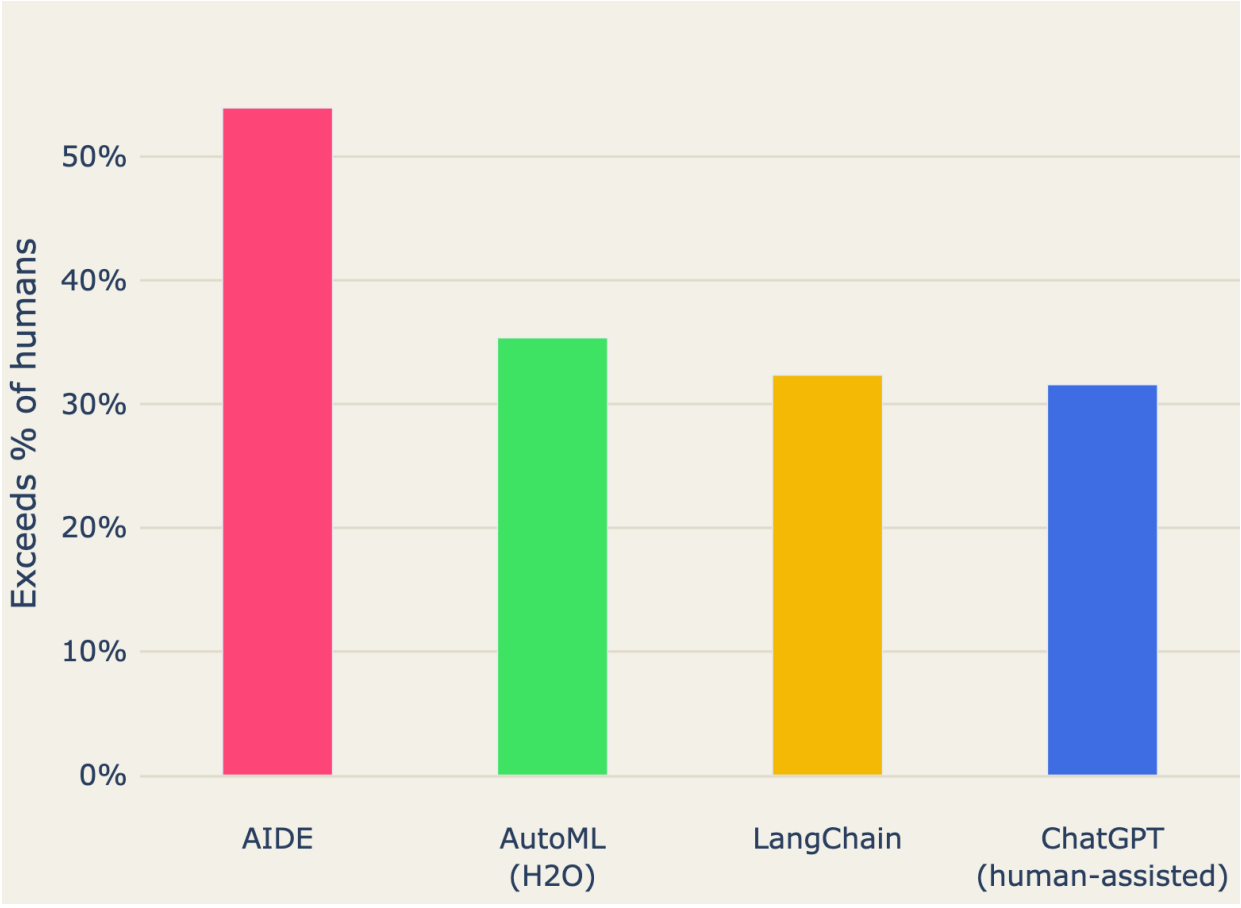
we·co

Blog Team Github

AIDE: Human-Level Performance in Data Science Competitions

April 4, 2024 • by Dominik Schmidt, Zhengyao Jiang, Yuxiang Wu

In the world of data science, Kaggle competitions have become a widely accepted standard for evaluating the skills and capabilities of data scientists. These competitions provide a platform for data scientists and machine learning enigneers to showcase their ability to solve complex real-world problems, work with diverse datasets, and develop high-performing machine learning solutions. Today, we're thrilled to announce a significant milestone: our AI-powered data science agent, AIDE, has achieved human-level performance on Kaggle competitions.



