Feature Engineering in Predicting Central Neuropathic Pain

LIANG Zeyan, 2627888L
Xie Dongzhan, 2663048X
Yang Suiyi, 2590260Y
Zhang Lingrui, 2647013Z

December 3, 2021

1 Introduction

In CASE2, we used 180 rows of data, each with a total of 432 features, to train a classification-based machine learning model to predict Central Neuropathic Pain. Excessive number of features not only increases the training cost of the model significantly, but also causes curse of dimensionality, which reduces the accuracy of the model. Feature engineering provides a solution to these problems. Feature engineering improves the performance of a model by transforming the feature space of the original dataset, which is the central task in the data preparation phase of machine learning[5]. Our primary focus is to optimize the original data using different feature engineering strategies and to evaluate their performance.

The data was obtained from 18 different participants, with each participant repeating the test 10 times, obtaining a total of 180 rows of data. Among these 18 participants, 8 participants did not develop a CPN six months after testing and 10 participants developed a CPN six months after testing, meaning that 80 rows of data had a label of 0 and 100 rows of data had a label of 1. Each row of data is provided by 48 electrode EEGs, each electrode EEG provides 9 different features, for a total of 432 features. These data have been previously signal denoising, normalization, temporal segmentation and frequency band power estimation. Different feature engineering strategies will be applied to the above data.

To implement and evaluate the different feature engineering strategies, feature selection and feature extraction are both used in our case. For feature selection, three main feature selection methods are used in our case, including Filtering Methods, Wrapper Methods, Embedding Methods. The specific algorithms used for each method are indicated in Table 1. For feature extraction, We implement and evaluate PCA and LDA algorithms as the specific methods.

Filtering Methods	**Wrapper Methods**	**Embedding Methods**
Pearson Correlation	SFS_forward	Embedded with penalty
Chi-square	RFE	Embedded with tree model
Mutual Information	RFECV	

Table 1: The specific algorithms of the three feature selection methods

In this report we will investigate and discuss which feature engineering strategy brings the best performance for classification-based models. In addition, we will explore the optimal parameters for each feature engineering strategy. Ultimately, we will also explore the reasons why some feature engineering strategies are effective and others are ineffective.

2 Methods

Scikit learn is a Python module that integrates a variety of the most advanced machine learning algorithms[6]. Scikit learn provides a simple and efficient API for implementing various feature engineering strategies. In our case, we implement our different feature engineering strategies through importing various functions from sklearn.feature selection or sklearn.decomposition.

2.1 Classification Model Selection

We need to select and apply a machine learning model to evaluate our feature engineering strategies. Through K-Fold method for implementing cross validation, we evaluated a variety of supervised classification machine learning models trained and tested with data from the original dataset, including random forest, decision tree, logistic regression and SVM. The following table shows the average accuracy and time of each model when trained on the original dataset, using K-Fold for cross validation, with K = 10:

	Random Forest	Decision Tree	Logistic Regression	SVM
Accuracy	0.878	0.767	0.944	0.889
Time	0.48s	0.03s	0.02s	0.01s

Table 2: Average accuracy and time of each model when trained on the original dataset, using K-Fold for cross validation, with K = 10

Although the feature engineering strategy gives a greater boost to less accurate models such as the random forest classification model. However, even though their accuracy is improved, they still cannot beat the accuracy of models that are already highly accurate using the original dataset, such as logistic regression classification models. Therefore, we finally chose to use a logistic regression classification model as the model for evaluating our feature engineering strategies. By comparing the model accuracy from data processed using different feature engineering strategies with the model accuracy from the original dataset, we can evaluate the performance of a feature engineering strategy.

2.2 Feature Selection

In feature engineering strategies, feature selection as a data pre-processing method which has proven effective in the field of machine learning, feature selection provides simpler, cleaner and more understandable data for the machine learning model[4].Feature selection is mainly divided into three different algorithms, Filtering Methods, Wrapping Methods and Embedding Methods. Filtering Methods rely on the mathematical properties of the data itself, such as variance, it is an independent feature selection method[3]. Wrapping Methods uses the performance of a classifier or regression model to evaluate the quality of the currently selected features by iteration. Embedding Methods embed feature selection into the underlying model[4].

