**Conjoint Probability:**

Writing means, the probability of occurrences A and B being both true.

In case of two independent occurrences, such as tossing a coin, the probability of both 2 occurrences to have heads is:

when two occurances are independent

However, if two occurrences are dependent, such as A being raining today and B being raining tomorrow, it is intuitive to infer that .

Thus, the probability of a conjunction is:

**Bayes Theorem**

Bayes theorem can be derived by picking up from above concept.

so, rewriting this is:

which leads to Bayes Theorem:

A simple example which utilizes this equation is, the cookie problem:

Suppose there are two bowls of cookies. Bowl 1 contains 30 vanilla cookies and 10 chocolate cookies. Bowl 2 contains 20 of each.

Now suppose you choose one of the bowls at random and, without looking, select a cookie at random. The cookie is vanilla. What is the probability that it came from Bowl 1?

Denoting B1 for the hypothesis that the cookie came from Bowl 1 and V for the vanilla cookie, we could write the problem as solving following equation:

Then assuming selecting either Bowl 1 or 2 is random, .

Selecting vanilla cookie from Bowl 1 is and selecting vanilla cookie out of total cookies, .

Thus,

**Bayesian Theorem with Diachronic Interpretation**

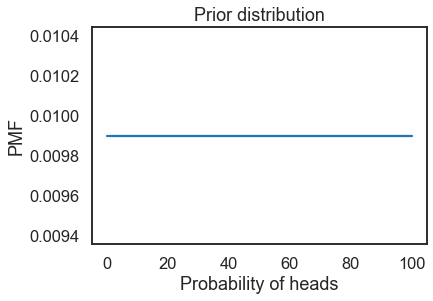
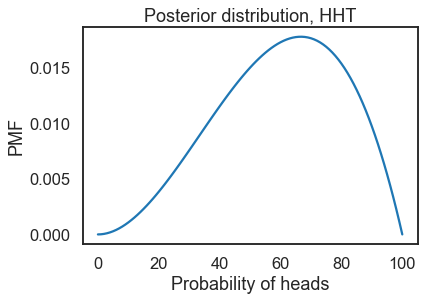
A more general way of interpreting Bayesian Theorem is to utilize it for updating the probability of a hypothesis, *H*, in light of some body of data, *D*.

This way of thinking about Bayes’s theorem is called the diachronic interpretation. “Diachronic” means that something is happening over time; in this case the probability of the hypotheses changes, over time, as we see new data.

In this interpretation, each term has a name:

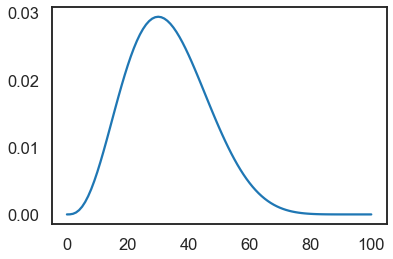
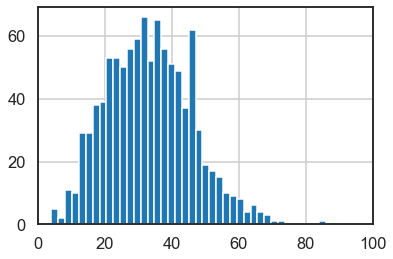
* is the probability of the hypothesis before we see the data, called the prior probability, or just **prior**.
* is what we want to compute, the probability of the hypothesis after we see the data, called the **posterior**.
* is the probability of the data under the hypothesis, called the **likelihood**.
* is the probability of the data under any hypothesis, called the **normalizing constant**

**Examples of Bayesian Statistics using Python**

1. Euro Problem  
     
     
   This problem requires proceeding in two steps. The first is to estimate the probability that the coin lands face up. The second is to evaluate whether the data support the hypothesis that the coin is biased.  
     
   Any given coin has some probability, *x*, of landing heads up when spun on edge. If a coin is perfectly balanced, we expect x to be close to 50%, but for a lopsided coin, x might be substantially different.  
     
   The key point in using Bayesian Statistics for solving this problem is not about the search for the exact value of *x*, per se, however the distribution of *x*.  
     
   Thus, *x* should be on a range between 0 to 100, where is the hypothesis that probability of heads is . Within python setup, one could starting the prior with a uniform distribution where the probability of is the same for all *x*, as such:  
     
      
     
   Then with incoming data, new information can be updated into the likelihood probability of coin toss resulting in heads with a function like:  
   If is true, the probability of heads is and the probability of tails is .  
     
   **def** Likelihood(self, data, hypo):  
    x = hypo  
    **if** data == ‘H’**:**  
    return x / 100.0  
    **else:**  
    return 1 – x / 100.0  
     
   If so, having updated with 3 sequential occurrences data, ie. **Head –** **Head -** **Tail**, the updated posterior distribution would look like:  
     
      
     
   In such matter, one could argue with 3 input data that, the maximum probability for having a head with a coin toss is 67% with a general average probability of 60% likelihood.  
     
   Again, the key take-away from this exercise is that, while one wishes to find the probability for the result of a coin toss, the output result from a Bayesian statistical analysis is not a single value but rather a probability distribution which holds the information for the general likelihood.

"When spun on edge 250 times, a Belgian one-euro coin came up heads 140 times and tails 110. 'It looks very suspicious to me,' said Barry Blight, a statistics lecturer at the London School of Economics. 'If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.' "

From “The Guardian” quoted by MacKay, Information Theory, Inference, and Learning Algorithms.

1. Random Selection with a Custom Probability Distribution  
     
   Numpy library in python, gives some powerful tools for generating random selection data.  
     
   Let’s say we would like to select a number between 0~100, under a custom probability distribution, such that the distribution could look like below:  
     
      
     
   Then, numpy has following function to make a random selection within above distribution:  
     
     
   Using this, one can create a 1000 sample of selections as following:  
     
      
   As can be observed, the sample roughly follows the distribution as designed.

np.random.choice(a, size=None, replace=True, p=None)

key-Parameters

-----------

a : 1-D array-like or int

If an ndarray, a random sample is generated from its elements.

If an int, the random sample is generated as if a were np.arange(a)

p : 1-D array-like, optional

The probabilities associated with each entry in a.

If not given the sample assumes a uniform distribution over all

entries in a.

1. Bandit Problem  
     
     
   Considering that one could argue that an investment strategy is analogical to creating a pick-and-choose slot-machine strategy, this example is crucial for applying Bayesian Statistics for investment strategy.  
     
   All being equal in process with that of the “euro problem,”

Suppose you have several "one-armed bandit" slot machines, and reason to think that they have different probabilities of paying off.

Each time you play a machine, you either win or lose, and **you can use the outcome to update your belief about the probability of winning.**

Then, to decide which machine to play next, you can use the "Bayesian bandit" strategy.