**Model için Veri Hazırlama: (EDA) | Enes Fehmi Manan**

**Moderatör : Göker Güner**

**Enes Fehmi Manan:** Şimdi herkes tekrardan hoş geldi diyelim. Bu sanıyorum... Denen şey, istenen şey, birazcık daha bu **data science**'ın çok bahsedilmeyen bir yanı var. Bir problem size geldiği zaman, o probleme yaklaşım ve diyelim siz bunu **machine learning** ile çözmeye karar verdiniz. Karar verdikten sonraki o kısımda, o problemi çözecek **data seti oluşturmak**. Bu genel olarak verilen eğitimlerde çokça bahsedilen bir şey değil. Bunu ancak işte **sektörde** tek bir işe girdiğin zaman, gerçek bir problem geldiği zaman, **database**'le baş başa kaldığın zaman anlıyorsun. Birazcık bunu hani önceden gör, böyle bir şey olduğunu ve zamanın büyük bir çoğunluğunun bu kısımda geçtiğini ufak bir simülasyonla göstermek istiyorum. Biraz bu yayının ana amacı bu. Bunun için de bir tane kurgu oluşturdum. **Machine learning için veri hazırlama**, gerçekçi bir senaryo. Buradaki senaryonun kapsamı birazcık daha bir bankada çalışan işte **veri bilimcinin kredi** tarafındaki bir... Ama buradaki senaryo işte atıyorum bir e-ticarete veya bir pazarlama tarafındaki başka bir case'e de... Nasıl bir senaryo, biraz yavaştan başlayalım.

Şimdi hikayeye başlamadan önce diyelim ki bir bankada veri bilimci olarak çalışıyorsunuz. Sabahleyin açtınız bilgisayarınızı. Bilgisayarınızda important ve öncelikli bir mail var. **Business**'tan gelebilir bu mail, pazarlamadan gelebilir veya daha kritik **üst yönetimden** gelen bir talep olabilir. Buradan üst yönetim, üst yönetim deniyor, üst yönetim dediğimizde kim hani ondan kısaca açıklamak gerekirse üst yönetim denen kişiler direktör ve onun üstü GMY, genel müdür gibi giden kişiler. Bu kişiler daha çok böyle sizin yaptığınız analizle değil de sayılarla ilgilenir, işin sonucu ile ilgilenir, işin sonucunda size bir şey söylerler. O dediği şey tüm şirkette dalga ile birlikte iş sizin oradaki modeli kurmaya kadar, işte önce pazarlamaya gider oradan business'a, business'tan size gelebilir falan filan. Şimdi diyelim ki böyle bir mail bize düştü. Mailde şöyle bir şey gözlemlenmiş; **batık kredi verme oranının** son 3 aylık periyotta çok yükseldiği. Üst yönetim de size diyor ki, bunu düşürün. X Bey, Hanım böyle dedi, bizim bunu düşünmemiz lazım. Nasıl düşürürüz?

Şimdi ilk soru. Hemen bize bir problem geldi, problemi anlayalım. **Batık kredi verme oranı**. **Batık kredi** verme ne demek? Bir kişiye bir kredi veriyoruz ve verdiğimiz krediyi güvenmişiz ona geri ödeyeceğine. Fakat kredi batırmış, bize geri ödeyememiş. Şu an biz elimizdeki paradan hem de alacağımız kardan olmuşuz. Bundan dolayı batık kredi oranı artarsa ciddi riskler barındırır. Bunu optimum halde tutmamız lazım. 3 aylık periyotta burası kaymış. Şimdi aklımıza gelen sorular. Oranı neden yükseldi? Şimdi **neden soruları sormak** lazım burada ilk adım olarak. Soruyorum, burada bir bankada batık kredi oranı neden yükselebilir? Buradaki case'i, işte buradaki kredi case'ini alıp işte pazarlamada başka bir şeyde mesela uyduruyorum, atıyorum X kitlesine bir kredi verme durumu olsun veya e-ticarette X, Y bir şeyine bir kampanya çıkma durumu olsun, aynı. Orada da mesela kitle bulacaksın, bir şey bulacaksın, bir şey uyduracaksın veya bir kitledeki, segmentteki payın azaldı, işte onun neden olduğunu bulman için ilk ilk adım neden soruları sormak.

Batık kredi oranı neden yükselir? Burada birkaç tane durum olabilir. Sizin de aklınıza gelen şeyler varsa chat'ten gönderebilirsiniz ben gördükçe yorumlarım. Batık kredinin oranı neden yükselebilir? Düşünelim. Orada çalışan bir kural, bir şeyler **drift olmuştur**. İşte orada çalışan bir model varsa sistem **drift olmuş olabilir**, **data kaymış olabilir**, **popülasyon kaymış olabilir**, oradaki modelin başarısı kaymış olabilir. Çok fazla etken. Atıyorum hükümet bir şey... Hükümetin yaptığı bir şeyle oradaki kural eski kalmıştır. O kuraldan kaynaklı oraya çok insan gelmiştir son 3 ayda ve o insanlar bu oranı batırmıştır gibi. Biz bunu nasıl düzeltiriz? Okey, tamam anladık hani buranın sonucu **pipeline**'daki Y modelinden veya Y yapısından

kaynaklanıyor diyelim. Bunu şey yaptık, kavradık. Müşterinin puanlaması yanlış olabilir, işten çıkarmalar arttığı için çekilen krediler... Bunların hepsi olabilir bu arada. Resesyon, döviz kuru. Resesyon ve döviz kuru çok olası değil çünkü döviz kuru tarafında alınırken o döviz kuru genel olarak izlenir o bir metrik olarak yani onu düşünürler. Resesyonu da düşünürler. Yani o tarafta işte enflasyon diye bir şey varsa, fiyatlarda bir dalgalanma bir şey varsa onu anlayıp ona göre bir düzenleme yaparlar limitlerde diyeyim. Konjonktürel dalgalanma olabilir. Burada **threshold**'da bir sıkıntı olabilir gerçekten. Hani oradaki kuralın **threshold**'u, **cutoff**'u düzgün ayarlanmamış olabilir. Orada belki ufak bir ayarlama ile iş birazcık daha oranlar düzelebilir.

Okey. Hani bir şey bulduk diyelim, akışın bir yerinde Y kuralı veya Y modeli bir şekilde kötü çalışıyor. Bunu düzeltmemiz lazım. Şimdi bunu nasıl düzeltiriz? Soru şu, başvurularda batık riskini önceden nasıl tahmin ederiz? Şimdi bu problemi **machine learning yapmadan çözemez miyiz**? Politik değişim mesela, hükümet değişimi ya da... Onlar çok bariz böyle bir şey geliyorsa zaten ciddi derecede hemen üst yönetim anlayıp şey der; hadi buraları hemen kısalım, hemen gidip işte limitlerde bir şeylerde bir kısıntıya gidilir ki önceden minimize edilebilir. Performansı düştü, aşırı öğrendi model gibi. Model train edildi ve model canlıda çalışıyor. Aşırı öğrenme gibi değil de modele gelen data popülasyonunda bir kayma olabilir. Finansal terimlerden gideceksek, "Finansal Terim Sözlüğü" gibi bir hedefimiz var mı acaba? Bu tarafta söylediğim şeylerden ben bahsedeyim. Genel olarak çok bilinen şeylerden bahsediyorum ama birkaç tane daha böyle finansal terim kullanırsam açıklamaya çalışacağım diyeyim.

Okey, bir düşünelim buradaki yapıyı. Acaba mevcuttaki kural setini veya modeli, araya bir kural ekleyebilir miyiz? İşte hafif bir **karar ağacı** belki ufak bir şey çalıştırıp birkaç... Veya oradaki business insanlar bir şeyler bulup oraya kural ekleyebiliriz. Ne olabilir başka düşünüyorum? Bu tarz şeylerle aslında machine learning'e girmeden batık riskini önceden bilip veya oradaki batık riskini cutoff'la kesebiliriz diyeyim. Diyelim ki böyle bir çözüme gittik ve machine learning'e giremedik ve hızlı zaten aksiyon almak gerekiyor. Yani batıyor, verdiğimiz kredileri tespit ettik, orayı kıstık. Bir kural koyduk oraya veya o kuralın cutoff'unu düşürdük. Okey. Önümüzdeki periyotta izleyeceğiz tekrardan model hala veya işte oradaki kural hala kötü performanslı seyrediyor mu? Diyelim ki seyrediyor, düzeltilmemiş. Başka bir şeyler daha var. İş **machine learning**'e doğru gidiyor artık. Böyle sistemde ciddi bir **drift** var, onun düzeltilmesi lazım. Okey, machine learning'le çözmeye karar verdik.

Ben neden machine learning ile çözme konusunda da şöyle bir durum var. Diyelim ki bu modeli eğiteceğiz. Model eğitimi ciddi süre. Hani oradaki dataları tekrardan çekeceksin, işte tanıma periyodu... Bunu geliştirilmesi, **deployment**, **monitoring**... Ciddi maliyeti var. Oradaki insanlar da yaptığı işi bırakıp halihazırda belki başka bir şey geliştiriyorlar, bu modeli analize odaklanacaklar. Bunun sonucunda ciddi bir maliyet olacak. Aslında bu model, modeli yaptın, modeli canlıya çıkardın. Bu sefer modelin canlıdaki maliyeti var. Machine learning'e girdiğin zaman oraya basit bir kural koymakla kocaman bir model deploy edip çıkarmak arasında ciddi bir fark oluyor. Eğer problem basitse ve ML'e girmeden çözebiliyorsak, hiç girmesek daha iyi diyeyim. Okey, ML'e girdik, çözemiyoruz.

Ne yapmamız lazım? Şimdi ben gerçekten bir şirkette çalışıyorum, **Kaggle**'dan oradan buradan bir data almıyorum. Bir **target tanımı**, işte ne bileyim bir CSV dosyası yok. Bunların hepsini benim sıfırda, gerçek hayatta kendim bulmam lazım. Ne yapacağım? Bir **target tanımı** ve **target periyodunu** belirlemem lazım. İşte bu problem ne olursa olsun benim elimde bir **veritabanı** var. Veritabanında benim çeşitli tablolardan oluşmuş, ilişkili kolonlarım var. İşte bunlar ne olabilir? Customer tablosundaki benim kullanıcılarımın, müşterilerimin **demografik özellikleri** olabilir. Bir isim, soyisim, bilgilerine entegre nerede oturduğu, arabasının olup olmadığı, maaşı gibi bir takım demografik özellikler. Benim bankamdan aldığı krediler ve

bankadaki kredilerdeki performansını, yani geçmiş ödeme performansını içeren bir tablo olabilir. Gene Türkiye'de **KKB** diye bir yapı var, **Kredi Kayıt Bürosu**. Orada KKB'den gelen input'lar olur bankalarda ve bu tarz modellerde çok kullanılır. Bu aldığım data sette de benzer input var, fakat bu yabancı bir data, rastgele eklenmişler, tam nereden geldiği belli değil. KKB'den gelen benim input'larım, bir şeylerin, bilgilerim olabilir. Bir de benim kendim oluşturduğum bir takım değişkenler olabilir. Ve bunlar **domain bazlı**, içeride işte bir kampanya yapmışızdır, o kampanyaya kişi kullandıysa kullandığının bilgisi vardır. Kullandığının bilgisi vardır, kullandığı tutarın bir takım sonuçları vardır.

Şimdi okey, target'ı belirleyeceğiz. **Target** ne? Target, krediyi ödeyemeyen kişiler. Bunu nasıl belirleyeceğiz? Burada da bir terim var, **30 Plus, 90 Plus**. Kredi verdik, kredisi onaylandı ve kişi kredi kullandı, parayı bizden çekti. Bu kişinin bize karşı belli bir faiz oranıyla geri ödeme yükümlülüğü var. Gecikmeye düşmeden, gecikmesiz demek 0 gün, bu kişi gecikmeye düşmeden ödedi. İşte **30 Plus** demek 1 ay içerisinde bu kişi gecikmeye düştü. **60 Plus** demek 2 ay içerisinde gecikmeye düştü. Başlangıçta target periyodunu, sonra **train ve testin periyodunu** belirlememiz lazım. Bunun için de gidip de tüm data seti, işte tüm DB'yi çekip baştan train etmeye gerek yok. Bozulma zaten son 3 ayda. E o zaman ona yakınca benim train'imi, aralığımı oluşturmam lazım. İşte bunun için 10 aylık bir periyot alabilirim oradaki 90 Plus'ı da ve iyi bir bad oranı yakalamak için. Gene test için de işte bir aylık, iki aylık bir test periyodu alabilirim. Okey, periyodu ayarladık, işte target'ı ayarladık, burada **data leak** olmamasına dikkat ettik. **Data leak** diye bir kavram var, burada not alabilirsiniz. İlerleyen yayınlarda daha çok bunun üzerinde durulur ama yanlış bir periyotlama hatası, canlıya çıktığımızda her şeyin mahvolmasına sebebiyet verir.

Şimdi benim problemim belli, target'ım belli, eğiteceğim periyotlar belli. O zaman elimdeki bağımlı değişkeni açıklayacak **bağımsız değişkenleri** kocaman **DB** içerisinde bulmam lazım. Gerçek hayatta... Aynen. Şey gelmiş, kredileri veritabanı var mı? Kredi değeri verdi tamam, kredi değeri dediğin şey mi? Verilen kredi, verme, kullandı, kullanmadı gibi mi yani? Krediyle alakalı her şeyin log'u tutulur bankalarda. Takibe düşme **90 Plus**. Aynen. Genel olarak bu modellerde 90 Plus'ı target olarak alırız. Önceden bilelim yani bu kişi 90 Plus'a düşecekse biz buna öyle bir olasılık çıkaralım ki ya bunu reddedelim ya da çok minimal... Şimdi buradaki durumda böyle bir durum.

