KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

YÜKSEK LİSANS TEZİ

DESTEK VEKTÖR MAKİNELERİ İLE OTEL ÖNERİ SİSTEMİ

NURSULTAN TURDALIEV

KOCAELİ 2018

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

DESTEK VEKTÖR MAKİNELERİ İLE OTEL ÖNERİ SİSTEMİ

NURSULTAN TURDALIEV

Doç. Dr. Ahmet SAYAR		
Danışman,	**Kocaeli Üniversitesi**	.....
Doç. Dr. Sevinç İLHAN OMURCA		
Jüri Üyesi,	**Kocaeli Üniversitesi**	.....
Doç. Dr. Mehmet Sıddık AKTAŞ		
Jüri Üyesi,	**Yıldız Teknik Üniversitesi**	.....

Tezin Savunulduğu Tarih: 28.06.2018

ÖNSÖZ VE TEŞEKKÜR

Bu çalışmada müşterilerin memnuniyetini arttırma, otel rezervasyon sistemlerini yeni seviyeye yükseltme amacında kullanılabilecek otel öneri sistemini makine öğrenme yöntemleriyle geliştirilmeye çalışılmıştır. Makine öğrenme ver seti olarak seyahat acentelerinin yardımıyla toplanan otel karşılaştırma sonuçları kullanılmıştır. Çalışmanın sonunda otel öneri sisteminin bir makine sınıflandırma algoritmasıyla otomatikleştirilebileceği kanıtlanmış ve 0,85 doğruluk puanındaki model elde edilmiştir. Geliştirilen sistem müşterilerin rezervasyon yaptıktan tatil başlaması arasındaki en uygun otelin otomatik seçilmesini sağlar veya öneri olarak sonradan sunulmasını sağlar.

Çalışmalarıma yön veren ve hazırlama aşamasında her türlü desteği veren danışman hocam sayın Doç. Dr. Ahmet SAYAR'a, TripRebel GmbH şirketindeki meslektaşlarıma teşekkürlerimi sunar ve bana her zaman destek olan aileme teşekkürler ederim.

Haziran – 2018 Nursultan TURDALIEV

İÇİNDEKİLER

ÖNSÖZ VE TEŞEKKÜR i
İÇİNDEKİLER ii
ŞEKİLLER DİZİNİ iii
TABLOLAR DİZİNİ iv
SİMGELER VE KISALTMALAR DİZİNİ v
ÖZET vi
ABSTRACT vii
GİRİŞ 1
1. TEMEL KAVRAMLAR 5
1.1. Makine Öğrenmesi 5
1.1.1. Denetimli makine öğrenmesi 5
1.1.2. Denetimsiz makine öğrenmesi 6
1.1.3. Yarı denetimli makine öğrenmesi 6
1.2. Kullanılan Algoritmaları 7
1.2.1. Destek vektör makinesi (SVM) 7
1.2.1.1. Biraz kompleks yapıldığında 8
1.2.1.2. Biraz daha kompleks yapıldığında 9
1.2.1.3. Ayar parametreleri: çekirdek, düzenlileştirme, gamma ve marjın 11
1.3. Makine Öğrenim Sıralaması 13
2. İLGİLİ ÇALIŞMALAR 15
2.1. Skyline Sorguları 16
3. SİSTEM TASARIMI VE UYGULAMA 18
3.1. Veri Toplama Sistemi 19
3.1.1. Sistem arayüzü 19
3.1.1.1. Seyahat acenteleri için arayüz 19
3.1.1.2. Süper yönetici arayüzü 20
3.1.2. Veritabanı yapısı 23
3.1.3. Kullanım durumu 24
3.1.3.1. Süper yönetici 25
3.1.3.2. Ajans uzmanı kullanıcısı 26
3.1.4. Sınıf diyagramı 27
3.1.5. Kullanılan araç ve teknolojiler 28
3.2. Otel Öneri Sistemi 28
3.2.1. Veri hazırlama süreci 28
3.2.2. Uygulama 35
3.2.2.1. Scikit-learn 35
3.2.2.2. Kodlama 36
4. SONUÇLAR VE ÖNERİLER 44
KAYNAKLAR 46
EKLER 48
KİŞİSEL YAYIN VE ESERLER 56
ÖZGEÇMİŞ 57

ŞEKİLLER DİZİNİ

Şekil 1.1. Siyah daireleri ve mavi kareleri ayıran bir çizgi çizin 8
Şekil 1.2. Siyah daire ve mavi kareleri ikiye ayıran örnek çizgi....8
Şekil 1.3. Bu düzlemde bir ayırma çizgisi çizebilir misiniz?....9
Şekil 1.4. zy ekseninin çizimidir. Ayırma burada yapılabilir....9
Şekil 1.5. xy düzlemine geri dönüştürüldüğü zaman 9
Şekil 1.6. Veri sınıflarında örtüşme olması.....10
Şekil 1.7. Birinci çözüm....10
Şekil 1.8. İkinci çözüm....10
Şekil 1.9. Düşük düzenlileştirme....11
Şekil 1.10. Yüksek düzenlileştirme....12
Şekil 1.11. Yüksek gamma....12
Şekil 1.12. Düşük gamma....12
Şekil 1.13. İyi marjin örneği....13
Şekil 1.14. Kötü marjin örneği 14
Şekil 2.1. Skyline örneği 17
Şekil 3.1. Sistem akış diyagramı 18
Şekil 3.2. Otel karşılaştırma arayüzü....20
Şekil 3.3. Süper yönetici ara sayfası....20
Şekil 3.4. Kullanıcı listesi arayüzü....21
Şekil 3.5. Kullanıcı ekleme arayüzü....21
Şekil 3.6. Dizide bulunan otellerin listesi....22
Şekil 3.7. Otel ekleme arayüzü....22
Şekil 3.8. Otel koleksiyonunun örneği 23
Şekil 3.9. Dizi koleksiyonunun örneği 23
Şekil 3.10. Otel karşılaştırma sonuçları koleksiyonu 24
Şekil 3.11. Kullanıcılar koleksiyonu 24
Şekil 3.12. "Süper yönetici" rolündeki kullanıcının "kullanım durumu" diyagramı....25
Şekil 3.13. Ajans uzmanı kullanım durumu diyagramı....26
Şekil 3.14. Sistemin sınıf diyagramı 27
Şekil 3.15. Süper yönetici sınıf diyagramı 28
Şekil 3.16. Biçimlendirmemiş veri formatı. MongoDB veritabanında 31
Şekil 3.17. Otel karşılaştırma sonucu MongoDB veritabanındaki yapısı 34
Şekil 3.18. Veri seti 36
Şekil 3.19. Karşılaştırma sonuçları....36
Şekil B.1. SVC algoritmasının kodu....53
Şekil 3.21. SVC algoritmasının en basıt uygulanması 38
Şekil C.1. Rastgele veri ayırması 54
Şekil D.1. SVC algoritmasının değişik kernel fonksiyonlarıyla uygulanması 55

TABLOLAR DİZİNİ

Tablo 3.1. Biçimlendirilmiş veri görünüşü 30
Tablo 3.2. Veri özellikleri üzerinde ölçekleme işleri 30
Tablo 3.3. Karşılaştırma sonuçlarının makine öğrenmesi algoritmasına uygunlaştırmış hali 34
Tablo 3.4. SVC kernel fonksiyonları sonuçları 38
Tablo 3.5. Rastgele veri ayırması doğruluk oranı istatistiksel bilgi 39
Tablo 3.6. Doğruluk puanı ortalaması 39
Tablo 3.7. C=10 ve gamma 1,0 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu 40
Tablo 3.8. C=10 ve 1,0 olduğu zaman ortalama doğruluk puanı 40
Tablo 3.9. C=10 ve gamma 0,005 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu 40
Tablo 3.10. C=10 ve 0,005 olduğu zaman ortalama doğruluk puanı 40
Tablo 3.11. C=10 ve gamma 0,0025 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu 41
Tablo 3.12. C=10 ve 0,0025 olduğu zaman ortalama doğruluk puanı 41
Tablo 3.13. C=10 ve gamma 0,0017 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu 41
Tablo 3.14. C=10 ve 0,0017 olduğu zaman ortalama doğruluk puanı 41
Tablo 3.15. C=10 ve gamma 0,0012 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu 41
Tablo 3.16. C=10 ve 0,0012 olduğu zaman ortalama doğruluk puanı 42
Tablo 3.17. Sınırlı özellikler kullanılarak elde edilen modelin doğruluk oranı 42
Tablo 3.18. Sadece yıldız sayıları ile eğitilmiş makine öğrenme modeli.....43

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar

API : Application Programming Interface (Uygulama Programlama Arayüzü)
BSD : Berkeley Software Distribution (Berkeley Yazılım Dağıtım)
CSV : Comma Separated Value (Virgülle Ayrılmış Değer)
PHP : PHP: Hypertext Preprocessor (PHP Hypertext Ön İşlemcisi)
RBF : Radial Basis Function (Radyal Temel İşlevi)
REST : Representational State Transfer (Temsilî Durum Aktarımı)
SVM : Support Vector Machine (Destek Vektör Makinesi)
UML : Unified Modeling Language (Birleştirilmiş Modelleme Dili)

DESTEK VEKTÖR MAKİNELERİ İLE OTEL ÖNERİ SİSTEMİ

ÖZET

Otel rezervasyon yöntemleri gelişen teknoloji ile birlikte sürekli yenilenmektedir. Rezervasyonda en önemli problem kullanıcıların en az para ile en çok memnuniyet verebilecek otellerin tespitidir. Bu tür problemlerin çözümünde son yıllarda gelişen makine öğrenmesi tekniklerinin kullanıldığı görülmektedir. Ancak mevcut sistemler, statik yapıda çalışmakta ve otelleri belirli aralıklarda puanlamaktadırlar. Rezervasyonlarda bu ölçüt anahtar rol oynamaktadır. Ancak, müşteriler oteli seçip rezervasyon yaptıktan sonra, eğer diğer otellerdeki fiyatlar ve diğer şartlar değişirse müşteriler fırsatları kaçırmaktadırlar ve dolayısıyla bu durum müşterilerde memnuniyet kaybına neden olmaktadır.

Bu çalışmada, müşterilerin memnuniyet/fiyat oranını yükseltmeye odaklı makine öğrenmesi tabanlı otel öneri sistemi geliştirilmiştir. Önerilen sistemde, müşteri rezervasyonunu yapmış olsa dahi, tatilin fiilen başlamasına kadar, daha iyi fırsatlar olduğunda müşteriye öneriler sunacak ve dolaylı olarak müşteri memnuniyetini ve otel doluluk oranlarını artırıcı bir etki yaratacaktır. Sistemin işleyişi makinenin eğitilmesine dayalıdır. Bunun için, çeşitli özelliklerine göre birbirine üstünlükleri bilinen oteller ile sistem eğitilmiştir. Daha sonra, sistemde olmayan bir otelin diğer otellerle otomatik olarak karşılaştırılıp daha iyi fırsatın oluşup oluşmadığı otonom ve dinamik olarak tespit edilmiştir. Karşılaştırma ve sınıflandırma işlemleri Destek Vektör Makinaları (DVM) algoritması ile gerçekleştirilmiştir. Eğitim verileri yaklaşık 680 satırdan oluşmuştur. Bu veriler, 1154983 otel üzerinden rastgele seçilen ikili eşleştirmelerin seyahat acentelerine gönderilip, karşılaştırma ve sonuç bilgilerinin manuel girişi ile elde edilmiştir. Sistemin başarımı, gerçek otel verileri üzerinde test edilmiş ve 0,85 doğruluğunda makine öğrenme modeli geliştirilmiştir.

Anahtar Kelimeler: Denetimli Öğrenme, Makine Öğrenmesi Sıralaması, Otel Sıralaması, Öneri Sistemleri, Sınıflandırma.

HOTEL RECOMMENDATION SYSTEM WITH SUPPORT VECTOR MACHINES

ABSTRACT

Hotel reservation methods are being upgraded with development of a technology. The most important problem in hotel reservation is proving that customers with least money get the most qualifying hotels. Machine learning methods are being used to solve this problem in recent years. However, the existing systems are working in static structure and scoring hotels in a limited interval. This criterion plays a key role in reservations. However, once customers select and book a hotel, if prices and other conditions change in other hotels, they are missing out on opportunities and this causes customer dissatisfaction.

In this study, a machine learning based hotel recommendation system focused on raising the customer satisfaction / price ratio has been developed. In the proposed system, even if the customer makes a reservation, he/she will get new suggestions if there are better opportunities, and this will indirectly increase customer satisfaction and occupancy rates. The operation of the system is dependent on the training dataset of the machine learning model. Then system was autonomously and dynamically tested whether there is a better opportunity or not with an external hotel successfully. Comparison and classification operations were performed with Support Vector Machines (SVM) algorithm. Training dataset contains approximately 680 rows. This dataset was obtained by sending random two hotels from 1154983 hotels to travel agents to compare them manually. The performance of the system has been tested on real hotel data and a machine learning model has been developed with an accuracy of 0,85.

Keywords: Supervised Learning, Machine Learned Ranking, Hotel Ranking, Recommender Systems, Classification.

GİRİŞ

Günümüzde otel rezervasyonunu telefon aramasıyla ya da katalog tarama yoluyla yapan çok az insan kalmıştır veya hiç kalmamıştır. Çünkü internet servislerinin gelişmesiyle otel arama, rezervasyon yapma işleri de iyice gelişmiştir. Bu alanda dünyaca tanınan birçok şirket kurulmuştur.

Var olan platformlar genelde benzer çözümler sunmaktadır. Neredeyse hepsinin tek amacı vardır, o da müşterinin oteli rezervasyon yapabilmesidir. Üstelik çoğu zaman tek amacı budur! Rezervasyon yapma olasılığını yükseltme amacında pek çok çalışmalar, araştırmalar yapılmıştır. Örneğin, önceden kalmış müşterilerin otele verdiği yorum ve değerlendirmesi çok önemli bilgidir. Otel sahipleri de genelde olumsuz yorum ve değerlendirme almamak için en yüksek seviyede hizmet göstermeye çalışırlar. Turistik şehirlerde otellerin sayısı çok olduğu için, arama yapıldığında müşteriye onun rezervasyon yapmasının maksimum olasılıkta olması için sıralama neredeyse tüm platformların en çok geliştirmeye çalıştığı alandır. Bu alanda çok akademik çalışmalar yapılmıştır. Örneğin, Nishith Khantal ve arkadaşları Expedia.com platformu için satın almayı yükseltmek amacında makine öğrenmesi yoluyla otel sıralama sistemini geliştirmiştir [1]. Ryosuke Saga ve arkadaşları ise, "Hotel Recommender System Based on User's Preference Transition" adlı çalışmasında Tokyodaki müşterilerin geçmişteki rezervasyonlarını kullanarak, model tabanlı yöntemle (model-based method) müşterilerin otel geçiş grafiğini kurarak problemi çözmeye çalışmıştır [2]. Michael Arruza ve arkadaşları Expedia şirketinin Kaggle websitesinde yaptığı otelleri en uygun sıralama yarışmasının sonucunda elde ettiği ortalama doğruluğu 0,02260 olan sonucu 0,30 oranında yükseltmişlerdir [3]. Anindya ve arkadaşları otel sıralama sonucunu arama sorgusuna göre en uygun değil, müşteriye göre en uygun şekilde sıralama yöntemini geliştirmişlerdir [4/18]. Otel sıralama sonucu bir iş seyahatcısı için başka, turistler için başka olması daha etkili olduğunu kanıtlamışlardır. İş sehayatçıları fiyata en az duyarlı, turistler ise en çok duyarlı geldiği öğrenilmiştir. İş seyahatcısı için otelin toplu taşımaya kolay erişimli, otoyola yakın olması çok önemlidir. Tersine, turistler için sahile yakın olması, hizmet seviyesinin en yüksek olması önemlidir. Aynı otel sırlama sonucunu değişik

ülkeler için değişik sırada göstermemiz daha etkili olabileceğini Elena Marchiori ve ardakaşları ”Harvesting Online Contents: an Analysis of Hotel Reviews Websites” çalışmasında kanıtlamışlardır [5]. Araştırmanın amacı otel sıralamasını daha iyi yapmak değil, aynı ülkeden gelen turistlerin otel değerlendirme yapmasında uyumlu yorumlar, değerlendirmeler yaptığını öğrenmektir. Bu sonuçları kullanarak potansiyel müşterinin geldiği ülkeye göre kişiselleştirmeyi daha bir adım daha iyileştirmek mümkündür. Otel sıralama, puanlama, karşılaştırma işleri yapmak için en önemli gereksinimlerden biri de otelin hangi özellikleri daha değerli olduğunu bilmemiz gerekir. Bu bilgi veri üzerinde ön işleme, dönüştürme ve seçim yapma işlerinde çok faydası vardır. Bazı özelliklerin veya olanakların hiç faydası olmadığını önceden bilmemiz veri seçiminde daha doğru karar vermemizi sağlar. Bu alanda araştırma işleri hangi aşamada, nasıl gelişmeler olduğu hakkındaki bilgileri Sara Dolnicar ve T.Otter “Which Hotel attributes Matter? A review of previous and a framework for future research” çalışmasında detaylı bilgiler sunmuştur [6]. Sonuç olarak otel özellikleri önemi geniş ve son derece heterojen araştırma alanı olduğu öğrenilmiştir. AltexSoft’un veri bilimi bloğunda yayındaldığı makalede ise makine öğrenme tekniklerinin otel ve sehayat servisinde nasıl kullanıldığı ve geliştirirken ne tür zorluklarla karşılaşıldığı, değişik şirketlerin otel rezervasyon işlerini kişiselleştirmeyi nasıl uyguladığı hakkında detaylı bilgi verilir [7]. Sonucu kişiselleştirmede özel verilerin etkisi çok olduğu için ilk önce müşterilerin kişisel verileri paylaşma isteği olması ve izin vermesi gerektiğinden, bazen yasal problemler üzerinden çalışılması gerektirdiğinen bahseder. Demek, müşteri verisi üzerinden kişiselleştirilmiş önerilerde bulunmak, kişisel sıralama uygulamak için yasal sorunları da göz önünde bulundurmamız lazım.

Günümüzde oteller, oda fiyatlarını, doluluk ve kârlılığını sürekli optimize eden modern sistemleri kullanıyorlar. Yapılan analize göre, on otelin sekizi konuk gelmeden, son otuz gün içinde, otel fiyatlarını birçok kez değiştiriyor. Sonuç olarak, gezginler için iyi fiyata en iyi otelin rezervasyon yapılması zorlaşıyor.

Otel rezervasyon yapma işi zorlaştığı için, gezginler de davranışını buna göre değiştirmiştir. Gfk’nın Google ile birlikte yaptığı araştırmasına göre, Almanyalı gezginler bir otel teklifini seçmeden önce 24 gün sürekli internette geçirir. Buna rağmen, gezginlerin yüzde sekseni otel odalarına çok fazla öderler.

Trivago gibi portaller bu problemi çözememiştirler. Aslında, onlar arama zamanındaki en iyi fiyatları gösterir. TripRebel, diğer taraftan, piyasa fiyatlarının ve değişen fiyatları izler. Eğer rezervasyon yaptıktan sonra otel fiyatı düşerse, önceki yapılan rezervasyonu iptal ederek, yeni rezervasyon yapar. Sonuç olarak, Triprebel müşterilere geleneksel fiyat karşılaştırma portallerinen on kez daha tassaruflu harcamayı sunar.

Fiyatlar düştüğünde aynı odayı yeniden rezervasyon yapmak kolaydır. Triprebel, eğer aynı veya düşük fiyata daha iyi otel rezervasyon yapabilme seçeneği varsa, otel değiştirme seçeneğini sunar. Bir otelin başka otelden iyi veya kötü olduğunu nasıl öğreniriz? Otel kalitesi ölçek üzerinde çalışmayan sübjektif bir alandır. Bu problemi çözmemiz için otelleri otomatik karşılaştırma yapabilecek otomatik otel öneri sistemine ihtiyacımız vardır.

