Predicting Experimental Conditions based on scRNA Gene Expression Data

I. Introduction

Machine Learning (ML) has become a scalable method for classification based tasks within our age of ubiquitous data. We have seen an explosive growth especially in biological data, which is allowing scientists to perform a wide range of experiments much more rapidly and inexpensively than ever before. Therefore in our project we take a look at an epigenetics problem where we try to classify experiential conditions (KAT5, CBP, eGFP) based solely on multiple gene expression measured from five thousands different cells. This project aims to leverage one of ML's strong suits which is learning patterns and the intuition to properly classify experimental conditions from a large feature space. The methods, the reasoning behind method selection, and the results of this project are discussed in future sections.

II. Data Description

We obtained our data from the Johannes Gräff lab at EPFL measured by Doctoral Assistant Giulia Santoni. In the lab she measured the gene expression levels of 5000 different cells which contain 32,285 different genes. These different cells were then also given an experimental label in which we obtained our training set. The testing set is the same as the training in that it has the normalized counts but it contains no label. Therefore our task intuitively is to predict the experimental conditions of our testing set. *The data is available in the footer of our report* [1,2].

III. Methods

For pedagogical purposes, we briefly discuss our preprocessing steps, our linear method of choice, our nonlinear method of choice, and their reasoning for selection.

A. Preprocessing

Our feature space for our particular data is beyond massive and is subject to overfitting due to its high dimensionality. Therefore before fitting data into our models, it is conventional practice to preprocess our data. Since our data is very sparse (contains many zeros), one of our first preprocessing steps was to remove features (genes) that were seen across less than 10% of the measured cells. This allows us with a computational advantage in trying to feed a small dimensionality which is both optimal and efficient during our training.

B. Linear Methods - Multinomial Logistic Regression

The obvious choice for a linear method was the Multinomial Logistic Regression due to the data design, mainly because of the 3 categorical dependent values. We also experimented linear regression with and without regularization and obtained better results from Multinomial logistic regression. The method is very much similar to a standard logistic regression except that we have more than two possible outcomes.

C. Nonlinear Methods - XGBoost

Our nonlinear method of choice was the XGBoost which is an ensemble method that has been proven to be a great method for many supervised learning related tasks especially on large datasets. The big intuition that makes XGBoost a widely popular method is that it iteratively attempts to correct its previous errors based on every new decision tree that is constructed. We played around with a cross validation grid search which played with the *learning rate, tree max-depth, and number of estimators* to build our optimal model (Fig 1.).

---

1 Training Data: https://lcnwww.epfl.ch/bio322/project2022/train.csv.gz

2 Testing Data: https://lcnwww.epfl.ch/bio322/project2022/test.csv.gz

IV. Results

Method	Accuracy
Multinomial Logistic Regression	0.901
XGBoost (Unfiltered Data)	0.877

In our result we can see that the Multinomial Logistic Regression (Fig. 4) has performed best based on the accuracy metric produced by the Python frameworks. This is quite a surprise as a more sophisticated nonlinear model we would think would be better at fitting our data. Also with a large dimensionality it surprises us that a linear method captures the trends of the data points. However we believe this is the case due to performing a dimensionality reduction. Therefore we believe our methods' performance is actually quite surprising from what we anticipated.

V. Discussion

While we can see that the linear method performs better in this particular classification task, we also wanted to provide several discussion points on our data and our best models performance. In terms of feature importance, it is safe to say that some genes played a bigger importance in determining the experimental condition and lucky for us, XGBoosts Python framework (Fig. 2) nicely provides us with feature importance in relation to our trained model. By deducing our feature space in our preprocessing, we have already removed some irrelevant features but even then we can see that some particular gene expressions helped guide our model into forming their classification.

Principal Component Analysis (PCA) was used to visualize the data, both raw and filtered sets, by reducing the dimensionality (Fig 3.). We can see that there is no clear distinction between the different experimental conditions, which can be explained with the low explained variance ratio obtained performing the PCA. Nevertheless, CBP and KAT5 are very closely related.

And lastly we wanted to address our models classification capabilities which gave us a good inference of what our model did well and what it didn't do well. We therefore plotted our confusion matrix to evaluate the performances of our models. In our logistic regression we can see that the model particularly struggled with the KAT5 experimental condition, across both nonlinear and linear methods but the overall accuracy was better in the XGBoost.

VI. Conclusion

Some thoughts to wrap up our analysis are the fact that there is always more we can do to fine tune our models. We also acknowledge that there are probably more robust methods found in the current literature like sparse-input neural networks that may provide an even better accuracy for our model. However we believe we covered a large portion of machine learning topics in this project alone and got to find practical use in all we learned this semester.

