# CSC411: Assignment 3

Due on Monday, March 19, 2018

**Yian Wu, Zhou Quan**

March 19, 2018

# Part 1

**Describe the datasets.**
**You will be predicting whether a headline is real or fake news from words that appear in the headline. Is that feasible? Give 3 examples of specific keywords that may be useful, together with statistics on how often they appear in real and fake headlines.**

All words in the datasets are in lower cases.

| 3 Examples with biggest difference of existence between fake and real | | | |
|---|---|---|---|
| Word | difference of existence between real file and fake file | probability of existence in real file | probability of existence in fake file |
| donald | 0.245076853696 | 0.42073170731707316 | 0.17565485362095531 |
| time | 8.22090270209e-05 | 0.007621951219512195 | 0.007704160246533128 |
| economy | 3.28836108084e-05 | 0.003048780487804878 | 0.0030816640986132513 |

For these three words in the table above, we think they might be the most useful words to distinguish whether a headline is real or fake news.
With the probabilities of their existence in real headline file and fake headline file, this 3 words have some what biggest differences between two files.

## Part 2

**Implement the Naive Bayes algorithm for predicting whether a headline is real or fake. Tune the parameters of the prior (called m and p̂ on slide 23 of the Generative Classifiers lecture) using the validation set. Report how you did it, and the result. Report the performance on the training and the test sets that you obtain. Note that computing products of many small numbers leads to underflow. Use the fact that**

$$a_1 a_2 ... a_n = exp(\log a_1 + \log a_2 + ... + \log a_n)$$

**In your report, explain how you used that fact.**

We've tried different range of m and p̂. We set the range of m values as 1 to 4 with 0.5 each step and set the range of p̂ values as 0.1 to 0.7 with step size of 0.1.

For each m and p̂, We calculated the performance of validation set, and we find the highest performance of validation set, then we record the corresponding m and p̂ value

The optimized m and p̂ are 1.0 and 0.3 respectively.

training set performance: 0.941794310722
valid set performace: 0.769387755102
test set performance: 0.775967413442

In Naive Bayes functino, We calculated $P(word_i|class)$, we need to calculate $P(word_1|class) \cdot P(word_2|class) \cdot ... \cdot P(word_k|class)$, Thus we calculated $exp(\log P(word_1|class) + \log P(word_2|class) + ... + \log P(word_k|class))$ instead to avoid underflow.

## Part 3

**(a)** We use conditional probability:
$P(class|word) = \frac{P(word|class) \times P(class)}{P(words)}$, $P(class|not\_word) = \frac{P(not\_word|class) \times P(class)}{P(not\_words)}$

(1) To calculate the words whose presence most strongly predicts that the class(real or fake). We need to know the probability of $P(class|word)$
$P(class|word) = \frac{P(word|class) \times P(class)}{P(word)}$
We have already calculated $P(word|class)$ in naive_bayes function which is $\frac{count(word, class) + mp\_hat}{count(class) + m}$
$P(class) = \frac{count(class)}{count(totalclass)}$
$P(word) = P(word|fake)P(fake) + P(word|real)P(real)$
Thus, we can obtain the value of $P(class|word)$

(2) If we want to calculate the words whose absence most strongly predicts that the class(real or fake). We need to know the probability of $P(class|not\_word)$
$P(class|not\_word) = \frac{P(not\_word|class) \times P(class)}{P(not\_word)}$ where $P(not\_word|class) = 1 - P(word|class)$, and We have already calculated $P(word|class)$ in naive_bayes function.
$P(class) = \frac{count(class)}{count(totalclass)}$

$P(not\_word) = P(not\_word|fake)P(fake) + P(not\_word|real)P(real)$
$= (1 - P(word|fake))P(fake) + (1 - P(word|real))P(real)$

