Hospital Readmission Prediction

CSI5155 – Machine Learning

Project Fall 2020

Sai Krishna Yadlapalli
(Student ID – 300175048)
School of Electrical Engineering and
Computer Science University of Ottawa,
Ottawa, Canada
syadl053@uottawa.ca

Abstract— Hospital Readmission Prediction is a multi-class classification problem addressing the prediction of the hospital readmission using cartographic features. The main aim of the project is to construct predictive models and then compare and evaluate their respective classification accuracies. The following are the tasks and goal associated with this project. The primary task would be to ensure the quality of the data via data cleansing. Post cleansing, dimensionality reduction (if required), feature Engineering, and sklearn packages would be used to build the models. One-vs-the-rest (OvR) classifier, a multiclass strategy would be used to train and test the model and the metrics like accuracy, precision, recall, F1 score and ROC Curve for each case would be recorded. The data set will be used to build classification models using various feasible supervised learning algorithms like Naive Bayes, SVM, Random forests, KNN, Decision tree, Extra Tree and Bagging and semi-supervised learning algorithms like Label Propagation and Label Spreading. In addition, we build classification models using above supervised learning algorithms by choosing an target variable from the available set of features other than prescribed target variable to attain powerful insights of the data. The performance and evaluation of each algorithm will be recorded and visualized to graphically illustrate the qualitative and quantitative analyses performed as part of the project.

Keywords—classification, multiclass, supervised learning, semisupervised learning

Introduction

Data plays a significant role in today's world and the buzz words that we often hear in the technical discussions 'have a lot to do with data'. Millions of businesses thrive on the decisions that are made based on the predictions that have been arrived after quality data analysis. Various other critical fields like Medical, Governance, Armed Forces, Telecommunication, Entertainment industry, Education, E- commerce businesses also rely on data analysis results as a guidance for carrying out the various operations which include critical decision making. Therefore, it is not surprising if someone infers that data science is one of the most 'happening' fields in today's world.

Machine learning, as we know, plays a vital role in data science as it is used by data-scientists to build components (machine learning models) which help them to achieve a variety of tasks. As building these models require a huge data for it to learn from, it is very important that we know every detail about the data that we are using and how it would impact the learning model which we select to use on the data.

In this project, we will discuss about a problem which involves a data set about the hospital readmission. This dataset includes cartographic information on race, gender,age,admissiontype,dischargedisposition,admissionsource,timeinhospital, medicalspecialty,labprocedures,medications,diagnoses,maxgluserum,A1Cresult, insulin. The task is to predict whether a patient will be readmitted and if readmitted, is it before 30 days based on the various cartographic features. The task of predicting readmission using the cartographic data can be done by developing a machine learning model that uses an algorithm that learns from a set of data (supervised learning) and from a part of data (semi-supervised learning) and performs multi-class classification. The challenge is to identify the algorithm and training method that provides the optimum result. (which is, in our case, performance measures like accuracy, precision, recall, F1 score and ROC curve of the model). We will be using various data cleansing and pre- processing techniques along with evaluation techniques like test-train paradigm, One-vs-the-Rest methodology to create the optimum possible machine learning model.

Problem Domain Description

The data we used was an extract representing 10 years (1999–2008) of clinical care at 130 hospitals and integrated delivery networks throughout the United States: Midwest (18 hospitals), Northeast (58), South (28), and West (16). Most of the hospitals (78) have bed size between 100 and 499, 38 hospitals have bed

size less than 100, and bed size of 14 hospitals is greater than 500. The dataset consists of 101765 samples and 55 features. The target variable '**readmitted'** has three classes namely **"<30", ">30", "No"**. The other target variable we chose as a part of second task is '**gender'** which has classes **"Male", "Female"**. The dataset is a mix of numerical and categorical features.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101765 entries, 0 to 101764
Data columns (total 50 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   encounter_id              101765 non-null  int64
 1   patient_nbr               101765 non-null  int64
 2   race                      99492 non-null   object
 3   gender                    101765 non-null  object
 4   age                       101765 non-null  object
 5   weight                    3197 non-null    object
 6   admission_type_id         101765 non-null  int64
 7   discharge_disposition_id  101765 non-null  int64
 8   admission_source_id       101765 non-null  int64
 9   time_in_hospital          101765 non-null  int64
 10  payer_code                61510 non-null   object
 11  medical_specialty         51816 non-null   object
 12  num_lab_procedures        101765 non-null  int64
 13  num_procedures            101765 non-null  int64
 14  num_medications           101765 non-null  int64
 15  number_outpatient         101765 non-null  int64
 16  number_emergency          101765 non-null  int64
 17  number_inpatient          101765 non-null  int64
 18  diag_1                    101744 non-null  object
 19  diag_2                    101408 non-null  object
 20  diag_3                    100343 non-null  object
 21  number_diagnoses          101765 non-null  int64
 22  max_glu_serum             101765 non-null  object
 23  A1Cresult                 101765 non-null  object
 24  metformin                 101765 non-null  object
 25  repaglinide               101765 non-null  object
 26  nateglinide               101765 non-null  object
 27  chlorpropamide            101765 non-null  object
 28  glimepiride               101765 non-null  object
 29  acetohexamide             101765 non-null  object
 30  glipizide                 101765 non-null  object
 31  glyburide                 101765 non-null  object
 32  tolbutamide               101765 non-null  object
 33  pioglitazone              101765 non-null  object
 34  rosiglitazone             101765 non-null  object
 35  acarbose                  101765 non-null  object
 36  miglitol                  101765 non-null  object
 37  troglitazone              101765 non-null  object
 38  tolazamide                101765 non-null  object
 39  examide                   101765 non-null  object
 40  citoglipton               101765 non-null  object
 41  insulin                   101765 non-null  object
 42  glyburide-metformin       101765 non-null  object
 43  glipizide-metformin       101765 non-null  object
 44  glimepiride-pioglitazone  101765 non-null  object
 45  metformin-rosiglitazone   101765 non-null  object
 46  metformin-pioglitazone    101765 non-null  object
 47  change                    101765 non-null  object
 48  diabetesMed               101765 non-null  object
 49  readmitted                101765 non-null  object
dtypes: int64(13), object(37)
memory usage: 38.8+ MB
```

Fig 1: Dataset Feature Details

Data Analysis

The primary and important step would be understanding the data that we work on. This will provide a overview of the data set and helps to know how it can be used. We will need be clear with the information like the number and types of features, the type of the target feature, how the data is distributed within each feature and many more. This will help us to understand which machine learning model or algorithms and training techniques can be employed to solve our problem. We performed the below activities as part of the initial data analysis to get a better understanding of the data.

A. Scan for missing values

Missing values are sometimes the nightmares of data scientists. There are various strategies to treat missing values. Some of them are ignore the missing values, exclude any records containing missing values, replace missing values with the mean, or infer missing values from existing values, ignore the records with missing values and many more. Each technique has its own cons and pros. On analyzing our dataset, we could see that there are missing values in features weight, payercode, medicalspeciality, race, diag1. The features weight, payercode and medicalspeciality has missing values greater than 50 percent, so we dropped the weight and payercode feature and medicalspeciality is maintained because it has its influence on the target variable, so null values are replace with unknown class as we should not impute as it has percentage of missing values and remaining features with missing values are left as it is.

