Module 2: Fundamentals of Machine Learning

Quick Reference Guide

Learning Outcomes:

1. Analyze measures of central tendency for univariate and multivariate distributions
2. Create and interpret visual plots, including probability density functions, histograms, scatter plots, pair plots, and correlation matrices
3. Apply the Law of Large Numbers to a given population
4. Discuss applications of the Central Limit Theorem
5. Interpret a sample covariance matrix using Python
6. Approximate correlation from scatterplots
7. Identify conditional probabilities

Uniform Distribution

Distributions are a basic building block of statistics. To work with distributions in code, you can use SciPy. SciPy is an open-source collection of Python libraries for doing scientific and mathematical work.

SciPy is divided into a number of packages covering different areas:

- Linear algebra (scipy.linalg)
- Optimization (scipy.optimize)
- Interpolation (scipy.interpolate)
- Statistics (scipy.stats)

The statistics package SciPy stats provides implementations of about 90 different probability distributions.

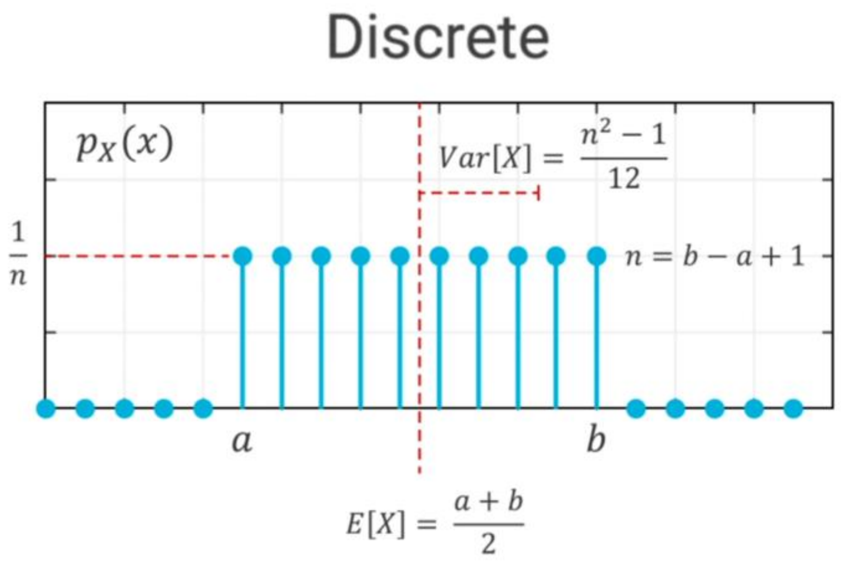
Two distributions that are important for data scientists are:

- **Uniform** distribution
- **Gaussian** or **normal** distribution

The uniform distribution:

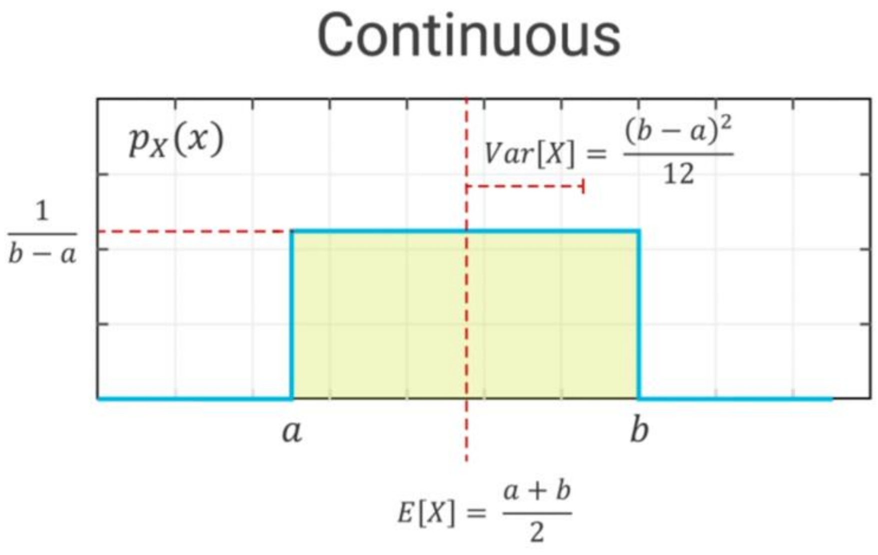
- is the simplest distribution
- models situations where the outcomes are between two values, a and b, and they are all equally probable

Two versions of the uniform distribution that are especially worth exploring are: **discrete** and **continuous**.



In a **discrete uniform distribution**, there are n possible outcomes ranging from a low value a to a high value b.

When you roll a die, for example, the low value is one, the high value is six, and there are a total of $n = 6$ possible outcomes. The probability of any one outcome, is $\frac{1}{n}$.



In a continuous uniform distribution, all values between a and b are possible. The probability density function equals $\frac{1}{b-a}$, so that the area under the curve, which is the base times the height, equals 1.

The **expectation** of both discrete and continuous uniform distributions is at the midpoint between a and b, because both are **symmetric distributions**.

The formulas for the **variance** are a little different. The formula for the variance of a discrete distribution is $\frac{n^2-1}{12}$ and the formula for the variance of a continuous distribution is $\frac{(b-a)^2}{12}$.

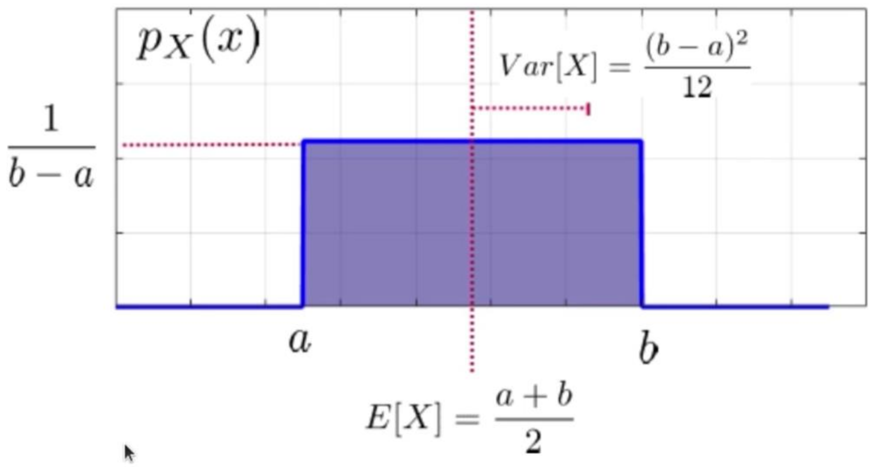
Uniform Python

You can use SciPy stats to work with **probability distributions**. If you look at **uniform distribution**, the proper import statement to use is:

```
from scipy.stats import uniform
import numpy as np
import matplotlib.pyplot as plt
```

You import the uniform distribution class, as well as your standard imports, **numpy** and **matplotlib.** Next, you have to create **a distribution object**.

