Topluluk Öğrenimi ve Random Forest

Zayıf Tahmincilerden Güçlü Kararlara: Kitlelerin Bilgeliği



NotebookLM

Tek bir uzmana güvenmek yerine, çoğunluğun fikrini almak neden daha güvenilirdir?



Arkadaşına Sormak

- Yüksek Yanlılık Riski (Bias)





Beta Sürümü Yayınlamak

- Kitlelerin Bilgeliği (Wisdom of Crowd)



Ensemble modelleri, birden fazla **'zayıf öğrenicinin'** kararlarını birleştirerek **daha yüksek tahmin gücüne** ve **kararlılığa** sahip **'güçlü bir öğrenici'** oluşturur.

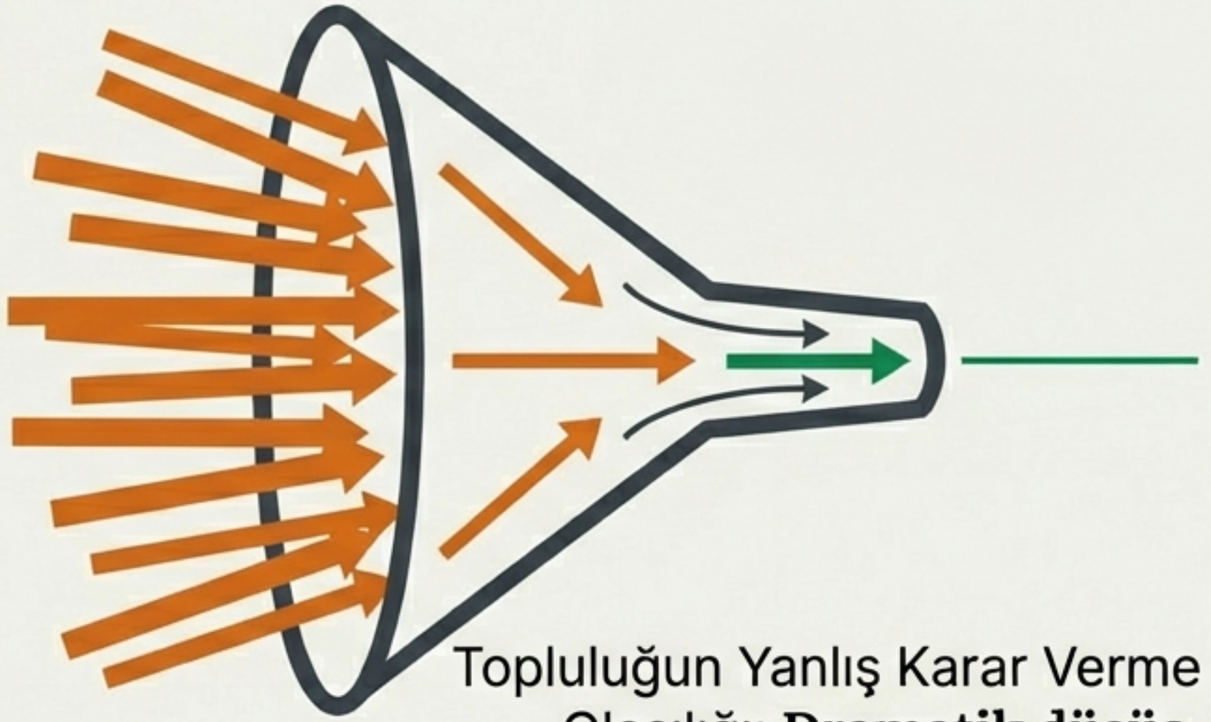
`Anahtar Kavram: Kolektif Zeka ve Güvenilirlik Artışı`

NotebookLM

Bağımsız tahmincilerin hataları birbirini telafi eder.