```
Out[7]: encounter_id                0
        patient_nbr                 0
        race                     1948
        gender                      0
        age                         0
        weight                  68664
        admission_type_id           0
        discharge_disposition_id    0
        admission_source_id         0
        time_in_hospital            0
        payer_code              31042
        medical_specialty       34477
        num_lab_procedures          0
        num_procedures              0
        num_medications             0
        number_outpatient           0
        number_emergency            0
        number_inpatient            0
        diag_1                     11
        diag_2                    293
        diag_3                   1224
        number_diagnoses            0
        max_glu_serum               0
        A1Cresult                   0
```

Fig 2: Missing values in Dataset

B. Replacing and Grouping

Information on values in each column of the dataset is very important for preprocessing. Our dataset consists of ID numbers for features like admissiontypeid, dischargedispositionid, admissionsourceid which are replaced with equivalent label and next we identified the unique values for each column. Categorical features admissiontypeid, dischargedispositionid, admissionsourceid and medicalspeciality are grouped into classes based on their percentage.

C. Correlation Analysis

Correlation analysis helps to identify the affinity between the features and how much extent a feature can help in deciding or predicting the class. If the correlation is positive, then the features are directly proportional to each other. If the correlation value is negative, then they are inversely proportional to each other. Two features are mutually exclusive and have no impact on each other when their correlation value is zero. Therefore, correlation is a way to understand the relationship between multiple variables and attributes in your dataset.

Fig 3: Correlation Matrix

Data Pre-Processing

Data pre-processing is the most important part of any data related project. The various pre-processing techniques helps us to transform the raw data set into a much better format which can be better processed and understood by learning algorithm. Data pre-processing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly impacts the ability of our model to learn; therefore, it is extremely important that we pre-process our data before feeding it into our model.

A. One hot encoding

One hot encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to perform better. Categorical data are sometimes difficult for the machine learning model to learn and understand. Hence, instead of providing a single feature of categorical type with 'n' number of categories, we could convert this into n features of binary data. This process of one hot encoding provides the categorical data in a better way for the machine learning model to understand and learn. There are about 36 categorical features and one-hot encoding is applied on these features.

B. Normalization

Normalization is a technique applied as part of data preparation for machine learning. The goal of normalization is to change the values of the numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. It is required only when features have different ranges, but in our case there is no need of normalization as all numerical features are in almost common scale.

C. Sampling

Class imbalance is a major issue that exists in most of the datasets. These imbalances datasets can be fixed to a certain extent by the application of sampling techniques. A dataset is imbalanced if at least one of the classes constitutes only a very few number of records which is called minority class. The issue of class imbalance can result in a serious bias towards the majority class, reducing the classification performance and increasing the number of false negatives. The most commonly used techniques are data resampling either under-sampling the majority class, or over-sampling the minority class, or a mix of both. In our case we have used balanced sampling as it produces better results when compared to rest two techniques. For Task 2 we haven't used any sampling because there is no much class imbalance.

