



# Doküman Sınıflandırma

## Text Categorization - TC



Prof.Dr.Banu Diri

# Akış

- Görev
- Eğitici Eğitici-siz Öğrenme
- Metin Özellikleri
- Metin Kümeleme
  - Hiyerarşik eklemeli kümeleme
  - Metin kümelerinin birbirine benzerliği
  - Diğer kümeleme yöntemleri
- Özellik Belirleme
  - Çok boyutlu verilerle çalışmak
  - Özellik belirleme yöntemleri
    - Stop - Functionwords
    - Gövdeleme
    - Filtreler (Information Gain, S2N vs.)
    - Kelime grupları
    - Kelime koordinatları
    - Projeksiyonlar (LSI, PCA, LDA)
    - Ağırlıklandırma
    - Metin resimleri
- Metin Sınıflandırmada bir Metot: Naive Bayes



# Görev

- Verilen: bir metin kümesi
- İstenen: metinlerin kategorilere ayrılması
- Örnekler:
  - Haber metinleri: POLİTİK, SPOR, SAĞLIK, MAGAZİN vs. haber başlıklarına ayırmak
  - Web siteleri: EĞİTİM, EĞLENCE, BİLİM vs. türlerine ayırmak, bir sayfaya benzeyen diğer sayfaların bulunması (Arama motorlarındaki gibi)
  - E-mailler: İSTENEN, İSTENMEYEN şeklinde ayırmak
  - Bir metnin yazarını/dilini bulmak



# EĞİTİCİLİ- EĞİTİCİSİZ

Elimizdeki örneklerin etiketleri varsa eğitici, yoksa eğitici-siz yöntemler kullanılır.

- Eğitici → sınıflandırma
- Eğitici-siz → kümeleme



# Metin Özellikleri

- Metinleri ifade etmek için kullanılan özellikler:
  - Kelimeler
  - Kelime türleri
  - N-gramlar
  - Ekler
  - Ek türleri
  - ... ?



# Yazar belirlemede kullanılan özellikler

ID	Style Markers	ID	Style Markers
1	Num. of sentences	12	Avg. Num. of pronoun in a sentence
2	Num. of words	13	Avg. Num. of conjunctions in a sentence
3	Avg. Num. of words in a sentence	14	Avg. Num. of exclamations in a sentence
4	Avg. word length	15	Num. of points
5	Num. of different words	16	Num. of commas
6	Word richness	17	Num. of colons
7	Avg. Num. of nouns in a sentence	18	Num. of semicolons marks
8	Avg. Num. of verbs in a sentence	19	Num. of question marks
9	Avg. Num. of adj. in a sentence	20	Num. of exclamation marks
10	Avg. Num. of adverb in a sentence	21	Num. of inverted / Num. of all sentences
11	Avg. Num. of particle in a sentence	22	Num. of incomplete / Num. of all sentences



# Metinlerin Kelime Frekanslarıyla İfadesi

## Metnin kelime sayılarıyla ifadesi

### Örnek metin

*Manchester United won 2 – 1 against Chelsea , Barcelona tied Madrid 1 – 1 , and Bayern München won 4 – 2 against Nürnberg*

Manchester	(1	(0.04
United	1	0.04
won	2	0.08
2	2	0.08
–	3	0.12
1	3	0.12
against	2	0.08
Chelsea	1	0.04
,	2	0.08
Barcelona	1	0.04
tied	1	0.04
Madrid	1	0.04
and	1	0.04
Bayern	1	0.04
München	1	0.04
4	1	0.04
Nürnberg	1	0.04



# Her metinde aynı kelimeler yer almaz

- dokümanlar \* kelimeler

$$\begin{pmatrix} & T_1 & T_2 & \dots & T_t \\ D_1 & d_{11} & d_{12} & \dots & d_{1t} \\ D_2 & d_{21} & d_{22} & \dots & d_{2t} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ D_n & d_{n1} & d_{n2} & \dots & d_{nt} \end{pmatrix}$$





# Metinlerin N-gram'larla ifadesi

*D1: "Army troops searched for nuclear weapons."*

*D2: "Military personnel investigated reports of dirty bombs"*

- Kelime (word-gram)
- Karakter (Character-gram)

	<b>D1</b>	<b>D2</b>
army troops	1	0
dirty bombs	0	1
for nuclear	1	0
investigated reports	0	1
military personnel	0	1
nuclear weapons	1	0
of dirty	0	1
personnel investigated	0	1
reports of	0	1
searched for	1	0
troops searched	1	0

İki metnin kelime bi-gramları ile ifadesi

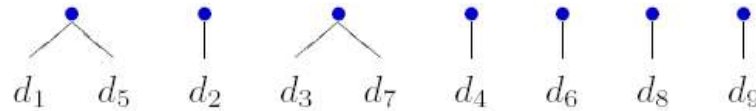
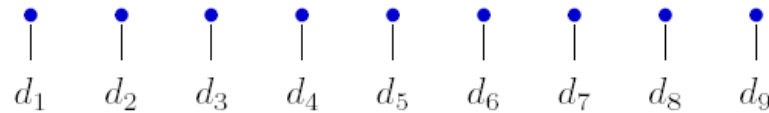
# Metin Kümeleme

- Hiyerarşik eklemeli kümeleme
  - Birbirine en benzeyen iki kümeyi birleştir
  - İşeme devam et

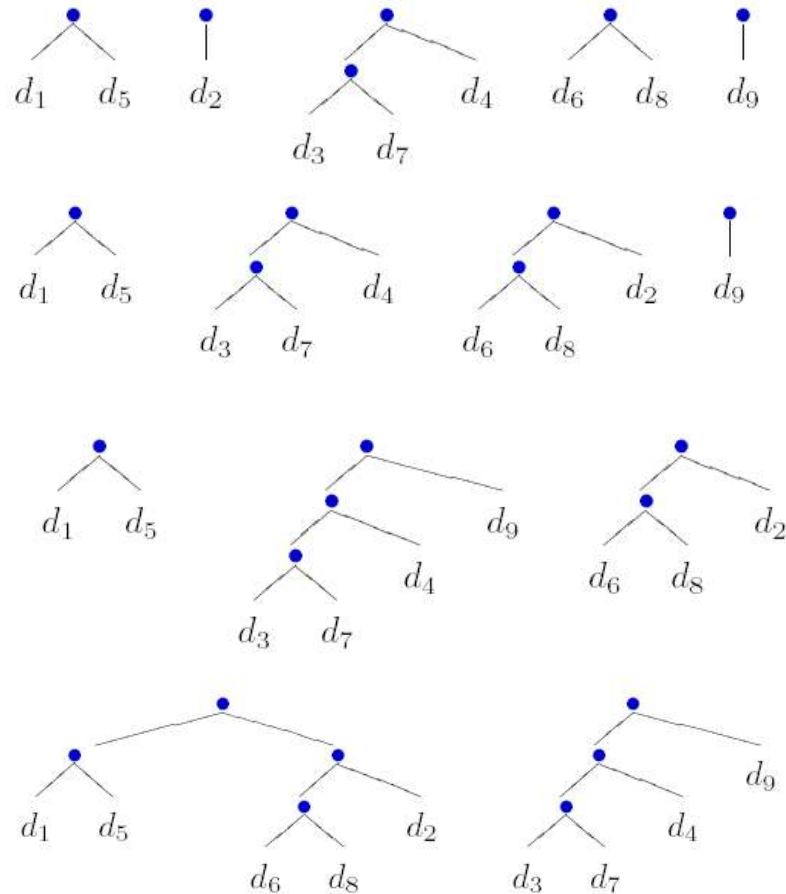


# Hiyerarşik eklemeli kümeleme

Başlangıçta küme sayısı = metin sayısı



# Hiyerarşik eklemeli kümeleme



Sonuçta küme sayısı = 2

# Ne zaman duracağız ?

- İstenen küme sayısına ulaşınca kadar
- Önceden belirlenmiş bir eşik benzerlik değerine ulaşınca kadar



# Metin kümelerinin birbirine benzerliği

- İki kümenin benzerliği
  - En benzer elemanları (Single link)
  - En benzemeyen elemanları (Compete link)
  - Ortalamaları (Group average)

kullanılarak bulunabilir.



# Başka Kümeleme Yöntemleri

- Top-down kümeleme
  - Tek bir kümeyle başlanır
  - Küme içinde birbirine en az benzeyen iki elemanı bulunur
  - Küme bu iki elemanın yakınlığına göre bölünür
  - Oluşan her alt küme için bu işleme tekrar edilir
- K-means
- SOM



# Özellik Belirleme

- Metinler, özellikle kelime frekanslarıyla ifade edildiğinde veri setimizin boyut sayısı çok yüksek olacaktır (binler mertebesinde)
- Çok yüksek boyutta işlem yapmak iyi değildir
- Neden ?
  - işlem hızı
  - başka ?