Varsayımlar:

- Sınıflandırıcı Sayısı (N): 25
- Bireysel Hata Oranı (ε): 0.35
- Koşul: **Tahminciler birbirinden bağımsızdır.**



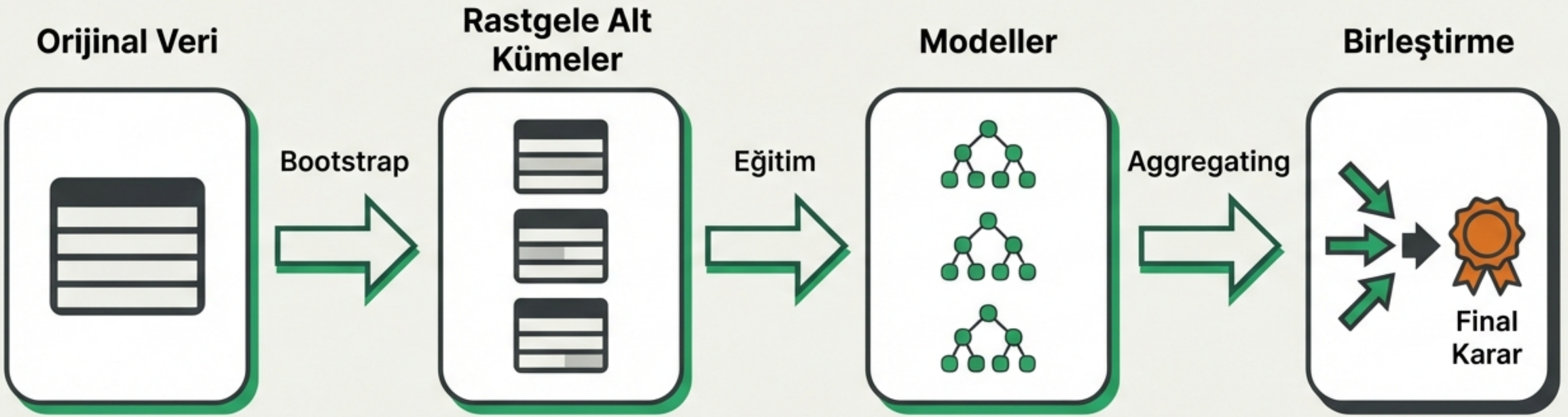
Topluluğun Yanlış Karar Verme Olasılığı: **Dramatik düşüş.**

Çoğunluk oylaması (Majority Voting), bireysel hata oranları yüksek olsa bile doğru sonucu bulma şansını artırır.