Üstelik, otomatik otel öneri sisteminin doğruluğunu gün geçtikçe arttıran sistemi geliştirmek isterdik. Aşamalı doğruluk iyileştirme yapabilen otomatik otel öneri sistemi geliştirebilmek için, sistemi makine öğrenme teknikleriyle geliştirerek sürekli doğru verilerle eğitmek gerekir.

Tüm var olan sistemler para işleminden sonraki adımlarda çok değişik veya karmaşık çözümlerde bulunmamaktadırlar. Örneğin, müşteriler rezervasyon yaptıktan sonra aynı paraya daha iyi otel rezervasyon yapabilir miydi veya aynı tipteki oteli daha ucuz paraya rezervasyon yapabilip bilmeyeceklerini öğrenmek isterdi.

İki türlü veri seti vardır: birincisi otel özellikleri, Tripadvisor sitesinden kullanıcı değerlendirmeler sayısı ve ortalaması, ekincisi seyahat acentelerinin yardımıyla elde edilen otel karşılaştırma sonuçları.

Otel özelliklerini içeren veri setinde aşağıdaki veriler vardır:

- Otel ID
- Yıldızlar
- Kullanıcı oyları (TripAdvisor veya TrustYou)
- Aralık (şehir merkezinden, denizden, vb.)
- Olanaklar (spa, jimnastik, vb.)

Otel karşılaştırma veri setinde ise:

- Otel ID – karşılaştırılıyor olan otel

- Otel ID – ile karşılaştırılan
- İyi Otel ID- iki otelin en iyisinin ID'si
- Seyahat acentesi numarası – karşılaştırma yapan seyahat acentesinin numarası

Bu çalışmada var olan sistemlerden değişik çözüm sunmakla kullanıcıların memnuniyetini arttırmak amacında otomatik otel rezervasyonu yükseltici sistemi geliştirmeye çalışılmıştır. Genelde iki durumda sistem otomatik otel yükseltme işini yapar. Birinci durum, eğer aynı fiyata daha iyi otel bulunduğunda kullanıcıya iletişim kurmadan oteli yükseltir. İkinci durum, ise aynı kategorideki otel daha düşük fiyata bulununca, rezervasyon yapılmış otel iptal edilerek yeni otel rezervasyon yapılır. Ayrıca geri kalan para tamamıyla müşteriye geri ödeme yapılır.

Çalışmanın sonunda elde edilen bulgular TripRebel GmbH şirketinin kurulma amacının temelini oluşturacaktır. Başka bir değişle otel rezervasyon tecrübesini daha bir yüksek seviyeye çıkaracaktır. Bu zamana kadar tüm otel rezervasyon servisi sunan şirketlerin müşterilere sunduğu sadece rezervasyon yapma işine kadar ise, TripRebel şirketi rezervasyon yaptıktan sonra da müşterilere en uygun veya daha iyi fırsatlarla değiştirerek müşterinin memnun kalmasını arttıracaktır. Tezin konusu olan otel öneri yapma özelliğiyle başka şirketlerden yenilikçi olmaktadır. Makine öğrenme alanı bilgi çağında, internet işinde yeni fırsatları sunmaktadır. Makine öğrenme alanındaki sıralama, karşılaştırma problemleri üzerinde yapılan çalışmaları üzerinden araştırma yaptıktan sonra, gelişmelerin ve uygulamanın hep 'bilgi alma' alanında oldu öğrenilir. 'Bilgi alma' alanında sıralama veya karşılaştırma her zaman 'sorguyla' benzerlik oranına göre sıralama yapılmıştır. Otelleri karşılaştırma için 'bilgi alma' alanındaki gelişmeler uygulanamaz. Hazır veya temel çözümlerin olmaması yeni bulguların, çözüm yöntemlerinin geliştirilmesini gerektirir. Tezin aşağıdaki bölümlerden oluşmaktadır.

Tezin diğer kısımlarında ilkönce temel kavramlar hakkında bilgi verilecektir. İkinci kısımda ise ilgili çalışmalar ve bu alandaki son gelişmeler hakkında bilgi verilecektir. Üçüncü kısımda sistem tasarımı ve geliştirilen uygulama iş akışı, UML tasarım ve sözde kodlarla birlikte detaylı anlatılacaktır. Bir sonraki kısımda ise elde edilen sonuçlar ve gelecek çalışmalar hakkında anlatılır.

1. TEMEL KAVRAMLAR

1.1. Makine Öğrenmesi

Makine öğrenmesinin tanımı "Makine öğrenmesi, kod yazmanıza gerek kalmadan, size belirli bir veri kümesi hakkında ilginç şeyler söyleyebilen genel algoritmalar oluşturma fikridir. Kod yazmak yerine bu genel algoritmayı veri ile beslersiniz ve bu şekilde algoritma, bu veriye dayanarak kendi mantığını oluşturur" olarak en basit terimlerle tanımlanmıştır [8].

Makine öğrenmesi iki türlü olur: birincisi denetimli öğrenme (supervised learning), ikincisi ise denetimsiz öğrenme (unsupervised learning). Literatürde çok bahsedilmeyen yarı-denetimli öğrenmesi de (semi supervised learning) vardır.

1.1.1. Denetimli makine öğrenmesi

Makine öğrenme problemlerinin çoğu denetimli öğrenme türünü kullanır. (x) giriş değişimi ve (y) çıkış değişimi olan ve ikisi arasında eşleştirme algoritması (1.1) Denklemindeki gibi kurulabilmesi denetimli öğrenme yöntemindendir,

$$y = f(x) \tag{1.1}$$

Amaç eşleştirme fonksiyonunu yeni (x) verisi için (y) çıkış değişkenini maksimum doğrulukta veya yakınlıkta bulabilecek gibi yapılandırmaktır.

Algoritmanın denetimli öğrenme adını algoritmanın eğitim verisinden öğrenmesini öğretmenin öğrenme surecini denetlediği gibi kafamızda canlandırabildiğimiz içindir. Biz doğru cevabı biliyoruz, algoritma iteratif olarak eğitim veri setini tahmin eder ve "öğretmen" tarafından düzeltilir.

Denetimli öğrenme regresyon ve sınıflandırma olarak daha ileri gruplandırma yapılabilir.

- Sınıflandırma: Sınıflandırma çıkış değişkeni bir sınıftan ibaret olan, "kırmızı" veya "yeşil", "hastalıklı" veya "sağ", problemlerdir
- Regresyon: Regresyon problemi ise çıkış değerinin "dolar" veya "ağırlık" gibi gerçek değerlerden oluşan problemlerdir

Sınıflandırma ve regresyon problemlerinde öneri sistemleri ve zaman serisi problemleri yaygındır.

Denetimli öğrenme algoritması için bazı popüler örnekleri şunlardır:

- Regresyon problemleri için lineer regresyon
- Sınıflandırma ve regresyon için rastgele orman (random forest)
- Sınıflandırma için destek vektör makinesi (DVM)

1.1.2. Denetimsiz makine öğrenmesi

Denetimsiz makine öğrenmesi sadece (x) giriş değişkeni ve karşılıklı çıkış değişkenine sahip olmamasıdır.

Denetimsiz öğrenmenin amacı verinin temel yapısından veya dağıtılışından model kurarak veriyi anlamaya çalışmaktır.

Doğru cevabın veya öğretmenin olmamasından dolayı denetimsiz makine öğrenmesi adı verilmiştir.

Denetimsiz makine öğrenme problemleri kümeleme (clustering) ve ilişkilendirme (association) olarak ileri gruplandırılabilir.

- Kümeleme: Verileri satın alma davranışına göre gruplandırmak gibi, verilerdeki doğal gruplamaları keşfetmek gibi problemleri çözmeye denir.
- İlişkilendirme: X'i satın alanlar Y'i de satın alma eğilimi gibi ilişkileri bulmaktır

Bazı denetimsiz makine öğrenmesi örnekleri şunlardır:

- Kümeleme problemleri için k-means
- İlişkilendirme kuralı için apriori algoritması

1.1.3. Yarı denetimli makine öğrenmesi

Çok sayıda (X) verisinin ve çok az sayıda (Y) çıkış verisinin olması, yani verilerin çoğunluğunun sınıflandırılmış olmamasına yarı denetimli makine öğrenme problemleri denir.

Bu problemler denetimli makine öğrenmesi ve denetimsiz makine öğrenme problemlerinin ortasından yer alır. Sadece bazı resimlerin sınıflandırılmış olduğu arşiv resimler bir örnektir.

1.2. Kullanılan Algoritmaları

1.2.1. Destek vektör makinesi (SVM)

Denetimli öğrenim yöntemidir ve sınıflandırma, regresyon ve aykırı değer bulmada kullanılır. Tezin problemini çözmede kullanılan en uygun ve doğruluk oranı da en yüksek olduğu için avantaj, dezavantaj veya başka özellikleri detaylı anlatılmasına karar verilmiştir.

Avantajları olarak aşağıdakiler sayılır:

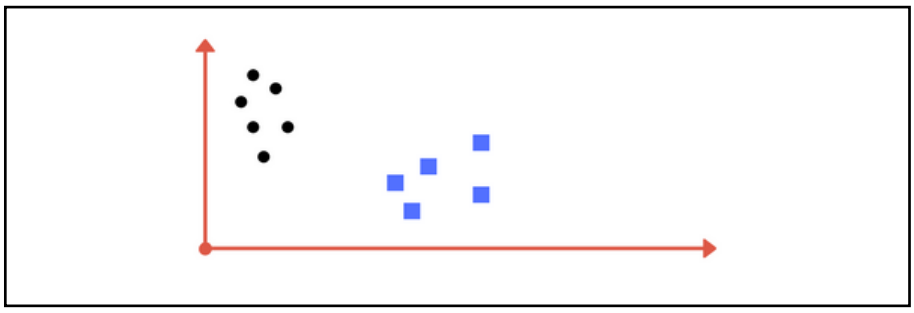
- Yüksek boyutlu alanlarda etkilidir
- Boyutların sayısı eğitim veri sayısından büyük olduğu durumlarda hala etkilidir
- Karar fonksiyonunda eğitim veri setinin bir alt kümesini kullanır, bu nedenle hafıza kullanımı etkilidir
- Çok yönlü: Karar fonksiyonu için değişik çekirdek fonksiyonları belirlenebilir. Ortak çekirdek sağlanır, ancak özel çekirdekler belirtmek de mümkündür.

Dezavantajları ise şunlardır:

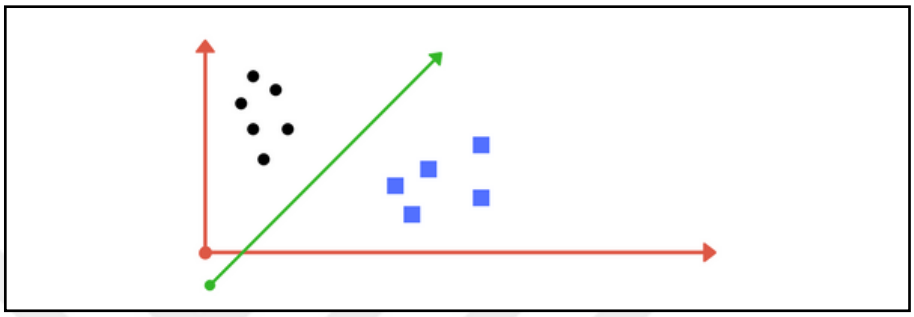
- Özelliklerin sayısı eğitim veri setinin sayısından çok daha fazlaysa, çekirdek seçimini yaparken aşırı uyumdan kaçınılmalıdır ve düzenleme terimi çok önemlidir
- Destek Vektör Makineleri olasılık tahminlerini sunmaz. Ancak, bunlar çok pahalı olan beş kat çapraz doğrulama yöntemini kullanarak elde edilebilir [9].

SVM bir geniş-marjlı sınıflandırıcıdır. Amaç, eğitim veri setindeki herhangi noktadan en fazla uzakta olan iki sınıf arasındaki bir karar sınırını bulmaya çalışan vektör uzay tabanlı makine öğrenme yöntemidir [10].

Aşağıdaki grafik resimde (Şekil 1.1) gösterildiği gibi iki sınıfın çiziminin verildiğini varsayalım. Sınıflar için bir ayırma çizgisine karar verebilir misiniz?



Şekil 1.1. Siyah daireleri ve mavi kareleri ayıran bir çizgi çizin

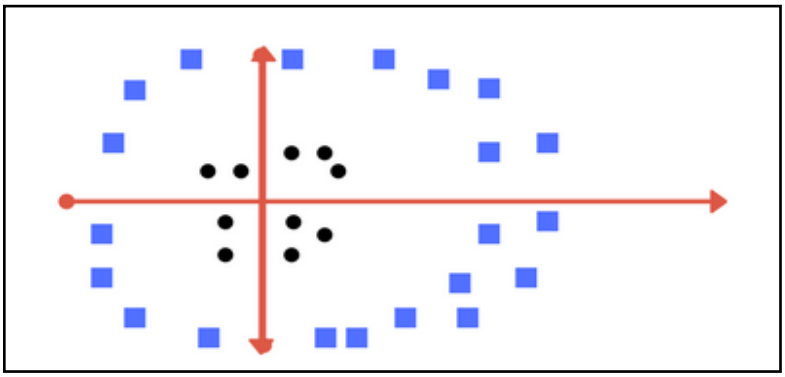


Şekil 1.2. Siyah daire ve mavi kareleri ikiye ayıran örnek çizgi

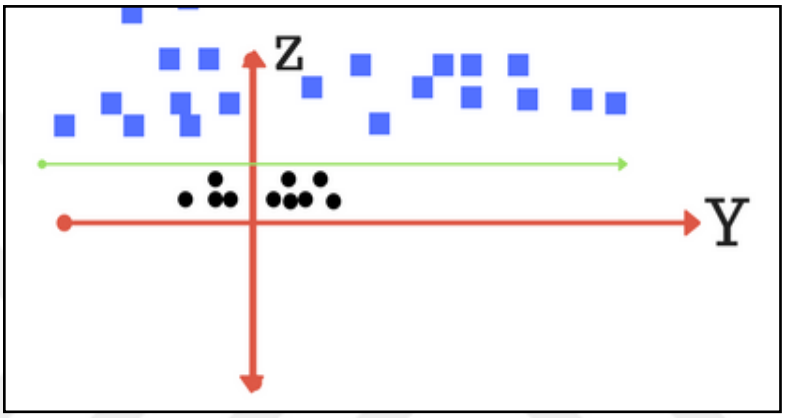
Yukarıdaki Şekil 1.2'ye benzer bir çözümde bulunmuş olabilirsiniz. Yeşil çizgi iki sınıfı oldukça iyi doğrulukta ikiye ayırır. Çizginin sol tarafındaki herhangi nokta siyah sınıfa, sağ tarafındaki herhangi nokta mavi sınıfa aittir. Sınıfları ayırma SVM'in yaptığı şey budur. SVM sınıfları ayırıcı çizgiyi veya hiperdüzlemi bulur.

1.2.1.1. Biraz kompleks yapıldığında

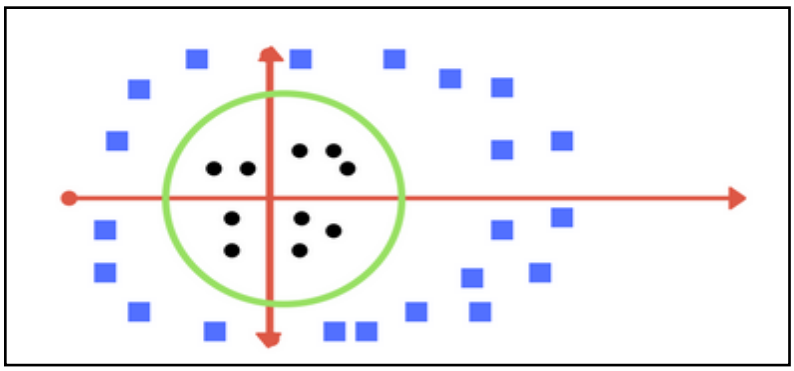
Şimdi, aşağıdaki resimde (Şekil 1.3) gösterilen verimizin olduğunu varsayalım. Bu resimdeki noktaları nasıl iki sınıfa ayırırdık? Açıkçası, bu x-y düzleminde iki sınıfı ayırabilecek bir çizgi yoktur. Peki ne yapacağız? Z ekseni olarak adlandırdığımız yeni bir boyutu dönüşüm uygulayarak elde ederiz. Noktaların z eksenindeki değerinin $w = x^2 + y^2$ olduğunu varsayalım. Şimdi z ekseninde çizim yapılırsa, net bir ayrım görünür ve bir çizgi çizilebilir. Şekil 1.4'te ayırma çizgisi gösterilmiştir.



Şekil 1.3. Bu düzlemde bir ayırma çizgisi çizebilir misiniz?



Şekil 1.4. zy ekseninin çizimidir. Ayırma burada yapılabilir

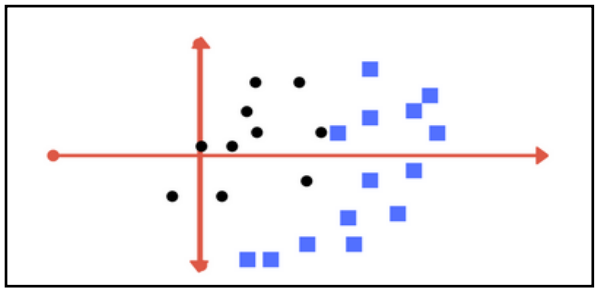


Şekil 1.5. xy düzlemine geri dönüştürüldüğü zaman

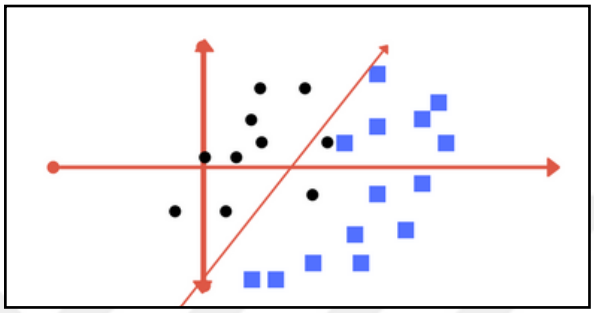
Geri dönüştürme yapıldığı zaman Şekil 1.5'te gösterildiği gibi dairesel sınırlama yapılmış olur.

1.2.1.2. Biraz daha kompleks yapıldığında

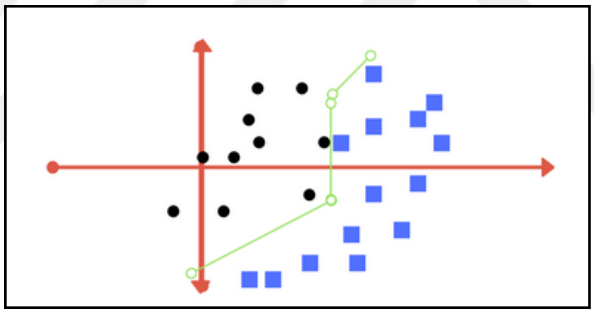
Şekil 1.6'da gösterilen veri çiziminde örtüşme olursa ne yapılmalı? Veya bazı siyah noktalar mavi noktalar arasında olursa? Şekil 1.6 ve Şekil 1.8 de gösterilen çizgilerin hangisi kullanılmalı?



