MILESTONE # 5: Final Report
IS 567 - Fall 2022

Muhammad Ibrahim Mian
UIN: 673846900

Email Spam Classification using Machine Leaning

Introduction

Of all the different communication methods, email is one of the most extremely important form of communication. It has replaced the traditional snail mail and can be accessed from any part of the world just with the help of internet connectivity. With the ever growing number of users, emails are set to increase to a record high, but the worst part of that is, out of that approximately 57 % emails are of no use as they are spam emails.

In my project, I will be using Enron Email database to identify and classify spam emails. Enron corporation was an energy company which came into the spotlight in 2001 due to accounting frauds and emails of many executives and was made publicly available.

Descriptive Statistics

1. We will first look at the dataset itself.

   a. Total emails (Testing & Training Set): 33,716
      - Spam Emails: 17,171
      - Ham Emails: 16,545

   b. Total number of columns: 5

2. We remove the null values in column "Message"

   a. Total emails (Testing & Training Set): 33,345
      - Spam Emails: 16,852
      - Ham Emails: 16,493

   b. Total number of columns: 5

3. Train/Test Split

   a. Total emails (Testing & Training Set): 33,345

   b. Train Set
      - Spam Emails: 11,813
      - Ham Emails: 11,528

   c. Test Set
      - Spam Emails: 5,039
      - Ham Emails: 4,965

4. Data extracted and in the data frame

Data present in data frame - (See Figure 1)

5. Email examples

Here is an example of the message. Initially we have it in raw form with all the numbers and characters. After cleaning it by removing non essential spaces, punctuation marks, special characters and opening contractions, we see the following result

   a. Raw email - (See Figure 2)
   b. Cleaned email - (See Figure 3)

6. Tokenized Words most common words (Plots)

Here, we can see the plots for the most common occurring words. We can see many stop words are present so we need to remove those first - (See Figure 4)

7. Stopwords removed most common words (Plots)

Here we get a better sense of what sort of words are common in the emails. This tells us some important information of the content of the emails - (See Figure 5)

8. Wordcloud (Messages column)

Similar to the plots, we can see in the word cloud the most occurring words. This is before any sort of filters on the data - (See Figure 6)

9. Wordcloud (ProcessedMessage column)

Similar to the plots, we can see in the word cloud the most occurring words. This is after filters on the data - (See Figure 7)

10. NLP Processing

When we performed some analysis on this data, we noticed that there were a lot of mentions regarding names and organizations. These connections help us analyze what sort of data was present in the emails.

- Email - (See Figure 8)
- POS Tagging - (See Figure 9)
- Dependency parsing - (See Figure 10)
- Name Entity Recognition - (See Figure 11)

Pre-processing / Transformation Steps

In this part, I will be discussing how I used my data in raw form and performed pre processing on it

1. We start with data in txt files labeled Ham and Spam. We need to join the files together and create a data frame from it
   - We start with downloading the dataset
   - We separate the emails from the subject
   - We check and and add date to the row for the email.

2. Next, we will load the data in our data frame and perform checks
   - Label wise counts
   - All email counts
   - Null values

3. After this, we will perform steps on the text data
   - Remove special characters
   - Reduce spaces with punctuation to improve data quality (contractions)
   - All lowercase
   - Remove links
   - Open contractions
   - Tokenization
   - Stopwords Removal
   - Lemmatization

The reason to conduct these steps was to clean the data of any elements that would affect our models performance. In order to create better sense of the data present, we used the following steps above.

The remaining features after cleaning the data are important as they build to the context of the email. If there are words such as money, bank account, private emails which raise a red flag and set as SPAM email. The model will learn from the features and see if the same sort of words occur, they will mark it as SPAM or pass it as HAM

Feature Extraction / Selection

For this task, we will be using 5 methods for feature extraction

1. Count Vectorizer
2. TF-IDF
3. K best feature using Chi2
4. Variance threshold
5. Word2Vec

Using these, we will now apply models on different combinations

Preliminary models, parameters, evaluation results, and error analysis

For all results combined - (See Table 1)

1. Naive Bayes

I started with the simplest model first and applied both count vectorization and TF-IDF features on it

- Count Vectorizer: We started with top 1000 features of the data and using that to create the vectors for the model. Here are the results of the model - (See Figure 12)
- TF-IDF: We started with top 1000 features of the data and using that to create the vectors for the model. Here are the results of the model - (See Figure 13)

Comparing both features, we see that as we increase the number of features, we see that the false negatives decrease and false positives increase meaning HAM emails are also being classified as SPAM which makes emails unnecessarily been marked wrong

2. Support vector machines

Now we look at support vector machines and their performance

- Count Vectorizer: We started with top 1000 features of the data and using that to create the vectors for the model. Here are the results of the model - (See Figure 14)
- TF-IDF: We started with top 1000 features of the data and using that to create the vectors for the model. Here are the results of the model - (See Figure 15)

Comparing Naive bayes with Support vector machines, we see that it gives us a better result in terms of accuracy and with TF-IDF, it gives a better result compared to count vectorizer

3. K Nearest Neighbor (KNN)

I also took a look at unsupervised learning compared to supervised learning to see how it performs. Using elbow method, we found that K=4 is the best parameter - (See Figure 16)

- Count Vectorizer: We started with top 1000 features of the data and using that to create the vectors for the model. Here are the results of the model - (See Figure 17)
- TF-IDF: We started with top 1000 features of the data and using that to create the vectors for the model. Here are the results of the model - (See Figure 18)

Comparing Naive bayes and Support vector machines, we see that it performs very poorly and thus is not a good idea to implement unsupervised learning in our dataset

4. **K Best features using Chi2**

Now, we have established that unigram and TF-IDF gives us the best features, we will now use this combination with Chi2 to select our features and see how the number of features effects the models performance - (See Figure 19 & 20)

Looking at both the models, we see that SVM is giving us the best performance compared to Naive Bayes. It is also interesting to note that as the number of features increase, the better the results so it will be good to see how more features can improve the results as well as looking at the computational cost of running the model.

5. **Variance Threshold**

Taking the previous assumptions here as well, now the features are filtered with variance threshold and see how the features effect the model.

For this, TF-IDF was giving very poor results so we used count vectorizer with variance threshold - (See Figure 21 & 22)

Here, we see that both models are performing pretty much similar with variance threshold even with a huge number of features. K best features using chi2 still performs better than variance threshold

6. **Decision Tree**

Here, I decided to use decision tree by trying to limit the features even more using depth of the decision trees. - (See Figure 23)

Notice that the accuracy has dropped severely compared to our other models so it is not recommended

7. **SVM - Word2Vec**

For our final task, I used the Word2Vec word embedding to create features.

In Word2Vec, we will take a reference word embedding which already have the vectors for words. The reference word embeddings I am using is of Google which is publicly available.Google has used its search engine to create references for words and their connections.

The data will match the library and copy the vector representation for the available words. For the remaining words not found, we have couple of options. First is to ignore those words from the features which we are using here. Second is we can create our own vector for it and include in the dictionary - (See Figure 24)

Conclusion and Insights

After analyzing the results, following observations can be made

- Models perform best when only unigram data is present and accuracy starts to decrease as more pairs are made
- TF-IDF performed better compared to Count Vectorizer
- SVM performed better compared to Naïve Bayes
- We are currently working with 1000 sample size and looking at K best model, we can see that adding more features will help us achieve better results even with simple models
- Using only one trained word embedding vector on the Word2Vec, it shows promising trend and it would be interesting to see in the future that how other word embeddings can improve the model results

Some future improvements we can make to this project

- Data can be trained on neural networks to see how the model performs
- Different word embedding vector model for Word2Vec
- Testing model on different dataset

References

1. Singh, V. K., & Bhardwaj, S. (2018). Spam mail detection using classification techniques and global training set. *Intelligent Computing and Information and Communication*, 623–632. https://doi.org/10.1007/978-981-10-7245-1_61

2. Peng, W., Huang, L., Jia, J., & Ingram, E. (2018). Enhancing the naive Bayes spam filter through intelligent text modification detection. 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). https://doi.org/10.1109/trustcom/bigdatase.2018.00122

3. Singh, M., Pamula, R., & shekhar, S. kumar. (2018). Email spam classification by support vector machine. 2018 International Conference on Computing, Power and Communication Technologies (GUCON). https://doi.org/10.1109/gucon.2018.8674973

4. Muhammad Abdulhamid, S., Shuaib, M., Osho, O., Ismaila, I., & K. Alhassan, J. (2018). Comparative analysis of classification algorithms for email spam detection. International Journal of Computer Network and Information Security, 10(1), 60–67. https://doi.org/10.5815/ijcnis.2018.01.07

5. Kulwinder Kaur, & Dr. Mukesh Kumar. (2015). Spam detection using KNN, back propagation and Recurrent Neural Network. International Journal of Engineering Research And, V4(09). https://doi.org/10.17577/ijertv4is090492

