Extract Textual Data from EDGAR 10-K Filings Using Python

Open in Colab Download notebook

This Python tutorial demonstrates how to extract specific sections of textual data from SEC EDGAR 10-K filings, without relying on regular expressions or custom BeautifulSoup extractors.

The tutorial covers the extraction of any of the 19 10-K filing sections, from "Item 1 - Business" to "Item 7 - MD&A, Management's Discussion of Financial Results" to "Item 14 - Principal Accountant Fees and Services".

Our approach is using the Extractor API to easily extract sections in either HTML, which includes iXBRL data, or text format. The text format includes the raw text without HTML, but with table starts and ends marked. The tutorial also touches on using pandas to convert extracted financial statements from HTML tables into dataframes.

Although an alternative extraction approach using BeautifulSoup and regular expressions is possible, it can be error-prone since every filing has a unique structure and only about 30% of the content can be extracted using regular expressions. There are several related stackoverflow questions, but none of them provide a solution that can cover the majority of EDGAR filings.

Getting Started

The first step is to install the `sec-api` Python package which provides access to the `ExtractorApi`.

```
API_KEY = 'YOUR_API_KEY'
```

```
!pip install -q sec-api
```

```
from sec_api import ExtractorApi

extractorApi = ExtractorApi(API_KEY)
```

We define the `pprint` helper function to convert long, single-line text into a multi-line, easily readable format. This function is used to output the extracted text sections in a more readable format, especially when running the code in a Jupyter notebook.

```
# helper function to pretty print long, single-line text to multi-line text
def pprint(text, line_length=100):
    words = text.split(' ')
    lines = []
    current_line = ''
    for word in words:
        if len(current_line + ' ' + word) <= line_length:
            current_line += ' ' + word
        else:
            lines.append(current_line.strip())
            current_line = word
    if current_line:
        lines.append(current_line.strip())
    print('\n'.join(lines))
```

Extract "Item 1 - Business" from 10-K Filings

We will begin by extracting the business section (Item 1) from a 10-K filing using the `.get_section(filing_url, section_id, output_type)` function. This function allows us to specify the URL of the 10-K filing, the ID of the item section to be extracted, and the desired output type (HTML or text), and returns the extracted section. Refer to the documentation for a complete list of all 10-K item section IDs.

As an example, let's extract Item 1 as text from Tesla's 10-K filing.

```
# URL of Tesla's 10-K filing
filing_10_k_url = 'https://www.sec.gov/Archives/edgar/data/1318605/000156459021004599/tsla-10k_2(

# extract text section "Item 1 - Business" from 10-K
item_1_text = extractorApi.get_section(filing_10_k_url, '1', 'text')

print('Extracted Item 1 (Text)')
print('----------------------')
pprint(item_1_text[0:1500])
print('
... cut for brevity')
print('----------------------')
```

```
Extracted Item 1 (Text)
----------------------
ITEM 1.

BUSINESS
```

```
##TABLE_END

Overview

We design, develop, manufacture, sell and lease
high-performance fully electric vehicles and energy generation and storage systems, an
services related to our sustainable energy products. We generally sell our products di
customers, including through our website and retail locations. We also continue to gro
customer-facing infrastructure through a global network of vehicle service centers, Mo
technicians, body shops, Supercharger stations and Destination Chargers to accelerate
adoption of our products. We emphasize performance, attractive styling and the safety
and workforce in the design and manufacture of our products and are continuing to deve
self-driving technology for improved safety. We also strive to lower the cost of owner
customers through continuous efforts to reduce manufacturing costs and by offering fin
services tailored to our products. Our mission to accelerate the world&#8217;s transit
sustainable energy, engineering expertise, vertically integrated business model and fo
experience differentiate us from other companies.

Segment Information

We operate as two
reportable segments: (i) automotive and (ii) energy generation and storage.

The automotive segment
```

Now let's see how we can extract the same section "Item 1 - Business" in HTML format.

```
from IPython.display import display, HTML
```

```
# extract HTML section "Item 1 - Business" from 10-K
item_1_html = extractorApi.get_section(filing_10_k_url, '1', 'html')

print('Extracted Item 1 (HTML)')
print('-----------------------')
display(HTML(item_1_html[0:3000]))
print('
... cut for brevity')
print('-----------------------')
```

```
Extracted Item 1 (HTML)
-----------------------
```

ITEM 1.

