INDIAN INSTITUTE OF TECHNOLOGY MADRAS

CH5019

Mathematical Foundations for Data Science Project

Prahlad Narayanan (BE19B005), Rohan Saswade (BS19B027), Hari Ranjan Meena (ME18B108), G. Saicharan (ED19B065), Ankit Srivastava (BE19B038)

Contents

1 Training Dataset

Training data for Machine Learning (ML) is a crucial input to algorithms that comprehends this dataset and memorizes the critical information for future prediction. Hence, Training data is a backbone of the entire ML project, without which it is not possible to train a machine.

For this project, we used a glue resource on TensorFlow. Specifically, the stsb dataset in the glue resource. The Semantic Textual Similarity Benchmark (stsb) is a collection of sentence pairs drawn from news headlines, video and image captions, and natural language inference data. Each pair is human-annotated with a similarity score from 0 to 5.

1.1 STS benchmark dataset and companion dataset

STS Benchmark comprises a selection of the English datasets used in the STS tasks organized in the context of SemEval between 2012 and 2017. The choice of datasets includes text from image captions, news headlines, and user forums. The dataset has been organized into train, development, and test to provide a standard benchmark to compare meaning representation systems in future years. The development part of the dataset can be used to develop and tune hyperparameters of the systems, and the test part of the dataset should be only used once for the final system.

1.2 Fetching Dataset

Dataset has been loaded using the load_dataset function of the datasets library in python.

Further, the dataset train, validation, and test are combined into a single array. The two answers are labeled with a similarity gradient of range from 0 to 5. They are transformed to binary with a similarity threshold of 3.5.

The answers or text are further processed to create relevant features in the model.

	answer1	answer2	is_duplicate
0	A plane is taking off.	An air plane is taking off.	1
1	A man is playing a large flute.	A man is playing a flute.	1
2	A man is spreading shreded cheese on a pizza.	A man is spreading shredded cheese on an uncoo...	1
3	Three men are playing chess.	Two men are playing chess.	0
4	A man is playing the cello.	A man seated is playing the cello.	1

2 Testing Dataset

The model's actual performance can only be understood by providing novel inputs to the model with a similar pattern as that of the input. 10 Test cases based on answers that include active and passive voice, synonyms, different answers, etc.

	Question	Target Answer	Provided Answer	Target	pred
0	Which is the largest country in the world	Russia is the largest country in the world	World's largest country is russia	1	1
1	Which country has the largest population in th...	China has the largest population in the world	China is the most populated country in the world	1	0
2	How many states of India share its border with...	Four states in India share a border with Bhutan	A border with Bhutan is shared by four states ...	1	1
3	What is the capital city of India?	New Delhi is the capital city of India	India's capital is New Delhi	1	1
4	In which country would you find the Leaning To...	Leaning Tower of Pisa is in Italy	Mexico has Leaning Tower of Pisa	0	0
5	Which is colder: The North Pole or the South P...	South Pole is colder than North Pole	South pole is the coldest	1	0
6	Which planet is nearest to the Earth?	Venus is nearest to Earth	Venus is closest to Earth	1	1
7	Which is the biggest desert in the world?	Sahara Desert is the largest desert	Largest Desert is Sahara Desert	1	1
8	Which is the hottest continent on Earth?	Africa is the hottest continent of Earth	Africa is hottest continent	1	1
9	Which is the highest mountain in the world?	Mount Evertest is the highest mountain	Mount Everest is the tallest mountain in the w...	1	0

The model showed 70% accuracy in the test cases and was close to its training accuracy, showcasing that the model has not overfitted to the training data.

