Mechanism of Action

Capstone Project

Problem Statement

An initial step in a drug discovery program for a pharmaceutical company is defining a candidate's mechanism of action (MoA). This process involves testing within an in vitro setting and observing patterns of interactions between the newly formulated compound and human cells and genes. By identifying trends and measuring levels of a compound's inhibition or stimulating effects, a MoA can be determined. In short, it clarifies the 'what' 'where' and 'how' as to why a given chemical compound is most effective at its prescribed strength.

As the pharmaceutical industry matures, specialized medicine is becoming a more popular course of drug development for treating some rare genetic diseases, and determining a reliably measurable MoA is a vital step in that process. Further, by validating a compound's MoA, its financial value becomes clearer and a more predictive clinical outcome becomes achievable. This in turn creates more certainty to the drug's effectiveness, and creates a comprehensive and robust argument for the drug's potential approval to treat certain conditions.

So with the total cost of developing a marketable medication safely in the millions of dollars, mitigating risk and uncertainty of the drug's safety, activity and effectiveness is paramount to generating interest from investors as well as potentially shifting the paradigm with how certain medical conditions are treated.

Data Wrangling

The data being used was supplied through Kaggle's competition: **Mechanisms of Action (MoA) Prediction** and presented in joint by Connectivity Map, the Laboratory for Innovation Science at Harvard (LISH), and the NIH COmmon Funds Library of Integrated Network Based Cellular Signatures (LINCS), and possesses five total datasets.

The 5 datasets are: 1) a sample submission csv file which was provided to visualize what a correct submission file should look like upon completion and submission of predictions. 2) A 'test features' dataset consisting of 3982 observations and 876 features. This file was provided to be explicitly used for predicting the probabilities of each observation. 3) A 'train features' dataset consisting of 23814 observations and 876 features. This file was provided to be explicitly used for training a model. 4) A 'train targets non-scored' dataset consisting of

23814 observations and 403 features. This file was provided as an auxiliary training dataset, but possessed observations with no scoring (label) association. 5) A ‘train targets scored’ file consisting of 23814 observations and 207 features. This file contained the true binary scores for each observation in the ‘train features’ dataset, and was to be used for the labels portion of model training.

All datasets were uploaded pre-cleaned, so most of my work was done on the feature engineering side. I did, however, convert a few columns into numerical values from string and object data types so that those features could be included in the model. I also dropped the first feature in the ‘train features’ dataset, as it contained unique identifiers for each compound, and held no significance when modeling.

Of important note, under the ‘cp_type’ feature, there were two values: ‘ctl_vehicle’ and ‘trt_cp’. The former represents a compound known as a ‘control’ where the latter represents an experimental compound. Controls are used to provide a baseline in a drug trial and are given to some patients in the test population to either validate or reject the null hypothesis. ‘Ctl_vehicle’ was converted to a ‘0’ quantity and ‘trt_cp’ became a ‘1’, signaling that the observation containing a ‘0’ would possess a control compound, and thus has no mechanism of action.

After some EDA, which I will explain next, I dropped the same 42 features from the ‘train features’ and ‘test features’ datasets and the final shape of the train, ‘train scored’ and test datasets became (23814, 833), (23814, 206), and (3982, 833), respectively.

Exploratory Data Analysis

For this competition, we were only provided 25% of the total test data. That meant that the other 75% would be used as the test for a valid submission at competition deadline, and that the provided train and test data may or may not be a proportional representation of the full dataset. Knowing this, I decided to focus on features that would appear as outliers.

The following figures illustrate the difference in distribution between the cell expression features versus the cell expression features. For reference, all cell expression features ranged between (13640, 14293) unique values, while all gene expression features ranged between (2999, 14317) unique values.