NotebookLM

Editorial Technical Manual

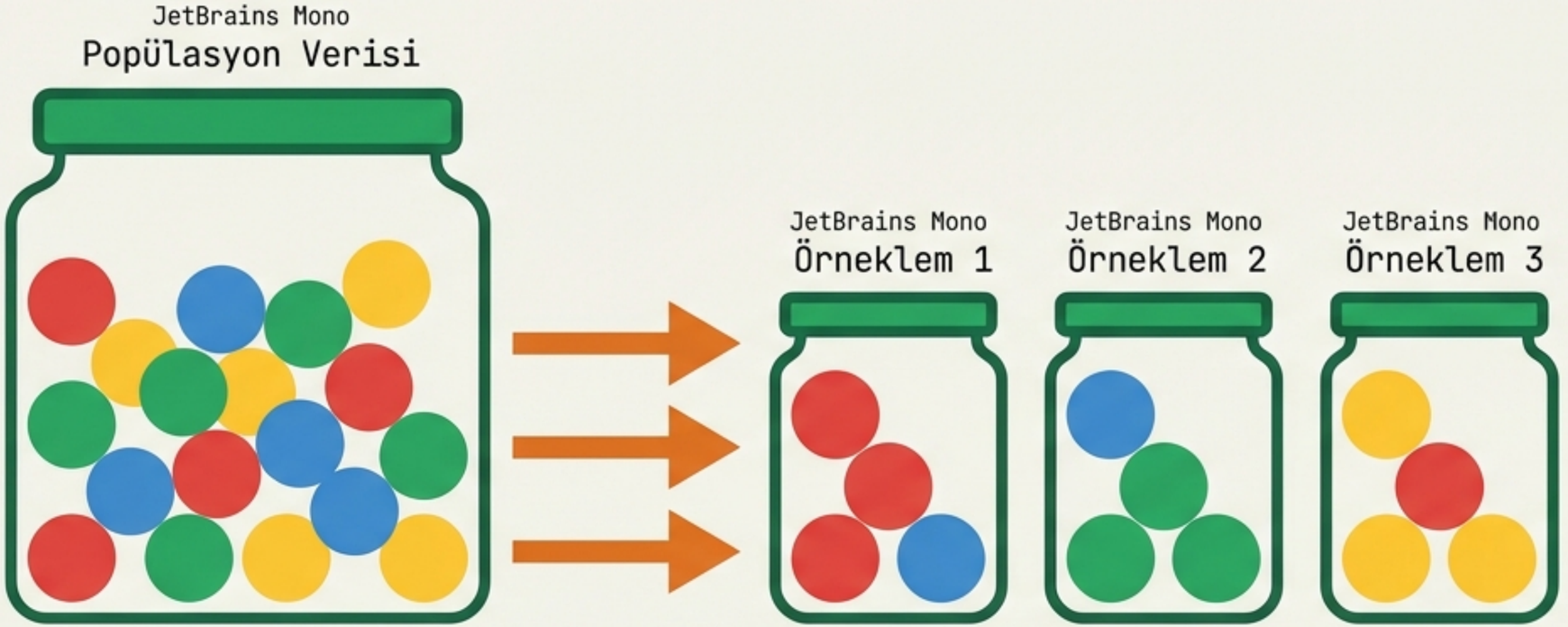
Mekanizma: Bagging (Bootstrap Aggregating)



Bagging, yüksek varyansa sahip modelleri (örneğin Karar Ağaçları) iyileştirmek için kullanılan bir yöntemdir.

NotebookLM

Bootstrap: Tek bir veri setinden sanal çeşitlilik yaratmak.



Yeni veri toplamak pahalıdır. Bootstrap, mevcut veriden istatistiksel olarak farklı ama temsili temsili alt kümeler oluşturarak bu sorunu çözer.

NotebookLM

Ortalamayı almak, modelin varyansını (kararsızlığını) düşürür.

Karar Ağaçları ve Yüksek Varyans

Karar ağaçları aşırı öğrenmeye (overfitting) meyillidir. Verideki küçük bir değişiklik, tamamen farklı bir ağaç yapısı üretebilir.



Tekil Ağaç Kararsızlığı

Bagging Etkisi

Farklı veri setlerinde eğitilen ağaçların tahminlerinin ortalamasını almak varyansı düşürür.

