Proposal of Kaggle's Personalized Medicine Competition

Wang Lu
October 28th, 2017

Domain Background

A lot has been said during the past several years about how precision medicine and, more concretely, how genetic testing is going to disrupt the way diseases like cancer are treated. But this is only partially happening due to the huge amount of manual work still required. Once sequenced, a cancer tumor can have thousands of genetic mutations. But the challenge is distinguishing the mutations that contribute to tumor growth (drivers) from the neutral mutations (passengers). The host, Memorial Sloan Kettering Cancer Center (MSKCC) , has been maintaining the database OncoKB [1] for the purpose of knowledge sharing of mutation effects among oncologist. Currently this interpretation of genetic mutations is being done manually. This is a very time-consuming task where a clinical pathologist has to manually review and classify every single genetic mutation based on evidence from text-based clinical literature. Therefore, the task is to develop a Machine Learning algorithm that, using an expert-annotated knowledge base as a baseline, automatically classifies genetic variations.

Historical Studies

Since 2004 there are consequent publications trying to address this problem by developing automatic computational methods, and achieved encouraging results, such as CHASM (**C**ancer-specific **H**igh-throughput **A**nnotation of **S**omatic **M**utations) [2,3] (led by Carter et al., published on Cancer Res and Gene Function Analysis respectively), and FIS (Functional Impact Score) [4] (leb by scientists in Computational Biology center, MSK). However, all of these methods requiring strong domain knowledge and high sensitivity to pattern such as protein first and secondary structure characteristics or evolutionary conservation features, while take little use of literature on clinical impacts or pathological mechanism of that variant. On the other hand, the natural language processing tools [5,6,7] in cancer study focus mainly on identifier detection and concepts locating, which are not shrewd enough to predict the functional consequences of a mutant. A NLP based automation method which is able to annotate the functional impact of variants of intense research interest is much needed.

References:

[1] *OncoKB: A Precision Oncology Knowledge Base*
[2] *Cancer-specific High-throughput Annotation of Somatic Mutations: computational prediction of driver missense mutations*
[3] *Predicting the Functional Consequences of Somatic Missense Mutations Found in Tumors*
[4] *Predicting the functional impact of protein mutations: application to cancer genomic*
[5] *tmVar: A text mining approach for extracting sequence variants in biomedical literature*
[6] *TaggerOne: Joint Named Entity Recognition and Normalization with Semi-Markov Models*
[7] *GNormPlus: An Integrative Approach for Tagging Gene, Gene Family and Protein Domain*

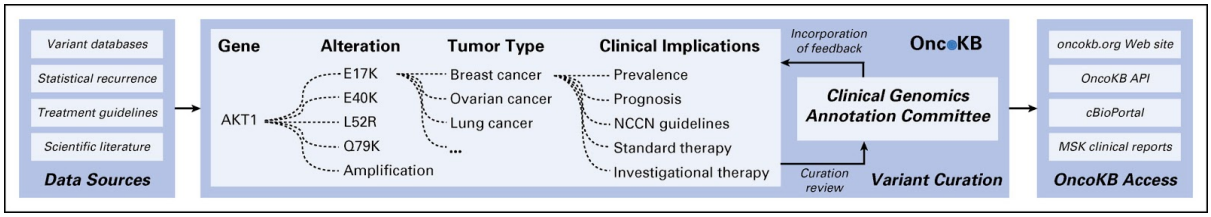
Problem Statement

According to the host, they want algorithms to classify genetic mutations based on clinical evidence (text). There are nine different classes a genetic mutation can be classified on. This is not a trivial task since interpreting clinical evidence is very challenging even for human specialists. Therefore, modeling the clinical evidence (text) will be critical for the success of final approach. Both, training and test, data sets are provided via two different files. One (training/test_variants) provides the information about the genetic mutations, whereas the other (training/test_text) provides the clinical evidence (text) that our human experts used to classify the genetic mutations. Both are linked via the ID field. Therefore, the genetic mutation (row) with ID=15 in the file training_variants, was classified using the clinical evidence (text) from the row with ID=15 in the file training_text. Noticebly, the host explicitly mentioned there were machine-generated data to prevent hand labeling. Participants will submit all the results of the classification algorithm develop, but they will ignore the machine-generated samples.

