Learning From Data 2022 Spring Project

1st Halil Faruk Karagöz
Computer Engineering
Istanbul Technical University
Istanbul, Turkey
karagozh18@itu.edu.tr

2nd Batuhan Can
Computer Engineering
Istanbul Technical University
Istanbul, Turkey
canb18@itu.edu.tr

3rd Hakan Tuğrul Otal
Computer Engineering
Istanbul Technical University
Istanbul, Turkey
otal18@itu.edu.tr

4th Göksu Eldemir
Computer Engineering
Istanbul Technical University
Istanbul, Turkey
eldemir18@itu.edu.tr

5th İsmail Çetin
Computer Engineering
Istanbul Technical University
Istanbul, Turkey
cetini18@itu.edu.tr

***Abstract*—Increasing the resolution of data can be useful when it comes to analyse it. Data frequency is crucial in terms of diagnosis of brain diseases. Therefore, it is necessary to have high resolution data while modelling a brain network. There are lots of features to consider, in a brain network. Thus, training a model might require high computational power. In this project, we proposed a machine learning model in which a low resolution data is taken as an input, then number of features are reduced by using several methods and Principal Component Analysis (PCA). Finally, high resolution data of the given brain network is given as an output.**

***Index Terms*—Artificial Intelligence, Machine Learning, SciKit-Learn, MultiTaskElasticNet, Correlation, Principal Component Analysis (PCA), Variance Thresholding, Leaky Clamping**

I. Introduction

In this project, our goal is to train low resolution brain connectivity matrix and estimate high resolution brain connectivity matrix by using it. To that end, we used low-resolution brain connectivity data as our train set and we used the given high resolution data as a ground truth to find a connection between the regions of the brain. In the given low-resolution train data there are more than twelve thousands features, 12720 to be more precise. To reduce the number features, firstly we used several feature selection methods. Afterwards, we decided to use Principal Component Analysis (PCA) in order to increase the performance of the model. Using 5-Fold cross validation and Linear Regression Model we tested our model and uploaded the results to Kaggle.

- Team Name: 1501800(14-33-48-65-68)
- Final Score: 0.02307
- Rank: 7

II. Datasets

The low-resolution brain connectivity dataset includes 189 samples and 12720 features. Due to a vast number of features, it takes a long time to train the model besides, the model becomes overfitted. As a solution, two different dimensionality reduction strategies were implemented. The first method is the variance threshold. The variance threshold removes all features that variance does not meet the threshold value. In this way, features that both impair the accuracy of the model and extend its training time were discarded.

The second method is highly correlated feature elimination. In this method coefficient matrix of the training data set has been generated. This matrix determines the similarity of the features; features with larger correlation coefficients have more similarity. With the given threshold indexes of highly correlated features are located then one of the feature that have high correlation is deleted.

Having used two feature selection methods, we reduced the 12720 features to approximately 11000.

III. Methods

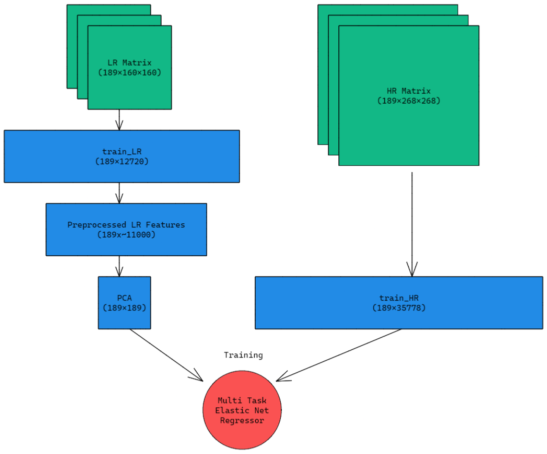

Fig. 1. Training the Model

Our proposed pipeline consists of multiple different stages as it can be seen at Figure 1. Firstly, "train_lr" input data gets preprocessed with some feature selection and extraction methods. Our core method in preprocessing phase is PCA and more methods are used to get a better performance on PCA. The first step of preprocessing is Variance Thresholding.

