Diabetes 130-US hospitals dataset

Weiyan Wu, Yao Jin, Jiwei Wang,
Lingyun Mao, Han Wu

ABSTRACT:
Our project is to predict whether a person will be readmitted within 30 days after being discharged from the hospital. We found the dataset from UCI machine learning lab. The data contains more than 50 features such as, race, gender, age, diabetic medications, etc. The prediction task is to determine, as accurately as possible, whether a person will be readmitted within 30 days. This was achieved by the following process:

1. Transformed feature values that contain string values into numeric value

2. Preprocessed data with handling the missing values and feature engineering

3. Solving the imbalanced data issue and randomly split the dataset into training data (70%) and test data (30%)

4. Implementing different classifiers to classify the test instances and then developed a classification algorithm to make the best prediction

PRELUDE:
The project involved developing and implementing a classification algorithm to determine whether a person will be readmitted within 30 days. The project is implemented in Python. A brief description of the process followed is below:

Data Preprocessing and transformation

1. Feature Transforming:
We use one hot to transform those features containing string values into numeric values.

2. Filling Missing Value:
When replacing the missing values, we use mode of records belonging to same class in dataset. This step is also performed on both training and test data.

3. Feature Engineering
In order to choose features that will give as good accuracy whilst requiring less data, we create new combination of attributes and remove unneeded, irrelevant or redundant attributes from data that do not contribute to the accuracy of our model.

4. Handling Imbalance Issue:
To handle imbalance in the dataset, we implement the data balancing technique, Synthetic Minority Oversampling Technique (SMOTE).

5. Rescaling Dataset

Some features in training and test dataset are continuous, while others are discrete. They are consist of different range, so we need to use scaling technique to solve the problem.

Building Classifiers

1. Classical

We applied different classical classification algorithms to generate classifiers, such as logistic regression, Neural Network, CNN and SVM

2. Ensemble

We also try to implement the ensemble methods like Xgboost and stacking to improve prediction accuracy.

1. DATA PREPROCESSING AND TRANSFORMATION

Data preprocessing step includes:

- Transforming string values into numerical values

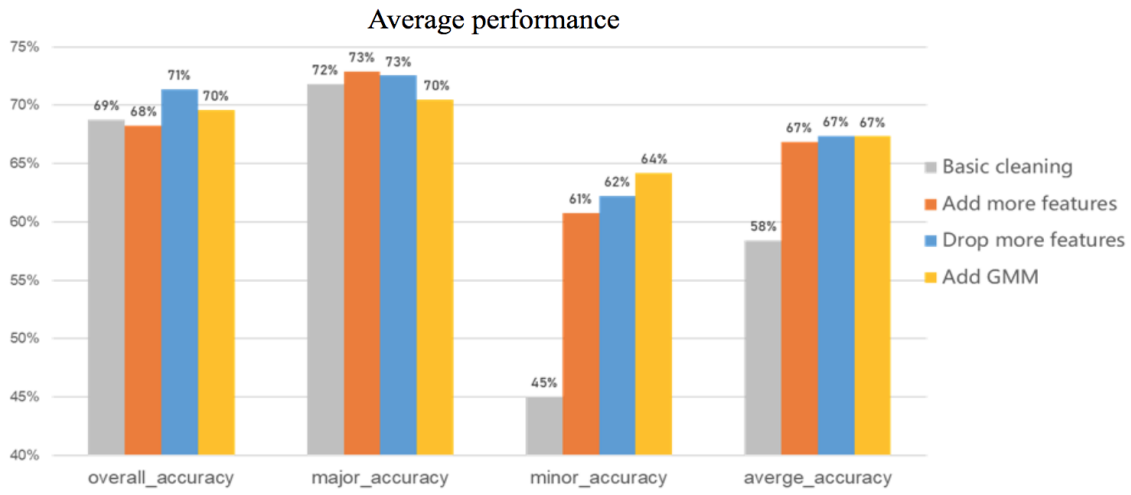
  In the dataset, features like race, diabetic medications, gender, and readmitted, etc contain string values, which should be converted into numerical values since models sometimes don't take string as input. After merging the training and test data, we implement encoding to transform categorical features to a format that works better with classification. We chose one hot encoding rather than label encoding because the classification algorithms we explored are distance based models and to avoid data frame mismatch.

