Personalized Recommendation System for Students by Grade Prediction

Pavan Kotha
113050010
Master Thesis

Under the guidance of
Prof. D.B PHATAK

Computer Science Department,
IIT Bombay
Predicting Student Performance
Classification
Recommendation System
Conclusion

Outline

- Motivation
- Problem Statement
- Prediction of Endsem Marks and Grades
- Theoretical Model for personalized recommendation system
- Conclusion and Future Work
Different learners - diff modes of learning, educational backgrounds, previous experience, intelligence

Different people process knowledge in different ways. So, different people will relate to a particular learning resource in different ways.

Human instructors can learn which style of presentation suits which learner and adjust their mode of presentation.

Different learners need to focus on different material to achieve the same learning objective.

Need to classify into different Learner types (Achiever, Struggler etc)
Problem Statement

- Proposed learning system: Gives learning support based on predicted grade of student.
- Different learning proposals provided to students in feedback according to student target class.
ML in education- to study the data available in the educational field, bring out the hidden knowledge from it

Instructors can identify weak students, help them to score better marks in endsem by giving additional practise questions.

Appropriate measures - taken to improve their performance in endsem i.e provide customized references so that he can read those materials and improve his grade.

Eg: Additional customized practise questions, different text books for different learners etc
Learning from Data

- ML branch of AI - deals with study of systems that can learn from data.
- ML focuses on prediction based on known properties learned from the training data. Eg: Email (spam, non-spam)
- So learn from one offering of CS101 course and predict marks and grades of another offering
Statistical data is represented in terms of tuples
- Data consists of many attributes
- Training data \( D = \{(x_1, y_1), \ldots, (x_N, y_N)\} \)
- Target Attribute to be predicted for test data
- Training goal: To learn function \( f(x) \rightarrow y \) s.t prediction error on unseen instances is small
- Learner extracts about the distribution.
Proposed System

Figure: Schematic diagram of system
Attributes taken: assignment, quiz, midsem. Exclude project.

Don’t have any information about the distribution of data.

We can find mean, variance and correlation between attributes from the given training data.

Relationship between the attributes. eg: Midsem good marks, mostly endsem also good marks.

Need to capture such a relationship from training data.
Linear Model in Single Variable

- Linear function: \( Y = a + bX \) which can best predict the target variable
- \( Y \) - endsem marks, \( X \) assignment or quiz or midsem marks.
- How to find parameters \( a, b \) from training data?
- To obtain the best linear function choose 'a' and 'b' s.t prediction error is minimised.

**Optimization Objective:** Minimise \( E[(Y - (a + bX))^2] \)

\[
a = E[Y] - bE[X], \quad b = \rho \frac{\sigma_y}{\sigma_x}
\]
Figure: Best line fit for data points

- Four \((x, y)\) data points: \((1, 6), (2, 5), (3, 7)\) and \((4, 10)\)
We need to find a line $y = a + bx$ that best fits these four points (training data).

Best approximation: minimize the sum of squared differences (i.e. error) between the predicted data values and their corresponding actual values.

Error function
\[
S(a, b) = [6 - (a + b)]^2 + [5 - (a + 2b)]^2 + [7 - (a + 3b)]^2 + [10 - (a + 4b)]^2
\]

Equate partial derivatives of $S(a, b)$ with respect to $a, b$ to zero. We get $a = 3.5, b = 1.4$

Best fit line is $y = 3.5 + 1.4x$
endsem = f(midsem)
endsem = f(assignment)
endsem = f(quiz)

Taken average of three endsem marks
Predicting Student Performance
Classification
Recommendation System
Conclusion
Need for Prediction
Regression

Linear Model in Multiple Variables

- \( \text{endsem} = f(\text{quiz, assignment, midsem}) \)
- Mathematically \( \sum_{j=1}^{3} (X_{ij}b_j + b_0) = y_i, \ (i = 1, 2, \ldots, m) \)
- In matrix form \( Xb = y \) where \( X = \begin{bmatrix} 1 & X_{12} & X_{13} & X_{14} \\ 1 & X_{22} & X_{23} & X_{24} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & X_{m2} & X_{m3} & X_{m4} \end{bmatrix} \)

\[
\begin{bmatrix}
  b_0 \\
  b_1 \\
  b_2 \\
  b_3 \\
\end{bmatrix}
\quad \begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_m \\
\end{bmatrix}
\]

- Overdetermined system with \( m \) linear equations and 4 unknowns \((b_0, b_1, b_2, b_3)\).
Goal: Find the coefficients $\mathbf{b}$ which fit the equations best in the sense of minimising mean square error ($= \mathbb{E}[(\text{actual value} - \text{predicted value})^2]$).