3 Exploratory Data Analysis

3.1 Data Preprocessing

Data Preprocessing is the most essential step for any Machine Learning model. How well the raw data has been cleaned and preprocessed plays a major role in the performance of the model. Likewise in the case of NLP, the very first step is Text Processing.
The various preprocessing steps that are involved are :

- Lower Casing
- Replace certain special characters with their string equivalents i.e % to percent
- Decontracting words i.e "ain't" to "am not"
- Tokenization
- Punctuation Mark Removal
- Stop Word Removal

3.2 Feature Engineering

Feature engineering refers to the process of using domain knowledge to select and transform the most relevant variables from raw data when creating a predictive model using machine learning or statistical modeling. The goal of feature engineering and selection is to improve the performance of machine learning (ML) algorithms.
The features that were generated manually were:

- a1_len and a2_len - number of character in answer
- a1_num and a2_num - number of words in answer
- word_common - count of common words in answers
- word_total - total number of words in answers
- word_share - ratio of word_common to word_total
- cwc_min - This is the ratio of the number of common words to the length of the smaller question
- cwc_max - This is the ratio of the number of common words to the length of the larger question

- csc_min - This is the ratio of the number of common stop words to the smaller stop word count among the two questions
- csc_max - This is the ratio of the number of common stop words to the larger stop word count among the two questions
- ctc_min - This is the ratio of the number of common tokens to the smaller token count among the two questions
- ctc_max - This is the ratio of the number of common tokens to the larger token count among the two questions
- last_word_eq - 1 if the last word in the two questions is same, 0 otherwise
- first_word_eq - 1 if the first word in the two questions is same, 0 otherwise
- mean_len - Mean of the length of the two questions (number of words)
- abs_len_diff - Absolute difference between the length of the two questions (number of words)
- longer_substr_ratio - Ratio of the length of the longest substring among the two questions to the length of the smaller question
- fuzz_ratio - fuzz_ratio score from fuzzywuzzy
- fuzz_partial_ratio - fuzz_partial_ratio from fuzzywuzzy
- token_sort_ratio - token_sort_ratio from fuzzywuzzy
- token_set_ratio - token_set_ratio from fuzzywuzzy

3.3 Feature Visualization

In the training dataset we first visualized the features present in it. In this section we shall report some of the salient features observed.

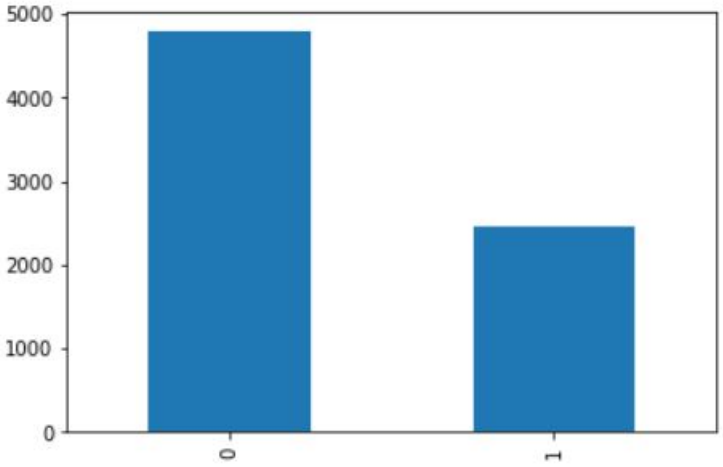


Figure 3.1: Visualizing the number of similar and dissimilar sentences

As we can see here, we report there are 2467 answers that are similar and 4782 answers that are not.

For the variable a1_len and a2_len, the variables that represent the length of both sentences, we report the following counts.

- minimum characters in answer1: 12
- maximum characters in answer1: 367
- average num of characters in answer1: 59
- minimum characters in answer2: 15
- maximum characters in answer2: 311
- average num of characters in answer2: 58

