Does Unlabelled data improve Job salary prediction

Shaolong Xu

17/5/2023

1 Introduction

This report delves into the evolving field of job salary prediction, a crucial element in today's job market landscape for both employers' budgeting and job seekers' decisions. Harnessing the power of semi-supervised learning methods, we explore the potential impact of unlabelled data, such as unstructured job descriptions, on the efficacy of predictive models. We focus on three classifier models: decision tree, logistic regression, and stacking classifier. Each undergoes rigorous preprocessing and adaptation to ensure optimal performance. This report aims to assess the impact of incorporating unlabelled data on these models' performance, thereby offering fresh insights into data-driven salary prediction and contributing to the progressive discourse on labour market analytics.

2 Literature review

This study leverages a dataset provided by Bhola et al. (2020), composed of 13,902 job descriptions, with 8,000 labelled and 5,902 unlabelled entries. Further sets for model tuning and final evaluation are also available.

The increasing accessibility of unlabelled data, such as unstructured job descriptions or uncategorized employee reviews, paves the way for semi-supervised learning (Yang et al., 2021). This approach combines labelled and unlabelled data to create more comprehensive and accurate models, especially beneficial when labelled data acquisition is challenging or costly.

However, semi-supervised learning also presents unique challenges. Self-training, an iterative method within this field, employs a 'teacher' model to assign pseudo-labels to unlabelled data. Despite its potential, this approach risks introducing noisy pseudo-labels from incorrect 'teacher' predictions, leading to a 'confirmation bias' problem over time (Radhakrishnan et al., 2023). This issue underscores the need for careful implementation of semi-supervised learning methods.

3 Method

Identify the newly engineered feature(s), and the rationale behind including them (Optional). Explain the ML models and evaluation metric(s) you have used (and why you have used them)

In this report, various approaches and techniques are employed in terms of data pre-processing, model selection and fine-tuning, as well as results assessment to ensure the robustness and reliability of the findings obtained.

3.1 Data Pre-Processing

3.11 Data representation

In this study, three data representations were used: Raw, TF-IDF, and embedding. Raw data remained unchanged, TF-IDF measured word importance, and embedding captured semantic connections. This diverse approach aimed to explore the predictive potential of job salary data thoroughly.

3.12 Encoding

To utilize categorical raw data in machine learning models, it was necessary to convert it into numerical form. This enabled the models to learn from non-numeric data and expanded their learning capabilities. Additionally, one-hot encoding was applied to TF-IDF data, transforming words into binary features, which is advantageous for algorithms that require binary or categorical inputs.

3.13 Feature Engineering

In our methodology, feature engineering included merging TF-IDF and embedding data to form a comprehensive feature set for our models. By joining the context-sensitive importance of terms from TF-IDF data with the nuanced semantic relationships from embeddings, we expected to achieve a robust representation of text data for our salary prediction models.

3.14 Create Pseudo-labelled Data

The unlabelled data were processed using a pre-trained model to generate pseudo-labels, effectively transforming them into labelled data. This pseudo-labelled data was then concatenated with the original labelled data. This augmented dataset was utilized to train a self-training model, facilitating semi-supervised learning.

3.15 Observation and Visualization of Label Data

We thoroughly analysed and visualized our target label, salary bands. Table 1 displays each band's minimum, maximum, and interval values, highlighting the potential impact of interval size on prediction accuracy. Larger intervals, encompassing broader value ranges, may increase prediction errors.

Band	Min Value	Max Value	Interval
0	0	56	56
1	57	75	18
2	76	98	22
3	99	119	20
4	120	148	28
5	149	175	26
6	176	215	39
7	216	245	29
8	246	310	64
9	311	455	144

Table 1-Maximumm, minimum and interval value of each band

Figures 1 and 2 visually display the salary band distribution in training and development datasets. These plots highlight the imbalance in data representation, with band 5 showing the least instances, potentially impacting prediction accuracy due to limited training examples.

