Ders Adı: Python ile İstatistik Uygulamaları

Ders Kodu: IST-XXX
Kredi: 3
Ön Koşullar: Temel İstatistik ve Python Programlama Bilgisi
Dersin Süresi: 14 Hafta

Dersin Amacı

Bu ders, Python programlama dili kullanılarak temel ve ileri düzey istatistiksel analizlerin nasıl gerçekleştirileceğini öğretmeyi amaçlamaktadır. Öğrenciler, veri analizi, görselleştirme, olasılık dağılımları, hipotez testleri ve regresyon analizi gibi konuları uygulamalı olarak öğrenecektir.

Ders İçeriği

Hafta 1: Dersin Tanıtımı ve Python'a Giriş

- Dersin kapsamı ve beklentiler
- Python temel veri yapıları (list, tuple, dictionary, set)
- NumPy ve Pandas kütüphanelerine giriş

Hafta 2: Veri Okuma, Temizleme ve Manipülasyon

- CSV, Excel ve SQL'den veri okuma
- Eksik verilerle başa çıkma
- Veri dönüştürme ve filtreleme

Hafta 3: Veri Görselleştirme

- Matplotlib ve Seaborn kullanımı
- Grafik türleri ve yorumlanması
- Veri dağılımlarının görselleştirilmesi

Hafta 4: Açıklayıcı İstatistikler

- Merkezi eğilim ve yayılım ölçüleri
- Özet istatistikler (ortalama, medyan, mod, standart sapma vb.)
- Boxplot ve histogramlarla analiz

Hafta 5: Olasılık Dağılımları

- Temel olasılık kavramları
- Binom, Poisson ve Normal dağılımlar
- Rastgele sayı üretimi

Hafta 6: Örnekleme ve Örnekleme Dağılımları

- Basit rastgele örnekleme
- Örneklem istatistiklerinin dağılımı
- Merkezî Limit Teoremi

Hafta 7: Hipotez Testleri - I

- Z-testi ve t-testi
- Tek ve çift örneklem testleri
- p-değeri ve anlamlılık kavramı

Hafta 8: Hipotez Testleri - II

- Khi-kare testi
- ANOVA
- Çoklu karşılaştırmalar

Hafta 9: Korelasyon ve Regresyon Analizi

- Pearson ve Spearman korelasyon katsayıları
- Basit doğrusal regresyon modeli
- Model değerlendirme

Hafta 10: Çoklu Regresyon ve Model Seçimi

- Çoklu regresyon modeli
- Değişken seçimi ve model optimizasyonu
- AIC, BIC, R^2 gibi ölçütler

Hafta 11: Zaman Serisi Analizi

- Zaman serisi veri yapısı
- Hareketli ortalamalar ve trend analizi
- ARIMA modeline giriş

Hafta 12: Kümeleme Analizi ve Makine Öğrenmesi

- K-means kümeleme
- Hiyerarşik kümeleme
- Temel makine öğrenmesi modelleri

Hafta 13: Büyük Veri ve Python ile Veri İşleme

- Büyük veri kavramı
- Dask ve PySpark kullanımı
- Performans iyileştirme yöntemleri

Hafta 14: Proje Sunumları ve Genel Değerlendirme

- Öğrenci projelerinin sunumu
- Derste öğrenilen tekniklerin değerlendirilmesi
- Genel tekrar

Değerlendirme Kriterleri

- **Ara Sınav (%30)**
- **Ödevler ve Uygulamalar (%20)**
- **Final Sınavı (%40)**
- **Proje Çalışması (%10)**

Ders Materyalleri

- **Kitaplar:**
  - "Think Stats: Exploratory Data Analysis in Python" – Allen B. Downey
  - "Python for Data Analysis" – Wes McKinney
- **Kaynaklar:**
  - Python resmi dokümantasyonu (https://docs.python.org/)
  - Pandas, NumPy, Seaborn ve Scikit-learn belgeleri

Öğretim Yöntemleri: Ders anlatımı, uygulamalı çalışmalar, ödevler ve proje tabanlı öğrenme.

