Python Notebook Viewer

In [1]:

```
import random
import numpy as np
import pandas as pd
import pandas.io.sql as psql
import psycopg2 as pg
import matplotlib.pyplot as plt
```

In [2]:

```
def randZipf(n, alpha, numSamples):
    # Calculate Zeta values from 1 to n:
    tmp = np.power( np.arange(1, n+1), -alpha )
    zeta = np.r_[0.0, np.cumsum(tmp)]
    # Store the translation map:
    distMap = [x / zeta[-1] for x in zeta]
    # Generate an array of uniform 0-1 pseudo-rando
    u = np.random.random(numSamples)
    # bisect them with distMap
    v = np.searchsorted(distMap, u)
    samples = [t-1 for t in v]
    return samples
```

To generate Zipf distribution, we need at least three parameters:

1. The number of k (rank)
2. Alpha which is must > 1
3. The sample size

In our case, we have a diabetes dataset with the 8 number of attributes (categorical attributes) and 8 measures (numerical attributes). The number of rows = 98052

We have two experiment settings:

1. Missing based on zipf distribution on Attributes

2. Missing based on zipf distribution on Measures

Before add missing values to the dataset, we generate the ideal-topk views, sorted the Attributes and Measures based on the highest utility score which is the most important one.

The best attribute/measure will have more missing rather than attributes/measures which have low ranking

In [3]:

```
column_rank = list(range(0,8))
rows = 98052 # 98052 rows
cols = 8 # 8 attributes # 8 measures
```

We used 11 alpha settings = [1.01, 1.03, 1.06, 1.07, 1.1, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0]

In [4]:

```
# The number of sample is 98052*8 for the attribute
# The number of sample is 98052*8 for the measures
N_sample = rows*cols
a_list =[1.01, 1.03, 1.06, 1.07, 1.1, 1.5, 2.0, 2.5
```

In [5]:

```
print("a, column rank, number of missing, percentag
for a in a_list:
    s = randZipf(cols, a, N_sample)
    unique, counts = np.unique(s, return_counts=Tru
    d = dict(zip(unique, counts))
    for i in column_rank:
        num_missing, percentage = d[i], d[i]/N_samp
        print(a, i+1, num_missing, percentage)
    print("\n")
```

Out [5]:

```
a, column rank, number of missing, percentage
1.01 1 291491 0.37160256802512953
1.01 2 143985 0.18355693917513155
1.01 3 95968 0.12234324644066413
1.01 4 72114 0.0919333618896096
1.01 5 57140 0.07284400114225105
1.01 6 47489 0.06054058050830172
1.01 7 40771 0.051976247297352424
1.01 8 35458 0.04520305552155999
```

1.03 1 295380 0.37656039652429324
1.03 2 144736 0.18451433933015135
1.03 3 95095 0.1212303165667197
1.03 4 71049 0.09057566393342308
1.03 5 56601 0.0721568657447069
1.03 6 46604 0.05941235262921715
1.03 7 39852 0.05080467507037082
1.03 8 35099 0.04474539020111777

1.06 1 303042 0.3863281728062661
1.06 2 145428 0.18539652429323217
1.06 3 94561 0.12054955533798392
1.06 4 69970 0.08920011830457308
1.06 5 54981 0.07009163505078937
1.06 6 44847 0.057172469709949825
1.06 7 38240 0.04874964304654673
1.06 8 33347 0.04251188145065883

1.07 1 304922 0.38872486027821973
1.07 2 146106 0.1862608615836495
1.07 3 94092 0.11995165830375719
1.07 4 69295 0.08833960551544079
1.07 5 54350 0.06928721494717088
1.07 6 44768 0.0570717578427773
1.07 7 38016 0.048464080283930976
1.07 8 32867 0.041899961245053643

1.1 1 312148 0.39793680904010115
1.1 2 146041 0.18617799738914045
1.1 3 92837 0.11835174193285196
1.1 4 68106 0.08682382817280626
1.1 5 53435 0.068120742055236
1.1 6 43551 0.05552028515481581
1.1 7 36945 0.0470987333251744
1.1 8 31353 0.039969862929873944