Figure 1- Salary Band vs Instance Number (Training Dataset)

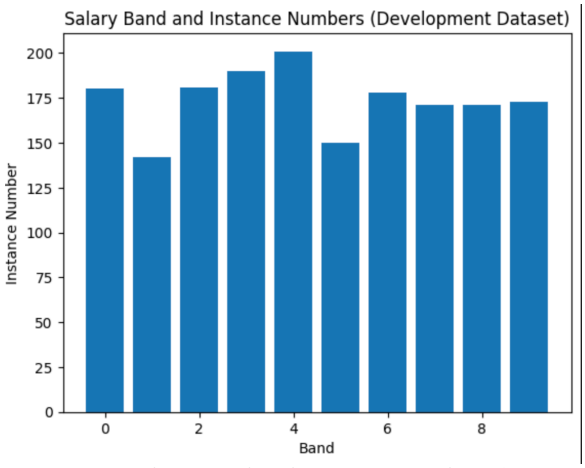

Figure 2- Salary Band and Instance Numbers (Development Dataset)

3.2 Model selection and adjustment

Model selection was informed by an initial overview of the accuracy of 10 models across various datasets (Table 2). We chose three models exhibiting distinct mechanisms for further analysis. This approach was designed to evaluate the impact of unlabelled data across diverse computational methods.

Model	Accuracy (Embedding)	Accuracy (TF-IDF)	Accuracy (Raw)	Accuracy (TF-IDF-OneHot)	Accuracy (TF-IDF and Embedding)
GNB	0.229131	0.219919	1	0.212435	0.10593
one-r	0.166955	0.156592	0.202073	0.156592	0.103627
1-nearest neighbour	0.231434	0.177317	0.150835	0.186528	0.10593
5-nearest neighbour	0.217617	0.146805	0.128958	0.161773	0.0857801
Decision Tree	0.150259	0.17156	0.393207	0.17559	0.112263
Logistic Regression	0.243523	0.218192	0.112263	0.189983	0.109384
Perceptron	0.181923	0.16753	0.130109	0.173863	0.10593
Neural Network	0.240069	0.222222	0.231434	0.217617	0.111687
Bagging	0.244099	0.213011	0.111687	0.201497	0.100173
Boosting	0.230858	0.215314	0.17156	0.194588	0.114565

Table 2-Accuracy of Each Model under Different Dataset

3.21 Model Tunning

Model tuning is essential for maximizing algorithm performance. Although default hyperparameters often suffice, they may not yield optimal results for specific applications (Iguazio, n.d.). Therefore, adjusting these parameters to our data is a critical step.

3.211 Model Tuning: Decision Tree

The decision tree model undergoes tuning by varying its depth, and its accuracy is calculated for training and development data sets. The results are presented in Tables 3 and 4, and learning curves in Figures 3 and 4, leading to identification of an optimal depth.

3.212 Model Tuning: Logistic Regression

To optimize the performance of the logistic regression model, we experiment with different combinations of solvers ('lbfgs', 'liblinear', 'newton-cg') and regularization parameters (C values: 10, 1.0, 0.1) on both the development and training data. The accuracy of the logistic regression model with varying hyperparameter combinations is illustrated in Tables 5 and 6, as well as Figures 5 and 6. This comprehensive assessment allows us to identify the optimal hyperparameter settings for our logistic regression model.

3.213 Stacking Classifier

This study utilized a model tuning approach for the Stacking Classifier, selecting the top 6 models based on accuracy. Various model combinations were evaluated, resulting in 42 distinct combinations. Table 7 displays the accuracy results for 5-model combinations, while Tables 8 and 9 outline the top 5 accuracies for 4-model and 3-model combinations, respectively.

3.3 Results assessment

The impact of integrating unlabelled data was evaluated by comparing the model's performance using accuracy, precision, recall, F1-score, and a confusion matrix. Results are presented in Tables 10-12 and Figures 7-12.

4 Results

4.1 Dataset

Based on the analysis of tables 2-6 and figures 3-6, the raw data had lower accuracy and tended to overfit. In contrast, the other four datasets showed consistent and reliable performance. Notably, the embedding dataset performed exceptionally well in terms of accuracy and stability. Therefore, the embedding dataset will be selected for further analysis

4.2 Decision Tree

Based on Table 3, 4 and Figures 3, 4, a depth of 6 was selected as the optimal depth for the Decision Tree model.

