Solution to the Capital One Labs Coding Challenge by **Shengwei Wang(2046)**

1. Code Test Part 1: Model building on a synthetic dataset

MySQL is used for data preprocessing; matlab is used for model building and prediction. The executable codes are attached with instructions.

The dataset has 1 target and 254 features including 4 nominal features (f_61,f_121,f_215,f_237) and 250 numeric features. I replace the missing numeric values with 0s and nominal values with the extra class labels. I create binary features for the nominal values, for example:

```
if f_61 = 'a', then f_61_0 = 0, f_61_1 = 0, f_61_2 = 0;
if f_61 = 'b', then f_61_0 = 0, f_61_1 = 0, f_61_2 = 1;
if f_61 = 'c', then f_61_0 = 0, f_61_1 = 1, f_61_2 = 0;
if f_61 = 'd', then f_61_0 = 0, f_61_1 = 1, f_61_2 = 1;
if f_61 = 'e', then f_61_0 = 1, f_61_1 = 0, f_61_2 = 0;
if f_61 = the missing value, then f_61_0 = 1, f_61_1 = 0, f_61_2 = 1;
```

Binarizing the nominal features is for the machine learning algorithms to consume and **the nominal features shouldn't be ordinal.**
After this step there are 1 target column and 262 feature columns.

I exam the dataset and the target is in the interval of (-27,27); all other features are in the interval (-4,4).

I calculate the rank of the data matrix and find one feature column is redundant because the rank is less than the number of columns by 1. Because **the matrix does not have a full rank**, so I write several lines of code to detect the redundant column and delete it. After this step there are 1 target column and 261 feature columns. This is very important because for algorithms like linear regression, one of the assumptions is the feature matrix should have the full rank. The target and features have distributions very close to normal distribution according to their histograms.

I am not sure which algorithm performs better so I try neural network, linear regression, random forest(regression). **I have solid reasons to pick these 3 algorithms. Linear regression** is simple and fairly accurate but the predicted targets are not bounded (remember the target is between -27 to 27); **Random forest of regression trees** is good for cases there are lots of missing values because a record with missing values may not reach a particular tree leaf but many other records can reach this leaf and the regression is the aggregation of these records; **Neural network** is good for discovering nonlinear interactions. In the following part I describe the algorithms and report their performances one by one:

1) Neural network with 2 hidden layers with 10 neurons each. The transfer functions are the hyperbolic tangent sigmoid transfer functions. It is showed in Fig 1.

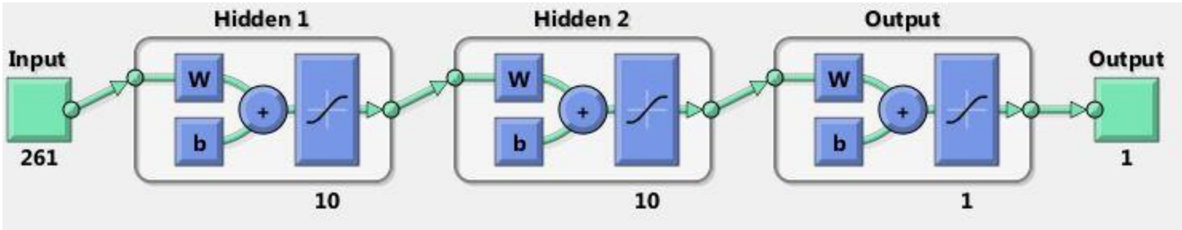

Fig 1: the neural network topology

The matlab modules (the neural network toolbox) include cross validation to prevent the over-fitting, the model is calculated by the back propagation method and **it is an online learning algorithm** which takes one instance per iteration, being different from the batch learning algorithm. This is good because there is no need to load the whole data set into the memory in case the data set is too large. One round in which every instance is used for update is an **epoch**. The cross validation chooses the best number of epoch, which is showed in Fig 2.