Açıklamalı:

Ders Adı: Python ile İstatistik Uygulamaları

Ders Kodu: IST-XXX
Kredi: 3
Ön Koşullar: Temel İstatistik ve Python Programlama Bilgisi
Dersin Süresi: 14 Hafta

Dersin Amacı

Bu ders, Python programlama dili kullanılarak temel ve ileri düzey istatistiksel analizlerin nasıl gerçekleştirileceğini öğretmeyi amaçlamaktadır. Öğrenciler, veri analizi, görselleştirme, olasılık dağılımları, hipotez testleri ve regresyon analizi gibi konuları uygulamalı olarak öğrenecektir.

Ders İçeriği

Hafta 1: Dersin Tanıtımı ve Python'a Giriş

- Dersin kapsamı ve beklentiler
- Python temel veri yapıları:
    - **List:** Dinamik yapısı, indeksleme, dilimleme, liste metodları (append, extend, pop, remove, sort vb.)
    - **Tuple:** Değişmez yapısı, listelerle farkları, indeksleme, dilimleme
    - **Dictionary:** Anahtar-değer yapısı, temel metodlar (get, keys, values, items vb.)
    - **Set:** Kümeler ve özellikleri, kesişim, birleşim, fark işlemleri
    - **List comprehensions ve dictionary comprehensions** kullanımı
    - **Veri yapılarını dönüştürme** teknikleri
- NumPy ve Pandas kütüphanelerine giriş

Hafta 2: Veri Okuma, Temizleme ve Manipülasyon

- CSV, Excel ve SQL'den veri okuma
- Eksik verilerle başa çıkma
- Veri dönüştürme ve filtreleme

Hafta 3: Veri Görselleştirme

- Matplotlib ve Seaborn kullanımı
- Grafik türleri ve yorumlanması
- Veri dağılımlarının görselleştirilmesi

Hafta 4: Açıklayıcı İstatistikler

- Merkezi eğilim ve yayılım ölçüleri
- Özet istatistikler (ortalama, medyan, mod, standart sapma vb.)
- Boxplot ve histogramlarla analiz

Hafta 5: Olasılık Dağılımları

- Temel olasılık kavramları
- Binom, Poisson ve Normal dağılımlar
- Rastgele sayı üretimi

Hafta 6: Örnekleme ve Örnekleme Dağılımları

- Basit rastgele örnekleme
- Örneklem istatistiklerinin dağılımı
- Merkezî Limit Teoremi

Hafta 7: Hipotez Testleri - I

- Z-testi ve t-testi
- Tek ve çift örneklem testleri
- p-değeri ve anlamlılık kavramı

Hafta 8: Hipotez Testleri - II

- Khi-kare testi
- ANOVA
- Çoklu karşılaştırmalar

Hafta 9: Korelasyon ve Regresyon Analizi

- Pearson ve Spearman korelasyon katsayıları
- Basit doğrusal regresyon modeli
- Model değerlendirme

Hafta 10: Çoklu Regresyon ve Model Seçimi

- Çoklu regresyon modeli
- Değişken seçimi ve model optimizasyonu
- AIC, BIC, R^2 gibi ölçütler

Hafta 11: Zaman Serisi Analizi

- Zaman serisi veri yapısı
- Hareketli ortalamalar ve trend analizi
- ARIMA modeline giriş

Hafta 12: Kümeleme Analizi ve Makine Öğrenmesi

- K-means kümeleme
- Hiyerarşik kümeleme
- Temel makine öğrenmesi modelleri

Hafta 13: Büyük Veri ve Python ile Veri İşleme

- Büyük veri kavramı
- Dask ve PySpark kullanımı
- Performans iyileştirme yöntemleri

Hafta 14: Proje Sunumları ve Genel Değerlendirme

- Öğrenci projelerinin sunumu
- Derste öğrenilen tekniklerin değerlendirilmesi
- Genel tekrar

Değerlendirme Kriterleri

- **Ara Sınav (%30)**
- **Ödevler ve Uygulamalar (%20)**
- **Final Sınavı (%40)**
- **Proje Çalışması (%10)**

Ders Materyalleri

- **Kitaplar:**
  - "Think Stats: Exploratory Data Analysis in Python" – Allen B. Downey
  - "Python for Data Analysis" – Wes McKinney
- **Kaynaklar:**
  - Python resmi dokümantasyonu (https://docs.python.org/)
  - Pandas, NumPy, Seaborn ve Scikit-learn belgeleri

Öğretim Yöntemleri: Ders anlatımı, uygulamalı çalışmalar, ödevler ve proje tabanlı öğrenme.

