# hwq3.3.1

January 11, 2017

```
In [2]: import numpy as np
        import pandas as pd
        import sklearn.linear_model as lm
        from sklearn.model_selection import GridSearchCV
        import matplotlib.pyplot as plt
        %matplotlib inline
```

```
In [3]: train = pd.read_csv('~/work/PresidentialTweets/tweets_train.csv')
        test = pd.read_csv('~/work/PresidentialTweets/tweets_test.csv')
```

## 1 Analyzing data

We will start with basic data profiling to so where were at.
First test and train:

```
In [4]: print ("Dimension of train data {}".format(train.shape))
        print ("Dimension of test data {}".format(test.shape))
        print ("Basic statistical description:")
        train.describe()
```

```
Dimension of train data (3999, 9)
Dimension of test data (2349, 11)
Basic statistical description:
```

Out[4]:

| | id | retweet_count | favorite_count |
|---|---|---|---|
| count | 3.999000e+03 | 3999.000000 | 3999.000000 |
| mean | 7.594926e+17 | 4964.529632 | 12911.969742 |
| std | 1.278198e+16 | 10017.445651 | 18009.926718 |
| min | 7.350000e+17 | 123.000000 | 415.000000 |
| 25% | 7.500000e+17 | 1409.000000 | 3644.000000 |
| 50% | 7.580000e+17 | 2980.000000 | 7537.000000 |
| 75% | 7.700000e+17 | 6213.500000 | 17551.500000 |
| max | 7.810000e+17 | 490180.000000 | 660384.000000 |

```
In [61]: test.describe()
```

Out[61]:

| | id | handle | retweet_count | favorite_count |
|---|---|---|---|---|
| count | 2.349000e+03 | 0.0 | 2349.000000 | 2349.000000 |
| mean | 7.108067e+17 | NaN | 3454.904640 | 9569.874415 |
| std | 1.498040e+16 | NaN | 3174.856432 | 7374.142302 |
| min | 6.840000e+17 | NaN | 169.000000 | 307.000000 |
| 25% | 6.980000e+17 | NaN | 1538.000000 | 4508.000000 |
| 50% | 7.100000e+17 | NaN | 2682.000000 | 7850.000000 |
| 75% | 7.250000e+17 | NaN | 4473.000000 | 12944.000000 |
| max | 7.340000e+17 | NaN | 82653.000000 | 115107.000000 |

We can see there are almost two times more samples in train and so when were checkig distributions we need to take this into account. Also we should take into account that although retweet and favorite count have the same median the top 75% already shows bias which might influence in favour of one candidate and may yield better results if we have a better prior there.

Now some comparisons between data on trump vs hillary:

In [4]:
```
hilary = train.loc[train.handle == 'HillaryClinton']
print(len(hilary))
hilary.describe()
```

```
2724
```

Out[4]:

| | id | retweet_count | favorite_count |
|---|---|---|---|
| count | 2.724000e+03 | 2724.000000 | 2724.000000 |
| mean | 7.601413e+17 | 3264.672907 | 7379.947137 |
| std | 1.256154e+16 | 10518.057451 | 15833.686327 |
| min | 7.350000e+17 | 123.000000 | 415.000000 |
| 25% | 7.510000e+17 | 1119.000000 | 2846.500000 |
| 50% | 7.590000e+17 | 1880.000000 | 4746.000000 |
| 75% | 7.710000e+17 | 3330.250000 | 7941.750000 |
| max | 7.810000e+17 | 490180.000000 | 660384.000000 |

In [5]:
```
trump = train.loc[train.handle != 'HillaryClinton']
print(len(trump))
trump.describe()
```

```
1275
```

Out[5]:

| | id | retweet_count | favorite_count |
|---|---|---|---|
| count | 1.275000e+03 | 1275.000000 | 1275.000000 |
| mean | 7.581067e+17 | 8596.223529 | 24730.973333 |
| std | 1.313872e+16 | 7686.186621 | 16637.305561 |
| min | 7.350000e+17 | 1165.000000 | 5166.000000 |
| 25% | 7.470000e+17 | 4834.500000 | 14838.000000 |
| 50% | 7.580000e+17 | 7159.000000 | 21540.000000 |
| 75% | 7.690000e+17 | 10215.500000 | 29469.000000 |
| max | 7.810000e+17 | 167274.000000 | 294162.000000 |

first thing to notice is that trump has about half of the tweets and that should be taken into account in the analasys. Also looking at the favorite count we see trump is a clear leader and the test data shouldn'e worry us as even though the test tweets were less popular the above 75 percent will not be problamatic as in both places they are probably trump and the destribution is the same sort of. in retweet count it doesn't affect us because of the same reasons.

