ECEN 5053-002

Developing the Industrial Internet of Things

Week 11 - Lecture

Big Data Analytics

Dave Sluiter - Spring 2018


BIG DATA
Volume
Useful
Metadata
AMP
Structured
data
Petabyte
analysis
content
People driven
Decision making
Zettabyte
Framework
Smart content
database
Concept

Preface

- Previous material looked at several machine learning algorithms
- We gained some insight into how these algorithms work
- Now we're going to look at how to apply machine learning algorithms to uncover hidden insights in a data set

Material

- Big Data, Predictive Analytics
- Getting started
  - Testing, validation
  - Bias, variance
  - Learning curves
  - Cross-validation
- Learning from Big Data
  - Processing your data
  - Visualizing your data
  - Predicting the future
- Examples

Learning Outcomes

- Understand what Big Data is and why we want to look at it
- Be able to describe the testing and validation process
- Understand how bias and variance can effect your results
- Be able to describe the cross-validation process
- Understand the importance of properly preparing your data
  - Be able to describe what “smart data” is, and what characteristics “good data” possess
- Learn ways to visualize your data

Introduction

Big Data Analytics

- **Big data** is a term for data sets that are so large or complex that traditional data processing application software is inadequate to deal with them. Challenges include capture, storage, analysis, data curation, search, sharing, transfer, visualization, querying, updating and information privacy
- The study of how to analyze huge amounts of data using machine learning algorithms
  - **This is the aspect we will look at**

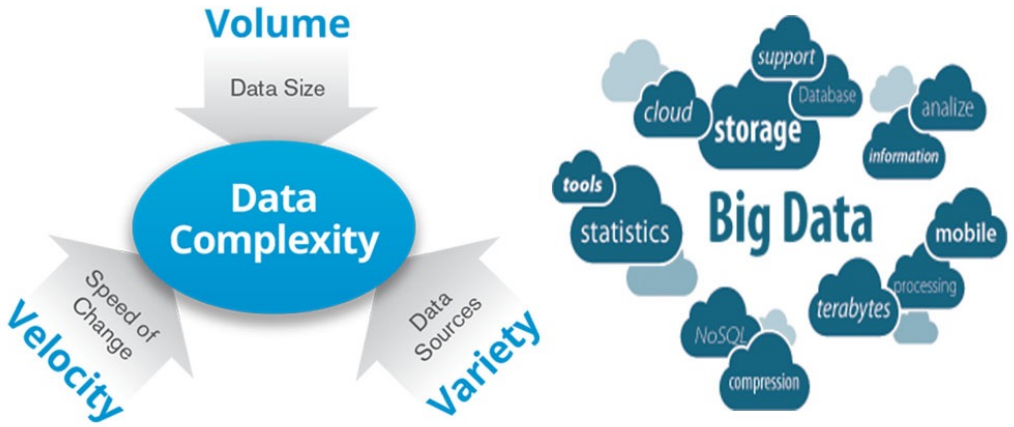
Source: https://en.wikipedia.org/wiki/Big_data

Importance of Big Data Analytics

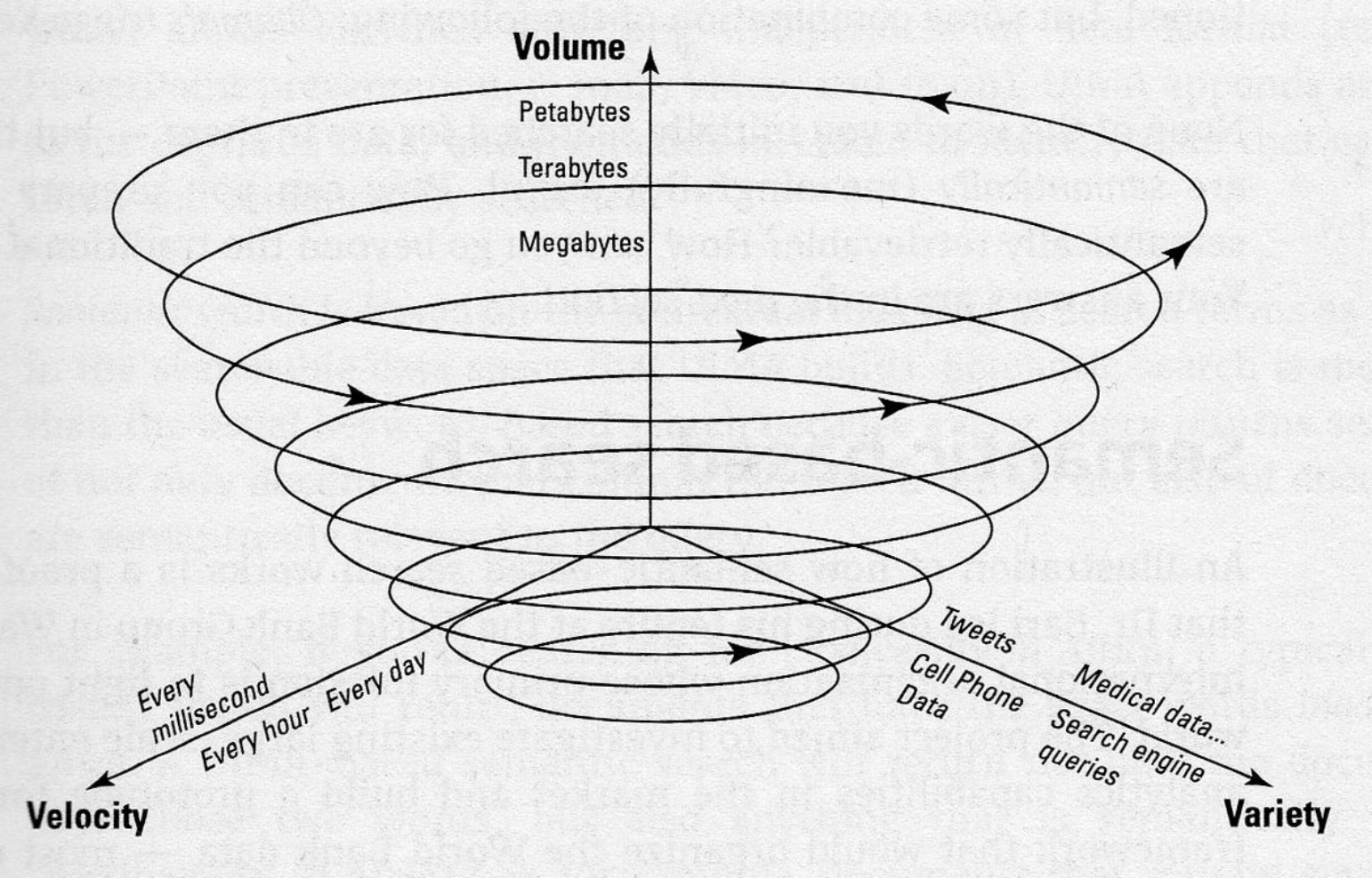
- Data is the new “currency”
  - Remember the notion of: “***Be the most important provider of information***” ?
- Trade-off:
  - Speed to an insight/decision
  - Depth of insight

Characteristics of Big Data

- **Volume**
  - How much?
- **Velocity**
  - How fast is it produced?
- **Variety**
  - How many types of data?
  - Structured; DB entry, or embedded system message
  - Unstructured; text, video, audio
- **Value**
  - Save and analyze all data? or just the important data?
- **Veracity**
  - How accurate is the data? Missing data (sensor off-line), errant data, old data?
- **Visibility**
  - Access from disparate geographic locations



Characteristics of Big Data



Source: "*Predictive Analytics for Dummies*", Anasse Bari, Ph.D.

Size of Big Data

- My opinion, based on what I see happening:
  - Data sets >= 100 TB
- According to the guru99 article below, Facebook ingests 500+ TB per day
- I heard a talk recently, the speaker indicated 1 self-driving car generates 30-40 PB per day

Source: https://www.guru99.com/what-is-big-data.html

See also: https://www.sas.com/en_us/insights/big-data/what-is-big-data.html

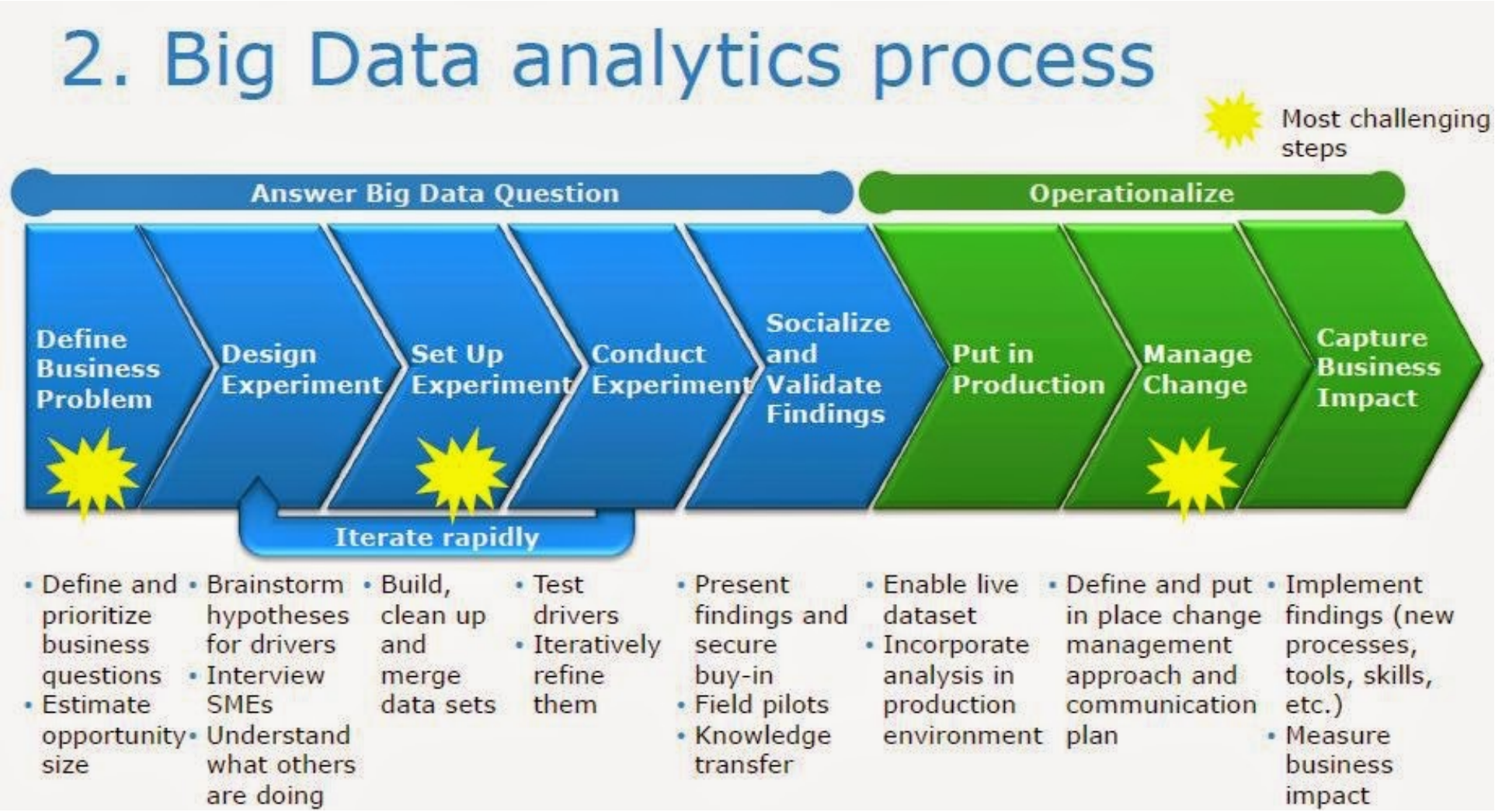
Why look at the data?

