PERSONALIZED CANCER DIAGNOSIS

Personalized cancer diagnosis

---

1. Business Problem

1.1. Description

Source: https://www.kaggle.com/c/msk-redefining-cancer-treatment/
Data: Memorial Sloan Kettering Cancer Center (MSKCC)
Download training_variants.zip and training_text.zip from Kaggle.
Context:
Source: https://www.kaggle.com/c/msk-redefining-cancer-treatment/discussion/35336#198462
Problem statement :
Whenever a patient has symptoms of cancer, the cancer tumour is taken out and sequenced. Thousands of genetic mutations may be present in the sequence. We need to distinguish the malignant mutations (drivers leading to tumour growth) from the benign (passenger) ones.
Classify the given genetic variations/mutations based on evidence from text-based clinical literature.

1.2. Source/Useful Links

Some articles and reference blogs about the problem statement

---

1. https://www.forbes.com/sites/matthewherper/2017/06/03/a-new-cancer-drug-helped-almost-everyone-who-took-it-almost-heres-what-it-teaches-us/#2a44ee2f6b25
2. https://www.youtube.com/watch?v=UwbuW7oK8rk
3. https://www.youtube.com/watch?v=qxXRKVompI8

1.3. Real-world/Business objectives and constraints.

- No low-latency requirement.
- Interpretability is important.
- Errors can be very costly.
- Probability of a data-point belonging to each class is needed. To ensure how sure the model is.

---

2. Machine Learning Problem Formulation

2.1. Data

2.1.1. Data Overview

- Source: https://www.kaggle.com/c/msk-redefining-cancer-treatment/data
- We have two data files: one conatins the information about the genetic mutations and the other contains the clinical evidence (text) that human experts/pathologists use to classify the genetic mutations.
- Both these data files are having a common column called ID
- Data file's information:
    - training_variants (ID , Gene, Variations, Class)
    - training_text (ID, Text)

2.1.2. Example Data Point

training_variants

```
ID,Gene,Variation,Class
0,FAM58A,Truncating Mutations,1
1,CBL,W802*,2
2,CBL,Q249E,2
...
```

training_text

```
ID,Text
0||Cyclin-dependent kinases (CDKs) regulate a variety of fundamental cellular processes. CDK10 stands out as one of the last orphan CDKs for which no activating cyclin has been identified and no kinase activity revealed. Previous work has shown that CDK10 silencing increases ETS2 (v-ets erythroblastosis virus E26 oncogene homolog 2)-driven activation of the MAPK pathway, which confers tamoxifen resistance to breast cancer cells. The precise mechanisms by which CDK10 modulates ETS2 activity, and more generally the functions of CDK10, remain elusive. Here we demonstrate that CDK10 is a cyclin-dependent kinase by identifying cyclin M as an activating cyclin. Cyclin M, an orphan cyclin, is the product of FAM58A, whose mutations cause STAR syndrome, a human developmental anomaly whose features include toe syndactyly, telecanthus, and anogenital and renal malformations. We show that STAR syndrome-associated cyclin M mutants
```

are unable to interact with CDK10. Cyclin M silencing phenocopies CDK10 silencing in increasing c-Raf and in conferring tamoxifen resistance to breast cancer cells. CDK10/cyclin M phosphorylates ETS2 in vitro, and in cells it positively controls ETS2 degradation by the proteasome. ETS2 protein levels are increased in cells derived from a STAR patient, and this increase is attributable to decreased cyclin M levels. Altogether, our results reveal an additional regulatory mechanism for ETS2, which plays key roles in cancer and development. They also shed light on the molecular mechanisms underlying STAR syndrome.Cyclin-dependent kinases (CDKs) play a pivotal role in the control of a number of fundamental cellular processes (1). The human genome contains 21 genes encoding proteins that can be considered as members of the CDK family owing to their sequence similarity with bona fide CDKs, those known to be activated by cyclins (2). Although discovered almost 20 y ago (3, 4), CDK10 remains one of the two CDKs without an identified cyclin partner. This knowledge gap has largely impeded the exploration of its biological functions. CDK10 can act as a positive cell cycle regulator in some cells (5, 6) or as a tumor suppressor in others (7, 8). CDK10 interacts with the ETS2 (v-ets erythroblastosis virus E26 oncogene homolog 2) transcription factor and inhibits its transcriptional activity through an unknown mechanism (9). CDK10 knockdown derepresses ETS2, which increases the expression of the c-Raf protein kinase, activates the MAPK pathway, and induces resistance of MCF7 cells to tamoxifen (6). ...

