Machine Learning: Data Foundation

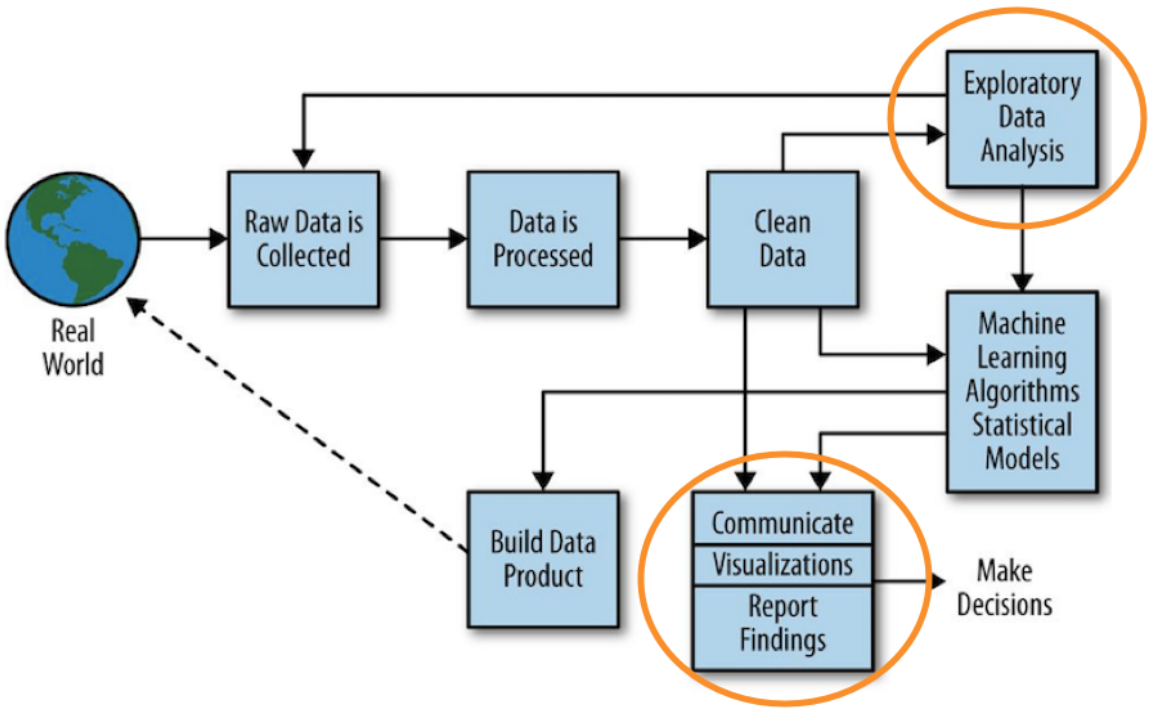
Day 2

Visualization

Objective

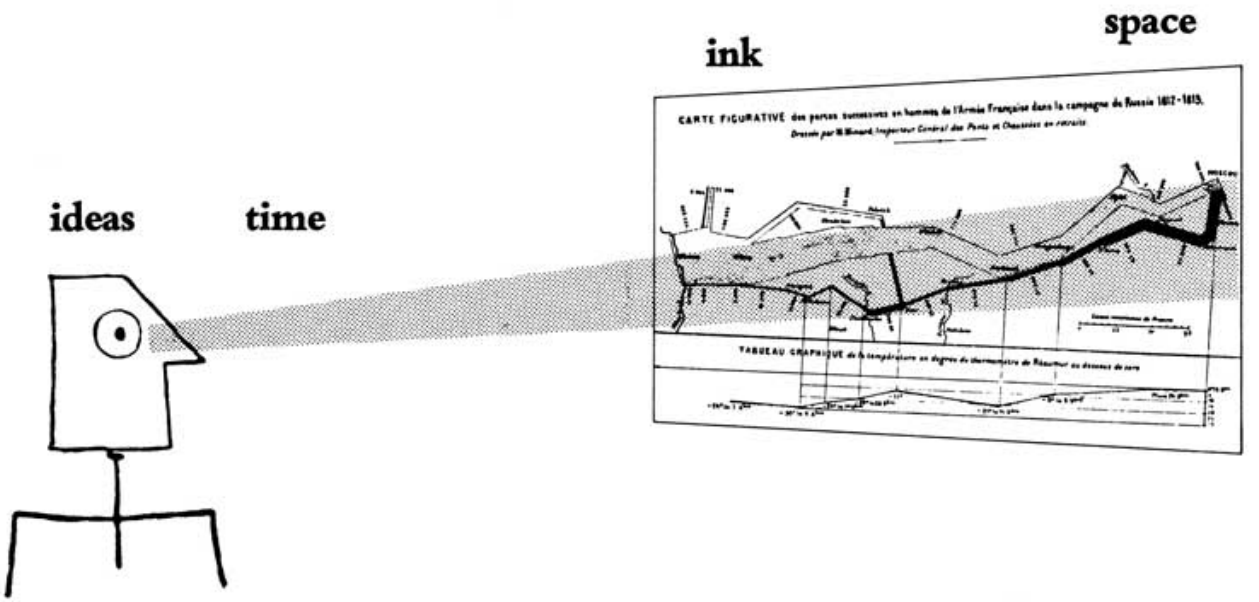
- Lay down some basic visualization theory
- Learn about the various types of data visualization tools
- Learn some basic Python techniques

Data Science Pipeline



"Doing Data Science"

GOAL: communicate the MOST ideas, in the LEAST amount of time, with the LEAST amount of ink, in the LEAST amount of space



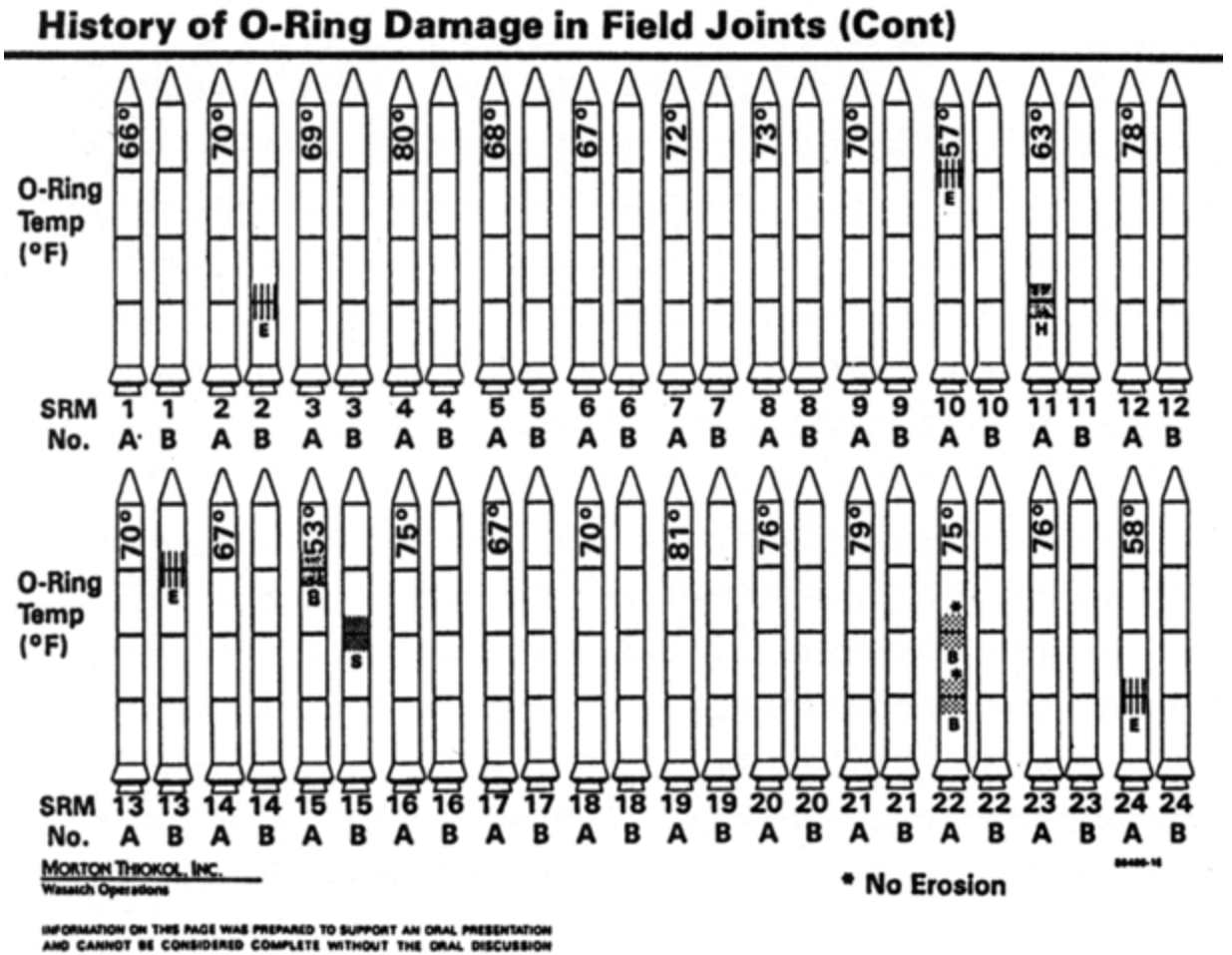
Edward Tufte

Charles Minard's graphic of Napoleon's Russian campaign of 1812

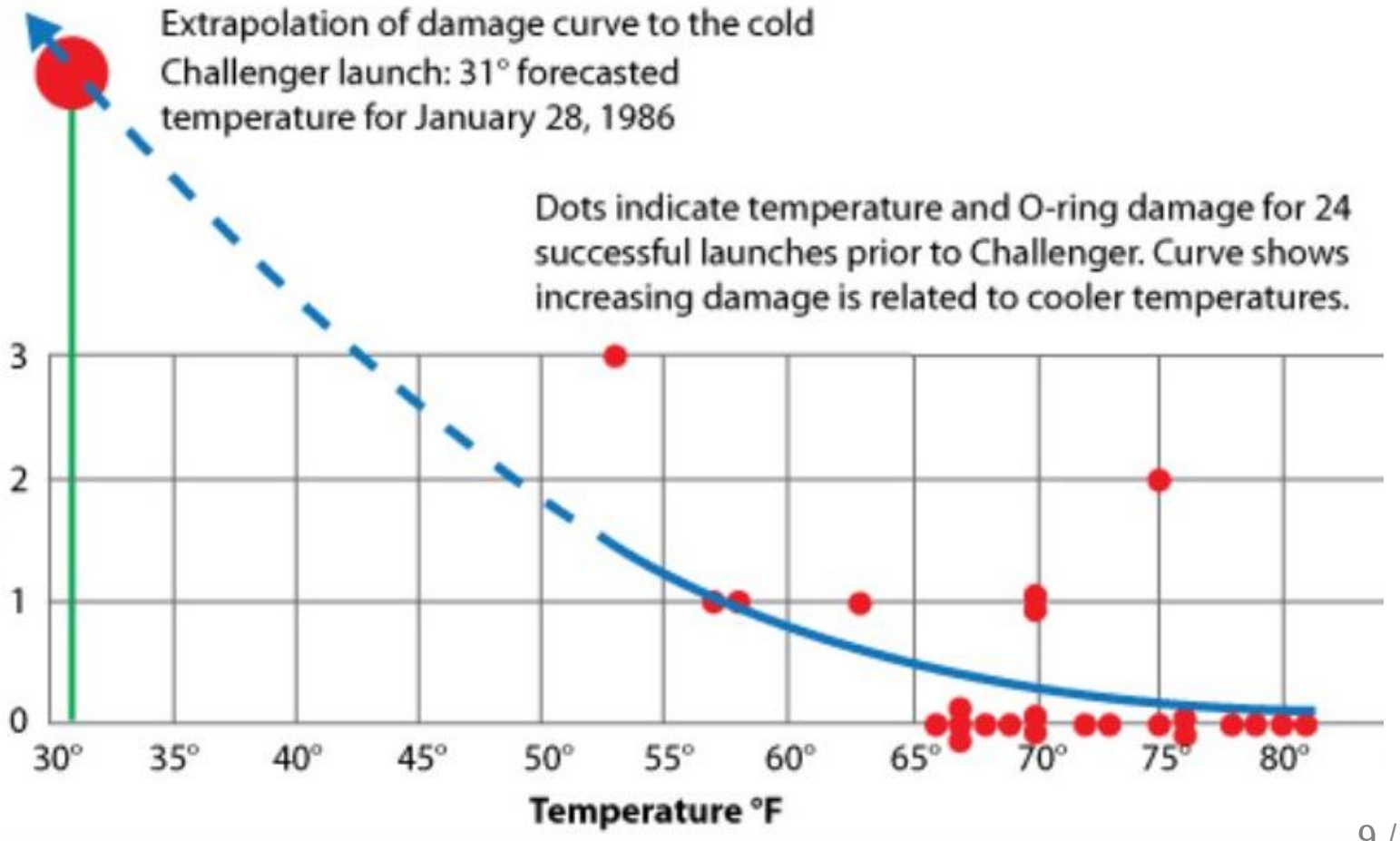
- Number of men present represented by widths of the colored zones
  - brown designates men who enter Russia, the black those who leave it
  - 1 mm = ten thousand men

Space Shuttle Challenger Explosion: January 28, 1986

Hiding the Story with "Chart Junk"?



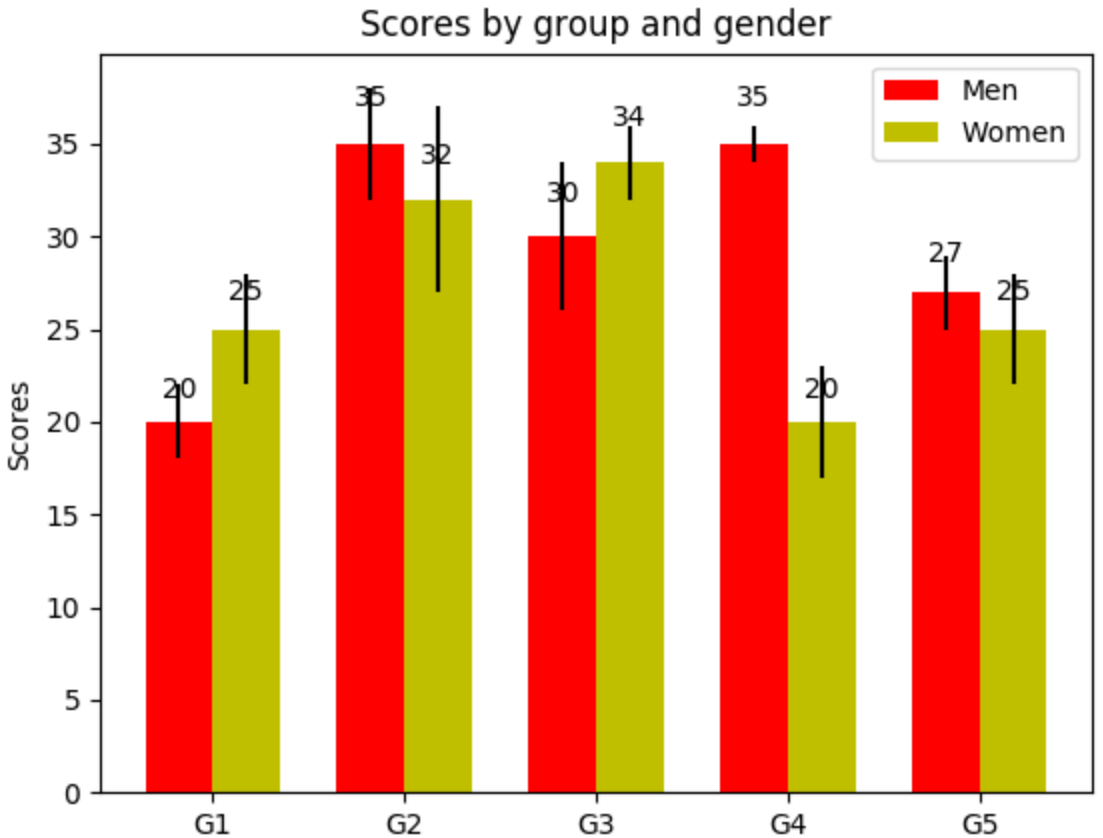
Challenger Data Redrawn by Tufte



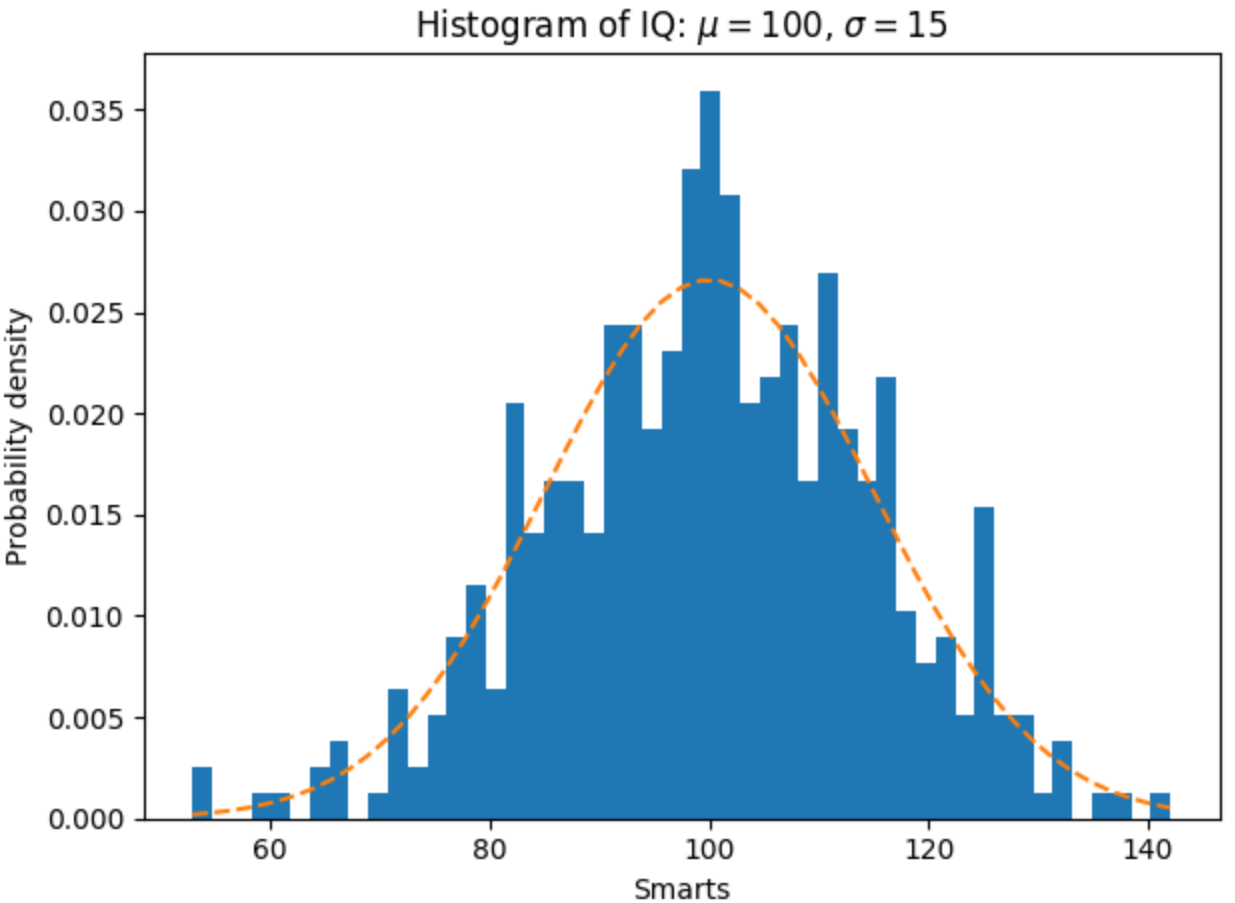

BY THE YEAR
2018
2019
2020
ALL
INFORMATION
ALL
NONE
2019
WILL BE
6 YEARS FROM NOW
(72 MONTHS)
COMMUNICATED
YES!
IN
THIS
CLEAR AND CONCISE
26
0
FORMAT

Timeseries Data

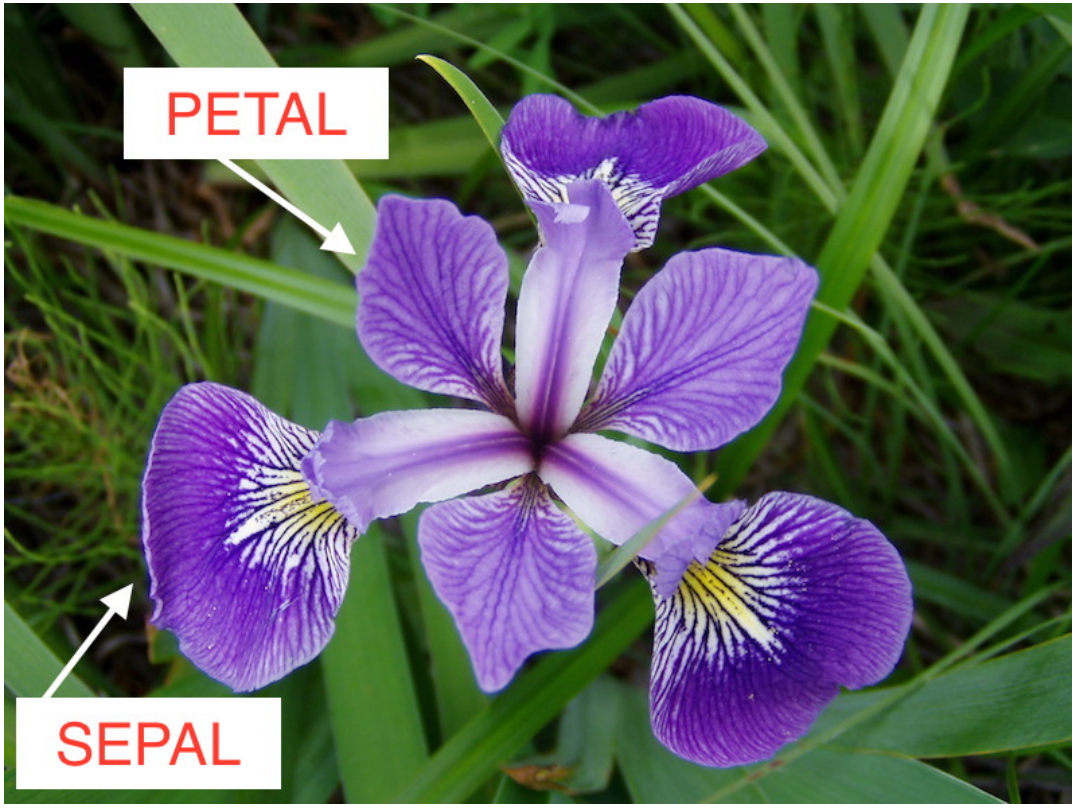
Bar Charts


Scores by group and gender
Men
Women
Scores
0
5
10
15
20
25
30
35
G1
G2
G3
G4
G5
20
25
35
32
30
34
35
20
27
25

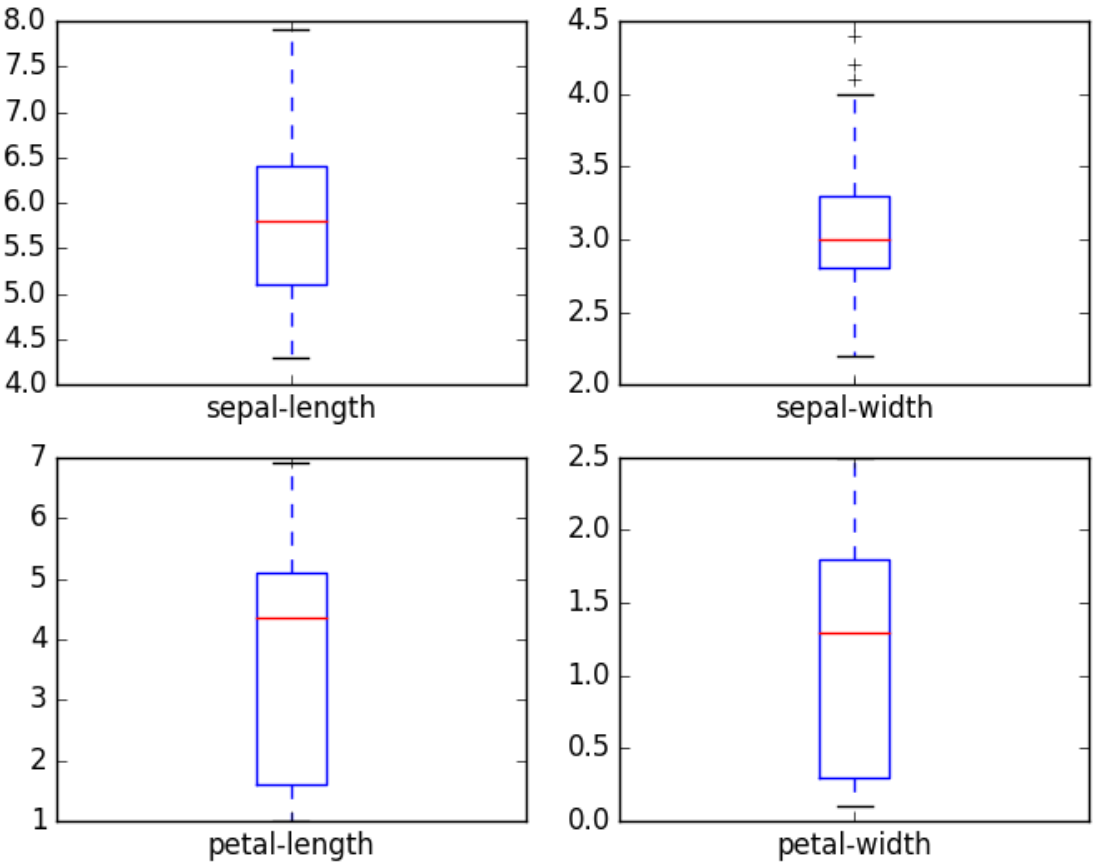
Histograms


Histogram of IQ: $\mu = 100$, $\sigma = 15$
Probability density
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
60
80
100
120
140
Smarts