Source: https://en.wikipedia.org/wiki/Big_data

Dependance

- **Big Data Analytics** (Predictive Analytics) and **Machine Learning** are intimately tied together

Google's Process



From google analytics

Predictive Analytics

- Mathematicians and statisticians have been around for decades
- So why are we seeing this explosion (increase) in analytics?
- Data
- Lots of data
- Only recently (last ~5 years) have we been able to to effectively “mine” this data
- Processing power + data storage increased “exponentially” while getting cheaper + refined machine learning algorithms

Predictive Analytics

- Combines:
  - **Statistics**
  - **Data mining**
  - **Machine learning**
- Goal:
  - **Create a mathematical model that can make useful business predictions. Embedded systems can:**
    - **Collect and analyze data**
    - **Allow an embedded system to make predictions, and possibly make autonomous decisions**
- 2 major ways to implement
  - Data-driven, based only on past collected data (learning in the lab)
  - User-driven, based on in the field data (learning and predicting in the field)

Statistics

- Statistics are numbers that result from measuring empirical data over time
- Infers relationships within data
- Can provide a basis for us to infer hypotheses
- The discipline is both
  - **Descriptive**: Min, max, mean, median, variance
  - **Inferential**: Allows us to make predictions about the whole dataset from a smaller sample, or make predictions as new data arrives

Data Mining

- Analyzing data and making sense of it
- Is mainly concerned with classifying and clustering the data
- Data can be analyzed with and without machine learning
- Often the first step in analytics is attempting to **uncover hidden insights**
- ***Note: Business Intelligence*** *(different from Data Mining) is about building a model that answers specific business questions: "What is my expected volume at a given average selling price (ASP)? Probably won't uncover hidden insights.*

Machine Learning

- Analyzing data and making sense of it
- **Uncover hidden insights**
- Using nontraditional programming approaches
  - Linear regression, SVM, K-means etc. vs.
  - Purpose programmed algorithms

Data

- Static
  - A DB, a folder of files, the Iris flower dataset
- Streamed
  - Constantly changing: Facebook updates, Twitter feeds, stock prices while the market is open
  - In IIoT: Messages containing temperature, velocity, position, pressure, voltage, current, vibration etc.

Streamed Data Approaches

- Examine the newest data points and make decisions about the state of the model and its next moves
- Evaluate the entire dataset (or subset) each time new data points arrive

The Algorithms

- The 5 Tribes revisited
  - Symbolic reasoning
  - Connections modeled on the neurons
  - Evolutionary Algorithms
  - **Bayesean inference**
  - Learn by analogy

"*Machine Learning for Dummies*" page 30

Coding a Solution

- Traditional programing
  - Inputs, 3 and 2
  - Function, add
  - Output 5, code it up and test it
- Machine learning
  - Inputs
  - Outputs (and sometimes we don't know the output)
  - We may not know the function, try many functions, try to discover a function that produces results we are interested in
  - Train the function
  - Visualize and test it
- The "**secret**" to machine learning is **generalization** and **discrimination**

Generalization and Discrimination

- Learning algorithms rely on 3 components
  - Representation
    - Part of the representation is to discover which features to use for the learning process
  - Evaluation
    - An evaluation function scores a model because the model doesn't know good results vs. bad results
  - Optimization
    - Training, tuning and making model selections

"*Machine Learning for Dummies*" page 33

Analysis Tools Beyond Python (and R)

- Statistical Analysis Tools
  - SAS
  - https://www.sas.com/en_us/home.html
  - Stata
  - http://www.stata.com
  - SPSS
  - https://www.ibm.com/analytics/us/en/technology/spss/

Frameworks

- A framework provides a set API's to use for programming

Framework Examples

- Apache Singa
  - Natural language processing
- Hadoop MapReduce
  - Looks for structure in a dataset
- Apache Spark MLib
  - Fast machine learning library
  - Used by eBay, & IBM
- Caffe
  - Deep learning image and general learning
  - Process 600 million images/day with a single GPU
- Google TensorFlow
  - Deep learning algorithms that rely on data flow graphs
- Oxdata H2O
  - Geared to address business needs
- Nervana Neon
  - Takes advantage of custom hardware, CPU's, GPU's and Nervana hardware
- Shogun
  - Library that can interface with Python, R, C++

Getting Started

Representations

- Represent the solution to a problem (the **output**) with a number
  - Real
  - Binary
  - Probability between 0 and 1
  - Label: 0, 1, 2 (like the Iris flower dataset)
- **Inputs** are called *features*
- 2 types of features
  - **Quantitative**
    - Defines values as a number (real, binary, probability)
  - **Qualitative**
    - Are labels or symbols that convey information in a non-numeric way, examples: blue, big, fast
    - You must **assign** numeric values to qualitative labels

The Baysean Way

- Statistics “expects” that the future won’t differ too much from the past
- So you can base future predictions on past data by employing random sampling theory
- You can expect the distribution of your present samples to closely resemble the distribution of future samples

“*Machine Learning for Dummies*” page 182

Testing

- Ensuring good results requires testing
- Your algorithm learns from *in-sample* data (training data)
- We test your algorithm(s) with *out-of-sample* data, data you didn't have at learning time

Validating

- Taking measurements of your model's performance
  - What is the models output error?
  - How does the model perform with new data that it has never seen before?
  - Are we getting predictions we expect, and are useful?

Issues with Data

Sample Bias

- A *bias* is an **offset** in a data set (also known as selection bias)
- Famous example, 1936 US Presidential election, Alfred Landon vs. Franklin D Roosevelt.
- A respectable magazine pulled 10 million names from every telephone directory, plus magazine subscription lists and clubs, polled them and got 2.4 million responses.
- In the poll 57 % voted for Landon, yet Roosevelt won with 62%, what happened? Largest error ever for a public opinion poll.
- In 1936 having a phone, magazine subscriptions and belonging to a club meant that you were rich, so the samples were pulled only from affluent voters.
- **Point**: If all of your training data is biased, all of your results will be biased

Sample Variance

- *Variance*
  - The average of the squared differences from the mean

$$= \frac{1}{n} \sum_{i}^{n} (x_i - u)^2$$

    - u is the mean value
  - The *variance* measures how far a dataset is **spread out**
    - This is important to understand which features **may** be the important ones
    - May indicate if your model has sensitivities to variance if the model starts to act erratically
    - May also indicate some samples may be invalid

Source: http://www.statisticshowto.com/variance/

How to get out-of-sample data?

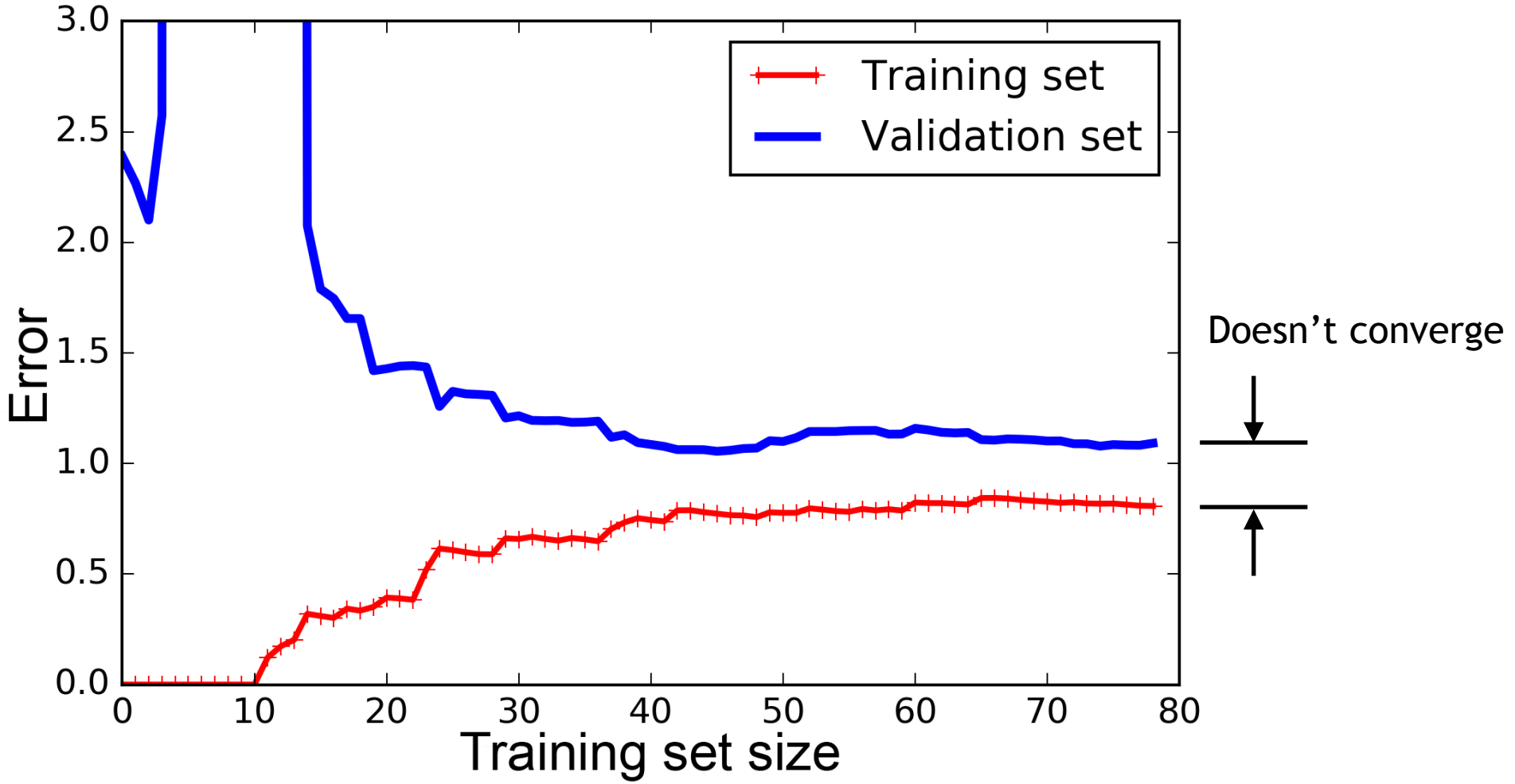
- You may have sales data now to train your algorithm from the past years, but you'd have to wait another year to get new *out-of-sample* data
- One approach is to divide your *in-sample* data into 2 portions, one used for **training** and the other for **testing**
- Example: For the sales data case, you could use a certain date as the dividing line between *in-sample* and *out-of-sample* data - but there is a better way…

