Modern E-Ticaret Öneri Sistemleri Mimarisi | Furkan Aksu

Konuşmacı: Furkan Aksu (Machine Learning Engineer)

Moderatörler: Enes Fehmi Manan, Selin Çıldam

Öne Çıkan Maddeler

I. Temel Mimari: "Two-Tower" Yaklaşımı

1. **Problem Tanımı:** Milyonlarca kullanıcı ve milyonlarca ürün arasında doğru eşleşmeyi milisaniyeler içinde bulmak.
2. **Two-Tower Mimarisi:** Sistemin kalbi; **User Tower** (Kullanıcı Kulesi) ve **Item Tower** (Ürün Kulesi) olmak üzere iki ayrı sinir ağından oluşur.
3. **Amaç:** Kullanıcı ve ürünü ortak bir matematiksel düzlemde (Vektör Uzayında) buluşturmak.
4. **Embeddings:** Her kullanıcı ve ürün için oluşturulan, onların karakteristik özelliklerini taşıyan sayısal kimlik kartları (vektörler).
5. **User Tower Girdileri:** Kullanıcının geçmiş tıklamaları, demografik bilgileri, cihaz bilgisi vb.
6. **Item Tower Girdileri:** Ürünün kategorisi, markası, metin özellikleri, resmi vb.
7. **Independent Learning:** İki kule birbirinden bağımsız çalışır; bu sayede ürün vektörleri önceden hesaplanıp saklanabilir (Hız kazandırır).
8. **Dot Product (Nokta Çarpımı):** Kullanıcı vektörü ile ürün vektörünün çarpılması; sonuç ne kadar yüksekse eşleşme (Similarity) o kadar güçlüdür.
9. **Cosine Similarity Farkı:** Cosine yerine Dot Product tercih edilir çünkü popüler ürünlerin (vektör büyüklüğü fazla olanların) avantajını korumak istenir.

II. Model Eğitimi ve Hedefleme (Target Engineering)

10. **Target (Hedef) Belirleme:** Model neyi öğrenecek? Kullanıcının tıklama zinciri (A -> B -> C) bir ilişkidir.
11. **Positive Samples (Pozitif Örnekler):** Kullanıcının gerçekten tıkladığı ürünler (Gerçek veri).
12. **Negative Samples (Negatif Örnekler):** Kullanıcının tıklamadığı, alakasız olduğu varsayılan rastgele seçilmiş ürünler.
13. **Loss Function (Kayıp Fonksiyonu):** Modelin hatasını ölçen matematiksel formül (Binary Cross Entropy). Amaç hatayı minimize etmektir.
14. **Derin Öğrenme Katmanları:** Kulelerin içinde genellikle standart Neural Network katmanları kullanılır, bazen Transformer yapıları da eklenebilir.
15. **Eğitim Döngüsü:** Model genellikle haftalık periyotlarla (Weekly Training) yeniden eğitilir.

III. Huni Stratejisi (The Funnel Strategy)

16. **Huni Mantığı:** Milyonlarca üründen en iyi 10 ürüne inmek için kullanılan kademeli eleme yöntemi.
17. **Aşama 1: Retrieval (Aday Belirleme):** Amaç hızdır. Two-Tower modeliyle milyonlardan -> binlerce adaya inilir. Kaba elemedir.
18. **Aşama 2: Ranking (Sıralama):** Amaç hassasiyettir. Seçilen 1000 aday, daha karmaşık modellerle (XGBoost, LightGBM vb.) puanlanır.
19. **Ranking Modelleri:** Genellikle **Pairwise** (İkili Karşılaştırma) yöntemleri kullanılır (Bu ürün mü daha iyi, şu mu?).
20. **Aşama 3: Re-Ranking (Yeniden Sıralama):** Modelin çıktısına son dokunuşların yapıldığı yer.
21. **Business Rules (İş Kuralları):** Re-Ranking aşamasında uygulanır (Stok kontrolü, yasaklı ürünler, kampanya öncelikleri).
22. **Çeşitlilik (Diversity):** Peş peşe aynı kategoriden ürün göstermemek için araya farklı ürünler serpiştirilir.

IV. Teknik Altyapı ve Teknoloji Yığını (Tech Stack)

23. **Veri Ambarı (Data Warehouse): BigQuery**. Milyonlarca satır verinin (30-40 milyon kullanıcı) tutulduğu yer.
24. **Model Geliştirme: Python** dili, **PyTorch** (Deep Learning için) ve **Scikit-learn** kütüphaneleri.
25. **Orkestrasyon: Apache Airflow**. Veri akışlarının, model eğitimlerinin zamanlanması ve otomatik yönetimi.
26. **Konteynerizasyon: Docker**. Uygulamanın her ortamda aynı çalışması için paketlenmesi.
27. **Ölçekleme (Scaling): Kubernetes**. Canlı sistemdeki trafiğe göre sunucu sayısını otomatik artırıp azaltan yapı.
28. **Hızlı Servis (Serving): Redis** (NoSQL). Canlıda öneriyi milisaniyeler içinde sunmak için kullanılan önbellek.
29. **Vektör Veritabanı (Vector DB):** Ürün embeddinglerini saklamak ve hızlıca aramak için kullanılan özel veritabanları.
30. **LLM Kısıtı:** Büyük Dil Modelleri (ChatGPT, LLaMA vb.) canlı öneri sistemleri için maliyetli ve yavaştır; bu yüzden tercih edilmez.
31. **Open Source Embedding Modelleri:** Metin/Resim işlemek için hazır eğitilmiş açık kaynak modeller kullanılır (örn: multilingual-e5).