Şekil 1.6. Veri sınıflarında örtüşme olması



Şekil 1.7. Birinci çözüm



Şekil 1.8. İkinci çözüm

Şekil 1.7'de ve Şekil 1.8'de gösterilen çizgilerin hangisi doğru olur? Doğrusu, iki cevap da doğru. İlki bazı dış noktalara toleranslı. İkincisi mükemmel bölme ile sıfır toleransı elde etmeye çalışıyor. Gerçek dünya uygulamasında milyonlarca eğitim veri seti için mükemmel bir sınıf bulmak çok zaman alır. Mükemmel çizgi ile toleranslı çizginin arasındaki bir tolerans değerini ayarlayıcı parametreye normalleştirme parametresi denir. Bir sonraki alt bölümde iki parametre, normalleştirme ve gamma, iyice anlatılacaktır. Bunlar SVM sınıflandırıcısında bir ayarlama parametresidir. Bunları değiştirerek önemli ölçüde doğrusal olmayan sınıflandırma çizgisine makul zaman aralığında erişilebilir.

Yine bir parametreye çekirdek parametresi denir. Doğrusal bir ayırma çizgisi isteyip istemediğimizi tanımlar.

1.2.1.3. Ayar parametreleri: çekirdek, düzenlileştirme, gamma ve marjın

Çekirdek

SVM veri setinin doğrusal çizgi ile bölünemediği zaman ne yapılmalı? Bu zamanda ortaya çekirdek fonksiyonun sihirli özelliği kullanılmalıdır.

Çekirdek veri setinin ayrılabilir olmadığı zaman ayrılabilir olan yüksek boyuta dönüştürücü fonksiyondur.

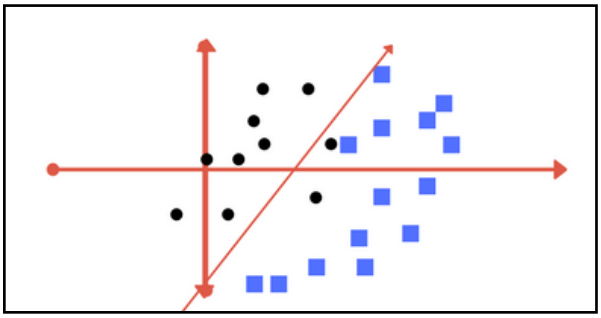
Basit kelimelerle, eğer problem iki boyutlu düzende ayrılabilir değilse, fonksiyonun boyutunu belirli yüksek düzene çevirerek, ayrılabilir düzen elde edilir.

Düzenlileştirme

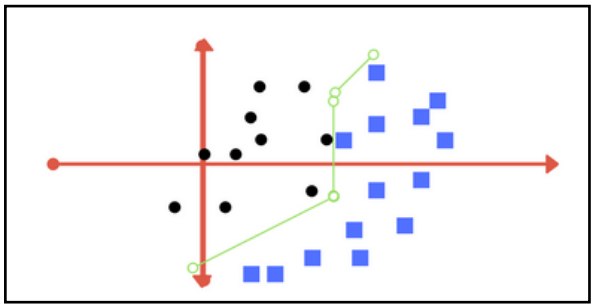
Düzenlileştirme parametresi SVM optimizasyonuna her bir eğitim verisini sınıflandırmada yanlış sınıflandırmayı ne kadar kaçınmak istendiğini belirtir. Python sklearn kütüphanesinde bu parametre C adla daha çok bellidir.

Büyük C değerleri için, optimizasyon, eğer bu hiperdüzen tüm eğitim veri setini sınıflandırmada daha iyi sonuca varırsa, daha küçük hiperdüzeni seçer. Tersine, C parametresinin değerinin küçük olması, iyileştiricinin daha büyük marjinli hiperdüzeni seçmesine yol açar. Büyük marjınli hiperdüzen'ler daha çok veri noktalarının yanlış sınıflandırmasına yol açabilir.

Aşağıdaki iki resimde (Şekil 1.9, Şekil 1.10) C değerinin düşük ve yüksek olduğu zaman ayırma çizgisinin nasıl etkilendiğini göstermektedir. C değerinin düşük olması daha çok noktanın yanlış sınıflandırmasına yol açar, yüksek olması ise çizginin daha detaylı ayırma yapmasını sağlar.



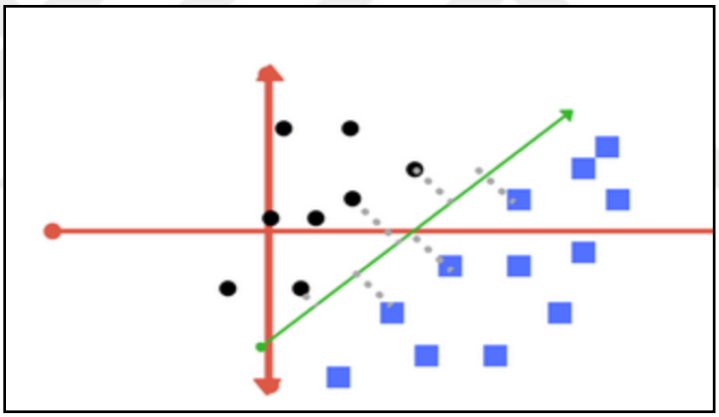
Şekil 1.9. Düşük düzenlileştirme



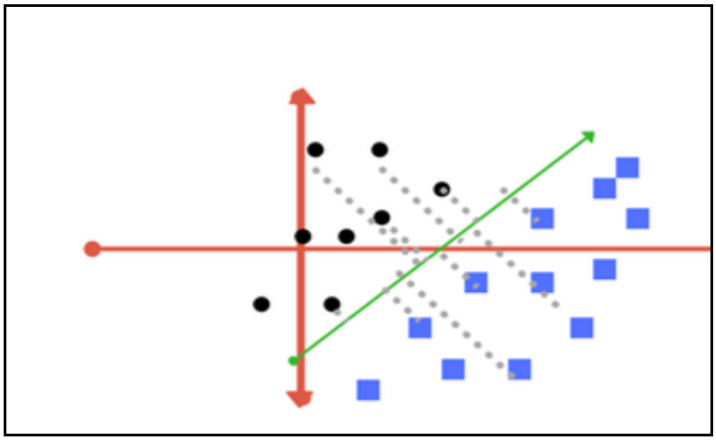
Şekil 1.10. Yüksek düzenlileştirme

Gamma

Yalnız eğitim örneğinin ne kadar etkisi olduğunu belirleyici parametreye gamma parametresi denir. Düşük olması "uzak", yüksek olması "yakın" anlamına gelir. Başka bir değişle, düşük gamma uzaktaki noktaların ayırma çizgisinin belirlenmesine etkisi vardır. Buna karşı, yüksek gamma değeri ayırma noktasına yakın olan noktaların kullanılmasını sağlar.



Şekil 1.11. Yüksek gamma



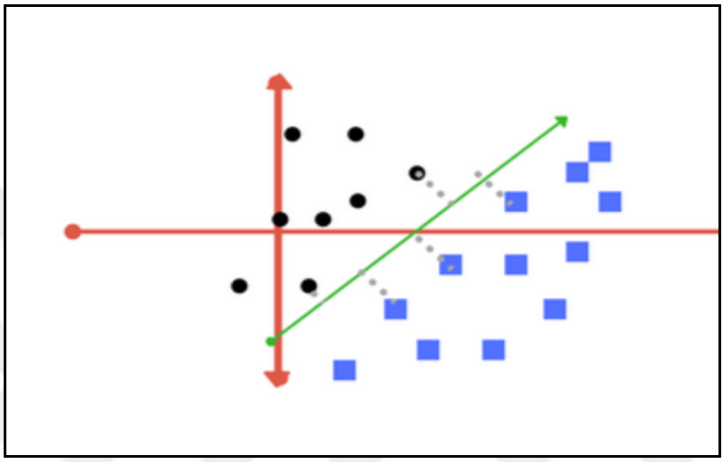
Şekil 1.12. Düşük gamma

Marjin

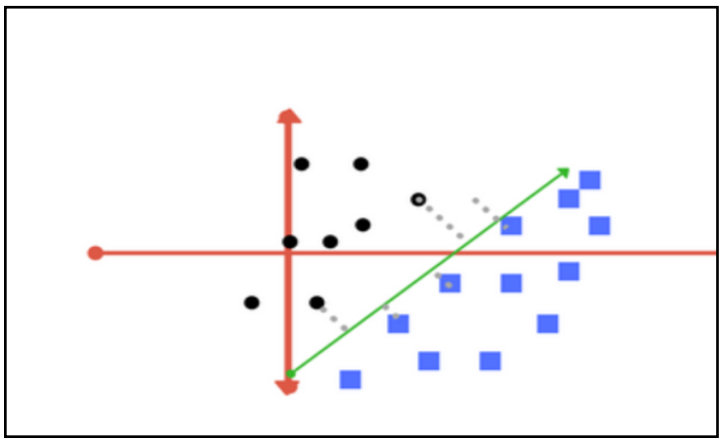
Marjin SVM sınıflandırıcısın en önemli özelliklerinden biridir. SVM temelinde iyi bir marjin elde etmeye çalışır.

Bir marjin, çizginin en yakın sınıf noktalarına göre ayrılmasıdır.

İki sınıfa da maksimum uzaklıkta olan marjin iyi bir marjindir. Aşağıdaki resimler iyi ve kötü marjinleri görsel olarak gösterir.



Şekil 1.13. İyi marjin örneği



Şekil 1.14. Kötü marjin örneği

SVM algoritmasının yukarıdaki özellikleri kullanılarak tezin ana konusu olan makine öğrenmesi problemi en yüksek doğruluğu elde etmeye çalışmıştır.

1.3. Makine Öğrenim Sıralaması

Son yıllarda makine öğrenme alanındaki makine eğitim tekniklerinin gelişmesiyle "Makine Öğrenim Sıralaması" yeni bir araştırma alanına dönüşmüştür. Başarılı

olarak tamamlanan araştırmaların sonucunda bulunan yeni teknik ve gelişmeler “Bilgi Alma”, metin özetleme alanlarında başarılı uygulanmıştır. Birçok alanlarda, makine öğrenim sıralaması eğitim verilerini kullanarak en uygun sıralama modelini kurabilmektedir. Geleneksel sıralama yöntemlerinden daha da iyi sonuçları elde edebilmektedir [11].

2. İLGİLİ ÇALIŞMALAR

Otel öneri problemi bir sıralama problemi olduğu için, literatür taraması sıralama alanında ve otel rezervasyon alanında yapılmıştır.

Sıralama problemi neredeyse günlük hayatımızın her alanında bulunmaktadır. İnternet alanında ise en önemli ve en çok kullanılan servislerin altında bulunan kaçınılmaz problemlerdendir. Örneğin, Google arama o kadar günlük internet hayatımıza bağlanmıştır ki, internetteki hiçbir işimizi Google'suz bitiremeyiz. Şimdi yazıyor olduğum bu tezi yazmak internetsiz, Google un arama motoru'suz ne kadar zor olacağını aklıma sığdıramam. Larry Page ile Sergey Brin'in doktora tezinde geliştirdiği 'PageRank' sıralama algoritması bugüne kadar Google şirketinin en büyük gelirlerini oluşturmaktadır. Netflix müşterilerinin memnuniyetini yükseltmek için makine öğrenme tekniklerini neredeyse her bölümünde, her gün kullanmaktadır [12][13]. Kişiselleştirilmiş sıralama algoritmasının en gelişmiş yöntemleri, bulguları Netflix'in en büyük servetidir. Facebook sosyal ağı da aynı probleme, kullanıcılara hangi bildiriyi önce gösterme sorusuna cevap vermede karşılaşmaktadır. Büyük şirketlerin neredeyse hepsinin en büyük alt yapısı, gelirinin nedeni makine öğrenme, yapay zekâ yöntemleri ile kullanıcıya en uygun, ilgili içeriği sunabildiğindedir veya kendi alanındaki servisi yeni seviyeye çıkarabildiğindedir.

Expedia online otel rezervasyon yapma platformu da otel işinde ayrıcalı olmak, kullanıcıların memnuniyetini arttırmak amacında birkaç araştırmada bulunmuştur [1]. Online seyahat acentelerinin tek amacı vardır: siteye gelen her misafir kullanıcının otel rezervasyon yapma olasılığını maksimuma çıkarmak. Bu problemi kullanıcının aradığı kriterleri, tarihsel verileri kullanarak var olan otelleri satın alma olasılığına göre sıralamayla çözmek mümkündür. Nishith veya başkaları bu problemi Naive Bayes, Lojistik regresyon, Random Forest, Ranking SVM algoritmalarını kullanarak en iyi doğruluğu elde etmeye çalışmışlardır [1]. Bu çalışma bizim çözmeye çalıştığımız probleme tamamen olmasa da biraz alakası vardır. Bizim de çözmeye çalıştığımız problem sıralama.

Ryosuke Saga veya başkaları “Kullanıcının geçiş tarihine göre otel öneri sistemi” çalışmasında öneri sistemini geçiş tarihini yönlendirilmiş grafiği kullanarak çözmeye çalışmıştır. Uygun olmayan otelleri Simspon katsayı formülünü kullanarak filtreleme yapmıştır. Aday otellerin listesini hazırladıktan sonra sıralamayı grafik düğümlerine giriş ve çıkış sayılarını kullanarak yapmaya çalışmıştır [2].

Walek ve arkadaşları “Otel rezervasyon sistemine uzman sistemi önerisi” çalışmasında otel rezervasyon sistemlerinin otel servislerinden başka otel etrafındaki veya şehirdeki etkinlik, aktivite veya ilgi noktalarıyla bağ kurmadığını söyleyerek, bu boşluğu dolduracak çözüm üzerinden çalışmışlardır. Aslında bu ilginç ve yaygın kullanılabilecek özelliktir [9]. Çünkü müşterilerin çoğunluğu otel rezervasyon yaparken etrafında ilgi noktalarının, tarihi noktaların, gezi yerlerinin oldukça çok olmasını ister.

Tezin ana konusu olan sıralamanın makine öğrenmesinin geleneksel algoritmalarıyla çözmeye çalışmak biraz yanlış olacağını, geleneksel algoritmalar sonucu verinin belirli bir parçasıyla elde edebileceğini, sıralama probleminde ise verinin tamamının sonuca etkisi olduğunu Tiziano Papini “SortNet: Sıralamayı sinirsel tercih fonksiyonu ile öğrenmeye çalışmak” adlı çalışmasında iyice kanıtlamaktadır [14]. Bu çalışma tezde kullanılan yöntemin daha karmaşık modelini kullanmaktadır. Tezde ise karşılaştırma verilerini geleneksel makine öğrenme yöntemi olan sınıflandırma problemine dönüştürerek, standart sınıflandırma algoritmalarıyla çözmeye çalışmıştır.

2.1. Skyline Sorguları

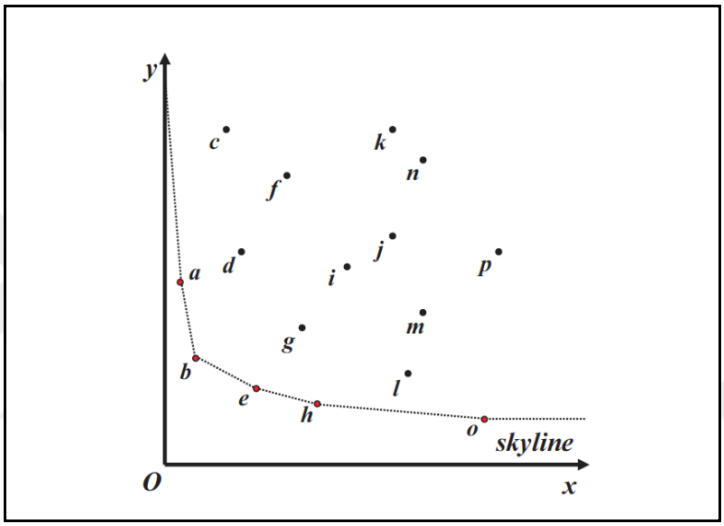
Bilgisayar literatüründe “skyline queries” adında araştırma dalı vardır. Bu alan çok kriterli karar destek uygulamalarında kullanılmıştır. Skyline sorgusu veri setinin ve üstünlük ilişkisinin verildiği zaman başka nesnelerle üstünlük edinemeyen nesneleri döndürür.

Belirli coğrafik sınırda bulunan otellerin en iyisini, en uygununu bir skyline sorgu problemi diye düşünülebileceği düşünüldüğü için bu alanda da literatür taraması yapılarak ne kadar uygun olup olmadığına karar verilmeye çalışılmıştır.

Veri setinin çok boyutlu nesnelerden olduğu durumda, nesne başka nesneden tüm boyutlarda üstün veya aynı güce sahip olduğu zaman nesne başka nesneden üstün

denilebilir.

İki boyutlu bir veri seti ile örnek olarak göstermek daha uygundur. Öklid düzleminde 16 noktanın, a(1, 12), b(2, 7), c(4, 22), d(5, 14), e(6, 5), f(8, 19), g(9, 9), h(10, 4), i(12, 13), j(15, 15), k(15, 22), l(16, 6), m(17, 10), n(17, 20), o(21, 3), p(22, 14), olduğunu varsayalım. Her bir düzlemde tercih fonksiyonu minimum nesne olsun. Böyle durumda skyline sorgusu {a, b, e, h, o} nesnelerini döndürür. Skyline sorgusunun grafiği aşağıdaki grafikte gösterilmiştir. Kırmızı noktalarla gösterilen nesneler ise sorgunun cevabıdır. Görüldüğü gibi bu nesneler kendi boyutunun minimumudur [15].



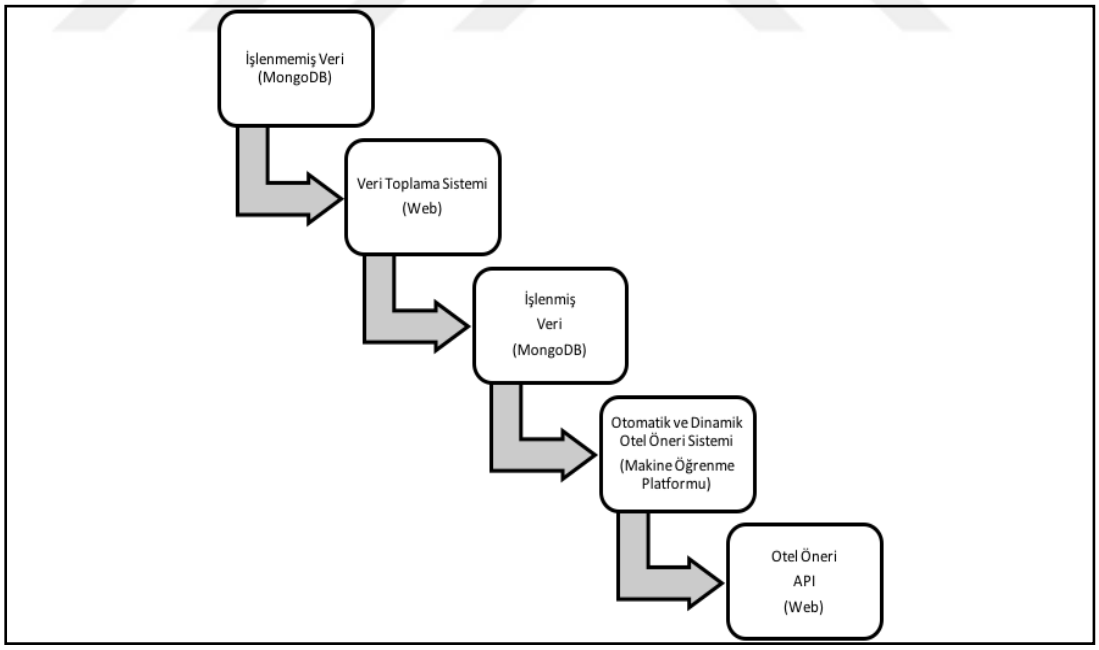
Şekil 2.1. Skyline örneği

Son yıllarda "Skyline sorgu" alanında yeterli gelişmeler olmadığı, araştırma kağıtlarında önerilen algoritmaları genelde teorik olarak kaldığı, açık kaynaklı kütüphanelerin bulunmadığı ve var olan algoritmalar çok yüksek boyutlu nesneler, büyük veri setinde etkili kullanılamadığı için tezin ana problemini bir "skyline sorgu" yöntemiyle çözmek daha zor olacağı kararına varılmıştır. Açık kaynakların yeterli olmadığı araştırma alanlarında istenilen sonuca erişmek çok büyük çaba gerektirir.

