# Bayesian Machine Learning:
# Probabilistic Reasoning and Programming for Machine Learning with Applications in Python

William F. Basener and Don E. Brown

January 25, 2022

# Chapter 1

# Probability

The notion of probability is understood by everyone from an early age. Children roll a die expecting each of the numbers from 1 through 6 to appear with equal probability. A coin flip is used to determine a choice at random, with the intuition that the heads and tails sides of the coin are equally likely to appear.

With some very simple math, we understand that the probability of getting a heads (henceforth called an H) on the coin is 0.5, as is the probability of getting a tails (henceforth, T). The coin flip has two possible outcomes, H and T, and the sum of the probabilities of these is 1.

Similarly, rolling a die has six possible outcomes $\{1, 2, 3, 4, 5, 6\}$, and the probability of each is $1/6$. With more than two outcomes we can ask more interesting questions like what is the probability of getting a 1 or 2, which we compute by adding the probabilities for the individual events: $1/6 + 1/6 = 1/3$; or what is the probability of getting an even number (which is $1/2$).

## 1.1 Probability Definitions

To give a rigorous foundation for probabilities, we consider the set of all possible outcomes (of some experiment or activity) and assign a number between 0 and 1 to each possible outcome, which represents the probability of the outcome. It is important to be able to assign a probability to a set of possible outcomes, for example rolling a 1 or 2 on a die, so we speak of the probability of an event occurring, where an event is a subset of possible outcomes. A probability distribution is what we use to assign probabilities to different events, defined rigorously as follows

> **Definition 1: Probability Distribution**
>
> A **probability distribution** is a set $S$ (called the domain, sample space, or state space) together with a real-valued function $P$ whose domain is all possible subsets of $S$, where furthermore the function $P$ satisfies the following:
>
> 1. $0 \leq P(E) \leq 1$ for every $E \subset S$
>
> 2. $P(S) = 1$
>
> 3. For any disjoint collection of subsets $\{S_i\}$ of $S$, $P(\cup_i S_i) = \sum_i P(S_i)$.

A probability distribution represents a random process that generates a single outcome $x$ from $S$, and the probability of a subset $E$ is the probability that the outcome $x$ is in $E$. The reader should take the time to confirm that the intuitive probabilities of flipping a coin (where $S = \{H, T\}$) or rolling a die ($S = \{1, 2, 3, 4, 5, 6\}$) satisfy these definitions. A subset $E$ is called an *event*, and may correspond to a

individual outcome occurring (rolling a 2 on a die, $E = \{2\}$) or correspond to an outcome being in a some collection of potential outcomes (rolling an even number, $E = \{2, 4, 6\}$).

Probabilities can be considered analogous to area; we can think of the sample space $S$ as a rectangle with area equal to one, each $E$ as subset of the rectangle, and $P(E)$ equal to the area of the region $E$. In this analogy, we consider a random process which has equal probability for all elements in $S$ (say, throwing a dart at a $S$) so that the probability of getting an outcome in $E$ is equal to the area of $E$. Axiom one corresponds to the area of a subset being in $[0, 1]$, axiom 2 corresponds to the total area of the rectangle being one, and axiom 3 is true for area: the area of the union is equal to the sum of the areas. In fact, it would be possible to develop the theory of probability from the mathematical field of measure theory, which gives a rigorous method for computing area, or measure, of subsets.

The most common way to define a probability distribution is by defining a **random variable**, which is typically denoted by a capital letter (generally $X$ or $Y$). The random variable may take on any value from a given sample space $S$ and has defined probabilities for each of these potential values. For example, the toss of a coin is modeled with a sample space $S = \{H, T\}$, a random variable $X$ which may take on the values $H$ or $T$, and a distribution satisfying $P(X = H) = 0.5$ and $P(X = T) = 0.5$. We usually use a lower case letter to denote a variable which is the outcome of the random variable. For example, rolling a die is modeled with a sample space $S = \{1, 2, 3, 4, 5, 6\}$, random variable $X$ that can take on values in $S$, and $P(X = x) = 1/6$ for any $x \in \{1, 2, 3, 4, 5, 6\}$. (All coins and die are fair unless otherwise noted.) The sample space associated with a random variable $X$ is called the **domain** of $X$, denoted $\mathrm{dom}(X)$. A random variable is usually defined in terms of a probability mass function or probability density function, defined as follows.

---

**Definition 2: Discrete Random Variable**

A **discrete random variable** is a random variable $X$ whose sample space $S = \mathrm{dom}(X)$ is either finite or countably infinite. That is, either $S$ is finite or we can index every element of $S$ using the positive integers: $S = \{s_i\}_{i=1}^{\infty}$.

For a discrete random variable, we define the probability distribution (which takes subsets of $S$ as inputs) using a **probability mass function** (or PMF) $p : S \to [0, 1]$ and defining the probability distribution $P$ by $P(A) = \sum_{x \in A} p(X = x)$ for any subset $A \subset S$. Note that $P(X = x) = p(x)$ for any $x \in X$.

---

**Definition 3: Continuous Random Variable**

A **continuous random variable** is a random variable $X$ whose sample space $S = \mathrm{dom}(X)$ is a subset of the real line, $S \subset \mathbb{R}$, or more generally a subset of $n$-dimensional space $\mathbb{R}^n$.

For a continuous random variable, we define the probability distribution using a **probability density function** (or PDF) $p : S \to \mathbb{R}^{\geq 0}$ and defining the probability distribution $P$ by $P(A) = \int_A p(x) dx$ for any subset $A \subset S$. Note that $P(X = x) = p(x) = 0$ for any single value $x \in S$. Often the domain is either an interval or the whole real line ($S = [a, b]$ or $S = \mathbb{R}$).

---

We will use the sum or integral notation for discrete or continuous random variables generally with the understanding that the notation generalizes for both discrete and continuous random variables.

We often drop the explicit notation for the random variable inside $P(\cdot)$ when the variable can be understood from context; in general we write $P(A) = P(X \in A)$. The first notation is more consistent with definition Definition 1 emphasizing the role of the set $A$. The latter emphasizes the role of the random variable. For example, the probability that a roll of a die is even is $P(X \in \{2, 4, 6\}) = P(\{2, 4, 6\})$. Also, we usually do not distinguish between a single element $s \in \mathrm{dom}(X)$ and the set continuing a single element, $\{s\} \subset \mathrm{dom}(X)$, so that for example $P(2) = P(\{2\}) = 1/6$ for a roll of a die. More generally, we use any reasonable notation to describe the event set inside $P$, for example we could write the probability that $X$ is even as $P(X \text{ is even})$.

Standard notation also includes

$$P(A \cup B) = P(X \in A \text{ or } X \in B) = P(A \text{ or } B)$$

and

$$P(A \cap B) = P(X \in A \text{ and } X \in B) = P(A \text{ and } B).$$

Axiom 3 from Definition 1 implies that

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

A central concept in probability is computing the probability that one condition is true about the outcome given that some other condition is known to be true, defined as follows:

---

**Definition 4: conditional probability**

For a probability distribution $P$, for events $A$ and $B$ the **conditional probability** of $A$ given $B$ is defined to be

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, \tag{1.1.0.1}$$

if $P(B) \neq 0$. The value of $P(A|B)$ is undefined if $P(B) = 0$. This can be written the equivalent formula

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

called the **product rule**.

---

The definition of conditional probability has a useful interpretation relating the probability of an event to its area shown in the figure to the right. Shown is a rectangle representing the sample space $S$ and two circles representing subsets for events $A$ and $B$. The conditional probability $P(A|B)$ is the probability that the outcome is in $A$ given that the outcome is in $B$. Interpreting probability of an event as the area of the event, the conditional probability is then the fraction of the area in $B$ that is occupied by $A \cap B$, which is the value given in Equation 1.1.0.1.



---
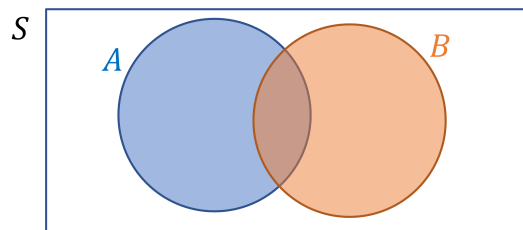
**Example 1: Test for Cancer**

Suppose that for a given type of cancer, the fraction of people who have the cancer and test positive for it are shown in the table below where the column corresponds to does not have cancer ($C = 0$) and has cancer ($C = 1$) and the row corresponds to test is negative ($T = 0$) and test is positive ($T + 1$).

|   |   | C | |
|---|---|-------|-------|
|   |   | 0 | 1 |
| T | 0 | 0.945 | 0.001 |
|   | 1 | 0.045 | 0.009 |

Use this distribution to answers the following questions.

1. If a person has cancer, what is the probability that they will test positive?

   ANSWER: The question is asking for $P(T = 1|C = 1)$. From Equation 1.1.0.1,

   $$P(T = 1|C = 1) = \frac{P(T = 1 \text{ and } C = 1)}{P(C = 1)} = \frac{0.009}{0.009 + 0.001} = 0.9. \tag{1.1.0.2}$$

---

2. What is the probability has cancer given that the test is positive?

ANSWER: The question is asking for $P(C = 1|T = 1)$. From Equation 1.1.0.1,

$$P(C = 1|T = 1) = \frac{P(C = 1 \text{ and } T = 1)}{P(T = 1)} = \frac{0.009}{0.045 + 0.009} \approx 0.17. \tag{1.1.0.3}$$

This example illustrates a principle that is common in medical testing. Test are designed so that if a person has the disease, there is a small chance that the disease mill be missed by the test because of the negative consequences of missing the disease. This often means that a significant fraction of people who test positive do not in fact have the disease, but these types of errors are acceptable because they are less costly.

We often want to consider a sample space where each outcome is determined by multiple variables. This was the case in Example 1, where each outcome was determined by the value of $C$ and $T$. We call the probability distribution for two or more variables the joint probability distribution, defined as follows.

---

**Definition 5: Marginal Distribution**

Suppose that each point in a sample space $S$ is determined by the values of two random variable $X$ and $Y$. The probability distribution $P(X, Y)$ is called a **joint probability distribution** defined by

$$P(X \in A, Y \in B) = P(X \in A \text{ and } Y \in B) \tag{1.1.0.4}$$

If $X$ and $Y$ are discrete random variables, the probability for individual values $x \in \text{dom}(X)$ and $y \in \text{dom}(Y)$ can be written as

$$P(X = x, Y = y) = P(X = x \text{ and } Y = x). \tag{1.1.0.5}$$

---

The order of the variables is not meaningful in our joint probability notation, $P(X, Y) = P(Y, X)$, and the reader should be careful to not interpret this comma as a vector of standard function notation $f(x, y)$ where order does matter. The case with more than two random variables, for example $P(X, Y, Z)$, is defined similarly

The following gives the probability distribution of one variable out of a joint distribution, separated from the values of the other variables.

---

**Definition 6: Marginal Distribution**

For a joint distribution $p(X, Y)$, the **marginal distribution** is the is the distribution for $X$ defined for discrete distributions by

$$P(A) = P(X \in A) = \sum_{y \in \text{dom}(Y)} P(X \in A, Y = y) \tag{1.1.0.6}$$

and for continuous distributions by

$$P(A) = P(X \in A) = \int_{\text{dom}(Y)} P(X \in A, Y = y) \mathrm{dy}. \tag{1.1.0.7}$$

---

**Example 2: Joint Distribution in a Table**

The table below defines the joint probability distribution for two variables $X$ and $Y$, where $dom(X) = \{A, B\}$ and $dom(Y) = \{1, 2, 3, 4, 5\}$.

|   |   | Y | | | | |
|---|---|------|------|------|------|-----|
|   |   | 1 | 2 | 3 | 4 | 5 |
| $X$ | A | 0.1 | 0.2 | 0.05 | 0.05 | 0.1 |
|   | B | 0.15 | 0.05 | 0 | 0.1 | 0.2 |

Use this distribution to answers the following questions.

1. Compute the marginal distributions $p(X)$ and $p(Y)$.

    ANSWER: From Equation 1.1.0.7 in the definition of the marginal distribution, we compute

    $$P(Y = 1) = P(Y = 1, X = A) + P(Y = 1, X = B) = .1 + 0.15 = 0.25.$$

    Similarly, $P(Y = 2) = 0.2 + 0.05 = 0.25$. The most natural way to compute and display marginal distributions is to sum down columns and across rows in the table to compute $P(Y)$ and $P(X)$ respectively, and write the results in an added column and row to the table, as shown below. The added column is located in the "margin" of the table, giving these distributions their names.

|   |   | Y | | | | | $\mathbf{P(X)}$ |
|---|---|------|------|------|------|-----|------|
|   |   | 1 | 2 | 3 | 4 | 5 | |
| $X$ | A | 0.1 | 0.2 | 0.05 | 0.05 | 0.1 | **0.5** |
|   | B | 0.15 | 0.05 | 0 | 0.1 | 0.2 | **0.5** |
| | $\mathbf{P(Y)}$ | **0.25** | **0.25** | **0.05** | **0.15** | **0.3** | |

    NOTE: The marginal distributions $P(X)$ and $P(Y)$ are distributions for each of the variables $X$ and $Y$ irrespective of the value of the other variable. The marginal distribution $P(X)$ shows that value of $X$ has equal probability of being $A$ or $B$ when considered apart from $Y$.

2. What is the value of the conditional probability $P(X = A|Y = 2)$? Compute the distribution $P(X|Y = 2)$.

    ANSWER: From Equation 1.1.0.1, we compute

    $$P(X = A|Y = 2) = \frac{P(X = A, Y = 2)}{P(Y = 2)} = \frac{0.2}{0.25} = 0.8.$$

    Computing the distribution $P(X|Y = 2)$ means computing $P(X = x|Y = 2)$ of every $x \in dom X = \{A, B\}$. Since we already computed $P(X = A|Y = 2) = 0.8$, all that remains is to compute $P(X = B|Y = 2) = \frac{P(X=B,Y=2)}{P(Y=2)} = \frac{0.05}{0.25} = 0.2$. An alternative computation would of course be $P(X = B|Y = 2) = 1 - P(X = A|Y = 2) = 1 - 0.8 = 0.2$.

    NOTE: Computing a conditional probability distribution $p(X|Y = 2)$ should be though of as just considering the values in the table for the column $Y = 2$ and normalizing these so they sum to one. This concept of taking a subset of probabilities and normalizing them to sum to one is a central theme in Bayesian statistics.

3. Are the variables $X$ and $Y$ independent? Why or why not.

    ANSWER: The are not independent because $P(X = A|Y = 2) = 0.8$ while $P(X = A) = 0.5$.

## Example 3: Unites States Accents, Part 1

The population of Virginia is 8,536,000 the population of New York is 8,399,000, and the population of Massachusetts is 6,893,000. The primary accents spoken in these states are Inland Northern Great Lakes (INGL), Western New York (WNY), Boston (Bost), and Coastal Southern (CSo). The tables

below show the percent of people in each state that speak which each of these accents.

|       | VA | NY | MA |
|-------|----|----|----|
| INGL  | 28 | 42 | 7  |
| WNY   | 23 | 51 | 29 |
| Bost  | 2  | 5  | 63 |
| CSo   | 47 | 2  | 1  |

For a person randomly selected from the collective population of these states, let $S$ denote the state of origin and $A$ denote the accent.

1. What does the number 23 percent represent using the terminology (joint probability, marginal probability, conditional probability) and notation of this chapter?

   ANSWER: This is the conditional probability $P(A = WNY | S = VA)$.

2. What does the series of numbers in the table under NY represent?

   ANSWER: The conditional probability distribution $P(A | S = NY)$.

3. What is the probability distribution for $P(S)$, the probabilities of the state of origin for our randomly selected person?

   ANSWER: The probabilities are $P(S = VA) = 8536000/23828000 = 0.36$, $P(S = NY) = 0.35$ and $P(S = MA) = 0.29$. This is the marginal distribution for $S$ in the joint distribution $P(S, A)$.

UP TO HERE

For example, $P(X = H, Y = 2)$ is the probability that we get a heads on the coin and roll a 2 on the die. Since the coin toss and die roll are independent events (the outcome of one does not affect the outcome of the other, defined rigorously below), the probability of this pair of outcomes is $P(X = H, Y = 2) = P(X = H)P(Y = 2) = (1/2)(1/6) = 1/12$.

The situation is more interesting when there is a relationship between the variables. Consider a die where the faces for $\{1, 2, 3\}$ are colored red (R) and the faces for $\{4, 5, 6\}$ are colored green (G). Let $V$ denote the numerical value on a face and $C$ denote the associated color after a roll of this die. Then clearly knowing the color would change the probabilities of getting different values, and knowing the value would change the probabilities of the colors. We can represent the resulting joint probability distribution with Table 1.1.

|     |     | V |
|-----|-----|-----|-----|-----|-----|-----|-----|

|   |   | 1   | 2   | 3   | 4   | 5   | 6   |
|---|---|-----|-----|-----|-----|-----|-----|
| C | R | 1/6 | 1/6 | 1/6 | 0   | 0   | 0   |
|   | G | 0   | 0   | 0   | 1/6 | 1/6 | 1/6 |

Table 1.1: The joint probability for rolling a die with faces $\{1, 2, 3\}$ colored red and faces $\{4, 5, 6\}$ colored green.

