Bank Customer Analysis*

*Note: Machine Learning and Data Analysis on Banking Customer Data

https://github.com/guneskahil/BankCustomerAnalysis

Güneş Kahiloğulları
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Kocaeli, Türkiye
211307015@kocaeli.edu.tr

Abstract*— Bu proje, gerçek zamanlı veri işleme ve analiz için büyük veri teknolojilerinin, özellikle Apache Kafka ve Apache Spark'ın uygulanmasını incelemektedir. Projenin ana odak noktası, bir bankacılık veri seti kullanarak sınıflandırma görevlerinin gerçekleştirilmesi ve makine öğrenimi modellerinin doğruluğunun değerlendirilmesidir. Random Forest ve Lojistik Regresyon algoritmaları, müşteri verilerini sınıflandırmak için uygulanmış ve GridSearchCV kullanılarak model performansı optimize edilmiştir. Veri seti, yaş, iş, medeni durum ve finansal bilgiler gibi çeşitli müşteri özelliklerini içermekte olup, bu veriler özellik mühendisliği, normalizasyon ve eksik verilerin işlenmesi ile ön işleme tabi tutulmuştur. Gerçek zamanlı metrikler, modellerin doğruluğu dahil, Kafka üzerinden iletilerek canlı izleme ve karar verme süreçleri sağlanmıştır. Büyük veri analitiği ile gerçek zamanlı veri akışının entegrasyonu, değerli içgörüler sunarak karar alma süreçlerinin verimliliğini artırmaktadır. Bu proje, makine öğrenimi ile büyük veri framework'lerinin birleştirilmesinin dinamik, gerçek zamanlı bir ortamda tahminsel modellerin optimizasyonu için sunduğu potansiyeli göstermektedir.

Keywords— Büyük Veri, Makine Öğrenimi, Apache Kafka, Apache Spark

I. GIRIŞ (*INTRODUCTION*)

Günümüzde veri analitiği, yapay zeka ve büyük veri teknolojileri birçok sektörde kritik bir rol oynamaktadır. Özellikle, verilerin doğru bir şekilde işlenmesi, analiz edilmesi ve anlamlı bilgiler haline getirilmesi, karar alma süreçlerinde büyük bir avantaj sağlamaktadır. Bu proje kapsamında, bir bankacılık veri seti üzerinde sınıflandırma ve analiz işlemleri gerçekleştirilmiş; bunun yanı sıra Apache Kafka ve Apache Spark gibi büyük veri işleme teknolojilerinden yararlanılarak gerçek zamanlı veri akışı ve analiz yetenekleri geliştirilmiştir.

Projenin temel amacı, büyük veri teknolojilerini kullanarak sınıflandırma modellerinin doğruluk oranlarını ölçmek, sonuçları optimize etmek ve bu sonuçları gerçek zamanlı bir ortamda paylaşmaktır. Bu bağlamda, Python programlama dili ile Random Forest ve Logistic Regression gibi makine öğrenimi algoritmaları uygulanmış, model performansları karşılaştırılmış ve en iyi sonuçlar GridSearchCV kullanılarak optimize edilmiştir. Ayrıca, Apache Kafka aracılığıyla model doğruluk oranlarının gerçek zamanlı olarak bir mesajlaşma sistemi üzerinden iletilmesi sağlanmıştır.

Proje, büyük veri analitiği ve gerçek zamanlı veri akışı teknolojilerinin entegrasyonuna odaklanmaktadır. Kullanılan veri setindeki eksik verilerin işlenmesi, kategorik değişkenlerin kodlanması, uç değer analizi ve veri ölçeklendirme gibi adımlarla model oluşturma süreçleri desteklenmiştir. Çıktılar, projeye ilişkin karar mekanizmalarının daha etkili ve hızlı bir şekilde işletilmesine olanak sağlamaktadır.

Bu raporda, projenin kapsamı, kullanılan yöntemler, analiz sonuçları ve elde edilen bulgular detaylı bir şekilde ele alınacaktır. Gerçekleştirilen çalışmanın büyük veri analitiği alanındaki uygulamalara katkı sağlaması ve veri odaklı teknolojilere yönelik yeni bir bakış açısı sunması hedeflenmektedir.

