

R İLE ENERJİ PİYASASI VERİ ANALİZİ EL KİTABI

Son Güncelleme: 29.01.2020

Contents

1 Giriş	1
1.1 Amaç	1
1.2 Kurulum	2
1.3 Dökümanda Kullanılan Veri Seti	3
2 dplyr ile Veri Manipülasyonu	5
2.1 Hazırlıklar	5
2.2 Önemli Fonksiyonlar	6
2.3 Örnekler	14
3 ggplot2 ile Veri Görselleştirme	16
3.1 Hazırlıklar	17
3.2 Nokta Grafiği (Scatter Plot)	17
3.3 Sütunlu Grafik (Bar Chart)	20
3.4 Çizgi Grafiği (Line Chart)	25
3.5 Isı Haritası (Heat Map)	27
Ekler	30
Yazarlar Hakkında	30
RStudio Cloud'da Çalışma	30
Veri Seti Düzenlemeleri	33
Kaynakça	33

1 Giriş

1.1 Amaç

Bu dökümanın temel amacı, R yazılım dilinin yüklenmesi ve kurulmasından başlayarak çeşitli veri manipülasyonlarının nasıl gerçekleştirilebileceğini ve bu verilerin nasıl kolaylıkla ve oldukça şık bir biçimde görselleştirilebileceğini enerji piyasası verileriyle ve yine enerji piyasasına uyumlu örneklerle uygulamaya dökmektir. Bu dökümanın, enerji sektöründe ve benzeri diğer sektörlerde rol alan bireyler için oldukça yararlı olduğuna inanıyor, şimdiden kolaylıklar diliyoruz.

1.2 Kurulum

1.2.1 R ve RStudio'nun İndirilmesi

Öncelikle ücretsiz bir istatistiksel programlama dili olan R'nin yüklenmesi gerekiyor. R'ı indirmek için <https://cran.rstudio.com/> sitesine ilerleyiniz ve işletim sisteminize uygun olan versiyonuna tıklayarak indirmeyi başlatınız.

Daha sonra güçlü bir kullanıcı arayüzü olan RStudio'nun indirilmesi gerekiyor. Onun için de <https://rstudio.com/products/rstudio/download/> linkini kullanabilirsiniz. İndirme tamamlandıktan sonra direktifler takip edilerek RStudio kurulumu da tamamlanmış olacaktır.

Eğer bütün bu indirmeleri yapmak istemezseniz ise, yine bir RStudio ürünü olan tamamen ücretsiz ve herhangi bir indirmeye ihtiyaç duymayan çevrimiçi platform RStudio Cloud uygulamasını ziyaret edebilirsiniz. Bu işlemlerin nasıl yapılacağına anlatıldığı RStudio Cloud'da Çalışma bölümüne göz atabilirsiniz.

1.2.2 Kurulumun Test Edilmesi

Kurulumlar tamamlandıktan sonra sisteminizin çalıştığından emin olmak adına RStudio'yu açın ve Konsol veya "Console" yazan yere tıklayarak basit bir kod yazın. Örneğin `x = 3 + 4` yazın ve ENTER'a tıklayın. Burada `x` nesnesine bir toplama işleminin sonucu atanmış oldu. Bu nesnenin **Environment** (Ortam) penceresi altında 7 değeriyle belirdiğine dikkat ediniz.

Herhangi bir hata almadığınızdan emin olduğunuzda bir sonraki bölüme geçebilirsiniz.

1.2.3 İlk R Script Dosyasının Oluşturulması

Yazılan kodları ileride inceleyebilmek, üzerinde değişiklikler yapabilmek için R Script olarak adlandırılan belgeler üzerinde çalışmanız daha mantıklı olacaktır. Dikkat edeceğimiz üzere yukarıda konsola yazdığımız `x <- 3 + 4` komutu üzerinde bir değişiklik yapamayacaksınız. Yeni bir R Script belgesi yaratmak için ise RStudio'nun üstünde göreceğiniz panelden "File", "New File" ve "R Script" sırasıyla seçin. İlk R belgenizin açıldığını göreceksiniz.

R Script'lerde her bir satır kodu ayrı ayrı çalıştırmanız gerekiyor. Bunun için de üzerinde olduğunuz satırı çalıştırmak için Windows kullanıcısı iseniz CTRL+ENTER, MacOS kullanıyorsanız CMD+ENTER kombinasyonlarını kullanmalısınız.

1.2.4 R'da Paketler Hakkında

R paketleri fonksiyonlar, hazır kodlar veya veri setleri içerebilen farklı amaçlar göz önünde bulundurularak oluşturulmuş paketlerdir. R programlama dilini ilk indirdiğinizde birçok paket beraberinde indirilmiş şekildedir ve eğer kullanmak istediğiniz paket bu paketler arasında değilse onu da indirmek ve yüklemek oldukça basittir.

1.2.5 Gerekli Paketlerin İndirilip Yüklenmesi

Bu bölümde kitabın ilerleyen aşamalarında kullanılacak paketlerin indirilmesi ve yüklenmesi tamamlanacaktır.

- Veri manipülasyonu için `dplyr`,
- Veri görselleştirmesi için `ggplot2`,

- Tarih - zaman verilerinin manipölasyonları için ise `lubridate`,
- Daha ileri seviye veri manipölasyonları için `tidyr`,

paketleri kullanılacaktır. Paketleri indirmek için aşğıdaki kodu ilk R Script belgenizde çalıştırabilirsiniz. (İndirme ile alakalı kodları bir kere çalıştırmanız yetecektir.)

R'da her bir satırı ayrı ayrı çalıştırmanız gerektiğine dikkat ediniz. (Üzerinde olduğunuz satırı çalıştırmak için Windows kullanıcısı iseniz **CTRL+ENTER**, MacOS kullanıyorsanız **CMD+ENTER** kombinasyonlarını kullanmalısınız.) İndirmeler internet bağlantınızın durumuna göre 1-5 dakika arası sürebilir.

```
install.packages("dplyr")
install.packages("ggplot2")
install.packages("lubridate")
install.packages("tidyr")
```

İndirilen paketlerin yüklenmesi için ise kullanılması gereken kod aşğıda bulunabilir. (Bu kodları ise programı her açtığınızda tekrar uygulamanız gerekmektedir.)

```
library(dplyr)
library(ggplot2)
library(lubridate)
library(tidyr)
```

Yüklemeler de tamamlandığında bu kitapta kullanılan veri setinin nasıl indirileceğini anlatan kısma geçmeye hazırsınız.

1.3 Dökümanda Kullanılan Veri Seti

1.3.1 Veri Setine İlk Bakış

Nasıl indirileceği ve üzerinde ne gibi işlemler uygulanabileceği daha sonraki bölümlerde anlatılacak olan veri setinin öncelikle ne gibi veriler içerdiğine ve bunların özellikle piyasaya hakim olmayanlar için kısa bir özetine ihtiyaç vardır.

Bu dökümanda EPİAŞ Raporlama Sayfası'nda paylaşılan Piyasa Takas Fiyatı (PTF) - Sistem Marjinal Fiyatı (SMF) verisi kullanılmıştır. Ham veride bu iki veriye ek olarak Pozitif Dengesizlik Fiyatı ve Negatif Dengesizlik Fiyatı verileri de yer almaktadır. Bu ifadelerin ne olduğundan kısaca bahsetmek gerekirse,

- *Piyasa Takas Fiyatı (PTF)*: Gün öncesi piyasasında (GÖP) verilmiş olan teklifler sonucu arz ve talebin kesiştiği noktada meydana gelen fiyat olarak adlandırılabilir.
- *Sistem Marjinal Fiyatı (SMF)*: Sistemin enerji açığı ya da fazlası gösterdiği durumlarda Dengeleme Güç Piyasası'nda YAL ve YAT talimatlarına göre net talimat hacmine tekabül eden teklif fiyatı olarak adlandırılabilir.
- *Pozitif Dengesizlik Fiyatı (PDF)*: Sistemde enerji fazlası olduğunda enerjinin normalden daha ucuza çıkarılmasını sağlayan fiyat değeri olarak düşünülebilir. Gelecekte buna benzer dengesizliklerin oluşmasına engel olma amaçlı bir nevi ceza olarak düşünülebilecek Pozitif Dengesizlik Fiyatı aşğıdaki gibi hesaplanır.

$$PDF(TL/MWH) = 0.97 * \min(PTF, SMF)$$

- *Negatif Dengesizlik Fiyatı (NDF)*: Sistemde enerji açığı olduğunda enerjinin normalden daha pahalıya alınmasını sağlayan fiyat değeri olarak düşünülebilir. Gelecekte buna benzer dengesizliklerin oluşmasına engel olma amaçlı bir nevi ceza olarak düşünülebilecek Negatif Dengesizlik Fiyatı aşağıdaki gibi hesaplanır.

$$NDF(TL/MWH) = 1.03 * maks(PTF, SMF)$$

Daha detaylı bilgi için EPIAŞ sitesini ziyaret edebilirsiniz.

1.3.2 Veri Setinin İndirilmesi

Bu dökümanda kullanılan veri seti EPIAŞ'ın Raporlama Sayfasından elde edilen 01.01.2019 - 31.12.2019 tarihli Piyasa Takas Fiyatı (PTF) - Sistem Marjinal Fiyatı (SMF) verisidir.

Veri setinin ham haline EPIAŞ'ın Raporlama Sayfası 'ndan ya da buraya tıklayarak ulaşabilirsiniz. Bu veri setini direkt olarak RStudio içerisinde indirmek de ayrıca mümkün. Bu işlemi ise aşağıdaki kutuda yer alan adımları izleyerek gerçekleştirebilirsiniz.

- Excel dosyasını okumayı sağlayacak `readxl` paketinin indirilmesi.

```
install.packages("readxl")
```

- `readxl` paketinin yüklenmesi.

```
library(readxl)
```

- Ham verinin okunması ve `ptfsmf` objesine atanması.

```
download.file("https://github.com/acikenerji/verianalizi101/blob/master/ptf-smf.xls?raw=true",destfile=
ptfsmf <- read_excel("ptf-smf.xls")
```

Burada `ptfsmf` objesi R'in pek çok obje tipinden biri olan "Data Frame" tipindedir. Data Frame'ler şekilsel anlamda matrislere benzese de birden çok veri tipini depolayabilmeleri açısından oldukça yararlıdır. Örneğin `ptfsmf` objesinin "Tarih" sütunu bir tarih tipi olan `POSIXct` tipinde, geri kalan sütunlar ise `numeric` tipindedir. Bu tiplere hakim olmanız bu döküman için gerekli değildir. Temel olarak bilinmesi gereken nokta, `ptfsmf` objesinin elde ettiğimiz tabloyu depolayan obje olduğudur.

Bu veri setinin ham halinin örneklerde rahatlıkla kullanılabilmesi adına bazı format ve isim değişiklikleri yapılmıştır ve bu yazının kapsamını aştığından bu dönüşümlerin nasıl yapıldığı anlatılmayacaktır. İlgilenenler için ise bu kodlar Veri Seti Düzenlemeleri kısmında bulunabilir.

Veri setinin örneklerle uygun şekilde hazırlanmış haline ise aşağıdaki kodu R'da çalıştırarak ulaşabilirsiniz.

```
ptfsmf <- readRDS(url("https://github.com/acikenerji/verianalizi101/blob/master/duzenlenmis_ptfsmf.rds?raw=true"))
```

Bu aşamada veri seti okunduktan sonra geri kalan bütün hazırlıklar her bölümde ayrı ayrı detaylı bir şekilde anlatılacaktır.

Örneklerle ilgilenmeye başlamadan önce çalışma dizininizin bilgisayarınızın hangi bölgesi olduğunu öğrenmek için `getwd()` komutunu konsola yazarak çalıştırabilirsiniz. Bir başka önemli nokta ise indirilen bir belge, R Script belgesi içerisinde kullanılacaksa bu belgenin `getwd()` komutunun vereceği sonuçla aynı çalışma dizininde olması gerekmektedir.

2 dplyr ile Veri Manipülasyonu

Bu bölümde hedeflenen şey, çok önemli ve oldukça güçlü bir data manipülasyonu paketi olan **dplyr** paketinin önemli fonksiyonlarına dair örnekler vererek nasıl kullanılabileceklerini göstermektir. Bu bölümde ele alınacak fonksiyonlar şu şekildedir:

```
select/rename  
filter  
distinct  
arrange  
mutate/transmute  
group_by/summarise
```

Yukarıda bahsedilen bütün fonksiyonlar ayrı ayrı ele alınacak ve her biri için örnek kullanımlar gösterilecektir. Ayrıca “pipe operator” olarak adlandırılan “Bağlantı Operatörü” (`%>%`) de kısaca anlatılacak ve bütün döküman boyunca kullanılacaktır.

