Prediction Of Brain Connectivity Values

O. Kürşat Karayılan[#1], Muhammed Said Dikici[*2], Selim Enes Kılıçaslan[#3]

[#]*Faculty of Computer and Informatics Engineering, İstanbul Technical University*
İstanbul Teknik Üniversitesi Ayazağa Kampüsü Bilgisayar ve Bilişim Fakültesi Maslak, İstanbul, 34469, Turkey

***Abstract*— In this project we tried to predict the brain connectivity for the next time point. This report contains results of our implementation and details about our approach.**

***Keywords*— machine learning, kaggle, prediction**

I. Introduction

In this project we are asked to predict brain connectivity features by using different type of machine learning techniques. We are given two datasets which are train_t0 and train_t1. Brain connectivity features measured at two different timepoints t0 and t1, spaced out by 6 months. Our goal is to predict what could be the values after six months given an ach brain dataset. Our team name is 150140011_150140007_150150026 at kaggle and we have managed to get 0.00208 score.

II. Datasets

Since datasets we are provided were pretty neat there was not much to do in terms of cleaning. We deleted ID columns in order to not include them into computations. Other than that we found some duplicated and empty columns and deleted them. We used pandas library when working with the data. We put datasets into pandas dataframes to work with them.

III. Methods

We used matplotlib library in order to visualise the data. Numpy and sklearn libraries helped us at computations and machine learning algorithms. At first, we simply tried linear regression. In order to have an idea about relationship between t0 and t1 values, we plot the values into scatter plot. You can see an example below.

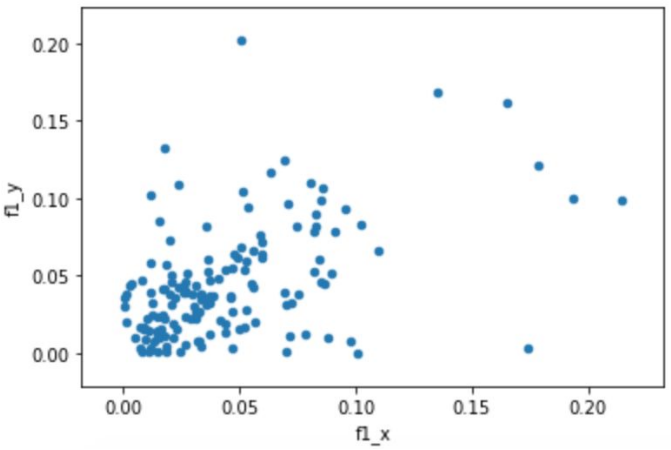

Fig. 1 An example scallet plot f1 values from t0 on x axis and f1 values from t1 on y axis.

But, in this approach we did not take into consideration features' effect on each other. We thought we can do more and looked for ways to group people. By so, we may get more accurate results by procession similar people together. Then we applied Principal Component Analysis (PCA) in order to reduce dimensions. We tried different number of component amount. We found that optimal component quantity is 11 in our model. After that we tried various machine learning algorithms such as Multiple Linear Regression, Polynomial Regression, Random Forest, Lasso Regression. But we got best result with Support Vector Regression. By using sklearn's GridSearchCV we tried to find best parameters for SVR algorithm.

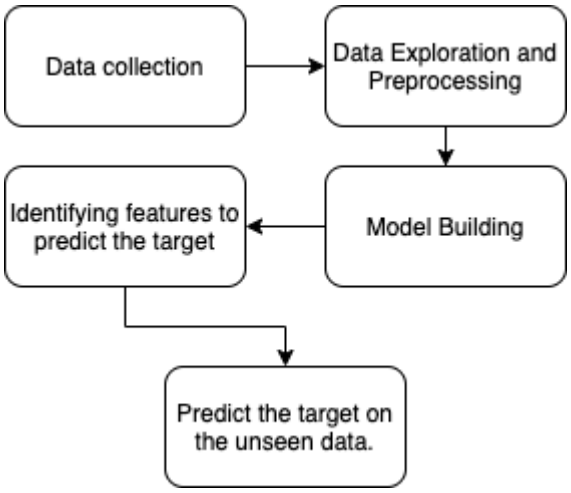

Fig. 2 Learning pipeline

IV. Conclusions

We get 0.0040 MSE from 5-fold cross-validation. Also we got 0.00208 score at the kaggle. That score made us 10th in the kaggle leaderboard.

References

[1] scikit-learn.org