Sonuç olarak, "skyline sorgu" yöntemiyle çözmek tamamıyla reddedilmiştir. Yine de problemi değişik yollarla çözmeye çalışmak her zaman yüksek başarıları getirir.

3. SİSTEM TASARIMI VE UYGULAMA

Bu bölümde çalışmanın ana amacı olan sistemin tamamını detaylı açıklamaya çalışılmıştır. Sistem iki ayrı uygulamadan oluşmaktadır. Birinci sistem makine öğrenmesi için temel gereksinimlerden olan veriyi toplamak için tasarlanmıştır. Veri toplama işini otel rezervasyon işinde seyahat acenteleri manuel olarak yapmaktadır. Veri toplama işi sadece iki oteli karşılaştırma ve sonucu veritabanına saklama işinden ibarettir. Ayrıca birinci sistem toplanmış veriyi ikinci sisteme aktarmada veri ön işleme, dönüştürme işlerinin yüzde yüz olmasa da yüzde seksenini uygulamaktadır. İkinci uygulama ise birinci uygulamanın yardımıyla toplanmış verileri makine öğrenme yöntemini kullanarak otel öneri sistemini geliştirmiştir. Veri toplama işini bir web platformu üzerinden, makine öğrenme problemi ise web platforma bağımsız olduğu için iki ayrı uygulama olarak geliştirilmiştir. Sistemin akış diyagramı Şekil 3.1'de gösterilmiştir.



Şekil 3.1. Sistem akış diyagramı

Görüldüğü gibi sistem verinin üç ayrı sistem arasında akıştan ibarettir. İlk önce otel hakkında işlenmemiş veri MongDB veritabanında saklanmaktadır. Sonra veri toplama sistemi işlenmemiş veriyi seyahat acentelerinin yardımıyla işlenmiş veriye

dönüştürmektedir ve geri MongoDB veritabanına ayrı koleksiyona saklamaktadır. Otel öneri sistemi ise işlenmiş veri üzerinden makine öğrenme yoluyla elde edilmiş eğitilmiş modeli hazırlamaktadır. Sonunda ise eğitilmiş model otel karşılaştırma ve öneri işini REST API olarak başka sistemlere veya halka sunmaktadır.

3.1. Veri Toplama Sistemi

Makine öğrenmesi yöntemiyle çözülen problemlerin hepsinin temel gereksinimi vardır. Temel gereksinim makineyi öğretmeye çalıştığımız veridir. Bu alt bölümde veri toplama sistemi detaylı anlatılacaktır.

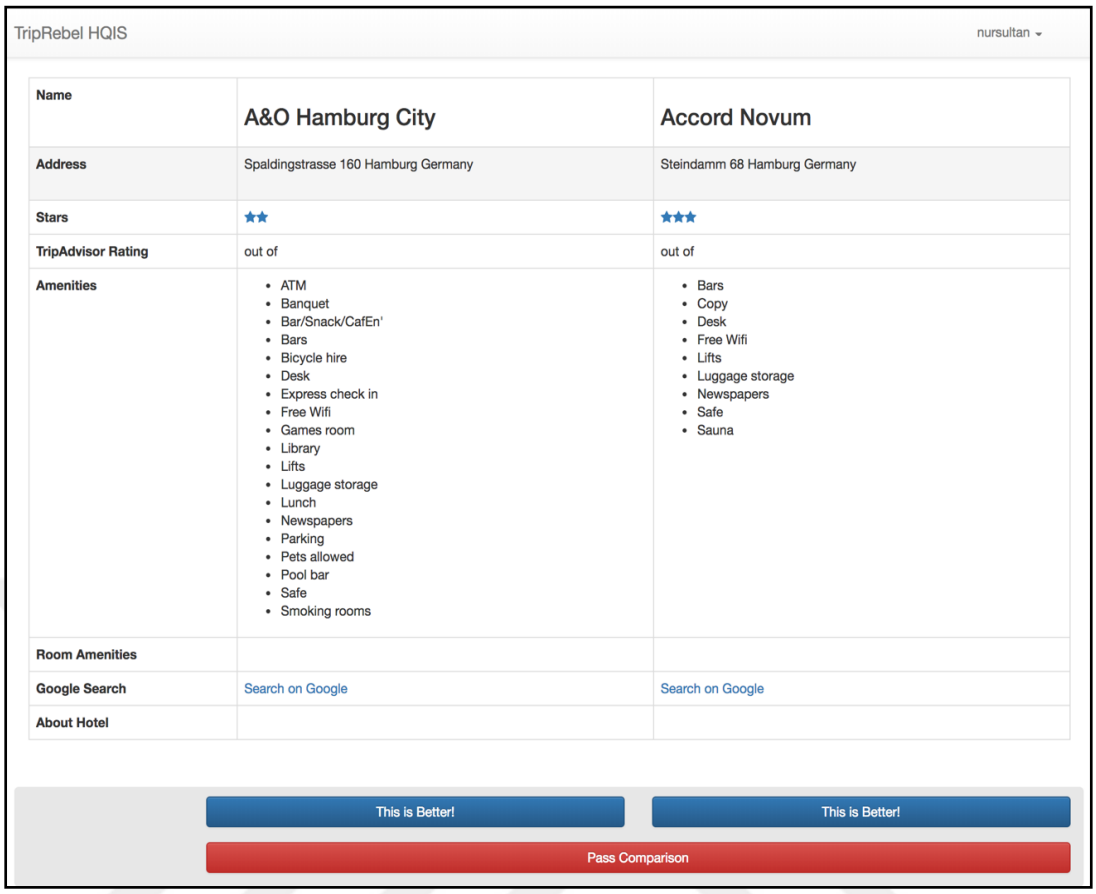
3.1.1. Sistem arayüzü

Sistemde iki türlü arayüz vardır. Birincisi seyahat acenteleri için. İkincisi ise yöneticiler içindir. Bu alt bölümde iki arayüz kısaca anlatılacaktır.

Maalesef, yazılım bir alman şirketi için hazırlandığı için arayüz İngilizce hazırlanmıştır. Üstelik, otel verilerinin tamamı İngilizce olduğu için, arayüzün Türkçe verilerin İngilizce gösterilmesi biraz uyumsuzluk yaratacağı göz ününe alındığı için de Türkçeye çevirme işi yapılmamıştır.

3.1.1.1. Seyahat acenteleri için arayüz

Seyahat acenteleri için arayüz tasarımı çok basit tasarlanmıştır. Sadece bir sayfa! Seyahat acenteleri için yapması gereken de tek iş vardır. O da iki otelin daha iyisini seçmek veya somut bir karara varılamazsa karşılaştırmayı geçmek. Her sayfa yenilemesi, daha iyi oteli seçmek veya geç tuşuna basmak yeni iki rastgele oteli dizi koleksiyonundan yükler. Seyahat acentesi ise istediği kadar otel karşılaştırmasını yapar. Ne kadar çok karşılaştırma yaparsa o kadar iyi. Çünkü veri özelliklerinin sayısı çok sayıda olduğu için, makine öğrenme algoritmalarına daha çok veri vermek sonucun doğruluk oranını arttırır.



Şekil 3.2. Otel karşılaştırma arayüzü

3.1.1.2. Süper yönetici arayüzü

Ana sayfa

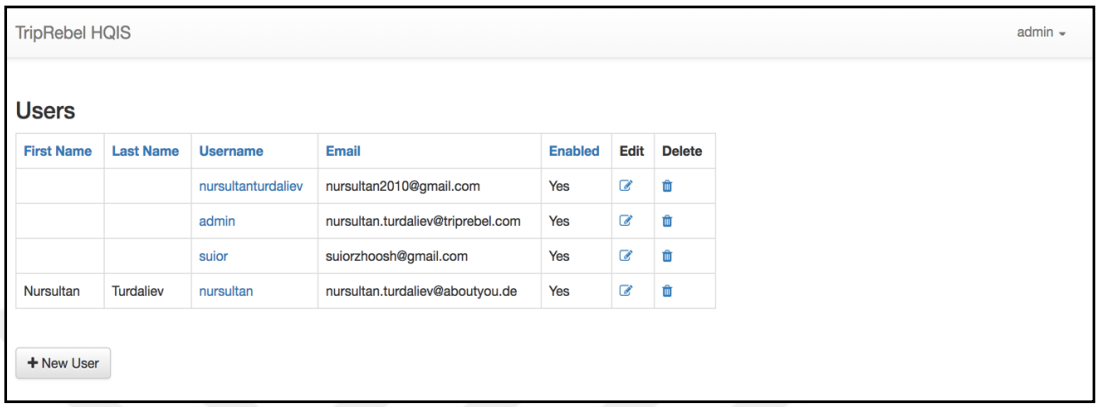


Şekil 3.3. Süper yönetici ara sayfası

Süper yönetici ana sayfasında sistemde kullanıcı ne türlü işleri yapabilirse, onlara bağlantı bulunmaktadır. Yukarıdaki Şekil 3.3'de görüldüğü gibi, süper yönetici yeni "seyahat acentesi" kullanıcınsın ekleyebilir, listesine bakabilir. Karşılaştırma otel listesine yeni otel ekleyebilir, otellerin listesine bakabilir. Her arayüz ayrıca anlatılacaktır.

Kullanıcı listesi

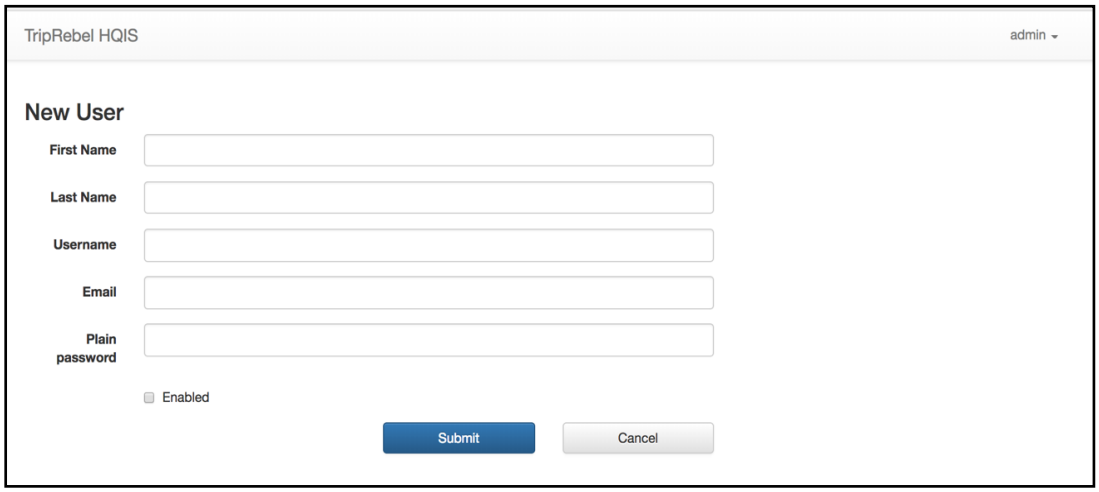
Kullanıcı listesi aşağıdaki Şekil 3.4'te gösterildiği gibidir. Görüldüğü gibi kullanıcılara uygulanılabilecek standart eylemler uygulanabilmektedir. Yani, kullanıcıyı düzenleme, silme, yeni kullanıcı ekleme.



Şekil 3.4. Kullanıcı listesi arayüzü

Kullanıcı ekleme arayüzü

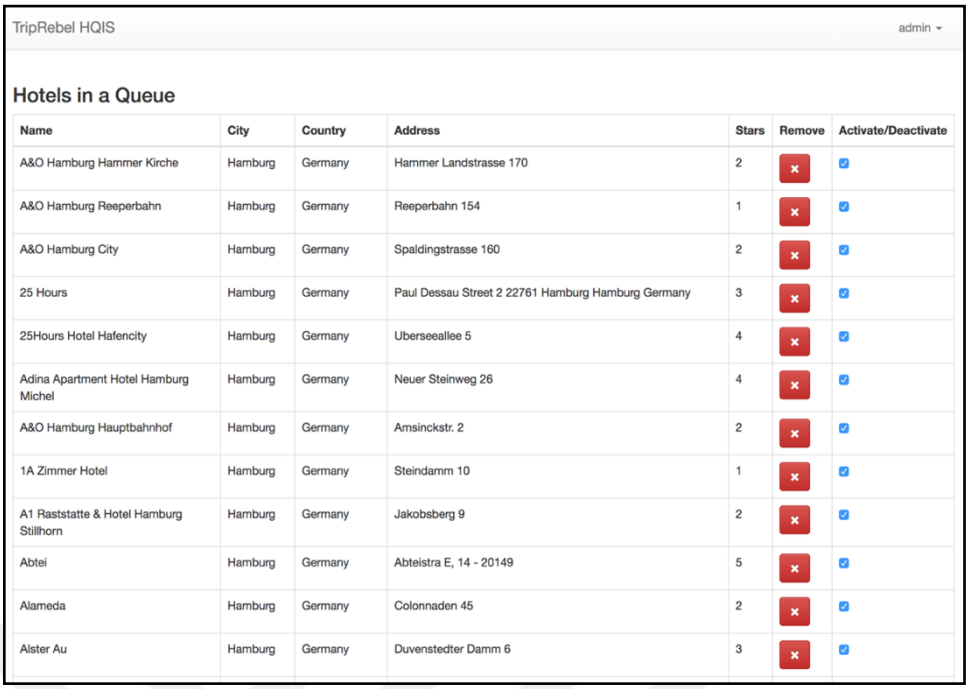
Kullanıcı ekleme arayüzü de standart arayüze sahiptir. Kullanıcının adı, soyadı, kullanıcı adı, e-posta ve şifre.



Şekil 3.5. kullanıcı ekleme arayüzü

Otel listesi arayüzü

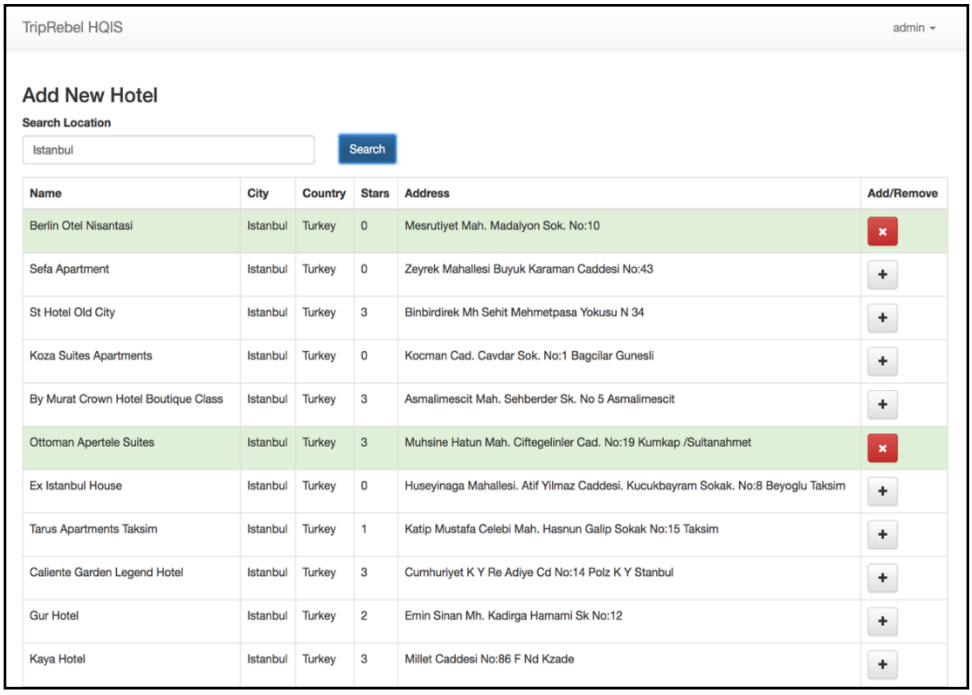
Otel listesi arayüzü de standart bir arayüze sahiptir. Şekil 3.6'daki gösterildiği gibi otelleri diziden silmek, etkinleştirmek, etkisizleştirmek mümkündür.



Şekil 3.6. Dizide bulunan otellerin listesi

Diziye otel ekleme arayüzü

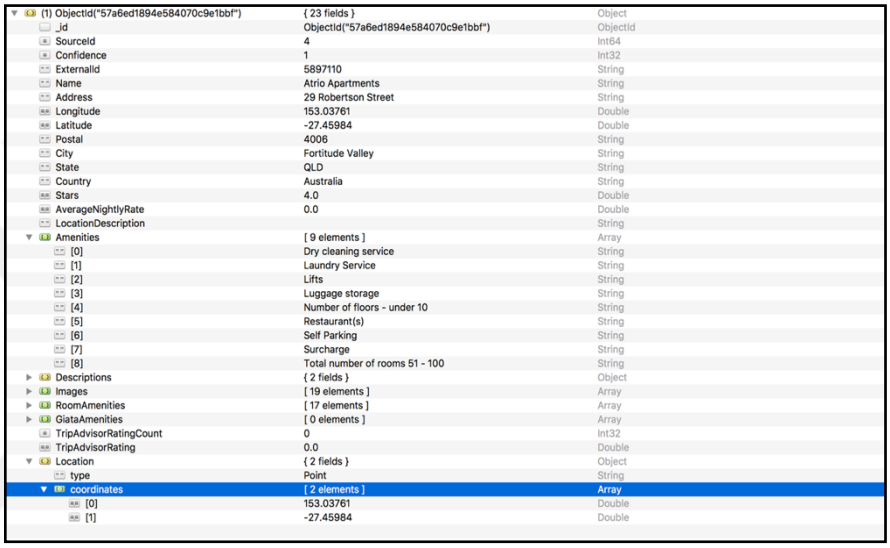
Oteli diziye eklemek için oteli bulunduğu şehir ile filtrelemek lazımdır. Dizide önceden eklenen otel yeşil ile gösterilir ve silmek mümkündür. Eklenmeyen otelleri ise artı butonuna tıklayarak eklemek mümkündür.



Şekil 3.7. Otel ekleme arayüzü

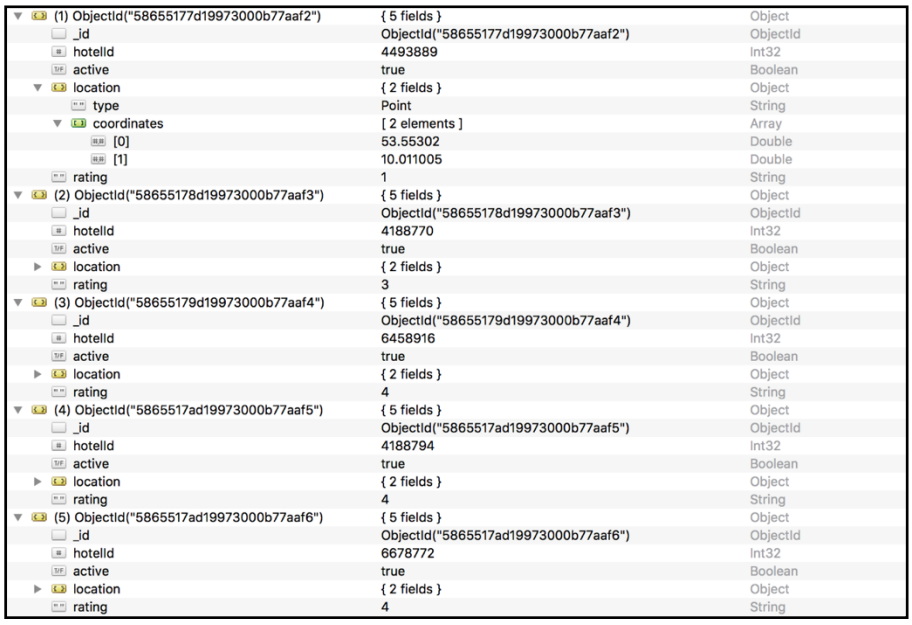
3.1.2. Veritabanı yapısı

Makine öğrenmesi problemlerinin temeli veritabanı olduğu için, veritabanının yapısını çok iyi bilmek ve istenilen formata dönüştürebilmek çok önemlidir. Aşağıda Şekil 3.8'de ilk olarak otel verisinin örneği gösterilmiştir. Görüldüğü gibi otelin neredeyse tüm otel rezervasyon yapma platformlarında bulunduğu özellikler bulunmaktadır. Özelliklerin çok olması değişik makine öğrenme yollarını denemeye yol açar. Veri tiplerinin çok değişik olması ise algoritmaya karmaşıklık katar.



