Week 1, video 3:

Classifiers, Part 1
Prediction

- Develop a model which can infer a single aspect of the data (predicted variable) from some combination of other aspects of the data (predictor variables)

- Sometimes used to predict the future

- Sometimes used to make inferences about the present
Classification

- There is something you want to predict ("the label")
- The thing you want to predict is categorical
  - The answer is one of a set of categories, not a number

- CORRECT/WRONG (sometimes expressed as 0,1)
  - We’ll talk about this specific problem later in the course within latent knowledge estimation

- HELP REQUEST/WORKED EXAMPLE REQUEST/ATTEMPT TO SOLVE

- WILL DROP OUT/WON’T DROP OUT

- WILL ENROLL IN MOOC A,B,C,D,E,F, or G
Where do those labels come from?

- In-software performance
- School records
- Test data
- Survey data
- Field observations or video coding
- Text replays
Classification

- Associated with each label are a set of “features”, which maybe you can use to predict the label

<table>
<thead>
<tr>
<th>Skill</th>
<th>pknow</th>
<th>time</th>
<th>totalactions</th>
<th>right</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTERINGGIVEN</td>
<td>0.704</td>
<td>9</td>
<td>1</td>
<td>WRONG</td>
</tr>
<tr>
<td>ENTERINGGIVEN</td>
<td>0.502</td>
<td>10</td>
<td>2</td>
<td>RIGHT</td>
</tr>
<tr>
<td>USEDIFFNUM</td>
<td>0.049</td>
<td>6</td>
<td>1</td>
<td>WRONG</td>
</tr>
<tr>
<td>ENTERINGGIVEN</td>
<td>0.967</td>
<td>7</td>
<td>3</td>
<td>RIGHT</td>
</tr>
<tr>
<td>REMOVECOEFF</td>
<td>0.792</td>
<td>16</td>
<td>1</td>
<td>WRONG</td>
</tr>
<tr>
<td>REMOVECOEFF</td>
<td>0.792</td>
<td>13</td>
<td>2</td>
<td>RIGHT</td>
</tr>
<tr>
<td>USEDIFFNUM</td>
<td>0.073</td>
<td>5</td>
<td>2</td>
<td>RIGHT</td>
</tr>
</tbody>
</table>

....
The basic idea of a classifier is to determine which features, in which combination, can predict the label.

<table>
<thead>
<tr>
<th>Skill</th>
<th>pknow</th>
<th>time</th>
<th>totalactions</th>
<th>right</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTERINGGIVEN</td>
<td>0.704</td>
<td>9</td>
<td>1</td>
<td>WRONG</td>
</tr>
<tr>
<td>ENTERINGGIVEN</td>
<td>0.502</td>
<td>10</td>
<td>2</td>
<td>RIGHT</td>
</tr>
<tr>
<td>USEDIFFNUM</td>
<td>0.049</td>
<td>6</td>
<td>1</td>
<td>WRONG</td>
</tr>
<tr>
<td>ENTERINGGIVEN</td>
<td>0.967</td>
<td>7</td>
<td>3</td>
<td>RIGHT</td>
</tr>
<tr>
<td>REMOVECOEFF</td>
<td>0.792</td>
<td>16</td>
<td>1</td>
<td>WRONG</td>
</tr>
<tr>
<td>REMOVECOEFF</td>
<td>0.792</td>
<td>13</td>
<td>2</td>
<td>RIGHT</td>
</tr>
<tr>
<td>USEDIFFNUM</td>
<td>0.073</td>
<td>5</td>
<td>2</td>
<td>RIGHT</td>
</tr>
<tr>
<td>....</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Classifiers

- There are hundreds of classification algorithms
- A good data mining package will have many implementations
  - RapidMiner
  - SAS Enterprise Miner
  - Weka
  - KEEL
Classification

- Of course, usually there are more than 4 features
- And more than 7 actions/data points
Domain-Specificity

- Specific algorithms work better for specific domains and problems

- We often have hunches for why that is

- But it’s more in the realm of “lore” than really “engineering”
Some algorithms I find useful

- Step Regression
- Logistic Regression
- J48/C4.5 Decision Trees
- JRip Decision Rules
- K* Instance-Based Classifiers

- There are many others!
Step Regression

- *Not step-wise regression*

- Used for binary classification (0,1)
Step Regression

- Fits a linear regression function
  - (as discussed in previous class)
  - with an arbitrary cut-off

- Selects parameters
- Assigns a weight to each parameter
- Computes a numerical value

- Then all values below 0.5 are treated as 0, and all values \( \geq 0.5 \) are treated as 1
Example

- $Y = 0.5a + 0.7b - 0.2c + 0.4d + 0.3$
- Cut-off 0.5

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>
Example

- \( Y = 0.5a + 0.7b - 0.2c + 0.4d + 0.3 \)
- Cut-off 0.5

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>
Example

- $Y = 0.5a + 0.7b - 0.2c + 0.4d + 0.3$
- Cut-off 0.5

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td></td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
Example

- $Y = 0.5a + 0.7b - 0.2c + 0.4d + 0.3$
- Cut-off 0.5

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>
Quiz

- Y = 0.5a + 0.7b – 0.2c + 0.4d + 0.3
- Cut-off 0.5

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Note

- Step regression is used in RapidMiner by using linear regression with binary data.