- Handling the missing values

  Three methods have been used to handle instances with missing values. We first replace them with mean, and then delete all the instances that contain missing values. Eventually, we choose to impute the missing value using mode since it gives the best prediction and it will not be influenced by outliers. In order to replace all "?" in the dataset, we use the NaN, a numpy function, which is convenient to calculate the count and to replace with a number like mean or mode.

- Feature transformation and engineering

  We create some new features to the dataset: patient's reactions for 24 medications and patient visit frequency. And then we use GMM/EM model to find clusters as new features based on the original numeric features. In addition, we transform some features' value from range to bigger numeric number to make it meaningful. At the end, we drop those features are unneeded.



- Handling imbalanced dataset

  There are three common ways to handle imbalanced data:

  Simple oversampling the minority class:
  The easiest way to oversample is to re-sample the minority class, duplicate the entries, or manufacture data which is exactly the same as what we already have. The disadvantage to do this is that by making exact copies of existing examples, it makes overfitting likely.

  Simple undersampling the majority:
  One of the most common and simplest strategies is to undersample the majority class. It will simply select n samples at random from the majority class, where n is the number of samples for the minority class, and use them during training phase. The disadvantage with undersampling is that it discards potentially useful data.

  SMOTE:
  We choose SMOTE to implement imbalance data in this project. Synthetic Minority Over-sampling Technique(SMOTE) is an advance oversampling method. It adds some new and artificial minority samples by extrapolating between pre-existing minority instances rather than simply sampling with replacement. The newly created samples make the minority regions of the feature-space to be more substantial and more genera while less the risk of overfitting. The disadvantages of SMOTE is that it may result in new samples overlapping since every minority will involve in new sample generation.

- Data normalization

  Before running the data, it's important to convert all numeric indicators to a common scale to avoid introducing aggregation distortions stemming from

differences in numerals' means. We implement StandardScaler from sklearn to process data normalization.

2. BUILDING CLASSIFIER

(1) SVM

Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. Here we apply kernel trick to compute the dot product of two vectors in some (possibly very high dimensional) feature space. The kernel function we implemented in our model is LinearSVC (kernel='linear', C=1) , in order to improve the efficiency of computation.

Confusion Matrix		
	Truth	
Prediction	<30 Readmitted	>30 Readmitted
<30 Readmitted	2518	929
>30 Readmitted	9712	17371

Overall Accuracy : 65.4%
Average Accuracy: 68.3%

(2) Logistic Regression

The process of Logistic regression is to solve the problem of classifying the parameters of straight line (or hyperplane) according to the sample. The solving method is maximum likelihood estimation method, because we cannot get analytical solution from the final derivation of the likelihood equation, so the gradient descent method is used to optimize gradually to get the extreme value.

Confusion Matrix		
	Truth	
Prediction	<30 Readmitted	>30 Readmitted
<30 Readmitted	2476	971

>30 Readmitted	9444	17639

Overall Accuracy : 66%
Average Accuracy : 68.3%

(3) Neural Network

Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated. Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabeled data according to similarities among the example inputs, and they classify data when they have a labeled dataset to train on.

Confusion Matrix		
	Truth	
Prediction	<30 Readmitted	>30 Readmitted
<30 Readmitted	1943	1504
>30 Readmitted	7591	19492

Overall Accuracy : 72.3%
Average Accuracy : 63.6%

(4) CNN:

Convolution is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery. Convolutional networks were inspired by biological processes[4] in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

Confusion Matrix	
	Truth

Prediction	<30 Readmitted	>30 Readmitted
<30 Readmitted	1491	1976
>30 Readmitted	4987	22076

Overall Accuracy : 78.8%
Average Accuracy : 60.9%

3. ENSEMBLING LEARNING :

(1) XGBoost

XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

Confusion Matrix		
	Truth	
Prediction	<30 Readmitted	>30 Readmitted
<30 Readmitted	2197	1250
>30 Readmitted	6595	20488

Overall Accuracy : 73.5%
Average Accuracy : 71%

(2) Stacking

Stacking is a way of combining multiple models, that introduces the concept of a meta learner. It is less widely used than bagging and boosting. Unlike bagging and boosting, Stacking may be (and normally is) used to combine models of different types.