Clearly for our example of colored-face die, $P(C = R | V = 5) = 0$ and $P(V = 4 | C = G) = 1/3$. The reader should confirm that these two statements are intuitively obvious, but also can be computed from Equation 1.1.0.1 with the joint probabilities in Table 1.1 as

$$P(C = R | V = 5) = \frac{P(C = R, V = 5)}{P(V = 5)} = \frac{0}{1/6} = 0, \quad P(V = 4 | C = G) = \frac{P(V = 4, C = G)}{P(C = G)} = \frac{1/6}{1/2} = 1/3.$$

It is often useful to fix the second variable in the conditional probability formula and treat the result as a new probability distribution on the first. For example, $P(V | C = G)$ is a probability function on $V$ whose PMF takes on the values $\{0, 0, 0, 1/3, 1/3, 1/3\}$ on the values $\{1, 2, 3, 4, 5, 6\}$ respectively.

If conditioning a random variable $X$ by a random variable $Y$ does not change the probabilities of outcomes of $X$, we say that $X$ and $Y$ are independent, as follows:

---

**Definition 7: Independent Random Variable**

For random variables $X$ and $Y$, if
$$P(X|Y) = P(X) \tag{1.1.0.8}$$
(explicitly, $P(X \in A | Y \in B) = P(X \in A)$ for all $A \subset \text{dom}(X)$ and $B \subset \text{dom}(Y)$) then we say that $X$ and $Y$ are **independent**.

---

It is straightforward to show that $p(X|Y) = p(X)$ if and only if $p(Y|X) = p(Y)$ as follows. Assume that $p(X|Y) = p(X)$ and from the definition of conditional probability:

$$p(X|Y) = P(X)$$
$$\frac{P(X,Y)}{P(Y)} = P(X)$$
$$\frac{P(X,Y)}{P(X)} = P(Y)$$
$$p(Y|X) = P(Y).$$

Similarly, $X$ and $Y$ are independent if and only if $p(X,Y) = p(X)p(Y)$, by the following computation.

$$p(X|Y) = P(X)$$
$$\frac{P(X,Y)}{P(Y)} = P(X)$$
$$P(X,Y) = P(X)P(Y).$$

We summarize this in a lemma.

**Lemma 1.1.1.** *The following are equivalent, and variables $X$ and $Y$ are independent if any of these are true.*
$$p(X|Y) = p(X),\ p(Y|X) = p(Y),\ \text{and } p(X,Y) = p(X)p(Y) \tag{1.1.0.9}$$

From the definition of conditional probability, we immediately get Bayes Theorem:

---

**Definition 8: Bayes Theorem**

Bayes Theorem, or Bayes' Rule, is

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{1.1.0.10}$$

---

INCLUDE PROOF!!! This theorem is extraordinary powerful because it allows us to invert probabilistic relationships between $P(A|B)$ and $P(B|A)$.

## 1.2 Probabilities, Likelihoods, and Priors in Bayes Theorem

Bayes Theorem has the mathematically simple formula given in Equation 1.1.0.10. But the power and effectiveness of this formula for probabilistic reasoning contrasts drastically to its simplicity. Bayes Theorem will play a central role in the machine learning algorithms presented in this book, and in this section we present the meaning of its components for probabilistic reasoning and computation.

Application of Bayes Theorem involved observing some values for data which we treat as outcomes from a probability distribution $D$. We assume that the probability distribution for $D$ has some parameter or parameters $\Theta$ (which may be a scalar or vector valued random variable). Then Bayes Theoem allows us to compute the probabilities for the values of $\Theta$ as follows.

---

**Definition 9: Bayes Thereom with Posterior, Prior, and Likelihood**

Let $D$ be a (scalar or vector-valued) random variable representing observed data and $\Theta$ be a random variable whose values represent parameters for a probability distribution for the random variable $D$. The the **posterior probability** (or just **posterior**) for $\Theta$ given $D$ is defined as

$$\underbrace{P(\Theta|D)}_{\text{Posterior Probability}} = \frac{\overbrace{P(D|\Theta)}^{\text{Likelihood}}\ \overbrace{P(\Theta)}^{\text{Prior}}}{\underbrace{P(D)}_{\text{Total Probability}}} \tag{1.2.0.1}$$

The **prior probability** (or just **prior**) is the term $P(\Theta)$ representing an estimate of this probability prior to using Bayes Theorem. The term **likelihood** is the probability (which may be from a PMF or PDF) of getting the data $D$ given the parameter value $\Theta$. The **total probability** is the probability $P(D)$ which can be computed by the formula $P(D) = \sum_{x \in \text{dom}(\Theta)} P(D|\Theta = x)P(\Theta = x)$.

---

The formula for the total probability $P(D)$ provided in Theorem 9 is derived by observing that

$$P(D) = \sum_{x \in \text{dom}(\Theta)} P(D, \Theta = x)$$

from Definition 6 of marginal probability. This formula for $P(D)$ is called the **sum rule**. Then using Definition 4 of conditional probability gives

$$P(D) = \sum_{x \in \text{dom}(\Theta)} P(D|\Theta = x)P(\Theta = x).$$

All notation here is for a discrete distribution on $\Theta$. The formula using a continuous distribution is $P(D) = \int_{\text{dom}(\Theta)} P(D|\Theta = s)P(\Theta = s)\mathrm{d}s$. Computing the value for $P(D)$ by using conditional probabilities on a second variable $\Theta$ is sometimes called **conditioning**.

In the following series of examples we show the flexibility by using Bayes Theorem to compute a posterior probability distribution over a series of increasingly realistic examples. This posterior probability distribution provides an accurate and useful measure of uncertainty that is often critical in machine learning.

---

**Example 4:** 🐍 **Single Flip of a Coin, Uninformative Prior**

Suppose you have two types of coins: coin Type $A$ is fair and coin Type $B$ will result in a Heads 75% of the time. You find a coin on the ground, and have no prior information on whether it is Type $A$ or $B$. (Perhaps you have an equal number of fair and unfair coins in your house.) You flip the coin once and get heads (H). What it the probability that this coin is Type $A$ or Type $B$?

ANSWER: We treat $D$ as a random variable with $\text{dom}(D) = \{H, T\}$, but we don't know the probability distribution for $D$. We have a single outcome $D = H$. There are two possible distributions, one for coin Type $A$ and one for coin Type $B$. This random variable is a Bernoulli random variable because it has two possible outcomes. It is standard to use $p$ to stand for the probability of one of these outcomes and $q$ for the other, so let $p = P(D = H)$ and $q = P(D = T)$, and of course $q = 1 - p$. We define $\Theta$ to be a random variable whose outcome is the value of $p$. The variable $\Theta$ in Bayes Theorem stands in for a parameter (or parameters) for the distribution of $D$, and once we have determined the parameter we

---

often use the parameter notation in place of $\Theta$ in the formulas. The problem is asking us to determine $P(p=0.5|D=H)$ (probability that the coin is Type $A$ given that we have one outcome of heads) and $P(p=0.75|D=H)$ (probability that the coin is Type $B$ given that we have one outcome of heads). We can now use Bayes Theorem to get the formula

$$P(p=0.5|D=H) = \frac{P(D=H|p=0.5)P(p=0.5)}{P(D=H)}. \tag{1.2.0.2}$$

It suffices to determine the likelihood, prior, and total probability on the right hand side. The likelihood is $P(D=H|p=0.5) = 0.5$. This is simple - if $p=0.5$ then the probability of getting a heads is 0.5. The prior probability $P(p=0.5)$ is the probability that the coin we found on the ground is Type $A$ apart from any consideration of the data. Estimating this prior is subjective and sometimes controversial. Since the problem suggests no information about whether the coin is Type $A$ or $B$, and in fact suggests the two types are equally likely, we assume $P(p=0.5) = 0.5$. (This type of prior, in which the possible outcomes of $\Theta$ have roughly equal probability, is called an **uninformative prior** because it carries little or no additional information.) Lastly, we compute the total probability as

$$P(D=H) = P(D=H|p=0.5)P(p=0.5) + P(D=H|p=0.75)P(p=0.75)$$
$$= 0.5 \times 0.5 + 0.75 \times 0.5 = 0.625.$$

Inserting these into Equation 1.2.0.2 yields

$$P(p=0.5|D=H) = \frac{P(D=H|p=0.5)P(p=0.5)}{P(D=H)} = \frac{0.5 \times 0.5}{0.625} = 0.4.$$

Similarly

$$P(p=0.75|D=H) = \frac{P(D=H|p=0.75)P(p=0.5)}{P(D=H)} = \frac{0.75 \times 0.5}{0.625} = 0.6.$$

**Example 5: Single Flip of a Coin, Prior from Data**

Consider the situation from Example 4, but instead of assuming that the coin is equal probability of being Type $A$ or $B$ before flipping it, assume that out of every 1000 coins you might find, all will be fair (Type $A$) and one will be Type $B$. What it the probability that this coin is Type $A$ or Type $B$?

ANSWER: The framework will be the same as before, with Bayes Theorem giving us

$$P(p=0.5|D=H) = \frac{P(D=H|p=0.5)P(p=0.5)}{P(D=H)}. \tag{1.2.0.3}$$

The difference is that the priors are $P(p=0.5) = 0.999$ and $P(p=0.75) = 0.001$ based on our belief that of every 1,000 randomly selected coins, 999 of them will be fair (Type $A$) and one will be Type $B$. Using these priors, we compute the total probability as

$$P(D=H) = P(D=H|p=0.5)P(p=0.5) + P(D=H|p=0.75)P(p=0.75)$$
$$= 0.5 \times 0.999 + 0.75 \times 0.001 = 0.50025.$$

Inserting these into Equation 1.2.0.2 yields

$$P(p=0.5|D=H) = \frac{P(D=H|p=0.5)P(p=0.5)}{P(D=H)} = \frac{0.5 \times 0.999}{0.50025} \approx 0.9985.$$

Similarly

$$P(p = 0.75|D = H) = \frac{P(D = H|p = 0.75)P(p = 0.5)}{P(D = H)} = \frac{0.75 \times 0.001}{0.50025} \approx 0.0015.$$

These two simple examples illustrate a number of principles that are common to all use of Bayes Theorem:

1. We usually want to compute the posterior probability $P(\Theta = x|D)$ for every possible $x \in \mathrm{dom}(\Theta)$. In Examples 4 and 1.2.0.3 we had two possible coin types, $A$ and $B$. Recall that $\Theta$ represents the parameter(s) for the distribution on the data $D$ which in this case is just the probability $p$ of getting heads in a single flip, so our coin types correspond to $\Theta = p = 0.5$ (Type $A$) and $\Theta = p = 0.75$ (Type $B$).

2. The total probability term $P(D)$ acts as a normalizing factor in the computation of posterior probabilities of all $P(\Theta = x|D)$, so that $\Sigma_{x \in \mathrm{dom}(\Theta)} P(\Theta = x|D) = 1$, which is required because $P(\Theta|D)$ is a probability distribution for $\Theta$. It is common to write Bayes Theorem as $P(\Theta|D) \propto P(D|\Theta)P(\Theta)$.

3. Bayes Theorem can be thought as reversing $P(D|\Theta)$ to get $P(\Theta|D)$.

Much of the Bayesian Machine Learning hinges on two major concepts that we will revisit repeatedly in this book. These are:

1. **Model Setup:** In every Bayesian Machine Learning application, there is a model setup step where the relationships between variables are determined along with distributions for random variables and parameter. This includes choosing a distribution for the data $D$, the parameters to include in $\Theta$, and the prior probabilities for these parameters.

2. **Sampling:** We ideally would like to compute the posterior probability for every possible parameter value $x \in \mathrm{dom}(\Theta)$. This is often computationally impossible, and there are different methods with different tradeoffs for sampling a subset of $x \in \mathrm{dom}(\Theta)$. Fortunately there are a growing toolbox of strategies using both computational and analytic methods for sampling.

When the model has additional parameters, we can express these parameters explicitly in Bayes Theorem as

$$P(\Theta|D, M) = \frac{P(D|\Theta, M)P(\Theta)}{P(D, M)} \tag{1.2.0.4}$$

where $M$ denotes the additional model parameters.

In out previous examples we computed probabilities of the type of coin using a single flip. We now present an example that is a little more realistic, using multiple flips of the coin. But, to model these multiple flips we need a different model which incorporates a different distribution for the data.

### Example 6: 🐍 Multiple Flips of a Coin

Consider the situation from Example 5, where you find a coin and believe that it is either fair (Type $A$ with $p = 0.5$) or unfair (Type $B$ with $p = 0.75$), and that you believe that out of every 1,000 coins one is Type $B$ and the rest are Type $A$. But instead of flipping the coin just once, you flip it 100 times and get 68 heads and 32 tails. What is the probability that this coin is Type $A$ and the probability that the coin is Type $B$?

ANSWER: As before, we want to use Bayes Theorem to compute the probability that the coins is Type $A$. However, the data is now the number of heads in 100 flips of the coin, which we model using a binomial distribution.

The binomial distribution involves $N$ independent Bernouli trials, each of which has a probability $p$ of success (Heads) and $q = 1 - p$ of failure (Tails). The probability that exactly $k$ of the trials produces success (given that the number of Bernoulli trials is equal to $N$ and the probability of success in each

is equal to $p$) is

$$P(k|N, p) = \binom{N}{k} \cdot p^k q^{N-k}.$$

Now Bayes Theorem provides the formula for the probability that $p = 0.5$ as

$$P(p = 0.5|k = 68, N = 100) = \frac{P(k = 68|p = 0.5, N = 100)P(p = 0.5)}{P(k = 68|N = 100)}. \tag{1.2.0.5}$$

Using the binomial distribution, $P(k = 68|p = 0.5, N = 100) = \binom{100}{68} \cdot 0.5^{68} 0.5^{32} \cong 0.000113$. The prior probability for Type $A$ is $P(p = 0.5) = 0.999$. The total probability is

$$\begin{aligned} P(k = 68|N = 100) =& P(k = 68|N = 100, p = 0.5)P(0.5) + P(k = 68|N = 100, p = 0.75)P(0.75) \\ =& (0.000113)(0.999) + (0.02475256392905903)(0.001) \cong 0.000137. \end{aligned}$$

Putting these together gives

$$P(p = 0.5|k = 68, N = 100) \cong \frac{(0.000113)(0.999)}{0.000137} \cong 0.8199. \tag{1.2.0.6}$$

A similar computing on Type $B$ with $p = 0.75$ and $P(p = 0.75) = 0.001$ yields

$$P(p = 0.75|k = 68, N = 100) \cong 0.1801. \tag{1.2.0.7}$$

So far, we have been assuming that only two types of coins are possible - Type $A$ with $p = 0.5$ and Type $B$ with $p = 0.75$ - and we computed the posterior probability for each type individually. This would be appropriate if circumstances dictated that these are the only types of coins that would reasonably be found. It would be reasonable to expect that a coin (that is randomly selected from circulation without bias) would have a value for $p = P(\text{Heads})$ anywhere in $[0, 1]$ but would most likely be very close to 0.5. To model this, we assume that the prior probability distribution for $p$ is a normal distribution with mean of 0.5 and standard deviation of 0.01, $p \sim \mathcal{N}(0.5, 0.1^2)$. This probability distribution is shown to the right.



Prior Probabilty for $p = P(\text{Heads})$
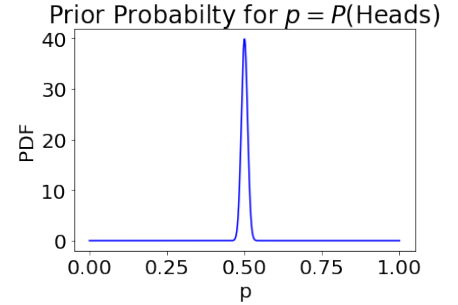
### Example 7: 🐍 Multiple Flips of a Coin

Consider the situation where you find a coin as in the previous series of examples. Before flipping the coin, you expect that coin is fair (i.e. $p \approx 0.5$). To quantify this belief, you decide that a good prior probability distribution for the value of $p = P(\text{Heads})$ is normal with mean 0.5 and $\sigma = 0.01$, $p \sim \mathcal{N}(0.5, 0.01^2)$. You then flip the coin 100 times and get 68 heads and 32 tails. (a) Use your prior and the evidence (or data) from flipping the coin to compute the posterior probability distribution for $p$. (b) Plot the prior, likelihood, and posterior as a function of $p$ in $[0, 1]$. (c) Provide an interval such that probability that the actual value of $p$ is in this interval is 95%. [This is called the 95% **credible interval** for the parameter $p$. This is sometimes called a confidence interval, but the term credible interval is preferred to indicate that this interval is quantifying uncertainty about a parameter from a posterior distribution.]

