In [1]:
```
# Initialize Otter
import otter
grader = otter.Notebook("projB2.ipynb")
```

Project B2: Spam/Ham Classification - Build Your Own Model

Feature Engineering, Classification, and Cross-Validation

Due Monday, August 5th, 11:59 PM PT

You must submit this assignment to Gradescope by the on-time deadline, Monday, August 5th, 11:59 PM PT. Please read the syllabus for the grace period policy. No late submissions beyond the grace period will be accepted. **We strongly encourage you to plan to submit your work to Gradescope several hours before the stated deadline.** This way, you will have ample time to reach out to staff for submission support. While course staff is happy to help you if you encounter difficulties with submission, we may not be able to respond to last-minute requests for assistance (TAs need to sleep, after all!).

Please read the instructions carefully when submitting your work to Gradescope.

Collaboration Policy

Data science is a collaborative activity. While you may talk with others about the project, we ask that you **write your solutions individually**. If you do discuss the assignments with others, please **include their names** in the collaborators cell below.

Collaborators: *list collaborators here*

Introduction

In this project, you will build and improve on the concepts and functions you implemented in Project B1 to create your own classifier to distinguish spam emails from ham (non-spam) emails. We will evaluate your work based on your model's accuracy and written responses in this notebook.

After this assignment, you should feel comfortable with the following:

- Using `sklearn` libraries to process data and fit classification models.
- Validating the performance of your model and minimizing overfitting.
- Generating and analyzing ROC curves.

Content Warning

This is a **real-world** dataset —— the emails you are trying to classify are actual spam and legitimate emails. As a result, some of the spam emails may be in poor taste or be considered inappropriate. We think the benefit of working with realistic data outweighs these inappropriate emails, but we wanted to warn you at the beginning of the project so that you are made aware.

If you feel uncomfortable with this topic, **please contact your TA, the instructors, or reach out via the accommodations form.**

In [2]:
```
# Run this cell to suppress all FutureWarnings
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

Grading

Grading is broken down into autograded answers and free responses.

For autograded answers, the results of your code are compared to provided and/or hidden tests.

For free response questions, readers will evaluate how well you answered the question and/or fulfilled the requirements of the question.

Question	Manual	Points
1a	Yes	4
1b	Yes	2
2	No	0
3a	No	5
3b	No	10
4	Yes	6
5	Yes	3
6a	Yes	3
6b	Yes	2
7ai	No	1
7aii	Yes	1
7bi	Yes	1
7bii	Yes	1
7c	Yes	1
7d	Yes	2

Question	Manual	Points
7e	Yes	2
Total	12	44

Before You Start

For each question in the assignment, please write down your answer in the answer cell(s) right below the question.

We understand that it is helpful to have extra cells breaking down the process of reaching your final answer. If you happen to create new cells below your answer to run code, **NEVER** add cells between a question cell and the answer cell below it. It will cause errors when we run the autograder, and it will sometimes cause a failure to generate the PDF file.

Important note: The local autograder tests will not be comprehensive. You can pass the automated tests in your notebook but still fail tests in the autograder. Please be sure to check your results carefully.

Debugging Guide

If you run into any technical issues, we highly recommend checking out the Data 100 Debugging Guide. This guide contains general questions about Jupyter notebooks / Datahub, Gradescope, common `pandas` errors, RegEx, visualizations, and more.

In [3]:
```
import numpy as np
import pandas as pd
import sys

import matplotlib.pyplot as plt
%matplotlib inline

import seaborn as sns
sns.set(style = "whitegrid",
        color_codes = True,
        font_scale = 1.5)

from datetime import datetime
from IPython.display import display, HTML
```

Setup and Recap

Here, we will provide a summary of Project B1 to remind you of how we cleaned the data, explored it, and implemented methods helpful in building your own model.

Loading and Cleaning Data

Remember that in the email classification task, our goal is to classify emails as spam or not spam (referred to as "ham") using features generated from the text in the email.

The dataset consists of email messages and their labels (0 for ham, 1 for spam). Your labeled training dataset contains 8,348 labeled examples, and the unlabeled test set contains 1,000 unlabeled examples.

Run the following cell to load the data into a `DataFrame`.

The `train` `DataFrame` contains labeled data (8,348 labeled emails) that you will use to train your model. It contains four columns:

1. `id`: An identifier for the training example.
2. `subject`: The subject of the email.
3. `email`: The text of the email.
4. `spam`: 1 if the email is spam, 0 if the email is ham (not spam).

The `test` `DataFrame` contains 1,000 unlabeled emails. You will predict labels for these emails and submit your predictions to the autograder for evaluation.

In [4]:
```
import zipfile
with zipfile.ZipFile('spam_ham_data.zip') as item:
    item.extractall()
```

In [5]:
```
original_training_data = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')

# Convert the emails to lowercase as the first step of text processing.
original_training_data['email'] = original_training_data['email'].str.lower()
test['email'] = test['email'].str.lower()

original_training_data.head()
```

Out[5]:

	id	subject	email	spam
0	0	Subject: A&L Daily to be auctioned in bankrupt...	url: http://boingboing.net/#85534171\n date: n...	0
1	1	Subject: Wired: "Stronger ties between ISPs an...	url: http://scriptingnews.userland.com/backiss...	0
2	2	Subject: It's just too small ...	<html>\n <head>\n </head>\n <body>\n <font siz...	1
3	3	Subject: liberal defnitions\n	depends on how much over spending vs. how much...	0
4	4	Subject: RE: [ILUG] Newbie seeks advice - Suse...	hehe sorry but if you hit caps lock twice the ...	0

Feel free to explore the dataset above along with any specific spam and ham emails that interest you. Keep in mind that our data may contain missing values, which are handled in the following cell.

In [6]:
```
# Fill any missing or NAN values.
print('Before imputation:')
print(original_training_data.isnull().sum())
original_training_data = original_training_data.fillna('')
print('------------')
print('After imputation:')
print(original_training_data.isnull().sum())
```

```
Before imputation:
id         0
subject    6
email      0
spam       0
dtype: int64
------------
After imputation:
id         0
subject    0
email      0
spam       0
dtype: int64
```

Training/Validation Split

Recall that the training data we downloaded is all the data we have available for both training models and **validating** the models that we train. Therefore, we split the training data into separate training and validation datasets. Once you have finished training, you will need this validation data to assess the performance of your classifier.

As in Project B1, we set the seed (`random_state`) to 42. **Do not modify this in the following questions, as our tests depend on this random seed.**

In [7]:
```
# This creates a 90/10 train-validation split on our labeled data.
from sklearn.model_selection import train_test_split
```

```
train, val = train_test_split(original_training_data, test_size = 0.1, random_st

# We must do this in order to preserve the ordering of emails to labels for word
train = train.reset_index(drop = True)
```

Feature Engineering

We need a numeric feature matrix $\mathbb{X}$ and a vector of corresponding binary labels $\mathbb{Y}$ to train a logistic regression model. In Project B1, we implemented the function `words_in_texts`, which creates numeric features derived from the email text and uses those features for logistic regression.

For this project, we have provided you with an implemented version of `words_in_texts`. Remember that the function outputs a 2-dimensional `NumPy` array containing one row for each email text. The row should contain a 0 or a 1 for each word in the list: 0 if the word doesn't appear in the text and 1 if the word does.

Run the following cell to see how the function works on some text.

In [8]:
```
from projB2_utils import words_in_texts

words_in_texts(['hello', 'bye', 'world'], pd.Series(['hello', 'hello worldhello'
```

Out[8]:
```
array([[1, 0, 0],
       [1, 0, 1]])
```

EDA and Basic Classification

In Project B1, we visualized the frequency of different words in spam and ham emails and used `words_in_texts(words, train['email'])` to train a classifier directly. We also provided a simple set of 5 words that might be useful as features to distinguish spam/ham emails.

We then built a model using the `LogisticRegression` classifier from `sklearn`.

Run the following cell to see the performance of a simple model using these words and the `train` `DataFrame`.

In [9]:
```
some_words = ['drug', 'bank', 'prescription', 'memo', 'private']

X_train = words_in_texts(some_words, train['email'])
Y_train = np.array(train['spam'])

