3
Alternative Data for Finance – Categories and Use Cases

The previous chapter covered working with market and fundamental data, which have been the traditional drivers of trading strategies. In this chapter, we'll fast-forward to the recent emergence of a broad range of much more diverse data sources as fuel for discretionary and algorithmic strategies. Their heterogeneity and novelty have inspired the label of alternative data and created a rapidly growing provider and service industry.

Behind this trend is a familiar story: propelled by the explosive growth of the internet and mobile networks, digital data continues to grow exponentially amid advances in the technology to process, store, and analyze new data sources. The exponential growth in the availability of and ability to manage more diverse digital data, in turn, has been a critical force behind the dramatic performance improvements of **machine learning** (**ML**) that are driving innovation across industries, including the investment industry.

The scale of the data revolution is extraordinary: the past 2 years alone have witnessed the creation of 90 percent of all data that exists in the world today, and by 2020, each of the 7.7 billion people worldwide is expected to produce 1.7 MB of new information every second of every day. On the other hand, back in 2012, only 0.5 percent of all data was ever analyzed and used, whereas 33 percent is deemed to have value by 2020. The gap between data availability and usage is likely to narrow quickly as global investments in analytics are set to rise beyond $210 billion by 2020, while the value creation potential is a multiple higher.

This chapter explains how individuals, business processes, and sensors produce **alternative data**. It also provides a framework to navigate and evaluate the proliferating supply of alternative data for investment purposes. It demonstrates the workflow, from acquisition to preprocessing and storage, using Python for data obtained through web scraping to set the stage for the application of ML. It concludes by providing examples of sources, providers, and applications.

This chapter will cover the following topics:

- Which new sources of information have been unleashed by the alternative data revolution
- How individuals, business processes, and sensors generate alternative data
- Evaluating the burgeoning supply of alternative data used for algorithmic trading

- Working with alternative data in Python, such as by scraping the internet
- Important categories and providers of alternative data

> You can find the code samples for this chapter and links to additional resources in the corresponding directory of the GitHub repository. The notebooks include color versions of the images.

The alternative data revolution

The data deluge driven by digitization, networking, and plummeting storage costs has led to profound qualitative changes in the nature of information available for predictive analytics, often summarized by the five Vs:

- **Volume**: The amount of data generated, collected, and stored is orders of magnitude larger as the byproduct of online and offline activity, transactions, records, and other sources. Volumes continue to grow with the capacity for analysis and storage.
- **Velocity**: Data is generated, transferred, and processed to become available near, or at, real-time speed.
- **Variety**: Data is organized in formats no longer limited to structured, tabular forms, such as CSV files or relational database tables. Instead, new sources produce semi-structured formats, such as JSON or HTML, and unstructured content, including raw text, "images"? and audio or video data, adding new challenges to render data suitable for ML algorithms.
- **Veracity**: The diversity of sources and formats makes it much more difficult to validate the reliability of the data's information content.
- **Value**: Determining the value of new datasets can be much more time- and resource-consuming, as well as more uncertain than before.

For algorithmic trading, new data sources offer an informational advantage if they provide access to information unavailable from traditional sources or provide access sooner. Following global trends, the investment industry is rapidly expanding beyond market and fundamental data to alternative sources to reap alpha through an informational edge. Annual spending on data, technological capabilities, and related talent is expected to increase from the current $3 billion by 12.8 percent annually through 2020.

Today, investors can access macro or company-specific data in real time that, historically, has been available only at a much lower frequency. Use cases for new data sources include the following:

- **Online price data** on a representative set of goods and services can be used to measure inflation.
- The number of **store visits or purchases** permits real-time estimates of company - or industry-specific sales or economic activity.
- **Satellite images** can reveal agricultural yields, or activity at mines or on oil rigs before this information is available elsewhere.

As the standardization and adoption of big datasets advances, the information contained in conventional data will likely lose most of its predictive value.

Furthermore, the capability to process and integrate diverse datasets and apply ML allows for complex insights. In the past, quantitative approaches relied on simple heuristics to rank companies using historical data for metrics such as the price-to-book ratio, whereas ML algorithms synthesize new metrics and learn and adapt such rules while taking into account evolving market data. These insights create new opportunities to capture classic investment themes such as value, momentum, quality, and sentiment:

- **Momentum**: ML can identify asset exposures to market price movements, industry sentiment, or economic factors.
- **Value**: Algorithms can analyze large amounts of economic and industry-specific structured and unstructured data, beyond financial statements, to predict the intrinsic value of a company.
- **Quality**: The sophisticated analysis of integrated data allows for the evaluation of customer or employee reviews, e-commerce, or app traffic to identify gains in market share or other underlying earnings quality drivers.
- **Sentiment**: The real-time processing and interpretation of news and social media content permits ML algorithms to both rapidly detect emerging sentiment and synthesize information from diverse sources into a more coherent big picture.

