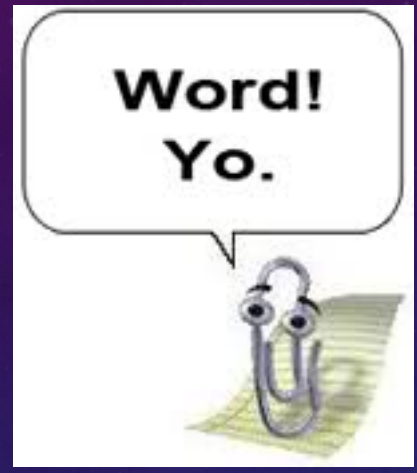


LITERARI.LY

A LITERARY ASSISTANT

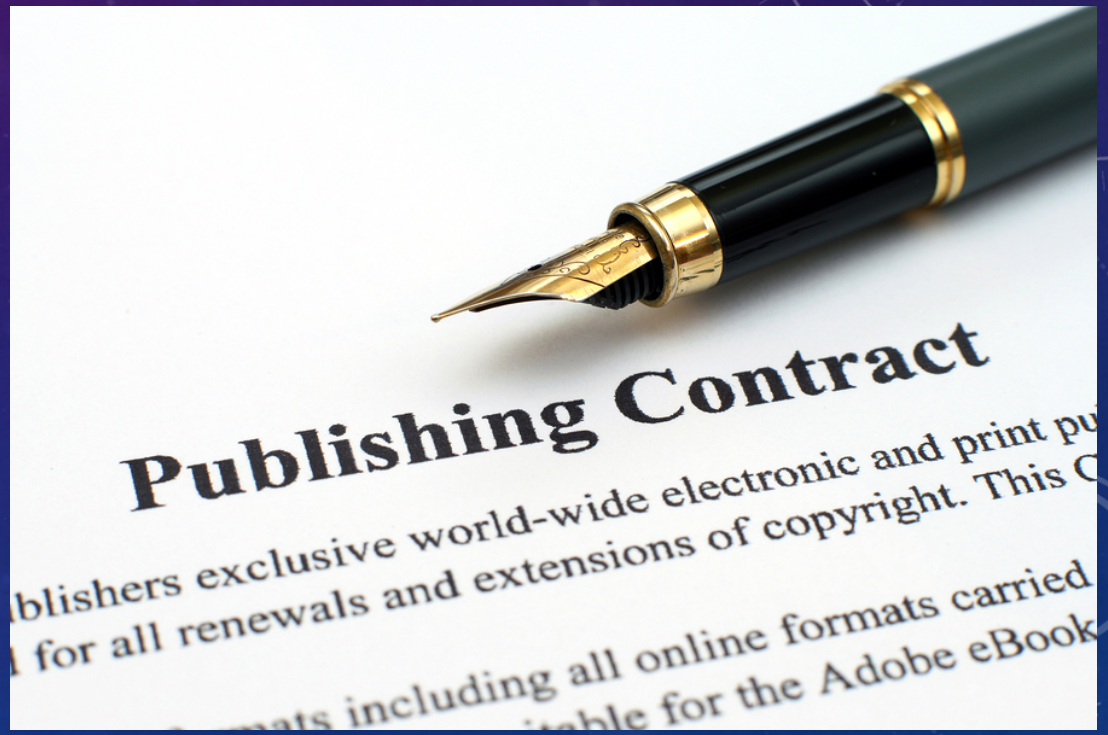
-ADAM LENART, ANKITH GUNAPAL & SANDIP PALIT

