PyPandas: A scalable data cleaning library

Pei-Lun Liao
New York University
pll273@nyu.edu

Chia-Hsien Lin
New York University
chl566@nyu.edu

Shang-Hung Tsai
New York University
st3127@nyu.edu

ABSTRACT

Data cleaning become a challenge[2] in data mining and machine learning tasks. In this paper, we propose PyPandas, a scalable library running on Spark[17] to solve data cleaning tasks. PyPandas provides basic cleaning features such as missing values handling, outlier detection and data scaling. Morevoer, advanced features like text cleaning are supported as well.

1 INTRODUCTION

In recent years, as storage devices become cheaper and cheaper, storing big data in thousands of computers is easier than before. People start to think how to obtain big value from the huge dataset. The term, Big Data, was created to describe this phenomenon[?]. As the result, data mining[5, 13] and machine learning[?] get popular nowadays. The quality of data is one of the key points to mine good value in the huge dataset[?]. Hence, data cleaning become a challenge in the first step of data management and analysis[2, 6, 14].

To process huge dataset efficiently, distributed systems and parallel computing frameworks[3, 4, 16] are introduced. Spark[17] is one of the most popular open source projects for industry and academia. It is built on Hadoop[16] and provides a way to manage distributed memory. PySpark[10] is an extended library for Python programmers.

In the Python machine learning and data mining ecosystem, Pandas[9] is the most popular data management library. Pandas dataframe provides a way to collect data and the data can be transformed from Python primitive data structures or Numpy[11] array easily. Moreover, Pandas provides several data management methods such as missing value filling, data indexing and data profiling.

However, Pandas can only be used in a single machine and could not be scaled to manage huge dataset. In this paper, we propose PyPandas, a library built on PySpark, which support basic data management features, missing value handling or duplicated data removing. Moreover, we add common data processing features into our library to provide user friendly usage such as outlier detection and numerical data scaling. These features are important for the machine learning tasks and are supported in Scikit-Learn[12], a mainstream machine learning library, as well.

Furthermore, text data is another common data type in our storage. Text data are hard to process because of multiple languages, newly invented word, typo, abbreviation, url and emoji etc. Although we have NLTK[1], a popular text cleaning library, there is no scalable text cleaning library. In this paper, we also provide tools for users to clean text easily.

The features of our library is summarized below.

- Saclable data management library runs on Spark
- Missing value handling, duplicated data removing, outlier detection and numerical data scaling
- Text processing such as url detection, punctuation removing, pattern searching and replacement.

2 RELATED WORK

There exists a number of open-source library for data cleaning and parallel data computing.

- Optimus[7] is a framework that can perform distributed data cleaning and preprocessing. This framework works well with Spark and its DataFrame can scale in big cluster. Optimus comes with helpful tools such as removing special characters and replacing null values.
- Dask[15] is another open-source library that supports parallel analytic computing. It provides scalable parallel data structures that extend interfaces like pandas. At the same time, it offers low latency and high responsiveness.
- SparklingPandas[8] attempts to combine the power of spark and pandas to scale data analysis. It provides a Pandas-like API that is built using Spark's DataFrame class. Unfortunately, it only support spark v1.4 and Python 2.7, and its development has ended.

3 PROPOSED METHODS

In this project, we prepare to build our library on top of the PySpark[10, 17]. We will handle RDD data structure and implement useful data cleaning fucntions in our library. The ways to do that are collecting common cleaning functions into library and customizing map functions to do advanced cleaning task such as pattern searching and replacement with regular expression. We will survey the Pandas[9] and Scikit-Learn[12] libraries to find out key features and create Pandas-liked and Scikit-Learn-liked API to let users start with easily.

4 EXPERIMENT

4.1 Dataset

Dataset 1: 311 Service Requests from 2010 to Present

- Summary: All 311 Service Requests from 2010 to present.
- Source URL: https://data.cityofnewyork.us/Social-Services/311-Service-Requests-from-2010-to-Present/erm2-nwe9
- Data property: text is majority
- Rows: 9,063,486, Columns: 53
- Size: 6.01G
- Missing value: There are about 15 out of 53 columns with null value which is about 28.3%
- Reason to choose the data: In this dataset, most of the data is consisted of alphabetical words which is one of the data property we are looking for. Besides, the column named "Resolution Description" contains the description about the service status notes which are all alphabetical words we need.

Dataset 2: Statistical Summary Period Attendance Reporting (PAR)

- Summary: Statistical report on attendance by borough, grade. Alternate views of same data by grade level and enrollment (register). All students including YABC, adults, LYFE babies and charters, home instruction, home/hospital, CBO UPK.
- Source URL: https://data.cityofnewyork.us/Education/Statistical-Summary-Period-Attendance-Reporting-PA/hrsu-3w2q
- Data property: number is majority
- Rows: 24,421, Columns: 39
- Size: 2.8Mb
- Missing value: There are about 5 out of 39 columns with null value which is about 13%
- Reason to choose the data: In this dataset, most of the data is consisted of numbers which is also one of the data property we are looking for.

