# Week 3 Video 3

## Feature Engineering

# Feature Engineering

# Feature Engineering

- Up until this point in the class, we've talked about building and validating prediction models
- Models that infer a predicted variable from predictor variables

# Where the Predicted Variable Comes From

- A couple lectures ago, we went into a little more detail about where the predicted variable can come from

# Where the Predictor Variables Come From

- Where do the predictor variables come from?
- Do they fall out of the sky?
- Do they come from the Office for Predictor Variables in Washington, DC?

# Feature Engineering

- The art of creating predictor variables
- A major topic in its own right

# Why is it important?

- Feature engineering is the least well-studied part of the process of developing prediction models
  - But it's arguably the most important part
  - Your model will never be any good if your features (predictors) aren't very good

# Why is it important?

- It is an art, it is human-driven design
- It involves lore rather than well-known and validated principles
- It is hard!

# The Big Idea

- How can we take the voluminous, ill-formed, and yet under-specified data that we now have in education
- And shape it into a reasonable set of variables
- In an efficient, effective, and predictive way?

# A process in its own right

1. Brainstorming features
2. Deciding what features to create
3. Creating the features
4. Studying the impact of features on model goodness
5. Iterating on features if useful
6. Go to 3 (or 1)

# Brainstorming Features

- Can be more or less formal

# IDEO tips for Brainstorming

1. **Defer judgment**
2. **Encourage wild ideas**
3. **Build on the ideas of others**
4. **Stay focused on the topic**
5. **One conversation at a time**
6. **Be visual**
7. **Go for quantity**

http://www.openideo.com/fieldnotes/openideo-team-notes/seven-tips-on-better-brainstorming

# Building on the Ideas of Others

- Doesn't just have to be people nearby
- There's a huge literature out there of features people have tried and what has worked, or failed to work, for a range of problems
- Read papers from researchers working on similar problems, and see what you can use
- Some folks have also tried crowd-sourcing (Veeramacheneni et al., 2014)

# Brainstorming Features

- On hard projects, my research group often meets as a team over pizza and beer to brainstorm

- On easier projects, one person brainstorms solo
  - And then often discusses their features with another person, who offers further suggestions

# Deciding what features to create

- There is never infinite time
- A trade-off between the effort to create a feature and how likely it is to be useful
  - "How likely it is to be useful" – the best you can do is to
    - Look at whether similar features have been useful for similar problems
    - Use your best intuition
- Worth biasing in favor of features that are different than anything else you've tried before
  - Explores a different part of the space

# Creating features

- Excel – Really good for prototyping features
- Distillation Code – The scalable solution… but harder to check yourself or explore

# Some useful tools in Excel

- Pivot Tables – great for aggregating data, and getting the average, min, max, stdev
- Vlookup – great for translating from aggregations (student-level data, for instance) back to action-level data

- Example in this week's Walkthrough

# Further resources

- http://www.howtogeek.com/howto/13780/using-vlookup-in-excel/
- http://www.excel-easy.com/data-analysis/pivot-tables.html
- http://spreadsheets.about.com/od/datamanagementinexcel/ss/8912pivot_table.htm

# Other useful things you can do in Excel

- Counts-so-far
- Counts-last-n-actions
- Differentiating first and subsequent attempts
- Ratios between events of interest
- Cut-off based features

# Feature Iteration

- Sometimes when a feature looks like it might be good
- It's worth iterating on that feature, trying close variants to see if they do better

# Example

- You have a feature “slow actions after hints” (cf. Shih, Koedinger, & Scheines, 2008)
- You define “slow action” as an action taking over 20 seconds
- What if 30 seconds is a better cut-off?

