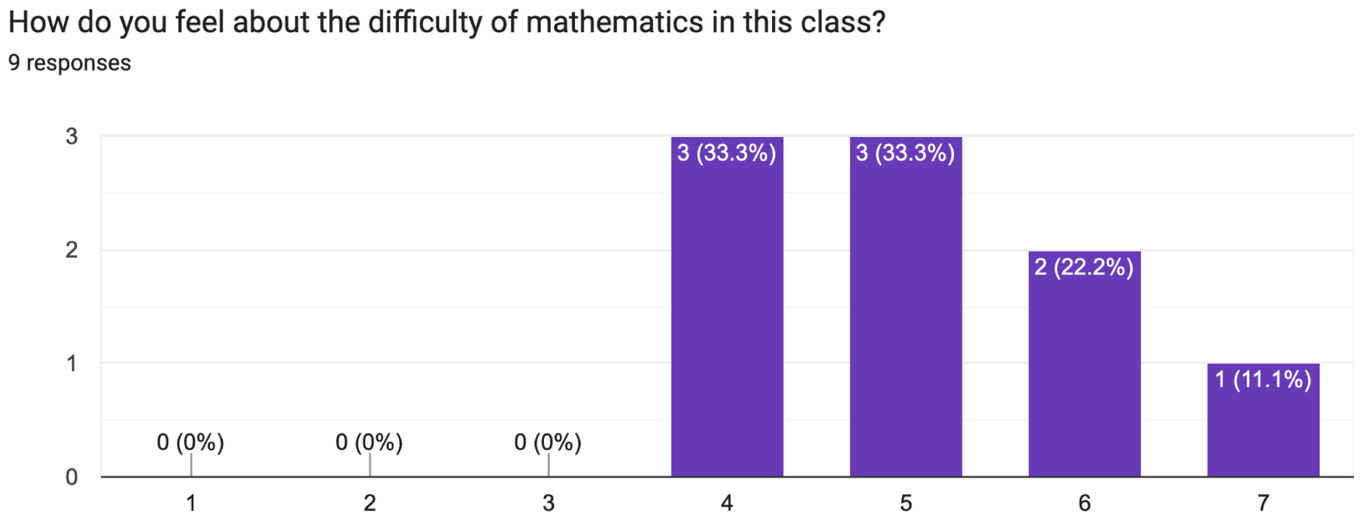
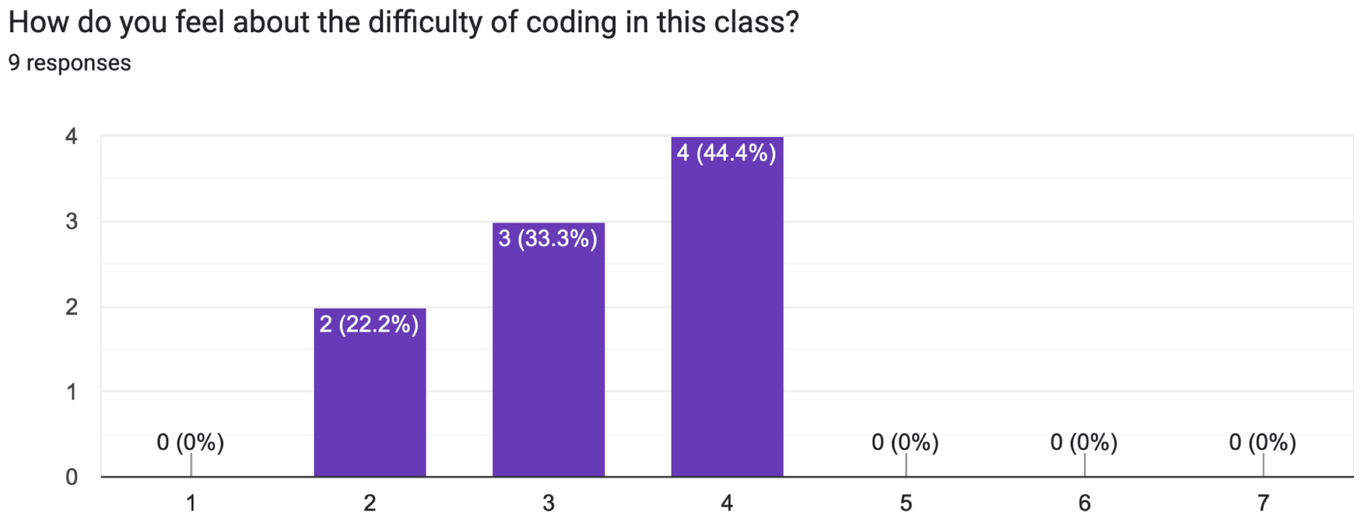
Survey results

