Machine Learning for Molecular Engineering

Problem Set 6

Date: May 7, 2021
Due: 10 pm ET on Thursday, May 21, 2021

Instructions This is the final problem set for the undergraduate version of this course (3.100, 10.402, 20.301). This exercise includes a machine learning competition hosted on Kaggle and a short exercise about unsupervised clustering. Here is the link to the competition. For submission, you need to submit a code notebook containing the code and a short writeup to describe your solutions. We have included some grading rubrics for your submission. You can find the template notebook here. You can also submit a separate notebook as the solution to the machine learning competition. As always, you are encouraged to ask for clarification on Piazza if you find any of the questions statements unclear or ambiguous.

Background

The immune system is a complex network of cells, tissues, and organs that is responsible for protecting us from disease by recognizing and destroying abnormal cells and pathogens in the body. Cancer cells are abnormal tissue cells that undergo many genetic changes that allow them to grow uncontrollably and become invasive. Because of how abnormal cancer cells are, many nascent cancers are identified by the immune system and destroyed before they fully develop, but why don't all cancers get caught by the immune system? Either the immune system is failing to recognize the specific changes the cancer cell has undergone, or the cancer cell is actively evading the immune system. Understanding the tumor- and immune-intrinsic features associated with cancers that progress despite treatment will help us develop better therapies.

In this problem set you will apply the techniques you've learned thus far to identify cancer immune sub-types and build models to predict progression-free survival with primary tumor data from The Cancer Genome Atlas (TCGA).

Registering for the competition

Go to kaggle.com to register for an account. After successful registration, click here to join the competition. Your can submit your solutions to the test set and your result will be displayed on the leaderboard. If you prefer anonymity, you can choose a nickname like `molmlmaster` rather than your real name. Note that you are not graded based on your ranking but a combination of different factors which will be described below. Our competition will have no cash prizes, but you can explore other competitions on Kaggle for potential cash prizes and employment opportunities.

Part 1: Characterizing Intra-Tumoral Immune States

To characterize immune states across cancers, you'll be analyzing the TCGA tumor samples which have been scored for their relative enrichment of 160 immune gene expression signatures (`ImmuneSignatures160.csv`). One way to identify these states is to cluster the immune gene expression signatures and find the groups of immune signatures that have shared associations. These clusters can subsequently be used to describe the cancers by these immune modules identified.

Part 1.1 (10 points) Visualization of Immune Modules

First download and load the data with the code we provided. Pearson's correlation is also known as the bivariate correlation, and measures the linear correlation between two sets of data. This calculation is simply the covariance of two variables divided by their standard deviations.

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

The `pandas` package has the `corr(method = 'pearson')` module for calculating Pearson's correlation of an existing DataFrame. In this case, we are calculating the linear correlation between the different immune signature scores. Immune signatures that are highly correlated with each other are likely to be involved in common processes corresponding to immune states.

Plot a heatmap of the pairwise Pearson correlation matrix of immune signature scores to visualize any shared associations. You can use `seaborn.clustermap` and we have provided the example code snippet for you. How many immune modules (clusters) do you observe in the heatmap? There will be ambiguities if you try to decide the number of clusters visually, so there are not wrong answers. In the next part, you will some statistical metric to decide the optimal number of clusters.

Part 1.2 (10 points)Clustering Analysis to Identify Modules

In this problem we will be using K-means clustering to identify the modules quantitatively. The module you can use `sklearn.cluster.KMeans` which requires the user to provide the expected number of clusters in the data. The input data you need to cluster are the Pearson's correlation coefficient computed in part 1.1. The K-means cluster algorithm will try to identify highly correlated immune signatures. You can read ref [1] for more detailed information.

After you performed clustering analysis for each immune signature, you can asses the quality of clustering by calculating the Silhouette Coefficient using the mean intra-cluster distance (a) and the mean nearest-cluster distance (b) for each sample. For a given sample, this is calculated as:

$$\frac{b-a}{max(a,b)}$$

Using Kmean alogrithm requires you to provide a pre-determined number of clusters K. You can identify the optimal number for K by computing the mean Silhouette Coefficient over all samples (Silhouette Score), we can identify the number of clusters in the data by identifying the value of K that corresponds with the maximum Silhouette Score ((see lecture 7)). We have provided code example for you to compute Silhouette score using `scikit-learn`.

Using K-means clustering, cluster the immune signatures to identify the modules with different number of K from 2 to 10. Determine the optimal number for K by plotting Silhouette Score as a function K and find K which corresponds to the maximum Silhouette score. What is the optimal K according to your analysis?

Part 2: Baseline Predict Cancer Progression

In the previous part, we identified clusters representing immune modules which were used to characterize the immune states of patients. With this immune state information, scientists are able to analyze the relationship between these immune states and response to treatment. In this problem, we will expand our analysis to include other tumor and immune features to identify what features are most predictive of treatment response.