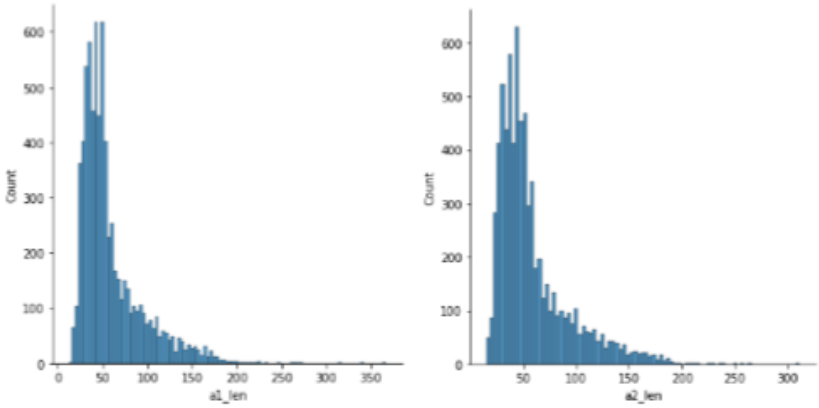


Figure 3.2: Plotting the counts of the above written features

For variables a1_num_words and a2_num_words we report the following numbers for this feature.

- minimum words in answer1: 3
- maximum words in answer1: 56
- average num of words in answer1: 10
- minimum words in answer2: 2
- maximum words in answer2: 48

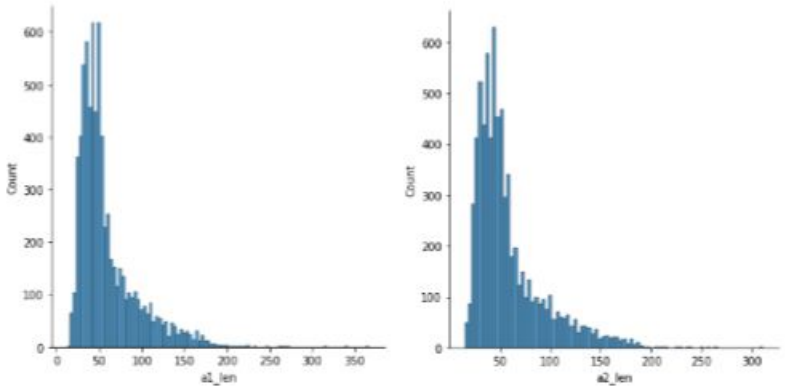


Figure 3.3: Counts of each of the above feature for answer1 and answer2

We also report that if there are less then 4 common words in answers, then there is more probability that answers are non duplicate.

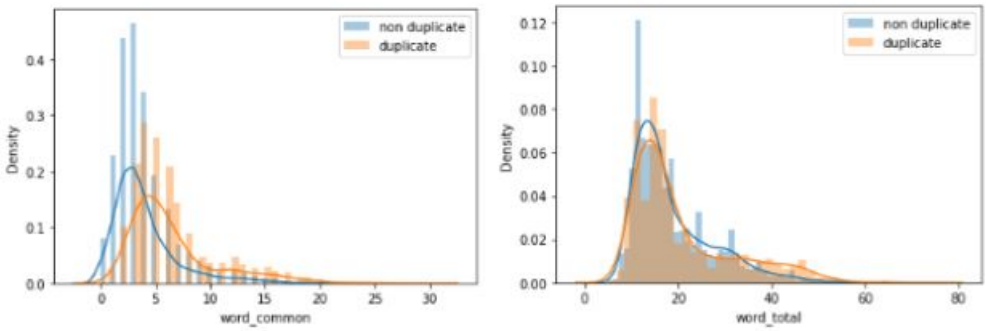


Figure 3.4: Probability density of duplicate answer based on number of shared words and total lengths

We also find that if the word_share value is more than 0.25, we noted that there is a greater probability of the sentences being a match, if it is less than 0.25 we see that the probability of a mismatch is higher

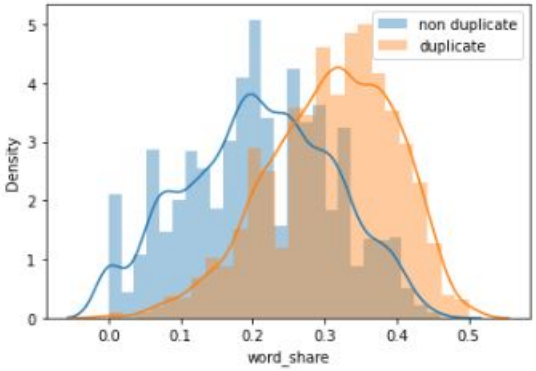


Figure 3.5: Probability density of match and mismatch based on number of shared words

Based on the correlation of features we also built a correlation heatmap.