---

1 Training Data: https://lcnwww.epfl.ch/bio322/project2022/train.csv.gz

2 Testing Data: https://lcnwww.epfl.ch/bio322/project2022/test.csv.gz

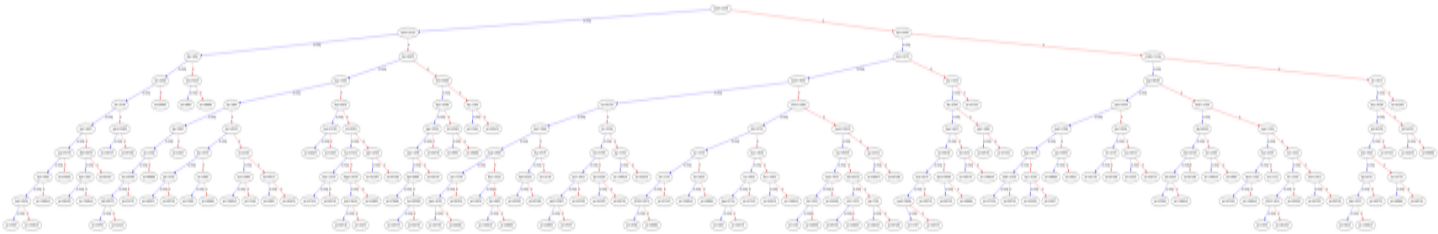

Fig 1. Optimal XGBoost Tree

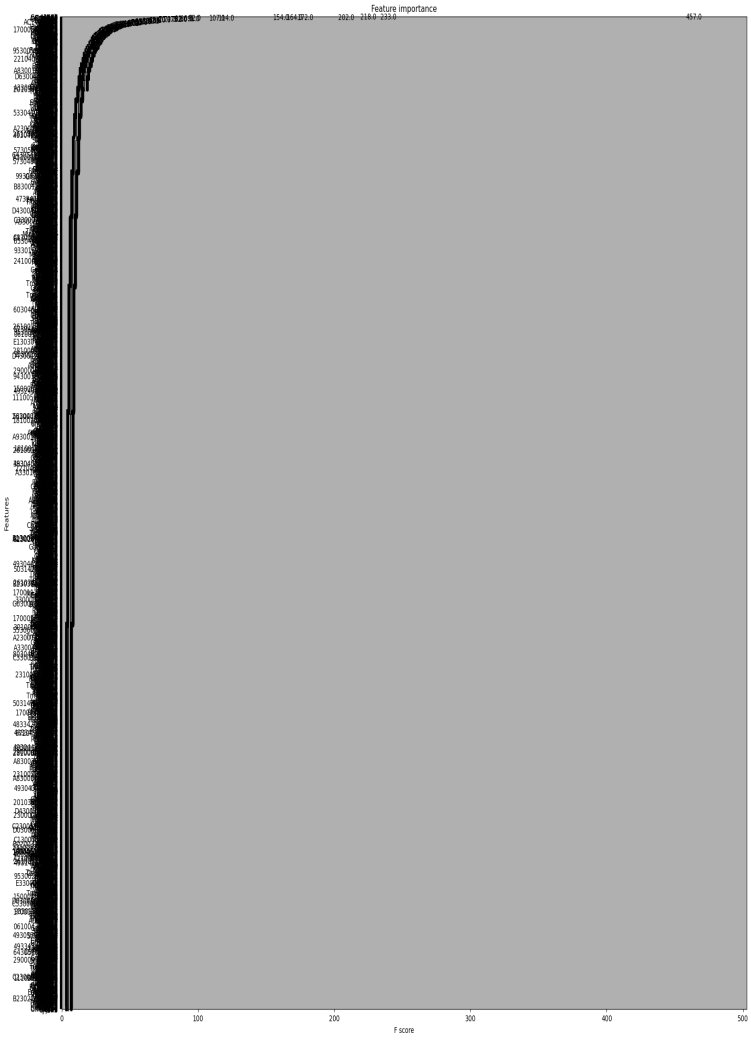

Fig 2. Feature Importance from XGBoost Python Framework

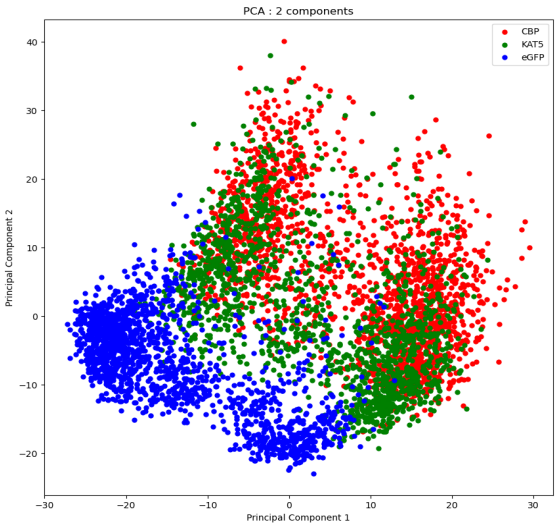

Fig 3. PCA in 2 dimensions

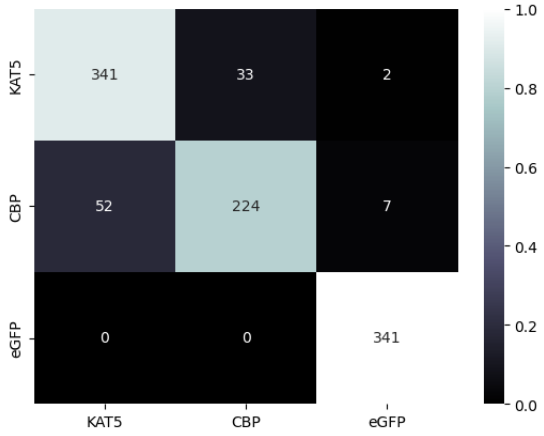

Fig 4. Confusion Matrix of Logistic Regression (best model)

1 Training Data: https://lcnwww.epfl.ch/bio322/project2022/train.csv.gz

2 Testing Data: https://lcnwww.epfl.ch/bio322/project2022/test.csv.gz