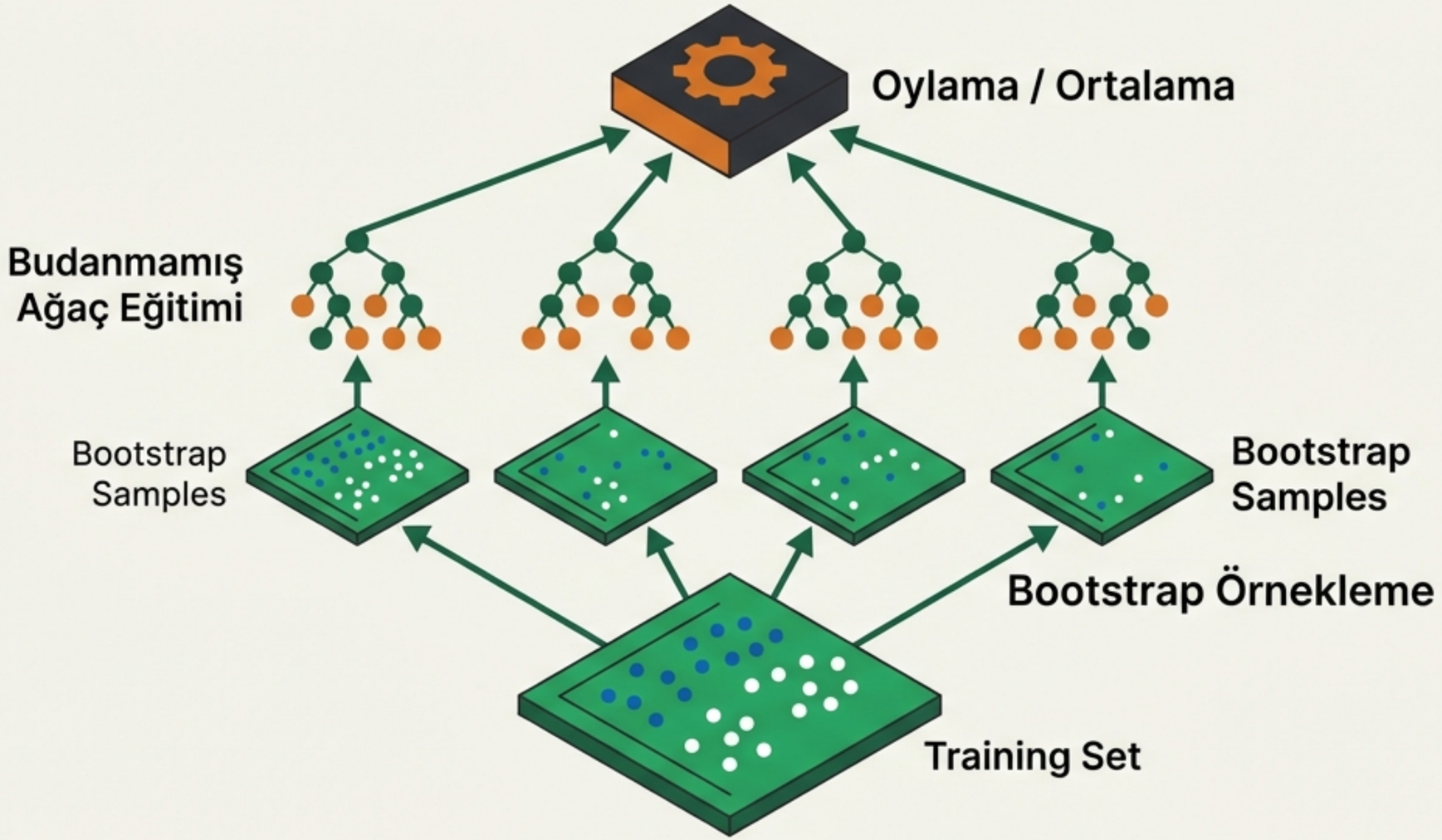


Ortalama Tahmin

$$f_{bag}(x) = \frac{1}{B} * \sum f_b(x)$$

NotebookLM

Bagged Trees: Aynı algoritma, farklı veri, ortak karar.



