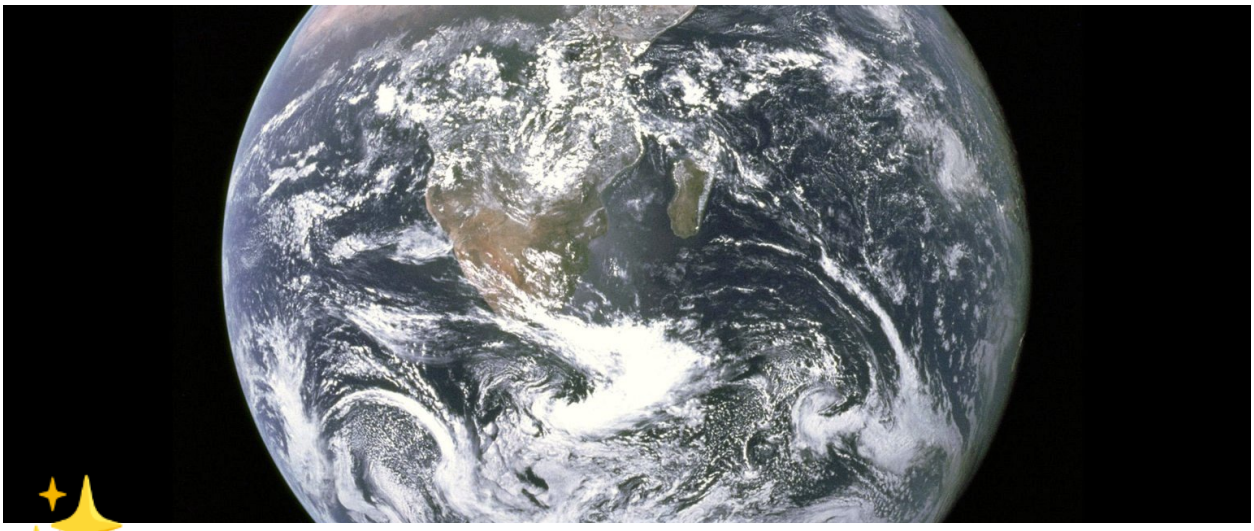


Keşifsel Veri Analizi (EDA) - ML

🧹 Veri Temizleme (Data Cleaning)

Keşifsel veri analizine giriş

Keşifsel Veri Analizi (Exploratory Data Anlaysis) Nedir?

İstatistikte, keşifsel veri analizi, genellikle istatistiksel grafikler ve diğer veri görselleştirme yöntemlerini kullanarak temel özelliklerini özetlemek için veri kümelerini analiz etme yaklaşımıdır. İstatistiksel bir model kullanılabilir veya kullanılamaz.

EDA neden kullanışlıdır?

- EDA veriler için ilk izlenim ve gözlem oluşturmamızı sağlar.
- Bu şekilde verilerin anlamlı olup olmadığını ya da daha fazla temizliğe ihtiyacı olup olmadığını anlayabiliriz.
- Verilerdeki kalıpları ve eğilimleri belirlememize yardımcı olur.

EDA teknikleri

- **Özetleyici nitelikler -** Veri hakkında bilgilenmemizi sağlayacak özet verilerdir. Bunlar ortalama, medyan , maksimum değer , korelasyon vs.

- **Görselleştirme nitelikleri -** Histogramlar , Saçılım grafikleri, kutu grafikleri vs.

Veriyi incelemek ve tartışmak için pythonda **Pandas** kütüphanesini kullancağız.

Veriyi görselleştirmek içinse **Matplotlib** ve **Seaborn** kütüphanelerini kullanacağız.,

Verinin bazı örneklerini karışık olarak görebilmek için aşağıdaki kod kullanılır.

```
örnekler = veri.sample(n=5 , replace = False)
print(örnekler.iloc[:,-3:])
```

Veri görselleştirme kütüphaneleri

- Matplotlib - Matplotlib temel grafikleri çizdirmemizi ve veri hakkında bilgi edinmemizi sağlayan grafik kütüphanesidir.
- Pandas (via Matplotlib) - Pandas kütüphaneside veri görselleştirmede kullanılabilir ancak matplotlib kadar hızlı ve esnek değildir. Yinede gayet yeterlidir.
- Seaborn - Seborn Kütüphanesi fantastik grafikler çizdirmek için kullanılabilir. Matplotlibe göre daha büyüleyici temalar kullanırlar. Matplotlibden dahil edilmiş bazı içerikler barındırır. Yani seabornda temel olarak matplotlibi kullanır.

