Data Science

Lecture 3: Modeling / Big Data

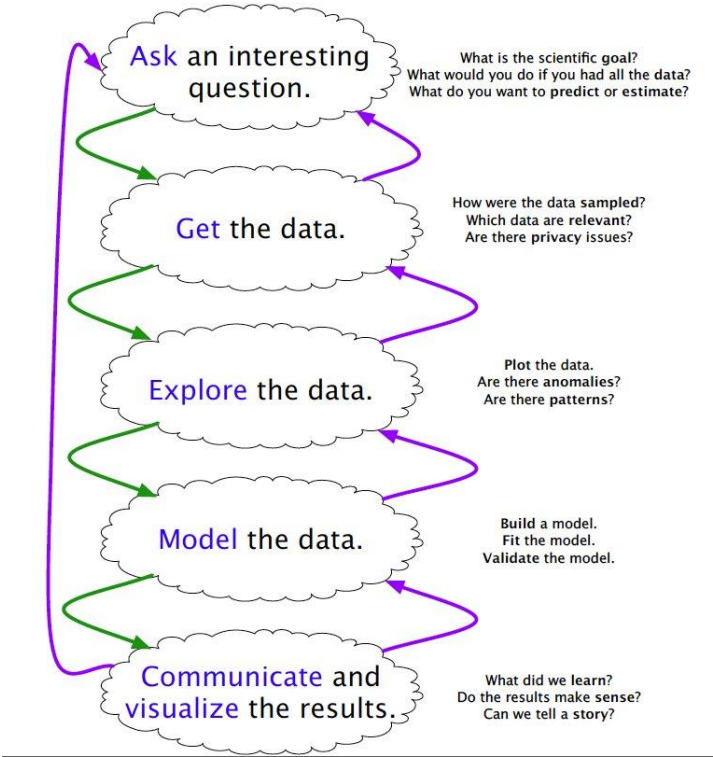
Acknowledgement: Lecture materials are prepared using The Data Science Design Manual by Steven S. Skiena, 2017.

Model the Data

The Data Science Analysis Pipeline

Modeling is the process of encapsulating information into a tool which can make forecasts/predictions.

The key steps are building, fitting, and validating the model.



Philosophies of Modeling

We need to think in some fundamental ways about modeling to build them in sensible ways.

- Occam’s Razor
- Bias-Variance tradeoffs

Occam’s Razor

This philosophical principle states that “the simplest explanation is best”.

When presented with competing hypothetical answers to a problem, one should select the answer that makes the fewest assumptions.

With respect to modeling, this often means minimizing the parameter count in a model.

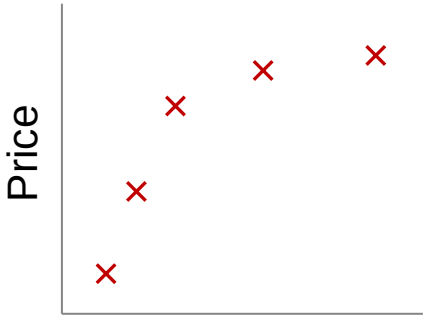
Bias-Variance Tradeoffs

"All models are wrong, but some models are useful."

– George Box (1919-2013)

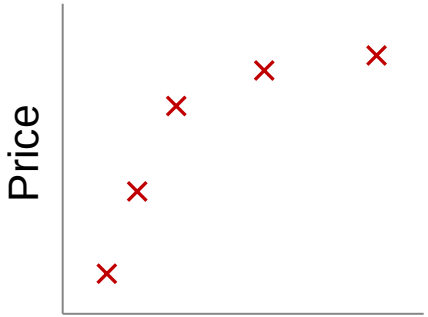
- *Bias* is error from erroneous assumptions in the model, like making it linear. (underfitting)
- *Variance* is error from sensitivity to small fluctuations in the training set. (overfitting)