Appendix

1. Figure 1

	Message ID	Subject	Message	Spam/Ham	Date
0	0	ena sales on hpl	just to update you on this project ' s status ...	ham	2000-05-10
1	1	98 - 6736 & 98 - 9638 for 1997 (ua 4 issues)	the above referenced meters need to be placed ...	ham	2000-02-18
2	2	hpl nominations for december 28 , 1999	(see attached file : hpll 228 . xls)\n- hpll...	ham	1999-12-27
3	3	revised nom - kcs resources	daren ,\nit ' s in .\nbob\n- - - - - - - - - -...	ham	2000-06-29
4	4	new production - sitara deals needed	daren ,\nfyi .\nbob\n- - - - - - - - - - - - -...	ham	2000-07-28

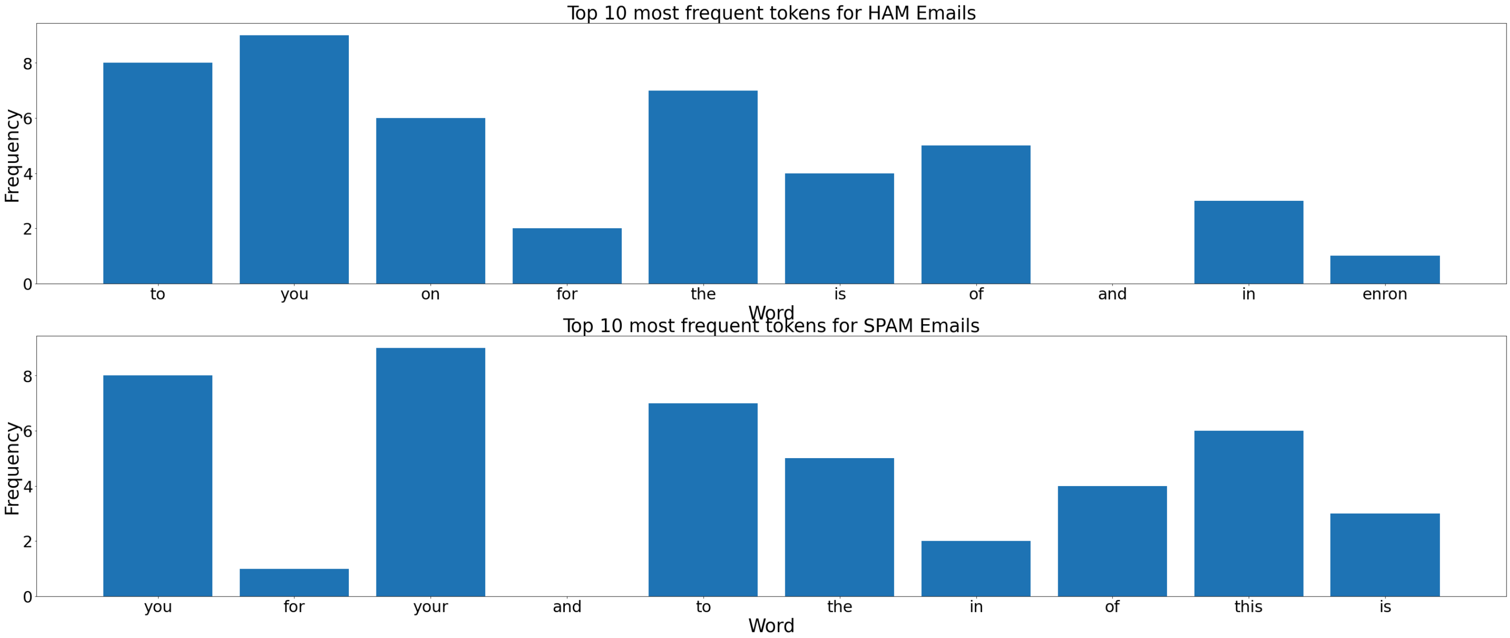
2. Figure 2

```
gentlemen :
please review and let me know if you have any questions .
for scheduling purposes , we will show a receipt from hpl ( transportation
agreement # 4047 ) at agua dulce of 45 , 000 , with deliveries to :
air products - la porte 5 , 000
oxy battleground 10 , 000
rohm & haas dp 20 , 000
dupont dp 10 , 000
dwight : you will need to coordinate these flow changes with the facilities .
thanks ,
>
```

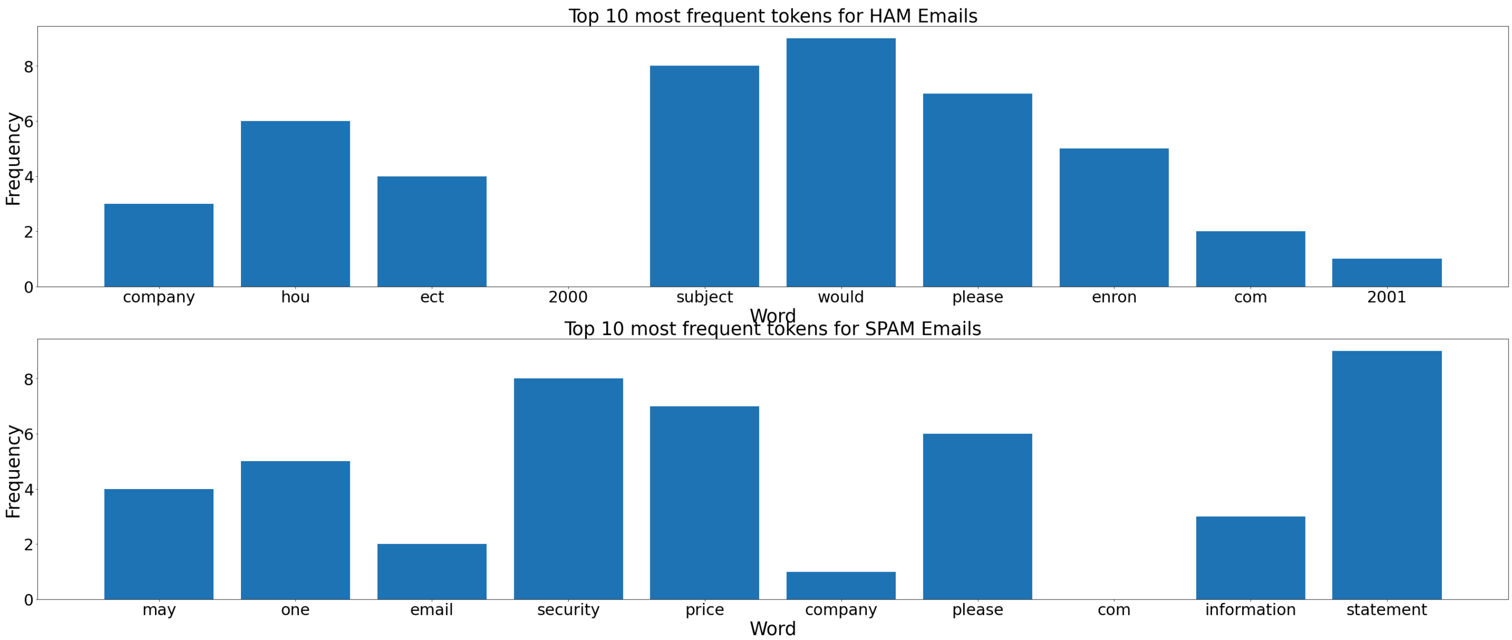
3. Figure 3

```
gentlemen please review and let me know if you have any questions for scheduling purposes we will show a receipt from
hpl transportation agreement 4047 at agua dulce of 45 000 with deliveries to air products la porte 5 000 oxy battlegr
ound 10 000 rohm haas dp 20 000 dupont dp 10 000 dwight you will need to coordinate these flow changes with the facil
ities thanks this email and any files transmitted with it from the elpaso corporation are confidential and intended s
olely for the use of the individual or entity to whom they are addressed if you have received this email in error ple
ase notify the sender smartpigging 812 lomax market takes plan final xls
```

