# Machine Learning basics

Son Bao Pham

# Content

- Motivation
- Practical Process in building a predictor: basic components
- Linear Regression and Decision Tree
- WEKA (Waikato Environment for Knowledge Analysis)

# Who predicts what

- Google predicts whether you will click on an ad -> increase revenue
- Amazon predicts what movies you will watch -> increase revenue
- Bank predicts the likelihood of loan default -> reduce risks/loss
- Lots of start-up!

# Netflix 1 million prize

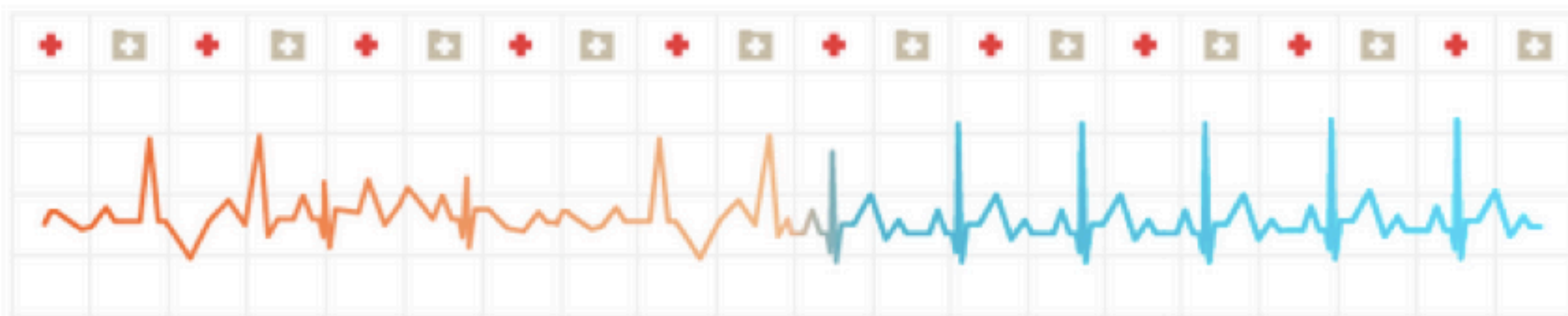
## Netflix Awards $1 Million Prize and Starts a New Contest

By STEVE LOHR SEPTEMBER 21, 2009 10:15 AM


Jason Kempin/Getty Images Netflix prize winners, from left: Yehuda Koren, Martin Chabbert, Martin Piotte, Michael Jahrer, Andreas Toscher, Chris Volinsky and Robert Bell.

# Heritage Health Prize 3 Million

Information   Data   Forum   Leaderboard



**Improve Healthcare,**
**Win $3,000,000.**

COMPETITION GOAL

**Identify patients who will be admitted to a hospital within the next year, using historical claims data.**

# Sport: Kaggle

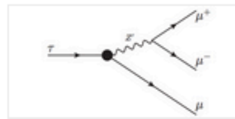
## Featured Competitions View All »

MACHINE LEARNING CHALLENGES FOR EDUCATION, RESEARCH, AND INDUSTRY.

**Coupon Purchase Prediction**
**$50,000**
Predict which coupons a customer will buy

**Caterpillar Tube Pricing**
**$30,000**
Model quoted prices for industrial tube assemblies

**Liberty Mutual Group: Property**
**$25,000**
Quantify property hazards before time of inspection

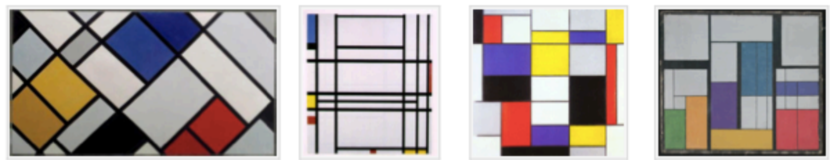


**Flavours of Physics: Finding τ**
**$15,000**
Identify a rare decay phenomenon

**ICDM 2015: Drawbridge Cross-**
**$10,000**
Identify individual users across their digital devices

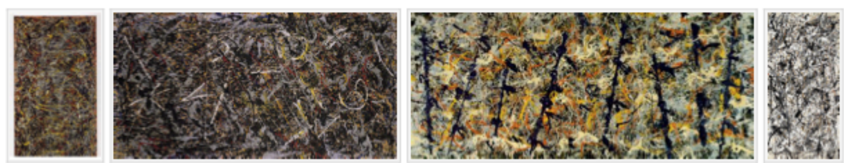
# IOI for high school students

Style 1 contains neoplastic modern art. For example:



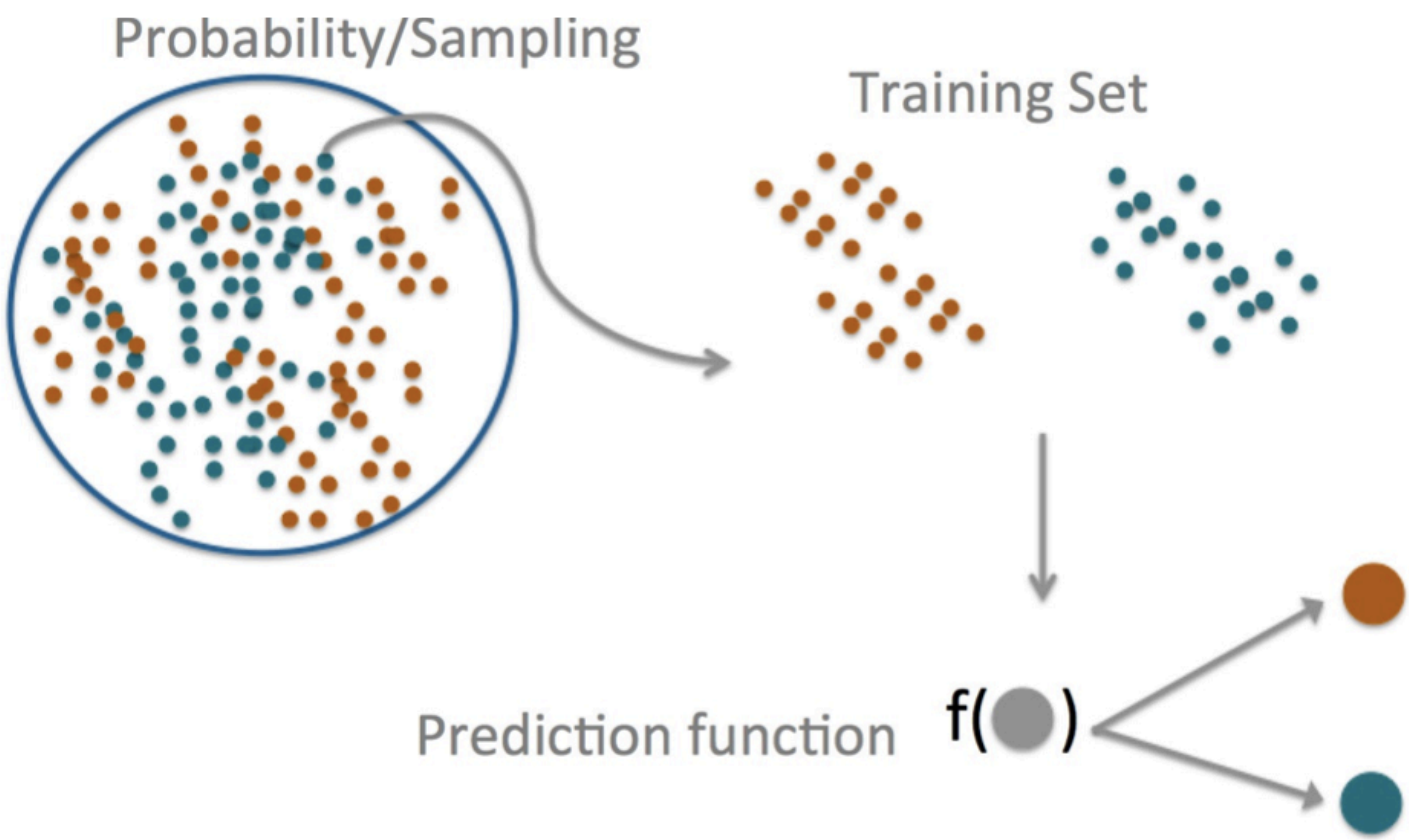
Style 2 contains impressionist landscapes. For example:

Style 3 contains expressionist action paintings. For example:



Style 4 contains colour field paintings. For example:

# Prediction



# What can go wrong

BIG DATA

## The Parable of Google Flu: Traps in Big Data Analysis

Large errors in flu prediction were largely avoidable, which offers lessons for the use of big data.

David Lazer,[1,2]* Ryan Kennedy,[1,3,4] Gary King,[3] Alessandro Vespignani[3,5,6]

In February 2013, Google Flu Trends (GFT) made headlines but not for a reason that Google executives or the creators of the flu tracking system would have hoped. *Nature* reported that GFT was predicting more than double the proportion of doctor visits for influenza-like illness (ILI) than the Centers for Disease Control and Prevention (CDC), which bases its estimates on surveillance reports from laboratories across the United States (*1*, *2*). This happened despite the fact that GFT was built to predict CDC reports. Given that GFT is often held up as an exemplary use of big data (*3*, *4*), what lessons can we draw from this error?

The problems we identify are not limited to GFT. Research on whether search or social media can predict *x* has become commonplace (*5*–*7*) and is often put in sharp contrast with traditional methods and hypotheses. Although these studies have shown the value of these data, we are far from a place where they can supplant more traditional methods or theories (*8*). We explore two

surement and construct validity and reliability and dependencies among data (*12*). The core challenge is that most big data that have received popular attention are not the output of instruments designed to produce valid and reliable data amenable for scien-

the algorithm in 2009, and this model has run ever since, with a few changes announced in October 2013 (*10*, *15*).

Although not widely reported until 2013, the new GFT has been persistently overestimating flu prevalence for a much longer time. GFT also missed by a very large margin in the 2011–2012 flu season and has missed high for 100 out of 108 weeks starting with August 2011 (see the graph). These errors are not randomly distributed. For example, last week's errors predict this week's errors (temporal autocorrelation), and the direction and magnitude of error varies with the time of year (seasonality). These patterns mean that GFT overlooks considerable information that could be extracted by traditional statistical methods.

Even after GFT was updated in 2009, the comparative value of the algorithm as a stand-alone flu monitor is questionable. A study in 2010 demonstrated that GFT accuracy was not much better than a fairly simple projection forward using already avail-

# Predictor’s Components

- Question
- Input Data
- Features
- Algorithm
- Parameters
- Evaluation

# SPAM example

Question -> input data -> features -> algorithm -> parameter -> evaluation

**Start with a general question**

Can I automatically detect emails that are SPAM or not?

**Make it concrete**

Can I use quantitative characteristics of the emails to classify them as SPAM/HAM?

# SPAM example

Question -> input data -> features -> algorithm -> parameter -> evaluation

About Citation Policy Donate a Data Set Contact

UCI

Search

Repository Web Google

Machine Learning Repository

Center for Machine Learning and Intelligent Systems

View ALL Data Sets

## Spambase Data Set

*Download*: Data Folder, Data Set Description

**Abstract**: Classifying Email as Spam or Non-Spam

| | | | | | |
|---|---|---|---|---|---|
| **Data Set Characteristics:** | Multivariate | **Number of Instances:** | 4601 | **Area:** | Computer |
| **Attribute Characteristics:** | Integer, Real | **Number of Attributes:** | 57 | **Date Donated** | 1999-07-01 |
| **Associated Tasks:** | Classification | **Missing Values?** | Yes | **Number of Web Hits:** | 141823 |

**Source:**

Creators:

Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt
Hewlett-Packard Labs, 1501 Page Mill Rd., Palo Alto, CA 94304

Donor:

George Forman (gforman at nospam hpl.hp.com) 650-857-7835

**Data Set Information:**

The "spam" concept is diverse: advertisements for products/web sites, make money fast schemes, chain letters, pornography...

# SPAM example

Question -> input data -> features -> algorithm -> parameter -> evaluation

Dear Jeff,

Can you send me your address so I can send you the invitation?

Thanks,

Ben

# SPAM example

Question -> input data -> features -> algorithm -> parameter -> evaluation

Dear Jeff,

Can you send me your address so I can send you the invitation?