Data hazırlığına gelelim. Şimdi şunu düşünmenizi istiyorum. Normalde bugüne kadar ne yaptınız? **Kaggle** yarışmasına girdiniz, işte ne bileyim Kaggle'dan X, Y, Z birtakım platformlardan data setler aldınız. Her zaman sizin bir target'ınız belliydi, bağımsız değişkenleri kullanıp kullanmamak veya içerisinden yeni değişkenler türetmek tabii ki de size kalmış. Ama elinizde belli bir CSV, işte Parquet veya Excel formatında bir şey vardı. Ama iş gerçekten kocaman bir database geldiği zaman, benim onlarca tablom, binlerce kolonum olabilir. Buradaki yüzlerce, binlerce kolonu ben nasıl seçeceğim? Siz olsaydınız bu durumla karşılaşsanız gerçek hayatta, buradaki kolonları nasıl seçerdiniz? Bu bana bir mülakatta sorulan bir soruydu bu arada. Ne dersiniz? Burada hafif bir bekleyeceğim cevaplar için çünkü merak ediyorum nasıl cevaplar geleceğini. Zaten ekranda bir şeyler de yazıyor. Korelasyon diyen... Kredi skoring. Evet ya, kredi skoring veritabanı var. Burada bütün... Sorry, bir daha alabilir miyim acaba? Ya burada dediğimiz şey şu, işte senin problemin belli, target'ın belli. Problemin senin, kredi verdiğimiz ama geri ödeyemeyen kişileri tahmin etmek. Target'ın da takibe düşmüş kişiler. Bunun periyodunu ayarladık, train testi ayarladık. Şimdi bunu tahmin etmek için koca database'den bir takım bağımsız değişkenleri bulacağım, bağımlı değişkeni en iyi açıklayan. Bunu sizce binlerce değişken arasından, bir sürü tablo arasından nasıl bulacağım, nasıl seçeceğim? Bağımlı değişkenle **korelasyonu** en yüksek olanlara bakarız. Ama binlerce değişken var ya hani, nasıl korelasyona bakmak için de böyle sürekli o zaman bir data set çekip veya bir sorguda sürekli korelasyon mu alacaksın ki? Karar ağacı diyen... Business ekibinden öğrenmek istediğin kriterleri de eklersin.

**Enes Fehmi Manan:** Potansiyel kolonları nasıl seçeriz? Bir, burada **domain bilgisi** çok önemli. Burada devreye bu işi yapan **business**'taki insanlar giriyor. Business insanlar niye bu kişilerin bir takım nedenlerden gecikmeye düştüğünü zaten gözlemliyor. Bunu banka sistemi içerisinde veya başka sistemler içerisinde görebiliyorlar. Business'la iletişim halinde olmak, bu tarz bir model geliştirirken çok önemli. Onlardan çeşitli **feature**... feature isimleri... Hatta sen o koca şeydeki, DB'deki her feature'ı bilme ihtimalin de yok. Bazı feature'lar sadece business için bile oluşturulmuş olabilir, birtakım analitik ekip neticesinde. Senin feature'ların ne anlama geldiğini anlamak için de gidip işte business'a sorman gerekiyor. O yüzden burada domain çok önemli. Zaten buradaki işte çok domain bazlı bir... İşte bankacılık domaininde en azından birkaç sene deneyimli olacak, o bankada çalışmış olacak, işte belki geçmişte bu tarz kredi süreçlerinde birtakım roller oynamış olacak, falan filan. Birçok etken var. Sana belli başlı feature'ı bu business söyler. Bir, o feature'ları ararsın sen DB'de, önce bir onları çekersin. Ardından o feature'a benzeyen, kulağa mantıklı gelen bir takım feature'ları çekmeye başlarsın. Çekerken de şöyle bir durum var. Mevcut bizim train ve test aralığında besleniyor mu, ona bakarsın.

Şimdi elinde CSV varken elindeki **boş değerlerin** sayısına bakmak okey, hani boşsa at, bir şeyse at, onda bir sorun yok. Ama sıfırdan database'den bir data çekerken, ismen güzel olabilir ama baktın senin train test aralığında o kolon doldurulmasında beslenmesi bırakılmış. Neden böyle bir şey olabilir? O kolon kampanya için oluşturulmuş olabilir. Özel bir dönem için oluşturulmuş olabilir, sadece o dönem çalışmış olabilir. Mesela bu özel döneme örnek olarak deprem zamanı verilebilir. Türkiye'de çok büyük bir deprem oldu ve o dönemde özel krediler verildi. Onunla alakalı birçok mesela özel karar ağacı çalışıldı. O dönemki modellerin kolonları DB'de özel olarak tutuldu, o kolonları... Ama dönem bittikten sonra, yani işte üzerinden şimdi epey zaman geçti, artık o kolonlar mantıken beslenmemeye başlandı. O kolonlar işte korelasyon bakıyorsun dedin, korelasyon olarak yüksek gelse de belli bir noktadan sonrası boş geleceği için senin bu train'e baktığın zaman, mesela testinde boşsa kullanmanın bir alemi yok zaten de train'in de yarısına kadar geldi diyelim, o zaman da kullanma. Datayı çekerken fark edip bu data besleniyor mu, beslenmiyor mu veya bu feature geçerli mi hala, beslenen bir feature mı diye bunu da kontrolünü yapman lazım.

BDDK, TCMB sistemi eleştirmiş olabilir. Ya burada BDDK dedin mesela, BDDK işte **regülasyonlar** önemli bir nokta. Burada seçerken feature regülatif bazı noktalarda var. Birtakım feature'ları mesela kullanamıyorsunuz. İleride onlara da değinirim nedir bunlar diye. Modelde yüksek importance'a sahip de çıkabilir bu seçilen, orada da biraz böyle sağından solundan, belki bir şekilde regülasyona takılmayacak şekilde içeri koymaya çalışıyorsun. **Analitik yaklaşım**. Analitik yaklaşım dediğim şu, sizin de dediğiniz gibi işte **korelasyon** bakabilirim, işte bunlar ne kadar dolu, güncel mi geliyor onlara bakabilirim, birtakım başka analizler yapabilirim. Evet tabii ki buradaki yaptığım tüm analizler **SQL** base'de. Hani ben direkt olarak daha şey değilim. Birtakım kolonlardan bir şeyler ID'den birleştirip bir data set oluşturmaya çalışıyorum. Tamam, diyelim ki kabaca bir data set oluşturduk train için. Birtakım da faydalı olabileceğini düşündüğümüz değişkenleri de seçtik. Bunları da ayrı ayrı tablolardan getirdik. Bunlar da bizim tablomuzda var ama bir tane de oluşturduğumuz bir train var. Ve bu train ile baş başayız. Bir de testimiz var tabii ayrıca çektiğimiz.

Ne yapacağız? Burada **EDA** dediğimiz olay devreye giriyor. EDA dediğimiz olay ne? **Exploratory Data Analysis**. Bunun Türkçesi çok anlamlı olmuyor o yüzden EDA olarak devam edeceğim. Bunun keşifsel, keşifçi veri analizi gibi böyle garip bir Türkçesi var ama yapılan şey aslında analizi, birtakım dağılımlarına bakmak. Bunlar train-test'te uyumlu mu? İşte birtakım betimsel istatistikler gibi falan. Ne var bu EDA'nın içerisinde? Veriye ilk bakış. Burada ne olabilir? **Veri tipleri**. Veri tipleri dediğimiz olay çok önemli. Bir sürü kolon çektik DB'den ve bu çektiğimiz kolonlar işte bunu atıyorum Python kullanıyoruz, Python

kullanırken içeri düzgün import edildi mi? Düzgün import edilmediyse kolon tipleri, bizim gidip onu düzeltmemiz gerekiyor yoksa ileride en part-time dönüştürmelerde veya işte modele girerken yanlış sonuçlara sebebiyet verir. İleride ciddi baş ağrıtır. **Betimsel istatistikler**, bunlara bakabilirsiniz ama betimsel istatistiklere bakmak hani eşittir dağılıma bakmak aslında. Varyasyon katsayısı, burada varyasyon katsayısından kastım işte oradaki kolonun bir takım varyansı, ne kadar kolon değişiyor, bakabilirsiniz. Analitik yaklaşım içerisinde yani siz orada SQL'de datayı çekerken bütün kolon aşağı yukarı aynı sayıdan oluşuyorsa veya aynı kategori değişkenden oluşuyorsa bunu modele sokmak çok da bir anlamı yok çünkü bir ayırt ediciliği olmayacak. Onları eleyebilirsiniz böyle şeyler varsa. Sürekli ve kategorik değişkenlerin dağılımı. **Aykırı değerlere** bakabilirsiniz. Şimdi aykırı değer konusu aşırı kritik. Neyin aykırı olup neyin aykırı olmadığına karar vermek çok zor. Business'la gene dirsek temasında olup birtakım şeyler, kararlar alıp ona göre karar vermek gerekiyor. Bu aykırı değer konusunda da gene korelasyonlara bakabilirsiniz. Bakarsanız da bakarsınız.

Şimdi ardından da ben bu notebook'ta ufak bir **baseline model** göstereceğim. **Baseline model** dediğimiz en minimal ön işleme ile seçtiğimiz domain ve işte bir takım buradaki boş olmama veya işte bir takım yüksek korelasyonlu feature'larla basit bir ilk model çalışması. Bu model kategorik değişkenleri tahmin ediyorsan veya bir olasılık hesaplıyorsan **lojistik regresyon** olabilir veya sürekli bir şey tahmin ediyorsan lineer regresyon olabilir. Bunun bir alternatifi **LightGBM modeli** olabilir, o da çok hızlı eğitildiğinden kaynaklı. Burada ufaktan kütüphaneleri içe aktaralım. Ben şimdi burada kullandığım datadan da bahsedeyim. Bu datayı nereden buldum? Eğer siz nasıl bunu kullanabilirsiniz... Burada gördüğünüz structure biraz karmaşık gelebilir size. Burada ben tek bir notebook üzerinden gideceğim bu eğitim için. Bunu özel oluşturdum ama bir yandan böyle bir structure, bu bootcamp'in... Çarşamba günkü yayında bundan bahsedeceğim. Hani nasıl bir proje bekliyoruz. Örnek olarak da bu projeyi göstereceğim. Bu projenin büyük bir çoğunluğu zaten şu an GitHub'da bulunuyor. Benim şeyime hesabıma gidip oradan bakabilirsiniz. Credit Risk Model olması lazım. Oradan inceleyebilirsiniz.

Ben şimdi bunu nereden buldum? Hızlıca şuradan data bölümüne geleyim. Ben hazır geçmişte yapılmış bir yarışmaya girdim. Fontu bir tık büyütebilir misiniz? O kadar büyüttük ki hala... Ama şey ya, bir daha bir deneyeyim. Şu an nasıl? Bir filmde görebilirseniz çok iyi olur. Şu an güzel gibi hani öncekine göre. Web cam'i aşağıya alırsanız... Şöyle. Aynen. Tamam şu an daha net oldu diye tahmin ediyorum.

Şimdi bu datayı şuradan buldum. **Kaggle**'da competition "**Home Credit Default Risk**" diye bir yarışma var. Bu daha önce yapılmış bir yarışma. Ta 2018'de başlayıp gene 2018'de bitmiş. Geçmiş verilerde de verilmiş bir yarışma. Bu ev kredilerinde default edenleri tahmin etmek üzerine. Metrik **AUC**. Bu bayağı eski bir yarışma. Kazananları falan var. Gerçekçi. Siz de böyle geçmiş yarışmalara girip oradaki discussion'ları işte bir... gerçekçi datalarda bir şeyler build etmeyi deneyebilirsiniz. Yararlı oluyor diyeyim özellikle discussion'lar. Ben burada bayağı bir şey öğrendim hani bu projeyi yaparken. Data kartına bakalım. Data bayağı büyük, 3 GB'lık bir data var burada. Şey edilmiş hali, sıkıştırılmış. Şuraya geri dönelim.

Bizim burada aynı gerçek bir DB gibi bir takım ID'den birleşen bir DB şeması var. Burada birçok kolon var. Ben burada kolonların anlamlarını yazmaya çalıştım. Var da var. Biz burada sadece train'deki kolonlar üzerinden bir **baseline** çalışacağız ama repo'da daha detaylı incelerseniz ben diğer kolonlar üzerinden de çalışmalar yaptım. Tabii bu iterasyon da devam edecek. Repo linkini de eğitim sonuna doğru chat'e atarım. Böyle başlangıç bir aşamada proje, bir haftadır uğraşıyorum.

Okey, buradaki **train**'i kullanacağız. Buradaki bizim için hazırlanmış **test**'i kullanacağız diyelim. Buradaki test'i hazırlamak için de bir takım bu arada gerçek dünyada analizler yapılabilir. Bu analizler ne olabilir? Bankacılıkta **vintage analizi** olabilir,

kredinin geri ödenip ödenmediği tarzında bir şey. İşte **onay-kullandırım analizleri** olabilir. Kaç kişi onayladık, bu onayladıklarımız kullandı mı? Kullananların veya işte başvuranların onay oranı kaçtaydı? **Roll rate** diye bir analiz var, roll rate analizi olabilir. Bu **30-60-90 Plus**'taki insanların ne kadar bize borcu olduğunu gösteren bir analiz, bundan oranlarını görebildiğimiz bir şey. Bu analizlere göre biz burada train'imizi ve test'imizi oluşturduğumuzu varsayalım.

Train ve testte ne varmış? Train 122 kolon. Target geliyor. Test'in içerisinde target yok. Kaggle'a attığınız zaman otomatik sizin şeyinizi, test skorunuzu söylüyor. Siz train'in içerisinde bir **validasyon** ayıracaksınız. Şimdi ilk olarak dataya hızlıca bir bakış atalım. Burada **EDA** dediğimiz şey başlamış oldu. Şurası benim ID kolonum. Buradaki tüm değerler eşsiz. O yüzden bunu modele sokmayız. Sadece ama benim bir takım tabloları birleştirmek için kullanacağım bir kolon. **Target**'ım 1 ve 0'dan oluşuyor. Bu muhtemelen işin sonunda bir olasılık değeriydi. Bu olasılık değerini bir **threshold**'dan sonra 1, bir **threshold**'un altındaysa 0 olarak ata gibi bir yapıları var burada. Onunla flag'lemişler. İşin sonunda işte düştüyse default'a 1, düşmediyse 0. Default'a düşmek de konuştuğumuz gibi gecikmeye düşmek, **90 Plus**.

