Practical Machine Learning

Project 1

Documentation

In this project, I have approached 2 methods, Logistic Regression and K-Nearest Neighbors in order to predict the sentiment of tweets from a dataset.

About dataset

The dataset consists 3 columns, “tweet_id”, having a unique value for each tweet, “sentiment” with 13 different attributes, and “content” with the respective text. The dataset is unbalanced, having the following distribution: neutral - 8638, worry - 8459, happiness – 5209, sadness- 5165,love - 3842, surprise – 2187, fun - 1776, relief – 1526, hate – 1323, empty – 827, enthusiasm – 759, boredom - 179, anger – 110, 40000 in total.

It is important to mention that the dataset was downsampled to include only five classes.

In addition, all the tweets were normalized with a function, which removes the urls, the user references, punctuation, lowercase the text, tokenize and uses stemming.

The dataset was split in training and test sets to evaluate the model on unseen data.

1. Logistic Regression

Logistic Regression transforms the linear combination of input features through the sigmoid function, which output is represented by a probability between 0 and 1.

There were 2 grid searches performed with 2 different type of features, one with TF-IDF and one with Bag of Words, 2 methods to represent in a different way the data.

Parameter	Values
ngram_range	(1, 1), (1, 2)
C	0.1, 0.2, 0.3, 0.5, 1, 2
solver	'lbfgs', 'saga', 'newton-cg'
class_weight	'balanced'
max_iter	3500

N gram range representing the range of n-values for different n-grams to be extracted, “C” meaning that smaller values specify stronger regularization, “Class_weight = balanced” for the imbalanced data, adjusting weights inversely proportional to class frequencies in the input data, “Max_iter = 3500” for the maximum number of iterations taken for the solvers to converge,

The first grid search was performed using bow and these parameters:

score	ngram_range	C	class_weight	max_iter	solver
0.448111	1,1	0.1	balanced	3500	lbfgs
0.448946	1,1	0.1	balanced	3500	saga
0.448156	1,1	0.2	balanced	3500	lbfgs
0.447646	1,1	0.2	balanced	3500	saga
0.447855	1,1	0.3	balanced	3500	lbfgs
0.446949	1,1	0.3	balanced	3500	saga
0.442402	1,1	0.5	balanced	3500	lbfgs
0.442563	1,1	0.5	balanced	3500	saga
0.425941	1,1	1	balanced	3500	lbfgs
0.426263	1,1	1	balanced	3500	saga
0.415584	1,1	2	balanced	3500	lbfgs
0.41545	1,1	2	balanced	3500	saga
0.447046	1,2	0.1	balanced	3500	lbfgs
0.448834	1,2	0.1	balanced	3500	saga
0.441904	1,2	0.2	balanced	3500	lbfgs
0.443018	1,2	0.2	balanced	3500	saga
0.441002	1,2	0.3	balanced	3500	lbfgs
0.441188	1,2	0.3	balanced	3500	saga
0.433913	1,2	0.5	balanced	3500	lbfgs
0.434044	1,2	0.5	balanced	3500	saga
0.427257	1,2	1	balanced	3500	lbfgs
0.427918	1,2	1	balanced	3500	saga
0.423113	1,2	2	balanced	3500	lbfgs
0.423595	1,2	2	balanced	3500	saga