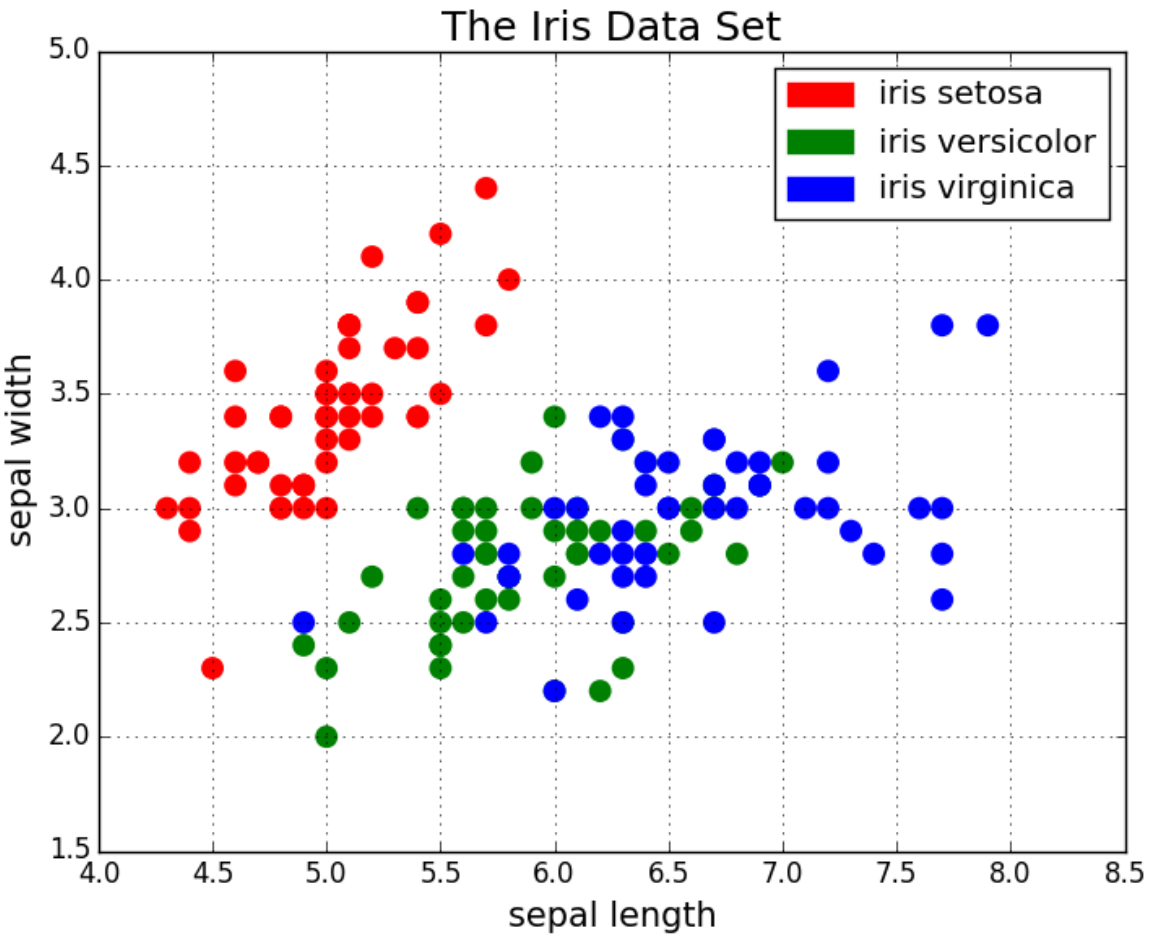
The Iris Data Set



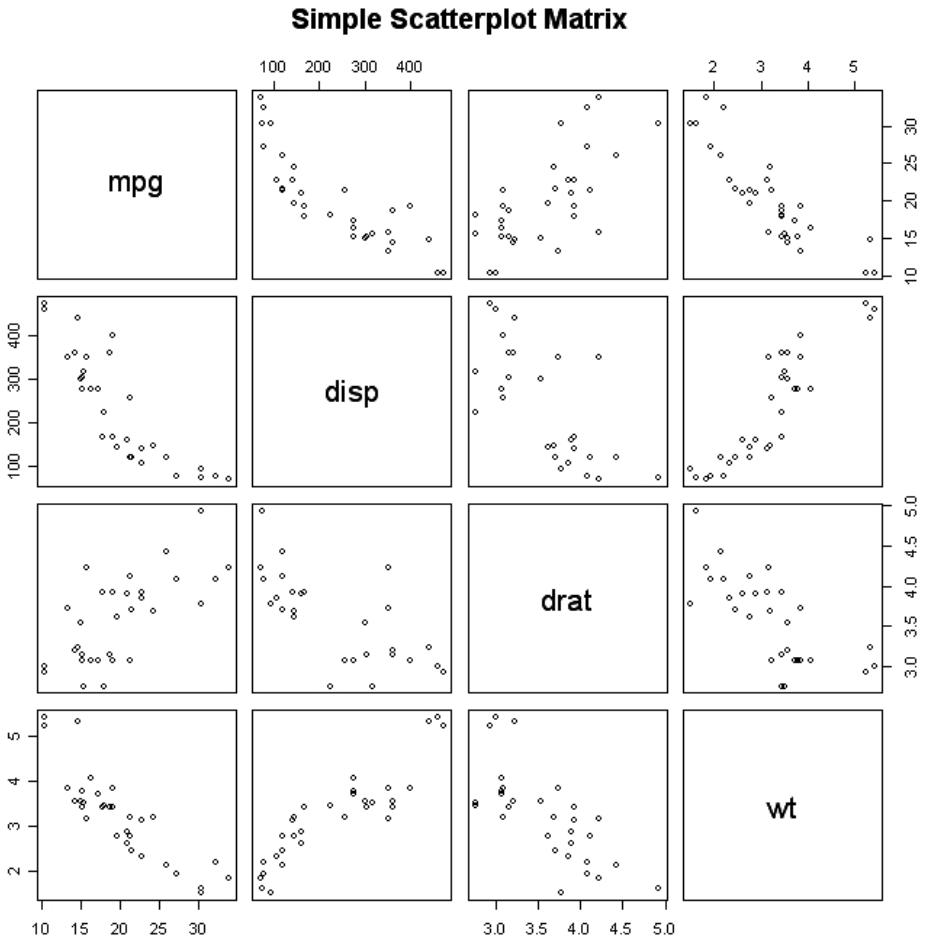
Box and Whiskers Plot



Scatter plots



Scatter Plot Matrices



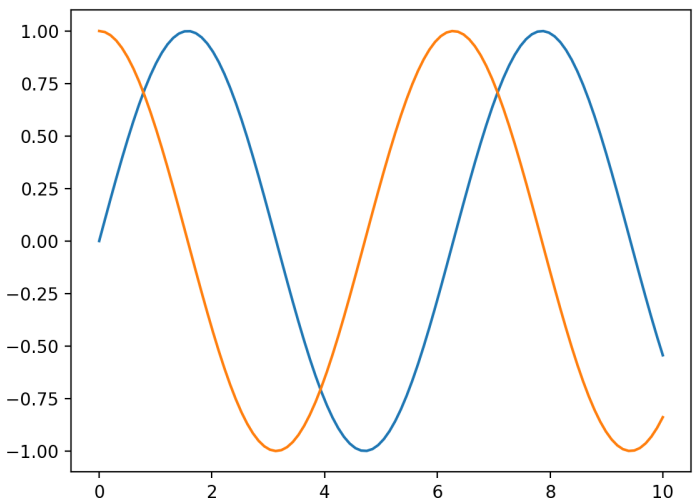
Demo: Data Visualization

Demo: Data Visualization

- there are screenshots in this presentation to maintain continuity, but let's open the notebook named **`Demo - Data Visualization.ipynb`** and go through it together, then there is an exercise to do on your own
- when done, click here to skip screesnhots

```
# show.py

import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 10, 100)

plt.plot(x, np.sin(x))
plt.plot(x, np.cos(x))

plt.show()
```

```
> python show.py
```



Saving Image Files

In [5]:
```
fig.savefig('my_figure.png')
```

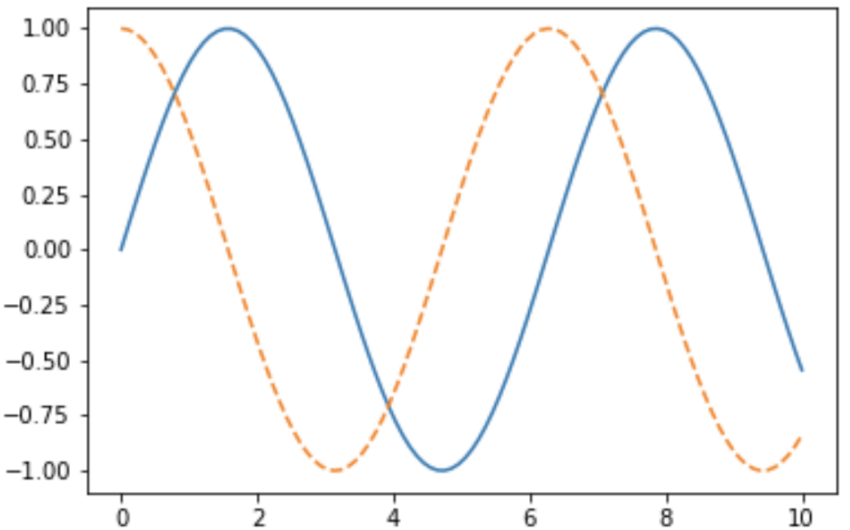
In [6]:
```
!ls -lh my_figure.png
```

```
-rw-r--r--  1 brian  staff    22K Jan 24 06:47 my_figure.png
```

Show Image Files

In [7]:
```
from IPython.display import Image
Image('my_figure.png')
```

Out[7]:



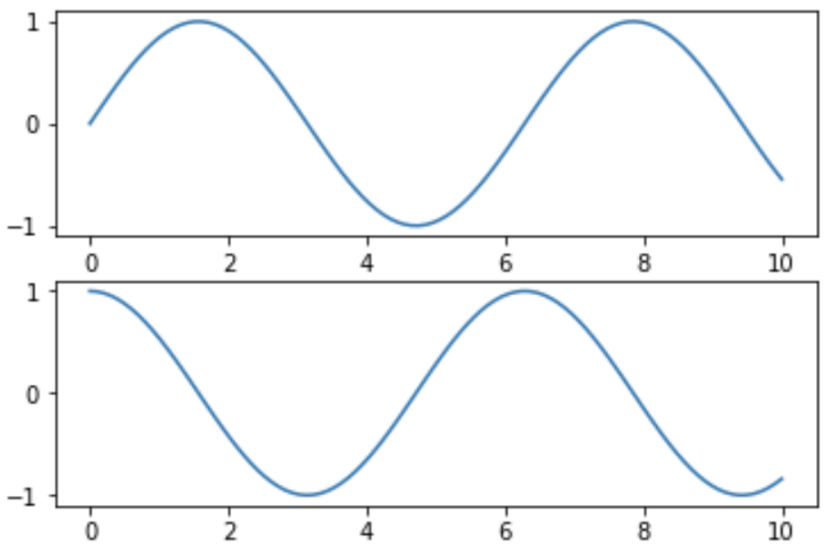
Show Available File Support

```
In [8]: fig.canvas.get_supported_filetypes()
```

```
Out[8]: {u'eps': u'Encapsulated Postscript',
         u'jpeg': u'Joint Photographic Experts Group',
         u'jpg': u'Joint Photographic Experts Group',
         u'pdf': u'Portable Document Format',
         u'pgf': u'PGF code for LaTeX',
         u'png': u'Portable Network Graphics',
         u'ps': u'Postscript',
         u'raw': u'Raw RGBA bitmap',
         u'rgba': u'Raw RGBA bitmap',
         u'svg': u'Scalable Vector Graphics',
         u'svgz': u'Scalable Vector Graphics',
         u'tif': u'Tagged Image File Format',
         u'tiff': u'Tagged Image File Format'}
```

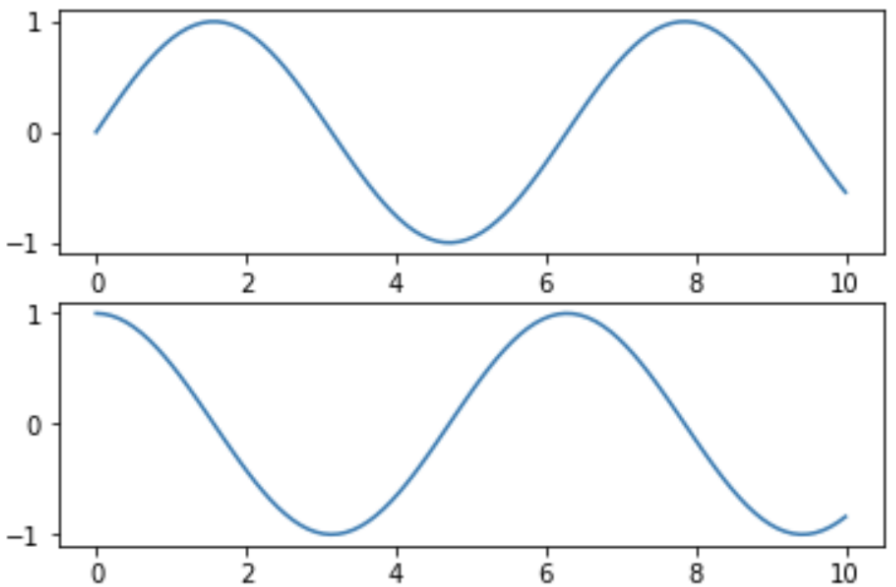
MATLAB-Style Interface

In [9]:
```
plt.figure()  # create a plot figure

# create the first of two panels and set current axis
plt.subplot(2, 1, 1) # (rows, columns, panel number)
plt.plot(x, np.sin(x))

# create the second panel and set current axis
plt.subplot(2, 1, 2)
plt.plot(x, np.cos(x));
```



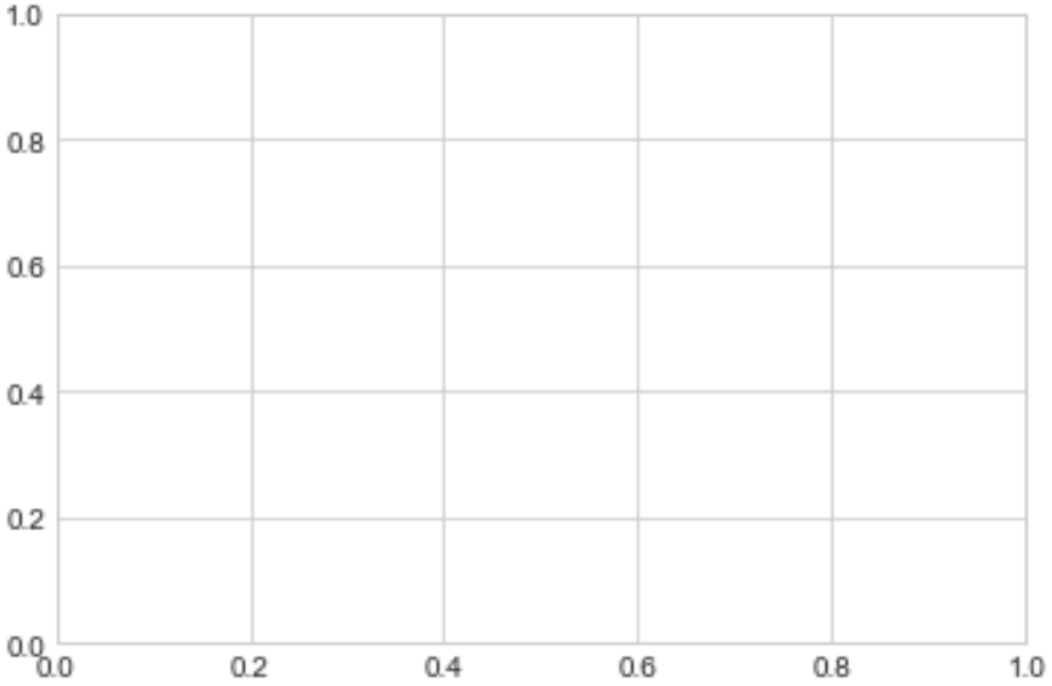
Object-Oriented Interface

In [10]:
```
# First create a grid of plots
# ax will be an array of two Axes objects
fig, ax = plt.subplots(2)

# Call plot() method on the appropriate object
ax[0].plot(x, np.sin(x))
ax[1].plot(x, np.cos(x));
```



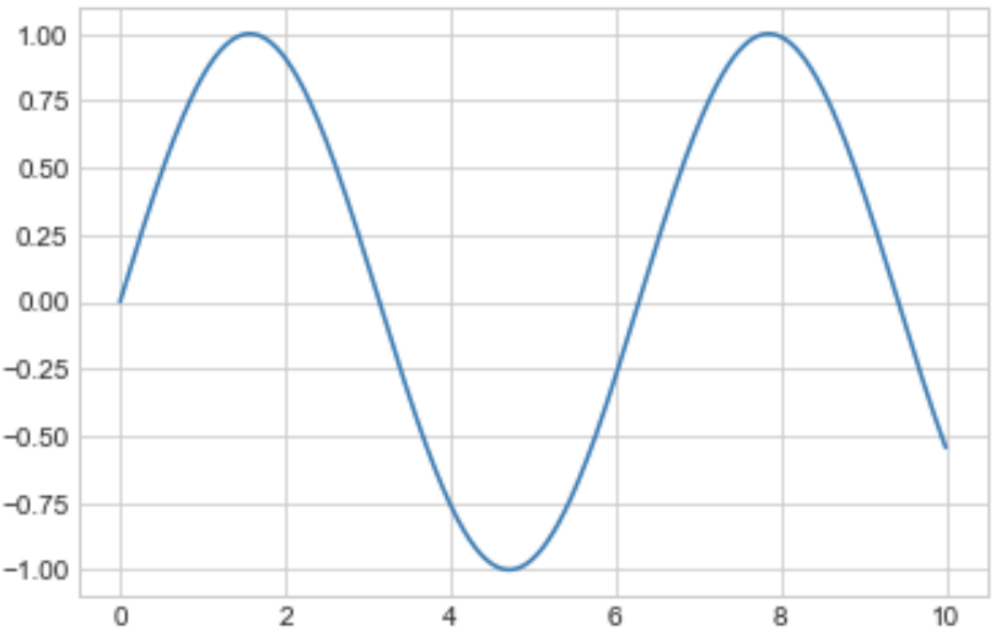
Set up a Grid

In [11]:
```
plt.style.use('seaborn-whitegrid')
fig = plt.figure()
ax = plt.axes()
```



Draw a Function

In [14]:
```
plt.style.use('seaborn-whitegrid')
fig = plt.figure()
ax = plt.axes()
x = np.linspace(0, 10, 1000)
ax.plot(x, np.sin(x));
```

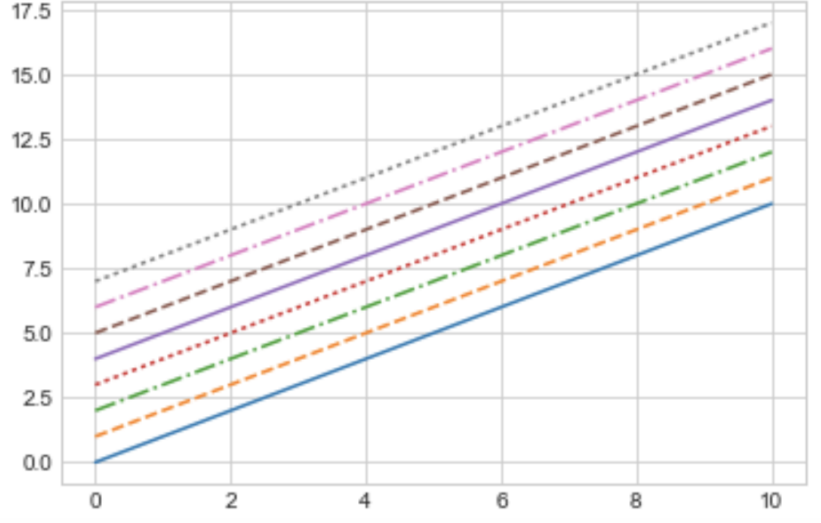


Ways to Specify Color

In [15]:
```
plt.plot(x, np.sin(x - 0), color='blue')        # specify color by name
plt.plot(x, np.sin(x - 1), color='g')           # short color code (rgbcmyk)
plt.plot(x, np.sin(x - 2), color='0.75')        # Grayscale between 0 and 1
plt.plot(x, np.sin(x - 3), color='#FFDD44')     # Hex code (RRGGBB from 00 to FF)
plt.plot(x, np.sin(x - 4), color=(1.0,0.2,0.3)) # RGB tuple, values 0 to 1
plt.plot(x, np.sin(x - 5), color='chartreuse'); # all HTML color names supported
```

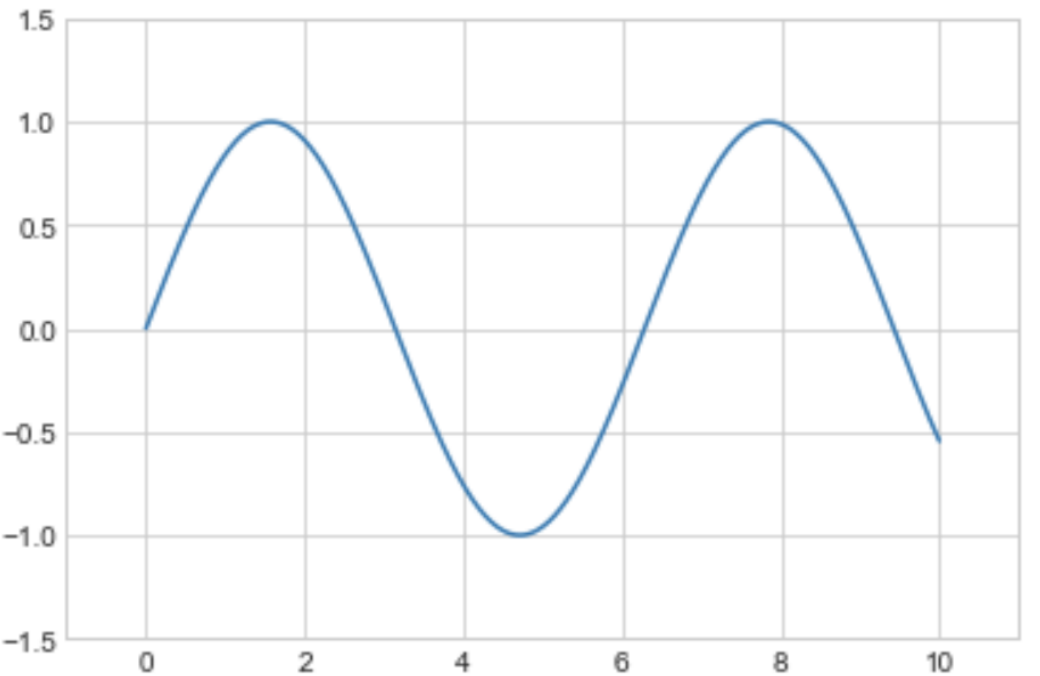
Ways to Specify Line Style

In [16]:
```
plt.plot(x, x + 0, linestyle='solid')
plt.plot(x, x + 1, linestyle='dashed')
plt.plot(x, x + 2, linestyle='dashdot')
plt.plot(x, x + 3, linestyle='dotted');

# For short, you can use the following codes:
plt.plot(x, x + 4, linestyle='-')  # solid
plt.plot(x, x + 5, linestyle='--') # dashed
plt.plot(x, x + 6, linestyle='-.') # dashdot
plt.plot(x, x + 7, linestyle=':');  # dotted
```



