Project1_online_news_popularity_prediction

November 2, 2015

0.1 News popularity data set

We assume that we have been asked by a local newspaper to build an ML system to predict the popularity of their online news articles. Their goal is to use this information to select how to present articles and sell advertisement.

In [41]:
```
import pandas as pd
import matplotlib.pyplot as plt

%matplotlib inline

path = r'OnlineNewsPopularity.csv'
data = pd.read_csv(path, usecols=list(range(2, 61)))
data.head()
```

Out[41]:
```
   n_tokens_title  n_tokens_content  n_unique_tokens  n_non_stop_words  \
0              12               219         0.663594                 1
1               9               255         0.604743                 1
2               9               211         0.575130                 1
3               9               531         0.503788                 1
4              13              1072         0.415646                 1

   n_non_stop_unique_tokens  num_hrefs  num_self_hrefs  num_imgs  num_videos  \
0                  0.815385          4               2         1           0
1                  0.791946          3               1         1           0
2                  0.663866          3               1         1           0
3                  0.665635          9               0         1           0
4                  0.540890         19              19        20           0

   average_token_length   ...    min_positive_polarity  max_positive_polarity  \
0              4.680365   ...                 0.100000                    0.7
1              4.913725   ...                 0.033333                    0.7
2              4.393365   ...                 0.100000                    1.0
3              4.404896   ...                 0.136364                    0.8
4              4.682836   ...                 0.033333                    1.0

   avg_negative_polarity  min_negative_polarity  max_negative_polarity  \
0              -0.350000                 -0.600              -0.200000
1              -0.118750                 -0.125              -0.100000
2              -0.466667                 -0.800              -0.133333
3              -0.369697                 -0.600              -0.166667
4              -0.220192                 -0.500              -0.050000

   title_subjectivity  title_sentiment_polarity  abs_title_subjectivity  \
```

```
0           0.500000                  -0.187500              0.000000
1           0.000000                   0.000000              0.500000
2           0.000000                   0.000000              0.500000
3           0.000000                   0.000000              0.500000
4           0.454545                   0.136364              0.045455

   abs_title_sentiment_polarity  shares
0                      0.187500     593
1                      0.000000     711
2                      0.000000    1500
3                      0.000000    1200
4                      0.136364     505

[5 rows x 59 columns]
```

We have 61 attribiutes(58 predictive attributes, 2 nonpredictive, 1 goal field) (Data set description):

0. url: URL of the article (nonpredictive)
1. timedelta: Days between the article publication and the dataset acquisition (nonpredictive)
2. n_tokens_title: Number of words in the title
3. n_tokens_content: Number of words in the content
4. n_unique_tokens: Rate of unique words in the content
5. n_non_stop_words: Rate of nonstop words in the content
6. n_non_stop_unique_tokens: Rate of unique nonstop words in the content
7. num_hrefs: Number of links
8. num_self_hrefs: Number of links to other articles published by Mashable
9. num_imgs: Number of images
10. num_videos: Number of videos
11. average_token_length: Average length of the words in the content
12. num_keywords: Number of keywords in the metadata
13. data_channel_is_lifestyle: Is data channel 'Lifestyle'?
14. data_channel_is_entertainment: Is data channel 'Entertainment'?
15. data_channel_is_bus: Is data channel 'Business'?
16. data_channel_is_socmed: Is data channel 'Social Media'?
17. data_channel_is_tech: Is data channel 'Tech'?
18. data_channel_is_world: Is data channel 'World'?
19. kw_min_min: Worst keyword (min. shares)
20. kw_max_min: Worst keyword (max. shares)
21. kw_avg_min: Worst keyword (avg. shares)
22. kw_min_max: Best keyword (min. shares)
23. kw_max_max: Best keyword (max. shares)
24. kw_avg_max: Best keyword (avg. shares)
25. kw_min_avg: Avg. keyword (min. shares)
26. kw_max_avg: Avg. keyword (max. shares)
27. kw_avg_avg: Avg. keyword (avg. shares)
28. self_reference_min_shares: Min. shares of referenced articles in Mashable
29. self_reference_max_shares: Max. shares of referenced articles in Mashable
30. self_reference_avg_sharess: Avg. shares of referenced articles in Mashable
31. weekday_is_monday: Was the article published on a Monday?
32. weekday_is_tuesday: Was the article published on a Tuesday?
33. weekday_is_wednesday: Was the article published on a Wednesday?
34. weekday_is_thursday: Was the article published on a Thursday?
35. weekday_is_friday: Was the article published on a Friday?
36. weekday_is_saturday: Was the article published on a Saturday?
37. weekday_is_sunday: Was the article published on a Sunday?