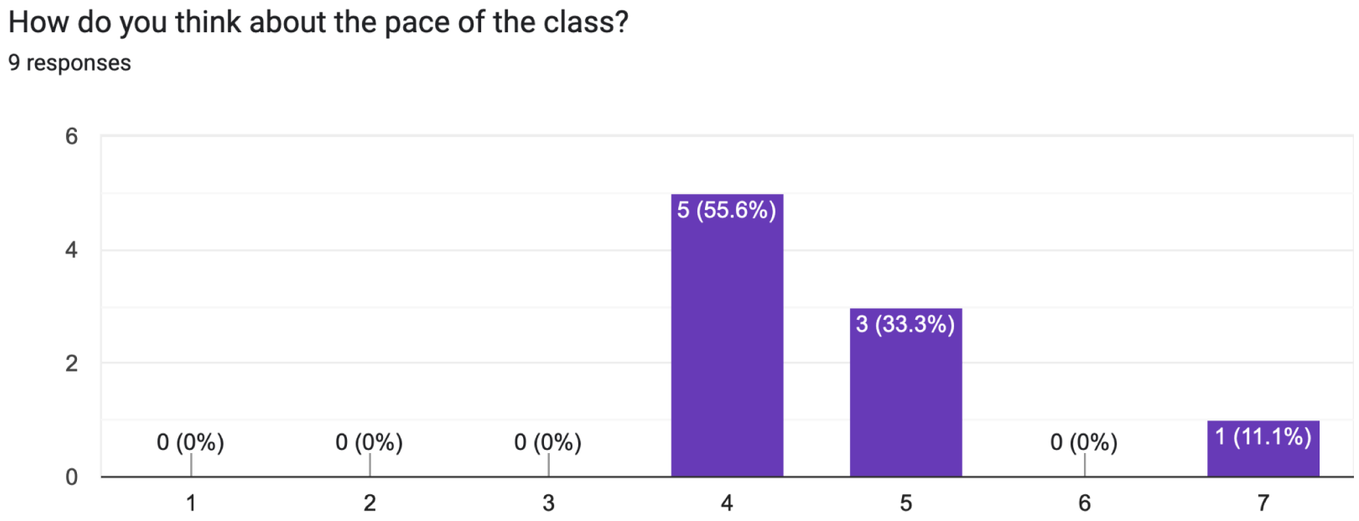
Interpretation. Math is challenging to most people



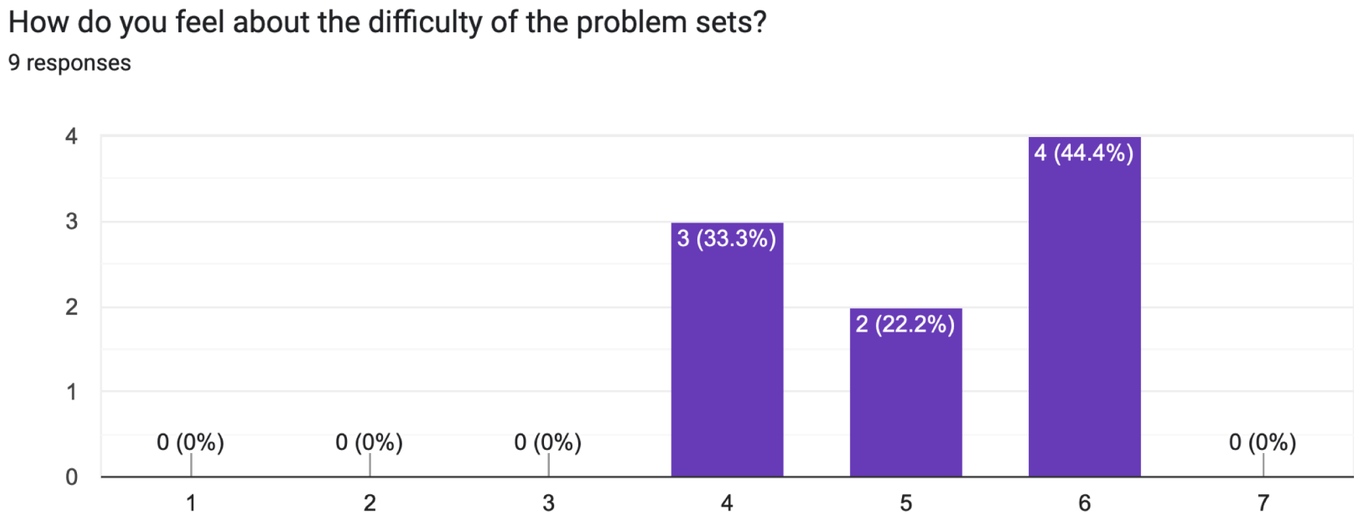
Interpretation. Coding is easy to most of the people



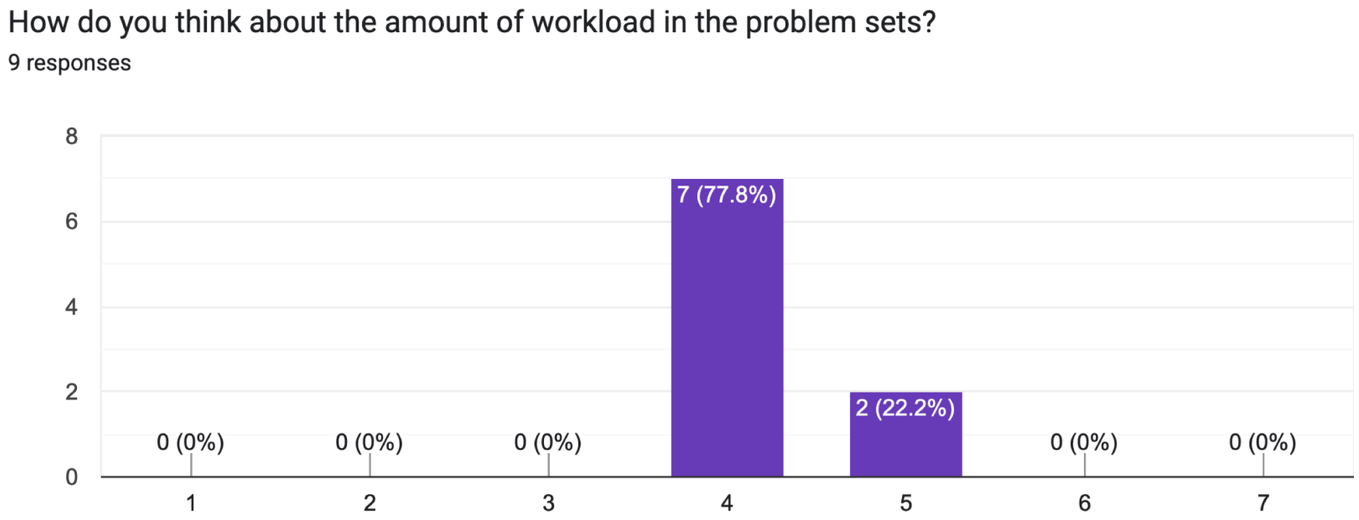
Interpretation. The class is relatively dense