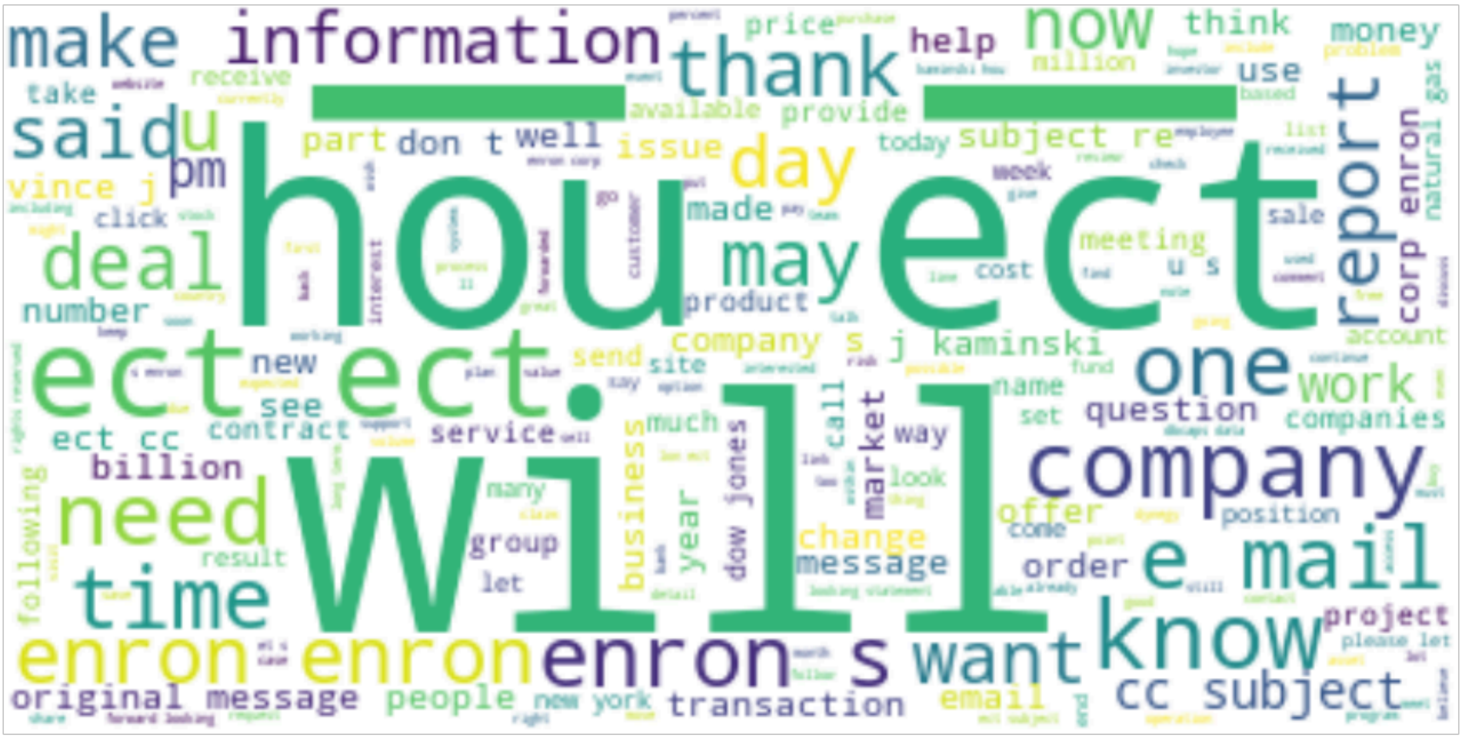
4. Figure 4



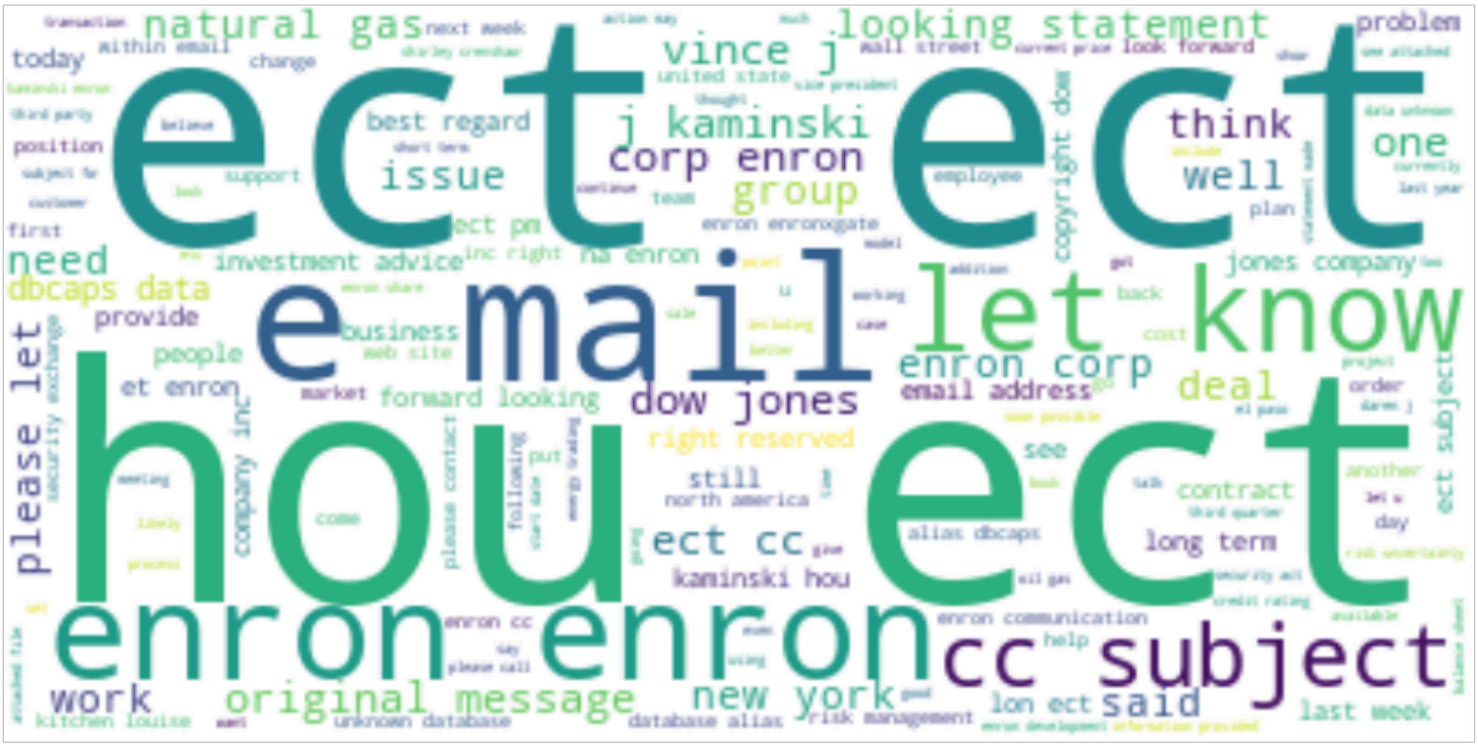
5. Figure 5



6. Figure 6



7. Figure 7



8. Figure 8

```
can you help me out on this darren ? mjj
- - - - - - - - - - - - - - - - - - - - - - forwarded by mary jo johnson / hou / ect on 11 / 09 / 200
10 : 04 am - - - - - - - - - - - - - - - - - - - - - - - - - - -
" john daugherty " on 11 / 08 / 2000 04 : 38 : 37 pm
to :
cc :
subject : re : driscoll ranch # 3 gas pricing and interconnect estimate
mary jo ,
thanks for the update . regarding the notice provision of 6 business days
prior to the close of business on the last business day of the month prior
to selected month , does that mean we need to give you notice for december
by tuesday , november 21 st at 5 : 00 pm or monday , november 20 th at 5 : 00 pm
assuming the 23 rd and 24 th are holidays ?
john daugherty
- - - - - original message - - - - -
from :
to :
cc : ; ;
; ;
; ;
;
sent : wednesday , november 08 , 2000 5 : 12 pm
subject : re : driscoll ranch # 3 gas pricing and interconnect estimate
```

9. Figure 9

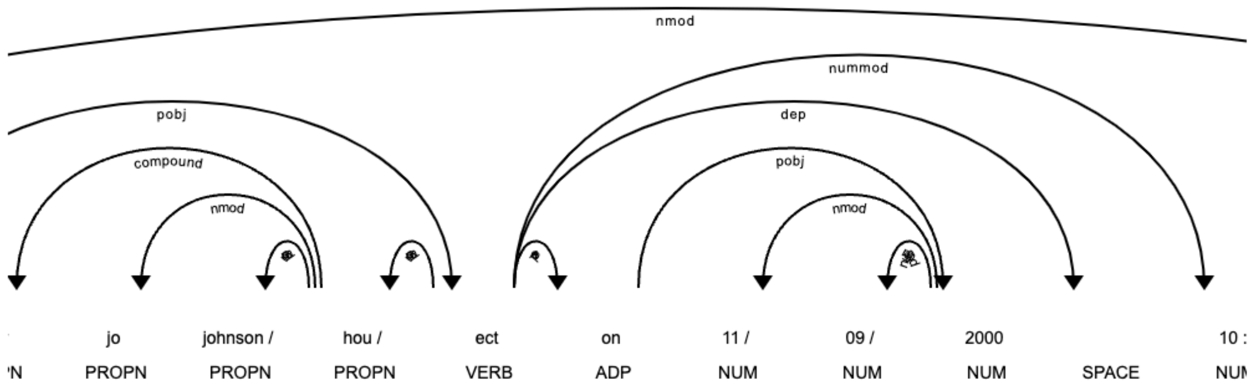
```
In [19]: #POS Tagging

         nlp = spacy.load("en_core_web_sm")
         doc = nlp(message)
         for token in doc:
             print(f'{token.text:{10}} {token.pos_:{10}} {token.tag_:{10}}')
```

```
can        AUX        MD
you        PRON       PRP
help       VERB       VB
me         PRON       PRP
out        ADP        RP
on         ADP        IN
this       DET        DT
darren     NOUN       NN
?          PUNCT      .
mjj        PROPN      NNP

          SPACE      _SP
-          PUNCT      HYPH
-          PUNCT      HYPH
-          PUNCT      HYPH
-          PUNCT      HYPH
-          PUNCT      HYPH
-          PUNCT      HYPH
-          PUNCT      HYPH
```

