probability

Release 1.0

Adam Richards

Dec 21, 2016

Before we dig into probability, one of the essential pillars of **data science**, lets provide some perspective. Some folks refer to data science as EDA followed by predictive modeling, but there are other schools of thought. Here is one I like:

Data science is OSEMN

- Obtaining data
- Scrubbing data
- Exploring data
- Modeling data
- iNterpreting data

At a very high level we are going to understand:

1. Basic Probability Theory
2. Probability distributions and application of probability distributions

CHAPTER ONE

A MOTIVATING EXAMPLE

Starting with some data we want to use statsitics to answer certain kinds of questions:

- How well does the data match some assumed (null) distribution [hypotehsis testing]?
- If it doesn't match well can we estiamte the parameters to approximate it [point estimate]?
- How accurate are the parameter estimates [interval estimates]?
- Can we estimate the entire distribution [function estimation or approximation]?

Most commonly, the computational approaches used to address these questions will involve

- Least squeares
- Numerical optimization
- maximum likelihood
- Numerical optimization
- Expectation maximization (EM)
- Monte Carlo methods
- Variational methods
- Simulation of null distribution (bootstrap, permutation)
- Estimation of posterior density (Monte Carlo integration, MCMC, EM)

Is my coin fair?

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as st

n = 100
pcoin = 0.62 # actual value of p for coin
results = st.bernoulli(pcoin).rvs(n)
h = sum(results)
print("we observed %s heads out of %s"%(h,n))
```

```
we observed 67 heads out of 100
The expected distribution for a fair coin is mu=50.0, sd=5.0
```

The **Expected distribution** for fair coin

```
p = 0.5
rv = st.binom(n,p)
mu = rv.mean()
sd = rv.std()
print("The expected distribution for a fair coin is mu=%s, sd=%s"%(mu,sd))
```

```
The expected distribution for a fair coin is mu=50.0, sd=5.0
```

Hypothesis testing

If we move into a hypothesis testing framework we can use the **binomial test**

```
print("binomial test - %s"%st.binom_test(h, n, p))
```

```
binomial test - 0.000873719836912
```

or a **normal approximation for binomal** (Z-test with continuity correction)

```
z = (h-0.5-mu)/sd
print("normal approx for binomial - %s"%(2*(1 - st.norm.cdf(z))))
```

```
normal approx for binomial - 0.000966848284768
```

Simulation

We **can use simulation** to test things as well

```
nsamples = 100000
xs = np.random.binomial(n, p, nsamples)
print("simulation p-value - %s"%(2*np.sum(xs >= h)/(xs.size + 0.0)))
```

```
simulation p-value - 0.00062
```

Interpretation

Can anyone interpret this p-value based on this level of significance (assuming $\alpha = 0.05$)

Maximum likelihood estimation (MLE)

```
print("Maximum likelihood %s"%(np.sum(results)/float(len(results))))
bs_samples = np.random.choice(results, (nsamples, len(results)), replace=True)
bs_ps = np.mean(bs_samples, axis=1)
bs_ps.sort()
print("Bootstrap CI: (%.4f, %.4f)" % (bs_ps[int(0.025*nsamples)], bs_ps[int(0.
 ↪975*nsamples)]))
```

```
Maximum likelihood 0.67
Bootstrap CI: (0.5800, 0.7600)
```

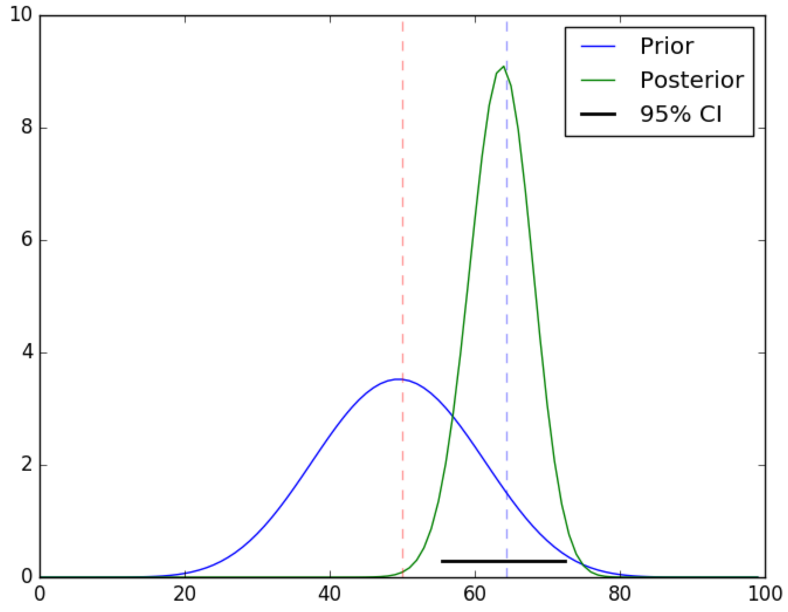
Bayesian estimation

The **Bayesian approach** directly estimates the posterior distribution, from which all other point/interval statistics can be estimated.

