Gezinme

Vektör Uzayı Modellerine Nazik Bir Giriş

kaydeden Adrian Tam 23 Ekim 2021'de Lineer Cebir'de 3

Vektör uzayı modelleri, vektörlerle temsil edilen veriler arasındaki ilişkiyi dikkate alacaktır. Bilgi erişim sistemlerinde popülerdir ancak başka amaçlar için de faydalıdır. Genel olarak bu, iki vektörün benzerliğini geometrik açıdan karşılaştırmamıza olanak tanır.

Bu dersimizde vektör uzayı modelinin ne olduğunu ve neler yapabileceğini göreceğiz.

Bu eğitimi tamamladıktan sonra şunları bileceksiniz:

- Vektör uzayı modeli nedir ve kosinüs benzerliğinin özellikleri
- Kosinüs benzerliğini iki vektörü karşılaştırmanıza nasıl yardımcı olabilir?
- Kosinüs benzerliğini ile L2 mesafesi arasındaki fark nedir?

Başlayalım.

Vektör Uzay Modellerine Nazik Bir Giriş
Fotoğraf: liamfletch, bazı hakları saklıdır.

Öğreticiye genel bakış

Bu eğitim 3 bölüme ayrılmıştır; bunlar:

1. Vektör uzayı ve kosinüs formülü
2. Benzerlik için vektör uzayı modelinin kullanılması
3. Vektör uzayı modellerinin ve kosinüs mesafesinin ortak kullanımı

Vektör uzayı ve kosinüs formülü

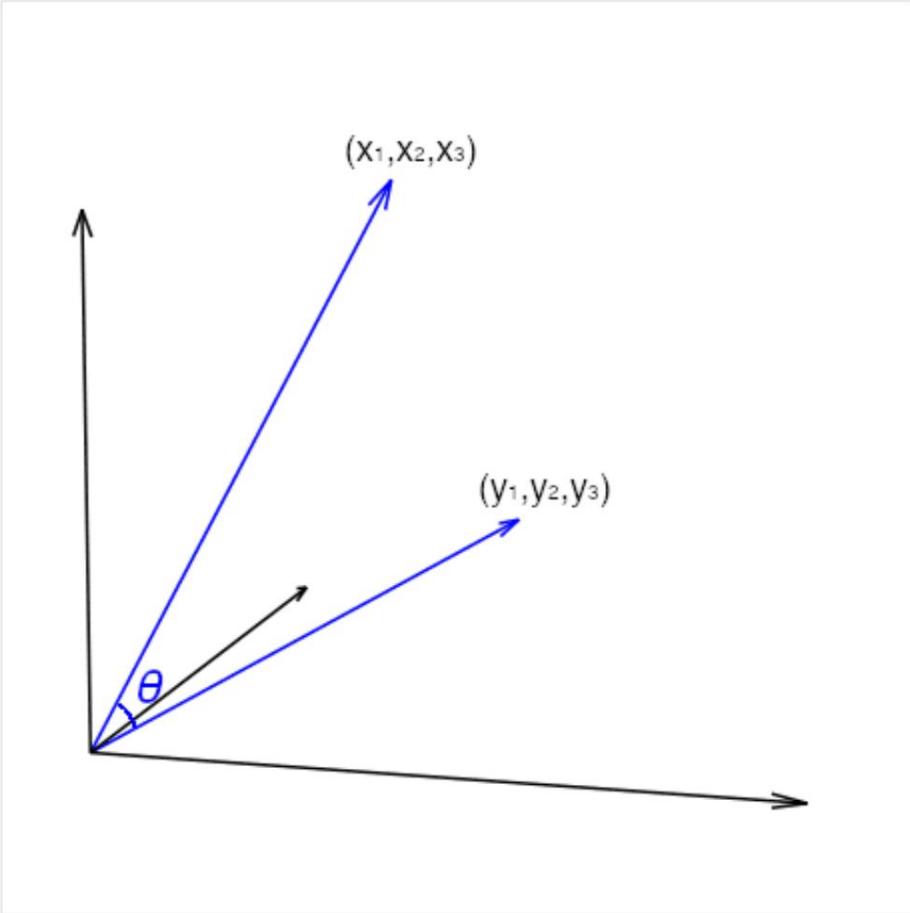
Vektör uzayı, bazı vektör işlemlerini tanımlayan matematiksel bir terimdir. Meslekten olmayanların terimiyle şunları yapabiliriz:
Bunun, her noktanın N boyutlu bir vektörle temsil edildiği N boyutlu bir metrik uzay olduğunu hayal edin.
Bu uzayda herhangi bir vektör toplamasını veya skaler-vektör çarpımını yapabiliriz.