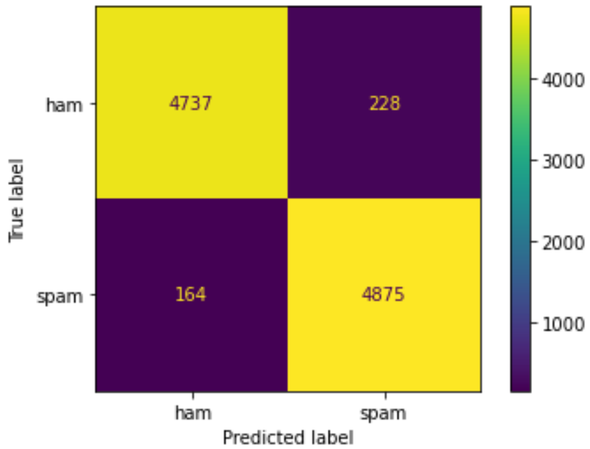
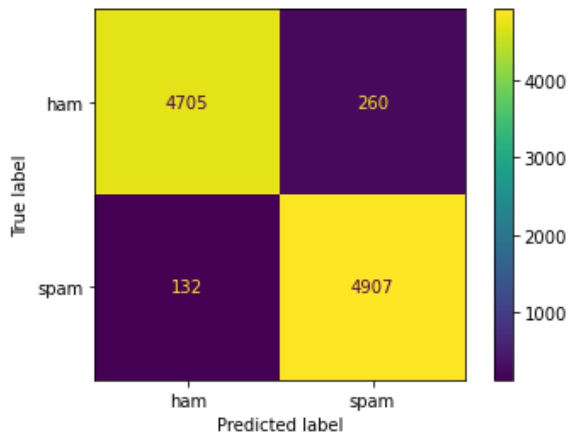
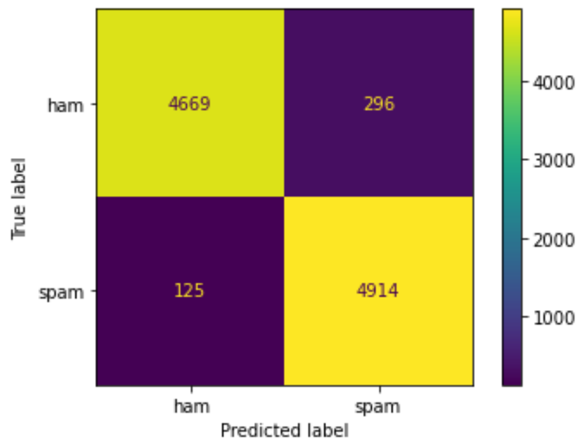
10. Figure10



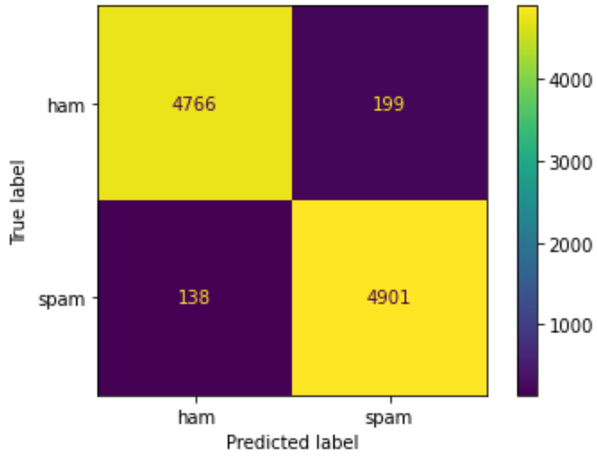
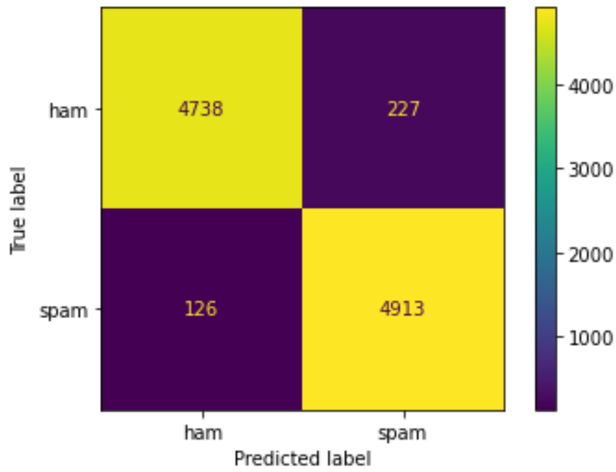
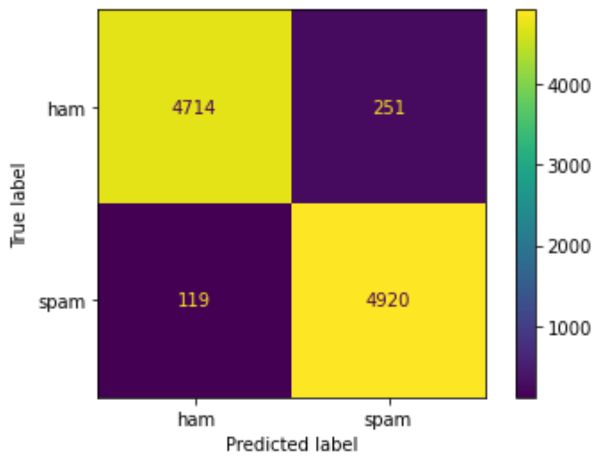
11. Figure 11

```
[('mary jo johnson / hou', 'PERSON'),
 ('11 / 09 / 2000', 'DATE'),
 ('04', 'CARDINAL'),
 ('john daugherty', 'PERSON'),
 ('11 / 08 / 2000 04', 'DATE'),
 ('38', 'CARDINAL'),
 ('37 pm', 'QUANTITY'),
 ('6', 'CARDINAL'),
 ('the last business day of the month', 'DATE'),
 ('month', 'DATE'),
 ('december', 'DATE'),
 ('tuesday', 'DATE'),
 ('5', 'CARDINAL'),
 ('00 pm', 'TIME'),
 ('monday , november 20', 'DATE'),
 ('5', 'CARDINAL'),
 ('00', 'CARDINAL'),
 ('23', 'CARDINAL'),
 ('24', 'CARDINAL'),
 ('john daugherty\n- - - - -', 'PERSON'),
 ('wednesday , november 08 , 2000 5', 'DATE'),
 ('12 pm', 'QUANTITY')]
```

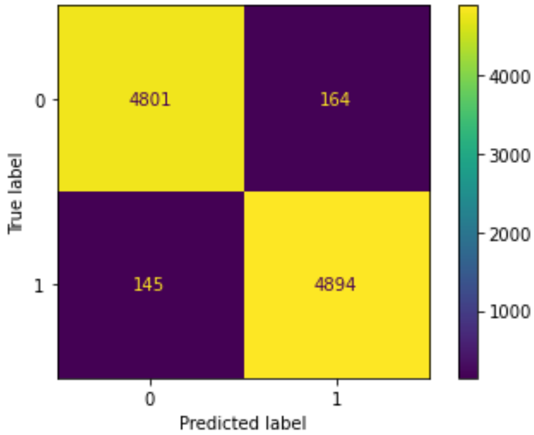
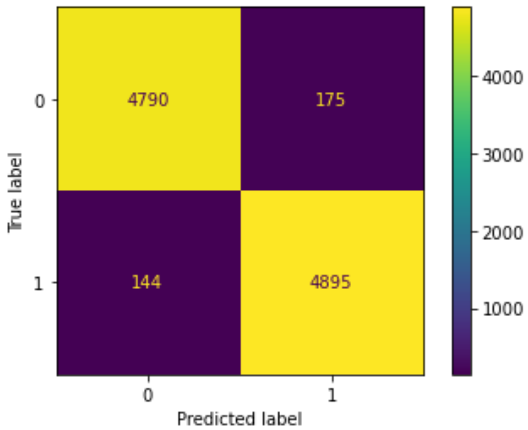
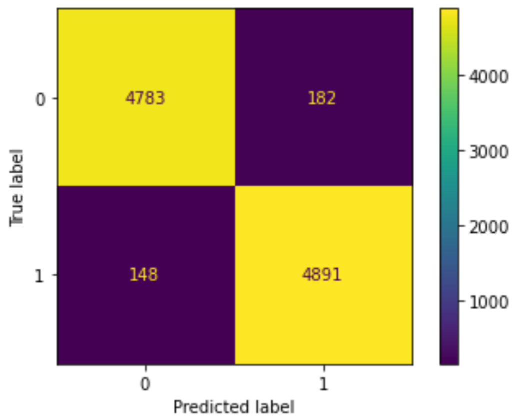
12. <u>Figure 12</u>