The second grid search was performed with TF-IDF

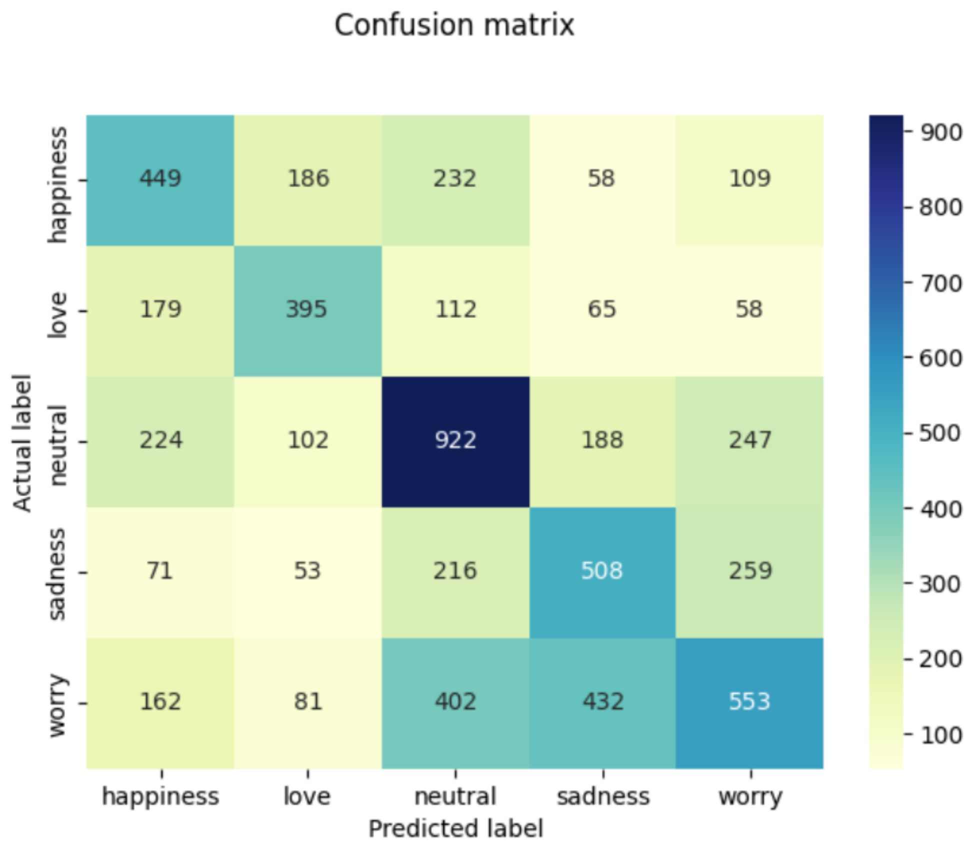
score	C	class_weight	max_iter	solver	ngram_range
0.438016	0.1	balanced	3500	lbfgs	1,1
0.43627	0.1	balanced	3500	lbfgs	1,2
0.438071	0.1	balanced	3500	saga	1,1
0.436654	0.1	balanced	3500	saga	1,2
0.44604	0.2	balanced	3500	lbfgs	1,1
0.441862	0.2	balanced	3500	lbfgs	1,2
0.446183	0.2	balanced	3500	saga	1,1
0.441996	0.2	balanced	3500	saga	1,2
0.44679	0.3	balanced	3500	lbfgs	1,1
0.444103	0.3	balanced	3500	lbfgs	1,2
0.446924	0.3	balanced	3500	saga	1,1
0.443847	0.3	balanced	3500	saga	1,2
0.442152	0.5	balanced	3500	lbfgs	1,1
0.443512	0.5	balanced	3500	lbfgs	1,2
0.44202	0.5	balanced	3500	saga	1,1
0.443319	0.5	balanced	3500	saga	1,2
0.440888	1	balanced	3500	lbfgs	1,1
0.439562	1	balanced	3500	lbfgs	1,2
0.440731	1	balanced	3500	saga	1,1
0.439409	1	balanced	3500	saga	1,2
0.429415	2	balanced	3500	lbfgs	1,1
0.43174	2	balanced	3500	lbfgs	1,2
0.42944	2	balanced	3500	saga	1,1
0.431515	2	balanced	3500	saga	1,2

The conclusion would be that the highest score, macro F1 score, was obtained with BoW, 0.448 and with TF-IDF, 0.446.

The classification report for the parameters: ngram_range = (1,1), C = 0.1 and solver = “saga” is:

	Precision	recall	F1-score
Happines	0.41	0.43	0.42
Love	0.48	0.49	0.49
Neutral	0.49	0.55	0.52
Sadness	0.41	0.46	0.43
Worry	0.45	0.34	0.39
Accuracy			0.45
Macro avg	0.45	0.45	0.45
Weighted avg	0.45	0.45	0.45

The confusion matrix is:



Overall, the model performs well in identifying ‘neutral’ sentiments but shows some confusion between emotions that may share similar language or context.