Variances of all 12720 features get calculated and some features get selected according to a threshold value. During training of the model, low variant features have no effect. Thus, these features are eliminated before PCA. Then, highly correlated features are determined and some of them has been removed from the dataset. After that, remaining features are reduced to 189 by using the Principal Component Analysis (PCA) method [1].

Subsequently, the pipeline starts to train the model. Our MultiTaskElasticNet [2] regression model trains with previous extracted features (189x189) and “train_hr” (189x35778). Thanks to Multi Task Learning instead of using individual features for each output, with just one model more generalized prediction can be found. We tried both Multitask Lasso and Elastic Net models but nevertheless Multi Task Elastic Net proved to be better in 5K fold. When the training is complete, the model can be used to make predictions from low resolution connectivity matrix to high resolution connectivity matrix.

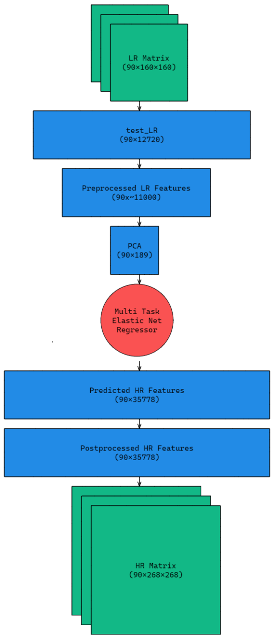

Fig. 2. Predicting with the Model

Lastly as it can be seen at Figure 5, we create a prediction with given “test_lr” data with our trained model. When data is examined, we observe that the value of features must be between the interval 0-1. The pipeline post-process the output by applying “clamping” which assures that outputs are between 0 and 1 in a leaky way. With this last step, predicted high resolution connectivity matrix is found and ready to be saved as an output file.

We trained our models using 5-Fold and evaluated them at each step. We tuned all the parameters in our model by trying the most promising combinations of parameters. By comparing their mean squared errors and observing the parameters at every change, we fine tuned the input parameters and found the best combinations to use.

IV. Results and Conclusions

As an evaluation metric Mean Absolute Distance(MAD) error and Mean Squared Error are selected to monitor how the model performs. MAD can be thought of as MSE in 1 dimension except the denominator here is n^2. MSE is used because it is the most common used evaluation criteria for Machine Learning and Deep Learning models. The following are the results of the metrics.

	1	**2**	**3**	**4**	**5**	**AVG**
Train MSE	0.00423	0.00426	0.00419	0.00423	0.00430	0.00424
Test MSE	0.02311	0.02548	0.02260	0.02372	0.02367	0.02371
Test MAD	0.26576	0.29779	0.26152	0.27846	0.28829	0.27836

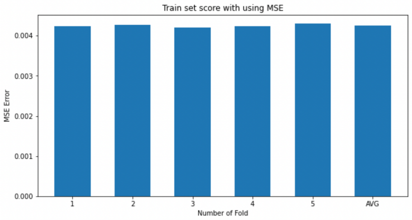

Fig. 3. Train Set Mean Squared Error(MSE)

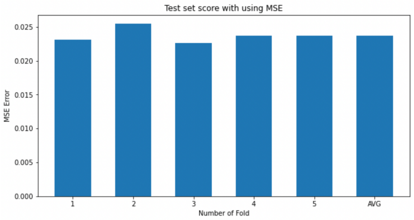

Fig. 4. Test Set Mean Squared Error(MSE)

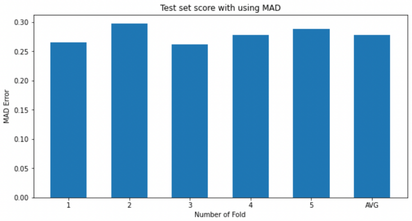

Fig. 5. Test Set Mean Absolute Distance(MAD)

References

[1] “sklearn.decomposition.PCA” scikit-learn. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html [Accessed 8 June 2022].

[2] “sklearn.linear_model.MultiTaskElasticNet” scikit-learn. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.MultiTaskElasticNet.html [Accessed 8 June 2022].