

Abstract geometric lines in the top left corner of the slide, consisting of several overlapping, irregular polygons and lines in black.

# STUDY AND IMPLEMENTATION OF NAIVE BAYES CLASSIFIER

## **Members:**

- o Davide Checchia - 2078232
- o Kiamehr Javid - 2084294

# OUTLINE

- I. INTRODUCTION AND THEORY
- II. CHALLENGES AND SOLUTIONS
- III. RESULTS & ANALYSIS
- IV. CONCLUSION

# I. INTRODUCTION

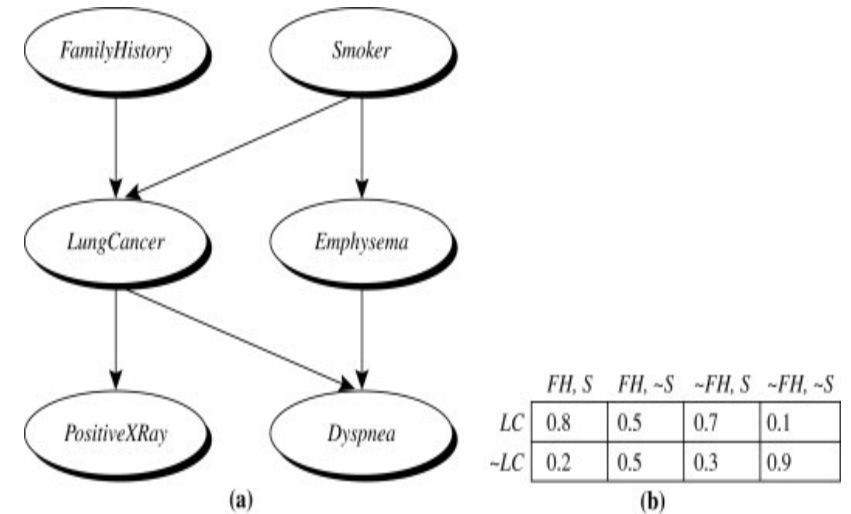
1. Main Goals
2. Naive Bayes Classifiers
3. Data Structure
4. Challenges Faced
5. Methods Used

# 1. MAIN GOALS

- Study, Explain and Implement the structure of a Multinomial Naive Bayes Classifier on two text classification tasks, consisting of six and two classes.
- Try different strategies to (possibly) improve the results obtained.
- Give a qualitative explanation of results, lay down the fundamentals for future analysis.

## 2. NAIVE BAYES CLASSIFIERS

- Naive Bayes Classifiers are a simple yet effective machine learning algorithm used for classification tasks.
- They are based on the Bayes' theorem and assume that the features are independent of each other, hence the term **naive**.
- Despite this assumption, Naive Bayes often performs well in practice and is widely used in various applications such as spam filtering, text classification, and sentiment analysis.



## 2. NBC - ASSUMPTIONS

All the features of the classified objects are **conditionally** and **positionally independent**:

- The Sun is round, big and shiny: all those features independently contribute to the chance of the object being the Sun, and the order does not change the information about it.

Words in a text (our case) are **correlated to the topic** at hand (or any considered classification):

- In a fraud/non-fraud classification, we expect to find that the words “money” and “currency” appear more often in fraudulent emails, given the number of emails considered.

Texts follow **multinomial distributions**, which can be approximated by counting (hence the name).

## 2. NBC - BAG OF WORDS

In a Multinomial NBC, the likelihood functions for the singular words are approximated by a simple counting of those for every single class:

- "Hello bag of words" labeled as 1.
- "Hello shopper of syllables" labeled as 2.

The corresponding bag of words will contain an occurrence number **one** for the word "Hello" in both classes 1 and 2, whereas bag will only have an occurrence number **one** in the class 1, and **zero** for class 2.

All the occurrences are then divided by the sum in a given class to obtain a **frequency**.