II. VERI SETI VE ÖZELLIKLERI

Bu çalışma, bank.csv adlı bir veri seti üzerinde gerçekleştirilen bir analiz üzerine odaklanmaktadır. Veri seti, 4.521 kayıttan oluşmakta olup, her bir satırda çeşitli müşteri özelliklerini ve bu müşterilerin finansal durumlarına dair bilgileri içermektedir. Ancak, ilk bakışta veri seti, tüm verileri tek bir sütun altında noktalı virgül (;) ile ayrılmış bir şekilde saklamaktadır. Bu sütun, farklı müşteri özelliklerini temsil eden çeşitli kategorik ve sayısal verilere sahiptir. Veri setindeki sütun başlıkları aşağıdaki gibi sıralanabilir:

- age: Müşterinin yaşı
- job: Müşterinin mesleği
- marital: Müşterinin medeni durumu
- education: Müşterinin eğitim durumu
- default: Müşterinin kredi borcu olup olmadığı (varsayılan)
- balance: Müşterinin hesap bakiyesi
- housing: Müşterinin ev kredisi olup olmadığı
- loan: Müşterinin diğer kredilerinin olup olmadığı
- contact: İletişim türü
- day: Son arama yapılan gün

- month: Son arama yapılan ay
- duration: Son arama süresi
- campaign: Son kampanyadaki iletişim sayısı
- pdays: Son aramadan bu yana geçen gün sayısı
- previous: Önceki kampanyada iletişim sayısı
- poutcome: Önceki kampanyanın sonucu
- y: Müşterinin başarılı bir şekilde abonelik alıp almadığı (hedef değişken)

A. Veri Yükleme ve Hazırlama

İlk olarak, veri seti tek bir sütun olarak yüklenmiş ve bu sütundaki veriler, noktalı virgül (;) karakterine göre ayrılacak şekilde işlenmiştir. Bu işlem sonrası veri seti doğru sütunlar ile yapılandırılmıştır. Veri setinin başlangıçta tüm sütunlar tek bir sütun olarak göründüğünden, bu verilerin her birinin uygun kategorilere ayrılması için ön işleme yapılması gerekmektedir.

B. Veri Seti Özellikleri

- Kayıt Sayısı: 4.521
- Sütun Sayısı: Veri setinde toplamda 17 sütun bulunmaktadır.
- Eksik V
- eriler: Veri setinde eksik veriler bulunmamaktadır.
- Veri Türleri: Veri setinde sayısal (örneğin, age, balance, duration) ve kategorik (örneğin, job, education, marital) sütunlar mevcuttur.

C. İyileştirme ve Dönüşüm Süreci

Veri seti işlenmeden önce şu iyileştirmeler yapılacaktır:

1. Veri Ayrıştırma: İlk olarak, sütunlar arasında noktalı virgül ile ayrılmış veriler doğru şekilde bölünecek ve her bir özellik kendi sütununda yer alacaktır.
2. Veri Tipi Dönüşümleri: Kategorik veriler (örneğin, job, education) uygun şekilde sayısal verilere dönüştürülecek veya kodlanacaktır.
3. Eksik Veri Kontrolü: Veri seti üzerinde eksik veri olup olmadığı kontrol edilip gerekirse uygun değerlerle doldurulacaktır.
4. Veri Normalizasyonu: Sayısal verilerin daha iyi bir şekilde modellenebilmesi için uygun normalizasyon teknikleri uygulanacaktır (örneğin, Min-Max Scaling veya Standartizasyon).
5. Özellik Mühendisliği: Eğitim ve test verilerinin daha iyi ayrılabilmesi için özellik mühendisliği teknikleri kullanılacaktır.

Bu adımların sonunda, veri seti modelleme ve analiz için hazırlanmış olacak ve sınıflandırma algoritmalarına uygulanacaktır.

III. KULLANILAN TEKNOLOJILER VE ARAÇLAR

Bu projede, büyük veri işleme ve gerçek zamanlı veri akışı analizi için bir dizi güçlü teknoloji ve araç kullanılmıştır. Bu araçlar, verinin etkili bir şekilde işlenmesi, dağıtılması ve analiz edilmesi için temel bir altyapı sağlamaktadır. Kullanılan başlıca teknolojiler Apache Spark, Apache Kafka ve Apache Zookeeper'dir.