Bir vektör uzayını dikkate almak faydalıdır çünkü şeyleri bir vektör olarak temsil etmek faydalıdır. Örneğin
makine öğreniminde genellikle birden fazla özelliğe sahip bir veri noktamız vardır. Bu nedenle, uygun
bir veri noktasını vektör olarak temsil etmemizi sağlar.

Bir vektör kullanarak normunu hesaplayabiliriz . En yaygın olanı L2 normu veya uzunluğudur.
vektör. Aynı vektör uzayındaki iki vektörle aralarındaki farkı bulabiliriz. 3- olduğunu varsayalım
boyutlu vektör uzayı, iki vektör $(x1, x2, x3)$ Ve $(y1, y2, y3)$. Aralarındaki fark vektördür
$(y1 - x1, y2 - x2, y3 - x3)$, ve farkın L2-normu mesafedir veya daha kesin olarak
Bu iki vektör arasındaki Öklid uzaklığı:

$$\sqrt{(y1 - x1)^2 + (y2 - x2)^2 + (y3 - x3)^2}$$

Uzaklığın yanı sıra iki vektör arasındaki açıyı da dikkate alabiliriz . Vektörü dikkate alırsak
$(x1, x2, x3)$ 3B koordinat sisteminde $(0, 0, 0)$ ile
'den başka bir çizgi bölümü daha var. Kesişme noktaları $(0, 0, 0)$ ve $(y1, y2, y3)$ açı yaparlar:



İki çizgi parçası arasındaki açı kosinüs formülü kullanılarak bulunabilir:

$$\text{çünkü } \theta = \frac{a \quad b}{a \quad 2 \quad b \quad 2}$$

vektör nokta çarpımı nerede ve vektörün L2 normudur. Bu formül şu şekilde ortaya çıkıyor:

nokta çarpımını, vektörün, vektör tarafından işaret edilen yöne izdüşümü olarak düşünürsek. A B

Kosinüsün doğası, açı 0'dan 90 dereceye arttıkça kosinüsün 1'den 0'a azaldığını söyler.

Bazen 1 çünkü θ derdik kosinüs mesafesi çünkü iki vektör olarak 0'dan 1'e kadar uzanır

birbirlerinden daha da uzaklaşıyorlar. Bu, yararlanacağımız önemli bir özelliktir.

vektör uzayı modeli.

Benzerlik için vektör uzayı modelini kullanma

Vektör uzayı modelinin ne kadar kullanışlı olduğuna dair bir örneğe bakalım.

Dünya Bankası dünyadaki ülkeler ve bölgeler hakkında çeşitli veriler toplamaktadır. Her ülke varken

farklıysa, ülkeleri vektör uzay modeli altında karşılaştırmaya çalışabiliriz. Kolaylık sağlamak için şunu kullanacağız:

Dünya Bankası'ndan veri okumak için Python'daki pandas_datareader modülü. Yükleyebilirsiniz

pandas_datareader pip veya conda komutunu kullanarak:

```
pip kurulumu pandas_datareader
```

Dünya Bankası tarafından toplanan veri serileri bir tanımlayıcıyla adlandırılmaktadır. Örneğin, “SP.URB.TOTL”

Bir ülkenin toplam kentsel nüfusu. Dizilerin çoğu yıllık. Bir diziyi indirdiğimizde,

başlangıç ve bitiş yıllarını girmeniz gerekir. Genellikle veriler zamanında güncellenmez. Bu nedenle, eksik verileri önlemek için en son yıl yerine birkaç yıl önceki verilere bakmak en iyisidir.