The influence of the presence of words are larger than the absence of words on predicting whether the headline is real or fake, as the list shows below.

```
10 words whose presence most strongly predicts that the news is real.
the  1  largest P(real|word) word: (’ban’, 0.9947924428727917)
the  2  largest P(real|word) word: (’korea’, 0.9942974275137846)
the  3  largest P(real|word) word: (’travel’, 0.9927895341215152)
the  4  largest P(real|word) word: (’turnbull’, 0.9922291345665467)
the  5  largest P(real|word) word: (’australia’, 0.9891320454737184)
the  6  largest P(real|word) word: (’climate’, 0.9872900157748868)
the  7  largest P(real|word) word: (’paris’, 0.9819303693618048)
the  8  largest P(real|word) word: (’refugee’, 0.9807720563684165)
the  9  largest P(real|word) word: (’trumps’, 0.9783780058273438)
the  10  largest P(real|word) word: (’tpp’, 0.9761939582215442)

10 words whose absence most strongly predicts that the news is real.
the  1  largest P(real|not word) word: (’trump’, 0.9420631360898537)
the  2  largest P(real|not word) word: (’the’, 0.6567525090158698)
the  3  largest P(real|not word) word: (’hillary’, 0.6295290124030137)
the  4  largest P(real|not word) word: (’a’, 0.6284243693309028)
the  5  largest P(real|not word) word: (’to’, 0.6271296907700552)
the  6  largest P(real|not word) word: (’and’, 0.6233912278808975)
the  7  largest P(real|not word) word: (’of’, 0.6223593722832202)
the  8  largest P(real|not word) word: (’is’, 0.6216530386474928)
the  9  largest P(real|not word) word: (’for’, 0.6199624082324078)
the  10  largest P(real|not word) word: (’in’, 0.6189276820403177)

10 words whose presence most strongly predicts that the news is fake.
the  1  largest P(fake|word) word: (’breaking’, 0.9829520350032682)
the  2  largest P(fake|word) word: (’u’, 0.9794490392820592)
the  3  largest P(fake|word) word: (’3’, 0.9794490392820592)
the  4  largest P(fake|word) word: (’soros’, 0.9779379442236888)
the  5  largest P(fake|word) word: (’m’, 0.9761869936625269)
the  6  largest P(fake|word) word: (’woman’, 0.9741341562433965)
the  7  largest P(fake|word) word: (’secret’, 0.9687454640256747)
the  8  largest P(fake|word) word: (’7’, 0.9687454640256747)
the  9  largest P(fake|word) word: (’steal’, 0.9687454640256747)
the  10  largest P(fake|word) word: (’reporter’, 0.9687454640256747)

10 words whose absence most strongly predicts that the news is fake.
the  1  largest P(fake|not word) word: (’donald’, 0.4846670834346303)
the  2  largest P(fake|not word) word: (’trumps’, 0.4242981100949487)
the  3  largest P(fake|not word) word: (’us’, 0.41866658711632077)
the  4  largest P(fake|not word) word: (’says’, 0.41209148147375985)
the  5  largest P(fake|not word) word: (’north’, 0.4070303737043043)
the  6  largest P(fake|not word) word: (’wall’, 0.40181104805919354)
the  7  largest P(fake|not word) word: (’comments’, 0.40117511020703106)
the  8  largest P(fake|not word) word: (’trade’, 0.4008207565562279)
the  9  largest P(fake|not word) word: (’deal’, 0.40073997420015695)
the  10  largest P(fake|not word) word: (’court’, 0.4007331148361253)
```

