A comparative analysis of various machine learning techniques for trend prediction on social media

Diana Lucaci
University of Ottawa
Ottawa, Canada
diana.lucaci22@gmail.com

***Abstract*—Early detection of the topics that may gain the public's attention on social media has a number of benefits for persons, authorities, and businesses, allowing them to prepare to act accordingly. For this reason, an extensive analysis of the social media data using machine learning is essential to process a large amount of data and to predict whether a certain subject will gain interest. We analyze the posts that evoke intense interaction on Twitter, using a dataset with numerical features that contains a number of 11 different statistics about users' activity. To benefit from both the insightful information available and the prediction power of machine learning algorithms, we present an ample report that shows the results obtained after performing various experiments. The paper covers details regarding the dataset, preprocessing techniques, feature selection, and model selection that lead to obtaining results surpassing the baseline.**

***Index Terms*—Social media, data mining, algorithms, classification, optimization**

I. Introduction

Social media has expanded rapidly in recent times, becoming one of the richest sources of information that can be explored and used to improve peoples well-being in many different directions such as detecting mental health problems, changes in behaviours, rumours or trends, thus having impact on areas such as industry, health care, business, politics, and security. Since there is such a rich information resource available, drawing useful insights from it becomes possible when using the computational power and the machine learning algorithms capable of managing and effectively using large amount of data.

Predicting whether a certain topic will become popular in the near future based on the activity of the past few days has a number of advantages. Firstly, it would help inform people about future trends so that they can prepare and act accordingly. Secondly, it can help authorities prepare for major events or consequences of the topics that are being intensively discussed online such as going on strikes, parades, demonstrations. Along with this, businesses can be informed about the latest trends and can predict and prepare for high sales growth.

The motivation behind this project is to not only analyze the performance of different machine learning techniques practically but also to understand what are the advantages and disadvantages of different algorithms for social media data. Moreover, gathering insights about the features that are most relevant for discovering trends and meaningful information from twitter data can help in focusing on the most promising ones that can be later extracted from the text. Applying the insights gathered not only from a technical point of view (algorithms and developing programming skills) but also from the meaning of the attributes that are used to draw conclusions on a decision, would accelerate the prediction power of the end system.

The report presents briefly the importance of this task in section II, followed by a description of the data set that has been used in this report, in section III. The next two sections describe the previous work and results and a thorough analysis of the dataset of discussion. For the experiments, we have described the environmental setup in section VI, the preprocessing steps considered, a description of the evaluation criteria, brief descriptions of the used algorithms and the results in sections VII, VIII, and IX respectively. A comparison between the algorithms is performed and described in section X, followed by concluding remarks, key learnings and future work in sections XI, XII, XIII, and XIV.

II. Problem domain

Social media is one of the networks that help propagate stories faster than traditional media. Among the content shared by the users, some of the topics become viral and evoke intense interaction over a certain period of time [7]. Early detection of a so-called *BUZZ* topic has a number of advantages for many different domains such as marketing, finance, or security. It helps companies, Public Safety Institutes, and persons to analyze the possible consequences and act in the early stages of a major event.

III. Data set

Buzz in social media - Buzz prediction on Twitter The UCI data set contains two different social networks: Twitter, a micro-blogging platform with exponential growth and extremely fast dynamics, and Toms Hardware (TH), a worldwide forum network focusing on new technology [10]. The Twitter community is larger than TH's, having 500 millions of visitors per month, compared to TH, that has 41 million. This project will be focusing on Twitter data analysis. The data set contains 11 primary features that are collected through time (7 days), thus leading to 77 attributes. There are two tasks of prediction that can be researched from the data set.

Firstly, there is the binary classification of a topic as being a buzz or not in the future couple of weeks based on the past 7 days history. For the training sets of this task, there are two criteria that lead to the classification of a topic as buzz or non-buzz: by absolute and relative labeling.

- **Absolute labeling** refers to classifying as buzz the topics that are ”sufficiently” popular for an established period of time in the future (data sets for different threshold values are provided). Concretely, $\sigma = 500$ implies that an example is labeled as a buzz if the sum of the predicted values of popularity ranging from $t + \beta + 1$ to $t + \beta + \delta$ is greater than 500, where the observed time-series is ranging from t to $t + \beta$ [10] .
- **Relative labeling** refers to the increment of popularity level before and after the observed time frame, namely the difference between the mean value of the active discussions of the first week of data collection (between t and $t + \beta$) and the mean value of the next couple of days that are to be predicted (between $t + \beta + 1$ and $t + \beta + \delta$).

Secondly, the regression task has a data set for predicting the number of active discussions, attribute that describes the popularity of the instances topic.

For this report we have chosen to focus on the **classification task**, analyzing the **relative labeling** data set.

IV. Previous work

This task has been previously researched, obtaining the best results using neural networks and deep learning methods. LI’s [11] report describes a deep neural network model that achieves an AUC score of 0.9816 and a neural network achieving 0.9821 AUC.

A number of implementations ([1], [3]) for different machine learning algorithms that have also been considered in this report, outline high accuracies up to 96%, but they lack comparisons with the baseline and other measurements scores that are suitable for such a highly imbalanced data set.

V. Data set analysis

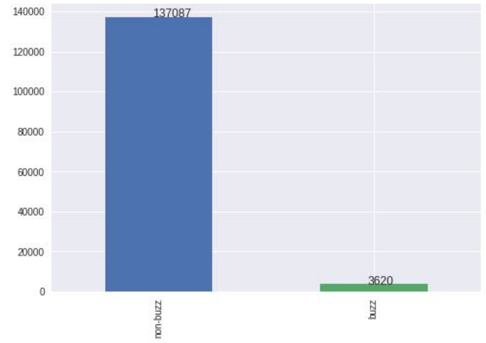
A. Class imbalance

The relative labeling data sets use the threshold values of 500, 1000 and 1500. The class imbalance problem is present in all of three them, the least prominent being for the threshold of 500 - which has a class ratio of 37.86. The distribution of the data is 2.5% positive instances (i.e. BUZZ) and 97.5% negative instances (i.e. non-BUZZ) (Fig. 1). The number of non-buzz is considerably larger for both remaining thresholds, the ratio between buzz and non-buzz for $\sigma = 1000$ being 118.54 (139530 non-buzz; 1177 buzz) and respectively 287.33 for $\sigma = 1500$ (140219 non-buzz; 488 buzz).

B. Attributes

The attributes of the data set are measured over a 7 days period. Our focus is the minority class of the data set (topics that are predicted to be very popular in the near future), thus the next analysis focuses on the instances labeled as BUZZ.

Fig. 1: Class imbalance problem of the dataset ($\sigma = 500$)



1) Number of Created Discussions (NCD): The number of created discussions associated to the instance’s keyword at the relative time 0 varies between 1 and 24210 for the instances classified as buzz, which means that the features of the instances in the data set classified as buzz were measured in different phases of the topic’s evolution and not only when the topic started to be discussed on social media. For this reason, it is worth taking into consideration the relative differences between the measurements of the chosen time period. The plot in fig. 6 summarizes this by showing the mean differences between the NCD’s of every day that was measured.

2) Author Increase (AI): This feature comprises the number of new users’ interactions on the topic represented by the instance. To depict this popularity measure, the plot in fig. 6 shows the relative differences of these mean values.

3) Number of Atomic Containers (NAC): This feature measures the number of atomic containers (comments) generated through the whole social media that are part of a discussion on the instance’s topic until the time t. An atomic container is defined as a publication of an author at a certain time t that also contains a keyword. A discussion is defined as a sequence of atomic containers chronologically ordered.

4) Number of Authors (NA): This feature measures the number of authors interacting on the instance’s topic at time t, namely the users that take part in the discussions associated with the same keyword. The evolution of the mean value is depicted in 6.

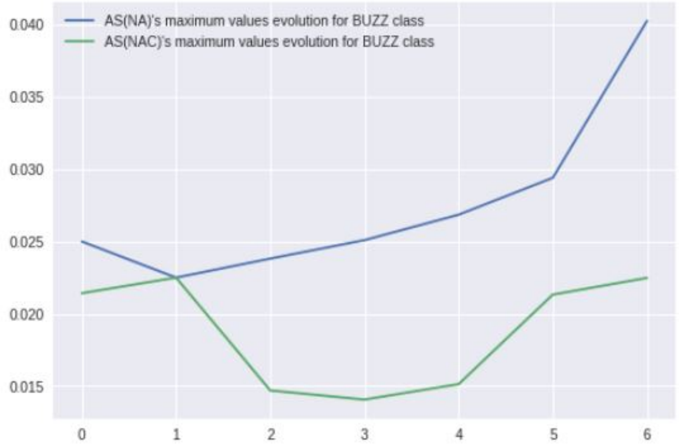
5) Attention Level: This data set’s feature is a measure of the attention paid to an instance’s topic on social media. The scores computed encapsulate the attention paid by the users to the keywords of the post. The attention measure can be defined by the Number of Atomic Containers (NAC), Number of Active Discussions (NAD), or the number of displays (ND), which is the number of the author’s mentions of the keyword. These feature values are computed by dividing the attention of a keyword at time t by the sum of attention scores of all the keywords at the same time t.

- AS(NA) - based on the number of authors (users)
  The values of this feature range from 0 to 0.04. They are computed as the ratio between the number of authors participating in discussions associated with the instance’s keyword at time t and the number of all the authors at

time t (regardless of keywords). The evolution of the maximum values registered for the buzz instances can be observed in fig. 2.

