VERİ BİLİMİ DERSİ

NumPy ve Pandas ile EDA

Keşifsel Veri Analizi: Veriden Anlam Çıkarmanın Temel Araçları ve Teknikleri

Dr. Murat Altun

Veri Bilimi ve Yapay Zekâ Eğitimi · 2026

2

Temel Kütüphane
NumPy + Pandas

%80

Veri bilimcilerin
vaktinin EDA'ya ayrılan oranı

5 dk

CSV'den ilk içgörüye
ortalama süre

∞

Pandas ile
cevaplanabilecek soru

İçindekiler

01 **NumPy Temelleri** — Slayt 3-5

Array oluşturma · Matematiksel işlemler · İstatistik fonksiyonları

02 **Pandas Temelleri** — Slayt 6-9

Series ve DataFrame · CSV/Excel okuma · Veri seçme (loc, iloc)

03 **Keşifsel Veri Analizi** — Slayt 10-16

EDA komutları · Eksik veri · Filtreleme · GroupBy · Apply/Lambda

04 **Uygulama ve Özet** — Slayt 17-22

Titanic EDA · Tips analizi · Kontrol listesi · Hatalar · Ödev

NumPy Nedir?

BÖLÜM 1

Numerical Python

NumPy, Python'da yüksek performanslı sayısal hesaplama için temel kütüphanedir. Çok boyutlu diziler (ndarray) üzerinde vektörize işlemler yaparak döngülerden 10-100x daha hızlı çalışır.

Neden NumPy?

1. Hızlı vektörize işlemler
2. N-boyutlu dizi desteği
3. Matematiksel fonksiyonlar
4. Bellek verimliliği

Python Listesi vs NumPy Array

Özellik	Python Listesi	NumPy Array
Hız	Yavaş (döngü)	Çok hızlı (vektörize)
Bellek	Fazla tüketir	Optimize edilmiş
Tip	Karışık tipler	Homojen (tek tip)
Boyut	1D (iç içe ile nD)	Doğal nD desteği
İşlemler	Manuel döngü	Elementwise otomatik

NumPy Array Oluşturma


```python
import numpy as np

# Listeden array
a = np.array([1, 2, 3, 4, 5])

# Sıfırlardan oluşan 3x3 matris
zeros = np.zeros((3, 3))

# Birlerden oluşan matris
ones = np.ones((2, 4))

# Aralıklı dizi
rng = np.arange(0, 10, 2)  # [0, 2, 4, 6, 8]

# Eşit aralıklı 5 sayı
lin = np.linspace(0, 1, 5)

# Rastgele 3x3 matris
rand = np.random.rand(3, 3)

# Birim matris
eye = np.eye(4)
```

`np.array()`

Listeden NumPy dizisi oluşturur

`np.zeros() / ones()`

Sıfır veya bir dolu matris

`np.arange()`

Belirli adımlarla aralık dizisi

`np.linspace()`

Eşit aralıklı n adet sayı

`np.random.rand()`

0-1 arası rastgele matris

`np.eye()`

nxn birim (identity) matris

NumPy İşlemler: Matematik ve İstatistik

Matematiksel İşlemler

```
a = np.array([10, 20, 30])
b = np.array([1, 2, 3])

print(a + b)    # [11, 22, 33]
print(a * b)    # [10, 40, 90]
print(a ** 2)   # [100, 400, 900]
```

İstatistik Fonksiyonları

```
veri = np.array([85, 90, 78, 92, 88])

np.mean(veri)    # 86.6  (ortalama)
np.median(veri)  # 88.0  (medyan)
np.std(veri)     # 4.84  (std sapma)
np.min(veri)     # 78    (minimum)
np.max(veri)     # 92    (maksimum)
```

mean()

Aritmetik Ortalama

median()

Ortanca Değer

std()

Standart Sapma

var()

Varyans

sum()

Toplam

Pandas Nedir? Series vs DataFrame

Panel Data → Pandas

Pandas, yapılandırılmış (tablo) verilerle çalışmak için Python'un en güçlü kütüphanesidir. İki temel veri yapısı sunar: Series (tek sütun) ve DataFrame (tablo). Veri okuma, temizleme, dönüştürme ve analiz için tek durak çözümdür.