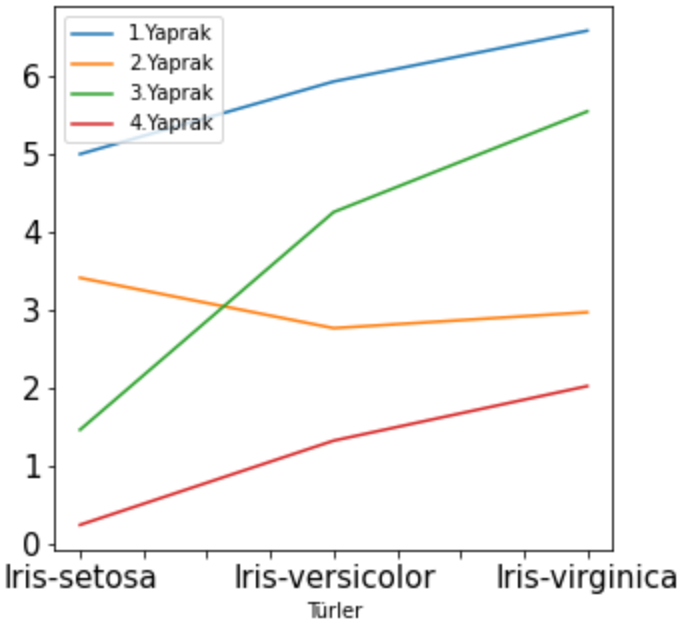
Veri Görselleştirme

1. **Gruplandırma**

   Verileri sınıfına göre gruplandırarak veriler hakkında bilgi sahibi olabilir. Burada iris veri kümseini ele alacağız.

   ```
   veri.groupby("Türler").mean().plot(fontsize=15,figsize=(5,5))
   ```

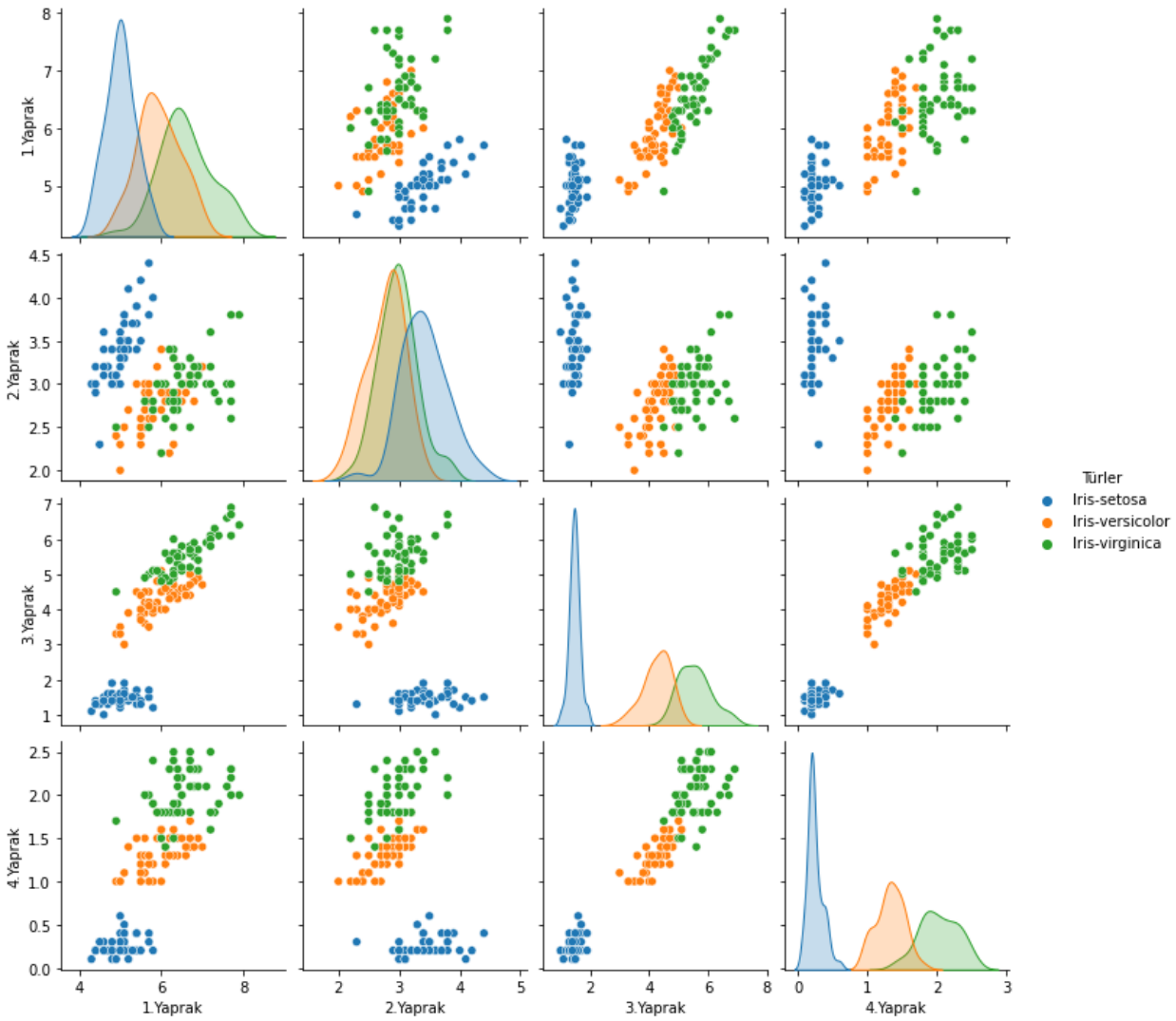
   Yukarıda ki kod karşılığında çıktımız aşağıdaki şekilde olacaktır. Bu grafik hangi değişkenin hangi durumda nasıl bir davranış sergilediğini gösterir.