- AS(NAC) - measure with number of contributions Computed similarly, this measure is the ratio between the number of atomic containers on the instance's topic until time t and the total number of atomic containers until time t. The values of this feature range from 0 to 0.022. The evolution of the maximum values registered for the instances labeled as buzz can be observed in fig. 2.

Fig. 2: Attention level's maximum values evolution for BUZZ class

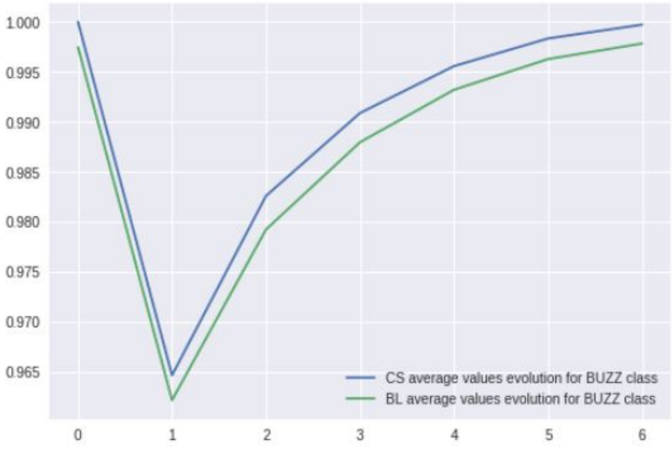


6) Burstiness Level (BL): The burstiness level for a topic at a time t is defined as the ratio between the number of newly created discussions (NCD) and the number of active discussions (NAD) on the same topic. The range of this feature for the first time point measured is 0.90-1.00. This indicates that at the time of the measurement, 90% of the discussions newly created on the same topic as the BUZZ topic of the training instance are active. The average evolution of this ratio is presented in fig. 3. As it can be observed, the tendency of having a sudden drop in the level of attention during the second day of measurement is consistent with the other measurements of topics' attention. This score is monotonically increasing since the third day of the monitored period, reaching a maximum of 0.9978 by the last day.

7) Contribution Sparseness (CS): This feature is a measure of spreading of contributions over discussion for the instance's topic at time t. Its values are within the interval $[0, 1]$. The mean values are displayed in fig. 3

8) Author Interaction (AT): This feature measures the average number of authors interacting on the instance's topic within a discussion. The average values for each day of the measurements are reflected in fig. 4a. It is worth mentioning that the minimum values are 0 for both classes starting with the second attribute (AT_1,AT_6), except the first one (AT_0) for BUZZ class, which means that for this class, there is at least one author interaction for each instance, even though it can drop to no interactions in the next measurements. Moreover, these features have various outliers and a large standard deviation. The maximum values for each attribute vary between 7.78 and 49.00 for the BUZZ class and 173 and 282 for the non-BUZZ class. This can be explained by the posts that have a large number of comments in one isolate day, but they do not get to have further attention in the following time frame or the retweet rate is low. Another possible explanation would be that the measurements were performed after the topic started to lose the public's attention (examples are shown in fig. 4b). Overall, the data suggests that the BUZZ class is characterized by interactions that spread across time rather than only a large number of interactions.

Fig. 3: CS and BL - average of BUZZ instances' evolution



9) Average Discussions Length (ADL): This feature directly measures the average length of a discussion belonging to the instance's topic. The average values for the BUZZ class are lower than for the non-BUZZ class, but the tendency to increase during the last days of the performed measurements is characteristic for the BUZZ instances, as it can be seen in fig. 5.

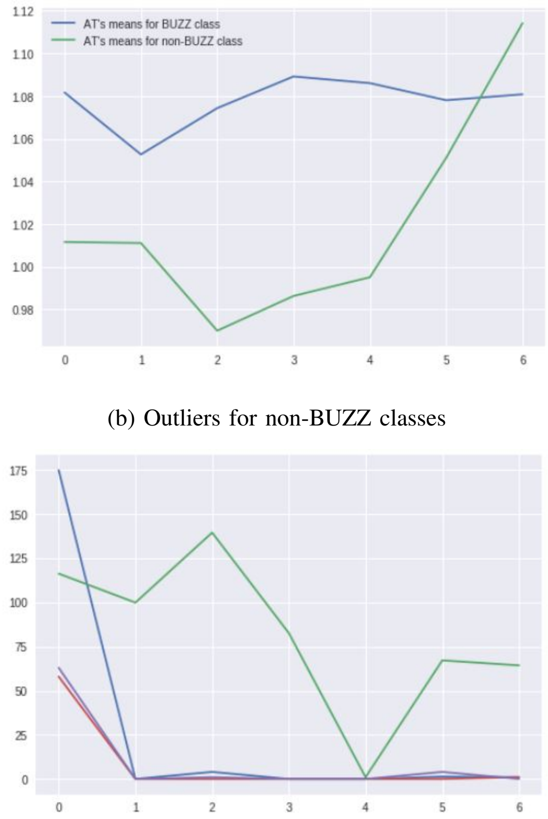
10) Number of Active Discussions (NAD): This features measures the number of discussions involving the instance's topic until time t. The relative differences for the BUZZ class are included in fig. 6.

VI. Experimental setup

The experiments presented in this report were performed using Python 3 with Jupyter notebook environment, and using the GPU cloud computational power through Colaboratory[1] and Kaggle Kernels[2]. For the rule-based approaches, we have used Orange, which is a data mining toolbox in Python [6]. All the other experiments have been performed using scikit-learn [12]. The implementations are available on GitHub at https://github.com/Dianna22/ML/tree/master/SocialMediaBUZZ.

[1] https://colab.research.google.com/

[2] https://www.kaggle.com/docs/kernels

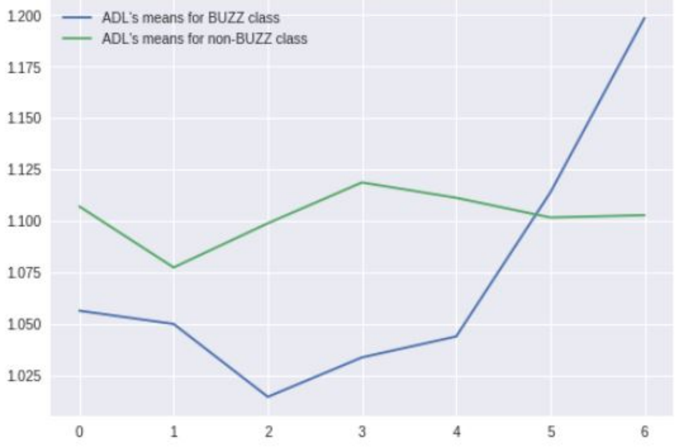


(a) Instances' mean values for each class

(b) Outliers for non-BUZZ classes

Fig. 4: Author Interaction

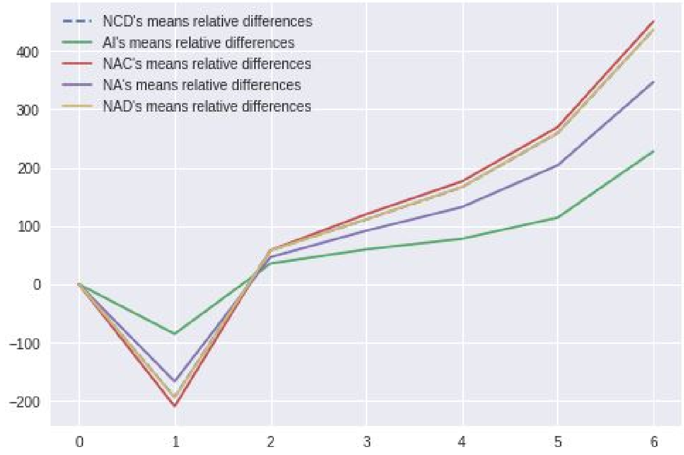
Fig. 5: ADL - average values for each class



VII. Preprocessing

During the experiments that have been performed, we have considered a number of preprocessing steps that have a different impact on the classifiers that were used. We are going to describe briefly each of them, focusing on the observed advantages and disadvantages for our problem.

Fig. 6: Means' tendencies for BUZZ class



A. Data transformation

1) Linear scaling to unit variance (standardization): This scaling process assumes the normal distribution of each feature ([2]) and it will scale them such that the new distribution is centered around 0 and has a standard deviation of 1. The results from the Linear Support Classifier described in section IX-D show that the results are impacted by this preprocessing method, obtaining slightly better performance in terms of F1 score.

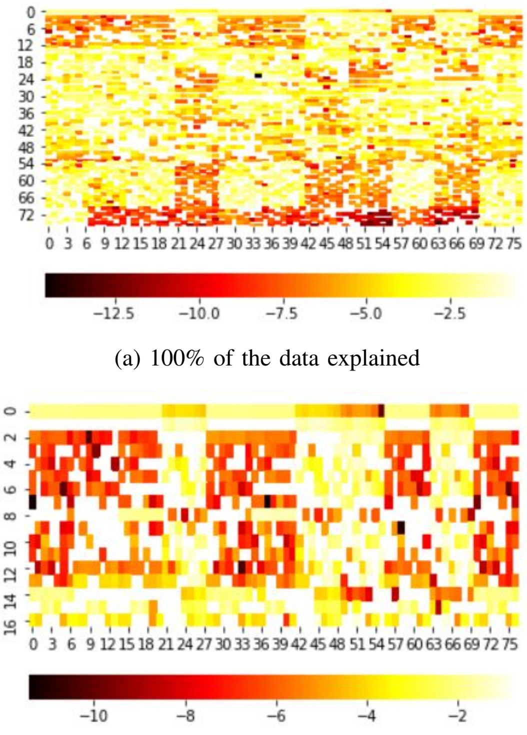
2) Min-Max Scaling: This transformation of the data set scales the features to a given range. From the experiments performed using the linear models, Min-Max Scaling obtained an F1 score ranging from 0.04 (for the scaling to the unit range) up to 0.337404 (range [0,400]). We have observed that because of the outliers that are present in the dataset, the larger the range of the features, the better the performance of the classifier. Using the range [0, 37505], (obtained from the minimum and maximum values in the data set), we obtained the best results for the linear classifiers.

