

PandaRec - A Recipe-Based Recommendation System for the Pandas Library

Frederick Vandermoeten

Overview

- **Introduction**
- Approach
- Evaluation
- Conclusion

Pandas

- Python library for data manipulation and analysis
- Works with DataFrames
- Existing recommendation systems don't work well with Pandas

Code Recommendation Systems

- Help developers to write code
- Design dimensions:
 - Input
 - Recommendation Engine
 - Output
 - User Feedback

Existing Code Recommendation Systems

- Code completion
 - Part of most IDEs
 - not that useful for Pandas
- Code generation
 - Boilerplate automation
 - Generative AI
 - ChatGPT
 - Copilot

Overview

- Introduction
- **Approach**
- Evaluation
- Conclusion

My Solution

- Jupyter Notebook
- Recipe-based recommendation system
- Swappable recommendation engine

Jupyter Notebook

Options

key	...pow	Origin	...Gall	Name	...erati	Year	...in_lt	...lnde	...eme
0	130	USA	18	chevro...	12	1970-0...	3504	8	307
1	165	USA	15	buick s...	11.5	1970-0...	3693	8	350
2	150	USA	18	plymo...	11	1970-0...	3436	8	318
3	150	USA	16	amc re...	12	1970-0...	3433	8	304
4	140	USA	17	ford to...	10.5	1970-0...	3449	8	302
5	198	USA	15	ford ga...	10	1970-0...	4341	8	429
6	220	USA	14	chevro...	9	1970-0...	4354	8	454
7	215	USA	14	plymo...	8.5	1970-0...	4312	8	440
8	225	USA	14	pontia...	10	1970-0...	4425	8	455
9	190	USA	15	amc a...	8.5	1970-0...	3850	8	390
10	115	Europe	NaN	citroen...	17.5	1970-0...	3090	4	133
11	165	USA	NaN	chevro...	11.5	1970-0...	4142	8	350
12	153	USA	NaN	ford to...	11	1970-0...	4034	8	351
13	175	USA	NaN	plymo...	10.5	1970-0...	4166	8	383

Search: remove missing values

dropna

0.55

dropna

Copy

Return a new Series with missing values removed.

See the :ref:`User Guide <missing_data>` for more on which values are considered missing, and how to work with missing data.

Parameters

axis : {0 or 'index'}
 Unused. Parameter needed for compatibility with DataFrame.
inplace : bool, default False
 If True, do operation inplace and return None.
how : str, optional
 Not in use. Kept for compatibility.

notna

0.49

notna

Copy

Detect existing (non-missing) values.

isna

0.47

isna

Copy

Detect missing values.

pad

0.45

pad

Copy

Synonym for :meth:`DataFrame.fillna` with ``method='ffill'``.

backfill

0.45

backfill

Copy

Synonym for :meth:`DataFrame.fillna` with ``method='bfill'``.

ne

0.44

ne

Copy

Return Not equal to of series and other, element-wise (binary operator `ne`).

Recipes

- A data structure that describes a task
- Contains:
 - Name
 - Code snippet
 - Description
- saved in JSON format
- Generated from: `dir()` function, existing code snippets, hand-written

The Recommendation Engine

- Strategy pattern
- Gets current context
- Search function that returns a ranked list of recipes

Ranking Strategies: Lexical Search

- Name Search
- Fuzzy Search
 - Levenshtein distance
- Index Search
 - Lemmatize words
 - Build an inverted index

Ranking Strategies: Semantic Search

- Use a NLP model to calculate the similarity between the query and the recipe descriptions
- BERT: Bidirectional Encoder Representations from Transformers
- Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks

Ranking Strategies: Other

- OpenAI Embeddings
- OpenAI Chat Completion
- Websocket

Overview

- Introduction
- Approach
- **Evaluation**
- Conclusion

Metrics

- Speed:
 - Setup and search time
 - Under 100ms for search time is acceptable
- Accuracy:
 - NDCG

NDCG

- Normalized Discounted Cumulative Gain
- Measures the ranking quality

$$DCG = \sum_{i=1}^k \frac{gains}{\log_2(i + 1)}$$

$$NDCG = \frac{DCG}{IDCG}$$

- Last rank k is important: NDCG@k

Speed

Name	Setup Delay	Search Delay
Name Search	68ns \pm 0.2ns	6.53μs \pm 0.08μs
Fuzzy Name Search	68.4ns \pm 0.5ns	756μs \pm 1.3μs
Fuzzy Description Search	276ns \pm 0.8ns	10ms \pm 0.05ms
Index Search	7.87s \pm 0.08s	1.74ms \pm 0.01ms
Semantic Search	85s \pm 1s	38.2ms \pm 0.9ms
OpenAI Embedding	3s \pm 1s	668ms \pm 949ms
OpenAI Chat Completion	530ns \pm 14.9ns	27.4s \pm 3.7s
Saved Index/ Embedding		
Index Search	1.5ms \pm 0.03ms	-
Semantic Search	682ms \pm 10ms	-
OpenAI Embedding	27.5ms \pm 0.25ms	-

Accuracy Setup

- Generate Test Dataset

Accuracy

Name	NDCG@5
Name Search	0.0
Fuzzy Name Search	0.41
Fuzzy Description Search	0.27
Index Search	0.22
Semantic Search	0.64
OpenAI Embedding	0.57

Accuracy on different recipe datasets

	docstring	cookbook	snippets
NameSearch	0.00	0.00	0.00
FuzzySearchName	0.41	0.45	0.53
FuzzySearchDescription	0.27	0.31	0.48
IndexSearch	0.22	0.40	0.29
SemanticSearch	0.64	0.71	0.80
OpenAIEmbeddings	0.57	0.44	0.88

Other

- Jupyter Notebooks
 - Different Websites use different versions
 - Widget library doesn't support DataFrame traitlets
 - Easy to set up and use
- Usefulness of the recipes
 - Fast
 - All-in-one solution
 - Restricted by quality of recipes

Overview

- Introduction
- Approach
- Evaluation
- **Conclusion**

Summary

- Lexical search is not accurate
- Using recipes together with semantic search is the best approach
 - fast
 - accurate
 - can still be improved
- Jupyter Notebooks good for prototyping

Future Work

- Improve the semantic search by fine-tuning the model
- Improve the recipe quality
- Get rid of most algorithms
- Port to standalone application or VSCode extension
- Easy to implement more ranking strategies