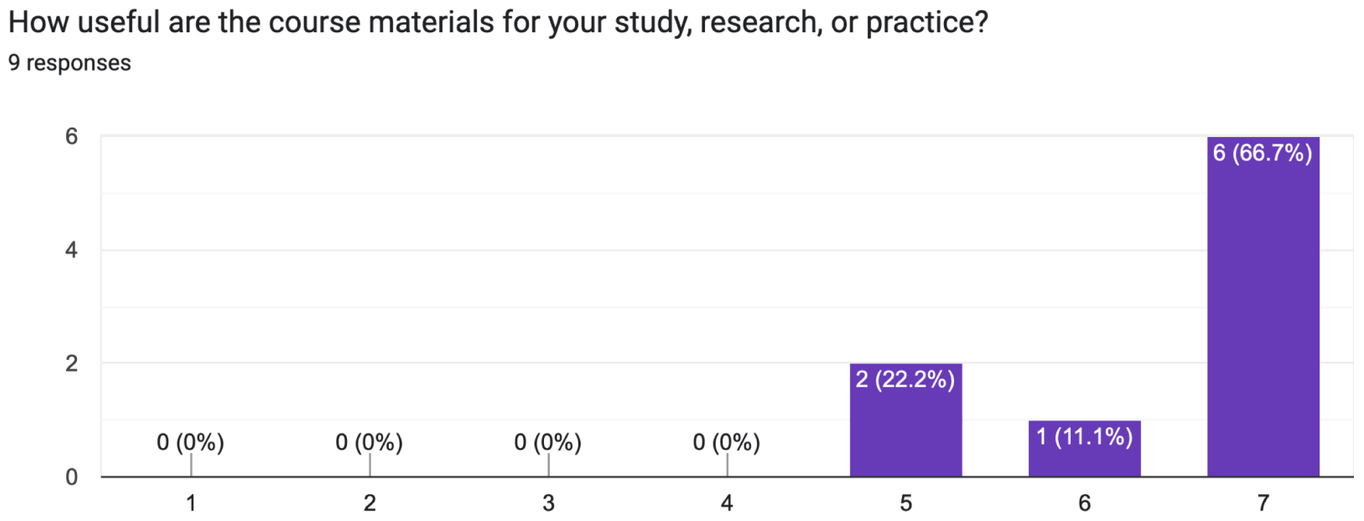
Interpretation. Problem sets are relatively challenging



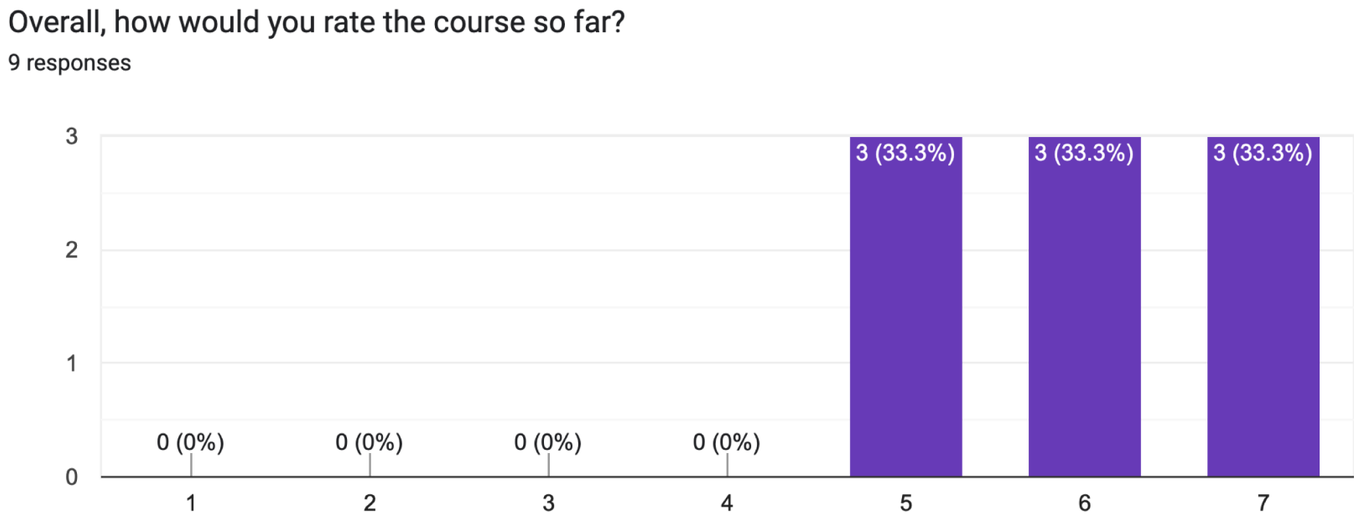
Interpretation. Workload in psets is roughly right (maybe a bit too much)



Interpretation. Overall useful



Interpretation. Overall good, but still some room to improve the class



Useful things in the class (ranking based on key word frequency)

1. Labs & computational applications
2. Network analysis
3. Statistical analysis
4. General introduction into the terminology
5. General diagram
6. Other key words: all topics; collaboration & teamwork; availability of notes and videos

How to improve?