```
fig  = plt.figure()
ax = fig.add_subplot(111)

a, b = 10, 10
prior = st.beta(a, b)
post = st.beta(h+a, n-h+b)
ci = post.interval(0.95)
map_ =(h+a-1.0)/(n+a+b-2.0)

xs = np.linspace(0, 1, 100)
ax.plot(prior.pdf(xs), label='Prior')
ax.plot(post.pdf(xs), label='Posterior')
ax.axvline(mu, c='red', linestyle='dashed', alpha=0.4)
ax.set_xlim([0, 100])
ax.axhline(0.3, ci[0], ci[1], c='black', linewidth=2, label='95% CI');
ax.axvline(n*map_, c='blue', linestyle='dashed', alpha=0.4)
ax.legend()
plt.savefig("coin-toss.png")
```



Note: The above calculations have simple analytic solutions. For most real life problems an appropriate model is generally more complex and more complex models statistical models make use of more advanced numerical methods and simulations.

CHAPTER

TWO

MAIN CONTENTS

More specifically:

Definitions and Concepts

Sets

The range of all possible outcomes or events, also called the **sample space**.

- Coin flips
- Cookies
- Heights
- Number of slices of pizza eaten before 10 am

Set Operations

Union: $A \cup B = \{x : x \in A \vee x \in B\}$

Intersection: $A \cap B = \{x : x \in A \wedge x \in B\}$

Difference: $A \setminus B = \{x : x \in A \wedge x \notin B\}$

Complement: $A^C = \{x : x \notin A\}$

The null (empty) set: $\emptyset$

DeMorgan's Law

$\neg(A \vee B) \iff \neg A \wedge \neg B$

$\neg(A \wedge B) \iff \neg A \vee \neg B$

Note: the $\vee$ and $\wedge$ refers to the logical *or* and the logical *and*. Also $\neg$ is the negation logic operator (NOT)

Another useful notation is $\mathbf{card}(A)$ which is the *cardinality* or number of elements in *A*

Sets in Python

```
>>> a = set(["A","B","C","D"])
>>> b = set(["C","D","E","F"])
>>> c = set(["A","C","E","G"])
```

- What is the intersection of the three groups?
- What is the union of the three groups?
- What do we get if we make a union of *a* and *b* then we intersect it with *c*

Note: When we get to NumPy be reminded that there are set routines in NumPy and they are **fast**.

Combinatorics

The mathematics of ordering, choosing sets, etc. Useful for counting events in your sample space.

Factorials

If there are 10 lottery balls and we want draw them all, how many possible orderings are there?

```
>>> import math
>>> math.factorial(10)
3628800
```

Combinations

Number of ways to choose things when order doesn't matter

How many different pairs are there for afternoons sprints with 24 students?

```
>>> from itertools import combinations
>>> list(combinations("ABC",2))
[('A', 'B'), ('A', 'C'), ('B', 'C')]
```

This is also know as *N* choose *K*

It is the number of ways to select *k* objects from a pool of *n* objects

```
from math import factorial
def comb(n, k):
    return factorial(n) / factorial(k) / factorial(n - k)
```

```
>>> from scipy.misc import comb
>>> comb(3,2)
3.0
```

Permutations

Number of ways to choose things when order does matter.

On a baseball team with 20 players, how many different batting orders are there?

```
>>> from itertools import permutations
>>> list(permutations("ABC",2))
[('A', 'B'), ('A', 'C'), ('B', 'A'), ('B', 'C'), ('C', 'A'), ('C', 'B')]
```

Note: High speed is retained by preferring *vectorized* building blocks over the use of for-loops and generators which incur interpreter overhead.

itertools also has a *groupby* that works like pandas.

Probability

Probability provides the mathematical tools we use to model randomness:

- Probability tells us how likely an event (Frequentist) or what our degree of beliefs in an event is (Bayesian)
- Provides the foundation for statistics and machine learning
- Often our intuitions about randomness are incorrect because we live only one realization
- Enumerating all possible outcomes (using combinatorics) can help us compute the probability of an event

Formalization

For some sample space *S*, a probability function *P* has three properties:

$P(A) \geq 0 \forall A \in S$

$P(S) = 1$

$\forall A_i, A_j : A_i \cap A_j = \emptyset \Rightarrow P(A_i \cup A_j) = P(A_i) + P(A_j)$

Independence

Events are independent (notation $A \perp B$) if:

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

or

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

The above is known as **conditional probability**.

- How could we use the definition of independence to test whether two events are independent?
- **What does knowing that B has occurred tell us about the likelihood of A?**
  - Under independence?
  - Without independence?

The $P(A \cap B) = P(A)P(B)$ is known as the multiplication rule

A problem

Take a moment to solve this question:

- Three types of fair coins are in an urn: HH, HT, and TT
- You pull a coin out of the urn, flip it, and it comes up H
- Q: what is the probability it comes up H if you flip it a second time?

So does one event provide information about the other.

Solution:

$$P(X_1 = H) = 1/2$$
$$P(X_2 = H|X_1 = H) = \frac{5}{6} \neq \frac{1}{2} = P(X_2 = H)$$

Conditional probability

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

in other words

$P(X_2 = H|X_1 = H) = P(X_2 = H \cap X_1 = H)/P(X_1 = H) = \frac{\frac{1}{3} + \frac{1}{3}\frac{1}{4}}{\frac{1}{2}}$

