INM427 Neural Computing Individual Project

Yumi Heo (Msc Data Science / 230003122 / [yumi.heo@city.ac.uk](mailto:yumi.heo@city.ac.uk))

**A Comparison of Multilayer Perceptrons and Support Vector Machines for Bank Churn Prediction**

**Abstract**

**Description and Motivation**

Analysing customer churn in the bank is important for maintaining profitability, customer satisfaction, and competitiveness against other banks. Therefore, banks must enhance their ability to identify potential customer churn. This can be achieved using a supervised classification model based on neural networks or support vector machines (SVM), using customer and churn data. In this paper, multilayer perceptrons (MLP) and SVM have been utilised to conduct an experiment to find which of the two models better performs in classifying customer churn. By comparing and evaluating the performance of the two models, we will learn about the characteristics of the two algorithms and determine which model is has the higher accuracy for this task and which model is appropriate for this case. Eventually, the best model will enable banks to predict future churn from customer data and proactively improve services to retain their customers.

**Initial Analysis of Dataset**

This bank churn dataset is obtained from Kaggle [1]. It is a tabular dataset which has 165,034 samples with 14 features including the target feature. The target is a column named with ‘Exited’, which is binary. ‘1’ means to close their account and ‘0’ means to remain their account. The class of customer churn is always imbalanced. In order to handle this issue, Since the number of sample is not very large, stratified k fold cross validation is planned to be used to perform model training and validation using a training set and validation set that balance the classes on both sides.

**Hypothesis Statement**

**SVM은 MLP에 비해 트레이닝에 걸리는 시간이 길다 왜냐하면 다수의 샘플을 돌리는데 적합하진 않은 모델이기 때문**

**둘다 왜 그런 피쳐들을 우선적으로 선택했는지 확인이 어려움 블랙박스 모델의 결과물 택한 것에 대한 설명이 어려움**

**Summary of the two ML methods with their pros and cons**

**MLP 장점**

**MLP는 SVM에 비해 샘플 수가 많은 경우 트레이닝 속도가 빠르다**

**MLP 단점**

**SVM 장점**

**SVM 단점**

**Pros:**

Non-linearity: MLPs can learn complex non-linear relationships in data due to their ability to model intricate decision boundaries through multiple layers and activation functions.

Feature Learning: MLPs can automatically learn relevant features from raw data, reducing the need for manual feature engineering.

Versatility: MLPs can be applied to a wide range of tasks, including classification, regression, and even unsupervised learning tasks like clustering and dimensionality reduction.

Parallel Processing: Training of MLPs can be efficiently parallelized, especially with the use of modern computational frameworks and hardware accelerators like GPUs.

Cons:

Overfitting: MLPs are prone to overfitting, especially when dealing with small datasets or when the model is too complex relative to the amount of available training data.

Hyperparameter Sensitivity: Proper tuning of hyperparameters such as learning rate, number of hidden layers, and number of neurons per layer can be challenging and time-consuming.

Computationally Intensive: Training large MLPs with many layers and neurons can be computationally intensive, requiring substantial computational resources and time.

Black Box Nature: Interpretability of MLP models can be challenging due to their complex, nonlinear nature, making it difficult to understand the reasoning behind predictions.

Support Vector Machine (SVM):

Pros:

Effective in High-Dimensional Spaces: SVMs are effective in high-dimensional spaces, making them suitable for tasks with a large number of features, such as text classification and image recognition.

Margin Maximization: SVMs aim to maximize the margin between classes, which often results in models that generalize well to unseen data and are less prone to overfitting.

Kernel Trick: SVMs can efficiently handle non-linear decision boundaries through the use of kernel functions, allowing them to capture complex relationships in the data.

Sparsity: SVMs typically use only a subset of training data points (support vectors) in the decision function, making them memory efficient, especially for large datasets.

Cons:

Limited Scalability: SVMs can become computationally expensive and memory-intensive, especially when dealing with large datasets, as the time complexity of training SVMs is typically quadratic in the number of samples.

Sensitive to Noise: SVMs can be sensitive to noisy data and outliers, which can negatively impact their performance if not properly handled or preprocessed.

Difficulty in Parameter Tuning: SVMs have several hyperparameters that need to be tuned, such as the choice of kernel and regularization parameter, which can be challenging and require careful optimization.

Limited Interpretability: Similar to MLPs, SVMs can be considered as black box models, as it's often difficult to interpret the learned decision function, especially when using complex kernel functions.

**Methodology**

•

Split the dataset into 80% for training and 20% for testing.

•

Add Gaussian distribution noise to balance the target in the training set.

•

Start with basic models and then perform hyperparameter tuning to optimize model performance.

•

Check the accuracy of models built at each step, using the test set and the confusion matrix.

•

Implement 5-fold cross-validation after hyperparameter tuning for a robust estimate of the model's performance.