**Looking at the attributes:**

```
In [13]: test.head()
```

Out[13]:

| | id | handle | text |
|---|---|---|---|
| 0 | 7.340000e+17 | NaN | #MichaelBrown would have been 20 years old tod... |
| 1 | 7.340000e+17 | NaN | Congratulations on becoming a U.S. citizen, Al... |
| 2 | 7.340000e+17 | NaN | We need a president who will unite leaders aro... |
| 3 | 7.340000e+17 | NaN | Dear Congress,\n\nLet's get this done.\n\nThan... |
| 4 | 7.340000e+17 | NaN | Failing @NYTimes will always take a good story... |

| | is_retweet | original_author | time | in_reply_to_screen_name |
|---|---|---|---|---|
| 0 | True | LSD_Esq | 2016-05-20T18:07:08 | NaN |
| 1 | False | NaN | 2016-05-20T17:24:12 | NaN |
| 2 | False | NaN | 2016-05-20T17:12:52 | NaN |
| 3 | False | NaN | 2016-05-20T16:21:13 | NaN |
| 4 | False | NaN | 2016-05-20T16:11:21 | NaN |

| | is_quote_status | lang | retweet_count | favorite_count |
|---|---|---|---|---|
| 0 | False | en | 594 | 1096 |
| 1 | False | en | 1701 | 4239 |
| 2 | False | en | 1817 | 3577 |
| 3 | False | en | 2530 | 6012 |
| 4 | False | en | 3750 | 12372 |

```
In [14]: test.tail()
```

Out[14]:

| | id | handle | t |
|---|---|---|---|
| 2344 | 6.840000e+17 | NaN | "@lilredfrmkokomo: @realDonaldTrump My Faceboo |
| 2345 | 6.840000e+17 | NaN | "@marybnall01: @realDonaldTrump watched lowell |
| 2346 | 6.840000e+17 | NaN | "@ghosthunter_lol: Iowa key endorsement for @r |
| 2347 | 6.840000e+17 | NaN | "@iLoveiDevices: @EdwinRo47796972 @happyjack22 |
| 2348 | 6.840000e+17 | NaN | "@SalRiccobono: @realDonaldTrump @troyconway D |

| | is_retweet | original_author | time | in_reply_to_screen_na |
|---|---|---|---|---|
| 2344 | False | NaN | 2016-01-05T03:47:14 | N |
| 2345 | False | NaN | 2016-01-05T03:44:17 | N |
| 2346 | False | NaN | 2016-01-05T03:42:10 | N |
| 2347 | False | NaN | 2016-01-05T03:39:11 | N |
| 2348 | False | NaN | 2016-01-05T03:36:53 | N |

| | is_quote_status | lang | retweet_count | favorite_count |
|---|---|---|---|---|

```
2344          False   en           1110            4024
2345          False   en            855            3181
2346          False   en           2315            5992
2347          False   en           1054            3258
2348          False   en            748            2658
```

In [15]: `hilary.head()`

Out[15]:

```
             id          handle  \
0  7.810000e+17  HillaryClinton
1  7.810000e+17  HillaryClinton
2  7.810000e+17  HillaryClinton
3  7.810000e+17  HillaryClinton
4  7.810000e+17  HillaryClinton

                                                text is_retweet  \
0  The question in this election: Who can put the...      False
1  Last night, Donald Trump said not paying taxes...       True
2  Couldn't be more proud of @HillaryClinton. Her...       True
3  If we stand together, there's nothing we can't...      False
4  Both candidates were asked about how they'd co...      False

  original_author                 time lang  retweet_count  favorite_count
0             NaN  2016-09-28T00:22:34   en            218             651
1        timkaine  2016-09-27T23:45:00   en           2445            5308
2           POTUS  2016-09-27T23:26:40   en           7834           27234
3             NaN  2016-09-27T23:08:41   en            916            2542
4             NaN  2016-09-27T22:30:27   en            859            2882
```

In [16]: `hilary.tail()`

Out[16]:

```
                 id          handle  \
3984  7.360000e+17  HillaryClinton
3985  7.360000e+17  HillaryClinton
3986  7.350000e+17  HillaryClinton
3987  7.350000e+17  HillaryClinton
3990  7.350000e+17  HillaryClinton

                                                text is_retweet  \
3984  We're stronger together. When we embrace immig...      False
3985  Donald Trump has spent his career looking to t...      False
3986  Trump bet against American families in the hou...       True
3987  Mientras 5 millones de personas perdían sus ho...      False
3990  Millions of families saw their life savings de...      False

      original_author                 time lang  retweet_count  favorite_c
3984              NaN  2016-05-25T18:02:01   en           1353
3985              NaN  2016-05-25T15:58:25   en            971
3986  TheBriefing2016  2016-05-25T15:17:32   en            899
```

```
3987              NaN  2016-05-25T14:24:10   es            801
3990              NaN  2016-05-25T12:57:16   en           1987
```

In [17]:
```
trump.head()
```

Out[17]:
```
                id           handle  \
5   7.810000e+17  realDonaldTrump
8   7.810000e+17  realDonaldTrump
11  7.810000e+17  realDonaldTrump
12  7.810000e+17  realDonaldTrump
13  7.810000e+17  realDonaldTrump

                                                 text is_retweet  \
5   Join me for a 3pm rally - tomorrow at the Mid-...      False
8   Once again, we will have a government of, by a...      False
11  On National #VoterRegistrationDay, make sure y...       True
12  Hillary Clinton's Campaign Continues To Make F...      False
13  'CNBC, Time magazine online polls say Donald T...      False

   original_author                 time lang  retweet_count  favorite_coun
5              NaN  2016-09-27T22:13:24   en           2181            617
8              NaN  2016-09-27T21:08:22   en           4132           1123
11             GOP  2016-09-27T20:31:14   en           2953            696
12             NaN  2016-09-27T20:14:33   en           3833            984
13             NaN  2016-09-27T20:06:25   en           4236           1094
```