```python
import random
import pandas as pd

coins = ['HH', 'HT', 'TT']
results = []
for i in range(10000):
    coin = random.choice(coins)
    results.append([random.choice(coin) for j in [1,2]])
df = pd.DataFrame(results, columns=['first', 'second']) == 'H'
df.groupby('first').mean()
```

```
       second
first
False  0.168256
True   0.838502
```

Probability Chain Rule

Note: In probability theory, the chain rule (also called the general product rule) permits the calculation of any member of the joint distribution of a set of random variables using only conditional probabilities.

We can rearrange the formula for conditional probability to get the product rule:

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

We can extend this for three or more variables:

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

More generally:

$$P(\cap_i^n X_i) = \prod_i^n P(X_i | \cap_k^{i-1} X_k)$$

Law of Total Probability

If $\{B_n\}$ is a partition of a sample space A, meaning $\cup_i B_i = A$ and $B_i \cap B_j = \emptyset \forall i, j$

Then

$P(A) = \sum P(A \cap B_i)$

or

$P(A) = \sum P(A|B_i)P(B_i)$

And we call A the **marginal distribution** of B

Bayes Rule

Use Bayes's Rule when you need to compute conditional probability for $A|B$ but only have probability for $B|A$:

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

$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)}$

Proof: use the definition of conditional probability

Recall that

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

Lets start with the conditional probability defination

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

If we write the reverse of that

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

Then multiply by $P(A)$

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

Then plug this back into the conditional probability.

Bayesian inference works by combining information about parameters θ contained in the observed data x as quantified in the likelihood function $p(x|\theta)$. Classical statistics works by making inference about a single point, while Bayesian inference works on the whole distribution. Parameters through the Bayesian lens are treated as random variables described by distributions.

Lets put Bayesian inference on hold and first look at and example of Bayes Rule.

Predictive value positive - Prob. person has disease given the test was positive. $PV^+ = P(D^+|T^+)$

Predicitve value negative - Prob. person does not have diease given test was negative $PV^- = P(D^-|T^-)$

Sensitivity - Prob. that test positive given person has disease $P(T^+|D^+)$

Specificity - Prob. that test negative given person does not have disease $P(T^-|D^-)$

Prevalance - $d = P(D^+)$

Note that: $P(T+|D-) = 1 - \text{specificity}$

Lets say we wanted to know PV^+.

$$P(D^+|T^+) = \frac{P(T^+|D^+)P(D^+)}{P(D^+)P(T^+|D+) + P(D^-)P(T^+|D^-)} \quad (2.1)$$

$$= \frac{d \times \text{sensitivity}}{d \times \text{sensitivity} + (1-d) \times (1 - \text{specificity})} \quad (2.2)$$

So if we were given

Sensitivity = 0.84, specificity = 0.77, prevalence = 0.20

Then

$$PV^+ = \frac{(0.2)(0.84)}{(0.2)(0.84) + (0.8)(0.23)} = 0.48$$

$$PV^- = \frac{(0.8)(0.77)}{(0.8)(0.77) + (0.2)(0.16)} = 0.95$$

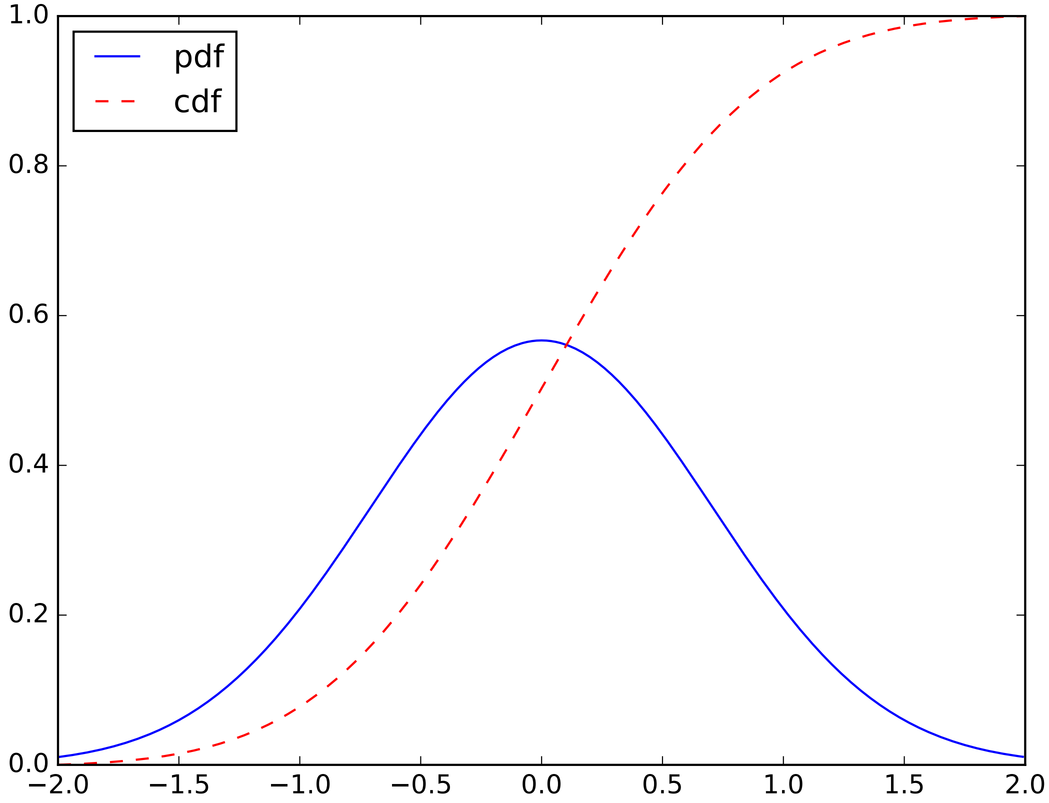
Random Variables

Random variables formalize a mapping we have been implicitly using already:

$X(s) : S \Rightarrow \Re$

- Capital letters refer to random variables.
- Lowercase to refer to specific realization.
- $P(X = x) = P(\{s \in S : X(s) = x\})$
- *X sim XYZ(alpha, beta, ...)* means X is distributed as, XYZ with parameters.
- "i.i.d."

PDFs and CDFs



Cumulative distribution function

$F_X(x) = P(X < x)$

- What kinds of bounds can we put on this function?
- This works for both continuous and discrete functions.

Probability mass function, PMF

For discrete variables:

$f_X(x) = P(X = x), \forall x$

For continuous variables, think of it as the derivative of the CDF:

$f_X(x)dx = P(x < X < x + dx)$

$f_X(x) = \frac{dF_X(x)}{dx}$

Expectation

Discrete: $E[X] = \sum_{s \in S} X(s) f_X(s)$

Continuous: $E[X] = \int_{-\infty}^{\infty} X(s) f_X(s) ds$

A measure, but not the only one, of the central tendecy of a distribution. Alternatives?

> Note, the sample mean is:
>
> $\bar{x} = \frac{1}{n} \sum_j^n x_j$

Variance

$Var[x] = E[(x - E[X])^2]$

What are the units?

Note, the sample variance is:

$s^2 = \frac{1}{n-1} \sum_j^n (x_j - \bar{x})^2$

Standard deviation

$\sigma(x) = \sqrt{Var[x]}$

Useful because its units are in units of our original RV.

Covariance

We can also compute the covariance between two different variables:

$Cov[X, Y] = E[(x - E[X])(y - E[Y])]$

Which is related to the

Correlation

$Corr[X, Y] = \frac{E[(x-E[X])(y-E[Y])]}{\sigma(X)\sigma(Y)} = \frac{Cov[X,Y]}{\sigma(X)\sigma(Y)}$

Anscombe’s quartet

