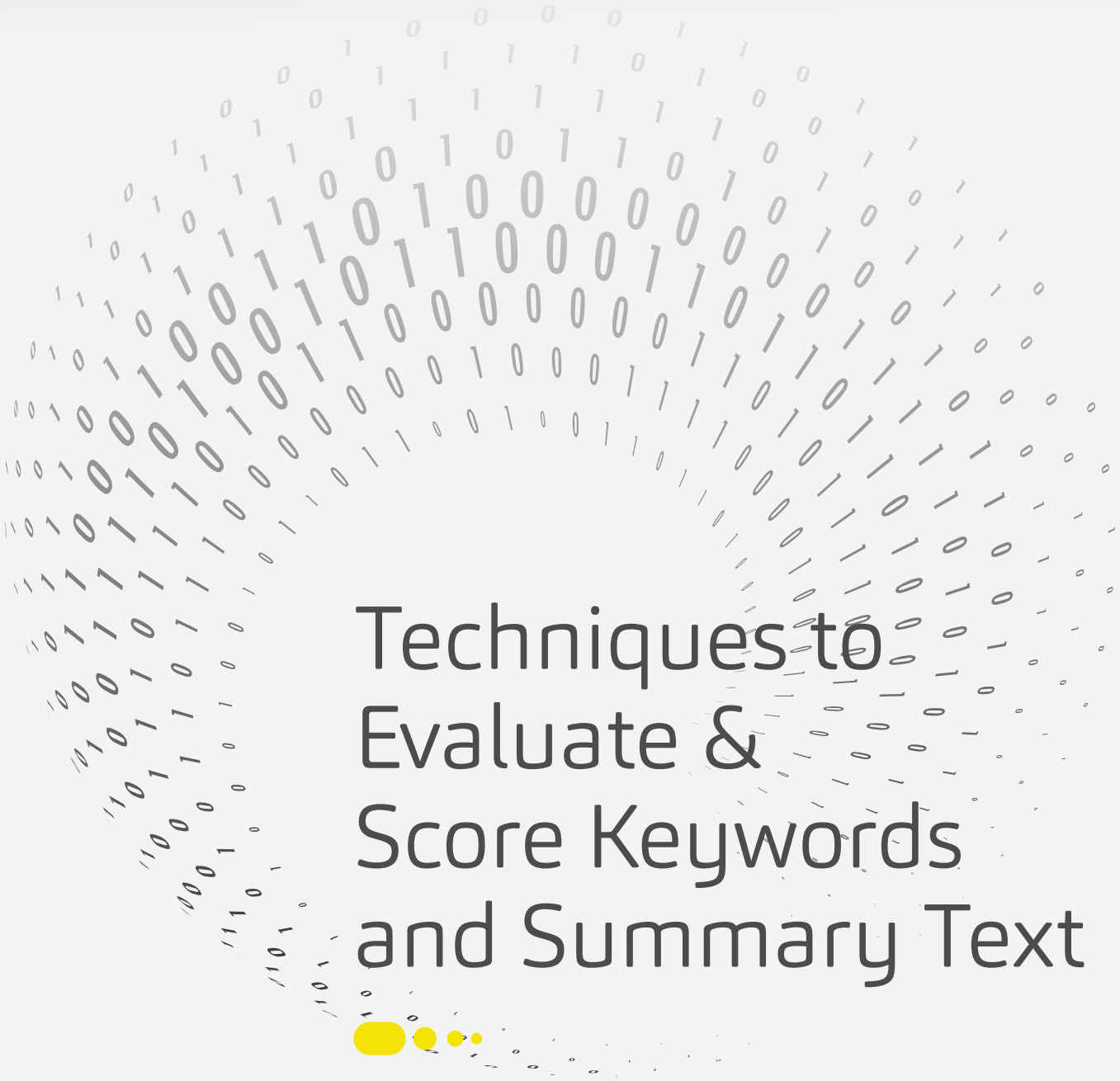
Synechron
Digital / Business Consulting / Technology



Techniques to Evaluate & Score Keywords and Summary Text

Authored by:

Anantha Sharma
Director - Technology
Data Scientist

Alekhya Akkinepally
Sr. Associate - Technology
Data Strategy and Services

python™

Scenario

Financial organizations amass large to a very large number of entries in their data dictionary over time and the upkeep of the definitions against the data elements is usually time-consuming and error-prone.