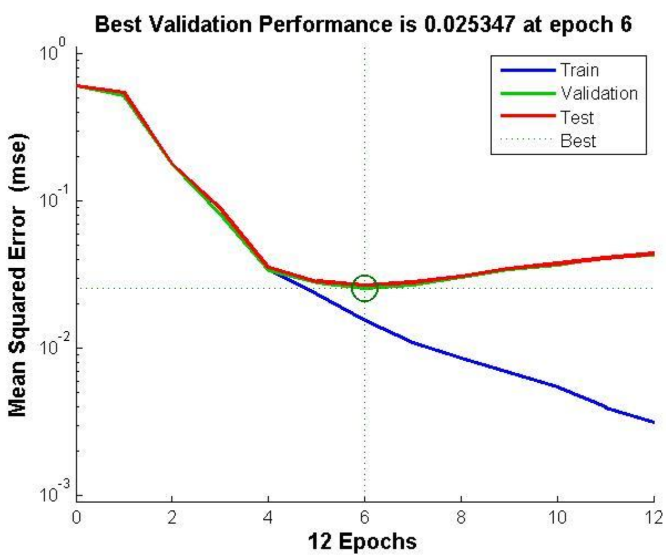

Fig 2: mean square errors v.s. epochs to choose the number of epochs

I split the data set into the 1st part which contains 4000 instances for training, cross validation and the 2nd part which contains 1000 instances for performance measurement. The performance's in Fig 3.

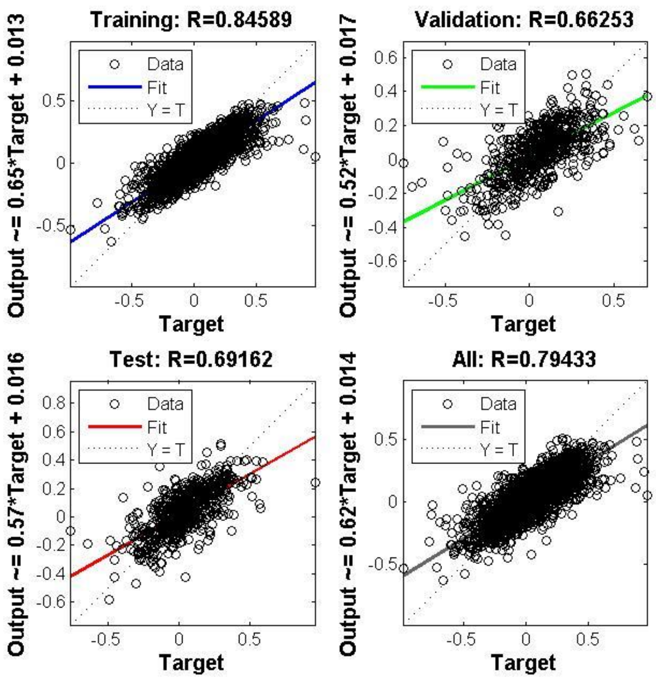

Fig 3: the performance measure: the correlation of the predicted and the target (truth). On the performance measurement set, it is **0.69162**

The mean square error is 17.9406049783251978 on the performance set.

2) Linear regression is a simple model with try to minimize the mean square error of modeling the target with a linear combination of the predictors. The model coefficients can be solved by the matrix iteration algorithm which is good for Hadoop MapReduce. Because there are no parameters to tune and the feature which makes the feature matrix not full ranked is already deleted, I don't perform the cross validation. The performance on the performance data set is in Fig 4.