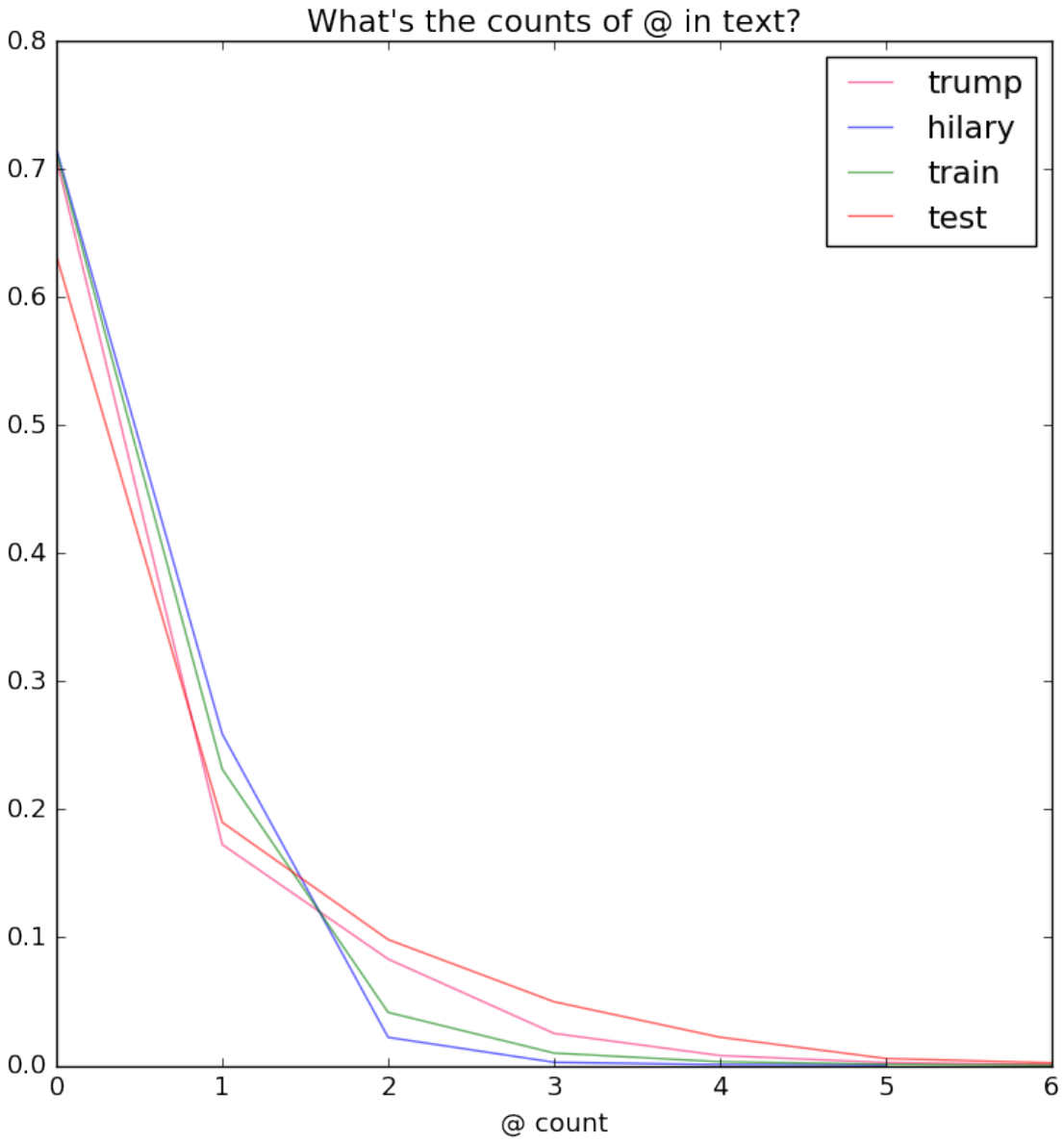
In [18]:
```
trump.tail()
```

Out[18]:
```
                 id           handle  \
3994  7.350000e+17  realDonaldTrump
3995  7.350000e+17  realDonaldTrump
3996  7.350000e+17  realDonaldTrump
3997  7.350000e+17  realDonaldTrump
3998  7.350000e+17  realDonaldTrump

                                                text is_retweet  \
3994     "@buiIdthewall: @realDonaldTrump high energy!"      False
3995  "@PiperSul: Great speech tonight Mr.Trump! Goo...      False
3996  "@DeepakS76435750: @realDonaldTrump congratula...      False
3997  "@oasisupernova: @realDonaldTrump UP TO 8.4 MI...      False
3998          "@jknatter: @realDonaldTrump #TrumpTrain"      False

     original_author                 time lang  retweet_count  favorite_co
3994             NaN  2016-05-25T05:45:19   en           1984            8
3995             NaN  2016-05-25T05:45:11   en           1827            7
3996             NaN  2016-05-25T05:42:59   en           1669            6
3997             NaN  2016-05-25T05:42:49   en           2684            9
3998             NaN  2016-05-25T05:41:38  und           1165            5
```

### 1.1 Charts

Let us see if we can learn something from special charecters. We normalised the counts so the graphs will match if the distributions match.