Aşağıda her ülkenin 2010 yılı bazı ekonomik verilerini toplamaya çalışıyoruz :

```
pandas_datareader'dan wb'yi içe aktar
pandaları pd olarak içe
aktar pd.options.display.width = 0

isimler =
[ "NE.EXP.GNFS.CD", # Mal ve hizmet ihracatı (mevcut ABD Doları)
"NE.IMP.GNFS.CD", # Mal ve hizmet ithalatı (cari ABD Doları)
"NV.AGR.TOTL.CD", # Tarım, ormancılık ve balıkçılık, katma değer (mevcut ABD Doları)
"NY.GDP.MKTP.CD", # GSYİH (mevcut ABD Doları)
"NE.RSB.GNFS.CD", # Mal ve hizmetlere ilişkin dış bakiye (mevcut ABD Doları) ]


df = wb.download(ülke='tümü', gösterge=isimler, başlangıç=2010, bitiş=2010).reset_index() ülkeler =
wb.get_countries() non_aggregates
= ülkeler[ülkeler["bölge"] != "Toplamlar" ]name df_nonagg =
df[df["ülke"].isin(non_aggregates)].dropna() print(df_nonagg)

```

```
ülke yılı NE.EXP.GNFS.CD NE.IMP.GNFS.CD NV.AGR.TOTL.CD NY.GDP.MKTP.
50 Arnavutluk 2010 3.337089e+09 5.792189e+09 2.141580e+09 1.192693e+ 51 Cezayir 2010 6.197541e+10 5.065473e+10 1.364852e+10
1.612073e+ 54 Angola 2010 5.157282e+10 3.568226e+10 5.179055e+09 8.379950 e+ 55 Antigua ve Barbuda 2010 9.142222e+08 8.415185e+08
1.876296e+07 1.148700e+ 56 Arjantin 2010 8.020887e+10 6.793793e+10 3.021382e+10 4.236274e +


.. ... ... ... ... ... .
259 Venezuela, RB 2010 1.121794e+11 6.922736e+10 2.113513e+10 3.931924e+ 260 Vietnam 2010 8.347359e+10 9.299467e+10 2.130649e+10
1.159317e+ 262 Batı Şeria ve Gazze 2010 1.367300e+09 5.264300e+09 8.716000e+08 9.681500e+ 264 Zambiya 2010 7.503513e+09 6.256989e+09
1.909207e+09 2.026556e+ 265 Zimbabve 2010 3.569254e+09 6.440274e+0 9 1,157187e+09 1,204166e+


[174 satır x 7 sütun]
```

Yukarıda her ülkenin 2010 yılındaki bazı ekonomik göçümlerini elde ettik. wb.download() işlevi, Dünya Bankası'ndan verileri indirecek ve bir panda veri çerçevesi döndürecektir. Benzer şekilde wb.get_countries(), Dünya Bankası tarafından tanımlanan ülke ve bölgelerin adını alacaktır; bunu "Doğu Asya" ve "Dünya" gibi ülke dışı toplamları filtrelemek için kullanacağız. Pandalar, satırların boole indeksleme yoluyla filtrelenmesine izin verir; bu, df["country"].isin(non_aggregates), satırın non_aggregates listesinde yer aldığı bir boole vektörü verir ve buna dayanarak, df[df["country"].isin(non_aggregates))] yalnızca bunları seçer. Çeşitli nedenlerden dolayı her ülke tüm verilere sahip olmayabilir. Bu nedenle eksik veriye sahip olanları kaldırmak için dropna() işlevini kullanıyoruz. Uygulamada bazı atama tekniklerini yalnızca kaldırmak yerine uygulamak isteyebiliriz. Ancak örnek olarak kalan 174 veri noktasıyla devam ediyoruz.

