Dq3 cleaning notes:

1/ Using regex:

test2 = '231000\*[13] 283000\*[14] 227898\*[15]'

print(re.split(r"[+\*]", test2)[0])

test4 = '8000+, 231000\*[13]'

print(re.split(r"[+\*]", test4))

def split\_formatting(sstr):

# Apply the split on the string. The result will be a string

slist = re.split(r"[\*+([]", str(sstr)) # [10,]

# The element we want to grab is the first element of that list

return slist[0]

# Now, we can simply replace the lambda function in apply() with this function

world\_earthquakes['test\_col'] = world\_earthquakes['others\_source\_deaths'].apply(split\_formatting)

1/ Using split:

def a\_different\_split(sstr):

slist = sstr.split("[")

# for 10[7]|, this would result in ["10", "7]|"]

# Again, we only want to return the first element

return slist[0]

However here, there is a couple of things to clean so regex is more efficient.

3/ Using replace(), it works but not applicable for huge datasets:

world\_earthquakes["others\_source\_deaths"] = world\_earthquakes["others\_source\_deaths"].replace("8000+", 8000).replace("164[6]", 164).replace("1500[6]", 1500).replace("33[8]", 33).replace("45000[9]", 45000).replace("2489[10]", 2489).replace("26271[11] 26000[12]", 26271).replace("231000\*[13] 283000\*[14] 227898\*[15]", 283000).replace("60[16]", 60).replace("222,517[17]", 517).replace("521[19]", 521).replace("15894[20]", 15894).replace("150+", 150).replace("111+", 111).replace("601 (as of October 30, 2011)", 601)