CS6510
Applied Machine Learning

Classifier System Design

17 Sep 2016

Vineeth N Balasubramanian

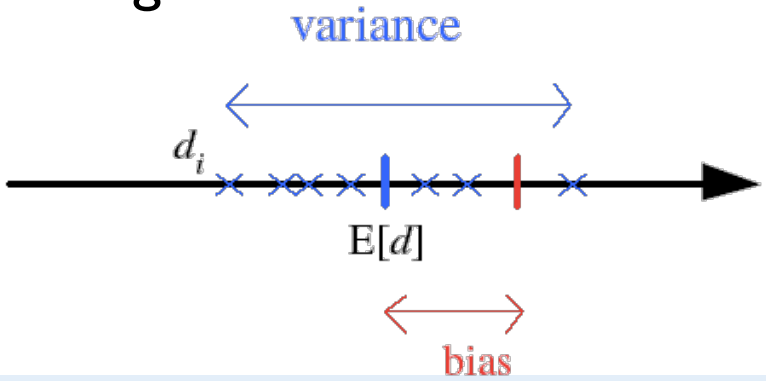
आई आई टी हैदराबाद
IIT Hyderabad

Today

- Bias-Variance Tradeoff
- Generative vs Discriminative Models
- Parametric vs Non-Parametric Models
- Handling Skewed Data
- Using Large Datasets
- Introduction to Learning Theory

Generalization Error

- Components of generalization error
  - **Bias:** how much the average model over all training sets differ from the true model?
    - Error due to inaccurate assumptions/ simplifications made by the model
  - **Variance:** how much models estimated from different training sets differ from each other



Unknown parameter q
Estimator $f_i = f(X_i)$ on sample X_i

Bias: $b_q(f) = E[f] - q$
Variance: $E[(f - E[f])^2]$

Mean square error:
$r(f,q) = E[(f-q)^2]$
$= (E[f] - q)^2 + E[(f - E[f])^2] +$
Error term
= Bias^2 + Variance + Error term

Bias-Variance Tradeoff

$$E(MSE) = noise^2 + bias^2 + variance$$

Unavoidable error

Error due to incorrect assumptions

Error due to variance of training samples

See the following for mathematical derivation of bias-variance:

- http://www.inf.ed.ac.uk/teaching/courses/mlsc/Notes/Lecture4/BiasVariance.pdf

Bias-Variance Tradeoff

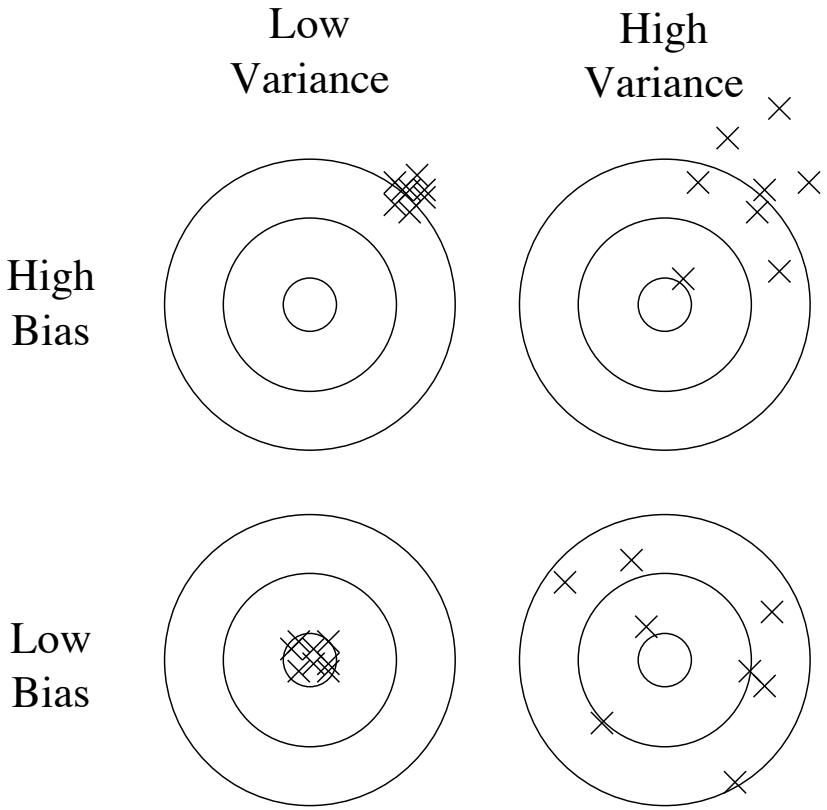
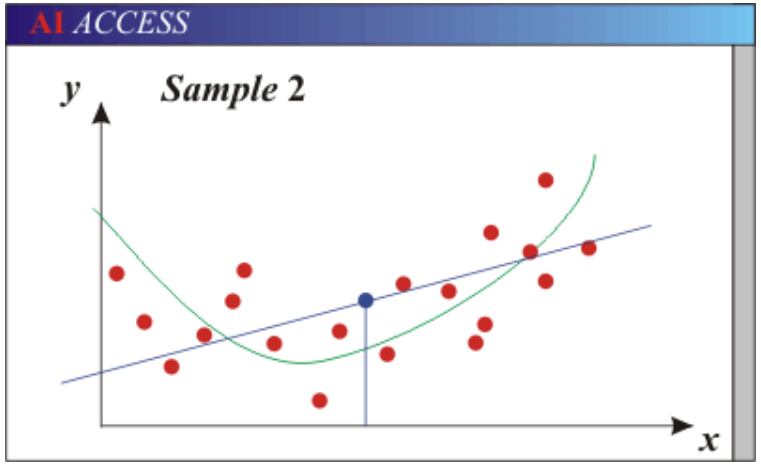
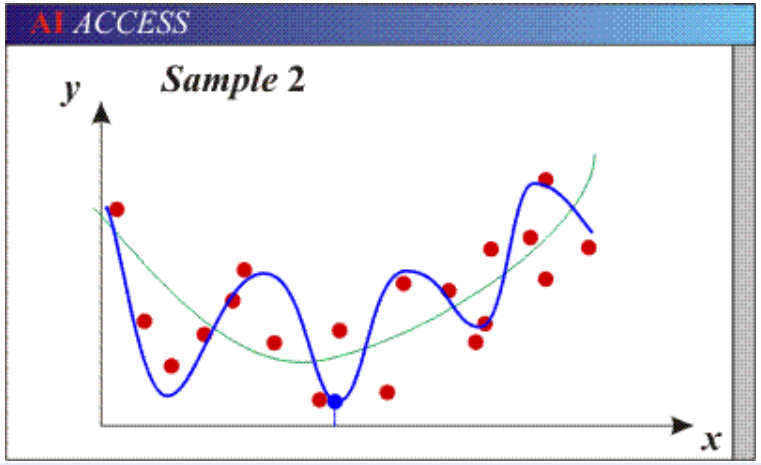


