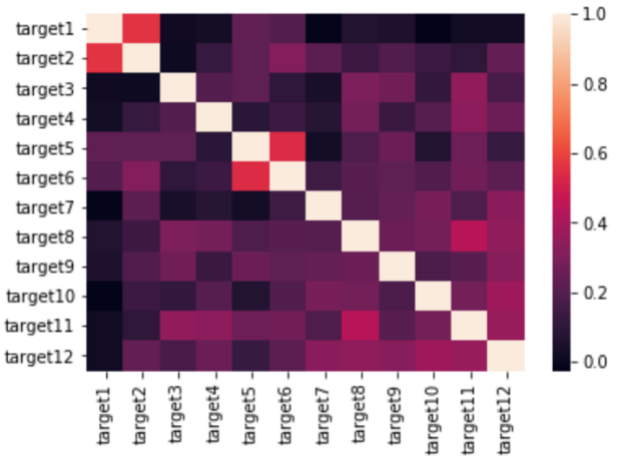
Invivo Technical Exam

Tashlin Reddy

Data Exploration

The data given contains SMILES representation of molecules and 12 binary target values to be predicted based on the molecule.
Given data features:

8014 rows Total	Missing values	Percent Equal 0
Target1	575	97.5
Target2	1112	96.4
Target3	1323	86.7
Target4	2074	94.5
Target5	1698	85.5
Target6	902	94.7
Target7	782	97.0
Target8	2079	80.7
Target9	782	96.2
Target10	1420	93.9
Target11	2094	81.2
Target12	1105	93.3



As we can see from the above chart, some targets have up to 1/4 of the total rows missing. We will most likely handle them by not including those rows in our model.
The other obvious issue is that the dataset is extremely unbalanced. Let's take "Target1" as an example, if our model predicts all zeros, we would have a 97.5% accurate algorithm. This sounds good on the surface, yet would hold no real predictive power. We must consider this unbalanced set when feeding in the data, as well as when scoring our model.
As for relationship between targets, above shown is the Pearson Correlation, Spearman was also considered, but not shown. Target1 and Target2 show some correlation, and so does 5/6. Other than that, the relationships seem to be quite weak, more in depth analysis needs to be if there are relationships to find.

Data Augmentation/Feature Engineering

The SMILES representation, which is only a string, says a lot about a molecule. Therefore, we have to engineer some features. My expertise does not lie in molecular biology and chemistry, but after a bit of research, I came across RDkit. A python library for chemistry. Learning to use this library took a bit of work but it enabled me to extract 19 descriptors such as Exact Molecular Weight and Number of Hydrogen Donors. I was also able to count 84 different

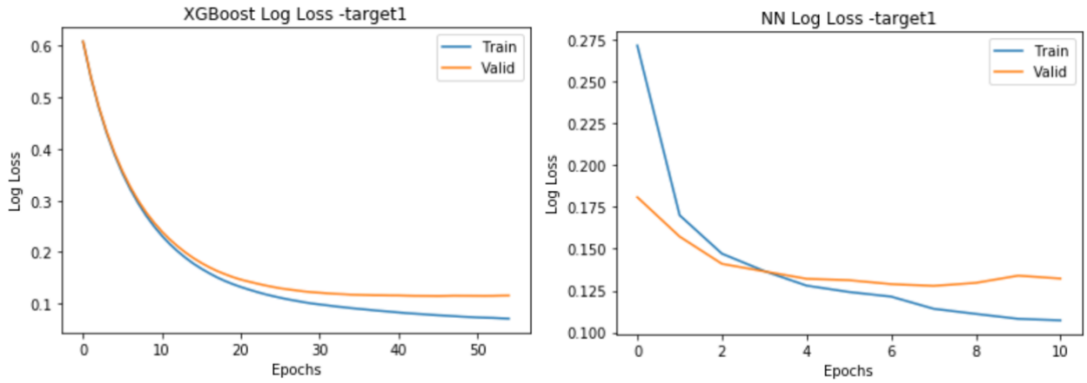
functional groups within each molecule. This created a grand total of 103 features for prediction.
There is definitely a lot of room for improvement here. I tried to extract as much useful information I could, but was ultimately quite limited.

Data Preparation

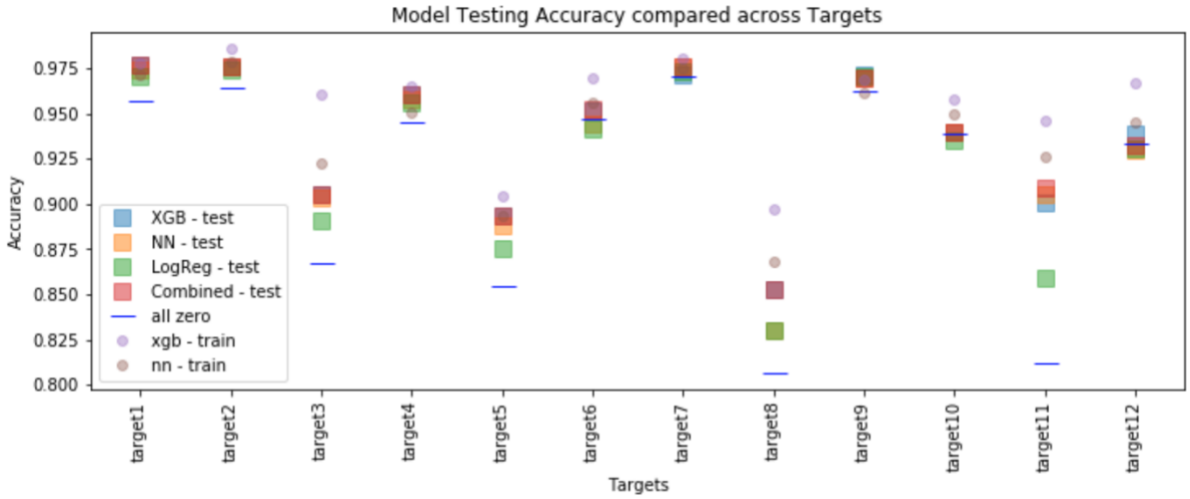
First, I merged the target column with my feature columns, and then I dropped any of the missing values. The number changed depending on the target.
From here I took two approaches. The first was to normalize and split the data into 3 parts (train, validation, test), then feed it directly.
The second was to undersample. This is a technique to help train models of unbalanced sets. I specifically chose all rows equal to 0, and then randomly select the same number of rows from the rest of the data. Thus, creating a 50/50 split. This has drawbacks of losing information from the data left behind, but helps our model weight which features are important for prediction.

