**Pilot Experiments on Feature Importance  
Using the Rutgers COVID 197-Subject Pilot Database**

We have conducted some pilot experiments designed to help us understand the second version of the Rutgers COVID data. This dataset had 197 subjects. We chose 10 features based on our discussions in our last meeting. These features are shown to the right in Table 1.

|  |  |
| --- | --- |
| (1) AGE | (6) PROBLEMS\_1 |
| (2) RACE | (7) ICD10\_1 |
| (3) ETHNICITY | (8) MEDICATION\_1 |
| (4) LOS\_1 | (9) PROCEDURES\_1 |
| (5) DISCHARGE\_1 | (10) LABS\_1 |

Table 1. A listing of the features used in the pilot experiments for v1.0.0 of the Rutgers COVID Corpus

In this dataset, we were asked to predict the deceased status of a patient using a three-way decision:

* Class 0: not deceased
* Class 1: deceased at follow up
* Class 2: deceased at index admission

We sorted the data by these classes and produced the summary statistics shown in Table 2. To gain a basic understanding of the data, we split data into two sets: the training set and the evaluation set. At this stage, we considered 75 % of the data as training and 25% of the data as evaluation. We filled in missing values in categorical features with the name “unknown” and numerical features were given a value of 0. The results of this partitioning are shown in in Table 2.

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Train** | **Evaluation** | **Total** |
| 0 | 128 ( 87%) | 43 ( 88%) | 171 ( 87%) |
| 1 | 6 ( 4%) | 2 ( 4%) | 8 ( 4%) |
| 2 | 14 ( 9%) | 4 ( 8%) | 18 ( 9%) |
| TOTAL | 148 ( 75%) | 49 ( 25%) | 197 (100%) |

Table 2. A histogram of the pilot database is shown. A blind evaluation data set was created by partitioning the data using a 75% (train) / 25% (eval) split.

Of concern, of course, is the fact that the data is heavily biased towards class 0. Given the small number of tokens for the other two classes, there is not much insight we can gain from these experiments. Nevertheless, we can explore some preliminary results using some standard machine learning approaches to deal with small data sets.

Since most of our dataset is categorical, and the machine learning algorithm needs numerical data to do prediction, we used a hybrid encoding method based on frequency encoding to turn categorical data into numerical values. This is often called feature engineering. We rank-ordered simple categorical variables such as sex and ethnicity, by frequency of occurrence and then assigned integer indices to these values. For more complex features like ICD10\_1, we sorted them in lexical order because the terms shared common prefixes and assigned integers accordingly. There are a variety of methods used today for converting categorical variables such as one-hot encoding or label encoding. Good machine learning algorithms are usually somewhat insensitive to the nuances of these techniques. Pilot experiments on our data showed the encoding system did not make a significant difference and our approach seems to work as well as any.

Figure 1. Feature importance for the 10 selected features using the Bootstrap Forest algorithm (RF) in JMP.

We then used the statistical package JMP to generate a baseline analysis of our features. Using the Bootstrap Forest algorithm in JMP, which we refer to as RF, we trained a model on the whole data set described in Table 2, and then we used the feature importance analytic tool to determine how strongly each feature contributes to the overall prediction of a label in the evaluation set. The results of this analysis are shown in Figure 1. DISCHARGE\_1 is by far the most powerful feature. We used this feature importance result order features in our subsequent analyses.

It is interesting to note that the RF algorithm achieved a training error rate of 7% and an evaluation error rate of 4%. This is slightly above chance since if we always guessed class 0, our error rates would be 13% and 12% respectively. However, the data set is small so we cannot read too much into these results.

For our second set of experiments, we used a “leave-one-out cross-validation” method. In this method, we create an evaluation set consisting of only one token and use the remainder for training. We permute the training set so that each of the 197 tokens appears once in the evaluation set. Using this method, we have 197 sets of training and evaluation data, and in each set, we have one sample as the evaluation sample and 196 samples as training samples. After calculating the training and evaluation error rates for each partition, or fold, the overall error rate is the sum of the errors on each partition. Using this method, the training and evaluation set error rates were 1% and 11% respectively.

In Figure 2, we show a comparison of performance for RF, in terms of error rate, using the features ordered as shown in Figure 1. We evaluate both the training set (closed-set testing) and the evaluation set (open-set testing). The difference in the performance oon these two sets is often an indication of the limitations of the data. We see that the training error is decreasing for all features. However, the evaluation error is increasing after adding the second feature. This second feature is PROBLEMS\_1. As you see the training error is decreasing by adding each feature, while this is not the same for the evaluation set.

Figure 2. An analysis of RF performance as a function of the number of features ordered by importance

We also examined the confusion matrix to analyze the types of errors being made. This analysis is shown in Table 3. As you see in the confusion matrix, the algorithm could not recognize any samples from class 1 because we have just 8 samples of class 1. For class 2 we have 18 samples, of which 9 of them have been recognized correctly. Since the number of samples in class 2 is almost double the number of samples in class 1, and since half of the samples in class 2 have been recognized correctly, we can say if we had as many samples as we have in class 2, in class 1, the algorithm could recognize at least half of the samples in class 1 correctly. This is another indication that the data set is too small. The accuracy has been dominated by the class 0 data. If we want just to look at each class separately, the algorithm had an error rate of 3% on class 0, 100% on class 1, and 50% on class 2. As a result, we can say the overall accuracy in the evaluation was dominated by class 0 (we had 171 samples in class 0, of which 166 of them have been recognized correctly).

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **0** | **1** | **2** |
| **0** | 166 | 0 | 5 |
| **1** | 6 | 0 | 2 |
| **2** | 9 | 0 | 9 |

Table 3. A confusion matrix for the RF algorithm on the evaluation data set is shown.

**Recommendations:**

This report represents our second set of pilot experiments on the Rutgers COVID data. These experiments were designed to help us understand the new data and prepare us for an NIH proposal. Unfortunately, it is clear there is insufficient data.

Our recommendations at this point are:

1. Understand where Rutgers is headed with an expanded data set.
2. Sign the data sharing agreement.
3. Execute pilot experiments on the complete data set.
4. Regroup proposal writing once again because we don’t have really strong preliminary results yet.