Thanks,

Ben

Frequency of you = 2/17 = 0.118

# SPAM example

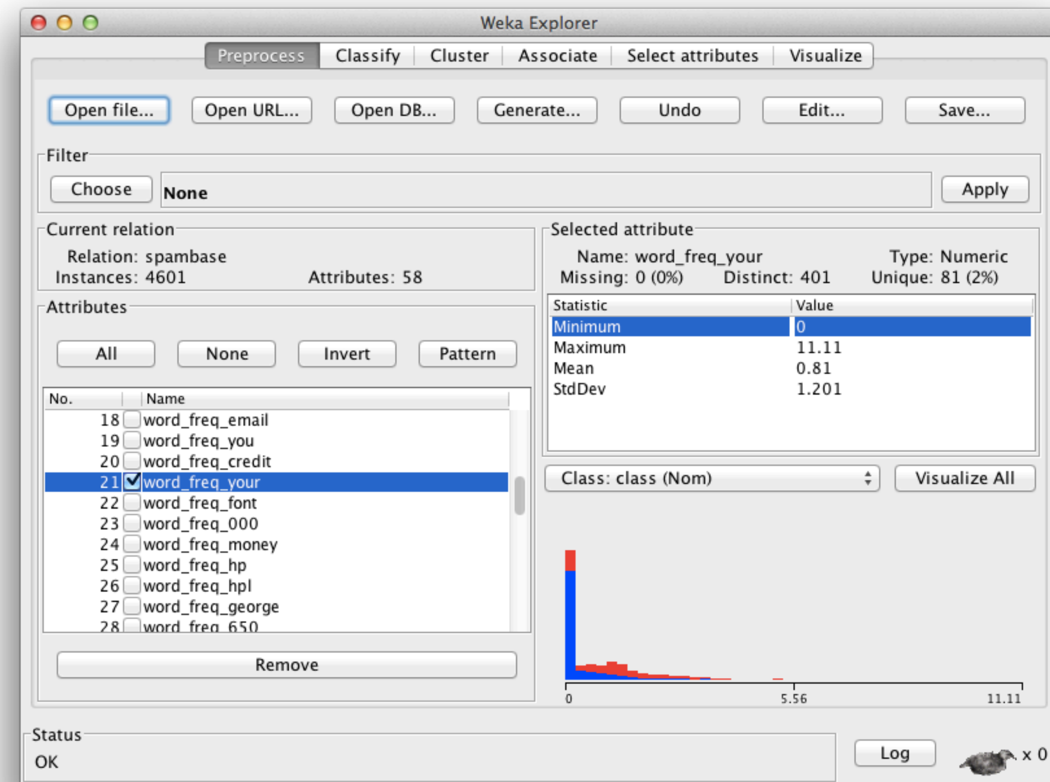
Question -> input data -> features -> algorithm -> parameter -> evaluation

**Our algorithm:**

Find a value C

If Frequency of 'your' > C

predict "SPAM"



# SPAM example

## Question -> input data -> features -> algorithm -> parameter -> evaluation

Scheme:weka.classifiers.trees.DecisionStump
Relation: spambase-weka.filters.unsupervised.attribute.Remove-R1-20,22-57
Instances: 4601
Attributes: 2
word_freq_your
class
Test mode:evaluate on training data
=== Classifier model (full training set) ===
Decision Stump
Classifications

word_freq_your <= 0.405 : 0
word_freq_your > 0.405 : 1
word_freq_your is missing : 0

Class distributions
word_freq_your <= 0.405
0 1
0.831111111111111 0.1688888888888889
word_freq_your > 0.405
0 1
0.3438381937911571 0.6561618062088429
word_freq_your is missing
0 1
0.6059552271245382 0.3940447728754618
Time taken to build model: 0.01 seconds

# SPAM example

## Question -> input data -> features -> algorithm -> parameter -> evaluation

=== Evaluation on training set ===
=== Summary ===

| | | |
|---|---|---|
| Correctly Classified Instances | 3452 | 75.0272 % |
| Incorrectly Classified Instances | 1149 | 24.9728 % |
| Kappa statistic | 0.4924 | |
| Mean absolute error | 0.3595 | |
| Root mean squared error | 0.424 | |
| Relative absolute error | 75.2817 % | |
| Root relative squared error | 86.7659 % | |
| Total Number of Instances | 4601 | |

=== Detailed Accuracy By Class ===

| | TP Rate | FP Rate | Precision | Recall | F-Measure | ROC Area | Class |
|---|---|---|---|---|---|---|---|
| | 0.738 | 0.231 | 0.831 | 0.738 | 0.782 | 0.754 | 0 |
| | 0.769 | 0.262 | 0.656 | 0.769 | 0.708 | 0.754 | 1 |
| Weighted Avg. | 0.75 | 0.243 | 0.762 | 0.75 | 0.753 | 0.754 | |

=== Confusion Matrix ===

```
    a    b   <-- classified as
 2057  731 |    a = 0
  418 1395 |    b = 1
```

# Relative order of importance

Question > input data > features > algorithm

# Data is important

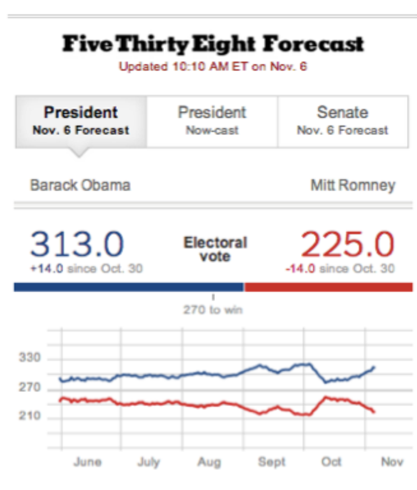
- "The combination of some data and an aching desire for an answer does not ensure that a reasonable answer extracted from a given body of data"

