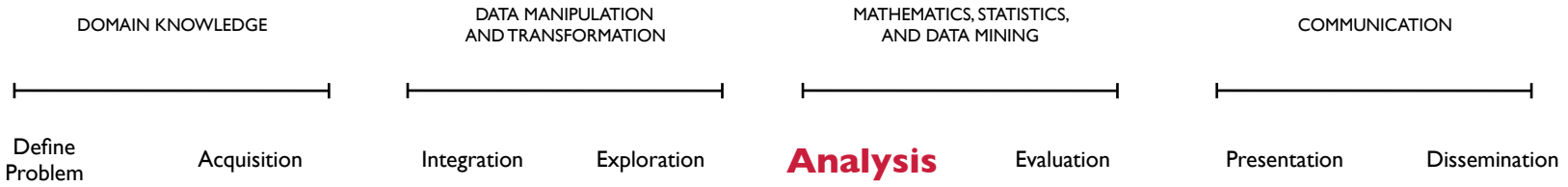
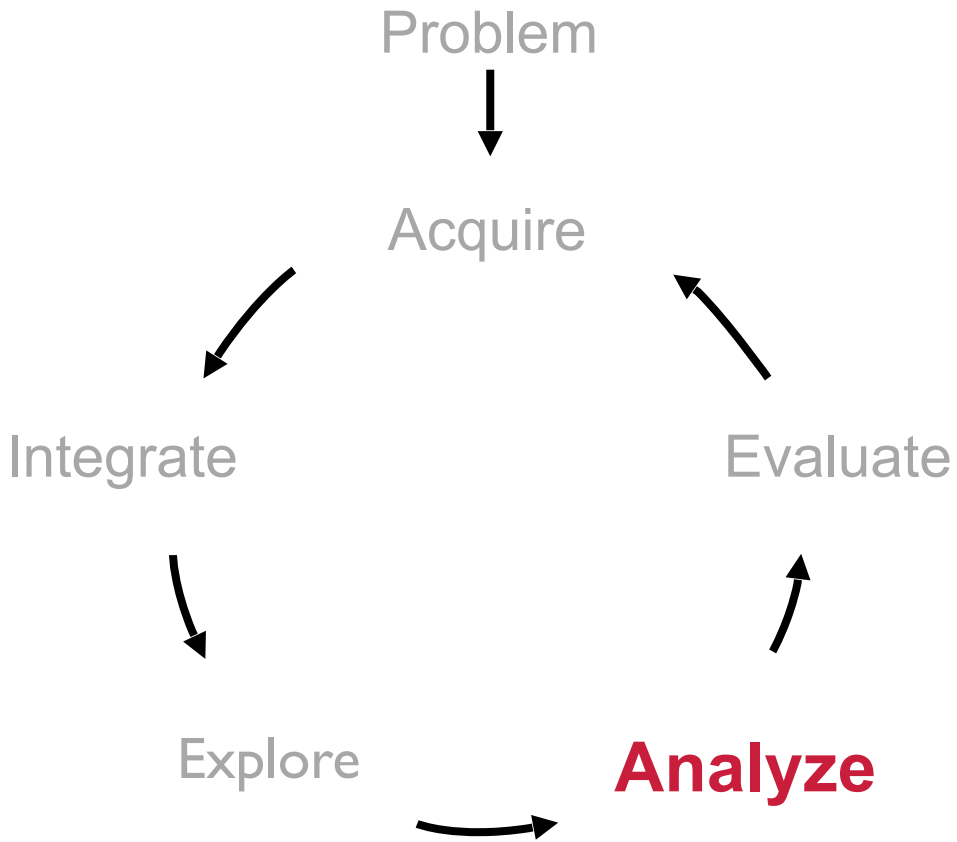
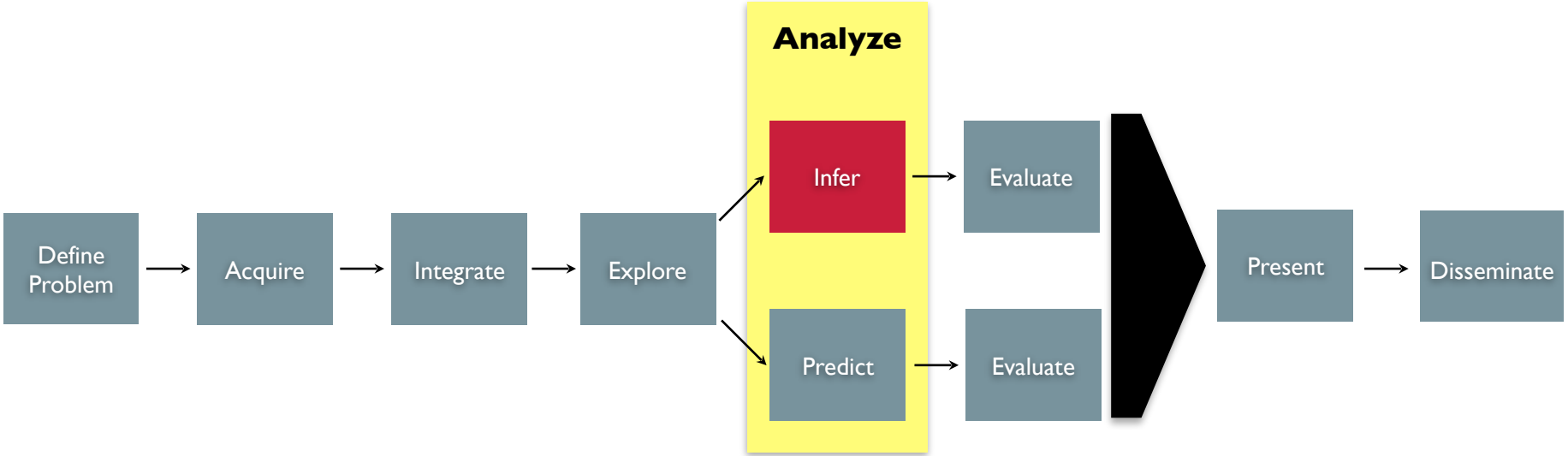
Process



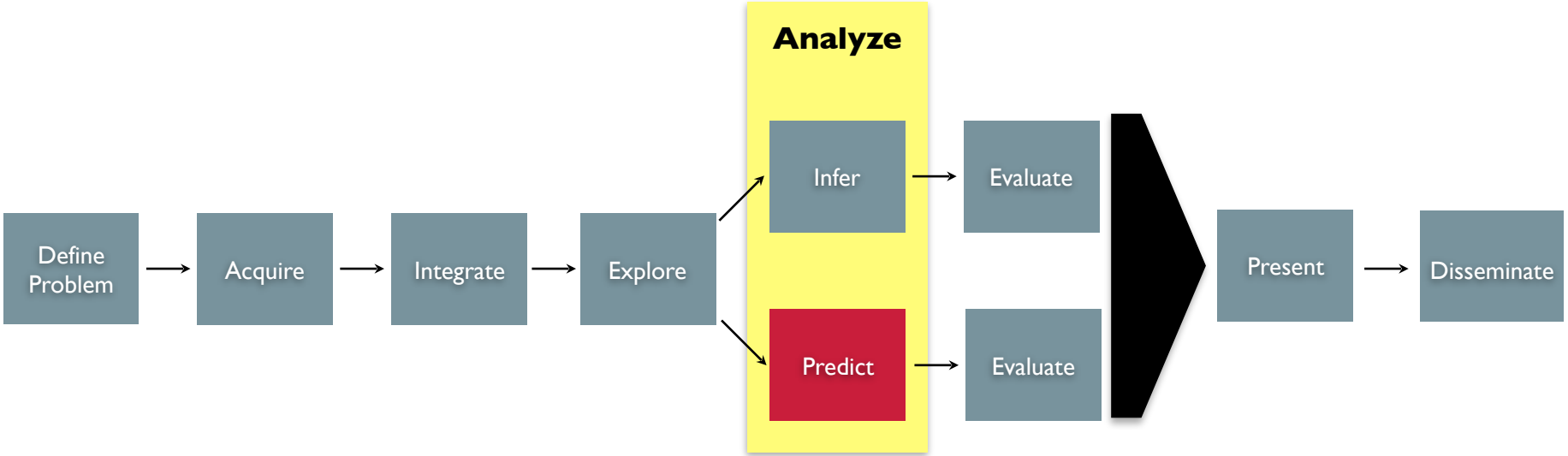
Process + Iteration



Process + Statistics



Process + Machine Learning



What is Machine Learning?

Field of study that gives computers the ability to learn without being explicitly programmed.

- Arthur Samuel circa 1959

What is Machine Learning?

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*
>
> - Tom M. Mitchell

What is Machine Learning?

- Automated knowledge acquisition through input
- Iterative improvement as more data is seen
- Adaptive Algorithms

What ISN'T Machine Learning?

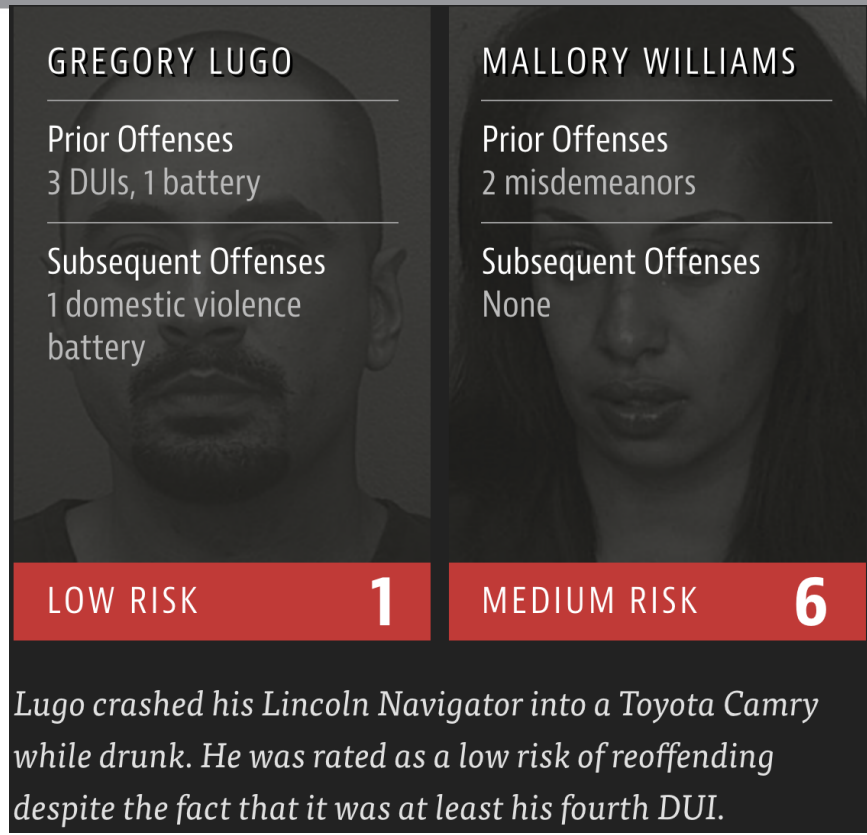
- Hard coded logic by programmer: `if` and `else`...
- Predefined results: completely deterministic
- Burden is placed on programmer at design time
- Must anticipate all inputs to program, and react

Recent Successes

- IBM Watson recommended the same treatment as oncologists in 990 out of 1000 cases
- A natural language system predicted the judicial decisions of the European Court of Human Rights with 79% Accuracy

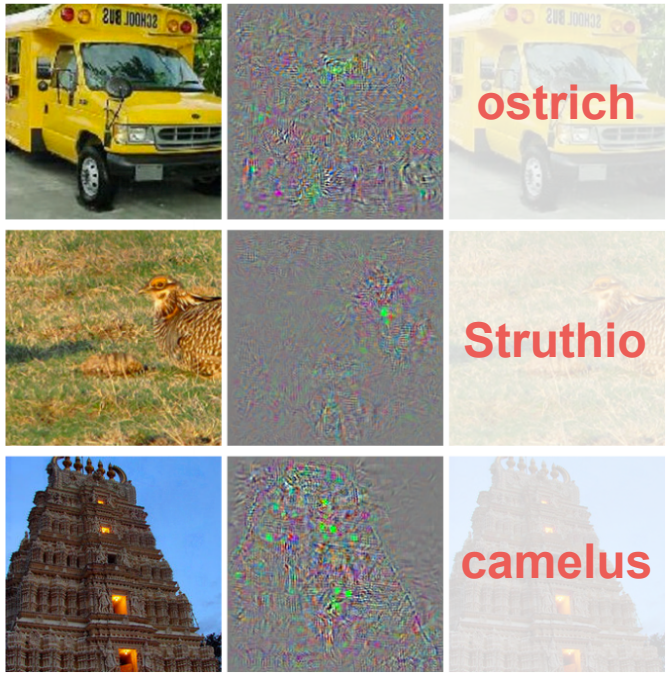
Recent Failures

Bias in the Machine



Source: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

Recent Failures: Adversarial Examples



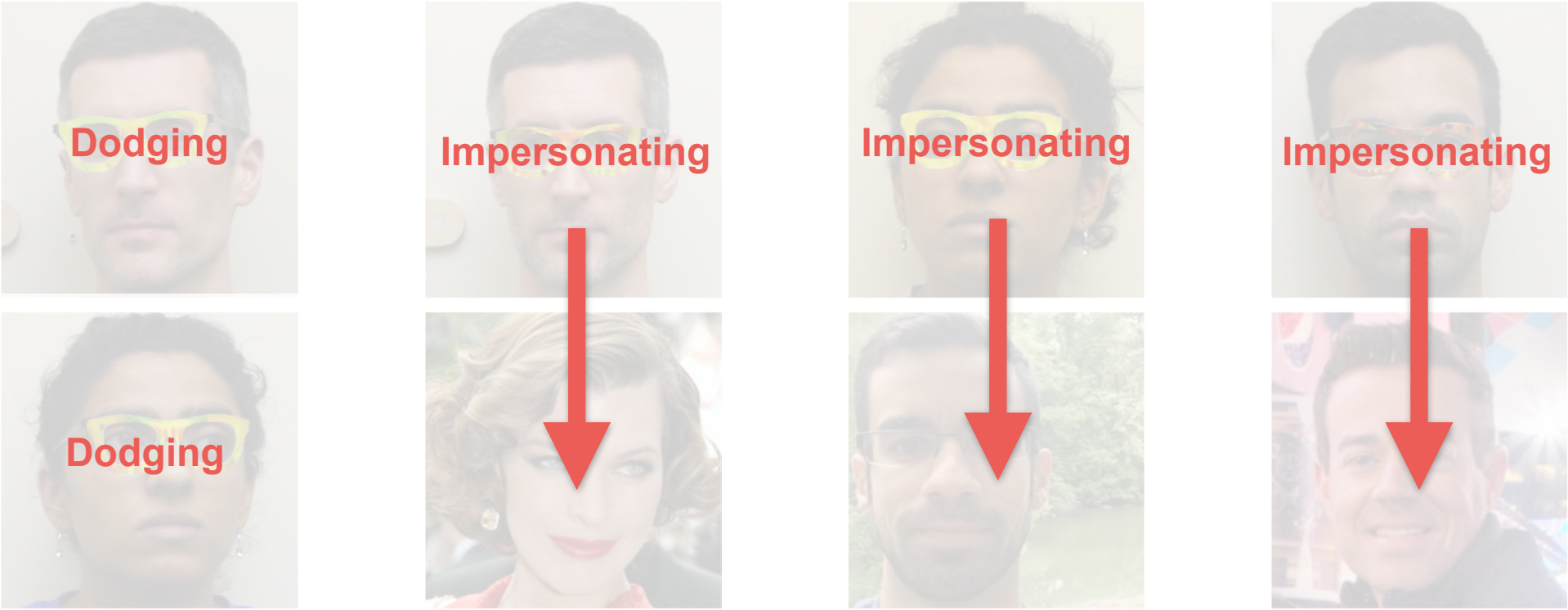
(a)

(b)

Source: https://arxiv.org/abs/1312.6199

Recent Failures: Adversarial Examples

Source: https://www.cs.cmu.edu/~sbhagava/papers/face-rec-ccs16.pdf

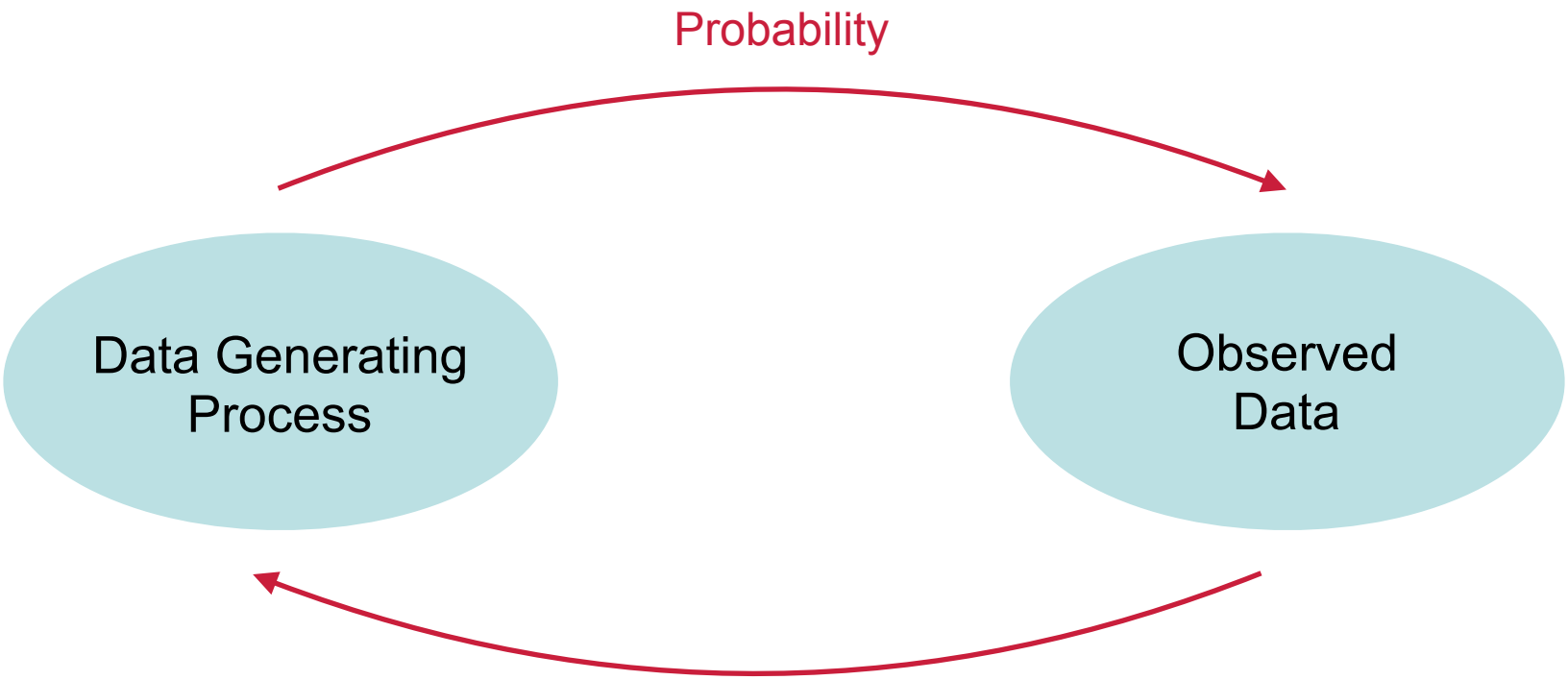


Statistics versus Machine Learning

Machine learning	Statistics
network, graphs	model
weights	parameters
learning	fitting
generalization	test set performance
supervised learning	regression/classification
unsupervised learning	density estimation, clustering
large grant = $1,000,000	large grant= $50,000
nice place to have a meeting: Snowbird, Utah, French Alps	nice place to have a meeting: Las Vegas in August

http://datavu.blogspot.com/2014/08/statistical-modeling-vs-machine-learning.html

Statistics versus Machine Learning



Inference versus Prediction

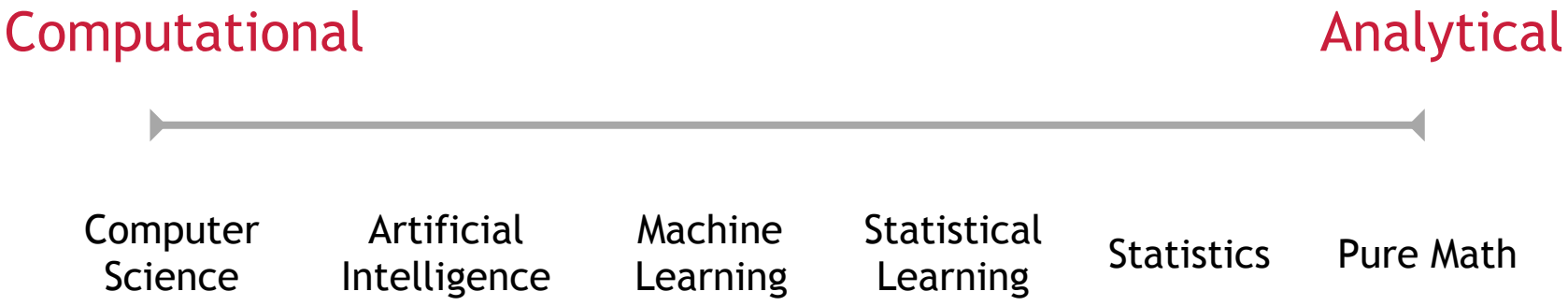
Statistics

- Models underlying process
- Assumptions about data
- Interpret Coefficients

Machine Learning

- Pure prediction/accuracy
- No concept of CI
- Often Black Box

The Spectrum of the Learning Arts



Types of Learning

Supervised Learning

- Training data includes desired output

Unsupervised Learning

- Training data does not include desired output

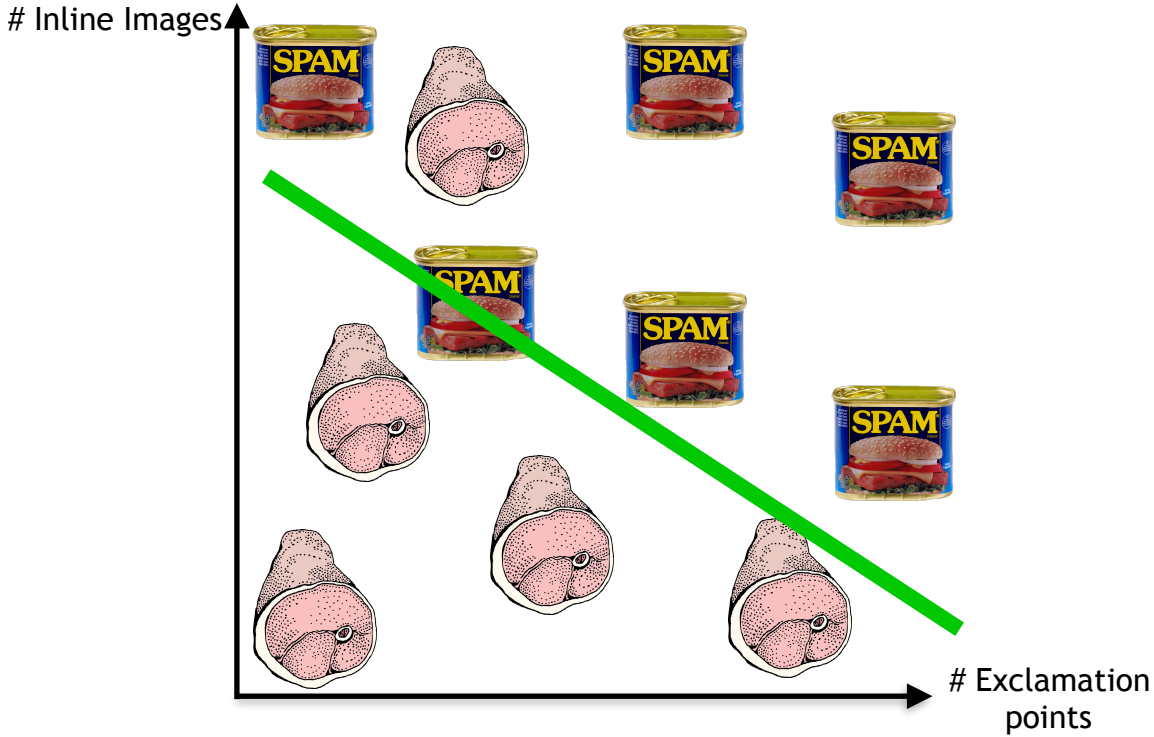
Semi-supervised Learning

- Training data includes some desired outputs

Reinforcement Learning

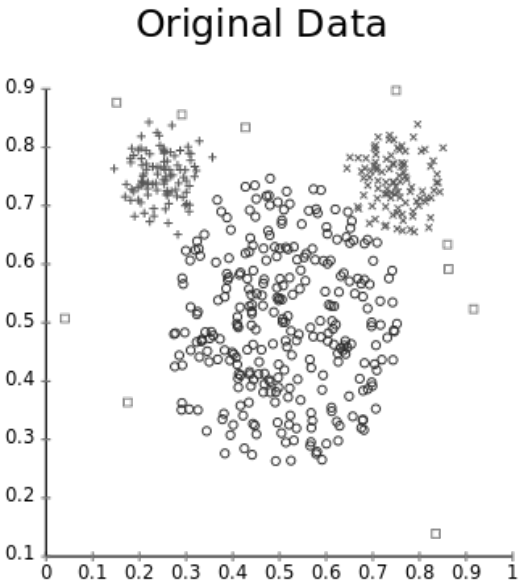
- Rewards from sequence of actions

Supervised Learning



Example: Spam email classifier

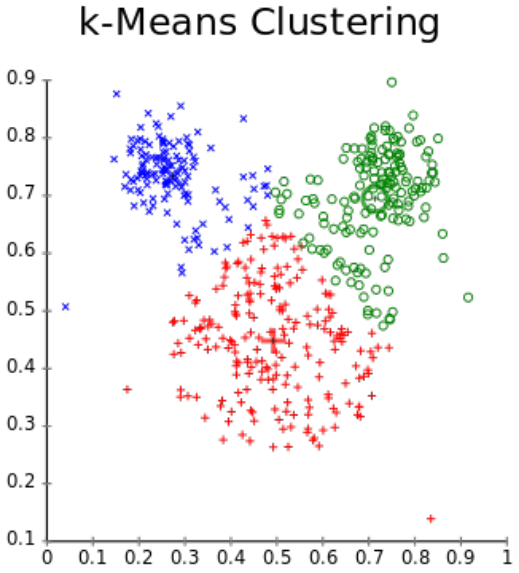
Unsupervised Learning



How many subgroups?

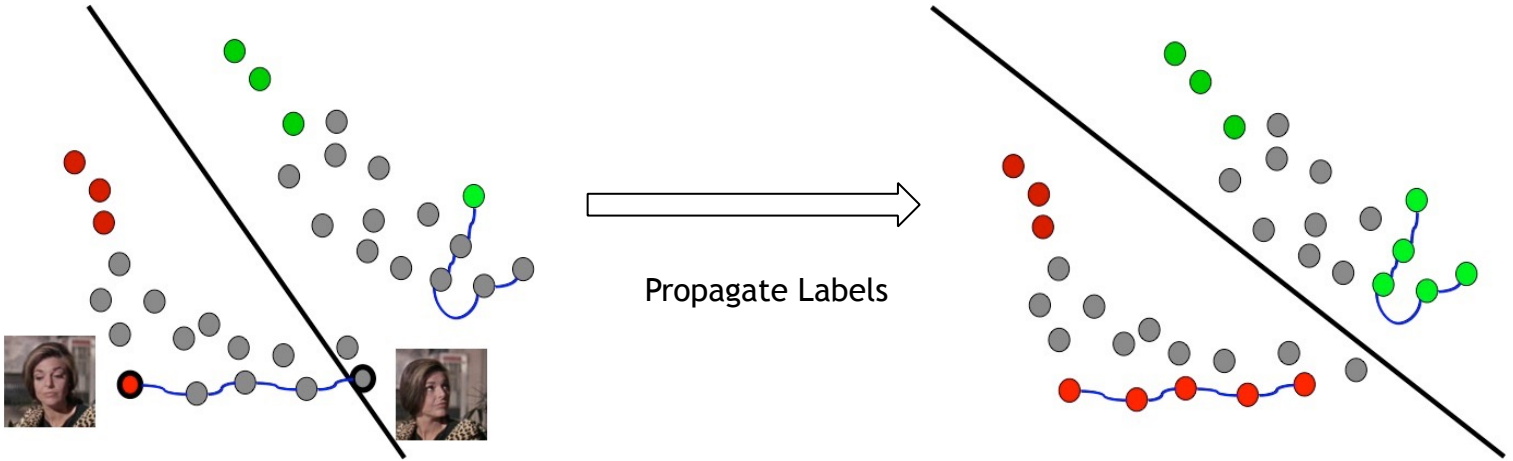
What's considered similar?

How can we even do this?