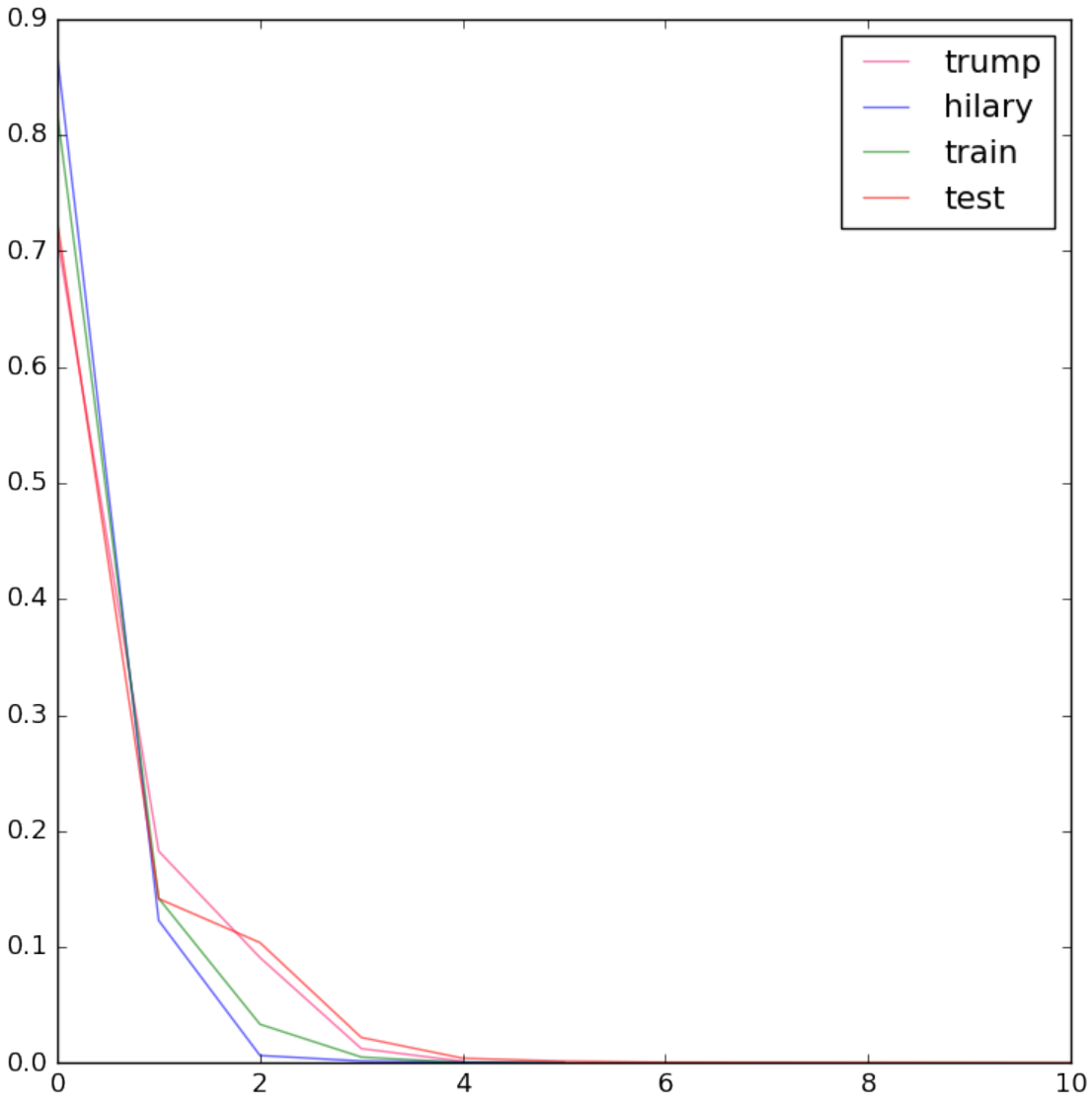
```
In [11]: plt.rc('font', size=13)
         fig = plt.figure(figsize=(30, 20))
         alpha = 0.6

         def get_special_chars(df):
             res = pd.DataFrame()
             res.reindex(df.index)
             res['@count'] = df['text'].apply(lambda row: row.count('@'))
             res['#count'] = df['text'].apply(lambda row: row.count('#'))
             res['.count'] = df['text'].apply(lambda row: row.count('.'))
             res[',count'] = df['text'].apply(lambda row: row.count(','))
             res['-count'] = df['text'].apply(lambda row: row.count('-'))
             return res

         counted_trump = get_special_chars(trump)
         counted_hilary = get_special_chars(hilary)
         counted_test = get_special_chars(test)
         counted_train = get_special_chars(train)

         ax1 = plt.subplot2grid((2,3), (0,0))
         # since we are counting values it makes sense to regularize the counts
         (counted_trump['@count'].value_counts()  / counted_trump.shape[0]).plot(ki
         (counted_hilary['@count'].value_counts()  / counted_hilary.shape[0]).plot
         (counted_train['@count'].value_counts() / counted_train.shape[0]).plot(kir
         (counted_test['@count'].value_counts() / counted_test.shape[0]).plot(kind=
         ax1.set_xlabel('@ count')
         ax1.set_title("What's the counts of @ in text?" )
         plt.legend(loc='best')

Out[11]: <matplotlib.legend.Legend at 0x7f89ecdfdef0>
```




```
In [12]: plt.rc('font', size=13)
         fig = plt.figure(figsize=(30, 20))
         alpha = 0.6

         ax2 = plt.subplot2grid((2,3), (0,1))
         # since we are counting values it makes sense to regularize the counts
         (counted_trump['#count'].value_counts()  / counted_trump.shape[0]).plot(ki
         (counted_hilary['#count'].value_counts()  / counted_hilary.shape[0]).plot
         (counted_train['#count'].value_counts() / counted_train.shape[0]).plot(kir
         (counted_test['#count'].value_counts() / counted_test.shape[0]).plot(kind=
         ax1.set_xlabel('# count')
```

```
        ax1.set_title("What's the counts of # in text?" )
        plt.legend(loc='best')
```

