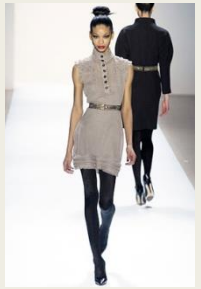
FOUNDATIONS OF MODELING AND MACHINE LEARNING

Stephen Coggeshall
Retired Chief Analytics and Science Officer, ID Analytics, Lifelock
Professor USC, UCSD
17 October, 2017

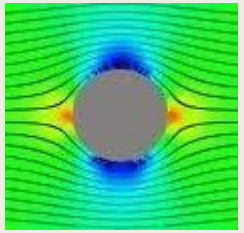
What is a Model?

- A model is a **representation of reality**

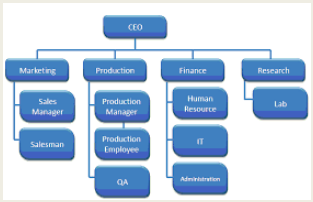


- Could be **first principles** equations

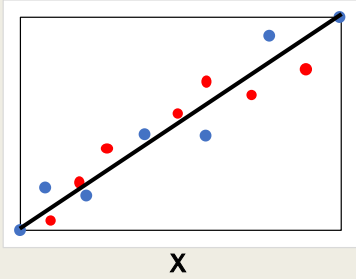
$$\rho_t + u\rho_r + \rho u_r + 2u\rho/r = 0$$
$$u_t + uu_r + \Gamma T\rho_r/\rho + \Gamma T_r = 0$$
$$T_t + uT_r + (\gamma - 1)T\left[u_r + 2u/r\right] = 0$$



- Could be **simulations**, rule systems



- Could be a **statistical model**, “learned” from data examples



What is a Statistical Model?

- A statistical model is a **functional relationship**, y = f(x), between a bunch of inputs and an output, where this relationship is **approximated from discrete data point** examples.

- Examples:
  - *Will this person pay back a loan? (credit score)*
  - *Is this healthcare claim a fraud?*
  - *Will John buy this product if I offer it to him?*
  - *Where will the stock market be in a month?*

- A statistical model is built (trained) by “showing” it a data set of examples.

There are Many Methods for Statistical Models

- Statistics community:
  - *Linear, logistic regressions*
  - *PCR, PLS*
  - *Factor analysis*
  - *ARIMA*

    …

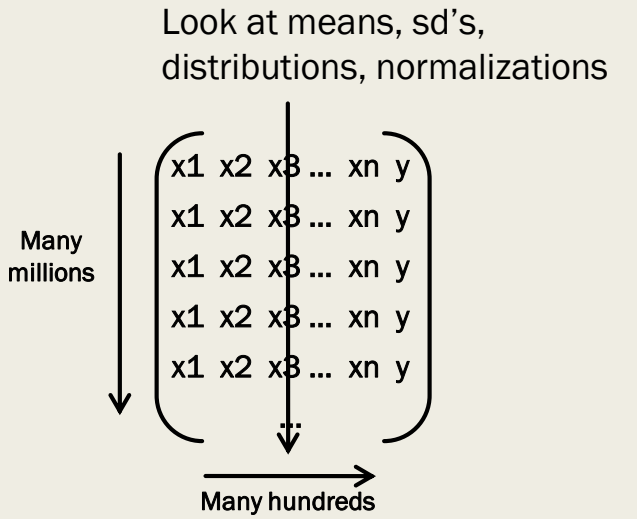
- *Generally show all the data at once.*
- *Generally linear.*

- Machine learning community
  - *Decision trees*
  - *Neural nets/MLP*
  - *SVM*
  - *Nearest neighbor*
  - *Clustering*
  - *Boosted trees*
  - *Random forests*
  - *Bayesian networks*
  - *Deep learning*
  - *Convolutional neural nets*

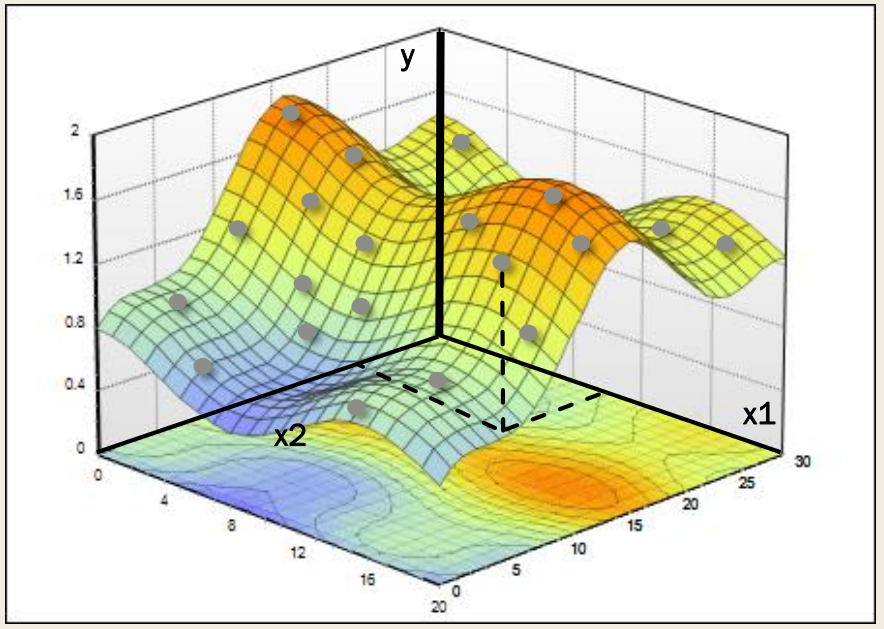
  …

- *Generally learn incrementally, point by point.*
- *Generally nonlinear.*

What Does Data Look Like?



Think this way for data preparation, statistical analysis, normalization, standardization…



Think this way for model building process, algorithm design and selection

Note – We first need to build these variables (x's) from the raw data.

How to Build Variables

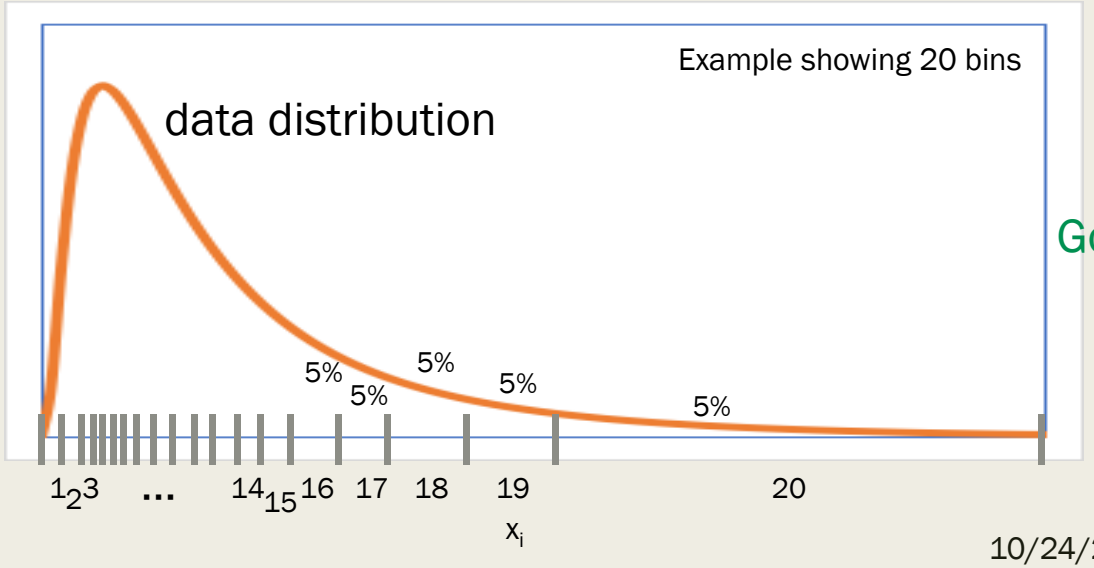
- Need to convert the raw data into normalized numeric variables
  - *Numeric fields: scale carefully, log transforms…*
  - *Categorical fields: Generally need to convert to a number*
  - *Build combinations of fields: sums, ratios, min/max's…*