PROBLEM STATEMENT

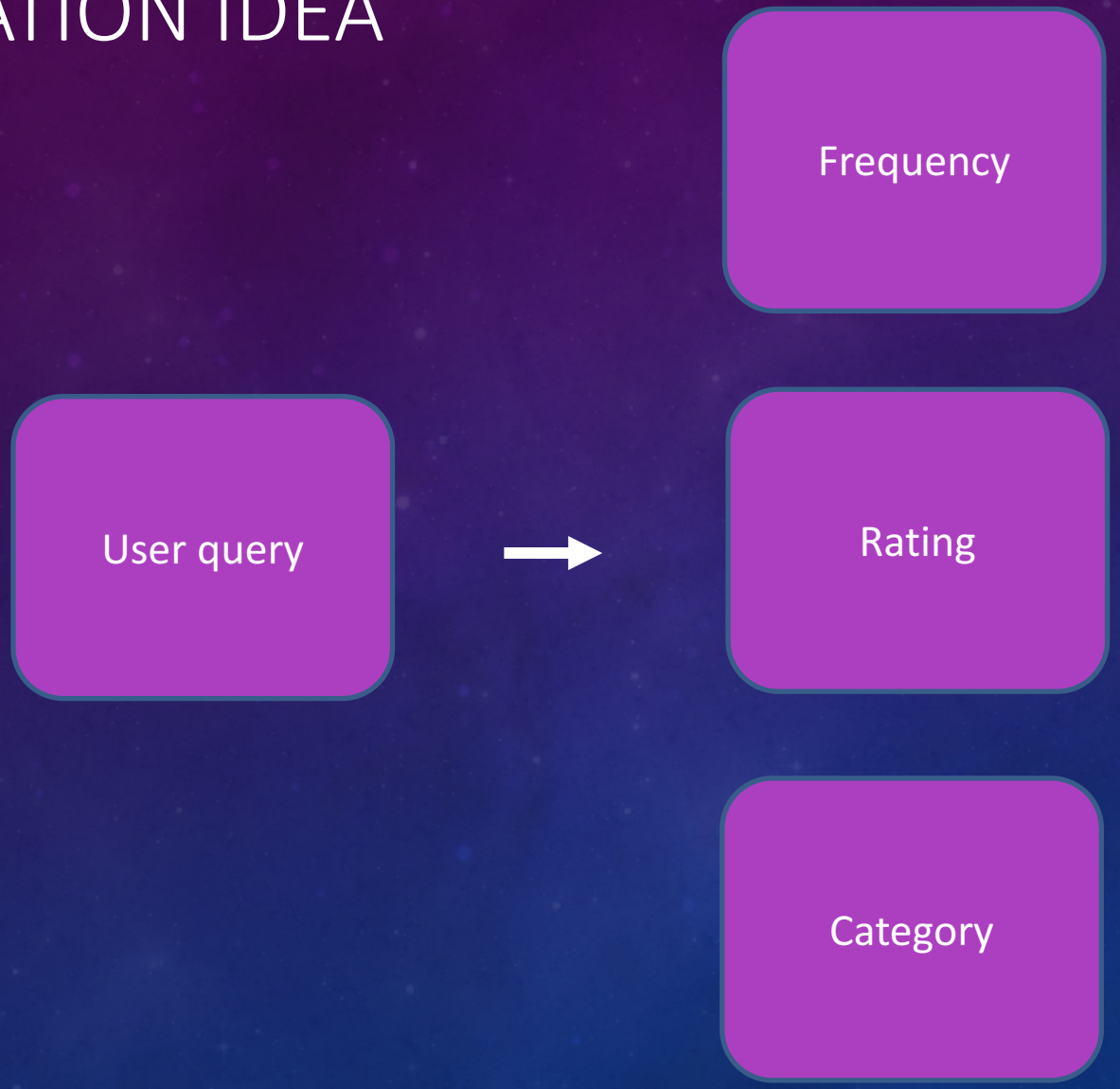
How do you write a successful piece of literature?

Helping authors choose the best expression
to reach their target audience

- Books
- Blogs
- Technical Publications



APPLICATION IDEA



DATA SOURCES

Frequency

Google N-grams

Rating, category

Google Books + Good Reads

- Google n-grams dataset: fixed dataset with the frequency of 1 to 5 word expressions in published books and magazines over time from about 1500 until now.
- Google Books API : text searches returning bibliographical information
- Good Reads API: reviews, ratings, tags of books