Figure 1: Bias and variance in dart-throwing.

Bias-Variance Tradeoff



- Models with too few parameters are inaccurate because of a **large bias** (not enough flexibility).



- Models with too many parameters are inaccurate because of a **large variance** (too much sensitivity to the sample).

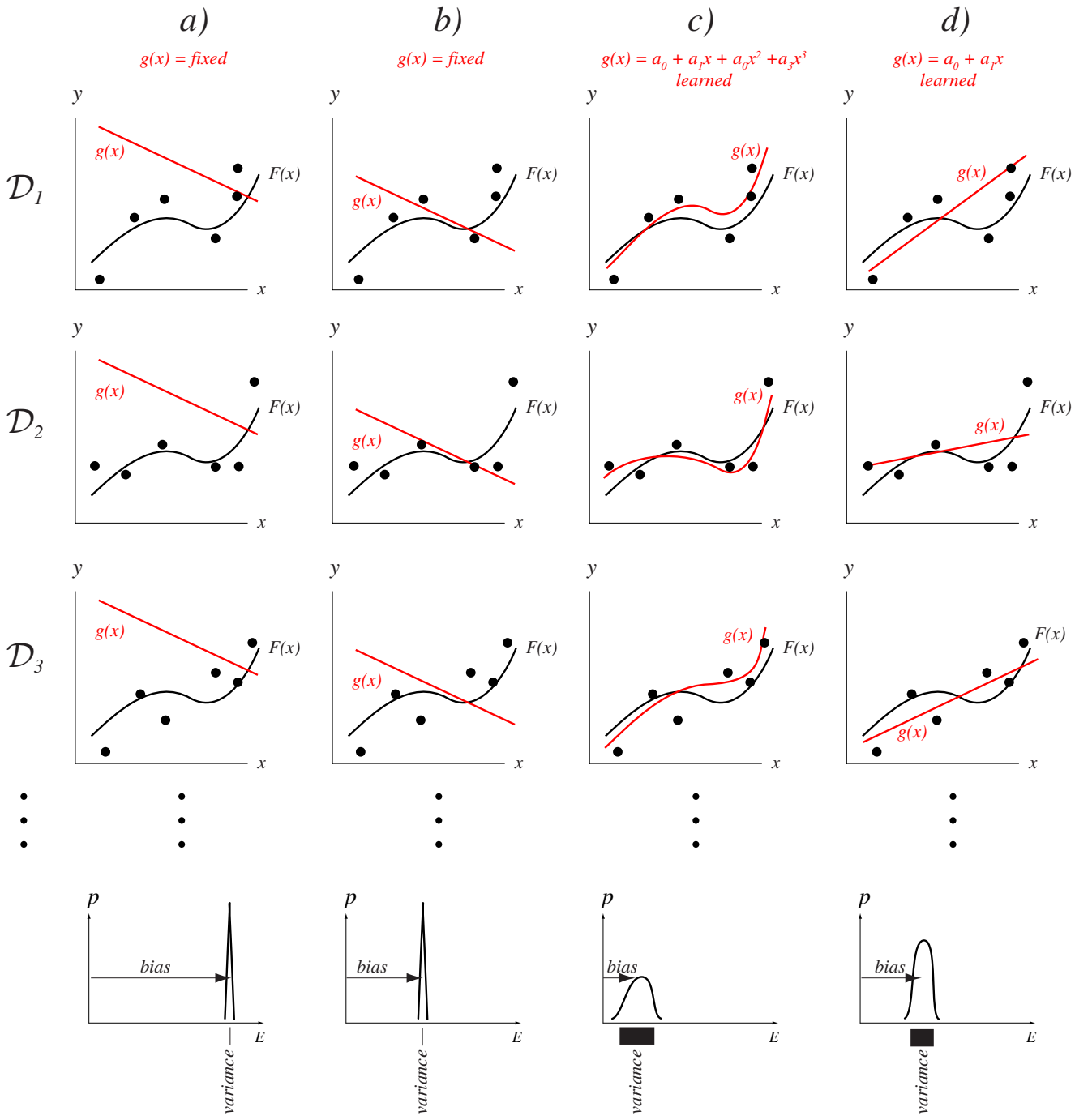
Slide Credit: D Hoiem

Generalization Error

- **Underfitting:** Model is too "simple" to represent all the relevant class characteristics
  - High bias and low variance
  - High training error and high test error

- **Overfitting:** Model is too "complex" and fits irrelevant characteristics (noise) in the data
  - Low bias and high variance
  - Low training error and high test error

Bias-Variance Tradeoff



How to reduce variance

- Choose a simpler classifier
- Regularize the parameters
- Get more training data

Training Error and Cross-Validation

- Suppose we use the *training error* to estimate the difference between the true model prediction and the learned model prediction.
- The training error is downward biased: on average it *underestimates* the generalization error.
- Cross-validation is nearly unbiased; it slightly overestimates the generalization error.