X_train[:5], Y_train[:5]
```

Out[9]:
```
(array([[0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0],
        [0, 0, 0, 1, 0]]),
 array([0, 0, 0, 0, 0]))
```

In [10]:
```
from sklearn.linear_model import LogisticRegression

simple_model = LogisticRegression()
simple_model.fit(X_train, Y_train)

training_accuracy = simple_model.score(X_train, Y_train)
print("Training Accuracy: ", training_accuracy)
```

```
Training Accuracy:  0.7576201251164648
```

Evaluating Classifiers

In our models, we evaluate the accuracy of the training set, which may provide a misleading accuracy measure. In Project B1, we calculated various metrics to consider other ways of evaluating a classifier, in addition to overall accuracy. Below is a reference to those concepts.

Presumably, our classifier will be used for **filtering**, or preventing messages labeled `spam` from reaching someone's inbox. There are two kinds of errors we can make:

- **False positive (FP)**: A ham email gets flagged as spam and filtered out of the inbox.
- **False negative (FN)**: A spam email gets mislabeled as ham and ends up in the inbox.

To be clear, we label spam emails as 1 and ham emails as 0. These definitions depend both on the true labels and the predicted labels. False positives and false negatives may be of differing importance, leading us to consider more ways of evaluating a classifier in addition to overall accuracy:

Precision: Measures the proportion of emails flagged as spam that are actually spam. Mathematically, $\frac{\text{TP}}{\text{TP} + \text{FP}}$.

Recall: Measures the proportion of spam emails that were correctly flagged as spam. Mathematically, $\frac{\text{TP}}{\text{TP} + \text{FN}}$.

False positive rate: Measures the proportion of ham emails that were incorrectly flagged as spam. Mathematically, $\frac{\text{FP}}{\text{FP} + \text{TN}}$.

The below graphic (modified slightly from Wikipedia) may help you understand precision and recall visually:

Note that a True Positive (TP) is a spam email that is classified as spam, and a True Negative (TN) is a ham email that is classified as ham.

Moving Forward

With this in mind, it is now your task to make the spam filter more accurate. To get full credit on the accuracy part of this assignment, you must get at least **85%** accuracy on both the train and test set (see Question 3 for the partial credit breakdown). To determine your accuracy on the test set, you will use your classifier to predict every email in the `test DataFrame` and upload your predictions to Gradescope.

You will only be able to submit your test set predictions to Gradescope up to 4 times per day. You will be able to see your accuracy on the entire test set when submitting to Gradescope. Note that attempts will not carry over across days, so we recommend planning ahead to make sure you have enough time to finetune your model! In the case that you are approved for an extension, you are granted 4 more submissions for each day the deadline has been extended.

Here are some ideas for improving your model:

1. Finding better features based on the email text. Some example features are:
    A. Number of characters in the subject/body
    B. Number of words in the subject/body
    C. Use of punctuation (e.g., how many '!'s were there?)
    D. Number/percentage of capital letters
    E. Whether the email is a reply to an earlier email or a forwarded email
2. Finding better words to use as features. Which words are the best at distinguishing emails? This requires digging into the email text itself. Alternatively, you can identify misclassified emails and see which relevant words are missing in your model.
3. Reducing dimensionality and/or multicollinearity.
    A. One way to do this is to interpret the model coefficients. Note that a feature will be more valuable in classification if its coefficient has a larger **absolute** value. If the coefficient has a lower **absolute** value, the feature likely isn't valuable in classifying emails.
4. Better data processing. For example, many emails contain HTML as well as text. You can consider extracting the text from the HTML to help you find better words. Or, you can match HTML tags themselves, or even some combination of the two.
5. Model selection. You can adjust the parameters of your model (e.g. the penalty type, the regularization parameter, or any arguments in `LogisticRegression`) to achieve higher accuracy. Recall that you should use cross-validation for feature and model selection! Otherwise, you will likely overfit to your training data.
    A. Consider implementing L1 regularization. The documentation for `LogisticRegression` may be helpful here.
    B. We have imported `GridSearchCV` for you. You may use sklearn's `GridSearchCV` (documentation) class to perform cross-validation. You do not need to code cross-validation from scratch, though you are welcome to do so.

Here's an example of how to use `GridSearchCV`. Suppose we wanted to experiment with 4 different solvers (numerical methods for optimizing the mode) models for a `LogisticRegression` model `lr_model`.

1. We could define a dictionary specifying the hyperparameters and the specific values we want to try out like so: `parameters = {'solver':[{'lbfgs', 'liblinear', 'newton-cg', 'saga']}`.
2. Running `grid = GridSearchCV(estimator=lr_model, param_grid=parameters)` would give us a model for each combination of hyperparameters we are testing - in this case, just 4 models.
3. We fit each model to some training data `X_train` and `Y_train` using `grid_result = grid.fit(X_train, Y_train)`.
4. Indexing into `grid_result.cv_results_` with a particular metric (in this case, `mean_test_score`), we get an array with the scores corresponding to each of the models. `grid_result.cv_results_['mean_test_score']`. Feel free to experiment with other hyperparameters and metrics as well. The documentation is your friend!

You may use whatever method you prefer to create features, but **you may only use the packages we've imported for you in the cell below or earlier in this notebook**. In addition, **you are only allowed to train logistic regression models**. No decision trees, random forests, k-nearest-neighbors, neural nets, etc.

Note 1: You may want to use your **validation data** to evaluate your model and get a better sense of how it will perform on the test set. However, you may overfit to your validation set if you try to optimize your validation accuracy too much. Alternatively, you can perform cross-validation on the entire training set.

Note 2: If you see a `ConvergenceWarning`, increase the maximum number of iterations the model runs for by passing in a parameter, `max_iter`, into `LogisticRegression()`. This should get rid of the warning. For a longer discussion on why this warning appears, you might find this StackOverflow post helpful. Convergence of solvers is not in scope for Data 100, but by understanding what the error messages are saying, you can get some useful context on what to do in these situations.

Question 1: Exploratory Data Analysis

To decide which features to use when building your model, it is helpful to conduct EDA. Show a visualization you used to select features for your model.

Please include:

1. A plot showing something meaningful about the data that helped you during feature selection, model selection, or both.
2. Two or three sentences describing what you plotted and its implications with respect to your features.

You can create as many plots as you want in your feature selection process, but you should select only one for the response question below.

You should not just produce an identical visualization to Question 3 in Project B1. For this section, we'd like you to go beyond the analysis you performed in Project B1. Choose some plot other than the 1-dimensional distribution of some quantity for spam and ham emails. In particular, do not produce a bar plot of proportions like you created in Question 3 of Project B1. Any other plot is acceptable, **as long as it comes with thoughtful commentary.** Here are some ideas:

1. Consider the correlation between multiple features (look up correlation plots and `sns.heatmap` (documentation)).
2. Try to show redundancy in a group of features (e.g., `body` and `html` might co-occur relatively frequently, or you might be able to design a feature that captures all HTML tags and compares them to these).
3. Visualize which words have high or low values for helpful statistics.
4. Visually depict whether spam emails tend to be wordier (in some sense) than ham emails.

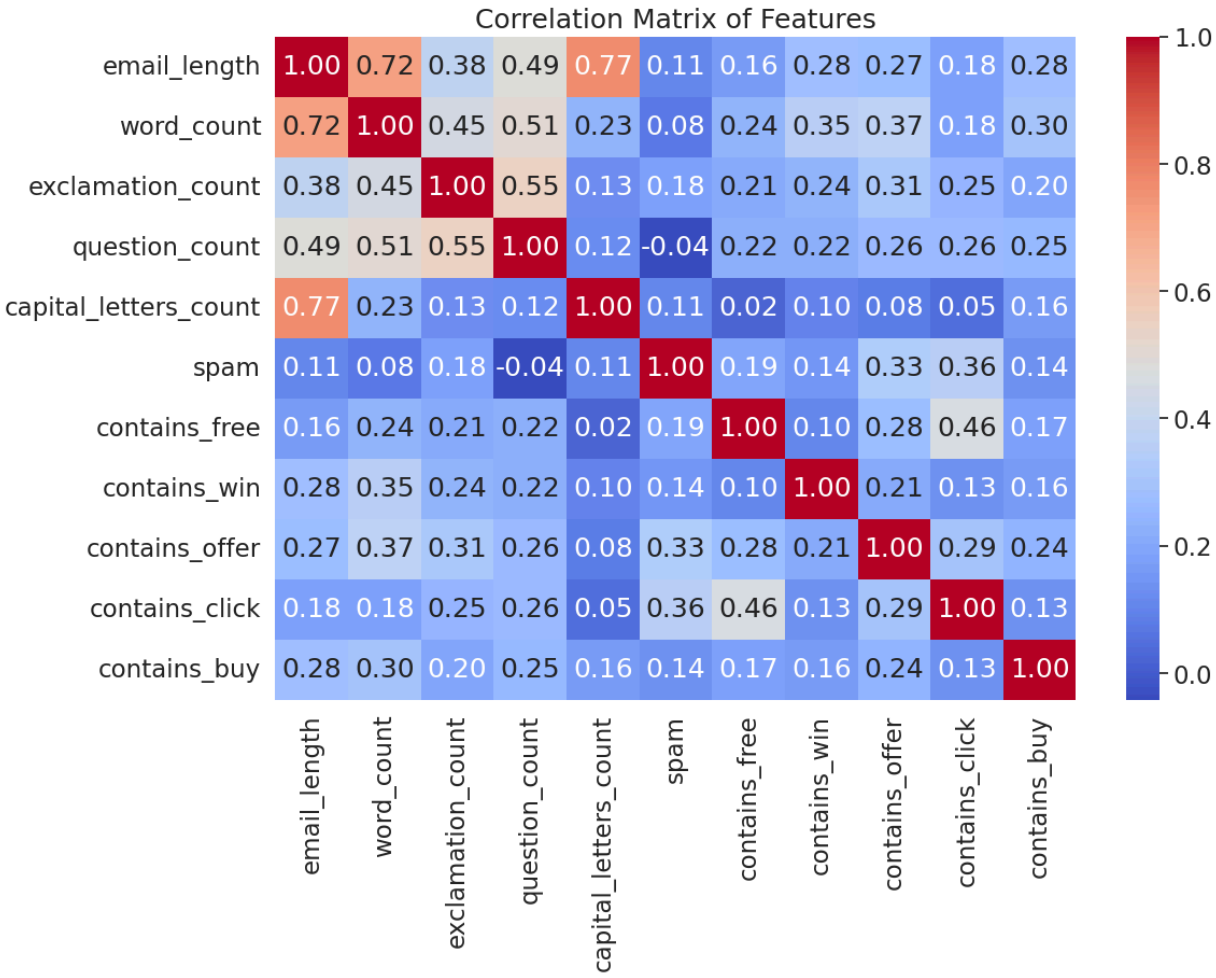
In [11]:
```
# import libraries
# You may use any of these to perform EDA (Question 1) or create your features (
from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, roc_curve, confusion_matrix
from sklearn.model_selection import GridSearchCV
from sklearn.decomposition import PCA
import re
from collections import Counter
```

Question 1a

Generate your visualization in the cell below.

In [12]:
```
train_data = pd.read_csv('train.csv')

