CHAPTER 1

Statistics and Probability

The purpose of this chapter is to instill in you the basic concepts of traditional statistics and probability. Certainly many of you might be wondering what it has to do with machine learning. Well, in order to apply a best fit model to your data, the most important prerequisite is for you to understand the data in the first place. This will enable you to find out distributions within data, measure the goodness of data, and run some basic tests to understand if some form of relationship exists between dependant and independent variables. Let's dive in.

■ **Note** This book incorporates **Python 2.7.11** as the de facto standard for coding examples. Moreover, you are required to have it installed it for the *Exercises* as well.

So why do I prefer Python 2.7.11 over Python 3x? Following are some of the reasons:

- Third-party library support for Python 2x is relatively better than support for Python 3x. This means that there are a considerable number of libraries in Python 2x that lack support in Python 3x.
- Some current Linux distributions and macOS provide Python 2x by default. The objective is to let readers, regardless of their OS version, apply the code examples on their systems, and thus this is the choice to go forward with.
- The above-mentioned facts are the reason why companies prefer to work with Python 2x or why they decide not to migrate their code base from Python 2x to Python 3x.

Case Study: Cycle Sharing Scheme—Determining Brand Persona

Nancy and Eric were assigned with the huge task of determining the brand persona for a new cycle share scheme. They had to present their results at this year's annual board meeting in order to lay out a strong marketing plan for reaching out to potential customers.

D. Haroon, *Python Machine Learning Case Studies*, DOI 10.1007/978-1-4842-2823-4_1

The cycle sharing scheme provides means for the people of the city to commute using a convenient, cheap, and green transportation alternative. The service has 500 bikes at 50 stations across Seattle. Each of the stations has a dock locking system (where all bikes are parked); kiosks (so customers can get a membership key or pay for a trip); and a helmet rental service. A person can choose between purchasing a membership key or short-term pass. A membership key entitles an annual membership, and the key can be obtained from a kiosk. Advantages for members include quick retrieval of bikes and unlimited 45-minute rentals. Short-term passes offer access to bikes for a 24-hour or 3-day time interval. Riders can avail and return the bikes at any of the 50 stations citywide.

Jason started this service in May 2014 and since then had been focusing on increasing the number of bikes as well as docking stations in order to increase convenience and accessibility for his customers. Despite this expansion, customer retention remained an issue. As Jason recalled, "We had planned to put in the investment for a year to lay out the infrastructure necessary for the customers to start using it. We had a strategy to make sure that the retention levels remain high to make this model self-sustainable. However, it worked otherwise (i.e., the customer base didn't catch up with the rate of the infrastructure expansion)."

A private service would have had three alternatives to curb this problem: get sponsors on board, increase service charges, or expand the pool of customers. Price hikes were not an option for Jason as this was a publicly sponsored initiative with the goal of providing affordable transportation to all. As for increasing the customer base, they had to decide upon a marketing channel that guarantees broad reach on low cost incurred.

Nancy, a marketer who had worked in the corporate sector for ten years, and Eric, a data analyst, were explicitly hired to find a way to make things work around this problem. The advantage on their side was that they were provided with the dataset of transaction history and thus they didn't had to go through the hassle of conducting marketing research to gather data.

Nancy realized that attracting recurring customers on a minimal budget required understanding the customers in the first place (i.e., persona). As she stated, "Understanding the persona of your brand is essential, as it helps you reach a targeted audience which is likely to convert at a higher probability. Moreover, this also helps in reaching out to sponsors who target a similar persona. This two-fold approach can make our bottom line positive."

As Nancy and Eric contemplated the problem at hand, they had questions like the following: Which attribute correlates the best with trip duration and number of trips? Which age generation adapts the most to our service?

Following is the data dictionary of the *Trips* dataset that was provided to Nancy and Eric:

Table 1-1. *Data Dictionary for the Trips Data from Cycles Share Dataset*

Feature name	Description
trip_id	Unique ID assigned to each trip
Starttime	Day and time when the trip started, in PST
Stoptime	Day and time when the trip ended, in PST
Bikeid	ID attached to each bike
Tripduration	Time of trip in seconds
from_station_name	Name of station where the trip originated
to_station_name	Name of station where the trip terminated
from_station_id	ID of station where trip originated
to_station_id	ID of station where trip terminated
Usertype	Value can include either of the following: short-term pass holder or member
Gender	Gender of the rider
Birthyear	Birth year of the rider

Exercises for this chapter required Eric to install the packages shown in Listing 1-1. He preferred to import all of them upfront to avoid bottlenecks while implementing the code snippets on your local machine.

