**Data Visualization with Python**

**In-Class Notları**

**Fonksiyonel & Object Oriented yöntem:**

Functional yöntemde doğrudan fonksiyonları kullanıyoruz. Datayı fonksiyonun argümanı olarak giriyoruz. Object Oriented yöntemde obeject'lere (nesnelere) metodlar uyguluyoruz. Aynı işlevi gören fonksiyon ve metod isimlerinde farklılıklar olabiliyor.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Şu an bir takım fonksiyonlar ile işlemler yapıyoruz. Bu fonksiyonları notebookun başında **matplotlib**modülünden import ettiğimiz **pyplot sub-modülü** üzerinden yapıyoruz.

Bu modülü **plt**adıyla import ettiğimiz için fonksiyonları yazmaya **plt.**ile başlıyoruz

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* functional'da plotu bir **fonksiyon**ile çizdiriyoruz: plt.plot(age, salary, 'r') (plot() fonksiyonu ile çizdirdik.)
* Object metodda ise öncesinde oluşturduğumuz bir **object**üzerine metod uygulayarak plot çizdiriyoruz. : ax.plot(age,salary, 'r') (ax nesnesine plot metodu uyguladık)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* ya **plt.subplots()**ile (içine gireceğimiz parametrelerle) istediğimiz ax'ları oluşturuyoruz.
* ya da **plt.figure()**fonksiyonu ile bir **figure**oluşturup sonra ona **add\_axes()**metodu ile dilediğimiz kadar ax ilave ediyoruz.

 kaç tane ayrı plotlama yapacaksak o kadar ax tanımlamalıyız.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Subplots()** fonksiyonu içindeki parametrelerde kaç adet axes tanımlandıysa figure içinde o kadar axes çizdirebiliyorsunuz.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Statistics derslerinden iki sayisal degiskenin arasinda olusacak **relationship/association**, **direction** ve **strength** konseptlerini hatirlamakta fayda var. Scatter plot bize ilk iki concept hakkinda bilgi verir

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Scatter Plot:**

**Scatter plot,** Scatterplot (dağılım grafiği veya nokta grafiği), iki değişken arasındaki ilişkiyi görselleştirmek için kullanılan bir grafik yöntemidir. Her bir veri noktası, iki değişkenin değerlerini birleştiren bir nokta olarak temsil edilir.

İki farklı sayısal değişkenin değerlerini temsil etmek için noktalar kullanır. Yatay ve dikey eksendeki her noktanın konumu, ayrı bir veri noktası için değerleri gösterir. Dağılım grafikleri, değişkenler arasındaki ilişkileri gözlemlemek için kullanılır

**Scatter plot**;

* Biri x ekseninin değerleri için, diğeri de y eksenindeki değerler için olmak üzere aynı uzunlukta iki diziyi parametre olarak alır.
* Her gözlem için bir nokta çizer.
* Bize iki değişkenin birbiri arasındaki ilişki, bu ilişkinin kuvveti ve yönü hakkında fikir verir.

**Ana kullanım yerleri şunlardır:**

**İki Değişken Arasındaki İlişkinin Gösterilmesi**: iki değişken arasındaki ilişkiyi anlamak için kullanılır. Eğer iki değişken arasında bir ilişki varsa, scatterplot bu ilişkiyi görsel olarak gösterir. İlişki pozitif yönlü ise noktaların genel bir artış eğilimi gösterdiği, ilişki negatif yönlü ise noktaların genel bir azalış eğilimi gösterdiği görülebilir. Ayrıca, ilişki lineer veya doğrusal olmayabilir, scatterplot farklı ilişki tiplerini göstermek için kullanılır

**Veri Noktalarının Dağılımının Görselleştirilmesi**: veri noktalarının dağılımını görsel olarak gösterir. Bu, veri setinde yoğunlaşma bölgelerini veya veri noktalarının yayılımını anlamak için faydalı olabilir. Yoğunlaşma bölgeleri, noktaların bir araya geldiği ve belirli bir alanda yoğunlaştığı bölgeleri temsil eder

**Outlierların Belirlenmesi**: Outlierları tespit etmek için kullanılan bir araç olabilir. Outlierlar, diğer noktalardan önemli ölçüde farklı değerlere sahip olan noktaları temsil eder. Scatterplot'ta aykırı değerler, genellikle diğer noktalardan uzakta veya belirgin şekilde farklı bir dağılım sergileyen noktalar olarak gösterilir.