2.1 Hazırlıklar

Tablonun ilk haline `glimpse()` fonksiyonu ile göz atılabilir.

```
ptfsmf %>% glimpse()
```

```
## Observations: 8,760  
## Variables: 5  
## $ Tarih <dtm> 2019-01-01 00:00:00, 2019-01-01 01:00:00, 2019-01-01 02:00:0...  
## $ PTF <dbl> 100.38, 96.72, 81.60, 38.58, 11.52, 11.14, 11.14, 24.37, 34.5...  
## $ SMF <dbl> 5.00, 95.04, 79.60, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00,...  
## $ PDF <dbl> 4.85, 92.19, 77.21, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00,...  
## $ NDF <dbl> 103.39, 99.62, 84.05, 39.74, 11.87, 11.47, 11.47, 25.10, 35.5...
```

Burada “Bağlantı Operatörü”nün (`%>%`) kullanımına dikkat edilmelidir. Eğer bu “Bağlantı Operatörü” kullanılmıyorsa aynı işlem şu şekilde gerçekleştirilecekti,

```
glimpse(ptfsmf)
```

```
## Observations: 8,760  
## Variables: 5  
## $ Tarih <dtm> 2019-01-01 00:00:00, 2019-01-01 01:00:00, 2019-01-01 02:00:0...  
## $ PTF <dbl> 100.38, 96.72, 81.60, 38.58, 11.52, 11.14, 11.14, 24.37, 34.5...  
## $ SMF <dbl> 5.00, 95.04, 79.60, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00,...  
## $ PDF <dbl> 4.85, 92.19, 77.21, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00,...  
## $ NDF <dbl> 103.39, 99.62, 84.05, 39.74, 11.87, 11.47, 11.47, 25.10, 35.5...
```

Yukarıdaki örnekte tek bir fonksiyon kullanıldığından daha kolay görünmeyebilir ancak şu örnek daha belirleyici olacaktır. (Bu örnekte kullanılan `select` fonksiyonu aşağıda daha detaylı anlatılacaktır, burada sadece örnek olması açısından kullanılmıştır.)

1. Bağlantı Operatörü kullanılarak PTF ve SMF sütunlarının seçilmesi ve gösterilmesi,

```
ptfsmf %>%
  select(PTF, SMF) %>%
  glimpse()
```

```
## Observations: 8,760
## Variables: 2
## $ PTF <dbl> 100.38, 96.72, 81.60, 38.58, 11.52, 11.14, 11.14, 24.37, 34.50,...
## $ SMF <dbl> 5.00, 95.04, 79.60, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0...
```

2. Bağlantı Operatörü kullanılmadan PTF ve SMF sütunlarının seçilmesi ve gösterilmesi,

```
glimpse(select(ptfsmf, PTF, SMF))
```

```
## Observations: 8,760
## Variables: 2
## $ PTF <dbl> 100.38, 96.72, 81.60, 38.58, 11.52, 11.14, 11.14, 24.37, 34.50,...
## $ SMF <dbl> 5.00, 95.04, 79.60, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0...
```

Görebileceğiniz üzere bu operatör sayesinde farklı fonksiyonları iç içe yazmak yerine alt alta daha tertipli bir şekilde yazabiliriz.

Aşağıdaki örnekleri incelerken aklınıza “Uygulanan bu işlemler neden kalıcı olmadı, ptfsmf objesi neden değişmedi?” gibi sorular gelebilir. Bu R programlama dilinin bir özelliğinden kaynaklanmaktadır. `<-` veya `=` atama operatörlerinden biri kullanılmadığı sürece yazılan işlemler gerçekleştirilecek ve ekrana yansıtılacak ancak `ptfsmf` objesinde bir değişiklik olmayacaktır.

Şimdi `dplyr` paketinin önemli fonksiyonları ile devam edebiliriz.

2.2 Önemli Fonksiyonlar

2.2.1 select/rename

`select` fonksiyonu belirli sütunları seçmek için kullanılır. Örneğin tablonun sadece Tarih ve Piyasa Takas Fiyatı’nı (PTF) göstermesi isteniyorsa,

```
ptfsmf %>% select(Tarih, PTF)
```

```
## # A tibble: 8,760 x 2
##   Tarih          PTF
##   <dtm>         <dbl>
## 1 2019-01-01 00:00:00 100.
## 2 2019-01-01 01:00:00 96.7
## 3 2019-01-01 02:00:00 81.6
## 4 2019-01-01 03:00:00 38.6
## 5 2019-01-01 04:00:00 11.5
## 6 2019-01-01 05:00:00 11.1
## 7 2019-01-01 06:00:00 11.1
## 8 2019-01-01 07:00:00 24.4
## 9 2019-01-01 08:00:00 34.5
## 10 2019-01-01 09:00:00 45.2
## # ... with 8,750 more rows
```

Veya (bu örnek için pek faydalı gibi görünmese de) içinde “P” harfini barındıran sütunlar seçilmek istendiğinde `contains` kelimesi kullanılabilir.

```
ptfsmf %>% select(contains("P"))
```

```
## # A tibble: 8,760 x 2
##       PTF    PDF
##   <dbl> <dbl>
## 1 100.    4.85
## 2  96.7  92.2
## 3  81.6  77.2
## 4  38.6   0
## 5  11.5   0
## 6  11.1   0
## 7  11.1   0
## 8  24.4   0
## 9  34.5   0
## 10 45.2   0
## # ... with 8,750 more rows
```

Bu kelime gibi `select` fonksiyonu içinde kullanılabilecek diğer kelimeler: `starts_with`, `ends_with`, `matches` olarak sıralanabilir.

Bu kelimeler ve daha fazlasının anlatıldığı dplyr Cheatsheet dökümanına göz atılabilir.

Aynı zamanda, eğer sütunların sırası biliniyorsa iki sütun arasındaki her sütunu seçmek için `:` operatörü kullanılabilir. Örneğin Sistem Marjinal Fiyatı (SMF) ve Negatif Dengesizlik Fiyatı (NDF) arasındaki sütunları seçmek için,

```
ptfsmf %>% select(SMF:NDF)
```

```
## # A tibble: 8,760 x 3
##       SMF    PDF    NDF
##   <dbl> <dbl> <dbl>
## 1     5    4.85  103.
## 2  95.0  92.2   99.6
## 3  79.6  77.2   84.0
## 4     0     0    39.7
## 5     0     0    11.9
## 6     0     0    11.5
## 7     0     0    11.5
## 8     0     0    25.1
## 9     0     0    35.5
## 10    0     0    46.6
## # ... with 8,750 more rows
```

`rename` fonksiyonu ise adından da anlaşılacağı üzere bir sütunun adını değiştirmek için kullanılır. Örneğin “PTF” sütununun ismini “ptf” olarak değiştirmek ve sonrasında bu sütunu seçmek istiyorsanız `rename` ve `select` fonksiyonlarını kullanabilirsiniz.

```
ptfsmf %>% rename(ptf = PTF) %>%
  select(ptf)
```

```
## # A tibble: 8,760 x 1
##   ptf
##   <dbl>
## 1 100.
## 2  96.7
## 3  81.6
## 4  38.6
## 5  11.5
## 6  11.1
## 7  11.1
## 8  24.4
## 9  34.5
## 10 45.2
## # ... with 8,750 more rows
```

2.2.2 filter

`filter` fonksiyonu temel olarak istenilen koşulları sağlayan satırları seçmek için kullanılır. Örneğin Piyasa Takas Fiyatı'nın 250'den düşük olduğu satırlar incelenebilir.

```
ptfsmf %>% filter(PTF < 250)
```

```
## # A tibble: 2,453 x 5
##   Tarih          PTF   SMF   PDF   NDF
##   <dtm>         <dbl> <dbl> <dbl> <dbl>
## 1 2019-01-01 00:00:00 100.    5    4.85 103.
## 2 2019-01-01 01:00:00  96.7  95.0  92.2  99.6
## 3 2019-01-01 02:00:00  81.6  79.6  77.2  84.0
## 4 2019-01-01 03:00:00  38.6    0    0    39.7
## 5 2019-01-01 04:00:00  11.5    0    0    11.9
## 6 2019-01-01 05:00:00  11.1    0    0    11.5
## 7 2019-01-01 06:00:00  11.1    0    0    11.5
## 8 2019-01-01 07:00:00  24.4    0    0    25.1
## 9 2019-01-01 08:00:00  34.5    0    0    35.5
## 10 2019-01-01 09:00:00  45.2    0    0    46.6
## # ... with 2,443 more rows
```

Gözlem sayısının 8670'ten 2453'e düştüğü görülebiliyor.

Aynı şekilde bir değerin büyüklüğü $>$, küçüklük veya eşitliği $<=$, büyüklük veya eşitliği $>=$, eşitliği $==$ sembolleriyle incelenebilir.

Birden fazla koşulun sağlanması gerekiyorsa ise, “VE” operatörü için $\&$, “VEYA” operatörü için $|$ sembolleri kullanılmalıdır. Örneğin Pozitif Dengesizlik Fiyatı'nın 200'den küçük, Negatif Dengesizlik Fiyatı'nın ise 200'den büyük olduğu satırlar incelenebilir.


```
ptfsmf %>% filter(PDF<200 & NDF>200)
```

```
## # A tibble: 1,256 x 5
##   Tarih          PTF   SMF   PDF   NDF
##   <dtm>         <dbl> <dbl> <dbl> <dbl>
## 1 2019-01-01 17:00:00 287.  172  167.  296.
## 2 2019-01-01 18:00:00 292.  172  167.  300.
## 3 2019-01-01 21:00:00 285.  173  168.  294.
## 4 2019-01-01 22:00:00 205.  173.  168.  211.
## 5 2019-01-02 07:00:00 288.  169.  164.  296.
## 6 2019-01-02 09:00:00 297.  173  168.  306.
## 7 2019-01-02 10:00:00 297.  190  184.  306.
## 8 2019-01-02 19:00:00 293.  190  184.  302.
## 9 2019-01-02 20:00:00 290.  173.  168.  299.
## 10 2019-01-02 21:00:00 289.  173.  168.  297.
## # ... with 1,246 more rows
```

Bu koşulları sağlayan 1256 satır olduğu görülüyor. Eğer bir koşulu sağlamayan satırlar aranıyorsa ise koşulun başına ! yazılmasıyla bu ters etki taratılabilir. Örneğin Piyasa Takas Fiyatı'nın (PTF) 0'dan farklı olduğu satırları bulmak için,

```
ptfsmf %>% filter(!(PTF==0))
```

```
## # A tibble: 8,708 x 5
##   Tarih          PTF   SMF   PDF   NDF
##   <dtm>         <dbl> <dbl> <dbl> <dbl>
## 1 2019-01-01 00:00:00 100.    5  4.85  103.
## 2 2019-01-01 01:00:00 96.7  95.0 92.2  99.6
## 3 2019-01-01 02:00:00 81.6  79.6 77.2  84.0
## 4 2019-01-01 03:00:00 38.6    0    0   39.7
## 5 2019-01-01 04:00:00 11.5    0    0   11.9
## 6 2019-01-01 05:00:00 11.1    0    0   11.5
## 7 2019-01-01 06:00:00 11.1    0    0   11.5
## 8 2019-01-01 07:00:00 24.4    0    0   25.1
## 9 2019-01-01 08:00:00 34.5    0    0   35.5
## 10 2019-01-01 09:00:00 45.2    0    0   46.6
## # ... with 8,698 more rows
```

Burada bu koşulu sağlayan 8706 satır olduğu görülüyor.

2.2.3 arrange

arrange fonksiyonu Excel'deki sıralama özelliğine benzetilebilir. Varsayılan durumunda A'dan Z'ye ya da küçükten büyüğe sıralanır. Tam tersi sıralama için desc() fonksiyonu kullanılmalıdır. Örneğin artan Piyasa Takas Fiyatlarına göre sıralamak için,

```
ptfsmf %>% arrange(PTF)
```

```
## # A tibble: 8,760 x 5
##   Tarih          PTF   SMF   PDF   NDF
```

```
##      <dtm>                <dbl> <dbl> <dbl> <dbl>
## 1 2019-02-17 09:00:00      0      30      0 30.9
## 2 2019-03-24 12:00:00      0       5      0 5.15
## 3 2019-03-24 13:00:00      0       0      0 0
## 4 2019-03-24 14:00:00      0       0      0 0
## 5 2019-03-24 15:00:00      0       0      0 0
## 6 2019-03-24 16:00:00      0      10      0 10.3
## 7 2019-04-14 09:00:00      0       0      0 0
## 8 2019-04-14 10:00:00      0       0      0 0
## 9 2019-04-14 13:00:00      0       0      0 0
## 10 2019-04-14 14:00:00      0       0      0 0
## # ... with 8,750 more rows
```

Bir başka örnekte Tarih, Pozitif Dengesizlik Fiyatı (PDF) ve Negatif Dengesizlik Farkı (NDF) sütunları seçilip azalan NDF'ye göre sıralanabilir.

```
ptfsmf %>% select(Tarih, PDF, NDF) %>%
  arrange(desc(NDF))
```

```
## # A tibble: 8,760 x 3
##   Tarih                PDF    NDF
##   <dtm>                <dbl> <dbl>
## 1 2019-06-27 15:00:00  436.  515
## 2 2019-06-27 16:00:00  436.  515.
## 3 2019-06-27 14:00:00  454.  502.
## 4 2019-11-11 18:00:00  385.  502.
## 5 2019-06-26 11:00:00  303.  462.
## 6 2019-06-26 14:00:00  307.  450.
## 7 2019-11-11 17:00:00  370.  439.
## 8 2019-11-27 18:00:00  340.  433.
## 9 2019-07-09 15:00:00  373.  433.
## 10 2019-07-09 16:00:00  373.  433.
## # ... with 8,750 more rows
```

Dikkat edileceği üzere NDF değerleri azalarak devam ediyor. Ayrıca `arrange` fonksiyonunun içine birden fazla değer girerek ilk değerler eşitliği durumunda ikinci değerin karar vermesi sağlanabilir. Örneğin PTF değerinin 0 olduğu son gün aşağıdaki gibi bulunabilir.

```
ptfsmf %>% select(Tarih, PTF) %>%
  arrange(PTF, desc(Tarih))
```

```
## # A tibble: 8,760 x 2
##   Tarih                PTF
##   <dtm>                <dbl>
## 1 2019-06-06 09:00:00      0
## 2 2019-06-05 09:00:00      0
## 3 2019-06-04 16:00:00      0
## 4 2019-06-04 15:00:00      0
## 5 2019-06-04 14:00:00      0
## 6 2019-06-04 13:00:00      0
## 7 2019-06-04 12:00:00      0
## 8 2019-06-04 11:00:00      0
```

```
## 9 2019-06-04 10:00:00 0
## 10 2019-06-04 09:00:00 0
## # ... with 8,750 more rows
```

Bu durumda çıkan tabloya göz atıldığında çıkan ilk veri yıl içerisinde PTF'nin 0 olduğu son günü verecektir.

