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–IGM$_{imp}$ (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:

$$DFS(t_i) = \sum_{j=1}^{q} \frac{P(c_j|t_i)}{P(\bar{t}_i|c_j) + P(t_i|\bar{c}_j) + 1},$$

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

$$W_{\text{TF-DFS}}(t_i) = f_i \cdot DFS(t_i).$$

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

$$MDFS_{cs}(t_i, c_j) = \frac{P(c_j|t_i)P(\bar{c}_j|\bar{t}_i)}{P(\bar{t}_i|c_j) + P(t_i|\bar{c}_j) + 1},$$

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

$$MDFS(t_i) = \sum_{j=1}^{q} w_{ij} \cdot MDFS_{cs}(t_i, c_j),$$

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

$$w_{ij} = \log\left(1 + \frac{d(t_i, c_j)}{\max\left(1, d(t_i, \overline{c_j})\right)} \cdot \frac{d(\overline{t_i}, \overline{c_j})}{\max\left(1, d(\overline{t_i}, c_j)\right)}\right),$$

4. COMPUTING TF-MDFS

Algorithm 1 TF–MDFS Learning (D)

```
Input:  $D$-a training document set
Output:  All term weights
1: for each term $t_i$ $(i = 1, 2, \cdots, m)$ and each class $c_j$ $(j = 1, 2, \cdots, q)$ do
2:    Calculate $MDFS_{cs}(t_i, c_j)$ by Eq. (3)
3:    Calculate $w_{ij}$ by Eq. (5)
4: end for
5: for each term $t_i$ $(i = 1, 2, \cdots, m)$ do
6:    Calculate $MDFS(t_i)$ by Eq. (4)
7:    Calculate $W_{\text{TF-MDFS}}(t_i)$ by Eq. (6)
8: end for
9: return  $W_{\text{TF-MDFS}}(t_i)$ $(i = 1, 2, \cdots, m)$
```

4.1. COMPARING ACCURACY

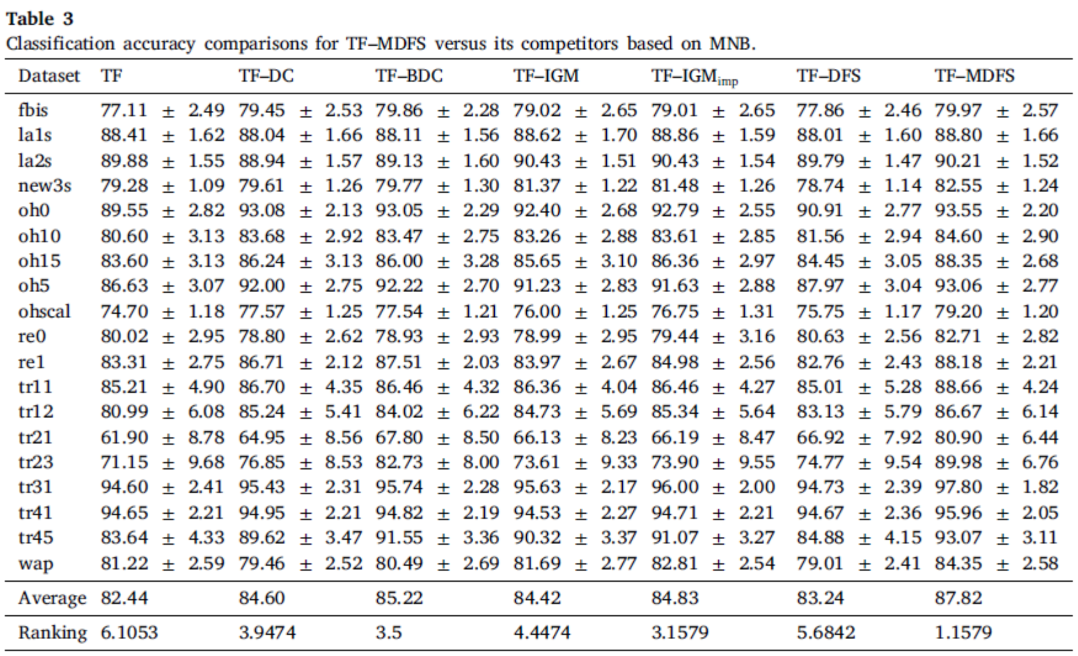


Table 3
Classification accuracy comparisons for TF–MDFS versus its competitors based on MNB.

