Notebook

April 27, 2020

0.0.1 Question 6c

Provide brief explanations of the results from 6a and 6b. Explain why the number of false positives, number of false negatives, accuracy, and recall all turned out the way they did.

The number of false positives is 0 because it never predicts positive. The number of false negatives is the number of emails labeled spam since it never predicts positive. The accuracy is the total number of emails labeled ham divided by the total number of emails since zero_predictor classifies every email as ham. The recall is 0 because there were 0 positives predicted.

0.0.2 Question 6e

Are there more false positives or false negatives when using the logistic regression classifier from Part A?

There are 122 false positives and 1699 false negatives, so there are more false negatives.

0.0.3 Question 6f

1. Our logistic regression classifier got 75.8% prediction accuracy (number of correct predictions / total). How does this compare with predicting 0 for every email?
2. Given the word features we gave you above, name one reason this classifier is performing poorly. Hint: Think about how prevalent these words are in the email set.
3. Which of these two classifiers would you prefer for a spam filter and why? Describe your reasoning and relate it to at least one of the evaluation metrics you have computed so far.

Our zero predictor accuracy was 74.4%, which is less than our logistic regression classifier. All 5 of the words are very common in both ham and spam emails, which lowers our accuracy. I would prefer the logistic regression classifer for a spam filter since it resulted in a higher predictor accuracy.

0.0.4 Question 7: Feature/Model Selection Process

In the following cell, describe the process of improving your model. You should use at least 2-3 sentences each to address the follow questions:

1. How did you find better features for your model?
2. What did you try that worked / didn't work?
3. What was surprising in your search for good features?

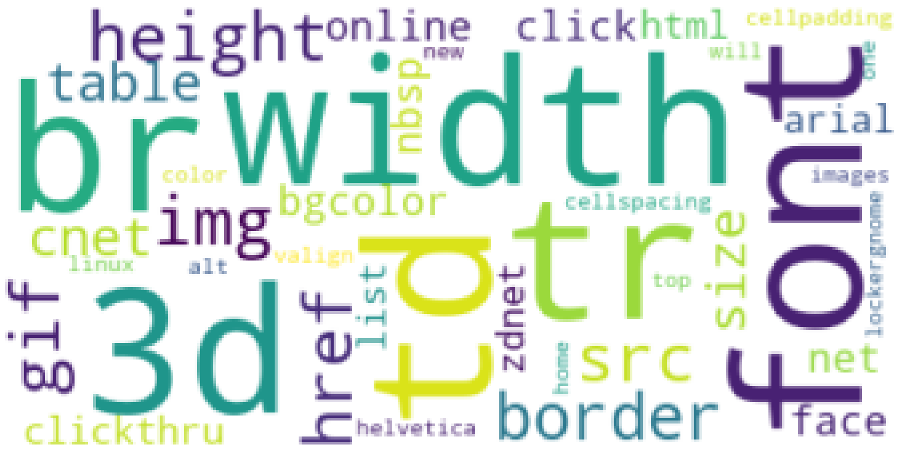
1. I used most of the ideas given for improving my model since I was a bit stuck as to where to begin.
2. The features that worked the best were finding which words appeared the most frequently, as well as whether an email was a reply or a forward. The features that did not work so well were searching for html tags and the number of characters in the subject and body of the emails. Adding the number of characters lowered my accuracy by a decent amount, and searching for html tags did not improve nor harm my accuracy.
3. Searching for punctuation, such as exclamation points, actually helped increase my accuracy by quite a bit. Also, in the beginning I filtered my list of best words from indices 100-200 since I thought I should avoid the most common words of the English dictionary and include words that are a bit less common but more able to help distinguish a ham from a spam. After having my accuracy hover in the 80s, I decided to try switching my indices from 0-100 instead and my accuracy rose drastically. I don't know why I expected a higher accuracy with less frequent words but I'm glad I tested different ranges and saw how much of a difference it made.