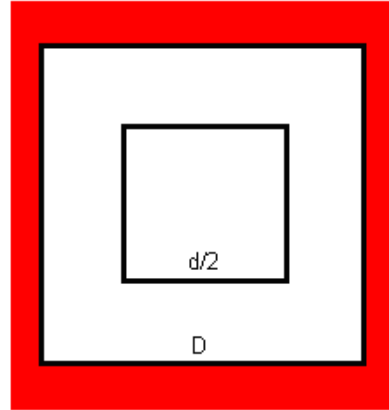




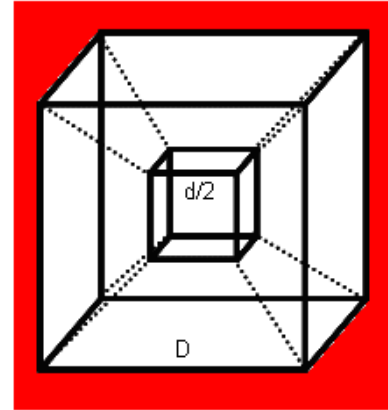
# Çok boyutlu verilerle çalışmak-1



1 boyutlu



2 boyutlu



3 boyutlu

Boyut Sayısı	Merkeze daha yakın noktaların oranı (%)
1	50
2	25
3	12,50
...	...
P	$(\frac{1}{2})^P$

Boyut sayısı arttığında verilerin çok büyük bir kısmı sınıfları ayıran sınırlara çok yakın yerlerde bulunacağından sınıflandırma yapmak zorlaşmaktadır.

# Çok boyutlu verilerle çalışmak-2

- Tek boyutlu uzayda  $[0,1]$  aralığı temsil eden 10 nokta
- Rasgele bir noktanın, uzayı temsil eden noktalardan en yakın olanına ortalama uzaklığı  $= 0.5$
- İki boyutlu uzayda rasgele bir noktanın en yakın noktaya olan ortalama uzaklığının düşey ya da dikey (Manhattan)  $0.5$  olması için gerekli temsilci nokta sayısı  $= 100$

Boyut Sayısı	Gerekli temsil eden nokta sayısı
1	10
2	100
3	1000
...	...
p	$10^p$

**Doğru sınıflandırma yapmak için boyut sayısı artarken gereken örnek sayısı da artmaktadır.**



# Özellik Belirleme Yöntemleri

- Stop - Function words
- Gövdeleme
- Filtreler (Information Gain, S2N vs.)
- Özellik alt küme seçicileri (Wrappers)
- Projeksiyonlar (LSI, PCA, LDA)



## Özellik seçimi neden gereklidir ?

- Sınıflandırıcının doğru karar verme yüzdesini arttırmak
- Daha az eğitim verisi ile çalışıp, aynı başarıyı elde etme şansını arttırmak
- Bellek ve işlem kapasiteleri gereksinimini azaltmak
- Veri üzerinde öngörülemeyen ilişkileri ortaya çıkarmak

## Özellik seçimi, boyut sayısı arttıkça üstel olarak zorlaşır.

- **Tüm alt kümeleri alarak seçim** (wrapper methods):

Tüm olası özellik kümeleri ele alınır ve en etkin alt küme seçilir.

Tam kapsamlı arama üstel hesaplama karmaşıklığına yol açtığından genetik algoritmalar veya SFFS (Sequential Forward Feature Selection) gibi pratik yöntemler seçilir.

- **Süzgeçle seçim** (filter method, feature score metrics):

Her özellik birbirinden bağımsız olarak bir ya da birden fazla metriğe göre değerlendirilir. Bunların arasında en başarılı olanlar seçilir.

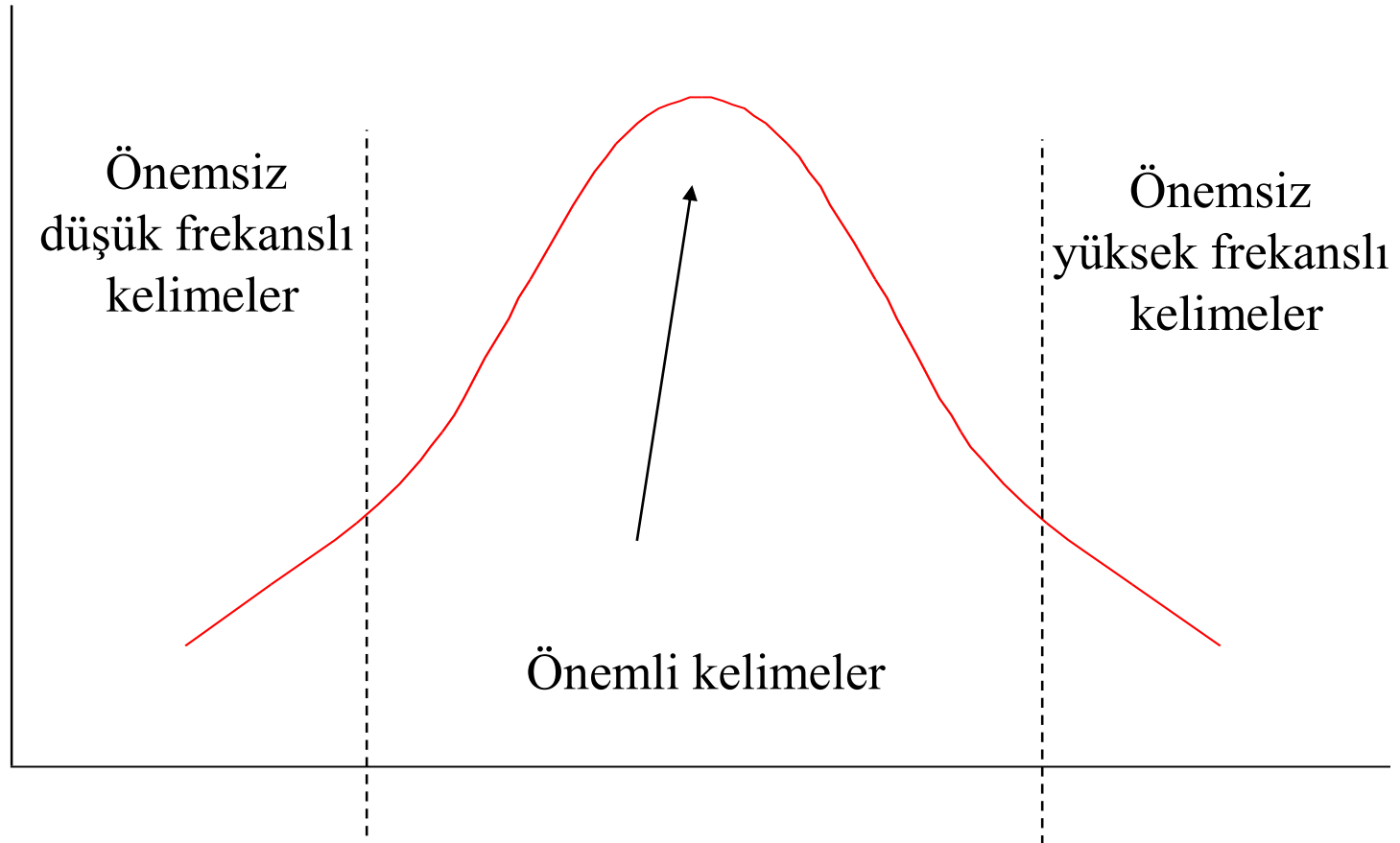


# Stop words - Function words – Etkisiz kelimeler

- Metinlerde geçen bütün kelimeleri kullanmak yerine bir kısmını almasak
  - “bir, ben, o, ve,...” gibi frekansı çok yüksek, ancak bir anlam ifade etmeyen (?) kelimeler (Stop-word elimination)
  - Bütün dokümanlarda sadece 1-3 kere geçen düşük frekanslı kelimeler (Document frequency thresholding)



# Stop - Function words



Frekanslarına göre küçükten büyüğe sıralanmış kelimeler

Dokümanları özellikler cinsinden ifade edebilmek için kullanılan yöntemlere **dizinleme dili** (*indexing language*) denir.

En basit dizinleme dili, “belgelerde geçen her kelimenin bir özellik olarak belirlenmesi ve dokümanları ifade eden özellik vektörleri olarak bu kelimelerin dokümanlarda geçme sıklıklarının kullanılması” olarak tanımlanır. Bazen kelimeler yerine kelime grupları da kullanılır.

**Kelime torbası** (*bag-of-words*) yönteminde öncelikle veri kümesini iyi temsil edebilecek kelimeler oluşturulur. Her doküman, kelimelerin dokümanda bulunma yerleri ve birbirlerine göre konumları dikkate alınmaksızın, bir kelime yığını olarak ifade edilir.