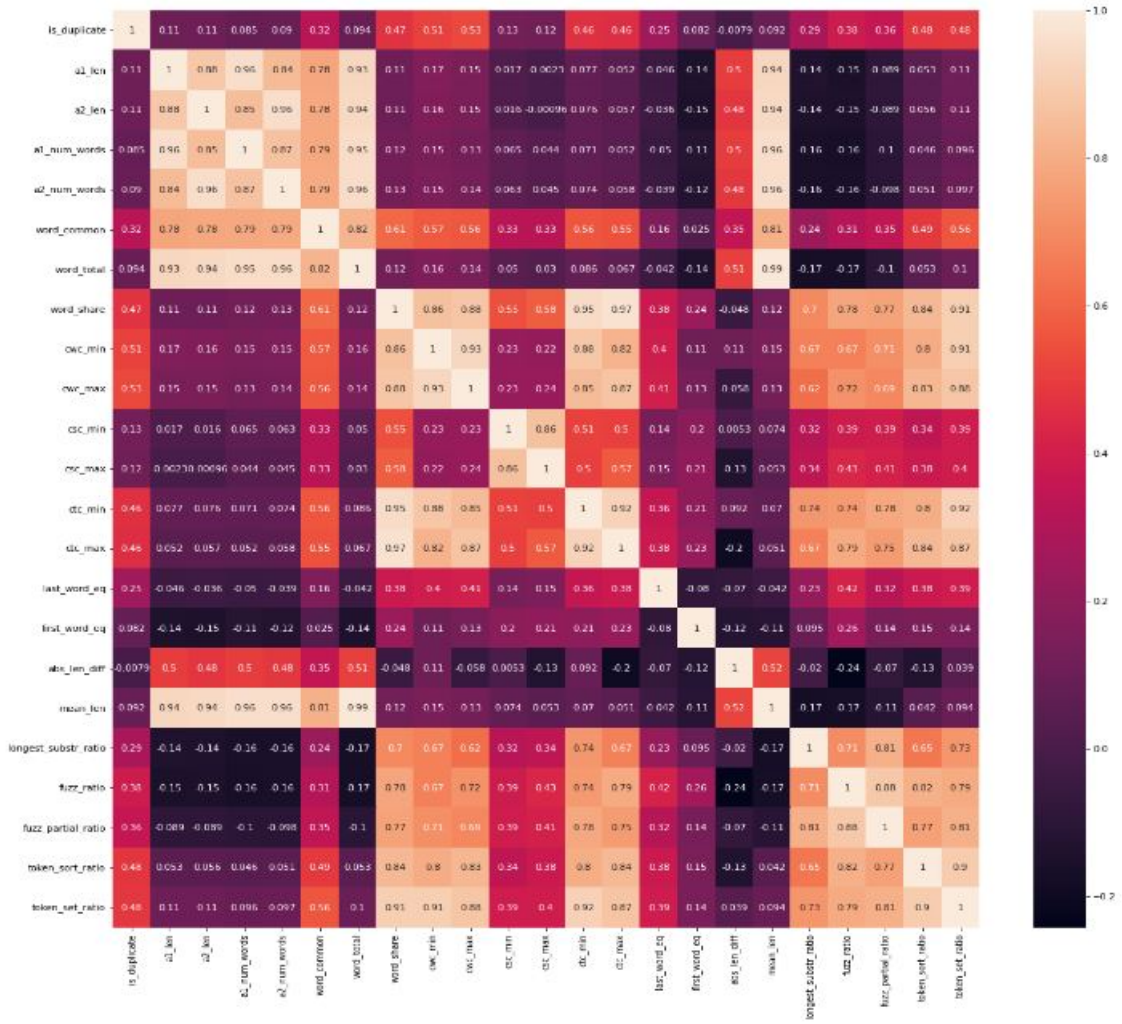


Figure 3.6: Correlation heatmap of all features

3.4 Dimensionality Reduction

The t-SNE algorithm calculates a similarity measure between pairs of instances in the high dimensional space and in the low dimensional space. It then tries to optimize these two similarity measures using a cost function.