However, for Eric to import these packages in his code, he needed to install them in the first place. He did so as follows:

1. Opened terminal/shell
2. Navigated to his code directory using terminal/shell
3. Installed `pip`:

   ```
   python get-pip.py
   ```

4. Installed each package separately, for example:

   ```
   pip install pandas
   ```

Listing 1-1. Importing Packages Required for This Chapter

```
%matplotlib inline

import random
import datetime
import pandas as pd
import matplotlib.pyplot as plt
import statistics
```



```
import numpy as np
import scipy
from scipy import stats
import seaborn
```

Performing Exploratory Data Analysis

Eric recalled to have explained Exploratory Data Analysis in the following words:

> *What do I mean by exploratory data analysis (EDA)? Well, by this I mean to see the data visually. Why do we need to see the data visually? Well, considering that you have 1 million observations in your dataset then it won't be easy for you to understand the data just by looking at it, so it would be better to plot it visually. But don't you think it's a waste of time? No not at all, because understanding the data lets us understand the importance of features and their limitations.*

Feature Exploration

Eric started off by loading the data into memory (see Listing 1-2).

Listing 1-2. Reading the Data into Memory

```
data = pd.read_csv('examples/trip.csv')
```

Nancy was curious to know how big the data was and what it looked like. Hence, Eric wrote the code in Listing 1-3 to print some initial observations of the dataset to get a feel of what it contains.

Listing 1-3. Printing Size of the Dataset and Printing First Few Rows

```
print len(data)
data.head()
```

Output

```
236065
```

Table 1-2. *Print of Observations in the First Seven Columns of Dataset*

trip_id	starttime	stoptime	bikeid	tripduration	from_station_name	to_station_name
431	10/13/2014 10:31	10/13/2014 10:48	SEA00298	985.935	2nd Ave & Spring St	Occidental Park/ Occidental Ave S & S Washing...
432	10/13/2014 10:32	10/13/2014 10:48	SEA00195	926.375	2nd Ave & Spring St	Occidental Park/ Occidental Ave S & S Washing...
433	10/13/2014 10:33	10/13/2014 10:48	SEA00486	883.831	2nd Ave & Spring St	Occidental Park/ Occidental Ave S & S Washing...
434	10/13/2014 10:34	10/13/2014 10:48	SEA00333	865.937	2nd Ave & Spring St	Occidental Park/ Occidental Ave S & S Washing...
435	10/13/2014 10:34	10/13/2014 10:49	SEA00202	923.923	2nd Ave & Spring St	Occidental Park/ Occidental Ave S & S Washing...

Table 1-3. *Print of Observations in the Last five Columns of Dataset*

from_station_id	to_station_id	usertype	gender	birthyear
CBD-06	PS-04	Member	Male	1960.0
CBD-06	PS-04	Member	Male	1970.0
CBD-06	PS-04	Member	Female	1988.0
CBD-06	PS-04	Member	Female	1977.0
CBD-06	PS-04	Member	Male	1971.0

After looking at Table 1-2 and Table 1-3 Nancy noticed that `tripduration` is represented in seconds. Moreover, the unique identifiers for `bike`, `from_station`, and `to_station` are in the form of strings, contrary to those for `trip identifier` which are in the form of integers.

Types of variables

Nancy decided to go an extra mile and allocated data type to each feature in the dataset.

Table 1-4. *Nancy's Approach to Classifying Variables into Data Types*

Feature name	Variable type

`trip_id`
`bikeid`
`tripduration`
`from_station_id`
`to_station_id`
`birthyear`
	Numbers

`Starttime` `Stoptime`	Date

`from_station_name to_station_name`
`Usertype`
`Gender`
	Text

After looking at the feature classification in Table 1-4 Eric noticed that Nancy had correctly identified the data types and thus it seemed to be an easy job for him to explain what variable types mean. As Eric recalled to have explained the following:

> *In normal everyday interaction with data we usually represent numbers as integers, text as strings, True/False as Boolean, etc. These are what we refer to as data types. But the lingo in machine learning is a bit more granular, as it splits the data types we knew earlier into variable types. Understanding these variable types is crucial in deciding upon the type of charts while doing exploratory data analysis or while deciding upon a suitable machine learning algorithm to be applied on our data.*

Continuous/Quantitative Variables

A continuous variable can have an infinite number of values within a given range. Unlike discrete variables, they are not countable. Before exploring the types of continuous variables, let's understand what is meant by a true zero point.