Learning Curves

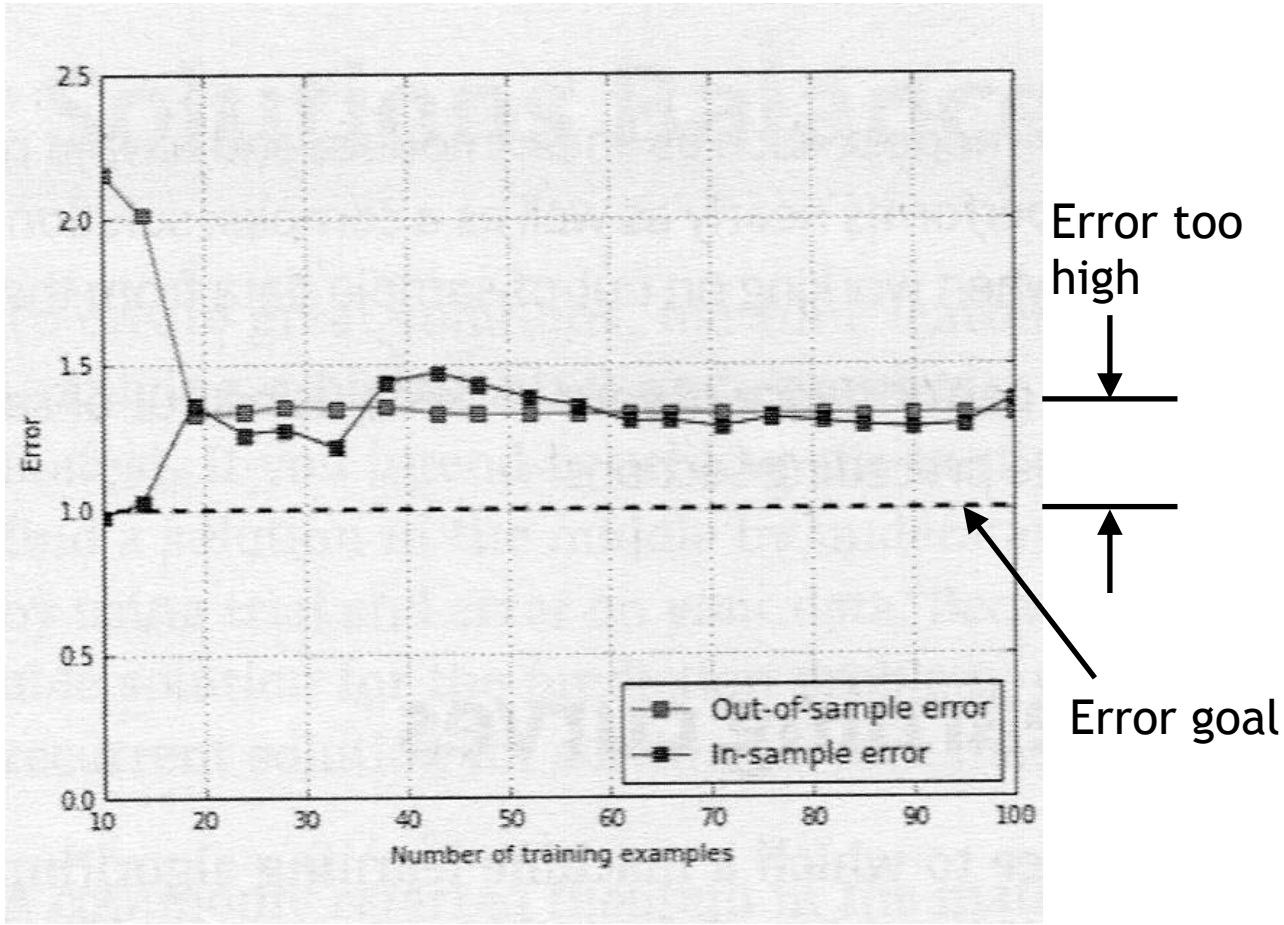
- Plot the performance (error) of a learning algorithm with respect to the quantity of data they use for training
- Helps visualize whether an algorithm is suffering from bias or variance problems
- Steps
  - Divide your data set into *in-sample* and *out-of-sample* sets, (70/30, or 90/10 for example)
  - Create sub-portions of the *in-sample* and *out-of-sample* data in 10% increments
  - Train the model on each of the *in-sample* data sub-portions
  - Run the *out-of-sample* data sub-portions
  - Record and plot the error for each sub-portion, for both *in-sample* and *out-of-sample*
  - Ideally they become close to a common error value

"*Machine Learning for Dummies*" page 189

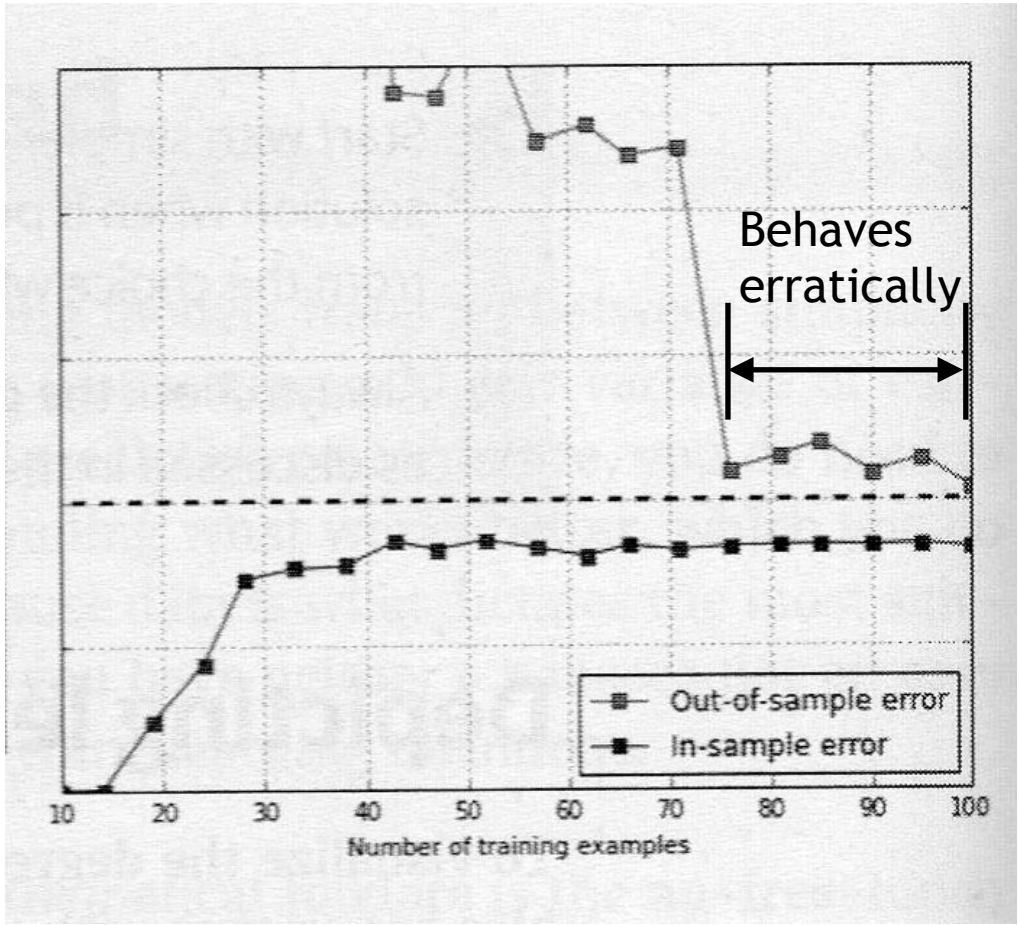
Learning Curves - Issues



Learning Curves - Issues



Bias Issues



Variance Issues

Learning Curves - Issues

- Tend to converge but doesn't converge
  - => Strong indication that your training set is too small, make it bigger
- Converges but both curves have too high of an error
  - => May be suffering from high bias issues. Adding more data won't help. Try increasing the number of features or using a more complex algorithm
- Don't converge or out-of-sample curve behaves erratically
  - => May be suffering from high variance issues. Try increasing the number of samples, reducing the number of features, adjusting some hyper-parameters of the algorithm

