fact for TP0, TP2, TP3 and TP4, it is above 0.95 which is very promising



(a) ROC curve

(b) PR Curve

Figure 4.4: Bayes Network ROC and PR curves for TP model

Table 5: Performance comparison of different classifiers for TP model.

Algorithm	**ROC-AUC**	**PR-AUC**	**F1-Score**	**Accuracy (%)**
Naive Bayes	**0.9593**	**0.9594**	**0.9338**	**91.8197**
Bayes Network	**0.9618**	**0.9593**	**0.9325**	**91.5126**

result. Area under PR curve for TP0 is 1.0 which is perfect but for other classes is below 0.25 which is not a prosiming result.

4.2.2.3 Comparison of Results Summary of the results containing the scores from various evaluation metrics as explained in the Methodology section is shown in Table 5.

Considering area under ROC curves as primary evaluation metrics, we can clearly observe that Bayes Network is performing slightly better than Naive Bayes.

4.3 Innovative Work

The data science approach used in this work is different from the the base papers [1, 2]. Our work includes building prediction model for Response Time and Throughout for each user-web service pair, enabling web service providers to estimate the performance for different users at the time of web service deployment. We have modeled it as a classification problem rather than a regression problem, as it is much practical and easier to get more insights in estimating the class of a performance rather than the exact

performance.

Moreover, our methodology flow enables us to compare multiple algorithms and choose the best one for the classification task. We have evaluated and observed the classifier performance from cost-effective measures like ROC curves and PR curves, which provide us the actual performance values even with the skewed class distribution. In addition, this work analysis huge dataset in considerable amount of time.

4.4 Individual Contribution

- Bhuvan M S (12IT16): Flowchart design, Framework integration.
- Nitin Jamadagni (12IT47): Data cleaning, Data preprocessing.
- Deepthi M Hegde (13IT208): Bayesian network classifier, Generating ROC and PRC plots.
- S Narasimha Raju (12IT65): Naive Bayes classifier, Getting F1 measure and percentage accuracy.

5 Conclusion & Future Work

The number of web services on the world wide web is increasing by the hour. The potential of these web services lies in how fast and efficiently they are accessible and usable by the user sitting elsewhere in the world. The prediction of the response and throughput times gives the user a fair idea as to whether or not to use the web service. If the productivity of the service is less than the time taken by the web service, the user will not opt to use the service. This dataset can also be used to predict the optimal location where the web service has to be hosted to give good performance to most users and to instigate users to use the service more and more.

We plan to perform feature importance mining in order to let only the important features determine our model. We also plan to implement various other machine learning algorithms on the data and draw an analysis using the ROC and precision-recall curves. We aim to develop a recommendation plugin which estimates the response time and throughput time of the service. For the service provider, we aim to build a recommender that gives the ideal location for hosting of his service.

References

[1] Zibin Zheng, Yilei Zhang, and Michael R. Lyu, Distributed QoS Evaluation for Real-World Web Services, in Proceedings of the 8th International Conference on Web Services (ICWS2010), Miami, Florida, USA, July 5-10, 2010, pp.83-90.

[2] Yilei Zhang, Zibin Zheng, and Michael R. Lyu, Exploring Latent Features for Memory-Based QoS Prediction in Cloud Computing, in Proceedings of the 30th IEEE Symposium on Reliable Distributed Systems (SRDS 2011), Madrid, Spain, Oct.4-7, 2011.

[3] Receiver Operating Characteristics. Wikipedia, The Free Encyclopedia.Web.2016.

[4] Precision Recall Curves. Wikipedia, The Free Encyclopedia.Web.2016.

[5] F1 Score. Wikipedia, The Free Encyclopedia.Web.2016.

[6] Naive Bayes. Wikipedia, The Free Encyclopedia.Web.2016.

[7] Bayesian Networks. Wikipedia, The Free Encyclopedia.Web.2016.

[8] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

[9] (W3C recommendations 2007) http://www.w3.org/TR/.

[10] C.Y. Huang, M.R. Lyu, Estimation and Analysis of Some Generalized Multiple Change-Point Software Reliability Models, IEEE Transaction on Reliability, vol. 60, no. 2, pp. 498-514, (2011).

[11] Henrik Madsen, Rzvan-Daniel Albu, Ioan Felea, FlorinPopeniu-Vldicescu, A New Ensemble Model For Short Term Wind PowerPrediction, PSAM11andESREL 2012, 11th International Probabilistic Safety Assessment and Management Conference and The Annual European Safety and Reliability Conference Scandic Marina Congress Center, Helsinki, Finland 2529 June (2012).

[12] K.Y. Cai , L. Cai , W.D. Wang , Z.Y. Yu , D. Zhang , On the neural network approach in software reliability modeling, The Journal of Systems and Software, vol. 58, no. 1, pp. 47-62, (2001).

[13] L.-J. Zhang, J. Zhang, and H. Cai. Services computing. In Springer and Tsinghua University Press, (2007).

[14] M. Alrifai and T. Risse. Combining global optimization with local selection for efficient qos-aware service composition. In Proc. 18th Intl Conf. World Wide Web (WWW09), pages 881890, (2009).

[15] N. Karunanithi , Y.K. Malaiya , D. Whitley, Prediction of software reliability using neural networks, Proceedings of the Second IEEE International Symposium on Software Reliability Engineering, pp. 124130, (1991).

[16] R. Sitte, Comparision of softwarereliability-growth predictions:neural networks vs parametric recalibration. IEEE Transactions on Reliability, vol. 48, no. 3, pp. 285- 291, (1999).

[17] T. Yu, Y. Zhang, and K.-J. Lin. Efficient algorithms for web services selection with end-to-end qos constraints. ACM Trans. the Web, 1(1):126, (2007).

[18] Y.S. Su, C.Y. Huang, Neural-Networks based approaches for software reliability estimation using dynamic weighted combinational models, The Journal of Systems and Software, vol. 80, no. 4, pp. 606-615, (2007).

[19] Yilei Zhang, Zibin Zheng, and Michael R. Lyu, WSPred: A Time-Aware Personalized QoS Prediction Framework for Web Services, in Proceedings of the 22th IEEE Symposium on Software Reliability Engineering (ISSRE 2011). http://www.wsdream.net/

[20] Yilei Zhang, Zibin Zheng, and Michael R. Lyu, Exploring Latent Features for Memory-Based QoS Prediction in Cloud Computing, in Proceedings of the 30th IEEE Symposium on Reliable Distributed Systems (SRDS 2011), Madrid, Spain, Oct.4-7, (2011). http://www.wsdream.net.

[21] Z. Zheng and M. R. Lyu. A qos-aware fault tolerant middleware for dependable service composition. In Proc. 39th Intl Conf. Dependable Systems and Networks (DSN09), pages 239248, (2009).

[22] Zibin Zheng, Yilei Zhang, and Michael R. Lyu, Distributed QoS Evaluation for Real-World Web Services, in Proceedings of the 8th International Conference on Web Services (ICWS2010), Miami, Florida, USA, pp.83-90, July 5-10, (2010).