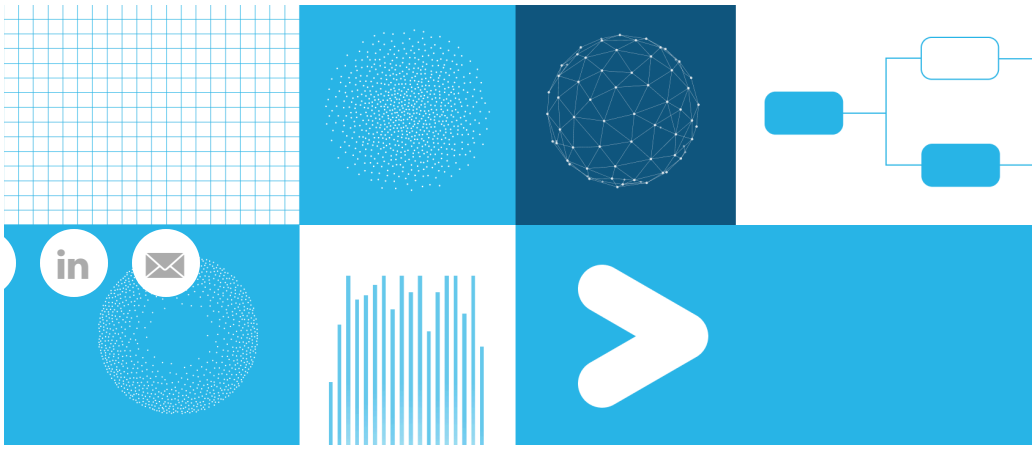
NOV 19, 2024

AUTHOR

Lukasz Borchmann

Benchmarking LLMs on Writing Feature Engineering Code

Gen AI



SHARE

Today, the limitations of LLMs are predominantly assessed using benchmarks focused on language understanding, world knowledge, code generation or mathematical reasoning in separation. This approach, however, overlooks some critical capabilities that can be measured in scenarios requiring the *integration* of skills and verification of their *instrumental* value in complex, real-world problems.

AUTHOR

Lukasz Borchmann

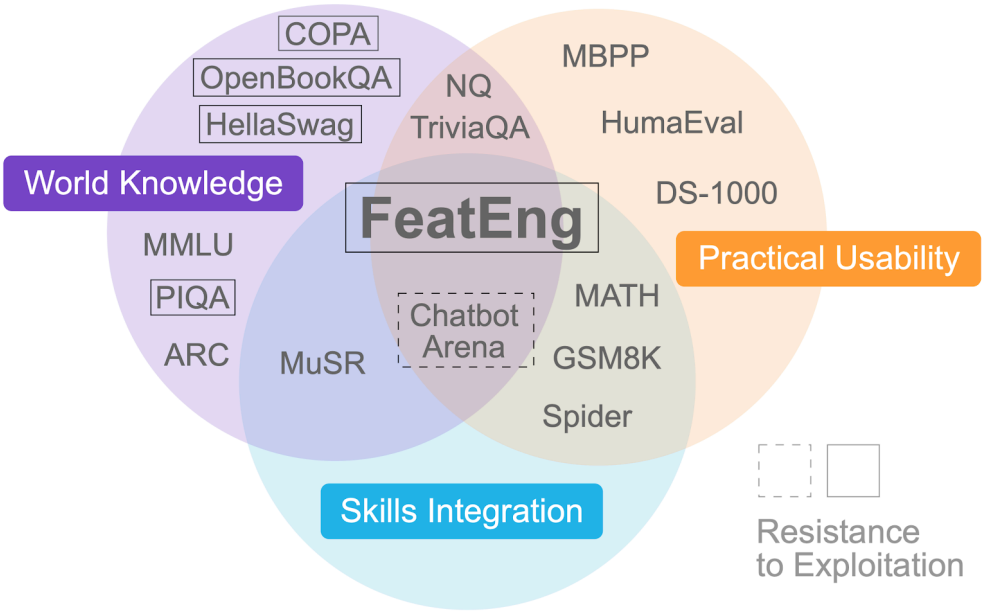


Figure 1. Unlike most popular benchmarks, FeatEng evaluates practical usability, requires extensive domain knowledge and complex skills integration, and is resistant to exploitation.

To this, we present a benchmark for LLMs tackling one of the most knowledge-intensive tasks in data science: writing feature engineering code. Models that perform well on FeatEng, as we call it, can reduce the time and expertise required for feature engineering in actual data science projects. They must creatively apply domain-specific knowledge to engineer features and integrate varied skills to produce meaningful and practical solutions.

How does it work?

We manually searched for appropriate and diverse tabular data sets based on criteria such as size, number of votes, license and popularity in the Kaggle community. Each of them was loaded and scrutinized (both the data set and feature descriptions were rewritten to make them more consistent and informative). The resulting information is fed to the LLM, which is asked to generate Python code.