Pandalar veya numpy işlevlerdeki gerçek manipülasyonu gizlemek yerine fikri daha iyi açıklamak için, önce her ülkeye ait verileri bir vektör olarak çıkarıyoruz:

```
...
vektörler = satır
kimliği için {} , df_nonagg.iterows() içindeki satır :
vektörler[satır["ülke"]] = satır[isimler].değerler

baskı(vektörler)
```

```
{'Arnavutluk': array([3337088824.25553, 5792188899.58985, 2141580308.0144,
```

```
11926928505.5231, -2455100075.33431], dtype=object), 'Cezayir':
array([61975405318.205, 50654732073.2396, 13648522571.4516, 161207310515.42, 11 320673244.9655],
dtype=object), 'Angola': array([51572818660.8665, 35682259098.1843,
5179054574.41704, 83799496611.2004, 15890559562.6822 ], dtype=nesne),

...
'Batı Şeria ve Gazze': array([1367300000.0, 5264300000.0, 871600000.0, 9681500000.0, -3897000000.0], dtype=object), 'Zambiya':
array([7503512538.82554, 6256988597)
.27752, 1909207437.82702, 20265559483.8548, 1246523941.54802], dtype=object), 'Zimbabve':
array([3569254400.0, 6440274000.0, 1157186600.0, 12041655200.0,
-2871019600.0], dtype=object)}
```

Oluşturduğumuz Python sözlüğü, anahtar olarak her ülkenin adını ve numpy dizisi olarak ekonomik göçümleri içeriyor. 5 metrik vardır, dolayısıyla her biri 5 boyutlu bir vektördür.

Bunun bize yardımcı olduğu şey, her ülkenin diğerine ne kadar benzer olduğunu görmek için vektör temsilini kullanabilmemizdir. Hem farkın L2 normunu (Öklid mesafesi) hem de kosinüs mesafesini deneyelim.
Avustralya gibi bir ülkeyi seçiyoruz ve onu listedeki diğer tüm ülkelerle karşılaştırıyoruz.
seçilmiş ekonomik göçütler.

```
...
numpy'yi np olarak içe aktar

öklid = {} kosinüs
= {} hedef =
"Avustralya"

vektörlerdeki ülke için : vecA =
vektörler[hedef] vecB = vektörler[ülke]
dist = np.linalg.norm(vecA - vecB) cos =
(vecA @ vecB) / (np.linalg.norm(vecA) * np.linalg .norm(vecB))
öklid[ülke] = uzaklık # Öklid mesafesi kosinüs[ülke] = 1-cos # kosinüs mesafesi
```

Yukarıdaki for döngüsünde, vecA'yı hedef ülkenin (yani Avustralya) vektörü olarak ve vecB'yi de diğer ülkenin vektörü olarak ayarladık. Daha sonra farklarının L2-normunu iki vektör arasındaki Öklid mesafesi olarak hesaplıyoruz. Ayrıca formülü kullanarak kosinüs benzerliğini hesaplıyoruz ve bunu 1'den çıkararak kosinüs mesafesini elde ediyoruz. Yüzden fazla ülke arasından hangisinin en kısa Öklidyen değerine sahip olduğunu görebiliriz.
Avustralya'ya uzaklık:

```
...
pandaları pd olarak içe aktar

df_distance = pd.DataFrame({"euclid": euclid, "cos": kosinüs})
print(df_distance.sort_values(by="euclid").head())
```

```
euclid cos Avustralya 0,000000e+00 -2,220446e-16 Meksika
1,533802e+11 7,949549e-03 İspanya 3,411901e+11 3,057903e-03
Türkiye 3,798221e+11 3,502849e-03 Endonezya 4,083531 e+11
7.417614e-03
```

Sonucu sıraladığımızda Öklid uzaklığı altında Avustralya'ya en yakın olanın Meksika olduğunu görebiliriz.
Ancak kosinüs mesafesiyle Avustralya'ya en yakın yer Kolombiya'dır.

```
...
df_distance.sort_values(by = "cos").head()
```

```
öklit çünkü
```

```
0.000000e+00 -2.220446e-16 Kolombiya 8.981118e+11 1.720644e-03
Küba 1.126039e+12 2.483993e-03 İ talya 1.088369e+12 2.677707e
-03 Arjantin 7.572323 e+11 2.930187e-03
```

