In [4]:
```
from future.builtins import next
import os
import csv
import re
import logging
import optparse

import dedupe
from unidecode import unidecode

import pandas as pd
```

In [5]:
```
pd.options.display.float_format = '{:20,.2f}'.format
pd.set_option('display.max_rows', 5000)
pd.set_option('display.max_columns', 5000)
pd.set_option('display.width', 1000)
pd.set_option('display.max_colwidth', -1)
```

In [11]:
```
ebooks1_all_path = (r'/home/ubuntu/jupyter/ServerX/1_Standard Data Inte
                    r'/Processed Data/product_samples/ebooks1_all.csv')
```

In [12]:
```
input_file = ebooks1_all_path
output_file = 'ebooks1_output2.csv'
settings_file = 'ebooks1_learned_settings2'
training_file = 'ebooks1_training2.json'
```

DF and corpus prep

In [19]:
```
fields_of_interest = [
    'Id',
    'name',
    'description',
    'producer',
    'price',
    'source'
]
```

In [20]:
```
ebooks1_all = pd.read_csv(ebooks1_all_path, sep=',', quotechar='"')[fie
```

In [21]:
```
ebooks1_all.columns
```

Out[21]:
```
Index(['Id', 'name', 'description', 'producer', 'price', 'source'], dt
ype='object')
```

In [48]:
```
nan_float_ids = [4614, 4770, 8449]
```

In [51]:
```
ebooks1_all = ebooks1_all[~(ebooks1_all['name'].isnull())]
```

```
In [57]: ebooks1_all[(ebooks1_all['description'].isnull())].head()
```

Out[57]:

	Id	name	description	producer	price	source
943	944	Running Technique	NaN	Brian Martin	8.99	itunes
7694	1195	Canoe Country	NaN	University of Minnesota Press	60.00	ebooks
8165	1666	Best Bike Rides Philadelphia	NaN	Falcon Guides	17.99	ebooks
10047	3548	A Guide to Improvised Weaponry	NaN	F+W Media	15.99	ebooks
10114	3615	The Official Gun Digest Book of Guns & Prices 2015	NaN	F+W Media	26.99	ebooks

```
In [41]: x = ebooks1_all[(ebooks1_all['name'].isnull()]
         x.columns

Out[41]: Index(['Id', 'name', 'description', 'producer', 'price', 'source'], dt
         ype='object')
```

```
In [37]: type(x['name'][11113])

Out[37]: float
```

```
In [12]: ebooks1_all[ebooks1_all['name'] == None]
```

Out[12]:

Unnamed: 0	Id	name	description	producer	price	source

```
In [14]: description_corpus = ebooks1_all['description'].to_list()
         description_corpus = [x for x in description_corpus if str(x) != 'nan']
```

```
In [58]: producer_corpus = ebooks1_all.drop_duplicates().to_dict('records')
```

```
In [59]: producers = list(ebooks1_all['producer'].unique())
         producers = [x for x in producers if str(x) != 'nan']
```

--

In [65]:
```
def preProcess(key, column):

    try : # python 2/3 string differences
        column = column.decode('utf8')
    except AttributeError:
        pass
    column = unidecode(column)
    column = re.sub('  +', ' ', column)
    column = re.sub('\n', ' ', column)
    column = column.strip().strip('"').strip("'").lower().strip()
    column = column.lower()
    if not column:
        return None

    if key == 'price':
        column = float(column)
    return column
```

In [66]:
```
def readData(filename):

    data_d = {}
    with open(filename) as f:
        reader = csv.DictReader(f)
        for row in reader:
            clean_row = [(k, preProcess(k, v)) for (k, v) in row.items(
            row_id = int(row['Id'])
            data_d[row_id] = dict(clean_row)

    return data_d
```

In [67]:
```
print('importing data ...')
data_d = readData(input_file)
```

```
importing data ...
```