From Fig. 9, one can observe that the distribution of the data set featurewise is highly skewed. This preprocessing technique achieves the best results because it preserves the skewness of the initial distribution which is an important incentive to be exploited by the machine learning models.

3) Robust Scaling: Robust Scaling uses a similar method as Min-Max Scaling, but it uses the interquartile range instead of the minimum and maximum values. Because of this, the technique is robust to outliers. This method brings the distributions of the features on the same scale and makes them overlap.

4) Normalization: We refer to the normalization as the process of row-wise scaling of the n-dimensional vectors, where n is the number of features from the data set. This method scales each value by diving them to the magnitude in n-dimensional space, normalizing them individually to unit norm.

Fig. 7: Feature's contribution to PCA's components



(a) 100% of the data explained

(b) 95% of the data explained

5) Other preprocessing techniques: It is worth mentioning that the dataset does not contain any missing values.

B. Feature selection

1) PCA: Principal Component Analysis is a dimension reduction technique where the data set is transformed from its original coordinate system to a new one [8]. This is a popular technique for identifying patterns in data and transforming it in such a way that it highlights their similarities and differences [13]. Considering the high dimensionality of our data (77 features), this method is used to reduce the dataset's complexity, extracting the most important components that explain the data. Since this method assumes that the data is zero centered, the standardization (mean normalization) of the data needs to be done beforehand in order to ensure performance improvement.

The components that are learned in an unsupervised manner are said to explain a certain percentage of the entire dataset. The contribution of each feature (on the x axis) to each principal component (on the y axis) is depicted in figure 7.

The indexes of the most important features of the principal components that explain the data can be seen in the plot in figure 18 (from the Appendix).

In our experiments, we have used 17 components, which explain 95% of the data, but this leads to poor results. The correspondence between the number of components and the value of the F1 score can be observed from fig. 8, which was analyzed for the Logistic Regression model.

We have empirically determined that the models using 35 features manage to get the most insightful information from the dataset, achieving performances close to the baseline.

Fig. 8: Average of F1 scores (over 10 folds) for the principal components used to train a Logistic Regression model

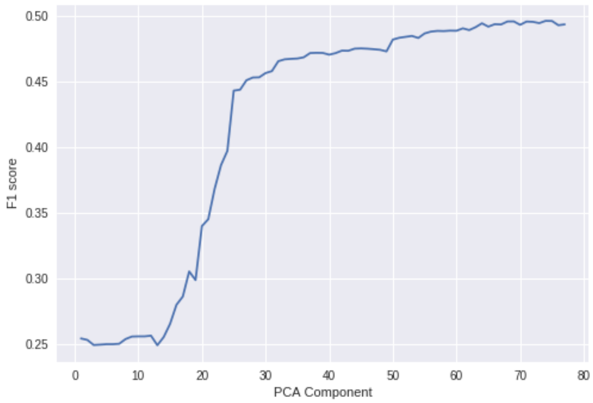


Fig. 9: Feature distribution (F35-F42)

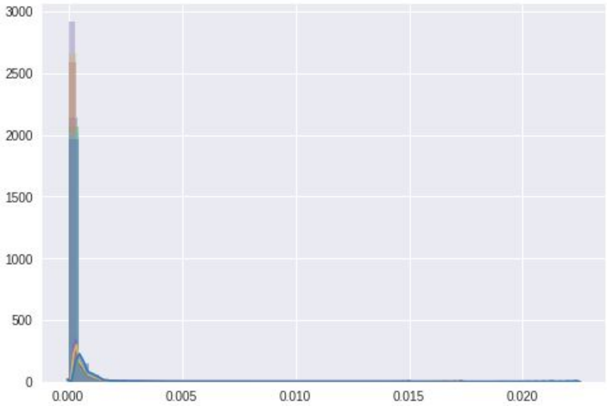
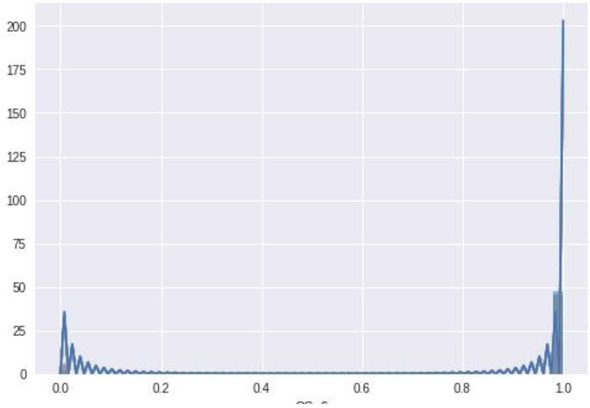


Fig. 10: Feature distribution (F42-F49)



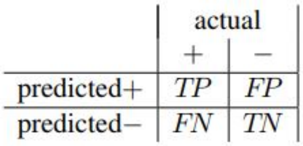
VIII. Evaluation criteria

To evaluate the predictions of the classifiers, one needs to consider the class distribution. The Twitter data set described in this report is highly imbalanced, thus, in an extreme case, a classifier that only predicts the majority class would achieve an accuracy of 97%. For this reason, accuracy would not be a good evaluation measure. Instead, one would be interested in the confusion matrix which provides all the information about the errors made by the classifiers (type I or type II). However, this is not a scalar measure. Further, we will consider a number of solutions.

A. Evaluation metrics

1) Balanced accuracy: To overcome the influence that the class imbalance problem has on the accuracy, we decided to report the balanced accuracy. This evaluation measure is defined as the average accuracy obtained on either class. Based on a confusion matrix as in fig.11: $\frac{1}{2}(\frac{TP}{P} + \frac{TN}{N})$ [4]. For the binary classification task, this metric has the same value as the AUC.

Fig. 11: Confusion matrix



	actual	
	+	−
predicted+	TP	FP
predicted−	FN	TN

2) Sensitivity or True Positive Rate (TPR): Sensitivity is also known as recall and is the probability of classifying as BUZZ a positive instance.

3) Specificity or True Negative Rate (TNR): Specificity is the probability if classifying as non-BUZZ a negative instance.

4) AUC: Some classifiers such as Naïve Bayes classifiers or neural networks yield a score that represents the degree to which an example is a member of a class. In these cases, varying the threshold value based on which the decision is made produces several classifiers. Receiver Operating Characteristic (ROC) curve is a common evaluation technique that presents the trade-off between the true positive rate and the false positive rate. This graphic plots TPR vs. FPR for a classifier at different classification thresholds. The advantage of ROC is its visualization power of depicting what region a model is more superior compared to another. To summarize the performance through a single scalar, the area under the ROC curve (AUC) is commonly used. This metric is insensitive to imbalanced classes, which is an advantage over the accuracy metric.

5) Precision and recall: One of the most common approaches of evaluation is represented by the measurements used in information retrieval: precision (that evaluates the quality of positive predictions and answers the question: **out of all the positive predictions, how many of them were indeed BUZZ topics?**) and recall (the percentage of the relevant results correctly classified answering the question: **out of all the true positive instances, how many of them were correctly identified?**). These measures are preferred because the main focus is to detect correctly positive examples (the minority class).

In our context, precision would express how accurate the model is when it predicts a topic to be a BUZZ one. A low precision score would mean that the model predicts the topics as being BUZZ when in reality they would not get much attention from the public. This can have a negative impact from a financial point of view for people who rely on this system when they make investments. High precision is thus important for **identifying only the popular topics**.

In terms of recall, a high rate would mean that the system has the ability to correctly **identify all the instances that are going to be popular**. Missing a number of BUZZ topics leads to missing the opportunity to exploit the popularity of the topic, which also has negative effects, but it may not have a direct impact.

6) F1 score: A commonly used trade-off between precision and recall is the F1 score, which is computed as the harmonic average between precision and recall. The creators of the data set mention a baseline of 0.55 F1 score. Hence, this paper reports the F1 score as one of the comparison criteria to the baseline.

B. Model selection

We have used Grid Search for hyperparameter tuning, tracking the evolution of the F1 score to choose the best model. Fig. 12 is an example of a plot that depicts the combination of the parameters considered for a linear model. To asses the power of generalization for the best model chosen in the previous step, we have chosen 10-fold validation. Since the highly skewed class distribution follows the distribution from real data, we have chosen to use a variation of this method where each fold preserves the class distribution of the data set. This version of k-fold cross validation is known as stratified.

IX. Experimental results

A. Tree-based approaches

We have performed experiments with a Decision Tree classifier and with an Extra-tree. An extra-tree is a randomized decision tree. The best split is chosen for a pair of randomly selected features. Since the Extra-trees are usually used for ensemble models, here we provide the results to consider just as a baseline.

Table VII reflects the metrics described in section VIII-A averaged over the 10 stratified folds for the tree based classifiers trained on both the data without normalization and with normalized data. The norm used to normalize the samples are $l1$, $l2$ and max (min-max normalization). For each case, we have reported the results using both the Gini impurity measure and Information Gain as criteria for measuring the quality of a split.