Burada şey var, "NAME_CONTRACT_TYPE" var, hangi type'tan krediyi aldı. Bunun... Erkek mi, kadın mı... Kategori kolonu. Arabası var mı, gene burada bir flag kolonu var. Çocuk sayısıyla alakalı bir kolon. Tehlikeli. Bu bilgiyi de iyi almışlar. Toplam geliriyle alakalı bir kolon var. Bu yıllık gelir olması lazım bu arada. İstenilen kredi tutarıyla alakalı, toplam kredi tutarıyla alakalı bir gene kolon. Goods price'la alakalı kolon varmış. Ya burada var da var. Hani bu kişi evli mi, bu kişi öğrenci mi? Bu kişi nasıl bir house type'ında oturuyor ya da talep ediyor? Apartman. Region population relative. Burada oturduğu yerin popülasyon popülaritesi oranı var. Çok enteresan bir feature var. Doğum tarihi, fakat DB'de bu doğum tarihi eksi olarak, yani işte her gün sayılacak şekilde artmış. Muhtemelen bir tarihten çıkarıyor gibi bir şey var bunun ya, sıfırdan... Öyle bir yapısı var, tutulmuş, bilmiyorum. Bunu organize etmemiz gerekiyor. İşte mevcut günden çıkarıp gerçek yaşa çevirmemiz lazım DAYS_BIRTH'ü. Bununla uğraşacağız, bu gözüküyor. Birçok flag değeri var. Çok var yani burada 122 tane kolon var. Biz bunları incele incele bitiremeyiz. Boş kolonlar var, bir şeyler var. Var da var diyelim, biraz ilerleyelim.

Value counts'lara bakalım. Şimdi en önemli şey... current day olabilir. Bizim en önemli kolonumuz ne? **Target**. Target bizim için epey önemli. Bu tarz problemlerde **target %99 dengesiz olur**. Çünkü zaten mantıken de default'a çok düşse banka batar. En minimalde, hatta ideali hiç düşmemesi. Öyle bir tahmin kurmamız lazım ki... Ama gerçekte tabii böyle bir şey olmuyor. Bir takım default oluyor ve buradaki default oranı bayağı bir dengesiz oluyor, buradaki gibi. Şimdi buradaki dengesizliği gidermek için birçok yöntem var. Ne yapabilirsiniz? İşte bunu araştırdığınız zaman çok saçma yöntemler de görebilirsiniz. En çok bahsedilen **sampling** yöntemleri oluyor. **Undersampling** yapılabilir veya işte... Yapılan bir modelin canlıya çıktığını da hiç görmedim o yüzden o **SMOTE**'u pek önermiyorum ama gerçekten yaptınız, uğraştınız, çok başarılı gelmiyor, birazcık oradaki bad oranını artırmak için belki periyottan kısıp öyle bir undersampling yapılabilir. Yani illa undersampling metoduyla değil de belki biraz train'in periyodunu kısıp oradaki bir takım good'ları azaltıp bir undersampling diyeyim. Çünkü sonucu prodda hüsran olur. Başka ne yapabilirsin? İşte **ağırlıklandırma** gibi işte birtakım modellerin parametreleri var, **regularization**... Birçok bunları deneme yanılma ile denemek gerekiyor. Buradaki oran fena değil bu arada, %8 bayağı yüksek. Normalde çok daha düşük olur yani %1, %2 falan.

Şimdi hızlıca bir value counts'lara bakalım. Benim float tipinde 64, integer tipinde 41, object tipinde de 16 tane kolonum var. Şimdi... Ehliyette örnekleme listesi. Yaşam süresi, krediyi ödeyecek kadar yaşar mı? Türkler olduğu için etkisi var. Burada güzel bayağı domain yorumları geliyor. Credit... ismini tam okuyamadığım kişi... Söyleyeyim, burada float 64 kolonları benim

sürekli değişkenlerime karşılık geliyor. İşte bu yukarıda bahsettiğimiz birtakım, aşağıda da göreceğimiz değişkenlerden. Bu integer 64'e gelenler benim sürekli değişkenlerime muhtemelen onlara karşılık geliyor. Object dedikleri de kategorik değişkenlerime karşılık geliyor. Çok hızlıca bunlar neymiş, bunların bir durumlarına bakalım. Ya şu zaten benim ID kolonumdu, unique. Bunda sorun yok. Burada yanlış bir şey varsa hızlıca bir göz gezdirip tipini düzelteceğim aslında. Şimdi Target'a bakıyorum, Target integer, okey. Diğerlerine bakayım. Object değişkeni var, bunları ben hızlıca taradım öncesinde, burada bir sorun gelmiyor genel olarak. Mesela şunlar KKB'ye karşılık gelen değişkenler. **External Source** değişkenleri ve muhtemelen o yurt dışında, bu datanın alındığı yerdeki birtakım skor puanları. Bunların önemi de bayağı yüksek geliyor. Bu skorlar şey olabilir, işte başka bankalardaki ödeme performansı veya kredi kuruluşlarındaki geçmiş ödeme performansı veya onların yaptığı bir scoring puanı olabilir, diyelim. Devam edelim. Şunlar zaten baktığımız zaman hepsi flag dökümanı ve hepsi integer 64 tipinde gelmiş. Şunlar saat, gününde, bunlar integer tipine gelmiş. Tamam.

**Eksik değerlere** bakalım. Eksik değer konsepti önemli. Eğer gerçekten yüksek bir eksik değer rasyosu varsa bunu... Çokça kaçırmadın ya... Yani başındaki senaryo biraz kaçtı ama olsun. Gerçekten eksik değer rasyosu yüksekse droplamakta fayda var. Çok fazla... gözden kaçırıp mesela eksik değerleri de gözden kaçırdık, içeri aldık değişkeni. Bu kadar... Birtakım değişkenleri birleştirmişim ID'den, gene hala eksik geliyor. Ya o zaman başımıza sıkıntı çıkaracağı belli. Şurada ne var? Birtakım böyle mod değerleri, aynı şeyin ortalaması, medyanı falan koyulmuş. Landing part... diye bir şey var. Ya bu değişkenlerin eksiklik oranı çok yüksek. Bunları atabiliriz. İlk baseline'ı kurarken almayabiliriz. Daha sonrasında nasıl doldurabiliriz onu ayrıca bakmak lazım. Ya şöyle baktığım zaman bayağı bir eksikliği olan bir veri seti. Eğer buradaki değişken işin sonunda, bir baseline kurdum mesela içeri aldım, importance'da önemli geldi, o zaman bakabilirim. Mesela ben bunları nasıl iyi doldurabilirim veya bunlar niye eksik geliyor? Sebebi benden kaynaklı veya işte business'dan kaynaklı bir şeyler ise, az önce konuştuğumuz gibi işte kampanya veya başka bir şeylerdir, o desek gene belki kaçırmış olsak bile onu gene iyi çıksa bile atabilirim. Çünkü devamında bir işime yaramayacak.

Oraya bir **ön işleme** yapmamız lazım demiştik. Ne zamandır bu kişinin çalıştığı vardı, o da bir garip gözüküyordu, ona da bir işleme yapmak lazım. **Numerical feature**'larımız var burada. İşte geliri, istediği miktar... good price, işte dışarıdan **scoring feature**'larım var. Bir de ne olabilir burada, region population relative oranı ve OWN_CAR_AGE arabasının yaşı ile alakalı bir feature var. Bunları seçiyoruz, topluyoruz. Şimdi burada ufak bir ön işleme işlemi, bunları **normalize** etmek kısmı var. Şu eksileri düzeltiyoruz. Ufak bir normalize burada da var. Daha sonrasında test ile birlikte bakıyorum. İstersek o data frame'i şuradaki feather metoduyla lokale kaydedebiliyoruz. Kaydetmedim ama isterseniz farklı feature'lara da bakıp hani bu şeyi sürekli, notebook'u sürekli çalıştırmak isterseniz bayağı kullanışlı. Sürekli sürekli bunu baştan çalıştırmak zorunda kalmazsınız. Kayıttan bakar.

Şimdi bunu neden **train'e, test'e birlikte görselleştiriyorum**? Bir önceki dersi hatırlarsanız, Engin Hoca'nın dersi. Engin Hoca derste şeyden bahsetti, işte train periyodu belirlemek ve test periyodu belirlemek ve oradaki **dağılım**, işte ortalamadan uzaklaşması... Şimdi ben train'de eğiteceğim, testte modeli test edeceğim. Train'de diyelim ki train'im elma, elmada eğittim ama testim armut, armutta test edeceğim. Mantıklı mı? Değil. Buradaki dağılımlar bize bunu söyleyecek. **Train'le test ne kadar birbirine benziyor** buradaki dağılımlar? Dağılımlar dediğimiz **olasılık yoğunluk fonksiyonları**. Age değişkenine baktığımız zaman genel olarak train ve test çok benzer. Yani neredeyse normal dağılım diyeceğim. Gerçekten çok güzel yaklaşmış. 20 ile 70 arasında, hani beklenen bir şey. Burada illaki outlier'lar vardır, birazcık ben uçlarını kestim bunun çünkü

buradaki sağ kuyruğu ve sol kuyruğu işte çok uzatıyor duruma göre. Gereksiz uzatmaması için kabataslak datanın geneline baktığımız için oradaki 99'luk kısmı kesebilirsiniz, sorun, problem değil.

Çalışma yılı var. Çalışma yılına baktığım zaman sağa çarpık bir şey ama buradaki dağılıma baktığım zaman gene birbiriyle oldukça örtüşüyor. Genel olarak krediye başvuranların da hani... o da enteresan yani yaş ilerledikçe çok bu home kredisi tarafında herhalde krediye çok başvuruluyor diye bir ufak yorum yapabilirim. Buradaki tepenin birazcık daha train'de fazla olmasının sebebi de buradaki yoğunluk olabilir. Yani daha fazla burada veri sayısı olabilir, bundan kaynaklı olabilir. Income total'e baktığım zaman burada hafif bir ortalama kayması var ama bu da çok problem değil. Buradaki dağılımda bir enteresanlık var yani birçok tepeden oluşuyor. Böyle böyle geliyor. Muhtemelen bu da yani insanların belli aralıklarla maaşını yuvarlak söylemesinden veya genel paternin böyle olmasından kaynaklı olabilir. Şu an burada anlık yorum yapıyorum. Ondan kaynaklı bu şekilde gelmiş olabilir ama gene baktığın zaman ben herhangi bir sorun görmüyorum. Üst üste denk gelmiş falan, gayet birbirine örtüşen bir şey.

İstenilen kredi oranı... Şimdi burada bir hafif farklılaşma var. %13,7'lik bir değişim var burada. Bunun oranında sıfır kayıp varmış train'le test'te baktığım zaman. Şimdi burada farklı olmasını beklerim aslında normalde. Özellikle test tarafında şu tarafta daha böyle bir yoğunlaşma var 350'lerde. Neyse çok sorun değil burası. Genel olarak baktığım zaman yani hepsine tek tek bakmaya gerek yok da şöyle bir baktığım zaman, özellikle dış kaynaklara baktığın zaman örtüşen bir data. Sadece mesela OWN_CAR_AGE'e baktığım zaman veya şuraya baktığımda şurada ekstra bir tepe var garip bir şekilde. Bu mesela bir şeyin ekstra habercisi ve flag'i olabilir. Bunu bir ekstra business'la bir yere sormak lazım çünkü şuradan şöyle bir tepecik oluşması bu kadar uzakta, bu normal bir şey değil. Bunu sormak gerekiyor. Burada arabası veya bir şeyi kendi üstüne olmayanları mı acaba sürekli bir şey mi yaptılar ya da nedir yani bunun sebebi, bu kadar uzakta böyle bir tane tepecik. Gene burada, burada da mesela biraz uzakta böyle garip bir tepecik oluşmuş. Bunların sebebi ne, business'la konuşup tartışmak lazım. Modeli böyle soktuğumuz zaman da bu tepecik bir şeyleri bozabilir. Onu düzeltmekte fayda olabilir.

Konut kredileri genellikle 30 yıllık süreler için başvurulur. Yaş arttıkça azalması gayet normal. Çünkü yukarıda belirttiğim gibi bankalar için yaşı ilerlemiş insan risk taşır. Aa kesinlikle ya o da bankadan bankaya değişir ya, yaşı ilerlemiş insanın risk taşıması. Genel bir gözlem olarak denebilir ama orada da işte senin diyelim ki yani kimsin? Maaşın yüksek yani istersen 70 olsun, sana gene de çıkar yani öyle söyleyeyim. Önemli bir takım... Çok zannetmiyorum ya şuradaki şu garip olayı missing'in %66 olmasıyla ilişkilendirir miyim? İlişkilendirmem ya. Bunun da kararını veremiyorum çünkü bunu da net olarak bizim bence incelemek lazım. Direkt ilişkilendirme ve ilişkilendiremeyeceğimizi... Emekli ikramiyesiyle alıyor mudur? Farklı dağılımları olan bir değişken olsaydı o değişkeni analizden çıkaracak mıydık? Çıkarmakta fayda var. Baseline yaparken veya işte modelin testle train'i çok farklılaşıyorsa, bir taraf aynayı bir taraf Konya'yı gösteriyorsa, train'in... Test'te de işte skora bakabilirsin, skoru artırıyor mu artırıyor mu veya düşürüyor mu. Ama genel olarak bir stabilite istiyorsak onu çıkarmak daha iyi olabilir. Çünkü feature kendi içerisinde zaten zaman içerisinde drift oluyor, sağa sola yamuluyor. Ona bakmak için de feature'ların stabilitesini görmek için de **PSI** dediğimiz bir şey var, **Population Stability Index**. Ekstradan feature seçerken ona da bakılabilir. Bu seçtiğim feature'ların zaman içerisindeki popülasyondaki stabilitesi nasıl işte geliyor? Onun bir skoru var. İşte %0-25'lik aralıktaysa iyidir gibi. O iyi PSI aralığındaysa hani zaten şöyle bir görsel yapmadan da sadece PSI'a bakıp da bir şeyler çıkarılabilir.

Bayağı bir yorum gelmiş. Hızlıca onlara bakmaya çalışayım. Kredi ödeme süresi azalır, bundan dolayı ilk 5 yıllık deneyimi olan genç neslin kredi alma şansı çok daha yüksek olur, ödeme süresi sorun olmaz. Emekli ikramiyesi diye bir şey yok. Xenetix bayağı bankacılık tarafında deneyimli biri olabilir. Her banka aynı davranmaz. Aracı emekli ikramiyesiyle alıyordur, ortadaki pik ondandır yukarıdaki. Yok mesela genetik oluşmuş. Maaşla alakası yok, bizzat deneyimle test edildi. Cumartesi bugün üzerinde saatlerce konuşabiliriz. Olabilir ya orada her bankanın kredi verme, feature yapısı farklı diyeyim. Burada tabii benim içinde çalıştığım kurumlarla alakalı bilgi vermek gibi olabilir o yüzden çok girmeyeceğim bu konuya ama her bankanın farklı feature'lara bakarak kredi verme stratejisi değişiyor. Burada sadece buradaki scoring modeli değil, işte stratejiler var. Oradaki stratejilerden çıkan bir takım limit hesaplama, cutoff'lar, bir şeyler var. Var da var yani öyle durumlar. Bankacı değilim ama kredi aldım ondan biliyorum. Makyaj değilim. Oradaki seçimlerin roll... Şeytan. Bu tarz keskin yorumlar yapmak için şey olmayabilir, iyi olmayabilir. Bu yorumları yapmak...