<table>
<tr><th>Model</th><th>Sub Type</th><th>Feature space</th><th>Accuracy</th><th>Confusion Matrix</th></tr>
<tr><td rowspan="3">Naive Bayes - Count Vectorizer</td><td>Unigram</td><td rowspan="3">1000</td><td>0.961</td><td>
</td></tr>
<tr><td>Unigram+Bi Gram</td><td>0.961</td><td>
</td></tr>
<tr><td>Unigram+Bi gram+Trigra m</td><td>0.958</td><td>
</td></tr>
</table>

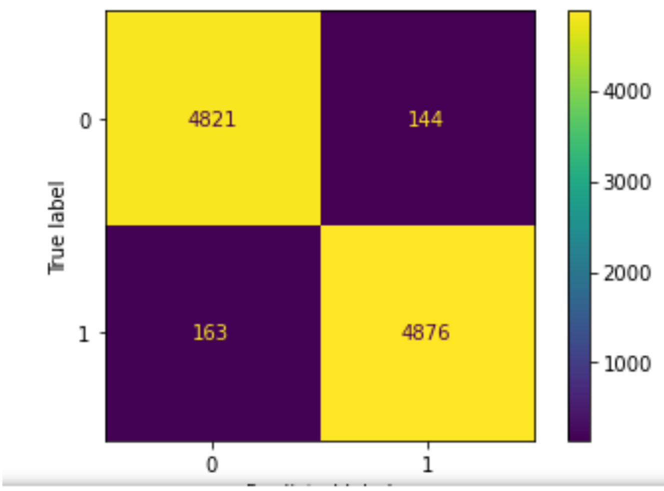
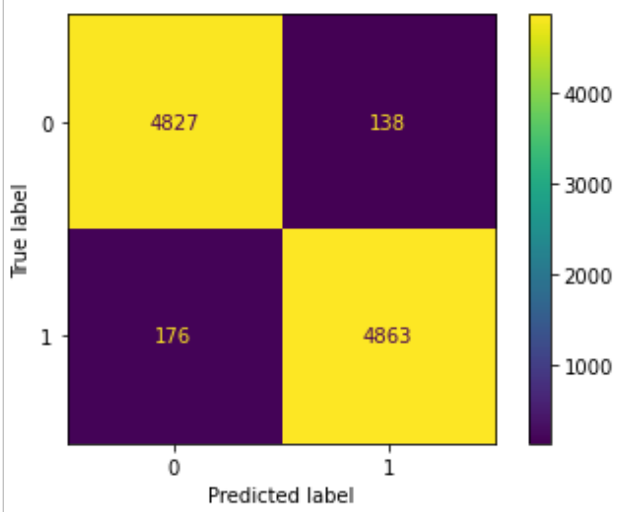
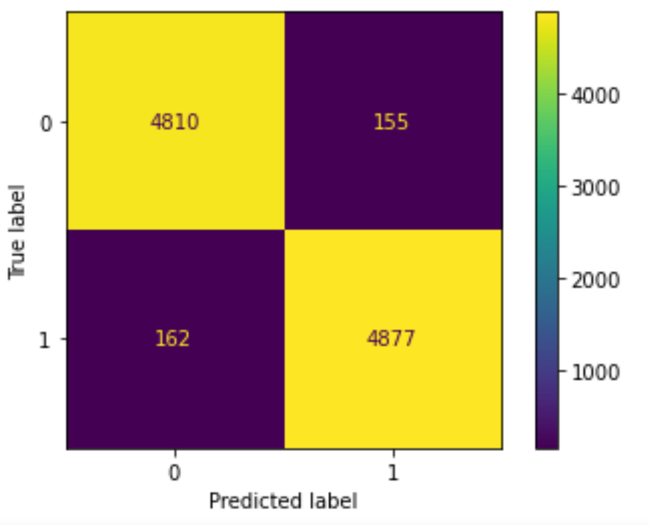
13. Figure 13

Model	Sub Type	Feature space	Accuracy	Confusion Matrix
Naive Bayes - TF-IDF	Unigram	1000	0.966	
	Unigram+Bi Gram		0.965	
	Unigram+Bigram+Trigram		0.963	

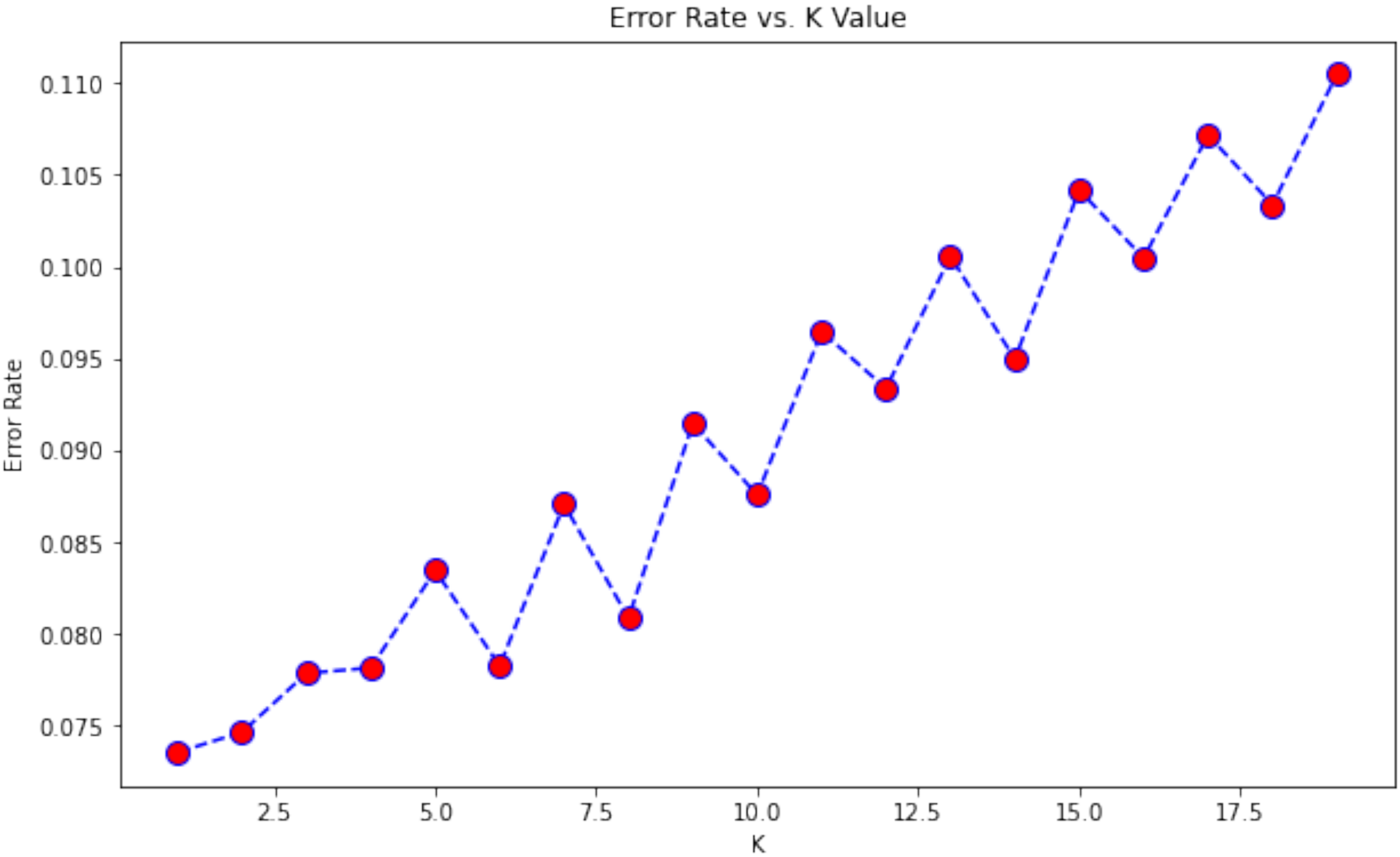
14. Figure 14

Model	Sub Type	Feature space	Accuracy	Confusion Matrix
SVM - Count Vectorizer	Unigram	1000	0.969	
	Unigram+ Bi Gram		0.968	
	Unigram+ Bigram+T rigram		0.967	