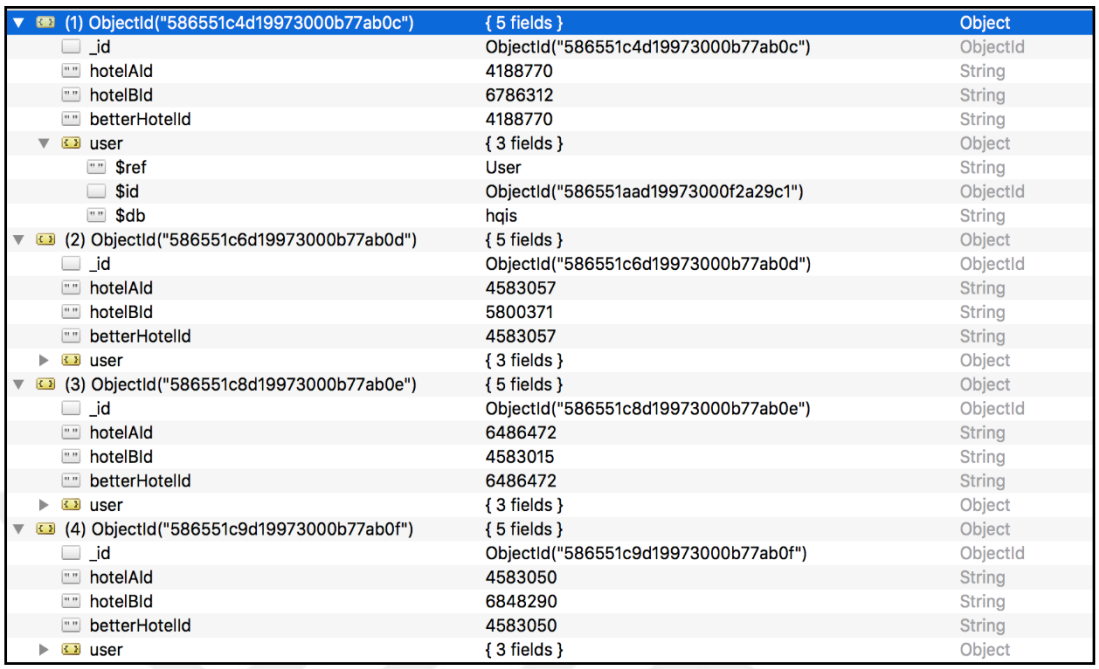
Şekil 3.8. Otel koleksiyonunun örneği

Dizi koleksiyonunda ise otel ID, aktif olup olmadığı ve konum kayıt edilmiştir.



Şekil 3.9. Dizi koleksiyonunun örneği

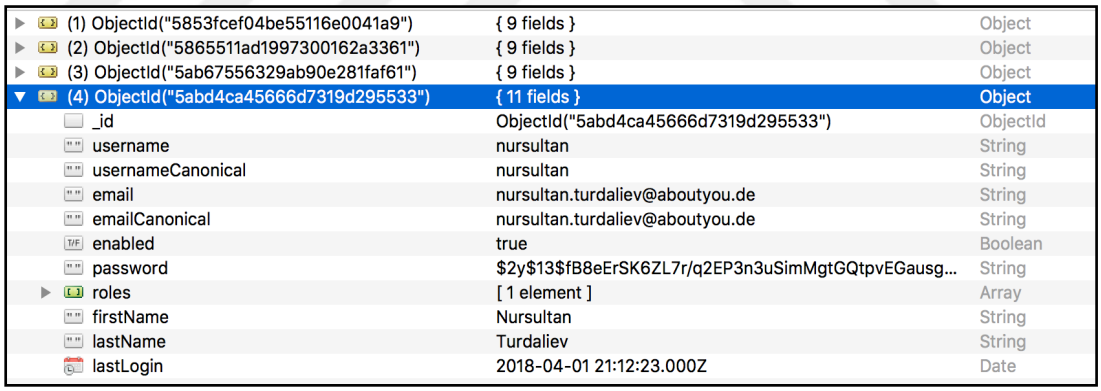
Ajans uzmanlarının karşılaştırma yaptığı otel ve cevaplar ise aşağıdaki gibidir.



▼ (1) ObjectId("586551c4d19973000b77ab0c")	{ 5 fields }	Object
_id	ObjectId("586551c4d19973000b77ab0c")	ObjectId
hotelAId	4188770	String
hotelBId	6786312	String
betterHotelId	4188770	String
▼ user	{ 3 fields }	Object
$ref	User	String
$id	ObjectId("586551aad19973000f2a29c1")	ObjectId
$db	hqis	String
▼ (2) ObjectId("586551c6d19973000b77ab0d")	{ 5 fields }	Object
_id	ObjectId("586551c6d19973000b77ab0d")	ObjectId
hotelAId	4583057	String
hotelBId	5800371	String
betterHotelId	4583057	String
▶ user	{ 3 fields }	Object
▼ (3) ObjectId("586551c8d19973000b77ab0e")	{ 5 fields }	Object
_id	ObjectId("586551c8d19973000b77ab0e")	ObjectId
hotelAId	6486472	String
hotelBId	4583015	String
betterHotelId	6486472	String
▶ user	{ 3 fields }	Object
▼ (4) ObjectId("586551c9d19973000b77ab0f")	{ 5 fields }	Object
_id	ObjectId("586551c9d19973000b77ab0f")	ObjectId
hotelAId	4583050	String
hotelBId	6848290	String
betterHotelId	4583050	String
▶ user	{ 3 fields }	Object

Şekil 3.10. Otel karşılaştırma sonuçları koleksiyonu

Son olarak da kullanıcılar koleksiyonu vardır.



▶ (1) ObjectId("5853fcef04be55116e0041a9")	{ 9 fields }	Object
▶ (2) ObjectId("5865511ad1997300162a3361")	{ 9 fields }	Object
▶ (3) ObjectId("5ab67556329ab90e281faf61")	{ 9 fields }	Object
▼ (4) ObjectId("5abd4ca45666d7319d295533")	{ 11 fields }	Object
_id	ObjectId("5abd4ca45666d7319d295533")	ObjectId
username	nursultan	String
usernameCanonical	nursultan	String
email	nursultan.turdaliev@aboutyou.de	String
emailCanonical	nursultan.turdaliev@aboutyou.de	String
enabled	true	Boolean
password	$2y$13$fB8eErSK6ZL7r/q2EP3n3uSimMgtGQtpvEGausg...	String
▶ roles	[1 element]	Array
firstName	Nursultan	String
lastName	Turdaliev	String
lastLogin	2018-04-01 21:12:23.000Z	Date

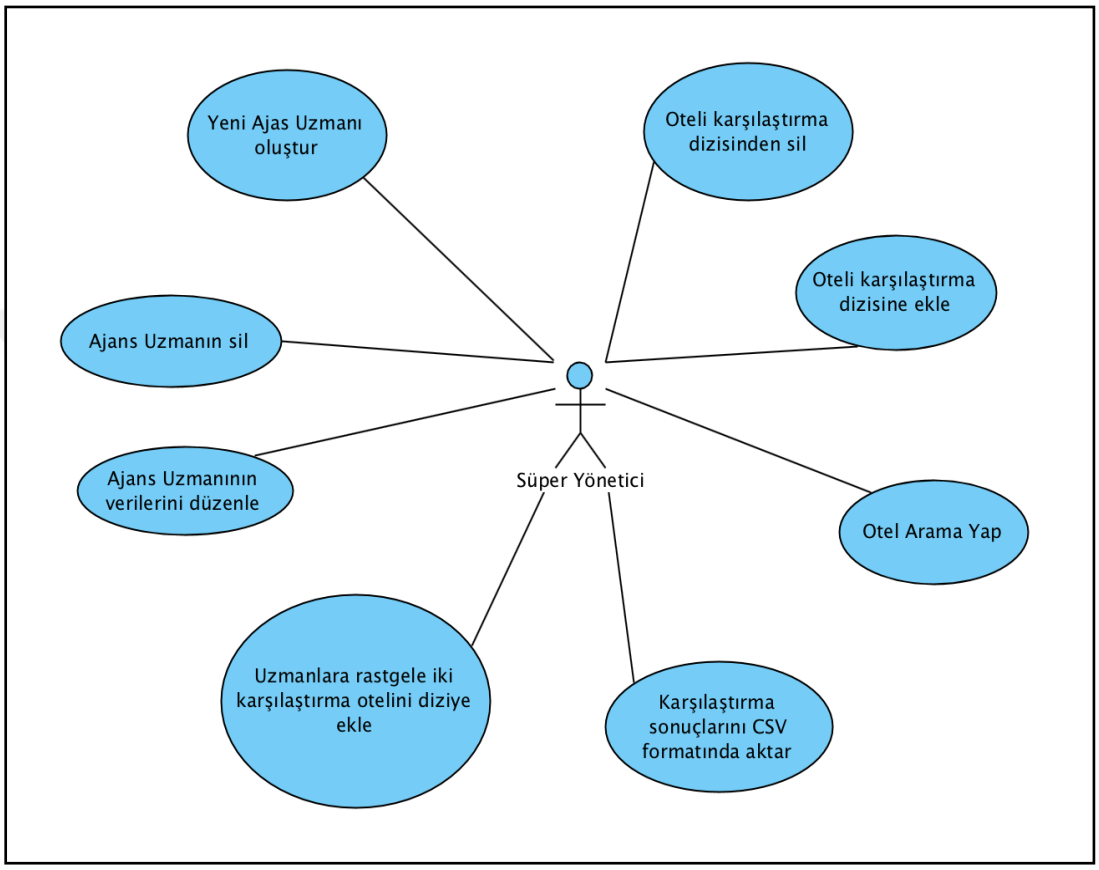
Şekil 3.11. Kullanıcılar koleksiyonu

3.1.3. Kullanım durumu

Sistem iki türlü kullanıcı rolünden oluşmaktadır. Birincisi “süper yönetici”, ikincisi ise “seyahat acentesi”. İki kullanıcının sistemde iki türlü görevi vardır.

3.1.3.1. Süper yönetici

"Süper yönetici" kullanıcısı sistemde neredeyse var olan tüm işi yapabilmektedir. Süper yöneticinin yaptığı işler aşağıdaki Şekil 3.12'deki "kullanım durumu" diyagramında detaylı gösterilmiştir.



Şekil 3.12. "Süper yönetici" rolündeki kullanıcının "kullanım durumu" diyagramı

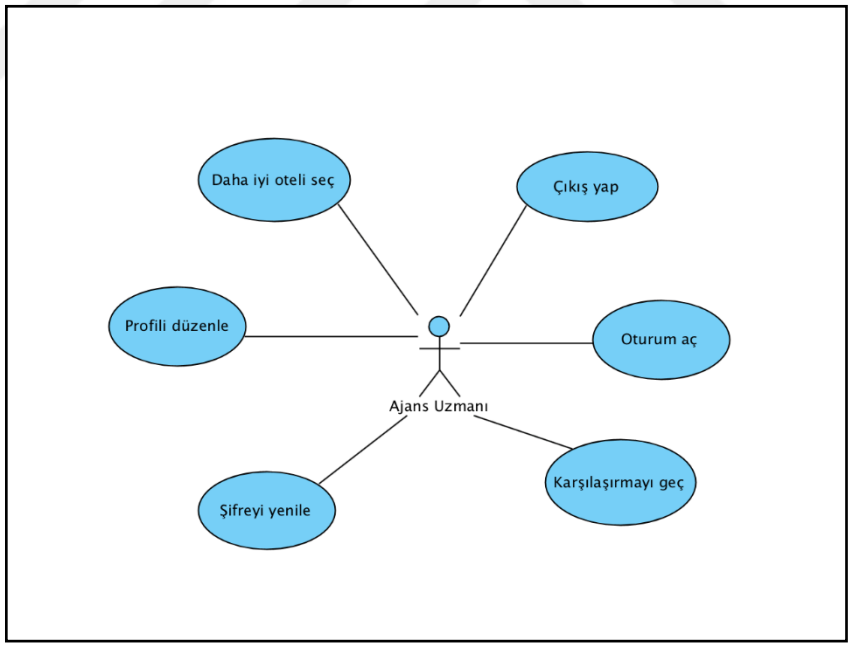
Her kullanım durumunun açıklaması ise aşağıda verilmiştir.

- Yeni ajans uzmanı oluştur: Otelleri karşılaştırıcı kullanıcıdır.
- Ajans uzmanın sil: Ajans uzmanını sadece süper yönetici silebilmektedir.
- Ajans uzmanının verilerini düzenle: Ajans uzmanının verilerini, adı, soyadı, şifresini değiştirir.
- Uzmanlara rastgele iki karşılaştırma otelini diziye ekle: Otel karşılaştırma dizisinden otelleri sil. Ajans uzmanları sadece süper yöneticinin seçtiği oteller arasında karşılaştırma yapabilmektedir. Sistemde tam olarak 1,5 milyon otel olduğu için, sadece biri birine benzer ve aynı şehirdeki otelleri karşılaştırma daha uygundur. İstanbul'daki otel ile Tokyo'daki oteli karşılaştırma hiç doğru değil.

- Oteli karşılaştırma dizisinden sil: Tesadüfen eklenmiş ve hiç uygun olmayan oteli silebilme özelliği olmalıdır.
- Oteli karşılaştırma dizisine ekle: Uzmanlara rastgele iki oteli karşılaştırma dizisine eklemek için önceden oteller seçilmelidir.
- Otel arama yap: Süper yöneticiye otel filtreleme kolaylığı sunmak için, otelleri şehir adıyla filtreleme yapabilme fonksiyonu sunulmuştur.
- Karşılaştırma sonuçlarını CSV formatında aktar: Hazırladığımız veriyi makine öğrenme ve otomatik karşılaştırma sistemine aktarmak için sistemden CSV formatında veri aktarılır.

3.1.3.2. Ajans uzmanı kullanıcısı

Ajans uzmanı kullanıcısı (seyahat acentesi) rolündeki kullanıcılar ise veri hazırlayan kullanıcılardır. Bu kullanıcılara rastgele iki otel sunulur ve kullanıcıların tek amacı iki oteli var olan özelliklere göre karşılaştırma yapmak. Kullanıcının kullanım durumu diyagramı aşağıdaki Şekil 3.13'de gösterilmiştir.



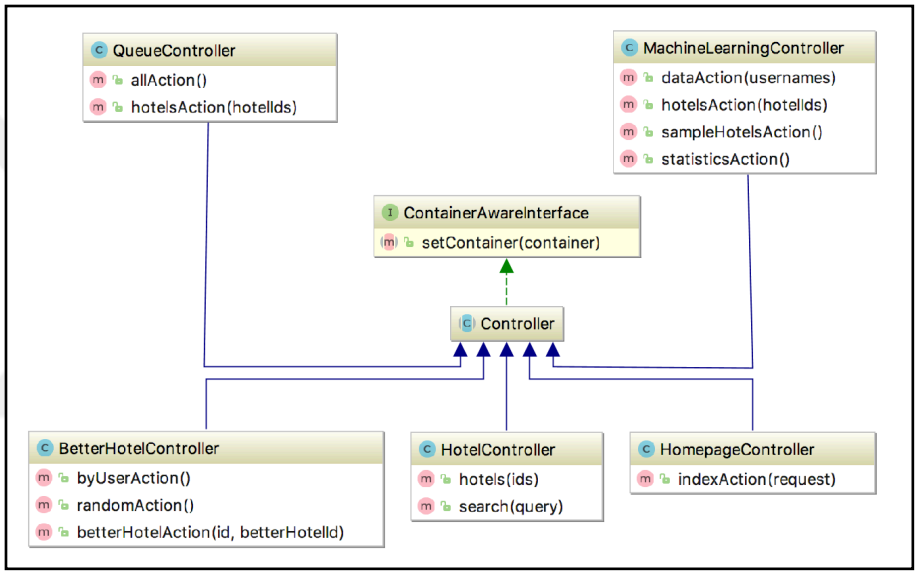
Şekil 3.13. Ajans uzmanı kullanım durumu diyagramı

- Daha iyi oteli seç: ajans uzmanı kullanıcılarının temel amacı otel karşılaştırmadır.
- Çıkış yap: Sadece giriş yapan kullanıcılar otel karşılaştırma işin yapabildiği için, çıkış fonksiyonu tüm kullanıcılar için mevcuttur.

- Oturum aç: Oturum açmadan tüm fonksiyonlar kapalıdır.
- Karşılaştırmayı geç: İki rastgele otel sunulduğu zaman, herhangisinin daha iyi olduğunu bilmediği zaman yapılması gereken eylemdir.
- Şifreyi yenile: Şifresini kaybeden ajanslar için şifre yenileme özelliği vardır.
- Profili düzenle: Ajanslar kendi profillerini düzenleme yapabilirler.

3.1.4. Sınıf diyagramı

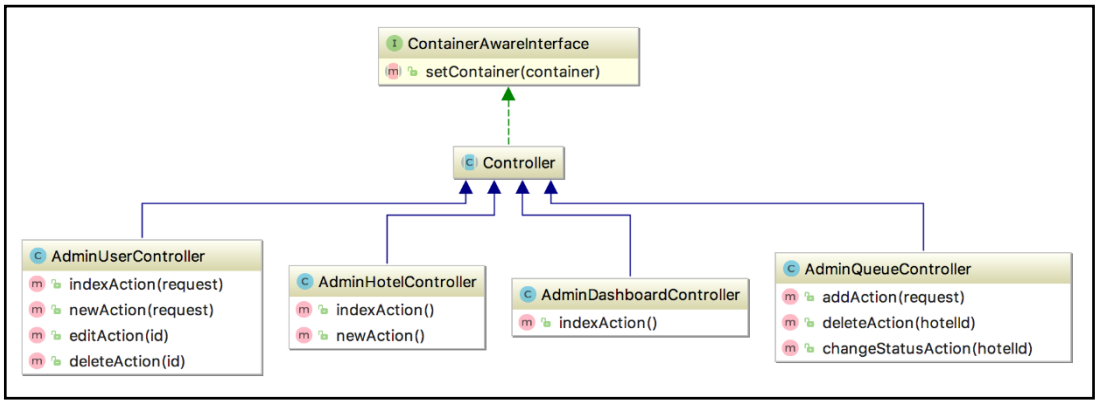
Veri toplama sisteminin genel halka açık (public) sınıf fonksiyonları aşağıdaki (Şekil 3.14) sınıf diyagramıyla gösterilmiştir.



Şekil 3.14. Sistemin sınıf diyagramı

Yukarıdaki sınıf diyagramından görüldüğü gibi, sistem kullanıcılara otelleri "HotelController" sınıfıyla sunmaktadır. Karşılaştırma, rastgele iki otel gösterme işini ise "BetterHotelController" sınıfında sunar. "MachineLearningController" sınıfı ise makine öğrenme platformu için gerekli olan fonksiyonları REST API üzerinden sunar.

Süper yönetici sistem üzerinde Şekil 3.15'te gösterilen sınıf diyagramındaki fonksiyonları kullanabilir. Süper yönetici ise kullanıcıları ekleme, silme ve düzenleyebilir ('AdminUserController'). Karşılaştırma için otelleri "Queue" koleksiyonuna ekleme, silme işini yapabilir ('AdminQueueController'). Yeni otelleri ekleyebilir ('AdminHotelController').