1. Dilute the course contents. (e.g. “Urban statistical analysis, Urban network analysis, and Machine learning in cities, could benefit from being extended into full semester courses”)
2. More labs & computational applications, and simplify mathematics (e.g. “don’t need to go very deep into some of the concepts, since we will use the packages anyways”)
3. More generous grading. (e.g. “other opportunities to pick up points”)
4. Uploading lecture slides before class.
5. More introduction into research frontier (e.g., “the latest developments in the field”, “recommend some extensive readings.”)

Actions

1. Diluting the next six lectures
   - Adding two review sessions to consolidate concepts and help you to connect to applications.
   - Adding two work sessions to help with course projects.
   - Introducing three guest lectures to present the efforts on the research frontier.
   - As a result, I will have to cut many contents from the previous syllabus.
   - If you are interested, please wait for the whole UA program (intro, intermediate, and potentially advanced UA) in the next year.

2. Adjusting the weights between lectures and practice (labs, work sessions, etc.).
   - Limiting the number of slides in lectures.
   - Putting more weights in the labs.
   - Emphasizing more urban applications (e.g. review sessions)

3. Others: grading more generously, providing some literature into the research frontier, uploading lecture slides before class, etc.

Week	Dates		Guest lectures	Lab sessions	Work sessions	Psets	Project
Module 3: Machine learning in cities							
8	Mar 7	supervised learning: classifications and regression revisited		Y			proposal guideline out
NA	Mar 14	no class (spring break)					
9	Mar 21	supervised learning: regularization		Y	Y	hw 3 out, hw2 due	
10	Mar 28	mid-term presentation and review session					mid-term presentation; proposal due on Mar 31
11	Apr 4	supervised learning: other classifiers and guest lecture	Y	Y			final report guideline out
12	Apr 11	unsupervised learning or fairness with an analytical perspective and guest lecture	Y	Y		hw 3 due	
Module 4							
13	Apr 18	work session and guest lecture	Y		Y		
14	Apr 25	final presentation & review session & course evaluation					final presentation; report due on Apr 28

Readjusting the syllabus

- Two work sessions.
  - Q&A to support your project
- Two review sessions
  - Knowledge structure & application
- HW3: Regularization and supervised learning.
- Three guest lectures.
- **Voting**: for April 11 – do you want to learn unsupervised learning or analytical fairness?
  - Unsupervised learning – textbook stuff
  - Analytical fairness – my own view

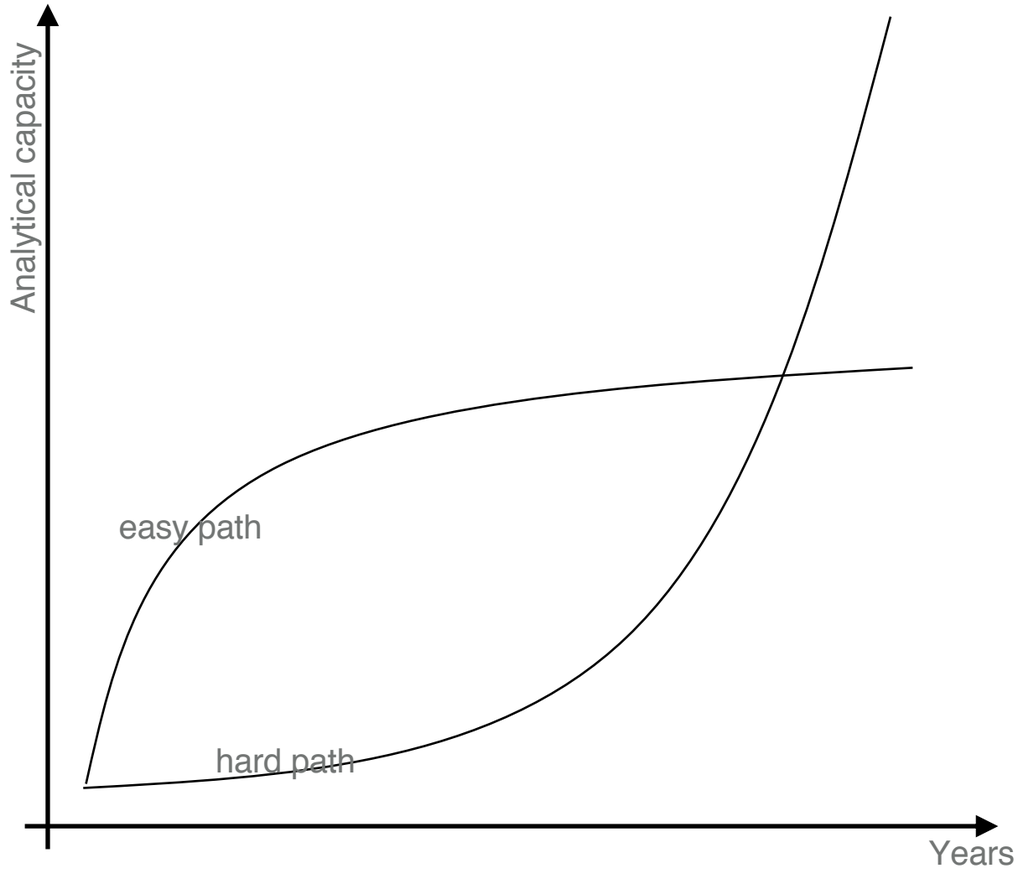
Some underlying reasoning in designing this class

Quote from the survey: “we don’t need to go very deep into some of the (math) concepts, since we will use the packages anyways.”