AUTHOR

Lukasz Borchmann

Figure 2. Components of LLM's prompt in FeatEng. By telling the model it is participating in a Kaggle competition, we intend to make it better understand the setup used in FeatEng, where there is a need to generate feature engineering code similar to those used on the Kaggle platform.

Specifically, we require the model to design a function that transforms the data set, represented as pandas DataFrame, based on the provided problem description and metadata. It can drop, add or alter columns and rows, making tabular data better suited to the problem.

Figure 3. Signature of the expected function in the LLM output.

The evaluation score is derived from the improvement achieved by an XGBoost model fit on the modified data set compared to the original data. For example, improving accuracy from 0.80 (baseline score) to 0.95 (model score) reduces the error by 75%, which is the score assigned to the model for this sample in FeatEng.

To establish a reference point for evaluating the impact of LLM-generated feature engineering code, we directly fit on the train set and predict on the test without performing any transformation.

You came here for the numbers, didn't you?

Globally, the leaderboard is dominated by the *o1-preview* model, achieving an impressive performance of 11%+, but despite its significant superiority in aggregate metric, it provided the best-performing features only for one-fifth of considered problems. We argue that it is because having in-depth knowledge relevant to varied disciplines is challenging to achieve.

AUTHOR

Lukasz Borchmann

SHARE

Table 1. Performance on our benchmark (*FeatEng*) compared to Chatbot Arena ELO (*Arena*), along with secondary metrics: score on classification (*cls*) and regression (*reg*) problems; amount of code executed with error (*fail*); amount of cases where model improved over baseline (*i cls*, *i reg*); and the chance the model offers the best solution (*best*).

Results obtained by Yi Coder and Codestral indicate that though strong code-generation capabilities are required, they don't suffice to master the benchmark. Finally, the scores are vastly affected by the ability of models to follow instructions and plan over a long context of code being generated. Weaker models struggle to generate a valid and coherent code that adheres to the complex requirements they are facing.

Interestingly, compared to the Chatbot Arena ELO rating, our results show a strong agreement, suggesting that our benchmark effectively estimates its results without requiring extensive human evaluations.

What makes the models best?

We identified the common strategies employed by LLMs when tackling the FeatEng benchmark and developed a taxonomy for these approaches. Subsequently, we analyzed how different models generate distinct categories of features and examined the resulting impact on their performance.

Figure 4. Percentage of generated features in LLMs' output according to our taxonomy.

An analysis of the model-generated code reveals that different models use various advanced feature engineering approaches. Moreover, we show how leveraging domain knowledge and reasoning to generate

meaningful data transformations significantly improves model performance.

One of the patterns we observe is that the most capable models generate significantly more features exploiting Strong Domain Knowledge (e.g., 11% in the case of o1-preview and 3% for Mixtral). This happens partially at the expense of basic data processing and normalization, which does not require advanced reasoning capabilities and quickly approaches a saturation level where it is hard to gain further accuracy improvement. This aligns with our core premise that LLMs can exploit expert and domain knowledge when designing or selecting features.

Additionally, we found that the positive impact on model scores can be attributed to emitting features classified as Feature Extraction from Existing Data, Scaling and Normalization, Basic Data Preprocessing, Feature Construction and Interaction, as well as Encoding Categorical Variables.

AUTHOR

Lukasz Borchmann

SHARE

Sound interesting?

Read our recently published paper, **Can Models Help Us Create Better Models? Evaluating LLMs as Data Scientists**, and **see the code on GitHub** (star to follow the updates).

SHARE

RELATED CONTENT

SEP 12, 2024

LLM Interactive Workloads: Optimizing GPU Capacity for

SEP 05, 2024

Model Hotswapping: Optimizing AI

SEP 06, 2024

Benchmarking Real World Customer-

Interactive and Batch Workloads

At Snowflake, we offer a wide variety of LLM-powered features in Cortex AI, including Cortex LLM Functions, Snowflake Copilot and our recently released Cortex Analyst, now in public preview. While…

Learn More

Infrastructure and Enhancing LLM Efficiency

At Snowflake, we support customer AI workloads by offering a diverse range of open source…

Discover

Experienced Performance Using the Snowflake Performance Index (SPI)

I'm excited to share some details about one of the projects that I've been working…

More

SHARE

START YOUR
30-DAY FREE TRIAL

START NOW

Snowflake Inc.

Sign up for Snowflake Communications

diana.shaw@snow United States

By submitting this form, I understand Snowflake will process my personal information in accordance with their Privacy Notice. Additionally, I consent to my information being shared with Event Partners in accordance with Snowflake's Event Privacy Notice. I understand I may withdraw my consent or update my preferences here at any time.

AUTHOR

SUBSCRIBE NOW

Lukasz Borchmann

SHARE