Module 5 - R Studio & GitHub

Bu notlar, veri bilimi projelerinde R ve RStudio ile istatistiksel analiz, veri görselleştirme ve modelleme süreçlerinin nasıl yürütüleceğini; ayrıca Git ve GitHub üzerinden sürüm kontrolü ve ekip çalışmasının nasıl sağlanacağı konularını bütüncül bir tasarım ve alt başlıklar çerçevesinde ele almaktadır. Notların hazırlanmasında, IBM Data Science Professional Certificate içeriği, internet kaynakları ve yapay zeka araçlarının sunduğu bilgiler referans alınmıştır.

---

1. R ve RStudio'ya Giriş

1.1 R Nedir?

- **İstatistiksel Programlama Dili:**

  R, veri işleme, manipülasyon, istatistiksel çıkarım, veri analizi ve makine öğrenimi algoritmaları için kullanılmakta olup; akademi, sağlık sektörü ve kamu kurumları tarafından yaygın biçimde tercih edilmektedir.

- **Veri Kaynakları:**

  Düz metin dosyalar, veritabanları, web kaynakları ve SPSS ya da STATA gibi istatistiksel yazılımlar aracılığıyla elde edilen veriler, R'nin kolay veri içe aktarma fonksiyonları sayesinde rahatlıkla işlenebilmektedir.

- **Neden R Tercih Edilir?**
  - **Kolay Fonksiyon Kullanımı:** Karmaşık ek kurulumlara gerek kalmadan pek çok işlevin doğrudan kullanılabilmesi sağlanmaktadır.
  - **Zengin Görselleştirme:** Güçlü paketleri sayesinde kaliteli grafikler ve görseller üretilmekte, veri görselleştirme sürecinin etkinliği artırılmaktadır.
  - **Entegre Paketler:** Veri analizi ve istatistiksel yöntemler için dahili veya tek komutla eklenebilen paketlerin bulunması, R'nin tercih edilme nedenleri arasında değerlendirilmektedir.

Not: R, özellikle istatistiksel analiz ve veri görselleştirme odaklı projelerde hızlı prototipleme için ideal bir seçenek olarak kullanılmaktadır.

1.2 RStudio Ortamı

- **RStudio:**

  R dili için popüler bir geliştirme ortamı (IDE) olarak kullanılmakta; R kodlarının geliştirilmesi, çalıştırılması ve yönetilmesi amacıyla işlevsel özellikler sunmaktadır.

- **RStudio Bileşenleri:**

  1. **Editör:** Kod renklendirme (syntax highlighting) ve doğrudan kod çalıştırma desteği sağlanmaktadır.
  2. **Konsol:** R komutlarının elle girilerek anlık çıktılar alınması mümkün kılınmaktadır.
  3. **Workspace & History:**
     - **Workspace:** Oturumda oluşturulan R nesnelerinin listelendiği alan sunulmaktadır.
     - **History:** Daha önce çalıştırılan komutların geçmişi görüntülenmektedir.
  4. **Files, Plots, Packages, Help:**
     - **Files:** Çalışma dizinindeki dosyaların görüntülendiği bölüm sağlanmaktadır.
     - **Plots:** Oluşturulan grafiklerin geçmişi saklanmakta ve dışa aktarılabilmektedir.
     - **Packages:** Kurulu R paketlerinin yönetilmesi, yüklenmesi ve güncellenmesi işlemleri gerçekleştirilmektedir.
     - **Help:** R ve paketler hakkında dokümantasyona erişim sağlanmaktadır.

Not: RStudio, kod yazımından görselleştirmeye kadar tüm sürecin tek bir ortamda yönetilmesini sağlayarak veri analizi deneyiminin kolaylaştırılmasına katkıda bulunmaktadır.

1.3 Veri Bilimi için R Paketleri

- **dplyr:** Veri manipülasyonu (filtreleme, düzenleme, birleştirme) işlemlerinde kullanılmakta olup verinin hızlıca işlenmesini sağlamaktadır.
- **stringr:** Metin işlemleri, pattern eşleştirme ve düzenli ifadeler konusunda destek sunulmaktadır.
- **ggplot2:** Veri görselleştirme (histogram, çubuk grafik, dağılım grafiği vb.) ve katmanlı grafik yapısı sayesinde esnek ve estetik görseller üretilmektedir.
- **caret:** Makine öğrenimi algoritmalarının tek arayüzde yönetilmesi ve modelleme işlemlerinin gerçekleştirilmesi sağlanmaktadır.