Bias/variance



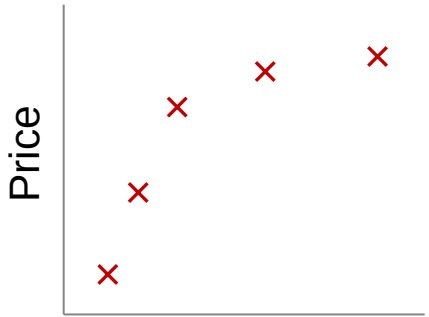
$\theta_0 + \theta_1 x$

High bias
(underfit)



$\theta_0 + \theta_1 x + \theta_2 x^2$

"Just right"



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

High variance
(overfit)

Taxonomy of Models

Models have different properties inherent in how they are constructed:

- Linear vs. nonlinear
- Black box vs. descriptive
- First-principle vs. data-driven
- Stochastic vs. deterministic
- Flat vs. hierarchical

Linear vs. Non-Linear Models

Linear models are governed by equations that weigh each feature variable by a coefficient reflecting its importance, and sum up these values to produce a score.

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Non-linear models involve higher-order polynomials, logarithms, and exponentials.

Blackbox vs. Descriptive Models

Ideally models are descriptive, meaning they explain why they are making their decisions.

Linear regression models are descriptive, because one can see which variables are weighed heaviest.

Neural network models are generally opaque.

Lesson: “Distinguishing cars from trucks.”

Deep Learning Models are Blackbox

Deep learning models for computer vision are highly-effective, but opaque as to how they make decisions.
They can be badly fooled by images which would never confuse human observers.

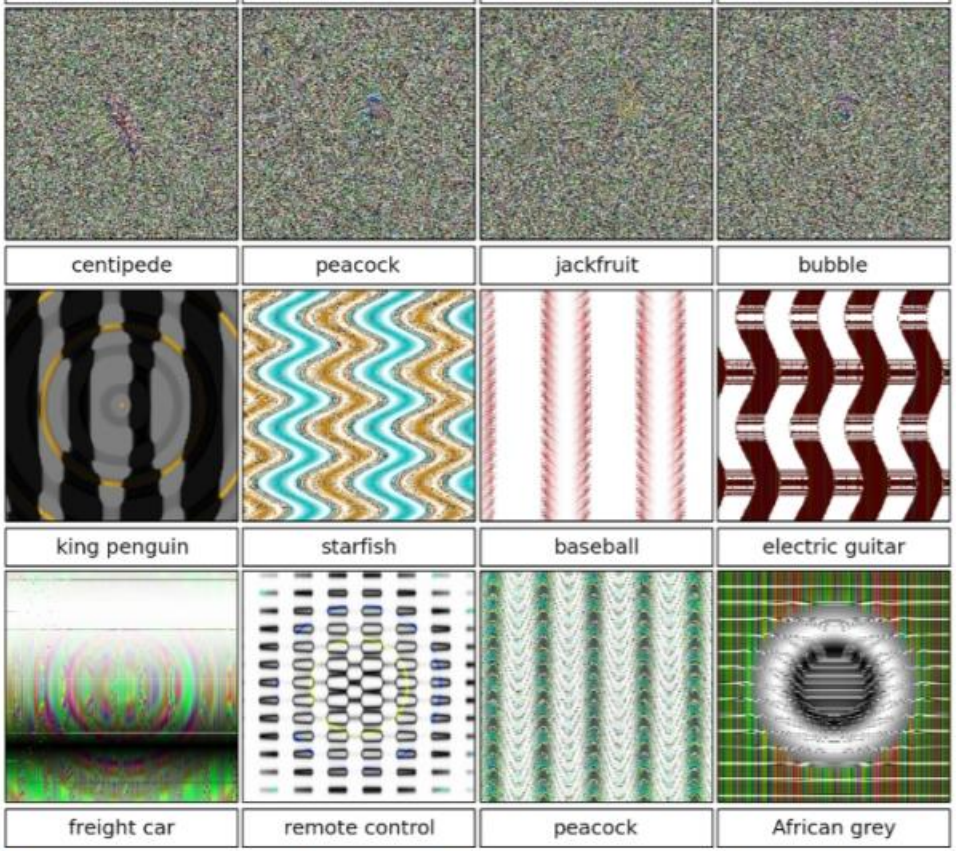


Figure 1. Evolved images that are unrecognizable to humans, but that state-of-the-art DNNs trained on ImageNet believe with $\geq$ 99.6% certainty to be a familiar object. This result highlights differences between how DNNs and humans recognize objects. Images are either directly (*top*) or indirectly (*bottom*) encoded.

