In [145]:

```
#HW2A ALEC CHEN

import matplotlib.pyplot as plt
import scipy as sp
import pandas as pd
import math
```

In [146]:

```
df = pd.read_csv("sample_data/spam.csv", sep=',', encoding="latin1")
df.drop(df.columns[[2,3,4]],axis=1,inplace=True)
```

In [147]:

```
#removing extra columns, run once
df = df.drop(labels=0, axis=0)
```

In [148]:

```
df = df.rename(columns={"v1": "label", "v2": "message"})
```

In [149]:

```
spam_total = 0

spam_word_counts = {}
for message in df[df["label"] == "spam"]["message"]:
    for word in message.split():
        if word in spam_word_counts:
            spam_word_counts[word] += 1
            spam_total += 1
        else:
            spam_word_counts[word] = 1

# Create a dictionary of word counts for ham messages
ham_total = 0

ham_word_counts = {}
for message in df[df["label"] == "ham"]["message"]:
    for word in message.split():
        if word in ham_word_counts:
            ham_word_counts[word] += 1
            ham_total += 1
        else:
            ham_word_counts[word] = 1

normalized_spam = spam_word_counts.copy()
normalized_ham = ham_word_counts.copy()

for word, index in enumerate(normalized_spam):
  normalized_spam[index] /= spam_total
for word, index in enumerate(ham_word_counts):
  normalized_ham[index] /= ham_total
```

In [150]:

```
# Create a function to score new text messages
def score_message(message):
    score = 0
    for word in message.split():
        if word in normalized_spam and word in normalized_ham:
            # Estimate P(spam)/P(ham) for the word
            ratio = min(max(math.log(2, normalized_spam[word] / normalized_ham[word]) +
math.log(2, spam_total/ham_total), -4.3), 4.3)
            # Use data regularization to cap the maximum absolute value at 20
            score += ratio
```

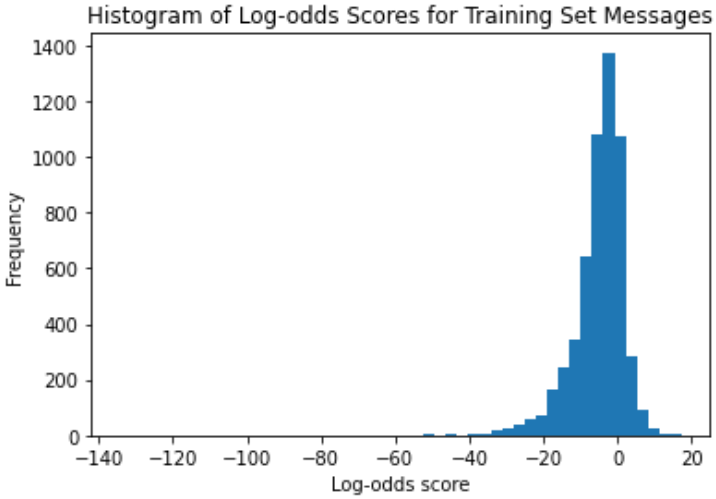
```
    return score
```

In [151]:

```
# Score all messages in the corpus
scores = []
for message in df["message"]:
    scores.append(score_message(message))
```

In [152]:

```
# Plot a histogram of log-odds scores for all messages in the training set
plt.hist(scores, bins=50)
plt.xlabel("Log-odds score")
plt.ylabel("Frequency")
plt.title("Histogram of Log-odds Scores for Training Set Messages")
plt.show()
```



In [153]:

```
#Load American text-message corpus

corpus = pd.read_csv("sample_data/test.txt", sep='\t')
corpus.drop(corpus.columns[[0,1,3]],axis=1,inplace=True)
corpus

c_incoming = corpus[corpus['type'] == 'Incoming']
c_outgoing = corpus[corpus['type'] == 'Outgoing']
```

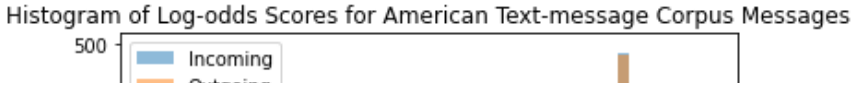
In [154]:

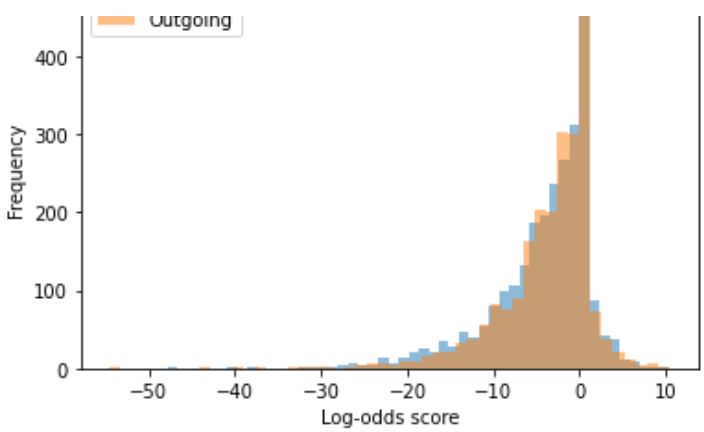
```
# Score all messages in the American text-message corpus
c_incoming_scores = []
for message in c_incoming["message_body"]:
    c_incoming_scores.append(score_message(message))

c_outgoing_scores = []
for message in c_outgoing["message_body"]:
    c_outgoing_scores.append(score_message(message))
```

In [155]:

```
# Plot second histogram
plt.hist(c_incoming_scores, bins=50, alpha = .5, label = 'Incoming')
plt.hist(c_outgoing_scores, bins=50, alpha = .5, label = 'Outgoing')
plt.legend(loc="upper left")
plt.xlabel("Log-odds score")
plt.ylabel("Frequency")
plt.title("Histogram of Log-odds Scores for American Text-message Corpus Messages")
plt.show()
```




Outgoing
400
300
200
100
0
Frequency
−50
−40
−30
−20
−10
0
10
Log-odds score