$\hat{\mathbf{b}} = \arg \min_{\mathbf{b}} S(\mathbf{b})$, where $\hat{\mathbf{b}}$ refers to optimal value of $\mathbf{b}$ where the objective function $S$ is given by $S(\mathbf{b}) = \sum_{i=1}^{m} |y_i - \sum_{j=1}^{3} (X_{ij}b_j + b_0)|^2 = \| \mathbf{y} - \mathbf{Xb} \|^2$.

Differentiating results in $\hat{\mathbf{b}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

$endsemmarks = b_0 + b_1 assignment + b_2 quiz + b_3 midsem$
Good value for the RMS depends on data we are trying to model or estimate.

Analysis result: 5% to 20% error is good estimate.

Our error is 5%. So we have achieved good estimates of endsem marks using our model.

<table>
<thead>
<tr>
<th>Endsem Prediction</th>
<th>Root Mean Square Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Model in Single Variable</td>
<td>5.09</td>
</tr>
<tr>
<td>Linear Model in Multiple Variable</td>
<td>5.1</td>
</tr>
</tbody>
</table>

Table: Root Mean Square Error
Classification - type of prediction where predicted variable is binary or categorical.

Classification methods—decision trees, nearest neighbour, LDA, SVM - applied on educational data for predicting grade in examination.

Classifier—mathematical function implemented by classification algorithm that maps input data to a category.

No single classifier works best on all given problems.

Determining a suitable classifier for a given problem is more an art than a science.

Type of classifier to be chosen depends on problem we are trying to solve.
Predicting Student Performance
Classification
Recommendation System
Conclusion

Decision Trees
Nearest Neighbour
LDA
SVM
Classifiers combination

Dataset

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

**Figure**: Training Dataset for playing tennis

[Tom Mitchell]
Play Tennis Dataset

- Target attribute to be predicted is Play Tennis
- Classification Problem bcz Play Tennis={yes,no}
- Learned function is represented by a decision tree
- Learned trees can also be represented by if-then rules
Decision Tree for Dataset

Figure: Learned Decision Tree

[Tom Mitchell]
Decision tree classifies whether morning of a day is suitable for playing tennis or not

Attributes of morning of day

- Outlook = Sunny
- Temperature = Hot
- Humidity = High
- Wind = Strong

PlayTennis = no

Given a new tuple we predict whether he can play Tennis or not by traversing decision tree.
Statistical test determines how well each attribute alone classifies the training examples.

Best attribute is selected to test at the root node of the tree.

Training examples are split to the branch corresponding to the example’s value for this attribute.
Information Gain

- Attribute to be tested at a node for classifying examples depends on information gain (statistical property) of attribute.

- Information gain measures how well a given attribute separates the training examples according to their target classification.
S: Training Examples, Target attribute: c different values.

Entropy of S relative to this c-wise classification:

- Entropy(S) = \( \sum_{i=1}^{c} p_i \log_2 p_i \) where \( p_i \) is proportion of S belonging to class i.
- Information gain of attribute A
- Gain(S, A) = Entropy(S) - \( \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \) Entropy(S_v) where Values(A) is the set of all possible values for attribute A
- \( S_v \) is the subset of S for which attribute A has value v.
In given dataset, \( S \) is a collection of 14 examples.

- Attribute Wind has the values Weak or Strong.
- 9 positive (i.e., play tennis = yes) and 5 negative examples denoted as \([9+,5-]\).
- Of these 14 examples, 6 of the positive and 2 of the negative examples have Wind = Weak.
Information gain from attribute Wind is calculated as follows:

Values(Wind)\(\Rightarrow\) Weak, Strong

\(S = [9+, 5-]\)

\(S_{\text{weak}} = [6+, 2-]\)

\(S_{\text{strong}} = [3+, 3-]\)

Gain(S, Wind)\(\Rightarrow\) Entropy(S) - \(\sum_{v \in \text{Weak, Strong}} \frac{|S_v|}{|S|} \text{Entropy}(S_v)\)