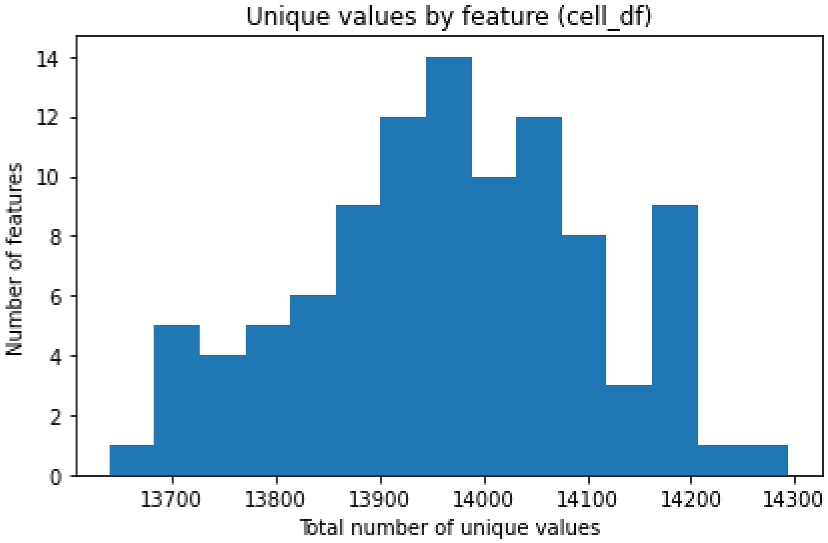

Figure 1: Histogram of unique value measures for each cell expression feature.

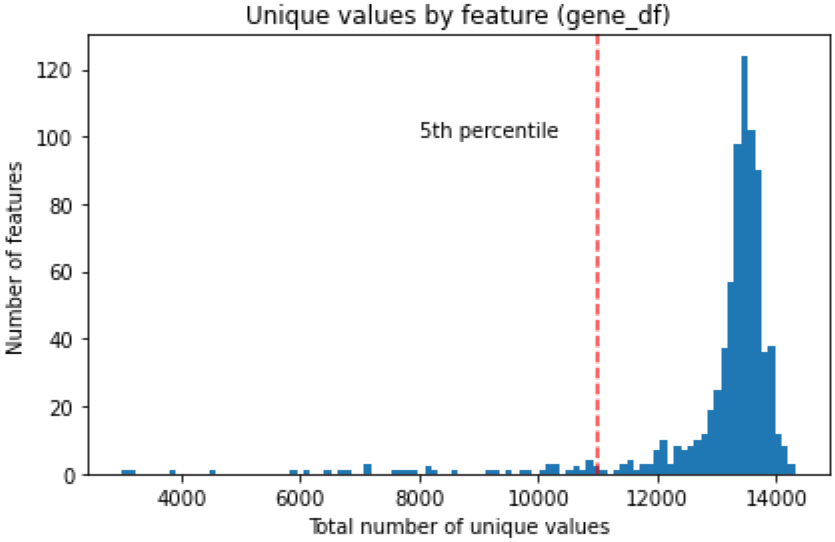

Figure 2: Histogram of unique value measures for each gene expression feature, with the 5th percentile marked.

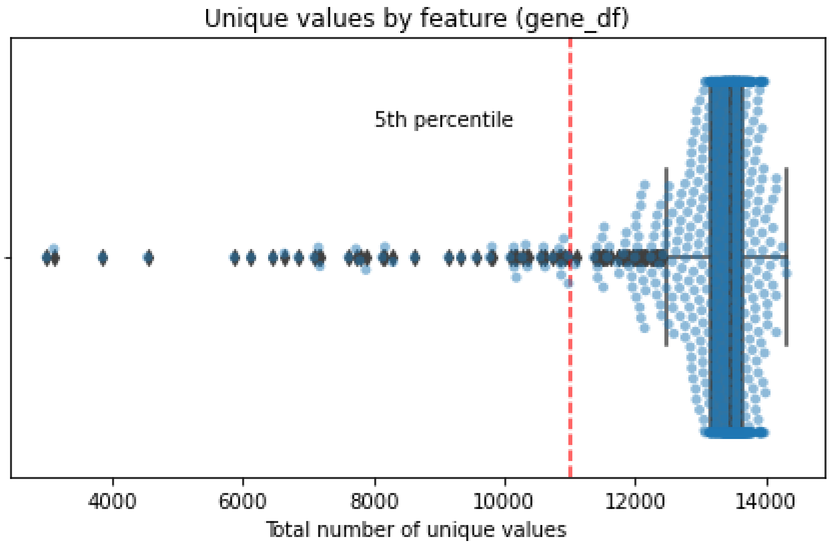

Figure 3: Box & Swarm plot of the unique values for each gene expression feature, with the 5th percentile marked.