2.2. Mapping the real-world problem to an ML problem

2.2.1. Type of Machine Learning Problem

There are nine different classes a genetic mutation can be classified into = > Multi class classification problem

---

2.2.2. Performance Metric

Source: https://www.kaggle.com/c/msk-redefining-cancer-treatment#evaluation

Metric(s):

- Multi class log-loss
- Confusion matrix

2.2.3. Machine Learing Objectives and Constraints

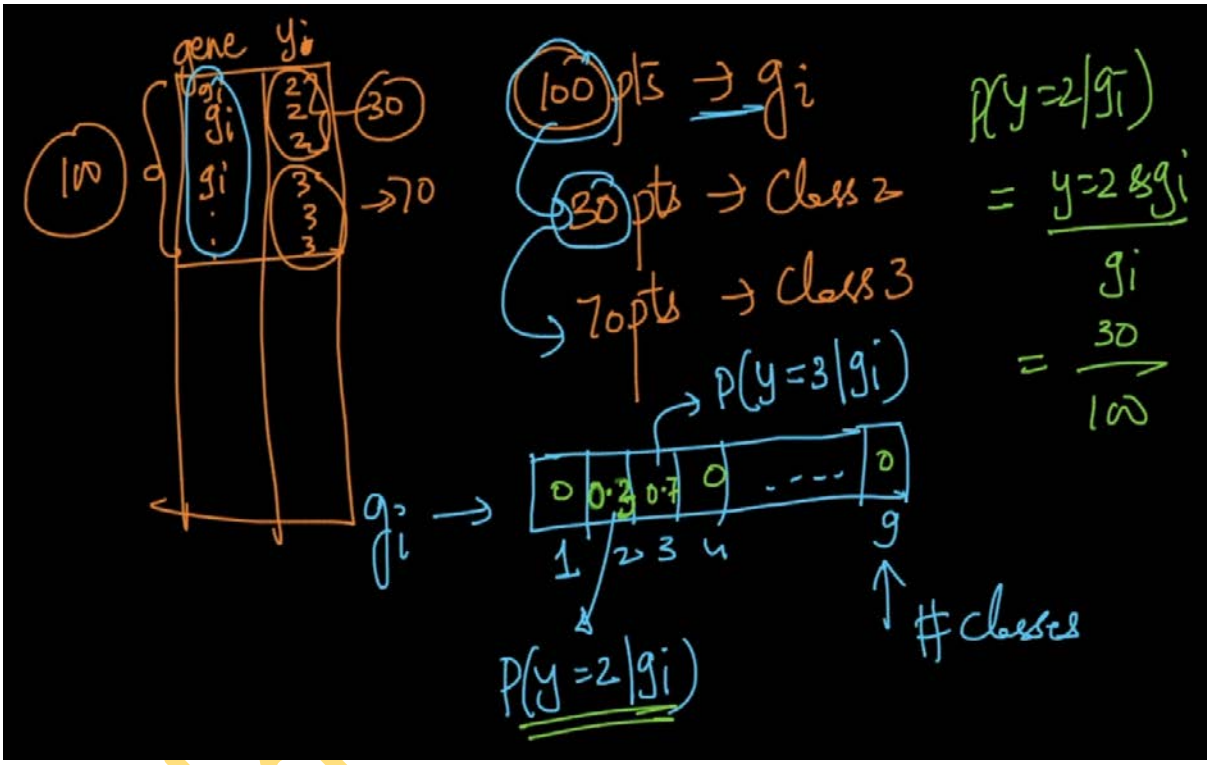
Objective: Predict the probability of each data-point belonging to each of the nine classes.
Constraints:
* Interpretability * Class probabilities are needed. * Penalize the errors in class probabilites => Metric is Log-loss. * No Latency constraints.