# I am going to create features that help us pull more meaningful data from the
train_data['email_length'] = train_data['email'].str.len()
train_data['word_count'] = train_data['email'].apply(lambda x: len(x.split()))
train_data['exclamation_count'] = train_data['email'].apply(lambda x: x.count('!
train_data['question_count'] = train_data['email'].apply(lambda x: x.count('?'))
train_data['capital_letters_count'] = train_data['email'].apply(lambda x: sum(1

# Pull out most common SPAM words as from 1B.
spam_keywords = ['free', 'win', 'offer', 'click', 'buy']
for word in spam_keywords:
    train_data[f'contains_{word}'] = train_data['email'].apply(lambda x: int(wor

correlation_matrix = train_data[['email_length', 'word_count', 'exclamation_cour
```

```
                                    'question_count', 'capital_letters_count', 'spa
                                   [f'contains_{word}' for word in spam_keywords]].

plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix of Features')
plt.show()
```



Question 1b

In two to three sentences, describe what you plotted and its implications with respect to your features.

The heat map shows a correlation between the different features of the email dataset, it showcases the connection between key characteristics such as email length word count exclamation count capital letters and specific key words that I selected after looking up the most highly trafficked words in spam emails. As you can see from the heat map specific words such as free, offer, and click seem to have a higher corrrelation to spam emails that other features. However, overall the features i have devloped dont seem to show high correlation to spam emails as the correlation values would need to be closer to 1.0. Thus, as I

go through further iterations of the project I should increase the accuracy of my model by increasing the features I have and more accurately capturing what makes a spam email a spam email.

Question 2: Building Your Own Model

Now that you've explored the data and relevant features through EDA, it's time to build your model! As mentioned earlier, you may use whatever method you prefer to create features, but **you may only use the packages we've imported for you earlier in this notebook**. In addition, **you are only allowed to train logistic regression models**. No decision trees, random forests, k-nearest-neighbors, neural nets, etc.

Please consider the ideas mentioned above when choosing features. We have not provided any code to do this, so feel free to create as many cells as you need to tackle this task.

In [13]:
```
# Define your processing function, processed data, and model here.
# You may find it helpful to look through the rest of the questions first!

original_training_data = pd.read_csv('train.csv')

train, val = train_test_split(original_training_data, test_size=0.1, random_stat

def preprocess_email(emails):
    exclamations = emails.apply(lambda x: x.count('!'))
    questions = emails.apply(lambda x: x.count('?'))
    capitals = emails.apply(lambda x: sum(1 for c in x if c.isupper()))
    email_length = emails.apply(len)
    word_count = emails.apply(lambda x: len(x.split()))
    html_tags = emails.apply(lambda x: int('<html>' in x or '</html>' in x))

    return np.vstack([exclamations, questions, capitals, email_length, word_cour

some_words = [
    'free', 'win', 'offer', 'click', 'buy', 'money', 'prescription', 'limited',
    'now', 'urgent', 'lottery', 'credit', 'card', 'discount', 'guarantee', 'deal
    'unsubscribe', 'congratulations', 'gift', 'bonus', 'prize', 'exclusive', 'ac
]

X_train_words = words_in_texts(some_words, train['email'])
X_train_additional = preprocess_email(train['email'])

X_train = np.hstack([X_train_words, X_train_additional])
Y_train = train['spam']

model = LogisticRegression(max_iter=1000, C=0.3, solver='liblinear', penalty='l1
model.fit(X_train, Y_train)

train_predictions = model.predict(X_train)
training_accuracy = np.mean(train_predictions == Y_train)
```

```
print("Training Accuracy:", training_accuracy)
```

```
Training Accuracy: 0.873153201118062
```

Question 3

Grading Scheme

Your grade for this question will be based on your model's accuracy when making predictions on the training set and your model's accuracy when making predictions on the test set. The tables below provide scoring guidelines. If your accuracy lies in a particular range, you will receive the number of points associated with that range.

Important: While your training accuracy can be checked at any time in this notebook, your test accuracy can only be checked by submitting your model's predictions to Gradescope. **You will only be able to submit your test set predictions to Gradescope up to 4 times per day**. In the case that you are approved for an extension, you are granted 4 more submissions for each day the deadline has been extended. Plan ahead to make sure you have enough time to fine-tune your model! The thresholds are as follows:

Points	**5**	**3**	**1.5**	**0**
Training Accuracy	85% and Above	[80, 85)	[70, 80)	Below 70%

Points	**10**	**6**	**3**	**0**
Testing Accuracy	85% and Above	[80, 85)	[70, 80)	Below 70%

Question 3a: Train Predictions

Assign your predictions for the class of each data point in the training set `train` to `train_predictions`.

In [14]:
```
train_predictions = model.predict(X_train)