İ ki mesafenin neden farklı sonuç lar verdiğ ini anlamak iç in üç ülkenin metriğ inin birbirleriyle nasıl karş ılaş tırıldığ ını gözlemleyebiliriz:

```
...
print(df_nonagg[df_nonagg.country.isin(["Meksika", "Kolombiya", "Avustralya"])])
```

```
ülke yılı NE.EXP.GNFS.CD NE.IMP.GNFS.CD NV.AGR.TOTL.CD NY.GDP.MKTP.CD NE.RSB.
59 Avustralya 2010 2.270501e+11 2.388514e+11 2.518718e+10 1.146138e+12 -1.180 91 Kolombiya 2010 4.682683e+10 5.136288e+10 1.812470e+10
2.865631 e+11 -4.536 176 Meksika 2010 3.141423e+11 3.285812 e+11 3.405226e+10 1.057801e+12 -1.443
```

Bu tablodan Avustralya ve Meksika metriklerinin büyüklük olarak birbirine ç ok yakın olduğ unu görüyoruz. Ancak aynı ülke iç indeki her metriğ in oranını karş ılaş tırırsanız, Avustralya'ya en iyi uyanın Kolombiya olduğ unu görürsünüz. Aslında kosinüs formülünden ş unu görebiliriz:

$$\text{çünkü } \theta = \frac{a \quad b}{a \quad 2 \quad b \quad 2} = \frac{A}{a \quad 2} \quad \frac{B}{b \quad 2}$$

bu, iki vektör arasındaki aç ının kosinüsünün, karş ılık gelen vektörlerin 1 uzunluğ a normalize edildikten sonraki nokta ç arpımı olduğ u anlamına gelir. Dolayısıyla kosinüs mesafesi, mesafeyi hesaplamadan önce verilere sanal olarak bir dç ekleyici uygulamaktır.

Bunları bir araya getirirsek kodun tamamı aş ağ ıdadır

```
pandas_datareader'dan wb iç e aktarma
numpy'yi np olarak iç e
aktar pandaları pd olarak
iç e aktar pd.options.display.width = 0

# Dünya Bankası adlarından veri indirme =

[ "NE.EXP.GNFS.CD", # Mal ve hizmet ihracatı (mevcut ABD Doları)
"NE.IMP.GNFS.CD", # Mal ve hizmet ithalatı (cari ABD Doları)
"NV.AGR.TOTL.CD", # Tarım, ormancılık ve balıkç ılık, katma değ er (mevcut ABD Doları)
"NY.GDP.MKTP.CD", # GSYİ H (mevcut ABD Doları)
"NE.RSB.GNFS.CD", # Mal ve hizmetlere iliş kin dış bakiye (mevcut ABD Doları) ] df = wb.download(ülke='tümü',

gösterge=isimler, start=2010, end=2010).reset_index( )

# Toplamaları kaldırıyoruz ve yalnızca eksik verisi olmayan ülkeleri tutuyoruz country = wb.get_countries
() non_aggregates = ülkeler[ülkeler["bölge"] !=
"Toplamlar"].name df_nonagg = df[df["ülke"].isin( toplanmamış )].dropna()


# Her ülke iç in vektör ç ıkar .




# Öklid ve kosinüs mesafelerini hesaplayın Öklid = {} kosinüs = {}



hedef = vektörlerdeki ülke iç in
"Avustralya" : vecA = vektörler[hedef]
```

```
vektörler[ülke] dist = np.linalg.norm(vecA - vecB)
cos = (vecA @ vecB) / (np.linalg.norm(vecA) * np.linalg .norm(vecB))
öklit[ülke] = uzaklık # Öklid mesafesi kosinüs[ülke] = 1-cos # kosinüs mesafesi



# Sonuçları yazdırın df_distance
= pd.DataFrame({"euclid": euclid, "cos": cosine}) print("Öklid mesafesine göre en yakın:")
print(df_distance.sort_values(by="euclid").head() ) print() print("Kosinüs
mesafesine göre en yakın:") print(df_distance.sort_values(by="cos").head())



# Ayrıntı düçümlerini yazdırın print() print("Detay
düçümleri:")
print(df_nonagg[df_nonagg.country.isin(["Meksika",
"Kolombiya", "Avustralya"])])
```