A uniform distribution looks like this:

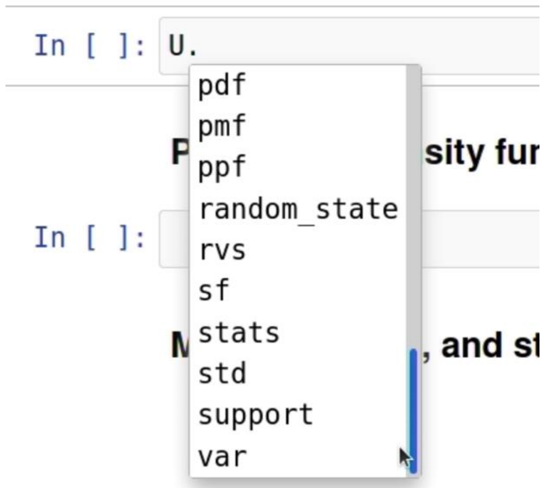


It spans from a to b, and its value is $\frac{1}{b-a}$. So to create the uniform distribution object, you call it uniform, assign it to the variable **U**, and pass in the **loc** and **scale** parameters: **U = uniform(loc=10, scale=5)**.

The parameters **loc** and **scale** are **common** for all of the distributions in SciPy stats and they take on different meanings depending on the distribution that you are using. For a uniform distribution**, loc is the left edge** of the distribution, and **scale is the width**.

Methods

What methods can you call for this uniform object? To obtain a list of all of the methods that are available in this object, you type **U**. and then press the Tab key.


```
In [ ]: U.
```

This list includes the **probability distribution function (PDF)**, rvs function (sampling function), standard deviation, variance, and various others.

The probability distribution function

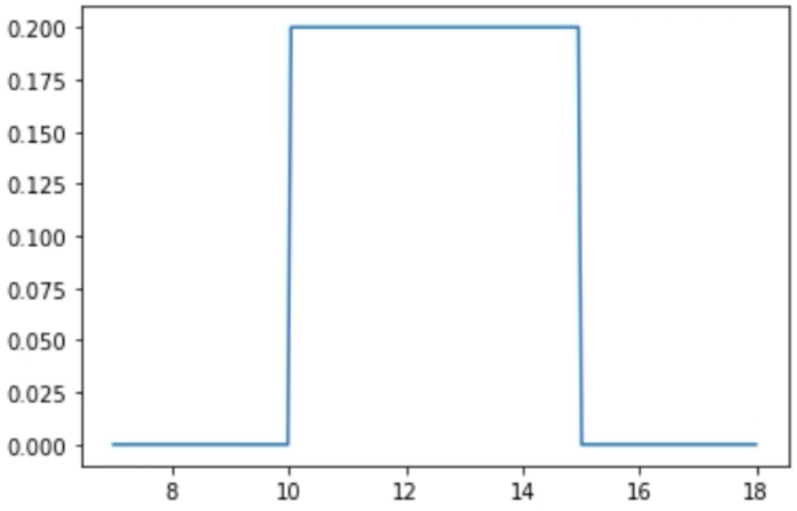
If you call **U.pdf(8)**, the output is 0.0.

You can also call **.pdf()** on an **array: U.pdf([8, 12, 20]).**

In addition, you can create a large array of evenly-spaced values between 7 and 18, using NumPy's **linspace()** function. For example, say you want to create 200 values, which you call **upoints: upoints = np.linspace(7, 18, 200).**

And you can pass upoints into the pdf, which you call ppoints: **ppoints = U.pdf(upoints).**

Next, you can plot upoints versus ppoints**: plt.plot(upoints,ppoints)**. This is the graph that you obtain:



Mean, Variance, and Standard Deviation

You can also evaluate the mean, the variance, and the standard deviation of the uniform distribution. Based on the theory, the values should be:

- **Mean**: $mean(U) = \frac{a+b}{2}$. The mean is the center value, which is between 10 and 15. It should be 12.5.

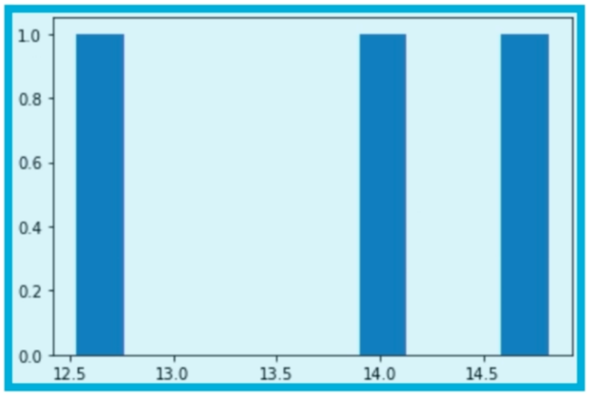
- **Variance**: $var(U) = \frac{b-(a)^2}{12}$. The variance is 25 divided by 12. It will be a little bit more than 2.
- **Standard deviation**: $std(U) = \frac{b-a}{\sqrt{12}}$. The standard deviation is the square root of the variance. The square root of 2 is about 1.4.

Random variates

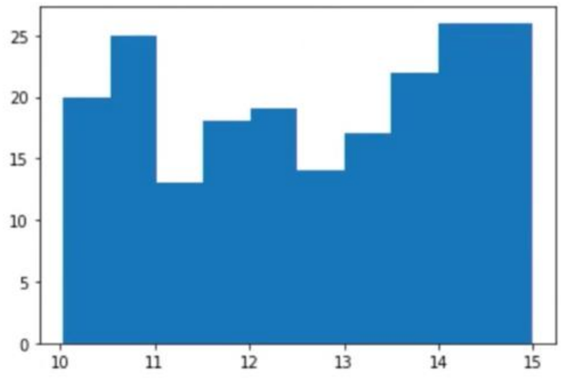
Another important method to use is **rvs**, which stands for **random variates**. Basically, rvs samples the distribution. So if you call **U.rvs(),** you get a single sample of the uniform distribution. In Jupyter Notebook, you can call it many times by pressing Ctrl + Enter.

You can sample many times at once by passing in a size parameter. If you say size equals three, you get an array of three samples: **U.rvs(size=3)**.

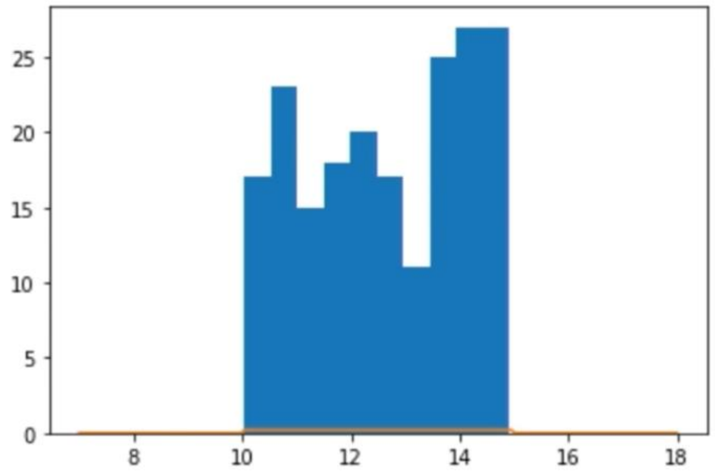
You can make a **histogram** with all of these samples. In this instance, the histogram with three samples does not look very uniform.



