PROJECT -3
Comparison of Different Classifiers

1]Introduction of Dataset.

The name of the dataset that we used was the Churn rate which defines the percentage of subscribers who want to discontinue their subscription to the service within a given period of time. For a company to expand its growth rate and clientele which is measured by the number of new customers, it is very important that it exceeds the churn rate.

The **third dataset** we have used here is Titanic.csv, which is about "The sinking of the Titanic is one of the most infamous shipwrecks in history."

On April 15, 1912, during her maiden voyage, the widely considered "unsinkable" RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren't enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

We are using Titanic.csv with information in 12 columns ,PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked. Based on this we will predict whether they survived or not given these circumstances.

The **fourth dataset** we have used is Indian_Liver_Patient ,which is about patients with Liver disease which have been continuously increasing because of excessive consumption of alcohol, inhalation of harmful gases, intake of contaminated food, pickles and drugs. This dataset was used to evaluate prediction algorithms in an effort to predict whether patient has disease or not.

2]Observation of Datasets and partitioning into subsets

The dataset had many columns namely CustomerId, Surname, CreditScore, Geography, Gender, Age, Tenure, Balance, NumOfProducts, HasCrCard, IsActiveMember, EstimatedSalary and Exited.

If we use Surname, CustomerId, Gender, Age, etc it does not give accurate predictions. So grouping should be done on the basis of higher prediction and higher accuracy.

So we divided the whole dataset into 2 different subsets namely dataset_1 and dataset_2. The dataset_1 had column names CustomerId, Surname, CreditScore, Geography, Gender, Age, Tenure, Balance, EstimatedSalary and Exited. The dataset_2 had column names NumOfProducts, HasCrCard, IsActiveMember, Balance, EstimatedSalary, and Exited.

We have used similar approaches for given datasets 3,4 as well ,which is to group more related data.

3]What are the accuracy or prediction metrics used for comparing classifiers?

Common accuracy measure is given by the following formula:-

Accuracy rate = (Number of correctly classified observations/Total number of observations)

Accuracy in % = Accuracy * 100

We used 2 metrics to compare classifiers namely Accuracy Score and Confusion Matrix.

1. Accuracy Score - This function computes the accuracy, either the fraction or the count of correct predictions.

For multilabel classification, the function returns the subset accuracy. If the entire set of predicted labels for a sample strictly matches the true set of labels, then the subset accuracy is 1.0; otherwise, it is 0.0.

2.Confusion Matrix - This function evaluates classification accuracy by computing the confusion matrix with each row corresponding to the true class.

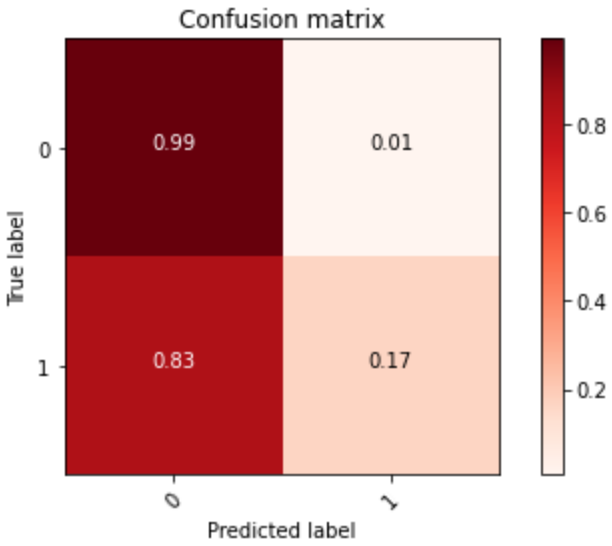
By definition, entry i,j in a confusion matrix is the number of observations actually in group i, but predicted to be in group j.

Precision Score - The precision is the ratio (tp / tp + fp) where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.

Recall Score - The recall is the ratio (tp / tp + fn)where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

F1 Score - The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and the worst score at 0.

F1 = 2 * (precision * recall) / (precision + recall)



ML Model	**Accuracy on Test Set (%)**	
	1 Dataset	2 Dataset
Naïve Bayes	0.77	0.79
KNN(Also provide the K value for which you got the highest accuracy)	0.83 k =2	0.84, k=2
SVM	kernel='rbf'	kernel='rbf'
	0.80 C=0.01.0.1 0.98 C=1	0.80 C=0.01,0.1 0.99 C=1
	kernel='Linear'	kernel=linear
	0.786 C=all values	0.786 C=all values
Decision Tree	0.81	0.82
Logistic Regression	0.788	0.79

Table 1:Accuracy Results of churn_modeling.csv

4] Which data subset proved to be more effective for prediction?

Clearly as we can see from the above table 1, dataset_2 proved to deliver more accurate results.KNN model and SVM model(kernel='rbf') have high accuracy for prediction with a value 0.84 and 0.99(for C=1). Here the C value for the SVM model has a larger influence.The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassified more points.

dataset_1=df[['CustomerId','CreditScore','Geography','Gender','Age','Tenure','Balance','Estimated Salary','Exited']]

dataset_2=df[['NumEProducts','HasCrCard','IsActiveMember','Balance','EstimatedSalary','Exited']]

Also dataset_2 has a more substantial form of training columns or fields to accurately predict whether he/she will continue subscription or not as compared to subset 1.