1. Hafta: Dersin Tanıtımı ve Python'a Giriş

Dersin Hedefleri:

- Dersin kapsamını ve beklentileri açıklamak
- Python programlama dilinin temel yapısını tanıtmak
- Python'daki temel veri yapılarını (list, tuple, dictionary, set) öğretmek
- NumPy ve Pandas kütüphanelerine kısa bir giriş yapmak

---

1. Ders Saati: Dersin Tanıtımı ve Python'a Genel Bakış

1. **Dersin Tanıtımı ve Beklentiler**
    - Dersin işleyişi, proje ve sınav değerlendirme kriterleri
    - Python'un istatistik ve veri analizi için neden kullanıldığı
2. **Python'a Giriş**
    - Python'un temel sözdizimi (syntax)
    - Değişken tanımlama ve veri türleri
    - Matematiksel işlemler ve operatörler

Uygulama:
Öğrencilere basit bir Python programı yazdırılır:

```
python
KopyalaDüzenle
print("Merhaba, Python!")
x = 5
y = 10
print("Toplam:", x + y)
```

---

2. Ders Saati: Python Temel Veri Yapıları

1. **List (Liste)**
    - Dinamik yapısı ve elemanlara erişim
    - Liste metodları: `append()`, `remove()`, `sort()`, `reverse()` vb.
    - Liste içindeki elemanları döngü ile işleme

    Örnek:

    ```
    python
    KopyalaDüzenle
    meyveler = ["Elma", "Armut", "Çilek"]
    meyveler.append("Muz")
    print(meyveler)
    ```

2. **Tuple (Demet)**
    - Değişmez (immutable) yapısı
    - İndeksleme ve dilimleme
    - Tuple ile list arasındaki farklar

Örnek:

```
python
KopyalaDüzenle
renkler = ("Kırmızı", "Mavi", "Yeşil")
print(renkler[1])  # Mavi
```

---

3. Ders Saati: Dictionary (Sözlük) ve Set (Küme) Kullanımı

1. **Dictionary (Sözlük)**
   - Anahtar-değer (key-value) yapısı
   - Dictionary metodları: `keys()`, `values()`, `items()`, `get()`
   - Dictionary'ler ile veri saklama ve erişim

Örnek:

```
python
KopyalaDüzenle
ogrenci = {"ad": "Ahmet", "yas": 21, "bolum": "İstatistik"}
print(ogrenci["ad"])  # Ahmet
```

2. **Set (Küme)**
   - Benzersiz elemanlardan oluşur
   - Kümeler arası işlemler: birleşim, kesişim, fark
   - `add()`, `remove()`, `union()`, `intersection()` kullanımı

Örnek:

```
python
KopyalaDüzenle
kume1 = {1, 2, 3, 4}
kume2 = {3, 4, 5, 6}
print(kume1.intersection(kume2))  # {3, 4}
```

---

4. Ders Saati: NumPy ve Pandas Kütüphanelerine Giriş

1. **NumPy Kullanımı**
   - Neden NumPy? (Büyük veri kümeleri ve hızlı işlemler)
   - NumPy dizileri oluşturma ve işlemler
   - `numpy.array()`, `reshape()`, `mean()`, `std()` vb. fonksiyonlar

Örnek:

```
python
KopyalaDüzenle
import numpy as np
dizi = np.array([1, 2, 3, 4, 5])
print(np.mean(dizi))  # Ortalama hesaplama
```

2. **Pandas Kullanımı**
   - Veri çerçeveleri (DataFrame) ve seriler (Series)