V. Zorluklar ve Çözümler (Challenges)

32. **Cold Start (Soğuk Başlangıç):** Verisi olmayan yeni kullanıcı veya ürün sorunu.
33. **Yeni Kullanıcı Çözümü:** Popüler ürünleri önermek veya onboarding (ilgi alanı seçtirme) ekranları.
34. **Yeni Ürün Çözümü:** Ürünün resminden veya isminden içerik bazlı (Content-based) benzerlik kurmak.
35. **Sparsity (Veri Seyrekliği):** Matrisin %99'unun boş olması (Kullanıcı az ürünle etkileşime girer). Two-Tower bunu iyi yönetir.
36. **Feedback Loop (Yankı Odası):** Model popüleri önerir -> Kullanıcı tıklar -> Ürün daha popüler olur. Yeni ürünler arada kaynar.
37. **Exploration (Keşif):** Yankı odasını kırmak için bazen risk alıp yeni/az tıklanan ürünleri kullanıcıya göstermek.
38. **Misafir (Guest) Kullanıcılar:** Üye olmayanlar için Çerezler (Cookies) üzerinden takip yapılır ve geçici ID ile öneri sunulur.

VI. Performans Ölçümü ve Testler

39. **Offline Metrics (Laboratuvar Testi):** Modeli canlıya almadan önce geçmiş veriyle yapılan testler.
40. **Time-Series Split:** Eğitim ve test verisi zamana göre ayrılır (Geçmişle eğit, geleceği tahmin et). Asla rastgele bölünmez.
41. **Precision@K:** Önerilen listenin doğruluğu (Ne kadar isabetli?).
42. **Recall@K:** Kullanıcının sevdiği ürünleri kapsama oranı (Kaçırdığımız var mı?).
43. **NDCG:** Sıralamanın kalitesi (Doğru ürün 1. sırada mı, 10. sırada mı?).
44. **Online Metrics (Canlı Test):** Modelin gerçek dünyadaki başarısı.
45. **A/B Testing:** Kullanıcıları iki gruba ayırıp (Eski Model vs Yeni Model) yarıştırmak.
46. **CTR (Click Through Rate):** Tıklama oranı. En temel online başarı metriği.
47. **Conversion Rate (CR):** Satın alma oranı. Tıklanan ürünün satışa dönme başarısı.
48. **Confidence Interval:** A/B testi sonuçlarının şans eseri olup olmadığını anlamak için istatistiksel güven aralığına bakılır.
49. **Basitlik İlkesi:** Bazen basit bir Lojistik Regresyon, karmaşık bir Deep Learning modelinden daha iyi çalışabilir; test etmeden bilemeyiz.
50. **Sürekli İyileştirme:** Öneri sistemi yaşayan bir organizmadır; veriler, modeller ve kurallar sürekli güncellenmelidir.

Kendime Notlar :

"Öneri Sistemleri (Recommendation Systems) Mimarisi" özeti:

---

1. Ana Mimari: "Two-Tower" (İki Kule) Modeli 

Sistemin kalbinde **Two-Tower** mimarisi

- Elimizde iki grup var: **Kullanıcılar (Sol Kule)** ve **Ürünler (Sağ Kule)**.
- Amacımız bunları bir "Vektör Uzayında" (Matematiksel bir oda) buluşturmak.
- **Sol Kule (User Tower):** Kullanıcının geçmişini, yaşını, tıkladığı şeyleri alıp ona bir **Kimlik Kartı (Embedding)** çıkarıyor.
- **Sağ Kule (Item Tower):** Ürünün resmini, kategorisini, adını alıp ona da bir **Kimlik Kartı** çıkarıyor.
- **Dot Product (Çarpım):** Bu iki kimlik kartını çarpıştırıyoruz. Eğer matematiksel sonuç yüksekse "Bu kullanıcı bu ürünü sever" diyoruz.

Eğitim Mantığı (Training):

- **Pozitif Örnek:** Ahmet "iPhone Kılıfı"na tıkladı. (Sistem bunu öğrensin: Ahmet <-> Kılıf uyumlu).
- **Negatif Örnek:** Ahmet "Bebek Bezi"ne tıklamadı. (Sistem rastgele ürünler seçip "Bak bunu sevmiyor" diye öğretiyor).

---

2. Süreç: Huni Stratejisi (Retrieval -> Ranking -> Re-Ranking) 🔽

Konuşmacı, milyonlarca üründen son listeye nasıl indiklerini anlatıyor.