Şekil 3.15. Süper yönetici sınıf diyagramı

3.1.5. Kullanılan araç ve teknolojiler

Sistemi geliştirmede kullanılan araç, teknolojilerin listesi aşağıdaki gibidir:

- PHP + Symfony Framework
- Javascript + Vue.js
- Mongo DB
- Nginx

3.2. Otel Öneri Sistemi

Bu alt bölümde bir önceki alt bölümde anlatılan veri toplama sistemi ile hazırlanan verini kullanarak makine öğrenme yöntemiyle otomatik otel öneri sistemi detaylı anlatılacaktır.

3.2.1. Veri hazırlama süreci

Makine öğrenme algoritmaları veriden öğrenir. Çözülmesi gereken problemi çözmek için doğru veri ile algoritmayı beslemek çok önemlidir. İyi veri olsa bile, yararlı bir ölçekte, formatta ve anlamlı özelliklerin dahil olması gerekir. İyi ve anlamlı veri formatını elde etmek için aşağıdaki adımlar toplanan veri üzerine uygulanmıştır:

1. Veri seçmesi
2. Veri ön işlemesi
3. Veri dönüştürümü

Veri Seçmesi**.** Bu adımda var olan tüm veri kümesinden belirli bir alt kümeyi seçme işi yapılır. Genelde “çok daha iyi” felsefesiyle var olan tüm veri setini seçme isteği vardır. Bu doğru olmayabilir.

Hangi verinin gerçekten çözülmesi gereken probleme etkisi olduğunu bilmek lazım. Önemli olan veriler hakkında varsayımlar üzerinde ve sonradan varsayımı doğrulayabilecek şekilde veri seçimi yapılabilir.

Tezin ana konusu olan otel öneri, yani otel sıralama problemi için otel adresi, adı sıralama için önemli olmadığı varsayılabilir. Çünkü bir otelin adının "Motel One" veya "İbis" otel olması hiç önemli değildir. Fiyat, olanakların (internet, ütüleme servisi, garaj) belirli derecede etkisi olduğu varsayılabilir.

Otelin aşağıdaki özellikleri seçilmiştir:

1. Yıldız sayısı
2. Açıklama metini
3. TripAdvisor derecelendirme sayısı
4. TripAdvisor derecelendirmesi
5. Oda olanakları (klima, televizyon, saç kurutma makinesi)
6. Genel olanaklar (ücretsiz internet, otopark, kapalı havuz)
7. Resimler

Veri ön işlemesi

Verini seçtikten sonra, veriyi nasıl kullanacağımız hakkında düşünmemiz lazım. Veri ön işlemesi seçilmiş veriyi üzerinden çalışabileceğimiz formata dönüştürmekten ibarettir.

Üç yaygın veri ön işleme adımı vardır. Onlar biçimlendirme, temizleme ve örnekleme.

- Biçimlendirme: Seçtiğimiz veri bizim çalışabileceğimiz formatta olmayabilir. Biz veriyi düz dosya formatında isterken verimiz ilişkisel veritabanında olabilir. Veya veri özel dosya formatında olabilir, aksince biz ilişkisel veritabanında olmasını isterken.
- Temizleme: Belirli veriyi kaldırma veya eksik veriyi düzeltme işine veri temizlemesi denir. Problemi çözmede hiç faydası olmayan eksik veriler olabilir. Bu gibi veriler kaldırılması lazım. Bunlara ek olarak hassas bilgileri içeren veri nitelikleri olabilir. Hassas bilgiler anonimleştirmesi veya tamamen veri setinden kaldırılması gerekir.

- Örnekleme: Seçtiğimiz veri tutarı bizim çalışmak istediğimiz veri tutarından daha çok büyük olabilir. Veri çokluğu algoritmanın daha çok zaman çalışmasına ve daha büyük hesaplamalı ve bellek gereksinimlerine ihtiyacımızı arttırır. Çözümleri keşfetmek ve prototipleme işini hızlandırmak amacında verinin örneği üzerinde çalışılabilir. İyi bir çözüme varıldığında verinin tümünü kullanabiliriz.

Yukarıdaki anlattığımız veri ön işleme adımlarının tümü otomatik otel öneri verisi üzerinde uygulamaya çalışılmıştır. Her bir adımın nasıl uygulandığı aşağıda detaylı anlatılmıştır.

Tablo 3.1. Biçimlendirilmiş veri görünüşü

Yıldız Sayısı	Açıklama	Resimler sayısı	TripAdvisor Değerlendirmesi	Tripadvisor Değerlendirme Sayısı	Teras	Özel Plaj	Veya başka olanaklar listesi
4	Make yourself at home in …	43	8.8	1230	1	0	0 veya 1
5	Make yourself at home in one of the 50 air-conditioned rooms featuring kitchens.	8	9,0	5000	1	1	1 veya 0

Bildiğimiz gibi veri üzerinde temizleme ve örnekleme adımlarını yapmadan önce biçimlendirme işi yapılması gerekir. Otomatik otel öneri sisteminin verileri MongoDB veri tabanında saklandığı için, biçimlendirme adımı kaçınılmazdır. Uyguladığımız makine öğrenme algoritması belge tabanlı veritabanı üzerinde uygulanamaz. Veri biçimlendirme işi PHP programlama dilinde yapılmıştır. Biçimlenen veri ise bildiğimiz CSV dosya formatında dışa aktarılmıştır. Veri görünüşü kısaca aşağıdaki gibidir.

Yukarıdaki Tablo 3.1'de gördüğümüz veri biçimlendirmesi aşağıdaki orijinal veri formatından daha anlamlı ve makine öğrenmesi algoritmalarının uygulanmasına daha uygundur. Tabloya verinin tüm nitelikler sığmadığı için. Gösterilmeyen nitelikler son kolonda nasıl olduğu gösterilmiştir. Tüm otel ve oda olanakları (internet, kapalı

havuz, yıkama servisi, garaj vb.) ayrıca bir kolon olarak biçimlendirilmiştir. Niteliklerin tüm sayısı 162 dir.

(7) ObjectId("57a6eb4a94e584070c9541eb")	{ 23 fields }	Object
_id	ObjectId("57a6eb4a94e584070c9541eb")	ObjectId
SourceId	4	Int64
Confidence	1	Int32
ExternalId	4000038	String
Name	Fujairah Rotana Resort & Spa	String
Address	Al Aqah Beach	String
Longitude	56.363382339478	Double
Latitude	25.508715524651	Double
Postal	1856	String
City	Fujairah	String
State		String
Country	United Arab Emirates	String
Stars	5.0	Double
AverageNightlyRate	0.0	Double
LocationDescription	By The Ocean	String
Amenities	[31 elements]	Array
Descriptions	{ 2 fields }	Object
Images	[48 elements]	Array
RoomAmenities	[20 elements]	Array
GiataAmenities	[0 elements]	Array
TripAdvisorRatingCount	0	Int32
TripAdvisorRating	0.0	Double
Location	{ 2 fields }	Object

Şekil 3.16. Biçimlendirmemiş veri formatı. MongoDB veritabanında

Veri ön işlemesinin ikinci adımı ise temizleme işidir. Otomatik otel öneri sisteminde veri seçme işini yaparken gereksiz diye düşünülen nitelikler atıldığı için ve verinin tutarlılık oranı, ticari veri seti olduğu için çok yüksek. Hemen hemen yüzde yüz tutarlı olduğu için temizleme işine çok da ihtiyaç duyulmamıştır. Bu adım atlandı.

Son adım olarak da örnekleme işidir. Ticari veriler genelde çok miktarda olur. Genelde tüm veri seti üzerinde makine öğrenme algoritmasını uygulamak biraz yanlış olur. Çünkü algoritma çalışma zamanı, bellek ve hesaplama gereksinimi de çok yüksek olması gerekir. MongoDB veritabanında otel sayısı 1154983 tür. Veri boyutu ise 4716MB'tır. Bir önceki bölümde veri toplama işinde, seyahat acentelerinin otel karşılaştırma yaptığında milyon otelin tamamından iki rastgele oteli karşılaştırmayı sormak yerine, belirli sayıdaki şehir içindeki oteller manuel olarak seçilmiştir. Otel türünün, sunduğu olanakların da çok çeşitli ve tüm veri setini temsil edebilecek şekilde hazırlanmıştır. Örnekleme işinin uygulamayla prototipleme ve uygun çözüm üzerinde çalışma birçok kolaylaşmıştır.

Veri Dönüştürümü

Son adım ise veri dönüştürüm işidir. Kullanılan algoritma, problem alanı bilgisi bu adımı etkiler. Genelde birden çok veri dönüştürüm işleri yapılması gerekebilir.

Üç genel veri dönüştürüm yöntemleri vardır. Onlar ölçekleme, öznitelik ayrıştırma, birleştirme. Bu adım özellik mühendisliği olarak da bilinir.

- Ölçekleme**:** Ön işlenmiş veri seti karışık ölçekteki, miktardaki öznitelikleri içerebilir. Bazı öznitelik dolar ise başkası kilo veya satış hacminde olabilir. Çok makine öğrenme algoritmaları veri öznitelikleri aynı ölçekte, örneğin sıfır ve bir arasındaki değeri taşıyabilir. Sıfır en küçük, bir en büyük değer için.
- Öznitelik ayrıştırma**:** Makine öğrenmesi yöntemine kurucu bölümlere ayrıldığında daha faydalı olabilecek karmaşık bir kavramı temsil eden özellikler olabilir. Örnek olarak tarih ve saat özniteliğiniz olabilir. Tarih ve saat özniteliği tarih ve saat olarak iki niteliğe ayrılabilir. Belki problem alanı için sadece günün saati alakalı olabilir. Ne tür bir ayrıştırma yapabileceğimiz düşünülmesi lazım.
- Birleştirme**:** Bazı özellikler birleştirildiğinde çözülmesi gereken probleme daha anlamlı özelliği sunan zaman olabilir. Örneğin, müşterinin ne zaman sisteme giriş yaptığını veriler giriş sayısı olarak yeni bir özelliği temsil edebilir. Birleştirme yapıldığı zaman orijinal veriler atılabilir. Ne tür veri birleştirme işleri yapılması gerektiği düşünülmesi lazım.

Daha çok zamanın özellik mühendisliğine ayrılması sonuçta algoritma performansı için yararlı olabilir.

Veri dönüştürümü adımı makine öğrenmesi problemini daha büyük oranda başarıya ulaşmayı sağladığı için, ön işlenme yapılan veri üzerinde tüm alt adımları ayrıca ve dikkatle uygulanmıştır.

İlk olarak ölçekleme işinin nasıl uygulandığı hakkında anlatalım. Şekil 3.16'da görüldüğü gibi, otel verilerinin özellikleri çok değişik formatta olduğu görülmektedir. Resimler URL'lerin dizisi formatında olduğu, açıklama ise uzun tekst, TripAdvisor değerlendirmesi 0 ve 10 arasında, değerlendirme sayısı ise tamsayı formatında, olanaklar ise olanakların dizisi formatındadır. Tablo 3.1'e bakıldığında ise bu problemlerin çoğuna birleştirme, ayrıştırma veya ölçekleme işi ön işleme adımında birlikte yapıldığı anlaşılır. Örneğin, URL'lerin dizisi formatındaki resimler üzerinde birleştirme yapılarak, resim sayısı adında yeni özellik eklenmiştir. Yine de veriler tamamen aynı formatta değildir. Aynı formata dönüştürmek için tüm sayısal özellikler 0 ve 1 arasındaki değere ölçeklendirilmiştir.

Aşağıdaki tabloda (Tablo 3.2) verilerin orijinal formatı ve ölçeklendirme yapıldıktan sonraki formatı sunulmuştur.

Tablo 3.2. Veri özellikleri üzerinde ölçekleme işleri

Özellik Adı	Orijinal Format	Yeni Format	Yeni Format Açıklama
Yıldız Sayısı	Tamsayı [0-5]	Rasyonel Sayı	0 ve 1 arasındaki rasyonel sayı
TripAdvisor Değerlendirmesi	Tamsayı [0-10]	Rasyonel Sayı	0 ve 1 arasındaki rasyonel sayı
TripAdvisor Değerlendirme Sayısı	Tamsayı [0-10000]	Rasyonel Sayı	0 ve 1 arasındaki rasyonel sayı
Açıklama	Uzun metin	Rasyonel Sayı	Metin uzunluğu 0 ve 1 arasındaki rasyonel sayı
Resimler	URL dizisi	Rasyonel Sayı	Resimler sayısı 0 ve 1 arasındaki rasyonel sayısına dönüştürülmüştür
Oda olanakları	Olanaklar adı dizisi	Boole	Her olanak ayrı bir özelliktir. Olanak var ise 1, yoksa 0
Otel olanakları	Olanaklar adı dizisi	Boole	Her olanak ayrı bir özelliktir. Olanak var ise 1 yoksa 0.

İkinci adım ise ayrıştırma işidir. Otel verilerinde hiçbir özellik üzerinde ayrıştırma işi yapılamadığı için, bu adım uygulanmamıştır.

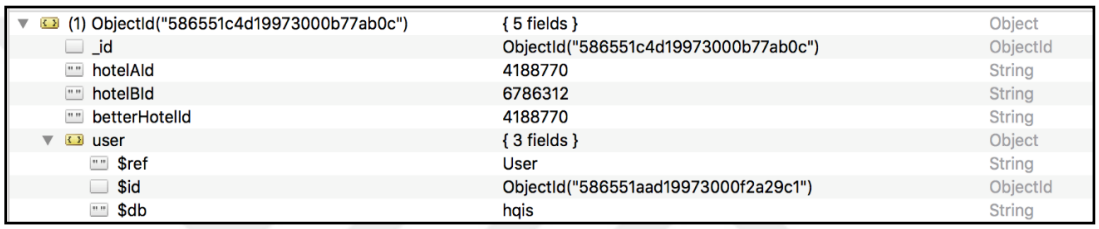
Son adım olarak da birleştirme işidir. Birçok özellikler üzerinde birleştirme işi yapılmıştır. Örneğin resimler, resimlerin URL'lerinin dizi formatında olduğu için, tüm resmi algoritmaya sunmak veri miktarını çok kat büyüteceği bilindiği için, URL metini olarak bir anlam taşımadığı için, resim sayısı olarak yeni özellikle temsil edilmiştir. Açıklama özelliğinde de benzer birleştirme işi yapılmıştır. Açıklama uzun metin olarak temsil edildiğinde bir anlamı taşımayabileceğini, metin üzerinde metin analiz yapma işi daha karmaşık olabileceği düşünüldüğü için, analiz yapıldığı anda bile ana problemi çözmede ektisi çok küçük olabileceği düşünüldüğü için, metin uzunluğunda yeni özellik eklenmiştir. Metin uzunluğu tamsayı olarak kullanıldığında sonuca etkisi çok büyük olabileceği bilindiği için, 0 ve 1 arasındaki rasyonel sayıya ölçekleme işi yapılmıştır.

Yukarıdaki bahsedilen üç türlü, ölçekleme, ayrıştırma, birleştirme dönüştürüm işinden başka ekstra dönüştürüm işi yapılmıştır. Sıralama, otomatik öneri işleri

üzerinde doğrudan doğruya hiçbir makine öğrenme algoritması uygulanamadığı için böyle ekstra adım yapılmıştır.

Bildiğimiz gibi makine öğrenme problemleri genelde üç türlü probleme ayrılır. Genelde literatürde ikiye ayırma yapılır. Birincisi denetimli öğrenme, ikincisi denetimsiz öğrenme ve son olarak da yarı denetimli öğrenme.

Otel öneri veri seti bir denetimli öğrenme kategorisine daha uygundur. Sınır ağı algoritması, doğrusal regresyon ve sınıflandırma algoritmalarına uydurma yapmaya çalışılmıştır. En sonunda çok basit bir çözüm bulunmuştur. Doğrusu veri toplama aşamasında topladığımız veri aşağıda gösterilmiştir.



Şekil 3.17. Otel karşılaştırma sonucu MongoDB veritabanındaki yapısı

Bu nedenle her bir karşılaştırma belgesi ayrıca bir sınıflandırma verisinin dizisi olarak temsil edebilir. MongoDB belge formatını makine öğrenme algoritmasına uygun formata dönüştürülmüş hali aşağıdaki gibidir.

Tablo 3.3. Karşılaştırma sonuçlarının makine öğrenmesi algoritmasına uygunlaştırmış hali

Birinci Otelin Özellikleri				İkinci Otelin Özellikleri				Sınıf
Yıldız Sayısı	TripAdvisor Değerlendirme Değeri	Açıklama	…	Yıldız Sayısı	TripAdvisor Değerlendirme Değeri	Açıklama	..	
3	6,6	Be our guest at this historic hotel…	…	4	88	Make yourself at home in one of the…	..	0
5	9,2	Located 400 metres from Saint-Méloir-des-Ondes …	…	5	9,0	Only 50 metres from Mitre Avenue…	..	1

Tablo 3.3'te görüldüğü gibi karşılaştırma sonucu basit bir sınıflandırma problemine dönüştürülmüştür. Verinin bu gibi formatta olması veri setinin özellik sayısını çok

yüksek yapmıştır. Bir otelin biçimlendirilmiş hali 162 özelliği içerirdi. Şimdi 162'den iki, yani 324 özellikli bir veriye dönüşmüş oldu.

Karşılaştırma sonuçlarını bir sınıflandırma veri setine dönüştürmede iki adım yapılmıştır. İlk olarak sistemde var olan tüm otel ve oda özellikleri toplanmıştır. Sonradan tüm karşılaştırma sonuçlarında mevcut olan otellerin oda ve otel özellikleri için bir, mevcut olmayan özellikler için sıfır değeri verilmiştir. Oda ve otel özellikleri benzersiz yüz elli yedi özellikten oluşmaktadır. Bir otelin özellik sayısı her bir olanağın ayrıca özelliği belirttiği için 162 olmuştur. Son olarak da iki otelin özellikleri yan yana birleştirilerek toplamda 324 özellik sayılı ve etiketin ise sıfır ve bir olarak belirlenmiştir. Eğer birinci otel ikinci otelden daha iyi ise etiket birdir, değilse etiket sıfırdır.

Genelde özelliklerin çok sayıda olması, eğitim veri setinin miktarı büyük olmasını gerektirir. Değilse makine öğrenme algoritma performansı düşük olma olasılığını yükseltir.

Bu yana kadar yapılan her bir adım veriyi analize hazırlama aşamasından ibaretti. Bundan sonraki her bir adım ise veri üzerinde makine öğrenme algoritmasını uygulamak, makineyi eğitmek, veri ölçekleme, veri ayrıştırma ve veri birleştirme yöntemlerini uygulayarak maksimum doğruluğu elde etmeye çalışmaktan ibarettir.

3.2.2. Uygulama

Bu alt bölümde bu yana kadar hazırlanan veri üzerinde geliştirilen uygulama hakkında detaylı anlatılacaktır.