## 2. NBC - BAG OF WORDS

words	C0	C1	C2	C3	C4	C5
<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
voted	0.005910937	0.0035797082	0.0037986411	0.0035151184	0.0027090194	0.0034480145
taxes	0.004823035	0.0026251193	0.0055281038	0.0046071940	0.0027090194	0.0040542589
clinton	0.004460400	0.0035115233	0.0033662755	0.0024912975	0.0024965473	0.0012503789
obamacare	0.004242820	0.0027614892	0.0023780111	0.0015357313	0.0029214916	0.0007956957
texas	0.004242820	0.0036478931	0.0055898703	0.0048802130	0.0024965473	0.0046605032
barack	0.003807659	0.0060343652	0.0040457072	0.0026278070	0.0053649209	0.0024628675
hillary	0.003735132	0.0023523797	0.0025633107	0.0020135144	0.0024965473	0.0008714762
medicare	0.002864810	0.0028296741	0.0026868437	0.0014674766	0.0018591310	0.0012503789
wisconsin	0.002864810	0.0048752216	0.0024397776	0.0029008259	0.0035589079	0.0018566232
senate	0.002792283	0.0026933042	0.0023780111	0.0027643164	0.0014341868	0.0026902092
republican	0.002719756	0.0024887495	0.0031809759	0.0037198826	0.0013279507	0.0028417702
scott	0.002502176	0.0035797082	0.0022544781	0.0012627124	0.0029214916	0.0011745984
spending	0.002429649	0.0013977908	0.0034280420	0.0031055901	0.0013279507	0.0027659897





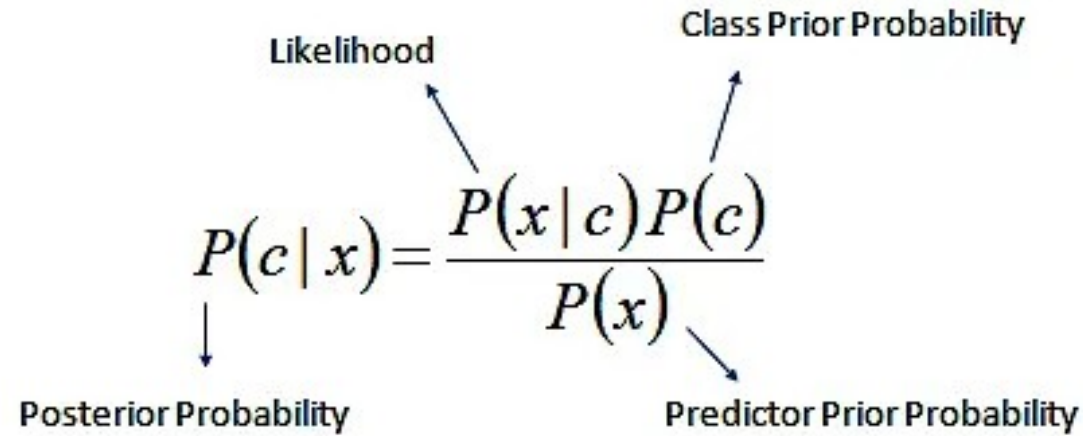
## 2. NBC – STOP WORDS

In the English language there are some words, such as “**the**”, “**as**”, “**or**” that are omnipresent, and thus are assumed not to be giving information on certain classifications.

These are called **stop words**, and they are removed from the bag of words in order to:

- Remove useless weight (computation cost)
- Exacerbate the differences between words in different classes (reliability).

## 2. NBC - BAYES THEOREM



The diagram shows the Bayes' Theorem formula with arrows pointing from descriptive labels to the corresponding parts of the equation:

$$P(c | x) = \frac{P(x | c)P(c)}{P(x)}$$

Labels and their corresponding parts in the formula:

- Likelihood** points to  $P(x | c)$
- Class Prior Probability** points to  $P(c)$
- Posterior Probability** points to  $P(c | x)$
- Predictor Prior Probability** points to  $P(x)$

$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \cdots \times P(x_n | c) \times P(c)$$

- Here **x** represents the given text, whereas **c** is the considered class.
- Note how the **independence** assumption lets us define the Likelihood as a **product**.

## 2. NBC - NORMALIZATION

In the English language we may find different words which have the same semantical registry/meaning. An example would be plural or adverbs, such as **nice, nicer, nicely**.

In our model, those words should be counted as a single class (**equivalence classes**).

In order to do this, we implement the library **SnowballC** to gather different terms under the same semantic umbrella, requiring a groupby on the counts.