  John Tukey

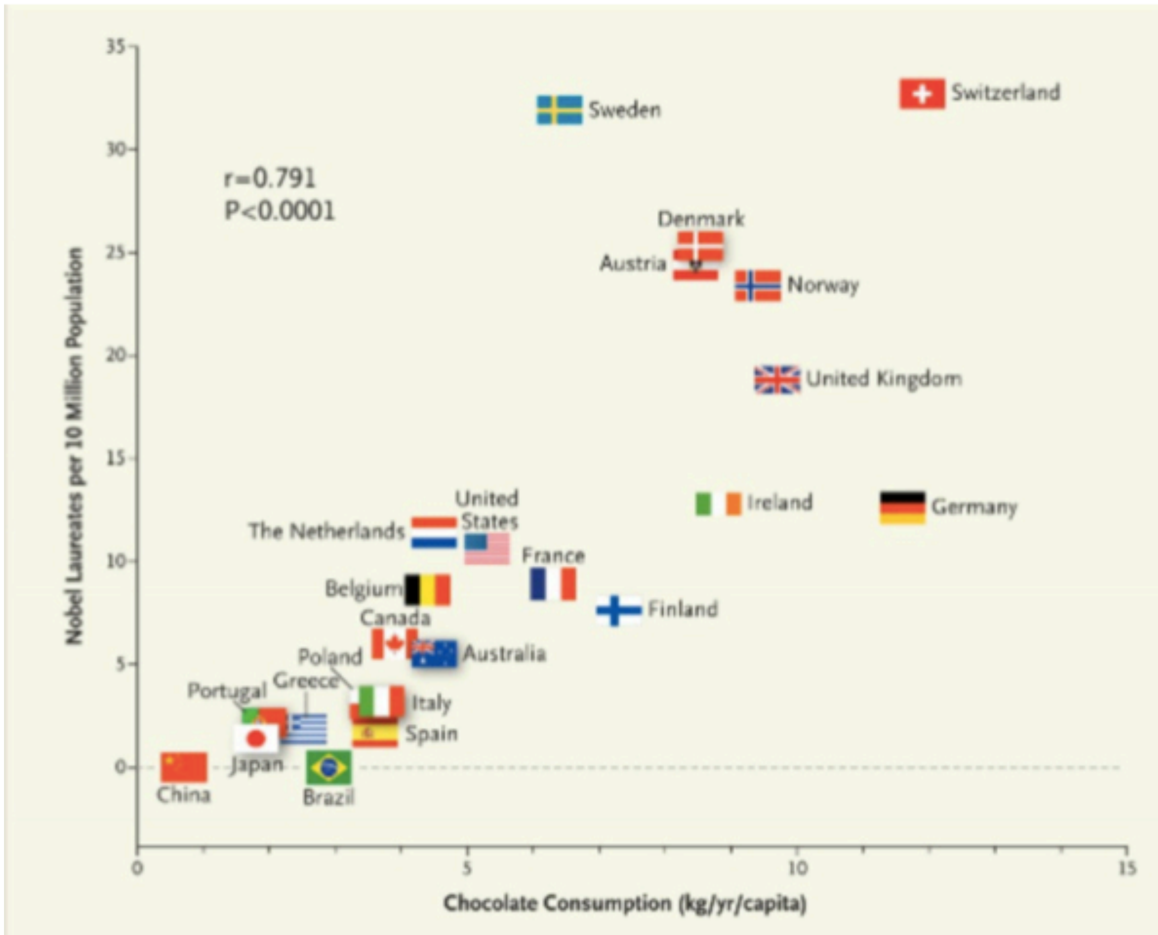
- Garbage in = Garbage out
- To predict X, use data related to X

# A successful predictor

- Election forecasting model: successful in 2008 and 2012 US elections
- Use polling information from a wide variety of polls: data asking the same questions
- Weight the polls by their bias: recognize the quirks in the data



# Unrelated data
# most common mistake



# Features matter

- Properties of good features
  - Lead to data compression
  - Retain relevant information
  - Are created based on expert application knowledge
- Common mistakes
  - Trying to automate feature selection blindly
  - Not paying attention to data-specific quirks
  - Throwing away information unnecessarily

# Features creation

- Raw data to features

HI

WE'VE DISCOVERED YOU ARE THE HEIR TO AN INCREDIBLE FORTUNE. PLEASE SUBMIT YOUR NAME, ADDRESS AND BANK ACCOUNT SO WE CAN SEND YOU $$$$$$$.

