Humor Detection using Support Vector Machine

Marina Pinho Garcia[1], **Giovana Pinho Garcia**[2], **Dr. Nádia Félix Felipe da Silva**[3]

[1]Faculdade de Tenologia
Universidade de Brasília (UnB)

[2]Departamento de Ciência da Computação
Universidade de Brasília (UnB)

{ninapgarcia.unb,giopgunb,nadia.felix}@gmail.

Abstract. *This paper aims classify texts in humorous and non-humorous, while exploring the different parameters and tactics that can be used alongside the Support Vector Machine (SVM) classifier, to see and understand their impact on the classification and find the best combinations that have the best performances considering the accuracy and the F1 score. After observing the plots and analyzing the data we were able to come to a conclusion of which combination would be best to classify the texts in the testing data provided by the* HaHackathon: Detecting and Rating Humor and Offense *CodaLab Competition [cod 2021]. With those results we were able to give a wide view of this type of problem solutions, which can be used in further related work in this field of research.*

1. Intoduction

Currently, we have a huge amount of texts that are available online, and the process of text classification done by humans started demanding a lot of time and effort. Due to that, now we have various ways of automatic text classification approaches to process the information [Berry 2003].

Text classification is the activity of labeling natural language texts with relevant categories from a predefined set. Typically, an automatic classifier has to learn from already labeled data, so that it can, later, automatically predict the category of unlabeled data.

In this paper, our goal was to predict if the text would be considered humorous (for an average user). This is a binary task that returns 1 if the text is considered humorous and 0 if not. Humor poses a linguistic challenge to natural language processing, for been a figurative language and highly subjective, depending on age, gender, and culture. With that, our objective will be to classify if the text was intended to be humorous.

The method that we used to solve this problem was SVM (Support Vector Machine). Which is an algorithm that finds a hyperplane in an N-dimensional space that classifies the data points. Where N would be the number of features that we use, and hyperplanes would be decision boundaries that helps classify the data points.[tow 2021].

The purpose of our work is to classify texts in humorous and non-humorous using SVM algorithm and fine tuning their parameters to see which combination would be the most fitting and have the highest accuracy and F1 score for the testing data given

by the *HaHackathon: Detecting and Rating Humor and Offense* CodaLab Competition [cod 2021]. Thereby, we than tried different combinations with the use of stopwords removal and stemming[Al-Khafaji H. 2017], the use of unigrama and bigrama and the use of different kernels.

2. Related Work

Given the amount of texts we have available nowadays, and the growing necessity of efficient text classifiers to facilitate and reduce the human efforts, studies that test that efficiency and compare different combinations and different classifiers are more necessary every day.

In [Sun and Liu 2009] [Zhuang and Chen 2005] studies, they used SVM classifier to solve problems in the context of imbalanced classification. The former chose the SVM for its good classification accuracy reported in many classification tasks, and they compared the effectiveness of many strategies, this same approach will be used in this paper. The latter based their method on a novel extension of the proximal SVM mode, afterwards they compared the performance of their method, the original SVM mode and the standard SVM.

Furthermore, in the study of [Xu and Wang 2003], the objective was to explore the clustering structure of uncertain documents and identify the representative samples to collect the user opinions, with the goal of speeding up the process of convergence of SVM classifiers. With that, the study compared representative sampling with random sampling and SVM active learning.

Moreover, there are also studies that investigates automatic humor detection, as will be done here. For example, in the paper [Bali T. and N. 2018] they approach classification of humor based on classical theories, like Script-based Semantic Theory of Humor and General Verbal Theory of Humor, with a few improvements and revisions.

3. Problem Definition

In this work, the problem that we aimed to solve was provided by the *HaHackathon: Detecting and Rating Humor and Offense* CodaLab Competition [cod 2021]. This competition gave us the training data already labeled, the testing data and the goal of our program, which was to classify texts in humorous(1) or non-humorous(0).

A couple of examples given were "TENNESSEE: We're the best state. Nobody even comes close. *Elevennessee walks into the room* TENNESSEE: Oh shit..." which was classified as humorous and "I got REALLY angry today and it wasn't about nothing, but you're going to have to take my word for that." which was classified as non-humorous.

