Assignment 3 - Data Cleaning

Due: Wednesday, March 25 at 11:55PM

NO LATE SUBMISSIONS WILL BE ACCEPTED

Goal

In this homework, you will practice cleaning and transforming data in the Spark environment with PySpark. We will use the same Parking Violation dataset from the Data Profiling lab.

Data

The Parking Violation dataset is in the HDFS directory /user/hc2660/hw3data/data_cleaning.csv
As before, here's how to load the dataset into a dataframe in Pyspark:

```
df = spark.read.csv(path='/user/hc2660/hw3data/data_cleaning.csv', header=True)
```

To view the first two entries and check that the data is properly loaded:

```
df.view(2)
```

Submission

We have provided you with two items: *template1.py* and *template2.txt*.

1. You are to write your codes in *template1.py*. We will generate your codes' outputs from this template for grading.
2. For questions that require qualitative responses, please provide them in *template2.txt*.

Submit your completed templates to NYU Classes.

Dependencies

You must use the following dependencies for Pyspark:

```
module load python/gnu/3.6.5
module load spark/2.4.0
```

Tasks

Task 1: Assessing Data Quality

As we discussed in class, there are many different types of data quality problems. In this task, you will explore some common issues that are encountered.

0

a. The column **summons_number** is a key for the Parking Violations table. Write a query that checks whether there are any violations to the *key constraint* and output the number of violations.

summons number是PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table

b. To better understand the different types of plates (**plate_type)** and how common they are in the data, write a query that outputs a table of all unique types of car plates and their counts, ordered from the most to the least common plate types.

b.
PAS,94928
OMS,1745
OMT,1739
SRF,613
COM,427
999,274
ORG,59
CMB,59
SPO,43
RGL,27
SRN,21
HIS,14
PHS,11
APP,6
OML,6

c:
PAS,94928
OMS,1745
OMT,1739
SRF,613
COM,427
NULL,274
ORG,59
CMB,59
SPO,43
RGL,27
SRN,21
HIS,14
PHS,11
APP,6

c. In the list of plate types, you will find one with value “999” which seems to indicate NULL values for this column. Transform all entries with value “999” assignment to “NULL”, and output all unique types of car plates and their counts (in descending order).

d. Suppose we are interested in analyzing violations based on what county they have occurred in. To do so, we need to exclude rows that have incomplete information in the **violation_county** column. Output the number of entries with an empty value in the **violation_county** column. Filter out these entries (i.e., remove the rows that contain empty values) and output the number of entries remaining in the dataset.

d1:467 d2:99533

Task 2: Clustering

Often when data is input by humans, there can be errors and inconsistencies in how values are represented. *Clustering* can help detect entries in a column that are similar and potentially represent the same value. One class of clustering strategies that can be used for data cleaning is *key collision*. Key collision methods are based on the idea of creating a key value that contains only the most valuable or meaningful part of a string; then, distinct strings are grouped based on their key (hence the name "key collision").
Some efficient key collision techniques are described in

https://github.com/OpenRefine/OpenRefine/wiki/Clustering-In-Depth

a. Write a function that implements the Fingerprint Method described. Use this template:

```
>>>import string, unicodedata
>>> def fingerprint(value):
        ……
        return (key, value)
```

b. Write a function that implements the N-Gram Fingerprint Method described. Use the template below.

```
>>>import string, unicodedata
>>> def ngram_fingerprint(value, n = ):
        ……
        return (key, value)
```

c. Apply your Fingerprint function to the **plate_id** column. Collect and print out all output. Note that you must first transform this column from DataFrame to RDD. Hint: You can use “map” to apply the function to the values in the column.

85136

d. Apply your N-Gram Fingerprint function to the **plate_id** column. Collect and print out all output.

85118

e. Apply your Fingerprint Method to the **street_name** column and cluster the results based on their keys. How many clusters are there? Print out the first 20 clusters.

8995

f. Apply your N-Gram Fingerprint Method to the **street_name** column and cluster the results based on their keys. How many clusters are there? Print out the first 20 clusters.

f:
n-1 6867
n-2 8888
n-3 8897
n-4 8823

g. Which fingerprint strategy works better for **plate_id**? What about for **street_name?** Explain why.

h. Explore some of these clusters from items 2.e. As discussed in the in-class lab session on data cleaning, when there are too many clusters, by normalizing the data, you can reduce the number of clusters and simplify the repair operation. Design and perform transformations to make values in the **street_name** column more consistent. After applying these transformations, how many clusters do you have? Output all clusters.

i. Examine the clusters in 2.h and propose how to repair the data. Explain your recommendation.

6938

Have a Good Spring Break!

1. In task 1-a, 1-d1 and 1-d2, your result should be an integer, which then will be fit into the output method we give in the template. Please do not generate your result in a dataframe or rdd.

2. Since the output for task1-d1 is an integer, the current output method for that part is not correct. I have made a new template attached to this announcement. Please use the newer version or you can simply modify the task1d output method to "sc.parallelize([str(task1d_result1)]).saveAsTextFile("hw3-task1-d1.out")".