2.2.4 mutate/transmute

`mutate` fonksiyonu genellikle var olan değişkenlerle yapılan operasyonlar sonucu yeni değişkenler (sütunlar) yaratmak için kullanılır.

Örneğin yeni bir gün sütunu eklenmek istenirse (burada `lubridate` paketinin `as_date()` fonksiyonu kullanılmıştır.),

```
ptfsmf %>% mutate(Gün = as_date(Tarih))
```

```
## # A tibble: 8,760 x 6
##   Tarih          PTF   SMF   PDF   NDF   Gün
##   <dtm>         <dbl> <dbl> <dbl> <dbl> <date>
## 1 2019-01-01 00:00:00 100.    5   4.85 103. 2019-01-01
## 2 2019-01-01 01:00:00 96.7  95.0 92.2  99.6 2019-01-01
## 3 2019-01-01 02:00:00 81.6  79.6 77.2  84.0 2019-01-01
## 4 2019-01-01 03:00:00 38.6    0    0   39.7 2019-01-01
## 5 2019-01-01 04:00:00 11.5    0    0   11.9 2019-01-01
## 6 2019-01-01 05:00:00 11.1    0    0   11.5 2019-01-01
## 7 2019-01-01 06:00:00 11.1    0    0   11.5 2019-01-01
## 8 2019-01-01 07:00:00 24.4    0    0   25.1 2019-01-01
## 9 2019-01-01 08:00:00 34.5    0    0   35.5 2019-01-01
## 10 2019-01-01 09:00:00 45.2    0    0   46.6 2019-01-01
## # ... with 8,750 more rows
```

Veya Veri Setine İlk Bakış bölümünde anlatıldığı üzere Pozitif Dengesizlik Fiyatı (PDF) hesaplanmak istenirse,

```
ptfsmf %>% mutate(PDF_Yeni = ifelse(PTF<SMF, 0.97*PTF, 0.97*SMF))
```

```
## # A tibble: 8,760 x 6
##   Tarih          PTF   SMF   PDF   NDF PDF_Yeni
##   <dtm>         <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 2019-01-01 00:00:00 100.    5   4.85 103.    4.85
## 2 2019-01-01 01:00:00 96.7  95.0 92.2  99.6   92.2
## 3 2019-01-01 02:00:00 81.6  79.6 77.2  84.0   77.2
## 4 2019-01-01 03:00:00 38.6    0    0   39.7    0
## 5 2019-01-01 04:00:00 11.5    0    0   11.9    0
## 6 2019-01-01 05:00:00 11.1    0    0   11.5    0
## 7 2019-01-01 06:00:00 11.1    0    0   11.5    0
## 8 2019-01-01 07:00:00 24.4    0    0   25.1    0
## 9 2019-01-01 08:00:00 34.5    0    0   35.5    0
## 10 2019-01-01 09:00:00 45.2    0    0   46.6    0
## # ... with 8,750 more rows
```

Dikkat edileceği üzere “PDF_Yeni” ve “PDF” sütunları aynı değerleri gösteriyor. Burada temel R fonksiyonlarının `mutate` içerisindeki kullanımına dikkat edilmelidir.

Burada Veri Setine İlk Bakış kısmında anlatıldığı üzere Pozitif Dengesizlik Fiyatı (PDF) hesaplamak için `ifelse` fonksiyonu kullanılmıştır. Burada temel olarak yapılan işlem şudur: $PTF < SMF$ koşulu doğruysa ilk denklemi ($0.97 * PTF$) uygula, değilse ikinci denklemi ($0.97 * SMF$) uygula.

`transmute` fonksiyonu da `mutate` fonksiyonu ile benzer bir işleve sahip olmasının yanında `select` fonksiyonunun belirli sütunları seçme işlevine de sahiptir. Örneğin `lubridate` paketinin `as_date()` fonksiyonunu kullanarak yeni bir “Gün” sütunu açıp “PTF” ve “SMF” sütunlarını da seçmek ve azalan PTF’ye göre sıralamak için,

```
ptfsmf %>%
  transmute(PTF, SMF, Gün = as_date(Tarih)) %>%
  arrange(desc(PTF))
```

```
## # A tibble: 8,760 x 3
##   PTF   SMF   Gün
##   <dbl> <dbl> <date>
## 1  500   450 2019-06-27
## 2  500   450 2019-06-27
## 3  488   468 2019-06-27
## 4  401   401 2019-06-27
## 5  400   400 2019-06-12
## 6  400   400 2019-06-12
## 7  400   400 2019-06-12
## 8  397   487 2019-11-11
## 9  385   415 2019-07-09
## 10 385   420 2019-07-09
## # ... with 8,750 more rows
```

2.2.5 group_by/summarise

Bu iki fonksiyon çoğunlukla özetleme tabloları çıkarmak için kullanılır. `group_by` sütunlara göre veriyi gruplamak, `summarise` ise gruplanan bu verilere göre istenen özeti çıkarmakla görevlidir. Örneğin günlük ortalama PTF fiyat bilgisi için,

```
ptfsmf %>%
  mutate(Gün = as_date(Tarih)) %>%
  group_by(Gün) %>%
  summarise(Gunluk_Ort_PTF = mean(PTF))
```

```
## # A tibble: 365 x 2
##   Gün           Gunluk_Ort_PTF
##   <date>           <dbl>
## 1 2019-01-01         121.
## 2 2019-01-02         229.
## 3 2019-01-03         239.
## 4 2019-01-04         212.
## 5 2019-01-05         244.
## 6 2019-01-06         237.
## 7 2019-01-07         247.
## 8 2019-01-08         248.
## 9 2019-01-09         220.
## 10 2019-01-10        163.
## # ... with 355 more rows
```

Burada kullanılan `mean` fonksiyonu dışında maksimum değer için `max`, minimum değer için `min`, medyan için `median`, total değer için `sum` ve grupladığımız değişkene ait gözlem sayısını saymak için ise `n()` fonksiyonları kullanılabilir.

2.2.6 distinct

`distinct` fonksiyonu bir veya birden fazla sütunu baz alarak özgün satırları bulmak için kullanılır. Örneğin var olan tabloda bir “Gün” sütunu var ve her güne dair yalnızca bir değer incelenmek isteniyor. Bu durumdan kurtulmak için `distinct` fonksiyonu kullanılabilir. Örneğin,

```
ptfsmf %>%
  mutate(Gün = as_date(Tarih)) %>%
  distinct(Gün)
```

```
## # A tibble: 365 x 1
##   Gün
##   <date>
## 1 2019-01-01
## 2 2019-01-02
## 3 2019-01-03
## 4 2019-01-04
## 5 2019-01-05
## 6 2019-01-06
## 7 2019-01-07
## 8 2019-01-08
## 9 2019-01-09
## 10 2019-01-10
## # ... with 355 more rows
```

Dikkat edileceği üzere yalnızca `Gün` sütunu korundu. Diğer sütunları da korumak için `.keep_all = TRUE` komutu kullanılmalıdır.

```
ptfsmf %>%
  mutate(Gün = as_date(Tarih)) %>%
  distinct(Gün, .keep_all = TRUE)
```

```
## # A tibble: 365 x 6
##   Tarih          PTF    SMF    PDF    NDF  Gün
##   <dtm>         <dbl> <dbl> <dbl> <dbl> <date>
## 1 2019-01-01 00:00:00 100.    5    4.85 103. 2019-01-01
## 2 2019-01-02 00:00:00 105.   103.  99.9 108. 2019-01-02
## 3 2019-01-03 00:00:00 126.    52   50.4 130. 2019-01-03
## 4 2019-01-04 00:00:00 104.    50   48.5 107. 2019-01-04
## 5 2019-01-05 00:00:00 178.   100    97   184. 2019-01-05
## 6 2019-01-06 00:00:00 286.   271   263.  294. 2019-01-06
## 7 2019-01-07 00:00:00 200.   100.   97.1  206. 2019-01-07
## 8 2019-01-08 00:00:00 199.   120   116.  205. 2019-01-08
## 9 2019-01-09 00:00:00 199.   98.8   95.8  205. 2019-01-09
## 10 2019-01-10 00:00:00  4.42    0.1    0.1   4.55 2019-01-10
## # ... with 355 more rows
```

Burada gözlem sayısının 365’e düştüğüne dikkat edilmelidir.

2.3 Örnekler

Bu bölümde, kolaydan zora örnekler ele alınacak ve dplyr paketi daha da pekiştirilmeye çalışılacaktır. Soruların cevapları hemen soruların altında olacaktır ancak cevaplara göz atmadan önce soruları çözmeye çalışmanızı şiddetle tavsiye ediyoruz.

Devamlı bir gelişim gösteren kitabımıza yeni örnekler eklemeyi sürdüreceğiz.

2.3.1 Basit Seviye

1. Piyasa Takas Fiyatı (PTF) verisinin aylık ortalama, medyan, maksimum ve minimum değerlerini bulunuz.

İpucu: lubridate paketinin month() fonksiyonunu kullanabilirsiniz.

Sonuç olarak şu şekilde bir tablo ortaya çıkmalı,

```
## # A tibble: 12 x 5
##   Ay   ORT   MED  MAKS  MIN
##   <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     1  223.  258.  327.  0.5
## 2     2  257.  275.  330.  0
## 3     3  256.  296.  332.  0
## 4     4  187.  200.  309.  0
## 5     5  196.  220.  317.  0
## 6     6  221.  275.  500  0
## 7     7  305.  315.  385  50.0
## 8     8  298.  310.  330.  0.99
## 9     9  298.  307.  325.  45
## 10    10  292.  309.  338  10
## 11    11  298.  313.  397.  50
## 12    12  291.  314.  338.  45.0
```

Cevap ise şu şekilde,

```
ptfsmf %>%
  mutate(Ay=month(Tarih)) %>%
  group_by(Ay) %>%
  summarise(ORT = mean(PTF), MED = median(PTF), MAKS = max(PTF), MIN = min(PTF))
```

2. Yaz aylarının (Haziran, Temmuz, Ağustos ayları) akşam saatlerinde (18-22 arası, bu değerler dahil olmak üzere) ortaya çıkan günlük toplam NDF değerlerini bulunuz.

İpucu: Saatleri bulmak için lubridate paketinin hour() fonksiyonunu, tarihi güne çevirmek için ise yine aynı paketten as_date() fonksiyonunu kullanabilirsiniz.

Sonuç olarak şu şekilde bir tablo ortaya çıkmalı,

```
## # A tibble: 92 x 2
##   Tarih      NDF
##   <date>    <dbl>
## 1 2019-06-01 1957
## 2 2019-06-02 1849.
## 3 2019-06-03 1516.
```

```
## 4 2019-06-04 726.
## 5 2019-06-05 1103.
## 6 2019-06-06 1496.
## 7 2019-06-07 1646.
## 8 2019-06-08 1513.
## 9 2019-06-09 1117.
## 10 2019-06-10 1558.
## # ... with 82 more rows
```

Cevap ise şu şekilde,

```
ptfsmf %>%
  mutate(Saat=hour(Tarih), Tarih=as_date(Tarih)) %>%
  filter((Tarih >= "2019-06-01" & Tarih <= "2019-08-31"), (Saat >= 18 & Saat <= 22)) %>%
  select(Tarih, Saat, NDF) %>%
  group_by(Tarih) %>%
  summarise(NDF = sum(NDF))
```

2.3.2 İleri Seviye

1. SMF/PTF oranının aylara göre en yüksek olduğu üçer gözlemi bulunuz.

İpucu: İstedığınız sayıda satırı seçmek için `slice()` fonksiyonunu kullanabilirsiniz. `slice()` fonksiyonunun kullanımı için dplyr Cheatsheet dökümanına göz atılabilir.

Sonuç olarak şu şekilde bir tablo ortaya çıkmalı,

```
## # A tibble: 36 x 3
##   Tarih          Ay SMF_PTF_Orani
##   <dtm>         <dbl>      <dbl>
## 1 2019-01-13 16:00:00 1         2.03
## 2 2019-01-28 04:00:00 1         1.71
## 3 2019-01-13 11:00:00 1         1.53
## 4 2019-02-24 13:00:00 2        51.3
## 5 2019-02-10 09:00:00 2        28.4
## 6 2019-02-24 15:00:00 2        18.6
## 7 2019-03-24 17:00:00 3        15.5
## 8 2019-03-29 23:00:00 3        15.3
## 9 2019-03-24 18:00:00 3         8.25
## 10 2019-04-21 12:00:00 4        100
## # ... with 26 more rows
```

Cevap ise şu şekilde,

```
ptfsmf %>%
  filter(!(PTF == 0)) %>%
  mutate(Ay = month(Tarih)) %>%
  mutate(SMF_PTF_Orani = SMF / PTF) %>%
  group_by(Ay) %>%
  arrange(desc(SMF_PTF_Orani)) %>%
  slice(1:3) %>%
  ungroup() %>%
  select(Tarih, Ay, SMF_PTF_Orani)
```

Burada önemli olan bir nokta, `group_by()` fonksiyonu ile gruplanmış bir tablo ile farklı işlemlere devam edilmek isteniyorsa hata alınmaması amacıyla `ungroup()` fonksiyonunun kullanılması gerektirir.