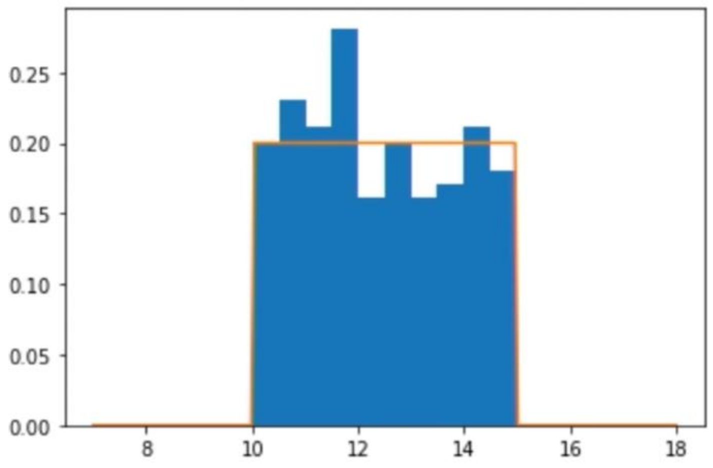
You can make it 200 samples if you pass in: **plt.hist(U.rvs(size=200)).** This is the resulting plot:



You can compare the two plots in the same plot. You add: **plt.plot(upoints,ppoints).**



The resulting plot shows that they are in the same range, but the histogram is a lot taller. So, to scale the histogram to the size of the distribution, you specify density=True: **plt.hist(U.rvs(size=200), density=True).**



So the histogram begins to resemble the distribution if you make the size of the sample larger. So if you make it 100,000 it is basically the same.

Gaussian Distribution

The **Gaussian** or **normal distribution** is the most important and widely-used distribution for statisticians and scientists. It is often referred to as the **bell curve**, and it pops up a lot in data science applications.

Examples of things that are often normally distributed, include:

- Measurements of body temperature and height
- Sizes of plants and animals
- Human technology, such as manufacturing and economics

So why is the normal distribution so ubiquitous? Normal distribution applies to things that are **aggregates of many similar parts**. So for example, in a measurement device, the reading you get can be affected by a myriad of small distortions, which add up to produce the total error. This adding up of small variations is what produces a normal distribution.

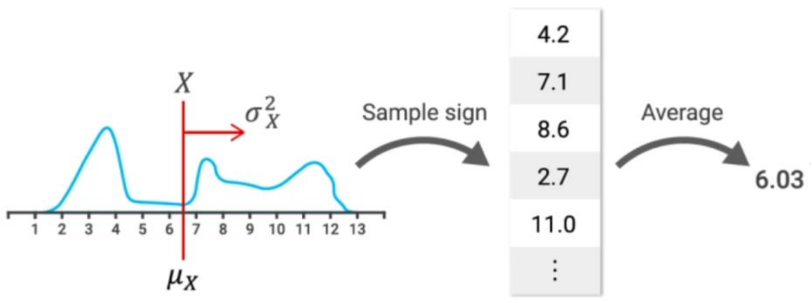
The formula for the Gaussian PDF has **two parameters**:

Mu (μ) is the mean of the distribution

Sigma squared (σ^2) is the variance

The mathematical concept that explains why the normal distribution is so common is called the **central limit theorem**.

Consider an experiment. Say you have a process that generates numbers according to some distribution, X. Assume you do not know the distribution of X. All you can see are the samples, the data. It is very common to want to estimate the **expected value** of X, μ_X. To do this, you collect n samples of X. You call this a sample of size n. This is your **dataset**.

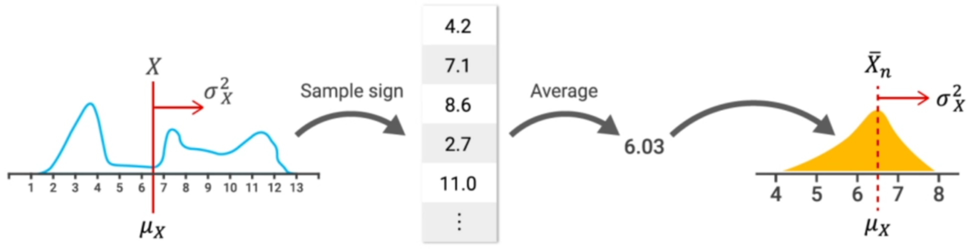


Consider one such sample, a list of numbers. A reasonable thing to do is to estimate μ_x by taking the **average** of these numbers. In this instance, say the average is 6.03. This is called the **sample mean**. But how good is 6.03 as an estimate of μ_x? To answer this, imagine you repeat this process many, many times.

Every time you do so, you:

- Obtain a **new sample mean**
- Create a **histogram** with all of these numbers

In fact, the mean of samples of size n, is itself a **random variable**: $\bar{X}_n$. The sample mean has its own expectation, $\mu_{\bar{X}_n}$, and variance, $\sigma^2_{\bar{X}_n}$.



There are two crucial facts:

- First, the expectation of the sample mean equals the expectation of X.

  $$\mu_{\bar{x}_n} = \mu_X$$

  Your sample mean may not exactly equal the true mean, so μ_X is probably not precisely 6.03, but the process of collecting samples and taking their mean produces the right result on average.

- Second, the variance of the sample mean decreases as $\frac{\sigma^2}{n}$.

  $$\sigma^2_{\bar{X}_n} = \frac{\sigma^2_X}{n}$$

  This implies that the larger the sample, the smaller $\sigma_{\bar{X}_n}$, and the tighter the sample means will be distributed around the true mean. Larger samples produce better estimates of μ_X.

Central limit theorem

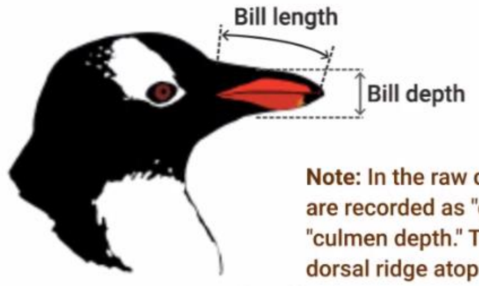
The central limit theorem is a tremendously consequential theorem for statistics and machine learning because it lets you **assume normality** in many situations. Whereas the two previous facts told you something about the mean and variance of $\bar{X}_n$, the central limit theorem tells you about its **shape**. It says that as n increases, $\bar{X}_n$ becomes closer and closer to a normal distribution.