2. **Çiftler Grafiği**

   Özelliklerin kendi aralarındaki bağlantıyı görmek için kullanabileceğimiz bir veri görselleştirme türüdür. Burada sınıflar esas alınarak özellikler arasındaki bağlantı gözlenir. Bunun için aşağıdaki python kodu kullanılır. Girdi olarak verinin kendisini vermek yeterlidir. renk tonu olarak sınıfların verilmesi, verinin görselleştirlmesi açısından çok daha iyi olacaktır

```
import seaborn as sns
sns.pairplot(veri,hue="Türler")
```

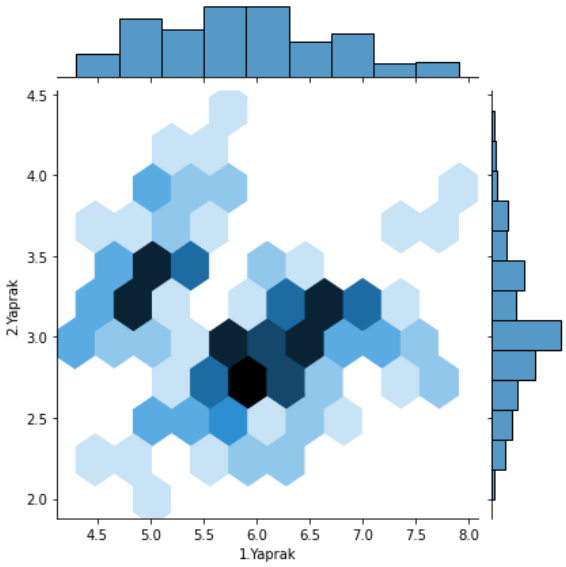


3. **Hekzagonel (Altıgen) Grafik oluşturma**

   İki veri arasındaki bağlantıyı görebilmek için kullanılabilecek grafik türlerinden biridir. kind parametresinin alacağı değere göre grafiğin şekli değişir. Örneğin kind= “kde” seçerseniz yoğunluğa göre bir izohips grafiği çıktısı alınır.

```
sns.jointplot(veri["1.Yaprak"],veri["2.Yaprak"],kind = "hex")
```

   Yukarıdaki kod yazılıdığında çıktımız aşağıdaki gibi olacaktır. Yoğunluğun yüksek olduğu yerlerde renkte koyulaşma olur.

