Report of Machine Learning Final Project

109550027 紀竺均

GitHub\_link: https://github.com/chuchunchi/ML-Final-Project

Model\_link: https:/drive.google.com/file/d/1jgrwX8ZtBEcTeN7if1GT\_Voe\_o7DdTvG/view?usp=share\_link

1. Brief Introduction

First, I use some sklearn package to preprocess the data. Then, I use a 1D convolutional neural network implemented by PyTorch for this binary classification task.

1. Methodology
2. Data pre-process
3. Drop ‘id’ column.
4. Use OrdinalEncoder() to turn the attributes (discrete features) into ordinal integers.
5. Use SimpleImputer() to turn the missing values into ‘median’ of that feature. I have tried ‘mean’, ‘most\_frequent’ as well but ‘median’ give the highest score.
6. Use sklearn.decomposition.PCA() with n\_components=2 to decompose the data from 24 features to 2 features.
7. Use MinMaxScaler() to scale the features. I have tried ‘RobustScaler’ and ‘StandardScaler’ as well and ‘MinMaxScaler’ give the highest score.
8. Finally, use sklearn.train\_test\_split() to split train and validation data with val ratio=0.1 for training process.
9. Model architecture

A CNN model implemented by pytorch, with a total of 2 1D\_convolution layers, 3 linear layers with relu activate function, and an linear output layer with sigmoid function.

input: a tensor with shape (batch\*2 features)

output: a tensor with shape (batch\*1), dtype=float between 0, 1

* Train the model for 10 epochs and choose the one with lowest validation loss to be my model.

1. Hyperparameters
2. loss function: nn.BCELoss()
3. optimizer: torch.optim.Adam
4. learning rate = 5e-4

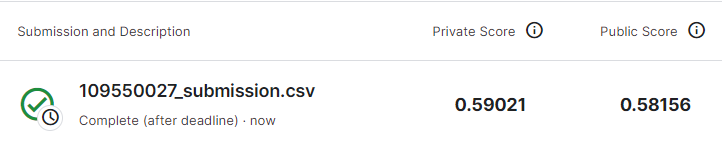
1. Comparison & Result
2. I think the key that I can beat baseline is the “PCA” decomposition method. Before I add it to my preprocess function, the accuracy can’t over 0.585.

Compare between decompose or not, 3 scaler function:

|  |  |  |  |
| --- | --- | --- | --- |
|  | RobustScaler | StandardScaler | MinMaxScaler |
| PCA(n=2) | 0.587 | 0.589 | 0.590 |
| Without PCA | 0.584 | 0.546 | 0.563 |

* I guess there should be some insignificant features that affect the result. By decomposing and scaling the features using sklearn package we reduce the noise made by them.

1. Result:



1. PEP8

I use pycodestyle\_magic to check if I write code under PEP8 guideline in iPython notebook.

The pycodestyle magic command:

%load\_ext pycodestyle\_magic

%pycodestyle\_on

1. Summary

I construct a convolution neural network to solve this real-world classification problem with score 0.59021. To achieve this, I have preprocessed and decomposed the data into two features. Then, I fed the features into a CNN model which is implemented using PyTorch. Finally I passed the output into a sigmoid function and thus get an likelihood value between 0, 1.