In practice, however, data containing valuable signals is often not freely available and is typically produced for purposes other than trading. As a result, alternative datasets require thorough evaluation, costly acquisition, careful management, and sophisticated analysis to extract tradable signals.

Sources of alternative data

Alternative datasets are generated by many sources but can be classified at a high level as predominantly produced by:

- **Individuals** who post on social media, review products, or use search engines
- **Businesses** that record commercial transactions (in particular, credit card payments) or capture supply-chain activity as intermediaries
- **Sensors** that, among many other things, capture economic activity through images from satellites or security cameras, or through movement patterns such as cell phone towers

The nature of alternative data continues to evolve rapidly as new data sources become available and sources previously labeled "alternative" become part of the mainstream. The **Baltic Dry Index (BDI)**, for instance, assembles data from several hundred shipping companies to approximate the supply/demand of dry bulk carriers and is now available on the Bloomberg Terminal.

Alternative data includes raw data as well as data that is aggregated or has been processed in some form to add value. For instance, some providers aim to extract tradeable signals, such as sentiment scores. We will address the various types of providers in *Chapter 4*, *Financial Feature Engineering – How to Research Alpha Factors*.

Alternative data sources differ in crucial respects that determine their value or signal content for algorithmic trading strategies. We will address these aspects in the next section after looking at the main sources in this one.

Individuals

Individuals automatically create electronic data through online activities, as well as through their offline activity as the latter is captured electronically and often linked to online identities. Data generated by individuals is frequently unstructured in text, image, or video formats, disseminated through multiple platforms, and includes:

- Social media posts, such as opinions or reactions on general-purpose sites such as Twitter, Facebook, or LinkedIn, or business-review sites such as Glassdoor or Yelp
- E-commerce activity that reflects an interest in or the perception of products on sites like Amazon or Wayfair
- Search engine activity using platforms such as Google or Bing
- Mobile app usage, downloads, and reviews
- Personal data such as messaging traffic

The analysis of social media sentiment has become very popular because it can be applied to individual stocks, industry baskets, or market indices. The most common source is Twitter, followed by various news vendors and blog sites. Supply is competitive, and prices are lower because it is often obtained through increasingly commoditized web scraping. Reliable social media datasets that include blogs, tweets, or videos have typically less than 5 years of history, given how recently consumers have adopted these tools at scale. Search history, in contrast, is available from 2004.

Business processes

Businesses and public entities produce and collect many valuable sources of alternative data. Data that results from business processes often has more structure than that generated by individuals. It is very effective as a leading indicator for activity that is otherwise available at a much lower frequency.

Data generated by business processes includes:

- Payment card transaction data possibly available for purchase from processors and financial institutions
- Company exhaust data produced by ordinary digitized activity or record-keeping, such as banking records, cashier scanner data, or supply chain orders

- Trade flow and market microstructure data (such as L2 and L3 order book data, illustrated by the NASDAQ ITCH tick data example in *Chapter 2*, *Market and Fundamental Data – Sources and Techniques*)
- Company payments monitored by credit rating agencies or financial institutions to assess liquidity and creditworthiness

Credit card transactions and company exhaust data, such as point-of-sale data, are among the most reliable and predictive datasets. Credit card data is available with around 10 years of history and, at different lags, almost up to real time, while corporate earnings are reported quarterly with a 2.5-week lag. The time horizon and reporting lag for company exhaust data varies widely, depending on the source. Market microstructure datasets have over 15 years of history compared to sell-side flow data, which typically has fewer than 5 years of consistent history.

Sensors

Networked sensors embedded in a broad range of devices are among the most rapidly growing data sources, driven by the proliferation of smartphones and the reduction in the cost of satellite technologies.

This category of alternative data is typically very unstructured and often significantly larger in volume than data generated by individuals or business processes, and it poses much tougher processing challenges. Key alternative data sources in this category include:

- Satellite imaging to monitor economic activity, such as construction, shipping, or commodity supply
- Geolocation data to track traffic in retail stores, such as using volunteered smartphone data, or on transport routes, such as on ships or trucks
- Cameras positioned at a location of interest
- Weather and pollution sensors

The **Internet of Things (IoT)** will further accelerate the large-scale collection of this type of alternative data by embedding networked microprocessors into personal and commercial electronic devices, such as home appliances, public spaces, and industrial production processes.

Sensor-based alternative data that contains satellite images, mobile app usage, or cellular-location tracking is typically available with a 3- to 4-year history.

Satellites

The resources and timelines required to launch a geospatial imaging satellite have dropped dramatically; instead of tens of millions of dollars and years of preparation, the cost has fallen to around $100,000 to place a small satellite as a secondary payload into a low Earth orbit. Hence, companies can obtain much higher-frequency coverage (currently about daily) of specific locations using entire fleets of satellites.