Analiz. Şimdi biraz da **kategorik değişkenlerin** nasıl değiştiğine bakalım. İçerisindeki durumlara bakalım. Burada neye bakabiliriz? Cinsiyet var, education var, buradaki ailesinin durumu var, income type var, occupation type var, name contract type var, vardı var, arabası var mı, evi var mı falan filan. Şimdi şunların hızlıca bir dağılımına baktığımız zaman, burada enteresan bir şey var mesela, ilk görsele baka baka. Kadın, krediye başvuran kadın daha fazla erkek başvuranlardan. Ben bununla alakalı da Xenetix'ten bir yorum bekliyorum. Niye kadınlar daha fazla başvurmuş? Bu da çok enteresan bir durum değil mi? Ev kredisi olduğu için mi acaba?

Burada education type var, ona bakalım. Academic degree, higher education, incomplete higher, lower secondary. Higher education bayağı düşük. Olabilir yani. Genel olarak da onlara baktığımda birbirine çok benziyor bu arada. Evliler çoğunlukla ev kredisine başvuruyor. Bu da beklediğim bir şey. Test'teki evli oranı hatta daha fazlaymış. Genel olarak çalışanlar başvuruyor. Yani çalışmayan bir kişinin gidip ev kredisine başvurması veya öğrenci birinin ev kredisine başvurması... başvurmamasından diyeyim daha doğrusu hiç şaşırtıcı değil. Bir takım kuralı... Buradaki meslekler var. Cash loan, revolving loans, kredi tipi. Bu analize de ilerlersek bu sırayla ilerliyoruz değil mi hocam? Grafik şekillendirme, tüm değişkenler analiz ediyoruz. Bu elindeki case'e göre değişir ama genel olarak böyle. Yani önce bir olasılık yoğunluk fonksiyonlarına bakarsın. Yani daha doğrusu veriye ilk bir bakış atarsın. Elinde ne var ne yok, missing değerlere bakarsın, işte o tarafta baktığımız betimsel istatistiklere bakabilirsin. Ardından data tiplerini bir düzeltirsin. Sonra da olasılık yoğunluk fonksiyonları, bir de kategorik değişkenlerle ufak ufak başlarsın. Bu da görselleştirmek çok önemli. Normal hani ekranda bir sayı görmekle bir görsel görmenin etkisi çok farklı.

Evliler, kadınlar üzerine krediye başvurdukları için kadınlar daha çok görünüyor olabilir. Muhtemelen yani kadın yüzü veya şey... evi kadının üzerine yapabilir düşündüm de... O da olabilir yani. Kadınlar üzerine başvuruyor olabilirler veya kadınlar daha çok ev almaya meyilli oluyor. Eşini böyle zorladığı için veya kadın direkt gidip başvuruyor olabilir. Durumu olabilir. Bunun tabii bu Türkiye'de datası olmadığı için tam şudur da diyemiyorum. Evliler daha çok başvuruyor çünkü sorumluluk ve mal edinme ihtiyacı yüksektir. Kesinlikle. Yüksek missing değere sahip kolonları almadık değil mi? Burada daha bir kolon seçimi yapmadık. Modele giderken... Şu an bakıyorum... Şurada missing değerlerini ben aldım, hani bu kolonları görselleştirmedim. Oraya bir ufak bir şey yapmamız gerekir. Ya dolduracağız, ya atacağız, ya da orada birkaç tane işte farklı farklı yöntem var, ona göre uygun bir yöntemi seçmek gerekir. Düşük faiz fırsatı varsa erkekler eşinin, annesinin, kız kardeşinin üzerinden kredi çekiyor olabilir bu. Ev kredisi olduğu için aslında şeydir yani... bir erkeğin kredi limiti düşüktür, işte kadının daha çoktur. Belki eşinin, kız kardeşinin üstüne dediğin gibi çekiyor olabilir Göker abi. Erkeğin büyük ihtimalle ya yorumlarda komik vardı ya, çok

doğru. Olabilir gerçekten de. Peki kodları aynı projelerde tüm... uyarlayabilir miyiz? İsim değişikliği vs. dışında? Ya uyarlayabilirsiniz kesinlikle. Gidişatı alıp, işte buradaki yapıyı tutup uyarlayabilirsiniz. İşte feature'ları değiştirirsin ama senin mesela yukarıda şey vardı, bir normalizasyon işlemi. Normalizledim, gerçek yaşa dönüştürüyorum. Onu senin age değişkenin o şekilde dağılmıyorsa yapmana gerek yok. Kendi notebook'una da uyarlayabilirsin diyeyim.

Burada önemli bir nokta, **target'a göre dağılımlarına bakalım**. Kredi skorunun da olabilir. Şimdi target'a göre dağılıma bakmak ne demek? Düştü, düşmedi. Ödeyebilecek, ödeyemedi. Burada ödeyememe durumunda, ödeyemeyen kişilerin kategorik değişkenlerdeki dağılımına bakalım. Buna neden bakıyoruz? Benim incelediğim yapıda default'a düşmüş olanları tahmin etmeye çalışıyorum. Bunların kategorilerdeki bir paterni varsa belki o paterne göre de bir şeyler dizayn edebilirim. Kafamda birazcık böyle... train, işte burada train değişkenin şey olduğu için, kategori bir şey olduğu için biraz daha kategorik feature'lar üzerinden inceliyorum ama mesela sürekli bir şey olsaydı biraz da bendeki değişik sürekli değişkenlerle de scatter plot'lar üzerinden inceleyebilirdim. Şimdi bakalım ne oluyor burada. Burada da enteresan yorumlar gelebilir. Şurası dönüştü kaldı, niye kaldı onu düşünüyorum. Name Contract Type, cash loan'a başvurmuşlardır, default'a olanlar. Kadınlar gene yoğunlukta. Default olma oranında da oradaki bir değişim değişmemiş yani. Mesela burada default olanların birazcık daha erkek olmasını düşünebiliriz ama oransal bir değişiklik olmadı. FLAG_OWN_REALTY, yes/no. no genel olarak... kadınlar başvurduğu için herhalde buradaki oranda kadın-erkek üzerinden benzer çıkıyor. FLAG_OWN_REALTY, yes. HOUSETYPE_SUITE_TYPE, unaccompanied. Şurası daima yoğunlukta çıkıyor. Partner var, 50 değil herhalde. Çocuklu. NAME_TYPE_SUITE, unaccompanied. Bu tanımlanmamış gibi bir şey mi? Unaccompanied, bunu tam ne olduğunu anlamadım. Default'a düşenlerde çoğunlukla unaccompanied. Çocuk, işler düşmemiş. Buraya bakayım, NAME_INCOME_TYPE, "Tek yaşayan" demek. Tek yaşayanlar daha çok default'a düşmüş, olabilir. NAME_INCOME_TYPE, working. Çalışanlar. Commercial associate, Pensioner, State servant, Unemployed. Çok enteresan, burada mesela Unemployed'in biraz oransal olarak yüksek çıkmasını bekleyebilirsiniz ama yukarıda baktığımız analizde Unemployed kişiler zaten krediye başvuramıyor, data içerisinde yoklar. O yüzden böyle bir oran olmasını beklediğim bir şey. Çalışanlar düşmüş default'a bu arada. Yetiştirememiş herhalde. NAME_EDUCATION_TYPE'a bakayım. Secondary. Buradaki şey de yukarıdaki dağılımla aynı. Burada birtakım evli, bekar durumları var. Evliler daha çok düşmüş ya. Single / not married, Civil marriage, Separated, Widow. Evli... ya bu muhtemelen data oranıyla alakalı. Çok fazla data içerisinde evliler başvurduğu için dominasyonda neresinden bakarsan bak evliler üzerinde oluyor. Büyütebilir misin? Ya burayı da büyütüyorum ya, bu görsel olduğu için maalesef... O zaten bitiyor. Genel olarak buradaki pattern de yukarıdaki pattern'e çok benziyor diyeyim. Burada mesela çok enteresan bir şeyler görseydik bunu böyle business'la da konuşup bir oraya özel feature veya işte birtakım bir şeyler üretebiliriz, şu anda net aklıma gelmeyen ama modele gittiğim zaman yeterli başarıyı sergilemediğini görüp, daha detaya dönüp gireceğim analizlerde bir şeyler çıkartabilirim.

Şimdi gelelim **korelasyona**. Şimdi korelasyonda önemli bir nokta... Baktığın zaman tabloya, korelasyonu yüksek bir şeyler görmüyorum. Kıpkırmızı bir şeyler çıkmadı veya masmavi bir şeyler. Bu bu arada kredi modelleri yaparken beklenen bir durum. Hani böyle çok yüksek korelasyonlu bir şeyler çıkmaz. Kendi içerisinde hani target ile koyu olmasını beklersin tabii de kendi içerisindeki korelasyonlara baktığımda... Şimdi yüksek olanlara bakayım, ilk gözüme çarpan 0,62 var. FLAG_DOCUMENT_3 ile EXT_SOURCE_3 arasında bir 0,62 korelasyon var. Ya çok yüksek değil ama **multicollinearity** durumundan belki düşünülebilir. Şöyle yüksek bir korelasyon alanı... Şurada 95 var. REGION_RATING_CLIENT_W_CITY ile REGION_RATING_CLIENT. Ya bu aynı değişkenin benzeri gibi bir şey. Biri 'with city', herhalde bu W'deki city ile birlikte

yazılmış hali olabilir. %95 korelasyon çıkmış, şaşırtmadı yani. Buradaki işlem ne olabilir? Aynı işlem... Mesela burada öyle bir şey var. APARTMENTS_AVG, APARTMENTS_MODE, APARTMENTS_MEDI diye bir şey var. Aynısının ortalaması, 1 korelasyon. Medyanı, ortalama değeri... Şimdi bunlardan birini atmak lazım. Bunlar bu arada ben bir ön işleme falan yapmadım, otomatik data içerisinde böyle geliyor. Yani DB içerisinde bulunan kolonlar diye varsayıyorum ben bunları. DB'de bir de ortalaması tutuluyormuş bu kolonun. Bir de işte geçmiş, ne bileyim, atıyorum 3 ayın toplamı tutulabilir, 3 ayın ortalaması tutulabilir. Bunlar akışta bir yerlerde kullanılıyor olabilir veya başka modelin içerisinde de kullanılıyor olabilir. Ama aynı değeri orada sokmaya kalktığım zaman, kolonların çok aynısı yani 1 korelasyon gelmiş. Bu **multicollinearity** durumuna işaret ediyor direkt olarak. İkisinden birini feature importance'a göre atabilirim veya atmayabilirim. Yani artık duruma göre bakacağız. Buradan sonra da artık baseline modele geçeceğiz. EDA ile alakalı kabaca bakılabilecek şeyler bu şekilde. Burada göstereceğim olan şey... -1 olan vardı. Multi-collinearity -1 olan dediğin? O tam anlamadım dediğini. Nüfus olarak büyük olduğu için evli insanların default'a düşmesi normal, aynen.

Şimdi... Okey. Yorumlar da güzeldi. Bu kısmı kapattıktan sonra ufak bir **baseline modellemeye** geçeceğim. Oraya geçmeden önce, şimdi bu EDA'ları yapmak için, fark ettiyseniz mesela şeyin de sorduğu soru vardı, "Biz bu aynı kodları kullanabilir miyiz?" Nurhan Hanım'ın. Bu kodlardan ziyade bu kodları otomatik çıkartan kütüphaneler var. İşte **Sweetviz**, **Pandas Profiling** var. Benim kendi yazmış olduğum bir tane kütüphane var, **elem-auto-eda** diye. Baseline'a geçmeden bir o kütüphaneler neler yapabiliyor, **oto EDA kütüphaneleri**, hızlıca ona bakalım. Ardından da baseline'a geçelim sonra da soru cevapla kapatırız.

Şimdi şuraya geliyorum. Şöyle bir pip install elem-auto-eda demişim. Benim kütüphanem. Bunu indirmek için... Şunun kodlarına da geleyim hatta. Şuradaki GitHub'dan kütüphanenin kodları direkt olarak burada, buradan inceleyebilirsiniz. Şimdi bu **auto EDA**, oto EDA kütüphaneleri neyi sağlıyor? Bu kütüphaneler otomatik olarak az önce gösterdiğimiz analizleri sadece data seti veriyorsunuz veya işte sadece seçtiğiniz feature'ları veriyorsunuz, sizin için otomatik bir HTML report formatında çıkartıyor. Benim yaptığım kütüphaneyi indirmek için pip install elem-auto-eda deyip otomatik indirebilirsiniz. Ben buraya aynı train'i attım, feature'ları belirledim hangileri olabilir diye. Bu LLM'li bir şekilde çalışıyor. Buradaki rapora bakıp ne demek olduğunu bunun bir konuşuruz. Bir de **Pandas Profiling** var, bunun ismini değiştirdiler **YData Profiling** yaptılar. Tabii birçok başka auto EDA kütüphanesi var ama en günceli bu diyeyim. İşte HTML raporunu kaydetmek için de şöyle bir kod yazmanız yeterli. Sadece böyle bir kod yazıyorsunuz ve size raporu veriyor. Ben raporları otomatik indirdim. İnen raporları hızlıca şuradan açayım. İlk olarak Pandas Profiling'inkini açayım. Bize nasıl bir şey veriyor? Az önce baktığım her şeyi bana veriyor. Ne kadar observation'ım var, ne kadar boş hücrem var, kaç tane... Target'ımın dağılımı... Şimdi burada mesela AMT_INCOME_TOTAL... Bir ölçeklendirme yapmadığım için şöyle garip bir şekilde dağılıyor. Bu, eğer veri setinde bir ön işleme yapmazsanız, mesela age değişkenini değiştirdik, o değişikliği yapmazsanız ve direkt olarak burada verdiğiniz zaman ki ben şu an öyle yaptım, oradaki ön işlemeyi yapmadım, bu şekilde biraz bozuk olarak çıkar. Yani doğru dönüşümleri yapmadan bu tarz kütüphaneleri kullanacaksanız da gene dataya bakıp illa dönüşümleri yapmak gerekiyor. Bunlar anlayıp dönüştürecek kadar akıllı değiller maalesef. Raporu otomatik olarak bize çıkartıyor. Şey değil, interaktif değil. Direkt ekranda bana böyle bir HTML report veriyor. Burada yaptığım tüm analizleri, işte birtakım betimsel istatistikleri ben görebiliyorum. Bu aşağı doğru uzuyor. Bir de mesela kategorik kolonları böyle word cloud yapmış. Güzel. Target'ı göstermiş, target'ın dengesizliğini istatistiki diyor. Interaction'ı seçebiliyorsun, birbirleri açısından nasıl bunların hareketi, onu görebiliyorsun. Korelasyonları görebiliyorsun. Şurada bir korelasyon çıkmış ama mesela şeyden dolayı anlayamıyorum yani, çok sıkıştırmış görseli. Boş

değerlerin oranını, sayısını görebiliyorsun matrix olarak. Heatmap olarak da göstermiş. Güzel. Bir de datanın en sondan bir bana first rows, last rows çıkarmış. Tıkladığım zaman da otomatik oralara gidebiliyorum. Şimdi "Alerts" kısmı var. Burada önerdiği bir takım... Aa bak bu güzel bir şey, high correlation'ı otomatik olarak çıkarıp sana bunlar arasında yüksek korelasyon var diye alarm vermiş. Neyle ne arasında? CNT_CHILDREN ile CNT_FAM_MEMBERS. Ya bu zaten aynı şey gibi bir şey. Normal yüksek korelasyon olması, birini atabiliriz. Atabiliriz. Bak, var da var bazılarında. Missing, aa bunda çok boş değer varmış mesela. Bunlar çok sıfıra müsait, ondan bahsediyor. Aynen, yaşlarda fazla sıfır var. Olabilir yani, çocuk... Bunlar çok sağa çarpık, onlardan bahsediyor. Güzel yani böyle bir raporu hızlıca alabiliyorsunuz. Bu neyi sağlıyor? Yukarıda yaptığım kodları yazmıyorsun da hızlıca tüm işte görselleştirmeleri, şeyleri bakabilmek için. Ama işte mesela train-test'i birlikte göremiyorsun falan gibi durumları var.