"*Machine Learning for Dummies*" page 190

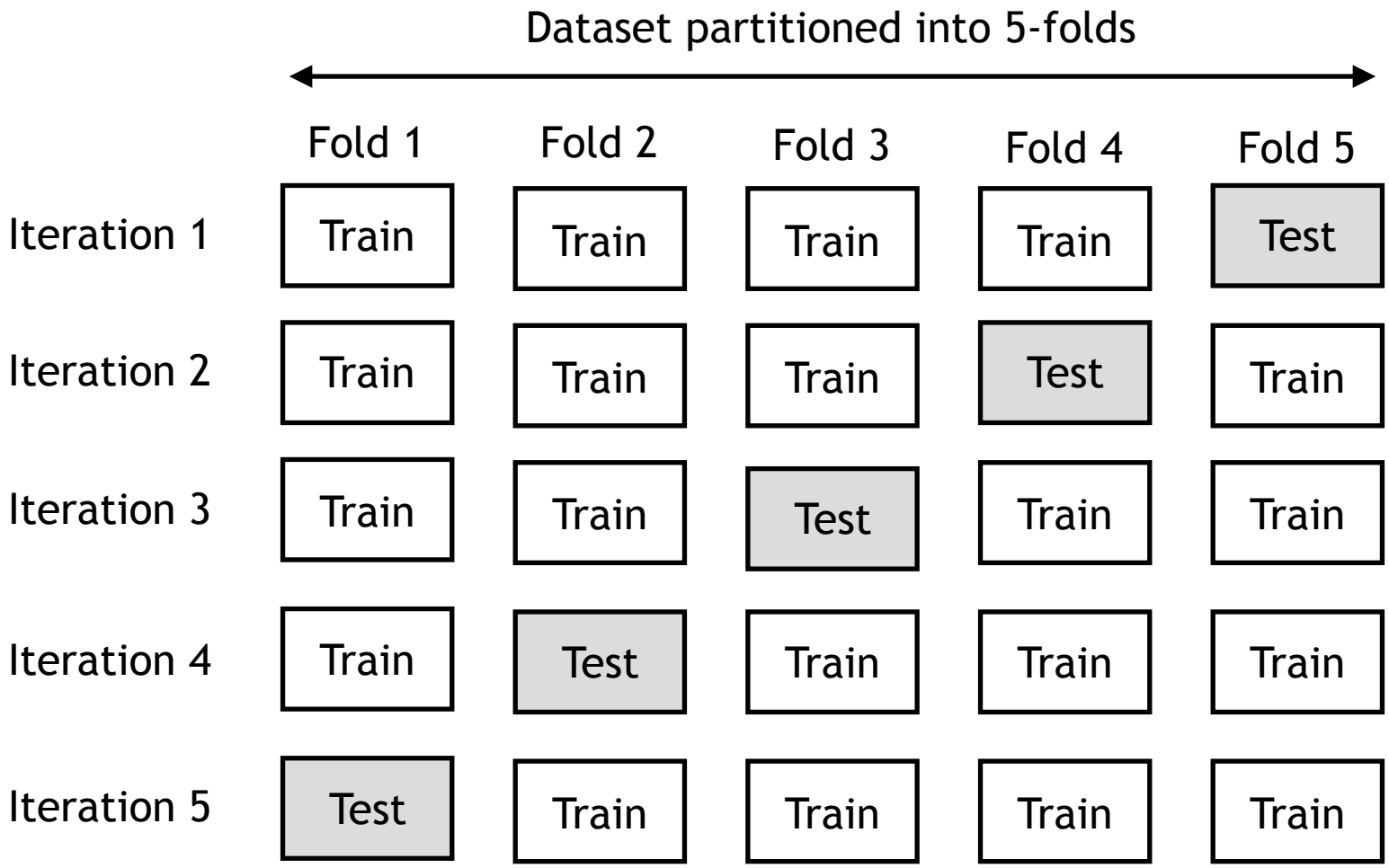
Sampling Bias

- A problem that can arise from splitting your dataset into *in-sample* and *out-of-sample* portions can introduce **bias**
  - Useful examples may be excluded from training
  - Sometimes your data set is so complex that a test set, though apparently similar, is not really similar
- Try using **cross validation**

Cross-Validation

Divide your data set into *k-folds*

Produces k error measurements, which can then be used to produce a **mean error**



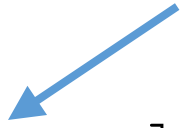
Cross-Validation Benefits

- Works well regardless of the dataset size because increasing the number of folds (k) you're effectively increasing the size of your training set
- Differences in distribution for individual folds don't matter as much
- You're testing all of your observations
- By taking the mean of the results, you can expect predictive performance
- Higher variation in the cross-validated performance informs you of extremely variegated data that the algorithm is incapable of properly identifying

R and Python Support

- Both R and Python offer functions to slice your data matrix into *train*, and *test* parts
  - See http://scikit-learn.org
    - `from sklearn import cross_validation`
    - `X_train, X_test, y_train, y_test = cross_validation.train_test_split(iris.data,`
    - `    iris.target, test_size=0.10, random_state=111)`

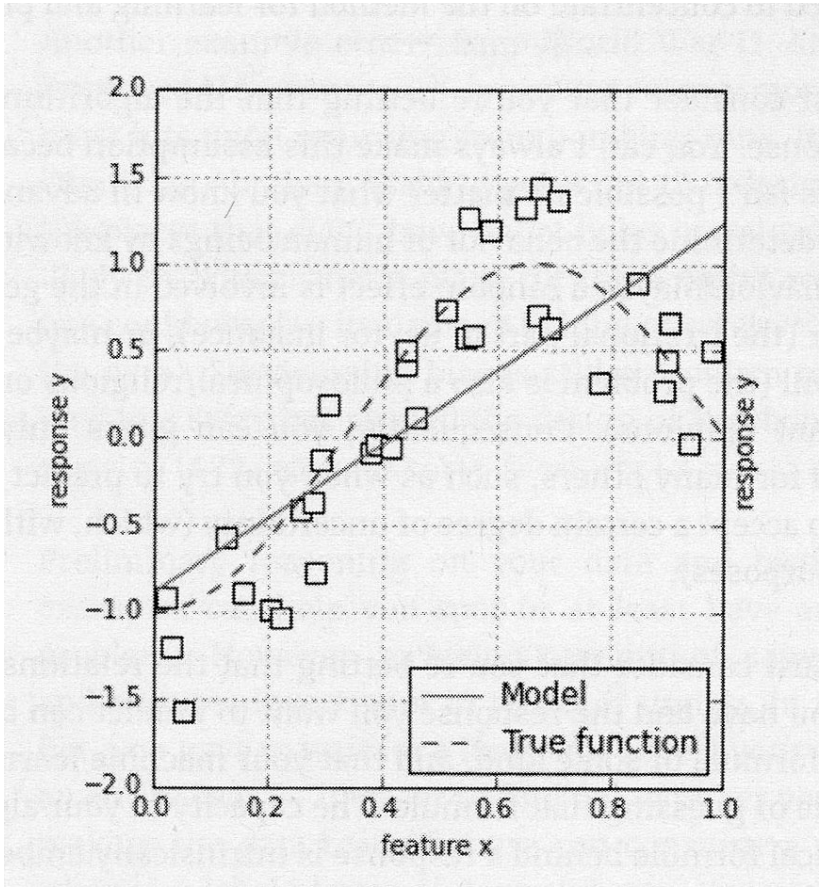
Deprecated: scikit-learn has a new method



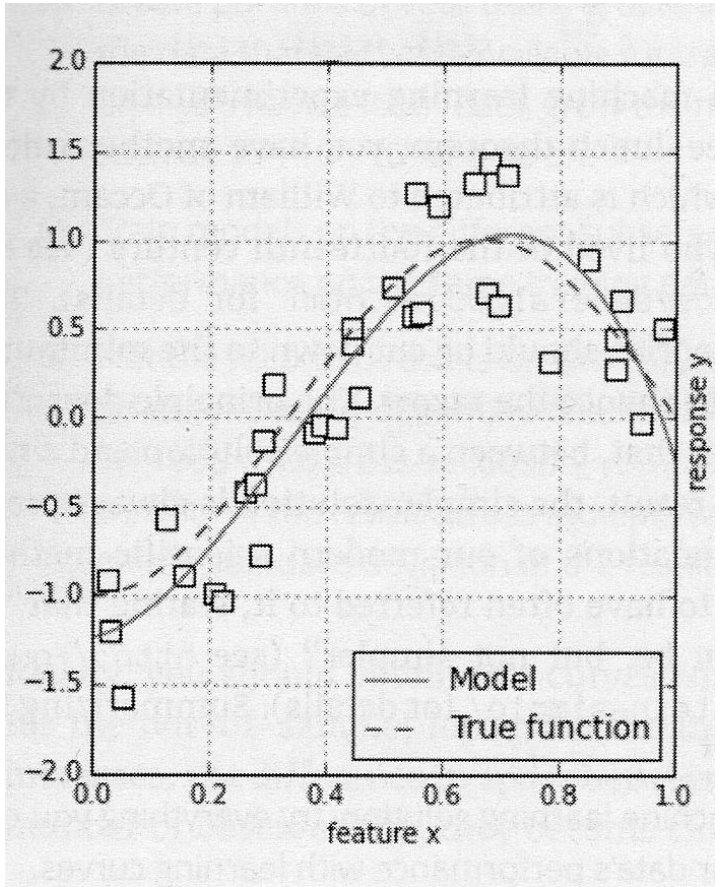
- *test* you're looking for variation between a *test set* and a *training set*
- *validation* is looking at the overall performance of the model

Model Complexity

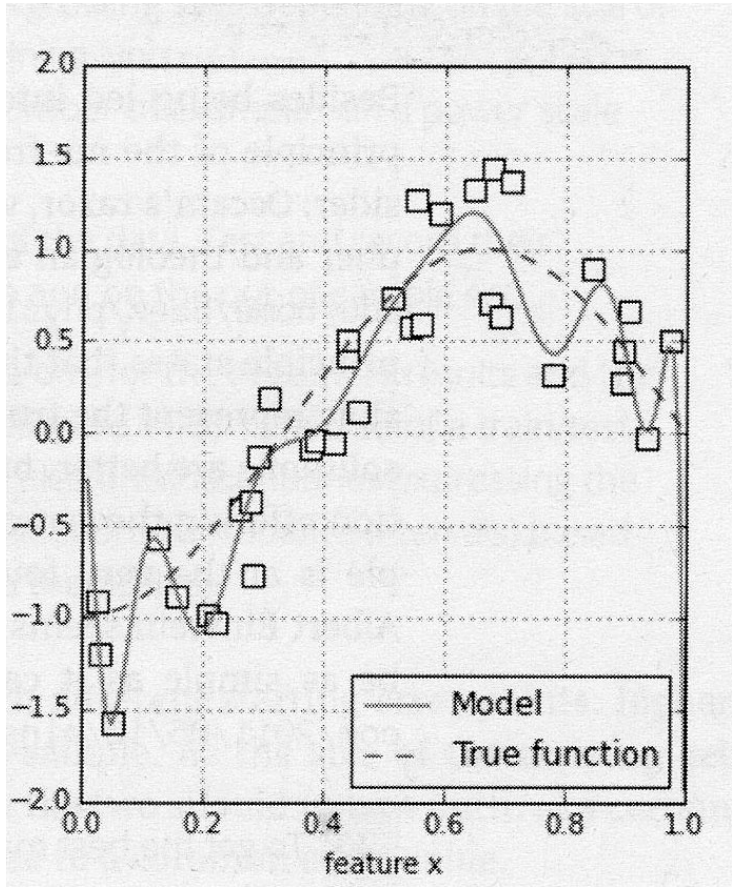
Under-fitting



Best fitting



Over-fitting



Under- and Over-fitting

- Under-fitting
  - Trying to fit a linear model to a dataset with non-linear areas
- Over-fitting
  - Many more parameters, in some cases more than the number of features
  - Results in “memorization” of the training data, implies mappings that don’t actually exist in the data