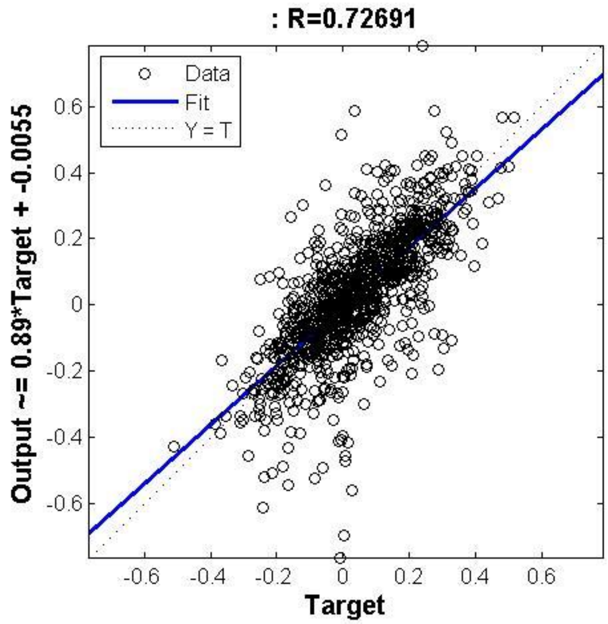

Fig 4 : the performance of the linear regression on the performance data set, **the correlation is 0.72691 between predicted and target**

The mean square error is 15.1466240581132792 on the performance set.

3) Random forest uses the bootstrap to randomly select a good number of subsets of instances to train different regression trees and each training only uses a small number of features to reduces the correlations of the regression trees. The result is the average of all the regression trees from the bootstrap. Because each regression tree is trained by a subset of instances and the training is independent from other trees' training so **it can be MapReduced: each node trains one regression tree independently.** This reduces the memory difficulty.

1. Draw ntree bootstrap samples from the original data.
2. For each of the bootstrap samples, grow an un-pruned classification or regression tree, with the following modification: at each node, rather than choosing the best split among all predictors, randomly sample mtry of the predictors and choose the best split from among those variables. (Bagging can be thought of as the special case of random forests obtained when
mtry = p, the number of predictors.)
3. Predict new data by aggregating the predictions of the ntree trees (i.e., majority votes for classification, average for regression).

I use 300 boostraps and 50% of the total features, which is 130. The result is showed in Fig 5.

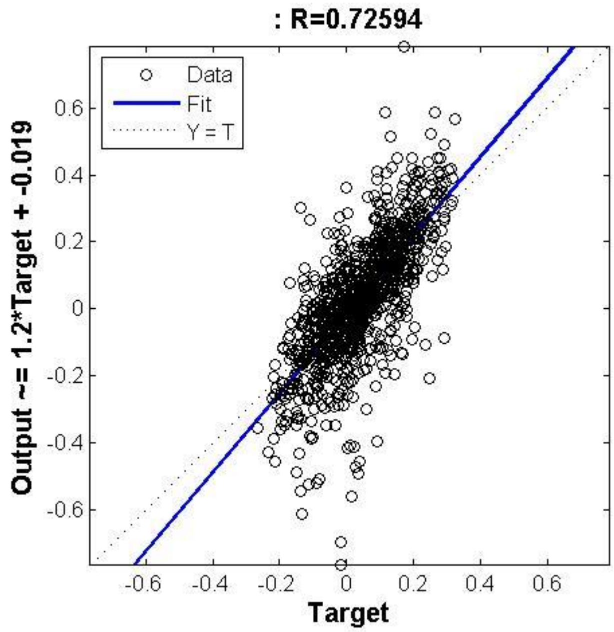

Fig 5: the performance of the random forest of the regression trees on the performance data set: **the correlation of predicted and target is 0.72594**

The mean square error is 11.1769873748467785 (the smallest amongst the 3 algorithms).

The predictions: I saved 3 arrays of predictions, each of size 1000, into the 'prediction.csv' for ols (linear regression model), random forest of regression trees and neural network. **If you ask me to pick one, I will take the result of random forest. Why the random forest is superior? It is good for datasets with lots of missing values and each regression tree can be trained on a separate node with the MapReduce model.** The weakness of random forest is it breaks with outliers and concept drift but the code challenge problem mentions it is generated from the same data generating process and I plot the histograms of the targets and features, they seem well behaved. **On this data set, random forest is better than linear regression which is better than neural network in terms of the mean square errors.**

Yes! Please take the result of the random forest and use the other two results as your reference when you find something strange going on.