```
Before BalancedSampling, counts of label '<30': 6277
Before BalancedSampling, counts of label '>30': 22222
Before BalancedSampling, counts of label 'NO': 41473

After BalancedSampling, counts of label '<30': 41039
After BalancedSampling, counts of label '>30': 39522
After BalancedSampling, counts of label 'NO': 39490
```

Fig 4: Task-1 Before and After Balanced Sampling

Model Construction

Once the pre-processing of the data is completed, the next step is to construct the machine learning models. We will choose an algorithm and a training technique based on the dataset and the problem that needs to be solved. As a part of this project, we ran the pre-processed dataset against models belonging to each type and obtained the results as below.

Out[47]:

	Algorithm	Accuracy	Precision	Recall	F1 score
0	DecisionTreeClassifier	0.628943	0.683401	0.628943	0.633684
1	BernoulliNB	0.574217	0.674026	0.574217	0.609818
2	KNNClassifier	0.641132	0.733875	0.641132	0.664724
3	LinearSVC	0.564832	0.837585	0.564832	0.672579
4	RandomForestClassifier	0.708741	0.747829	0.708741	0.717795
5	ExtraTreeClassifier	0.683696	0.711178	0.683696	0.690766
6	BaggingClassifier	0.717681	0.741656	0.717681	0.723221

Fig 5: Task-1 Classification Report

Out[16]:

	Algorithm	Accuracy	Precision	Recall	F1 score
0	DecisionTreeClassifier	0.560288	0.559974	0.560288	0.560090
1	BernoulliNB	0.561050	0.593529	0.561050	0.570150
2	KNNClassifier	0.532180	0.535120	0.532180	0.533277
3	LinearSVC	0.566671	0.708336	0.566671	0.604784
4	RandomForestClassifier	0.561765	0.568471	0.561765	0.563943
5	ExtraTreeClassifier	0.556715	0.565413	0.556715	0.559479
6	BaggingClassifier	0.559573	0.572234	0.559573	0.563422

Fig 6: Task-2 Classification Report

Supervised Learning:

A. Tree Based Models:

1) Decision Tree

