Robbie Allen Follow

Startup Founder & Author turned PhD Student @UNCCS focused on Artificial Intelligence. Found...

Jun 1 · 4 min read

Cheat Sheet of Machine Learning and Python (and Math) Cheat Sheets

If you like this article, check out another by Robbie:
My Curated List of AI and Machine Learning Resources

There are many facets to Machine Learning. As I started brushing up on the subject, I came across various “cheat sheets” that compactly listed all the key points I needed to know for a given topic. Eventually, I compiled over 20 Machine Learning-related cheat sheets. Some I reference frequently and thought others may benefit from them too. This post contains 27 of the better cheat sheets I’ve found on the web. Let me know if I’m missing any you like.

Given how rapidly the Machine Learning space is evolving, I imagine these will go out of date quickly, but at least as of June 1, 2017, they are pretty current.

If you want all of the cheat sheets without having to download them individually like I did, I created a zip file containing all 27. Enjoy!

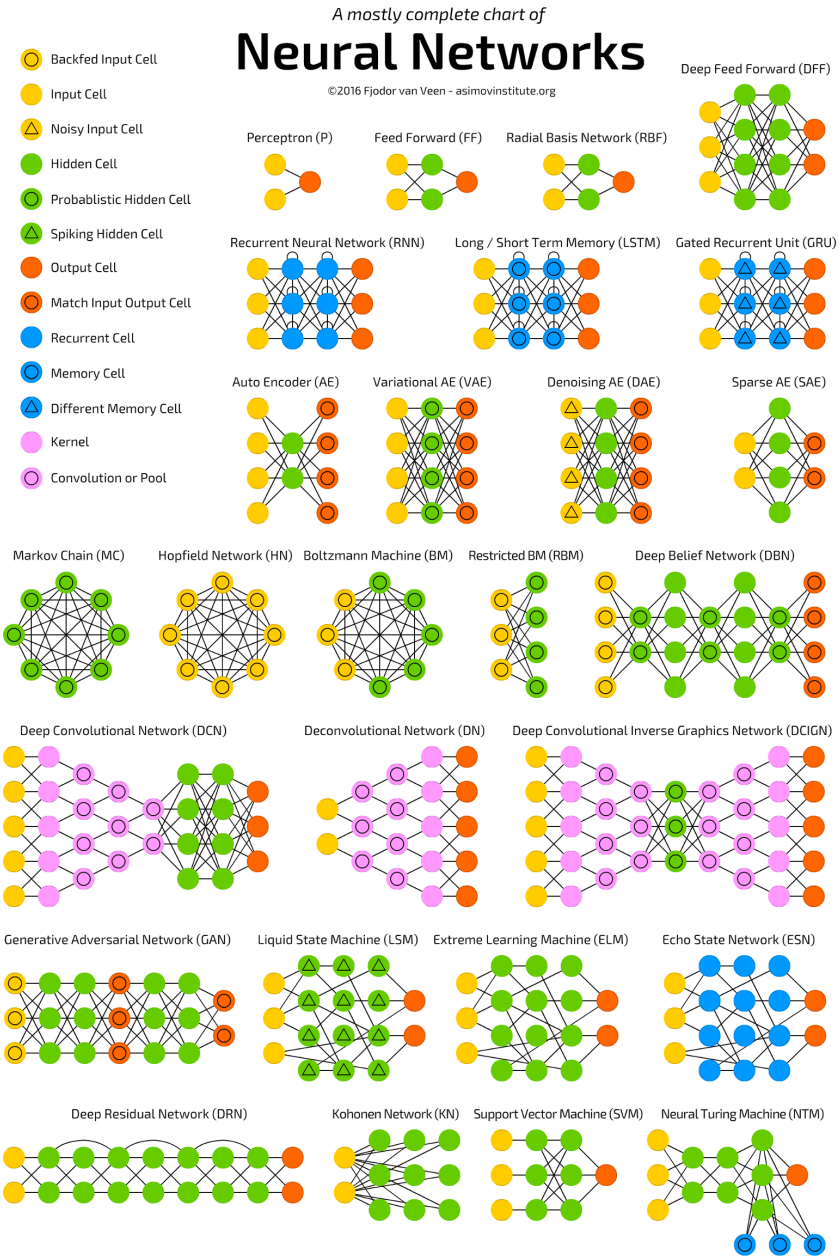
If you like this post, give it a below.

Machine Learning

There are a handful of helpful flowcharts and tables of Machine Learning algorithms. I've included only the most comprehensive ones I've found.

Neural Network Architectures

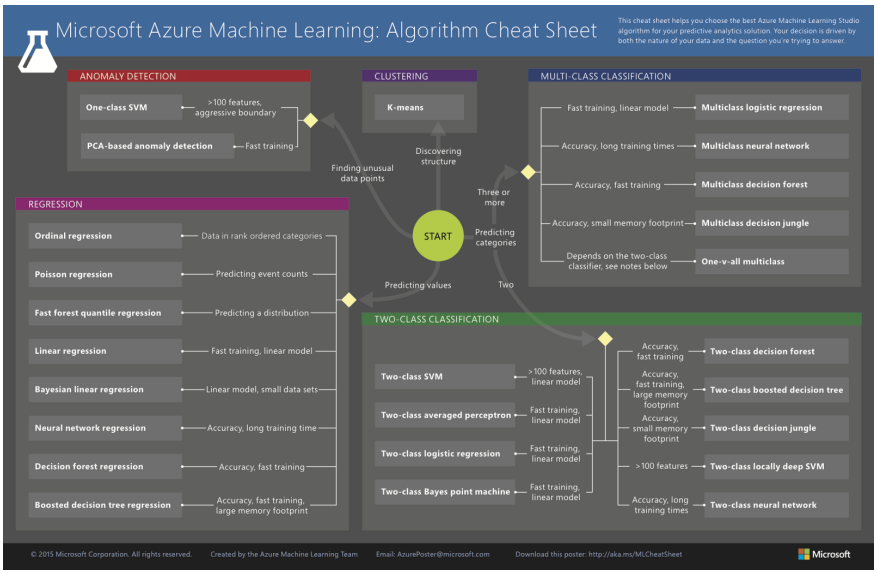
Source: http://www.asimovinstitute.org/neural-network-zoo/



The Neural Network Zoo

Microsoft Azure Algorithm Flowchart

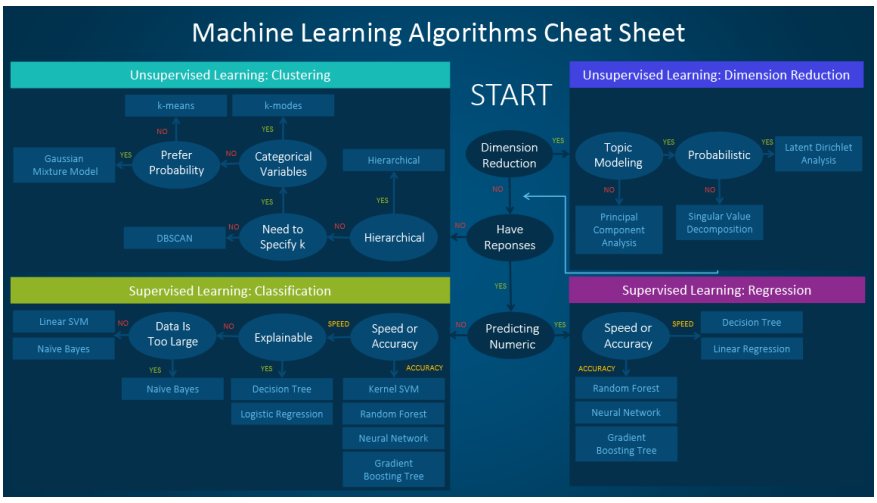
Source: https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-algorithm-cheat-sheet



Machine learning algorithm cheat sheet for Microsoft Azure Machine Learning Studio

SAS Algorithm Flowchart

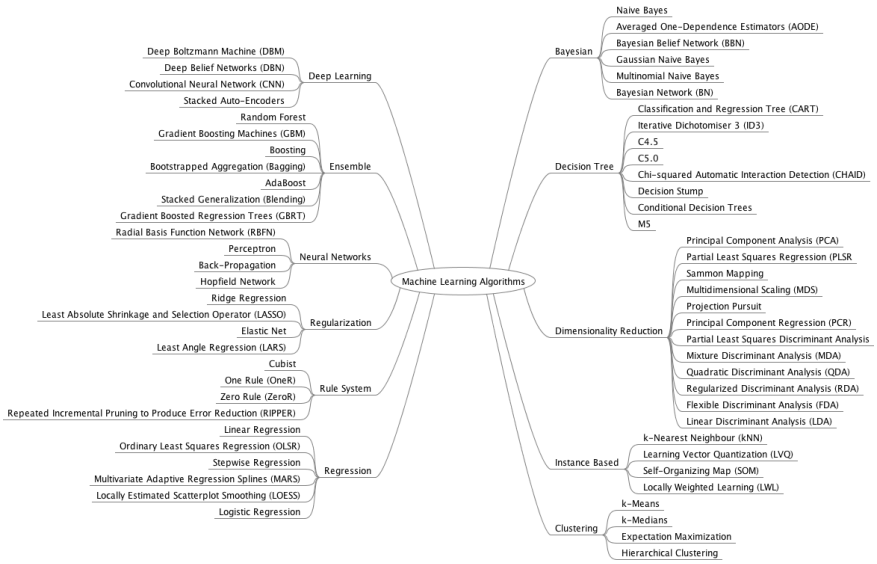
Source: http://blogs.sas.com/content/subconsciousmusings/2017/04/12/machine-learning-algorithm-use/



SAS: Which machine learning algorithm should I use?

