**National Institute of Technology Karnataka, Surathkal**



Department of Computer Science and Engineering

CS351 - Machine Learning

**Modified DFS-based term weighting scheme for text classification (MDFS)**

Submitted to:

Dr. M. Venkatesan

Submitted by:

Abhishek Kumar - 181CO201

Sasidhar Swarangi - 181CO245

# 1.MDFS- INTRODUCTION

With the rapid growth of textual data on the Internet, properly organizing,managing, and utilizing them is becoming a great challenge. Text classification (TC) is the task of automatically classifying a set of textual documents into different classes from a predefined set. As a first and vital step of TC, text representation converts the content of a textual document into a compact format so that the textual document can be classified by a classifier. Of the numerous text representation methods, the vector space model (VSM) (Salton & McGill, 1984) is widely used. In the VSM, the content of a textual document is represented as a term (feature) vector in the term space, where each term refers to a word occurring in the document (Jiang et al., 2013, 2016b; Wang et al., 2015b), and the term value corresponds to its weight, indicating the importance of the word in distinguishing document categories. One of the most common ways to indicate this weight is term frequency (TF). However, TF alone is insufficient because the terms that occur more often will have a very large weight in a document. Therefore, term weighting is an important factor in improving the effectiveness of TC by assigning appropriate weights to different terms (Jiang et al., 2016a, 2019a).

Based on whether class information in the document is used, related work can be broadly divided into two main categories: namely unsupervised term weighting and supervised term weighting (Dogan & Uysal, 2019). Unsupervised term weighting mainly includes TF–IDF (inverse document frequency-based TF) (Salton & Buckley, 1988)and its variants. Unsupervised term weighting ignores the available class information of training documents and is thus often ineffective for TC. To make use of the class information of training documents,the study of supervised term weighting schemes has attracted increasing attention. Debole and Sebastiani (2003) proposed TF–CHI(Chi-square statistic-based TF), TF–IG (information gain-based TF),and TF–GR (gain ratio-based TF); Lan et al. (2009) proposed TF–RF (relevance frequency-based TF); Liu et al. (2009) proposed TF–PB(probability-based TF); Wang and Zhang (2013) proposed TF–ICF (inverse class frequency-based TF); Ren and Sohrab (2013) proposed TF–IDF–ICF (inverse document frequency and inverse class frequency based eTF) and TF–IDF–ICSDF (inverse document frequency and inverse class space density frequency-based TF); Wang et al. (2015a) proposed TF–DC (distributional concentration-based TF) and TF–BDC (balanced distribution concentration-based TF); Chen et al. (2016) proposed TF–IGM (inverse gravity moment-based TF); and Dogan and Uysal (2019)proposed TF–IGM𝑖𝑚𝑝 (improved inverse gravity moment-based TF).Although there exist many term weighting schemes for TC, finding a more effective and practical term weighting scheme remains a great challenge. By analyzing the existing term weighting schemes, we found that most of them do not take full advantage of the distribution information of terms in all training documents. The distinguishing feature selector (DFS) (Uysal & Günal, 2012) is a well-accepted term (feature) selection method, which assigns a high score to a term that frequently occurs in a single class and does not occur in the other classes. This Raises the question of whether using the term selection score of DFS directly as the term weight can provide better performance. To answer this question, we first adapted it as a term weighting scheme, namely TF–DFS (DFS-based TF), and found that there exist some defects in DFS when it comes to term weighting. To polish these defects, we propose a modified version of DFS (MDFS). Specifically, we first decomposed the term selection score of DFS into multiple (i.e., the number of classes)class-specific scores. Then, we calculated each class-specific score from‘‘positive’’ and ‘‘negative’’ perspectives and defined the final class specific score as their product. Finally, we assigned different weights to different class-specific scores and used the weighted sum across all class-specific scores as the whole term selection score of MDFS.Based on MDFS, we propose a new term weighting scheme simply called TF–MDFS. Extensive comparison results validate the advantages of TF–MDFS in terms of classification accuracy of widely used base classifiers, such as multinomial naive Bayes (MNB), support vector machines (SVM) and logistic regression (LR).

# 2. PROPOSED SCHEME AS PER THE PAPER

The DFS argues that an ideal term (feature) selection method should assign high scores to distinctive terms while assigning lower scores to irrelevant ones. Specifically, four requirements must be satisfied in DFS:

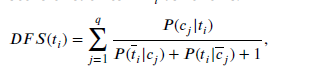
1. If a term occurs frequently in a single class and does not occur in other classes, it is instinctive and must be assigned a high score.