Confusion Matrix	
	Truth

Prediction	<30 Readmitted	>30 Readmitted
<30 Readmitted	2515	932
>30 Readmitted	7797	19286

Overall Accuracy : 71.4%
Average Accuracy : 72.1%

4. DATA EVALUATION

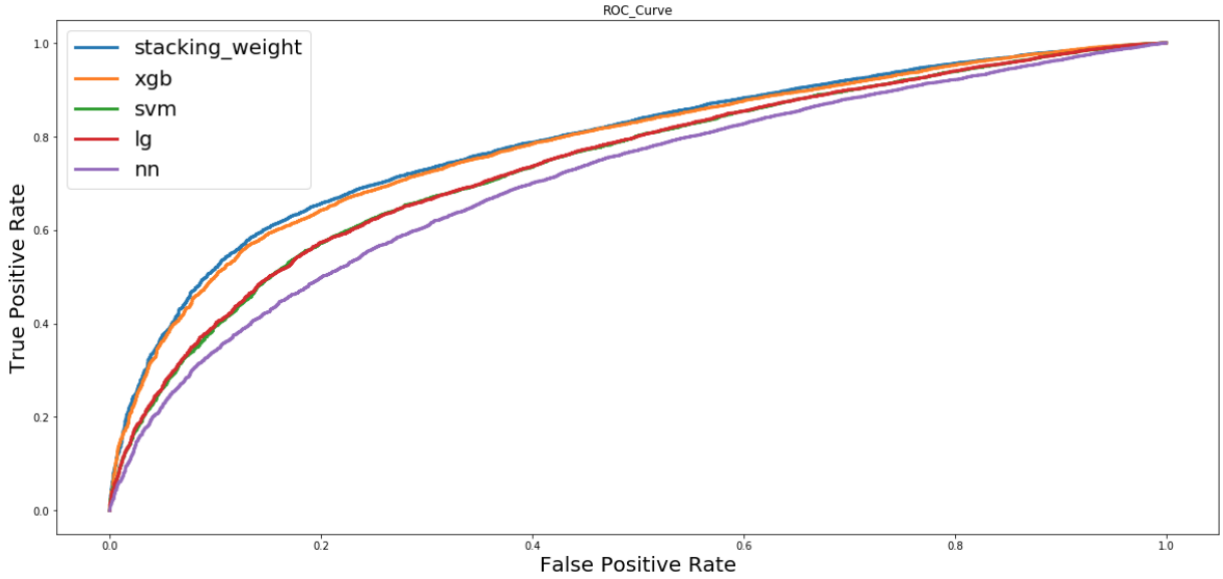
In this project, we applied multiple methods to evaluate our classifier's performance.

Classification Accuracy:
Accuracy is defined as the number of correct predictions from all predictions made.
However, classification accuracy alone is not adequate to predictive performance when:
• there is a large class skew
• there are differential misclassification costs

Confusion Matrix:
A confusion matrix is a summary of prediction results on a classification problem.
The number of correct and incorrect predictions are summarized with count values and broken down by each class. It gives insight not only into the errors being made by your classifier but more importantly the types of errors that are being made. It overcomes the limitation of using classification accuracy alone.