2. Her bir günün ortalama PTF değerleri ile birlikte yeni bir sütunda tam bir hafta önceki ortalama PTF değerini gösteriniz.

İpucu: Haftanın günlerini çıkarmak için R'nin kendi fonksiyonu olan `weekdays()` fonksiyonunu, tabloda kaydırma yapmak için ise `dplyr` paketinin `lag()` fonksiyonu kullanılabilir.

Sonuç olarak şu şekilde bir tablo ortaya çıkmalı (İlk 7 günün geçmiş verisi olmadığından o satırlar `slice()` fonksiyonu yardımıyla silinmiştir.),

```
## # A tibble: 716 x 4
##   Tarih      Gün      PTF_Degisim    PTF_Degerleri
##   <date>    <chr>    <chr>          <dbl>
## 1 2019-01-08 Tuesday  ÖncekiHaftaPTF    121.
## 2 2019-01-08 Tuesday   PTF                248.
## 3 2019-01-09 Wednesday ÖncekiHaftaPTF    229.
## 4 2019-01-09 Wednesday   PTF                220.
## 5 2019-01-10 Thursday  ÖncekiHaftaPTF    239.
## 6 2019-01-10 Thursday   PTF                163.
## 7 2019-01-11 Friday    ÖncekiHaftaPTF    212.
## 8 2019-01-11 Friday     PTF                180.
## 9 2019-01-12 Saturday  ÖncekiHaftaPTF    244.
##10 2019-01-12 Saturday   PTF                237.
## # ... with 706 more rows
```

Cevap ise şu şekilde,

```
ptfsmf %>%
  mutate(Tarih = as_date(Tarih)) %>%
  group_by(Tarih) %>%
  summarise(PTF = mean(PTF)) %>%
  ungroup() %>%
  mutate(Gün = weekdays(Tarih)) %>%
  mutate(ÖncekiHaftaPTF = lag(x = PTF, n = 7)) %>%
  select(Tarih, Gün, ÖncekiHaftaPTF, PTF) %>%
  slice(8:n()) %>%
  pivot_longer(cols=c(ÖncekiHaftaPTF, PTF), names_to="PTF_Degisim",
               values_to="PTF_Degerleri") %>%
  arrange(Tarih)
```

Burada `tidyr` paketinin `pivot_longer()` fonksiyonu bulunuyor. Bu fonksiyon temel olarak geniş tabloları belirli sütunlarını bir ortak sütun içerisinde birleştirip bunu yaparken de satır sayısını birleştirilen sütun sayısı oranında artıran bir fonksiyondur.

3 ggplot2 ile Veri Görselleştirme

Bu bölümde `ggplot2` paketinden yararlanarak veri görselleştirmenin nasıl yapılabileceği farklı fonksiyonlar ve örnekler üzerinden gösterilecektir.

3.1 Hazırlıklar

`dplyr` paketinde birden fazla fonksiyonu aynı veri setine uygulamak için “Bağlantı Operatörü” (`%>%`) kullanılırken `ggplot2` paketinde ise birden fazla etkiyi aynı anda uygulamak için `(+)` operatörü kullanılıyor. Bu durum başlangıç grafiğinin üzerine farklı görseller eklenmesi olarak düşünülebilir.

`ggplot()` fonksiyonu ile de ilk kez karşılaşıldığından bazı özelliklerinden bahsedilmelidir. Fonksiyonun içine ilk yazılan parametre kullanılacak veri setinin ismidir. Daha sonra `aes()` yardımıyla grafiğin `x` ve `y` koordinatları belirlenir. `aes()` içerisinde ayrıca `color`, `fill`, `alpha`, `shape`, `size` gibi özellikler de belirlenebilir. Bu özelliklerden işimize yarayacak olanlar kullanıldıkları yerlerde açıklanacaktır.

`ggplot()` fonksiyonu tanımlandıktan sonra istenilen grafik şekline göre farklı fonksiyonlar `(+)` operatörü ile üstüne eklenir. Örneğin,

- Nokta grafiği için `geom_point()` fonksiyonu,
- Sütunlu grafik için `geom_bar()` fonksiyonu,
- Çizgi grafiği için `geom_line()` fonksiyonu,
- İsi haritası ya da grafiği için ise `geom_tile()` fonksiyonları eklenir.

3.2 Nokta Grafiği (Scatter Plot)

İlk olarak bahsedilecek grafik türü olan “Scatter Plot” temel olarak var olan verinin noktasal dağılımını göstermek için kullanılır.

“Scatter Plot” için kullanılan fonksiyon `geom_point()` tir. `x` ekseninde “PTF”, `y` ekseninde ise “SMF” değerleri renkleri günlere göre değişecek şekilde gösterilirse (Öncelikle gün sütunu ekleniyor ve yeni bir tabloya ataması yapılıyor.),

```
plot_df <- ptfsmf %>%  
  mutate(Gün = as_date(Tarih))
```

Tabloya göz atalım,

```
plot_df
```

```
## # A tibble: 8,760 x 6  
##   Tarih          PTF   SMF   PDF   NDF   Gün  
##   <dtm>         <dbl> <dbl> <dbl> <dbl> <date>  
## 1 2019-01-01 00:00:00 100.    5    4.85 103. 2019-01-01  
## 2 2019-01-01 01:00:00 96.7   95.0 92.2   99.6 2019-01-01  
## 3 2019-01-01 02:00:00 81.6   79.6 77.2   84.0 2019-01-01  
## 4 2019-01-01 03:00:00 38.6    0    0    39.7 2019-01-01  
## 5 2019-01-01 04:00:00 11.5    0    0    11.9 2019-01-01  
## 6 2019-01-01 05:00:00 11.1    0    0    11.5 2019-01-01  
## 7 2019-01-01 06:00:00 11.1    0    0    11.5 2019-01-01  
## 8 2019-01-01 07:00:00 24.4    0    0    25.1 2019-01-01  
## 9 2019-01-01 08:00:00 34.5    0    0    35.5 2019-01-01  
## 10 2019-01-01 09:00:00 45.2    0    0    46.6 2019-01-01  
## # ... with 8,750 more rows
```

Şimdi grafiği çizdirecek olursak,

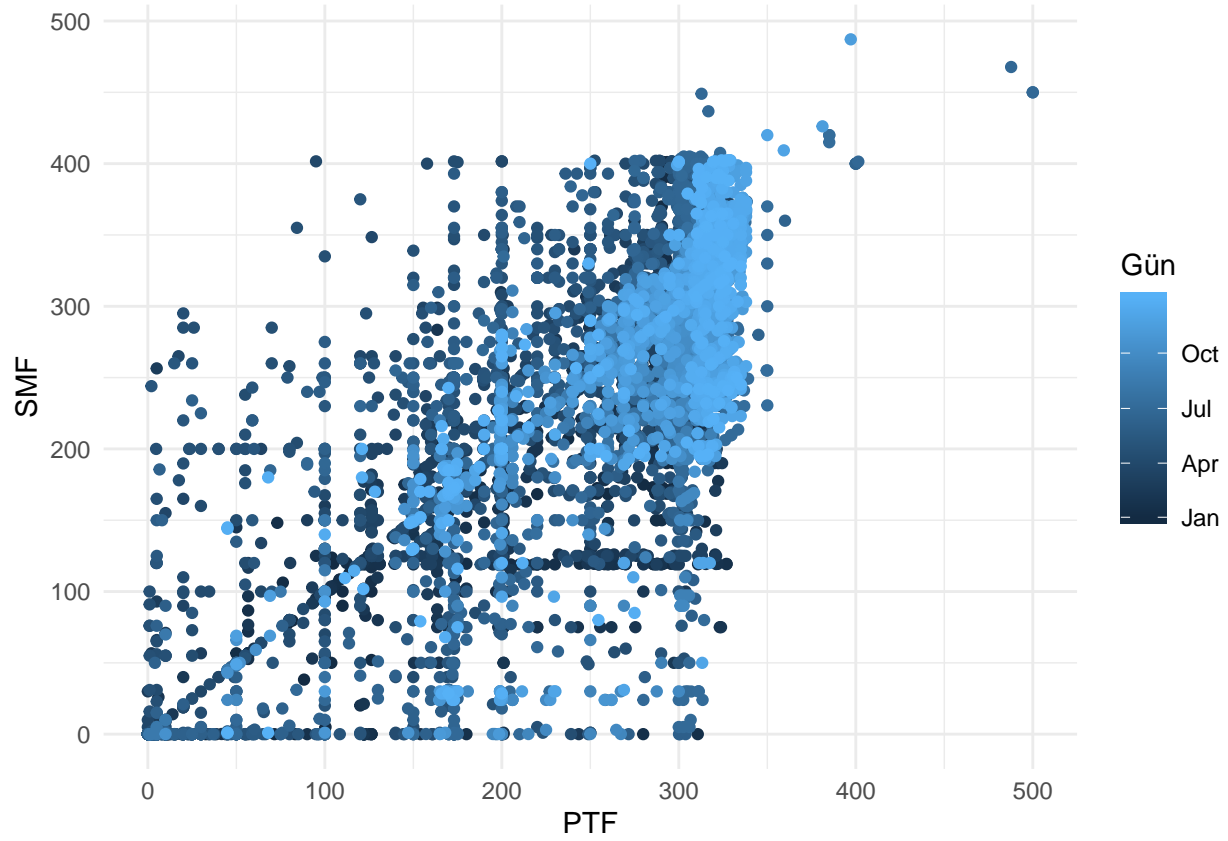
```
ggplot(plot_df, aes(x=PTF, y=SMF, color=Gün)) +  
  geom_point()
```



Görülebileceği üzere öncelikle veri seti, sonrasında ise `aes()` içerisinde `x` ve `y` koordinatları ile renk `color` yardımıyla belirlendi.

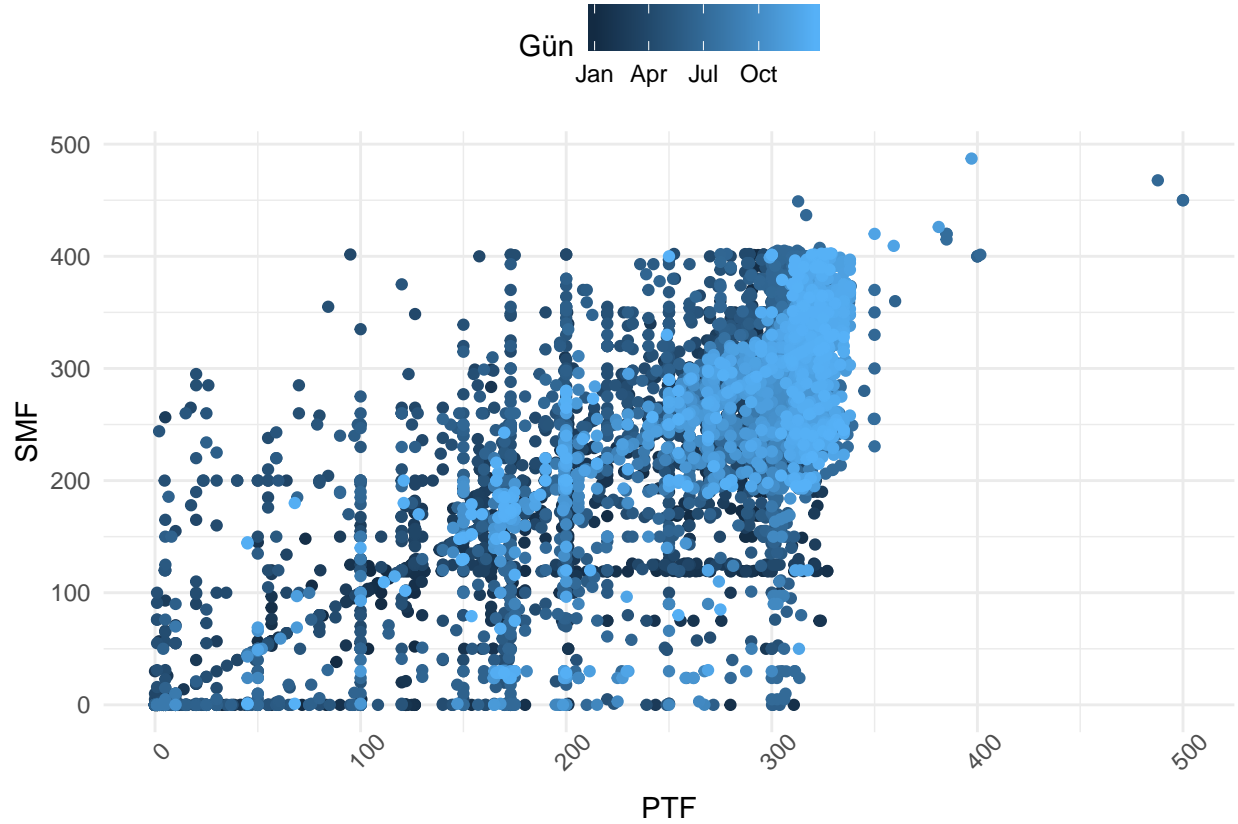
Grafiği görsel olarak daha güzel hale getirmek için öncelikle arka plandaki gri kısım `theme_minimal()` fonksiyonu yardımıyla kaldırılabilir.

```
ggplot(plot_df, aes(x=PTF, y=SMF, color=Gün)) +  
  geom_point() +  
  theme_minimal()
```



Eksenlerinin ismini deęiřtirmek, x eksenindeki yazıları döndürmek ve lejantı istenilen řekilde konumlamak için ařaęıdakiler uygulanmalıdır.

```
ggplot(plot_df, aes(x=PTF, y=SMF, color=Gün)) +  
  geom_point() +  
  theme_minimal() +  
  labs(x = "PTF",  
       y = "SMF") +  
  theme(axis.text.x = element_text(angle = 45), legend.position = "top")
```



Burada `labs()` fonksiyonu içinde `x` ve `y` eksenlerinin isimleri sırayla değiştirildi, `theme()` fonksiyonu içerisinde `axis.text.x = element_text(angle = 45)` komutu ile `x` eksenindeki yazılar 45 derece döndürüldü, `legend.position = "top"` komutu ile lejantın pozisyonu sayfanın üst kısmı olarak belirlendi.