**Değişkenler Arasındaki Kolerasyonun Gösterilmesi**: İki değişken arasındaki korelasyonu görsel olarak gösterir. Eğer iki değişken arasında güçlü bir korelasyon varsa, scatterplot'ta noktaların bir doğru veya eğri üzerinde yoğunlaştığı görülebilir. Bu, değişkenler arasındaki ilişkinin ne kadar güçlü olduğunu ve hangi yönde olduğunu anlamak için faydalıdır.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**colors**parametresi ile noktalar için bir renk belirleyebildiğimiz gibi, her gözlem için ayrı bir renk de verebiliyorsunuz :

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])

y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

colors = np.array(["red", "green", "blue", "yellow", "pink", "black", "orange", "purple", "beige", "brown", "gray", "cyan", "magenta"])

plt.scatter(x, y, c=colors)

plt.show()

**Dikkat**: **Tek renk vermede sorun yok. Ancak her gözlem için ayrı renk verecekseniz gözlem sayısı kadar renk girmeniz gerekiyor!**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Pandas dokümantasyonundan:**

The categorical data type is useful in the following cases:

* A string variable consisting of only a few different values. Converting such a string variable to a categorical variable will save some memory, see [here](https://pandas.pydata.org/pandas-docs/stable/categorical.html#categorical-memory).
* The lexical order of a variable is not the same as the logical order (“one”, “two”, “three”). By converting to a categorical and specifying an order on the categories, sorting and min/max will use the logical order instead of the lexical order, see [here](https://pandas.pydata.org/pandas-docs/stable/categorical.html#categorical-sort).
* As a signal to other Python libraries that this column should be treated as a categorical variable (e.g. to use suitable statistical methods or plot types).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Barplot:**

Bir **sayısal**ve **kategorik**değişken arasındaki ilişkiyi görmek için **barplot** kullanırız.

Kategorik değişkeninin her bir entity'si bir bar (çubuk) olarak temsil ediliyor. Barın boyutu ise o kategoriye ait sayısal değeri temsil ediyor.

**Ana kullanım yerleri şunlardır:**

**Kategorik verilerin karşılaştırılması**: kategorik verilerin farklı gruplar arasında karşılaştırılmasında kullanışlıdır. Örneğin, bir ürünün satış miktarlarını farklı bölgeler arasında karşılaştırmak için bar plot kullanılabilir.

**Dağılım gösterimi**: Bir değişkenin farklı kategorilere göre dağılımını görselleştirmek için kullanılabilir. Örneğin, bir e-ticaret sitesindeki ürün kategorilerinin sayısal dağılımını göstermek için bar plot kullanılabilir.

**Sıralama ve sıralı veri gösterimi**: Verilerin sıralanması gerektiği durumlarda kullanışlıdır. Örneğin; en çok satılan ürünlerin sıralamasını göstermek veya bir yıl boyunca aylara göre gelirin dağılımını göstermek için bar plot kullanılabilir.

**Toplamsal verilerin gösterimi**: Barplot, toplamsal verilerin görselleştirilmesi için etkili bir yöntemdir. Örneğin, bir yıl içindeki aylara göre bir şirketin gelirlerini göstermek için bar plot kullanılabilir.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Histogram:**

Bir veri setinin dağılımını görselleştirmek için kullanılan bir grafik yöntemidir. Veri setini belirli aralıklara böler ve her aralıkta veri noktalarının sayısını gösteren bir sütun grafiği şeklinde sunar.