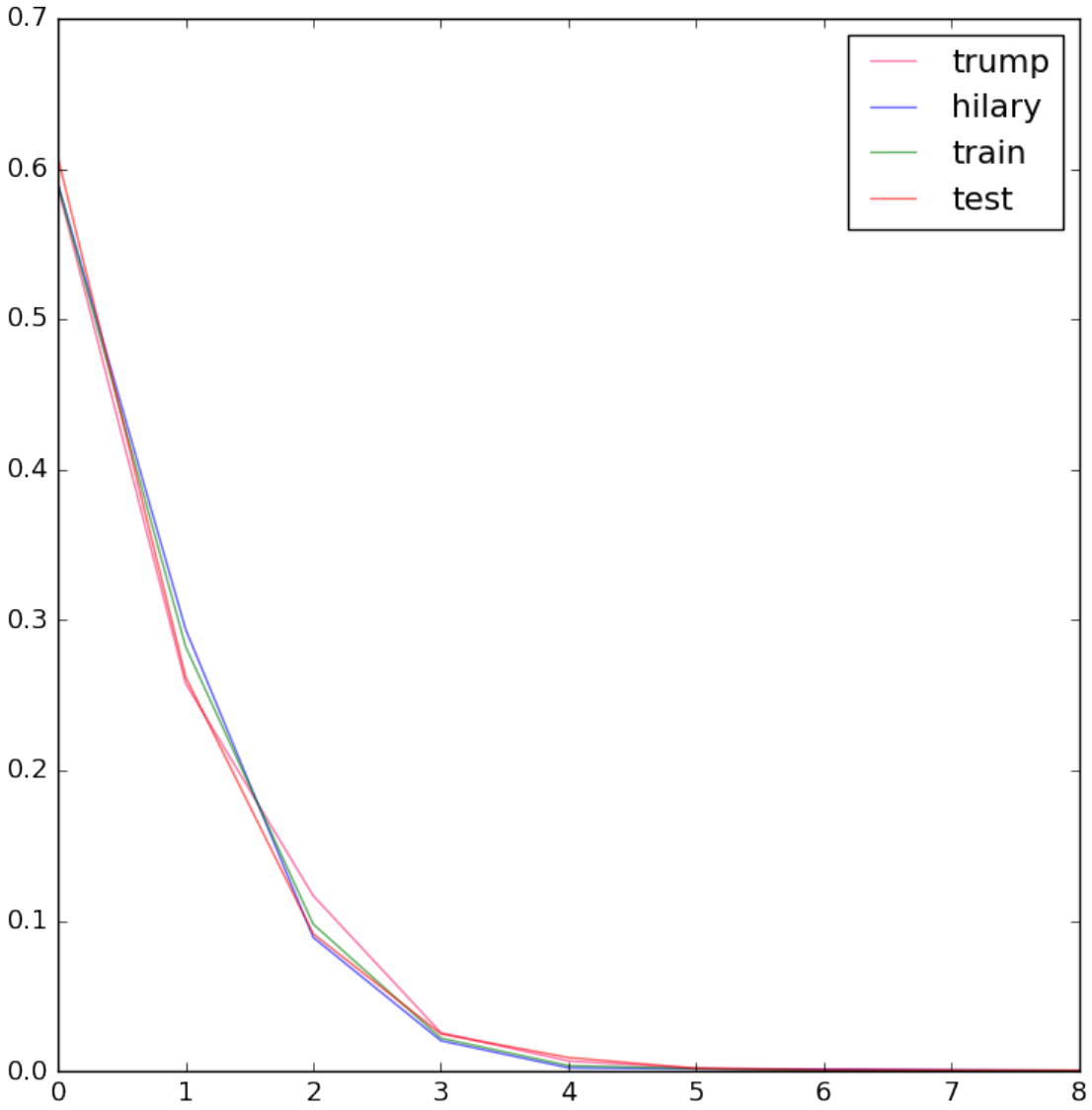
```
Out[12]: <matplotlib.legend.Legend at 0x7f89f055cfd0>
```




```
In [9]: plt.rc('font', size=13)
        fig = plt.figure(figsize=(30, 20))
        alpha = 0.6

        ax4 = plt.subplot2grid((2,3), (1,0))
        # since we are counting values it makes sense to regularize the counts
        (counted_trump[',count'].value_counts()  / counted_trump.shape[0]).plot(kin
        (counted_hilary[',count'].value_counts()  / counted_hilary.shape[0]).plot(k
```

```
        (counted_train[',count'].value_counts() / counted_train.shape[0]).plot(kind
        (counted_test[',count'].value_counts() / counted_test.shape[0]).plot(kind='
        ax1.set_xlabel(', count')
        ax1.set_title("What's the counts of , in text?" )
        plt.legend(loc='best')
```