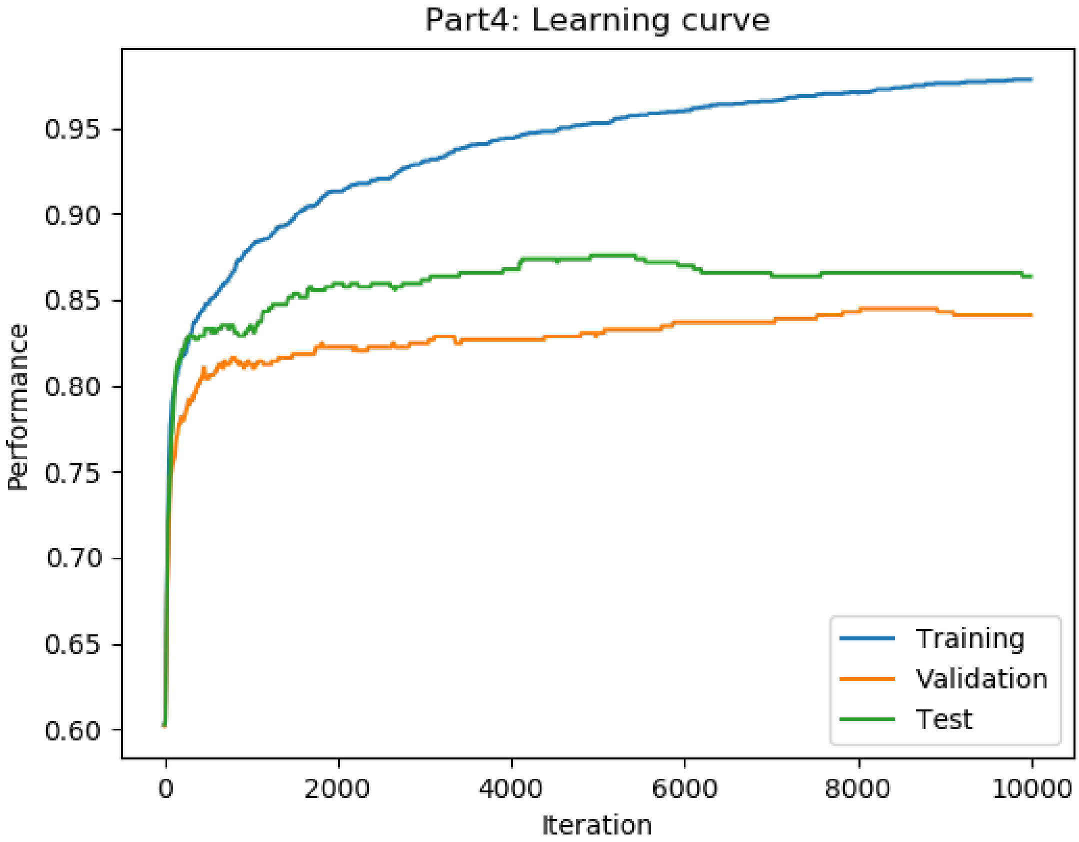
**(b)** 10 non-stopwords whose presence most strongly predicts that the news is real.
the 1 largest P(real|word) non-stopwords: ban
the 2 largest P(real|word) non-stopwords: korea
the 3 largest P(real|word) non-stopwords: travel
the 4 largest P(real|word) non-stopwords: turnbull
the 5 largest P(real|word) non-stopwords: australia
the 6 largest P(real|word) non-stopwords: climate
the 7 largest P(real|word) non-stopwords: paris
the 8 largest P(real|word) non-stopwords: refugee
the 9 largest P(real|word) non-stopwords: trumps
the 10 largest P(real|word) non-stopwords: tpp
10 non-stopwords whose absence most strongly predicts that the news is real.
the 1 largest P(real|not word) non-stopwords: trump
the 2 largest P(real|not word) non-stopwords: hillary
the 3 largest P(real|not word) non-stopwords: clinton
the 4 largest P(real|not word) non-stopwords: just
the 5 largest P(real|not word) non-stopwords: win
the 6 largest P(real|not word) non-stopwords: new
the 7 largest P(real|not word) non-stopwords: victory
the 8 largest P(real|not word) non-stopwords: america
the 9 largest P(real|not word) non-stopwords: watch
the 10 largest P(real|not word) non-stopwords: black
10 non-stopwords whose presence most strongly predicts that the news is fake.
the 1 largest P(fake| word) non-stopwords: breaking
the 2 largest P(fake| word) non-stopwords: u
the 3 largest P(fake| word) non-stopwords: 3
the 4 largest P(fake| word) non-stopwords: soros
the 5 largest P(fake| word) non-stopwords: m
the 6 largest P(fake| word) non-stopwords: woman
the 7 largest P(fake| word) non-stopwords: secret
the 8 largest P(fake| word) non-stopwords: 7
the 9 largest P(fake| word) non-stopwords: steal
the 10 largest P(fake| word) non-stopwords: reporter
10 non-stopwords whose absence most strongly predicts that the news is fake.
the 1 largest P(fake|not word) non-stopwords: donald
the 2 largest P(fake|not word) non-stopwords: trumps
the 4 largest P(fake|not word) non-stopwords: says
the 5 largest P(fake|not word) non-stopwords: north
the 6 largest P(fake|not word) non-stopwords: wall
the 7 largest P(fake|not word) non-stopwords: comments
the 8 largest P(fake|not word) non-stopwords: trade
the 9 largest P(fake|not word) non-stopwords: deal
the 10 largest P(fake|not word) non-stopwords: court
(c)
Why might it make sense to remove stop words when interpreting the model? Why might it make sense to keep stop words?
It makes sense to remove stop words because those stop words are not meaningful, they are not importance in the semantic of the headline. Moreover, our model may not generalized to other data.

However, if there are pattern on how real news and fake news using stop words, and the pattern of them are unique. Then it makes sense to use stop words. For example, we can see in the list of 10 words whose absence most strongly predicts that the news is real. There are many stop words in it, and this is not what happened in fake new's list.