Datasets and Inputs

More background about input datasets:

As stated before, variants in those datasets are from real-world cancer cell sequencing, all annotations (labels) are manually added by oncologists based on clinical or molecular studies on the corresponding variant. Text information annotated to each variant are also hand-picked by those domain experts. Given gene AKT1, variant E17K, their workflow according to their official website [8] is like this:



Details of input datasets:

- training_variants - a comma separated file containing the description of the genetic mutations used for training. Fields are ID (the id of the row used to link the mutation to the clinical evidence), Gene (the gene where this genetic mutation is located), Variation (the aminoacid change for this mutations), Class (1-9 the class this genetic mutation has been classified on)
- training_text - a double pipe (||) delimited file that contains the clinical evidence (text) used to classify genetic mutations. Fields are ID (the id of the row used to link the clinical evidence to the genetic mutation), Text (the clinical evidence used to classify the genetic mutation)
- test_variants - a comma separated file containing the description of the genetic mutations used for training. Fields are ID (the id of the row used to link the mutation to the clinical evidence), Gene (the gene where this genetic mutation is located), Variation (the aminoacid change for this mutations)
- test_text - a double pipe (||) delimited file that contains the clinical evidence (text) used to classify genetic mutations. Fields are ID(the id of the row used to link the clinical evidence to the genetic mutation), Text (the clinical evidence used to classify the genetic mutation)
- submissionSample - a sample submission file in the correct format

References:

[8] *OncoKB: A Precision Oncology Knowledge Base*

Solution Statement

For this natural language based multi classification problem, the two key problems would be: firstly, how to extract most useful features for later model training; and secondly, how to find the most suitable algorithm for that extracted features. To best capture effective features, I need to have deeper understanding of the problem and datasets by asking myself questions such as: what the 9 anonymized labels actually mean? What's the biological relationship of these catagories? Why those papers instead of others, stand out as basis of classification? What are the characteristics of narratives in papers that most convince oncologists in classification? How to reduce the noise and maximize signal-to-noise ratio in text? Then, to find the most suitable algorithms, I should know how big the datasets are and what's the relationship within features. For this specific problem, since each data point is one cancer-related variant, our input are relatively small datasets. While since most features are extracted from text by vectorizer such as bag of words, tfidf etc, I would expect lots of features. I would try Logistic Regression, Decision Trees, Gradient Boosting, Linear SVM and xgb for this problem, and use log loss for evaluating the performance of my models. (AUC should also work for classification problem here but since log loss is taken by host to calculate rankings I would use only log loss for optimizing the method)

Benchmark Model

In this section, provide the details for a benchmark model or result that relates to the domain, problem statement, and intended solution. Ideally, the benchmark model or result contextualizes existing methods or known information in the domain and problem given, which could then be objectively compared to the solution. Describe how the benchmark model or result is measurable (can be measured by some metric and clearly observed) with thorough detail.

For benchmark model, I use gene-encoding features only based on the consensus that variants from same gene should have more related behaviours than those not in terms of functional consequences. According to the formula of metric the host uses for evaluation, the best score is 0.00, meaning 100% hit. The worst possible score is 34.26246 since the host had added a smoothing transformation. Score can be easily improved to 2.1 by simply throwing uniform guesses and it will keep climbing to 1.8 once weights of each class considered. I want to know how much it can improve with only addition of gene one-hot-encoding information. After running Logistic Regression on 1022 gene one-hot encoding features, I got an average score of 1.21216637352 from 5-fold validation without tuning. This gives the baseline from where the text features should start with.