Notable Advantages of t-SNE:

- Handles Non-Linear Data Efficiently
- Preserves Local and Global Structure

From the below plot we can say that , we are able separate both the categories with

low nearest-neighbor accuracy because there is overlapping in some area.

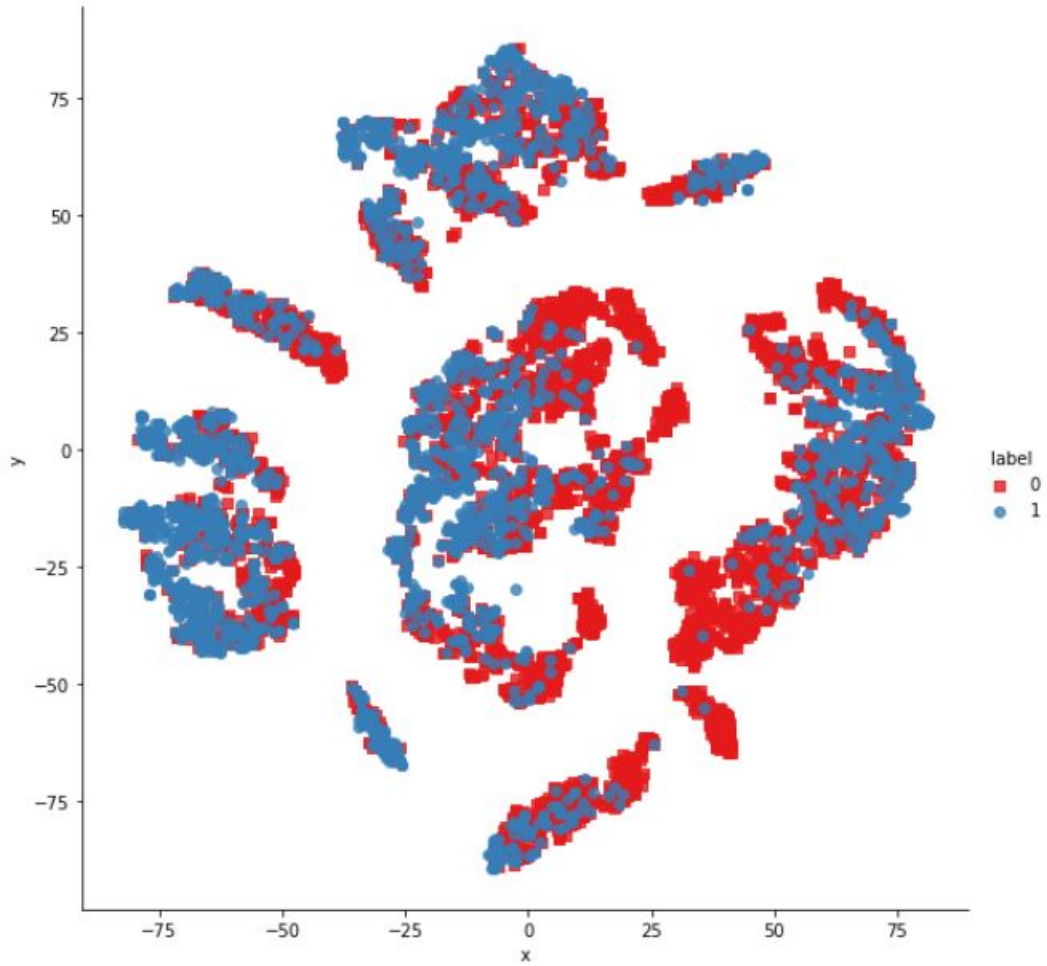


Figure 3.7: Seperation of categories based on Nearest Neighbour Accuracy

4 Model Building and Results

All models have some predictive error, given the statistical noise in the data, the incompleteness of the data sample, and the limitations of each different model type. Therefore, the notion of a perfect or best model is not useful. Instead, we must seek a model that is "good enough."