Dedupe Processing

In [68]:
```
fields = [
    {'field' : 'name', 'type': 'Name'},
#    {'field' : 'name', 'type': 'String'},
 #   {'field' : 'description',
 #    'type': 'Text',
 #    'corpus': description_corpus,
 #    'has_missing': True
 #   },
#    {'field' : 'category',
#     'type': 'FuzzyCategorical',
#     'categories': categories,
#     'corpus': category_corpus,
#     'has missing' : True
#    },
    {'field' : 'producer',
     'type': 'FuzzyCategorical',
     'categories': producers,
     'corpus': producer_corpus,
     'has_missing': True
    },
    {'field' : 'price',
     'type': 'Price',
     'has_missing': True
    },
]
```

In [69]:
```
deduper = dedupe.Dedupe(fields)
```

In [70]:
```
deduper.prepare_training(data_d)
```

```
INFO:dedupe.canopy_index:Removing stop word  d
INFO:dedupe.canopy_index:Removing stop word  h
INFO:dedupe.canopy_index:Removing stop word de
INFO:dedupe.canopy_index:Removing stop word en
INFO:dedupe.canopy_index:Removing stop word es
INFO:dedupe.canopy_index:Removing stop word s
INFO:dedupe.canopy_index:Removing stop word t
INFO:dedupe.canopy_index:Removing stop word y
INFO:dedupe.canopy_index:Removing stop word  b
INFO:dedupe.canopy_index:Removing stop word  r
INFO:dedupe.canopy_index:Removing stop word an
INFO:dedupe.canopy_index:Removing stop word fo
INFO:dedupe.canopy_index:Removing stop word in
INFO:dedupe.canopy_index:Removing stop word nd
INFO:dedupe.canopy_index:Removing stop word or
INFO:dedupe.canopy_index:Removing stop word r
INFO:dedupe.canopy_index:Removing stop word to
INFO:dedupe.canopy_index:Removing stop word un
INFO:dedupe.canopy_index:Removing stop word  l
```

In [71]:
```
dedupe.consoleLabel(deduper)
```

```
Do these records refer to the same thing?
(y)es / (n)o / (u)nsure / (f)inished / (p)revious

n

name : 100 things bulls fans should know & do before they die
producer : triumph books
price : 11.99

name : the psycho 100
producer : triumph books
price : 11.99

0/10 positive, 2/10 negative
Do these records refer to the same thing?
(y)es / (n)o / (u)nsure / (f)inished / (p)revious

n

name : 100 things syracuse fans should know & do before they die
producer : triumph books
```

In [72]:
```
deduper.train()
```

```
INFO:rlr.crossvalidation:using cross validation to find optimum alph
a...
INFO:rlr.crossvalidation:optimum alpha: 0.100000, score 0.634112458416
3941
INFO:dedupe.training:Final predicate set:
INFO:dedupe.training:(SimplePredicate: (wholeFieldPredicate, produce
r), TfidfTextCanopyPredicate: (0.6, name))
INFO:dedupe.training:(PartialIndexLevenshteinCanopyPredicate: (4, nam
e, CorporationName), SimplePredicate: (oneGramFingerprint, name))
INFO:dedupe.training:(SimplePredicate: (oneGramFingerprint, name), Sim
plePredicate: (wholeFieldPredicate, price))
```

In [73]:
```
with open(training_file, 'w') as tf:
    deduper.writeTraining(tf)
```

In [74]:
```
with open(settings_file, 'wb') as sf:
    deduper.writeSettings(sf)
```

In [75]:
```
threshold = deduper.threshold(data_d, recall_weight=1)
threshold
```

```
INFO:dedupe.canopy_index:Removing stop word and
INFO:dedupe.canopy_index:Removing stop word the
INFO:dedupe.canopy_index:Removing stop word of
INFO:dedupe.blocking:10000, 45.9522732 seconds
INFO:dedupe.api:Maximum expected recall and precision
INFO:dedupe.api:recall: 0.676
INFO:dedupe.api:precision: 0.548
INFO:dedupe.api:With threshold: 0.303
```

