Module 4 - Jupyter Notebook

Bu notlar, veri bilimi projelerinde interaktif kod yazma, analiz yapma ve deneysel çalışmaları belgeleyebilme imkânı sağlayan Jupyter Notebooks ve JupyterLab uygulamalarının tanım, özellik, kurulum ve kullanım pratiklerini içermektedir. Notların hazırlanması sırasında, IBM Data Science Professional Certificate içeriği, internet kaynakları ve yapay zeka araçlarının sunduğu bilgiler referans alınmış; bu sayede Jupyter ekosisteminin temel bileşenleri detaylı biçimde ele alınmıştır.

---

1. Jupyter Notebooks

1.1 Tanım ve Amacı

Jupyter Notebooks, tarayıcı tabanlı bir uygulama olarak kod, denklemler, görselleştirmeler, anlatı metinleri ve bağlantıları tek bir dosyada birleştirerek saklama imkânı sağlamaktadır. Bu sayede veri bilimcilerin deneysel çalışmalarını ve analiz sonuçlarını, geleneksel laboratuvar defterlerine benzer şekilde, yeniden üretilebilir biçimde belgelemeleri mümkün kılınmaktadır.

1.2 Özellikler

- **Birleştirilmiş Yapı:**

  Açıklayıcı metinler, kod blokları ve kod çıktıları aynı dosyada yer alarak bütünsel bir çalışma ortamı oluşturulmaktadır.

- **Kolay Paylaşım:**

  Not defterleri, HTML veya PDF gibi farklı formatlara dönüştürülerek paylaşım imkânı sunmakta; bulut platformları veya e-posta yoluyla erişim kolaylaştırılmaktadır.

Not: Jupyter Notebooks, veri analizi süreçlerinin interaktif ve dokümantasyonunun sağlanmasında büyük kolaylıklar sunmaktadır.

---

2. JupyterLab

2.1 Tanım ve İşlev

JupyterLab, Jupyter Notebooks'un gelişmiş sürümü olarak tanımlanmakta; çoklu not defterleri, metin düzenleyiciler, terminaller ve özel bileşenleri bir araya getirerek genişletilmiş ve entegre bir çalışma ortamı sunmaktadır.

- **Uyumluluk:**CSV, JSON, PDF, Vega gibi çeşitli dosya formatlarını desteklemekte; açık kaynaklı ve esnek yapısı sayesinde eklenti ve uzantılarla geliştirilebilmektedir.

Not: JupyterLab, birden fazla dosya ve not defterinin aynı anda yönetilmesi gereken büyük ölçekli projeler için ideal bir ortam sunmaktadır.

2.2 Bulut ve Yerel Kurulum

2.2.1 Bulut Tabanlı Hizmetler

- **Bulut Platformları:**

  Skills Network Labs, IBM Cloud ve Google Colab gibi platformlar üzerinden tarayıcı tabanlı olarak kullanılabilmekte; yerel kurulum gerekmemesi sebebiyle hızlıca çalışmaya başlanabilmektedir.

- **Dosya Transferi:**

  Standart IPython Notebook dosya formatı desteklenmekte, böylece dosyalar kolayca içe ve dışa aktarılabilmektedir.

2.2.2 Yerel Kurulum

- **Kurulum Yöntemleri:**

  Komut satırından `pip install jupyter` veya `conda install jupyter` komutlarıyla kurulabilmekte; ayrıca Anaconda dağıtımı ile Python ve veri bilimi kütüphaneleri tek paket halinde sunulmaktadır.

- **Entegre Çalışma Ortamı:**

  Anaconda, Jupyter ve JupyterLab'ı varsayılan olarak içeren entegre bir ortam sağlamaktadır.

Not: Projeyi bulut ortamında mı yoksa yerel makinede mi yürüteceğiniz, internet bağlantısı, performans gereksinimleri ve kişisel tercihler doğrultusunda değerlendirilmektedir.