1. Bootstrap Örnekleme 2. Budanmamış Ağaç Eğitimi 3. Oylama / Ortalama

NotebookLM

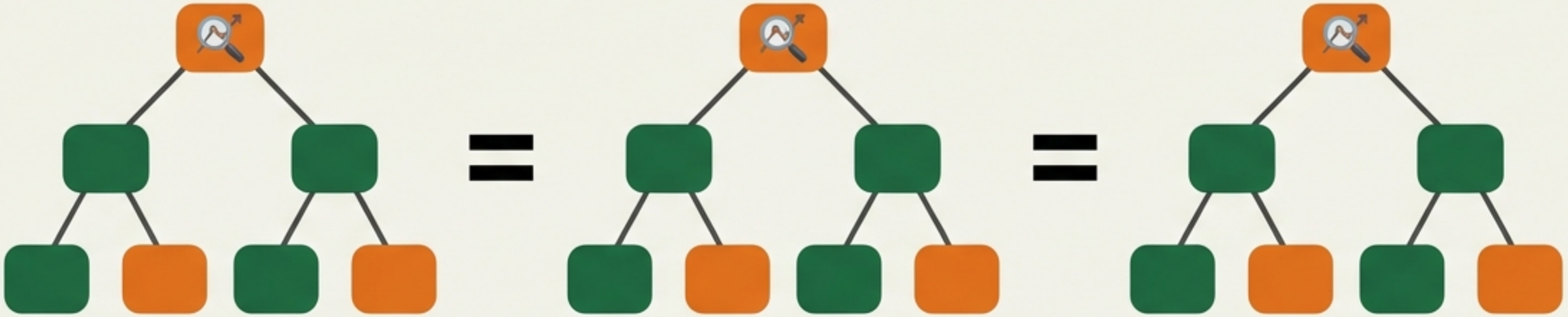
Peki, Bagging her zaman yeterli mi? (Korelasyon Sorunu)

Durum: Baskın Özellik (Dominant Feature)

Veri setinde çok güçlü bir tahmin edici (feature) varsa, tüm ağaçlar ilk bölünmede (root node) bu özelliği kullanır.

Sonuç: Yüksek Korelasyon

Üretilen ağaçlar birbirinin kopyası gibi olur. Birbirinin aynısı olan ağaçların ortalamasını almak varyansı düşürmez.