Different depth of DT	Accuracy (Embedding)	Accuracy (TF-IDF)	Accuracy (Raw)	Accuracy (TF-IDF-OneHot)	Accuracy (TF-IDF and Embedding)
Depth 1	0.166955	0.156592	0.102476	0.156592	0.103627
Depth 5	0.170409	0.176742	0.712723	0.175014	0.108233
Depth 7	0.172136	0.177893	0.419689	0.176742	0.108808
Depth 10	0.169257	0.164652	0.704663	0.192861	0.113414
Depth 20	0.148532	0.173287	0.410478	0.190043	0.111111
Depth 35	0.146229	0.180771	0.394358	0.181923	0.111111

Table 3- Accuracy of Decision Tree with Varying Depths on Different Datasets(Development Data)

Different depth of DT	Accuracy (Embedding)	Accuracy (TF-IDF)	Accuracy (Raw)	Accuracy (TF-IDF-OneHot)	Accuracy (TF-IDF and Embedding)
Depth 1	0.164625	0.15875	0.2195	0.15875	0.164625
Depth 5	0.2185	0.200125	0.825375	0.20425	0.21725
Depth 7	0.28825	0.231	1	0.248	0.274875
Depth 10	0.45025	0.293625	1	0.3335	0.409625
Depth 20	0.96025	0.5275	1	0.66475	0.887625
Depth 35	0.999875	0.775875	1	0.88125	1

Table 4- Accuracy of Decision Tree with Varying Depths on Different Datasets(Training Data)

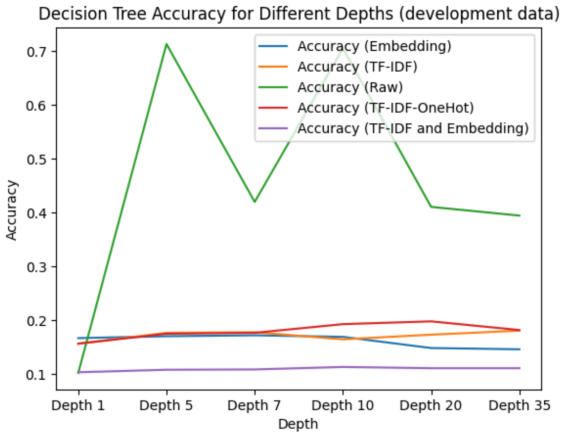

Figure 3- Accuracy of Decision Tree with Varying Depths on Different Datasets(Development Data)

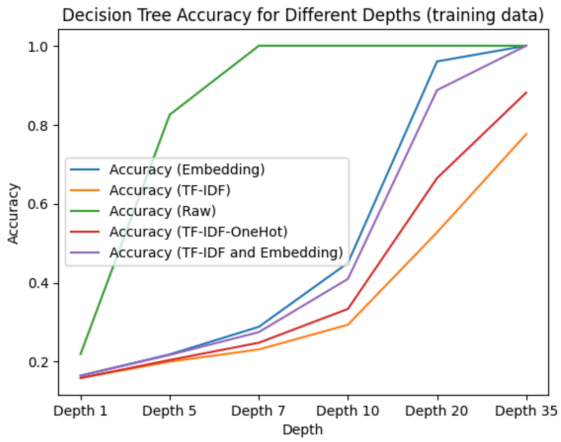

Figure 4- Accuracy of Decision Tree with Varying Depths on Different Datasets(Training Data)

4.3 Logistic Regression

As shown in Tables 5, 6 and Figures 5, 6, the default hyperparameters of 'lbfgs' as the solver and 1.0 as the C value were chosen for Logistic Regression.

Hyperparameter tuning	Accuracy (Embedding)	Accuracy (Raw)	Accuracy (TF-IDF)
Solver: lbfgs, C: 10	0.230858	0.112263	0.208405
Solver: lbfgs, C: 1.0	0.242372	0.112838	0.218192
Solver: lbfgs, C: 0.1	0.237766	0.112263	0.227979
Solver: liblinear, C: 10	0.233736	0.248129	0.204951
Solver: liblinear, C: 1.0	0.241796	0.204375	0.219344
Solver: liblinear, C: 0.1	0.233161	0.221071	0.229131
Solver: newton-cg, C: 10	0.230858	0.189983	0.206102
Solver: newton-cg, C: 1.0	0.243523	0.298215	0.218192
Solver: newton-cg, C: 0.1	0.237766	0.407024	0.227979

Table 5- Accuracy of Logistic Regression with Varying Hyperparameter Tuning on Different Datasets (Development Data)