Vektör uzayı modellerinin ve kosinüs mesafesinin ortak kullanımı

Vektör uzayı modelleri bilgi erişim sistemlerinde yaygındır. Belgeleri (örneğin bir paragraf, uzun bir pasaj, bir kitap, hatta bir cümle) vektör olarak sunabiliriz. Bu vektör, belgenin içerdiği sözcüklerin sayılması kadar basit (örneğin, bir kelime çantası modeli) veya karmaşık bir yerleştirme vektörü (örneğin, Doc2Vec) olabilir. Daha sonra en alakalı belgeyi bulma sorgusu, tüm belgeler kosinüs mesafesine göre sıralanarak yanıtlanabilir. Daha uzun veya daha kısa belgeleri tercih etmek istemediğimiz, içeriğine odaklanmak istediğimiz için kosinüs mesafesi kullanılmalıdır. Bu nedenle, sorgudaki kelimelerin bir belgede kaç kez belirtildiğinden ziyade, belgelerin sorguyla ne kadar alakalı olduğunu dikkate almak için normalleştirmeden yararlanırız.

Bir belgedeki her kelimeyi bir özellik olarak ele alırsak ve kosinüs mesafesini hesaplarsak, bu "zor" mesafedir çünkü benzer anlamlara sahip kelimeleri önemsemeyiz (örneğin, "belge" ve "geçit" benzer anlamlara sahiptir ancak "mesafe" değildir) ". Word2vec gibi vektörlerin yerleştirilmesi ontolojiyi dikkate almamıza izin verecektir. Kosinüs mesafesinin, dikkate alınan kelimelerin anlamları ile hesaplanmasına "yumuşak kosinüs mesafesi" denir. Gensim gibi kütüphaneler bunu yapmanın bir yolunu sağlar.

Kosinüs mesafesi ve vektör uzayı modelinin başka bir kullanım durumu bilgisayarlı görmedir. El hareketini tanıma görevini hayal edin, elin belirli kısımlarını (örneğin beş parmak) kilit noktalar haline getirebiliriz.
Daha sonra anahtar noktaların (x,y) koordinatlarını vektör olarak ortaya koyarak mevcut veri tabanımızla karşılaştırarak hangi kosinüs mesafesinin en yakın olduğunu görebilir ve hangi el hareketi olduğunu belirleyebiliriz. Herkesin elinin boyutu farklı olduğundan kosinüs mesafesine ihtiyacımız var. Bunun hangi hareketi gösterdiğine ilişkin kararımızı etkilemesini istemiyoruz.

Tahmin edebileceğiniz gibi bu tekniği kullanabileceğiniz çok daha fazla örnek var.

daha fazla okuma

Daha derine inmek istiyorsanız bu bölüm konuyla ilgili daha fazla kaynak sağlar.

Kitabın

- Lineer Cebire Giriş , Beşinci Baskı, 2016.
- Bilgi Erişimine Giriş , 2008.

Yazılım

- Dünya Bankası açık veri
- pandaları veri okuyucusu
- Gensim

Nesne

- Wikipedia'da vektör uzay modeli

Özet

Bu derste vektörlerin benzerliklerini ölçmek için vektör uzayı modelini keşfettiniz.

Özellikle şunları öğrendiniz:

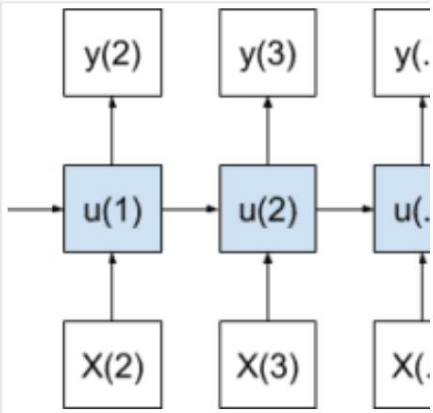
- Bir vektör uzayı modeli nasıl oluşturulur
- Vektör uzayı modelinde kosinüs benzerliği ve dolayısıyla iki vektör arasındaki kosinüs mesafesi nasıl hesaplanır?
- Kosinüs mesafesi ile diğer mesafe ölçümleri arasındaki fark nasıl yorumlanır?
  Öklid mesafesi
- Vektör uzayı modelinin kullanımı nedir?