Example: User Cohorts

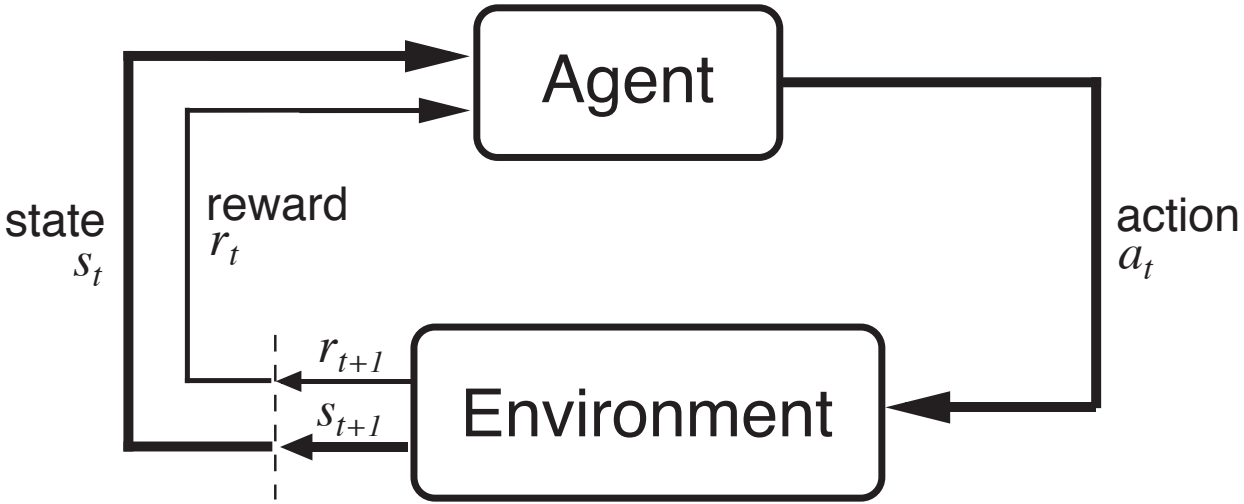
Semi-supervised Learning



Example: Image/Video Classification

Source: http://www.di.ens.fr/willow/research/attribute_learning/

Reinforcement Learning



Example: Autonomous Video game Agent

Source: Sutton and Barto (1998)

A Taxonomy of Models

Labeled Data

Supervised

Unsupervised

Unlabeled Data

The Unreasonable Effectiveness of Data

[Banko and Brill, 2001]

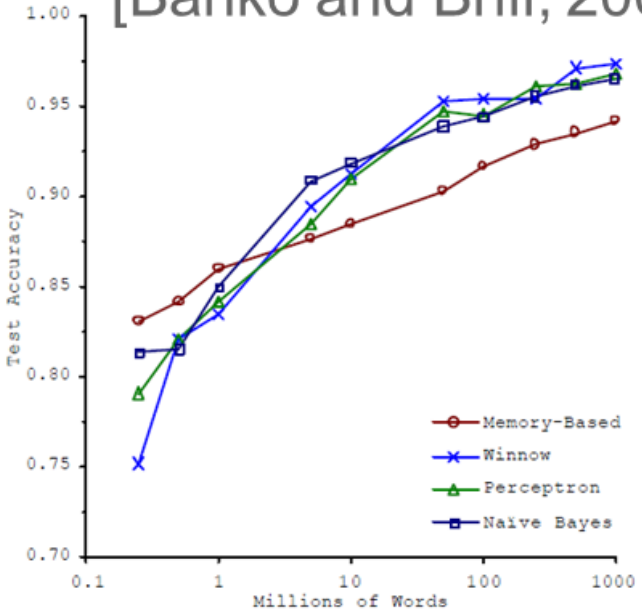
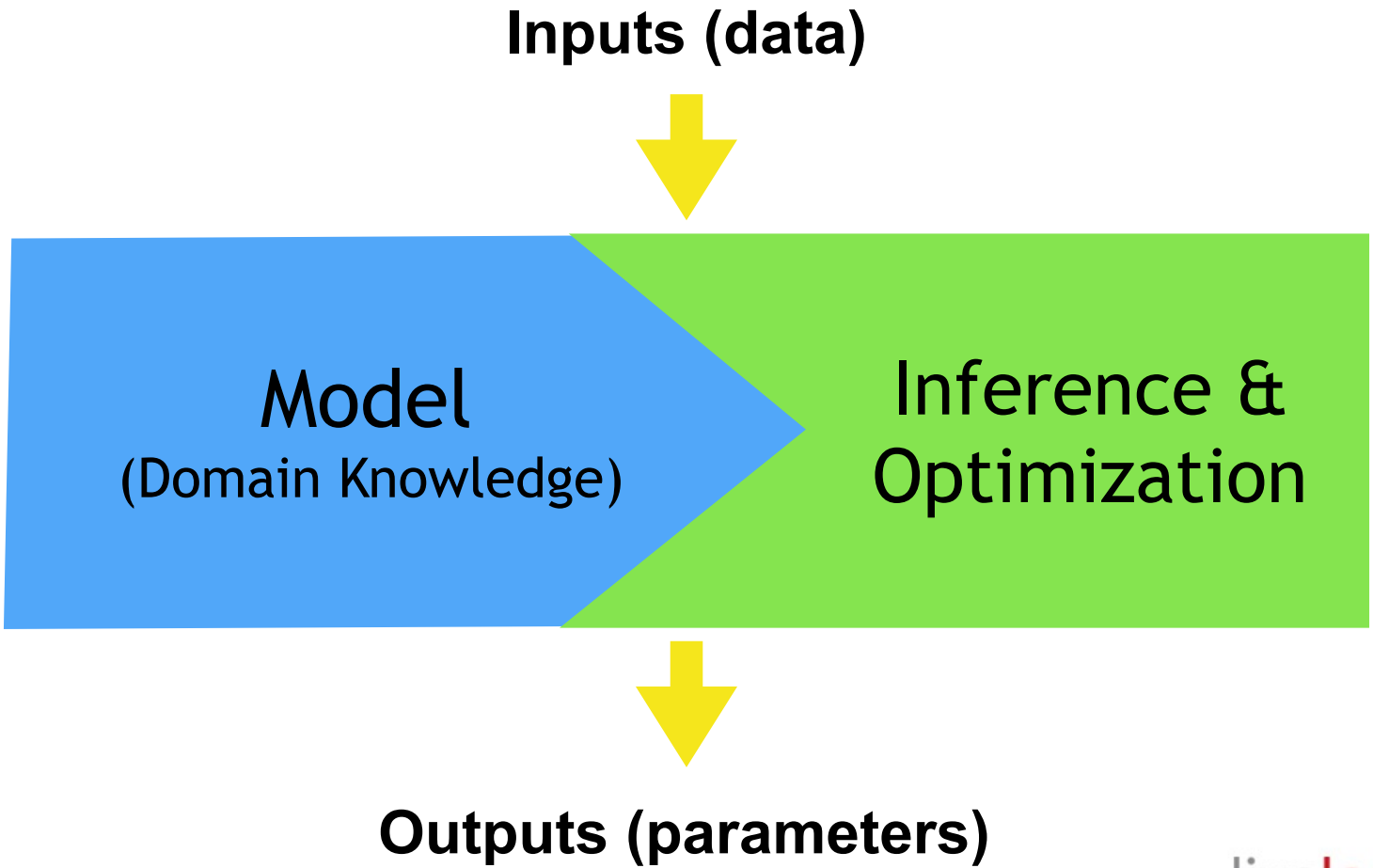
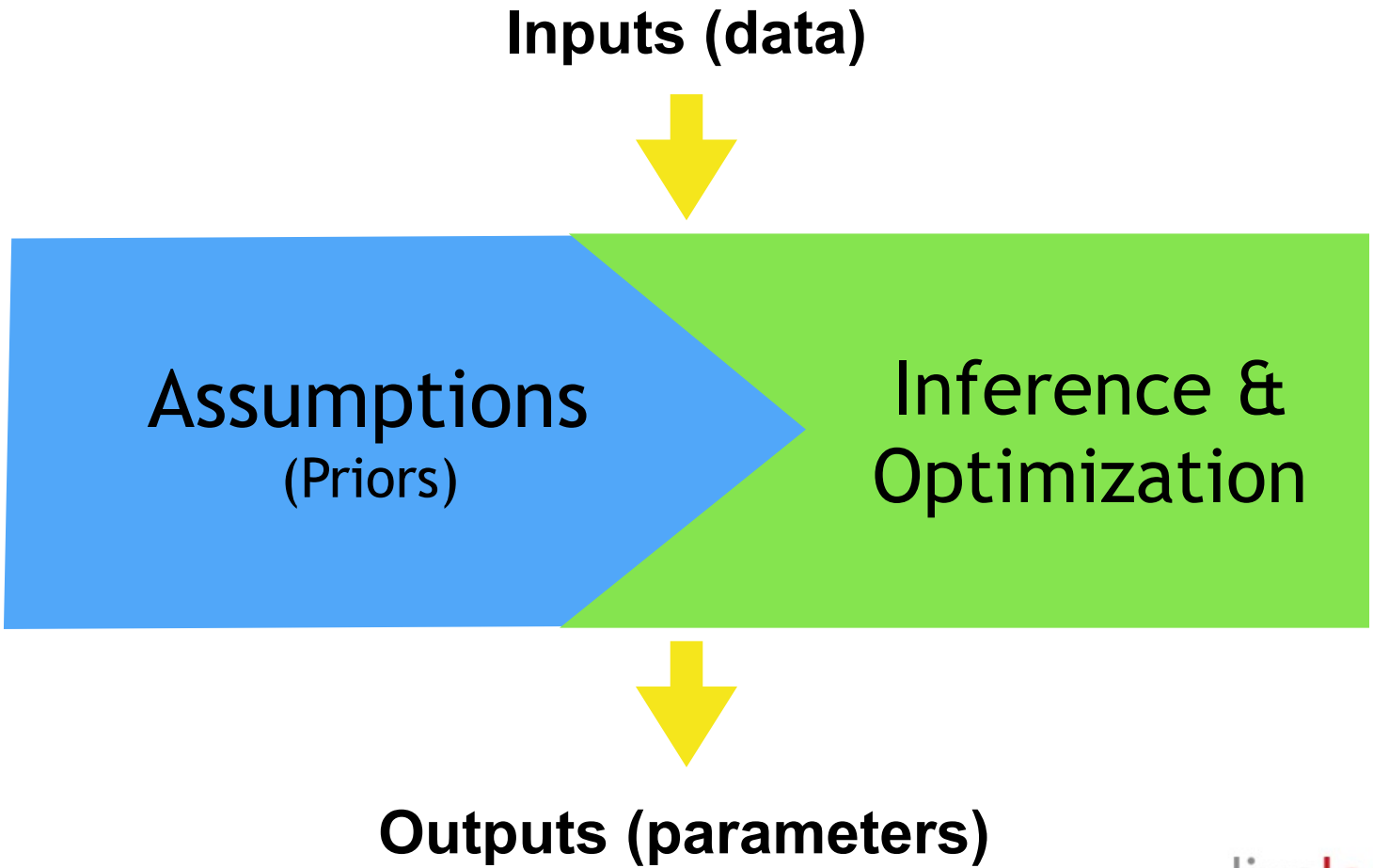


Figure 1. Learning Curves for Confusion Set Disambiguation

Source: https://www.microsoft.com/en-us/research/publication/scaling-to-very-very-large-corpora-for-natural-language-disambiguation/

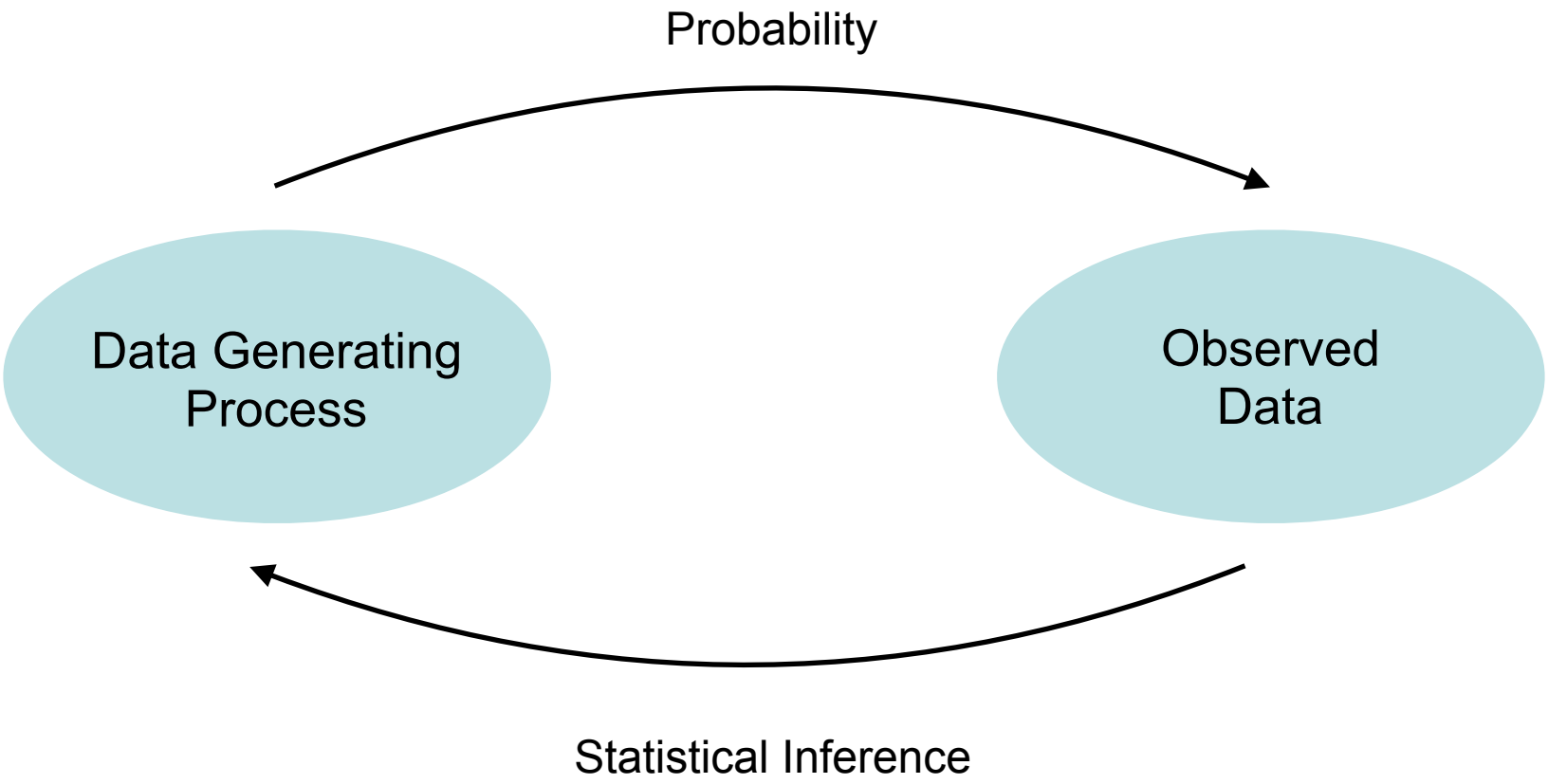


inspiration: http://www.mbmlbook.com/



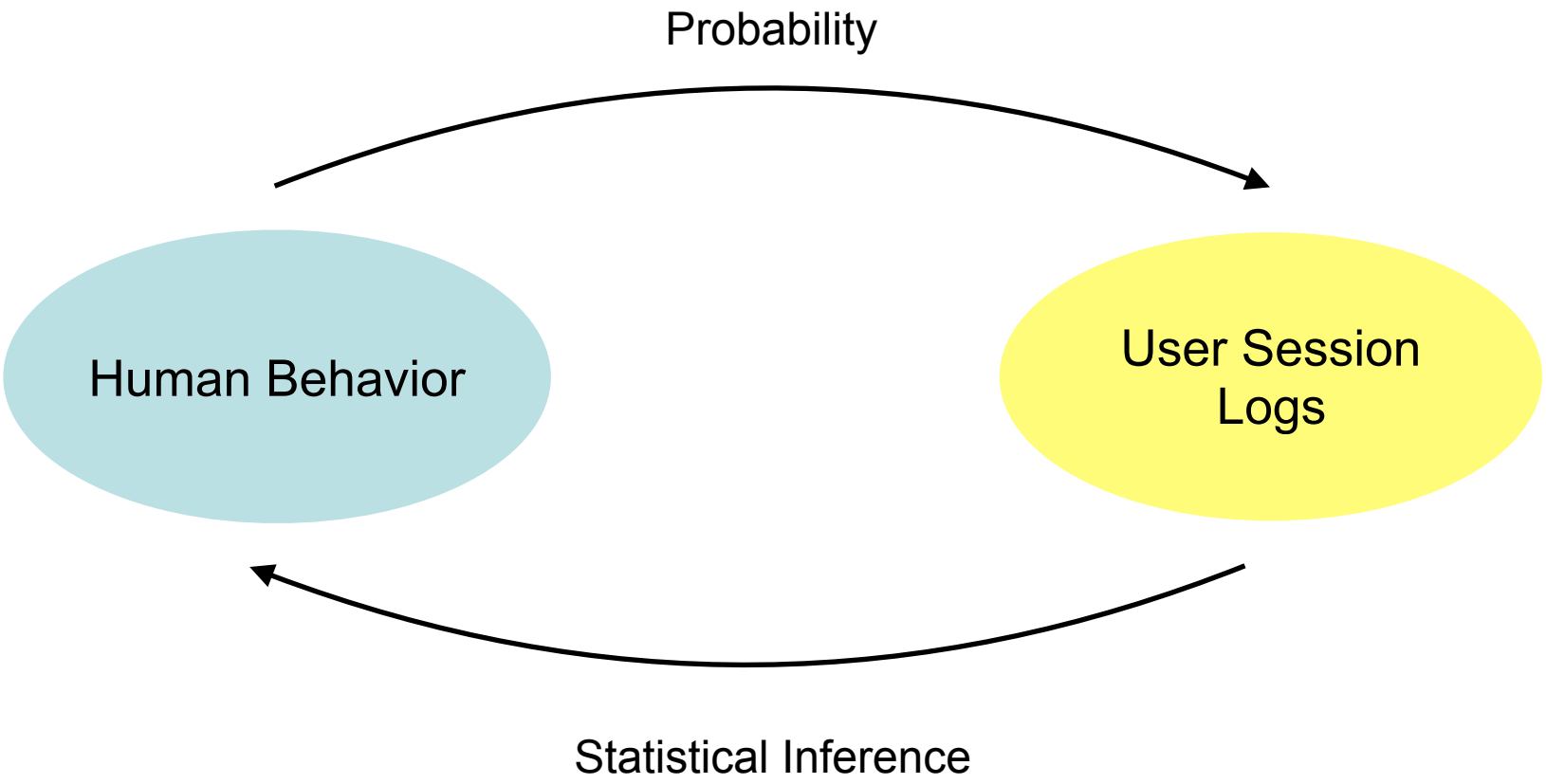
inspiration: http://www.mbmlbook.com/

Probability and Inference



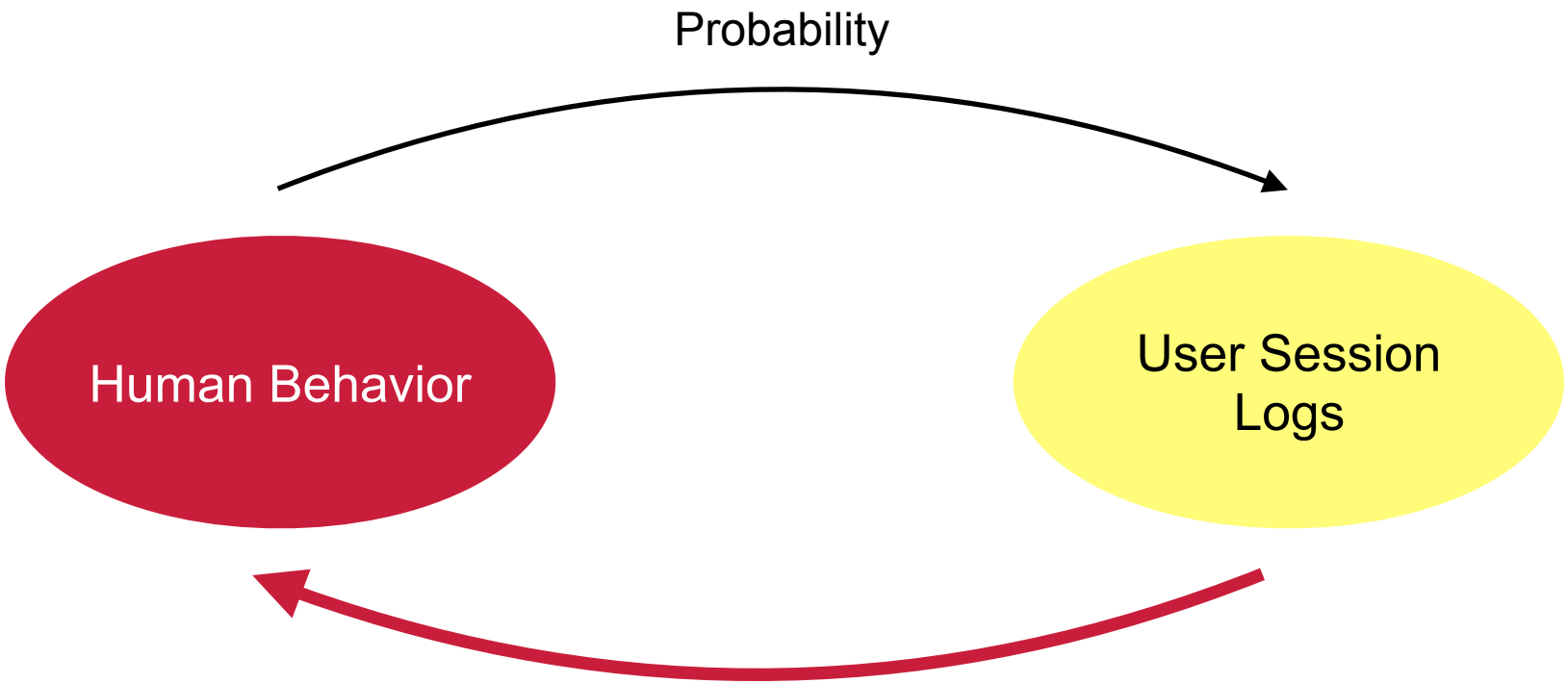
source: All of Statistics, Larry Wasserman