Out[75]:
```
0.3033746
```

In [76]:
```
clustered_dupes = deduper.match(data_d, threshold)
print('# duplicate sets', len(clustered_dupes))
```

```
INFO:dedupe.canopy_index:Removing stop word and
INFO:dedupe.canopy_index:Removing stop word the
INFO:dedupe.canopy_index:Removing stop word of
INFO:dedupe.blocking:10000, 47.4198362 seconds

# duplicate sets 327
```

In [80]:
```
for key, values in data_d.items():
    values['price'] = str(values['price'])
```

In [81]:
```
cluster_membership = {}
cluster_id = 0
for (cluster_id, cluster) in enumerate(clustered_dupes):
    id_set, scores = cluster
    cluster_d = [data_d[c] for c in id_set]
    canonical_rep = dedupe.canonicalize(cluster_d)
    for record_id, score in zip(id_set, scores):
        cluster_membership[record_id] = {
            "cluster id" : cluster_id,
            "canonical representation" : canonical_rep,
            "confidence": score
        }
```

In [84]:
```
singleton_id = cluster_id + 1

with open(output_file, 'w') as f_output, open(input_file) as f_input:
    writer = csv.writer(f_output)
    reader = csv.reader(f_input)

    heading_row = next(reader)
    heading_row.insert(0, 'confidence_score')
    heading_row.insert(0, 'Cluster ID')
    canonical_keys = canonical_rep.keys()
    for key in canonical_keys:
        heading_row.append('canonical_' + key)

    writer.writerow(heading_row)

    for row in reader:
        row_id = int(row[0])
        if row_id in cluster_membership:
            cluster_id = cluster_membership[row_id]["cluster id"]
            canonical_rep = cluster_membership[row_id]["canonical repre
            row.insert(0, cluster_membership[row_id]['confidence'])
            row.insert(0, cluster_id)
            for key in canonical_keys:
                row.append(canonical_rep[key].encode('utf8'))
        else:
            row.insert(0, None)
            row.insert(0, singleton_id)
            singleton_id += 1
            for key in canonical_keys:
                row.append(None)
        writer.writerow(row)
```

In [91]:
```
ebooks1_output = pd.read_csv('ebooks1_output2.csv', sep=',', quotechar=
```

In [93]:
```
ebooks1_output.columns
```

Out[93]:
```
Index(['Cluster ID', 'confidence_score', 'Unnamed: 2', 'Id', 'name',
'description', 'producer', 'price', 'source', 'canonical_', 'canonical
_Id', 'canonical_name', 'canonical_description', 'canonical_producer',
'canonical_price', 'canonical_source'], dtype='object')
```

In [94]:
```
fields_to_compare = [
    'Cluster ID',
    'name',
    'price',
    'producer',
    'confidence_score'
]
ebooks1_output = ebooks1_output[fields_to_compare]
```

```
In [96]: ebooks1_output[ebooks1_output['confidence_score'] > 0.6].sort_values('C
```

Out[96]:

	Cluster ID	name	price	producer	confidence_score
16014	0	Steve Cooper's Australian Fishing Guide	17.50	Hardie Grant Books	0.64
19562	0	College Football's Most Memorable Games	29.95	McFarland & Company, Inc., Publishers	0.64
19341	0	The Gun Digest Book of Firearms Assembly/Disassembly Part IV - Centerfire Rifles	24.99	F+W Media	0.64
19502	1	The Complete Sailor, Second Edition	18.00	McGraw-Hill Education	0.64
19474	1	Aussie Rules For Dummies	0.00	Wiley	0.64
16606	1	Heroes are Forever	12.78	Mainstream Publishing	0.64
19607	1	Death to the BCS: Totally Revised and Updated	17.99	Penguin Publishing Group	0.64
17881	2	The Last Days of Shea	9.99	Taylor Trade Publishing	0.64
19654	2	Ice Time	11.99	Crown/Archetype	0.64
19810	3	Have Glove, Will Travel	9.99	Crown/Archetype	0.64

```
In [ ]:
```