Problem Statement

The effort involved in keeping large data dictionaries is great and error-prone. Understanding the quality of the description against a specific data element is key to determine their completeness and effectiveness given their end users aren't always domain experts. This is especially true while dealing with domain-specific jargon and acronyms.

Solution

There are multiple techniques to identify and resolve this issue and one of the techniques (using machine learning) was built by Saumya Banthia and Anantha Sharma from the Synechron Innovation team (arXiv:2009.04953 [cs.CL]).

In this paper, we will discuss using standard Python packages to address the same problem statement.

The data source for our research is from PyPi (a package repository from Python with over 2 million entries), each follows the following pattern (note this is a subset of data available from PyPi)

Figure 1: PyPi Dataset

	Package	Release_version	Summary
1	hare	0.6	A python ORM based on pymysql with ActiveRecord
2	pylordeckcodes	1.0.0	A python implementation for the Legends of Run...
3	marcopolo.bindings	0.1.3	A python binding for MarcoPolo
4	televize	0.5	Script to play Czevch Television (Česk&#22...
5	daal-include	2020.0.133	Intel(R) Data Analytics Acceleration Library h...
6	django-listutils	1.0.0	Django template tag to split list into sub lists.
7	pycopy-cpython-uzlib	0.2	Pycopy module uzlib ported to CPython
8	apyref-puppet	0.1.1	Reserve names for current development items u...
9	django-eventtools	1.0.0	Recurring event tools for django
10	Tiaaer	0.1.1	Command-line tagging tool.

The characteristics of each summary can contain a description of the package along with technical keywords. Our work is to evaluate the readability and the complexity of each package summary and classify them into various classes including good, needs-information, poor, and unknown.

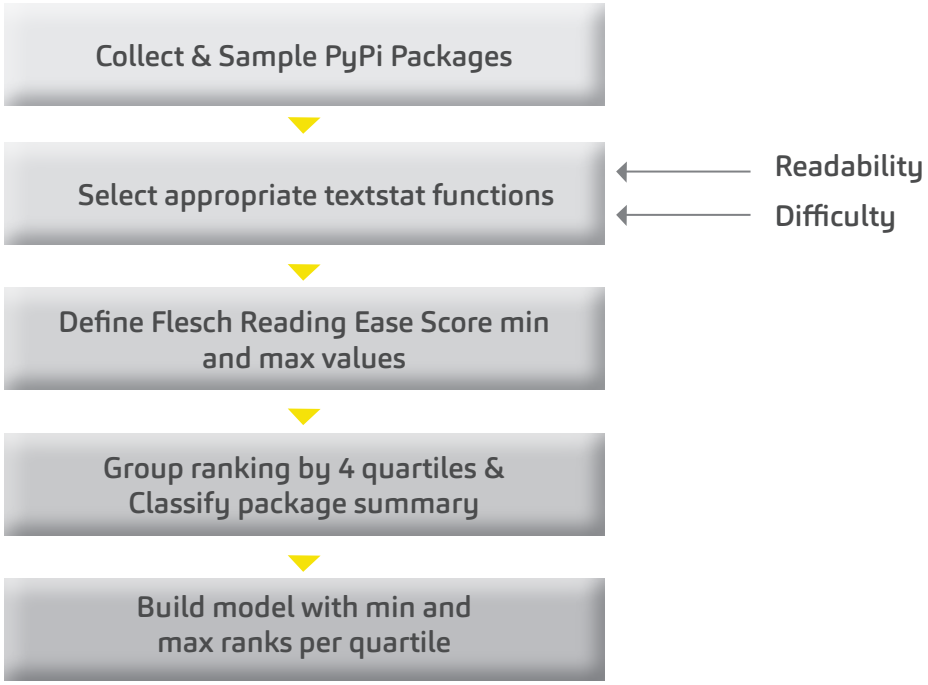
We ran text analytics on 1000 row dataset at each package summary level rather than across the package summary because each package is independent, as a rudimentary approach to evaluate the readability and complexity of the package summary and Python packages that perform well with the shorter sentences than a corpus of documents are selected.

