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

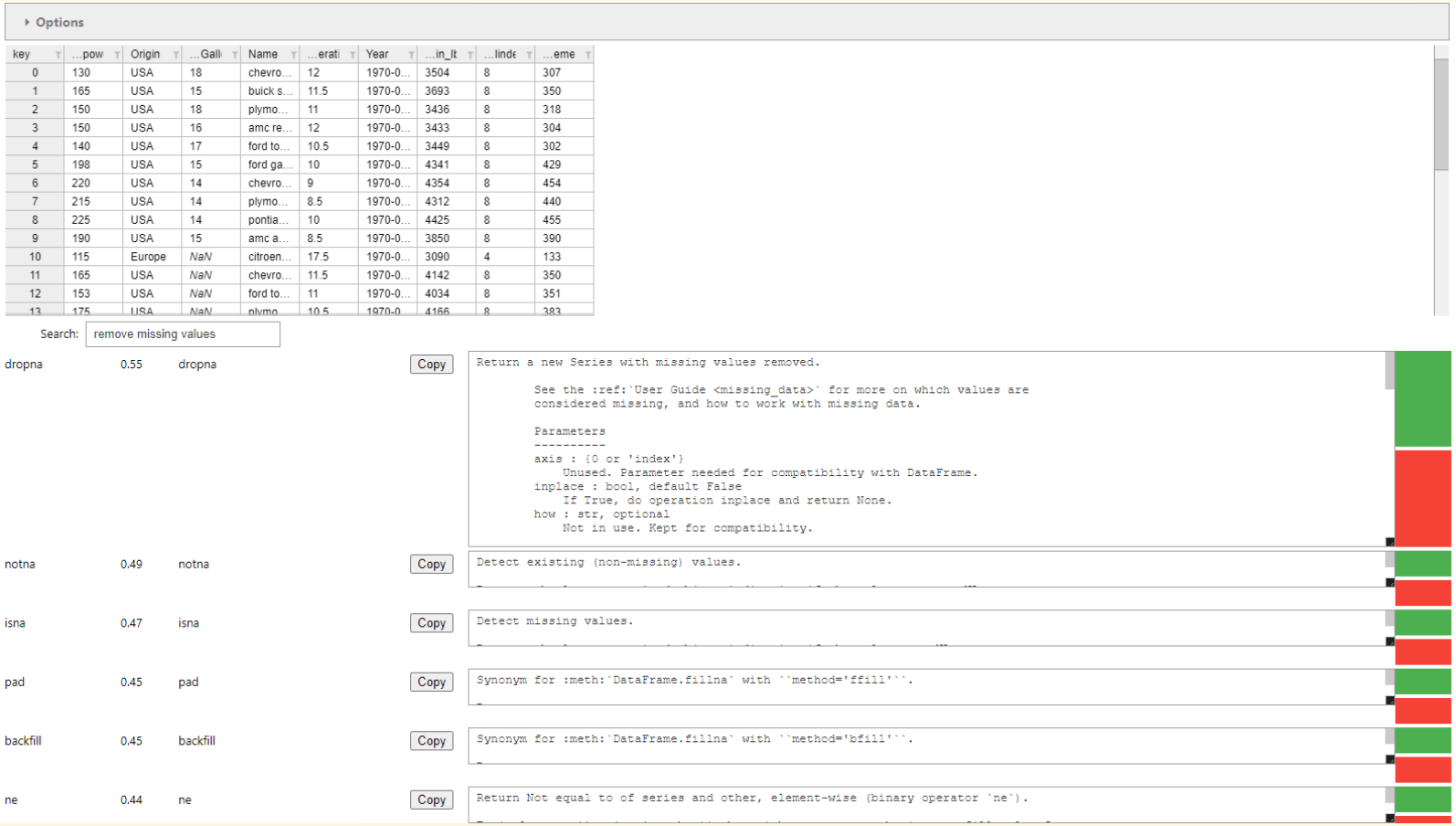
Overview

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

My Solution

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

Jupyter Notebook



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