Dataset	TF	TF–DC	TF–BDC	TF–IGM	TF–IGM$_{imp}$	TF–DFS	TF–MDFS
fbis	77.11 ± 2.49	79.45 ± 2.53	79.86 ± 2.28	79.02 ± 2.65	79.01 ± 2.65	77.86 ± 2.46	79.97 ± 2.57
la1s	88.41 ± 1.62	88.04 ± 1.66	88.11 ± 1.56	88.62 ± 1.70	88.86 ± 1.59	88.01 ± 1.60	88.80 ± 1.66
la2s	89.88 ± 1.55	88.94 ± 1.57	89.13 ± 1.60	90.43 ± 1.51	90.43 ± 1.54	89.79 ± 1.47	90.21 ± 1.52
new3s	79.28 ± 1.09	79.61 ± 1.26	79.77 ± 1.30	81.37 ± 1.22	81.48 ± 1.26	78.74 ± 1.14	82.55 ± 1.24
oh0	89.55 ± 2.82	93.08 ± 2.13	93.05 ± 2.29	92.40 ± 2.68	92.79 ± 2.55	90.91 ± 2.77	93.55 ± 2.20
oh10	80.60 ± 3.13	83.68 ± 2.92	83.47 ± 2.75	83.26 ± 2.88	83.61 ± 2.85	81.56 ± 2.94	84.60 ± 2.90
oh15	83.60 ± 3.13	86.24 ± 3.13	86.00 ± 3.28	85.65 ± 3.10	86.36 ± 2.97	84.45 ± 3.05	88.35 ± 2.68
oh5	86.63 ± 3.07	92.00 ± 2.75	92.22 ± 2.70	91.23 ± 2.83	91.63 ± 2.88	87.97 ± 3.04	93.06 ± 2.77
ohscal	74.70 ± 1.18	77.57 ± 1.25	77.54 ± 1.21	76.00 ± 1.25	76.75 ± 1.31	75.75 ± 1.17	79.20 ± 1.20
re0	80.02 ± 2.95	78.80 ± 2.62	78.93 ± 2.93	78.99 ± 2.95	79.44 ± 3.16	80.63 ± 2.56	82.71 ± 2.82
re1	83.31 ± 2.75	86.71 ± 2.12	87.51 ± 2.03	83.97 ± 2.67	84.98 ± 2.56	82.76 ± 2.43	88.18 ± 2.21
tr11	85.21 ± 4.90	86.70 ± 4.35	86.46 ± 4.32	86.36 ± 4.04	86.46 ± 4.27	85.01 ± 5.28	88.66 ± 4.24
tr12	80.99 ± 6.08	85.24 ± 5.41	84.02 ± 6.22	84.73 ± 5.69	85.34 ± 5.64	83.13 ± 5.79	86.67 ± 6.14
tr21	61.90 ± 8.78	64.95 ± 8.56	67.80 ± 8.50	66.13 ± 8.23	66.19 ± 8.47	66.92 ± 7.92	80.90 ± 6.44
tr23	71.15 ± 9.68	76.85 ± 8.53	82.73 ± 8.00	73.61 ± 9.33	73.90 ± 9.55	74.77 ± 9.54	89.98 ± 6.76
tr31	94.60 ± 2.41	95.43 ± 2.31	95.74 ± 2.28	95.63 ± 2.17	96.00 ± 2.00	94.73 ± 2.39	97.80 ± 1.82
tr41	94.65 ± 2.21	94.95 ± 2.21	94.82 ± 2.19	94.53 ± 2.27	94.71 ± 2.21	94.67 ± 2.36	95.96 ± 2.05
tr45	83.64 ± 4.33	89.62 ± 3.47	91.55 ± 3.36	90.32 ± 3.37	91.07 ± 3.27	84.88 ± 4.15	93.07 ± 3.11
wap	81.22 ± 2.59	79.46 ± 2.52	80.49 ± 2.69	81.69 ± 2.77	82.81 ± 2.54	79.01 ± 2.41	84.35 ± 2.58
Average	82.44	84.60	85.22	84.42	84.83	83.24	87.82
Ranking	6.1053	3.9474	3.5	4.4474	3.1579	5.6842	1.1579