Hyperparameter tuning	Accuracy (Embedding)	Accuracy (Raw)	Accuracy (TF-IDF)
Solver: lbfgs, C: 10	0.365375	0.30575	0.408125
Solver: lbfgs, C: 1.0	0.3295	0.30325	0.380125
Solver: lbfgs, C: 0.1	0.277625	0.3055	0.30675
Solver: liblinear, C: 10	0.37075	0.6525	0.402125
Solver: liblinear, C: 1.0	0.324625	0.6345	0.374875
Solver: liblinear, C: 0.1	0.272375	0.595875	0.3015
Solver: newton-cg, C: 10	0.372375	0.87175	0.408625
Solver: newton-cg, C: 1.0	0.328875	0.945	0.380375
Solver: newton-cg, C: 0.1	0.277625	0.908625	0.30675

Table 6- Accuracy of Logistic Regression with Varying Hyperparameter Tuning on Different Datasets (Training Data)

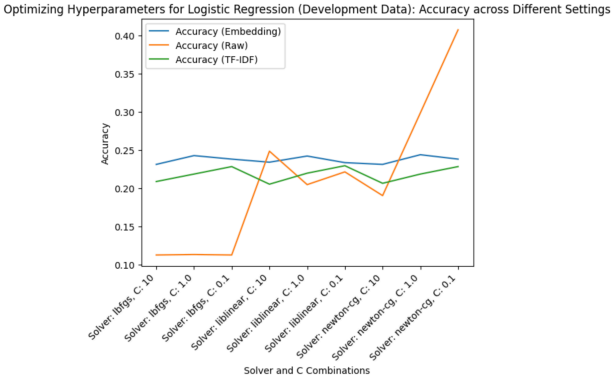

Figure 5- Accuracy of Logistic Regression with Varying Hyperparameter Tuning on Different Datasets (Development Data)

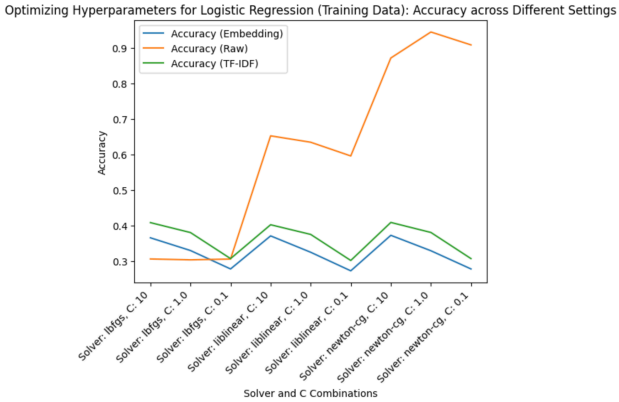

Figure 6- Accuracy of Logistic Regression with Varying Hyperparameter Tuning on Different Datasets (Training Data)

4.3 Stacking Classifier

As per Tables 7, 8, 9, the final optimised Stacking Classifier was configured using a combination of five models - Gaussian Naive Bayes, 1 Nearest Neighbour, Logistic Regression, Neural Network, and Boosting Classifier.

Other accuracies of combination are shown below.

Model Combination	Accuracy (Embedding)
GNB+1-nearest neighbour+Logistic Regression+Neural Network+Bagging	0.256765
GNB+1-nearest neighbour+Logistic Regression+Neural Network+Boosting	0.261946
GNB+1-nearest neighbour+Logistic Regression+Bagging+Boosting	0.25331
GNB+1-nearest neighbour+Neural Network+Bagging+Boosting	0.257916
GNB+Logistic Regression+Neural Network+Bagging+Boosting	0.244675
1-nearest neighbour+Logistic Regression+Neural Network+Bagging+Boosting	0.261946

Table 7- Accuracy of Stacking Classifier with 5-Model Combination

Model Combination	Accuracy (Embedding)
1-nearest neighbour+Logistic Regression+Neural Network	0.261946
1-nearest neighbour+Neural Network+Bagging	0.26137
1-nearest neighbour+Neural Network+Boosting	0.259643
GNB+1-nearest neighbour+Neural Network	0.257916
1-nearest neighbour+Logistic Regression+Bagging	0.255037

Table 8- Top 5 Accuracy of Stacking Classifier with 4-Model Combination

Model Combination	Accuracy (Embedding)
1-nearest neighbour+Neural Network+Bagging+Boosting	0.262522
1-nearest neighbour+Logistic Regression+Neural Network+Bagging	0.261946
1-nearest neighbour+Logistic Regression+Neural Network+Boosting	0.261946
GNB+1-nearest neighbour+Logistic Regression+Neural Network	0.26137
GNB+1-nearest neighbour+Neural Network+Bagging	0.258492

Table 9- Top 5 Accuracy of Stacking Classifier with 3-Model Combination

For the 6-model combination, it resulted in an accuracy of 0.25676.