A decision tree is used to classify future observations given a body of already labelled observations. The tree begins with a root then comes a series of branches whose intersections are called nodes and ends are called leaves, each corresponding to one of the classes to predict. The depth of the tree refers to the maximum number of nodes before reaching a leaf. Each node of the tree represents a rule.

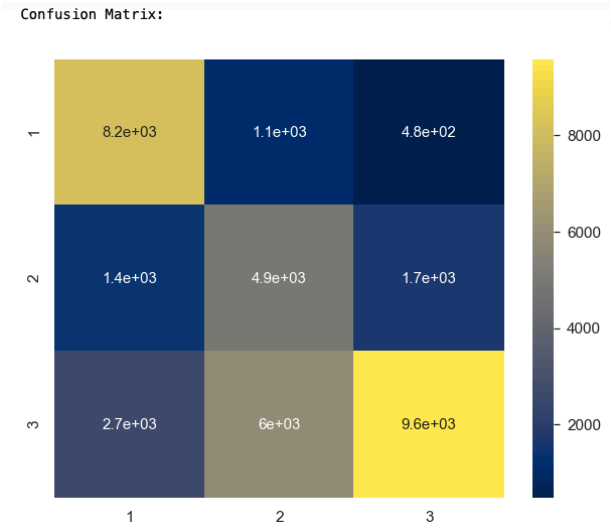


Fig 7: Task-1 Decision Tree

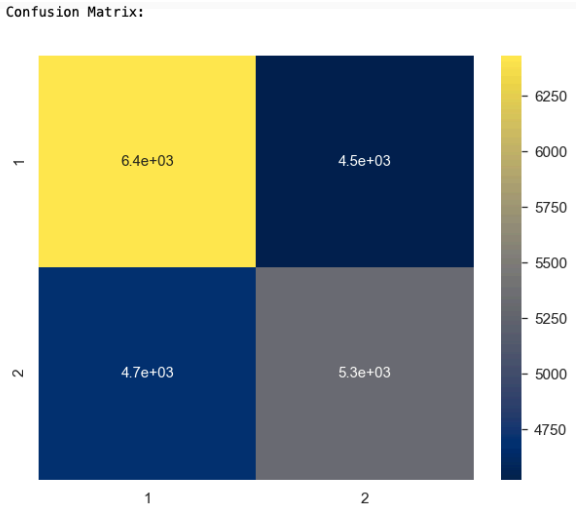


Fig 8: Task-2 Decision Tree

2) Random Forest

Random Forest is a flexible and easy to use machine learning algorithm that produces even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because of it's simplicity it can be used for both classification and regression tasks.

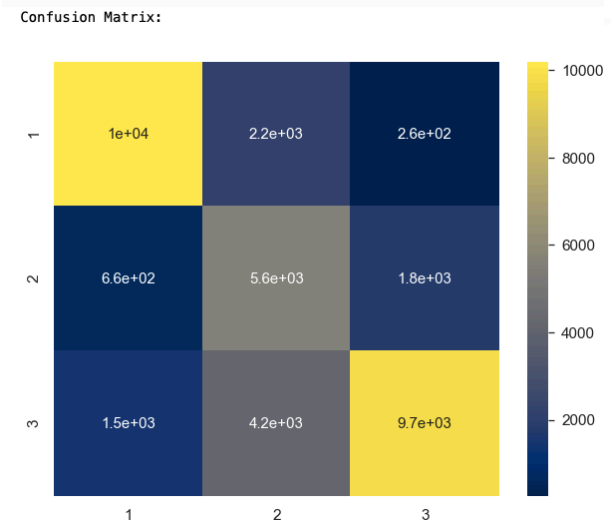


Fig 9: Task-1 Random Forest

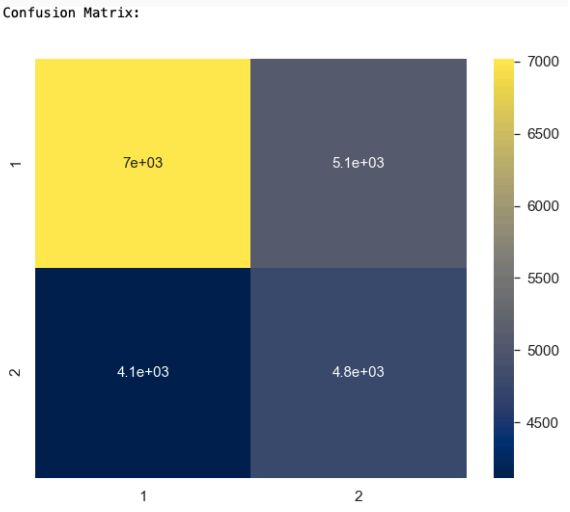


Fig 10: Task-2 Random Forest

B. Bayesian Model:

1) Bernoulli Naive Bayes

A Bernoulli Naive Bayes algorithm is a special type of NB algorithm. It's specifically used when the features have continuous values. The naive Bayes

classifier assumes all the features are independent to each other. Even if the features depend on each other or upon the existence of the other features.

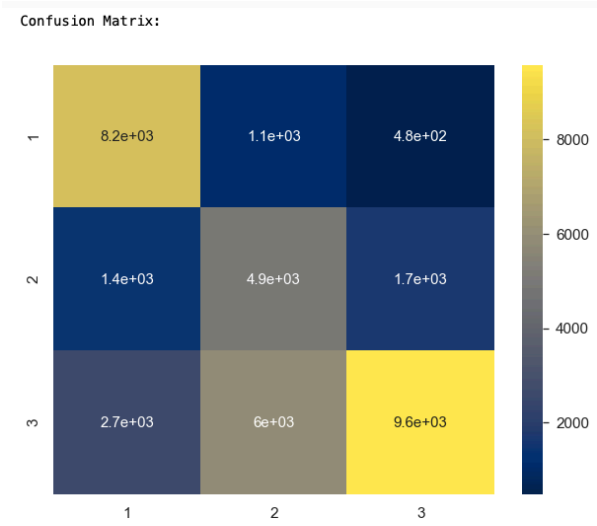


Fig 11: Task-1 Bernoulli NB

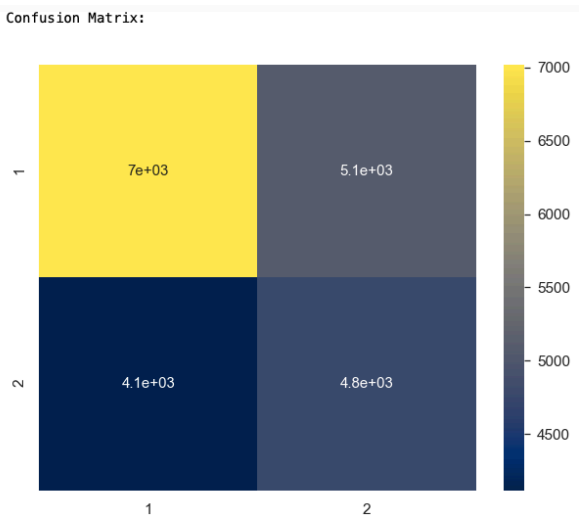


Fig 12: Task-2 Bernoulli NB

C. Distance Based Method:

1) KNN (K- Nearest Neighbors)

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. The KNN algorithm assumes that similar things exist in the close proximity. In other words, similar things are near to each other. To select the ‘k’ value, we could use the ‘Grid search’ technique which is used to find an optimum hyper-parameter valued of an algorithm for the data set which is under consideration.

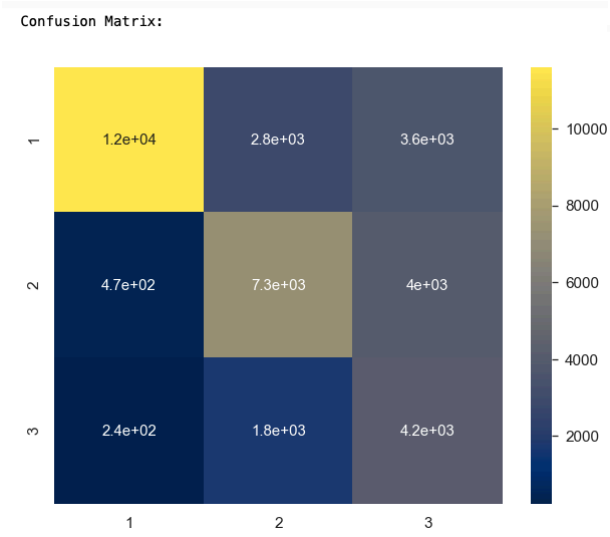


Fig 13: Task-1 KNN

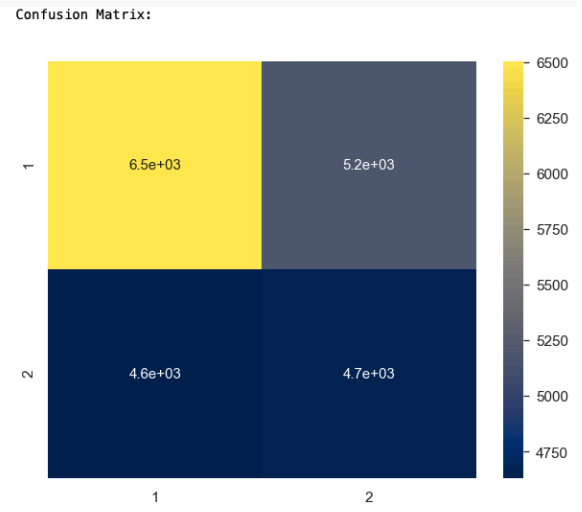


Fig 14: Task-2 KNN

D. Linear Model:

1) Linear SVC (Support Vector Classifier)

