Applications of Machine Learning on Cancerous mRNA Expression Data

Benjamin P. Deutsch
Department of Physics & Astronomy, California State University, Long Beach, Long Beach, CA 90840
(Dated: December 17, 2017)

Cancer is the second largest killer of Americans, according to the Center for Diesease Control [1]. Work on cancer research must expand to analytically intensive engines, for means to locate similar traits, predictions in the populations and with these tools eventually stablize the apparent growth of the disease, the world over. Here, we present a numerical and data intensive apppoach for reconginizing and predicting these cancers given a dataset of clinical sequenced mRNA results, provided by the Synpase BioSage Networks (Illumina HiSeq). Through programmatic development of a protocol, using aggregative scaling and fine tuned supervised learning modules; K-nearest Neighbor, Decision Trees and Support Vector modeling. We acheive a prediction engine for five various cancers, with ranging preformance up to 50%. Using confusion matrices in conjucation with test data we encounter rough similarities in various cancer strains and discuss implications. We continue onto visualizing the strenghts and weakness of the predictions preformance on one layer of data, subsetting work done by The Cancer Genome Atlas Network.

Keywords: Oncology, Supervised Learning, Data Visualization, Gene Expression, Machine Learning

I. INTRODUCTION

Within the last 50 years, there has been meteoric rise in the abilities of first level genomic sequencing, from early location-specific primer methods to modern-day high throughput (HPT) assays. With this advancement in experimental technologies, the analysis side has fallen exceedingly short. It has only been within the last few decades that large volume computational analysis has been possible in the biological sciences. As there are many effects and outcomes of a test genome the process of sequencing can happen through a variety of different methods, from "Sanger sequencing" to "Ion Torrent sequencing", each method, unique in its own approach and disadvantages.

RNA, a component in the transcription-translation engine, can also be sequence through a Illumnia HiSeQ (HPT) Platform, such is the case with (syn:4301332) that was used further in this paper was gather from the Synapse BioSage Database[2]. RNA "reads" the output of transcription (nucleus) and translates or expresses the message into proteins for use within the body. Complications here may arise a message may be over and under expressed this in turn affects the proteins, creating cancerous regions within the cells. Regions of this genetic make-up also have the ability to mutate or change more commonly than other areas, this is thought to be for many reasons and is still an open area of research. However, in the clinical stage identification and early detection are the utmost importance to the patient, as most mortality rates are dependent on when the cancer is detected.

Current literature is thin, containing in the hundreds of papers relating cancer prognosis through machine learning, while surprising this is to be expected as the field is young. The difficultly here is the safety and attitude toward automated healthcare in the publics knowledge, much work must be done to bridge the gap between the two. Previous attempts to categorize the mutation sequencing with the analogous cancers they promote, have utilized visualizations, and machine learning algorithms; neural nets, Gradient boost method, SVM, and tree building. Most are met with adequate results, but real interest lies in the creation of complexity through layering types of tests and various cancer to find similarities in origin. This data hungry approach has led to creation of loci for gathering this work, The Cancer Genome Atlas Pan-Cancer Atlas Project (PAN-CAN). Here we attempt to exploit a layer of the recent Nature article with different purpose, to reverse engineer commonalties and closeness between cancer se-

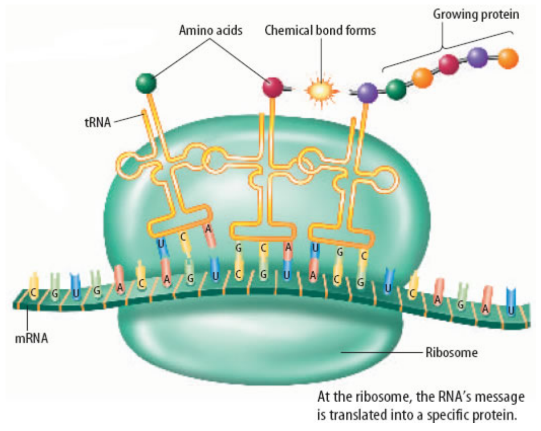

Amino acids
Chemical bond forms
Growing protein
tRNA
mRNA
Ribosome
At the ribosome, the RNA's message is translated into a specific protein.

