Skillsets Prediction from LinkedIn Profiles: A Machine Learning Approach to Enhancing Recruitment Efficiency

BT5153 Applied Machine Learning in Business Analytics – Group 06 Project Report

Huang Daopeng (A0280418E), Kong Dehao (A0172766U), Phan Le Ha Linh (A0280506J), Qiu Zixuan (A0280464A), Sun Yixin (A0280547Y)

Github for Code and Dataset: https://github.com/Tracyyyyy17/BT5153-GP06.git

Abstract

This project aims to develop an automated system for resume screening, utilizing AI/ML tools to identify skillsets from candidates' LinkedIn profiles to enhance the efficiency and accuracy of HR processes. The core deliverable is a multi-label classifier that analyzes 3,388 profiles, leveraging NLP techniques and machine learning, integrated with HR software for real-time skill prediction. The project focuses on technical metrics like model performance and robustness in the current phase, with future applications planned to assess real-world efficiency and improvement in HR satisfaction. Ultimately, while the model aims to significantly reduce the time HR spends on screening and improve candidate match quality, it also recognizes the irreplaceable value of human judgment in recruitment.

1. Problem Description

Resume screening is the process of reviewing potential employees' education and work/internship experience profiles to determine whether the candidate is qualified for the position. It is a major part of the pre-employment screening process since 88% of candidates are unqualified at the initial process and need to be screened out (Bahr, 2021). However, the convenience brought by the internet has made it easier to publish job descriptions and deliver resumes, drastically increasing the number of resumes HRs (Human Resource) receive. A single job posting can receive an average of 250 resumes nowadays, and the average cost-per-hire is $4,425 by traditional manual screen (Nordmark, 2020). In time consumption, recruiters spend 23 hours on average screening resumes for a single hire (Bahr, 2021). Besides, candidates' backgrounds can be messy across different job post platforms, making the screening process non-standardized and even more time-consuming. Therefore, using AI/ML tools to speed up resume screening can not only reduce hiring costs and HRs' workload but de-bias the screening process.

Our group's project aims to develop a skillsets identification model based on candidates' resumes and profiles, which is under the big theme of an automated resume screening system. By accurately identifying candidates' skillsets, the system can further match them with position requirements, boosting HRs' resume screening efficiency and accuracy.

This project aims to develop a predictive model to identify skill sets from LinkedIn profiles, aiming to streamline the recruitment process. Since our current dataset limits the direct measurement of real-world application metrics, the project's success will primarily focus on technical performance.

2. Data Collection and Exploration

2.1 Data Sources

Our dataset consists of 6,000 LinkedIn profiles, primarily from equity research analysts. The data is sourced through LinkedIn's API, third-party API, and supplementary web scraping techniques. This dataset features a variety of elements such as ID, country, languages, the "About Me" section, education, experience, and listed skills.

2.2 Data Description

In our analysis, the 50-class skills serve as the multi-label classification target while the profile features are the input data. The dataset, after removal missing values, consists of 3,388 valid profiles. Skills are encoded using one-hot encoding across 50 columns.

Skill distribution is highly imbalanced, as illustrated in Figure 1. The plot shows the frequency of analysts possessing certain skills, highlighting a significant skewness towards a few skills. For instance, 'financial analysis' is a skill labeled by 90.23% of analysts. The skills

'Management/Microsoft Office/Customer Service' and 'Strategic Planning/Team Leadership/Sales' exhibit relatively balanced distributions, each with about 40% representation. Most skills, however, have only limited representation among this group of analysts.

Profile features comprise several textual and numerical columns. Textual columns include analysts' languages, their summary, education description, experience descriptions. Numerical columns consist of the number languages spoken, the number of educational entries, and the number of professional experiences. English is the most commonly listed language, followed by Spanish and French. Analysts typically list 2.00 educational entries and 6.39 professional experiences on average. Only 37.47% of analysts have completed the 'summary' section of their LinkedIn profiles.

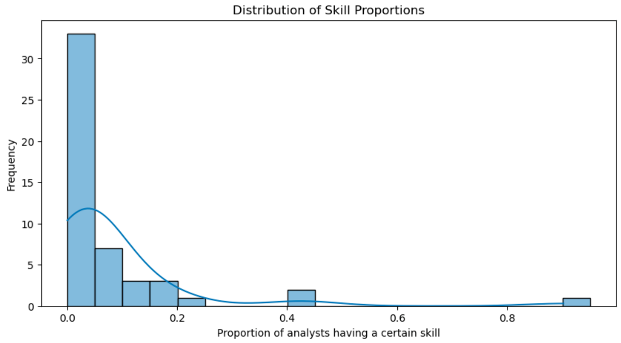

Figure 1. Distribution of skill labels proportion. Most of the skills are parsley labelled. 90% analysts labelled the skill of 'financial analysis'.

