Data Mining: Lessons Learned

Report of the CFS e-Commerce Taskforce

Team 7: Epitome

Scope of Data Mining Applications

Kinds of business problems for which Data Mining techniques are in principle applicable – such as the two in the exercise and others of like type that you may imagine. What about a business-decision-making problem makes it addressable by Data Mining?

Data mining is a process of analyzing data from different perspectives and converting it into useful information. It finds patterns and correlations between different attributes in database.

Data mining is majorly used by companies which have a strong customer base and focus on their activities. It helps to determine the relationship between different factors such as price, product ranking, competitions, customer demographics etc and their impact on sales, profit, CRM etc.

Some of the major applications where data mining techniques are used are:

1. Credit Scoring

   Some banks and credit card companies, use credit scores to predict which customers can qualify for loan and which customers are likely to bring more revenue to the company based on past applicant profiles, credit given and payback.

2. Fraud Detection

   It can be used to prevent frauds by approving or blocking the transactions for example during credit card transactions based on the legitimate and fraudulent transactions in the history.

3. Demographic segmentation

   Based on the customers' profile, attributes such as age, gender, education, income, previously bought products, a new product can be targeted to customers after analysing which customer is likely to buy the new product.

   For example, with the help of data mining, a bank can send a customer new offers and policies based on their historical data. This type of predictive analysis can help a company to advertise their products or services based on customer's preferences.

Similarly a retailer may send targeted promotion based on customer's purchase history and can display recommended items online based on what products they viewed. Also by studying what items a customer purchased in the past, the retailers can give offers and discount on related products to improve their sales and also attract more customers.

4. Marketing/Advertisement Effectiveness

   Based on the past advertisement campaigns, demographic targets and product categories, new advertisements can be proposed which are likely to be successful based on data mining. This can improve the sales and profit of a company.

5. Product reliability

   Analyzing the past product or parts, their specifications, customer usage, customer reviews, maintenance requirement etc., it can be predicted how successful a new product can be and what would be its maintenance requirement.

6. Manufacturing Tolerances

   Based on the data about the product in the past, data mining can be used to analyze or predict the precision required for manufacturing a product and thus cost of failures can b reduced. As well as mechanical diagnosis can be done to predict the cause of failure and recommended repairs can also be predicted in a similar way.

7. Billeting (job assignments)

   Based on the performance of existing employees and their skillset, data mining can predict the behavior, performance and suitable position to be assigned to a new employee. The training data for such model will be the past data of employee, their performance and their skillset. The outcome can be the predicted behavior of the new employee and a appropriate position of the employee

Data requirements (e.g. what kind of data, how much do you need for Data Mining to be effective?)

For the purpose of data mining, the input data should be consistent and cleaned so that the pre-processing of data becomes easier.

For the analysis of data, we require train and test data. In order to do a predictive analysis we need a larger amount of train data. The more the train data is, the better would be the accuracy or in other words, the prediction error would be less.

The train data should be diverse in the sense that it should include most of the special/edge cases so that accuracy is not reduced when attributes for the test data are predicted.

Attributes in test data should be same as the train data.

Feature attributes in train data set has to be correlated to the class label, which is the prerequisite for classifier. Because in classifier, we actually decide the class label of each test instance from the similarity of other feature attributes with other instance.

Expected kinds of results: results expected from the Data Mining exercise. Should you expect definitive, probabilistic or other kinds of decision support (based on 12a & 12b)?

We expected definitive results from the task. In the first part we wanted to find the most appropriate products for new customers. We wanted our classifier to classify any new customer in certain classes. Each customer's record will be classified in a particular class of product. Company can then sell that product to the customer. Our classifier will classify each customer record in a class which is a definitive outcome of the data mining exercise.

II. Kinds of Classifiers

a. KNN and D-Tree, how they work, i.e. on what do they base their predictive abilities? What information do they use from past data?

KNN**:**

For KNN algorithm, it mainly consists of 3 parts: pre-processing, get k nearest similarity instances, get label for test data based on its k nearest instances.