BUSINESS

Overview

We design, develop, manufacture, sell and lease high-performance fully electric vehicles and energy generation and storage systems, and offer services related to our sustainable energy products. We generally sell our products directly to customers, including through our website and retail locations. We also continue to grow our customer-facing infrastructure through a global network of vehicle service centers, Mobile Service technicians, body shops, Supercharger stations and Destination Chargers to accelerate the widespread adoption of our products. We emphasize performance, attractive styling and the safety of our users and workforce in the design and manufacture of our products and are continuing to develop full self-driving technology for improved safety. We also strive to lower the cost of ownership for our customers through continuous efforts to reduce manufacturing costs and by offering financial services tailored to our products. Our mission to accelerate the world's transition to sustainable energy, engineering expertise, vertically integrated business model and focus on user experience differentiate us from other companies.

Segment Information

We operate as two reportable segments: (i) automotive and (ii) energy generation and storage.

The automotive segment includes the design, development, manufacturing, sales and leasing of electric vehicles as well as sales of automotive regulatory credits. Additionally, the automotive segment is also comprised of services and other, which includes non-warranty after-sales vehicle services,

```
... cut for brevity
-----------------------
```

Extract "Item 6 - Financial Data" from 10-K Filings

In this example, we'll show you how to extract the "Selected Financial Data" section (Item 6) of a 10-K filing in HTML format. This section includes financial statements in the form of HTML tables. We can then use pandas to convert the tables and access the financial data in a dataframe.

```
# extract the HTML version of section "Item 6 - Selected Financial Data"
item_6_html = extractorApi.get_section(filing_10_k_url, '6', 'html')
```

```
print('Extracted Item 6 (HTML)')
print('-----------------------')
display(HTML(item_6_html[0:150000]))
print('
... cut for brevity')
print('-----------------------')
```

```
Extracted Item 6 (HTML)
-----------------------
```

ITEM 6.

SELECTED CONSOLIDATED FINANCIAL DATA

The following selected consolidated financial data should be read in conjunction with "Management's Discussion and Analysis of Financial Condition and Results of Operations" and the consolidated financial statements and the related notes included elsewhere in this Annual Report on Form 10-K and from the historical consolidated financial statements not included herein to fully understand factors that may affect the comparability of the information presented below (in millions, except per share data).

	Year Ended December 31,				
	2020	2019 (3)	2018 (2)	2017	2016 (1)
Consolidated Statements of Operations Data:					
Total revenues	$ 31,536	$ 24,578	$ 21,461	$ 11,759	$ 7,000
Gross profit	$ 6,630	$ 4,069	$ 4,042	$ 2,223	$ 1,599
Income (loss) from operations	$ 1,994	$ (69)	$ (388)	$ (1,632)	$ (667)
Net income (loss) attributable to common stockholders	$ 721	$ (862)	$ (976)	$ (1,962)	$ (675)
Net income (loss) per share of common stock attributable to common stockholders (4)					
Basic	$ 0.74	$ (0.98)	$ (1.14)	$ (2.37)	$ (0.94)
Diluted	$ 0.64	$ (0.98)	$ (1.14)	$ (2.37)	$ (0.94)
Weighted average shares used in computing net income (loss) per share of common stock (4)					
Basic	933	887	853	829	721
Diluted	1,083	887	853	829	721

	As of December 31,				
	2020	2019 (3)	2018 (2)	2017	2016 (1)
Consolidated Balance Sheet Data:					
Working capital (deficit)	$ 12,469	$ 1,436	$ (1,686)	$ (1,104)	$ 433
Total assets	$ 52,148	$ 34,309	$ 29,740	$ 28,655	$ 22,664
Total long-term liabilities	$ 14,170	$ 15,532	$ 13,434	$ 15,348	$ 10,923

(1) We acquired SolarCity Corporation ("SolarCity") on November 21, 2016. SolarCity's financial results have been included in our financial results from the acquisition date as previously reported in our Annual Report on Form 10-K for the year ended December 31, 2016.