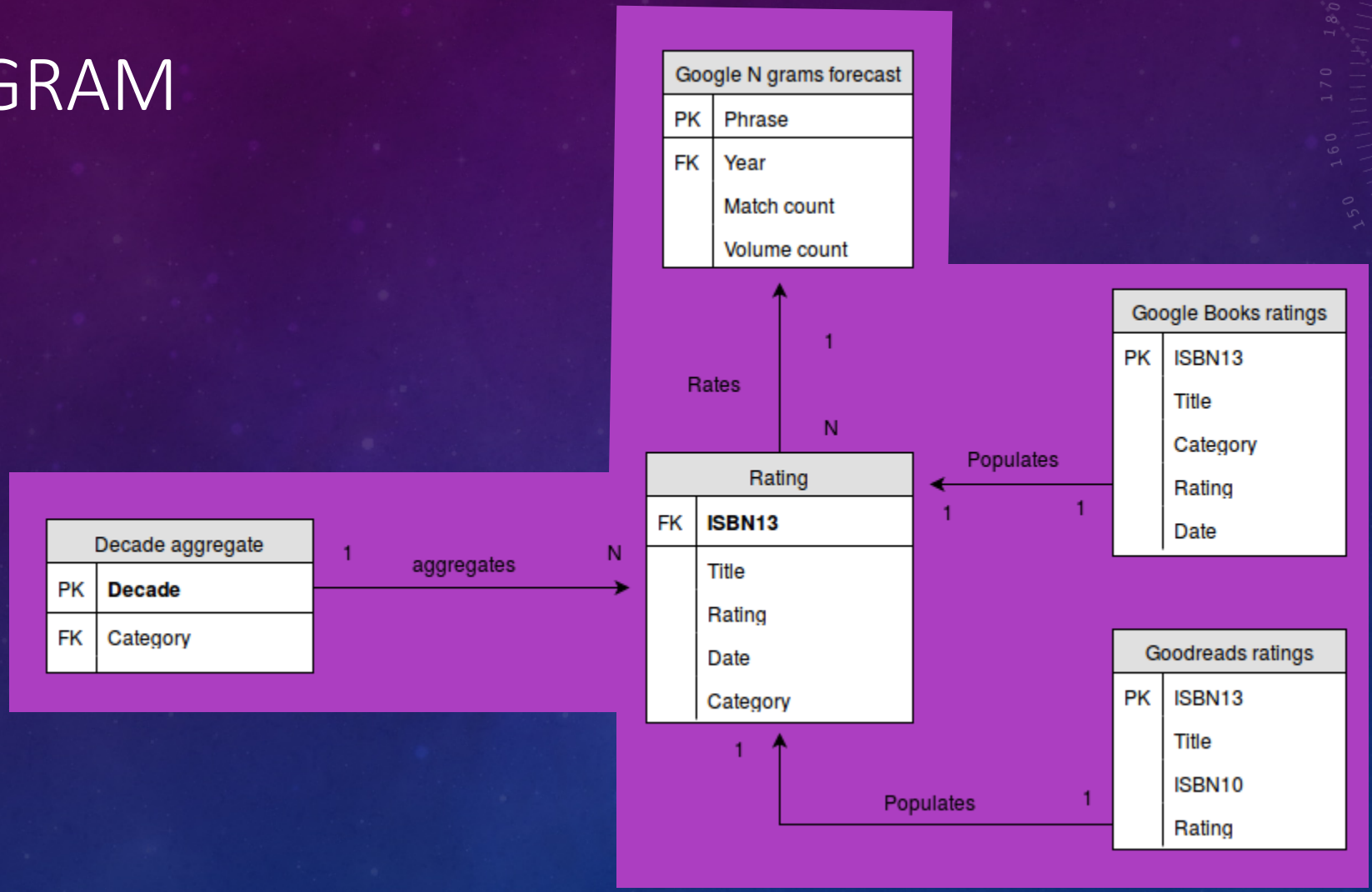
VOLUME * VARIETY * VELOCITY

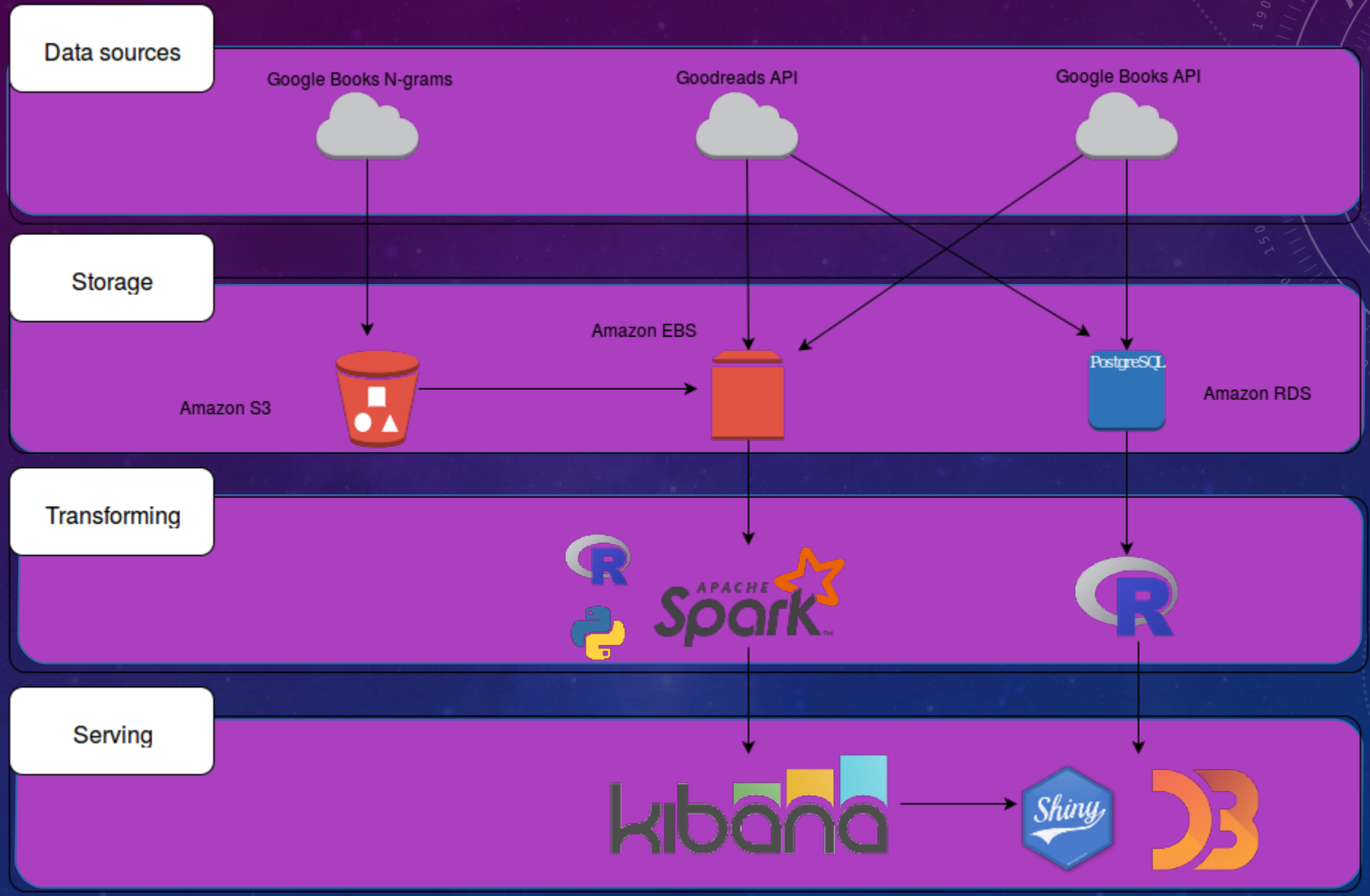
Volume:	*Google N-gram: 3.5 TB data stored in our S3 bucket (only English bigrams) * we used a 15 GB subset of it for bigrams starting with “AB...”
Variety:	* JSONs from Google Books, GoodReads API calls * Google N-gram CSVs
Velocity:	* Kibana dashboard refreshes search history every 5 seconds
Services used:	* Amazon EC2 * Amazon S3 * Amazon RDS * Amazon ES Services
Tools used:	* Apache Spark (PySpark, SparkR) * PostgreSQL * ElasticSearch, Kibana * Shiny, D3 * Python, R

