CSCI 729: Topics in Data Management (Web Services)

Programming Assignment 4

Proposal, Design & Evaluation Document

Name: Harshit Shah (hrs8207@rit.edu) **Application:** Service classification using Weka J48

1. Proposal

- ***General Idea of the Topic (Web service classification):***

  I am proposing an interesting service data mining problem of 'Service Classification'. Various data mining approaches can be used with web services. According to paper [1], four mining task can be applied to web services: Predictive Modelling (Classification, Value prediction), Clustering or Segmentation, Link Analysis (Association, Sequential, Similar time discovery), Deviation analysis.

  We will more focus on Predictive Modelling as data mining approach is Service classification. Predictive modelling is to predict new value for given attributes based on past experiences or available service data. The 'classification' form of predictive modelling is used to predict nominal or discrete values while the 'value prediction' or 'regression' is used to predict continuous values.

- ***Motivation:***

  Motivation for the project is to define a criteria and classify new web services based on previous experience of related service data.

- ***Proposed Approach:***

  I am proposing a service classification technique for a **Mashup** data collected from Programmable Web.

  In this approach, I will classify '*rating*' into 5 different categories (One, Two, Three, Four, Five). Here, '*rating*' is our class label or class variable. It will be predicted based on the following 5 attributes.

  - *downloads* : Total number of mashup downloads
  - *useCount* : Use count for a mashup
  - *numComments* : Total number of mashup downloads
  - *updated* : Updated year for the mashup

  On mashup data (which is collected from programmable web during summer 2014), I am applying **J48** classification algorithm of 'Weka'. J48 is a decision tree algorithm which is java implementation of C4.5 algorithm. It will take our mashup data as input and make the prediction model.

  This model is later on used in our application to predict the ratings for the new services for given input attributes (*downloads, useCount, numComments, updated*). We will show the working of application, pre-processing of the data and results in the following sections.

- ***Benefits:***

  Benefits of this approach is to predict the rating of any services from our prediction model based on certain given parameters. It will help any of the service consumer to go ahead with the service.

- ***Relation with what we learned in the class:***

  We learned Web service classification and different data mining techniques to classify Web services. From studied different classification methods like decision trees, support vector machine, nearest neighbor, our proposed method is similar to decision tree classifier. I have made a model to predict the class variable for unseen instances similarly we learned in the class.

2. Design of Web service data analysis system

Application is made using Spring 4 framework on a maven based web application. It uses Spring Data MongoDB library to perform all MongoDb operations from java.

Basically, it is an extension of programming assignment 3 in terms of application interface. It also provides a functionality to classify ratings of the 'unseen' services related to Mashup based on fixed attributes. We will see in testing application section.

Design of the data structures:

I have created two data structures to parse and store APIs and Mashup files into MongoDb and querying them from it.

1. **API:**

   Package: *com.rit.pa4.model*

   API is a simple POJO class to represent the fields of API collection of database into a Java Object. API models a data structure for the API collection of the database.

2. **Mashup**

   Package: *com.rit.pa4.model*

   Mashup class is a simple POJO class to represent the fields of Mashup collection of database into a Java Object. Mashup class models a data structure for the Mashup collection of the database.

3. **MashupClassifierView**

   Package: *com.rit.pa4.view*

   MashuClassifierView is a simple POJO class to render mashup_classifier view details into mashup controller and vice versa. It models view details into controller.

Classifying Mashup:

When user enters details about *downloads, useCounts, numComments* and *updated_year*, using MashupClassifierView data structure it will transfer contents to MashupController.

The MashupController will perform following tasks:

- Load the Weka 'model' which is generated from historical data (prediction model).
- Now make an '.arff' file which contains test data to be predicted.
- Controller will call classifier class from 'weka.jar' and predict the ratings.

Code Walkthrough:

Spring MVC pattern is followed while developing code.

1) Package: com.rit.pa4.controller
   HomeController.java, APIController.java, MashupController.java: Controls the request received from the users and process it to MongoDB (Inserting data, querying database).
2) Package: com.rit.pa4.model
   API.java, Mashup.java: Models respective data structures for the respective collections of the database.
3) Package: com.rit.pa4.model
   APIView.java, MashupView.java, MashupClassifierView.java: Models a list of fields to render respective views.
4) home.jsp, api.jsp, mashup.jsp, mashup_classifier.jsp: view files accessible to clients.
5) Resources: This directory contains javascript, css and images for the UI.

3. The Development Process

Overall application development process took place in the following steps:

Step 1: Reuse 'Mashup' inserted into MongoDB from Programming Assignment 3.