2. K-Nearest Neighbor

K-Nearest Neighbor works by finding the “k” training samples closest in distance to a new point and predict the label based on these neighboring samples.

There were another 2 grid searches performed with 2 different type of features, one with TF-IDF and one with bag of words.

The first grid search was performed using bow and these parameters:

Parameter	Values
ngram_range	(1, 1), (1, 2)
N_neighbors	3,5,7
metric	‘euclidean’,’manhattan’,’minkovski’
weights	‘uniform’,’distance’

‘N_neighbors’ is a core parameter, which specifies the number of neighbors to use for the k-nearest neighbors voting. ‘Metric’ defines the distance metric, ‘weights’ specifies the weight function used in prediction, ‘uniform’ means all points in each neighborhood are weighted equally, ‘distance’ means weighting points by the inverse of their distance.

The first grid search was performed with BoW representation:

score	**ngram_range**	**metric**	**n_neighbors**	**weights**
0.285455	1-1	euclidean	3	uniform
0.296341	1-1	euclidean	3	distance
0.28537	1-1	euclidean	5	uniform
0.289871	1-1	euclidean	5	distance
0.269691	1-1	euclidean	7	uniform
0.279015	1-1	euclidean	7	distance
0.290379	1-1	manhattan	3	uniform
0.2968	1-1	manhattan	3	distance
0.282112	1-1	manhattan	5	uniform
0.285973	1-1	manhattan	5	distance
0.285455	1-1	minkowski	3	uniform
0.296341	1-1	minkowski	3	distance
0.28537	1-1	minkowski	5	uniform
0.289871	1-1	minkowski	5	distance
0.269691	1-1	minkowski	7	uniform
0.279015	1-1	minkowski	7	distance
0.269721	1-2	euclidean	3	uniform

0.28363	1-2	euclidean	3	distance
0.242076	1-2	euclidean	5	uniform
0.254267	1-2	euclidean	5	distance
0.234613	1-2	euclidean	7	uniform
0.241697	1-2	euclidean	7	distance
0.272507	1-2	manhattan	3	uniform
0.285262	1-2	manhattan	3	distance
0.247394	1-2	manhattan	5	uniform
0.25485	1-2	manhattan	5	distance
0.225855	1-2	manhattan	7	uniform
0.234592	1-2	manhattan	7	distance
0.269721	1-2	minkowski	3	uniform
0.28363	1-2	minkowski	3	distance
0.242076	1-2	minkowski	5	uniform
0.254267	1-2	minkowski	5	distance
0.234613	1-2	minkowski	7	uniform
0.241697	1-2	minkowski	7	distance

The second grid search was performed using TF-IDF vectorization:

score	**ngram_range**	**metric**	**n_neighbors**	**weights**
0.185751	1-1	euclidean	3	uniform
0.150323	1-2	euclidean	3	uniform
0.191733	1-1	euclidean	3	distance
0.154638	1-2	euclidean	3	distance
0.172134	1-1	euclidean	5	uniform
0.135449	1-2	euclidean	5	uniform
0.177575	1-1	euclidean	5	distance
0.139135	1-2	euclidean	5	distance
0.158776	1-1	euclidean	7	uniform
0.127755	1-2	euclidean	7	uniform
0.161865	1-1	euclidean	7	distance
0.132203	1-2	euclidean	7	distance
0.104452	1-1	manhattan	3	uniform
0.088887	1-2	manhattan	3	uniform
0.105519	1-1	manhattan	3	distance
0.08978	1-2	manhattan	3	distance
0.10116	1-1	manhattan	5	uniform
0.088101	1-2	manhattan	5	uniform
0.104184	1-1	manhattan	5	distance
0.089905	1-2	manhattan	5	distance
0.099854	1-1	manhattan	7	uniform
0.087763	1-2	manhattan	7	uniform
0.102247	1-1	manhattan	7	distance
0.08952	1-2	manhattan	7	distance

