Linkedin Mentor-Mentee Matching

Ben Hellmann, Yonatan Levin & Tomer Katz

1 Introduction

Whether for personal growth or career goals, mentors play a valuable role in the lives of people looking to achieve new levels of success. This is especially true at the beginning of a professional career. Help and guidance are key for juniors looking to succeed in initiating their careers. Mentors are experienced people with specialized knowledge whom mentees can enlist to educate and motivate them, either in their personal life, their career, or both.

The most popular website for connecting professionals to one another is LinkedIn. LinkedIn is a business and employment-focused online professional platform that has more than 1 billion registered members from over 200 countries and territories. We believe that this is the best place for young juniors to connect with experienced mentors. One way we think this can work is by creating the option for people to register themselves as mentors. Then, mentees will look for mentors using our tool, which will find the mentors most suitable for them.

Experienced people could be incentivized to register themselves as mentors by getting benefits from LinkedIn, such as certain LinkedIn Pro features or more exposure to their profiles. In this project, we will explain how we implemented our tool.

2 Data Collection and Integration

The main objective of our project is deciding how suitable a mentor-mentee match is. This means that an item in our dataset is a pair of two different LinkedIn profiles, with the first being a mentor and the second being a mentee—our goal is to give this item a score between 0 and 1.

2.1 LinkedIn Data

Each item includes details from the two LinkedIn profiles: the about section, positions, degrees, number of posts, number of followers, number of following, number of certifications, and number of months of professional experience.

The following transformation were performed on the profiles data:

- Number of followers and number of following remained as is.
- Number of posts and Number of certifications were calculated by taking the length of the posts and certifications columns, which were originally arrays.
- Number of months of professional experience was calculated by adding the Durations Short attribute in the experience column, which is an array of structs.
- The degree column (intermediary column) was made by concatenating the degree attribute in the education column, which is also an array of structs.
- The about, positions and degree columns were concatenated to create a full description of the LinkedIn user and then embedded to a dense vector using Bert.

2.2 Scraped Data

To enrich the data, we wanted to focus on what makes a good mentor. For this, we scraped data from mentoring-club.com, a mentoring platform where people can find their desired mentor. Using Beautiful Soup, we scraped profiles of 3,500 mentors. We extracted their names, positions, categories, about section, mentoring topics, and more.

We scraped the first 40 pages of profiles from each category on the website. Notably, the first 20 pages alone provided 3,200 of the 3,500 profiles we collected, covering most of the platform's active mentors. This approach gave us a well-rounded dataset representing various fields and expertise levels.

2.3 Mentor Similarity

Using the 2.2 data we made a "mentor archetype", which represents the traits that mentors use to describe themselves in their about section. We took only the about sections of the 2.2 data, made a Bert embedding for each row, and finally averaged them to get a single dense vector that we intend to represent the average mentor, i.e., "mentor archetype".

In order to create a scale of how similar a user is to the average mentor, we computed the cosine similarity of the full description embedding in 2.1 and the mentor archetype to create the mentor similarity column, the final feature for a user.

2.4 Data Labeling

Since the original data does not have labels for our specific task, we had to find a way to label large amounts of data. Instead of manually labeling the train set, we used LLMs to create the mentor-mentee labels.

To do so, we entered the about, education, experience, followers and following of two users and used the query: "Information is given about a potential mentor mentee pair. Label it as Good or Bad.[mentor data] + [mentee data] Is the mentor mentee pair a good match? Label:". The model returned a binary result on whether a the first user can be a good mentor the the second one. The usage of LLMs for the task of labeling is well known in the field of social connection recommendations (Li et al., 2025).

After testing multiple models, such as GPT2, Llama 3.2-1b and Llama 3.2-3b (AI@Meta, 2024) we found GPT-4o-mini (Hurst et al., 2024) to be the best balance of cost and effectiveness.

In addition to the data labeled by GPT-4o, we also labeled 1,000 pairs of mentor and mentee using Cohere (Cohere) so we can compare between the two.

The features discussed above are for a user, but as we mentioned, an element includes two users. Therefore each element has two of each said features, one for the mentor and one for the mentee.

There are 24,000 rows of labeled mentor-mentee pairs in the train set, as well as another 10,000 rows in the test set (We experienced an unexpected increase in the LLM querying rate which lead to a train-test imbalance).

