1. Ay 1. Hafta - Pandas ile Veri Manipülasyonu Teorik Eğitim

🎯 Bu Haftanın Öğrenme Hedefleri

Bu hafta sonunda şunları bileceksiniz:

- Pandas'ın veri bilimindeki rolü ve önemi
- Series ve DataFrame yapılarının mantığı ve kullanım alanları
- Veri okuma/yazma işlemlerinin farklı yöntemleri
- Veri temizleme süreçlerinin kritik önemi
- Veri filtreleme ve seçim mantığı
- Temel veri dönüştürme teknikleri

📚 Pandas'a Giriş: Veri Biliminin Omurgası

Pandas Nedir ve Neden Bu Kadar Önemli?

Pandas, Python ekosisteminde veri analizi için geliştirilmiş en güçlü kütüphanelerden biridir. Adını "Panel Data"dan alan bu kütüphane, finansal analiz için tasarlanmış olmasına rağmen bugün her türlü veri analizi işleminde kullanılmaktadır.

Pandas'ın Güçlü Yanları:

- **Hız**: C dilinde yazılmış alt yapısı sayesinde büyük veri setlerinde bile hızlı çalışır
- **Esneklik**: Excel'den SQL'e, JSON'dan CSV'ye kadar onlarca format destekler
- **Güçlü API**: Kompleks veri operasyonlarını tek satırda yapabilme imkanı
- **Memory Efficiency**: Bellek kullanımını optimize eder
- **Ecosystem Integration**: NumPy, Matplotlib, Scikit-learn gibi kütüphanelerle mükemmel uyum

Excel vs Pandas: Neden Pandas?

Excel'in Sınırları:

- Maksimum 1,048,576 satır sınırı
- Büyük dosyalarda yavaş performans
- Versiyon kontrolü zorluğu
- Otomasyonda sınırlı imkanlar
- Tekrarlanabilir analiz zorluğu

Pandas'ın Avantajları:

- Milyonlarca satırla çalışabilme
- Programatik kontrol ve otomasyon
- Versiyon kontrolü (Git uyumluluğu)
- Karmaşık veri dönüştürmeleri
- Tekrarlanabilir ve paylaşılabilir analizler

🏗️ Veri Yapıları: Series ve DataFrame

Series: Tek Boyutlu Veri Yapısı

Series, Pandas'ın temel veri yapılarından biridir. Tek boyutlu, etiketli bir veri dizisidir. Excel'deki tek bir sütunu düşünebilirsiniz.

Series'in Anatomisi:

- **Values (Değerler)**: Gerçek veriyi barındıran array
- **Index (İndeks)**: Her değeri tanımlayan etiketler
- **Name (İsim)**: Series'in ismi (opsiyonel)
- **dtype**: Veri tipi (int64, float64, object vb.)

Series Kullanım Alanları:

- Zaman serisi verileri (günlük satışlar, hisse fiyatları)
- Kategorik veriler (şehir isimleri, ürün kategorileri)
- Sayısal diziler (yaşlar, puanlar, maaşlar)

DataFrame: İki Boyutlu Veri Yapısı

DataFrame, Series'lerin bir araya gelmesiyle oluşan iki boyutlu veri yapısıdır. Excel tablosu veya SQL tablosu gibi düşünebilirsiniz.

DataFrame'in Anatomisi:

- **Columns (Sütunlar)**: Her sütun bir Series'dir
- **Index (Satır İndeksi)**: Her satırı tanımlayan etiketler
- **Values**: 2D NumPy array olarak saklanan değerler
- **Shape**: (satır_sayısı, sütun_sayısı) tuple'ı

DataFrame'in Güçlü Yanları:

- Farklı veri tiplerini aynı yapıda saklayabilme
- SQL benzeri operasyonları destekleme
- Eksik verileri otomatik yönetme
- Büyük veri setlerinde efficient çalışma

📖 Veri Okuma ve Yazma: Dünyayla Bağlantı

CSV Dosyaları: Veri Alışverişinin Dili

CSV (Comma Separated Values), veri alışverişinde en yaygın kullanılan formattır. Pandas'ın CSV desteği çok güçlüdür.

CSV Okumada Önemli Parametreler:

- **encoding**: Türkçe karakterler için 'utf-8' kritik
- **sep**: Ayırıcı karakter (virgül, noktalı virgül, tab)

- **header**: Başlık satırının konumu
- **index_col**: Hangi sütunun index olacağı
- **usecols**: Sadece belirli sütunları okuma
- **nrows**: Sınırlı sayıda satır okuma
- **skiprows**: Belirli satırları atlama

CSV Yazmada Dikkat Edilecekler:

- **index**: Index'in dosyaya yazılıp yazılmayacağı
- **encoding**: Türkçe karakterlerin korunması
- **sep**: Hedef sistemle uyumlu ayırıcı seçimi

Excel Dosyaları: İşletme Dünyasının Standardı

Excel dosyaları iş dünyasında hala çok yaygın. Pandas'ın Excel desteği oldukça kapsamlı.