- Build expert variables
  - *Talk to domain experts, understand the phenomenon as well as possible*
  - *Build special, explicit variables that best relate the raw fields to the output*
  - *Examples:*
    - Credit risk scores: open-to-buy, revolve/transact (pay/bal), max # delinquencies in past 12 months…
    - General fraud: velocity (# events/time window), # events grouped by address…
    - Tax preparer fraud: % returns with foster children, % returns with schedule C…
    - Marketing: Income/age, % purchases in a category (e.g. travel, shopping, online…)

How to Scale Variables

- Most ML algorithms are sensitive to the differences in scale of the input values:
  - *Counts are typically a few, maybe less than ~10*
  - *$ values can be thousands, millions…*
- Generally need to put all variables on the same footing or scale; Critical for clustering.
- Could divide by the range: $x' = \frac{x-\mu}{|x_{max} - x_{min}|}$, but this is very sensitive to outliers. Bad
- Better is z scaling: $x' = \frac{x-\mu}{\sigma}$, but be careful when calculating μ and σ. Good

- Another very good method is binning:
  - *Bin the data into bins of equal population*
  - *Replace the value by the bin number*
  - *Good to use a large number of bins, like 1000*
  - *Particularly good when combining multiple model outputs/scores*



Good

How to Handle Categorical Variables

Let's say you have a field x that can take values A, B, C…
How do you encode these categories A, B, C… into numbers to go into a modeling algorithm?

- Ordinal coding: Assign an integer to each category: — Bad
  - *[A, B, C…] -> [1, 2, 3…]*
  - *Problem: many modeling algorithms assume there is a metric (1 is closer to 2 than 3)*
- Dummy, one hot: Expand the category list into new variables: — Bad
  - *[A, B, C…] -> x1, x2, x3… with binary values. So a record with value "B" -> 0,1,0,0…*
  - *Problem: explodes dimensionality, which is the opposite of what we want in modeling*
  - *There are many dimensionality expanding methods (contrast, Helmert, GLM, comparison…); All have this problem*
- Risk tables (for supervised models only): For each possible category assign a specific value. Replace the field by its assigned value. — Good
  - *No dimensionality expansion. Each categorical variable becomes one continuous variable*
  - *Assigned value: average of the dependent variable for all records in that category*
  - *Good – direct encoding to what you're trying to predict*
  - *Bad - loss of interaction information*

How to Handle Missing Values

- First, don't forget that "Missing" can be a relevant value for a categorical variable (particularly for fraud)
- Ignore the records that have a missing field? – not a good approach. Could be many records
- Replace the missing field value with something reasonable:
  - *Build a model to predict the missing value, given the other values. Usually too much work; Do this only if you think its very important.*
  - *Fill in the missing value with the average or most common value of that field over all records, or*
  - *The average or most common value of that field over a relevant subset of records:*
    - Select one or more other fields that you think are important in determining the missing field
    - Bin or group the selected field(s) into categories
    - Replace the missing field with the average or most common value for its binned or other appropriate group

Fill in with
- Average # stories across all records, or
- Average # stories for records in that zip
- …

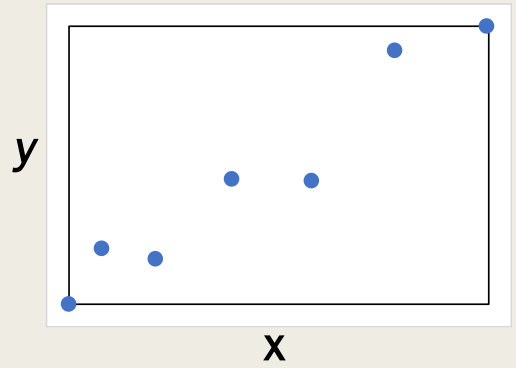
Given data (about buildings):

Record	# stories	sq feet	Zip	value
1	2	294	22041	1032
2	1	495	22043	539
3	3	3847	22042	NaN
4	3	9278	22041	4837
5	2	129	22052	462
6	NaN	948	22041	583
7	4	847	22043	2094
8	3	1029	22052	7632
9	2	947	22051	489

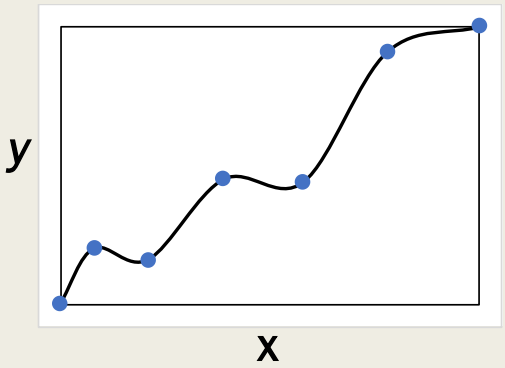
Fill in with
- Average value across all records, or
- Average value for records in zip 22042, or
- Average value for all records with # stories = 3, or
- Average value for all records in zip 22042 and # stories = 3
- …

The Dark Side of Modeling: Overfitting

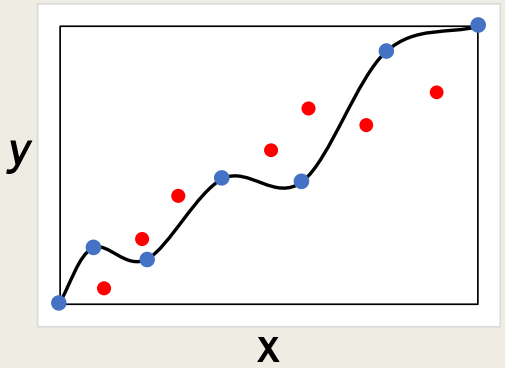
- What if you want to fit a model to this 1-d data:



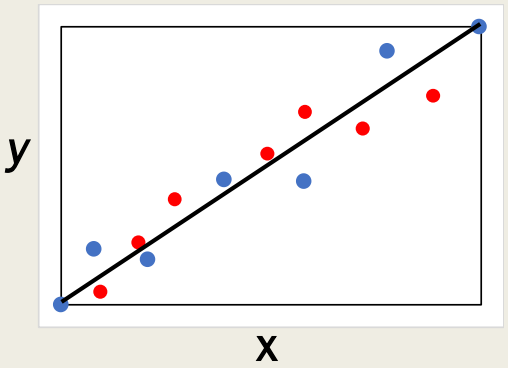
- With a sufficiently complex model you can always get a perfect fit



- But when new data comes in that your model hasn't seen before this fit can be very bad