Fig. 12: Grid search - parameter optimization

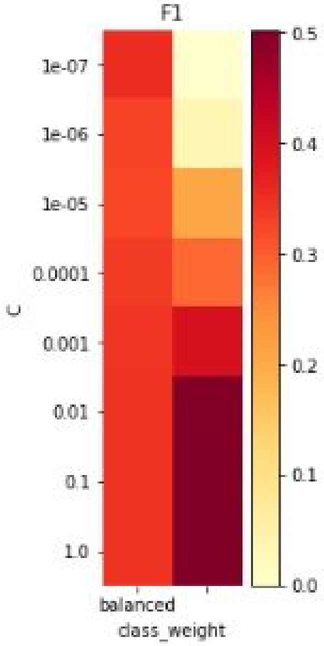
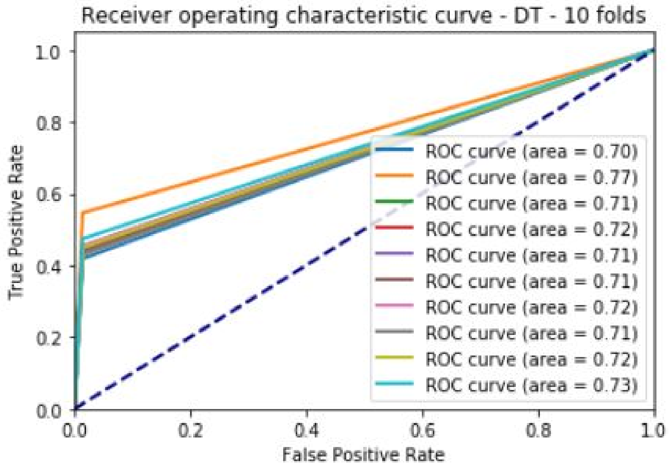


Fig. 13: ROC (weighted AUC) - Decision tree (denormalized data)



Because tree-based models are not distance-based and can handle varying ranges of features, scaling is not a necessary preprocessing step. Furthermore, since decision trees use splits as decision boundaries between features, a preprocessing step that does not influence the order of the instances is not necessary.

The results confirm that normalization has no significant impact on the performance of the classifiers since this preprocessing process preserves the order of the training instances. The differences between the values of the metrics come from the shuffling of the data.

A visualization of the ROC curves for each fold used to evaluate the decision tree classifier (with denormalized data) can be observed in fig. 13.

B. *Rule-based approaches - CN2*

The CN2 algorithm is a rule-based approach that generates a list of rules in the form of if-clauses:

```
if <complex> then predict <class>
```

where

```
<complex>
```

is a conjunction of attribute tests [5]. Table II reflects the results obtained with both the ordered version of the algorithm where the last rule is a default rule and the unordered version.

We have used a subset of 12663 of instances that preserve the class distribution from the data set. The results are very poor and the classifiers end to predict only the majority class. This can be explained by looking at the rules that have been deducted from the data and which can be seen in section B of the Appendix. The rules are very complex and focus mainly on the majority class. The testing examples fall under one of the identified rules that characterizes the majority class, thus leading to a nil recall rate.

This approach is not suitable for our data set due to the complexity of the attributes, the size of the data set, and the class imbalance problem. The test instances fall under one of the conditions of the rules of the majority class, thus not being able to identify correctly the BUZZ topics.

C. *Distance-based methods*

1) K-nearest neighbors: The K-nearest neighbors algorithm (KNN) is a supervised machine learning algorithm that assumes that similar instances are in close proximity. The decision of such a classifier is made by taking the majority vote or the weighted vote of the closest k neighbors. To determine the closest neighbors, different distance metrics (similarity measure) can be used (Euclidean, Manhattan, Chebyshev, Minkowski).

The best results obtained from the experiments performed using this algorithm are obtained when the data is previously scaled to unit variance and normalized.

We have performed a number of experiments comparing the F1 score when using linear unit variance scaling and varying the number of neighbors that should contribute to the prediction, weight function, and the power parameter of the Minkowski metric. The possible options for the weight functions are uniform - where each neighbor's class has the same load in the final decision - and distance - where the influence of the closest neighbors are higher than of those which are further away. The results are reported in table III and the ROC curve can be seen in figure 14.

The best F1 measure was obtained for $k = 3$, with distance-based weighting. Thus, for this model, we have reported the results of the experiments with different power parameters and distance metrics. Better results were obtained when we added

TABLE I: Decision tree results

	F1	AUC	B_ACC	TPR	TNR
DT_Gini_Impurity	0.428484	0.701160	0.701160	0.416136	0.986184
DT_Information_Gain	**0.456540**	0.721000	0.721000	0.456233	0.985767
Normalized_max_DT_GI	0.427897	0.702173	0.702173	0.418460	0.985885
Normalized_max_DT_IG	0.451379	0.717653	0.717653	0.449531	0.985775
Normalized_l2_DT_GI	0.433320	0.705541	0.705541	0.425183	0.985900
Normalized_l2_DT_IG	0.455376	0.720272	0.720272	0.454785	0.985759
Normalized_l1_DT_GI	0.423648	0.698122	0.698122	0.410061	0.986184
Normalized_l1_DT_IG	0.446607	0.713138	0.713138	0.440261	0.986016
ET_Gini_Impurity	0.426125	0.697188	0.697188	0.407639	0.986737
ET_Information_Gain	0.416215	0.698602	0.698602	0.412114	0.985090

TABLE II: Rule-based models results

	F1	AUC	B_ACC	TPR(recall)	Precision	TNR
CN2 Learner	0.0	0.558262	0.5	0.0	0.0	1.0

Fig. 14: ROC - KNN

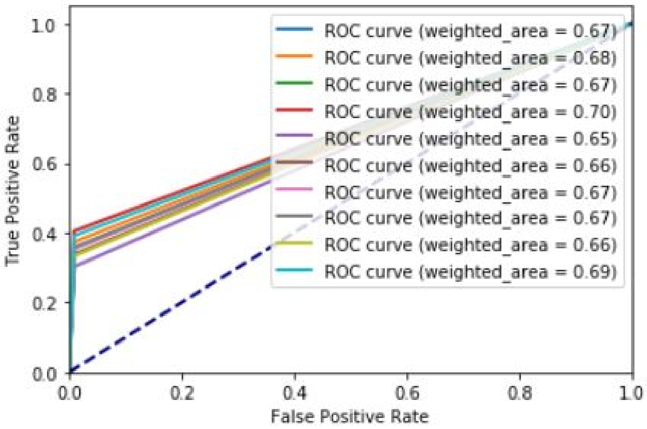


TABLE III: KNN F1-scores comparisons for different parameters

K	Weight function	Power parameter	Distance metric	F1
1	uniform	1	Manhattan	0.306122
1	uniform	2	Euclidean	0.304473
1	distance	1	Manhattan	0.306122
1	distance	2	Euclidean	0.304473
3	uniform	1	Manhattan	0.314885
3	uniform	2	Euclidean	0.320845
3	distance	1	Manhattan	0.310442
3	distance	2	Euclidean	0.323251
5	uniform	1	Manhattan	0.316115
5	uniform	2	Euclidean	0.303477
5	distance	1	Manhattan	0.317718
5	distance	2	Euclidean	0.310699
7	uniform	1	Manhattan	0.303162
7	uniform	2	Euclidean	0.293478
7	distance	1	Manhattan	0.308189
7	distance	2	Euclidean	0.300107
9	uniform	1	Manhattan	0.288268
9	uniform	2	Euclidean	0.278145
9	distance	1	Manhattan	0.299334
9	distance	2	Euclidean	0.281628
11	uniform	1	Manhattan	0.291759
11	uniform	2	Euclidean	0.268456

TABLE IV: KNN F1-scores comparisons for different distance metrics

Power parameter	Distance metric	F1
1	**Manhattan**	**0.584482**
2	Euclidean	0.558219
1	Chebyshev	0.494496
1	**Minkowski**	**0.584482**
2	Minkowski	0.558219
3	Minkowski	0.533678
4	Minkowski	0.537800

the normalization step for each instance, as it has been reported in table IV.

2) *Nearest Centroid (NC):* Nearest Centroid is an algorithm that computes the centroid of each class in the data set and assigns the new instances to the closest centroid's class, based on some chosen similarity measure. We have performed different experiments and observed that the following distance metrics obtained results in decreasing order in terms of F1 score measured performance: *seuclidean* , *mahalanobis*, *l1*,*cityblock*, *euclidean*, *l2*, *minkowski*, *sqeuclidean*, *chebyshev*, *manhattan*[3]. This ordering of decreasing performance does not hold when applying 10-fold cross validation. It can be seen from table V that *seuclidean* does not obtain the best F1 score, which means that it cannot generalize very well, but rather it works for particular subsets of the data.

For preprocessing, only Min-Max Scaling leads to acceptable results since the other 2 methods drastically shrink the range of each feature, thus influencing the decision in a negative way due to the smaller distances between the instances. From the experiments, we have observed that scaling leads to poorer results.

The evaluation of the distance-based models using the metrics described in section VIII-A using the 10 fold CV method is presented in table V.

D. Linear models

1) *Linear Support Vector Classifier - LinSVC:* For the Linear Support Vector Classifier, we have performed a number of experiments, analyzing the impact of different scaling techniques on the results together with the impact of PCA (explained in section VII-B1).