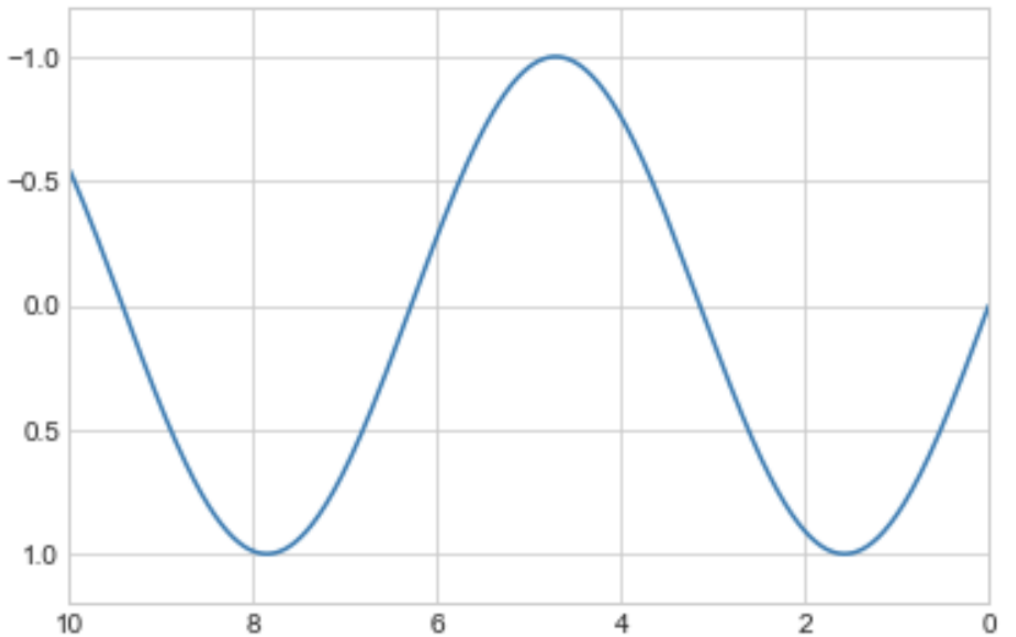
Setting Axes Limits

In [17]:
```
plt.plot(x, np.sin(x))

plt.xlim(-1, 11)
plt.ylim(-1.5, 1.5);
```



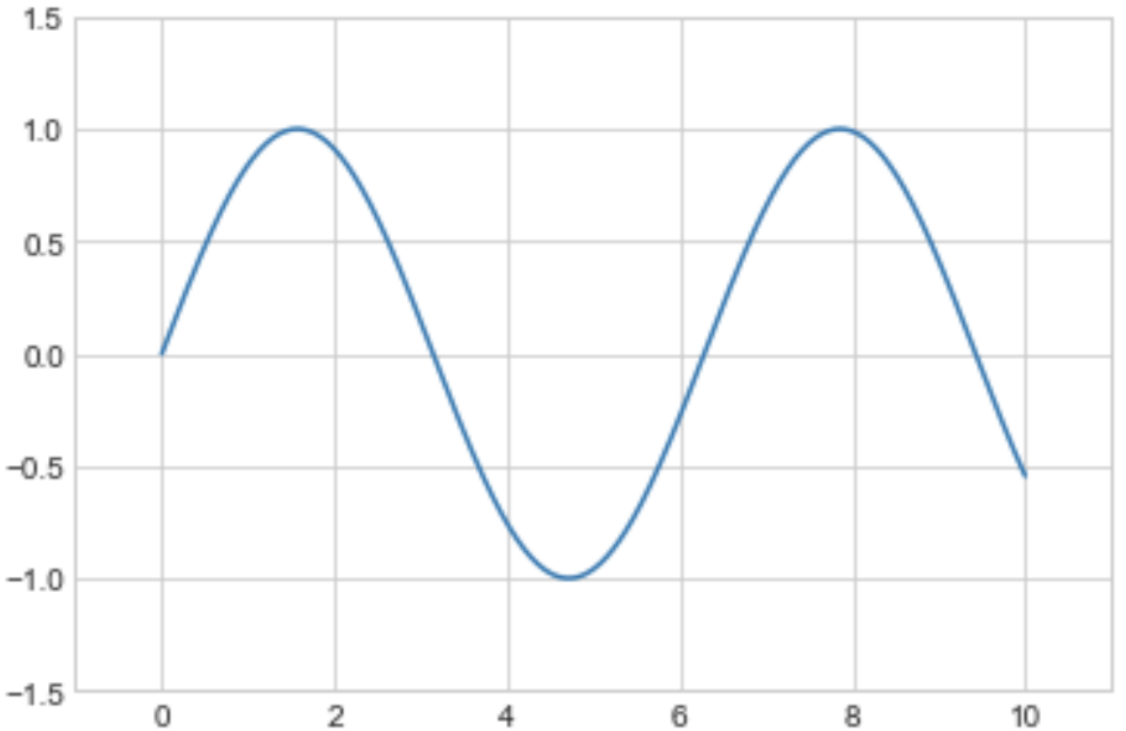
Flipping the Axes Limits

In [18]:
```
plt.plot(x, np.sin(x))

plt.xlim(10, 0)
plt.ylim(1.2, -1.2);
```



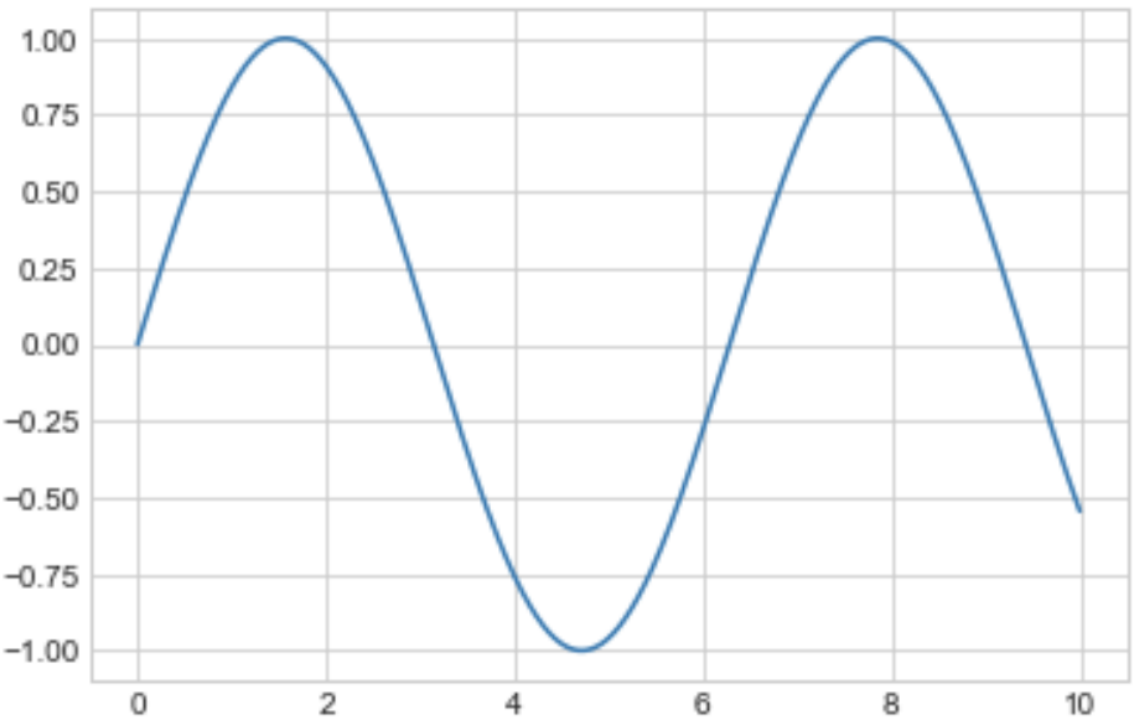
Axis

In [19]:
```
plt.plot(x, np.sin(x))
plt.axis([-1, 11, -1.5, 1.5]);
```



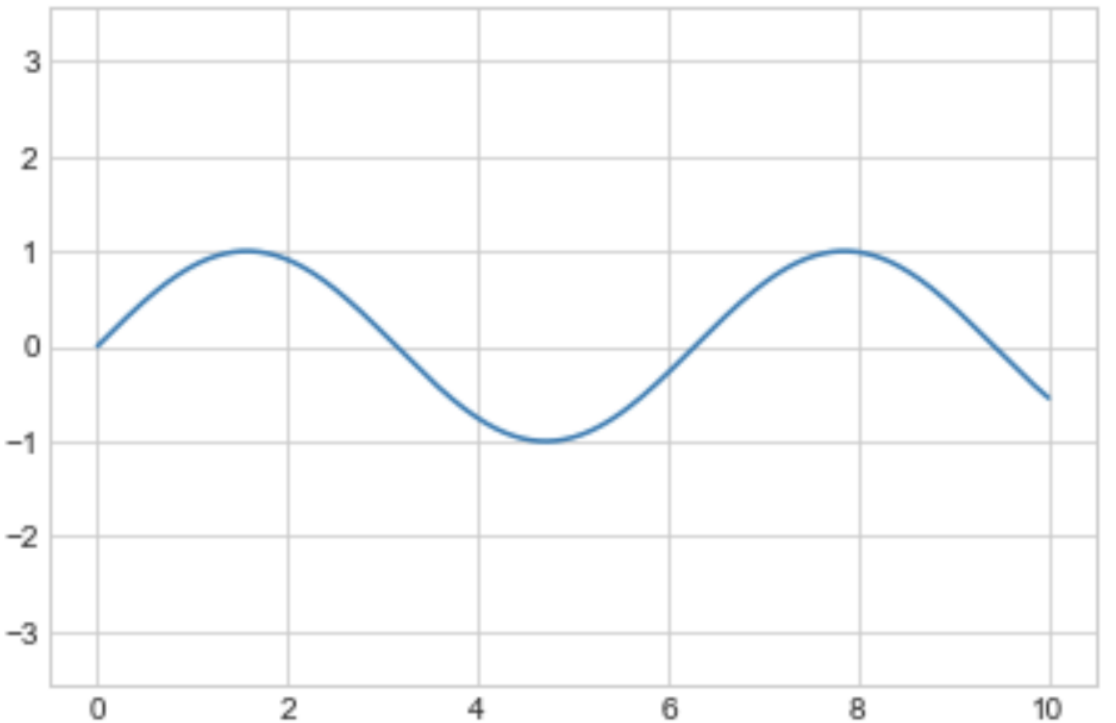
Tight Fit

In [20]:
```
plt.plot(x, np.sin(x))
plt.axis('tight');
```



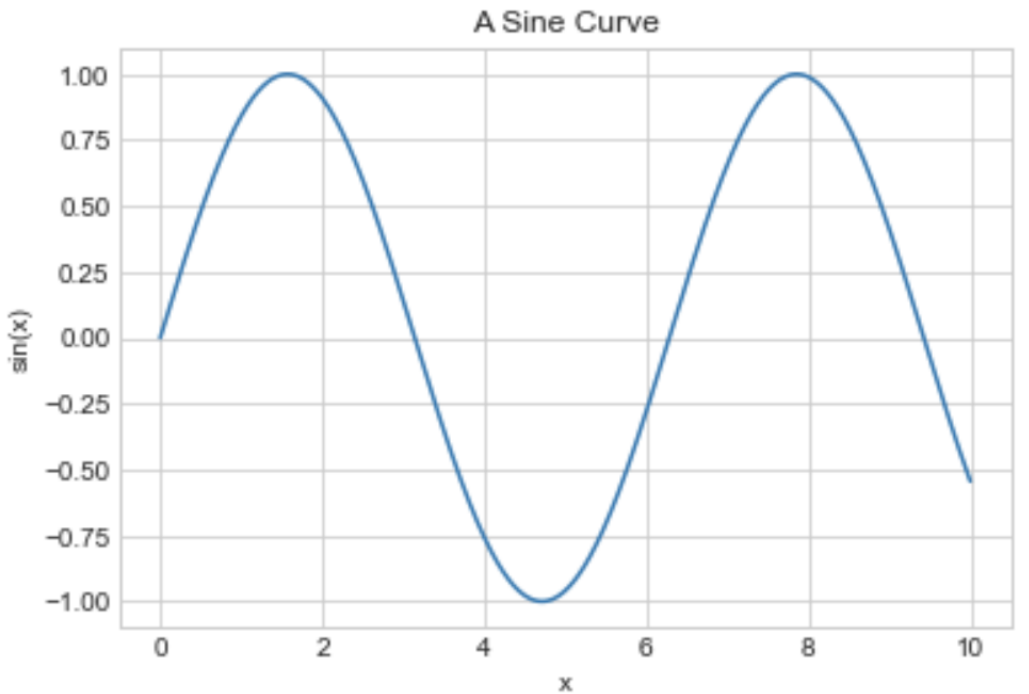
Equal Aspect Ratio

In [21]:
```
plt.plot(x, np.sin(x))
plt.axis('equal');
```



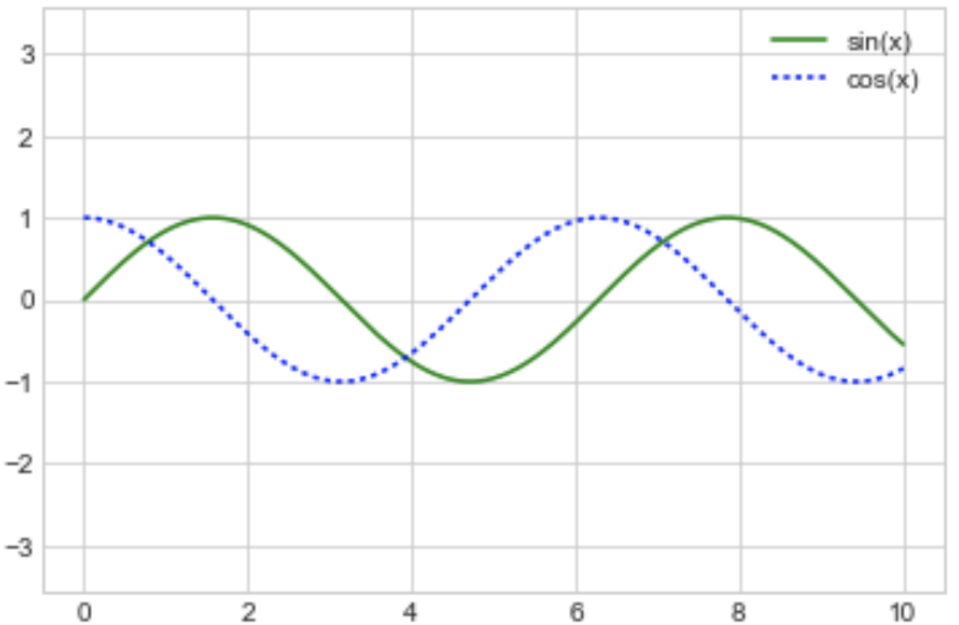
Labels

In [22]:
```
plt.plot(x, np.sin(x))
plt.title("A Sine Curve")
plt.xlabel("x")
plt.ylabel("sin(x)");
```



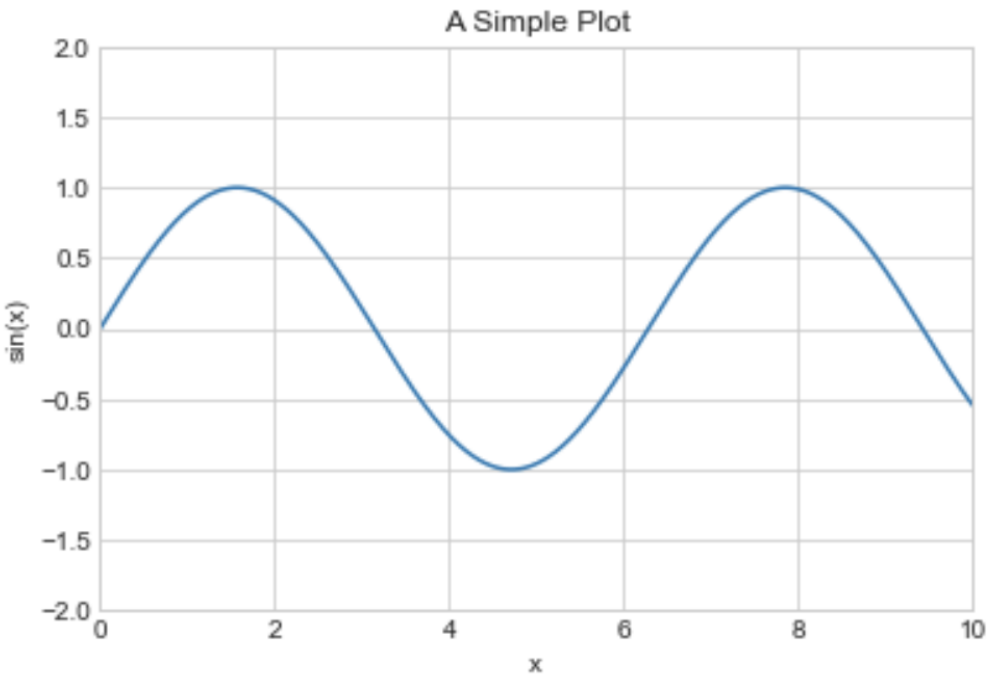
Legends

In [23]:
```
plt.plot(x, np.sin(x), '-g', label='sin(x)')
plt.plot(x, np.cos(x), ':b', label='cos(x)')
plt.axis('equal')

plt.legend();
```



Object-Oriented Interface

In [24]:
```
ax = plt.axes()
ax.plot(x, np.sin(x))
ax.set(xlim=(0, 10), ylim=(-2, 2),
       xlabel='x', ylabel='sin(x)',
       title='A Simple Plot');
```

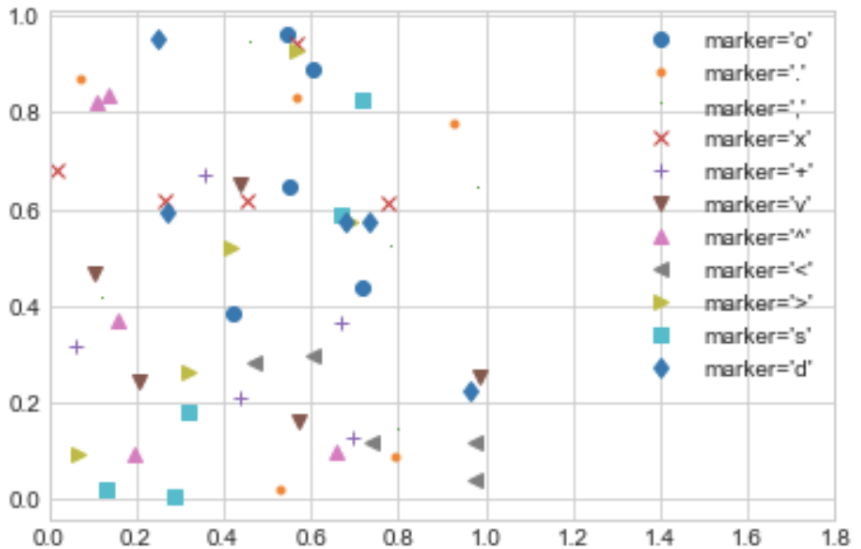


Interface Differences

MATLAB-Style	OO Style
plt.xlabel()	ax.set_xlabel()
plt.ylabel()	ax.set_ylabel()
plt.xlim()	ax.set_xlim()
plt.ylim()	ax.set_ylim()
plt.title()	ax.set_title()

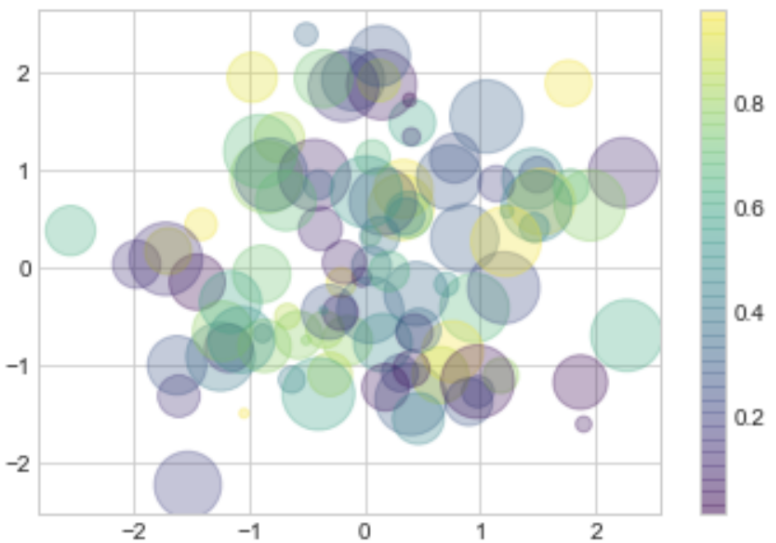
Specifying Markers

In [25]:
```
rng = np.random.RandomState(0)
for marker in ['o', '.', ',', 'x', '+', 'v', '^', '<', '>', 's', 'd']:
    plt.plot(rng.rand(5), rng.rand(5), marker,
             label="marker='{0}'".format(marker))
plt.legend(numpoints=1)
plt.xlim(0, 1.8);
```



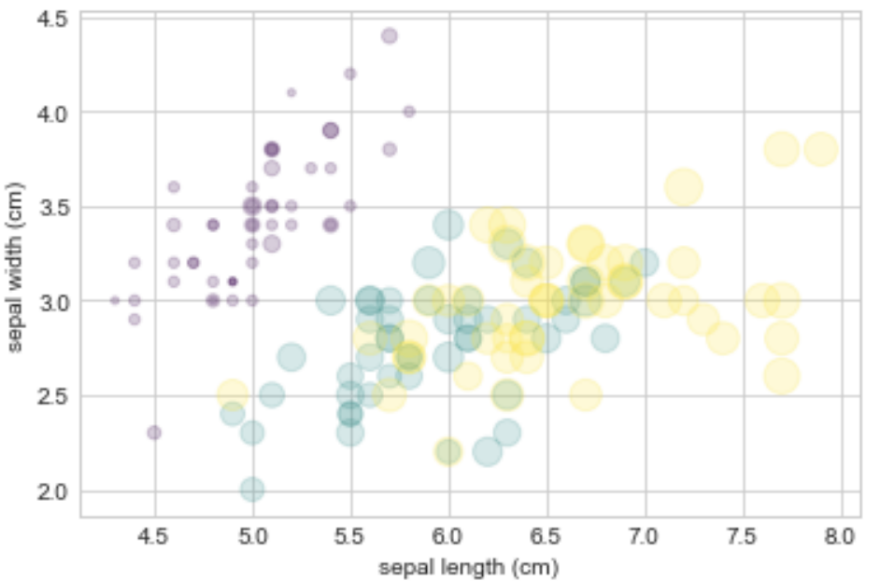
Scatterplot with Colors and Sizes

In [26]:
```
rng = np.random.RandomState(0)
x = rng.randn(100)
y = rng.randn(100)
colors = rng.rand(100)
sizes = 1000 * rng.rand(100)

plt.scatter(x, y, c=colors, s=sizes, alpha=0.3,
            cmap='viridis')
plt.colorbar();  # show color scale
```



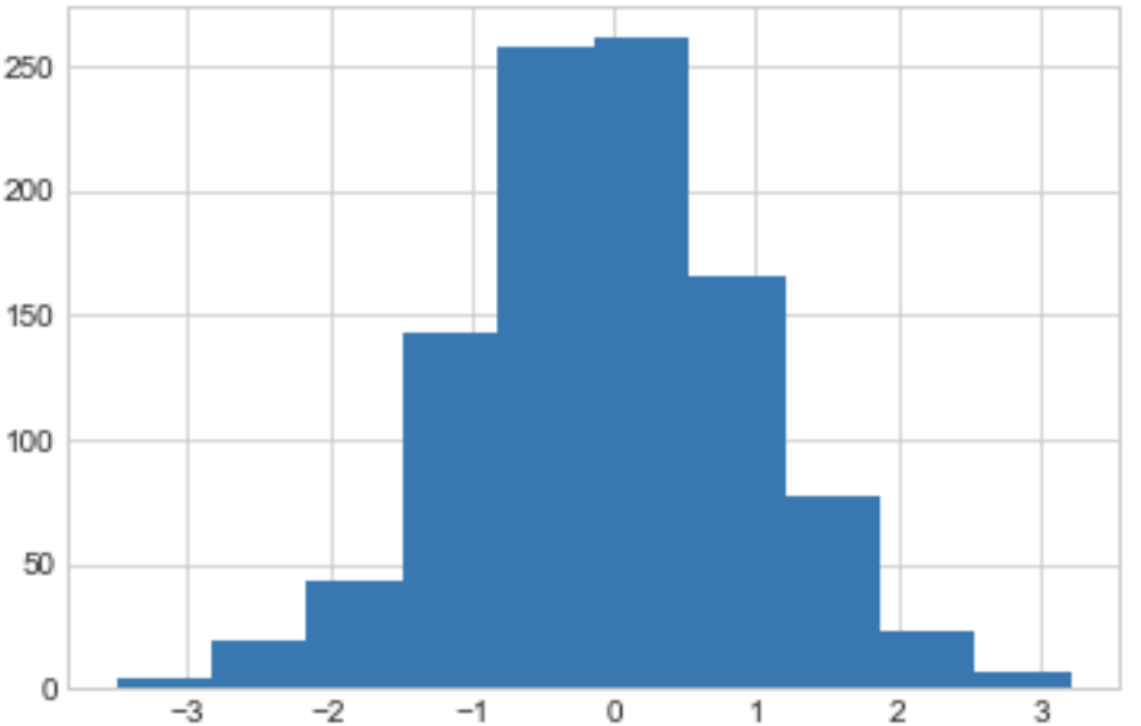
Visualizing Multiple Dimensions

In [27]:
```
from sklearn.datasets import load_iris
iris = load_iris()
features = iris.data.T

plt.scatter(features[0], features[1], alpha=0.2,
            s=100*features[3], c=iris.target, cmap='viridis')
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1]);
```