Your capacity will be limited by your weakest point in the three pillars.

Some underlying reasoning in designing this class

Quote from the survey: “we don’t need to go very deep into some of the (math) concepts, since we will use the packages anyways.”



Two learning paths

- Easy path: focusing on coding & applications only (Program level: directly learning the urban analytics courses and skipping basic math & coding courses.)
- Hard path: balancing math & coding & applications (Program level: learning the basic math & coding courses and then taking the urban analytics courses)

The easy path is hard. The hard path is easy.

Questions & Discussions

- Maybe the easy path should be designed for master students, while the hard one for PhD students?
- Maybe we should separate the easy and hard paths in the curriculum design for the Master of Urban Analytics program?

URP 6931. Introduction to Urban Analytics

Lecture 09: Supervised learning - regularization

Instructor: Shenhao Wang
Assistant Professor, Director of Urban AI Lab
Department of Urban and Regional Planning
University of Florida

Supervised Learning

1

Revisiting logistic regression
(Review)

2

Regularization

3

Diverse supervised learning algorithms

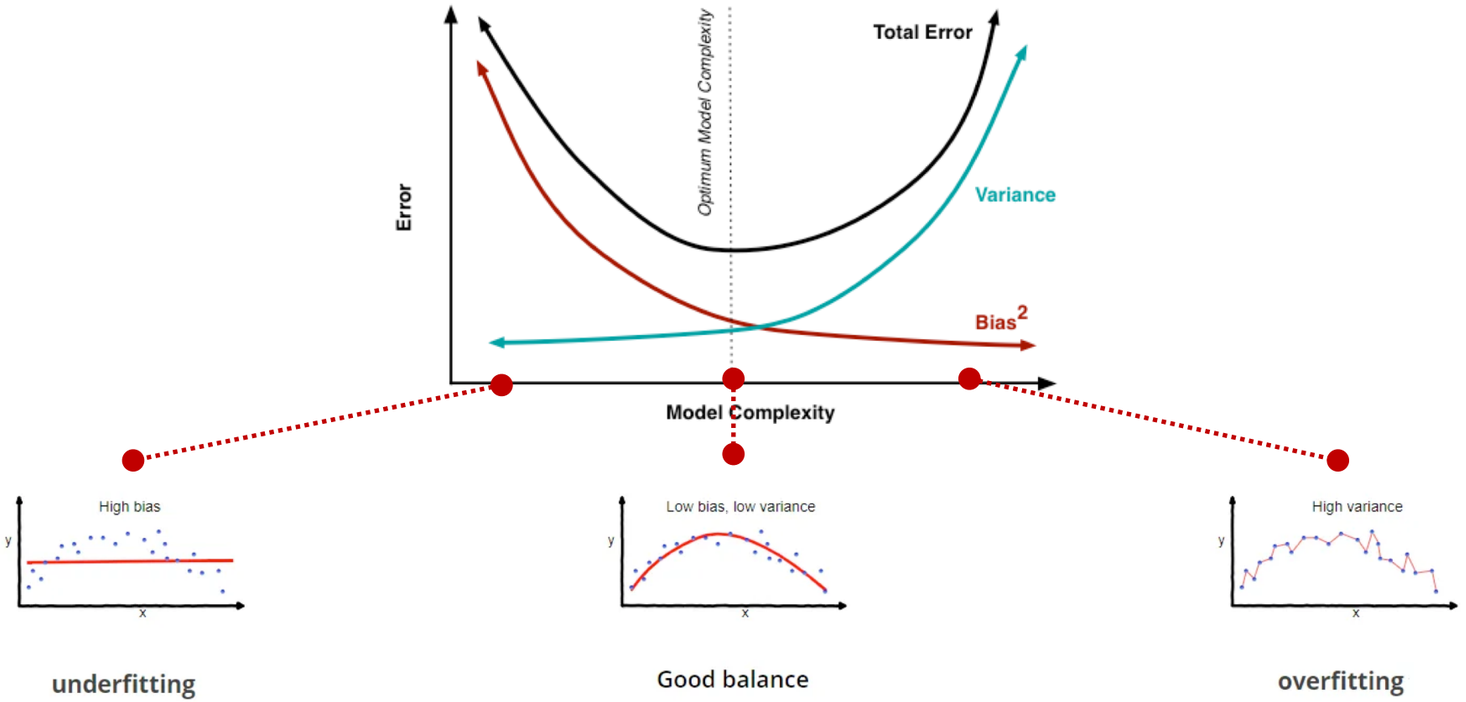
Minor differences

Major differences

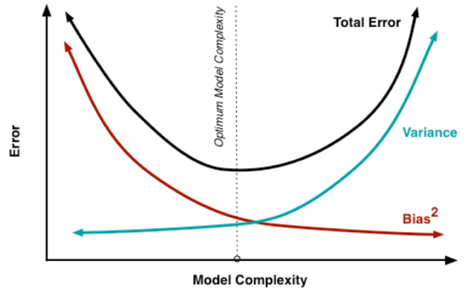
Part 2. Regularization

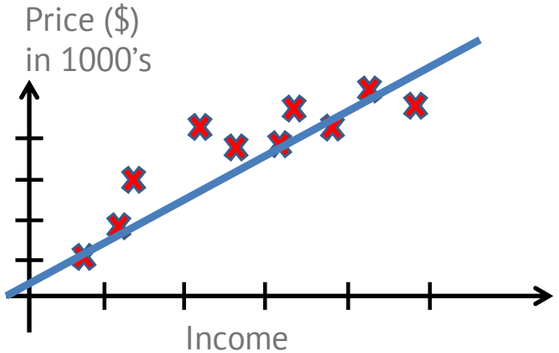
Why training + testing split?