To explore which feature selection approaches perform better, we have selected some representative algorithms within each of these three different approaches. We implemented the different algorithms by calling their APIs in Scikit learn separately and tuned their specific parameters through extensive experimentation to get the best results. In addition, we have set an upper limit on the maximum number of features to be selected. We have tried to use a mixture of feature selection methods, but ultimately did not retain them in our final version. For example, we apply Filtering Methods to the raw data before applying Embedding Methods to the processed data. We found that using a mixture of methods did not help to improve the performance of feature selection. The speculative reason for this is that Filtering Methods discard mathematically invalid features, but these discarded features are actually useful for feature selection in the underlying model.

2.2.1 Filtering Methods

Pearson Correlation, Chi-square and Mutual Information are used as our Filtering Algorithms. The three different algorithms can be implemented indirectly and efficiently by calling the functions in sklearn.feature selection. These three algorithms rank the importance of each feature according to the properties of the data itself. We need to set an upper limit on the number of features selected, otherwise an excessive number of features would result in a large proportion of features becoming invalid features that would not improve the accuracy of the model. In this dataset, we set the upper limit of the number of features to 20. In addition to this we implemented a function that automatically adjusts the parameters by iteration in order to find the optimal number of features to select.

2.2.2 Wrapper Methods

SFS_forward, Recursive Feature Elimination (REF) and Recursive Feature Elimination with cross validation(REFCV) are used as our Wrapping Algorithms. The three different algorithms can be implemented indirectly and efficiently by calling the functions in sklearn.feature selection. We chose the SFS_forward algorithm instead of the SFS_backward algorithm. The reason is that SFS_backward is not suitable for our dataset and would cause failure to converge, probably due to the small size of our dataset. In addition to this we used the SVM algorithm in combination with REF and REFCV instead of using the regression model to evaluate the merit of the features. We use the accuracy of logistic regression models to evaluate the merits of feature engineering strategies. Therefore it makes more sense to use the SVM algorithm, which is also a classification model, to evaluate the features in Wrapping Methods.

2.2.3 Embedding Methods

Embedded with penalty and Embedded with tree model are used as our Embedding Algorithms. The two different algorithms can be implemented indirectly and efficiently by calling the functions in sklearn.feature selection. We have used not only Embedded with penalty but also Embedded with tree model, which allows for faster feature selection.

2.3 Feature Extraction

Feature extraction is another approach in feature engineering strategies. The main purpose of feature extraction is to obtain the most relevant information from the original data and represent this information in a low-dimensional space, which is a special form of dimensionality reduction[2]. Through feature extraction, the original data vector, which contains many features but does not actually contain much information, is transformed into a simplified feature vector, thus removing the negative impact of useless features on the model.

2.3.1 PCA and LDA

To implement feature extraction, we used two different algorithms, PCA and LDA. Principal component analysis (PCA) and Linear discriminant analysis (LDA) are two popular algorithms for feature extraction. PCA is an unsupervised feature extraction algorithm that does not require the Label of the data to complete the algorithm[7]. LDA is a supervised feature extraction algorithm, which must require the Label of the data ito complete the algorithm. In PCA the shape and position of the original dataset changes as it is transformed into a different space, LDA does not change position, it only tries to provide more ways of combining features and drawing decision regions between given features[1]. These two algorithms can be implemented indirectly and efficiently by calling the functions in sklearn.decomposition. We optimise the performance of PCA and LDA algorithms by manually tuning the function parameters.

3 Results

3.1 Result of Feature Selection

3.1.1 Result of Filtering Methods

In our CASE, we implement and evaluate three different specific algorithms for Filtering Methods, which are Pearson Correlation, Chi-square and Mutual Information. Not only do we need to optimise each algorithm to achieve the best performance for each algorithm within the maximum number of features, we also need to compare the best performance of the three algorithms to select the optimal algorithm for Filtering Methods.