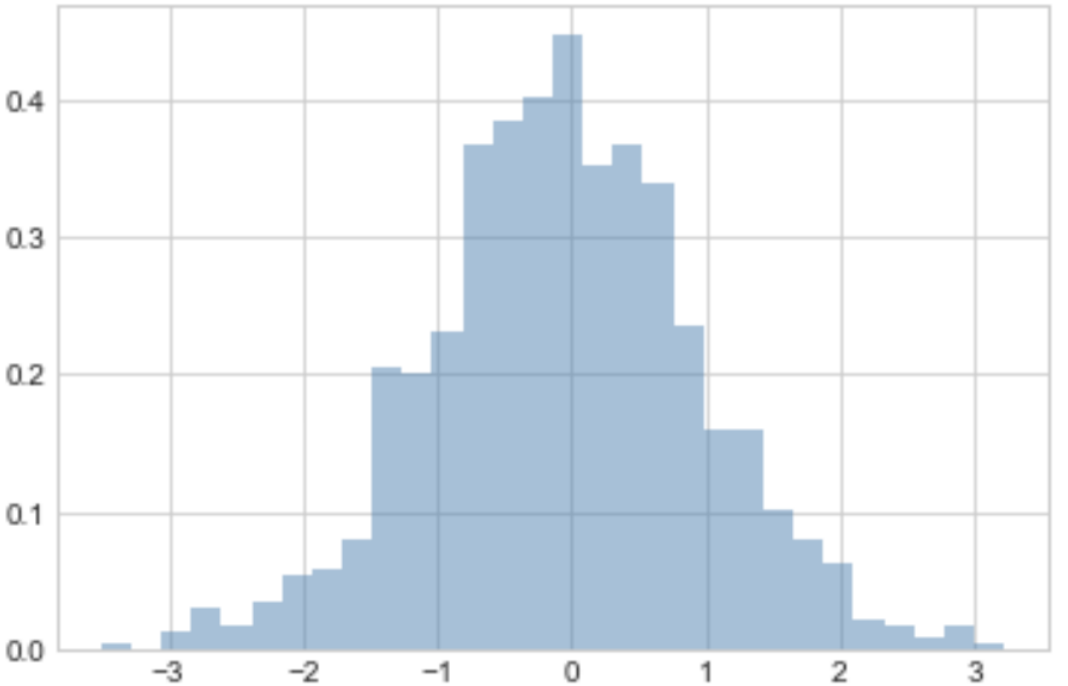
Histograms

In [28]:
```
data = np.random.randn(1000)
plt.hist(data);
```



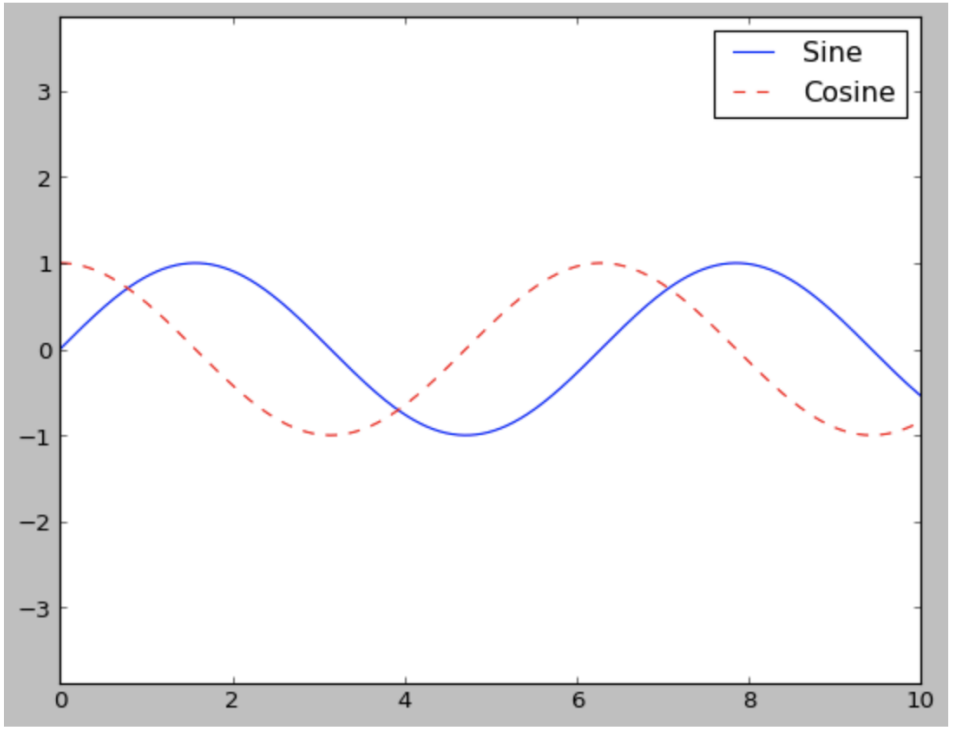
Customizing Histograms

In [29]:
```
plt.hist(data, bins=30, normed=True, alpha=0.5,
         histtype='stepfilled', color='steelblue',
         edgecolor='none');
```



Customizing Legends

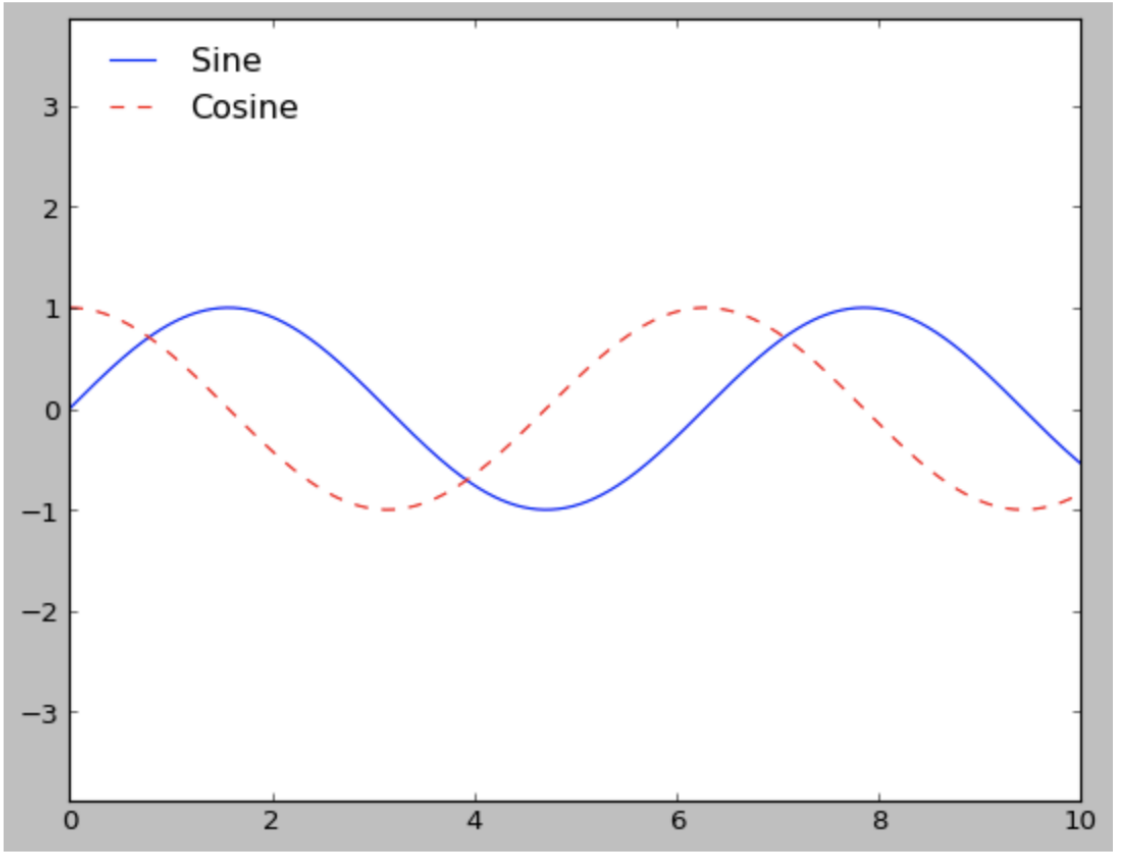
In [33]:
```
plt.style.use('classic')
x = np.linspace(0, 10, 1000)
fig, ax = plt.subplots()
ax.plot(x, np.sin(x), '-b', label='Sine')
ax.plot(x, np.cos(x), '--r', label='Cosine')
ax.axis('equal')
leg = ax.legend();
```



Customizing Legends

In [34]:
```
ax.legend(loc='upper left', frameon=False)
fig
```

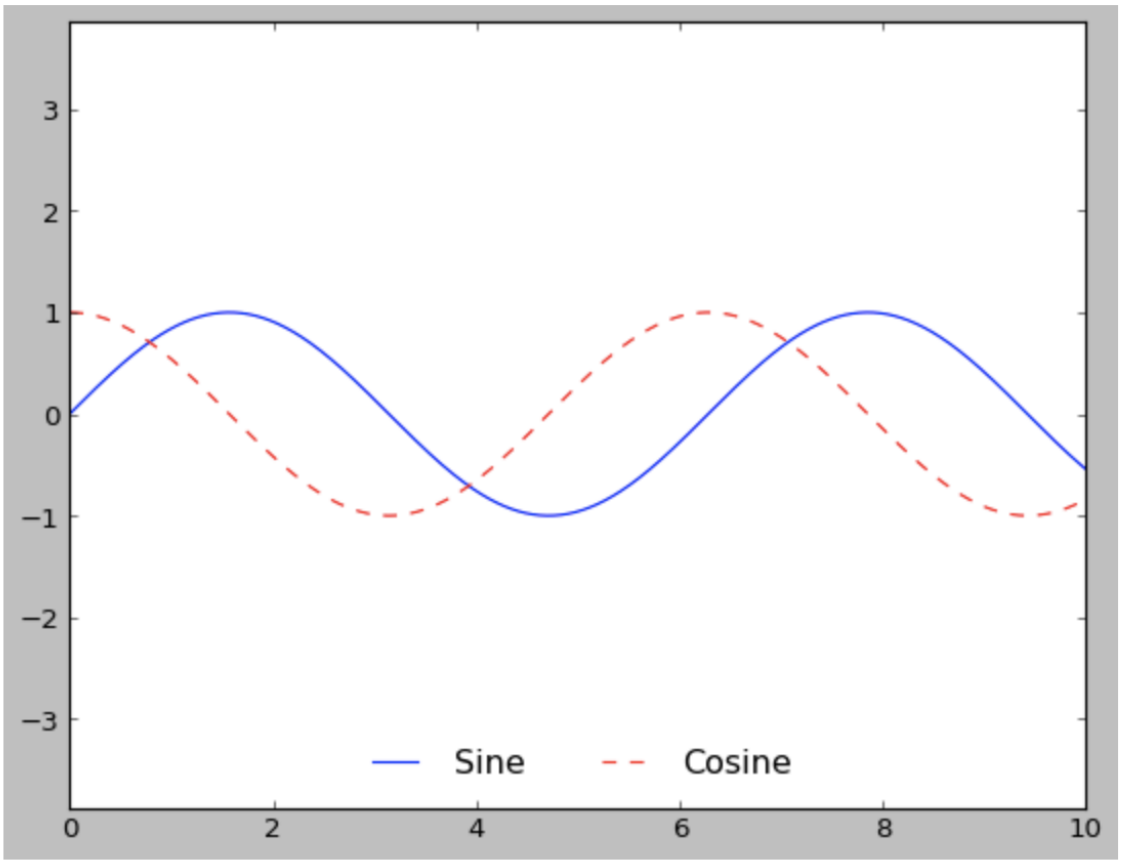
Out[34]:



Customizing Legends

In [35]:
```
ax.legend(frameon=False, loc='lower center', ncol=2)
fig
```

Out[35]:



Exercise: Data Visualization

(open the notebook named **`Exercise 5 - Data Visualization.ipynb`**)

Data Science

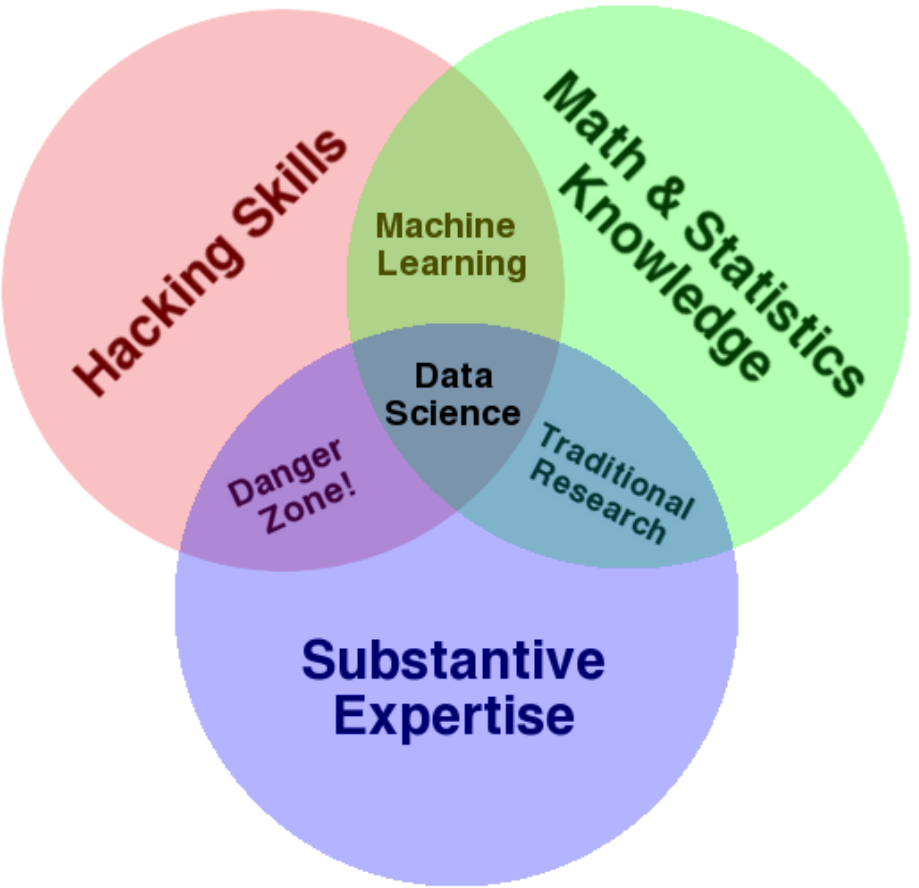
Objective

- Understand the various sides of Data Science
- Consider the activities of a data scientist
- Understand the issues of personal bias and narratives

"Data scientist: n. person who is better at statistics than any software engineer and better at software engineering than any statistician."

Josh Wills

Data Science

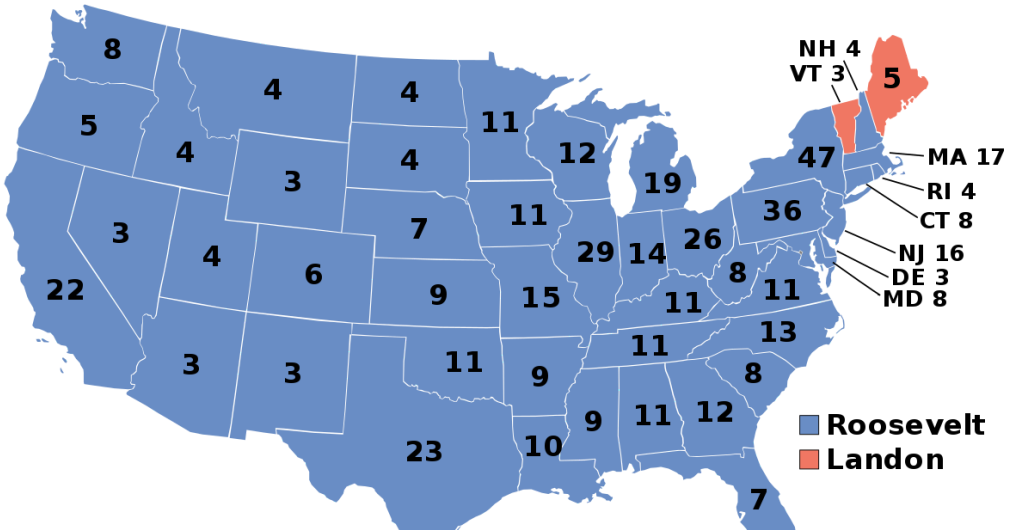


"Drew Conway"

Why is the discipline so important?

Famous example of the consequences of bad data: 1936 U.S. Presidential Election

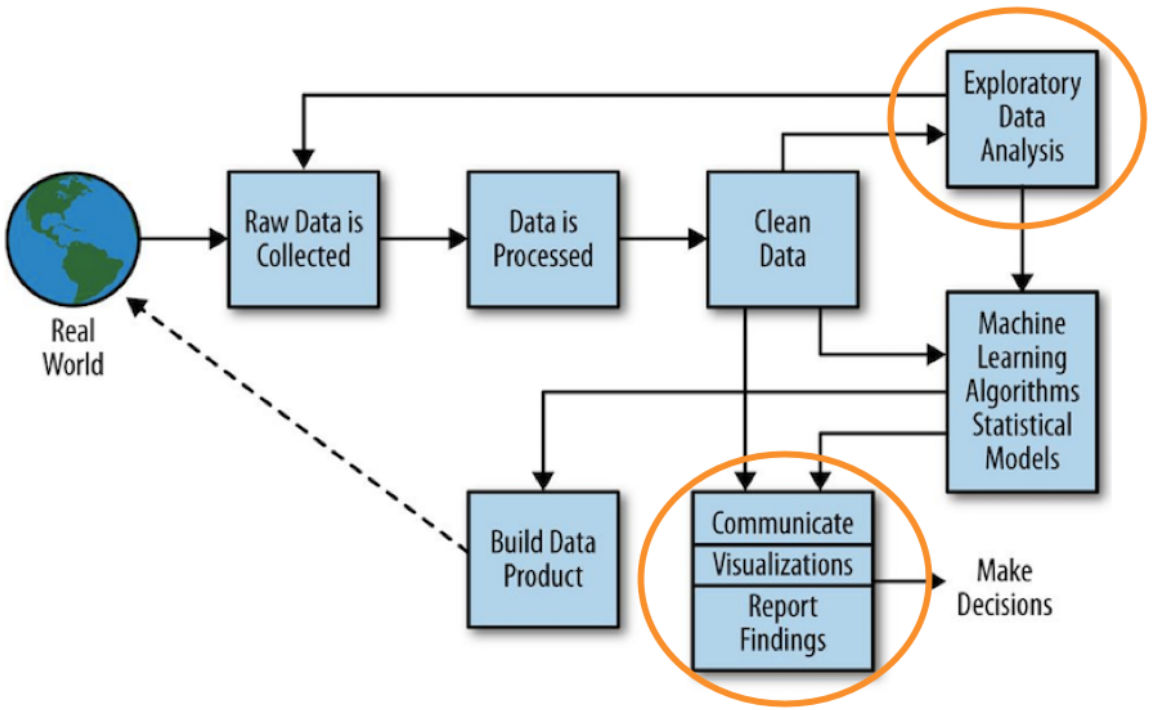
- Literary Digest conducted a mailed survey of 10 million (!) people in order to predict the winner of the 1936 U.S. presidential election
- participants chosen from telephone directories, lists of magazine subscribers, club rosters
- 2.5 million (!) people responded
- overwhelmingly believed Alf Landon would beat Franklin Roosevelt
- George Gallup polled 50,000 people and got it right…why?



"Long before worrying about how to convince others, you first have to understand what's happening yourself."

Andrew Gelman

Data Science Pipeline



"Doing Data Science"

"Naïve realism, also known as direct realism or common sense realism, is a philosophy of mind rooted in a theory of perception that claims that the senses provide us with direct awareness of the external world."

http://en.wikipedia.org/wiki/Naïve_realism

1951 Princeton/Dartmouth Football Game

- Storied rivalry
- Princeton's star player's nose was broken
- Another Princeton player retaliated and broke Dartmouth player's leg
- Princeton won (13-0)
- Both teams blamed the other side
- Two versions of the Truth

"They Saw a Game"

- Albert Hastorf (Dartmouth) and Hadley Cantril (Princeton) showed the game again to students from both schools
- Asked them to notice infractions, penalties, fill out a questionnaire
- Princeton students 'saw' twice as many infractions by Dartmouth players than Dartmouth students did
- Dartmouth students saw a 'rough but fair' game

"In brief, the data here indicate that there is no such 'thing' as a 'game' existing 'out there' in its own right which people merely 'observe.' The game 'exists' for a person and is experienced by him only insofar as certain happenings have significances in terms of his purpose."

Hastorf and Cantril

"Everything that has ever happened to you has happened inside your skull."

David McRaney, "You Are Now Less Dumb"

Compare the Students

- All male
- Mostly similar ethnically and socioeconomically
- Geographically similar
- Similar in age
- Similar basic cultural and religious beliefs
- Went to different schools

"It’s a real problem, though, when politicians, CEOs, and other people with the power to change the way the world works start bungling their arguments for or against things based on self-delusion generated by imperfect minds and senses."

David McRaney, "You Are Now Less Dumb"

"Narratives are meaning transmitters. They are history-preservation devices. They create and maintain cultures, and they forge identities that emerge out of the malleable, imperfect memories of life events."

David McRaney, "You Are Now Less Dumb"

"Your narrative bias makes it nearly impossible for you to really absorb the information from the outside world without arranging it into causes and effects."

David McRaney, "You Are Now Less Dumb"

"Your ancestors invented the scientific method because the common belief fallacy renders your default strategies for making sense of the world generally awful and prone to error."

David McRaney, "You Are Now Less Dumb"

"Your natural tendency is to start from a conclusion and work backward to confirm your assumptions, but the scientific method drives down the wrong side of the road and tries to disconfirm your assumptions."

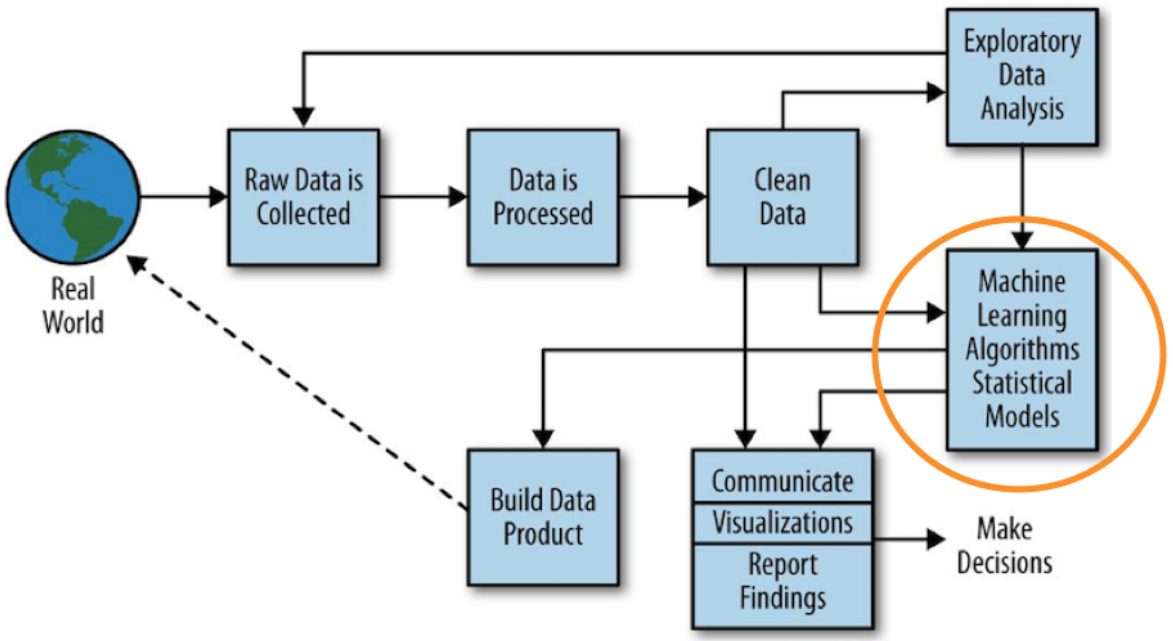
David McRaney, "You Are Now Less Dumb"

Data-Directed

Objective

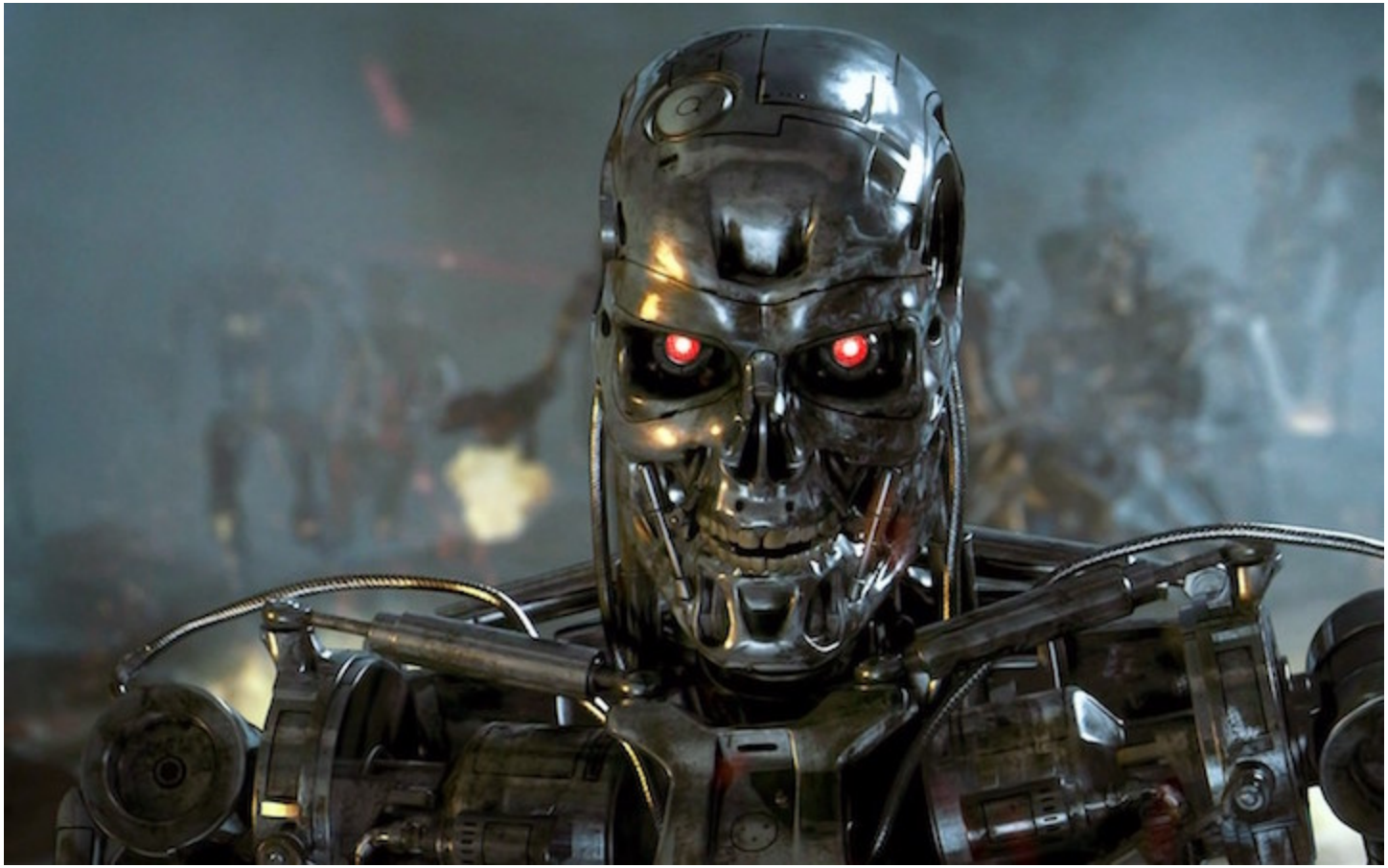
- Understand the basics of machine learning and how it can direct our activities
- Consider the issues of generalizing from observations of the world
- Explore *supervised* and *unsupervised* learning strategies
- Be introduced to a handful of basic and widely-used techniques
- Learn how to use them on basic problems