# Print your training accuracy.
training_accuracy = np.mean(train_predictions == train["spam"])
training_accuracy
```

Out[14]:
```
0.873153201118062
```

In [15]:
```
grader.check("q3a")
```

Out[15]:

q3a passed! ✨

Question 3b: Test Predictions

The following code will write your predictions on the test dataset to a CSV file. **You will need to submit this file to the "Project B2 Test Set Predictions" assignment on Gradescope to get credit for this question.**

Assign your predictions for the class of each datapoint in the test set `test` to a 1-dimensional array called `test_predictions`. **Please make sure you save your predictions to `test_predictions`, as this is how part of your score for this question will be determined.**

Remember that if you've performed transformations or featurization on the training data, you must also perform the same transformations on the test data in order to make predictions. For example, if you've created features for the words "drug" and "money" on the training data, you must also extract the same features in order to use `scikit-learn`'s `.predict` method.

Gradescope limits you to 4 submissions per day to meet the threshold. If you are approved for an extension, you are granted 4 more submissions for each day the deadline has been extended.

The provided tests check that your predictions are in the correct format but are worth 0 points in the *Project B2 Coding assignment*. To evaluate your classifier accuracy, you must submit the CSV file to the *Project B2 Test Set Predictions* assignment.

In [16]:
```
# Below implementation was done through the help of Stackoverflow

test_data = pd.read_csv('test.csv')

def preprocess_email(emails):
    exclamations = emails.apply(lambda x: x.count('!'))
    questions = emails.apply(lambda x: x.count('?'))
    capitals = emails.apply(lambda x: sum(1 for c in x if c.isupper()))
    email_length = emails.apply(len)
    word_count = emails.apply(lambda x: len(x.split()))
    html_tags = emails.apply(lambda x: int('<html>' in x or '</html>' in x))

    return np.vstack([exclamations, questions, capitals, email_length, word_cour

X_test_words = words_in_texts(some_words, test_data['email'])
X_test_additional = preprocess_email(test_data['email'])

X_test = np.hstack([X_test_words, X_test_additional])

test_predictions = model.predict(X_test)
```

In [17]:
```
grader.check("q3b")
```

Out[17]:

q3b passed! ✨

The following cell generates a CSV file with your predictions. **You must submit this CSV file to the "Project B2 Test Set Predictions" assignment on Gradescope to get credit for this question.** You can only submit to Gradescope a maximum of 4 times per day, so please use your submissions wisely!

In [18]:
```
# Assuming that your predictions on the test set are stored in a 1-dimensional a
# test_predictions. Feel free to modify this cell as long you create a CSV in th

# Construct and save the submission:
submission_df = pd.DataFrame({
    "Id": test['id'],
    "Class": test_predictions,
}, columns=['Id', 'Class'])
timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
filename = "submission_{}.csv".format(timestamp)
submission_df.to_csv(filename, index=False)

print('Created a CSV file: {}.'.format("submission_{}.csv".format(timestamp)))
display(HTML("Download your test prediction <a href='" + filename + "' download>
print('You may now upload this CSV file to Gradescope for scoring.')#
```

```
Created a CSV file: submission_20240805_023647.csv.
```
Download your test prediction here.
```
You may now upload this CSV file to Gradescope for scoring.
```

Analyzing Your Model

Congratulations on completing your model! In the next few questions, we'll ask you to comment on your process for building a successful model and better understand the amount of misclassifications your model makes.

Question 4

Describe the process of improving your model. You should use at least 2-3 sentences each to address the following questions:

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

1. I analyzed the correlation between various features from the email and determing their correlation with my target variable (spam or ham) I then identified features that had a stronger assosiation with spam emails and biased towards them.
2. I initially tried adding features based only on the presence of common words as we did with 2A as well as the email identifiers (email length, capitalization and so fourth) and it actually did rather well. However, it was not nearly enough to surpass the 80% threshold for accuracy. Thus, I decided to take the hints the project kept throwing as me and include a broader range of features such as the email structure (the html tags we discussed in project 2A) This improved the models performance but I was still right under that 85% threshold. I believe this was because I added too many similar features which led to quite a bit of multicollinearity. Thus, I decided to choose more impactful features to mitgate this issue. I added a larger variety of trigger words and ended up increasing my models accuracy from 79% to a whopping 87%. Given i struggled with the model developement in project 1A this made me feel extremely happy.
3. What I found the most suprising about what determined a features as a good feature was the affect certain trigger words had on the detection of spam emails. Infact, the largest jump I had in model accuracy was not from addding HTML tag captures but rather increasing my data base of searchable words. This suprised me as I assumed the key to incerasing accuracy was creating extremely complex features. But rather, it was much more simplistic than I first assumed.

Question 5: ROC Curve

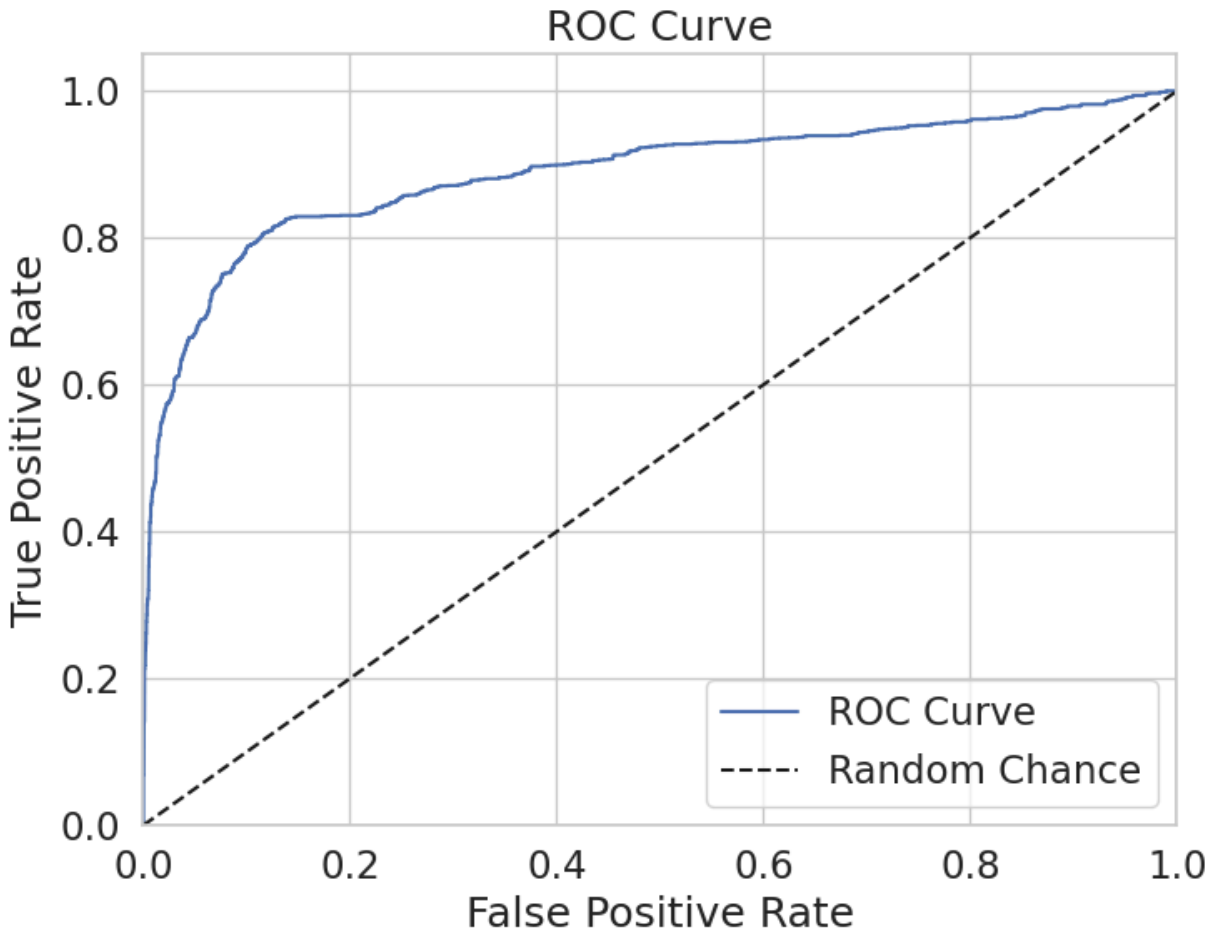
In most cases, we won't be able to get 0 false positives and 0 false negatives, so we have to compromise. For example, in the case of cancer screenings, false negatives are comparatively worse than false positives — a false negative means that a patient might not discover that they have cancer until it's too late. In contrast, a patient can receive another screening for a false positive.

Recall that logistic regression calculates the probability that an example belongs to a particular class. To classify an example, we say that an email is spam if our classifier gives it ≥ 0.5 probability of being spam. However, **we can adjust that cutoff threshold**. We can say that an email is spam only if our classifier gives it ≥ 0.7 probability of being spam, for example. This is how we can trade off false positives and false negatives.

The Receiver Operating Characteristic (ROC) curve shows this trade-off for each possible cutoff probability. In the cell below, plot an ROC curve for your final classifier (the one you use to make predictions for Gradescope) on the training data. Lecture 22 may be helpful.

Hint: You'll want to use the `.predict_proba` method (documentation) for your classifier instead of `.predict` to get probabilities instead of binary predictions.

In [19]:
```
probabilities = model.predict_proba(X_train)[:, 1]

fpr, tpr, _ = roc_curve(Y_train, probabilities)