- First, generate the csv file of Mashup data using instructions/command given in ReadMe.pdf file.
- This file will be called 'Mashup.csv'.

Step 2: Preprocess 'Mashup.csv' file.

- Use ***python script*** 'ReadCSV.py' available inside a python project 'readcsv.
- This script will generate '*ratings*' into 'One' to 'Five' class category [r>=4.7:"Five", 4<=r<4.7:"Four", 3<=r<4:"Three", 2<=r<3:"Two", r<2:"one"] and also generate '*update year*' from 'updated date'.
- This file will be called 'Mashup_new.csv'

Step 3: Clean 'Mashup_new.csv' file.

- This file will have total 7392 records.

- First step of cleaning is to remove all instances without any class (ratings) or 'blank' value. (7065 records are left after performing this step)
- Second step is to remove all records with '*downloads*' as 'blank' value. (4897 records are left after this step)
- Third step is to clean all the values with 'ratings' as 'One' or 'Two' ratings. Only 62 records with this values are there, so can't help much in predicting and for them data is producing biased result towards ratings 'Five'. So better to remove them. (4835 records are left after this step)
- Overall cleaning step is tedious and manual task. I have cleaned this data manually using Excel.
- Name this file as 'Mashup_cleaned.csv' file.

Step 4: Apply J48 classifier of Weka on an input file 'Mashup_new.csv'.

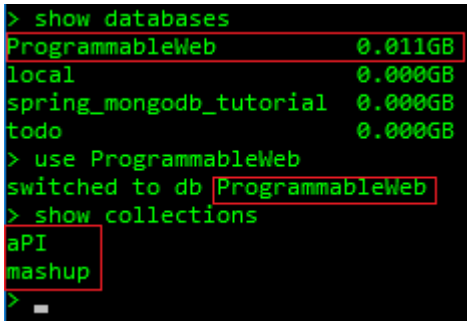
- This decision tree technique is applied considering training set on the input file and generated model is saved.
- Overall accuracy of the model is '82.0062%'. We will discuss it more in the results and evaluation section.
- Save this model file as 'ratings_3_labels.model'.

Step 5: Include this model file into our data analysis system and predict ratings for 'unseen' services.

- Give input values/parameters and predict the ratings for 'unseen' web services. We will see through the testing and screenshots of the application section.

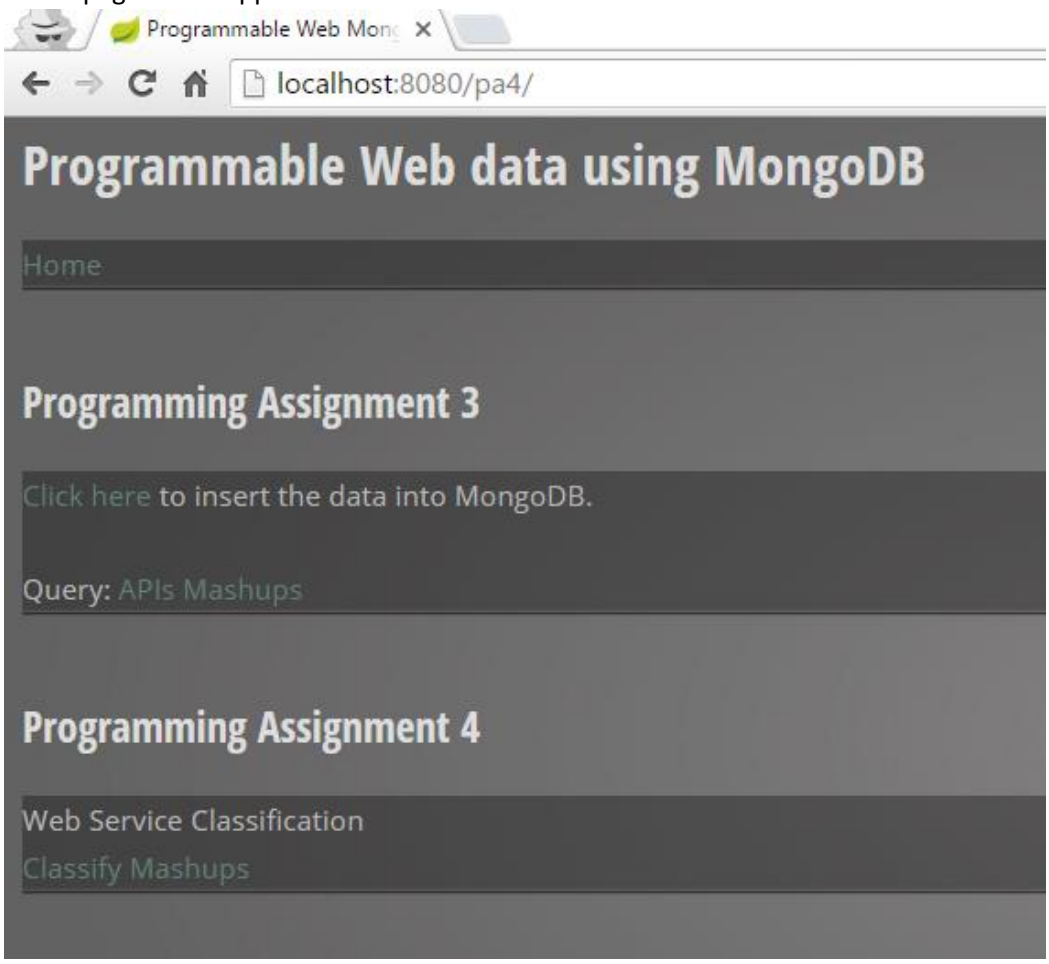
4. Screenshots of the application and testing scenarios