[3] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.DistanceMetric.html

TABLE V: Distance based models results

	F1	AUC	B_ACC	TPR(recall)	Precision	TNR
KNN_standard_sc_norm_L1	0.550322	0.727907	0.727907	0.461468	0.682488	0.994346
KNN_standard_sc_norm_L2	**0.553923**	0.731709	0.731709	0.469306	0.676864	0.994113
NC_seuclidean	0.365512	0.770466	0.770466	0.583441	0.266270	0.957491
NC_seuclidean_minmax_sc	0.344582	0.830574	0.830574	0.727927	0.226370	0.933222
NC_mahalanobis	0.488551	0.775588	0.775588	0.572034	0.432421	0.979141
NC_l1	0.362006	0.762778	0.762778	0.566867	0.266088	0.958689
NC_l1_minmax_sc	0.327343	0.823901	0.823901	0.719212	0.212413	0.928590
NC_cityblock	0.362006	0.762778	0.762778	0.566867	0.266088	0.958689
NC_euclidean	0.371304	0.766410	0.766410	0.572731	0.274899	0.960089
NC_l2	0.371304	0.766410	0.766410	0.572731	0.274899	0.960089
NC_minkowski	0.371304	0.766410	0.766410	0.572731	0.274899	0.960089
NC_sqeuclidean	0.371304	0.766410	0.766410	0.572731	0.274899	0.960089
NC_chebyshev	0.388955	0.756293	0.756293	0.545847	0.302385	0.966738
NC_manhattan	0.311913	0.848548	0.848548	0.782626	0.194871	0.914470

Firstly, we have observed that the larger the range of the scaled values for the features, the larger F1 score is. If the results obtained by scaling the data to the interval $[0, 1]$ were very poor because of the large number of different values of each feature and because of the outliers present in the data, the best results were obtained for the range between 0 and the maximum value in the data set (out of all the features), as it can be observed from the results table (VI) that depicts the average value obtained from the results of the 10 folds. Compared with the other techniques, the best F1 measure is obtained from high precision and lower recall. Thus, the range of the scaling leads to very different results both in terms of recall (ranging from 0.94 down to 0.35) and in terms of precision (0.73-0.20).

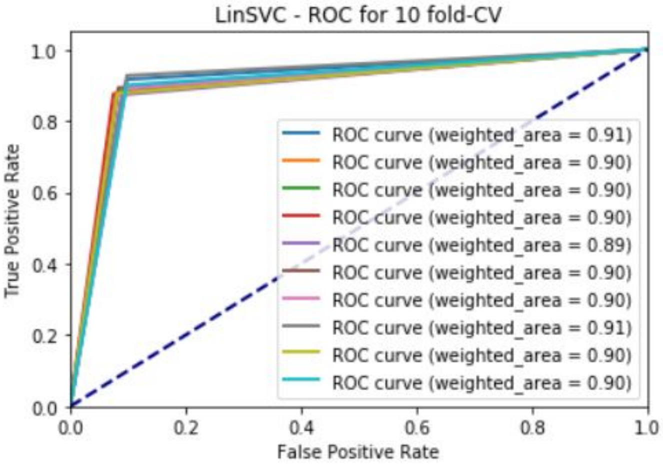
Figure 15 plots the ROC curves for the 10 folds together with the area under the curve for the second best scaling technique (Min-Max scaling with the range $[-300, 300]$).

We have also reported the results obtained using the principal components that explain 95% of the data. Because this method assumes the data is zero-centered, this feature selection method improves the results only when we previously transform the data by standardization. Combining PCA with other scaling techniques leads to poor results.

2) SVM with SGD training: We have used an implementation of SVM that uses Stochastic Gradient Descent (SGD) learning. The SGD optimization technique estimates the gradient of the loss for each sample at a time, thus being more suitable especially for large data sets since it does not wait to go through the entire data set before updating the weights. For best results, this method needs to be used on unit variance data, having mean 0. This is also reflected in the results from table VI. This method, along with the standardization and normalization of the data obtains the best results among the linear classifiers.

3) Logistic regression: For the logistic regression classifier, we have reported the experiments that show the impact on the results of the 3 scaling techniques considered in this report. Because this method is easily impacted by the outliers in the training data, the standardization of the data has improved the model the most, as it can be seen in the results table (LogReg_unitvar_sc).

Fig. 15: ROC - Linear SVC



E. Ensembles

1) Random forest: Random forest method consists of an ensemble of decision trees trained on samples of the data sets. The decision of the model is made using averaging as a technique of both improving the performance and of avoiding overfitting. The ROC curve for the 10 folds can be seen in figure 16.

2) Bagging: Bagging is an abbreviation of Bootstrap Aggregating. The Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original data set and then aggregates their individual predictions (either by voting or by averaging) to form a final prediction. In our case, we have chosen to use as base classifiers the ones that obtained the best results in our previous experiments.

The best results obtained with this method were obtained when using the random forest classifier. This is due to the high number of features and to the complexity of our training. The

TABLE VI: Linear models results

	F1	AUC	B_ACC	TPR(recall)	Precision	TNR
LinSVC_minmax_sc(0,37505)	0.473080	0.676131	0.676131	0.356028	0.731774	0.996234
LinSVC_minmax_sc(-300,300)	**0.338376**	**0.901529**	0.901529	0.892426	0.208842	0.910633
LinSVC_minmax_sc(0,400)	0.337491	0.898187	0.898187	0.885142	0.208568	0.911232
LinSVC_minmax_sc(-200,200)	0.335334	0.899154	0.899154	0.885142	0.206739	0.909903
LinSVC_minmax_sc(-100,100)	0.329144	0.894413	0.894413	0.880551	0.202474	0.908275
LinSVC_minmax_sc(0,37505)+PCA	0.060033	0.582786	0.582786	0.949431	0.031004	0.216141
LinSVC_unitvar_sc	0.337404	0.898048	0.898048	0.884865	0.208517	0.911232
LinSVC_unitvar_sc+PCA	0.366148	0.686265	0.686265	0.391111	0.392707	0.981418
LinSVC_robust_sc	0.317415	0.693161	0.693161	0.429722	0.307566	0.956601
LinSVC_robust_sc+PCA	0.158933	0.551775	0.551775	0.108666	0.560911	0.994883
LogReg_robust_sc	0.459168	0.669467	0.669467	0.342897	0.704928	0.996038
LogReg_unitvar_sc	**0.493299**	0.684882	0.684882	0.373457	0.735563	0.996307
LogReg_minmax_sc	0.471874	0.677706	0.677706	0.359748	0.696080	0.995664
LogReg	0.455466	0.690312	0.690312	0.388994	0.554297	0.991631
SVM_SGD	0.490591	0.737924	0.737924	0.488891	0.508609	0.986957
SVM_SGD_L1_norm_unitvar_sc	0.496249	0.755965	0.755965	0.527551	0.483569	0.984379
SVM_SGD_unitvar_sc_L1_norm	0.405680	0.650148	0.650148	0.305207	0.652126	0.995089
SVM_SGD_L2_norm_unitvar_sc	0.516521	0.760515	0.760515	0.535105	0.514327	0.985925
SVM_SGD_unitvar_sc_L2_norm	0.314965	0.598974	0.598974	0.199480	0.781593	0.998468

random forest proves to be able to gather insights from the structure of the data set.

3) Ensemble of independent weak classifiers: We have considered an ensemble method that would directly address the imbalance problem. After creating a testing set of 3518 instances, preserving the label distribution, this method creates mini-datasets to train classifiers that will be averaged for the final prediction of the ensemble. The majority class is split into subsets of approximately the same size as the whole positive data set (which is of 3529 instances) and each classifier is trained on this new balanced mini-dataset. Averaging the predictions of these fitted models and thresholding at the mid point of the probability-like scores will give us the final prediction.

As base classifiers, we have used some of the classifiers that performed well on the whole dataset, but they seem to not perform as well on a smaller dataset. For this reason, tuning the hyperparameters and using models that are suited for small datasets should be considered when using this approach.

X. Performance comparison

The performances of the algorithms from the 10-fold method can be visualized in figure 17, where the mean values are represented by the horizontal line.

To assess whether the mean's differences of the scores that we have considered in the previous section for evaluating our models are significant or they are only due by chance, we have performed a statistical T-test on the results obtained on the 10 folds. The table X shows the p-values obtained when making the pairwise comparisons between algorithms. A p-value smaller than 0.05 proves that there is a statistical difference between the distributions of the scores. We can thus confirm with 95% confidence that the algorithms obtained significantly different results. The decision between the two of them can be made taking into consideration the average F1 score.

