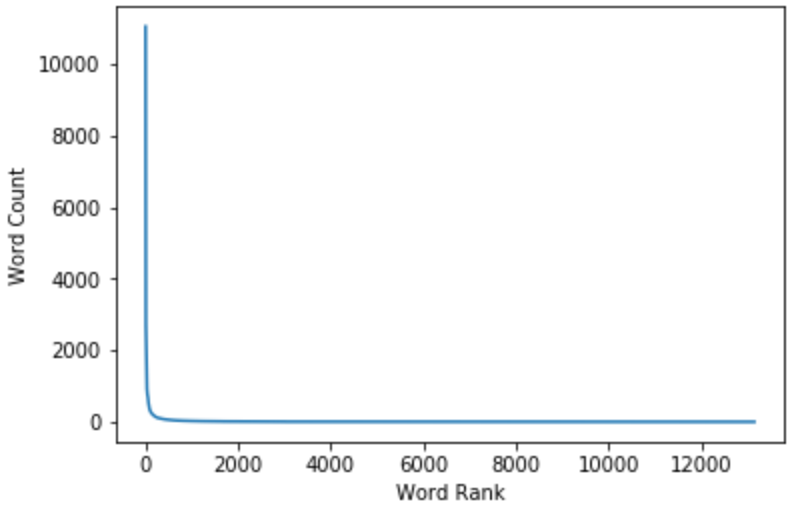
Page 1 Distribution graph (5 points)

Show the distribution graph of words counts vs word rank.



Page 2 Identify the stop words (5 points)

List the stop words you choose as well as the frequency threshold.

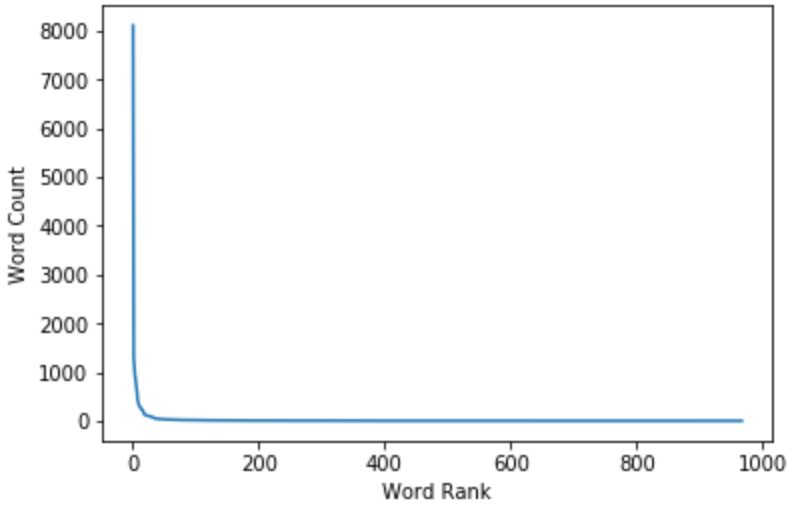
stopwords = ['the', 'and', 'to', 'was', 'it', 'of', 'for', 'in', 'my', 'is',
'that', 'they', 'this', 'we', 'you', 'with', 'on', 'not', 'have',
'but', 'had', 'me', 'at', 'so', 'were', 'are', 'be', 'place',
'food', 'there', 'as', 'he', 'if', 'all', 'when', 'out', 'would',
'get', 'our', 'she', 'back', 'one', 'up', 'time',
'from', 'very', 'an', 'just', 'their', 'here', 'will',
'go', 'about', 'them', 'or', 'can',
'what', 'your', 'us', 'been', 'do', 'because', 'only',
'don', 'even', 'after', 'by', 'which', 'did', 'got', 'said',
'more', 'her', 'really', 'told', 'also', 'could', 'some', 'other',
'then', 'went', 'over', 've', 'has', 'service']

max_df = 0.99
min_df = 0.01

We chose words that do not convey positive or negative connotation as stop words.