ANSWER: (a) In Examples 4 through 6 we computed the posterior probability for two values $P(p = 0.5|D)$ and $P(p = 0.75|D)$. Now we want to compute the posterior probability $P(p = x|D)$ for every

$x \in [0, 1]$. The goal is to use Bayes Theorem to compute

$$P(p = x | k = 68, N = 100) = \frac{P(k = 68 | p = x, N = 100) P(p = x)}{P(k = 68 | N = 100)}, \tag{1.2.0.8}$$

for every $x \in [0, 1]$. The likelihood function is given by

$$P(k = 68 | p = x, N = 100) = \binom{100}{68} \cdot x^{68} (1 - x)^{32} = \frac{100!}{68! \times 32!} \cdot x^{68} (1 - x)^{32}.$$
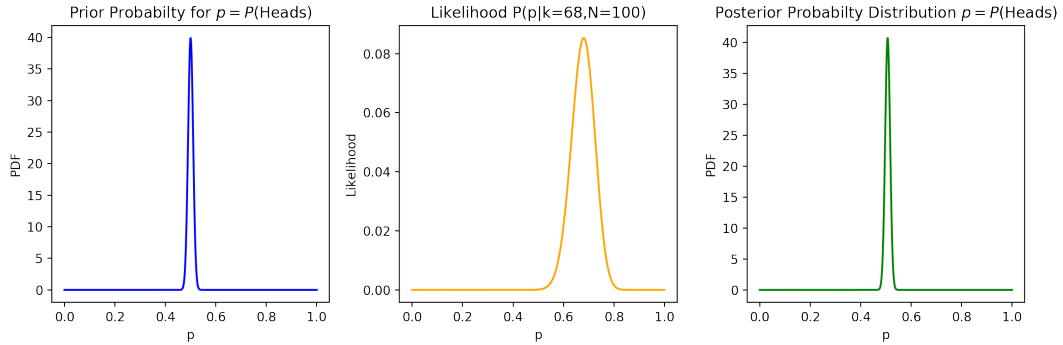
The prior is

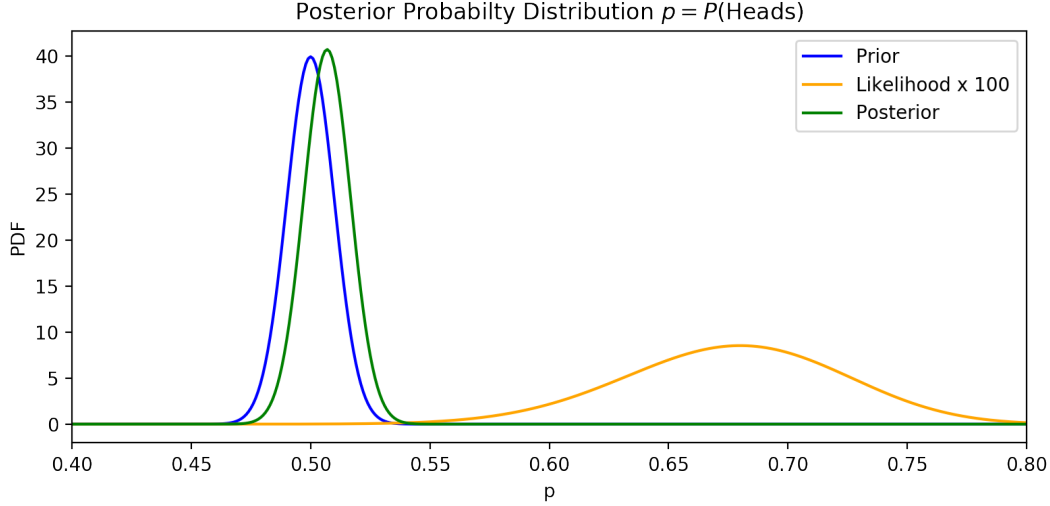$$P(p = x) = \frac{1}{0.01\sqrt{2\pi}} e^{-(x - 0.5)^2 / 20.01^2}.$$

Combining these gives

$$P(p = x | k = 68, N = 100) = \frac{\frac{100!}{68! \times 32!} \cdot x^{68} (1 - x)^{32} \times \frac{1}{0.01\sqrt{2\pi}} e^{-(x - 0.5)^2 / 20.01^2}}{\int_0^1 \frac{100!}{68! \times 32!} \cdot s^{68} (1 - s)^{32} \times \frac{1}{0.01\sqrt{2\pi}} e^{-(s - 0.5)^2 / 20.01^2} ds}$$

(b) The likelihood, prior, and posterior are shown in Figure **??**. In computing these numerically, it is necessary to choose a choose a subset of points $\{x_i\}$ in $[0, 1]$, then compute the likelihood and prior for these $\{x_i\}$ as $l_i = P(k = 68 | p = x_i, N = 100)$ and $pr_i = P(p = x_i)$, and then estimate the total probability integral by the sum

$$\int_0^1 \frac{100!}{68! \times 32!} \cdot s^{68} (1 - s)^{32} \times \frac{1}{0.01\sqrt{2\pi}} e^{-(s - 0.5)^2 / 20.01^2} ds \cong \Sigma_i l_i pr_i \delta x_i.$$

Posterior Probabilty Distribution $p = P(\text{Heads})$

(c) The 95% credible interval is $[0.488, 0.527]$. This is computed from the pdf for the posterior distribution by computing the cumulative density function (cdf) from the pdf, and then finding the values or the cdf that correspond to 2.5% and 97.5%.

Example 7 provides a useful illustration of the standard framework for Bayesian Machine Learning. Before analysis we have a prior probability distribution.. New data is collected (in this case coming from flips of a coin) provides a likelihood function. The prior and likelihood are combined via Bayes Theorem, and observe in the plots of Example 7 the prior has a mean centered on 0.5 but the likelihood from the new data modifies the prior to obtain the posterior distribution which has a mean slightly above 0.5, and with a 95% credible interval of $[0.488, 0.527]$. One could conclude from this that we could not reject the hypothesis that $p = 0.5$, or compute other information such as the probability that $p > 0.5$.

## 1.3 Probabilistic Computations

In this section we use Bayes Theorem to compute probabilities and relationships between variables. In the first example, we are given a table of percentages of people with different accents in each state, which are conditional probabilities of the form

$$P(Accent|State).$$

Bayes Theorem allows us to switch the order of the variables in the conditional probability to get the form

$$P(State|Accent).$$

Observe that the table given percentage of people with different accents would be easy to determine from a survey, and the number of people in each state would also be easy to compile. Bayes Theorem enables us to compute the probability of state of origin from a person's accent. This framework is useful in many situations, for example determine the probability of authorship of a document based on word usage, or a general class label based on variable values.

### Example 8: Unites States Accents, Part 2

The following data was first presented in Example 3. The population of Virginia is 8,536,000 the population of New York is 8,399,000, and the population of Massachusetts is 6,893,000. The primary accents spoken in these states are Inland Northern Great Lakes (INGL), Western New York (WNY), Boston (Bost), and Coastal Southern (CSo). The tables below show the percent of people in each state that speak which each of these accents.

|       | VA | NY | MA |
|-------|----|----|----|
| INGL  | 28 | 42 | 7  |
| WNY   | 23 | 51 | 29 |
| Bost  | 2  | 5  | 63 |
| CSo   | 47 | 2  | 1  |

If a person Lila has a Wester New York (WNY) accent, find out the probability that Lila is from each of the given states.

ANSWER: We compute the probability that Lila is from VA using Bayes Theorem as

$$P(State = VA | Accent = WNY) = \frac{P(Accent = WNY | State = VA)P(State = VA)}{P(Accent = WNY)}.$$

The prior $P(State = VA)$ is the proportion of people from all the states considered that are from VA,

$$P(State = VA) = \frac{8,536,000}{8,536,000 + 8,399,000 + 6,893,000} \approx 0.358234.$$

The likelihood $P(Accent = WNY | State = VA)$ is the proportion of people from VA who have an WNY accent, $P(Accent = WNY | State = VA) = 0.23$. Therefore the numerator of the Bayes Theorem fraction is

$$P(Accent = WNY | State = VA)P(State = VA) = (0.358234)(0.23) \approx 0.0823938$$

The denominator is the total probability, computed by conditioning over the states, similar to the numerator for each state and summing.

$$\begin{aligned}
P(Accent = WNY) &= P(Accent = WNY | State = VA)P(State = VA) \\
&+ P(Accent = WNY | State = NY)P(State = NY) \\
&+ P(Accent = WNY | State = MA)P(State = MA) \\
&\approx (0.358234)(0.23) + (0.352484)(0.51) + (0.2892815)(0.29) \\
&\approx 0.346052.
\end{aligned}$$

Putting these back in Bayes Theorem yields

$$\begin{aligned}
P(State = VA | Accent = WNY) &= \frac{P(Accent = WNY | State = VA)P(State = VA)}{P(Accent = WNY)} \\
&= \frac{0.0823938}{0.346052} \approx 0.02851.
\end{aligned}$$

Our second example involves the classic television game show Monte Hall. Again, we are provided probabilistic information about relationships between variables in the form of conditional probabilities and use Bayes Theorem to compute additional probabilities.

### Example 9: Monte Hall

In the Monte Hall television show, a contestant is shown three large doors. Behind one of the doors is a valid prize like a new car, and behind the other two doors are dud objects, for example goats. The contest is asked to choose a door. After the choice is made, the host Monte Hall opens one of the remaining doors to reveal a humorous dud prize. The contestant is then asked if he or she wants to change the door selection to the other unopened door. Our goal in this example is to determine the optimal strategy: what is the probability of winning if the contestant does not change selection, and

what is the probability of winning if the selection is changed.

ANSWER: The first step is to reformulate the situation in terms of variables and probabilities. Label the doors 1, 2, and 3. Without loss of generality, assume that the contestant has chosen door 1. (The phrase "without loss of generality" means that we are making an arbitrary choice that will simplify the proof/discussion/computation but does not actually change the level of generality.) Let $C$ denote the door with the car. Let $H$ denote the number for the door that the host reveals after the selection is made.

Initially, we have $P(C = 1) = P(C = 2) = P(C = 3) = \frac{1}{3}$ since all doors are equally likely to contain the car. Since the contestant chose door 1, there are two possibilities for the host, either $H = 2$ or $H = 3$. We compute the probability for the location of the door in each case seperately.

Case 1: $H$=2: By Bayes Theorem, the probability that the car is behind door 1 is

$$P(C = 1|H = 2) = \frac{P(H = 2|C = 1)P(C = 1)}{P(H = 2)}. \tag{1.3.0.1}$$

Our prior is $P(C = 1) = \frac{1}{3}$. The likelihood $P(H = 2|C = 1)$ represents the probability that the host opens door 2 if the prize is behind door 1 (and the contestant chose door 1). Since there are 2 non-car doors to select, the host is equally likely to select either of these and $P(H = 2|C = 1) = \frac{1}{2}$. The numerator is then

$$P(H = 2|C = 1)P(C = 1) = (\frac{1}{2})(\frac{1}{3}) = \frac{1}{6}$$

The total probability in the denominator is computed as

$$P(H = 2) = P(H = 2|C = 1)P(C = 1) + P(H = 2|C = 2)P(C = 2) + P(H = 2|C = 3)P(C = 3)$$

We already showed that $P(H = 2|C = 1)P(C = 1) = \frac{1}{6}$. The computation for $P(H = 2|C = 2)P(C = 2)$ is simple; if the car is behind door 2, then the probability that the host opens door 2 is zero, because he always opens a door revealing a non-prize door. Thus $P(H = 2|C = 2)P(C = 2) = 0$. The probability $P(H = 2|C = 3)$ represent the probability that the host opens door 2 if the car is behind door 3 (and the contestant chose door 1). This probability equals one, since the host in this case is required to open door 2, the only remaining door. Thus, $P(H = 2|C = 3)P(C = 2) = (1)(\frac{1}{3}) = \frac{1}{3}$. Putting these together we get

$$P(H = 2) = P(H = 2|C = 1)P(C = 1) + P(H = 2|C = 2)P(C = 2) + P(H = 2|C = 3)P(C = 3)$$
$$= \frac{1}{6} + 0 + \frac{1}{3} = \frac{1}{2}.$$

Putting these into equation 1.3.0.1 yields

$$P(C = 1|H = 2) = \frac{P(H = 2|C = 1)P(C = 1)}{P(H = 2)} = \frac{1/6}{1/2} = \frac{1}{3}. \tag{1.3.0.2}$$

This is the probability that the contestant will win if he or she does not change the door selection; the probability that the car is behind door 1, given that the host has opened door two.

Still assuming $H = 2$ (the host revealed door 2 to show a non-prize), trivially $P(C = 2|H = 2) = 0$, so it remains to compute $P(C = 3|H = 2) = 0$. Bayes Theorem gives

$$P(C = 3|H = 2) = \frac{P(H = 2|C = 3)P(C = 3)}{P(H = 2)}. \tag{1.3.0.3}$$

As before, $P(H = 2|C = 3)P(C = 2) = (1)(\frac{1}{3}) = \frac{1}{3}$ and $P(H = 2) = \frac{1}{2}$. Thus,

$$P(C = 3|H = 2) = \frac{P(H = 2|C = 3)P(C = 3)}{P(H = 2)} = \frac{1/3}{1/2} = \frac{2}{3}. \tag{1.3.0.4}$$

This is the probability that the contestant wins if he or she switches doors.

The computation for the alternate case, assuming $H = 3$ is exactly the same.

An alternative approach would be to create a simulation of the Monte Hall game with randomly selected doors, run the simulation many times, and compute the probability of the different outcomes given each strategy. The following is a Python implementation of a simulation. It is left as an exercise to the reader to modify this code to counts wins and losses for each strategy to see that this probabilities match those computed using Bayes Theorem.

```
[4]: # This is a simulation of repeating the Monte Hall problem with randomly selected␣
     ↪strategies.

     import numpy as np
     doors = [0,1,2]

     for i in range(5):
         # choose a strategy at random
         strategy = np.random.choice(['switch','stay'])
         # car location is chosen randomly
         Car_location = np.random.choice([0,1,2])
         # contestant makes a random choice
         Contestant_Choice = np.random.choice([0,1,2])
         # Host reveals a door:
         remaining_doors = [door for door in doors if door not in (Car_location,␣
     ↪Contestant_Choice)]
         Host_door = np.random.choice(remaining_doors)

         # switch doors if this is the strategy
         if strategy == 'switch':
             Contestant_Choice = [door for door in doors if door not in (Contestant_Choice,␣
     ↪Host_door)]

         # determine if we won or not
         if Contestant_Choice == Car_location:
             print('Strategy: '+strategy+' and you win!')
         else:
             print('Strategy: '+strategy+' and you get a goat.')
```

```
Strategy: stay and you win!
Strategy: stay and you get a goat.
Strategy: switch and you get a goat.
Strategy: switch and you win!
Strategy: stay and you get a goat.
```

## 1.4   Single Variable Probability Distributions

In this section we provide a review of probability distributions that will be useful as a reference in subsequent chapters. For each distribution, we include the formula for the **mean**, which is the expected value of the random variables defined as

$$E[X] = \sum_{x \in \mathrm{dom}(X)} x p(x) \tag{1.4.0.1}$$

for a discrete random variable and

$$E[X] = \int_{\mathrm{dom}(X)} x f(x) \mathrm{dx} \tag{1.4.0.2}$$

for a continuous random variable, where $f(x)$ is associated PDF. More generally for any real-valued function $g$, we define the expected value of $g$ to be

$$E[g(X)] = \sum_{x \in \mathrm{dom}(X)} g(x) p(x) \tag{1.4.0.3}$$
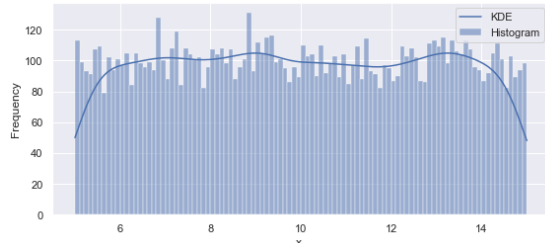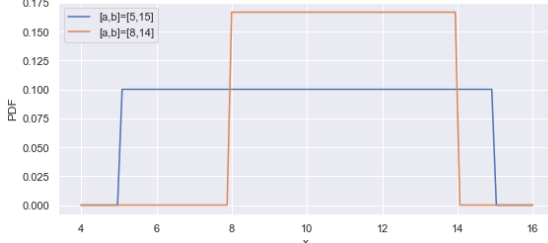
and similarly for the continuous case. This allows us to define the **variance** as
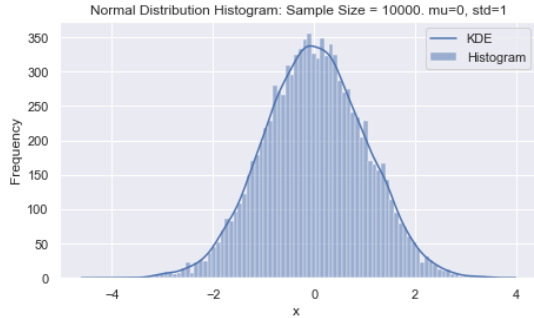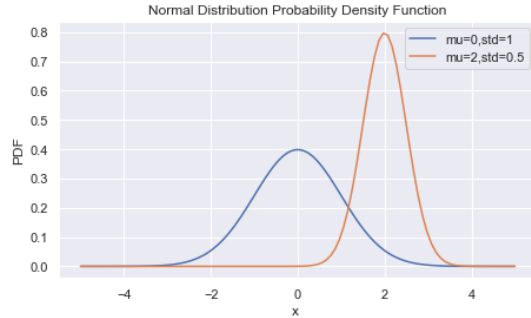
$$\mathrm{VAR}[X] = E[(X - E[X])^2] = \sum_{x \in \mathrm{dom}(X)} (x - E[x])^2 p(x) \tag{1.4.0.4}$$
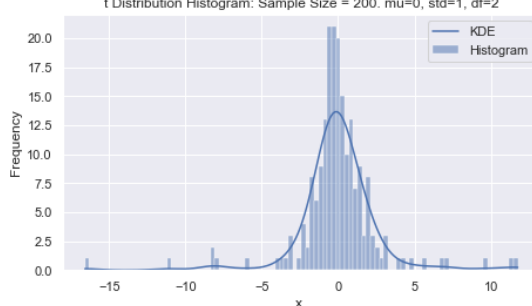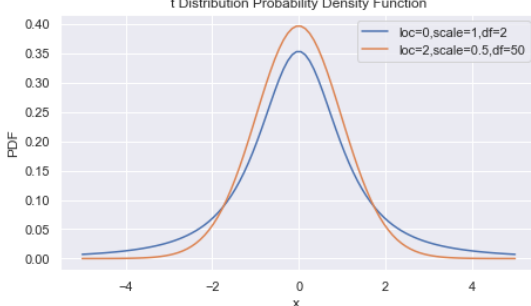
and again similarly for the continuous case.