5. NUMERICAL ANALYSIS

Term Frequency Modified Distinguishing Feature Selector models the above intuition with the following equations:

$$tf - mdfs(t,d) = mdfs(t) \times tf(t,d)$$

$$mdfs(t_i) = \sum_{j=1}^{q} w_{ij} mdfs(t_i, c_j)$$

$$w_{ij} = log(\,1 + \frac{df(t_i, c_j)}{max(1, df(\bar{t_i}, c))} \times \frac{df(\bar{t_i}, \bar{c_j})}{max(1, df(t_i, \bar{c_j}))})$$

$$mdfs(t_i, c_j) = \frac{P(c_j|t_i)P(\bar{c_j}|\bar{t_i})}{1 + P(\bar{c_j}|t_i) + P(c_j|\bar{t_i})}$$

where

$tf(t,d)$ is the term frequency of term t in document d,

$df(t,c)$ is the number of documents of class c that have term t and

$P(c_j|t_i)$ represents the probablity of document belong to class c_j given that it has the term t_i.

5.1. DATASETS USED

We have used datasets like:

1. **Amazon Review Dataset**
   Source:https://raw.githubusercontent.com/Gunjitbedi/Text-Classification/master/corpus.csv
2. **Polarity v2.0 Movie Review Dataset**
   Source: https://www.kaggle.com/anindya2906/movie-review-polarity
3. **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

Dataset	TF (%)	TF-IDF (%)	TF-MDFS (%)	Improvement over TF (%)
Amazon Review Dataset	81.783	82.483	82.783	1.000
Polarity v2.0 Movie Review Dataset	79.833	81.500	81.667	1.833
RSS Feed Topics Dataset	64.681	64.965	65.362	0.681

Therefore, we find that TF-MDFS outperforms the existing term weighting schemes.

Accuracies for Bigram

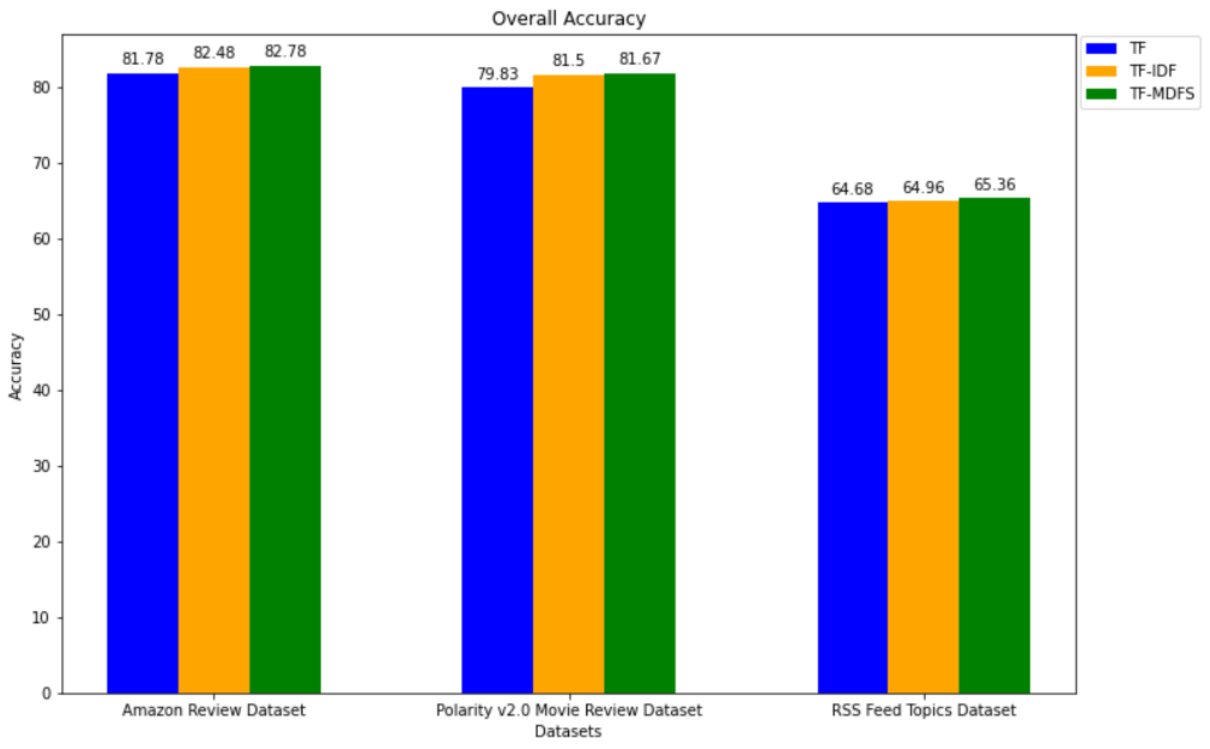
Dataset	TF (%)	TF-IDF (%)	TF-MDFS (%)	Improvement over TF (%)
Amazon Review Dataset	83.500	84.600	84.933	1.433
Polarity v2.0 Movie Review Dataset	81.667	82.167	82.833	1.167
RSS Feed Topics Dataset	65.730	63.574	67.972	2.241