Şimdi bir de benim şeyin, elemin çıkardığı rapora bakalım. Çıkardığı rapor... Bu raporun boyutu büyük, biraz yavaş geliyor. Bu da bu kütüphaneyi de yapalım gerçekten bayağı oldu yani, 10 ay oldu. O yüzden biraz eskimiş bir şey. Geliştirecek, geliştireceğiz umarım. Bunun yaptığı şey de aynen işte, number of value, kaç tane kolon var, gene kategorik şeyleri gösteriyor. Bunun güzelliği **LLM** ile birlikte yaptığı için sana bir takım değerlendirmeler verebiliyor. Ve şuradaki dataya baktığınız zaman, buraya bir domain verebiliyorsunuz. Atıyorum ben buraya "Credit Risk" dedim. Buradaki tüm feature'ları credit risk domain'inde değerlendirebiliyor. Öyle bir güzelliği var. Hızlıca LLM bazlı bir feature değerlendirmesi almak için güzel olabilir bu kütüphane. Türkçe desteği yok maalesef ya, Türkçe desteğini getireceğim ama... Peki yüksek korelasyon çıkınca hangi feature'ı atacağımıza nasıl karar veriyoruz? Modele sokup hangisinin daha yüksek **importance** getirdiğine bakabilirsin. **Feature importance**'ına göre birini atabilirsin. Her zaman bu feature'ları atmak zorunda da değilsin, tutabilirsin. Business olarak bu feature'ların değerini bir değerlendirirsin. Ama genel olarak feature'lar birbirinin çok aynısı olduğu için yüksek korelasyonda atmakta fayda var modelin ayrıştırıcı gücü açısından diyelim. Feature importance'a göre birini atabilirsin. Onun için tabii eğitmek gerekiyor. Bunun güzelliği interaktif olması. Böyle tatlı grafikleri var. Bir de sana buradaki şeyleri yorumluyor. LLM olarak da default olarak GPT-4 Turbo modeli çalışıyor. Ama ilerleyen zamanlarda tüm modelleri getirmeyi düşünüyorum. Hatta birisi yakın zamanda pull request atmış, ben de inceleyeceğim. Detaya döneceğim, Mehmet Çelik hocam. Anlama desteği getirmiş, o da... Teşekkürler yani böyle katkılara da açığım.

Şimdi rapora tekrardan gelirsek, şimdi rapor daha deminki gibi birazcık daha interaktif. Oradaki her şey yukarıda burada da var ama bir de credit risk domain'inde bunun neden kullanılabilir veya işte default rate etkisinde olabiliyor olduğundan bahsediyor. Aynen bayağı 250k... Bak burada bir törpüleme gerekebilir. Mesela gerçekten bu modeli yaparken ben birazcık %99'luk kısmı kestiğim için burası çok belli olmuyor. Böyle bir rapor çıkartıyor size. Bunun da en altına gideyim. İşte kategorik değişkenleri böyle gösteriyor aynı şekilde. Korelasyon tarafı burada da bayağı batırmış ama interaktif olduğu için en azından şöyle gezerken görebiliyorum. Bu korelasyon için ayrı bir şey düşünmek lazım. Gerçekten feature sayısı artınca görsel karışıyor. Kategorik... bana göstermiş. Burada ekstradan **kutu grafiği** yapmış. Biz fark ettiyseniz hiç kutu grafiği bakmadık. Veriyi biraz normalize ettiğimiz için kutu grafiği bakmak çok mantıklı olmayacaktı. Ama zaten sağ tarafa veya sol tarafa çarpık... Yani çarpık dağılım... Şöyle bir kutu grafiği... Buna bakmak da çok anlamlı değil. Yani şuradan gerisini direkt outlier'dır diyemem ben. Onu da business'la konuşmam lazım. Gerçekten bu veri böyle dağılıyor olabilir, bilmiyorum. Hepsini böyle deneme yanılmalarla deneyip deneyip sürekli iteratif olarak sürekli modeli, feature'ları gezmem lazım yani. Direkt olarak şudur asla diyemiyorum machine learning'de veya credit risk modellemesinde. Bu böyle gidiyor tek tek. Bu raporda incelenebilir.

O zaman yeniden trenimize dönelim, noktamıza dönersek. Burada yavaştan **baseline çalışmasına** geçiyoruz. Baseline çalışması dediğimiz şey neydi? Elimdeki en basit feature'larla, en az, en böyle şeysiz ön işleme ile, efor sarf etmeyen ön işleme ile hızlıca bir model çalışmak ve modelin işte oradaki ayrıştırıcı gücüne bakmak diyeyim. Bu çok basit bir model olabilir. İşte **lojistik regresyon** olabilir. Ondan sonra, işte ben modelden sonra, baseline'dan sonra **feature ekleme deneyleri**, tarafında deneyler olabilir, **model değişimi deneyleri** olabilir, **model optimizasyon deneyleri** olabilir. Tek tek de benim bunları bir yere kaydetmem gerekiyor. Birazdan **MLflow**'un ne olduğuna bakarız.

Burada biraz yukarıda bahsettiğimiz boş değerleri olanları atacağım. Gerek yok onlara. Gerçekten önemli olanları almaya çalışacağım. Bunu seçerken de domain'den, işte buradaki çalışan business'taki insanlardan faydalanacağım. Bir de hani kendi geçmiş tecrübelerimden faydalanabilirim. Oradan gerisi de artık deneme yanılmaya kalacak. Şimdi burada şöyle bir durum var. Mesela veri seti içerisinde cinsiyet var, evli-bekar durumları var, education type var, işte ne bileyim children, çocuk sayısı var, bir şeyler var, var da var. Bu değişkenleri biz model içerisine alabiliriz ve bunlar **feature importance**'da veya **SHAP**'ta gerçekten çok önemli değişkenler de çıkabilir. Modelin tahmin gücünü ciddi artırıyor da olabilir. Ama domain ve business ile konuştuğumuz zaman, diyelim ki model çıktı canlıya ve deniyoruz. Bir kişi red aldı. Neden red aldığıyla alakalı bankaya mail attı veya Şikayetvar'da şikayet etti. Bizim bu kişiye bir cevap vermemiz lazım, şundan dolayı red aldı. Modelin red cevabına bakıyoruz, model diyor ki, "Sen erkek olduğun için sana red verdim." Veya diyor ki, "Senin evlilik durumun..." Veya diyor ki, "Senin eğitimin yetersiz, o yüzden red." Bunların hepsi gerçekten bir sebep, hani bunlardan dolayı sen red alabilirsin eğer modeli böyle kurgularsan. Ama bu şekilde bir dönüş yapmak banka istemiyor. Çocuğun, çocuk sayısından dolayı red almak... dönüş anlamında ve regülatif olarak çok etik olmayabilir. Bunlar regülatif olarak da bu tarz modellerde Basel falan var, böyle bir şeyler var, oralarda bunlardan da bahsediliyor. Bu değişkenler **unfair** olarak geçiyor. Bunları modelden çıkartabilirsiniz. Final modelde çıkarsanız iyi olur, başınızın ağrımaması için. Ama biz buradaki modelde, modelin içerisinde tutacağım. Tutmamın sebebi de hani bunlar önemli çıkıyor mu, çıkmüyor mu bakacağım. Zaten daha ilk baseline'a bakıyorum. Ondan sonra bir çıkartıp bakarım. Öyle öyle denerim yani. Final modelde olmayacağını biliyorum. Önemli gelecek mi onu da görmek istiyorum diyeyim. Önemli geliyorsa oraya bir takım sağından solundan dolanma, başka bir şeyler de belki yapmak gerekebilir. Yani scoring mesajının kısmında olmaz da pipeline'ın başka bir kısmında olur.

Şimdi 36 tane feature seçtim. Bunların 25'i numerik, 11'i kategori. Ne var kategorilerde? Sürekliler var. Var da var. Bakalım bunlar'daki boşluk oranı nasıl? Şu an EXT_SOURCE_1'de mesela boş değer varmış %56. AMT_GOODS_PRICE'da %19. Okey, bunları bir şekilde handle etmem lazım, modele böyle sokamam. Bir yandan da ilerliyorum. Şimdi validasyon ayrımına geldik. Burada **scikit-learn** içerisindeki **train_test_split** metoduyla bunu ayırıyorum. Bağımsız değişkenlerim ve bağımlı değişkenim. Klasik bir oran kullandım 80'e 20. Test size'ı 20 verdim, validasyon sayım. random_state'i set ediyorum. Ve burada görmediğiniz belki bir şey, **stratify** parametresi var. stratify parametresi ne anlama geliyor? Şimdi notebook'un en başına gidersek hatırlıyorsunuz, notebook'un en başındaki görselde, ilk görselimiz bizim target oranı dengesizdi. Şimdi biz bir train-validasyon seti ayırırken buradaki oran dengesiz olduğu için biraz dengeli şekilde dağıtmasını isteriz. Onu yapmanın yolu stratify etmek. stratify parametresini girdiğiniz zaman oranlar şu şekilde geliyor. Train'deki oranın 0.080, senin validasyondaki oranın 0.080. Oranı eşit şekilde bölmüş oluyor. Buradaki validasyon şeman, mesela atıyorum stratified k-fold, k-katlı bir validasyon şeması kuracaksındır, o zaman da her kattaki senin o oranda verileri dağıtmasını sağlar. Bunun olayı aynı oranda bölünmeyi koruması.

Şimdi geldik güzel kısma. Burada **pipeline** diye bir yapı var. Bu pipeline yapısı **scikit-learn** içerisinde klasiktir ve işleri çok kolaylaştırıyor. Burada bir bakalım ne yapıyor bu? Şimdi "numerical_pipeline" diye bir şey belirlemişim. Numerical feature'larımda benim boş değerlerim vardı ve aynı uzayda olmayan da çok değişkenim vardı. Nümerik feature'larım vardı. İşte bunlar neydi? Biri çok yüksek maaş değişkeni işte yıllık maaş, 250.000, işte ne bileyim 100.000 dolar, kocaman sayılar. Biri yaş. Gene yani işte... Basitçe bunu halleden **StandardScaler** var. Çokça boş değer vardı nümerik feature'larda da gördünüz. Bunları neyle doldurabilirim? Ben bunları değişken bazlı inceleyip işte ortalama, mod, medyan bir şeylerle doldurabilirim ama genel strateji işte... mod zaten şeylerde kullanılan, kategorilerde kullanılan. En büyük, en küçük, ortalama veya işte farklı bir takım değişkenlerle doldurabilirim diyeyim. Daha doğru oldu. Medyan seçmemin sebebi, ortalama değişkeni veya en büyük, en küçük değişkeni aykırı değerlerden çok etkileniyor. Bizim burada medyan seçerek aslında veri setinin ortasındaki, aykırı değerlerden etkilenmeyen değeri bulup onunla doldurmak, varyasyonu da bir nebze uygun noktada tutmak.

**KNNImputer** kullanılıyor mu? Basel... Hani orada Basel'le alakalı birçok regülatif noktalar var. O tarafları ben çok bilmiyorum. Ben daha çok modelleme tarafındayım. O Basel tarafında şirketlerde validasyon birimleri oluyor. Validasyon birimleri daha çok kontrol ediyor yapılan modelin Basel'e uygun mu değil mi. Kullanılıyor mu yapısını çok anlamadım ya. Ben kullanmıyorum. Bankalar uluslararası düzenleyici standartları... Aynen öyle. Şimdi kategorik kolonlar için neler var? Burada gene pipeline class'ıyla bir tane daha pipeline oluşturuyorum. Burada ne var? **SimpleImputer** var aynı şekilde. Burada bu sefer mod'la dolduruyorum, en çok gözüken değerle. Bunlar bu arada akla gelen ve bir miktar canlıda da skoru ufak düşürecek doldurmalar diyeyim, hareketler. Buradaki yapıların hepsi değişken bazında farklı farklı analitiklerle değişebilir, illa böyle olmak zorunda değildir yani. Bir de **one-hot** ediyorum. Burada pd.get_dummies işte veya işte LabelEncoder gibi şeyleri kullanmamak lazım. Genel olarak zaten datayı inceledik, çok fazla şey değişkeni yoktu, fazlaca sınıflı değişken yoksa ki kategoriklerde çok fazla varsa onu da gruplayabilirsiniz, one-hot kullanmakta fayda var.