DATA CHALLENGES

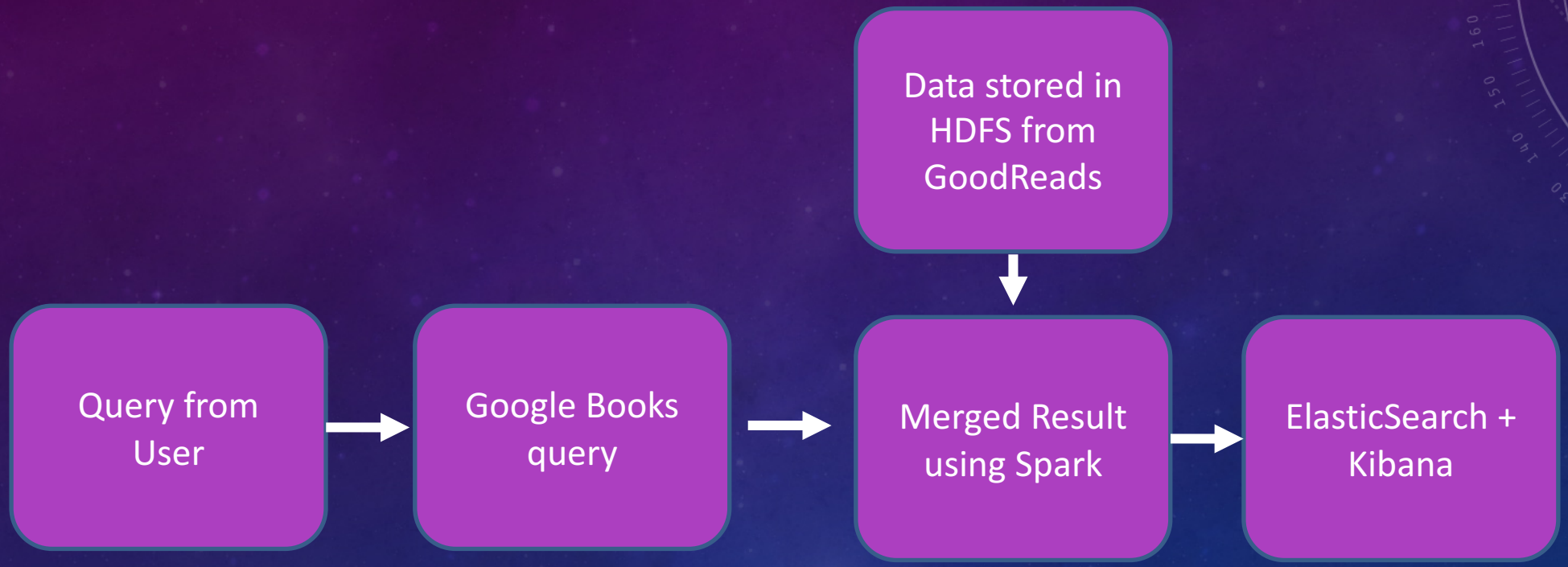
- Thousands of compressed CSV files in Google N-gram database, with some files taking 2-3 GB of space
- Good Reads has a restriction of 1 API call per second
- Filtering data from Good Reads such that we get Ratings for books written in English with a valid ISBN
- Filtering data from Google Books such that we get Ratings, ISBN and valid category for books .
- Amazon S3 maintains the Google N-gram database in an encoded format, requires building Hive/Spark with LZOCodec.

ER DIAGRAM


Google N grams forecast
PK Phrase
FK Year
Match count
Volume count
Rates
1
N
Google Books ratings
PK ISBN13
Title
Category
Rating
Date
Populates
1
1
Rating
FK ISBN13
Title
Rating
Date
Category
Decade aggregate
PK Decade
FK Category
1
aggregates
N
1
Populates
1
Goodreads ratings
PK ISBN13
Title
ISBN10
Rating


Data sources
Google Books N-grams
Goodreads API
Google Books API
Storage
Amazon EBS
Amazon S3
PostgreSQL
Amazon RDS
Transforming
R
APACHE
Spark
R
Serving
kibana
Shiny
D3

ARCHITECTURE * KIBANA DASHBOARD



~/ShinyApps/literally - Shiny - Mozilla Firefox

ec2-54-89-238-69.compute-1.amazonaws.com/?view=shiny

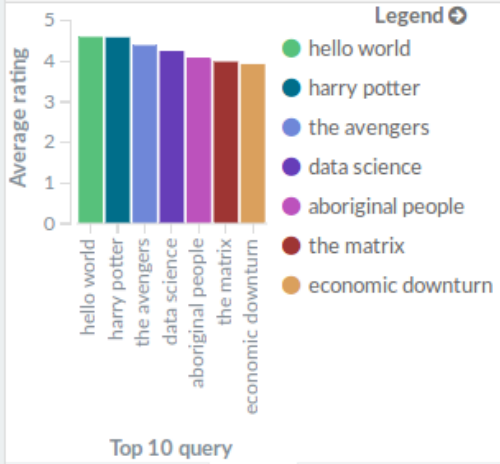
http://ec2-54-89-238-69.compute-1.amazonaws.com/p/6565/ | Open in Browser | Publish