Measuring progression-free survival (PFS) is one metric clinicians use to quantify how well a tumor responds to treatment. PFS is the length of time (e.g. days) after treatment that a patient lives without the cancer getting worse.

Download and load the data with the code we provided to you. In your dataframes, you will see columns like `'leukocyte'` `'tcr'` which are tumor and immune features that could be predictive of progression-free survival. There are 29 features in total. For a complete reference for each immune feature, see ref. [1]. You will train two baseline models to get started. You don't need to submit your predictions to Kaggle for this part.

Part 2.1 (10 points) Train a Logistic Regressor

Train a baseline logistic regressor to predict whether tumor samples will progress slowly(0) or quickly(1) based on the tumor and immune features provided (`covariates_train.csv`). Show your code and report the AUC-ROC results of a 5-fold cross validation (mean and standard deviation).

Part 2.2 (10 points) Train a Random Forest classifier

Train a Random Forest classifier with `max_depth=2` to predict whether tumor samples will progress slowly or quickly based on the tumor and immune features provided (`covariates_train.csv`). Show your code and report the AUC-ROC results of a 5-fold cross validation (mean and standard deviation).

Part 3: Machine Learning Competition and Report

In this part, you will apply technique you learned in this class to propose machine learning solutions to a classification problem. Based on the training data we provided to you, you will try to make predictions for the held-out test data. We provided the test feature set in `covariates_test.csv`. You will use your model to predict whether tumor samples will progress slowly(0) or quickly(1) using this test feature set. The evaluation of test performance will be performed by Kaggle, and you can see your results in a Leaderboard. The goal is find the best model possible. The teaching team will also participate in the competition. You can submit up to 20 solutions per day. We provided a utility function for you to generate a submission file.

For evaluation, the metric we use is the ROC-AUC score, ranging from 0 to 1. The score on Kaggle is calculated with only 40% percent of the test data, so your final performance might change during the final evaluation; this is to prevent you from tuning performance to the test data (which is bad practice!). We will release the final performance values when the competition is over.

You need to submit your commented code (Jupyter notebook) with some paragraphs that address the following points:

1. Data preprocessing
2. Model interpretation

   Describe the method you apply to quantify relative importance for each feature. For each of the method you applied, apply at least one method covered in lecture 11 to quantitatively analyze feature importance. For example, if you use a random forests regressor implemented in scikit-learn, you can obtain importance for each feature with the Gini importance equation.
3. Choice of model architecture

   You are expected to try at least two methods and select the best-performing model to submit the best solution. If you are designing a novel model in pytorch, describe your model architecture. For all your model, report cross-validation scores for the model. You need to provide brief description of your model choice or design and mention any open-sourced code/packages you used. if you decide to adopt a model architecture or method from a paper, please include the reference in your writeup.
4. Model evaluation and selection

   Describe how you performed hyperparameter search. Describe how the model performance is evaluated to select the best hyperparameter.

Grading Rubric

Here we describe how we will grade your notebook submission.

(15 points) Creativity

There are many modelling choices for classification questions like this. You can use logistic regressions and random forest classifiers from part 2. You can also apply support vector machines, gradient boosted trees and Neural Network regressors. You are encouraged to survey some literature to get some inspirations. In Ref [2], you can find a wide range of possible models which you can apply to this problem.

We also encourage you to apply pytorch in your solutions. With pytorch, you can build more complex models that is optimizable by gradient descent. For example, you can write an MLP architecture augmented with attention mechanisms which has been very popular these days. github.com is a nice resource to explore different creative solutions contributed by the ML community.

You can also try interesting methods to quantify feature importance. For example, you can apply feature masking (introduced in lecture 11) to quantify how much it influences your prediction performance.

(30 points) Technical correctness

We will examine your code and texts to evaluate your solution on technical correctness. please comment and document your code so that it is easier for us to understand each process from data preprocessing, train/validation/test split, hyperparameter optimization, and cross-validation.

(15 points) Model Performance (10 + 5 points)

We will provide a logistic regression baseline, if your model performs the same as the logistic baseline, you will automatically get half credit at least. We will grade based the prediction performance between your solutions and the solutions submitted by the teaching team. We may award extra points depending on how strong your performance is.

Part 4: Acknowledgement

We thank Ifrah Tariq for designing the exercise and preparing the data.

References

[1] Thorsson, V. *et al.* The immune landscape of cancer. *Immunity* **48**, 812–830 (2018).

[2] Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V. & Fotiadis, D. I. Machine learning applications in cancer prognosis and prediction. *Computational and structural biotechnology journal* **13**, 8–17 (2015).