- A simpler model likely fits new data better

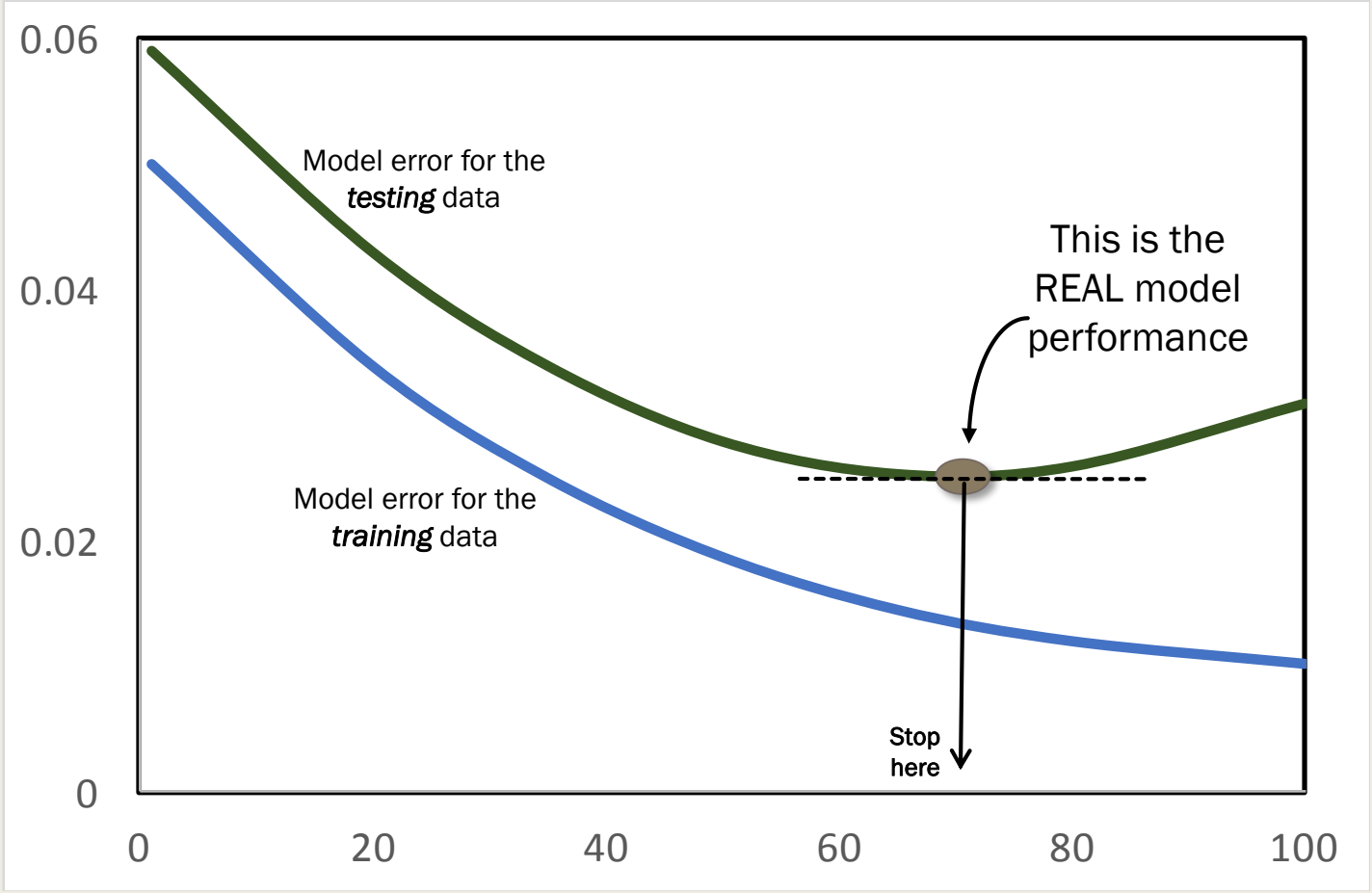


Overly-complex models don't generalize well – **Overfitting**

Must balance model complexity with generalization

How to Avoid Overfitting: Training/Testing Data

- Randomly separate the data into two sets: one for ***training*** and one for ***testing***
- Build the model on the ***training*** data, then evaluate it on the ***testing*** data



Feature Selection Methods

Always best to work in as low dimensionality as possible
Try to eliminate less important features (aka dimensions, inputs, variables…)

- **Filter** – independent of any modeling method
  - *PCA, Pierson correlations, mutual information, univariate KS…*
- **Wrapper** – uses a model "wrapped" around the feature selection process
  - *stepwise selection*
- **Embedded** – does feature selection as the model is built
  - *Trees, use of a regularization/loss function (e.g., Lasso)*

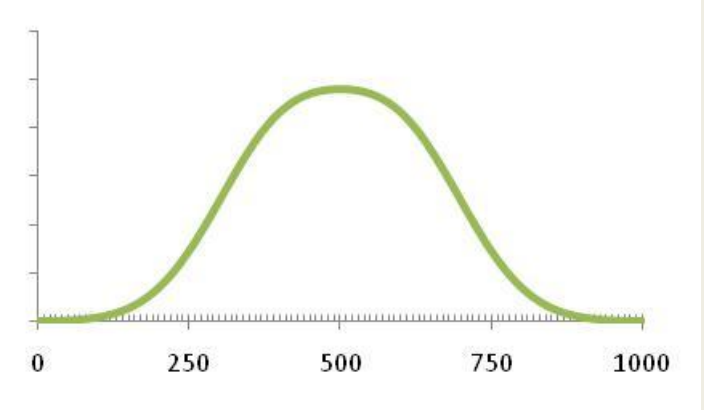
- Methods can be linear or nonlinear
- Some supervised, some unsupervised

Model Measures of Goodness

- Continuous target ($ loss, profit…). Use sum of errors
    - *Each record has an error. Just add up the errors, or the square of the errors (MSE)*

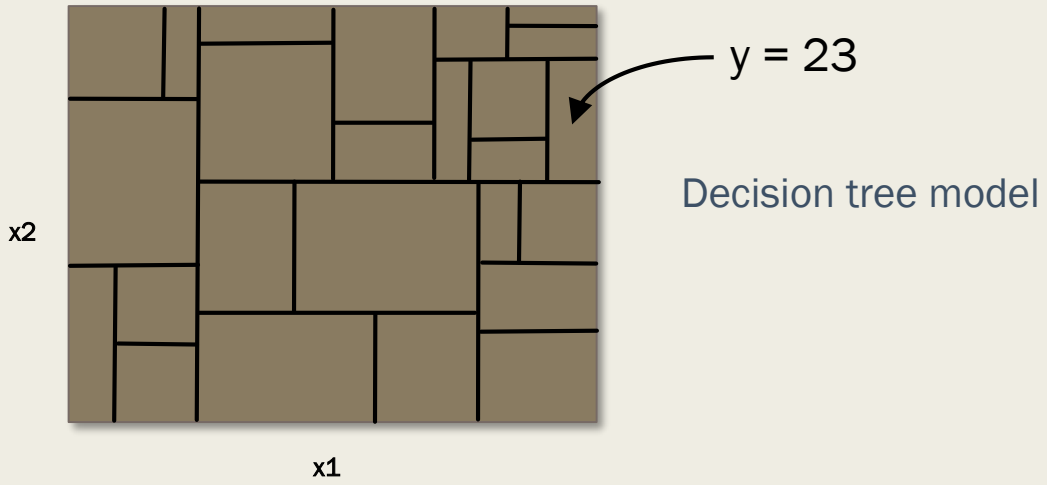
- Marketing models typically use lift
    - *Lift of 3 means response is 3 times random*

- Binary target (good/bad). Use FDR, KS, ROC
    - *KS is a robust measure of how well two distributions are separated (goods vs bads)*



Decision Trees, Boosted Trees

- A decision tree cuts up input space into boxes and assigns a value to each box



Decision tree model

- A boosted tree model is a linear combination of many weak models

Boosted decision tree model

Boosting

- A way for training a series of weak learners to result in a strong learner. Any learner can be used but trees are the most common.
- First boosting algorithms (1989) built the next weak learner using misclassified records.
- Adaptive Boosting (AdaBoost, 1995) is assigning a new weight on each data record for training the next weak learner in the series (uses all records but with weights).
- AdaBoost increases the weights on misclassified records so the next iteration can pay more attention to them. Popular AdaBoost algorithms: GentleBoost, LogitBoost, BrownBoost, DiscreteAB, KLBoost, RealAB…
- It's often called **gradient boosting** because one descends down decreasing model error in a greedy (local gradient) way.
- Stochastic gradient boosting uses a random subset of the data at each iteration.
- Gradient boosting is currently one of the most popular ML methods.