True Zero Point

If a level of measurement has a true zero point, then a value of 0 means you have nothing. Take, for example, a ratio variable which represents the number of cupcakes bought. A value of 0 will signify that you didn't buy even a single cupcake. The true zero point is a strong discriminator between interval and ratio variables.

Let's now explore the different types of continuous variables.

Interval Variables

Interval variables exist around data which is continuous in nature and has a numerical value. Take, for example, the temperature of a neighborhood measured on a daily basis. Difference between intervals remains constant, such that the difference between 70 Celsius and 50 Celsius is the same as the difference between 80 Celsius and 100 Celsius. We can compute the mean and median of interval variables however they don't have a true zero point.

Ratio Variables

Properties of interval variables are very similar to those of ratio variables with the difference that in ratio variables a 0 indicates the absence of that measurement. Take, for example, distance covered by cars from a certain neighborhood. Temperature in Celsius is an interval variable, so having a value of 0 Celsius does not mean absence of temperature. However, notice that a value of 0 KM will depict no distance covered by the car and thus is considered as a ratio variable. Moreover, as evident from the name, ratios of measurements can be used as well such that a distance covered of 50 KM is twice the distance of 25 KM covered by a car.

Discrete Variables

A discrete variable will have finite set of values within a given range. Unlike continuous variables those are countable. Let's look at some examples of discrete variables which are categorical in nature.

Ordinal Variables

Ordinal variables have values that are in an order from lowest to highest or vice versa. These levels within ordinal variables can have unequal spacing between them. Take, for example, the following levels:

1. Primary school
2. High school
3. College
4. University

The difference between primary school and high school in years is definitely not equal to the difference between high school and college. If these differences were constant, then this variable would have also qualified as an interval variable.

Nominal Variables

Nominal variables are categorical variables with no intrinsic order; however, constant differences between the levels exist. Examples of nominal variables can be gender, month of the year, cars released by a manufacturer, and so on. In the case of month of year, each month is a different level.

Dichotomous Variables

Dichotomous variables are nominal variables which have only two categories or levels. Examples include

- Age: under 24 years, above 24 years
- Gender: male, female

Lurking Variable

A lurking variable is not among exploratory (i.e., independent) or response (i.e., dependent) variables and yet may influence the interpretations of relationship among these variables. For example, if we want to predict whether or not an applicant will get admission in a college on the basis of his/her gender. A possible lurking variable in this case can be the name of the department the applicant is seeking admission to.

Demographic Variable

Demography (from the Greek word meaning "description of people") is the study of human populations. The discipline examines size and composition of populations as well as the movement of people from locale to locale. Demographers also analyze the effects of population growth and its control. A demographic variable is a variable that is collected by researchers to describe the nature and distribution of the sample used with inferential statistics. Within applied statistics and research, these are variables such as age, gender, ethnicity, socioeconomic measures, and group membership.

Dependent and Independent Variables

An independent variable is also referred to as an exploratory variable because it is being used to explain or predict the dependent variable, also referred to as a response variable or outcome variable.

Taking the dataset into consideration, what are the dependent and independent variables? Let's say that Cycle Share System's management approaches you and asks you to build a system for them to predict the trip duration beforehand so that the supply

of cycles can be ensured. In that case, what is your dependent variable? Definitely tripduration. And what are the independent variables? Well, these variables will comprise of the features which we believe influence the dependent variable (e.g., usertype, gender, and time and date of the day).

Eric asked Nancy to classify the features in the variable types he had just explained.

Table 1-5. *Nancy's Approach to Classifying Variables into Variable Types*

Feature name	Variable type

`trip_id`
`bikeid`
`tripduration`
`from_station_id`
`to_station_id`
`birthyear`
	Continuous

`Starttime` `Stoptime`	DateTime
`from_station_name` `to_station_name`	String
`Usertype gender`	Nominal

Nancy now had a clear idea of the variable types within machine learning, and also which of the features qualify for which of those variable types (see Table 1-5). However despite of looking at the initial observations of each of these features (see Table 1-2) she couldn't deduce the depth and breadth of information that each of those tables contains. She mentioned this to Eric, and Eric, being a data analytics guru, had an answer: perform univariate analysis on features within the dataset.