# Gövdemele (Stemming)

- Özellikle Türkçe gibi eklemeli diller için gerekli
- Ağaçlarımı = ağaçlarını = ağaç





# Abstract Feature Extraction (AFE)

- Öz vektör veya Öz değerler kullanılmaz.
- Tekil Değer Ayırıştırımı yapılmaz.
- Terimlerin ağırlıkları ve sınıflar üzerindeki olasılıksal dağılımları kullanılır.
- Terim olasılıklarının sınıflara olan izdüşümü alınıp, bu olasılıklar toplanarak, her terimin sınıfları ne kadar etkilediği bulunur.

i terim sayısı

j doküman sayısı

k sınıf sayısı

$n_{i,j} \rightarrow$  ti teriminin dj dokümanında kaç kere geçtiği

$N_i \rightarrow$  ti terimine sahip olan doküman sayısı

ti teriminin ck sınıfında kaç kere geçtiği

$$nc_{i,k} = \sum_j n_{i,j}, \quad d_j \in c_k$$

dj dokümanında yer alan ti teriminin ck sınıfını ne kadar etkilediği

$$w_{i,k} = \log(nc_{i,k} + 1) \times \log\left(\frac{N}{N_i}\right)$$



Bir belgedeki tüm terimlerin  $c_k$  sınıfına olan toplam etkisi

$$Y_k = \sum_i w_{i,k}, \quad w_i \in d_j$$

Sonuçta  $i$  adet terim, sınıf sayısı kadar boyutlu bir hiper düzleme yansıtılır.

Bu işlem tüm dokümanlar için uygulandığında  $j$  satır (her doküman için bir satır) ve  $k$  sütundan oluşan (çıkarılan özelliklerin sayısı sınıf sayısına eşittir) indirgenmiş sonuç matrisi elde edilir.

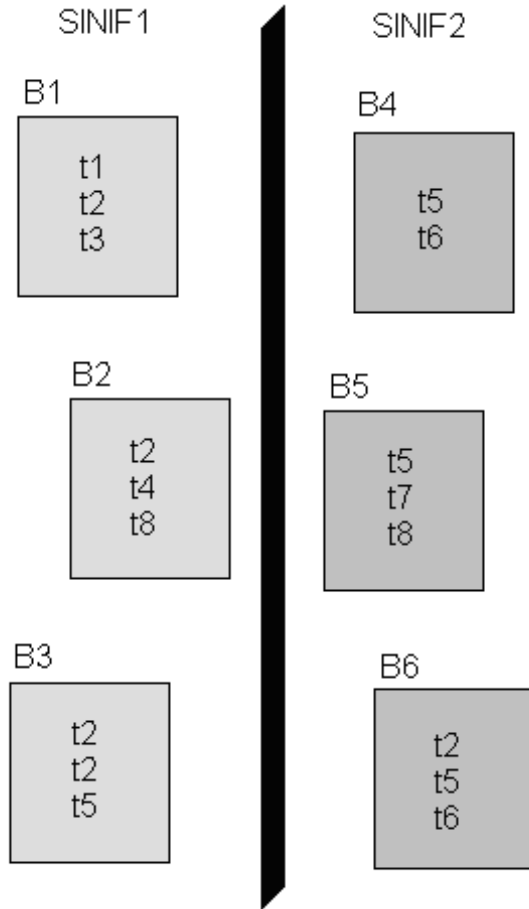
Yeni terimler

$$yeniTerim_k = \frac{Y_k}{\sum_k Y_k} \quad \text{normalize edilir.}$$



## Örnek Uygulama

İki sınıfta toplam 6 doküman ve 8 terimden oluşan bir veri kümesi olsun.



Veri kümesine ait olan terim-doküman matrisi

	d1	d2	d3	d4	d5	d6
Terim1	1	0	0	0	0	0
Terim2	1	1	2	0	0	1
Terim3	1	0	0	0	0	0
Terim4	0	1	0	0	0	0
Terim5	0	0	1	1	1	1
Terim6	0	0	0	1	0	1
Terim7	0	0	0	0	1	0
Terim8	0	1	0	0	1	0



Örnek: Terim5'in Sınıf1 ve Sınıf2 üzerindeki ağırlıklarını hesaplayalım.

$$i=5, k=1 \text{ için } w_{5,1} = \log(1+1) \times \log(6/4) = 0,053$$

$$i=5, k=2 \text{ için } w_{5,2} = \log(3+1) \times \log(6/4) = 0,106$$

	Sınıf1	Sınıf2
<b>Terim1</b>	0.234	0
<b>Terim2</b>	0.123	0.053
<b>Terim3</b>	0.234	0
<b>Terim4</b>	0.234	0
<b>Terim5</b>	0.053	0.106
<b>Terim6</b>	0	0.228
<b>Terim7</b>	0	0.234
<b>Terim8</b>	0.144	0.144



Şimdi Doküman4 için Sınıf1 ve Sınıf2 üzerindeki ağırlıkları hesaplayalım.

$j=4, k=1$  için

$$Y1 = 0,053 + 0 = 0,053$$

$$yeniTerim1 = 0,053/(0,053+0+0,106+0,228) = 0,137 \text{ olarak hesaplanır.}$$

$j=4, k=2$  için

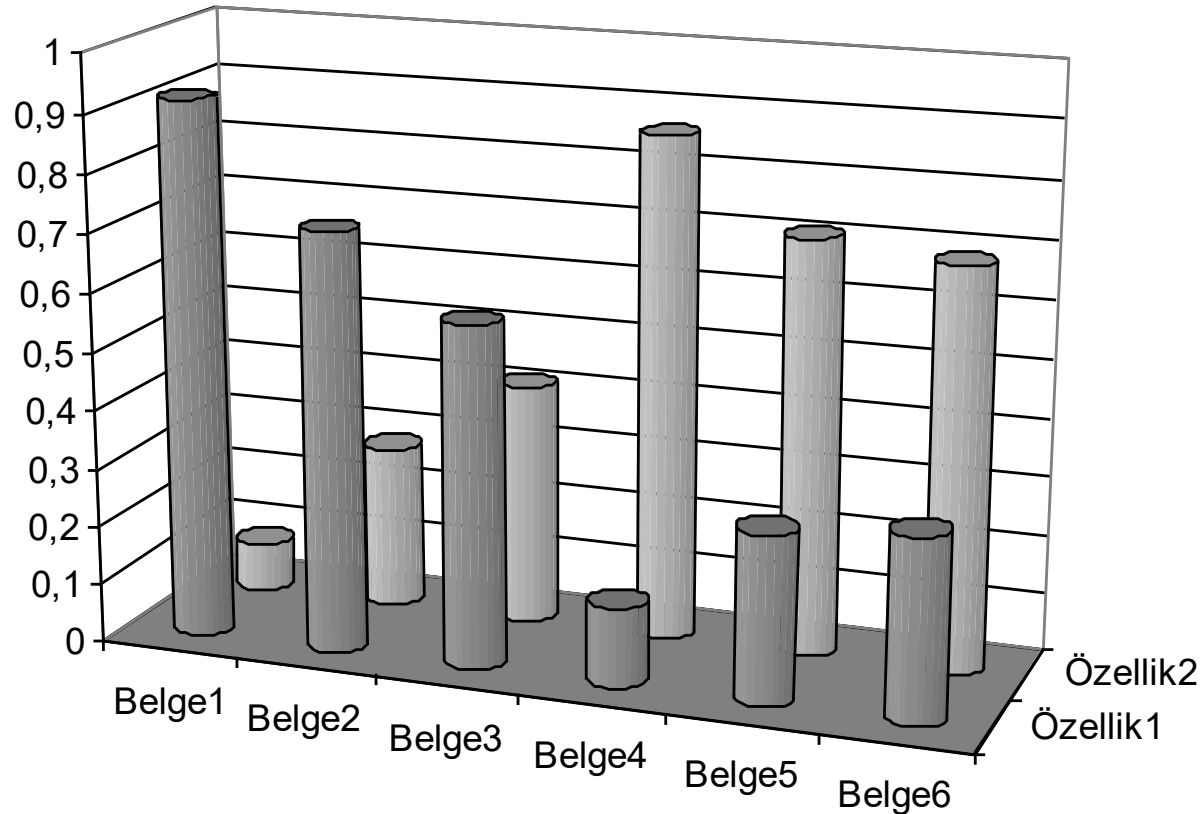
$$Y2 = 0,106 + 0,228 = 0,334 \text{ olarak bulunur.}$$

$$yeniTerim2 = 0,334/(0,053+0+0,106+0,228) = 0,863 \text{ olarak hesaplanır.}$$

	Özellik1	Özellik2
<b>Doküman 1</b>	0.918	0.082
<b>Doküman 2</b>	0.718	0.282
<b>Doküman 3</b>	0.585	0.415
<b>Doküman 4</b>	0.137	0.863
<b>Doküman 5</b>	0.289	0.711
<b>Doküman 6</b>	0.313	0.687