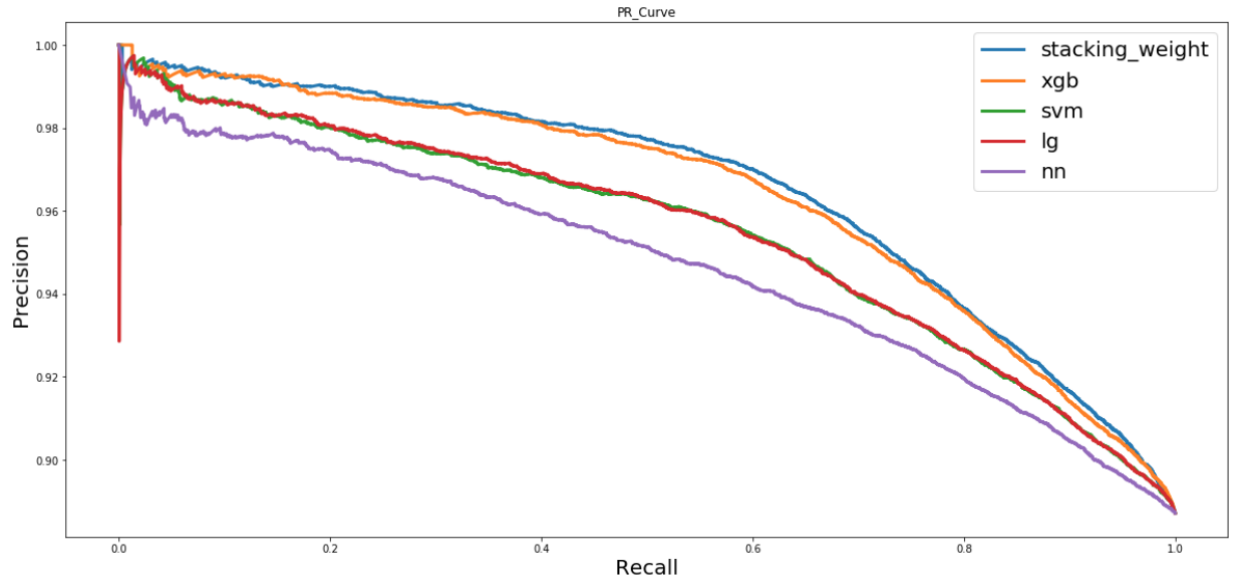
Model-wide evaluation measures:
The majority of machine learning models produce some kind of scores in addition to predicted labels. These scores can be discriminant values, posterior probabilities, and so on. Model-wide evaluation measures are calculated by moving threshold values across the scores.
Selecting a single threshold value results in determining predicted labels. For instance, a label is predicted as positive if the corresponding score is larger than a certain threshold value or predicted as negative, otherwise.
It is usually difficult to find an optimized threshold value. Hence, model-wide evaluation aims to evaluate the model of interest under various conditions by considering multiple threshold values.

•ROC:
The ROC curve is created by plotting the true positive rate(TPR) against the false positive rate(FPR) at various threshold settings. The true-positive rate is also known as sensitivity or recall. The false-positive rate is also known as the fall-out or probability of false alarm. The ROC curve is thus the sensitivity as a function of fall out. Another advantage of using the ROC plot is a single measure called the AUC (area under the ROC curve) score. As the name indicates, it is an area under the curve calculated in the ROC space.The theoretical range of AUC score is between 0 and 1, the actual scores of meaningful classifiers are greater than 0.5, which is the AUC score of a random classifier.



•PR-Curve:

The precision-recall curve is based on two basic evaluation measures – recall and precision. Recall is a performance measure of the whole positive part of a dataset, calculate by the number of positive predictions divided by the number of positive class values, whereas precision is a performance measure of positive predictions, calculated by the number of positive predictions divided by the total number of positive class values predicted. Similar to ROC curves, the AUC score can be used as a single performance measure for precision-recall curves. However, in this project, we use F1 score,which can be interpreted as a weighted average of the precision and recall, to evaluate the model, where an F1 score reaches its best value at 1 and worst score at 0. Unlike ROC, precision-recall curve is sensitive to imbalanced testing set.



CONCLUSION:

Among all classifiers, Stacking_weight classifiers performance overpass other type of classifiers. Ensemble algorithm classifiers performance better than classical algorithm classifiers.