2. If a term occurs frequently in all classes, it is irrelevant and must be assigned a low score.

3. If a term occurs rarely in a single class and does not occur in other classes, it is irrelevant and must be assigned a low score.

4. If a term occurs in some of the classes, it is relatively distinctive and must be assigned a medium score. To meet the above four requirements, DFS defines the term selection

score of each term 𝑡𝑖 as follows:



where 𝑞 is the total number of classes, 𝑃 (𝑐𝑗 |𝑡𝑖) is the conditional probability of class 𝑐𝑗 given the presence of term 𝑡𝑖, 𝑃 (𝑡𝑖|𝑐𝑗 ) is the conditional probability of the absence of term 𝑡𝑖 given class 𝑐𝑗 , and 𝑃 (𝑡𝑖|𝑐𝑗 ) is the conditional probability of term 𝑡𝑖 given the absence of class 𝑐𝑗

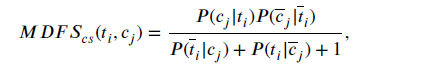
DFS (Uysal & Günal, 2012) has been proved to be an effective and efficient term (feature) selection method. This raises the question of whether using 𝐷𝐹𝑆(𝑡𝑖) directly as the weight of term 𝑡𝑖 can provide better performance. To answer this question, we adapted it as a term weighting scheme, which we call TF–DFS (DFS-based TF). The detailed formula is



The experimental results in Section 4 show that the performance of TF–DFS, compared to the raw TF, does not produce an expected improvement. Why does TF–DFS perform so poorly? The fundamental reason is that DFS assigns scores to all terms between 0.5 and 1.0 according to their significance (Uysal & Günal, 2012). With the DFS evaluated in a small range of values, a natural question arises: ‘‘is the specificity of a term in a class adequately demonstrated?’’ To answer this question, a simple example could be helpful to get some intuitive feeling. Suppose there are 100 documents, 10 of which belong to class 𝑐1 and 90 of which belong to class 𝑐2. The number of documents containing term 𝑡𝑖 in both classes is 5. It is obvious that the specificity score of term 𝑡𝑖 in class 𝑐1 should be much higher than that in class 𝑐2. However, according to 𝐷𝐹𝑆(𝑡𝑖) calculated by Eq. (1), the specificity scores of term 𝑡𝑖 for each class are relatively close, only 0.3214 and 0.2045, respectively.

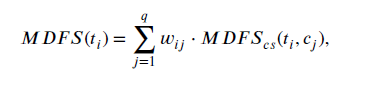
# 3. ALGORITHM OF MDFS

The existing 𝐷𝐹𝑆(𝑡𝑖) does not satisfy this requirement. Therefore, we must modify the existing 𝐷𝐹𝑆(𝑡𝑖). Specifically, we first decomposed 𝐷𝐹𝑆(𝑡𝑖) into 𝑞 class-specific scores 𝑀𝐷𝐹𝑆𝑐𝑠(𝑡𝑖, 𝑐𝑗 ). Then, we calculated each class-specific score 𝑀𝐷𝐹𝑆𝑐𝑠(𝑡𝑖, 𝑐𝑗 ) from both ‘‘positive’’ and ‘‘negative’’ perspectives and defined the final class-specific score as their product.The detailed formula is

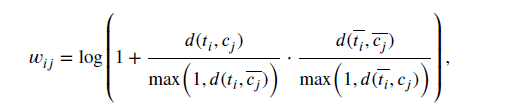


where 𝑃 (𝑐𝑗 |𝑡𝑖) is the conditional probability of the absence of class 𝑐𝑗 given the absence of 𝑡𝑖. As can be seen from Eq. (3), the conditional probability 𝑃 (𝑐𝑗 |𝑡𝑖) reflects the inter-class distribution of documents containing term 𝑡𝑖. The multiplication factor of 𝑃 (𝑐𝑗 |𝑡𝑖) reflects the inter-class distribution of documents not containing term 𝑡𝑖. For each class 𝑐𝑗 , this will make better use of the distribution information of term 𝑡𝑖 in all training documents.Now, again for the above example, according to 𝑀𝐷𝐹𝑆𝑐𝑠(𝑡𝑖, 𝑐𝑗 )calculated by Eq. (3), the specificity scores of term 𝑡𝑖 for each class are widely spaced at 0.3036 and 0.0114, respectively. We can see that the specificity score of term 𝑡𝑖 in class 𝑐1 decreases only slightly, while the specificity score in class 𝑐2 decreases to a very small value. This is the reason why 𝑀𝐷𝐹𝑆𝑐𝑠(𝑡𝑖, 𝑐𝑗 ) is more consistent with the evaluation criteria of the class-specific specificity of terms.