\(= \text{Entropy}(S) - \left(\frac{8}{14}\right) \text{Entropy}(S_{\text{weak}}) - \left(\frac{6}{14}\right) \text{Entropy}(S_{\text{strong}})\)

\(= 0.940 - \left(\frac{8}{14}\right) 0.811 - \left(\frac{6}{14}\right) 1.00\)

\(= 0.048\)

Information gain from attribute Wind is calculated as follows:

Values(Wind)\(\Rightarrow\) Weak, Strong

\(S = [9+, 5-]\)

\(S_{\text{weak}} = [6+, 2-]\)

\(S_{\text{strong}} = [3+, 3-]\)

Gain(S, Wind)\(\Rightarrow\) Entropy(S) - \(\sum_{v \in \text{Weak, Strong}} \frac{|S_v|}{|S|} \text{Entropy}(S_v)\)

\(= \text{Entropy}(S) - \left(\frac{8}{14}\right) \text{Entropy}(S_{\text{weak}}) - \left(\frac{6}{14}\right) \text{Entropy}(S_{\text{strong}})\)

\(= 0.940 - \left(\frac{8}{14}\right) 0.811 - \left(\frac{6}{14}\right) 1.00\)

\(= 0.048\)
Information Gains of attributes

- Information gain: Selects best attribute at each step in constructing the tree.
- Information gain values for all four attributes are:
  - $\text{Gain}(S, \text{Outlook}) = 0.246$
  - $\text{Gain}(S, \text{Humidity}) = 0.151$
  - $\text{Gain}(S, \text{Wind}) = 0.048$
  - $\text{Gain}(S, \text{Temperature}) = 0.029$
Constructing Tree

- Outlook attribute provides the best prediction of the target attribute.
- Outlook is selected as the decision attribute for the root node.
- Training examples - splitted to the branch corresponding to the example’s value.
- Every example Outlook = Overcast is a positive example.
- Outlook = Sunny and Outlook = Rain do not have all positive.
- Repeat procedure using only the training examples associated with that node.
Predicting Student Performance

Classification

Recommendation System

Conclusion

Decision Trees

Nearest Neighbour

LDA

SVM

Classifiers combination

Partial Decision Tree

Figure: Partial Decision Tree

[Tom Mitchell]
Grades prediction using Decision Trees

- Showing marks of other students not good-not done in US
- Build decision tree using historical information(model works per prof)
- Attributes are assignment, quiz, midsem, project, predicted endsem marks.
- Student yet to write endsem-know his grade approx
- Improve performance in endsem and improve their grade
Accuracy using Decision Tree Classifier

<table>
<thead>
<tr>
<th>Classifier Name</th>
<th>Accurate Accuracy (Single Variable)</th>
<th>Approx. Accuracy (Single Variable)</th>
<th>Accurate Accuracy (Multi Variable)</th>
<th>Approx. Accuracy (Multi Variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree</td>
<td>65.32</td>
<td>98.24</td>
<td>65.14</td>
<td>98.42</td>
</tr>
</tbody>
</table>

**Table:** Accuracy using Decision Trees Classifier

**Figure:** Accuracy using Decision Trees
Nearest Neighbour Algorithm

- Non-parametric method for classifying objects based on closest training examples.
- An object is assigned to the class of its nearest neighbor
- Neighbors taken from a set of objects for which the correct classification is known
- Training examples: Vectors in multidimensional feature space each with a class label.
- Test point is classified by assigning the label which is closest to that point in the training examples. (Euclidean distance as distance metric.)
Example for Nearest Neighbour Algorithm

Figure: Example for Nearest Neighbour
Accuracy achieved using Nearest Neighbour Algorithm

<table>
<thead>
<tr>
<th>Classifier Name</th>
<th>Accurate Accuracy (Single Variable)</th>
<th>Approx. Accuracy (Single Variable)</th>
<th>Accurate Accuracy (Multi Variable)</th>
<th>Approx. Accuracy (Multi Variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest Neighbour</td>
<td>66.37</td>
<td>97.89</td>
<td>66.02</td>
<td>98.59</td>
</tr>
</tbody>
</table>