Choosing Balanced Solutions

- Good machine learning solutions live in the trade-space between
  - Simplicity, and
  - Complexity
- The data dictates what works and how well
  - Cannot rely on a single algorithm
  - Many algorithms must be tried and evaluated to find the best one for any given problem

Learning from Big Data

Learning from Big Data

- If you neglect to build a solid foundation, nothing you build upon it will be stable
- What this means for machine learning and big data analytics is:
  - **Preparing the data**
- Data preparation can typically take about 80% of the total time of a machine learning project

Processing Your Data

- Identifying the data
- Cleaning the data
- Generating derived data, if any
- Reducing the dimensionality of your data
  - Hierarchical data needs to be **flattened** to reduce any duplications
  - Principal Component Analysis (PCA, more on this below)
    - PCA studies a dataset to learn the most relevant features responsible for the highest variation in the dataset
    - Can be used for visualization only, or
    - Can be used to reduce the number of features in your data that can then feed your machine learning algorithm(s)

Processing Your Data

- Obtain meaningful data (aka ground truth), data that someone has correctly measured and labeled
- Acquire enough data for the algorithm to work correctly
  - Only after **testing** and **validation** will you be able to determine if you are suffering from *bias* or *variance* issues
- Arrange the data into a matrix
- Deal with bad data
  - missing cases
  - distorted distributions
  - redundancies
  - anomalous examples
  - noise

Processing Your Data

- Extract features (select features)
  - DB queries, PCA, manually
- Rank features. What are the most important features?
  - There are statistical and entropy based approaches to help you identify these:
    - *Gain ratio*
    - *Information gain*
    - *Chi-squared*
    - *SVM ranker*

Not covered in class

Processing Your Data - FYI

- Extraction, Transform and Load (**ETL**)
  - See: https://docs.oracle.com/cd/B19306_01/server.102/b14223/ettover.htm
  - Extraction: Collects raw data from operational systems
  - Transform: The cleaning, repairing and processing of the data (PCA perhaps)
  - Load: Places the data in a designated location for algorithms to work on (perhaps an SSD array, Hadoop or Lustre file system)

“Good Data” Characteristics

- Sufficient quantity of data for your algorithm(s)
- It represents the features we care about

Unstructured Data

- Time consuming, but necessary:
  - **structure** the unstructured data!

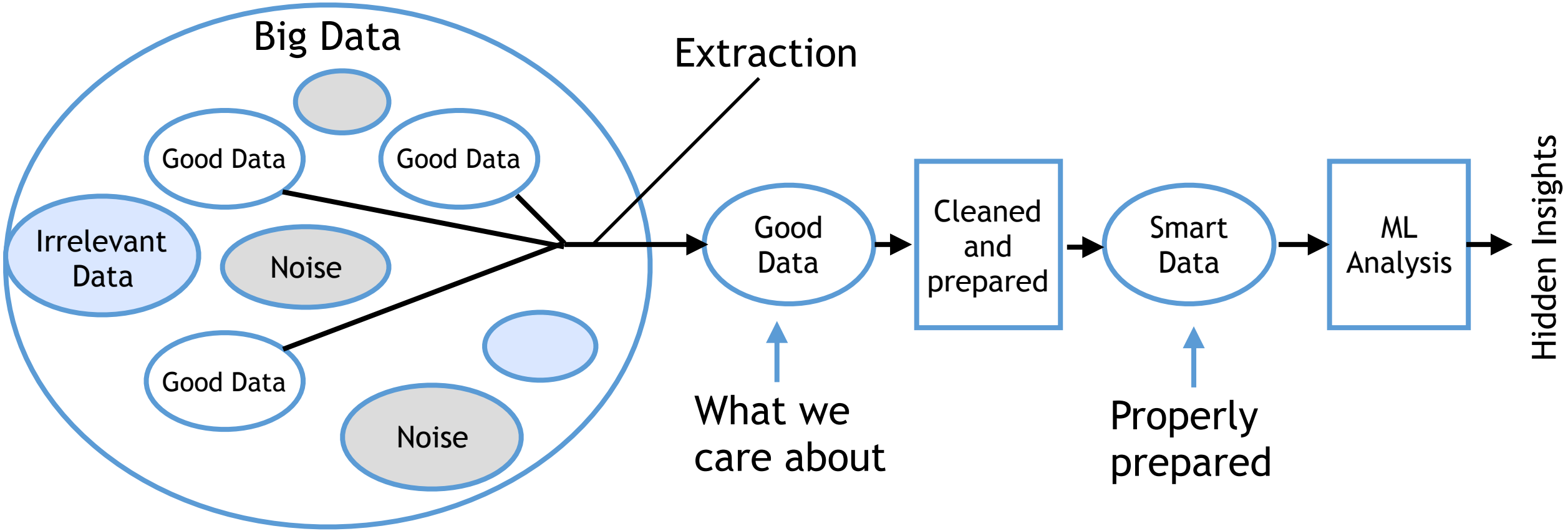
Missing Data

- Features missing at random is ideal, because you can use an average to compute a replacement value from the other feature vectors
- A high number of a specific missing feature is more difficult to replace, and requires additional analysis to synthesize a replacement. Some options to try:
  - Compute a mean or median value
  - Use 0 values, ok for regression models
  - Interpolate

Outliers

- Outliers, also called *anomalies*, have something unusual about them
- Fall into 2 basic categories
  - They really are anomalies as something went wrong with them. We call these *manifest outliers*. These require repairing or removal.
  - Novelties, a new kind of example you haven't seen before
- Use Exploratory Data Analysis (EDA), a term coined by John Tukey in his 1977 book "*Exploratory Data Analysis*".
- Rule of thumb: If you see a large difference between the **mean** and the **median** values in your data, there "***may***" be a problem.

Goal: Extracting “Smart” data



Visualizing Your Data

- Tabular
- Word clouds
- Flocking Birds
  - separation
  - alignment
  - cohesion
- Graphs/charts
- Glueviz
- Bokeh (in Anaconda)

"*Predictive Analytics*", Anasse Bari, Ph.D. page 83

PCA

- Info: https://en.wikipedia.org/wiki/Principal_component_analysis
- **Demo**: http://setosa.io/ev/principal-component-analysis/
- One side-benefit is it helps protect yourself from overfitting the model
- It helps you spot redundancy
- It helps you find out that two (or more) variables are telling you the same thing
- The output features are:
  - Uncorrelated
  - First principal component has the highest variance, second principal component has second highest variance etc
- PCA can be thought of as **feature extraction**, or **feature reduction**

PCA

- Exercise caution when reducing features
- Make sure that the performance of the model is not adversely effected
- Carefully evaluate each variable, measuring its usefulness
- PCA is not a magic bullet. It may help you, or not. You may have to try other techniques

Predicting the Future

Prognostic Health Management (PHM)

- Design, production and operation of machinery is a multi-trillion dollar global industry
- Developed over the last few decades
  - See: https://www.phmsociety.org
- Central idea is to service machines in a production line when they need it, and not according to some conservative time-based schedule

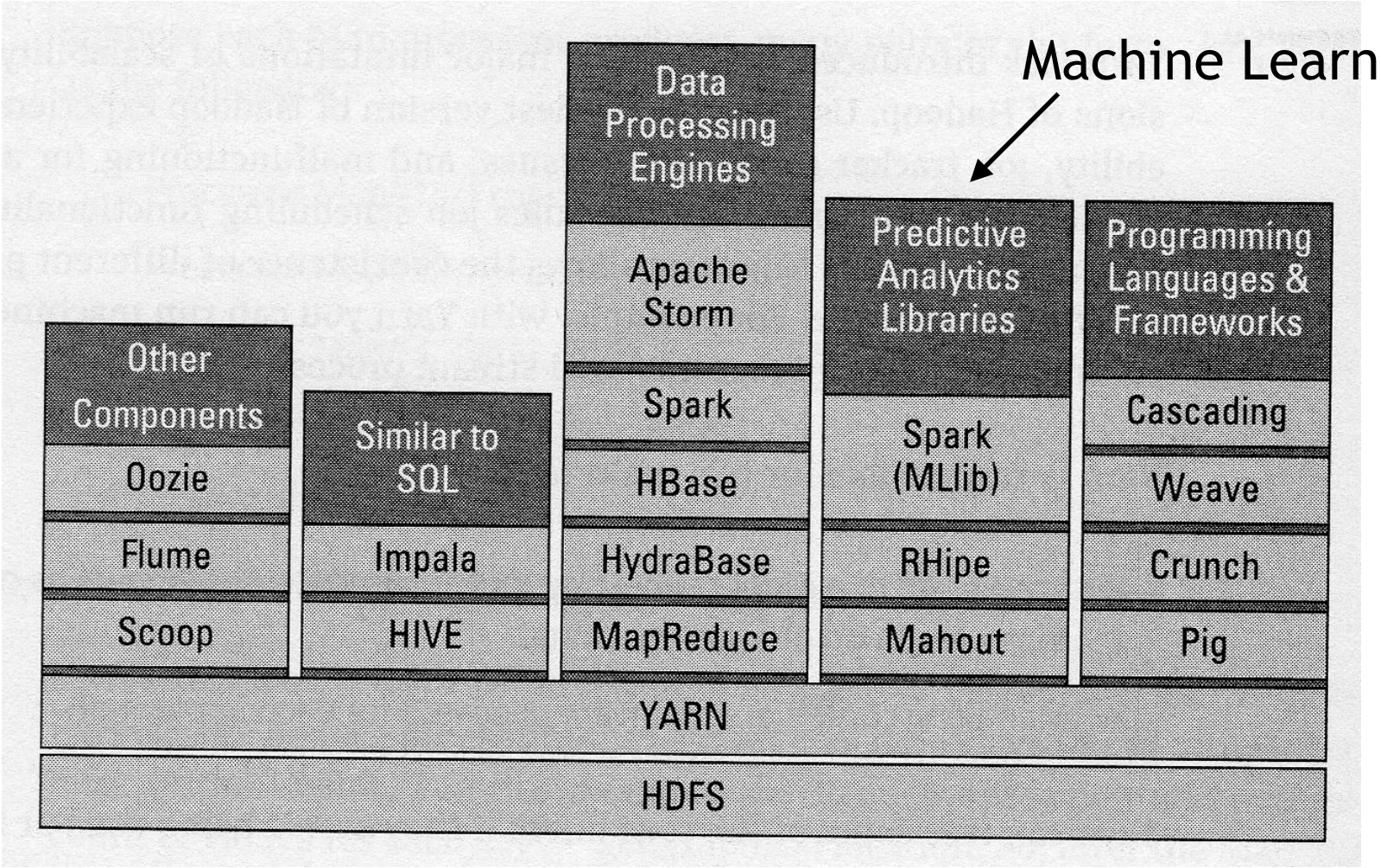
Prognostic: An advance indication or portent of a future event