4.1 Testing Different Classification Algorithms

Computation of Model Performance is the next logical step to choose the right model. The model performance metrics will decide on final model selection, that include computation of accuracy, precision, recall, F1 score (weighted average of precision and recall) , along with confusion matrix for classification models and coefficient of determination for regression models. It is not recommended to use accuracy as a measure to determine the performance of classification models that are trained with imbalanced/skewed datasets, rather precision and recall are recommended to be computed to choose the right classification model.

Algorithm	Accuracy	Precision	f1_score	Recall
LR	0.795862	0.741379	0.699187	0.661538
RF	0.799310	0.775904	0.688770	0.619231
AdaBoost	0.785517	0.726681	0.682977	0.644231
xgb	0.795172	0.769976	0.681672	0.611538
GBDT	0.796552	0.780549	0.679696	0.601923
BgC	0.793793	0.775561	0.675353	0.598077
DT	0.783448	0.741784	0.668076	0.607692
ETC	0.785517	0.763224	0.660851	0.582692
KN	0.726897	0.648325	0.577825	0.521154
NB	0.640690	0.499248	0.560338	0.638462
SVC	0.641379	0.000000	0.000000	0.000000

Table 1: Performance of each Classification Model

We can employ ensemble learning techniques to improve the performance of machine learning models. For example it can increase the accuracy of classification models. Ensembling will also result in a more stable model. The models in the ensemble would then improve performance on the dataset by combining their predictions.

4.2 Ensemble Learning

Ensemble learning is a combination of several machine learning models in one problem. These models are known as weak learners. The intuition is that when you combine several weak learners, they can become strong learners. Each weak learner

is fitted to the training set and provides predictions obtained. The final prediction result is computed by combining the results from all the weak learners.

4.2.1 Voting Classifier

A voting classifier is a machine learning estimator that trains various base models or estimators and predicts on the basis of aggregating the findings of each base estimator. The aggregating criteria can be combined decision of voting for each estimator output. The voting criteria can be of two types:

- **Hard Voting:** Voting is calculated on the predicted output class.
- **Soft Voting:** Voting is calculated on the predicted probability of the output class.

How Voting classifiers improve performance?
The voting classifier aggregates the predicted class or predicted probability on the basis of hard voting or soft voting. So if we feed a variety of base models to the voting classifier it makes sure to resolve the error by any model.

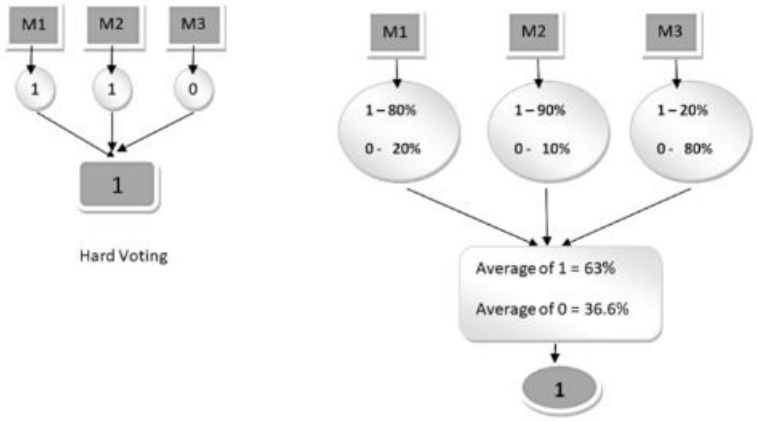


Figure 4.1: How a voting classifier works

Soft Voting
Soft voting classifier classifies input data based on the probabilities of all the predictions made by different classifiers. Weights applied to each classifier get applied appropriately based on the equation given below.

$$\hat{y} = \underset{i}{\operatorname{argmax}} \sum_{j=1}^{m} w_j x_A(C_j(x) = i) \tag{4.1}$$

Base Models for Voting Classifier:

- **XGBoost** - XGBoost stands for eXtreme Gradient Boosting.It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. XGBoost is an ensemble tree method that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture.