3.2.2.1. Scikit-learn

Scikit-learn bir Python programlama dilinde yazılan ve açık kaynaklı, makine öğrenme kütüphanelerinde en popüler ve aktif olarak geliştirilmekte olan gelişmiş bir kütüphanedir. İlk olarak Google "Summer Of Code" projesinde David Cournapeau tarafından geliştirilmiştir. Kütüphane türlü sınıflandırma, regresyon ve kümeleme algoritmalarının uygulamasını içerir. "SVM", "random forest", "gradient boosting", "k-means" algoritmalarını da içerir. Scikti-learn kütüphanesi aşağıdaki özelliklerden dolayı kullanmaya karar verilmiştir:

1. Veri madenciliği ve veri analizi için kolay ve etkin araç sunar

2. Herkes için ulaşılabilir ve değişik çevrelerde tekrar kullanılabilir
3. NumPy, SciPy ve matplatlib üzerinde geliştirilmiştir
4. Açık kaynaklı, ticari olarak kullanmaya mümkün olan – BSD lisanslı

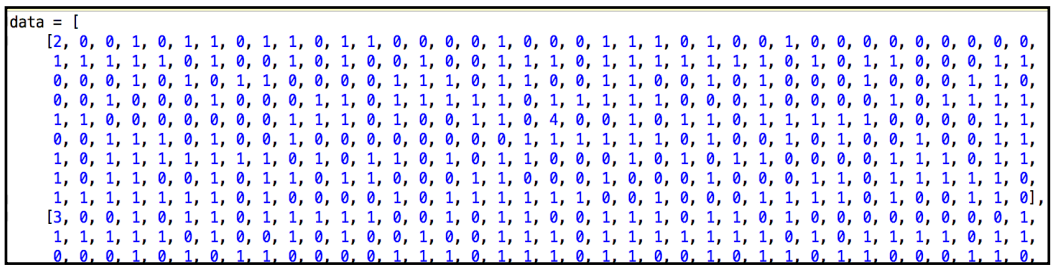
Kullanılan kütüphanenin en popüler olması internet ortamında çok bilginin olduğunu bildirir. Demek kütüphane üzerinde bir problem çıkınca çözmek kolay. TripRebel GmbH gibi İT şirketleri için kütüphane açık kaynaklı olması ve ticari maksatta kullanma imkânının sağlanması çok önemlidir.

3.2.2.2. Kodlama

Veri toplama platformunun en önemli parçası, hazırlanan veriyi otel öneri sistemine uygun formata çevirme sınıfıdır. Otel öneri sistemi için verinin dizi formatında ve aynı veri tipinde olması istenilen özelliklerdir. Veri formatını dönüştürücü sınıf sadece veriyi dışarı aktarma değil, ön işleme, düzenleme işlerinin çoğunluğunu uygular. Veri ne kadar istenilen formata yakınsa, analiz kısmı daha kolay olur. Sınıf EK-A'da gösterilmiştir.

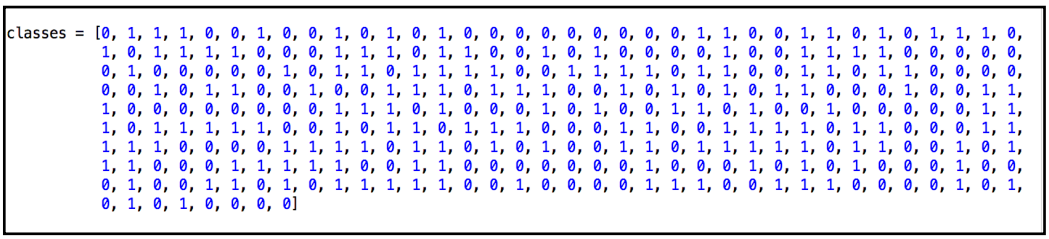
Hazırlanmış veri

İlk olarak hazırlanan veri seti Python programlama dilinde aşağıdaki gibidir.



Şekil 3.18. Veri seti

Şekil 3.18'te görüldüğü gibi en sonunda basit bir dizidir. Her sıra karşılaştırma yapılan iki otelin veri seçme, ön işleme, dönüştürme yaptıktan sonraki hâlinin birleştirilmiş biçiminde göstermektedir.



Şekil 3.19. Karşılaştırma sonuçları

Sınıflar ise ayrıca “classes” (Şekil 3.19) adındaki değişkende saklanmaktadır. Sınıfın sıfır olması ikinci otelin iyi olduğunu, bir olması birinci otelin iyi olduğunu bildirir.

SVC algoritması

SVC algoritmasından en önemli parametrelerden olan “kernel” adında parametre vardır. Kernel “linear”, “poly”, “rbf”, “sigmoid” “precomputed” değerlerinden birini alabilir. SVC algoritması “kernel” fonksiyonunu kullanarak giriş verisini çıkış verisine değişik yollarla değiştirme yapar.

Kernel fonksiyonlarının matematiksel gösterilişi aşağıdaki gibidir [16]:

1. Polinom kernel

Polinom kernel’i genelde görüntü işleme problemlerinde kullanılır. Denklem,

$$k(x_i,x_{j)} = (x_i \times x_j + 1)^d \tag{3.1}$$

d polinom derecesidir.

2. Gaussian kernel

Genel amaçlı kernel’dir; veri hakkında ön bilgi olmadığı zaman kullanılır. Denklem,

$$k(x,y) = \exp(-||x - y||^2 \times 0.5\sigma^{-2}) \tag{3.2}$$

3. Gaussian radial basis function (RBF)

Genel amaçlı kernel’dir; veri hakkında ön bilgi olmadığı zaman kullanılır. Denklem,

$$k(x_i,x_j) = \exp(-\gamma||x_i - x_j||^2) \tag{3.3}$$

4. Laplace RBF Kernel

Genel amaçlı kernel’dir; veri hakkında ön bilgi olmadığı zaman kullanılır,

$$k(x,y) = \exp(||x-y|| \times \sigma^{-1}) \tag{3.4}$$

5. Sigmoid kernel

Sinir ağları için proksi olarak kullanılabilir. Denklem,

$$k(x,y) = \tanh(\alpha x^T y + c) \tag{3.5}$$

6. Doğrusal spline

Çok seyrek veri vektörleri ile çalışırken faydalı. Metin kategorileştirmede de çok kullanılır. Regresyon problemlerinde de çok iyi sonuçları verir. Denklem,

$$k(x,y) = 1 + xy + xy\min(x,y) - (x+y)\times 0.5\min(x,y)^2 + 3^{-1}\min(x,y)^3 \quad (3.6)$$

Yukarıdaki anlatılan SVC kernel fonksiyonlarının (3.1), (3.2), (3.3), (3.4), (3.5), (3.6) uygulama kodu EK-B'de verilmiştir.

Şimdi EK-B'deki kod parçasını satır ne yaptığını anlatalım. Kodun ilk beş satırında sklearn kütüphanesinden bize lazım olan SVC modülünü, doğruluk puanı gösteren fonksiyonu, ön işlemi yapan "preprocessing" modülü ve veri seti eklenmiştir.

classes ve data değişkenlerine ise gerek olan veriler uygulanmıştır.

x_train ve y_train değişkenlerine ise verinin yüzde sekseni atanmıştır.

x_test ve y_test değişkenlerine ise verinin yüzde yirmisi sonra eğitilmiş modeli test etmek için atanmıştır.

Sonra da 20/80 biçiminde ayrılmış olan veri seti üzerinde SVC algoritması değişik kernel fonksiyonlarla uygulanmıştır.

```
clf = SVC(C=5.0, kernel='rbf')
clf.fit(x_train, y_train)
predicted = clf.predict(x_test)
print "Normal RBF", accuracy_score(y_test, predicted)
```

Şekil 3.21. SVC algoritmasının en basit uygulanması

SVC algoritması C adında parametreyi alır. C hata teriminin penaltı parametresidir. Algoritma kurulduktan sonra, fit adındaki fonksiyonu eğitim verisiyle çağrılır. Şimdi model eğitilmiştir.

Tablo 3.4. SVC kernel fonksiyonları sonuçları

Kernel Fonksiyon	Doğruluk Puanı
RBF	0,89552238806
Poly	0,865671641791
Linear	0,865671641791
Sigmoid	0,7910447761

Eğitim zamanında kullanılmayan veri yardımıyla modelin doğruluk oranını öğrenebiliriz. Doğruluk puanı ne kadar yüksek ise o kadar algoritma doğru çalışmaktadır demektir. SVC algoritmasını var olan tüm kernel fonksiyonlarıyla çalıştırdıktan sonra aşağıdaki tabloda gösterilmiş sonucu elde etmişizdir.

Yukarıdaki tabloda (Tablo 3.4) görüldüğü gibi RBF kernel fonksiyonunun doğruluk oranı en büyüktür. Ama gerçekten de RBF kernel fonksiyonu bizim problem için en uygun mudur? Yoksa veriyi eğitim ve test parçalarına bölerken RBF için en uygun biçimde bölünmüş müdür? Bu soruya cevap almak için veri setini 20 ve 80 oranına birkaç sefer rastgele ayırma yaparak, ayırma yapılmış veri üzerinde tüm kernel fonksiyonlarını çalıştırarak genel ortalama doğruluk oranını almamız lazım. Veriyi rastgele 20 defa bölerek ortalama doğruluk puanını ve her doğruluk puanın hesaplayan kod EK-C'de verilmiştir. EK-C'deki kod parçasını işlettikten sonra Tablo 3.5'teki gibi bir istatistiksel bir sonuç elde edilmiştir.

Tablo 3.5. Rastgele veri ayırması doğruluk oranı istatistiksel bilgi

RBF	0,87	0,85	0,84	0,9	0,88	0,81	0,84	0,87	0,84	0,81	0,79	0,88	0,82	0,85	0,85	0,87	0,84	0,82	0,79	0,84
Poly	0,88	0,84	0,82	0,91	0,87	0,81	0,81	0,84	0,88	0,81	0,78	0,87	0,81	0,87	0,85	0,85	0,84	0,87	0,79	0,85
Linear	0,85	0,87	0,76	0,87	0,9	0,81	0,82	0,79	0,87	0,85	0,84	0,81	0,81	0,88	0,81	0,82	0,84	0,84	0,82	0,76
Sigmoid	0,81	0,84	0,88	0,85	0,88	0,84	0,87	0,81	0,82	0,82	0,82	0,88	0,81	0,87	0,84	0,85	0,82	0,85	0,81	0,87

Ortalama doğruluk puanı ise aşağıdaki gibidir.

Tablo 3.6. Doğruluk puanı ortalaması

Kernel Fonksiyon	Ortalama Puan
RBF	0,843
Poly	0,8425
Linear	0,831
Sigmoid	0,842

Görüldüğü gibi RBF rastgele veri ayırması yapıldığı zaman da en yüksek doğruluk puanını vermektedir. Bu yana kadar edinebildiğimiz en yüksek doğruluk puanı 0,843 oldu.

Bir makine öğrenme mühendisinin soracağı sadece bir soru vardır. Yine doğruluk puanını yükseltmek mümkün mü? Bu soruya cevap bulmak için aşağıdaki kod parçası veri üzerinde gerçekleştirilmiştir. Bu kod kısaca değişle sadece kernel

parametresine değil C ve gamma parametreleri için de en uygun değer bulmaya çalışır.

Tablo 3.7. C=10 ve gamma 1,0 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu

RBF	0,64	0,58	0,66	0,67	0,67	0,67	0,64	0,57	0,55	0,49	0,6	0,58	0,6	0,69	0,64	0,73	0,49	0,49	0,51
Poly	0,79	0,79	0,82	0,76	0,85	0,85	0,85	0,81	0,75	0,78	0,81	0,72	0,81	0,78	0,78	0,73	0,78	0,79	0,76
Linear	0,85	0,87	0,79	0,88	0,84	0,84	0,84	0,78	0,87	0,84	0,84	0,73	0,76	0,88	0,84	0,78	0,84	0,85	0,73
Sigmoid	0,58	0,6	0,52	0,51	0,54	0,54	0,57	0,49	0,46	0,45	0,36	0,51	0,54	0,6	0,51	0,54	0,45	0,45	0,42

EK-D'deki kod çalıştırıldıktan sonra Tablo 3.7'deki gibi çıkışı elde edilir. C=10 ve gamma 1,0 olunca aşağıdaki sonuç elde edilmiştir.

Tablo 3.8. C=10 ve 1,0 olduğu zaman ortalama doğruluk puanı

Kernel Fonksiyon	Ortalama Puan
RBF	0,60
Poly	0,79
Linear	0,82
Sigmoid	0,51

C=10 ve gamma 0,005 olunca aşağıdaki sonuç elde edilmiştir.

Tablo 3.9. C=10 ve gamma 0,005 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu

RBF	0,75	0,84	0,82	0,78	0,78	0,81	0,82	0,85	0,87	0,88	0,82	0,82	0,85	0,87	0,82	0,85	0,85	0,84	0,85
Poly	0,75	0,79	0,79	0,78	0,79	0,84	0,84	0,84	0,84	0,88	0,84	0,84	0,81	0,85	0,84	0,87	0,85	0,82	0,82
Linear	0,79	0,81	0,84	0,7	0,78	0,84	0,88	0,85	0,78	0,87	0,82	0,88	0,81	0,84	0,81	0,84	0,85	0,81	0,73
Sigmoid	0,6	0,67	0,75	0,58	0,72	0,7	0,7	0,72	0,73	0,66	0,64	0,7	0,69	0,66	0,75	0,67	0,72	0,64	0,73

Tablo 3.10. C=10 ve 0,005 olduğu zaman ortalama doğruluk puanı

Kernel Fonksiyon	Ortalama Puan
RBF	0,83
Poly	0,83
Linear	0,82
Sigmoid	0,69

C=10 ve gamma 0,0025 olunca aşağıdaki sonuç elde edilmiştir.

Tablo 3.11 C=10 ve gamma 0,0025 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu

RBF	0,85	0,85	0,82	0,81	0,87	0,82	0,87	0,85	0,91	0,87	0,82	0,79	0,81	0,76	0,88	0,85	0,81	0,85	0,9
Poly	0,88	0,91	0,82	0,81	0,87	0,78	0,87	0,85	0,9	0,9	0,82	0,81	0,76	0,79	0,88	0,85	0,75	0,85	0,91
Linear	0,84	0,85	0,81	0,81	0,85	0,79	0,87	0,79	0,91	0,82	0,75	0,75	0,84	0,76	0,85	0,85	0,78	0,9	0,81
Sigmoid	0,84	0,82	0,84	0,88	0,81	0,76	0,84	0,85	0,9	0,85	0,79	0,82	0,82	0,76	0,9	0,84	0,81	0,88	0,88

Tablo 3.12. C=10 ve 0,0025 olduğu zaman ortalama doğruluk puanı

Kernel Fonksiyon	Ortalama Puan
RBF	0,84
Poly	0,84
Linear	0,82
Sigmoid	0,84

C=10 ve gamma 0,0017 olunca aşağıdaki sonuç elde edilmiştir.

Tablo 3.13. C=10 ve gamma 0,0017 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu

RBF	0,87	0,88	0,9	0,81	0,81	0,85	0,81	0,82	0,88	0,93	0,82	0,82	0,9	0,78	0,84	0,82	0,85	0,88	0,81
Poly	0,84	0,85	0,88	0,76	0,76	0,85	0,79	0,84	0,85	0,9	0,78	0,88	0,84	0,76	0,87	0,81	0,85	0,88	0,75
Linear	0,87	0,9	0,97	0,82	0,82	0,79	0,81	0,85	0,81	0,9	0,84	0,78	0,88	0,76	0,84	0,81	0,79	0,85	0,88
Sigmoid	0,85	0,88	0,9	0,84	0,81	0,87	0,87	0,88	0,87	0,91	0,82	0,84	0,87	0,76	0,84	0,82	0,84	0,9	0,79

Tablo 3.14. C=10 ve 0,0017 olduğu zaman ortalama doğruluk puanı

Kernel Fonksiyon	Ortalama Puan
RBF	0,85
Poly	0,83
Linear	0,84
Sigmoid	0,85

C=10 ve gamma 0,0012 olunca aşağıdaki sonuç elde edilmiştir.

Tablo 3.15. C=10 ve gamma 0,0012 olduğu zaman rastgele eğitim setinin bölünmesi ve doğruluk puan tablosu

RBF	0,87	0,87	0,79	0,82	0,87	0,75	0,93	0,81	0,85	0,88	0,9	0,82	0,81	0,84	0,87	0,84	0,81	0,9	0,82
Poly	0,78	0,76	0,7	0,64	0,78	0,67	0,79	0,69	0,69	0,81	0,78	0,72	0,75	0,78	0,78	0,82	0,76	0,84	0,72
Linear	0,85	0,85	0,87	0,84	0,84	0,69	0,94	0,81	0,85	0,85	0,85	0,84	0,81	0,84	0,84	0,88	0,82	0,9	0,84
Sigmoid	0,88	0,87	0,81	0,81	0,88	0,73	0,93	0,81	0,79	0,9	0,85	0,81	0,9	0,84	0,9	0,85	0,85	0,91	0,81

Tablo 3.16. C=10 ve 0,0012 olduğu zaman ortalama doğruluk puanı

Kernel Fonksiyon	Ortalama Puan
RBF	0,84
Poly	0,75
Linear	0,84
Sigmoid	0,85

Yukarıdaki en yüksek doğruluk puanı sunan algoritma parametre değerlerini arama sonuçları üzerinden ilginç analizler yapılabilir. İlk olarak, 0,843 en yüksek puan olmadığını görürüz. Yeni en yüksek puan ise 0,85'tir.

RBF yine de en yüksek puanı gösteren kernel fonksiyonu olduğunu ikinci defa kanıtlamıştır. Gamma değeri sıfıra yaklaşınca sigmoid kernel fonksiyonu da en yüksek doğruluğu vermektedir. Demek parametrelere doğru değerleri vererek doğruluk oranın yükseltmek mümkün.

Elde ettiğimiz sonuçlar yüksek doğruluk puanını göstermektedir. Ancak şimdilik hangi özelliklerin daha büyük etki gösterdiğini bilmiyoruz. Özellik etkilerini öğrenmek için etkisi büyük olduğunu düşündüğümüz özellikleri seçerek yeni model oluşturarak doğruluk puanını ölçmemiz gerekmektedir.

Otel yıldız sayısı, TripAdvisor ortalama değerlendirmesi, TripAdvisor değerlendirme sayılarının müşteri kararına çok etkisi olduğunu düşündüğümüz için sadece bu özellikleri seçerek yeni model oluşturmaya çalıştık. Sonuçlar gerçekten de bu özelliklerin etkisi büyük oluğunu kanıtlamaktadır.

Tablo 3.17. Sınırlı özellikler kullanılarak elde edilen modelin doğruluk oranı

Kernel Fonksiyon	Ortalama Puan
RBF	0,80
Poly	0,79
Linear	0,79
Sigmoid	0,78

İki oteli karşılaştırdığımızda eğer birinci otelin yıldız sayısı yüksek ise, o otelin çoğu zaman ikinci otelden daha iyi olduğunu biliyoruz. Bunu kanıtlamak için sadece veri setinden sadece otel yıldız sayılarını seçerek makine öğrenme modelini elde edelim. Sonuçlar gerçekten ilginç. Aşağıdaki Tablo 3.18'de gördüğünüz gibi doğruluk oranı maksimum 0,65'i göstermektedir.

Tablo 3.18. Sadece yıldız sayıları ile eğitilmiş makine öğrenme modeli

Kernel Fonksiyon	Ortalama Puan
RBF	0,65
Poly	0,65
Linear	0,64
Sigmoid	0,63

Sonuç olarak, kurduğumuz model ile en yüksek 0,85 doğruluk puanına erişile bilinmiştir. Yüzde seksen beş makine öğrenme alanında iyi bir performans olarak bilinmektedir.

4. SONUÇLAR VE ÖNERİLER

Bu tez çalışmasının amacı; TripRebel GmbH alman şirketinin kurulmasının temel nedeni olan otomatik otel yükseltici, otomatik daha iyi otele değişim yapma işini otomatikleştirmekti. Şirketinin ilk açıldığından beri seyahat acentelerinin manuel yükseltme, değiştirme işini makine öğrenme tekniklerini kullanarak sağlam, doğru sonuçları sunan makine modelini sunmaktı.

Uygulama geliştirmede ilk önce makine öğrenme için temel gereksinimlerden olan veriyi toplayan platformu geliştirme, platform üzerinde seyahat acenteleri için iki rastgele ama benzer seviyedeki oteli karşılaştırma olanağını sunma, yeterince veri toplandıktan sonra veri üzerinden ön işleme, dönüştürme işlerini uygulayarak makine öğrenme algoritmaları için uygun formatta veri hazırlama, veri üzerinde en yüksek doğruluk puanın gösteren makine öğrenme modelini hazırlama işi de amaçlanmıştı.

