TOPLU TAŞIMA ARAÇLARININ DURAKLARA VARIŞ SÜRELERİNİN TAHMİNİNDEKİ SAPMALARI İYİLEŞTİRMEYE YÖNELİK BİR VERİ MADENCİLİĞİ UYGULAMASI (İETT ÖRNEĞİ)

Kadir KUŞTUL
111450101

DOKTORA TEZİ

Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Doktora
Danışman: Prof. Dr. Emin Murat ESİN

İstanbul
T.C. Maltepe Üniversitesi
Lisansüstü Eğitim Enstitüsü
Aralık, 2022

TOPLU TAŞIMA ARAÇLARININ DURAKLARA VARIŞ SÜRELERİNİN TAHMİNİNDEKİ SAPMALARI İYİLEŞTİRMEYE YÖNELİK BİR VERİ MADENCİLİĞİ UYGULAMASI (İETT ÖRNEĞİ)

Kadir KUŞTUL
111450101
ORCID: 0000-0002-2657-3814

DOKTORA TEZİ
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Doktora
Danışman: Prof. Dr. Emin Murat ESİN

İstanbul
T.C. Maltepe Üniversitesi
Lisansüstü Eğitim Enstitüsü
Aralık, 2022



JÜRİ VE ENSTİTÜ ONAYI

Bu belge, Yükseköğretim Kurulu tarafından 19.01.2021 tarihli "Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge" ile bildirilen 6689 Sayılı Kişisel Verilerin Korunması Kanunu kapsamında gizlenmiştir.

ETİK İLKE VE KURALLARA UYUM BEYANI

Bu belge, Yükseköğretim Kurulu tarafından 19.01.2021 tarihli "Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge" ile bildirilen 6689 Sayılı Kişisel Verilerin Korunması Kanunu kapsamında gizlenmiştir.

TEŞEKKÜR

Tez çalışmam süresince her zaman anlayışla ve sabırla yol gösteren, destek ve yardımlarını esirgemeyen değerli danışman hocam Sn. Prof. Dr. Emin Murat ESİN'e ve Maltepe Üniversitesi Bilgisayar Mühendisliği Bölümü hocalarıma teşekkürlerimi sunarım. Çalışmamda her türlü desteği sağlayan İETT ailesine özellikle yönetici ve mesai arkadaşlarıma teşekkürlerimi sunarım.

Akademik hayatımın her aşamasında beni yalnız bırakmayan ve çalışmalarımda başarılı olmamı sağlayan sevgili eşime ve aileme sonsuz teşekkür ederim.

Kadir KUŞTUL

Aralık,2022

ÖZET

TOPLU TAŞIMA ARAÇLARININ DURAKLARA VARIŞ SÜRELERİNİN TAHMİNİNDEKİ SAPMALARI İYİLEŞTİRMEYE YÖNELİK BİR VERİ MADENCİLİĞİ UYGULAMASI (İETT ÖRNEĞİ)

Kadir Kuştul
Doktora Tezi
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Programı
Danışman**:** Prof. Dr. Emin Murat Esin
Maltepe Üniversitesi Lisansüstü Eğitim Enstitüsü, 2022

Metropollerde yaşayan insanların zamanının büyük bir kısmı şehir içi yolculuklarda geçmektedir. Bu durum, İstanbul gibi büyük metropollerde kişisel ulaşım planlaması yapmayı zorunlu kılar.

İstanbul da ulaşım için kullanılan Akıllı Kart verilerine göre en çok tercih edilen ulaşım türü yaklaşık %50'lik oranla otobüslerdir. Bu nedenle yolcunun ulaşım planını sağlıklı biçimde yapabilmesi ve kurumsal kaynakların verimli kullanılabilmesi için Akıllı ulaşım sistemleri oluşturulmuştur. Bu kapsamda verilen hizmetlerden birisi de otobüslerin duraklara varış zamanını dinamik olarak tahmin etmek ve bu verileri mobil uygulama ya da akıllı durak gibi platformlarda kullanıcısı ile paylaşmaktır.

Sistem, hizmetteki her bir otobüsün planlanmış sefer başlangıç saatinden itibaren güzergâhı üzerindeki her bir duraktan bir sonrakine varışı için geçecek zaman aralığını tahmin etmek ve kümülatif toplamı mutlak zaman cinsinden yolculara bildirmek fikrine dayanmaktadır. Bu süreler için yapılan tahminler, otobüsün güzergâhında ilk duraklar için daha isabetli olmasına karşılık sonraki duraklar için hata payı artmaktadır. Yolculuk süresi; yolculuğun yapıldığı zaman dilimi, bu zaman diliminde güzergâhın trafik yoğunluğu, yolcu sayısı, şoförün sürüş performansı, araç tipi gibi birçok parametreye bağlıdır. Yolculuk verilerinin kayıtlarına bakıldığında bu parametrelerin etkisiyle tekrarlanan patternlerin varlığı görülmektedir.

Bu çalışmada, yolculuk sürelerinin tahmininde geçmiş kayıtlara dayalı genel geçer istatistikler yerine, daha ayrıntılı kümelendirilmiş parametrelere dayalı bir yöntem teklif edilmektedir.

Önerilen yöntemi geliştirmek ve sınamak için İstanbul'da bir otobüs hattı, örnek veri seti olarak ele alınmıştır. Geçmiş veriler, veri madenciliğine dayalı olarak değerlendirilmiştir. Bu yöntemle yapılan tahminler halen kullanılan yöntemlerle yapılmış yolculuk süresi tahminleri ile kıyaslanarak elde edilen sonuçlar tartışılmıştır.

Anahtar Sözcükler: Kümeleme Algoritmaları, Veri Madenciliği, Akıllı Ulaşım, Seyahat süresi tahmini, Toplu Taşıma.

ABSTRACT

A DATA MINING APPLICATION FOR IMPROVING THE DEVIATIONS IN THE ESTIMATION OF TIME OF ARRIVAL OF PUBLIC TRANSPORT VEHICLES (IETT EXAMPLE)

Kadir Kuştul
PhD Thesis
Department of Computer Engineering
Computer Engineering Programme
Thesis Advisor: Prof. Dr. Emin Murat Esin
Maltepe University Graduate School, 2022

A large part of the time of people living in metropolises is spent on urban journeys. This situation necessitates personal transportation planning in big metropolises such as Istanbul.

According to the Smart Card data used for transportation in Istanbul, the most preferred mode of transportation is buses with a ratio of approximately 50%. For this reason, Intelligent transportation systems have been created in order for the passenger to make a healthy transportation plan and to use corporate resources efficiently. One of the services provided in this context is to dynamically predict the arrival time of the buses at the stops and share this data with the user on platforms such as mobile applications or smart stops.

The system is based on the idea of estimating the time interval for each bus in service to arrive at the next from each stop on its route from the scheduled start time and reporting the cumulative total to passengers in absolute time. Although the estimations made for these times are more accurate for the first stops on the bus route, the margin of error increases for the next stops. Travel time; The time period of the journey depends on many parameters such as the traffic density of the route in this time period, the number of passengers, the driving performance of the driver, and the type of vehicle. When the records of the trip data are examined, it is seen that there are repeated patterns with the effect of these parameters.

In this study, a method based on more detailed clustered parameters is proposed for the estimation of travel times, rather than the usual statistics based on historical records.

In order to develop and test the proposed method, a bus line in Istanbul is taken as a sample data set. Historical data has been evaluated based on data mining. The estimations made by this method are compared with the travel time estimations made by the currently used methods and the results obtained are discussed.

Keywords: Clustering Algorithms, Data Mining, Smart Transportation, Travel Time Prediction, Public Transport

İÇİNDEKİLER

JÜRİ VE ENSTİTÜ ONAYI ii
ETİK İLKE VE KURALLARA UYUM BEYANI iii
TEŞEKKÜR iv
ÖZET v
ABSTRACT vii
İÇİNDEKİLER ix
TABLOLAR LİSTESİ xi
ŞEKİLLER LİSTESİ xii
KISALTMALAR xv
1. GİRİŞ 1
1.2. Duraklara Varış Zamanını Tahmin Etme Problemi 3
1.3. Literatür Araştırması 7
2. TEMEL KAVRAMLAR VE ALGORİTMALAR 13
2.1. Kümeleme Algoritmaları 13
2.1.1. Küme nedir? 13
2.1.2. Kümeleme nedir? 13
2.1.3. Veri madenciliği nedir? 14
2.1.4. Kümeleme algoritmalarının örnek uygulama alanları 14
2.1.5. Kümeleme algoritmasının çalışmalara kazandırdığı faydalar 14
2.1.6. K-means kümeleme algoritması 15
2.1.7. Hiyerarşik kümeleme 18
2.1.8. Aglomeratif kümeleme algoritması 19
2.1.9. DBSCAN (Density Based Spatial Clustering of Applications with Noise) kümeleme algoritması 22
3. PROBLEMİN ÇÖZÜMÜNDE ÖNE ÇIKAN ÇÖZÜM YÖNTEMLERİ 25
3.1. Ortalama Seyahat Süresi (Average Travel Time) 26
3.2. Makine Öğrenmesi Yapay Sinir Ağları (Machine Learning Artificial Neural Network) 27

3.3. Kalman Filtresi (Kalman Filter) 28
4. ÖNERİLEN YÖNTEMDE KULLANILAN YAKLAŞIM 29
4.1. Verilerin Elde Edilmesi 30
4.2. Verilerin Gün Tipine ve Saat Dilimine Göre Kümelenmesi 33
4.2.1. Gün tiplerine göre kümeleme 34
4.2.2. Saat dilimlerine göre kümeleme 35
4.3. Yöntemin Uygulanması 40
4.3.1. Verilerin temizlenmesi 42
4.3.2. Arşiv verilerinin ortalamasını belirlemek 47
4.3.3. Anlık geçiş verilerinin ortalamaya dâhil edilmesi 50
4.3.4. Yolcu ile veriyi kontrollü paylaşmak 52
4.4. Uygulamanın Çalışması ve Ekran Görüntüleri 54
4.5. Deneysel Sonuçlar 62
4.5.1. Doğruluk oranı karşılaştırması 64
5. SONUÇ ve ÖNERİLER 67
KAYNAKLAR 68
ÖZGEÇMİŞ 71

TABLOLAR LİSTESİ

Tablo 1. İETT'nin 2019 Yılında Bir Günde Tamamlanan Sefer Örnekleri 3
Tablo 2. Sefer İptallerine Örnekler 5
Tablo 3. İETT Eyüp Sultan Durağında 2018 Yılında Yapılan Analiz Çalışması 6
Tablo 4. İETT Fabrikalar Durağında 2019 Yılında Yapılan Analiz Çalışması 6
Tablo 5. Sıralı İki Durak Arası Varış Süreleri Sorgu Sonucu 32
Tablo 6. 55 Hattının 2019 Yılı Sefer Tamamlama Süreleri Ve Gün Tipleri 34
Tablo 7. 2019 Yılı Gün Tiplerine Göre Ortalama Sefer Tamamlama Süreleri 35
Tablo 8. 2019 Yılı Sefer Verileri 36
Tablo 9. 2019 Yılında Hafta İçi Yapılan Seferlerin Saat Bazlı Ortalama Sefer Süreleri 37
Tablo 10. 2019 Yılı En Uzun Sefer Süreleri 42
Tablo 11. 2019 Yılı Hatalı Sefer Süreleri – Başlama Süresinden Önce Biten Seferler . 42
Tablo 12. 2019.06.01 İtibari İle Oluşan Hafta İçi Ortalama Arşiv Verisi 50
Tablo 13. Gerçekleşen Süre Tablosu 56
Tablo 14. 1 Aylık Ortalama Tahmin Veri Tablosu 57
Tablo 15. 6 Aylık Ortalama Tahmin Veri Tablosu 58
Tablo 16. Anlık Ortalama 59
Tablo 17. Tahmin Sonucu 60
Tablo 18. Gerçekleşen Yolculuk Süresi İle Tahmin Edilen Yolculuk Süresi Arasındaki Fark Tablosu 61
Tablo 19. Yapılan Tahmin Çalışmasından Bir Bölüm (Tahmin Sütunu: Tahmin Verisi, Tahmin -1 Sütunu: Yolcu İle Paylaşılan Tahmin Verisi) 62
Tablo 20. 16 Günlük 1600 Adet Tahmin Verilerinin Günlük Ortalama Gösterimi 63
Tablo 21. Hata oranları 65
Tablo 22. Hata Oranları 66

ŞEKİLLER LİSTESİ

Şekil 1. Mobiett, İETT Web Sitesi ve İETT Duraklarında Yer Alan Bilgilendirme Ekranları 4
Şekil 2. SVR Algoritmasının Fonksiyonel Akışı 7
Şekil 3. GPS Konum Bilgisi İşleme Akış Diyagramı 9
Şekil 4. Tahmin Modelleme Türleri 10
Şekil 5. Sistem Mimarisi 11
Şekil 6. Sistem Görünümü 12
Şekil 7. Kümeleme Gösterim 13
Şekil 8. K-Means Örnek Çalışma 16
Şekil 9. K-Means Örnek Çalışma Adım 1 ve 2 16
Şekil 10. K-Means Örnek Çalışma Adım 3 17
Şekil 11. K-Means Dezavantajlı Durumlara Örnek Bir Grafik 18
Şekil 12. Hiyerarşik Kümeleme 18
Şekil 13. Aglomeratif Kümeleme Algoritması Adım 1 19
Şekil 14. Aglomeratif Kümeleme Algoritması Adım 2 20
Şekil 15. Aglomeratif Kümeleme Algoritması Adım 3 20
Şekil 16. Aglomeratif Kümeleme Algoritması Adım 4 21
Şekil 17. Kümeler Arası Benzerlik 3 Farklı Yöntemin Gösterimleri 21
Şekil 18. DBSCAN Kümeleme Örnek Gösterimi 23
Şekil 19. Şekil 19: K-Means - DBSCAN Karşılaştırma 24
Şekil 20. İstanbul Trafik Yoğunluk Haritası 25
Şekil 21. Çok Katmanlı Yapay Sinir Ağları 27
Şekil 22. Kalman Filtresi Formül Gösterimi 28
Şekil 23. İki Durak Arasındaki Varış Süreleri 31
Şekil 24. Veri madenciliği araçlarından biri olan Weka'da oluşturulan, sürelere ait nokta yoğunluk grafiği 37
Şekil 25. Python da kullanılacak veri seti dosya içeriği 38
Şekil 26. Hiyerarşik kümeleme için Python da yazılan kod 39

Şekil 27. Python da hazırlanan kümeleme algoritmasının dendrogram çıktısı 40
Şekil 28. Tahmin Algoritmasının İşletilmesi..... 40
Şekil 29. Tahmin Algoritmasının İşletilmesi..... 41
Şekil 30. 120982-118651 Durak Aralığına Ait 2019 Yılı Ocak Ayı Süre Grafiği (Ham Veri)..... 43
Şekil 31. Local Outlier Factor Algoritması İle Anormal Verilerin Tespiti 44
Şekil 32. 120982-118651 Durak Aralığına Ait 2019 Yılı Ocak Ayı Süre Grafiği (Veri Temizliği Yapılmış Hali)..... 46
Şekil 33. OrerTahminleme Uygulamasında Örnek Tahminleme 48
Şekil 34. Orer Tahminleme Programında Günün İlk Seferi İçin Yapılan Tahmin..... 50
Şekil 35. Orertahminleme Uygulamasında Örnek Tahminleme 51
Şekil 36. Anlık Geçiş Verilerinin Ortalamasında Tüm Hatların Kullanılması..... 51
Şekil 37. 5 Duraktan Az Ve 5 Duraktan Fazla Tahminleme İçin Tahmin Sonuçları 53
Şekil 38. Orertahminleme Uygulaması Veri Giriş Alanı 55
Şekil 39. Tarih, Hat, Kalkış Durak, Varış Durak, Saat Dilimi verilerinin girildiği alan, Saat alanı diğer verilere göre otomatik doluyor..... 55
Şekil 40. Tahmin Edilmesi İstenen Aralığın Ne Kadar Sürede Geçildiği Bilgisi Bu Alanda Verilmiştir. Son Kesitin Süresinden Saat Kısmında Seçilen Süre Çıkarıldığında Gerçekleşen Süre Ortaya Çıkar..... 56
Şekil 41. 1 Aylık Tahmin Verisinin Hesaplandığı Alandır, Arşiv Verisi İçerisinde %65 Oranında Ağırlığı Vardır, Genel Tahmin Verisine %23 Gibi Bir Etkisi Vardır..... 57
Şekil 42. 6 Aylık Tahmin Verisinin Hesaplandığı Alandır, Arşiv Verisi İçerisinde %35 Oranında Ağırlığı Vardır, Genel Tahmin Verisine %12 Gibi Bir Etkisi Vardır..... 58
Şekil 43. Saatlik Tahmin Verisinin Hesaplandığı Alandır, Genel Tahmin Verisine %65 Gibi Bir Etkisi Vardır. 59
Şekil 44. Tahmin Verilerinin Hesaplandığı Alandır..... 60
Şekil 45. Yolcuya Bildirilen Tahmin Süresi..... 61
Şekil 46. Tahmin Verisi Yolcuya Bildirilirken, Algoritmaya Göre Tahmin Edilen Kesit 5 Duraktan Büyük İse Yolcu İle 1 Dakika Eksik Paylaşılmaktadır..... 61
Şekil 47. Fark Tablosuna Ait Kodlar..... 61
Şekil 48. 16 Günlük 1600 Adet Tahmin Verilerinin Günlük Ortalamasının Grafiksel Gösterimi 63

Şekil 49. Hata Oranları Grafiksel Gösterimi 65
Şekil 50. Hata Oranları Grafiksel Gösterimi 66

KISALTMALAR

İBB : İstanbul Büyükşehir Belediyesi

İETT : İstanbul Elektrik Tramvay ve Tünel İşletmeleri

AUS : Akıllı Ulaşım Sistemleri

YSA : Yapay Sinir Ağları

SVR : Support Vector Regression

GPS : Global Positioning System

QR : Quick Response

DBSCAN : Density Based Spatial Clustering of Applications with Noise

LOF : Local Outlier Factor

MAE : Mean Absolute Error

MAPE : Mean Absolute Percentage Error

RMSE : Root Mean Squared Error

KNIME : Konstanz Information Miner

SPSS : Statistical Package for the Social Sciences

WEKA : Waikato Environment for Knowledge Analysis

1. GİRİŞ

Dünya nüfusunun yarısından fazlasının kentlerde yaşadığı bilinmektedir. 2050 yılına doğru kentlerde yaşayan nüfus oranının %68'e çıkacağı tahmin edilmektedir (Gözellik, 2018). Bu nüfusun büyük bölümü günlük şehir içi yolculuklar yapmaktadır. İstanbul'da ulaşım için kullanılan Akıllı Kart verilerine göre en çok kullanılan şehir içi ulaşım aracı yaklaşık %50 lik oranla otobüslerdir. Bu hesapla İstanbul'da otobüslerle gerçekleştirilen günlük ortalama yolculuk sayısı yaklaşık 2 milyondur.

İstanbul'da otobüslerle ulaşım hizmetleri İBB bünyesinde İETT tarafından yürütülmektedir. İETT tarafından kurulmuş "Akıllı Ulaşım Sistemleri" (AUS) dinamik yolcu bilgilendirme kapsamında yapılan işlerden birisi de otobüslerin duraklara varış zamanını tahmin etmek ve bu verileri mobil uygulama ya da akıllı durak gibi platformlarda göstermektir.

Tahmin sistemi; bir araç daha sefere başlamadan önce güzergâhı üzerindeki duraklara kaçar dakika ara ile ulaşabileceğini tahmin etme fikrine dayanmaktadır. Güzergâhın ilk durakları için daha doğru bir sonuç üretilmesine rağmen sonraki duraklar için hata payı artmaktadır. Hata payı; trafik yoğunluğu, yolcu sayısı, şoförün sürüş performansı, araç tipi gibi birçok parametreden etkilenmekledir. Ancak asıl hatanın halen kullanılan ve bu parametreleri yeterince ayrıntılı değerlendiremeyen yöntemlerden kaynaklandığı değerlendirilmektedir.

Literatürde otobüsün durağa varış süresinin tahmin edilmesi probleminin çözümü için yapılan çalışmalar üç farklı model kullanılarak ele alınmaktadır.

Bunlardan ilki geleneksel yöntem olan arşiv verisine dayalı modeller, bu modelde arşiv verisinin ortalaması alınarak tahmin verisini belirleme üzerine çalışmalar yapılmıştır (Chen, Yang, An, ve Zhang, 2012). Bu çalışmalarda kullanılan algoritmada mesafe arttıkça hata yüzdesi artmaktadır. Çalışma içerisinde performans değerlendirmeleri (Şekil 6, Şekil 7) de görülmektedir.

İkinci olarak; kullanılan istatistiksel verilere dayalı modellerdir. Bu çalışmalarda zaman serilerinin kalman filtresi ile sınıflandırılması yolu önerilmektedir. Son zamanlarda bu yöntemlerle yapılan birçok çalışma vardır (Zhang, ve diğerleri, 2022).

Bu tarz tahminleme çalışmalarından biri Malezya üniversitesinde yapılmış ve %12 hata oranıyla bir sonraki durak tahminini gerçekleştirmiştir (Md Noor, Yik, Kolandaisamy, ve Ahmedy, 2020).

Tahmin çalışmalarında makine öğrenmesi yöntemleri son zamanlarda popüler hale gelmiştir (Boylu, 2021), (Hua, Wang, Wang, ve Ren, 2017). Makine öğrenmesi yöntemlerinin sakıncalarından ilki; yüksek hesaplama süre ve kapasitesine ihtiyaç duymaları, ikincisi ise beklendiği kadar isabetli çalışmamalarıdır.

Bu çalışmada; öncelikle yapılmış daha önceki çalışmalar incelenmiştir. Otobüslerin duraklara varış sürelerini etkileyen parametreler ele alınmış, bu parametreleri kullanarak arşiv ve anlık verileri birlikte kullanan daha etkin bir yöntem tasarlanmaya çalışılmıştır.

Literatürde yer alan çalışmaların daha çok algoritma veya yöntemin iyileştirilmesine odaklanmış olduğu anlaşıldığından çalışmamızda veri güvenirliliğinin arttırılmasının daha iyi bir başlangıç olacağı düşünülmüştür.

İETT anlık konum başta olmak üzere, aracın güzergâh boyunca tüm kayıtlarını toplayıp merkezle paylaşan bir araç bilgisayarı kullanmaktadır. Burada toplanan verilerin kayıtlarına daha sonra ulaşmak ve bunlar üzerinde çalışmak mümkün olabilmektedir.

GPS ve diğer nedenlerden kaynaklanan yanlış kayıtların bu teknolojinin kullanıldığı tüm toplu taşıma araçları için geçerli olduğu bellidir. Bu nedenle hatalı olma ihtimali yüksek olan verilerin kullanımından kaçınılması gerektiği de açıktır.

Verilerin karakteristik yapısını belirleyebilmek amacıyla öncelikle İETT nin halen kullandığı gerçek bir ulaşım hattı seçilerek bu hatta ilişkin arşiv verileri alınmıştır.

Hat verilerinin haftanın günlerine ve gün içinde saat dilimlerine göre farklı karakteristikler göstereceği hipotezimiz sınanmıştır. Toplanan veriler; bu iki kriter kullanılarak karakteristik olarak ayrışan zaman dilimleri haline getirilebilmiştir. Bu

dilimler içinde kalan verilerde olağandışı olanları ayırarak temizleme işlemi yapılmış, çalışma bu temiz kümeler içinde sürdürülmüştür.

Çalışmada arşiv ya da anlık veriler değil, birçok çalışmanın aksine hem anlık hem de arşiv verilerinden faydalanılarak daha başarılı sonuç elde edilebileceği öngörülmüştür.

1.2. Duraklara Varış Zamanını Tahmin Etme Problemi

İstanbul da otobüslerle, 13708 duraktan bir alt kümeyi kullanan 828 hat oluşturulmuştur. Bu şartlarda şehrin tüm noktalarına ulaşan günlük ortalama 60.000 sefer gerçekleştirilmektedir. Bu seferler yolcu yoğunluğuna göre dağılarak merkezi noktalarda daha sık, kırsal noktalarda daha seyrek seferler olacak şekilde düzenlenmiştir.

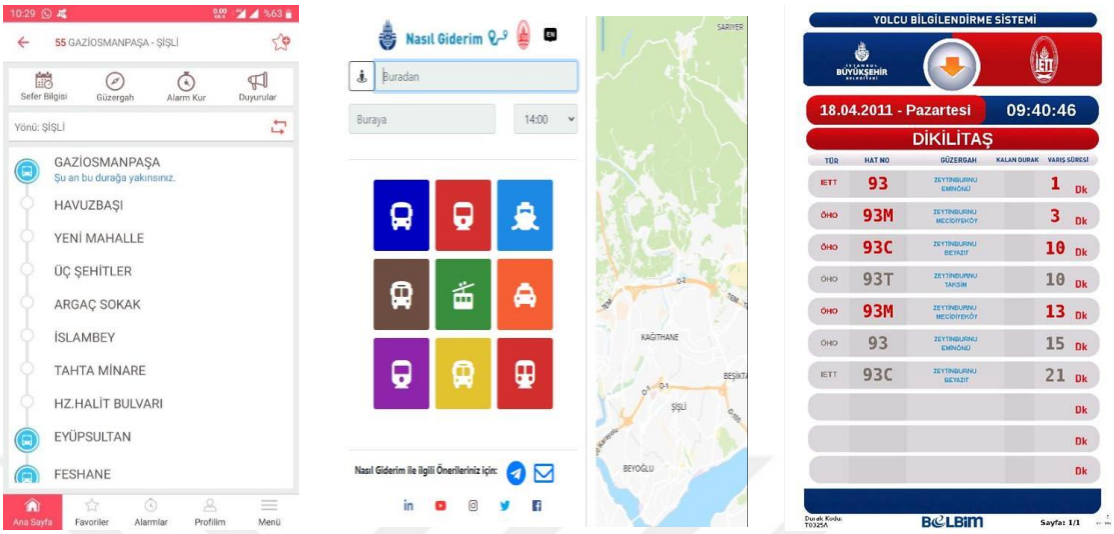
Tablo 1. İETT'nin 2019 Yılında Bir Günde Tamamlanan Sefer Örnekleri

1165	11M	11M_G_D0	317091	O3108	T	20
1165	11M	11M_D_D0	317091	O3108	T	20
1165	11M	11M+SON+217431+G_AND	317091	O3108	T-.	20
-5	11H	11H_G_D2018	H12365	C-1073	T	20
-5	11H	11H_D_D2019	H12365	C-1073	T	20
-9	448	448_D_D0	H10941	A-741	T	20
-9	448	448_G_D0	H10941	A-741	T.	20
-9	448	448_D_D0	H10941	A-741	T	20
-9	43R	43R_G_D3045	H10181	A-041	T	
-9	43R	43R_G_D3045	H10181	A-041	T	20
-9	43R	43R_G_D3045	H10181	A-041	T	20

Save Cancel Script Record Panels Grid Text 59615 row(s) fetched

Bu seferlerin ne zaman başlayacağı hangi güzergâh üzerinden devam edeceği otobüsün ne zaman geleceği gibi bilgiler web sitesinde ve mobil uygulamada yolcularla paylaşılmaktadır. Aynı zamanda otobüs duraklarında ki dijital ekranlar sayesinde duraktan geçen hatlar ve bu hatların durağa varış zamanları yolculara sunulmaktadır.

Mobiett programı beklediğiniz otobüsün durağa varış zamanını ya da bulunduğunuz duraktan hangi hatların geçtiğini ve kaç dakika sonra geleceğini görebileceğiniz bir programdır.



Şekil 1. Mobiett, İETT Web Sitesi ve İETT Duraklarında Yer Alan Bilgilendirme Ekranları

Mobil cihazlar haricinde İETT web sitesi altındaki nasilgiderim.iett.gov.tr adresi üzerinden de durak ve otobüslere dair bilgi alınabilir, rota belirlenebilir, rotayı belirlerken İstanbul içinde kullanılan tüm toplu ulaşım araçlarından (otobüs, minibüs, vapur, tramvay, metro vs.) faydalanılarak rota oluşturulur.

İstanbul'da 960 noktada akıllı durak mevcuttur. Bu duraklarda ki ekranlar sayesinde mobil uygulama ya da farklı bir kaynağa ihtiyaç duymadan, duraktan 40 dakika içerisinde geçecek olan otobüsler listelenmektedir. Durağa varış tahmin sürelerine göre otobüsler listelenmektedirler.

Üç uygulama da yoğun bir şekilde kullanılmakta ve hatalı bildirimle karşılaşıldığında yolcular tarafından geri bildirim kanalları üzerinden geri dönüşler yapılmaktadır.

Bu çalışmaya başlarken öncelikle yolcu şikâyetleri incelenmiştir. Bunlardan anlaşıldığına göre; otobüsün duyurulan zamandan bir dakika önce ile üç dakika sonraya kadar olan süre içinde gelmesi olağan kabul edilmektedir.

Yolcu şikayetleri; otobüsün beklendiği durağa bildirilen zamandan daha geç gelmesinden çok, tesadüfen uygun bir trafik bulan otobüsün bildirilen zamandan önce gelip-geçmiş

olmasından kaynaklanmaktadır. Yine önemli bir şikayet konusu otobüs seferinin iptal edildiği halde yolculara bildirilmemiş olmasıdır.