For pre-processing phase, we read train data, test data, weight and similarity matrix for processing. Train data is the history data which used to predict labels for test data. Test data is the information for new customer, except that it misses labels. Weight is a double array, each element in the array stores a weight of an attribute of the data set corresponding by their indexes. Similarity matrix is a map of discrete attribute name and its double[][] array of similarities between different attribute values corresponding the order of attribute declaration in arff file.

After reading essential files, the pre-processing part also includes finding max and min attributes for each numeric values in train data set, and normalize each numeric value in train data and test data to be a real number in the range of [0,1]. The calculate formula is (xi - ximin)/(ximax - ximin), where ximax and ximin are the maximum and minimum values for the ith attribute in the training set.

After pre-processing the data, the distance between each instance in test data with each instance in train data is calculated. An priority queue is generated for each

instance in test data to maintain the top k instances in train data of high similarity. Similarity between two instance is:

$$\frac{1}{\sqrt{w_1(a_1-b_1)^2+w_2(a_2-b_2)^2+w_3(1-sim(a_3,b_3))+w_4(a_4-b_4)^2+\cdots+w_n(a_n-b_n)^2}}$$

The similarity calculation takes the difference of the attribute values (between the test and train) in the denominator. It take the difference for normalized numeric attributes value but for non- numeric attributes, use the corresponding similarity value from the similarity matrix.In the above formula, the attribute 3 is nonnumeric. The train data vector is a(a1 a2 ... an) and test data vector is b(b1b2.....bn). And the weight is also added (as w1, w2...wn) for each attribute to weight their importance.

After getting the k nearest instances for one test instance, the label could be decided based on the k nearest instances. Each potential discrete label is traversed to get a highest score for that test instance. The formula of the score calculation is:

$$Value_{obj}(y)=\text{argmax}_{Ci}\left[\Sigma_{x_j \in kNN(y)}\ sim(x_j,y) * \delta\left(\ class(x_j)\ ,\ C_i\ \right)\right]$$

where,

$$\delta\left(\ class(x_j)\ ,\ C_i\ \right) = 1 \quad \text{if,} \quad class(x_j) = C_i$$

$$\delta\left(\ class(xj)\ ,\ Ci\ \right) = 0 \quad \text{if,} \quad class(xj) \neq Ci$$

The above formula is for discrete label calculation. For real number labeled data, we use kvote = average of the label of k nearest neighbors to get label for test instance.

Decision Tree:

Working: Decision tree is predictive learning technique which using certain attributes of an instance and provides the value of a target attribute of the same instance. In other words it provides a mapping of attributes of an instance to some target attribute. In the learning phase the decision tree algorithm takes as input a set of input records and creates a tree of rules. Each input variable represents a particular input attribute and leaf node represents target attributes value. The path from root to leaf form a path and any input records to have a particular target value will have a path from root to leaf. The input record must satisfy all the conditions on the tree path to belong to the class or have the value represented by the leaf node. The basis of prediction for decision tree is based on entropy of the attributes. At each level the attribute which provides the most information gain is chosen to separate the records in different category and same happens at each level of the tree. Following are the terminologies which are used in a construction of decision tree.

S is a sample of training examples
• p⊕ is the proportion of positive examples in S
• p* is the proportion of negative examples in S
• Entropy measures the impurity of S

$$\text{Entropy}(S) \equiv H(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p^* \log_2 p^*$$

H(S) = 0 if sample is pure (all + or all -), H(S) = 1 bit if $p_{\oplus} = p^* = 0.5$

Information Gain

The information gain is based on the decrease in entropy after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches).

The attribute with largest information gain is chosen

The dataset is then split on the different attributes. The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split. The resulting entropy is subtracted from the entropy before the split. The result is the Information Gain, or decrease in entropy.

Relative advantages and disadvantages of the two, such as data requirements, applicability to different business situations, etc. Or, are they totally interchangeable?