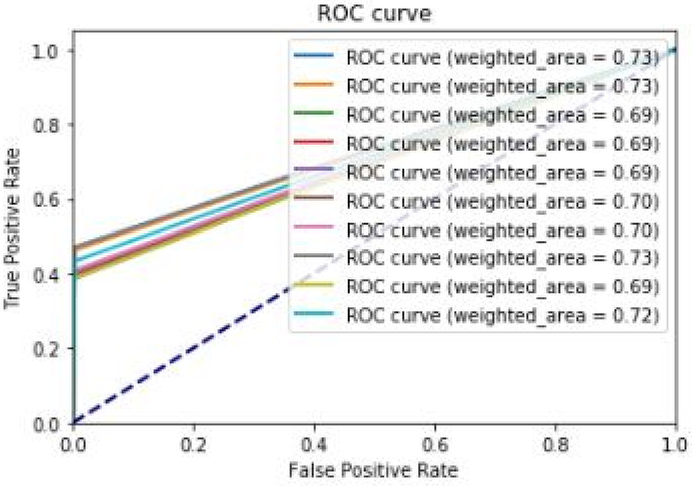


Fig. 16: ROC for bagging with random forest base model

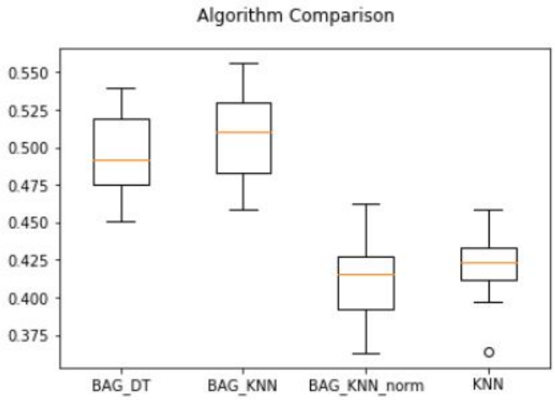


Fig. 17: Algorithms' comparison

TABLE VII: Ensemble models results

	F1	AUC	B_ACC	TPR	TNR	
Random_forest_st	**0.570278**	0.723979	0.723979	0.451403	0.776747	0.996556
BAG+random_forest	**0.548917**	0.708710	0.708710	0.420316	0.794784	0.997103
BAG+decision_tree	0.495020	0.695199	0.695199	0.395747	0.663054	0.994652
BAG+KNN	0.508262	0.696389	0.696389	0.397098	0.708743	0.995681
BAG+KNN_standard_sc_norm_L2	0.412021	0.651740	0.651740	0.308435	0.623174	0.995046
weak_LogReg	0.408679	0.887766	0.887766	0.830882	0.270983	0.944650
weak_LinSVC	0.381294	0.686750	0.686750	0.389705	0.373239	0.983795
weak_KNN	0.351060	0.883402	0.883402	0.839705	0.221919	0.927099
weak_DT	0.308155	0.9024756	0.9024756	0.9029411	0.1857791	0.9020100

TABLE VIII: T-Test - AUC

Algotrithm pair	p-value	Statistical difference (95% confidence)
DT - LinSVC	6.67e-10	yes
DT - PCA_LinSVC	0.0932	no
DT - Bagging_randfor	0.0962	no
LinSVC - PCA_LinSVC	1.21e-06	yes
LinSVC - Bagging_randfor	1.60e-11	yes
PCA_LinSVC - Bagging_randfor	0.3235	no

TABLE IX: T-Test - F1 scores on 10-fold

	BAG_DT	BAG_KNN	BAG+KNN_stsc_L2	KNN_stsc_L2
	0.520674	0.542945	0.391026	0.428360
	0.526678	0.531034	0.418079	0.426073
	0.479860	0.526502	0.429091	0.409600
	0.451025	0.458716	0.424107	0.446565
	0.461017	0.475352	0.362963	0.363636
	0.483363	0.488971	0.413043	0.397415
	0.474227	0.480874	0.379447	0.421405
	0.512821	0.555891	0.444444	0.434911
	0.500956	0.495127	0.395918	0.416974
	0.539580	0.527211	0.462094	0.458136
mean	0.495020	0.508262	0.412021	0.420308
st_dev	0.028085	0.030845	0.028638	0.02508

XI. Discussion

While the linear models seem to not be able to be better than the baseline, we can conclude that using the 77-dimension vector space we fall under the curse of dimensionality phenomenon, which states that the predictive power of a classifier first increases as the number of dimensions/features used is increased but then decreases [14]. This phenomenon is also known as *Hughes phenomenon*[9].

The rule-based models perform very poorly and require a lot of time to come up with a rule list that can explain the large dataset. This approach does not manage to detect the minority class due to the complexity of the data and due to the high range of the values of some attributes, as it can be seen in figures 9 and 10.

TABLE X: T-Test - F1

Algotrithm pair	p-value	Statistical difference (95% conf)
BAG_DT - BAG_KNN	0.05712	yes
BAG_DT - BAG+KNN_stsc_norm_L2	1.17e-05	yes
BAG_KNN - BAG+KNN_stsc_norm_L2	5.12e-06	yes
KNN_stsc_L2 - BAG_DT	1.55e-05	yes
KNN_stsc_L2 - BAG+KNN_stsc_norm_L2	1.93e-05	yes
KNN_stsc_L2 - BAG+KNN_stsc_norm_L2	0.2565	no

The distance-based approach manages to achieve the baseline accuracy mentioned by the creators of the dataset, while a tree-based ensemble method surpasses this result (random forest 0.57). The comparison between pairs of algorithms shows that there is a statistical difference between the Bagging algorithm using KNN as a base model. The ensemble models try to overcome the class imbalance issue, but the averaging of the weak classifiers does not succeed in achieving better results. The reason behind it is that the models that we have tried were tuned for the whole dataset, while for very small datasets they seem to not generalize well.

XII. Key learnings

1) **Evaluation metrics importance:** From our results, it is worth mentioning that a high AUC can correspond to a high recall rate and a low precision rate. For example, the LinSVC_minmax_sc(-300,300) model has a 0.9015 AUC, but a low F1 score - 0.3383. Since high precision relates to a low false positive rate, and high recall relates to a low false negative rate, even though a model has a high AUC, it can have a high false positive rate, which means that the model fails to identify the true BUZZ topics. Choosing the appropriate evaluation metric for a highly imbalanced data set it is paramount not only for comparison purposes with other approaches but also for establishing the power of the system to detect the minority class, which is the one that is of the most interest.
2) **Testing the generalization power:** To assess the performance of a machine learning algorithm, it is important to report the results obtained in multiple runs. When using the 10-fold cross-validation method and analyzing the standard deviation of the results, one can see that even though at one run the model achieves good results, on average its performance may drop.
3) **Comparing multiple algorithms:** When the means of the performances over the 10 folds are very close to each other, to compare their corresponding algorithms, performing a statistical test can convey whether there is indeed a significant difference between the results.
4) **The class-imbalance problem needs to be addressed using techniques suitable for the data set on dis-**

cussion. Since the minority class contains very few instances, training models for these mini sets containing a 50:50 ratio of positives and negatives examples does not lead to effective models.

5) **Important features:** Although the dataset contains 11 features extracted from 7 consecutive days, thus leading to a total of 77 features, only 35 of them proved to be insightful, which means that certain peaks in the distribution of the data during the week have more influence on the label.
6) **Dependent features** Features obtained by dividing two other features (for example Attention Level - AS(NAC)), prove to have very little influence on the most important components extracted using PCA, compared to the features that they are obtained from (NAC), which are in the top most important features.

XIII. Conclusions

In the context of extensive social media communication, it is important to monitor and predict the topics that will gain a lot of attention in the near future, based on past data in order to act accordingly and prevent unfortunate situations that may appear as consequences of the trends discussed online.

This report comprises a number of research directions that can be further investigated in more details. It is an extensive comparison of linear, tree-based, distance-based, rule-based, and ensemble models that have been exemplified. Furthermore, we have done an analysis of preprocessing techniques (such as normalization and scaling) and data reduction, together with methods for overcoming the class imbalance problem. The report succeeded to present a thorough analysis of the dataset containing Twitter data, together with a number of experiments that confirm the claims regarding the used techniques that were made during the report. The evaluation of the models has also been thoroughly done, outlining different measurements criteria suitable for a class-imbalanced dataset, that can help one draw conclusions regarding the performance of the systems under test. We are also emphasizing the importance of comparing with the baseline on the same measurement, in our case - the F1 score. We have reported it (along with other insightful metrics) for comparison to the baseline and considered the previous work done on this specific task. The best results obtained using the distance-based model reach the baseline of 0.55 F1 score, while the random forest model obtains the best F1 score of 0.57. To compare different algorithms with similar average performance on a 10-fold cross validation evaluation, the t-test helped us determine if there is a true difference between the results.

XIV. Future work

As mentioned in section IV, neural networks and deep learning proved to have good results in social media analysis. Different architectures have already obtained a good AUC, so it is worth computing the other metrics in order to compare their results to the baseline.

The class imbalance problem needs to be addressed using new techniques such as focusing on determining patterns in the minority class rather than trying to describe it by comparing with the majority class' instances [15]. Such an analysis would allow one to detect the most important characteristics of the BUZZ topics, thus allowing to perform oversampling (random or cluster-based) and undersampling (by eliminating the outliers of the majority class and the instances that are very similar) in an effective way. Another possible solution for the problem encountered in the reported method of approaching this problem would be to determine using grid search or heuristics, models that perform well on each mini-data set and then creating a neural network with their votes to detect the weights the classifiers should have in the final decision. Boosting algorithms can also have a good impact on the results since at each step they concentrate on the instances wrongly classified in a previous step.

Future datasets can be enhanced with feature selected from the text used by the users since they may contain BUZZ words. Such patterns can be exploited using word embeddings and natural language techniques.

There is also room for improvement in the preprocessing steps. Other feature selection such as univariate selection should be considered for further analysis, along with a more in-depth investigation of the outliers of the dataset.

Another direction of research is analyzing the remaining datasets on the same task of classification and on the regression task, both on the Twitter data and Toms Hardware.

References

[1] Tech. rep. 2018. URL: https://github.com/nilabja9/TwitterBuzzML.

[2] Selim Aksoy and Robert M. Haralick. "Feature Normalization and Likelihood-based Similarity Measures for Image Retrieval". In: *Pattern Recogn. Lett.* 22.5 (Apr. 2001), pp. 563–582. ISSN: 0167-8655. DOI: 10.1016/S0167-8655(00)00112-4. URL: http://dx.doi.org/10.1016/S0167-8655(00)00112-4.