Data Science Pipeline



"Doing Data Science"

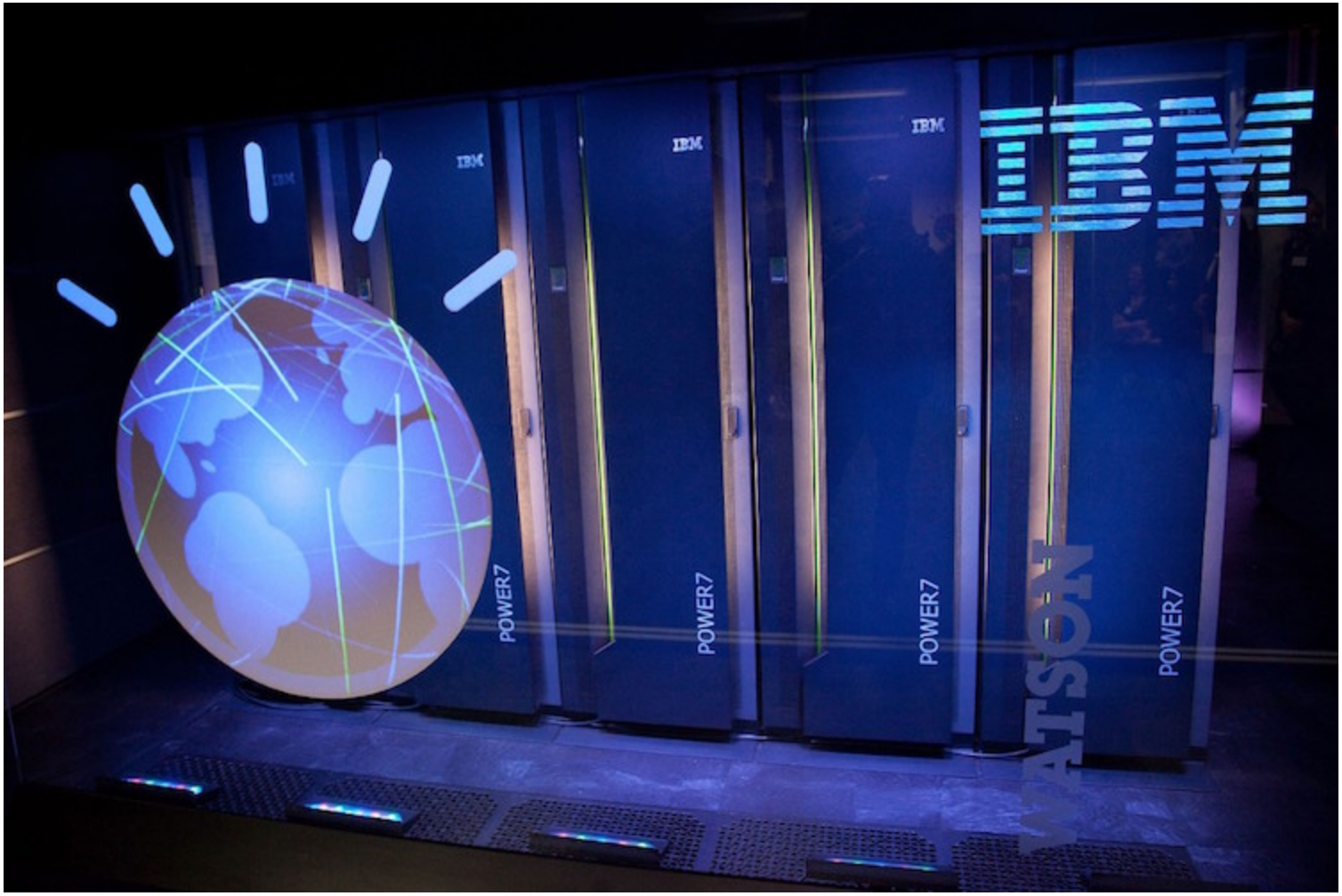
https://www.youtube.com/watch?v=s93KC4AGKnY



https://youtu.be/toK10SLep3s

Skin Cancer Image Classification (TensorFlow Dev Summit 2017)

https://youtu.be/rVlhMGQgDkY?t=85



https://www.youtube.com/watch?v=WFR3lOm_xhE&t=20

B.F. Skinner

Don't train your models to be superstitious!

"The term machine learning refers to the automated detection of meaningful patterns in data."

Source: Shavel-Shwartz and Ben-David, "Understanding Machine Learning: From Theory to Algorithms"

Advancement of Machine Learning

- Stock market prediction in the 1980s
- Mining corporate databases in the 1990s (direct marketing, CRM, credit scoring, fraud detection)
- E-commerce (personalization, click analysis)
- 9/11 brought interest to applying ML to fighting terror
- Web 2.0 (social networks, sentiment analysis, etc.)
- Science (molecular biologists and astronomers were early adopters)
- Housing bust freed up a lot of talent
- Big Data

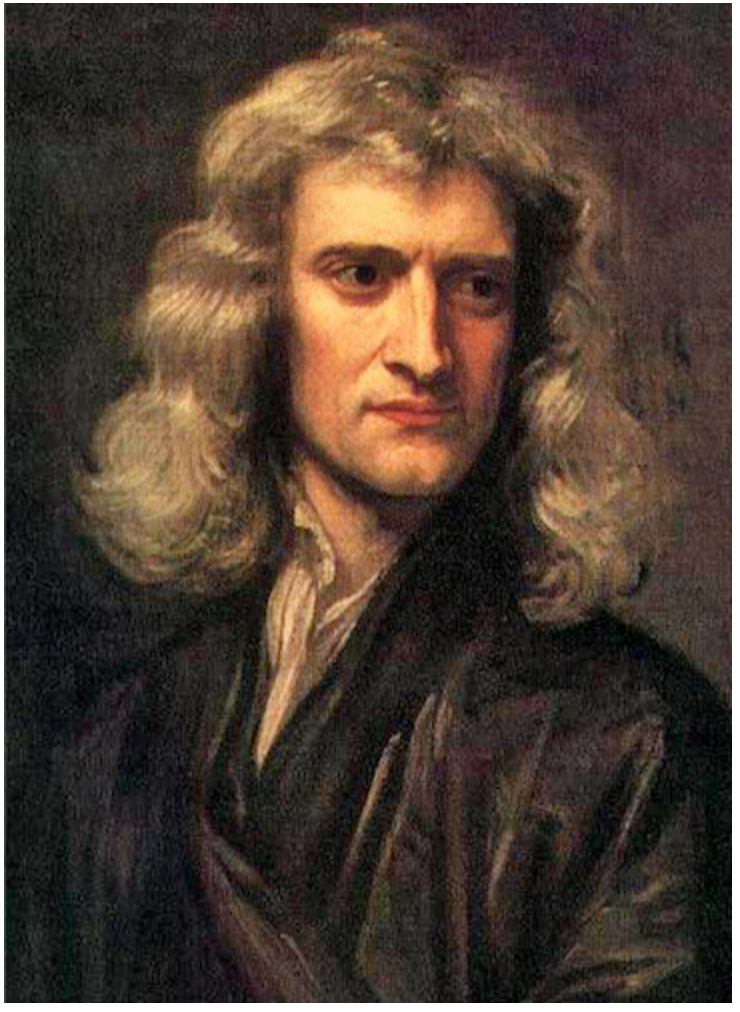
David Hume

Inductivist Turkey

Tycho Brahe

Johannes Kepler

Isaac Newton



Machine Learning Approaches

Supervised Learning

Given input variables x and an output variable Y

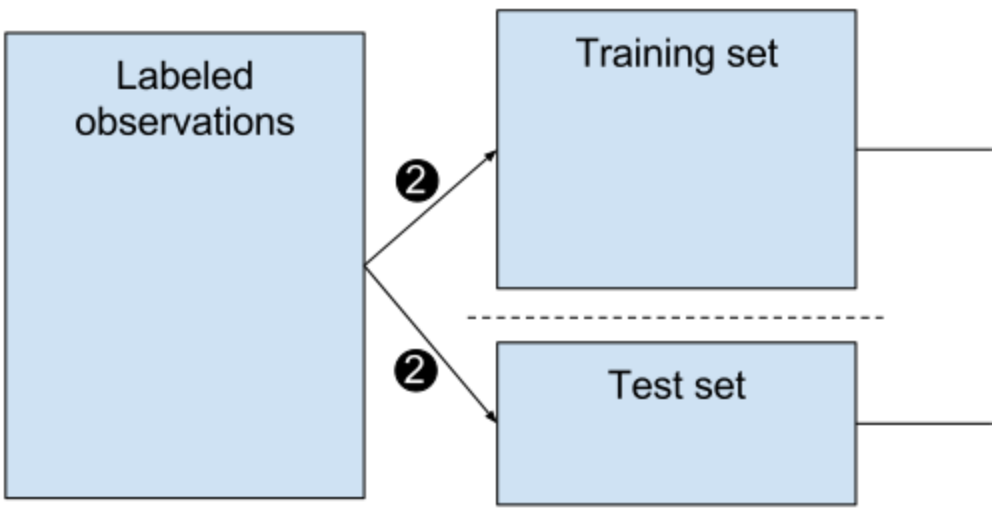
$$Y = f(x)$$

To train the model, the inputs and outputs are known and are used to determine f(x). The trained model f(x) outputs one of two types of outputs: classification and regression.

- Classification, i.e., classifying input into two or more categories (e.g., benign vs. malignant tumors)
- Regression - the output variable takes continuous values (e.g., how much would we expect this house to sell for, based on square footage, location, etc.)

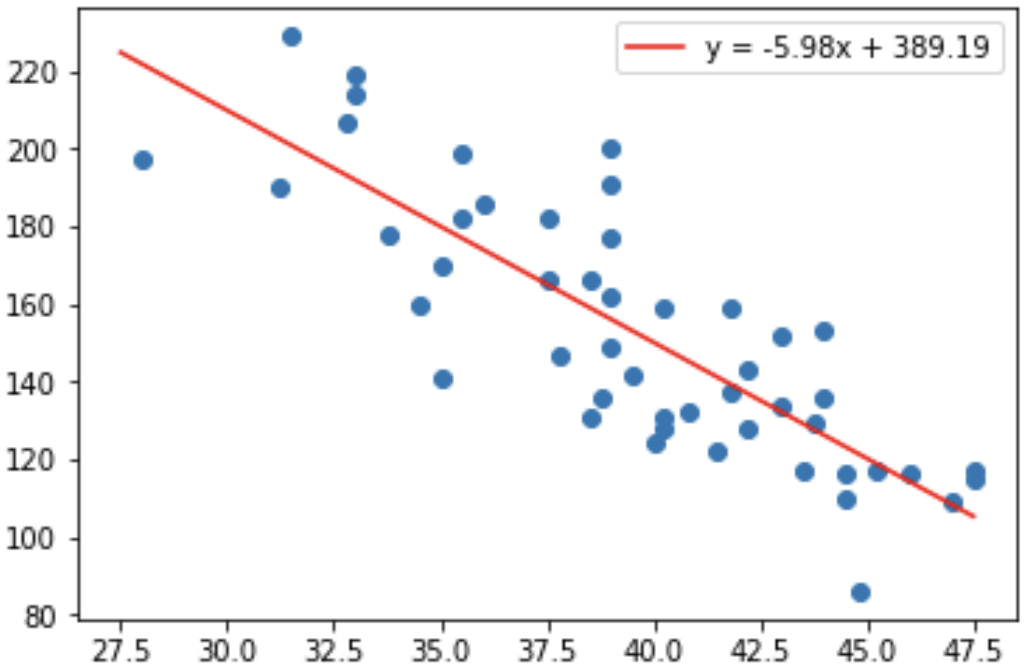
Supervised Learning

partition dataset into training set and test set



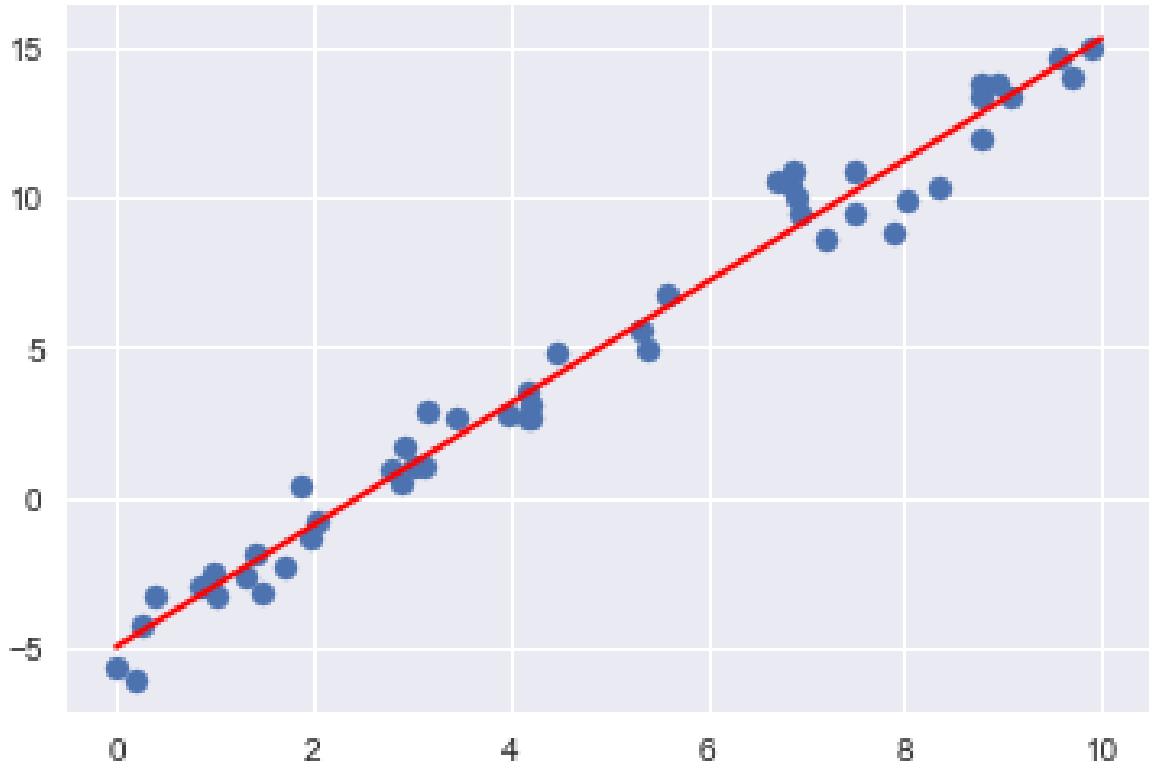
Linear Regression

relating your input (sometimes called "independent variable") to your output ("dependent variable") by fitting a straight line through your data



Ordinary Least Squares

- Simple Linear Regression
- Fit a straight line through the observed points
- Minimizes the sum of square residuals (errors) of the model



Hypothetical Business: The Zappos of Pants

- Where is the business risk?
  - too many returns
- If customers don't know their inseam, it'll be hard for them to measure
- We'd like to ask questions of our customers and use that as a proxy for their inseam
  - ...but we don't know what to ask
- So we gather a bunch of data from a population, which needs to be representative of the population we want to market to
- ...then what?
  - look for a relationship between one of the variables and inseam

Linear Regression

Pros	Cons
Common approach for numeric data	Strong assumptions about the data (linearity)
Can handle most modeling tasks	Sensitive to outliers
Estimates strength and size of relationships among features and outcomes	Only numeric features

Demo: Linear Regression

Demo: Linear Regression

- let's open the notebook named **`Demo - Linear Regression.ipynb`** and go through it together

Exercise: Linear Regression

(open the notebook named **`Exercise 6 - Linear Regression.ipynb`**)

Naive Bayes

Naive Bayes

- Family of algorithms to produce probabilistic classifiers based on Bayes Theorem
- Bayes Theorem is named after the Reverend Bayes an 18th century statistician and philosopher
- Never published in his lifetime

Features

- Determines probability based upon context, and the probabilities are updated as more information is received
- Starting probabilities can be arbitrarily set
- Requires relatively little training data
- Often used for text/document classification
- Assumes independence of the features

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A) \cdot P(A)}{P(B)}$$

$$P(spam|Viagra) = \frac{P(Viagra|spam) \cdot P(spam)}{P(Viagra)}$$

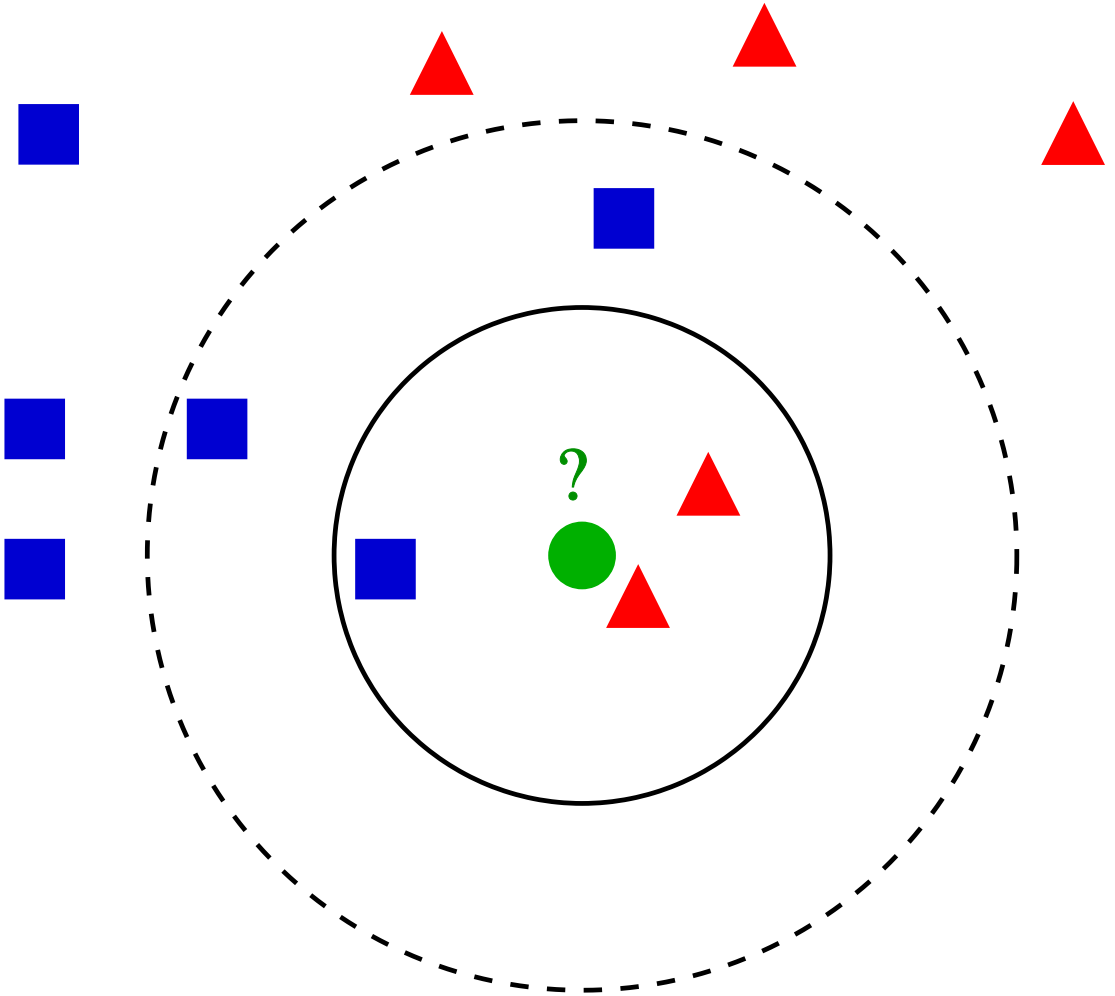
Naive Bayes

Pros	Cons
Simple and effective	Assumption of the independence of features is usually wrong
Does well with noisy and missing data	Doesn't work well with lots of numeric features
Works well with arbitrary sizes of training data	Estimated probabilities aren't as reliable as the classifications
Easy to produce the estimated probability for predictions	

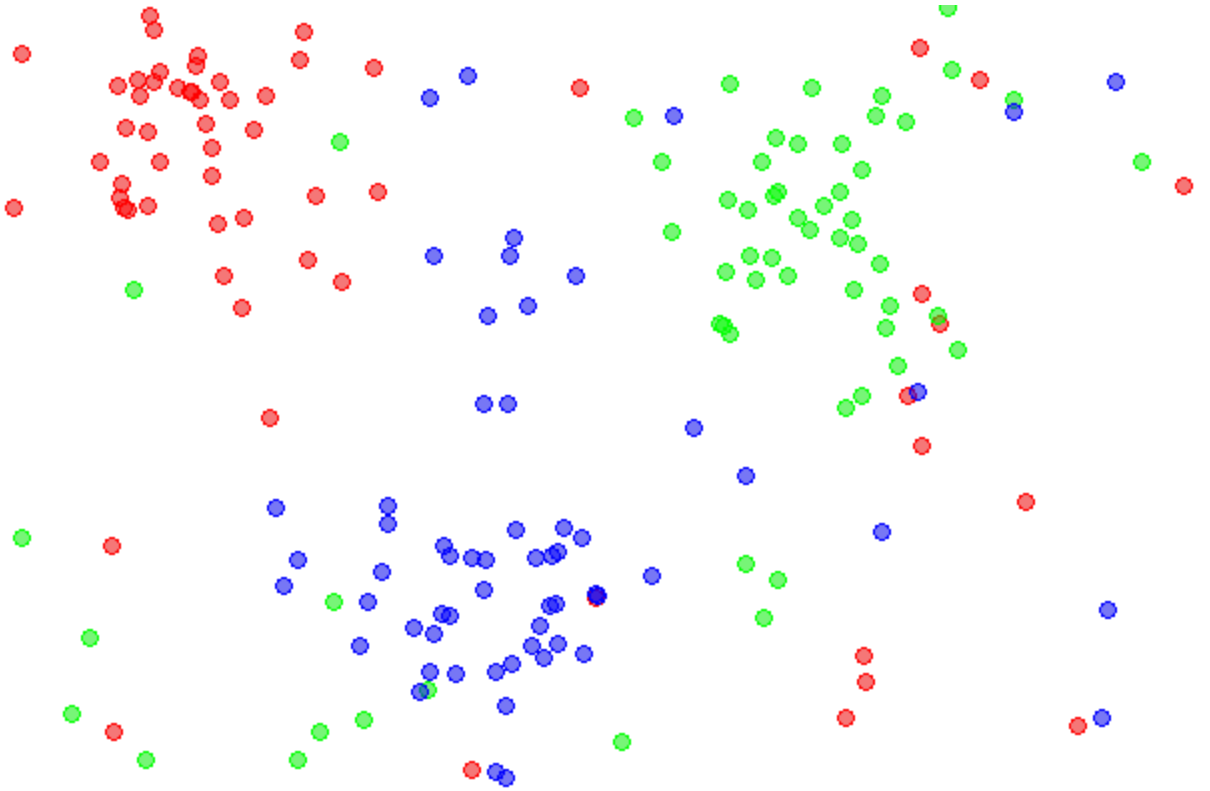
k-Nearest Neighbors

k-Nearest Neighbors

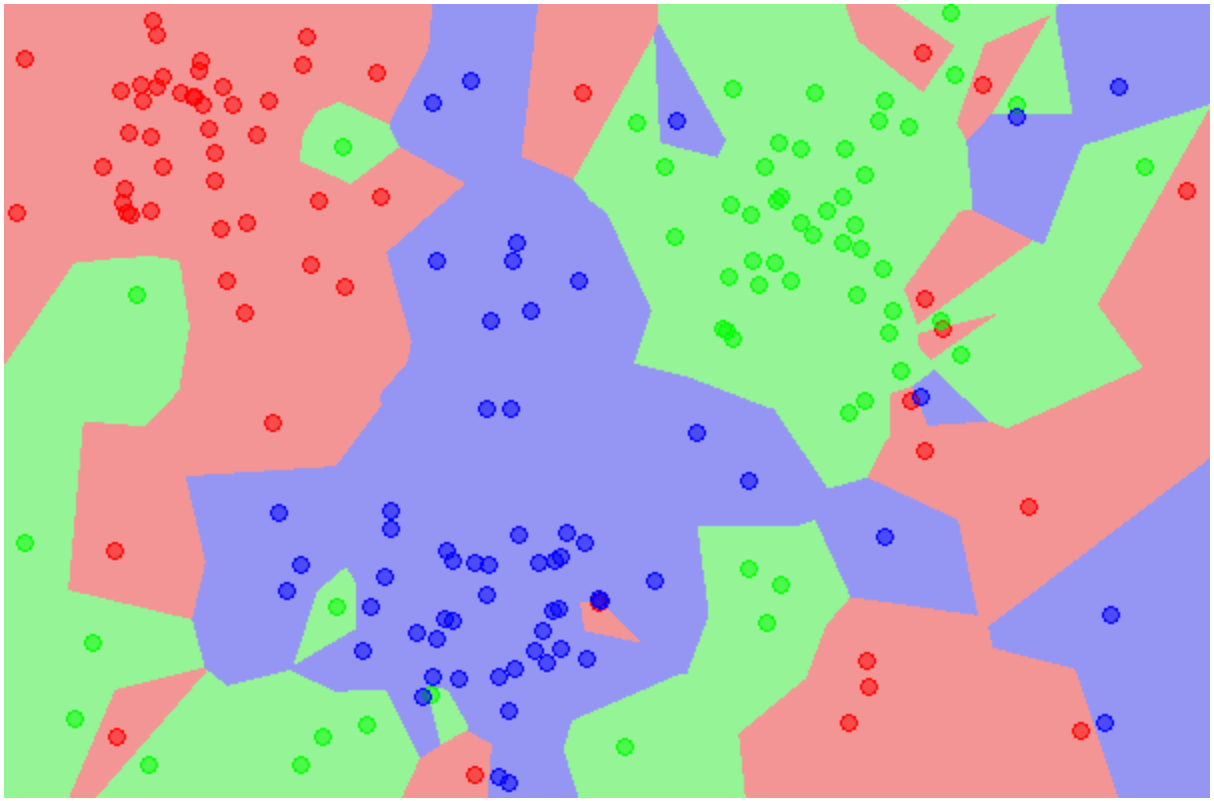
- k-NN Classification
  - output is class membership
  - object is classified by a majority vote of its neighbors
  - for k = 1, then the object is simply assigned to the class of that single nearest neighbor
- k-NN Regression
  - output is the property value for the object
  - this value is the average of the values of its k nearest neighbors



