Outline

Defining Your Questions

- Is it a decision to be made?
- Is there a pattern to detect?
- Do you have data?
- What kinds of questions do you have about the data?

  Yes/No questions - Did X happen? Are A and B correlated?

  Timing - When did X happen?

  Anomaly detection - Is X strange/abnormal/unexpected?

  Classification - What kind of Y is X?

  Prediction - Weve seen lots of (X,Y) now we want to know (X',?)
- Do you have labels?
  - Can you give the right answer for some portion of the data?
  - Collecting labels: Automatic? Manual? Crowd-sourced? (eg. Amazon Mechanical Turk) Y
  - Yes $\rightarrow$ Supervised Learning - Lots of options
  - No $\rightarrow$ Unsupervised Learning - Some options (getting better all the time)

Answers and Constraints

What kind of answer do you need? (increasing difficulty)

- Find patterns which are present in the data and view them
- Most likely explanation for a pattern
- Probability of (fact about X,A,B...) being true
- A policy for actions to take in the future to maximize benefit
- Guarantees that X will (or will not) happen (very hard)

How big is your data?

- Is it static?
- MB, GB, TB?
- Is it streaming?
- KB/sec, MB/sec
- How many data points/rows/events will there be?

How to Design your AI/ML Question

Define your task:

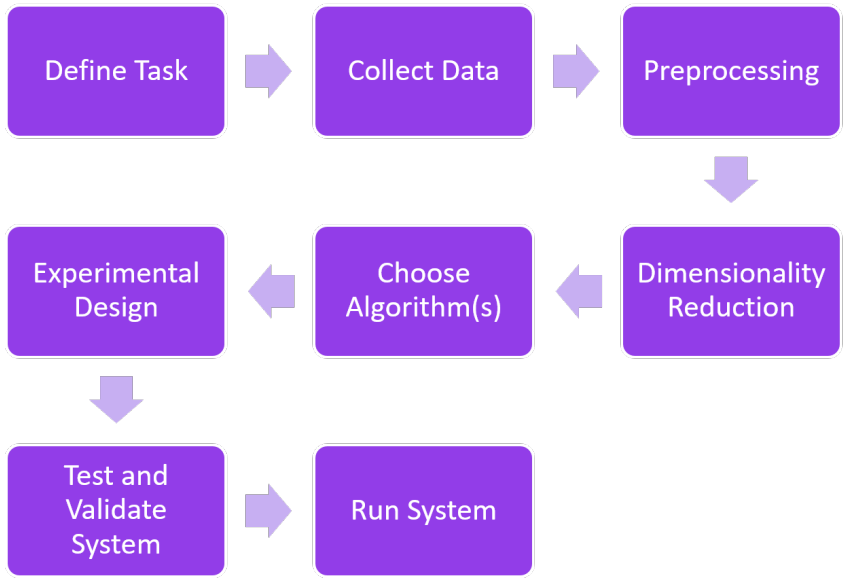
- Prediction, Clustering, Classification, Anomaly Detection?
- Define objectives, error metrics, performance standards

Collect Data:

- Set up data stream (storage, input flow, parallelization, Hadoop)

Preprocessing:

- Noise/Outlier Filtering
- Completing missing data (histograms, interpolation)
- Normalization (scaling data)



How to Design your AI/ML Question

Dimensionality Reduction / Feature Selection:

- Choose features to use/extract from data
- PCA/LDA/LLE/GDA

Choose Algorithm:

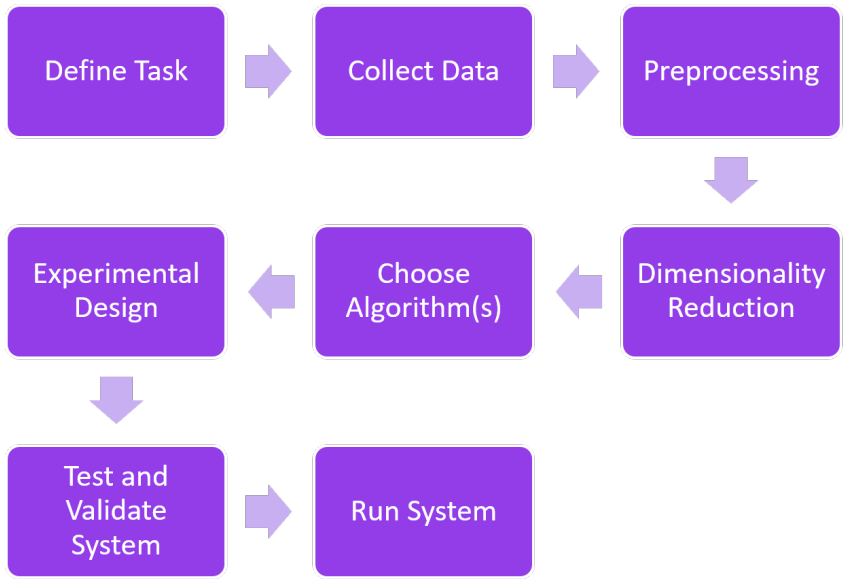
- Consider goals, questions
- Tractability

