BTK Datathon 2026

Kariyer Başarı Skoru Tahmini

Uçtan Uca Makine Öğrenmesi ve NLP Proje Dokümantasyonu

Problem Türü	Regresyon (0-100 arası sürekli hedef değişken)
Değerlendirme Metriği	Ortalama Kare Hata (MSE)
Veri Seti	10.000 train / 10.000 test satırı, 46 özellik
En İyi Kaggle Public Skoru	88.94 (V6 Target Encoding + Soft-Blend Ceiling)
Kullanılan Modeller	XGBoost, LightGBM, CatBoost, Ridge Meta-Model, Stacking
Araçlar	Google Colab, Python, scikit-learn, Optuna

Hazırlayan: Serdar Arıcı
GitHub: github.com/serdararici/btk-datathon-2026
Kaggle: kaggle.com/serdararici
Haziran 2026

Özet

Bu bölüm, dokümanın tamamını okumak istemeyenler için projenin tüm akışını kısaca özetler. Detaylar ve gerekçeler ilgili bölümlerde bulunabilir.

Problem ve Veri

Hedef, öğrenci verilerinden (akademik, teknik, proje, portfolyo, mülakat, sosyal beceri ve mentor geri bildirim metni) yola çıkarak 0-100 arasında sürekli bir career_success_score değerini tahmin etmekti. Değerlendirme metriği MSE (Ortalama Kare Hata) idi, düşük skor daha iyi.

Uygulanan Adımlar

- EDA — eksik değerler, hedef dağılımı, korelasyonlar incelendi; hobby ve sosyal medya platformu gibi sinyal taşımayan "gürültü" sütunları tespit edilip atıldı.
- Ön işleme — eksik değerler mantığa dayalı şekilde dolduruldu (örn. staj yoksa süre 0); university_tier sıralı olduğu için Label Encoding, department/target_role için One-Hot Encoding kullanıldı.
- Feature engineering — beceri ortalamaları, deneyim skoru, role göre ağırlıklandırılmış skor, etkileşim özellikleri (örn. cgpa × proje kalitesi) ve log dönüşümleri türetildi.
- NLP — mentor_feedback_text alanından Türkçe sentiment kelime sayımı ve TF-IDF ile özellikler çıkarıldı; sentiment_score hedefle 0.40 korelasyona ulaştı.
- Modelleme — XGBoost, LightGBM, CatBoost modelleri Optuna ile ayarlandı; bu üç modelin OOF tahminleri bir Ridge meta-modeliyle birleştirilen stacking ensemble kuruldu.
- İleri analiz — modelin neden daha fazla iyileşmediğini anlamak için gürültü analizi, adversarial validation ve birkaç hipotez testi (örnek ağırlıklandırma, yıl sütunlarını çıkarma) yapıldı; bazıları doğrulandı, bazıları kanıta dayalı olarak elendi.

Sonuç

Kaggle public MSE skoru, altı versiyon boyunca 99.09'dan 88.94'e düşürüldü — toplam 10.15 puanlık iyileştirme. En iyi model; ayarlanmış XGBoost, LightGBM ve CatBoost'u Ridge meta-model ile birleştiren, Target Encoding ve Soft-Blend Ceiling Correction uygulanan stacking ensemble'dır. Proje boyunca en büyük tekil iyileştirme hiperparametre tuning'den geldi (~6 puan); NLP ve V6 iyileştirmeleri de belirgin katkı sağladı.

1. Projeye Genel Bakış

Bu proje, Google ve Girişimcilik Vakfı iş birliğiyle BTK Akademi tarafından düzenlenen Datathon 2026 yarışması için geliştirilmiştir. Amaç, öğrencilerin akademik, teknik, proje, portfolyo, mülakat ve sosyal profil verilerinden yola çıkarak career_success_score (kariyer başarı skoru) adı verilen, 0 ile 100 arasında sürekli bir hedef değişkeni tahmin etmektir.

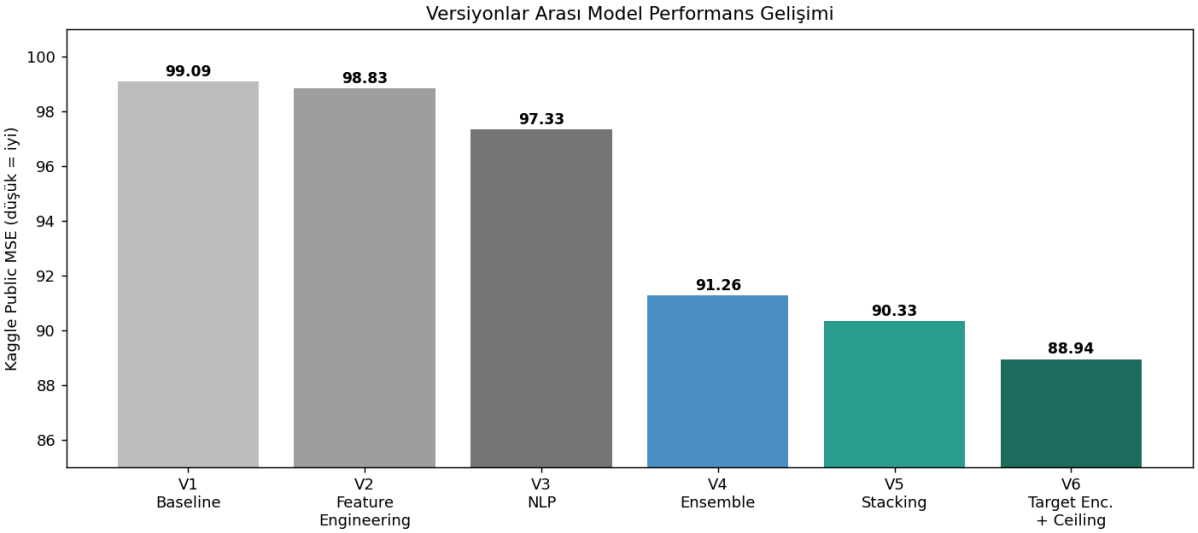
Veri seti üç farklı veri türünü tek bir problemde birleştiriyor: sayısal özellikler (beceri skorları, sayılar), kategorik özellikler (bölüm, hedef rol, üniversite seviyesi) ve bir doğal dil alanı (mentor_feedback_text) — mentor tarafından yazılmış kısa Türkçe değerlendirme metni. Yarışmanın önemli bir gereksinimi, sadece klasik sayısal değişkenlerden değil, bu serbest metin alanından da NLP teknikleriyle bilgi çıkarmaktı.

Yarışma Ortalama Kare Hata (MSE) ile değerlendirilir: düşük değer daha iyi anlamına gelir. MSE her hatayı kareye aldığı için büyük hatalar küçük hatalardan çok daha ağır cezalandırılır, bu da modeli tutarlı ve dengeli tahminlere yönlendirir.