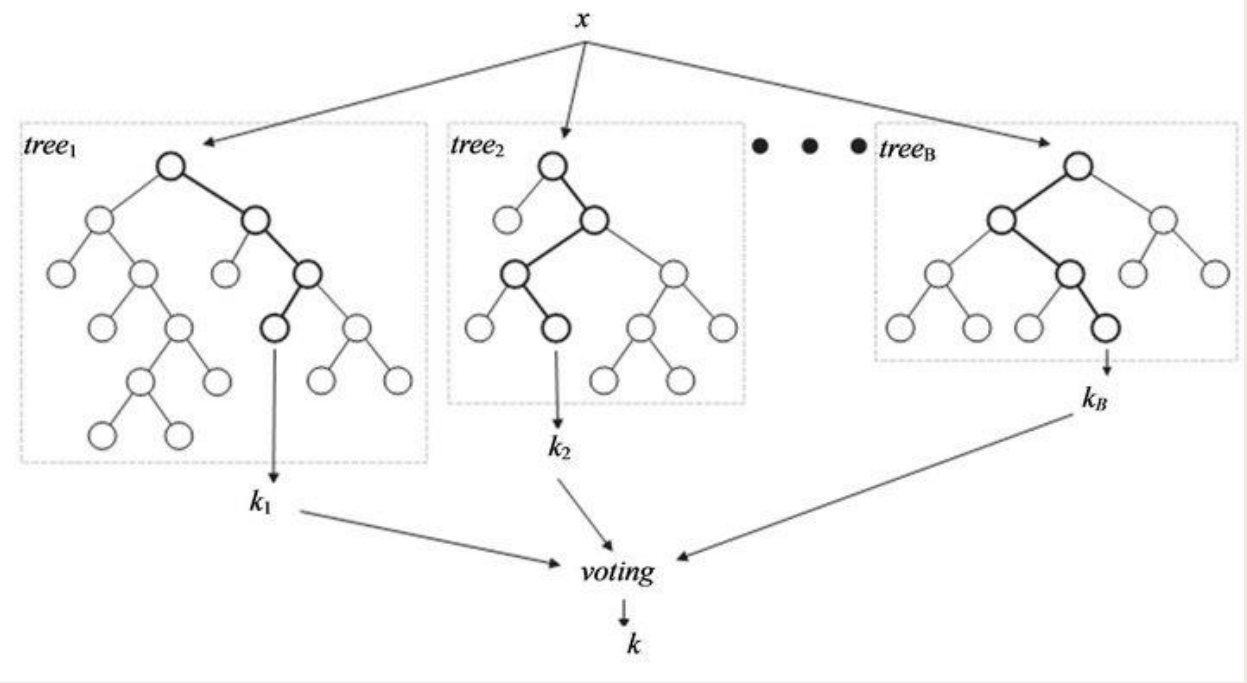
Bootstrap and Bagging

- Bootstraping - Build many models, each uses only a sample of the records, with replacement, so there is overlap between the many data samples.

- Bagging – Bootstrap Aggregation. The final model is a simple combination of all the sampled models, typically the average over the sample models.

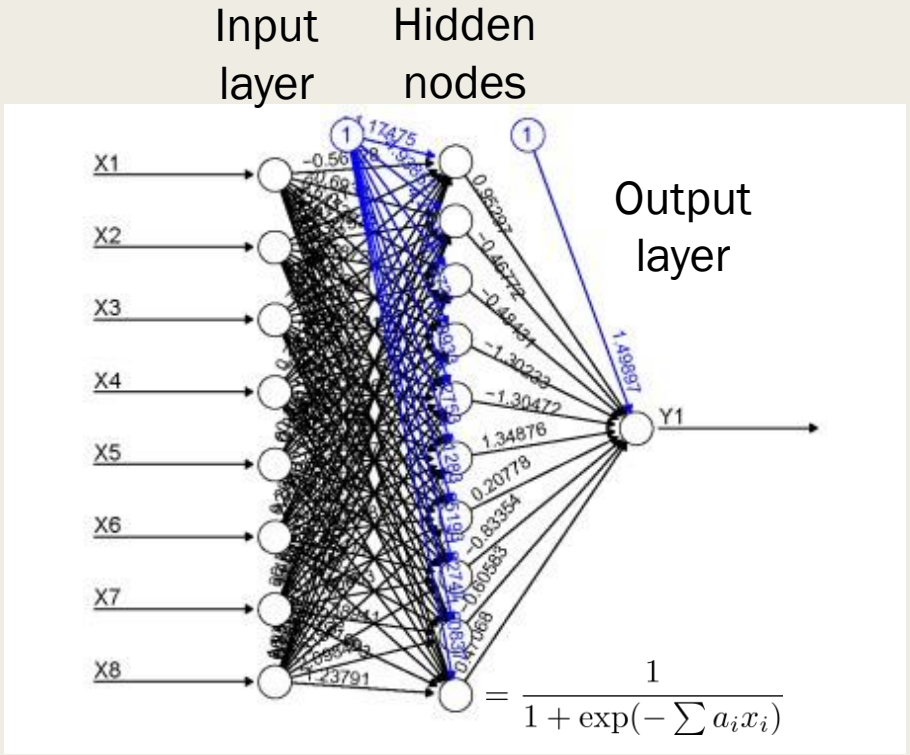
Random Forests

- Build many trees, each uses only a randomly-chosen subset of *features*
- Different from bootstrap, which uses a random subset of *records*
- Combine all the results by averaging or voting