Page 3 Distribution graph again (5 points)

After choosing the stop words, show the distribution graph of words counts vs word rank.



Page 4 Code snippets (15 points)

Show the snippet of your code that you convert all the reviews into bag-of-words formulation using your chosen stop words and your code for nearest-neighbours with cos-distance.

```
# PART 1
# Generate bag-of-word
vectorizer = CountVectorizer()
bow_vec = vectorizer.fit_transform(data.text)
words = vectorizer.get_feature_names()
x = bow_vec.toarray()
# Create numpy array with word and the corresponding word count, then sort the word count
freq = np.sum(x,axis=0)
dictionary = dict(zip(words, freq))
ranked_dict = np.array(sorted(dictionary.items(), key=itemgetter(1), reverse=True))
# Generate bag-of-word from stop words, maxfq, and min fq
stopwords = ['the', 'and', 'to', 'was', 'it', 'of', 'for', 'in', 'my', 'is',
       'that', 'they', 'this', 'we', 'you', 'with', 'on', 'not', 'have',
       'but', 'had', 'me', 'at', 'so', 'were', 'are', 'be', 'place',
       'food', 'there', 'as', 'he', 'if', 'all', 'when', 'out', 'would',
       'get', 'our', 'she', 'back', 'one', 'up', 'time',
       'from', 'very', 'an', 'just', 'their', 'here', 'will',
       'go', 'about', 'them', 'or', 'can',
       'what', 'your', 'us', 'been', 'do', 'because', 'only',
       'don', 'even', 'after', 'by', 'which', 'did', 'got', 'said',
       'more', 'her', 'really', 'told', 'also', 'could', 'some', 'other',
       'then', 'went', 'over', 've', 'has', 'service']
vectorizer2 = CountVectorizer(stop_words = stopwords, max_df = 0.99, min_df = 0.01)
bow_vec2 = vectorizer2.fit_transform(data.text)
words2 = vectorizer2.get_feature_names()
x2 = bow_vec2.toarray()
# Create numpy array with word and the corresponding word count, then sort the word count
freq2 = np.sum(x,axis=0)
dictionary2 = dict(zip(words2, freq2))
ranked_dict2 = np.array(sorted(dictionary2.items(), key=itemgetter(1), reverse=True))

# PART 2
# Take in query and find the closest 5 reviews
query = ['horrible customer service']
bow_q = vectorizer2.transform(query)
x_query = bow_q.toarray()
# Use nearest neighbors to find 5 most relevant reviews using cosine distance metric
nn = NearestNeighbors(n_neighbors = 5, metric = 'cosine', n_jobs = -1)
nn.fit(x2)
neighs = nn.kneighbors(x_query, return_distance=True)
nn_reviews = data.loc[neighs[1][0],:]
nn_cosdist = neighs[0][0]
print(nn_reviews, '\n', nn_cosdist)
# Print all cosine distances
nn2 = NearestNeighbors(n_neighbors = len(x2), metric = 'cosine', n_jobs = -1)
nn2.fit(x2)
neighs2 = nn2.kneighbors(x_query, return_distance=True)
nn_cosdist2 = neighs2[0][0]
print(nn_cosdist2)
```