Using TF-MDFS on more sophisticated text representations improves the advantage even further!

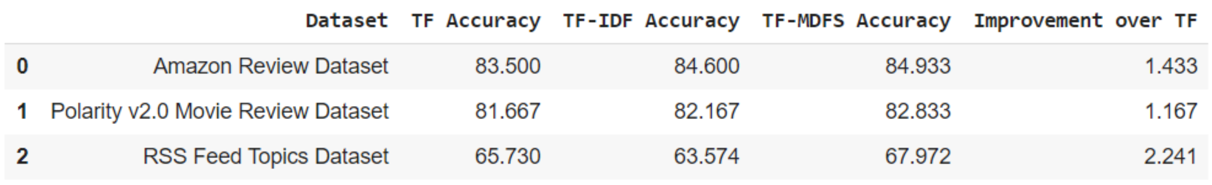
6.3. ACCURACIES COMPARISON BETWEEN THE ALGORITHMS AND DATASET

6.3.1 OVERALL ACCURACY

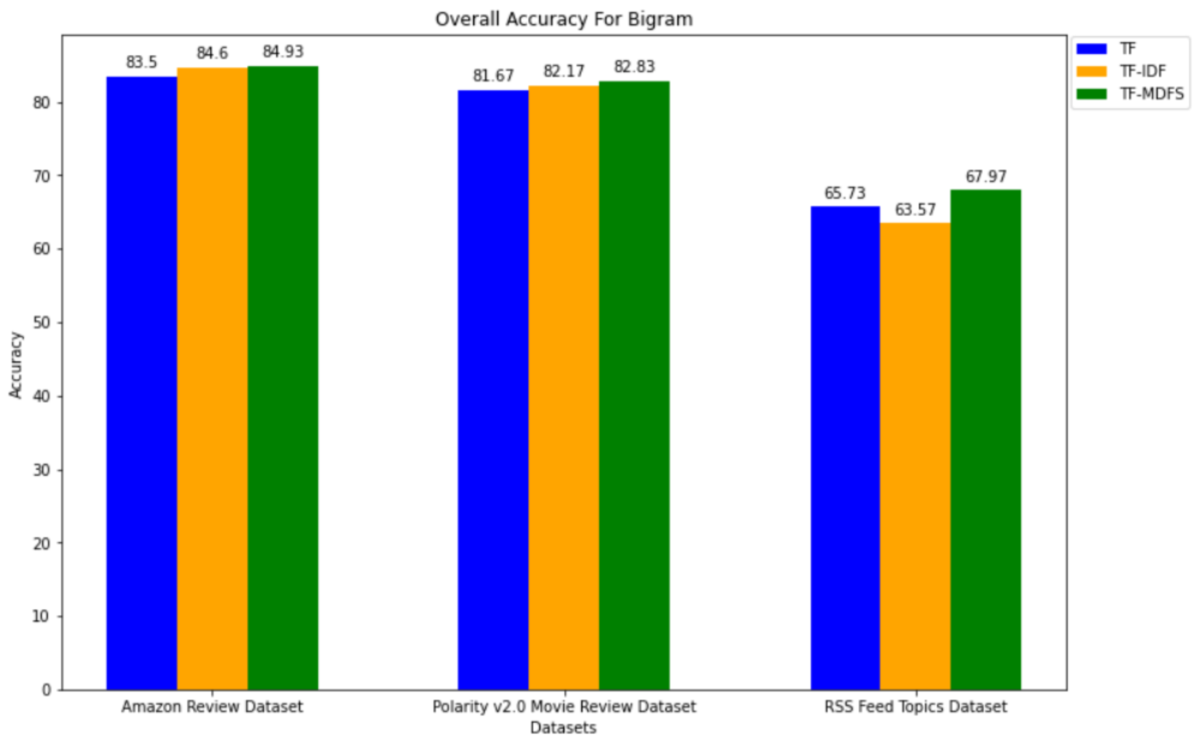
	Dataset	TF Accuracy	TF-IDF Accuracy	TF-MDFS Accuracy	Improvement over TF
0	Amazon Review Dataset	81.783	82.483	82.783	1.000
1	Polarity v2.0 Movie Review Dataset	79.833	81.500	81.667	1.833
2	RSS Feed Topics Dataset	64.681	64.965	65.362	0.681