AirBnB



source: All of Statistics, Larry Wasserman

AirBnB



source: All of Statistics, Larry Wasserman

Data is proxy for people and processes

A (mathematical) representation for DATA…

Can’t Actually get that far without talking about probability…

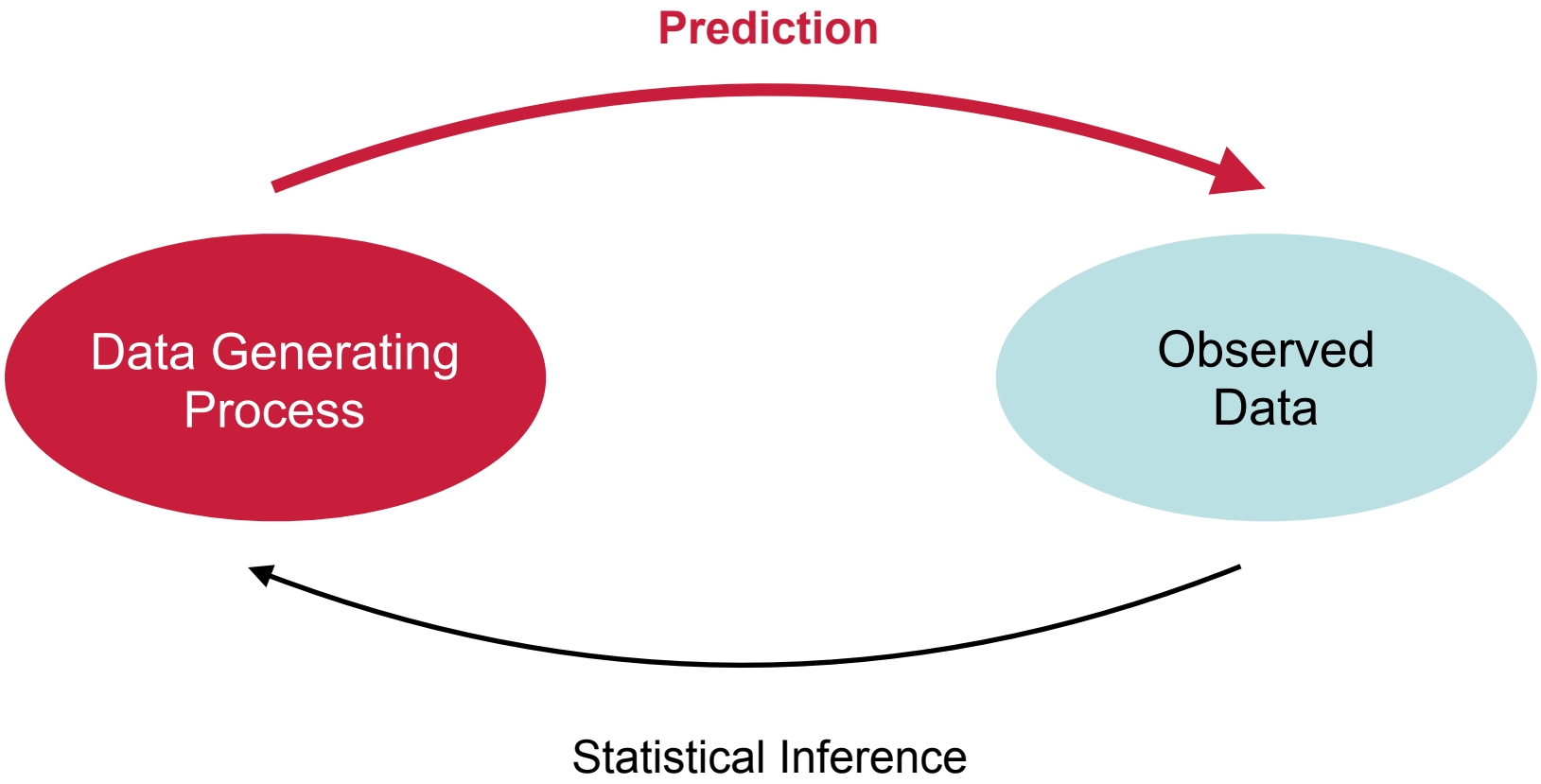
Parametric or Non-parametric

Both rely on the ideas of sampling and stochasticity

Also…

The problem with non-parametric statistics is going the other way

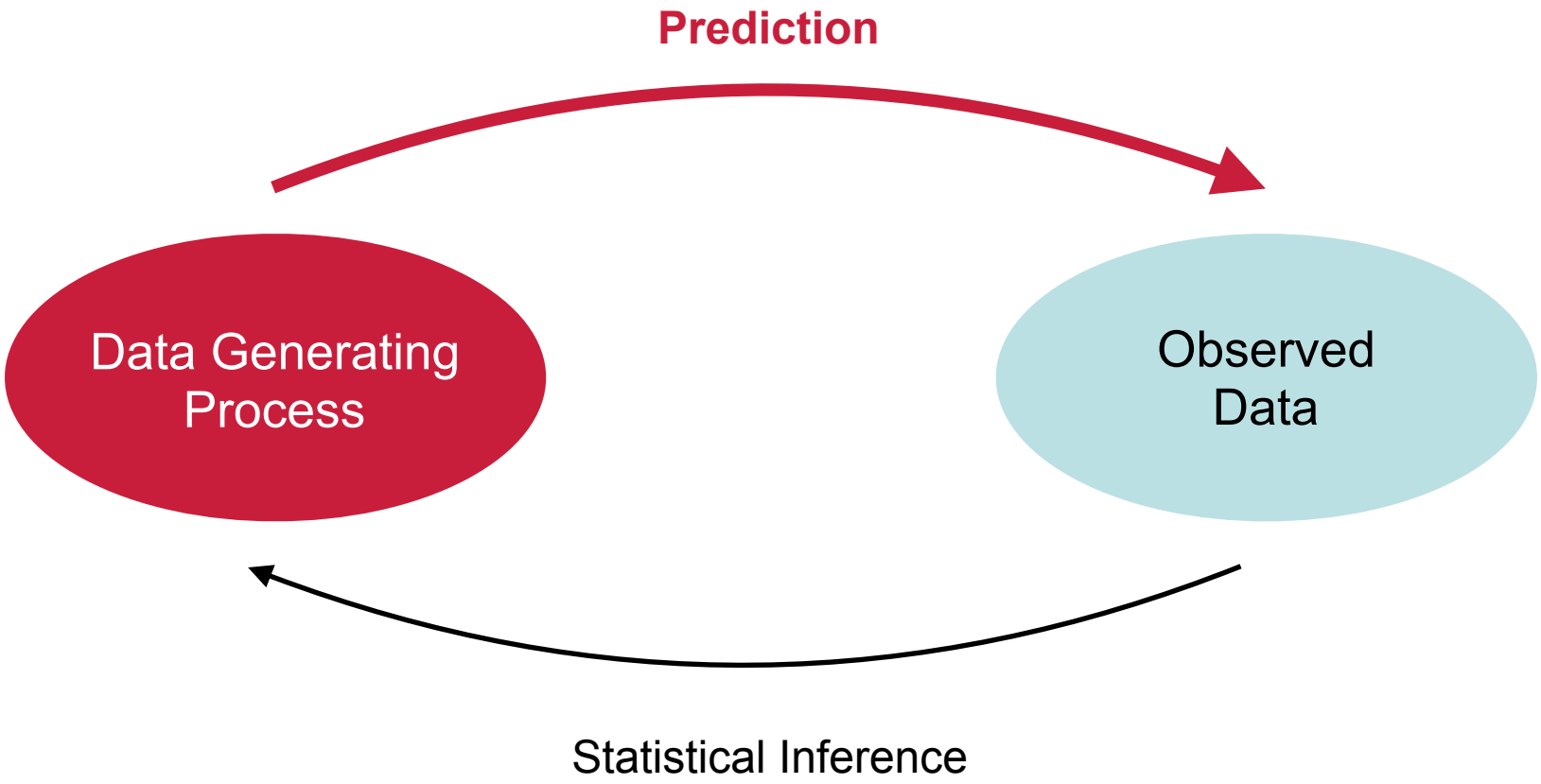
Holy Grail

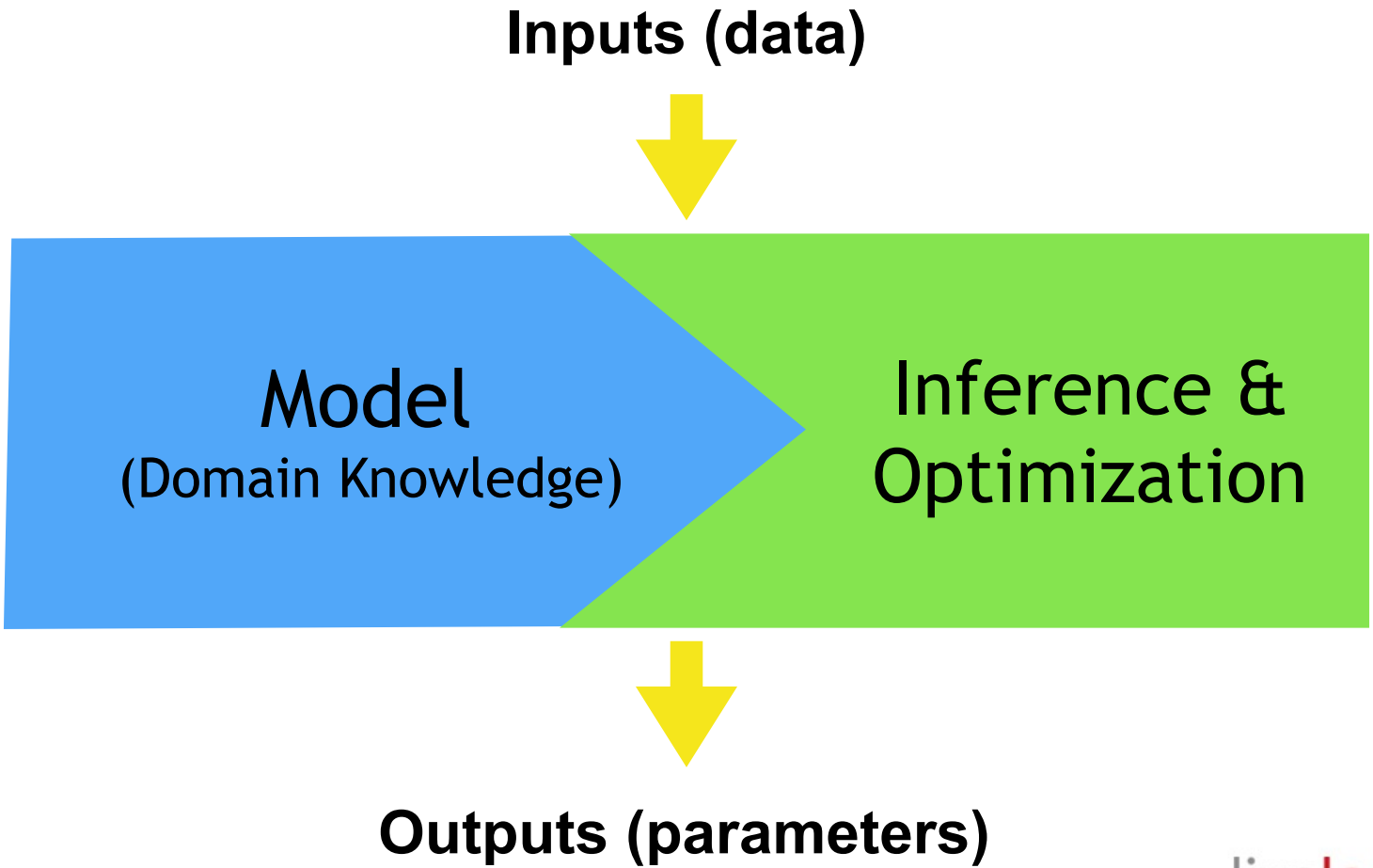


Without a generative model we can't go forward

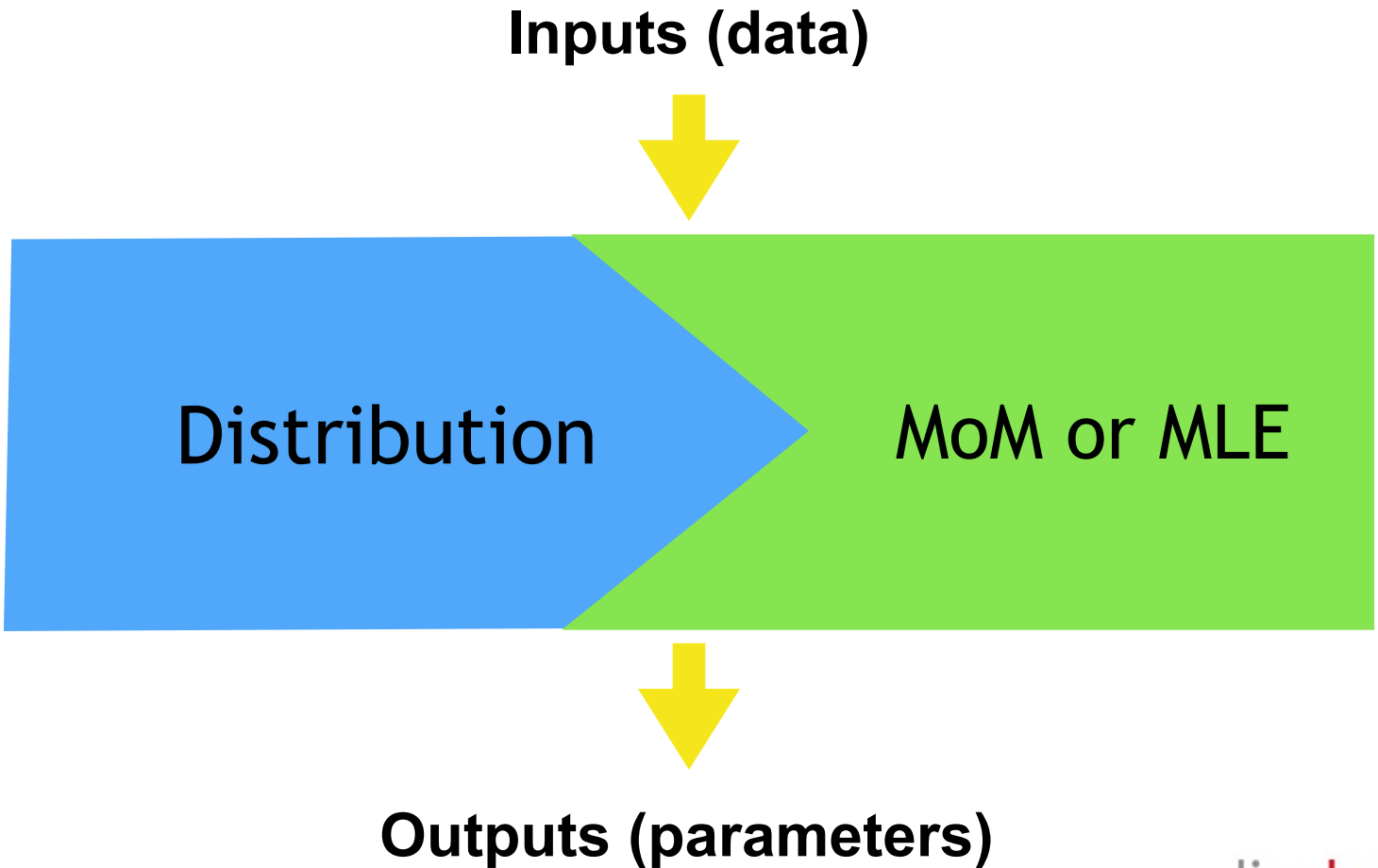
Lets take a step back

Specifying a Model

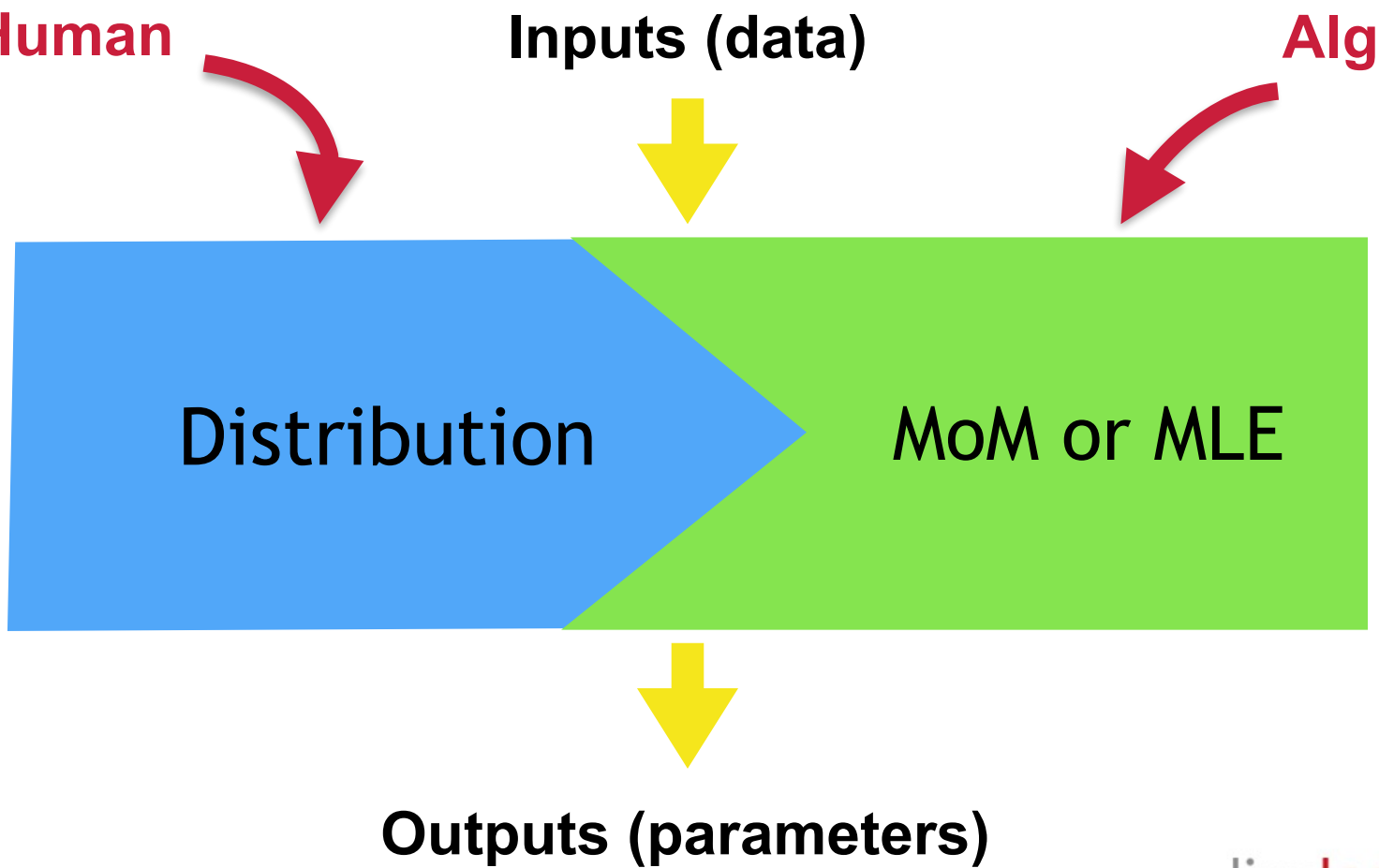




inspiration: http://www.mbmlbook.com/



inspiration: http://www.mbmlbook.com/



inspiration: http://www.mbmlbook.com/

Assume Ratings are Normally Distributed

Distributions

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Distributions

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Parameters

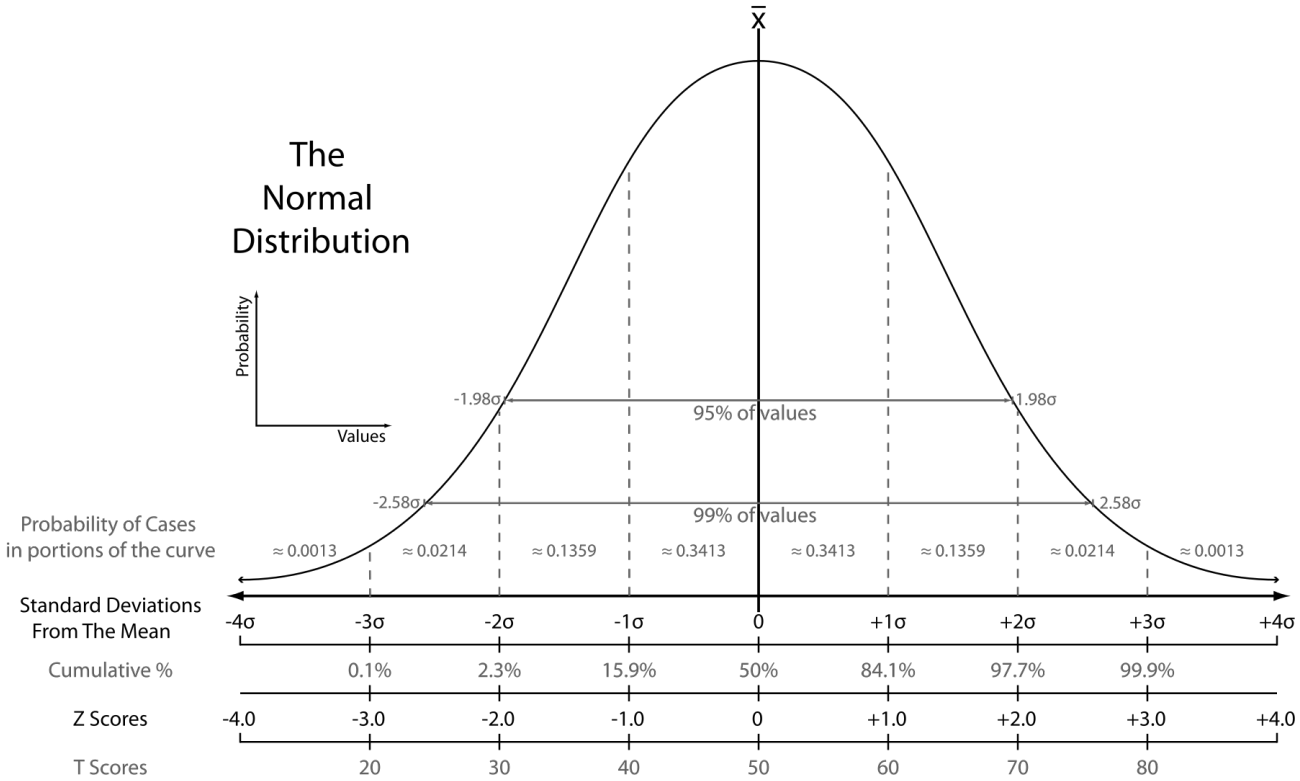
Distributions

Data

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

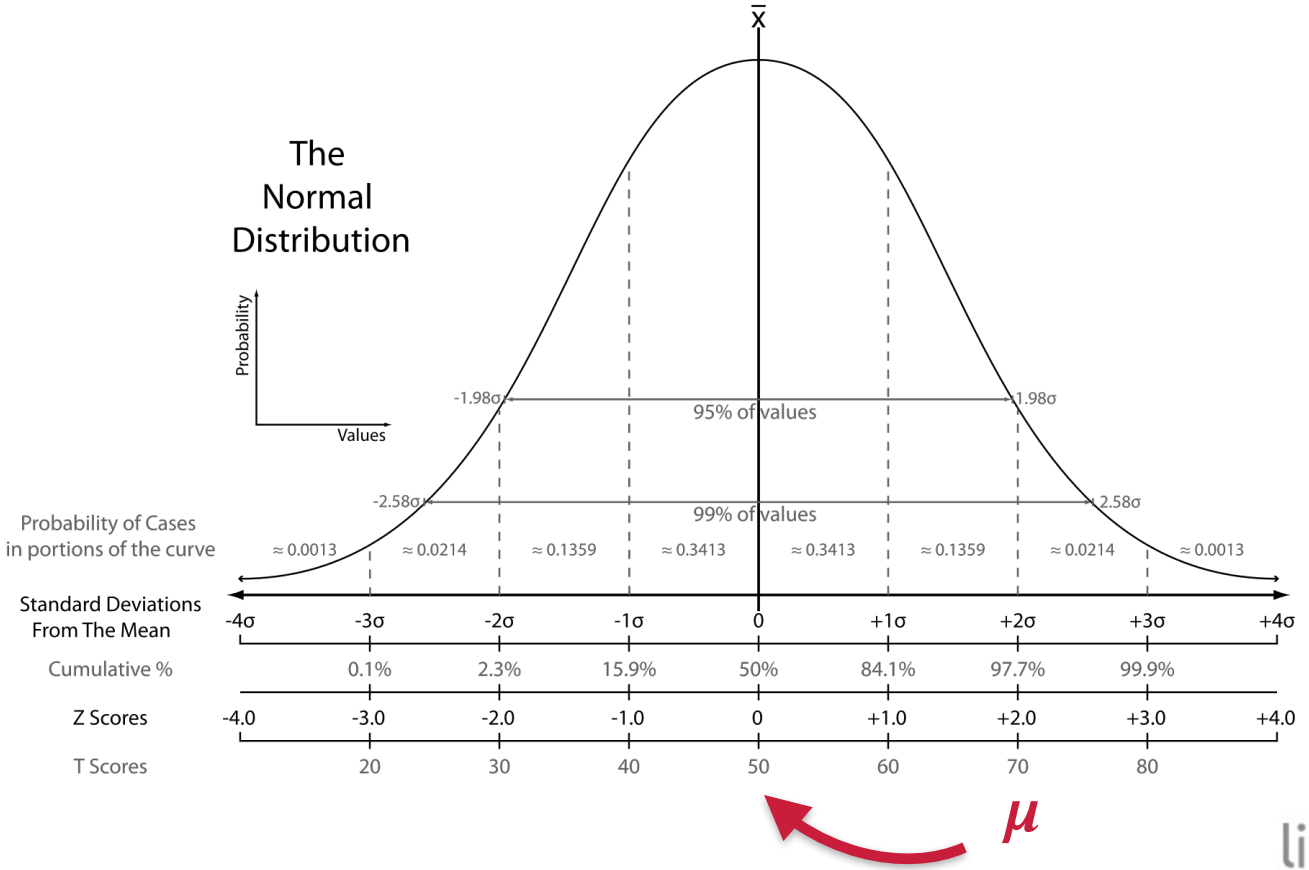
Parameters

The Normal Distribution



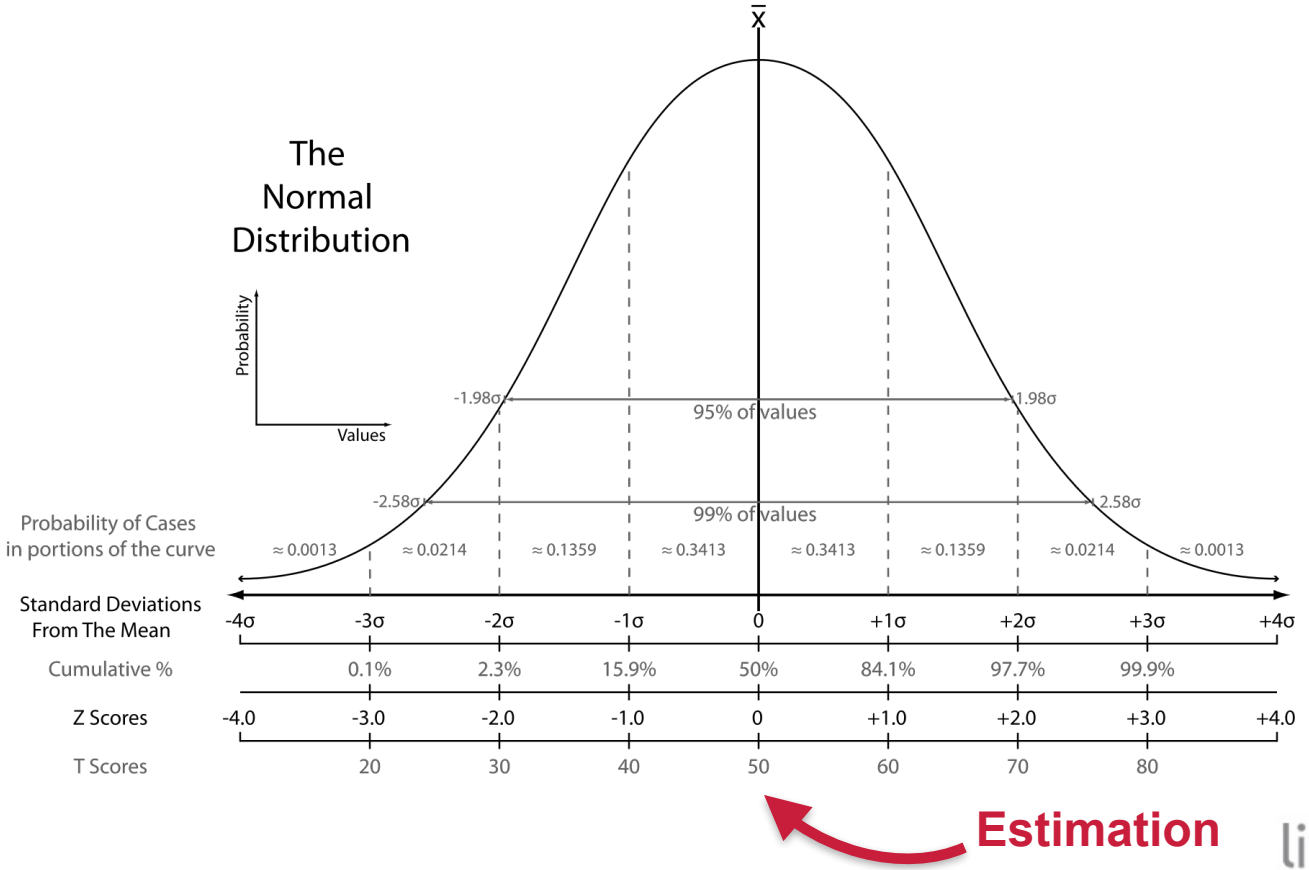
source: Wikipedia

The Normal Distribution



source: Wikipedia

The Normal Distribution



Estimation

source: Wikipedia

Moments (mean, variance, skewness, etc.)

Moments

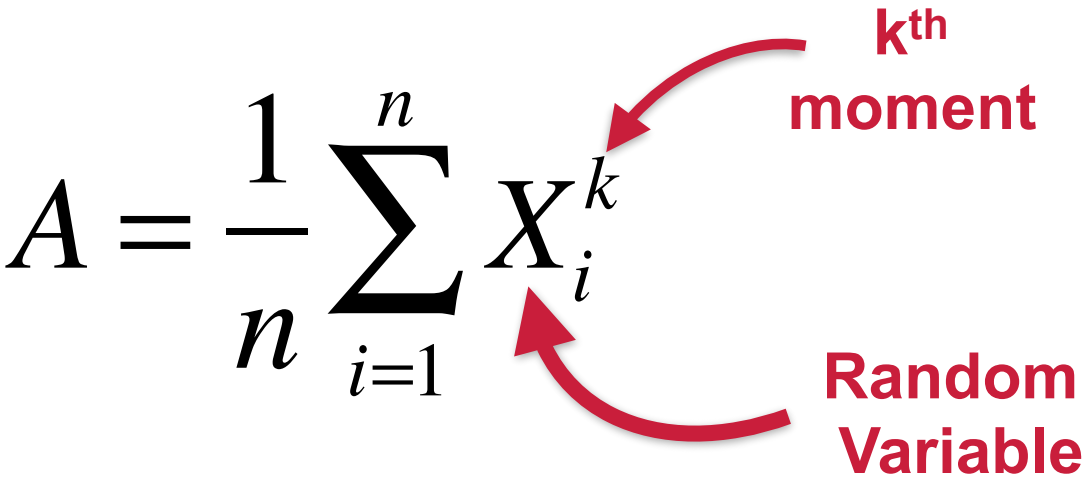
$$A = \frac{1}{n} \sum_{i=1}^{n} X_i^k$$

Moments

$$A = \frac{1}{n} \sum_{i=1}^{n} X_i^k$$

Random Variable

Moments



Mean <- First Moment

$$\mu = \frac{1}{n}\sum_{i=1}^{n} X_i^1$$

Mean <- First Moment

$$\mu = \frac{1}{n}\sum_{i=1}^{n} X_i$$

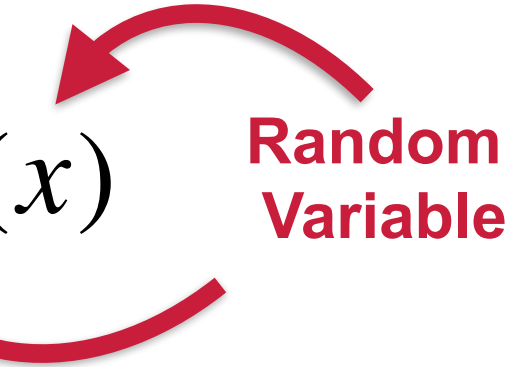
Mean <- First Moment

$$\mu = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Population Samples

Expected Value

$$\mu = \sum xP(x)$$



Expected Winnings

*At a casino, a die game cost **3 dollars** to play and pays out a number of dollars equal to the **die number***

Expected Winnings

Should you play?

Expected Winnings

$$Winnings = \sum_{i=1}^{n} x - 3$$

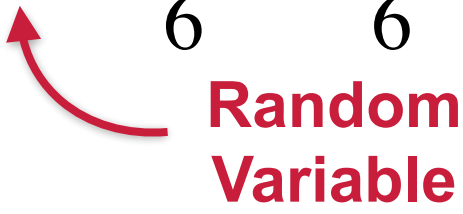
Expected Winnings

$$E[X] = \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 2 + \frac{1}{6} \cdot 3 + \frac{1}{6} \cdot 4 + \frac{1}{6} \cdot 5 + \frac{1}{6} \cdot 6$$

The outcome of a (fair) six sided die

Expected Winnings

$$E[X] = \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 2 + \frac{1}{6} \cdot 3 + \frac{1}{6} \cdot 4 + \frac{1}{6} \cdot 5 + \frac{1}{6} \cdot 6$$



Expected Winnings

$$E[X] = \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 2 + \frac{1}{6} \cdot 3 + \frac{1}{6} \cdot 4 + \frac{1}{6} \cdot 5 + \frac{1}{6} \cdot 6 = 3.5$$

Expected Winnings

What if whenever a 6 comes up the casino always wins (and pays out $0)?

Expected Winnings

$$E[X] = \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 2 + \frac{1}{6} \cdot 3 + \frac{1}{6} \cdot 4 + \frac{1}{6} \cdot 5 + \frac{1}{6} \cdot 0 = 2.5$$

Expected Winnings

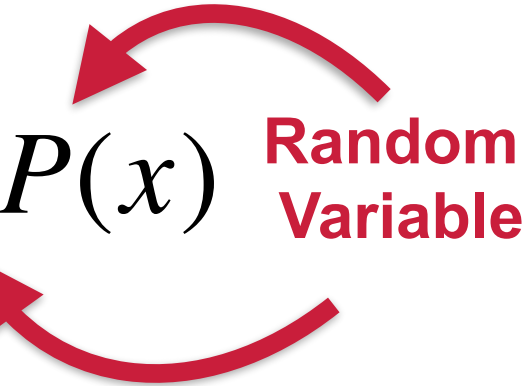
What if the casino uses a special dice that only has the numbers 1-4 but even numbers appear twice?