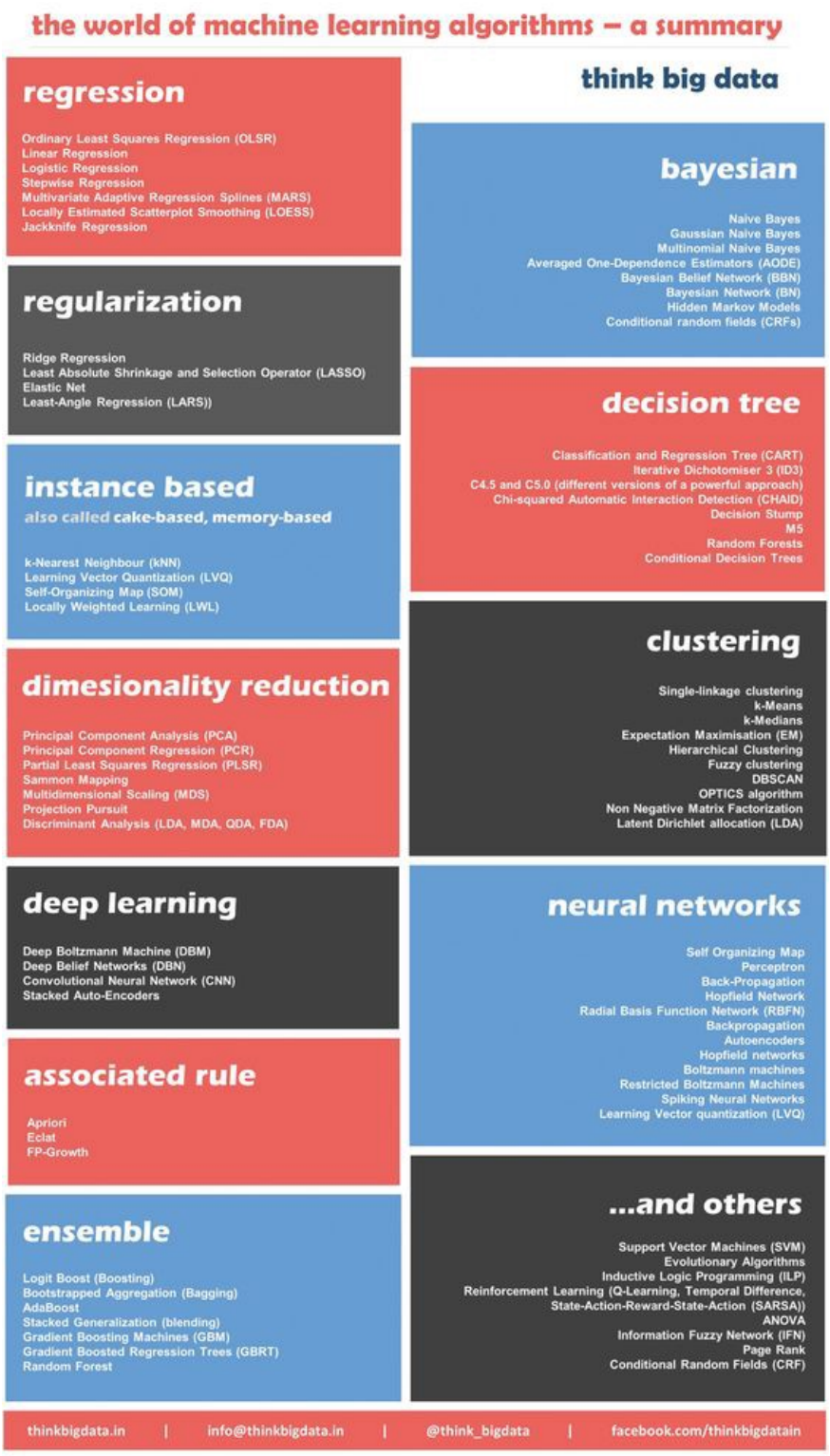
Algorithm Summary

Source: http://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/



A Tour of Machine Learning Algorithms

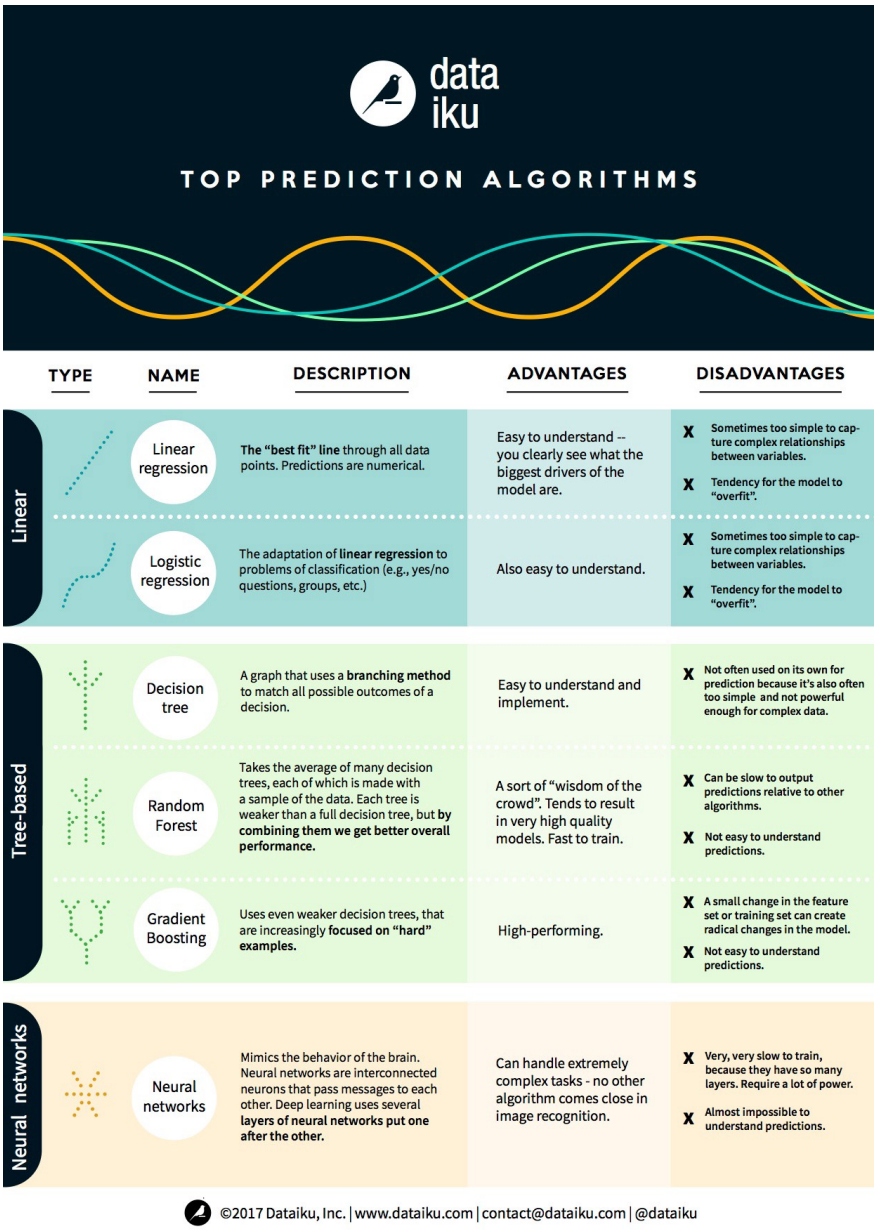
Source: http://thinkbigdata.in/best-known-machine-learning-algorithms-infographic/



Which are the best known machine learning algorithms?