(2) We adopted ASC 606 in 2018. Prior periods have not been revised. For further details, refer to Note 2, *Summary of Significant Accounting Policies*, of the notes to the consolidated financial statements included in our Annual Report on Form 10-K for the year ended December 31, 2018.

(3) We adopted ASC 842 in 2019. Prior periods have not been revised. For further details, refer to Note 2, *Summary of Significant Accounting Policies*, of the notes to the consolidated financial statements included in our Annual Report on Form 10-K for the year ended December 31, 2019.

(4) Prior period results have been adjusted to give effect to the Stock Split. See Note 1, *Overview*, of the notes to the consolidated financial statements included elsewhere in this Annual Report on Form 10-K for further details.


```
... cut for brevity
-----------------------
```

Let's now take a look at the actual content of the extracted section. This will display the raw HTML of the section, including all HTML elements and their corresponding style attributes.

```
print('Extracted Content of Item 6 (HTML)')
print('----------------------')
pprint(item_6_html[0:1000])
```

```
print('
... cut for brevity')
print('----------------------')
```

```
Extracted Content of Item 6 (HTML)
----------------------
<span style="font-weight:bold;font-family:Times New Roman
Bold;font-size:10pt;font-style:normal;text-transform:none;font-variant: normal;">ITEM
6.</span></p></td> <td valign="top"> <p
style="margin-bottom:0pt;margin-top:0pt;font-weight:bold;font-style:normal;text-transfor
normal;font-family:Times New Roman Bold;font-size:10pt;"
id="ITEM_6_SELECTED_CONSOLIDATED_FINANCIAL_D">SELECTED CONSOLIDATED FINANCIAL
DATA</p></td></tr></table></div> <p
style="margin-top:4pt;margin-bottom:0pt;text-indent:4.54%;font-family:Times New
Roman;font-size:10pt;font-weight:normal;font-style:normal;text-transform:none;font-varia
normal;">The following selected consolidated financial data should be read in conjunctio
&amp;#8220;Management&amp;#8217;s Discussion and Analysis of Financial Condition and Results of
Operations&amp;#8221; and the consolidated financial statements and the related notes includ
in this Annual Report on Form 10-K and from the historical consolidated financia

... cut for brevity
----------------------
```

Convert Financial Statements in an HTML Table to a DataFrame

Next, you can use pandas to convert the financial statements table from section 6 of the filing, which is a HTML table, into a dataframe. Using `pd.read_html(item_6_html)`, pandas can locate and convert all HTML tables into dataframes. Once the financial statements table is extracted, we clean the dataframe by removing empty columns and unnecessary `NaN` values. This makes it easy to access all financial data, such as total revenue, gross profit, or net income.

```
import pandas as pd

# read HTML table from a string and convert to dataframe
tables = pd.read_html(item_6_html)
# first table includes the financial statements
df = tables[0]
```

```
# drop all columns with NaN values except if the first cell is not NaN
mask = (df.iloc[1:, :].isna()).all(axis=0)
financial_statements = df.drop(df.columns[mask], axis=1).fillna('')
print('Consolidated financial statements as dataframe:')
financial_statements
```

```
Consolidated financial statements as dataframe:
```

Out[12]:

	0	2	3	6	7	8	10	11	
0		Year Ended December 31,	Year Ended December 31,	Year Ended December 31,	Year Ended December 31,	Year Ended December 31,	Year Ended December 31,	Year Ended December 31,	Y End Decemb
1		2020	2020	2019 (3)	2019 (3)		2018 (2)	2018 (2)	

	0	2	3	6	7	8	10	11
2	Consolidated Statements of Operations Data:							
3	Total revenues	$	31536	$	24578		$	21461
4	Gross profit	$	6630	$	4069		$	4042
5	Income (loss) from operations	$	1994	$	(69	)	$	(388
6	Net income (loss) attributable to common stock	$	721	$	(862	)	$	(976