Not: Bu paketlerin, R dilini kullanarak veri bilimi projelerinde temel araçlar olarak sıklıkla başvurulan unsurlar olduğu vurgulanmaktadır.

---

2. R ile Veri Görselleştirme

2.1 R'da Veri Görselleştirme Seçenekleri

- **Yerleşik Grafik Fonksiyonları:**

  `plot()` gibi fonksiyonlar kullanılarak basit dağılım grafikleri oluşturulabilmekte; farklı argümanlar ile grafik üzerinde çizgi ekleme veya başlık verme işlemleri gerçekleştirilebilmektedir.

- **Gelişmiş Paketler:**
  - **ggplot2:** Katmanlı (layered) yapısı sayesinde güçlü ve esnek grafikler üretilmektedir.
  - **plotly:** İnteraktif web tabanlı grafiklerin oluşturulması ve HTML formatında dışa aktarılması sağlanmaktadır.
  - **lattice:** Çok değişkenli veri setlerinin karmaşık ancak hızlı görselleştirilmesi mümkün kılınmaktadır.
  - **leaflet:** Etkileşimli harita tabanlı görselleştirme desteği sunulmaktadır.

Not: R, hem yerleşik fonksiyonlarla hızlı çözümler sunmakta hem de ihtiyaç durumunda gelişmiş paketlerden yararlanılarak zengin görselleştirme imkânı

sağlamaktadır.

2.2 `plot()` Fonksiyonu

- **Basit Kullanım:** `plot(veri)` komutu ile verideki değerler indekse karşılık gelecek şekilde dağılım grafiği oluşturulmakta; `lines()` fonksiyonu ile grafiğe çizgi eklenmekte, `title()` fonksiyonu ile görsele başlık verilebilmektedir.

Örnek:

```
x ← c(1, 2, 3, 4, 5)
plot(x, type="o")  # Noktalar ve çizgi
title("Örnek Plot")
```

Not: Küçük veri setlerinde hızlı bir görselleştirme elde edilmesi sağlanmaktadır.

2.3 `ggplot2` Kütüphanesi

- **Katmanlı Yapı (Grammar of Graphics):**

  `ggplot()` fonksiyonu içinde veri kümesi, eksen tanımları ve geometrik nesneler (ör. `geom_point()`) katmanlar halinde eklenmekte; böylece grafikler esnek ve özelleştirilebilir şekilde oluşturulmaktadır.

- **Örnek Kullanım (Mtcars):**

  ```
  library(ggplot2)
  ggplot(data=mtcars, aes(x=mpg, y=wt)) +
    geom_point() +
    ggtitle("Mtcars Dağılım Grafiği") +
    labs(x="Miles/Gallon", y="Ağırlık")
  ```

  - `ggtitle()` ve `labs()` fonksiyonları ile başlık ve eksen isimleri düzenlenebilmektedir.

- **GGally:**

  Birden çok grafik veya karmaşık veri dönüşümlerinin tek komutla yapılmasını kolaylaştıran genişletme paketi olarak kullanılmaktadır.

Not: `ggplot2`, veri bilimi projelerinde estetik ve fonksiyonel grafikler üretilmesi açısından sıklıkla tercih edilen temel kütüphanelerden biri olarak değerlendirilmektedir.

---

3. Özet (R ve RStudio)

Bu bölümde, **R** dilinin temel özellikleri, **RStudio** geliştirme ortamının bileşenleri, veri bilimi için kullanılan temel R paketleri ve R ile veri görselleştirme seçenekleri detaylı biçimde ele alınmaktadır.

- R, istatistiksel analiz ve veri görselleştirme ihtiyaçlarını karşılamak üzere güçlü ve tercih edilen bir dil olarak konumlandırılmaktadır.
- RStudio, kod düzenleme, konsol kullanımı, dosya ve paket yönetimi gibi tüm süreçleri tek bir ortamda birleştirerek verimli çalışma imkânı sağlamaktadır.
- `ggplot2` gibi paketlerin katmanlı grafik yapısı, veri görselleştirme sürecinde esnek ve estetik çözümler sunmaktadır.