Linear SVC is the most efficient flavour of SVM for classification problems. The objective of a Linear SVC (Support Vector Classifier) is to fit to the data you provide, returning the "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is.

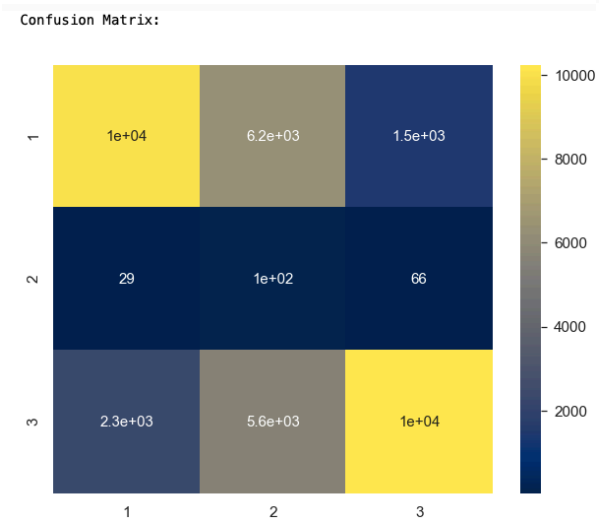


Fig 15: Task-1 SVM

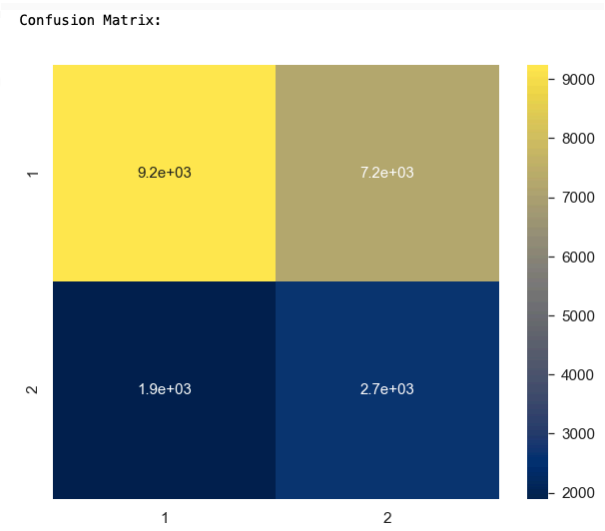


Fig 16: Task-2 SVM

E. Ensembles:

1) Bagging

Bagging is a shorthand for the combination of bootstrapping and aggregating. Bootstrapping is a method to decrease the variance of the classifier and reduce overfitting, by resampling data from the training set with the same cardinality as the original set.

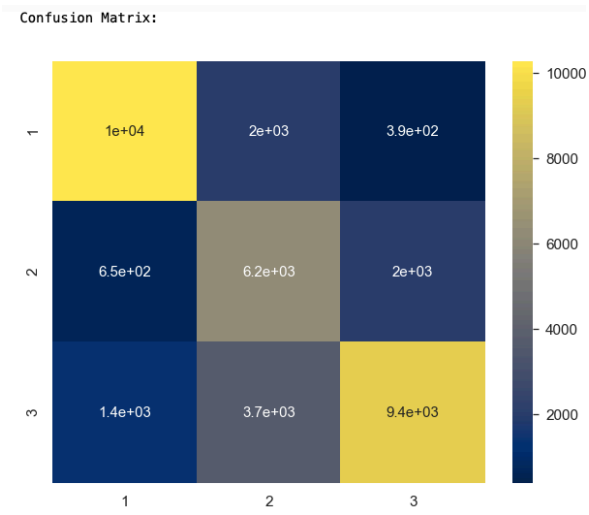


Fig 17: Task-1 Bagging

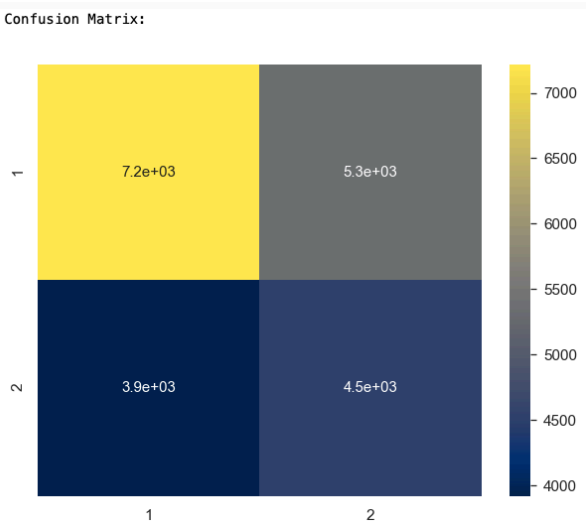


Fig 18: Task-2 Bagging

Semi-Supervised Learning:

1)Label Propagation:

Label propagation is a semi-supervised machine learning algorithm which assigns labels to the previously unlabelled data points. At the start of the algorithm, a subset of data points have labels. These labels are propagated to the unlabelled points throughout the course of algorithm

Classifcation report

	precision	recall	f1-score	support
0	0.11	0.05	0.07	384
1	0.22	0.34	0.27	1323
2	0.78	0.70	0.74	5291
accuracy			0.60	6998
macro avg	0.37	0.36	0.36	6998
weighted avg	0.63	0.60	0.61	6998

Confusion matrix
[[18 126 240]
[44 451 828]
[103 1471 3717]]

Classifcation report

	precision	recall	f1-score	support
0	0.08	0.18	0.11	935
1	0.25	0.28	0.26	3187
2	0.72	0.60	0.66	9873
accuracy			0.50	13995
macro avg	0.35	0.35	0.34	13995
weighted avg	0.57	0.50	0.53	13995

Confusion matrix
[[167 257 511]
[520 906 1761]
[1439 2515 5919]]

Classifcation report

	precision	recall	f1-score	support
0	0.08	0.97	0.15	2799
1	0.30	0.00	0.01	10669
2	0.70	0.03	0.06	21518
accuracy			0.10	34986
macro avg	0.36	0.34	0.07	34986
weighted avg	0.53	0.10	0.05	34986

Confusion matrix
[[2719 15 65]
[10390 53 226]