4.5 Incorporating Unlabelled Data into Model

The table and figure in this section demonstrate the effects of incorporating unlabelled data into various models.

4.51 Decision Tree

Model	Precision	Recall	F1	Accuracy
Decision Tree	0.14762	0.16475	0.14383	0.16637
Decision Tree Self training	0.15706	0.16746	0.14404	0.16580

Table 10- Comparison of Precision, Recall, F1 Score, and Accuracy Before and After Integration of Unlabelled Data to Decision Tree

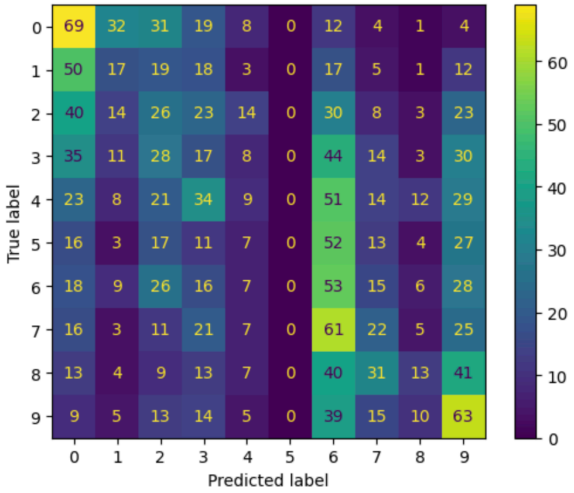

Figure 7- Confusion Matrix of Decision Tree without Incorporation of Unlabelled Data

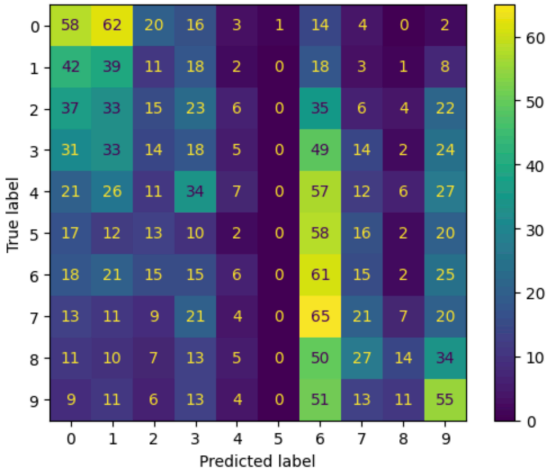

Figure 8- Confusion Matrix of Decision Tree Model with Incorporation of Unlabelled Data

4.52 Logistic Regression

Model	Precision	Recall	F1	Accuracy
Decision Tree	0.21792	0.24004	0.21940	0.24237
Decision Tree Self training	0.21870	0.24063	0.22028	0.24294

Table 11- Comparison of Precision, Recall, F1 Score, and Accuracy Before and After Integration of Unlabelled Data to Logistic Regression

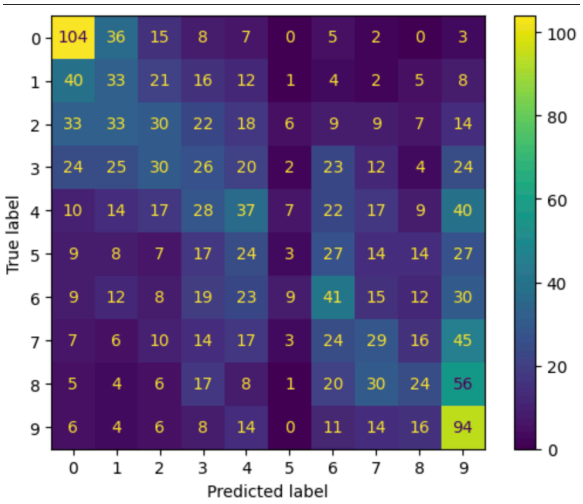

Figure 9- Confusion Matrix of Logistic Regression without Incorporation of Unlabelled Data