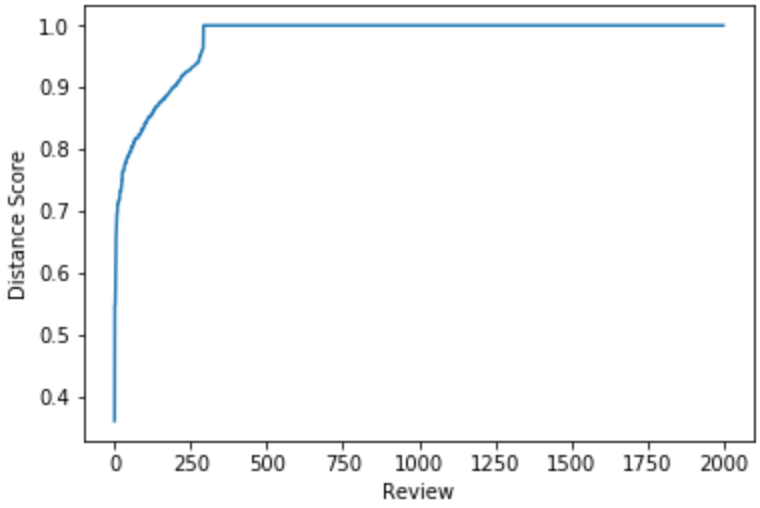
Page 5 Reviews with score (10 points)

Show the original reviews with the distance scores

Stars	Text	Distance Scores
1	Rogers ... 1) is over priced 2) have horrible customer service 3) faulty and incorrect billing 4) poor customer service 5) not enough options 6) never arrive for an appointment	0.36039785
1	Horrible service, horrible customer service, and horrible quality of service! Do not waste your time or money using this company for your pool needs. Dan (602)363-8267 broke my pool filtration system and left it in a nonworking condition. He will not repair the issue he caused, and told me to go somewhere else. Save yourself the hassle, there are plenty of other quality pool companies out there. Take care!	0.54708919
1	Service was horrible came with a major attitude. Payed 30 for lasagna and was no where worth it. Won't ever be going back and will NEVER recommend this place. was treated absolutely horrible. Horrible.	0.54773298
1	HORRIBLE HORRIBLE HORRIBLE!!! AVOID AT ALL COSTS!!! I had some work done at Swing Shift Auto and I was helped by Keith. He was very arrogant and had little time for me. I just needed new brake discs and pads. I was overcharged, the repairs took TWO DAYS, and when I got home i noticed that the discs had NOT been replaced, only the pads!!!! TOTAL RIPOFFF!!! NEVER GO HERE, PLEASE!!!	0.57573593
1	Horrible. Absolutely horrible. Seems like they are completely insensitive with what the customer is calling about. The Girl I spoke with on the phone didn't care for my situation, and had no sense of urgency for our situation even to the point where she just said "we can schedule you in 2 weeks". I said we have an active leak and her response "find someone else". I was so nice and trying to be agreeable, and she just didn't want to provide any customer service or even find out if someone could come sooner. We spent over 2K about 3 months ago which seemed incredibly overpriced, we didn't argue just paid. She didn't care that I wanted to give them repeat business and just turned me away. Not a nice feeling when you are trying to be a returning customer and have water leaking in your home. I had heard of this tone from this company, but ignored it as I wanted to give them the benefit of the doubt. Lesson learned.	0.63915608

Page 6 Query results (10 points)

Show your document results and explain the reasons that you choose them.



We chose to show the first 55 reviews that have distance scores less than or equal to 0.8 because the distance scores start to climb less steeply at around 0.8, indicating the boundary of relevant reviews.

Page 7 Accuracy with threshold 0.5 (10 points)

Show your code for creating classifier. Report the accuracy on train and test dataset with threshold 0.5.

```
# Logistic Regression on both test and train sets
clf = LogisticRegression().fit(X_train, y_train)
test_results = clf.predict(X_test)
acc_test = (test_results == y_test).sum()/y_test.size

train_results = clf.predict(X_train)
acc_train = (train_results == y_train).sum()/y_train.size
print('Test accuracy: ',acc_test,'\nTrain accuracy: ', acc_train)
```