Extract Other Text Sections from 10-K Filings

We conclude the tutorial with an overview of all the available parameters for the 10-K extractor.

```
# # extract text sections
item_1_text    = extractorApi.get_section(filing_10_k_url, '1', 'text')
item_1_a_text  = extractorApi.get_section(filing_10_k_url, '1A', 'text')
item_1_b_text  = extractorApi.get_section(filing_10_k_url, '1B', 'text')
item_2_text    = extractorApi.get_section(filing_10_k_url, '2', 'text')
item_3_text    = extractorApi.get_section(filing_10_k_url, '3', 'text')
item_4_text    = extractorApi.get_section(filing_10_k_url, '4', 'text')
item_5_text    = extractorApi.get_section(filing_10_k_url, '5', 'text')
item_6_text    = extractorApi.get_section(filing_10_k_url, '6', 'text')
item_7_text    = extractorApi.get_section(filing_10_k_url, '7', 'text')
item_7_a_text  = extractorApi.get_section(filing_10_k_url, '7A', 'text')
item_8_text    = extractorApi.get_section(filing_10_k_url, '8', 'text')
item_9_text    = extractorApi.get_section(filing_10_k_url, '9', 'text')
item_9_a_text  = extractorApi.get_section(filing_10_k_url, '9A', 'text')
item_9_b_text  = extractorApi.get_section(filing_10_k_url, '9B', 'text')
item_10_text   = extractorApi.get_section(filing_10_k_url, '10', 'text')
item_11_text   = extractorApi.get_section(filing_10_k_url, '11', 'text')
item_12_text   = extractorApi.get_section(filing_10_k_url, '12', 'text')
item_13_text   = extractorApi.get_section(filing_10_k_url, '13', 'text')
item_14_text   = extractorApi.get_section(filing_10_k_url, '14', 'text')
item_15_text   = extractorApi.get_section(filing_10_k_url, '15', 'text')

# # extract HTML sections
item_1_html    = extractorApi.get_section(filing_10_k_url, '1', 'html')
item_1_a_html  = extractorApi.get_section(filing_10_k_url, '1A', 'html')
item_1_b_html  = extractorApi.get_section(filing_10_k_url, '1B', 'html')
item_2_html    = extractorApi.get_section(filing_10_k_url, '2', 'html')
item_3_html    = extractorApi.get_section(filing_10_k_url, '3', 'html')
item_4_html    = extractorApi.get_section(filing_10_k_url, '4', 'html')
item_5_html    = extractorApi.get_section(filing_10_k_url, '5', 'html')
item_6_html    = extractorApi.get_section(filing_10_k_url, '6', 'html')
```

```
item_7_html    = extractorApi.get_section(filing_10_k_url, '7', 'html')
item_7_a_html  = extractorApi.get_section(filing_10_k_url, '7A', 'html')
item_8_html    = extractorApi.get_section(filing_10_k_url, '8', 'html')
item_9_html    = extractorApi.get_section(filing_10_k_url, '9', 'html')
item_9_a_html  = extractorApi.get_section(filing_10_k_url, '9A', 'html')
item_9_b_html  = extractorApi.get_section(filing_10_k_url, '9B', 'html')
item_10_html   = extractorApi.get_section(filing_10_k_url, '10', 'html')
item_11_html   = extractorApi.get_section(filing_10_k_url, '11', 'html')
item_12_html   = extractorApi.get_section(filing_10_k_url, '12', 'html')
item_13_html   = extractorApi.get_section(filing_10_k_url, '13', 'html')
item_14_html   = extractorApi.get_section(filing_10_k_url, '14', 'html')
item_15_html   = extractorApi.get_section(filing_10_k_url, '15', 'html')
```

CIK, CUSIP, Ticker

Mapping

Executive

Compensation API

Form 13D/13G

Search API

10-K/10-Q/8-K Item

Extractor

Float (Outstanding

Shares) API