With that, the main focus of our work was to test the different parameters and tactics to use with SVM classifier algorithm to find which of the combinations would be better at classifying the texts of the testing data provided and to understand better how each of them work and the impact they have on the classification. Thus, we first explored different combinations with the use of stopwords removal and stemming, the use of unigrama and bigrama and the use of different kernels(utilizing the default values of each kernel). After analyzing that data we acquired, we than proceeded to pick the combinations that had the highest accuracy and F1 score to test some other parameters, so for each kernel we changed the decision function shape, the gamma and the c.

4. Methods

In Figure 1 we can see the steps taken in our experiments to classify the dataset texts in humorous and non-humorous, while exploring different parameters and tactics. In the following subsections each step will be explained in detail.

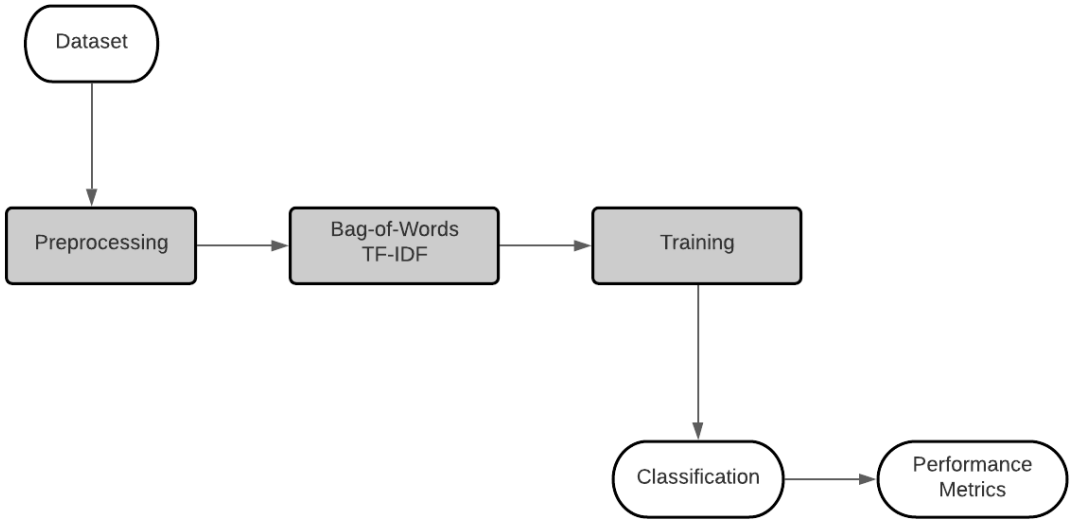

Figure 1. Process Flowchart

4.1. Dataset

The dataset used in our experiments was the one provided by the *HaHackathon: Detecting and Rating Humor and Offense* CodaLab Competition [cod 2021]. Were provided 9000 short texts, divided in 8000 for training and the remaining 1000 for the testing. All the texts were labeled, with the following [is_humor, humor_rating, humor_controversy, offense_rating], although, in this experiment, only the *is_humor* classification label was used.

4.2. Libraries and Environment

This experiment was carried out using the *Google Colaboratory* platform with the *Python 3* programming language. The following libraries were used:

- Pandas [pan 2021], for the data manipulation;
- NLTK [nlt 2021], for the tokenization and stemming in the preprocessing stage;
- Scikit Learn [skl 2021], for the bag of words and training process;
- Numpy [num 2021], for matrix operations.

4.3. Preprocessing

For this experiment we performed the preprocessing of the text in four different ways: 'Only Stemming', 'Stemming and StopWords Removal', 'Only StopWords Removal' and none (just the raw text tokenized).

Before getting started with the preprocessing, we tokenize the text. For the Stemming was used the *stem.snowball.SnowballStemmer* function from the *nltk* library. The

StopWords Removal was performed using the *corpus.stopwords.words('english')*, also from *nltk* [nlt 2021], set of words, punctuation was also added to that set of words for its removal.

The preprocessing was performed in both the training and testing dataset.

4.4. Bag of Words: TF-IDF

After performing the preprocessing of all the texts, we generate the bag of words. For this part we used 3 different methods to create this bag of words: 'Unigrams', 'Bigrams' and 'Unigrams and Bigrams'. Those were made varying the ngram_range parameter [(1,1), (2,2), (1,2)] respectively in the function *feature_extraction.text.TfidfVectorizer* from the *sklearn* library [skl 2021].

The bag of words was generated using the training and testing datasets combined, so they would have the same number of features, and then for the training, they were again separated.