Bu Konuda Daha Fazla Bilgi

GAN Nasıl Keş fedilir?
Gizli Alan Ne Zaman
Yüz Oluş turma

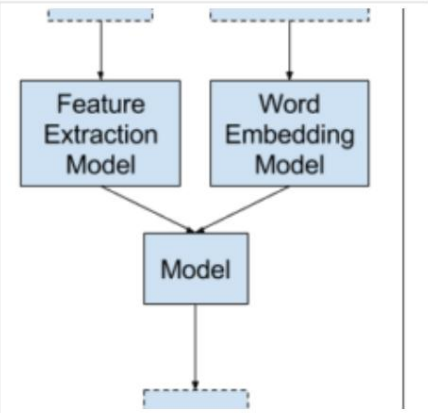
Vektöre Nazik Giriş
Makine Öğ reniminde Normlar

Nazik Bir Giriş
Vektör Değ erli Fonksiyonlar



Modellere Nazik Giriş
Sıra için...

Nazik Giriş
İ statistiksel Dil
Modelleme...



Derin Öğ renmeye Nazik Bir Giriş
Altyazısı...

Adrian Tam hakkında

Adrian Tam, PhD bir veri bilimci ve yazılım mühendisidir.

Adrian Tam'ın tüm gönderilerini görüntüle

Görselleş tirme için Temel Bileş en Analizi

Tavsiye Sistemi Oluş turmak İ çin Tekil Değ er Ayrış tırmanın Kullanılması

Vektör Uzayı Modellerine Nazik Bir Giriş için 3 Yanıt

William Smith 30 Ekim 2021, 04:59 #

CEVAP

Teşekkür ederim, çok net bir giriş ve güzel örnekler.

CEVAP

Imola 26 Haziran 2023, 18:49 #

Merhaba Jason!

İ simlendirmeyle ilgili bazı şüphelerim var.. ML ile vektörleri öğreniyorsak buna Vektör Gömme denir, aksi takdirde (bunu daha geleneksel yaklaşımlarla, daha geniş vektörleştirmeyle yapıyorsak) bunları gömme olarak düşünmüyoruz?

Teşekkür ederim,
İ mola

CEVAP

James Carmichael 27 Haziran 2023, 07:44 #

Merhaba Imola... Terminoloji kafa karıştırıcı olabilir! Aşağıdaki kaynak ilginizi çekebilir
Sen:

https://www.analyticsvidhya.com/blog/2021/06/part-5-step-by-step-guide-to-master-nlp-text-vectorization-approaches/

Cevap bırakın

İ sim (gerekli)

E-posta (yayınlanmayacak) (gerekli)

YORUM GÖNDER

Hoş geldin!
Ben Jason Brownlee PhD
ve geliştiricilerin makine öğrenimi ile sonuç almasına yardımcı oluyorum.
Devamını oku

Hiçbir öğreticiyi kaçırmayın:

Sizin için seçtik:


Makine Öğrenimi için Doğrusal Cebir (7 Günlük Mini Kurs)


Makine Öğrenimi için NumPy Dizilerini İndeksleme, Dilimleme ve Yeniden Şekillendirme


Python'da Temel Bileşen Analizi (PCA) Sıfırdan Nasıl Hesaplanır?


Makine Öğrenimi için Seyrek Matrislere Nazik Bir Giriş


Python ile SVD Sıfırdan Nasıl Hesaplanır?

Öğreticileri seviyor musunuz?

Makine Öğrenimi için Doğrusal Cebir E-Kitap Gerçekten İyi şeyleri bulacağınız yerdir .