The following figure shows the Average scores and accuracy of Logistic regression model in different number of feature selection for each algorithm (Use K-Fold for cross validation, with K = 10). The figure points out that for Pearson Correlation and Chi-square algorithms, they only need to select a small number of features to reach the convergence state, and even after increasing the number of features selected, the performance of the model does not show a significant improvement, and the performance fluctuates up and down in a reasonable interval. For Mutual Information, when the number of features is less than 15, increasing the number of features selected will significantly improve the performance; after the number of features selected is greater than 15, increasing the number of features selected does not show a significant improvement in the performance of the model. In addition, we can conclude in the figure that the best number of features to select is 14 for Pearson Correlation, 16 for Chi-square, and 15 for Mutual Information.

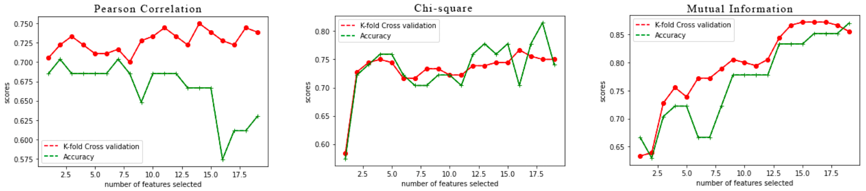

Figure 1: Average scores and accuracy of Logistic regression model in different number of feature selection by Pearson Correlation, Chi-square and Mutual Information (Use K-Fold for cross validation, with K = 10)

The following table shows the number of feature selections, average K-Fold CV scores with K=10, and average accuracy of each Filtering Methods algorithm when they perform optimally. The table shows that Mutual Information is ahead of the other two algorithms in terms of both average accuracy and average K-Fold CV scores with K=10. Therefore, we can conclude that Mutual Information has the best performance for Filtering Methods, with an optimal feature selection of 15.

	Pearson Correlation	Chi-square	Mutual Information
Average K-Fold CV scores with K=10	0.750	0.767	0.872
Average Accuracy	0.666	0.704	0.833
Best number of selected features	14	16	15

Table 3: Each Filtering Methods algorithm in terms of the number of features selected, average K-Fold CV scores, and average accuracy when they perform optimally.

3.1.2 Result of Wrapping Methods

In our CASE, we implement and evaluate three different specific algorithms for Wrapping Methods, which are SFS_forward, Recursive Feature Elimination (REF) and Recursive Feature Elimination with cross validation(REFCV). For SFS_forward and Recursive Feature Elimination (REF), we need to find their optimal number of feature choices in an iterative way to get the best performance. However, for REFCV we do not need to choose the optimum number of features, the REFCV algorithm will return an optimum number of features directly. Besides, we also need to compare the best performance of the three algorithms to select the optimal algorithm for Wrapping Methods.

The following figure shows the Average scores and accuracy of Logistic regression model in different number of feature selection for each algorithm (Use K-Fold for cross validation, with K = 10). The figure indicates that for the SFS_forward algorithm, the algorithm is close to convergence at around 15 feature selections and the performance of the algorithm stops improving. For the REF algorithm, within the limit of the maximum number of feature selections, the performance of the algorithm improves as the number of feature selections increases. Furthermore, we can conclude that the best performance is obtained for the SFS_forward algorithm when the number of feature selections is 16, and for the REF algorithm when the number of feature selections is 21.