JOE JOHNSON

→

| capitalAve | you | numDollar | ... |
|---|---|---|---|
| 1 | 2 | 8 | ... |

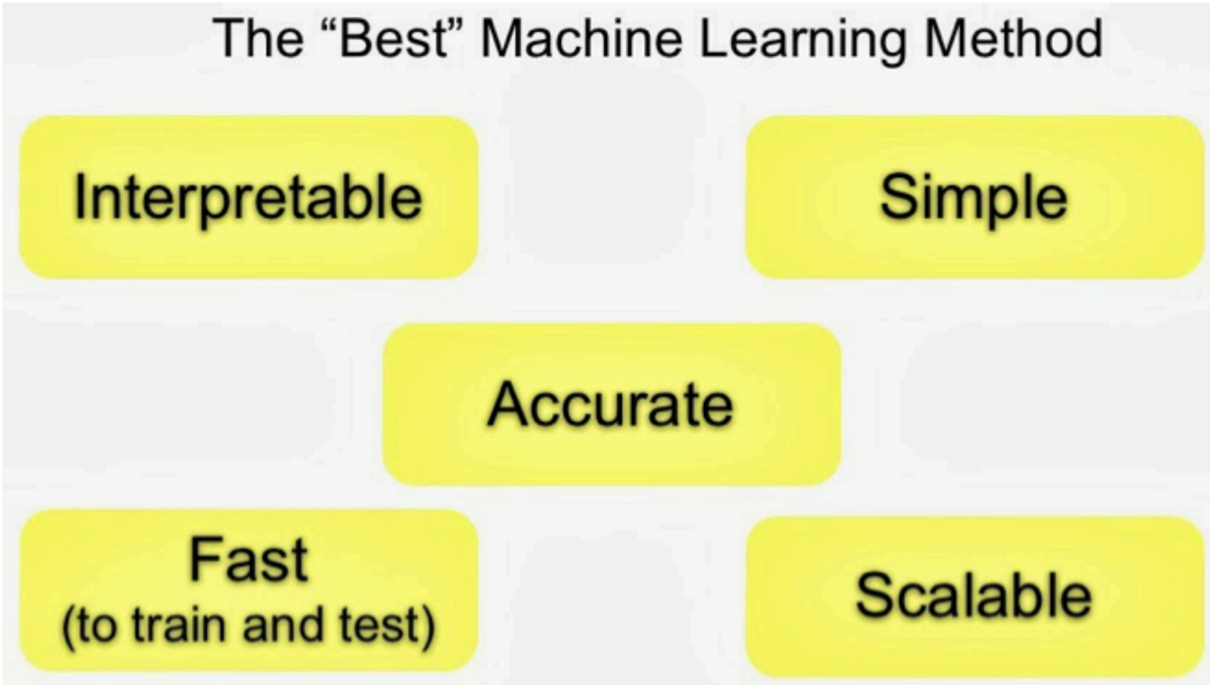
# Features creation

- Depends heavily on application
- Balance between summarization and information loss
- Examples:
  - Text files: freq. of words, freq. of phrases (ngrams), freq. of capital letters
  - Images: edges, corners, blobs, ridges
  - Webpages: number and types of images, positions of elements, colors, videos
  - People: height, weight, hair color, sex, nationality
- When in doubt -> use more features

# Algorithms matter less than you'd think

- A sensible approach will get you quite far in solving the problem
- Getting the best method can improve but not that much.

# Issues to consider




- Prediction is about accuracy tradeoffs
- Google Flu Trend: interpretability
- Netflix prize: scalability

# Type of errors: basic terms

- Binary prediction:
  - *Positive* = identified; *negative* = rejected
- **True positive** = correctly identified
  - Sick people correctly diagnosed as sick
- **False positive** = incorrectly identified
  - Healthy people incorrectly diagnosed as sick
- **True negative** = correctly rejected
  - Healthy people correctly diagnosed as healthy
- **False negative** = incorrectly rejected
  - Sick people incorrectly identified as healthy

# Errors: key quantities

| prediction | Positive | Negative |
| --- | --- | --- |
| Positive | TP | FP |
| Negative | FN | TN |

- **Sensitivity** (***recall***): TP/(TP+FN)
- Specificity: TN/(FP+TN)
- Positive Predictive Value (***precision***):
  TP/(TP+FP)
- Negative Predictive Value: TN/(FN+TN)
- **Accuracy**: (TP+TN)/(TP+FP+FN+TN)

# Error: other measures

**Continuous data**

- Mean Squared Error (MSE)

$$\frac{1}{n}\sum_{i=1}^{n}(\text{Prediction}_i - \text{Truth}_i)^2$$

- Root Mean Squared Error (RMSE)

**Multiclass cases**

- Concordance e.g. kappa
- Confusion matrix

# Evaluation

- Training error vs. testing error
- Training error: the error rate you get on the same data set you use to build your predictor
- Testing error: The error rate you get on a new data set.
- Overfitting: matching your algorithm to the data you have

# Prediction design study

- Decide on your error measure
- Split data into: Training, Testing, Validation (optional)
- On the training set:
  - Pick features and algorithms
- If no validation: Apply ONCE to the test set
- If validation:
  - Apply to test set and refine
  - Apply ONCE to validation
- Set the test/validation data aside, DO NOT look at it

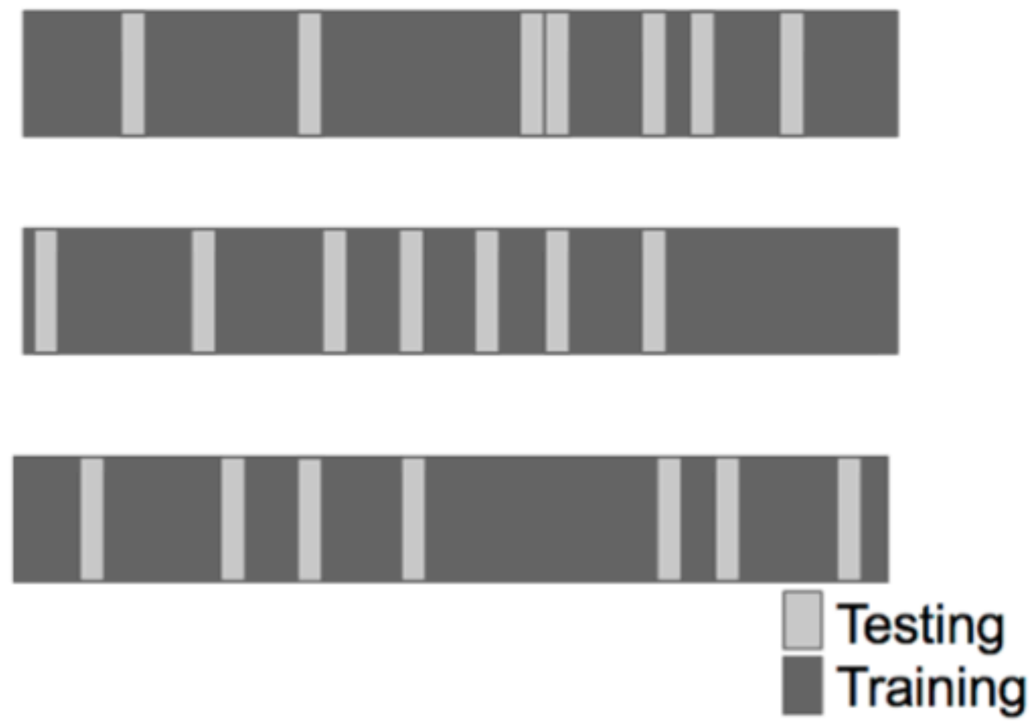
# Common practice

- If you have a large sample size
  - 60% training
  - 20% test
  - 20% validation
- If you have a medium sample size
  - 60% training
  - 40% testing
- If you have a small sample size
  - Cross validation