3. Methodology

3.1 Data Preprocessing

For models not using pretrain language models, we need to first preprocess the text data to strip irrelevant information. The steps include converting text to lowercase, removing URLs, mentions, and hashtags, removing punctuations, tokenization, removing stop words, and lemmatization. We didn't include stemming since it would cause over-stemming problem. To ensure a high quality of lemmatization, we utilized Stanza Lemmatizer, which is developed by Stanford NLP Group.

To address the significant class imbalance observed in the dataset, we employed MLSMOTE. This technique identifies labels that are underrepresented in the training data, selects the nearest neighbors for each of these minority labels, and synthesizes new samples by interpolating between selected samples and their neighbors, thereby enriching the dataset with additional examples of minority classes (Charte et al., 2015).

3.2 Feature Processing

Feature Extraction: Bag of words & TF-IDF

Once the data is clean and standardized, feature extraction techniques are applied to transform the raw data into a format that can be efficiently used by machine learning algorithms. Here we used Bag of Words (BoW) and TF-IDF.

BoW: Represents text data as a matrix of token counts, disregarding word order but monitoring frequency.

TF-IDF: Advances on BoW by weighting word frequencies based on their uniqueness in a document relative to the corpus, highlighting words that are frequent in specific documents but rare overall.

Dimension Reduction: PCA & Truncated SVD

Given the high dimensionality from BoW and TF-IDF, we use PCA and Truncated SVD for dimension reduction. They helped us to reduce over-fitting and noise due to high-dimensionality and improve computational efficiency. They also helped in identifying latent factors or underlying structures in the data. Unlike PCA, Truncated SVD does not center the data and is suitable for sparse datasets, especially those produced by BoW and TF-IDF.

Pretrained Model: RoBERTa, DistillBERT & all-miniLM-L6-v2

We also incorporate RoBERTa (Robustly Optimized BERT Pretraining Approach), DISTILBERT (Distilled Bidirectional Encoder Representations from Transformers) & ALL-MINILM-L6-V2 (All Mini Language Model Layer-6 Version 2).

Unlike BoW and TF-IDF that ignore word order and context, RoBERTa understands the context and semantics of words in sentences, which greatly enhances its ability to discern nuances in text data. Besides, RoBERTa has been pre-trained on an extensive corpus and can be fine-tuned on smaller datasets for specific tasks. It outperforms traditional machine learning models that rely on feature extraction combined with dimension reduction techniques. Additionally, DistilBERT, a lighter version of BERT, provides nearly the same effectiveness while being more efficient in terms of computational resources. This makes it suitable for applications where deploying heavier models might be restrictive. On the other hand, all-MiniLM-L6-V2, known for its capacity to generalize across multiple languages, expands the scope of multilingual text analysis. This model leverages a deep transformer network, yet remains compact, making it ideal for embedding in applications requiring real-time processing and low latency.

Together, these models represent a powerful toolkit for tackling a wide array of NLP tasks, demonstrating superior performance and flexibility compared to older statistical methods.

3.3 Modeling

In this crucial phase of our project, we focus on the development and refinement of various machine learning models designed to harness the full potential of AI in the resume screening process. The following subsections detail the specific models employed, highlighting their setup, the rationale behind their use, and the specific challenges they address within the scope of our HR-focused AI solutions.

3.3.1 Model 1 - RoBERTa

One of our machine learning models applied for predicting LinkedIn skills is the RoBERTa (Robustly Optimized BERT Pretraining Approach) model, renowned for its natural language processing capabilities. Compared to classical machine learning methods, RoBERTa makes use of the self-attention algorithm, and the model has been pre-trained on large amounts of data, which makes it highly accurate and robust. Below introduces the training setup and the strategies employed to enhance the model's accuracy and efficiency.

3.3.1.1 Tokenization

We utilized the RoBERTa tokenizer, a crucial component for preparing the input data. This tokenizer converts text into tokens that can be processed by the model. We set the maximum sequence length to 512 tokens to balance between capturing sufficient context and managing computational efficiency. This tokenization process ensures that the text data from LinkedIn profiles is appropriately formatted for the model, maintaining the semantic integrity necessary for effective learning.

3.3.1.2 Training Parameters

In the training of our model, we utilized a batch size of 16 to strike an optimal balance between computational efficiency and memory constraints, ensuring timely updates to the model's parameters. The AdamW optimizer was selected for its robustness in handling sparse gradients and effectiveness across our large dataset. We set the learning rate at 1e-5 to allow for gradual and thorough learning, preventing the model from overlooking subtle yet critical patterns within the data. The training process was designed to run for a maximum of 20 epochs, giving the model ample opportunity to iteratively learn and adapt to the dataset without the risk of overfitting. This approach ensured a systematic and deep learning process, tailored to maximize performance and accuracy.