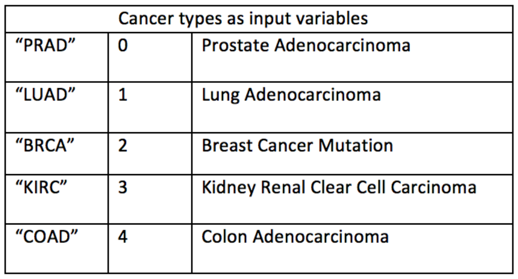
FIG. 1: Example of Translation. From lower left to upper right: 1. mRNA fed into Ribosome(green) 2. tRNA attachs to complement 3. Bond creation 4. Resultant protein chain

quencings. Perhaps in the future to offer early prognosis.

With intent, the data gathered from Synapse, is a subset of the full project. Careful notice has been taken to take a manageable portion of the vast library. The data was observed to have, cases/patients and their respective gene profiles which are represented quantitatively, this was then split over to create a train and test array. Finally, in a separate file was placed the different cancers of each patient to have has a validating set.

II. EXPERIMENT

The data was loaded in Jupyter Notebook, with Python 3 as is the standard, we then set the framework by loading pandas, to use a data frame navigator and Numpy for simple operation on series and the arrays. Most preprocessing of the data occurred with using the above two libraries as not much was needed in terms of cleaning, just simple column shuffling. All algorithms used a are part of the Scikit.learn libraries and will be spoke on individually. Scikit.learn, represents an availability to machine learning for academic purposes as it is free and not too difficult in practice. Firstly, what was done, was the creation of a categorical variable to the train data frame, this meant aligning the labels of the various cancers with their respective patients. Each cancer ['PRAD':0,'LUAD':1, 'BRCA':2, 'KIRC':3,'COAD':4] received a dummy index to call on later.



Cancer types as input variables		
"PRAD"	0	Prostate Adenocarcinoma
"LUAD"	1	Lung Adenocarcinoma
"BRCA"	2	Breast Cancer Mutation
"KIRC"	3	Kidney Renal Clear Cell Carcinoma
"COAD"	4	Colon Adenocarcinoma

FIG. 2: Chart to distinguish catagorical variables of cancer placed in coding

The Sckikit.learn scaler was used as a means to make the programming as efficient has possible. This is to say we can represent the same data value points without the incredible volume of genetic variance. We proceeded to check if any information had been lost and running a simple cumulative sum program, visually indicates the opposite. It standardizes the data and allow of easy manipulation in the early stages. Using this in conjunction with the principal component analysis (PCA), creates a dimensional set of eigenvectors unique to each of the cases/patients and can be placed into 2 dimensional graph [3]. However, as we wish to select on different variable projection in different planes can help see this more clearly. PCA by unsupervised learning reduces linearity the degrees of freedom in the data so indication and noise in the data can be filtered out later [4].

This data or the approximations of this data can then be used to fit the test data which will be calculated in much the same way. It is important to note that the N-space or component space here is not defined arbitrarily, but each individual column in the test and training data set indicate the same genetic information as to which gene is being monitored. Containing different value here or shifting, would destroy any order to the analysis. This can be checked by seeing if the learner is picking out more commonly one value than another, we can run simple histogram showing a Gaussian distribution.

A. K-nearest Neighbors (KNN)

We can at this point run a variety of agglomerative functions under SciKit.learn to tune our model of these, we will begin with one supervised learning model, K clustering or as it is known in Scikit.learn, âĂIJK-Nearest NeighborsâĂİ. For the data scientist, this learner is routinely used as a benchmark for large dataset. However, it can be quite powerful in its straight forward approach, exercising over the training data the algorithms begin by uniformity positioning the arguments in Cartesian space and analyzing the Euclidean distance between points(1) by distinguishing closer points having stronger weights than those of more distance neighbors, running through the whole dataset and establishing a bias [5]. The function will now associate conditional probability with the variables indicated grouping each into specialized. Now each occupancy x can be placed "next to" its neighbor of relative weight, which are then further grouped with others(2).

$$d(x, x') = \sqrt{(x_1 - x'_1)^2 + \cdots + (x_n - x'_n)^2} \tag{1}$$

$$P(y = j|X = x) = \frac{1}{\kappa} \sum_{n=i\epsilon A} I(y^i = j) \tag{2}$$

For basic data of few parameters, this works remarkably well as the loci for each group is distinct large and as is with gravity, highly attractive. Where difficulty arises is with multiple input parameters and non-linearly related data. Some packages have been created to facilitate such errors however, knowledge on the data shape or momentum is necessary. (N-neighbors=30, 60 ,100) Here we will use it as before to benchmark success in the prediction of test data, that has been bundled for the same directory.