Experimental Design:

- train/validate/test data sets
- cross-validation

Run it! :

- Deployment



Language Choices

Any language can be used for implementing/using AI/ML algorithms, but some make it much easier

- C++: you can do it, may need to implement many things yourself
- Java: many of libraries for ML (Weka is a good open source one, Deeplearning4j)
- Scala: leaner, functional language that compile to JVM bytecode, good for prototyping, can reuse libraries for Java (Deeplearning4j)
- R: focussed on statisical methods, more and more machine learning libraries implemented for this
- Matlab: good for all the calculations, if you have the right libraries it's great (not cheap or very portable beyond school)
- Python: most commonly used right now for deep learning, we're gonna need another slide ...

Python

numpy - numerical libraries, implementation of matrix and linear algerbra datastructures, graphing tools

pandas - table datastructure, statistical analysis tools (implements many useful features from R)

scipy - includes all of the above and more, full installation of scientific libraries, basically turns Python into R+Matlab

scikit-learn - many standard machine learning algorithms implemented as easy-to-use Python APIs

jupyter notebooks - these are powerful web-based interfaces to python for data analysis and machine learning.

Deep Learning Frameworks

Caffe - older, easy to set up mockups, harder to install?

Theano - made out of University of Montreal, great theoretical setup, very flexible, python only

Tensorflow - made by Google, scales to many GPUs, servers, lots of optimization, requires planning of the whole system beforehand, most languages

PyTorch - easier to mock things up, try different designs, not as optimized for large scale performance as tensorflow

MXNet - made by Microsoft, supports most languages and OS's

Deeplearning4j - Java focussed framework

Keras - open interface to create models in multiple frameworks (tensorflow, theano, MXNet)

Cloud Services

There are several powerful, free services you can access via a student account which you can request directly.

AWS: Amazon Web Service - very large, has accessible APIs to connect to, many options for hardware to run on (but the best ones will cost extra)

Azure: Microsoft - lots of visual tools for composing AI/ML components.

Google Cloud ML Engine: - uses all the latest tools and tensorflow models

None of these provide GPU servers for free, that will cost extra. (It will still work, just be slower for deep learning.)

Summary

Useful Books

A book for of three eras of Machine Learning:

[Goodfellow, 2016]
Goodfellow, Bengio and Courville. *"Deep Learning"*, MIT Press, 2016.

- http://www.deeplearningbook.org/
- Website has free copy of book as pdf's.

[Murphy, 2012]
Kevin Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012.

[Duda, Pattern Classification, 2001]
R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification (2nd ed.)*, John Wiley and Sons, 2001.

Useful Papers and Blogs

[lecun2015]
Y. LeCun, Y. Bengio, G. Hinton, L. Y., B. Y., and H. G., "Deep learning", *Nature*, vol. 521, no. 7553, pp. 436444, 2015. Great references at back with comments on seminal papers.

[bengio2009]
Y. Bengio, "Learning Deep Architectures for AI", *Foundations and Trends in Machine Learning*, vol. 2, no. 1. 2009. An earlier general referenceon the fundamentals of Deep Learning.

[krizhevsky2012]
A. Krizhevsky, G. E. Hinton, and I. Sutskever,"ImageNet Classification with Deep Convolutional Neural Networks", *Adv. Neural Inf. Process. Syst.* pp. 19, 2012. The beginning of the current craze.

[Karpathy, 2015]
Andrej Karpathy's Blog - `http://karpathy.github.io` Easy to follow explanations with code