3.3.1.3 Handling Class Imbalance

To address the challenge of class imbalance, we have applied three main approaches elaborated below:

- **Different Loss Functions**: We experimented with Focal Loss but settled on using the Binary Cross-Entropy with Logits Loss (BCEWithLogitsLoss) adjusted with class weights. This loss function is particularly effective as it combines a sigmoid activation with the binary cross-entropy loss in one single function.
- **Class Weights**: Calculated based on the inverse frequency of the labels to give more importance to rarer labels, thus helping the model to pay more attention to underrepresented classes.
- **Dynamic Upsampling**: We implemented a dynamic upsampling mechanism within the training dataset. This method involves artificially enhancing the minority class by duplicating samples, varied at each epoch, to provide balanced exposure of all classes during the training phase.

3.3.2 Model 2 – NLP Feature Extractors with ML models

Instead of advanced Transformer-based pre-trained language model, we also utilized traditional NLP methods and machine learning models, Random Forest, and LightGBM, to study text data.

Bag of Words (BoW) and TF-IDF were used to extract features from text data. We set ngram_range parameter of CounterVectorizer to be (1,2) to capture two-word phrases.

Since there were more than 330k columns after vectorization, dimension reduction needed to be conducted to save computation power and memory capacity. Three methods, PCA (n_components=200), TruncatedSVD (n_components=200), and set max_features=300 in CounterVectorizer, were utilized.

There were 12 models in total to compare 2 different feature extraction techniques, 3 dimension reduction techniques, and 2 machine learning models. One combination of model design would be chosen for future improvement.

3.3.3 Model 3 - Embedding with PCA

We also intended to combine the powerful embedding models with the traditional machine learning models. Compared to Bag of Words and TF-IDF, embedding models, such as DistilBERT and all-MiniLM-L6-v2 tokenizer, are able to capture the context and semantic relationships between words.

After obtaining the embedding vectors that are of high dimensions, we applied Principal Component Analysis (PCA) to reduce the dimensionality of the data. By reducing the dimensions, we hoped to improve computational efficiency and avoid the curse of dimensionality.

We experimented with different values of the n_components parameter in PCA, which determines the number of principal components to keep in the transformed data. We tried values from 0.5 to 0.95, which means that PCA will choose the minimum number of components such that the explained variance ratio is greater than or equal to the given value. We investigated how this parameter affects the performance of the subsequent machine learning models.

We also applied some preprocessing steps to the other features in the data, such as standardizing the numerical features and one-hot encoding the categorical features.

The next step was to train and evaluate different machine learning models on the reduced data. We chose multiple popular and powerful models: Random Forest, XGBoost, and LightGBM. All models are capable of handling complex and nonlinear patterns in the data and have many hyperparameters that can be tuned to improve their performance. We ran multiple runs of grid search to obtain the optimal hyperparameters for the models.

Lastly, we also experimented with Stacking Classifier, which is an ensemble method that combines the predictions of multiple base models using a meta-model. The idea is to leverage the strengths of different models and reduce the variance and bias of the final prediction. We used Random Forest and XGBoost as the base models, and Logistic Regression as the meta-model. We trained the base models on the reduced data and the meta-model on the cross-validated predictions of the base models. This way, we hoped to achieve higher F-1 score and robustness than any single model.

3.3.4 Model 4 - Specific Embedding and Upsampling

Learning from previous models, in this section, the focus was on enhancing the model performance through targeted column embeddings and the application of MLSMOTE, accompanied by the exploration of Random Forest and Stacking Classifier ensemble method.

We leveraged specialized embeddings for key columns—specifically, the summary, education, and experience columns—rather than utilizing the entire combined text feature as presented in Model 3. This approach aimed to capture more nuanced and directed semantic meanings. The model selected for this task was 'all-MiniLM-L6-v2', part of the Sentence Transformers library. This model is recognized for its efficiency and compact size, yet maintains commendable performance levels, demonstrating an advantage over larger models like 'distilbert-base-uncased'.

In Sections 3.3.2 and 3.3.3, LightGBM was identified as the highest performing model due to its efficiency in handling large datasets and its ability to manage complex non-linear relationships effectively; however, here in Model 4, the focus shifted to leveraging Random Forest and Stacking Classifier to enhance the handling of class imbalance and incorporate the strengths of multiple predictive models, thereby providing a more robust and generalized approach suitable for the complex and diverse data structure presented by the specialized embeddings and upsampling techniques. Other features mirror the most suitable configurations from Sections 3.3.2 and 3.3.3, such as limiting max_features to 300 in TF-IDF without dimension reduction for the combined text and employing the same embedding techniques.