0.185751	1-1	minkowski	3	uniform
0.150323	1-2	minkowski	3	uniform
0.191733	1-1	minkowski	3	distance
0.154638	1-2	minkowski	3	distance
0.172134	1-1	minkowski	5	uniform
0.135449	1-2	minkowski	5	uniform
0.177575	1-1	minkowski	5	distance
0.139135	1-2	minkowski	5	distance
0.158776	1-1	minkowski	7	uniform
0.127755	1-2	minkowski	7	uniform
0.161865	1-1	minkowski	7	distance
0.132203	1-2	minkowski	7	distance

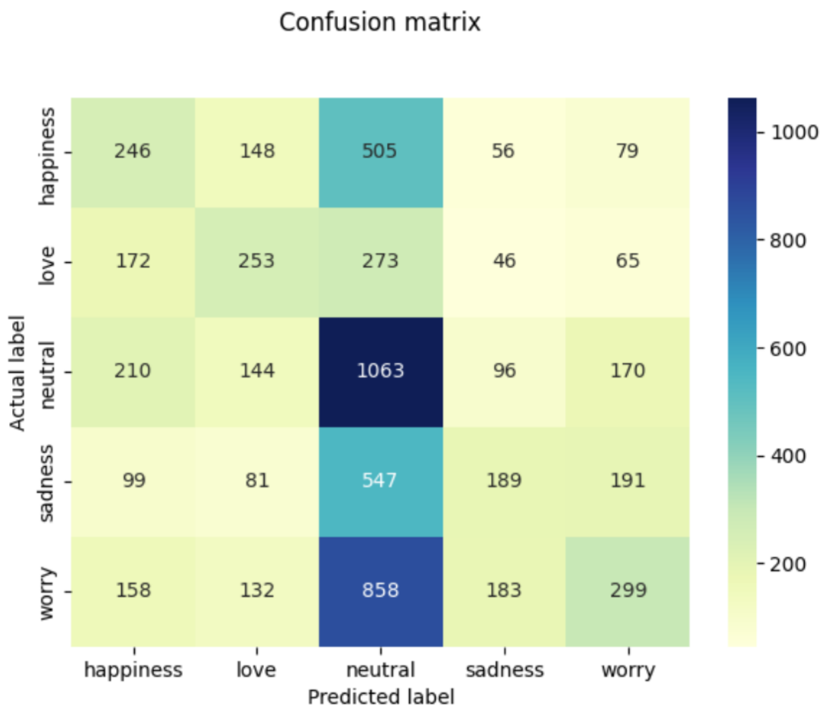
The conclusion would be that BoW consistently outperformed the TF-IDF across various configurations, the best performance with BoW was observed with the configuration using the ‘manhattan’ metric, ‘neighbors’ = 3, ‘weights’ = distance, the macro F1 score was 0.2968, while using TF-IDF, the best score was lower, approximately 0.191.

Using the optimal hyperparameters, the following classification report is:

	Precision	recall	F1-score
Happines	0.28	0.24	0.26
Love	0.33	0.31	0.32
Neutral	0.33	0.63	0.43
Sadness	0.33	0.17	0.23
Worry	0.37	0.18	0.25
Accuracy			0.33
Macro avg	0.33	0.31	0.30
Weighted avg	0.33	0.33	0.30

The model performs best in identifying ‘neutral’ emotions, it struggles with ‘sadness’ and ‘happiness’, which could be due to various factors like class imbalance, a strategy in order to improve might include resampling the dataset to balance the class distribution.

The confusion matrix for the following configuration is:



A point to mention is that the matrix shows ‘neutral’ is the most misclassified class and it has the highest number of correct predictions, 1063.

Conclusion

The results demonstrate that Logistic Regression significantly outperforms K-Nearest Neighbors in the domain of sentiment analysis, highlighting the potential superiority of linear models for the processed dataset. Despite this, there remains a considerable opportunity to enhance the overall performance of the models.