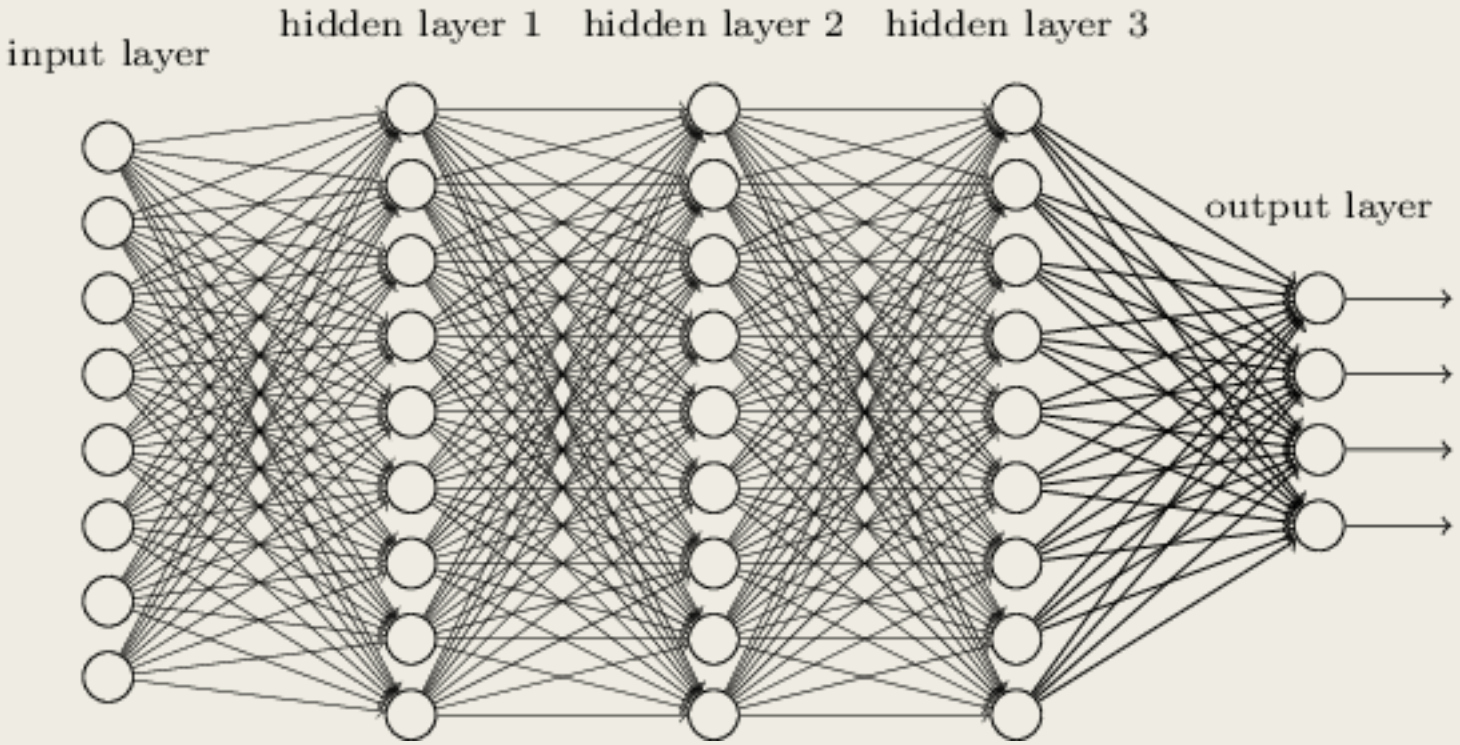
Neural Net

- A typical neural net consists of an input layer, some number of hidden layers and an output layer.
- All the independent variables (x's) are the input layer.
- The dependent variable y is the output layer.
- Each node in the hidden layer receives weighted signals from all the nodes in the previous layer and does a nonlinear transform on this linear combination of signals.
- The nonlinear transform/activation function can be a logistic function (hyperbolic tangent, sigmoid), or something else.
- The weights are trained by backpropagating the error, record by record.
- Data is shown to the neural net, record by record, and the weights are slightly adjusted for each record.
- The entire data set is passed through many times, each complete pass is called a training epoch.



Deep Learning Net

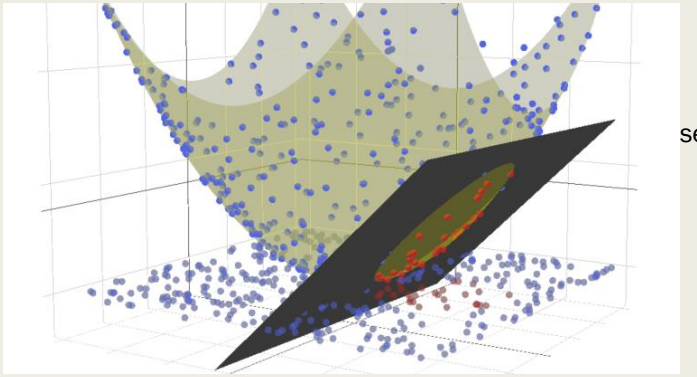
Deep Learning is a neural net architecture with more than one layer (hence “deep”). It’s not really new but wasn’t very practical until hardware and algorithms improved sufficiently.



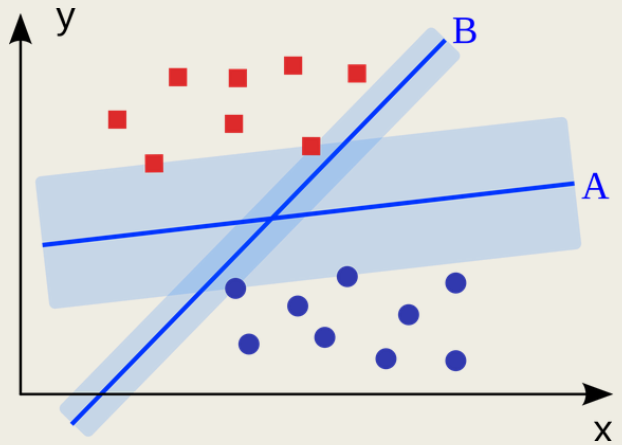
Convolutional Neural Nets (CNN) are Deep NNs with loosely connected first layers followed by fully connected layers. It is designed to work well with images.

Support Vector Machine (SVM)

- Use a linear classifier separator, but with the data projected to a higher dimension.
- Since all we care about is distance we can use a “kernel trick” to construct a distance measure in an **abstract** higher dimension.
- Also use the concept of “margin” to find a more robust linear separator location.
- The separator is completely defined by the location of the data points on the boundary, which are called the “support vectors.”



Data is separable in a higher dimension



Margin optimization gives better separation

Each ML method Has An Objective Function and Learning Rules

adjustable parameters

Objective function is typically a Loss (error) plus Regularization: $O(\mathbf{a}) = L(\mathbf{a}) + R(\mathbf{a})$

Error Regularization

Loss/error term $L = \Sigma_i l(y_i, \hat{y}_i)$ can be

- MSE: $l(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$
- Logistic error: $l(y_i, \hat{y}_i) = y_i \ln(1 + e^{-\hat{y}_i}) + (1 - y_i) \ln(1 + e^{\hat{y}_i})$

The regularization term minimizes model complexity by minimizing the parameters **a**.

Can use a variety of regularization forms:

- L1 norm (Lasso): $R(\mathbf{a}) = \alpha \|\mathbf{a}\|$
- L2 norm: $R(\mathbf{a}) = \alpha \|\mathbf{a}\|^2$

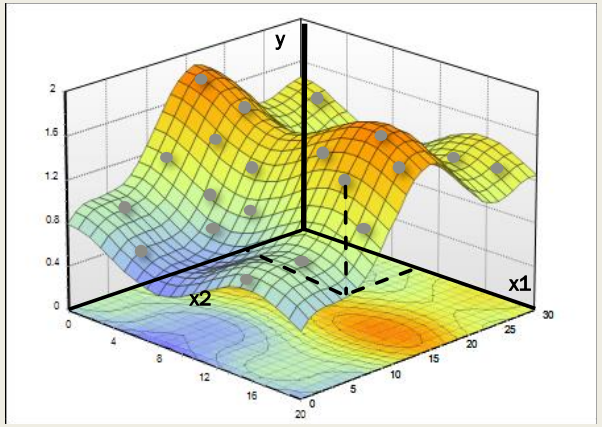
Each modeling method also has a set of training rules to adjust the parameters **a**:

$$a_i^{n+1} = a_i^n + \eta \Delta a_i, \quad \Delta a_i = -\frac{\partial O(\mathbf{a})}{\partial a_i}$$

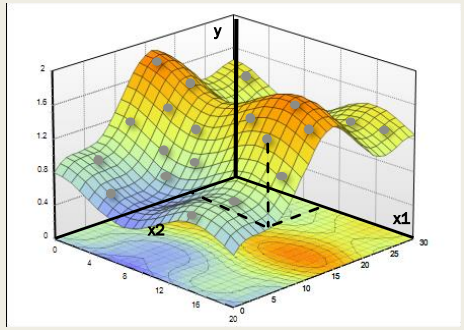
learning rate

How To Choose Which Method to Use

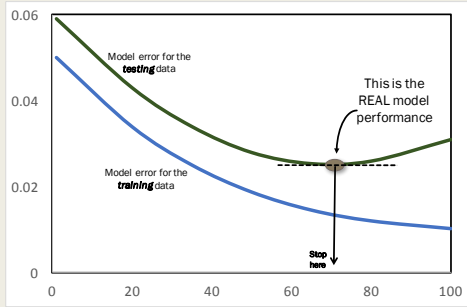
- Do you have labeled data? supervised vs. unsupervised
- Nature of output to be predicted – categorical, binary, continuous
- Dimensionality, noise, nature of raw data/features/variables
- Try several methods
- Start simple (linear/logistic regression), increase complexity
- Always strive to minimize dimensionality. In high dimensions:
    - *All points become outliers (points become closer to outside boundaries than center)*
    - *Everything looks linear (# data points required to see nonlinearity increases exponentially)*



