5 Obscure Python Libraries Every Data Scientist Should Know

Enhance your data science project

Andre Ye

Source:

1 | Scrapy

Every data scientist’s project begins with data, and the Internet is the largest, richest, and most accessible trove of data. Unfortunately, beyond `pd.read_html`, most data scientists are clueless when it comes to scraping data off websites that have complex data structures. Scrapy makes building web crawlers that analyze the anatomy of a website and store the extracted information much easier than building it from scratch would be.

Scrapy's biggest advantage, besides its clean user interface, is its efficiency. Requests to websites sent with Scrapy are scheduled and processed asynchronously, meaning that if one request takes longer to be finished and processed, another request can be sent in the meantime. Very fast crawling can be employed with Scrapy by sending multiple concurrent requests to a website, iterating over its content in the most productive manner possible.

On top of this, Scrapy allows data scientists to export their saved data in multiple formats — JSON, CSV, or XML — as well as different backends — FTP, S3, local.

Documentation.

2 | Pattern

Some websites that are more well-established may already have a more concrete method of retrieving data, and in this case using Scrapy to write a web crawler could be overkill. Pattern is a more high-level web mining and natural language processing module in Python.

Not only does it have seamless integration with Google, Twitter, and Wikipedia data, as well as a less customizable web crawler and an HTML DOM parser, Pattern employs POS (part of speech) tagging, *n*-grams searching, sentiment analysis, and WordNet. The result of preprocessed text data can be used in a variety of implemented machine learning algorithms, from clustering to classification, or visualized with network analysis.

Pattern has everything in the data science pipeline, from data retrieval to preprocessing to modelling and visualization, and avoids clumsy transferring of data across a mix of different libraries.

Documentation.