4.5. Training

To perform the training we used the Support Vector Machine Algorithm for classification from the *sklearn* library, *svm.SVC*. In the first set of test, were only used the default parameters for each kernel [linear, rbf, polynomial, sigmoid]. Then for the second round of tests, we selected some parameters to vary in the combinations that had the best performances.

4.6. Performance Metrics

For this experiment we opted for the most commonly-used performance measures, those beeing *Accuracy (acc), Precision (prec), Recall* and *F1 score (F1)*, wich are calculated as shown in the equations 1, 2, 3 and 4 respectively.

$$acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$prec = \frac{TP}{TP + FP} \tag{2}$$

$$recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = \frac{2 * prec * recall}{prec + recall} \tag{4}$$

where TP = true positive, FP = false positive, TN = true negative and FN = false negative (table 1)

The Accuracy measure refers to the percentage of texts that were correctly classified, Precision talks about how accurate your model is out of those predicted positive, how many of them are actual positive, whereas the recall calculates how many

Table 1. Explanation of True/False Positive and Negative

		Predicted	
		negative	positive
Actual	negative	true negative	false positive
	positive	false negative	true positive

of the Actual Positives our model capture through labeling it as Positive (true Positive). Finally, the F1 score is the harmonic mean of precision and recall.

5. Experiments

The purpose of this experiment is to compare the results of several preprocessing methods and bag-of-words 'ngram ranges' for different training parameters. And so we separated the tests into two rounds.

5.1. First Round

For this first round of tests, we chose 4 different preprocessing methods, those being 'Only Stemming', 'Only Stopwords Removal', 'Stopwords Removal and Stemming' and none, as explained in the Section 4.3. For each one of these 4 preprocessing methods, 3 different TF-IDF bag of words were used: 'Unigrams', 'Bigrams' and 'Unigrams and Bigrams', explained in Section 4.4.

And finally, the training and testing were performed for each category cited above for the default parameters of the *svc.SVM* function, varying only the kernel as shown in Section 4.5.

5.2. Second Round

As for this second round of tests, we selected the combinations that had the best performance, when analyzing the accuracy and F1 score, to vary other parameters. Three new parameters were chosen to be varied: 'Decision Function Shape', 'Gamma' and 'C'.

The Decision Function Shapes are OVO (one-vs-one) and OVR (one-vs-rest), the Gamma parameter vary from 'auto' to 'scale' and as for the C parameter, it was chosen from the following values [0.01, 0.1, 0.5, 1, 2, 5, 10, 100].

6. Results

In Table 2 we can see the results from the first round of tests and analyzing them it is possible to see that the ones that achieved the best performances were the ones with only stemming and no preprocessing, and from those, the bag of words with only 'unigrams' had the highest scores, probably due to the fact that we were able to train them with much more data, 8000 texts, in contrast with the 2000 used in 'bigrams' and 'bigrams and unigrams'. And as for the kernels, the only one not chosen to integrate the second round of tests was the 'polynomial'.

In Tables 3 and 4 we have the results obtained in the second round, and with these it is finally possible to see that the highest values of accuracy and F1 score are in the 'Only