2.3 Eğitim Kapsamı

- **Skills Network Labs:**Sanal ortamda barındırılan JupyterLab sürümü sayesinde, kullanıcıların kendi cihazlarına kurulum yapmadan veri bilimi pratiklerini deneyimlemeleri desteklenmektedir.

---

3. Jupyter Notebooks ile Çalışma Pratikleri

3.1 Not Defteri Başlatma ve Temel İşlemler

- **Not Defteri Oluşturma:**

  Bulut tabanlı platformlarda “Open tool” veya “New Notebook” seçenekleriyle not defteri oluşturulabilmekte; dosya adı, “File → Rename Notebook” menüsüyle değiştirilebilmektedir.

- **Kod Hücreleri ile Çalışma:**

  Kod hücreleri, Shift + Enter veya “Run Selected Cells” komutlarıyla çalıştırılmakta; tüm hücreler “Run All Cells” komutu ile tek seferde yürütülebilmektedir. Yeni hücre ekleme, silme ve hücreleri taşıma işlemleri araç çubuğu veya ilgili menüler üzerinden gerçekleştirilebilmektedir.

Not: Hücre yapısı, kodun adım adım çalıştırılmasını sağlayarak okunabilirlik ve hata ayıklama süreçlerini kolaylaştırmaktadır.

3.2 Birden Fazla Not Defteri ile Çalışma

- **Yeni Not Defteri Açma:**

  Araç çubuğundaki + butonuna tıklanarak veya “File → New Notebook” seçeneği kullanılarak yeni not defterleri açılabilmektedir.

- **Yan Yana Yerleştirme:**

JupyterLab arayüzünde pencereler sürüklenerek farklı sekmeler halinde paralel olarak görüntülenebilmekte; böylece bir defterde yapılan işlemler diğer defterde referans alınabilmektedir.

3.3 Sunum Hazırlama

- **Markdown Hücreleri:**

  Başlıklar, listeler ve kalın yazı gibi biçimlendirmeler kullanılarak metin hücreleri oluşturulabilmekte; hücre tipi “Markdown” olarak değiştirilerek sunum zenginleştirilebilmektedir.

- **Slayt Oluşturma:**

  RISE gibi eklentiler yardımıyla kod, görsel ve metin içerikleri slayt veya alt slayt formatına dönüştürülmekte; böylece etkileşimli sunumlar hazırlanabilmektedir.

Not: Jupyter Notebooks, teknik sunum ve raporlamalarda canlı örneklerin gösterilmesini destekleyerek etkileşimli bir çalışma ortamı sunmaktadır.

3.4 Not Defteri Oturumlarını Kapatma

- **Bellek ve Kaynak Yönetimi:**Kullanılmayan not defteri oturumları, stop ikonuna tıklanarak kapatılabilmekte; kapatılan oturumlarda “no kernel” ifadesi gözlemlenmekte ve kaynak kullanımı optimize edilmektedir.

---

4. Jupyter Kernels

4.1 Tanım ve İşlev

Kernel, Jupyter Notebooks’ta kodların çalıştırılmasından sorumlu hesaplama motoru olarak görev görmekte; not defteri açıldığında varsayılan veya seçilen dilin kernel’i devreye girmektedir.

4.2 Kernel Türleri ve Seçimi

- **Desteklenen Diller:**

Jupyter, Python, R, Julia, Swift gibi pek çok dili desteklemekte; Skills Network ortamında bazı diller önceden yüklenmiş olarak sunulmaktadır.

- **Python Kernel:**

  Python kodlarının çalıştırılması için varsayılan kernel kullanılmakta; çıktılar, ilgili hücre altında görüntülenmektedir.

- **Kernel Değiştirme:**

  Sağ üst köşedeki kernel simgesine tıklanarak “Change Kernel” seçeneği ile mevcut oturumda kernel güncellenebilmekte.