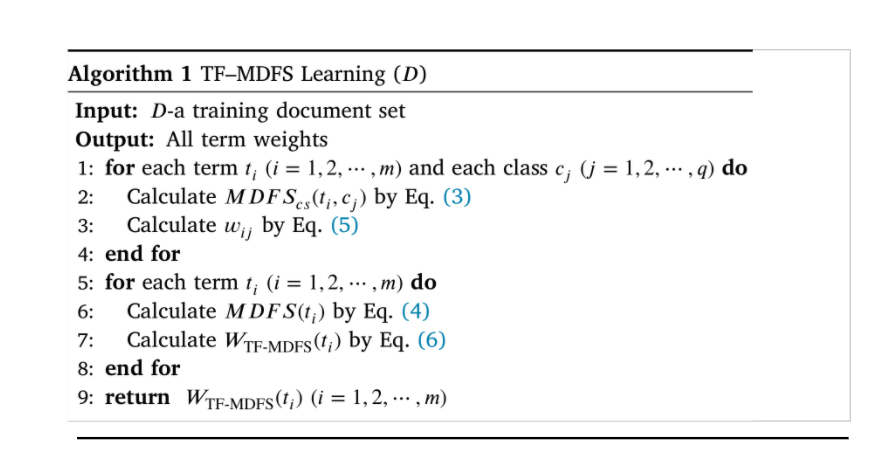
Besides, we argue that for each term, different class-specific scores should have different contributions (importance) to the whole term score. Therefore, we should assign different weights for different class specific scores and then use the weighted sum across all class-specific scores as the whole term score. Based on this premise, we propose a modified DFS (MDFS). The detailed formula is



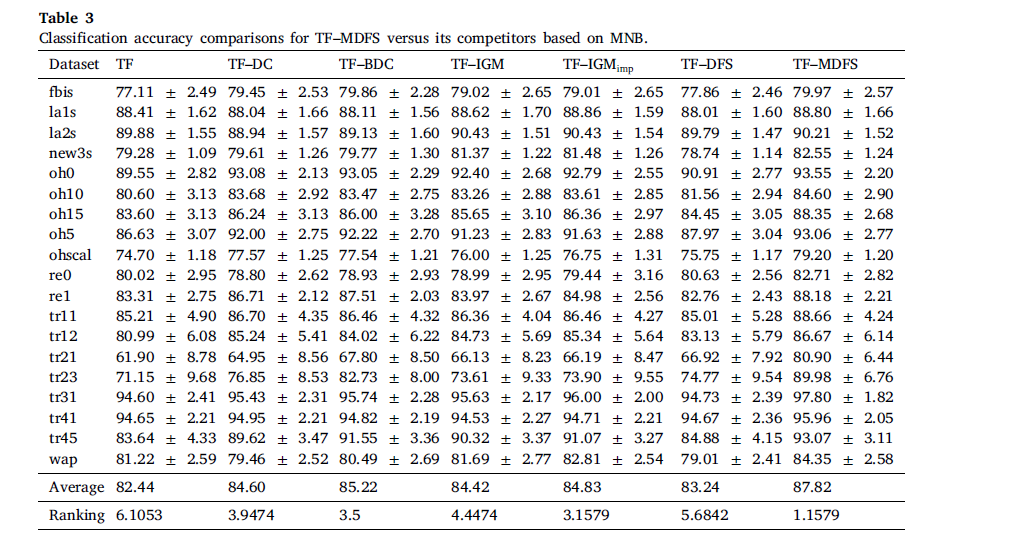
where 𝑤𝑖𝑗 represents the specific weighting factor of term 𝑡𝑖 for class 𝑐𝑗 , which can be defined as