Algorithm Pro/Con

Source: https://blog.dataiku.com/machine-learning-explained-algorithms-are-your-friend

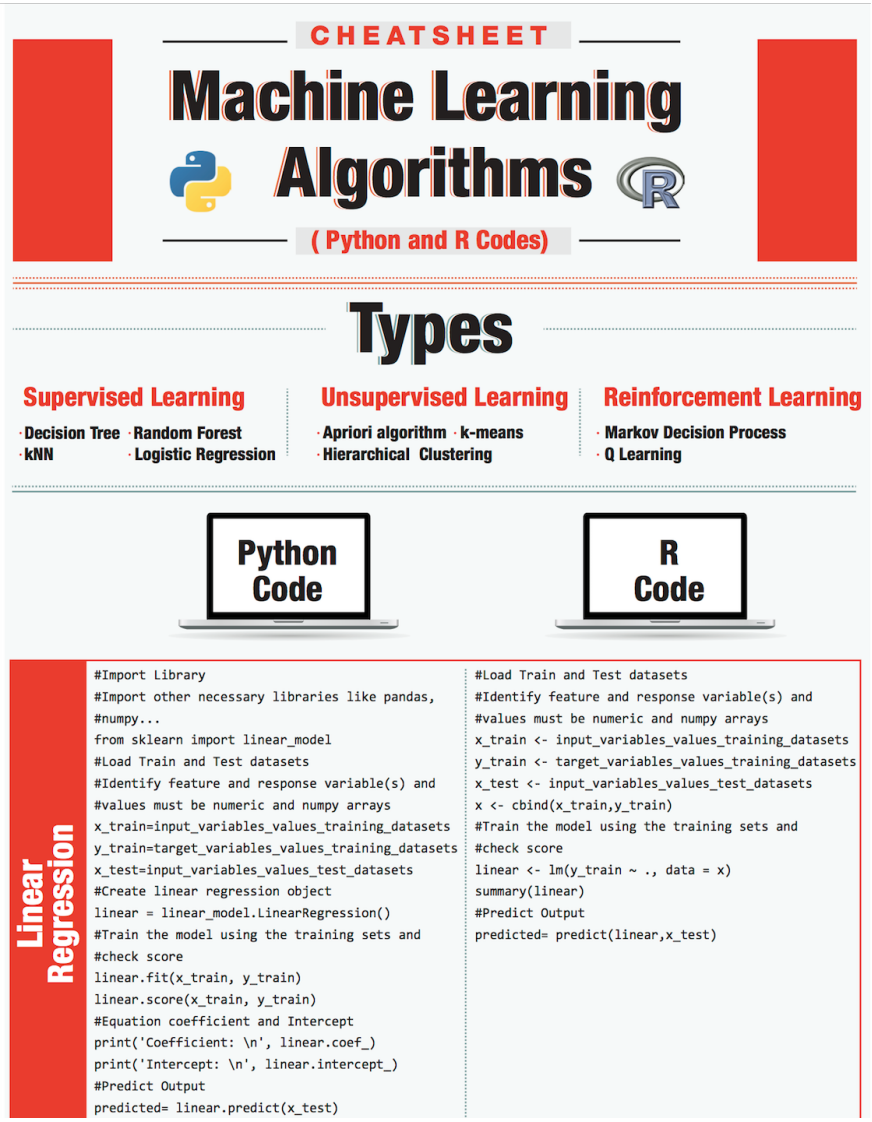


Python

Unsurprisingly, there are a lot of online resources available for Python. For this section, I've only included the best cheat sheets I've come across.

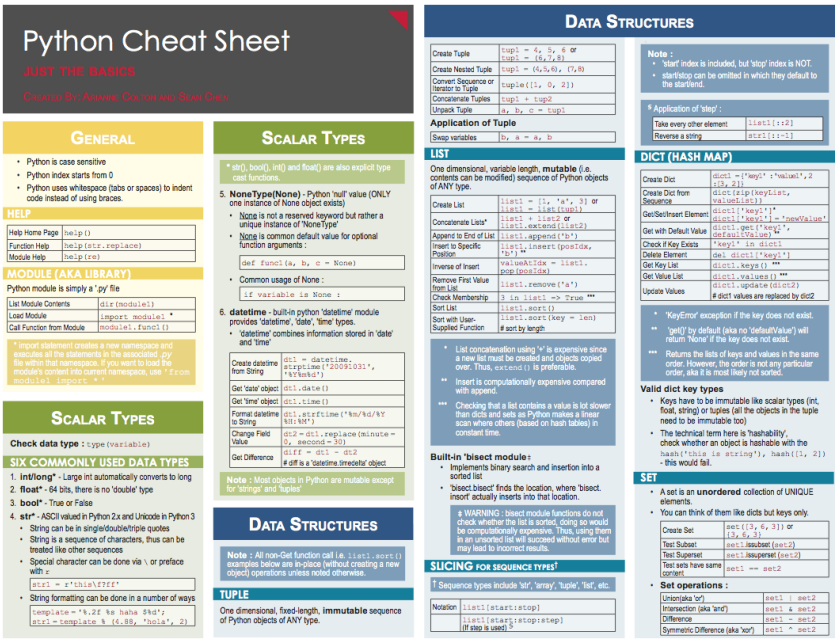
Algorithms

Source: https://www.analyticsvidhya.com/blog/2015/09/full-cheatsheet-machine-learning-algorithms/

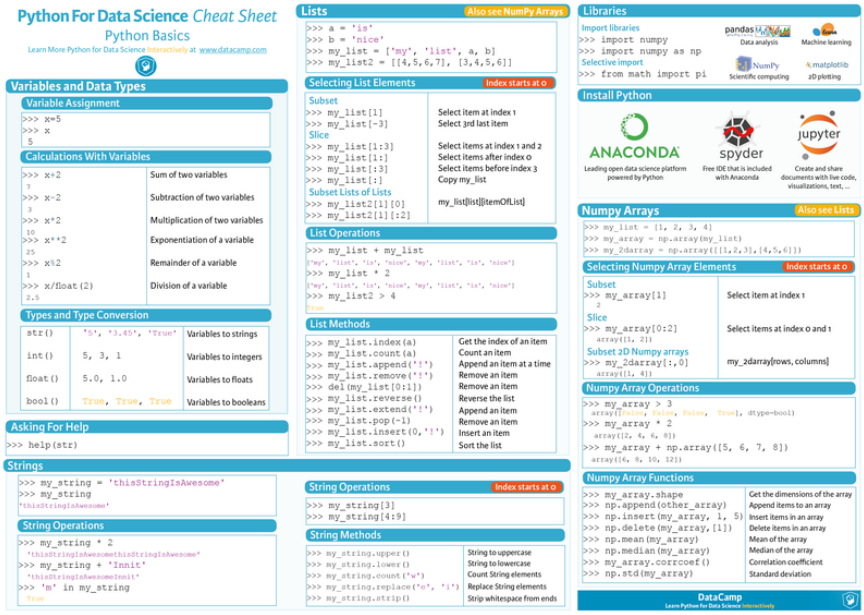


Python Basics

Source: http://datasciencefree.com/python.pdf



Source: https://www.datacamp.com/community/tutorials/python-data-science-cheat-sheet-basics#gs.0x1rxEA



Numpy

Source: https://www.dataquest.io/blog/numpy-cheat-sheet/

Data Science Cheat Sheet

NumPy

KEY

We'll use shorthand in this cheat sheet

`arr` - A numpy Array object

IMPORTS

Import these to start

```
import numpy as np
```

IMPORTING/EXPORTING

`np.loadtxt('file.txt')` - From a text file

`np.genfromtxt('file.csv',delimiter=',')` - From a CSV file

`np.savetxt('file.txt',arr,delimiter=' ')` - Writes to a text file

`np.savetxt('file.csv',arr,delimiter=',')` - Writes to a CSV file

CREATING ARRAYS

`np.array([1,2,3])` - One dimensional array

`np.array([(1,2,3),(4,5,6)])` - Two dimensional array

`np.zeros(3)` - 1D array of length 3 all values 0

`np.ones((3,4))` - 3x4 array with all values 1

`np.eye(5)` - 5x5 array of 0 with 1 on diagonal (Identity matrix)

`np.linspace(0,100,6)` - Array of 6 evenly divided values from 0 to 100

`np.arange(0,10,3)` - Array of values from 0 to less than 10 with step 3 (eg [0,3,6,9])

