Project 6: What's in a Name?

Due to 10/7/2011

Chong Ding

How to generate the attributes?

I use Ruby to write a script which generates some attributes that might help to find the pattern in the lists. The attributes that can be extracted from names are listed as below:

1. Initial character of first name
2. Second character of first name
3. Third character of first name
4. Fourth character of first name
5. Fifth character of first name
6. Length of full name (ignore middle name)
7. Length of first name
8. Length of last name
9. Ratio of length of first name to length of full name
10. Ratio of length of last name to length of full name
11. Boolean Variable (if there is a middle name or not)
12. Boolean Variable (if first name is shorter than last name)
13. Boolean Variable (if there is '-' in the first name)
14. Boolean Variable (if there is '-' in the last name)

The simple and messy Ruby code can be found in the same directory with where this report is.

Content in the output file is formatted, so it's very easy to import them into Excel. The reason I use Excel to explore the pattern is that data can be easily sorted and filtered in Excel. It's much straight forward to see the result after sorting or filtering which help to recognize the pattern if it shows.

How to find the pattern?

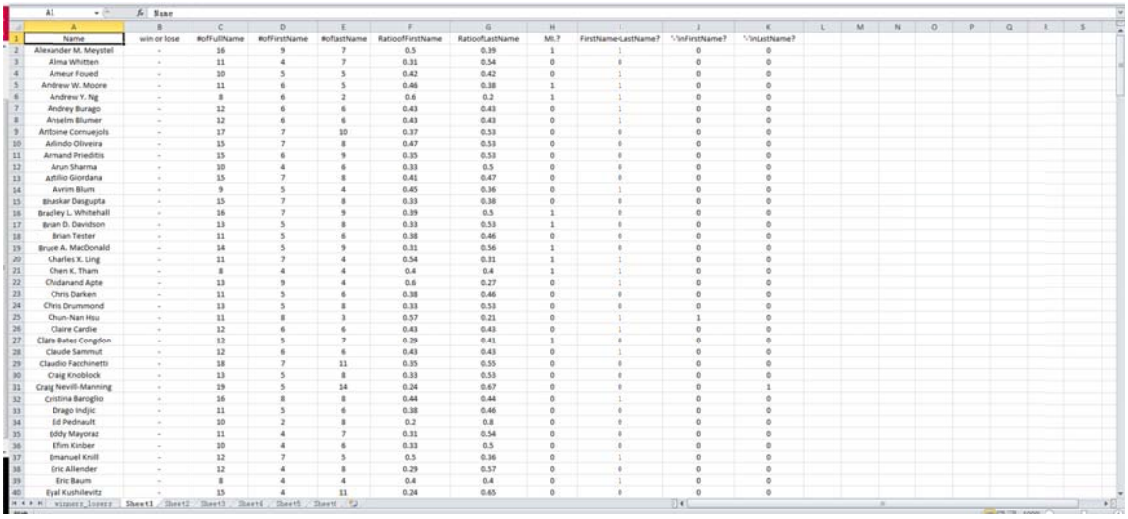
First try: analyze the numeric attributes.

From a snapshot 1, you can see how those numeric attributes are in the Excel. Firstly, I sorted the whole table by following the class attribute "+" or "-". You can find it is split into two groups. Then I find the range of attribute 6 to attribute 10, and count the number different Boolean value in two different groups. Result can be seen from table 1. We didn't see too much variance between each attribute in different groups which are sorted by the class that it belongs to. There are too many overlap between these two groups. Only the distinct attribute is useful to find the pattern or at least the starting. But, some weak pattern we can extract from this numeric attributes is that the areas that is not an overlap between two classes can be used to classify the names (e.g.

LengthofFirstName>9 or RatioofFirstName>0.73 or RatioofLastName<0.2 or if FirstName includes “-” can be used to find some names in group “+”), but accuracy is not high enough.

"+" or "-"	Lengthof FullName	Length ofFirst Name	Lengtho fLastNa me	RatioofF irstNam e	Ratioof LastNa me	MI.?	FirstName< LastName?	'- 'inFirstN ame?	'- 'inLastN ame?
"+"	8 to 24	2 to 9	2 to 16	0.125 to 0.73	0.2 to 0.8	#0: 61 #1: 24	#0:54 #1: 31	#0:84 #1: 1	#0:83 #1: 2
"-"	6 to 20	3 to 11	2 to 15	0.216 to 0.82	0.166-- 0.7897 74	#0:166 #1: 47	#0:119 #1: 94	#0:207 #1: 6	#0:212 #1: 1

