COMP90049 Report

1 Introduction

The research topic for this report is if unlabeled data improve Job salary prediction. We predict that unlabeled data can improve Job salary prediction after semi-supervised learning or active learning. As part of the project, Bhola et al. (2020) provide three sets of data, including a training data set, a development data set, and a test data set. There are 8000 job descriptions with labels and 5902 job descriptions without labels in the train data set. For tuning and model selection, we can use the 1738 label data in the development set. There are 1737 data points in the test set used for Kaggle competitions. Due to a large number of unlabeled targets and limited label data in the train data set, the accuracy of the job salary prediction is too low, so we want to think of ways to improve it with different appropriate models and get more label data.

2 Literature review

Model accuracy is influenced by data, especially label data. In contrast, Zhu and Ghahramani (n.d.) state that label data can be helpful for supervised learning, however those label data are often limited in quantity, while unlabeled data can be found in large quantities. This paper discusses how to combine labelled data with unlabeled data to see whether the unlabeled data can enhance model performance through self-training.

2.1 Data augmentation

One of the most common ways to mitigate the need for labelled data is by data augmentation. Normally, data augmentation can be challenging since it involves combining labelled and unlabeled data. Therefore, an effective form of data augmentation can be important for model classification and dealing with lack of label data. Based on Chen et al.'s (n.d.) findings, pseudo labels can be an effective method for achieving data augmentation by iteratively assigning to unlabeled samples in semi-supervised learning. Since augment the original data set, according to the Vu et al. (n.d.) research, demonstrate the newly formed labelled dataset by data augmentation is critical to a successful of NLP deployment. As a result, data augmentation by the pseudo label is the key factor to increase model data set.

2.2 Active learning VS Self-tranining

Initially, active learning is our priority method for selecting unlabeled data. According to Maja Stikic et al. (2008) we can determine that as part of active learning, unlabeled samples are automatically detected and asked to be labelled by the oracle. However, the problem is that we do not have a human annotator in this model, so finally we decided to use semi-supervised learning to predict unlabeled data.

The self-training is semi-supervised learning which can help model learn from labelled and unlabeled data. As stated by Pise and Kulkarni (2008), self-training can be used to establish a threshold for utilizing large amounts of unlabeled data on training, which are usually composed of the most reliable unlabeled data. According to Tang et al. (2021), self-training setups enhance deep learning model accuracy in different tasks. Thus, self-training is an effective method for improving model performance using unlabeled data.

3 Method

The dataset contains labeled and unlabeled data, so we need to preprocess those data sets to make sure they are valid for tuning and selection. Several classification models will be applied to different datasets, and the appropriate type of data will be chosen based on the comparison between these models. Optimize the models through hyperparameter optimization, and feature selection also affects the model per-

formance. We can use self-training to augment unlabeled data once we get the best model with best hyperparameters and features. Lastly, accuracy comparison from self-training to evaluate if unlabeled datasets can improve salary bin prediction.

3.1 Hypotheses

There are three hypotheses will be discussed for this research.Firstly, unlabeled data may contain useful information that can be used for model improvement through semi-supervised learning or active learning. In addition, different classification models may perform differently on different datasets. Therefore, it is necessary to compare them to select the appropriate model for a comprehensive evaluation. Lastly, feature representation has a significant impact on model performance, so choose the suitable feature representation can gain more information from the datasets.