- CSV dosyası okuma
- `head()`, `describe()`, `info()` metodları

Örnek:

```
python
KopyalaDüzenle
import pandas as pd
df = pd.read_csv("veri.csv")
print(df.head())  # İlk 5 satırı göster
```

Haftanın Ödevi:

- Python'da liste, tuple, dictionary ve set kullanarak bir öğrenci not takip sistemi oluşturma
- NumPy ve Pandas kullanarak basit bir veri seti analizi yapma

Python'da **liste, tuple, dictionary ve set** kullanarak bir **öğrenci not takip sistemi**:

Özellikler:

- Öğrenci ekleme
- Not ekleme ve güncelleme
- Öğrencinin notlarını görüntüleme
- En yüksek ve en düşük notları bulma
- Bütün öğrencileri listeleme

Kod:

```
python
KopyalaDüzenle
# Öğrenci not takip sistemi
class NotTakipSistemi:
    def __init__(self):
        self.ogrenciler = {}  # Dictionary: Öğrencileri ve notlarını saklar

    def ogrenci_ekle(self, ogrenci_no, ad, soyad):
        if ogrenci_no in self.ogrenciler:
            print("Bu öğrenci zaten ekli!")
        else:
            self.ogrenciler[ogrenci_no] = {"ad": ad, "soyad": soyad,
"notlar": []}
            print(f"{ad} {soyad} sisteme eklendi.")

    def not_ekle(self, ogrenci_no, not_degeri):
        if ogrenci_no in self.ogrenciler:
            self.ogrenciler[ogrenci_no]["notlar"].append(not_degeri)
```

```
            print(f"{self.ogrenciler[ogrenci_no]['ad']} için {not_degeri}
notu eklendi.")
        else:
            print("Öğrenci bulunamadı!")

    def notlari_goster(self, ogrenci_no):
        if ogrenci_no in self.ogrenciler:
            ogrenci = self.ogrenciler[ogrenci_no]
            print(f"\n{ogrenci['ad']} {ogrenci['soyad']} - Notlar:
{ogrenci['notlar']}")
        else:
            print("Öğrenci bulunamadı!")

    def en_yuksek_not(self, ogrenci_no):
        if ogrenci_no in self.ogrenciler and
self.ogrenciler[ogrenci_no]["notlar"]:
            return max(self.ogrenciler[ogrenci_no]["notlar"])
        return None

    def en_dusuk_not(self, ogrenci_no):
        if ogrenci_no in self.ogrenciler and
self.ogrenciler[ogrenci_no]["notlar"]:
            return min(self.ogrenciler[ogrenci_no]["notlar"])
        return None

    def tum_ogrencileri_listele(self):
        print("\n--- Sistemdeki Öğrenciler ---")
        for ogr_no, ogrenci in self.ogrenciler.items():
            print(f"{ogr_no}: {ogrenci['ad']} {ogrenci['soyad']}")

# Örnek Kullanım
sistem = NotTakipSistemi()

# Öğrenci ekleme
sistem.ogrenci_ekle(101, "Ali", "Yılmaz")
sistem.ogrenci_ekle(102, "Zeynep", "Kaya")

# Not ekleme
sistem.not_ekle(101, 85)
sistem.not_ekle(101, 92)
sistem.not_ekle(102, 78)

# Notları görüntüleme
sistem.notlari_goster(101)
sistem.notlari_goster(102)

# En yüksek ve en düşük notları gösterme
print(f"Ali'nin en yüksek notu: {sistem.en_yuksek_not(101)}")
print(f"Zeynep'in en düşük notu: {sistem.en_dusuk_not(102)}")

# Tüm öğrencileri listeleme
sistem.tum_ogrencileri_listele()
```

Kullanılan Veri Yapıları:

✅ **Dictionary (`ogrenciler`):** Öğrencilerin bilgilerini saklamak için kullanıldı.
✅ **List (`notlar`):** Öğrencinin aldığı notları saklamak için kullanıldı.