Not: Her not defterinin kendi kernel sürecine sahip olması, farklı not defterlerinde farklı dillerin veya ortamların kullanılmasına olanak tanımaktadır.

4.3 Yerel Kurulumda Kernel Ekleme

- **CLI Kullanımı:**

  Ek dil desteği sağlamak amacıyla “pip install ipykernel” gibi komutlar kullanılarak kernel eklenebilmekte.

- **Anaconda ile Kernel Eklenmesi:**

  Anaconda ortamında `conda install -c conda-forge <kernel-ismi>` komutu ile yeni kernel’ler kolayca eklenebilmektedir.

---

5. Jupyter Architecture

5.1 İki Süreç Modeli (Kernel ve Client)

- **Kernel:**

  Kod yürütme işlemlerinden sorumlu arka plan süreci olarak, girdileri alıp çıktıları üreterek ekrana göndermektedir.

- **Client (Kullanıcı Arayüzü):**

  Tarayıcı üzerinden etkileşim kurulan bölüm olarak, kod hücrelerini kernel’e gönderip gelen sonuçları görüntülemektedir.

5.2 Notebook Sunucusu

- **.ipynb Dosya Formatı:**

  Jupyter Notebooks, içerikleri JSON formatında saklayan `.ipynb` dosyaları olarak kaydedilmekte; sunucu bu dosyaları okuyarak arayüzde görüntülemektedir.

- **Kaydetme ve Yükleme:**

  “Save” butonu ile yapılan kayıtlar anında dosyaya yansımakta; aynı dosya farklı makine veya platformlarda sorunsuzca açılabilmektedir.

Not: Notebook sunucusu, yerel makinede 8888 gibi bir port üzerinden veya bulut ortamında size özel URL aracılığıyla çalıştırılabilmektedir.

5.3 Dosya Formatı Dönüşümü (NB Convert)

- **NB Convert Aracı:**

  Not defteri dosyalarının HTML, PDF gibi farklı formatlara dönüştürülmesini sağlayarak paylaşım ve sunum işlemlerinde esneklik kazandırmaktadır.

- **Dönüşüm Süreci:**

  Preprocessor, exporter ve postprocessor adımları ile not defteri dosyalarının hedef formata uygun hale getirilmesi sağlanmaktadır.

- **Örnek Kullanım:**

  `jupyter nbconvert --to html my_notebook.ipynb` komutu kullanılarak not defteri HTML formatına dönüştürülebilmektedir.

---

6. Özet

Bu notlarda, **Jupyter Notebooks** ve **JupyterLab** uygulamalarının tanım, özellik, kurulum seçenekleri, çalışma pratikleri, kernel işleyişi ve mimarisi detaylı biçimde ele alınmaktadır.

- **Jupyter Notebooks,** interaktif kod yazma, analiz ve dokümantasyon süreçlerinin tek bir dosya içerisinde bütünleştirilmesiyle yeniden üretilebilir çalışmaların desteklenmesinde kritik rol oynamaktadır.

- **JupyterLab,** çoklu dosya ve not defteri yönetimi ile genişletilebilir çalışma ortamı sunarak, büyük ölçekli projelerde esnek ve entegre çözümler sağlamaktadır.
- **Kernel yapısı,** farklı programlama dillerinin desteklenmesi ve çalışma süreçlerinin izole edilmesi açısından önemli avantajlar sunmakta; yerel ve bulut ortamlarında kernel ekleme ve değiştirme seçenekleri desteklenmektedir.
- **Notebook sunucusu ve NB Convert aracı,** dosya formatı dönüşümü ve veri paylaşımında esnek çözümler sağlayarak projelerin dokümantasyon ve sunum süreçlerini desteklemektedir.

Her proje için Jupyter ekosisteminin doğru ve entegre kullanımı, interaktif analizlerin, dokümantasyonun ve sunumların başarılı bir şekilde gerçekleştirilmesinde temel bir etken olarak değerlendirilmektedir.