Verinin olasılık dağılımını hızlıca değerlendirmek istediğimizde **histogram**kullanırız. Histogram barlarının yüksekliği bize o aralığa giren değerlerin kaç kere tekrar ettiğini gösterir. Dolayısıyla **histogram, frekans dağılımlarını gösteren bir grafiktir.**

**Ana kullanım yerleri şunlardır:**

**Veri Dağılımının Görselleştirilmesi**: Histogram, veri setinin nasıl dağıldığını hızlı bir şekilde anlamak için kullanılır. Sütunların yükseklikleri, her bir aralıktaki veri noktalarının frekansını veya yoğunluğunu temsil eder. Bu sayede veri setinin simetrik mi, çarpık mı veya normal dağılıma benzer mi olduğu gibi özellikler görsel olarak anlaşılabilir

**Central Tendency (Merkezi Eğilim) Ölçülerinin Gösterilmesi**: Histogram, veri setindeki merkezi eğilim ölçülerini görselleştirmek için kullanılabilir. Örneğin, sütunların en yüksek olduğu bölge, veri setindeki mod değerini temsil eder. Ayrıca, histogramda simetrik bir dağılım varsa, ortalama değeri daha net bir şekilde görebiliriz

**Outlierların Belirlenmesi**: Histogram, veri setindeki outlierlerı tespit etmek için kullanılan bir araç olabilir. Outlierlar, genellikle sütunların diğerlerinden önemli ölçüde yüksek veya düşük olduğu bölgede gösterilir. Bu, veri setindeki istatistiksel anormallikleri belirlemek veya potansiyel hataları tespit etmek için faydalı olabilir.

**Veri Setinin Bölgesel Dağılımının Analizi**: Histogram, veri setini belirli aralıklara böldüğü için bölgesel dağılımı gözlemlemek için kullanılabilir. Bu, özellikle verinin belirli bir değer aralığında yoğunlaşma veya seyrelme eğilimini anlamak için faydalıdır.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Histogram plot ile Bar plot farkı nedir.?**

**Barplot** : kategorik verilerin grafiksel gösterimidir.  
Her ardışık çubuk çifti arasında eşit boşluk vardır.  
Çubukların yüksekliği frekansı gösterir ve çubukların genişliği aynıdır.

**Histogram**: Nicel verilerin grafiksel gösterimidir.  
Ardışık çubuklar arasında boşluk yoktur.  
Dikdörtgen çubukların alanı verinin sıklığını gösterir ve çubukların genişliğinin aynı olması gerekmez.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**KDE Plot:**

KDE (Kernel Density Estimation) plot, veri dağılımını görselleştirmek için kullanılan bir grafik türüdür. KDE plot, veri setindeki yoğunluğu tahmin etmek ve verinin sürekli bir olasılık yoğunluk fonksiyonu olarak temsilini sağlamak için Kernel Density Estimation yöntemini kullanır.

**(KDE) grafiği:** histograma benzer şekilde bir datasetindeki gözlemlerin **dağılımını**görselleştiren bir yöntemidir.

KDE (Kernel Density Estimation) plot, veri dağılımını görselleştirmek için kullanılan bir grafik türüdür. KDE plot, veri setindeki yoğunluğu tahmin etmek ve verinin sürekli bir olasılık yoğunluk fonksiyonu olarak temsilini sağlamak için Kernel Density Estimation yöntemini kullanır.

KDE, Histogramın aksine bir veya daha fazla boyutta **sürekli olasılık yoğunluk eğrisi**kullanarak verileri temsil eder. (Histogram kesintili olarak bu temsili yapıyordu. KDE kesintisiz şekilde bunu gösteriyor)

Dolayısıyla histograma göre, KDE özellikle birden çok dağıtım çizerken daha az karmaşık ve daha kolay yorumlanabilir bir grafik sunuyor arkadaşlar.

**Ana kullanım yerleri şunlardır:**

**Veri Dağılımının Görselleştirilmesi**: KDE plot, veri setinin nasıl dağıldığını görsel olarak anlamak için kullanılır. Yoğunluğun yüksek olduğu bölgeler, verinin yoğun olduğunu veya pik değerlere sahip olduğunu gösterir. Bu sayede veri setinin çarpıklığı, çoklu modları veya keskinlikleri gibi özellikler görselleştirilebilir

**İki Veri Setinin Karşılaştırılması**: KDE plot, farklı veri setlerinin dağılımını karşılaştırmak için kullanılabilir. İki KDE plot'un üst üste bindirilmesi, veri setlerinin nasıl örtüştüğünü veya farklılık gösterdiğini görselleştirebilir. Bu, veri setleri arasındaki benzerlikleri veya farklılıkları anlamak için faydalı olabilir