- Other functions in different packages.
Step regression: should you use it?

- Step regression is not preferred by statisticians due to lack of closed-form expression.
  
- But often does better in EDM, due to lower over-fitting.
Logistic Regression

- Another algorithm for binary classification (0,1)
Logistic Regression

- Given a specific set of values of predictor variables

- Fits logistic function to data to find out the frequency/odds of a specific value of the dependent variable
Logistic Regression
Logistic Regression

\[ m = a_0 + a_1v_1 + a_2v_2 + a_3v_3 + a_4v_4 \ldots \]

\[ p(m) = \frac{1}{1 + e^{-m}} \]
Logistic Regression

\[ m = 0.2A + 0.3B \]

\[ p(m) = \frac{1}{1 + e^{-m}} \]
Logistic Regression

\[ m = 0.2A + 0.3B \]

\[ p(m) = \frac{1}{1 + e^{-m}} \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>M</th>
<th>P(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Logistic Regression

\[ m = 0.2A + 0.3B \]

\[
p(m) = \frac{1}{1 + e^{-m}}
\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>M</th>
<th>P(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
</tr>
</tbody>
</table>
Logistic Regression

\[ m = 0.2A + 0.3B \]

\[ p(m) = \frac{1}{1 + e^{-m}} \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>M</th>
<th>P(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.73</td>
</tr>
</tbody>
</table>
Logistic Regression

\[ m = 0.2A + 0.3B \]

\[
p(m) = \frac{1}{1 + e^{-m}}
\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>M</th>
<th>P(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>0.27</td>
</tr>
</tbody>
</table>
Logistic Regression

\[ m = 0.2A + 0.3B \]

\[
p(m) = \frac{1}{1 + e^{-m}}
\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>M</th>
<th>P(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.88</td>
</tr>
</tbody>
</table>
Logistic Regression

\[ m = 0.2A + 0.3B \]

\[
p(m) = \frac{1}{1 + e^{-m}}
\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>M</th>
<th>P(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0.95</td>
</tr>
</tbody>
</table>
Logistic Regression

\[ m = 0.2A + 0.3B \]

\[
p(m) = \frac{1}{1 + e^{-m}}
\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>M</th>
<th>P(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>~1</td>
</tr>
</tbody>
</table>
Relatively conservative

- Thanks to simple functional form, is a relatively conservative algorithm
  - I’ll explain this in more detail later in the course
Good for

- Cases where changes in value of predictor variables have predictable effects on probability of predicted variable class

- \( m = 0.2A + 0.3B + 0.5C \)

- Higher A always leads to higher probability
  - But there are some data sets where this isn’t true!
What about interaction effects?

- A = Bad
- B = Bad
- A+B = Good
What about interaction effects?

- Ineffective Educational Software = Bad
- Off-Task Behavior = Bad
- Ineffective Educational Software PLUS Off-Task Behavior = Good
Logistic and Step Regression are good when interactions are not particularly common

- Can be given interaction effects through automated feature distillation
  - We’ll discuss this later

- But is not particularly optimal for this
What about interaction effects?

- Fast Responses + Material Student Already Knows - > Associated with Better Learning

- Fast Responses + Material Student Does not Know - > Associated with Worse Learning
Decision Trees

- An approach that explicitly deals with interaction effects
Decision Tree

### Skill

<table>
<thead>
<tr>
<th>knowledge</th>
<th>time</th>
<th>totalactions</th>
<th>right?</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPUTESLOPE</td>
<td>0.544</td>
<td>9</td>
<td>1</td>
</tr>
</tbody>
</table>
Skill: knowledge, time, totalactions, right?

COMPUTESLOPE 0.544 9 1 RIGHT
Decision Tree

Skill knowledge time totalactions right?
COMPUTESLOPE 0.444 9 1 ?
Decision Tree Algorithms

- There are several

- I usually use J48, which is an open-source re-implementation in Weka/RapidMiner of C4.5 (Quinlan, 1993)
J48/C4.5

- Can handle both numerical and categorical predictor variables
  - Tries to find optimal split in numerical variables

- Repeatedly looks for variable which best splits the data in terms of predictive power for each variable

- Later prunes out branches that turn out to have low predictive power

- Note that different branches can have different features!
Can be adjusted...

- To split based on more or less evidence
- To prune based on more or less predictive power
Relatively conservative

- Thanks to pruning step, is a relatively conservative algorithm
  - We’ll discuss conservatism in a later class
Good when data has natural splits
Good when multi-level interactions are common
Good when same construct can be arrived at in multiple ways

- A student is likely to drop out of college when he
  - Starts assignments early but lacks prerequisites

- OR when he
  - Starts assignments the day they’re due
Later Lectures

- More classification algorithms
- Goodness metrics for comparing classifiers
- Validating classifiers
- What does it mean for a classifier to be conservative?
Next Lecture

- Building regressors and classifiers in RapidMiner