Tablo 2 de yolcuya önceden duyurulmayan, hatta tahmin verisi yolcu ile paylaşılan, fakat aslında iptal edilmiş olan seferlere örnekler gösterilmiştir. Tablolarda yer alan "I" harfi "Sefer İptal", "OY" harfleri "Otobüs Yokluğu" anlamına gelmektedir. Bu durum sefer saati geldiği halde hareket amirinin hatta tahsis edecek otobüs bulamamasından kaynaklanmaktadır. Bazı dönemlerde bu olay kronikleşmektedir. Bazı hatlarda her gün aynı saatlerde görevlendirilecek otobüs yokluğu sıkıntısı yaşanmaktadır. Böyle durumlar yolcuları mağdur etmekte ve şikâyet konusu olmaktadır.

Tablo 2. Sefer İptallerine Örnekler

9	4 - *36CE*	-511	36CE_G_D5639	B-1885	O03415		06:22	[10 5 2]				I OY	
10	4 - #36CE#	-513	36CE_G_D5639	B-1886	O04909		06:30	06:28				T CA2	
11	- 36CE	-515	36CE_G_D5639	B-1838	860465		06:40		06:40	06:51	08:36	T	
12	4 - !36CE!	-1	36CE_G_D5639	B-047	806085		06:50					T CA2	
13	5 - !36CE!	-3	36CE_G_D6843	B-048	801016		07:00					T CA2	
14	- 36CE	-5	36CE_G_D5639	B-049	801018		07:00		07:01	07:14	09:01	T	
15	1 - !36CE!	-7	36CE_G_D5639	B-050	H11734		07:10					T CA2	
16	1 - 36CE	-9	36CE_G_D5639	B-051	805625	06:58	07:20		07:22	07:34	09:31	T	
17	1 - 36CE	-11	36CE_G_D5639	B-052	803805		07:30		07:31	08:41	09:30	T	
18	4 - 36CE	-501	36CE_G_D5639	B-1853	O05321	07:34	07:40		07:40			T	
19	- 36CE	-503	36CE_G_D5639	B-1852	871042	07:34	07:58		07:58	09:03	10:09	T	
20	1 - 36CE	-505	36CE_G_D5639	B-1881	854407	07:48	08:16		08:17	09:55	10:22	T	
21	1 - 36CE	-507	36CE_G_D5639	B-1808	860379	08:04	08:34		08:35	08:47	10:38	T	
22	- 36CE	-509	36CE_G_D5639	B-1919	855564	08:09	08:52		08:52	09:04	10:58	T	
23	4 - *36CE*	9000	36CE_G_D5639	B-1885	1		09:10	[10 5 2]				I OY	

Aşağıdaki iki tablo, belli bir durakta bir yıl ara ile yapılmış gözlem sonuçlarını vermektedir.

Tablo 3. İETT Eyüp Sultan Durağında 2018 Yılında Yapılan Analiz Çalışması

NO	DURAK ADI	DURAK YÖN	HAT BİLGİ	KAPI N	TARİH	KONTROL SAA	MOBİETT(DK)	GERÇEK ZAMAN	SAPMA
1	EYÜP SULTAN	ALİBEYKÖY	55	B-090	21.5.2018	16:20	10	16:31	1
2	EYÜP SULTAN	ALİBEYKÖY	55-T	B-291	21.5.2018	16:20	11	16:32	1
3	EYÜP SULTAN	ALİBEYKÖY	99-A	B-1747	21.5.2018	16:20	12	16:29	-3
4	EYÜP SULTAN	ALİBEYKÖY	48-A	O1208	21.5.2018	16:20	13	16:33	0
5	EYÜP SULTAN	ALİBEYKÖY	86-V	A-287	21.5.2018	16:20	13	16:35	2
6	EYÜP SULTAN	ALİBEYKÖY	55	B-091	21.5.2018	16:20	17	16:33	-4
7	EYÜP SULTAN	ALİBEYKÖY	55-T	B-292	21.5.2018	16:20	17	16:35	-2
8	EYÜP SULTAN	ALİBEYKÖY	99-A	B-050	21.5.2018	16:20	18	16:35	-3
9	EYÜP SULTAN	ALİBEYKÖY	39-B	B-1920	21.5.2018	16:20	18	16:37	-1
10	EYÜP SULTAN	ALİBEYKÖY	39-Y	B-1799	21.5.2018	16:20	20	16:43	3
11	EYÜP SULTAN	ALİBEYKÖY	86-V	A-288	21.5.2018	16:20	20	16:42	2
12	EYÜP SULTAN	ALİBEYKÖY	55-T	B-293	21.5.2018	16:20	23	16:41	-2
13	EYÜP SULTAN	ALİBEYKÖY	86-V	A-234	21.5.2018	16:20	25	16:50	5
14	EYÜP SULTAN	ALİBEYKÖY	99-A	B-051	21.5.2018	16:20	26	16:44	-2
15	EYÜP SULTAN	ALİBEYKÖY	55	B-162	21.5.2018	16:20	27	16:41	-6
16	EYÜP SULTAN	ALİBEYKÖY	55-T	B-294	21.5.2018	16:20	29	16:46	-3
17	EYÜP SULTAN	ALİBEYKÖY	39	K1102	21.5.2018	16:20	32	17:00	-2
18	EYÜP SULTAN	ALİBEYKÖY	94-Y	K1175	21.5.2018	16:20	32	16:52	0
19	EYÜP SULTAN	ALİBEYKÖY	39-D	M2352	21.5.2018	16:20	32	16:49	-3

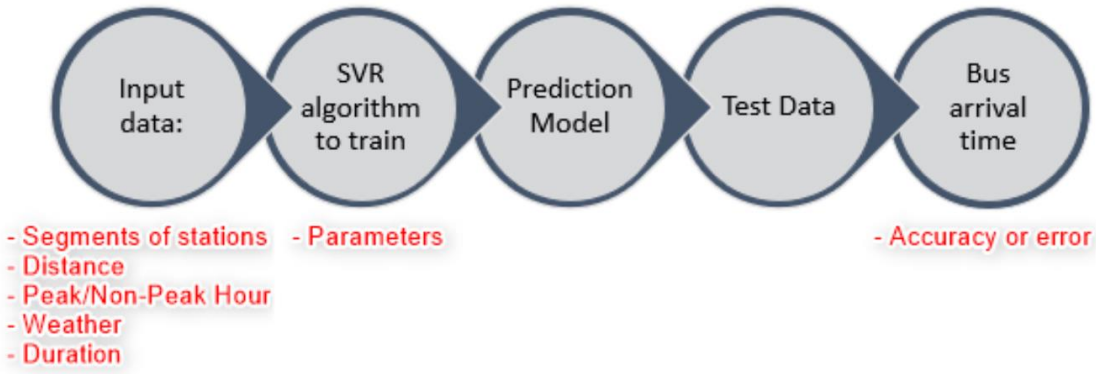
Tablo 4. İETT Fabrikalar Durağında 2019 Yılında Yapılan Analiz Çalışması

A	B	C	D	E	F	G	H	I	J
NO	DURAK ADI	DURAK YÖNÜ	HAT BİLGİSİ	KAPI NO	TARİH	KONTROL SAATİ	MOBİETT(DK)	GERÇEK ZAMAN	SAPMA
1	FABRİKALAR	FSM KÖPRÜSÜ YÖN	122C	C1006	17.5.2019	11:42	3DK	11:48	3+
2	FABRİKALAR	FSM KÖPRÜSÜ YÖN	E3	C1086	17.5.2019	11:42	3DK	11:46	1+
3	FABRİKALAR	FSM KÖPRÜSÜ YÖN	121B	C231	17.5.2019	11:42	7DK	11:50	1+
4	FABRİKALAR	FSM KÖPRÜSÜ YÖN	121B	C230	17.5.2019	11:42	7DK	GELMEDİ	GELMEDİ
5	FABRİKALAR	FSM KÖPRÜSÜ YÖN	522ST	C140	17.5.2019	11:42	10DK	11:47	-5
6	FABRİKALAR	FSM KÖPRÜSÜ YÖN	121B	C232	17.5.2019	11:42	12DK	11:58	4+
7	FABRİKALAR	FSM KÖPRÜSÜ YÖN	122C	C073	17.5.2019	11:42	13DK	11:56	1+
8	FABRİKALAR	FSM KÖPRÜSÜ YÖN	122B	C1842	17.5.2019	11:42	16DK	11:56	-2
9	FABRİKALAR	FSM KÖPRÜSÜ YÖN	121A	C295	17.5.2019	11:42	18DK	19:52	-8
10	FABRİKALAR	FSM KÖPRÜSÜ YÖN	522ST	C141	17.5.2019	11:42	18DK	11:55	-5
11	FABRİKALAR	FSM KÖPRÜSÜ YÖN	522B	C097	17.5.2019	11:42	18DK	12:03	3+
12	FABRİKALAR	FSM KÖPRÜSÜ YÖN	121B	C233	17.5.2019	11:42	19DK	12:09	8+
13	FABRİKALAR	FSM KÖPRÜSÜ YÖN	622	C1870	17.5.2019	11:42	23DK	12:02	-3
14	FABRİKALAR	FSM KÖPRÜSÜ YÖN	E3	C1059	17.5.2019	11:42	23DK	12:05	0
15	FABRİKALAR	FSM KÖPRÜSÜ YÖN	121BS	C302	17.5.2019	11:42	26DK	12:10	-4
16	FABRİKALAR	FSM KÖPRÜSÜ YÖN	522ST	C142	17.5.2019	11:42	28DK	12:06	-4
17	FABRİKALAR	FSM KÖPRÜSÜ YÖN	121B	C234	17.5.2019	11:42	32DK	12:18	4+
18	FABRİKALAR	FSM KÖPRÜSÜ YÖN	522B	C099	17.5.2019	11:43	34DK	12:21	5+

İki nokta arasında ulaşım süresini tahminleme problemi, sistemi etkileyen parametrelerin değişken olmadığı durumlarda kolay bir problemdir, örneğin metro, tramvay gibi bu etkenlerden arındırılmış araçlarda zaman kaymaları çok daha az görülmektedir. Otobüslerde ise trafik yoğunluğu, operatör, araç tipi, kaza, yağış vb. etkenler bu problemi karmaşık hale getirmektedir.

1.3. Literatür Araştırması

Malezyanın Petaling Jaya şehrindeki toplu taşıma sisteminde PJ03 hattı verileri üzerinden tahmin verilerinin iyileştirilmesine yönelik çalışma yapılmıştır (Md Noor, Yik, Kolandaisamy, ve Ahmedy, 2020). Tahmin verisinin doğruluğunu arttırmak için kullanılan algoritma SVR (Support vector regression, Destek vektör regresyonu) algoritmasıdır.



Şekil 2. SVR Algoritmasının Fonksiyonel Akışı (Md Noor, Yik, Kolandaisamy, ve Ahmedy, 2020)

SVR modeli, performans ve genelleme yeteneği ile veriyi eğitmek için küçük miktarda veri kümesine ihtiyaç duymaktadır. Modelin eğitimi için, bir yolun mesafesi, hava durumu ve yoğun veya yoğun olmayan saatlerde çalışma girdi olarak kullanılmıştır. SVR modelinin Çalışmanın sonuçlarına bakıldığında, RMSE(Root Mean Square Error) ortalama karekök sapması'nda elde ettiği 22 saniyelik hata oranı ile daha önceki çalışmalara göre daha iyi sonuç elde edilmiştir. Önerilen modelin daha fazla veri ve özellik ile test edilmesi gerekliliği vurgulanmıştır (Md Noor, Yik, Kolandaisamy, ve Ahmedy, 2020).

ABD nin Wisconsin eyaletindeki trafik verileri kullanılarak yapılan çalışma da Kalman filtresi kullanılarak tahminleme modeli oluşturulmuştur, modelin değerlendirme kriterleri ve indeksleri için Madison'daki 2. rotanın seyahat süresi seçilmiş ve tahmin sonuçlarının hata analizi gerçekleştirilmiştir. Sonuçlar, modelin seyahat süresi tahmininin doğruluk gereksinimlerini karşılayabildiğini ve güzergahın bütününün tahmin verisinin iki durak arası tahminleme verisinden daha iyi sonuçlar ürettiğini göstermektedir. Çalışma tek bir

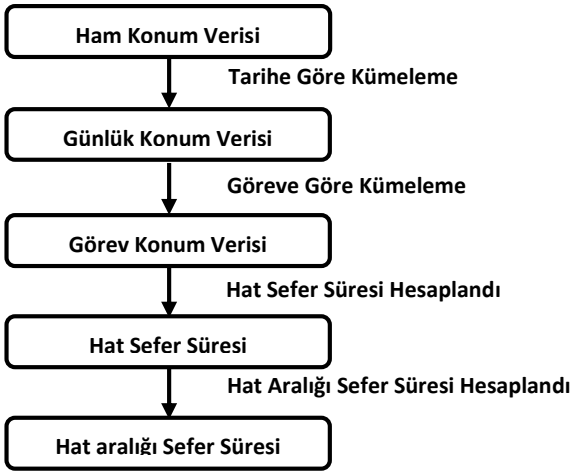
hat üzerine gerçekleştirilmiştir ancak büyük ölçekli toplu taşıma ağında testinin yapılması gerekliliği vurgulanmıştır (Zhang, ve diğerleri, 2022).

Yapılan çalışmada tahminleme problemi üzerine istatiksel modeller ve makine öğrenmesi modelleri ve algoritmalar incelenmiştir. Karşılaştırmalar yaparak en iyi algoritma ya da yöntemi tespit etmeye çalışmışlardır, ancak inceledikleri literatürde çalışmalarda kullanılan veriler standart olmadığı için anlamlı bir karşılaştırma yapılamamıştır. Çalışma sonunda yapılan öneride, kullanılacak yöntemlerin bir standarda oturtulması gerekliliği ve tahmin doğruluğunun ölçülebilmesi için dünya çapında bir standardın olması gerekliliğine (Reich, Budka, ve Hulbert, 2021) vurgu yapılmış ve 3 maddelik öneride bulunulmuştur.

Bunlar;

- Toplu taşıma verilerinin standartlaştırılmış bir çerçeve içerisinde kullanılması gerekmektedir.
- Veriye dayalı uygulamaların kullanılabilmesi için güzergâh modellerinin standartlaştırılması gerekmektedir.
- Şehir içi otobüs ağlarında tahmine dayalı algoritmalar için özel olarak bir kıyaslama yapısı geliştirilmesi gerekmektedir.

Bir başka araştırmada Chen ve arkadaşları yolculuk verileri arşivine dayanan bir algoritma geliştirilmiştir ve iki model belirlenmiştir. Bu modeller link-based (bağlantı tabanlı) model ve section-based (bölüm tabanlı) modeldir. Bu modeller üç farklı yöntemle sınanmıştır, ilk olarak hafta içi ve hafta sonuna göre ikinci olarak haftanın günlerine göre ve son olarak da günün yoğun saatleri ve yoğun olmayan saatlerine göre testler yapılmıştır.

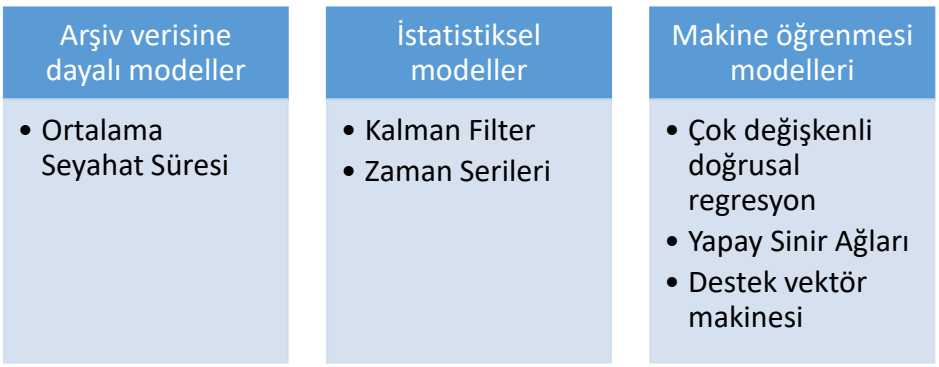


Şekil 3. GPS Konum Bilgisi İşleme Akış Diyagramı (Chen, 2012 Esas Alınarak Çizilmiştir.)

Sonuç olarak bağlantı tabanlı modellerin bölüm tabanlı modellerden daha iyi performans göstermesi mümkün olsa da, bölüm tabanlı modellerin genellikle bağlantı tabanlı modellerden daha güvenilir ve üstün olduğu ifade edilmiştir (Chen, Yang, An, ve Zhang, 2012).

İstanbul daki toplu taşıma sistemi üzerine yapılan çalışmada, Akıllı ulaşım sistemleri ile Türkiye ve dünya örnekleri ile ilgili genel bir bilgilendirme yapılmıştır, kullanılan ekipmanlardan bahsedilmiştir, ülkelerin akıllı ulaşım projelerinden bahsedilmiştir. Tahminlemeyi iyileştirme metodu olarak Üstel Düzleştirme (Exponential Smoothing) yöntemi kullanılarak veri temizleme işlemi yapılmıştır. Sonrasında veriler karşılaştırılarak sonuçlar değerlendirilmiştir. Basit ortalama yöntemi ile elde edilen sonuçlarla, çalışmada elde edilen sonuçlar karşılaştırılmış önerilen yöntemin çok daha başarılı sonuçlar verdiği gözlemlenmiştir (Tiryaki, 2015).

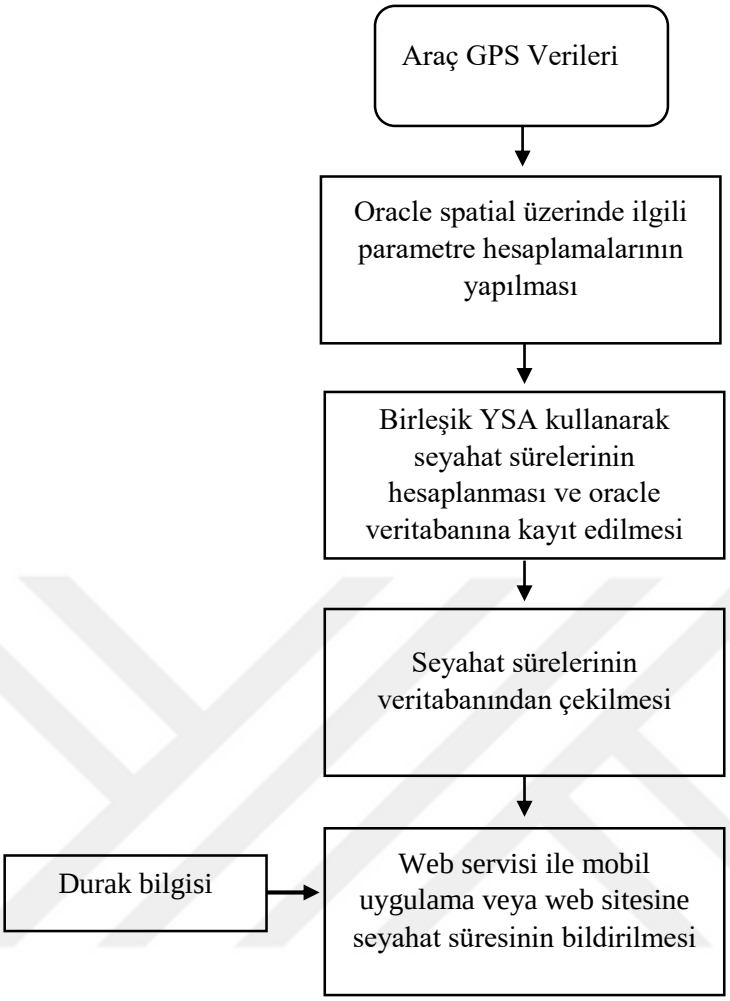
Yapılan çalışma da çok yoğun bir literatür araştırması yapılmış ve öncelikle tahminleme modelleri irdelenmiştir. Arşiv verisine dayalı modeller, istatiksel modeller ve makine öğrenmesi modelleri olarak 3 ana tahmin modeli incelenmiştir.



Şekil 4. Tahmin Modelleme Türleri (Boylu, 2021 Esas Alınarak Çizilmiştir)

İstanbul un farklı bölgelerinden 6 hat (11US, 17, 14, 252, 16C, 18K) ile çalışılmış ve bu hatlarda çalışan araçların GPS cihazlarından elde edilen veriyi kullanarak, makine öğrenmesi tekniği ile çok değişkenli doğrusal regresyon algoritmasını kullanarak olumlu sonuçlar elde etmiştir (Boylu, 2021).

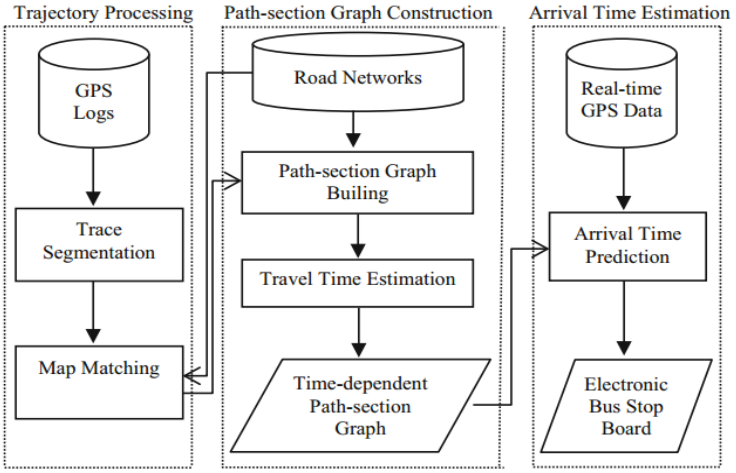
Kayseri büyük şehir belediyesindeki veriler kullanılarak, toplu taşıma araçlarının duraklara varış zamanını doğru bir şekilde tahmin edebilmek için birleşik yapay sinir ağları (YSA) algoritması tabanlı bir sistem üzerine çalışılmıştır.



Şekil 5. Sistem Mimarisi (Taşyürek,2020 Esas Alınarak Çizilmiştir)

Akıllı otobüs durak verileri ve 800 adet farklı otobüs hattının GPS verisi ve akıllı duraklarda bulunan kare kodlar (QR) okutularak duraktan hangi otobüsün ne zaman geçeceği birleşik Yapay sinir ağları modeli ile tahmin edilebilme üzerine bir çalışma yapılmıştır. Geliştirilen yapay sinir ağları tahmin algoritmasının sonuçları lineer regresyon yöntem ile elde edilen sonuçları ile karşılaştırılmış ve daha başarılı sonuçlar verdiği gözlemlenmiştir (Taşyürek ve Çelik, 2020).

2017 yılında Haitao Xu tarafından, 2 ay boyunca 1000 otobüsten elde edilen GPS verileri kullanılarak yapılan çalışmada, elde edilen veriler hafta içi ve hafta sonu olmak üzere iki kümeye ayrılmıştır. Herhangi iki durak arasındaki katedilen mesafe / süre orantısının değişken dağılımlarını öğrenmek için bir kümeleme algoritması sunulmuştur. Aylık periyotta elde edilen veri setini kullanarak otobüs varış zamanı tahmin sistemi oluşturulmuştur. Yapılan kapsamlı deneyler, yöntemin diğer yöntemlere göre daha az anlık hesaplama ile daha doğru varış zamanını tahmin edebileceğini göstermektedir (Xu ve Ying, 2017).



Şekil 6. Sistem Görünümü, (Xu ve Ying, 2017)

Başkaya ve arkadaşları tarafından 2020 yılında yapılan çalışmada, ulaşıma etki eden faktörlerin ve parametrelerin dinamik bir yapıda toplanarak yapay zekâ teknikleri analizleri neticesinde otomatik planların hazırlanması ve etkin toplu ulaşım faaliyetlerinin yürütülmesi ile çevre kirliliğine, trafik yoğunluğuna, yüksek maliyet problemlerine çözüm getirilmesi, bunun sonucunda, belirli periyotlar da hazırlanan ulaşım planlaması iş yükü ve maliyetinin azaltılması hedeflenmiştir. Gerçekleşen ulaşım hareketlerinin kayıt altında tutulması ile ileride oluşacak olan ulaşım hareketlerinin analizi yapılabilecek ve gelecekte gerçekleşmesi beklenen ulaşım hareketlerinin düşük bir yanılma payı ile yaklaşık olarak tahmin edilerek ulaşım planlanmasının somut bir bilgiye bağlı kalarak yapılması amaçlanmıştır. Akıllı şehirler kapsamında yapay zekâ teknikleri kullanarak etkin ulaşım planlarının oluşturulması üzerine bir model önerisi çalışılmıştır. Haftalık olarak gerçekleştirilen sefer sayılarının %52 oranında optimize edildiği başarılı bir çalışma yapılmıştır (Başkaya, Ağaçsapan, ve Çabuk, 2020).

2. TEMEL KAVRAMLAR VE ALGORİTMALAR

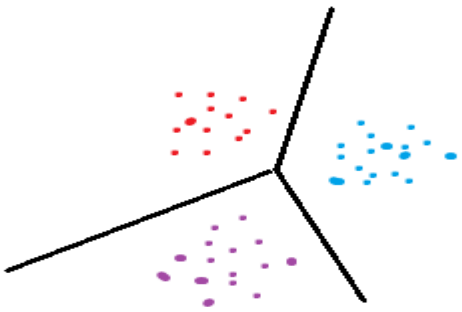
2.1. Kümeleme Algoritmaları

2.1.1. Küme nedir?

Tanımı yapılmış birbirine yakın ya da benzer nesneler topluluğuna küme denir. Bir küme gösterimlediği elemanlarının küme içi uzaklıklarını minimum, kümeler arası uzaklıkları maksimum seviyede tutacak biçimde düzenlenir. İçinde barındırdığı nesneleri en iyi şekilde temsil eder ve o yönde sınırları belirlenir. Kümelerin barındırdığı veriler nesne olarak ifade edilir. Her bir nesne iki boyutlu gösterimde noktalarla ifade edilir.

2.1.2. Kümeleme nedir?

Kümeleme, birbirine benzer ya da ortak özellikte üyeleri olan grupları, küme küme olacak şekilde düzenleme süreci olarak tanımlanabilir. Kümeleme de önemli olan nokta en iyi kümelemenin oluşması için nasıl karar verileceğidir. Ölçütü belirlemesi gereken kullanıcıdır. Kümeleme işlemi bittikten sonra çıkan sonucun kullanıcının gereksinimlerini karşılaması gerekir.



Şekil 7. Kümeleme Gösterim

2.1.3. Veri madenciliği nedir?

Veri madenciliği, gözlem ya da farklı yöntemlerle elde edilmiş büyük çaplı verilerden oluşan kümeleri inceleyerek, kümeler arası fayda sağlayacak benzerlikler ya da ilişkiler bulmak, verileri anlaşılır ve kullanılabilir halde özetlemektir. Veri madenciliği çalışması ile elde edilmiş bu ilişki ve özetler çoğunlukla model ve örüntüler olarak tanımlanır. Kümeler, çizge ve ağaç yapıları, zaman serilerinde tekrarlı örüntüler örnek olarak gösterilebilir (Hand, Mannila, ve Padhraic, 2001). Veri madenciliği ile ilgili ön plana çıkan araçlar kontrol edildiğinde, bir çok aracın var olduğu görülecektir. Bunlar arasında popüler olan birkaç aracı belirtmemiz gerekirse, R, Python, Kaggle, Knime, SPSS ve WEKA programları örnek olarak verilebilir.

2.1.4. Kümeleme algoritmalarının örnek uygulama alanları

Kümeleme algoritmalarının kullanıldığı birçok alan vardır. Örnek vermek gerekirse;

- Toplu ulaşım kartı: Kart basış bilgileri üzerinden yoğunluk belirleme ile ilgili çalışmalar yapılmaktadır.
- Marketing: Geçmişte yaptığı satın almalardan ve öne çıkan özelliklerinden benzer müşterileri gruplayarak pazarlama teknikleri geliştirilmek üzere kullanılmaktadır.
- Zooloji, Botanik: Hayvanlar ve bitkilerin sınıflandırılması için kullanılmaktadır.
- Metin Madenciliği: Kullanıcının talebine göre, metinde geçen her bir harfi ya da anlam ifade eden her bir kelimeyi kaynak kabul ederek çalışmalar yapılmaktadır.
- Jeoloji: Yağışlar ile ilgili çalışmalar ya da deprem çalışmalarında kullanılmaktadır.
- Kütüphaneler, Kitap satışı yapılan siteler: En çok tercih edilen türler kitap siparişlerinin özellikleri vs. çalışmalarda kullanılmaktadır.
- Paralı Kanallar: Müşterinin takip ettiği film, dizi ve program türlerine göre, müşteriye önerilerde bulunma, en çok tercih edilen türlerde içerik üretme gibi alanlarda kullanılmaktadır.

2.1.5. Kümeleme algoritmasının çalışmalara kazandırdığı faydalar

- Ölçeklenebilirlik
- Özellik farklılıklarını yakalayabilmesi
- Gürültü dediğimiz küme dışı değerleri belirleyebilmesi

- Ortaya çıkan sonuçların yorumlanabilir olması
- Verinin büyüklüğünün, girdi sayısının çokluğunun sorun teşkil etmemesi

Aşağıda sıklıkla kullanılan bazı algoritmalardan kısaca bahsedilmektedir.