1.1 Metodoloji — Aşamalı (İteratif) Geliştirme

Tek seferde büyük bir model kurmak yerine proje, aşamalı ve ölçülebilir bir yaklaşım izledi. Her versiyon (V1–V5) öncekinin üzerine tek bir büyük fikir ekledi ve her versiyon Kaggle'a gönderilerek her değişikliğin gerçek etkisi gözlemlendi. Bu yaklaşım profesyonel veri bilimi iş akışlarını yansıtır ve hangi fikirlerin işe yaradığını, hangilerinin yaramadığını kesin olarak görmemizi sağladı.

Versiyon	Eklenen Geliştirme	Kaggle MSE
V1	Baseline XGBoost (feature engineering yok)	99.09
V2	Feature engineering (toplamlar, oranlar)	98.83
V3	NLP özellikleri (sentiment + TF-IDF)	97.33
V4	Ensemble (XGB+LGB+Cat) + Optuna tuning	91.26
V5	Stacking + etkileşimler + log dönüşümleri	90.33
V6	Target Encoding + Soft-Blend Ceiling Correction	88.94



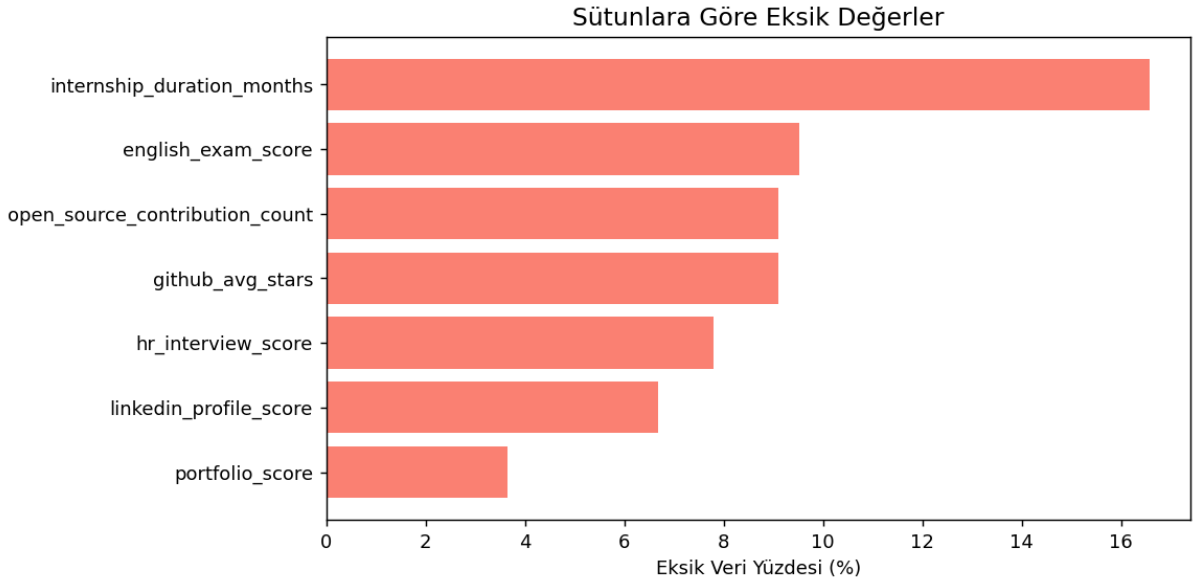
Şekil 1: MSE, altı iterasyon boyunca 99.09'dan 88.94'e düzenli şekilde düştü.

2. Keşifsel Veri Analizi (EDA)

Her veri bilimi projesi EDA ile başlar — modellemeden önce veriyi anlamak. EDA şu sorulara cevap verir: veri nasıl şekillenmiş, eksik değerler nerede, hedef değişken nasıl dağılmış ve hangi özellikler hedefle ilişkili. Buradaki bulgular sonraki tüm kararları şekillendirdi.

2.1 Eksik Değerler

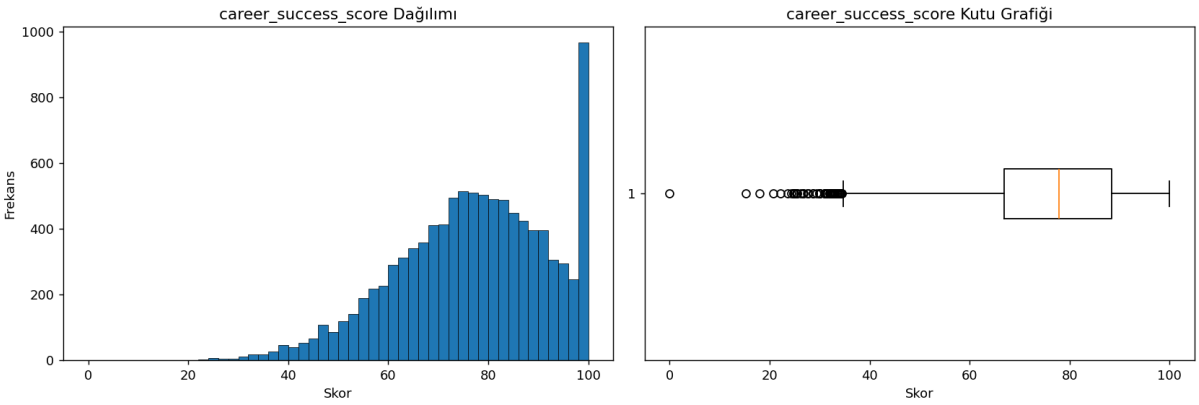
Yedi sütunda eksik değer bulundu. En kötüsü internship_duration_months idi (~%16.57 eksik). Bunu incelemek önemli bir patern ortaya çıkardı: eksik süreye sahip satırların %82'si sıfır staj yapmış öğrencilere aitti. Bu, değerin gerçekte "eksik" olmadığı, mantıksal olarak 0 olması gerektiği anlamına geliyordu. Bu bulgu, kör medyan doldurma yerine akıllı bir doldurma stratejisi geliştirmemizi sağladı.



Şekil 2: Sütunlara göre eksik değerler. En çok eksik internship_duration_months'ta.