Dataset 3: DOB Job Application Filings

- Summary: A list of job applications filed for a particular day and associated data. Prior weekly and monthly reports are archived at DOB and are not available on NYC Open Data.
- Source URL: https://data.cityofnewyork.us/Housing-Development/DOB-Job-Application-Filings/ic3t-wcy2
- Data Property: categorical dataset
- Rows: 5,260,437, Columns: 89
- Size: 2.88G
- Missing Value: There are about 36 out of 89 columns with null value which is about 40%.
- Reason to choose the data: In this dataset, there are 89 columns which satisfies the categorical property we need in this project.

4.2 Evaluation

We will evaluate our project by comparing it against other existing open-source libraries, including Optimus[7], Dask[15], and SparkingPandas[8]. We will compare our data cleaning features with Optimus, particularly those frequently used functionality in data science such as null value handling, duplicate detection, etc. Furthermore, we will evaluate the performance of our implementation of parallel data management by measuring the time and space consumed because it is important to have an efficient implementation to make this library useful in production. We will also examine and compare the overall software architecture of the project. We might perform a survey to assess the usability of the library.

4.2.1 Features.

4.2.2 Performance.

4.2.3 Userbility.

5 CONCLUSIONS

Due to the lack of useful scalable data cleaning library on Spark, we propose PySpark, a scalable data cleaning library. We provide basic data cleaning features such as missing value handling, duplicated data removing and data scaling. Moreover, our library supports text cleaning feature with regular expression.

6 REFERENCE

REFERENCES

[1] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python* (1st ed.). O'Reilly Media, Inc.

[] Li Cai and Yangyong Zhu. 2015. The Challenges of Data Quality and Data Quality Assessment in the Big Data Era. *Data Science Journal* 14, 2 (2015), 1–10. https://doi.org/10.5334/dsj-2015-002

[2] Xu Chu, Ihab F. Ilyas, Sanjay Krishnan, and Jiannan Wang. 2016. Data Cleaning: Overview and Emerging Challenges. In *Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16)*. ACM, New York, NY, USA, 2201–2206. https://doi.org/10.1145/2882903.2912574

[3] Jeffrey Dean and Sanjay Ghemawat. 2008. MapReduce: Simplified Data Processing on Large Clusters. *Commun. ACM* 51, 1 (Jan. 2008), 107–113. https://doi.org/10.1145/1327452.1327492

[4] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. 2003. The Google File System. In *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles (SOSP '03)*. ACM, New York, NY, USA, 29–43. https://doi.org/10.1145/945445.945450

[5] Jiawei Han, Micheline Kamber, and Jian Pei. 2011. *Data Mining: Concepts and Techniques* (3rd ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[6] Ihab F. Ilyas and Xu Chu. 2015. Trends in Cleaning Relational Data: Consistency and Deduplication. *Foundations and TrendsÂő in Databases* 5, 4 (2015), 281–393. https://doi.org/10.1561/1900000045

[7] Iron. 2018. Optimus. (2018). https://github.com/ironmussa/Optimus

[8] Holden Karau and Juliet Hougland. 2015. SparklingPandas. (2015). https://github.com/sparklingpandas/sparklingpandas

[] A. Katal, M. Wazid, and R. H. Goudar. 2013. Big data: Issues, challenges, tools and Good practices. In *2013 Sixth International Conference on Contemporary Computing (IC3)*. 404–409. https://doi.org/10.1109/IC3.2013.6612229

[9] Wes McKinney. [n. d.]. pandas: a Foundational Python Library for Data Analysis and Statistics. ([n. d.]).

[10] Amit Nandi. 2015. *Spark for Python Developers*. Packt Publishing.

[11] Travis E. Oliphant. 2015. *Guide to NumPy* (2nd ed.). CreateSpace Independent Publishing Platform, USA.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[13] Anand Rajaraman and Jeffrey David Ullman. 2011. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA.

[14] Vijayshankar Raman and Joseph M. Hellerstein. 2001. Potter's Wheel: An Interactive Data Cleaning System. In *Proceedings of the 27th International Conference on Very Large Data Bases (VLDB '01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 381–390. http://dl.acm.org/citation.cfm?id=645927.672045

[15] Matthew Rocklin. 2015. Dask: Parallel Computation with Blocked algorithms and Task Scheduling. In *Proceedings of the 14th Python in Science Conference*, Kathryn Huff and James Bergstra (Eds.). 130 – 136.

[] Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA.

[16] Tom White. 2009. *Hadoop: The Definitive Guide* (1st ed.). O'Reilly Media, Inc.

[17] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: Cluster Computing with Working Sets. In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud'10)*. USENIX Association, Berkeley, CA, USA, 10–10. http://dl.acm.org/citation.cfm?id=1863103.1863113