Burada bir tane bunların birleştiği, kategorik ve numerik değişkenlerde processor fonksiyonu, class'ı oluşturuyorum diyeyim, **ColumnTransformer** ile. Burada yaptığım şey, kategorik feature'lara şunu uygula, numeriklerde bunu uygula. Yani git boşsa medyanla doldur, sonra standardize et. İşte git, kategoriyse mod'la doldur, ondan sonra da modelin anlaması için **one-hot encoding** yap yani 1-0... Modelim için processor'ımı oluşturuyorum. Final pipeline'ımı oluşturuyorum, buraya atıyorum. Ardından classifer'ımı oluşturuyorum. Bu da **Logistic Regression** ile en basit parametrelerle. class_weight='balanced' dengesiz olduğu için bunu giriyorum. max_iter'ı 1000 bırakıyorum, bu zaten default'u. random_state'i 42, şu da default solver değeri. Ardından pipeline'ımı tamamladıktan sonra artık modelimi fit ediyorum. Fit ederken şurada şöyle bir yazım tarzı var, %%time. Bunu eski çakallarda çok görüyorsunuzdur. Günümüzde artık çok kullanılmıyor. Bu senin hücrenin işte çalışma süresini gösteriyor. Eskiden Jupyter Notebook'larda şeyi göremiyorduk. Eskiden kalma bir şey o. İlk böyle 5-6 sene önce çok kullanılıyordu. Şimdi okey, çalıştı model. 19 saniye sürdü. CPU tarafında öyle bir süre var.

Şimdi bundan inference alalım. Bizim metriklerimiz kaç gelecek ona bakalım. Train **AUC**'m 0,74. Validasyon **AUC**'m 0,74. Çok benzer geldi, biri 55, biri 49. Bunların arasındaki açılma gittikçe model **overfit** oluyor olabilir, ona dikkat etmek lazım. Gene Kaggle'dan aldığımız bir data olduğu için ve yarışma dosyası olduğu için submission yaptıktan sonra, submission eğitiminden sonra Kaggle'a gönderip işte lead... public'te olabilir veya yarışma anında da public submission olabilir. Oradaki skora da bakmak lazım. Overfit olduysa model, oraya attığınızda buradakine farklı bir manzarayla karşılaşırsınız. Şu an ben baktığım zaman train validasyon arasında ciddi bir gap yok, neredeyse aynı. O yüzden modelin tutarlı olduğunu görüyorum. **Precision**, **recall** epey düşük. Recall özellikle 0.19. **Confusion matrix**'e bakayım. Confusion matrix dediğimiz şey de şu uçtaki değerler

doğrular, şuralara kaçanlar da hatalı olarak tahmin ettiklerimiz. **True Negative**, **True Positive** muhabbetleri var ya, onlar. Şimdi onların detayına hızlıca bir bakalım.

Link'i daha atmadık. **KNNImputer** olarak mı kullanılıyor mu genel olarak? Kenan'a mesaj. Çok güzel söyledin. **KNN Imputer** konusu enteresan bir konu. Küçük datalarda k-NN algoritması iyi olabilir ama k-NN algoritması CPU'da çalıştığından ve çok yüksek, nasıl diyeyim ona, çalışma algoritma mantığı açısından data büyük olduğu zaman k-NN'in eğitimi bitmiyor. Yani k-NN yapmak bayağı ölüm gibi bir şey sıfırdan model eğitmek yani. Onu dolduracağım diye... Ona ona bakmak yerine bu tarz simple imputer ile bir şeyler yapmak çok daha efektif gerçek hayatta. Bekleyecek, sabaha kadar bekleriz yani öyle söyleyeyim. Onu kendiniz de test edebilirsiniz elinizdeki data ile. Pek kullandığını görmedim. Öyle söyleyeyim. Kendim de denedim. Tek tek komşular arası mesafeleri hesaplıyor... Ya aynen ya, gerçekten mesafe bazlı bir algoritma olduğu için en yakın komşulara bakıyor tek tek böyle. Data büyük olduğu zaman iterasyonlar sonsuza doğru gidiyor ve CPU üzerinde çalıştığı için GPU desteği olmadığı için bitmiyor. Yani... Güzel, hızlıca ön işlemeyi bitirip de model train'inde vakit harcamak için o taraflarda simple şeyler kullanmak daha iyi oluyor. Bu şey için de öyle bu arada. Modeli canlıya çıkarırken, pipeline'ı canlıya çıkartırken k-NN'le doldurmak... Oradaki ön işleme senin ya sabit bir kod ya da eğitilmiş modelden inference alırken pipeline'ıyla birlikte modeli kaydedersin, öyle inference alacaksın. Oradaki k-NN'i çalıştırmak gibi bir şey mümkün olmaz. Zaten bir krediye başvurdun, anında sana cevap dönmesi lazım. Hızlı bir şekilde saniyeler içerisinde veya ekranda tuşa tıkladın... Orada "Dur ya ben bir k-NN yapayım da şöyle bir şey olsun" demek çok mantıklı değil. Basit yöntemlerle ilerlemek daha iyi. Nerede olabilir? Hazır, batch bir şey çalışıyorsa, hani bir süre sınırı yoksa, önemli aylık bir şey çıkacaksa falan o zaman kullanılabilir. Ama gene de hesaplama zaman maliyetinden dolayı çok kullandığını görmedim diyeyim. Bu konuya da bayağı uzun değindim.

Şimdi kaç dakika oldu? Saat 10:27. Güzel. Şunları bir yorumlayalım ya. Bunlar ne oldu? Şimdi bu model başarılı mı çıktı? Bu feature'lar iyi miydi? Bundan sonrasında ben ne yapabilirim? Gerçekten business'a bu modeli versem işe yarayacak mı? Sizce? Bakalım. Şimdi validasyona ayırdığı kısım... tane **True Negative** var. Benim iyi müşteri, modelin kabul ettiği, doğru. Şimdi bu şu kısımdaki müşteriler, default'a düşmemiş, krediye başvurmuş, almış. Model de bunlara "Evet bu kişiler gelmiş." Olasılık olarak threshold'un üstünde, yüksek bir olasılıkla bu kişileri ben, atıyorum, bu kişiler gerçekten de bu krediyi alabilir ve daha sonrasında da default'a düşmeyecek dediğimiz kişiler. Bunlar 38984 taneymiş ve model bunları iyi tahmin etmiş diyelim. Bilemediği kısım, gene iyi müşteri kitlesinde, 17554 tanesi iyi müşteri imiş, model reddetmiş. Burada ne olmuş? Bankaya veya işte kredi kurulumuna müşteri iyi bir müşteri, ödeyecek, buna default'a düşmeyecek önümüzdeki sürede ama model diyor ki "yok". Yani ve bu 17554 tane ciddi yüksek bir sayı. Oransal olarak baktığımızda neredeyse yarısı.

**False Negative** oranına bakalım. Şurası bizim için... Şimdi buradaki sayı çok yüksek. Buradaki sayı biraz az olsa iyi olurdu. Şuradaki sayılar daha önemli. Çünkü burada hem kredi veriyoruz hem de adam krediyi batırıyor. İki taraflı kaybediyoruz. **False Positive**'e bakalım. Riskli müşteri ama model kabul etmiş. Buradakilerin tamamı batacak. Yani şuradaki orana baktığım zaman da beni yarı yarıya diyebiliriz. Modelin batacak olanları bilme sayısına bakarsak, burada da riskli müşterileri evet gerçekten de bu model dediği de 3354 kişi var. Şimdi sizce, soruyorum size, bu model nasıl bir model? Yani iyi bir model mi? Bu modeli nasıl geliştirebiliriz? Bu modeldeki feature'lar buradaki default oranını açıklamak için yeterli miydi? Böyle bir sonuçla gerçek hayatta karşılaştınız, devamında ne yapardınız? Baştaki data science senaryosuna geri dönersek. Biraz... duracağım, biraz cevap bekleyeceğim.

Kovulduk bence. Cevapların gerçekten beni benden alıyor. Neler yapılabilir? Aşağıda diğer metriklerin anlamları var. İşte ayırt etme gücü, ROC'un altında kalan alan. Precision, recall'un nasıl hesaplandığı veya ne anlama geldiği... Precision'ı çok anlamlı yok bu arada. Recall açısından fena değil ama **F1-score**... Model default tahminde fena değil ama false positive oranı yüksek. Aynen öyle. 17.000 ciddi yüksek bir sayı. Yani yarı yarıya, genel olarak yarı yarıya yani. Default'u hani şurayı yakalaması çok iyi. Bir de şeye bakmak lazım hani gerçekten bu... Peki kaçak ne kadardı? Bu modeli implemente etsek ne kadar yakalıyor? Burada tabii ben yukarıdaki örneği mesela 3 aydan verdim. Çok mantıklı olmayabilir. Buradaki dataset çünkü bütün yılları kapsayan şekilde modelliyorum bunu sayı biraz fazla olsun diye. Gerçekte bu kadar fazla olmayabilir. Başka modeller, modeller, modellere de bakılabilir. Aynen, başka model deneyip karşılaştırılabilir. Kovulmadık diyelim, dedik olmasınlar ya. Daha yeni başladık yani uzun bir süreç. Bismillah yani, ilk datayı aldık hani, bir tane şey yaptık. Bayağı model denenebilir, denenebilir.

Şimdi ciddi derecede kötü durumdayız. Seçilen feature'lar nasıl? Buna bakarken de şurada eğitimimiz için, eğittiğimiz için **coefficient**'lar üzerinden bir importance bakıyorum. AMT_INCOME_TOTAL... Niye bu kadar yüksek gelmiş ya? INCOME_TYPE, ORGANIZATION_TYPE, DAYS_EMPLOYED... Kategorikler çok daha... EXT_SOURCE değişkenler bayağı aşağıda çıkmış, üçüncü sırada. Coefficient sıralamasında ve bayağı düşük EXT_SOURCE'un sırası. **SHAP** bakılabilir. Hedefin dağılımını normalize edebiliriz. Hedefin dağılımını normalize edebiliriz? Hedefin dağılımı normalize edilemez. Genel olarak target üzerinde bir işlem yapmayız bu işe... **Auto-ML** artabilir, ön işlemeye tekrar bakardım. Train-test oranını tekrar düzenlerdim. SHAP bakılabilir. Train-test oranını tekrar düzenlemek bir şeyleri değiştirir bu arada. Özellikle final modele bakarken test artık, validasyon seti tüm data üzerinde de train edebilirsiniz, o da bir yöntem. Dünkü derste train-val-test diye ayırın demişti hoca, o ayrımı yaptık. Yani bunu gönderdiğim zaman ben, logistic regression... Sonucunu da açayım mesela. Gönderiyorum, ilk lojistik baseline. Public'te 0,73, private'da 0,74. Baktığınız zaman lokaldeki ile Kaggle'daki aynı, o yüzden babalar gibi model.

Şimdi genel olarak okey, tamam. Yorumlar eğlenceliydi, güzeldi. Güzel şeyler de vardı. Benzer şeyler söyleyeceğim. İlk bakılacak olan şey aslında gerçekten de **model değiştirmek**. Logistic regression gerçekten çok simple. Hani bu tarz modellemelerde çok kullanılan bir model hem açıklanabilir hem de iyi. Ama artık **karar ağacı** bazlı, yapılan işte **gradient boosting**, **XGBoost**, **LightGBM** gibi modeller birazcık daha bu tarz modellemelerde öne geçmiş durumda şu anda. Hızlıca bir LightGBM deneyebilirim. Ben burada LightGBM ile, burada yaptığımız bir de çok... normalizasyon kısmı, hem de ön işleme bakmak... genel olarak o ön işlemeye bakmak lazım. Bayağı bir boş değer vardı ve ben o boş değerleri medyanla dolduruyorum. Normalizasyon da aynı. Gene aşağıda öyle... boş değerleri doldurma kısmı var. LightGBM'e geldiği zaman bu kadar ön işleme yapmam gerekmeyebilir. Logistic regression scale kısımlarına da hassas bir model diyeyim. Yani ona giderken scale etmek, işte boşları doldurmak falan gerekiyor. Ama ağaç bazlı modellerde, diyeceksiniz boşlukları biraz kendi handle edebiliyor, scaling'i... scale etmeseniz de kendi içerisinde bir şeyleri var, bir şeyleri düzeltiyor falan. Birazcık daha böyle uğraştırmadan daha iyi sonuca otomatik sizi götürebilecek bir LightGBM veya işte boosting metotları ilk deneme olarak onu deneyebilirim aynı feature set içerisinde. Bir de o modelin şeyine bakalım. Mesela AutoML diyen biri oldu. İlk bakarken aslında baseline'da AutoML de denenebilir. Burada şimdi baktık coefficient'a. Beklediğimden değişik geldi. Burada kategorikler birazcık daha yukarıda. Böyle bir şey beklemezdim mesela. Özellikle şey verirken, kredi verirken bir sürü feature var ama gerçekten bu feature'lar... O zaman modeli değiştireyim, bir de oradan importance bakayım. Tamam, modeli değiştirelim, bir de oradan importance'a bakalım.

Baktık, bu sefer baktık, gene başarı gelmiyor. Bu sefer ne yapıyorum? Mesela business'a gideceğim. Business'la biraz konuşacağım şöyle, böyle falan. Oradan aldığım döngülerle tekrardan feature setime döneceğim. İlk yapacağım iş buradaki **feature setini genişletmek**. Benim başlangıçta hızlıca modellememiz için küçük bir feature setinde ben çalıştım ama bu data 3 GB'lık bir data. Gerçekten kocaman, içerisinde sayısız, 10 üstü tablo olan bir şey. Bu var da var. Bu feature'lardan ben yeni feature'lar da üretebilirim. Stage by stage, more feature, aşama aşama... İlk yapacağım şey bu.

Nüfus sınıfı oransal olarak çok düşüktü. **SMOTE** ile sentetik veri oluşturulabilir mi? Maalesef oluşturulamaz. Özellikle işte bu dersin başında da konuştuk, **oversampling** yöntemleri gerçek hayatta pek tercih edilmiyor. Özellikle SMOTE gibi yöntemler hiç tercih edilmiyor. Çünkü zaten senaryoda datan var hani. SMOTE yapacağına o zaman bad oranını artır. Hani madem böyle bir problem var, periyodu biraz artır bad oranını, işte good taraftaki periyodu biraz ondan biraz törpüle. SMOTE'a asla gidilmez. Öyle söyleyeyim. Deneysel olarak denenebilir mi? SMOTE'la nasıl gelecek bir de görelim onu... denenebilir yani. Onu da track edebilirsin. Sonra karşılaştırırsın, bakalım nasıl geldi falan. Ama finalde asla onun olması söz konusu olamaz. Şimdi çok da kesin konuşmak... Şöyle bir şey de söz konusu olabilir. Modeli işte pickle veya bu tarz direkt formatlarda çıkarmasın da bir tane işte Random Forest veya şey, buradaki gibi Logistic Regression eğittim diyelim. Logistic Regression'da ne yaparsın? İşte oradaki formülü alıp sisteme entegre edebilirsin. İşte Random Forest'ta ne yaparsın? Belli bir... veya karar ağacında, bu sana belli bir **if-else kuralları** verir sana. O kuralları işte not alırsın, içeri dizayn stüdyoda veya işte bankanın sisteminde bir şekilde karar olarak kodlatabilirsin veya işte farklı bir yapı olarak kodlatabilirsin ama monitoring'i kararlar üzerinden gerçekleştirirsin. Oradaki senaryoda evet oversampling kullanılıp da belki oradaki karar ağacının şeye gittiği olmuştur ama model olarak, serialize edilmiş bir modelin gitmesi anca böyle çıkan kural setini gidebilir. Onu da zannetmiyorum.