Data Modelling

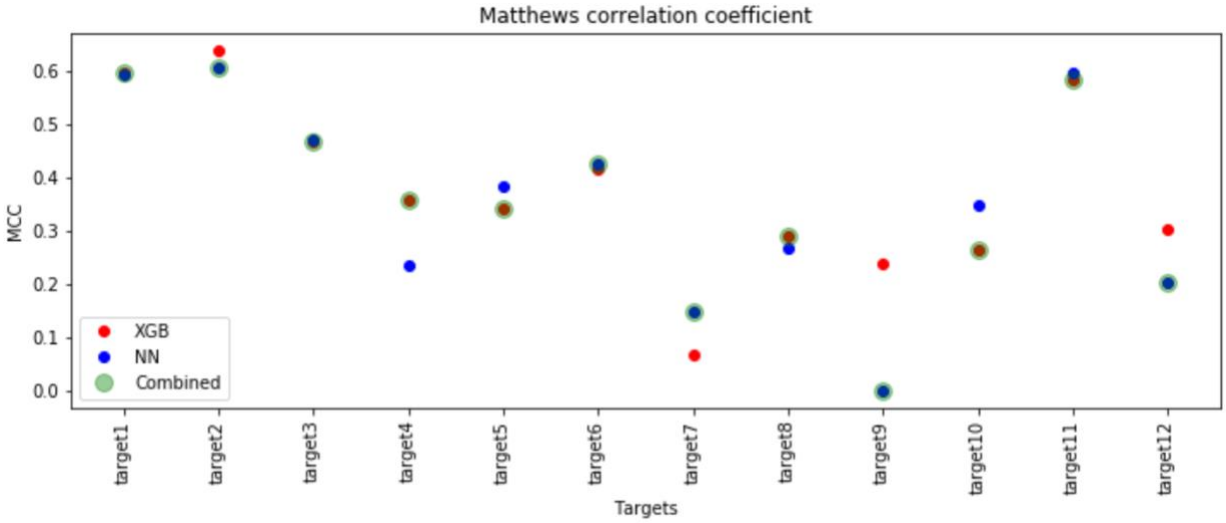
Since we are ultimately trying to predict binary values, logistic regression, is the first model that comes to mind.
I used a total of 3 models. Sci-Kit learns' Logistic Regression model as a baseline. As well as a neural network, and a boosted tree algorithm. The first being programmed with Keras, and the second with XGBoost. After looking at some of the confusion matrices from the predictions of the XGB and Keras model, sometimes one would predict too many False Positives while the other would predict too many True Negatives, I decided it might be worthwhile to combine these two models.



Above are learning curves on Target1 for both XGB and NN models as an example. The NN model was earlystopping quickly, after 5-10 epochs. I had to reduce the complexity/parameters of the neural network to make it simpler to try to avoid the issue, but with such little data, along with it being so unbalanced XGB performed slightly better.



Above is a graph showing all model accuracies across all targets. It includes training accuracies, testing accuracies as well as a baseline i.e. when the model only predicts 0. It shows that XGB is generally outperforming NN as well as either on par or better than our combined algorithm.



This graph shows the Matthews correlation coefficient. Which considers true and false positives and negatives. It is a good measure when the data set is unbalanced. +1 means a perfect prediction, 0 a random prediction, and -1 an inverse prediction.

We can clearly see that targets 1/2/11, are being predicted relatively well. Whereas targets 7/9 have something left to be desired.

Moving Forward

I think there are many areas for improvement:
Feature Engineering:

- The most important in my opinion
- I believe there was a lot of information hidden in the SMILES representation that I was not able to extract
- Lead to better prediction

Data Preparation:

- With such an unbalanced set, techniques such as undersampling were tried, but to no avail.
- Try other methods: ie resampling, kfolds

Data Modeling:

- Tune parameters and hyperparameter better
    - Either by gridsearch or genetic algorithms
- When combining two models I could have predicted a probability of 0/1 then used another binary classifier/ logistic regression to make the final prediction versus a simple if/else statement, which in the end did not outperform sole XGB.

Conclusion

I think my efforts, next time, would be to focus mainly on feature engineering useful feature and optimizing XGB for prediction. In this case, a neural network was unrealistic given the small amount of data, whereas XGB is proven to perform well with unbalanced data sets.

Running the Code:
Hopefully should be straightforward through the Jupyter Notebook.

References:

https://machinelearningmastery.com/binary-classification-tutorial-with-the-keras-deep-learning-library/
http://rdkit.org/docs/source/rdkit.Chem.html
https://www.rdkit.org/docs/source/rdkit.Chem.Fragments.html
https://seaborn.pydata.org/examples/index.html
https://stackoverflow.com/
https://scikit-learn.org/stable/
https://www.tensorflow.org/guide
https://xgboost.readthedocs.io/en/latest/build.html