# Plot the ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label='ROC Curve', color='b')
plt.plot([0, 1], [0, 1], 'k--', label='Random Chance')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc='lower right')
plt.show()
```



Diving Deeper

So far, we've been looking at our model through the lens of accuracy. In the next two questions, we'll dive deeper into the complexities of analyzing our model's performance. In particular, we'll ask you to explore some ambiguous cases that can arise, even within the

training data itself, and the consequences of misclassification. You may have already come across some of these cases unknowingly when building your model!

Question 6

To help you better understand some of the challenges that arise with classification, we've selected three emails from the `train` `DataFrame` and provided them below. Each email highlights a different issue that could arise. Skim through each of the emails below before answering part a).

Example 1

In [20]:
```
# Just run this cell, don't modify it.

print("spam: " + str(train.loc[5216]["spam"]))
print("\nemail:\n" + train.loc[5216]["email"])
```

```
spam: 1

email:
MR MABO LOKO
 BRANCH OFFICER,
 UNITED BANK FOR AFRICA PLC
 ILUPEJU LAGOS NIGERIA


 ATT:PRESIDENT/C.E.O

 SIR

           AN URGENT BUSINESS PROPOSAL
 I am pleased to get across to you for a very urgent and profitable
 business proposal,Though I don't know you neither have I seen you before but my
confidence was reposed On you when the Chief Executive of Lagos State
 chamber of Commerce and Industry handed me your contact for a
 confidential business.

 I am AN OFFICER with the United Bank for Africa Plc (UBA),Ilupeju branch,
 Lagos Nigeria.

 The intended business is thus; We had a customer, a Foreigner (a
 Turkish)resident in Nigeria, he was a Contractor with one of the Government Par
astatals. He has in his Account in my branch the sum of US$35 Million(Thirty fiv
e Million U.S. Dollars).

 Unfortunately, the man died four years ago until today non-of his next
 of kin has come Forward to claim the money.
 Having noticed this, I in collaboration with two other top Officials
 of the bank we have covered up the account all this while.
  Now we want you (being a foreigner) to be fronted as one of his next
 of kin and forward Your account and other relevant documents to be
 advised to you by us to attest to the Claim.

 We will use our positions to get all internal documentations to back
 up the claims .The whole procedures will last only ten working days to
 get the fund retrieved successfully Without trace even in future.

 Your response is only what we are waiting for as we have arranged all
 necessary things  As soon as this message comes to you kindly get back to me in
dicating your interest ,Then I will furnish you with the whole procedures to ens
ure that the deal is successfully Concluded

 For your assistance we have agreed to give you thirty percent (30%) of
 the Total sum at the end of the transaction. It is risk free and a mega
 fortune.
 All correspondence Towards this transaction will be through the e-mail
 addresses.

 I await your earliest response.
 Thanks,

 Yours Sincerel

 MR MABO LOKO
```

```
--DeathToSpamDeathToSpamDeathToSpam--


_______________________________________________________
This sf.net email is sponsored by:ThinkGeek
Welcome to geek heaven.
http://thinkgeek.com/sf
_____________________________________________
Spamassassin-Sightings mailing list
Spamassassin-Sightings@lists.sourceforge.net
https://lists.sourceforge.net/lists/listinfo/spamassassin-sightings
```

Example 2

In [21]:
```
# Just run this cell, don't modify it.

print("spam: " + str(train.loc[36]["spam"]))
print("\nemail:\n" + train.loc[36]["email"])
```

spam: 1

email:

Educate yourself about everything you ever wanted to know !!!

1. Satellite TV/RCA Dish Descrambler

Our unique, complete plans for building your own home satellite TV descrambler. For access to a clear reception of pay TV signals on your home satellite dish! I t does not require any additional equipment! Complete PC board template & instru ctions! This also includes a manual for accessing the RCA/DSS satellite dish to receive free pay channels. All the latest information for doing it right.

2. X-Ray Envelope Spray

Have you ever wanted to read the contents of an envelope without opening it? Ma ny government and other organizations use what is known as X-Ray Envelope Spray to do this! An envelope is sprayed with this secret chemical and it becomes tran slucent for a short period of time, which allows the contents to be read without opening. Private supply houses sell small cans of this aerosol spray for up to $50 a can! The spray is actually a commonly available item found in major grocer y and discount stores. No modification of the spray is needed, as it is ready to use as X-Ray Envelope Spray and sells for about $1.99 in retail stores!

3. How to Find Anyone and Obtain Unlisted Phone Numbers

Tired of getting the wrong number? Stop looking! We can help! We'll show you ho w to get the unlisted phone number of anyone. No one can hide now! Simple. Skip tracers use these tricks. We also include everything you need to know about find ing missing people or loved ones from the comfort of your home. Why pay money wh en you can do it yourself?

4. Radar Zapper

This simply technique converts existing radar detectors into a device that will jam police radar. This device sends false readings back to the police radar! Wor ks on virtually all detectors and easy to use!

5. Untraceable E-Mail

How to send totally anonymous and untraceable E-mail. We're not talking about t hose generic Yahoo! Accounts -- this is the real McCoy, anonymous email. Everyth ing explained. Absolutely untraceable.

6. Underground Guide to Utility Meters

The illustrated guide to gas, water & electric meters! We show you in detail me thods many people use to stop, slow down, even reverse all three types! This und erground manual is one of our most popular items! Complete illustrated technique s and easy to do. Shows how to defeat all major brands & models of gas, water & electric meters.

7. Scan-Tron Genius

Here at last!! This very controversial report describes in detail how any stude

nt can easily defeat Scan-Tron test readers to pass a test even though he does n ot know the answer! This simple method will fool the scan reader into thinking y ou answered correctly! No tools needed. Completely tested. You won't believe how simple this method is!

8. Bad Credit Cleaning Manual

Simple ways to restore your bad credit rating to A++. Don't pay a credit counse lor good money to do what you can do yourself. Many methods presented here - som e legal & some "not so legal". Wipe your slate clean from your own home. Get a f resh start.

9. Pass Drug Tests

Don't use of drugs! However, many innocent people are victims of drug testing. Some over the counter medicine can trigger false results & cost you your job. Pr oven methods to beat drug tests. We show you how to build a simple device that c an fool the best! Protect yourself & your job, even if you don't use drugs.

10.Cable TV Decoders

How to get cable TV and turn your converter box into "full service" mode. This is the latest and best way to gain access. Also, how to build your own "snooper stopper" for pennies. Prevents cable companies from spying on you.

11. Free Long Distance

You can make long distance calls to other countries at no cost! The information in these reports explains everything you need to call other countries! Country c odes, city codes, overseas sender codes! Call England, Germany, the UK, practica lly anywhere!

12. Dissolving Checks

We show you in detail the "insufficient funds" checks scam used by people to ob tain goods & cash without having any money in the account. Many people do not ev en use false ID's in pulling this scam off. Complete detailed instructions plus rare information on the famous "dissolving" checks. These checks "dissolve" afte r being chemically coated & deposited in the bank leaving no trace of the writer or account number. Not for illegal purposes. See how others do it.

13. Outsmart Lie Detector Tests

Hundreds of thousands of people in this country are wrongfully fired or not hir ed simply because they did not pass the lie detector test even though they've do ne nothing wrong! Read drugless methods to help pass whether you are lying or no t! A valuable tool for any job seeker. Don't be harassed by your employer ever a gain. Tested and proven.

14. Lock-picks & Lock-picking.

Why buy expensive lock-picks & pay for rip-off mail order locksmith courses? W e'll show you how to make your own professional lock-picks. Exact detailed drawi ngs & construction techniques! This is perhaps the easiest to understand course ever published on this hush-hush subject. You won't believe how easy it is to ma ke these tools! We also show you how a basic lock works & how they are picked. T his publication is complete with detailed drawings & illustrations.

15. Guaranteed Unsecured Credit Cards up to $50,000 Regardless of Bad Credit.

We can show you how and where you can obtain unsecured Gold and Platinum credit cards with credits limits up to $50,000 per card! Regardless of your past credit history! You are guaranteed to receive at least 2 unsecured credit cards of up to $20,000 through our exclusive list. The secret to getting the credit you deserve is knowing how and where to look!

* No credit checks! No employment check!
* No credit turndowns! No credit, Bad credit!
* Some with as low as 0% APR for the life of the card!
* Guaranteed approval!

ALL IN ONE!!! PREVIOUSLY SOLD FOR HUNDREDS!!! ORDER NOW!!!

That's 15 products, all for just $19.95 [shipping & handling included].

We accept cash, personal checks, money orders and cashiers checks. You must include a

Primary and secondary E-mail address, as we will be emailing you all the reports as soon as your payment is received.

Print the following form & mail it to:

Info 4 Edu Only
1300 N. Cahuenga Blvd # 362
Los Angeles, CA 90028

Please Print Clearly

Name: _____

Primary Email Address: _____

Secondary Email Address: _____

Make your check payable to: Info 4 Edu Only

DISCLAIMER: Please note that this information is being provided for educational purposes only. The information itself is legal, while the usage of such information may be illegal. We do not advocate unauthorized use or theft of any services. If in doubt, check your local laws and act accordingly.
NOTE: All of the publications are Copyright 2001 by Info 4 Edu Only. We aggressively protect our copyrights and will seek prosecution of any website, web-master, web hosting service or anyone else that is in violation of US & International Copyright Laws.

To be opt out from our future mailing please email optmeoutnow@aol.com with the word remove in the subject line

Example 3

In [22]:
```
# Just run this cell, don't modify it.