These definitions enable computations, for example showing that the expected value of the sum is the sum of the expected values, as follows. (Note that the definition of the marginal distribution is used in the middle of these equations.)
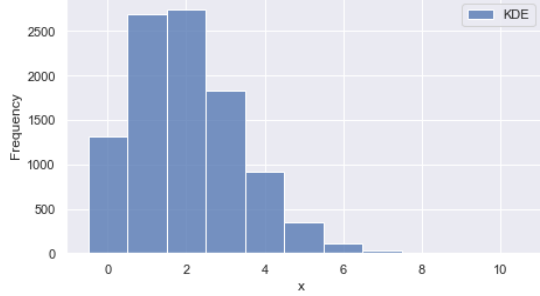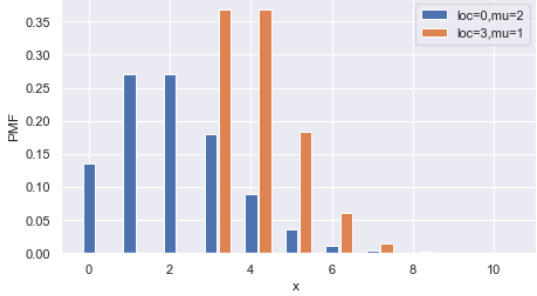
$$
\begin{aligned}
E[X + Y] &= \sum_x \sum_y (x + y) P_{XY}(x, y) \\
&= \sum_x \sum_y x P_{XY}(x, y) + \sum_y \sum_y y P_{XY}(x, y) \\
&= \sum_x x P_X(x) + \sum_y y P_Y(y) \\
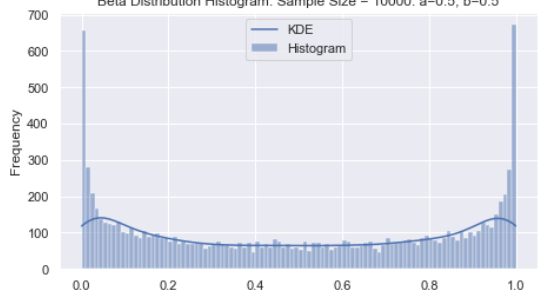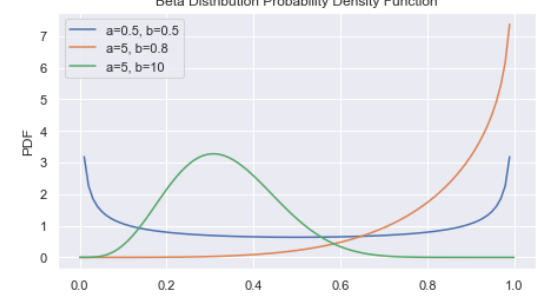&= E[X] + E[Y].
\end{aligned}
$$

| Uniform Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $x \sim \mathcal{U}(a, b)$ | Parameters: | $-\infty < a < b < \infty$ | Domain: | $x \in [a, b]$ |
| PDF: | $f(x\mid a, b) = \begin{cases} \frac{1}{b-a} & \text{if } x \in [a,b] \\ 0 & \text{otherwise} \end{cases}$ | | CDF: | | $\begin{cases} 0 & \text{if } x < a \\ \frac{x-a}{b-a} & \text{if } x \in [a,b] \\ 1 & \text{if } x > b \end{cases}$ |
| Mean: | $(b-a)/2$ | Median: | $(b-a)/2$ | Variance: | $\frac{1}{12}(b-a)^2$ |

Uses: A uniform random variable is used to represent a process where there all values in an interval are equally probable, for example spinning a spinner which will come to rest at any angle in $[0, 2\pi]$. This is also useful when an outcome is constrained to an interval but no additional information is available.

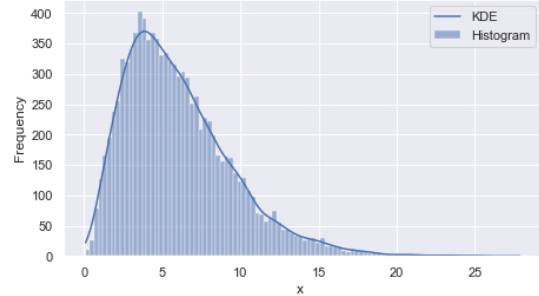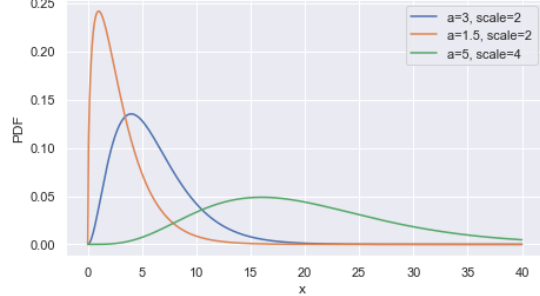| Normal Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $x \sim \mathcal{N}(\mu, \sigma^2)$ | Parameters: | $\mu \in \mathbb{R}$ (mean) $\sigma > 0$ (standard deviation) | Domain: | $x \in \mathbb{R}$ |
| PDF: | $f(x\|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$ | | CDF: | | $\frac{1}{2}\left[1 + \mathrm{erf}(\frac{\mu+x}{\sigma\sqrt{2}})\right]$ |
| Mean: | $\mu$ | Median: | $\mu$ | Variance: | $\sigma^2$ |

Uses: The normal distribution is the most common probability distribution, used for biological and physical measurements and quantifying noise or measurement error. the Central Limit Theorem provides support for this common use, stating that under common circumstances the sum of arbitrary random variables is well-approximated by a normal distribution.



| t-Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $x \sim T_\nu$ | Parameters: | $\nu > 0$ (d.o.f.) $\mu \in \mathbb{R}$ (location) $\sigma > 0$ (scale) | Domain: | $x \in \mathbb{R}$ |
| PDF: | $f(x\|\nu, \mu, \sigma) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi\hat{\sigma}^2}\Gamma\left(\frac{\nu}{2}\right)}\left(1 + \frac{1}{\nu}\frac{(x-\mu)^2}{\hat{\sigma}^2}\right)^{-\frac{\nu+1}{2}}$ | | CDF: | | $\frac{1}{2} + x\Gamma\left(\frac{\nu+1}{2}\right) \times \frac{{}_2F_1\left(\frac{1}{2}, \frac{\nu+1}{2}; \frac{3}{2}; -\frac{x^2}{\nu}\right)}{\sqrt{\pi\nu}\Gamma\left(\frac{\nu}{2}\right)}$ |
| Mean: | $\mu$ if $\nu > 0$ else undef. | Median: | $0$ | Variance: | $\begin{cases} \frac{\nu\hat{\sigma}}{\nu-2} & \text{if } \nu > 2 \\ \infty & \text{if } 1 < \nu \leq 2 \\ \text{undef.} & \text{otherwise} \end{cases}$ |

Uses: The t-distribution is used for $n$ samples of data whose full population is assumed to have a normal distribution, in which case $\nu = n - 1$. In comparison to a normal distribution, the t-Distribution has fatter tails and the difference becomes insignificant for large values of $n$. NOTE: d.o.f stands for degrees of freedom.
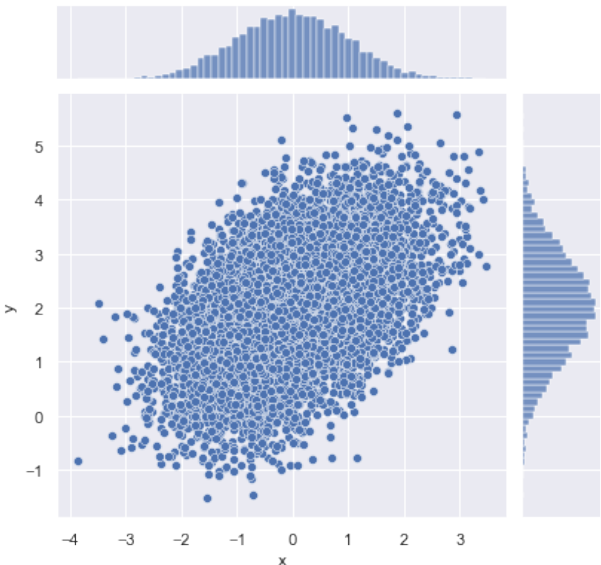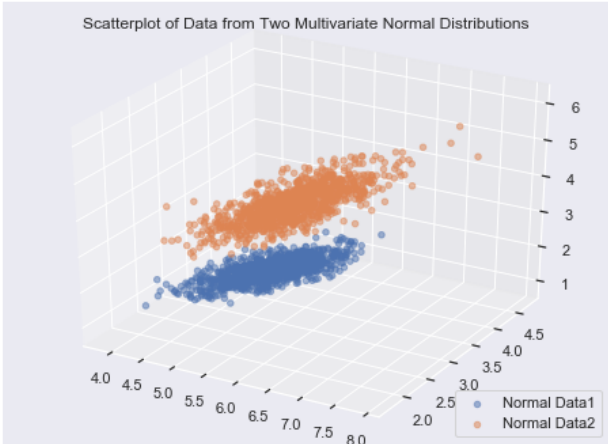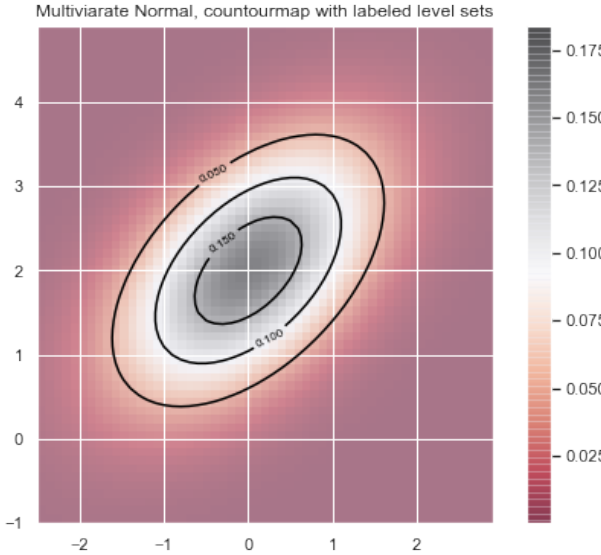
| Bernoulli Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: $k \sim \text{Bern}(p)$ | | Parameters: | $p \in [0,1]$ (probability of success) $q = 1 - p$ | Domain: | $k \in \{0,1\}$ |
| PMF: $P(k\|p) = p^k(1-p)^{1-k}$ | | CDF: | | $\begin{cases} 0 & \text{if } k < 0 \\ 1-p & \text{if } 0 \le k < 1 \\ 1 & \text{if } k \ge 1 \end{cases}$ | |
| Mean: $p$ | | Median: | $\begin{cases} 0 & \text{if } p < 1/2 \\ [0,1] & \text{if } p = 1/2 \\ 1 & \text{if } p > 1/2 \end{cases}$ | Variance: | $p(1-p) = pq$ |

Uses: The Bernoulli distribution is a simple distribution used to model any random variable, or experiment, with two possible outcomes. The quintessential example is the flip of a coin.



| Binomial Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: $k \sim B(n,p)$ | | Parameters: | $p \in [0,1]$ (probability of success) $n \in \{1,2,3,...\}$ (# trials) $q = 1 - p$ | Domain: | $k \in \{0,1,...,n\}$ |
| PMF: $P(k\|n,p) = \binom{n}{k}p^k(1-p)^{n-k}$ | | CDF: | | $\sum_{i=0}^{\lfloor k \rfloor} \binom{n}{i}p^i(1-p)^{n-i}$ | |
| Mean: $np$ | | Median: | (no simple formula) | Variance: | $npq$ |

Uses: The Binomial distribution is used to model the number of success from multiple independent Bernoulli trials, for example the number of heads in multiple flips of a coin or the number of mutation errors in coding a DNA segment of length $n$.

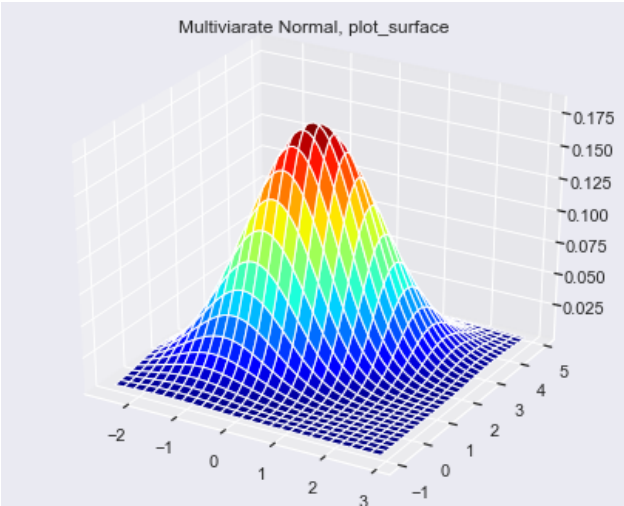| Poisson Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $k \sim \text{Pois}(\lambda)$ | Parameters: | $\lambda \in (0, \infty)$ (rate) | Domain: | $k \in \mathbb{N}_0$ |
| PMF: | $P(k\|\lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$ | | CDF: | | $e^{-\lambda} \sum_{i=0}^{\lfloor k \rfloor} \frac{\lambda^i}{i!}$ |
| Mean: | $\lambda$ | Median: | (no simple formula) | Variance: | $\lambda$ |

Uses: The Poisson distribution is used to model the number of occurrences of some processes over and interval of time, if the events occur with a fixed mean rate and are independent. Examples include the number of accidents on a stretch of road per month, or number of manufacturing defects per week.
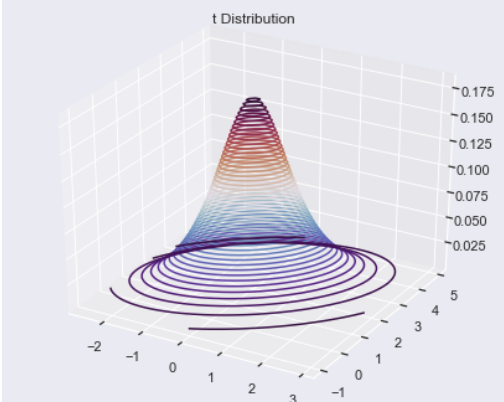