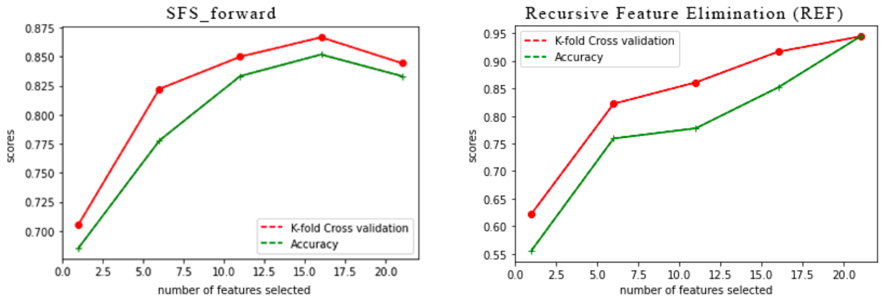

Figure 2: Average scores and accuracy of Logistic regression model in different number of feature selection by SFS_forward and Recursive Feature Elimination (REF) (Use K-Fold for cross validation, with K = 10)

The table indicates that REFCV has a significant lead over the other two algorithms in terms of both average accuracy and average K-Fold CV scores with K=10. Therefore, we can conclude that for Wrapping Methods, REFCV has the best performance with an optimal feature selection number of 22.

	SFS_forward	REF	REFCV
Average K-Fold CV scores with K=10	0.867	0.944	0.956
Accuracy	0.852	0.944	0.944
Best number of selected features	16	21	22

Table 4: Each Wrapping Methods algorithm in terms of the number of features selected, average K-Fold CV scores, and average accuracy when they perform optimally.

3.1.3 Result of Embedding Methods

In our CASE, two different Embedding methods algorithms are implemented, Embedded with penalty and Embedded with tree model, and we need to explore which of these two embedded models is more suitable for our CASE.

The following table shows the number of feature selections, average K-Fold CV scores with K=10, and average accuracy of each Embedding Methods algorithm when they perform optimally. We can conclude that Embedded with penalty is more suitable than Embedded with tree model for our CASE. However, the number of features selected by Embedded with penalty is too high when compared with Wrapping Methods and Filtering Methods.

	Embedded with penalty	Embedded with tree model
Average K-Fold CV scores with K=10	0.939	0.850
Accuracy	0.926	0.870
Best number of selected features	180	180

Table 5: Each Embedding Methods algorithm in terms of the number of features selected, average K-Fold CV scores, and average accuracy when they perform optimally.

3.1.4 Result of Best Feature Selection Strategy

By combining the evaluation results of the eight different feature selection algorithms mentioned above, we obtained a feature selection strategy that is the best for the current CASE, i.e. REFCV algorithm. We will select the 22 most representative features from the 432 original features by the REFCV algorithm. The correlation matrix of the optimal feature selection results is shown below.

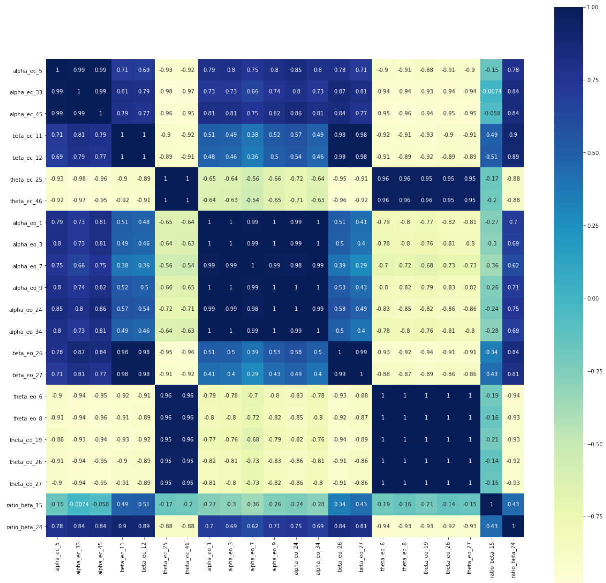

Figure 3: Correlation Matrix of Best Selected Features

The correlation matrix of these 22 features was drawn to assess the correctness of the feature selection. The closer to blue, the stronger the positive correlation between the features and the closer to yellow, the stronger the negative correlation between the features. The correlation matrix shows that most of the features selected by the REFCV algorithm have a strong correlation with each other. This indicates that our feature selection strategy eliminates features that are weakly correlated with other features, which are of minimal use for machine learning. The features that are retained carry more weight to the machine learning model. Therefore we conclude that our use of the features selected by REFCV is reasonable.