2.1.6. K-means kümeleme algoritması

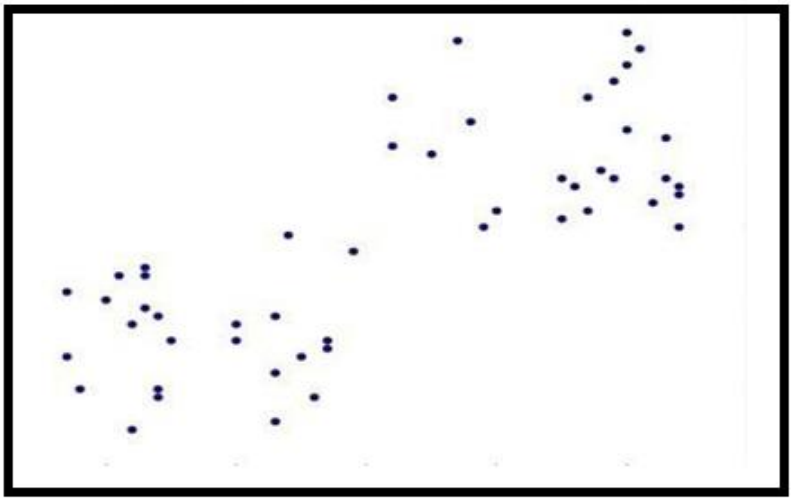
K-Means bölümlemeli kümeleme yaklaşımıdır. Kümeleme algoritmalarından biri olan k-means algoritmasının mantığı n tane datadan oluşan data setini, belirlenen k değeri kadar kümeye bölümlemektir. Amaç, gerçekleşen kümeleme işleminde elde edilen kümelerin küme içi nesnelerin benzerliklerinin maksimum olması, kümelerin birbiri ile benzerliklerinin ise minimum olmasıdır. Küme de benzerlik, kümenin ağırlık merkezi olan nesne (centroid) ile küme içerisindeki diğer nesnelerin aralarındaki uzaklıklarının ortalama değeri ile belirlenmektedir.

Algoritma adımları:

- Öncelikle kümelenecek nesnelerin kaça bölüneceğini ifade eden merkez noktalar (centroid) belirlenir. Merkez noktalar genellikle rastgele noktalar olarak belirlenir. Örnek olarak nesneler 2 kümeye bölünecekse iki adet ağırlık merkezi var demektir. K=2 şeklinde ifade edilir.
- Küme sayısı belirlendikten sonra her bir nesne 2 adet merkez nokta ile olan yakınlıklarına göre kümelere yerleştirilir.
- Bütün nesneler yerleştirildikten sonra merkez noktaların yerleri yeniden hesaplanır.

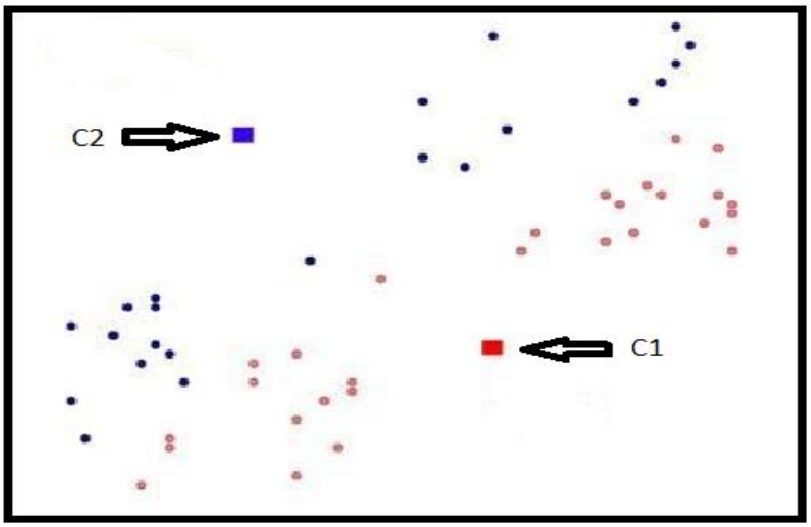
  Bu hesaplama adımları merkez noktaların yeri değişmeyecek duruma gelene kadar devam eder. Böylelikle kümeleme süreci tamamlanmış olur.

  Örneğin aşağıdaki veri setinin k=2 şeklinde iki ayrı gruba ayırmaya çalışıldığı varsayılsın.



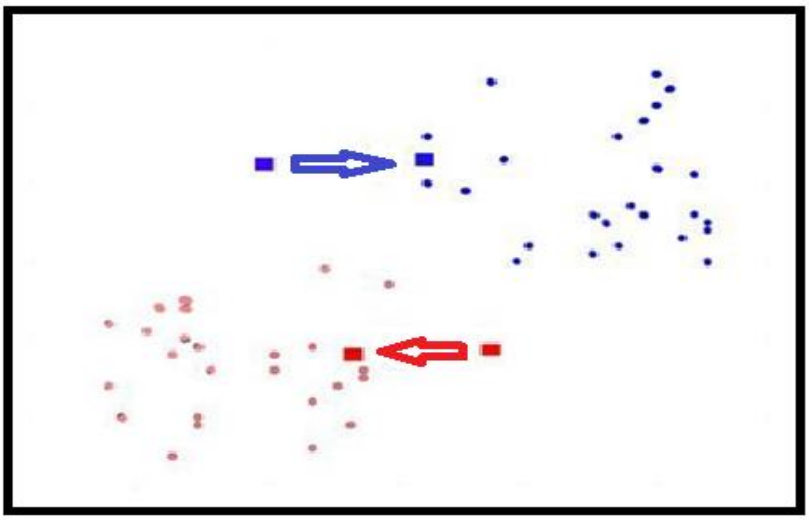
Şekil 8. K-Means Örnek Çalışma

- İki farklı gruba ayrılmak istendiği için c1 ve c2 adında rastgele iki sınıf merkezi belirlenir.
- Her bir noktanın belirlenen merkez noktalara uzunluğu hesaplanarak c1 nolu merkeze yakınsa 1, c2 nolu merkeze yakınsa 2 şeklinde etiketlenir.



Şekil 9. K-Means Örnek Çalışma Adım 1 ve 2

- Daha sonra mavi noktaların ağırlık merkezi bulunur ve bu nokta yeni merkez olur. Kırmızı noktaların ağırlık merkezi bulunur ve bu nokta yeni merkez kabul edilir. Bu işlemler yeniden yapılır ve üyeler bu yeni merkezlere göre yeniden kümelere ayrılır. Bu merkez güncelleme işlemi alttaki gibi gösterilebilir.



Şekil 10. K-Means Örnek Çalışma Adım 3

Kümelere ayırma işlemi, merkez noktanın yeri değişmeyinceye kadar yani kümelerin elemanları tam olarak belirleninceye kadar devam eder. Tüm merkez noktalar belirlenip kümelerin sınırları belirlendikten sonra işlem bitirilir.

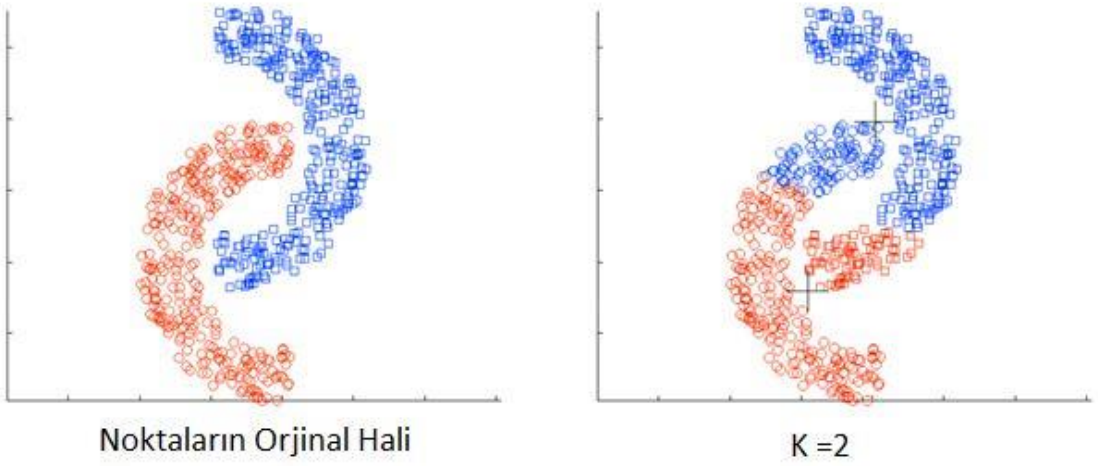
K-means algoritmasının avantajları

- Gerçekleştirilmesi kolayca yapılır
- Karmaşıklığı diğer kümeleme algoritmalarına göre daha azdır

K-Means algoritmasının dezavantajları

- Veriler gruplar halinde ve boyutları farklı ise
- Verilerin yoğunlukları farklı ise
- Veri gruplarının şekli küresel değilse
- Veri içinde aykırılıklar var ise

Yukarıda dezavatajlar kısmında sayılan özelliklere sahip veri grupları K-Means algoritması çalıştırılarak kümeleme yapılmak istendiğinde, başarılı kümeleme yapamaz ve elde edilen sonuç başarısız olur Şekil 11'de örneği görünmektedir.

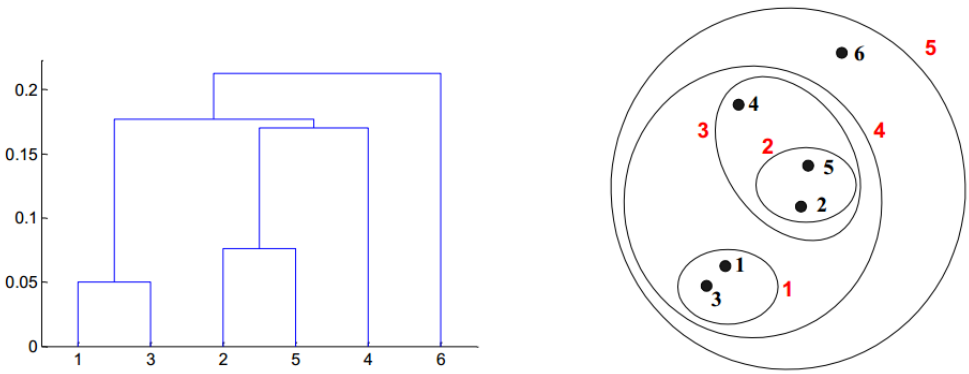


Şekil 11. K-Means Dezavantajlı Durumlara Örnek Bir Grafik (K-Means Algorithm for Cluster Analysis, 2018)

Şekil 11 incelendiğinde, şekli küresel olmayan iki farklı veri grubu görüyoruz, K-means ile kümeleme yaptığımızda sonuç istenildiği gibi çıkmamaktadır. K-means in bu tarz kümeleme çalışmalarında başarısız olduğu bilinmektedir.

2.1.7. Hiyerarşik kümeleme

Hiyerarşik ağaç şeklinde organize edilmiş iç içe kümeler üretir. Ağaç modeli olarak görselleştirilebilir. Aglomeratif ve Bölücü kümeleme olarak iki çeşidi vardır. Bu çalışmada aglomeratif kümeleme anlatılmıştır.



Şekil 12. Hiyerarşik Kümeleme (Bilgin, 2019)

2.1.8. Aglomeratif kümeleme algoritması

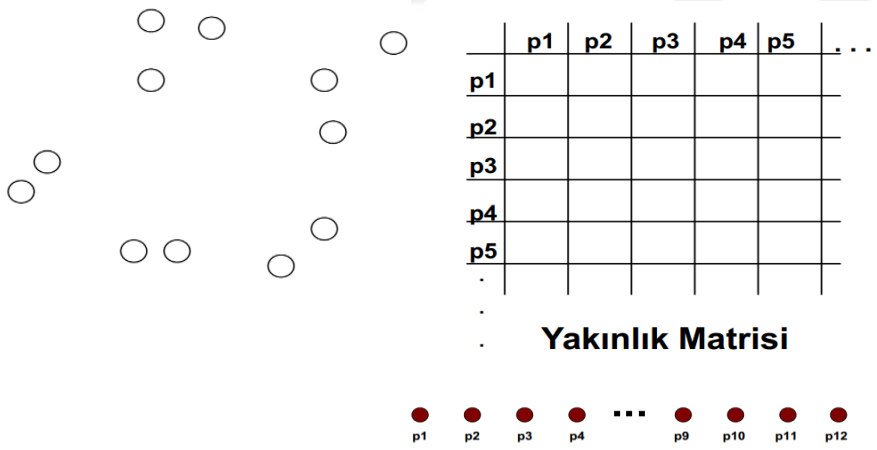
En çok tercih edilen hiyerarşik kümeleme tekniğidir. Algoritmanın adımları;

- Yakınlık matrisi hesaplanır
- Her bir nokta bir küme olarak alınır
- Adımlar tekrar edilir
- Birbirine en yakın iki küme birleştirilir
- Yakınlık matrisi güncellenir
- Bir küme kalana kadar işlem devam ettirilir

İki kümeye ait yakınlık hesaplanması önemlidir. İki küme arasındaki mesafenin tanımlanmasında kullanılan birbirinden farklı yaklaşımlar algoritmaları ayırır.

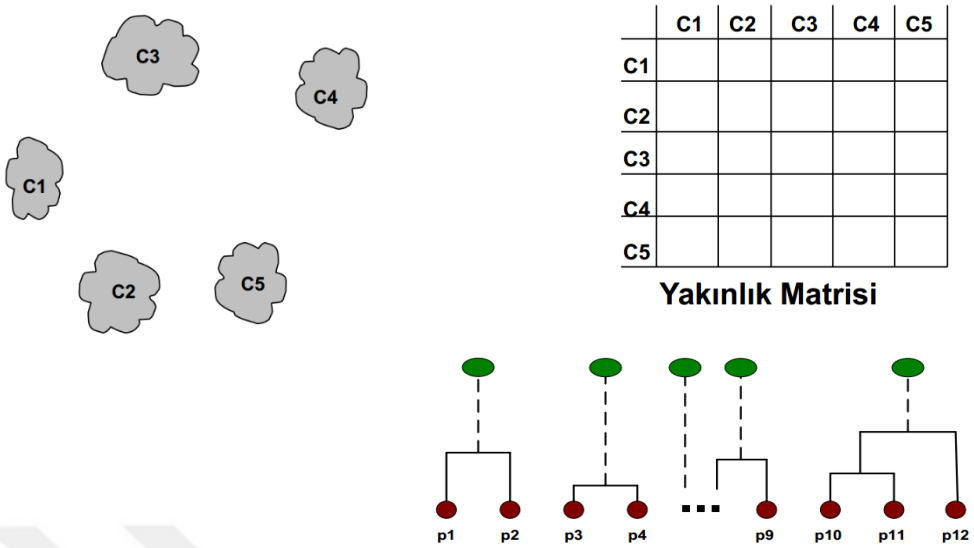
İşleyiş bir örnekle açıklanacak olursa;

- Başlangıç Durumu: ilk olarak her bir noktayı küme olarak al ve buna göre yakınlık matrisini oluştur.



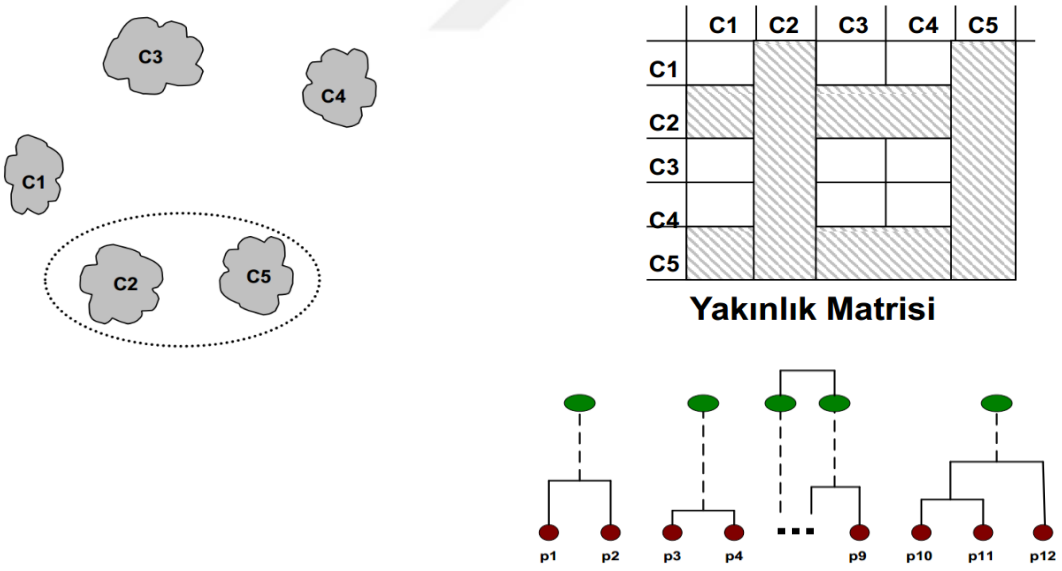
Şekil 13. Aglomeratif Kümeleme Algoritması Adım 1 (Bilgin, 2019)

- Ara Durum: Birbirine yakın birkaç nokta için birleştirme adımından sonra, bazı yeni kümeler ortaya çıkar



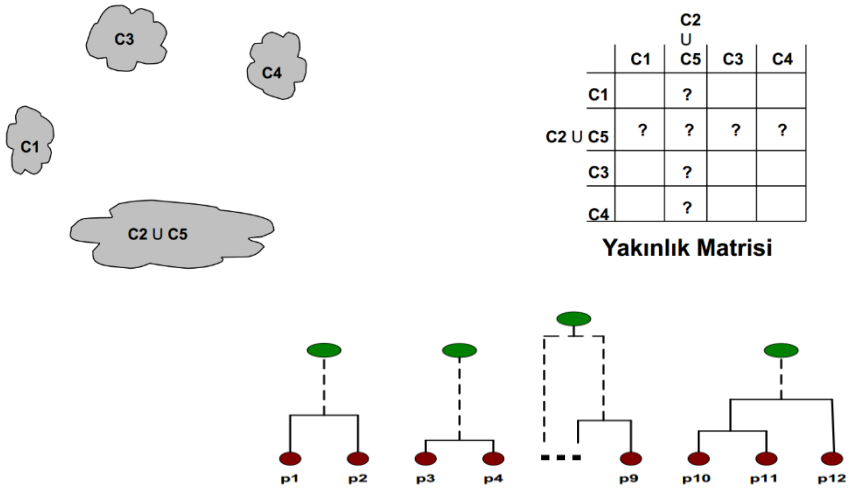
Şekil 14. Aglomeratif Kümeleme Algoritması Adım 2 (Bilgin, 2019)

- Ara Durum: İki en yakın küme olan (C2 ve C5) kümelerini birleştirerek yakınlık matrisinin güncellenmesini sağlıyoruz.



Şekil 15. Aglomeratif Kümeleme Algoritması Adım 3 (Bilgin, 2019)

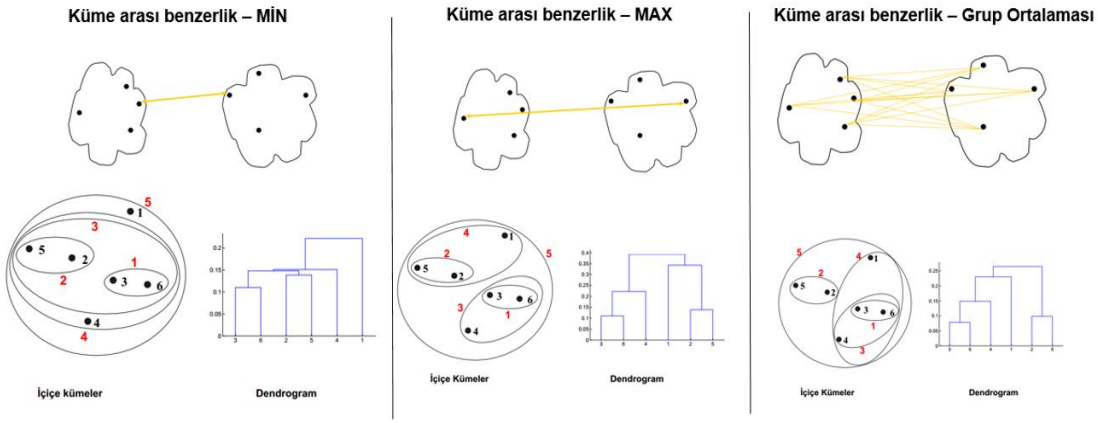
Birleştirmeden sonra: Yakınlık matrisi şu şekilde güncellenecektir.



Şekil 16. Aglomeratif Kümeleme Algoritması Adım 4 *(Bilgin, 2019)*

Bu aşamada kümeler arası mesafe tanımlanır: Kümeler arası mesafe 3 farklı yöntemle yapılabilir

- Kümeler arası benzerlik - MİN
- Kümeler arası benzerlik - MAX
- Kümeler arası benzerlik - Grup Ortalama



Şekil 17. Kümeler Arası Benzerlik 3 Farklı Yöntemin Gösterimleri (Bilgin, *2019)*

Hiyerarşik Kümelemenin Güçlü Yanları

- Küme sayısının belirlenmesine gerek yoktur
- Uygun bir seviyeden dendrogramı keserek istenilen sayıda küme bulunabilir

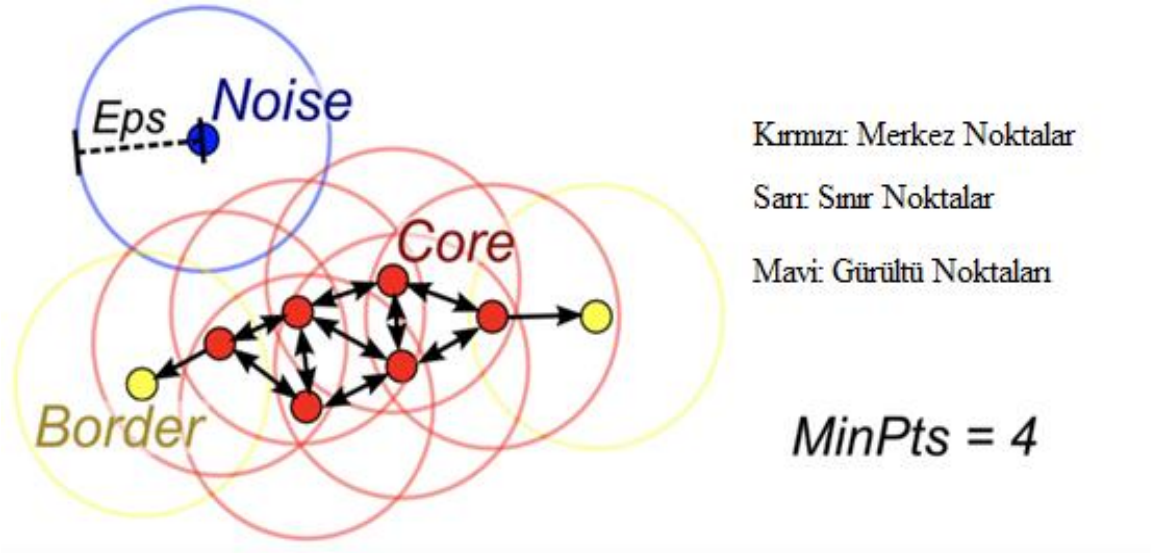
Hiyerarşık Kümelemenin Zayıf Yanları

- İki kümenin birleştirilmesine karar verildikten sonra geri al yapılamaz.
- Gürültü ve aykırılıklara duyarlıdır.
- Bölünmemesi gerektiği halde büyük kümeleri bölebilir.

2.1.9. DBSCAN (Density Based Spatial Clustering of Applications with Noise) kümeleme algoritması

Yüksek yoğunluklu kümeleri düşük yoğunluklu kümelerden ayırmak için makine öğreniminde kullanılan bir kümeleme yöntemidir. DBSCAN'ın yoğunluğa dayalı bir kümeleme algoritması olduğu göz önüne alındığında, verilerde gözlemlerle çok yoğun olmayan alanlara karşı yüksek gözlem yoğunluğuna sahip alanları aramak için başarılı sonuç verir. DBSCAN, verileri farklı şekillerde kümeler halinde de sıralayabilir, bu da bir başka güçlü avantajdır. DBSCAN şu şekilde çalışır:

- Veri kümesini böler n boyutlar
- DBSCAN, veri kümesindeki her nokta için o veri noktasının etrafında n boyutlu bir şekil oluşturur ve ardından bu şekle kaç tane veri noktasının düştüğünü sayar.
- DBSCAN bu şekli bir küme olarak sayar . DBSCAN, küme içindeki her bir noktadan geçerek ve yakındaki diğer veri noktalarının sayısını sayarak kümeyi yinelemeli olarak genişletir.



Şekil 18. DBSCAN Kümeleme Örnek Gösterimi (Lutins, 2017)

Yukarıda bahsedilen süreci adım adım gerçekleştiren DBSCAN, verileri n boyuta bölerek başlayacaktır. DBSCAN bunu yaptıktan sonra rastgele bir noktadan başlayacak (bu durumda kırmızı noktalardan biri olduğunu varsayalım) ve yakınlarda kaç tane daha nokta olduğunu sayacaktır. DBSCAN, yakınlarda başka veri noktası kalmayıncaya kadar bu işleme devam edecek ve ardından ikinci bir küme oluşturmaya çalışacaktır.

Grafikten de açıkça görüldüğü gibi, DBSCAN işini yapmadan önce verilmesi gereken iki parametre şunlardır:

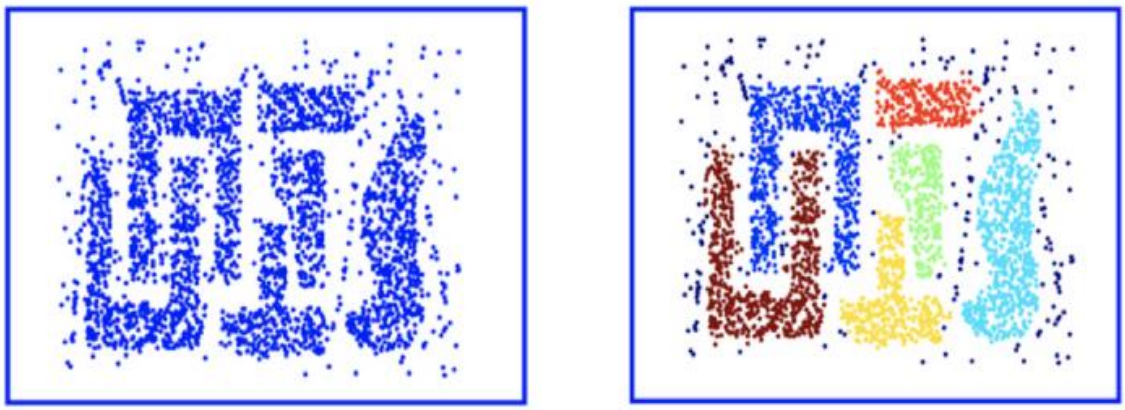
Tek bir kümeyi belirlemek için gereken minimum veri noktası sayısı nedir?

Aynı küme içinde bir nokta bir sonraki noktadan ne kadar uzakta olabilir?

Grafiğe geri dönülürse epsilon, veri noktaları arasındaki mesafeyi test etmek için verilen yarıçaptır. Bir nokta, başka bir noktanın epsilon mesafesi içindeyse, bu iki nokta aynı kümede olacaktır.

Ayrıca, bu senaryoda gereken minimum puan sayısı 4'e ayarlanmıştır. Her bir veri noktasından geçerken, DBSCAN birbirinden epsilon mesafesinde 4 nokta bulduğu sürece bir küme oluşur.

Önemli: Bir noktanın "çekirdek" nokta olarak kabul edilebilmesi için, epsilon mesafesi içinde minimum sayıda noktayı içermesi gerekir. Noktanın kendisi sayıma dahildir. DBSCAN algoritmasında mutlaka her veri noktası kategorize edilmiştir. Aşağıdaki grafik incelenirse;



Şekil 19. Şekil 19: K-Means - DBSCAN Karşılaştırma (Lutins, 2017)

Soldaki görüntü, çok boyutluluğu hesaba katmayan K-Means gibi daha geleneksel bir kümeleme yöntemini göstermektedir. Doğru görüntü, benzer kümeleri bulmak için DBSCAN'ın verileri nasıl farklı şekil ve boyutlara dönüştürebileceğini gösterir. Ayrıca, sağdaki resimde, veri kümesinin dış kenarı boyunca yer alan noktaların bir sınıfa dahil edilmedikleri görülmektedir. Bu da bazı verilerin aykırılık taşıdığını düşündürmektedir.