Classification

- Can do bias-variance analysis for classifiers as well.
- General principle: variance dominates bias.
- Very roughly, this is because we only need to make a discrete decision rather than get an exact value.

Measuring Bias and Variance

- In practice (unlike in theory), we have only ONE training set S.
- We can simulate multiple training sets by bootstrap replicates
  - S' = {x | x is drawn at random with replacement from S} and |S'| = |S|.
- Construct B bootstrap replicates of S (e.g., B = 200): $S_1, \ldots, S_B$
- Apply learning algorithm to each replicate S_b to obtain hypothesis h_b
- Let $T_b = S \setminus S_b$ be the data points that do not appear in S_b (out of bag points)
- Compute predicted value $h_b(x)$ for each x in T_b

Measuring Bias and Variance

- For each data point x, we will now have the observed corresponding value y and several predictions $y_1, \ldots, y_K$.
- Compute the average prediction $\underline{h}$.
- Estimate bias as $(\underline{h} - y)$
- Estimate variance as $\Sigma_k (y_k - \underline{h})^2/(K - 1)$
- Assume noise is 0
  - If we have multiple data points with the same x value, then we can estimate the noise – not generally available in machine learning
  - We can also estimate noise by pooling y values from nearby x values

Today

- Bias-Variance Tradeoff
- Generative vs Discriminative Models
- Parametric vs Non-Parametric Models
- Handling Skewed Data
- Using Large Datasets
- Introduction to Learning Theory

Categories of ML Methods

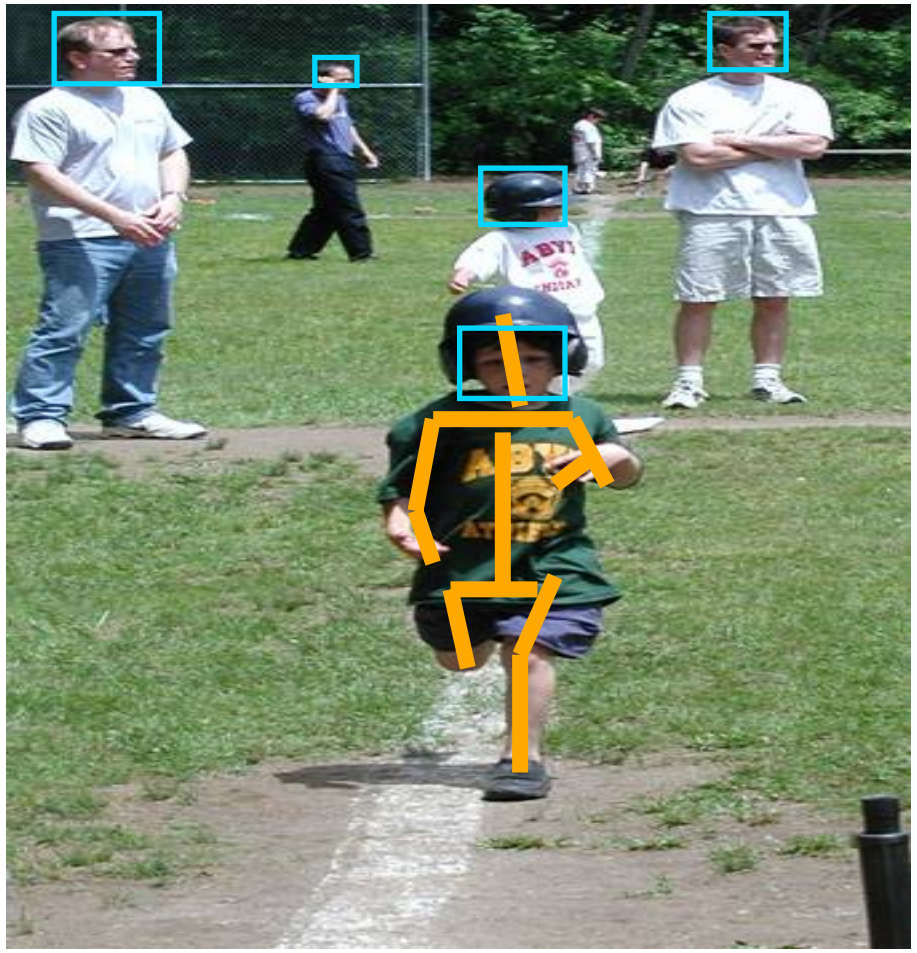
- Supervised vs Unsupervised Models
- Generative vs Discriminative Models
- Parametric vs Non-Parametric Models
- Inductive vs Transductive Models

The gap between each such complementary category pair is becoming increasingly small.