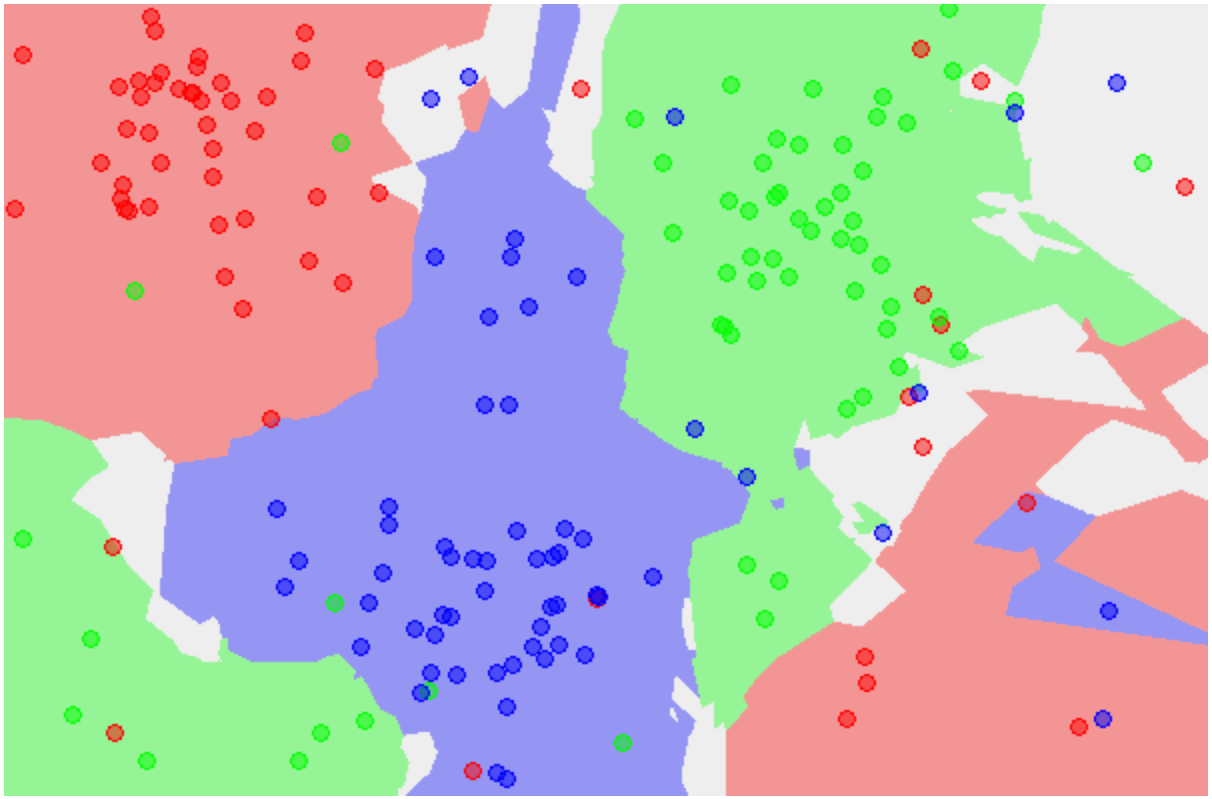
http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm#/media/File:KnnClassification.svg



http://commons.wikimedia.org/wiki/File:Data3classes.png



http://commons.wikimedia.org/wiki/File:Map1NN.png



http://commons.wikimedia.org/wiki/File:Map5NN.png

k-Nearest Neighbors

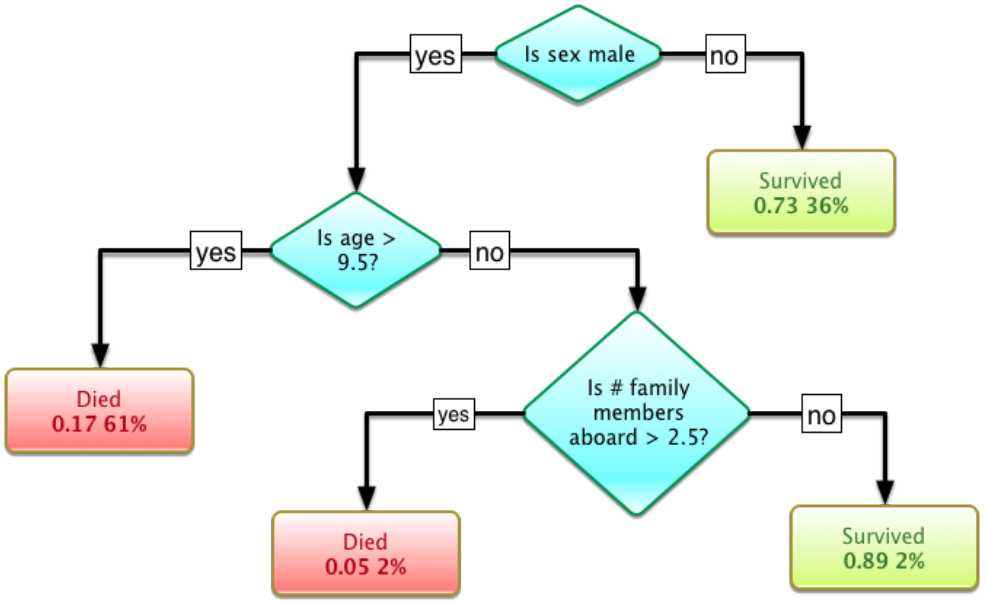
Pros	Cons
Simple and effective	Does not produce a model, but does produce a function
Makes no assumptions about the data distribution	Efficacy affected by choice of k
Fast training phase	Slow classification phase

Decision Trees

Decision Trees

- Tree-based classifier (like a bunch of *if-then* statements)
- Models the relationships between features and outputs
- Easy to explain to users
- Can be turned into external representation beyond the classifier
- Supports both classification and regression
- Builds a tree where each node divides the set of items using a feature and feature value
- The feature and feature value are chosen based upon which one "best" splits the set of items
- The "best" split can be determined by several approaches, two common ones are: *gini impurity* (default in SciKit-Learn) and *information gain*
    - Gini Impurity seeks to maximize the homogeneity of the subnodes
    - Information Gain seeks to minimize the entropy of the subnodes
    - In-depth coverage of these approaches is deferred to the Applied ML Class

Decision Tree: Titanic Dataset



Decision Tree

Pros	Cons
Useful classifier for most problems	Can be biased toward feature splits with several levels
Automated learning process	Easy to misfit the model
Supports numeric, nominal and missing data	Small changes in the training data can have been impact on decision logic
Works with large and small data sets	Large trees may be hard to interpret
Easily interpreted and efficient	

Demo: Decision Trees

- there are screenshots in this presentation to maintain continuity, but let's open the notebook named **`Demo - Decision Trees.ipynb`** and go through it together
- when done, click here to skip screen shots

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
```

```
iris = load_iris()
X = iris.data[:, 2:]
y = iris.target
```

```
tree_clf = DecisionTreeClassifier(max_depth=2)
tree_clf.fit(X, y)
```

```
from sklearn.tree import export_graphviz
export_graphviz(tree_clf, out_file="iris_tree.dot",
                feature_names=iris.feature_names[2:],
                class_names=iris.target_names,
                rounded=True,
                filled=True)
```

```
# dot -Tpng iris_tree.dot -o iris_tree.png
```

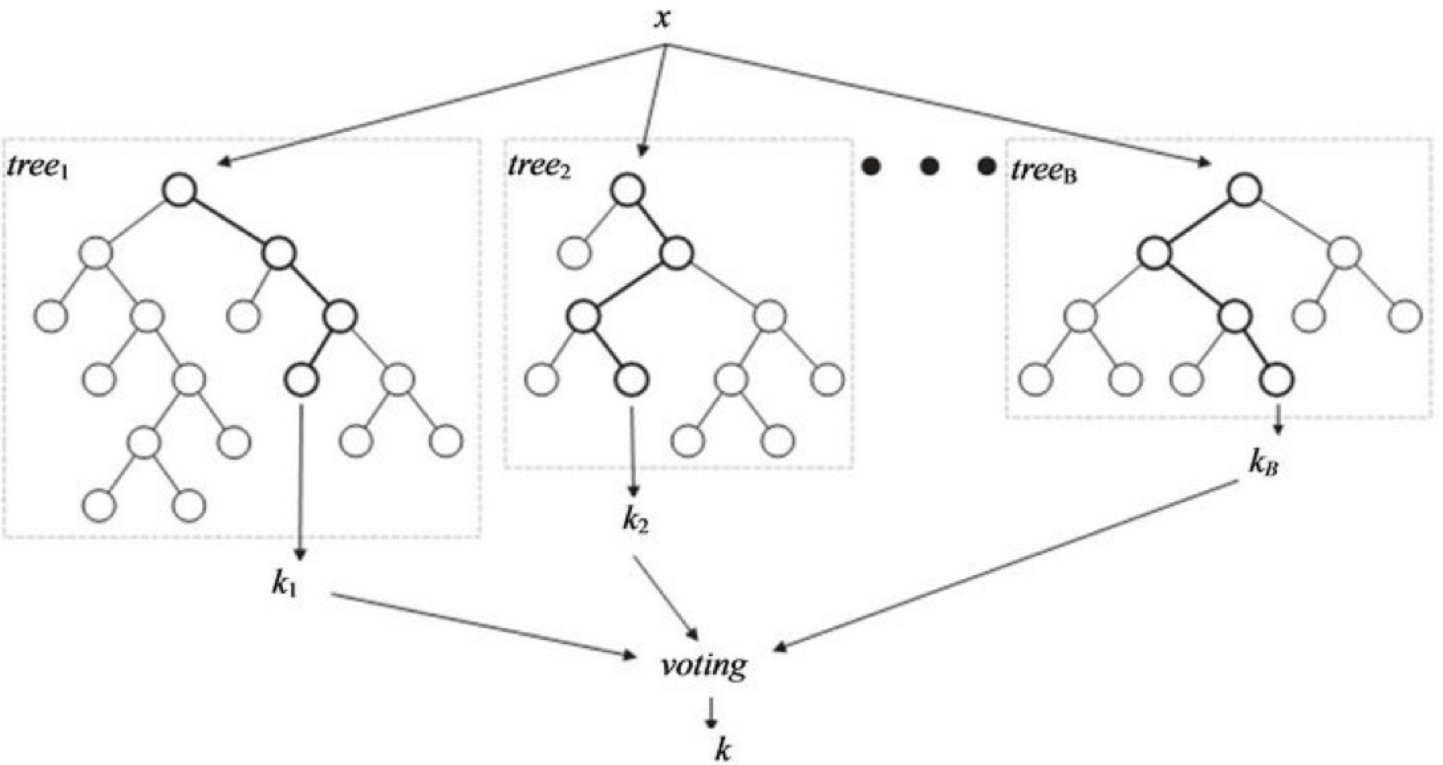
```
tree_clf.predict_proba([[5, 1.5]])
```

