Soru1
Spark yapılandırması yapılmış ve local[*] modunda çalışan bir SparkContext oluşturulmuştur. Data.csv dosyası okunup , ilk satır başlık olarak kabul edilip, bu satır veri kümesinden çıkarılmıştır.

```
 // Belirtilen parametreyi arama ve konsola "alert" yazısı
bastırma
        String searchParam = "IndiGo"; // Aramak istediğiniz
parametre
        dataLines.filter(line -> line.contains(searchParam))
                .foreach(line -> System.out.println("alert: "
+ line));
```

Soru2

Bu soruda training_data.py , app.py ve PredictionApp.java sınıfları oluşturulmuştur.
Dataset : Raisin_Dataset seçilmiştir.

Python kullanılarak Kmeans algoritması ile veri kümeleme işlemi gerçekleştirilmiştir.(3 küme olarak belirlendi) Kümeleme sonuçları, veri setine eklenmiş ve clustered_dataset.csv olarak kaydedilmiştir. Eğitimli model ve özellik isimleri pickle formatında saklanmıştır (decision_tree_model.pkl ve feature_names.pkl).

ModelTrainer Sınıfı

Bu sınıf, veri kümeleme ve karar ağacı modelinin eğitim işlemlerini içerir. Kodun amacı, veri kümesindeki özellikleri kullanarak KMeans ile kümeleme yapmak ve ardından bu verilerle bir karar ağacı modeli eğitmektir.

App.Py Sınıfı

Bu sınıf, Flask ve Flask-CORS kullanarak bir RESTful servis oluşturur. Servis, eğitilmiş bir model kullanarak tahminler yapar ve gelen taleplere yanıt verir.

def load_model_and_features(self): Bu fonksiyon, pickle formatında kaydedilmiş modeli ve özellik isimlerini dosyadan yükler.

def predict(self, features): Bu fonksiyon, verilen özellikler listesi (features) ile modelden tahmin yapar. Özellikler bir DataFrame'e dönüştürülür ve sütun isimleri modelin beklediği özellik isimleriyle eşleştirilir. Model kullanılarak tahmin yapılır ve sonucu döndürülür.

```
@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json(force=True)
    features = data['features']  # Expecting 'features' to be a list of feature values
    prediction = service.predict(features)
    return jsonify({'prediction': prediction})
```

/predict endpoint'i tanımlanır. Bu endpoint, POST isteklerini alır. İstekten JSON formatında veriler alınır ve features listesi olarak çıkarılır. PredictionService kullanılarak tahmin yapılır ve sonuç JSON formatında yanıt olarak döndürülür.

Post Başarılı : 127.0.0.1 - - [01/Aug/2024 16:37:52] "POST /predict HTTP/1.1" 200 -

App.java sınıfı, uygulama başlatıldığında model ve özellik isimlerini dosyalardan yükler. /predict endpoint'ine bir POST isteği gönderildiğinde, istekten gelen özellikler kullanılarak model tahmin yapar. Daha sonra json formatında döndürür.

Bu ödev boyunca çok fazla 500 hatasıyla karşılaştım en uğraştığım kısım buydu. Veri değişikliği yapıldı karakterler düzgün işlenmiyordu. Aldığım hatalardan biri de 13 özellik isteniyor 14 özellik gönderdiniz hatasıydı. (Error during prediction: Expected 13 features, but got 14) Daha sonra Error during prediction: 2030 columns passed, passed data had 10 columns hataları aldım.

PredictionApp.java

Veri dosyasını (Raisin_Dataset.csv) okur, her bir veri satırı için özellikleri çıkarır ve Python tabanlı REST servisinden tahmin alır. Tahmin "Kecimen" ise, alarm oluşturur.
Verilen veri satırı virgüllere bölünür ve ilk 7 sütunu Double list olarak döner. Bu sütunlar, modelin kullanacağı özellikler. Daha sonra feature'lar json formatına çevrilir.
http://localhost:5000/predict adresine POST isteği gönderir. (pyhton tabanlı rest servisine) ve tahmin alınır. Gelen yanıtı JSON olarak işler ve tahmin sonucunu döner. Eğer tahmin "Kecimen" ise, alert oluşturulur ve bu veri satırı konsola yazdırılır.

Çıktı : (ilk 3 ü kopyalanmıştır)

Alarm! Record:
87524,442.2460114,253.291155,0.819738392,90546,0.758650579,1184.04,Kecimen
Alarm! Record:
75166,406.690687,243.0324363,0.801805234,78789,0.68412957,1121.786,Kecimen
Alarm! Record:
90856,442.2670483,266.3283177,0.798353619,93717,0.637612812,1208.575,Kecimen