3.2 Data Pre-processing

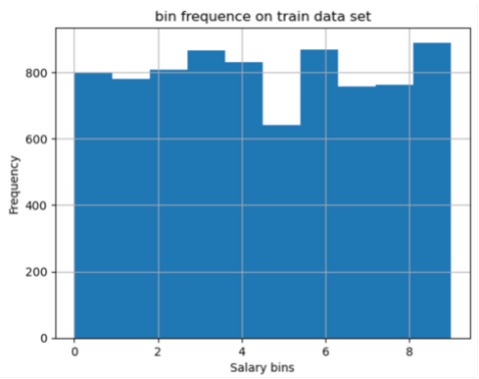

Figure 1: Salary bin frequency

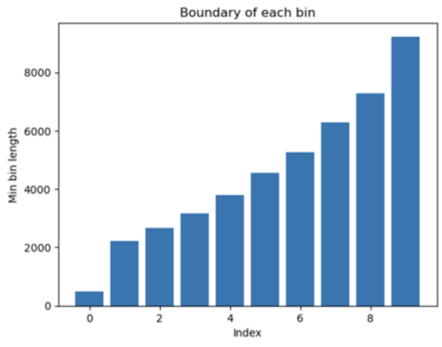

Figure 2: Boundary of each bin

Firstly, there are three different data set stored in different types of files, we need to read all these data at the beginning. Due to the training dataset have limited label data, and there are still 5902 unlabeled data. To separate those label and unlabeled data are essential, because only label data can be used for training. Secondly, target labels are mean salary with corresponding salary bin, and the job salary prediction can be influenced by the bin frequency and bin size. Therefore, we need to find out the salary band of those target label and frequency bins. From figure 1, we can figure out salary bin in train data set is not equal frequency. The amount of bin 5 is the lowest, and the amount of bin 6 and bin 9 is the highest. From figure 2, the bin length of 0 and bin length of 9 is the biggest.

3.3 Model selection

We used the decision tree as our baseline model, since it is a simplistic model when compared to other models. However, each model has a different algorithm, hyperparameter, and limitations. To determine the comprehensive performance of this dataset. The different models will perform differently on different data sets. As part of our task, we will use some well-known models, including the Gaussian NB, Decision Tree, KNN, logistic regression and Multi-layer perceptron model, on a dataset to get a comprehensive analysis of their performance based on accuracy.

3.4 Feature representation selection

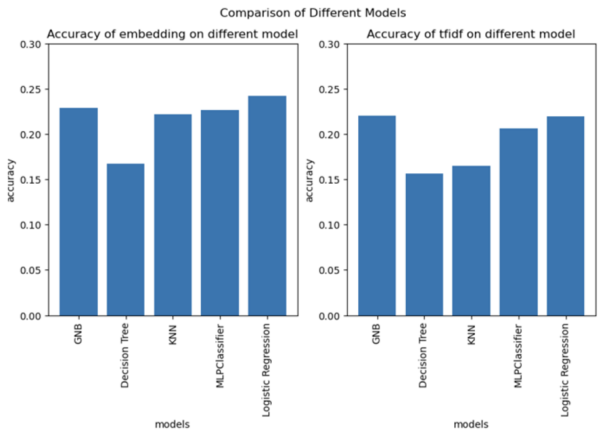

Figure 3: Model accuracy on Feature representations

The project contains three different feature representations: raw, TFIDF, and embedding. There may be some outliers or significant differences in the raw feature set which may adversely affect the performance of the classification model. Thus, we used TFIDF and embed-

ding data for pre-processing to improve classification performance. We constructed a bar chart to compare which feature representation performs best on the different models. In figure 3, embedding features in each model are more accurate than TFIDF. Four models perform over 0.22 on embedding, but only the GNB model performs above 0.22 on TFIDF. To obtain better performance on different models, we would choose the embedding feature representation.

3.5 Model Tuning

Model	Best Parameters	Best Score
GaussianN	{'var_smoothing': 1e-10}	0.21530524363178644
Decision tree	{'max_depth': 2}	0.18250852959687303
KNN	{'n_neighbors': 21, 'p': 1}	0.2066746165821988
MLPClassifier	{'random_state': 2, 'max_iter': 100, 'learning_rate': 'adaptive'}	0.20320149723409192
LogisticRegression	{'multi_class': 'auto', 'penalty': 'l2', 'solver': 'liblinear'}	0.23603133591705588

Figure 4: Best parameters on different models

To improve the performance of each model, the model tuning on hyperparameters is a crucial step. To determine the optimal hyperparameters, we decide to use two approaches. First, we set some parameters, such as depth, which is crucial to the decision tree's performance. Following that, I provide a range of values for those parameters, and we use grid search to determine the most effective combination of those hyperparameters. To prevent model overfitting and underfitting, we need to cross-validate the grid search. Trying all possible combinations of parameters is time-consuming with the Multi-layer Perceptron. As a result, a random search is a better approach since it doesn't need to try all possible combinations to find the best one.

