Light Gradient Boosted Machine (LightGBM) Algoritması

Makine öğrenimi dünyasında, verimli ve hızlı çalışan algoritmalar, büyük veri setleriyle başa çıkmada kritik bir rol oynar. Özellikle sıralama, sınıflandırma ve regresyon gibi görevlerde yüksek performans gösteren modeller tercih edilir. **LightGBM** (Light Gradient Boosting Machine), bu tür zorlukların üstesinden gelmek için geliştirilmiş, hızlı ve etkili bir makine öğrenimi algoritmasıdır.

LightGBM nedir?

LightGBM, Denetimli bir **topluluk** makine öğrenimi algoritmasıdır. XGBoost veya Gradient Boosting algoritmasının yaptığı gibi çalışır, ancak bazı gelişmiş ve benzersiz özelliklere sahiptir. Light GBM, sıralama, sınıflandırma ve diğer birçok makine öğrenimi görevi için kullanılan, karar ağacı algoritmasına dayalı hızlı, dağıtılmış, yüksek performanslı bir gradyan artırma çerçevesidir.

LightGBM neden popüler?

Verilerin boyutu her geçen gün hızla arttığı için geleneksel algoritmaların hızlı sonuç vermesi zorlaştı. LightGBM, hesaplama gücü ve daha hızlı sonuç vermesi nedeniyle "**Light**" olarak adlandırılır. **Çalıştırmak için daha az bellek** gerektirir ve **büyük miktarda veriyle başa çıkabilir**. Hackathon'larda en yaygın kullanılan algoritma, çünkü algoritmanın amacı sonuçların iyi bir şekilde doğruluğunu elde etmek ve aynı zamanda GPU eğilimini desteklemektir.

LightGBM ne zaman kullanılır?

LightGBM, küçük hacimli veri kümeleri için değildir. Hassasiyeti nedeniyle küçük verileri kolayca sığdırabilir. 10.000+ satırdan fazla olan veriler için kullanılabilir. LightGBM kullanımına karar vermede yardımcı olan sabit bir eşik yoktur. Özellikle yüksek bir doğruluk elde edilmesi gerektiğinde büyük hacimli veriler için kullanılabilir.

LightGBM algoritmasının temel özellikleri

LightGBM'yi benzersiz güçlendirme algoritmalarından biri yapan temel özelliklerinden bazıları şunlardır:

- Eksik değerlerle otomatik olarak ilgilenir - bu, eksik değerleri işlemek için herhangi bir ön işleme adımı yapmamız gerekmediği anlamına gelir.
- Bölme noktaları oluşturmak için bölme için histogram tabanlı bir algoritma kullanır - bölme noktaları, bir karar ağacı düğümünde hangi verilerin bölündüğüne bağlı olarak özellik değerleridir.
- Örnekleri gradyanlara göre alt örnekleyen yeni bir örnekleme yolu olan Gradyan Tabanlı Tek Taraflı Örnekleme (GOSS) kullanır.
- LightGBM'deki karar ağaçları, yaprak bazında ağaç büyümesidir. Yaprak bazında ağaç büyümesi ile seviye bazında ağaç büyümesi arasındaki fark, yaprak bazında stratejinin, verileri en yüksek kayıp değişikliğine sahip düğümlere bölerek ağacı büyütmesi ve seviye bazında stratejinin ağacı seviye seviye büyütmesidir.
- LightGBM'nin bir diğer özelliği de, etkili özelliklerin sayısını azaltmak için neredeyse kayıpsız bir yöntem olan Özel Özellik Paketi Tekniğini kullanmasıdır.
- Daha hızlı eğitim hızı ve daha yüksek verimlilik. Daha düşük bellek kullanımı ve daha iyi doğruluk.

LightGBM neden bu kadar hızlı?

LightGBM'nin hızlı olmasının üç nedeni vardır:

- **Histogram tabanlı bölme:** LightGBM, ağaç yapılarını oluştururken histogram tabanlı bir bölme yöntemi kullanır. Bu yöntemde, özelliklerin sürekli değerleri belirli aralıklara (bin) bölünerek histogramlar oluşturulur. Bu işlem, potansiyel bölünme noktalarının sayısını önemli ölçüde azaltır ve bölme işlemlerini hızlandırır. Histogram tabanlı bölme, bellek kullanımını optimize ederken, hesaplama süresini de azaltır.
- **Gradyan Tabanlı Tek Taraflı Örnekleme (GOSS) :** Bu yöntem, gradyan boosting işlemi sırasında kullanılan örneklerin sayısını azaltarak hesaplama süresini kısaltır. GOSS, en büyük gradyan değerlerine sahip örnekleri korur ve geri kalan örneklerden rastgele bir alt küme seçer. Böylece, veri setinin tamamını kullanmadan modelin doğruluğunu koruyarak hesaplamaları hızlandırır. Bu yöntem, özellikle dengesiz veri setleriyle çalışırken etkilidir.
- **Özel Özellik Paketleme (EFB):** Özellik Paketleme, yüksek boyutlu veri setleriyle çalışırken özellikler arasındaki bağımsızlığı kullanarak bellek ve zaman verimliliğini artırır. EFB, nadir olarak bir arada görülen kategorik özellikleri tek bir grupta toplar ve bunları aynı pakette işler. Bu sayede, özellik sayısını azaltarak bellek kullanımını ve işlem süresini optimize eder. Özellikle yüksek boyutlu ve seyrek veri setlerinde bu yöntem büyük bir avantaj sağlar.