The code instructions: If you want to replicate my results, you need to do these in order:
1. Run the 'sql_test_1.txt' script with your mySQL database. Of course you have to change the paths in the script to your values. This is for

the pre processing of the data like handling missing values, converting nominal features to binary features.
2. Run 'nn.m' with your matlab and its neural network toolbox. The prediction on the test data set is 'predict_nn'
3. Run 'forest.m' with your matlab and its ensemble learning toolbox. The prediction on the test data set is 'predict_forest'. This may take 10 to 30 minutes depending on your hardware configuration.
4. Run 'ols.m' with your matlab. The prediction on the test data set is 'predict_ols'.
5. You may see different predictions if you run it yourself. Don't panic: we are just using different random seed. Neural network and random forest both depend on some random number generators.

I suggest you take the prediction by random forest which is in 'best predict.csv'.

Warning: DO NOT try to merge the 3 matlab scripts into a big one and run. This is because the names of the functions and variables conflict. Matlab, like Python or other scripting languages, can't be built with a dependency file, which is different from languages like C, JAVA. It is also to prevent loading too much stuff into the memory in one time.

If you want to save the time and look into the raw matlab outputs. Yes. I save the workspace for each of the 3 algorithms into 'forest_result.mat', 'ols_result.mat' and 'nn_result.mat'. It is impossible to send them to you by email as they are large (50Mbs each) so I put them in my GitHub account. I upload everything, including the codes, images and the worksapce... into a folder under my GitHub account. This is the link:

https://github.com/givenwong/capitalone_1

2. Code Test Part 2: Baby Names!

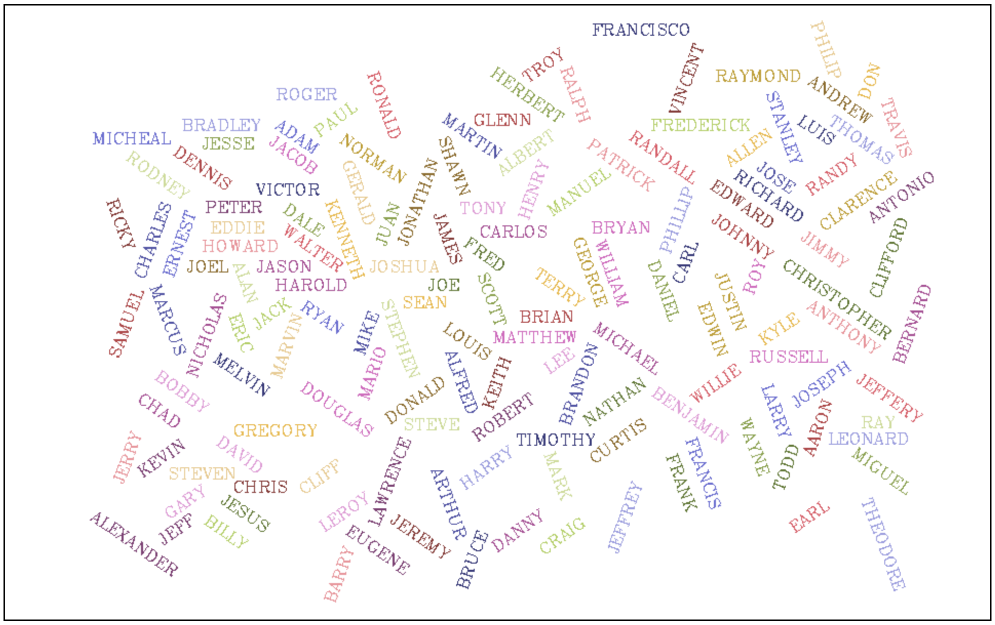

Fig 6: world cloud for the names. Generated by D3.js

Java is used to aggregate the by state data sets into one data set which contains data of all states. MySQL is used for data processing to get the most popular names and to calculate the gender ambiguities.

The dataset you provide (**http://www.ssa.gov/oact/babynames/state/namesbystate.zip**) has 51 txt files for each of the state. Each state data set has state, sex, year(from 1910 to 2013), name, frequency. I also download the country level data sets which have name, sex, frequency by year, from 1880 to 2013 (**http://www.ssa.gov/oact/babynames/names.zip**). I aggregate both the state level and country level data sets by Java. It is easy for MYSQL to process just one big table instead of many small tables.