- Apache Spark: Apache Spark, büyük veri işleme ve analiz için açık kaynaklı bir platformdur. Spark, paralel işlemeyi destekleyen in-memory (bellek içi) işleme yetenekleri ile büyük veri kümeleri üzerinde çok daha hızlı analizler yapılmasına olanak sağlar. Bu proje kapsamında, Apache Spark, verilerin hızlı bir şekilde işlenmesi ve sınıflandırma modelinin eğitilmesi amacıyla kullanılmıştır. Spark, özellikle büyük veri analitiği ve makine öğrenimi uygulamalarında yüksek verimlilik sağlar. PySpark, Spark'ın Python API'sidir ve bu projede verinin işlenmesi için kullanılmıştır.

- Apache Kafka: Apache Kafka, yüksek verimli ve ölçeklenebilir bir veri akış platformudur. Gerçek zamanlı veri iletimi ve işlenmesi için kullanılan Kafka, özellikle log verileri, işlem akışları ve zaman serisi verileri gibi sürekli akış halindeki verilere yönelik güçlü bir çözüm sunmaktadır. Bu projede, Kafka Producer ve Kafka Consumer kullanılarak, makine öğrenimi modelinin doğruluk oranı gibi metrikler gerçek zamanlı olarak bir Kafka topic'i üzerinden gönderilmekte ve tüketilmektedir.

- Apache Zookeeper: Apache Zookeeper, dağıtık sistemlerdeki koordinasyonu yönetmek için kullanılan açık kaynaklı bir araçtır. Zookeeper, genellikle büyük ölçekli, dağıtık uygulamalarda, konfigürasyon yönetimi, dağıtık senkronizasyon ve grup yönetimi gibi görevleri kolaylaştırmak için kullanılır. Bu projede, Kafka'nın dağıtık ortamda yönetilmesinde Zookeeper kullanılmıştır. Kafka, Zookeeper'a bağlı olarak çalışan bir sistemdir ve Zookeeper, Kafka'nın doğru çalışmasını sağlamak için broker'lar arasındaki iletişimi ve koordinasyonu yönetir.

IV. VERI ANALIZI VE MODELLEME

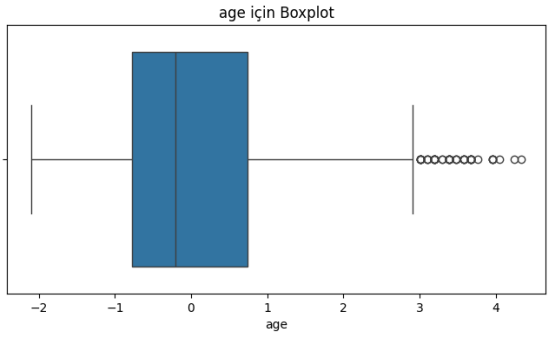
A. Veri Seti Ön İşleme

1. Veriyi Yükleme ve İnceleme: Çalışmanın ilk adımı, veri setini uygun bir kütüphane ile sisteme yüklemek ve veriyi incelemektir. Bu aşamada veri setinin genel yapısı analiz edilmiş, sütun isimleri ve veri türleri incelenmiştir. Ayrıca, verinin yapısal eksiklikleri ve potansiyel sorunları belirlemek amacıyla ilk birkaç satır görüntülenmiş, sütunlardaki veri tipleri ve kayıt sayıları detaylı olarak değerlendirilmiştir.

2. Eksik Verilerin İncelenmesi ve Tamamlanması: Eksik veri problemleri, veri analizi ve model geliştirme süreçlerinde büyük bir zorluk teşkil edebilir. Bu nedenle eksik veriler, sütun bazında tespit edilmiş ve uygun istatistiksel yöntemler kullanılarak doldurulmuştur. Örneğin, sayısal veriler için sütunun ortalama veya medyan değeri kullanılmış, kategorik verilerde ise mod veya en yaygın değer tercih edilmiştir.

3. Veri Normalizasyonu: Verilerin farklı ölçeklerde olması, model performansını olumsuz etkileyebilir. Bu problemi çözmek için sayısal veriler belirlenmiş ve standartlaştırma işlemi gerçekleştirilmiştir. Bu süreçte, tüm değişkenler aynı ölçekte normalize edilerek analiz ve modelleme süreçleri için uygun hale getirilmiştir.

