21 - Finding Drug Treatments Faster Through Machine Learning (Kaggle Competition - MoA)

Ho Jie Feng, Sanraj Mitra, Ong Bing Jue, Quek Hian Khun Joshua, Ning Shengying, E0014740 Lee
{E0406765,E0407056,E0309174,E0335659,E0309568,E0014740}@u.nus.edu

Introduction

The power to find drug treatments **quickly** is imperative, as the consequences of the COVID crisis have made amply clear. Currently, drug discovery is driven by targeted modelling based on understanding the underlying biological mechanisms of diseases. Scientists seek to identify a protein target associated with the disease and develop a molecule that can influence it. A drug molecule's influence is determined by its cellular effects—its Mechanism of Action(MoA)—which must be identified beforehand. Classifying the MoA of drugs **accurately** and **quickly** would allow **leaps** in **efficiency**–in terms of **theorisation** and **time**–during drug development. In this project, we are given data on how human cells react after being treated with a given drug, and attempt to build a **multi-label classifier** that identifies a drug's MoA.

Proposed Model

We implemented a Neural Network (NN) model using a single hidden layer NN, along with a PyTorch implementation of **TabNet** [1] which is an attention-based deep NN for representation learning from tabular data. In particular, we used `TabNetRegressor` as this is a **multi-label classification problem**. We have chosen to use NNs to optimize for log loss, the metric used in this competition. We also drew inspiration from a few Kaggle notebooks such as Pytorch NN+PCA+RankGauss and Pytorch | MoA.

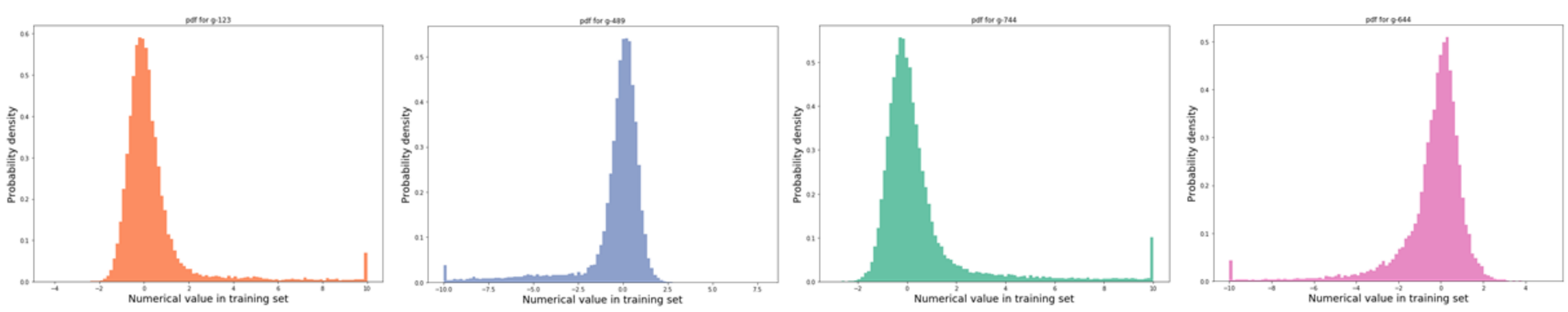

Figure 1: Exploratory Data Analysis

Exploratory Data Analysis (EDA)

- **Research Question 1:** Are we able to find insightful patterns amongst features and target labels that can help improve the model?
- Due to the abundance of features (>800 to be exact), we relied on our exploratory data analysis (EDA) to yield insights into our dataset, which we then exploited to improve our model's performance. For instance, we discovered that certain features, such as the **gene expression data** in Figure 1, have **skewed distributions**. This guided us in our feature engineering in the next stage.

Preprocessing and Feature Engineering

- **Research Question 2:** Can we pre-process the features to make our model learn better?
- Given multivariate normal distribution of the data and that the dataset had large number of features, we ran **Principal Component Analysis (PCA)** on feature columns. Information was compressed into 100 components, minimizing the "Curse of Dimensionality" effect on training efficiency of our model.
- On top of that, we ran `sklearn.preprocessing.QuantileTransformer` to transform the distribution of each feature to approximate a normal distribution and correct/standardize the distribution of feature values (which were offset by their individual variance, skewness, discovered through EDA).