Univariate Analysis

Univariate comes from the word "uni" meaning one. This is the analysis performed on a single variable and thus does not account for any sort of relationship among exploratory variables.

Eric decided to perform univariate analysis on the dataset to better understand the features in isolation (see Listing 1-4).

Listing 1-4. Determining the Time Range of the Dataset

```
data = data.sort_values(by='starttime')
data.reset_index()
print 'Date range of dataset: %s - %s'%(data.ix[1, 'starttime'],
data.ix[len(data)-1, 'stoptime'])
```

Output

```
Date range of dataset: 10/13/2014 10:32 - 9/1/2016 0:20
```


Eric knew that Nancy would have a hard time understanding the code so he decided to explain the ones that he felt were complex in nature. In regard to the code in Listing 1-4, Eric explained the following:

> *We started off by sorting the data frame by* `starttime`*. Do note that data frame is a data structure in Python in which we initially loaded the data in Listing 1-2. Data frame helps arrange the data in a tabular form and enables quick searching by means of hash values. Moreover, data frame comes up with handy functions that make lives easier when doing analysis on data. So what sorting did was to change the position of records within the data frame, and hence the change in positions disturbed the arrangement of the indexes which were earlier in an ascending order. Hence, considering this, we decided to reset the indexes so that the ordered data frame now has indexes in an ascending order. Finally, we printed the date range that started from the first value of* `starttime` *and ended with the last value of* `stoptime`*.*

Eric's analysis presented two insights. One is that the data ranges from October 2014 up till September 2016 (i.e., three years of data). Moreover, it seems like the cycle sharing service is usually operational beyond the standard 9 to 5 business hours.

Nancy believed that short-term pass holders would avail more trips than their counterparts. She believed that most people would use the service on a daily basis rather than purchasing the long term membership. Eric thought otherwise; he believed that new users would be short-term pass holders however once they try out the service and become satisfied would ultimately avail the membership to receive the perks and benefits offered. He also believed that people tend to give more weight to services they have paid for, and they make sure to get the maximum out of each buck spent. Thus, Eric decided to plot a bar graph of trip frequencies by user type to validate his viewpoint (see Listing 1-5). But before doing so he made a brief document of the commonly used charts and situations for which they are a best fit to (see Appendix A for a copy). This chart gave Nancy his perspective for choosing a bar graph for the current situation.

Listing 1-5. Plotting the Distribution of User Types

```
groupby_user = data.groupby('usertype').size()
groupby_user.plot.bar(title = 'Distribution of user types')
```

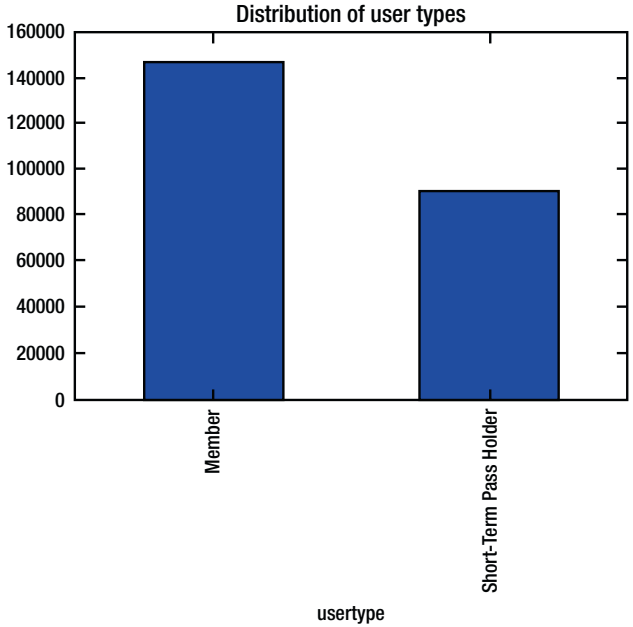

Figure 1-1. *Bar graph signifying the distribution of user types*

Nancy didn't understand the code snippet in Listing 1-5. She was confused by the functionality of groupby and size methods. She recalled asking Eric the following: "I can understand that groupby groups the data by a given field, that is, usertype, in the current situation. But what do we mean by size? Is it the same as count, that is, counts trips falling within each of the grouped usertypes?"

Eric was surprised by Nancy's deductions and he deemed them to be correct. However, the bar graph presented insights (see Figure 1-1) in favor of Eric's view as the members tend to avail more trips than their counterparts.