15. Figure 15

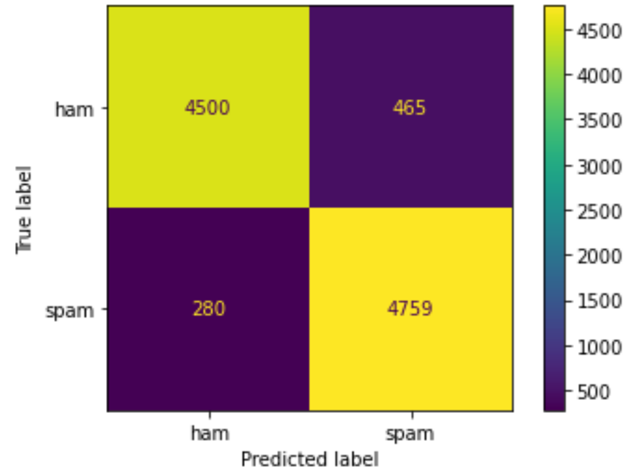
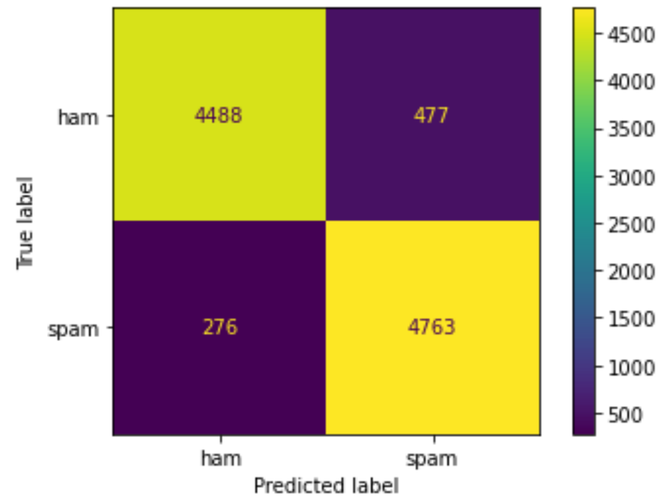
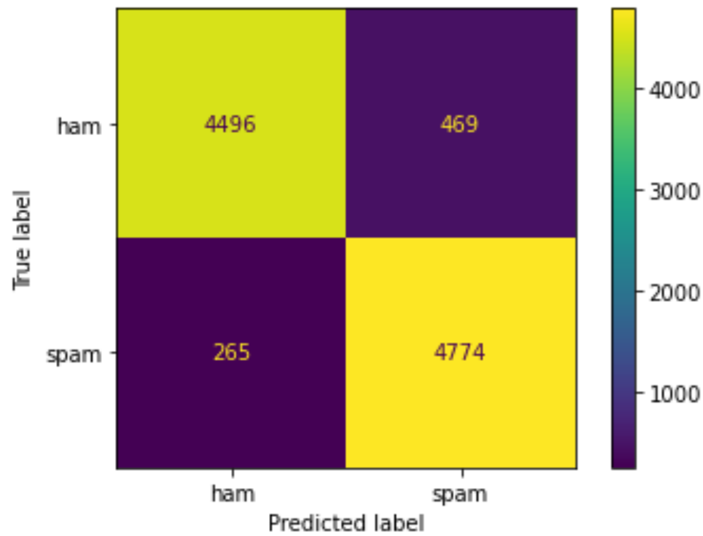
<table>
<tr><th>Model</th><th>Sub Type</th><th>Feature space</th><th>Accuracy</th><th>Confusion Matrix</th></tr>
<tr><td rowspan="3">SVM - TF-IDF</td><td>Unigram</td><td rowspan="3">1000</td><td>0.969</td><td>
</td></tr>
<tr><td>Unigram+ Bi Gram</td><td>0.968</td><td>
</td></tr>
<tr><td>Unigram+ Bigram+T rigram</td><td>0.968</td><td>
</td></tr>
</table>

16. Figure 16

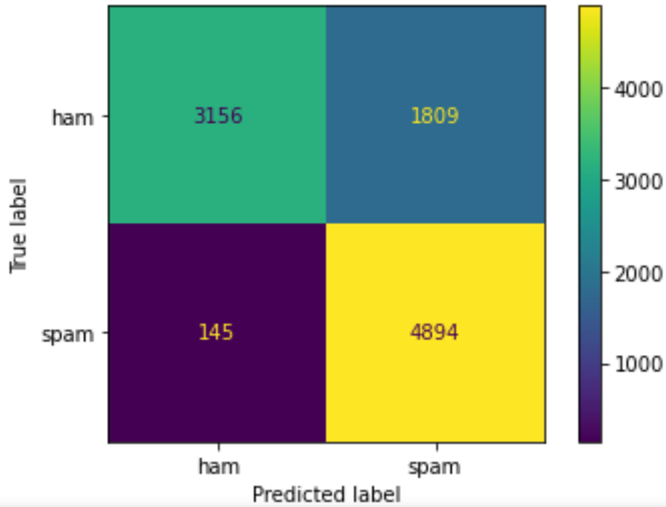
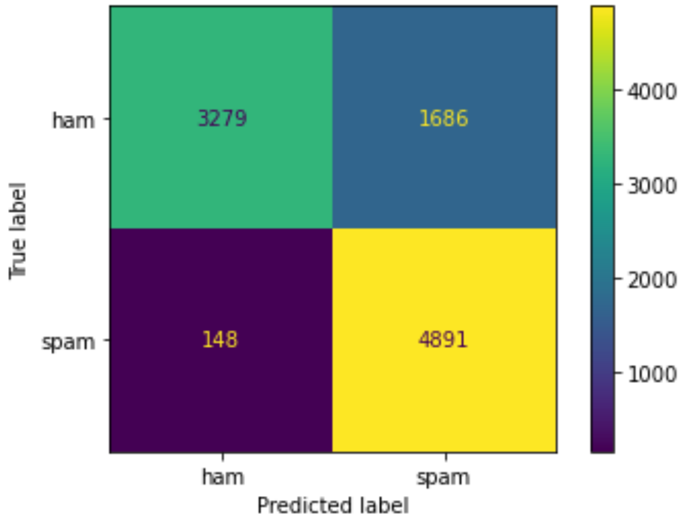
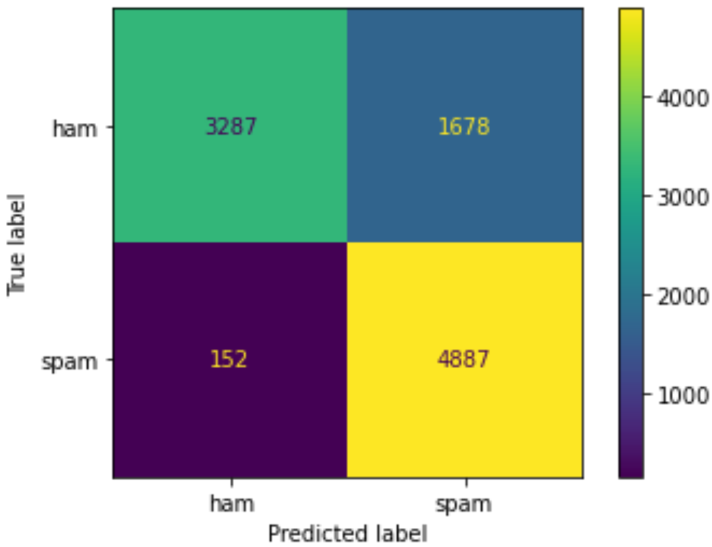


Error Rate vs. K Value
0.110
0.105
0.100
0.095
0.090
0.085
0.080
0.075
Error Rate
2.5
5.0
7.5
10.0
12.5
15.0
17.5
K

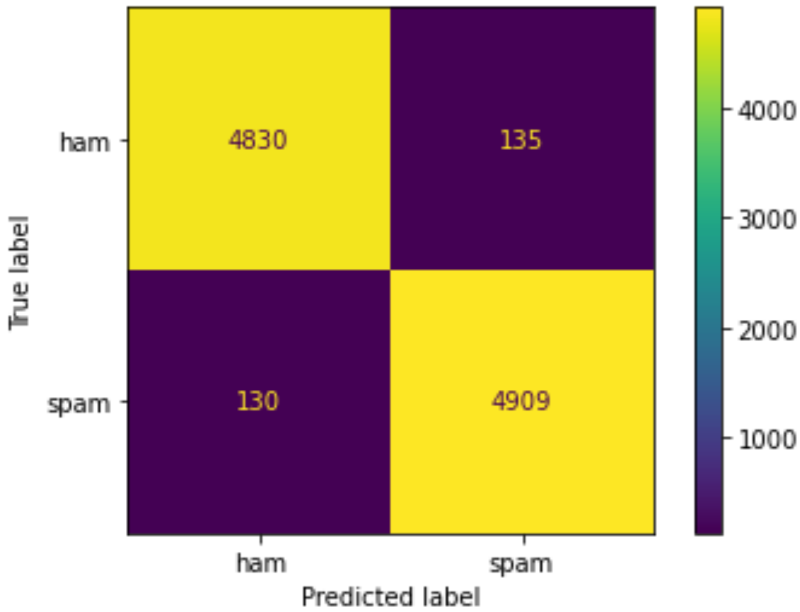
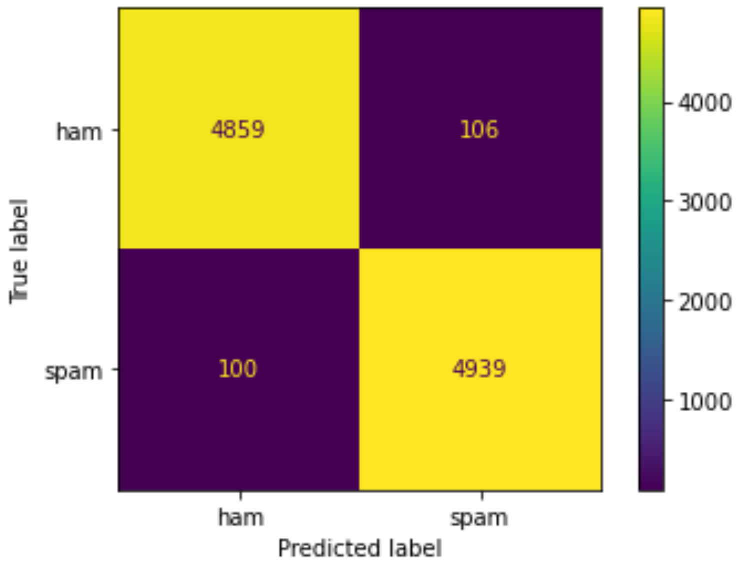
17. Figure 17