3.2 Result of Feature Extraction

In our CASE, two different Feature Extraction algorithms are implemented, including Principal component analysis (PCA) and Linear discriminant analysis (LDA). The table below shows the number of feature selections, average K-Fold CV scores with K=10, and average accuracy of the PCA and LDA algorithms when they perform optimally. The table shows that the LDA achieves 100% accuracy, which is not seen in the other algorithms. In contrast, the PCA algorithm performed worse. The reason for this may be due to the fact that the LDA algorithm is supervised algorithm and the PCA algorithm is unsupervised algorithm. In contrast to the PCA algorithm, LDA uses the Label dataset to improve the accuracy of the algorithm. Therefore we conclude that for the feature extraction strategy, the LDA algorithm is significantly better than the

PCA algorithm in our CASE.

	PCA	LDA
Average K-Fold CV scores with K=10	0.756	1.000
Accuracy	0.778	1.000
Best number of selected features	180	180

Table 6: PCA and LDA in terms of the number of features selected, average K-Fold CV scores, and average accuracy when they perform optimally.

4 Discussion

4.1 Valuable Questions in Case Study

In the process of our experiments, we found that not all feature engineering strategies led to an increase in the accuracy of the models, but instead some of them led to a decrease in accuracy. However, these feature engineering strategies still have a positive effect. Although the accuracy of the models decreased, the training cost decreased significantly. This is not apparent in our smaller datasets, but for large datasets of several million rows, such feature engineering is necessary. We need to balance the accuracy of the model with the cost of training. In addition , for Filtering Methods, some algorithms will get better results with very high feature selection numbers, but this does not mean that this is a reasonable feature selection strategy. Since Filtering Methods rank the importance of features according to the properties of the data itself. If a large number of features is selected, features that are not strongly correlated will be selected. These features are not helpful for the training of the model.

4.2 Limitation

For some of the non-reference methods, such as RFECV and embedding methods, we only provide a correlation matrix, which only shows static results. In the future we need to use some additional figures to show the dynamics of the parameters, accuracy etc. during the experiment. This would help to provide a better experience for the researcher. Besides, we only used logistic regression as our evaluation model. In the future we can try to use other models as our evaluation model, such as SVM, Random Forest, etc. It will lead to more enlightening conclusions.

4.3 Conclusion

The feature engineering strategies are flexible and diverse. Model selection, Feature Selection Strategies selection, Feature Extraction Strategies selection and the selection of validation methods all require extensive experimentation and comparison to make the final decision. Although feature engineering is only a small part of machine learning, it determines the upper limit of accuracy of machine learning. In our CASE, for the feature selection strategy, the best performing algorithm is REFCV; for the feature extraction strategy, the best performing algorithm is LDA. From our final experimental data, we can demonstrate that feature engineering strategies not only removes unnecessary feature data and reduces computation time, but also significantly increases model accuracy.

References

[1] Balakrishnama, S., and Ganapathiraju, A. Linear discriminant analysis-a brief tutorial. *Institute for Signal and information Processing 18*, 1998 (1998), 1–8.

[2] Kumar, G., and Bhatia, P. K. A detailed review of feature extraction in image processing systems. In *2014 Fourth international conference on advanced computing & communication technologies* (2014), IEEE, pp. 5–12.

[3] Kumar, V., and Minz, S. Feature selection: a literature review. *SmartCR 4*, 3 (2014), 211–229.

[4] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. Feature selection: A data perspective. *ACM Computing Surveys (CSUR) 50*, 6 (2017), 1–45.

[5] Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E. B., and Turaga, D. S. Learning feature engineering for classification. In *Ijcai* (2017), pp. 2529–2535.

[6] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research 12* (2011), 2825–2830.

[7] Ringnér, M. What is principal component analysis? *Nature biotechnology 26*, 3 (2008), 303–304.