# **Exploratory analysis of KPIs**

## 1. [1.] Exploratory analysis of KPIs [KPI'ların keşifsel analizi]

Great work on the exercises! [Egzersizler üzerinde harika bir çalışma!] Let us pick up where we left off. [Bıraktığımız yerden devam edelim.]

## 2. [2.] KPIs [kpi]

We want to determine which conversion rate metric is the most appropriate. [Hangi dönüşüm oranı metriğinin en uygun olduğunu belirlemek istiyoruz.] Note that most companies will have many KPIs each serving a different purpose, and that here we are only working through one of these cases. [Çoğu şirketin, her birinin farklı bir amaca hizmet eden birçok KPI'si olacağını ve burada bu vakalardan yalnızca biri üzerinde çalıştığımızı unutmayın.]

## 3. [3.] Methods for calculating KPIs [KPI'ları hesaplama yöntemleri]

To calculate our potential KPIs and measure performance across different groups we will use the `groupby()` and `agg()` pandas methods. [Potansiyel KPI'larımızı hesaplamak ve farklı gruplar arasındaki performansı ölçmek için 'groupby()' ve 'agg()' panda yöntemlerini kullanacağız.] This lesson will focus on these methods and the next lesson will more fully explore applying them in practice. [Bu ders bu yöntemlere odaklanacak ve bir sonraki ders bunları pratikte uygulamayı daha kapsamlı bir şekilde keşfedecek.]

## 4. [4.] Grouping Data: .groupby() [Verileri Gruplandırma: .groupby()]

We can call the `groupby()` method on a dataFrame to specify groups to aggregate over. [Üzerinde toplanacak grupları belirtmek için dataFrame üzerinde `groupby()` yöntemini çağırabiliriz.] Here we will use it on our combined demographics and purchase data dataset. [Burada, birleşik demografimizde kullanacağız ve veri veri setini satın alacağız.] The primary argument is `by` to which we provide a list of dataFrame fields that we want to group on. [Birincil argüman, üzerinde gruplandırmak istediğimiz dataFrame alanlarının bir listesini sağladığımız 'by'dir.] Here, the potentially relevant fields are "country", "device", "gender", and "age". [Burada potansiyel olarak ilgili alanlar 'ülke', 'cihaz', 'cinsiyet' ve 'yaş'tır.] Let us group by "country" and "device". ['Ülke' ve 'cihaz'a göre gruplandıralım.] The next relevant argument is `axis` which specifies whether we are grouping by row or column values. [Bir sonraki ilgili argüman, satır değerlerine mi yoksa sütun değerlerine göre mi gruplandırdığımızı belirten 'eksen'dir.] The default value, "0", groups by columns, which is what we will do here, and for the remainder of the course. [Varsayılan değer olan '0', sütunlara göre gruplar, burada yapacağımız şey ve kursun geri kalanı için.] The other argument of interest is `as\_index`. [İlgilenilen diğer argüman 'as\_index'tir.] By default, this argument is “True”, which means that the grouped by fields become indices. [Varsayılan olarak, bu bağımsız değişken 'Doğru'dur; bu, alanlara göre gruplananların dizin haline geldiği anlamına gelir.] We want to set this to "False" so that this does not happen. [Bunun olmaması için bunu 'Yanlış' olarak ayarlamak istiyoruz.] This returns a dataFrameGroupedBy object. [Bu, bir dataFrameGroupedBy nesnesi döndürür.] The next step is to aggregate over these groups. [Bir sonraki adım, bu gruplar üzerinde bir araya gelmektir.]