3.3 Sütunlu Grafik (Bar Chart)

“Bar Chart”ta ise veriler sütunlarda gösterilir. “Scatter Plot”taki `geom_point()` yerine `geom_bar()` fonksiyonu, `aes()` içerisinde `color` yerine ise `fill` kullanılır. (Burada `geom_bar` içerisinde `stat="identity"` yazılması zorunludur.)

Örneğin Negatif Dengesizlik Fiyatı’nın ne kadarının PTF’den ne kadarının ceza payından geldiği Haziran ayı için saatlere göre incelenmek isteniyor.

Bu durumu incelemek için gerçekleştirilen veri manipülasyonu kodunu aşağıda bulabilirsiniz.

Burada bir önceki bölümde İleri Seviye örneklerden aşına olabileceğiniz `tidyr` paketinin `pivot_longer()` fonksiyonu bulunuyor. Bu fonksiyon temel olarak geniş tabloları belirli sütunlarını bir ortak sütun içerisinde birleştirip bunu yaparken de satır sayısını birleştirilen sütun sayısı oranında artıran bir fonksiyondur.

Bu paketin indirme ve yüklenme aşamaları için Gerekli Paketlerin İndirilip Yüklenmesi bölümüne göz atabilirsiniz.

Örneğimizle devam edecek olursak,

```
plot_df <- ptfsmf %>%
  mutate(Saat = hour(Tarih), Tarih = as.Date(Tarih), Fark_NDF_PTF = NDF - PTF) %>%
  filter(Tarih>"2019-05-31" & Tarih<"2019-07-01") %>%
  group_by(Saat) %>%
  summarise(PTF = mean(PTF), Fark_NDF_PTF = mean(Fark_NDF_PTF)) %>%
  ungroup() %>%
  pivot_longer(cols=c(PTF, Fark_NDF_PTF), names_to="Piyasa_Degiskeni", values_to="Fiyatlar")
```

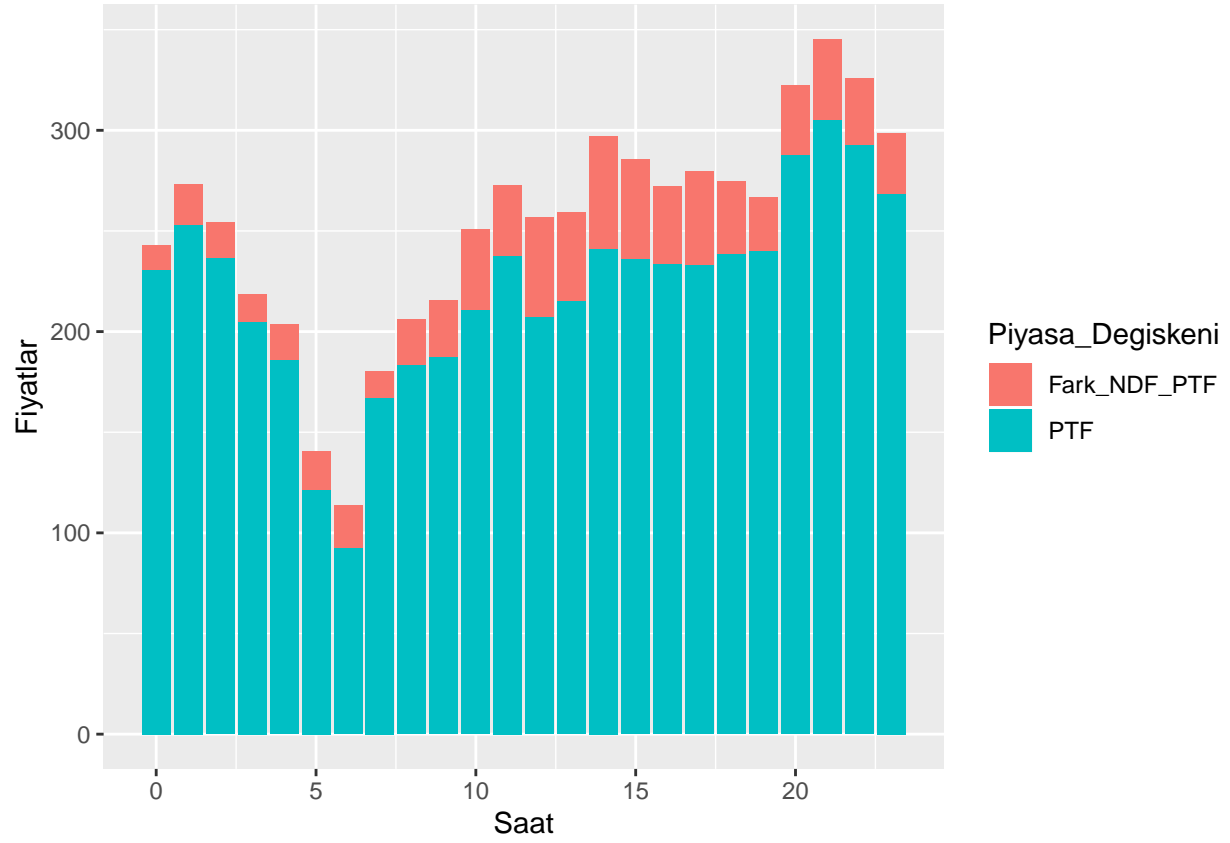
Tabloya göz atalım,

```
plot_df
```

```
## # A tibble: 48 x 3
##   Saat Piyasa_Degiskeni Fiyatlar
##   <int> <chr>          <dbl>
## 1     0 PTF            231.
## 2     0 Fark_NDF_PTF    12.4
## 3     1 PTF            253.
## 4     1 Fark_NDF_PTF    20.2
## 5     2 PTF            237.
## 6     2 Fark_NDF_PTF    17.4
## 7     3 PTF            205.
## 8     3 Fark_NDF_PTF    13.6
## 9     4 PTF            186.
## 10    4 Fark_NDF_PTF    17.4
## # ... with 38 more rows
```

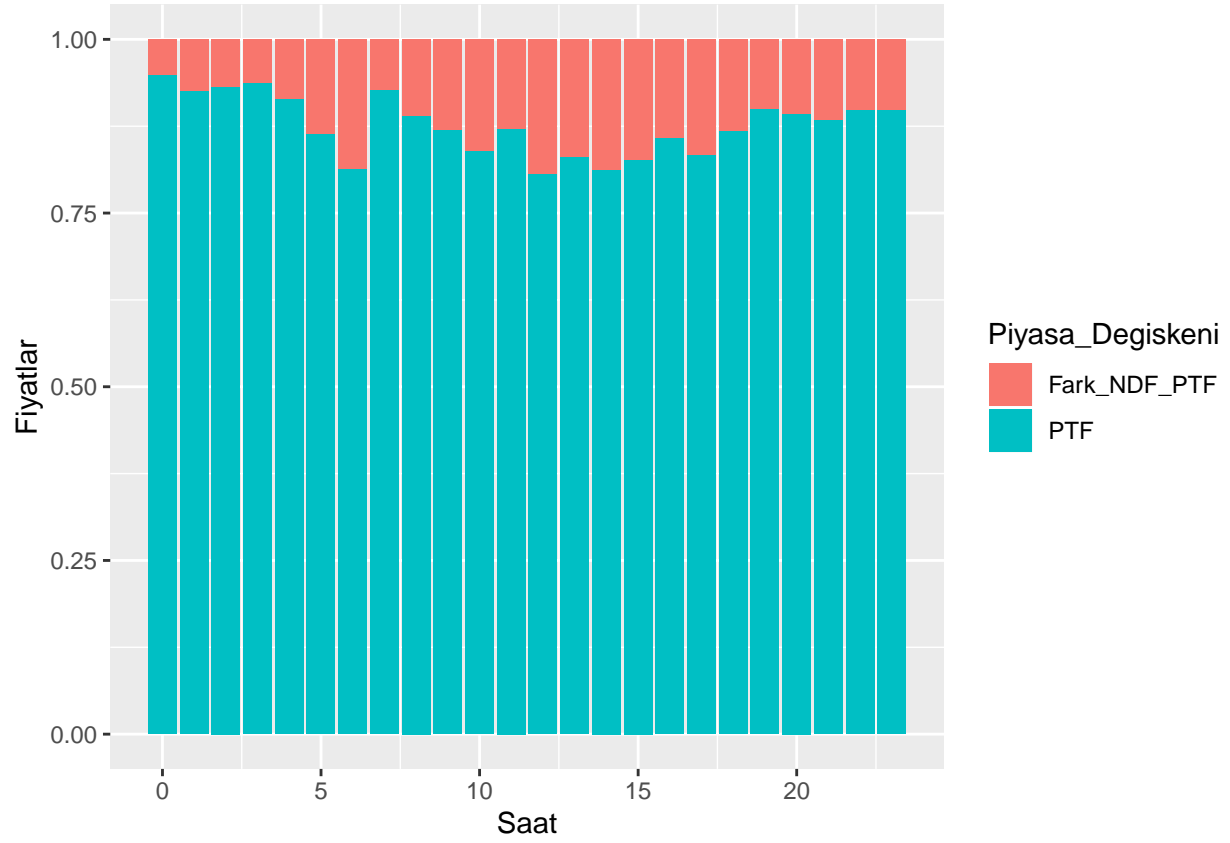
Bu şekilde bir tablo ortaya çıktı. Şimdi grafiği çizdirmek için,

```
ggplot(plot_df, aes(x=Saat, y=Fiyatlar, fill=Piyasa_Degiskeni)) +
  geom_bar(stat="identity")
```



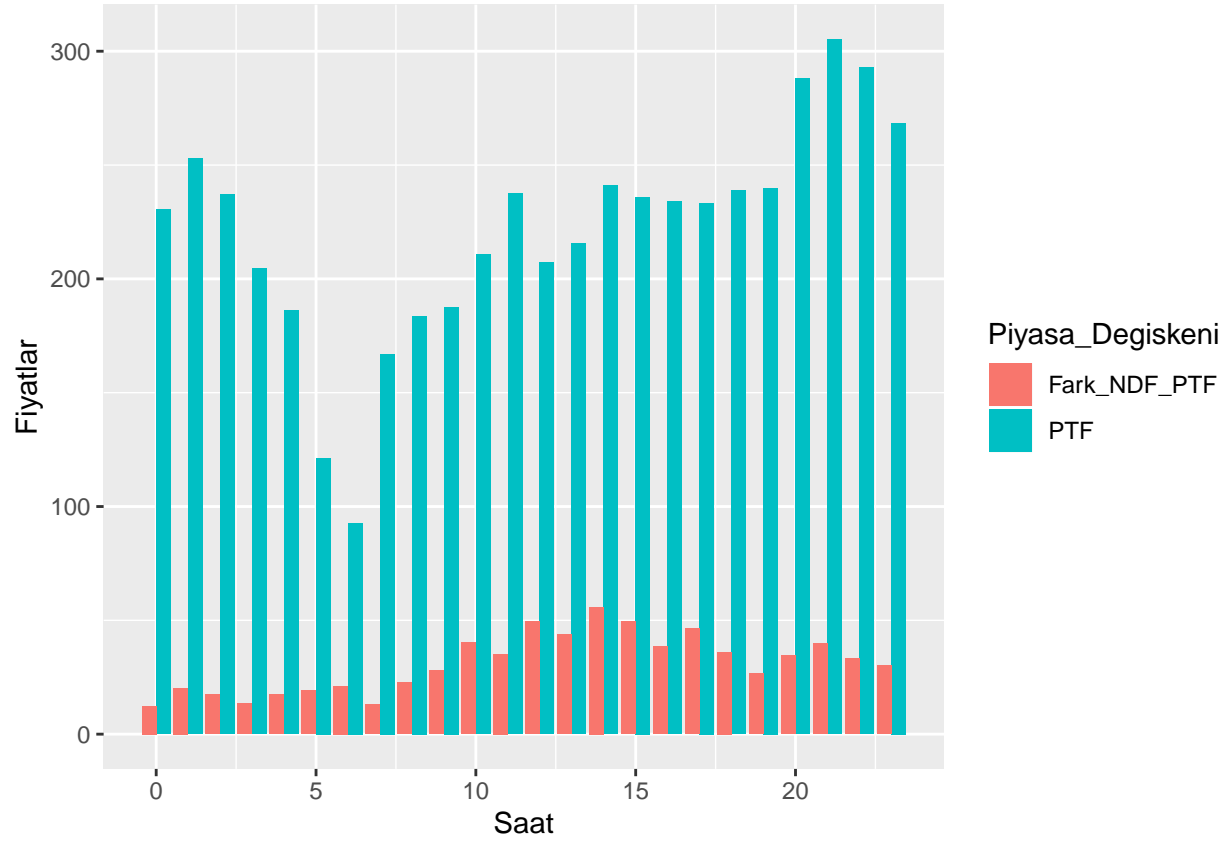
Bu grafięi yzdesel oranlarına gre tam olarak sıędırmak iin ise,

```
ggplot(plot_df, aes(x=Saat, y=Fiyatlar, fill=Piyasa_Degiskeni)) +  
  geom_bar(stat="identity", position="fill")
```