Series (1 Boyutlu)

```
import pandas as pd

notlar = pd.Series(
  [85, 90, 78, 92],
  index=['Ali','Ayşe','Can','Defne']
)
```

- Etiketli tek boyutlu dizi
- Index + değerlerden oluşur
- NumPy array'in etiketli hâli
- Sözlükten de oluşturulabilir

DataFrame (2 Boyutlu)

```
df = pd.DataFrame({
  'İsim': ['Ali','Ayşe','Can'],
  'Not':  [85, 90, 78],
  'Yaş':  [20, 22, 21]
})
print(df.shape)  # (3, 3)
```

- Satır ve sütunlu tablo yapısı
- Her sütun bir Series'dir
- SQL tablosu / Excel sayfası gibi
- Sözlük, liste veya dosyadan oluşturulur

DataFrame Oluşturma ve Dosya Okuma

BÖLÜM 2

CSV Dosya Okuma

```
# Basit okuma
df = pd.read_csv('veri.csv')

# Parametreli okuma
df = pd.read_csv(
  'veri.csv',
  sep=';', encoding='utf-8',
  index_col=0, nrows=1000
)
```

Excel Dosya Okuma

```
# Excel okuma
df = pd.read_excel('veri.xlsx')

# Belirli sayfa
df = pd.read_excel(
  'veri.xlsx',
  sheet_name='Sayfa2',
  header=0
)
```

Diğer Veri Kaynakları ve Kaydetme

Fonksiyon	Açıklama	Örnek
pd.read_json()	JSON dosya okuma	pd.read_json('veri.json')
pd.read_sql()	SQL veritabanından okuma	pd.read_sql(query, conn)
pd.read_html()	Web sayfasından tablo çekme	pd.read_html(url)[0]
df.to_csv()	CSV olarak kaydetme	df.to_csv('sonuc.csv', index=False)
df.to_excel()	Excel olarak kaydetme	df.to_excel('sonuc.xlsx')

Veri Seçme: .loc[] ve .iloc[]

.loc[] — Etiket Bazlı Seçim

```
# Tek satır (etiketle)
df.loc[0]  # 0 indexli satır

# Satır + sütun seçimi
df.loc[0:2, 'İsim':'Not']

# Koşullu seçim
df.loc[df['Not'] > 85]

# Belirli sütunlar
df.loc[:, ['İsim', 'Not']]

# Tek hücre değeri
df.loc[0, 'İsim']  # 'Ali'
```

Etiket (label) ile erişim — aralıklar dahil (inclusive)

.iloc[] — Pozisyon Bazlı Seçim

```
# İlk satır (indexle)
df.iloc[0]  # 0. pozisyon

# İlk 3 satır, ilk 2 sütun
df.iloc[0:3, 0:2]

# Son satır
df.iloc[-1]

# Belirli satır ve sütunlar
df.iloc[[0, 2], [0, 1]]

# Tek hücre
df.iloc[0, 1]  # 85
```

Pozisyon (integer) ile erişim — aralıklar hariç (exclusive)

Koşullu Filtreleme


```python
# Tek koşul
yuksek_not = df[df['Not'] > 85]

# Birden fazla koşul (VE)
filtre = df[(df['Not'] > 80) & (df['Yaş'] < 22)]

# Birden fazla koşul (VEYA)
filtre2 = df[(df['Bölüm'] == 'BT') | (df['Bölüm'] == 'YBS')]

# isin() ile çoklu değer kontrolü
secim = df[df['Şehir'].isin(['İstanbul', 'Ankara', 'İzmir'])]

# Metin içeriğine göre filtreleme
ara = df[df['İsim'].str.contains('Ali')]

# between() ile aralık
aralik = df[df['Not'].between(70, 90)]
```

&

VE Koşulu

Her koşul parantez içinde & operatörü kullan

|

VEYA Koşulu

Her koşul parantez içinde | operatörü kullan

~

DEĞİL (NOT)

Koşulun tersini almak için ~ operatörü kullan

?

isin()

Birden fazla değer kontrolü için idealdir

Temel EDA Komutları

Komut	Açıklama	Ne Döner?
df.shape	Satır ve sütun sayısı	(1000, 8)
df.info()	Sütun tipleri, null sayısı	Özet tablo (konsol)
df.describe()	Sayısal sütunların istatistikleri	count, mean, std, min, max...
df.dtypes	Her sütunun veri tipi	int64, float64, object...
df.head(n)	İlk n satır (varsayılan 5)	DataFrame
df.tail(n)	Son n satır	DataFrame
df.columns	Sütun isimleri listesi	Index objesi
df.nunique()	Her sütundaki benzersiz değer	Sayısal Series
df.value_counts()	Değer frekansları	Sıralı Series

İpucu: Her yeni veri setinde ilk 5 komutu (shape, info, describe, head, dtypes) mutlaka çalıştırın. Bu, verinin genel yapısını anlamanın en hızlı yoludur.