Discriminative vs Generative Models

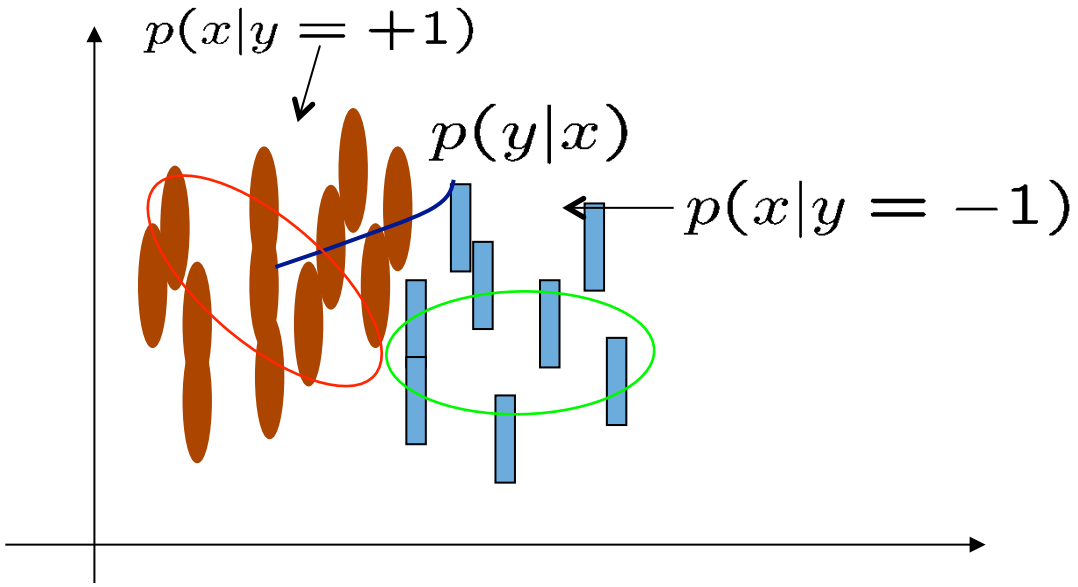
If you are asking, "Are there any faces in this image?", then you would probably want to use discriminative methods.

If you are asking, "Find a 3-d model that describes the runner",
then you would use generative methods.



ICCV W. Freeman and A. Blake

Discriminative vs Generative Models



$$p(y|x)$$

Discriminative models, either explicitly or implicitly, study the posterior distribution directly.

$$p(x|y),\ p(y)$$

Generative approaches model the likelihood and prior separately.

$$p(y|x) = \frac{p(x|y)p(y)}{\sum_y p(x|y)p(y)}$$

Generative vs Discriminative Classifiers

- Generative classifier, e.g., Naïve Bayes:
  - Assume some functional form for **P(X|Y), P(Y)**
  - Estimate parameters of P(X|Y), P(Y) directly from training data
  - Use Bayes rule to calculate P(Y|X=x)
  - This is 'generative' model
    - Indirect computation of P(Y|X) through Bayes rule
    - But, can generate a sample of the data,

- Discriminative classifier, e.g., Support Vector Machines:
  - Assume some functional form for **P(Y|X)**
  - Estimate parameters of P(Y|X) directly from training data
  - This is the 'discriminative' model
    - Directly learn P(Y|X)
    - But cannot sample data, because P(X) is not available

Discriminative vs Generative Models

- Discriminative Models: Examples
  - SVM
  - Decision Trees
  - Neural Networks
  - Boosting
  - K-NN
- Generative Models: Examples
  - Naïve Bayes classifier
  - We will see more if we have time later in the course (Markov Random Fields, Conditional Random Fields)

Parametric vs Non-parametric Models

- Recall: the core objective of prediction in machine learning is to learn a function f such that:
  - Y = f(x)
- Form of function is unknown - evaluate different machine learning algorithms and see which is better at approximating the underlying function.
- Different algorithms make different assumptions or biases about the form of the function and how it can be learned.

Parametric vs Non-parametric Models

- Parametric Models
  - Have fixed number of parameters
  - Make some strong assumptions about data
  - E.g. Linear regression
- Non-parametric Models
  - Flexible number of parameters
  - Number of parameters often grows with more data
  - Makes fewer assumptions about the data
  - E.g. k-Nearest Neighbors

Parametric Models

- "A learning model that summarizes data with a set of parameters of fixed size (independent of the number of training examples) is called a parametric model. No matter how much data you throw at a parametric model, it won't change its mind about how many parameters it needs."

  - Artificial Intelligence: A Modern Approach, page 737

Parametric Models

- Parametric algorithms involve two steps:
  - Select a form for the function.
  - Learn the coefficients for the function from the training data.
- An easy to understand functional form for the mapping function is a line, as is used in linear regression:
  - b0 + b1*x1 + b2*x2 = 0
  - Where b0, b1 and b2 are the coefficients of the line that control the intercept and slope, and x1 and x2 are two input variables.

Parametric Models

- Examples:
  - Logistic Regression (we will see this later)
  - Linear Discriminant Analysis (we will see this later)
  - Perceptron
  - Naive Bayes (in traditional usage settings)
  - Simple Neural Networks

Parametric Models: Pros and Cons

- Pros:
  - **Simpler:** These methods are easier to understand and interpret results.
  - **Speed:** Parametric models are very fast to learn from data.
  - **Less Data:** They do not require as much training data and can work well even if the fit to the data is not perfect.
- Cons:
  - **Constrained:** By choosing a functional form these methods are highly constrained to the specified form.
  - **Limited Complexity:** The methods are more suited to simpler problems.
  - **Poor Fit:** In practice the methods are unlikely to match the underlying mapping function.

Non-Parametric Models

- "Nonparametric methods are good when you have a lot of data and no prior knowledge, and when you don't want to worry too much about choosing just the right features."

  - Artificial Intelligence: A Modern Approach, page 757

Non-Parametric Models

- Examples:
  - k-Nearest Neighbors
  - Decision Trees
  - Support Vector Machines (kernel SVMs)
  - Deep Neural Networks (with complex activation functions)

Non-Parametric Models: Pros and Cons

- Pros:
  - **Flexibility:** Capable of fitting a large number of functional forms.
  - **Power:** No assumptions (or weak assumptions) about the underlying function.
  - **Performance:** Can result in higher performance models for prediction.
- Cons:
  - **More data:** Require a lot more training data to estimate the mapping function.
  - **Slower:** A lot slower to train as they often have far more parameters to train.
  - **Overfitting:** More of a risk to overfit the training data and it is harder to explain why specific predictions are made.