	Advantages	Disadvantages
KNN	• Simple implementation, good option in small data set • Can be used to predict both continuous and discrete values • Weight attributes can improve accuracy significantly	• Large storage requirements and computation expensive when the number of train example is large. • Lazy learner
Decision Tree	• Interpretable • Generate rules	• Over-fitting • Can only used to predict discrete value • Perform less when many attribute interaction are present

The main advantage of Decision tree is interpretable. Decision trees are "white boxes" in the sense that the acquired knowledge can be expressed in a readable form, while KNN are generally black box i.e. you cannot read the acquired knowledge in a comprehensible way. The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.

Another advantage of decision tree is that it generates rules from a single tree with the ability to transform multiple decision trees into a set of classification rules and it can be used to better scale up rule generation in terms of size and number of rules and learning time.

Advantages of KNN are, it has simple implementation and works well on basic recognition problem, especially in small data set. Another advantage of KNN is that it can predict continuous value as the average value of k nearest instances. While decision tree, more specify, C4.5 can only be used to predict discrete class value.

Decision tree can be used to handle both numerical and categorical data. When applying this technique to real valued data, each numeric value is likely to be unique, the result may be huge number of splits each of which create a subset of just one data point. In C4.5 in order to make splits we need some threshold values. We need to find optimal threshold value which will give best split. There can also be multiple threshold values to create multiple splits for the same attribute.

Decision-tree learners can create over-complex trees that do not generalise the data well. This is called over-fitting. Mechanisms such as pruning (not currently supported), setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem. Besides, when we have large amount of data, over-fitting is less an issue.

As decision trees use the “divide and conquer” method, they tend to perform well if a few highly relevant attributes exist, but less so if many complex interactions are present. For KNN, weight array can be passed in to weight more for those highly relevant attributes and less for those irrelevant attributes.

As for KNN algorithm, it might be slow since in large number of train data examples since It computes the distance and sort all training data in the prediction function. To store all train data set, it also requires large data storage in large data set.

Another disadvantage of KNN is that it is a lazy learner. No model created in training phase, the classification is obtained in computation of prediction. It doesn’t learn anything from train data set, which results in not being robust to noisy data.

b. Other classifiers to consider in future Data Mining opportunities (from readings)

Method	Training Data Requirements	Random Noise Tolerance	Scalability (atts + data)	Quality of Prediction	Explanatory Power	Popularity of Usage
Rule	Sparse	None	Good	Good, brittle	Very clear	Med, stable

Induction						
Naïve Bayes	Medium-Dense	Some-Good	Medium	Medium/cat	Partial	Med, declining
Regression	Medium-Dense	Some-Good	Good	Good/both	Partial-Poor	High, stable
SVM	Medium-Dense	Some-Good	Good-Excellent	Very good/cat	Poor	Med, increasing
Neural Nets	Dense	Good	Poor-Medium	Good/cat	Poor	Med, declining

Rule Induction

- It is one of the important techniques of machine learning.
- The rules are extracted here from a set of observations.
- Data from which rules are induced are usually presented in a form similar to a table in which cases (or examples) are labels (or names) for rows and variables are labeled as attributes and a decision.

Naïve Bayes

- These are set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features.
- They are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem.
- It assumes that the value of a particular feature is independent of the value of any other feature, given the class variable.

Logistic Regression

- It is a type of probabilistic statistical classification model.
- It can be used to predict binary response and class labels based on one or more features or attributes.
- The relationship between dependent and one or more independent variablesby using probability scores.

SVM- Support Vector Machines

- It analyze data and recognize patterns, used for classification and regression analysis.
- It builds a model that assigns new examples into one category or the other.

Neural networks

- ANN can be used to estimate the functions that can depend on a large number of inputs or are unknown.
- They can be used to compute values from inputs and for pattern recognition.
- They can be used to infer a function from observations where the data is too complex.

III. How to set up a Data Mining Problem

a. What do you need before you get started?

Any Data mining task comprises of five major steps

1. Extract Transform Load the data in a database or flat files to be analyzed
2. Use the data to build set of rules or prediction models
3. Provide the model with new data to make the prediction based on past data.