It is because we are worried about not only a too simple but also too complex model – bias & variance tradeoff



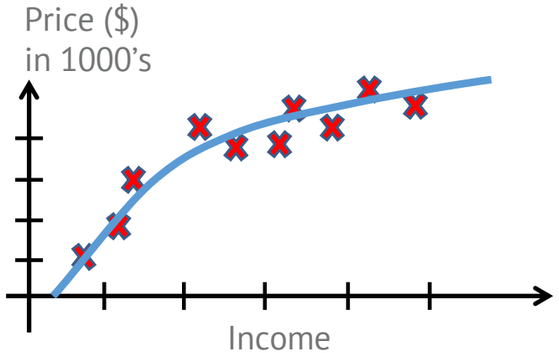
Underfitting and overfitting in linear regression





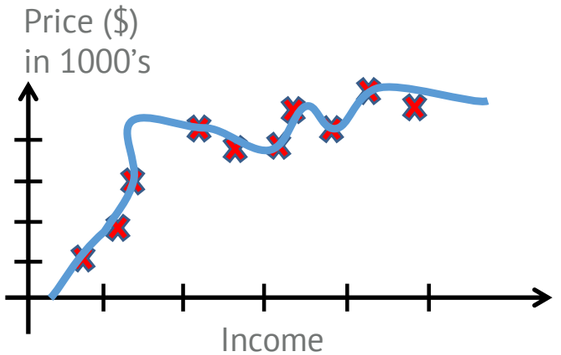
$h_\theta(x) = \theta_0 + \theta_1 x$

Underfitting



$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$

Just right



$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \cdots$

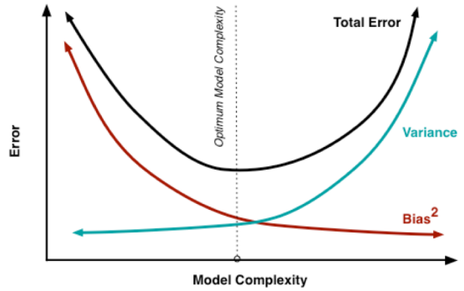
Overfitting

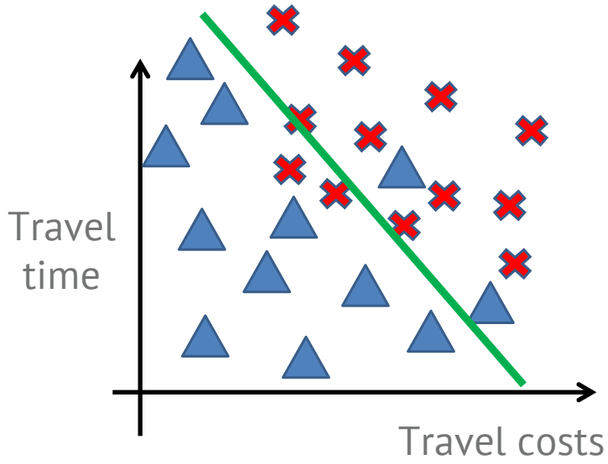
High Bias — Enrich the model →

← Simplify the model — High variance

Slide credit: Andrew Ng

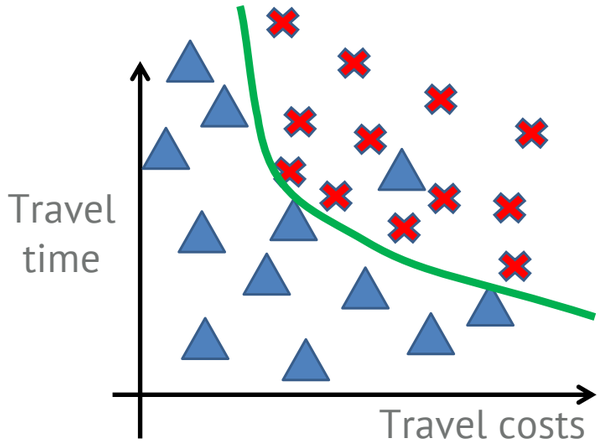
Underfitting and overfitting in classification





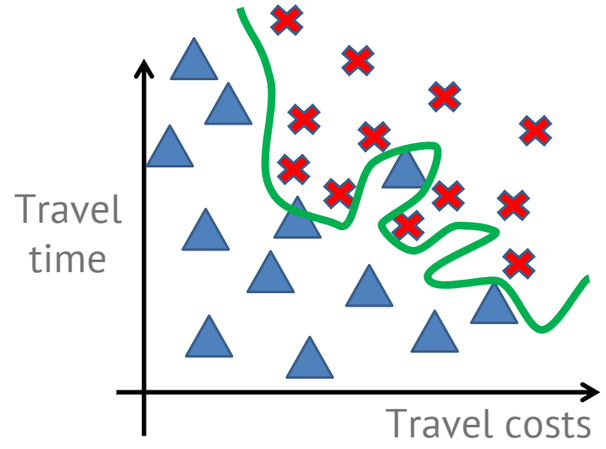
$h_\theta(x) = g(\theta_0 + \theta_1 x + \theta_2 x_2)$