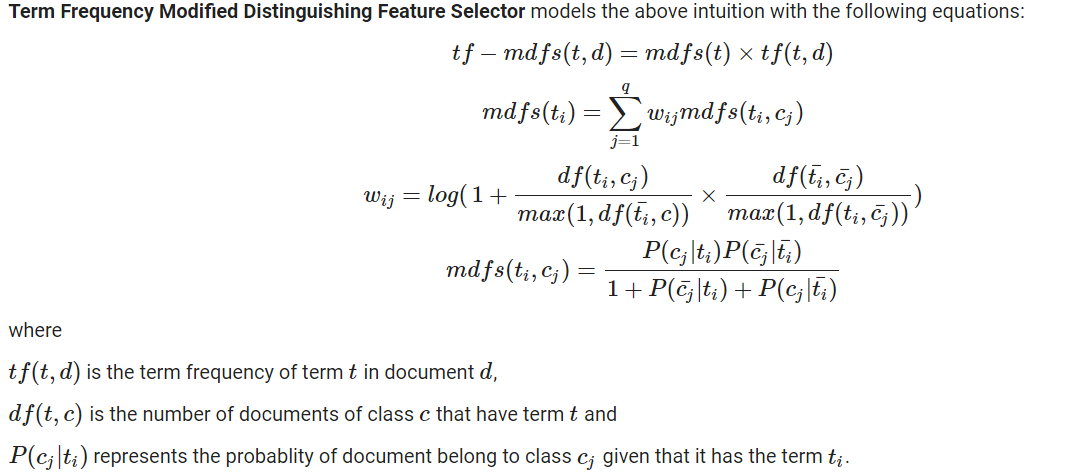
# 4. COMPUTING TF-MDFS



# 4.1. COMPARING ACCURACY



# 5. NUMERICAL ANALYSIS



## 5.1. DATASETS USED

We have used datasets like:

1. **Amazon Review Dataset**

Source:<https://raw.githubusercontent.com/Gunjitbedi/Text-Classification/master/corpus.csv>

1. **Polarity v2.0 Movie Review Dataset**

Source: <https://www.kaggle.com/anindya2906/movie-review-polarity>

### **RSS Feed Topics Dataset**

### Source: <https://www.kaggle.com/brobear1995/rss-feed-topic-classifier>

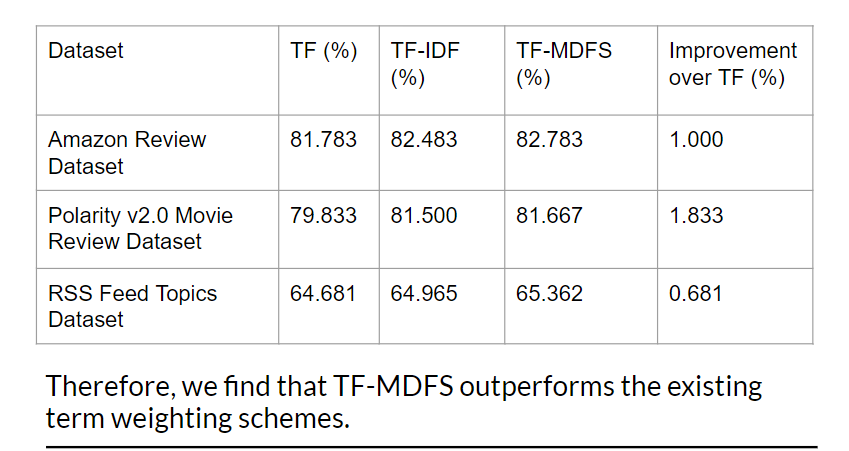
These DATASETS are most commonly used for text classification and analysis, we brought a comparison of the MDFS algorithm with the other algorithms

# 6. IMPLEMENTATION

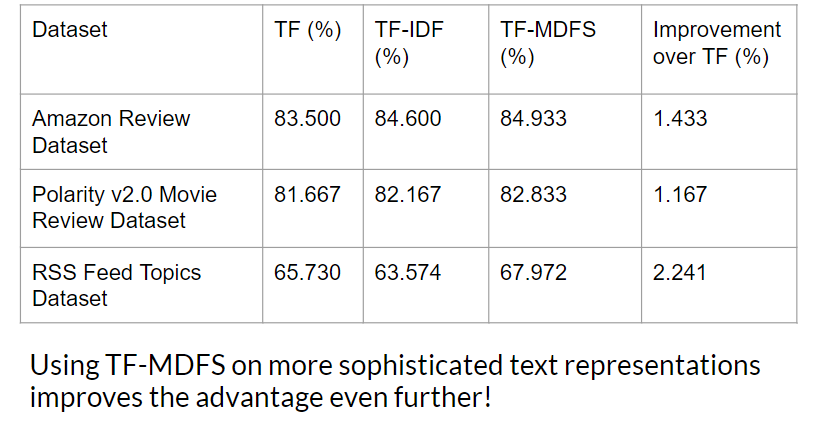
## 6.1. COLLAB NOTEBOOK

<https://colab.research.google.com/drive/1oms4SQnK_x56Ht-SyK97UKD5gmCFEWER#scrollTo=o2vTZBT-WnFT>