Son olarak da daha aşına olunduğu üzere sütunları yan yana yerleştirmek için,

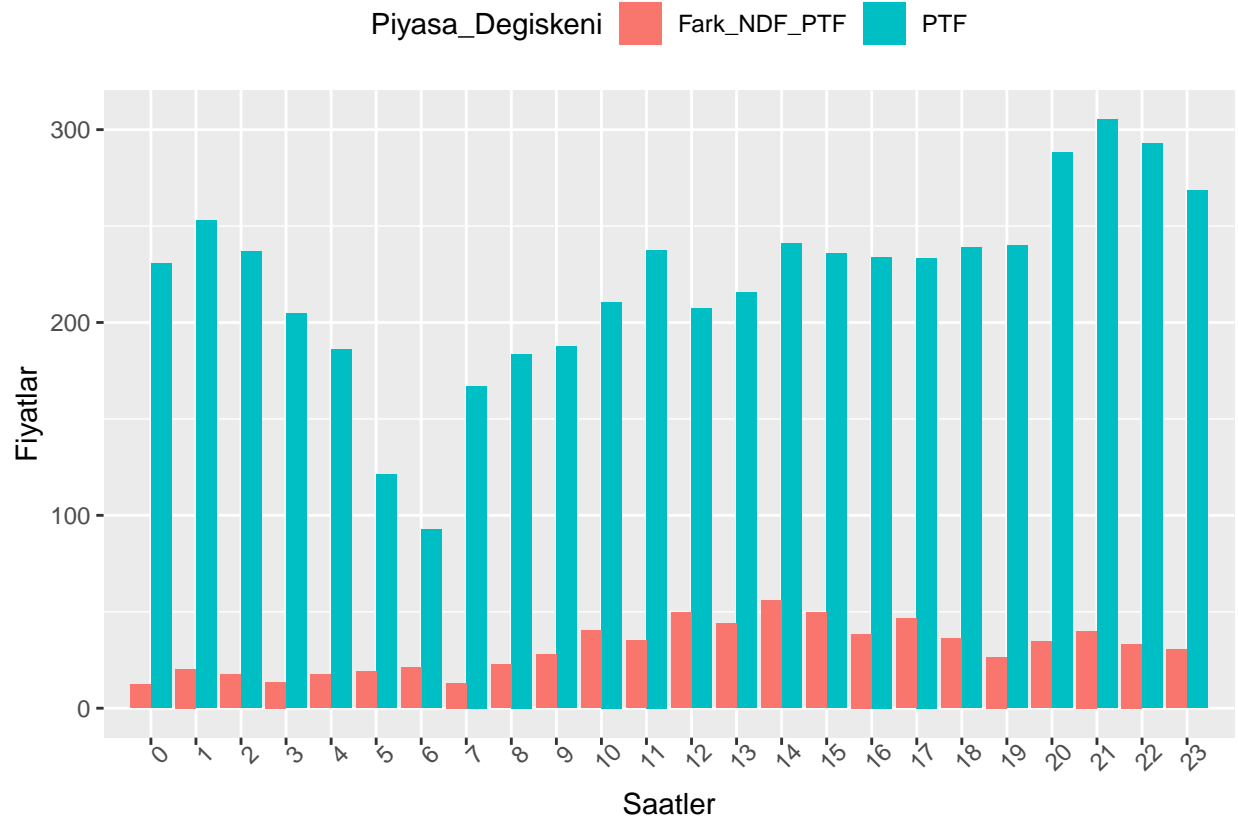
```
ggplot(plot_df, aes(x=Saat, y=Fiyatlar, fill=Piyasa_Degiskeni)) +  
  geom_bar(stat="identity", position="dodge")
```



Son iki grafikte dikkat edilmesi gereken nokta sütunları dikeyde oranları göz önüne alınarak sığdırmak için `position="fill"`, yan yana yerleştirmek için ise `position="dodge"` kullanılır.

En sonda yukarıda yapılan güzelleştirme operasyonları da uygulanırsa,

```
ggplot(plot_df, aes(x=Saat, y=Fiyatlar, fill=Piyasa_Degiskeni)) +
  geom_bar(stat="identity", position="dodge") +
  labs(x = "Saatler",
       y = "Fiyatlar") +
  scale_x_discrete(limits=c(0:23)) +
  theme(axis.text.x = element_text(angle = 45), legend.position = "top")
```

3.4 Çizgi Grafiği (Line Chart)

“Line Chart”larda ise tahmin edilebileceği üzere `geom_line()` kullanılacaktır. Örneğin Şubat ayı saatlik ortalama PTF ve SMF değerleri incelenmek isteniyor. Öncelikle veri manipülasyonu yapılırsa,

```
plot_df <- ptfsmf %>%
  mutate(Tarih = as_date(Tarih)) %>%
  filter(Tarih>"2019-01-31" & Tarih<"2019-03-01") %>%
  group_by(Tarih) %>%
  summarise(PTF = mean(PTF), SMF = mean(SMF)) %>%
  ungroup() %>%
  pivot_longer(cols=c(PTF, SMF), names_to="Piyasa_Degiskeni", values_to="Fiyatlar")
```

Bu şekilde bir tablo ortaya çıkıyor.

```
plot_df
```

```
## # A tibble: 56 x 3
##   Tarih      Piyasa_Degiskeni Fiyatlar
##   <date>      <chr>              <dbl>
## 1 2019-02-01 PTF                224.
## 2 2019-02-01 SMF                262.
## 3 2019-02-02 PTF                251.
## 4 2019-02-02 SMF                179.
```

```
## 5 2019-02-03 PTF 187.
## 6 2019-02-03 SMF 127.
## 7 2019-02-04 PTF 253.
## 8 2019-02-04 SMF 234.
## 9 2019-02-05 PTF 257.
## 10 2019-02-05 SMF 220.
## # ... with 46 more rows
```

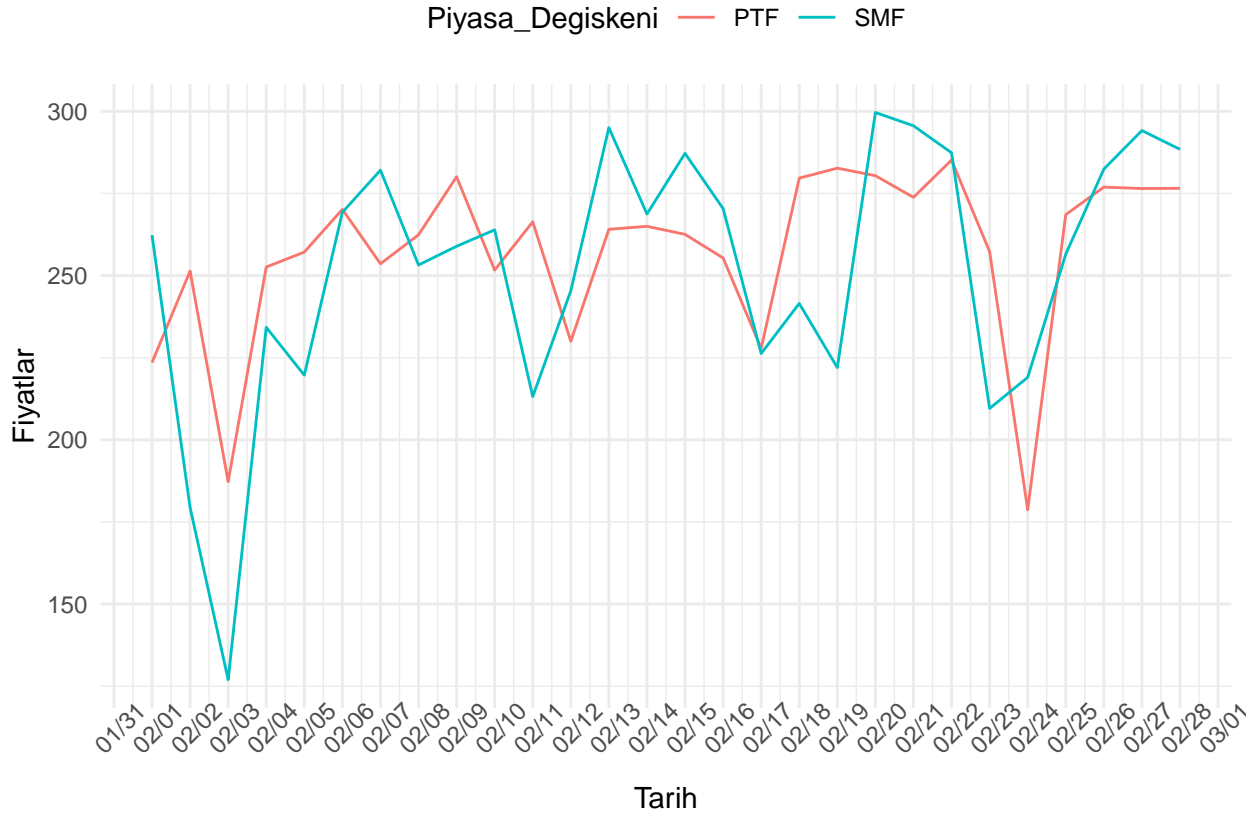
Şimdi grafik çizdirilirse,

```
ggplot(plot_df, aes(x=Tarih, y=Fiyatlar, color=Piyasa_Degiskeni)) +
  geom_line()
```



Yine daha güzel görünümlü bir grafik elde etmek için gerekli fonksiyonlar da yazılırsa,

```
ggplot(plot_df, aes(x=Tarih, y=Fiyatlar, color=Piyasa_Degiskeni)) +
  geom_line() +
  theme_minimal() +
  labs(x = "Tarih",
       y = "Fiyatlar") +
  scale_x_date(date_breaks = "1 day", date_labels = "%m/%d") +
  theme(axis.text.x = element_text(angle = 45), legend.position = "top")
```



Burada önceden gördüğümüz fonksiyonlara ek olarak, `scale_x_date(date_breaks = "1 day", date_labels = "%m/%d")` komutunu görüyoruz. Bu komut sayesinde `date` tipinde olan bir değişken istenilen formatta bastırılmış oluyor.

3.5 Isı Haritası (Heat Map)

Bu grafik türünde ise iki farklı değişken baz alınarak temelde gözlenmek istenen değişkeni farklı renk tonlarında göstererek incelenmesi hedefleniyor. Bu işlem için ise `geom_tile()` fonksiyonu kullanılıyor ve asıl olarak gösterilmek istenen değişken `fill` fonksiyonuna veriliyor.

Örneğin Temmuz ayı için `x` ekseninde saatler, `y` ekseninde ise ayın günleri olacak şekilde Piyasa Takas Fiyatı (PTF) değişkeninin ısı grafiği için (Öncelikle veri manipülasyonu yapılıyor.),

```
plot_df <- ptfsmf %>%
  mutate(Saat = hour(Tarih), Tarih = as_date(Tarih)) %>%
  filter(Tarih > "2019-06-30" & Tarih < "2019-08-01") %>%
  select(Saat, Tarih, PTF)
```

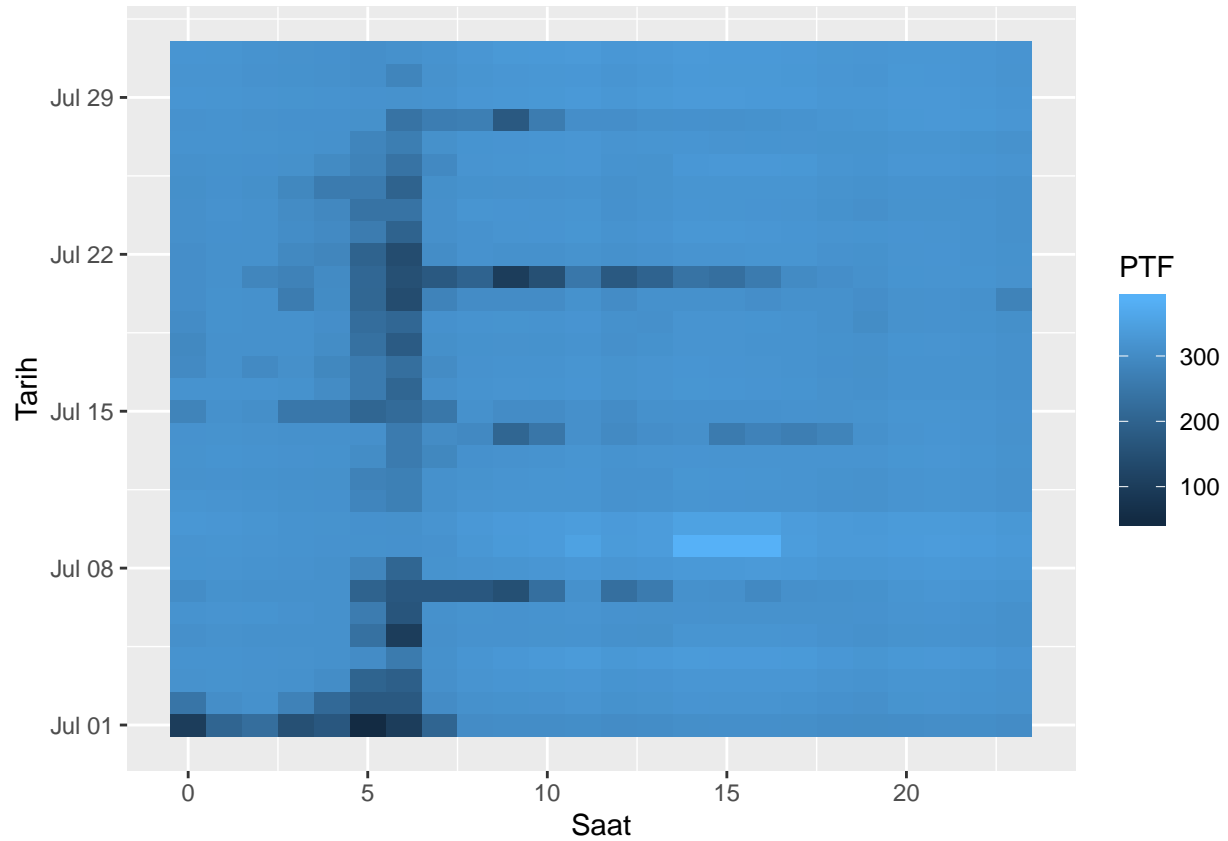
```
plot_df
```

```
## # A tibble: 744 x 3
##   Saat Tarih      PTF
##   <int> <date>    <dbl>
## 1     0 2019-07-01 100.
## 2     1 2019-07-01 204.
```

```
## 3      2 2019-07-01 227
## 4      3 2019-07-01 150
## 5      4 2019-07-01 170.
## 6      5 2019-07-01  50.0
## 7      6 2019-07-01 100
## 8      7 2019-07-01 204.
## 9      8 2019-07-01 300.
## 10     9 2019-07-01 302.
## # ... with 734 more rows
```