print("spam: " + str(train.loc[1092]["spam"]))
print("\nemail:\n" + train.loc[1092]["email"])
```

```
spam: 0

email:
-------------------------------------------------------------------------
 1. iSilo(TM) 3.25 for Palm OS, Pocket PC, and Windows enters beta1
 2. iSilo(TM) 3.25 for Windows CE Handheld PC enters beta1
 3. iSiloX 3.25 for Windows and Mac OS enters beta1
 4. iSiloXC 3.25 for Windows, Linux, FreeBSD, Mac OS X,
    and Solaris enters beta1
 -------------------------------------------------------------------------

 =========================================================================
 1. iSilo(TM) 3.25 for Palm OS and Pocket PC enters beta1
 -------------------------------------------------------------------------
 iSilo(TM) 3.25b1 addresses the following issues from iSilo 3.2:

 - When highlighting content with a non-zero top margin, the highlight
   may be offset vertically in the downward direction.
 - (iSilo for Palm OS) Tapping the application launcher/home silkscreen
   icon while in the password dialog results in a fatal error.
 - (iSilo for Palm OS) Documents in /Palm/Launcher may fail to be added
   to the document list.
 - (iSilo for Palm OS) Text displays cutoff on the NX60/70V.

 iSilo(TM) 3.25b1 adds the following new features to iSilo 3.2:

 - Added support for hyperlink mode for iterating through currently
   visible hyperlinks.
 - Added support for starting on the home page of a document if the
   document was converted with the option for doing so if the document
   date differs from the document date the last time the document was
   opened.
 - (iSilo for Palm OS) For Palm OS 5, added ability to display an
   image at double the image's width and height.
 - (iSilo for Pocket PC) When renaming files, if the original file
   name had a three letter extension, then add it to the new file name
   if it does not end with a period nor have a three letter extension.

 Please note that this is a beta version of the software, and as
 such is intended for testing purposes to help eliminate problems
 and issues before the release.

 Please report any problems or issues you encounter by sending an
 email to email@iSilo.com.  When reporting problems, please include
 all potentially relevant details, including the exact steps used
 to reproduce the problem, OS version, and hardware configuration.

 To download iSilo(TM) 3.25b1 for Palm OS(R), please go to this URL:

     <http://www.iSilo.com/info/beta/iSiloPalmOS.htm>

 To download iSilo 3.25b1 for Pocket PC, please go to this URL:

     <http://www.iSilo.com/info/beta/iSiloPPC.htm>

 To download iSilo 3.25b1 for Windows , please go to this URL:

     <http://www.iSilo.com/info/beta/iSiloW32.htm>
```

```
================================================================================
2. iSilo(TM) 3.25 for Windows CE Handheld PC enters beta1
--------------------------------------------------------------------------------
iSilo(TM) 3.25 will be the first version of iSilo for Windows CE
Handheld PC and has the requirements shown here.

Basic requirements:
- Windows CE Handheld PC Edition version 3.0, Windows CE Handheld PC
  Professional Edition version 3.0, or Windows CE Handheld PC 2000
- Storage memory for application executable file (dependent on
  processor): ARM: 579K, MIPS: 627K
- To determine your processor type go to the Start menu, tap Settings, tap
  Control Panel, double-tap System, and then go to the Device tab.
- 86K storage memory for supporting documents
- 512K free program memory (1MB recommended)

Additional functionality specific requirements:
- Additional memory required for document storage.
- iSiloX required for creating documents utilizing maximum document
  presentation capabilities.

Please note that this is a beta version of the software, and as
such is intended for testing purposes to help eliminate problems
and issues before the release.

Please report any problems or issues you encounter by sending an
email to email@iSilo.com.  When reporting problems, please include
all potentially relevant details, including the exact steps used
to reproduce the problem, OS version, and hardware configuration.

To download iSilo 3.25b1 for Windows CE Handheld PC, please go
to this URL:

    <http://www.iSilo.com/info/beta/iSiloHPC.htm>


================================================================================
3. iSiloX 3.25 for Windows and Mac OS enters beta1
--------------------------------------------------------------------------------
iSiloX 3.25b1 addresses the following issues from iSiloX 3.2:

- Maximum width and height settings for images would not be applied
  to images referenced by an image tag with a width or height
  attribute specification greater than the native width or height
  of the image.
- Within a block with a non-zero text indent, a line following a
  forced break with the <br> tag with markup at the beginning
  of the line would incorrectly be indented.
- If a parent element specifies a font-size using ems, then the
  child element would incorrectly also inherit the specification
  instead of the computed font-size value.  For example, in the
  case where the parent element specified 2em, the actual computed
  value of the child would be 4em (e.g., 2 * 2em = 4em).
- Upon reopening a .ixl file containing a document entry with the
  receive cookies option on, the option would be reset to off.
- In a .ixl file, if an element is specified but has no content,
```

```
  the .ixl file may fail to open.

iSiloX 3.25b1 adds the following features to iSiloX 3.2:

- In the Document properties, added a home page section for specifying
  the document home page number and whether or not to open the
  document on the home page when the document date changes.
- In the Text properties, for preformatted text, added an option
  for specifying whether or not to use a monospace font.
- In the Text properties, for preformatted text, added an option
  for specifying whether to keep all single line breaks, remove all
  single line breaks, or only keep single line breaks if a space or
  tab character follows.

Please note that this is a beta version of the software, and as
such is intended for testing purposes to help eliminate problems
and issues before the release.

Please report any problems or issues you encounter by sending an
email to email@iSilo.com.  When reporting problems, please include
all potentially relevant details, including the exact steps used
to reproduce the problem, OS version, and hardware configuration.

