Brain Tumor Prediction

Name:	**Vasu Dhull**
Roll No.:	21302
Institute Name:	IISER Bhopal
Stream:	DSE
Problem Release date:	August 17, 2023
Date of Submission:	November 19, 2023

1 Introduction

The dataset contains 24 features and 775 instances of which 449 belong to one class, namely LGG and 326 belong to the other class GBM. Their are 17 missing values and all the missing values are filled or one-hot encoded. Most of the data i.e. 21 out of 24 only have two values(True or False) so instead of assigning 0 and 1 to each we do one-hot encoding of the entire data set, along with these 21 features 2 other features can also be one-hot encoded and their is only one feature(Age_at_diagnosis) which can be converted to integer values. After all the one-hot encoding the number of features from 24 now becomes 54. We don't not change the number of instances and the final dataframe which we are left with is 775 X 54.
We remove a feature that has the highest correlation, as it leads to overfitting of the models which we built. Then on the new dataframe which we have, we apply feature selection technique to find the best features for our data set, then we apply SVM [1], decision tree[2], KNN[3], Adaptive Boosting[4], Random Forest[5] and Logistic Regression[6], then, if required we fine tune the parameters and get the best possible pression, recall and f1-score[7].

2 Methods

Pseudo-code
1. Remove the feature that has the highest corelation with the labels.
2. Fill the missing values with mean and mode, according to requirement.
3. Do one-hot encoding of the dataset, except for one integer column.
4. Scale the Age column by using MinMax scaling.
5. Built the models from the given data set that is left after the entire processing.

We find that Primary_diagnosis has a really high correlation with the labels, and that might lead to over fitting of over model, so we drop that feature and then carry on with the rest of the features. We start by clearing out the dataset.The first thing is to make all the columns with '–' and'Not 'to None then, we fill the missing values of the 'Gender' and 'Race' column, as they can be replaced by the mode of that particular column. For the age column we write a simple code for changing the string value to integer value by rounding of the years based on the number of days there are, then filling the missing values with the round off, of the mean value of that column.Next step is one-hot encoding of the remaining columns while leaving the Age column as it is.

After making Our dataset, we scale our Age column to be between 0 and 1, and to do that we use MinMaxScaler, as it reduces the values to be between 0 and 1. This way all the entires in our data are now between 0 and 1. The next step is feature selection, we are doing to select features based on their correlation score, the features with high correlation score can be removed, as they match so much with some other feature, the remaining data set left is now ready to built our models.

Table 1: Performance Of Different Classifiers Using All Features

Classifier	Precision	Recall	F-measure
Adaptive Boosting	0.88	0.89	0.88
Decision Tree	0.82	0.83	0.82
K-Nearest Neighbor	0.89	0.90	0.88
Logistic Regression	0.88	0.89	0.88
Random Forest	0.88	0.89	0.88
Support Vector Machine	0.88	0.88	0.87

We start by doing the Train test Split.The first model which we built is SVM then other model were built, the model were built to get the highest possible accuracy that can be achieved. The rest of the procedures are mentioned in the Experimental Setup.

3 Experimental Setup

The first model which we built is **SVM**, we run the SCV while using grid search, the main output is based on the f1- score, i.e. the hyper parameters that have the highest f1-score would be showen in the output. Best Parameters: 'C': 10, 'degree': 2, 'gamma': 'scale', 'kernel': 'linear', 'random_state': 42.
The next model which we make is **Adaptive Boosting**, we have to tune the same we turned SVM, i.e. by using grid search, and the grid search is again based on the f1-score, the model gives the Best Parameters: 'algorithm': 'SAMME.R', 'base_estimator': DecisionTreeClassifier(max_depth=1), 'learning_rate': 0.1, 'n_estimators': 100, 'random_state': 42
The next model we built is **Decision tree**, we use the grid Search to find the best possible f1-score for the decision tree, after running the code a couple of times, we land on the following values('criterion': 'entropy', 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2) these values were found after a couple of time we do the grid search parameter tuning and the final f1-score comes out to be
The next model is **LogisticRegression** and for this model we again do the grid search, the output, that is based on f1-score, has best parameters as 'C': 1, 'fit_intercept': False, 'max_iter': 50, 'penalty': 'l2', 'solver': 'newton-cg' .
The next model is **Random Forest**,we make the grid, which we would be folding 10 times to create the best possible model, the output is based on f1-score which we land on Best Parameters: 'bootstrap': False, 'criterion': 'entropy', 'max_depth': None, 'max_features': 'log2', 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 50
The next model is **KNN**, we make the grid, and tune the model again and again until we get the highest possible f1-score, and Best Parameters: 'algorithm': 'auto', 'leaf_size': 30, 'metric': 'minkowski', 'n_neighbors': 10, 'p': 1, 'weights': 'uniform'.
The pression, recall and f - measure are all given in the table and they are all based on the macro avg. The confusion matrix are also found from the code, the train and test spilt is done on 80% for training and the rest for the validation.

4 Results and Discussion

The best performing model is KNN, as the precision and recall are the highest in it1, another factor for KNN being the best classifier is that, it only predicted wrong for 2 cases of GBM, which should be the main focus, as GBM is more dangerous then LGG, and it also falsely predicts LGG only 16 times out of 90, which is not the best, in regards to other models2, but due to more accuracy on GBM prediction, KNN is the best classifier. We use KNN as our main classifier to predict the labels on the test data, some features were actually added to the test data, because they were present during the training and were not present in the test data, we add those features as a zero array of 87 size(the

Table 2: Confusion Matrices of Different Classifiers

Actual Class	Predicted Class	
	GBM	LGG
GBM	63	2
LGG	17	73

Adaptive Boosting

Actual Class	Predicted Class	
	GBM	LGG
GBM	56	9
LGG	19	71

Decision Tree

Actual Class	Predicted Class	
	GBM	LGG
GBM	63	2
LGG	16	74

K-Nearest Neighbor

Actual Class	Predicted Class	
	GBM	LGG
GBM	62	3
LGG	16	74

Logistic Regression

Actual Class	Predicted Class	
	GBM	LGG
GBM	61	4
LGG	14	76

Random Forest

Actual Class	Predicted Class	
	GBM	LGG
GBM	63	2
LGG	18	72

SVM

size of the test data) this way we would be able to predict the test data and get the labels.

5 Conclusion

We must check for over fitting of any individual feature by taking the correlation with the target labels, as it leads to really high accuracy on the validation set, but we might not be that sure about it on the test data.

The main draw-back actually comes here, as the columns are to be re_arranged, if they are not re-arranged then the model would throw an error.
Another draw-back is that we have no way to tell if the highly correlated feature overfits or simply is a really good estimator.
According to the Results we can see that all the models almost predict the same, but we take the one which is best at the prediction of GBM as it is a more serious case and should be predicted as such as well.
The code and report -¿ `https://github.com/VasuDhull/lgg-gbm-prediction-ML`

References

[1] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[2] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, S Yu Philip, et al. Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1):1–37, 2008.

[3] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[4] Robert E Schapire. Explaining adaboost. In *Empirical inference*, pages 37–52. Springer, 2013.

[5] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[6] David R Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(2):215–232, 1958.

[7] Juri Opitz and Sebastian Burst. Macro f1 and macro f1, 2019.