## Part 4

**Plot the learning curves (performance vs. iteration) of the Logistic Regression model. Describe how you selected the regularization parameter (and describe the experiments you used to select it).**

I used Sigmoid activation function for this part. I have tried different values for alpha and lambda. The alpha and lambda I chose are 0.1 and 0.0005 respectively. The learning curve of the logistic regression model see following figure.



final performance on training: 0.9763676148796498
final performance on validation: 0.8428571428571429
final performance on test: 0.8676171079429735

# Part 5

Naive Bayes model: (slides: Naive Bayes)

$$log\frac{P(y=c|x_1,...,x_p)}{P(y=c'|x_1,...,x_p)} = \theta_0 + \sum_j \theta_j x_j$$

$$= \left(log\frac{P(y=c)}{P(y=c')} + \sum_{i=1}^{p} log\frac{P(x_j=0|y=c)}{P(x_j=0|y=c')}\right) + \sum_{i=1}^{p}\left(log\frac{P(x_j=1|y=c)}{P(x_j=1|y=c')} - log\frac{P(x_j=0|y=c)}{P(x_j=0|y=c')}\right)x_j$$

$\theta_0 + \theta_1 I_1(x) + \theta_2 I_2(x) + ... + \theta_k I_k(x) > thr$
For Naive Bayes, $\theta_i$ is the one shows above. $l_i(x)$ means that for the headline x, if the $word_i$is in this headline, then it will be 1, vice versa.
In logistic regression, $\theta_1....\theta_k$are the weights for each word. They determine how much each word contribute to the result(whether fake or real). $theta_0$ is the bias. $l_i(x)$ means that for the headline x, if the $word_i$is in this headline, then it will be 1, vice versa.
For both Naive Bayes and logistic regression, the threshold is 0. If $\theta > 0$, indicates it is real and if $\theta < 0$, indicates it is fake.

# Part 6

**6(a)**

**Top10 positive theta(s) obtained from Logistic Regression with the words**

The 1st word is hillary with theta: 2.2544468
The 2nd word is breaking with theta: 2.0707388
The 3rd word is just with theta: 1.7770123
The 4th word is watch with theta: 1.7486714
The 5th word is black with theta: 1.6112970
The 6th word is victory with theta: 1.5975143
The 7th word is video with theta: 1.4857363
The 8th word is war with theta: 1.4855826
The 9th word is go with theta: 1.4734819
The 10th word is that with theta: 1.4359586

**Top10 negative theta(s) obtained from Logistic Regression with the words**

The 1st word is trumps with theta: -2.6839701
The 2nd word is drives with theta: -2.0345477
The 3rd word is australia with theta: -1.7535033
The 4th word is turnbull with theta: -1.6520346
The 5th word is says with theta: -1.6261315
The 6th word is us with theta: -1.4512048
The 7th word is climate with theta: -1.4089813
The 8th word is ban with theta: -1.3858742
The 9th word is north with theta: -1.3400625
The 10th word is tax with theta: -1.3389058

The words I produced in this part, there are some overlap with results from 3(a).

**6(b)**

**Top10 positive theta(s) obtained from Logistic Regression without stopwords**

The 1st word is hillary with theta: 2.2544468
The 2nd word is breaking with theta: 2.0707388
The 3rd word is just with theta: 1.7770123
The 4th word is watch with theta: 1.7486714
The 5th word is black with theta: 1.6112970
The 6th word is victory with theta: 1.5975143
The 7th word is video with theta: 1.4857363
The 8th word is war with theta: 1.4855826
The 9th word is soros with theta: 1.4283883
The 10th word is u with theta: 1.3874369

**Top10 negative theta(s) obtained from Logistic Regression without stopwords**

The 1st word is trumps with theta: -2.6839701
The 2nd word is drives with theta: -2.0345477
The 3rd word is australia with theta: -1.7535033
The 4th word is turnbull with theta: -1.6520346
The 5th word is says with theta: -1.6261315
The 6th word is climate with theta: -1.4089813
The 7th word is ban with theta: -1.3858742
The 8th word is north with theta: -1.3400625
The 9th word is tax with theta: -1.3389058
The 10th word is korea with theta: -1.3335532

The words I produced in this part have no stopwords. There are some words overlap with the results from 3(b).