Today

- Bias-Variance Tradeoff
- Generative vs Discriminative Models
- Parametric vs Non-Parametric Models
- Handling Skewed Data
- Using Large Datasets
- Introduction to Learning Theory

Standard Assumption

- The data sets are balanced: i.e., there are as many positive examples of the concept as there are negative ones.
- ***Example:*** Our database of sick and healthy patients contains as many examples of sick patients as it does of healthy ones.
- Not true in practice!
- Many more examples
  - Helicopter Gearbox Fault Monitoring
  - Discrimination between Earthquakes and Nuclear Explosions
  - Document Filtering
  - Detection of Oil Spills
  - Detection of Fraudulent Telephone Calls

Why the problem?

- Standard learners are often biased towards the majority class.
- That is because these classifiers attempt to reduce global quantities such as the error rate, not taking the data distribution into consideration.
- As a result, examples from the overwhelming class are well-classified whereas examples from the minority class tend to be misclassified.

Are all classifiers sensitive to class imbalance?

- **Decision Tree:** Very sensitive to class imbalances. This is because the algorithm works globally, not paying attention to specific data points.
- **Multi-Layer perceptrons (MLPs):** MLPs are less prone to the class imbalance problem. This is because of their flexibility: their solution gets adjusted by each data point in a bottom-up manner as well as by the overall data set in a top-down manner.
- **Support Vector Machines (SVMs)** SVMs are even less prone to the class imbalance problem than MLPs because they are only concerned with a few support vectors, the data points located close to the boundaries.

See Nathalie Japkowicz' work on "Inductive Learning from Imbalanced Datasets"

Handling Class Imbalance: Ideas

- Collect more data!
- Change your performance metric:
  - Confusion Matrix, Precision-Recall, F1-score, etc.
- Resample dataset
- Generate synthetic samples
- Try penalized models
- Try a different perspective – anomaly/change detection

http://machinelearningmastery.com/

Handling Class Imbalance

- At the data Level: Re-Sampling
  - Oversampling (Random or Directed)
  - Undersampling (Random or Directed)
  - Active Sampling
- At the Algorithmic Level:
  - Adjusting the Costs
  - Adjusting the decision threshold / probabilistic estimate at the tree leaf

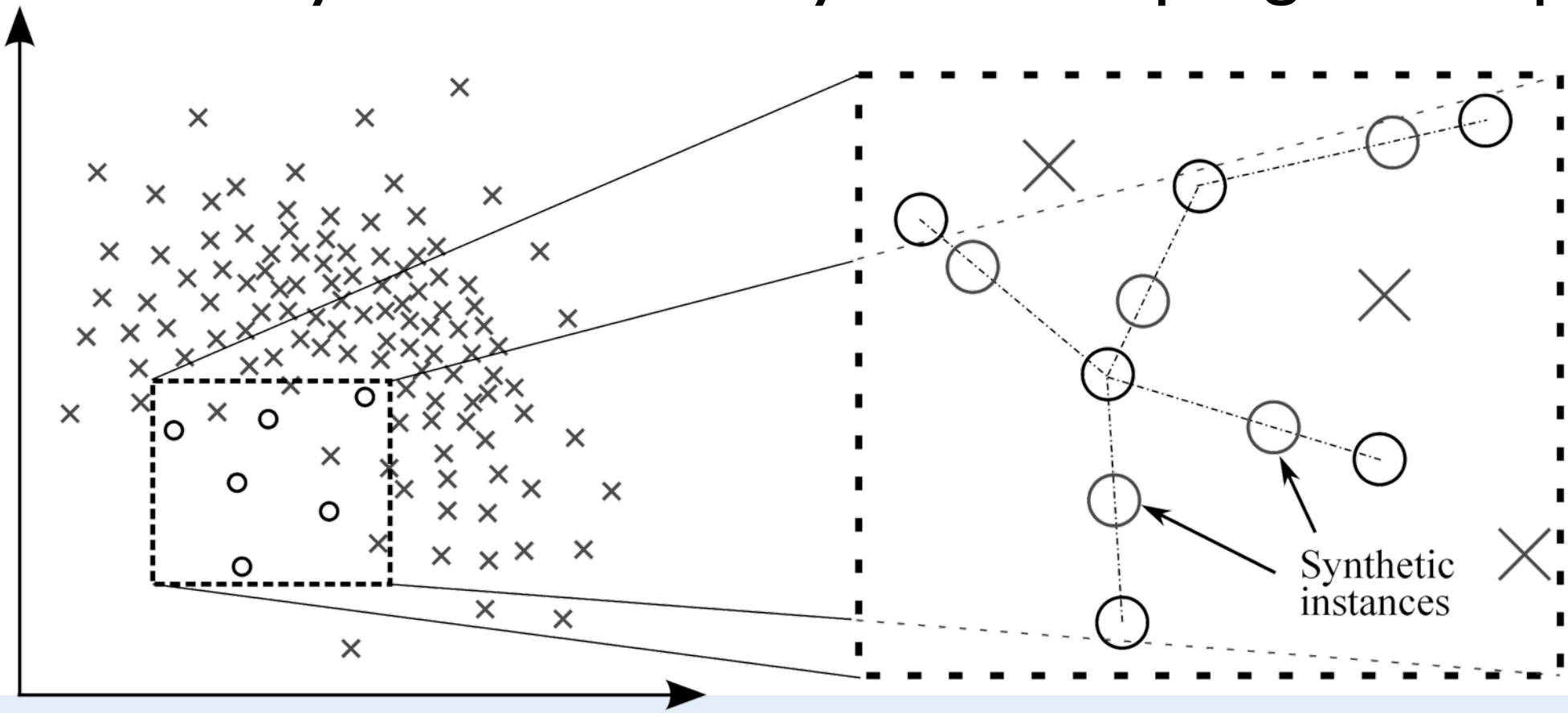
Handling Class Imbalance

- Undersampling (random and directed) is not effective and can even hurt performance.
- Random oversampling helps quite dramatically at all complexity. Directed oversampling makes a bit of a difference by helping slightly more.
- Cost-adjusting is about as effective as Directed oversampling. Generally, however, it is found to be slightly more useful.