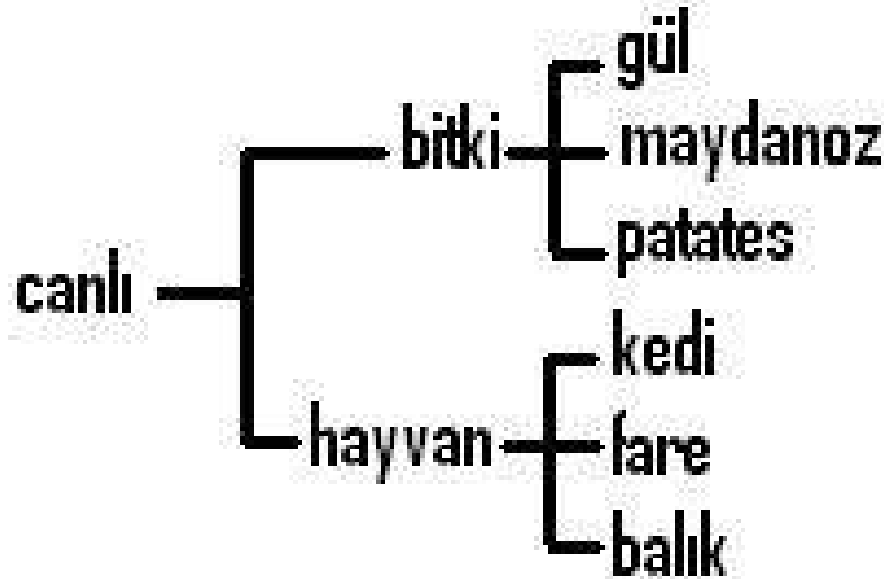


- Çıkarılan özelliklerin aynı zamanda dokümanların sınıflara olan üyelik olasılıkları olarak da okuması mümkündür.
- Doküman 4'ün içeriğine bakıldığında, %12 olasılıkla birinci sınıfa, %87 olasılıkla da ikinci sınıfa ait olduğu anlaşılmaktadır.
- AFE özellik çıkarımı yöntemini aynı zamanda bir sınıflandırıcı olarak da kullanılabilir



# Anlamsal sınıf

- “Tüm X’ler, Y’dir” anlamındaki ilişkilerle birbirine bağlanmış bir kelime ağacı



- Aynı üst kavrama sahip kelimeler, aynı anlamsal sınıftandır.

# Uygulama Alanları

- Metin sınıflandırma
- Otomatik soru cevaplama
- Kelime anlamını durulaştırma
- Otomatik metin özetleme





# Anlamsal sınıfların elde edilmesi

- Elle yapılması çok zahmetlidir
- Otomatik olarak gerçekleştirmek için kelimelerin birbirlerine yakınlığının bir şekilde ölçülmesi gereklidir

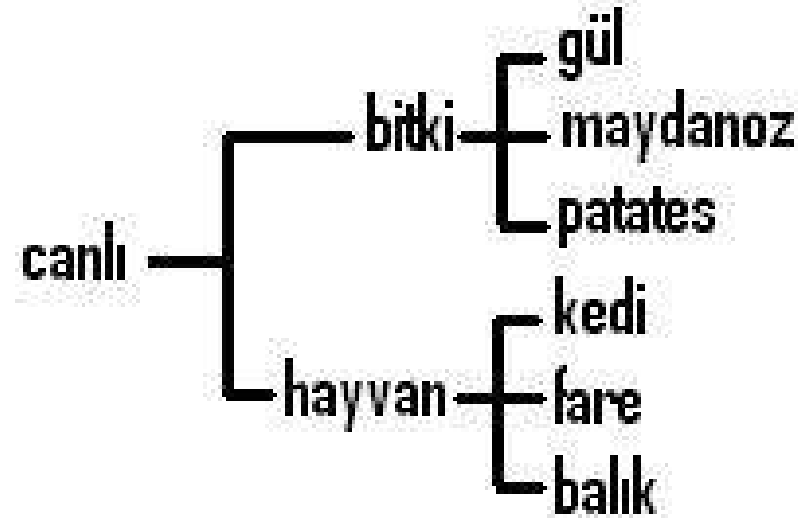


# Kelimelerin anlamsal olarak birbirine yakınlığının bulunması

- 2 tür yaklaşım mevcut
  - Bilgisayarların okuyabildikleri sözcüklerin, kavramsal haritalarının kullanılması
  - Büyük metin kütüphanelerinden elde edilen istatistiklerin kullanılması



# Hazır Kavramsal Haritalar



- İngilizce için Wordnet
- iki kelimeyi birleştiren en kısa yolun uzunluğu  
“balık” ile “kedi” arasındaki yol

*“balık-hayvan-kedi”*

“balık” ile “maydanoz” arasındaki yol

*“balık- hayvan- canlı- bitki- maydanoz”*

- Kedi, balığa maydanozdan daha çok benzer.

# Hazır haritalar yoksa veya yeterli değilse ?

- Büyük metinlerden (Internet) elde edilen istatistikler kullanılabilir.
- Bu bölümde, bu yaklaşım benimsenmiş ve Türkçe kelimeler için kolay uygulanabilir ve etkili bir benzerlik metodu önerilmiştir. Bu metoda göre hesaplanan yakınlıklar kullanılarak kelimelere karşılık gelen vektörler bulunmuştur.



# Anlamsal Benzerlik Ölçümü

- İki kelimenin, İnternet'te yer alan sayfaların kaç tanesinde arka arkaya kullanıldıkları bulunarak çıkarılır.
- Bunun için arama motoruna “kelime1 kelime2” sorgusu gönderilerek gelen sonuç sayfalarındaki sonuç sayısı alınır.



# Kelime Koordinatları-Örnek

- Kelimelerin koordinatlarını bulmak için birlikte geçtikleri doküman sayılarından yararlanılır.
- Birlikte geçtikleri doküman sayılarını nasıl bulabiliriz ?
  - Kendi veri kümemizle
  - Ya da?

	akbaba	ay1	baykuş	araba	limuzin
akbaba		0	20	1	0
ay1	0		33	4	0
baykuş	20	33		0	0
araba	1	4	0		38
limuzin	0	0	0	38	

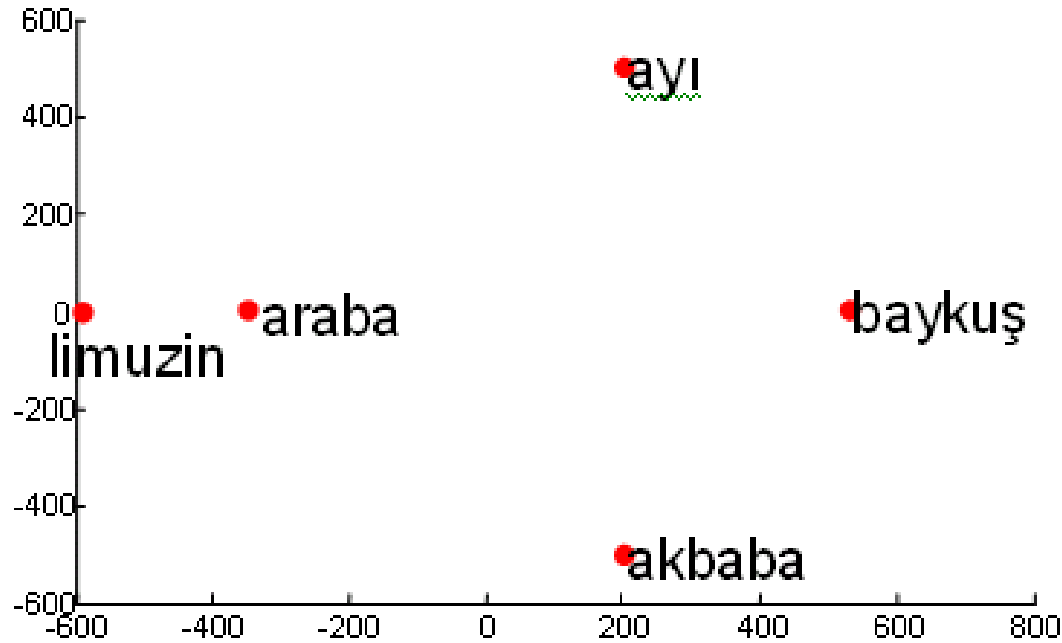


# Çok Boyutlu Ölçekleme (Multi-dimensional Scaling)

- Aralarındaki mesafelerin/yakınlıkların bilindiği ancak uzaysal koordinatlarının bilinmediği durumlarda örneklerin mümkün olduğunca az boyutlu bir uzayda orijinal şekle (eldeki mesafelere) yakın bir biçimde ifade edilmesi için Çok Boyutlu Ölçekleme (ÇBÖ - Multi-dimensional Scaling) metodu kullanılmaktadır.
- Burada örnekler kelimelerimizdir. Kelimelerin arasındaki mesafeler/benzerlikler ise arama motorlarıyla oluşturulan benzerlik matrisidir. ÇBÖ sonucunda kelimelerin  $X$  boyutlu bir uzaydaki koordinatları bulunmaktadır.