DBSCAN'ın avantajları:

- Belirli bir veri kümesinde yüksek yoğunluklu kümeleri düşük yoğunluklu kümelere karşı ayırmada olağanüstü başarılıdır
- Veri kümesi içindeki aykırı değerlerin işlenmesinde olağanüstü başarılıdır

DBSCAN'ın dezavantajları:

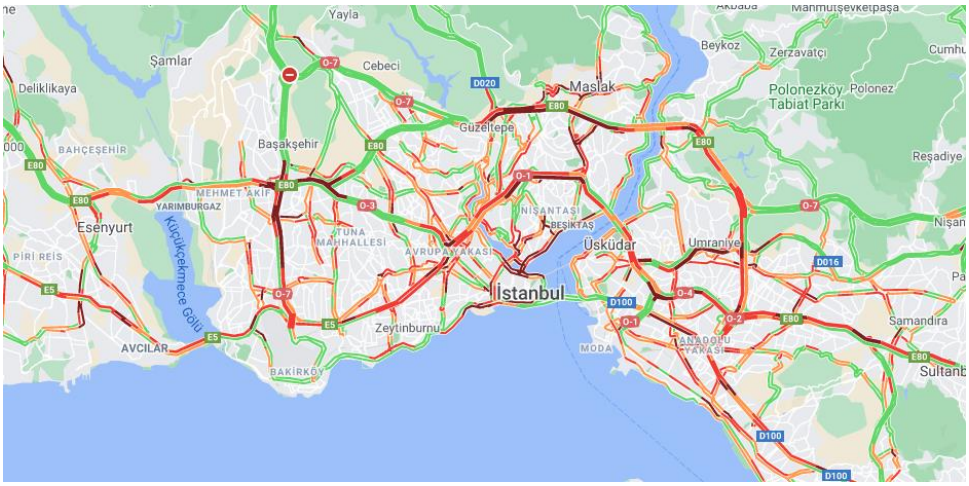
- DBSCAN, gürültü ve aykırılıklara duyarlılığı olduğu için belirgin yoğunluk farkı olan kümeleri ayırmada olağanüstü başarılı olsa da, birbirine benzer yoğunluklu kümeleri ayırırırken kötü bir performans sergiler.

3. PROBLEMİN ÇÖZÜMÜNDE ÖNE ÇIKAN ÇÖZÜM YÖNTEMLERİ

Duraklara varış zamanının tahmini çok uzun zamandan beridir çözülmeye çalışılan ve en iyi sonucun alınmaya çalışıldığı bir problemdir. Günümüzde tahmin verisi üretirken, farklı yaklaşımlar üzerinde durulmaktadır.

İki nokta arasındaki mesafeyi tahminleme problemini karmaşık hale getiren durumlara adım adım bakıldığında;

- Trafik Yoğunluğu: Gün içinde aynı güzergâh için yapılan tahminlerde hatanın değiştiği gözlenmektedir. Veriler ayrıntılı olarak analiz edildiğinde yolcuların günlük yer değiştirme eğilimlerine göre oluşan trafik yoğunluğunun buna etken olduğu anlaşılmaktadır. Nitekim yolculuk süresini uzatan tıkanmaların günün hep aynı saatlerinde ve aynı noktalarda olduğu görülmektedir. Yolcuya yapılan bilgilendirmelerde trafik yoğunluğunun pik seviye de olduğu zamanlarda hata oranları daha fazla olmaktadır. Burada üzerinde durulan yoğunluk, trafik kazası yada anlık bir olaydan kaynaklı trafik yoğunluğu değil, her gün tekrar eden ve artık rutinleşmiş yoğun trafiktir.



Şekil 20. İstanbul Trafik Yoğunluk Haritası

- Araç tipi: Otobüs işletmecileri envanterlerin de solo ve körüklü olmak üzere iki tip araç bulundurmaktadır. Solo araçlar trafikte körüklü araçlara göre çok daha rahat hareket etmektedir. Körüklü araçların tercih nedeni ise solo araçlara göre daha fazla yolcu kapasitesine sahip olmasıdır. Ancak körüklü araçlar şehir içinde manevra yapmaları gerekirken çok zorlanmakta ve zaman kaybına uğramaktadır. Bu farklılık tahminlemede sapmalara neden olmaktadır.
- Operatör: Otobüs işletmeciliği birden fazla şirket tarafından yürütülebilmektedir, örneğin İstanbul'da üç farklı şirket tarafından otobüs işletmeciliği yapılmaktadır. Bu süreçte ticari kaygılar nedeniyle duraklara normalden geç varma ya da erkenden geçme gibi durumlar yoğunlukla yaşanmaktadır. Bu gibi durumlar da tahminleme sürecini zorlaştırmaktadır.
- Şoför sürüş performansı: Şoförlerin otobüsü sakin yada agresif kullanımı da tahmin süreçlerini etkilemektedir. Bu nokta da otobüs işletmeleri şoförlerin sürüş eğitimi ile ilgili olarak, simülasyon merkezleri kurmakta ve ulaşım da çalışacak şoförlere güvenli ve ekonomik sürüş teknikleri eğitimi vermekte ve sürücü sertifikası alacak kişilerden bu belge talep edilmektedir.
- Kaza, yağış vb. planlanmamış olaylar: Plan dahilinde olmayan ve trafiği etkileyen her olay tahminleme performansını düşürmektedir. İstanbul gibi büyük metropollerde ana güzergahtaki bir kaza yada 5 dakika gibi kısa bir süre yağan yağmur trafiği ciddi manada etkilemektedir.

Aşağıda arşiv verilerinin ortalamasını kullanarak tahminleme, makine öğrenmesi yöntemlerini kullanarak yapay sinir ağları yöntemi ile tahminleme, Kalman Filtresi vb. algoritmalarla tahminleme gibi başlıca yöntemlere değinilmiştir.

3.1. Ortalama Seyahat Süresi (Average Travel Time)

Belirlenen iki nokta arasındaki belli bir zaman aralığına ait sefer sürelerinin ortalaması alınarak tahmin verisi üretmektedir. Buradaki verilerde değerlendirilirken de kaza, yol kapanması, okul vb. anlık durumlar dikkate alınmadığı halde verinin eğitilmesi çok uzun zaman almaktadır. Ortalamayı hesaplamak için arşiv verisi alınırken, hangi hat için çalışma yapılıyorsa, o hat ile ilgili veriler kullanılmaktadır.

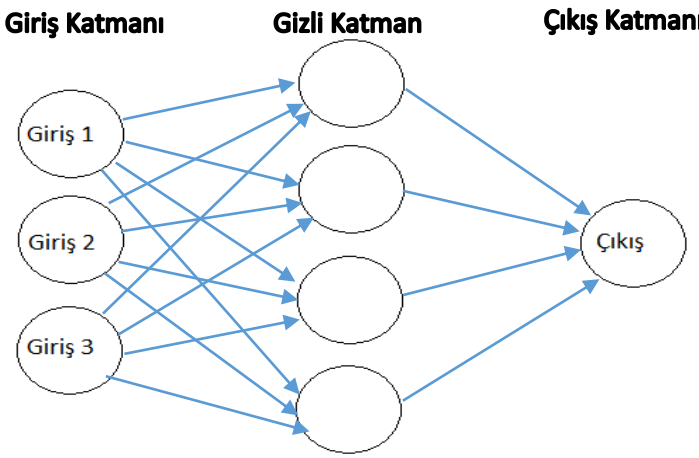
3.2. Makine Öğrenmesi Yapay Sinir Ağları (Machine Learning Artificial Neural Network)

Son yıllarda tahminleme ile ilgili birçok çalışma yapay sinir ağları kullanılarak yapılmıştır. Ülkemizde de bu yönde akademik çalışmalar yapılmaktadır. Yapılan çalışmalar incelendiğinde uygulamada en büyük dezavantajlarının çok güçlü donanım birimlerine ihtiyaç olduğu anlaşılmaktadır.

Yapay sinir ağlarının günümüzde en yaygın olarak kullanılan modeli çok katmanlı algılayıcı ağlarıdır. Çok katmanlı yapay sinirleri temelde 3 kısımdan oluşmaktadır:

- Giriş katmanı: Bu katmanda bilgi işleme gerçekleşmez. Giriş verileri alınarak gizli katmanına iletilir.
- Gizli katmanlar: Bir dizi proses elemanından oluşur. Girdi katmanındaki her bir elemanı, gizli katmandaki proses elemanlarının tümüne bağlıdır. Bu kısımda girdi katmanından gelen bilgiler işlenir. Bir adet gizli katman ile birçok problemi çözmek mümkündür. Ancak problemin karmaşıklığına bağlı olarak birbirine ardışık bağlanmış birden fazla gizli katman da kullanılabilir.
- Çıkış katmanı: Gizli katmandan gelen işlenmiş veri sonuçlarını dışarı iletir.

Yapay sinir ağlarında girdi değerleri bağlantı ağırlıkları ile çarpılır ve birleştirme fonksiyonu ile birleştirilerek ağın net girdisi ortaya çıkar. Net girdi, aktivasyon fonksiyonunda işlem gördükten sonra ağın net çıktısı elde edilmiş olur. (Ulgen, 2017)

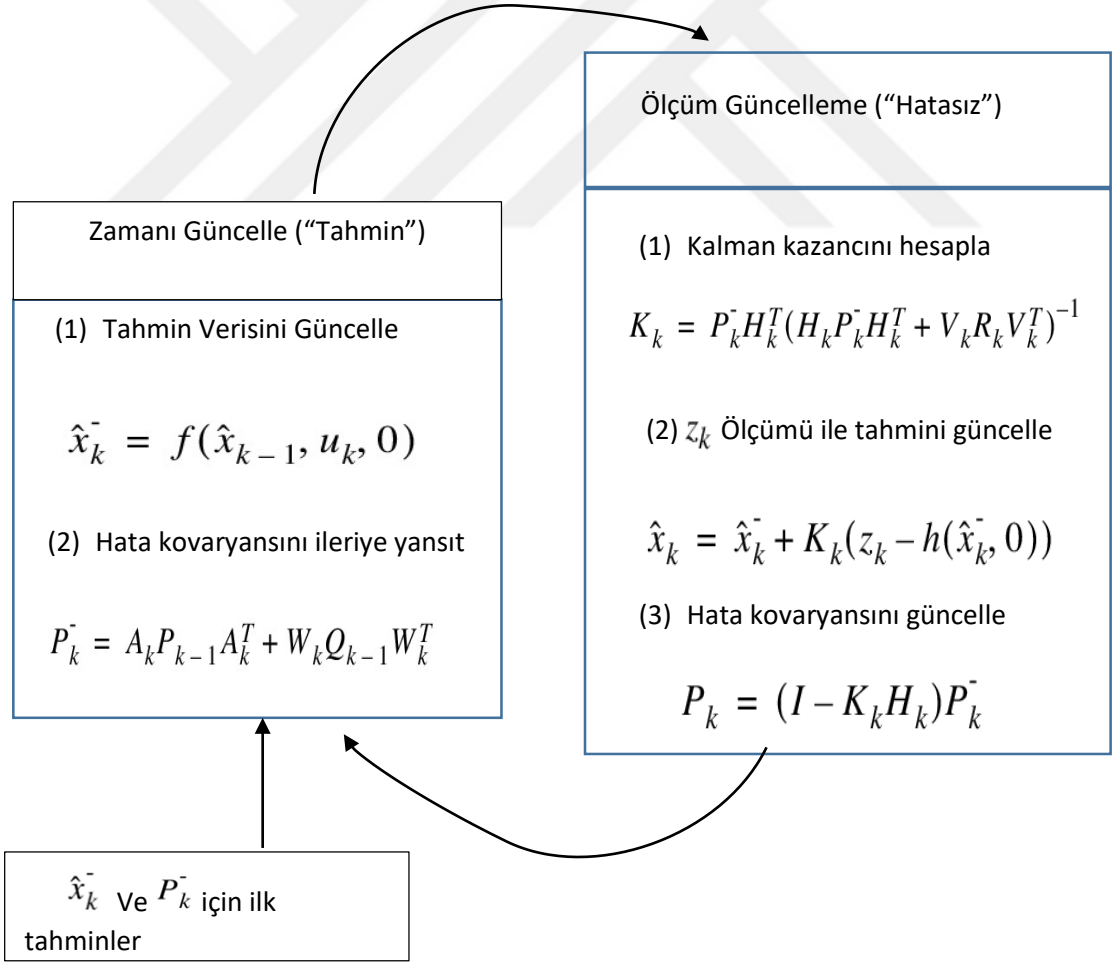


Şekil 21. Çok Katmanlı Yapay Sinir Ağları

3.3. Kalman Filtresi (Kalman Filter)

Kalman filtresi algoritması 20. Yüzyılın önemli buluşlarından biri olarak kabul edilir. Matematikçi Rudolf Kalman tarafından bulunmuş ve algoritma soyadı ile isimlendirilmiştir. Kalman filtresi, optimal tahminleme algoritması olarak tarif edilebilir. İsminde filtre ifadesi geçse de; önceki durum verilerinden yararlanarak, sonraki durum verilerini tahmin etmeye yarayan bir algoritmadır. Kalman filtresi algoritması, içerisinde gürültülü verilerinde bulunduğu verilerden, gerçekleşen veriye mümkün olduğunca yakın tahminleme yapılabilmesini mümkün kılar.

Arşiv verilerini ağırlıklandırarak yakın zamandaki verilerin ağırlığını ortalamada yüksek tutan bir algoritmadır, arşiv verileri ile anlık veriyi kullanarak tahminleme yapmaktadır, yapılan bir çalışmaya göre otobüsün bir sonraki istasyona varış zamanını %12 hata oranıyla tahmin edilebilmektedir (Md Noor, Yik, Kolandaisamy, ve Ahmedy, 2020).



Şekil 22. Kalman Filtresi Formül Gösterimi (Esme, 2009)

4. ÖNERİLEN YÖNTEMDE KULLANILAN YAKLAŞIM

Yapılan çalışmalar incelendiğinde, yakın zaman verileri kullanılmakta, veri seti çok küçük tutulmaktadır. Kullanılan veriler gün tipi olarak, iş günleri ve tatil günleridir. Saat tipi olarak ise trafiğin yoğun olduğu saatler ve rahat olduğu saatlerdir. Bu krtiterler dikkate alınarak veri düzenlemesi yapılmaktadır. Ancak gün tipini sadece tatil ve iş günü olarak ele almak elde edilecek tahmin verisinin kalitesini düşürmektedir. Hafta içi sefer sürelerinin cumartesi günkü sefer sürelerinden daha uzun olduğu düşünülebilir. Ancak Tablo7 deki veriler incelendiğinde, ortalama sefer sürelerinin en uzun olduğu günün Cumartesi günü olduğu görünmektedir. Yine saat dilimlerini sadece yoğun saatler ve rahat saatler olarak ele almak elde edilecek tahmin verisindeki hata oranını arttıracaktır. Saat dilimleri incelendiğinde her saat aralığında ki trafik verisi diğer saat aralığından mutlaka farklılık göstermektedir (Tablo 8). Mevcut algoritma incelendiğinde, büyük oranda hatalı tahmin verilerinin, negatif ya da çok yüksek sürelerin veri seti içerisinde yer almasından kaynaklandığı görülmektedir.

Bu çalışmada; verilerin kümelenmesi ile ilgili diğer çalışmalardan farklı bir çalışma yapılarak 3 farklı gün tipi belirlenmiştir. Gün tipleri Hafta içi, Cumartesi ve Pazar olarak değerlendirilmiştir. Ayrıca her bir gün için tüm saat dilimleri ayrı birer küme olarak ele alınmış ve bu kümeler üzerinde yapılan kümeleme çalışması sayesinde benzerlik gösteren saatler kümelenmiştir. Hem arşiv hem de anlık verilerden aynı zamanda faydalanılarak ortalama belirlenmektedir. Bu sayede arşiv verileri sayesinde, tahmin yapılacak rota üzerinde, rutin yaşanan trafik bilgisi anlık verinin olmadığı zamanlarda tahmin verisi olarak kullanılırken, anlık veri sayesinde rota üzerindeki kaza, yağmur vb. geçici durumların etkileri ortalamaya dâhil edilecektir. Verilerin temizlenmesinde kullanılan yöntem sayesinde rota üzerinde ortaya çıkan minimum ve maksimum süreler hesaplanmış bu sayede küme içerisinde negatif ya da aşırı yüksek süre verilerinin ortalamaya dahil edilmesi engellenmiştir.

Yapılacak çalışmanın adımları aşağıdaki gibidir;

- Verilerin elde edilmesi
- Verilerin kümelenmesi

a. Verilerin gün tipine göre kümelenmesi
b. Gün tipi verilerin saat dilimine göre kümelenmesi

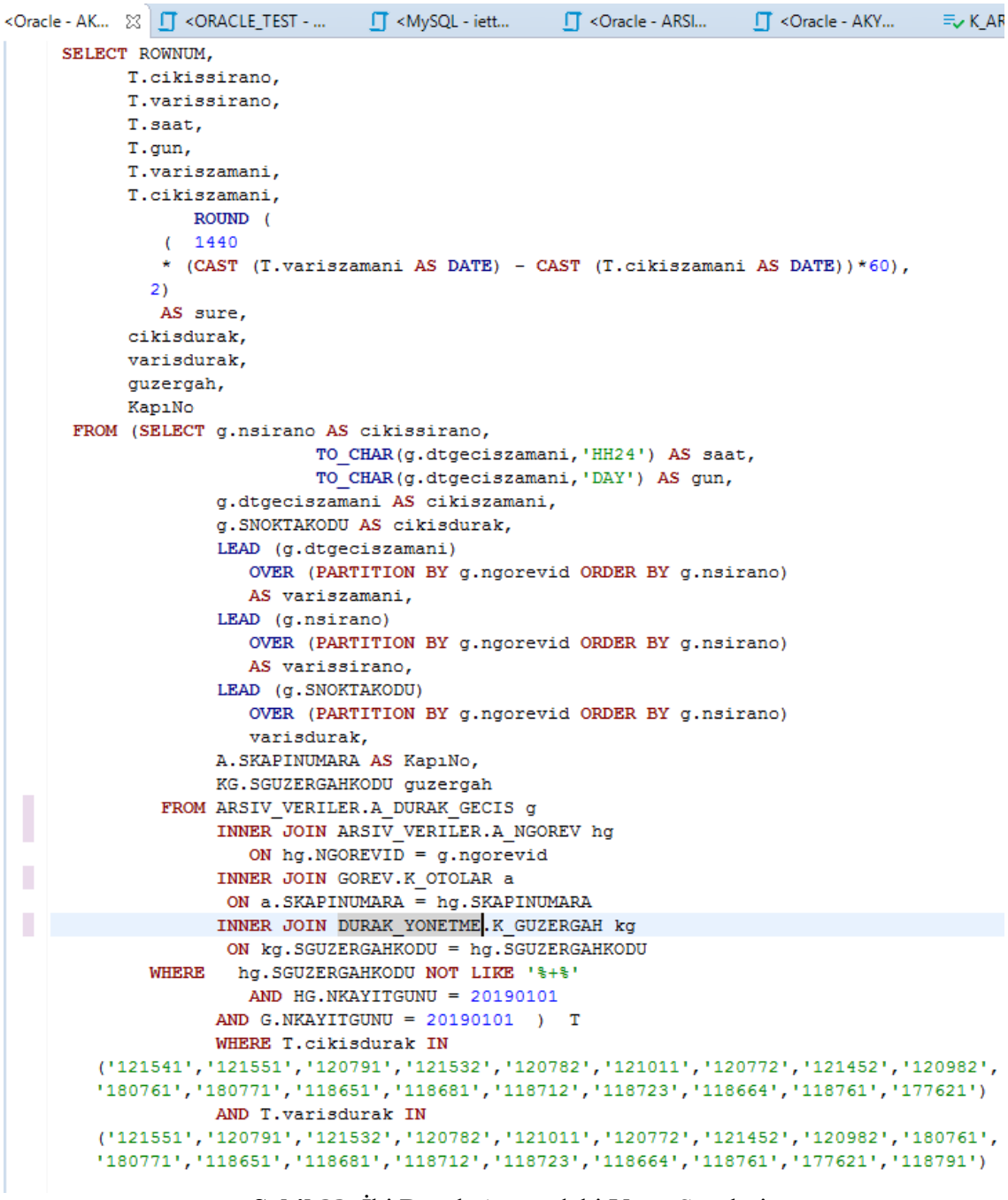
- Yöntemin algoritmasının Uygulanması

a. Verilerin temizlenmesi
b. Arşiv verilerinin ortalamasının belirlenmesi
i. Altı aylık arşiv verilerinin ortalaması
ii. Aylık arşiv verilerinin ortalaması
iii. Arşiv verilerinin ortalamasını belirlenmesi

- Anlık geçiş verilerini ortalamaya dahil edilmesi
- Belli periyodlarla arşiv verisinin güncellenmesi
- Verinin paylaşılması

4.1. Verilerin Elde Edilmesi

Verilerin elde edilebilmesi için IETT nin izni dahilinde gerçek veri tabanındaki birçok şema ve o şemaların altındaki tablolarla çalışmak gerekmiştir. Bu çalışma için sorguda isimlerini değiştirerek verdiğim GOREV, DURAK_YONETME, ARSIV_VERILER şemalarının altındaki ilgili tabloların ilgili kolonları çekilmiştir. (Şekil 23) 2019 yılının ilk altı ayında gerçekleştirilen 121541 kodlu durak ile 118791 kodlu duraklar (55 hattının ilk 20 durağı) arasında kalan duraklardan geçen tüm hatlara dair sefer gerçekleştiren araçların iki durak arası mesafeyi alma süresi saniye cinsinden hesaplanmaktadır. Bu hesaplamaların yapılabilmesi için Şekil 23' deki sorgu yazılmıştır. Bu sorgunun sonucu günlüktür. Her gün için ortalama 5500 satır veri gelmektedir. Bu çalışmada 6 aylık veriyi elde etmek için; toplamda 1.5 milyon satırın üzerinde veri kullanılmış, 59 farklı hat ve bu hatlarda görevli 80 farklı güzergâh kullanan 1324 adet farklı kapı nolu aracın durak geçiş verileri ile çalışılmıştır.



Şekil 23. İki Durak Arasındaki Varış Süreleri

Durak geçiş verileri için yazılan sorguda LEAD fonksiyonu kullanılmıştır. LEAD fonksiyonunu kullanarak, “VARISSIRANO” kolonundaki verinin “CIKISSIRANO” dan bir fazla olması sağlanmıştır, bu sayede sıralı iki durak arasındaki seyahat süresi hesaplanmaktadır.(Tablo 5)

Tablo 5. Sıralı İki Durak Arası Varış Süreleri Sorgu Sonucu

SELECT ROWNUM, T.cikissirano, T.varissirar | Enter a SQL expression to filter results (use Ctrl+Space)

	CIKISSIRANO	VARISSIRANO	SAAT	GUN	VARISZAMANI	CIKISZAMANI	SURE	CI	VAR	GUZE	KAPI
153	13	15	15	SALI	2019-01-01 15:43:15	2019-01-01 15:41:11	124	118723	118761	EM2_G_D0	B-1604
154	15	16	15	SALI	2019-01-01 15:45:12	2019-01-01 15:43:15	117	118761	177621	EM2_G_D0	B-1604
155	13	15	16	SALI	2019-01-01 16:15:42	2019-01-01 16:14:06	96	118723	118761	EM2_G_D0	A-107
156	15	16	16	SALI	2019-01-01 16:17:46	2019-01-01 16:15:42	124	118761	177621	EM2_G_D0	A-107
157	13	15	16	SALI	2019-01-01 16:26:05	2019-01-01 16:22:59	186	118723	118761	EM2_G_D0	B-1718
158	15	16	16	SALI	2019-01-01 16:27:17	2019-01-01 16:26:05	72	118761	177621	EM2_G_D0	B-1718
159	13	15	17	SALI	2019-01-01 17:28:55	2019-01-01 17:27:07	108	118723	118761	EM2_G_D0	A-107
160	13	14	17	SALI	2019-01-01 17:38:00	2019-01-01 17:34:46	194	118723	118664	EM2_G_D0	B-1718
161	14	15	17	SALI	2019-01-01 17:38:14	2019-01-01 17:38:00	14	118664	118761	EM2_G_D0	B-1718
162	15	16	17	SALI	2019-01-01 17:38:54	2019-01-01 17:38:14	40	118761	177621	EM2_G_D0	B-1718
163	15	16	18	SALI	2019-01-01 18:17:46	2019-01-01 18:16:23	83	118761	177621	EM2_G_D0	A-105
164	13	15	18	SALI	2019-01-01 18:34:45	2019-01-01 18:33:09	96	118723	118761	EM2_G_D0	A-107
165	15	16	18	SALI	2019-01-01 18:36:11	2019-01-01 18:34:45	86	118761	177621	EM2_G_D0	A-107
166	13	15	17	SALI	2019-01-01 17:53:38	2019-01-01 17:52:11	87	118723	118761	EM2_G_D0	B-1750
167	15	16	17	SALI	2019-01-01 17:55:00	2019-01-01 17:53:38	82	118761	177621	EM2_G_D0	B-1750
168	15	16	18	SALI	2019-01-01 18:07:45	2019-01-01 18:06:32	73	118761	177621	EM2_G_D0	B-1604
169	13	15	19	SALI	2019-01-01 19:33:43	2019-01-01 19:31:45	118	118723	118761	EM2_G_D0	A-106
170	13	15	19	SALI	2019-01-01 19:12:23	2019-01-01 19:10:26	117	118723	118761	EM2_G_D0	B-1604
171	15	16	19	SALI	2019-01-01 19:13:31	2019-01-01 19:12:23	68	118761	177621	EM2_G_D0	B-1604
172	13	15	21	SALI	2019-01-01 21:02:27	2019-01-01 21:00:33	114	118723	118761	EM2_G_D0	B-1718
173	15	16	21	SALI	2019-01-01 21:04:07	2019-01-01 21:02:27	100	118761	177621	EM2_G_D0	B-1718
174	13	15	19	SALI	2019-01-01 19:03:29	2019-01-01 19:02:10	79	118723	118761	EM2_G_D0	B-1750
175	15	16	19	SALI	2019-01-01 19:04:53	2019-01-01 19:03:29	84	118761	177621	EM2_G_D0	B-1750
176	13	15	19	SALI	2019-01-01 20:01:02	2019-01-01 19:59:30	92	118723	118761	EM2_G_D0	B-1657
177	15	16	20	SALI	2019-01-01 20:01:52	2019-01-01 20:01:02	50	118761	177621	EM2_G_D0	B-1657
178	13	15	20	SALI	2019-01-01 20:43:26	2019-01-01 20:41:58	88	118723	118761	EM2_G_D0	A-106
179	15	16	20	SALI	2019-01-01 20:44:34	2019-01-01 20:43:26	68	118761	177621	EM2_G_D0	A-106
180	13	15	21	SALI	2019-01-01 21:17:30	2019-01-01 21:16:15	75	118723	118761	EM2_G_D0	B-1657
181	15	16	21	SALI	2019-01-01 21:18:05	2019-01-01 21:17:30	35	118761	177621	EM2_G_D0	B-1657
182	13	14	19	SALI	2019-01-01 19:53:17	2019-01-01 19:50:46	151	118723	118664	EM2_G_D0	B-1718
183	14	15	19	SALI	2019-01-01 19:53:59	2019-01-01 19:53:17	42	118664	118761	EM2_G_D0	B-1718
184	15	16	19	SALI	2019-01-01 19:55:20	2019-01-01 19:53:59	81	118761	177621	EM2_G_D0	B-1718
185	8	9	07	SALI	2019-01-01 07:41:29	2019-01-01 07:40:40	49	180761	180771	500A_G_D0	A-286
186	9	10	07	SALI	2019-01-01 07:42:49	2019-01-01 07:41:29	80	180771	118651	500A_G_D0	A-286
187	10	11	07	SALI	2019-01-01 07:43:50	2019-01-01 07:42:49	61	118651	118681	500A_G_D0	A-286
188	11	13	07	SALI	2019-01-01 07:45:22	2019-01-01 07:43:50	92	118681	118723	500A_G_D0	A-286
189	15	16	07	SALI	2019-01-01 07:48:39	2019-01-01 07:47:47	52	118761	177621	500A_G_D0	A-286
190	16	17	07	SALI	2019-01-01 07:49:52	2019-01-01 07:48:39	73	177621	118791	500A_G_D0	A-286
191	3	4	07	SALI	2019-01-01 07:48:59	2019-01-01 07:48:13	46	118712	118723	522_D_D0	C-002
192	4	4	07	SALI	2019-01-01 07:48:43	2019-01-01 07:48:59	-16	118723	118723	522_D_D0	C-002
193	4	4	07	SALI	2019-01-01 07:48:59	2019-01-01 07:48:43	16	118723	118723	522_D_D0	C-002
194	4	6	07	SALI	2019-01-01 07:50:18	2019-01-01 07:48:59	79	118723	118761	522_D_D0	C-002
195	6	7	07	SALI	2019-01-01 07:51:07	2019-01-01 07:50:18	49	118761	177621	522_D_D0	C-002
196	7	8	07	SALI	2019-01-01 07:52:07	2019-01-01 07:51:07	60	177621	118791	522_D_D0	C-002
197	2	3	08	SALI	2019-01-01 08:56:55	2019-01-01 08:55:45	70	118761	177621	54E_G_D8573	B-046
198	3	4	08	SALI	2019-01-01 08:58:27	2019-01-01 08:56:55	92	177621	118791	54E_G_D8573	B-046
199	21	22	06	SALI	2019-01-01 06:44:21	2019-01-01 06:41:45	156	120982	180761	37M_G_D0	B-1971
200	22	25	06	SALI	2019-01-01 06:46:39	2019-01-01 06:44:21	138	180761	118681	37M_G_D0	B-1971
201	25	27	06	SALI	2019-01-01 06:48:19	2019-01-01 06:46:39	100	118681	118723	37M_G_D0	B-1971
202	27	30	06	SALI	2019-01-01 06:51:35	2019-01-01 06:48:19	196	118723	177621	37M_G_D0	B-1971
203	30	31	06	SALI	2019-01-01 06:52:19	2019-01-01 06:51:35	44	177621	118791	37M_G_D0	B-1971
204	24	25	08	SALI	2019-01-01 08:25:39	2019-01-01 08:25:00	39	180761	180771	41AT_D_D0	B-1796
205	25	26	08	SALI	2019-01-01 08:27:03	2019-01-01 08:25:39	84	180771	118651	41AT_D_D0	B-1796
206	26	27	08	SALI	2019-01-01 08:27:46	2019-01-01 08:27:03	43	118651	118681	41AT_D_D0	B-1796
207	27	29	08	SALI	2019-01-01 08:28:59	2019-01-01 08:27:46	73	118681	118723	41AT_D_D0	B-1796
208	29	31	08	SALI	2019-01-01 08:31:33	2019-01-01 08:28:59	154	118723	118761	41AT_D_D0	B-1796
209	31	31	08	SALI	2019-01-01 08:31:33	2019-01-01 08:31:33	0	118761	118761	41AT_D_D0	B-1796
210	31	32	08	SALI	2019-01-01 08:32:29	2019-01-01 08:31:33	56	118761	177621	41AT_D_D0	B-1796
211	32	33	08	SALI	2019-01-01 08:33:56	2019-01-01 08:32:29	87	177621	118791	41AT_D_D0	B-1796
212	35	36	07	SALI	2019-01-01 07:21:37	2019-01-01 07:20:39	58	180761	180771	36M_G_D0	O2047
213	36	37	07	SALI	2019-01-01 07:22:31	2019-01-01 07:21:37	54	180771	118651	36M_G_D0	O2047
214	37	38	07	SALI	2019-01-01 07:23:11	2019-01-01 07:22:31	40	118651	118681	36M_G_D0	O2047
215	38	40	07	SALI	2019-01-01 07:26:47	2019-01-01 07:23:11	216	118681	118723	36M_G_D0	O2047
216	40	42	07	SALI	2019-01-01 07:27:59	2019-01-01 07:26:47	72	118723	118761	36M_G_D0	O2047
217	42	43	07	SALI	2019-01-01 07:28:48	2019-01-01 07:27:59	49	118761	177621	36M_G_D0	O2047
218	43	44	07	SALI	2019-01-01 07:29:59	2019-01-01 07:28:48	71	177621	118791	36M_G_D0	O2047
219	35	36	15	SALI	2019-01-01 15:10:53	2019-01-01 15:09:58	55	180761	180771	36M_G_D0	O1004
220	36	37	15	SALI	2019-01-01 15:11:43	2019-01-01 15:10:53	50	180771	118651	36M_G_D0	O1004
221	37	38	15	SALI	2019-01-01 15:12:36	2019-01-01 15:11:43	53	118651	118681	36M_G_D0	O1004
222	38	40	15	SALI	2019-01-01 15:14:11	2019-01-01 15:12:36	95	118681	118723	36M_G_D0	O1004
223	40	42	15	SALI	2019-01-01 15:16:36	2019-01-01 15:14:11	145	118723	118761	36M_G_D0	O1004

Bu tabloda veriler ham hali ile çekilmiştir. Veriler dikkatli bir şekilde incelendiğinde bazılarının sorunlu olduğu görülmektedir. 192. satırdaki -16 saniyelik veri örnek olarak gösterilebilir. Tabloda "CIKISSIRANO" sütunu otobüsün çıkış yaptığı durağı, "VARISSIRANO" sütunu otobüsün geldiği durağı belirtir, burada verinin kabul edilebilir olması için "VARISSIRANO" nun "CIKISSIRANO" dan bir fazla olması gerekir, ancak tabloya dikkatli bakıldığında, bu kurala uymayan fazlaca veri olduğu fark edilmektedir. Bu veriler veri temizleme aşamasında temizlenecektir.