Eksik Veri Analizi

Eksik Veriyi Tespit Etme

```
df.isnull().sum()        # Her sütundaki eksik sayısı
df.isnull().mean() * 100 # Eksik oranı (%)
df.isna().any()          # Hangi sütunda eksik var?
```

Silme: dropna()

```
# Eksik olan satırları sil
df.dropna()

# Sadece tümü eksik olanları sil
df.dropna(how='all')

# Belirli sütuna göre sil
df.dropna(subset=['Not'])

# Eşik: en az 3 dolu değer
df.dropna(thresh=3)
```

Doldurma: fillna()

```
# Sabit değerle doldur
df['Not'].fillna(0)

# Ortalama ile doldur
df['Not'].fillna(df['Not'].mean())

# Medyan ile doldur
df['Not'].fillna(df['Not'].median())

# Önceki değerle doldur (ffill)
df['Not'].fillna(method='ffill')
```

Veri Manipülasyonu: Sıralama ve Filtreleme

BÖLÜM 3

Sıralama: sort_values()

```
# Artan sıralama
df.sort_values('Not')

# Azalan sıralama
df.sort_values('Not', ascending=False)

# Çoklu sütun sıralama
df.sort_values(
  ['Bölüm', 'Not'],
  ascending=[True, False]
)
```

Sütun İşlemleri

```
# Sütun silme
df.drop('Gereksiz', axis=1)

# Sütun adı değiştirme
df.rename(columns={
  'eski_ad': 'yeni_ad'
})

# Tip dönüştürme
df['Not'] = df['Not'].astype(float)
```

Tekrar Eden ve Benzersiz Değerler

```
df.duplicated().sum()          # Tekrar eden satır sayısı
df.drop_duplicates()           # Tekrarları temizle
df['Şehir'].unique()           # Benzersiz değerler
df['Şehir'].value_counts()     # Frekans tablosu
```

GroupBy ile Gruplama ve Özet Tablolar

BÖLÜM 3

```python
# Tek sütuna göre gruplama
df.groupby('Bölüm')['Not'].mean()

# Çoklu aggregation
df.groupby('Bölüm')['Not'].agg(['mean', 'std', 'min', 'max', 'count'])

# Çoklu sütuna göre gruplama
df.groupby(['Bölüm', 'Cinsiyet'])['Not'].mean()

# Farklı sütunlara farklı işlem
df.groupby('Bölüm').agg({
    'Not': 'mean',
    'Yaş': 'median',
    'İsim': 'count'
})
```

Örnek Sonuç Tablosu

Bölüm	Ort. Not	Std Sapma	Öğrenci Sayısı
Bilişim Tek.	82.4	8.3	45
Yönetim Bil.	78.9	10.1	38
İstatistik	86.2	6.7	32
Matematik	84.1	7.5	28

GroupBy Akışı

1. Böl (Split)
2. Uygula (Apply)
3. Birleştir (Combine)

Yeni Sütun Oluşturma: Apply ve Lambda

Basit Yeni Sütun

```
# Doğrudan hesaplama
df['Harf_Notu'] = df['Not'] >= 50

# Matematiksel işlem
df['Not_Yuzde'] = df['Not'] / 100 * 40
```

Apply + Lambda

```
# Lambda ile harf notu
df['Durum'] = df['Not'].apply(
  lambda x: 'Geçti' if x >= 50 else 'Kaldı'
)
```

Fonksiyon ile Apply — Karmaşık Dönüşümler

```
def harf_notu(not_degeri):
    if not_degeri >= 90: return 'AA'
    elif not_degeri >= 80: return 'BA'
    elif not_degeri >= 70: return 'BB'
    else: return 'FF'
```

```
# Fonksiyonu uygula
df['Harf'] = df['Not'].apply(harf_notu)

# map() ile sözlük eşleme
cinsiyet_map = {'E': 'Erkek', 'K': 'Kadın'}
df['Cinsiyet'] = df['C'].map(cinsiyet_map)

# applymap() — tüm hücrelere
df[['N1','N2']].applymap(
  lambda x: round(x, 1)
)
```

Pivot Table ve Çapraz Tablolar

BÖLÜM 3

pd.pivot_table() — Excel Pivot Benzeri

```
# Pivot tablo oluşturma
pivot = pd.pivot_table(
    df,
    values='Not',           # Değer sütunu
    index='Bölüm',          # Satır grupları
    columns='Cinsiyet',     # Sütun grupları
    aggfunc='mean',         # Toplama fonksiyonu
    margins=True            # Genel toplam satırı
)
```