PHM Example

- In 2015 GM announced a prognostic technology that aims to determine whether certain vehicle components need service/maintenance because of a possible future failure
- Data sources:
  - Manufacturers data: engineering data, stress-tests etc
  - Field data: Oil temperature, manifold pressure, vibrations, acoustic noise
- Models:
  - Linear regression
  - Non-linear regression
  - SVM

Examples

Hadoop


Machine Learning Libraries
Other Components
Oozie
Flume
Scoop
Similar to SQL
Impala
HIVE
Data Processing Engines
Apache Storm
Spark
HBase
HydraBase
MapReduce
Predictive Analytics Libraries
Spark (MLlib)
RHipe
Mahout
Programming Languages & Frameworks
Cascading
Weave
Crunch
Pig
YARN
HDFS

Hadoop

- YARN (Yet Another Resource Negotiator), relatively new
- Responsible for
  - Servicing client requests by allocating processes (*containers*) on physical machines
  - Managing containers

Hadoop

- MapReduce is really a two process steps
  - Map
  - Reduce
- Map() function (on the *master node*) divides a query request into subtasks which is then distributed to *worker nodes* that process the subtask and pass results back to the master node
- Reduce() function collects the results from all of the subtasks, combined them to produce an aggregate result as the answer to the original query request

See: http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

MapReduce Example

- We want to compute the number of social friends that Bob has, and arrange them into geographical locations
- Our DB has 2 billion people, their addresses and connections
- One MapReduce implementation might be:
  - Compute the average number of social friends
  - Arrange those friends by geography

MapReduce Example (con't)

- Suppose we have each subtask handle 1 million data records
- Each Map() subtask procedure produces multiple records of <country, count>. For example:
  - <France, 25>
  - <Germany, 67>
  - <India, 31>
- In the Reduce() phase, Hadoop assigns a task to a certain number of processors to accumulate the final results:
  - <France 25> … <Germany, 67> (sorted in ascending order)

Apache Spark

- IBM: Commenting on Apache Spark "*the most important new open source project in a decade that is being defined by data*"
- So what is it?
- The next generation data analytics processing engine that can coexist with Hadoop
- An extension to MapReduce, adds two main features:
  - Interactive queries in lieu of data explorations that could take hours using MapReduce
  - Ability to process streaming data

June 2015: https://www-03.ibm.com/press/us/en/pressrelease/47107.wss

Apache Spark (con't)

- Spark's main data abstraction is based on resilient distributed datasets (RDDs)
  - A parallelized collection of data in your program
- Spark's main components:
  - Spark Core, primary API's, task scheduling, memory management
  - Spark Streaming, enables an application to process streaming data
  - Spark SQL, queries on structured data
  - Spark MLib, machine learning library, dimension reduction
  - Spark GraphX, graph creation

A Personal Example

NFL Football Winners

- Hypothesis: *Is it possible to look at a whole year of NFL team statistics using machine learning algorithms and data analytics techniques to predict the winner of each game of the regular season with 80+% accuracy?*

What Data to Look At?

Critical statistics for the entire season for each team

	Offense					Defense								
Team	**OPts/ G**	**OYds/ G**	**OYds/ P**	**Fum**	**O3rdDown Pct**	**DPts/ G**	**DYds/ G**	**DYds/ P**	**Fum**	**DSacks**	**Wins Home**	**Wins Road**	**TotWins**	**Super Bowl**
49ers	19.3	308	4.9	29	35	30	406	5.9	25	33	1	1	2	0
Bears	17.4	356	5.9	26	38	24.9	347	5.5	13	37	3	0	3	0
Bengals	20.3	356	5.4	20	39	19.7	351	5.4	12	33	4	2	6	0
Bills	24.9	354	5.6	15	41	23.6	357	5.6	23	39	4	3	7	0
Broncos	20.8	323	5.1	24	34	18.6	316	4.7	23	42	5	4	9	0
Browns	16.5	311	5.1	26	36	28.2	392	5.9	12	26	1	0	1	0
Buccaneers	22.1	346	5.2	19	44	23.1	368	5.8	21	38	4	5	9	0

Excerpt

What Data to Look At?

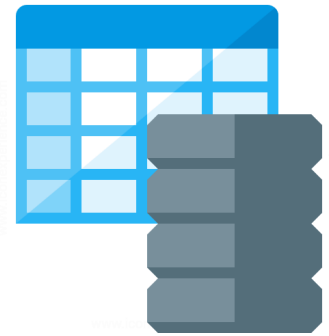
Win-Loss for each week

	49ers	Bears	Bengals	Bills	Broncos	Browns	Buccaneers	Cardinals
49ers	0	0	0	0	0	0	0	0
Bears	0	0	0	0	0	0	0	0
Bengals	0	0	0	0	0	0	0	0
Bills	0	0	0	0	0	0	0	0
Broncos	0	0	0	0	0	0	0	0
Browns	0	0	0	0	0	0	0	0
Buccaneers	0	0	0	0	0	0	0	7-40
Cardinals	0	0	0	0	0	0	0	0
Chargers	0	0	0	0	0	0	0	0
Chiefs	0	0	0	0	0	0	0	0
Colts	0	0	0	0	20-34	0	0	0
Cowboys	0	0	0	0	0	0	0	0

Excerpt

My Process

Spreadsheet



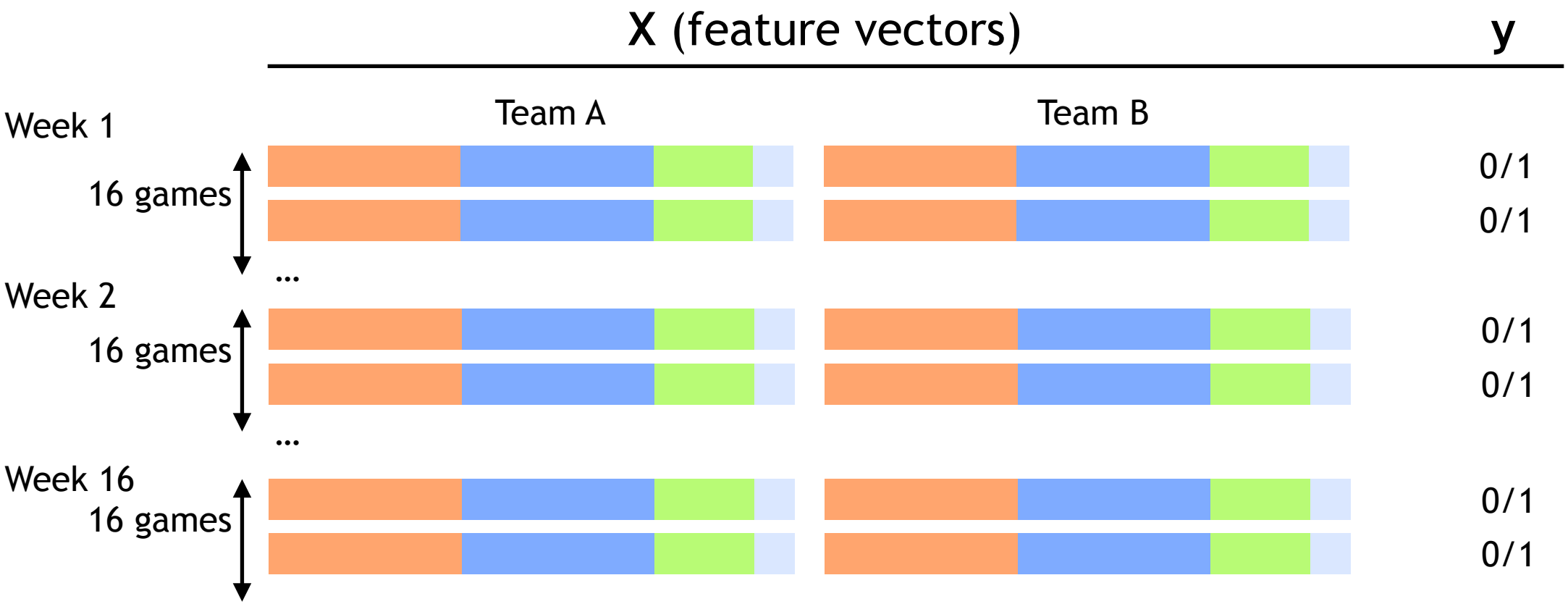
CSV files

Python
Scikit-learn

Results

Structuring the Data

The "prediction" is a label.
1 = team A won
0 = team B won

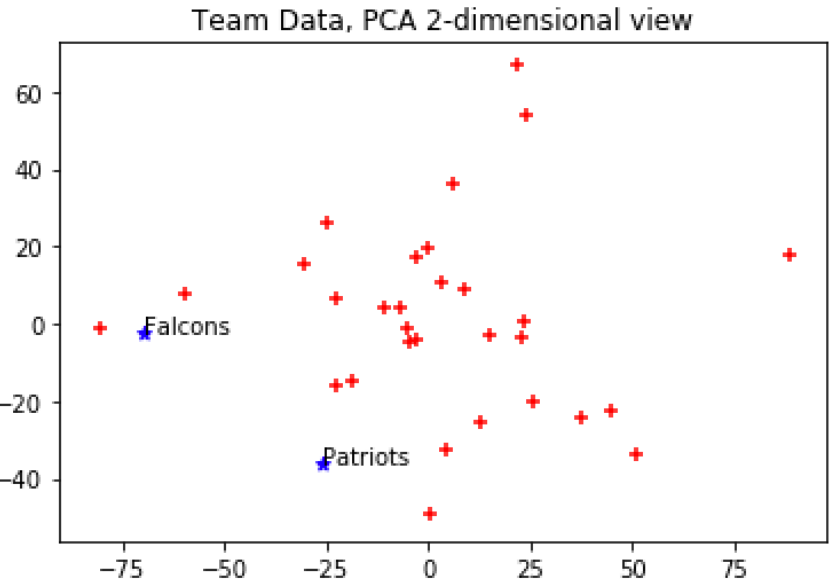


Do I have Good/Smart Data?