2.2 Hedef Değişken Dağılımı

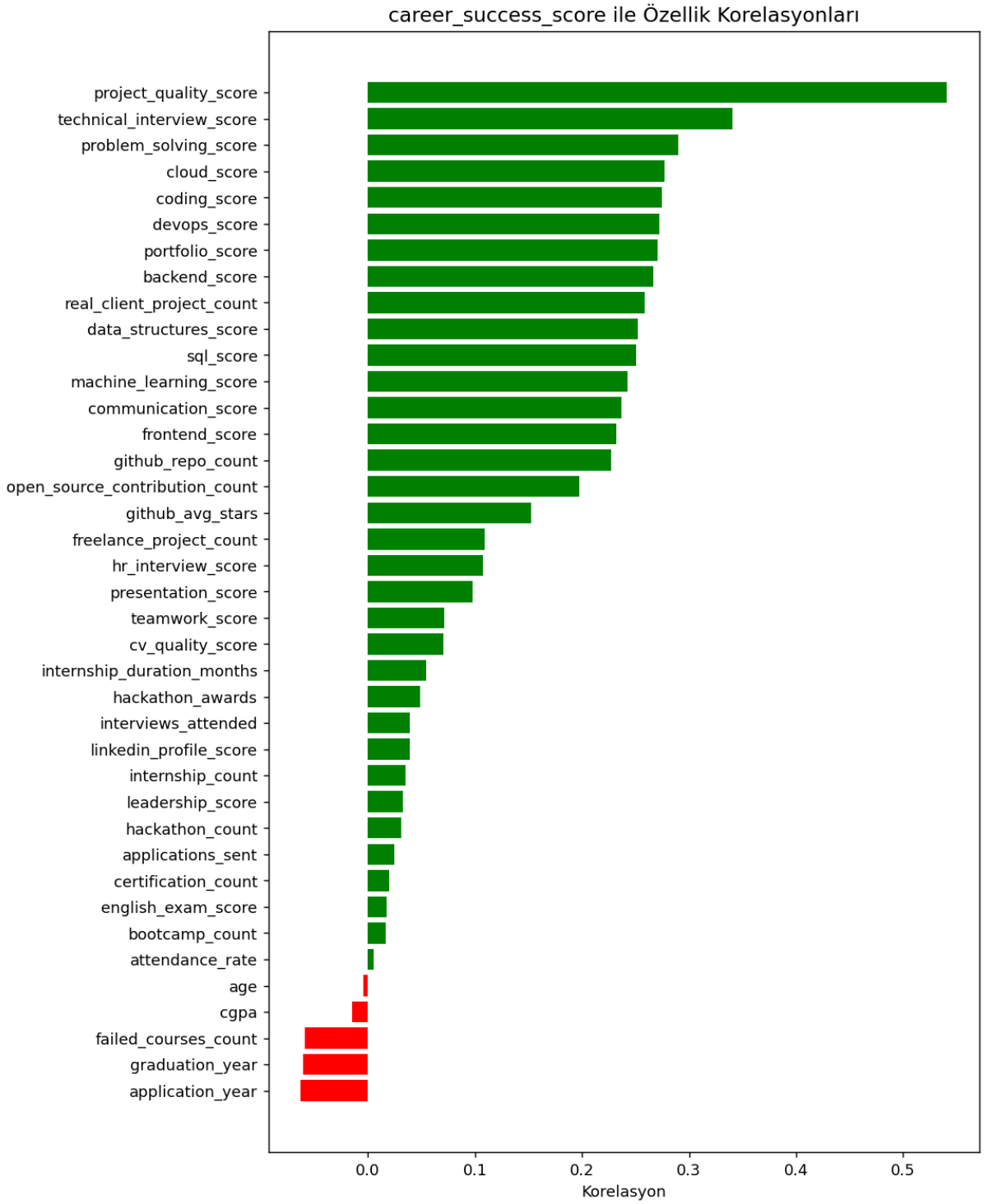
Hedef değişkenin (career_success_score) ortalaması 76.9, standart sapması 15.2 ve aralığı 0–100. Dağılım hafif sola çarpık (çarpıklık −0.45): öğrencilerin çoğu 67–88 aralığında skor alırken, düşük skorlara doğru daha uzun bir kuyruk var. Bu bize "sadece ortalamayı tahmin et" şeklinde basit bir modelin RMSE değerinin ~15 civarında olacağını gösterdi — bunu geçmemiz gereken net bir başlangıç noktası oldu.



Şekil 3: Hedef değişken 77 civarında yoğunlaşmış; 773 öğrenci tam 100 puan almış (tavan etkisi).

2.3 Özellik Korelasyonları

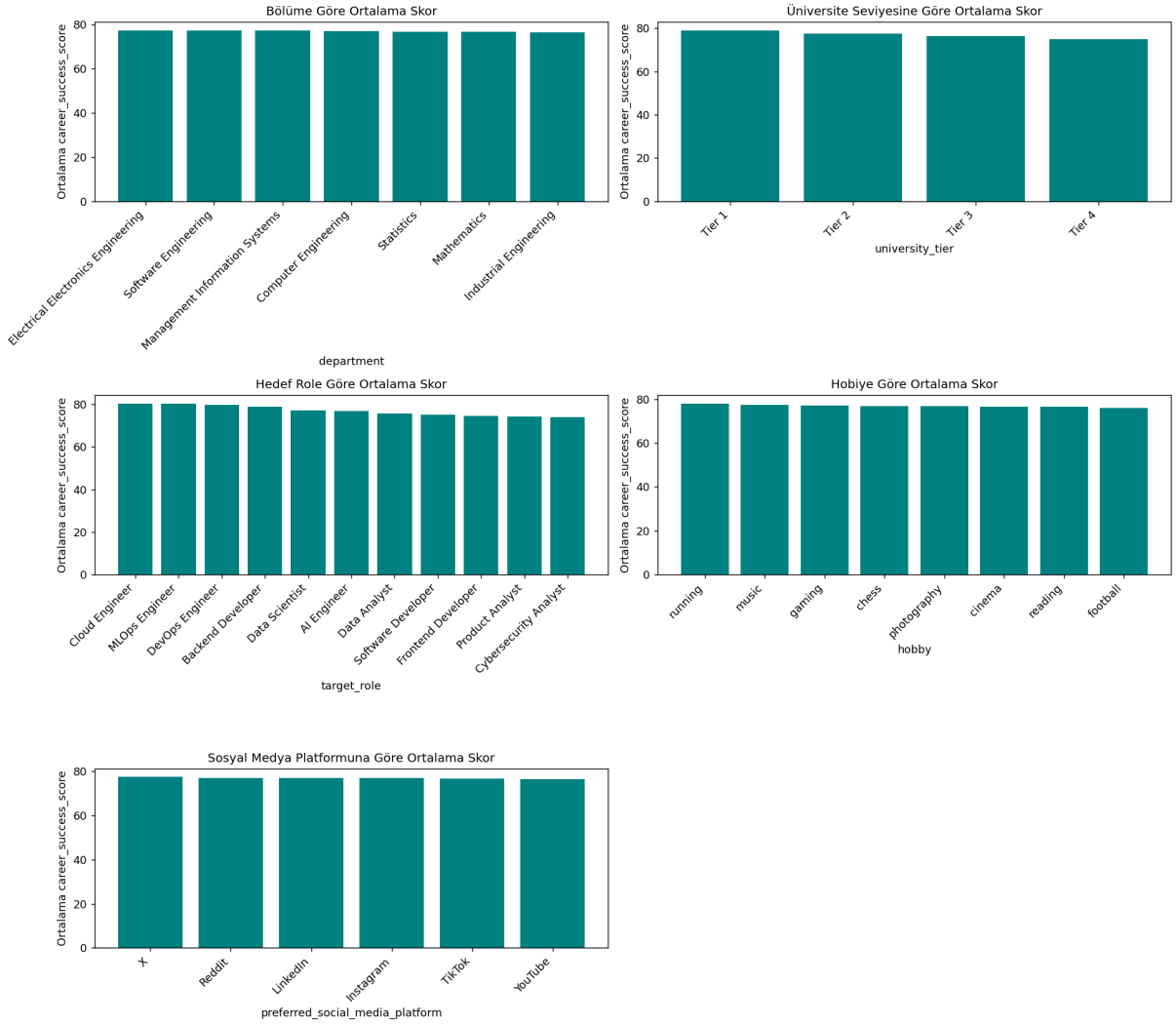
Korelasyon analizi hangi özelliklerin sinyal taşıdığını ortaya koydu. project_quality_score açık farkla en güçlüsüydü (0.54), ardından technical_interview_score (0.34) ve bir grup teknik beceri (0.25–0.29) geliyordu. Çarpıcı bir bulgu: cgpa'nın hedefle korelasyonu neredeyse sıfırdı (−0.01) — akademik not ortalaması tek başına kariyer başarısını öngörmüyor. Bu sezgiye aykırı sonuç, ileride yapılan ileri analizde önemli bir rol oynadı.