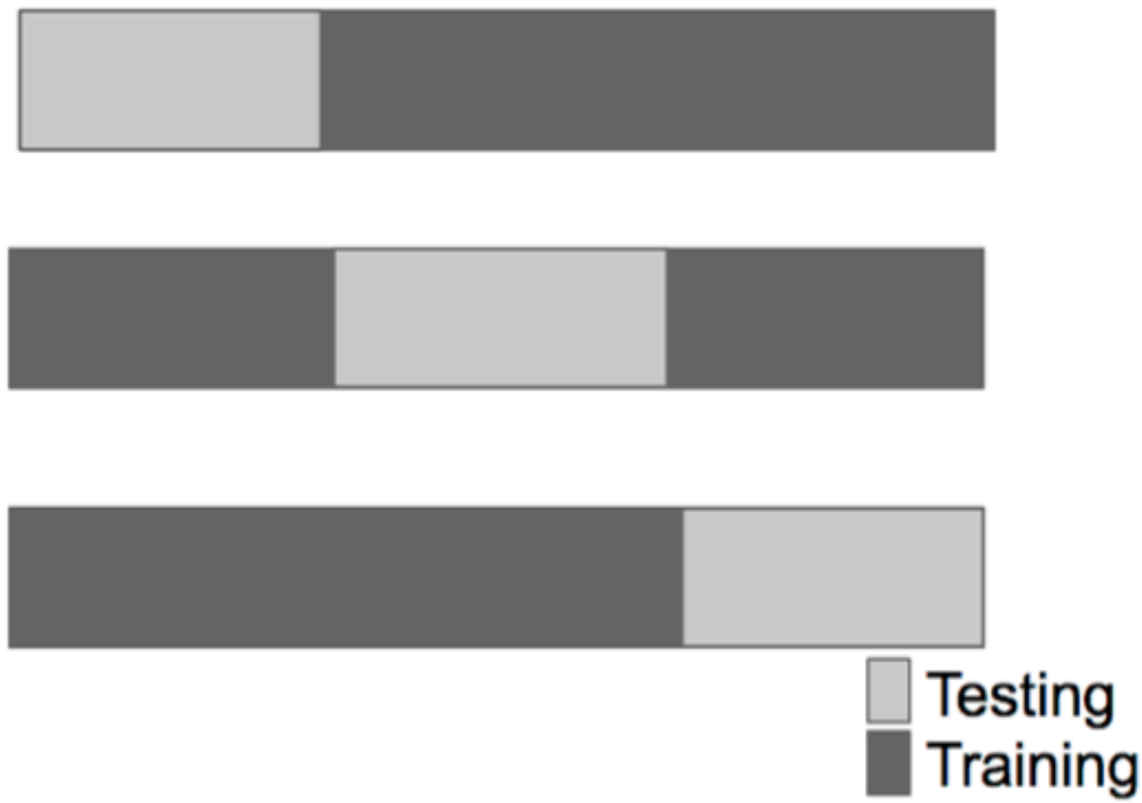
# Cross validation

- Use the training set
- Split it into training/test sets
- Build a model on the training set
- Evaluate on the test set
- Repeat and average the estimated errors
- Used for:
  - Picking features
  - Picking the type of prediction function
  - Picking parameters
  - Comparing different predictors