38. is_weekend: Was the article published on the weekend?
39. LDA_00: Closeness to LDA topic 0
40. LDA_01: Closeness to LDA topic 1
41. LDA_02: Closeness to LDA topic 2
42. LDA_03: Closeness to LDA topic 3
43. LDA_04: Closeness to LDA topic 4
44. global_subjectivity: Text subjectivity
45. global_sentiment_polarity: Text sentiment polarity
46. global_rate_positive_words: Rate of positive words in the content
47. global_rate_negative_words: Rate of negative words in the content
48. rate_positive_words: Rate of positive words among nonneutral tokens
49. rate_negative_words: Rate of negative words among nonneutral tokens
50. avg_positive_polarity: Avg. polarity of positive words
51. min_positive_polarity: Min. polarity of positive words
52. max_positive_polarity: Max. polarity of positive words
53. avg_negative_polarity: Avg. polarity of negative words
54. min_negative_polarity: Min. polarity of negative words
55. max_negative_polarity: Max. polarity of negative words
56. title_subjectivity: Title subjectivity
57. title_sentiment_polarity: Title polarity
58. abs_title_subjectivity: Absolute subjectivity level
59. abs_title_sentiment_polarity: Absolute polarity level
60. shares: Number of shares (target)

URL and timedelta columns has been ommited when loading the CSV file due to the fact that they can not be considered as features.

What are we looking for? - Number of shares (attribiute number 60)

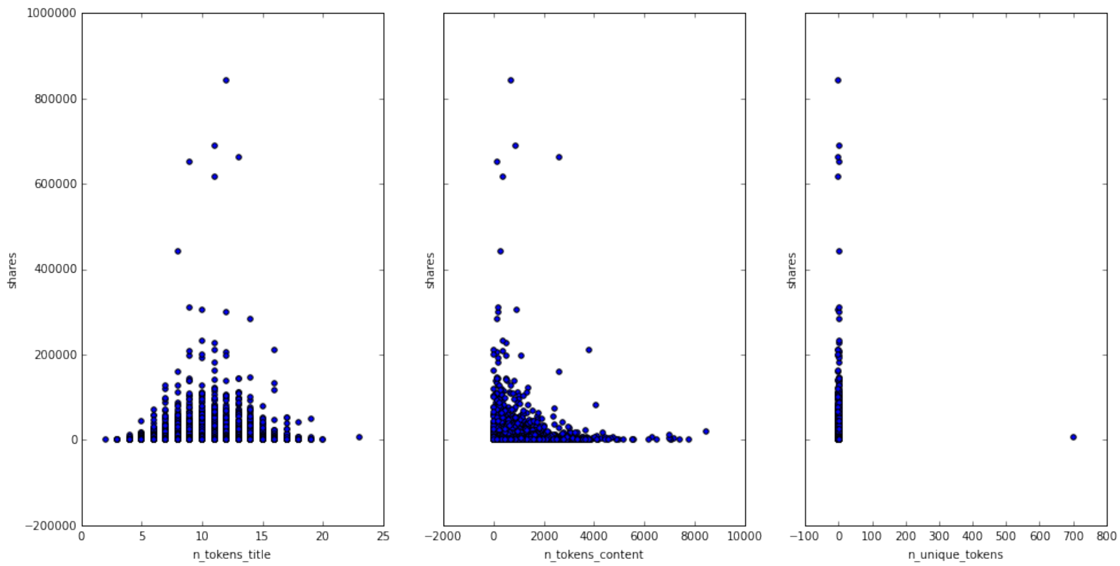
In [5]:
```
# print the shape of the DataFrame
data.shape
```