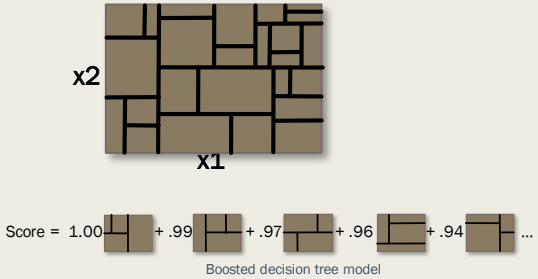
Summary

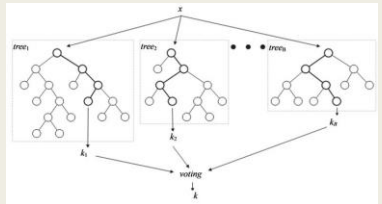


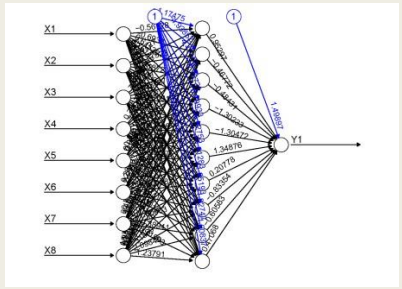
- A statistical model is a functional relationship y=f(x) found using data
- Modeling process:
  - ***Prepare data****: build expert variables, encode categoricals, scale, missing values, feature selection*
  - ***Build models****: training/testing to avoid overfitting, measures of goodness*

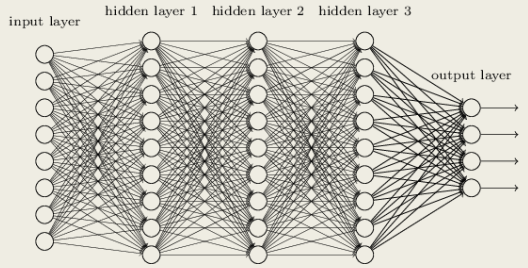


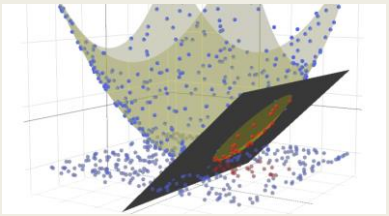
- There are a lot of nonlinear modeling methods. Understand them before you use them.









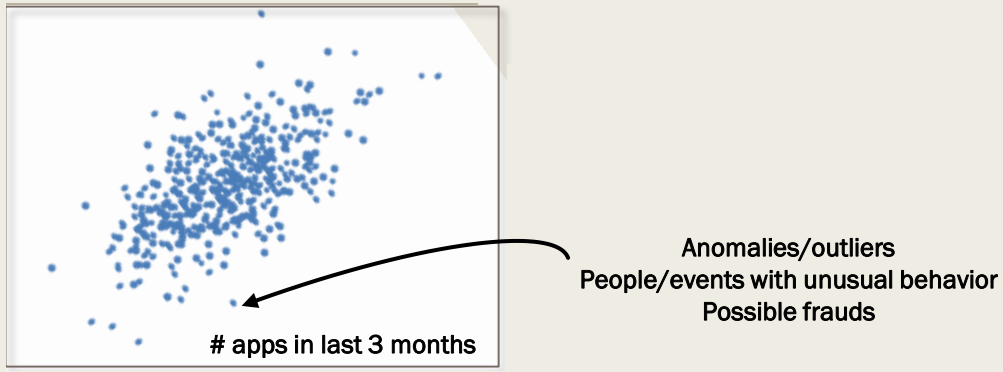


...

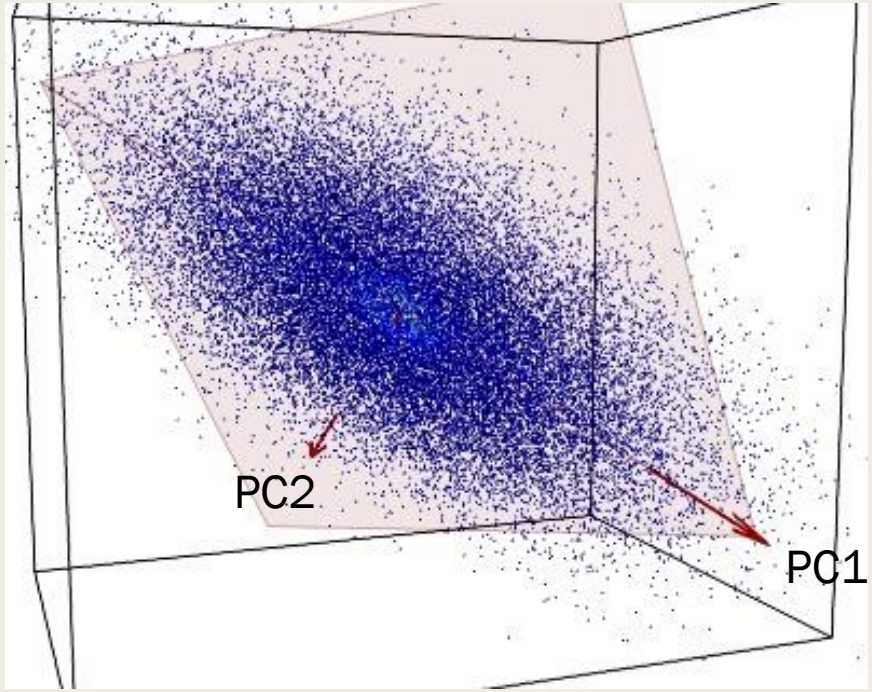
- Always seek simplicity
  - *Minimize dimensionality via feature selection. Data is always sparse in high dimensions*
  - *Minimize complexity. Start simple*

What To Do When You Don't Have Tags?

- How can you build a model to predict something when you don't have examples of it? In this situation you can build an **unsupervised model**
- Unsupervised modeling examines how the data points are distributed in the input space only (there is no output/label)
- You simply look at how the data points scatter. Can you find **clusters**? Can you see **outliers**?
    - ***Clusters:*** *groups of people or events that are similar in nature*
    - ***Outliers:*** *unusual, anomalous events – potential frauds*
- Can work well for fraud models where you're looking for anomalies
- Hardest part is figuring out what variables (features, dimensions) to examine

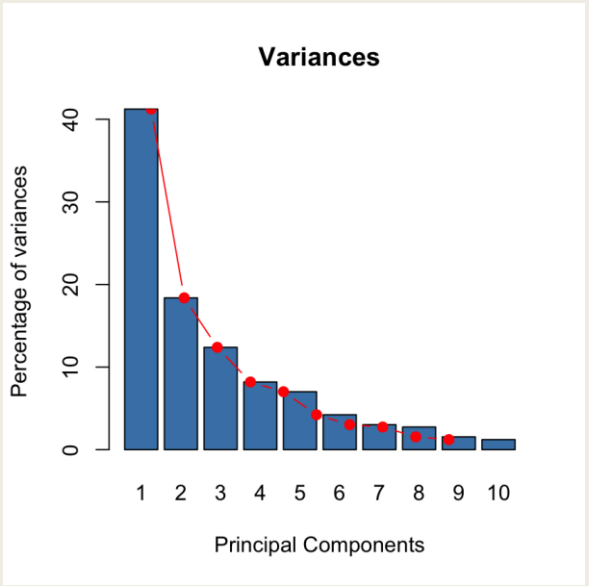


Principal Component Analysis and Regression (PCA, PCR)