Expected Winnings

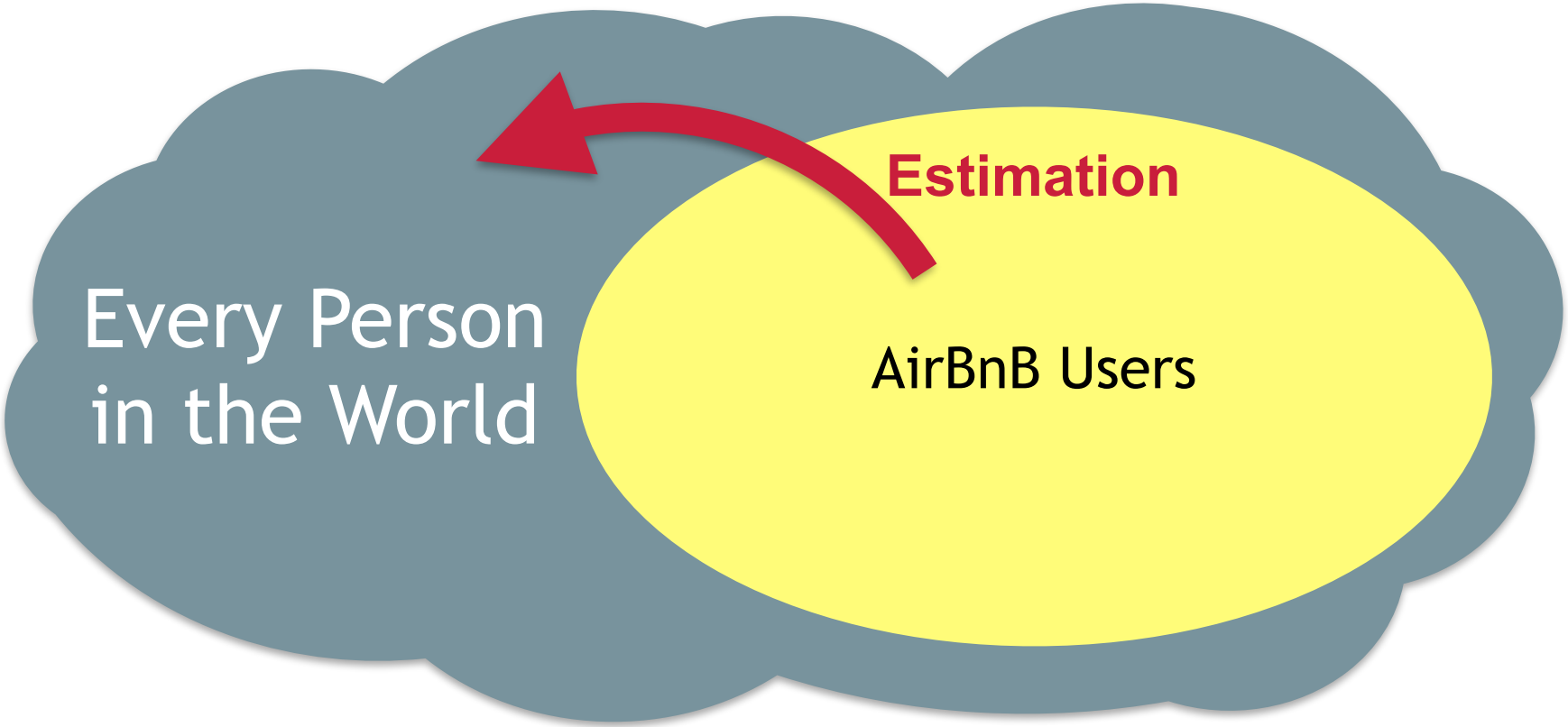
$$E[X] = \frac{1}{6} \cdot 1 + \boxed{\frac{2}{6}} \cdot 2 + \frac{1}{6} \cdot 3 + \boxed{\frac{2}{6}} \cdot 4 = 2.66$$

Expected Earnings

$$E[X] = \sum_{i=1}^{n} xP(x)$$



Remember



MoM of Reviews

```
listing = pd.read_sql('listing', \
 'sqlite:///data/better_breakfasts_complete.db')

listing.reviews_per_month.mean()
```

Issues

- Point Statistic (one number trying to represent many)
- Often not the best (lowest error) estimator

https://www.yelp.com/biz/yelp-san-francisco

yelp

Find tacos, cheap dinner, Max's

Near Noe Valley, San Francisco, CA

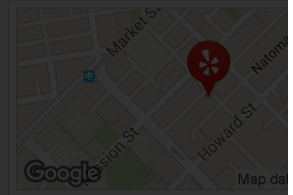
Home About Me Write a Review Find Friends Messages Talk Events

San Francisco › Financial District ›

Yelp

8381 reviews

Local Flavor, Mass Media



140 New Montgomery St
San Francisco, CA 94105
b/t Natoma St & Minna St
Financial District, SoMa

Get Directions

(415) 908-3801

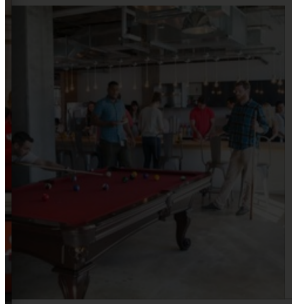
officialblog.yelp.com

"Locally, I enjoy bei
way I do about writ

"Reviews galore, ta
know, love, hate or

"Of course, all thos
forward to attend m

Photo Share Bookmark



See all 1240

8:00 am - 6:00 pm Open now

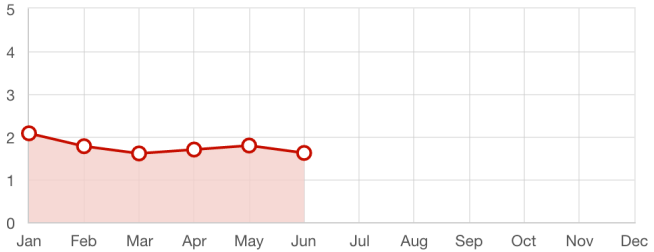
:00 am - 6:00 pm
:00 am - 6:00 pm
:00 am - 6:00 pm
:00 am - 6:00 pm Open now
:00 am - 6:00 pm
Closed

Sun Closed

Rating Details

Monthly Trend

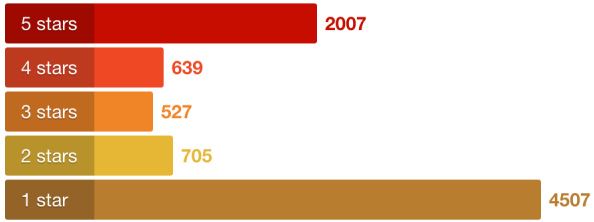
2016 2015 2014 2013 2012



Understand how a business' rating changes month-to-month. Learn more.

Overall Rating

Yelping since 2004 with 8381 reviews



We calculate the overall star rating using only reviews that our automated software currently recommends. Learn more.

https://www.yelp.com/biz/yelp-san-francisco

yelp

Find tacos, cheap dinner, Max's

Near Noe Valley, San Francisco, CA

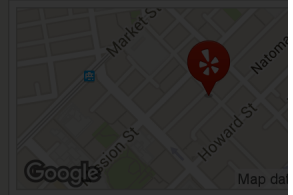
Home About Me Write a Review Find Friends Messages Talk Events

San Francisco › Financial District ›

Yelp

8381 reviews

Local Flavor, Mass Media



140 New Montgomery St
San Francisco, CA 94105
b/t Natoma St & Minna St
Financial District, SoMa

Get Directions

(415) 908-3801

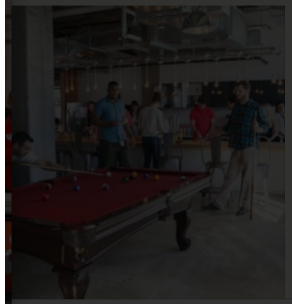
officialblog.yelp.com

"Locally, I enjoy bei... way I do about writ...

"Reviews galore, ta... know, love, hate or...

"Of course, all thos... forward to attend m...

Photo Share Bookmark



See all 1240

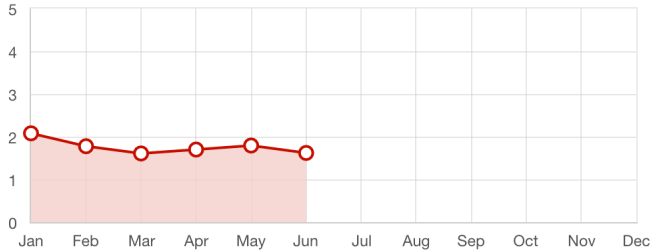
8:00 am - 6:00 pm Open now

:00 am - 6:00 pm
:00 am - 6:00 pm
:00 am - 6:00 pm
:00 am - 6:00 pm Open now
:00 am - 6:00 pm
Closed

Sun Closed

Rating Details

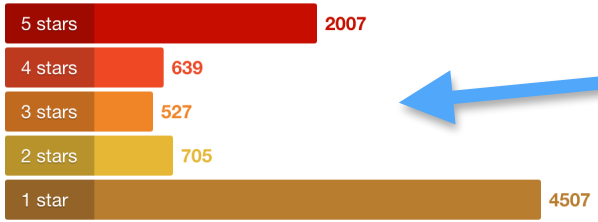
Monthly Trend 2016 2015 2014 2013 2012



Understand how a business' rating changes month-to-month. Learn more.

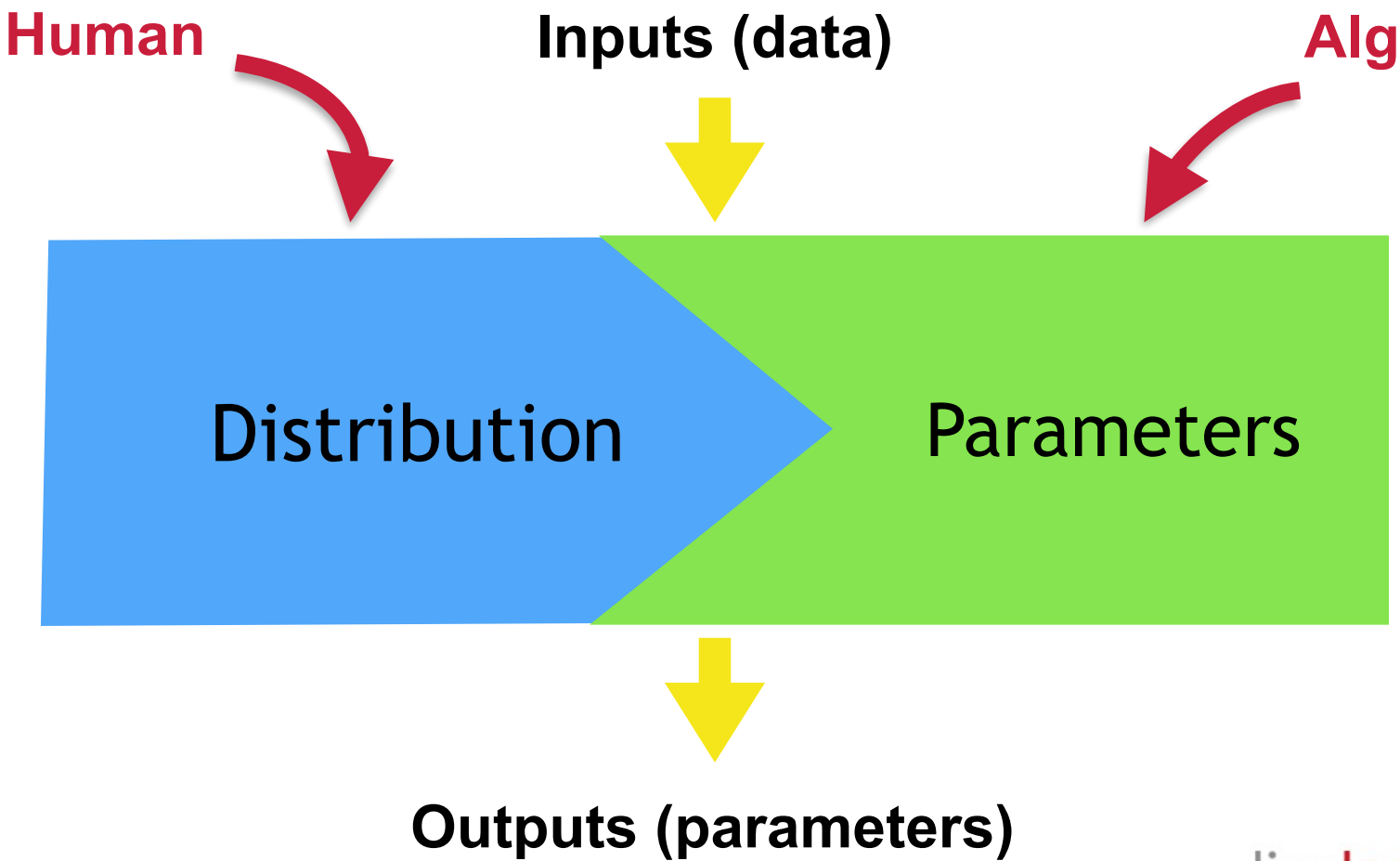
Overall Rating

Yelping since 2004 with 8381 reviews



We calculate the overall star rating using only reviews that our automated software currently recommends. Learn more.

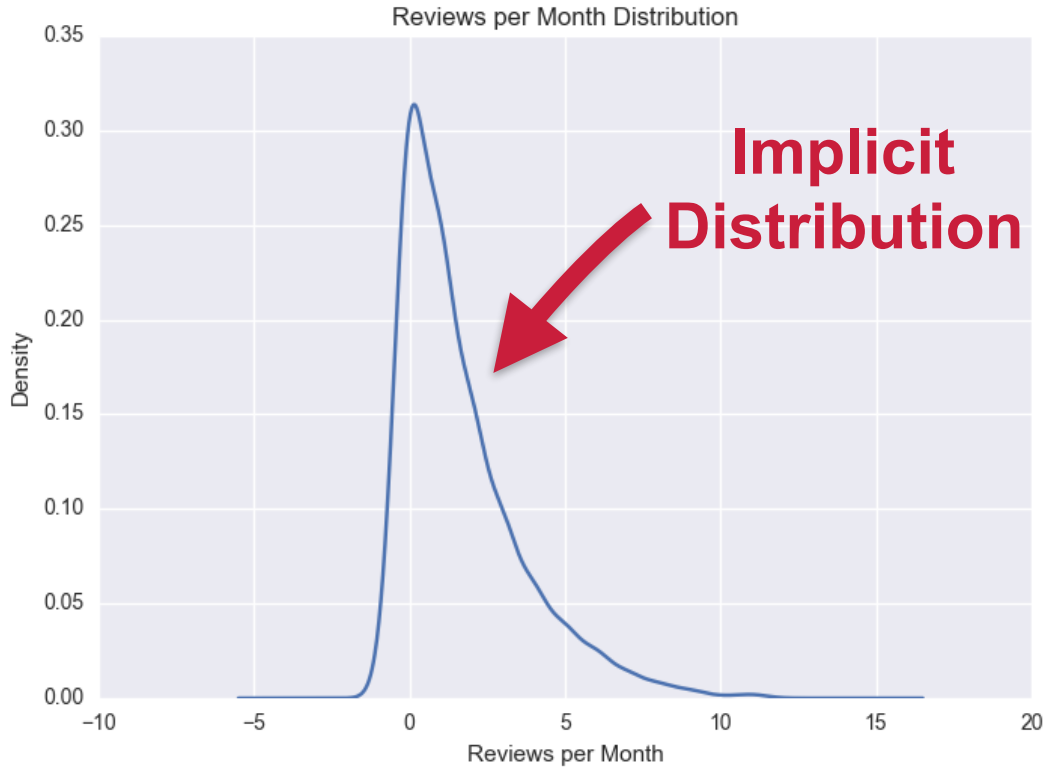
Definitely not normal



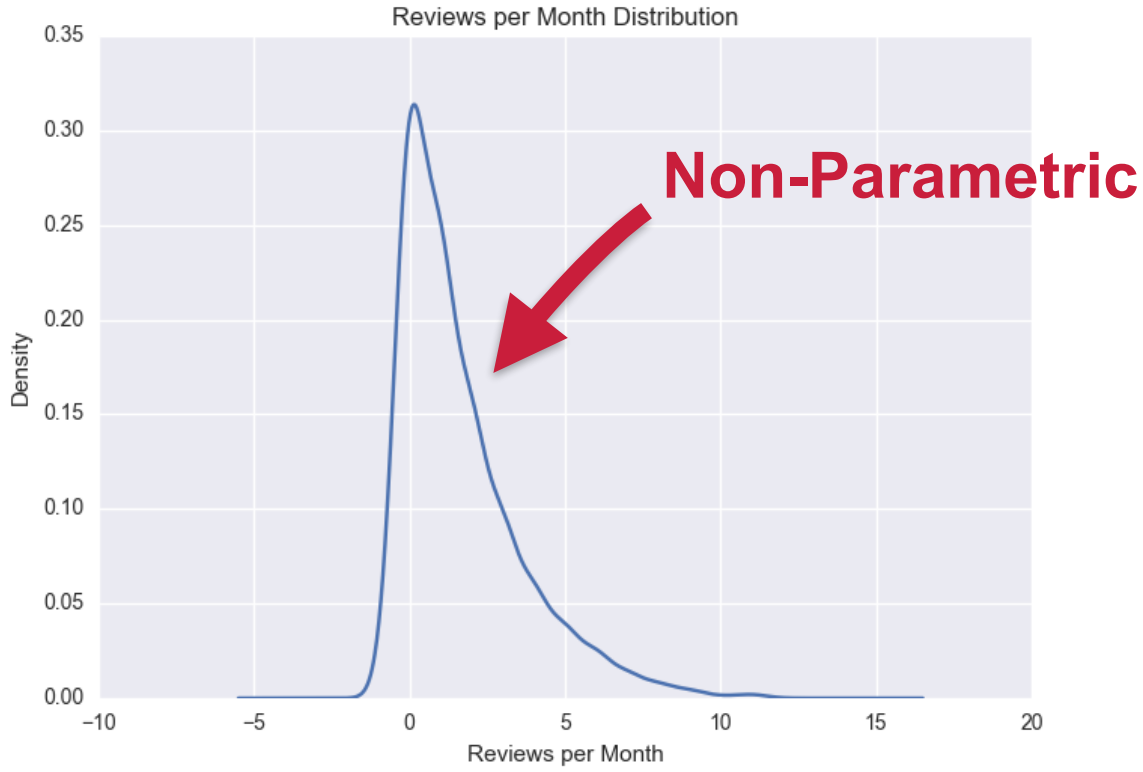
inspiration: http://www.mbmlbook.com/

Reviews per Month

Reviews per Month



Reviews per Month



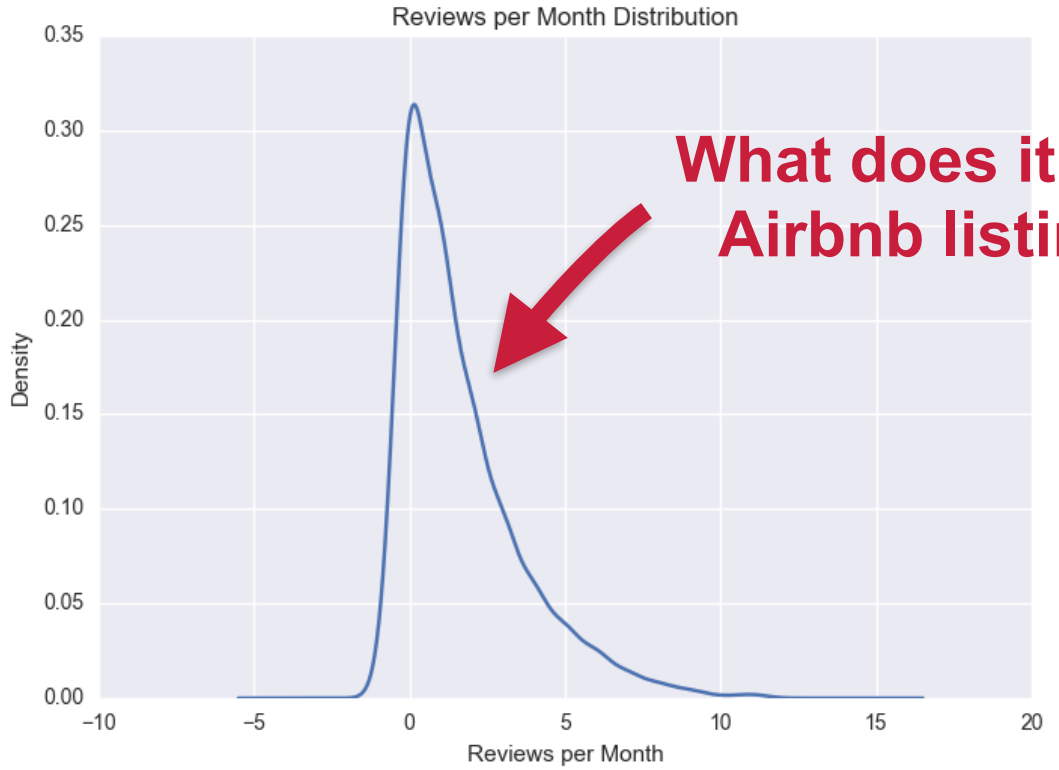
Reviews per Month

Reviews per Month

Hard to make general inferences

Reviews per Month

Reviews per Month



What does it tell us about Airbnb listing activity?

Building a Model

1. Specify Distribution
2. Estimate distribution parameters from data
3. Evaluate

What are the most likely parameters given the data we have?

$$P(\theta \mid x_1, x_2, \ldots, x_n)$$

$$P(\theta \mid x_1, x_2, \ldots, x_n) = P(\theta \mid x_1) \times P(\theta \mid x_2) \times \cdots \times P(\theta \mid x_n)$$

What is this?

$$P(\theta \mid x_1, x_2, \ldots, x_n) = P(\theta \mid x_1) \times P(\theta \mid x_2) \times \cdots \times P(\theta \mid x_n)$$

What parameters make the data most likely?

$$P(x_1, x_2, \ldots, x_n \mid \theta)$$

Likelihood

$$L(\theta; x_1, x_2, \ldots, x_n) = P(x_1, x_2, \ldots, x_n \mid \theta)$$

$$P(x_1, x_2, \ldots, x_n \mid \theta) = P(x_1 \mid \theta) \times P(x_2 \mid \theta) \times \cdots \times P(x_n \mid \theta)$$

What is this?

$$P(x_1, x_2, \ldots, x_n \mid \theta) = P(x_1 \mid \theta) \times P(x_2 \mid \theta) \times \cdots \times P(x_n \mid \theta)$$

Just probability mass (or density) function

$$P(x_1, x_2, \ldots, x_n \mid \theta) = P(X = x_1; \theta) \times P(x_2 \mid \theta) \times \cdots \times P(x_n \mid \theta)$$

$$P(x_1, x_2, \ldots, x_n \mid \theta) = \prod_{i}^{n} P(x_i \mid \theta)$$

$$L(\theta; x_1, x_2, \ldots, x_n) = \prod_i^n P(x_i \mid \theta)$$

$$L(\theta; x_1, x_2, \ldots, x_n) = \prod_{i}^{n} P(x_i \mid \theta)$$

Solving for Parameters

1. Analytically with differential calculus
2. Computationally with optimization methods
3. Approximately with iterative methods

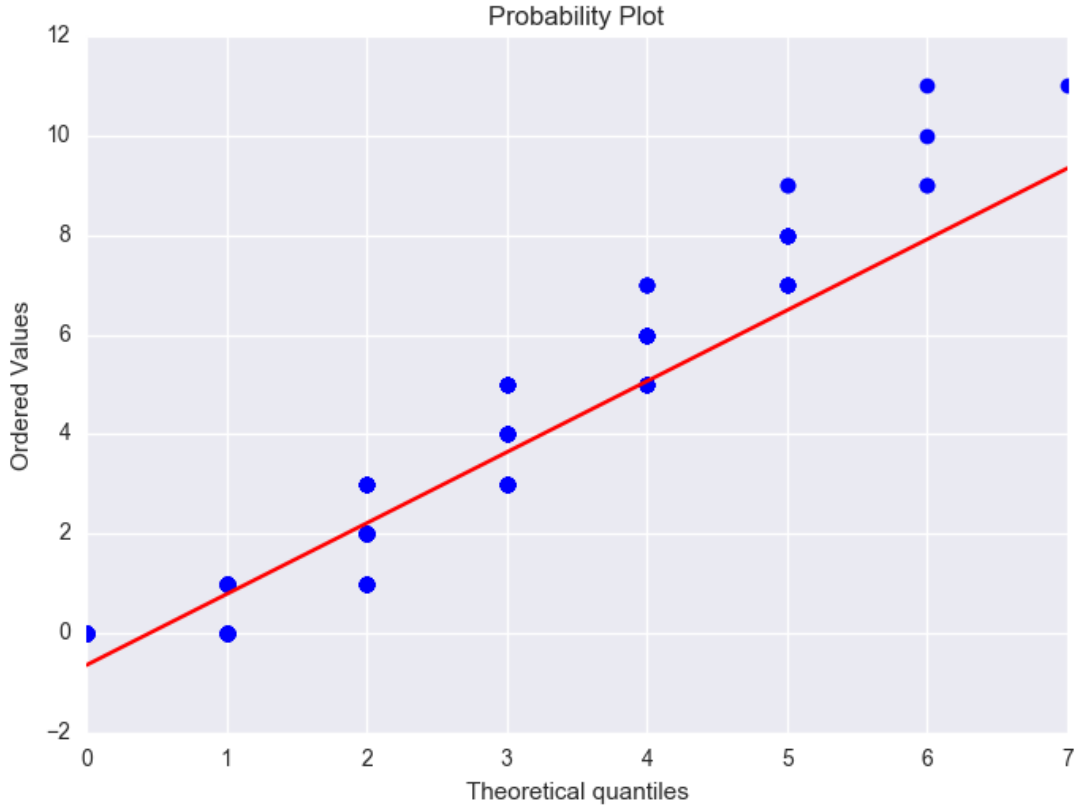
Solving for Parameters

1. Analytically with differential calculus
2. Computationally with optimization methods
3. **Approximately with iterative methods**