## 5. [5.] Aggregating data - mean price paid per group [Toplama verileri - grup başına ödenen ortalama fiyat]

The easiest way to do this is to call an aggregation method on the dataFrameGroupedBy object. [Bunu yapmanın en kolay yolu, dataFrameGroupedBy nesnesi üzerinde bir toplama yöntemi çağırmaktır.] Let’s call `mean` on the `price` value of our dataFrame. [DataFrame'imizin 'fiyat' değerine 'mean' diyelim.] The output is the mean amount paid per subscription across all purchasing users. [Çıktı, satın alan tüm kullanıcılar genelinde abonelik başına ödenen ortalama tutardır.] In this case rather than being calculated over the entire set of data, it is calculated over each-device country combination. [Bu durumda, tüm veri seti üzerinden hesaplanmak yerine, her bir cihaz ülke kombinasyonu üzerinden hesaplanır.]

## 6. [6.] Aggregate data: .agg() [Toplu veriler: .agg()]

Any built in function similar to mean can be called on a dataFrameGroupedBy object. [Ortalamaya benzer herhangi bir yerleşik işlev, bir dataFrameGroupedBy nesnesinde çağrılabilir.] However more flexible options exist through the dot agg() method The easiest way to use this method is to pass a function like mean to it. [Ancak dot agg() yöntemi aracılığıyla daha esnek seçenekler mevcuttur. Bu yöntemi kullanmanın en kolay yolu, ona ortalama gibi bir işlev iletmektir.] As we can see, this has the same result as when we called mean directly. [Gördüğümüz gibi, bu, doğrudan ortalama dediğimiz zamankiyle aynı sonuca sahiptir.]

## 7. .agg(): multiple functions [7. .agg(): çoklu işlevler]

It can be further expanded by passing in a list of functions, like mean and median, and calculating both. [Ortalama ve medyan gibi bir işlev listesinden geçerek ve her ikisini de hesaplayarak daha da genişletilebilir.]

## 8. .agg(): multiple functions, multiple columns [8. .agg(): birden çok işlev, birden çok sütun]

The true flexibility comes from a third type of argument. [Gerçek esneklik, üçüncü tip bir argümandan gelir.] We can pass in a dictionary where the keys are column names within our dataset, like "purchase" or "age” and the values are a list of functions to be applied over those columns; still broken out by groups. [Anahtarların 'satın alma' veya 'yaş' gibi veri kümemizdeki sütun adları olduğu ve değerlerin bu sütunlar üzerine uygulanacak işlevlerin bir listesi olduğu bir sözlükten geçebiliriz; yine de gruplara ayrılmış halde.] Let us find the mean, minimum, and maximum value of each of purchase and age, as an example. [Örnek olarak her bir satın alma ve yaşın ortalama, minimum ve maksimum değerini bulalım.]

## 9. .agg(): custom functions [9. .agg(): özel işlevler]

Another great flexibility of the agg method is that we can also pass our own functions in to aggregate over, not only built in ones. [agg yönteminin bir başka büyük esnekliği de, yalnızca yerleşik olanları değil, aynı zamanda kendi işlevlerimizi de iletebilmemizdir.] Here is a function that finds the truncated mean value, that is it removes the top and bottom ten-percent of values before calculating the average. [Burada, kesilen ortalama değeri bulan, yani ortalamayı hesaplamadan önce değerlerin üst ve alt yüzde onunu kaldıran bir fonksiyon bulunmaktadır.] We can aggregate our age over the country and device groupings with this function. [Bu fonksiyon ile ülke ve cihaz gruplamaları üzerinden yaşımızı toplayabiliriz.] The only distinction is that when inputting this function, we do not want its name in quotations as we did for the built in functions. [Tek fark, bu işlevi girerken, yerleşik işlevler için yaptığımız gibi adını tırnak içinde istemiyoruz.]

## 10. [10.] Let's practice! [Hadi pratik yapalım!]

It is important to cover the techniques before proceeding to applying them. [Uygulamaya geçmeden önce teknikleri kapsamak önemlidir.] In the next video we will look at how to use these to examine KPIs across cohorts and discuss why this is valuable. [Bir sonraki videoda, kohortlar arasında KPI'ları incelemek için bunların nasıl kullanılacağına bakacağız ve bunun neden değerli olduğunu tartışacağız.] Let’s practice these tools before moving onto that! [Buna geçmeden önce bu araçları uygulayalım!]