**6(c)**

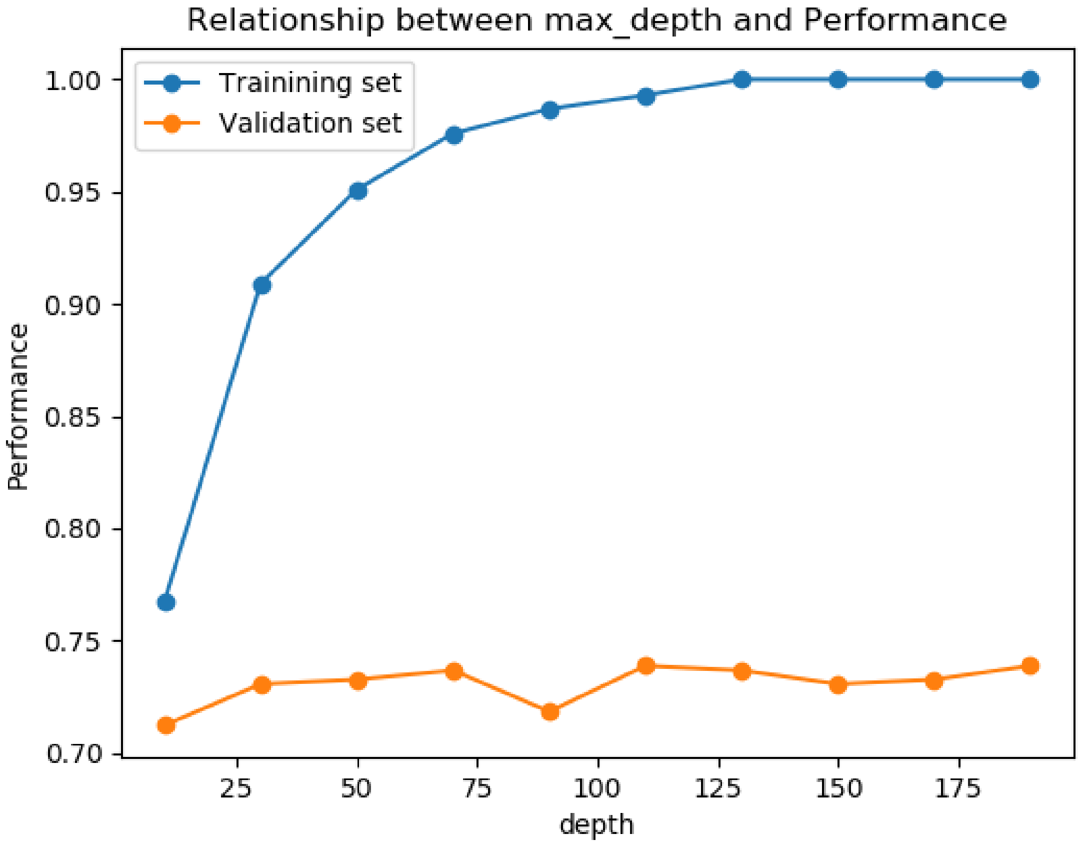
We used the magnitude of the logistic regression parameters to indicate importance of a feature.

This is probably a bad idea in general. If the features are not normalized, certain features' magnitude will be much larger than the others depend on their importance. This probably will effect the resulting performance.

It is reasonable to use the magnitude in this problem is becuase that the inputs are binary signals. We are performing a binary classification in this case. The magnitude is only used to indicate a headline is real or fake.