- **Random Forest Classifier** - Random forest, as its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. Random Forest uses a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. This process is sometimes called feature bagging. The reason for doing this is the correlation of the trees in an ordinary bootstrap sample: if one or a few features are very strong predictors for the response variable (target output), these features will be selected in many of the B trees, causing them to become correlated.

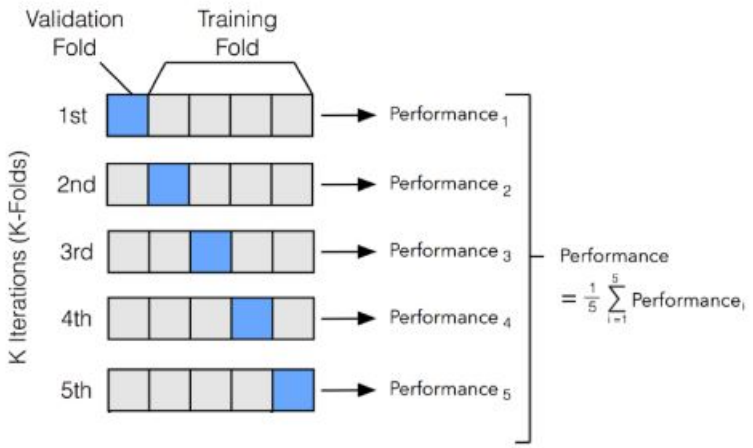
- **Logistic Regression** - Logistic regression is a technique used when the dependent variable is categorical. Logistic regression employs binomial probability theory in which there are only two values to predict: that probability (p) is 1 rather than 0, i.e. the event belongs to one group rather than the other. Consider a model with two predictors, 1x1 and x2, and one binary (Bernoulli) response variable, Y, which we call p=P(Y=1). We assume that the predictor variables and the logodds (logit) of the occurrence that Y=1 have a linear relationship.

4.3 Hyperparamater Optimization

Often the general effects of hyperparameters on a model are known, but how to best set a hyperparameter and combinations of interacting hyperparameters for a given dataset is challenging. There are often general heuristics or rules of thumb for configuring hyperparameters.
A better approach is to objectively search different values for model hyperparameters and choose a subset that results in a model that achieves the best performance on a given dataset. This is called hyperparameter optimization or hyperparameter tuning and is available in the scikit-learn Python machine learning library. The result of a hyperparameter optimization is a single set of well-performing hyperparameters that you can use to configure your model.

4.3.1 RandomsearchCV to Tune Base models

In Random Search, we create a grid of hyperparameters and train/test our model on just some random combination of these hyperparameters. In our case, we additionally decided to perform Cross-Validation on the training set.
When performing Machine Learning tasks, we generally divide our dataset in training and test sets. This is done so that to test our model after having trained it (in this way we can check it's performances when working with unseen data).
When using Cross-Validation, we divide our training set into N other partitions to make sure our model is not overfitting our data.One of the most common used Cross-Validation methods is K-Fold Validation. In K-Fold, we divide our training set into N partitions and then iteratively train our model using N-1 partitions and test it with the left-over partition (at each iteration we change the left-over partition). Once having trained N times the model we then average the training results obtained in each iteration to obtain our overall training performance results.



Best Estimator

- **Xgb classfifer** - 'subsample': 1.0, 'n_estimators': 100, 'min_child_weight': 10, 'max_depth': 8, 'learning_rate': 0.1, 'gamma': 1, 'colsample_bytree': 0.6

- **RF classfifer** - 'n_estimators': 115, 'min_samples_split': 6, 'min_samples_leaf': 5, 'max_features': 'auto', 'max_depth': 13, 'criterion': 'entropy'

- **LR classfifer** - 'solver': 'newton-cg', 'penalty': 'l2', 'max_iter': 5000, 'C': 0.012742749857031334