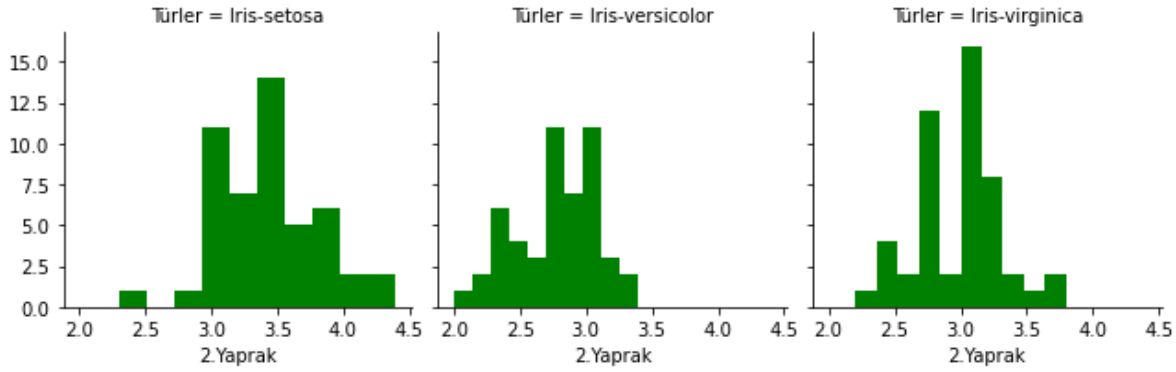


4. **Izgara methodu**

   Eğer her bir kolonun sınıf üzerindeki etkisini görmek istiyorsak bunun için Facet_grid kullanılır.

```
grafik = sns.FacetGrid(veri,col="Türler")
grafik.map(plt.hist,"2.Yaprak",color="green")
```

   Bu kodun çıktısı aşağıdaki şekilde olacaktır.

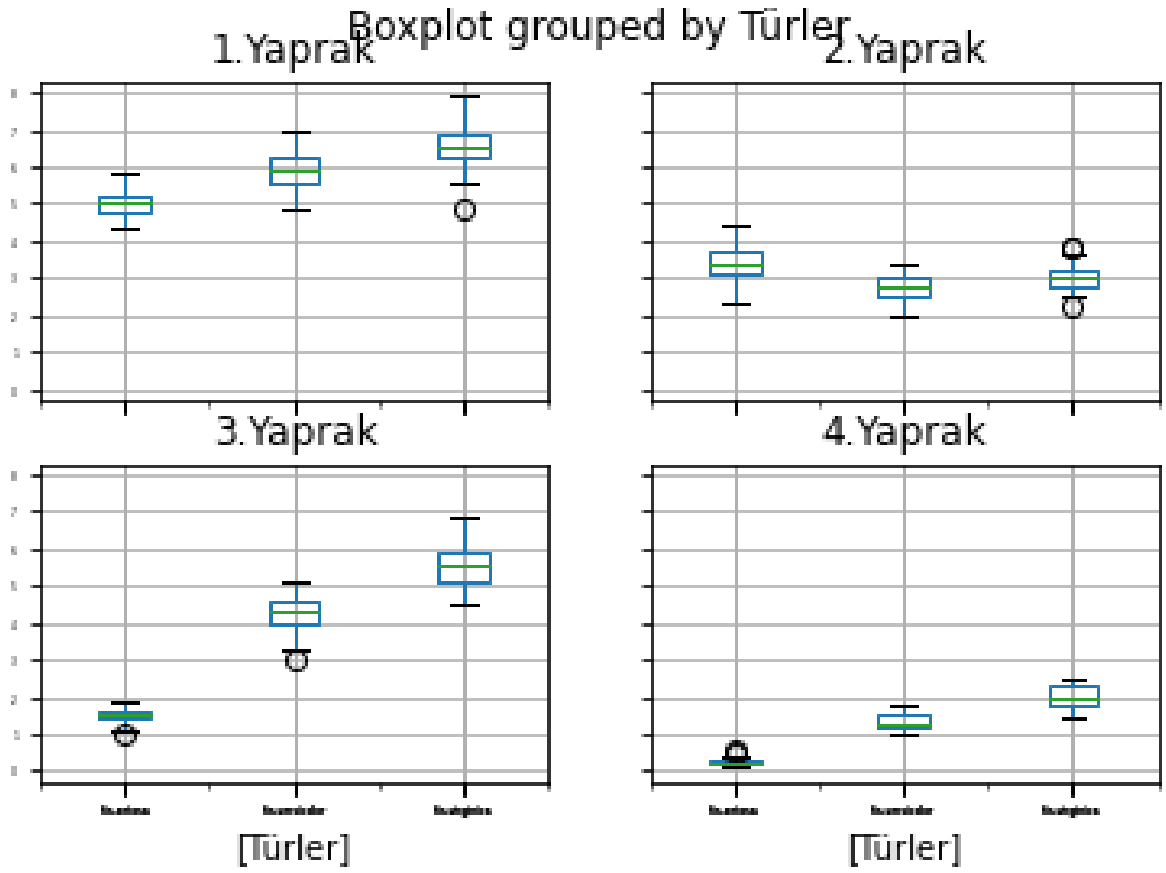


5. **Kutu Grafiği**

   Verileri istediğimiz referansa göre kutu biçiminde göstermemizi sağlayan grafik methodudur. Aşağıdaki kod ile kullanılabilir.

```
veri.boxplot(by="Türler")
```

Bu kodun çıktısı aşağıdaki şekilde olacaktır.



Değişken dönüşümleri

Mert Aydoğan Gazi University Department of ME