The above figures clearly visualizes the variance and lack thereof among the grouping of features. In other words, when tested on the same chemical compound, certain genes behave the same more often, while others experience a varying amount of change that is much more dependent on the compound being tested. And to say this another way, some genes have a more predictive outcome versus others.

Note: The 5th percentile is marked on the gene expression dataframe because that value lies beyond the outer gate of the 1st quartile, which means those features are statistically extreme outliers.

To illustrate the difference between even the most extreme features, grouping by gene expression features and cell expression features, the following figure visualizes the two features with the most identical values at any given value.

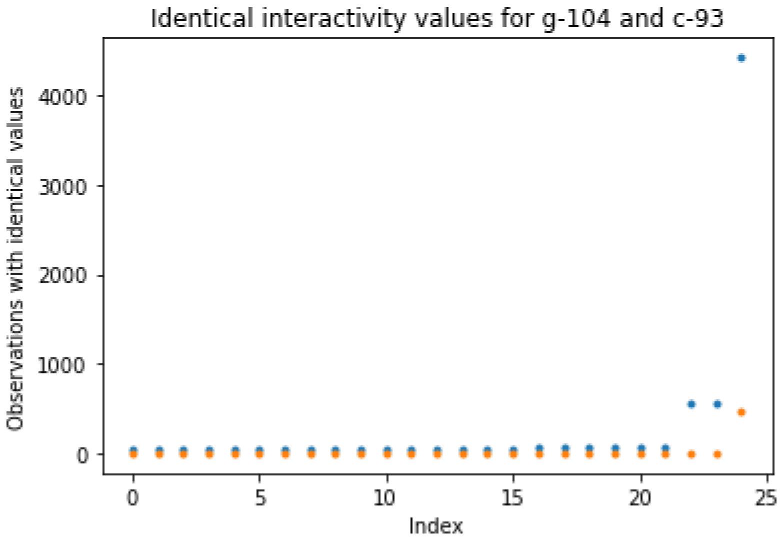

Figure 4: g-104 plotted against c-93. Two features possessing the most identical values at a single value, from their respective groupings. The index represents the trailing 25 values from each feature, that are also identical at their respective value.

As illustrated above, g-104 has an identical expression value for over 4000 observations. That represents approximately 20% of the total observations (not accounting for 'control' observations). The proportion of observations being identical by the most extreme cell expression feature, c-93, is far less at approximately 2% of total observations. The following figure will further illustrate the variance among the different feature groupings.

