



Technical Test - ML Engineer / Software Engineer - Fundamental



Instructions:

01 Task 🎯

02 Tools you can use 🧰

03 Deliverables 📦

04 Supplementary Information ⓘ

All the best 🚀

Instructions:

01 Task 🎯

You are a machine learning engineer working on Lung Cancer treatment at Hospital Gustave Roussy

You have access to historical data of **+23 000 patients**

You have to predict the **treatment** that should be applied to the patients based on descriptive variables. For that you can split the dataset into a train set **80%** and test set **20%**

You need to benchmark the **CARTE** approach on this dataset and implement improvements and monitor those improvements.

Are considered improvements:

- **+ 10%** gain in classification metrics - version 0.0.24 vs your version
- **- 20%** in run time in seconds - version 0.0.24 vs your version
- **- 20%** in memory usage - version 0.0.24 vs your version


Once you are satisfied with one of the improved version deploy it on a simple **streamlit app** that would enable a physician to enter new patient information and make proper prediction.

Disclaimer: The CARTE approach originated as a research project developed to be released quickly, without adhering to the standards expected of production-ready solutions. A new approach, called TARTE, is in development. TARTE aims to surpass previous benchmarks and will meet the highest standards of the best tech companies.

 **Code Base:**

<https://github.com/soda-inria/carte>

Google Colab

 <https://colab.research.google.com/drive/1PeltEmNLeHQ26VQtFJhl7OxnzCS8rPMT?usp=sharing>



!! Main files:

- https://github.com/soda-inria/carte/blob/main/carte_ai/src/carte_table_to_graph.py
- https://github.com/soda-inria/carte/blob/main/carte_ai/src/carte_estimator.py

Dataset:

[lung_cancer_data 2.csv](#)

02 Tools you can use

You are free to use any libraries for this project. Here are a few suggestions to help you get started:

- **AI Libraries:** Leverage tools like scikit-learn, PyTorch, or TensorFlow for model development and training.
- **Experiment Tracking:** Use platforms such as MLFlow or AIM to efficiently track and manage your experiment runs.
- **Model Deployment and User Testing:** Utilize Streamlit for straightforward deployment and to enable interactive user testing of your models.

03 Deliverables

1. GitHub Repository:

Provide an updated repository with the improved version of the project, including all relevant code, documentation, and instructions for use.

2. Streamlit Link:

Share a hosted Streamlit application showcasing the functionality and allowing for user testing.

3. One-Page Document:

Prepare a concise document detailing:

- The improvements you would implement if given a full month to work on the project.

- The benchmarking pipeline you would design, taking into account additional potential use cases.

04 Supplementary Information

Here are the main advances brought by CARTE:

- **Graph Representation of Tables:** CARTE represents each row of a table as a star-like graph, with nodes corresponding to cell values and edges to their relationships with column names. This structure captures the context of each entry, facilitating the integration of data from tables with varying schemas without requiring explicit schema matching.

[ArXiv](#)

- **Open Vocabulary Embeddings:** By employing string embeddings for both entries and column names, CARTE effectively handles diverse and previously unseen categorical values. This approach allows the model to process an open set of vocabularies, addressing challenges such as typos or varying nomenclature across datasets.

[ArXiv](#)

- **Graph-Attentional Network:** The architecture utilizes a graph-attentional network to contextualize each table entry by considering its associated column name and neighboring entries. This mechanism enables the model to capture intricate relationships within the data, enhancing its predictive capabilities.

[ArXiv](#)

- **Pretraining on Unmatched Background Data:** CARTE can be pretrained on large, unmatched datasets without the need for data integration techniques like schema or entity matching. This pretraining strategy allows the model to learn from a broad spectrum of knowledge, which can then be fine-tuned for specific downstream tasks, improving performance even in scenarios with limited data.

[ArXiv](#)

- **Joint Learning Across Multiple Tables:** The model supports joint learning from multiple tables with different schemas, enabling it to enhance smaller datasets by leveraging information from larger, related tables. This capability is particularly beneficial in domains where data is fragmented across various sources.

ArXiv

arxiv.org

<https://arxiv.org/pdf/2402.16785>

All the best 🚀