Nancy had recently read an article that talked about the gender gap among people who prefer riding bicycles. The article mentioned a cycle sharing scheme in UK where 77% of the people who availed the service were men. She wasn't sure if similar phenomenon exists for people using the service in United States. Hence Eric came up with the code snippet in Listing 1-6 to answer the question at hand.

Listing 1-6. Plotting the Distribution of Gender

```
groupby_gender = data.groupby('gender').size()
groupby_gender.plot.bar(title = 'Distribution of genders')
```


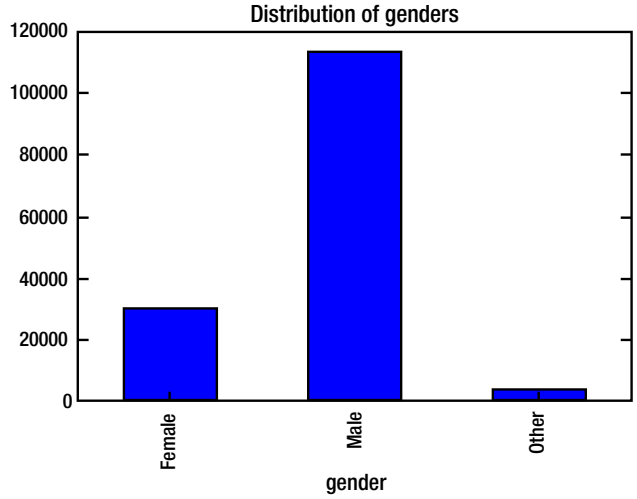

Figure 1-2. *Bar graph signifying the distribution of genders*

Figure 1-2 revealed that the gender gap resonates in states as well. Males seem to dominate the trips taken as part of the program.

Nancy, being a marketing guru, was content with the analysis done so far. However she wanted to know more about her target customers to whom to company's marketing message will be targetted to. Thus Eric decided to come up with the distribution of birth years by writing the code in Listing 1-7. He believed this would help the Nancy understand the age groups that are most likely to ride a cycle or the ones that are more prone to avail the service.

Listing 1-7. Plotting the Distribution of Birth Years

```
data = data.sort_values(by='birthyear')
groupby_birthyear = data.groupby('birthyear').size()
groupby_birthyear.plot.bar(title = 'Distribution of birth years',
figsize = (15,4))
```

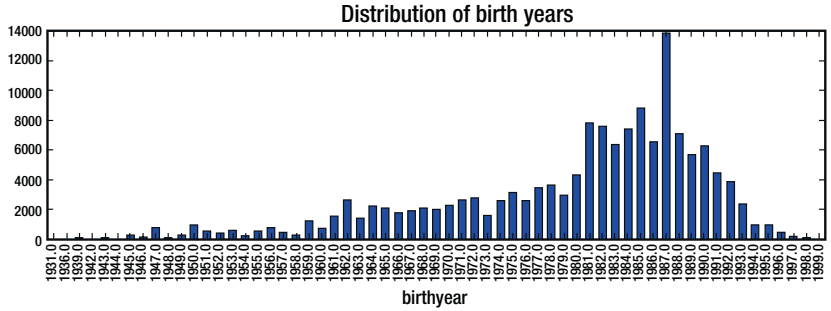

Figure 1-3. *Bar graph signifying the distribution of birth years*

Figure 1-3 provided a very interesting illustration. Majority of the people who had subscribed to this program belong to Generation Y (i.e., born in the early 1980s to mid to late 1990s, also known as millennials). Nancy had recently read the reports published by *Elite Daily* and *CrowdTwist* which said that millennials are the most loyal generation to their favorite brands. One reason for this is their willingness to share thoughts and opinions on products/services. These opinions thus form a huge corpus of experiences—enough information for the millenials to make a conscious decision, a decision they will remain loyal to for a long period. Hence Nancy was convinced that most millennials would be members rather than short-term pass holders. Eric decided to populate a bar graph to see if Nancy's deduction holds true.