# Benzerlik matrisinden çok boyutlu ölçekleme (ÇBÖ) ile elde edilen kelime koordinatları





# Kelimeler

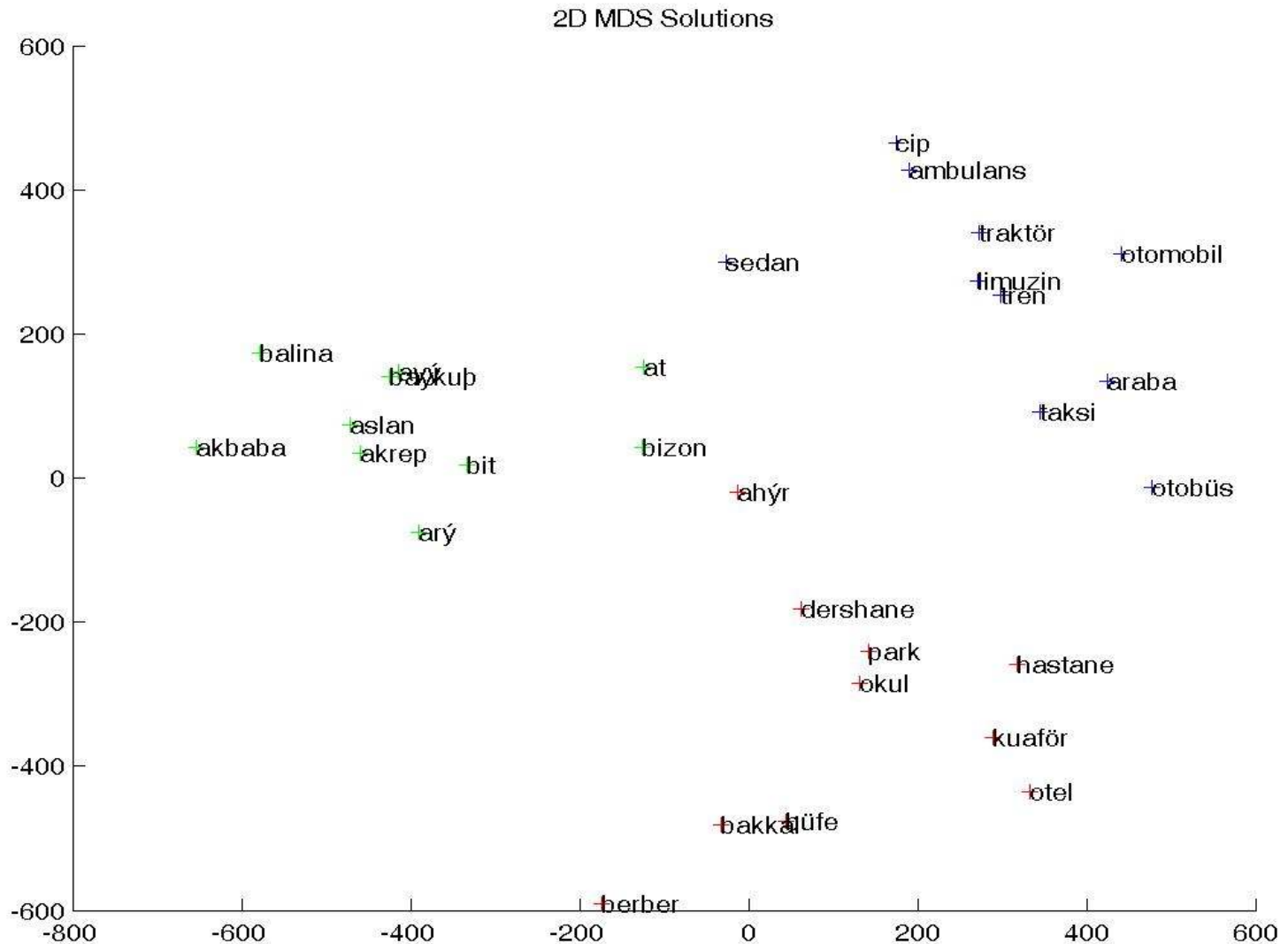
Veri kümesi-I	<i>Yer</i> berber – bakkal- kuaför- hastane- otel- okul- dershane- büfe- ahır- park		<i>Havvan</i> akbaba- arı- baykuş- balina- aslan- at – bizon – akrep - ayı- bit		<i>Taşıt</i> otobüs- cip- taksi- ambulans- araba- sedan- tren- limuzin- otomobil- traktör	
Veri kümesi -II	<i>Ev eşyası</i> divan - televizyon- avize- merdiven- bilgisayar- buzdolabı- kapı- sandalye- lamba- koltuk		<i>Hayvan</i> akbaba- arı- baykuş- balina- aslan- at – bizon – akrep - ayı- bit		<i>Giyecek</i> bikini- mayo- bere- süveter- etek- ayakkabı- eldiven- gömlek- şapka- gözlük	
Veri kümesi -III	<i>Giyecek</i> tereyağı kaymak baharat peynir şeker tuz	<i>Giyecek</i> ayakkabı eldiven şapka gömlek gözlük etek	<i>Ev eşyası</i> avize sandalye lamba koltuk buzdolabı televizyon	<i>Hayvan</i> aslan arı at akrep ayı akbaba	<i>Yer</i> otel dershane park berber hastane okul	<i>Taşıt</i> taksi limuzin ambulans cip otomobil araba

# Denemeler

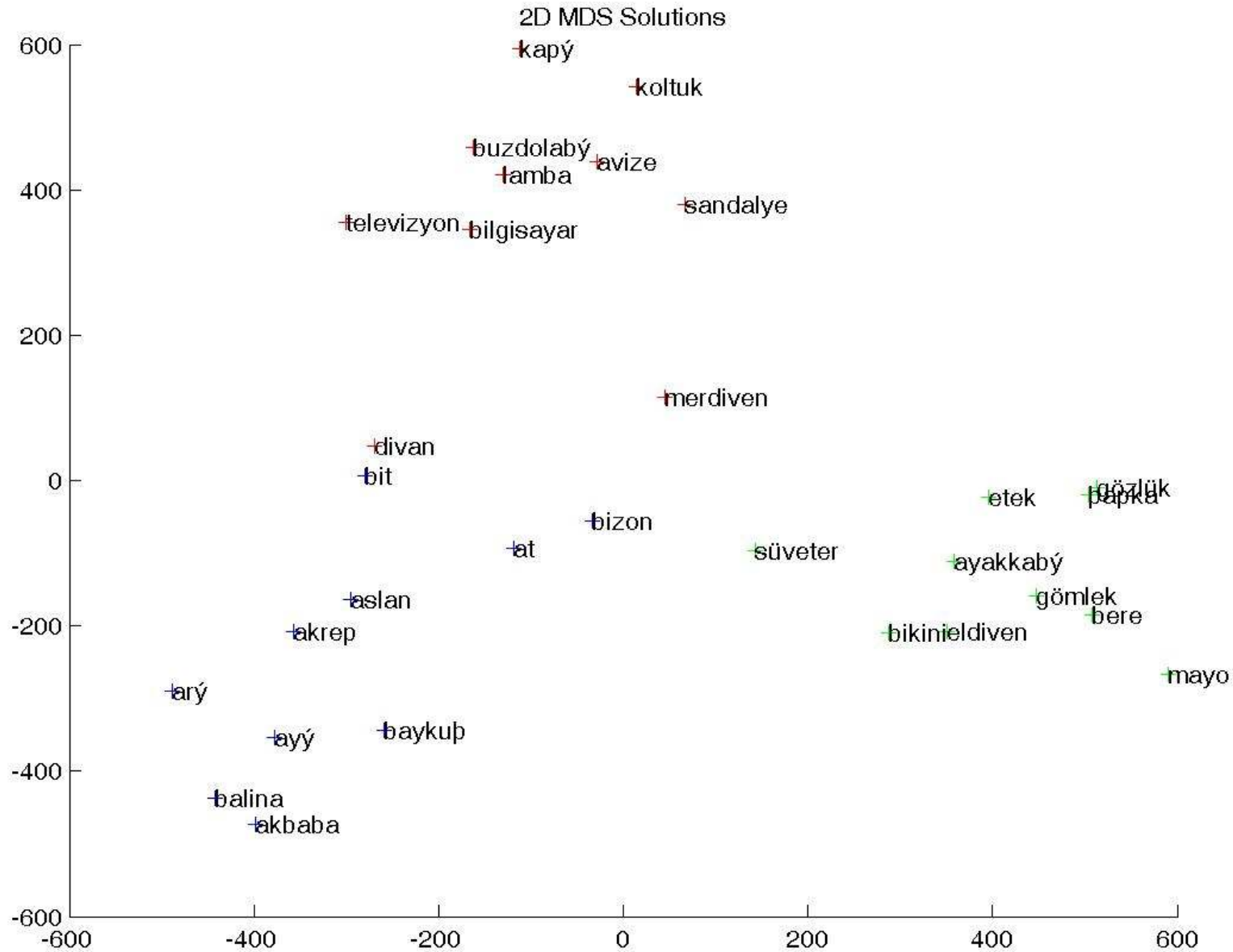
- Her bir veri kümesi için iki farklı arama motorunun sonuçları kullanılarak benzerlik matrisleri oluşturulmuştur.



