Suki Zhu & Tien Li Shen

November 18, 2020

CS589 Machine learning

Mini Project Report

1) Scheme used to validate the model:

We're using train_ test_split from SKlearn to split our training data into a training and test set so we can get a rough idea of how well our model would do. We attempted twice the number of approaches to compensate for the required workload of 2 people. We're using Sklearn's OneVsRestClassifier to work with multilabel classification since only a few of Sklearn's built-in models work directly with multilabel data.

2) How class imbalance and sparse output labels are handled:

Originally, we set the class_weight parameter to be "balanced" to account for the class imbalance; however, we saw that we get better log loss results by not setting that parameter.

3) Describe any dimensionality reduction or feature selection we intend to do as a preprocessing step.

We used PCA to reduce the dimensions of the features before splitting our training data into test and training sets. We've reduced our features down to 50 for the logistic regression models and the ensemble models. For our deep learning models and linear models, we've reduced it to 100 features.

4) Linear model- describe any regularization, how hyperparameters are set and other details necessary. Report local test score.

We used OneVsRest with Ridge regression classifier. For the Ridge regression classifier, changing the alpha value did not result in significant change in the score. We tried 3 different solvers: Stochastic Average Gradient descent, conjugate gradient solve, and SVD. All of which produced similar scores. We did not report the probability log loss because Ridge did not have functions for producing prediction probability.

	Ridge(sag)	Ridge(svd)	Ridge(sparse_cg)
F1	0.2601	0.2614	0.2620
Accuracy	0.4665	0.4667	0.4671

Additionally, We used Logistic Regression as well as Stochastic Gradient Descent Classifier. Since our focus was to minimize probability log loss, we only considered models that could return prediction probability. We used Logistic Regression and Stochastic Gradient Descent classifier, we used the default l2 penalty. In order to calculate log loss, the loss parameter in SGDClassifier needed to be set to "log." We increased the max iteration to 2000 so that the model could converge and then slowly turned the alpha value for SGDClassifier and the C value for LR. We increased the regularization strength by tuning those values.

Sklearn's log loss function doesn't work well for multilabel classification so. In order to calculate log loss, we averaged the log loss of each column. For LR, with a max_iteration of 2000 and a C of 0.005, we got an average log loss of about 0.01645. For SGDClassifier, with a max_iteration of 2000, loss of "log", an alpha of 0.35, we got an average log loss of about 0.01978

	Logistic Regression	SGDClassifier
Average Log Loss	0.0164605	0.019782
Accuracy	0.4806853	0.449278
Hamming Loss	0.0028772	0.003061
F1_Score ("micro")	0.3073602	0.217264

Roc_Auc_Score	0.5923910	0.561481

When we submitted the Logistic Regression prediction we got a score of 0.11595.

The SGDClassifier got a score of 0.11612.

5) Ensemble model- what are used as base classifiers, how are they combined and what are the pertinent values for the hyperparameters. Report local test score.

We used a random forest classifier and AdaBoost. We got the best log loss for the random forest with a max depth of 3-- 0.01819. For Adaboost, we decided to use LR as our base estimator after seeing LR performed well with OneVsRestClassifier. However, we ended up with an average log loss of about 0.5317, which is worse than LR in OneVsRestClassifier. Doing some research, it was suggested that we try Gradient Boosting Classifier given that Logistic regression is a regression model. However, we noticed that as the number of estimators increases, the log loss increases as well so for our purposes, we decided not to use Gradient Boosting Classifier (we used alpha 0.35 which we found in the previous part). Because of that, we decided to just stick with Adaboost with a base estimator of Decision Tree with a depth of 5 and 75 estimators.

	Random Forest	Adaboost
Average Log Loss	0.018154	0.033798
Accuracy	0.444239	0.473463
Hamming Loss	0.003085	0.002938
F1_Score (‘micro”)	0.197625	0.29027
Roc_Auc_Score	0.554998	0.586949

6) Deep learning- what network architecture made the most sense here, try and combine deep learning with another model. Report local test score

I think that a neural network with fully connected layers makes the most sense for the given problem. However, we had unsolvable technical bugs implementing a Pytorch neural network for multi-label classification so we turned to SKlearn MLP. We attempted using MLP from SKlearn with layer sizes of (400, 350, 300, 250, 206), and (1280, 1024, 768, 576, 432, 206), validated by splitting training data into 80% training and 20% testing.

(1280, 1024, 768, 576, 432, 207)

log_loss: 5.59161211662517

f1 score: 0.34710743801652894

accuracy: 0.45811463363426413

(400, 350, 300, 250, 207)

log_loss: 10.898337815849029

f1 score: 0.3318914667715297

accuracy: 0.45496535796766746

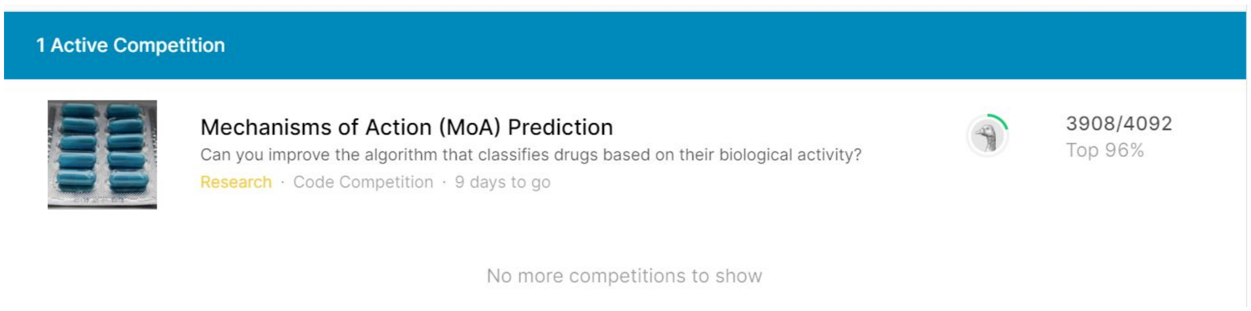
We got a score of about 0.13111.

notebook7bf75351cd
MLP (version 2/2)
24 minutes ago by Suki Z.99
From Notebook [notebook7bf75351cd]

Succeeded 0.13111

7) Submit solution to the challenge leaderboard and report score

We submitted our Logistic Regression model and with a score of 0.11595.


1 Active Competition

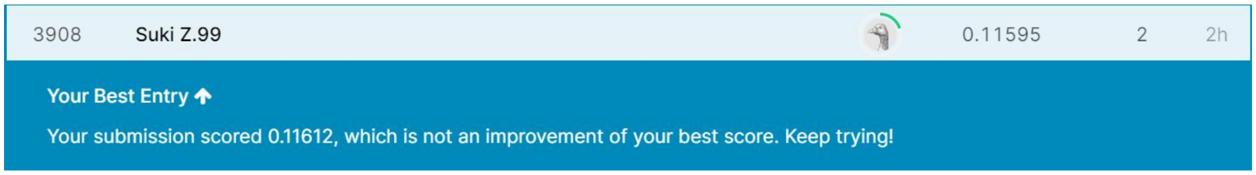
Mechanisms of Action (MoA) Prediction

Can you improve the algorithm that classifies drugs based on their biological activity?

Research · Code Competition · 9 days to go

3908/4092
Top 96%

No more competitions to show


3908 Suki Z.99 0.11595 2 2h

Your Best Entry

Your submission scored 0.11612, which is not an improvement of your best score. Keep trying!

(SGDClassifier had a worse score than LR)