Otomatik... şey yaptık. Doldurduk, neler log'luyorum burada? İşte parametreleri log'luyorum, kullandığım feature'ları log'luyorum, sonuçları log'luyorum. Diğer senaryoda ben artık gideceğim, başka bir şey deneyeceğim. X değişkeni veya Y modelini deneyeceğim. Tekrardan çalıştırırken şuradaki işte değişkenleri değiştirebilirim, bir de ona log atabilirim. Şimdi bu **MLflow** dediğimiz şey ne, nasıl gözüküyor, ona bakalım. Ben hızlıca şuradan gitmek için şuradaki... Oradan... CD içinde bulunduğum klasöre gideceğim, notebook klasörü, oradan da analize gireceğim. Otomatik oluştu. Bu oluşturduğum klasörü açacağım. Şimdi bunu açmak için de şuradan... Ben default olarak datayı, şeyi, code base'i maintain ederken uv diye bir şey var burada, uv'i varsa kullanıyorum. Bu uv işleri bayağı kolaylaştırıyor. Uvicorn'u başlatmam var bu arada. Onu da hızlıca inference almak için mlflow ui içerisine koydum.

Şimdi geldi benim **MLflow** ekranım. Şu default'ta, 30 yaşında default bulunuyor. Az önce eğittiğim modelin bilgileri de buraya gelmiş oldu. Şimdi az önceki değerleri burada, class weight'leri burada. Model burada, modelin model matrix... Assistant matrix... **Artifact**'ler burada. Artifact'te ne var? Confusion matrix, label encoder'ım... future... future ayrımları var. Ya bu feature olayı özellikle kritik. Hani modelin sonuçlarını bir şekilde track edebiliyorsun bir yere falan da future listesini track etmek harbiden bir eziyet. Yani MLflow bu açıdan ciddi işi kurtarıyor. Şimdi sen şimdi şeyler kuracaksın, **deneyler** kuracaksın. Modeli değiştirdin, yeni feature'lar ekledin. Buradaki feature sayın artık uçtu yani. Şimdi şu an var 30 feature, ileride olacak 100 feature. Ve sen buradaki her feature'ı bir yerde tutman lazım. İşte burada imdadına gerçekten her bu deneyimin sonucunu görmek için MLflow yetişiyor. Bunu da özellikle yapacağınız final projesinde kullanmanızı ben tavsiye ediyorum. Şurada ayrıca modele gidersek de... mesela şuraya geleyim. Burada benim modelimi de otomatik olarak tutuyor, pipeline

formatında eğitilmiş modeli de otomatik şey yapıyor, tutuyor. Eğitilmiş... Onu da tutuyor. Gerçekten mutlaka kullanın, kullandırın dedim.

Şimdi gel buraya dönersek, şurayı da kapatayım. Genel olarak bunu eğitmek için dediğim şeyler yapılabilir. Yani bunu güzelleştirmek için. Gelelim şeye, Kaggle'daki insanlar kaça kadar getirmiş buradaki skoru? Yani ona baktığımız zaman onun modeli de çok iyi ayrıştırıyor. Aslında atladığım bir farklar yok. Buradaki yarışma da biraz zorlayıcı bir yarışma. Her ne kadar 7 yıl önce de açılmış olsa. Ama credit risk'le ve genel olarak hani default modelleme gibi konularla ilginiz varsa bu data setine böyle bir girip uğraşmanızı tavsiye ederim. Hani ben bu Kaggle'ın... Final projesi örnek proje olması açısından biraz da kendim domain'de bildiklerimi böyle açık kaynak olsun, aktarayım diye uğraştığım bir şey. Benim şey de bayağı karıştı bu arada, notebook'larım da karıştı. Onu da zaman içerisinde iyileştireceğim. Onun linki de şurada. Bugünkü notebook daha buraya yüklemedim. Bunun linkini de ben chat'e atayım. Şurada bir To-Do var. Bu To-Do'ları yavaş yavaş geçip, benle işte oradaki Ar-Ge yaparak aslında modelin başarısını 0.80'e getirmeye çalışacağım. Daha buradakileri pek incelemeden sıfır yetmiş yedi, sıfır yetmiş sekizlere geldi diyeyim. Ve finalde bir de **reduction** yaptım. Oradaki reduction'da da 40 feature vardı ama o feature'ları business anlamında değerlendirirsek o kadar da anlamlı olmayan feature'lar vardı. Model anlamında çıkıp business anlamlı çıkmayan feature'lar da olabilir. Onu da işin sonunda bir daha bir döngüde dönüp bakmak lazım. Yani bir şeyleri de prod'a alamazsın. Gene niye red dediğin zaman, "Bundan dolayı red." "İyi de bu ne?" yani, falan. O yüzden business anlamında da değerlendirmek lazım. Tabii buradaki insanlar, Kaggle'daki insanlar... Yani özellikle bu hocanın yazdığı write-up var. Ee 700 feature varmış. Yani 700 feature ile bir kredi skoring modelini muhtemelen production'a almazsın. Business olarak iyi değil. Feature-feature yorucu olur. Böyle bir şey açıklanabilirlik açısından da çok tercih edilmez. Ama Kaggle'da skor yükseltmek için mübah. Hani denenebilir. O deneyleri de işte ne yapacaksın? Buradaki 9... Tek tek veya işte denediğin notebook içerisinde yazmak lazım. Hani sen ne yaptın? Feature engineering'de şu değişkenleri ekledim. Validasyonda şu kadar artış oldu. Kaggle'a submit ettim, bu kadar artış oldu. Tek tek bunları Google Docs'tan tuttum, Markdown dosya yapıp, Markdown...

Genel olarak benim anlatacaklarım bu kadar. Düşünüyorum, atladığım bir şey kaldı mı? Sanıyorum kalmadı. **MLflow** ya, çok iyi kesinlikle. Sizin sorularınız varsa soru cevap yapabiliriz. Ben yavaştan Göker abi buradaysa onu da yayına çağırabilirim.

**Göker Güner:** Vallahi ağzına sağlık. Düşeceğim. Eğitimin verimliliği, önemi, sektör... Yani burada hiç sektör verisine falan değinmeden EDA şöyle yapılır, böyle yapılır da diyebiliriz ama niye önemli? **Sektörel deneyim**... Çünkü sonuçta bu kadar şeyi öğreniyoruz, işte üniversiteden mezun oluyoruz ya da profesyonel olarak sektör değiştiriyoruz. Bir sektörde çalışıyoruz sonuç olarak. Perakende, telekom, bankacılık... Bu tip sektör deneyimlerinin... sektör deneyimlerine ihtiyaç duyulduğunu bilmek önemli. Birincisi bu. Bir ikincisi de data tarafı. Yani şimdi biz bu bootcamp kapsamında işte bir sürü eğitim vereceğiz. Supervised, unsupervised, bir sürü şey anlatacağız, machine learning'e dair ama yapay zeka çözümlerinin genelde %80'i, belki daha fazlası **datayı anlamak**. Datayı anlamadığın noktada modeli ne kadar iyi seçersen seç, modeli koy, olmuyor. Yani bir yerde sektöre çıktığın zaman, canlıya çıktığı zaman patlıyor. Datayı iyi anlamak, datayı iyi anlayıp dataya göre modeli kurgulamak, yanlış model seçimini ya da işte birkaç parametreyle ilgili böyle çok optimize olmamış şeyleri, çalışmaları nasıl diyeyim, telafi edebiliyor. O yüzden de veriyi anlama üzerine bu kadar vakit ayırmanın ben önemli olduğunu düşünüyorum. Ufaktan böyle Murat'ın sorusu var en son sırada. Proje hakkında bilgi verir misiniz? Çarşamba akşamı zaten yeteri kadar bilgi vereceğiz o konuda. Öncesinde proje yayınına onu anlatıyor olacağız. Düzenli katılımcılarımızın Tuğrul'un bir sorusu oldu. Feature sayısını geri azaltarak optimuma çekebilir miyiz?

**Enes Fehmi Manan:** Tam da onun sorduğu soruya ithafen bu ekranı açtım. Toplamda ben feature'lar ekleye ekleye, hand-crafted feature ekleme ve basit ortalama alma, işte bir rolling feature ekleyerek şeyin, AUC artışının nasıl olduğuna bakarak ilerledim. Detaylı feature engineering'e bir daha gireceğim. Bir de oradaki engineering notebook'u çok kendini tekrar ediyor, şey kısmında, selection kısmında hem de MLflow ve artifact'lerin log'lanması kısmında aşırı fazla tekrar eden kod var. Onları da helper'a alıp o notebook'u böyle bir rahatlatacağım çünkü çok uzamış durumda. Oraya geçersek... İşin sonunda şu anda finale geldiğin zaman, daha bir tane daha level atlatacağım ama hani... bunaltmamak için... Finalde 95 falan vardı feature. Train AUC'si 81, val AUC'si 77 gelmiş, 77.8. Önemli olan benim burada validasyon skorum. Burada ciddi bir düşüş yaşadım 40 feature'ı seçtim. Burada benim hem train zamanım düştü hem de ciddi derecede feature atmama rağmen, ne kadar feature atmış oldum? Kaç, 55 tane feature atmış oldum aslında. 55 feature atmış olmama rağmen işin sonunda benim validasyon AUC'mde atla deve bir fark oluşmadı. Dolayısıyla business'a ben model çıkartırken bu 40 feature'la modeli seçebilirim. Finalde de o 40 feature'ı seçtim. O 40 feature da şuna karşılık geliyormuş. Ardından modeli **pickle** olarak kaydediyorum. Sonra da modelime inference almak için işte source klasöründeki fonksiyona bağlıyorum.

**Göker Güner:** Mülakatlarda ne gibi sorular sorulabilir bu konu ile ilgili?

**Enes Fehmi Manan:** Sahibi örnek verdi yayının başında, bu soru bana işe girerken soruldu. Aynen, oradan değinebilirsin. Belki mülakatlarda ilk başta mülakat senin temelleri ne kadar bildiğin ile ilgileniyor. Yani işte **cross-validation**, işte **dengesiz veri setlerinde** nasıl modelleme yaparsın? Sonrasında şöyle bir şeyle karşılaştım, business sana şöyle bir şey söylüyor ama sen şöyle yaparsın mesela. Daha deminki soru gibi mesela, işte cinsiyet değişkenini modele alır mısın? Tarzında sorular gelebilir diyeyim. Bu tarafta mülakatla alakalı direkt ne söyleyebilirim... Tabular data ile alakalı, kredi risk domaininde soruyorsun değil mi, mülakatlarda ne gelebilir diye? Mülakat soruları gene bir düşünüyorum geçmiş mülakatlardan da hani direkt olarak "Şu soru da müthişti" gibi bir şey aklıma gelmiyor çünkü klasik sorular soruluyor ve laf lafı açıyor mülakatta. Yani sen bir projenle alakalı bir şey anlatıyorsun, karşıdaki teknik kişi de projenle alakalı seni challenge ediyor. Orada işte mesela... ben mesela bir mülakatta chatbot ile alakalı bir şey anlatmıştım ve bana şey sormuştu, böyle bir yapının başarısını nasıl ölçeriz, generative bir yapının? Ben de cevap veremedim pek. O dönemlerde generative AI yeniydi ve pek bir evaluation, bir şeyi yoktu, çok da üzerine düşünmemiştim. Gözle ölçerim yani. İyi cevap veriyor, vermiyor, deneme yanılma. Şu an mesela bir sürü evaluation üzerine generative AI'da, chatbot'ların, RAG sistemlerinin, bir sürü metrik var, bir şeyler var. Şu an bir sürü konuşurum ama o anda cevap veremedim mesela mülakatta. Net bir cevabım yok buna. Belki Göker abi bir şeyler diyebilir. Benim net bir şeyim yok.

**Göker Güner:** Mülakatlar için artık tek bir soru sorulmuyor. Soruyu ekrana veriyorlar... Ben eğitimde jargona çok yabancı kaldım, eğitimlere katılıp sağlam... Aslen inşaat mühendisiyim. Yazılım bilgin varsa nereden başlasam iyi olur? Bu bootcamp direkt zaten bunun için tasarlandı. İlk etapta SQL yayını, Python yayını... Python yayını gerekmiyor. Bütün yayınları tane tane izlerseniz, hatta biraz yani zihin dünyamızın da buna alışması için direkt veri kavramına giriş yayınından, klasik olmadan veri bilimi ekosistemi... 3. yayından başlarsanız bence tane tane oturacaktır yerine bir şeyler. Edanur, EDA ile ilgili bir yorumu var. EDA tutulması her yerde var. Bu aşamada ben de biraz aslında öyle yapmaya çalışmışım. Çünkü EDA dediğin şey... Daha onlara domain bazında yorumluyorsun. Ama bu EDA'ya gelene kadar önce bir problem geliyor sana, probleme yaklaşımın var, business var bir yerden, bir yerden maliyetler var. Daha burada konuşmadık yani, zaman baskısı var gerçek hayatta. Başarı baskısı var. Modeli... ne yapacağız yani? Buraya gelene kadar da ciddi emek verildi bu arada. O DB'de toplamalar, bir şeylerdir, var da var. Başlangıç... bu problem mesela elimizdeki data ile çözülemiyor ve bunu birileri öngörüyorsa onun

yorumu da çok değerli yani. Hiç başlamamak... Var da var diyeyim. Hani birazcık bunları anlatmaya çalıştım. Umarım faydalı olmuştur.