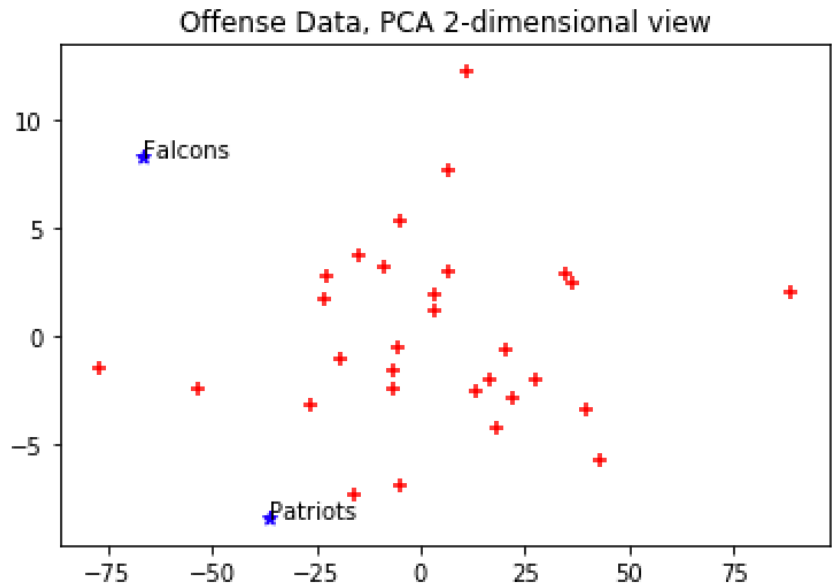
- Do I have enough data?
- Is the data properly prepared and arranged?
- Do I have the “right” features?
- Are there ambiguous samples?
  - Team A plays Team B, team A wins
  - Team B plays Team A, team B wins
- Am I using the “right” algorithm?
- Don’t know, work continues…

PCA Analysis

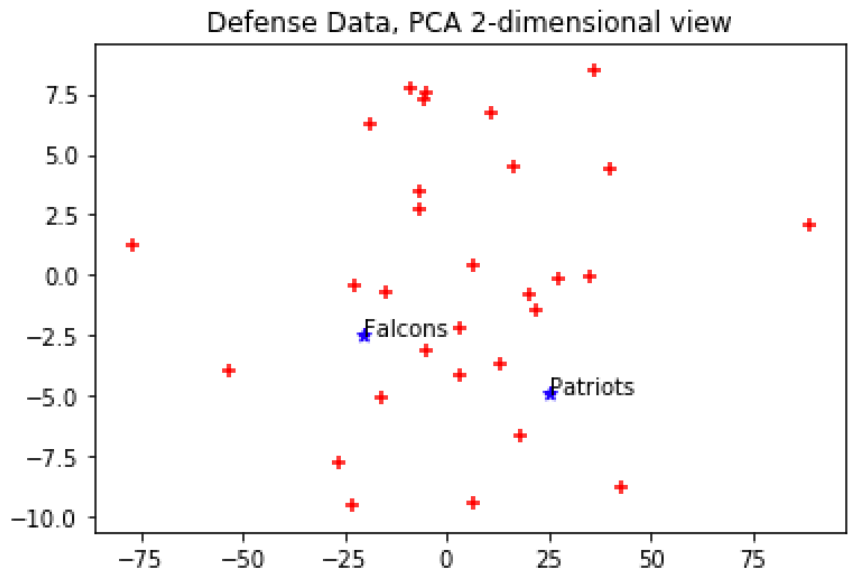
32 rows, 14 columns



32 rows, 7 columns



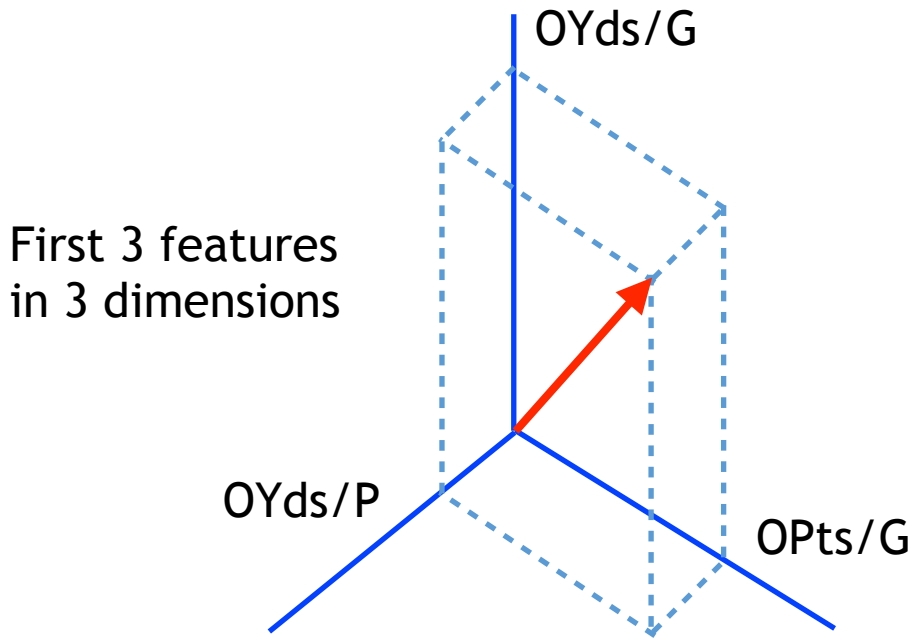
32 rows, 7 columns



Force Vector

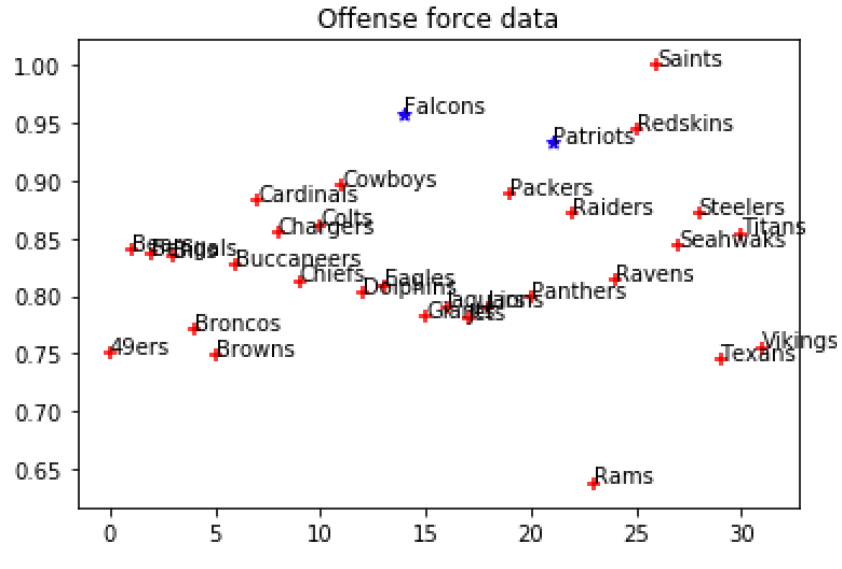
- Is it possible to create an **offensive** and **defensive** "*force vector*"?
- These would be 5 dimensional "vectors" that show magnitude using square-root of the sum of squares
- These 2 vectors could then be combined into a single team "force vector"

OPts/G	OYds/G	OYds/P	Fum	O3rdDown Pct
19.3	308	4.9	29	35

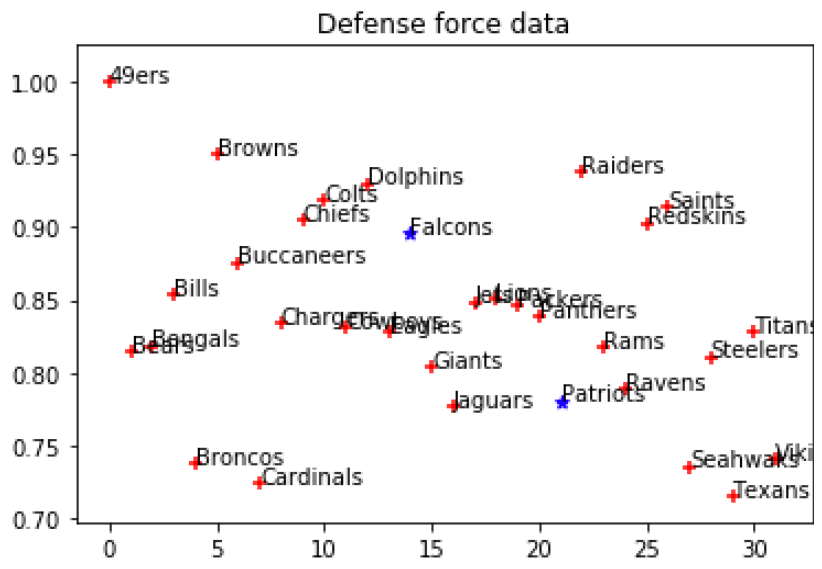


Force Vectors

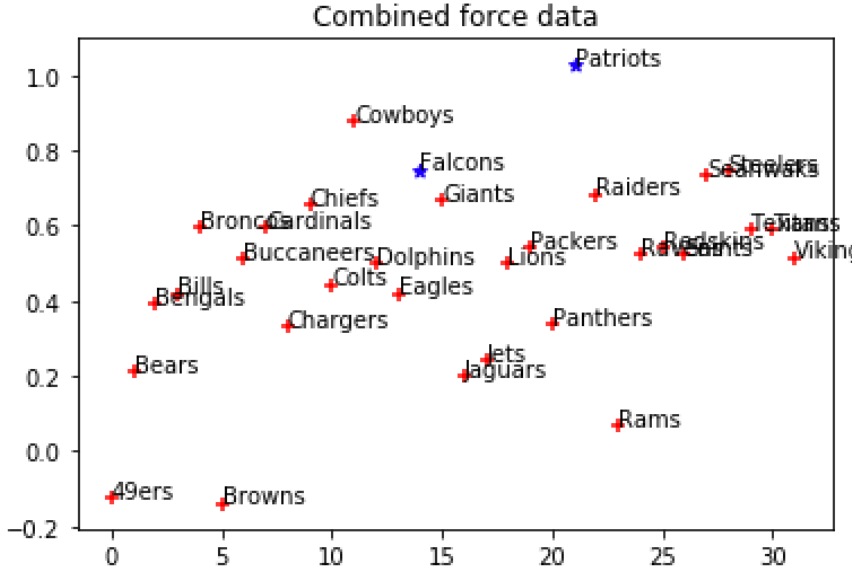
Larger number = **better offense** (normalized to 1)