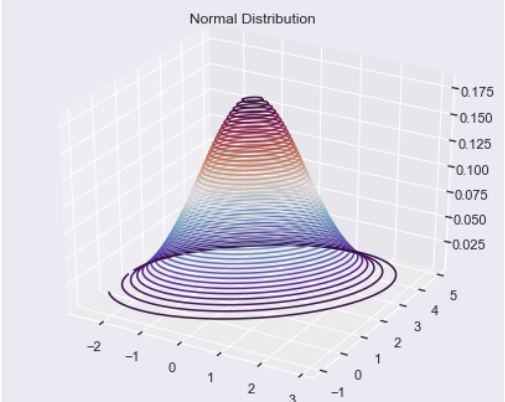| Beta Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $x \sim \text{Beta}(\alpha, \beta)$ | Parameters: | $\alpha \in (0, \infty)$ (shape) $\beta \in (0, \infty)$ (shape) | Domain: | $x \in [0, 1]$ (possibly excluding endpoints) |
| PDF: | $f(x\|\alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)}$, where $B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ | | CDF: | | (no simple formula) |
| Mean: | $\frac{\alpha}{\alpha+\beta}$ | Median: | (no simple formula) | Variance: | $\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$ |

Uses: The Beta distribution is used to model a random variable on an interval of finite length in many applications, and is used in Bayesian methods as a prior probability distribution. The shape parameter $\alpha$ determines if the limiting value of the distribution at 0 is 0, 1, or $\infty$ depending on whether $\alpha > 1$, $\alpha = 1$, or $\alpha < 1$ respectively, and a similar relationship holds for $\beta$ and the limiting value at 1. As a prior, $B(\alpha, \beta)$ represents a belief that $\alpha + \beta - 2$ trials would result in $\alpha - 1$ heads and $\beta - 1$ tails. Also, $B(1, 1)$ is the uniform distribution on $[0, 1]$.
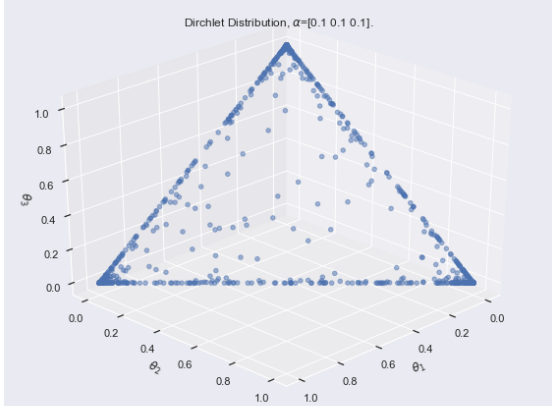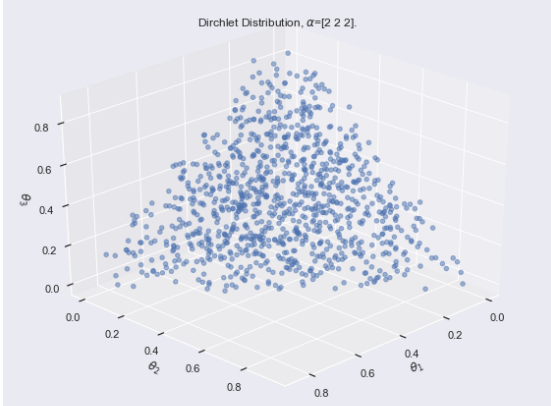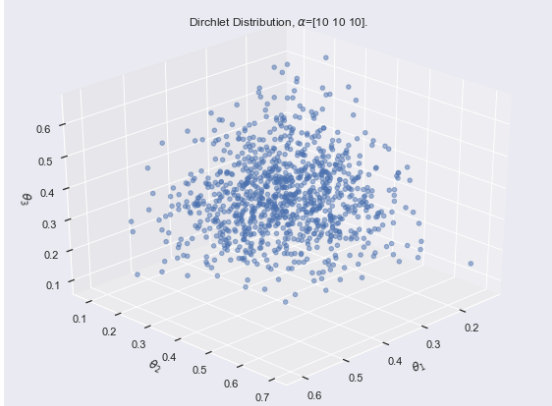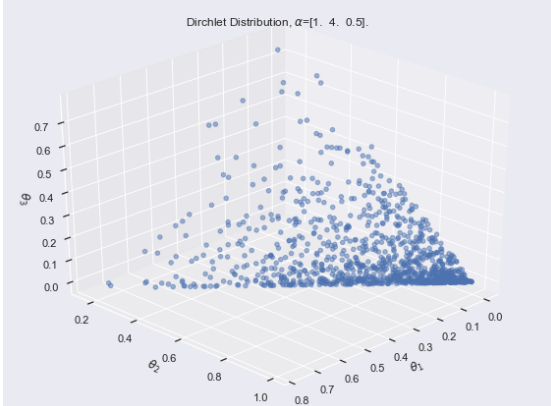
| Gamma Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $x \sim \Gamma(k, \theta)$ | Parameters: | $k \in (0, \infty)$ (shape) $\theta \in (0, \infty)$ (shape) | Domain: | $x \in (0, \infty)$ |
| PDF: | $f(x|k, \theta) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-x/\theta}$ | CDF: | | | $\frac{1}{\Gamma(k)} \gamma(k, \frac{x}{\theta})$ |
| Mean: | $k\theta$ | Median: | (no simple formula) | Variance: | $k\theta^2$ |

Uses: The Gamma distribution is used to model accumulation over a unit of time, for example rainfall accumulated in a reservoir, total insurance claims, or total impact of mutations on fitness. The Gamma distribution is also used to model the time until the $k^{\text{th}}$ occurrence of an event which occurs at mean rate of $1/\theta$ (equivalently, the sum of $k$ random variables with Poisson distributions and rate $\gamma = 1/\theta$). An alternative parametrization uses the shape parameter $\alpha = k$ and rate parameter $\beta = 1/\theta$. In Bayesian methods, the Gamma distribution is a useful prior distribution.

## 1.5 Multi-variate Probability Distributions

| Multivariate Normal Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $x \sim \mathcal{N}(\mu, \Sigma)$ | Parameters: | $\mu \in \mathbb{R}^k$ (mean) $\Sigma$ (covariance matrix) | Domain: | $x \in \mathbb{R}^k$ |
| PDF: | $f(x\|\mu, \Sigma) = \frac{1}{(2\pi)^{-k/2}\|\Sigma\|^{-1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$ | | | Mean: | $\mu$ |

Uses: The multivariate normal distribution (aka multivariate Gaussian distribution) is the most common generally used multivariate distribution, partly supported by the multivariate central limit theorem. It is easy to fit to data by computing the mean and covariance of the data, and has general practical utility if the number of observations in the data is sufficient for the number of dimensions and correlations (higher dimensions and correlation between component variables require more observations for a robust estimation of the covariance).

| Multivariate t-Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $x \sim \mathcal{N}(\mu, \Sigma)$ | Parameters: | $\mu \in \mathbb{R}^k$ (mean) $\Sigma$ (covariance matrix) | Domain: | $x \in \mathbb{R}^k$ |
| PDF: | | $f(x|\nu, \mu, \Sigma) = \frac{\Gamma(\nu+k)}{\Gamma(\nu/2)(\nu\pi)^{k/2}|\Sigma|^{1/2}} \left[1 + \frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right]^{-(\nu+k)/2}$ | | Mean: | $\mu$ |

Uses: The multivariate normal distribution (aka multivariate Gaussian distribution) is the most common generally used multivariate distribution, partly supported by the multivariate central limit theorem. It is easy to fit to data by computing the mean and covariance of the data, and has general practical utility if the number of observations in the data is sufficient for the number of dimensions and correlations (higher dimensions and correlation between component variables require more observations for a robust estimation of the covariance).

| Dirchlet Distribution: | | | | | |
|---|---|---|---|---|---|
| Notation: | $x \sim \mathcal{N}(\mu, \Sigma)$ | Parameters: | $\mu \in \mathbb{R}$ (mean) $\Sigma$ (covariance matrix) | Domain: | $x \in [a, b]$ |
| PDF: | $\begin{cases} \frac{1}{b-a} & \text{if } x \in [a,b] \\ 0 & \text{otherwise} \end{cases}$ | CDF: | $\begin{cases} 0 & \text{if } x < a \\ \frac{x-a}{b-a} & \text{if } x \in [a,b] \\ 1 & \text{if } x > b \end{cases}$ | | |
| Mean: | $(b-a)/2$ | Median: | $(b-a)/2$ | Variance: | $\frac{1}{12}(b-a)^2$ |

Uses: The multivariate normal distribution (aka multivariate Gaussian distribution) is the most common generally used multivariate distribution, partly supported by the multivariate central limit theorem. It is easy to fit to data by computing the mean and covariance of the data, and has general practical utility if the number of observations in the data is sufficient for the number of dimensions and correlations (higher dimensions and correlation between component variables require more observations for a robust estimation of the covariance).

# Appendix A

# Jupyter Notebooks for Examples and Figures

**Example 2:**

1. Compute the marginal distributions P(X) and P(Y).

```
[1]: import numpy as np
     import matplotlib.pyplot as plt
```

```
[2]: # Build the matrix M which will hold our joint probability distribution, and print M
     M = np.asarray([[0.1,0.2,0.05,0.05,0.1],[0.15,0.05,0,0.1,0.2]])
     print(M)
```

```
[[0.1  0.2  0.05 0.05 0.1 ]
 [0.15 0.05 0.   0.1  0.2 ]]
```

```
[3]: # Compute the marginal probabilities
     P_Y = np.sum(M,axis=0)
     P_X = np.sum(M,axis=1)
     print('The distribution P(X) is '+str(P_X)+'.')
     print('The distribution P(Y) is '+str(P_Y)+'.')
```

```
The distribution P(X) is [0.5 0.5].
The distribution P(Y) is [0.25 0.25 0.05 0.15 0.3 ].
```

```
[4]: # Create a bar plot to show the distribution P(Y).
     # We use a bar plot here instead of a line plot because the distribution is discrete.

     # Plot of the marginal distribution P(Y)
     plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to the 1st␣
      ↪location.
     dom_Y = [1,2,3,4,5]
     plt.bar(dom_Y,P_Y)
     plt.title('Marginal Distribution P(Y)');
     plt.xlabel('Y');
     plt.ylabel('Probablity');

     # Plot of the marginal distribution P(X)
```

```
plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to the 2nd␣
 ↪location.
dom_X = ['A','B']
plt.bar(dom_X,P_X)
plt.title('Marginal Distribution P(X)');
plt.xlabel('X');
plt.ylabel('Probablity');
plt.tight_layout() # this cleans up the spacing for plots and labels.
```
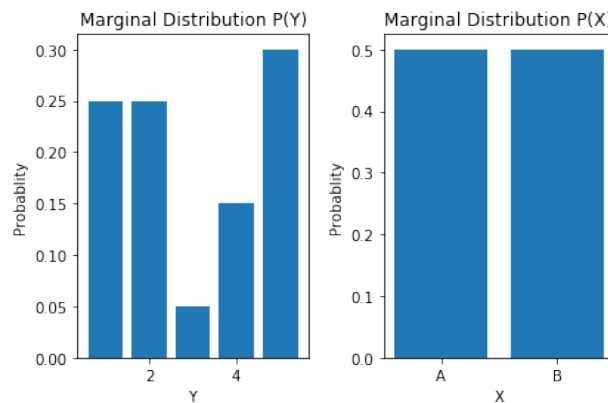


2. What is the value of the conditional probability $P(X = A | Y = 2)$? Compute the distribution $P(X|Y=2)$.

```
[5]: cond_prob = M[0,1]/P_Y[1]
     print('P(X = A|Y = 2) = '+str(cond_prob))
```

```
P(X = A|Y = 2) = 0.8
```

```
[6]: cond_prob_dist = M[:,1]/P_Y[1]
     print('P(X|Y = 2) = '+str(cond_prob_dist))
```

```
P(X|Y = 2) = [0.8 0.2]
```

3. Are the variables X and Y independent? Why or why not?

```
[7]: print('The are not independent because P(X=A|Y=2) = 0.8 while P(X=A) = 0.5.')
```

```
The are not independent because P(X=A|Y=2) = 0.8 while P(X=A) = 0.5.
```

**Example 6:**

```
[1]: import numpy as np
     from scipy.stats import binom
```

```
[2]: # Comoute the likelihood for a type A coin
     k = 68
     N = 100
     p = 0.5
     likelihood_A = binom.pmf(k, N, p)
     print(likelihood_A)
```

0.00011281697127222492

```
[3]: # Comoute the likelihood for a type B coin
     k = 68
     N = 100
     p = 0.75
     likelihood_B = binom.pmf(k, N, p)
     print(likelihood_B)
```

0.02475256392905903

```
[4]: # Create variables for the priors
     prior_A = 0.999
     prior_B = 0.001
```

```
[5]: # Compute the total probability
     total_probabilty = likelihood_A*prior_A + likelihood_B*prior_B
     print(total_probabilty)
```

0.00013745671823001173

```
[6]: # Compute the posterior probabilities
     probability_type_A = likelihood_A*prior_A/total_probabilty
     print(probability_type_A)
     probability_type_B = likelihood_B*prior_B/total_probabilty
     print(probability_type_B)
```

0.8199246697593958
0.18007533024060413

```
[7]: # Same computation, but using vectors
     likelihoods = np.asarray([likelihood_A,likelihood_B])
     priors = np.asarray([prior_A,prior_B])
     posterior_probabilities = likelihoods*priors/total_probabilty
     print(posterior_probabilities)
```

[0.81992467 0.18007533]

**Figure ??:**

```
[1]: import numpy as np
     import matplotlib.pyplot as plt
     from scipy.stats import norm
```

```
[2]: # Increase the font size (since this will be a small plot figure)
     plt.rcParams.update({'font.size': 20})
     # Create the x-values and pdf-values
     x = np.linspace(0,1,1000)
     pdf = norm.pdf(x,loc=0.5,scale=0.01)
     # Create the plot, then set the labels and title
     plt.plot(x, pdf,'b-', label='PDF ');
     plt.xlabel('p')
     plt.ylabel('PDF');
     plt.title('Prior Probabilty for $p=P($Heads$)$');
```
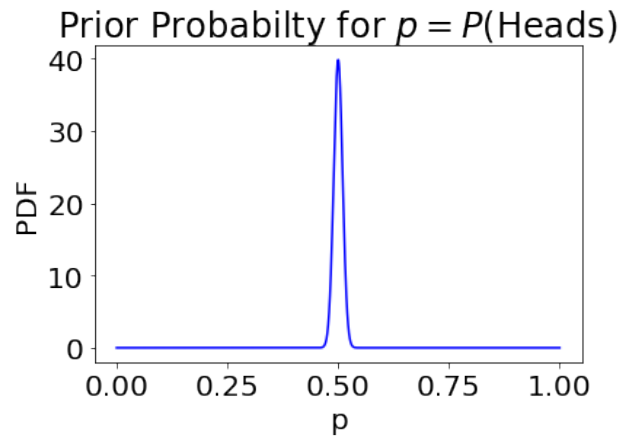
**Example 7:**

```
[1]: import numpy as np
     import matplotlib.pyplot as plt
     plt.rcParams['figure.dpi'] = 200
     from scipy.stats import norm, binom
```