Literari-ly

Visualizations

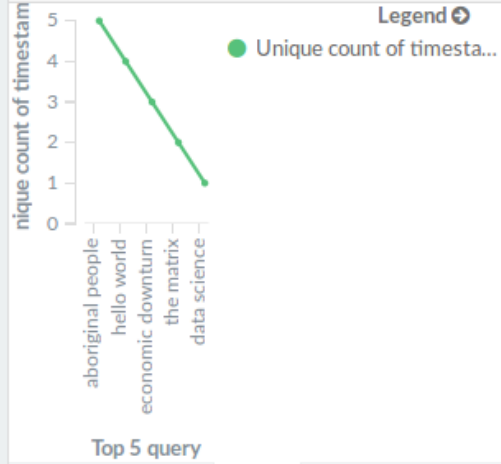
- Forecast
- Network
- Treemap
- Ratings
- Kibana

Average Rating



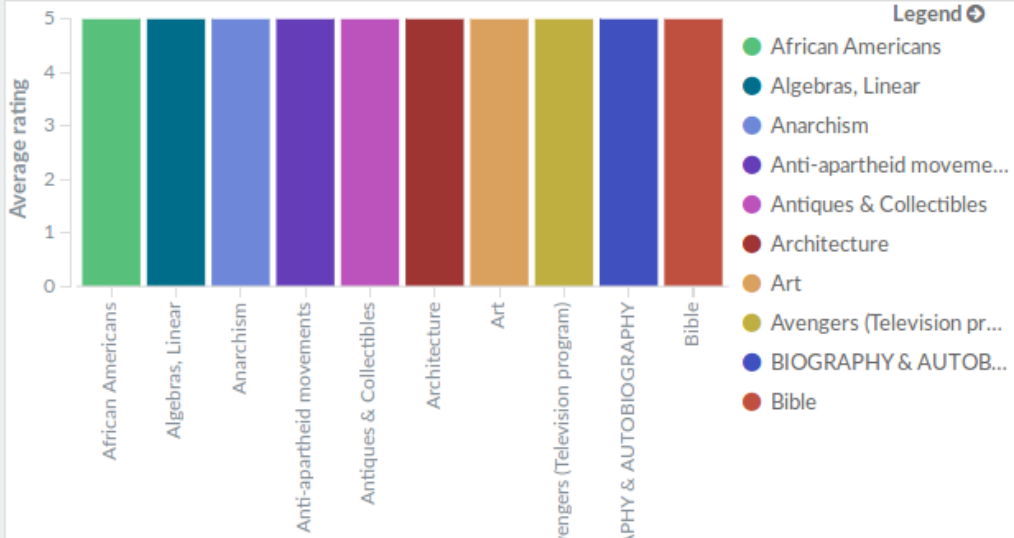
Top 10 query

Query string

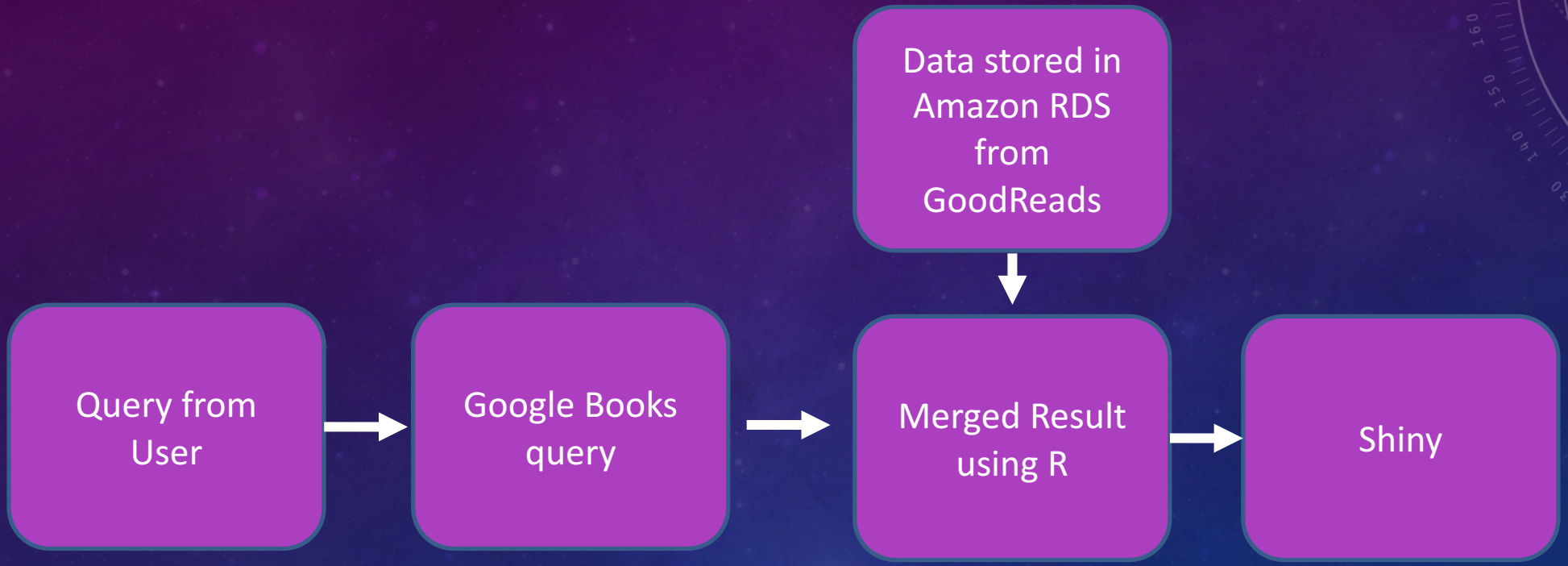


Top 5 query

Rating per category



ARCHITECTURE * RATING AND CATEGORIES



~/ShinyApps/literally - Shiny - Mozilla Firefox

ec2-54-89-238-69.compute-1.amazonaws.com/?view=shiny

http://ec2-54-89-238-69.compute-1.amazonaws.com/p/6565/ | Open in Browser | Publish

Literari-ly

Visualizations

- Forecast
- Network
- Treemap
- Ratings
- Kibana

Links between categories and decades of publication

By default, the 5 highest frequency categories are plotted but the data will likely have many more.

Search expression

aboriginal people

Search

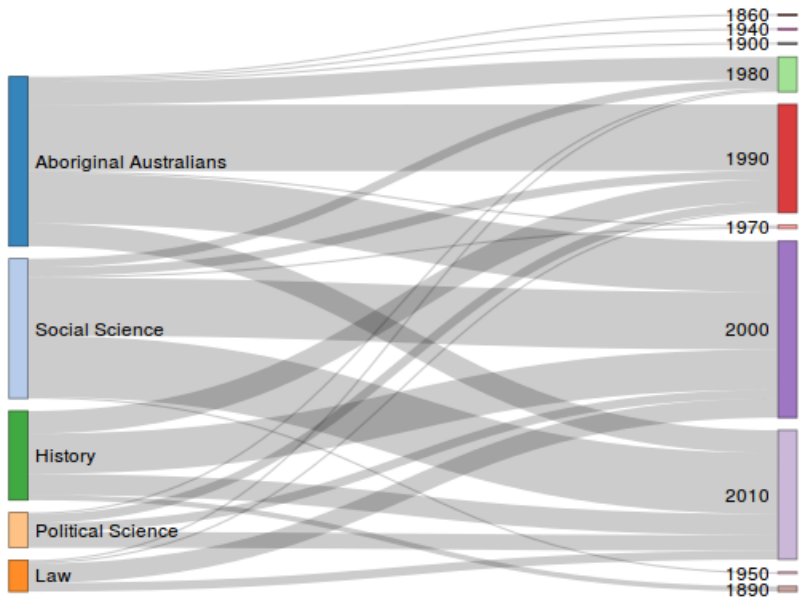
Font

1 15 20

1 3 5 7 9 11 13 15 17 19 20

Number of most frequent categories to keep

5



~/ShinyApps/literally - Shiny - Mozilla Firefox

ec2-54-89-238-69.compute-1.amazonaws.com/?view=shiny

http://ec2-54-89-238-69.compute-1.amazonaws.com/p/6565/ | Open in Browser | Publish