3 Data Analysis

3.1 Feature Selection

Our goal in this project was finding the most suitable mentor-mentee pairs. Subsequently, we wanted to choose the most relevant features in the LinkedIn profiles dataset for this task. For each mentor and mentee we used some numerical features: number of followers, number of following, number of posts, number of certifications and experience length. In addition, for each pair we embedded the concatenation of position, degrees, and about section - this embedding is very important as it captures the semantic similarity between mentors' and mentees' professional backgrounds and aspirations. We used BERT to generate these embeddings, which enabled us to represent the textual information as dense vectors. This enabled us to compare the profiles to a mentor archetype.

The combination of numerical features and text embeddings provides a detailed view of each profile, considering both engagement metrics and professional content. This approach allows us to assess matches based on both quantitative career indicators and qualitative profile information.

3.2 Labeling

When using relatively small LLM models, we received only positive labels, independent of the query. We suspect that it is a result of a positivity bias that small models have, in order to satisfy user requests. However, when using gpt-4o-mini, we received a significant negativity biased (2:1 odds) when labeling random pairs. The combination of the two statements implies that the labels created are meaningful. Another model we experimented with is Gemini-1.5-flash, which labeled 90% of pairs as negative after rewriting the query with a positivity bias, which we think is too strict to be used for mentor-mentee matching. Also, we labeled examples using Cohere and got the following confusion matrix when comparing with GPT-4o-mini:

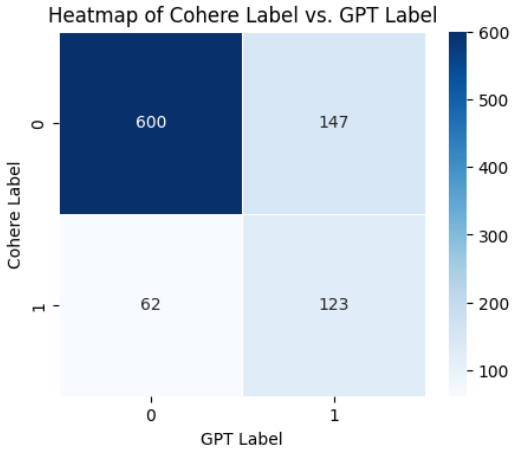


Figure 1

Looking at (1) we can see that the 2 LLMs agree on most labels being negative. But, we can see that there is a significant amount of examples labeled by Cohere as 0 and by GPT as 1 - showing that Cohere is more strict than GPT.

Since there are obvious differences between different LLMs in this task, we can infer that the performance of our models are heavily influenced by the different LLMs and using a larger and/or better model would result in a much more accurate model, though it would require a significant budget.

Another factor that changes the performance of the labeling is the actual wording of the query.

When experimenting with the Gemini-1.5-flash model, we tested a different wording to the query and the difference in the result was perceptible without dedicated testing.

Based on these observations, we can conclude that both the choice of LLM and the specific wording of queries significantly impact model performance. While optimizing these factors could improve accuracy, the method is not very robust, as its performance varies depending on these variables. This highlights the need for further refinement to achieve more consistent and reliable results, though improvements may come with trade-offs in cost and complexity.

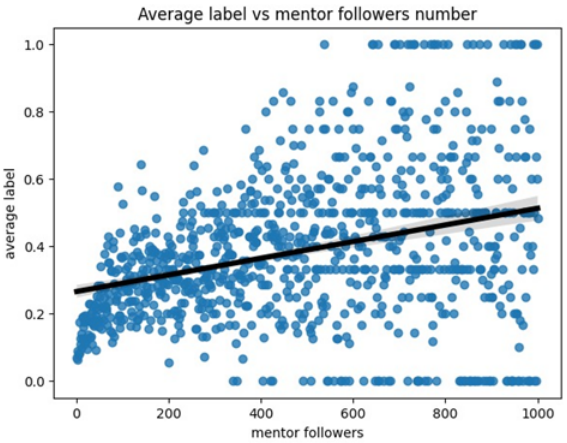


Figure 2

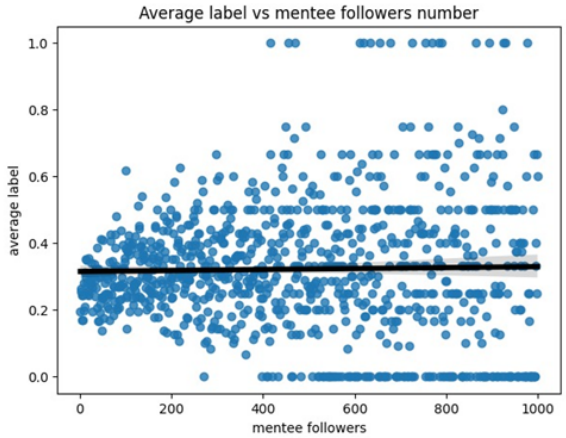


Figure 3

A sanity check showing GPT-4o-mini gives us results that make sense is shown in figures 2 and 3. In figure 2 the average label (between 0 and 1) is shown to grow as the number of the followers the mentor has grows. However, in figure 3 the average label doesn't seem to change at all by the number of followers. This makes sense as it shows that profiles with more followers are more likely to be suited to be mentors but in the case of mentees the number of followers shouldn't matter.