NOTE: In this notebook we provide the numerical computation and plots for the example. Analytic solutions are provided in the example text.

```
[2]: # Choose sample points for the plot.  we select 1001 points evenly spaced in from 0 to␣
     ↪1.
     # We choose 1001 so the the stepsize is exactly 0.001, which is important for the␣
     ↪numerical integration.
     x = np.linspace(0,1,1001)
     dx = x[1]-x[0]
     k = 68
     N = 100
     # Compute the terms in Bayes Theorem
     prior = norm.pdf(x,loc=0.5,scale=0.01)
     likelihood = binom.pmf(k, N, x)
     total_probabiltiy = np.sum(prior*likelihood*dx)
     # Compute the posterior probability
     posterior = likelihood*prior/total_probabiltiy
```

```
[3]: # Confirm that the posterior is a true probability distribution by checking that its␣
     ↪integral is equal to 1.
     np.sum(posterior*dx)
```
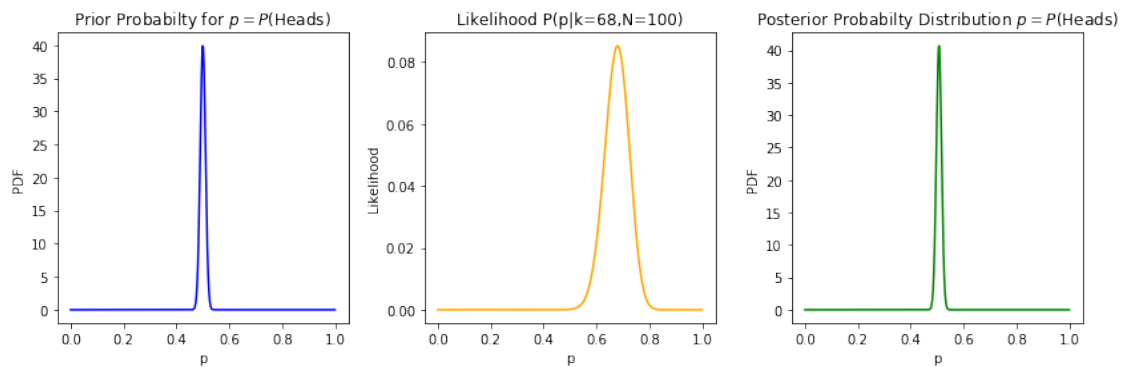
```
[3]: 1.0
```

```
[4]: # Set the figure size for the plots
     plt.rcParams['figure.figsize'] = [12, 4]

     # Plot of the prior probability distribution
     plt.subplot(1, 3, 1) # layout to make 1 row, 3 columns of plots, and create to the 1st␣
     ↪location.
     plt.plot(x, prior,'b-', label='Prior ');
     plt.xlabel('p')
     plt.ylabel('PDF');
     plt.title('Prior Probabilty for $p=P($Heads$)$');

     # Plot of the likelihood
     plt.subplot(1, 3, 2) # layout to make 1 row, 3 columns of plots, and create to the 1st␣
     ↪location.
     plt.plot(x, likelihood, color='orange', label='Likelihood ');
     plt.xlabel('p')
     plt.ylabel('Likelihood');
     plt.title('Likelihood P(p|k=68,N=100)');

     # Plot of the posterior probability distribution
     plt.subplot(1, 3, 3) # layout to make 1 row, 3 columns of plots, and create to the 1st␣
     ↪location.
     plt.plot(x, posterior,'g-', label='Posterior ');
```

```
plt.xlabel('p')
plt.ylabel('PDF');
plt.title('Posterior Probabilty Distribution $p=P($Heads$)$');

plt.tight_layout() # this cleans up the spacing for plots and labels.
```



```
[5]:  # Set the figure size for the plots
      plt.rcParams['figure.figsize'] = [8, 4]

      # Plot of the prior probability distribution
      plt.plot(x, prior,'b-', label='Prior ');
      plt.xlabel('p')
      plt.ylabel('PDF');
      plt.title('Prior Probabilty for $p=P($Heads$)$');

      # Plot of the likelihood
      plt.plot(x, likelihood*100, color='orange', label='Likelihood x 100');
      plt.xlabel('p')
      plt.ylabel('Likelihood');
      plt.title('Likelihood P(p|k=68,N=100)');

      # Plot of the posterior probability distribution
      plt.plot(x, posterior,'g-', label='Posterior ');
      plt.xlabel('p')
      plt.ylabel('PDF');
      plt.title('Posterior Probabilty Distribution $p=P($Heads$)$');

      plt.legend()
      plt.xlim([0.4,0.8])
      plt.tight_layout() # this cleans up the spacing for plots and labels.
```
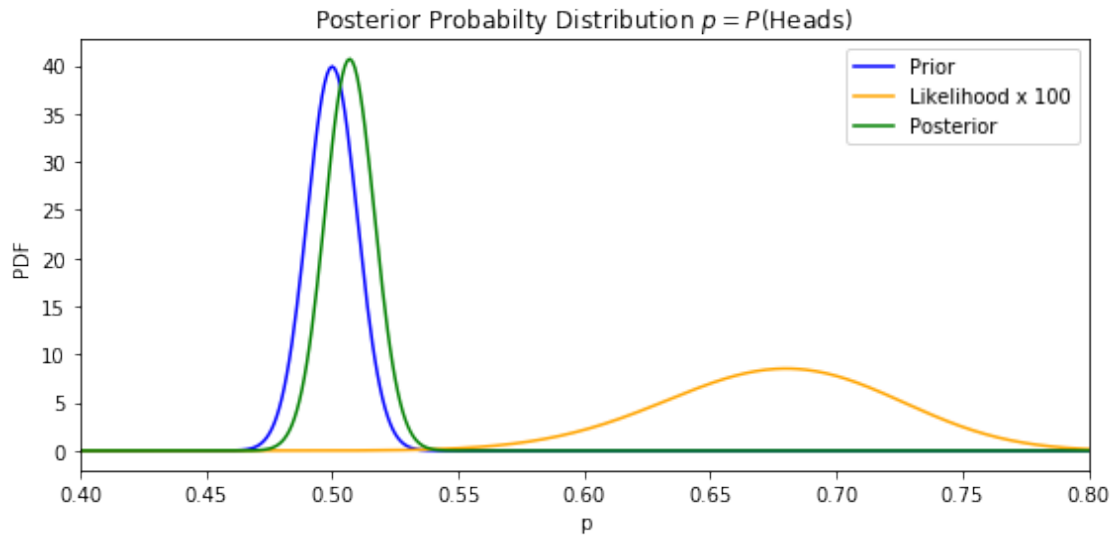
Posterior Probabilty Distribution $p = P(\text{Heads})$

```
[6]:  # Compute the middle 95%
      # Step 1: Compute the Cumulative Density Function (CDF)
      cdf = np.zeros(len(posterior))
      for idx in range(len(posterior)):
          cdf[idx] = np.sum(posterior[:idx])*dx
      # Step 2: Compute the credible interval
      a = x[np.argmin(np.abs(cdf-0.025))] # find the x-value where the cdf is closest to 0.
      →025
      b = x[np.argmin(np.abs(cdf-0.975))]# find the x-value where the cdf is closest to 0.975
      print(f'The probaiblity is 0.95 that the value for p in the interval [{a},{b}].')
```

The probaiblity is 0.95 that the value for p in the interval [0.488,0.527].

**Section 1.4:**

```
[1]: import warnings
     warnings.simplefilter(action='ignore', category=FutureWarning)
```

```
[2]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
     sns.set(color_codes=True) # set seaborn plotting style
     plt.rcParams['figure.figsize'] = [16, 4] # set the figure size for the plots
     sample_size = 10000
```
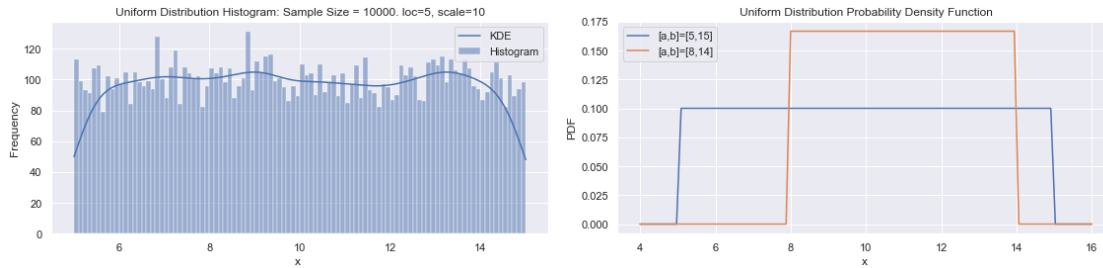
Uniform Distribution:

```
[3]: # import uniform distribution
     from scipy.stats import uniform

     # set parameters for a uniform distribution
     param_loc = 5
     param_scale = 10
     # generate samples from this distribution
     data_uniform = uniform.rvs(size=sample_size, loc=param_loc, scale=param_scale)

     # plot a historgram of the output
     plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to␣
      ↪location 1.
     ax1 = sns.histplot(data=data_uniform, bins=100, kde=True);
     ax1.set(xlabel='x ', ylabel='Frequency',
             title=f'Uniform Distribution Histogram: Sample Size = {sample_size}.␣
      ↪loc={param_loc}, scale={param_scale}');
     ax1.legend(['KDE','Histogram']);

     # plot the probability density function
     ax2 = plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to␣
      ↪location 2.
     x = np.linspace(4, 16, 100) # generate a list of x-values
     y1 = uniform.pdf(x, loc=param_loc, scale=param_scale) # compute the PDF at these␣
      ↪x-values
     ax2.plot(x,y1)
     y2 = uniform.pdf(x, loc=8, scale=6) # compute the PDF at these x-values
     ax2.plot(x,y2)
     ax2.legend(['[a,b]=[5,15]','[a,b]=[8,14]'])
     ax2.set(xlabel='x ', ylabel='PDF',
             title=f'Uniform Distribution Probability Density Function');
     plt.tight_layout()
```

```
[4]:  # compute the uniform pdf at three data points:
      x = np.asarray([3,8,10])
      print(uniform.pdf(x, loc=param_loc, scale=param_scale))
```

```
[0.  0.1 0.1]
```
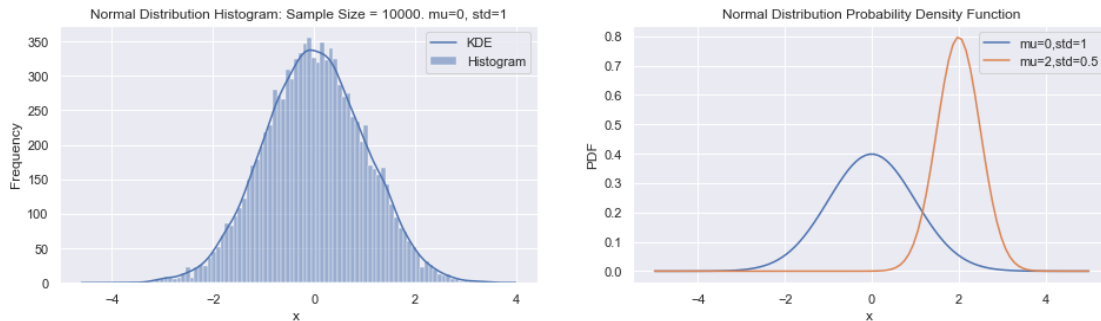
Normal Distribution:

```
[5]:  # import uniform distribution
      from scipy.stats import norm

      # set parameters for a uniform distribution
      param_loc = 0
      param_scale = 1
      # generate samples from this distribution
      data_normal = norm.rvs(size=sample_size, loc=param_loc, scale=param_scale)

      # plot a historgram of the output
      plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to␣
      ↪location 1.
      ax1 = sns.histplot(data=data_normal, bins=100, kde=True);
      ax1.set(xlabel='x ', ylabel='Frequency',
              title=f'Normal Distribution Histogram: Sample Size = {sample_size}.␣
      ↪mu={param_loc}, std={param_scale}');
      ax1.legend(['KDE','Histogram']);

      # plot the probability density function
      ax2 = plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to␣
      ↪location 2.
      x = np.linspace(-5, 5, 100) # generate a list of x-values
      y1 = norm.pdf(x, loc=param_loc, scale=param_scale) # compute the PDF at these x-values
      ax2.plot(x,y1)
      y2 = norm.pdf(x, loc=2, scale=0.5) # compute the PDF at these x-values
      ax2.plot(x,y2)
      ax2.legend(['mu=0,std=1','mu=2,std=0.5'])
      ax2.set(xlabel='x ', ylabel='PDF',
              title=f'Normal Distribution Probability Density Function');
```

```
[6]:  # compute the normal pdf at three data points:
      x = np.asarray([-2,0,1])
      print(norm.pdf(x, loc=param_loc, scale=param_scale))
```

```
[0.05399097 0.39894228 0.24197072]
```
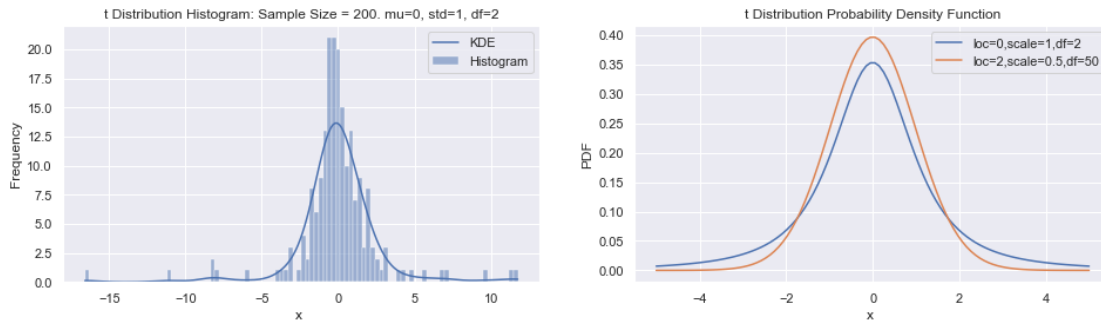
t-Distribution:

```
[7]:  from scipy.stats import t

      # set parameters for a uniform distribution
      param_loc = 0
      param_scale = 1
      param_df = 2
      # generate samples from this distribution
      data_tDist = t.rvs(size=200, loc=param_loc, scale=param_scale, df=param_df)

      # plot a historgram of the output
      plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to␣
      ↪location 1.
      ax1 = sns.histplot(data=data_tDist, bins=100, kde=True);
      ax1.set(xlabel='x ', ylabel='Frequency',
             title=f't Distribution Histogram: Sample Size = {200}. mu={param_loc},␣
      ↪std={param_scale}, df={param_df}');
      ax1.legend(['KDE','Histogram']);

      # plot the probability density function
      ax2 = plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to␣
      ↪location 2.
      x = np.linspace(-5, 5, 100) # generate a list of x-values
      y1 = t.pdf(x, loc=param_loc, scale=param_scale, df=param_df) # compute the PDF at␣
      ↪these x-values
      ax2.plot(x,y1)
      y2 = t.pdf(x, loc=param_loc, scale=param_scale, df=50) # compute the PDF at these␣
      ↪x-values
      ax2.plot(x,y2)
      ax2.legend(['mu=0,std=1,df=2','mu=2,std=0.5,df=50'])
      ax2.set(xlabel='x ', ylabel='PDF',
             title=f't Distribution Probability Density Function');
```

t Distribution Histogram: Sample Size = 200. mu=0, std=1, df=2      t Distribution Probability Density Function

```
[8]:  # compute the normal pdf at three data points:
      x = np.asarray([-2,0,20])
      print(t.pdf(x, loc=param_loc, scale=param_scale, df=param_df))
```

```
[6.80413817e-02 3.53553391e-01 1.24068325e-04]
```

Bernouli Distribution:

```
[9]:  # import uniform distribution
      from scipy.stats import bernoulli

      # set parameters for a uniform distribution
      param_p = 0.3
      # generate samples from this distribution
      data_bern = bernoulli.rvs(size=sample_size,p=param_p)

      # plot a historgram of the output
      plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to␣
      ↪location 1.
      ax1 = sns.histplot(data=data_bern, bins=10, kde=False);
      ax1.set(xlabel='x ', ylabel='Frequency',
              title=f'Bernoili Distribution Histogram: Sample Size = {sample_size}.␣
      ↪p={param_p}');
      ax1.legend(['Histogram']);

      # plot the probability density function
      ax2 = plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to␣
      ↪location 2.
      x = np.asarray([0,1])# generate a list of x-values
      y1 = bernoulli.pmf(x, p=param_p) # compute the PDF at these x-values
      ax2.bar(x,y1, width =0.3)
      y2 = bernoulli.pmf(x, p=0.5) # compute the PDF at these x-values
      ax2.bar(x+0.35,y2, width =0.3)
      ax2.legend(['p=0.3','p=0.5'])
      ax2.set(xlabel='x ', ylabel='PMF',
              title=f'Bernoili Distribution Probability Mass Function');
```
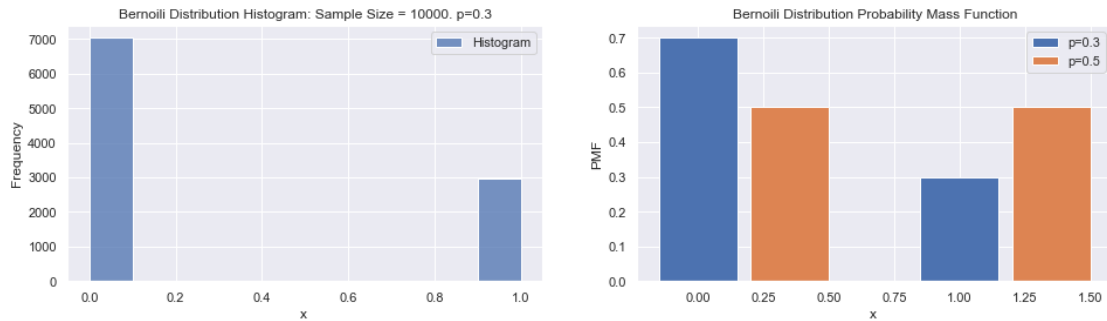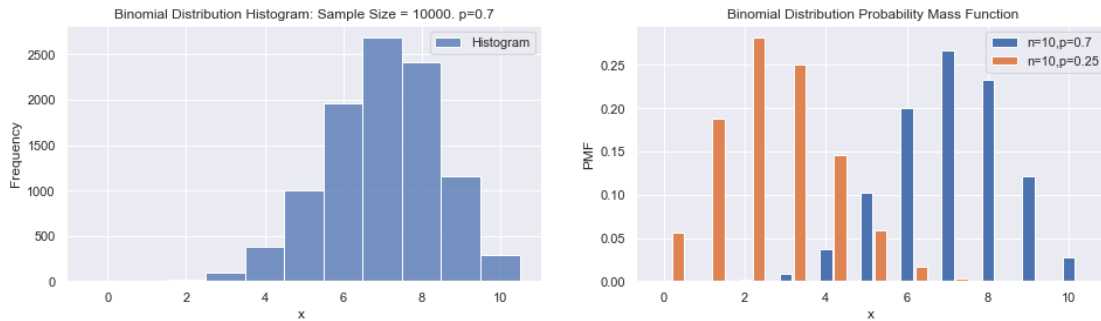
Binomial Distribution:

```
[10]: from scipy.stats import binom

      # Generate Binomial Data
      param_n = 10
      param_p = 0.7
      data_binom = binom.rvs(size=sample_size, n=param_n,p=param_p)

      # plot a historgram of the output
      plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to␣
       ↪location 1.
      ax1 = sns.histplot(data=data_binom, bins=[-0.5,0.5,1.5,2.5,3.5,4.5,5.5,6.5,7.5,8.5,9.
       ↪5,10.5], kde=False);
      ax1.set(xlabel='x ', ylabel='Frequency',
              title=f'Binomial Distribution Histogram: Sample Size = {sample_size}.␣
       ↪p={param_p}');
      ax1.legend(['Histogram']);

      # plot the probability density function
      ax2 = plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to␣
       ↪location 2.
      x = np.asarray([0,1,2,3,4,5,6,7,8,9,10])# generate a list of x-values
      y1 = binom.pmf(x, n=param_n,p=param_p) # compute the PDF at these x-values
      ax2.bar(x,y1, width =0.3)
      y2 = binom.pmf(x, n=param_n,p=0.25) # compute the PDF at these x-values
      ax2.bar(x+0.35,y2, width =0.3)
      ax2.legend(['n=10,p=0.7','n=10,p=0.25'])
      ax2.set(xlabel='x ', ylabel='PMF',
              title=f'Binomial Distribution Probability Mass Function');
```