2.3. Train, CV and Test Datasets

Split the dataset randomly into three parts train, cross validation and test with 64%,16%, 20% of data respectively
WORKFLOW-

1. Genetic information in a cell is stored in the form of DNA. It is then transcribed to form RNA which is then translated to form proteins/amino acids. In case of a mutation, or a mistake in DNA sequence, the resultant amino acid is affected giving rise to a variation for the particular gene. Text is studied for this gene,variation pair.
1. The data consists of 2 csv files. One containing {id,gene,variant,class} and the other {id, text}. The text is a large corpus pertaining to the particular gene and variant from research literature. They are joined on basis of common id using pd.merge.
2. Train-cv-test-split is done randomly on 64-16-20.
3. The text_csv is not comma separated but separated by ||. This is taken care by using separator='/|/|' while loading. /=escape char.
4. NLTK preprocessing is done on text data. Special characters (including /n) are replaced by ' '. Lowercased letters. Multiple spaces are replaced by a single space. If words not in stopwords it is added to an empty string.
5. Train-test-split is done using stratify=y_labels to preserve class distribution in train,cv,test splits.
6. Bar plots to show distribution of classes in each split is plotted. They are similar since they were stratified.
7. The dataset is imbalanced.
8. 7,4,1,2 occur frequently.
9. Because our KPI is log-loss which varies till inf, we need to judge our model wrt a random model. In our case, random model generated log loss of 2.5.
10. The precision, recall and confusion matrices for the random model are plotted. Ideally, the off diagonal elements should be 0 and diag elements shld be 1.

11. Univariate analysis of categorical gene feature is done. There are 229 different genes and their histogram of no of distributions is plotted. It shows a skewed distribution. [sum-normalized no. of occurrences].
12. Using np.cumsum(sum), CDF is plotted.
13. We can one-hot encode or response code the gene features.



This is response coding. In one-hot-encoding, each feature is represented by an n-dim vector where n is the no. of unique values of the feature. In response coding, each feature is represented by n-dimensional vector where n is the no. of classes. The vector is filled with probability values (captured from the entire dataset) for the gene feature to belong to the one of the 9 classes.

This is followed by Laplace (additive) smoothing. It is done to ensure very small values are handled well.

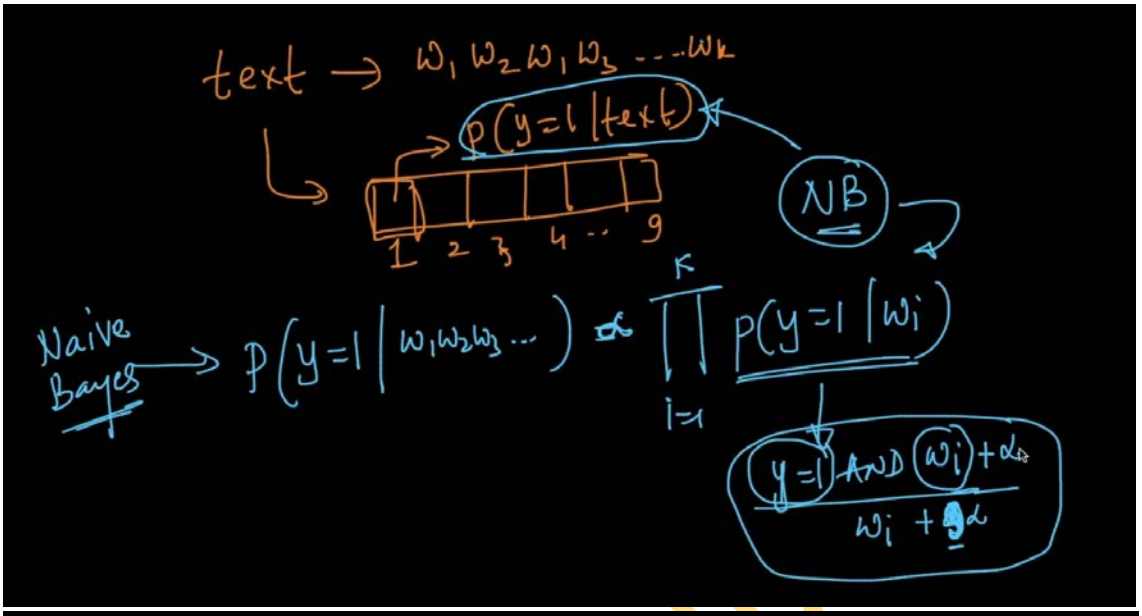
the first element = (number of times it occured in class1 + 10*alpha / nu mber of time it occurred in total data+90*alpha)

