Machine Learning for Molecular Engineering

Problem Set 6

Date: May 2, 2022
Due: 5 PM ET on Monday, May 9, 2022

Instructions This is the final problem set for the undergraduate version of this course (3.C01, 10.C01, 20.C01). This exercise includes a machine learning competition hosted on Kaggle and a short preliminary exercise about unsupervised clustering. Here is the link to the competition. For submission, you will need to submit a code notebook containing the code and a short writeup to describe your solutions. You can find the template notebook here. You can also submit a separate notebook as the solution to the machine learning competition. **Because this is the final pset, you will need to use your discretion in deciding how to tackle these problems; we have not specified exactly how each part should be approached.**

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 you register successfully, click here to join the competition. You 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; if you do this, please include your nickname in your final writeup. 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 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 (5 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.

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

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.

Use the code we have provided to plot a heatmap of the pairwise Pearson correlation matrix of immune signature scores to visualize any shared associations. We use `seaborn.clustermap` for the visualization. How many immune modules (clusters) do you observe in the heatmap? There are ambiguities if you try to decide the number of clusters visually, so there isn't a single right answer. In the next part, you will use a statistical metric to decide the optimal number of clusters.

Part 1.2 (5 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 clustering algorithm will try to identify highly correlated immune signatures. You can read [1] for more detailed information.

After you perform 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 the k-means 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)). You can use `sklearn.metrics.silhouette_score` to compute the Silhouette Score.

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 of k and find k which corresponds to the maximum Silhouette score. What is the optimal k according to your analysis?

Part 2: Baseline Prediction of 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'` and `‘tcr’` which are tumor and immune features that could be predictive of progression-free survival; the label for your data is in column `'label'`. There are 189 features in total. For a complete reference for each immune feature, see [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 to predict whether tumor samples will progress slowly or quickly based on the same features. Show your code and report the AUC-ROC results of a 5-fold cross validation (mean and standard deviation).

Part 3: (70 points) Machine Learning Competition and Report

In this part, you will apply techniques you have 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 have provided the test feature set in `covariates_test.csv`. You will use your model to predict whether tumor samples will progress slowly (1) or quickly (0) using this test feature set. The evaluation of test performance will be performed by Kaggle, and you can see your results in the leaderboard. The goal is find the best model possible.

You can submit up to 20 solutions per day. We have provided a utility function for you to generate a submission file. For evaluation, the metric we use is the AUC-ROC score, ranging from 0 to 1. The reported public score on Kaggle is calculated with only 50% percent of the test data, but your performance will be graded on the private score based on the other 50% of the test data; 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 (as a `.ipynb` notebook) with a writeup that addresses the following points:

1. Data preprocessing
2. Model interpretation

   Describe the method you apply to quantify relative importance for each feature. For at least one model you train, apply at least one method covered in lecture 11 to quantitatively analyze feature importance. For example, if you use a random forest 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 models 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 models, report cross-validation scores or some other rigorous metric of performance.[1] You need to provide a brief description of your model choice or design and mention any open-source 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.

(25 points) Creativity

There are many modeling choices for classification questions like this. You can use logistic regressions and random forest classifiers from part 2. You can also use support vector machines, gradient boosted trees and neural network regressors. You are encouraged to survey the literature for inspiration. In [2], you can find some possible model types to apply to this problem. **Warning:** You should not merely copy publically available code.

We 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.

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 each feature influences your prediction performance.

(35 points) Technical correctness

We will examine your code and text to evaluate your solution on technical correctness and the appropriateness of your chosen methods. Please comment and document your code so that we

[1]If you have a very good model but cross-validation is too compute-intensive, you can report a test set performance instead.

understand your approach to data preprocessing, train/validation/test split, hyperparameter optimization, and cross-validation. It is always important to compare your model performance to that of appropriate baseline methods. While we don't expect every idea you come up with to outperform the baseline, we do expect you to make this comparison and accurately report the resulting performance of your models.

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

We have provided a logistic regression baseline and 3 additional TA-prepared models as benchmarks. The logistic regression baseline is similar to something you might have done for Part 2.1. You can earn up to 10 points plus 5 bonus points on this part, depending on your performance as follows:

1. Beat or tie the logistic baseline - between 2 and 6 points.
2. Beat or tie 1 TA benchmark - between 6 and 10 points.
3. Beat or tie 2 TA benchmarks - 10 points, with up to 5 bonus points possible.
4. Beat or tie 3 TA benchmarks - 10 points plus 5 bonus points; potentially more bonus points if you impress us.

Warning: This dataset is rather small, so your ranking may change considerably between the public and private scores. Be very careful to make sure you don't overfit.

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).