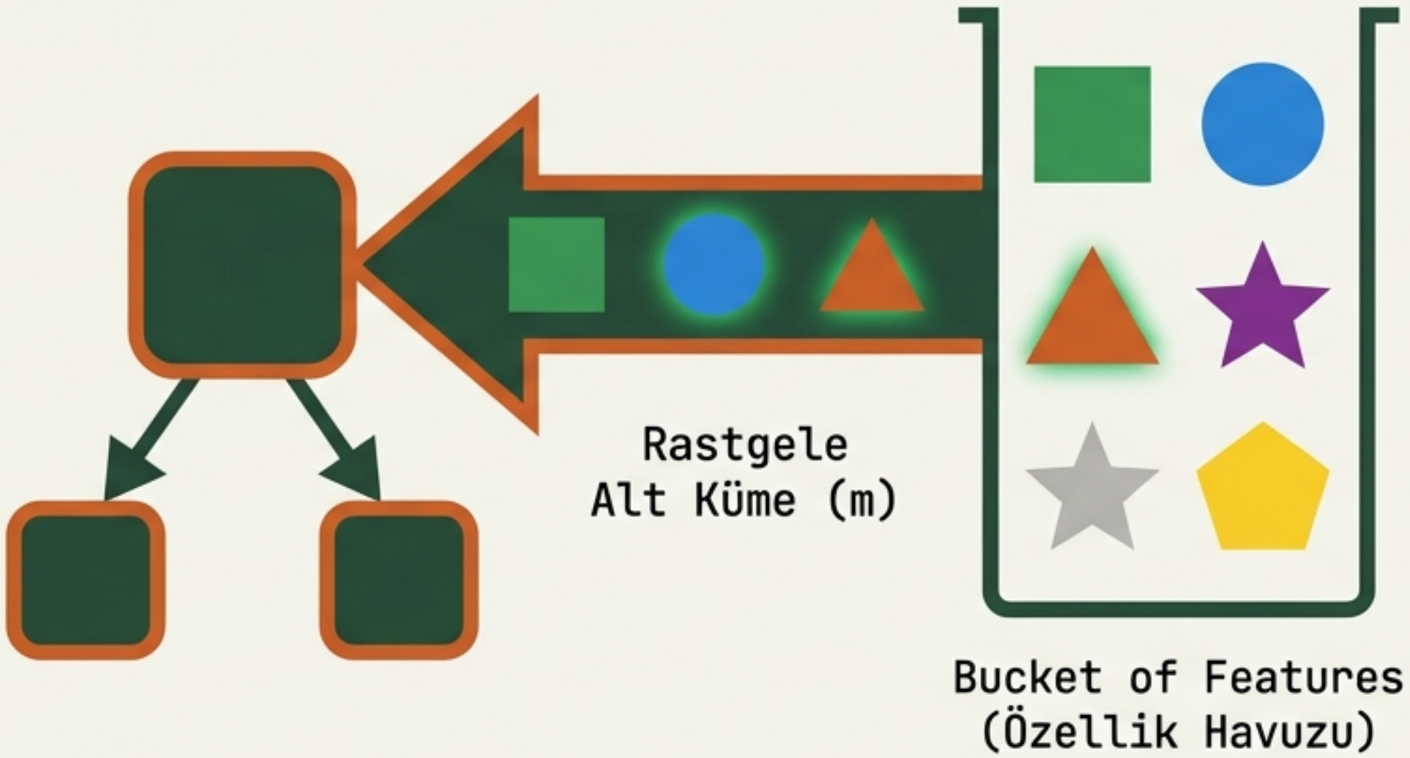


Yüksek korelasyonlu ağaçların ortalaması, tekil bir ağaçla aynı varyans seviyesini korur.

NotebookLM

Çözüm: Random Forest (Rastgele Orman)

Sadece satırları (veri) değil, sütunları da (özellikleri) **rastgele** seçmek.



Her düğüm bölünmesinde **(split)**, mevcut **özelliklerin tamamı** yerine, rastgele seçilen bir alt kümesi (m) değerlendirilir.

Bu kısıtlama, baskın özelliklerin her ağacı domine etmesini engeller ve ağaçları birbirinden bağımsızlaştırır (De-correlation).

Kavram: Rastgele Özellik Seçimi
(Random Feature Selection)

NotebookLM

Adım Adım Random Forest Algoritması



1. Bootstrap

Eğitim verisinden rastgele N büyüklüğünde örneklem al.



2. Rastgele Özellik Seçimi (Random Feature Selection)

Her düğümde, en iyi bölünmeyi bulmak için rastgele 'm' adet özellik seç.



3. Büyütme (Split & Grow)

Seçilen özellikler arasından en iyisini kullanarak düğümü böl. Ağacı budamadan sonuna kadar büyüt.



4. Birleştirme (Aggregate)

Tüm ağaçların tahminlerini 'Çoğunluk Oylaması' ile birleştir.

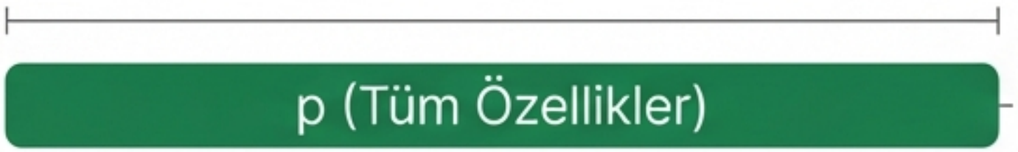
NotebookLM

Kritik Fark: Özellik Seçimi (m vs p)

Bagging

$m = p$

Her bölünmede tüm özellikler değerlendirilir.