The models were configured with varying combinations of features and strategies as detailed in Table 1 below:

Table 1. Model 4 Specific Embedding and Upsampling combinations

	Model 4a	Model 4b	Model 4c	Model 4d	Model 4e
TF-IDF Text	√	√	√	√	√
Numerical features	×	√	√	√	√
Embedded columns	×	×	√	√	√
ML-SMOTE	×	×	×	√	√
Stacking Classifier	×	×	×	×	√

3.4 Model Evaluation and Selection

3.4.1 Evaluation Criteria

In a multi-label classification problem, especially when dealing with imbalanced datasets, selecting the appropriate averaging method for precision, recall, and F1-score is crucial for a more accurate assessment of model performances. Definitions of commonly used averaging methods are summarized in Table 2.

Table 2. Definitions of different averaging methods

Micro avg	Aggregates the individual TP, FP, TN, and FN counts across all classes before calculating the overall precision, recall, and F1 score.
Macro avg	Calculates the precision, recall, and F1 score independently for each class and then takes the unweighted mean of these per-class metrics.
Weighted avg	Calculates the precision, recall, and F1 score independently for each class and then takes the weighted mean of these per-class metrics, where the weights are proportional to the class sizes.

Sample avg	Calculates the precision, recall, and F1 score for each individual sample and then takes the unweighted mean of these per-sample metrics.

Macro-average and weighted-average metrices would be our prior focus. They take average on class level, helping us understand how model performs on minority/majority classes. Besides, sample-average metrics may ignore the imbalanced class distribution and be informative on instance level, showing the average performance across all samples. However, performances on high-frequency classes dominate in micro-average metrics, making the result biased. Therefore, our models would be evaluated based on macro-average, weighted-average, and sample-average metrics, the higher the better. If the difference between macro-average and weighted-average metrics are smaller, it means that the model performs more balanced between majority and minority classes.

3.4.2 COMPARISON WITHIN EACH MODEL

3.4.2.1 MODEL 1 - ROBERTA

Various improvements have been implemented to address the issue of class imbalance. Table 3 below compares the performance of the RoBERTa model with and without dynamic upsampling, demonstrating its ability to balance better precision and recall, which in turn enhances F1 scores:

Table 3. Model 1 - RoBERTa comparison of performance

	Without upsampling			With upsampling		
	Precision	Recall	F1	Precision	Recal	F1
Micro avg	0.12	0.77	0.21	0.24	0.63	0.35
Macro avg	0.07	0.54	0.10	0.14	0.46	0.19
Weighted avg	0.32	0.77	0.39	0.39	0.63	0.46
Samples avg	0.12	0.84	0.20	0.52	0.69	0.49

3.4.2.2 MODEL 2 - NLP FEATURE EXTRACTORS WITH ML MODELS

Comparing two feature extraction techniques, BoW and TF-IDF, the result depends on dimension reduction methods and ML models. TF-IDF significantly performs better across nearly all metrics with PCA ((1)&(3), (2)&(4)). It also performs better with TruncatedSVD and LightGBM ((6)&(8)). The beats are more significance on LightGBM model. BoW insignificantly wins in the rest cases. Therefore, TF-IDF is a better feature extractor.

Comparing two ML models, LightGBM significantly outperforms Random Forest across all metrics ((1)&(2), (3)&(4), (5)&(6), (7)&(8), (9)&(10), (11)&(12)). Macro-average and weighted-average F1-scores increments are between 0.05 and 0.09.

Comparing three dimension reduction techniques, setting max_features=300 in CounterVectorizer outperforms the others with BoW feature extractor ((1)&(5)&(9), (2)&(6)&(10)). However, its performance with TF-IDF feature extractor is unsatisfactory. PCA and TruncatedSVD's performances are hard to distinguish with TF-IDF ((3)&(7)&(11), (4)&(8)&(12)).

In conclusion, with our focus on Macro-average and weighted-average F1-score, PCA/TruncatedSVD+TF-IDF+LightGBM should be our final choice ((4)&(8)). However, there is no general rule that a combination can outperform across all metrics.

All performance of different techniques combination is shown in Table 4 below.