Excel Okumada Avantajlar:

- Birden fazla sheet okuyabilme
- Karmaşık formatları anlayabilme
- Formülleri değerlendirebilme
- Metadata bilgilerini koruma

Excel Yazmada İmkanlar:

- Birden fazla sheet'e yazabilme
- Formatting bilgilerini koruma
- Charts ve pivot tabloları dahil etme

🧹 Veri Temizleme: Analizin Temeli

Eksik Veriler: Kaçınılmaz Gerçek

Gerçek dünya verilerinde eksik veri kaçınılmazdır. Pandas bu durumu çok iyi yönetir.

Eksik Veri Türleri:

- **None**: Python'un eksik veri gösterimi
- **NaN (Not a Number)**: NumPy'ın eksik veri gösterimi
- **NaT (Not a Time)**: Pandas'ın zaman verisi için eksik gösterimi
- **pd.NA**: Pandas'ın yeni generic eksik veri gösterimi

Eksik Veri Sebepleri:

- Veri toplama sırasında hatalar
- Sistem arızaları
- Kullanıcı hataları
- Gizlilik sebepleri
- Ölçüm zorluğu

Eksik Veri İşleme Stratejileri:

1. **Silme (Deletion)**
    - Listwise deletion: Tüm eksik veri içeren satırları silme
    - Pairwise deletion: Sadece ilgili analiz için eksik olanları silme
    - Ne zaman kullanılır: Eksik veri oranı düşükse (%5'ten az)
2. **Doldurma (Imputation)**
    - Mean/Median imputation: Ortalama/medyan ile doldurma
    - Mode imputation: En sık görülen değerle doldurma
    - Forward/Backward fill: Önceki/sonraki değerle doldurma
    - Interpolation: Matematiksel interpolasyon
3. **Modelleme**
    - KNN imputation: En yakın komşuların ortalaması
    - Regression imputation: Regresyon modeli ile tahmin
    - Multiple imputation: Birden fazla tahmin modelinin ortalaması

Tekrarlanan Veriler: Kalite Sorunu

Tekrarlanan veriler veri kalitesini düşürür ve analizleri yanıltabilir.

Tekrar Sebepleri:

- Veri girişi hataları
- Sistem entegrasyonu sorunları
- Farklı kaynaklardan veri birleştirme
- Kullanıcı davranışı

Tekrar Tespit Yöntemleri:

- Tam eşleşme: Tüm sütunlar aynı
- Kısmi eşleşme: Belirli sütunlar aynı
- Fuzzy matching: Benzer ama tamamen aynı olmayan

🔍 Veri Seçimi ve Filtreleme: Doğru Verileri Bulma

İndeksleme Mantığı

Pandas'ta veri seçimi çok güçlü ve esnek bir sistemdir.

İndeksleme Türleri:

- **Label-based (loc)**: Etiket adlarına göre seçim
- **Position-based (iloc)**: Pozisyona göre seçim
- **Boolean indexing**: True/False maskesi ile seçim
- **Query method**: SQL benzeri sorgu dili

Boolean İndeksleme: Güçlü Filtreleme

Boolean indeksleme, Pandas'ın en güçlü özelliklerinden biridir.

Mantıksal Operatörler:

- **&**: VE (AND) - Her iki koşul da doğru olmalı
- **|**: VEYA (OR) - En az bir koşul doğru olmalı
- **~**: DEĞİL (NOT) - Koşulun tersi
- **^**: XOR - Sadece bir koşul doğru olmalı

Karşılaştırma Operatörleri:

- **==**: Eşit
- **!=**: Eşit değil
- **>**, **<**: Büyük/küçük
- **>=**, **<=**: Büyük eşit/küçük eşit
- **isin()**: Liste içinde var mı
- **str.contains()**: String içeriyor mu

Query Method: SQL Benzeri Syntax

Query methodu, SQL bilgisi olanlar için çok tanıdık bir syntax sunar.

Query Avantajları:

- Daha okunabilir kod
- Karmaşık koşulları daha kolay yazma
- Variable substitution desteği
- Performance optimizasyonları

📊 Veri Dönüştürme ve Gruplandırma

Apply Fonksiyonları: Güçlü Dönüştürme

Apply fonksiyonları, DataFrame'in her satırına veya sütununa özel işlemler uygulamanıza olanak tanır.