The two facts and the central limit theorem apply regardless of the distribution of X. Even if X were a discrete distribution, it still holds true that its sample mean will approach a continuous normal distribution as n grows.

Multivariate Distributions

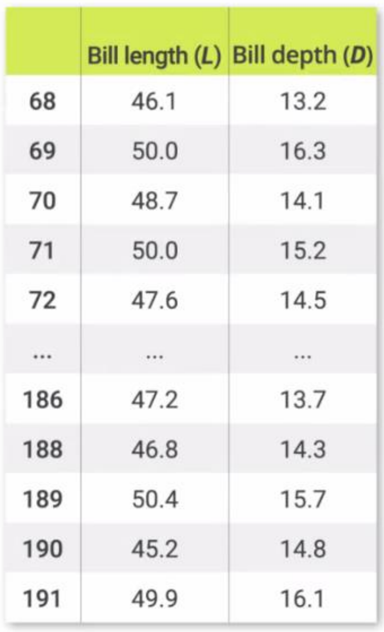
So far you have studied individual random variables, their distributions, and the central limit theorem, which helps to explain why the normal distribution is so prevalent in nature. However, the world is more complicated than just a bunch of isolated random variables. Most interesting phenomena cannot be understood as samples from a single distribution, but rather involve interactions between many elements.

Consider a study of Gentoo penguins that live in large numbers in the Falkland Islands and in parts of Antarctica. A study by Dr. Kristen Gorman measured the length and depth of the beaks of 123 Gentoos.



Note: In the raw data, bill dimensions are recorded as "culmen length" and "culmen depth." The "culmen" is the dorsal ridge atop the bill.

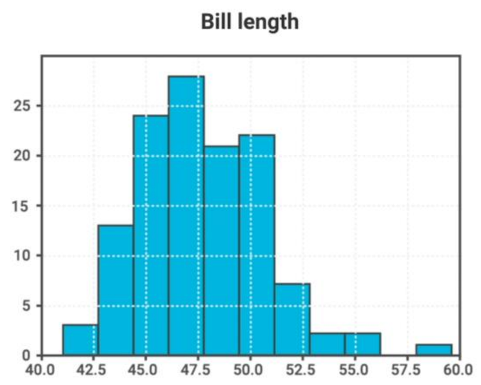
Artwork by @allison_horst"

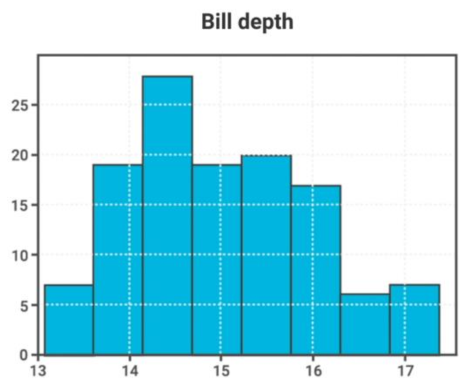


	Bill length (*L*)	Bill depth (*D*)
68	46.1	13.2
69	50.0	16.3
70	48.7	14.1
71	50.0	15.2
72	47.6	14.5
...	...	...
186	47.2	13.7
188	46.8	14.3
189	50.4	15.7
190	45.2	14.8
191	49.9	16.1

123 rows x 2 columns

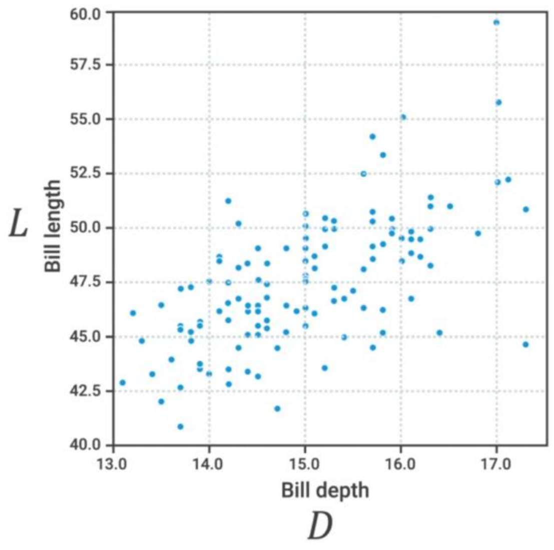
If you load this data into a **pandas dataframe** you can easily create histograms for the two columns, Bill length and Bill depth.





These histograms show that the mean bill length is around 47mm and the mean bill depth is around 15mm.

Now consider a scatterplot of this data:



Each point in the scatterplot is a single sample or row in the table, and its coordinates are the beak, or bill, depth and length for that penguin. In this view, you can recognize a trend: larger beak lengths tend to go with larger beak depths. There is a **correlation** in this data that was not visible when the variables were considered in isolation.

To preserve this relationship, you use a **multivariate random variable**:

$$X = \begin{bmatrix} D \\ L \end{bmatrix}$$

A multivariate random variable is a collection of random variables, in this case, capital D, the Bill depth, and capital L, the Bill length, arranged into a vector and endowed with a joint probability distribution:

$$f_X(d, l)$$

When sampling a multivariate random variable, you get a vector instead of a scalar. But all of the concepts for single random variables will extend easily to the multivariate case:

$X \rightarrow \begin{bmatrix} d \\ l \end{bmatrix}$

The concept of **expectation** is the same. The mean of a multivariate random variable is the vector of the means of the individual components. So, it is still the center of mass of the PDF. The expectation function is:

$E[X] = \begin{bmatrix} E[D] \\ E[L] \end{bmatrix}$

The law of **large numbers** still applies. The mean of a sample with size n will converge to the true mean as the sample size becomes large. The law of large numbers function is:

$\bar{\mu}_n \rightarrow E[X] as\ n \rightarrow \infty$

Keep in mind, however, that these are all now vector quantities and the central limit theorem remains unchanged. Its function is:

$$\bar{X}_n \rightarrow \mathcal{N}\left(\mu_X, \frac{\sigma_X^2}{n}\right) as\ n \rightarrow \infty$$

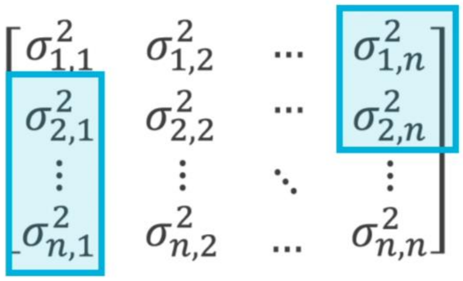
The concept of variance, however, changes a bit. Instead of a scalar quantity called variance, you have a matrix quantity: the covariance matrix.