Underfitting



$h_\theta(x) = g(\theta_0 + \theta_1 x + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$

Just right



$h_\theta(x) = g(\theta_0 + \theta_1 x + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2 + \theta_6 x_1^3 x_2 + \theta_7 x_1 x_2^3 + \cdots)$

Overfitting

High Bias — Enrich the model → ← Simplify the model — High variance

Slide credit: Andrew Ng

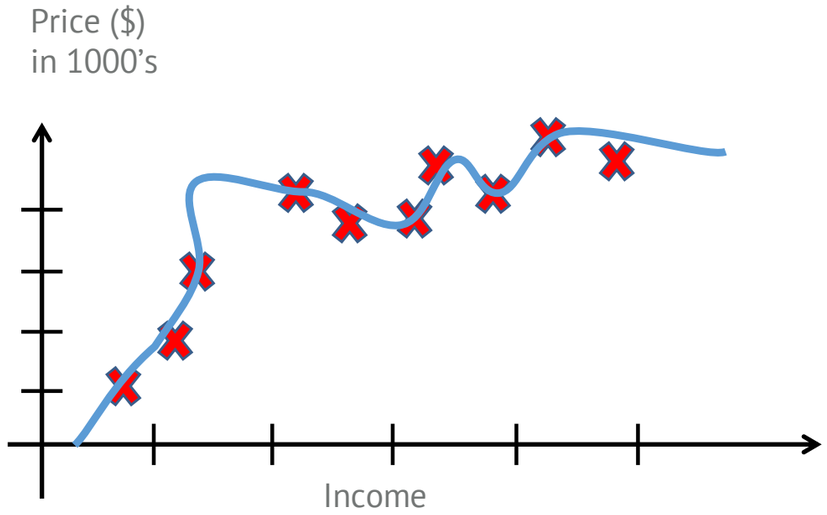
What is too complex a model? An example

Outputs

- $y =$ housing prices

Inputs

- linear: $x_1 =$ income; $x_2 =$ education; $x_3 =$ age; $x_4 =$ travel time; … x_{10}.
- Quadratic: x_1^2, x_2^2, x_3^2, x_4^2, … x_{10}^2
- Interaction terms: x_1x_2, x_1x_3, x_1x_4, …, x_2x_3, x_2x_4, …
- Higher order terms.
- …
- Adding them together, we can easily convert 10 features to **millions of input features**.



Consequences of underfitting and overfitting

Underfitting (too simple a model)

- Low performance in training set
- Low performance in testing set

Overfitting (too complex a model)

- **High** performance in training set
- **Low** performance in testing set

To address overfitting, we have to split the training & testing sets!

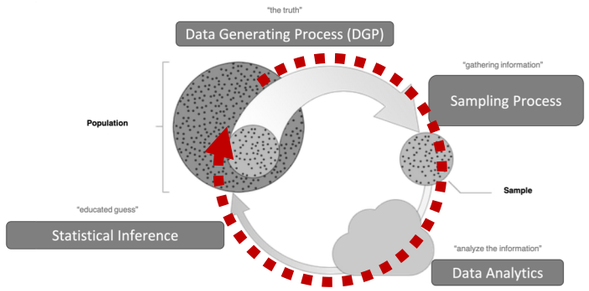
How to simplify a model?

1. Manually reduce the number of features

2. **Regularization**
   - Keep all the features, but automatically reduce magnitude or numbers of parameters θ.

Regularization in the general diagram

Linear Regression



1. Establish the goal (DGP)
   e.g. recovering $E[Y|X]$
2. Make modeling assumptions (e.g. i.i.d.)
   e.g. $E[Y|X] = \theta_0 + \theta_1 x_{1i} + \ ... \theta_k x_{ki}$
3. Estimate the model by minimizing an objective
   e.g. $\underset{\theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} (y_i \ - \theta_0 - \theta_1 x_{1i} - \ ... \ \theta_k x_{ki})^2$
4. Examine the performance
   e.g. R^2
5. Use the model (interpretation, prediction, etc.)
   e.g. β

3. Minimizing an objective + an extra term
   e.g. $\underset{\theta}{\operatorname{argmin}} \frac{1}{N} [\sum_{i=1}^{N} (y_i \ - \boldsymbol{\theta'} \boldsymbol{x}_i)^2 + \lambda \ \sum_{j=1}^{k} \theta_j^2]$

Intuition: prefer a model with smaller coefficients.