```
from numpy import array, amin, amax

def fit(x):
   return 3+0.5*x

def anscombe():
    x =  array([10, 8, 13, 9, 11, 14, 6, 4, 12, 7, 5])
    y1 = array([8.04, 6.95, 7.58, 8.81, 8.33, 9.96, 7.24, 4.26, 10.84, 4.82, 5.68])
    y2 = array([9.14, 8.14, 8.74, 8.77, 9.26, 8.10, 6.13, 3.10, 9.13, 7.26, 4.74])
    y3 = array([7.46, 6.77, 12.74, 7.11, 7.81, 8.84, 6.08, 5.39, 8.15, 6.42, 5.73])
    x4 = array([8,8,8,8,8,8,8,19,8,8,8])
    y4 = array([6.58,5.76,7.71,8.84,8.47,7.04,5.25,12.50,5.56,7.91,6.89])
    xfit = array( [amin(x), amax(x) ] )
```

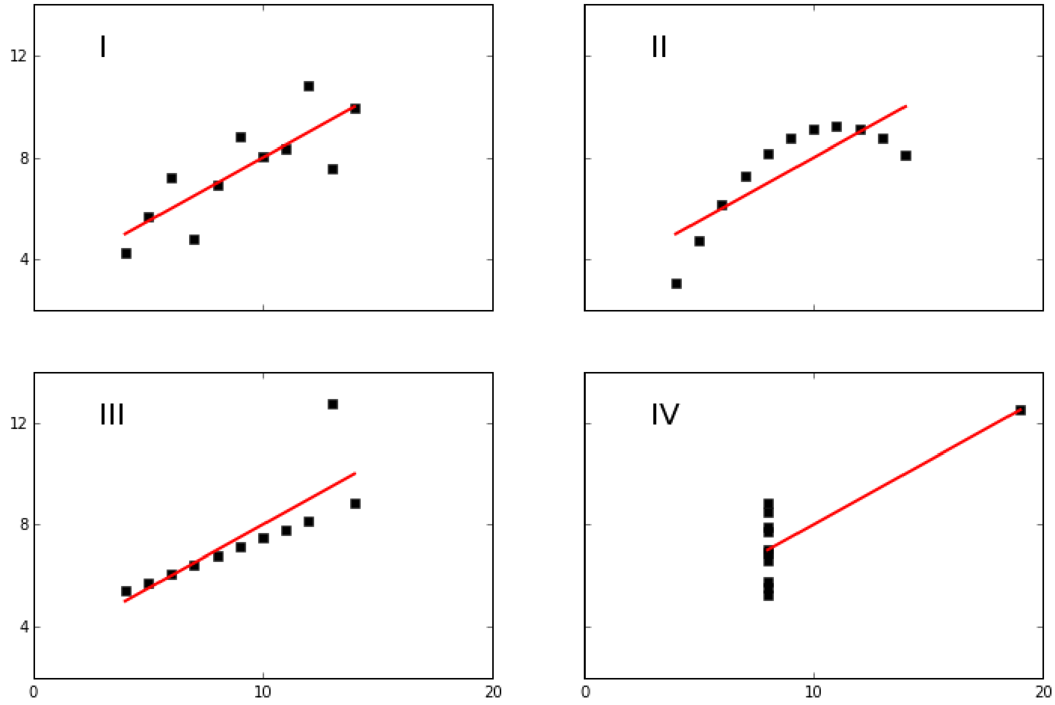
```
figure(figsize(12,8))
subplot(221)
plot(x,y1,'ks', xfit, fit(xfit), 'r-', lw=2)
axis([2,20,2,14])
setp(gca(), xticklabels=[], yticks=(4,8,12), xticks=(0,10,20))
text(3,12, 'I', fontsize=20)