Random Forest

$m < p$

Genellikle: $m = \sqrt{p}$

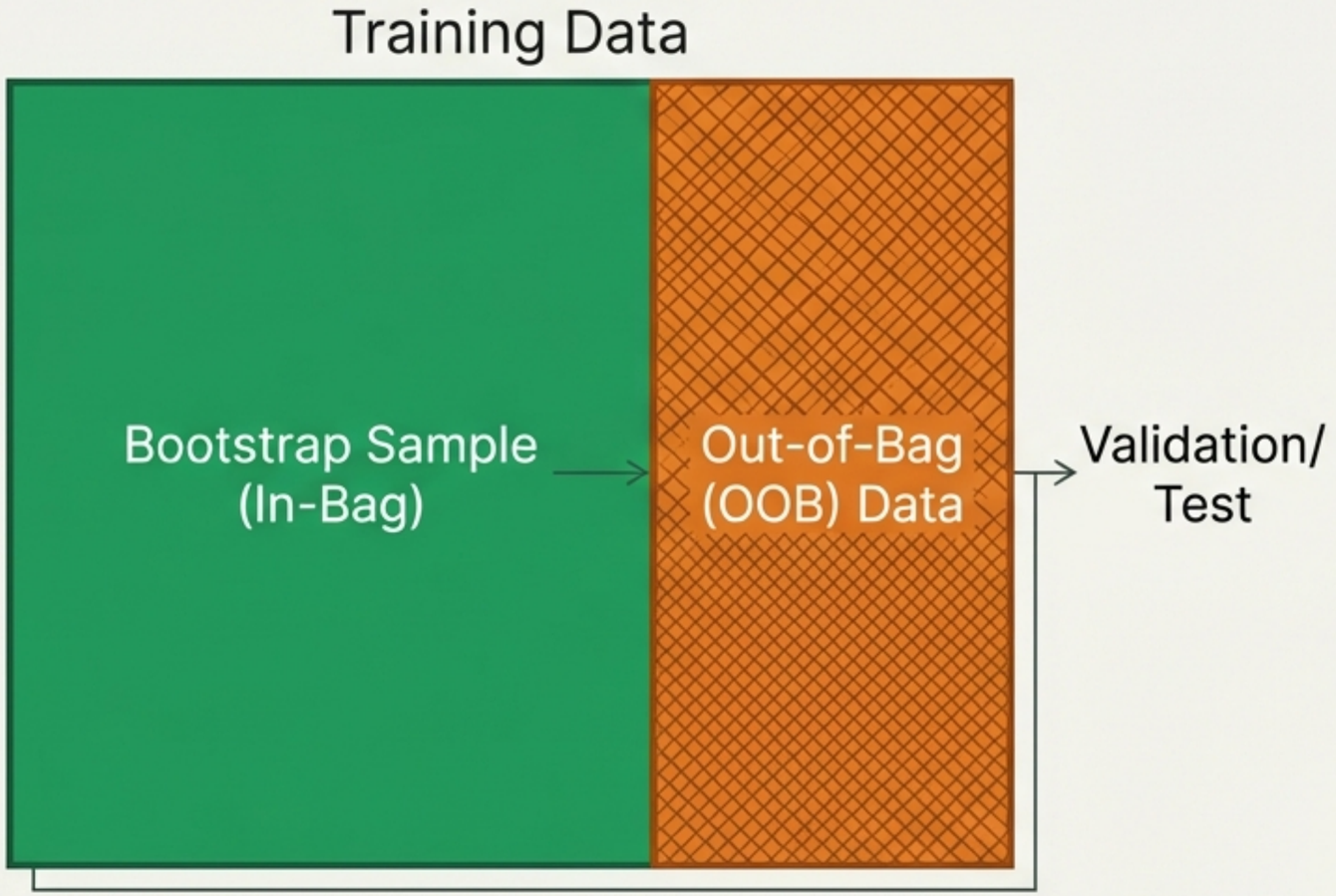
Her bölünmede özelliklerin rastgele bir alt kümesi değerlendirilir.



Bu küçük değişiklik, ağaçlar arasındaki korelasyonu azaltarak modelin genelleme yeteneğini artırır. (m vs p ilkesi)

NotebookLM

Out-of-Bag (OOB) Hatası: Bedava Doğrulama



- Bootstrap işlemi sırasında, verinin yaklaşık **%36.8**'i o ağacın eğitiminde hiç kullanılmaz.
- Bu seçilmeyen veri (**OOB**), o ağacı test etmek için kullanılabilir (Validation Set).