Şekil 4: project_quality_score baskın; cgpa ve yıl özellikleri tek başına neredeyse sıfır.

2.4 Kategorik Özellikler — Gürültü Tespiti

Kritik bir EDA adımı, kategorik sütunların gerçekten önemli olup olmadığını kontrol etmekti. Her kategori için ortalama hedef skorunu çizdirince, hobby ve preferred_social_media_platform sütunlarının kategoriler arasında neredeyse hiç fark göstermediği görüldü — tüm barlar ~77 civarındaydı. Bunlar etkin olarak gürültü sütunları (yarışma verisine kasıtlı olarak eklenmiş tuzaklar olabilir). Bunları tespit edip atmak, tüm sütunları körü körüne modele veren diğer yarışmacılara karşı ince bir avantaj sağladı.



Şekil 5: department, hobby ve platform düz ortalamalar gösteriyor (gürültü); target_role ve university_tier hafif bir yapı gösteriyor.

3. Veri Ön İşleme (Preprocessing)

Ön işleme, ham veriyi modelleme için hazırlar. Bu aşamada iki ilke yol gösterdi: data leakage'dan kaçınmak (test bilgisinin asla eğitimi etkilememesi) ve alana özgü akıllı doldurma (eksikleri kör kör değil, mantıklı şekilde doldurmak).

3.1 Akıllı Eksik Değer Doldurma

internship_duration_months için mantığı ikiye ayırdık: sıfır staj yapan öğrencilere 0, staj yapmış ama süresi eksik olanlara ise train medyanı (8.0) verildi. Diğer tüm sayısal eksikler (english_exam_score, github_avg_stars vb.) sadece train medyanı ile dolduruldu — aynı medyanlar test setine de uygulandı. İstatistikleri sadece train'den hesaplamak data leakage'ı önleyen temel kuraldır.

3.2 Kategorik Değişken Kodlama (Encoding)

Kategorinin sıralı olup olmamasına göre iki strateji seçildi:

- Label Encoding: university_tier için — sıralı bir yapı var (Tier 1 > Tier 2 > ...), 4,3,2,1 olarak eşlendi.
- One-Hot Encoding: department ve target_role için — sırasız kategoriler, her biri 0/1 sütununa dönüştürüldü.

Burada ufak ama önemli bir hata yakalandı ve düzeltildi: sütun hizalama (align) işlemi sırasında hedef sütun, yanlışlıkla sıfırlarla doldurularak test setine kopyalanıyordu. Hedefi encoding'den önce ayırmak bu sorunu çözdü — dikkatli doğrulamanın neden önemli olduğunu gösteren iyi bir örnek.

3.3 Atılan Sütunlar

EDA bulgularına göre gürültü sütunları (hobby, preferred_social_media_platform), kimlik sütunu (student_id — sadece final submission dosyası için ayrı tutuldu) ve ham metin alanı (mentor_feedback_text — çıkarılan NLP özellikleriyle değiştirildi) model girdisinden çıkarıldı.

4. Feature Engineering (Özellik Mühendisliği)

Ham veriden yeni ve daha anlamlı sütunlar türetmek olan feature engineering, çoğu zaman bir yarışmadaki en değerli aşamadır. Aşağıdaki özellikler oluşturuldu:

• avg_technical_score — 9 teknik beceri skorunun ortalaması. Önem sıralamasında 2. sıraya yerleşti.

• avg_soft_skill_score — iletişim, takım çalışması, liderlik, sunum ortalaması.

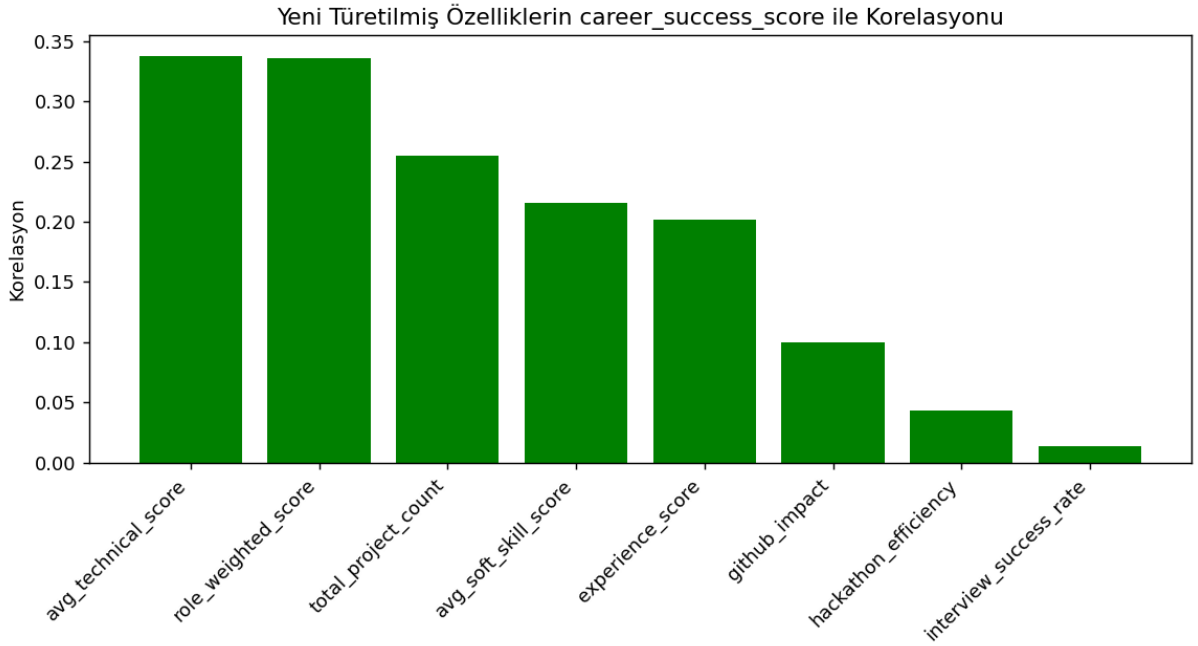
• experience_score — staj, gerçek müşteri projesi, freelance ve hackathon'ların ağırlıklı bileşimi.