Table 2. First Round Results

		Linear				Polynomial				RBF				Sigmoid			
		Acc	Prec	Recall	F1	Acc	Prec	Recall	F1	Acc	Prec	Recall	F1	Acc	Prec	Recall	F1
No stemming or stopwords removal	Unigrams	82.19	83.28	89.87	86.45	68.6	66.84	99.84	80.07	80.2	80.64	90.34	85.22	72	70.65	95.25	81.13
	Bigrams	82.3	83.4	89.87	86.51	63.5	63.39	100	77.59	64.1	63.8	99.84	77.85	70.39	68.87	96.99	80.55
	Unigrams and Bigrams	76.6	77.71	88.29	82.66	63.5	63.39	100	77.59	74.2	71.79	97.46	82.68	76.8	77.7	88.76	82.86
With stopwords removal	Unigrams	80.9	82.47	88.61	85.43	66.7	65.52	99.84	79.12	80.7	80.44	91.77	85.74	80.2	82.01	87.97	84.89
	Bigrams	65.4	64.71	99.53	78.43	63.3	63.26	100.0	77.50	63.7	63.54	99.84	77.66	64.4	64.00	99.84	78.00
	Unigrams and Bigrams	77.3	77.25	90.82	83.49	63.3	63.26	100.0	77.50	69.1	67.35	99.21	80.23	76.5	76.02	91.77	83.15
With stemming	Unigrams	81.6	82.46	90.03	86.08	71.5	69.05	99.53	81.53	80.80	81.07	90.82	85.67	81.10	82.05	89.72	85.71
	Bigrams	72.8	71.48	94.78	81.50	63.5	63.39	100.0	77.59	64.2	63.87	99.84	77.90	71.40	69.53	97.47	81.16
	Unigrams and Bigrams	77.7	78.44	89.24	83.49	63.5	63.39	100.0	77.59	75.4	72.87	97.31	83.33	77.6	78.25	89.40	83.45
With stopwords removal and stemming	Unigrams	78.7	80.05	88.29	83.97	68.60	66.84	99.84	80.07	80.1	79.06	93.19	85.54	79	80.4	88.29	84.16
	Bigrams	65.3	64.61	99.68	78.4	63.3	63.26	100	77.49	63.7	64.54	99.84	77.66	64.5	64.06	99.84	78.04
	Unigrams and Bigrams	76	75.65	91.45	82.80	63.4	63.32	100	77.54	68.6	66.91	99.52	80.02	75.4	74.55	92.72	82.65

Stemming' table 4, with Gamma set as 'Scale', the kernel 'RBF' and C = [5, 10, 100] (for both OVO and OVR decision function shapes).

Table 3. Second Round Results

No StopWords removal or Stemming

Decision Function Shape	Gamma	C	Linear				RBF				Sigmoid			
			Acc	Prec	Recall	F1	Acc	Prec	Recall	F1	Acc	Prec	Recall	F1
One-vs-Rest	auto	0.01	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.1	75.70	74.04	94.77	83.13	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.5	80.60	82.30	88.29	85.19	72.80	78.03	79.27	78.64	65.80	70.08	80.06	74.74
		1	82.19	83.28	89.87	86.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		2	80.70	82.32	88.44	85.27	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		5	80.30	81.65	88.76	85.06	64.30	63.90	100.00	77.97	63.20	63.20	100.00	77.45
		10	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		100	78.90	82.33	84.81	83.55	77.30	79.64	86.07	82.73	76.20	78.63	85.60	81.96
	scale	0.01	76.00	78.32	85.75	81.87	76.00	78.32	85.75	81.87	63.40	63.32	100.00	77.54
		0.1	75.70	74.04	94.77	83.13	63.40	63.32	100.00	77.54	75.60	73.95	94.77	83.07
		0.5	80.40	81.87	88.60	85.10	78.80	78.37	91.77	84.54	66.00	73.47	72.31	72.88
		1	82.19	83.28	89.87	86.45	80.20	80.64	90.34	85.22	72.00	70.65	95.25	81.13
		2	80.70	82.32	88.44	85.27	81.39	82.41	89.71	85.90	65.70	73.41	71.67	72.53
		5	80.30	81.65	88.76	85.06	81.69	82.67	89.87	86.12	65.50	73.18	71.67	72.42
		10	79.40	82.07	86.23	84.10	81.69	82.67	89.87	86.12	78.10	80.95	85.44	83.14
		100	78.90	82.33	84.81	83.55	81.69	82.67	89.87	86.12	74.50	79.59	80.22	79.90
One-vs-One	auto	0.01	78.80	81.15	86.55	83.76	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.1	80.00	81.76	87.97	84.75	72.00	76.34	80.69	78.46	63.20	63.20	100.00	77.45
		0.5	80.60	82.30	88.29	85.19	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		1	82.19	83.28	89.87	86.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		2	81.69	83.06	89.24	86.04	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		5	80.40	82.34	87.81	84.99	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		10	79.40	82.07	86.23	84.10	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		100	78.90	82.33	84.81	83.55	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
	scale	0.01	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.1	75.70	74.04	94.77	83.13	63.40	63.32	100.00	77.54	75.60	73.95	94.77	83.07
		0.5	80.40	81.32	89.55	85.24	78.80	78.37	91.77	84.54	80.20	81.08	89.55	85.11
		1	82.19	83.28	89.87	86.45	80.20	80.64	90.34	85.22	82.30	83.40	89.87	86.51
		2	81.69	83.06	89.24	86.04	81.39	82.41	89.71	85.90	82.39	83.62	89.71	86.56
		5	80.40	82.34	87.81	84.99	81.89	82.67	89.87	86.12	80.90	82.87	87.97	85.34
		10	79.40	82.07	86.23	84.10	81.89	82.67	89.87	86.12	78.10	80.95	85.44	83.14
		100	78.90	82.33	84.81	83.55	81.69	82.67	89.87	86.12	74.50	79.59	80.22	79.90