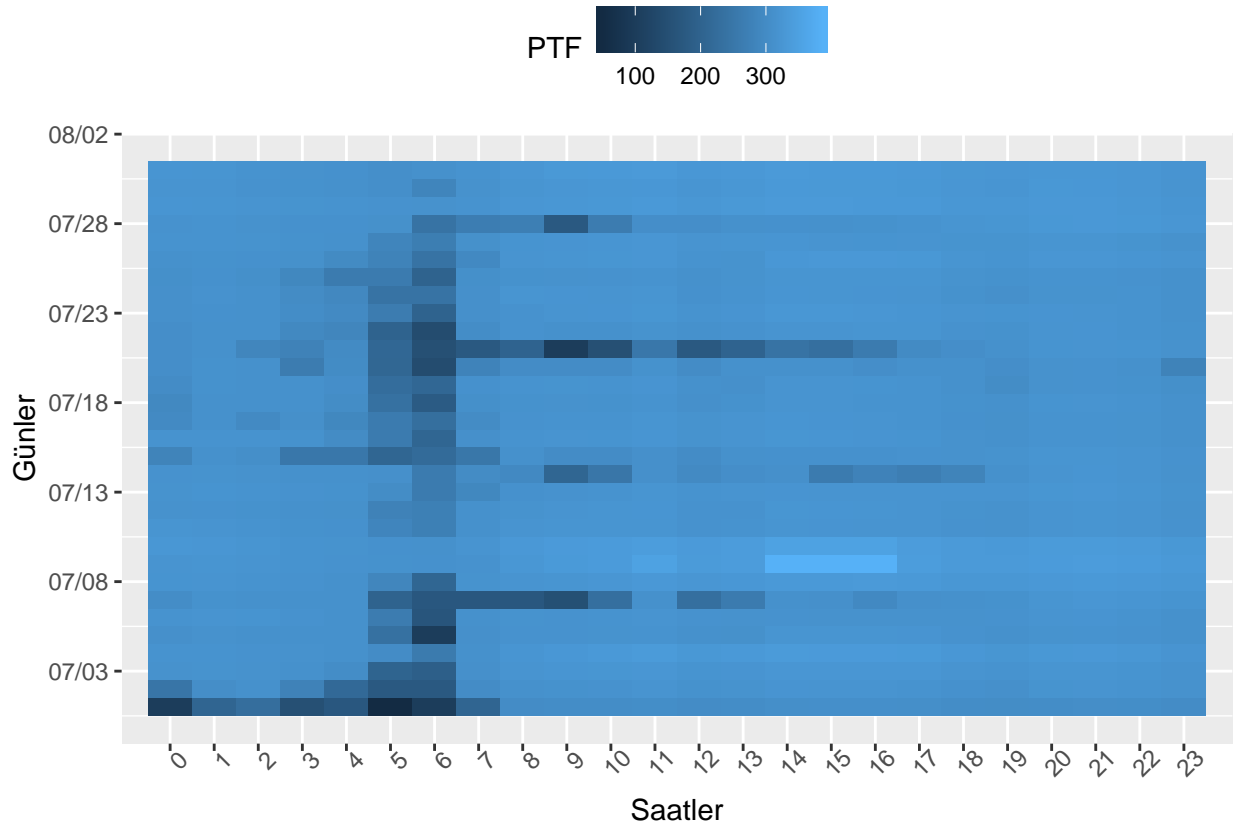
Bu şekilde bir tablo ortaya çıktı. Şimdi grafiği çizdirecek olursak,

```
ggplot(plot_df, aes(x=Saat, y=Tarih, fill=PTF)) +
  geom_tile()
```



Güzelleştirmek için yukarıdaki fonksiyonlar eklenirse,

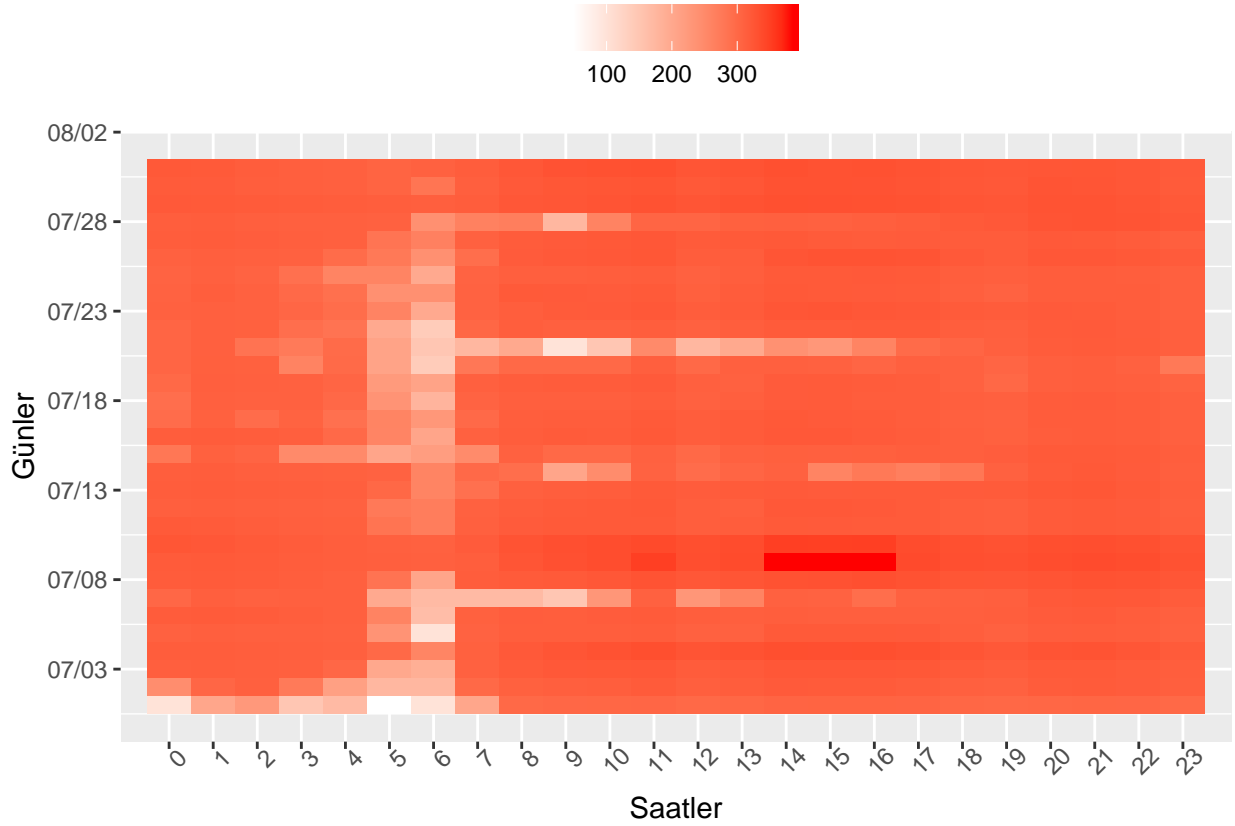
```
ggplot(plot_df, aes(x=Saat, y=Tarih, fill=PTF)) +
  geom_tile() +
  labs(x = "Saatler",
       y = "Günler") +
  scale_y_date(date_breaks = "5 day", date_labels = "%m/%d") +
  scale_x_discrete(limits=c(0:23)) +
  theme(axis.text.x = element_text(angle = 45), legend.position = "top")
```



Burada ise önceden aşına olmadığımız `scale_x_discrete(limits=c(0:23))` komutunu görüyoruz. Bu komut sayesinde x ekseninde 0'dan 23'e bütün sayıları bastırabiliyoruz. (`scale_y_date` komutu bir önceki bölümde anlatılan `scale_y_date` komutuyla tamamen aynı işlevde olup y eksenini ile ilgilenmektedir.)

Isı grafiğinin rengi değiştirilmek istenirse ise `scale_fill_gradient` fonksiyonu içerisinde iki renk belirtilerek aralarındaki renkler kullanılabilir.

```
ggplot(plot_df, aes(x=Saat, y=Tarih, fill=PTF)) +
  geom_tile() +
  scale_fill_gradient(low="white", high="red") +
  labs(x = "Saatler",
       y = "Günler") +
  scale_y_date(date_breaks = "5 day", date_labels = "%m/%d") +
  scale_x_discrete(limits=c(0:23)) +
  theme(axis.text.x = element_text(angle = 45), legend.position = "top",
        legend.title = element_blank())
```



Ekler

Yazarlar Hakkında

- *Baran Doğru*

Baran Doğru, Boğaziçi Üniversitesi Endüstri Mühendisliği öğrencisidir. Veri bilimine büyük ilgi duyan Baran Doğru Algopoly’de staj yaptığı dönemde bu dökümanın oluşturulmasında büyük pay sahibi olmuştur. Kendisine ulaşmak için LinkedIn hesabını ziyaret edebilirsiniz.

- *Selçuk Can Güven*

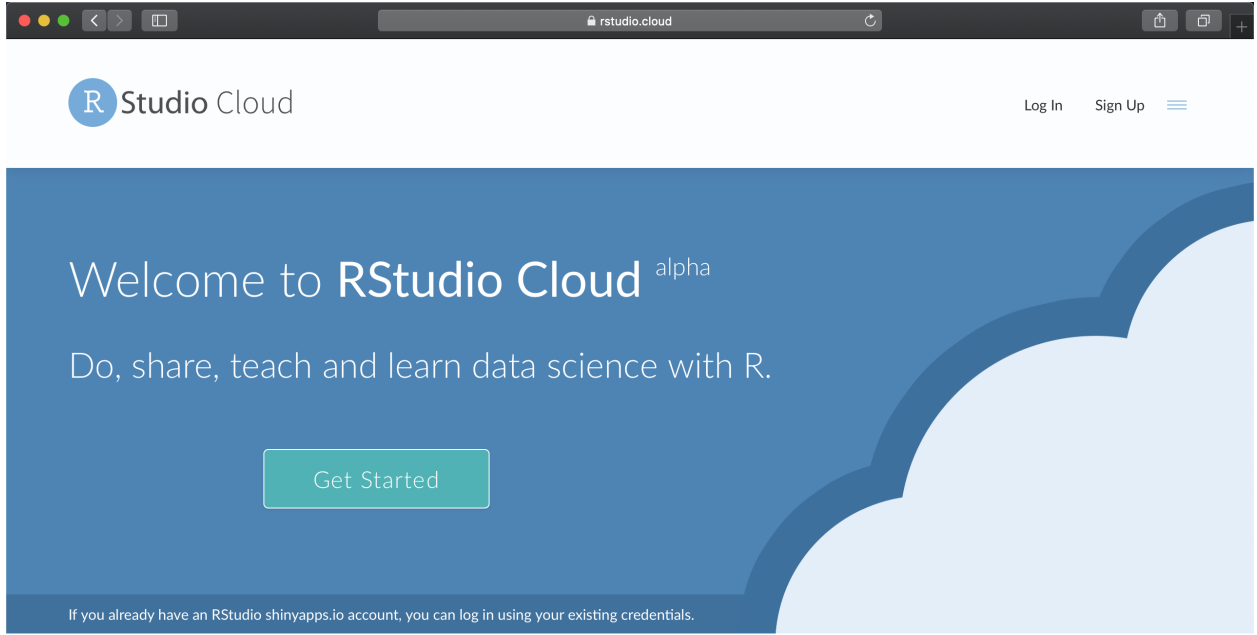
Selçuk Can Güven, Algopoly şirketinde İş Geliştirme Direktörü olarak kariyerine devam etmektedir. Bu dökümanın sektöre sunulabilir ve eğitim verilebilir hale getirilmesinde büyük katkıları olmuştur. Kendisine ulaşmak için LinkedIn hesabını ziyaret edebilirsiniz.