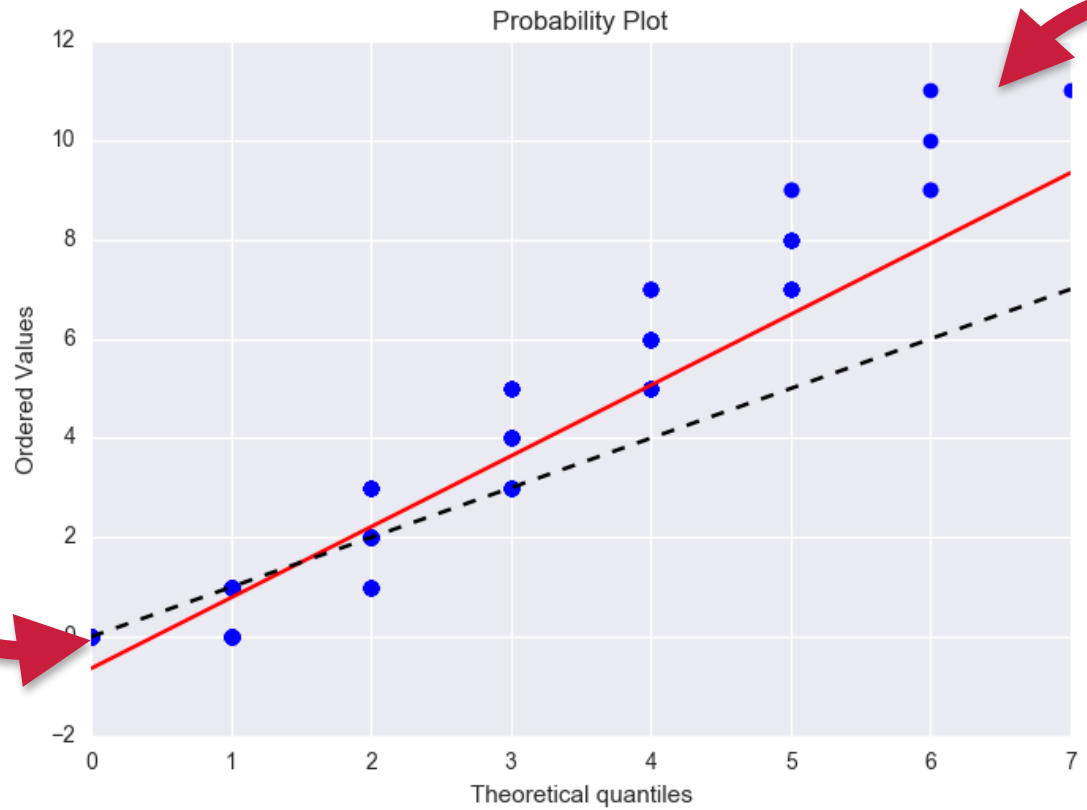
Estimation

Evaluation

Evaluation



Evaluation



Diverges

Diverges

Types of Learning

Supervised Learning

- Training Data includes desired output

Unsupervised Learning

- Training Data does not include desired output

Semi-supervised Learning

- Training Data includes some desired outputs

Reinforcement Learning

- Rewards from sequence of actions

Labeled Data

Polynomial

SVM

Linear

Regression

kNN

Classification

Trees

Logistic

Trees

Continuous Target

Discrete Target

K-means

Apriori

Clustering/
Dimensionality Reduction

Association Analysis

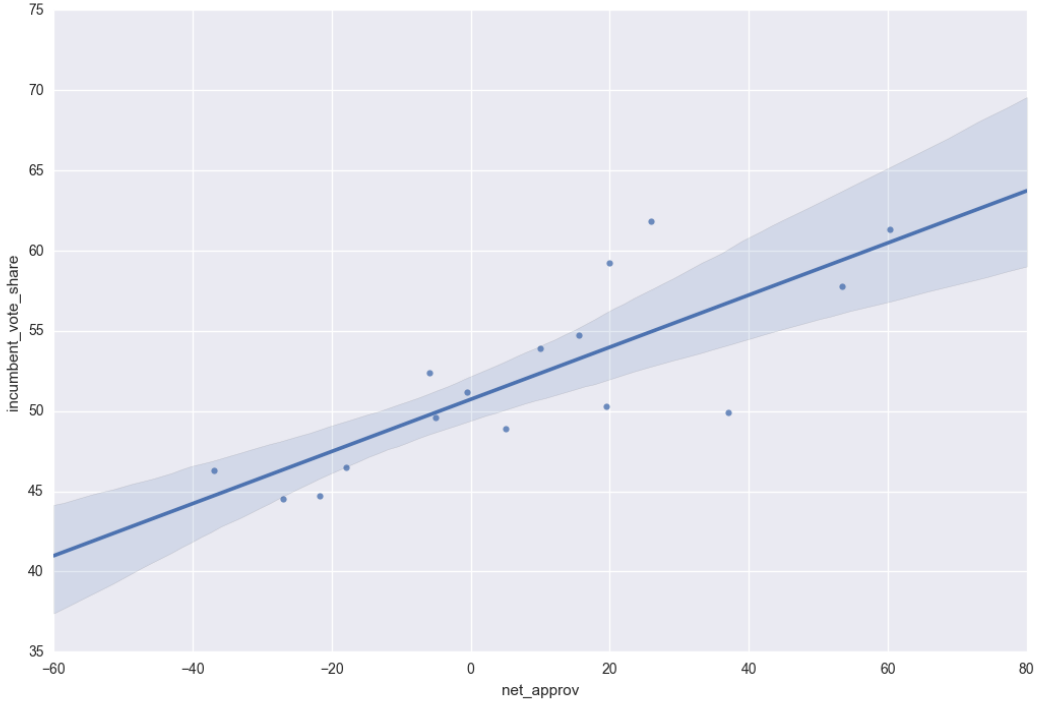
SVD

PCA

FP-Growth

Unlabeled Data

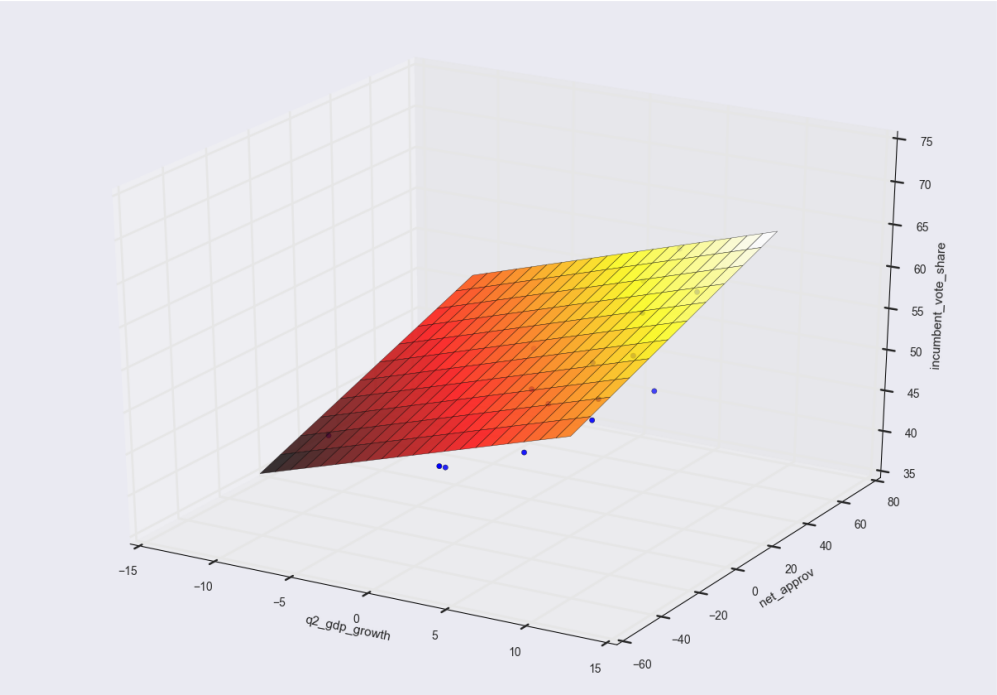
Regression: Single Variable



Parameters

$$y = \beta_0 + \beta_1 x_1$$

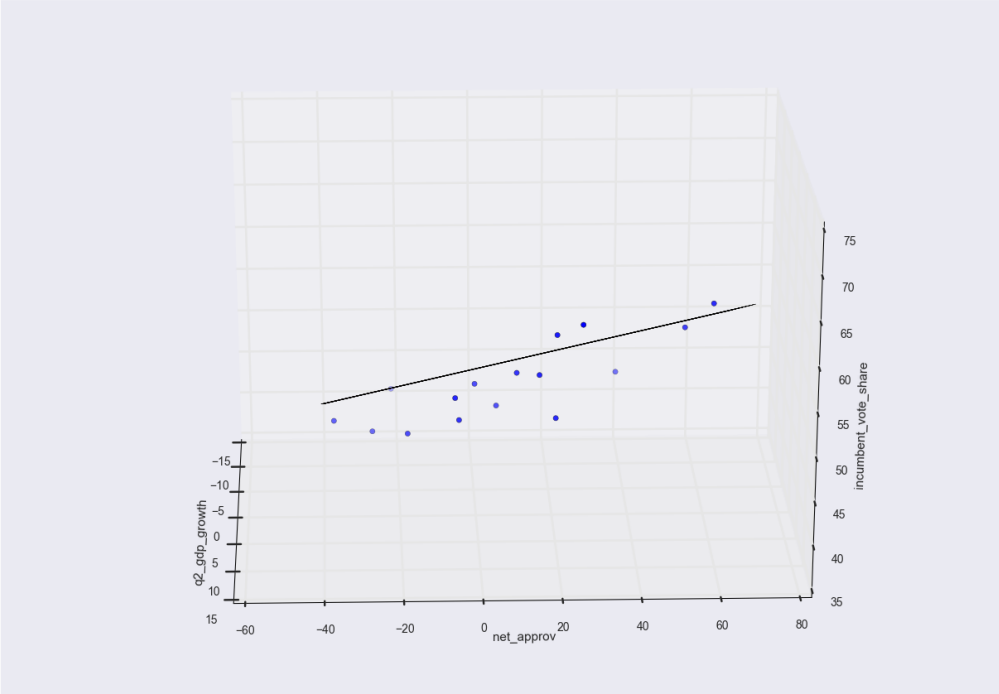
Regression: Multiple Variables



Parameters

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

Regression: Multiple Variables



Parameters

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

Model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \varepsilon$$

Hypothesis

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \varepsilon$$

Hypothesis: Time for Change Model

$$voteshare = \beta_0 + \beta_1 * q2GDP + \beta_2 * netapp - \beta_3 * twoterms$$

Source: http://sites.suffolk.edu/bconleygovt120/files/2009/08/abramowitz.pdf

Hypothesis: Time for Change Model

$$voteshare = 51.5 + 0.6 * q2GDP + 0.1 * netapp - 4.3 * twoterms$$

Source: http://sites.suffolk.edu/bconleygovt120/files/2009/08/abramowitz.pdf

Assumptions

- Independence
- Monotonicity and Linearity
- Random Noise Normally Distributed

Thought Experiment: Google

Given a new ad (query, creative, landing page) can we predict the bounce rate?

Regression Three Ways

- MLE (probabilistic)
- Normal Equation (analytical)
- Gradient Descent (computational)

Regression as MLE

Remember…. all a model is, is a set of assumptions about the world

Regression as MLE

Remember…. all a model is, is a set of assumptions about the data

Regression as MLE

$$y = f(X) + \varepsilon$$

Regression as MLE

$$y = f(X, \beta) + \varepsilon$$

Regression as MLE

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \varepsilon$$

Regression as MLE

Remember….
Normally distributed

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \varepsilon$$

Regression as MLE

Independently and Identically

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \varepsilon$$

Regression as MLE

$$f(\varepsilon \mid 0, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\varepsilon^2}{2\sigma^2}}$$

Regression as MLE

$$y = X\beta + \varepsilon$$

Regression as MLE

$$\varepsilon = y - X\beta$$

Regression as MLE

$$\varepsilon = y - X\beta$$

Regression as MLE

Randomness due to noise

Assume Fixed

$$\varepsilon = y - X\beta$$

Regression as MLE

$$p(y \mid X = x; \beta, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y - X\beta)^2}{2\sigma^2}}$$

$$L(y \mid X; \beta, \sigma^2) = \prod_{i}^{n} P(y_i \mid x_i; \beta, \sigma^2)$$

$$L(y \mid X; \beta, \sigma^2) = \prod_{i}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - x_i\beta)^2}{2\sigma^2}}$$

Least Squares

$$y = X\beta + \varepsilon$$

Least Squares

Prediction

$$\hat{y} = X\beta$$

Least Squares

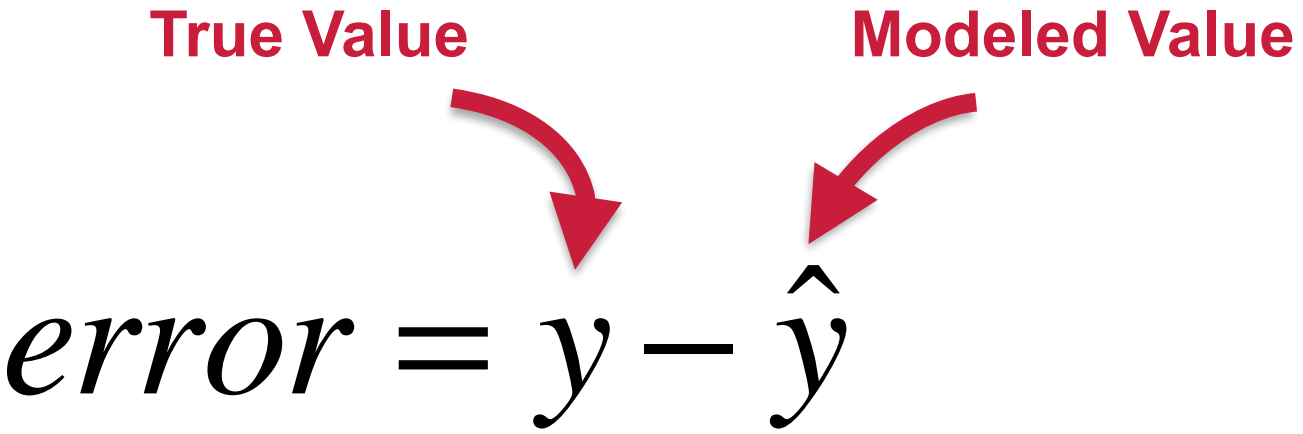
$$error = y - \hat{y}$$

Least Squares

True Value

$$error = y - \hat{y}$$

Least Squares



Least Squares

$$error = y - X\beta$$

Least Squares

Residuals

$$error = y - X\beta$$

Least Squares

$$RSS = (y - \hat{y})^2$$

Least Squares

Squared

$$RSS = (y - \hat{y})^2$$

Least Squares

$$RSS = (y - X\beta)^2$$

Least Squares

Minimize This

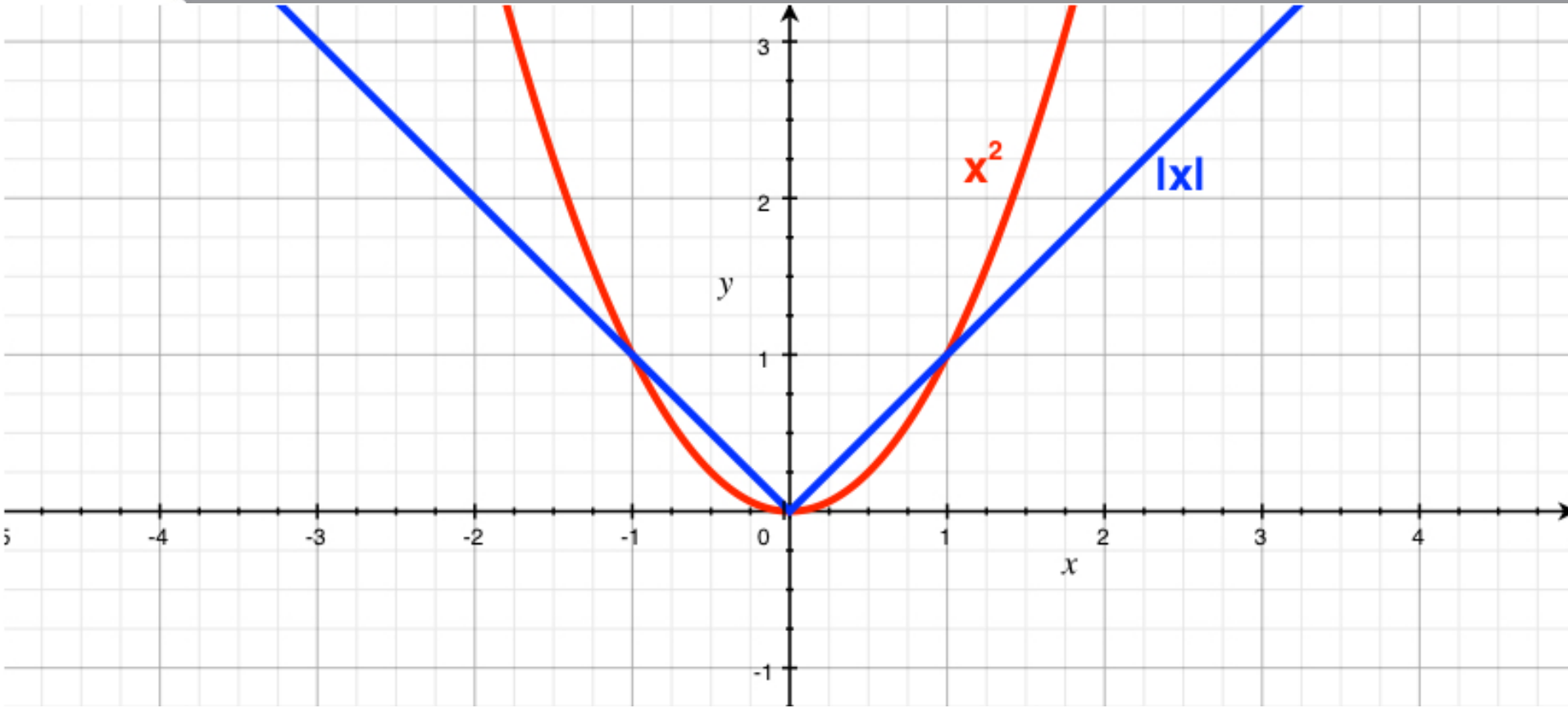
$$RSS = (y - X\beta)^2$$

Least Squares

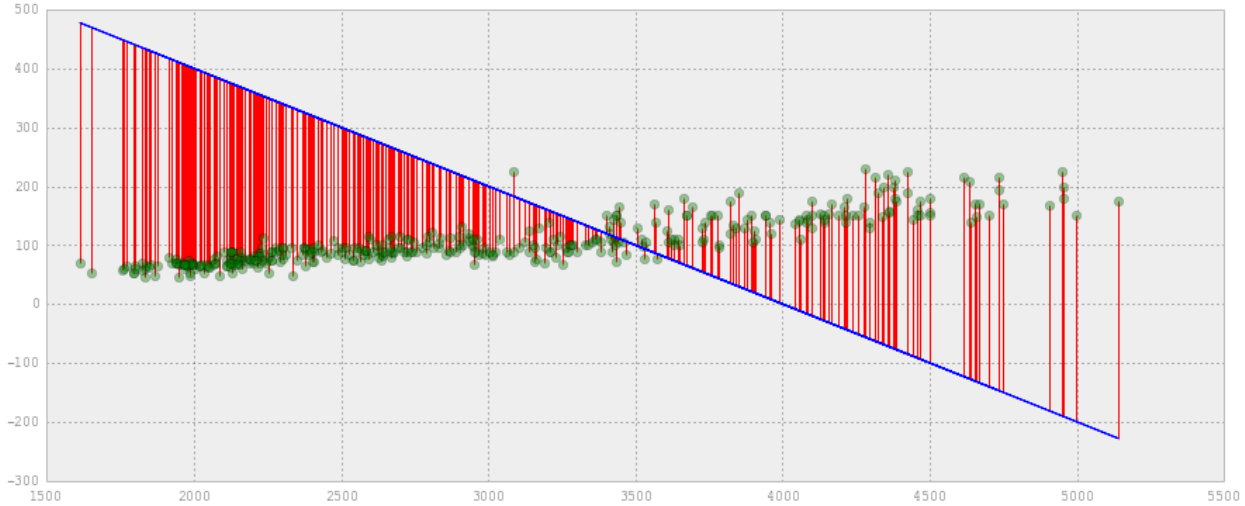
Loss

$$RSS = (y - X\beta)^2$$

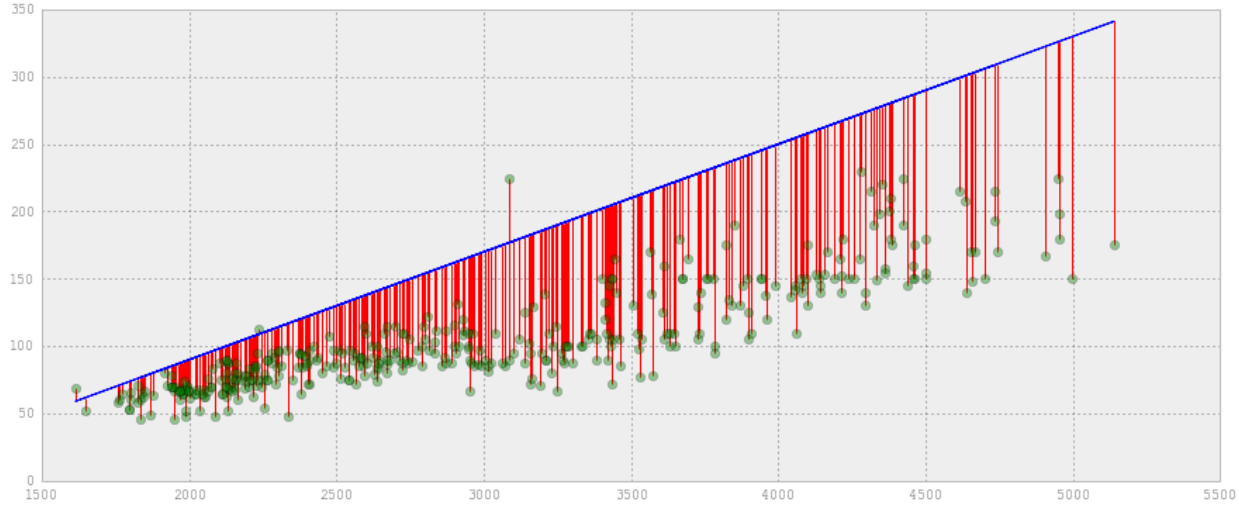
Least Squares



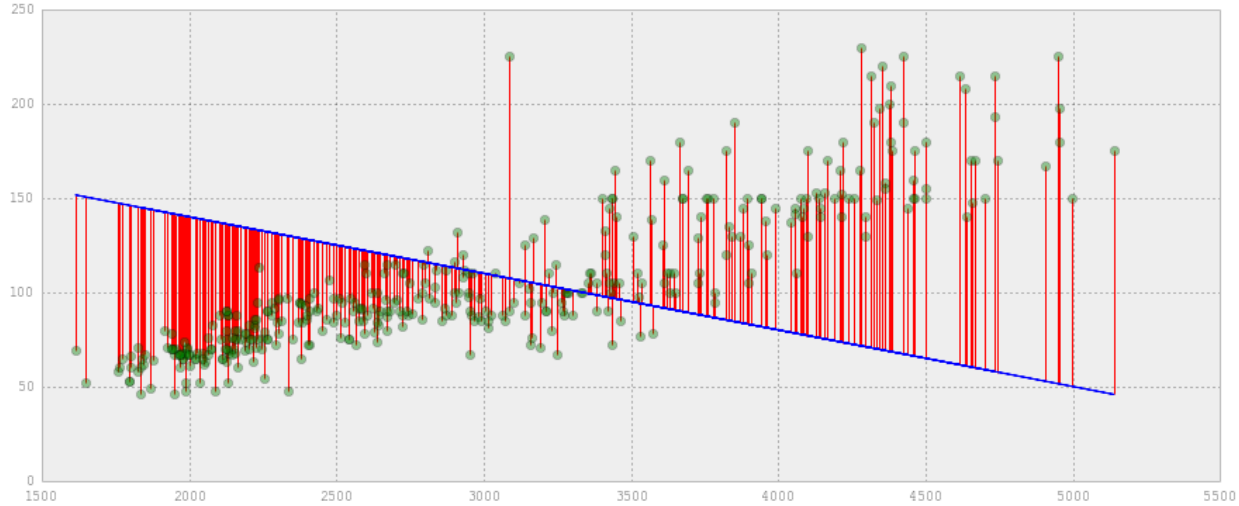
Residuals



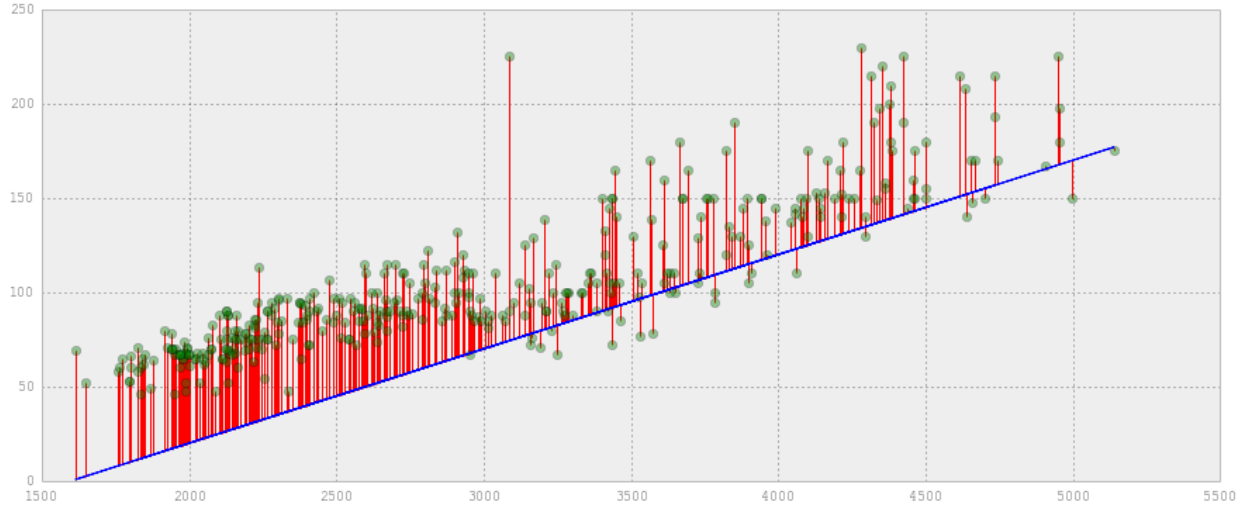
Residuals



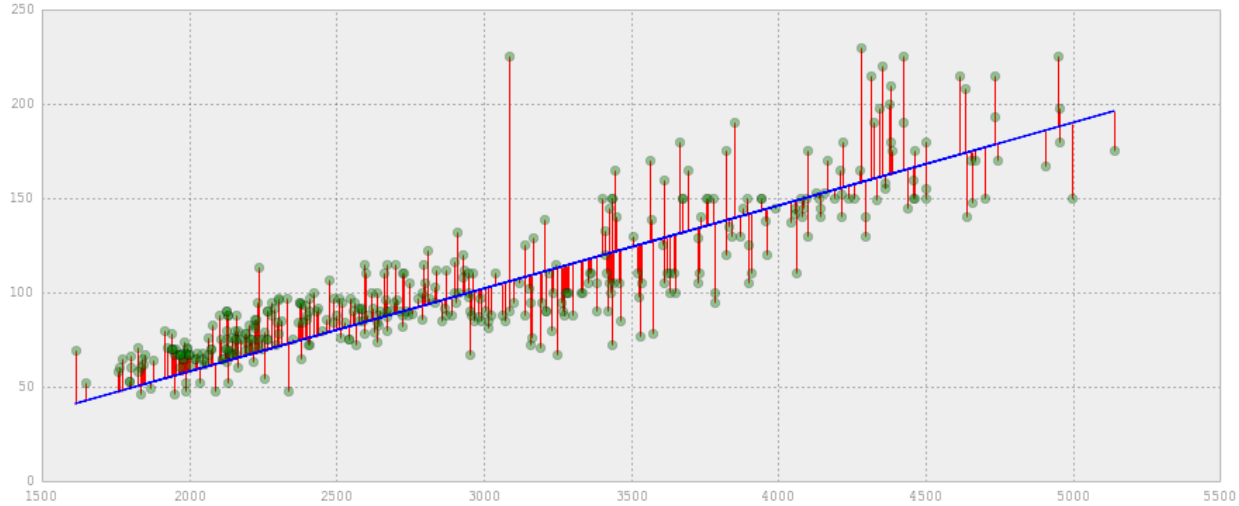
Residuals



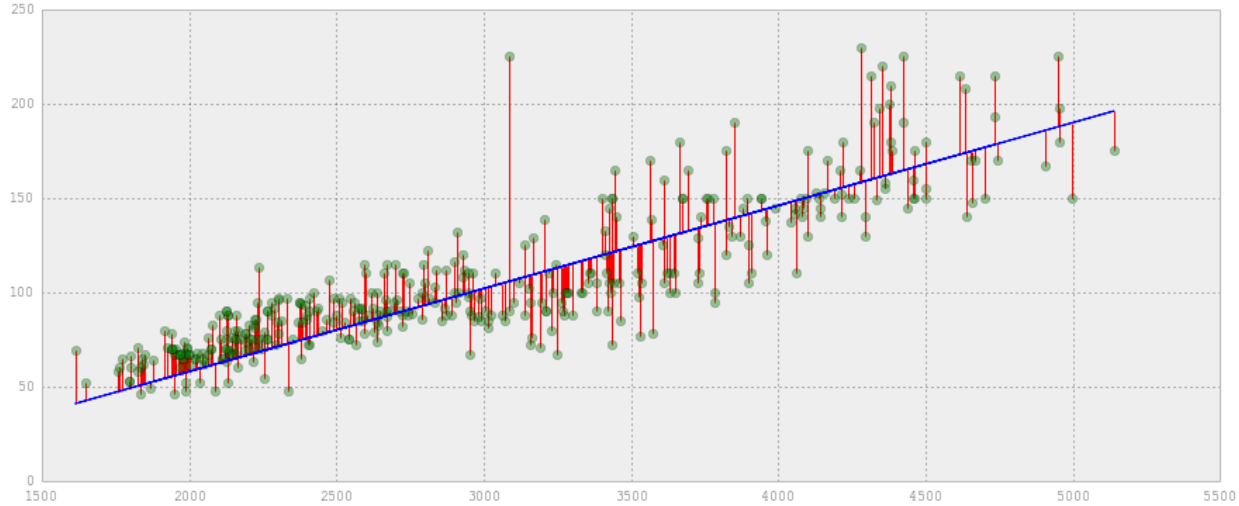
Residuals



Residuals



Residuals



How?

Normal Equation (Analytical)

Gradient Descent (Computational)

Normal Equation

$$(X^T X)\beta = X^T y$$

Normal Equation

$$\beta = (X^T X)^{-1} X^T y$$

Gradient Descent

$$J = (y - X\beta)^2$$

Gradient Descent

Cost

$$J = (y - X\beta)^2$$

Gradient Descent

Loss

$$l = (y_i - X_i\beta)^2$$

Gradient Descent

$$J = \sum_{i}^{n} (y_i - X_i \beta)^2$$

Gradient Descent

$$\beta_i \leftarrow \beta_i - \frac{\partial}{\partial \beta_i} J(\beta)$$

Gradient Descent

Repeat until Convergence….

