The training data was originally split into train and validation data. 70% of the data were randomly sampled into the training set and the remaining 30% were sampled into the validation set with seed = 123.

For the first task, we were asked to create a model predicting whether something would be a static or dynamic activity. For the baseline model, I used a naïve Bayes prior probability approach of just predicting the most commonly seen outcome in the training data which was “static”. So, by using this strategy and guessing “static” for all of the data in the validation set, regardless of covariates, I had an accuracy of 0.584.

1. Complete the Task 1 and 2 and submit your classification results via Canvas.

2. Set up a Github repository and upload all your code used for training, evaluation, and

generating results of test data. Provide the url to your Github repository as the answer

to this question.

<https://github.com/allistho/Biostat626-Midterm>

3. Write a text file (name the file “README.md”) to provide the necessary instructions, so

that other people can reproduce all your results.

4. Describe your baseline algorithm and provide necessary tables and/or figures to summarize its performance based on the training data.

The training data was originally split into train and validation data. 70% of the data were randomly sampled into the training set and the remaining 30% were sampled into the validation set with seed = 123.

The baseline algorithm for the task 1 approach uses a general naïve bayes prior approach with the distribution of the training classes. This would predict the most commonly seen outcome in the training data which was “static”. So, by using this strategy and guessing “static” for all of the data in the validation set, regardless of covariates, I had an accuracy of 0.584. The precision and recall are shown below:

The baseline algorithm for task 2’s approach would also be a general naïve bayes approach as well, except since I was able to do task 1 and get 100% accuracy. Therefore, within each static and dynamic class, I would predict the most common type within each. For “static” the most common was “Lying” while for “dynamic” the most common was “Walking”.

5. Describe your final algorithm and provide necessary tables and/or figures to summarize

its performance based on the training data.

For the algorithm development process in task 1, I first used a regular generalized logistic regression model with no penalty term. However, since there were so many variables available to be trained and some variables were colinear with each other, a lot of noise was captured. To have the model only use the subset of variables instead which is most important, I used lasso regression with alpha = 1 and lambda = 0.0002. The model was tuned using 10-fold cross-validation to find the optimal lambda. This model was able to have an accuracy of 1, perfectly predicting the outcome being either static or dynamic in the validation set.

For the second task, since the model for task 1 was able to classify the results perfectly, I decided to first have model 1 predict whether the observation was going to be static or dynamic. Then I trained 2 separate models, a static predictor model, and a dynamic predictor model. Afterward, I fed data based on the output from model 1. The idea behind this was since a model could already accurately separate the static and dynamic classes, create models that can be more exact and identify the differences within the categories. This was then tuned within the training data using 10-fold cross-validation. The lambda for the dynamic model was 0.0007 while the lambda for the static model was 0.0003, This resulted in an accuracy of .984 in the validation set. Afterward, to submit the final prediction, I retrained the model with the final tuned model parameters using the full training data.

6. Use a figure or a table to show your leaderboard performance. Describe your efforts to

improve the performance.

As of 4/5, I was tied for the highest accuracy at 1 with SID 6139. 24 students at this time were also having an accuracy of 1. For multiclass classification, I had an accuracy of 0.955 in the test data. The model performance of my submission compared to other’s has been highlighted in yellow.

|  |  |
| --- | --- |
| Binary Classification Accuracy | Multiclass Classification Accuracy |
| 1.000 (24 times) | 0.978 |
| 0.999 | 0.977 |
| 0.999 | 0.970 |
| 0.999 | 0.962 |
| 0.999 | 0.961 |
| 0.999 | 0.961 |
| 0.999 | 0.960 |
| 0.999 | 0.959 |
| 0.999 | 0.958 |
| 0.999 | 0.958 |
| 0.998 | 0.956 |
| 0.990 | 0.955 |
| 0.987 | 0.955 |
| 0.969 | 0.955 |
| 0.949 | 0.954 |
| 0.892 | 0.952 |
| 0.561 | 0.951 |
| 0.561 | 0.950 |
| 0.065 | 0.948 |
|  | 0.948 |
|  | 0.948 |
|  | 0.942 |
|  | 0.938 |
|  | 0.936 |
|  | 0.935 |
|  | 0.933 |
|  | 0.933 |
|  | 0.931 |
|  | 0.931 |
|  | 0.931 |
|  | 0.930 |
|  | 0.928 |
|  | 0.926 |
|  | 0.919 |
|  | 0.896 |
|  | 0.883 |
|  | 0.827 |
|  | 0.807 |
|  | 0.116 |
|  | 0.044 |
|  | 0.000 |

7. Comment on your final results and potential ways to further improve the classification

accuracy.

The model had pretty high accuracy, and that was expected given everyone else’s model accuracy results. This leads to thinking that healthcare devices are extremely effective in tracking the movement of humans and can lead to health modeling and prediction outcomes. Other ways to improve the classification accuracy would be heavier levels of hyperparameter tuning in the multiclassification model. For example, instead of doing the extremes of lasso regression or ridge regression, tune the alpha parameter to intermediate values too. By doing 5-fold cross-validation, hyperparameter tuning is less dependent on the train-test splits as there are 5 sets of training and test sets. Depending on time with computational efficiency, more advanced deep learning or neural network models may be advantageous to get a higher accuracy as well. Another way to improve the cross-validation would be to make sure the train and validation splits have equal representation of outcome variables to ensure more balance.