subplot(222)
plot(x,y2,'ks', xfit, fit(xfit), 'r-', lw=2)
axis([2,20,2,14])
setp(gca(), xticklabels=[], yticks=(4,8,12), yticklabels=[], xticks=(0,10,20))
text(3,12, 'II', fontsize=20)

subplot(223)
plot(x,y3,'ks', xfit, fit(xfit), 'r-', lw=2)
axis([2,20,2,14])
text(3,12, 'III', fontsize=20)
setp(gca(), yticks=(4,8,12), xticks=(0,10,20))

subplot(224)
xfit = array([amin(x4),amax(x4)])
plot(x4,y4,'ks', xfit, fit(xfit), 'r-', lw=2)
axis([2,20,2,14])
setp(gca(), yticklabels=[], yticks=(4,8,12), xticks=(0,10,20))
text(3,12, 'IV', fontsize=20)

#verify the stats
pairs = (x,y1), (x,y2), (x,y3), (x4,y4)
for x,y in pairs:
    print ('mean=%1.2f, std=%1.2f, r=%1.2f'%(mean(y), std(y), corrcoef(x,y)[0][1]))
```

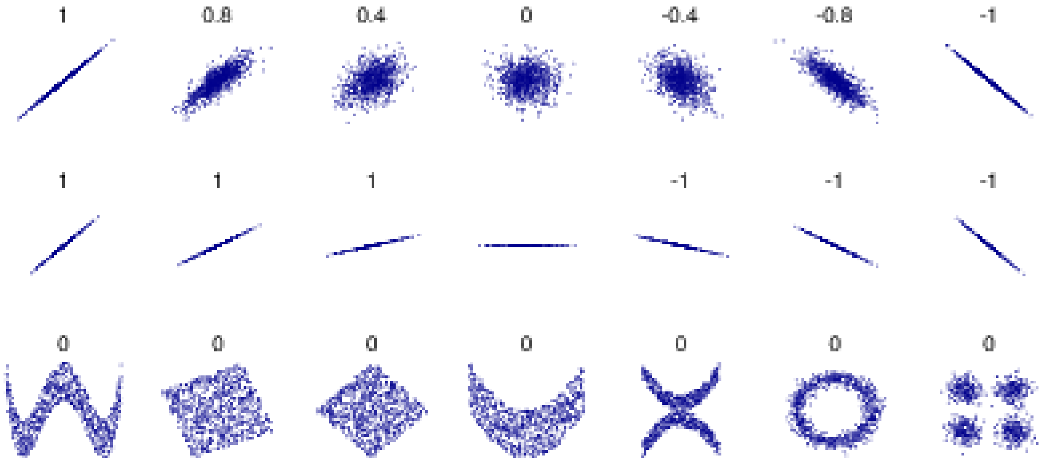


```
mean=7.50, std=1.94, r=0.82
mean=7.50, std=1.94, r=0.82
mean=7.50, std=1.94, r=0.82
mean=7.50, std=1.94, r=0.82
```

Correlation

A **spurious relationship** is a relationship where two or more events, that are not causally related to each other have a relationship. This may be due to a “common response variable” or a “confounding factor”.

Correlation coefficients vary between -1 and +1 with 0 implying no correlation.



Pearson

```
>>> from scipy.stats import pearsonr
>>> pearsonr([1,2,3,4,5],[5,6,7,8,7])
(0.83205029433784372, 0.080509573298498519)
```

The Pearson correlation coefficient measures the linear relationship between two datasets. The p-value roughly indicates the probability of an uncorrelated system producing datasets that have a Pearson correlation at least as extreme as the one computed from these dataset.

In other words null hypothesis is that two sets of data are uncorrelated.

Spearman