Model	Sub Type	Feature space	Accuracy	Confusion Matrix
KNN - Count Vectorizer (K=4)	Unigram	1000	0.926	
	Unigram+ Bi Gram		0.925	
	Unigram+ Bigram+Tri gram		0.927	

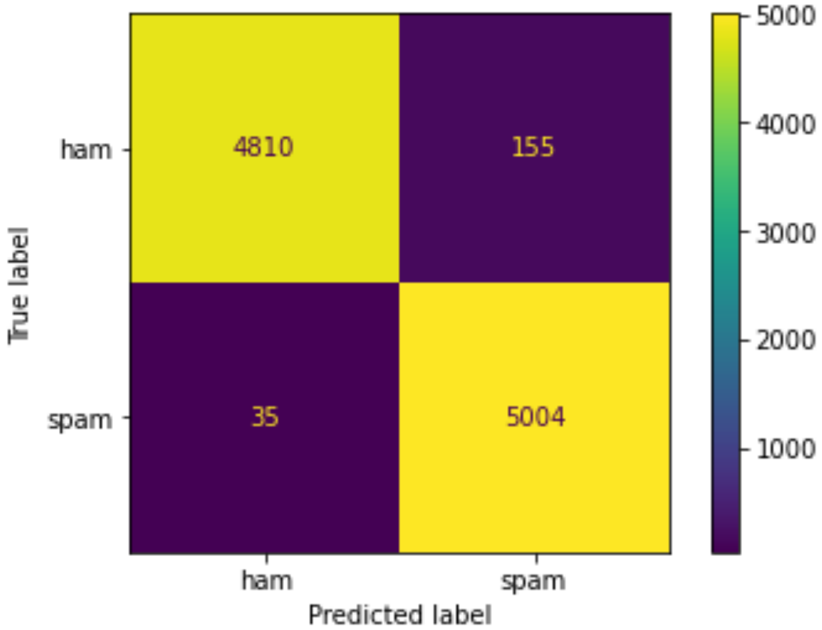
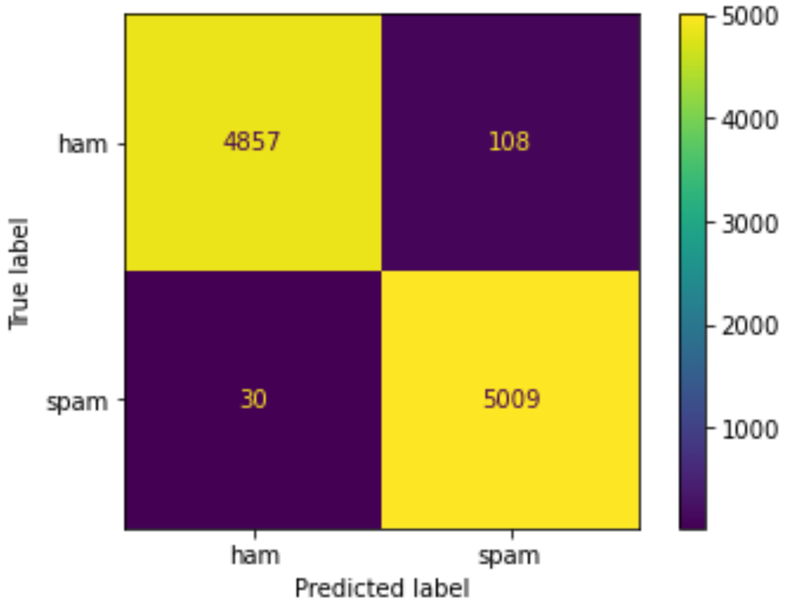
18. <u>Figure 18</u>

<table>
<tr><th>Model</th><th>Sub Type</th><th>Feature space</th><th>Accuracy</th><th>Confusion Matrix</th></tr>
<tr><td rowspan="3">KNN - TF-IDF (K=4)</td><td>Unigram</td><td rowspan="3">1000</td><td>0.805</td><td>
</td></tr>
<tr><td>Unigram+Bi Gram</td><td>0.816</td><td>
</td></tr>
<tr><td>Unigram+Bigram+Trigram</td><td>0.817</td><td>
</td></tr>
</table>

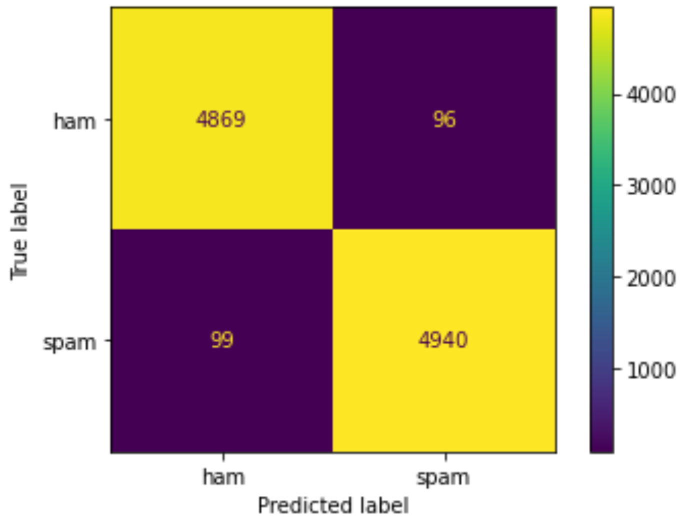
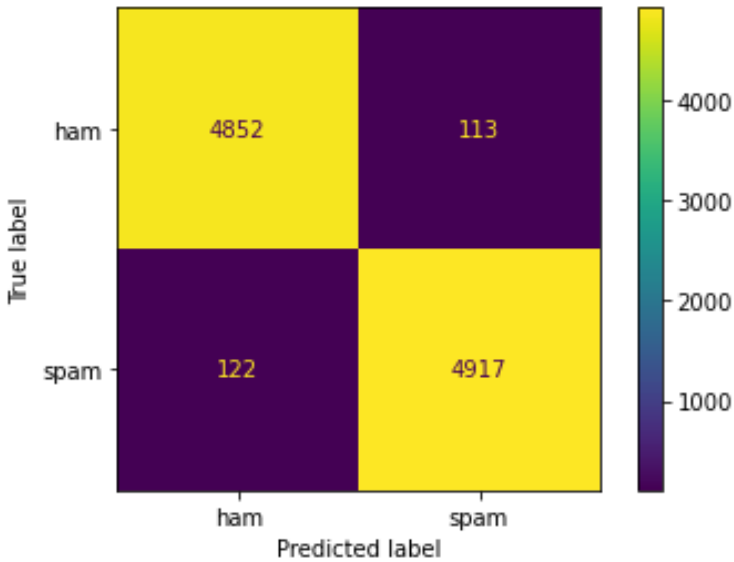
19. Figure 19

Model	Sub Type	Feature space	Accuracy	Confusion Matrix
Naive Bayes - K Best TF-IDF	Unigram	1000	0.974	
		5000	0.979	

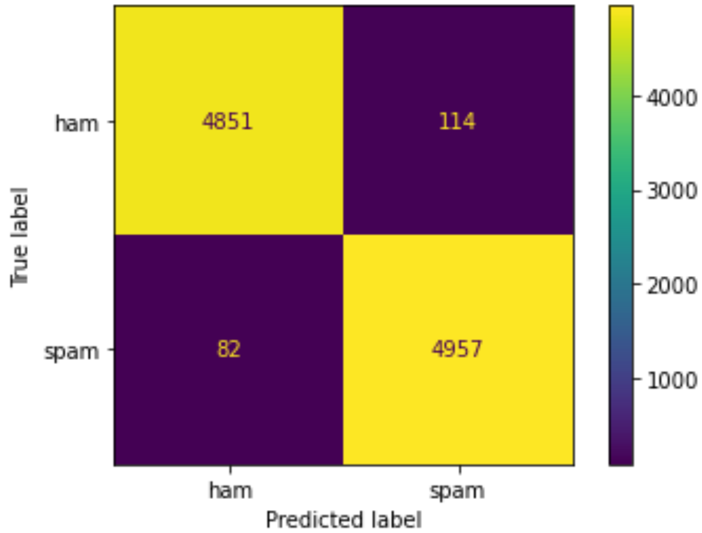
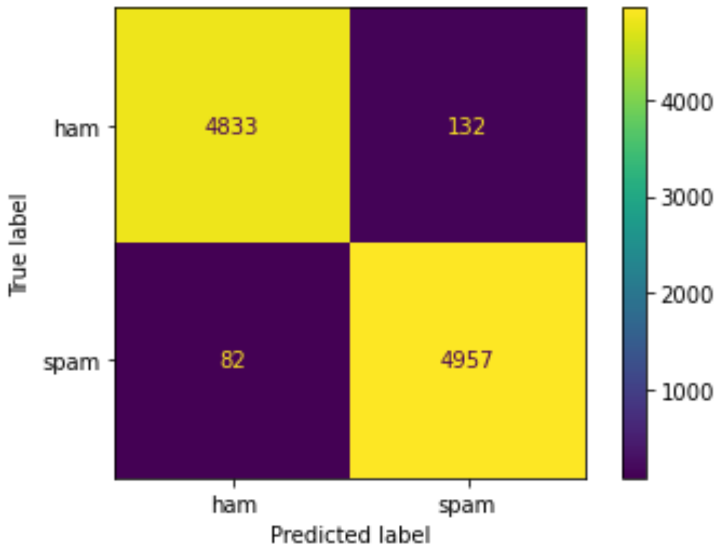
20. Figure 20