Before starting the problem we need what we need as the outcome of our task to be. What kind of problem do we need to solve using Data Mining. These are the basic question that one should ask before starting a data mining task. The approach of doing data mining for different kind of outcomes is different so understanding of the goal and final result is necessary before starting the data mining task.

b. How do you select a classifier?

Since our train data is small, and knn is a good choice for small data set as a classifier, and it has simple implementation, we decide to choose KNN algorithm.

For decision tree, ID 3 is a classical algorithm to generate a decision tree from a data set. It only handles discrete attributes. However, there are some numeric attributes to consider in our data set. So we use C4.5 decision tree instead, because it can handle both continuous and discrete attribute, and it can be pruned after the tree creation.

c. What do you do in the Training Phase?

In KNN algorithm, the training phase of the algorithm stores every attributes, including the feature vectors and class labels of training data set. For discrete attributes, we also predefine their similarity matrix for future similarity calculation. For continuous attributes, we go through all instances in train data set and get max attribute and min attribute for each continuous attribute, and then normalize each attribute to scale them into the range of [0,1], using (xi - ximin)/(ximax - ximin), where ximax and ximin are the maximum and minimum values for the ith attribute in the training set.

Decision tree is a supervised learning technique. To build a model it needs to be trained first from a known set of data. The goal is to make smallest tree possible with high accuracy of prediction. In the training phase the algorithm is given a set of input data which has all the data. This known data is called training data. The algorithm uses this data to build a tree. Entropy of know attributes is used to find the split points of the tree. The decision of having a split points is made by using based on the information gain which is inversely related to entropy. At every level of decision tree a split is made for the attributes on a threshold value.

What do you do in the runtime/Testing Phase?

For KNN algorithm, in testing phase, we pass the test data set, the weight array for each attributes, similarity matrix to the Knn class, and calculate k nearest instance for each test data according to the weight and similarity value from the matrix.

In testing phase of decision tree we used K fold cross validation. This is a model validation technique to find what will be the accuracy of the model on an unknown data set. Testing phase is used to evaluate the accuracy of the model created by the algorithm. In testing phase we divide the data in dataset into K non-overlapping subsets (folds), train a model using K-1 folds and predict its performance using the fold you left out. This you do for each possible combination of folds (first leave 1st fold out, then 2nd then kth and train with the remaining folds). After finishing you estimate the mean performance of all folds. This mean performance is the accuracy of the model.

References:

http://en.wikipedia.org/wiki/Cross-validation_(statistics)

http://www.eecs.berkeley.edu/~russell/classes/cs194/f11/lectures/CS194%20Fall%202011%20Lecture%2008.pdf

http://decisiveminds.com/easy-step-decision-making-process/4675

http://www.selba.org/EngTaster/Social/Facilitation/ThreeStagesinDecisionConsensus.html

http://www.ise.bgu.ac.il/faculty/liorr/hbchap9.pdf

http://research.microsoft.com/pubs/65569/splits.pdf

https://www.cs.princeton.edu/courses/archive/spring07/cos424/papers/mitchell-dectrees.pdf

http://www.sciencedirect.com/science/article/pii/0004370295000607

http://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart

http://www.sussex.ac.uk/Users/christ/crs/ml/lec05a.html

http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.htm

http://www.saedsayad.com/decision_tree.htm

http://homepages.inf.ed.ac.uk/amos/lfd/lectures/decisiontree.pdf

http://genome.tugraz.at/MedicalInformatics2/dtree.pdf

http://www.csie.ntnu.edu.tw/~bbailey/ClassifTrainingEval.pdf

http://en.wikipedia.org/wiki/Decision_tree_learning

http://courses.cs.washington.edu/courses/csep521/07wi/prj/leonardo_fabricio.pdf

http://cse3521.artifice.cc/k-nn.html

http://www.cs.cmu.edu/afs/cs/project/theo-20/www/mlbook/ch8.pdf