```
>>> from scipy.stats import spearmanr
>>> spearmanr([1,2,3,4,5],[5,6,7,8,7])
(0.82078268166812329, 0.088587005313543812)
```

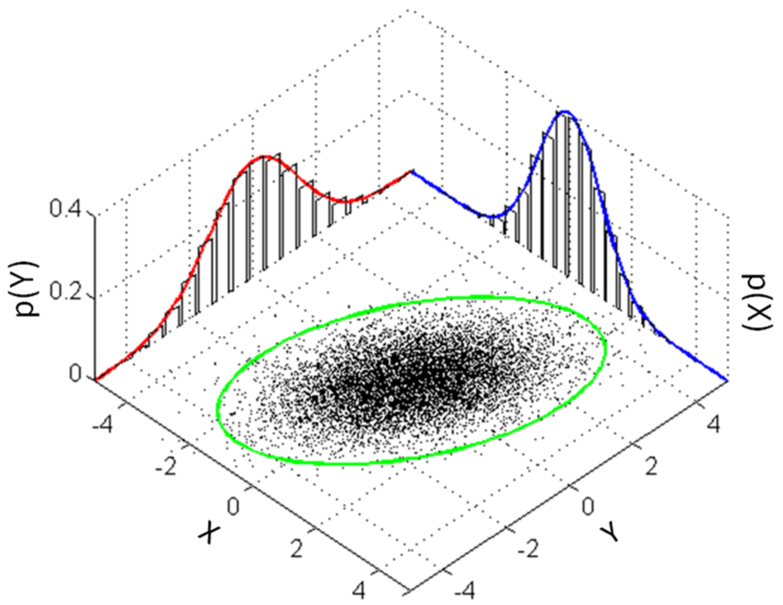
The Spearman correlation is a nonparametric measure of the monotonicity of the relationship between two datasets. Unlike the Pearson correlation, the Spearman correlation does not assume that both datasets are normally distributed.

Marginal Distributions

Marginal distribution takes a–possibly not independent–multivariate distribution. And considers only a single dimension.

Accomplished by summing (discrete) or integrating (continuous).

$$f_X(x) = \int_{-\infty}^{\infty} f_{XY}(x, s) ds$$



Discrete case

•	x1	x2	x3	x4	py(Y)
y1	$\frac{4}{32}$	$\frac{2}{32}$	$\frac{1}{32}$	$\frac{1}{32}$	$\frac{8}{32}$
y2	$\frac{2}{32}$	$\frac{4}{32}$	$\frac{1}{32}$	$\frac{1}{32}$	$\frac{8}{32}$
y3	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{8}{32}$
y4	$\frac{8}{32}$	0	0	0	$\frac{8}{32}$
px(X)	?	?	?	?	?

Conditional Distributions

$$f(y|x) = \frac{f_{XY}(x, y)}{f_X(x)}$$

Probability distributions

The SciPy docs for statistics is quite useful.

Properties of distributions

Use these properties to characterize a distribution:

- Expectation/mean
- Variance/standard deviation
- Skewness (asymmetry)
- Kurtosis (fat tails)
- Correlation

Rules for choosing a good distribution

Main questions

- Are my data discrete or continuous?
- Are my data symmetric?
- What limits are there on possible values for my data?

Other questions to keep in mind

- How likely are extreme values?
- Are there missing values?

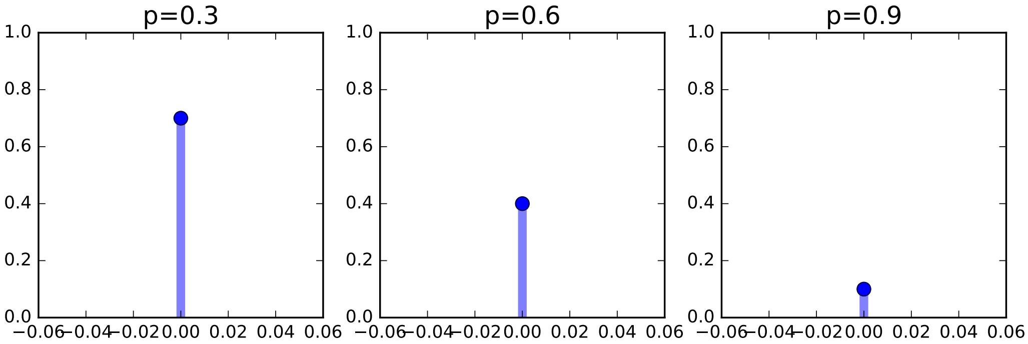
Bernoulli:

A Bernoulli distribution is a discrete probability distribution for a Bernoulli trial. The distribution takes the value 1 with success probability of p and the value 0 with failure. Success could be heads on a coin flip.