**Enes Fehmi Manan:** Modelde faydaları oluyor mu? Oluyor ya, özellikle Kaggle yarışmalarında bunun faydasını daha net görüyorsun ama gerçek hayatta modeli çıkartırken abartı, abartı feature artırımı yapmıyoruz diyeyim. Ya ne oluyor mesela gerçekte bir modeli yaparken, işte son 3 ayı toplayıp mesela toplamını alabilirsin. Ben de şu ekranı yukarı alayım mı? Şöyle kameraya doğru... Son 3 ayın işte toplamını alabilirsin. O şekilde bir şey, bir feature olabilir. Atıyorum son 6 ayın toplamını alırsın. İşte bir zamanlar bir zaman seçersin, o zamandakilerin ortalamasını alırsın. Bu tarz böyle DB'de aslında ay ay duran veya işte belli bir zaman zaman duranları birleştirebilirsin, skoru artırabilir, modelin anlamlılık seviyesini artırabilir. Bütün değişkenleri birbirine oranla, bütün onu ona çarp, bunu buna böldürürsün... Orada hiçbir problem yok. Ama gerçek hayatta birazcık daha ihtiyatlı, feature sayısını abartmamak önemli. Çünkü bunun monitoring'i sana kalacak yani yapan kişiye.

**Göker Güner:** Vallahi ilerlemek için Keras gibi bir kütüphaneyi öğrenmek gerekli mi?

**Enes Fehmi Manan:** Şey için gerekli değil. Kredi risk modelling için gerekli değil veya bankada çalışacak, özellikle kredi tarafında çalışacaksa gerekli değil. Çünkü **derin öğrenme modelleri regülasyonlardan** dolayı pek kullanılamıyor. Daha çok ya lojistik regresyon gibi modeller ya da karar ağacı bazlı, if-else kurallarını görebildiğimiz ve neden bunun reddedildiğini anlayabildiğimiz yapılar tercih ediliyor. Sen başka bir, mesela gene bankada çalışacaksın ama pazarlama tarafında veya atıyorum **NLP** tarafında bir şey yapacaksın, böyle tabular datadan ziyade, o zaman bunları öğrenmen gerekir. Ben burada kredi riski için söyledim. Veya işte gidip başka bir startup'ta veya şeyde...

**Göker Güner:** Yani genelde tabular datalarla çalışıyor, işte istatistiksel modellerle çalışacaksa **Keras**, **TensorFlow** bu alanda, machine learning alanında kullanılmıyor. Enes'in dediği gibi NLP olur, **computer vision** olur, o tip şeyler olur, onlar için gerekiyor. Yani **PyTorch** öğrenebilirsiniz, Keras, TensorFlow, o PyTorch çalışmanızda bir fark yok ama genel olarak bu alan için... Bir sorumuz daha var. Buna ben bir ekleme yapayım. İkimiz de analist değiliz. Buna ne kadar net bir cevap verebiliriz o yüzden bilmiyorum. İstanbul'da yaşıyorsan ya da cumartesi gelebiliyorsan ekibimizde analist arkadaşlar var. Direkt birebir görüştürebiliriz. Orada network alanında hani analistin bir günü nasıl geçiyor, nasıl bir bakış açıları var, direkt birinci ağızdan öğrenme şansın olabilir.

**Enes Fehmi Manan:** Analiz yapıyor yani. Sektörde özellikle iç içe geçmiş kavramlar diyeyim hani. Çok da böyle über ayrılmıyor sektörde. Sen de analiz yapabilirsin veya senin ekibinin sorumluluğunda bir şeyse, gene sen analiz yaparsın. Mesela şöyle düşünelim, modeli çıkardım hani canlıya. Modelin track edilmesinde bir şey sordular veya bir kampanya ile alakalı bir şey yaptın, modeli çıkardın canlıya, oradaki analizi gene senin yapıp onlara dönmen lazım. Bir gün nasıl geçiyor? Hani gün nasıl geçiyor? Gün, çalışma metodolojisine göre değişir. Hani ilk **Agile** çalışanlar bir takım toplantılarla, "Bugün ne yapacağız?" toplantıları yapıyor. Biz çok Agile çalışmıyoruz. Genel olarak işte herkesin devam ettirdiği işler var. O işlerle alakalı yapılacaklar bellidir. Bir şey, anlık toplantı olmuyorsa mail açma trafiği, öyle devam ediyor. Ben bir de uzaktan çalıştığım için çok şey olmuyor yani ofise gidelim.

**Göker Güner:** Sorulara devam etmeden önce sesli de belirteyim böyle yorulan, dinleyen arkadaşlar varsa yoklama formunu paylaştık. Biz soru cevap yaparken yoklama formunu doldurabilirsiniz diyeyim. Şöyle bir soru değerli sırada. "ML'de en

önemli olmazsa olmaz nedir hocam?" Yani machine learning öğreniyorum, "şunu öğrenmezsen olmaz" kesin dediğiniz bir şey.

**Enes Fehmi Manan: İstatistik** var. Hahaha. Ne olabilir? Temeller ya. Temel kavramlar yani. Temelleri... **Train-test ayrımı**. Train neydi, test neydi? Geçen eğitimde Engin Hoca'nın anlattığı her şey bence aşırı temeldi ve çok önemliydi. Onları iyi oturtmak lazım. Ondan sonra **validasyon** dediğimiz olayı iyi anlamak lazım. Validasyon şeması nedir, ne değildir... Bunları da nasıl öğreneceksin? Yani gerçekten öğrenmenin en güzel yolu **Kaggle yarışmasına girmek**. Çünkü orada gerçek bir interaktif süreç var. Yani seninle birlikte diğer insanlar da aynı problem üzerinde çalışıyor, bir şeyler deniyor. Discussion'ı çok hareketli. TR dilindeki için demiyorum ben, TR dilindekiler o kadar hareketli olmuyor ama yabancı yarışmalar özellikle... Öyle diyebilirim.

**Göker Güner:** Rasim'in güzel bir sorusu var. "AI Engineer deep learning bilmek zorunda mı?"

**Enes Fehmi Manan:** Sayılır mı? Deploy ederse...

**Göker Güner:** Sayılır. Şöyle... **AI Engineer** denen title Türkiye'de, dünyada şirketten şirkete değişebilen bir şey. O şirket AI ile ne yapıyor? Yani AI Engineer dediği aslında böyle bir uygulama geliştiricisi olabilir. Generative AI'dan bir API ile onunla birlikte bir generative AI çözümü de geliştirebilir. Bu senaryoda machine learning bilmene gerek yok. Ama deep learning'de çalışıyorsa, az önce bahsettiğimiz örnekler, NLP, computer vision çalışan bir şey ise AI Engineer'ları... o zaman onları bilmen gerekir mesela. Bilmene gerek yok. Ya da tam tersi böyle hani tahminleme gibi bir projelerde çalışıyor, perakende sektöründe, bankacılıkta bir yerde... O zaman da o tip engineer'lar arıyorlar Sayar. Dolayısıyla title değil de yani iş ilanı içeriği belirler.

**Enes Fehmi Manan:** Veya AI Engineer hepsi... Bana sana bir soru. "Junior olarak işe girmek için çoğu konuyu en detayına kadar bilmeye gerek var mı?" Ya kesinlikle yok ya. Burada çok ucu açık çoğu konu. Bir, en önemli konu temelleri bilmek. İki, iyi bir **portfolyo** olması. İyi bir portfolyo ne demek? En azından bir tane **uçtan uca bir proje** ve bence bir yarışmaya girip güzel bir sıralamada bir deneyim elde etmek önemli. Yani bu bir **Kaggle yarışması** olabilir, bir **hackathon** olabilir. Oralarda bir ufak bir problem üzerine inovatif bir çözüm getirip en azından finalist olup bir sunum yapmak çok değerli junior için diyeyim. Yani ben daha mezun olmadan bayağı TR geneli bir 10 tane falan sunum yapmıştım herhalde. Bayağı da bir dereceye girdiğimiz oldu.

**Göker Güner:** Veri bilimcinin de aynen, bilmesi gereken...

**Enes Fehmi Manan:** Temeller ya. Hani temelleri oturduktan sonra bir tane uçtan uca proje yapacaksınız. O proje de bizim size çarşamba günü vereceğimiz proje olabilir. Hakkıyla o projeyi yaparsanız, gerçekten hoşunuza bir proje. Birçok şey düşünmeniz gerekecek oraya. Özellikle birtakım sorular yazdım. Data şey... Repo içerisinde cevaplanması gereken, yazılı olarak bulunması gereken birtakım sorular var. Cevapları yazılı olarak. Onları zaten düşünürken kesinlikle ortaya geleceksiniz yani. 101'in ötesinde geçeceksiniz ona emin olabilirsiniz özellikle. Hani üzerine eğildik o projenin.

**Göker Güner:** Son bir soruyla kapatalım. Yoksa çok uzar o soru cevaplar. Her yayında yaparız gene böyle. Enes'in aslında az önce söylediği şeye bağlayacağım bunu. "Sektör sanki çok daralıyor gibi hissettiriyor. Stajyerden bile çok şey bekleniyor. Bir saatte... kendimizi veriyle alakalı alanlarda öne çıkarmak için neler yapabiliriz?" Enes'in az önce o bahsettiği "Ben üniversitedeyken Türkiye geneli 10 sunum yapmıştım" gibi detaylar artık önemli. Yani stajyerden beklenenler, sektörde dediğiniz gibi normali bu. O yüzden hani şu an bir üniversite öğrencisi olarak buradaysanız, çarşamba akşamı zaten

"Kendimizi öne çıkarmak için neler yapabiliriz?" sorusuna güzel bir... Ben nasıl hazırladım... Bence yani bir sektöre dayalı, sektörel bir proje geliştirmek, derinlemesine bir şeyler... Hakikaten ya, gerçek bir projeyle, Kaggle dataset'iyle değil de gerçek bir veri setiyle uğraşmak sizi öne çıkartır. Nereden çıkmanız gerektiğini de bileceksiniz. Onu da yine kariyer oturumlarında ya da işte yüz yüze oturumlarda konuşuruz. Büyük şirketler... artı 100 başvuru için söylüyorum, büyük şirketlerin işe alım programlarına uygun olmayabilir. Çok fazla sayıda başvuran var. Ama küçük şirketler, startup'lar vesaire denenebilir o proje geliştirme deneyiminden sonra. Selim Bey'in yorumuna...

**Enes Fehmi Manan:** Bence de kesinlikle yani... Dediğin şeylere katılıyorum abi. Sektörün hani daralmasına da katılıyorum. Burada da bir takım şirketlerin, bootcamp şirketlerinin ücretli, sektörde sanki çok açık varmış da yani herkes bu alana gelmeliymiş, burası dünyanın en popüler mesleğiymiş ve çok kazanıyormuş gibi bir algı yaratmasından kaynaklı. Biraz da herkesin bu alana girmesinden kaynaklı ufak bir değer düşüşü var. O 100 başvuru anında açılıyor ve bir saatte 100... Hatta 200, 300, 500 plus başvuru olmasının sebebi bu. Bunların önüne geçmek gerekiyor. Yani gerçekten değerimizi ortaya koymak gerekiyor. Orada da ortaya koymak için, yani yarışmalarda bir başarı elde etmek, bir çözüme, bir startup'ın, o şirketin çalıştığı çözüme inovatif bir bakış açısı getirmek gibi gibi...

**Göker Güner:** Aynen öyle. Yani çok öğrenme süreci uçtan uca ilerliyor. Yani bir yerden başlıyorsunuz, öğrene öğrene ilerliyorsunuz. Herkes kendi hızında ilerliyor, bunu unutmayın. Hani Berk'in cevabına cevap olarak, "Her şeyi bir anda öğrenmeliyim" diye bir şey yok. Kendinize o zamanı vereceksiniz. Herkesin zamanı da farklı. 3 ay, 6 ay, 1 yıl, 2 yıl. Enes şu an 1,5 yıl olmadı galiba değil mi?

**Enes Fehmi Manan:** Yok, olmadı. 1.9'da şu an.

**Göker Güner:** Yani ben hani 26 yaşında çalışmaya başladım. Enes 26 yaşında 4 yıllık deneyimli olacak belki bir yerlerde. Herkesin zamanı da farklı. O yüzden kendinize de o zamanı verin. Önemli olan öğrenmeye devam etmek diyelim ve yavaştan...

**Enes Fehmi Manan:** Ben de aynı şeyi söylerim. Genel olarak yeni başlayanların sorunu zaten devam etmemeleri, yeterince emek göstermemeleri. Ciddi emek vermek gerekiyor ya. Hani gece gündüzlü, bu işi sevmek, hani buna ciddi vakit harcamak, sürekli bir şeyler denemek, sürekli yarışmaları takip etmek... Çok önemli yani. Bu basit bir alan değil. Yani data science dediğim zaman şu an popüler generative AI'sından tut, işte derin öğrenmesine, oradan tut da ne bileyim veri ön işlemelerine, bilmem nelerine, NLP'sine... çok geniş bir alan. Her şeyde uzmanlaşmak da mümkün değil. Ama ciddi vakit ayırman lazım buna. İyi olmak için, sıyrılmak için bir portfolyo yapman lazım. Portfolyo yapmak için de deneye deneye... Yani gidin benim hani GitHub'a da bakabilirsiniz, önce geçmişteki repo'lara bakın. Hani günümüze geldikçe daha böyle şey yapmış oluyor. Birazcık daha kod yazım tarzları, repo yapıları, commit'ler falan değişmeye başlıyor. O da zamanla... Bence bir sene kendini kasmaları lazım data science'a başlayan arkadaşların. Şimdi bak şurada yanlış anlaşılabilir bu arada. Hani o bir seneden önce iş bulamaz gibi bir şey sakın anlaşılmasın. Yani bu dün başlayan kişi yarın da bulabilir. Yani orada da bir şanslı, bir şey var ya, ona da ben bir ekleme yapayım.

**Göker Güner:** Bir senede de bir şeyler... Belki sizin için 11. ayda... Öyle şeyler de yok değil ya. Öyle şeyler de var ama hani bir sene gerçekten elle tutulur bir süre istiyorsanız, bazen insanlar öyle şeyler istiyor, bir sene ortalama bir süre olarak düşünülebilir bence. Kapatalım mı?

**Enes Fehmi Manan:** Olur olur. Vallahi ben de hafiften yoruldum ama güzel bir yayın oldu. Biraz kredibiliteler ettikten sonra herkese anlatmaya çalıştım. Umarım güzel bir yayın olmuştur.

**Göker Güner:** Çarşamba akşamı o zaman projeyle... Projeyle ilgili sorularınızı alın gelin. Nihai cevap olacak hepsi. Çarşamba akşamı görüşürüz.

**Enes Fehmi Manan:** İyi akşamlar.