LightGBM'in Avantajları

- **Hızlı Eğitim**: Büyük veri setleri üzerinde hızlı eğitim süresi.
- **Düşük Bellek Kullanımı**: Verimli bellek yönetimi.
- **Yüksek Doğruluk**: Leaf-wise tree growth, modelin doğruluğunu artırabilir.
- **Overfitting Azaltma**: Yüksek derinlikteki ağaçlar nedeniyle oluşabilecek overfitting'i azaltmak için çeşitli parametrelerle kontrol edilebilir.

LightGBM Nerelerde Kullanılır?

LightGBM, hız ve performans gerektiren çeşitli makine öğrenimi görevlerinde kullanılır. Sıkça kullanıldığı alanlar şunlardır:

- **Sıralama Problemleri**: Arama motorları ve öneri sistemlerinde.
- **Sınıflandırma**: Finansal risk analizi, sağlık, ve sahtekarlık tespiti gibi alanlarda.
- **Regresyon**: Fiyat tahmini ve pazar analizi gibi görevlerde.
- **Büyük Veri Setleri**: Büyük ve yüksek boyutlu veri setlerinde hızlı ve etkili sonuçlar elde etmek için.

LightGBM Kullanımı

LightGBM, Python ve R gibi birçok programlama diliyle uyumludur. Python'da LightGBM'i kullanmak için önce lightgbm kütüphanesini kurmanız gerekir:

```
pip install lightgbm
```

Aşağıda, LightGBM ile temel bir modelin nasıl eğitileceğini gösteren bir Python örneği bulunmaktadır:

```
#Gerekli kütüphaneler import edilir.
import lightgbm as lgb
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
# Veri setini okuma
data = pd.read_csv('dataset.csv')
#Hedef değişkeni veriden ayırmak
X = data.drop('target', axis=1)
y = data['target']
# Veriyi eğitim ve test setlerine ayırma
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# LightGBM dataset formatına dönüştürme
train_data = lgb.Dataset(X_train, label=y_train)
test_data = lgb.Dataset(X_test, label=y_test)
# Modelin parametreleri
params = {
  'objective': 'regression',
  'metric': 'rmse',
  'boosting_type': 'gbdt',
  'learning_rate': 0.1,
  'num_leaves': 31,
  'max_depth': -1,
  'feature_fraction': 0.9,
  'bagging_fraction': 0.8,
```

```
    'bagging_freq': 5,

    'verbose': 0

}

# Modeli eğitme

model = lgb.train(params, train_data, valid_sets=[test_data], early_stopping_rounds=100)

# Tahmin yapma

y_pred = model.predict(X_test, num_iteration=model.best_iteration)

#Sonucu görme

mse = mean_squared_error(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
```

LightGBM'in Önemli Parametreleri

- **objective**: Modelin amacını belirler (örneğin, regresyon, sınıflandırma).
- **metric**: Değerlendirme metriğini belirtir (örneğin, RMSE, logloss).
- **boosting_type**: Kullanılan boosting türü (GBDT, DART, GOSS gibi).
- **learning_rate**: Her adımda modelin ne kadar öğrenmesini gerektiğini belirler.
- **num_leaves**: Bir ağacın sahip olabileceği maksimum yaprak sayısı.
- **max_depth**: Bir ağacın maksimum derinliği.
- **feature_fraction**: Her iterasyonda seçilen rastgele özelliklerin oranı.
- **bagging_fraction**: Her iterasyonda kullanılan rastgele veri alt kümesinin oranı.
- **bagging_freq**: Bagging'in kaç iterasyonda bir yapılacağını belirler.
- **early_stopping_rounds**: Eğitim sırasında performans iyileşmediğinde durmayı sağlar.

LightGBM'de Overfitting'i Azaltma Yolları

- **num_leaves** ve **max_depth** parametrelerini sınırlamak: Çok fazla yaprak veya çok derin ağaçlar overfitting'e neden olabilir.
- **feature_fraction** ve **bagging_fraction** kullanmak: Her iterasyonda modelin sadece belirli bir kısmını kullanarak overfitting'i azaltabilir.
- **lambda_l1** ve **lambda_l2** parametreleri ile regularization eklemek.

Kaynakça

- What is LightGBM Algorithm, How to use it? | Analytics Steps
- What is LightGBM used for? – KnowledgeBurrow.com
- LightGBM: Boosting Machine Learning Performance | by Mert Şükrü Pehlivan | Medium
- LightGBM in Machine Learning | Aman Kharwal (thecleverprogrammer.com)
- LightGBM algorithm: Supervised Machine Learning in Python (hands-on.cloud)
- ChatGPT