## 2. NBC - FEATURE SELECTION

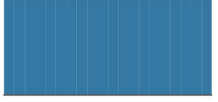

Some words may be more related to a certain class than the others, thus giving more information about it. Furthermore, it would be ideal to remove useless words (i.e. ones that do not give information about specific classes) → We associate to each term in each class a score, based on the **Mutual Information** function:

$$I(U;C) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} P(U = e_t, C = e_c) \log_2 \frac{P(U = e_t, C = e_c)}{P(U = e_t)P(C = e_c)},$$

$$I(U;C) = \frac{N_{11}}{N} \log_2 \frac{NN_{11}}{N_{1.}N_{.1}} + \frac{N_{01}}{N} \log_2 \frac{NN_{01}}{N_{0.}N_{.1}} \\ + \frac{N_{10}}{N} \log_2 \frac{NN_{10}}{N_{1.}N_{.0}} + \frac{N_{00}}{N} \log_2 \frac{NN_{00}}{N_{0.}N_{.0}}$$

# 3. DATASET STRUCTURE

# Labels	Text	Text_Tag
Label	Text	Label/Tag
	<b>10223</b> unique values	<b>health-care</b> 4% <b>taxes</b> 3% <b>Other (9551)</b> 93%
1	Says the Annies List political group supports third-trimester abortions on demand.	abortion
2	When did the decline of coal start? It started when natural gas took off that started to begin in (P...	energy,history,job-accomplishments

id	title	author	text	# label
	<b>[null]</b> 3% <b>The Dark Agenda B...</b> 0% <b>Other (20237)</b> 97%	<b>nan</b> 9% <b>Pam Key</b> 1% <b>Other (18600)</b> 89%	<b>20387</b> unique values	
0	House Dem Aide: We Didn't Even See Comey's Letter Until Jason Chaffetz Tweeted It	Darrell Lucas	House Dem Aide: We Didn't Even See Comey's Letter Until Jason Chaffetz Tweeted It By Darrell Lucas o...	1
1	FLYNN: Hillary Clinton, Big Woman on Campus - Breitbart	Daniel J. Flynn	Ever get the feeling your life circles the roundabout rather than heads in a straight line toward th...	0
2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, 2016 The tension between intelligence analysts and po...	1
3	15 Civilians Killed In Single US Airstrike Have Been Identified	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstrike Have Been Identified The rate at which civilians a...	1



## 4. CHALLENGES

COMPUTATIONAL COST

ARGMAX vs PROBABILITY

ZERO PROBABILITY,  
UNSEEN WORDS

UNDERFLOW AND OVERFLOW



## 5. METHODS

BINARY SEARCH  
PARALLELIZATION

THRESHOLD USE-CASE

LAPLACE SMOOTHING  
EPSILON  
DISCARD

LOGS-CONVERSION



## 5. METHODS – CONFUSION MATRIX

ACTUAL LABEL	1	0
	1	0
1	<b>TRUE POSITIVE</b>	<b>FALSE NEGATIVE</b>
0	<b>FALSE POSITIVE</b>	<b>TRUE NEGATIVE</b>

- o **True Positive:** Predicting crash when actual crash is present.
- o **False Negative:** Predicting no crash when actual crash is present.
- o **False Positive:** Predicting crash when no actual crash is present.
- o **True Negative:** Predicting no crash when no crash is present.

## II. CHALLENGES AND SOLUTIONS

1. Computational Time
2. Zero-Probability
3. Float Under/Over-flow
4. Unseen Words
5. Argmax vs Probability
6. GridSearches

# 1. COMPUTATIONAL COST

Good portion of computational time is spent on searching the terms in the bag of words. This directly implies the necessity of the **binary search**.

- Avg times on 2C Dataset (600 lookups):  $1.13 \pm 0.14s \rightarrow 0.017 \pm 0.004s$   
**(100x Faster)**

Another good portion of computational time is spent on applying the same function (either **bag\_of\_word\_generator** or **text\_evaluation**) on long vectors/lists. This also directly implies the necessity of **parallelization**, via the library ***future.apply***.

- Avg times on 2C Dataset (Read all dataset):  $\sim 161 \pm 1.2s \rightarrow \sim 43 \pm 1.1s$   
**(4x Faster)**

## 2. ZERO-PROBABILITY

- o There are cases where a word is identified in one class, but not in the other, leading to a **zero-probability**
- o in the product for the likelihood.
- o No amount of biased words can recover from a zero factor, which means they need to be **eliminated**:
  - **Laplace Smoothing**: add 1 to (possibly) every occurrence, in every class, so that there is a baseline frequency for every word (usually an integer).
  - **Epsilon-Substitution**: when faced with a zero-probability, use epsilon instead (arbitrary parameter).