Covariance matrix

To introduce the covariance matrix, consider a multivariate random variable, not with two entries, but with n entries, X_1 through X_n.

$$Covar[X] = E[(X - E[X])(X - E[X])^T] = \begin{bmatrix} \sigma_{1,1}^2 & \sigma_{1,2}^2 & \cdots & \sigma_{1,n}^2 \\ \sigma_{2,1}^2 & \sigma_{2,2}^2 & \cdots & \sigma_{2,n}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n,1}^2 & \sigma_{n,2}^2 & \cdots & \sigma_{n,n}^2 \end{bmatrix}$$

The covariance matrix is an n by n matrix, and it is defined in a way that is similar to the variance of a univariate random variable, by taking X minus the expectation of X, and squaring it. Except that now, since you are working with vectors, the result is a **square matrix**. The entries along the diagonal of the matrix are the variances of the individual random variables. So, for example, $\sigma_{1,1}^2$ is the variance of X_1. Recall that the variance is a measure of the spread of a distribution.



The off-diagonal entries are the covariance of pairs of random variables. So, $\sigma_{1,2}^2 = \sigma_{2,1}^2$ is the covariance of X_1 and X_2. The covariance of two random variables is a measure of their tendency to vary together. This is the information that you lose out on when considering the variables in isolation.

The covariances are related to the correlation between two quantities. The covariance between X_1 and X_2 is the same as the covariance between X_2 and X_1. So the covariance matrix has an axis of symmetry along its diagonal. Note: as you would use a lowercase sigma (σ) for the variance, a capital letter sigma (Σ) can be used as shorthand for the covariance matrix.

Gentoo dataset

Returning now to the Gentoo example, the dataset should be understood as a 123-size sample from a multivariate distribution of all Gentoo penguins, which remains hidden. The true (population) covariance function is:

$$\Sigma_X^2 = E[(X - E[X])(X - E[X])^T]$$

From the data, however, you can estimate the covariance matrix by computing the sample covariance, sigma-hat $(\hat{\Sigma})$. The formula for the sample covariance matrix is:

$$\hat{\Sigma}_X^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \hat{\mu})(x_i - \hat{\mu})^T$$

Note the similarity to the true covariance matrix.

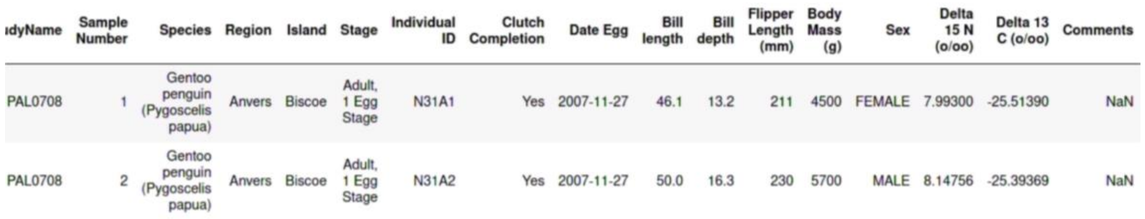
Covariance

The pandas package can be used to compute covariance and correlation matrices from data. This can be used in conjunction with the Seaborn package for plotting. Seaborn is built on Matplotlib.

First, import both packages. Note that the common alias for seaborn is **sns**:
import pandas as pd
import seaborn as sns

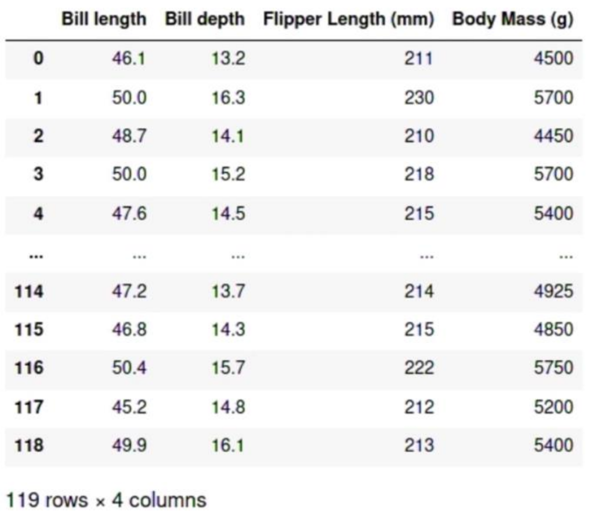
And then, you can load your data. In this instance the Gentoo dataframe is used:

gentoo = pd.read_csv('data/gentoo.csv')



ıdyName	Sample Number	Species	Region	Island	Stage	Individual ID	Clutch Completion	Date Egg	Bill length	Bill depth	Flipper Length (mm)	Body Mass (g)	Sex	Delta 15 N (o/oo)	Delta 13 C (o/oo)	Comments
PAL0708	1	Gentoo penguin (Pygoscelis papua)	Anvers	Biscoe	Adult, 1 Egg Stage	N31A1	Yes	2007-11-27	46.1	13.2	211	4500	FEMALE	7.99300	-25.51390	NaN
PAL0708	2	Gentoo penguin (Pygoscelis papua)	Anvers	Biscoe	Adult, 1 Egg Stage	N31A2	Yes	2007-11-27	50.0	16.3	230	5700	MALE	8.14756	-25.39369	NaN

The Gentoo dataframe has 17 columns – not all of this information is needed. For this example, you want to keep four columns: Bill length, Bill depth, Flipper Length, and Body Mass. So you can make strings of the names and assign that to override the Gentoo dataframe. This creates a smaller dataframe:



	Bill length	Bill depth	Flipper Length (mm)	Body Mass (g)
0	46.1	13.2	211	4500
1	50.0	16.3	230	5700
2	48.7	14.1	210	4450
3	50.0	15.2	218	5700
4	47.6	14.5	215	5400
...	...	...	...	...
114	47.2	13.7	214	4925
115	46.8	14.3	215	4850
116	50.4	15.7	222	5750
117	45.2	14.8	212	5200
118	49.9	16.1	213	5400

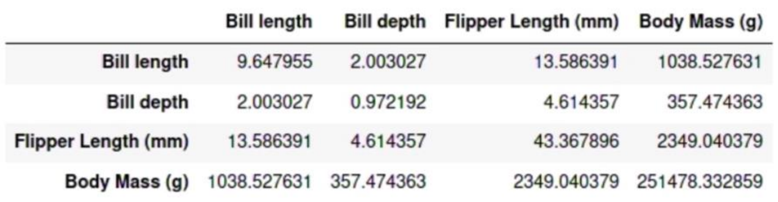
119 rows × 4 columns

Covariance matrix

To **compute a covariance matrix**, the formula is:

$$\hat{\Sigma}\,^2_X X^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{\mu})(x_i - \hat{\mu})^T$$

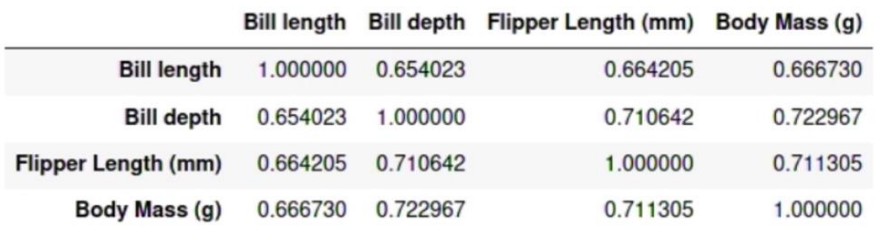
And you call **cov()** on it to compute the covariance matrix: **gentoo.cov()**.