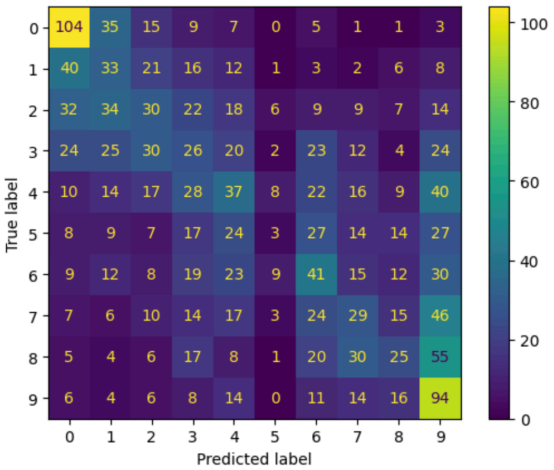

Figure 10- Confusion Matrix of Logistic Regression Model with Incorporation of Unlabelled Data

4.53 Stacking Classifier

Model	Precision	Recall	F1	Accuracy
Stacking Classifier	0.24907	0.25883	0.24103	0.26137
Stacking Classifier Self training	0.23829	0.24909	0.23275	0.25331

Table 12- Comparison of Precision, Recall, F1 Score, and Accuracy Before and After Integration of Unlabelled Data to Stacking Classifier

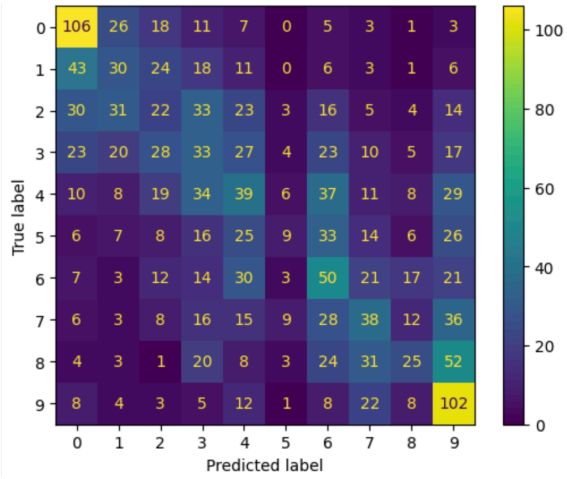

Figure 11- Confusion Matrix of Stacking Classifier without Incorporation of Unlabelled Data

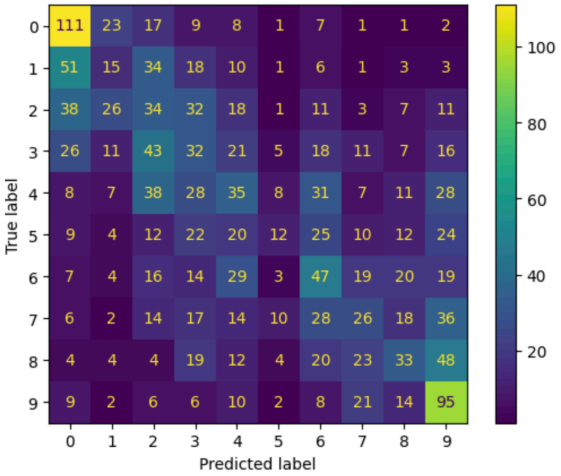

Figure 12- Confusion Matrix of Stacking Classifier Model with Incorporation of Unlabelled Data

5 Discussion / Critical Analysis

5.1 Dataset

As per the findings in subsection 4.1, the embedding dataset outshone the rest. Two new datasets were devised, but neither outperformed the embedding data.

The first new dataset converted TF-IDF data to one-hot encoding, a step that turned out to be unnecessary. TF-IDF, a numeric representation, already provides term frequency and significance information, all of which is lost in one-hot encoding.

The second dataset, a fusion of TF-IDF and embedding data, aimed to harness the strengths of both. Unfortunately, it fell short, possibly due to increased complexity and dimensionality causing noise and overfitting.

5.2 Model

5.21 Decision Tree

The optimal depth for the Decision Tree model was determined to be 6. Figures 3 and 4 show that a depth of 6 maximized accuracy, while lower depths compromised accuracy and larger depths risked overfitting without significant accuracy improvement. However, despite setting the optimal depth, the overall accuracy of the Decision Tree model remained relatively low, as observed in Tables 3 and 4 and depicted in Figure 7.