### 3. FLOAT UNDER/OVER-FLOW

- o When handling and multiplying very small/big numbers we run the risk of underflow/overflow.
- o This essentially leads to an erroneous approximation of the numbers, thus deleting the relationships between them (everything becomes either **zero** or **Infinity**).
- o Solution: use **logarithms** instead, which translates the **product** to a **sum**. In case we want to retrieve the final, normalized probabilities, it is sufficient to center and then apply the exponential:

```
final_probabilities <- final_probabilities + exponent*log(p)
final_probabilities <- final_probabilities - mean(final_probabilities)
final_probabilities <- exp(final_probabilities)
final_probabilities <- final_probabilities / sum(final_probabilities)
```

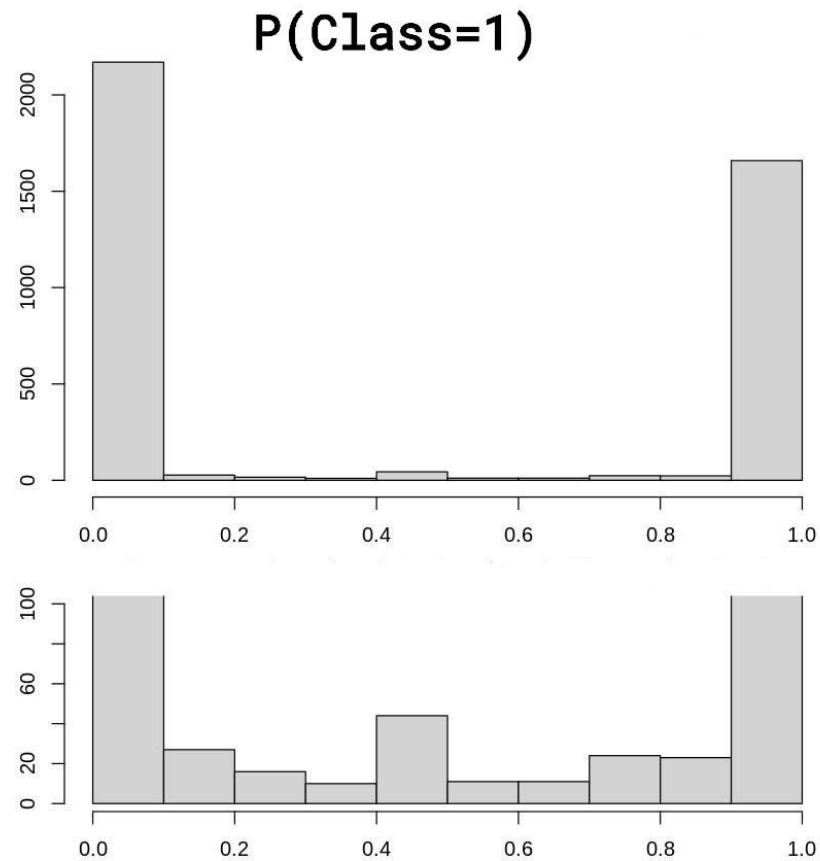
## 4. UNSEEN WORDS

- o In text evaluation, there can be words that do not appear in the training vocabulary.
- o Default strategy is to ignore them, but there are cases where they occupy a big portion of the sentences.
- o This leads to unreliable results, hence we introduce a **threshold**:
  - In case more than **X%** of the text is composed by unknown words, it is **discarded** due to insufficient training data.
  - X depends on how lenient we are when considering doubt classifications. In order to remain neutral, we choose **50%**.
  - This may change **depending** on **practical use-cases**.

## 5. ARGMAX VS PROBABILITY

- o Final result of the calculations is the logarithm of probability for each class (unnormalized).
- o Literature approach stops here, gets the **argmax** (most probable class) and assigns the corresponding label.
- o However, this runs the risk of being biased against certain topics/classifications, where one class might prevail ever so slightly:
  - A consistent value of  $P_0 \sim 45\%$ ,  $P_1 \sim 55\%$  would reliably get class = 1).
- o In order to avoid this, we could go back to probabilities for each class, make a set in  $[0, 1]$  and compare an uniform draw  $\sim U(0, 1)$  against it, **then** assign label → **No biases** against particular topics, **same results** in the broadest datasets.
- o Particularly in the case of two classes, the usage of argmax is equivalent to the application of a threshold valued at 0.5.
  - This approach might be chosen when a **favored bias** for **doubt cases** is present.

## 5. ARGMAX VS PROBABILITY



- Choosing a different threshold basically corresponds to change how to redistribute the central (doubt) cases.
- In case we are determining the chance of raining, one individual could:

Be indifferent towards bringing an umbrella on a sunny day, or getting wet on a rainy day (threshold 0.5)

Heavily prefer not getting wet, at the cost of bringing an umbrella on a sunny day (threshold ~ 0.2, 0.1...)