Visualizations

- Forecast
- Network
- Treemap
- Ratings
- Kibana

Frequency of published books in a given category per decade

Search expression

aboriginal people

Search

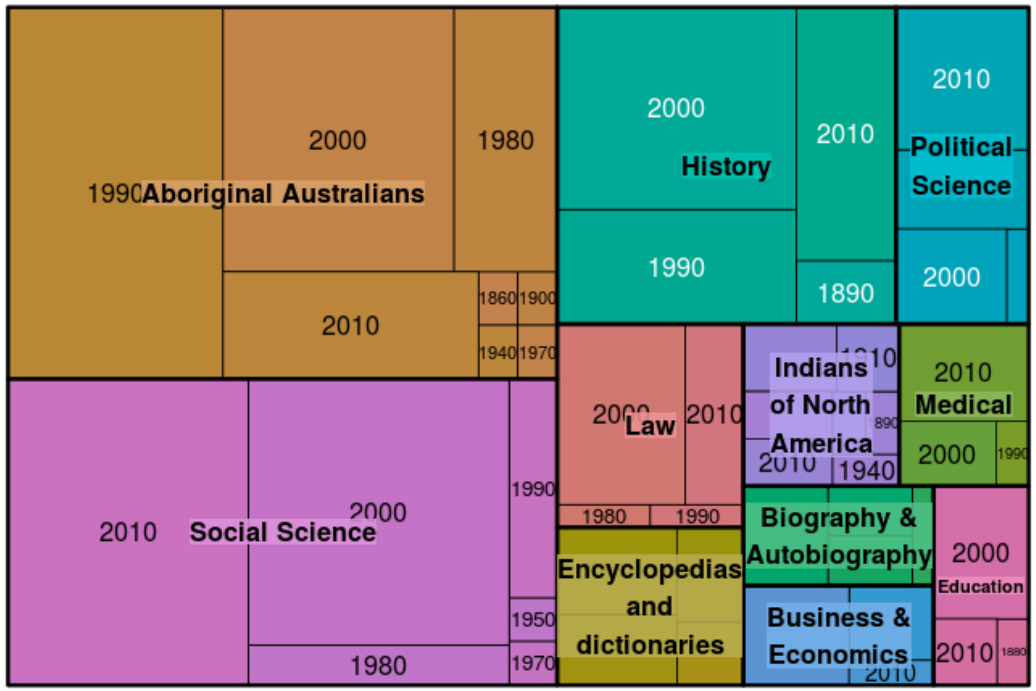
Font size

1 15 20

1 3 5 7 9 11 13 15 17 19 20

Number of most frequent categories to keep

10



~/ShinyApps/literally - Shiny - Mozilla Firefox

ec2-54-89-238-69.compute-1.amazonaws.com/?view=shiny

http://ec2-54-89-238-69.compute-1.amazonaws.com/p/6565/ | Open in Browser | Publish