Table 4. Model 2 – NLP Feature Extractors with NL models comparison of performance

Dimension reduction method: PCA		BoW RF (1)	BoW LGBM (2)	TF-IDF RF (3)	TF-IDF LGBM (4)
Macro avg	Precision	0.15	0.33	0.21	0.43
	Recall	0.04	0.07	0.05	**0.11**
	F1 Score	0.05	0.10	0.06	**0.15**
Weighted avg	Precision	0.46	0.61	0.49	0.66
	Recall	0.31	0.34	0.30	0.37
	F1 Score	0.32	0.37	0.32	**0.41**
Samples avg	Precision	0.85	0.85	0.86	0.85
	Recall	0.47	0.50	0.46	0.51
	F1 Score	0.54	0.56	0.54	**0.58**
Dimension reduction method: TuncatedSVD		BoW RF (5)	BoW LGBM (6)	TF-IDF RF (7)	TF-IDF LGBM (8)
Macro avg	Precision	0.18	0.33	0.17	**0.45**
	Recall	0.05	0.08	0.05	**0.11**
	F1 Score	0.05	0.10	0.06	**0.15**
Weighted avg	Precision	0.49	0.59	0.46	**0.67**
	Recall	0.31	0.34	0.31	0.37
	F1 Score	0.32	0.37	0.32	**0.41**
Samples avg	Precision	0.86	0.82	**0.87**	0.83
	Recall	0.47	0.50	0.47	0.52
	F1 Score	0.54	0.56	0.55	0.57
Dimension reduction method: max_features=300		BoW RF (9)	BoW LGBM (10)	TF-IDF RF (11)	TF-IDF LGBM (12)
Macro avg	Precision	0.19	0.37	0.16	0.33
	Recall	0.05	0.11	0.05	0.10
	F1 Score	0.06	0.14	0.05	0.13

Weighted avg	Precision	0.52	0.61	0.52	0.59
	Recall	0.33	**0.39**	0.32	0.38
	F1 Score	0.34	0.43	0.33	**0.41**
Samples avg	Precision	0.87	0.80	0.86	0.81
	Recall	0.48	**0.53**	0.47	0.52
	F1 Score	0.56	0.57	0.55	0.57

3.4.2.3 MODEL 3 - EMBEDDING WITH PCA

Table 5 shows the performances of models with various PCA n_estimator parameters. Though the number of columns remaining increases from 123 to 638, the models' performances for all the metrics are almost constant. No improvement is observed along with the increasing number of columns.

Table 5. Model 3 performance across various PCA n_estimator parameters

	PCA(0.5)	PCA(0.75)	PCA(0.85)	PCA(0.95)
Number of columns remaining	123	310	430	638
F1 macro	0.04	0.04	0.04	0.04
F1 micro	0.43	0.43	0.43	0.42
F1 weighted	0.30	0.30	0.30	0.30
F1 sample	0.52	0.52	0.53	0.52

Therefore, high-dimension vectors for the text are not necessary for better performance. We might use smaller embeddings like all-MiniLM-L6-v2 that can still handle our multilabel classification task well but with better computational efficiency.

In terms of optimal hyperparameters, we ran multiple rounds of grid search to find the best set of parameters for the classifier. The summary of grid search results is listed below in Table 6:

Table 6. Model 3 summary of grid search results

learning_rate	max_depth	n_estimators	f1_macro	rank	f1_micro	rank	f1_weighted	rank	f1_samples	rank
0.01	3	300	0.038	21	0.420	20	0.293	21	0.526	**1**
…	…	…	…	…	…	…	…	…	…	…
0.1	3	300	0.046	4	0.432	**1**	0.315	4	0.522	6
0.01	7	300	0.038	20	0.421	17	0.296	20	0.521	7
…	…	…	…	…	…	…	…	…	…	…
0.2	7	200	0.041	13	0.425	8	0.306	8	0.513	20
0.2	3	100	0.050	**1**	0.425	10	0.316	2	0.513	21
…	…	…	…	…	…	…	…	…	…	…
0.2	3	300	0.048	3	0.426	6	0.316	**1**	0.509	27

Models with smaller maximum depth, like 3, outperform models with larger maximum depth. However, No single set of hyperparameter could outperform the others in terms of various the objectives, like f1_macro, f1_weighted and f1_sample. We might adjust the hyperparameters for the models according to the specific business need or purpose to achieve the most suitable outcome.

3.4.2.4 MODEL 4 - SPECIFIC EMBEDDING AND UPSAMPLING

The results for models with configurations detailed in Table 7 are as follows:

Table 7. Model 4 combinations performance

		Model 4a	Model 4b	Model 4c	Model 4d	Model 4e
Micro avg	Precision	0.83	0.83	0.84	0.81	0.74
	Recall	0.33	0.33	0.3	0.31	0.39
	F1 Score	0.47	0.47	0.44	0.45	0.51
Macro avg	Precision	0.22	0.20	0.15	0.18	0.25
	Recall	0.05	0.06	0.04	0.04	0.10
	F1 Score	0.06	0.07	0.05	0.05	0.12
Weighte d avg	Precision	0.56	0.54	0.46	0.50	0.54
	Recall	0.33	0.33	0.30	0.31	0.39
	F1 Score	0.34	0.35	0.32	0.32	0.41
Samples avg	Precision	0.87	0.87	0.88	0.85	0.79
	Recall	0.48	0.49	0.47	0.47	0.54
	F1 Score	0.56	0.57	0.55	0.55	0.58

Model 4a, which only includes TF-IDF features for the combined text, shows slow macro and weighted averages indicate that while the model can predict the dominant classes well, it fails to capture the minority classes effectively. Adding numerical features in Model 4b results in performance similar to Model 4a, suggesting that the numerical features didn't significantly contribute to distinguishing classes or weren't influential enough to improve recognition of minority classes.