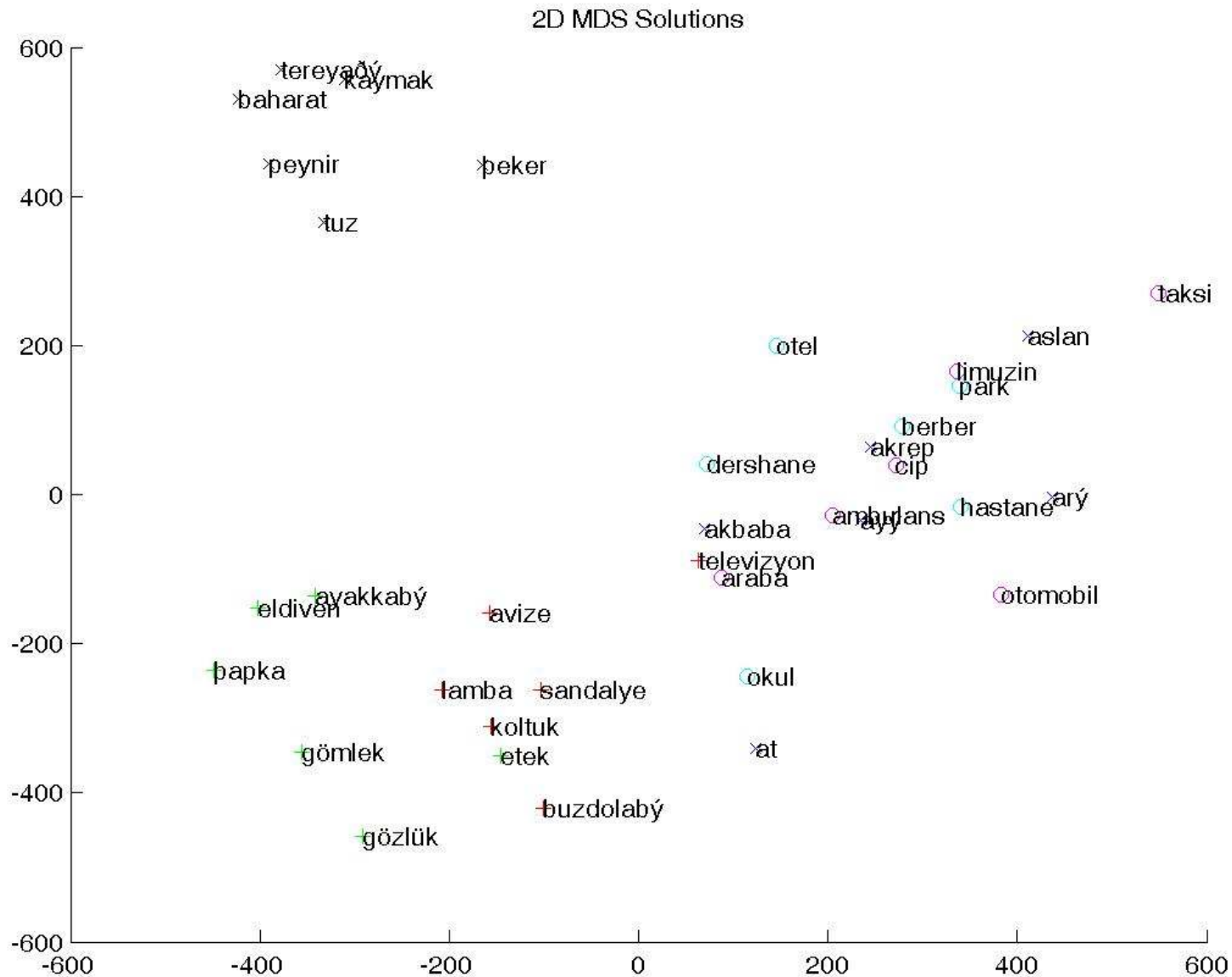
*Veri kümesi-1 için Google ile bulunan benzerlik matrisinden hesaplanmış  
kelime koordinatları*



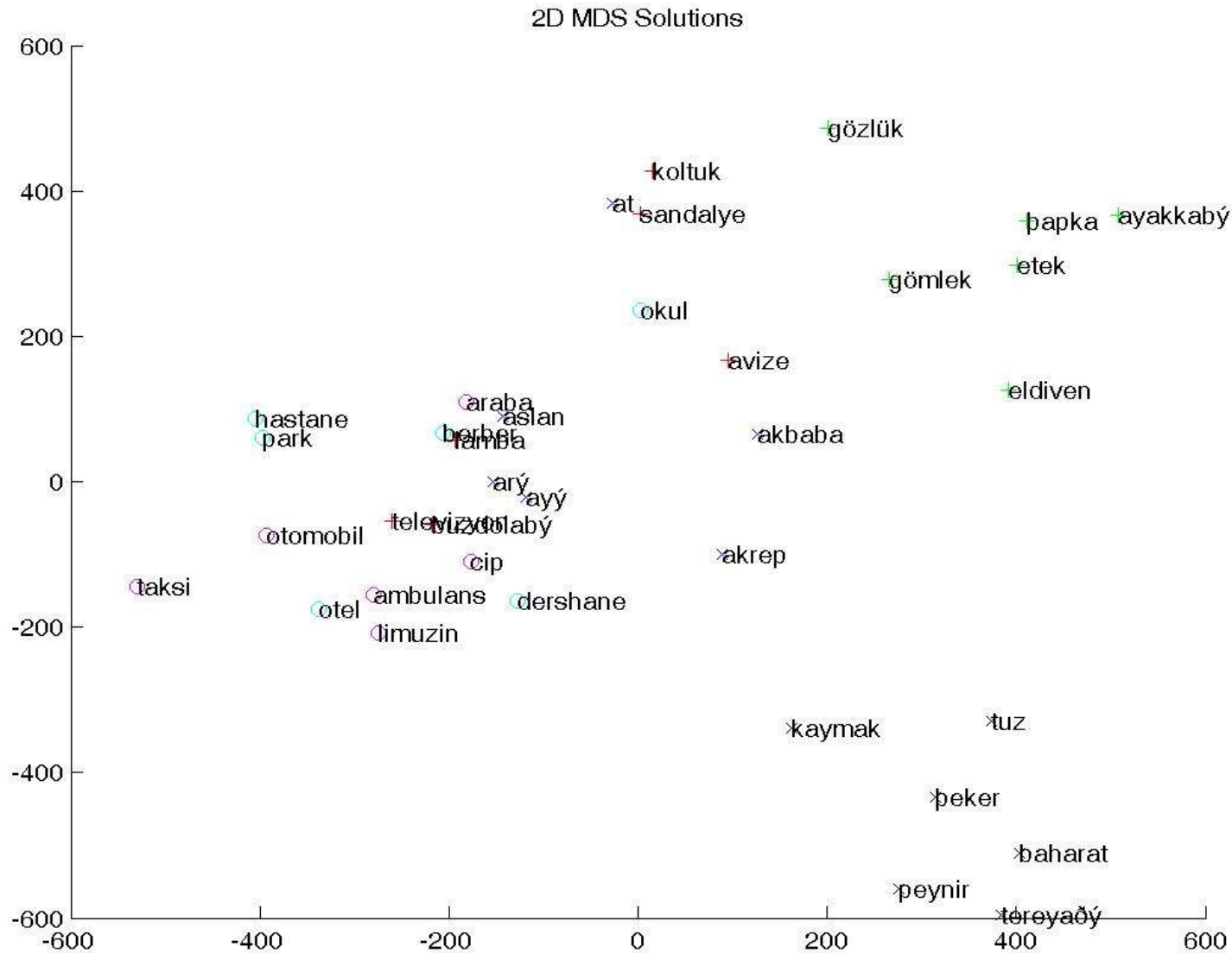
## Veri kümesi-2 için Google ile bulunan benzerlik matrisinden hesaplanmış kelime



## Veri kümesi-3 için Yahoo ile bulunan benzerlik matrisinden hesaplanmış kelime



# Veri kümesi-3 için Google ile bulunan benzerlik matrisinden hesaplanmış kelime koordinatları



# Herhangi bir sınıf/grup bilgisi kullanılmadan elde edilen şekillerde

- Bulunan koordinatlar Türkçe Wordnet'e uygun
- Veri kümesi-1 ve Veri kümesi-2'deki kelimeler koordinatlarına göre grup içi varyansın düşük, gruplar arası varyansın büyük olduğu için çok iyi gruplandırılmışlar.