## 6.2. RESULTS

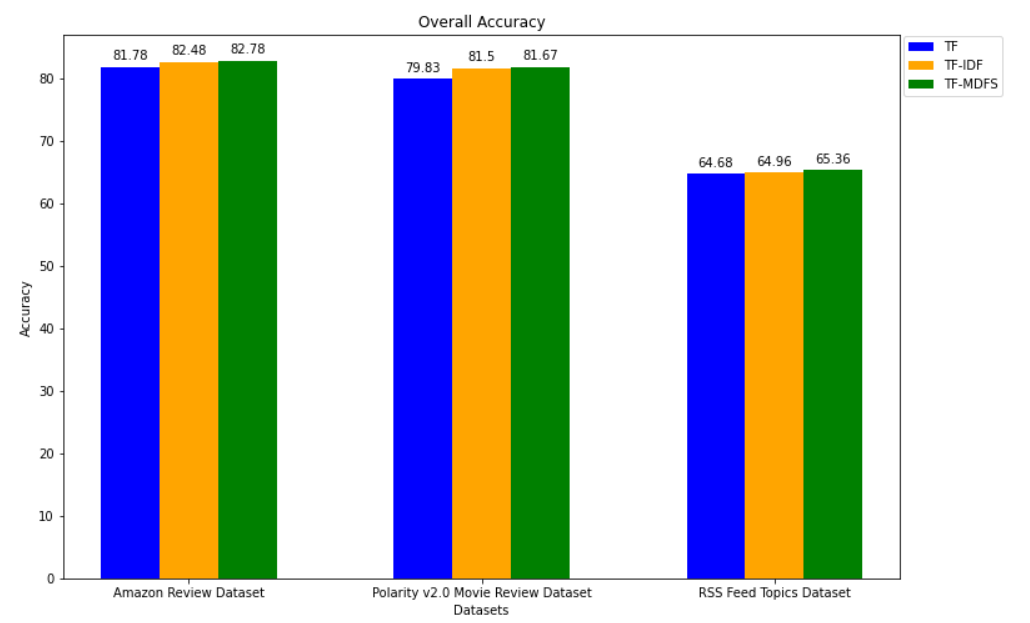


Accuracies for Bigram



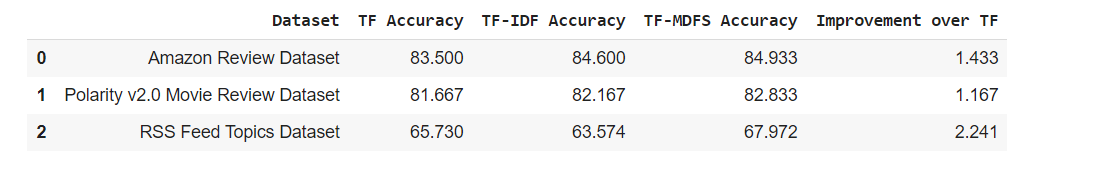
## 6.3. ACCURACIES COMPARISON BETWEEN THE ALGORITHMS AND DATASET

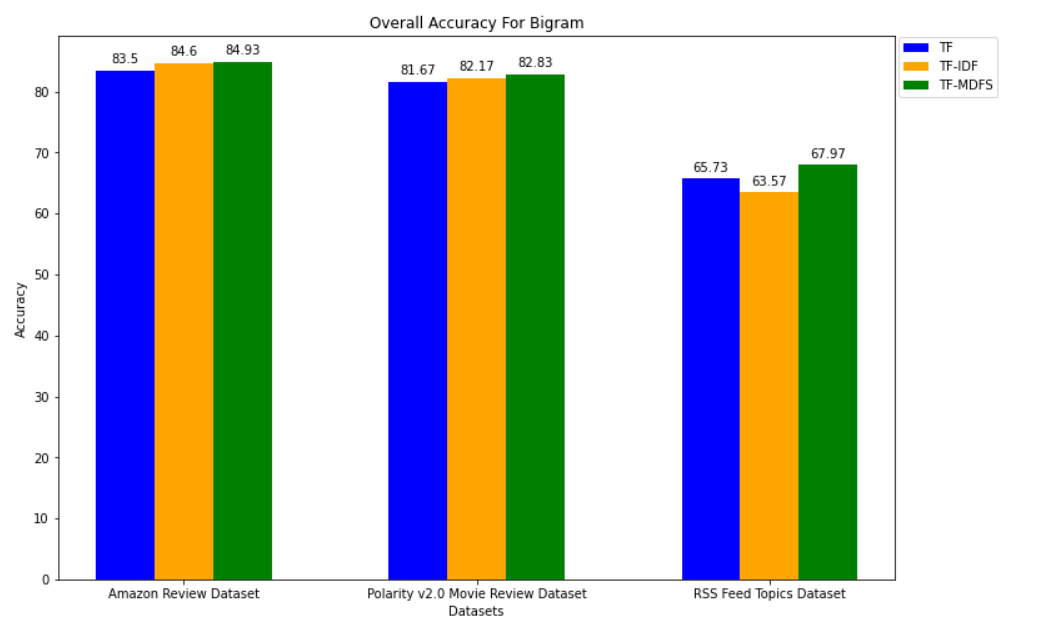
## 6.3.1 OVERALL ACCURACY



## 

## 6.3.2 OVERALL ACCURACY FOR BIAGRAM



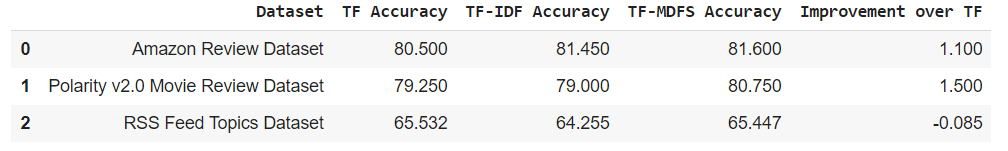


## 

## 

## 

## 6.3.3 Accuracy for Naive Bayes



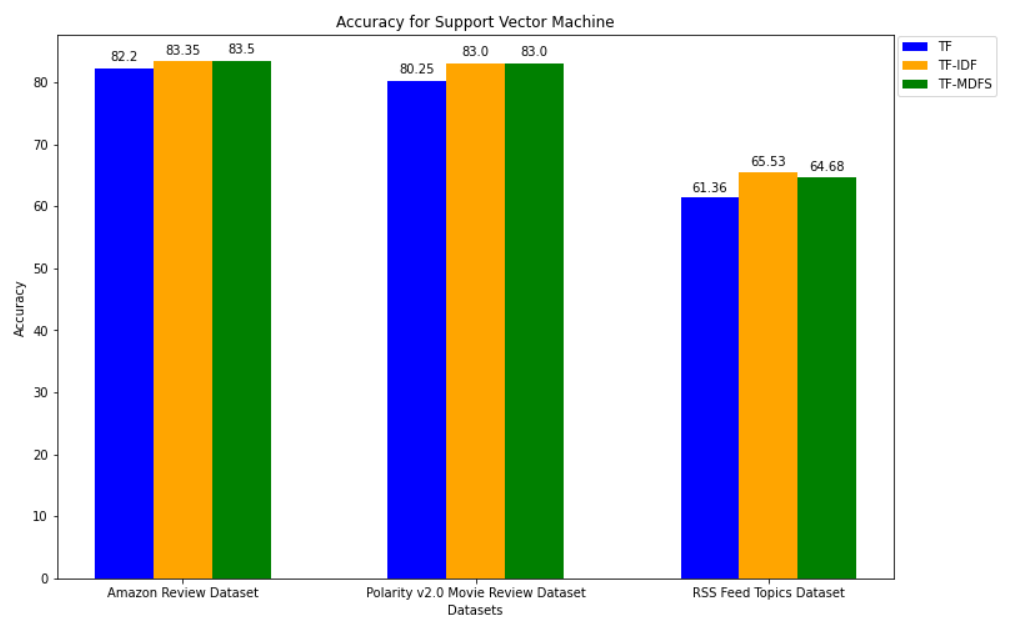
## 

## 

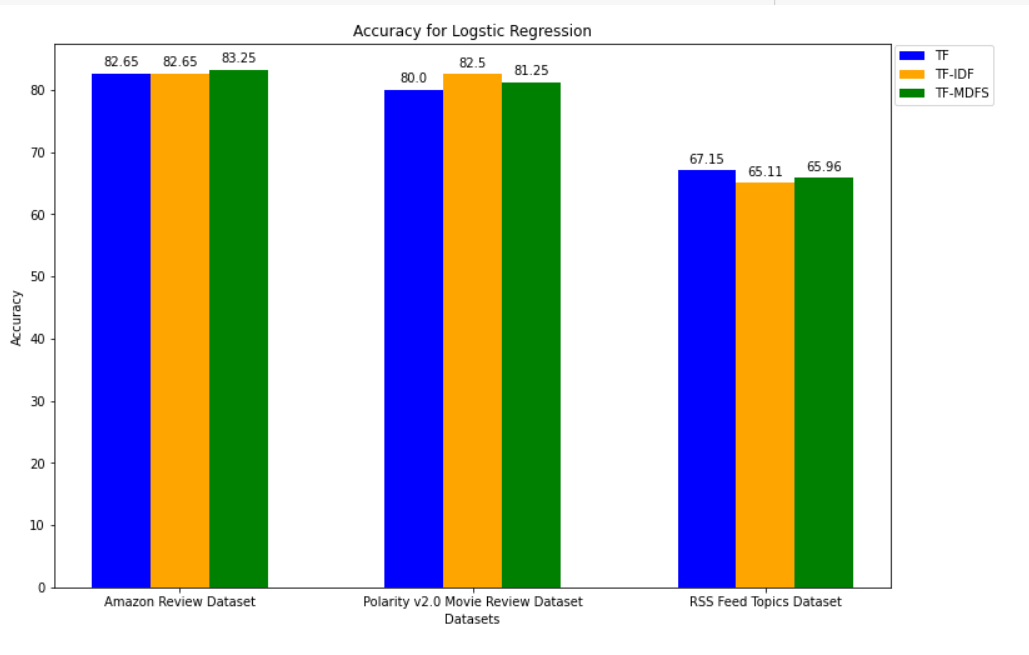
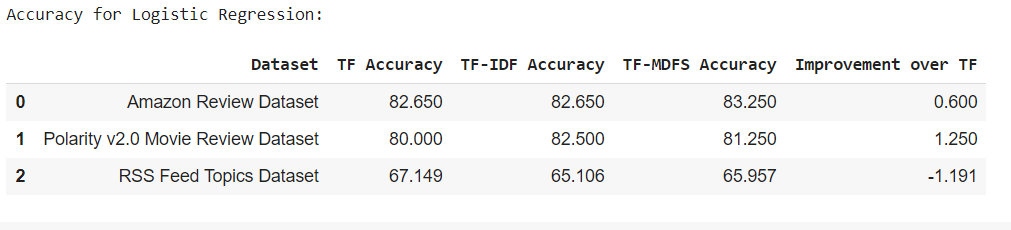