[3] Rao Amruutha. Tech. rep. 2018. URL: https://github.com/amruutha/Twitter-buzz-prediction/blob/master/Classification_Models.ipynb.

[4] Kay Henning Brodersen et al. "The Balanced Accuracy and Its Posterior Distribution". In: *Proceedings of the 2010 20th International Conference on Pattern Recognition*. ICPR '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 3121–3124. ISBN: 978-0-7695-4109-9. DOI: 10.1109/ICPR.2010.764. URL: https://doi.org/10.1109/ICPR.2010.764.

[5] Peter Clark and Tim Niblett. "The CN2 Induction Algorithm". In: *Mach. Learn.* 3.4 (Mar. 1989), pp. 261–283. ISSN: 0885-6125. DOI: 10.1023/A:1022641700528. URL: http://dx.doi.org/10.1023/A:1022641700528.

[6] Janez Demšar et al. "Orange: Data Mining Toolbox in Python". In: *Journal of Machine Learning Research* 14 (2013), pp. 2349–2353. URL: http://jmlr.org/papers/v14/demsar13a.html.

[7] Clemens Deusser et al. “Buzz in Social Media: Detection of Short-lived Viral Phenomena”. In: *Companion Proceedings of the The Web Conference 2018*. WWW ’18. Lyon, France: International World Wide Web Conferences Steering Committee, 2018, pp. 1443–1449. ISBN: 978-1-4503-5640-4. DOI: 10.1145/3184558.3191591. URL: https://doi.org/10.1145/3184558.3191591.

[8] Peter Harrington. *Machine Learning in Action*. Greenwich, CT, USA: Manning Publications Co., 2012, pp. 270–278. ISBN: 1617290181, 9781617290183.

[9] G. Hughes. “On the mean accuracy of statistical pattern recognizers”. In: *IEEE Transactions on Information Theory* 14.1 (Jan. 1968), pp. 55–63. ISSN: 0018-9448. DOI: 10.1109/TIT.1968.1054102.

[10] F. Kawala et al. “Prédictions d’activité dans les réseaux sociaux en ligne”. In: *In Actes de la Conférence sur les Modèles et l’Analyse des Réseaux : Approches Mathématiques et Informatique (MARAMI)*. 2013, p. 16. URL: https://hal.archives-ouvertes.fr/hal-00881395/document.

[11] LI Ning. *Deep Neural Network in classification of Twitter Buzz*. Tech. rep. University of Washington, USA, 2015. URL: https://github.com/ninginthecloud/Buzz-prediction-on-Twitter/blob/master/report/Ning_final_report.pdf.

[12] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[13] Lindsay I Smith. *A tutorial on principal components analysis*. Tech. rep. Cornell University, USA, Feb. 2002. URL: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf.

[14] G. V. Trunk. “A Problem of Dimensionality: A Simple Example”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.3 (July 1979), pp. 306–307. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1979.4766926.

[15] Gary M. Weiss. “Mining with Rarity: A Unifying Framework”. In: *SIGKDD Explor. Newsl.* 6.1 (June 2004), pp. 7–19. ISSN: 1931-0145. DOI: 10.1145/1007730.1007734. URL: http://doi.acm.org/10.1145/1007730.1007734.

APPENDIX

A. PCA

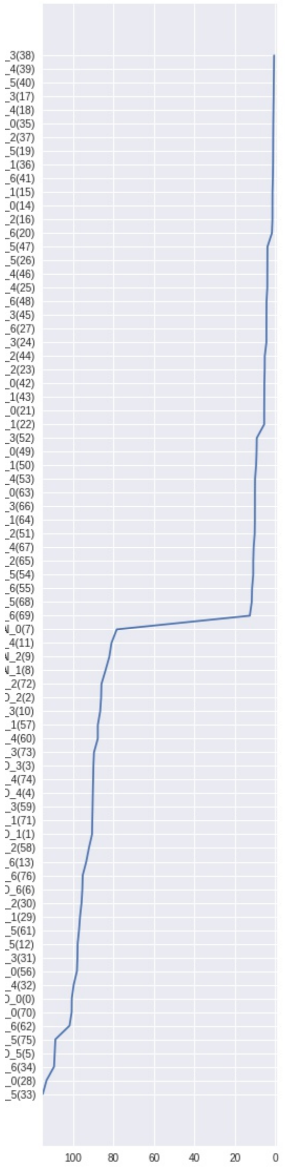


Fig. 18: Feature’s importance

B. Rules learned by the CN2 Unordered Learner

IF NCD$_5$ $<= 76.0ANDBL_6 >=$ $0.9977879999999999THENBUZZ = 0IFNCD_6 <=$ $425.0ANDAI_0 <= 58.0ANDADL_5 >=$ $1.0037040000000002THENBUZZ = 0IFNAC_6 <=$ $425.0ANDAS(NA)_1 <= 0.00021ANDBL_5 >=$ $0.994764ANDAS(NAC)_5 >= 4.1e-05THENBUZZ =$ $0IFBL_6 <= 0.980282ANDBL_5 >=$ $0.9931270000000001THENBUZZ = 0IFNAC_6 <=$ $815.0ANDBL_5 <= 0.9929819999999999ANDBL_1 >=$ $0.9285709999999999THENBUZZ = 0IFNA_6 <=$ $377.0ANDNA_1 >= 238.0ANDAI_0 >=$ $28.0THENBUZZ = 0IFAS(NAC)_6 <=$ $0.00057ANDNA_2 <= 92.0ANDNCD_6 >=$ $101.0THENBUZZ = 0IFNAC_6 <=$ $948.0ANDAI_1 >= 333.0THENBUZZ =$ $0IFNAC_6 <= 815.0ANDAI_0 >= 188.0ANDNAC_3 >=$ $334.0ANDNCD_4 >= 157.0THENBUZZ =$ $0IFNAC_6 <= 425.0ANDAS(NA)_3 <=$ $0.00037400000000000004ANDAS(NA)_1 >=$ $0.000242THENBUZZ = 0IFAS(NAC)_6 <=$ $0.00057ANDAS(NAC)_5 <=$ $0.00015900000000000002ANDAS(NAC)_6 >=$ $0.00011599999999999999THENBUZZ = 0IFAI_6 <=$ $312.0ANDBL_3 <= 0.996132ANDBL_0 >=$ $0.9959180000000001THENBUZZ =$ $0IFBL_2 <= 0.9939530000000001ANDAT_0 >=$ $1.228956ANDAI_0 >= 68.0THENBUZZ =$ $0IFNAC_6 <= 1208.0ANDNAC_3 <=$ $71.0ANDBL_4 >= 0.9964540000000001THENBUZZ =$ $0IFAI_6 <= 312.0ANDNAC_4 <=$ $424.0ANDAS(NA)_1 >= 0.000336ANDAT_3 <=$ $1.124352ANDBL_6 >= 0.994783THENBUZZ =$ $0IFBL_3 <= 0.9943639999999999ANDAT_5 <=$ $1.0803989999999999ANDAS(NA)_3 >=$ $0.000543THENBUZZ = 0IFBL_3 <=$ $0.9940620000000001ANDAT_6 >=$ $1.166835ANDBL_6 >= 0.9866469999999999THENBUZZ =$ $0IFAS(NAC)_6 <= 0.000629ANDNAC_5 >=$ $1243.0ANDBL_1 >= 0.99902THENBUZZ =$ $0IFBL_0 <= 0.9963719999999999ANDADL_5 <=$ $1.038902ANDAI_3 >= 46.0THENBUZZ =$ $0IFAS(NAC)_6 <= 0.00057ANDBL_3 <=$ $0.998966ANDAI_3 >= 266.0THENBUZZ =$ $0IFBL_2 <= 0.993243ANDAS(NA)_2 >=$ $0.00034500000000000004ANDAS(NA)_2 <=$ $0.001457ANDADL_2 >= 1.083045THENBUZZ =$ $0IFNCD_6 <= 1147.0ANDAI_5 <=$ $116.0ANDNA_5 >= 180.0THENBUZZ =$ $0IFNCD_6 <= 1147.0ANDAS(NAC)_6 >=$ $0.001004ANDAI_6 >= 270.0THENBUZZ =$ $0IFNAC_6 <= 1202.0ANDNCD_0 >=$ $434.0ANDNA_5 <= 598.0ANDAS(NAC)_6 >=$ $0.00040599999999999995THENBUZZ =$ $0IFAS(NAC)_6 <= 0.000365ANDAS(NAC)_4 <=$ $0.000181ANDAI_0 >= 83.0ANDNCD_2 >=$ $99.0THENBUZZ = 0IFBL_6 <=$ $0.997015ANDNA_2 >= 430.0ANDNA_2 <=$ $734.0ANDNCD_1 >= 228.0THENBUZZ =$ $0IFADL_6 >= 1.3472709999999999ANDAS(NA)_3 <=$ $0.0055130000000000005ANDBL_0 >=$ $0.9923860000000001ANDAT_3 >=$ $1.0281049999999998THENBUZZ =$ $0IFAS(NAC)_6 <= 0.0012779999999999998ANDNA_1 >=$ $453.0ANDBL_6 <= 0.996283ANDNAC_6 >=$ $1111.0THENBUZZ = 0IFAS(NAC)_6 <=$ $0.000985ANDNA_0 >= 968.0ANDAS(NA)_3 >=$ $0.0013109999999999999ANDAS(NAC)_1 >=$ $0.00029THENBUZZ = 0IFAS(NAC)_6 <=$ $0.000493ANDNCD_4 >= 451.0ANDAS(NAC)_4 <=$ $0.000342ANDNCD_1 >= 275.0THENBUZZ =$ $0IFNAC_6 <= 3055.0ANDNCD_6 <=$ $643.0ANDAI_3 <= 86.0ANDAS(NA)_3 >=$ $0.000155THENBUZZ = 0IFNAC_3 <=$ $1899.0ANDNAC_1 >= 609.0ANDAI_2 >=$ $426.0ANDNCD_5 <= 1499.0ANDNCD_3 >=$ $1318.0THENBUZZ = 0IFNAC_6 <=$ $1373.0ANDNAC_4 >= 854.0ANDAS(NAC)_5 <=$ $0.000603ANDAI_1 >= 149.0THENBUZZ =$ $0IFAS(NAC)_1 <= 0.001417ANDNAC_1 >=$ $609.0ANDNCD_1 <= 683.0ANDNA_3 >=$ $535.0ANDAS(NAC)_6 >=$ $0.00041500000000000006ANDAI_1 >=$ $88.0THENBUZZ = 0IFAS(NAC)_1 <=$ $0.001417ANDNAC_6 <= 398.0ANDADL_3 <=$ $1.044118ANDNCD_3 >=$ $75.0ANDNA_1 <= 213.0ANDAS(NA)_5 <=$ $0.0012029999999999999THENBUZZ = 0IFNAC_0 <=$ $2331.0ANDNAC_1 >= 717.0ANDNA_5 <=$ $1132.0ANDNA_5 >= 965.0ANDNAC_3 >=$ $924.0THENBUZZ = 0IFNCD_5 <=$ $3385.0ANDNAC_6 <= 1093.0ANDAS(NAC)_6 >=$ $0.000728ANDNCD_3 >= 207.0ANDAI_3 >=$ $87.0ANDAS(NA)_2 >= 0.000113THENBUZZ =$ $0IFNA_6 <= 3791.0ANDAI_5 >= 1266.0ANDNCD_2 >=$ $825.0ANDAI_2 <= 1582.0ANDAI_1 >=$ $927.0ANDNCD_0 >= 850.0ANDBL_2 >=$ $0.995755THENBUZZ = 0IFAS(NA)_3 <=$ $0.001408ANDNCD_1 >= 226.0ANDNCD_6 <=$ $643.0ANDNCD_6 >= 458.0ANDAI_1 <=$ $297.0ANDAI_2 <= 286.0ANDBL_3 >=$ $0.996THENBUZZ = 0IFNA_0 <=$ $1542.0ANDNAC_0 >= 734.0ANDAI_4 >=$ $738.0ANDNCD_4 <= 1647.0ANDBL_0 >=$ $0.9963290000000001ANDNCD_5 >=$ $1059.0ANDBL_3 >= 0.9946280000000001THENBUZZ =$ $0IFAS(NA)_4 <= 0.001106ANDNA_4 >=$ $291.0ANDAI_1 >= 353.0ANDAI_0 >=$ $347.0ANDNCD_2 >= 511.0ANDAS(NA)_6 >=$ $0.000884THENBUZZ =$ $0IFBL_3 <= 0.997091ANDADL_4 <=$

