Bu not defterinde, tutarsız metin girişlerini nasıl temizleyeceğimizi öğreneceğiz.

Hadi başlayalım!

Get our environment set up[¶](https://www.kaggle.com/code/alexisbcook/inconsistent-data-entry#Get-our-environment-set-up)

Yapmamız gereken ilk şey kullanacağımız kütüphaneleri ve veri setini yüklemek olacak.

*# modules we'll use*

import pandas as pd

import numpy as np

*# helpful modules*

import fuzzywuzzy

from fuzzywuzzy import process

import charset\_normalizer

*# read in all our data*

professors = pd.read\_csv("../input/pakistan-intellectual-capital/pakistan\_intellectual\_capital.csv")

*# set seed for reproducibility*

np.random.seed(0)

# Do some preliminary text pre-processing[¶](https://www.kaggle.com/code/alexisbcook/inconsistent-data-entry#Do-some-preliminary-text-pre-processing)

Öncelikle verilerin ilk birkaç satırına hızlıca bir göz atalım.

professors.head()

Diyelim ki "Ülke" sütununu temizleyip veri girişi tutarsızlıkları olmadığından emin olmak istiyoruz. Elbette her satırı elle kontrol edip tutarsızlık bulduğumuzda elle düzeltebiliriz. Ancak bunu yapmanın daha verimli bir yolu var!

*# get all the unique values in the 'Country' column*

countries = professors['Country'].unique()

*# sort them alphabetically and then take a closer look*

countries.sort()

countries

array([' Germany', ' New Zealand', ' Sweden', ' USA', 'Australia',

'Austria', 'Canada', 'China', 'Finland', 'France', 'Greece',

'HongKong', 'Ireland', 'Italy', 'Japan', 'Macau', 'Malaysia',

'Mauritius', 'Netherland', 'New Zealand', 'Norway', 'Pakistan',

'Portugal', 'Russian Federation', 'Saudi Arabia', 'Scotland',

'Singapore', 'South Korea', 'SouthKorea', 'Spain', 'Sweden',

'Thailand', 'Turkey', 'UK', 'USA', 'USofA', 'Urbana', 'germany'],

dtype=object)

Sadece şuna bakarak, tutarsız veri girişinden kaynaklanan bazı sorunlar görebiliyorum: örneğin, ' Germany' ve 'germany' veya ' New Zealand' ve 'New Zealand'.

Yapacağım ilk şey, her şeyi küçük harfe dönüştürmek (istersem sonunda geri çevirebilirim) ve hücrelerin başındaki ve sonundaki tüm boşlukları kaldırmak. Büyük harf kullanımı ve sondaki boşluklardaki tutarsızlıklar metin verilerinde çok yaygındır ve bunu yaparak metin veri girişi tutarsızlıklarınızın yaklaşık %80'ini düzeltebilirsiniz.

*# convert to lower case*

professors['Country'] = professors['Country'].str.lower()

*# remove trailing white spaces*

professors['Country'] = professors['Country'].str.strip()

Şimdi daha zor tutarsızlıkları ele alacağız.

# Use fuzzy matching to correct inconsistent data entry[¶](https://www.kaggle.com/code/alexisbcook/inconsistent-data-entry#Use-fuzzy-matching-to-correct-inconsistent-data-entry)

Tamam, 'Ülke' sütununa tekrar bakalım ve yapmamız gereken başka veri temizliği olup olmadığına bakalım.

*# get all the unique values in the 'Country' column*

countries = professors['Country'].unique()

*# sort them alphabetically and then take a closer look*

countries.sort()

countries

array(['australia', 'austria', 'canada', 'china', 'finland', 'france',

'germany', 'greece', 'hongkong', 'ireland', 'italy', 'japan',

'macau', 'malaysia', 'mauritius', 'netherland', 'new zealand',

'norway', 'pakistan', 'portugal', 'russian federation',

'saudi arabia', 'scotland', 'singapore', 'south korea',

'southkorea', 'spain', 'sweden', 'thailand', 'turkey', 'uk',

'urbana', 'usa', 'usofa'], dtype=object)

Yine bir tutarsızlık olduğu görülüyor: 'southkorea' ve 'south korea' aynı olmalı.

Birbirine en yakın olan dizeleri (string) belirlemek için **fuzzywuzzy** paketini kullanacağız. Bu veri seti o kadar küçük ki hataları elle düzeltebiliriz, ancak bu yaklaşım büyük ölçekli işler için uygun değildir. (Binlerce hatayı elle düzeltmek ister miydiniz? Ya on binlerce? İşleri olabildiğince erken otomatikleştirmek genellikle iyi bir fikirdir. Ayrıca, eğlenceli!)