B. Desision Tree Classifier (DTC)

We will also utilize a decision tree classifier, again this is a simple supervised model good for understanding the output or prediction. Decision trees have made leaps in machine learning methods, more advanced package and shorter run times have made this a staple of the data science community. This was chosen for the straight forward physical understanding of how the algorithm is classifying, If we wish to understand the ramifications of associating a patient with one group over another we wish to know why that split on data was made.

Decision trees consist of two main parts, leaves and branches as you would imagine. Leaves, represent labels of parameters each one different and trying to give all possibilities for optimal binning. The branches are the intersection of the different parameters, allowing for all possibility to be monitored. Scaling must be done for tree modeling as it can be difficult for large skewed data unbiased learners. We monitor this in preprocessing against our train data. Further difficult is encountered in setting leaf sizes and max depth of the data, these parameters explicitly deal with the learnerâĂŹs sensitivity and offsetting these has the potential to over fit the learner, releasing any possible predictions as outliers.

C. Support Vector Machines (SVM)

Alternatively, to the pervious test which have issues with large numbers of discretionary variable, support vector machines utilizes these variables to create an estimator that it overlay on top of the data referred to as a hyperplane. The support vectors can be aligned against the data set in multiple ways, the algorithm will preserve which alignment of the support vector best categorizes the clusters. We can see in documentations that when grouping the clusters the SVM engine will fit a kernel function to avoid an over fit if flagged for outliers, these can be of any class; linear, polynomial, rbf, and sigmoid [6]. Certain packages and customized kernels can extend the fitting capabilities here, our choice centers on the separating multiple clusters of which the majority have similar dimensions.

III. RESULTS AND ANALYSIS

Before aggregating results, it was important to distinguish if our training data was taking bias toward certain parameters over other. This would lead to overall kurtosis in the later stage. So to normalize we perform a simple frequency test on the reduced parameters (gene expressions).

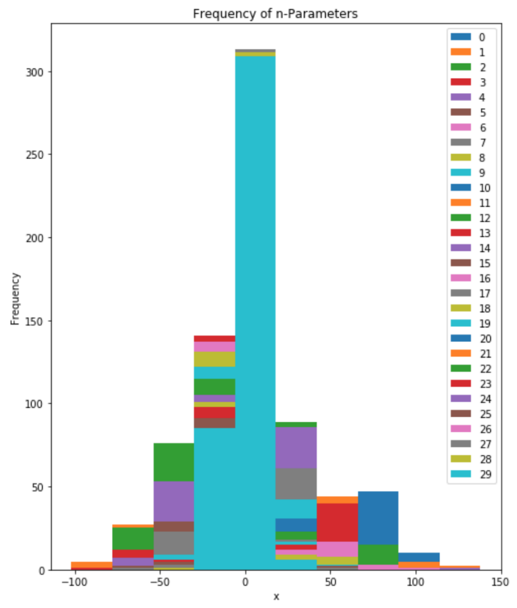


FIG. 3: A gaussian distribution of n-Parameters set to 30.

Fig 3 shows a a normailzed gaussian distribution indicated that the learned set, weights no columns of the patient condition over the other.

For analysis of the machine learning algorithms, we selected a confusion matrix as the most direct way to visualize the real and prediction outcomes. After each engine was tuned to its optimal parameters via area under the curve measurements (AUC), a confusion matrix (prediction, real) was output.

	PRAD	LUAD	BRCA	KIRA	COAD
PRAD	24	13	18	9	6
LUAD	16	21	18	10	5
BRCA	41	30	31	14	11
KIRA	24	15	18	17	4
COAD	15	11	9	0	3

TABLE I: Confusion Matrix for K-Neighbor Classifier

These confusion matrices contain true values across the y and prediction values along the x axis, so they can be read on the diagonal as a marking of the classifiers

	PRAD	LUAD	BRCA	KIRA	COAD
PRAD	26	7	24	10	3
LUAD	25	11	22	12	0
BRCA	41	24	37	42	5
KIRA	24	10	23	17	1
COAD	15	7	13	0	1

TABLE II: Confusion Matrix for Decision Tree Classifier

	PRAD	LUAD	BRCA	KIRA	COAD
PRAD	29	13	14	9	5
LUAD	19	23	15	9	4
BRCA	43	35	20	33	17
KIRA	24	21	14	17	1
COAD	17	12	6	1	2

TABLE III: Confusion Matrix for Support Vector Machine

success. For example, the K-neighbors cell [1,1] contains value 24, meaning 24 times the classifier predicted Prostate Adenocarcinoma and it was proved true. Below this value we see 16, the classifier selected Prostate Adenocarcinoma but was a false positive of Lung Adenocarcinoma. Quick algebra on these matrices we really get a good idea of how each classifier is doing, and also when it does not select correctly, how it is going wrong. This information is also important to not only improving the classifiers but distinguishing the similarity in the cancers expression data. The higher the values along the diagonal and lower values in off-diagonal indicate a stronger ability to predict true cancers markers.