	Bill length	Bill depth	Flipper Length (mm)	Body Mass (g)
Bill length	9.647955	2.003027	13.586391	1038.527631
Bill depth	2.003027	0.972192	4.614357	357.474363
Flipper Length (mm)	13.586391	4.614357	43.367896	2349.040379
Body Mass (g)	1038.527631	357.474363	2349.040379	251478.332859

Recall that the covariance matrix is symmetric. So it has an axis of symmetry about the diagonal. And this means that there is covariance between all of these, but the scales are quite different. Along the diagonal, you have the variances and even though the covariance between the Flipper Length and the Body Mass seems huge, it is apparent that the variance of Body Mass is really large. So, that is a little bit difficult to interpret.

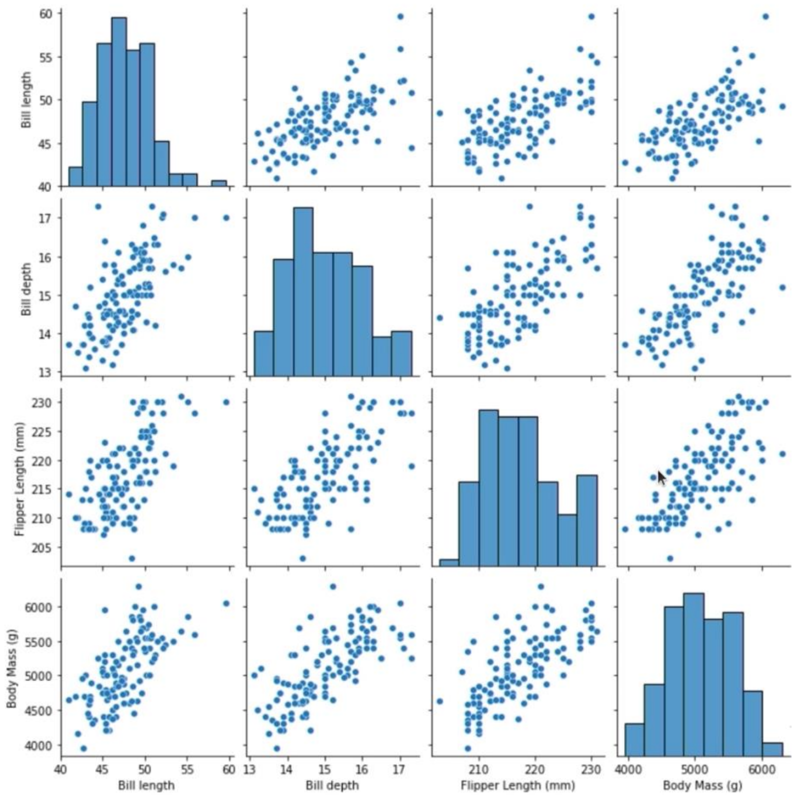
It is easier to interpret a **correlation matrix**, which is also very easy to produce in pandas with the **corr()** function: **gentoo.corr().**



	Bill length	Bill depth	Flipper Length (mm)	Body Mass (g)
Bill length	1.000000	0.654023	0.664205	0.666730
Bill depth	0.654023	1.000000	0.710642	0.722967
Flipper Length (mm)	0.664205	0.710642	1.000000	0.711305
Body Mass (g)	0.666730	0.722967	0.711305	1.000000

The correlation matrix, as you may recall, has ones along the diagonal because everything is perfectly correlated with itself. It is also symmetric. In this instance, there are somewhat strong correlations between all of the random variables. The strongest correlated are the Bill depth and the Body Mass with a correlation factor of 0.72.

Next, you can explore the **correlations matrix plot** that is provided by Seaborn. It is known as a **pair plot**. So you call **pairplot()** on the Gentoo dataframe: **sns.pairplot(gentoo)**. As a result, Seaborn produces this plot, which is essentially the same information as the correlations matrix:

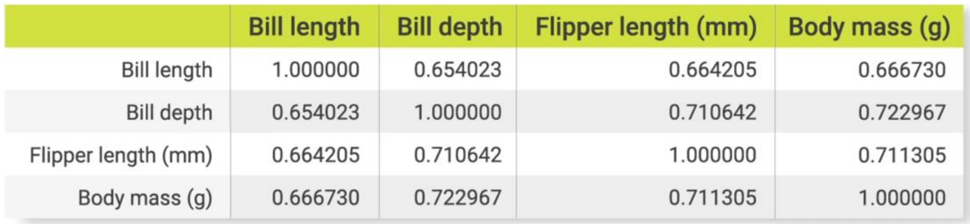


It is a **four-by-four matrix of plots** and each cell in the matrix is the scatterplot of the two corresponding variables. So, the correlations can be seen a little bit more directly. Along the diagonal it includes a histogram.

Correlation, Conditional Probabilities, and Independence

The entries in a correlation matrix are denoted by a **rho** (ρ), and they represent the correlations between pairs of variables.

Here is the correlation matrix for the Gentoo penguin data:



	Bill length	Bill depth	Flipper length (mm)	Body mass (g)
Bill length	1.000000	0.654023	0.664205	0.666730
Bill depth	0.654023	1.000000	0.710642	0.722967
Flipper length (mm)	0.664205	0.710642	1.000000	0.711305
Body mass (g)	0.666730	0.722967	0.711305	1.000000

This is a generic correlation matrix for a random variable of size n:

$$Corr[X] = \begin{bmatrix} 1 & \rho_{1,2} & \cdots & \rho_{1,n} \\ \rho_{2,1} & 1 & \cdots & \rho_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n,1} & \rho_{n,2} & \dots & 1 \end{bmatrix}$$

$$\rho_{i,j} = \frac{\sigma_{i,j}^2}{\sigma_i \sigma_j} \in [-1, 1]$$

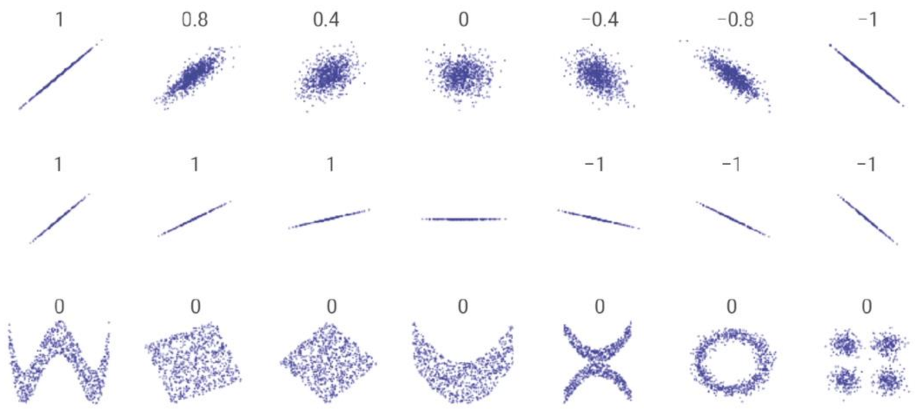
The diagonal entries are all 1, because a variable is always perfectly correlated with itself. The i_j entry in the correlation matrix is obtained by taking the i_j covariance and dividing it by the standard deviations of X_i and X_j. It can be shown that the result is between –1 and 1.