---

4. Git ve GitHub

Bu bölümde, **Git** ve **GitHub** kavramlarına temelden başlanarak, sürüm kontrolü, Git komutları, repository yönetimi, branch (dal) yönetimi ve pull request süreçleri detaylandırılmaktadır.

4.1 Sürüm Kontrolü (Version Control) Nedir?

Projelerde yer alan dosyaların sürekli güncellenmesi ve farklı sürümlerinin oluşması durumunda yapılan değişikliklerin kayıt altına alınması, sürüm kontrolü sistemleri aracılığıyla sağlanmaktadır.

- **Değişiklik Takibi:**

  Her düzenleme adımının “kim tarafından, ne zaman ve hangi satırlarda” gerçekleştirildiği kayıt altına alınmakta; gerekirse önceki bir sürüme dönüş sağlanabilmektedir.

- **Ekip Çalışması:**

Birden fazla kişinin aynı dosya üzerinde çalışması durumunda, değişikliklerin çakışmadan birleşmesi desteklenmektedir.

- **Proje Organizasyonu:**

  Farklı özellikler veya denemeler için dal (branch) açılarak, ana koddaki kararlılığın korunması sağlanmaktadır.

4.2 Git'e Giriş

Git, dağıtık sürüm kontrol sistemleri arasında en popüler olanı olarak değerlendirilmektedir.

4.2.1 Git'in Temel Özellikleri

- **Ücretsiz ve Açık Kaynak:**

  GNU GPL lisansı kapsamında dağıtılmakta; komut satırı veya çeşitli arayüz araçlarıyla kullanılabilmektedir.

- **Dağıtık Mimari:**

  Her kullanıcının kendi bilgisayarında projenin tam kopyasını (tüm geçmişiyle birlikte) tutabilmesi sağlanmaktadır.

- **Performans:**

  Büyük dosya ve projelerle başa çıkacak şekilde optimize edilmiş, kod değişikliklerinin hızlı bir şekilde birleştirilmesi imkânı sunulmaktadır.

Not: Git, metin dosyalarının yanı sıra resim, belge ve veri setleri gibi farklı formatlardaki dosyaların da sürüm kontrolü altında tutulmasını desteklemektedir.

4.2.2 Git Temel Komutları

- **git init:**

  Yeni bir proje klasörünü Git tarafından izlenebilir bir depoya dönüştürmekte; `.git` adlı gizli klasör oluşturulmaktadır.

- **git add <dosya veya klasör>:**

  Yapılan değişikliklerin sahneleme alanına eklenmesi sağlanmaktadır.

- **git status:**

Hangi dosyaların değiştirildiği ve sahneleme durumlarının gösterilmesi desteklenmektedir.

- **git commit -m "Mesaj":**

  Sahneleme alanındaki değişikliklerin kalıcı olarak depoya kaydedilmesi ve yeni bir versiyon oluşturulması sağlanmaktadır.

- **git reset:**

  Yapılan değişikliklerin veya commit'lerin geri alınması desteklenmektedir.

- **git log:**

  Yapılan commit'lerin tarihçesi, mesajları ve yazar bilgileri listelenmektedir.

- **git branch, git checkout, git merge:**

  Branch (dal) yönetimi, farklı özelliklerin geliştirilmesi ve bu özelliklerin ana koda birleştirilmesi işlemleri gerçekleştirilmektedir.

Not: Bu komutlar, Git'in temel yapı taşlarını oluşturmakta olup, ileri seviye komutlar zamanla ihtiyaç duyulmaktadır.

4.3 GitHub Nedir?

GitHub, Git depolarını internet ortamında barındıran, projelerin uzaktan erişimini ve ekip çalışmasını kolaylaştıran en popüler platformlardan biri olarak değerlendirilmektedir.

- **Uzaktan Erişim:**

  Depoların online olarak saklanması ve başkalarının erişimine açılması sağlanmaktadır.

- **İş Birliği Araçları:**

  Pull request, Issues, Wiki, Proje Yönetimi gibi özelliklerle ekip çalışmasının desteklenmesi sağlanmaktadır.