In this article, our approach in analyzing the packages summaries using two Python text analytics packages is detailed, and a model to predict the classification for future package summaries is recommended and described in detail below.

PyTextRank and Textstat are both text analytics packages that excel in summarizing the phrases in a text and calculating the statistics to determine the readability and complexity.

1 Using Textstat

Figure 2: Textstat summary Ranking pipeline



Textstat has various functions used to determine the readability of the package summary. In our approach, the flesch reading ease, flesch kincaid grade, and textstat automated readability indexes are used.

Textstat has provided a very good description of how to classify a text based on the statistics it generated. There are multiple approaches in classifying our requirement of PyPi summary text, including classification based upon Textstat provided ranges or classifying by quartiles on Textstat provided statistics. In our approach, the latter is implemented.

Below are our high-level execution steps to classify the summary:

Flesch Statistics

By setting the English as the language, each package summary is iterated and statistics like Flesch reading ease, Flesch kincaid grade, and Textstat the automated readability index is calculated. We observed the Flesch reading ease statistic as the appropriate standard for our requirement because it has a wide range of statistic values and it determines the readability for a wide range of readers or individuals like professionals and graduates. Since Python is used extensively by both professionals and graduates, this could be the best statistic to use.

Quartiles as classification criteria

For the better distribution of summary ranks in the sample dataset, we have considered quartiles as an approach to divide the sample data into 4 groups for classification.

Classification

Then we group them at each classification level and obtain the minimum and maximum ranking value. A mapping is created with minimum and maximum value as the range and key as the desired classification. Finally, a model is established where for classifying future summary, its score is calculated using Flesch reading ease statistic and its class is determined based on the key and range mapping created above.

Drawbacks

The Textstat attempt had the following drawbacks:

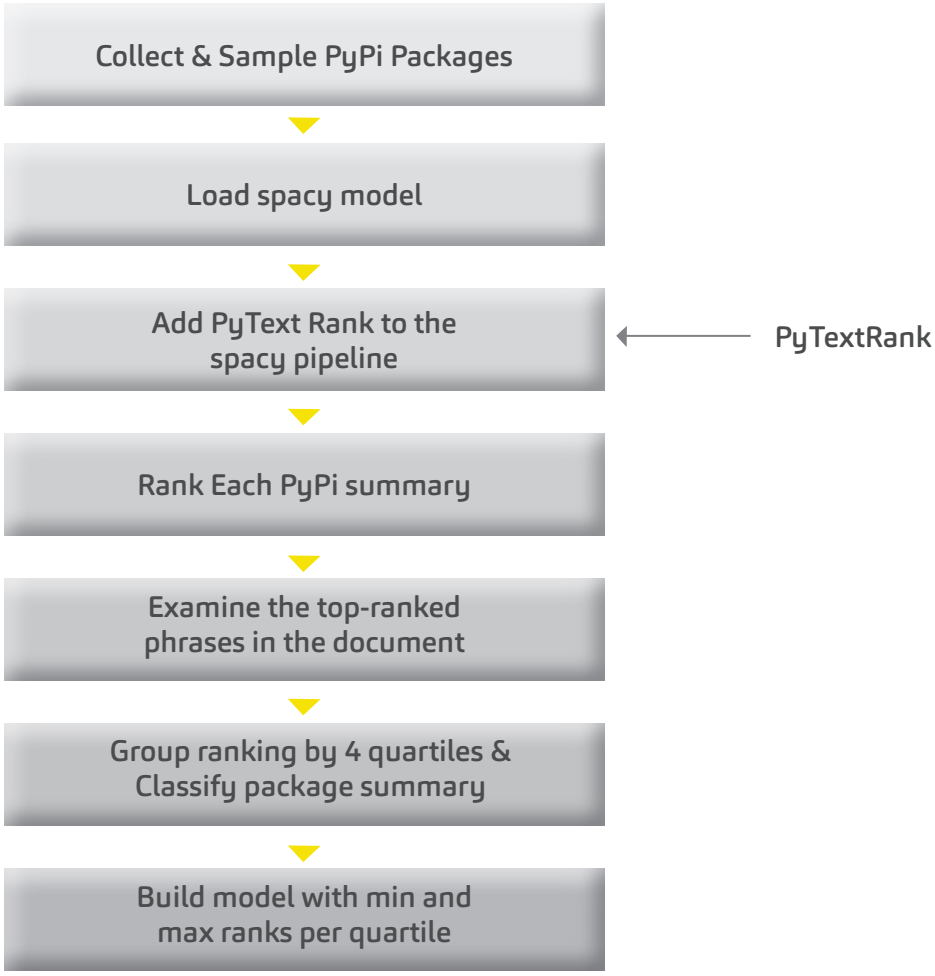
Although it has multiple functions, its primary focus is towards large corpus like books and their quality of readability, unlike the PyPi summary which are short descriptions.