For a more specific analysis we have the plots shown in Figures 2, 3, 4 and 5. The first 2 one is the comparison of the F1 Score for each Stemming-OVR-Scale combination, and 3 is the comparison of the accuracy of this same combination. As for figures 4 and 5, follow the same pattern, though for the 'no processing' group.

Observing the plots, we can once more conclude the same as above, that the best performances occur with the Stemming-Scale-RBF combination with C as [5, 10, 100].

Table 4. Second Round Results

Only Stemming

Decision Function Shape	Gamma	C	Linear				RBF				Sigmoid			
			Acc	Prec	Recall	F1	Acc	Prec	Recall	F1	Acc	Prec	Recall	F1
One-vs-Rest	auto	0.01	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.1	77.80	75.95	94.94	84.39	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.5	80.30	81.12	89.72	85.20	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		1	81.60	82.46	90.03	86.08	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		2	80.90	82.19	89.08	85.50	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		5	80.70	82.42	88.29	85.26	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		10	79.90	82.31	86.87	84.53	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		100	77.90	81.96	83.39	82.67	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
	scale	0.01	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.1	77.80	75.95	94.94	84.39	64.20	63.84	100.00	77.93	77.60	75.82	94.78	84.25
		0.5	80.30	81.12	89.72	85.20	79.30	78.60	92.41	84.95	80.60	81.20	90.19	85.46
		1	81.60	82.46	90.03	86.08	80.80	81.07	90.82	85.67	81.10	82.05	89.72	85.71
		2	80.90	82.19	89.08	85.50	82.30	82.45	91.46	86.72	81.30	82.86	88.77	85.71
		5	80.70	82.42	88.29	85.26	82.30	82.64	91.14	86.62	80.70	82.23	88.61	85.30
		10	79.90	82.31	86.87	84.53	82.30	82.64	91.14	86.68	78.80	80.79	87.18	83.67
		100	77.90	81.96	83.39	82.67	82.30	82.64	91.14	86.68	76.50	80.49	82.91	81.68
One-vs-One	auto	0.01	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.1	77.80	75.95	94.94	84.39	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.5	80.30	81.12	89.72	85.20	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		1	81.60	82.46	90.03	86.08	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		2	80.90	82.19	89.08	85.50	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		5	80.70	82.42	88.29	85.26	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		10	79.90	82.31	86.87	84.53	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		100	77.90	81.96	83.39	82.67	79.30	80.66	88.45	84.38	63.20	63.20	100.00	77.45
	scale	0.01	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45	63.20	63.20	100.00	77.45
		0.1	77.80	75.95	94.94	84.39	64.20	63.84	100.00	77.93	77.60	75.82	94.78	84.25
		0.5	80.30	81.12	89.72	85.20	79.30	78.60	92.41	84.95	80.60	81.20	90.19	85.46
		1	81.60	82.46	90.03	86.08	80.80	81.07	90.82	85.67	81.10	82.05	89.72	85.71
		2	80.90	82.19	89.08	85.50	82.30	82.45	91.46	86.72	81.30	82.87	88.77	85.71
		5	80.70	82.42	88.29	85.26	82.30	82.64	91.14	86.68	80.70	82.23	88.61	85.30
		10	79.90	82.31	86.87	84.53	82.30	82.64	91.14	86.68	78.80	80.79	87.18	83.67
		100	77.90	81.96	83.39	82.67	82.30	82.64	91.14	86.68	76.50	80.49	82.91	81.68