**Outlierların Tespiti**: KDE plot, outlierları tespit etmek için kullanılan bir araç olabilir. Yoğunluğun düşük olduğu bölgeler, veri setindeki nadir veya outlierları gösterir. Bu, veri setinin istatistiksel anormalliklerini belirlemek veya potansiyel hataları gözlemlemek için kullanılabilir

**Veri Setinin Düzgünlüğünün Görselleştirilmesi**: KDE plot, veri setinin düzgünlüğünü ve sürekli bir dağılıma ne kadar uyduğunu gösterir. Düzgün bir KDE plot, veri setinin sürekli bir dağılımı olduğunu ve verinin tahmin edilen olasılık yoğunluk fonksiyonuna uyduğunu gösterir

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Boxplot:**

Bir datasetinin dağılımını ve merkezi eğilim ölçülerini görselleştirmek için kullanılan istatistiksel bir grafik yöntemidir. Bir boxplot, verilerin çeyrekler arası yayılımını, median değerini, outlierları ve bazen veri setinin simetrisini veya çarpıklığını gösterir.

* whisker grafiği olarak da bilinir.
* minimum, 1. quartile, median, 3. quartile ve maksimum özelliklerine sahip datanın özetini görüntülemek için oluşturulur.
* quantitative (sayısal-nicel) verilerin dağılımını, kategorik bir değişkenin level'ları arasında kolayca karşılaştırma yapabilecek şekilde gösterir.
* Birinci çeyrek ile 3. çeyrek arasındaki veriler kutu içinde gösterilir. Bu kutu içinde median değeri ayrı bir çizgi olarak gösterilir.

**Ana kullanım yerleri şunlardır:**

**Veri dağılımının görselleştirilmesi**: Box plot, veri setinin dağılımını göstermek için kullanılır. Central Tendency, skewness, outlier değerler ve çeyrekler arası yayılım gibi istatistiksel bilgileri sunar.

**Gruplar arasındaki karşılaştırma**: Box plot, farklı gruplar veya kategoriler arasındaki veri dağılımlarını karşılaştırmak için kullanışlıdır. Örneğin, farklı tedavi gruplarının sağlık sonuçlarını veya farklı şehirlerin ortalama sıcaklık dağılımlarını karşılaştırmak için box plot kullanılabilir.

**Outlierların tespiti**: Box plot, veri setindeki outlier’ları görselleştirmek için etkili bir araçtır. Bu, veri setindeki istatistiksel anormallikleri veya potansiyel hataları tespit etmek için önemli ipuçları sağlayabilir.

**Dağılımın simetri veya skewnes analizi**: Box plot, veri setinin simetri veya skewnes özelliklerini görselleştirmek için kullanılabilir. Box’ın sağ ve sol tarafındaki simetri veya sapmalar, veri setinin dağılım karakteristikleri hakkında bilgi sağlayabilir.

**Datasetinin istatistiksel özetinin sunumu**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Pie chart:**

Pie chart, verilerin oransal dağılımını hızlı ve kolay bir şekilde dairesel olarak görselleştirmek için kullanılan bir grafik türüdür. Ancak çok sayıda kategori veya çok küçük oranlara sahip verilerin görselleştirilmesinde uygun olmayabilir.

Her bir dilimin alanı, datanın o parçasının yüzdesini temsil eder.

Pie chartlar datanın hızlı bir özetini verdiği için satış, operasyon, anket sonuçları, resources (kaynaklar) gibi business sunumlarında yaygın olarak kullanılır.

**Ana kullanım yerleri şunlardır:**

**Kategori Dağılımı**: Bir Datasetindeki farklı kategorilerin oransal dağılımını göstermek için kullanılır. Örneğin, bir anket sonucunda farklı cevap seçeneklerinin yüzdelik dağılımını görselleştirmek veya bir şirketin gelirinin farklı ürün kategorilerine göre dağılımını göstermek için kullanılabilir.