On average, AIDE outperforms half of human contestants. What sets AIDE apart is its ability to autonomously understand competition requirements, design and implement solutions, and generate submission files, all without any human intervention. Moreover, AIDE surpasses the performance of a conventional AutoML system H2O, as well as the LangChain Agent and ChatGPT (with human assistance).

Weco provides a cloud-hosted version of AIDE. We also understand that expert users may want to run it locally to leverage their own computing resources, ensure better data privacy, and achieve more customization. Therefore, AIDE will be open-sourced in the coming weeks at here. This repository currently showcases AIDE's solutions to over 60 Kaggle competitions.

Inside AIDE: A Peek Under the Hood

AIDE's problem-solving approach is inspired by how human data scientists tackle challenges. It starts by generating a set of initial solution drafts and then iteratively refines and improves them based on performance feedback. This process is driven by a technique we call Solution Space Tree Search.

At its core, Solution Space Tree Search consists of three main components:

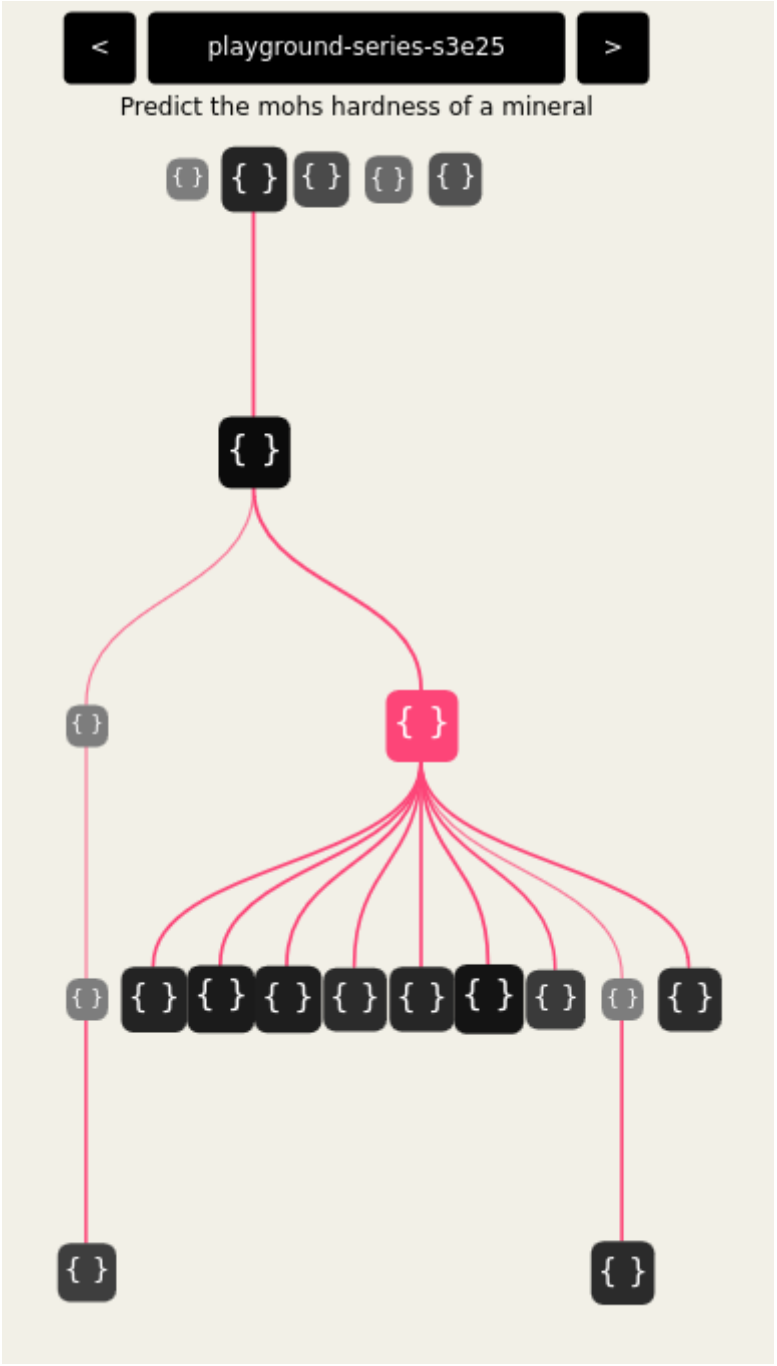
1. **Solution Generator:** This component proposes new solutions by either creating novel drafts or making changes to existing solutions, such as fixing bugs or introducing improvements.
2. **Evaluator:** The evaluator assesses the quality of each proposed solution by running it and comparing its performance against the objective.
3. **Solution Selector:** The solution selector picks the most promising solution from the explored options to serve as the starting point for the next iteration of refinement.