Literari-ly

Visualizations

- Forecast
- Network
- Treemap
- Ratings
- Kibana

Ratings of books by categories

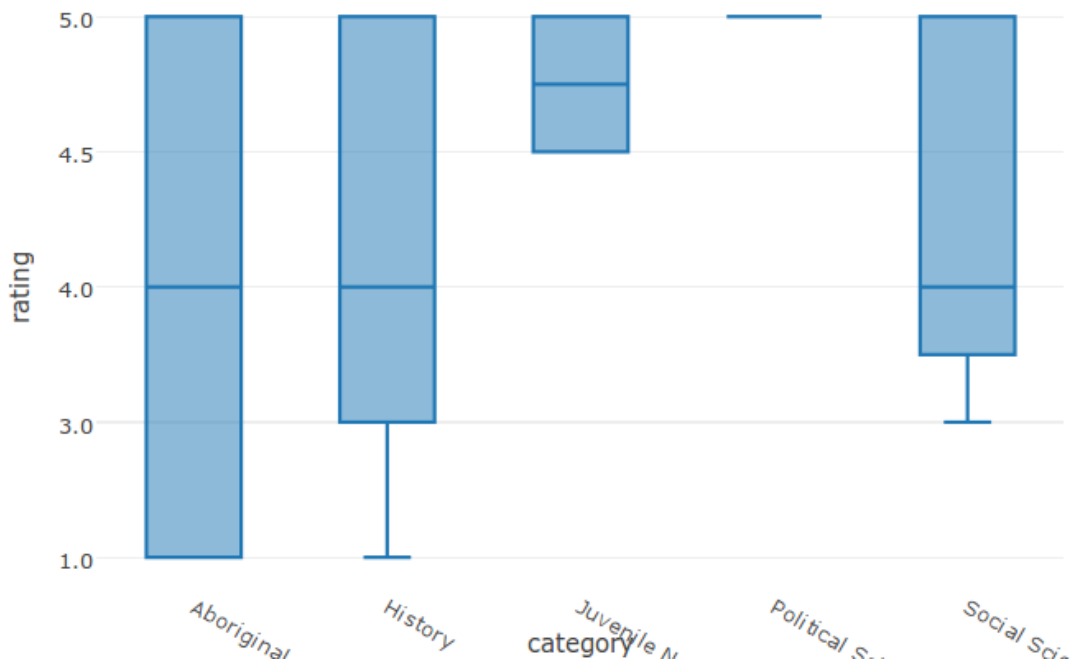
Search expression

aboriginal people

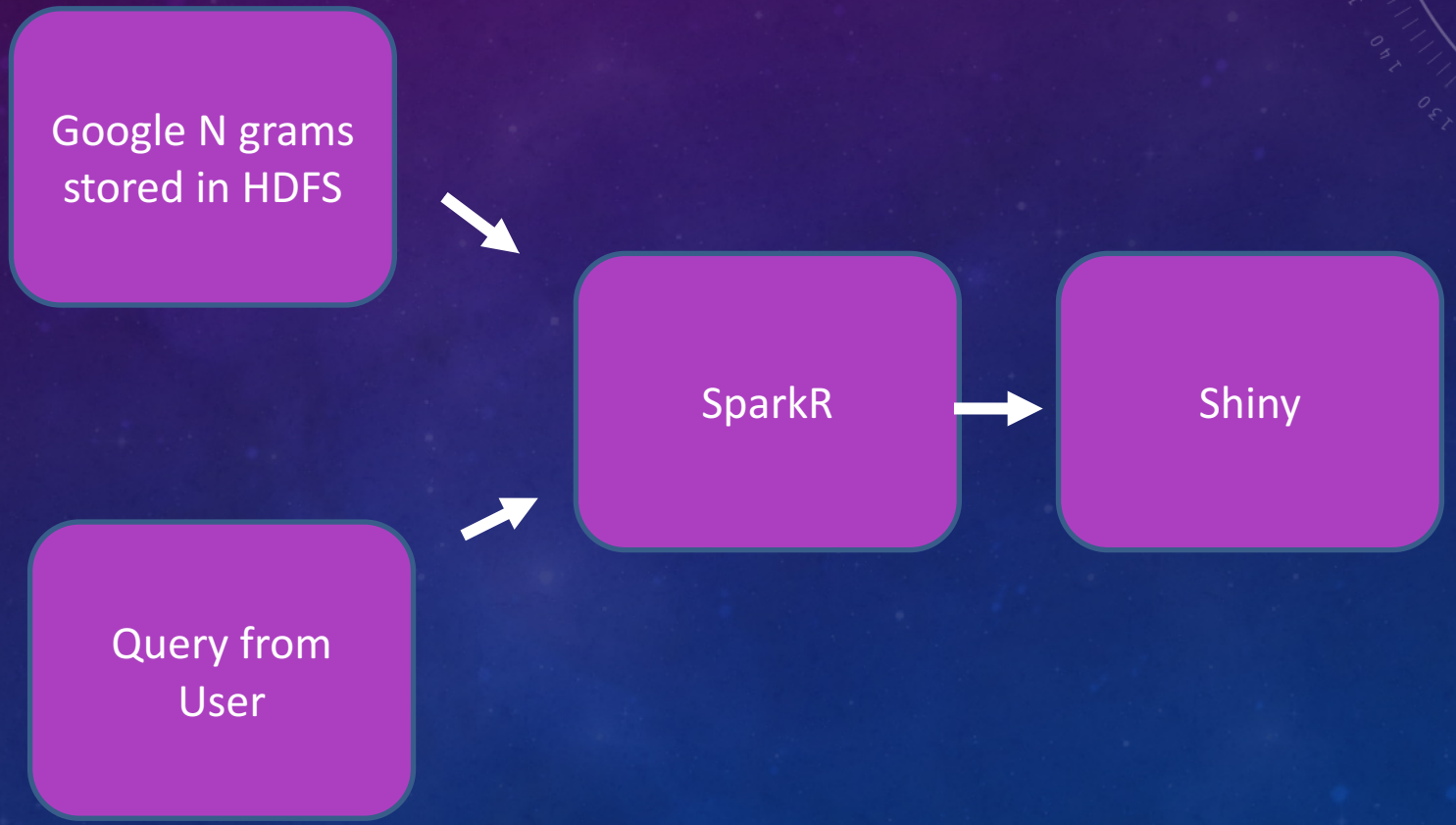
Search

Number of most frequent categories to keep

5



ARCHITECTURE * FORECASTING



~/ShinyApps/literally - Shiny - Mozilla Firefox

ec2-54-89-238-69.compute-1.amazonaws.com/?view=shiny

http://ec2-54-89-238-69.compute-1.amazonaws.com/p/6565/ | Open in Browser | Publish

Literari-ly

Visualizations

- Forecast
- Network