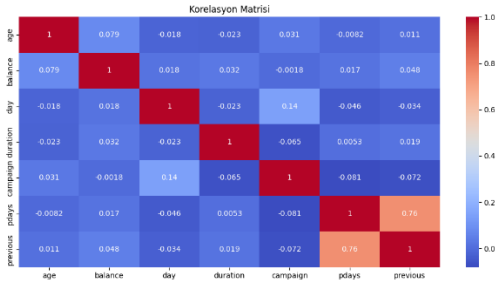
4. Uç Değer Tespiti ve İşleme: Uç değerler, veri dağılımını bozabilir ve yanlış sonuçlara yol açabilir. Uç değerleri belirlemek için görselleştirme teknikleri (örneğin, kutu grafikleri) ve istatistiksel yöntemler (örneğin, Z-score analizi) kullanılmıştır. Uç değerlerin analizi sonucunda, belirli eşik değerlerinin dışındaki veriler veri setinden çıkarılmış veya düzeltilmiştir.



Şekil 1. Age kategorisine ait boxplot görseli

B. *Veri Görselleştirme*

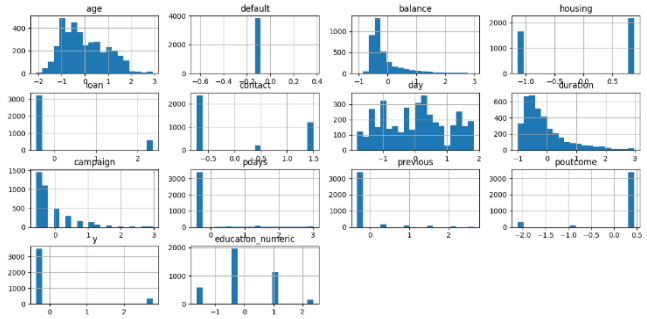
1. Korelasyon Matrisi: Veri setindeki değişkenler arasındaki ilişkileri anlamak için korelasyon matrisi oluşturulmuştur. Bu görselleştirme, değişkenler arasındaki pozitif veya negatif ilişkileri sayısal değerlerle ifade etmenin yanı sıra, görsel bir harita üzerinden sunmuştur. Korelasyon katsayısı 1'e yaklaştıkça güçlü bir ilişki, 0'a yaklaştıkça zayıf bir ilişki olduğu görülmüştür.



Şekil 2. Korelasyon Isı Haritası. Verisetindeki değişkenler arasındaki korelasyon katsayılarının görselleştirilmesi. Renk yoğunluğu korelasyonun pozitif veya negatif yönünü ve gücünü ifade eder

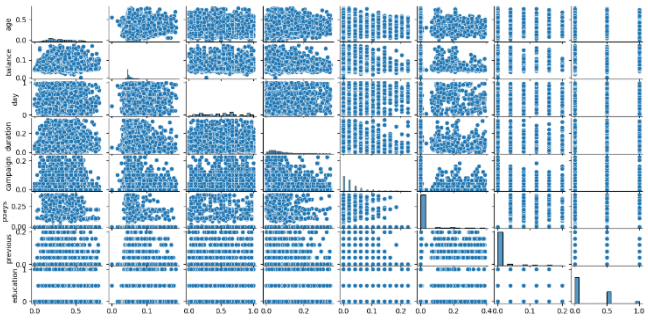
Veri setindeki değişkenler arasındaki korelasyon katsayılarının görselleştirildiği ısı haritası yukarıda gösterilmiştir. Renk yoğunluğu, korelasyon katsayısının büyüklüğünü ve yönünü göstermektedir. Pozitif korelasyonlar kırmızı tonlarla, negatif korelasyonlar ise mavi tonlarla ifade edilmiştir. "Previous" ve "Pdays" değişkenleri arasında güçlü bir pozitif ilişki gözlemlenirken, diğer değişken çiftlerinde korelasyon oldukça zayıftır.

2. Dağılım ve Yoğunluk Analizleri: Veri setindeki sayısal sütunların dağılımlarını analiz etmek için histogramlar oluşturulmuş ve bu dağılımlar grafikler üzerinde yorumlanmıştır. Ayrıca, değişkenler arasındaki potansiyel doğrusal ilişkileri incelemek için dağılım (scatter) grafikleri kullanılmıştır.