By repeatedly applying these steps, AIDE navigates the vast space of possible

solutions, progressively refining its approach until it converges on the optimal solution for the given data science problem.

This iterative, feedback-driven approach mirrors the way human data scientists work, combining the systematic rigor of algorithms with the adaptive creativity of human problem-solving. The result is a powerful tool that can tackle complex data science challenges with unprecedented efficiency and effectiveness.

Examples



```
To improve the previous solution, I propose to implement feature scaling on the
dataset before training the RandomForestRegressor model. Feature scaling can
help in improving the performance of many machine learning algorithms,
especially those that are sensitive to the scale of the data like KNN or SVM.
Although tree-based models like RandomForest are generally less sensitive to the
scale of the features, scaling might still help in convergence and performance,
especially when using ensemble methods. I will use the StandardScaler from
sklearn.preprocessing to scale the features.


import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import cross_val_score, KFold, GridSearchCV
from sklearn.metrics import make_scorer, median_absolute_error
from sklearn.preprocessing import StandardScaler

# Load the data
train_data = pd.read_csv("./input/train.csv")
test_data = pd.read_csv("./input/test.csv")

# Prepare the data
X_train = train_data.drop(["id", "Hardness"], axis=1)
y_train = train_data["Hardness"]
X_test = test_data.drop(["id"], axis=1)

# Scale the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Define the model
model = RandomForestRegressor(random_state=42)

# Define the evaluation metric
medae_scorer = make_scorer(median_absolute_error, greater_is_better=False)

# Set up the parameter grid for hyperparameter tuning
param_grid = {"n_estimators": [50, 100, 150], "max_depth": [10, 20, 30, None]}

# Perform grid search with 10-fold cross-validation
grid_search = GridSearchCV(
    estimator=model, param_grid=param_grid, cv=10, scoring=medae_scorer, n_jobs=-1
)
grid_search.fit(X_train_scaled, y_train)
```

Putting AIDE to the Test: Our Comprehensive Benchmarking Protocol

To truly understand AIDE's capabilities, we put it through its paces with a rigorous benchmarking process. Our benchmark suite comprises over 60 data science competitions. While the majority of our evaluation focuses on tabular data tasks, we also included a set of deep learning tasks to assess AIDE's ability to handle messy datasets, efficiently utilize GPU resources, and fine-tune open-source language models.

So, what exactly goes into each task in our benchmark? It's a three-part recipe:

1. **Task Description:** We provide AIDE with the same textual competition description that human data scientists would receive, ensuring a fair comparison.
2. **Dataset:** Each task comes with its own dataset, including an unlabeled test set for AIDE to make predictions on, just like in real-world scenarios.
3. **Evaluation Function:** To measure AIDE's performance, we devel-