- Treemap
- Ratings
- Kibana

Forecast popularity of an expression

Search expression

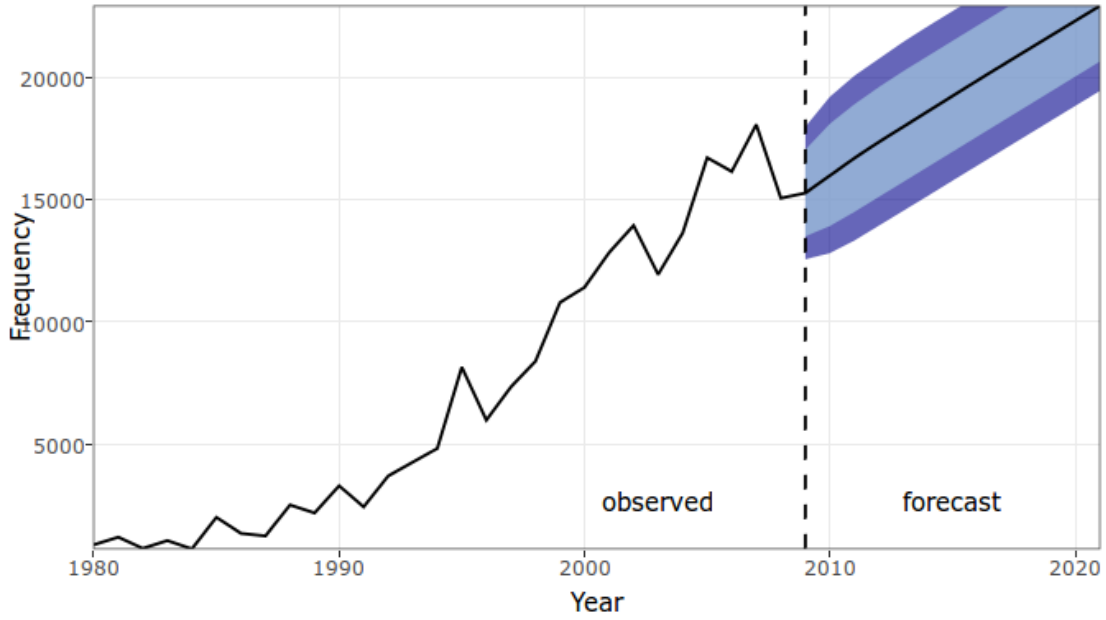
aboriginal people

Role of first word

Adjective

- Adjective
- Adverb
- Noun
- Verb

Search



FUTURE WORK

- The expressions could be queried in the Twitter API and return the distribution of user interests who tweeted the queried expressions.
- Example implementation of finding out user interests at scale:

  http://www.mpi-sws.org/~mzafar/papers/recsys14-userinterests.pdf
  http://twitter-app.mpi-sws.org/who-likes-what/