5.22 Logistic Regression

Under nine configurations, we tested our logistic regression model, using a mix of three solvers 'lbfgs', 'liblinear', 'newton-cg', and three regularization strengths (C values) 10, 1.0, 0.1. As Tables 5 and 6 and Figures 5 and 6 show, most combinations yielded similar accuracies, leading us to retain default settings.

The choice of solver and regularization strength, which guide model optimization and complexity respectively, is crucial in logistic regression. However, our data's potential linear separability resulted in consistent results across combinations. This suggests minimal initial overfitting, lessening the effect of regularization changes.

5.23 Stacking Classifier

We utilized six classifiers for Stacking, which yielded higher accuracy compared to non-ensemble models. Using models with diverse mechanisms for stacking capitalizes on their individual strengths and compensates for their weaknesses. This diversity enhances the model's ability to generalize by leveraging the different learning mechanisms, thus improving overall performance.

5.24 Confusion Matrix

Analysing the confusion matrices in Figures 7, 9, and 11, we note that lighter colours represent higher prediction counts for respective labels. All models accurately predict labels 0 and 9, likely due to more data and larger prediction intervals, as seen in Table 1, Figures 1 and 2.

For the logistic regression and stacking classifiers, the lighter diagonal from top-left to bottom-right compared to the top-right and bottom-left corners indicates an overall correct prediction trend. Misclassifications are generally close to the correct label.

In contrast, the decision tree's confusion matrix is less structured, signifying lower accuracy. Particularly, no predictions were made for label 5, potentially due to less available data or model constraints, as supported by Figures 1 and 2. Even the better-performing models exhibit a similar issue, with darker colours for predicted label 5, indicating fewer correct predictions for this class.

5.3 Results Evaluation

Our analysis, as shown in Tables 10-12, revealed that the introduction of unlabelled data into our models did not lead to significant changes in precision, recall, F1 score, or accuracy. According to Figure 7-12, the confusion matrices for these models also remained largely unchanged. In fact, the incorporation of unlabelled data somewhat diminished the performance of the stacking classifier, our most robust model.

This outcome can be attributed to the self-training model's mechanism. In self-training, the model uses its own predictions on unlabelled data to generate additional training data. If the model's predictions are incorrect, these errors can get propagated back into the training process, leading to a decrease in performance.

In our case, it's plausible that the unlabelled data was of insufficient quality or not relevant enough to the task at hand. Consequently, when these data were used in self-training, it may have led to incorrect pseudo-labels, introducing noise into the model. As a result, the models trained with these pseudo-labels did not improve the salary prediction task, highlighting the importance of the quality and relevance of unlabelled data in semi-supervised learning.

6 Conclusion

This report has comprehensively explored the impact of incorporating unlabelled data in job salary prediction models using semi-supervised learning techniques. Despite the potential merits of utilizing unlabelled data, our investigation indicates that its inclusion did not improve salary prediction. It is plausible that the quality of the unlabelled data was not adequate, leading to flawed predictions that then influenced the self-training model negatively. In conclusion, while unlabelled data holds promise in various data science applications, its value for salary prediction appears limited under the conditions of our study. Future investigations may focus on refining data quality for improved outcomes.

References

Bhola, A., Halder, K., Prasad, A., & Kan, M.-Y. (2020). Retrieving Skills from Job Descriptions: A Language Model Based Extreme Multi-label Classification Framework. In Proceedings of the 28th International Conference on Computational Linguistics (pp. 5832–5842). Barcelona, Spain (Online). International Committee on Computational Linguistics.

Radhakrishnan, A., Davis, J., Rabin, Z., Lewis, B., Scherreik, M. and Ilin, R., 2023. Enhancing Self-Training Methods. arXiv preprint arXiv:2301.07294.

Iguazio. (n.d.). Model Tuning. Available at: https://www.iguazio.com/glossary/model-tuning/#:~:text=Tuning%20a%20machine%20learning%20model,model%20learns%20its%20trainable%20parameters. [Accessed: 10 May 2023].

Yang, X., Song, Z., King, I., Xu, Z. (2021). A Survey on Deep Semi-supervised Learning. arXiv preprint arXiv:2103.00550v2 [cs.LG], 23 Aug 2021.