Use cases include monitoring economic activity that can be captured using aerial coverage, such as agricultural and mineral production and shipments, or the construction of commercial or residential buildings or ships; industrial incidents, such as fires; or car and foot traffic at locations of interest. Related sensor data is contributed by drones that are used in agriculture to monitor crops using infrared light.

Several challenges often need to be addressed before satellite image data can be reliably used in ML models. In addition to substantial preprocessing, these include accounting for weather conditions such as cloud cover and seasonal effects around holidays. Satellites may also offer only irregular coverage of specific locations that could affect the quality of the predictive signals.

Geolocation data

Geolocation data is another rapidly growing category of alternative data generated by sensors. A familiar source is smartphones, with which individuals voluntarily share their geographic location through an application, or from wireless signals such as GPS, CDMA, or Wi-Fi that measure foot traffic around places of interest, such as stores, restaurants, or event venues.

Furthermore, an increasing number of airports, shopping malls, and retail stores have installed sensors that track the number and movements of customers. While the original motivation to deploy these sensors was often to measure the impact of marketing activity, the resulting data can also be used to estimate foot traffic or sales. Sensors to capture geolocation data include 3D stereo video and thermal imaging, which lowers privacy concerns but works well with moving objects. There are also sensors attached to ceilings, as well as pressure-sensitive mats. Some providers use multiple sensors in combination, including vision, audio, and cellphone location, for a comprehensive account of the shopper journey, which includes not only the count and duration of visits, but extends to the conversion and measurement of repeat visits.

Criteria for evaluating alternative data

The ultimate objective of alternative data is to provide an informational advantage in the competitive search for trading signals that produce alpha, namely positive, uncorrelated investment returns. In practice, the signals extracted from alternative datasets can be used on a standalone basis or combined with other signals as part of a quantitative strategy. Independent usage is viable if the Sharpe ratio generated by a strategy based on a single dataset is sufficiently high, but that is rare in practice. (See *Chapter 4*, *Financial Feature Engineering – How to Research Alpha Factors*, for details on signal measurement and evaluation.)

Quant firms are building libraries of alpha factors that may be weak signals individually but can produce attractive returns in combination. As highlighted in *Chapter 1*, *Machine Learning for Trading – From Idea to Execution*, investment factors should be based on a fundamental and eco-

nomic rationale; otherwise, they are more likely the result of overfitting to historical data than persisting and generating alpha on new data.

Signal decay due to competition is a serious concern, and as the alternative data ecosystem evolves, it is unlikely that many datasets will retain meaningful Sharpe ratio signals. Effective strategies to extend the half-life of the signal content of an alternative dataset include exclusivity agreements, or a focus on datasets that pose processing challenges to raise the barriers to entry.

An alternative dataset can be evaluated based on the quality of its signal content, qualitative aspects of the data, and various technical aspects.

Quality of the signal content

The signal content can be evaluated with respect to the target asset class, the investment style, the relation to conventional risk premiums, and most importantly, its alpha content.

Asset classes

Most alternative datasets contain information directly relevant to equities and commodities. Interesting datasets targeting investments in real estate have also multiplied after Zillow successfully pioneered price estimates in 2006.

Alternative data on corporate credit is growing as alternative sources for monitoring corporate payments, including for smaller businesses, are being developed. Data on fixed income and around interest-rate projections is a more recent phenomenon but continues to increase as more product sales and price information are being harvested at scale.

Investment style

The majority of datasets focus on specific sectors and stocks, and as such, naturally appeal to long-short equity investors. As the scale and scope of alternative data collection continues to rise, alternative data will likely also become relevant to investors in macro themes, such as consumer credit, activity in emerging markets, and commodity trends.

Some alternative datasets that reflect broader economic activity or consumer sentiment can be used as proxies for traditional measures of market risk. In contrast, signals that capture news may be more relevant to high-frequency traders that use quantitative strategies over a brief time horizon.

Risk premiums

Some alternative datasets, such as credit card payments or social media sentiment, have been shown to produce signals that have a low correlation (lower than 5 percent) with traditional risk premiums in equity markets, such as value, momentum, and quality of volatility. As a result, combining signals derived from such alternative data with an algorithmic

trading strategy based on traditional risk factors can be an important building block toward a more diversified risk premiums portfolio.

Alpha content and quality

The signal strength required to justify the investment in an alternative dataset naturally depends on its costs, and alternative data prices vary widely. Data that scores social sentiment can be acquired for a few thousand dollars or less, while the cost of a dataset on comprehensive and timely credit card payments can cost several million per year.

We will explore in detail how to evaluate trading strategies driven by alternative data using historical data, so-called *backtests*, to estimate the amount of alpha contained in a dataset. In isolated cases, a dataset may contain sufficient alpha signal to drive a strategy on a standalone basis, but more typical is the combined use of various alternative and other sources of data. In these cases, a dataset permits the extraction of weak signals that produce a small positive Sharpe ratio that would not receive a capital allocation on its own but can deliver a portfolio-level strategy when integrated with similar other signals. This is not guaranteed, however, as there are also many alternative datasets that do not contain any alpha content.