•

For random forest, utilise ROC(Receiver Operating Characteristic) curves and AUC(Area Under the Curve) to find the final model.

•

For logistic regression, use cross-entropy error to find the best model.

•

Evaluate the final models of RF and LR using the confusion matrix, ROC curves, and AUC on the test set.

**방법 설명..**

**두 모델 모두 normalised 된 데이터를 사용**

**왜냐하면 두 모델 모두 데이터 스케일링이 필요한 모델이기 떄문**

**Choice of Parameters and Experimental Results**

**(e.g. cross-validation, choice of parameters and experimental results)**

**Analysis and Critical Evaluation of Results**

**두 모델 성능 비교는 confusion matrix와 ROC로 가능할 듯**

**+ 각 모델의 프레시젼 리콜 커브도**

**Lessons Learned**

**굳이 테이블형 데이터에 뉴럴네트워크가 필요하지 않다**

**Future Work**

**내장 GPU를 사용하여 컴퓨팅 속도를 빠르게 해서 결과를 보기로**

**Reference**

**[1] kaggle**

Comparison plots: lr curve & accuracy / prevision & recall curve(https://ai-com.tistory.com/entry/ML-%EB%B6%84%EB%A5%98-%EC%84%B1%EB%8A%A5-%EC%A7%80%ED%91%9C-Precision%EC%A0%95%EB%B0%80%EB%8F%84-Recall%EC%9E%AC%ED%98%84%EC%9C%A8)

Precision(정밀도)는 얼마나 정확하게 유저 이탈이라고 예측하는지에 대한 지표입니다. Recall(재현율)은 실제 이탈자에 대해서 얼마나 정확하게 이탈자라고 예측하는지에 대한 지표입니다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mlp baseline(1 hidden layer / 6 hidden units) | Test Accuracy: 0. 84.38 | Precision of the best-trained model = 74.32  % | Recall of best-trained model = 48.67  % | F1 score of best-trained model = 58.82  % |
| MLP classifier(1 hidden layer / 6 hidden units) | Best Accuracy: 0.8625098466945403 |  |  |  |
| MLP classifier(2 hidden layer / 9:9 hidden units) | Best Accuracy: 0.8649336484275586  Best Hidden Units: (9, 9) |  |  |  |
| Mlp 2(2 hidden layer / 9:9 hidden units) | Test Accuracy: 0.8606 | Precision of the best-trained model = 73.40% | Recall of best-trained model = 61.06% | F1 score of best-trained model = 66.67% |
| Mlp 2(2 hidden layer / 9:9 hidden units), lr 스케줄러 적용  scheduler = lr\_scheduler.LinearLR(optimizer, start\_factor=0.33, total\_iters=4) | Test Accuracy: 0.8630 | Precision of the best-trained model = 76.74% | Recall of best-trained model = 58.41% | F1 score of best-trained model = 66.33% |
| Mlp 2(2 hidden layer / 9:9 hidden units), lr 스케줄러 적용  scheduler = lr\_scheduler.LinearLR(optimizer, start\_factor=0.33, total\_iters=100) | Test Accuracy: 0.8641 | Precision of the best-trained model = 77.65% | Recall of best-trained model = 58.41% | F1 score of best-trained model = 66.67% |
| Mlp 2(2 hidden layer / 9:9 hidden units), lr 스케줄러 적용  scheduler = lr\_scheduler.LinearLR(optimizer, start\_factor=0.33, total\_iters=100) | Test Accuracy: 0.8616 | Precision of the best-trained model = 75.61% | Recall of best-trained model = 54.87% | F1 score of best-trained model = 63.59% |

lr어떻게 변화하는지 print 뽑기

시그모이드가 왜 output layer에 쓰이는지, tanh에 대한 설명(Activation Functions: Comparison of Trends in Practice and Research for Deep Learning)

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

https://machinelearningmastery.com/data-preparation-without-data-leakage/  
k fold cross validation batches train validation test set 설명

<https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/>

loss plot과 오버피팅 설명

pytorch로 모델링을 햇지만 gridsearch를 sklearn 패키지를 이요해서 햇으므로 정확성이 양 패키지 사이에 얼마나 있을지 모르겠음 future work로 냅둬

어차피 아담의 특징이 00이니 러닝레이트 스케줄러는 적용하지 않음

대신 그리드 서치 정도 적용

Support Vector Machine algorithms are not scale invariant, so **it is highly recommended to scale your data**. For example, scale each attribute on the input vector X to [0,1] or [-1,+1], or standardize it to have mean 0 and variance 1. Note that the same scaling must be applied to the test vector to obtain meaningful results. (<https://scikit-learn.org/stable/modules/svm.html>)

Reference

[1] W. Reade and A. Chow, ‘Binary Classification with a Bank Churn Dataset’. Kaggle, Jan. 02, 2024. Accessed: Feb. 05, 2024. [Online]. Available: <https://www.kaggle.com/competitions/playground-series-s4e1>