Larger number = **worse defense** (normalized to 1)



(~normalized to 1)



Currently

- 6 weeks of win/loss data
- 16 games in regular season, so missing 10 weeks
- **X** has 96 rows with 28 features (14 offense, 14 defense)
- **y** has 96 rows (1/0)
- Use cross-validation and select 10% for test, 90% for training
- Using LinearSVC

Results

('X_train shape=', (86, 28))
('X_test shape=', (10, 28))

Expected, Predicted:
(0.0, 1.0)
(0.0, 0.0)
(0.0, 0.0)
(1.0, 0.0)
(0.0, 0.0)
(1.0, 1.0)
(0.0, 0.0)
(0.0, 0.0)
(0.0, 0.0)
(0.0, 0.0)

Accuracy:
0.8

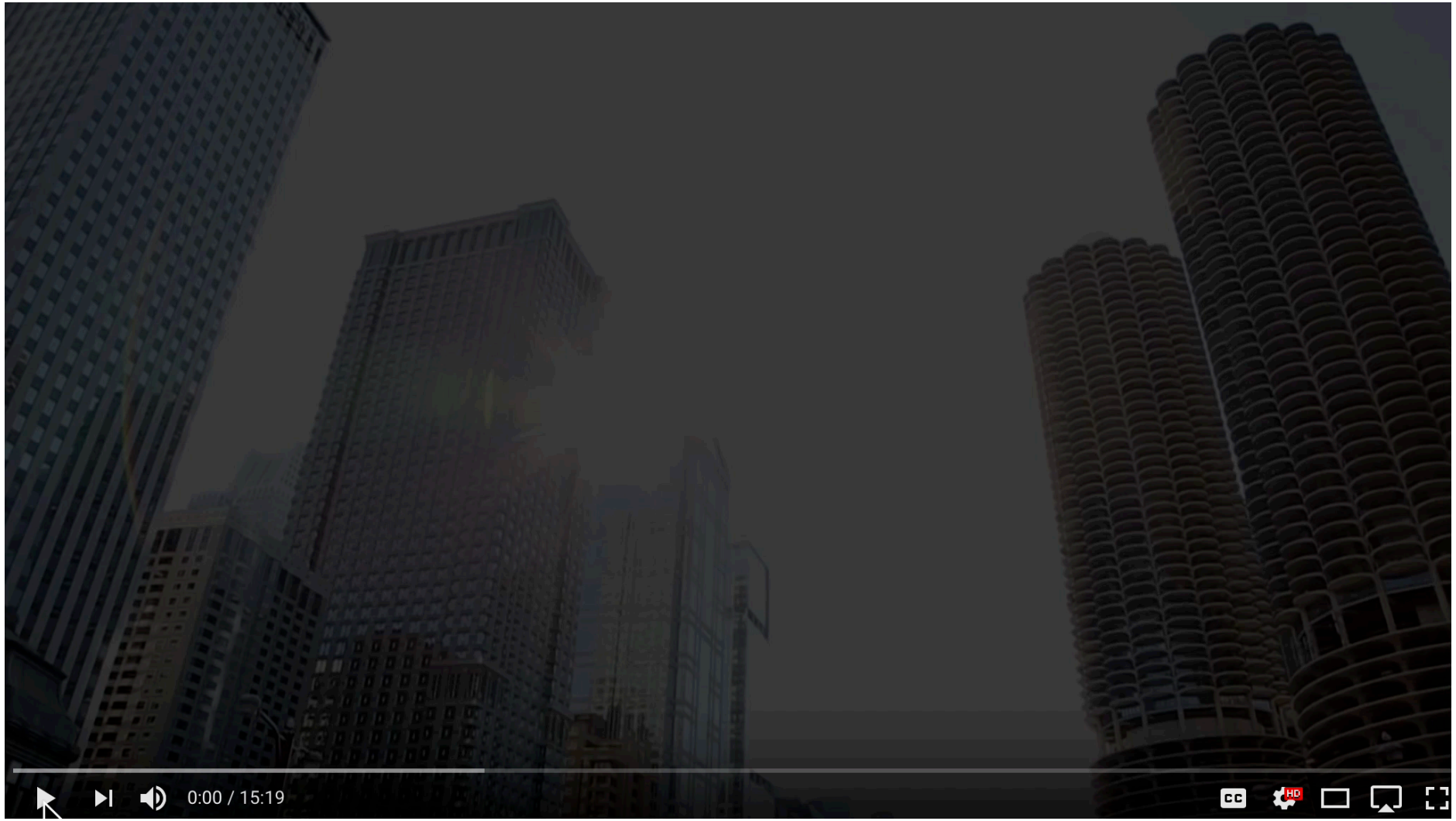
Results (Con't)

- This is misleading…
- More work to do…
  - Have to add the other 10 weeks of win/loss
  - Probably need to do cross-validation with k-folds
  - May introduce the force vectors as features?
  - Need to try other algorithms, want to investigate Spark and other SVM libraries

Predictive Analytics Summary

- Combines:
  - **Statistics**
  - **Data mining**
  - **Machine learning**
- Extract “Good Data” - what we care about
- Properly prepare the data to get “Smart Data”
- Use machine learning algorithms to make predictions
  - Time intensive
  - Failure more common than success

Hewlett Packard's Viewpoint



Source: https://www.youtube.com/watch?v=u3IaXvjDiOE Bosch ConnectedWorld Conference 2016 (16 minutes)

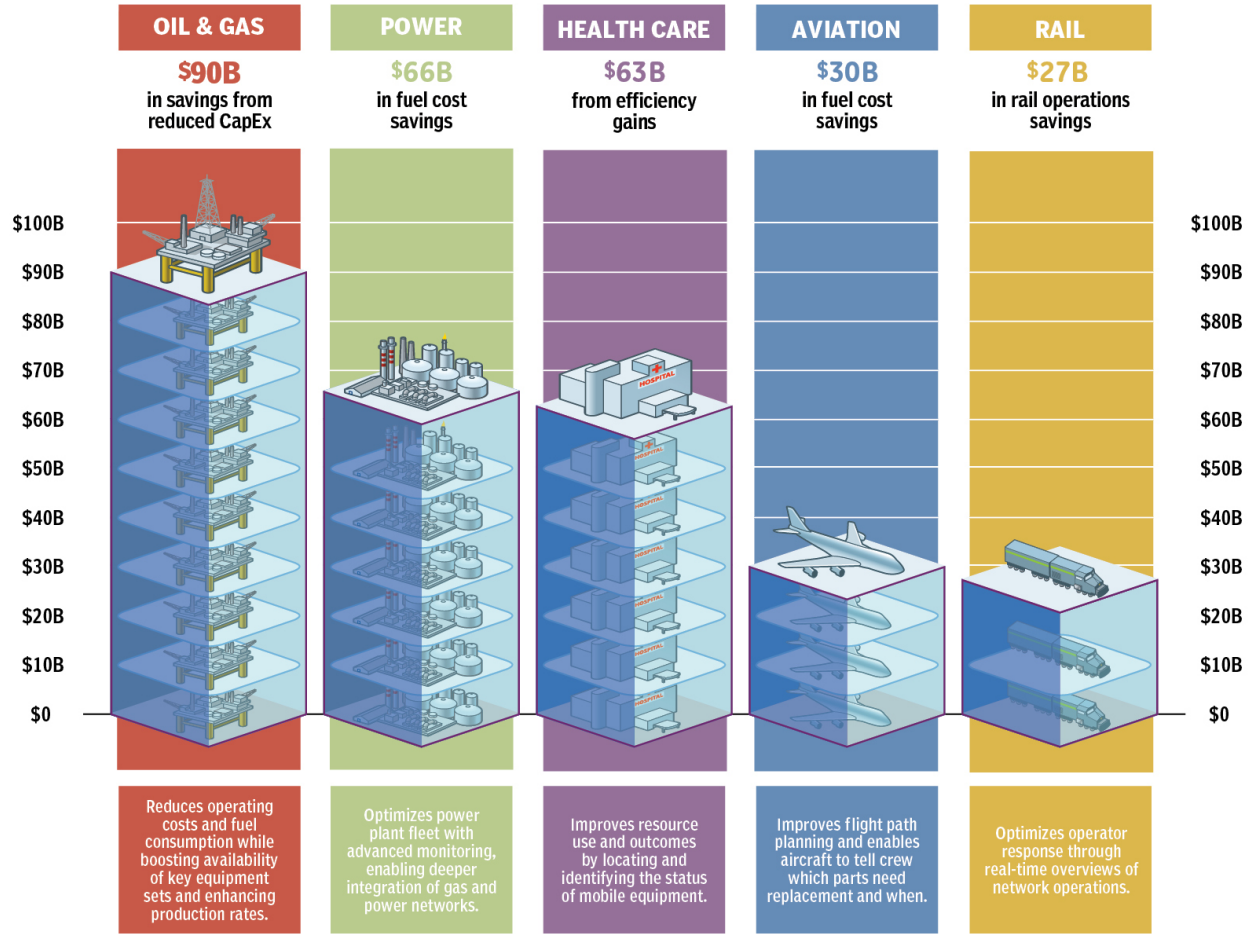
Executive Skepticism

- IEEE article, *Digital Transformation*
- Survey: 1 in 5 executives thinks it's a waste of time
- Recommendations:
  - Focus on business outcomes
    - Operational efficiencies, increased revenues, new business opportunities
  - Commit to execution, takes a lot of work
  - Sharpen your soft skills, see "technical skills stack" link
  - Put people first. Transformation results in changes in to your workforce
    - Some previous jobs will be obsolete, while new jobs are created

Source: https://insight.ieeeusa.org/articles/how-can-organizations-succeed-digital-transformation/

INDUSTRIAL INTERNET: THE POWER OF 1%

Efficiency gains as small as 1% could have sizable benefits over 15 years when scaled up across the economic system.



INDUSTRIAL INTERNET BENEFITS

Source: GE estimates / Postmedia

End