595 Final Project

Pikaso Dutta

Function 1: Login to webpage

This part of code allows us to login to Facebook, with the user id and password being updated in the text file. **Selenium** is used here for the web-automation library. Selenium will need the last **Chrome driver**, however the latest driver works only with Google chrome **version 71/72/73**.

```
# 1. Login to webpage
def login():

    # Import username and password from text file
    Input = open("User_Id.txt","r")
    data = Input.readlines()
    username = data[1][5:-1].lstrip()
    password = data[2][5:-1].lstrip()

    #access website
    try:
        driver.get('https://www.facebook.com/')

        #Accessing Login frame
        form=driver.find_element_by_id('login_form')
        form.click()

        #Entering email details
        email = form.find_element_by_id('email')
        email.send_keys(username)
        #time.sleep(1)

        #Entering password details
        pwd = form.find_element_by_id('pass')
        pwd.send_keys(password)
        time.sleep(1)

        #Clicking the login button
        button=WebDriverWait(driver, 1000).until(EC.element_to_b
        button.click()

    except Exception as e:
        print('Exception encountered during Login')
        print(e)
```

Function 9: Clean Sentences

User comments might be not in the exact format required to do further analysis. Thus using **regex (regular expression)** we can correctly format them.

```
# 9. Clean Sentences
def cleanup_data(sentences):
    clean_sentences = []
    for sentence in sentences:
        sentence=re.sub(r"http.?://[^\s]+[\s]?","",sentence)
        sentence=re.sub(r"@[^\s]+[\s]?","",sentence)
        sentence=re.sub(r"\s?[0-9]+\.?[0-9]*","",sentence)
        sentence=re.sub(r"[^a-zA-Z0-9 ]","",sentence)
        clean_sentences.append(sentence)

    return clean_sentences
```

Function 10: Naive Based Classifier

We have Naïve Based Classifier to predict how well the NB can predict from the comments if it is a positive or a negative comment. We have achieved the below accuracy

- T-Mobile: 61%
- AT&T: 59%

```
# 10. Naive Based Classifier
def multinomialNB(clean_sentences,label):

    # initialize the TfidfVectorizer
    tfidf_vect = TfidfVectorizer()
    # with stop words removed
    tfidf_vect = TfidfVectorizer(stop_words = "english")
    # generate tfidf matrix
    dtm = tfidf_vect.fit_transform(clean_sentences).toarray()
    X_train, X_test, y_train, y_test = train_test_split(dtm, label,
    clf = MultinomialNB().fit(X_train,y_train)
    accuracy = [cross_val_score(clf, dtm, label, cv=80)]

    return (str(round(np.array(accuracy[0]).mean()*100,2))+'%')
```