$1.1041370000000001ANDAI_3 <=$
$912.0ANDAS(NA)_1 >= 0.000193ANDNCD_2 >=$
$190.0ANDAI_0 <= 1308.0ANDADL_6 >=$
$1.018106THENBUZZ = 0IFAS(NA)_5 <=$
$0.000898ANDNCD_0 >= 426.0ANDNA_2 <=$
$596.0ANDNAC_4 >= 439.0ANDAI_0 >=$
$80.0ANDBL_5 >= 0.998188ANDBL_2 >=$
$0.996024THENBUZZ = 0IFAI_4 <=$
$377.0ANDNCD_0 >= 944.0ANDAS(NAC)_4 >=$
$0.000426ANDNCD_1 >= 399.0ANDAS(NA)_0 >=$
$0.000596ANDAI_0 <= 551.0THENBUZZ =$
$0IFNA_5 <= 566.0ANDAI_4 <=$
$188.0ANDAI_4 >= 94.0ANDAI_0 >=$
$64.0ANDAS(NA)_3 <= 0.000965ANDBL_0 >=$
$0.9908680000000001ANDNCD_3 >=$
$116.0ANDAS(NAC)_0 <= 0.000659THENBUZZ =$
$0IFNCD_5 >= 1843.0ANDAI_5 <=$
$1207.0ANDAI_3 >= 378.0ANDBL_4 >=$
$0.9956360000000001ANDAS(NAC)_0 <=$
$0.002933ANDAI_1 >= 316.0ANDAI_2 >=$
$364.0ANDNAC_0 <= 3794.0ANDAS(NAC)_5 >=$
$0.00088ANDAS(NAC)_4 >= 0.000683ANDNA_6 >=$
$1059.0THENBUZZ = 0IFAT_1 >=$
$1.011364ANDADL_1 <= 1.173258ANDBL_3 >=$
$0.9959899999999999ANDAS(NA)_2 >=$
$0.000259ANDAS(NAC)_6 >=$
$0.0006490000000000001ANDAS(NA)_6 <=$
$0.007495ANDNCD_4 >=$
$101.0ANDBL_5 >= 0.995726ANDAT_3 >=$
$1.006557ANDAS(NA)_6 >= 0.000926ANDBL_6 >=$
$0.9948049999999999ANDAS(NAC)_0 <=$
$0.004070000000000001ANDAT_6 <=$
$1.133333ANDAI_0 >= 59.0THENBUZZ =$
$0IFNA_6 >= 3791.0ANDBL_1 >=$
$0.9992610000000001ANDAS(NAC)_1 >=$
$0.002301THENBUZZ =$
$1IFAS(NA)_6 >= 0.007428ANDBL_0 >=$
$0.995188ANDAI_1 >= 458.0THENBUZZ =$
$1IFNAD_6 >= 2205.0ANDAI_2 <=$
$690.0ANDAS(NA)_6 <= 0.004763ANDBL_1 >=$
$0.9926200000000001ANDAI_6 >= 446.0ANDBL_0 >=$
$0.993877THENBUZZ = 1IFNCD_2 >=$
$2118.0ANDAS(NAC)_5 <= 0.003159ANDAT_2 <=$
$1.3207010000000001ANDAS(NA)_2 >=$
$0.002094ANDAS(NA)_0 >= 0.00165ANDNCD_1 <=$
$5425.0THENBUZZ = 1IFNAD_6 >=$
$1164.0ANDNCD_1 <= 425.0ANDNA_5 >=$
$597.0ANDNAC_6 >= 1386.0ANDAS(NA)_5 <=$
$0.002506ANDNCD_5 >= 873.0THENBUZZ =$
$1IFNCD_6 >= 967.0ANDNAC_0 <=$
$465.0ANDAS(NA)_6 <= 0.001549ANDAS(NA)_4 >=$
$0.00023799999999999998ANDBL_2 >=$
$1.0ANDNCD_6 >= 994.0ANDAI_2 >=$
$48.0THENBUZZ = 1IFNAD_6 >=$
$1143.0ANDNAC_1 <= 497.0ANDAI_3 <=$
$597.0ANDAI_6 >= 454.0ANDAT_0 <=$
$1.363636THENBUZZ = 1IFAS(NA)_1 >=$
$0.001674ANDAS(NA)_1 >=$
$0.0022140000000000003ANDBL_6 >=$
$0.9964360000000001ANDAS(NAC)_2 <=$
$0.002803ANDNCD_1 >= 889.0ANDADL_2 <=$
$1.236256ANDCS_3 >= 1.0THENBUZZ =$
$1IFNCD_5 >= 867.0ANDAS(NA)_1 <=$
$0.0007639999999999999ANDBL_2 >=$
$0.9970030000000001ANDAI_1 <=$
$353.0ANDNA_5 <= 979.0ANDAI_5 <=$
$579.0ANDAI_2 <= 506.0ANDAI_6 >=$
$131.0ANDNA_5 >= 486.0THENBUZZ =$
$1IFNA_1 >= 679.0ANDADL_2 <=$
$1.1443709999999998ANDAS(NAC)_1 <=$
$0.001179ANDAS(NAC)_6 >=$
$0.000817ANDAS(NA)_4 >= 0.001108ANDAT_5 <=$
$1.18203ANDADL_6 <= 1.210417ANDBL_0 >=$
$0.9930389999999999THENBUZZ = 1IFNAC_6 >=$
$398.0ANDAT_0 <= 1.004802ANDAI_0 <=$
$186.0ANDAS(NAC)_5 >= 0.000165ANDAS(NA)_6 <=$
$0.001887ANDNCD_5 <= 915.0ANDAT_6 <=$
$1.00639ANDNA_3 >= 101.0ANDAI_5 >=$
$6.0ANDNA_5 >= 151.0ANDBL_6 >=$
$1.0THENBUZZ = 1IFNAC_6 >=$
$729.0ANDAI_5 <= 659.0ANDAI_6 >=$
$244.0ANDAT_6 <= 1.1976879999999999ANDNAD_3 <=$
$1166.0ANDNAC_5 <= 1418.0ANDAS(NAC)_4 <=$
$0.0010199999999999999ANDCS_3 >=$
$1.0ANDNAC_3 <= 1333.0ANDAS(NAC)_4 <=$
$0.001019ANDCS_2 >= 1.0ANDCS_4 >=$
$1.0ANDCS_5 >= 1.0ANDNAD_3 <=$
$1162.0THENBUZZ = 1IFTRUETHENBUZZ = 0$