38

[11]: 
```python
# compute the binomial pmf at three data points:
x = np.asarray([0,1,5,7,8,9])
print(binom.pmf(x, n=param_n, p=param_p))
```

```
[5.90490000e-06 1.37781000e-04 1.02919345e-01 2.66827932e-01
 2.33474440e-01 1.21060821e-01]
```

Poisson Distribution:

[12]: 
```python
# import uniform distribution
from scipy.stats import poisson

# set parameters for a uniform distribution
param_mu = 2
param_loc = 0
# generate samples from this distribution
data_poisson = poisson.rvs(size=sample_size, loc=param_loc, mu=param_mu)

# plot a historgram of the output
plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to␣
↪location 1.
ax1 = sns.histplot(data=data_poisson, bins=[-0.5,0.5,1.5,2.5,3.5,4.5,5.5,6.5,7.5,8.5,9.
↪5,10.5], kde=False);
ax1.set(xlabel='x ', ylabel='Frequency',
        title=f'Poisson Distribution Histogram: Sample Size = {sample_size}.␣
↪mu={param_loc}, std={param_scale}');
ax1.legend(['KDE','Histogram']);

# plot the probability density function
ax2 = plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to␣
↪location 2.
x = np.asarray([0,1,2,3,4,5,6,7,8,9,10])# generate a list of x-values
y1 = poisson.pmf(x, loc=param_loc, mu=param_mu) # compute the PDF at these x-values
ax2.bar(x,y1, width =0.3)
y2 = poisson.pmf(x, loc=3, mu=1) # compute the PDF at these x-values
ax2.bar(x+0.35,y2, width =0.3)
ax2.legend(['loc=0,mu=2','loc=3,mu=1'])
ax2.set(xlabel='x ', ylabel='PMF',
        title=f'Poisson Distribution Probability Mass Function');
```
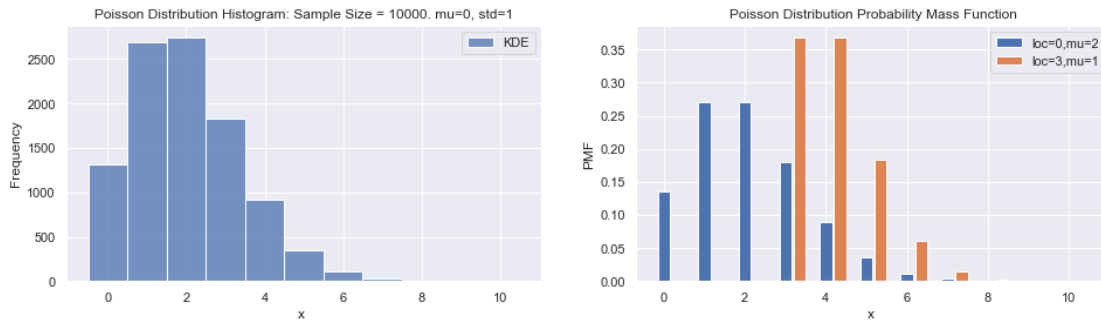
Poisson Distribution Histogram: Sample Size = 10000. mu=0, std=1 | Poisson Distribution Probability Mass Function

```
[13]:  # compute the binomial pmf at three data points:
       x = np.asarray([0,1,5,7,8,9])
       print(poisson.pmf(x, loc=param_loc, mu=param_mu))
```

```
[1.35335283e-01 2.70670566e-01 3.60894089e-02 3.43708656e-03
 8.59271640e-04 1.90949253e-04]
```

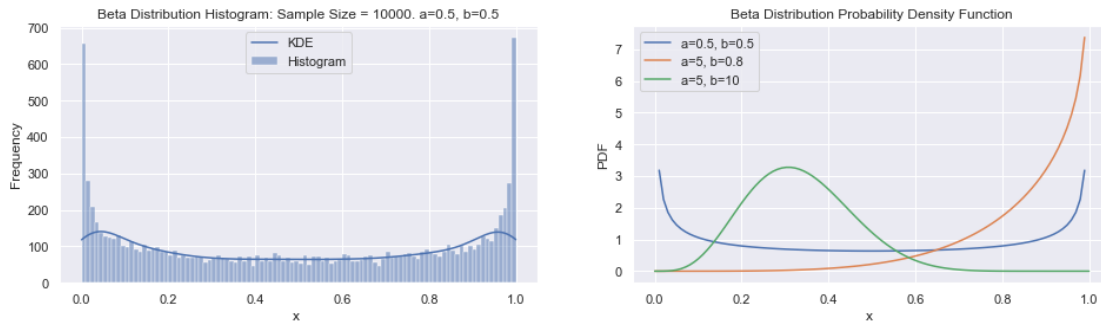Beta Distribution:

```
[14]:  from scipy.stats import beta

       # Generate Poisson Data
       param_a = .5
       param_b = .5
       data_beta = beta.rvs(a=param_a, b=param_b, size=sample_size)

       # plot a historgram of the output
       plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to␣
        ↪location 1.
       ax1 = sns.histplot(data=data_beta, bins=100, kde=True);
       ax1.set(xlabel='x ', ylabel='Frequency',
               title=f'Beta Distribution Histogram: Sample Size = {sample_size}. a={param_a},␣
        ↪b={param_b}');
       ax1.legend(['KDE','Histogram']);

       # plot the probability density function
       ax2 = plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to␣
        ↪location 2.
       x = np.linspace(0, 1, 100) # generate a list of x-values
       y1 = beta.pdf(x, a=param_a, b=param_b) # compute the PDF at these x-values
       ax2.plot(x,y1)
       y2 = beta.pdf(x, a=5, b=0.8) # compute the PDF at these x-values
       ax2.plot(x,y2)
       y3 = beta.pdf(x, a=5, b=10) # compute the PDF at these x-values
       ax2.plot(x,y3)
       ax2.legend(['a=0.5, b=0.5','a=5, b=0.8','a=5, b=10'])
       ax2.set(xlabel='x ', ylabel='PDF',
               title=f'Beta Distribution Probability Density Function');
```

Beta Distribution Histogram: Sample Size = 10000. a=0.5, b=0.5 | Beta Distribution Probability Density Function

[15]:
```python
# compute the beta pdf at three data points:
x = np.asarray([0,1,5,7,8,9])
print(binom.pmf(x, n=param_n, p=param_p))
```

```
[5.90490000e-06 1.37781000e-04 1.02919345e-01 2.66827932e-01
 2.33474440e-01 1.21060821e-01]
```

Gamma Distribution:

[16]:
```python
from scipy.stats import gamma

# Generate Gamma Data
param_a = 3 # shape parameter, sometimes denoted k or alpha
param_scale = 2 # this is the scale parameter theta.  Sometime this is given as rate␣
↪parameter called beta, where theta=1/beta.
data_gamma = gamma.rvs(size=sample_size, a=param_a, scale=param_scale)

# plot a historgram of the output
plt.subplot(1, 2, 1) # layout to make 1 row, 2 columns of plots, and create to␣
↪location 1.
ax1 = sns.histplot(data=data_gamma, bins=100, kde=True);
ax1.set(xlabel='x ', ylabel='Frequency',
        title=f'Gamma Distribution Histogram: Sample Size = {sample_size}. a={param_a},␣
↪scale={param_scale}');
ax1.legend(['KDE','Histogram']);

# plot the probability density function
ax2 = plt.subplot(1, 2, 2) # layout to make 1 row, 2 columns of plots, and create to␣
↪location 2.
x = np.linspace(0, 40, 500) # generate a list of x-values
y1 = gamma.pdf(x, a=param_a, scale=param_scale) # compute the PDF at these x-values
ax2.plot(x,y1)
y2 = gamma.pdf(x, a=1.5, scale=2) # compute the PDF at these x-values
ax2.plot(x,y2)
y3 = gamma.pdf(x, a=5, scale=4) # compute the PDF at these x-values
ax2.plot(x,y3)
ax2.legend(['a=3, scale=2','a=1.5, scale=2','a=5, scale=4'])
ax2.set(xlabel='x ', ylabel='PDF',
        title=f'Gamma Distribution Probability Density Function');
```

Gamma Distribution Histogram: Sample Size = 10000. a=3, scale=2



Gamma Distribution Probability Density Function
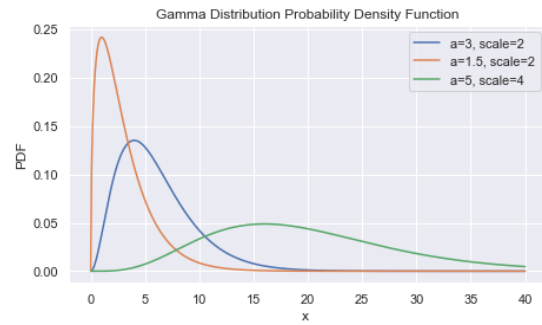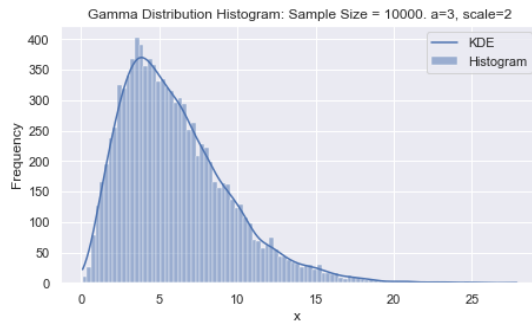
[17]:
```python
# compute the beta pdf at three data points:
x = np.asarray([0,1,5,10,15])
print(gamma.pdf(x, a=param_a, scale=param_scale))
```

```
[0.         0.03790817 0.12825781 0.04211217 0.00777775]
```

**Section 1.5:**

```
[1]: import warnings
     warnings.simplefilter(action='ignore', category=FutureWarning)
```

```
[2]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     from mpl_toolkits.mplot3d import Axes3D # <--- This is important for 3d plotting
     import seaborn as sns
     sns.set(color_codes=True) # set seaborn plotting style
     plt.rcParams['figure.figsize'] = [16, 4] # set the figure size for the plots
     sample_size = 10000
```

Multivariate Gaussian

```
[3]: # Generate Mulitvariate Gaussian Data and plot with Seaborn
     param_mean = [0, 2]
     param_cov = [(1, .5), (.5, 1)]
     data = np.random.multivariate_normal(param_mean, param_cov, size=sample_size)
     # create a data frame from the data
     df = pd.DataFrame(data, columns=["x", "y"])

     # Create the Plot
     ax = sns.jointplot(x="x", y="y", data=df);
     ax.fig.subplots_adjust(top=0.9)
     ax.fig.suptitle("Scatterplot of Guassian Distribution Data");

     ax = sns.jointplot(x="x", y="y", data=df, kind="kde");
     ax.fig.subplots_adjust(top=0.9)
     ax.fig.suptitle("Kenel Density Estimation of Gausssian Distribution Data");

     ax = sns.jointplot(x="x", y="y", data=df, kind="hex", color="k");
     ax.fig.subplots_adjust(top=0.9)
     ax.fig.suptitle("Hexbin Plot of Guassian Distribution Data");
```

Scatterplot of Guassian Distribution Data | Kenel Density Estimation of Gausssian Distribution Data

Hexbin Plot of Guassian Distribution Data
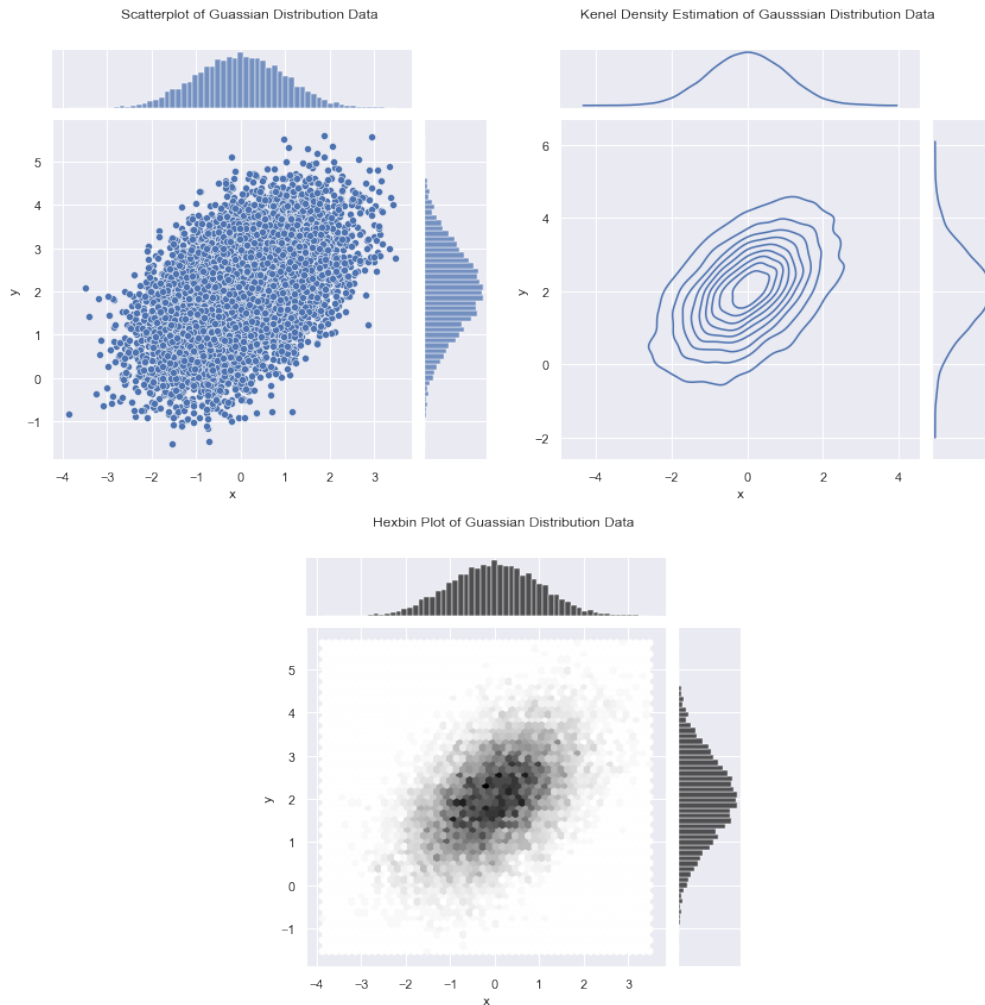
```
[4]: plt.rcParams['figure.figsize'] = [8, 6] # set the figure size for the plots
     sample_size = 1000
     # 3D scatterplot of data from two 3D  multivariate normal distributions
     # mean and cov computed from Fisher's Iris dataset using features
     ↪'sepal_length','sepal_width', and 'petal_length'
     mu_Iris_setosa = [5.01,3.42,1.46]
     cov_Iris_setosa = [(.124249,.100298,0.016139), (.100298,.145180,0.011682),(0.016139,0.
     ↪011682,0.030106)]
     data_setosa = np.random.multivariate_normal(mu_Iris_setosa, cov_Iris_setosa,
     ↪size=sample_size)
     # mean and cov computed from Fisher's Iris dataset using features
     ↪'sepal_length','sepal_width', and 'petal_length'
     mu_Iris_versicolor = [5.94,2.77,4.26]
     cov_Iris_versicolor = [(.2664327,.0851837,.1828980), (.0851837,.0984694,.0826531),(.
     ↪1828980,.0826531,.2208163)]
     data_versicolor = np.random.multivariate_normal(mu_Iris_versicolor,
     ↪cov_Iris_versicolor, size=sample_size)
```