Tez üç ana bölümden oluşmaktadır. Birinci bölüm makine öğrenmesi hakkında temel kavramlar, kullanılan algoritmalar, makine öğrenmede sıralama problemleri için yer verilmiştir. İkinci ana bölümde ise genel literatür taraması, ilgili çalışmalar yer almıştır. Problemi çözmeye çalışmadan önce var olan çözümler üzerinde araştırma detaylıca yapılmıştır. Araştırmanın sonucunda hemen hemen hepsi tezin çözmeye amaçladığı problem üzerinde doğrudan uygulanamadığı kanıtına varılmıştır. Üçüncü bölümde veri toplama web platformu ve makine öğrenme modeli geliştirilmiştir. İki sistem hakkında detaylı açıklama yapılmıştır.

Uygulama iki ayrı sistemden oluşmuştur. Birinci sistem veri toplamak için kullanılırsa, ikincisi ise toplanmış veri üzerinden makine öğrenme modeli kurar ve otomatik otel öneri, karşılaştırma yapar. Birinci sistem PHP, Javascript (Vue.js), MongoDB, Nginx teknolojileri üzerinde kurulmuştur. İkinci sistem Python ve Sklearn makine öğrenme kütüphanesi yardımıyla geliştirilmiştir. Makine öğrenme sistemi açık olarak sadece bir API sunmaktadır. API iki otel karşılaştırma sonucunu sunar. TripRebel şirketinin otel rezervasyon yapma ve otomatik otel değiştirme, otel yükseltme işlerini gerçekleştiren sistemler API sorgu sonuçlarına göre karar verebilmektedir ve gereken ekstra iş mantığını uygulamaktadır.

Otel öneri sisteminin doğruluk oranı 85% erişilmiştir. Modelin birinci versiyonunda sadece daha iyi olan otelin ID'sini geri göndermektedir. Sistemin gelecek versiyonunda kararın ne kadar olasılıkla verildiği hakkında bilgi verecek olanağı ekleme planlanmıştır. Çünkü birinci otelin ikinci otelden daha iyi olma olasılığı yüzde 51-55% ise, genelde otel değiştirmeye gerek yoktur. Müşterilerin seçimi ile oynamak, istediğimiz kadarıyla otomatik değiştirmek çok hassas bir konudur. Bu nedenle, otomatik değiştirme yapmandan önce sistem müşterinin yüzde kaç oranda yeni değişmelerle mutlu kalacağını hesaplamak çok önemlidir.

Sonunda, tezin anan konusu olan otel öneri sistemi başarıyla tamamlanmıştır. Model kurulmadan önce doğruluk oranı en azında 96% olması temel bir gereksinimlerdendi. Makine öğrenmesinde bazı problemler ne kadar algoritmayı ayarlama yapsan bile, veri üzerinden en doğru ön işleme, veri seçme, dönüştürme işleri yapılsa bile bazen doğruluk puanını yükseltmek tek bir ekstra adıma bağlıdır: veri miktarını çoğaltmak. Maalesef, bu iş genelde çok zaman gerektirir. Seyahat acenteleri manuel otel karşılaştırma işini yine de devam ederek, makine öğrenme verisini her gün çoğaltmaktadır. Bundan dolayı makine öğrenme modeli her gün gelişmeyi devam etmekte ve doğruluk oranını yükseltmektedir.

KAYNAKLAR

[1] Kroshilina V., Maini D., Khantal N., Rank Hotels on Expedia.com to Maximize Purchases, 14 Aralık 2013.

[2] Ryosuke S., Yoshihiro H., Hiroshi T., Hotel Recommender System Based on User's Preference Transition, *2008 IEEE International Conference on Systems*, Man and Cybernetics, 2008, DOI: 10.1109/ICSMC.2008.4811660.

[3] Michael A., John P., Michael S., The Automated Travel Agent: Hotel Recommendations Using Machine Learning, 2016.

[4] Anindya G., Panagiotis G., Beibei L., Designing Ranking Systems for Hotels on Travel Search Engines to Enhance User Experience, *International Conference on Information Systems,* 2010.

[5] Marchiori E., Eynard D., Inversini A., Cantoni L., Cerretti F., Harvesting Online Contents: an Analysis of Hotel Reviews Websites, *Information and Communication Technologies in Tourism*, Vienna, Austria, 05-06 Mayıs 2011

[6] Otter T., Dolnicar S., Which Hotel Attributes Matter? A Review of Previous and a Framework for Future Research, *Proceedings of The 9th Annual Conference of The Asia Pacific Tourism Association (APTA),* 2003, 1, 178-188.

[7] https://www.altexsoft.com/blog/datascience/customer- experience - personalization-in-travel-and-hospitality-using-behavioral-analytics-and-machine-learning (Ziyaret tarihi: 05 Mayıs 2018)

[8] https://medium.com/deep-learning-turkiye/makine-%C3%B6%C4%9Frenmesi-e%C4%9Flencelidir-b9d50aad3a62 (Ziyaret tarihi: 25 Mart 2018)

[9] http://scikit-learn.org/stable/modules/svm.html (Ziyaret tarihi: 27 Mart 2018).

[10] Christopher D., Raghavan P., Schütze H., Manning, *Introduction to Information Retrieval*, Cambridge: Cambridge University Press, New York,100-115, 2009.

[11] Dammak F., Kammoun H., Hamadou A., Improving Pairwise Learning to Rank Algorithms for Document Retrieval, *IEEE 2017 Symposium Series*, Honolulu, HI, USA, 01 Aralık 2018.

[12] https://medium.com/netflix-techblog/netflix-recommendations-beyond-the-5-stars-part-1-55838468f429 (Ziyaret tarihi: 26 Mart 2018).

[13] https://medium.com/netflix-techblog/netflix-recommendations-beyond-the-5-stars-part-2-d9b96aa399f5 (Ziyaret tarihi: 26 Mart 2018).

[14] Papini T., SortNet: Learning to Rank by a Neural Preference Function, *IEEE Transactions on Neural Networks*, 2011, **22**(9), 1368-1380.

[15] Tiakas E., Papadopoulos A., Manolopoulos Y., Skyline Queries: an Introduction, *Information, Intelligence, Systems and Applications (IISA)*, *6th International Conference*, Corfu, Greece, 10 Temmuz 2015.

[16] https://data-flair.training/blogs/svm-kernel-functions/ (Ziyaret tarihi: 01 Mayıs 2018).

[17] Walek B., Hosek O., Farana R., Proposal of Expert System for Hotel Booking System, *17th International Carpathian Control Conference (ICCC)*, Tatranska Lomnica, 29 Mayıs – 1 Haziran 2016.

[18] Anindya G., Panagiotis G., Beibei L., Designing Ranking Systems for Hotels on Travel Search Engines by Mining User-Generated and Crowd-Sourced Content, *Marketing Science*, 2012, **33**(1), 493-520.

[19] Hang L., A Short Introduction to Learning to Rank, *IEICE Transactions*, 2011.

EKLER

EK-A

DataGenerator sınıfı. MongoDB koleksiyon formatında saklanan işlenmemiş veriyi makine öğrenme platformu için uygun formata çevirir. Veri analizinin, makine öğrenme yönteminin en önemli olan adımı ön işleme işini de yüzde doksanını uygular.

```php
<?php

namespace AppBundle\Util;

use AppBundle\Document\BetterHotel;
use AppBundle\Document\Queue;
use AppBundle\HotelRepositoryAPI\HotelRepositoryAPI;
use Doctrine\ODM\MongoDB\Query\Builder;
use Doctrine\ORM\Query;
use Symfony\Component\DependencyInjection\ContainerInterface;

class DataGenerator
{
    const HOTEL_NUMBER_PER_REQUEST = 100;
    const HOTEL_REPOSITORY_URL = 'http://localhost:8000/hotels/';

    private $hotelsAsAssociativeArray = [];
    private $amenities = [];
    private $baseTemplate = [
        'externalId'        => 0,
        'stars'             => 0,
        'distances'         => array(),
        'imageUrls'         => 0,
        'description'       => 0,
        'tripAdvisorRating' => 0,
        'tripAdvisorRatingCount'
    ];

    private $hotels = [];
    private $data = [];

    /** @var  ContainerInterface */
    private $container;
    private $usernames;
    private $tripAdvisorRatings;
    private $hotelIds;

    public function __construct(ContainerInterface $containerInterface)
    {
        $this->container = $containerInterface;
    }

    public function setUsernames($usernames)
    {
        if (is_array($usernames)) {
            $this->usernames = $usernames;
        } elseif (is_string($usernames)) {
            $this->usernames = explode(',', $usernames);
        }
    }

    public function getData()
    {
        $this->populateHotels();
```

```
        $this->populateTripAdvisor();
        $this->populateAmenities();
        $this->normalizeHotels();
        $this->prepareRatings();
        return [$this->getLabels(), $this->data];
    }

    private function populateHotels()
    {
        $hotelIdArray = $this->getHotelIDs();
        while (sizeof($hotelIdArray) > 0) {
            $hotelArrayChunk = array_splice($hotelIdArray, 0,
self::HOTEL_NUMBER_PER_REQUEST);
            $hotelIdsAsString = implode(',', $hotelArrayChunk);
            $hotelsAsArray =
HotelRepositoryAPI::getHotels($hotelIdsAsString);
            $this->hotelsAsAssociativeArray = array_merge($this-
>hotelsAsAssociativeArray, $hotelsAsArray);
        }
    }

    private function populateAmenities()
    {
        foreach ($this->hotelsAsAssociativeArray as $hotel) {
            $this->amenities = array_merge($this->amenities,
array_flip($hotel['Amenities']));
        }
        $this->amenities = array_keys($this->amenities);
        sort($this->amenities, SORT_NATURAL | SORT_FLAG_CASE);
        $this->amenities = array_map(function ($amenity) {
            return str_replace(',', '.', $amenity);
        }, $this->amenities);
    }

    private function getEmptyHotelTemplate()
    {
        return array_merge($this->baseTemplate, array_fill_keys($this-
>amenities, 0));
    }

    private function normalizeHotels()
    {
        $emptyHotelTemplate = $this->getEmptyHotelTemplate();
        foreach ($this->hotelsAsAssociativeArray as $hotel) {
            $emptyHotelTemplate['stars'] = $hotel['Stars'];
            $emptyHotelTemplate['externalId'] = $hotel['ExternalId'];
            foreach ($hotel['amenities'] as $amenity) {
                $emptyHotelTemplate[$amenity] = 1;
            }
            $emptyHotelTemplate['imageUrls'] = sizeof($hotel['ImageUrls']);
            $emptyHotelTemplate['distances'] = $hotel['Distances'];
            $emptyHotelTemplate['description'] =
strlen($hotel['Description']);
            $emptyHotelTemplate['tripAdvisorRating'] = $this-
>tripAdvisorRatings[$hotel['ExternalId']]->getTripAdvisorRating();
            $emptyHotelTemplate['tripAdvisorRatingCount'] = $this-
>tripAdvisorRatings[$hotel['ExternalId']]->getTripAdvisorRatingCount();
            $this->hotels[$emptyHotelTemplate['externalId']] =
$emptyHotelTemplate;
        }
    }

    private function prepareRatings()
    {
        $manager = $this->container-
```

```
>get('doctrine_mongodb.odm.default_document_manager');
        /** @var Builder $queryBuilder */
        $queryBuilder = $manager->getRepository('AppBundle:BetterHotel')-
>createQueryBuilder();
        foreach ($this->getUsers() as $user) {
            $queryBuilder->field('user')->references($user);
            $betterHotels = $queryBuilder->field('betterHotelId')-
>notEqual(null)->getQuery()->execute();
            /** @var BetterHotel $betterHotel */
            foreach ($betterHotels as $betterHotel) {
                if (!array_key_exists($betterHotel->getHotelAId(), $this-
>hotels) ||
                    !array_key_exists($betterHotel->getHotelBId(), $this-
>hotels)
                ) {
                    continue;
                }
                $firstHotel = $this->hotels[$betterHotel->getHotelAId()];
                $secondHotel = $this->hotels[$betterHotel->getHotelBId()];
                if (rand(1, 2) === 1) {
                    $temp = $secondHotel;
                    $secondHotel = $firstHotel;
                    $firstHotel = $temp;
                }
                $class = $betterHotel->getBetterHotelId() ==
$firstHotel['externalId'] ? 1 : 0;
                ksort($firstHotel, SORT_NATURAL | SORT_FLAG_CASE);
                ksort($secondHotel, SORT_NATURAL | SORT_FLAG_CASE);

                $this->data[] = array('first_hotel' => $firstHotel,
'second_hotel' => $secondHotel, 'class' => $class);
            }
        }
    }

    private function getLabels()
    {
        $labels = $this->getEmptyHotelTemplate();
        ksort($labels, SORT_NATURAL | SORT_FLAG_CASE);
        return array_keys($labels);
    }

    private function getUsers()
    {
        $manager = $this->container-
>get('doctrine_mongodb.odm.default_document_manager');
        /** @var Builder $queryBuilder */
        $queryBuilder = $manager->getRepository('AppBundle:User')-
>createQueryBuilder();

        if (sizeof($this->usernames) > 0 && strlen($this->usernames[0]) >
0) {
            $queryBuilder->field('username')->in($this->usernames);
        }

        $users = $queryBuilder->getQuery()->execute()->toArray();
        return array_values($users);
    }

    private function populateTripAdvisor()
    {
        $this->tripAdvisorRatings = array();
        $queues = $this->container-
>get('doctrine_mongodb.odm.default_document_manager')-
>getRepository('AppBundle:Queue')
```

```
            ->findAll();
        /** @var Queue $queue */
        foreach ($queues as $queue) {
            $this->tripAdvisorRatings[$queue->getHotelId()] = $queue;
        }
    }

    public function setHotelIds($ids)
    {
        $this->hotelIds = $ids;
    }

    private function getHotelIDs()
    {
        return $this->container-
>get('doctrine_mongodb.odm.default_document_manager')
            ->getRepository('AppBundle:Queue')->findAllHotelIdsAsArray();
    }
}
```

EK-B

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn import preprocessing
from data import get_classes, get_data
from sklearn.model_selection import train_test_split

classes = get_classes()
data = get_data()

eightyPercent = int(len(data) * 0,8)

x_train = data[:eightyPercent]
x_test = data[eightyPercent:]

y_train = classes[:eightyPercent]
y_test = classes[eightyPercent:]

# Normal RBF
clf = SVC(C=5.0, kernel='rbf')
clf.fit(x_train, y_train)
predicted = clf.predict(x_test)
print "Normal RBF", accuracy_score(y_test, predicted)

# Normal POLY
clf = SVC(C=5.0, kernel='poly')
clf.fit(x_train, y_train)
predicted = clf.predict(x_test)
print "Normal POLY", accuracy_score(y_test, predicted)

# Normal LINEAR
clf = SVC(C=5.0, kernel='linear')
clf.fit(x_train, y_train)
predicted = clf.predict(x_test)
print "Normal LINEAR", accuracy_score(y_test, predicted)
```

Şekil B.1. SVC algoritmasının kodu

EK-C

```
dictionary = [0,0,0,0]
dictionary[0] = []
dictionary[1] = []
dictionary[2] = []
dictionary[3] = []
for x in range(0,20):
    x_train, x_test, y_train, y_test = train_test_split(data, classes, train_size=0,8,
                                        random_state=randint(1, len(data)))

    # Random RBF
    clf = SVC(C=5.0, kernel='rbf')
    clf.fit(x_train, y_train)
    predicted = clf.predict(x_test)
    rbfScore = round(accuracy_score(y_test, predicted),2)
    dictionary[0].append(rbfScore)
    print "Random RBF", rbfScore

    # Normal POLY
    clf = SVC(C=5.0, kernel='poly')
    clf.fit(x_train, y_train)
    predicted = clf.predict(x_test)
    polyScore = round(accuracy_score(y_test,predicted), 2)
    dictionary[1].append(polyScore)
    print "Random POLY", polyScore

    # Normal LINEAR
    clf = SVC(C=5.0, kernel='linear')
    clf.fit(x_train, y_train)
    predicted = clf.predict(x_test)
    linearScore = round(accuracy_score(y_test, predicted), 2)
    dictionary[2].append(linearScore)
    print "Random LINEAR", linearScore

    # Random SIGMOID
    clf = SVC(C=5.0, kernel='sigmoid')
    clf.fit(x_train, y_train)
    predicted = clf.predict(x_test)
    sigmoidScore = round(accuracy_score(y_test, predicted), 2)
    dictionary[3].append(sigmoidScore)
    print "Random SIGMOID", sigmoidScore


for values in dictionary:
    print values

print "Average"
for values in dictionary:
    print mean(values)
```

Şekil C-1. Rastgele veri ayırması

EK-D

```
penaltyC = 5.0
for x in range(1,2):
   penaltyC=10*x

   for y in range(1,1000,100):
      x_train, x_test, y_train, y_test = train_test_split(data, classes, train_size=0,8,
                                          random_state=randint(1, len(data)))
      dictionary = [0, 0, 0, 0]
      dictionary[0] = []
      dictionary[1] = []
      dictionary[2] = []
      dictionary[3] = []
      for x in range(1, 20):
         # Random RBF
         clf = SVC(C=penaltyC, kernel='rbf', gamma=1.0/y)
         clf.fit(x_train, y_train)
         predicted = clf.predict(x_test)
         rbfScore = round(accuracy_score(y_test, predicted), 2)
         dictionary[0].append(rbfScore)

         # Normal POLY
         clf = SVC(C=penaltyC, kernel='poly', gamma=1.0/y)
         clf.fit(x_train, y_train)
         predicted = clf.predict(x_test)
         polyScore = round(accuracy_score(y_test, predicted), 2)
         dictionary[1].append(polyScore)
         #
         # # Normal LINEAR
         clf = SVC(C=penaltyC, kernel='linear', gamma=1.0/y)
         clf.fit(x_train, y_train)
         predicted = clf.predict(x_test)
         linearScore = round(accuracy_score(y_test, predicted), 2)
         dictionary[2].append(linearScore)
         #
         # # Random SIGMOID
         clf = SVC(C=penaltyC, kernel='sigmoid', gamma=1.0/y)
         clf.fit(x_train, y_train)
         predicted = clf.predict(x_test)
         sigmoidScore = round(accuracy_score(y_test, predicted), 2)
         dictionary[3].append(sigmoidScore)
      for values in dictionary:
         print values

      print "Average accuracy score for penalty C = " + str(penaltyC) + " and gamma = " +
str(round(1.0/y, 4))
      for values in dictionary:
         print mean(values)
```

Şekil D.1. SVC algoritmasının değişik kernel fonksiyonlarıyla uygulanması

KİŞİSEL YAYIN VE ESERLER

[1] **Nursultan T.**, Burcu B., Gencay D., Pınar Onay D., Davut İ., Alev M., Araştırma Eğilimleri: Bir Sistematik Eşleme Çalışması, *UYMS,* İzmir, Türkiye, 9-11 Eylül 2015.

ÖZGEÇMİŞ

1992 yılında Kırgızistan'da doğdu. İlk, orta ve lise öğrenimini Batkende tamamladı. 2010 yılında girdiği Kırgızistan Türkiye "Manas" Üniversitesi Bilgisayar Mühendisliği Bölümünü 2014 yılında bölüm birincisi olarak tamamladı. Aynı yıl Kocaeli Üniversitesi Fen Bilimleri Enstitüsü altında Bilgisayar Mühendisliği Bölümünde Yüksek Lisans eğitimine başladı. 2015-2016 arası Hollanda, Groningen şehrinde, Axtion şirketinde Yazılım Mühendisi olarak staj yaptı. Sonra üç ay Almanya'nın Hamburg şehrinde TripRebel şirketinde tez araştırma projesi üzerinde çalıştı. Şimdi Almanya, Hamburg'da AboutYou GmbH şirketinde Yazılım Mühendisi olarak çalışmaktadır.