Table 1



	A	B	C	D	E	F	G	H	I	J	K
1	Name	win or lose	#ofFullName	#ofFirstName	#oflastName	RatioofFirstName	RatioofLastName	MI.?	FirstName<LastName?	'-'inFirstName?	'-'inLastName?
2	Alexander M. Meystel	-	16	9	7	0.5	0.39	1	1	0	0
3	Alma Whitten	-	11	4	7	0.31	0.54	0	0	0	0
4	Ameur Foued	-	10	5	5	0.42	0.42	0	1	0	0
5	Andrew W. Moore	-	11	6	5	0.46	0.38	1	1	0	0
6	Andrew Y. Ng	-	8	6	2	0.6	0.2	1	1	0	0
7	Andrey Burago	-	12	6	6	0.43	0.43	0	1	0	0
8	Anselm Blumer	-	12	6	6	0.43	0.43	0	1	0	0
9	Antoine Cornuejols	-	17	7	10	0.37	0.53	0	0	0	0
10	Arlindo Oliveira	-	15	7	8	0.47	0.53	0	0	0	0
11	Armand Prieditis	-	15	6	9	0.35	0.53	0	0	0	0
12	Arun Sharma	-	10	4	6	0.33	0.5	0	0	0	0
13	Attilio Giordana	-	15	7	8	0.41	0.47	0	0	0	0
14	Avrim Blum	-	9	5	4	0.45	0.36	0	1	0	0
15	Bhaskar Dasgupta	-	15	7	8	0.33	0.38	0	0	0	0
16	Bradley L. Whitehall	-	16	7	9	0.39	0.5	1	0	0	0
17	Brian D. Davidson	-	13	5	8	0.33	0.53	1	0	0	0
18	Brian Tester	-	11	5	6	0.38	0.46	0	0	0	0
19	Bruce A. MacDonald	-	14	5	9	0.31	0.56	1	0	0	0
20	Charles X. Ling	-	11	7	4	0.54	0.31	1	1	0	0
21	Chen K. Tham	-	8	4	4	0.4	0.4	1	1	0	0
22	Chidanand Apte	-	13	9	4	0.6	0.27	0	1	0	0
23	Chris Darken	-	11	5	6	0.38	0.46	0	0	0	0
24	Chris Drummond	-	13	5	8	0.33	0.53	0	0	0	0
25	Chun-Nan Hsu	-	11	8	3	0.57	0.21	0	1	1	0
26	Claire Cardie	-	12	6	6	0.43	0.43	0	1	0	0
27	Clare Bates Congdon	-	12	5	7	0.29	0.41	1	1	0	0
28	Claude Sammut	-	12	6	6	0.43	0.43	0	1	0	0
29	Claudio Facchinetti	-	18	7	11	0.35	0.55	0	0	0	0
30	Craig Knoblock	-	13	5	8	0.33	0.53	0	0	0	0
31	Craig Nevill-Manning	-	19	5	14	0.24	0.67	0	0	0	1
32	Cristina Baroglio	-	16	8	8	0.44	0.44	0	1	0	0
33	Drago Indjic	-	11	5	6	0.38	0.46	0	0	0	0
34	Ed Pednault	-	10	2	8	0.2	0.8	0	0	0	0
35	Eddy Mayoraz	-	11	4	7	0.31	0.54	0	0	0	0
36	Efim Kinber	-	10	4	6	0.33	0.5	0	0	0	0
37	Emanuel Knill	-	12	7	5	0.5	0.36	0	1	0	0
38	Eric Allender	-	12	4	8	0.29	0.57	0	0	0	0
39	Eric Baum	-	8	4	4	0.4	0.4	0	1	0	0
40	Eyal Kushilevitz	-	15	4	11	0.24	0.65	0	0	0	0

Snapshot 1

Second try:

After trying those numeric attributes, I tried the alphabetic attributes. Firstly, I did the same thing to split all the names with attributes into two groups by their class attribute “+” or “-”. Then I count the number of different initial letters appearing in their fist name and last name. The result is show in the table 2. Sub-table (a) is the statistics of first name’s initial letter, sub-table (b) is the statistics of last name’s initial letter. We can find some distinct from sub-table (a), such as the names whose first name’s initial letter is H, V, X or Y belong to group “+”, and the majority of the names whose first name’s initial letter is D, J, K, L, M, N or W belong to group “+” as well. However, they are not distinct enough and the number of names that start with these letters only 66.7% of the total names in the group “+”, which means if this is the pattern, the accuracy is only 66.7%. So we look further to the second letter in the first name.

First Init	loser	winner
A	13	3
B	5	6
C	13	4
D	1	25

Last Init	loser	winner
A	3	8
B	7	17
C	6	11
D	6	8

E	7	1
F	1	3
G	2	7
H	0	8
I	1	0
J	3	29
K	2	8
L	2	11
M	2	33
N	0	10
O	2	0
P	4	10
Q	0	0
R	2	18
S	16	9
T	6	10
U	1	0
V	0	2
W	1	10
X	0	1
Y	0	5
Z	1	0

(a)

E	1	3
F	3	12
G	1	18
H	1	10
I	1	3
J	1	7
K	9	16
L	3	9
M	14	15
N	3	2
O	1	4
P	3	11
Q	1	0
R	3	15
S	7	28
T	4	1
U	1	2
V	1	2
W	4	5
X	0	0
Y	0	3
Z	1	3

(b)

Table 2

Table 3 shows the statistical result of second letter in first name.

FirstName 2nd	loser	winner
a	0	85
b	0	0
c	1	0
d	3	0
e	0	28
f	1	0
g	0	0
h	19	0
i	1	37
j	9	0
k	0	0
l	9	0
m	2	0
n	5	1

o	0	49
p	0	0
q	0	0
r	22	0
s	1	0
t	11	0
u	0	11
v	1	0
w	0	1
x	0	0
y	4	0
z	0	0
.	5	0

Table 3

We are so happy to see this that all the winners has their first name's 2nd letter dropping in the small set of letters {a, e, i, n, o, u, w}. And the fortunate thing is, the losers' first name's 2nd letter drops nothing in this set. One exception is the letter n. But, it don't affect too much on the accuracy. Yes, I think I find the pattern!!!!!!!!!!!!!!

Conclusion: if the 2nd letter of first name is in the set {a, e, i, o, u, w}, it's in the winner's side, otherwise, the name stands on the opposite side.