`np.full((2,3),8)` - 2x3 array with all values 8

`np.random.rand(4,5)` - 4x5 array of random floats between 0-1

`np.random.rand(6,7)*100` - 6x7 array of random floats between 0-100

`np.random.randint(5,size=(2,3))` - 2x3 array with random ints between 0-4

INSPECTING PROPERTIES

`arr.size` - Returns number of elements in `arr`

`arr.shape` - Returns dimensions of `arr` (rows, columns)

`arr.dtype` - Returns type of elements in `arr`

`arr.astype(dtype)` - Convert `arr` elements to type `dtype`

`arr.tolist()` - Convert `arr` to a Python list

`np.info(np.eye)` - View documentation for `np.eye`

COPYING/SORTING/RESHAPING

`np.copy(arr)` - Copies `arr` to new memory

`arr.view(dtype)` - Creates view of `arr` elements with type `dtype`

`arr.sort()` - Sorts `arr`

`arr.sort(axis=0)` - Sorts specific axis of `arr`

`two_d_arr.flatten()` - Flattens 2D array `two_d_arr` to 1D

`arr.T` - Transposes `arr` (rows become columns and vice versa)

`arr.reshape(3,4)` - Reshapes `arr` to 3 rows, 4 columns without changing data

`arr.resize((5,6))` - Changes `arr` shape to 5x6 and fills new values with 0

ADDING/REMOVING ELEMENTS

`np.append(arr,values)` - Appends `values` to end of `arr`

`np.insert(arr,2,values)` - Inserts `values` into `arr` before index 2

`np.delete(arr,3,axis=0)` - Deletes row on index 3 of `arr`

`np.delete(arr,4,axis=1)` - Deletes column on index 4 of `arr`

COMBINING/SPLITTING

`np.concatenate((arr1,arr2),axis=0)` - Adds `arr2` as rows to the end of `arr1`

`np.concatenate((arr1,arr2),axis=1)` - Adds `arr2` as columns to end of `arr1`

`np.split(arr,3)` - Splits `arr` into 3 sub-arrays

`np.hsplit(arr,5)` - Splits `arr` horizontally on the 5th index

INDEXING/SLICING/SUBSETTING

`arr[5]` - Returns the element at index 5

`arr[2,5]` - Returns the 2D array element on index [2][5]

`arr[1]=4` - Assigns array element on index 1 the value 4

`arr[1,3]=10` - Assigns array element on index [1][3] the value 10

`arr[0:3]` - Returns the elements at indices 0,1,2 (On a 2D array: returns rows 0,1,2)

`arr[0:3,4]` - Returns the elements on rows 0,1,2 at column 4

`arr[:2]` - Returns the elements at indices 0,1 (On a 2D array: returns rows 0,1)

`arr[:,1]` - Returns the elements at index 1 on all rows

`arr<5` - Returns an array with boolean values

`(arr1<3) & (arr2>5)` - Returns an array with boolean values

`~arr` - Inverts a boolean array

`arr[arr<5]` - Returns array elements smaller than 5

SCALAR MATH

`np.add(arr,1)` - Add 1 to each array element

`np.subtract(arr,2)` - Subtract 2 from each array element

`np.multiply(arr,3)` - Multiply each array element by 3

`np.divide(arr,4)` - Divide each array element by 4 (returns `np.nan` for division by zero)

`np.power(arr,5)` - Raise each array element to the 5th power

VECTOR MATH

`np.add(arr1,arr2)` - Elementwise add `arr2` to `arr1`

`np.subtract(arr1,arr2)` - Elementwise subtract `arr2` from `arr1`

`np.multiply(arr1,arr2)` - Elementwise multiply `arr1` by `arr2`

`np.divide(arr1,arr2)` - Elementwise divide `arr1` by `arr2`

`np.power(arr1,arr2)` - Elementwise raise `arr1` raised to the power of `arr2`

`np.array_equal(arr1,arr2)` - Returns `True` if the arrays have the same elements and shape

`np.sqrt(arr)` - Square root of each element in the array

`np.sin(arr)` - Sine of each element in the array

`np.log(arr)` - Natural log of each element in the array

`np.abs(arr)` - Absolute value of each element in the array

`np.ceil(arr)` - Rounds up to the nearest int

`np.floor(arr)` - Rounds down to the nearest int

`np.round(arr)` - Rounds to the nearest int

STATISTICS

`np.mean(arr,axis=0)` - Returns mean along specific axis

`arr.sum()` - Returns sum of `arr`

`arr.min()` - Returns minimum value of `arr`

`arr.max(axis=0)` - Returns maximum value of specific axis

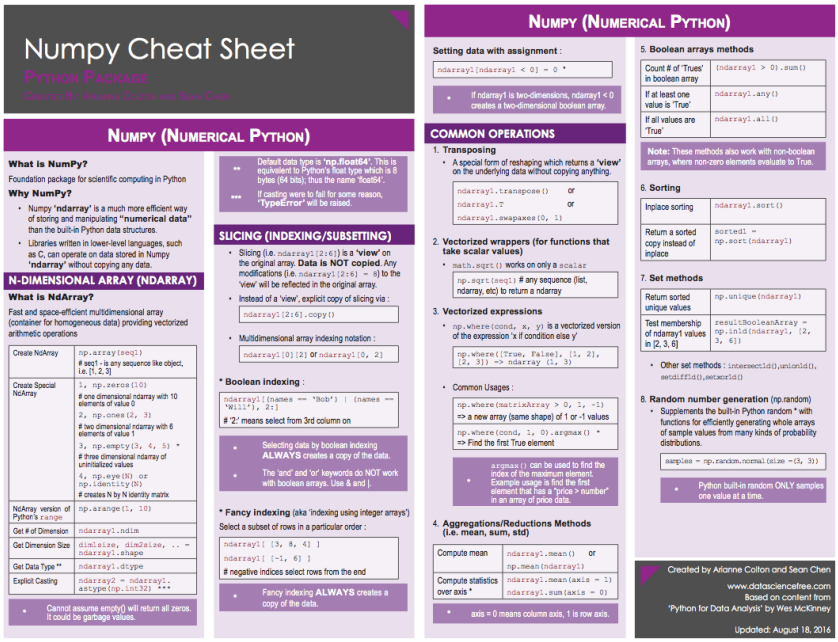
`np.var(arr)` - Returns the variance of array

`np.std(arr,axis=1)` - Returns the standard deviation of specific axis

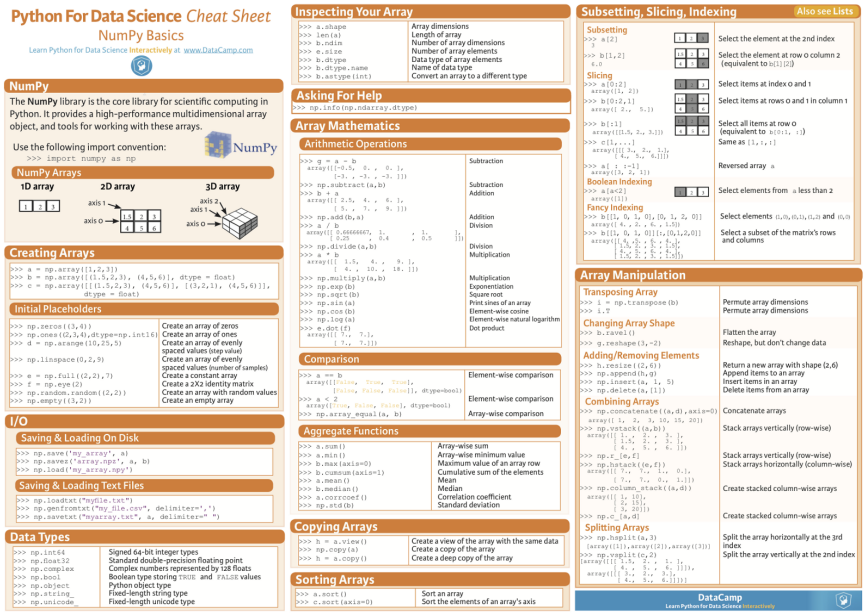
`arr.corrcoef()` - Returns correlation coefficient of array

LEARN DATA SCIENCE ONLINE
Start Learning For Free - www.dataquest.io

Source: http://datasciencefree.com/numpy.pdf



Source: https://www.datacamp.com/community/blog/python-numpy-cheat-sheet#gs.Nw3V6CE



Source: https://github.com/donnemartin/data-science-ipython-notebooks/blob/master/numpy/numpy.ipynb

NumPy

Credits: Forked from Parallel Machine Learning with scikit-learn and IPython by Olivier Grisel

- NumPy Arrays, dtype, and shape
- Common Array Operations
- Reshape and Update In-Place
- Combine Arrays
- Create Sample Data