6.3.2 OVERALL ACCURACY FOR BIAGRAM

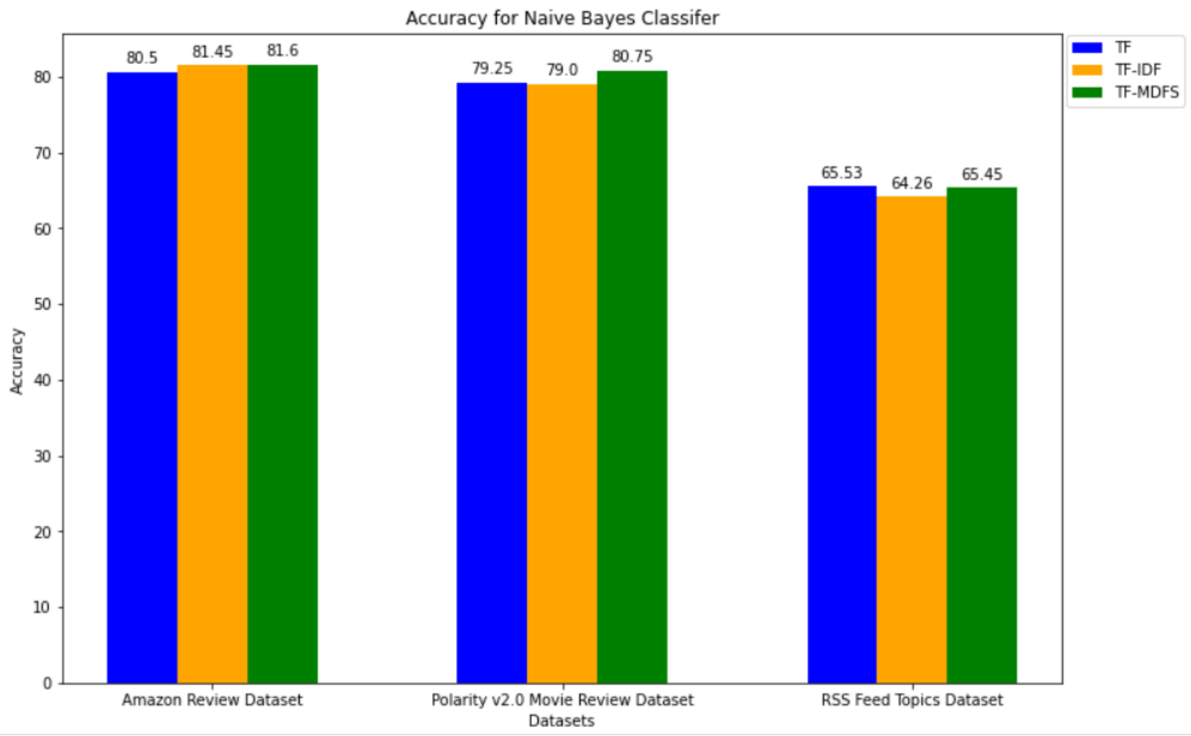


	Dataset	TF Accuracy	TF-IDF Accuracy	TF-MDFS Accuracy	Improvement over TF
0	Amazon Review Dataset	83.500	84.600	84.933	1.433
1	Polarity v2.0 Movie Review Dataset	81.667	82.167	82.833	1.167
2	RSS Feed Topics Dataset	65.730	63.574	67.972	2.241



