Finding the formula for Popularity

POPULARITY PREDICTION USING MASHABLE ARTICLES FROM 2011-2012

Early success - Growth - 52/1

- Mashable reached 2 million readers in its first 18 months.
- Today Mashable has over 25 million visitors.
- Part of its success is because Mashable has gone beyond the 80/20 rule to reach the 52/1 rule, where 1% of its posts drives 52% of the traffic. In this scenario, know what is the recipe of the 1% is critical to Mashable's business objectives.

Data

- It has 61 variables
- It has about 40,000 articles published in 2013-2014
- Variables:
  - Day of the week
  - Number of words in the title
  - Number of words in the content
  - Links
  - Videos
  - Photos

Variables

```
In [48]: feature_cols = ['n_tokens_title', 'n_tokens_content', 'num_keywords', 'num_imgs', 'num_videos', 'kw_avg_avg', 'weekday_

         X = mashable[feature_cols]
         y = mashable.total
```

```
In [50]: X.dtypes
```

```
Out[50]: n_tokens_title          float64
         n_tokens_content        float64
         num_imgs                float64
         num_videos              float64
         kw_avg_avg              float64
         weekday_is_monday       float64
         weekday_is_tuesday      float64
         weekday_is_wednesday    float64
         weekday_is_thursday     float64
         weekday_is_friday       float64
         weekday_is_saturday     float64
         dtype: object
```

Classification Problem

- Total = Dummy variable
- >= 10.000 shares ==1, otherwise 0
- Only 2108 articles were really popular

Weak correlations

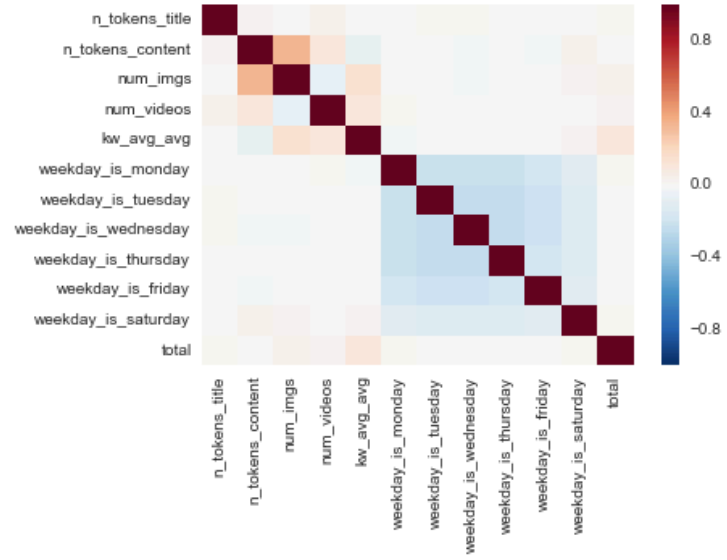
click to expand output; double click to hide output

Out[73]:

	n_tokens_title	n_tokens_content	num_imgs	num_videos	kw_avg_avg	weekday_is_monday	weekday_is_tuesday	week
n_tokens_title	1.000000	0.018160	-0.008858	0.051460	0.004296	0.004274	0.009322	0.008
n_tokens_content	0.018160	1.000000	0.342600	0.103699	-0.079624	-0.002484	-0.004027	-0.01
num_imgs	-0.008858	0.342600	1.000000	-0.067336	0.145236	-0.005249	-0.003767	-0.02
num_videos	0.051460	0.103699	-0.067336	1.000000	0.106815	0.009453	0.006740	-0.00
kw_avg_avg	0.004296	-0.079624	0.145236	0.106815	1.000000	-0.020546	-0.003915	-0.01
weekday_is_monday	0.004274	-0.002484	-0.005249	0.009453	-0.020546	1.000000	-0.215107	-0.21
weekday_is_tuesday	0.009322	-0.004027	-0.003767	0.006740	-0.003915	-0.215107	1.000000	-0.22
weekday_is_wednesday	0.008935	-0.016891	-0.024674	-0.001397	-0.014054	-0.215912	-0.229976	1.000
weekday_is_thursday	-0.015472	-0.007395	-0.005824	-0.003663	-0.004056	-0.212904	-0.226772	-0.22
weekday_is_friday	-0.002015	-0.015949	-0.007537	0.003473	0.003330	-0.184173	-0.196169	-0.19
weekday_is_saturday	-0.015013	0.034538	0.028970	-0.007135	0.031116	-0.115413	-0.122931	-0.12
total	0.008783	0.002459	0.039388	0.023936	0.110413	0.009726	-0.007941	-0.00

Heat map

```
In [74]: sns.heatmap(Xy.corr())
Out[74]: <matplotlib.axes._subplots.AxesSubplot at 0x116bd9cd0>
```



```
In [26]: pd.DataFrame({'feature':feature_cols,
'importance':treeclf.feature_importances_})
Out[26]:
        feature  importance
0         title    0.000000
1       content    0.000000
2      num_imgs    0.087024
3    num_videos    0.037054
4    kw_avg_avg    0.875922
5        monday    0.000000
6       tuesday    0.000000
7     wednesday    0.000000
8      thursday    0.000000
9        friday    0.000000
10     saturday    0.000000
11       sunday    0.000000
```

Logistic regression

- Accuracy score:
- **0.94**
- Sensitivity: 0.008
- Specificity: 0.99
- AUC:0.67

Next Steps

- Get the true values
- kNN
- Create new variables
- Random Forest