- **Adım 1: Aday Belirleme (Retrieval / Candidate Generation):**
    - **Durum:** 30 Milyon ürün var. Hepsini tek tek hesaplayamayız.
    - **İşlem:** Two-Tower modelini kullanarak "Ahmet'e en yakın **1000 ürünü** getir" diyoruz. Hızlı ama kaba bir eleme. (Vektör veritabanı kullanılıyor).
    - **Analoji:** Garsonun mahzene inip, müşterinin sevebileceği **tüm** şarapları (1000 şişe) yukarı taşıması.
- **Adım 2: Sıralama (Ranking):**
    - **Durum:** Elimizde 1000 aday var ama müşteriye 10 tane göstereceğiz.
    - **İşlem:** Daha hassas modellerle (Learning to Rank) bu 1000 ürünü puanlıyoruz. "Ahmet buna %90 tıklar, buna %10 tıklar" diye sıraya diziyoruz.
    - **Analoji:** 1000 şişeyi tek tek inceleyip en iyi 20 tanesini seçmesi.
- **Adım 3: İş Kuralları (Re-Ranking / Business Rules):**
    - **Durum:** Model en iyi ürünleri seçti ama...
    - **İşlem:** Şirket kuralları devreye girer:
        - "Stokta olmayanları listeden at."
        - "Peş peşe 5 tane buzdolabı gösterme, araya çeşit at."
        - "Yeni gelen kampanyalı ürünü 3. sıraya sıkıştır."

---

3. Kullanılan Teknolojiler (Tech Stack) 🛠️

Konuşmacı, bu "Fabrika"nın hangi makinelerle döndüğünü anlatıyor:

1. **BigQuery: (Depo)** Milyonlarca satır veri (tıklamalar, ürünler) burada tutuluyor.
2. **Python & PyTorch: (AR-GE Laboratuvarı)** Modellerin (beyinlerin) kodlandığı ve eğitildiği yer.
3. **Airflow: (Vardiya Amiri)** "Her Pazartesi modeli yeniden eğit", "Her gece veriyi temizle" diyen zamanlayıcı.
4. **NoSQL / Vector DB (Redis vb.): (Sıcak Servis Bankosu)** Müşteri siteye girdiğinde öneriyi milisaniyeler içinde sunmak için kullanılan süper hızlı hafıza.
5. **Docker & Kubernetes: (Lojistik)** Sistemin her sunucuda aynı çalışmasını sağlayan konteyner yapısı.

---

4. Karşılaşılan Zorluklar (Challenges) 🚧

- **Cold Start (Soğuk Başlangıç):**
    - *Sorun:* Yeni kullanıcı veya yeni ürün geldiğinde verisi yok.
    - *Çözüm:* Ürünlerin sadece ID'sine değil, **resmine ve ismine** bakıyoruz (Content-based). Resimden ne olduğunu anlayıp benzerlerine yönlendiriyoruz.
- **Feedback Loops (Yankı Odası):**
    - *Sorun:* Model hep popüler olanı öneriyor -> Kullanıcı popüleri tıklıyor -> Model "Bak bildim" diyip daha çok popüler öneriyor. Yeni ürünler arada kaynıyor.
    - *Çözüm:* Yeni ürünlere manuel "puan desteği" (boost) vererek onları yukarı taşıyoruz.

---

5. Performans Ölçümü: A/B Testleri 🧪

"Modelim iyi çalışıyor mu?"

- **Offline Test (Lab):** Geçmiş veriye bakarak (Recall, Precision) "Eğer bunu önerseydim tutar mıydı?" analizi.
- **Online Test (A/B Testi - Gerçek Hayat):**
    - **Analoji:** Müşterilerin yarısına **Eski Menüyü**, diğer yarısına **Yeni Menüyü** veriyoruz.
    - Hangi grup daha çok sipariş verirse (CTR - Tıklama Oranı, Conversion - Satın Alma), o model kazanır.

---

6. Soru-Cevap Kısmından Önemli Notlar

- **LLM (Büyük Dil Modelleri) Kullanımı:** "LLM'leri (ChatGPT gibi) öneri sisteminde kullanmayı düşündük mü?" sorusuna; "Çok maliyetli ve yavaş. Milyonlarca istekte çok para yazar, o yüzden şu an sadece embedding (vektör) çıkarmak için open-source modeller kullanıyoruz" cevabı verildi.
- **Veri Güncelliği:** Model her hafta yeniden eğitiliyor (Retraining). Ama kullanıcı anlık bir şeye tıkladığında, o anlık veri (Real-time) hemen sisteme dahil ediliyor.

Bir e-ticaret sitesi, **milyonlarca ürünü** alıp, o an siteye giren müşteriye **en uygun 10 ürünü** 1 saniyenin altında göstermek için **'İki Kule'** dediğimiz bir sistem kurmuş.

1. Önce kaba eleme yapıyorlar (Retrieval).
2. Sonra ince işçilikle puanlıyorlar (Ranking).
3. En son patronun kurallarını (Stok, Kampanya) uyguluyorlar.

Ve bunu yaparken **'Yeni gelen müşteriyi küstürmemek' (Cold Start)** ve **'Sürekli aynı şeyleri önermemek'** için özel ayarlar yapıyorlar."