Tabloda dikkat edilmesi gereken bir başka nokta da durak kodu aynı olmasına rağmen durak numarası farklı olan satırlar var, bunun sebebi verinin bozuk olması değildir, her hatta ait sıra numaralarının farklılığıdır, örneğin belirlemiş olduğumuz kesit içerisinde 36M hattında 118651 kodlu durak 37. sıradayken, 55 hattında 13. sıradadır.

4.2. Verilerin Gün Tipine ve Saat Dilimine Göre Kümelenmesi

Bir küme temsil ettiği nesneleri en iyi şekilde ifade edecek biçimde düzenlenir. Kümeleme işleminin uygulandığı veri setindeki her bir veriye nesne adı verilir. Bu nesneler iki boyutlu düzlem üzerinde noktalarla gösterilir. Kümeleme analizi, veri indirgeme veya nesnelerin doğal sınıflarını bulma gibi çeşitli amaçlarla kullanılmaktadır. Bu alanlardan en çok gündemde olanlar örüntü tanıma, veri analizi, resim tanıma, metin madenciliği alanında yapılan araştırmalardır (Oğuz, 2018). Tahminleme problemlerinin çözümünde öne çıkan kümeleme algoritmalarından 2 tanesi olan K-Means ve DBScan kümeleme algoritmaları incelenmiş, bu çalışmada gün tipi ve saat dilimine göre kümeleme probleminin çözümüne uygun olması sebebiyle DBScan kümeleme algoritması tercih edilmiştir.

Bu çalışmaya konu algoritmanın temelinde yapılan işlem ilk olarak, mevcut arşiv verileri üzerinde kümeleme işlemi yapılmaktadır. Kümeleme işleminde takip edilecek adımlar şunlardır.

- Gün tiplerine göre kümeleme
- Saat dilimlerine göre kümeleme

4.2.1. Gün tiplerine göre kümeleme

Her günün aynı saatlerindeki sefer verileri incelendiğinde hafta içindeki günlerin aynı saatlerdeki sefer süreleri uyumluluk göstermekle birlikte Cumartesi ve Pazar günleri belirgin farklılık göstermektedir (Tablo6, Tablo 7). Bu nedenle veriler öncelikle gün tipleri olarak kümelere ayrılmalıdır. Mevcut algoritma da gün tipleri olarak Hafta İçi = H, Cumartesi= C ve Pazar(Tatil Günleri)=P şeklinde kümeleme yapılmaktadır.

Tablo 6. 55 Hattının 2019 Yılı Sefer Tamamlama Süreleri Ve Gün Tipleri

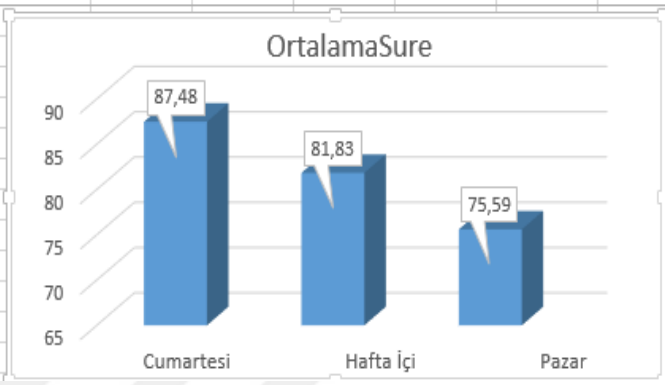
NARSIVGOREVID	NGOREVI	SHATKODU	SSOFORSICIL	SKAPINUMA	DTBASLAMAZAMANI	DTBITISZAMANI	NKAYITGUNU	SEFER SÜRESİ	GÜN
173.149.653	124.213	55	H00377	B-135	19.2.2019 16:40	19.2.2019 19:12	20190219	02:32:38	H
173.097.958	134.754	55	H00377	B-135	19.2.2019 19:07	19.2.2019 20:38	20190219	01:30:42	H
173.214.267	128.628	55	H00377	B-135	20.2.2019 16:47	20.2.2019 18:46	20190220	01:59:41	H
173.157.724	128.099	55	H00377	B-135	20.2.2019 07:00	20.2.2019 08:16	20190220	01:16:37	H
173.157.725	134.903	55	H00377	B-135	20.2.2019 08:22	20.2.2019 10:06	20190220	01:44:43	H
173.157.726	128.152	55	H00377	B-135	20.2.2019 10:26	20.2.2019 12:26	20190220	02:00:15	H
173.157.727	128.577	55	H00377	B-135	20.2.2019 12:30	20.2.2019 14:21	20190220	01:50:27	H
173.157.728	128.604	55	H00377	B-135	20.2.2019 14:31	20.2.2019 16:25	20190220	01:54:27	H
173.157.729	134.945	55	H00377	B-135	20.2.2019 19:02	20.2.2019 20:55	20190220	01:52:55	H
173.397.678	136.620	55	H00377	B-135	23.2.2019 12:52	23.2.2019 16:30	20190223	03:38:58	C
173.375.216	130.314	55	H00377	B-135	23.2.2019 07:19	23.2.2019 08:35	20190223	01:16:06	C
173.375.217	132.702	55	H00377	B-135	23.2.2019 08:45	23.2.2019 09:58	20190223	01:13:06	C
173.375.218	136.602	55	H00377	B-135	23.2.2019 10:53	23.2.2019 12:14	20190223	01:20:41	C
173.375.219	136.638	55	H00377	B-135	23.2.2019 14:59	23.2.2019 16:48	20190223	01:48:22	C
173.447.255	137.150	55	H00377	B-135	24.2.2019 12:44	24.2.2019 16:14	20190224	03:30:07	P
173.403.169	129.807	55	H00377	B-135	24.2.2019 10:33	24.2.2019 11:38	20190224	01:05:18	P
173.403.170	137.168	55	H00377	B-135	24.2.2019 15:00	24.2.2019 16:27	20190224	01:27:40	P
173.403.171	137.186	55	H00377	B-135	24.2.2019 17:09	24.2.2019 18:26	20190224	01:16:58	P
173.517.504	135.533	55	H00377	B-135	26.2.2019 09:25	26.2.2019 11:10	20190226	01:44:47	H
173.517.505	127.940	55	H00377	B-135	26.2.2019 11:24	26.2.2019 13:19	20190226	01:55:35	H
173.517.507	127.958	55	H00377	B-135	26.2.2019 15:35	26.2.2019 17:40	20190226	02:05:09	H
173.517.508	135.602	55	H00377	B-135	26.2.2019 17:52	26.2.2019 20:02	20190226	02:10:29	H
172.684.920	137.789	55	H00570	B-141	12.2.2019 06:45	12.2.2019 07:58	20190212	01:13:21	H
172.684.921	133.608	55	H00570	B-141	12.2.2019 08:11	12.2.2019 09:49	20190212	01:38:00	H
172.684.922	133.218	55	H00570	B-141	12.2.2019 10:13	12.2.2019 11:41	20190212	01:28:39	H
172.684.923	133.227	55	H00570	B-141	12.2.2019 12:11	12.2.2019 13:44	20190212	01:33:41	H
172.684.924	133.236	55	H00570	B-141	12.2.2019 14:14	12.2.2019 15:56	20190212	01:42:12	H
172.684.925	133.244	55	H00570	B-141	12.2.2019 16:30	12.2.2019 18:39	20190212	02:08:51	H
172.684.926	133.650	55	H00570	B-141	12.2.2019 18:49	12.2.2019 20:55	20190212	02:06:15	H
172.792.577	123.124	55	H00570	B-141	14.2.2019 10:26	14.2.2019 12:11	20190214	01:44:37	H
172.792.578	123.133	55	H00570	B-141	14.2.2019 12:28	14.2.2019 13:59	20190214	01:31:02	H
172.840.968	123.142	55	H00570	B-141	14.2.2019 14:29	14.2.2019 16:29	20190214	02:00:36	H
172.849.337	123.150	55	H00570	B-141	14.2.2019 16:44	14.2.2019 18:43	20190214	01:58:45	H
172.849.441	134.041	55	H00570	B-141	14.2.2019 18:59	14.2.2019 20:32	20190214	01:32:30	H
172.946.268	128.144	55	H00570	B-141	16.2.2019 07:15	16.2.2019 08:13	20190216	00:57:46	C
172.946.269	128.170	55	H00570	B-141	16.2.2019 08:37	16.2.2019 09:50	20190216	01:12:25	C
172.946.270	128.594	55	H00570	B-141	16.2.2019 10:44	16.2.2019 12:01	20190216	01:16:56	C

Tablo 6'daki veriler, 2019 yılında 55 hattında tamamlanmış sefer verilerinin ayrıntılarını göstermektedir. Bu tablonun tamamı dikkate alınarak 55 hattının 2019 yılı için saat 10 diliminde başlayan seferleri dikkate alınarak gün tiplerine göre sefer tamamlama sürelerini kontrol ettiğimizde Tablo 7 deki sonuçlar ortaya çıkmaktadır. Bu sonuçları

yorumlamak gerektiğinde Cumartesi gününün 87 dakika ile diğer günlere göre daha uzun, Pazar gününün ise 75 dakika ile en kısa sefer ortalamasına sahip olduğu görülmektedir (Tablo 7).

Tablo 7. 2019 Yılı Gün Tiplerine Göre Ortalama Sefer Tamamlama Süreleri

Yil	OrtalamaSure	HaftaninGu	Saat
2019	87,48	Cumartesi	10
2019	81,83	Hafta İçi	10
2019	75,59	Pazar	10



4.2.2. Saat dilimlerine göre kümeleme

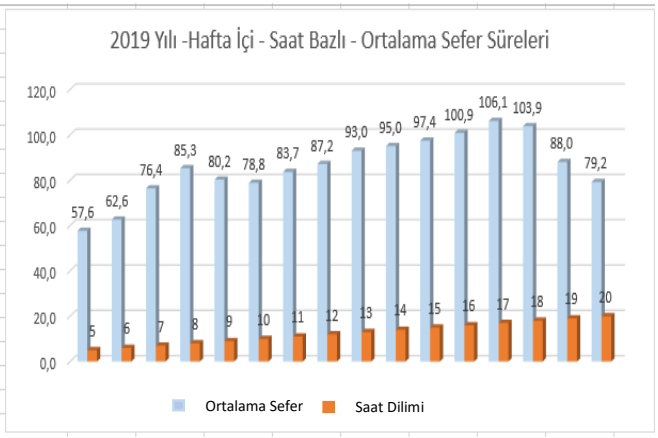
Gün tiplerine göre veriler incelendiğinde saat bazlı sefer süre farklılıkları göze çarpmaktadır. Gün tiplerine göre kümeleme yaptıktan sonra saat bazlı kümeleme yapmak verinin kalitesini daha da arttıracaktır. 2019 yılında 55 hattında hafta içi gerçekleşen sefer sürelerini saat bazlı incelediğimizde 57 dakika ile 106 dakika arasında sefer tamamlama süreleri olduğu ve diğer saatlerde de birbirinden farklı sefer sürelerinin gerçekleştiği görülmektedir (Tablo 8, Tablo 9). Saat dilimine göre kümeleme de, saat dilimleri seferlerin başlangıç saatlerine göre belirlenmektedir. Örneğin saat 14:01'de başlayan sefer için saat dilimi 14 olarak belirlenir, 15:58 de başlayan seferin saat dilimi ise 15 olarak belirlenir sefer başlangıç anındaki saat dilimi kümeleme için dikkate alınmaktadır. Tablo 9 ve Şekil 24 incelendiğinde, sefer sürelerinin saatlere göre benzerlikler içerdiği gözlemlenmektedir.

Tablo 8. 2019 Yılı Sefer Verileri

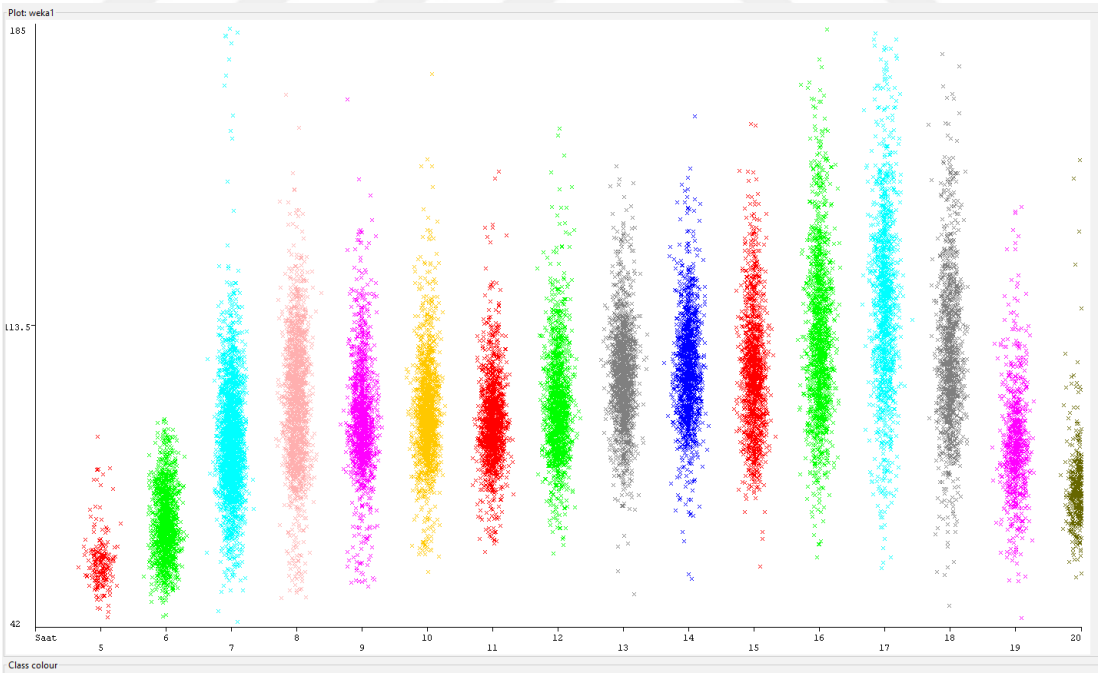
SHATKODU	SSOFORSICIL	SKAPINUM	DTBASLAMAZAN	DTBITISZAMANI	NKAYITGUNU	SEFER SÜRESİ	GÜN TİPİ	SAAT DİLİMİ
55	O05619	B-1717	18.1.2019 07:46	18.1.2019 09:24	20190118	01:38:23	H	9
55	O05619	B-1717	18.1.2019 09:45	18.1.2019 11:22	20190118	01:36:31	H	11
55	O05619	B-1717	18.1.2019 12:01	18.1.2019 13:44	20190118	01:43:09	H	13
55	O05619	B-1717	18.1.2019 16:34	18.1.2019 17:57	20190118	01:23:33	H	17
55	O05619	B-1717	19.1.2019 16:50	19.1.2019 18:24	20190119	01:34:43	C	18
55	O05619	B-1717	19.1.2019 14:28	19.1.2019 16:06	20190119	01:37:37	C	16
55	O05619	B-1717	20.1.2019 11:27	20.1.2019 12:32	20190120	01:04:43	P	12
55	O05619	B-1717	20.1.2019 13:35	20.1.2019 15:09	20190120	01:33:32	P	15
55	O05506	B-1772	7.1.2019 09:34	7.1.2019 11:13	20190107	01:38:52	H	11
55	O05506	B-1772	7.1.2019 11:52	7.1.2019 13:12	20190107	01:19:49	H	13
55	O05506	B-1772	7.1.2019 14:07	7.1.2019 15:45	20190107	01:38:11	H	15
55	O05506	B-1772	7.1.2019 16:29	7.1.2019 18:13	20190107	01:43:32	H	18
55	O05506	B-1772	10.1.2019 10:03	10.1.2019 11:49	20190110	01:45:05	H	11
55	O05506	B-1772	10.1.2019 12:19	10.1.2019 13:38	20190110	01:18:28	H	13
55	O05506	B-1772	10.1.2019 14:35	10.1.2019 16:16	20190110	01:40:42	H	16
55	O05506	B-1772	9.3.2019 07:44	9.3.2019 08:50	20190309	01:05:36	C	8
55	O05506	B-1772	9.3.2019 09:24	9.3.2019 10:34	20190309	01:09:36	C	10
55	O05506	B-1772	9.3.2019 11:32	9.3.2019 13:02	20190309	01:29:25	C	13
55	O05506	B-1772	9.3.2019 15:44	9.3.2019 17:20	20190309	01:36:20	C	17
55	O05506	B-1772	9.3.2019 18:11	9.3.2019 19:44	20190309	01:33:22	C	19
55	O05506	B-1772	11.3.2019 06:15	11.3.2019 07:15	20190311	00:59:51	H	7
55	O05506	B-1772	11.3.2019 09:43	11.3.2019 11:12	20190311	01:29:09	H	11
55	O05506	B-1772	11.3.2019 11:48	11.3.2019 13:13	20190311	01:24:50	H	13
55	O05506	B-1772	11.3.2019 13:49	11.3.2019 15:24	20190311	01:35:15	H	15
55	O05506	B-1772	11.3.2019 18:30	11.3.2019 20:03	20190311	01:32:43	H	20
55	O05498	B-1775	6.1.2019 12:45	6.1.2019 13:51	20190106	01:05:26	P	13
55	O05498	B-1775	6.1.2019 14:53	6.1.2019 16:16	20190106	01:22:51	P	16
55	O05498	B-1775	6.1.2019 17:00	6.1.2019 18:18	20190106	01:17:30	P	18
55	O05498	B-1775	6.3.2019 09:41	6.3.2019 11:07	20190306	01:25:57	H	11
55	O05498	B-1775	6.3.2019 11:30	6.3.2019 12:55	20190306	01:25:09	H	12
55	O05498	B-1775	6.3.2019 13:34	6.3.2019 15:01	20190306	01:26:42	H	15
55	O05498	B-1775	6.3.2019 15:51	6.3.2019 17:27	20190306	01:36:13	H	17
55	O05498	B-1775	8.3.2019 11:48	8.3.2019 13:14	20190308	01:25:33	H	13
55	O05498	B-1775	8.3.2019 13:48	8.3.2019 15:27	20190308	01:39:08	H	15
55	O05498	B-1775	9.3.2019 16:10	9.3.2019 17:46	20190309	01:35:55	C	17
55	O05498	B-1775	9.3.2019 18:40	9.3.2019 19:59	20190309	01:19:21	C	19
55	O05429	B-092	10.1.2019 14:45	10.1.2019 16:27	20190110	01:41:43	H	16
55	O05429	B-092	12.1.2019 15:04	12.1.2019 16:33	20190112	01:29:10	C	16

Tablo 9. 2019 Yılında Hafta İçi Yapılan Seferlerin Saat Bazlı Ortalama Sefer Süreleri

Yil	OrtalamaSure	HaftaninGunuAdi	Saat
2019	57,6	Hafta İçi	5
2019	62,6	Hafta İçi	6
2019	76,4	Hafta İçi	7
2019	85,3	Hafta İçi	8
2019	80,2	Hafta İçi	9
2019	78,8	Hafta İçi	10
2019	83,7	Hafta İçi	11
2019	87,2	Hafta İçi	12
2019	93,0	Hafta İçi	13
2019	95,0	Hafta İçi	14
2019	97,4	Hafta İçi	15
2019	100,9	Hafta İçi	16
2019	106,1	Hafta İçi	17
2019	103,9	Hafta İçi	18
2019	88,0	Hafta İçi	19
2019	79,2	Hafta İçi	20

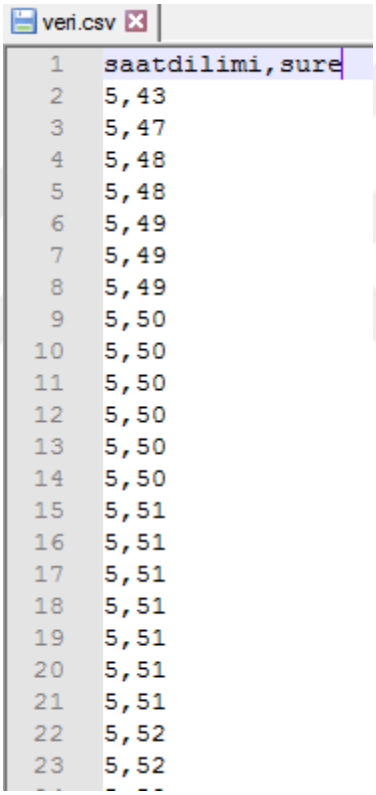


Tablo 9 incelendiğinde 2019 yılı içinde hafta içi yapılmış olan seferlerin saat aralıklarına göre ortalama sefer süreleri belirtilmiştir. Saat dilimlerine göre ortalama sefer süreleri 57.6 dakika ile 106.1 dakika arasındadır. Saat dilimlerine göre kümeleri incelediğimizde 16 adet küme oluşmuştur. Benzer kümeleri birleştirerek küme sayısını belirleyebilmek için, kümeleme algoritmalarından hiyerarşik kümeleme algoritması kullanılacaktır.



Şekil 24. Veri madenciliği araçlarından biri olan Weka'da oluşturulan, sürelere ait nokta yoğunluk grafiği

Hiyerarşik kümeleme algoritmasındaki temel mantık, benzer verilerin birleşmesine ya da bölünmesine dayanmaktadır. Bizim mevcuttaki 16 adet kümemizi, ideal küme sayısına düşürmemiz gerekmektedir. Bu yöntemi uygularken mevcut veri madenciliği araçlarından birinden faydalanmamız gerekir. Günümüzde Python, R, Spss, Weka, vb. birçok program kullanılarak veri madenciliği işlemleri kolaylıkla gerçekleştirilmektedir. Bizim yapacağımız işlem için Python tercih edilmiştir. Öncelikle verilerimizi hazırlayarak dosyamızı oluşturuyoruz. Kullanacağımız veriler, 55 hattına ait 2019 yılında gerçekleşen 20829 adet seferin, süreleri olacaktır. Veri etiketi olarak sefer saat dilimleri kullanılacaktır.



Şekil 25. Python da kullanılacak veri seti dosya içeriği

Hiyerarşik kümelemeyi içeren Python da hazırlanmış kodu incelediğimizde(Şekil 26), sırası ile ağağıdaki işlemler gerçekleştirilir;

- Uygulama da kullanılacak python kütüphaneleri çağrılır.
- Sonrasında csv formatında hazırlanmış veri setinin olduğu dosya tanımlanır.
- Süreler saat dilimine göre gruplanır ve ortalamaları belirlenir.

- Elde edilecek sonuçların gösterimi için dendrogram kullanılmıştır. Dendrogram üretilirken bağlantı yöntemi olarak “ward” kullanılmıştır.
- Dendrogram ın nereden kesileceği belirlenir. Bu uygulama için 20. İtesrasyondan sonra kesilmesi belirlenmiştir.

C:\Users\kkustul11\Desktop\kumelemesonnn.py

kumelemesonnn.py* | kmeans.py

```
import os
import pandas as pd
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import AgglomerativeClustering

dir_path = os.path.dirname(os.path.realpath(__file__))
veri = pd.read_csv(dir_path + "/veri.csv")

ortalama = veri.groupby('sure').saatdilimi.mean()

veri_ = pd.DataFrame({'saat': sorted(veri['sure'].unique()),'ortalama_sure':ortalama})
veri_ = veri_.reset_index(drop=True)

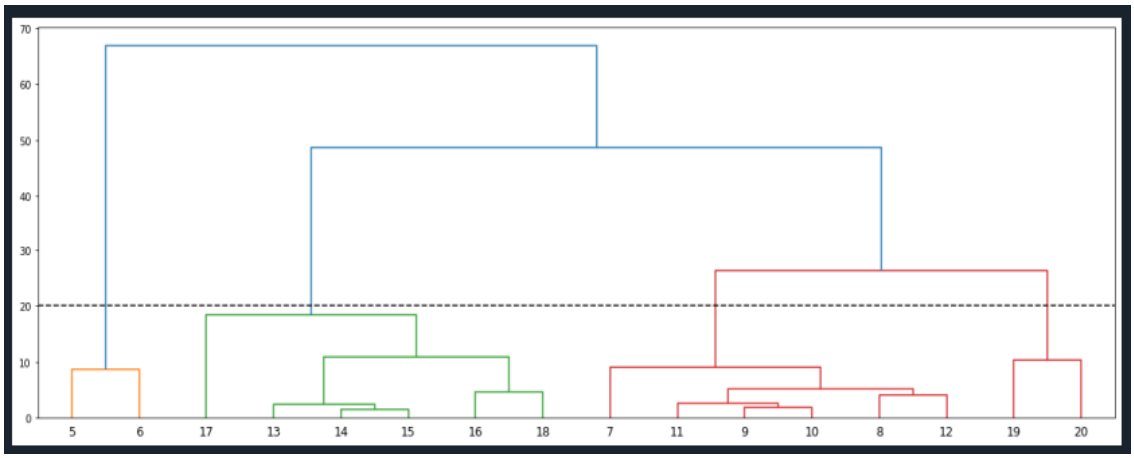
labels = range(5, 21) # Birleştirici hiyerarşik kümele analizinde başlaşgıçta her veri noktası küme olarak
# kabul ediliyor. Bu sebeple 20.000 verinin takibini yapmak için saat etiketine göre ortalamalar alındı.
linked = linkage(veri_, 'ward') #Kümeleme yapılırken, bağlantı metodu olarak ward kullanıldı.
# Ward, varyans  minimizasyonuna dayanan bir yöntem.
plt.figure(figsize=(20, 7))
dendrogram(linked, orientation='top', labels=labels, show_leaf_counts=True)
plt.axhline(20, ls="--", c="k")
plt.show()

```

Şekil 26. Hiyerarşik kümeleme için Python da yazılan kod

Elde edilen sonuç Şekil 27 deki gibi görünmektedir. Dendrogram incelendiğinde kesikli çizginin dendrogramı kestiği yerlerden aşağı doğru ağaç yapılarını alırsak, birleştirilmesi gereken kümeler şu şekilde belirlenmiştir;

- 1. Küme: 5, 6
- 2. Küme: 14, 15, 13, 16, 18, 17
- 3. Küme: 9, 10, 11, 8, 12, 7
- 4. Küme: 19, 20

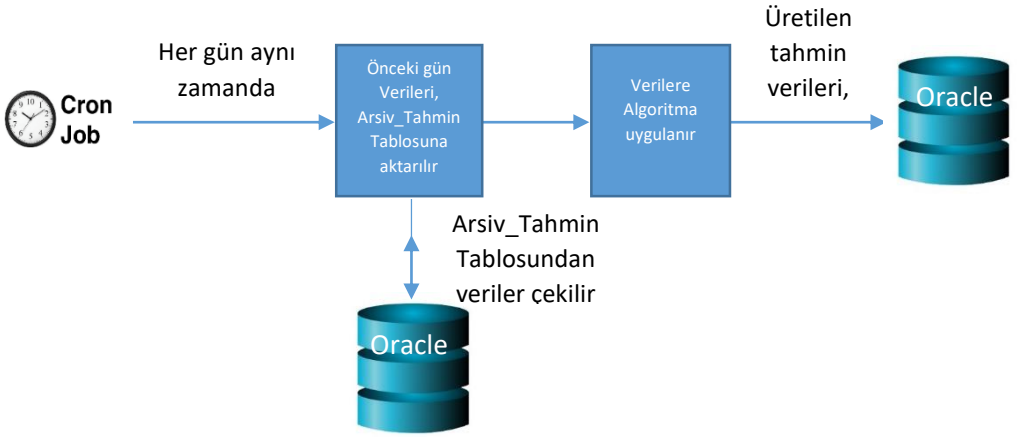


Şekil 27. Python da hazırlanan kümeleme algoritmasının dendrogram çıktısı

Dolayısıyla gün tiplerine bölündükten sonra saat olarak belirlenen kümelere bölündüğünde tahmin verisi daha sağlıklı hale gelecektir.