[20721 108 689]]

Classifcation report

	precision	recall	f1-score	support
0	0.09	1.00	0.16	5600
1	0.00	0.00	0.00	19598
2	0.84	0.00	0.00	37777
accuracy			0.09	62975
macro avg	0.31	0.33	0.06	62975
weighted avg	0.51	0.09	0.02	62975

Confusion matrix
[[5599 0 1]
[19591 0 7]
[37734 1 42]]

Classifcation report

	precision	recall	f1-score	support
0	0.09	1.00	0.16	5937
1	0.00	0.00	0.00	20855
2	0.71	0.00	0.00	39682
accuracy			0.09	66474
macro avg	0.27	0.33	0.05	66474
weighted avg	0.43	0.09	0.01	66474

Confusion matrix
[[5937 0 0]
[20853 0 2]
[39677 0 5]]

Fig 19: Label Propagation Classification Reports for different percentages of unlabelled data

2)Label Spreading:

The label spreading algorithm works by representing a document as a point in space, and then finding all the other points that are closest to it.

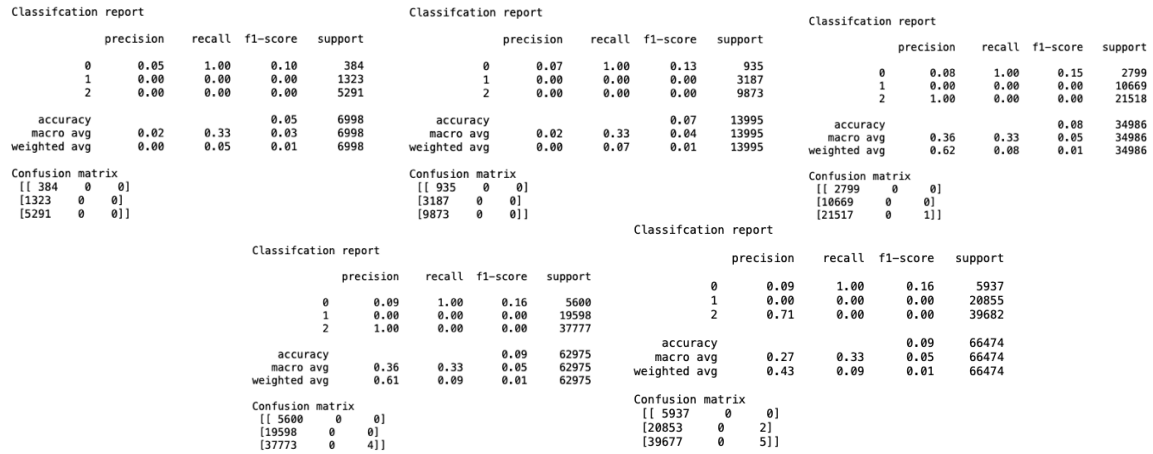


Fig 20: Label Spreading Classification Reports for different percentages of unlabelled data

Evaluation Techniques:

A. Test-train Split:

The concept of the test train split is of the techniques that is used in machine learning process to partition the data set into train and test data. This can easily be done by passing the pandas data frame to the train_test_split() function or library of scikit learn. We would need to provide the ratio in which the data set needs to be split. As a thumb rule or most commonly followed practice, we have used the 70:30 as the ratio for splitting the dataset into train and test datasets respectively.

B. Performance Measures:

For all the models or machine learning algorithms we have calculated and recorded various performance metrics like Accuracy, Recall, Precision, F1 Score. This would greatly help us to compare various models based the numbers obtained. We also calculated the Confusion Matrix or Contigency Table for all the models. This helps to understand the no. of instances that are correctly classified and the number of ones that are not. If a model has a heavy leading diagonal, then it means that the accuracy is high. These are the various performance measures that we used to evaluate the best model for our machine learning problem.