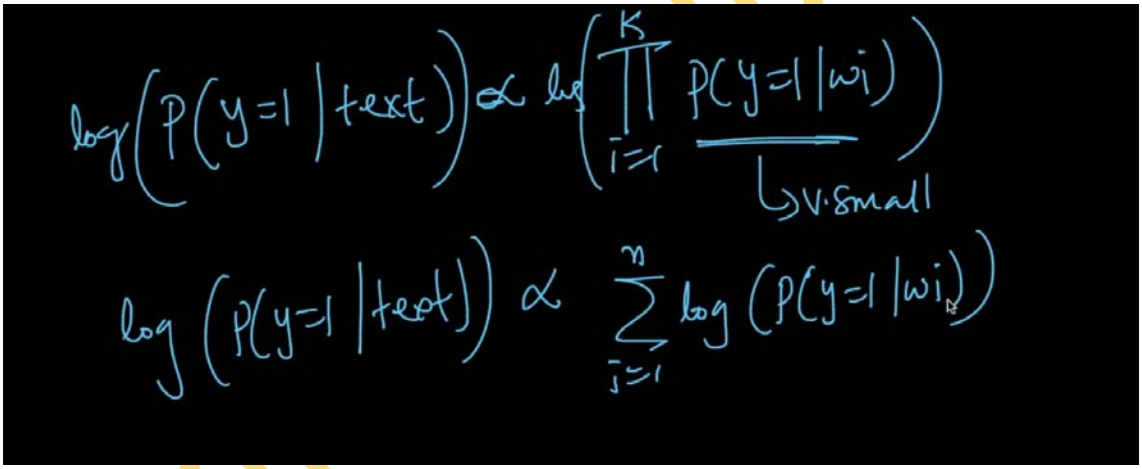
A simple code snippet get_gv_feature is written to get response coding.

Simple one-hot-encoding works well for Logistic Regression, Linear SVM while response coding works better for Naïve Bayes, Random Forest etc.

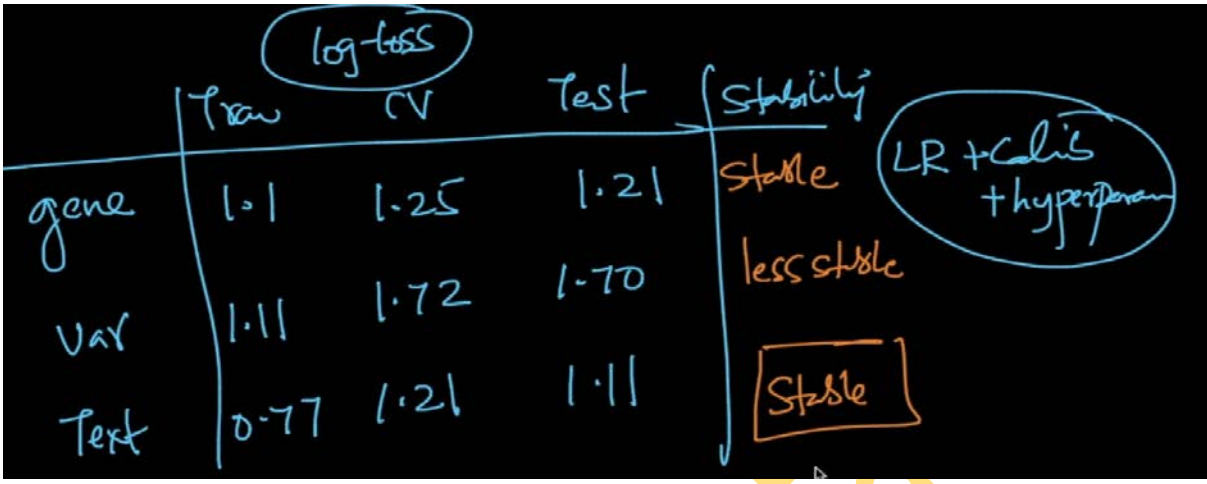
14. We need to infer how good the gene feature is. A model is built using only gene feature and checked whether it's log loss is less than 2.5.