4.4 Performance

4.4.1 Classification Report

	Precision	Recall	f1-score	support
0	0.82	0.90	0.86	930
1	0.78	0.65	0.71	520
accuracy			0.81	1450
macro avg	0.80	0.78	0.79	1450
weighted avg	0.81	0.81	0.81	1450

Table 2: Performance Scores of test data

4.4.2 Confusion Matrix

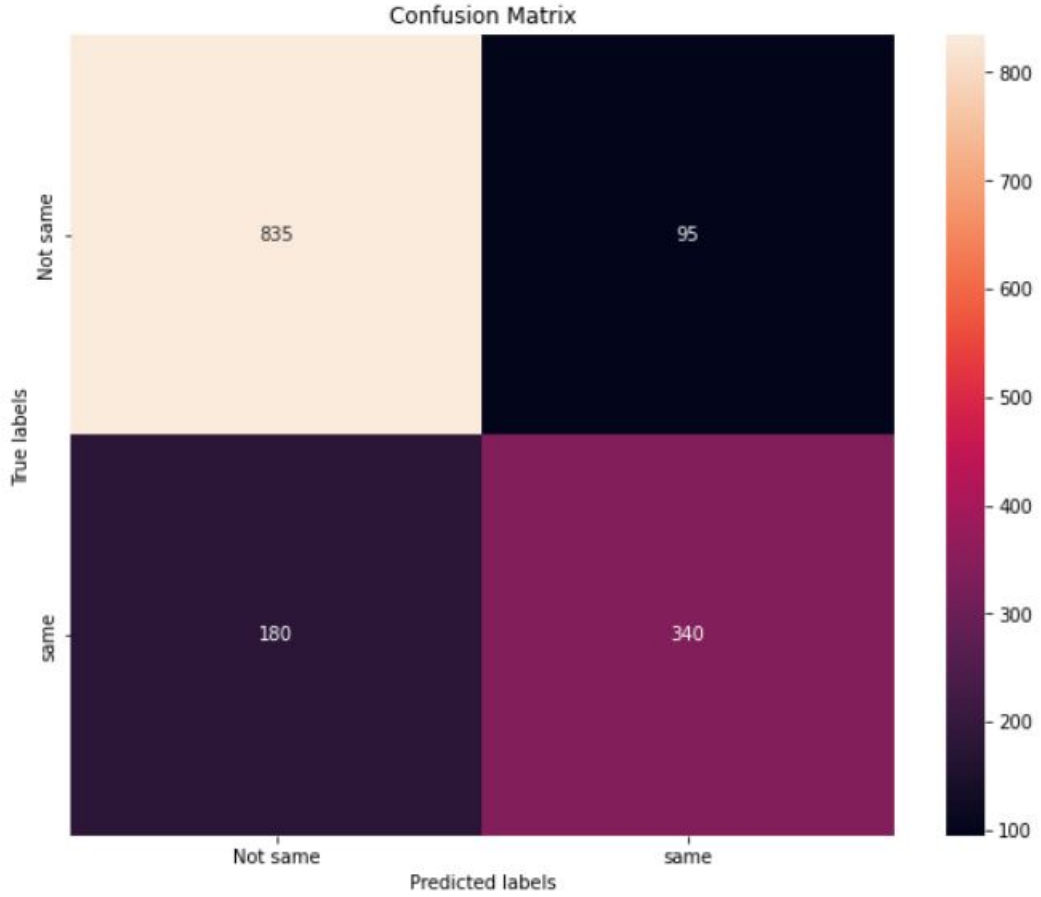


Figure 4.2: Confusion Matrix for testing data

5 Contribution Statement

This section details each member's contribution as follows:

- **Prahlad Narayanan** worked on building the model and found the dataset to be used for the project
- **Rohan Saswade** worked on feature visualization and the Exploratory Data Analysis
- **Hari Ranjan Meena** worked on coding the model and testing
- **G. Saicharan** worked on hyperparameter optimization and feature visualization
- **Ankit Srivastava** worked on coding the model.