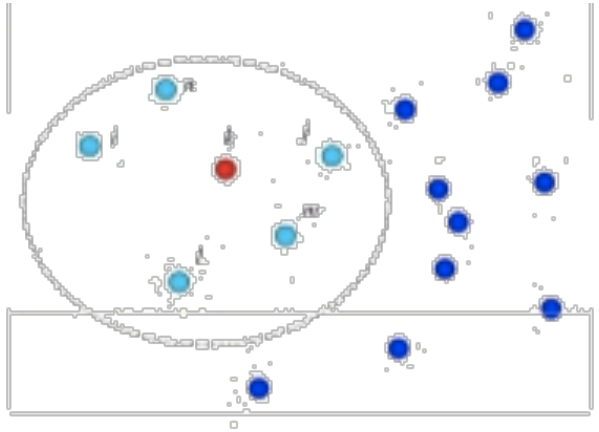
See Nathalie Japkowicz' work on "Inductive Learning from Imbalanced Datasets"

Example: SMOTE

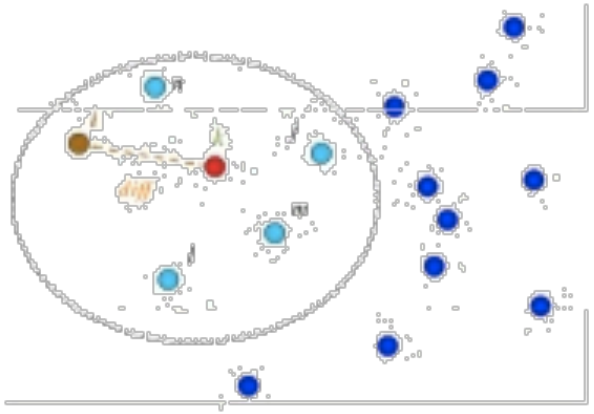
- SMOTE = Synthetic Minority Oversampling Technique



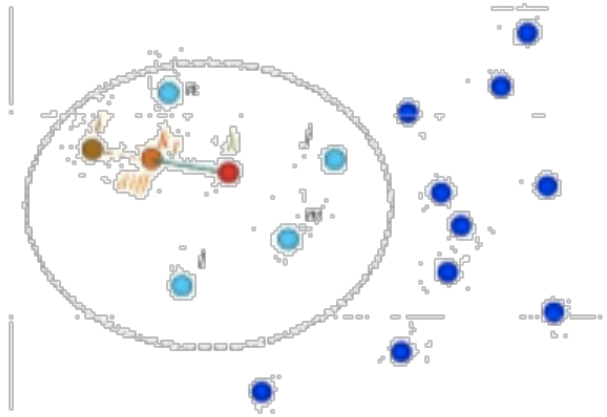
Example: SMOTE



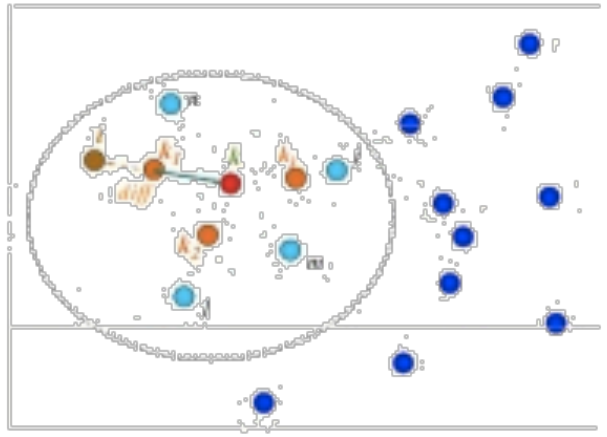
1. For each minority example k compute nearest minority class examples (i, j, l, n, m)



2. Randomly choose an example out of 5 closest points



3. Synthetically generate event k_1, such that k_1 lies between k and i



4. Dataset after applying SMOTE 3 times

Today

- Bias-Variance Tradeoff
- Generative vs Discriminative Models
- Parametric vs Non-Parametric Models
- Handling Skewed Data
- Using Large Datasets
- Introduction to Learning Theory