4 AI Methodologies

In order to predict whether a pair of a mentor and mentee is a good match we used various different machine learning methods. The input to each of these methods is a vector $v = (x_1, \ldots, x_n, y_1, \ldots, y_n)$ with $x_1, \ldots, x_n$ being features of the mentor and $y_1, \ldots, y_n$ being features of the mentee. The output of the methods is a number between 0 and 1 representing the probability the model is confident that v represents a good mentor-mentee match.

With the aim of training a reliable method we used **active learning**. First, we trained a logistic regression model on our data. Then, we chose the data that the logistic regression was least confident deciding whether they are a good match or not (e.g. the matches that got the scores closest to 0.5), labeled and added them to the training data and trained different machine learning models on the new revised train set.

On the new train set we decided to focus on training classic machine learning models. The main reason we chose to do this is that deep learning models require a large labeled datasets. While we did have big data, we did not have many labels and used LLMs for labeling which took a significant amount of time and money.

Since there is a great imbalance in the labels, the metric we focused on is weighted F1. We also tested positive example sampling for this reason, but received worse results, so we abandoned this direction.

List of machine learning models we used:

- Logistic Regression
- Support Vector Machine
- Random Forest
- Gradient Boosted Trees
- Decision Tree
- Multi Layered Perceptron

5 Evaluation and Results

As explained in previous section, the evaluation in our project is based on labels given to pairs of mentors and mentees by GPT-4o.

Initially we labeled 21,000 pairs for train and 10,000 pairs for test, adding 3,000 labeled pairs to train after the active learning process.

In the table (1), the results achieved by the different models are shown. The first 2 columns correspond to results achieved by logistic regression before and after active learning process. As can

	LR		SVM	RF	GBT	DT	MLP
Metric	**Pre AL**	**Post AL**					
Accuracy	0.653	**0.746**	**0.748**	0.687	0.741	0.678	0.682
Precision	0.556	0.648	0.659	**0.856**	0.663	0.511	0.534
Recall	0.472	**0.498**	0.481	0.058	0.431	0.489	0.263
Weighted F1	0.647	**0.736**	**0.736**	0.580	0.724	0.676	0.646
F1	0.511	**0.563**	**0.556**	0.109	0.522	0.499	0.353

Table 1: Comparison of Model Performance Metrics Over **GPT-4o-mini** test labels

Metric	**LR Post AL**	**SVM**	**RF**	**GBT**	**DT**	**MLP**
Accuracy	0.754	0.748	**0.797**	0.747	0.667	0.079
Precision	0.408	0.389	**0.588**	0.358	0.269	0.348
Recall	**0.438**	0.389	0.049	0.286	0.360	0.263
Weighted F1	**0.757**	0.748	0.722	0.737	0.684	0.721
F1	**0.423**	0.389	0.091	0.318	0.308	0.129

Table 2: Comparison of Model Performance Metrics Over **Cohere** test labels

be seen, the weighted F1 improve significantly by 13.8% after the active learning procedure.

The rest of the columns correspond to the results achieved by the rest of the machine learning models on the data **after** the active learning process.

Analyzing the results, the best performance is achieved by Logistic Regression and Support Vector Machines.

We can see that all models get a low recall score. This can probably be explained by the data imbalance that lets the models get a high accuracy by giving a label of 0 to most pairs. A model that seems to operate this way in our case is Random Forest.

To further analyze the results we tested our model also on 1,000 pairs labeled by **Cohere** (2). We can see that the models perform similarly on these examples though with lower precision and recall. This can be explained by the different balance between negative and positive examples with negative examples being roughly 80% of the labels given by Cohere. This shows that the learned models learned in the end the labels of GPT-4o and would be significantly different if trained on a different LLM. In addition, if the model was created using real feedback from humans it would probably behave differently.