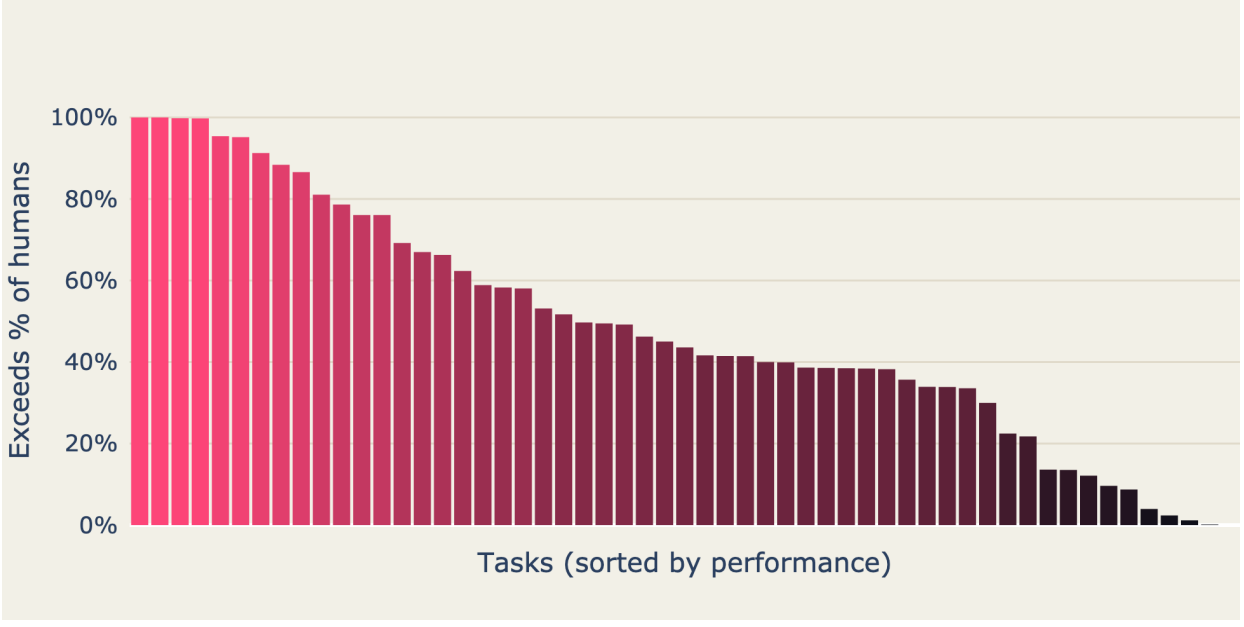
oped evaluation functions that assess the quality of its solutions and compare them against a wide range of submissions from human data scientists.

By subjecting AIDE to this comprehensive benchmarking protocol, we can gain deep insights into its strengths, limitations, and potential for real-world applications. In the following sections, we'll dive into the results of our evaluation and explore what they mean for the future of data science and AI.

Results & Analysis

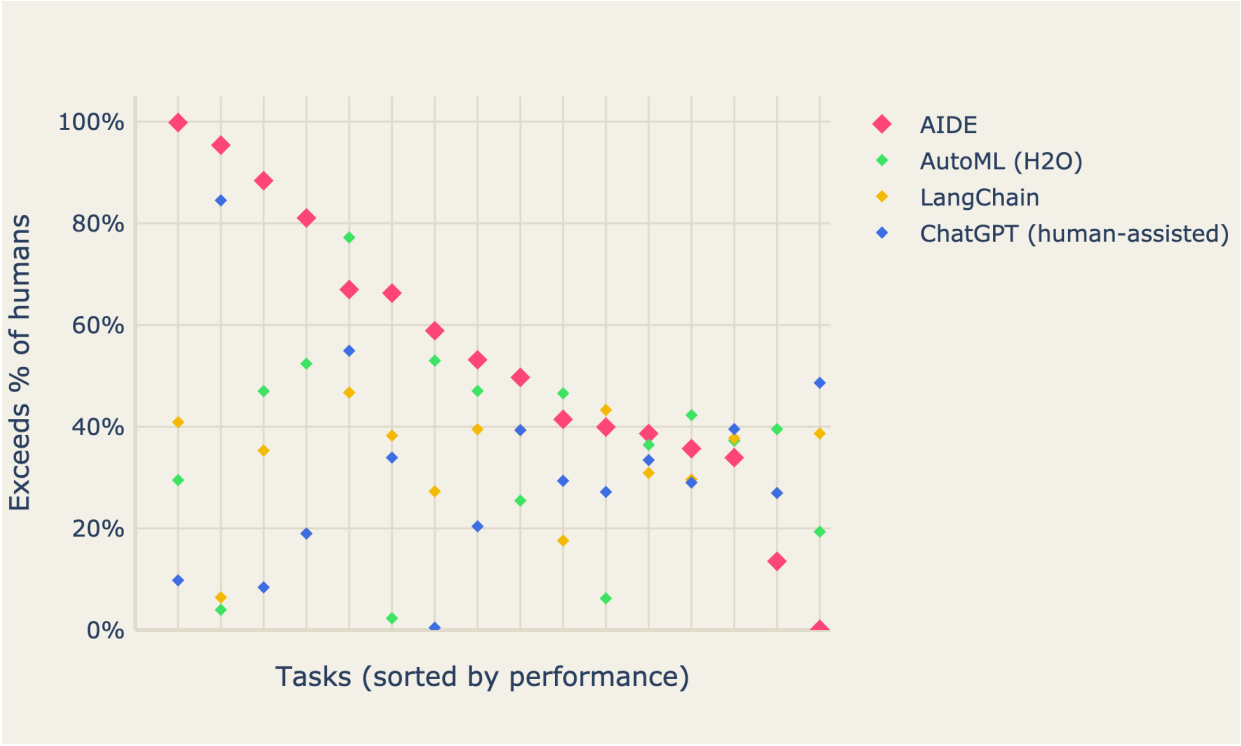
We now present the results of our benchmark. All results are provided on a human-normalized scale, meaning a performance of "50%" corresponds to the agent outperforming approximately 50% of human data scientists according to the leadboard of a competition.