Model	Sub Type	Feature space	Accuracy	Confusion Matrix
SVM - K Best TF-IDF	Unigram	1000	0.981	
		5000	0.986	

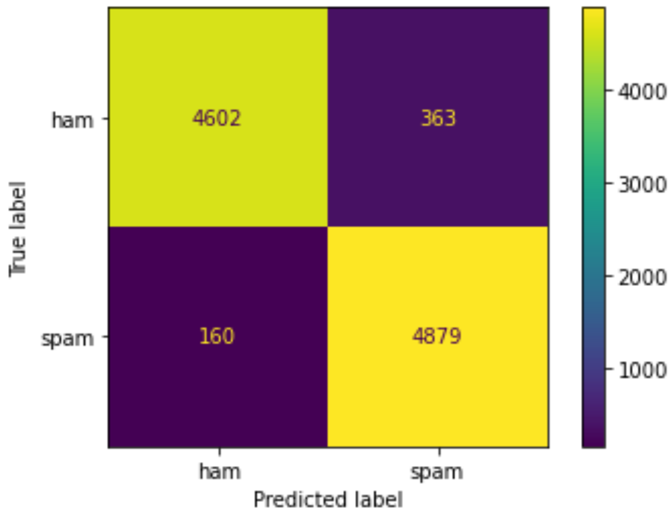
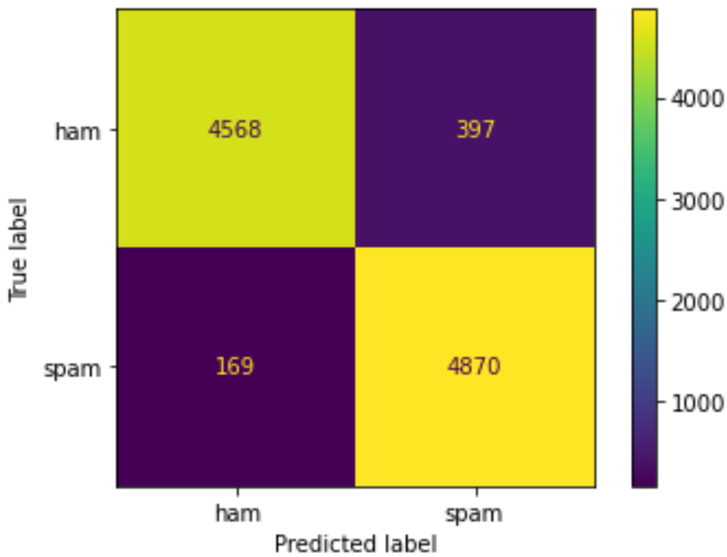
21. Figure 21

Model	Sub Type	Feature space	Accuracy	Confusion Matrix
Naive Bayes - Variance Threshold (0.001)	Unigram CV	14,910	0.980	
Naive Bayes - Variance Threshold (0.005)	Unigram CV	6,380	0.976	

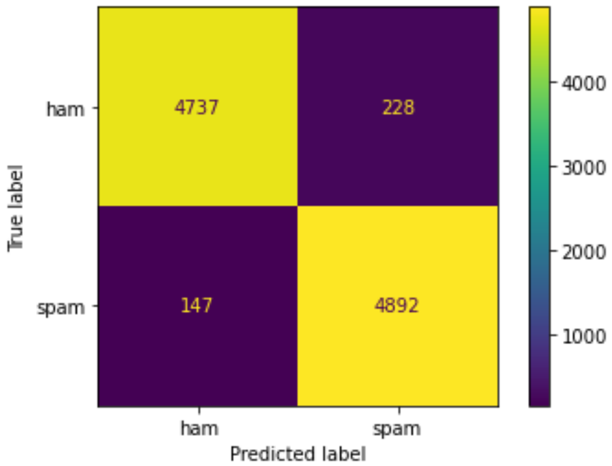
22. <u>Figure 22</u>

Model	**Sub Type**	**Feature space**	**Accuracy**	**Confusion Matrix**
SVM - Variance Threshold (0.001)	Unigram	14,910	0.980	
SVM - Variance Threshold (0.005)	Unigram	6,380	0.978	

23. Figure 23

Model	Sub Type	Feature space	Accuracy	Confusion Matrix
Decision Tree (GINI) Depth =35	Unigram TF-IDF	1000	0.948	
Decision Tree (Entropy) Depth =35	Unigram TF-IDF	1000	0.943	

24. Figure 24

Model	Sub Type	Feature space	Accuracy	Confusion Matrix
SVM - Word2Vec	-	23,341	0.963	

25. Table 1

Model	Sub Type	Feature space	Micro	Macro			Weighted		
			Precision/ Recall/ F1	Precision	Recall	F1	Precision	Recall	F1
Naive Bayes - Count Vectorizer	Unigram	1000	0.961	0.96	0.96	0.96	0.96	0.96	0.96
	Unigram +Bi Gram		0.961	0.96	0.96	0.96	0.96	0.96	0.96
	Unigram +Bigram +Trigram		0.958	0.96	0.96	0.96	0.96	0.96	0.96
Naive Bayes - TF-IDF	Unigram	1000	0.966	0.97	0.97	0.97	0.97	0.97	0.97
	Unigram +Bi Gram		0.965	0.96	0.96	0.96	0.96	0.96	0.96
	Unigram +Bigram +Trigram		0.963	0.96	0.96	0.96	0.96	0.96	0.96
SVM - Count Vectorizer	Unigram	1000	0.969	0.97	0.97	0.97	0.97	0.97	0.97
	Unigram +Bi Gram		0.968	0.97	0.97	0.97	0.97	0.97	0.97
	Unigram +Bigram +Trigram		0.967	0.97	0.97	0.97	0.97	0.97	0.97

SVM - TF-IDF	Unigram	1000	0.969	0.97	0.97	0.97	0.97	0.97	0.97
	Unigram +Bi Gram		0.968	0.97	0.97	0.97	0.97	0.97	0.97
	Unigram +Bigram +Trigram		0.968	0.97	0.97	0.97	0.97	0.97	0.97
KNN - Count Vectorizer (K=4)	Unigram	1000	0.926	0.93	0.93	0.93	0.93	0.93	0.93
	Unigram +Bi Gram		0.925	0.93	0.92	0.92	0.93	0.92	0.92
	Unigram +Bigram +Trigram		0.927	0.93	0.93	0.93	0.93	0.93	0.93
KNN - TF-IDF (K=4)	Unigram	1000	0.805	0.84	0.80	0.80	0.84	0.80	0.80
	Unigram +Bi Gram		0.816	0.85	0.82	0.81	0.85	0.82	0.81
	Unigram +Bigram +Trigram		0.817	0.85	0.82	0.81	0.85	0.82	0.81
Naive Bayes - K Best	Unigram	1000	0.974	0.97	0.97	0.97	0.97	0.97	0.97

TF-IDF	Unigram	5000	0.979	0.98	0.98	0.98	0.98	0.98	0.98
SVM - K Best TF-IDF	Unigram	1000	0.981	0.98	0.98	0.98	0.98	0.98	0.98
		5000	0.986	0.99	0.99	0.99	0.99	0.99	0.99
Naive Bayes - Variance Threshold (0.001)	Unigram CV	14,910	0.980	0.98	0.98	0.98	0.98	0.98	0.98
Naive Bayes - Variance Threshold (0.005)	Unigram CV	6,380	0.976	0.98	0.98	0.98	0.98	0.98	0.98
SVM - Variance Threshold (0.001)	Unigram	14,910	0.980	0.98	0.98	0.98	0.98	0.98	0.98
SVM - Variance Threshold (0.005)	Unigram	6,380	0.978	0.98	0.98	0.98	0.98	0.98	0.98
Decision Tree (GINI) Depth =35	Unigram TF-IDF	1000	0.948	0.95	0.95	0.95	0.95	0.95	0.95
Decision Tree (Entropy) Depth =35	Unigram TF-IDF	1000	0.943	0.94	0.94	0.94	0.94	0.94	0.94
SVM - Word2Vec	-	23,341	0.963	0.96	0.96	0.96	0.96	0.96	0.96