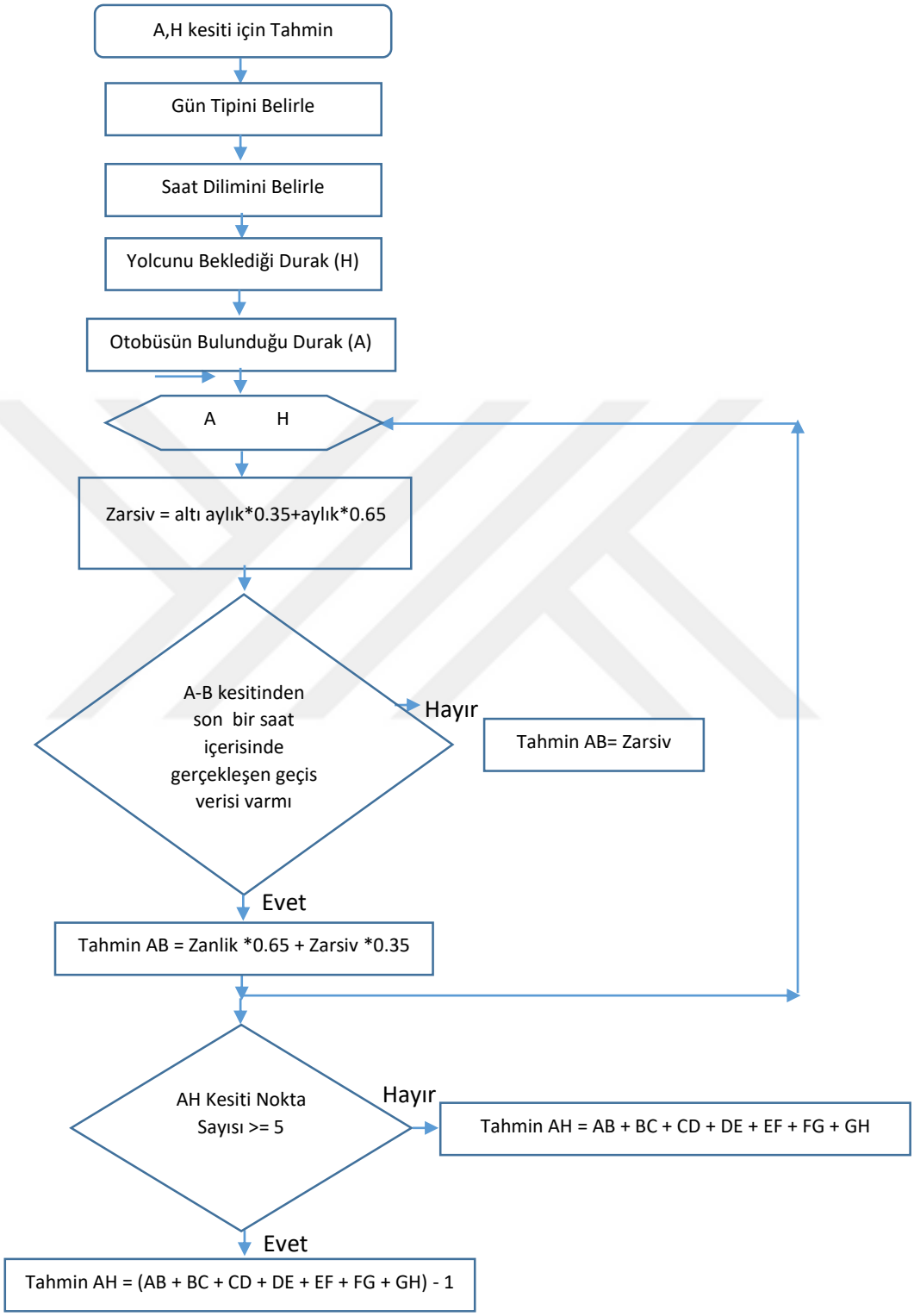
4.3. Yöntemin Uygulanması

Bu çalışmada uygulanacak yöntemin adımları şu şekilde olacaktır. Her gün, bir önceki günün gerçekleşen yolculuk verileri arşivlenecektir. Buradaki yolculuk verileri üzerinde veri temizleme işlemi yapıldığı için sorunlu veriler tabloda yer almayacaktır. Her bir kesit (ardışık iki durak aralığı) için 6 aylık ve 1 aylık ortalamalar belirlendikten sonra arşiv ortalamaları alınacaktır. Bu işlem her günün sonunda yapılacaktır, bu sayede veri temizleme arşiv verisinin oluşturulması süreci tamamlanmış olacak ve işlem yoğunluğu noktasında kolaylık sağlayacaktır. Tahmin üretimi için ilgili veriler arşiv ortalamalarının tutulduğu tablodan çekilerek algoritma uygulanacaktır.



Şekil 28. Tahmin Algoritmasının İşletilmesi

Ayrıntılarına baktığımızda takip edilecek adımları şu şekilde ifade edebiliriz;



Şekil 29. Tahmin Algoritmasının İşletilmesi

4.3.1. Verilerin temizlenmesi

Otobüslerin GPS antenlerinin arızalanması, Şoförün aktif seferi bitirememesi ya da çok erken bitirmesi, aracın yolda arıza yapması, araç bilgisayarı yazılımındaki bir hata vb. sebeplerden veri tabanında anlamsız veriler oluşur. Bu sefer süreleri bazen çok uzun olmakla birlikte bazen de çok kısa hatta negatif veriler olabilir. Bu veriler kirli veriler olarak düşünülebilir. Doğruluk oranı yüksek bir sonuç verisi elde edebilmemiz için mevcut veriyi kaliteli hale getirmemiz gerekir. 2019 yılı için 55 hattı sefer sürelerine baktığımızda, ilgili tablolarda (Tablo 10, 11) görüldüğü üzere 9 saatin üzerinde süren sefer yanında başlama saatinden çok önce biten seferin de varlığı görülmektedir.

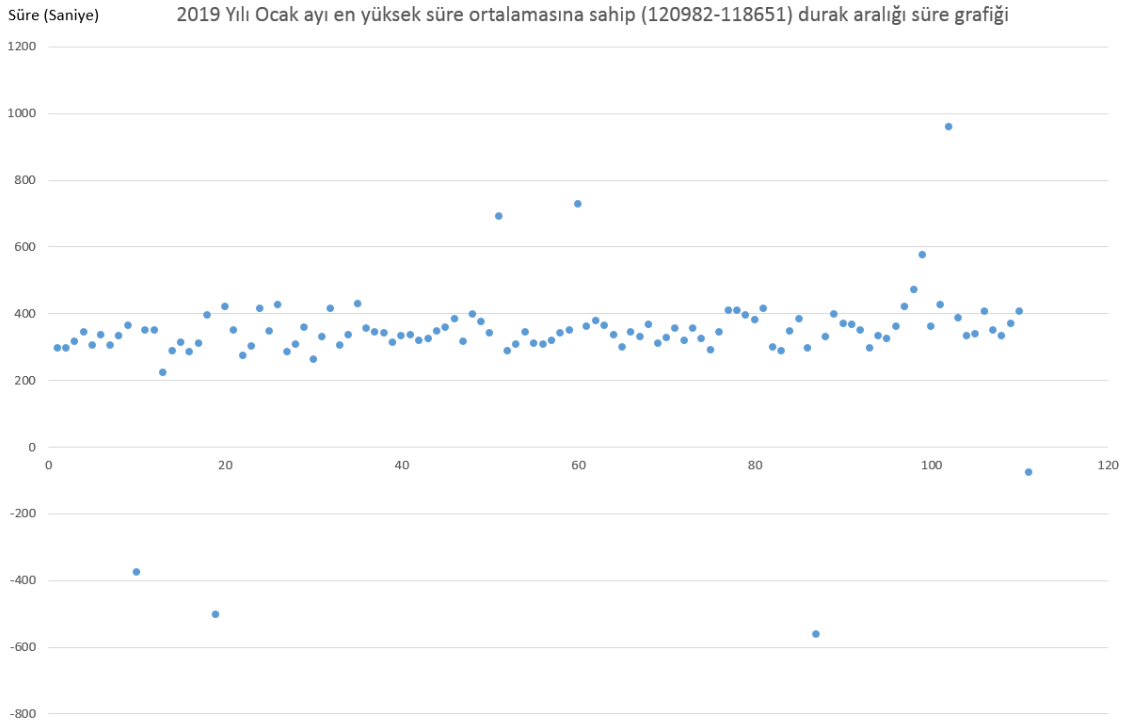
Tablo 10. 2019 Yılı En Uzun Sefer Süreleri

HAT	KAPI N	DURUM	BAŞLAMA	BİTİŞ	GÜN	SÜRE
55	B-102	T	5.9.2019 06:36	5.9.2019 20:32	20190905	13:56:25
55	B-1531	T	6.6.2019 06:20	6.6.2019 17:54	20190606	11:33:19
55	B-094	T	5.4.2019 10:44	5.4.2019 21:04	20190405	10:20:06
55	B-136	T	30.7.2019 06:02	30.7.2019 16:11	20190730	10:08:34
55	B-119	T	5.11.2019 06:00	5.11.2019 16:06	20191105	10:06:49
55	B-1531	T	6.6.2019 07:48	6.6.2019 17:54	20190606	10:06:15
55	B-089	T	26.6.2019 06:05	26.6.2019 16:05	20190626	10:00:58
55	B-1771	T	10.1.2019 06:10	10.1.2019 16:08	20190110	09:57:47
55	B-1722	T	17.7.2019 07:12	17.7.2019 17:09	20190717	09:57:38
55	B-1511	T	24.6.2019 06:35	24.6.2019 16:26	20190624	09:51:38
55	B-112	T	28.8.2019 06:55	28.8.2019 16:40	20190828	09:45:41

Tablo 11. 2019 Yılı Hatalı Sefer Süreleri – Başlama Süresinden Önce Biten Seferler

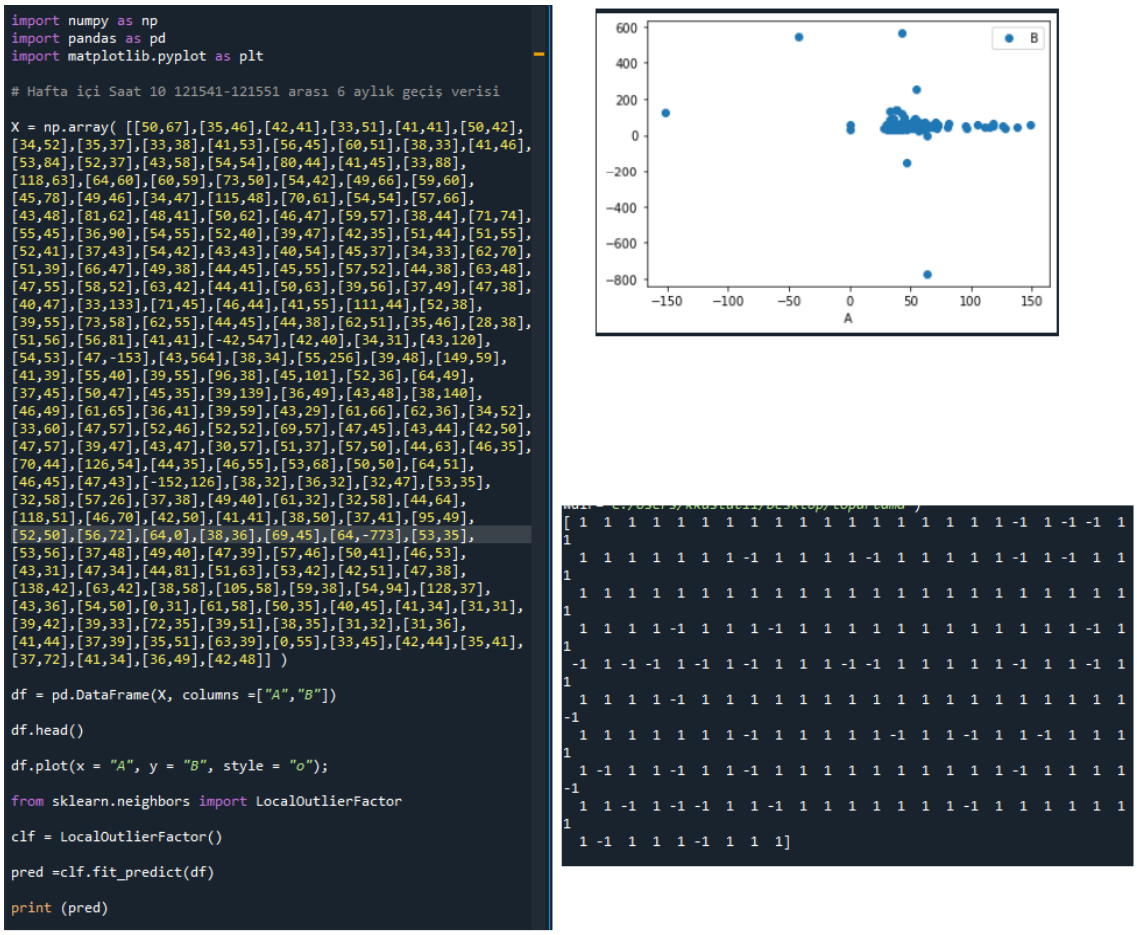
HAT	KAPI N	DURUM	BAŞLAMA	BİTİŞ	GÜN	SÜRE
55	B-1774	T.	7.9.2019 18:10	7.9.2019 12:16	20190907	##########
55	B-1633	T	11.3.2019 18:11	11.3.2019 13:02	20190311	##########
55	B-091	T.	27.6.2019 17:56	27.6.2019 12:50	20190627	##########
55	B-106	T.	1.1.2019 17:55	1.1.2019 14:54	20190101	##########
55	B-098	T	7.1.2019 17:36	7.1.2019 14:36	20190107	##########
55	B-158	T.	24.4.2019 17:31	24.4.2019 14:40	20190424	##########
55	B-1735	T	16.5.2019 12:30	16.5.2019 09:46	20190516	##########
55	B-1794	T	2.1.2019 14:26	2.1.2019 11:44	20190102	##########
55	B-1633	T	11.3.2019 15:43	11.3.2019 13:02	20190311	##########
55	B-141	T	10.5.2019 14:59	10.5.2019 12:21	20190510	##########

2019 yılının ocak ve şubat ayındaki 55 hattına ait tamamlanan 6500 seferi incelediğimizde, 45 adet sefer negatif tamamlanma süresi ya da çok düşük sefer süresine sahipken, 100 ün üzerinde sefer ise çok yüksek sürelerle tamamlanmış görünmektedir. İki durak arası geçiş verilerini kontrol ettiğimizde bu sayı çok daha fazla olmaktadır ve bu verilerin ortalamaya etkisi üretilen tahmin verisinin kalitesini düşürmektedir (Şekil 30). Öncelikle bu verilerin temizlenmesi ve sonrasında tahmin verisinin üretilmesi gerekmektedir.



Şekil 30. 120982-118651 Durak Aralığına Ait 2019 Yılı Ocak Ayı Süre Grafiği (Ham Veri)

Şekil 30' da grafiği verilen 2019 yılı ocak ayı geçiş verilerini kontrol ettiğimizde, negatif geçiş verileri ve ortalamanın çok üstündeki geçiş süreleri dikkat çekmektedir.



Şekil 31. Local Outlier Factor Algoritması İle Anormal Verilerin Tespiti

Bu çalışma da belirlenen küme sınırları incelenen 1.500.000 durak geçiş verisi üzerinde veri ön inceleme çalışması yapılmıştır. İdeal küme sınırlarını belirlemek için durak geçiş süre ortalaması 50.05 saniye ile en düşük olan durak aralığı ile (121541-121551) ve durak geçiş süre ortalaması 344.4 saniye ile en yüksek olan durak aralığı (120982-118651) geçiş verileri ile çalışılmıştır.

Yapılan çalışmada Python programlama dilinde elimizdeki verilere dbscan kümeleme algoritması uygulanmıştır. Ayrıca yine Python dilinde anomaly detection (aykırılık tespiti) için kullanılan local outlier factor (aykırı değer tespit) algoritması kullanılmıştır. İki algoritmayı inceleyelim.

1. DBSCAN kümeleme algoritması: DBSCAN, popüler bir yoğunluk tabanlı veri kümeleme algoritmasıdır. Algoritma adımlarını inceleyelim.

- Adım 1: İlk olarak, veri setinden rastgele bir nokta seçilir ve ona yakın tüm komşu noktaları bularak başlar, bulunan noktaların sayısı, küme oluşturmak için gerekli sayıdan büyük veya eşit olmasını dikkate alırız. Ardından, en yakın noktalar ile ilk küme oluşturulur. İlk küme oluşturduktan sonra, tüm noktalar incelenir ve ilgili küme içindeki elemanlara yakın noktalar varsa yeni elemenlar eklenerek küme genişletilir. Bu kümeye eklenecek başka nokta kalmayıncaya kadar devam eder.
- Adım 2: Kümeye atanmamış noktalar arasında herhangi bir nokta çekirdek nokta olarak belirlenir ve yeni bir küme oluşturulur.
- Adım 3: belirlenen kümelere, yinelemeli olarak bağlı tüm noktalar bulunur ve atanır.
- Adım 4: Verilerin içerisindeki tüm katılımsız noktalar bulunur ve bunlar kendilerine en yakın mesafesindeki noktaların olduğu kümeye atanır. Bir nokta mevcuttaki herhangi bir kümeye uymuyorsa, o nokta gürültü noktası olarak tespit edilmiş olur.

2. Local Outlier Factor (aykırı değer tespit) LOF algoritması: LOF algoritması, bir veri kümesinde olağandışı veya anormal değerleri bulmakla ilgilenen bir veri bilimi tekniğidir.

- LOF algoritması, bir noktanın yerel yoğunluğunu, en yakın k komşusunun yerel yoğunluklarıyla karşılaştırarak çalışır.
- Yerel yoğunluğu komşularından daha düşük olan noktalar aykırı değer olarak sınıflandırılacaktır.
- LOF algoritmasında analist kontrolü altındaki birincil parametre, komşu sayısı olan k'dir. Minimum k değerinin en az verideki bilinen en küçük kümenin boyutu kadar olması gerekir, bu sayede bu kümenin etrafında yerel aykırı değerler tespit edilebilir, aksi durumda kümenin noktaları aykırı değer olarak sınıflandırılabilir.

Bu iki algoritma uygulanarak küme sınırları ile ilgili şu kurallar belirlenmiştir;

1. Süre ortalaması en düşük olan (121541-121551) durak aralığı için, mesafe ve azami hız dikkate alınarak bir otobüsün en iyi ortamda katedebileceği en düşük süre 15 saniye olarak belirlenmiştir. 15 saniyeden düşük sefer süreleri varsa, bu veriler küme dışına çıkarılır.
2. Süre ortalaması en yüksek olan (120982-118651) durak aralığı için, mesafe dikkate alınarak bir otobüsün kat edebileceği en yüksek süre 1000 saniye olarak

belirlenmiştir. 1000 saniyeden büyük sefer süreleri varsa, bu veriler küme dışına çıkarılır.

3. İlk iki maddedeki kurallar, Tüm gün tipleri için(H,C ve P) ve tüm saat dilim kümeleri için, uygulanacaktır ve 15 saniyeden küçük ve 1000 saniyeden büyük veriler, geçiş verisi olarak kaydedilmeyecektir. Bu sayede negatif ya da çok yüksek geçiş verileri küme dışında kalarak, elde edilecek ortalama geçiş verisinin daha sağlıklı olması sağlanmıştır.
4. İki durak arası sefer sürelerinin ortalaması alınır, sefer süreleri içerisinde ortalama sefer süresinin iki katından fazla sefer süreleri varsa, bu veriler küme dışına çıkarılır.
5. İki durak arası sefer sürelerinin ortalaması alınır, sefer süreleri içerisinde ortalama sefer süresinin ½ sinden daha az sefer süreleri varsa, bu veriler küme dışına çıkarılır.

Şekil 30 teki verilere veri temizleme işlemi için belirlenen kurallar uygulandıktan sonra şekil 32 deki gibi yeni bir grafik elde edilmiştir.



Şekil 32. 120982-118651 Durak Aralığına Ait 2019 Yılı Ocak Ayı Süre Grafiği (Veri Temizliği Yapılmış Hali)

Veriler temizlenmeden önce gün tipi ve saat dilimine göre kümelendiği için, iki durak arası sefer geçiş süreleri birbirine yakın gerçekleşen süreler olmaktadır. Kirli verilerin

tespiti ve temizlenmesi için belirlenen yöntem hem 6 aylık ortalama hem de 1 aylık ortalama belirlenirken kullanılacaktır.

4.3.2. Arşiv verilerinin ortalamasını belirlemek

Kümelenen ve kirli verileri temizlenen arşiv verileri artık kullanmaya müsait verilerdir, bu çalışmadaki en önemli iddia gürültülü verilerden arındırılmış arşiv verilerinin tahmin algoritmasının içine katıldığında algoritmayı güçlendireceğidir. Arşiv verileri gün ve saat dilimi dikkate alınarak oluşturulur. Örneğin 15:52 deki sorgu için arşiv de gün tipine göre saat 15 diliminde geçiş süresi verilerinden faydalanılır.

4.3.2.1. Altı aylık arşiv verilerinin ortalamasını belirlemek

6 aylık verilerin veri seti içerisinde yer alması çok önemlidir. Merkez bölgelere göre daha az sefer düzenlenen taşra bölgelerde bazen saatlik veri olmaz, aylık veriler ise verinin kalitesi için istenilen sayıda olmayabilir. Bu gibi durumlarda tahmin verisinde ki doğruluk oranına etkisi olumlu olacaktır. Ayrıca belirlenen güzergâhta tüm koşulların rutin gerçekleşmesi, 6 aylık ortalamanın diğerlerine göre daha doğru tahmin verisi üretmesi muhtemeldir(Şekil 33). Bir kesit için altı aylık ortalama veri elde edilmesi için yapılacak işlem şu şekilde olacaktır.

A ={ x1,x2,x3…,xn}

Aort =(x1+x2+x3+…+xn)/n

A = {x: 15< x <1000 ve Aort/2< x <Aort*2)

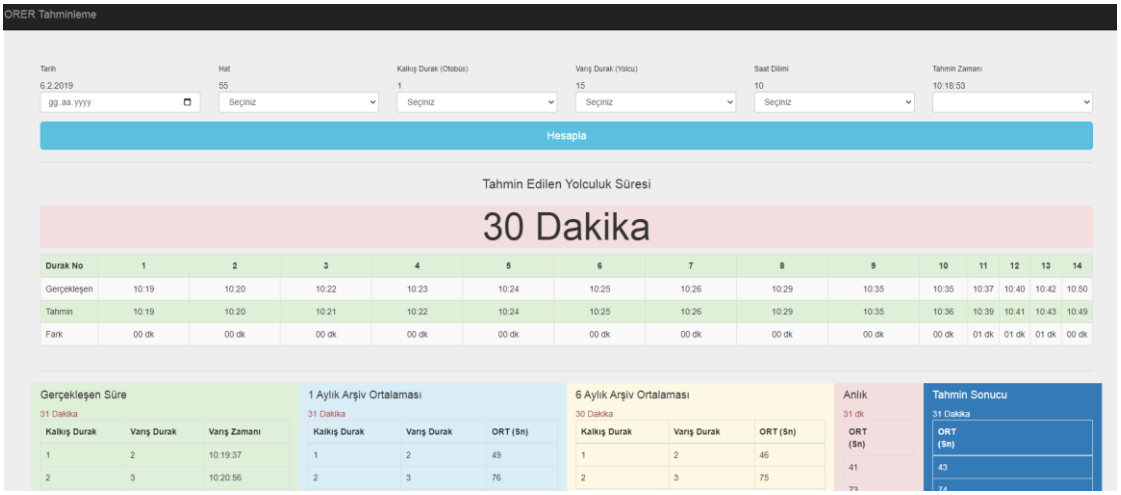
Aort =(x1+x2+x3+…+xn)/n

A : Son 6 aylık geçiş verilerinin kümesi

X : Geçiş süresi verileri (saniye)

Aort : Son 6 aylık geçiş verilerinin ortalaması

Altı aylık geçiş verilerinin olduğu kümenin öncelikle ortalaması alınır, geçiş verileri içerisindeki 15 saniyeden küçük ve 1000 saniyeden büyük geçiş süreleri küme dışına çıkarılır, ortalamanın yarısından küçük olanlar ve ortalamanın iki katından büyük olanlar küme dışına çıkarılarak yeni ortalama bulunur, elde edilen değer 6 aylık ortalama geçiş verisidir.



Şekil 33. OrerTahminleme Uygulamasında Örnek Tahminleme

4.3.2.2. Aylık arşiv verilerinin ortalamasını belirlemek

Arşiv verileri içerisinde son 1 aylık arşiv verilerinin ağırlığı daha fazla olacaktır. Güzergâh üzerinde olabilecek, sefer süresine etki edecek süreçler daha hızlı yakalanacak ve arşivden gelen verinin ağırlığını aylık verinin ortalaması belirlemiş olacaktır. Bir kesit için altı aylık ortalama veri elde edilmesi için yapılacak işlem şu şekilde olacaktır.

B ={ x1,x2,x3…,xn}

Bort =(x1+x2+x3+…+xn)/n

B = {x: 15< x <1000 ve Aort/2< x <Aort*2)

Bort =(x1+x2+x3+…+xn)/n

B: Son 1 aylık geçiş verilerinin kümesi

X: Geçiş süresi verileri (saniye)

Bort: son 1 aylık geçiş verilerinin ortalaması

Son 1 aylık geçiş verilerinin olduğu kümenin öncelikle ortalaması alınır. Geçiş verileri içerisindeki 15 saniyeden küçük ve 1000 saniyeden büyük geçiş süreleri küme dışına çıkarılır. Ortalamanın yarısından küçük olanlar ve ortalamanın iki katından büyük olanlar küme dışına çıkarılarak yeni ortalama bulunur, elde edilen değer son 1 aylık ortalama geçiş verisidir.

4.3.2.3. Arşiv verilerinin ortalamasını belirlemek

Arşiv verileri bazı durumlarda doğrudan tahmin verisi olarak kullanılacaktır, son bir saat içerisinde kesitten ortalama geçiş verisi gelmemesi durumunda, ortalama verisi olarak direk arşiv verisi kullanılacaktır. Genelde bu gibi durumlarla, günün ilk saatlerinde gerçekleştirilen seferlerde ve çok az sefer gerçekleştirilen güzergâhlarda son bir saat içerisinde gerçekleşmiş sefer bilgisi bulunmaması durumunda tahmin verileri için arşiv verileri kullanılacaktır. Şekil 34 kontrol edildiğinde, saatlik ortalamanın hesaplandığı sütun incelendiğinde ilk durak harici ortalama verisi gelmemektedir. Dolayısıyla verinin gelmediği kesitlerdeki ortalama tamamen arşiv verisinden elde edilmiştir. Arşiv ortalama verileri, günün sonunda her gece saat 02.00 itibari hesaplanarak güncel bir tabloda tutulacaktır. Bu sayede tahmin verisi hesaplanırken arşiv verisi hızlıca ilgili tablodan çekilecek ve işlem yoğunluğunun önüne geçilmiş olacaktır. Tablo 12' de, 12.01.2019 tarihli veriler için örnek bir arşiv geçiş süre ortalama tablosu oluşturulmuştur.