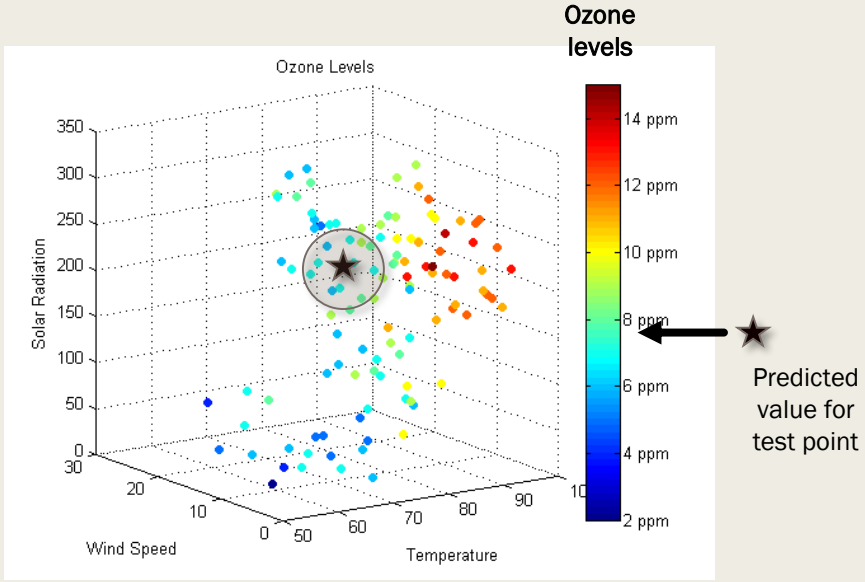
- PCA finds the dominant directions in the data
- The PC's are orthogonal, and ordered by the variance (spread) in their direction
- The magnitude of the PC's are the eigenvalues of the PC's
- The eigenvalues are proportional to the variance in that PC's direction

One can look at the decaying of the eigenvalues and select a subset of the PC's as a subspace to work in. One can then regress in this PC subspace. That's PCR.



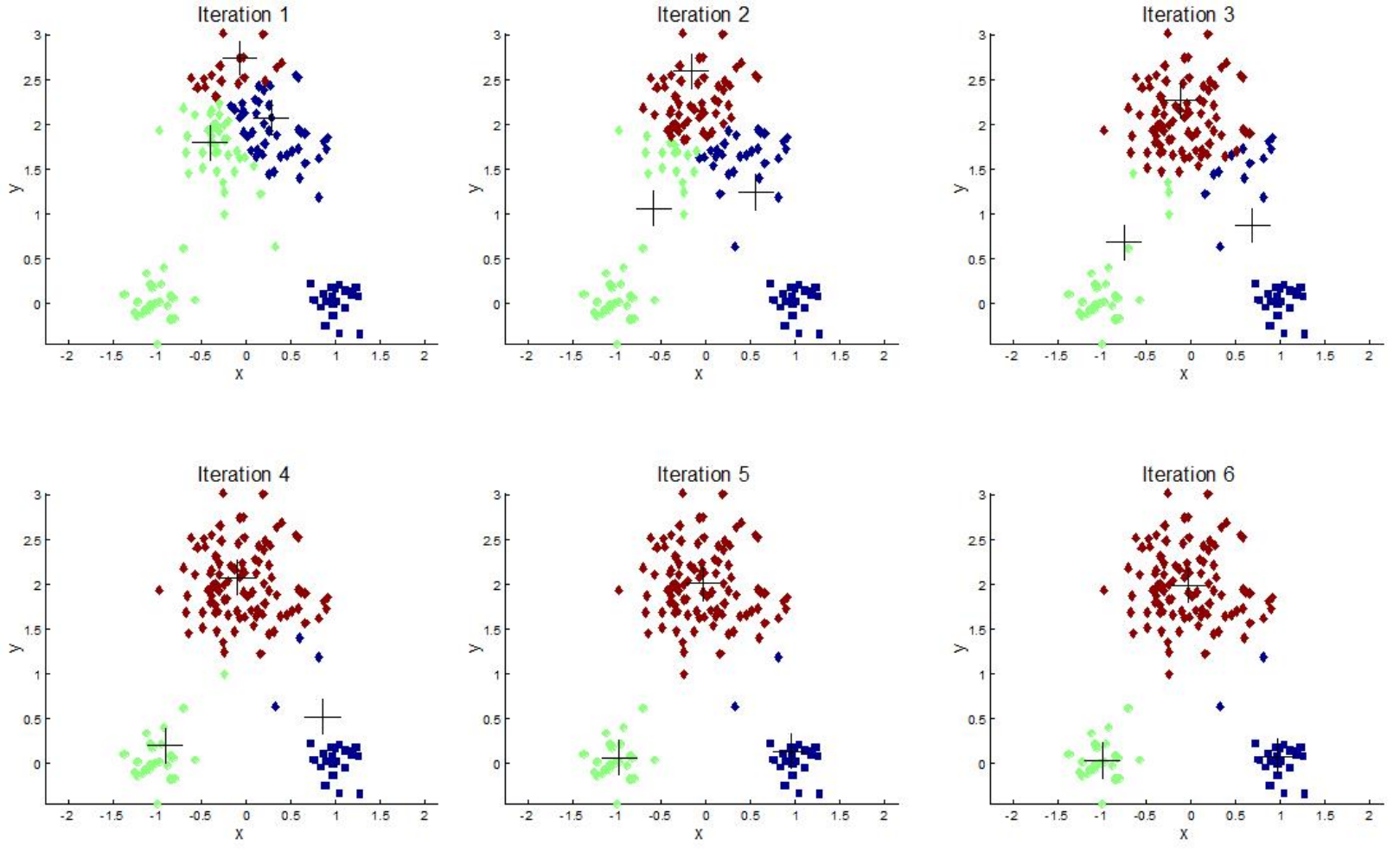
Modeling Method – k Nearest Neighbor (kNN)

- Put all data into a selected subspace (variable creation and selection)
- For any new (test) record ★, see where it falls in this space, grow a sphere around it until it contains the “k nearest neighbors”
- Return the average value for these k records as the predicted value
- Selecting the right features is important
- Requires no training, but can be slow for test/go-forward use



Modeling Method – K-Means, Fuzzy C-Means, SOM

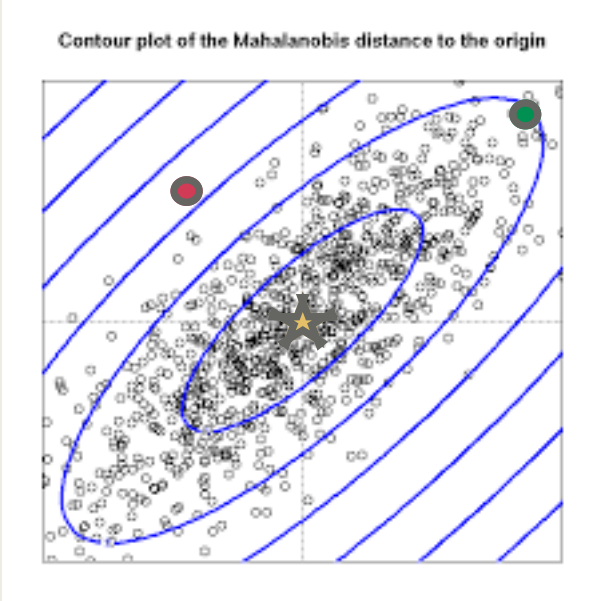
Unsupervised modeling method – finds natural groupings of data points in independent variable (x) space



Mahalanobis Distance

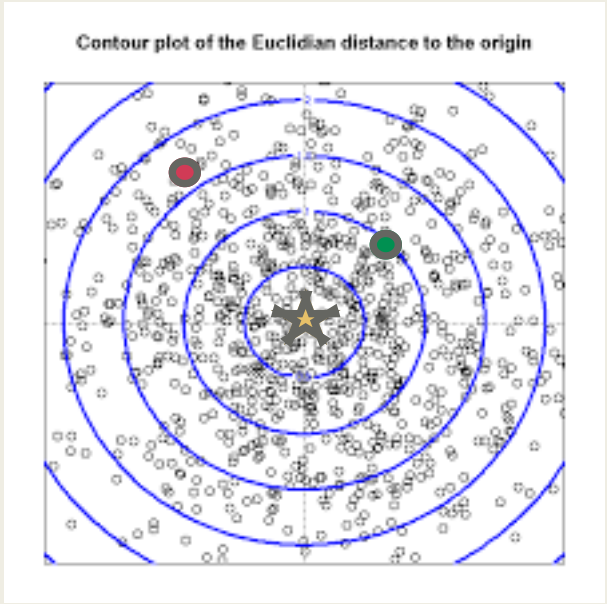
Which point is more anomalous, the red or green?