Şekil 3. Değişkenlerin Dağılımı (Histogramlar)

Veri setindeki farklı değişkenlerin dağılımlarını gösteren histogramlar. Her bir grafik, ilgili değişkenin değerlerinin sıklığını ve dağılımını temsil eder. Bazı değişkenler, belirli bir değere yoğunlaşmışken (örneğin, "campaign" ve "previous" değişkenlerinde olduğu gibi), diğerleri daha geniş bir aralıkta yayılmaktadır. Verilerin çoğu belirli aralıklarla yoğunlaşmış olup, bu da veri setinin özelliklerine dair önemli bilgiler sunmaktadır.



Şekil 4. Değişkenler Arası İlişkiler (Pairplot)

Veri setindeki değişkenler arasındaki ilişkilerin görselleştirilmiş hali. Her bir nokta, iki değişken arasındaki bireysel ilişkiyi gösterir. Bazı değişkenler arasında doğrusal ilişkiler görülürken, bazı değişkenler arasında belirgin bir ilişki yoktur. Örneğin, "balance" ve "loan" değişkenleri arasındaki ilişkiyi ve "previous" ile "pdays" gibi diğer ikili ilişkileri gözlemleyebilirsiniz. Ayrıca, her bir değişkenin kendi dağılımı ve potansiyel anormallikler veya uç değerler de histogramlar şeklinde sunulmuştur.

C. Yapay Zeka Modeli Geliştirme

1. Model Seçimi ve Eğitimi: Veri seti, bağımlı ve bağımsız değişkenlere ayrılmış ve bu değişkenler kullanılarak bir sınıflandırma modeli geliştirilmiştir. Model olarak lojistik regresyon seçilmiş ve veriler eğitim ve test olarak ikiye ayrılmıştır. Eğitim seti kullanılarak model eğitilmiş, test seti ile performansı değerlendirilmiştir.

   Bu çalışmada, sınıflandırma problemi çözmek için üç farklı makine öğrenmesi modeli kullanılmıştır: Random Forest, Logistic Regression ve Decision Tree. Her bir modelin doğruluk oranı, sınıflandırma raporu ve hata matrisleri detaylı bir şekilde incelenmiş ve karşılaştırılmıştır.

1. Random Forest Modeli

Random Forest modeli de %100 doğruluk oranı ile sonuçlanmıştır. Confusion matrix'e göre, model 162 doğru pozitif (TP), 1155 doğru negatif (TN), 0 yanlış pozitif (FP) ve 0 yanlış negatif (FN) tahmin yapmıştır. Modelin precision, recall ve f1-score metrikleri aşağıdaki gibidir:

Precision: 1.00 (her iki sınıf için)
Recall: 1.00 (her iki sınıf için)
F1-Score: 1.00 (her iki sınıf için)

2. Logistic Regression Modeli

Logistic Regression modeli %90,96 doğruluk oranı ile sonuçlanmıştır. Confusion matrix'e göre, model 23 doğru pozitif (TP), 782 doğru negatif (TN), 15 yanlış pozitif (FP) ve 65 yanlış negatif (FN) tahmin yapmıştır. Modelin precision, recall ve f1-score metrikleri aşağıdaki gibidir:

Precision: 0.61 (sınıf 0 için), 0.61 (sınıf 1 için)
Recall: 0.98 (sınıf 0 için), 0.26 (sınıf 1 için)
F1-Score: 0.75 (sınıf 0 için), 0.36 (sınıf 1 için) Decision Tree Modeli

3. Decision Tree Modeli

Decision Tree modeli %100 doğruluk oranı elde etmiştir. Confusion matrix'e göre, model 162 doğru pozitif (TP), 1155 doğru negatif (TN), 0 yanlış pozitif (FP) ve 0 yanlış negatif (FN) tahmin yapmıştır. Modelin precision, recall ve f1-score metrikleri aşağıdaki gibidir:

Precision: 1.00 (her iki sınıf için)
Recall: 1.00 (her iki sınıf için)
F1-Score: 1.00 (her iki sınıf için)

4. Model Karşılaştırması

Modellerin karşılaştırılması sonucunda, Decision Tree ve Random Forest modellerinin mükemmel doğruluk oranı (%100) gösterdiği ve bu iki modelin en başarılı modeller olduğu görülmüştür. Logistic Regression modeli ise biraz daha düşük doğrulukla (%90,96) sonuçlanmış ancak sınıf 1 (pozitif) üzerinde precision ve recall değerleri daha düşük kalmıştır.
Logistic Regression iyi bir doğruluk oranına sahip olsa da, özellikle sınıf 1 üzerinde düşük recall ve precision değerleri göstermektedir.
Decision Tree ve Random Forest modelleri ise her iki sınıf için de mükemmel sonuçlar elde etmiş ve yanlış pozitif veya negatif üretmemiştir.