Seçilmeme olasılığı ≈ $1/e \approx 0.368$

NotebookLM

Python ile Uygulama (Scikit-Learn)

python

```python
from sklearn.ensemble import RandomForestClassifier

# Modeli Tanımla
clf = RandomForestClassifier(n_estimators=100, max_features='auto', random_state=42)


# Modeli Eğit
clf.fit(X_train, y_train)

# Tahmin Yap
predictions = clf.predict(X_test)
```

Karmaşık matematiksel arka plana rağmen, uygulama sadece birkaç satır kod gerektirir.

NotebookLM

Editorial Technical Manual

Performansı Yönetmek: Hiperparametreler



`n_estimators`

Ağaç sayısı. Ne kadar çok olursa varyans o kadar düşer, ancak işlem süresi artar. (Varsayılan: 100)



`max_features`

Her bölünmede bakılacak özellik sayısı. Varyansı kontrol eden ana parametredir.



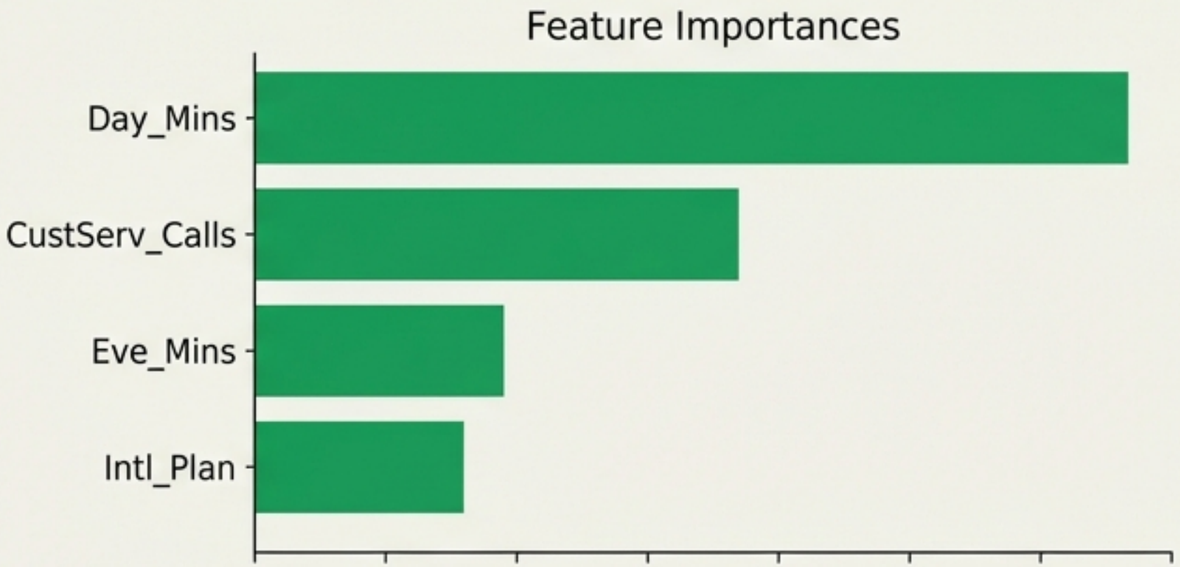
`max_depth`

Ağacın derinliği. Daha derin ağaçlar bias'ı düşürür, sığ ağaçlar varyansı düşürür.

NotebookLM

Editorial Technical Manual

Random Forest sadece tahmin yapmaz, veriyi anlamamızı da sağlar.



Özellik Önemi (Feature Importance): Hangi değişkenlerin tahminde daha etkili olduğunu matematiksel olarak hesaplar. Bu, modelin "yorumlanabilirliğini" artırır.

Random Forest: Yüksek doğruluk, aşırı öğrenmeye direnç ve güçlü özellik analizi.

NotebookLM