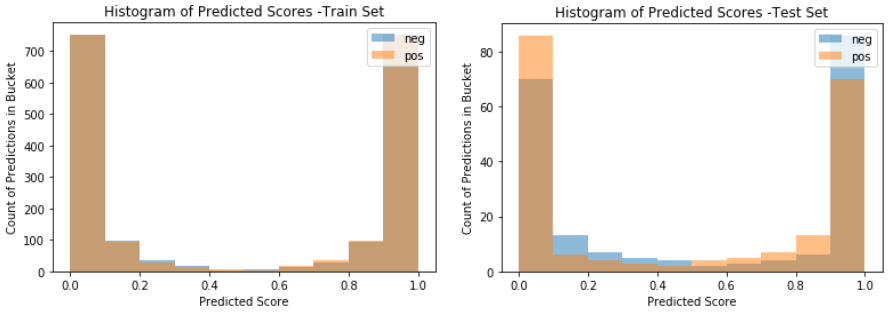
Test accuracy: 0.93

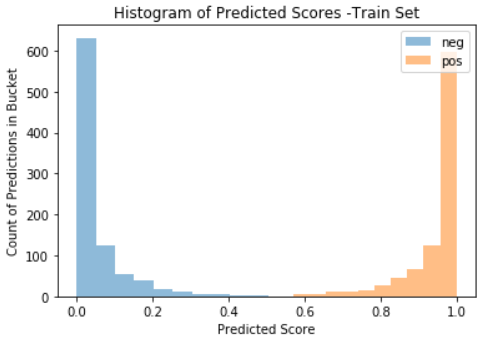
Train accuracy: 0.9972222222222222

Page 8 Predicted scores (10 points)

Show your code for plotting predicted scores and show the figure.

```
# Modify threshold(theta)

test_prob = clf.predict_proba(X_test)
train_prob = clf.predict_proba(X_train)

train_prob_neg = train_prob[:,0]
train_prob_pos = train_prob[:,1]

test_prob_neg = test_prob[:,0]
test_prob_pos = test_prob[:,1]

theta = 0.56

def generateHist(train_prob_neg, train_prob_pos):
    plt.hist(train_prob_neg, alpha=0.5, label='neg')
    plt.hist(train_prob_pos, alpha=0.5, label='pos')
    plt.legend(loc = 'upper right')
    plt.ylabel("Count of Predictions in Bucket")
    plt.xlabel("Predicted Score")
    plt.title("Histogram of Predicted Scores")
    plt.show()

generateHist(train_prob_neg, train_prob_pos)
generateHist(test_prob_neg, test_prob_pos)

train_prob_neg_t = train_prob_pos[train_prob_pos < theta]
train_prob_pos_t = train_prob_pos[train_prob_pos >= theta]

generateHist(train_prob_neg_t, train_prob_pos_t)

print('Theta: ', theta)
acc_test_t = (y_test==5)==(test_prob_pos >= theta)
print('Test accuracy: ', acc_test_t.sum()/acc_test_t.size)

acc_train_t = (y_train==5)==(train_prob_pos >= theta)
print('Train accuracy: ',acc_train_t.sum()/acc_train_t.size)
```





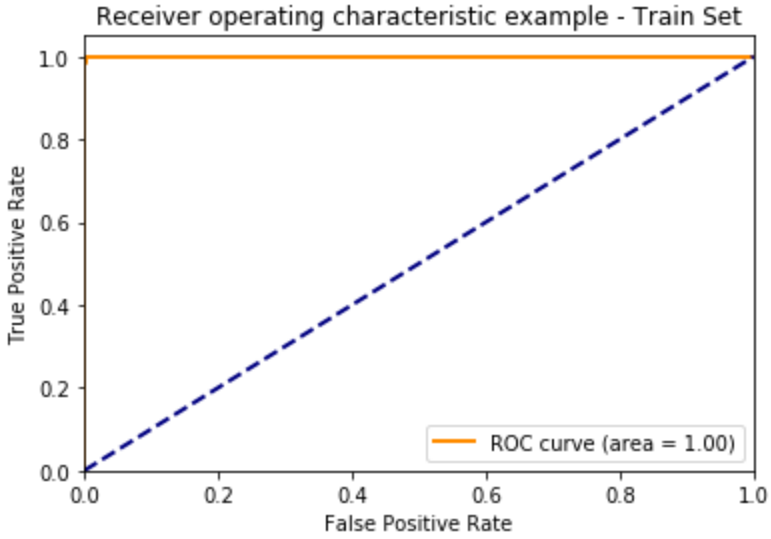
Page 9 Accuracy again and curve (20 points)