Pivot Sonucu

Bölüm	Erkek	Kadın	Genel
Bilişim Tek.	80.2	84.8	82.4
Yönetim Bil.	76.5	81.3	78.9
İstatistik	85.0	87.4	86.2
Genel	**80.6**	**84.5**	**82.5**

pd.crosstab()

```
# Çapraz tablo
pd.crosstab(
  df['Bölüm'],
  df['Cinsiyet'],
  margins=True,
  normalize='index'  # oran
)
```

String ve Tarih İşlemleri

String (.str) İşlemleri

```
# Küçük/büyük harf
df['İsim'].str.lower()
df['İsim'].str.upper()

# İçerik kontrolü
df['İsim'].str.contains('Ali')
df['İsim'].str.startswith('A')

# Değiştirme ve bölme
df['Tel'].str.replace('-', '')
df['Ad_Soyad'].str.split(' ')

# Uzunluk ve kırpma
df['İsim'].str.len()
df['İsim'].str.strip()
```

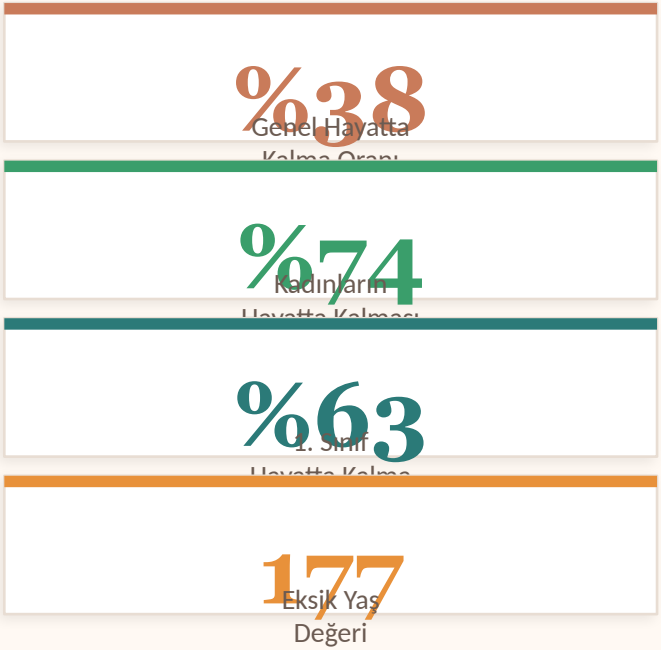
Tarih (datetime) İşlemleri

```
# String → datetime dönüşümü
df['Tarih'] = pd.to_datetime(
  df['Tarih']
)

# Yıl, ay, gün çıkarma
df['Yıl'] = df['Tarih'].dt.year
df['Ay']  = df['Tarih'].dt.month
df['Gün'] = df['Tarih'].dt.day_name()

# Tarih farkı hesaplama
df['Süre'] = (
  df['Bitiş'] - df['Başlangıç']
).dt.days

# Tarih aralığı filtreleme
mask = df['Tarih'] > '2024-01-01'
```

Gerçek Veri Seti: Titanic EDA

Titanic veri seti: 891 yolcu, 12 sütun — hayatta kalma (Survived), cinsiyet (Sex), yaş (Age), bilet sınıfı (Pclass), ücret (Fare) gibi değişkenler içerir.

```python
import pandas as pd

df = pd.read_csv('titanic.csv')

# Genel bakış
print(df.shape)          # (891, 12)
print(df.isnull().sum()) # Age:177 eksik

# Hayatta kalma oranı
df['Survived'].value_counts(normalize=True)
# 0: %61.6  |  1: %38.4

# Cinsiyete göre hayatta kalma
df.groupby('Sex')['Survived'].mean()
# female: 0.74  |  male: 0.19

# Sınıfa göre hayatta kalma
df.groupby('Pclass')['Survived'].mean()
# 1: 0.63  |  2: 0.47  |  3: 0.24
```



Gerçek Veri Seti: Tips (Bahşiş) Analizi


```python
import seaborn as sns

tips = sns.load_dataset('tips')

# Bahşiş oranı hesapla
tips['tip_pct'] = (
  tips['tip'] / tips['total_bill'] * 100
)

# Güne göre bahşiş ortalaması
tips.groupby('day')['tip_pct'].mean()

# Cinsiyet + sigara durumuna göre
pd.pivot_table(
  tips, values='tip_pct',
  index='sex', columns='smoker',
  aggfunc='mean'
)
```