To download iSiloX 3.25b1 for Windows, please go to this URL:

    <http://www.iSiloX.com/info/beta/iSiloXWindows.htm>

To download iSiloX 3.25b1 for Mac OS, please go to this URL:

    <http://www.iSiloX.com/info/beta/iSiloXMac.htm>


==========================================================================
4. iSiloXC 3.25 for Windows, Linux, FreeBSD, Mac OS X,
   and Solaris enter beta1
--------------------------------------------------------------------------
iSiloX 3.25b1 addresses the following issues from iSiloX 3.2:

- Maximum width and height settings for images would not be applied
  to images referenced by an image tag with a width or height
  attribute specification greater than the native width or height
  of the image.
- Within a block with a non-zero text indent, a line following a
  forced break with the <br> tag with markup at the beginning
  of the line would incorrectly be indented.
- If a parent element specifies a font-size using ems, then the
  child element would incorrectly also inherit the specification
  instead of the computed font-size value.  For example, in the
  case where the parent element specified 2em, the actual computed
  value of the child would be 4em (e.g., 2 * 2em = 4em).
- Upon reopening a .ixl file containing a document entry with the
  receive cookies option on, the option would be reset to off.
- In a .ixl file, if an element is specified but has no content,
  the .ixl file may fail to open.

iSiloX 3.25b1 adds the following features to iSiloX 3.2:

- Under <DocumentOptions>, added the <HomePageNumber> and
```

```
  <OpenHomePageOnDateChange> elements for specifying the document
  home page number and whether or not to open the document on the
  home page when the document date changes.
- Under <DocumentOptions>, added the <PreUseMonospaceFont> element
  for preformatted text for specifying whether or not to use a
  monospace font.
- Under <DocumentOptions>, added the <PreSingleLineBreaks> element
  for preformatted text for specifying whether to keep all single
  line breaks, remove all single line breaks, or only keep single
  line breaks if a space or tab character follows. See .ixl file
  format for more information.

Please note that this is a beta version of the software, and as
such is intended for testing purposes to help eliminate problems
and issues before the release.

Please report any problems or issues you encounter by sending an
email to email@iSilo.com.  When reporting problems, please include
all potentially relevant details, including the exact steps used
to reproduce the problem, OS version, and hardware configuration.

To download iSiloXC 3.25b1 for Windows, please go to this URL:

    <http://www.iSiloX.com/info/beta/iSiloXCWindows.htm>

To download iSiloXC 3.25b1 for Linux, please go to this URL:

    <http://www.iSiloX.com/info/beta/iSiloXC386.htm>

To download iSiloXC 3.25b1 for FreeBSD, please go to this URL:

    <http://www.iSiloX.com/info/beta/iSiloXCBSD.htm>

To download iSiloXC 3.25b1 for Mac OS X, please go to this URL:

    <http://www.iSiloX.com/info/beta/iSiloXCOSX.htm>

To download iSiloXC 3.25b1 for Solaris, please go to this URL:

    <http://www.iSiloX.com/info/beta/iSiloXCSOL.htm>


------------------------------------------------------------------
You have received this message because you subscribed to the
iSilo(TM) mailing list.  If you would like to be removed from
this list, please send an email with the subject "unsubscribe"
to: <list@iSilo.com>.

Please note that this email address only handles subscriptions
and unsubscriptions of the mailing list.  Any other messages
sent to this email address are automatically deleted.

------------------------------------------------------------------
iSilo(TM)            http://www.iSilo.com            email@iSilo.com
```

Question 6a

Pick at least **one** of the emails provided above to comment on. How would you classify the email (e.g., spam or ham), and does this align with the classification provided in the training data? What could be a reason someone would disagree with *your* classification of the email? In 2-3 sentences, explain your perspective and potential reasons for disagreement.

The first email in example 1 from MR. Mabo Loko which discusses a "urgent business proposal" involving a inheritence scam. The classification in the training data is also spam so this does align. While someone might disagree with me it is rather unlikely. While it could be a genuine business proposal and they could be trying to use you to make some sort of financial gain, the email is clearly a phishing scam. The email promises large sums of money and also requests us as the user to provide sensative personal informatnion. Thus, aside from the formal tone of the spam email, it is far more probable to be a scam and phishing attempt and the program should properly identify the email as such.

Question 6b

As data scientists, we sometimes take the data to be a fixed “ground truth,” establishing the “correct” classification of emails. However, as you might have seen above, some emails can be ambiguous; people may disagree about whether an email is actually spam or ham. How does the ambiguity in our labeled data (spam or ham) affect our understanding of the model's predictions and the way we measure/evaluate our model's performance?

The abiguity of laveled data in data sicence affects our undersanding of models predicitons because is introduced a bias. THe bias is that we often times believe the information is the ground truth, and this train our models on that data. Since we train and evaluate our models on said data, there can sometiems be discrepancies between the models predictions and actual human judgement. To be more clear, the models may be showcasing high accuracyl, however, the models performance may be underestimating or overestimating the subjective accuracy in a given situation due to the interpretation of the data. Thus, as data scientists our jobs start as grund zero, making sure our labeled data is as clear from potential biases as possible and thus mitigating the effect of these biases in our end result increasing the models TRUE effectiveness.

As a data scientist, we encourage you to think more critically about your data before establishing it as the "ground truth." Whenever you're working on a specific problem, ask yourself:

1. Who “made” the data? Think about all the stages from when it was first generated, collected, and labeled before it ended up in a CSV file.
2. What assumptions and biases are inherently present in the data?
3. And finally, how does all this affect how you interpret your model’s performance?

Question 7

In Question 6, we explored the instability present in the “ground truth” and how this affects our evaluation of our model. Now, let's start thinking about your model's interpretability and what that means more broadly for an email classification task. A model is considered interpretable if humans can easily understand the reasoning behind its predictions and classifications.

Question 7a

First, let's see if we can understand how our choice of features relates to how a particular email is classified.

Part i

Let’s take a look at the `simple_model` we provided you earlier that uses 5 features. We have provided the code below for ease of reference. You will examine how a particular feature influences how an email is classified.

In [23]:
```
# Simple model introduced at the start of this notebook. Just pay attention to t
some_words = ['drug', 'bank', 'prescription', 'memo', 'private']

X_train = words_in_texts(some_words, train['email'])
Y_train = np.array(train['spam'])

simple_model = LogisticRegression()
simple_model.fit(X_train, Y_train);
```

Pick an email from the training set and assign its index to `email_idx`. Then, find **one** feature used in `simple_model` such that **removing** it changes how that email is classified. Assign this feature to `feature_to_remove`.

In [24]:
```
# # Use this cell for scratch work when determining `email_idx`
# for email_idx in range(len(train)):
#     prob_spam = simple_model.predict_proba(X_train)[:, 1]
#     initial_prob = prob_spam[email_idx]
#     initial_class = "spam" if np.round(initial_prob) else "ham"

#     for feature_to_remove in some_words:
#         changed_words = some_words.copy()
#         changed_words.remove(feature_to_remove)

#         X_changed = words_in_texts(changed_words, train['email'])
#         changed_model = LogisticRegression()
#         changed_model.fit(X_changed, Y_train)
```

```
#         changed_prob = changed_model.predict_proba(X_changed)[:, 1][email_idx]
#         changed_class = "spam" if np.round(changed_prob) else "ham"

#         if changed_class != initial_class:
#             print(f"Email index {email_idx} worked with feature '{feature_to_r
#             print(f"Initially classified as {initial_class} (Probability: {np.
#             print(f"Now classified as {changed_class} (Probability: {np.round(
#             break
#     else:
#         continue
#     break
```

```
Email index 50 worked with feature 'bank' removed.
Initially classified as spam (Probability: 54.29%)
Now classified as ham (Probability: 25.36%)
```

In [25]:
```
email_idx = 50

prob_spam = simple_model.predict_proba(X_train)[:, 1]
initial_prob = prob_spam[email_idx]
initial_class = "spam" if np.round(initial_prob) else "ham"
print(f"\nPredicted probability of being spam: {np.round(initial_prob*100, 2)}%"
print("\nEmail:\n" + train.loc[email_idx]["email"])
```

```
Predicted probability of being spam: 54.29%

Email:
On 06 September 2002, Tim Peters said:
 > > Note that header names are case insensitive, so this one's no
 > > different than "MIME-Version:".  Similarly other headers in your list.
 >
 > Ignoring case here may or may not help; that's for experiment to decide.
 > It's plausible that case is significant, if, e.g., a particular spam mailing
 > package generates unusual case, or a particular clueless spammer
 > misconfigures his package.

 Case of headers is definitely helpful.  SpamAssassin has a rule for it
 -- if you have headers like "DATE" or "SUBJECT", you get a few more
 points.