General vs. Ad Hoc Models

Machine learning models for classification and regression are general, meaning they employ no problem-specific ideas, only specific data.

Ad hoc models are built using domain-specific knowledge to guide their structure and design.

Evaluating Models: Baselines

The first step to assess whether your model is any good is to build baselines: the simplest *reasonable* models to compare against.

Only after you decisively beat your baselines can your models be deemed effective.

Representative Baseline Models

- Uniform or random selection among labels.
- The most common label in the training data.
- The best performing single-variable model.
- Same label as the previous point in time.
- Mean or median.
- Linear regression.
- Existing models.

Baseline models must be fair: they should be simple but not stupid.

Evaluating Classifiers

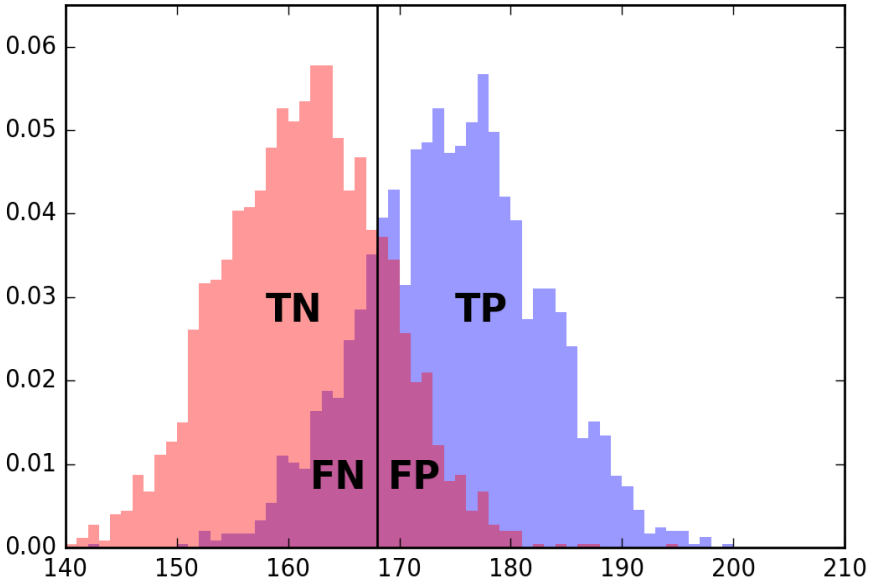
There are four possible outcomes for a binary classifier:

- True positives (TP) where + is labeled +
- True negative (TN) where - is labeled -
- False positives (FP) where - is labeled +
- False negatives (FN) where + is labeled -

		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

Threshold Classifiers

Identifying the best threshold requires deciding on an appropriate evaluation metric.



Accuracy

The accuracy is the ratio of correct predictions over total predictions:

$$accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

The monkey would randomly guess P/N, with accuracy 50%.

Picking the biggest class yields >=50%.

Precision

When |*P*|<<|*N*|, accuracy is a silly measure.

e.g. only 5% of tests say cancer.

$$precision = \frac{TP}{(TP + FP)}$$

		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

Recall

In the cancer case, we would tolerate some false positive (scares) to identify real cases:

$$recall = \frac{TP}{(TP + FN)}$$

		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

F-Score

To get a meaningful single score balancing precision and recall use F-score:

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

The harmonic mean is always <= arithmetic mean, making it tough to get a high F-score.