• role_weighted_score — öğrencinin hedef rolüne göre ağırlıklandırılmış teknik beceriler (örn. Data Scientist için ML daha ağırlıklı).

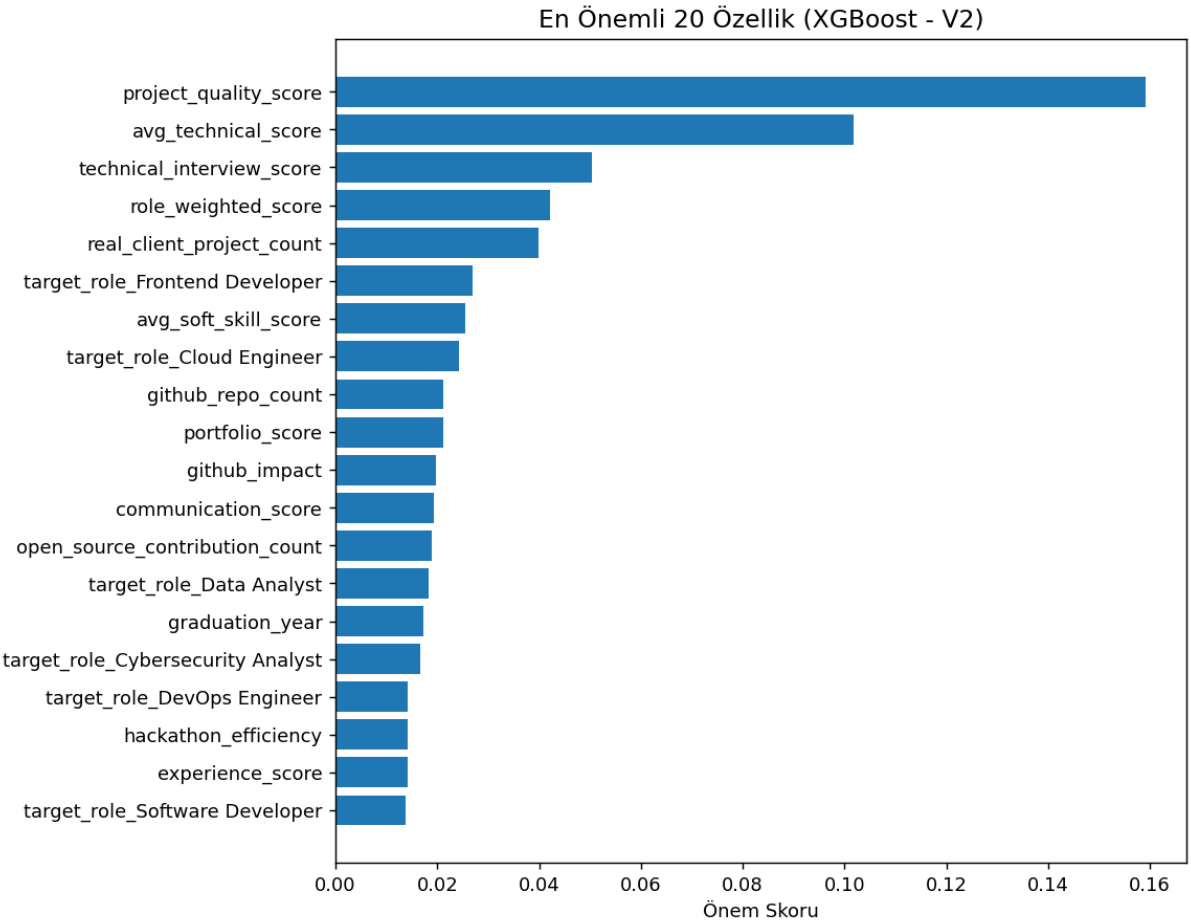
• Etkileşim özellikleri — technical_interview × problem_solving gibi çarpımlar, "güçlü VE gösterilebilir" profilleri yakalamak için.

• cgpa etkileşimleri — cgpa × project_quality gibi, cgpa'nın gizli/kontrollü önemini ortaya çıkarmak için.

• Log dönüşümleri — github_avg_stars gibi sağa çarpık sayım özelliklerine log1p uygulandı (korelasyonu 0.15'ten 0.18'e çıkardı).



Şekil 6: V2'de türetilen 8 yeni özelliğin hedefle korelasyonu — avg_technical_score ve role_weighted_score en güçlüsü.



Şekil 7: XGBoost modelinde en önemli 20 özellik — avg_technical_score, ham sütunlar arasında 2. sıraya yerleşti.

Türetilen her özellik eşit derecede faydalı olmadı. interview_success_rate ve hackathon_efficiency neredeyse sıfır korelasyon gösterdi ve az katkı sağladı — "daha fazla özellik otomatik olarak daha iyidir" yanılgısına karşı bir hatırlatma. Ağaç tabanlı modeller birçok etkileşimi zaten kendiliğinden yakalayabildiğinden, elle yapılan etkileşimler burada sadece sınırlı bir kazanç sağladı.

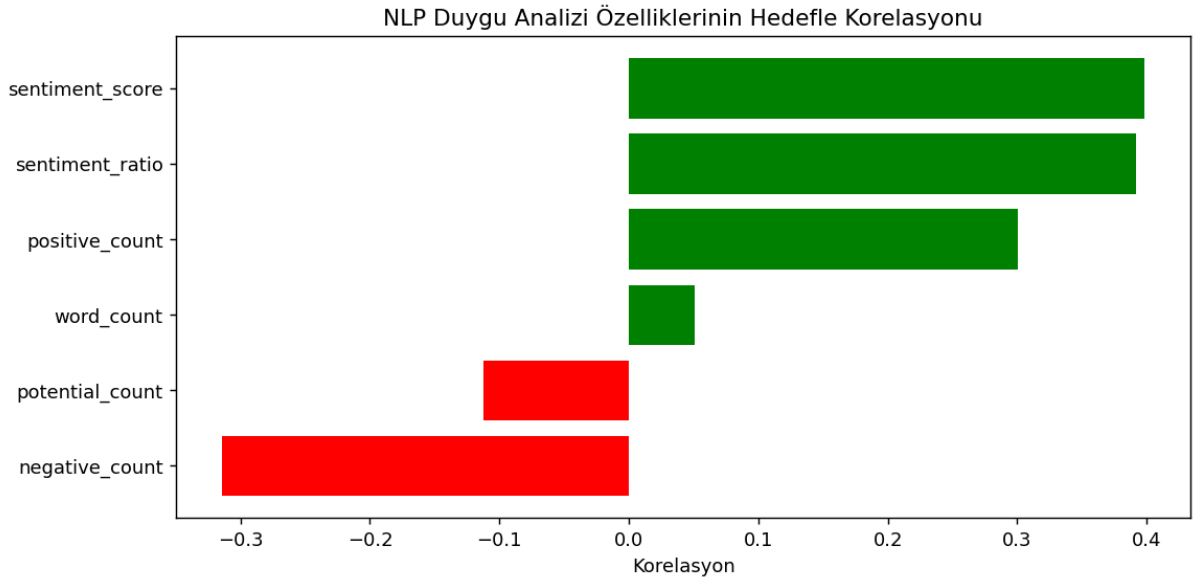
5. Doğal Dil İşleme (NLP)

mentor_feedback_text alanı, mentor tarafından yazılmış kısa bir Türkçe değerlendirmedir. Örnekleri okumak net bir patern ortaya çıkardı: yüksek skorlu öğrenciler "dikkat çekici", "güçlü", "potansiyeli büyük" gibi kelimeler alırken, düşük skorlular "ancak", "çalışması gerekiyor", "gelişmekte" gibi ifadeler alıyordu. Üç katmanlı bir NLP stratejisi kuruldu:

- Duygu (sentiment) kelime sayımı — elle oluşturulmuş Türkçe pozitif / negatif / potansiyel kelime listeleri, metin başına sayıldı.
- Sentiment skoru ve oranı — pozitif eksi negatif, ve pozitifin toplam içindeki payı.
- TF-IDF — en ayırt edici 100 kelime/kelime grubu otomatik olarak seçildi (Türkçe stopword temizliği ve Türkçe'ye özel küçük harfe çevirme ile).

Sonuçlar yaklaşımı doğruladı. sentiment_score, hedefle 0.40 korelasyona ulaştı — tüm veri setinde project_quality_score'dan sonra ikinci en güçlü sinyal. TF-IDF, elle oluşturulan

listeleri bağımsız olarak doğruladı: "ancak" kelimesi en güçlü negatif sinyaldi (−0.24), çünkü Türkçe'de genellikle bir övgüden sonra eleştiri getirir. İlginç bir nüans: "potansiyel" kelimeleri negatif korelasyon gösterdi — mentorlar bu kelimeleri henüz başarılı olmayan ama umut veren öğrenciler için kullanıyor.



Şekil 8: sentiment_score ve sentiment_ratio en güçlü NLP sinyalleri; negatif kelimeler negatif korelasyon gösteriyor.

6. Modelleme ve Ensemble

6.1 Neden Gradient Boosting

Bu tür tablo verisi için gradient boosting modelleri kanıtlanmış en güçlü yöntemdir. Üç model kullanıldı: XGBoost, LightGBM ve CatBoost. Her biri, önceki ağaçların hatalarını düzeltmeye çalışan sıralı bir karar ağacı dizisi kurar. İç mekanikleri farklılaşır — CatBoost özellikle kategorik yapıyı çok iyi işler, bu yüzden burada sürekli en güçlü tekil model oldu.

6.2 Optuna ile Hiperparametre Optimizasyonu

Varsayılan model ayarları nadiren en uygunudur. Optuna, çapraz doğrulanmış MSE'yi hedef alarak en iyi hiperparametreleri (ağaç sayısı, öğrenme hızı, derinlik, örnekleme oranları) otomatik olarak aramak için kullanıldı. Tuning, projenin tek başına en büyük iyileştirmesini sağladı — Kaggle skorunda yaklaşık 6 puanlık bir düşüş (97.33 → 91.26) — bu da doğru ayarların herhangi bir tekil özellikten daha fazla önem taşıdığını doğruladı.

6.3 Stacking Ensemble ve Meta-Model Seçimi

Basit bir ortalama yerine, final model stacking kullandı. Her temel model, hiç eğitilmediği veri üzerindeki tahminler olan out-of-fold (OOF) tahminler üretti; bunlar daha sonra bir "meta-model"e girdi olarak verildi. OOF tekniği leakage'ı önlemek için gereklidir: temel modeller eğitildikten sonra aynı satırları tahmin etmeleri istenseydi, o satırları "ezberler" ve meta-modeli yanıltırlardı.

Meta-model olarak Ridge Regresyon seçildi. Üç temel model (XGBoost, LightGBM, CatBoost) birbirine çok benzer tahminler ürettiği için, meta-modelin görevi karmaşık yeni ilişkiler öğrenmek değil, üç tahmini dengeli bir şekilde birleştirmekti — bu da basit ve sağlam bir modelin (Ridge) karmaşık bir modelden (örneğin başka bir XGBoost) daha uygun olduğu bir durumdur. Ridge, sıradan Linear Regression'dan farklı olarak L2 regularization içerir; bu, birbirine çok benzeyen girdiler (üç model tahmini) arasında ağırlıkların aşırı uçlara kaymasını önleyerek daha dengeli ve genellenebilir bir birleşim sağlar.

Ridge, sabit ağırlıklar kullanmak yerine her temel modele ne kadar güvenileceğini veriden öğrendi: final_tahmin = 0.35 × XGBoost + 0.08 × LightGBM + 0.60 × CatBoost − 1.98. En yüksek ağırlık, en düşük OOF hatasına sahip CatBoost'a verildi; LightGBM'in ağırlığı ise neredeyse sıfıra yakındı çünkü performansı diğer ikisinin gerisinde kalmıştı. Stacking, hem basit ortalamayı hem de en iyi tekil modeli geride bıraktı.

7. İleri Analiz ve Hipotez Testleri

V5 sonrası skor ~90 seviyesinde sabitlenince, nedenini anlamak için daha derin bir teşhis aşaması yürütüldü. Birden fazla hipotez test edildi — bazıları doğrulandı, bazıları reddedildi. Her iki türden sonucu da dürüstçe belgelemek, sağlam veri biliminin bir parçasıdır.

• Gürültü analizi — Lineer bir model ~9.9 standart sapmalı bir kalıntı (residual) bıraktı (R^2 ~0.57), bu da hedef değişkende anlamlı miktarda indirgenemeyen gürültü olduğunu gösterdi. Yapılan bir deneyde çapraz doğrulama MSE'si ~82 olarak ölçüldü; ancak bu kesin bir matematiksel alt sınır değildi — ilerleyen adımlarda daha fazla feature engineering ve daha kapsamlı tuning ile bu değerin altına (~79'a) inilebildi. Veri sentetik üretildiği için gerçek gürültü oranı yalnızca veriyi üreten taraf tarafından bilinebilir; bizim rakamlarımız sadece o ana kadar ulaşılan en iyi tahminlerdir.

• cgpa paradoksu — cgpa tek başına ~0 korelasyona sahip, ama diğer özellikler kontrol edildiğinde yüksek bir katsayı alıyor. Etkisi etkileşimlerde gizli; bu bulgu cgpa etkileşim özelliklerinin eklenmesine yol açtı.

• Adversarial validation — Bir sınıflandırıcı train ile test'i ayırt edebildi (AUC 0.66), bunun başlıca sebebi application_year ve graduation_year idi: test seti daha yakın yıllardan geliyor.

• Örnek ağırlıklandırma denemesi — Train satırlarını test'e benzerliklerine göre ağırlıklandırmak denendi, ancak çapraz doğrulamayı kötüleştirdi (83.1 → 84.5) ve bu yöntem kullanılmadı.