Hate bringing unnecessary umbrellas, not minding getting wet every once in a while (threshold ~ 0.8, 0.9...)



## 6. GRIDSEARCH (2 CLASSES)

sw_rm	only_na	add_to_counts	unseen_threshold	epsilon	accuracy
<lgl>	<lgl>	<lgl>	<dbl>	<dbl>	<dbl>
TRUE	FALSE	TRUE	0.3	1e-06	0.9009990
FALSE	TRUE	TRUE	0.3	1e-05	0.9004950
FALSE	FALSE	TRUE	0.3	1e-05	0.9004950
FALSE	TRUE	TRUE	0.3	1e-06	0.9000000
FALSE	FALSE	TRUE	0.3	1e-06	0.9000000
FALSE	TRUE	TRUE	0.3	1e-03	0.8960396
FALSE	TRUE	TRUE	0.3	1e-04	0.8960396
FALSE	FALSE	TRUE	0.3	1e-03	0.8960396
FALSE	FALSE	TRUE	0.3	1e-04	0.8960396
FALSE	FALSE	TRUE	0.3	1e-01	0.8935644
FALSE	TRUE	TRUE	0.3	1e-01	0.8930693
FALSE	TRUE	TRUE	0.3	1e-02	0.8930693
FALSE	FALSE	TRUE	0.3	1e-02	0.8930693
TRUE	TRUE	TRUE	0.3	1e-06	0.8925743
TRUE	TRUE	TRUE	0.3	1e-05	0.8920792

sw_rm	accuracy
<lgl>	<dbl>
FALSE	0.8308168
TRUE	0.8251061

epsilon	accuracy
<dbl>	<dbl>
1e-06	0.8884282
1e-05	0.8791460
1e-04	0.8616337
1e-03	0.8407178
1e-02	0.8160272
1e-01	0.7812500
1e+00	0.7285272

only_na	accuracy
<lgl>	<dbl>
FALSE	0.8410007
TRUE	0.8149222

add_to_counts	accuracy
<lgl>	<dbl>
FALSE	0.7643211
TRUE	0.8916018

## 6. GRIDSEARCH (6 CLASSES)

Accuracy	SW_RM	Only_NA	Add_Counts	Threshold	Epsilon
0.311594	True	False	False	0.3	0.0010
0.304348	True	True	False	0.3	0.0001
0.297101	True	False	False	0.3	0.0001
0.289744	True	False	False	0.7	0.0001
0.289744	True	False	False	0.7	0.0010
...	...	...	...	...	...
0.152174	True	False	False	0.3	0.1000
0.151624	False	True	False	0.3	1.0000
0.150259	False	True	False	0.7	1.0000
0.146769	False	True	False	0.5	1.0000
0.130435	True	False	False	0.3	1.0000