5. İyileştirme Önerileri

Model performansını artırmak için aşağıdaki iyileştirme önerileri sunulabilir:

- Veri Dengelemesi: Logistic Regression modelinin sınıf 1 (pozitif) üzerindeki performansını iyileştirmek için SMOTE gibi tekniklerle veri setinde dengeleme yapılabilir.
- Hiperparametre Optimizasyonu: Decision Tree ve Random Forest modelleri, hiperparametre ayarlamaları ile daha da iyileştirilebilir. Ancak, bu modeller halihazırda mükemmel performans göstermektedir.
- Özellik Mühendisliği: Logistic Regression modelinin performansını artırmak için daha fazla özellik eklenebilir veya mevcut özellikler dönüştürülebilir.

D. Kafka Yapılandırması ve Veri Üretimi/ Tüketimi

Bu projede, verilerin gerçek zamanlı olarak işlenebilmesi ve dağıtılabilmesi amacıyla Apache Kafka kullanılmıştır. Kafka, yüksek hacimli verilerin hızlı ve güvenilir bir şekilde iletilmesini sağlayan bir mesajlaşma sistemidir. Bu adımda, Kafka'nın Producer ve Consumer yapılandırmaları gerçekleştirilmiştir. Verisetindeki işlenmiş veriler önce Producer ile bir Kafka Topic'ine yüklenmiş, ardından Consumer ile bu veriler okunmuştur.

1. Kafka Producer Yapılandırması

İlk olarak, Kafka'ya veri gönderebilmek için bir Producer uygulaması oluşturulmuştur. Producer, veriyi bir veya daha fazla Kafka topic'ine gönderme işlevini üstlenir. Bu adımda, "newtopic" adında bir topic oluşturulmuş ve işlenmiş verisetindeki veriler bu topic'e aktarılmıştır. Kafka Producer, verilerin belirli bir formatta (örneğin JSON veya CSV) düzenli bir şekilde Kafka'ya iletilmesini sağlar.

Producer Yapılandırması:

- Topic Adı: "newtopic"
- Veri Formatı: İşlenmiş veri, Kafka producer tarafından uygun bir formatta (örneğin JSON veya CSV) hazırlanarak Kafka'ya gönderilmiştir.
- Producer Bağlantısı: Producer, Kafka cluster'ına bağlanabilmek için gerekli yapılandırmalar (Kafka broker adresi, port numarası vb.) yapılmıştır.
- Veri Gönderimi: İşlenmiş verisetindeki her bir veri satırı, Kafka producer kullanılarak "newtopic" topic'ine gönderilmiştir. Bu işlem sırasında, her bir veri satırının bir mesaj olarak Kafka'ya iletilmesi sağlanmıştır.

2. Kafka Consumer Yapılandırması

Kafka Consumer, Kafka'ya gönderilen verilerin okunmasını ve işlenmesini sağlayan bir bileşendir. Producer tarafından "newtopic" topic'ine gönderilen

veriler, Kafka Consumer ile okunmuştur. Consumer, Kafka topic'inden verileri alır ve işleme sürecine dahil eder.

Consumer Yapılandırması:

- Topic Adı: "newtopic"
- Veri Formatı: Kafka'dan alınan veriler, uygun şekilde işlenebilmek için aynı formatta (örneğin JSON) çözülmüştür.
- Consumer Bağlantısı: Kafka consumer, Kafka cluster'ına bağlanarak ilgili topic'ten verileri almaktadır.
- Veri Okuma: Consumer, "newtopic" topic'inden verileri okumuş ve alınan veriler üzerinde işlem yapmıştır.

3. Kafka Producer ve Consumer Arasındaki Veri Akışı

Kafka Producer ve Consumer arasındaki veri akışı şu şekilde gerçekleşmiştir:

- Producer, işlenmiş veriyi "newtopic" topic'ine gönderir. Her bir veri, mesaj olarak gönderilir ve Kafka cluster'ına iletilir.
- Kafka Consumer, "newtopic" topic'inden gelen mesajları alır. Consumer, bu mesajları çözümler ve istenilen işlemleri yapar.