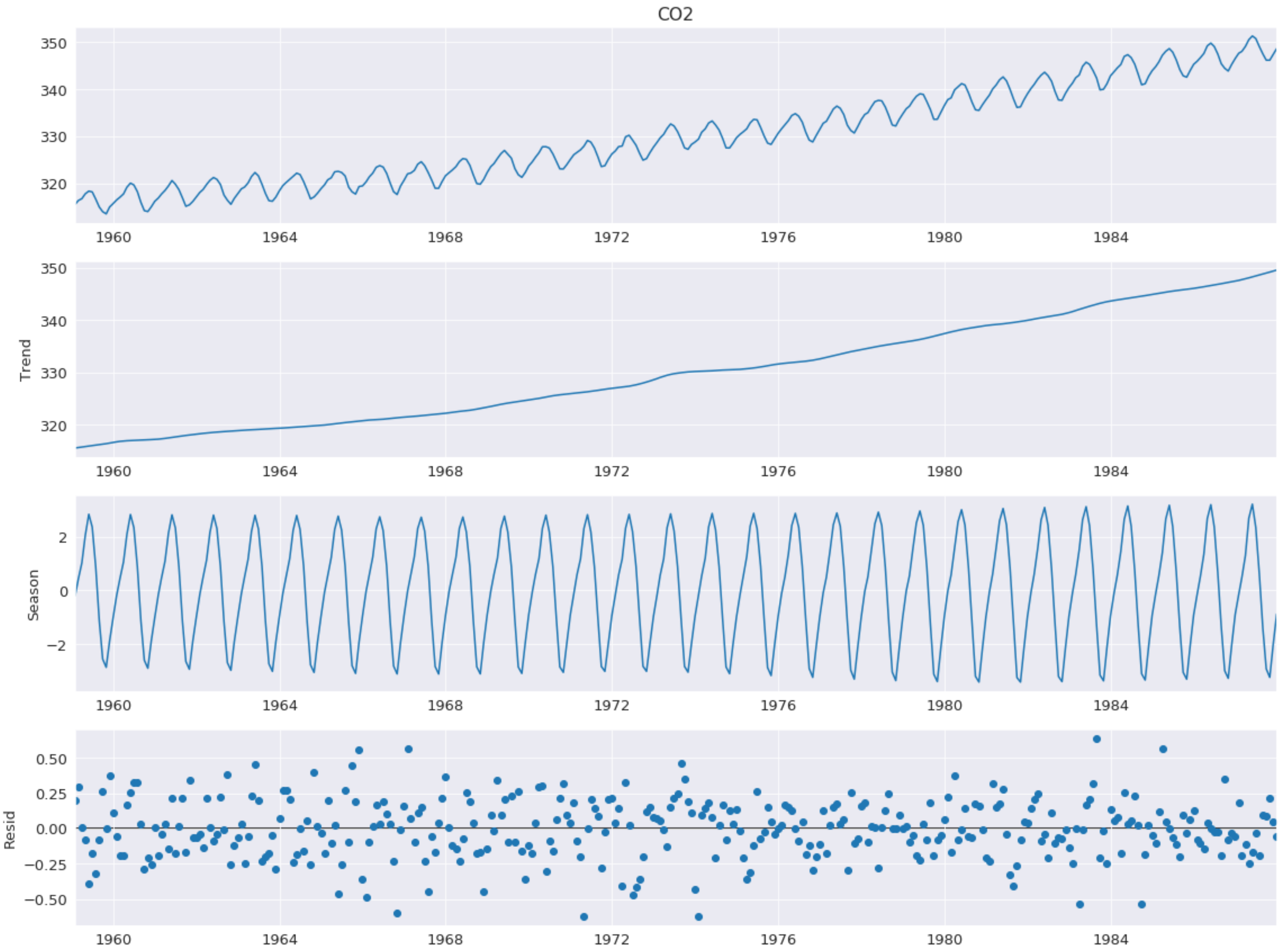
3 | Statsmodels

Although data scientists are generally hesitant to approach statistical modelling methods, Statsmodels is a must-know library. Besides offering important implementations of algorithms like ANOVA and ARIMA that standard machine learning libraries like Sci-kit Learn do not have, perhaps what is most valuable about Statsmodels is the sheer level of detail and information it provides.

Consider, for instance, an ordinary least squares regression trained using Statsmodels. All the information a data scientist may possibly want, from useful metrics to detailed information on the coefficients, are given by Statsmodels, and the same is with every other model implemented in the library. You’re not going to get this with Sci-kit learn!

Having all this information is incredibly valuable for data scientists, who too often place too much trust into a model they don’t really understand. Because high-dimensional data is naturally unintuitive for us, it is necessary for us to gain an understanding of the data and the model before we rush too hastily to deploy it. Blindly chasing pure performance metrics like accuracy or mean squared error will have negative repercussions.

Beyond incredibly detailed statistical modelling, Statsmodels also offers a variety of helpful data features and metrics. Consider, for instance, their implementation of Seasonal-Trend decomposition, which can help data scientists better understand their data and which transformations and algorithms are better suited to it — this information is tremendously valuable.



Source: Statsmodels. Image free to share.

Documentation.

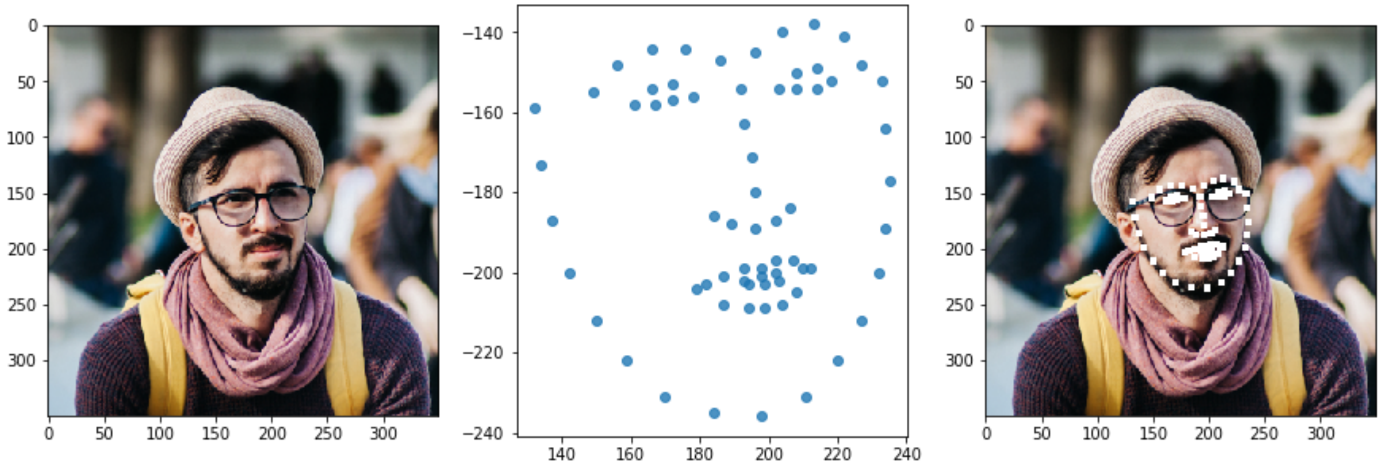
Alternatively, consider using pinguoin for a less complex but still incredibly insightful statistical functions.

4 | Mlxtend

Mlxtend is a library that should accompany any data science project. Considered as an extension of the Sci-kit learn library, Mlxtend has useful automation of common data science tasks:

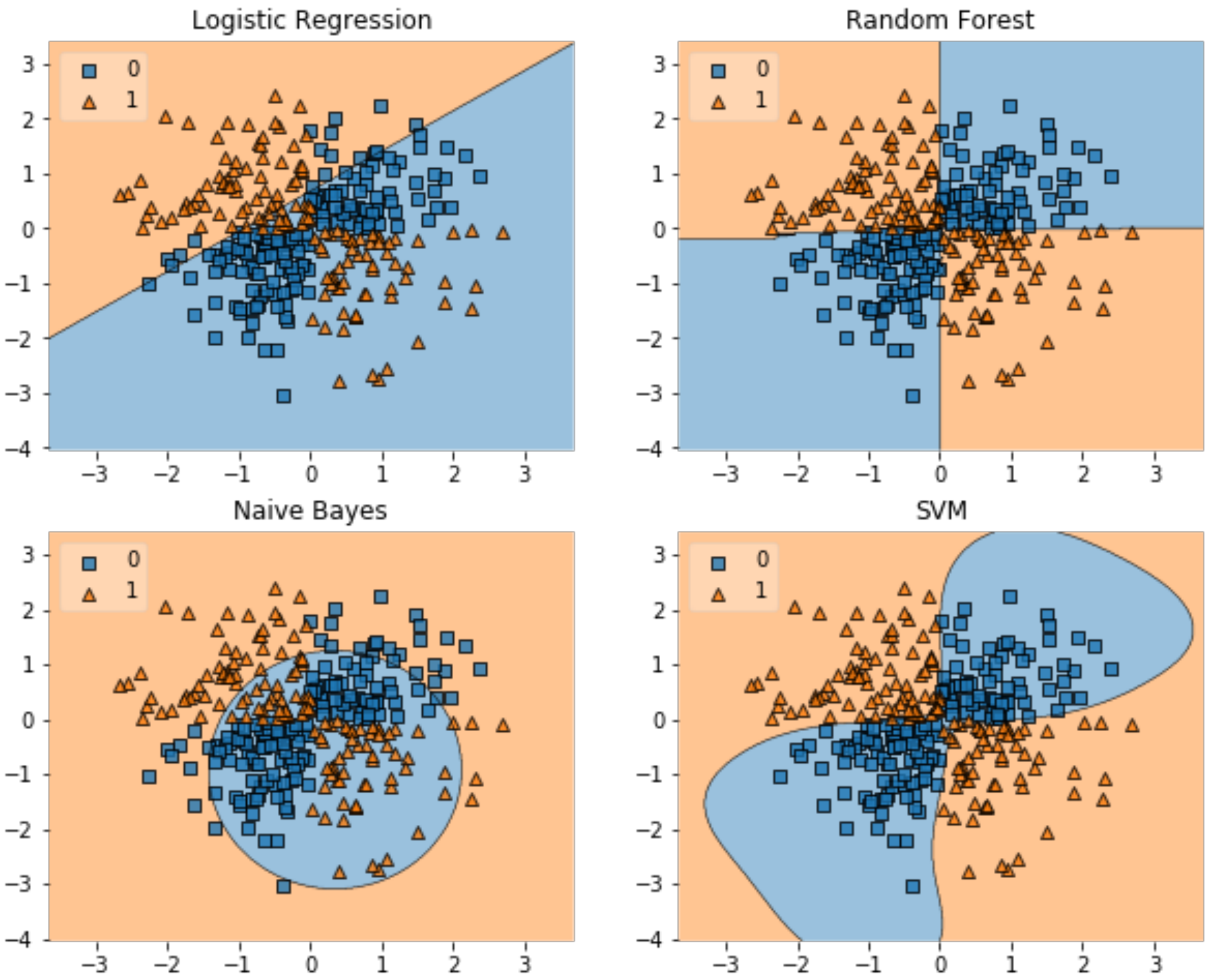
- Completely automated feature extraction and selection.
- An extension on Sci-kit learn's existing data transformers, like mean centering and transaction encoders.
- A vast array of evaluation metrics: a few include bias-variance decomposition (measure how bias and variance your model contains), lift tests, McNemar's test, F-test, and many more.
- Helpful model visualizations, including feature boundaries, learning curves, PCA correlation circles, and enrichment plots.
- Many built-in datasets that are not included in Sci-kit Learn.
- Helpful preprocessing functions for images and text, like a name generalizer that can identify and convert text with different naming systems ("Deer, John", "J. Deer", "J. D.", and "John Deer" are the same).

Mlxtend also has useful image processing features, like one that can extract facial landmarks:



Source: Mlxtend. Image free to share.

Or, consider its decision boundary drawing capabilities:



Source: Mlxtend. Image free to share.

Documentation.

5 | REP

Like Mlxtend, REP can be thought of as an extension of sorts to the Sci-kit learn library, but more in the machine learning department. Primarily, it is a unified Python wrapper for different machine learning libraries extending from Sci-kit learn. Use XGBoost, Pybrain, Neurolab, and many other more specialized machine learning libraries in integration with Sci-kit learn.

Consider, for instance, altering a XGBoost classifier into a bagging classifier through a simple wrapper, then converting it into a Sci-kit learn model—

such ease in wrapping and converting algorithms would never be able to be found in other libraries.

Beyond this, REP has other implementations to transform models from any library into cross-validation (folded) and stacked models. Other features include an extremely fast grid search and model factories, which can train several machine learning classifiers on the same dataset efficiently.

By using REP with Sci-kit learn, you will be able to build models with a greater freedom and ease.

Documentation.