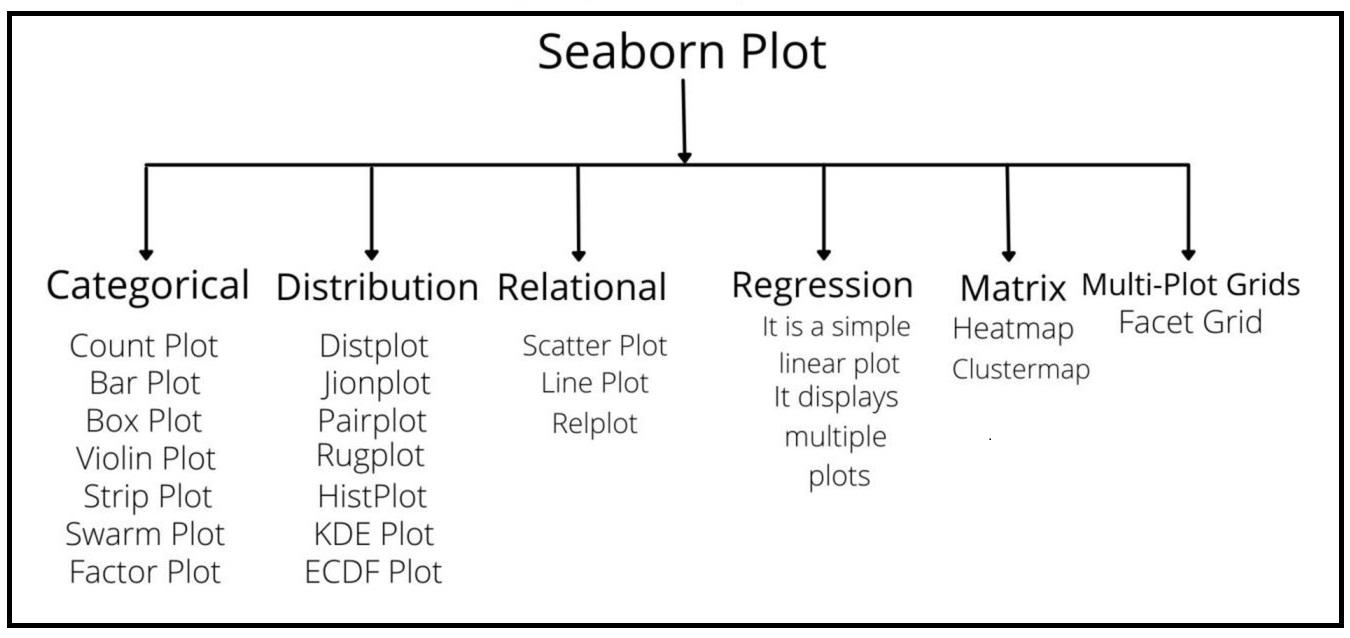
**Parçaların Oransal Karşılaştırılması**: Datasetindeki farklı parçaların bir bütün içindeki oranlarını göstermek için kullanışlıdır. Örneğin, bir pazarlama bütçesindeki harcama kalemlerinin yüzdelik olarak karşılaştırılması veya bir ülkenin enerji kaynaklarının yüzdelik dağılımının gösterilmesi için pie chart kullanılabilir.

**İlgili Verilerin Görselleştirilmesi**: Bir ana kategorinin alt kategorilere göre dağılımını göstermek için de kullanılır. Örneğin, bir satış raporunda bir ürün kategorisinin farklı markalar arasındaki pazar payını görselleştirmek veya bir şirketin departmanları arasındaki bütçe dağılımını göstermek için pie chart kullanılabilir.

**Yüzdelik Payın Vurgulanması**: Datasetindeki bir yüzdelik payın önemli olduğu durumlarda kullanışlıdır. Örneğin, bir anket sonucunda belirli bir seçeneğin diğerlerine göre yüksek bir yüzdeyle öne çıkması veya bir şirketin karının farklı gelir kaynaklarına göre oransal olarak vurgulanması için pie chart kullanılabilir.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**SEABORN**



**Plot çeşitleri:**

* **Distribution Plots**
* **Categorical Plots**
* **Comparison Plots**
* **Matrix Plots**

İstatistikte iki tür veri vardır: Nümerik ve Kategorik veriler.