Listing 1-8. Plotting the Frequency of Member Types for Millenials

```
data_mil = data[(data['birthyear'] >= 1977) & (data['birthyear']<=1994)]
groupby_mil = data_mil.groupby('usertype').size()
groupby_mil.plot.bar(title = 'Distribution of user types')
```

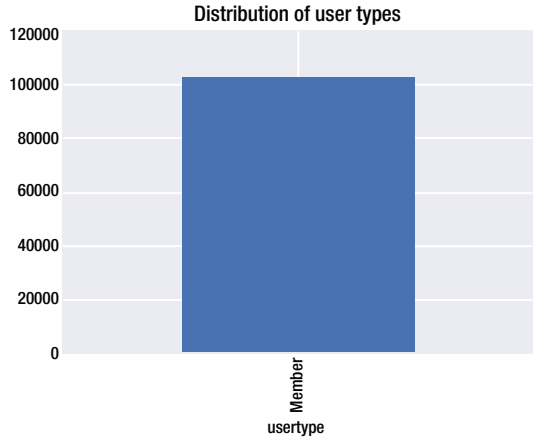

Figure 1-4. *Bar graph of member types for millenials*

After looking at Figure 1-4 Eric was surprised to see that Nancy's deduction appeared to be valid, and Nancy made a note to make sure that the brand engaged millennials as part of the marketing plan.

Eric knew that more insights can pop up when more than one feature is used as part of the analysis. Hence, he decided to give Nancy a sneak peek at multivariate analysis before moving forward with more insights.

Multivariate Analysis

Multivariate analysis refers to incorporation of multiple exploratory variables to understand the behavior of a response variable. This seems to be the most feasible and realistic approach considering the fact that entities within this world are usually interconnected. Thus the variability in response variable might be affected by the variability in the interconnected exploratory variables.

Nancy believed males would dominate females in terms of the trips completed. The graph in Figure 1-2, which showed that males had completed far more trips than any other gender types, made her embrace this viewpoint. Eric thought that the best approach to validate this viewpoint was a stacked bar graph (i.e., a bar graph for birth year, but each bar having two colors, one for each gender) (see Figure 1-5).

Listing 1-9. Plotting the Distribution of Birth Years by Gender Type

```
groupby_birthyear_gender = data.groupby(['birthyear', 'gender'])
['birthyear'].count().unstack('gender').fillna(0)
groupby_birthyear_gender[['Male','Female','Other']].plot.bar(title =
'Distribution of birth years by Gender', stacked=True, figsize = (15,4))
```

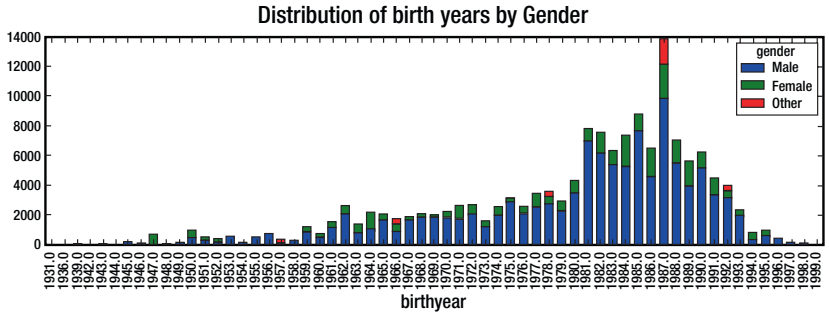

Figure 1-5. *Bar graph signifying the distribution of birth years by gender type*

The code snippet in Listing 1-9 brought up some new aspects not previously highlighted.

> *We at first transformed the data frame by unstacking, that is, splitting, the gender column into three columns, that is, Male, Female, and Other. This meant that for each of the birth years we had the trip count for all three gender types. Finally, a stacked bar graph was created by using this transformed data frame.*

It seemed as if males were dominating the distribution. It made sense as well. No? Well, it did; as seen earlier, that majority of the trips were availed by males, hence this skewed the distribution in favor of males. However, subscribers born in 1947 were all females. Moreover, those born in 1964 and 1994 were dominated by females as well. Thus Nancy's hypothesis and reasoning did hold true.

The analysis in Listing 1-4 had revealed that all millennials are members. Nancy was curious to see what the distribution of user type was for the other age generations. Is it that the majority of people in the other age generations were short-term pass holders? Hence Eric brought a stacked bar graph into the application yet again (see Figure 1-6).