In [1]:
```
import numpy as np
```

NumPy Arrays, dtypes, and shapes

In [2]:
```
a = np.array([1, 2, 3])
print(a)
print(a.shape)
print(a.dtype)
```

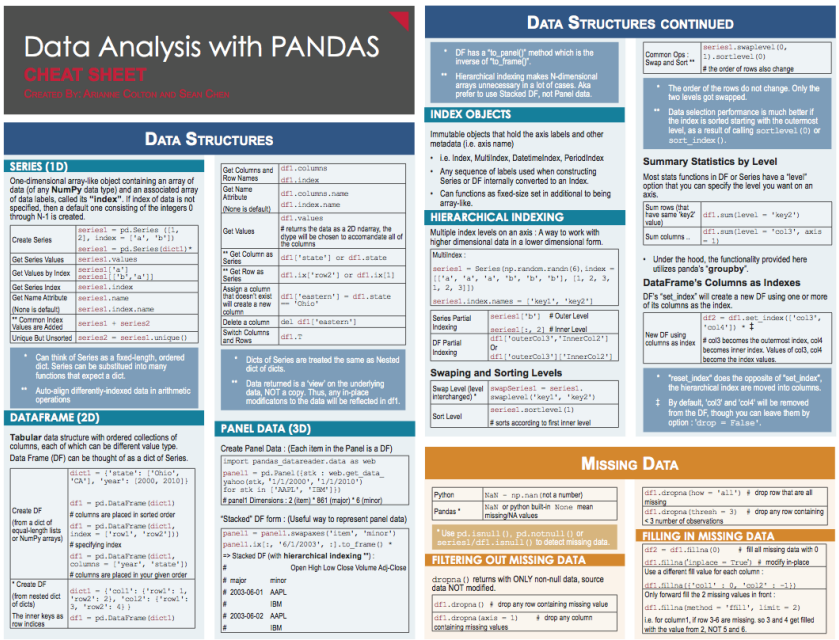
```
[1 2 3]
(3,)
int64
```

In [3]:
```
b = np.array([[0, 2, 4], [1, 3, 5]])
print(b)
print(b.shape)
print(b.dtype)
```

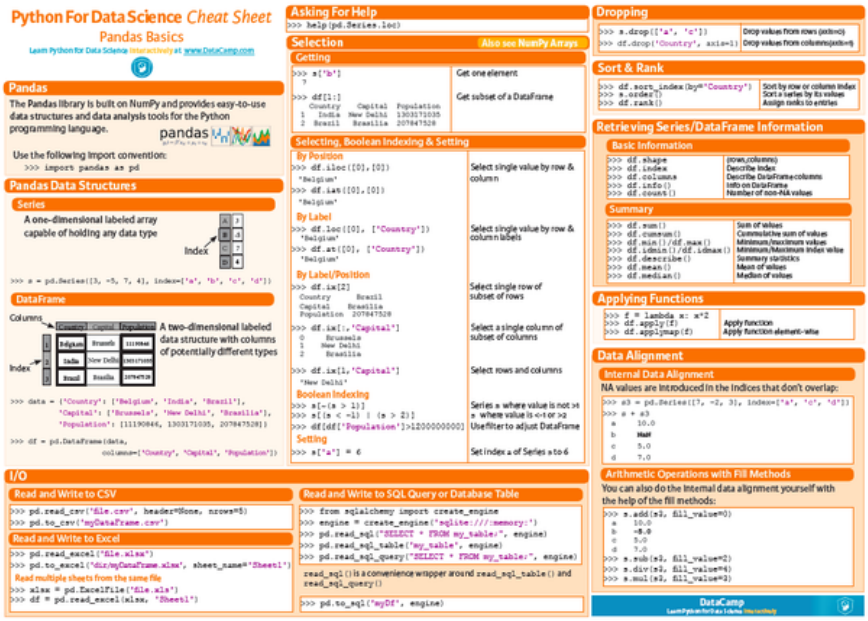
```
[[0 2 4]
 [1 3 5]]
(2, 3)
int64
```

Pandas

Source: http://datasciencefree.com/pandas.pdf



Source: https://www.datacamp.com/community/blog/python-pandas-cheat-sheet#gs.S4P4T=U



Source: https://github.com/donnemartin/data-science-ipython-notebooks/blob/master/pandas/pandas.ipynb

Pandas

Credits: The following are notes taken while working through Python for Data Analysis by Wes McKinney

- Series
- DataFrame
- Reindexing
- Dropping Entries
- Indexing, Selecting, Filtering
- Arithmetic and Data Alignment
- Function Application and Mapping
- Sorting and Ranking
- Axis Indices with Duplicate Values
- Summarizing and Computing Descriptive Statistics
- Cleaning Data (Under Construction)
- Input and Output (Under Construction)

In [1]:
```
from pandas import Series, DataFrame
import pandas as pd
import numpy as np
```

Series

A Series is a one-dimensional array-like object containing an array of data and an associated array of data labels. The data can be any NumPy data type and the labels are the Series' index.

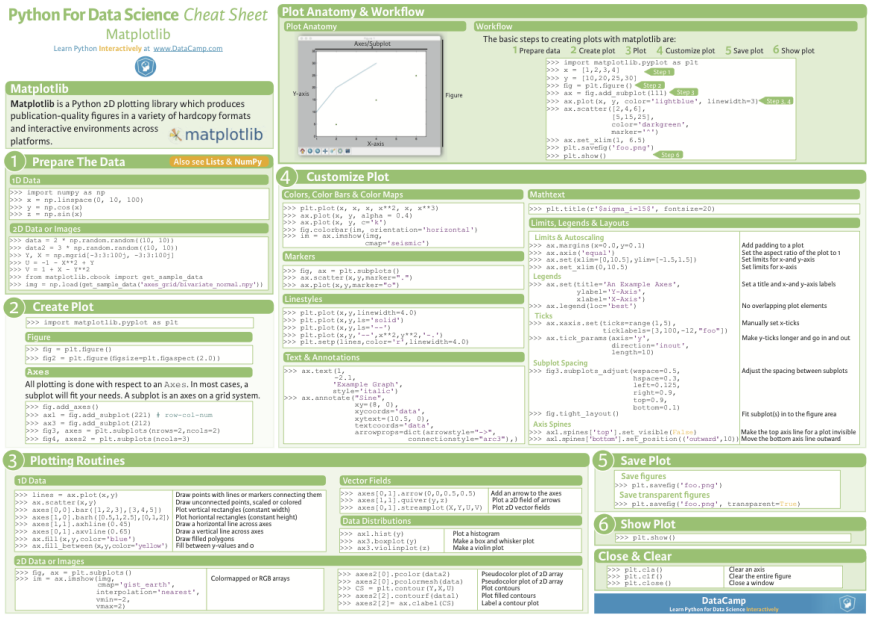
Create a Series:

In [2]:
```
ser_1 = Series([1, 1, 2, -3, -5, 8, 13])
ser_1
```

Out[2]:
```
0     1
1     1
2     2
```

Matplotlib

Source: https://www.datacamp.com/community/blog/python-matplotlib-cheat-sheet



Source: https://github.com/donnemartin/data-science-ipython-notebooks/blob/master/matplotlib/matplotlib.ipynb

matplotlib

Credits: Content forked from Parallel Machine Learning with scikit-learn and IPython by Olivier Grisel

- Setting Global Parameters
- Basic Plots
- Histograms
- Two Histograms on the Same Plot
- Scatter Plots

In [1]:
```
%matplotlib inline
import pandas as pd
import numpy as np
import pylab as plt
import seaborn
```

Setting Global Parameters

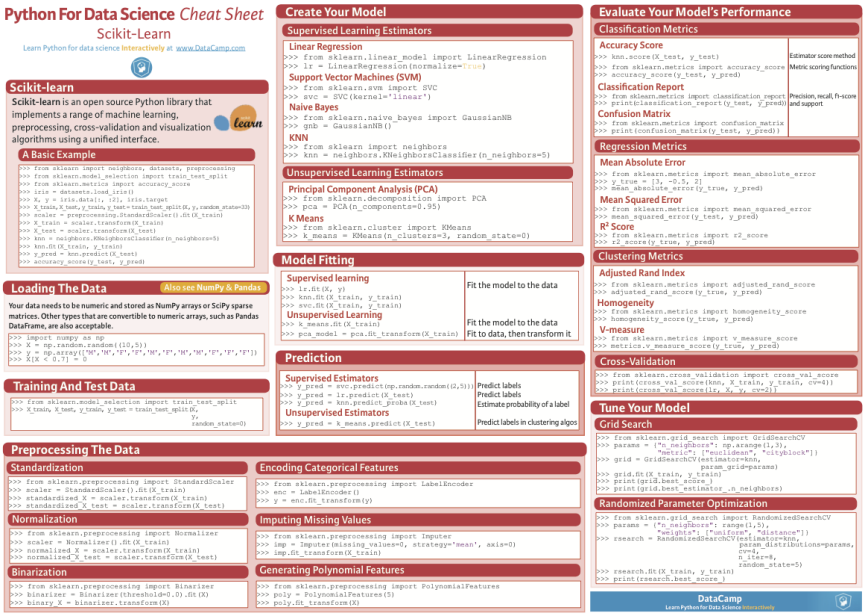
In [2]:
```
# Set the global default size of matplotlib figures
plt.rc('figure', figsize=(10, 5))

# Set seaborn aesthetic parameters to defaults
seaborn.set()
```