• Yıl sütunlarını çıkarma denemesi — Dağılım kaymasıyla başa çıkmak için yıl sütunları çıkarılıp Kaggle'da test edildi, ancak skor kötüleşti (90.3 → 100.5). Sonuç: kayma olmasına rağmen yıl özellikleri gerçekten faydalı, bu yüzden korundu.

Toplanan kanıtlar, modelin veride mevcut olan gerçek sinyale yakın bir performansa ulaştığını gösteriyor. Çapraz doğrulama (~79) ile Kaggle (~90) arasındaki kalan fark, modelleme hatasından değil, public test alt kümesinin doğal zorluğundan ve yıl dağılımı kaymasından kaynaklanıyor; bu farkın tamamen kapatılıp kapatılamayacağı, yarışma süresi içinde kesin olarak test edilememiştir.

8. V6 İyileştirmeleri — Target Encoding ve Tavan Düzeltmesi

V5'in 90.33 skorunun ardından, iki somut iyileştirme fırsatı tespit edilip uygulandı. Her iki yöntem de önce çapraz doğrulamada test edildi, ardından Kaggle'a gönderildi ve kanıtlandı. Bu aşama, "işe yarayan" ve "yaramayan" fikirlerin sistematik olarak değerlendirildiği bir örnek oluşturmaktadır.

8.1 OOF Target Encoding

V5'te department ve target_role sütunları One-Hot Encoding ile sayıya çevriliyordu — her kategori için 0/1 sütunları oluşturuluyordu. V6'da bu, Out-of-Fold Target Encoding ile değiştirildi: her kategoriye, o kategorideki öğrencilerin ortalama kariyer başarı skoru atanır. Veri sızıntısını (leakage) önlemek için bu ortalamalar, fold dışındaki train verisiyle hesaplandı.

Bu yöntem, kategorik bilgiyi daha özlü biçimde modele iletir — model, "bu sütun 1 mi 0 mı" sorusu yerine "bu kategorinin ortalama skoru kaç" sorusuna doğrudan cevap alır. CV'de -0.64 MSE kazancı sağladı.

8.2 Soft-Blend Tavan Düzeltmesi

EDA'da tespit edilen kritik bulgu: 773 öğrenci tam 100 puan almıştı (kırpılmış/sansürlü hedef). Ağaç modelleri bu grubu sistematik olarak ~95 civarında tahmin ediyordu — ortalama 4.73 puanlık bir eksik. MSE kare aldığı için bu hata orantısız büyüktü.

Doğal çözüm gibi görünen "sert eşik" yöntemi (94 üstünü 100'e çek) test edildi fakat başarısız oldu — yanlış 100 atamaları (false positive) kazançtan fazla zarar verdi. Bunun yerine öğrenilmiş bir sınıflandırıcıya dayalı yumuşak harmanlama kullanıldı:

final = P(100) × 100 + (1 − P(100)) × stacking_tahmini

Model ne kadar emin olursa tahmini o kadar 100'e yaklaştırır; emin değilse neredeyse dokunmaz. Bu yöntem CV'de -0.71 MSE kazancı sağladı. İki iyileştirme birlikte uygulandığında CV'de -1.50 puan, Kaggle'da 1.39 puan (90.33 → 88.94) iyileşme sağlandı.

8.3 Reddedilen Yöntemler

Aynı aşamada birden fazla başka yöntem de denendi ve kanıta dayalı olarak elendi:

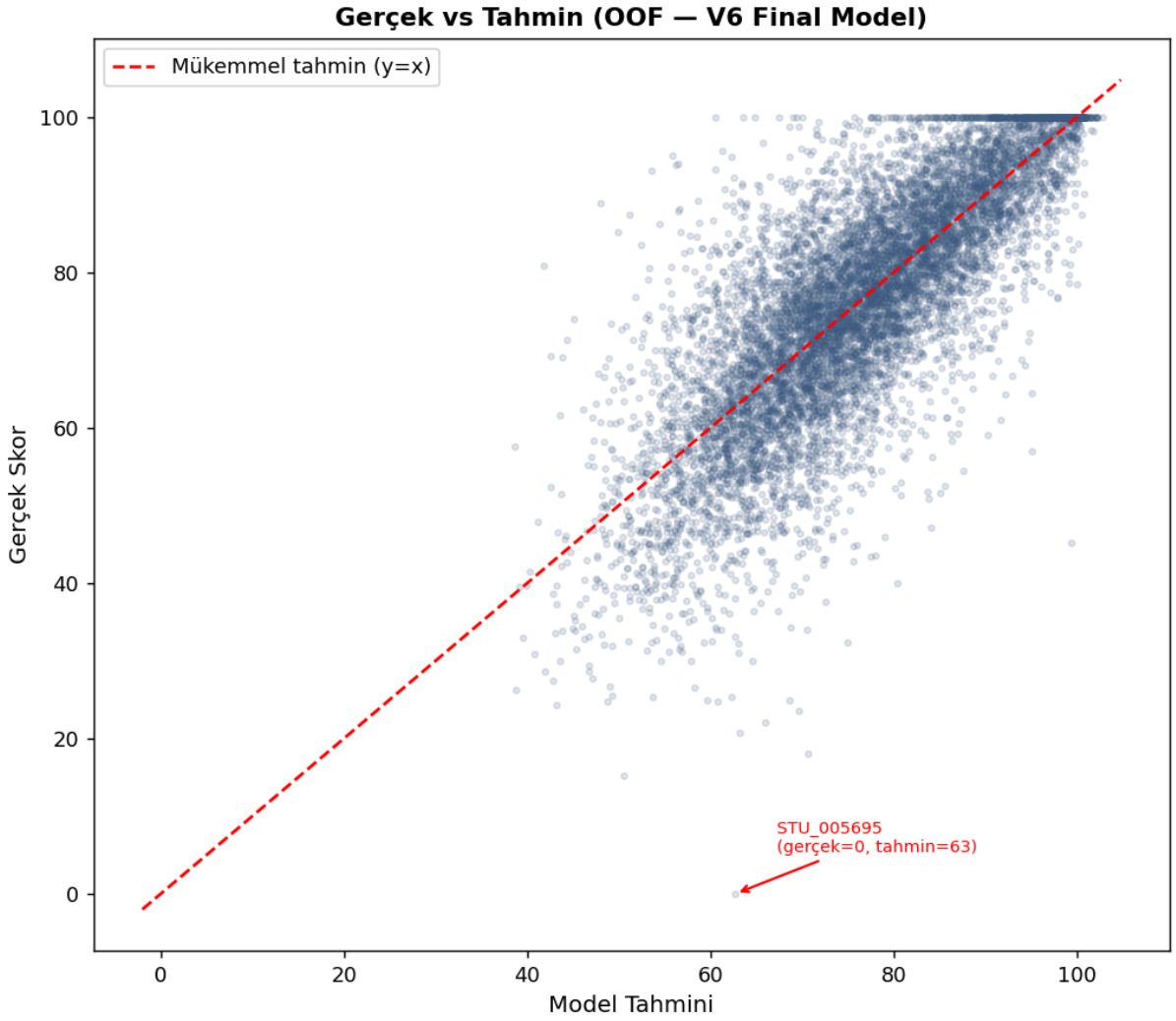
- Early stopping: CV'de fark sağlamadı (82.38 → 82.39). Mevcut regularizasyon parametreleri zaten yeterliydi.
- CatBoost native kategorik: Manuel target encoding'den daha kötü çıktı (80.95 vs 80.50).
- TF-IDF 200 özellik (100 yerine): Anlamlı fark oluşturmadı.
- Sert eşik tavan düzeltmesi (>94 → 100): CV'de kötüleşti; soft-blend yöntemi tercih edildi.

