Predicting Stay of Patients in Hospital using Naïve Bayes Classifier

Utkarsh Patel
18EC30048

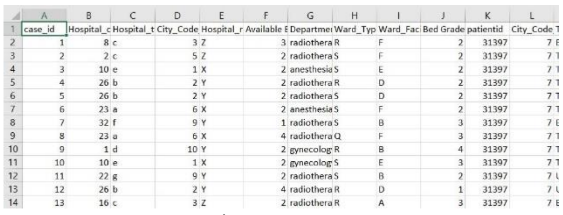
Akhil
18CS10070

Abstract

This report is submitted as part of second assignment to the course CS60050 – Machine Learning, IIT Kharagpur. The dataset contained patients records across various hospitals spread across various cities. The objective is to predict the stay of the patients in the hospitals using these records. We used Naïve Bayes classifiers, one with simple kernel and other with *Gaussian* kernel, to perform the tasks. Dimensionality reduction techniques like Principal Components Analysis (PCA) and Backward-Selection method are also employed. However, we concluded that Naïve Bayes is not the optimal algorithm to perform tasks in this domain as observed accuracy were quite low. This paper briefly explains the procedure we followed to achieve the goal and the results obtained.

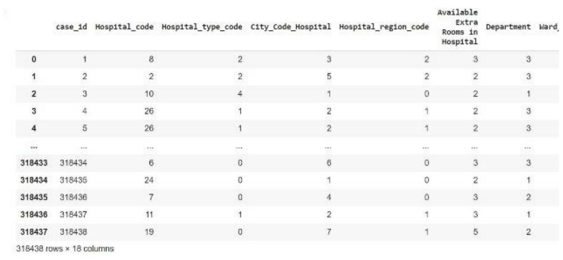
1 Accessing Dataset

The dataset is provided to us as a `CSV` file. We accessed the data using `pandas.read_csv` API and created a `pandas.DataFrame` object for the dataset handling. This is done because it is easy to handle and manipulate data using such objects.



	A	B	C	D	E	F	G	H	I	J	K	L
1	case_id	Hospital_c	Hospital_t	City_Code	Hospital_r	Available E	Departmei	Ward_Typ	Ward_Fac	Bed Grade	patientid	City_Code
2	1	8	c	3	Z	3	radiothera	R	F	2	31397	7
3	2	2	c	5	Z	2	radiothera	S	F	2	31397	7
4	3	10	e	1	X	2	anesthesia	S	E	2	31397	7
5	4	26	b	2	Y	2	radiothera	R	D	2	31397	7
6	5	26	b	2	Y	2	radiothera	S	D	2	31397	7
7	6	23	a	6	X	2	anesthesia	S	F	2	31397	7
8	7	32	f	9	Y	1	radiothera	S	B	3	31397	7
9	8	23	a	6	X	4	radiothera	Q	F	3	31397	7
10	9	1	d	10	Y	2	gynecolog	R	B	4	31397	7
11	10	10	e	1	X	2	gynecolog	S	E	3	31397	7
12	11	22	g	9	Y	2	radiothera	S	B	2	31397	7
13	12	26	b	2	Y	4	radiothera	R	D	1	31397	7
14	13	16	c	3	Z	2	radiothera	R	A	3	31397	7

Fig. 1. Raw Dataset



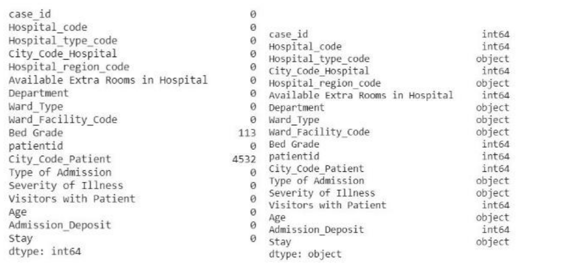
	case_id	Hospital_code	Hospital_type_code	City_Code_Hospital	Hospital_region_code	Available Extra Rooms in Hospital	Department	Ward_
0	1	8	2	3	2	3	3	
1	2	2	2	5	2	2	3	
2	3	10	4	1	0	2	1	
3	4	26	1	2	1	2	3	
4	5	26	1	2	1	2	3	
...	...	...	...	...	...	...	...	
318433	318434	6	0	6	0	3	3	
318434	318435	24	0	1	0	2	1	
318435	318436	7	0	4	0	3	2	
318436	318437	11	1	2	1	3	1	
318437	318438	19	0	7	1	5	2	