Basic Plots

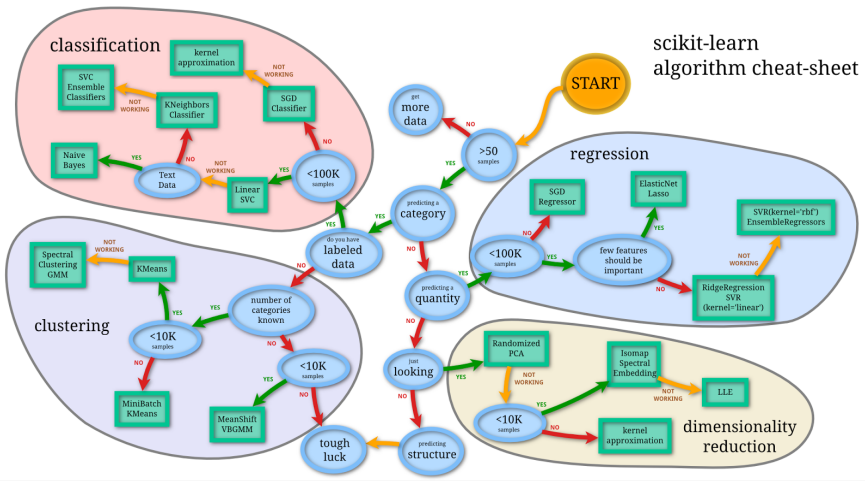
In [3]:
```
x = np.linspace(0, 2, 10)

plt.plot(x, x, 'o-', label='linear')
plt.plot(x, x ** 2, 'x-', label='quadratic')

plt.legend(loc='best')
plt.title('Linear vs Quadratic progression')
```

Scikit Learn

Source: https://www.datacamp.com/community/blog/scikit-learn-cheat-sheet#gs.fZ2A1Jk



Source: http://peekaboo-vision.blogspot.de/2013/01/machine-learning-cheat-sheet-for-scikit.html



Source: https://github.com/rcompton/ml_cheat_sheet/blob/master/supervised_learning.ipynb

In [1]:
```
import sklearn
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn.datasets
%pylab inline

#sklearn two moons generator makes lots of these...
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [2]:
```
"""
Build some datasets that I'll demo the models on
"""
Xs = []
ys = []

#low noise, plenty of samples, should be easy
X0, y0 = sklearn.datasets.make_moons(n_samples=1000, noise=.05)
Xs.append(X0)
ys.append(y0)

#more noise, plenty of samples
X1, y1 = sklearn.datasets.make_moons(n_samples=1000, noise=.3)
Xs.append(X1)
ys.append(y1)

#less noise, few samples
X2, y2 = sklearn.datasets.make_moons(n_samples=200, noise=.05)
Xs.append(X2)
ys.append(y2)

#more noise, less samples, should be hard
X3, y3 = sklearn.datasets.make_moons(n_samples=200, noise=.3)
Xs.append(X3)
```

Tensorflow

Source: https://github.com/aymericdamien/TensorFlow-Examples/blob/master/notebooks/1_Introduction/basic_operations.ipynb

In [1]:
```
# Basic Operations example using TensorFlow library.
# Author: Aymeric Damien
# Project: https://github.com/aymericdamien/TensorFlow-Examples/
```

In [2]:
```
import tensorflow as tf
```

In [3]:
```
# Basic constant operations
# The value returned by the constructor represents the output
# of the Constant op.
a = tf.constant(2)
b = tf.constant(3)
```

In [4]:
```
# Launch the default graph.
with tf.Session() as sess:
    print "a: %i" % sess.run(a), "b: %i" % sess.run(b)
    print "Addition with constants: %i" % sess.run(a+b)
    print "Multiplication with constants: %i" % sess.run(a*b)
```

```
a=2, b=3
Addition with constants: 5
Multiplication with constants: 6
```

In [5]:
```
# Basic Operations with variable as graph input
# The value returned by the constructor represents the output
# of the Variable op. (define as input when running session)
# tf Graph input
a = tf.placeholder(tf.int16)
b = tf.placeholder(tf.int16)
```

In [6]:
```
# Define some operations
add = tf.add(a, b)
mul = tf.multiply(a, b)
```

Pytorch

Source: https://github.com/bfortuner/pytorch-cheatsheet

Pytorch Cheatsheet

Imports

In [1]:
```
import torch
import torch.nn as nn
import torch.nn.init as init
import torch.optim as optim
import torch.nn.functional as F
from torch.autograd import Variable
import torchvision.datasets as datasets
import torchvision.transforms as transforms
import torchvision.utils as tv_utils
from torch.utils.data import DataLoader
import torchvision.models as models
import torch.backends.cudnn as cudnn
import torchvision
import torch.autograd as autograd
from PIL import Image
import imp
import os
import sys
import math
import time
import random
import shutil
import cv2
import scipy.misc
from glob import glob
import sklearn
import logging

from tqdm import tqdm
import numpy as np
import matplotlib as mpl
mpl.use('Agg')
import matplotlib.pyplot as plt
plt.style.use('bmh')