A) Descriptive analysis

1. Please describe the format of the data files. Can you identify any limitations or distortions of the data?

The data files are all comma-separated flat ASCII txt files(state, sex, year, name, frequency) and each has the corresponding state as its name.
Limitations are:
1. ASCII is good for 256 characters and ASCII can't display some of the UTF-8 characters. This is bad for foreign names.
2. The data doesn't include those who don't have SSN.
3. The frequency which is below 5 is not counted to protect privacy.
4. "Social Security Numbers were first issued by the Social Security Administration in November 1935 as part of the New Deal Social Security program." from wiki but the data is from 1910 to 2013.
5. Data is not normalized. If you want to see the trend or dynamics of popularities from year to year, the absolute values is not as good as the portions which are normalized by the total population of that year. The population grows every year.

2. What is the most popular name of all time? (Of either gender.) use the nation's data and the state level data.

The country level: **the most popular male name is 'James' with 10182378; the most popular female name is 'Mary' with 8224928.**

The state level: **look into this table:**

state	name	freq_eachstate	sex
AK	Michael	8041	M
AL	Paul	15821	M
AR	Thomas	19014	M
AZ	Richard	20460	M
CA	Ethan	40331	M
CO	William	33190	M
CT	Paul	22613	M
DC	Thomas	14937	M
DE	John	15090	M
FL	Jacob	32100	M
GA	Raymond	12788	M
HI	Michael	13706	M
IA	Steven	25496	M
ID	Robert	16698	M
IL	Samuel	26631	M
IN	Gary	25152	M
KS	Donald	22404	M
KY	Ronald	18966	M
LA	Andrew	15909	M
MA	Philip	18093	M
MD	Paul	21629	M
ME	John	21840	M
MI	Lawrence	24235	M
MN	Jacob	20993	M
MO	Edward	24030	M

state	name	freq_eachstate	sex
WY	Robert	8975	M
WV	Richard	26289	M
WI	Gary	25362	M
WA	Richard	39892	M
VT	Robert	11637	M
VA	Larry	20036	M
UT	William	16769	M
TX	Caleb	21972	M
TN	Gary	19987	M
SD	James	15316	M
SC	Edward	16630	M
RI	Robert	28422	M
PA	Zachary	27095	M
OR	Richard	22967	M
OK	Donald	22580	M
OH	Samuel	27884	M
NY	Nathan	22198	M
NV	David	9612	M
NM	David	19907	M
NJ	Peter	36805	M
NH	Robert	19439	M
NE	John	33540	M
ND	James	15008	M
NC	Samuel	26304	M
MT	Robert	18602	M
MS	Joseph	21929	M

Fig 7: most popular name by state for all the time, male

state	name	freq_eachstate	sex
AK	Mary	3929	F
AL	Mary	115253	F
AR	Mary	58485	F
AZ	Mary	22713	F
CA	Jennifer	173755	F
CO	Mary	31531	F
CT	Mary	41609	F
DC	Mary	22322	F
DE	Mary	8402	F
FL	Mary	66066	F
GA	Mary	124255	F
HI	Mary	5769	F
IA	Mary	61690	F
ID	Mary	9877	F
IL	Mary	199701	F
IN	Mary	100427	F
KS	Mary	43012	F
KY	Mary	103042	F
LA	Mary	78273	F
MA	Mary	115994	F
MD	Mary	63733	F
ME	Mary	14572	F
MI	Mary	137028	F
MN	Mary	70587	F
MO	Mary	105337	F

state	name	freq_eachstate	sex
MS	Mary	86627	F
MT	Mary	12878	F
NC	Mary	132909	F
ND	Mary	13032	F
NE	Mary	30038	F
NH	Mary	8434	F
NJ	Mary	90209	F
NM	Mary	23739	F
NV	Jennifer	5927	F
NY	Mary	276451	F
OH	Mary	200619	F
OK	Mary	55728	F
OR	Mary	19681	F
PA	Mary	291350	F
RI	Mary	16499	F
SC	Mary	83268	F
SD	Mary	14211	F
TN	Mary	105392	F
TX	Mary	208924	F
UT	Mary	13551	F
VA	Mary	95784	F
VT	Mary	7652	F
WA	Mary	33970	F
WI	Mary	83741	F
WV	Mary	58456	F
WY	Mary	6169	F

Fig 8: the most popular names by state, female

It is very interesting to notice that male names are more diversified than female names.