318438 rows × 18 columns

Fig. 2. Dataset after using `pandas` API

2 Handling Missing Values

The dataset contained several records which contained missing values for some attributes.



```
case_id                                0
Hospital_code                          0
Hospital_type_code                     0
City_Code_Hospital                     0
Hospital_region_code                   0
Available Extra Rooms in Hospital      0
Department                             0
Ward_Type                              0
Ward_Facility_Code                     0
Bed Grade                            113
patientid                              0
City_Code_Patient                   4532
Type of Admission                      0
Severity of Illness                    0
Visitors with Patient                  0
Age                                    0
Admission_Deposit                      0
Stay                                   0
dtype: int64
```

```
case_id                               int64
Hospital_code                         int64
Hospital_type_code                   object
City_Code_Hospital                    int64
Hospital_region_code                 object
Available Extra Rooms in Hospital     int64
Department                           object
Ward_Type                            object
Ward_Facility_Code                   object
Bed Grade                             int64
patientid                             int64
City_Code_Patient                     int64
Type of Admission                    object
Severity of Illness                  object
Visitors with Patient                 int64
Age                                  object
Admission_Deposit                     int64
Stay                                 object
dtype: object
```

Fig. 3. Number of missing values and data types for each field

As it can be observed from Fig. 2, that fields `Bed_Grade` and `City_Code_Patient` have 113 and 4532 missing values. As both these fields contain categorical data, we have filled the missing values with most common value of the fields.

3 Encoding Categorical Variables

Most of the fields in our dataset contain categorical data only. To encode these variables, we used `sklearn`'s `LabelEncoder` API. It assigns discrete integral labels 0, 1, 2, … and so on to various distinct categories. Futhermore, the field `Admission_Deposit` contains continuous values which might be a problem to the Discrete Naïve Bayes classifier. To avoid problems, we also encoded this field as follows: It was observed that the values in the field vary between 1000 – 12000. We divided the values by 1000 and used the rounded-off value as the category of that values. It was also observed that fields `case_id` and `patient_id` contained a lot of distinct values and might pose a hindrance to the Discrete Naïve Bayes classifier, hence, we dropped these two fields from the dataset.

4 K-Fold Cross Validation and Test Accuracy

Just after pre-processing the dataset, we used 5-fold cross validation to get an idea how well the model is performing over the dataset.

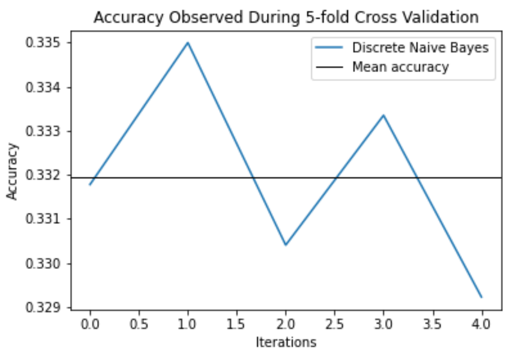


Fig. 4. 5-fold cross-validation on Discrete Naïve Bayes

The mean accuracy observed over cross-validation was just 33.19%. It implies that either the model used in not well-suited for the given task or we need to further process the dataset. The final test accuracy was observed to be 33.21%.

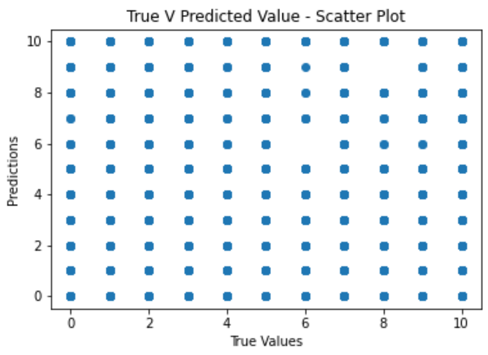


Fig. 5. Scatter plot of True vs Predicted value on test set

As it can be observed from Fig. 5 the model is performing poorly on the test set. Instead of a straight line, we are getting a lot of scattering between the actual and predicted values.

5 Applying Principal Components Analysis

The discrete Naïve Bayes is performing poorly on the dataset. Now there are two possible reasons for that: Either Naïve Bayes is not suited for the task or the dataset needs further pre-processing. At present, let's have faith in Naïve Bayes and try to further pre-process the dataset. We are using the dimensionality reduction technique. Firstly, we are using the unsupervised PCA approach. We observe that the dimension of the instances in the dataset is 15. This implies, we can have at maximum 15 principal components. Let's try to find those components and analyse their variance share. For PCA, we are using `sklearn`'s `PCA` API.

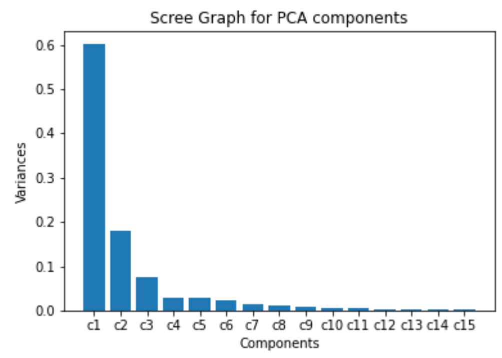


Fig. 6. Scree Graph for the Principal Components

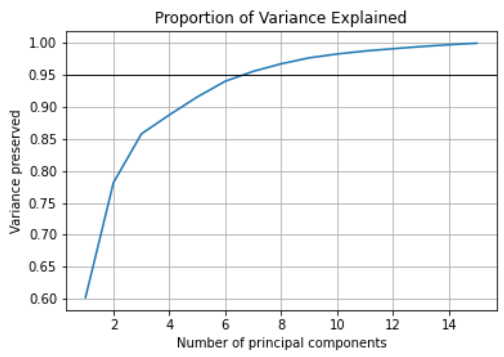


Fig. 7. Proportion of variance explained by selecting k features

As we have to preserve 95% of variance, we need to select seven principal components. After this, we transform the training set samples and test set samples using the first seven principal components and perform cross-validation using Naïve Bayes classifier with *Gaussian* kernel.