Sensitivity vs. Specificity

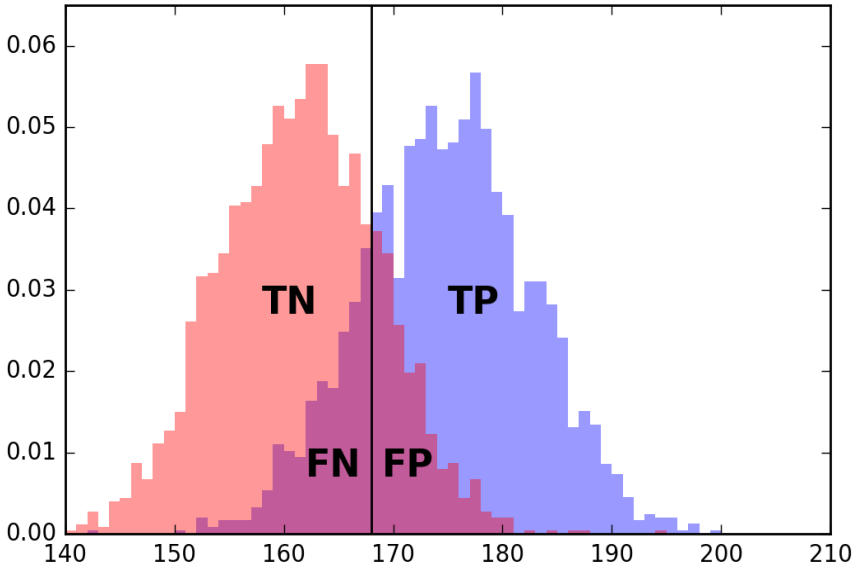
		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

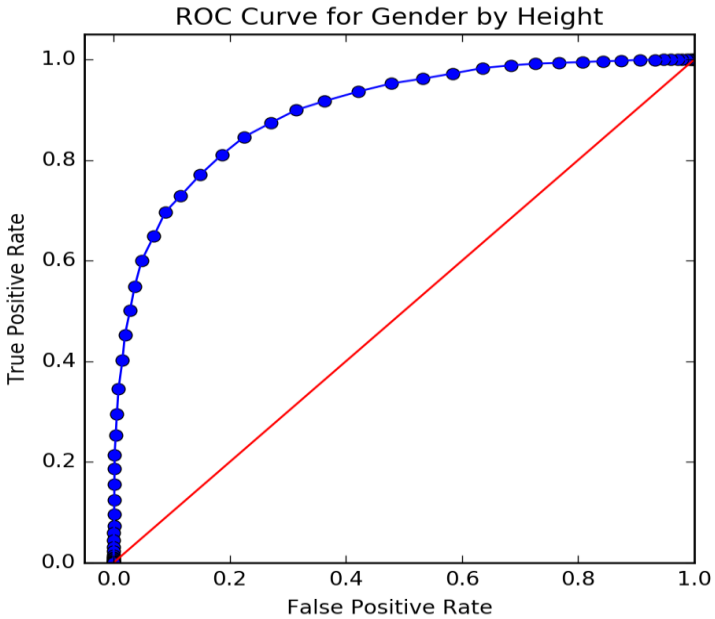
		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

Receiver-Operator Characteristic (ROC) Curves

Varying the threshold changes recall/precision, TPR/FPR.

Area under ROC curve (AUC) is a measure of accuracy.





Summary Statistics: Numerical Error

For numerical values, error is a function of the delta between forecast *f* and observation *o*:

- Absolute error: *(f - o)*
- Relative error: *(f - o) / o* (typically better)

These can be aggregated over many tests:

- Mean or median squared error

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2.$$

- Root mean squared error

$$\mathrm{RMSD}(\hat{\theta}) = \sqrt{\mathrm{MSE}(\hat{\theta})} = \sqrt{\mathrm{E}((\hat{\theta} - \theta)^2)}.$$

Evaluation Data

The best way to assess models involve out-of-sample predictions, results on data you never saw (or even better did not exist) when you built the model.

Partitioning the input into training (60%), validation (20%) and testing (20%) data works only if you never open testing data until the end.

Dealing with Big Data

Big Data as Bad Data

Massive data sets are the result of opportunity instead of design, with problems of:

- Unrepresentative participation (bias)
  - Instagram (too young), The New York Times (too liberal), Fox News (too conservative), or The Wall Street Journal (too wealthy).
- Spam and machine-generated content
  - e.g., bots, fake product reviews.