3. What is the most gender ambiguous name in 2013? 1945?

I calculate the ambiguity as: take the absolute value of the difference between the male frequency and the female frequency, then the ambiguity is the absolute value divided by the total frequency of that name.

One thing to notice is some name may have a big ambiguity number but its frequencies for both male and female are very low. For example, Ryley has equal female and male frequencies but the frequencies are very small, 194 for male and 192 for female. We are not interested in a gender ambiguous but rare name. So the question is asked in a bad way and it should be " What is the most gender ambiguous name in 2013 with at least 100, 200, 1000 counts ?" I experiment on different threshold numbers and they are showed in the following figures:

Year 2013

name	ambiguity	freq_total_m	freq_total_f
Ryley	0.0052	194	192
Arie	0.0064	156	158
Salem	0.0109	186	182
Teegan	0.0114	178	174
Milan	0.0136	968	942
Lennon	0.0285	578	546
Oakley	0.0286	576	544
Daylin	0.0333	124	116
Jael	0.0388	322	348
Jules	0.0448	128	140
Aven	0.0456	298	272
Reilly	0.0512	226	204
Ever	0.0522	242	218
Gracen	0.0556	136	152
Jaidyn	0.0640	278	316
Gentry	0.0667	182	208
Palmer	0.0694	228	262
Azariah	0.0740	508	438
Charlie	0.0823	3102	2630
Kylin	0.0897	158	132

Fig 9: counts >= 100, the most ambiguous name is Ryley

name	ambiguity	freq_total_m	freq_total_f
Charlie	0.0823	3102	2630
Justice	0.0909	1160	1392
Dakota	0.0937	1780	2148
Phoenix	0.1143	1550	1232
Skyler	0.1244	2224	1732
Emerson	0.2160	1946	3018
Amari	0.2456	1902	1152
Rowan	0.2507	2350	1408
Hayden	0.2733	5866	3348
Riley	0.3190	5062	9804
Finley	0.3192	1124	2178
Peyton	0.4247	3666	9078
Quinn	0.5013	1750	5268
Alexis	0.6009	2364	9482
Avery	0.6350	4072	18242
Sawyer	0.6429	6284	1366
Parker	0.6494	11244	2390
Taylor	0.6679	1636	8216
Payton	0.6766	1002	5194
Angel	0.6800	12640	2408

Fig 10: counts >= 1000, the most ambiguous name is Charlie

name	ambiguity	freq_total_m	freq_total_f
Hayden	0.2733	5866	3348
Riley	0.3190	5062	9804
Peyton	0.4247	3666	9078
Avery	0.6350	4072	18242

Fig 11: counts >= 3000, the most ambiguous name is Hayden

Year 1945

name	ambiguity	freq_total_m	freq_total_f
Artie	0.0000	120	120
Leigh	0.0263	156	148
Lavern	0.0345	240	224
Toby	0.0391	246	266
Frankie	0.0643	1158	1018
Leslie	0.0745	3954	3406
Jessie	0.0805	1884	2214
Jackie	0.0850	2988	2520
Gerry	0.1646	552	396
Ivory	0.1915	168	114
Sydney	0.2000	176	264
Gale	0.2163	576	894
Marty	0.2332	238	148
Carrol	0.2340	216	348
Tracy	0.2448	300	182
Guadalupe	0.2471	646	1070
Johnnie	0.2623	3200	1870
Alva	0.2637	230	134
Cleo	0.2639	212	364
Jan	0.2706	992	1728