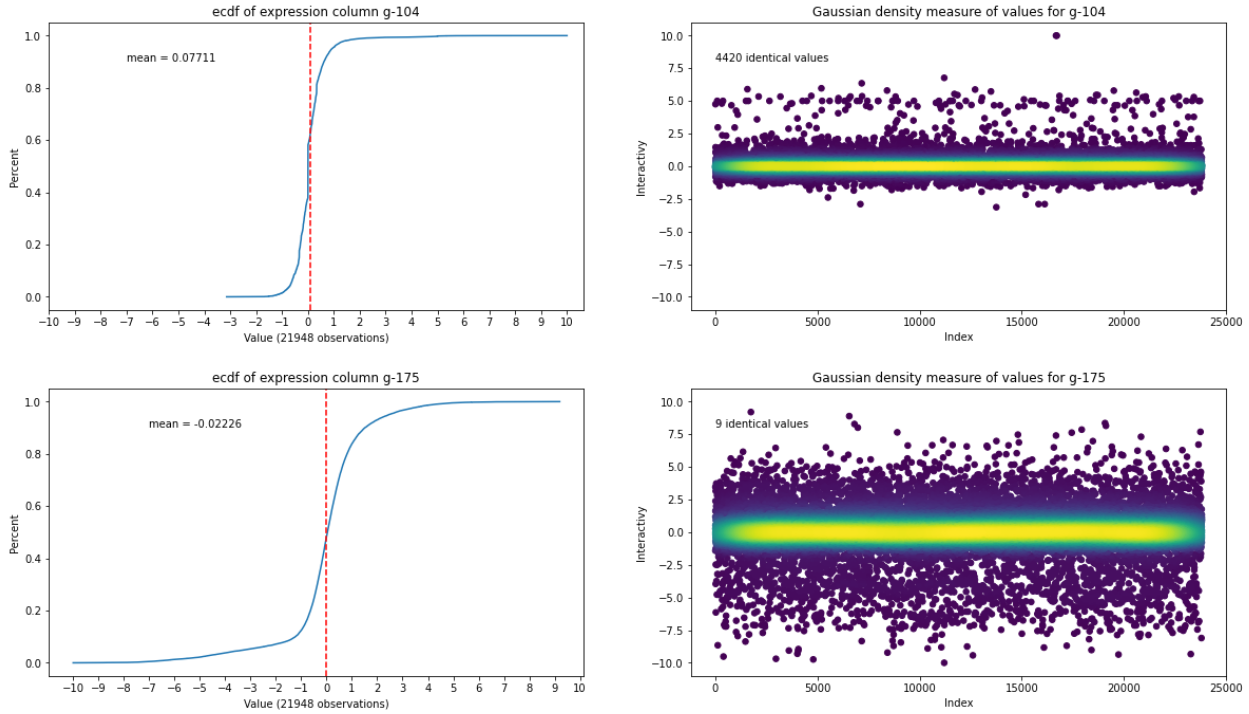

Figure 5: ECDF & Gaussian density plots of features g-104 and g-175, respectively, with g-104 containing the most identical expression values at 0.0 interactivity, and g-175 containing the least amount of identical expression values, with the most being at 0.0 interactivity.

Examining g-104 first, the ECDF plot experiences a sharp incline at and around 0.0, signifying that a disproportional amount of observations are made at and around this value. Then to the right of that, the Gaussian density plot visualizes through brightness of plotting that the evaluation of the ECDF plot is indeed true.

Interestingly enough, g-104 has very few negative values which could mean any number of things, but for certain, it means that the likelihood of experiencing a negative outcome for a random sample of observations for this feature is very slim.

Inversely, g-175's ECDF plot experiences a much more gradual incline as it traverses the graph. It also has a much longer head and tail than g-104. A gradual rise coupled with a longer head and tail means that the observations are distributed much more sporatically, and implies that the observation is likely dependent on the tested chemical compound. Whereas g-104 has a much higher likelihood of experiencing the same amount of change.

On opposite ends of the spectrum of alikeness, g-175 behaves in a much less predictive nature than g-104. Because there is no explanation for where or what each gene affects within the body, the observations hold a key to theorizing what they may do. For example, g-104 could exist within the body as a gene that experiences little to no structural change for any given test compound, where g-175 is in an area of the body that is extremely sensitive to test compounds.