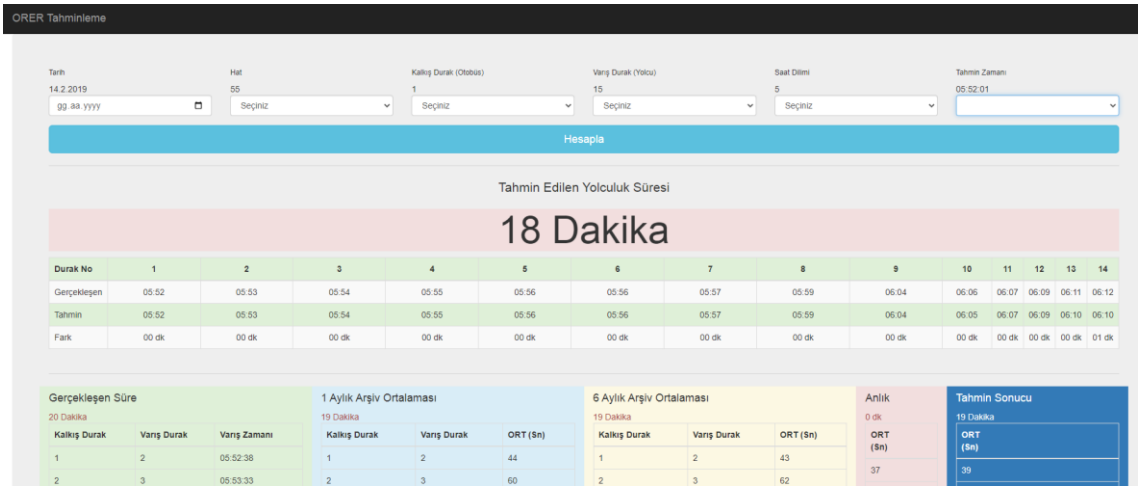
Arşiv ortalama verisi şu şekilde hesaplanacaktır.

Cort = Aort * 0.35 + Bort * 0.65

Cort : Arşiv geçiş verilerinin ortalaması

Bort : Son 1 aylık geçiş verilerinin ortalaması

Aort : Son 6 aylık geçiş verilerinin ortalaması



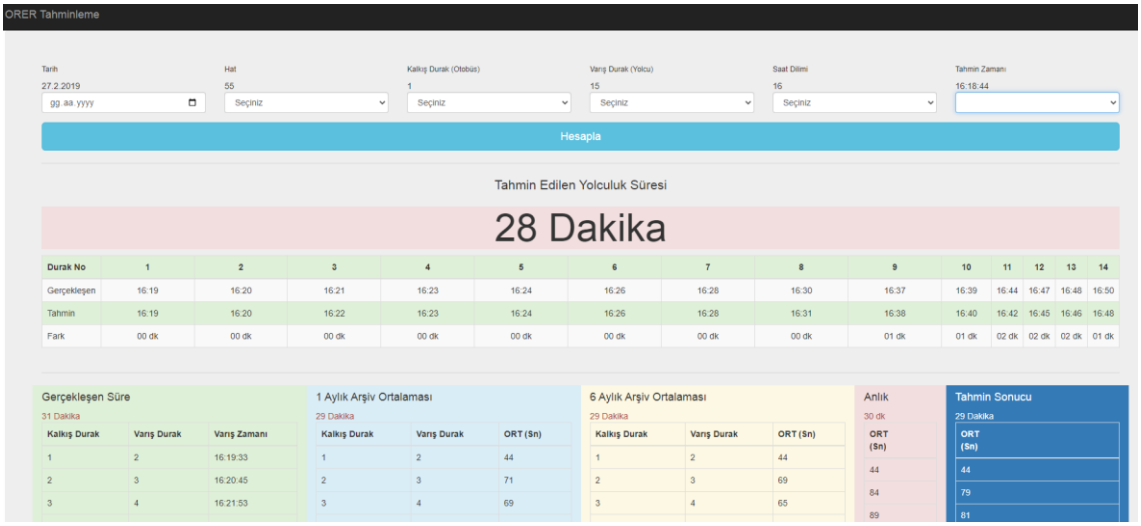
Şekil 34. Orer Tahminleme Programında Günün İlk Seferi İçin Yapılan Tahmin

Tablo 12. 2019.06.01 İtibari İle Oluşan Hafta İçi Ortalama Arşiv Verisi

12.1.2019	SAAT DİLİMİ																	
DURAK ARALIĞI	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
121541 - 121551	40,25	48,16	47,29	49,324	47,35	47,20	48,35	46,462	47,69	45,03	46,76	45,64	44,89	45,37	45,06	45,26	46,25	46,09
121551 - 120791	61,49	64,16	69,70	75,536	74,79	80,24	75,26	79,857	76,87	78,06	77,28	77,79	72,18	75,82	73,45	74,99	66,14	65,45
120791 - 121532	51,03	51,48	55,69	61,842	60,41	61,67	63,27	67,949	64,91	70,84	66,98	69,49	62,37	67,00	63,99	65,94	55,73	55,52
121532 - 120782	43,6	48,53	55,73	61,984	59,08	63,27	67,28	76,606	70,55	76,38	72,59	75,05	64,18	71,25	66,65	69,64	61,82	57,11
120782 - 121011	48,42	50,15	60,06	70,991	62,81	62,46	63,98	70,873	66,39	71,83	68,30	70,59	62,26	67,67	64,15	66,44	52,65	54,99
121011 - 120772	42,59	45,27	52,20	59,114	57,24	60,27	68,11	69,788	68,70	76,72	71,51	74,89	98,60	83,19	93,21	86,70	53,86	59,00
120772 - 121452	53,3	53,72	64,38	72,555	70,92	75,51	86,54	99,271	90,99	127,83	103,88	119,45	136,97	125,58	132,99	128,17	82,25	80,31
121452 - 120982	111,14	118,44	132,03	146,97	143,65	152,17	163,07	175,41	167,39	191,93	175,98	186,34	178,54	183,61	180,32	182,46	150,10	153,60
118651 - 118651	272,02	288,47	304,73	323,09	311,77	323,84	352,69	380,27	362,34	418,23	381,90	405,52	389,91	400,05	393,46	397,75	381,18	330,77
118651 - 118681	52	54,15	58,27	62,469	62,19	62,93	63,90	94,282	74,53	123,76	91,76	112,56	88,35	104,08	93,86	100,50	78,89	58,19
118681 - 118723	91,06	98,26	105,78	123,21	114,39	122,94	149,02	198,97	166,50	238,82	191,81	222,37	213,25	219,18	215,33	217,83	108,52	101,74
118723 - 118761	131,57	135,31	154,76	165,27	152,93	151,36	158,14	165,88	160,85	169,65	163,93	167,65	161,33	165,44	162,77	164,50	140,45	136,07
118761 - 177621	34,45	59,16	68,09	81,841	78,51	82,40	90,70	94,371	91,98	103,43	95,99	100,82	81,52	94,07	85,91	91,21	59,65	63,17
177621 - 118791	25,33	51,87	60,39	71,736	71,38	76,72	80,35	85,417	82,12	93,00	85,93	90,52	80,70	87,09	82,94	85,63	71,37	67,75

4.3.3. Anlık geçiş verilerinin ortalamaya dâhil edilmesi

Tahmin verisini üretirken iki veriden yararlanırız, arşiv verileri ve anlık geçiş verileri. Anlık geçiş verileri diye belirttiğimiz veriler, tahmini yapılacak kesitten son bir saat içerisindeki geçiş süre ortalamasıdır. Anlık geçiş verileri, tahmin verisinin kalitesi için çok önemlidir. Normal şartlarda arşiv verisine benzer bir trafik durumu varsa, anlık veride arşiv verisine benzer çıkacaktır ve iyi bir tahmin verisi elde edilecektir. Ancak o güne özel oluşan bir trafik ya da kaza, yol çalışması vb. herhangi bir sebepten dolayı arşiv verilerine göre farklı bir veri oluşacaksa, bu durumu anlık geçiş verisi ile yakalamış oluruz. Anlık geçiş verisinin ortalama da ağırlığı fazla olduğu için iyi bir tahmin verisi elde edilmiş olacaktır.



Şekil 35. Orertahminleme Uygulamasında Örnek Tahminleme

Anlık geçiş verilerine bakarken sadece bakılan hat'ın geçiş verileri yerine de tüm geçiş verilerine bakmak elimizdeki verinin kalitesini arttıracaktır. Örneğin 55 hattının 1 Ocak 2019 tarihinde 120782 durağından 121011 durağına varışı ile ilgili tahmin yapmak istediğimizde sadece 55 hattının geçiş verisini kullanılırsa, toplamda 4 satırlık veriden yararlanabilirken, aynı saat diliminde aynı durak aralığından geçen tüm hatlar incelendiğinde sonuç olarak 22 satırlık veri gelmektedir.

Sira No	Cikis S	Varis S	Saat	Gün	Varis Zamani	Cikis Zamani	Sure	Cikis D	Varis D	Guzergah	Kapi n
545	6	7	10	H	01.01.2019 10:29:45	01.01.2019 10:28:47	58	120782	121011	99A_G_D0	B-074
672	6	7	10	H	01.01.2019 10:37:44	01.01.2019 10:36:35	69	120782	121011	99A_G_D0	B-067
758	6	7	10	H	01.01.2019 10:45:17	01.01.2019 10:44:29	48	120782	121011	99A_G_D0	B-1758
1045	6	7	10	H	01.01.2019 10:22:39	01.01.2019 10:21:44	55	120782	121011	99A_G_D0	B-073
1279	6	7	10	H	01.01.2019 10:41:58	01.01.2019 10:41:02	56	120782	121011	99A_G_D0	B-068
1399	6	7	10	H	01.01.2019 10:56:51	01.01.2019 10:56:00	51	120782	121011	99A_G_D0	B-1770
2993	6	7	10	H	01.01.2019 10:31:48	01.01.2019 10:31:13	35	120782	121011	55T_G_D546	B-263
3002	6	7	10	H	01.01.2019 10:55:55	01.01.2019 10:55:21	34	120782	121011	55T_G_D546	B-268
3026	6	7	10	H	01.01.2019 10:10:38	01.01.2019 10:09:50	48	120782	121011	55T_G_D546	B-1792
3032	6	7	10	H	01.01.2019 10:45:30	01.01.2019 10:44:46	44	120782	121011	55T_G_D546	B-266
3061	6	7	10	H	01.01.2019 10:01:56	01.01.2019 10:01:19	37	120782	121011	55T_G_D546	B-1783
3076	6	8	10	H	01.01.2019 10:53:38	01.01.2019 10:52:05	93	120782	120772	55T_G_D546	B-267
3539	6	7	10	H	01.01.2019 10:22:25	01.01.2019 10:21:20	65	120782	121011	55_G_D5463	B-1794
3618	6	7	10	H	01.01.2019 10:11:36	01.01.2019 10:10:56	40	120782	121011	55_G_D5463	B-1791
3859	6	7	10	H	01.01.2019 10:15:07	01.01.2019 10:14:29	38	120782	121011	55T_G_D546	B-271
3867	6	7	10	H	01.01.2019 10:21:48	01.01.2019 10:20:42	66	120782	121011	55T_G_D546	B-272
4375	6	7	10	H	01.01.2019 10:37:48	01.01.2019 10:37:03	45	120782	121011	55_G_D5463	B-1795
4716	6	8	10	H	01.01.2019 10:16:10	01.01.2019 10:14:23	107	120782	120772	99A_G_D511	B-072
4723	6	7	10	H	01.01.2019 10:09:19	01.01.2019 10:08:29	50	120782	121011	99A_G_D511	B-071
4778	6	7	10	H	01.01.2019 10:49:20	01.01.2019 10:48:27	53	120782	121011	55_G_D5463	B-101
4950	6	7	10	H	01.01.2019 10:36:13	01.01.2019 10:35:38	35	120782	121011	55T_G_D546	B-264

Şekil 36. Anlık Geçiş Verilerinin Ortalamasında Tüm Hatların Kullanılması

Bir kesit için anlık ortalama geçiş verisi elde edilmesi için yapılacak işlem şu şekilde olacaktır.

D ={ x1,x2,x3…,xn}

Dort =(x1+x2+x3+…+xn)/n

D = {x: 15< x <1000 ve Aort/2< x <Aort*2)

Dort =(x1+x2+x3+…+xn)/n

E = Dort * 0.65 + Cort * 0.35

D: Son 1 saatlik geçiş verilerinin kümesi

X: Geçiş süresi verileri (saniye)

Dort: son 1 saatlik geçiş verilerinin ortalaması

Cort : Arşiv geçiş verilerinin ortalaması

E : Elde edilen tahmin verisi

4.3.4. Yolcu ile veriyi kontrollü paylaşmak

Yolcu ile paylaşılan tahmin verisinin -1 / +3 dakika aralığında olması yolcu tarafından makul karşılanmaktadır. Gerek İETT ye gelen şikâyetler, gerekse İBB Beyaz Masaya gelen otobüs gecikmesi ile ilgili şikâyetler incelendiğinde otobüsün beklenenden 1 dakika önce gelmesi ya da 3 dakika ya kadar geç kalması yolcu tarafından sorun edilmiyor. Ancak çok fazla geç kalma ya da otobüsün çok erken geçmesi yolcuların şikâyetçi olmasına neden oluyor. Bu çalışmada yapılan en iyi yöntemlerden biriside yolcuya tahmin verisi sunulurken, eğer beklenen otobüs 5 durak ya da 5 duraktan daha uzakta ise tahmin verisinin 1 dakika daha düşürülerek yolcu ile paylaşılmasıdır(Şekil 37). Bu sayede yolcuların daha temkinli davranmasına neden olacaktır. Bir kesit için elde edilen tahmin verisini iyileştirme işlemi şu şekilde olacaktır.

F = {y1,y2,y3,…,yn}

G = F >= 5 ise E-1

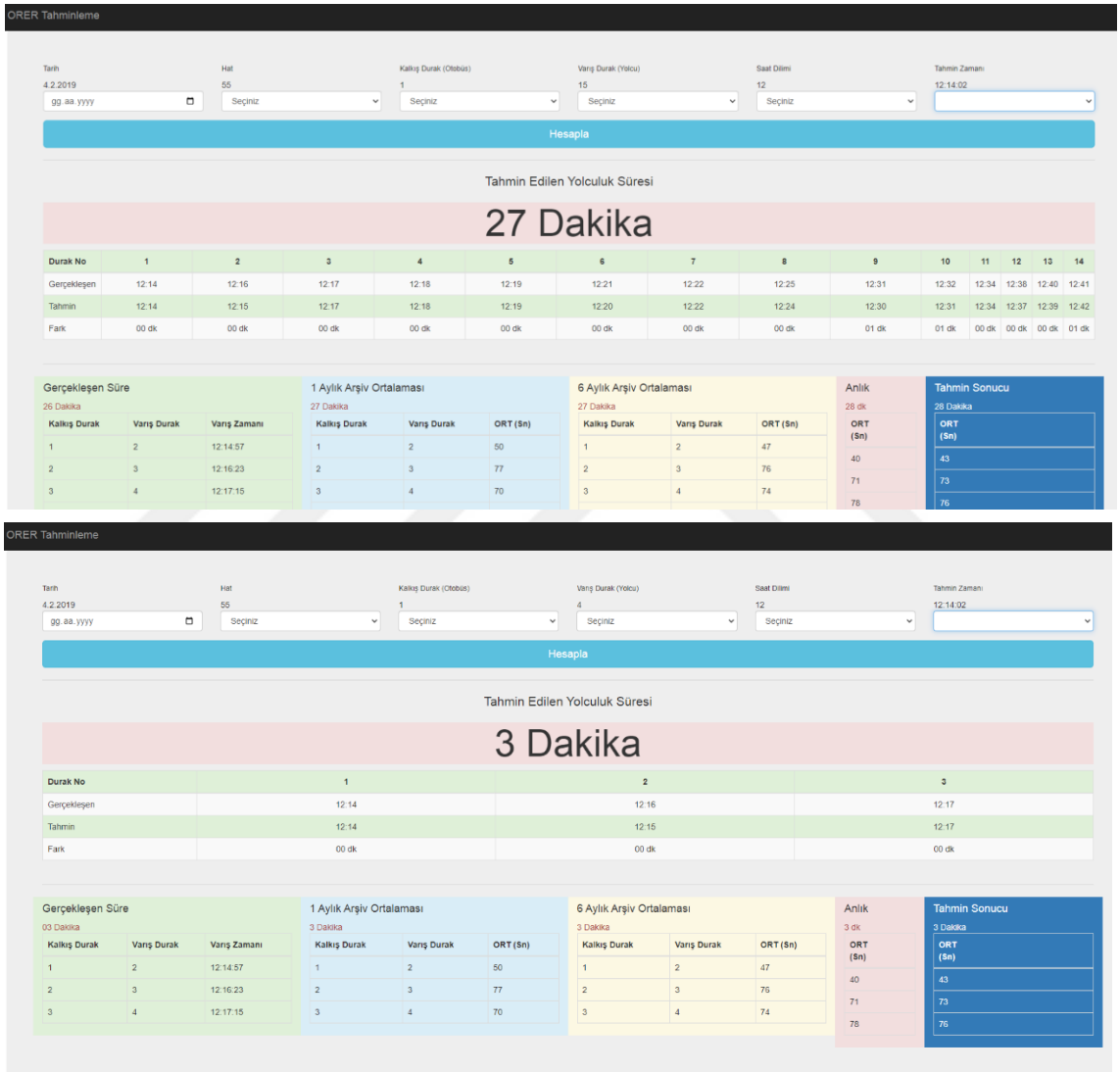
G = F < 5 ise E

F : Tahminleme yapılacak olan durak sayısı

y : Tahminleme yapılacak olan duraklar

E : Ortalama Tahmin Verisi

G : Elde edilen tahmin verisi



Şekil 37. 5 Duraktan Az Ve 5 Duraktan Fazla Tahminleme İçin Tahmin Sonuçları

Şekil 37’ teki resimler incelendiğinde 5 duraktan fazla mesafe için ve 5 duraktan az mesafe için yapılan tahminleme görünmektedir. ilk resimde 1. ve 15. durak aralığı için tahminleme yapılmıştır, ortalama değer 28 dakika 6 saniye çıkmıştır, 5 duraktan fazla olduğu için tahmin verisi 27 dakika olarak üretilmiştir. İkinci resimde 6. ve 10. durak

aralığı için tahminleme yapılmıştır, ortalama değer 10 dakika 45 saniye çıkmıştır, 5 duraktan az olduğu için tahmin verisi 10 dakika olarak üretilmiştir.

4.4. Uygulamanın Çalışması ve Ekran Görüntüleri

Bir durumun tahminlemesi problemi üzerine birçok uygulama geliştirilmiştir. Bu tarz uygulamalar geliştirilirken Microsoft .Net, Java, Python vb tarzda platformlar uygulamanın geliştirme ortamları olarak kullanılmıştır. Bu çalışma da .Net Framework programlama dili, OrerTahminleme uygulamasının geliştirildiği platformdur. OrerTahminleme uygulaması web tabanlı bir uygulamadır. Yapılan çalışmanın sonuçlarını görebilmek için geliştirilmiştir.

OrerTahminleme uygulamasın da önerilen çalışmanın. Bir testi niteliğinde olmak üzere 55 hattına ait Mayıs 2018 ile Mart 2019 aralığındaki 1.500.000 geçiş verisi, veri kaynağı olarak kullanılmıştır. Uygulama da istenilen verileri tek tek incelendiğinde;

Tarih: Tahminlemenin yapılacağı tarih verisidir, yani yolcunun durakta beklediği gün olarak düşünebiliriz.

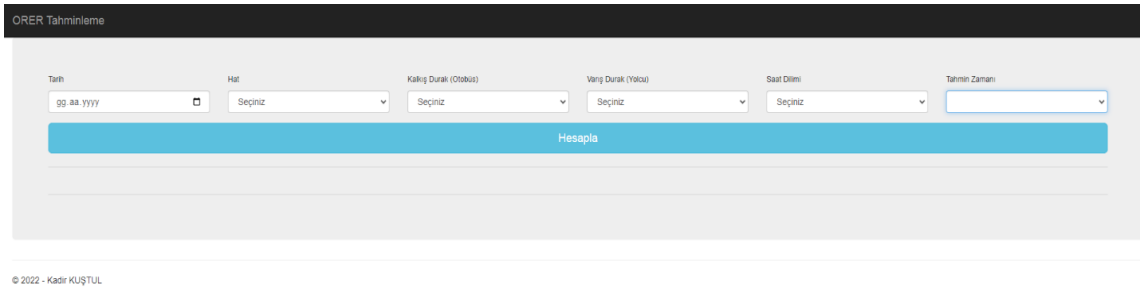
Hat: Yolcunun beklediği hat, yapılan çalışma 55 hattı ile ilgili olduğu için hat verisi olarak 55 seçilmesi gerekmektedir.

Varış Durak: Yolcunun beklediği duraktır.

Kalkış Durak: Otobüsün hangi durakta olduğu bilgisidir, Kalkış ve Varış durak verilerinin bu şekilde istenme nedeni, durak aralıklarını istediğimiz gibi(bazen 2 durak, bazen 14 durak) ayarlayarak, uygulamanın çalışma performansını ölçmek yani tahmin verisinin kalitesini ölçmek için önemlidir.

Saat: Tarih, Hat, Kalkış durak ve Saat Dilimi verilerine göre listelenir. Girilen tarih, saat ve Hat verisine göre, seçilen kalkış duraktan gerçekleşmiş geçiş verilerini listeler. Diğer veriler hazır şablondan çekilirken, bu veri diğer verilere göre otomatik doldurulacaktır.

Saat Dilimi: Seçilen kalkış durak tan gerçekleşmiş geçiş verilerini seçilen saat dilimine göre listeler.



Şekil 38. Orertahminleme Uygulaması Veri Giriş Alanı

Programa dair kod yapısı ve ekran görüntüleri aşağıdaki gibidir.

```
0 başvuru
public ActionResult hesapla(DateTime girilentarih, string hat, int kalkisDurak, int varisDurak, int saat, string saatdetay)
{
    #region ViewInfo
    ViewData["tarih"] = girilentarih.ToShortDateString();
    ViewData["hat"] = hat;
    ViewData["kalkisdurak"] = kalkisDurak;
    ViewData["varisdurak"] = varisDurak;
    ViewData["saat"] = saat;
    ViewData["saatdetay"] = saatdetay;
    #endregion


    var gun = girilentarih.DayOfWeek;
    #region tanimlar
    var birsonrakidurak = kalkisDurak + 1;

    switch (gun)...

    DateTime eksi6Ay = girilentarih.AddMonths(-6);
    List<ortalamaModel> altiAylikList = new List<ortalamaModel>();
    DateTime eksi1Ay = girilentarih.AddMonths(-1);
    List<ortalamaModel> aylikList = new List<ortalamaModel>();
    List<ortalamaModel> eksiSaatList = new List<ortalamaModel>();
    List<ortalamaModel> TahminList = new List<ortalamaModel>();
    #endregion
    switch (gun)
    {
        case DayOfWeek.Sunday:
            // Gerçekleşen Veri Listesi
            var gorevid7 = db.Pazar.First(x => x.CikisZamani == saatdetay && x.Hat == hat && x.Saat == saat && x.Tarih == girilentarih).Gorevid;
            if (gorevid7 != null)...
```

Şekil 39. Tarih, Hat, Kalkış Durak, Varış Durak, Saat Dilimi verilerinin girildiği alan, Saat alanı diğer verilere göre otomatik doluyor

Tablo 13. Gerçekleşen Süre Tablosu

Gerçekleşen Süre

26 Dakika

Kalkış Durak	Varış Durak	Varış Zamanı
1	2	12:14:57
2	3	12:16:23
3	4	12:17:15
4	5	12:18:34
5	6	12:19:43
6	7	12:21:22
7	8	12:22:48
8	9	12:25:48
9	10	12:31:31
10	11	12:32:49
11	12	12:34:55
12	13	12:38:21
13	14	12:40:14
14	15	12:41:00

```
// Gerçekleşen Veri Listesi
var gorevid7 = db.Pazar.First(x => x.CikisZamani == saatdetay && x.Hat == hat && x.Saat == saat && x.Tarih == girilentarih).Gorevid;
if (gorevid7 != null)
{
    var liste = db.Pazar.Where(x => x.Gorevid == gorevid7 && x.Hat == hat && x.Tarih == girilentarih && x.CikisNo >= kalkisDurak && x.VarisNo <= varisDurak).OrderBy(x => x.CikisNo).ToList();
    //Bu aşamada yukarıda çekilen liste içerisinde işlem yapıyoruz..
    var cikissaat = liste.Min(x => x.CikisZamani);
    var varissaat = liste.Max(x => x.VarisZamani);
    ViewBag.liste = liste.OrderBy(x => x.CikisNo).ToList();
    TimeSpan timecikissaat = TimeSpan.Parse(varissaat.ToString()) - TimeSpan.Parse(cikissaat.ToString());
    ViewBag.cikissaat = timecikissaat;
}
```

Şekil 40. Tahmin Edilmesi İstenen Aralığın Ne Kadar Sürede Geçildiği Bilgisi Bu Alanda Verilmiştir. Son Kesitin Süresinden Saat Kısmında Seçilen Süre Çıkarıldığında Gerçekleşen Süre Ortaya Çıkar.

Tablo 14. 1 Aylık Ortalama Tahmin Veri Tablosu

1 Aylık Arşiv Ortalaması

27 Dakika

Kalkış Durak	Varış Durak	ORT (Sn)
1	2	50
2	3	77
3	4	70
4	5	74
5	6	70
6	7	72
7	8	99
8	9	172
9	10	354
10	11	78
11	12	153
12	13	164
13	14	106

```
// 1 Aylık Ortalama
if (varisDurak != 0)
{
    int dongu = kalkisDurak;
    int deger = varisDurak;
    int rownumber = 2;
    int bironcekisira = kalkisDurak;
    int birsonrakisira = kalkisDurak + 1;
    while (dongu <= deger)
    {
        if (dongu == deger)
        {
            var son = (from a in db.Pazar
                       where a.Saat == saat
                       where a.CikisNo == bironcekisira - 1
                       where a.VarisNo == birsonrakisira
                       where a.Tarih >= eksi1Ay && a.Tarih <= girilentarih
                       select a.Sure).Average(x => (double?)x.Value) ?? 0;
            var snc = Math.Round(son, 2);

            var temizVeriOrt = (from a in aylikList
                                select a.sure).Sum();
            var snc3 = Math.Round(Convert.ToDouble(temizVeriOrt / 60), 2);
            ViewBag.Ort1AylikH4 = snc3;
            ViewBag.Ort1Aylik = aylikList.ToList();

            break;
        }
        else
        {
            var son = (from a in db.Pazar
                       where a.Saat == saat
                       where a.CikisNo == bironcekisira
                       where a.VarisNo == birsonrakisira
                       where a.Tarih >= eksi1Ay && a.Tarih <= girilentarih
                       select a.Sure).Average(x => (double?)x.Value) ?? 0;

            var snc = Math.Round(son, 2);
            aylikList.Add(new ortalamaModel { id = rownumber, cikisdurak = bironcekisira, varisdurak = birsonrakisira, sure = snc });

            rownumber++;
            bironcekisira++;
            birsonrakisira++;
            dongu++;
        }
```

Şekil 41. 1 Aylık Tahmin Verisinin Hesaplandığı Alandır, Arşiv Verisi İçerisinde %65 Oranında Ağırlığı Vardır, Genel Tahmin Verisine %23 Gibi Bir Etkisi Vardır.

Tablo 15. 6 Aylık Ortalama Tahmin Veri Tablosu

6 Aylık Arşiv Ortalaması

27 Dakika

Kalkış Durak	Varış Durak	ORT (Sn)
1	2	47
2	3	76
3	4	74
4	5	74
5	6	66
6	7	70
7	8	100
8	9	173
9	10	361
10	11	70
11	12	162
12	13	160
13	14	104

```
// 6 Aylık Ortalama
if (varisDurak != 0)
{
    int dongu = kalkisDurak;
    int deger = varisDurak;
    int rownumber = 2;
    int bironcekisira = kalkisDurak;
    int birsonrakisira = kalkisDurak + 1;
    while (dongu <= deger)
    {
        if (dongu == deger)
        {
            var son = (from a in db.Pazar
                       where a.Saat == saat
                       where a.CikisNo == bironcekisira - 1
                       where a.VarisNo == birsonrakisira
                       where a.Tarih >= eksi6Ay && a.Tarih <= girilentarih
                       select a.Sure).Average(x => (double?)x.Value) ?? 0;
            var snc = Math.Round(son, 2);
            var temizVeriOrt = (from a in altiAylikList
                                select a.sure).Sum();

            var snc5 = Math.Round(Convert.ToDouble(temizVeriOrt / 60), 2);
            ViewBag.Ort6AylikH4 = snc5;
            ViewBag.Ort6Aylik = altiAylikList.ToList();
            break;
        }
        else
        {
            var son = (from a in db.Pazar
                       where a.Saat == saat
                       where a.CikisNo == bironcekisira
                       where a.VarisNo == birsonrakisira
                       where a.Tarih >= eksi6Ay && a.Tarih <= girilentarih
                       select a.Sure).Average(x => (double?)x.Value) ?? 0;

            var snc = Math.Round(son, 2);
            altiAylikList.Add(new ortalamaModel { id = rownumber, cikisdurak = bironcekisira, varisdurak = birsonrakisira, sure = snc });
            rownumber++;
            bironcekisira++;
            birsonrakisira++;
            dongu++;
        }
```

Şekil 42. 6 Aylık Tahmin Verisinin Hesaplandığı Alandır, Arşiv Verisi İçerisinde %35 Oranında Ağırlığı Vardır, Genel Tahmin Verisine %12 Gibi Bir Etkisi Vardır.