Fig 12: counts >= 100, the most ambiguous name is Artie

name	ambiguity	freq_total_m	freq_total_f
Frankie	0.0643	1158	1018
Leslie	0.0745	3954	3406
Jessie	0.0805	1884	2214
Jackie	0.0850	2988	2520
Johnnie	0.2623	3200	1870
Marion	0.3126	1678	3204
Lynn	0.3832	2434	5458
Willie	0.5969	14062	3550
Lee	0.6201	5876	1378
Terry	0.7558	13674	1902

Fig 13: counts >= 1000, the most ambiguous name is Frankie

name	ambiguity	freq_total_m	freq_total_f
Leslie	0.0745	3954	3406
Willie	0.5969	14062	3550

Fig 14: counts >= 3000, the most ambiguous name is Lessie

4. Of the names represented in the data, find the name that has had the largest percentage increase in popularity since 1980. Largest decrease?

The same thing should be noticed here: some name may have a big increase/decrease portion but its frequencies for both 1980 and 2013 are very low. We are not interested in an increasing/decreasing but rare name. So the question is asked in a bad way and it should be " What is the most growing/dying name since 1980 with at least 100, 200, 1000 counts ?" I experiment on different threshold numbers and they are showed in the following figures:

name	change_portion	freq_1980	freq_2013
Misty	-0.9978	5541	12
Jill	-0.9971	4553	13

Fig 15: the most dying name is Misty since 1980. (but in 2013 it is almost zero)

name	change_portion	freq_1980	freq_2013
Colton	1286.8000	5	6439
Aria	1018.4000	5	5097

Fig 16: the most growing name is Colton since 1980 but in 1980 it is almost zero.

If we restrict that the frequency of the name ALWAYS has to be more than 100 from 1980 to 2013, the results (more significant) are different:

name	change_portion	freq_1980	freq_2013
Heather	-0.9871	19988	257
Lisa	-0.9807	15681	302

Fig 17: the most dying name is Heather since 1980 and it is always greater than 100 from 1980 to 2013.

name	change_portion	freq_1980	freq_2013
Sebastian	60.9091	121	7491
Avery	59.5924	184	11149

Fig 18: the most growing name is Sebastian since 1980 and it is always greater than 100 from 1980 to 2013.

B) Onward to Insight!

One conjecture is that our names are becoming more and more international and diversified. How to measure this diversification? We can use the entropy as the metrics for the diversification: the high the entropy is, the more diversified the names are. We can even calculate the entropy of names for each year from 1910 to 2013 and the time series of the names' entropy would verify the conjecture.

Another question we would like to ask what factors are driving the choices of the names? Things like fashion, manliness, cultural background may play very important role. Unfortunately we don't have the data for these factors which are unobservable or latent. We can borrow the idea of topic modeling from NLP: from the data set, we can build a matrix whose rows are different states and columns are the names and each entry is the frequency of the name within that state. We can do the singular value decomposition (SVD) of this matrix.

The matrix is a 51 X 1000 matrix M; each of the 51 rows is a state; each of the 1000 columns is a name. SVD gives M = P*V*Q: P is a 51 X t matrix whose t is the number of latent factors like fashion, manliness or cultural background and we need to choose what t is; V is a t X t matrix which presents the strength of the factors; the Q is a t X 1000 matrix.

For each year, we use SVD to decompose the state-name co-occurrence matrix to see what factors are driving the choices of the names. We do this for all the years from 1910 to 2013 then we may know the evolution of the latent factors.

The SVD approach is very classical for discovering the latent factors and we can apply it here.

The code instructions: If you want to replicate my results, you need to do these in order:

0. Download the country level data from http://www.ssa.gov/oact/babynames/names.zip and unzip.
1. Run the java programs MergeFiles.java and MergeStates.java to merge the small txt files into a big txt file which is good for MYSQL.
2. Run the 'sql_test_2.txt' script with your mySQL database. Of course you have to change the paths in the script to your values.

I upload everything, including the codes, reports into a folder under my GitHub account. This is the link:

https://github.com/givenwong/capitalone_2