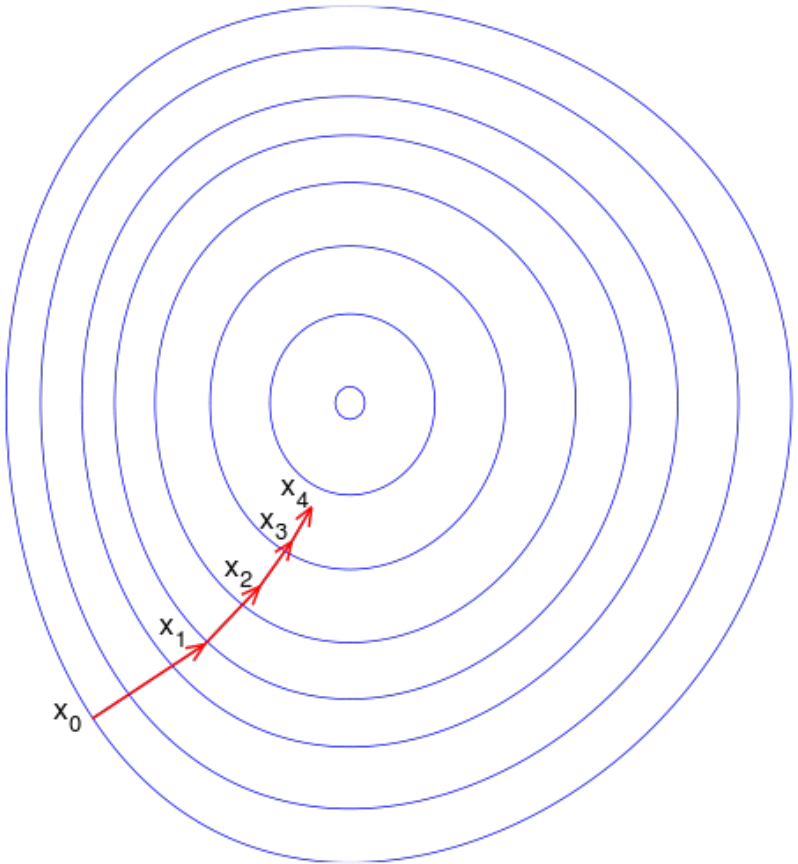
$$\beta_i \leftarrow \beta_i - \frac{\partial}{\partial \beta_i} J(\beta)$$

Gradient Descent

"Learning Rate"

$$\beta_i \leftarrow \beta_i - \alpha \frac{\partial}{\partial \beta_i} J(\beta)$$

Gradient Descent



source: Wikipedia

Optimization

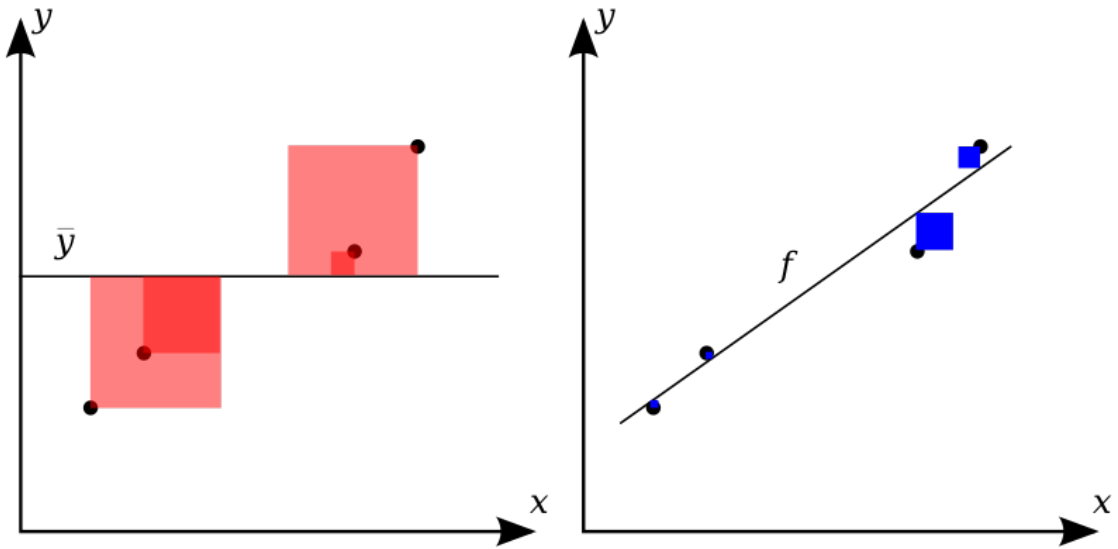
Normal Equation

- Closed Form Analytical Solution
- Can be Inefficient for Large Data Matrices
- Susceptible to Mathematical Instability

Gradient Descent

- Iterative
- Generalized Optimization Technique
- Can be Less Efficient for Smaller Datasets

Evaluating Regression



$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

source: Wikipedia

Debugging Regression

- Multicollinearity
- Heteroskedasticity
- Nonlinearities
- m >> n
- Outliers

Regression in Python

`statsmodels`

- Focus on Statistical Inference
- Generalized Linear Models and Statistical Tests
- Exploration and Analysis

`scikit-learn`

- Focus on Prediction
- Assortment of Machine Learning Models
- Systems and Applications

Types of Learning

Supervised Learning

- Training Data includes desired output

Unsupervised Learning

- Training Data does not include desired output

Semi-supervised Learning

- Training Data includes some desired outputs

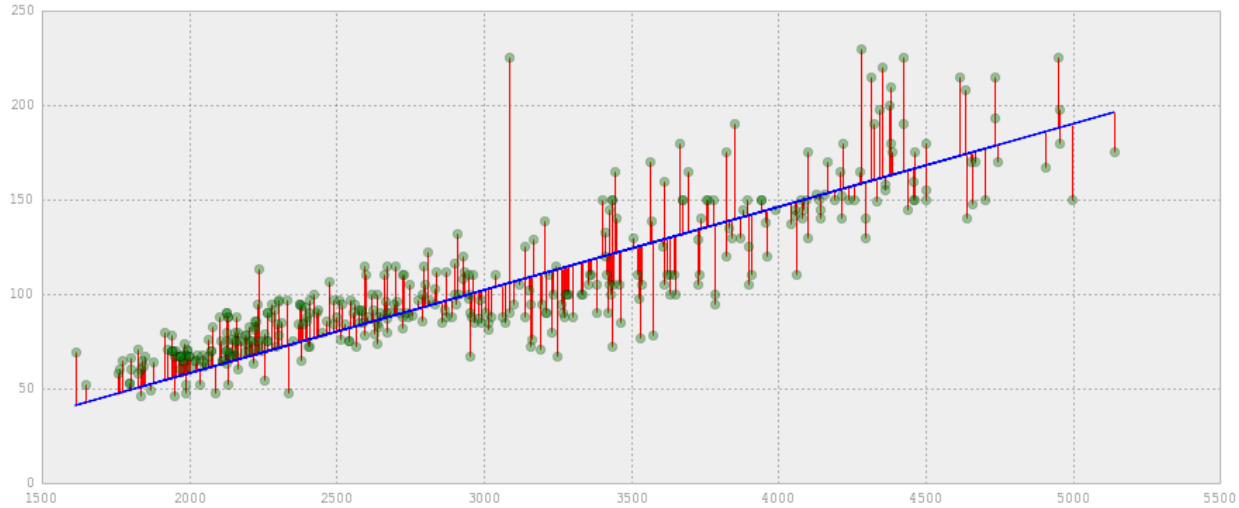
Reinforcement Learning

- Rewards from sequence of actions

Use Cases

- Spam Filtering and document classification
- Finance: Fraud detection and loan default prediction
- Sentiment Analysis: People like to do this with Tweets
- Customer relationship management: Churn Analysis

Linear Regression



Parameters

$$A = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

Logistic Regression

Want:

$$0 < P(label \mid X) < 1$$

Have:

$$A = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

Logistic Regression

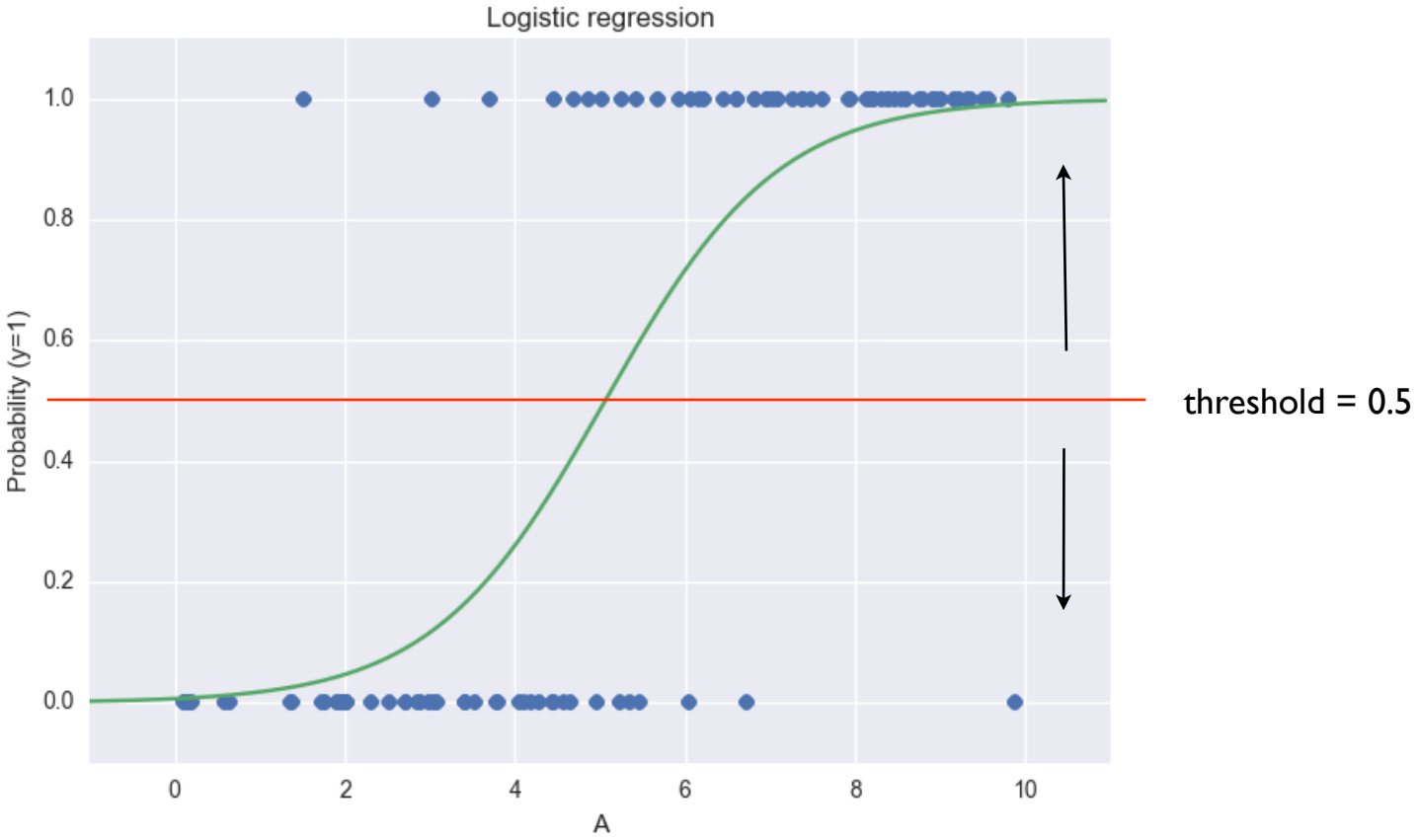
$$A = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

$$P(label \mid X) = \sigma(A)$$

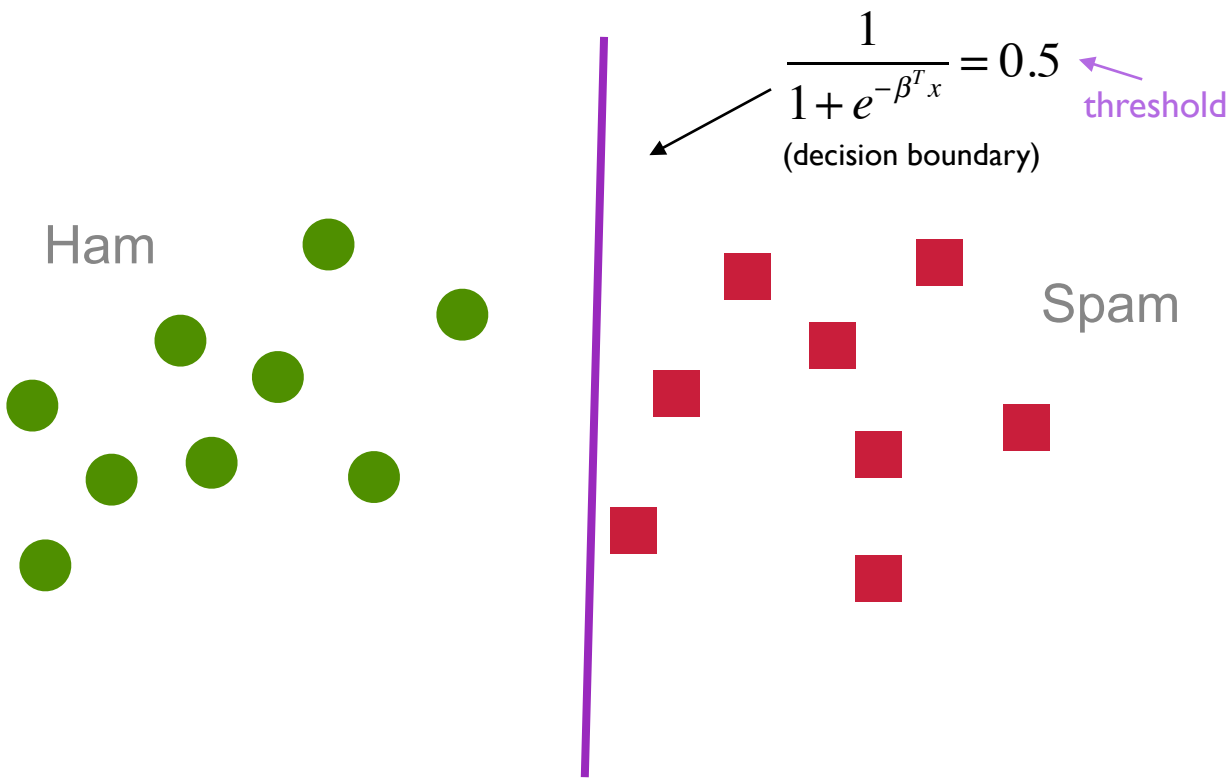
$$\sigma = \frac{1}{1 + e^{-A}}$$ (function bound between 0 and 1)

Logistic Regression

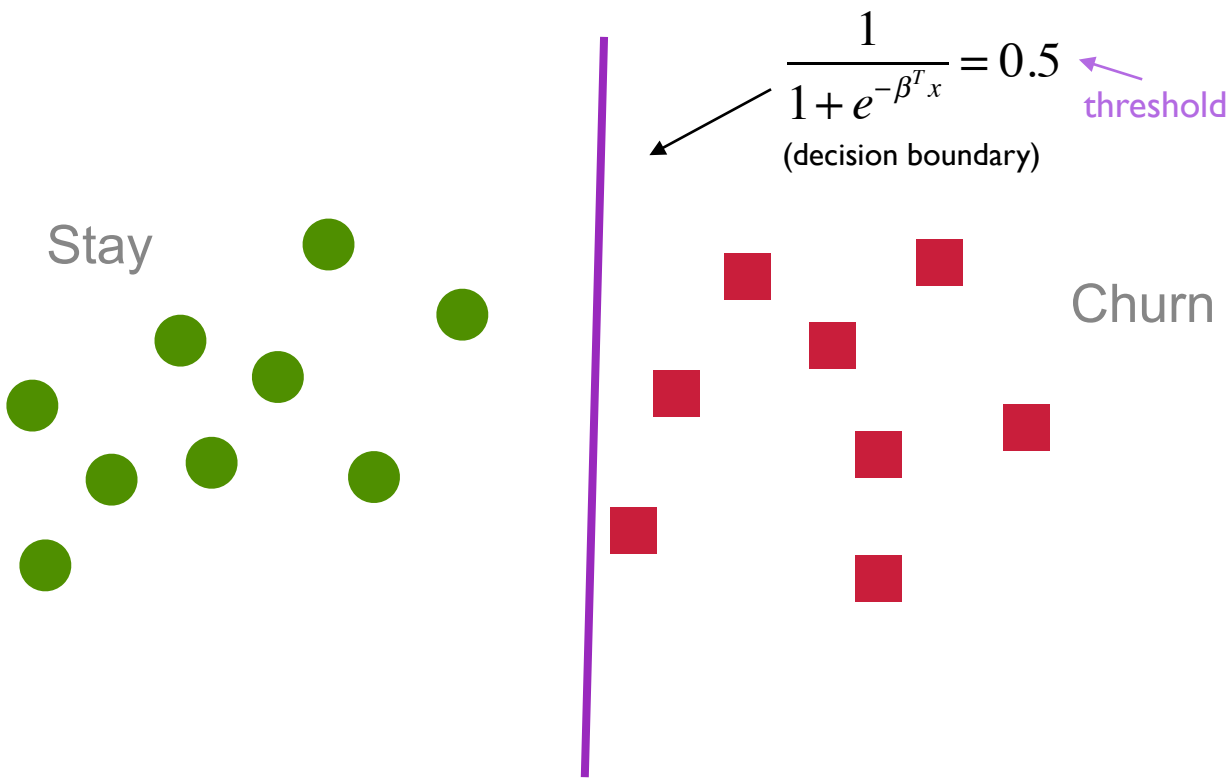
(contrary to its name... actually used to classify)