**Table:** Accuracy using Nearest Neighbour Classifier

**Figure:** Accuracy using Nearest Neighbour
Nearest Neighbour Algorithm with Clustering

- Drawback: mis-classification occurs when class distribution is skewed
- Distribution of grades is not uniform. Grades like BB and BC dominates
- Examples of a more frequent class tend to dominate the prediction of the new example
- Reduced the data set by replacing a cluster of similar grades, regardless of their density in the original training data with single point which is its cluster center.
- Nearest Neighbour is then applied to the reduced data set.
Accuracy using Nearest Neighbour Algorithm with Clustering

<table>
<thead>
<tr>
<th>Classifier Name</th>
<th>Accurate Accuracy(Single Variable)</th>
<th>Approx Accuracy(Single Variable)</th>
<th>Accurate Accuracy(Multi Variable)</th>
<th>Approx Accuracy(Multi Variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest Neighbour Cluster</td>
<td>66.55</td>
<td>98.77</td>
<td>63.56</td>
<td>98.94</td>
</tr>
</tbody>
</table>

**Table:** Accuracy using Nearest Neighbour Cluster Classifier

**Figure:** Accuracy using Nearest Neighbour Cluster
Linear Discriminant Analysis

- LDA - find a linear combination of features which separates two or more classes of objects
- Training set: Attributes : \( \vec{x} \). Known class label “y”
- Classification problem :Find a good predictor for the class “y” given only an observation \( \vec{x} \)
- LDA assumes cpd \( P(\vec{x}|y = 0) \sim N(\vec{x}, \mu_0, \Sigma) \) where \( \mu_0 \) is a vector containing means of d-attributes, taking samples whose target label is 0.
- \( P(\vec{x}|y = 1) \sim N(\vec{x}, \mu_1, \Sigma) \) where \( \mu_1 \) is a vector containing means of d-attributes, taking samples whose target label is 1.
Covariance is a $d \times d$ matrix where $d$ refers to number of attributes.

To classify any new observation $\vec{x}$, we assign label $y=0$ for this observation if

$$P(y = 0|\vec{x}) > P(y = 1|\vec{x})$$

Solving above equation results in linear decision boundary.

For multi class LDA, grade is $\max(P(y = i|\vec{x}), i = 1..c)$
Table: Accuracy using LDAC Classifier

<table>
<thead>
<tr>
<th>Classifier Name</th>
<th>Accurate Accuracy (Single Variable)</th>
<th>Approx Accuracy (Single Variable)</th>
<th>Accurate Accuracy (Multi Variable)</th>
<th>Approx Accuracy (Multi Variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAC</td>
<td>63.73</td>
<td>98.06</td>
<td>64.26</td>
<td>98.59</td>
</tr>
</tbody>
</table>

Figure: Accuracy using LDAC
Support Vector Machines

- SVM based on the concept of decision planes that define decision boundaries.
- A decision plane separates objects which are having different class memberships.
- Many hyperplanes exist that classify the data.
- SVM tries to find a hyperplane with maximum margin.
- Most classification tasks - complex decision boundaries.
Support Vector Machines Example

**Figure:** Linearly Seperable Data

**Figure:** Non Linearly Seperable Data
Training Data: \( \mathcal{D} = \{(x_i, y_i) | x_i \in \mathbb{R}^p, y_i \in \{-1, 1\}\}_{i=1}^n \)

Find hyperplane that separates two classes and should maximize the margin.

- equation of hyperplane (eg. line1) is of form \( w \cdot x - b = 0 \)
- hyperplane2 - \( w \cdot x - b = 1 \) (assume: green objects has class label=1)
- hyperplane3 - \( w \cdot x - b = -1 \) (assume: red objects has class label=-1)
- distance between two hyperplanes 2 and 3 is \( \frac{2}{\|w\|} \).
- For maximum margin - minimize \( \|w\| \).
As training set is linearly separable we add the constraints

\[ \mathbf{w} \cdot \mathbf{x}_i - b \geq 1 \quad \text{for } \mathbf{x}_i \text{ belongs to green class} \]
\[ \mathbf{w} \cdot \mathbf{x}_i - b \leq -1 \quad \text{for } \mathbf{x}_i \text{ belongs to red class.} \]

Combining the above two equations, we get

\[ y_i (\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1, \quad \text{for all } 1 \leq i \leq n. \]

The optimization objective for SVM then becomes

Minimize (in \( \mathbf{w}, b \)) \( \frac{1}{2} \mathbf{w}^T \mathbf{w} \)

subject to (for any \( i = 1, \ldots, n \)) constraints \( y_i (\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1 \)
Working procedure of SVM

Figure: Linearly Separable in Feature Space
Introduce non-negative slack variables, $\xi_i \geq 0$, one slack variable for each training point.