General performance — In the figure below, we illustrate the performance distribution of AIDE across various tasks in our benchmark. AIDE's performance significantly varies with the task. In some tasks, AIDE's performance matches that of top-level experts, while in a small fraction of the tasks, the agent's solutions are relatively basic and cannot rival those crafted by humans.

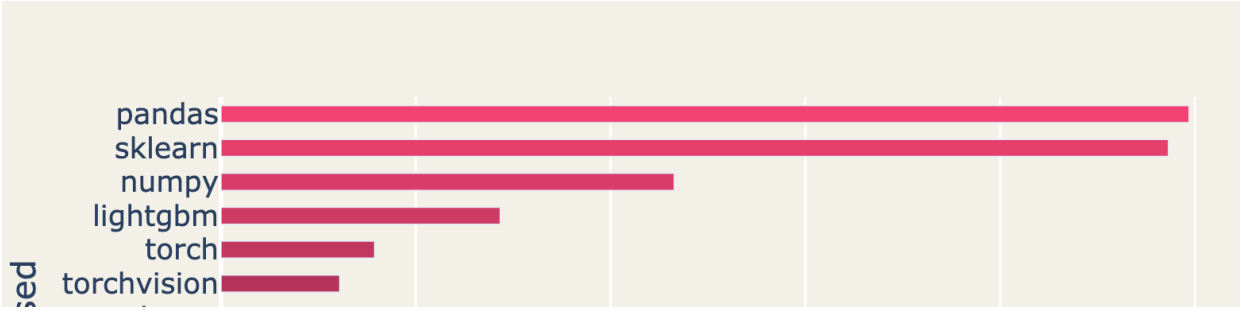


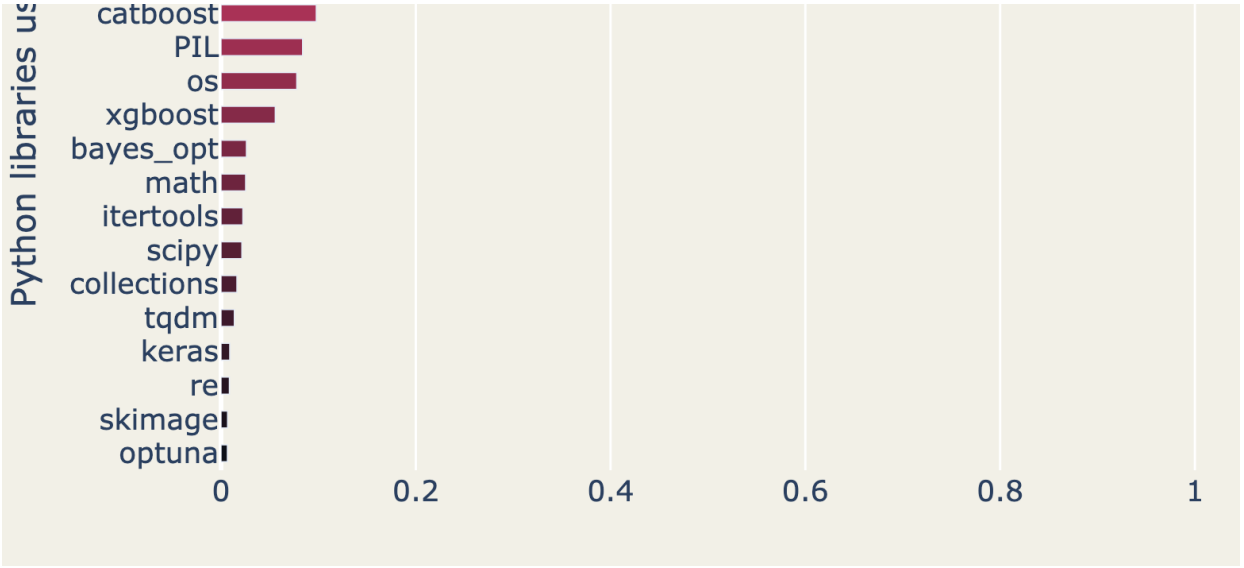
Comparison to baselines — Next, we compare AIDE to two baselines: a