Listing 1-10. Plotting the Distribution of Birth Years by User Types

```
groupby_birthyear_user = data.groupby(['birthyear', 'usertype'])
['birthyear'].count().unstack('usertype').fillna(0)

groupby_birthyear_user['Member'].plot.bar(title = 'Distribution of birth
years by Usertype', stacked=True, figsize = (15,4))
```

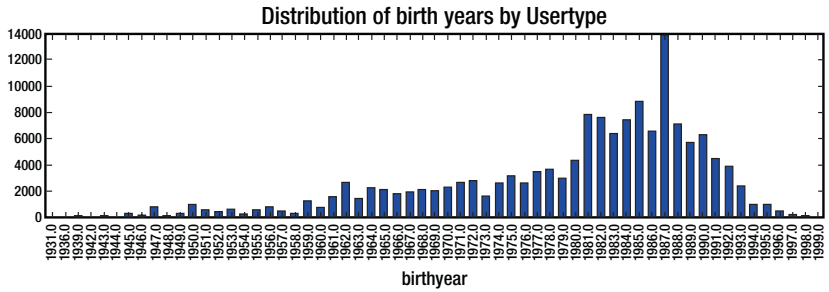

Figure 1-6. *Bar graph signifying the distribution of birth years by user types*

Whoa! Nancy was surprised to see the distribution of only one user type and not two (i.e., membership and short-term pass holders)? Does this mean that birth year information was only present for only one user type? Eric decided to dig in further and validate this (see Listing 1-11).

Listing 1-11. Validation If We Don't Have Birth Year Available for Short-Term Pass Holders

```
data[data['usertype']=='Short-Term Pass Holder']['birthyear'].isnull().
values.all()
```

Output

```
True
```

In the code in Listing 1-11, Eric first sliced the data frame to consider only short-term pass holders. Then he went forward to find out if all the values in birth year are missing (i.e., null) for this slice. Since that is the case, Nancy's initially inferred hypothesis was true—that birth year data is only available for members. This made her recall her prior deduction about the brand loyalty of millennials. Hence the output for Listing 1-11 nullifies Nancy's deduction made after the analysis in Figure 1-4. This made Nancy sad, as the loyalty of millenials can't be validated from the data at hand. Eric believed that members have to provide details like birth year when applying for the membership, something which is not a prerequisite for short-term pass holders. Eric decided to test his deduction by checking if gender is available for short-term pass holders or not for which he wrote the code in Listing 1-12.

Listing 1-12. Validation If We Don't Have Gender Available for Short-Term Pass Holders

```
data[data['usertype']=='Short-Term Pass Holder']['gender'].isnull().values.
all()
```

Output

```
True
```

Thus Eric concluded that we don't have the demographic variables for user type 'Short-Term Pass holders.'

Nancy was interested to see as to how the frequency of trips vary across date and time (i.e., a time series analysis). Eric was aware that trip start time is given with the data, but for him to make a time series plot, he had to transform the date from string to date time format (see Listing 1-13). He also decided to do more: that is, split the datetime into date components (i.e., year, month, day, and hour).

Listing 1-13. Converting String to datetime, and Deriving New Features

```
List_ = list(data['starttime'])

List_ = [datetime.datetime.strptime(x, "%m/%d/%Y %H:%M") for x in List_]
data['starttime_mod'] = pd.Series(List_,index=data.index)
data['starttime_date'] = pd.Series([x.date() for x in List_],index=data.index)
data['starttime_year'] = pd.Series([x.year for x in List_],index=data.index)
data['starttime_month'] = pd.Series([x.month for x in List_],index=data.index)
data['starttime_day'] = pd.Series([x.day for x in List_],index=data.index)
data['starttime_hour'] = pd.Series([x.hour for x in List_],index=data.index)
```

Eric made sure to explain the piece of code in Listing 1-13 as he had explained to Nancy:

> *At first we converted start time column of the dataframe into a list. Next we converted the string dates into python datetime objects. We then converted the list into a series object and converted the dates from datetime object to pandas date object. The time components of year, month, day and hour were derived from the list with the datetime objects.*

Now it was time for the time series analysis of the frequency of trips over all days provided within the dataset (see Listing 1-14).