- It measures the degree of misclassification of the data point $x_i$.
- Slack variable is 0 for data points that are on or inside the margin
- $\xi_i = |y_i - (w \cdot x_i - b)|$ for other points.
- Data point on decision boundary - $w \cdot x_i - b = 0$, hence $\xi_i = 1$
- Points which have $\xi_i \geq 1$ are misclassified
Support Vector Machines for Linearly non separable data

The optimization objective then becomes

\[ \text{minimize } \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \]

subject to constraints

\[ y_i (w^T \Phi(x_i) - b) \geq 1 - \xi_i, \text{ and } \xi_i \geq 0, i = 1, \cdots, N \]
### Table: Accuracy using SVM Classifier

<table>
<thead>
<tr>
<th>Classifier Name</th>
<th>Accurate Accuracy (Single Variable)</th>
<th>Approx. Accuracy (Single Variable)</th>
<th>Accurate Accuracy (Multi Variable)</th>
<th>Approx. Accuracy (Multi Variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVMLinear</td>
<td>64.44</td>
<td>98.59</td>
<td>64.79</td>
<td>98.94</td>
</tr>
<tr>
<td>SVMRbf</td>
<td>61.8</td>
<td>97.89</td>
<td>66.37</td>
<td>98.42</td>
</tr>
</tbody>
</table>

### Figure: Accuracy using Linear Kernel Function
**Figure**: Accuracy using Gaussian Kernel Function
Combination of Classifiers

- Previous models: Single classifiers like decision trees, LDA, NN, SVM
- To improve the accuracy - used combination of classifiers
- Used many classifiers - not possible to come up with a single classifier that can give good results.
- Optimal classifier is dependent on problem domain
Combining multiple classifiers (CMC)

- Method 1: Choose classifier which has least error rate on given dataset
  - Nearest Neighbour algorithm with Clustering similar classes has least error rate
- Method 2: Class getting the maximum votes.
- If many classifiers predict student fails, then we assign “fail” label to student.
Combination of Classifiers

Table: Accuracy using Combination of Classifiers (Method 2)

<table>
<thead>
<tr>
<th>Classifiers Combination</th>
<th>Accurate Accuracy (Single Variable)</th>
<th>Approx Accuracy (Single Variable)</th>
<th>Accurate Accuracy (Multi Variable)</th>
<th>Approx Accuracy (Multi Variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>63.38</td>
<td>97.89</td>
<td>65.49</td>
<td>98.59</td>
</tr>
</tbody>
</table>

Figure: Accuracy using Classifiers Combination
Difficult Classification Problem

- Tried various ways to improve accuracy, but we always got 63 to 67%
- Analysis Result: $p=11$ — difficult classification problem
- Even best methods achieve around 40% errors on test data. So we achieved good accuracy. [18]
- Only way to improve accuracy is to reduce the number of class labels.
Target Classes Reduced

- Grouped them into four classes
- “high” grades - AB, AA, AP
- “middle” grades - CC, BC, BB,
- “low” grades - DD, CD
- “fail” - FR and FF.
Accuracy when target classes reduced

<table>
<thead>
<tr>
<th>Classifier Name</th>
<th>Accurate Accuracy (Single Variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest Neighbour</td>
<td>89.43</td>
</tr>
</tbody>
</table>

Table: Accuracy using Nearest Neighbour Classifier when Labels Reduced

- Accuracy is improved to 89% as compared to 66% when there are 10 class labels.
- Clearly depicts that accuracy depends on number of target classes
## Comparison Among Different Classifiers