6 Limitations and Reflection

While we think that this product has potential to be effective, there are quite a few limitations for implementation at our current capacity

- **Dependence on LLM Labeling** - As mentioned above, with a non-existent budget for labeling, the model is limited by the label quality and the number of labels available. Since all labels were generated by LLMs rather than human annotators, the dataset reflects the biases and inconsistencies of different models. Additionally, the method is not very robust—small changes in the prompt or the model used for labeling resulted in noticeable shifts in the labeled dataset.

- **Real World Application** - In practice, when given a mentee, the best performing models, by regular test set method, will suggest the same mentors with very high probability. We suggest the explanation that the labeling did not put enough weight on fitting the industry of the mentor and the mentee, as well as the fact that the best performing model are linear and therefore don't reflect the nature of the assignment as well as others could.

- **Runtime** - Our feature engineering involved various computationally expensive tasks such as embeddings, joins, and vector-based operations, which significantly impacted runtime performance. These operations required substantial computational resources, particularly when processing large-scale datasets. Although, it is still significantly less than using LLMs.

While the current product may not be fully effective—yielding similar results for different mentees—we believe the project as a whole has been successful. The existing framework provides a strong foundation for future growth, and our current findings have led to valuable insights. We are confident that with relatively simple adjustments, such as query optimization, more non-linear models, and additional active learning cycles, performance can be significantly enhanced. However, the most substantial gains will likely come from incorporating larger datasets and deep learning models.

7 Conclusions

Summarizing the achievements and outcomes of the article, we arrive at the following conclusions:

1. **High variance between different LLMs Behavior**: Different LLMs achieve very different labeling results. Therefore, when using LLMs, it is important to analyze which one suits the task.

2. **Prompt Engineering Significance**: Imprecise instructions to GPT-4o-mini results in unsuitable mentor-mentee matches across different industries, demonstrating that precise prompting is crucial when using LLMs for data labeling tasks.

3. **Logistic Regression and SVM Success**: These methods were the most successful in predicting the labels in our model, achieving a weighted F1 score of 0.74.

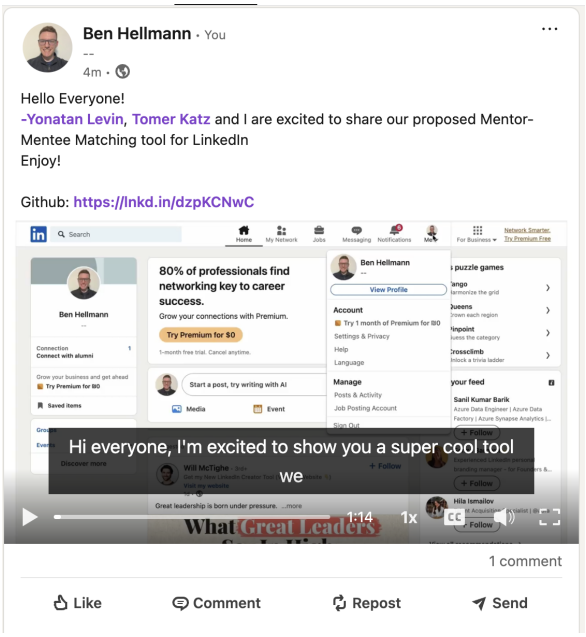
4. **Active Learning Impact**: Active learning significantly improved our performance, increasing the weighted F1 score by 13.8% and F1 score by 10.2%.

5. **Data Balance Challenges**: The imbalance in our labeled dataset affected model performance, particularly in recall scores. This suggests the need for more sophisticated sampling or weighting techniques in future iterations.

A Appendix

A.1 User's Journey

Link to LinkedIn post with video



A.2 Technical Overview

Technical Overview Video

A.3 Images, Graphs, Plots

mentee_followers	mentee_following	mentee_experience_	mentee_vector_bert	mentee_num of posts	mentee_num of certifications	mentee_similarity

mentee_id	mentor_id	mentor_followers	mentor_following	mentor_experience_length	mentor_vector_bert	mentor_num of posts	mentor_num of certification	mentor_similarity	label

Figure 4: The Train Set Features, to use in spark, the two user's id are concatenated and all the features are converted to a vector, resulting in a table with 3 columns: id, features, label.

References

AI@Meta. 2024. Llama 3 model card.

Cohere. Coheremodelsoverview.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Dongyang Li, Yuhan Wu, Jianshan Sun, and Yuanchun Jiang. 2025. Disentangling the factors driving friendship formation: An llm-enhanced graph convolutional approach for friend recommendation.