%matplotlib inline
```

Basics

- http://pytorch.org/tutorials/beginner/pytorch_with_examples.html
- http://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

Datasets

File Management

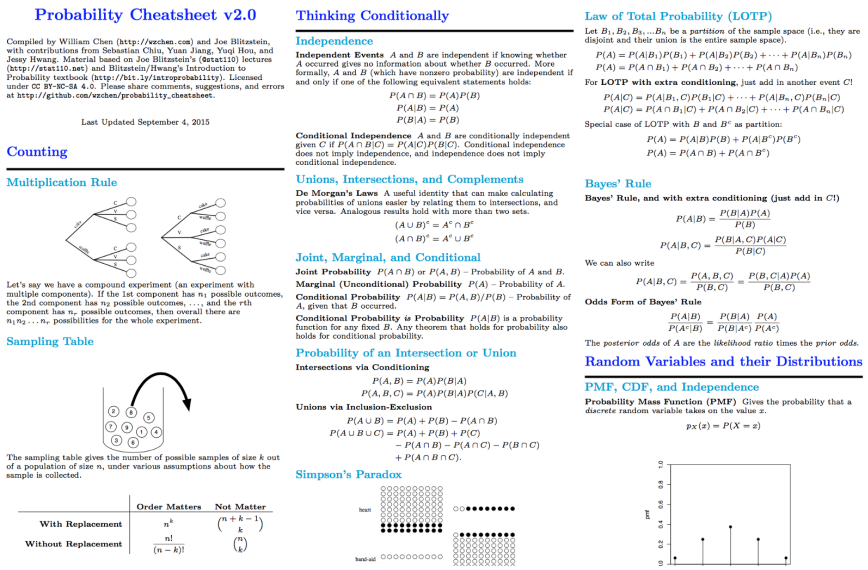
In [ ]:
```
random.seed(1)
torch.manual_seed(1)
DATA_PATH='/media/bfortuner/bigguy/data/'
```

Math

If you really want to understand Machine Learning, you need a solid understanding of Statistics (especially Probability), Linear Algebra, and some Calculus. I minored in Math during undergrad, but I definitely needed a refresher. These cheat sheets provide most of what you need to understand the Math behind the most common Machine Learning algorithms.

Probability

Source: http://www.wzchen.com/s/probability_cheatsheet.pdf



Probability Cheatsheet v2.0

Compiled by William Chen (http://wzchen.com) and Joe Blitzstein, with contributions from Sebastian Chiu, Yuan Jiang, Yuqi Hou, and Jessy Hwang. Material based on Joe Blitzstein's (@stat110) lectures (http://stat110.net) and Blitzstein/Hwang's Introduction to Probability textbook (http://bit.ly/introprobability).

Last Updated September 4, 2015

Counting

Multiplication Rule

Let's say we have a compound experiment (an experiment with multiple components). If the 1st component has n_1 possible outcomes, the 2nd component has n_2 possible outcomes, ..., and the rth component has n_r possible outcomes, then overall there are $n_1 n_2 \ldots n_r$ possibilities for the whole experiment.

Sampling Table

The sampling table gives the number of possible samples of size k out of a population of size n, under various assumptions about how the sample is collected.

	Order Matters	Not Matter
With Replacement	n^k	$\binom{n+k-1}{k}$
Without Replacement	$\frac{n!}{(n-k)!}$	$\binom{n}{k}$

Thinking Conditionally

Independence

Independent Events A and B are independent if knowing whether A occurred gives no information about whether B occurred. More formally, A and B (which have nonzero probability) are independent if and only if one of the following equivalent statements holds:

$$P(A \cap B) = P(A)P(B)$$
$$P(A|B) = P(A)$$
$$P(B|A) = P(B)$$

Conditional Independence A and B are conditionally independent given C if $P(A \cap B|C) = P(A|C)P(B|C)$. Conditional independence does not imply independence, and independence does not imply conditional independence.

Unions, Intersections, and Complements

De Morgan's Laws A useful identity that can make calculating probabilities of unions easier by relating them to intersections, and vice versa. Analogous results hold with more than two sets.

$$(A \cup B)^c = A^c \cap B^c$$
$$(A \cap B)^c = A^c \cup B^c$$

Joint, Marginal, and Conditional

Joint Probability $P(A \cap B)$ or $P(A, B)$ – Probability of A and B.

Marginal (Unconditional) Probability $P(A)$ – Probability of A.

Conditional Probability $P(A|B) = P(A, B)/P(B)$ – Probability of A, given that B occurred.

Conditional Probability is Probability $P(A|B)$ is a probability function for any fixed B. Any theorem that holds for probability also holds for conditional probability.

Probability of an Intersection or Union

Intersections via Conditioning

$$P(A, B) = P(A)P(B|A)$$
$$P(A, B, C) = P(A)P(B|A)P(C|A, B)$$

Unions via Inclusion-Exclusion

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$
$$P(A \cup B \cup C) = P(A) + P(B) + P(C) - P(A \cap B) - P(A \cap C) - P(B \cap C) + P(A \cap B \cap C).$$

Simpson's Paradox

Law of Total Probability (LOTP)

Let $B_1, B_2, B_3, \ldots B_n$ be a *partition* of the sample space (i.e., they are disjoint and their union is the entire sample space).

$$P(A) = P(A|B_1)P(B_1) + P(A|B_2)P(B_2) + \cdots + P(A|B_n)P(B_n)$$
$$P(A) = P(A \cap B_1) + P(A \cap B_2) + \cdots + P(A \cap B_n)$$

For **LOTP with extra conditioning**, just add in another event C!

$$P(A|C) = P(A|B_1, C)P(B_1|C) + \cdots + P(A|B_n, C)P(B_n|C)$$
$$P(A|C) = P(A \cap B_1|C) + P(A \cap B_2|C) + \cdots + P(A \cap B_n|C)$$

Special case of LOTP with B and B^c as partition:

$$P(A) = P(A|B)P(B) + P(A|B^c)P(B^c)$$
$$P(A) = P(A \cap B) + P(A \cap B^c)$$

Bayes' Rule

Bayes' Rule, and with extra conditioning (just add in C!)

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

We can also write

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

Odds Form of Bayes' Rule

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

The *posterior odds* of A are the *likelihood ratio* times the *prior odds*.

Random Variables and their Distributions

PMF, CDF, and Independence

Probability Mass Function (PMF) Gives the probability that a discrete random variable takes on the value x.

$$p_X(x) = P(X = x)$$

Probability Cheatsheet v2.0

Linear Algebra

Source: https://minireference.com/static/tutorials/linear_algebra_in_4_pages.pdf

Linear algebra explained in four pages

Excerpt from the NO BULLSHIT GUIDE TO LINEAR ALGEBRA by Ivan Savov

***Abstract*—This document will review the fundamental ideas of linear algebra. We will learn about matrices, matrix operations, linear transformations and discuss both the theoretical and computational aspects of linear algebra. The tools of linear algebra open the gateway to the study of more advanced mathematics. A lot of *knowledge buzz* awaits you if you choose to follow the path of *understanding*, instead of trying to memorize a bunch of formulas.**

I. Introduction

Linear algebra is the math of vectors and matrices. Let n be a positive integer and let $\mathbb{R}$ denote the set of real numbers, then $\mathbb{R}^n$ is the set of all n-tuples of real numbers. A vector $\vec{v} \in \mathbb{R}^n$ is an n-tuple of real numbers. The notation "$\in S$" is read "element of S." For example, consider a vector that has three components:

$$\vec{v} = (v_1, v_2, v_3) \in (\mathbb{R}, \mathbb{R}, \mathbb{R}) \equiv \mathbb{R}^3.$$

A matrix $A \in \mathbb{R}^{m \times n}$ is a rectangular array of real numbers with m rows and n columns. For example, a 3×2 matrix looks like this:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \in \begin{bmatrix} \mathbb{R} & \mathbb{R} \\ \mathbb{R} & \mathbb{R} \\ \mathbb{R} & \mathbb{R} \end{bmatrix} \equiv \mathbb{R}^{3\times 2}.$$

The purpose of this document is to introduce you to the mathematical operations that we can perform on vectors and matrices and to give you a feel of the power of linear algebra. Many problems in science, business, and technology can be described in terms of vectors and matrices so it is important that you understand how to work with these.

Prerequisites

The only prerequisite for this tutorial is a basic understanding of high school math concepts[1] like numbers, variables, equations, and the fundamental arithmetic operations on real numbers: addition (denoted $+$), subtraction (denoted $-$), multiplication (denoted implicitly), and division (fractions).

B. Matrix operations

We denote by A the matrix as a whole and refer to its entries as a_{ij}. The mathematical operations defined for matrices are the following:

- addition (denoted $+$)

$$C = A + B \quad \Leftrightarrow \quad c_{ij} = a_{ij} + b_{ij}.$$

- subtraction (the inverse of addition)
- matrix product. The product of matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times \ell}$ is another matrix $C \in \mathbb{R}^{m\times \ell}$ given by the formula