Besides evaluating a dataset's alpha content, it is also important to assess to which extent a signal is incremental or orthogonal—that is, unique to a dataset or already captured by other data—and in the latter case, compare the costs for this type of signal.

Finally, it is essential to evaluate the potential capacity of a strategy that relies on a given, that is, the amount of capital that can be allocated without undermining its success. This is because a capacity limit will make it more difficult to recover the cost of the data.

Quality of the data

The quality of a dataset is another important criterion because it impacts the effort required to analyze and monetize it, and the reliability of the predictive signal it contains. Quality aspects include the data frequency and the length of its available history, the reliability or accuracy of the information it contains, the extent to which it complies with current or potential future regulations, and how exclusive its use is.

Legal and reputational risks

The use of alternative datasets may carry legal or reputational risks, especially when they include the following items:

- **Material non-public information (MNPI)**, because it implies an infringement of insider trading regulations
- **Personally identifiable information (PII)**, primarily since the European Union has enacted the **General Data Protection Regulation (GDPR)**

Accordingly, legal and compliance requirements need a thorough review. There could also be conflicts of interest when the provider of the data is also a market participant that is actively trading based on the dataset.

Exclusivity

The likelihood that an alternative dataset contains a signal that is sufficiently predictive to drive a strategy on a standalone basis, with a high Sharpe ratio for a meaningful period, is inversely related to its availability and ease of processing. In other words, the more exclusive and harder to process the data, the better the chances that a dataset with alpha content can drive a strategy without suffering rapid signal decay.

Public fundamental data that provides standard financial ratios contains little alpha and is not attractive for a standalone strategy, but it may help diversify a portfolio of risk factors. Large, complex datasets will take more time to be absorbed by the market, and new datasets continue to emerge on a frequent basis. Hence, it is essential to assess how familiar other investors already are with a dataset, and whether the provider is the best source for this type of information.

Additional benefits to exclusivity or being an early adopter of a new dataset may arise when a business just begins to sell exhaust data that it generated for other purposes. This is because it may be possible to influence how the data is collected or curated, or to negotiate conditions that limit access for competitors at least for a certain time period.

Time horizon

A more extensive history is highly desirable to test the predictive power of a dataset in different scenarios. The availability varies greatly between several months and several decades, and has important implications for the scope of the trading strategy that can be built and tested based on the data. We mentioned some ranges for time horizons for different datasets when introducing the main types of sources.

Frequency

The frequency of the data determines how often new information becomes available and how differentiated a predictive signal can be over a given period. It also impacts the time horizon of the investment strategy and ranges from intra-day to daily, weekly, or an even lower frequency.

Reliability

Naturally, the degree to which the data accurately reflects what it intends to measure or how well this can be verified is of significant concern and should be validated by means of a thorough audit. This applies to both raw and processed data, where the methodology used to extract or aggregate information needs to be analyzed, taking into account the cost-benefit ratio for the proposed acquisition.

Technical aspects

Technical aspects concern the latency, or delay of reporting, and the format in which the data is made available.

Latency

Data providers often provide resources in batches, and a delay can result from how the data is collected, subsequent processing and transmission, as well as regulatory or legal constraints.

Format

The data is made available in a broad range of formats, depending on the source. Processed data will be in user-friendly formats and easily integrated into existing systems or queries via a robust API. On the other end of the spectrum are voluminous data sources, such as video, audio, or image data, or a proprietary format, that require more skills to prepare for analysis, but also provide higher barriers to entry for potential competitors.

The market for alternative data

The investment industry spent an estimated $2-3 billion on data services in 2018, and this number is expected to grow at a double-digit rate per year in line with other industries. This expenditure includes the acquisition of alternative data, investments in related technology, and the hiring of qualified talent.

A survey by Ernst & Young shows significant adoption of alternative data in 2017; 43 percent of funds were using scraped web data, for instance, and almost 30 percent were experimenting with satellite data (see *Figure 3.1*). Based on the experience so far, fund managers considered scraped web data and credit card data to be most insightful, in contrast to geolocation and satellite data, which around 25 percent considered to be less informative:

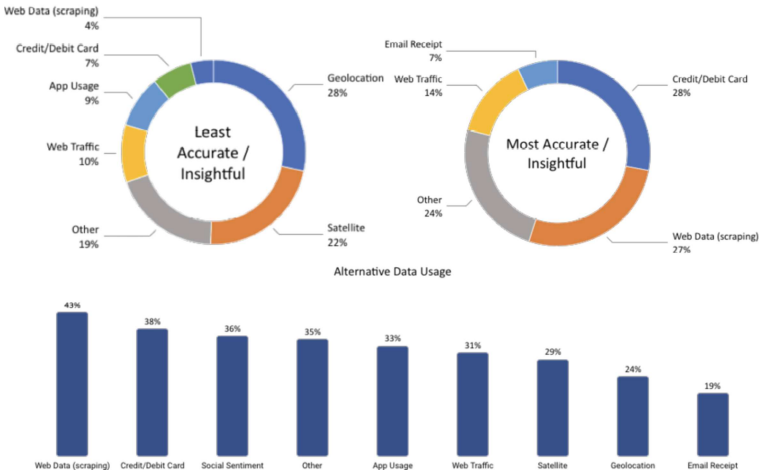

Figure 3.1: Usefulness and usage of alternative data (Source: Ernst & Young, 2017)

Reflecting the rapid growth of this new industry, the market for alternative data providers is quite fragmented. J.P. Morgan lists over 500 specialized data firms, while `AlternativeData.org` lists over 300. Providers play numerous roles, including intermediaries such as consultants, aggregators, and tech solutions; sell-side supports deliver data in various formats, ranging from raw to semi-processed data or some form of a signal extracted from one or more sources.

We will highlight the size of the main categories and profile a few prominent examples to illustrate their diversity.

Data providers and use cases

`AlternativeData.org` (supported by the provider YipitData) lists several categories that can serve as a rough proxy for activity in various data-provider segments. Social sentiment analysis is by far the largest category, while satellite and geolocation data have been growing rapidly in recent years:

Product category	# Providers
Social sentiment	48
Satellite	26
Geolocation	22
Web data and traffic	22
Infrastructure and interfaces	20
Consultants	18
Credit and debit card usage	14
Data brokers	10
Public data	10
App usage	7
Email and consumer receipts	6
Sell side	6
Weather	4
Other	87

The following brief examples aim to illustrate the broad range of service providers and potential use cases.

Social sentiment data

Social sentiment analysis is most closely associated with Twitter data. Gnip was an early social-media aggregator that provided data from numerous sites using an API and was acquired by Twitter in 2014 for $134 million. Search engines are another source that became prominent when researchers published, in *Nature*, that investment strategies based on Google Trends for terms such as debt could be used for a profitable trading strategy over an extended period (Preis, Moat, and Stanley 2013).

Dataminr

Dataminr was founded in 2009 and provides social-sentiment and news analysis based on an exclusive agreement with Twitter. The company is one of the larger alternative providers and raised an additional $391 million in funding in June 2018, led by Fidelity, at a $1.6 billion valuation, bringing total funding to $569 billion. It emphasizes real-time signals extracted from social media feeds using machine learning and serves a wide range of clients, including not only buy - and sell-side investment firms, but also news organizations and the public sector.

StockTwits

StockTwits is a social network and micro-blogging platform where several hundred thousand investment professionals share information and trading ideas in the form of StockTwits. These are viewed by a large audience across the financial web and social media platforms. This data can be exploited because it may reflect investor sentiment or itself drive trades that, in turn, impact prices. Nasseri, Tucker, and de Cesare (2015) built a trading strategy on selected features.

RavenPack

RavenPack analyzes a large amount of diverse, unstructured, text-based data to produce structured indicators, including sentiment scores, that aim to deliver information relevant to investors. The underlying data sources range from premium newswires and regulatory information to press releases and over 19,000 web publications. J.P. Morgan tested a long-short sovereign bond and equity strategies based on sentiment scores and achieved positive results, with a low correlation to conventional risk premiums (Kolanovic and Krishnamachari, 2017).

Satellite data

RS Metrics, founded in 2010, triangulates geospatial data from satellites, drones, and airplanes with a focus on metals and commodities, as well as real estate and industrial applications. The company offers signals, predictive analytics, alerts, and end-user applications based on its own high-resolution satellites. Use cases include the estimation of retail traffic at certain chains or commercial real estate, as well as the production and storage of certain common metals or employment at related production locations.

Geolocation data

Advan, founded in 2015, serves hedge fund clients with signals derived from mobile phone traffic data, targeting 1,600 tickers across various sectors in the US and EU. The company collects data using apps that install geolocation codes on smartphones with explicit user consent and track location using several channels (such as Wi-Fi, Bluetooth, and cellular signal) for enhanced accuracy. The use cases include estimates of customer traffic at physical store locations, which, in turn, can be used as input to models that predict the top-line revenues of traded companies.

Email receipt data

Eagle Alpha provides, among other services, data on a large set of online transactions using email receipts, covering over 5,000 retailers, including SKU-level transaction data categorized into 53 product groups. J.P. Morgan analyzed a time series dataset, covering 2013-16, that covered a constant group of users active throughout the entire sample period. The dataset contained the total aggregate spend, number of orders, and number of unique buyers per period (Kolanovic and Krishnamachari, 2017).