```
tree_clf.predict([[5, 1.5]])
```

Exercise: Decision Trees

(open the notebook named **`Exercise 7 - Decision Trees.ipynb`**)

Random Forests



http://file.scirp.org/Html/6-9101686_31887.htm

Demo: Random Forests

- there are screenshots in this presentation to maintain continuity, but let's open the notebook named **`ML - Decision Tree 2.ipynb`** and go through it together

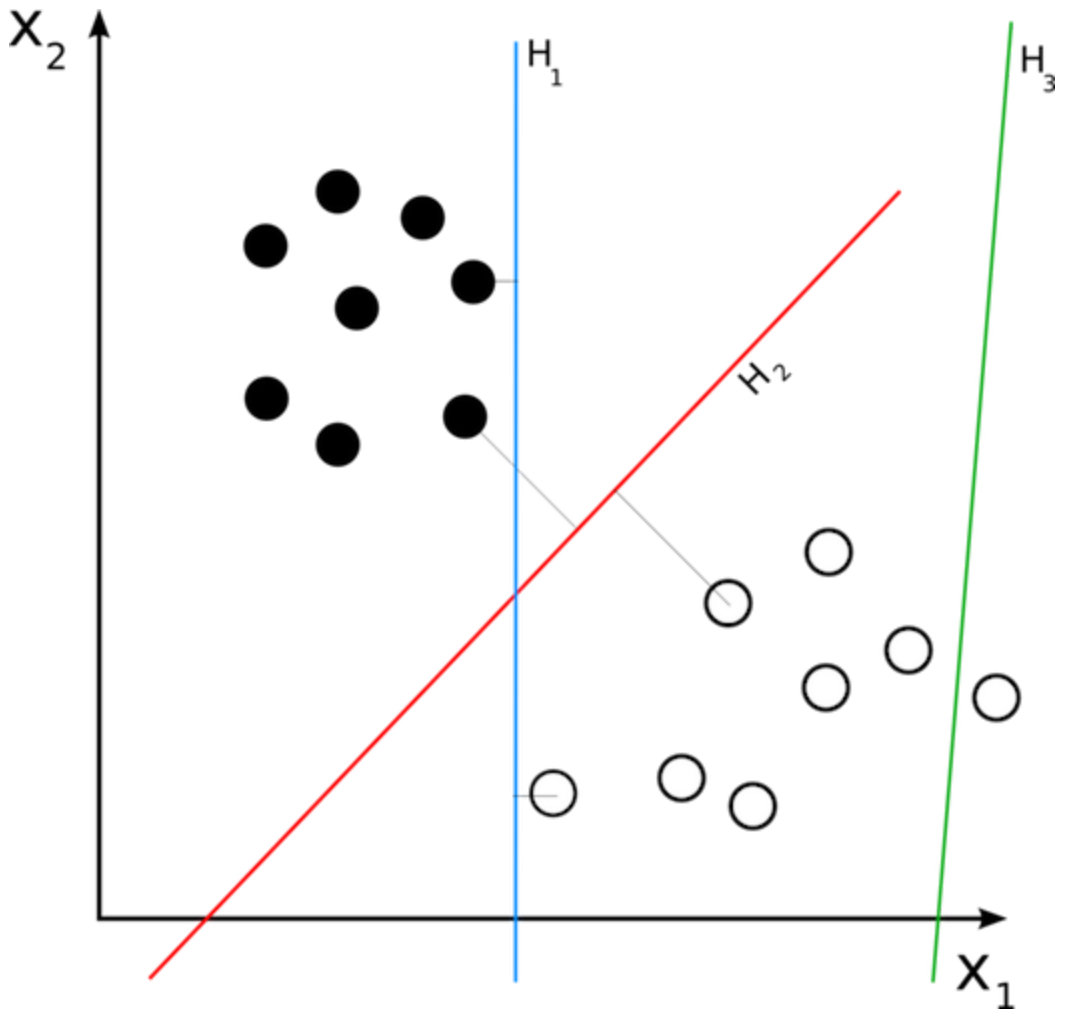
Random Forests

Pros	Cons
Supports both classification and regression.	Computationally expensive.
Averages out potential bias from Decision Trees.	Less interpretable than Decision Tree.

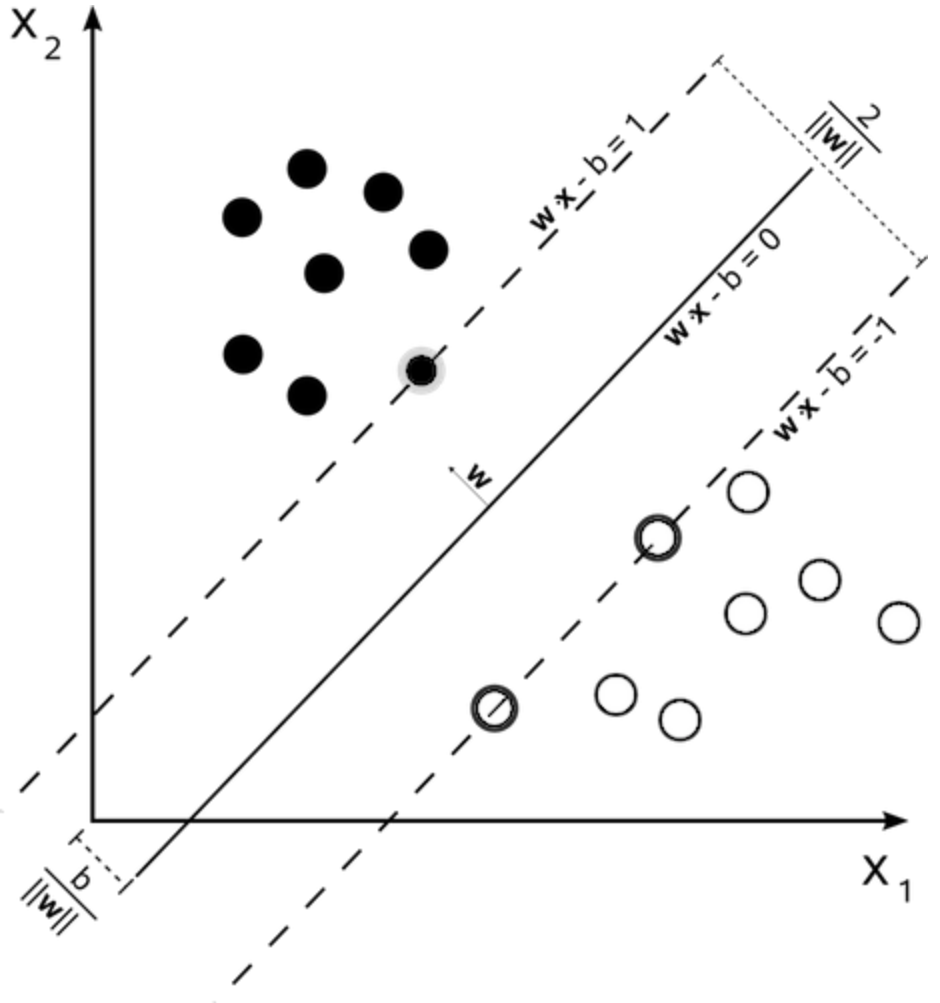
Support Vector Machines

"[a] support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks"

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



https://en.wikipedia.org/wiki/Support_vector_machine



https://en.wikipedia.org/wiki/Support_vector_machine

Demo: Support Vector Machines

- let's open the notebook named **`ML - Support Vector Machine.ipynb`** and go through it

Support Vector Machines

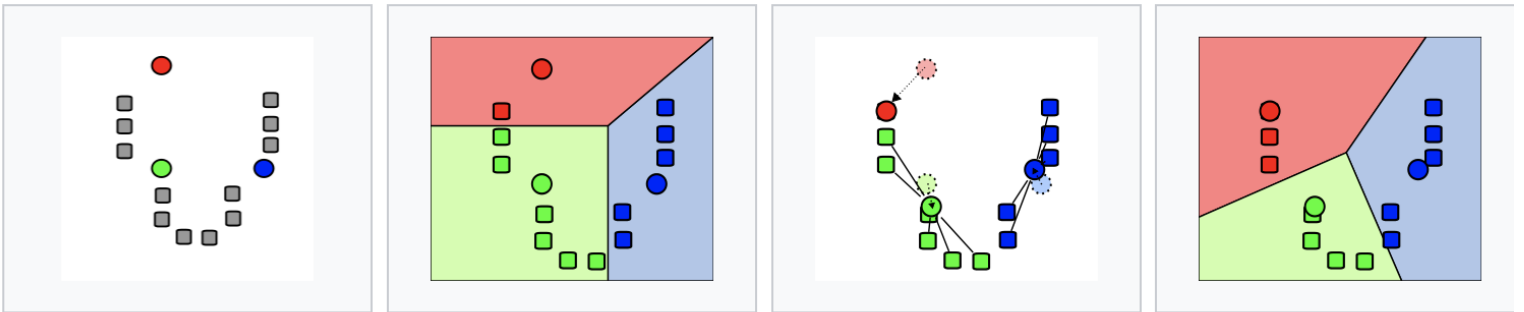
Pros	Cons
Similiar performance to logistic regression on linear boundaries.	Kernel choice can make it susceptible to overfitting.
Can handle non-linear boundaries.	Hard to interpret in high dimensions.
Handles high dimensional data	

Unsupervised Learning

Given only input variables x, find some underlying structure.

- Clustering
- Association rules

k-Means Clustering



- Select k-means randomly
- Associate every cluster with a mean
- The centroid of each cluster becomes the new mean
- Repeat until convergence

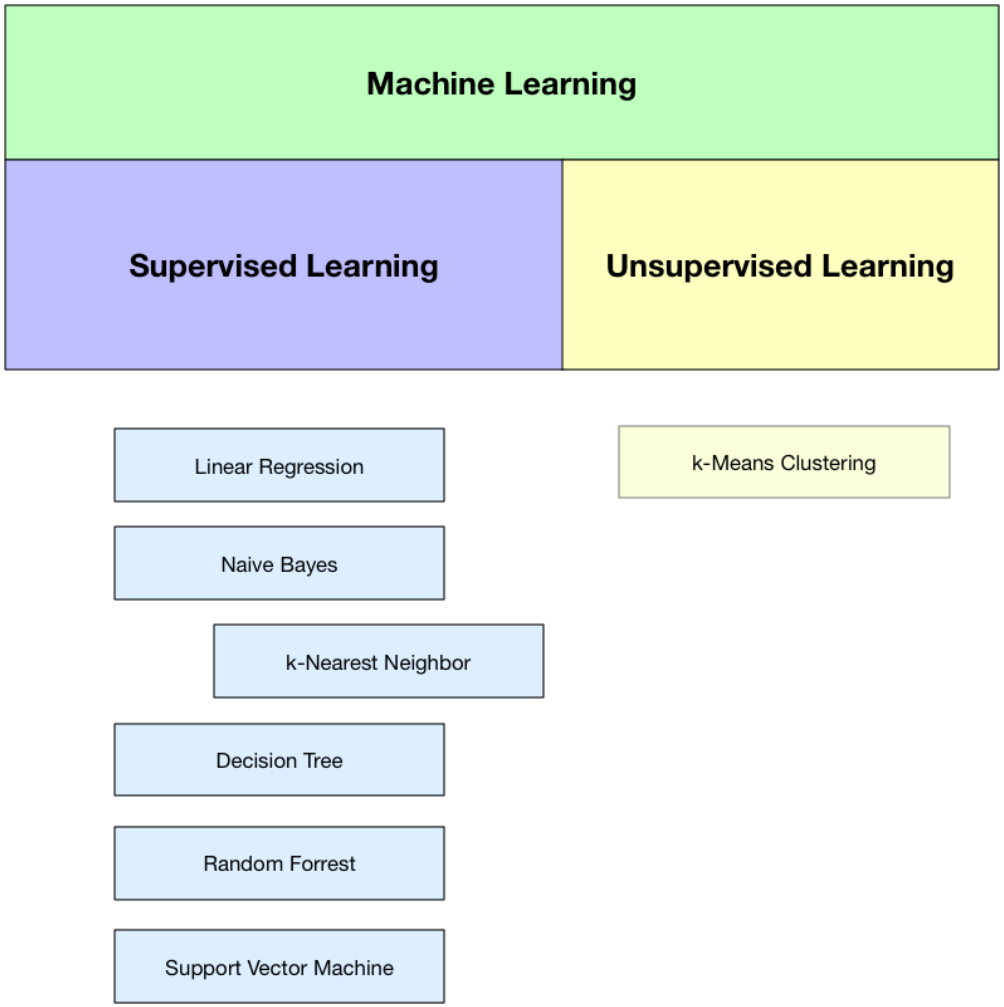
https://en.wikipedia.org/wiki/K-means_clustering

k-Means Clustering

Pros	Cons
Linear complexity O(n)	Random initialization may impact repeatability
Simple Euclidean distance calculation	Clusters don't really mean anything

Demo: k-Means Clustering

- let's open the notebook named **`ML - k-Means Clustering.ipynb`** and go through it


Machine Learning
Supervised Learning
Unsupervised Learning
Linear Regression
Naive Bayes
k-Nearest Neighbor
Decision Tree
Random Forrest
Support Vector Machine
k-Means Clustering

Infrastructure Demo

Objective

- Explore a real data science and machine learning activity at SalesForce

"

Our customers trust that the infrastructure where they host their data is reliable and that we proactively assess its health so that there are minimal impacts on customer experience. In order to ensure a good customer experience, we need to assess the risk of all pods in our infrastructure with enough lead time for mitigation or management of risks. An important metric in determining the risk of a pod is its Time-To-Live (TTL). The TTL determines when a pod will need remediation.

"

Pod Risk Assessment

What's so hard about this?

- System is
  - Multi-tier (app, db, search, storage)
  - Multi-tenant

Definitions

Org (Organization) – A deployment of Salesforce with a defined set of licensed users. Your organization includes all data and metadata, standard and custom objects, applications and security access. Your org is separate from all other orgs.

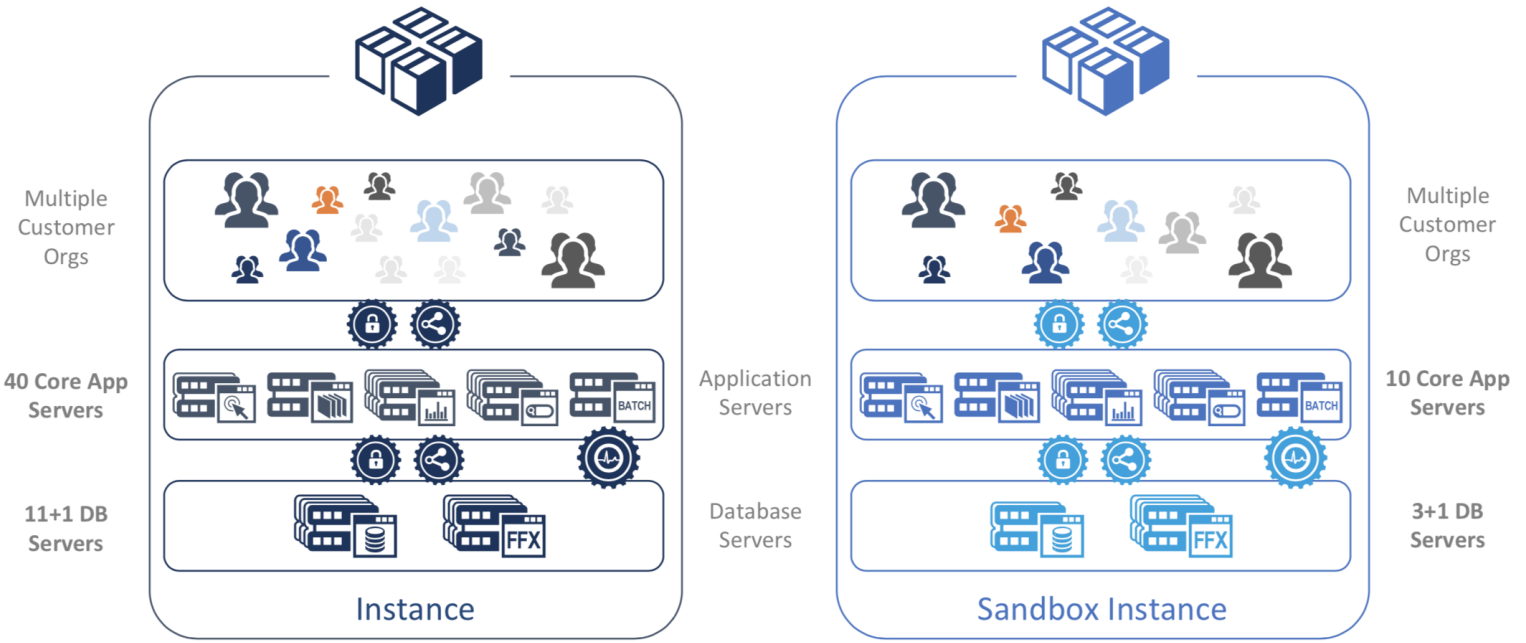
Core Instance – An instance is the collection of systems that provide service for a customer's Org. Salesforce's innovative multitenancy allows each Instance to support thousands of Orgs. There are two Instance types – Core (Production), Sandbox

Sandbox Instance – A sandbox allows you to create multiple copies of your Org in separate environments for development, testing, or training purposes without compromising the data and applications in your Salesforce product Org.

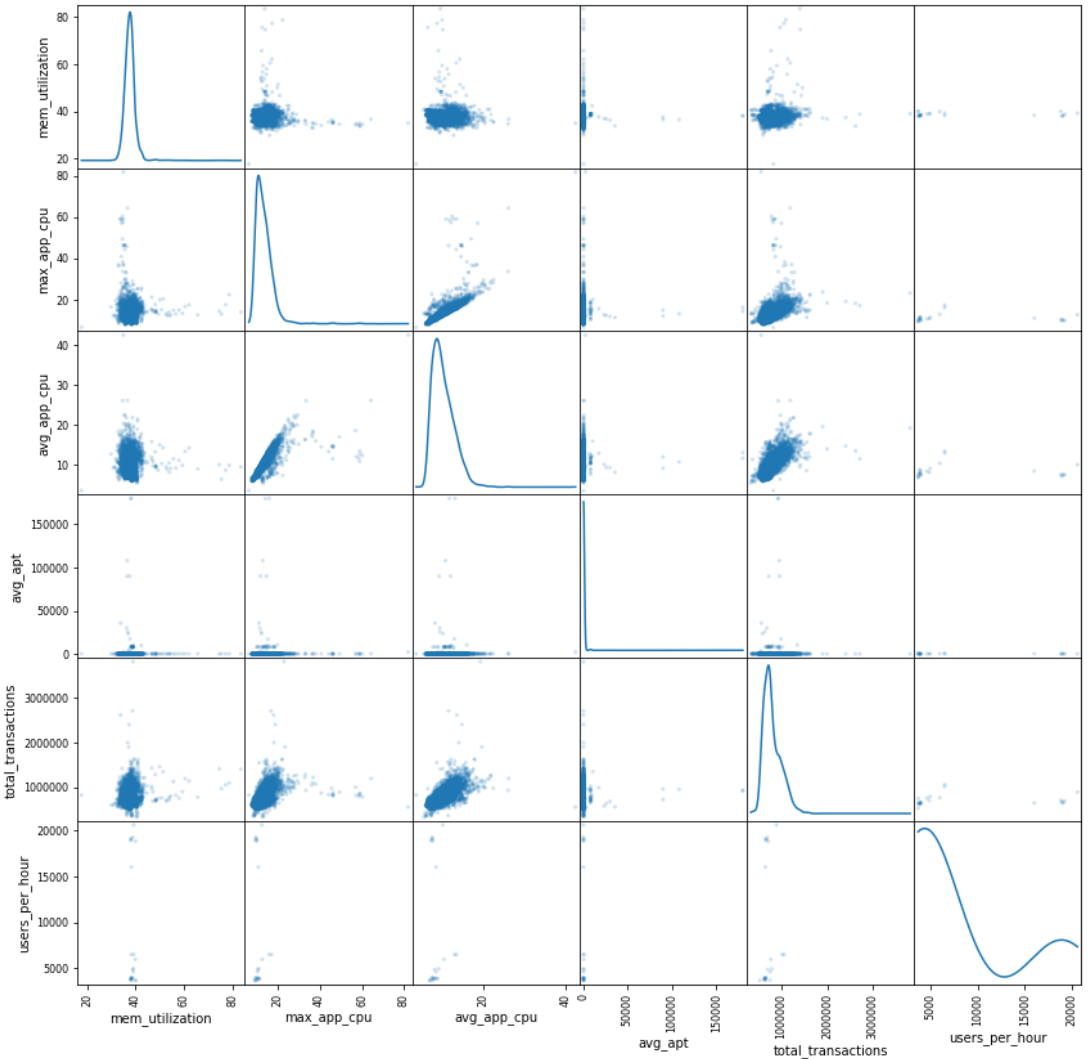
Instance Group – An Instance Group is a group of Instances that are supported by the same set of shared services (network and data) to provide additional service isolation within a single Data Center. Instance Groups can contain varying numbers of Core and Sandbox Instances.

Data Center – A Data Center is the physical location that houses one or more Instance Groups. It also includes the redundant shared services (power, cooling, networking) required to operate the Instance Groups and the Instances within those groups.

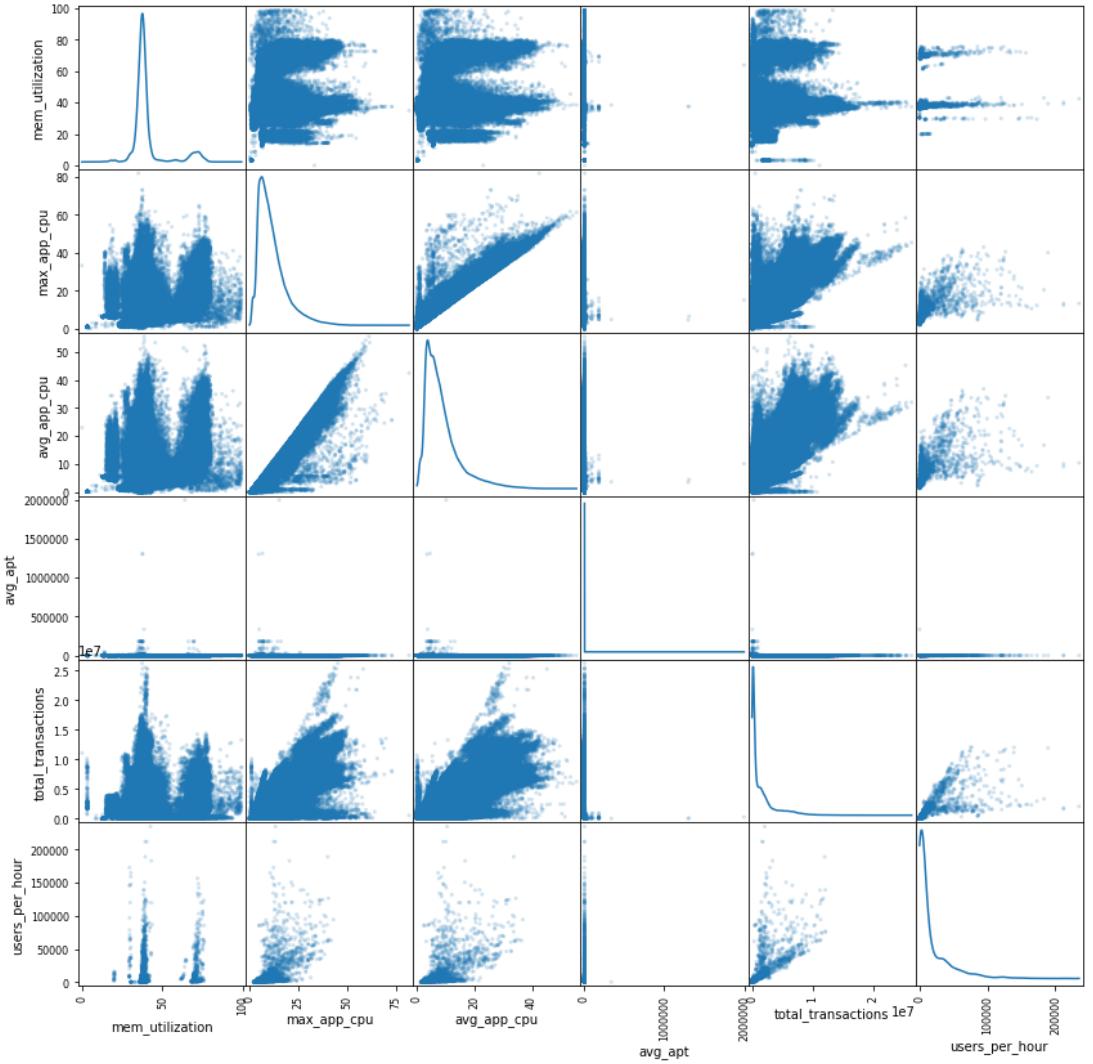
Core and Sandbox Instances



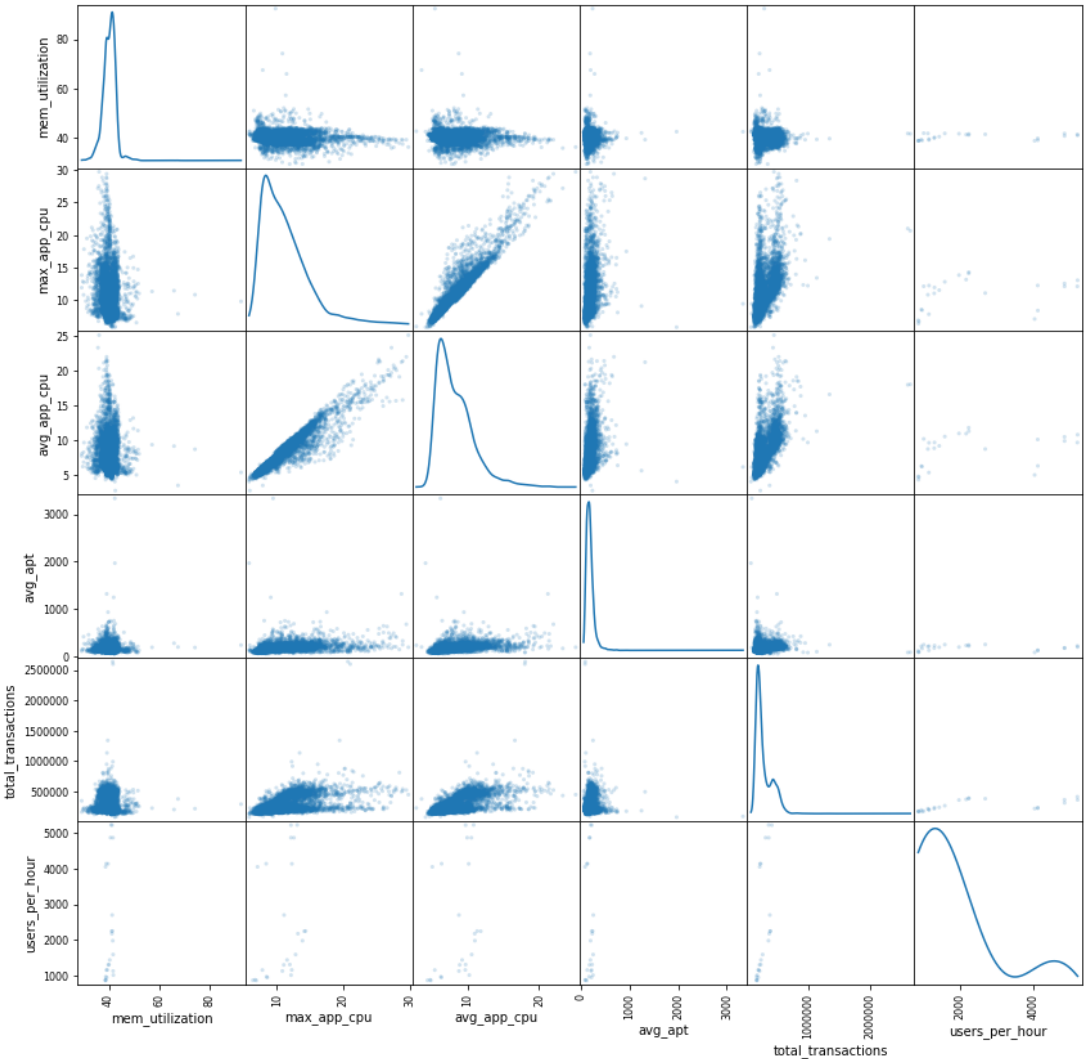
scatter-matrix for datacenter: CHI, pod: gs0, superpod: SP1



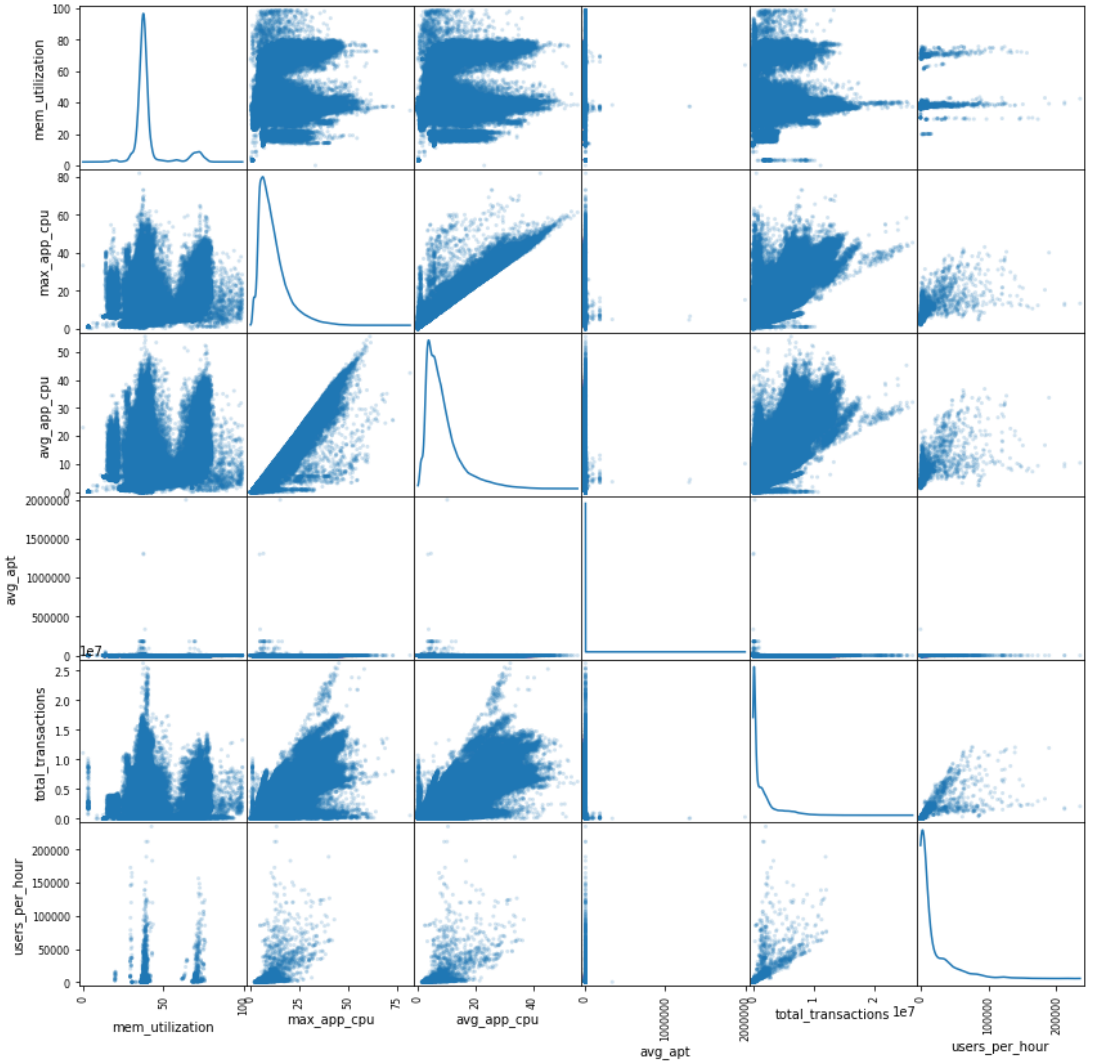
scatter matrix for Metrics List: mem_utilization,max_app_cpu,avg_app_cpu,avg_apt,total_transactions,users_per_hour



scatter-matrix for datacenter: LON, pod: cs86, superpod: SP9



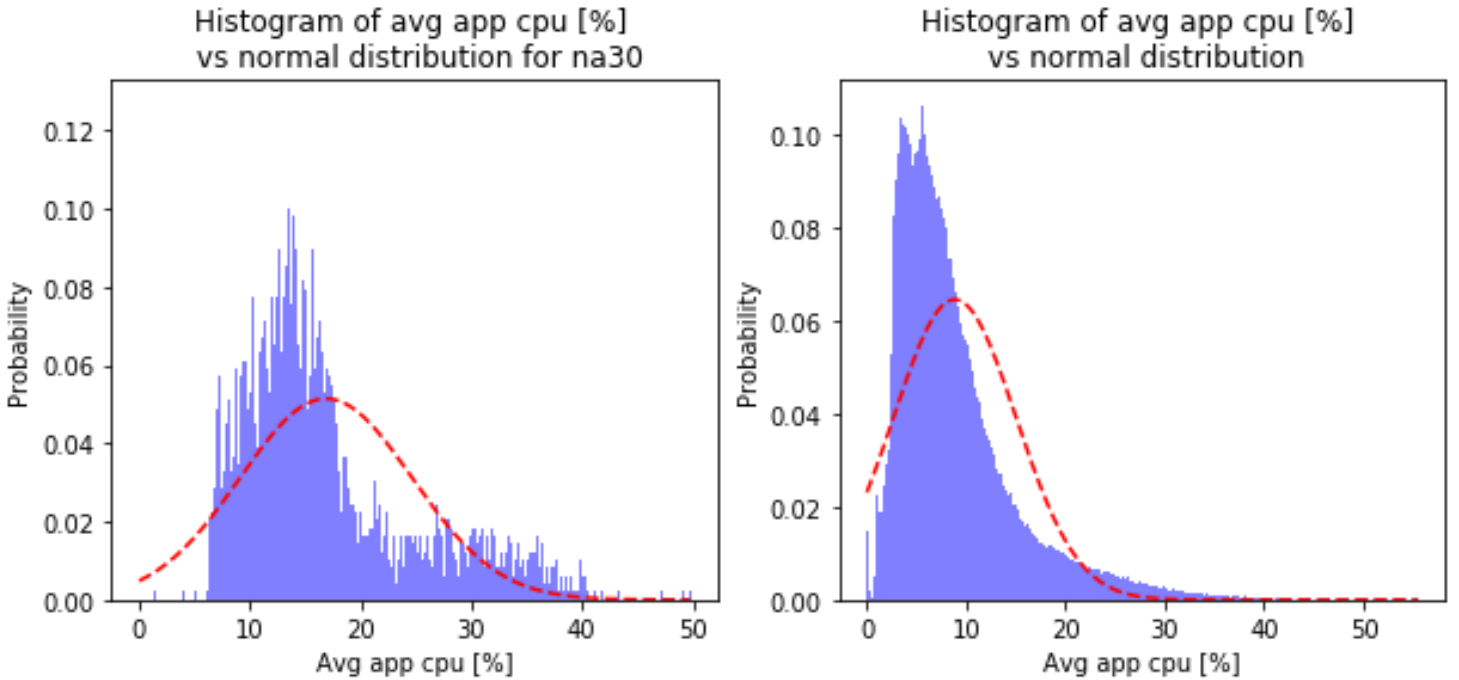
scatter matrix for Metrics List: mem_utilization,max_app_cpu,avg_app_cpu,avg_apt,total_transactions,users_per_hour



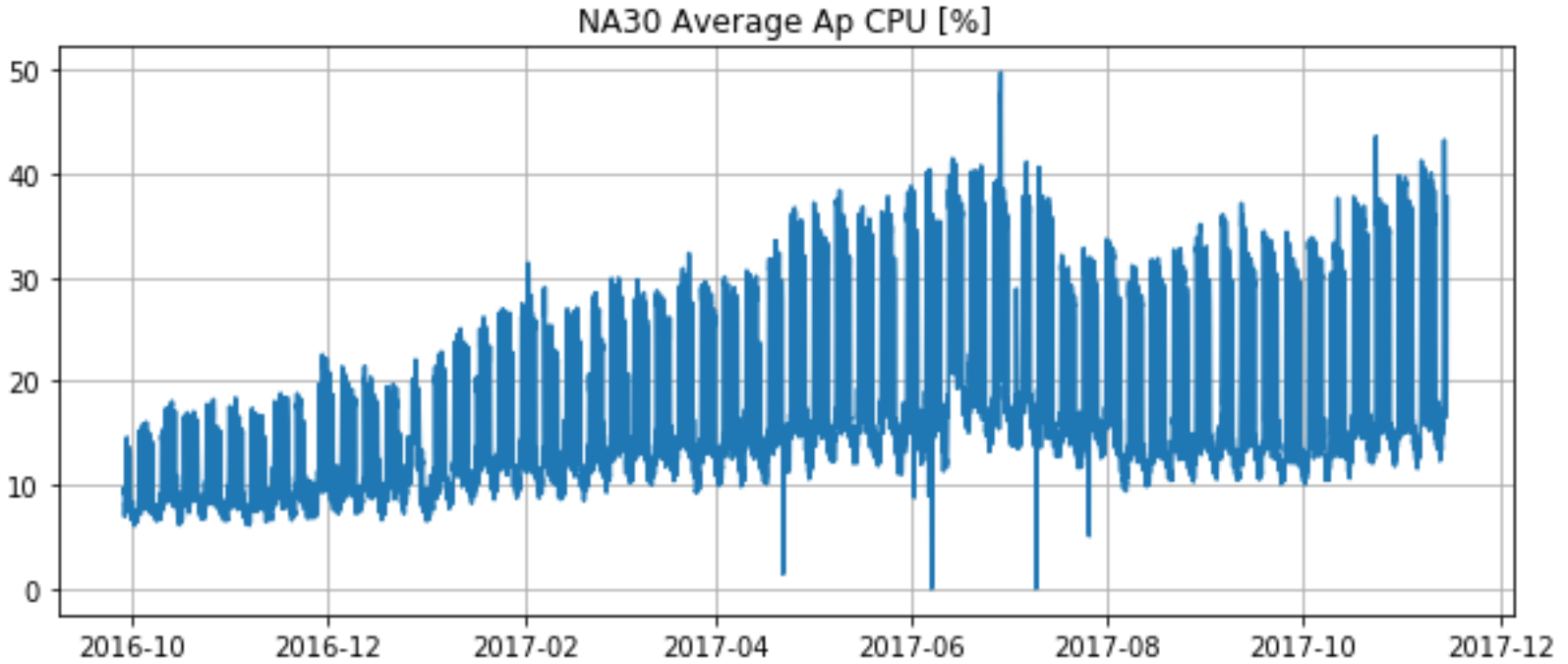