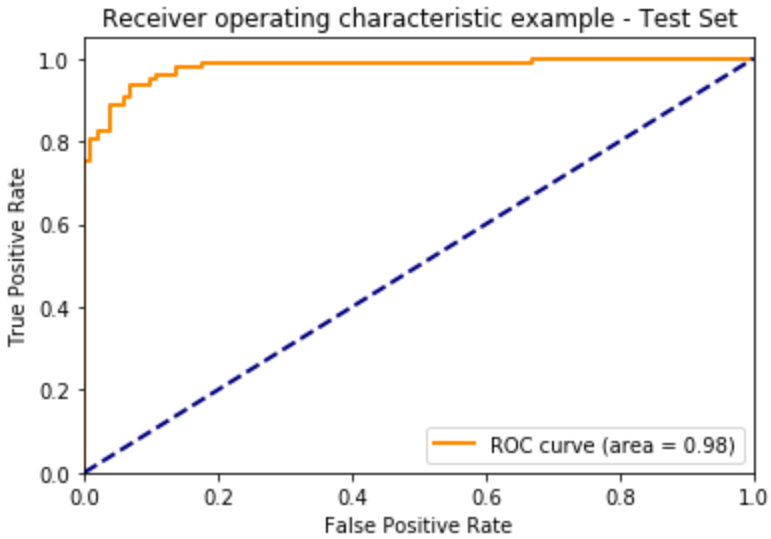
Report the accuracy on train and test dataset with a different threshold. Explain why you choose that threshold.

Plot the ROC curve.

Theta: 0.56
Test accuracy: 0.935
Train accuracy: 0.9966666666666667

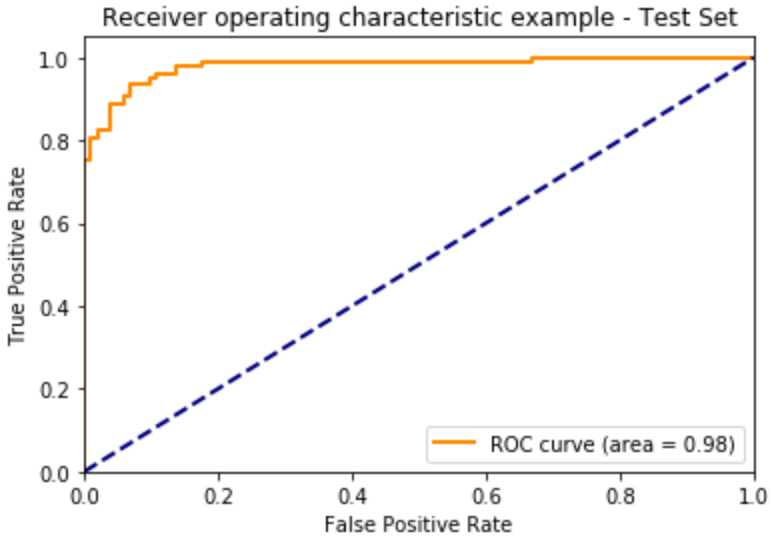
We chose this threshold because it yields highest test accuracy while maintaining high train accuracy. Train accuracy is already very high; hence, a slight trade off between test and train accuracy is acceptable.





Page 10 Best threshold (10 points)

Choose the threshold that minimizes false positives while maximizing true positives. Explain your reason.



The threshold that yields high TPR while keeps FPR low is 0.126. The TPR is 0.990 while FPR is 0.175. If the threshold decreases further, the FPR will dramatically increase while TPR will not change as much.