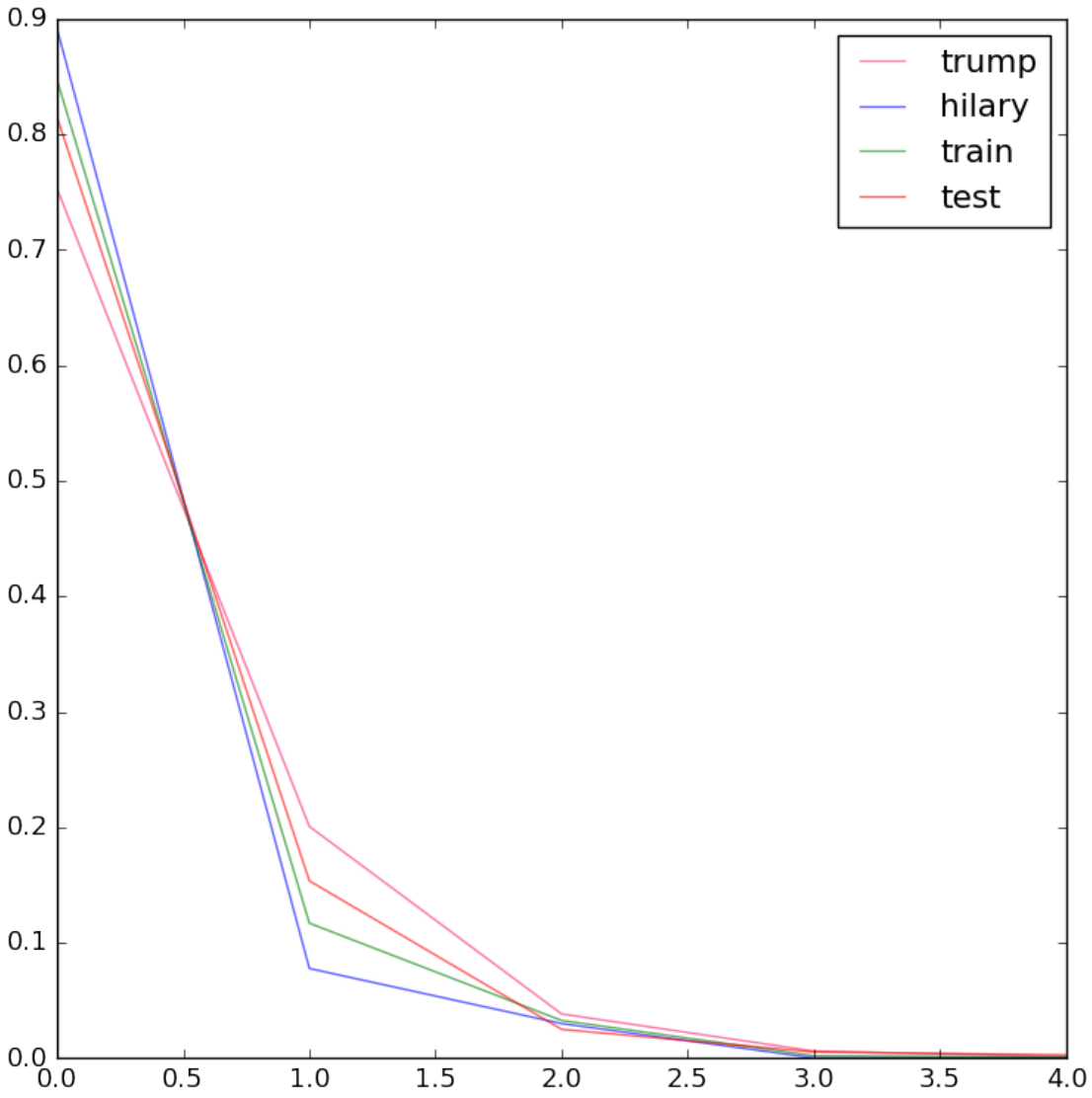
Out[9]: <matplotlib.legend.Legend at 0x7f89ecf79dd8>




In [10]:
```
plt.rc('font', size=13)
fig = plt.figure(figsize=(30, 20))
alpha = 0.6

ax5 = plt.subplot2grid((2,3), (1,1))
```

```
# since we are counting values it makes sense to regularize the counts
(counted_trump['-count'].value_counts()  / counted_trump.shape[0]).plot(ki
(counted_hilary['-count'].value_counts()  / counted_hilary.shape[0]).plot
(counted_train['-count'].value_counts() / counted_train.shape[0]).plot(kir
(counted_test['-count'].value_counts() / counted_test.shape[0]).plot(kind=
ax1.set_xlabel('- count')
ax1.set_title("What's the counts of - in text?" )
plt.legend(loc='best')
```

```
Out[10]: <matplotlib.legend.Legend at 0x7f89ecf009b0>
```