Apply Türleri:

- **DataFrame.apply()**: Satır veya sütun bazında işlem
- **Series.apply()**: Her elemana işlem
- **DataFrame.applymap()**: Her hücreye işlem (deprecated)
- **DataFrame.map()**: Series için mapping işlemi

Lambda Fonksiyonları: Lambda fonksiyonları, kısa ve basit işlemler için idealdir. Tek satırda fonksiyon tanımlamanıza olanak tanır.

GroupBy: Analitik Düşüncenin Temeli

GroupBy işlemi, veri analizinde "split-apply-combine" paradigmasını uygular.

GroupBy Süreci:

1. **Split**: Veriyi gruplara ayırma
2. **Apply**: Her gruba işlem uygulama
3. **Combine**: Sonuçları birleştirme

Aggregation Fonksiyonları:

- **count()**: Sayma
- **sum()**: Toplama
- **mean()**: Ortalama
- **median()**: Medyan
- **std()**: Standart sapma
- **min()**, **max()**: Minimum/maksimum
- **first()**, **last()**: İlk/son değer
- **agg()**: Özel aggregation

Pivot Tables: Excel'in Pandas'taki Karşılığı

Pivot tablolar, veriyi özetlemek için çok güçlü araçlardır.

Pivot Table Bileşenleri:

- **Index**: Satır başlıkları
- **Columns**: Sütun başlıkları
- **Values**: Özetlenecek değerler
- **Aggfunc**: Özet fonksiyonu
- **Fill_value**: Eksik değerler için dolgu

🎲 Veri Tipleri ve Memory Optimization

Pandas Veri Tipleri

Doğru veri tipi seçimi hem performans hem de memory açısından kritiktir.

Temel Veri Tipleri:

- **int64**: Tam sayılar (64-bit)
- **float64**: Ondalık sayılar (64-bit)
- **object**: String ve karışık tipler
- **bool**: True/False değerleri
- **datetime64**: Tarih ve zaman
- **category**: Kategorik veriler
- **string**: Özel string tipi (yeni)

Memory Optimization:

- **int32 vs int64**: Küçük sayılar için int32 kullanımı
- **category**: Tekrarlayan string'ler için ideal
- **sparse**: Çoğu değeri aynı olan veriler için
- **nullable integers**: Eksik değer içeren tam sayılar

String İşlemleri: Metin Verilerinin Gücü

String işlemleri, modern veri analizinde çok önemlidir.

String Accessor (.str): Pandas'ta string işlemleri için özel accessor vardır.

Yaygın String İşlemleri:

- **Temizleme**: strip(), replace(), clean()
- **Dönüştürme**: upper(), lower(), title()
- **Bölme**: split(), partition()
- **Birleştirme**: cat(), join()
- **Arama**: contains(), startswith(), endswith()
- **Regex**: extract(), findall(), match()

🚀 Performans ve Best Practices

Memory Efficiency

Büyük veri setleriyle çalışırken memory yönetimi kritiktir.

Memory Tasarrufu Teknikleri:

- Gereksiz sütunları okumama
- Doğru veri tiplerini kullanma
- Kategorik veriler için category dtype
- Chunk'lar halinde okuma
- Gereksiz kopyaları önleme

Vectorization vs Loops

Pandas'ta vectorized işlemler her zaman loops'tan hızlıdır.

Vectorization Avantajları:

- C-level implementasyon
- Parallel processing imkanı
- Memory locality
- Daha temiz kod

Kaçınılması Gerekenler:

- DataFrame'de for loop kullanımı
- apply() yerine vectorized işlemler mevcut ise
- Gereksiz intermediate DataFrames
- Chain indexing

Code Organization

Temiz ve maintainable kod yazma prensipleri:

Best Practices:

- Descriptive variable names
- Function decomposition
- Error handling
- Documentation
- Version control

🎯 Gerçek Dünya Uygulamaları

Business Intelligence

Pandas, business intelligence'ta çok yaygın kullanılır:

Typical Use Cases:

- Sales analysis
- Customer segmentation
- Financial reporting
- Performance dashboards
- Trend analysis

Data Preprocessing for ML

Machine learning'de veri ön işleme aşamasında Pandas vazgeçilmezdir:

ML Pipeline'da Pandas:

- Feature engineering
- Data cleaning
- Normalization
- Train/test split preparation
- Cross-validation setup

Financial Analysis

Finans sektöründe Pandas çok güçlüdür:

Financial Applications:

- Portfolio analysis
- Risk assessment
- Algorithmic trading
- Market research
- Compliance reporting