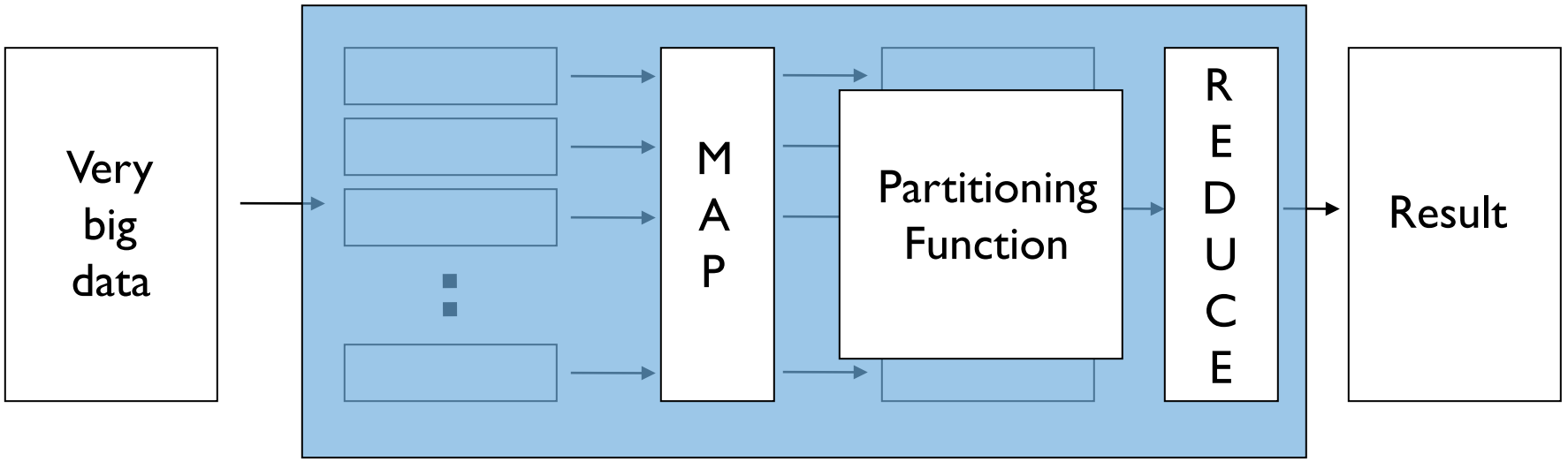
Using Large Datasets

- At large data scales, the performance of different algorithms converge such that performance differences virtually disappear.
- Given a large enough data set, the algorithm you'd want to use is the one that is computationally less expensive.
- It's only at smaller data scales that the performance differences between algorithms matter.
- More:
  - Data-Intensive Text Mining with MapReduce by Lin and Dyer
  - Mining of Massive Datasets by Rajaraman, Leskovec and Ullman

Using Large Datasets

- CPUs vs GPUs
  - Deep learning has greatly benefited from GPUs
- Map-Reduce/Hadoop, Apache Spark, Vowpal Wabbit frameworks
  - Many learning algorithms amenable to partitioning of computations

Map-Reduce



- Map:
  - Accepts *input* key/value pair
  - Emits *intermediate* key/ value pair
- Reduce :
  - Accepts *intermediate* key/ value* pair
  - Emits *output* key/value pair

Today

- Bias-Variance Tradeoff
- Generative vs Discriminative Models
- Parametric vs Non-Parametric Models
- Handling Skewed Data
- Using Large Datasets
- Introduction to Learning Theory

Computational Learning Theory: Overview

- Are there general laws that govern learning?
  - ***Sample Complexity:*** How many training examples are needed for a learner to converge (with high probability) to a successful hypothesis?
  - ***Computational Complexity:*** How much computational effort is needed for a learner to converge (with high probability) to a successful hypothesis?
  - ***Mistake Bound:*** How many training examples will the learner misclassify before converging to a successful hypothesis?
- These questions can be answered within two analytical frameworks:
  - The ***Probably Approximately Correct (PAC)*** framework
  - The ***Mistake Bound*** framework

Ref: Chapter 7 of Machine Learning by Tom Mitchell

Computational Learning Theory: Overview

- Rather than answering these questions for individual learners, we will answer them for broad classes of learners. In particular we will consider:
  - The size or complexity of the hypothesis space considered by the learner.
  - The accuracy to which the target concept must be approximated.
  - The probability that the learner will output a successful hypothesis.
  - The manner in which training examples are presented to the learner.

Ref: Chapter 7 of Machine Learning by Tom Mitchell

PAC Learning Model

- **Definition:** Consider a concept class ***C*** defined over a set of instances ***X*** of length ***n*** and a learner ***L*** using hypothesis space ***H***. ***C*** is ***PAC-learnable*** by ***L*** using ***H*** if for all $c \in C$, distributions ***D*** over ***X***, ε such that $0 < \varepsilon < 1/2$, and δ such that $0 < \delta < 1/2$, learner ***L*** will, with probability at least $(1-\delta)$, output a hypothesis $h \in H$ such that $error_D(h) \leq \varepsilon$, in time that is ***polynomial*** in $1/\varepsilon$, $1/\delta$, n, and ***size(c)***.

Ref: Chapter 7 of Machine Learning by Tom Mitchell

Sample Complexity for Finite Hypothesis Spaces

- Given any ***consistent*** learner (commits zero errors over training data), the number of examples sufficient to assure that any hypothesis will be probably (with probability $(1-\delta)$) approximately (within error ε) correct is:

$$m = 1/\varepsilon \, (\ln|H| + \ln(1/\delta))$$

- If the learner is ***not consistent***

$$m = 1/2\varepsilon^2 \, (\ln|H| + \ln(1/\delta))$$

Ref: Chapter 7 of Machine Learning by Tom Mitchell

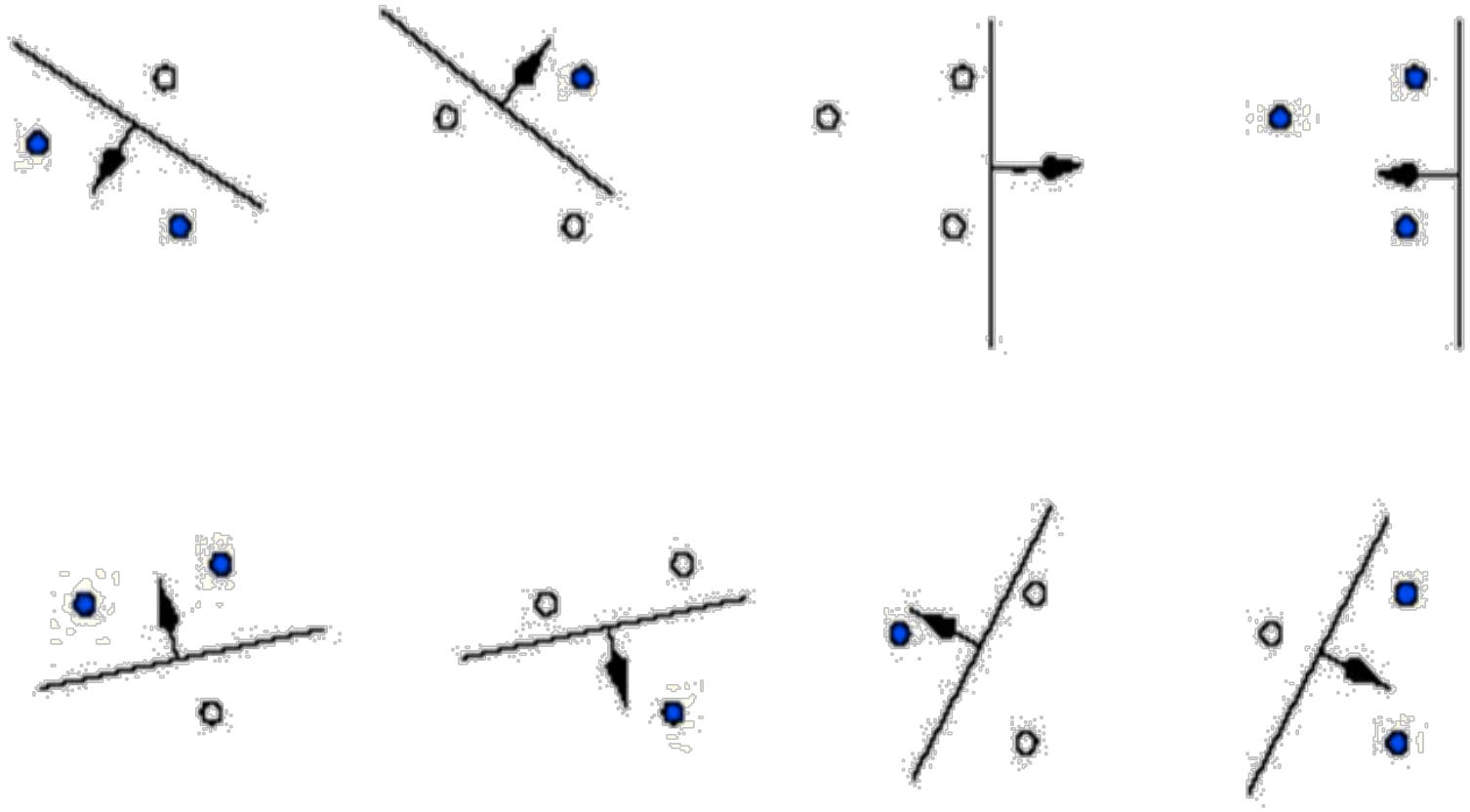
Sample Complexity for Infinite Hypothesis Spaces

- The PAC Learning framework has 2 disadvantages:
  - It can lead to weak bounds
  - Sample Complexity bound cannot be established for infinite hypothesis spaces
- There are other ideas for dealing with these problems:
  - **Definition:** A set of instances S is shattered by hypothesis space H iff for every dichotomy of S there exists some hypothesis in H consistent with this dichotomy.
  - **Definition:** The **Vapnik-Chervonenkis dimension**, VC(H), of hypothesis space H defined over instance space X is the size of the largest finite subset of X shattered by H. If arbitrarily large finite sets of X can be shattered by H, then VC(H)=∞

Ref: Chapter 7 of Machine Learning by Tom Mitchell

VC-Dimension

- VC-Dimension of set of oriented hyperplanes in R^n is (n+1)



Ref: Chapter 7 of Machine Learning by Tom Mitchell

Sample Complexity for Infinite Hypothesis Spaces

- ***Upper-Bound*** on sample complexity, using the ***VC-Dimension***

$$m \geq 1/\varepsilon\,(4\log_2(2/\delta)+8VC(H)\log_2(13/\varepsilon))$$

- ***Lower Bound*** on sample complexity, using the ***VC-Dimension***:

  Consider any concept class ***C*** such that ***VC(C) ≥ 2,*** any learner ***L,*** and any $0 < \varepsilon < 1/8$, and $0 < \delta < 1/100$. Then there exists a distribution ***D*** and target concept in ***C*** such that if ***L*** observes fewer examples than $\max[1/\varepsilon \log(1/\delta),(VC(C)-1)/(32\varepsilon)]$, then with probability at least δ, ***L*** outputs a hypothesis ***h*** having $error_D(h) > \varepsilon$.

Ref: Chapter 7 of Machine Learning by Tom Mitchell

VC Dimension for Neural Networks

- Let G be a layered directed acyclic graph with n input nodes and $s \geq 2$ internal nodes, each having at most r inputs. Let C be a concept class over $\mathbf{R}^r$ of VC dimension d, corresponding to the set of functions that can be described by each of the s internal nodes. Let C_G be the G-composition of C, corresponding to the set of functions that can be represented by G. Then $VC(C_G) \leq 2ds \log(es)$, where e is the base of the natural logarithm.
- This theorem can help us bound the VC-Dimension of a neural network and thus, its sample complexity (See, [Mitchell, p.219])!

Ref: Chapter 7 of Machine Learning by Tom Mitchell