Training and Validation

- We trained our TabNet model using a SOTA `Lookahead` optimizer [2] wrapping an `Adam` inner optimizer to achieve higher learning stability and faster convergence while minimizing the need for extensive hyperparameter tuning as compared to if we had used `SGD`/`Adam` optimizer alone.
- **Label smoothing** [3]—a regularization technique—was applied to prevent our (multi-label classification) model from predicting labels too confidently during training which would've resulted in poor ability to generalize.
- We used **iterative stratification** [4] (`MultilabelStratifiedKFold`) to stratify our multi-label data across splits for model cross validation to ensure a good distribution of target variables in each fold.
- Last but not least, we **ensembled models together**, mainly a simple NN with a single hidden layer with TabNet. Different architecture NNs are likely to have uncorrelated variances, and an ensemble using their weighted averages can reduce the variance. Training additional models trained on different splits also will present different variance, which can be ensembled to reduce variance even further.

Results and Evaluation

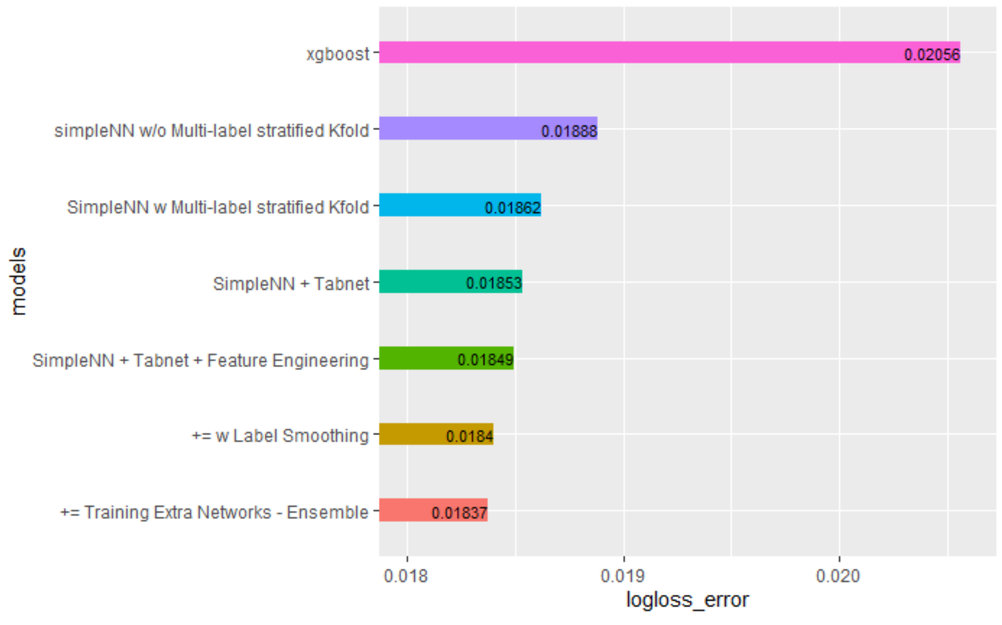

Figure 2: Different model logloss error

- **Main Goal:** After conducting EDA, Feature Engineering, can we fine tune a model that consistently gets better target scores on Kaggle?
- The metric used in this competition is the **mean columnwise logloss**. It measures how close to the actual prediction our model from the ground truth is. Using the techniques, we got a **score of 0.01837**, a **silver** on Kaggle. (38^{th} place as of 22^{nd} October)

Conclusions

- We have obtained a very satisfactory score for this competition (**silver** on Kaggle) and have met our main goal of this project.
- NNs perform better than Gradient Boosted trees, likely because of very high dimensionality and the scoring metric used.
- Preventing overfitting is the key as most targets are 0 and some targets are easier to predict than others.
- Ensemble is a powerful tool to achieve better metrics.
- We believe our model will be useful in quickly identifying the MoAs of given drugs, but we can further improve its performance by conducting more EDA and feature engineering.

References

[1] Sercan O. Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning, 2020.

[2] Michael R. Zhang, James Lucas, Geoffrey Hinton, and Jimmy Ba. Lookahead optimizer: k steps forward, 1 step back, 2019.

[3] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help?, 2020.

[4] Konstantinos Sechidis, Grigorios Tsoumakas, and Ioannis Vlahavas. On the stratification of multi-label data. In Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba, and Michalis Vazirgiannis, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 145–158, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.