As with covariance, the correlation indicates a tendency of two variables to vary together:

- A **negative correlation** means that when one goes up, the other tends to go down. And when one goes down, the other tends to go up.
- A **positive correlation** means that they both go up and down together. And a zero correlation means that neither of these is true.

Data clouds and correlation coefficients

Consider this diagram of data clouds and their corresponding correlation coefficients:



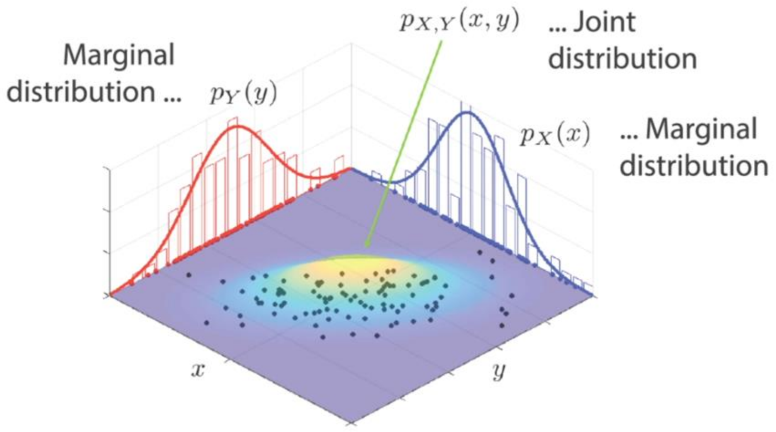
The **top row** is generated by Gaussian multivariates. In the middle plot of the top row, the two variables are uncorrelated. When the correlation is positive, the data cloud is sloped upward. And the closer to 1, the more the data coalesces into a line. Negative correlations indicate downward-sloping data clouds.

The **middle row** shows cases of perfect correlation, where p_{ij}equals 1 or –1. In the middle plot, the correlation coefficient is undefined, since one of the standard deviations is 0. These plots are making the point that the correlation is not capturing the slope of the data. All upward-sloping plots have correlation equal to 1, and all downward-sloping plots have correlation equal to –1.

The **bottom row** shows various examples of uncorrelated data. It is evident that the lack of correlation does not mean that the variables are not predictive of each other. On the bottom left, for example, the data seems to follow a sine wave. So, the value of x gives a good sense of the value of y. And yet, these two have zero correlation.

In all of these cases, there is no tendency of one variable to go up or down whenever the other goes up or down. These variables, although predictive of each other, do not obey a simple linear or monotonic relationship. They are **uncorrelated**, but not independent.

Conditional probabilities



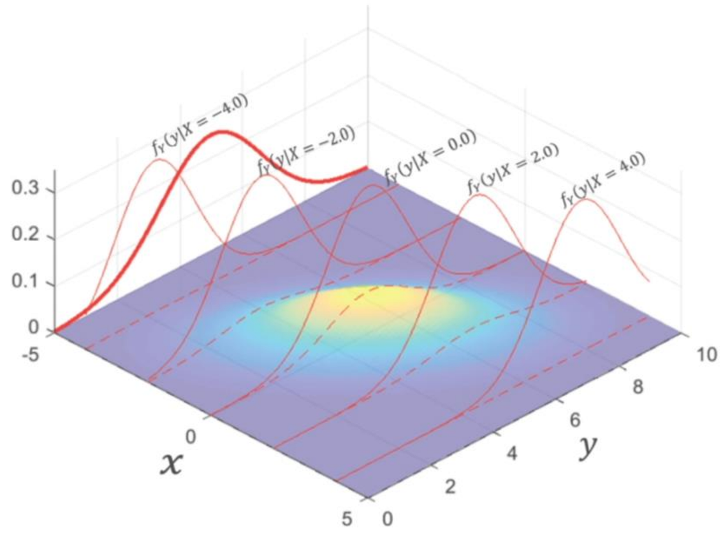
This graph depicts two random variables, x and y. These could be, for example, the length and depth of Gentoo beaks. The **joint distribution**, $p_{X,Y}$, is the green and yellow bump in the middle on the x, y plane. When collecting data, it is sampled from this distribution – as indicated by the cloud of black dots.

Consider that one of the variables in isolation, say y, is like projecting the data onto the y-axis – represented by the red dots.

It is possible to construct histograms using these, but the correlation among the variables is lost if this is done.

In the graph, the red and blue lines are the so-called marginal distributions of x and y. These are the distributions for the individual variables, x or y, if others are ignored. The marginal distributions would produce the blue dots if measuring only x, and the red dots if measuring only y.

The **conditional probability** allows you to find distributions in one random variable when the others are fixed at particular values. For example, what is the distribution of Gentoo beak depths amongst birds with beak lengths between 43 and 45mm? This amounts to taking a slice of the joint PDF along the specified value, and then scaling it up so that its integral is 1.

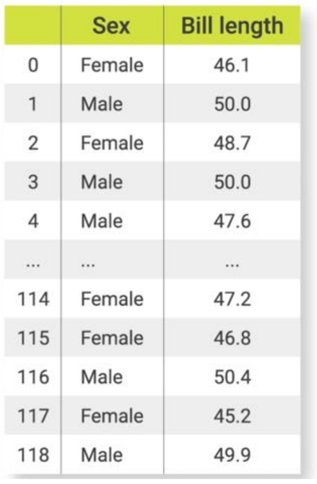


$$p_Y(y|X=x) = \frac{p_{X,Y}(x,y)}{p_X(x)}$$

The **notation for the conditional probability** involves a vertical bar. This is read as the conditional probability of y, given that the variable X has a value of lowercase x. This is a PDF over values of y. You compute it by dividing the slice of the joint PDF by the marginal distribution of X, evaluated at lowercase x.