✅ **Tuple (Ad & Soyad):** Öğrenci adı ve soyadı dictionary içinde saklanabilir.
✅ **Set (Gelecekte Kullanım):** Benzersiz öğrenci numaralarını tutmak için düşünülebilir.

Jupyter Notebook'a Uygun Not Takip Sistemi

```
python
KopyalaDüzenle
import pandas as pd  # Veri çerçevesi (DataFrame) oluşturmak için
kullanılır

# Öğrenci not takip sistemi
class NotTakipSistemi:
    def __init__(self):
        self.ogrenciler = {}  # Dictionary: Öğrenci bilgilerini saklar

    def ogrenci_ekle(self, ogrenci_no, ad, soyad):
        """Yeni bir öğrenci ekler."""
        if ogrenci_no in self.ogrenciler:
            print("⚠️ Bu öğrenci zaten ekli!")
        else:
            self.ogrenciler[ogrenci_no] = {"Ad": ad, "Soyad": soyad,
"Notlar": []}
            print(f"✅ {ad} {soyad} başarıyla eklendi!")

    def not_ekle(self, ogrenci_no, not_degeri):
        """Belirtilen öğrenciye not ekler."""
        if ogrenci_no in self.ogrenciler:
            self.ogrenciler[ogrenci_no]["Notlar"].append(not_degeri)
            print(f"✅ {self.ogrenciler[ogrenci_no]['Ad']} için
{not_degeri} notu eklendi.")
        else:
            print("⚠️ Öğrenci bulunamadı!")

    def notlari_goster(self, ogrenci_no):
        """Belirtilen öğrencinin notlarını tablo olarak gösterir."""
        if ogrenci_no in self.ogrenciler:
            ogrenci = self.ogrenciler[ogrenci_no]
            notlar = ogrenci["Notlar"]
            if not notlar:
                print(f"ℹ️ {ogrenci['Ad']} {ogrenci['Soyad']} henüz not
almadı.")
            else:
                df = pd.DataFrame({"Notlar": notlar})
                print(f"\n📊 {ogrenci['Ad']} {ogrenci['Soyad']} - Not
Listesi:")
                display(df)
        else:
            print("⚠️ Öğrenci bulunamadı!")

    def ogrenci_listesi(self):
        """Tüm öğrencileri pandas tablosu olarak gösterir."""
        if not self.ogrenciler:
            print("⚠️ Sistemde öğrenci bulunmamaktadır.")
```

```
            return

        data = [{"Numara": no, "Ad": o["Ad"], "Soyad": o["Soyad"], "Not
Sayısı": len(o["Notlar"])}
                for no, o in self.ogrenciler.items()]
        df = pd.DataFrame(data)
        print("\n📌 Sistemdeki Öğrenciler:")
        display(df)

# 🔹 Jupyter Notebook İçin Interaktif Kullanım 🔹
sistem = NotTakipSistemi()

# Öğrenci ekleme
sistem.ogrenci_ekle(101, "Ali", "Yılmaz")
sistem.ogrenci_ekle(102, "Zeynep", "Kaya")

# Not ekleme
sistem.not_ekle(101, 85)
sistem.not_ekle(101, 92)
sistem.not_ekle(102, 78)

# Notları görüntüleme
sistem.notlari_goster(101)
sistem.notlari_goster(102)

# Tüm öğrencileri listeleme
sistem.ogrenci_listesi()
```

https://www.youtube.com/watch?v=bEcxnR2V-_8

📝 Yapılan Güncellemeler:

✅ **Jupyter Notebook uyumlu hale getirildi**
✅ **Pandas eklendi:** Notları ve öğrenci listesini **tablo formatında** göstermek için `DataFrame` kullanıldı
✅ **Interaktif kullanım:** `display()` fonksiyonu ile tablolar görüntülenebilir
✅ **Uyarılar ve bilgilendirmeler eklendi**

Bu haliyle Jupyter Notebook'ta daha okunaklı ve kullanışlı olacaktır.

Google Colab

https://colab.research.google.com/#

Visual Studio Code | Jupyter NoteBook Kullanım
https://www.youtube.com/watch?v=I2v3fUVW7SQ