Listing 1-14. Plotting the Distribution of Trip Duration over Daily Time

```
data.groupby('starttime_date')['tripduration'].mean().plot.bar(title =
'Distribution of Trip duration by date', figsize = (15,4))
```

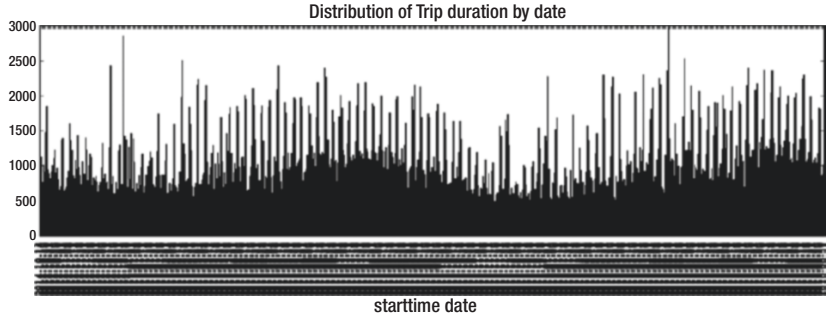

Figure 1-7. *Bar graph signifying the distribution of trip duration over daily time*

Wow! There seems to be a definitive pattern of trip duration over time.

Time Series Components

Eric decided to brief Nancy about the types of patterns that exist in a time series analysis. This he believed would help Nancy understand the definite pattern in Figure 1-7.

Seasonal Pattern

A seasonal pattern (see Figure 1-8) refers to a seasonality effect that incurs after a fixed known period. This period can be week of the month, week of the year, month of the year, quarter of the year, and so on. This is the reason why seasonal time series are also referred to as periodic time series.

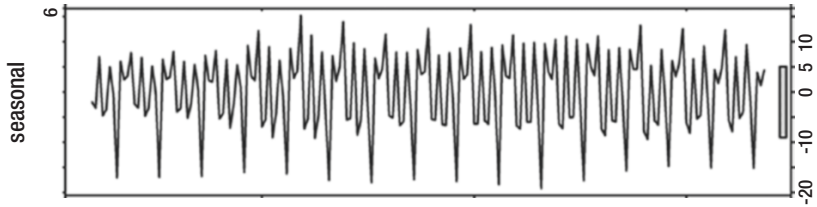

Figure 1-8. *Illustration of seasonal pattern*

Cyclic Pattern

A cyclic pattern (see Figure 1-9) is different from a seasonal pattern in the notion that the patterns repeat over non-periodic time cycles.

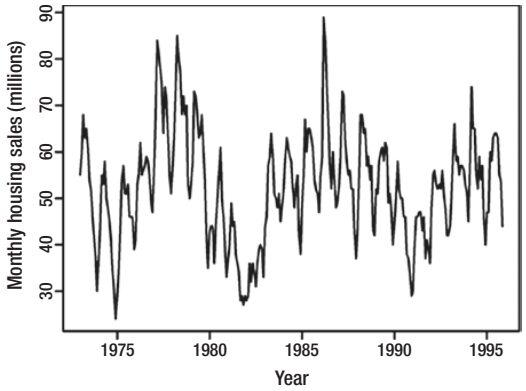

Figure 1-9. *Illustration of cyclic pattern*

Trend

A trend (see Figure 1-10) is a long-term increase or decrease in a continuous variable. This pattern might not be exactly linear over time, but when smoothing is applied it can generalize into either of the directions.

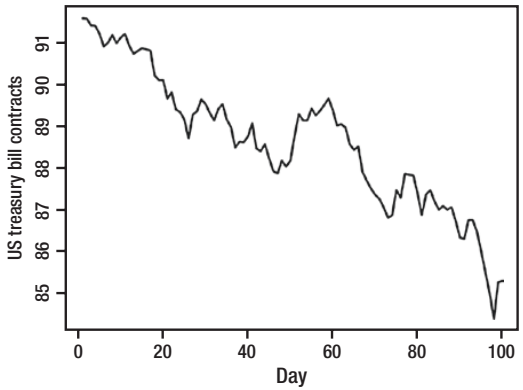

Figure 1-10. *Illustration of trend*

Eric decided to test Nancy's concepts on time series, so he asked her to provide her thoughts on the time series plot in Figure 1-7. "What do you think of the time series plot? Is the pattern seasonal or cyclic? Seasonal is it right?"

Nancy's reply amazed Eric once again. She said the following:

> *Yes it is because the pattern is repeating over a fixed interval of time—that is, seasonality. In fact, we can split the distribution into three distributions. One pattern is the seasonality that is repeating over time. The second one is a flat density distribution. Finally, the last pattern is the lines (that is, the hikes) over that density function. In case of time series prediction we can make estimations for a future time using both of these distributions and add up in order to predict upon a calculated confidence interval.*

On the basis of her deduction it seemed like Nancy's grades in her statistics elective course had paid off. Nancy wanted answers to many more of her questions. Hence she decided to challenge the readers with the Exercises that follow.