         Greg
 --
 Greg Ward <gward@python.net>                          http://www.gerg.ca/
 God is omnipotent, omniscient, and omnibenevolent
 ---it says so right here on the label.
```

In [26]:
```
feature_to_remove = 'bank'

changed_words = some_words.copy()
changed_words.remove(feature_to_remove)

changed_model = LogisticRegression()
X_changed = words_in_texts(changed_words, train['email'])
y = train['spam']
changed_model.fit(X_changed, y)
changed_prob = changed_model.predict_proba(X_changed[[email_idx]])[:,1][0]
changed_class = "spam" if np.round(changed_prob) else "ham"
```

```
print(f"Initially classified as {initial_class} (Probability: {np.round(initial_
print(f"Now classified as {changed_class} (Probability: {np.round(changed_prob*1
```

```
Initially classified as spam (Probability: 54.29%)
Now classified as ham (Probability: 25.36%)
```

In [27]:
```
grader.check("q7ai")
```

Out[27]:

q7ai passed! 🙌

Part ii

In 2-3 sentences, explain why you think the feature you chose to remove changed how your email was classified.

I believe the reason the removal of bank changed the classification of the email was because the word "bank" more than likely plays a significant role in phishing scams. If we look at question 6 we can even clearly see that banks, financial institutions, and other areas of interest play a large role in phishing scams. This can be troublesome when developing models that use words as classification features as many users will get emails about financial information which can be very urgent if those items were sorted as spam incorrectly it could be disasterous to the user. Thus, when we removed bank we could see that the models confidence in the email being ham dropped significantly attributing to the fact that there was very likely less atrritubtes i nthe email that attributed it to being spam.

Question 7b

Now, let's say that instead of working with a small model containing 50-100 features, you're working with a much larger, more accurate model containing 1000 features.

Part i

In this context, do you think you could easily find a feature that could change an email's classification as you did in part a)? Why or why not?

No, the reason why is because as the complexity of features increases the weight that a specific feature carries is diluted. The features importance in the overall model thus would have less of a down stream affect than a model that contains 50-100 features. To further reiterate, as the number of features n increases to a larger number >> the overall prediction contribution of a specific feature is smaller in proportion to the entire model. In contrast, as the number of features n decreases to a smaller number << the overall prediction contribtuion of a specific feature increases and thus the removal of a feature can be far more significant that a more complex model running 100s if not 1000s of features.

Part ii

Would you expect this new model to be more or less interpretable than `simple_model`?

Note: A model is considered interpretable if you can easily understand the reasoning behind its predictions and classifications. For example, the model we saw in part a), `simple_model`, is considered interpretable as we can identify which features contribute to an email's classification.

A model with a large variety of features can have pros and cons. While it may be more accurate in some cases, it would certainly be less interpretable than the simple_model. This is because as models increase in complexity the understanding of a certain feature's over all specific contribution is less interpretable. Thus, for this instance of understanding models contribution to email classification it would be easier for us to visualize the effect of a specific models removal and effect on accuracy or confidence than if a model has 1000 features.

Question 7c

Now, imagine you're a data scientist at Meta, developing a text classification model to decide whether to remove certain posts / comments on Facebook. In particular, you're primarily working on moderating the following categories of content:

- Hate speech
- Misinformation
- Violence and incitement

Pick one of these types of content to focus on (or if you have another type you'd like to focus on, feel free to comment on that!). **What content would fall under the category you've chosen?** Refer to Facebook's Community Standards, which outline what is and isn't allowed on Facebook.

WHile hate speech or violence and incitement are a bit easier to classify I thought misinformation would be a bit more interesting to select when discussing a moderation model as the development of such a model can be rather interesting and in my opinion more complex as certain posts may discuss ideas not directly stated publics or be common knowledge thus, how could a post be accurately sorted as a misinformation post. Looking at Facebooks Community Standards, content that falls under misinformation would be false or intentionally misleading information, as well as the use of misleading information to intentionally or unintelltionally decieve others. Examples of such would be fake news, claims about public health (such as new viruses or conspircy weapons to insight public hysteria), incorrect facts about events, or manipulated / artifically generated visual content to decieve views (such as deep fakes of popular public figures to promote a crypto currency or some other form of business model.)

Question 7d

What are the stakes of misclassifying a post in the context of a social media platform? Comment on what a false positive and false negative means for the category of content you've chosen (hate speech, misinformation, or violence and incitement).

False Posititive: A false posititive is when a post is classified as misinformation but is accurate information. A post in the context of misinformation would mean that we are labeling or sorting TRUE posts as misinformation. This thus infridges on the right of all americans to free speech and furthermore decrease the pipe line of accurate information thus, decreasing the avaibility of knowledge to the public and thus negatively impacting the trust users have in metas platform. False Negative: A false negative is when a post is classified as true and thus not restricted but actually contains misinformation. The downstream affects of this would be users beliveing information that propogates onto the platform. An example of extremely disaterous information not being accurately sorted would be posts about trumps assasination attempt. Many users created posts discribing the event as a planned hit by the democratic party. Of course such an egregious claim is false, however, if any posts were not sensored it could cause many users to become radicalized against the corresponding party and could potentially negatively impact society. THis could lead to acts of violence, terrorism, and other acts. Thus, it is important as developers and data scientists that we decrease the rate of false positives and negatives as much as possible to maintain both trust and reliability on the platform but also maintain national security and safety.

Question 7e

As a data scientist, why might having an interpretable model be useful when moderating content online?

The more interpretable a model the greater the ability of said model to allow moderators and developers to understand why specific posts were flagged. This also helps for users to understand why their posts were flagged in the first place. The main caviot with a model like this on online platforms isthat you as the platform can not simply take down a post but not express the reason why a specific post was moderated in the first place. Thus, it is useful and morally important to have interpretable models so that as a platform we can both align with community guidelines but also ethical standards. Furthermore, the more interpretable a model the more likely it is to aid in fixing biases and errors in the model leading to a more effective model and a more fair and enjoyable platform.

As you explored throughout this question, interpretability is incredibly important. However, it is equally important to note that interpretability on its own isn’t a fix to all the problems that may arise when moderating content or when building a model more generally. As we touched on in Project A2, these models don’t operate in a vacuum; they exist in a wider sociotechnical system. Everything from the data used to train these models to the metrics we choose to evaluate our models builds on that notion.

Posey congratulates you for finishing Project B2!

 No description has been provided for this image  No description has been provided for this image

Course Content Feedback

If you have any feedback about this assignment or about any of our other weekly assignments, lectures, or discussions, please fill out the Course Content Feedback Form. Your input is valuable in helping us improve the quality and relevance of our content to better meet your needs and expectations!

Submission Instructions

Below, you will see a cell. Running this cell will automatically generate a zip file with your autograded answers. Once you submit this file to the Project B2 Coding assignment on Gradescope, Gradescope will automatically submit a PDF file with your written answers to the Project B2 Written assignment. If you run into any issues when running this cell, feel free to check this section in the Data 100 Debugging Guide.

If there are issues with automatically generating the PDF, you can try downloading the notebook as a PDF by clicking on `File -> Save and Export Notebook As... -> PDF`. If that doesn't work either, you can manually take screenshots of your answers to the manually graded questions and submit those.

Please make sure you submit the following to the right assignments:

- **Project B2 Coding:** Submit the zip file generated by using the `grader.export()` cell provided below.
- **Project B2 Written:** Gradescope will automatically submit the PDF from the zip file submitted earlier. You do not need to submit anything to this assignment yourself, but *please check that the submission went through properly and that all plots rendered correctly*.
- **Project B2 Test Set Predictions:** Submit the CSV file generated in `q3b`.

You are responsible for ensuring your submission follows our requirements and that everything was generated and submitted correctly. We will not be granting regrade requests nor extensions to submissions that don't follow instructions. If you encounter any difficulties with submission, please don't hesitate to reach out to staff prior to the deadline.

Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit. **Please save before exporting!**

In [28]:
```
# Save your notebook first, then run this cell to export your submission.
grader.export(run_tests=True)
```

```
Running your submission against local test cases...


Your submission received the following results when run against available test c
ases:

    q3a results: All test cases passed!

    q3b results: All test cases passed!

    q7ai results: All test cases passed!
```

Your submission has been exported. Click here to download the zip file.