1.5 1 407353 0.5193073573206054
1.5 2 143843 0.18337591278097337
1.5 3 78107 0.09957344062334271
1.5 4 51280 0.06537347529882104
1.5 5 36527 0.04656585281279321
1.5 6 27364 0.03488455105454249
1.5 7 22078 0.028145779790315344

```
1.5 8 17864 0.022773630318606453


2.0 1 513727 0.6549165238852853
2.0 2 128413 0.1637052278382899
2.0 3 56802 0.07241310733080407
2.0 4 32258 0.041123587484192065
2.0 5 20539 0.026183810631093707
2.0 6 14199 0.0181013645820585
2.0 7 10414 0.013276118794109249
2.0 8 8064 0.010280259454167176


2.5 1 596903 0.7609520968465712
2.5 2 105394 0.13435982947823605
2.5 3 38187 0.048682076857177826
2.5 4 18446 0.023515583567902745
2.5 5 10785 0.013749082119691593
2.5 6 6765 0.008624250397748133
2.5 7 4579 0.005837463794721168
2.5 8 3357 0.004279616937951291


3.0 1 656029 0.8363279178395138
3.0 2 82415 0.10506542446864929
3.0 3 24249 0.030913443886917146
3.0 4 10300 0.013130787745278016
3.0 5 5269 0.006717099090278627
3.0 6 2921 0.003723789417859911
3.0 7 1921 0.002448955656182434
3.0 8 1312 0.0016725818953208503


3.5 1 697147 0.8887465324521683
3.5 2 61831 0.0788242463182801
3.5 3 14821 0.01889431118182189
3.5 4 5568 0.007098274385020194
3.5 5 2491 0.003175610900338596
3.5 6 1268 0.0016164892098070411
3.5 7 810 0.0010326153469587566
3.5 8 480 0.0006119202056051891


4.0 1 725230 0.9245476889813569
4.0 2 45244 0.05767857871333578
4.0 3 9016 0.011493901195284135
4.0 4 2718 0.0034649981642393833
4.0 5 1149 0.0014647839921674215
4.0 6 562 0.0007164565740627422
```

```
4.0 7 322 0.00041049647126014765
4.0 8 175 0.0002230959082935585
```

If we see the result above, using alpha = 1.01, the number of missing in each columns will be:

`a, column rank, number of missing, percentage 1.01 1 291734 0.3719123526292171 1.01 2 144161 0.1837813099171868 1.01 3 95464 0.12170073022477869 1.01 4 71719 0.091429802553747 1.01 5 57363 0.07312828907110513 1.01 6 47763 0.06088988495900135 1.01 7 40522 0.051658813690694735 1.01 8 35690 0.04549881695426916`

And using alpha = 4.0, the number of missing in each columns will be:

`a, column rank, number of missing, percentage 4.0 1 725361 0.9247146922041366 4.0 2 45188 0.05760718802268184 4.0 3 8886 0.011328172806266064 4.0 4 2795 0.0035631603638885487 4.0 5 1150 0.0014660588259290989 4.0 6 570 0.0007266552441561621 4.0 7 294 0.0003748011259331783 4.0 8 172 0.00021927140700852608`

The sum number of missing = N_sample and the sum of percentage missing = 100%

In this experiment we used different missing percentage: 10%, 20%, 30%,..... 90% missing based on Zipf distribution

Let check the real dataset, for instance, dataset with 10% missing on measure based on zipf distribution with alpha = 1.01

In [6]:

```
# Example count zipf missing from DB
# Example missing zipf on db_10zipf101_missing_meas

conn = pg.connect("dbname=same_len_col_large_experi
db_10zipf101_missing_measure1 = psql.read_sql("SELE
db_10zipf101_missing_measure1.drop(db_10zipf101_mis
db_10zipf101_missing_measure1.isnull().sum()
```

Out [6]:

```
race                    0
gender                  0
age                     0
```

```
admission_type_id       0
diag_1                  0
insulin                 0
change                  0
readmitted              0
number_emergency      493
number_outpatient     243
number_inpatient      162
number_diagnoses      121
num_procedures         97
num_medications        77
time_in_hospital       70
num_lab_procedures     61
dtype: int64
```

In [7]:

```
# Sum all missing values
db_10zipf101_missing_measure1.isnull().sum().sum()
```

Out [7]:

```
1324
```

As shown in the result above the number of missing is very small. Then let see if we use the extream setting:

1. 100% of missing on attributes
2. Zipf distribution with alpha = 4

In [8]:

```
# Example count zipf missing from DB
# Example missing zipf on db_100zipf400_missing_att

conn = pg.connect("dbname=same_len_col_large_experi
db_100zipf400_missing_attr1 = psql.read_sql("SELECT
db_100zipf400_missing_attr1.drop(db_100zipf400_miss
db_100zipf400_missing_attr1.isnull().sum()
```

Out [8]:

```
gender                  90634
admission_type_id        5657
age                      1122
insulin                   357
race                      145
diag_1                     71
change                     40
readmitted                 22
```

```
time_in_hospital          0
num_lab_procedures        0
num_procedures            0
num_medications           0
number_outpatient         0
number_emergency          0
number_inpatient          0
number_diagnoses          0
dtype: int64
```

The first column has the highest number of missing which is 92% of cells will be replaced with NaN. However, with this condition we cannot compare between a certain percent missing on random setting and on zipf distribution setting.

Let's check the sum of missing values from 100% missing based on zipf distribution with alpha = 4

In [9]:

```
db_100zipf400_missing_attr1.isnull().sum().sum()
```

Out [9]:

```
98048
```