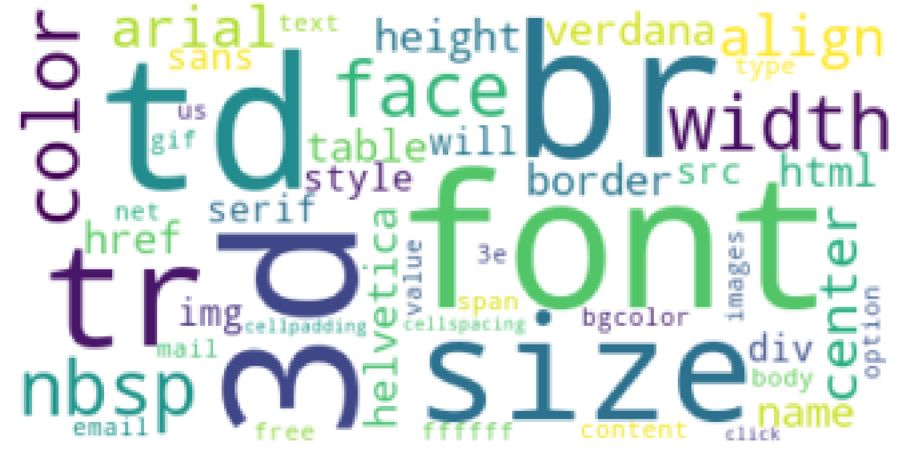
Generate your visualization in the cell below and provide your description in a comment.

```
In [22]: %pip install wordcloud
         from wordcloud import WordCloud, STOPWORDS

         # Write your description (2-3 sentences) as a comment here:
         # To depict which words had high or low values in spam and hams, I created two word clouds tha
         # with the most frequency between the two email types. In hams, "width", "3d", and "td" appear
         # "3d", "font", and "td" appeared very often. Looking at the most frequent words in the body o
         # large overlap between common words used between the two types. Although this may imply that
         # to filter spams and hams, spam emails more frequently contained inchoherent words and certai
         # "free" that makes it more liekly to distinguish between the two with a higher accuracy.

         # Write the code to generate your visualization here:

         def best_words(df):
             words_count = {}
             for row in df['email']:
                 words = re.findall('\w+', row)
                 for word in words:
                     if word in words_count:
                         words_count[word] += 1
                     else:
                         words_count[word] = 1
             return words_count

         def listToString(words):
             toString = ""
             for word in words:
                 toString = toString + word + " "
             return toString

         stopwords = set(STOPWORDS)

         ham = train[train['spam'] == 0]
         ham_best_list = (pd.Series(best_words(ham)) / train.shape[0]).sort_values(ascending = False).i
         ham_words = listToString(ham_best_list)
         wordcloud = WordCloud(stopwords = stopwords, max_words = 100, background_color = "white").gene

         plt.imshow(wordcloud, interpolation = 'bilinear')
         plt.axis("off")
         plt.show()

         spam = train[train['spam'] == 1]
         spam_best_list = (pd.Series(best_words(spam)) / train.shape[0]).sort_values(ascending = False)
         spam_words = listToString(spam_best_list)
         wordcloud = WordCloud(stopwords = stopwords, max_words = 100, background_color = "white").gene

         plt.imshow(wordcloud, interpolation = 'bilinear')
         plt.axis("off")
         plt.show()


         # Note: if your plot doesn't appear in the PDF, you should try uncommenting the following line
```

```
    # plt.show()
```

```
Requirement already satisfied: wordcloud in /srv/conda/envs/data100/lib/python3.7/site-packages (1.6.0)
Requirement already satisfied: numpy>=1.6.1 in /srv/conda/envs/data100/lib/python3.7/site-packages (fro
Requirement already satisfied: pillow in /srv/conda/envs/data100/lib/python3.7/site-packages (from word
… Omitting 5 lines …
Requirement already satisfied: setuptools in /srv/conda/envs/data100/lib/python3.7/site-packages (from
Note: you may need to restart the kernel to use updated packages.
```


br
width
font
td
tr
3d
height
online
click
html
cellpadding
table
new
will
one
nbsp
arial
color
images
img
bgcolor
cellspacing
cnet
linux
alt
valign
gif
href
list
zdnet
size
top
lockergnome
src
net
home
clickthru
helvetica
border
face


td
br
font
3d
size
tr
color
arial
text
height
verdana
align
sans
type
face
width
us
gif
table
will
style
border
src
html
net
serif
href
helvetica
value
3e
images
center
option
img
cellpadding
cellspacing
span
bgcolor
mail
nbsp
div
body
email
free
ffffff
content
click
name