# Ways to accomplish this…

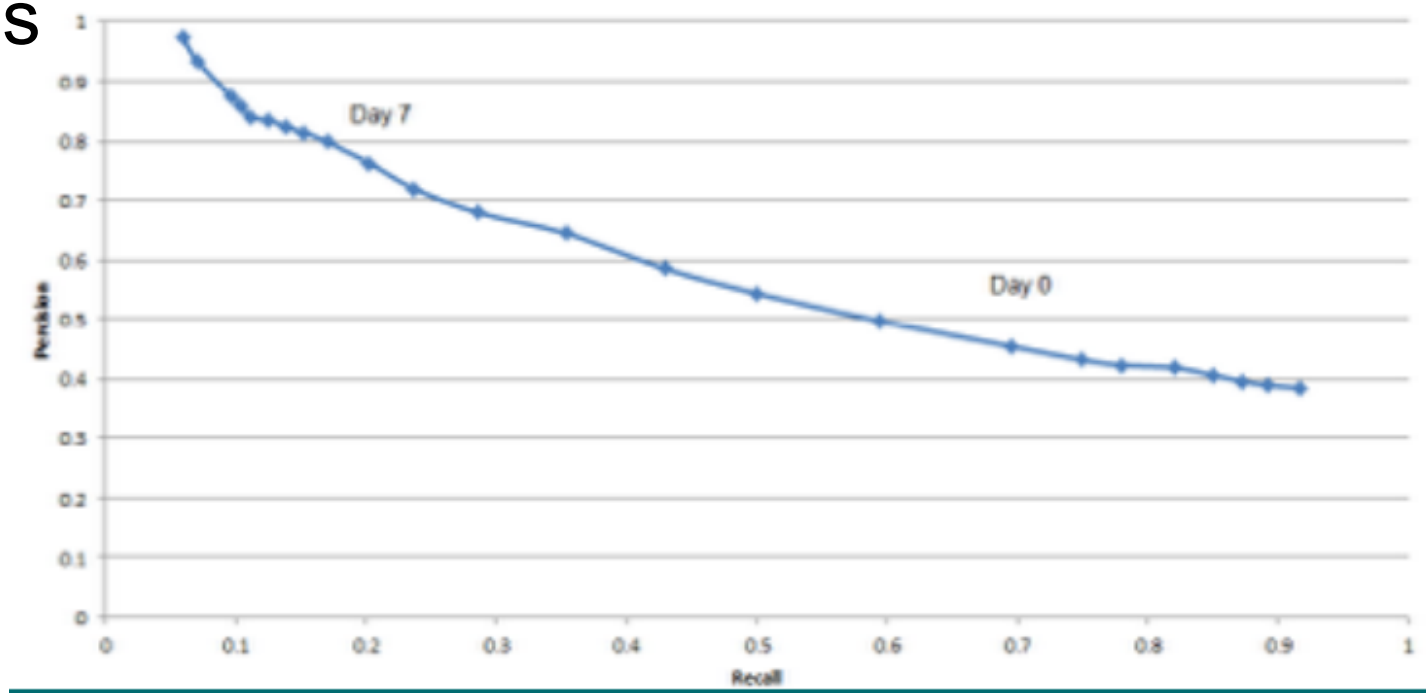
- By hand
- Programming (Java? Matlab?)
- Excel Equation Solver

# Details of features matter

- For example, the same feature can have different impact depending on context

# (Baker et al., 2015)

- Whether a student has opened their e-textbook predicts whether they fail the course
- But with totally different precision and recall on the first day of the class versus the 7th day of the class



# Excel Equation Solver Tutorials

- http://office.microsoft.com/en-us/excel-help/define-and-solve-a-problem-by-using-solver-HP010072691.aspx
- http://www.youtube.com/watch?v=K4QkLA3sT1o
- One tip: multistart option avoids local minima (that can sometimes block the solver from even getting started)

# A few thoughts

# Does feature engineering over-fit?

- It can
- Which is why it's useful to remember
- The true test of a model is whether it works on entirely unseen data

- If you iterate a lot and use cross-validated goodness
- Then the true test of your model will be either a held-out data set or newly-collected data later on

# Feature Engineering

- Your features come from somewhere
- You can take a standard set of variables or pre-existing variables
  - No question it's faster
- But thinking about your variables is likely to lead to better models
  - Actually evidence for this, see (Sao Pedro et al., 2012)

# Next Lecture

- Automated feature generation and selection