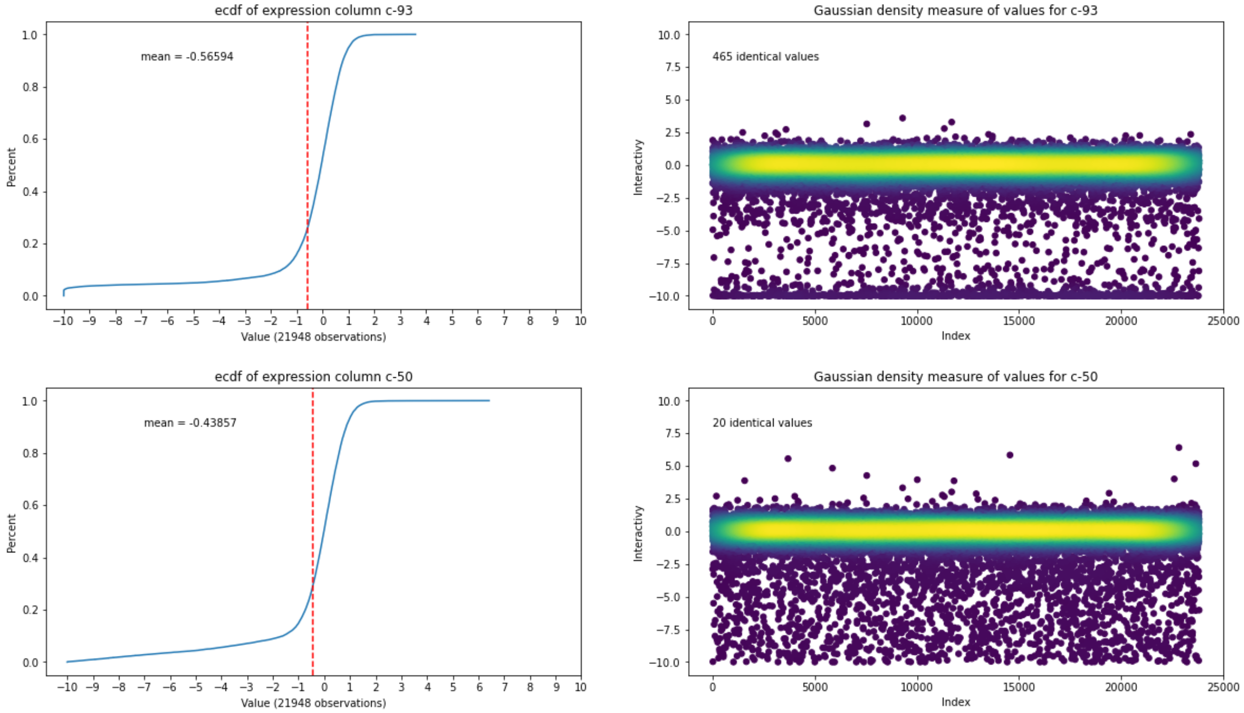

Figure 6: ECDF and Gaussian density plots for features c-93 & c-50, respectively, with c-93 containing the most identical expression values at -10.0 interactivity, and c-50 containing the least amount of identical expression values, with the most being at -10.0 interactivity.

Similar to the gene expression feature g-175 examined earlier, c-93 has a gradually inclining shape with a long tail. Except this feature has approximately 2% of its total observations experiencing a level of -10.0 interactivity. From there it slopes more severely around 0.0, and experiences no values greater than 5.0. The Gaussian density plot shows that more clearly, but given the liklihood of outcomes, experiencing exactly a -10.0 value is far more likely than experiencing any other individual value.

Like the other three features examined earlier, c-50 experiences a severe slope around 0.0 before leveling off. This feature has the highest likelihood of experiencing a different value for any given test compound.

Both features, c-93 and c-50, experienced very little interactivity at levels above 2.5. Further examination may be required of all cell expression features to see if little positive interactivity is common or not. However, one could theorize that it is much more likely that cells experience a decay in viability versus enhancement from a test compound.

Model Selection

This competition is a multilabel classification problem. Meaning, probabilities of each class are being predicted on an individual basis, i.e., one compound may belong to multiple classes. Then within the larger scale of the problem, the log loss was the evaluation metric, and it is found by computing the log of the likelihood function across a range of probability predictions, then multiplying that output by -1.

Having a predetermined evaluation criteria, I decided to implement the classification models: SGDClassifier, LogisticRegression, XGBClassifier and RandomForestClassifier, and wrapped each of these estimators within a OneVsRestClassifier. For tuning each of these models, I wanted to balance computational speed with accuracy of predictions. Time was on my side, so if I wanted to implement a model that took hours or days to train, I could have. But from a business sense, computational speed is money, so I wanted to find a model that balanced speed and efficacy with accuracy.

There were some drawbacks by having this dataset be hosted by a competition. For example, I could only remove a certain number of features during the preprocessing stage or my predictions would be disqualified during the submission process. Even though the code was syntactically correct, the hidden checks and balances in place by Kaggle prevented the removal of too many features from the testing dataset. Not a huge issue, but after I had identified features that were far beyond the outer gate of 1QR, I could only remove some of them and not all of them. This limitation was for the competition only, but on my own I could do as I please. Still though, for my own evaluation, I decided to only remove outliers past the 5th percentile and not do a mass feature removal.