Comparison to baselines — Next, we compare AIDE to two baselines: a popular industrial-level AutoML system called H2O, as well as the use of ChatGPT (interactive, with user guidance). The following result is obtained from a smaller subset of 16 tabular benchmark tasks that are solvable via both H2O and ChatGPT, and it shows that AIDE outperforms both baselines by a significant margin. Notably, unlike AIDE, which requires no human intervention, both AutoML and ChatGPT need human supervision. For AutoML, users need to read the task description and set up the configuration, and after the run is finished, users should post-edit the submission file to make it compatible with the format required by the competition. As for ChatGPT, users will have to provide ad-hoc instructions and, on average, need 3 rounds of interactions to get the result.

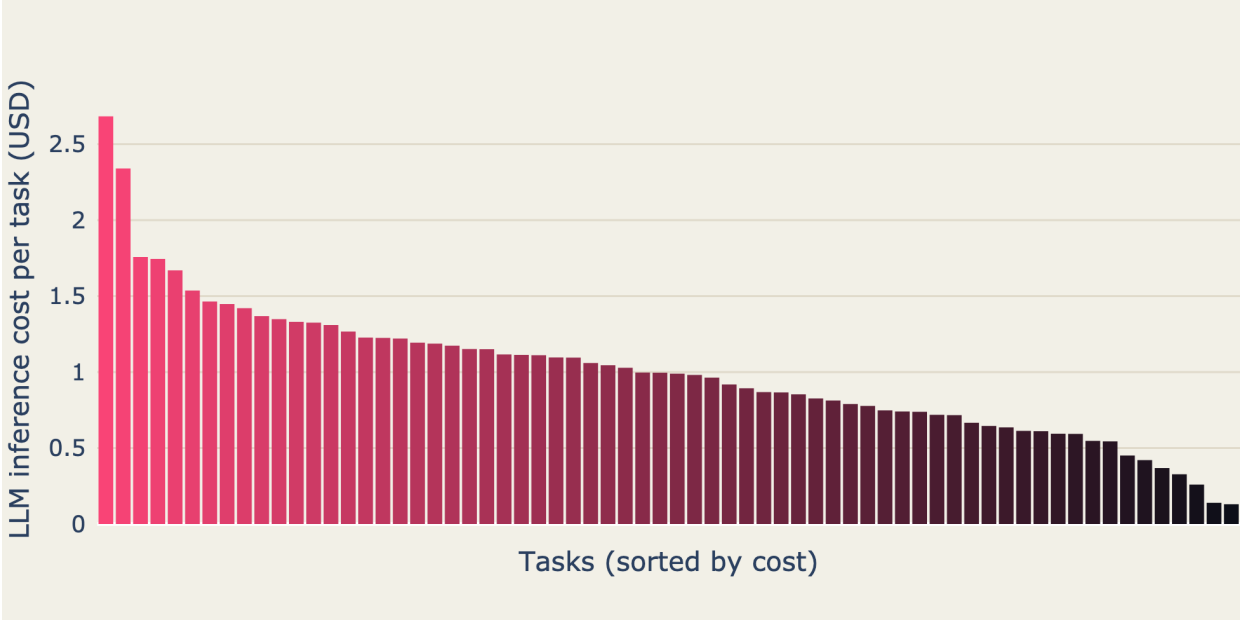


Versatility — AIDE is able to solve a wide range of tasks by leveraging a diverse set of existing libraries and tools. The following figure shows the distribution of Python libraries used by AIDE across a large set of (primarily) tabular tasks.





Cost — We also highlight the extraordinary cost-efficiency of AIDE. As seen in the following figure, AIDE is able to solve most of the tasks while incurring an LLM inference cost of less than 1$ per task when using gpt-4-turbo as the LLM backend.



Try out AIDE!

Want to experience the power of AIDE for yourself? We've launched a cloud-hosted version, join the waitlist below or get in touch at contact@weco.ai.

We believe open-sourcing AIDE is the best way to make it useful for expert users. We're thrilled to announce that AIDE will be open-sourced later this month! Check it out at https://github.com/WecoAI/aideml.

Join our waitlist

Please enter your email address below.*

name@example.com

OK