## 

## 6.3.4 Accuracy for Support Vector Machine

## 



## 6.3.5 Accuracy for Logistic Regression



# 7. REFERENCES

[1] Al-Zaidy, R., Fung, B. C. M., & Youssef, A. M. (2011). Towards discovering criminal

communities from textual data. In Proceedings of the 2011 ACM symposium on applied

computing. (pp. 172–177).

[2] Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., & García, S. (2011). KEEL

data-mining software tool: Data set repository, integration of algorithms and

experimental analysis framework. Multiple-Valued Logic and Soft Computing, 17(2–3),

255–287.

[3] Aseervatham, S., Gaussier, É., Antoniadis, A., Burlet, M., & Denneulin, Y. (2012).

Logistic regression and text classification. In Textual information access: Statistical

models (pp. 61–84).

[4] Chen, K., Zhang, Z., Long, J., & Zhang, H. (2016). Turning from TF-IDF to TF-IGM for

term weighting in text classification. Expert Systems with Applications, 66, 245–260.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3),

273–297.

[5] Debole, F., & Sebastiani, F. (2003). Supervised term weighting for automated text

categorization. In Proceedings of the 2003 ACM symposium on applied computing.

(pp. 784–788).

[6] Sklearn https://scikit-learn.org/stable/tutorial/text\_analytics/working\_with\_text\_data.html

## 