6.3.3 Accuracy for Naive Bayes

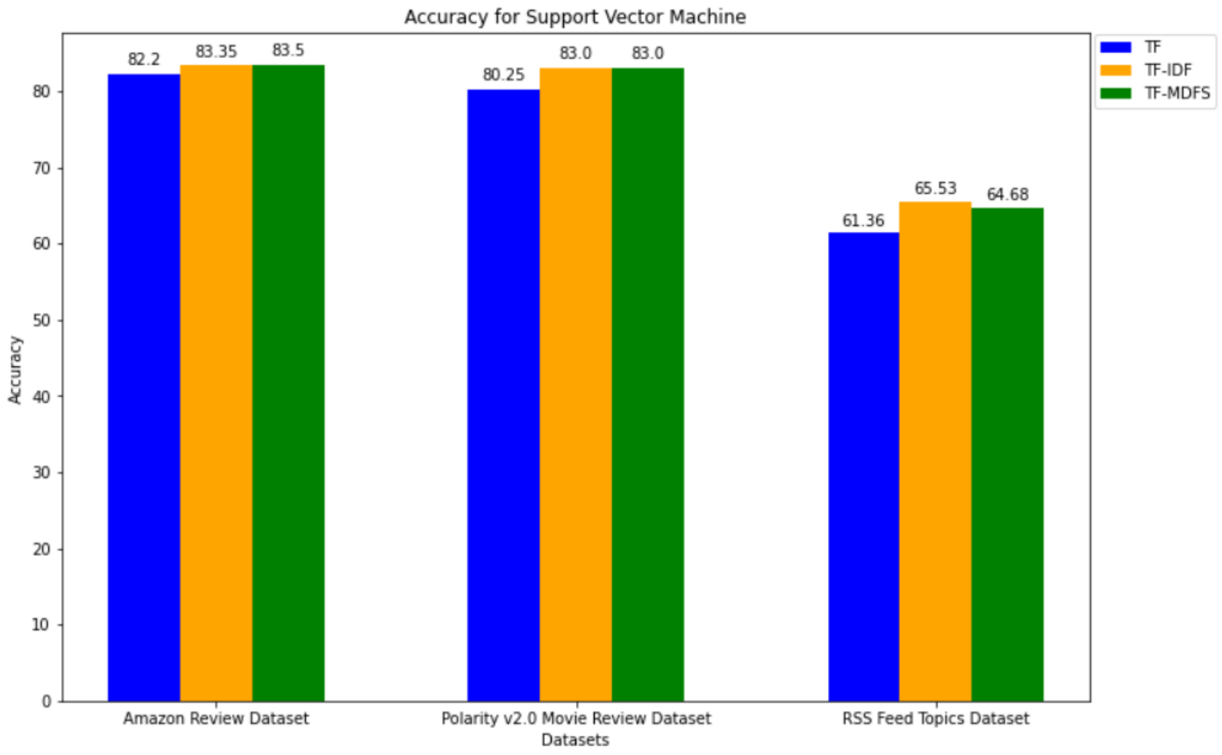
	Dataset	TF Accuracy	TF-IDF Accuracy	TF-MDFS Accuracy	Improvement over TF
0	Amazon Review Dataset	80.500	81.450	81.600	1.100
1	Polarity v2.0 Movie Review Dataset	79.250	79.000	80.750	1.500
2	RSS Feed Topics Dataset	65.532	64.255	65.447	-0.085



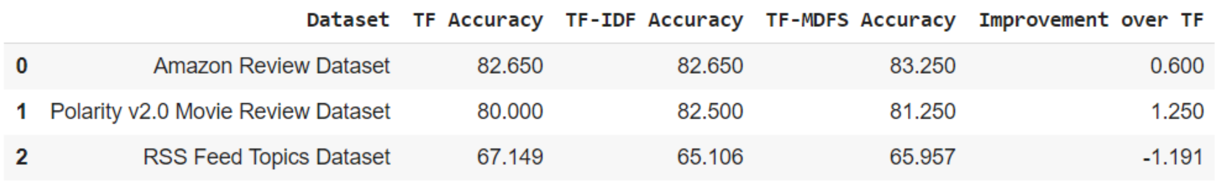
6.3.4 Accuracy for Support Vector Machine

```
Accuracy for Support Vector Machine:
```