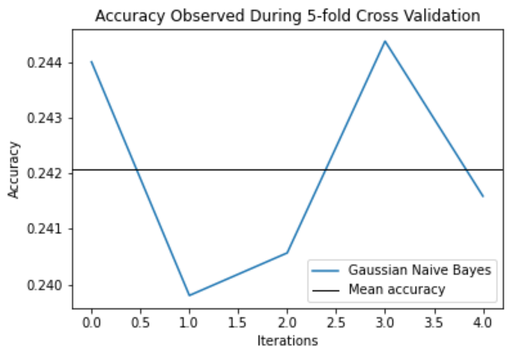


Fig. 8. 5-fold cross-validation after applying PCA

The mean accuracy of cross-validation was observed to be 24.20% and the final test accuracy was observed to be 26.38%. From this, we draw the conclusion that PCA was not effective in improving the performance of the Naïve Bayes classifier.

6 Removing Outliers

Consider a field f which contains continuous values. A record R will be an outlier if $\left|\frac{R_f-\mu}{\sigma}\right| > 3$, where μ is the mean of field f, and σ is the standard deviation. The dataset contained only one field which contained continuous data, which was `Admission_Deposit`. However, in the initial steps, we encoded this field to contain categorical values. So, first, we decoded the field again, removed outliers and re-decoded it. However, for fields containing categorical data, we cannot detect outliers as this process doesn't involves mean and standard deviation and is much more complex.

7 Applying Backward Selection Method for Feature Selection

As the performance of the model is not improving, it might be reasonable to remove some fields that are not contributing to the accuracy. For this we are using backward-selection method for feature selection. After the execution, five fields were dropped from the dataset.

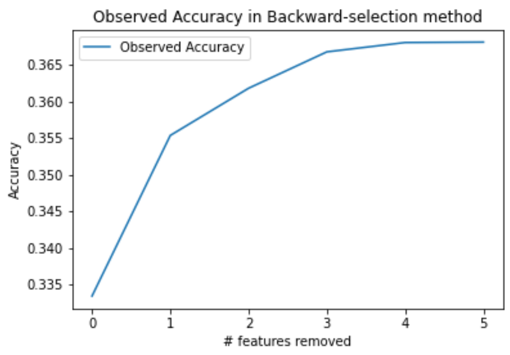


Fig. 9. Increase in accuracy while fields are dropped

We trimmed the training and test samples as per the best feature selected and performed 5-fold cross-validation and observed final test accuracy.

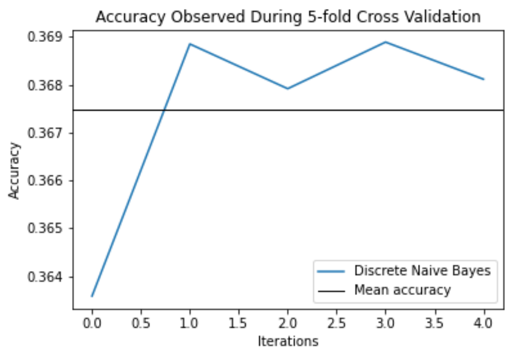


Fig. 10. 5-fold cross-validation on best features selected

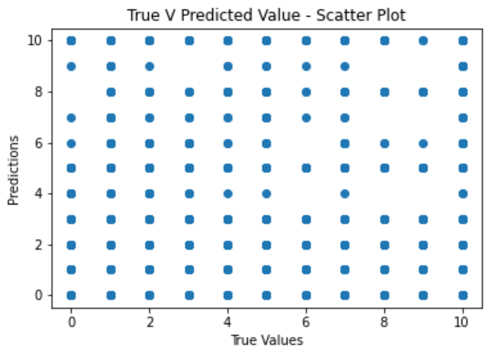


Fig. 11. Scatter plot of true labels and predicted labels

The mean accuracy of 5-fold cross-validation was observed to be 36.74% and the final test accuracy was observed to be 36.79%.

8 Conclusion

Table 1: Comparing Different Approaches

Technique Used	**Mean Accuracy (Cross Val)**	**Final Test Set Accuracy**
Encoding only	0.3319	0.3321
PCA	0.2420	0.2638
Feature Selection	0.3674	0.3679

We can conclude from Table 1 that the PCA approach was not useful in this task, however the feature selection approach increased the accuracy of the model. However, it may be noted that this increase is not substantial. Having faith in the Naïve Bayes, we used the `sklearn`'s `GaussianNB` API for the task. However, it also performed poorly on the dataset after using all the techniques discussed in this paper. Therefore, now, we can safely say that Naïve Bayes is ill-suited for this task.