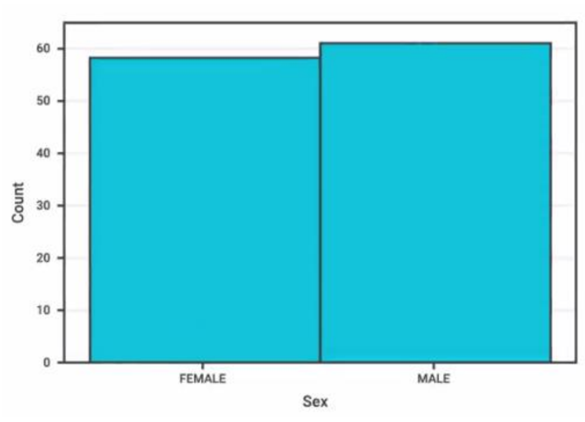
Beak length and sex

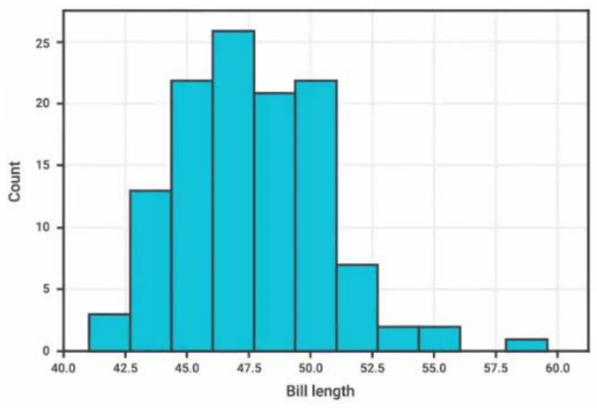
Consider an example to explore beak lengths and the sex of birds.



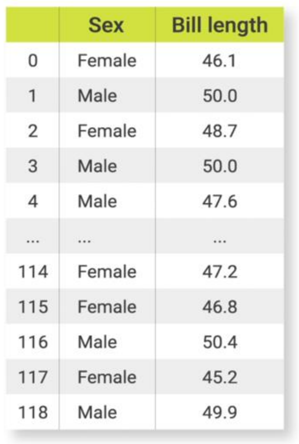
	Sex	Bill length
0	Female	46.1
1	Male	50.0
2	Female	48.7
3	Male	50.0
4	Male	47.6
...	...	...
114	Female	47.2
115	Female	46.8
116	Male	50.4
117	Female	45.2
118	Male	49.9

The multivariate variable can contain a **continuous quantity**, such as beak lengths, and a discrete quantity, such as the sex of penguins. A multivariate can contain any collection of random variables continuous and/or discrete. If you consider the histograms, there are about equal proportions of males and females in the dataset.



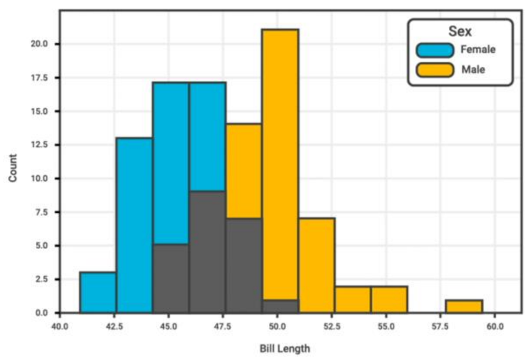


For beak lengths, consider this distribution. The conditional distributions of beak lengths for males are in orange, and females in blue:



	Sex	Bill length
0	Female	46.1
1	Male	50.0
2	Female	48.7
3	Male	50.0
4	Male	47.6
...	...	...
114	Female	47.2
115	Female	46.8
116	Male	50.4
117	Female	45.2
118	Male	49.9

```
sns.histplot(data=gentoo, x='Bill length', hue='Sex')
```



In Seaborn, you can create this easily by passing a parameter called **hue** to the **histplot()** method. This tells the program to split the data by sex and paint the various conditional distributions.

One thing to note here is that male Gentoos tend to have longer beaks than females. In other words, knowing the sex of the bird indicates its probable beak length. These are note correlated, because that concept only applies to numerical values, and you have not yet attached any numbers to the labels, male and female. But beak lengths and sex are dependent random variables.

Independence of variables

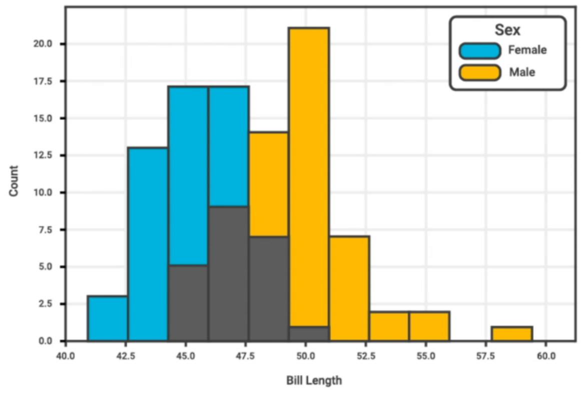
Now, consider the concept of dependence of random variables:

$$p_Y(y|X = x_1) = p_Y(y|X = x_2) = p_Y(y)$$

Two random variables are said to be independent when knowing the value of one tells you nothing about the value of the other. In terms of probability distributions, this means that the distribution of y, given X equals some value, x_1, is the same as the distribution conditioned on X being any other value, x_2. This is also equal to the marginal distribution of y.

For example, if the distribution of beak lengths for male Gentoos were identical to the distribution of beak lengths for females, and therefore also equal to the general distribution of beak lengths over the entire population – the marginal distribution – then the sex and beak lengths are independent quantities.

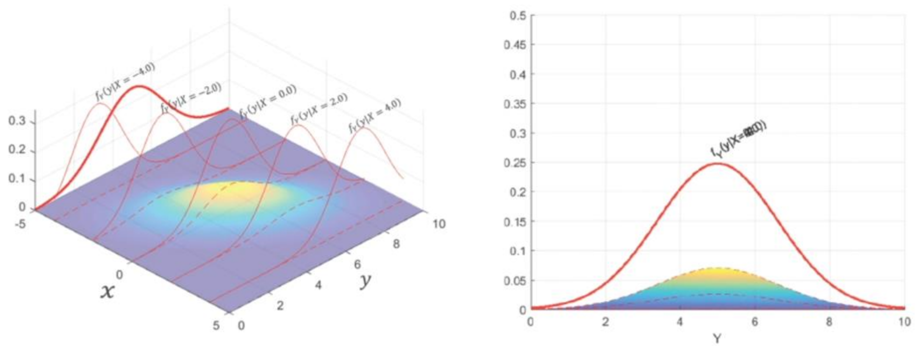
$$p_L(l|Sex = male) = p_L(l|Sex = female) = p_L(l)$$



Clearly this is not the case. The blue and orange distributions are not the same. And so, beak length is not independent of sex.

With two continuous variables, independence means that all conditional distributions are identical:

$$p_{X,Y}(x, y) = p_X(x)p_Y(y)$$



So all slices of the joint distribution parallel to the y-axis and scaled up to a conditional distribution are exactly the same. If you were to look at them head on, as in the second graph, they would all line up and also coincide with the marginal distribution of y, which is shown as the bold red line.

Independence is a very strong condition. If it holds, the joint distribution has a special form that is the product of the marginal distributions.