3.6 Feature selection

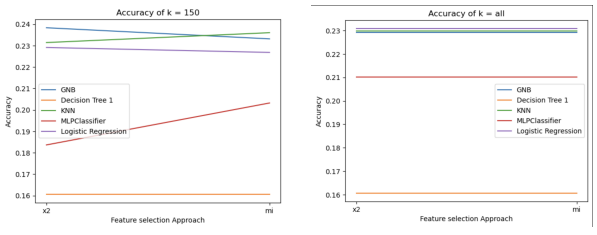

Figure 5: Accuracy on embedding with different K

Feature selection is another influence on the model performance, so we decide to use 'SelectKBest' method and chi-square and mutual information as score functions to get comprehensive evaluation on feature selection. Firstly, we try to divide the feature amount into 3 parts, which are 100, 300 and all features. Since the chi-square and mutual information can only compute the non-negative data, we use the min-max to scale the TFIDF feature from 0 to 1. Then, these score functions, which only accept non-negative values, are applied to the embedding and TFIDF feature representations to train and test different models. Finally, we use the line graph to display each model's accuracy under different amounts of features.Based on Graphs 5 and 6, we can conclude that the model with all embedding features may have better performance, as the majority of models achieved the highest accuracy of 0.23 with different score functions. And once again, it was validated that embedding feature representation is better than TFIDF feature representation with all features.

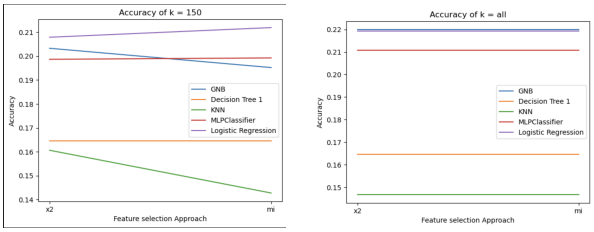

Figure 6: Accuracy on TFIDF with different K

3.7 Self-training

Self-training plays a crucial role in this research. In order to combine the labeled data with the unlabeled data, we must first augment the train data set. Thus, we need to assign a negative value to the unlabeled data that cannot affect the original label bin and combine the label and pseudo labels. The next step is to use self-training to predict those pseudo labels, as well as to set the threshold for self-training in order to find the most confident prediction to expand the label data set. At the beginning, we try to use 0.9 as our threshold, however, the accuracy of the models will not change after self-training. Because the high value of threshold cannot capture any pseudo labels. Thus, we use a smaller threshold value, allowing each model to capture more pseudo labels and expand the data set.

4 Result

After obtaining the results from self-training, using the Gaussian NB and KNN model as our test target to create the confusion matrix. By comparing the average macro before and after

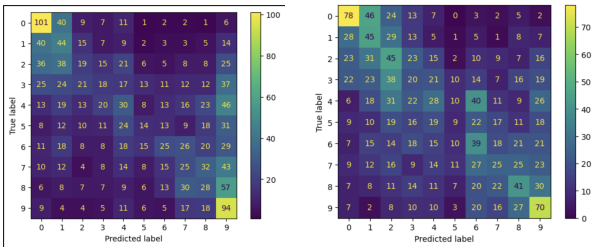

Figure 7: Confusion matrix before self-training

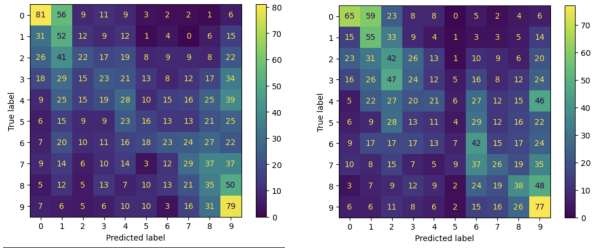

Figure 8: Confusion matrix after self-training

self-training from Figure 7 and Figure 8, we observed a decrease in the number of true positives, particularly for bins 0 and 9, which experienced a decrease of around 10-20 true positives. In addition, the number of false negatives increased, therefore, the precision was decreased after self-training.

Also, it is important to note that after self-training, the data was augmented, resulting in a total of 13902 labels. In Figure 9, we can see that the maximum accuracy of each model decreases by approximately 1-2 percent as a result of self-training. Although the multi-layer perceptron showed improved performance after semi-supervised learning, the other five models demonstrated a decrease in accuracy, particularly the stacking model.