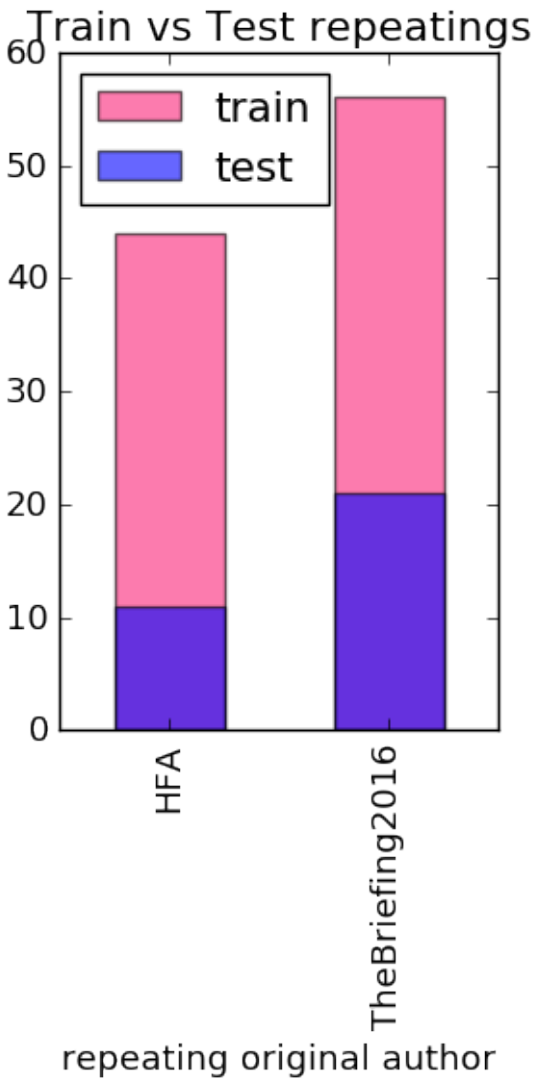



seems like trump is alot more likely to use special charecters more then once in a tweet which will probably be a usefull feature. Here we see a bias in train/test, train is really missing alot of this compared to test which might indicate trump is more common in test or we have a bias. Let's hope for the first. ### Now let us to look at other features our columns are:

```
In [13]: print(" ".join(train.columns))

id handle text is_retweet original_author time lang retweet_count favorite_count
```

So it seems logical to look at languages, repeating authors and times. maybe also find if there is a repeating original auther. Repeating authors are only interesting if they are both in train and test:

```
In [14]: def get_all_repeating_original(df):
             counts = df['original_author'].value_counts()
             return counts[counts > 3]

         (get_all_repeating_original(train) + get_all_repeating_original(test)).dro

Out[14]: HFA                55.0
         TheBriefing2016    77.0
         Name: original_author, dtype: float64

In [16]: plt.rc('font', size=13)
         fig = plt.figure(figsize=(10, 10))
         alpha = 0.6

         ax1 = plt.subplot2grid((2,3), (0,0))
         # since we are counting values it makes sense to regularize the counts
         repeating_train = (get_all_repeating_original(train) + get_all_repeating_o
         repeating_test = (get_all_repeating_original(train) + get_all_repeating_or
         repeating_train.plot(kind='bar', color='#FA2379', label='train', alpha=alp
         repeating_test.plot(kind='bar', label='test', alpha=alpha)
         ax1.set_xlabel('repeating original author')
         ax1.set_title("Train vs Test repeatings" )
         plt.legend(loc='best')

Out[16]: <matplotlib.legend.Legend at 0x7f89eccda5c0>
```




Lets test trump vs hilary with these to see what we can learn

```
In [18]: print('Trump original authors')
         print(get_all_repeating_original(trump))
         print('\nHillary original authors')
         print(get_all_repeating_original(hilary))
```

```
Trump original authors
TeamTrump         6
DRUDGE_REPORT     6
DanScavino        5
IvankaTrump       4
Name: original_author, dtype: int64

Hillary original authors
```

```
TheBriefing2016    56
HFA                44
timkaine           42
Hillary_esp        22
JoeBiden           15
BernieSanders       9
mayaharris_         9
johnpodesta         9
rosenbergerlm       7
Jorge_Silva         6
billclinton         6
ChelseaClinton      5
WhiteHouse          5
HillaryforVA        5
mikereedschmidt     5
lorellapraeli       4
elizabethforma      4
dominiclowell       4
repjohnlewis        4
HillaryforPA        4
POTUS               4
mpshapiro           4
Name: original_author, dtype: int64
```

If we will see an original_author in the test we will now it is alot more likely that hillary retweeted it.