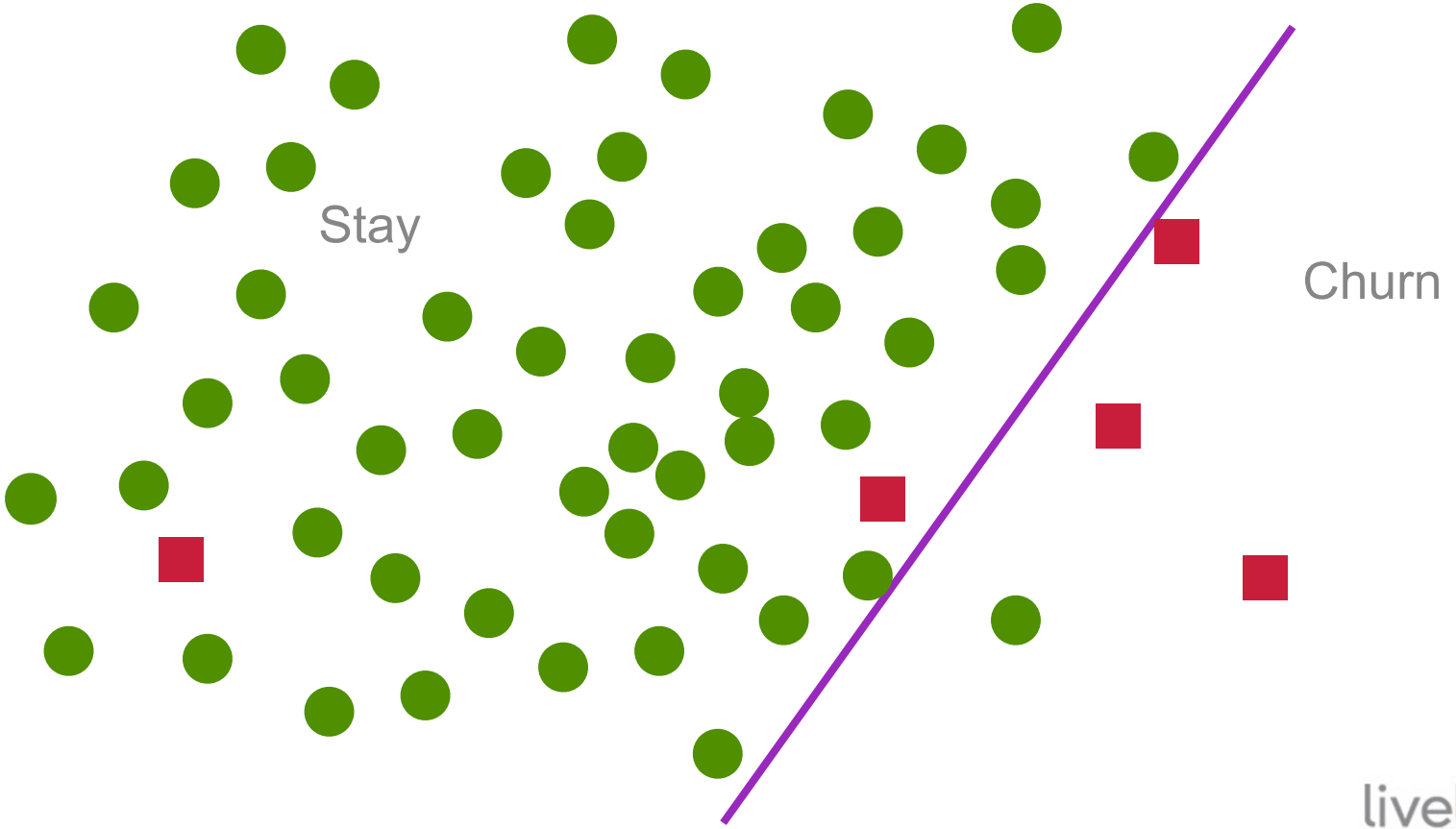
Linear Separator



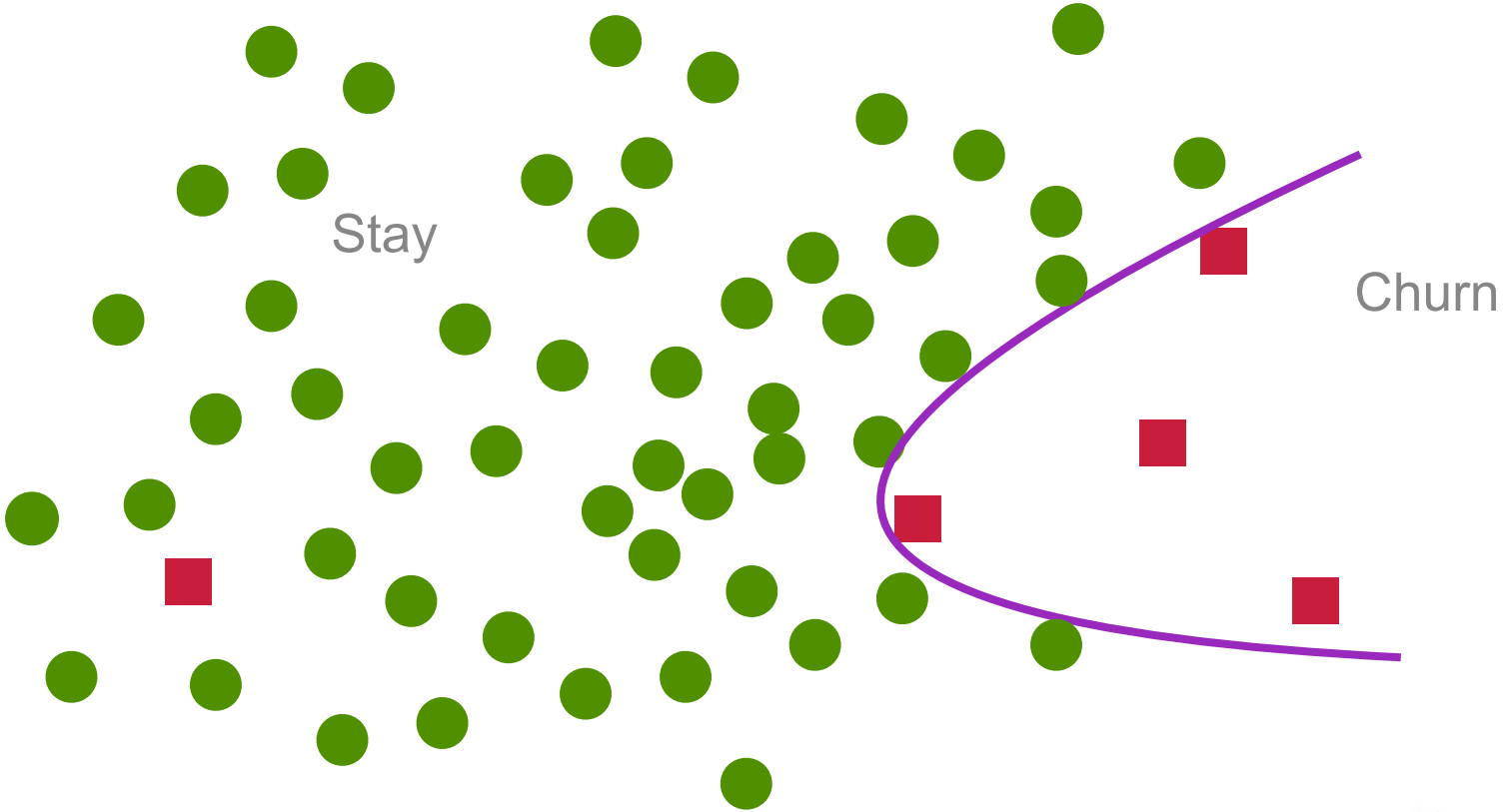
Linear Separator

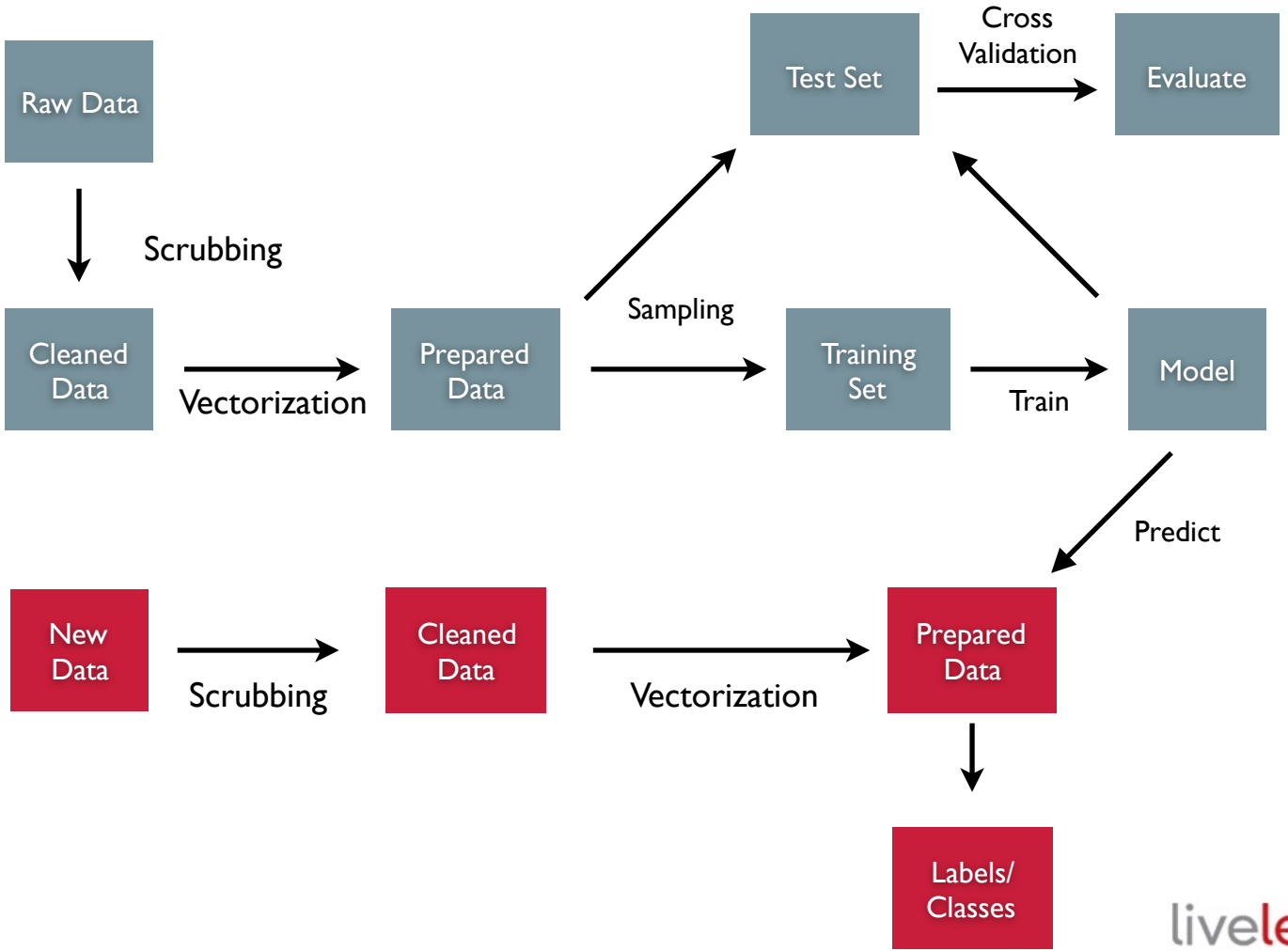


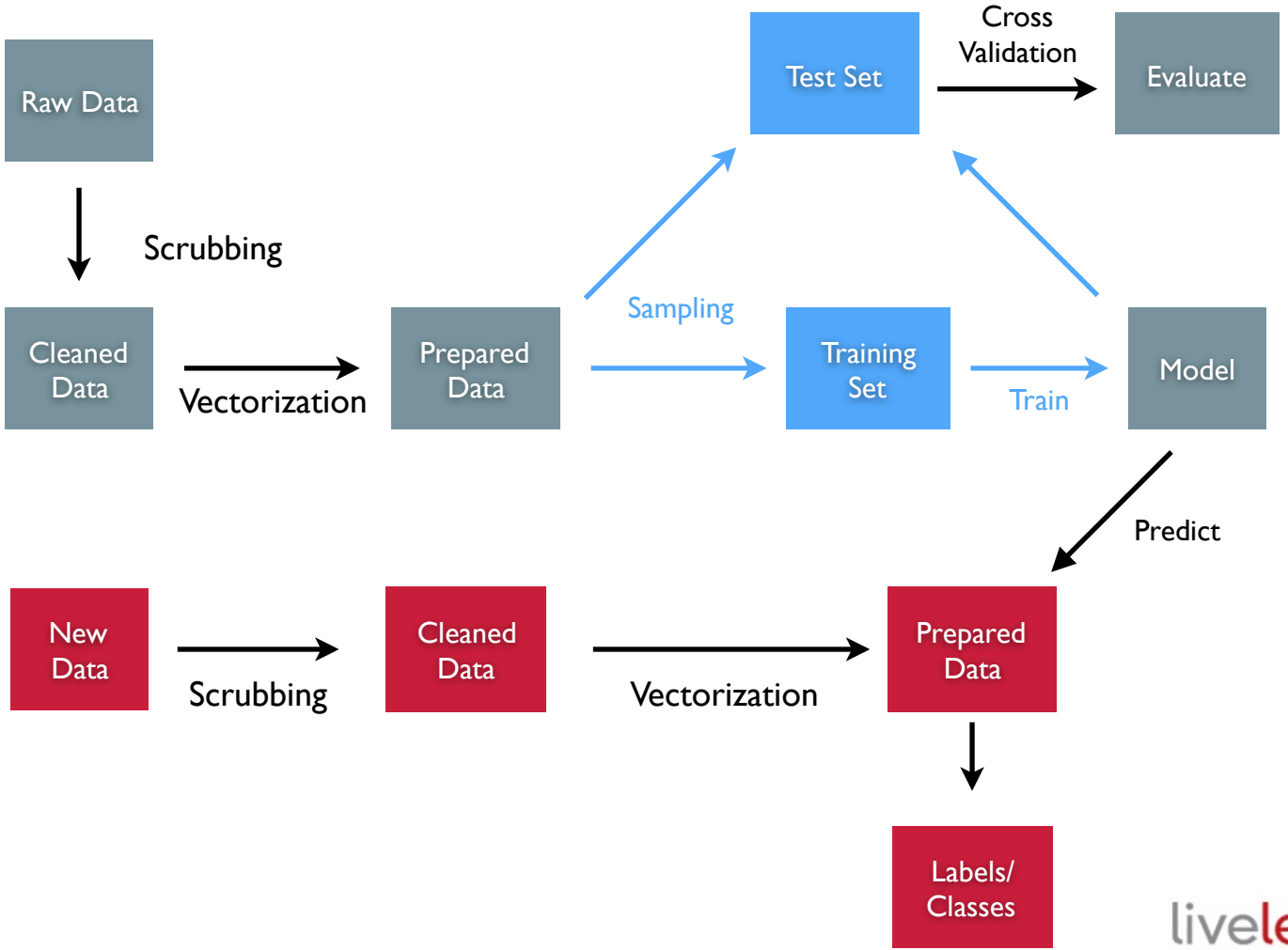
Linear Separator



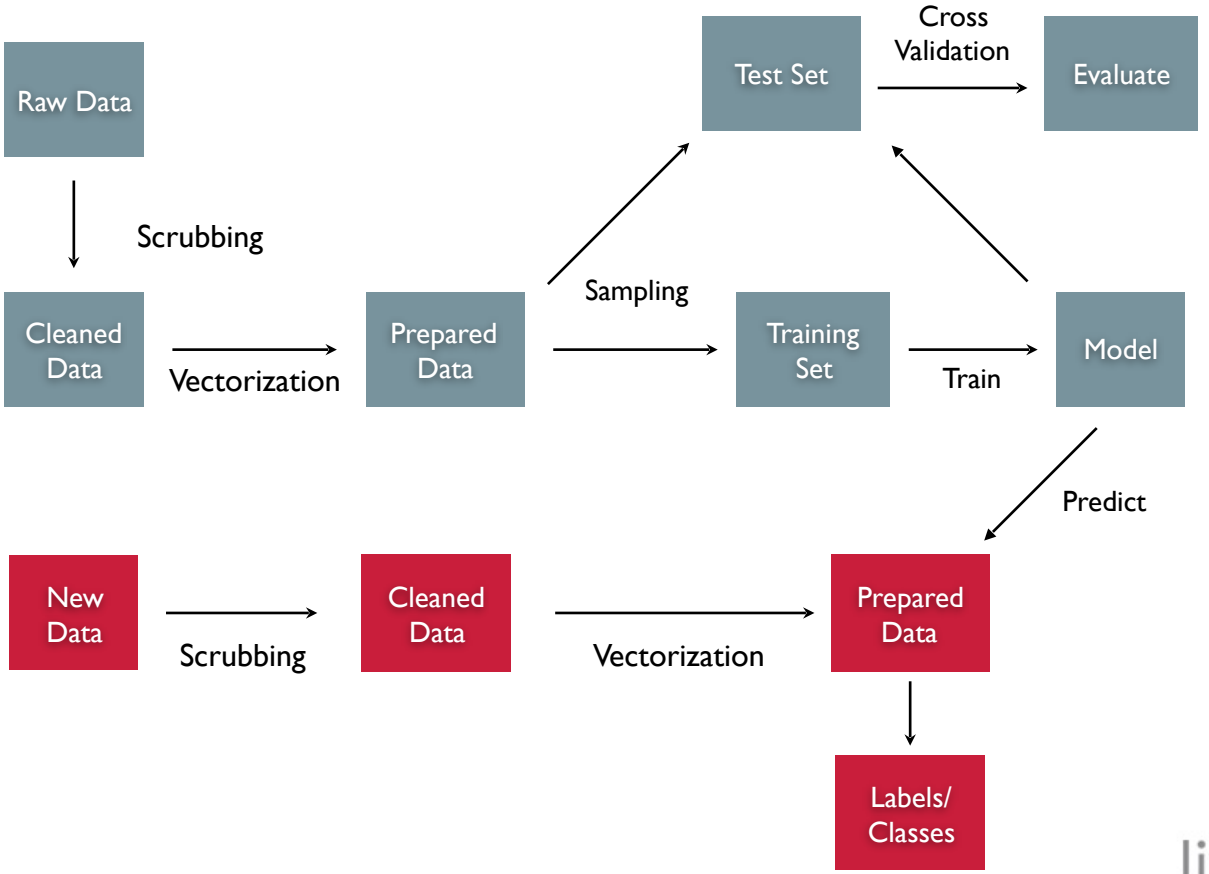
Nonlinear Separator



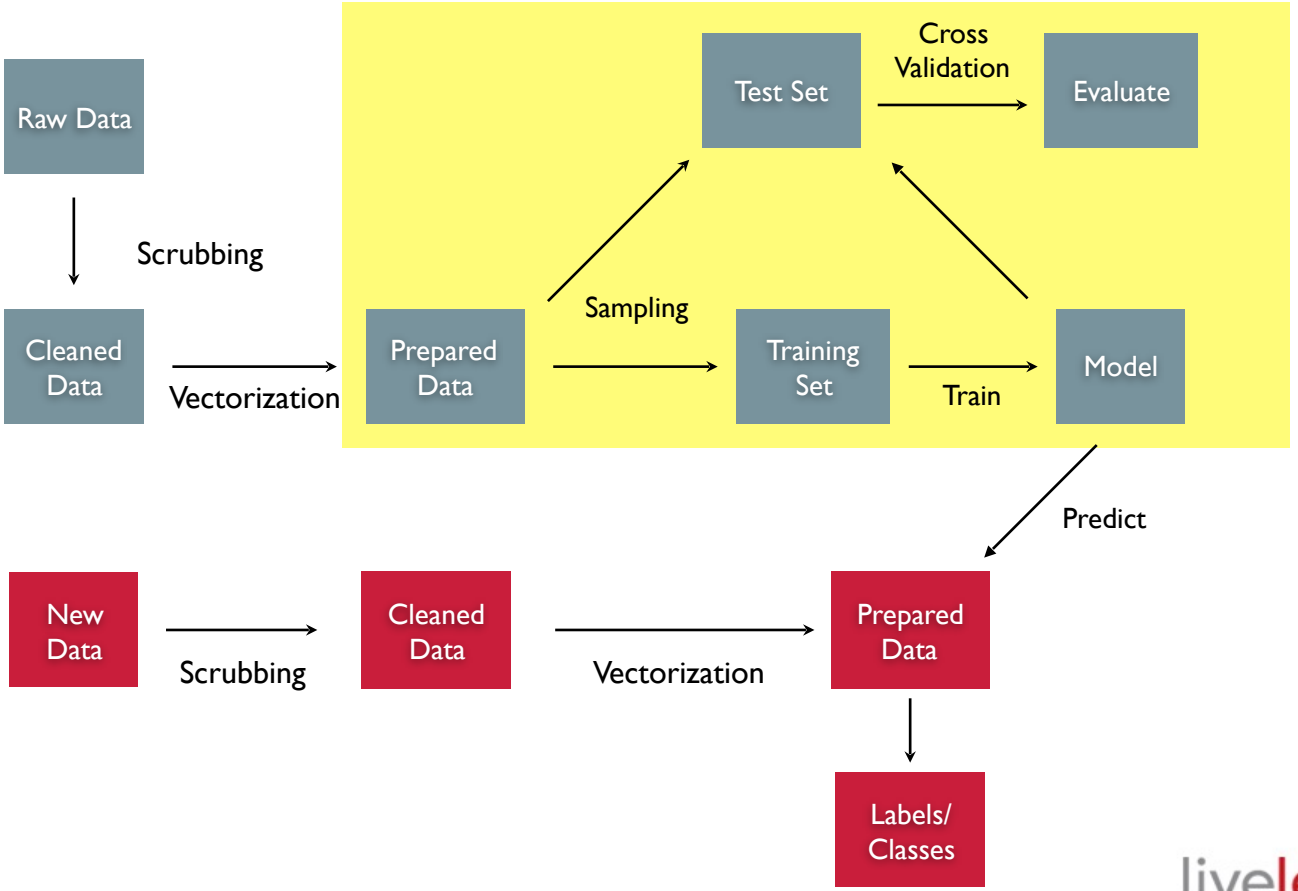




Process + Machine Learning



Process + Machine Learning



Iris Dataset

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	label
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

Features
(feature matrix)

Target

Iris Dataset

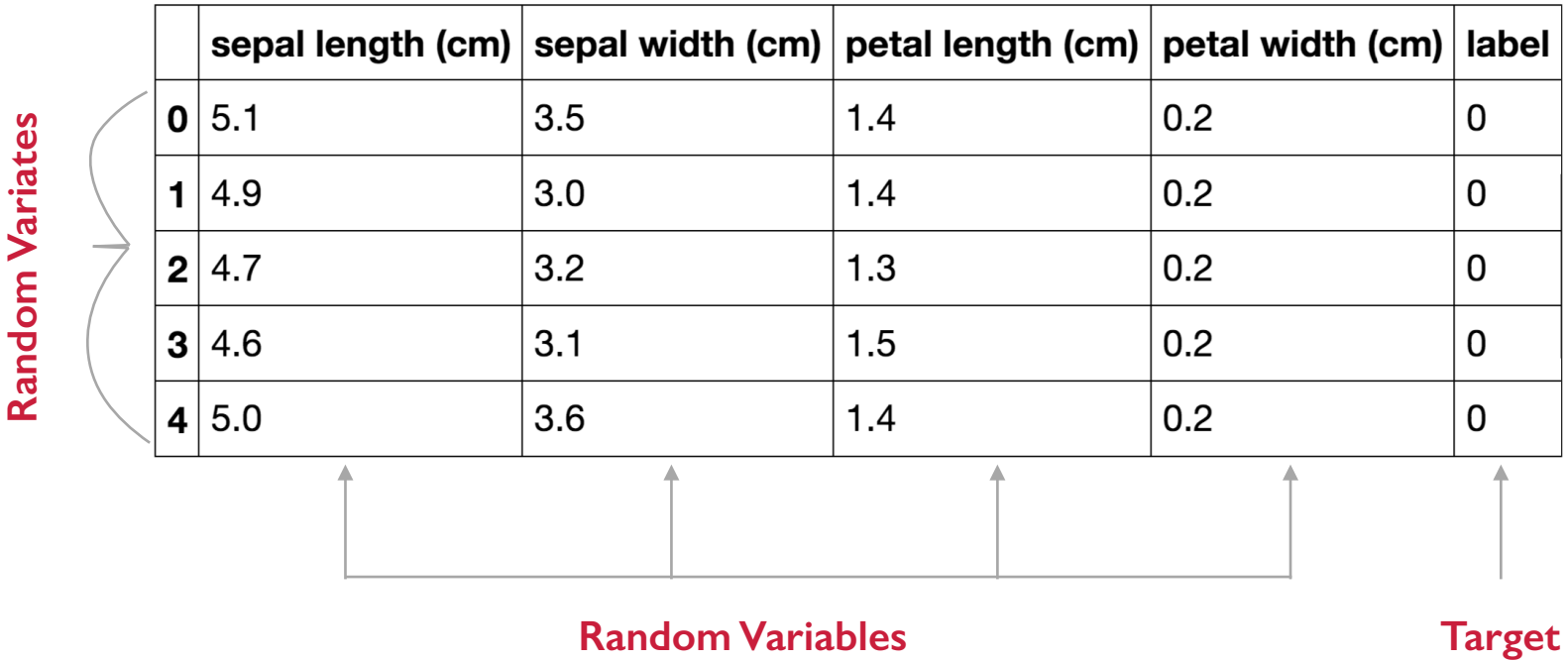
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	label
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

Random Variates

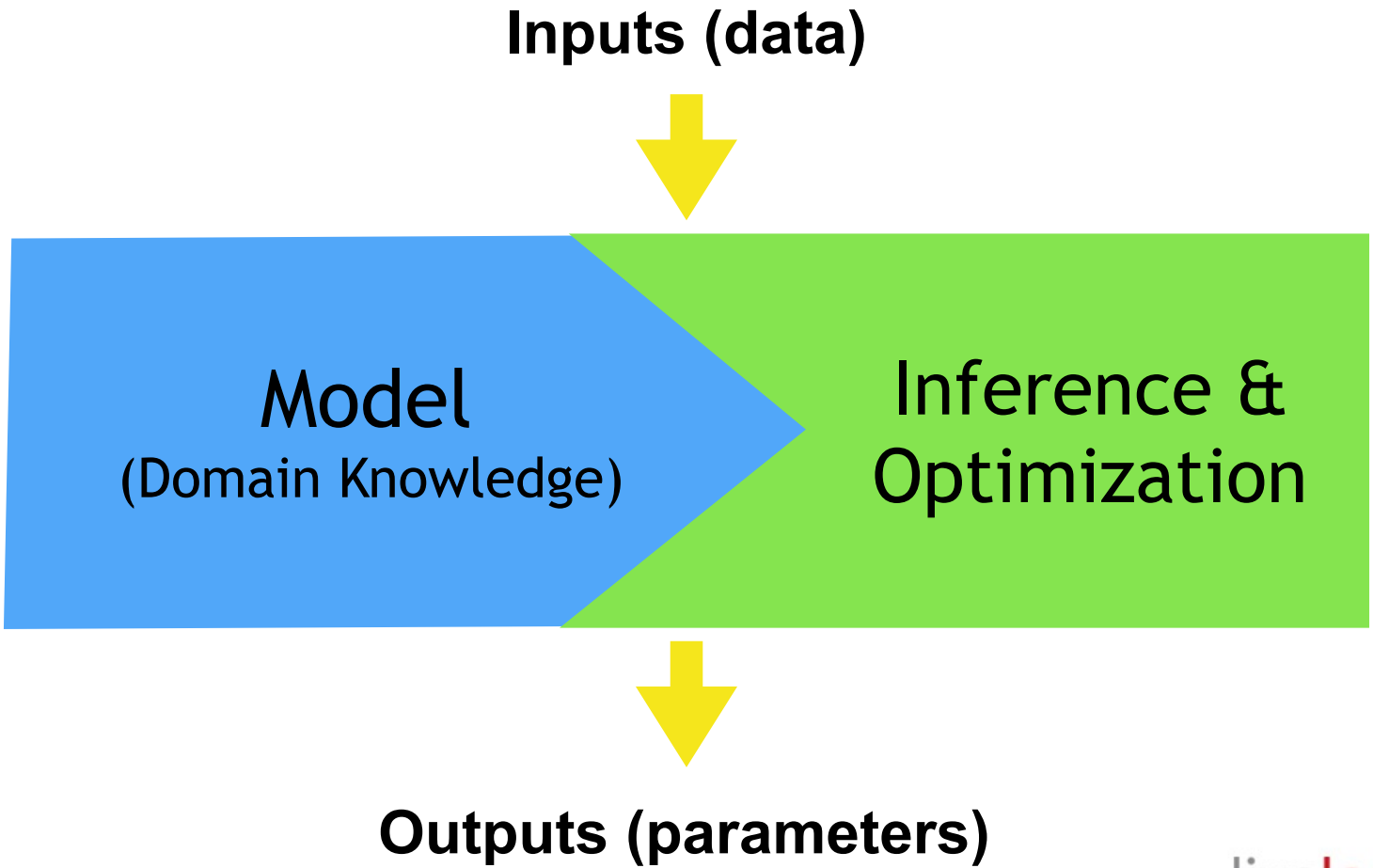
Random Variables

Target

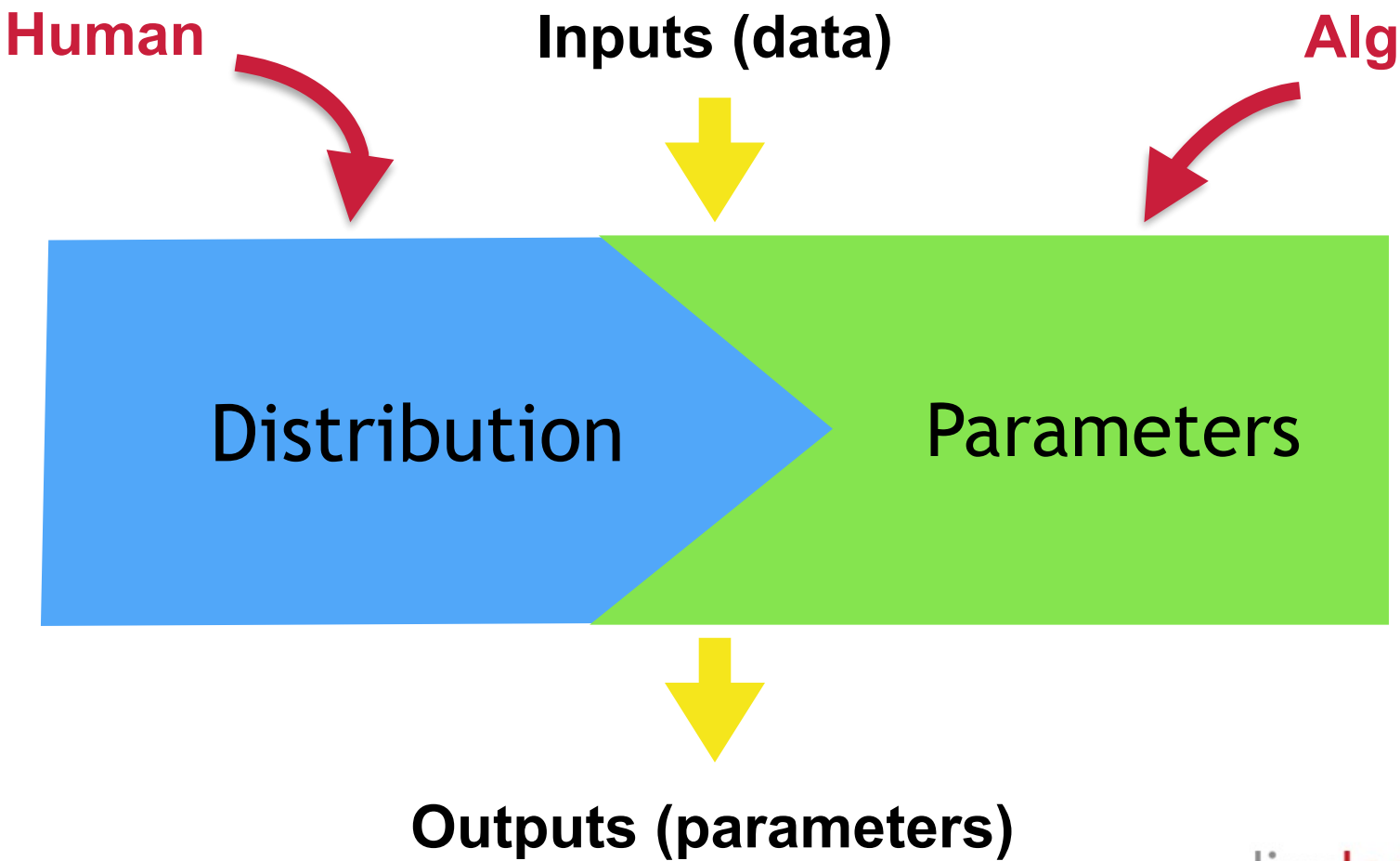
Iris Dataset



	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	label
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0



inspiration: http://www.mbmlbook.com/



inspiration: http://www.mbmlbook.com/

Train

Input: historical labeled data

+

(hypothesis) function with unknown parameter values
<linear, logistic, etc.>

=

Predict

Output: parameter values

What to learn an unknown target function f()

Input: labeled training set (x_i, y_i)

- y_i = f(x_i)

Output: hypothesis h() function “close” to f()

Many possible hypothesis families:

- Logistic
- Linear
- decision trees
- example-based (nearest neighbor)
- etc.

Throughout this lesson we will work to understand the answers the following questions:

- Which hypothesis space to choose?
- How do we measure goodness of fit?
- How do we balance goodness of fit with complexity?
- How do we make h() a good method?
- How do we pick the right kind of h()?
- How do we know if a good h() will predict well?

Throughout this lesson we will work to understand the answers the following questions:

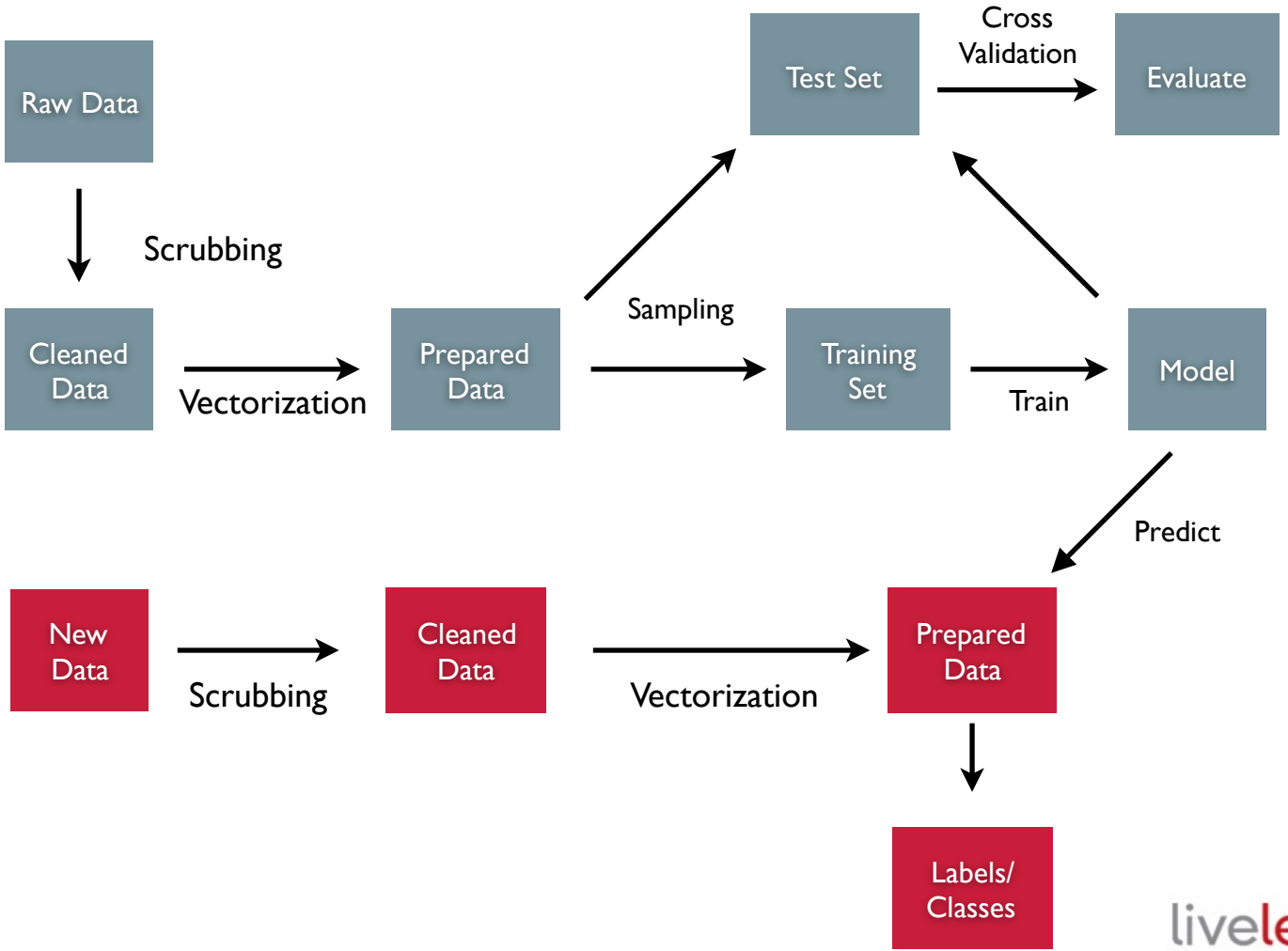
- Which hypothesis space to choose?
  Hypothesis Function
- How do we measure goodness of fit?
  Cost Function
- How do we balance goodness of fit with complexity?
  Regularization
- How do we make h() a good method?
  Optimization
- How do we pick the right kind of h()?
  Cross Validation
- How do we know if a good h() will predict well?
  Hold-out Evaluation

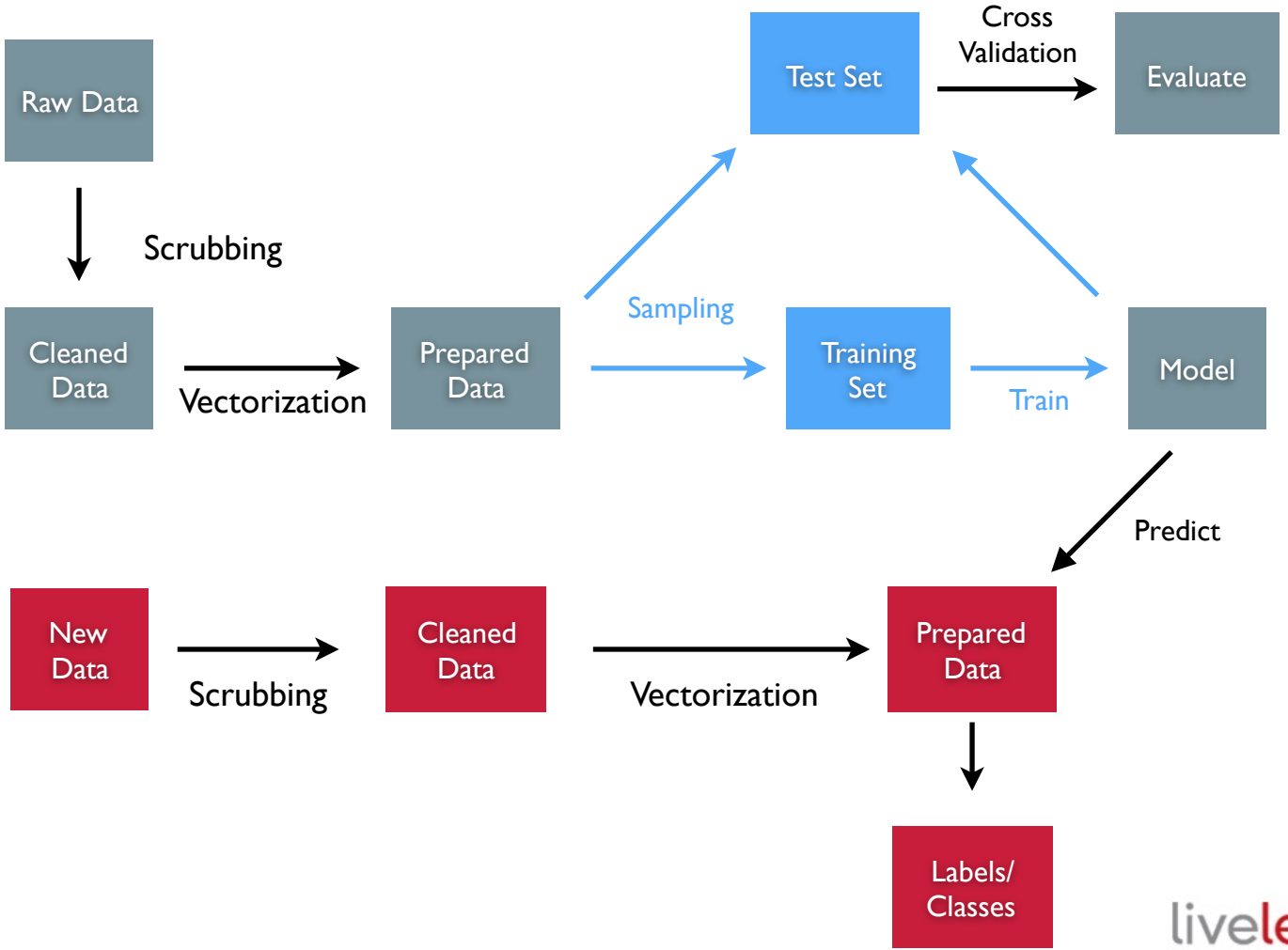
`scikit-learn` Supported Methods

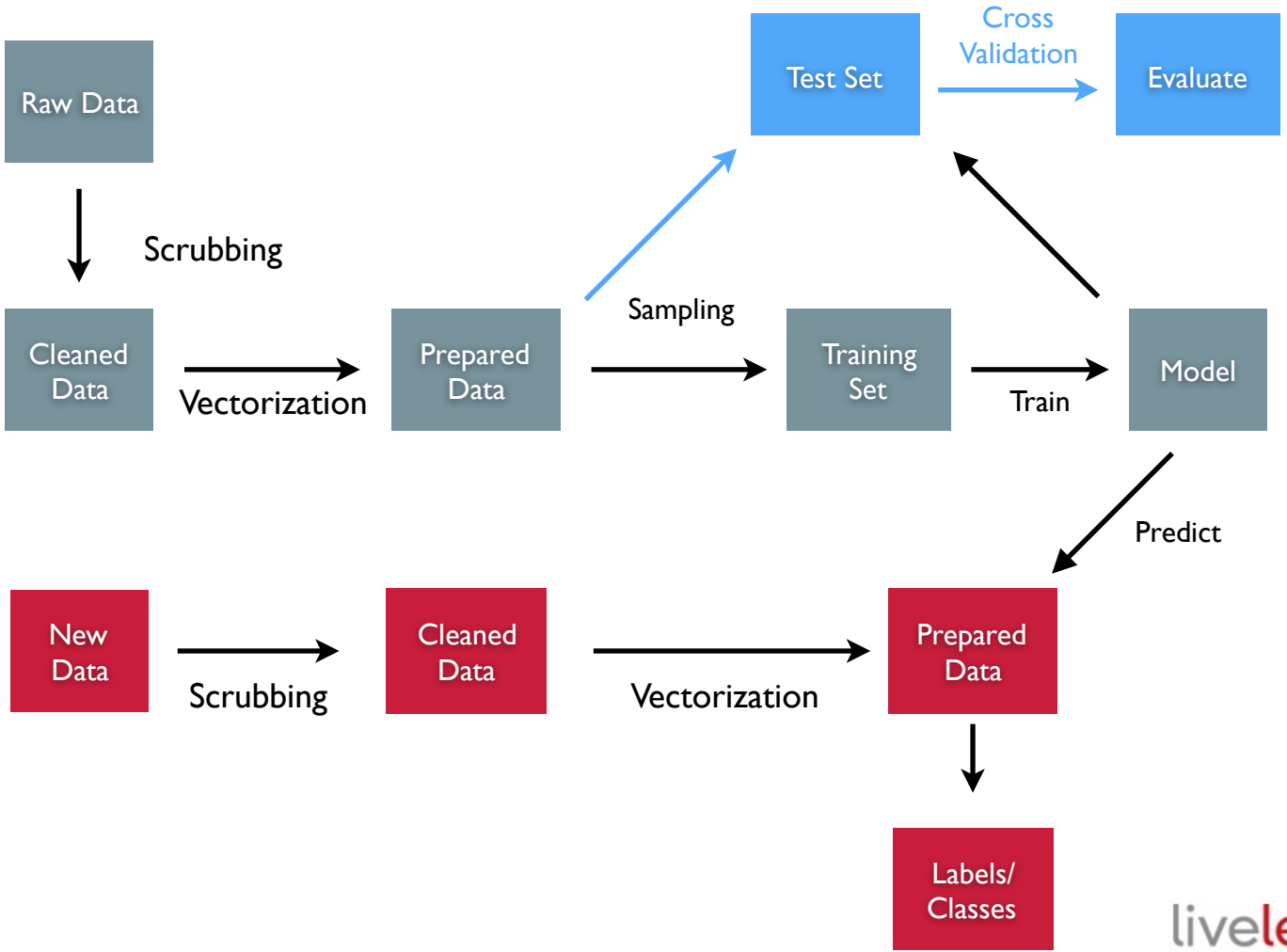
Problem	Method
Binary Classification	*SVMs, logistic regression, decision trees, random forests, gradient-boosted trees, naive Bayes*
Multiclass Classification	*decision trees, random forests, naive Bayes*
Regression	*linear least squares, Lasso, ridge regression, decision trees, random forests, gradient-boosted trees, SVR*