## Grafiklerin yorumları - devam

- Veri kümesi–3'te ise yiyecek, giyecek ve ev eşyası isimleri diğerlerinden ayrılabilmiştir.
- Yer, taşıt ve hayvan isimleri birbirlerinden ayıramamışlardır.
- Sebep ?





## Sebep-1: Sınıf içindeki kelime azlığı

“akbaba ayı” için sonuç sayısı = 0

“akbaba baykuş” için sonuç sayısı = 20

“ayı baykuş” için sonuç sayısı = 33

- Bu sayede “akbaba” ve “ayı” kelimeleri beraber kullanılmıyor olsalar bile ortak kullanıldıkları kelime olan “baykuş” sayesinde birbirlerine yakın oldukları söylenebilmektedir.
- Bu örnekten yola çıkılarak her bir sınıf içindeki kelime sayısının fazlalığının böyle ortak kullanılan kelime sayısını arttıracak ve bu sayede daha başarılı sınıflandırmalar yapılabileceği düşünülmektedir.



## Sebep-2: Benzerlik Ölçütü

- Kelimelerin benzerlikleri ölçülürken denklem-1 yerine denklem-2 kullanılabilir.

$$\textit{benzerlik}(a,b) = \textit{sayfasayisi}("a \& b")$$

$$\textit{benzerlik}(a,b) = \frac{\textit{sayfasayisi}("a \& b")}{\textit{sayfasayisi}(a) * \textit{sayfasayisi}(b)}$$



- Arama motorlarının performansları arasında belirgin bir fark yoktur
- Sadece Veri kümesi–3 için ev eşyaları sınıfı Yahoo ile Google’dan daha iyi sınıflandırılmıştır.
- Daha büyük ölçekteki verilerle işlem yapıldığında belirgin bir performans farkının ortaya çıkması olasıdır.



# Kelimelerin Sınıflandırılması

- Karar Destek Makineleri (SVM)
- Karar Ağaçları (C4.5)
- Karar Ormanları (Random Forest)
- Kümeleme / gruplama için ise Beklenti Enbüyütme (Expectation Minimization-EM)
- Google'dan elde edilen benzerlik matrisleri
- Korelasyon Tabanlı Özellik seçimi (CFS) ile boyut sayısı azaltma
- 10'lu çapraz geçерleme 10- fold cross validation)



# Sınıflandırma Başarıları

	v1	v1	v2	v2	v3	v3
	10 boyut	2 boyut	10 boyut	2 boyut	10 boyut	4 boyut
SVM	96,6	100	83,3	96,6	88,8	77,7
C4.5	83,3	83,3	90	90	77,7	75
RF	90	90	93,3	93,3	72,2	75
EM	66,6	96,6	66,6	96,6	66,6	83,3



- En başarılı sınıflandırıcı SVM'dir.
- Boyut azaltma genelde başarıyı yükseltir.
- SVM algoritması ile Veri kümesi-1, 2 boyutta %100 başarılı olmuştur.
- 3 veri kümesinde sınıflandırıcıların başarılarının ortalaması %87'dir.
- Ayrıca EM algoritmasıyla hiçbir sınıf bilgisi kullanılmadan düşük boyutlarda ulaşılan başarı oldukça yüksektir.
- Bununla birlikte yapılan denemeler küçük ölçeklidir.
- Daha sağlıklı yorumlara ulaşabilmek için daha fazla sınıf ve kelime içeren veri kümeleriyle çalışmak gerekmektedir.



# Avantajlar

- Önerilen metod sadece Türkçe'ye özgü bir metot değildir. Her dil için kullanılabilir.
- Denemeler küçük ölçekte yapılmış olsalar da görsel temsilleriyle ve başarılı sınıflandırma sonuçlarıyla (% 87) daha geniş çaptaki çalışmalar için umut vericidirler.
- Bulunan koordinatlar sayesinde kelimelerin klasik makine öğrenmesi metotlarıyla sınıflandırma/gruplandırma yapılması mümkündür. Örneğin metin sınıflandırmada metinlerin içindeki kelimelerin koordinatları kullanılabilir.



## *Kısıtlar ve öneriler*

- N adet kelimenin sınıflandırılabilmesi için arama motoruna  $N*(N-1)/2$  adet sorgu gönderilmelidir.
  - Arama motorları sorgu sayısını sınırlandırır
  - N nin büyük değerlerinde benzerlik matrisinin çıkarılması zaman alabilir ☹
  - Bu sorunu çözebilmek için arama motorlarını kullanmak yerine büyük metin kütüphaneleri kullanılabilir
- Kelimelerin benzerliğinin ölçümünde sadece arka arkaya geçtikleri sayfa sayıları kullanılmıştır. Near-yakınında gibi farklı özellikler kullanılabilir.





# Ağırlıklandırma

## *Terim frekansı - Ters doküman frekansı*

(term frequency – inverse document frequency, tf-idf):

- Terimlerin dokümanda geçme sıklığının terimlerin tüm veri kümesinde geçme sıklığına bölünmesi ile hesaplanır. Böylece her belgede bulunan terimler cezalandırılmış olur.
- $TF * IDF = \text{terim frekansı} * \text{ters doküman frekansı}$
- $t_k$  kelimesinin,  $d_j$  dokümanı için ağırlığı

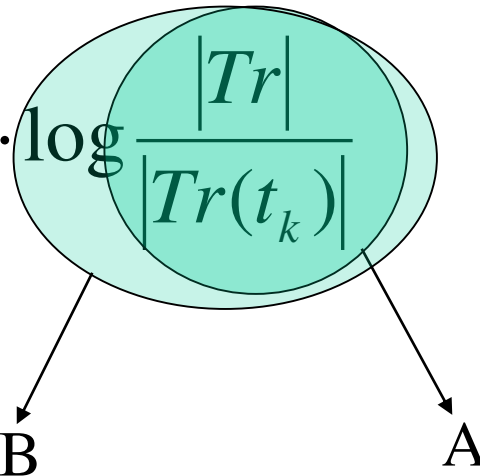
$$tfidf(t_k, d_j) = \#(t_k, d_j) \cdot \log \frac{|Tr|}{|Tr(t_k)|}$$

- $\#(t_k, d_j)$ :  $t_k$  kelimesinin  $d_j$  dokümanında geçme sayısı
- $Tr$ : tüm dokümanların sayısı
- $Tr(t_k)$ : içinde en az bir kere  $t_k$  kelimesi geçen dokümanlar



# Ağırlıklandırma ama Neden ?

Bütün dokümanlarda geçen kelimelerin önemini azaltmak için.

$$tfidf(t_k, d_j) = \#(t_k, d_j) \cdot \log \frac{|Tr|}{|Tr(t_k)|}$$


A'nın payı ve paydası birbirine eşit/yakın olursa 1'e yaklaşır

B'nin içi 1'e yaklaşırsa, B de 0'a yaklaşır ve terimin ağırlığı azalır.