It has various categorical values for various functions, so using it off-the-shelf requires a lot of source text preparation to meet its categorical values.

Textstat has a different scoring mechanism for different functions and is more applicable for different school grade levels than for professionals.

2 Using PyTextRank

Figure 3: PyTextRank Summary Ranking Pipeline



Preprocessing

PyTextRank natively used Spacy models and we have used the en core web sm 2.1.0 model. A pipeline is developed wherein scipy is loaded first, each package summary is iterated, and the ranking is determined for each summary with the below considerations. For the training data set we have considered a sample of PyPi packages.

Ranking

While PyTextRank ranks a sentence at the phrase level, we decided to sum all the ranks for a given summary text. PyTextRank provides ranking at phrase levels, where each phrase is identified by itself.

Quartiles

After ranking each summary, using the quartiles across the package's summary corpus, each summary was classified into good, needs-work, poor and unknown. They were then grouped at each classification level to obtain the minimum and maximum ranking value. A mapping is created with a minimum and maximum value as the range and key for the desired classification.

Classification

Finally, a model is established where for classifying future summary, its text rank is calculated using PyTextRank and its class is determined based on the key and range mapping created above.

Evaluation and Results

Evaluating a classification model to find the correct label is important and it can be done using a confusion matrix.

A confusion matrix contains information about the actual and predicted values (column and row). This matrix provides four possible outcomes:

TP = True Positive , FP = False Positive , FN = False Negative, and TN = True Negative. The other metrics that can be evaluated are F1-score, Precision, and Recall. The classification report obtained through our PyTextRank algorithm is shown below.

Figure 4: Evaluation Report

Classification	Precision	Recall	F1-score	Support
Good	0.87	0.95	0.91	21
Needs-information	0.77	0.85	0.81	20
Poor	1	0.72	0.84	18
Unknown	0.96	1	0.98	23

Figure 5: Metrics Report

	Precision	Recall	F1-score	Support
Accuracy			0.89	82
Macroavg	0.9	0.88	0.88	82

This is an initial approach and can be extended to apply Logistic Regression instead of quartiles to build a classification model and predict the readability and complexity of the PyPi package summary text.

Alternatively, we can optimize PyTtext and Textstat hyperparameters for better phrase identifications and word embeddings. We also need to consider transforming or cleaning some of the text that has a package name followed by a special character which is currently reducing its rank in the above approach. Overall, PyText and Textstat have phrase analytics tools that are performing as required.

Challenges

1. Some of the summaries are not comprehensive and not simple.
2. The readability and simplicity of each summary can be improved using a ranking-based approach that helps the summary/text to outline or summarize their features using important points or keywords. Such improvisation should help in making the context better.
3. This approach can be extended to legal documents or product descriptions that need better readability and enhanced simplicity.

Anantha Sharma
Director - Technology
Data Scientist

Alekhya Akkinepally
Sr. Associate - Technology
Data Strategy and Services

Global Footprint

Synechron

Digital / Business Consulting / Technology

www.synechron.com
info@synechron.com