PMF = $P[success] = p$, $P[failure] = 1 - p$

Mean: $E[x] = p$

Variance: $Var(x) = p(1 - p)$



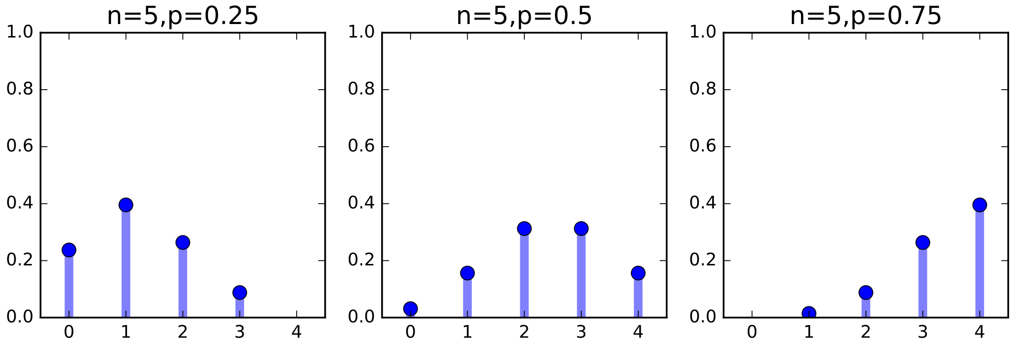
Binomial:

The Binomial distribution gives the discrete probability distribution of obtaining exactly *p* successes out of *n* trials

PMF: $P[X=k] = \binom{n}{k} p^k (1-p)^{n-k}, \forall k \in \{0, 1, ..., n\}$

Mean: np

Var: $np(1-p)$



Geometric:

The probability of some number (*X*) of Bernoulli trials needed to get one success. It also refers to probability of (*X-1*) failures before the first success.

PMF: $P[X=k]=p(1-p)^{k-1}, \forall k \in \{0,1,...\}$

Mean: $\frac{1}{p}$

Variance: :*frac{1-p}{p^2}*

Hypergeometric

Hypergeometric distribution is a discrete probability distribution that describes the probability of *k* successes in *n* draws, without replacement.

The hypergeometric test uses the hypergeometric distribution to calculate the statistical significance of having drawn a specific k successes n total draws

Think of an urn with two types of marbles, red ones and green ones. Define drawing a green marble as a success and drawing a red marble as a failure (analogous to the binomial distribution).

Did I draw the **expected** number of green marbles?

The data are not accurately modeled by the binomial distribution, because the probability of success on each trial is not the same.

Note: Think Texas Hold em

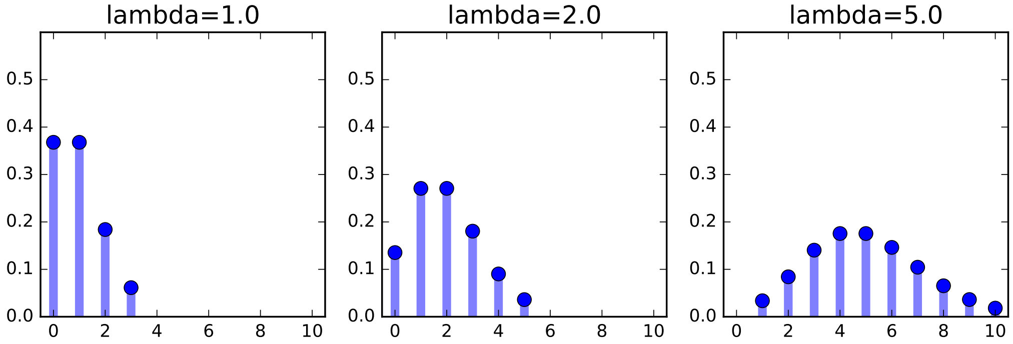
Poisson

If a mean of an event happening per unit time is observed and you need the probability of n events happening

PMF: $P[X = k] = \frac{\lambda^k e^{-\lambda}}{k!}, \forall k \in \{0, 1, 2, ...\}$

Mean: λ

Variance: λ



Exponential

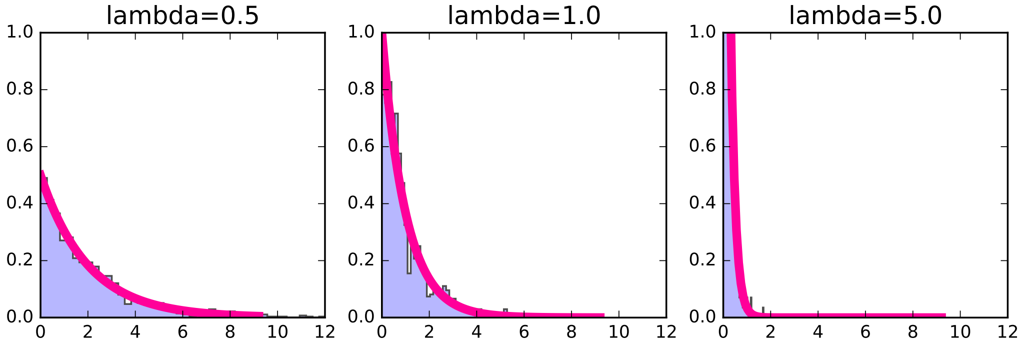
A good way to model the time between events for a poisson process. It is a particular case of the gamma distribution. It is governed by a rate parameter λ.