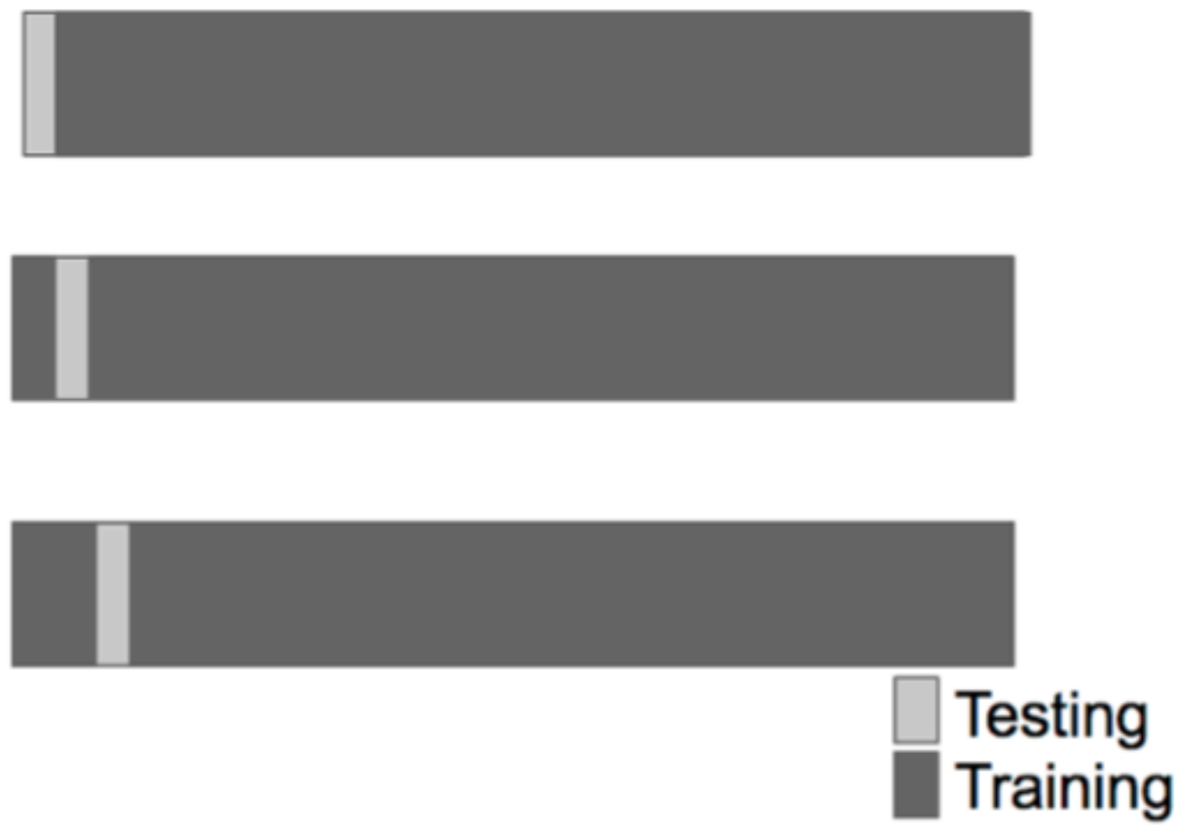
# Cross validation: random subsampling



# Cross validation: k-fold

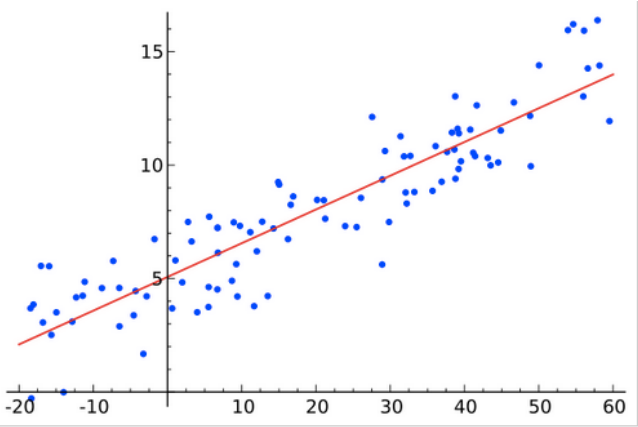


# Cross validation: leave one out

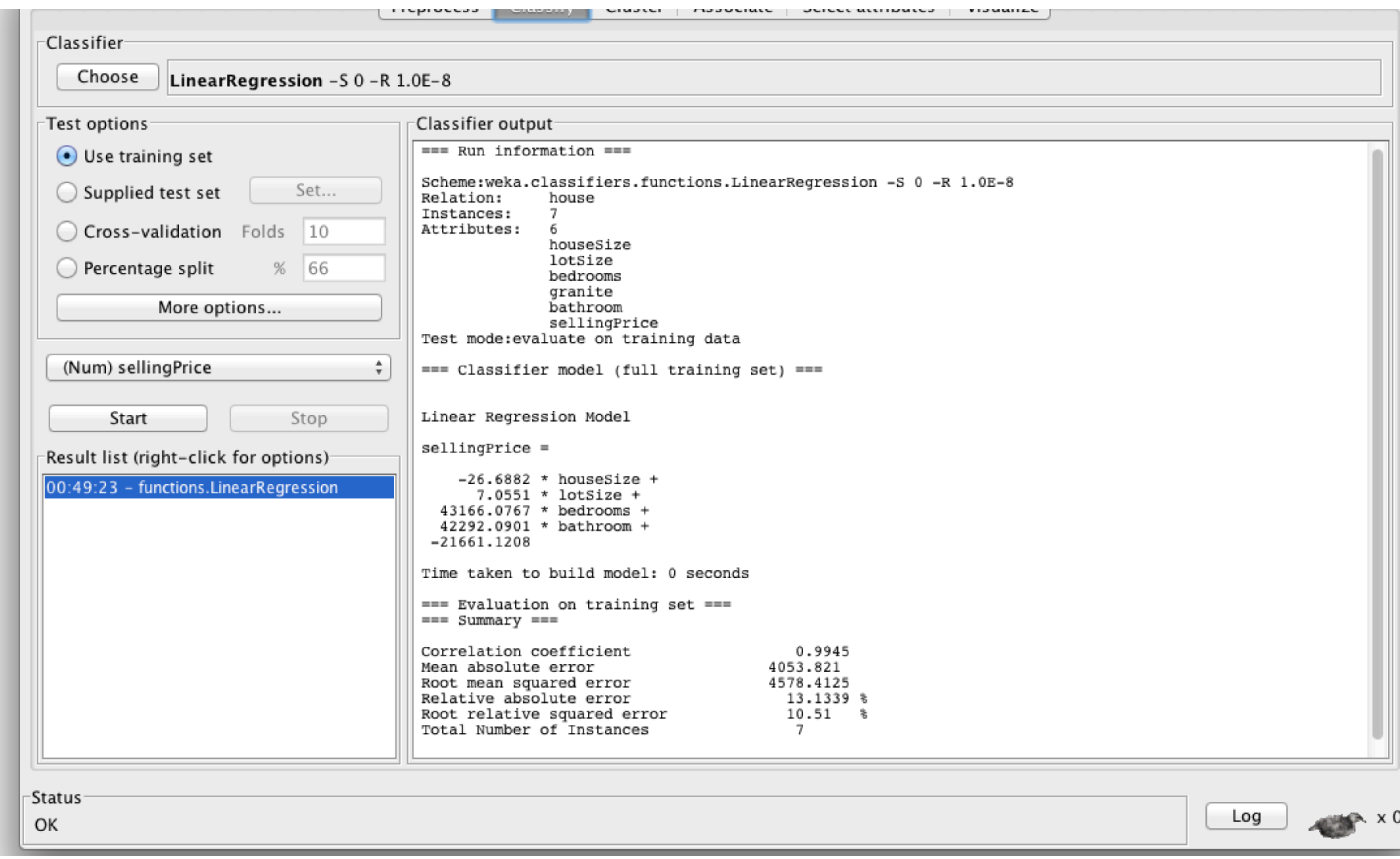


# Linear Regression

- Key ideas:
  - Fit a simple regression model: fit a line to a set of data
  - Plug in new variables and multiply by the coefficients
  - Useful when the linear model is (nearly) correct
- Pros
  - Easy to implement
  - Easy to interpret
- Cons
  - Often poor performance in nonlinear setting