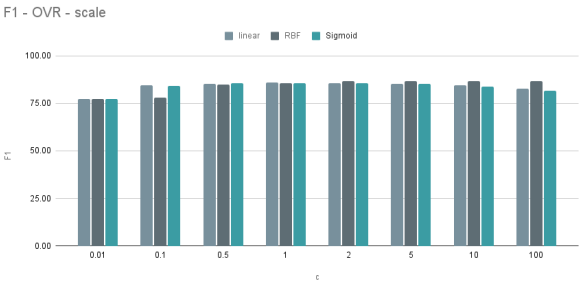

Figure 2. Only Stemming F1 score

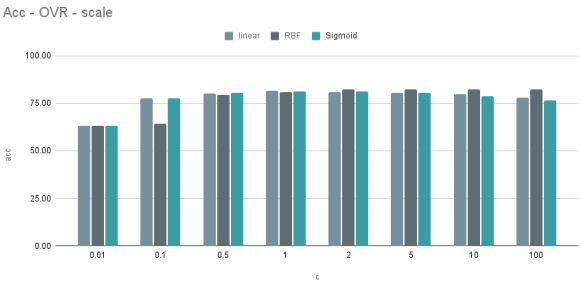

Figure 3. Only Stemming accuracy

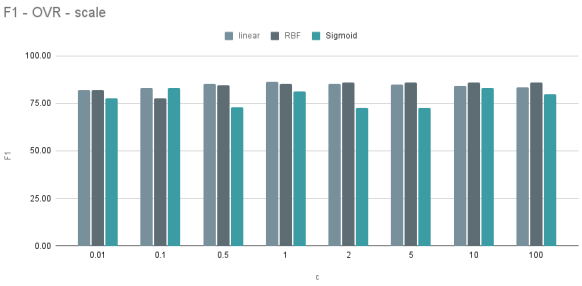

Figure 4. No Preprocessing F1 score

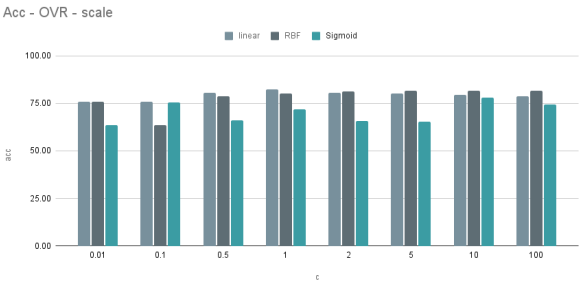

Figure 5. No Preprocessing accuracy

7. Conclusion and Future Work

In this paper we attempt to investigate the parameter and preprocessing combinations of an SVM training algorithm for best classifications of short texts in humorous and non-humorous categories. Through the tests performed we were able to have a wide view of how each combination behave, and which ones are better for the classification of this text format. As shown in 6 the best performances occur with the Stemming-Scale-RBF combination with C as [5, 10, 100]. And though these tests can get a lot more directed and refined, our work results might provide a good starting point for several researches in this field of study.

Based on what was achieved, future work may lean in the direction of obtaining better performances by focusing on more specific and refined strategies using language models.

References

[cod 2021] (2021). Hahackathon:detecting and rating humor and offense codalab competition. https://competitions.codalab.org/competitions/27446. Accessed 25 May 2021.

[nlt 2021] (2021). Nltk library documentation. https://www.nltk.org/api/nltk.html. Accessed 25 May 2021.

[num 2021] (2021). Numpy library documentation,. https://numpy.org/doc/stable/. Accessed 25 May 2021.

[pan 2021] (2021). Pandas library documentation. https://pandas.pydata.org/docs/. Accessed 25 May 2021.

[skl 2021] (2021). Scikit-learn library documentation. https://scikit-learn.org/stable/. Accessed 25 May 2021.

[tow 2021] (2021). Support vector machine — introduction to machine learning algorithms. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47. Accessed 25 May 2021.

[Al-Khafaji H. 2017] Al-Khafaji H., H. A. (2017). Efficient algorithms for preprocessing and stemming oftweets in a sentiment analysis system.

[Bali T. and N. 2018] Bali T., A. V. and N., S. (2018). What makes us laugh? investigations into automatichumor classification.

[Berry 2003] Berry, M. (2003). Survey of text mining: Clustering, classification and retrieval.

[Sun and Liu 2009] Sun, A., L. E. and Liu, Y. (2009). On strategies for imbalanced text classification using svm:a comparative study.

[Xu and Wang 2003] Xu, Z., Y. K. T. V. X. X. and Wang, J. (2003). Representative sampling for text classi-fication using support vector machines.

[Zhuang and Chen 2005] Zhuang, D., Z. B. Y. Q. Y. J. C. Z. and Chen, Y. (2005). Efficient text classifi-cation by weighted proximal svm.