RStudio Cloud’da Çalışma

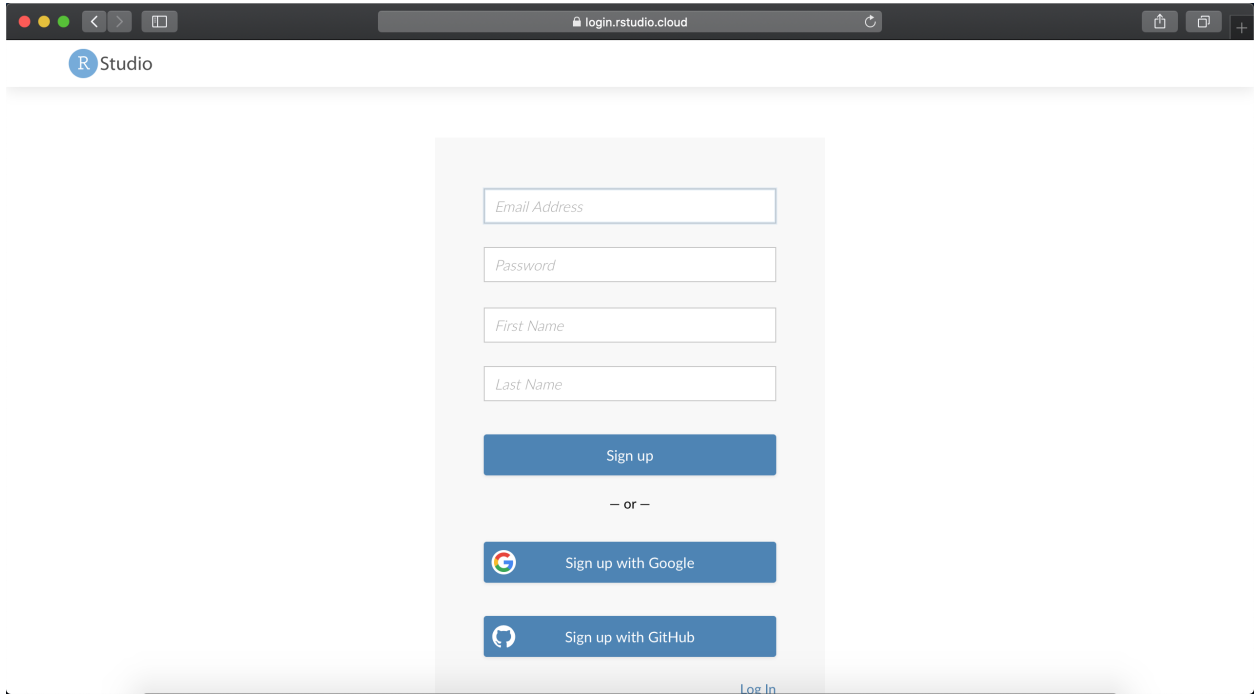
RStudio Cloud, R ve RStudio’yu bilgisayarınıza indirmeden çevrimiçi olarak kodlarınızı yazabileceğiniz, çalışmalarınızı iş arkadaşlarınızla rahatça paylaşabileceğiniz, gerekli paketleri bilgisayarınıza yükleyip yüklemediğinizi dert etmeyeceğiniz tamamen ücretsiz bir platformdur.

RStudio Cloud’a erişmek ve platformu kullanmak için aşağıdaki adımları izleyebilirsiniz.

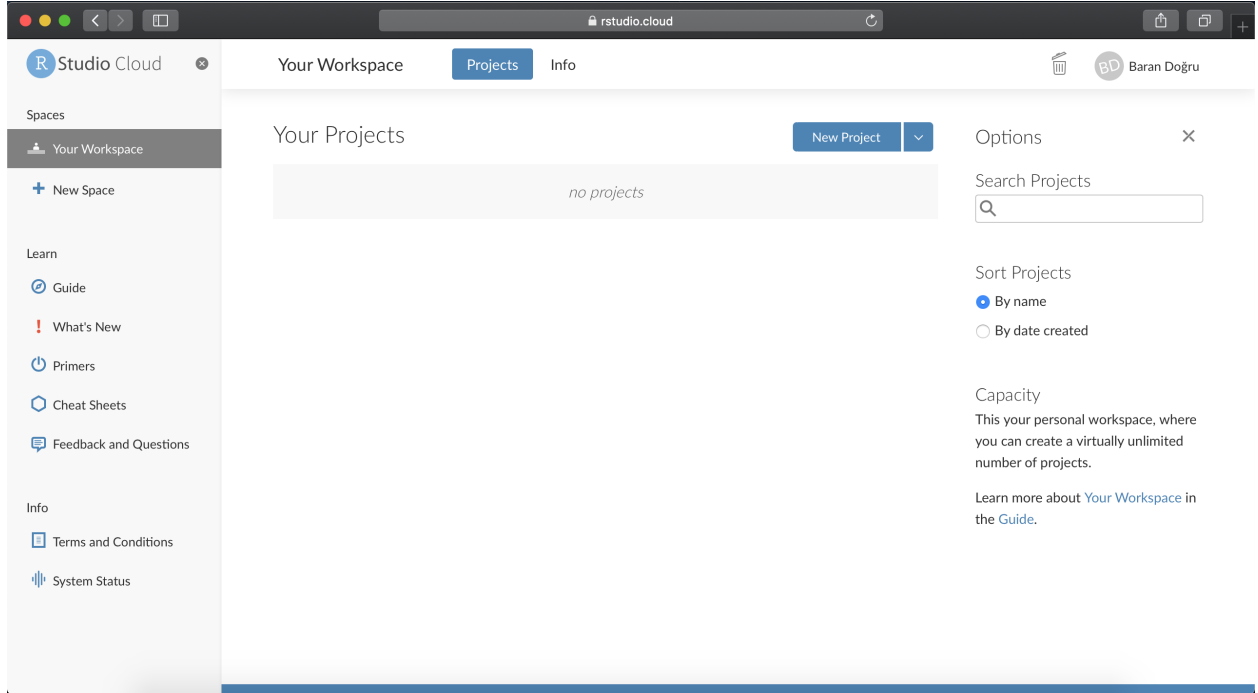
1. Tercih ettiğiniz web tarayıcıda “RStudio Cloud” yazarak aratılınca karşınıza çıkan ilk linke tıkladığınızda bu ekranla karşılaşacaksınız. Ana ekranın sağ üst köşesindeki “Sign Up” butonuna tıklayarak kayıt ekranına ulaşın.



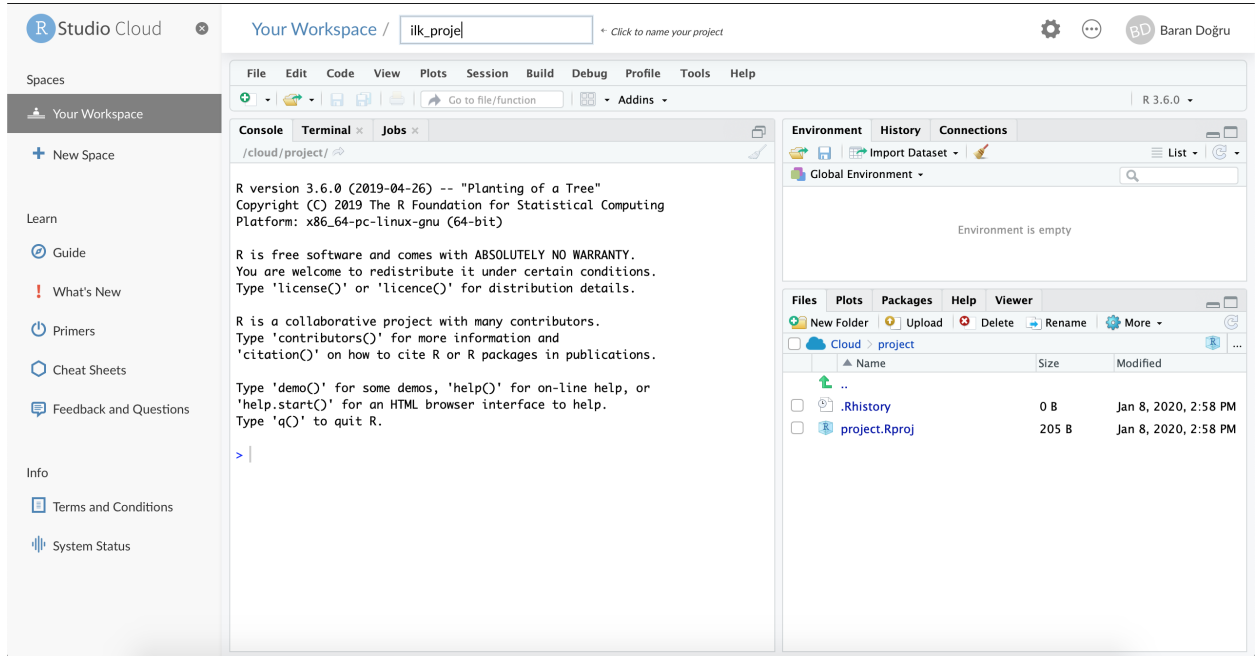
2. Karşınıza çıkan kayıt formunu doldurun.



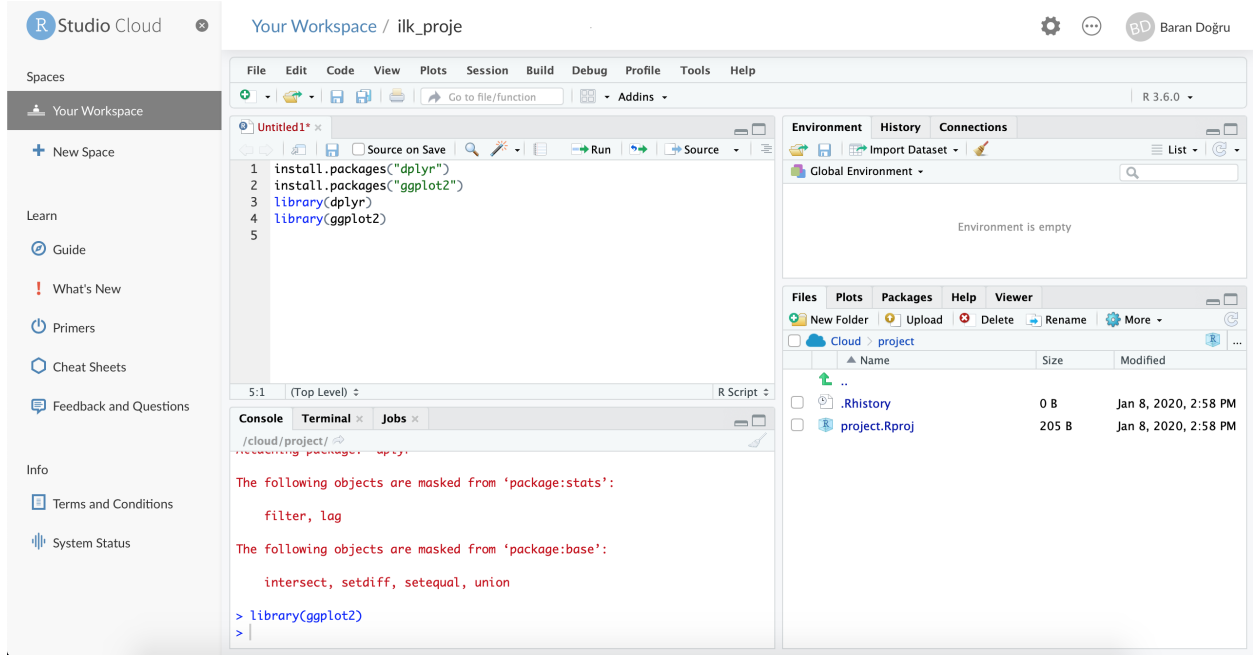
3. Kaydınızı tamamlayın ve hesabınıza giriş yapın. Artık RStudio Cloud’u kullanmaya hazırsınız.



4. İlk projenizi oluşturmak için “New Project” butonuna tıklayın.



5. Açılan ilk projenize R Script dosyası eklemek için ise üstteki bardan “File”, “New File” ve “R Script” sırasıyla seçin. Artık R’da kod yazmaya hazırsınız.



Veri Seti Düzenlemeleri

Bu bölümde ilgilenenler için veri setinin ham hali elde edildikten sonra dplyr ile Veri Manipülasyonu bölümünde kullanılan haline dönüştürmek için gereken kodlar paylaşılacaktır.

```
# pozitif ve negatif dengesizlik sutunlarinin isimlerinin duzenlenmesi
colnames(ptfsmf)[4] = "PDF"
colnames(ptfsmf)[5] = "NDF"

# tarih ve saatin POSIX formatina cevrilmesi
ptfsmf <- ptfsmf %>% mutate(Tarih_Yeni =
  as.POSIXct(ptfsmf$Tarih,format="%d.%m.%y %H:%M", "GMT")) %>%
  select(Tarih_Yeni, PTF, SMF, PDF, NDF) %>%
  rename(Tarih = Tarih_Yeni)

# geri kalan 4 sutunun numeric classina cevrilmesi
ptfsmf[,c(2:5)] <- lapply(ptfsmf[,c(2:5)],
  function(x) as.numeric(gsub(",", ".", gsub("\\.", "", x))))
```

Kaynakça

- dplyr Paketi : dplyr CheatSheet
- ggplot2 Paketi : ggplot2 Cheatsheet
- Bookdown Paketi ile Kitap Oluşturma : bookdown
- Veri Seti : EPIAŞ Raporlama Sayfası
- Piyasa Hakkında : Türkiye Elektrik Piyasası