5 Critical Analysis

The following statements address two key points. The model performance will differ due to different model characteristics, so we would decrease accuracy if we used the wrong model on non-equal frequency bin or non-linear datasets after data augmentation. By using the wrong model, more incorrect labels may be produced that can be used for training the next round of models. Therefore, when performing larger label data sets, it is important to use the correct model. As an example, the decision tree should be used in even distributed datasets, however, after we complete the data preprocessing, we find that the salary bin is not equally distributed as specified in figure 1. Consequently, when we apply the incorrect model to this project, figure 3 demonstrates the model with the lowest accuracy as a result of the incorrect model propagating the error.

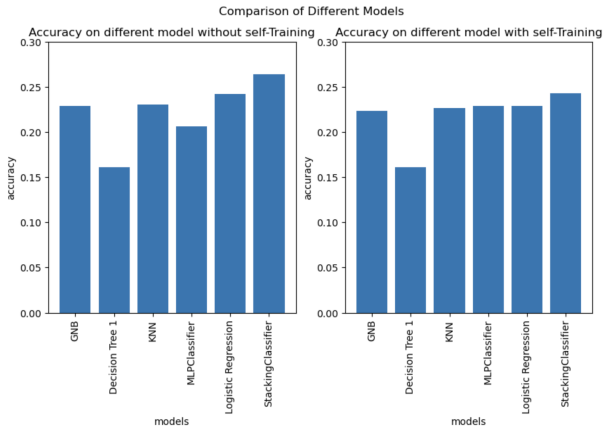

Figure 9: Self-training on different models

Additionally, the limited label dataset would be a second reason for reducing self-training performance. There are 13902 data points, but only 8000 label data can be used for training. As the self-training can only be learned from label data, the error can propagate after self-learning. The prediction accuracy was significantly reduced after self-training, as shown in figure 7 and figure 8. More wrong prediction bins are located in close proximity to the true label bins. Because the ability of the model learning is limited by the size of the labelled data, self-training learns more incorrect information from unlabeled data. For example, the stacking model, which combines four models, may lead to error propagation and decrease the original accuracy. As a result of the limited label size, the self-training cannot learn some useful information, error predictions were learned which resulted in error propagation and lower original salary bin predictions.

6 Conclusion

The goal of this report was to explore if unlabeled data improve Job salary prediction after semi-supervised learning, we trained and test different models on both the 8000 labelled dataset and the 13902 labelled dataset after data augmentation. From the results, we can conclude that the unlabeled data cannot increase the model performance which can be influenced by error propagation in this project. Rosenberg et al. (2005) mentioned that there are still many critical issues in self-training that

need to be explored for the practical application of these training concepts. Although the different versions of training approaches did not lead to a significant improvement in the performance of salary bin prediction, there are still other useful models and techniques that can be explored in the future to improve the performance of the mode on unlabeled datasets.

References

Bhola, A., Halder, K., Prasad, A., and Kan, M.-Y. (2020). Retrieving skills from job descriptions: A language model based extreme multi-label classification framework.

Chen, B., Jiang, J., Wang, X., Wan, P., Wang, J., and Long, M. *Debiased Self-Training for Semi-Supervised Learning.*

Pise, N. N. and Kulkarni, P. (2008). A survey of semi-supervised learning methods.

Rosenberg, C., Hebert, M., and Schneiderman, H. (2005). Semi-supervised self-training of object detection models.

Stikic, M., Van Laerhoven, K., and Schiele, B. (2008). Exploring semi-supervised and active learning for activity recognition. *International Symposium on Wearable Computers.*

TAN, S. (2006). An effective refinement strategy for knn text classifier. *Expert Systems with Applications*, 30(2):290–298.

Tang, C. I., Perez-Pozuelo, I., Spathis, D., Brage, S., Wareham, N., and Mascolo, C. (2021). Selfhar: Improving human activity recognition through self-training with unlabeled data. *arXiv:2102.06073 [cs].*

Vu, T., Luong, M.-T., Le, Q., Simon, G., and Research, G. *STraTA: Self-Training with Task Augmentation for Better Few-shot Learning.*

Yang, Q., Wei, X., Wang, B., Hua, X.-S., and Zhang, L. *Interactive Self-Training with Mean Teachers for Semi-supervised Object Detection.*

Zhu, X. and Ghahramani, Z. *Learning from Labeled and Unlabeled Data with Label Propagation.*