**Bulanık eşleştirme (Fuzzy matching):** Hedef dizeye (string) çok benzeyen metin dizelerini otomatik olarak bulma sürecidir. Genel olarak, bir dizeyi diğerine dönüştürmek için değiştirmeniz gereken karakter sayısı ne kadar azsa, o dize diğerine o kadar "yakın" kabul edilir. Örneğin, "apple" ve "snapple" birbirinden iki değişiklik uzaktayken ("s" ve "n" ekle), "in" ve "on" birbirinden bir değişiklik uzaklıktadır ("i" yerine "o" koy). Bulanık eşleştirmeye her zaman %100 güvenemezsiniz, ancak genellikle size en azından biraz zaman kazandırır.

Fuzzywuzzy, iki dize verildiğinde bir oran döndürür. Oran 100'e ne kadar yakınsa, iki dize arasındaki düzenleme mesafesi o kadar küçüktür. Burada, şehirler listemizden "south korea" kelimesine en yakın mesafeye sahip on dizeyi alacağız.

*# get the top 10 closest matches to "south korea"*

matches = fuzzywuzzy.process.extract("south korea", countries, limit=10, scorer=fuzzywuzzy.fuzz.token\_sort\_ratio)

*# take a look at them*

matches

[('south korea', 100),

('southkorea', 48),

('saudi arabia', 43),

('norway', 35),

('austria', 33),

('ireland', 33),

('pakistan', 32),

('portugal', 32),

('scotland', 32),

('australia', 30)]

Şehirler listemizdeki iki öğenin "south korea" kelimesine çok yakın olduğunu görebiliyoruz: "south korea" ve "southkorea". Şimdi "Country" sütunumuzda 47'den yüksek bir orana sahip tüm satırları "south korea" ile değiştirelim.

Bunu yapmak için bir fonksiyon yazacağım. (Belirli bir görevi birden fazla kez yapmanız gerekeceğini düşünüyorsanız, yeniden kullanabileceğiniz genel amaçlı bir fonksiyon yazmak iyi bir fikirdir. Bu, kodu çok sık kopyalayıp yapıştırmaktan sizi kurtarır, zaman kazandırır ve hataları önlemeye yardımcı olur.)

*# function to replace rows in the provided column of the provided dataframe*

*# that match the provided string above the provided ratio with the provided string*

def replace\_matches\_in\_column(df, column, string\_to\_match, min\_ratio = 47):

*# get a list of unique strings*

strings = df[column].unique()

*# get the top 10 closest matches to our input string*

matches = fuzzywuzzy.process.extract(string\_to\_match, strings,

limit=10, scorer=fuzzywuzzy.fuzz.token\_sort\_ratio)

*# only get matches with a ratio > 90*

close\_matches = [matches[0] for matches **in** matches if matches[1] >= min\_ratio]

*# get the rows of all the close matches in our dataframe*

rows\_with\_matches = df[column].isin(close\_matches)

*# replace all rows with close matches with the input matches*

df.loc[rows\_with\_matches, column] = string\_to\_match

*# let us know the function's done*

print("All done!")

Artık bir fonksiyonumuz olduğuna göre onu teste tabi tutabiliriz!

*# use the function we just wrote to replace close matches to "south korea" with "south korea"*

replace\_matches\_in\_column(df=professors, column='Country', string\_to\_match="south korea")

All done!

Şimdi "Ülke" sütunundaki benzersiz değerleri tekrar kontrol edelim ve "Güney Kore"yi doğru bir şekilde düzenlediğimizden emin olalım.

*# get all the unique values in the 'Country' column*

countries = professors['Country'].unique()

*# sort them alphabetically and then take a closer look*

countries.sort()

countries

array(['australia', 'austria', 'canada', 'china', 'finland', 'france',

'germany', 'greece', 'hongkong', 'ireland', 'italy', 'japan',

'macau', 'malaysia', 'mauritius', 'netherland', 'new zealand',

'norway', 'pakistan', 'portugal', 'russian federation',

'saudi arabia', 'scotland', 'singapore', 'south korea', 'spain',

'sweden', 'thailand', 'turkey', 'uk', 'urbana', 'usa', 'usofa'],

dtype=object)

Harika! Artık veri çerçevemizde sadece "Güney Kore" var ve elle hiçbir şey değiştirmemize gerek kalmadı.