Big Data as Bad Data

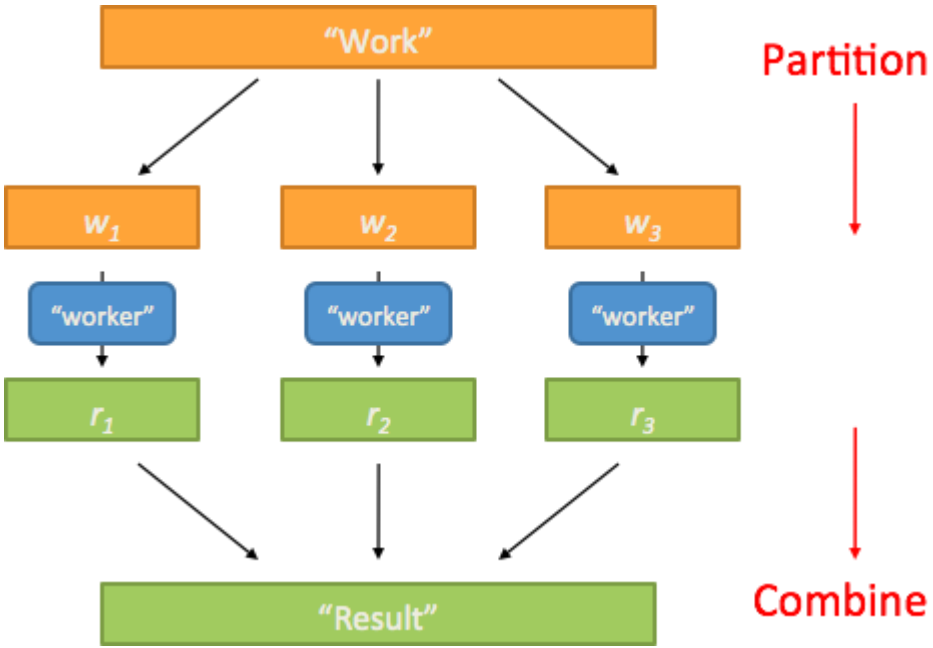
- Power-laws mean too much redundancy
  - 1000s of Empire State Building photo but none for many buildings
- Susceptibility to temporal bias (e.g Google Flu Trends)
  - auto-complete mechanism changed the distribution of search queries

MapReduce / Hadoop

Google’s MapReduce paradigm for distributed computing has spread widely through the open-source implementation like Hadoop and Spark, offering:

- Simple parallel programming model
- Straightforward scaling to hundreds/thousands of machines.
- Fault tolerance through redundancy

Divide and Conquer



Ideas Behind MapReduce

- Scale “out”, not “up”: recognize limits of large shared-memory machines
- Move processing to the data: clusters have limited bandwidth
- Process data sequentially, avoid random access: seeks are expensive, disk throughput is reasonable

Components of Hadoop

- Core Hadoop has two main systems:
  - **Hadoop/MapReduce**: distributed big data processing infrastructure (abstract/paradigm, fault-tolerant, schedule, execution)
  - **HDFS (Hadoop Distributed File System)**: fault-tolerant, high-bandwidth, high availability distributed storage

Map and Reduce

•Programmers specify two functions:

map (k, v) → [(k’, v’)]

