Büyük Veri Kalitesi Yönetimi:

Great Expectations ve Deequ ile Otomasyon ve Deneyimsel Analiz

Proje Raporu

Ağustos 2025

Contents

Chapter 1

Giriş

Kurumların karar destek sistemlerinde, güvenilir ve kaliteli veri temel bir gerekliliktir. Bu projede, yüksek hacimli çalışan verisi üzerinde iki modern veri kalitesi platformu olan **Great Expectations** ve **Deequ**'nun karşılaştırmalı bir şekilde uygulanması, sürecin otomasyonu, yaşanan zorluklar ve edinilen bireysel/kolektif kazanımlar detaylı olarak ele alınmaktadır.

Amaç, veri setinin profilini tam otomasyon ile çıkarırken iş kuralı kontrollerini ve teknik validasyonları iki farklı paradigmayla analiz etmektir. Sonuçlar, yalnızca teknik başarıya değil, aynı zamanda sürdürülebilirlik, ekip içi iş birliği, geliştirme becerileri ve mesleki vizyon gibi alanlara da katkı sağlamıştır.

Chapter 2

Proje Yaklaşımı ve Yöntemi

2.1 Çok Katmanlı Mimari Tasarım

Projede **modüler, Docker tabanlı** bir yapı tercih edildi. Her framework (GE/Deequ), kendi ortamında izole olarak çalıştırıldı. Veri erişimi doğrudan CSV dosyaları üzerinden yapılarak veri güncelliği ve ölçeklenebilirlik korundu. Çıktılar ve loglar ayrı klasörlerde sistematik biçimde toplandı.

2.2 Otomasyon, Sürdürülebilirlik ve Tekrarlanabilirlik

Tüm süreçler, kodda parametreleştirildi. Verinin yerel veya uzak olmasından bağımsız şekilde çalışabilecek, testten canlıya kolayca taşınabilen bir pipeline tasarlandı. Docker Compose ile tüm servisler tek komutla ayağa kaldırıldı. Frameworkler arası karşılaştırma, ortak veri ve konfigürasyon üzerinden sağlandı.

2.3 Çalışan Verisinin Profil Analizi ve Kontrol Matrisi

- Veri setindeki altı ana tablo (`employees`, `departments`, `salaries`, `titles`, `dept_emp`, `dept_manager`) otomatik tanındı.
- Kolon düzeyinde eksiklik, tekillik, tip uygunluğu ve iş kuralı kontrolleri tanımlandı.
- Tüm raporlar JSON ve HTML olarak, ayrıca özet tablo biçiminde oluşturuldu.
- Hata ve istisna yönetimi için detaylı loglama uygulandı.

Chapter 3

Geliştirme Süreci ve Yaşanan Problemler

3.1 Great Expectations Deneyimi

Başlangıçta, bazı expectation fonksiyonları framework'te doğrudan yer almadığı veya yanlış referanslandığı için "suite creation" aşaması başarısız oldu. Şu mesajla karşılaşıldı:

> *module 'great_ expectations.expectations' has no attribute 'ExpectColumnValuesToNotBeNull'*

Hataların kök nedeni, framework API'sinde fonksiyon isimlerinin ve importların hassas olmasıydı. Ayrıca, suite tanımlama sırasında mevcut expectation kütüphanesine bağlı kalınmalı ve string isimler doğru kullanılmalıydı.

Bu sorunun aşılmasıyla birlikte, her tablo için temel expectation'lar oluşturularak validasyon başlatıldı. Sonuçta, **toplamda 24 kontrolün 21'i geçti**, başarı oranı %83,3 oldu. `Data Docs` ile etkileşimli görsel raporlar kolayca incelenebildi.

3.2 Deequ Deneyimi

Deequ ile çalışırken ilk denemelerde aşağıdaki gibi hatalar alındı:

> *Invalid argument, not a string or column: [1, 1] of type <class 'list'>...*

Bunun temel nedeni, constraint fonksiyonlarına (örneğin `isComplete` ve `isUnique`) birden fazla kolon ismi içeren bir liste veya generator verilmesiydi. Deequ ve PySpark API'si bu fonksiyonların sadece string olarak tek bir kolon adı ile çağrılmasını gerektirir. Kodun, kolonları döngü ile gezip tek tek constraint ekleyecek şekilde revize edilmesiyle sorun giderildi.

Sonuçta, **tüm 11 kalite kontrolü eksiksiz geçti** ve sistem, 3,9 milyon satırlık veriyi hatasız şekilde analiz etti.