Working with alternative data

We will illustrate the acquisition of alternative data using web scraping, targeting first OpenTable restaurant data, and then move on to earnings call transcripts hosted by Seeking Alpha.

Scraping OpenTable data

Typical sources of alternative data are review websites such as Glassdoor or Yelp, which convey insider insights using employee comments or guest reviews. Clearly, user-contributed content does not capture a representative view, but rather is subject to severe selection biases. We'll look at Yelp reviews in *Chapter 14*, *Text Data for Trading – Sentiment Analysis*, for example, and find many more very positive and negative ratings on the five-star scale than you might expect. Nonetheless, this data can be valuable input for ML models that aim to predict a business's prospects or market value relative to competitors or over time to obtain trading signals.

The data needs to be extracted from the HTML source, barring any legal obstacles. To illustrate the web scraping tools that Python offers, we'll retrieve information on restaurant bookings from OpenTable. Data of this nature can be used to forecast economic activity by geography, real estate prices, or restaurant chain revenues.

Parsing data from HTML with Requests and BeautifulSoup

In this section, we will request and parse HTML source code. We will be using the Requests library to make **Hypertext Transfer Protocol** (**HTTP**) requests and retrieve the HTML source code. Then, we'll rely on the Beautiful Soup library, which makes it easy to parse the HTML markup code and extract the text content we are interested in.

We will, however, encounter a common obstacle: websites may request certain information from the server only after initial page-load using JavaScript. As a result, a direct HTTP request will not be successful. To sidestep this type of protection, we will use a headless browser that retrieves the website content as a browser would:

```
from bs4 import BeautifulSoup
import requests
# set and request url; extract source code
url = https://www.opentable.com/new-york-restaurant-listings
html = requests.get(url)
html.text[:500]
' <!DOCTYPE html><html lang="en"><head><meta charset="utf-8"/><meta http-equiv="X-UA-Compatible" c
```

Now, we can use Beautiful Soup to parse the HTML content, and then look for all span tags with the class associated with the restaurant names that we obtain by inspecting the source code, `rest-row-name-text` (see the GitHub repository for linked instructions to examine website source code):

```
# parse raw html => soup object
soup = BeautifulSoup(html.text, 'html.parser')
# for each span tag, print out text => restaurant name
for entry in soup.find_all(name='span', attrs={'class':'rest-row-name-text'}):
    print(entry.text)
Wade Coves
Alley
Dolorem Maggio
Islands
...
```

Once you have identified the page elements of interest, Beautiful Soup makes it easy to retrieve the contained text. If you want to get the price category for each restaurant, for example, you can use:

```
# get the number of dollars signs for each restaurant
for entry in soup.find_all('div', {'class':'rest-row-pricing'}):
    price = entry.find('i').text
```

When you try to get the number of bookings, however, you just get an empty list because the site uses JavaScript code to request this information after the initial loading is complete:

```
soup.find_all('div', {'class':'booking'})
[]
```

This is precisely the challenge we mentioned earlier—rather than sending all content to the browser as a static page that can be easily parsed, JavaScript loads critical pieces dynamically. To obtain this content, we need to execute the JavaScript just like a browser—that's what Selenium is for.

Introducing Selenium – using browser automation

We will use the browser automation tool Selenium to operate a headless Firefox browser that will parse the HTML content for us.

The following code opens the Firefox browser:

```
from selenium import webdriver
# create a driver called Firefox
driver = webdriver.Firefox()
```

Let's close the browser:

```
# close it
driver.close()
```

Now, we retrieve the HTML source code, including the parts loaded dynamically, with Selenium and Firefox. To this end, we provide the URL to our driver and then use its `page_source` attribute to get the full-page content, as displayed in the browser.

From here on, we can fall back on Beautiful Soup to parse the HTML, as follows:

```
import time, re
# visit the opentable listing page
driver = webdriver.Firefox()
driver.get(url)
time.sleep(1) # wait 1 second
# retrieve the html source
html = driver.page_source
html = BeautifulSoup(html, "lxml")
for booking in html.find_all('div', {'class': 'booking'}):
    match = re.search(r'\d+', booking.text)
    if match:
        print(match.group())
```

Building a dataset of restaurant bookings and ratings

Now, you only need to combine all the interesting elements from the website to create a feature that you could use in a model to predict economic activity in geographic regions, or foot traffic in specific neighborhoods.

With Selenium, you can follow the links to the next pages and quickly build a dataset of over 10,000 restaurants in NYC, which you could then update periodically to track a time series.