Building upon Model 4b, Model 4c introduces embedded columns for summary, education and experience data. Here, macro and weighted averages are still low, and there is a marginal decrease in recall and F1 scores. This suggests the embeddings might introduce noise or complexity that the model struggles to generalize from.

Using MLSMOTE technique for upsampling, Model 4d aims to address class imbalance. Macro-average precision and F1 score see a slight improvement compared to 4c. This suggests that the model improved handling minority classes, due to the more balanced dataset provided by the upsampling technique.

Finally, using a stacking classifier instead of solely relying on random forest, Model 4e is used on top of all previous features and techniques. Here, there is a noticeable improvement across all scores, especially in macro and samples averages. The stacking approach, which combines the strengths of multiple learning algorithms, provides a significant performance boost, particularly in recognizing less frequent skills. This makes Model 4e ideal for applications requiring accurate prediction across the entire skill range on individual LinkedIn profiles. It manages the class imbalance well and improves the recognition of less frequent skills without compromising much on the overall precision.

After identifying Model 4e as the optimal choice, we aimed to ascertain whether the ratio of positive to negative labels significantly affects the model's performance. We set this threshold at 0.05, which narrowed the skill set to 18, and reran Model 4e. Furthermore, we incorporated domain knowledge relevancy into our analysis by requesting from ChatGPT a list of skills considered most relevant to equity research analysts (the primary profile within our dataset), resulting in 8 targeted skills for a subsequent run of Model 4e.

Table 8. Model 4e with further experiments

		Model 4e	Ratio threshold	Selected skills
Micro avg	Precision	0.74	0.74	0.67
	Recall	0.39	0.57	0.56
	F1 Score	0.51	0.64	0.61
Macro avg	Precision	0.25	0.19	0.17
	Recall	0.1	0.22	0.25
	F1 Score	0.12	0.2	0.19
Weighted avg	Precision	0.54	0.46	0.42
	Recall	0.39	0.57	0.56
	F1 Score	0.41	0.5	0.47
Samples avg	Precision	0.79	0.71	0.67
	Recall	0.54	0.68	0.66
	F1 Score	0.58	0.64	0.61

As seen in Table 8, the adjustment to a 0.05 threshold led to a marked improvement across all recall measures, with corresponding increases in the F1 score across micro, macro, weighted, and sample metrics. This improvement suggests that filtering skills by their frequency of positive instances enhances the model's generalization capability and its accuracy in identifying skills across profiles. Notably, there was a substantial increase in sample recall (from 0.54 to 0.68) and F1 score (from 0.58 to 0.64), which indicates a significant enhancement in the model's performance on individual profiles.

In contrast, the model refined to focus on the 8 skills identified through domain expertise yielded the highest recall scores across all variations, particularly in macro recall (0.25) and sample recall (0.66). Although there was a slight decrease in precision compared to the threshold-adjusted model, the F1 scores remained competitive. This boost in recall, especially in the macro aspect, demonstrates that selecting features based on domain knowledge can significantly increase the model's sensitivity to less frequently occurring classes.

3.4.3 COMPARISON BETWEEN DIFFERENT MODELS

This section compares the best performances of various models elaborated above in Table 9.

Table 9. Performance comparison between different models

		Model 1	**Model 2**	**Model 3**	**Model 4**
Precision	Micro avg	0.24	-	0.73	0.74
	Macro avg	0.14	0.45	0.14	0.25
	Weighted avg	0.39	0.67	0.41	0.54
	Samples avg	0.52	0.83	0.79	0.79
Recall	Micro avg	0.63	-	0.30	0.39
	Macro avg	0.46	0.11	0.04	0.10
	Weighted avg	0.63	0.37	0.30	0.39
	Samples avg	0.69	0.52	0.46	0.54
F1	Micro avg	0.35	-	0.42	0.51
	Macro avg	0.19	0.15	0.04	0.12
	Weighted avg	0.46	0.41	0.29	0.41
	Samples avg	0.49	0.57	0.53	0.58

In a business context, particularly for the task of resume screening in HR processes, each model offers distinct advantages that can be strategically employed depending on the specific needs of the recruitment strategy:

Model 1, with its superior recall scores, is particularly effective in scenarios where it is critical to capture as many qualified candidates as possible. This model minimizes the risk of missing potential candidates with relevant skills, which is crucial for roles where the talent pool is rare or highly specialized. It ensures a comprehensive inclusion of potential candidates, reducing the risk of overlooking skilled individuals due to model oversight. Besides, it has the highest macro-average F1-score, which demonstrates the extraordinary ability of pre-trained language model’s and neural networks in processing text data.