15. First, an SGDClassifier (with 'log' loss and l2 regularization and h-param tuning) is used on one-hot encoded features. On top of it, a CalibratedCLassifier is used.
16. The best h-param is chosen and corresponding train,cv,test log loss is measured. It is better than a random model and is not overfitting.
17. Now, we have to check if the gene feature is stable. A stable feature is one whose values have significant overlapping between train, cv and test sets. If such is not possible, ML models are impossible.
18. The lack of overfitting shows feature stability. Also, we observe most points in cv and test data are present in train data too.
19. Above 90 percent of values in test and cv gene features are also present in set (train_df[gene]).
20. Next, the categorical feature variation is considered.
21. Histogram plotted for variation.value_counts shows extremely skewed data. Most
    values have 1 occurrence.
22. Cdf shows linear line. This infers that most value counts for features are 1.
23. Now, one -hot-encoding followed by LR+Calibrated CV is done. The log loss with best h-params is good. However, the model does overfit which arises from feature instability.
24. Only around 10% of variation points in test and cv df contained in set(train df[variation]). Thus, the variation feature, though useful, is not stable.
25. Now, the text features are analyzed. The same questions are asked.
    How many unique words are present in train data?
    How are word frequencies distributed?
    How to featurize text field?
    Is the text feature useful in predicitng y_i?
    Is the text feature stable across train, test and CV datasets?
26. Response coding for text is essentially Naïve Bayes.

27. 

28. 

29. A function is written for the same. Define a dictionary containing key as word and value as occurrence and another as (word, class) as key and occurrence as value.
30. BoW encoding for text is done.
31. On training LR+CalibratedCV, the training log-loss with optimal h-params is good. The model does not really overfit which shows that text feature is stable.

32. 

33. Thus our feature importance is text>gene>variation. This is deduced from the cv and test data log loss.
34. Above 95 percent of words present in test data was present in set(train data).
35. Next, we go for proper ML modelling.
36. The one-hot gene, variation, text features are horizontally stacked and converted to csr matrix. The response-coded gene, text, features are done same too.
37. One-hot-encoded features are 55517-dim while response-coded are just 27-dim. Thus one-hot features are good for Log Reg, LinearSVM models while response-coded are better for KNN, Boosting, Random Forest, Decision Trees etc.
38. The h-stacked features are fed into a Multinomial Naïve Bayes Classifier +Calibrated Classifier with h-param tuning with Grid Search.
39. Log loss was inferior to LR model just using text features. Moreover from precision and recall matrix, model gets confused between 2-7 and 1-4 classes which form majority of data points.
40. As per biz-requirements, for a particular query point from test data, both the predicted label as well as probability of classes was predicted.
41. In our case, label 7 was predicted and it was correct.
42. The features present in the query point out of the top 100 features for the prediction of class 7 are too printed for interpretability. (user-defined function)
43. Next, KNN with response coded features is used. The model performs better than NBClassifier. Confusion still occurs in the majority classes 2-7, 1-4. But performance on minority classes is better
44. For interpretability of KNN, using .neighbours, classes of next n of nearest neighbours can be found out. We can't find out feature importance for text (for eg) unlike NB.

45. Next, an SGDClassifier (loss='log' and class='balanced') is used with one-hot encoded features. The model is interpretable and gives the best accuracy as of now. From the metric matrices, we observe good results for low density classes 8,9 too. The classes were oversampled by the LR model. Thus balancing helps in better precision and recall for minority classes.
46. Next, an SGDClassifier(loss='hinge', class='balanced') is implemented. The accuracy is lower than LR model but other metrics are comparable for LinearSVM with LR. LinearSVM too is interpretable.
47. Random Forests with both one-hot-encoding and response-coding is trained with h-params=no. of estimators and max depth of tree. For this model, precision and recall for minority classes is 1 or near to 1. But majority classes are not that well done.
48. Random Forest with response coding overfits and has worst accuracy.
49. Using MLXtend , a Stacking Classifier was now tried. Base learners LR, LinearSVM and Naïve Bayes classifiers' probability outputs were passed to the final meta classifier LR Model.
50. Stacking faces issues of interpretability and the training data is split among the base models. Thus, errors for individual models are increased.
51. Thus, stacking does not give great value for less data points.
52. Next, a Majority Voting Classifier with voting='soft' (i.e taking the probability outputs of base models), was used.
53. Scopes for improvement-

1. Apply All the models with tf-idf features
2. Instead of using all the words in the dataset, use only the top 1000 words based of tf-idf values
3. Apply Logistic regression with CountVectorizer Features, including both unigrams and bigrams
4. Try feature engineering techniques to reduce the CV and test log-loss to a value less than 1.0