Pod Risk Assessment Component Metrics

- App Tier: app cpu
- DB Tier: db cpu
- SAN Tier: db size and db file sequential read latency
- Combinations of these help determine the overall health of a pod
- Attempt to attach TTLs to system metrics (# transactions, average page time (APT))

Non-Normal Distribution



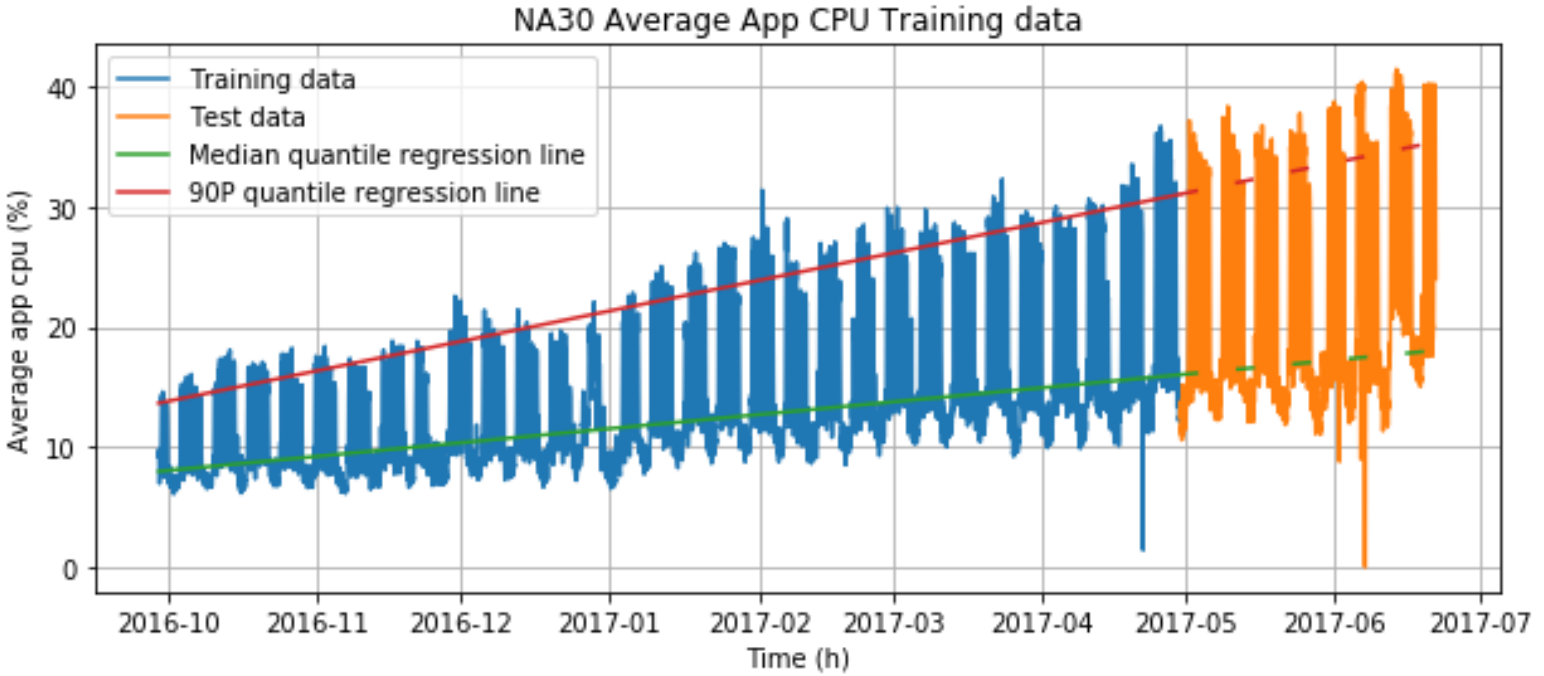
Seasonality



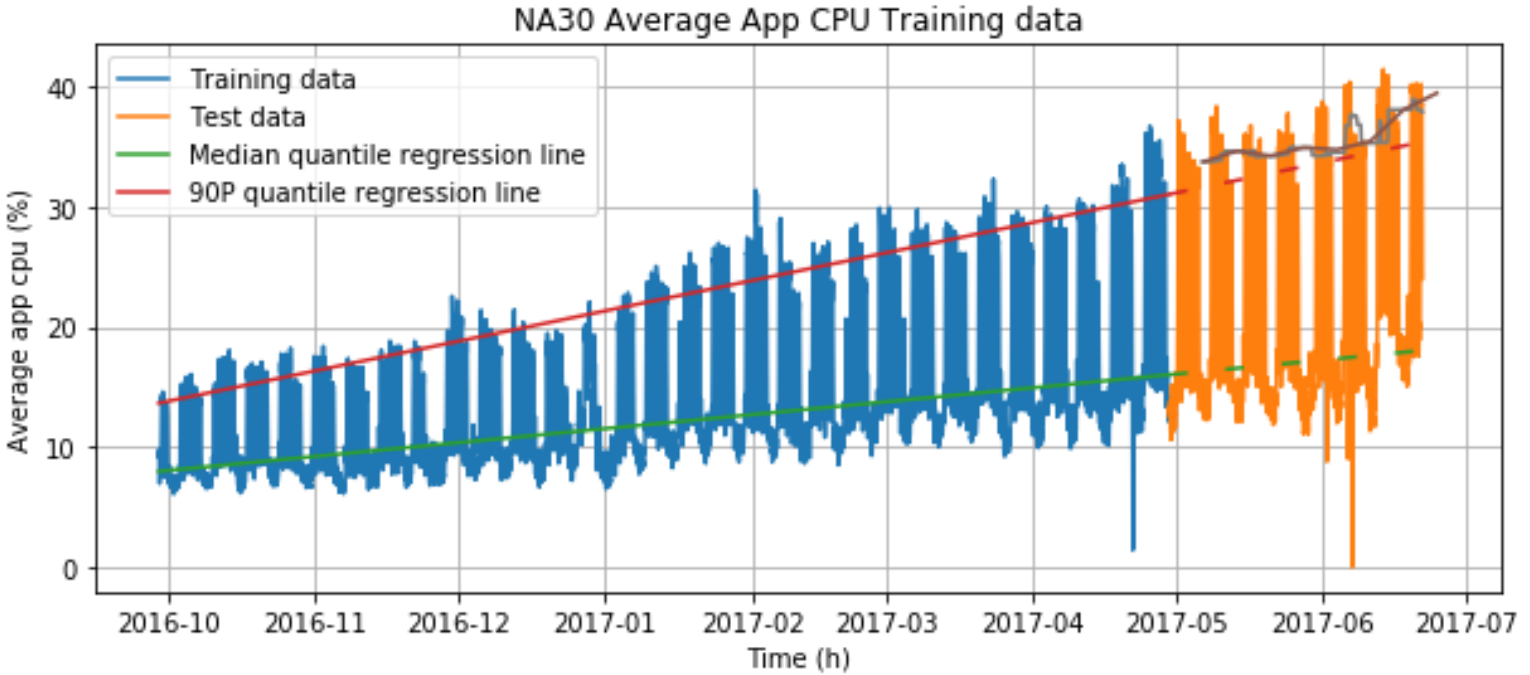
In [6]:
```
data['max_db_cpu_user'].dropna().describe()
```

Out[6]:
```
count     50913.000000
mean       1504.906890
std       25605.315296
min           0.197738
25%          12.048861
50%          18.999514
75%          29.516888
max      759605.900000
Name: max_db_cpu_user, dtype: float64
```

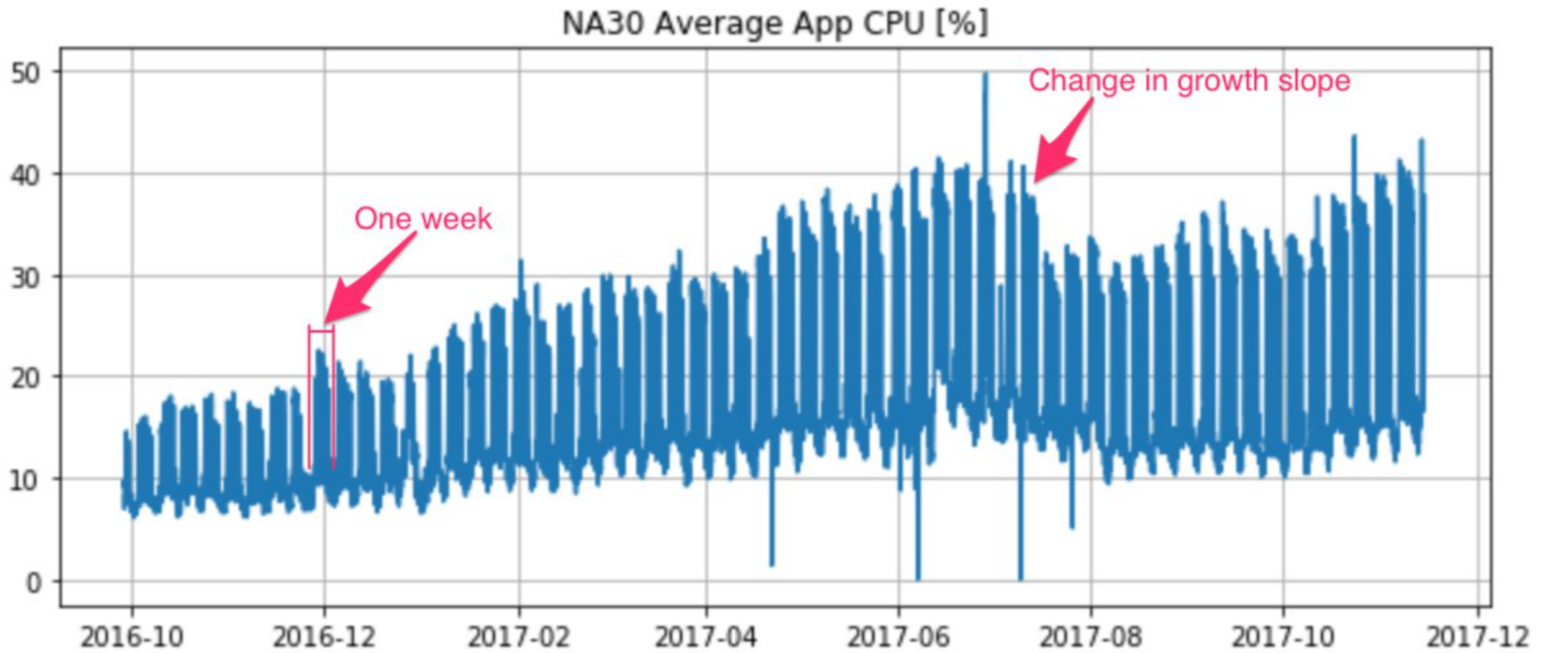
Quantile Regression



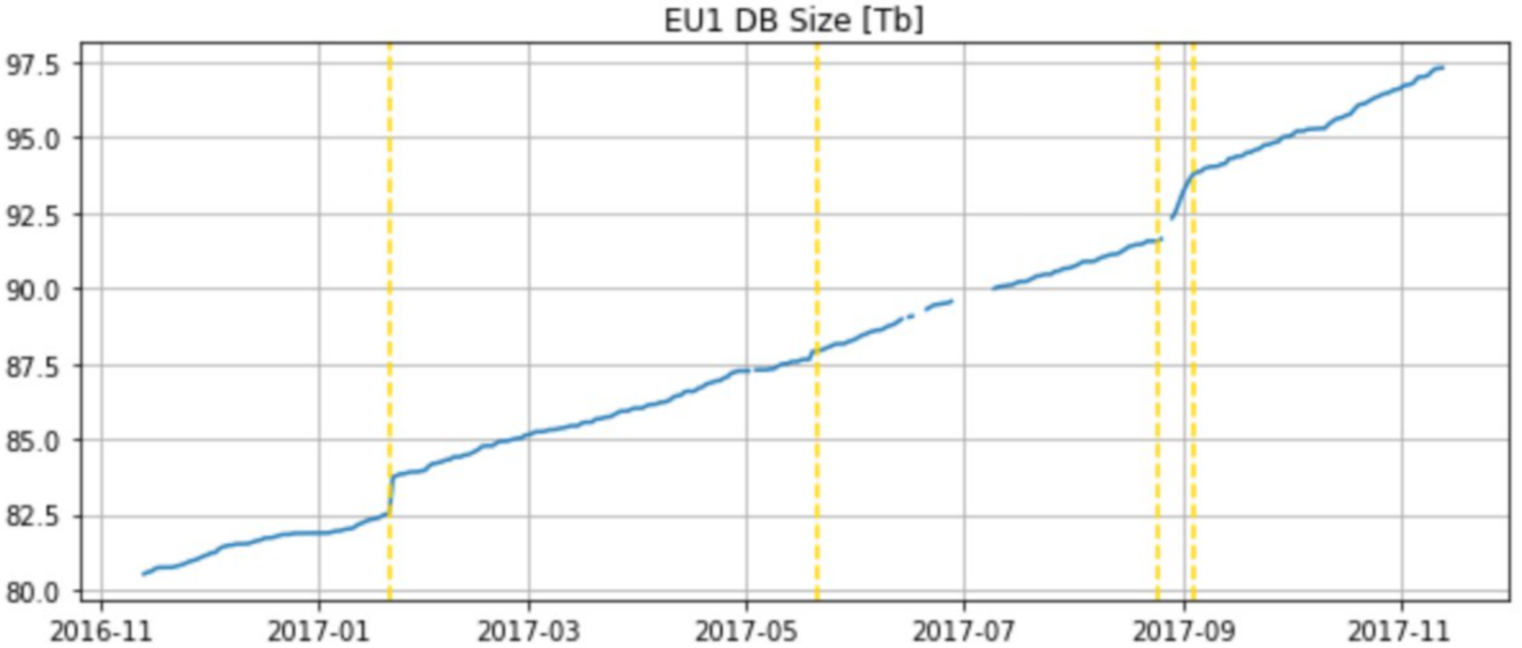
Error Measurements



Seasonality Impacts



Low Variability Metric



Complications

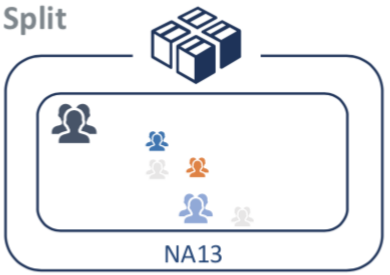
- Hitting a new bottleneck
    - On November 2017 NA30 run into capacity bottlenecks due to MQ - Message Queue and emergency capacity additions had to be done. Our overall TTL was not showing On pod risk assessment, the binding TTL at the time for NA30 was 3 months due to dB CPU. We were not showing a lower TTL due to not having an MQ metric as an input to our model. MQ is going to be added in the near future to Pod Risk Assessment.
- Successful clients
    - We had Uber in one pod and the transactions accelerated. Before we onboard a customer we have no idea how their demand is going to grow or what their usage patterns will be in terms of both their workload seasonality and the types of workload.

Instance Split

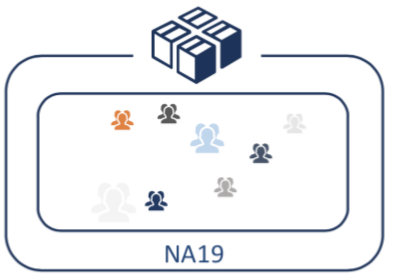
Before Split

120M Transactions
3.9K Active Orgs
APT: 228ms
Peak App CPU: 20%
Peak DB CPU: 40%

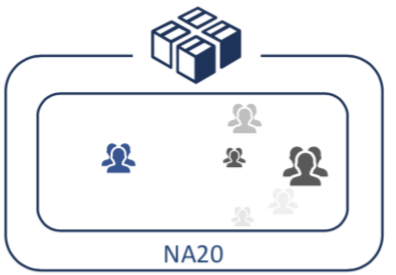
After Split



50M Transactions
1.3K Active Orgs
APT: 149ms
Peak App CPU: 7%
Peak DB CPU: 15%

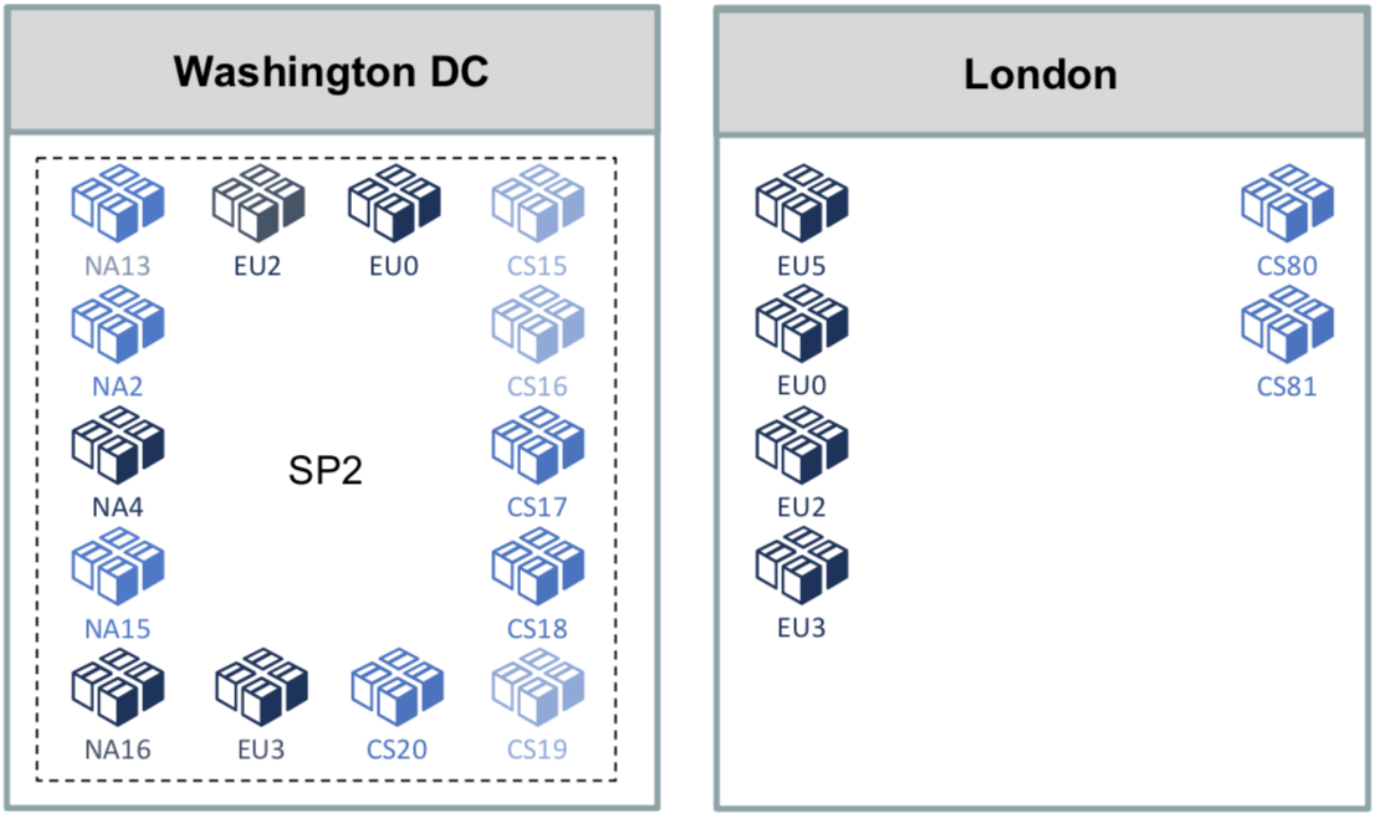


35M Transactions
1.2K Active Orgs
APT: 150ms
Peak App CPU: 4%
Peak DB CPU: 13%



40M Transactions
1.4K Active Orgs
APT: 150ms
Peak App CPU: 5%
Peak DB CPU: 15%

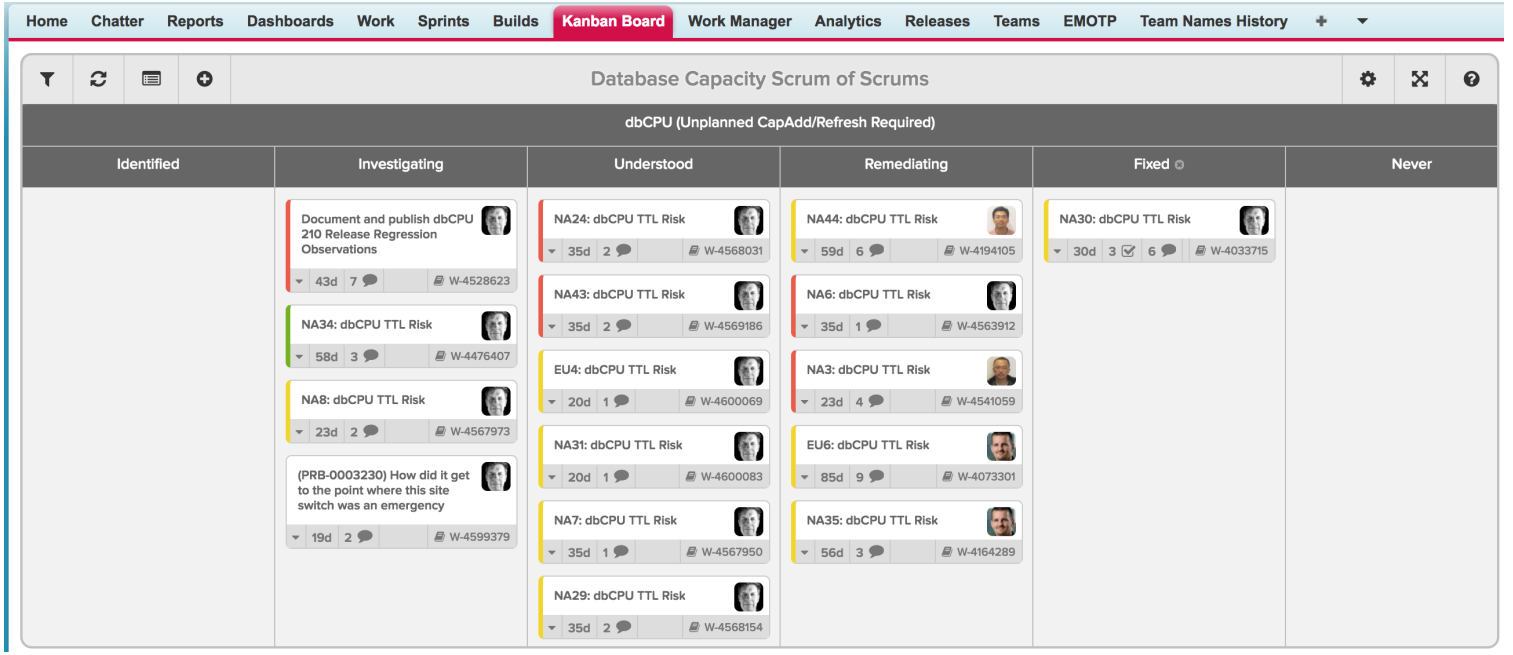
Geo Migration



Environment

Production | Sandbox | About | Download | Data Source Details

DC	SP	Pod	TTL(Months) - Non Binding: tions	TTL(Months) - Non Binding: APT	Performance Metrics: ASM Used MGR (TB/month)	Performance Metrics: App CPU MGR (%/month)	Performance Metrics: Daily Tx % Change (30 vs 30days)	Performance Metrics: Daily Tx % Change (30 vs 90days)	Remediation: Work ID	Remediation: Subject	Remediation: Additional Details
LON	SP9	EU6		3	3.65	3.77	33.32	33.32	W-4073301,...	EU6: dbCPU TTL Risk, EU6:...	Pri-side hardware refresh completed. 2-node CapAdd and...
CHI	SP4	NA30		> 24	1.53	2.75	11.14	12.94	W-4484995	NA30 dbStorage TTL Risk	Planned IR date Jun-2018 will resolve the issue.
FRF	SP1	EU11		4	2.07	2.41	27.19	27.74	W-4054192	EU11: dbStorage/dbIO TT...	dbIO and dbStorage risks. Investigating technical feasibili...
LON	SP9	EU1		1	1.58	1.13	11.99	5.45	W-4464768	EU1 dbStorage TTL Risk	Allocating additional space to ASM
FRF	SP1	EU4		4	2.13	1.17	12.62	5.91	W-4476835,...	EU4: dbStorage TTL Risk, E...	Allocating additional space to ASM, 210 release regressio...
DFW	SP2	NA35		3	N/A	2.59	26	26	W-4164289	NA35: dbCPU TTL Risk	Hardware refresh completed 18-Dec# CapAdd in Jan.
UKB	SP1	AP1		> 24	N/A	3.06	10.93	2.33	W-4105214	AP1 dbStorage TTL Risk	IR=2018/06. DBSR wont be enough. CapAdd planned.


Home
Chatter
Reports
Dashboards
Work
Sprints
Builds
Kanban Board
Work Manager
Analytics
Releases
Teams
EMOTP
Team Names History
Database Capacity Scrum of Scrums
dbCPU (Unplanned CapAdd/Refresh Required)
Identified
Investigating
Understood
Remediating
Fixed
Never
Document and publish dbCPU 210 Release Regression Observations
43d 7 W-4528623
NA34: dbCPU TTL Risk
58d 3 W-4476407
NA8: dbCPU TTL Risk
23d 2 W-4567973
(PRB-0003230) How did it get to the point where this site switch was an emergency
19d 2 W-4599379
NA24: dbCPU TTL Risk
35d 2 W-4568031
NA43: dbCPU TTL Risk
35d 2 W-4569186
EU4: dbCPU TTL Risk
20d 1 W-4600069
NA31: dbCPU TTL Risk
20d 1 W-4600083
NA7: dbCPU TTL Risk
35d 1 W-4567950
NA29: dbCPU TTL Risk
35d 2 W-4568154
NA44: dbCPU TTL Risk
59d 6 W-4194105
NA6: dbCPU TTL Risk
35d 1 W-4563912
NA3: dbCPU TTL Risk
23d 4 W-4541059
EU6: dbCPU TTL Risk
85d 9 W-4073301
NA35: dbCPU TTL Risk
56d 3 W-4164289
NA30: dbCPU TTL Risk
30d 3 6 W-4033715

Homework Assignment: k-Means Iris notebook

- Build 3 k-means clusters
  - one will have 8 clusters
  - one will have 3 clusters
  - one will have 3 clusters and use different initialization data
- Render in a 3-dimensional axis, pulling 3 features from the iris data set.
- Figure out which features work for classification

Resources

https://gist.github.com/bsletten/9a71a1a9ef7c8006dc3ee47b6e0b1f62