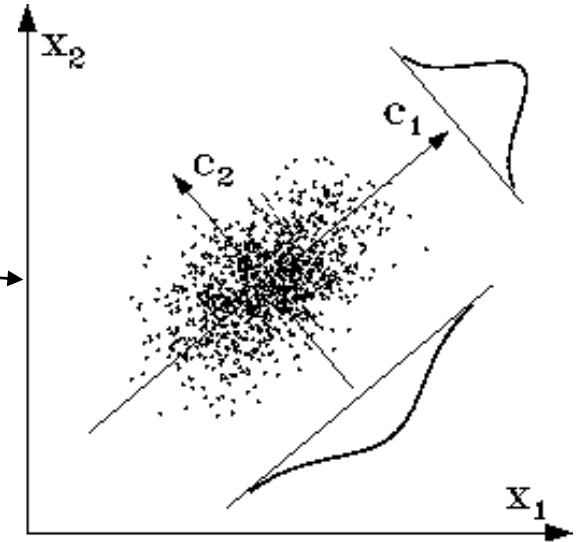
# Projeksiyonlar

- Verileri  $n$  boyuttan,  $r$  boyuta indirgeyen bir projeksiyon matrisi bulunur. Tüm veriler bu projeksiyon matrisiyle çarpılarak boyutları indirgenmiş olur.
- Gerçek veri =  $k$  örnek \*  $n$  boyut
- Projeksiyon matrisi =  $n$  \*  $r$  boyut
- Yeni veri seti = gerçek veri \* projeksiyon matrisi  
 $= (k*n) * (n*r) = (k*r)$  boyut

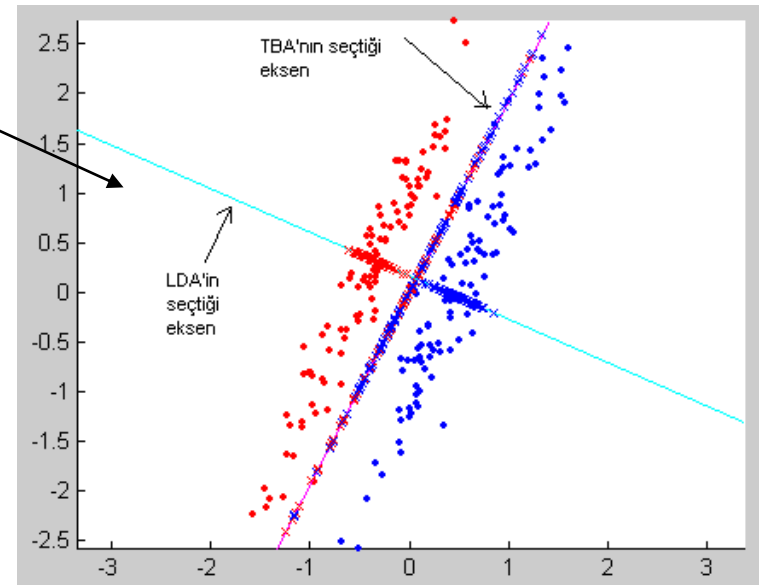


# Projeksiyonlar

- PCA
- LDA
- Latent Semantic Indexing (LSI)



$$D_{l \times n} = U_{l \times r} \begin{pmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_r \end{pmatrix} V_{r \times n}^T$$



- **Ana Bileşenler Analizi (Principal Component Analysis, PCA):** Özellik uzayı, daha küçük bir vektör alt uzayında daha az sayıda koordinat ile temsil edilir. PCA'nın izlediği yol, kovaryans matrisinin işaret ettiği saçılım enerjisinin mümkün olduğunca en büyük kısmını barındıran ortogonal bir koordinat sistemi kurmaktır. PCA koordinat boyutu, tipik olarak enerjinin %90 ya da %95'ini içerecek şekilde seçilir.
- **Doğrusal Ayırtaç Analizi (Linear Discriminant Analysis, LDA):** Özellik uzayı küçültürken, aynı zamanda sınıflar arası farklılaşma en büyük yapılmaya çalışılır. Böylece, hem sınıf içi dağılımların küçük hem de sınıflar arası dağılımın büyük olması istenir. Bu yöntemde koordinat sayısı seçime bağlı değildir ve sınıf sayısından küçüktür. LDA'nın sınıflandırılacak verinin sınıflar üzerinde eşit dağılmadığı durumlarda daha az hata yaptığı gösterilmiştir.
- **Gizli Anlambilimsel Dizinleme (Latent Semantic Indexing, LSI):** Terim – doküman matrisi üzerinde tekil değer ayrışımı (singular value decomposition) uygulayıp, yüksek özdeğerli (eigenvalue) özvektörlerin (eigenvector) oluşturduğu daha küçük bir öznitelik uzayı elde eder. Bu yöntem, terimler arasında gizli olan anlam bilimsel ilişkileri yakalayabilmektedir. Özellikle eş sesli ve eşanlamlı sözcüklerin anlamlarını çıkarma açısından avantaj sağlamaktadır.



# Metin Sınıflandırmada için bir örnek: Naive Bayes

Her özellik için sınıflar içinde bulunma olasılıkları ve sınıfların veri üzerinde görülme olasılıklarını hesaplayarak karar veren bir modeldir. “koşullu bağımsızlık kabulü” ile bir özelliğin bir sınıfta belirli bir olasılıkla geçmesi, bir başka özelliğin aynı sınıfta geçiş olasılığından etkilenmez ve o olasılığı etkilemez.

Dokümanın,  $c$  sınıfına ait olma olasılığı **dokümandaki her kelimenin  $c$  sınıfına ait olma olasılıklarının çarpımının,  $c$  sınıfının olasılığı** ile çarpımına eşittir.

❖ Sınıfı bulunmak istenen  $X$  dokümanının her bir sınıf için olasılık değeri bulunur ve doküman en yüksek olasılığa sahip sınıfa dahil edilir.

$$\operatorname{argmax}_{c_i \in C} P(c_i) \prod_{i=1}^n P(a_i | c_i)$$

**Bir dokümanın  $c_i$  sınıfından olma olasılığı**

**$a_i$  kelimesinin  $c_i$  sınıfında yer alma olasılığı**

$w_k$  : sözlükteki kelimeler olsun

$n_k$  : k. kelimenin frekansı

$P(w_k | v_k)$  : j.sınıfta  $w_k$  nın bulunma olasılığı

$w_k$  kelimesinin  
ilgili sınıftaki  
sayısı

$$P(w_k | v_k) = \frac{n_k + 1}{n_j + |V|}$$

j.sınıfın  
eleman sayısı

Sınıf içerisindeki  
eleman sayısı

$$V_{NB} = \arg \max_{v_j \in V} P(v_j) * \prod_i P(w_i | v_j)$$

## Örnek

Sınıf A : ‘The cat crabs the croll off the stairs’

Sınıf B : ‘It’s rainnig cats and dogs’

**Hangi sınıf ?** ‘Cats eat mice and dogs bury bones’

**Sınıf A** kelimeleri (cat, crab, croll, stair)  $n_1=4$

**Sınıf B** kelimeleri (rain, cat, dog)  $n_2=3$

Ortak sözlük (cat, crab, croll, stair, rain, dog)  $|V| = 6$

Her bir sınıfın olasılığı nedir ?

$$P(A)=1/2 \quad p(B)=1/2$$

Gelen yeni cümle hangi sınıfa aittir?

Test edilecek cümle içerisindeki kelimelerden sözlük içerisinde yer alanlar çıkarılır.

**{ cat, dog }**





{cat, dog} A sınıfına ait olma olasılığı nedir ?

$$P(A) * P(\text{cat}|A) * P(\text{dog}|A) = 1/2 * 2/10 * 1/10 = 0.01$$

$$P(\text{cat}|A) = \frac{1+1}{4+6} = \frac{2}{10}$$

$$P(\text{dog}|A) = \frac{0+1}{4+6} = \frac{1}{10}$$

{cat, dog} B sınıfına ait olma olasılığı nedir ?

$$P(B) * P(\text{cat}|B) * P(\text{dog}|B) = 1/2 * 2/9 * 2/9 = 0.0247$$

$$P(\text{cat}|B) = \frac{1+1}{3+6} = \frac{2}{9}$$

$$P(\text{dog}|B) = \frac{1+1}{3+6} = \frac{2}{9}$$

$$\left. \begin{array}{l} A \rightarrow 0.01 \\ B \rightarrow 0.0247 \end{array} \right\} \max \arg v_j$$

**B** sınıfına aittir



# Sonuç

- Metinlerin makine öğrenmesi metotlarıyla işlenebilmesi için öncelikle sayısallaştırılması gerekir.
- Bu derste bunun için birçok metod gördük.
- Artık metinler sayılar ile ifade edildiğine göre dokümanlar üzerinde kümeleme, sınıflandırma işlemlerini gerçekleştirebilir.



# Kaynaklar

- Alpaydın E. (2004) “Introduction to Machine Learning”, The MIT Press
- Helena Ahonen-Myka, Processing of large document collections
- Philip Koehn, Data Intensive Linguistics — Lecture 12, Text Classification and Clustering
- SUNY Learning Network, Text Classification
- Christopher Manning, Opportunities in Natural Language Processing
- M.Fatih Amasyalı, Arama Motorları Kullanarak Bulunan Anlamsal Benzerlik Ölçütüne Dayalı Kelime Sınıflandırma
- Biricik, G., Diri, B., Sönmez, A.C., "Linear Abstract Feature Extraction for Text Classification", Turkish Journal of Electrical Engineering and Computer Enginerring



# Son ☺

*Sabriniz için  
teşekkürler  
i*