Evaluation Metrics

For this competition, the evaluation metric used is Logarithmic Loss metric. Each observation is in one class and for each observation, there should be a predicted probability for each class. The metric is negative the log likelihood of the model that says each test observation is chosen independently from a distribution that places the submitted probability mass on the corresponding class, for each observation.

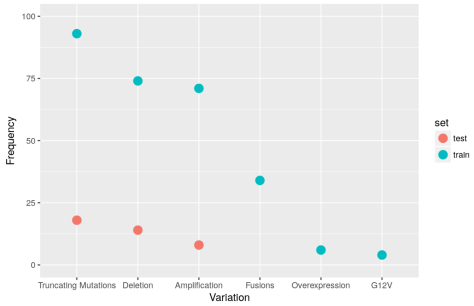
$$\text{logloss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j}\log(p_{i,j})$$

where N is the number of observations, M is the number of class labels, loglog is the natural logarithm, yi,jyi,j is 1 if observation ii is in class jj and 0 otherwise, and pi,jpi,j is the predicted probability that observation ii is in class jj.
Both the solution file and the submission file are CSV's where each row corresponds to one observation, and each column corresponds to a class. The solution has 1's and 0's (exactly one "1" in each row), while the submission consists of predicted probabilities.
The submitted probabilities need not sum to 1, because they will be rescaled (each is divided by the sum) so that they do before evaluation.
(Note: the actual submitted predicted probabilities are replaced
with max(min(p,1−10−15),10−15)max(min(p,1−10−15),10−15).)

Project Design

First, I will start with explorative data analysis using R. I will check the statistical description of individual features of the dataset by *summary(data)) t*o check how many unique genes, frequent variants etc. And I will compare the distribution in training and testing dataset. (Example is given below). Then I will look at interactions between pairs of features such as gene vs class, variation vs class. After all this I start to build the model.

```
##        ID            Gene                    Variation    Class
##  Min.   :   0   BRCA1  : 264   Truncating Mutations:  93   1:568
##  1st Qu.: 830   TP53   : 163   Deletion            :  74   2:452
##  Median :1660   EGFR   : 141   Amplification       :  71   3: 89
##  Mean   :1660   PTEN   : 126   Fusions             :  34   4:686
##  3rd Qu.:2490   BRCA2  : 125   Overexpression      :   6   5:242
##  Max.   :3320   KIT    :  99   G12V                :   4   6:275
##                 BRAF   :  93   E17K                :   3   7:953
##                 ALK    :  69   Q61H                :   3   8: 19
##                 (Other):2241   (Other)             :3033   9: 37
```



I will heavily need pandas, numpy and sklearn libraries and sometimes I need sns to show correlation or feature importance. Since the inputs are separated in 4 files I will combine them and handle it as one file. Primary features are only 3: *Gene, Variation and Text.* For the first two I will implement one-hot-encoding, label encoding, bag of words extraction, encoding length and all (basically as much engineering as I can think of). For *Text* I will consider more complex way to treat it such word-embedding, topic modeling and tfidf vectorising. But how to improve signal to noise ratio would be major concern. Preprocessing seemingly not necessary here, as most of data are counts, frequency. But it is to be confirmed.

Condering I will need to try different combinations of: ways to engineer features, extract features (tfidf,bow, word2vec,doc2vec,LDA), features selection (PCA, SVD), algorithms (logistic regression, SVM, DecsionTress, GradientBoosting, KNearestNeighbors, LinearDiscriminantAnalysis, XGB), parameters of algorithms etc., I need to build a *pipeline* gives flexibility to try all this. And an automation method GridSearchCV to reduce repeated work of hyper-parameters tuning.

Once I get a few good performing models, I will need to ensemble them for a better solution, by methods such as blending, stacking, bagging. It will be a trade-off that whether using strong performers but less number of ensembling, or weaker performers but greater size is better strategy. And balance the complexity with robustness of the model over unseen data.