First, we set up a function that parses the content of the pages that we plan on crawling, using the familiar Beautiful Soup parse syntax:

```
def parse_html(html):
    data, item = pd.DataFrame(), {}
    soup = BeautifulSoup(html, 'lxml')
```

```
    for i, resto in enumerate(soup.find_all('div',
                                        class_='rest-row-info')):
        item['name'] = resto.find('span',
                                  class_='rest-row-name-text').text
        booking = resto.find('div', class_='booking')
        item['bookings'] = re.search('\d+', booking.text).group() \
            if booking else 'NA'
        rating = resto.find('div', class_='star-rating-score')
        item['rating'] = float(rating['aria-label'].split()[0]) \
            if rating else 'NA'
        reviews = resto.find('span', class_='underline-hover')
        item['reviews'] = int(re.search('\d+', reviews.text).group()) \
            if reviews else 'NA'
        item['price'] = int(resto.find('div', class_='rest-row-pricing')
                            .find('i').text.count('$'))
        cuisine_class = 'rest-row-meta--cuisine rest-row-meta-text sfx1388addContent'
        item['cuisine'] = resto.find('span', class_=cuisine_class).text
        location_class = 'rest-row-meta--location rest-row-meta-text sfx1388addContent'
        item['location'] = resto.find('span', class_=location_class).text
        data[i] = pd.Series(item)
    return data.T
```

Then, we start a headless browser that continues to click on the **Next** button for us and captures the results displayed on each page:

```
restaurants = pd.DataFrame()
driver = webdriver.Firefox()
url = https://www.opentable.com/new-york-restaurant-listings
driver.get(url)
while True:
    sleep(1)
    new_data = parse_html(driver.page_source)
    if new_data.empty:
        break
    restaurants = pd.concat([restaurants, new_data], ignore_index=True)
    print(len(restaurants))
    driver.find_element_by_link_text('Next').click()
driver.close()
```

A sample run in early 2020 yields location, cuisine, and price category information on 10,000 restaurants. Furthermore, there are same-day booking figures for around 1,750 restaurants (on a Monday), as well as ratings and reviews for around 3,500 establishments.

Figure 3.2 shows a quick summary: the left panel displays the breakdown by price category for the top 10 locations with the most restaurants. The central panel suggests that ratings are better, on average, for more expensive restaurants, and the right panel highlights that better - rated restaurants receive more bookings. Tracking this information over time could be informative, for example, with respect to consumer sentiment, location preferences, or specific restaurant chains:

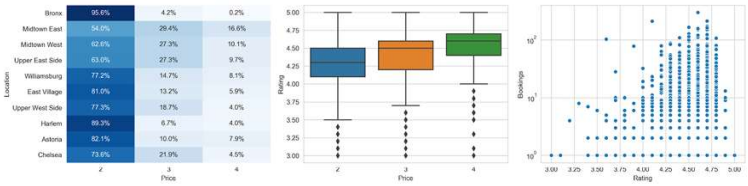

Figure 3.2: OpenTable data summary

Websites continue to change, so this code may stop working at some point. To update our bot, we need to identify the changes to the site navigation, such as new class or ID names, and correct the parser accordingly.

Taking automation one step further with Scrapy and Splash

Scrapy is a powerful library used to build bots that follow links, retrieve the content, and store the parsed result in a structured way. In combination with the Splash headless browser, it can also interpret JavaScript and becomes an efficient alternative to Selenium.

You can run the spider using the `scrapy crawl opentable` command in the `01_opentable` directory, where the results are logged to `spider.log`:

```
from opentable.items import OpentableItem
from scrapy import Spider
from scrapy_splash import SplashRequest
class OpenTableSpider(Spider):
    name = 'opentable'
    start_urls = ['https://www.opentable.com/new-york-restaurant-
                  listings']
    def start_requests(self):
        for url in self.start_urls:
            yield SplashRequest(url=url,
                                callback=self.parse,
                                endpoint='render.html',
                                args={'wait': 1},
                                )
    def parse(self, response):
        item = OpentableItem()
        for resto in response.css('div.rest-row-info'):
            item['name'] = resto.css('span.rest-row-name-
                                     text::text').extract()
            item['bookings'] =
                  resto.css('div.booking::text').re(r'\d+')
            item['rating'] = resto.css('div.all-
                  stars::attr(style)').re_first('\d+')
            item['reviews'] = resto.css('span.star-rating-text--review-
                                        text::text').re_first(r'\d+')
            item['price'] = len(resto.css('div.rest-row-pricing >
                                i::text').re('\$'))
            item['cuisine'] = resto.css('span.rest-row-meta—
                                        cuisine::text').extract()
            item['location'] = resto.css('span.rest-row-meta—
                                location::text').extract()
            yield item
```

There are numerous ways to extract information from this data beyond the reviews and bookings of individual restaurants or chains.

We could further collect and geo-encode the restaurants' addresses, for instance, to link the restaurants' physical location to other areas of interest, such as popular retail spots or neighborhoods to gain insights into particular aspects of economic activity. As mentioned previously, such data will be most valuable in combination with other information.