Model 2 shines in precision, making it ideal for contexts where the cost of evaluating false positives is high—such as in highly competitive fields or high-stakes positions where the volume of applications is large and processing resources are limited. This model helps focus HR efforts on applicants most likely to be a fit, thereby optimizing the use of time and resources in the recruitment process.

Model 4 offers the best balance between precision and recall, as evidenced by its leading performance in the samples average F1 score. This model is well-suited for generalist roles where it is important to maintain a balanced approach: accurately identifying suitable candidates while also ensuring a broad capture of potential talents. It is ideal for scalable HR processes where automation needs to effectively balance accuracy and coverage without significant manual oversight.

Choosing the right model depends on the strategic goals of the HR department:

- If the priority is to ensure no potential candidate is missed, Model 1 would be the choice.
- If reducing the workload of HR by minimizing less likely candidates is crucial, Model 2 would be beneficial.
- If the HR process requires a robust model that provides a reasonable assurance of precision and comprehensive coverage, Model 4 would be optimal.

Ultimately, integrating these models into HR software aims to enhance the efficiency and accuracy of resume screening processes, significantly reducing the time HR personnel spend on manual screening and improving the overall quality of candidate matching. These improvements are expected to not only streamline HR operations but also enhance satisfaction among HR professionals by allowing them to focus on more strategic and less repetitive tasks.

4. Insights and Discussion

4.1 Reflections on model design

In the experiments of various machine learning models, we have conducted solid comparison of different ways to process the text data and carry out the multilabel classification task.

The most complex model we tried is RoBERTa for sequence classification, which has 124 million parameters.

There are few advantages of RoBERTa Classification model. As it is pre-trained on massive amounts of text data, it excels at understanding context and semantics, making it suitable for sequence classification tasks. Additionally, as an end-to-end approach, LLM is more convenient to be trained and deployed as less pre-process and feature extraction are required.

There are also some noticeable drawbacks of this huge neural network. The model is resources-intensive for training and deployment. It also lacks interpretability due to its black-box nature.

Our best model tries to incorporate both embeddings and TF-IDF and then uses stacking classifier that consists of multiple base models. Since we learned from model 3 that reduced vectors still contain sufficient information to do the classification, we used all-MiniLM-L6-v2, which generate embedding with dimension of 384, to covert summary, education, and experience to vectors respectively and then concatenate them. We also intend to leverage the traditional TF-IDF method, which attempts to capture the importance of a word to vectorize the text data. Since we are not using the sophisticated pre-trained neural network as our classifier, and ensemble models consisting of random forest and XGB will be the final classification model, we try to incorporate the information extracted by TF-IDF to complement the model.

This model allows us to incorporate domain-specific features like languages, company names, etc. Its flexibility also allows us to add more information like user behavior once it is obtained. The traditional models also have high interpretability, offering the feature importance score to assist in understanding which features have the most influence on the predictions.

However, this model also has some drawbacks. It needed meticulous design in each step, data preprocessing, feature extraction and was prone to overfitting the train data. In the cross-validation results, the model performed almost 100% correctly in the training set.

4.2 Reflections on dataset

Despite attempting various text data preprocessing techniques and machine learning models, the performances of the models remain unsatisfactory. Macro-average F1-score is below 0.2 for all models. Sample-average ones are only around 0.5. Besides, we noticed that the difference between weighted-average and macro-average metrics are substantial, indicating that models perform significantly different across classes. From an example of class-level model performance shown in Table 10, we can see that Model 2 performs pretty well for classes with more or balanced positive values, but becomes lazy when positive ratios are low. However, our dataset is highly imbalanced, with only 2 skill classes balanced, 1 skill class 90% positively skewed, and rest of 47 classes less than 10% positively skewed on average (also mentioned in Section 2.2). The diversity of skills is not adequately supported by the labeled dataset.

Reported skills are too concentrated since the data mainly comes from candidates working in financial industries, so that most of them can report "Financial Analysis" skill but skills like "security", "telecommunications", and "software development" are less common in daily routine. We believe that our models can perform well with more adequate labels coming from profiles working in different industries.