Accuracy	
SW_RM	
False	0.233995
True	0.245919

Accuracy	
Only_NA	
False	0.244374
True	0.235541

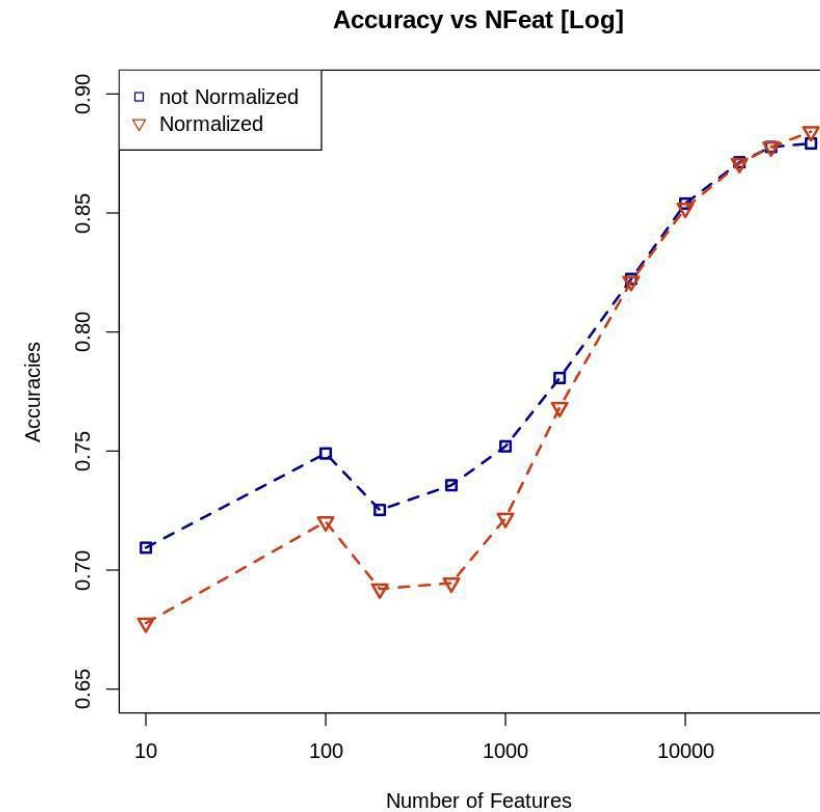
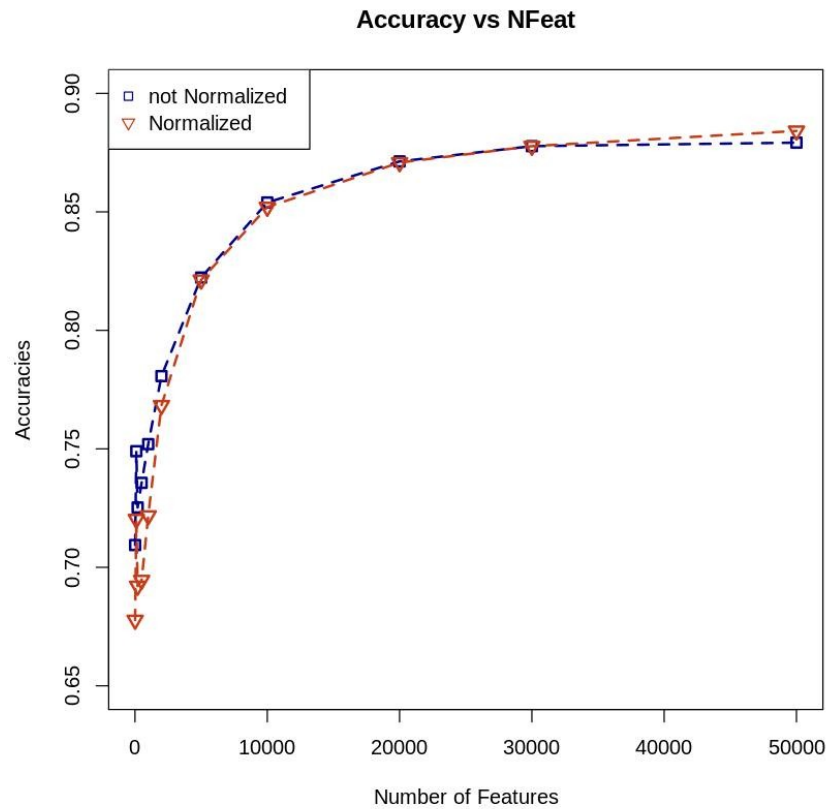
Accuracy	
Epsilon	
0.000001	0.249777
0.000010	0.251712
0.000100	0.261365
0.001000	0.249043
0.010000	0.233178
0.100000	0.215903
1.000000	0.218724

Accuracy	
Add_Counts	
False	0.230077
True	0.249838

# III. RESULTS AND ANALYSIS

1. Two-Classes Results
2. Six-Classes Results

# 1. TWO CLASSES RESULTS



# 1. TWO CLASSES RESULTS

## Confusion Matrix and Statistics

	A	B
A	1007	220
B	30	763

Accuracy : 0.8762

95% CI : (0.8611, 0.8903)