Out[5]:
```
(39644, 61)
```

There are 39644 **observations**, and thus 39644 samples in the dataset. Now lets take a look at the data and see the relationship between couple of featrues and the number of shares using scatter plots.

In [42]:
```
fig, axs = plt.subplots(1, 3, sharey=True)
data.plot(kind='scatter', x='n_tokens_title', y='shares', ax=axs[0], figsize=(16, 8))
data.plot(kind='scatter', x='n_tokens_content', y='shares', ax=axs[1])
data.plot(kind='scatter', x='n_unique_tokens', y='shares', ax=axs[2])
```

Out[42]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a0f0d3c10>
```



Let's use **Statsmodels** to estimate the model coefficients for the advertising data:

```
In [61]: import statsmodels.formula.api as smf
         lm = smf.ols(formula='shares ~ n_tokens_title + n_tokens_content + n_unique_tokens + n_non_stop

         # print the coefficients
         lm.params
```

```
Out[61]: Intercept                          -149956.308986
         n_tokens_title                          89.858979
         n_tokens_content                         0.593618
         n_unique_tokens                       3985.330237
         n_non_stop_words                     -1483.571598
         n_non_stop_unique_tokens             -1640.510010
         num_hrefs                               26.541264
         num_self_hrefs                         -57.643727
         num_imgs                                11.897267
         num_videos                               5.644627
         average_token_length                  -586.728729
         num_keywords                            49.493780
         data_channel_is_lifestyle            -1050.027477
         data_channel_is_entertainment        -1180.498442
         data_channel_is_bus                   -802.320002
         data_channel_is_socmed                -602.940931
         data_channel_is_tech                  -550.945215
         data_channel_is_world                 -483.077555
         kw_min_min                               2.208830
         kw_max_min                               0.087176
         kw_avg_min                              -0.346792
         kw_min_max                              -0.002067
         kw_max_max                              -0.000515
         kw_avg_max                              -0.000719
         kw_min_avg                              -0.365938
         kw_max_avg                              -0.202627
```

```
kw_avg_avg                           1.662482
self_reference_min_shares            0.026155
self_reference_max_shares            0.005762
self_reference_avg_sharess          -0.005779
weekday_is_monday               -26106.874558
weekday_is_tuesday              -26645.714671
weekday_is_wednesday            -26479.660908
weekday_is_thursday             -26656.096541
weekday_is_friday               -26618.024734
weekday_is_saturday              -8532.685213
weekday_is_sunday                -8917.257133
is_weekend                      -17449.942344
LDA_00                          176224.886563
LDA_01                          175361.655431
LDA_02                          174959.955235
LDA_03                          175783.404458
LDA_04                          175777.680137
global_subjectivity               2470.459851
global_sentiment_polarity          678.935329
global_rate_positive_words      -13429.504529
global_rate_negative_words        2097.118545
rate_positive_words               2117.277821
rate_negative_words               2002.836102
avg_positive_polarity            -1614.006468
min_positive_polarity            -1964.576603
max_positive_polarity              349.150153
avg_negative_polarity            -1723.359814
min_negative_polarity              129.493717
max_negative_polarity             -182.766895
title_subjectivity                -100.190367
title_sentiment_polarity           212.812119
abs_title_subjectivity             644.503787
abs_title_sentiment_polarity       610.847555
dtype: float64
```

Which means, for a given amount of n_tokens_content and n_unique_tokens, we will see 48 share increase with increasing the n_tokens_title by one.

A lot of the information we have been reviewing piece-by-piece is available in the model summary output:

```
In [73]: lm.pvalues
```

```
Out[73]: Intercept                        3.603446e-02
         n_tokens_title                   9.275381e-03
         num_hrefs                        2.712460e-10
         num_self_hrefs                   4.632086e-02
         average_token_length             1.223637e-06
         data_channel_is_entertainment    4.131922e-05
         kw_min_max                       8.780387e-03
         kw_min_avg                       6.907790e-10
         kw_max_avg                       1.353095e-23
         kw_avg_avg                       1.670263e-64
         self_reference_min_shares        1.004274e-18
         global_subjectivity              6.007813e-08
         dtype: float64
```

Now if we only choose the features with p-values lower than 0.05, we get to the same result with reducing the feature space and thus complexity.

In [72]:
```
import statsmodels.formula.api as smf
lm = smf.ols(formula='shares ~ n_tokens_title + num_hrefs + num_self_hrefs + average_token_leng

# print the coefficients
lm.params
```

Out[72]:
```
Intercept                         -1068.918601
n_tokens_title                       72.038303
num_hrefs                            36.534433
num_self_hrefs                      -32.798184
average_token_length               -440.093473
data_channel_is_entertainment      -627.955151
kw_min_max                           -0.002800
kw_min_avg                           -0.430381
kw_max_avg                           -0.195783
kw_avg_avg                            1.761833
self_reference_min_shares             0.026095
global_subjectivity                3465.061130
dtype: float64
```

In [74]:
```
# print a summary of the fitted model
lm.summary()
```

Out[74]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 shares   R-squared:                       0.020
Model:                            OLS   Adj. R-squared:                  0.020
Method:                 Least Squares   F-statistic:                     73.96
Date:                Sun, 01 Nov 2015   Prob (F-statistic):          1.30e-165
Time:                        17:59:50   Log-Likelihood:            -4.2696e+05
No. Observations:               39644   AIC:                         8.539e+05
Df Residuals:                   39632   BIC:                         8.540e+05
Df Model:                          11
Covariance Type:            nonrobust
=================================================================================================
                                    coef    std err          t      P>|t|      [95.0% Conf. In
-------------------------------------------------------------------------------------------------
Intercept                     -1068.9186    509.832     -2.097      0.036     -2068.201    -69.
n_tokens_title                   72.0383     27.687      2.602      0.009        17.771    126.3
num_hrefs                        36.5344      5.784      6.316      0.000        25.197     47.8
num_self_hrefs                  -32.7982     16.461     -1.993      0.046       -65.061     -0.5
average_token_length           -440.0935     90.694     -4.853      0.000      -617.855   -262.3
data_channel_is_entertainment  -627.9552    153.143     -4.100      0.000      -928.120   -327.7
kw_min_max                       -0.0028      0.001     -2.621      0.009        -0.005     -0.0
kw_min_avg                       -0.4304      0.070     -6.170      0.000        -0.567     -0.2
kw_max_avg                       -0.1958      0.020    -10.018      0.000        -0.234     -0.1
kw_avg_avg                        1.7618      0.104     16.989      0.000         1.559      1.9
self_reference_min_shares         0.0261      0.003      8.839      0.000         0.020      0.0
global_subjectivity            3465.0611    639.357      5.420      0.000      2211.907   4718.2
==============================================================================
```

```
Omnibus:                   108726.927   Durbin-Watson:                   1.993
Prob(Omnibus):                  0.000   Jarque-Bera (JB):       5848284629.554
Skew:                          34.471   Prob(JB):                         0.00
Kurtosis:                    1883.353   Cond. No.                     6.62e+05
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 6.62e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

Explanations:

- We have the 95% **confidence interval**. It means, out of 100 samples, 95 of them will have a coefficient that will be in this interval.
- The coefficient of each feature shows its **p-value**. It expalains how much the feature influence the number of shares.
- This model has a **R-squared** of 0.023 which is very low. R-squared is the proportion of variance explained, meaning the proportion of variance in the observed data that is explained by the model, or the reduction in error over the null model. (The null model just predicts the mean of the observed response, and thus it has an intercept and no slope.) R-squared is between 0 and 1, and higher is better because it means that more variance is explained by the model.

0.2 Applying Linear Regression in scikit-learn

Let's redo some of the Statsmodels code above in scikit-learn:

```
In [77]: feature_cols = ['n_tokens_title' , 'n_tokens_content' , 'n_unique_tokens' , 'n_non_stop_words'

         X = data[feature_cols]
         y = data.shares

         from sklearn.linear_model import LinearRegression
         lm = LinearRegression()
         lm.fit(X, y)

         print lm.intercept_
         print lm.coef_
```

```
-176397.245028
[  8.98589790e+01   5.93617818e-01   3.98532973e+03  -1.48359263e+03
  -1.64050950e+03   2.65412628e+01  -5.76437259e+01   1.18972675e+01
   5.64462711e+00  -5.86728561e+02   4.94937782e+01  -1.05002747e+03
  -1.18049843e+03  -8.02319986e+02  -6.02940908e+02  -5.50945202e+02
  -4.83077566e+02   2.20882975e+00   8.71760840e-02  -3.46792231e-01
  -2.06703600e-03  -5.15266881e-04  -7.18603840e-04  -3.65937523e-01
  -2.02627041e-01   1.66248246e+00   2.61546692e-02   5.76183822e-03
  -5.77889578e-03   3.56004342e+02  -1.82835768e+02  -1.67820124e+01
  -1.93217645e+02  -1.55145834e+02   2.88274420e+02  -9.62975007e+01
   1.91976919e+02   1.76202944e+05   1.75339713e+05   1.74938012e+05
   1.75761462e+05   1.75755737e+05   2.47045999e+03   6.78935328e+02
  -1.34295045e+04   2.09711886e+03   2.11729787e+03   2.00285615e+03
  -1.61400652e+03  -1.96457653e+03   3.49150156e+02  -1.72335980e+03
```

```
   1.29493727e+02  -1.82766921e+02  -1.00190375e+02   2.12812119e+02
   6.44503788e+02   6.10847558e+02]
```

```
In [76]: lm.score(X, y)
```

```
Out[76]: 0.023098806869788824
```

```
In [80]: import numpy as np
         from math import sqrt

         print("Residual sum of squares: %.2f"
               % np.mean((lm.predict(X) - y) ** 2))

         print("Root square error : %.2f"
               % sqrt(np.mean((lm.predict(X) - y) ** 2)))
```

```
Residual sum of squares: 132060017.55
Root square error : 11491.74
```

This is the error for our training samples. In order to see the generalized model, in the next step we devide the dataset in to two sets of traing and test and compute the error on the test set.

```
In [81]: import matplotlib.pyplot as plt
         import numpy as np
         from sklearn import datasets, linear_model
         import csv
         import pylab as pl
         from math import sqrt

         path = r'OnlineNewsPopularity.csv'
         data = np.loadtxt(open(path,"rb"),delimiter=",", skiprows=1,usecols= list(range(1, 61)))

         X1 = data[:,0:59]
         Y = data[:,59]

         m = Y.size
         X = np.ones(shape=(m, 60))
         X[:, 1:60] = X1

         indices = np.random.permutation(X.shape[0])
         training_idx, test_idx = indices[:13214], indices[13214:]
         X_training, X_test = X[training_idx,:], X[test_idx,:]
         Y_training, Y_test = Y[training_idx], Y[test_idx]

         regr = linear_model.LinearRegression()
         regr.fit(X_training, Y_training)

         print("Residual sum of squares: %.2f"
               % np.mean((regr.predict(X_test) - Y_test) ** 2))

         print("Root square error : %.2f"
               % sqrt(np.mean((regr.predict(X_test) - Y_test) ** 2)))
```

```
Residual sum of squares: 268740114.83
Root square error : 16393.29
```