Chapter 4

Tecrübe Notları ve Kişisel Gelişim Üzerindeki Etkileri

4.1 Teknik ve Problem Çözme Becerilerinin Gelişimi

Projede karşılaşılan hatalar, yalnızca teknik bilgiyle değil, aynı zamanda sistematik problem çözme yaklaşımıyla yönetildi. Özellikle:

- Hataların kök neden analizini hızlıca yapabilmek,
- API dokümantasyonlarını etkin ve derinlemesine kullanmak,
- Dinamik kod üretiminde (örn. otomatik suite/constraint oluşturma) test ve doğrulama yapısının önemini kavramak,
- Modüler ve sürdürülebilir kod yazmanın uzun vadede ekip içi iş birliğine ve tekrarlanabilirliğe katkısını görmek,
- Geliştirme-işletme döngüsünde (DevOps) otomasyonun kritik değerini tecrübe etmek,

bu projenin bana kazandırdığı başlıca teknik tecrübelerdir.

4.2 Kurumsal Bakış Açısı ve Takım Çalışması Kazanımları

Projede konteynerleşme ve otomasyonun sağladığı kolaylık, takım çalışmasında "herkesin aynı ortamda" çalışabilmesinin, ortam uyuşmazlıklarının ve manuel bağımlılıkların ortadan kalkmasının ne kadar kıymetli olduğunu gösterdi. Ayrıca, hem teknik ekibe hem iş birimlerine hitap eden raporlar hazırlamak, farklı paydaşlar için çıktı üretmekte sunum ve iletişim becerilerimi artırdı.

4.3 Kişisel Vizyon ve Kariyer Etkisi

Büyük veri analitiği ve otomatik veri kalitesi süreçleri üzerinde çalışmak, ileride bulut tabanlı veri platformlarında ve veri mühendisliği alanında kendimi daha güvenli ve yetkin hissetmemi sağladı. Modern açık kaynak araçlarla çalışmak, ileri düzey Python ve Spark bilgisini birleştirmek hem teknik derinliğimi hem de çözüm üretme kapasitemi artırdı. Ayrıca, kodun sürdürülebilirliği, dokümantasyonu ve hata yönetimi gibi profesyonel pratikler konusundaki bakışımı geliştirdi.

Chapter 5

Sonuçlar ve Karşılaştırmalı Değerlendirme

5.1 Dosya Bazlı Sonuçların İncelenmesi

5.1.1 Great Expectations

- 6 tablo analiz edildi, 24 kontrol uygulandı, 21'i başarıyla geçti.
- Eksiklik ve tekillik kontrollerinde yüksek başarı; iş kurallarında (örn. tarih uyumsuzlukları) bazı zorluklar.
- Data Docs ile kullanıcı dostu, interaktif raporlar üretildi.

5.1.2 Deequ

- 6 tablo ve toplam 11 kontrol, tamamı başarıyla geçti.
- Spark altyapısı ile yüksek hacimli veride hızlı ve ölçeklenebilir analiz sağlandı.
- Raporlar daha teknik ve istatistik odaklı, iş kuralı açısından daha fazla manuel geliştirme gerekebilir.

5.1.3 Hata Yönetimi

- Her iki frameworkte hata anında detaylı log ve JSON hata raporu oluşturuldu.
- Hatalar kök nedenleriyle birlikte dokümante edilerek sürecin şeffaflığı sağlandı.

5.2 Frameworkler Arası Kapsamlı Karşılaştırma

Özellik	**Great Expectations**	**Deequ**

Kullanım Dili	Python	Scala / PySpark
Raporlama	HTML Data Docs, JSON	JSON, teknik istatistik
Otomasyon	Yüksek, esnek API	Yüksek, Spark'a entegre
Hata Mesajları	Açıklayıcı, kullanıcı dostu	Spark/PySpark'a özgü, daha teknik
Performans	Küçük-orta veri için hızlı	Büyük veri için yüksek performans
Business Rule	Kolay ve detaylı tanım	API bazında manuel tanım gerekebilir
Başarı Oranı	%83,3	%100

5.3 Projenin Sonuçları ve Geleceğe Yönelik Öneriler

- Büyük veri ve performans gerektiren işlerde Deequ, detaylı business rule ve kullanıcı dostu raporlama gerektiren işlerde Great Expectations öne çıkmaktadır.
- Hibrit kullanım (ör. veri hacmi için Deequ, rapor ve iş kuralı yönetimi için GE) kurumsal veri platformlarında en yüksek katma değeri sağlar.
- Tüm veri kalitesi süreçlerinin otomasyonu ve Docker tabanlı dağıtımı, sürdürülebilir ve tekrarlanabilir projeler için en iyi yaklaşımdır.
- Hatalar, yalnızca teknik detay olarak değil, öğrenme ve gelişim fırsatı olarak değerlendirilmelidir.

Bibliography

[1] Great Expectations Documentation.
https://docs.greatexpectations.io/

[2] Deequ Documentation.
https://github.com/awslabs/deequ

[3] CTU Relational Database.
https://relational.fel.cvut.cz/dataset/Employee