No Information Rate : 0.5134

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.751

McNemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9711

Specificity : 0.7762

Pos Pred Value : 0.8207

Neg Pred Value : 0.9622

Prevalence : 0.5134

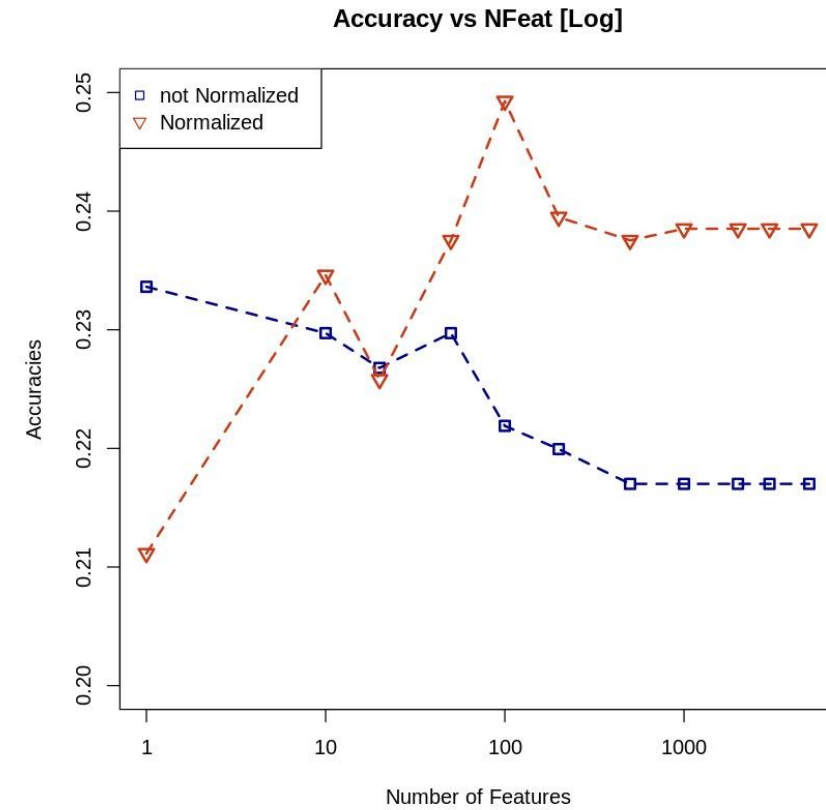
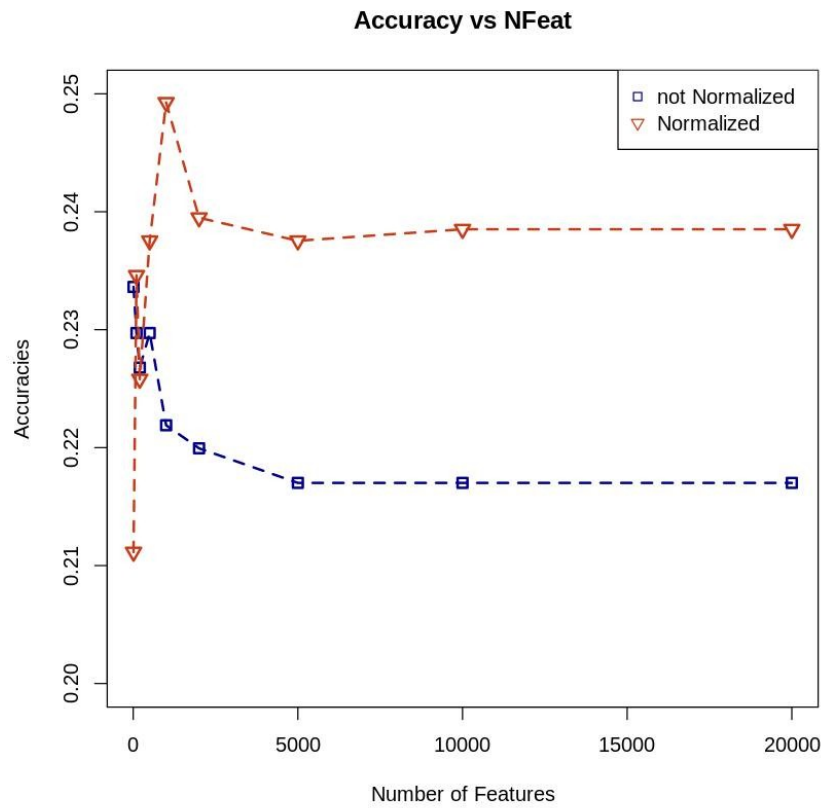
Detection Rate : 0.4985

Detection Prevalence : 0.6074

Balanced Accuracy : 0.8736

'Positive' Class : A

# 1. SIX CLASSES RESULTS



## 2. SIX CLASSES RESULTS

words	C0
<chr>	<dbl>
voted	0.0013215717
says	0.0012863439
obamacare	0.0012609412
clinton	0.0012238238
hillary	0.0010668858
supports	0.0009059029
social	0.0008789088
plan	0.0007546164
real	0.0007529709
class	0.0007326026
security	0.0006810177
deciding	0.0006291032
taxpayer	0.0006234269
since	0.0006137369
card	0.0006035487