**Nümerik veri örnekleri**: Yaş, kilo, bir şeyin sayısı, ayakkabı numarası, toplam fatura.

**Kategorik veri örnekleri**: Göz rengi, cinsiyet, haftanın günleri, kan grubu, hava durumu, haftanın günleri.

* Nümerik verilerin analizinde şu **distribution** plotları kullanıyoruz:
* **Histogram**
* **Rugplot**
* **Kdeplot**
* Kategorik verilerin analizinde şu **categorical** plot’ları kullanıyoruz:
* **Barplot**
* **Countplot**
* **Boxplot**
* **Violinplot**
* **Swarmplot**
* **Boxenplot**
* Karşılıklı iki feature’ı iki boyutlu olarak plotlamak istediğimizde şu **comparision** plot’ları kullanıyoruz:
* **Joinplot**
* **Pairplot**

**Pairplot**, hem tek bir feature’ın dağılımını hem de iki feature arasındaki ilişkileri görmenizi sağlar.

Pairplot, varsayılan olarak yalnızca sayısal sütunları çizer, ancak daha sonra hue parametresiyle kategorik değişkenleri renklendirmek suretiyle kullanabilirsiniz.

Datasetini pairplot'un içine atıyorsun ve o tüm featureların birbiri ile grafiklerini çiziyor. Bunu parça parça da yapabilirsin. ilk 5'ini sonraki 5'ini gibi.

Genellikle bir EDA'ya veya bir görselleştirmeye başlarken tüm datasetinin birbirleri ile görsel ilişkisini yakalamak için bir pairplot kullanırız. patern yakalamaya çalışırız. daha sonra orda bir ilişki gördünüz tip ile total\_bill arasında. orada detay çalışmak için diğer plotları kullanırsın.

* Verileri renk kodlu matrisler olarak çizebilmek için Matris grafikleri (**heatmap, clustermap**) kullanırız.

En sık kullanılanı **heatmap’tir**.

**Gridplots**:

Otomatik olarak birden çok subplot oluşturabilmek için grid’leri kullanıyoruz. Bazen orta boyutlu datanın analizinde aynı plotun birden fazla subplotunu çizmek faydalı bir yaklaşımdır. Böylece karmaşık verilerden hızlı bir şekilde insight elde edebiliriz.

Bunun için catplot, pairplot, PairGrid ve FacetGrid kullanırız.

**Histogram** grafiğini iki şekilde çizdirebiliyoruz, ya displot ile ya histplot ile..

Displot, matplotlib'deki histograma karşılık gelir!

Displot ve histplot, rugplottaki verilerin miktarlarını aralıklar halinde gösterir.

**Rugplotta** her bir veri için bir çizgi vardır. Displot ve histplot ise bu çizgilerin sayısını gösteriyor.

**Displot**default olarak **histogram**çizer. Ama **sadece histogram çizmez, üzerinde başka plotlar da çizer.**

Displot kendisi bir canvastır yani müstakil bir figure'u var. Onu bir axes’in üzerine çizdiremezsiniz. Dolayısıyla bir ax tanınlamanıza gerek yoktur. figsizegibi matplotlib'in object oriented yöntemlerini displotta kullanmıyoruz! Bunun yerine height veaspect parametrelerinikullanıyoruz. Bu parametreler ile grafik boyutunu ve en-boy oranını belirleyebiliyoruz.

**KDE: kernel density estimation.**Grafiğinin altındaki alan 1’e eşittir. Verilerin görülme yüzdesi hakkında dolayısıyla **yoğunlaştığı yerler ve dağılım**hakkında fikir verir.

**Jointplot**: joint plot **iki tane numeric değer ister**.

Scatterplot ile histogramın birleşimi gibi bir şeydir.

(kind= “scatter”) kind attribute’unun default’u scatter’dır ama bunu değiştirebilirsin.

**Violinplot = kdeplot + boxplot**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Categorical scatterplots**:

* [**stripplot()**](https://seaborn.pydata.org/generated/seaborn.stripplot.html#seaborn.stripplot) (with kind="strip"; the default)
* [**swarmplot()**](https://seaborn.pydata.org/generated/seaborn.swarmplot.html#seaborn.swarmplot) (with kind="swarm")

**Categorical distribution plots:** (Distributions of observations within categories)

* [**boxplot()**](https://seaborn.pydata.org/generated/seaborn.boxplot.html#seaborn.boxplot) (with kind="box")
* [**violinplot()**](https://seaborn.pydata.org/generated/seaborn.violinplot.html#seaborn.violinplot) (with kind="violin")
* [**boxenplot()**](https://seaborn.pydata.org/generated/seaborn.boxenplot.html#seaborn.boxenplot) (with kind="boxen")