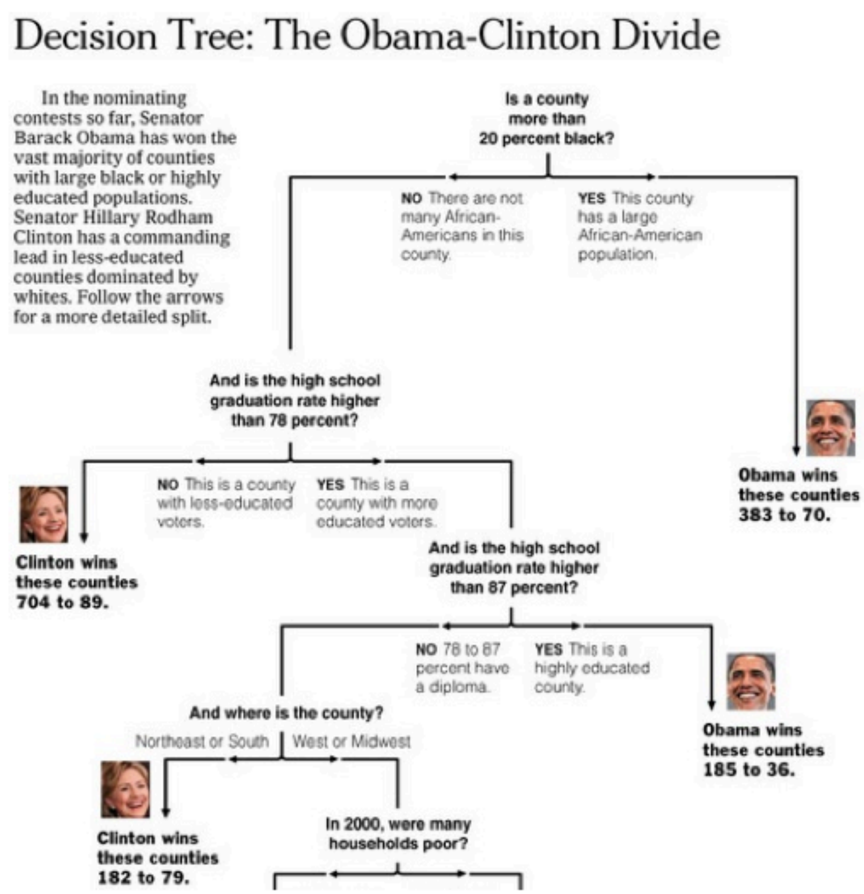
# Example: House price prediction



# Decision Tree

- Key ideas
  - Iteratively split features into groups
  - Evaluate "homogeneity" within each group
  - Split again if necessary
- Pros
  - Easy to interpret
  - Better performance in nonlinear settings
- Cons
  - Without pruning can lead to overfitting
  - Result may be variable

# Decision Tree Example



# Decision tree: basic algorithm

1. Start with all features in one group
2. Find the features/split that best separates the outcomes
3. Divide the data into two groups (leaves) on that split (node)
4. Within each split, find the best feature/split that separate the outcomes
5. Continue until the groups are too small or sufficiently "pure"

# Measure of impurity

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \text{ in Leaf } m} \mathbb{1}(y_i = k)$$

**Misclassification Error**:

$$1 - \hat{p}_{mk(m)}; k(m) = most; common; k$$

- 0 = perfect purity
- 0.5 = no purity

**Gini index**:

$$\sum_{k \neq k'} \hat{p}_{mk} \times \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) = 1 - \sum_{k=1}^{K} p_{mk}^2$$

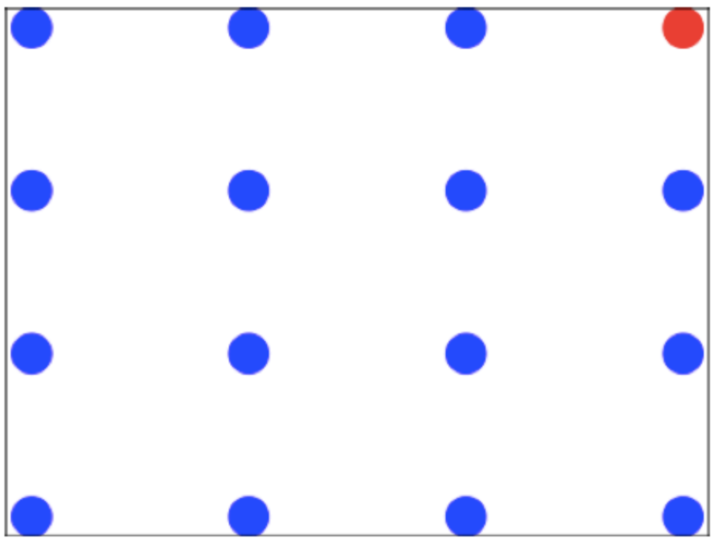
- 0 = perfect purity
- 0.5 = no purity

**Deviance/information gain**:

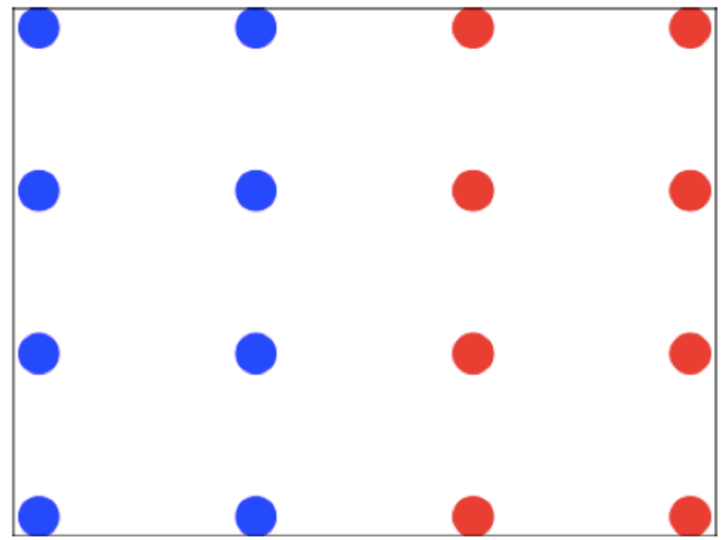
$$-\sum_{k=1}^{K} \hat{p}_{mk} \log_2 \hat{p}_{mk}$$

- 0 = perfect purity
- 1 = no purity

# Measure of purity



- **Misclassification:** 1/16 = 0.06
- **Gini:** $1 - [(1/16)^2 + (15/16)^2] = 0.12$
- **Information:**
  $-[1/16 \times \log2(1/16) + 15/16 \times \log2(15/16)] = 0.34$



- **Misclassification:** 8/16 = 0.5
- **Gini:** $1 - [(8/16)^2 + (8/16)^2] = 0.5$
- **Information:**
  $-[1/16 \times \log2(1/16) + 15/16 \times \log2(15/16)] = 1$

# Useful resourses

- The Element of Statistical Learning. T. Hastie, R. Tibshirani, J. Friedman. http://statweb.stanford.edu/~tibs/ElemStatLearn/
- https://www.coursera.org/learn/machine-learning - Stanford ML by Andrew Ng
- https://www.coursera.org/specialization/jhudatascience - Data Science specialization by Johns Hopkins