a. 'ProgrammableWeb' database and collections.



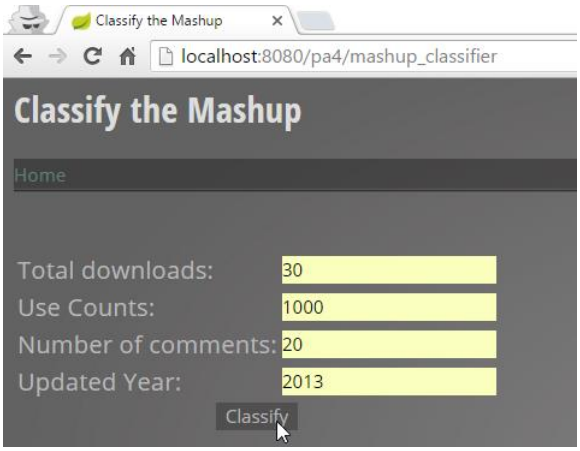
b. Data in 'mashup' collection.

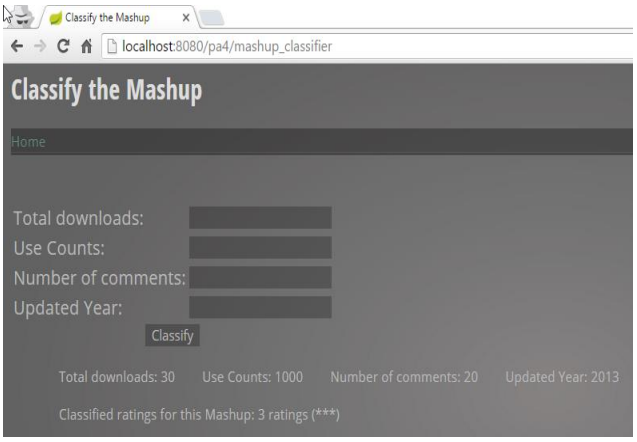
c. Home page of the application.



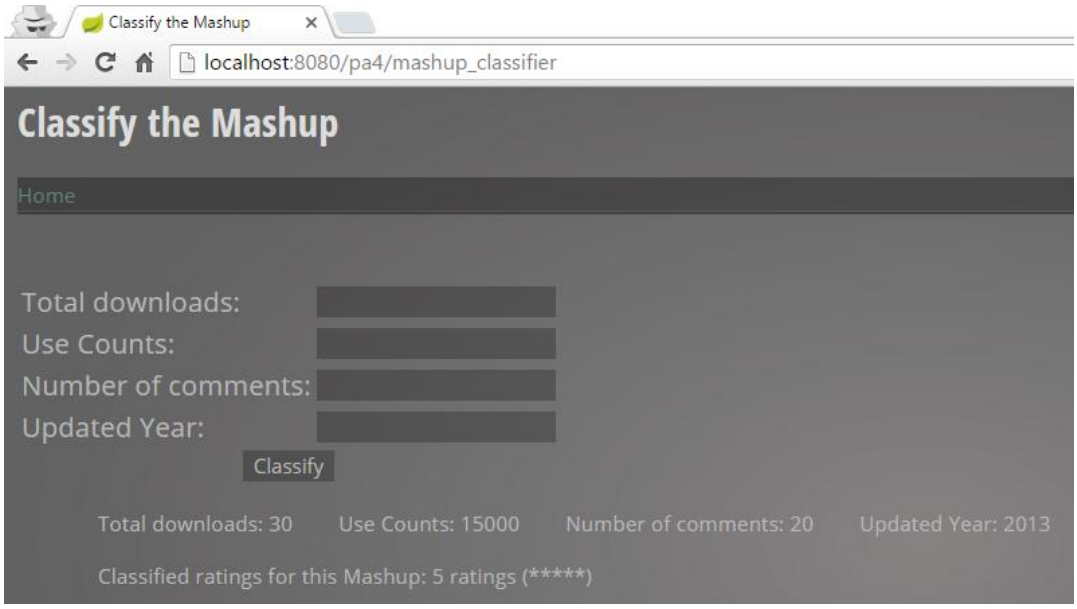
d. Click on 'Classify Mashups' link and you will see the following screen of our data analysis system.