**Categorical estimate plots:** (Statistical estimation within categories)

* [**pointplot()**](https://seaborn.pydata.org/generated/seaborn.pointplot.html#seaborn.pointplot) (with kind="point")
* [**barplot()**](https://seaborn.pydata.org/generated/seaborn.barplot.html#seaborn.barplot) (with kind="bar")
* [**countplot()**](https://seaborn.pydata.org/generated/seaborn.countplot.html#seaborn.countplot) (with kind="count")

**Categorical Plots:**

**swarmplot:** grafikte yer alan değerlerin **benzer veya ilişkili olduğunu gösterir**. Verilerin **dağılımını ve gruplar arasındaki farklılıkları görselleştirmenin etkili bir yoludur.** Ancak, **büyük veri setleri için okunması zor**olabilir. Ayrıca, **verilerin dağılımı hakkında ayrıntılı bilgi sağlayamazlar.**

**countplot**: kategoriye göre *kayıt sayısını* (observation sayısı) çizer.

**barplot**: her kategori için bir *değer veya metrik* çizer. Default olarak **mean**değerini çizer. (feature'ın her bir kategorisinin ortalama değerlerini gösterir)

**Countplot**, kategorik bir değişkenin dağılımını bize verir ve genellikle **her kategorinin sıklığını görüntülemek için** kullanılır.

**Barplot**, sayısal bir değişkenin ortalama değerini farklı kategoriler arasında karşılaştırmak için kullanılır ve genellikle**iki değişken arasındaki ilişkiyi görselleştirmek için** kullanılır.

**Güven aralığı (Ci)**değerini azalttıkça çubuk küçülecek, arttırdıkça çubuk büyüyecektir. Çünkü bir ortalamayı daha büyük bir doğrulukla tutturabilmeniz için tahminlerinizin daha büyük bir aralıkta olması gerekir.Başka bir deyişle; verinin ortalama değeri şu ile şu değerler arasındadır dediğinde bu ifadenin doğruluğunun daha yüksek olması (hata payının daha küçük olması) için o aralık değerini büyük tutmak gerekir. Aralık azalttıkça doğruluk oranı da tabi ki azalacaktır.

**Box plot**, median’ı (50. percentile) merkez çizgisi olarak gösterir, ardından box sınırları olarak 25. ve 75. Persentile’ı kullanır. Ardından, oulier sınırlarını hesaplamak için IQR yöntemi kullanır. (üst sınır için 1.5 \* IQR + Q3, alt sınır için Q1 - 1.5 \* IQR ). Outlierlar, whisker’ların dışında farklı noktalar olarak gösterilir.

**Boxenplot,** dağılımı farklı bir şekilde gösterir ve daha **büyük veri setleri için daha iyidir.** Klasik boxplot çok fazla aykırı değer içerebilir ve dağılım hakkında daha az bilgi sunar. Boxenplotlar, median değerini (Q2, 50. yüzdelik) merkezi çizgi olarak başlar. **Dışarı doğru her bir seviye, geriye kalan verinin yarısını içerir.**Merkezi çizgiden başlayarak ilk iki bölüm, verinin %50'sini içerir. Bundan sonra, bir sonraki iki bölüm, verinin %25'ini içerir. Bu, aykırı değer seviyesine gelene kadar devam eder.

**Violin plot**: Genel olarak violinplot, sayısal verilerin görselleştirilmesi için kullanılan bir yöntemidir ve bir boxplot ile kernel density plot’un birleşimi olarak düşünülebilir**. Violinplotta boxplotta bulunan aynı biilgileri bulabiliriz.**

* median (a white dot on the violin plot)
* interquartile range (the black bar in the center of violin)
* the lower/upper adjacent values (the black lines stretched from the bar) — defined as first quartile — 1.5 IQR and third quartile + 1.5 IQR respectively. These values can be used in a simple outlier detection technique (Tukey’s fences) — observations lying outside of these “fences” can be considered outliers.

 Özetle : **Violinplot = kdeplot + boxplot**