SUPPORT: $x \in (0, \inf)$.

PDF: $\lambda e^{-\lambda x}$

MEAN: $\frac{1}{\lambda}$

VARIANCE: $\frac{1}{\lambda^2}$



Uniform

PDF: $f(x) = \frac{1}{b-a}, \forall x \in [a, b]$, 0 otherwise

MEAN: $\frac{a+b}{2}$

VARIANCE: $\frac{(b-a)^2}{2}$

Normal aka Gaussian

The Gaussian is the most widely used distribution for continuous variables. The distribution is governed by the mean μ and variance :*sigma*^2.

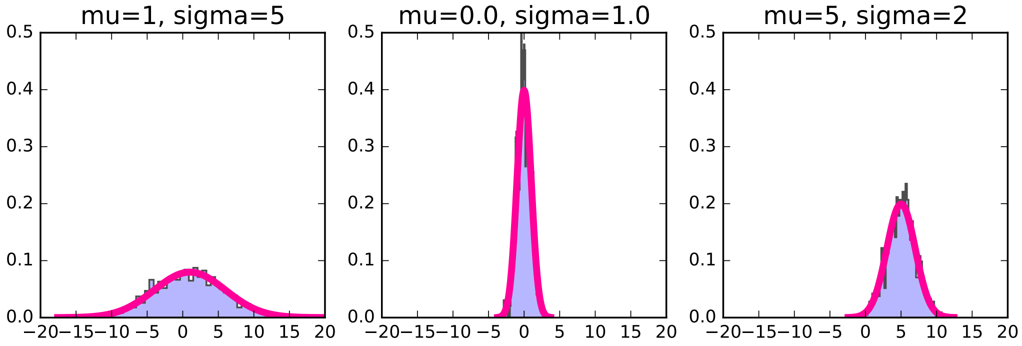
SUPPORT $x \in (-\inf, \inf)$

PDF: $\frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(x-\mu)^2}{2\sigma^2})$

MEAN: μ

VARIANCE: σ^2

The inverse of the variance is known as the **precision** ($\tau = 1/\sigma^2$).



Note: Tomorrow we get into the central limit theorem and we will start to learn how important this distribution can be.

Beta Distribution

The Beta density function is a very versatile way to represent outcomes like proportions or probabilities. It works on a space between between 0 and 1.

There are two parameters which work together to determine if the distribution has a mode in the interior of the unit interval and whether it is symmetrical.

PDF:

a value x on the interval (0,1):

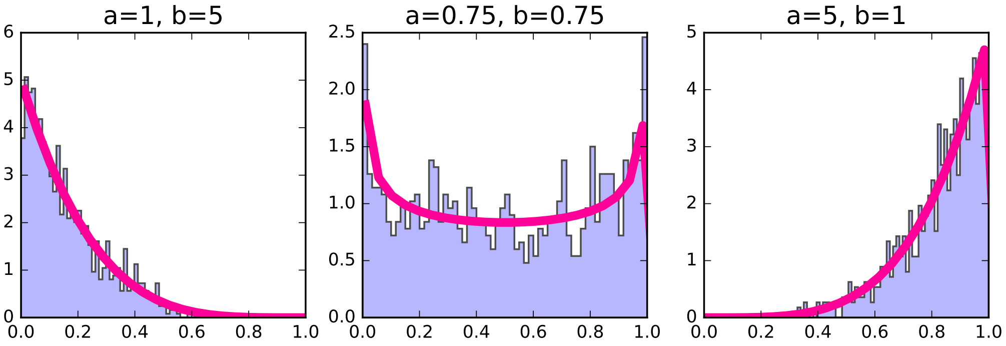
$$Beta(\alpha,\beta):\ prob(x|\alpha,\beta)=\frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)}$$

where B is the **beta function**

$$B(\alpha,\beta)=\int_0^1 t^{\alpha-1}(1-t)^{\beta-1}dt$$

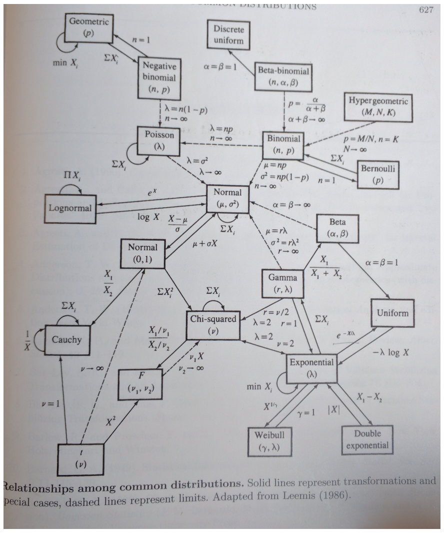
Mean: $\frac{a}{a+b}$

Variance: $\frac{ab}{(a+b)^2(a+b+1)}$



Distributions are related

There are many more distributions than the ones mentioned above. Here is an illustration from *Casella and Berger* that does a pretty good job making that point.



Relationships among common distributions. Solid lines represent transformations and pecial cases, dashed lines represent limits. Adapted from Leemis (1986).