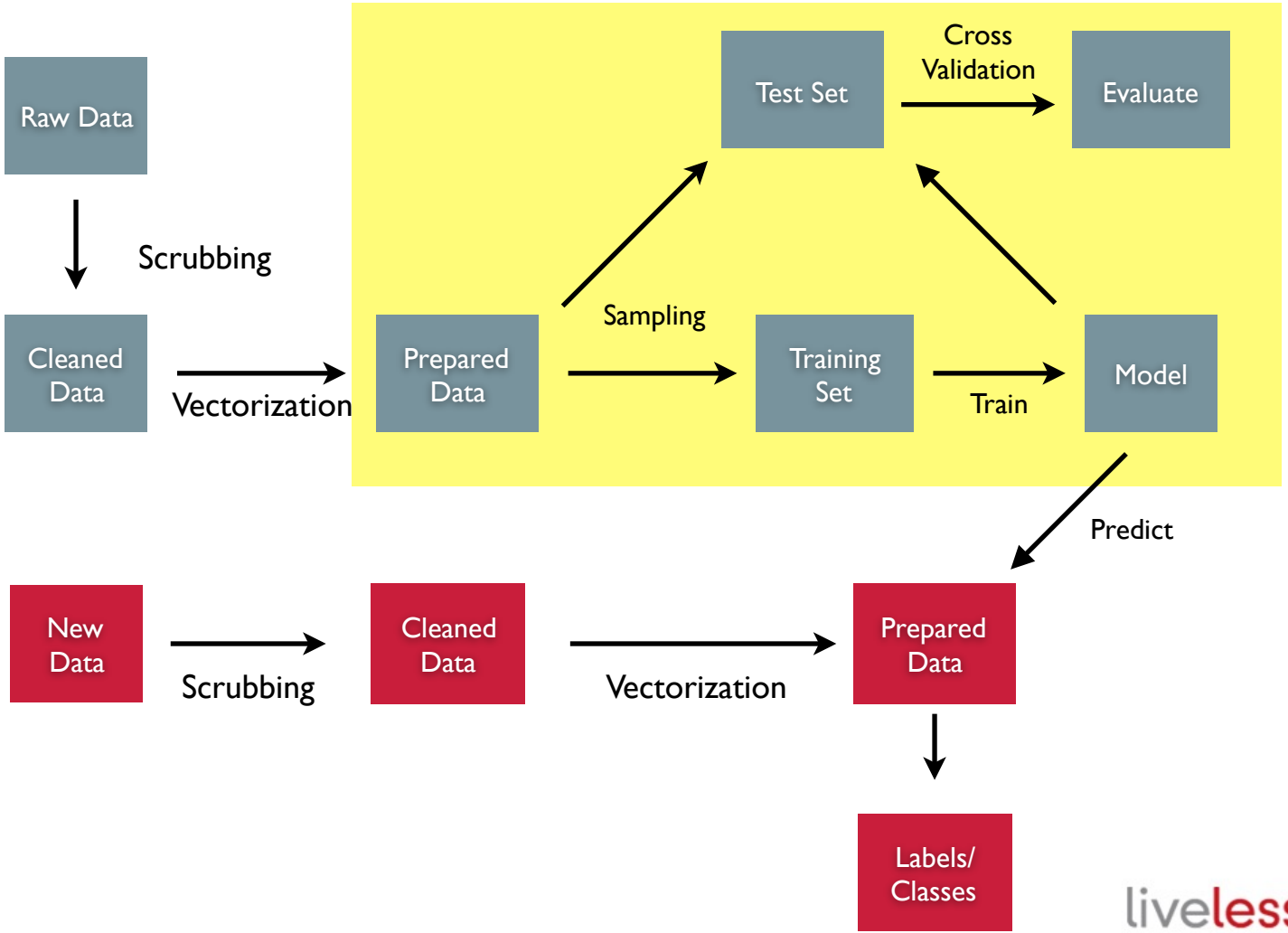
scikit-learn API

```
# create model (estimator) object
model = LogisticRegression(penalty='l1')

# fit model to training data
model.fit(X_train, y_train)

# make predictions
predictions = model.predict(X_test)

# evaluate model
accuracy = (predictions == y_test).sum() / len(y_test)
```







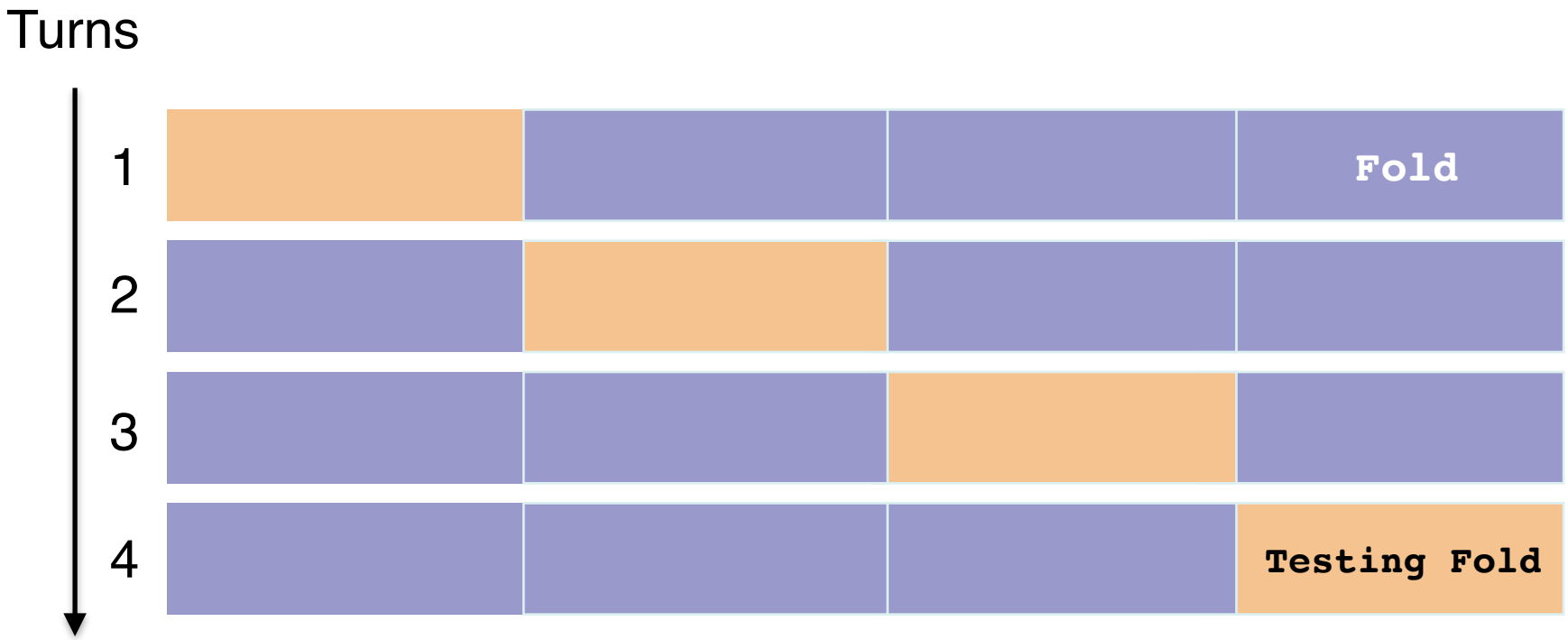


Evaluation

Split labeled data into a training set and a test set

Whatever you do, DO NOT cross the streams

k-fold Cross Validation



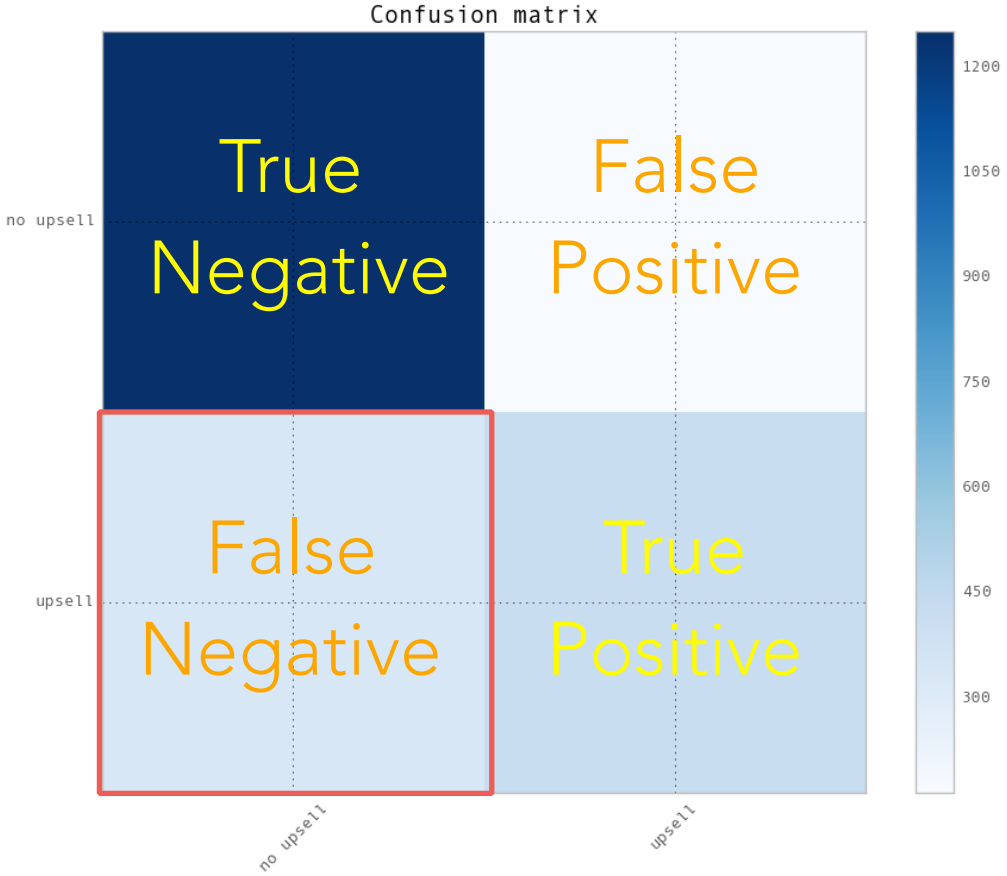
Evaluation Metrics

- Accuracy
- Precision/Recall
- F1
- AUC (area under the curve)

Short Comings of Accuracy

- Sensitive to imbalanced classes
- Misclassifications carry equal weight

$$\text{Accuracy} = \frac{\#\ \text{correct}}{\text{total examples}}$$



True Label

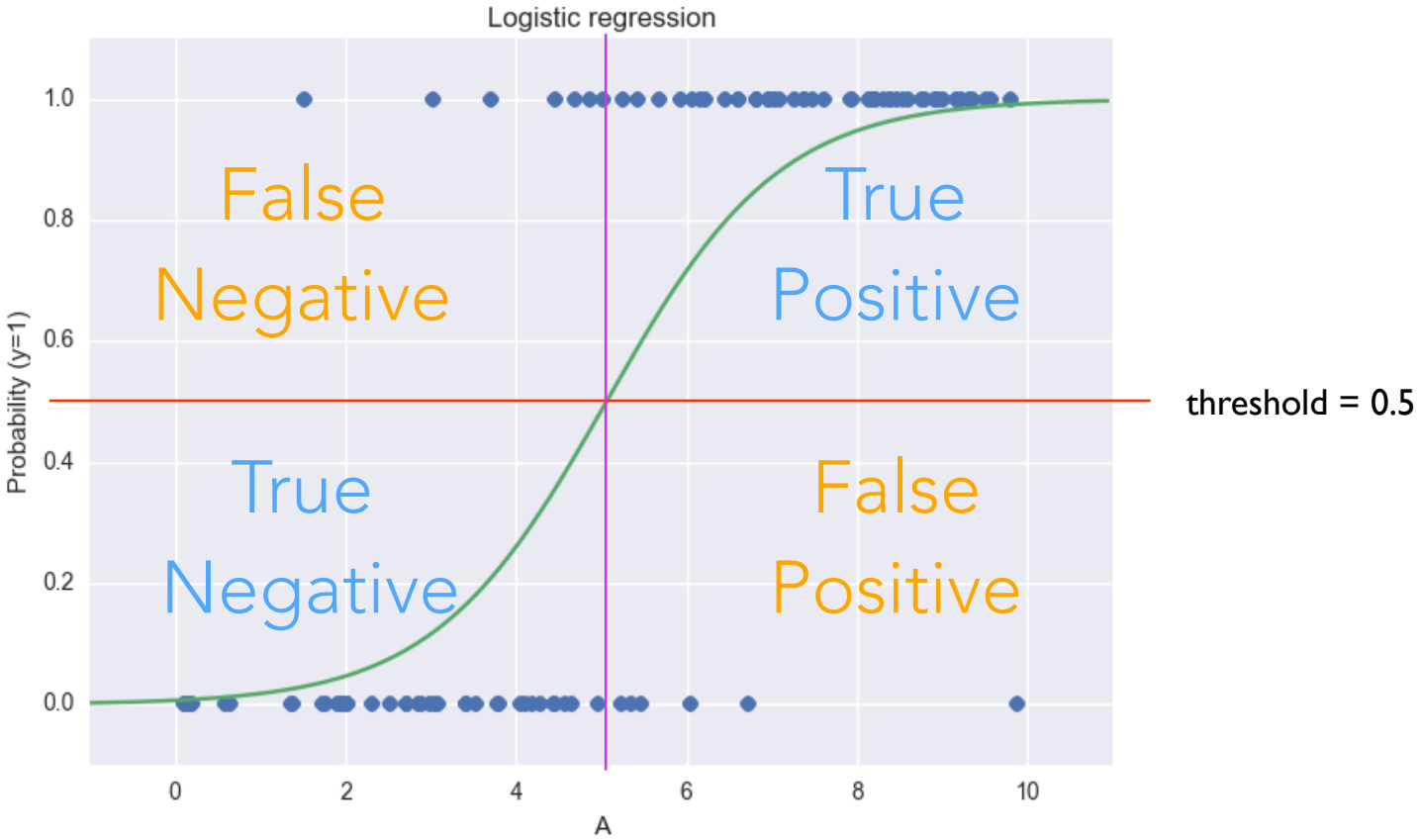
Predicted Label

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

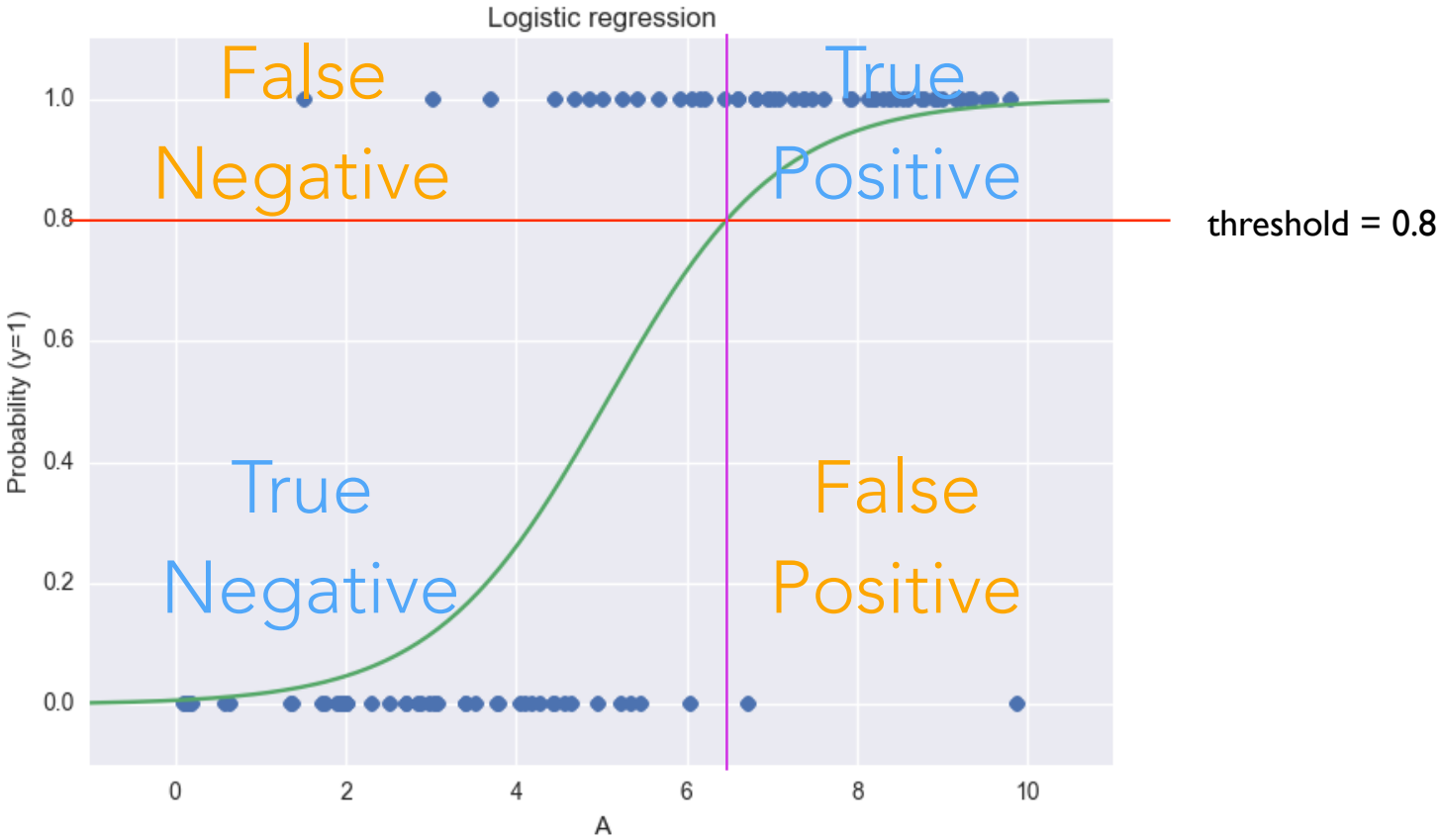
$$\text{True positive rate} = \frac{TP}{TP + FN}$$

$$\text{False positive rate} = \frac{FP}{FP + TN}$$

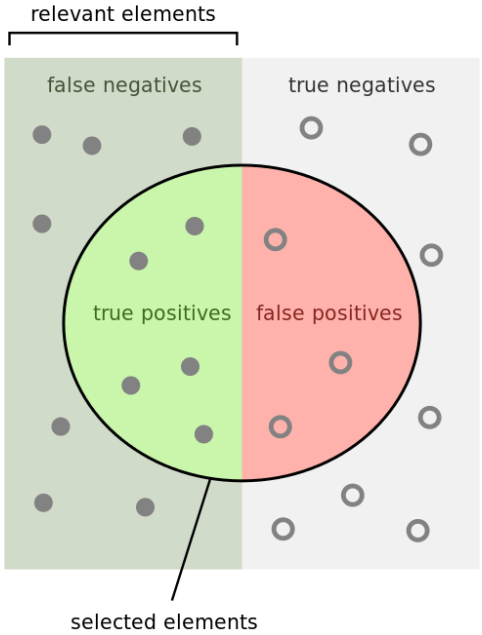
Logistic Regression

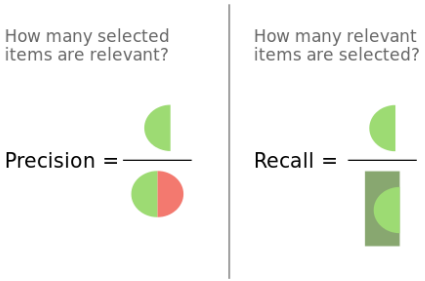


Logistic Regression



Precision and Recall





$$\text{Precision} = \frac{TP}{TP + FP}$$

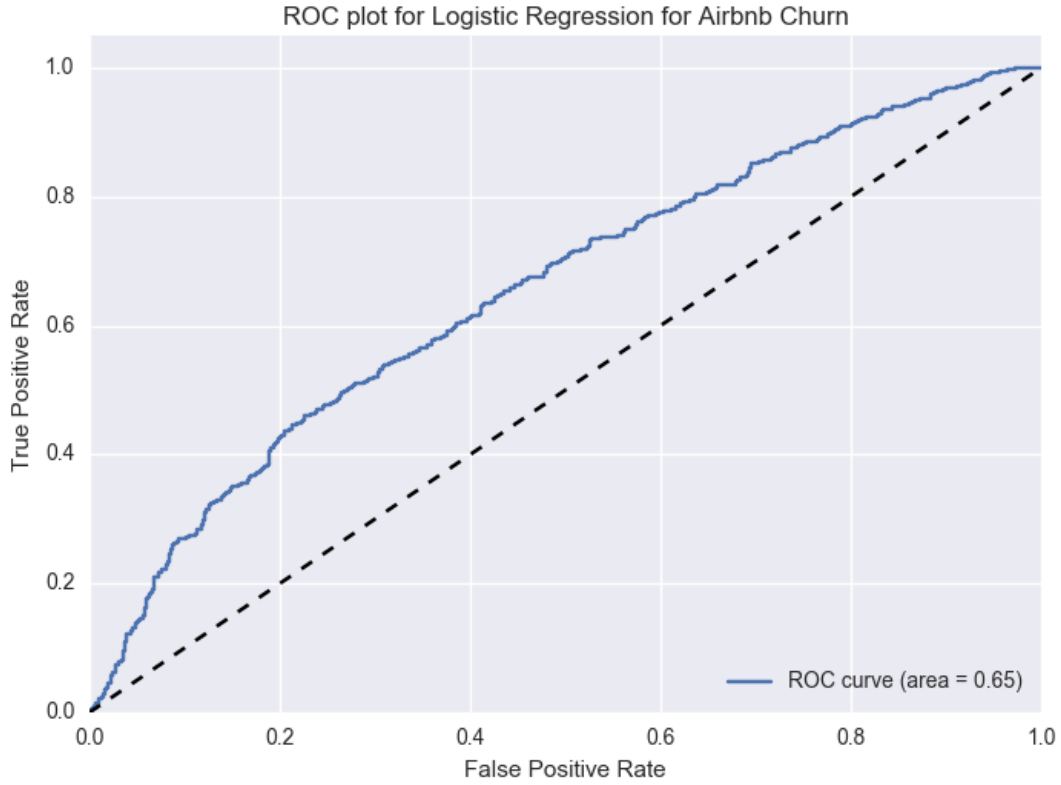
$$\text{Recall} = \frac{TP}{FN + TP}$$

$$F1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}}$$

Confusion Matrix vs. Point Metrics

- Confusion matrix has fine-grained information about misclassifications
- Precision/Recall/F1 can be used in automated comparison (grid search)

ROC plots



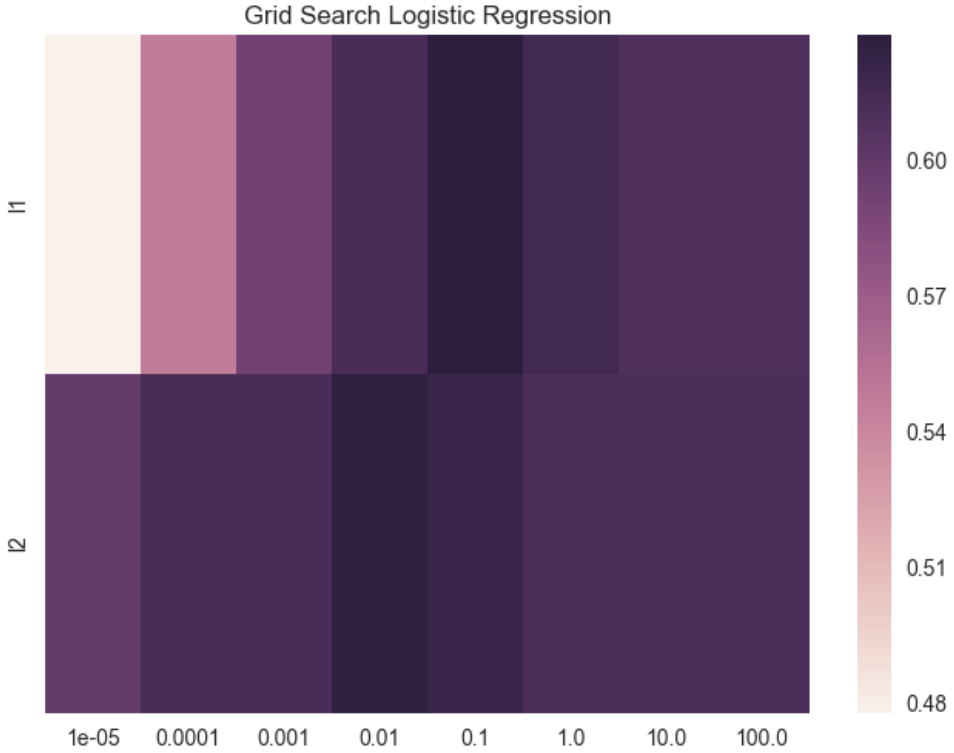
Grid Search

```
X_train, X_test, y_train, y_test = train_test_split(feature_matrix,
labels, test_size=0.2)

scores = np.zeros((2, 8))
penalties = ['l1', 'l2']
regularization = [0.00001, 0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0]

for i, penalty in enumerate(penalties):
  for j, C in enumerate(regularization):
    clf = LogisticRegression(penalty=penalty, C=C)

    scores[i, j] = cross_val_score(clf, X_train, y_train, cv=5).mean()
```

Hyperparameters

```
X_train, X_test, y_train, y_test = train_test_split(feature_matrix,
labels, test_size=0.2)

scores = np.zeros((2, 8))
penalties = ['l1', 'l2']
regularization = [0.00001, 0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0]

for i, penalty in enumerate(penalties):
  for j, C in enumerate(regularization):
    clf = LogisticRegression(penalty=penalty, C=C)

    scores[i, j] = cross_val_score(clf, X_train, y_train, cv=5).mean()
```

Grid Search



Holdout Evaluation

```
# Prefer L1 for sparsity constraints
scores[0].argmax()

# Create model with best performing hyperparameters
clf = LogisticRegression(penalty='l1', C=regularization[3])

# Fit on all of your (training) data
clf.fit(X_train, y_train)

# Evaluate on Holdout set
print(clf.score(X_test, y_test))
# => 0.69231267405
```

Grid Search

- Exhaustive brute force search
- Find optimal hyperparameters or models
- Computationally costly
- But embarrassingly parallel!

Picking a Model

- Performance (optimizing for metric of interest)
- Training time vs testing time
- Online vs. batch
- Interpretability
- Multiclass vs. single class
- High dimensionality
- Nonlinear vs. linear

Review

- When evaluating models, always split into a testing and training dataset
- Accuracy can be a very misleading measure, especially when errors do not have equal weight
- To tune a model, we can grid search over possible parameter values

What We Didn't Cover

- k-nearest neighbors
- Naive Bayes
- Neural Networks
- Support Vector Machines (SVM)
- Decision Trees
- Ensembles (Random Forest and Gradient Boosted Trees)
- Feature Engineering and Selection