# Part 7

**(a)**

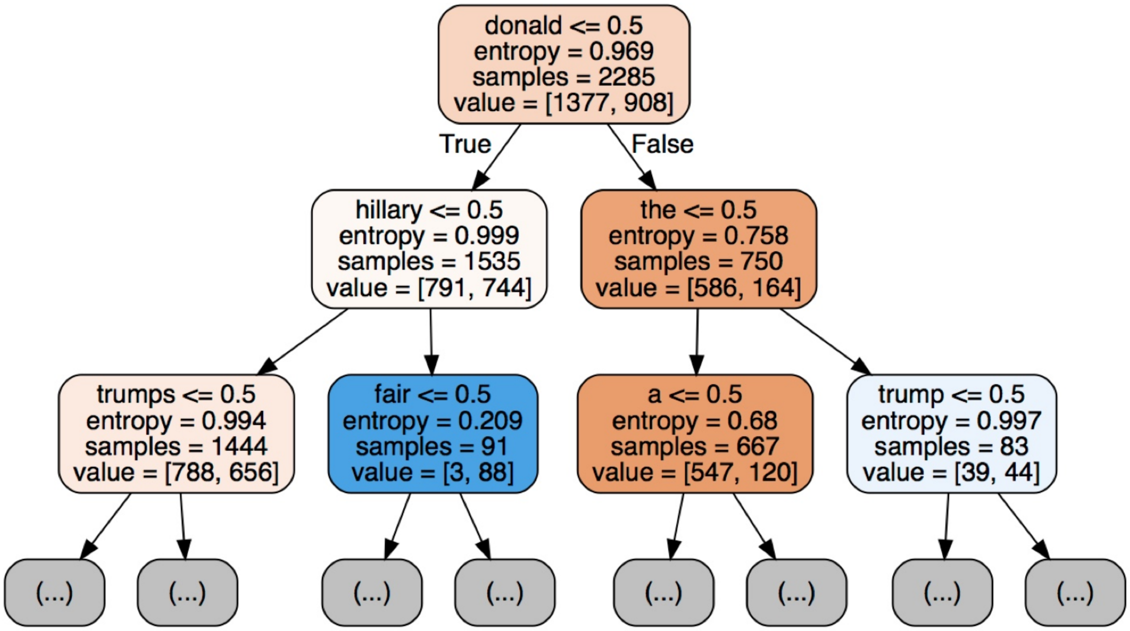


I have tried max_depth = $np.arange(10, 210, 20)$, to find the highest performance on validation set. The final max_depth is 110 as it produce the highest accuracy.

We use criterion = 'entropy', because entropy is more likely to deal with class while Gini is more likely to deal with continuous attributes.

The use default setting for the rest parameter.

best performance training: 0.992997811816 best performance validation: 0.738775510204



(b)
overlap with logistic regression: hillary, trumps
overlap with naive bayes: donald, hilliary, the, trumps, a, trump
It seems that the decision trees have more common features with naive bayes than logistic regression.

(c)
Naive Bayes:
training set performance: 0.941794310722
valid set performance: 0.769387755102
test set performance: 0.775967413442

Logistic Regression:
final performance on training: 0.9763676148796498
final performance on validation: 0.8428571428571429
final performance on test: 0.8676171079429735

Decision Tree:
best performance training: 0.992997811816
best performance validation: 0.738775510204
best performance test: 0.738775510204

Overall, the decision tree has the lowest performance. Although, it has the highest performance in training set, but this can not be generalized to validation and test set and may considered as over fitting. Logistic Regression performs better than Naive Bayes, and they are more related to each other than they related to decision tree. The over fitting are less likely to happen in naive bayes and logistic regression than in the decision tree.
In conclusion, logistic regression performs the best, decision tree performs the worst. Decision tree overfit the most

# Part 8

### 8(a)

```
def conditional_entropy(s1, s2):
    l = float(s1)/(s1+s2) * math.log(float(s1)/(s1+s2), 2)
    r = float(s2)/(s1+s2) * math.log(float(s2)/(s1+s2), 2)
    return -(l+r)

def count_pNa(train_data, w):
    presence, absence = 0, 0
    for line in train_data:
        if w in line:
            presence += 1
        else:
            absence += 1
    return float(presence), float(absence)

def part8(x_var):

    real_data = read_data("./clean_real.txt")
    fake_data = read_data("./clean_fake.txt")

    real_train, real_valid, real_test = get_set(real_data)
    fake_train, fake_valid, fake_test = get_set(fake_data)
    rtrain_size, ftrain_size = len(real_train), len(fake_train)
    total_size = len(real_train) + len(fake_train)
    print("dividing sets...")

    # compute the mutual information of the split on the training data
    presence_real, absence_real = count_pNa(real_train, x_var)
    presence_fake, absence_fake = count_pNa(fake_train, x_var)
    absence_p = (conditional_entropy(absence_real, absence_fake) * \
                      ((absence_real + absence_fake) / total_size))
    presence_p = (conditional_entropy(presence_real, presence_fake) * \
                      ((presence_real + presence_fake) / total_size))

    return conditional_entropy(rtrain_size, ftrain_size) - (absence_p + presence_p)

# printing results for x_i and x_j which x_i != x_j
donald_result = part8("donald")
print(donald_result)

hillary_result = part8("hillary")
print(hillary_result)
```

(a) $I(Y, x_i) = I(Y, ”\textit{donald}”) = 0.049369429536$

(b) $I(Y, x_j) = I(Y, ”\textit{hillary}”) = 0.0377288495484$