I used KFold cross-validation, splitting the data into 5 folds for each model to be trained an equal amount. During this stage, the computational speed of each model became very apparent. The best model was my tuned SGDClassifier, which had the lowest mean log loss score, and was the fastest at making predictions.

Takeaways

In general, when you parallelize work, you increase the amount of efficiency and energy consumption on the job while decreasing reliability, which is achieved through more distributed processes. Rather than computing each label probability individually for the train set, splitting the dataset up and solving each classification problem simultaneously would save computational time and money, thus, parallelization was a priority for training models. A slower, more methodical algorithm could end up being the more accurate model, but I was willing to sacrifice some reliability for a speedy and efficient model.

The Kaggle hosts had purposely negated any domain specific knowledge by leaving out what each gene represented and what cell line each viability measure belonged to. This made it a purely numerical trend identification problem. You could, however, find trends and correlations among the values represented across gene and cell features. One way I achieved this was by grouping all gene features and cell features within a respective dataframe, and digging for common signatures across the observations. I then could theorize where one gene or cell feature may exist in the body in relation to another feature.

To bring more color to this theory, the emphasis the FDA and pharmaceutical companies place on developing medications with a clear, definable mechanism of action, is to limit what is referred to as “off-target” effects. In other words, the FDA needs to believe that, if given to the entirety of the population, they know it will behave in a safe, consistent manner, and activate within the body at its chemically-specified genetic and/or cellular level. So without any prior knowledge of what each gene or cell feature represents, it becomes reasonable to conclude that a cell feature that has a positive or negative signature value exists within an effective “on-target” or “off-target” relationship with the test compound. Also, where features express a value at or around 0.0 interactivity, it becomes reasonably clear that those features exist outside of an effective “on-target” or “off-target” relationship with the test compound, and therefore register a value that is otherwise unchanged from activation of the compound.

My focus on the outliers existed in a similar vein as my prior theory. If it was an outlier, and in my investigative case, existed outside of the 5th percentile of the total number unique values across all observations, those features existed in a capacity of either high-likelihood of the exact same interaction, whether it be positive, negative or unchanged. The operative phrase in the last sentence is “exact same interaction”. In other words, much more so than other features, the 5th percentile outliers interacted the exact same way across different test compounds. By removing those features, less ambiguity was present within the dataset, and less ambiguity and more certainty creates a more accurate model.

Future Research

This project unlocked a new world for me. The dry, non-descriptive nature of the dataset forced me to craft a theory and hypothesis purely from numerical trends, so I gained valuable experience with manipulating data. Further, the focus on hyperparameter tuning required me to look under the hood of some classification algorithms to find a way to maximize their performance.

Limiting dimensionality will be a continuing theme throughout my career as well. All features are valuable and possess a story within themselves, but not all are required to complete a generalization task. That’s why modeling with too many features creates a type of paralysis where conclusions can be made, but without much certainty. Also, that is why training a model on the most pertinent features is paramount to do, and is a vital step in the preprocessing of any dataset. Steps taken to limit dimensionality save computation time, efficiency, energy, money, and business-practical solutions can be made from accurate conclusions.

Long term, pharmaceutical companies could derive an incredible amount of value from models that can predict genetic and cellular outcomes from a given compound. In this case the task was to assign drug class labels to a range of chemical compounds, so the model I developed will be very good at generalizing many different compounds and assigning probabilities of their class(es), but the real value comes from models that can identify the safest signature values across a single drug class. Take for example ace-inhibitors. If a model could be developed with the same features as this Kaggle dataset with a few additions including length of treatment, decay of each genetic and cellular line represented as additional features (g-104_d1 signifying the first decay measure, g-104_d2 measuring the second measure, and so on) and life-expectancy, except the observations were all the different ace-inhibitors, a model could predict outcomes of the most safe and/or efficacious ace-inhibitor medication.

After creating and implementing a successful classification model, applying a deep learning model is the next step. Just like our nervous system, a model that could accurately weigh the vastness of the interactivity of this dataset may perform the best. Beyond that, differing values within each feature could have downstream effects on features that are closely correlated with the root feature, so mapping an accurate representation of the possible outcomes may only be achievable through deep learning.