Scraping and parsing earnings call transcripts

Textual data is an essential alternative data source. One example of textual information is the transcripts of earnings calls, where executives do not only present the latest financial results, but also respond to questions by financial analysts. Investors utilize transcripts to evaluate changes in sentiment, emphasis on particular topics, or style of communication.

We will illustrate the scraping and parsing of earnings call transcripts from the popular trading website **www.seekingalpha.com**. As in the OpenTable example, we'll use Selenium to access the HTML code and Beautiful Soup to parse the content. To this end, we begin by instantiating a Selenium `webdriver` instance for the Firefox browser:

```
from urllib.parse import urljoin
from bs4 import BeautifulSoup
from furl import furl
from selenium import webdriver
transcript_path = Path('transcripts')
SA_URL = 'https://seekingalpha.com/'
TRANSCRIPT = re.compile('Earnings Call Transcript')
next_page = True
page = 1
driver = webdriver.Firefox()
```

Then, we iterate over the transcript pages, creating the URLs based on the navigation logic we obtained from inspecting the website. As long as we find relevant hyperlinks to additional transcripts, we access the webdriver's `page_source` attribute and call the `parse_html` function to extract the content:

```
while next_page:
    url = f'{SA_URL}/earnings/earnings-call-transcripts/{page}'
    driver.get(urljoin(SA_URL, url))
    response = driver.page_source
    page += 1
    soup = BeautifulSoup(response, 'lxml')
    links = soup.find_all(name='a', string=TRANSCRIPT)
    if len(links) == 0:
        next_page = False
    else:
        for link in links:
            transcript_url = link.attrs.get('href')
            article_url = furl(urljoin(SA_URL,
                           transcript_url)).add({'part': 'single'})
            driver.get(article_url.url)
```

```
            html = driver.page_source
            meta, participants, content = parse_html(html)
            meta['link'] = link
driver.close()
```

To collect structured data from the unstructured transcripts, we can use regular expressions in addition to Beautiful Soup.

They allow us to collect detailed information not only about the earnings call company and timing, but also about who was present and attribute the statements to analysts and company representatives:

```
def parse_html(html):
    date_pattern = re.compile(r'(\d{2})-(\d{2})-(\d{2})')
    quarter_pattern = re.compile(r'(\bQ\d\b)')
    soup = BeautifulSoup(html, 'lxml')
    meta, participants, content = {}, [], []
    h1 = soup.find('h1', itemprop='headline').text
    meta['company'] = h1[:h1.find('(')].strip()
    meta['symbol'] = h1[h1.find('(') + 1:h1.find(')')]
    title = soup.find('div', class_='title').text
    match = date_pattern.search(title)
    if match:
        m, d, y = match.groups()
        meta['month'] = int(m)
        meta['day'] = int(d)
        meta['year'] = int(y)
    match = quarter_pattern.search(title)
    if match:
        meta['quarter'] = match.group(0)
    qa = 0
    speaker_types = ['Executives', 'Analysts']
    for header in [p.parent for p in soup.find_all('strong')]:
        text = header.text.strip()
        if text.lower().startswith('copyright'):
            continue
        elif text.lower().startswith('question-and'):
            qa = 1
            continue
        elif any([type in text for type in speaker_types]):
            for participant in header.find_next_siblings('p'):
                if participant.find('strong'):
                    break
                else:
                    participants.append([text, participant.text])
        else:
            p = []
            for participant in header.find_next_siblings('p'):
                if participant.find('strong'):
                    break
                else:
                    p.append(participant.text)
            content.append([header.text, qa, '\n'.join(p)])
    return meta, participants, content
```

We'll store the result in several `.csv` files for easy access when we use ML to process natural language in *Chapters 14-16*:

```
def store_result(meta, participants, content):
    path = transcript_path / 'parsed' / meta['symbol']
    pd.DataFrame(content, columns=['speaker', 'q&a',
              'content']).to_csv(path / 'content.csv', index=False)
    pd.DataFrame(participants, columns=['type', 'name']).to_csv(path /
                 'participants.csv', index=False)
    pd.Series(meta).to_csv(path / 'earnings.csv')
```

See the `README` in the GitHub repository for additional details and references for further resources to learn how to develop web scraping applications.

Summary

In this chapter, we introduced new sources of alternative data made available as a result of the big data revolution, including individuals, business processes, and sensors, such as satellites or GPS location devices. We presented a framework to evaluate alternative datasets from an investment perspective and laid out key categories and providers to help you navigate this vast and quickly expanding area that provides critical inputs for algorithmic trading strategies that use ML.

We also explored powerful Python tools you can use to collect your own datasets at scale. We did this so that you can potentially work on getting your private informational edge as an algorithmic trader using web scraping.

We will now proceed, in the following chapter, to the design and evaluation of alpha factors that produce trading signals and look at how to combine them in a portfolio context.