EDA Adımları: 1) shape & info → 2) describe → 3) isnull → 4) value_counts → 5) groupby → 6) yeni sütun → 7) pivot_table → 8) görselleştirme

Güne Göre Bahşiş %

Gün	Ort. Bahşiş %
Perşembe (Thur)	%16.1
Cuma (Fri)	%16.9
Cumartesi (Sat)	%15.3
Pazar (Sun)	%16.7

Öne Çıkan Bulgular

1. Cuma günü bahşiş oranı en yüksek
2. Sigara içenler daha az bahşiş veriyor
3. Erkekler daha yüksek hesap ödüyor
4. Akşam yemeklerinde bahşiş oranı düşük

EDA Kontrol Listesi

1	**Veriyi Yükle**	`pd.read_csv() / pd.read_excel()`
2	**Boyut Kontrol**	`df.shape, df.columns, df.dtypes`
3	**İlk Bakış**	`df.head(), df.tail(), df.sample(5)`
4	**İstatistik Özet**	`df.describe(), df.info()`
5	**Eksik Veri**	`df.isnull().sum(), df.isnull().mean()*100`
6	**Tekrarlar**	`df.duplicated().sum(), df.drop_duplicates()`
7	**Kategorik Analiz**	`df['col'].value_counts(), df.nunique()`
8	**Gruplama**	`df.groupby('col')['val'].agg([...])`
9	**Yeni Sütunlar**	`df['new'] = ..., df['col'].apply(func)`

Pro İpucu: Bu listeyi her yeni veri setine uygulamak, analiz kalitesini ve tutarlılığını önemli ölçüde artırır.

Sık Yapılan Hatalar ve Çözümleri

HATA: `df['Not'] > 80 & df['Yaş'] < 25`

DOĞRU: `(df['Not'] > 80) & (df['Yaş'] < 25)`

Parantez unutulması — & operatörünün önceliği > dan yüksek

HATA: `df.dropna(inplace=True)  # sessiz kayıp`

DOĞRU: `df = df.dropna()  # açık atama`

inplace=True yerine açık atama tercih edin, iz sürülebilirlik artar

HATA: `df[df['Yaş'] == NaN]`

DOĞRU: `df[df['Yaş'].isna()]`

NaN == NaN her zaman False döner, isna() kullanın

HATA: `df.groupby('A').mean()  # SettingWithCopy`

DOĞRU: `df.loc[mask, 'col'] = value  # .loc[] kullanın`

Zincirleme indexleme (chained indexing) kopya üzerinde değişiklik yapar

Haftalık Ödev

Ödev: Titanic Veri Seti ile EDA Raporu

Titanic veri setini (CSV) kullanarak kapsamlı bir Keşifsel Veri Analizi raporu hazırlayın. Jupyter Notebook formatında teslim edin.

1	**Veri Yükleme ve İlk Bakış**	CSV oku, shape, info, describe, head — ilk 5 gözlemi incele
2	**Eksik Veri Analizi**	Her sütundaki eksik veri oranını hesapla, uygun strateji ile doldur/sil
3	**Keşifsel Analiz**	Kategorik: value_counts() | Sayısal: describe() | Korelasyon matrisi
4	**GroupBy Analiz**	Cinsiyet, sınıf ve binme limanına göre hayatta kalma analizi
5	**Feature Engineering**	En az 2 yeni sütun oluştur (yaş grubu, aile büyüklüğü vb.)

Teslim: Hafta 3 dersine kadar · Format: Jupyter Notebook (.ipynb) · Bonus: Matplotlib/Seaborn ile en az 3 görselleştirme ekleyin

Özet ve Çıkarımlar

1. NumPy, Python'da sayısal hesaplamanın temelidir — vektörize işlemlerle performans kazanın
2. Pandas, yapılandırılmış verilerle çalışmanın en güçlü aracıdır — Series ve DataFrame'i iyi öğrenin
3. EDA, her veri projesinin ilk ve en kritik adımıdır — sistematik yaklaşın, kontrol listesini takip edin
4. Eksik veri ve veri tipleri göz ardı edilmemeli — isnull(), dtypes ve describe() ile başlayın
5. GroupBy ve Apply, veriyi anlamlı özetlere dönüştürür — pivot table ile görselleştirmeye hazırlayın

“Veri bilimi, merak ve kodun buluştuğu noktada başlar.”