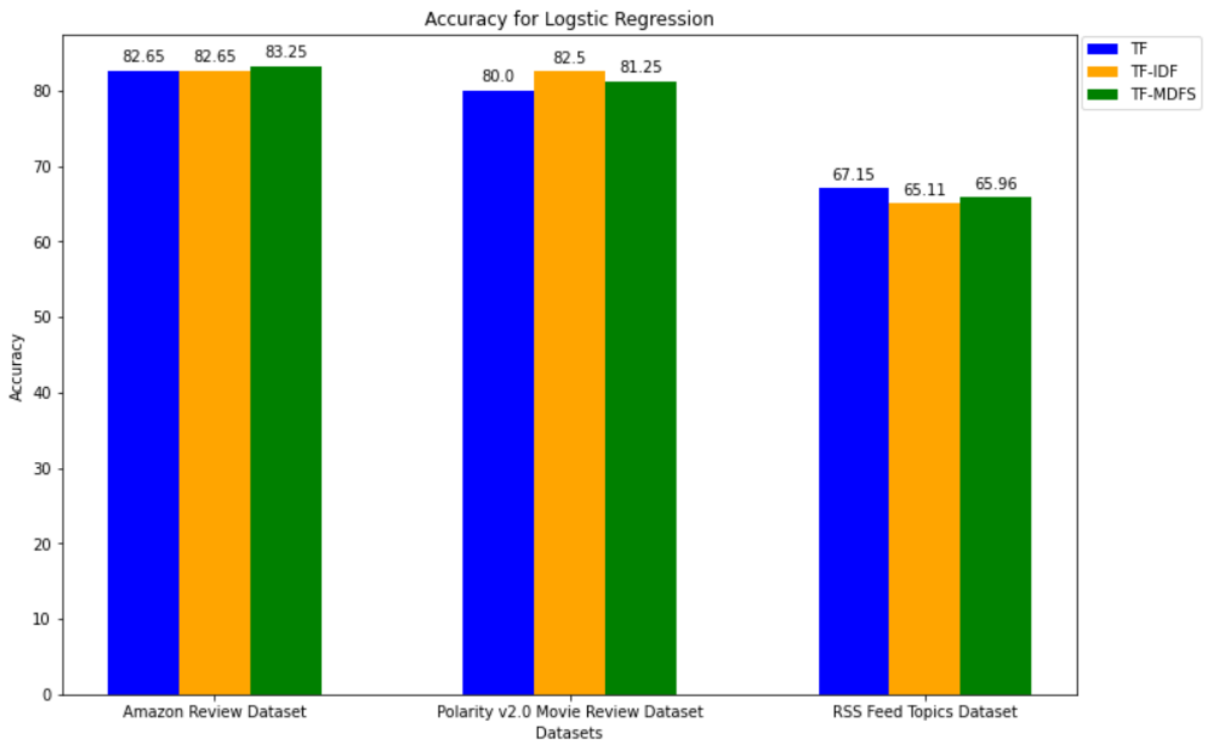
	Dataset	TF Accuracy	TF-IDF Accuracy	TF-MDFS Accuracy	Improvement over TF
0	Amazon Review Dataset	82.200	83.350	83.500	1.300
1	Polarity v2.0 Movie Review Dataset	80.250	83.000	83.000	2.750
2	RSS Feed Topics Dataset	61.362	65.532	64.681	3.319



6.3.5 Accuracy for Logistic Regression



	Dataset	TF Accuracy	TF-IDF Accuracy	TF-MDFS Accuracy	Improvement over TF
0	Amazon Review Dataset	82.650	82.650	83.250	0.600
1	Polarity v2.0 Movie Review Dataset	80.000	82.500	81.250	1.250
2	RSS Feed Topics Dataset	67.149	65.106	65.957	-1.191



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