Similarly for dataset 3 and 4 we can write as -
dataset_3=df[['Survived','Pclass','Age','SibSp','Parch','Fare']]

dataset_4=df[['Direct_Bilirubin','Alkaline_Phosphotase','Alamine_Aminotransferase','Aspartate_Aminotransferase','Albumin','Dataset']]

5]Which Classifier proved out to be better for classifying data?

Clearly KNN (for k=2) and SVM (kernel='rbf') models turn out to be more effective classifiers. AS we can see from the plot for KNN, k=2 value of accuracy decreases as we tend to increase k neighbours. Similarly, we can observe a constant line for C values from 0.01 to 1. At an average, both perform equally well for classifying.

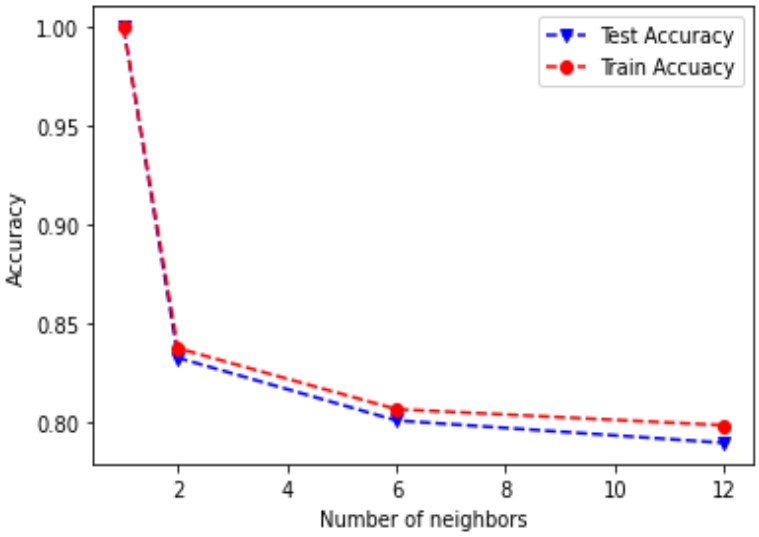


Figure 1: KNN

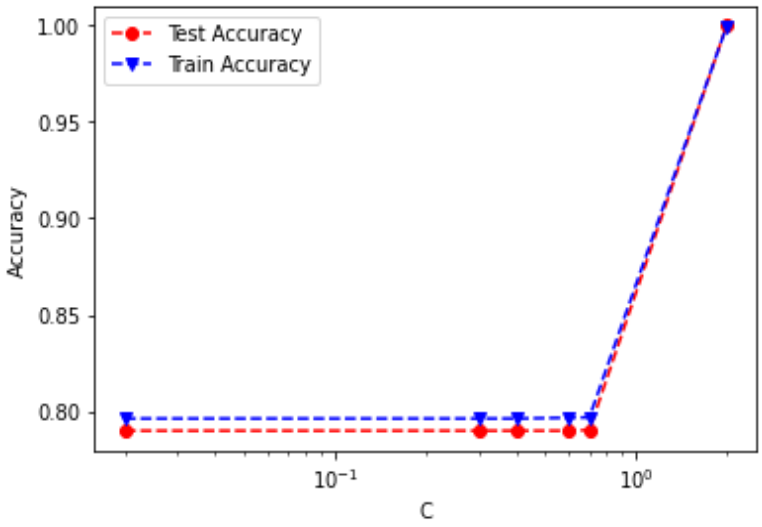


Figure 2: SVM

Also, Naive Bayes turns out to be less effective and efficient when with predicted results of only 0.79%.

ML Model	Accuracy on Test Set (%) 3 Dataset(Titanic)	 4 Dataset (Liver patient)
Naïve Bayes	0.73	0.53
KNN(Also provide the K value for which you got the highest accuracy)	0.73 k =2	0.84, k=2
SVM	kernel='rbf' 0.69 C=0.01.0.1 0.91 C=1 kernel='Linear' 0.676 C=all values	kernel='rbf' 0.71 C=0.01,0.1 0.99 C=1 kernel=linear 0.707 C=all values
Decision Tree	0.89	0.71
Logistic Regression	0.73	0.699

Table 2:Accuracy Results of Titanic.csv file and Indian_Patient_Liver.csv file

Further, we introduced two more datasets for investigation namely, ***Titanic.csv file and Indian_Patient_Liver.csv.***
Accuracy results for the Titanic dataset are high for decision tree classifiers followed by SVM. Similarly, for dataset_4, Indian_Patient_Liver.csv it has a lower accuracy result for naive Bayes but has a high accuracy rate for KNN. Here KNN for k=2 proves out to be a more effective model followed by SVM.

Conclusion

We can conclude KNN works better than SVM, given that SVM uses linear hyperplanes to separate classes, and if you provide a different kernel, then it will change the shape of the decision manifold that can be used.

KNN can generate a highly convoluted decision boundary as it is driven by the raw training data itself. SVM uses a highly restricted parametric approximation of the decision boundary, which is an excellent trade-off for classification performance against data storage space/ processing speed.

By Suraj Rawat and Khushali Upadhyay
Code + Documentation (50 %) Code + Documentation (50 %)