44

```
ax = plt.axes(projection='3d');
ax.scatter(data_setosa[:,0],data_setosa[:,1],data_setosa[:,2], label='Normal Data1',␣
 ↪alpha=0.5);
ax.scatter(data_versicolor[:,0],data_versicolor[:,1],data_versicolor[:,2],␣
 ↪label='Normal Data2', alpha=0.5);
ax.legend(loc='lower right');
ax.set_title('Scatterplot of Data from Two Multivariate Normal Distributions');
```
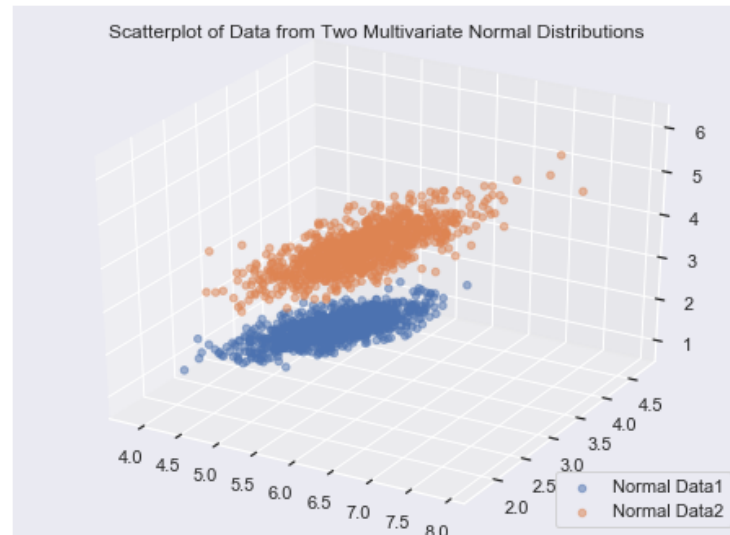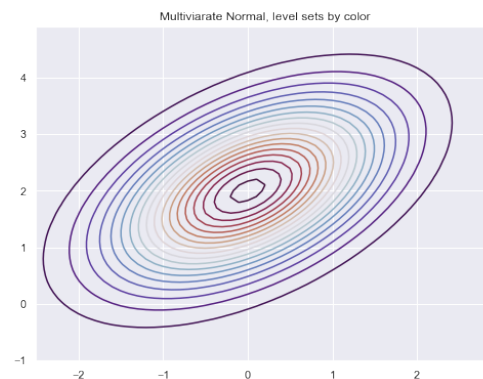


[5]:
```
from scipy.stats import multivariate_normal
# Create the Y,Y grid
X, Y = np.meshgrid(np.arange(-2.5, 3, 0.1), np.arange(-1, 5, 0.1))
pos = np.dstack((X,Y))
# Compute the PDF at the grid values
Z = multivariate_normal.pdf(pos, mean=param_mean, cov=param_cov);
```
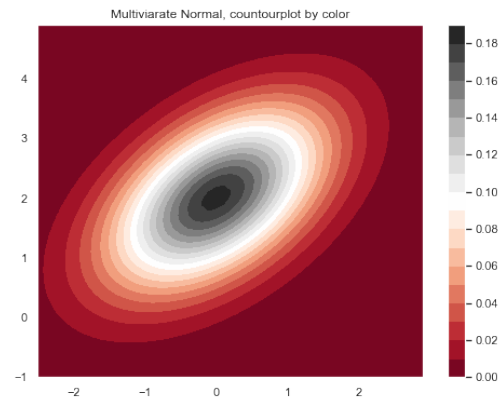
[6]:
```
# basic countorplot showing level sets
plt.contour(X, Y, Z, 20, cmap='twilight_shifted');
plt.title('Multiviarate Normal, level sets by color');
```

```
[7]:   # contourplot using colors to show z-values
       plt.contourf(X, Y, Z, 20, cmap='RdGy')
       plt.colorbar();
       plt.title('Multiviarate Normal, countourplot by color');
```



```
[8]:   # contour plot with colors and level sets
       contours = plt.contour(X, Y, Z, 3, colors='black') # create black countors
       plt.clabel(contours, inline=True, fontsize=8) # add countors with labels
       plt.imshow(Z, extent=[np.min(X), np.max(X), np.min(Y), np.max(Y)], origin='lower',
                  cmap='RdGy', alpha=0.5) # add background color
       plt.colorbar();
       plt.title('Multiviarate Normal, countourmap with labeled level sets');
```
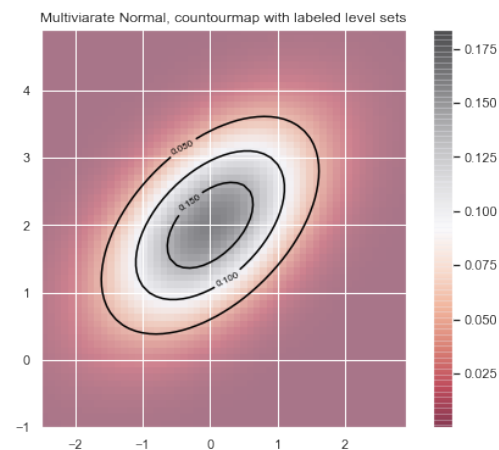


```
[9]:   plt.rcParams['figure.figsize'] = [16,6] # set the figure size for the plots

       # 3D plot using a grid
```

```python
ax1 = plt.subplot(1, 2, 1, projection='3d') # layout to make 1 row, 2 columns of
 ↪plots, and create to location 2.
ax1.plot_surface(X, Y, Z, cmap='jet');
ax1.set_title('Multiviarate Normal, plot_surface');

# 3D plot using a colored grid and transparent faces
from matplotlib import cm# Normalize the colors based on Z value
ax2 = plt.subplot(1, 2, 2, projection='3d') # layout to make 1 row, 2 columns of
 ↪plots, and create to location 2.
norm = plt.Normalize(Z.min(), Z.max())
colors = cm.jet(norm(Z))
surf = ax2.plot_surface(X, Y, Z, facecolors=colors, shade=False)
surf.set_facecolor((0,0,0,0))
ax2.set_title('Multiviarate Normal, plot_surface with transparent faces');
```
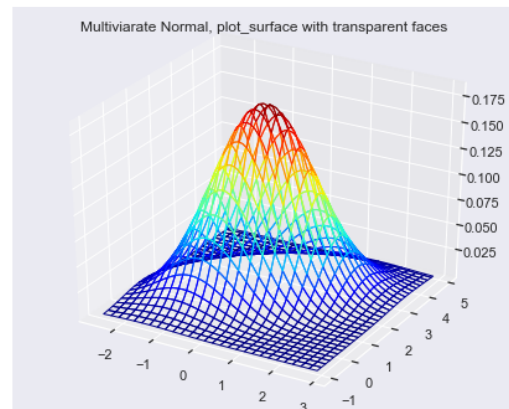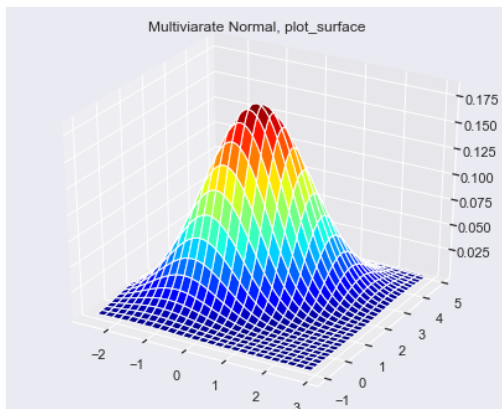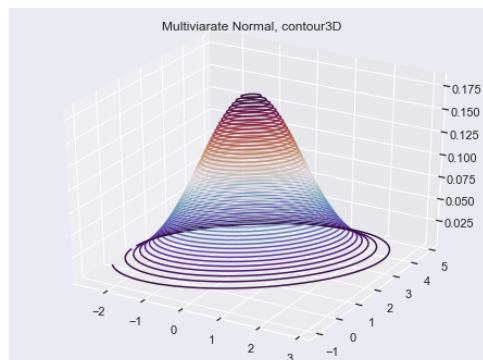


```python
[10]: plt.rcParams['figure.figsize'] = [8, 6] # set the figure size for the plots
      # 3d plot using level-set countours
      ax = plt.axes(projection='3d')
      ax.contour3D(X, Y, Z, 55, cmap='twilight_shifted');
      ax.set_title('Multiviarate Normal, contour3D');
```



47

```
[11]:  # scipy does not have a multivariate t-distribution.  we have to create the function␣
       ↪for it
       from math import gamma

       def multivariate_t_pdf(x, mu, Sigma, df):
           '''
           Multivariate t-student density. Returns the density
           of the function at points specified by x.

           input:
               x = parameter (n-d numpy array; will be forced to 2d)
               mu = mean (d dimensional numpy array)
               Sigma = scale matrix (dxd numpy array)
               df = degrees of freedom

           Edited from https://stackoverflow.com/questions/29798795/
       ↪multivariate-student-t-distribution-with-python
           which previously edited from Edited from: http://stackoverflow.com/a/29804411/
       ↪3521179
           '''
           x = np.atleast_2d(x) # requires x as 2d
           nD = Sigma.shape[0] # dimensionality
           numerator = gamma(1.0 * (nD + df) / 2.0)
           denominator = (
                   gamma(1.0 * df / 2.0) *
                   np.power(df * np.pi, 1.0 * nD / 2.0) *
                   np.power(np.linalg.det(Sigma), 1.0 / 2.0) *
                   np.power(
                       1.0 + (1.0 / df) *
                       np.diagonal(
                           np.dot( np.dot(x - mu, np.linalg.inv(Sigma)), (x - mu).T)
                       ),
                       1.0 * (nD + df) / 2.0
                       )
                   )
           return 1.0 * numerator / denominator
```

```
[12]:  # Generate Mulitvariate Gaussian Data and plot with Seaborn
       plt.rcParams['figure.figsize'] = [16,6] # set the figure size for the plots
       sample_size = 100
       param_mean = np.asarray([0, 2])
       param_cov = np.asarray([(1, .5), (.5, 1)])
       param_df = 1.5

       X, Y = np.meshgrid(np.arange(-2.5, 3, 0.1), np.arange(-1, 5, 0.1))
       pos = np.dstack((X,Y))
       # Compute the PDF at the grid values
       Z = multivariate_t_pdf(np.reshape(pos,[60*55,2]), param_mean, param_cov, param_df);
       Z = np.reshape(Z,[60,55])

       # T-DISTRIBUTION
       ax1 = plt.subplot(1, 2, 1, projection='3d') # layout to make 1 row, 2 columns of␣
       ↪plots, and create to location 2.
```

```
# 3d plot using level-set countours
ax1.contour3D(X, Y, Z, 55, cmap='twilight_shifted');
ax1.set_title('t Distribution');

# NORMAL DISTRIBUTION
ax2 = plt.subplot(1, 2, 2, projection='3d') # layout to make 1 row, 2 columns of␣
 ↪plots, and create to location 2.
# Compute the PDF at the grid values
Z_n = multivariate_normal.pdf(pos, mean=param_mean, cov=param_cov);
# 3d plot using level-set countours
ax2.contour3D(X, Y, Z_n, 55, cmap='twilight_shifted');
ax2.set_title('Normal Distribution');
```

c:\python37\lib\site-packages\numpy\core\_asarray.py:136:
VisibleDeprecationWarning: Creating an ndarray from ragged nested sequences
(which is a list-or-tuple of lists-or-tuples-or ndarrays with different lengths
or shapes) is deprecated. If you meant to do this, you must specify
'dtype=object' when creating the ndarray
  return array(a, dtype, copy=False, order=order, subok=True)
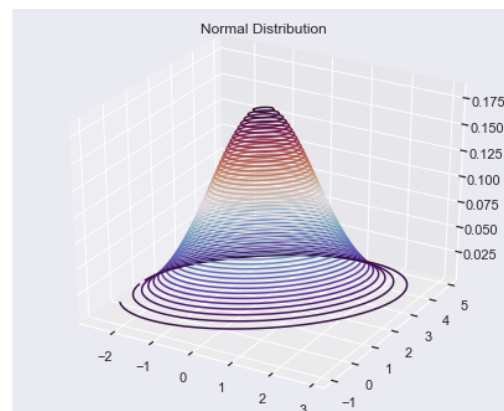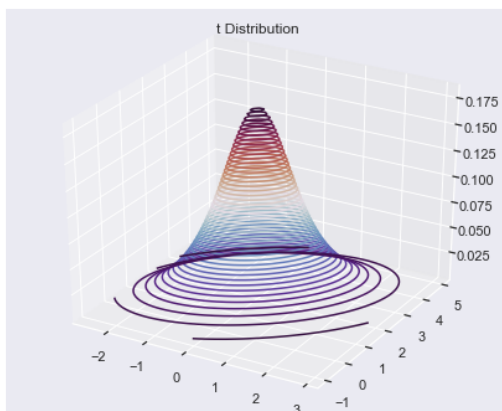


Dirichlet Distribution:

```
[13]: from scipy.stats import dirichlet
      plt.rcParams['figure.figsize'] = [16,12] # set the figure size for the plots
      sample_size = 1000

      ax1 = plt.subplot(2, 2, 1, projection='3d') # layout to make 1 row, 2 columns of␣
       ↪plots, and create to location 2.
      alpha = np.array([.1,.1,.1])  # specify concentration parameters
      data = dirichlet.rvs(alpha=alpha, size=sample_size)
      ax1.scatter(data[:,0],data[:,1],data[:,2], alpha=0.5);
      ax1.set_xlabel(r'$\theta_1$')
      ax1.set_ylabel(r'$\theta_2$')
      ax1.set_zlabel(r'$\theta_3$')
      ax1.set_title(rf'Dirchlet Distribution, $\alpha$={alpha}.')
      ax1.view_init(30, 45)
```

```python
ax2 = plt.subplot(2, 2, 2, projection='3d') # layout to make 1 row, 2 columns of
 →plots, and create to location 2.
alpha = np.array([2,2,2])  # specify concentration parameters
data = dirichlet.rvs(alpha=alpha, size=sample_size)
ax2.scatter(data[:,0],data[:,1],data[:,2], alpha=0.5);
ax2.set_xlabel(r'$\theta_1$')
ax2.set_ylabel(r'$\theta_2$')
ax2.set_zlabel(r'$\theta_3$')
ax2.set_title(rf'Dirchlet Distribution, $\alpha$={alpha}.')
ax2.view_init(30, 45)


ax3 = plt.subplot(2, 2, 3, projection='3d') # layout to make 1 row, 2 columns of
 →plots, and create to location 2.
alpha = np.array([10,10,10])  # specify concentration parameters
data = dirichlet.rvs(alpha=alpha, size=sample_size)
ax3.scatter(data[:,0],data[:,1],data[:,2], alpha=0.5);
ax3.set_xlabel(r'$\theta_1$')
ax3.set_ylabel(r'$\theta_2$')
ax3.set_zlabel(r'$\theta_3$')
ax3.set_title(rf'Dirchlet Distribution, $\alpha$={alpha}.')
ax3.view_init(30, 45)



ax4 = plt.subplot(2, 2, 4, projection='3d') # layout to make 1 row, 2 columns of
 →plots, and create to location 2.
alpha = np.array([1,4,0.5])  # specify concentration parameters
data = dirichlet.rvs(alpha=alpha, size=sample_size)
ax4.scatter(data[:,0],data[:,1],data[:,2], alpha=0.5);
ax4.set_xlabel(r'$\theta_1$')
ax4.set_ylabel(r'$\theta_2$')
ax4.set_zlabel(r'$\theta_3$')
ax4.set_title(rf'Dirchlet Distribution, $\alpha$={alpha}.')
ax4.view_init(30, 45)

plt.tight_layout()
```

Dirchlet Distribution, $\alpha=[0.1\ 0.1\ 0.1]$.

Dirchlet Distribution, $\alpha=[2\ 2\ 2]$.

Dirchlet Distribution, $\alpha=[10\ 10\ 10]$.

Dirchlet Distribution, $\alpha=[1.\ 4.\ 0.5]$.