Let us look at languages:

```
In [79]: lang_train = train['lang'].value_counts()
         lang_test = test['lang'].value_counts()
         print("Showing training value counts")
         print(lang_train)
         print("")
         print("Showing test value counts")
         print(lang_test)

Showing training value counts
en     3890
es       68
und      40
fr        1
Name: lang, dtype: int64

Showing test value counts
en     2265
und      42
es       34
```

```
da         3
tl         2
et         1
fr         1
fi         1
Name: lang, dtype: int64
```

```
In [80]: lang_trump = trump['lang'].value_counts()
         lang_hilary = hilary['lang'].value_counts()
         print("Showing trump value counts")
         print(lang_trump)
         print("")
         print("Showing hilary value counts")
         print(lang_hilary)
```

```
Showing trump value counts
en     1241
und      34
Name: lang, dtype: int64

Showing hilary value counts
en     2649
es       68
und       6
fr        1
Name: lang, dtype: int64
```

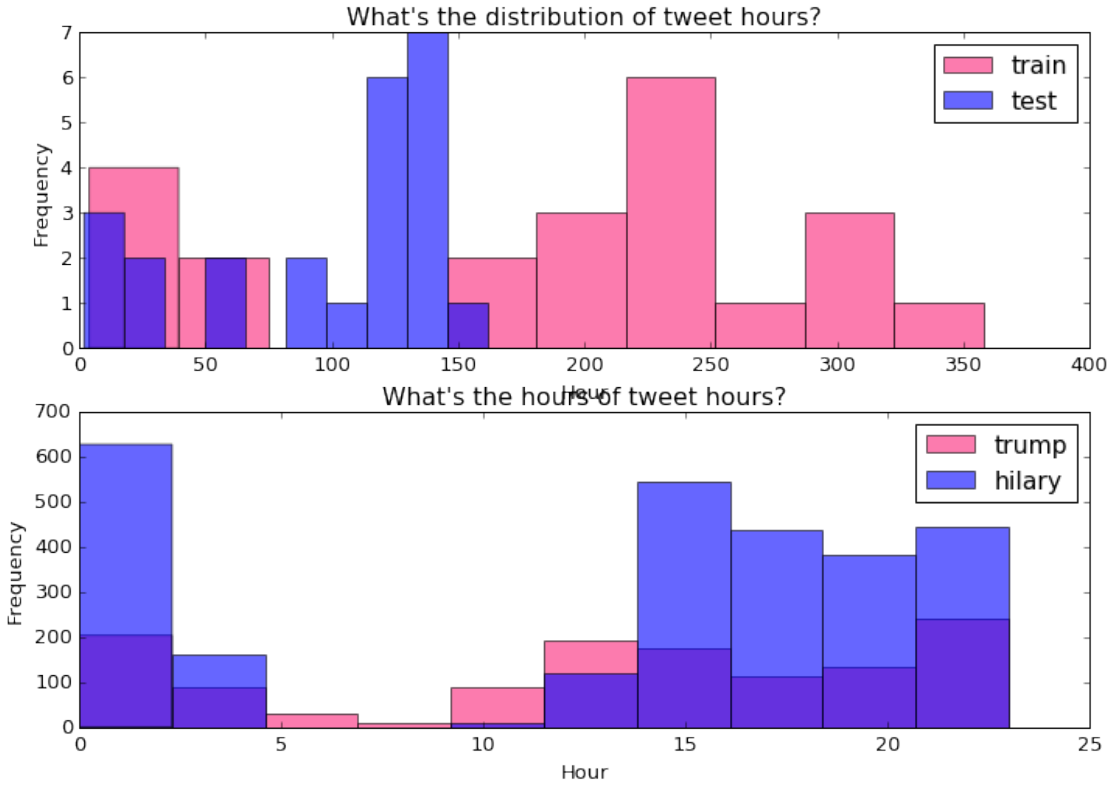
as both use different languages we will want a strong prior here to get over the bad distribution (missing values) of languages.

Now taking a look at time. time is like a float or a vary large range integer and so we will want to get a smaller range so it'll make sense.

```
In [88]: import datetime
         def their_time_to_p(time_s):
             return datetime.datetime.strptime(time_s, '%Y-%m-%dT%H:%M:%S')


         fixed_train = train.copy(False)
         fixed_test = test.copy(False)
         fixed_train['time'] = train['time'].map(lambda time_s: their_time_to_p(tim
         fixed_test['time'] = test['time'].map(lambda time_s: their_time_to_p(time_

         plt.rc('font', size=13)
         fig = plt.figure(figsize=(18, 8))
         alpha = 0.6

         ax1 = plt.subplot2grid((2,3), (0,0), colspan=2)
```

```
fixed_train['time'].value_counts().plot(kind='hist', color='#FA2379', labe
fixed_test['time'].value_counts().plot(kind='hist', label='test', alpha=al
ax1.set_xlabel('Hour')
ax1.set_title("What's the distribution of tweet hours?" )
plt.legend(loc='best')

fixed_trump = trump.copy(False)
fixed_hilary = hilary.copy(False)

fixed_trump['time'] = trump['time'].map(lambda time_s: their_time_to_p(tim
fixed_hilary['time'] = hilary['time'].map(lambda time_s: their_time_to_p(t

ax2 = plt.subplot2grid((2,3), (1,0), colspan=2)
fixed_trump['time'].plot(kind='hist', color='#FA2379', label='trump', alph
fixed_hilary['time'].plot(kind='hist', label='hilary', alpha=alpha)
ax2.set_xlabel('Hour')
ax2.set_title("What's the hours of tweet hours?" )
plt.legend(loc='best')
```

Out[88]: <matplotlib.legend.Legend at 0x7f9c683f5f28>




Seeing this we will have a huge bias towards trump. Really huge. It might be smart to ignore this or use a huge prior.