Table 10. Class-level performance of Model 2 (selected skills)

	Analysis/ Financial Analysis/ Finance	Management/ Microsoft office / Customer service	Human resources/ Recruiting/ Performance mgt
Precision	0.93	0.68	1.00
Recall	0.99	0.57	0.04
F1 Score	0.96	0.62	0.08
Positive ratio	90.05%	43.68%	3.31%

Beside the skill classes, the quality and adequacy of profile data are also limited. Table 11 shows two examples of the profile data.

Table 11. Examples of profile data

User ID	Education Description	Experience Description
2619424	The University of Iowa Tippie College of Business, Master of Business Administration - MBA: Business Administration and Management, General 2021-2024	GuideOne Insurance: Senior Financial Analyst 2022.6-. **No description** Centene Corporation: Manager, Finance 2019.7-2022.6 **No description**
34763403	University of Rochester - Simon Business School MBA: Finance, Financial Accounting 1999-2001 Franklin & Marshall College BA: Spanish/History 1988-1992	J.L. Kaplan Associates, LLC : Sr. Equity Analyst 2006.6-2009.6 **Invested in small and mid capitalization domestic companies. I was primarily a generalist, but had responsibility for the packaging, homebuilding and consumer discretionary sectors.**

Some candidates provide detailed descriptions for their education and working experiences, while some only list the names of universities, majors, companies, and positions. It is challenging to determine an individual's skills or capabilities solely based on information such as professions and job titles. It is necessary to take their actual business practices and performance into consideration. We believe that our model designs would perform better with candidates' full CV as enhancements.

5. Implications, Further Work and Conclusions

BUSINESS IMPLICATIONS

Our models demonstrated promising capabilities in extracting and predicting skill sets from LinkedIn profiles, particularly enhancing the efficiency of the screening process by reducing manual efforts. Implementing this ML-driven system within HR operations can drastically reduce the time and costs associated with traditional resume screening. By automating the initial screening phase, HR professionals can allocate more time to qualitatively engage with potential candidates, improving the overall recruitment strategy. Moreover, the system's ability to de-bias the screening process supports a more equitable recruitment landscape.

However, there are certain implications worth noting for HR when considering adapting these models. Firstly, the exploration of different models revealed that while the models provide strong general accuracy, they struggle with precision and the recognition of less frequent skills, crucial in diverse job markets. Moreover, there is a notable difficulty in identifying rare skills, as evidenced by lower macro averages across the models. This indicates that while common skills are recognized efficiently, the system's ability to detect unique or less frequently listed skills requires improvement.

LIMITATIONS AND FURTHER WORK

The first major limitation pertains to data diversity. Our current dataset, while substantial in financial services industry, does not fully capture the vast heterogeneity of profiles needed to ensure robust model generalization and effectiveness across various industries. To address this, there is a pressing need to incorporate a more expansive and varied set of LinkedIn profiles, which would help the models better understand and adapt to the nuanced differences among candidates from diverse professional backgrounds.

Secondly, our approach requires ongoing innovation in model development, specifically through the adoption of hybrid models. These models would integrate the strengths of various advanced machine learning algorithms, creating a synergistic effect that could significantly improve the precision and recall of skill set predictions. This is particularly crucial for accurately identifying and classifying rare skill sets that are currently underrepresented or challenging to detect due to their infrequent occurrence in the dataset.

Lastly, the next steps in our project will include extensive pilot testing in actual HR operational settings. This phase is critical as it will allow us to refine the models in a live environment, gathering invaluable feedback directly from HR professionals. These insights will not only help in fine-tuning the technical aspects of the models but also in evaluating their practical impact on recruitment efficiency and the associated costs. This iterative process of testing and feedback is essential for transitioning from theoretical model performance to tangible, actionable results that can revolutionize HR practices.

In conclusion, this project represents a significant step forward in the application of AI in HR, pointing towards a future where recruitment is faster, more accurate, and equitable. Continued advancements in machine learning models and their integration into HR practices promise not only to enhance operational efficiencies but also to transform how organizations attract and retain talent. As we expand our dataset and refine our methodologies, the potential to support HR professionals in making informed decisions will undoubtedly grow, reinforcing the essential role of human judgment enriched by AI insights.

References

Bahr, K. (2021, August 3). Resume screening. Eddy. https://eddy.com/hr-encyclopedia/resume-screening/

Charte, F., Rivera, A. J., del Jesus, M. J., & Herrera, F. (2015). MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation. Knowledge-Based Systems, 89, 385–397. https://doi.org/10.1016/j.knosys.2015.07.019

Nordmark, V. (2020, April 5). 5 reasons you should be using automated resume screening. Hubert. https://www.hubert.ai/insights/5-reasons-you-should-be-using-automated-resume-screening