X2 – time since last event



X1 – distance from home

The Mahalnobis distance takes into account the different scales and correlations. It draws equal contours by scaling based on the correlations and different standard deviations. This is also called “whitening” the data (without rotation).

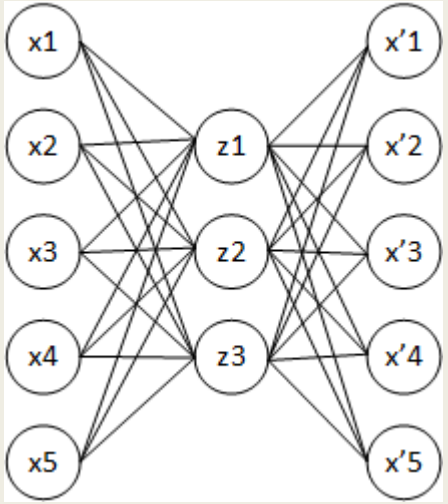


The Mahalanobis distance from the center is the same as first transforming to the Principal Component space and scaling by the eigenvectors, or first PCA then z scaling.

Another Unsupervised Model Method

- Use an Autoencoder:

Original data records



Reproduced data records

- Train the model to reproduce the original data
- The reproduction error is a measure of the record's unusualness, and is thus a fraud score

- This is nonlinear and thus more general than the previous methods (PCA, Mahalanobis), which are linear

Some Fuzzy Matching Algorithms for Individual Field Values

- Levenshtein edit distance: minimum # single character edits (insertion, deletion, substitutions)
- Damerau-Levenshtein: Levenshtein plus transpositions
- Jaro-Winkler: single adjacent character transpositions
- Hamming: % characters that are different (substitutions only)
- N grams: a sliding window of n characters. Can count how many are the same between two fields.
- Soundex: a phonetic (”sounds like”) comparison
- Stemming, root match: determine the root of the word and compare by roots
- Field specific, ad hoc

Some Fuzzy Matching Algorithms for Columns

- Linear correlation
- Divergence
- Kulback-Leibler distance (KL)
- Kolmogorov-Smirnov distance (KS)

Some Multidimensional Distance Metrics (Rows)

- Euclidean (L-2) (only useful after proper scaling)
- Mahalanobis: z scaling then Euclidian
- Minkowski (L-n)
- Manhattan: distance (L-1) traversed along axes
- Chebyshev (L-infinity): max along any axis

Glossary 1

Model – A representation of something. Models can be physical (e.g., a model airplane, a person wearing new clothes) or mathematical. They can be dynamic (models of processes) or static (no time evolution). Mathematical models could be based on first principles (e.g., solving differential equations), expert knowledge (e.g., *a priori* rule systems), or based on data (statistical models). Statistical models are the ones built in the discipline called data science, machine learning, etc.

Input, x, variable, dimension, feature, independent variables – The numbers that are the predictors or inputs to the functional form of the model y = f(x).

Output, tag, label, y, target, dependent variable – The quantity or category that you're trying to predict with a model. It could be a continuous value (regression) or a categorical value (classification). Many times for classification the target/output is binary: yes/no, 0/1, good/bad.

Supervised modeling – Frequently in modeling one has an identified output that one is trying to fit/predict for each record. The output or dependent variable is the target of the model, and we try to learn the functional relationship between the inputs and this output. This is called supervised learning since the learning algorithm is supervised during training by constantly looking at the error between the predicted and actual outputs.

Unsupervised modeling – Here we don't have an output or dependent variable, all we are given is a set of independent variables. There are several things we can do with independent variables only, for example we can

- Look for interrelationships between the variables, either linear (e.g. PCA) or nonlinear
- Look for macro structure in the given data in input/independent variable space alone. Methods include PCA and clustering.
- Look for outliers or anomalies in the records, looking only at the inputs/independent variables. This is frequently a task in building fraud models in situations when you have no labeled data, that is, no records that have already been determined to be fraud. Generally you look for what is normal in the data and then look for outliers to this typical data.

Glossary 2

Data science, machine learning, statistical modeling, data mining, predictive analytics, data analytics – Roughly the same meaning but many people ascribe some subtle differences between these terms. They all relate to the process of building statistical models from sets of data.

Variable creation, feature engineering, variable encoding – the process of constructing thoughtful inputs to a model. Typically one examines the fields in the raw data, does analysis, cleaning, standardization, and then transformations and combinations of these fields to create special variables that are candidate inputs to models. Examples of this process are encoding of categorical variables, risk tables, z-scaling, other normalizations and outlier suppression, nonlinear transformations such as taking the log or binning, construction of ratios or products of fields.

Feature selection, variable reduction, dimensionality reduction – The process of reducing the number of inputs to a model by considering which inputs are the most important to the model. Most modeling methods degrade when presented with more inputs than is needed for a robust, stable model.

Model validation – Typically one separates the data into multiple sets to ensure that the model is robust. In good modeling practices one reserves a set of data that is never used during training and testing that is used for evaluation of the model on data that it has never seen before. Frequently we reserve this holdout sample from a time that was not used during training and testing, and that is called out of time validation.

Boosting – an iterative procedure to create a series of weak models, where the final model is then a linear combination of this series of weak models. Generally the next model in the series is trained on a weighted data set where the records with the largest error so far are more heavily weighted.

Bagging – the term comes from "**b**ootstrap **agg**regation." Technique to improve model stability and accuracy. It combines/aggregates the outcomes of many models, each having been built via bootstrap sampling.

Glossary 3

Cross validation – The process of splitting data into many separate bunches. We train on all but one of the bunches, test on the remaining bunch, and then continue by designating a different bunch as the testing data set and training on all the others. We get an ensemble of model performance measures which we can average. Cross validation uses all records once per iteration whereas bootstrap will use records a random # of times.

Bootstrap – A method of selecting training data sets that are a subset of the data through random sampling of the records many times, always with replacement. Provides estimates for the variance of standard statistical measures that would otherwise be point estimates without bootstrap.

SOM and K Means - The difference is that in SOM the distance of each input from all of the reference vectors instead of just the closest one is taken into account, weighted by the neighborhood kernel *h*. Thus, the SOM functions as a conventional clustering algorithm if the width of the neighborhood kernel is zero.

Random Forests – Using an ensemble of trees and averaging the result across all the trees. Each tree is built from a random subset of features from the entire feature set. It uses bagging to select the features that are used for each tree.

Reject inference – The process of inferring the labels on records that were rejected at some point in the process, where the actual outcome isn't known. This is done so that the model is hardened, or made more robust, to this larger population that it will see when implemented (the entire TTD, Through The Door, population).

Goods, Bads – When we are doing a 2-class classification problem (as opposed to a regression where the output is continuous) we frequently refer to one class as Goods and the other as Bads. For fraud problems the Bads are the records labeled as fraud and the Goods are the records labeled as non fraud.

False Positives – (FP) Goods that have been incorrectly classified as Bads.

True Positives – (TP) Bads correctly classified as Bad.

True Negatives – (TN) Goods correctly classified as Good.

False Negative – (FN) Bads that have been incorrectly classified as Goods.

False Positive Rate is FPR = #FP / #examined = #FP / (#FP + #TP).

False Positive Ratio = #FP / #TP.