- **Profil ve Topluluk:**

  Kişisel GitHub profili oluşturarak açık kaynak projelere katkı sağlanmakta ve projeler sergilenebilmektedir.

4.3.1 GitHub Hesabı Oluşturma

- GitHub adresine gidilerek Sign up işlemi gerçekleştirilmektedir; kullanıcı adı, e-posta ve şifre bilgileri girilmekte ve e-posta onayı ile hesap aktif hale getirilmektedir. Ücretsiz plan, çoğu ihtiyaç için yeterli görülmektedir.

Not: GitHub, web arayüzü üzerinden depo oluşturma, dosya ekleme veya düzenleme imkânı sunmakla birlikte, Git komut satırı kullanımına hakim olunması önerilmektedir.

4.4 GitHub'da Repository Oluşturma ve Dosyalarla Çalışma

4.4.1 Yeni Repository Oluşturma

- Sağ üstteki "+" butonuna tıklanarak **New repository** seçeneği ile depo oluşturulmakta; depo adı, açıklama ve görünürlük ayarları belirlenmekte, "Initialize this repository with a README" seçeneği işaretlenerek başlangıçta bir README.md dosyası oluşturulabilmektedir.

4.4.2 Dosya Ekleme veya Düzenleme

- **Yeni Dosya Oluşturma:**

  Depo içindeyken "Add File" → "Create new file" seçeneği kullanılarak dosya oluşturulmakta; dosya adı ve içerik girildikten sonra commit mesajı eklenerek dosya kaydedilmektedir.

- **Var Olan Dosyayı Düzenleme:**

  Depodaki dosya açılarak sağ üstteki kalem simgesine tıklanmakta; değişiklikler yapılıp commit mesajı ile kaydedilmektedir.

- **Yerelden Dosya Yükleme:**

  "Add File" → "Upload files" seçeneği ile dosyalar yüklenmekte; dosya seçimi yapıldıktan sonra commit mesajı ile onaylanmaktadır.

Not: Her commit, depoda bir versiyon olarak kaydedilmekte; commit geçmişi incelenerek değişiklikler gerektiğinde geri alınabilmektedir.

4.5 Branch Yönetimi ve Pull Request

4.5.1 Branch (Dal) Nedir?

Branch, projenin mevcut kararlı sürümünün (master/main) bir kopyası olarak oluşturulmakta; yeni özellik geliştirme veya hata düzeltme çalışmaları için ayrı bir dal oluşturularak ana kodun bozulmaması sağlanmaktadır.

4.5.2 Branch Oluşturma ve Merge Süreci

- GitHub web arayüzünde “Branch: master/main” açılır menüsünden yeni branch oluşturulabilmekte; oluşturulan child branch üzerinde yapılan değişiklikler commit edilmekte ve ana dala yansıtılmadan saklanmaktadır.
- Değişiklikleri ana dala eklemek amacıyla Pull Request oluşturulmakta; ekip arkadaşlarının incelemesi ve onayı sonrası Merge pull request işlemi gerçekleştirilmekte.
- İşlem tamamlandıktan sonra, gereksiz branch’ler silinerek depo düzeni sağlanmaktadır.

---

5. Özet

Bu notlarda, **R ve RStudio** ile veri analizi, görselleştirme ve temel R paketlerinin kullanımı; **Git** ve **GitHub** üzerinden sürüm kontrolü, repository yönetimi, branch yönetimi ve pull request süreçleri detaylı biçimde ele alınmaktadır.

- **R**, istatistiksel analiz ve veri görselleştirme için tercih edilen güçlü bir dil olup, RStudio ile entegre çalışması sayesinde tüm veri bilimi süreci tek bir ortamda yönetilebilmektedir.
- **Git** ve **GitHub**, projelerin sürüm kontrolü ve ekip çalışması açısından vazgeçilmez araçlar olarak kullanılmakta; temel komutlar, dal yönetimi ve pull request süreçleri ile proje akışı düzenlenmekte ve verimlilik sağlanmaktadır.

Her proje için uygun araç ve yöntemlerin seçilmesi, entegre bir yaklaşımın benimsenmesi ve sürekli öğrenme ile desteklenmesi, başarılı veri bilimi ve yazılım geliştirme süreçlerinin temelini oluşturmaktadır.