Out[47]:

	Algorithm	Accuracy	Precision	Recall	F1 score
0	DecisionTreeClassifier	0.628943	0.683401	0.628943	0.633684
1	BernoulliNB	0.574217	0.674026	0.574217	0.609818
2	KNNClassifier	0.641132	0.733875	0.641132	0.664724
3	LinearSVC	0.564832	0.837585	0.564832	0.672579
4	RandomForestClassifier	0.708741	0.747829	0.708741	0.717795
5	ExtraTreeClassifier	0.683696	0.711178	0.683696	0.690766
6	BaggingClassifier	0.717681	0.741656	0.717681	0.723221

Fig 21: Task-1 Classification Report

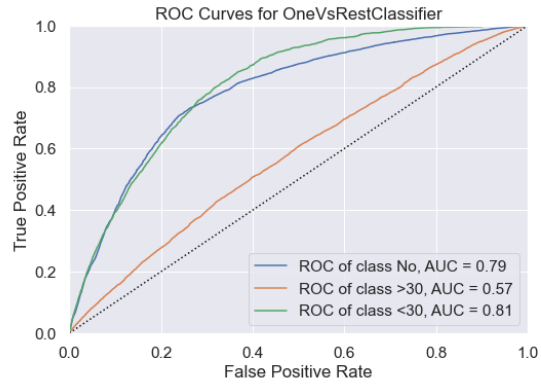
Out[16]:

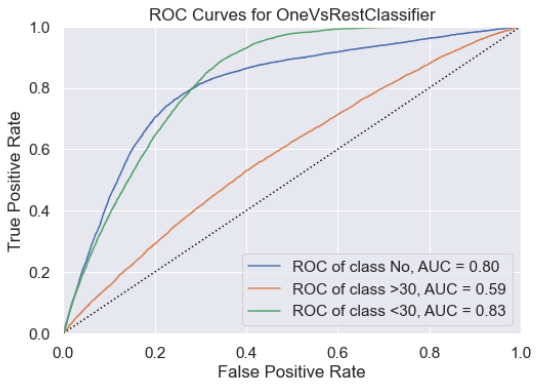
	Algorithm	Accuracy	Precision	Recall	F1 score
0	DecisionTreeClassifier	0.560288	0.559974	0.560288	0.560090
1	BernoulliNB	0.561050	0.593529	0.561050	0.570150
2	KNNClassifier	0.532180	0.535120	0.532180	0.533277
3	LinearSVC	0.566671	0.708336	0.566671	0.604784
4	RandomForestClassifier	0.561765	0.568471	0.561765	0.563943
5	ExtraTreeClassifier	0.556715	0.565413	0.556715	0.559479
6	BaggingClassifier	0.559573	0.572234	0.559573	0.563422

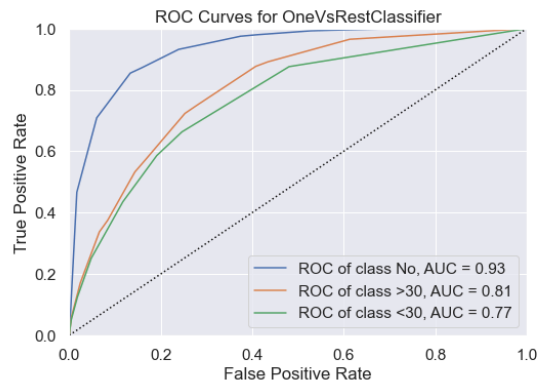
Fig 22: Task-2 Classification Report

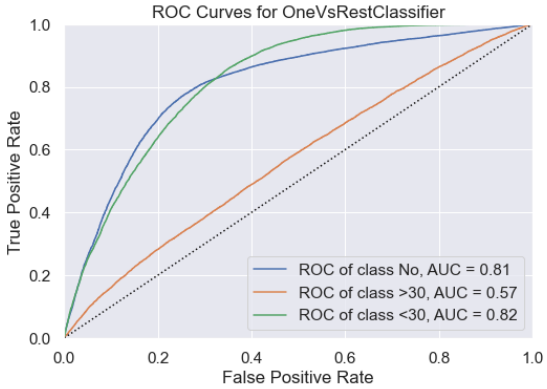
C. ROC Curve:

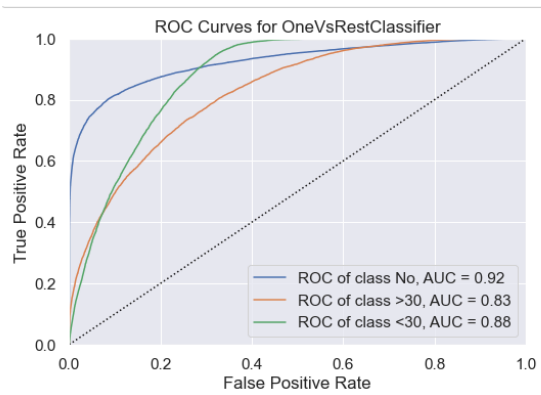
An ROC curve (receiver operating characteristic curve) is a graph showing performance measure of a classification model at all the classification thresholds. This curve plots two parameters true positive rate, false positive rate.

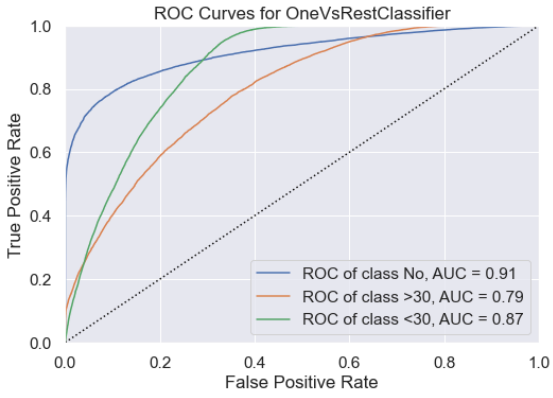












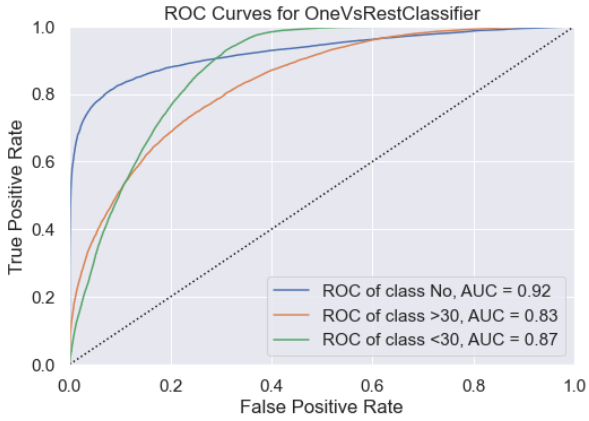


Fig 23: ROC Curves for different algorithms for Task-1

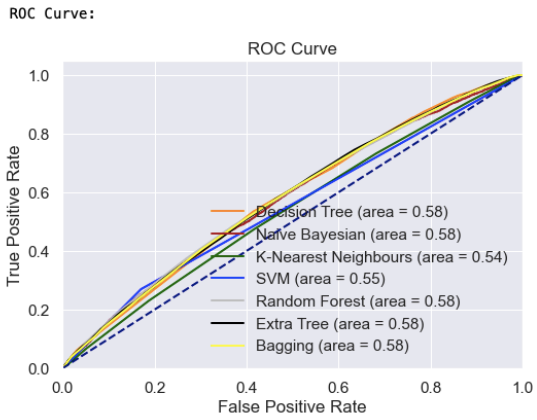


Fig 23: ROC Curves for different algorithms for Task-2

D. T-Test Evaluation:

```
Paired-t-test:
RandomForest-BaggingClassifier: Ttest_relResult(statistic=6.588961254883599, pvalue=0.00010054530079059334)
```

The Random Forest and Bagging Classifier are Statistically Significant because the pair has p value below 0.05.

Fig 24: Paired T Test for Random Forest and Bagging Classifier for Task-1

```
Paired-t-test:
NaiveBayesian-RandomForest: Ttest_relResult(statistic=7.422487115643609, pvalue=4.006882219865603e-05)
```

The Naive Bayesian and Random Forest are not Statistically Significant because the pair has p value above 0.05.

Fig 25: Paired T Test for Random Forest and Bagging Classifier for Task-1

Observations, Insights, and Lessons Learnt

1) Bagging(Decision Tree)Classifier has obtained highest accuracy for Task-1

The Bagging classifier with decision tree has obtained highest accuracy of (0.717) when compared to the remaining classification algorithms for the Task-1.

2) LinearSVC has obtained highest accuracy for Task-2

The LinearSVC classifier has obtained highest accuracy of (0.566) when compared to the remaining classification algorithms for the Task-2.

3) Bagging (Decision Tree) ensemble model's accuracy was lower than that of stand-alone Decision Tree model

The accuracy which we obtained for bagging ensemble (0.559) was slightly lower than that of the stand-alone Decision Tree model (0.560). Therefore, ensembling couldn't increase the accuracy of the model in this case.

4)Gaussian NB classifier works terrible for both Task-1 and Task-2

The accuracy obtained by Gaussian NB classifier was far less than the variants of Bayesian Classifiers like Bernoulli NB and Multinomial NB in both the tasks.

5) Sampling does not always increase the accuracy

In semi-supervised learning the accuracies for models before sampling is higher when compared to the accuracies after sampling.

6) Particle Competition and Combination algorithm (semi-supervised learning algorithm) issue

The Particle Competition and Combination a semi-supervised learning algorithm, when executed, is forcing the kernel to stop (kernel dead) and restart it.

Conclusion

Finally, after all the analysis, experimental evaluation and statistical testing, we can say that Bagging(Decision Tree) Classifier provided the best performance amongst the chosen set of Machine learning algorithms for the Task-1, LinearSVC provided the best performance amongst the chosen set of Machine learning algorithms for the Task-2 and there is no much significant difference in performance of semi-supervised learning algorithms for Readmission Prediction dataset.

Future work

As a future part of this analysis, we want to improve the performance of the supervised learning algorithms incorporating feature engineering techniques and also improve the performance of semi-supervised learning algorithms for high percentage of unlabelled data. There is also a scope for investigating on algorithms used in semi-supervised learning.

References

[1] https://www.hindawi.com/journals/bmri/2014/781670/

[2] https://archive.ics.uci.edu/ml/machine-learning-databases/00296/

[3] https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.BaggingClassifier.html

[4] https://minds.wisconsin.edu/bitstream/handle/1793/60444/TR1530.pdf?sequence=1

[5] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

[6] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[7] https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets

[8] https://seaborn.pydata.org/generated/seaborn.heatmap.html

[9] https://seaborn.pydata.org/examples/many_pairwise_correlations.html

[10] https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html

[11] https://www.scikit-yb.org/en/latest/api/classifier/rocauc.html

[12] https://scikit- learn.org/stable/modules/label_propagation.html#