words	C1
<chr>	<dbl>
percent	0.0019443938
georgia	0.0011409859
since	0.0010350117
barack	0.0009914697
whether	0.0009687094
less	0.0009641817
spending	0.0009424305
increased	0.0008509373
wisconsin	0.0007791807
government	0.0007706770
scheme	0.0007397337
american	0.0007360136
countries	0.0007114646
debunked	0.0006890396
mcauliffe	0.0006668009

words	C2
<chr>	<dbl>
jobs	0.0014531127
put	0.0012667867
increased	0.0010282631
cut	0.0009928658
billion	0.0008445747
indiana	0.0008287517
millions	0.0007645408
trillion	0.0006681246
among	0.0006604152
work	0.0006409711
lost	0.0006022859
minnesota	0.0005974432
creation	0.0005882243
sector	0.0005697386
counties	0.0005478788

words	C3
<chr>	<dbl>
percent	0.002471916
your	0.0013131981
average	0.0012123505
today	0.0012105414
obama	0.0011977137
rep	0.0011735951
highest	0.0009638605
less	0.0008396402
poverty	0.0008113255
walker	0.0008081167
half	0.0007898004
barack	0.0007756157
says	0.0007483314
obamas	0.0007344014
jeb	0.0007236414

words	C4
<chr>	<dbl>
obama	0.0027631550
socialists	0.0024758953
reps	0.0020750935
percent	0.0019936571
barack	0.0018002238
muslim	0.0017579531
obamas	0.0014271282
face	0.0013570559
border	0.0012711868
average	0.0011890626
rep	0.0010471532
bachmann	0.0009317999
sic	0.0009317999
wants	0.0009305084
serving	0.0008845540

words	C5
<chr>	<dbl>
obamacare	0.0017729559
mccain	0.0013867339
obama	0.0013218427
says	0.0011908321
hillary	0.0011579700
barack	0.0011442286
clinton	0.0011050848
under	0.0010312403
making	0.0009708272
african	0.0009544214
since	0.0009331015
billion	0.0009038290
georgia	0.0009009453
medicare	0.0008591648
seniors	0.0008244381

## 2. SIX CLASSES RESULTS

### Confusion Matrix and Statistics

	A	B	C	D	E	F
A	33	30	21	26	11	12
B	40	53	47	30	25	27
C	45	45	71	59	13	33
D	32	38	46	65	10	38
E	3	7	6	1	5	7
F	17	21	34	28	7	37

### Overall Statistics

Accuracy : 0.2581

95% CI : (0.2315, 0.286)

No Information Rate : 0.2199

P-Value [Acc > NIR] : 0.0021324

Kappa : 0.0853

McNemar's Test P-Value : 0.0003032

	Class: A	Class: B	Class: C	Class: D	Class: E	Class: F
Sensitivity	0.19412	0.27320	0.3156	0.31100	0.070423	0.24026
Specificity	0.88277	0.79614	0.7556	0.79853	0.974790	0.87687
Pos Pred Value	0.24812	0.23874	0.2669	0.28384	0.172414	0.25694
Neg Pred Value	0.84607	0.82397	0.7966	0.81864	0.933602	0.86689
Prevalence	0.16618	0.18964	0.2199	0.20430	0.069404	0.15054
Detection Rate	0.03226	0.05181	0.0694	0.06354	0.004888	0.03617
Detection Prevalence	0.13001	0.21701	0.2600	0.22385	0.028348	0.14076
Balanced Accuracy	0.53844	0.53467	0.5356	0.55477	0.522606	0.55856



## IV. CONCLUSION

# CONCLUSION

What we understood from the analysis:

- **Lagrangian Smoothing/Epsilon Substitution** are valid solutions for Zero-Probability
- On Large dataset, with each element as a large text (TwoClasses) the **Normalization** and **Feature Selection** did not improve the accuracy substantially, implying that we had enough data to have a valid decision-making without adjustments. However, there is a small peak at around 100 Feat-Selected.
- On the Smaller Dataset, with each element as a small text (SixClasses) both the **Normalization** and **Feature Selection** shine as they improve the accuracy (albeit, not substantially due to the lack of data)

Possible future work:

- Syllable analysis: **Bouba-Kiki Effect**
- Other Language Implementations: Chinese problems, etc...

A series of white, thin, overlapping geometric lines on a black background, forming various polygons and intersecting points, primarily located on the left side of the slide.

THANK YOU



# REFERENCES

- o [Text Classification and Naive Bayes Classifier](#)
- o [Token Normalization](#)
- o [Bouba-Kiki Effect](#)