e. Less number of '*UseCount*' results in 'Three' ratings depending upon other parameters.





f. Keep all parameters same and update useCounts to '15000' and you will get 'Five' ratings as shown below.



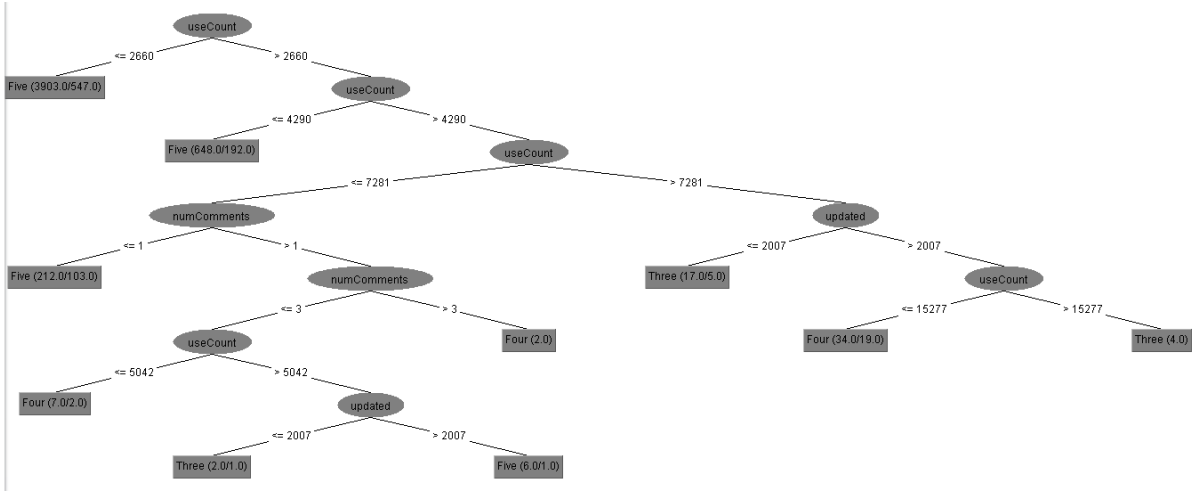
5. Results and Evaluation

In this section we will discuss the results obtained and the evaluation of our prediction model.

Decision Tree algorithm: J48
Tool: Weka
Test Options: Using training set

Number of Instances: 4835

Classification Model (J48 pruned tree shown in the following diagram):



From the above tree, we can conclude that '*downloads*' from our data is not very significant attribute as it is not present in the tree for predictions. The most important attribute is '*useCount*' because the information gain of this attributes is highest among all other attributes. '*Update_Year*' also plays an important role while predicting a class variable.

Evaluation of this model:

Correctly classified instances: 82.0062% (model accuracy)

From 4835 instances, 3965 instances are correctly classified. So, this model has overall good accuracy. But while evaluating this model, I came to know that this data was little bit biased towards rating "Five". And that might be the reason of getting low accuracy earlier. So, I had to clean the data more and remove all instance with ratings 'One' and 'Two' as only 62 instances were present with this ratings and all of them are biased towards 'Five' rating while predicting.

I have also tried to evaluate this model using different approaches and trying to get more accurate results but couldn't able to get better accuracy than this. My few experiments details are as below:

- Removed all records with class variable (rating) is 'blank' (7065 total records remaining, accuracy = 68.1812%)
- Removed all records with *numComments, useCount & downloads* = 0 (5026 total records remaining, accuracy = 61.7788%)
- Removed all records with *useCount* = 0 (6472 records, accuracy = 61.7788%)
- Removed all records with *downloads* = 0 (2129 records, accuracy = 61.7788%)
- Removed all records with *numComments* = 0 (4894 records, accuracy = 61.7788%)

I have also evaluated this model on few of the records as test records which were removed while making classification model. For almost all of these instances, it has predicted the correct values.

Reference:

1) Nayak, Richi, and Cindy Tong. "Applications of data mining in web services."*Web Information Systems–WISE 2004*. Springer Berlin Heidelberg, 2004. 199-205.