Bu işlem, verilerin farklı sistemler arasında hızlı ve güvenilir bir şekilde iletilmesini sağlayarak, gerçek zamanlı veri işleme süreçlerinin temelini oluşturur.

4. Sonuçlar ve Değerlendirme

Kafka'nın bu yapılandırması, büyük veri işleme ve gerçek zamanlı veri akışını yönetme konusunda etkin bir çözüm sunmaktadır. Producer tarafından gönderilen veriler, Kafka Consumer tarafından hızlı bir şekilde okunmuş ve işlenmiştir. Bu yapı, yüksek hacimli verilerin dağıtık bir şekilde işlenmesini ve verilerin farklı sistemler arasında kolayca paylaşılmasını sağlamaktadır.

Kafka'nın sağladığı bu özellikler, özellikle büyük veri projelerinde ve gerçek zamanlı veri analizlerinde büyük avantajlar sunmaktadır. Bu adım, sistemin ölçeklenebilirliğini ve veri iletiminin güvenilirliğini artıran önemli bir adımdır.

E. Kafka ve Spark Streaming ile Anomali Tespiti

Projenin son adımında, Kafka ve Spark Streaming kullanılarak verisetinde anomalilerin tespiti yapılmıştır. Bu adımda, age >= 35 ve balance >= 1000 koşullarını sağlayan veri satırlarında anomali tespiti gerçekleştirilmiş ve bu anomaliler JSON formatında bir dosyaya yazdırılmıştır.

1. Spark Streaming ile Kafka'dan Veri Okuma

Önceki adımlarda veriler Kafka'ya yüklenmiş ve Kafka Consumer tarafından okunmuştu. Bu adımda ise Apache Spark Streaming kullanılarak Kafka'dan gelen veriler gerçek zamanlı olarak işlenmiştir. Spark Streaming, verilerin sürekli bir akış halinde işlenmesini sağlar ve büyük veri kümeleri üzerinde hızlı analizler yapılmasına olanak tanır.

Kafka ve Spark Streaming yapılandırması:

- Kafka Topic: "newtopic"
- Veri Formatı: JSON formatındaki veriler Kafka'dan alınarak Spark tarafından işlenmiştir.
- Stream İşleme: Veriler, belirli aralıklarla (mini-batch) işlenmiş ve belirlenen koşullara göre anomaliler tespit edilmiştir.

2. Anomali Tespiti ve Sonuçlar

Bu adımda, **age >= 35** ve **balance >= 1000** koşullarını sağlayan verilerdeki anomali tespit edilmiştir. Tespit edilen anomaliler, sistem tarafından gerçek zamanlı olarak işlenmiş ve **JSON formatında** dışa aktarılmıştır. Bu işlem, verisetindeki belirli özellikler üzerinden anomali tespiti yapmak için kullanılır.

{"balance":1811,"age":42}
{"balance":11971,"age":38}

Şeklinde JSON formatında yazdırılmıştır. Bu JSON formatındaki veriler, tespit edilen anomalilerin doğru bir şekilde saklanmasını ve gerektiğinde analiz edilmesini sağlar.

3. Sonuçlar ve Değerlendirme

Kafka ve Spark Streaming kullanılarak gerçekleştirilen bu adım, gerçek zamanlı veri işleme ve anomali tespiti için etkili bir çözüm sunmuştur. Kafka, verilerin hızlı bir şekilde iletilmesini sağlarken, Spark Streaming de bu verileri yüksek verimlilikle işlemeye imkan tanımaktadır. Anomali tespiti işlemi, verinin belirli koşullara göre filtrelenmesi ve anormal durumların doğru bir şekilde tanımlanması ile gerçekleştirilmiştir.

Bu yapı, büyük veri projelerinde veri akışının sürekli izlenmesini ve anormal durumların hızlıca tespit edilmesini sağlar. Ayrıca, elde edilen anomali verileri JSON formatında saklanarak, daha sonraki analizler ve raporlama işlemleri için kolayca kullanılabilir hale getirilmiştir.

Bu adım, projede veri akışının yönetilmesi ve önemli anomali durumlarının tespit edilmesi açısından kritik bir rol oynamıştır.