<table>
<thead>
<tr>
<th>Classifier Name</th>
<th>Accurate Accuracy(Single Variable)</th>
<th>Approx Accuracy(Single Variable)</th>
<th>Accurate Accuracy(Multi Variable)</th>
<th>Approx Accuracy(Multi Variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree</td>
<td>65.32</td>
<td>98.24</td>
<td>65.14</td>
<td>98.42</td>
</tr>
<tr>
<td>SVM Linear</td>
<td>64.44</td>
<td>98.59</td>
<td>64.79</td>
<td>98.94</td>
</tr>
<tr>
<td>SVM Rbf</td>
<td>61.8</td>
<td>97.89</td>
<td>66.37</td>
<td>98.42</td>
</tr>
<tr>
<td>Nearest Neighbour</td>
<td>66.37</td>
<td>97.89</td>
<td>66.02</td>
<td>98.59</td>
</tr>
<tr>
<td>Nearest Neighbour Cluster</td>
<td>66.55</td>
<td>98.77</td>
<td>63.56</td>
<td>98.94</td>
</tr>
<tr>
<td>Idac</td>
<td>63.73</td>
<td>98.06</td>
<td>64.26</td>
<td>98.59</td>
</tr>
<tr>
<td>Classifiers Combination</td>
<td>63.38</td>
<td>97.89</td>
<td>65.49</td>
<td>98.59</td>
</tr>
</tbody>
</table>

**Table:** Comparison Among Different Classifiers
Motivation for Recommendation System

- No two students are identical, they learn at different rates, different educational backgrounds, different intellectual capabilities, different modes of learning.
- Need to design a real-time recommendation system which captures the characteristics of each student.
- Need to ensure that every student progresses through the course material so that he maximizes his learning.
- It is an immensely complex task. No personal care on students
Theoretical Model for Recommendation System

- Customized suggestions: Need to know student mastery level on topic
- Consider question difficulty level eg. 2 students got 5/10 marks
- Test questions - no uniform characteristics.
- Model question level performance instead of aggregate perf.
- Give different internal score - says mastery level in the topic
Extract info each question provides about a student.
prob of a correct response to a test question is a mathematical function of parameters such as a person's abilities and question characteristics (such as difficulty, guessability, and specific to topic).
Model helps to better understand how a students performance on tests relates to his ability
learn about the user, update its parameters about the user to suggest him properly and make him to achieve his learning objectives.
Theoretical Model for Recommendation System contd

- Student struggles with a particular set of questions - our system should know where that particular student’s weaknesses lie in relation to the concepts assessed by those questions.
- System should then deliver content to increase the student’s proficiency on those concepts.
- Personalized learning cannot be done just based on total marks.
- Attributes for each question: difficulty level, time taken to answer, number of attempts to get correct answer, marks he secured, probability of guessing the answer.
Theoretical Example

Figure: Dependency Graph for C Language Course
Theoretical Example contd

**Figure:** DAG for Internal Score
Theoretical Example contd

- Calculate Internal score of the student in that topic - mastery on the topic.
- Each node in dependency graph, attributes are mastery level on topic, intelligence level of student, difficulty level of topic etc.
- Classification algorithms can be applied on this data. Classify him as weak or average or intelligent at topic level and accordingly suggest him personalized learning resources.
- Topic level classification: new video tutorials on this topic, personalized materials, personalized problem sets etc.
Conclusion

- Predicting endsem marks, grade of student apriori, providing customized learning materials - improve performance in endsem, grade.
- Learners - different backgrounds, previous experience, so different learners need to focus on different material to achieve the same learning objective.
- Proposed learning system gives learning support based on individual learning characteristic.
- Different learning proposals provided to students in feedback - learners learning rate.
Future Work

- Personalized resource recommendation - many granularities: problem level, topic level, course level
- Redirect him to a short segment video lecture - problem level
- Continuously learn from the system –not single point learning
- Struggling with particular questions: Identify them, how similar students improved

**Figure:** Proposed System
References I


Evgeniya Georgieva et.al, *Design of an e-Learning Content Visualization Module*, 3rd E-Learning Conference, 2006


Stephnaos Mavromoustakos et.al, *Human And Technological Issues In The E-Learning Engineering Process*, Journal Article in ResearchGate


Mingming Zhou et.al *Data Mining and Student e-Learning Profiles* International Conference on E-Business and E-Government, 2010

S. Anupama Kumar et.al *Efficiency of Decision Trees in Predicting Students Academic Performance*, CS And IT-CSCP 2011


www.it.iitb.ac.in/ekshiksha/(referred on 31st Aug 2012)

The Elements of Statistical Learning by Friedman, Hastie, Tibshirani printed in 2001, pg 85