λ: regularization parameter

Regularization for both linear and logistic regressions

Linear regression + smaller coefficients

$$\underset{\theta}{\operatorname{argmin}} \frac{1}{N}[\sum_{i=1}^{N}(y_i - \boldsymbol{\theta}'\boldsymbol{x}_i)^2 + \lambda \sum_{j=1}^{k}\theta_j^2]$$

Ridge Regularization
(a.k.a. L2 or Tikhonov)

Linear regression + fewer coefficients

$$\underset{\theta}{\operatorname{argmin}} \frac{1}{N}[\sum_{i=1}^{N}(y_i - \boldsymbol{\theta}'\boldsymbol{x}_i)^2 + \lambda \sum_{j=1}^{k}|\theta_j|]$$

Lasso Regularization
(a.k.a. L1)

Logistic regression + smaller coefficients

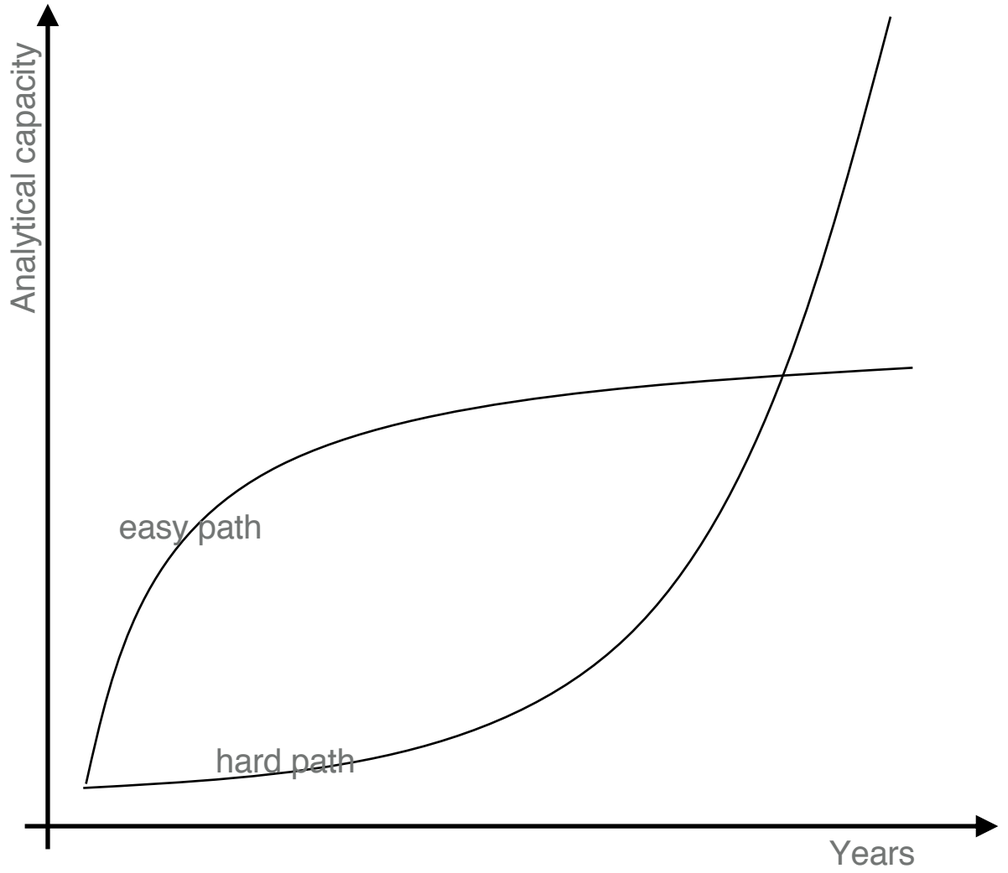
$$\underset{\theta}{\operatorname{argmin}} \frac{1}{N}[-\sum_{i=1}^{N} y_i \log h_\theta(x_i) + (1 - y_i) \log(1 - h_\theta(x_i)) + \lambda \sum_{j=1}^{k}\theta_j^2]$$

Logistic regression + fewer coefficients

$$\underset{\theta}{\operatorname{argmin}} \frac{1}{N}[-\sum_{i=1}^{N} y_i \log h_\theta(x_i) + (1 - y_i) \log(1 - h_\theta(x_i)) + \lambda \sum_{j=1}^{k}|\theta_j|]$$

Regularization vs. sklearn

Quote from the survey: “we don’t need to go very deep into some of the (math) concepts, since we will use the packages anyways.”



Why did I introduce “regularization”, while we will use packages anyway?

- It was invented in 1963 (Tikhonov), and probably will be used for at least other five decades.
- Python packages (e.g. sklearn) was invented in only 2007, and could easily be replaced in next ten years.

The easy path is hard. The hard path is easy.

- However, it is still fun to click a button to see the results, and they are most helpful in our daily work.

General Diagram

Linear Regression

1. Establish the goal (DGP)
   e.g. recovering $E[Y|X]$
2. Make modeling assumptions (e.g. i.i.d.)
   e.g. $E[Y|X] = \theta_0 + \theta_1 x_{1i} + \ ... \theta_k x_{ki}$
3. Estimate the model by minimizing an objective
   e.g. $\underset{\theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} (y_i \ - \boldsymbol{\theta' x}_i)^2$
4. Examine the performance
   e.g. R^2
5. Use the model (interpretation, prediction, etc.)
   e.g. β

Regression in ML

1. Establish the goal (DGP)
   e.g. learning $E[Y|X]$ or $\mathrm{P}[Y|X]$
2. Make modeling assumptions (e.g. i.i.d.)
   e.g. $E[Y|X] = \theta_0 + \theta_1 x_{1i} + \ ... \theta_k x_{ki}$ (but k can be extremely large.)
3. Estimate the model by minimizing an objective + regularization
   e.g. $\underset{\theta}{\operatorname{argmin}} \frac{1}{N} [\sum_{i=1}^{N} (y_i \ - \boldsymbol{\theta' x}_i)^2 + \lambda \ \sum_{j=1}^{k} \theta_j^2]$
4. Examine the performance
   e.g. R^2 in training and testing sets.
5. Use the model (interpretation, prediction, etc.)
   e.g. $\hat{E}[Y|X]$ or $\hat{P}[Y|X]$ for prediction.