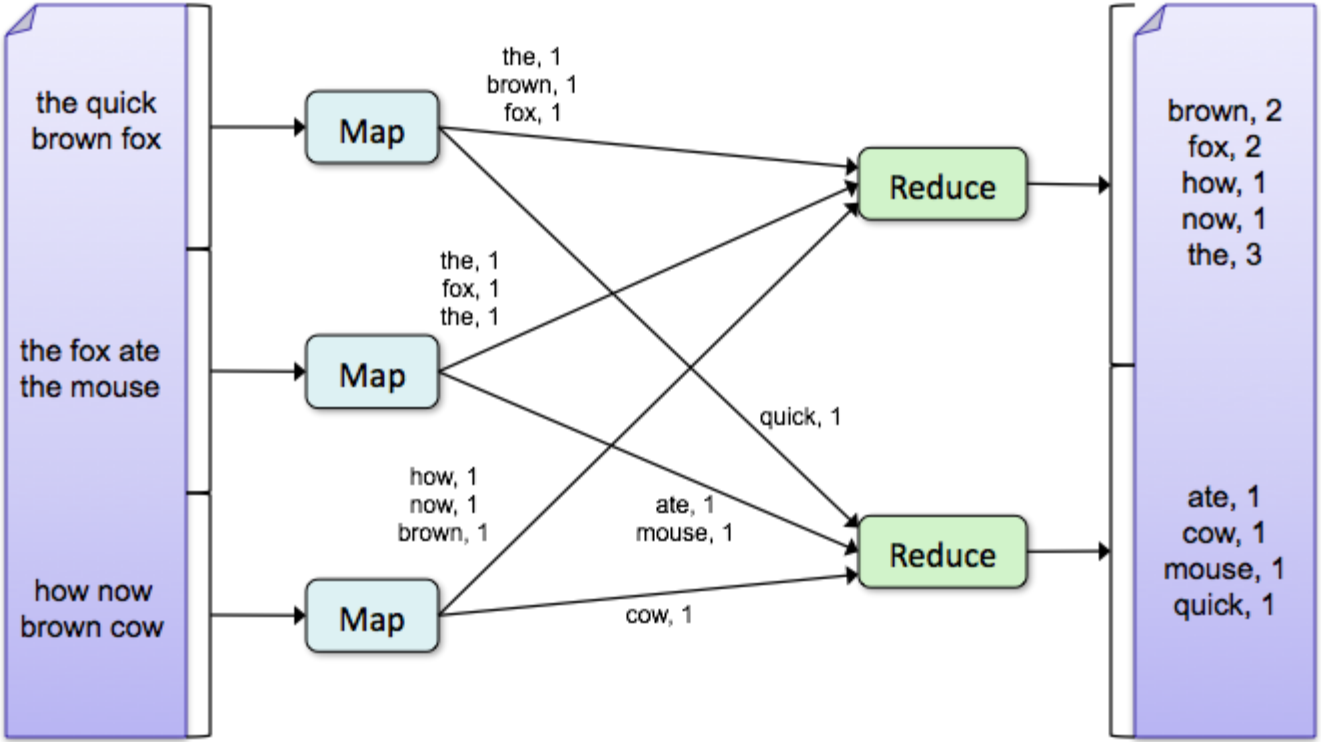
reduce (k’, [v’]) → [(k’, v’)]

–All values with the same key are sent to the same reducer

MapReduce Word Count

```
Map(String docid, String text):
    for each word w in text:
        Emit(w, 1);
Reduce(String term, Iterator<Int> values):
    int sum = 0;
    for each v in values:
        sum += v;
    Emit(term, sum);
```

Word Count Execution



Hadoop Distributed File System (HDFS)

Store data on the local disks of nodes in the cluster, because not enough RAM to hold all the data in memory

Disk access is slow, but disk throughput is reasonable, so linear scans through files are fine.

Replicate everything 3 times for reliability on commodity hardware.

Cloud Computing Services

Platforms like Amazon make it easy to rent large numbers of machines for short-term jobs.

There are charges on bandwidth, processors, memory, long-term storage: making it non-trivial to price exactly.

Feel Free to Experiment

Micro instances are only 1GB, single processor virtual machines.

Reasonable machines rent for 10 to 30 cents/hr.

Free Tier*

As part of AWS's Free Usage Tier, new AWS customers can get started with Amazon EC2 for free. Upon sign-up, new AWS customers receive the following EC2 services each month for one year:

- 750 hours of EC2 running Linux, RHEL, or SLES t2.micro instance usage
- 750 hours of EC2 running Microsoft Windows Server t2.micro instance usage
- 750 hours of Elastic Load Balancing plus 15 GB data processing
- 30 GB of Amazon Elastic Block Storage in any combination of General Purpose (SSD) or Magnetic, plus 2 million I/Os (with Magnetic) and 1 GB of snapshot storage
- 15 GB of bandwidth out aggregated across all AWS services
- 1 GB of Regional Data Transfer