8.4 Model Hata Analizi (Residual Analizi)

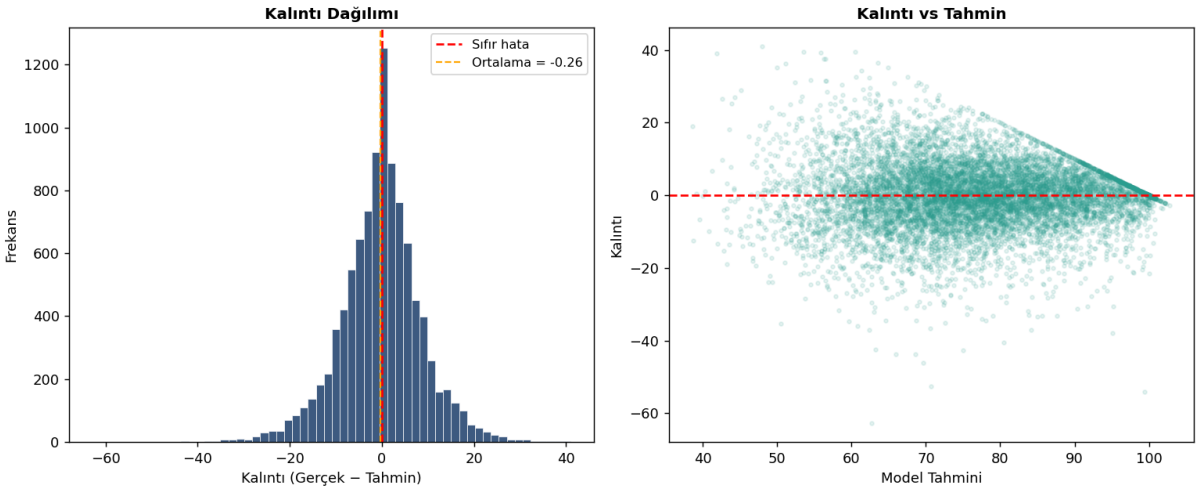
Modelin nerede başarılı nerede başarısız olduğunu anlamak için iki temel grafik üretildi. Bu grafikler, modelin tarafsız (unbiased) ama belirli yapısal gürültülerden etkilenen bir model olduğunu ortaya koyuyor.

Actual vs Predicted: Model, 50-95 arasındaki öğrencileri başarıyla yakalıyor. İki belirgin sorun öne çıkıyor: (1) tam 100 alan öğrenciler için sistematik alt-tahmin — soft-blend ile büyük ölçüde giderildi, (2) veri setindeki aykırı değerler (örn. STU_005695: tüm özellikleri ortalama-üstü ama gerçek skoru 0) — bunlar modelin öngöremediği etiket gürültüsü.

Residual Dağılımı: Kalıntılar (gerçek − tahmin) sıfır merkezli, simetrik bir dağılım gösteriyor (ortalama −0.26). Bu, modelin sistematik bir yönde yanıltmadığını, yani tarafsız (unbiased) olduğunu kanıtlar. Uzun kuyruklardaki hatalar, veri setine kasıtlı olarak eklenmiş yapısal gürültüden kaynaklanıyor.



Şekil 9: Model 50-95 bandını iyi yakalıyor. STU_005695 (gerçek=0, tahmin~64) en büyük hata kaynağı — etiket gürültüsü.



Şekil 10: Kalıntılar sıfır merkezli (ort. −0.26), simetrik dağılım → model tarafsız. Uzun kuyruklar yapısal gürültüden.

9. Sonuçlar ve Değerlendirme

Proje, Kaggle public MSE skorunu 99.09'dan 88.94'e düşürdü — disiplinli ve ölçülebilir iterasyon yoluyla 10.15 puanlık bir iyileştirme. En iyi model; ayarlanmış XGBoost, LightGBM ve CatBoost'u Ridge meta-model ile birleştiren, Target Encoding ve Soft-Blend Ceiling Correction uygulanan, 167 özellikli bir stacking ensemble'dır.

Aşama	Temel Katkı	Etki
EDA	Gürültü sütunlarının ve akıllı doldurma hedeflerinin belirlenmesi	Temel
Feature Eng.	Toplamlar, oranlar, etkileşimler	Orta
NLP	Sentiment (0.40 korelasyon) + TF-IDF	Güçlü
Tuning (Optuna)	Optimal hiperparametreler	En büyük (~6 puan)
Stacking	Ridge meta-model ile dinamik model birleştirme	İnce ayar
V6 (Target Enc. + Ceiling)	Encoding iyileştirmesi + tavan düzeltmesi	1.39 puan

9.1 Temel Çıkarımlar

• Hiperparametre tuning en büyük tekil kazancı sağladı — doğru ayarlar bazen özelliklerden daha önemli olabiliyor.

• NLP gerçekten değer kattı; Türkçe "ancak" kelimesi metin tabanlı en güçlü negatif sinyaldi.

• Target Encoding ve Soft-Blend Ceiling Correction birlikte 1.39 Kaggle puanı kazandırdı — küçük görünen iyileştirmeler birikimli olarak anlamlı.

• Her fikir işe yaramadı — early stopping, CatBoost native encoding, sert eşik düzeltmesi gibi denemeler kanıta dayalı olarak elendi.

• Veri setinde anlamlı miktarda indirgenemeyen gürültü var; residual analizi modelin tarafsız çalıştığını (ort. hata −0.26) doğruladı.

• Her değişikliği çapraz doğrulama ve Kaggle ile ölçmek, süreci dürüst ve tekrarlanabilir tuttu.

9.2 Araçlar ve Tekrarlanabilirlik

Tüm çalışma Google Colab üzerinde yapıldı (ücretsiz GPU, sıfır kurulum, yerleşik GitHub entegrasyonu). Proje, versiyonlanmış notebook'lara (v1_baseline'dan v6_final'a kadar) ayrılmıştır; tüm veriler, çıktılar ve grafikler versiyon kontrolündedir. Her notebook GitHub'a kaydedilirken kendi Colab açma linkini içerecek şekilde eklenmiştir, bu sayede her notebook tek başına çalıştırılabilir ve doğrudan Colab'da açılabilir durumdadır.

Kaynak	Bağlantı
GitHub Deposu (tüm notebook'lar)	github.com/serdararici/btk-datathon-2026
Kaggle Profili	kaggle.com/serdararici
Çalışma Ortamı	Google Colab (her notebook içinde gömülü Colab linki mevcuttur)