$$C = AB \quad \Leftrightarrow \quad c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj},$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \\ a_{31}b_{11} + a_{32}b_{21} & a_{31}b_{12} + a_{32}b_{22} \end{bmatrix}$$

- matrix inverse (denoted A^{-1})
- matrix transpose (denoted $^{\mathsf{T}}$):

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \end{bmatrix}^{\mathsf{T}} = \begin{bmatrix} \alpha_1 & \beta_1 \\ \alpha_2 & \beta_2 \\ \alpha_3 & \beta_3 \end{bmatrix}.$$

- matrix trace: $\mathrm{Tr}[A] \equiv \sum_{i=1}^{n} a_{ii}$
- determinant (denoted $\det(A)$ or $|A|$)

Note that the matrix product is not a commutative operation: $AB \neq BA$.

C. Matrix-vector product

The matrix-vector product is an important special case of the matrix-matrix product. The product of a 3×2 matrix A and the 2×1 column vector $\vec{x}$ results in a 3×1 vector $\vec{y}$ given by:

Linear algebra explained in four pages

Statistics

Source: http://web.mit.edu/~csvoss/Public/usabo/stats_handout.pdf

Statistics Cheat Sheet

Population
The entire group one desires information about

Sample
A subset of the population taken because the entire population is usually too large to analyze
Its characteristics are taken to be representative of the population

Mean
Also called the arithmetic mean or average
The sum of all the values in the sample divided by the number of values in the sample/population
μ is the mean of the population; $\bar{x}$ is the mean of the sample

Median
The value separating the higher half of a sample/population from the lower half
Found by arranging all the values from lowest to highest and taking the middle one (or the mean of the middle two if there are an even number of values)

Variance
Measures dispersion around the mean
Determined by averaging the squared differences of all the values from the mean
Variance of a population is σ^2

$$\sigma^2 = \frac{\sum(x-\mu)^2}{n}$$

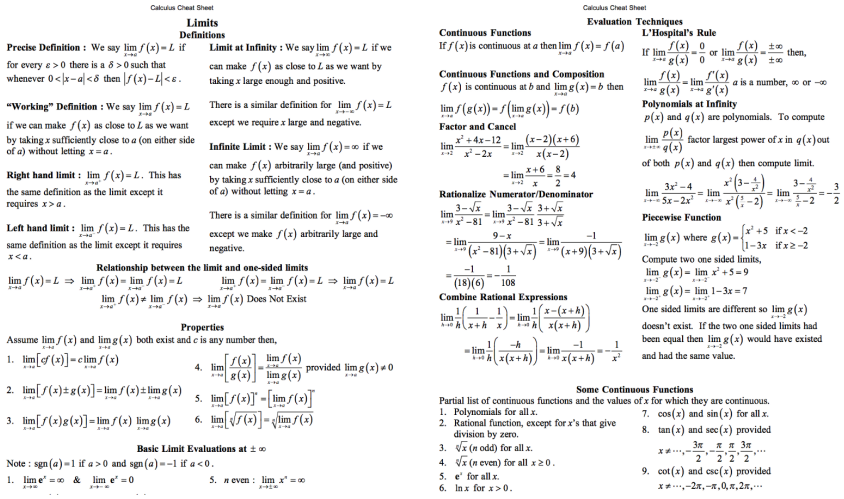
Can be calculated by subtracting the square of the mean from the average of the squared scores:

$$\sigma^2 = \frac{\sum x^2}{n} - \mu^2$$

Statistics Cheat Sheet

Calculus

Source: http://tutorial.math.lamar.edu/getfile.aspx?file=B,41,N



Calculus Cheat Sheet

Limits

Definitions

Precise Definition : We say $\lim_{x\to a} f(x) = L$ if for every $\varepsilon > 0$ there is a $\delta > 0$ such that whenever $0 < |x-a| < \delta$ then $|f(x)-L| < \varepsilon$.

"Working" Definition : We say $\lim_{x\to a} f(x) = L$ if we can make $f(x)$ as close to L as we want by taking x sufficiently close to a (on either side of a) without letting $x = a$.

Right hand limit : $\lim_{x\to a^+} f(x) = L$. This has the same definition as the limit except it requires $x > a$.

Left hand limit : $\lim_{x\to a^-} f(x) = L$. This has the same definition as the limit except it requires $x < a$.

Limit at Infinity : We say $\lim_{x\to\infty} f(x) = L$ if we can make $f(x)$ as close to L as we want by taking x large enough and positive.

There is a similar definition for $\lim_{x\to-\infty} f(x) = L$ except we require x large and negative.

Infinite Limit : We say $\lim_{x\to a} f(x) = \infty$ if we can make $f(x)$ arbitrarily large (and positive) by taking x sufficiently close to a (on either side of a) without letting $x = a$.

There is a similar definition for $\lim_{x\to a} f(x) = -\infty$ except we make $f(x)$ arbitrarily large and negative.

Relationship between the limit and one-sided limits

$\lim_{x\to a} f(x) = L \Rightarrow \lim_{x\to a^+} f(x) = \lim_{x\to a^-} f(x) = L$ $\quad \lim_{x\to a^+} f(x) = \lim_{x\to a^-} f(x) = L \Rightarrow \lim_{x\to a} f(x) = L$

$\lim_{x\to a^+} f(x) \neq \lim_{x\to a^-} f(x) \Rightarrow \lim_{x\to a} f(x)$ Does Not Exist

Properties

Assume $\lim_{x\to a} f(x)$ and $\lim_{x\to a} g(x)$ both exist and c is any number then,

1. $\lim_{x\to a}\left[cf(x)\right] = c\lim_{x\to a} f(x)$
2. $\lim_{x\to a}\left[f(x)\pm g(x)\right] = \lim_{x\to a} f(x) \pm \lim_{x\to a} g(x)$
3. $\lim_{x\to a}\left[f(x)g(x)\right] = \lim_{x\to a} f(x)\lim_{x\to a} g(x)$
4. $\lim_{x\to a}\left[\frac{f(x)}{g(x)}\right] = \frac{\lim_{x\to a} f(x)}{\lim_{x\to a} g(x)}$ provided $\lim_{x\to a} g(x) \neq 0$
5. $\lim_{x\to a}\left[f(x)\right]^n = \left[\lim_{x\to a} f(x)\right]^n$
6. $\lim_{x\to a}\left[\sqrt[n]{f(x)}\right] = \sqrt[n]{\lim_{x\to a} f(x)}$

Basic Limit Evaluations at $\pm\infty$

Note : $\operatorname{sgn}(a) = 1$ if $a > 0$ and $\operatorname{sgn}(a) = -1$ if $a < 0$.

1. $\lim_{x\to\infty} e^x = \infty$ & $\lim_{x\to-\infty} e^x = 0$
5. n even : $\lim_{x\to\pm\infty} x^n = \infty$

Calculus Cheat Sheet

Evaluation Techniques

Continuous Functions
If $f(x)$ is continuous at a then $\lim_{x\to a} f(x) = f(a)$

Continuous Functions and Composition
$f(x)$ is continuous at b and $\lim_{x\to a} g(x) = b$ then $\lim_{x\to a} f(g(x)) = f\left(\lim_{x\to a} g(x)\right) = f(b)$

Factor and Cancel

$$\lim_{x\to2}\frac{x^2+4x-12}{x^2-2x} = \lim_{x\to2}\frac{(x-2)(x+6)}{x(x-2)} = \lim_{x\to2}\frac{x+6}{x} = \frac{8}{2} = 4$$

Rationalize Numerator/Denominator

$$\lim_{x\to9}\frac{3-\sqrt{x}}{x^2-81} = \lim_{x\to9}\frac{3-\sqrt{x}}{x^2-81}\frac{3+\sqrt{x}}{3+\sqrt{x}} = \lim_{x\to9}\frac{9-x}{\left(x^2-81\right)\left(3+\sqrt{x}\right)} = \lim_{x\to9}\frac{-1}{(x+9)\left(3+\sqrt{x}\right)} = \frac{-1}{(18)(6)} = -\frac{1}{108}$$

Combine Rational Expressions

$$\lim_{h\to0}\frac{1}{h}\left(\frac{1}{x+h}-\frac{1}{x}\right) = \lim_{h\to0}\frac{1}{h}\left(\frac{x-(x+h)}{x(x+h)}\right) = \lim_{h\to0}\frac{1}{h}\left(\frac{-h}{x(x+h)}\right) = \lim_{h\to0}\frac{-1}{x(x+h)} = -\frac{1}{x^2}$$

L'Hospital's Rule
If $\lim_{x\to a}\frac{f(x)}{g(x)} = \frac{0}{0}$ or $\lim_{x\to a}\frac{f(x)}{g(x)} = \frac{\pm\infty}{\pm\infty}$ then,

$$\lim_{x\to a}\frac{f(x)}{g(x)} = \lim_{x\to a}\frac{f'(x)}{g'(x)}$$ a is a number, ∞ or $-\infty$

Polynomials at Infinity
$p(x)$ and $q(x)$ are polynomials. To compute $\lim_{x\to\pm\infty}\frac{p(x)}{q(x)}$ factor largest power of x in $q(x)$ out of both $p(x)$ and $q(x)$ then compute limit.

$$\lim_{x\to-\infty}\frac{3x^2-4}{5x-2x^3} = \lim_{x\to-\infty}\frac{x^2\left(3-\frac{1}{x^2}\right)}{x^2\left(\frac{5}{x}-2\right)} = \lim_{x\to-\infty}\frac{3-\frac{1}{x^2}}{\frac{5}{x}-2} = -\frac{3}{2}$$

Piecewise Function

$$\lim_{x\to-2} g(x) \text{ where } g(x) = \begin{cases} x^2+5 & \text{if } x < -2 \\ 1-3x & \text{if } x \geq -2 \end{cases}$$

Compute two one sided limits,

$\lim_{x\to-2^-} g(x) = \lim_{x\to-2^-} x^2+5 = 9$

$\lim_{x\to-2^+} g(x) = \lim_{x\to-2^+} 1-3x = 7$

One sided limits are different so $\lim_{x\to-2} g(x)$ doesn't exist. If the two one sided limits had been equal then $\lim_{x\to-2} g(x)$ would have existed and had the same value.

Some Continuous Functions

Partial list of continuous functions and the values of x for which they are continuous.

1. Polynomials for all x.
2. Rational function, except for x's that give division by zero.
3. $\sqrt[n]{x}$ (n odd) for all x.
4. $\sqrt[n]{x}$ (n even) for all $x \geq 0$.
5. e^x for all x.
6. $\ln x$ for $x > 0$.
7. $\cos(x)$ and $\sin(x)$ for all x.
8. $\tan(x)$ and $\sec(x)$ provided $x \neq \cdots, -\frac{3\pi}{2}, -\frac{\pi}{2}, \frac{\pi}{2}, \frac{3\pi}{2}, \cdots$
9. $\cot(x)$ and $\csc(x)$ provided $x \neq \cdots, -2\pi, -\pi, 0, \pi, 2\pi, \cdots$

Calculus Cheat Sheet