🧠 Teorik Kavramların Pratikte Birleşimi

Veri Analizi Süreci

Pandas ile tipik bir veri analizi süreci şu adımları içerir:

1. **Data Loading**: Veriyi okuma ve ilk inceleme
2. **Data Exploration**: Veri kalitesi ve yapı analizi
3. **Data Cleaning**: Eksik ve hatalı verileri düzeltme
4. **Data Transformation**: Analiz için uygun hale getirme
5. **Data Analysis**: İstatistiksel analiz ve pattern arama
6. **Data Visualization**: Sonuçları görselleştirme
7. **Reporting**: Bulguları raporlama

Pandas Ecosystem

Pandas tek başına güçlüdür ama ecosystem'in parçası olarak daha da güçlü hale gelir:

Core Libraries:

- **NumPy**: Numerical computing foundation
- **Matplotlib/Seaborn**: Visualization
- **Scikit-learn**: Machine learning
- **Statsmodels**: Statistical analysis
- **Jupyter**: Interactive development

Extended Ecosystem:

- **Dask**: Parallel computing
- **Modin**: Pandas acceleration
- **Vaex**: Big data exploration
- **Polars**: Fast DataFrame library
- **Apache Arrow**: Columnar data format

💡 Öğrenme Stratejileri ve Tavsiyeleri

Effective Learning Approach

Pandas öğrenirken en etkili yaklaşım:

1. **Start Small**: Küçük veri setleriyle başlayın
2. **Practice Daily**: Her gün biraz pratik yapın
3. **Real Data**: Toy dataset'ler yerine gerçek verilerle çalışın
4. **Read Documentation**: Official documentation'ı okuyun
5. **Join Community**: Stack Overflow, Reddit gibi topluluklar

Common Pitfalls

Yeni başlayanların sık yaptığı hatalar:

- **SettingWithCopyWarning**: Chain indexing yapmak
- **Memory Issues**: Büyük veri setlerini yanlış yükleme
- **Performance**: Vectorization yerine loops kullanma
- **Data Types**: Yanlış veri tipi seçimi

- **Index Reset**: Index'i reset etmeyi unutma

Advanced Topics to Explore

Temel seviyeden sonra keşfedilecek konular:

- **MultiIndex**: Hierarchical indexing
- **Time Series**: Temporal data analysis
- **Categorical Data**: Category dtype optimization
- **Extension Arrays**: Custom data types
- **Styling**: DataFrame presentation
- **Performance Tuning**: Optimization techniques

📝 Bu Haftanın Teorik Özeti

Temel Kavramlar

Bu hafta öğrendiğiniz temel kavramlar:

1. **Pandas'ın Rolü**: Veri biliminde neden kritik olduğu
2. **Veri Yapıları**: Series ve DataFrame'in mantığı
3. **I/O Operations**: Veri okuma/yazma süreçleri
4. **Data Quality**: Temizleme ve validasyon
5. **Data Selection**: Filtreleme ve seçim teknikleri
6. **Data Transformation**: Dönüştürme ve gruplama

Kritik Noktalar

Unutmamanız gereken kritik noktalar:

- **Vectorization is Key**: Her zaman vectorized işlemleri tercih edin
- **Memory Matters**: Büyük veri setlerinde memory yönetimi kritik
- **Data Quality First**: Temizleme analiz kadar önemli
- **Documentation**: Kod ve süreçlerinizi dokümante edin
- **Practice with Real Data**: Gerçek verilerle çalışın

Gelecek Hafta Hazırlığı

NumPy haftasına hazırlık için:

- Pandas'taki matematiksel işlemleri gözden geçirin
- Array düşüncesini kavramaya çalışın
- Linear algebra temellerini hatırlayın
- Scientific computing kavramlarını araştırın

🎖️ Başarı Kriterleri

Bu haftanın sonunda kendinizi test edin:

Theoretical Understanding

- [ ] Pandas'ın veri bilimindeki rolünü açıklayabiliyorum
- [ ] Series ve DataFrame arasındaki farkları biliyorum
- [ ] Eksik veri işleme stratejilerini anlıyorum
- [ ] Boolean indexing mantığını kavradım
- [ ] GroupBy paradigmasını anlıyorum

Conceptual Mastery

- [ ] Hangi durumlarda hangi veri tipini kullanacağımı biliyorum
- [ ] Memory optimization prensiplerini anlıyorum
- [ ] Vectorization'ın önemini kavradım
- [ ] Real-world veri problemlerini Pandas ile nasıl çözeceğimi biliyorum

Bu teorik temeli attıktan sonra, pratik uygulamalara geçmeye hazırsınız! 🚀