Taking the trace of each matrix give the following; KNN at 96, DTC at 92 and SVM at 91. Remarkably, most of the classifier algorithms had about the same true-prediction output with KNN containing most correct analysis. Taking the off-diagonals we find that the most widely varied result for false positives being the DTC with entries of 42 in both hemispheres.

Finally, perhaps most surprising of all this the apparent shift in the learners on every confusion matrix, we can notice from left to the right the numbers deteriorating, especially near the lower right quadrant. This is strange because we selected data that was unskewed and normal, there should be lack in any column. The reason becomes clear when the "learned" data (red points) is plotted in 3D against that of the original set (colorized indicating cancer type) in PCA space. We see from the graph that for KIRA: Brown, COAD: Light Blue, there was hardly any overlap to train or even plot these points, alternatively for PRAD: Yellow, LUAD: Green, BRCA: Purple there was stronger correlation, up to 47.7% in BRCA and PRAD (DTC) with the training data.

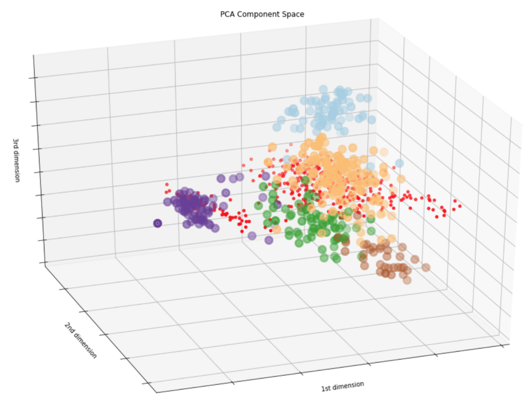


FIG. 4: PCA space plot with cancers Train data [PRAD: Yellow, LUAD: Green, BRCA: Purple, KIRA: Brown, COAD: Light Blue] and Test data as red points.

IV. CONCLUSION & DISSCUSSION

All in all, current day cancer prevention is only as good as the patients ability to seek out early screenings. In this project, we proposed a 2-fold approach, a pipeline for the detection and classification of various types of cancer from a machine learning model and the ability to connect the similarities of gene expressions via mRNA to other closely associated type of tumors. While the final results of this projects seem dubious with prediction rate capping at about 50%, it is important to remember this is but one level of the literatures progress. Current standards many research teams routinely gather percentages of 60 to 70%[7], still have a long way to enter into clinical realms but momentum is growing. Similarities were found in the mRNA expressions of 3 deadly tumors [PRAD, LUAD and BRCA], from here we guide further work into uncover the specific genes which share these traits. Moreover, we may be able to quantify these connection, this would also be interesting to pursue. In this data hungry field many perspectives must be taken in order to retrieve the full picture. Like the complement of principal component analysis, supervised learning and confusion matrices, using differing tools such as DNA methylation or processed clinical data can shed light into the complex and necessarily computational nature of this illness.

V. ACKNOWLEDGMENTS

We would like to acknowledge the California State University at Long Beach for their resources and continued support. Damien Benvenste PhD. for his mentorship and my peers for their patience.

[1] Center for Disease Control. *Number of deaths for leading causes of death* CDC, https://www.cdc.gov/nchs/fastats/leading-causes-of-death.htm

[2] The Cancer Genome Atlas Network, et al. *The Cancer Genome Atlas Pan-Cancer analysis project*, https://www.nature-.com/articles/ng.2764.pdf

[3] Damien Benveniste. *Introduction to Data Science, Lecture notes*. California State University, Long Beach; https://github.-com/damienGitUserName/Intro-to-Data-Science/tree/master/Lectures

[4] Jake VanderPlas. *In Depth Principal Component Analysis* Python Data Science Handbook, Oreilly; 2016

[5] Kevin Zakka. *A Complete Guide to K-Nearest Neighbors with Applications in Python and R*, https://kevinzakka.github.io-/2016/07/13/k-nearest-neighbor/

[6] Sunil Ray. *Understanding support vector algorithim...* Analytic Vidhya, .analyticsvidhya.com/blog-/2017/09/understaing-support-vector-machine-example-code/

[7] Konstantina Kourou, et al. *Machine learning applications in cancer prognosis and prediction*, Computational and Structural Biotechnology, Journal 13 (2015) 8-17