Tablo 16. Anlık Ortalama

Anlık

28 dk

ORT (Sn)
40
71
78
63
59
64
91
165
299
80
238
164
107
183

```
// 1 Saatlik Ortalama
if (varisDurak != 0)
{

    int dongu = kalkisDurak;
    int deger = varisDurak;
    int rownumber = 2;
    int bironcekisira = kalkisDurak;
    int birsonrakisira = kalkisDurak + 1;
    while (dongu <= deger)
    {
        string yenitarih = girilentarih.ToString("yyyy-MM-dd");
        if (dongu == deger)
        {
            var son = db.Pazar.SqlQuery($"Select * from Pazar where  Tarih ='{yenitarih}'  And  CikisNo='{bironcekisira - 1}' And VarisNo='{birsonrakisira}' And hat ='{hat}' And CikisZamani BETWE
            var ortAl = (from a in son
                         select a.Sure).Average(x => (double?)x.Value) ?? 0;

            var VeriOrt = (from a in eksiSaatList
                           select a.sure).Sum();
            var snc6 = Math.Round(Convert.ToDouble(VeriOrt / 60), 2);
            ViewBag.SaatlikVerih4 = snc6;
            ViewBag.SaatlikVeri = eksiSaatList;
            break;
        }
        else
        {
            var son = db.Pazar.SqlQuery($"Select * from Pazar where  Tarih ='{yenitarih}'  And  CikisNo='{bironcekisira}' And VarisNo='{birsonrakisira}' And hat ='{hat}' And CikisZamani BETWEEN D

            var ortAl = (from a in son
                         select a.Sure).Average(x => (double?)x.Value) ?? 0;
            var snc = Math.Round(ortAl, 2);
            var cd = (from x in altiAylikList
                      where x.id == rownumber
                      select (double)x.sure).FirstOrDefault();
            var s1 = Convert.ToDouble(cd);
            if (snc > s1 * 2 || snc < s1 / 2)
            {
                eksiSaatList.Add(new ortalamaModel { id = rownumber, cikisdurak = bironcekisira, varisdurak = birsonrakisira, sure = 0 });
            }
            else
            {
                eksiSaatList.Add(new ortalamaModel { id = rownumber, cikisdurak = bironcekisira, varisdurak = birsonrakisira, sure = snc });

            }
            ////////////////////////////////////////////////////////////////
            rownumber++;
            bironcekisira++;
            birsonrakisira++;
            dongu++;
        }
```

Şekil 43. Saatlik Tahmin Verisinin Hesaplandığı Alandır, Genel Tahmin Verisine %65 Gibi Bir Etkisi Vardır.

Tablo 17. Tahmin Sonucu

Tahmin Sonucu

28 Dakika

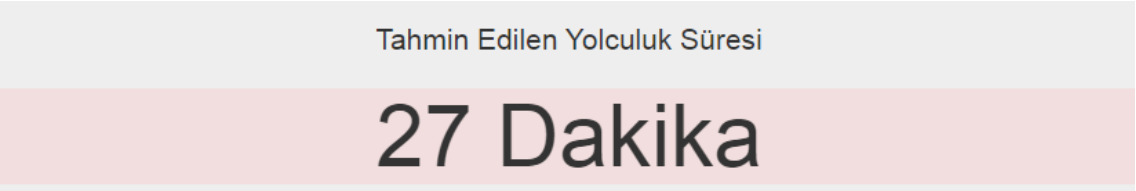
ORT (Sn)
43
73
76
67
62
66
94
168
319
78
209
163
106
159

```
//Tahminleme
if (eksiSaatList != null)
{
    // Bu alanda tahmin Hesabını yapıyoruz;
    int dongu2 = kalkisDurak;
    int deger2 = varisDurak;
    int rownumber2 = 2;
    int bironcekisira2 = kalkisDurak;
    int birsonrakisira2 = kalkisDurak + 1;
    while (dongu2 <= deger2)
    {
        if (dongu2 != deger2)
        {
            var A = (from x in altiAylikList where (x.cikisdurak == dongu2) select (double)x.sure).FirstOrDefault();     //  65%  alınacak
            var B = (from x in aylikList where (x.cikisdurak == dongu2) select (double)x.sure).FirstOrDefault();         //  35%  alınacak
            var C = eksiSaatList.Where(x => x.cikisdurak == dongu2).Select(z => (double)z.sure).FirstOrDefault(); // 0 ise formüle girmeyecek değilse  diğerlerinin 35 i 65 i baz alınacak;
            if (C > 0)
            {
                var k = (Convert.ToDouble(C) * 65 / 100);
                var l = (Convert.ToDouble(A) * 35 / 100 + (Convert.ToDouble(B) * 65 / 100));
                TahminList.Add(new ortalamaModel { id = rownumber2, cikisdurak = bironcekisira2, varisdurak = birsonrakisira2, sure = k + l * 35 / 100 });
            }
            else
            {
                TahminList.Add(new ortalamaModel { id = rownumber2, cikisdurak = bironcekisira2, varisdurak = birsonrakisira2, sure = (Convert.ToDouble(B) * 0.35) + (Convert.ToDouble(A) * 0.65) });
            }
        }

        rownumber2++;
        bironcekisira2++;
        birsonrakisira2++;
        dongu2++;
    }
    var VeriOrt2 = (from a in TahminList
                    select a.sure).Sum();

    var snc6 = Math.Round(Convert.ToDouble(VeriOrt2 / 60), 2);
    ViewBag.OrtalamaVerih4 = snc6;
    ViewBag.OrtalamaVeri = TahminList;
    ViewBag.TahminSonuc = snc6;
}
```

Şekil 44. Tahmin Verilerinin Hesaplandığı Alandır.



Şekil 45. Yolcuya Bildirilen Tahmin Süresi

```
var VeriOrt2 = (from a in TahminList
                select a.sure).Sum();

var snc6 = Convert.ToDouble(VeriOrt2 / 60);

var snc7 = snc6 - (Math.Floor(snc6));
snc7 = snc7 * 60 / 100;
snc6 = (Math.Floor(snc6)) + Math.Round(snc7, 2);

ViewBag.OrtalamaVerih4 = snc6;
ViewBag.OrtalamaVeri = TahminList;
if ((varisDurak - kalkisDurak) > 4)
{
    ViewBag.TahminSonuc = (Math.Floor(snc6) - 1);
}
else
{
    ViewBag.TahminSonuc = (Math.Floor(snc6));
}
```

Şekil 46. Tahmin Verisi Yolcuya Bildirilirken, Algoritmaya Göre Tahmin Edilen Kesit 5 Duraktan Büyük İse Yolcu İle 1 Dakika Eksik Paylaşılmaktadır.

Tablo 18. Gerçekleşen Yolculuk Süresi İle Tahmin Edilen Yolculuk Süresi Arasındaki Fark Tablosu

Durak No	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Gerçekleşen	12:14	12:16	12:17	12:18	12:19	12:21	12:22	12:25	12:31	12:32	12:34	12:38	12:40	12:41
Tahmin	12:14	12:15	12:17	12:18	12:19	12:20	12:22	12:24	12:30	12:31	12:34	12:37	12:39	12:42
Fark	00 dk	00 dk	00 dk	00 dk	00 dk	00 dk	00 dk	00 dk	01 dk	01 dk	00 dk	00 dk	00 dk	01 dk

```
case DayOfWeek.Sunday:
    var list = from x in db.Pazar
               where x.Tarih == girilentarih & x.Hat == hat & x.CikisNo == kalkisDurak & x.Saat == saat
               select x.CikisZamani;
    return Json(list, JsonRequestBehavior.AllowGet);
case DayOfWeek.Monday:
    var list1 = from x in db.Haftaici
                where x.Tarih == girilentarih & x.Hat == hat & x.CikisNo == kalkisDurak & x.Saat == saat
                select x.CikisZamani;
    return Json(list1, JsonRequestBehavior.AllowGet);
case DayOfWeek.Tuesday:
    var list2 = from x in db.Haftaici
                where x.Tarih == girilentarih & x.Hat == hat & x.CikisNo == kalkisDurak & x.Saat == saat
                select x.CikisZamani;
    return Json(list2, JsonRequestBehavior.AllowGet);
case DayOfWeek.Wednesday:
    var list3 = from x in db.Haftaici
                where x.Tarih == girilentarih & x.Hat == hat & x.CikisNo == kalkisDurak & x.Saat == saat
                select x.CikisZamani;
    return Json(list3, JsonRequestBehavior.AllowGet);
case DayOfWeek.Thursday:
    var list4 = from x in db.Haftaici
                where x.Tarih == girilentarih & x.Hat == hat & x.CikisNo == kalkisDurak & x.Saat == saat
                select x.CikisZamani;
    return Json(list4, JsonRequestBehavior.AllowGet);
case DayOfWeek.Friday:
    var list5 = from x in db.Haftaici
                where x.Tarih == girilentarih & x.Hat == hat & x.CikisNo == kalkisDurak & x.Saat == saat
                select x.CikisZamani;
    return Json(list5, JsonRequestBehavior.AllowGet);
case DayOfWeek.Saturday:
    var list6 = from x in db.Cumartesi
                where x.Tarih == girilentarih & x.Hat == hat & x.CikisNo == kalkisDurak & x.Saat == saat
```

Şekil 47. Fark Tablosuna Ait Kodlar

4.5. Deneysel Sonuçlar

OrerTahminleme uygulaması üzerinde test amaçlı olarak 16 gün için her gün farklı zaman dilimleri olmak üzere 100 tahmin verisi üretilerek toplamda 1600 adet tahmin verisi elde edilmiştir. Sonuçlar gayet başarılı olmuştur, bu sonuçların gösterimi için günlük ortalama veriler alınarak grafik gösterimi sağlanmıştır(Tablo 18). Özellikle yolcu ile paylaşılmadan önce tahmin verisi üzerinde yapılan dönüşüm sayesinde çok olumlu sonuçlar elde edilmiştir. Bu test verileri tahmin verilerinin, doğru tahmin olarak kabul edilebildiği aralık -1 dakika ile +3 dakika olarak kabul edilmiştir. Gerçekleşen veri, çıkan tahmin verisi, yolcu ile paylaşılan tahmin verisi verileri üzerinde durulmuş ve aşağıdaki tablodaki sonuçlar elde edilmiştir.

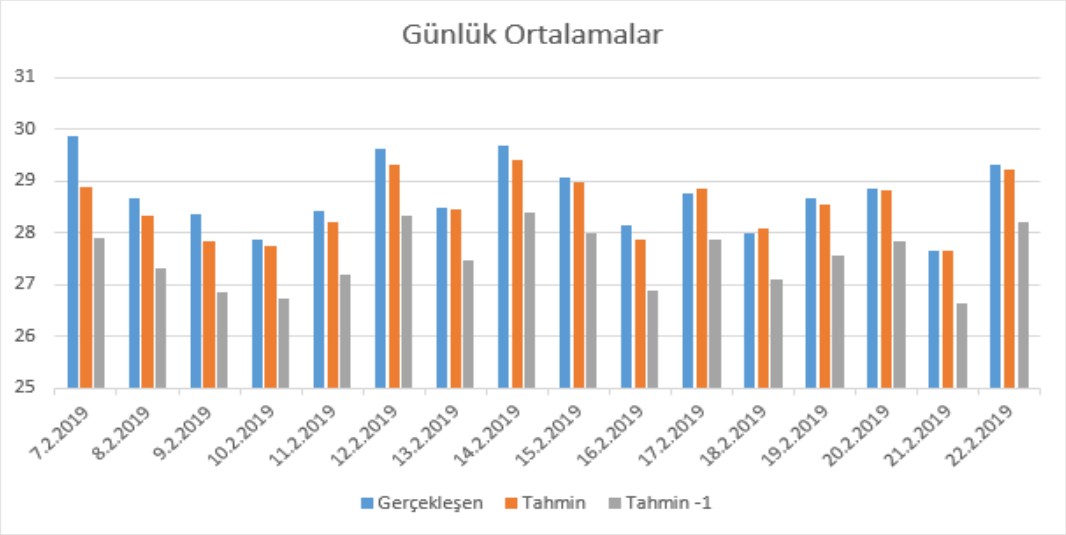
Tablo 19. Yapılan Tahmin Çalışmasından Bir Bölüm (Tahmin Sütunu: Tahmin Verisi, Tahmin -1 Sütunu: Yolcu İle Paylaşılan Tahmin Verisi)

Tarih	Kalkış Durak(Otobüs)	Varış Durak(Yolcu)	Gerçekleşen	Tahmin	Tahmin -1
7.2.2019	1	15	28	30	29
7.2.2019	1	15	26	28	27
7.2.2019	1	15	30	29	28
7.2.2019	1	15	32	29	28
7.2.2019	1	15	36	35	34
7.2.2019	1	15	35	33	32
7.2.2019	1	15	30	28	27
7.2.2019	1	15	21	19	18
7.2.2019	1	15	34	36	35
7.2.2019	1	15	34	35	34
7.2.2019	1	15	30	28	27
7.2.2019	1	15	25	24	23
7.2.2019	1	15	36	33	32
7.2.2019	1	15	30	25	24
7.2.2019	1	15	26	28	27
7.2.2019	1	15	35	35	34
7.2.2019	1	15	21	21	20
7.2.2019	1	15	21	23	22
7.2.2019	1	15	29	30	29
7.2.2019	1	15	36	35	34
7.2.2019	1	15	27	26	25
7.2.2019	1	15	23	20	19
7.2.2019	1	15	31	32	31
7.2.2019	1	15	25	23	22
7.2.2019	1	15	36	36	35
7.2.2019	1	15	24	34	33
7.2.2019	1	15	27	26	25
7.2.2019	1	15	25	25	24
7.2.2019	1	15	31	29	28
7.2.2019	1	15	24	23	22
7.2.2019	1	15	34	33	32
7.2.2019	1	15	26	28	27
7.2.2019	1	15	35	37	36
7.2.2019	1	15	32	32	31
7.2.2019	1	15	29	27	26

Tahmin verilerini incelendiğinde, 1600 adet tahmin verisinin 141 tanesinin hatalı olduğu gözlemlenmiştir, tahmin edilecek kesit 5 duraktan fazla ise 1 dakika eksiltilerek tahmin paylaşılır kuralını uygulayarak üretilen yeni tahmin verisi ile hatalı veri sayısı 129 a düşürülmüştür. Bu kuralı uygulamaktaki amaç yolcuya bildirilen tahmin verisinin gerçekleşen süreden az olmamasıdır. Bu kural sayesinde, gerçekleşen süreden daha büyük hesaplanan 68 adet hatalı tahmin verisi, kabul edilebilir aralığa çekilmiştir.

Tablo 20. 16 Günlük 1600 Adet Tahmin Verilerinin Günlük Ortalama Gösterimi

Tarih	Gerçekleşen	Tahmin	Tahmin -1
7.2.2019	29,87	28,9	27,9
8.2.2019	28,68	28,33	27,33
9.2.2019	28,36	27,86	26,86
10.2.2019	27,88	27,75	26,75
11.2.2019	28,43	28,21	27,21
12.2.2019	29,62	29,33	28,33
13.2.2019	28,5	28,47	27,47
14.2.2019	29,7	29,4	28,4
15.2.2019	29,08	28,99	27,99
16.2.2019	28,15	27,88	26,88
17.2.2019	28,76	28,87	27,87
18.2.2019	28	28,1	27,1
19.2.2019	28,69	28,56	27,56
20.2.2019	28,86	28,84	27,84
21.2.2019	27,67	27,65	26,65
22.2.2019	29,33	29,22	28,22



Şekil 48. 16 Günlük 1600 Adet Tahmin Verilerinin Günlük Ortalamasının Grafiksel Gösterimi

Üretilen tahmin verileri, günlük ortalamaları alınarak Tablo 20 de gösterilmiştir. Tablo ve tabloya bağlı grafik incelendiğinde tahmin verileri, Tahmin -1 verilerine göre daha iyi performans göstermiş gibi görünmektedir. Ancak sonuçlara baktığımızda, tahmin verilerindeki hata oranı %10, Tahmin -1 deki hata oranı ise %8.9 dur.

4.5.1. Doğruluk oranı karşılaştırması

Yapılan çalışmanın sonuçlarını görebilmek için, önerilen yöntem ile mevcutta kullanılan algoritmanın sonuçları karşılaştırılmıştır. Tahmin sonuçlarını karşılaştırırken üç adet performans göstergesinin uygun olduğu değerlendirilmiştir. Bunlar; MAE, MAPE ve RMSE olacaktır. Bu üçü sayesinde, iki algoritma sonuçları karşılaştırılarak, tahmin edilen süre ile gerçekleşen süre arasındaki farklardan kaynaklı hata oranları değerlendirilmiştir. Ayrıca tahmin verilerinin -1 dakika ile +3 kuralına uygunluk durumunu da değerlendirilmiştir.

MAE (Mean Absolute Error): Ortalama mutlak hata ölçüm işlemini yaparken gerçekleşen değer ile tahmin arasındaki farkın mutlak değerini alırız, bu sayede negatif değerlerden kurtulmuş oluruz, çıkan değerlerin ortalamasını alırız ve MAE Ortalama Mutlak Hata değerini buluruz.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} | k_i |$$

MAPE (Mean Absolute Percentage Error): Ortalama mutlak yüzde hata ölçüm işlemini yaparken gerçekleşen değer ile tahmin arasındaki farkın mutlak değerini alırız, bu sayede negatif değerlerden kurtulmuş oluruz, çıkan değerlerin ortalamasını alırız ve MAE Ortalama Mutlak Hata değerini buluruz.

$$MAPE = \frac{100}{N}\sum_{i}^{N} \frac{|k_i|}{|A_i|}$$

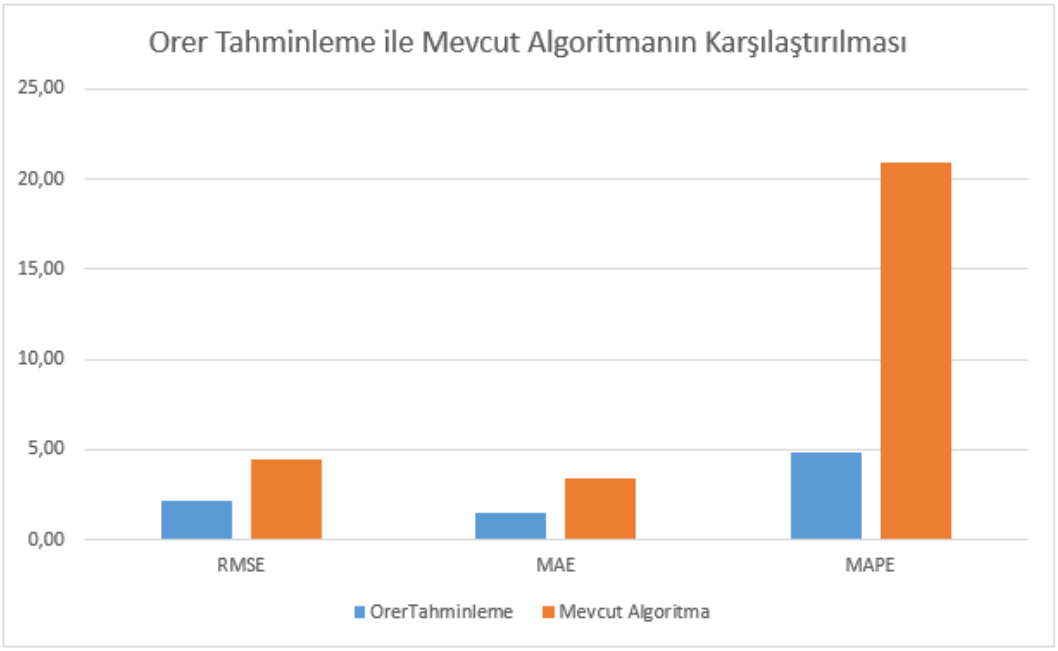
RMSE (Root Mean Squared Error): Ortalama karekök hata ölçüm işlemini yaparken gerçekleşen değer ile tahmin arasındaki farkın karesini alırız, farkların ortalamasını aldıktan sonra karekökünü alırız ve RMSE Ortalama Karekök Hata değerini buluruz.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} k_i^2}{N}}$$

MAE, MAPE ve RMSE değerleri üzerinden mevcut algoritma ile OrerTahminleme algoritmasının karşılaştırmalarına bakıldığında;

Tablo 21. Hata oranları

	RMSE	MAE	MAPE
OrerTahminleme	2,19	1,52	4,82
Mevcut Algoritma	4,45	3,44	20,97

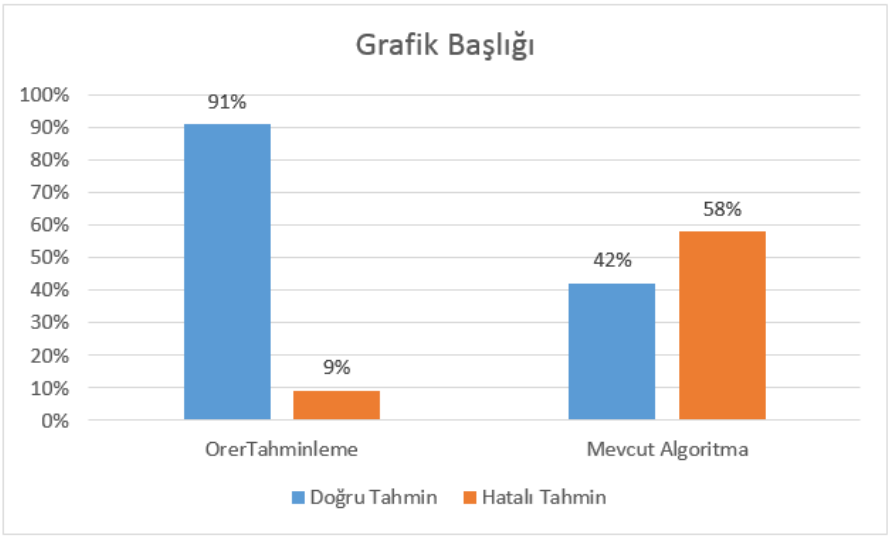


Şekil 49. Hata Oranları Grafiksel Gösterimi

Tablo 21 de hata oranlarını kontrol ettiğimizde, RMSE hata oranı mevcut algoritmada 4.45 çıkarken OrerTahminleme de 2.19 çıkmaktadır. MAE hata rakamlarına baktığımızda mevcut algoritma 3.44 çıkarken OrerTahminleme 1.52 çıkmaktadır. Son olarak MAPE hata oranlarını incelediğimizde mevcut algoritmada 20.97 çıkarken OrerTahminleme 4.82 çıkmaktadır. Hata rakamları kontrol edildiğinde OrerTahminleme ile mevcut algoritma arasındaki fark belirgin bir şe kilde görünmektedir.

Tablo 22. Hata Oranları

	OrerTahminleme	Mevcut Algoritma
Doğru Tahmin	91%	42%
Hatalı Tahmin	9%	58%



Şekil 50. Hata Oranları Grafiksel Gösterimi

Mevcut algoritma da yapılan tahminlemelerde -1 dakika ile +3 dakika aralığında kalan ve doğru tahmin olarak kabul edilen değerlerin oranı %42'dir. OrerTahminleme de yapılan doğru tahmin oranı ise %91'dir. Mevcut algoritmadaki hatalı oranın bu kadar yüksek olması doğrudan ham veri üzerinde çalıştırılmasıdır. Oysa öncelikle veriler incelenmeli, negatif ya da çok yüksek olan veriler geçiş verisi olarak kayıt edilmemelidir.

5. SONUÇ ve ÖNERİLER

Toplu taşıma araçlarını kullanan birçok yolcu yanlış tahmin verilerinden kaynaklı zaman kayıpları yaşamaktadır. Yapılan bu çalışmada tahmin verilerindeki sapmalar minimize edilmeye çalışılmıştır. Arşiv verilerinden faydalanılarak yeni bir algoritma oluşturulmuştur. Yapılan testler sonucunda önerilen yöntemin mevcut yönteme göre daha sağlıklı sonuçlar verdiği gösterilmiştir.

Bu çalışmada, 55 hattının ilk 20 durağındaki 1.500.000 geçiş verisinden oluşan arşiv verileri üzerinde veri madenciliği çalışması yapılarak, tahmin verisi üreten bir yaklaşım sunmaktadır. Bizim yöntemimizde sırasıyla, gün tipi ve saat dilimi kümesi verileri dikkate alınarak, veriler kümelenmekte ve bu veriler üzerinden gerçekleşecek seferin zamanı tahmin edilmektedir. Verinin temizlenmesi ve arşiv verilerinin gün tipi ve saat dilim kümesine göre gün sonunda oluşturulması sayesinde hesaplamadaki yük azaltılmıştır. Önerilen yönteme uygun, OrerTahminleme uygulaması geliştirilmiştir. Yapılan testlerde %92 doğru tahmin rakamına ulaşarak başarılı bir sonuç elde edilmiştir.

Varış zamanının tahmini probleminde, arşiv verilerinin saat dilim kümelerine göre ayrılması ve arşivlenmesi hiyerarşik kümeleme algoritması kullanılarak gerçekleştirilmiştir. Bu çalışmada kullanılan gün tipi mantığı farklı çalışmalarda da vardır ancak, farklı çalışmalarda gün tipi olarak hafta içi, hafta sonu olarak iki gün tipi belirlenirken bu çalışmada gün tipleri, hafta içi, cumartesi ve Pazar(tatil günleri) olarak üç gruba ayrılmıştır.

Yapılan işlemler sonucunda elde edilen tahmin verisinin yolcu ile paylaşılmadan önce tahmin edilen aralıktaki durak sayısı 5 ten fazla ise tahmin verisinden 1 dakika çıkarılarak yolcu ile paylaşılması da ilk defa bu çalışmada kullanılmıştır. Sonuçları kıyaslandığında çok iyi sonuçlar elde edilmiştir.

Algoritmada elde edilen %92'lik doğru tahmin sonucunun çok daha yukarılara taşınması için farklı verilerde algoritmaya dahil edilebilir. Hava durumu ve trafik verilerinin algoritmaya dahil edilmesi durumunda doğruluk oranının çok daha yüksek olacağı muhtemeldir.

KAYNAKLAR

Alkan, A. (2014). *Gezgin Satıcı Problemi İçin Veri Madenciliği Tabanlı Bir Model Önerisi* (Yayın No. 372716). [Yayınlanmış doktora tezi, Kocaeli Üniversitesi].: https://tez.yok.gov.tr/UlusalTezMerkezi/tezSorguSonucYeni.jsp adresinden alındı

Başkaya, O., Ağaçsapan, B., ve Çabuk, A. (2020). Akıllı Şehirler Kapsamında Yapay Zekâ Teknikleri Kullanarak Etkin Ulaşım Planlarının Oluşturulması Üzerine Bir Model Önerisi. *GSI Journals Serie C*, 1-21.

Bilgin, T. T. (2019). *Kümeleme Analizi: Temel Kavramlar ve Algoritmalar*. ttbilgin: Erişim Tarihi: 24 Mayıs 2021. http://www.ttbilgin.com/2018-2019-guz/veri-bilimine-giris/hafta7/kumeleme-analizi.pdf adresinden alındı

Boylu, B. (2021). *Travel Time Prediction İn Public Transportation* (Yayın no. 670397). [Yayınlanmış yüksek lisans tezi, İstanbul Ticaret Üniversitesi].: https://tez.yok.gov.tr/UlusalTezMerkezi/tezSorguSonucYeni.jsp adresinden alındı

Chen, G., Yang, X., An, J., ve Zhang, D. (2012). Bus-Arrival-Time Prediction Models: Link-Based and Section-Based. *JOURNAL OF TRANSPORTATION ENGINEERING © ASCE*, 60-66.

Dondurmacı, G. (2011). *Veri Madenciliğinde Regresyon Ağaçları İle Sınıflandırma Ve Bir Uygulama* (Yayın No. 304788). [Yayınlanmış doktora tezi, Mimar Sinan Güzel Sanatlar Üniversitesi].: https://tez.yok.gov.tr/UlusalTezMerkezi/tezSorguSonucYeni.jsp adresinden alındı

Esme, B. (2009). *Kalman Filter For Dummies*. Bilgin's Blog: Erişim Tarihi: 24 Mayıs 2021. http://bilgin.esme.org/BitsAndBytes/KalmanFilterforDummies adresinden alındı

Gözellik, E. Ö. (2018, 05 17). *2050'de Dünya Nüfusunun Üçte İkisi Şehirli Olacak.* Anadolu Ajansı: Erişim Tarihi: 20 Mart 2021. https://www.aa.com.tr/tr/dunya/2050de-dunya-nufusunun-ucte-ikisi-sehirli-olacak/1148769 adresinden alındı

Hand, D., Mannila, H., ve Padhraic, S. (2001). *Principles Of Data Mining.* Massachusetts Institute Of Technology.

Hua, X., Wang, W., Wang, Y., ve Ren, M. (2017). Bus Arrival Time Prediction Using Mixed Multi-Route. *Taylor ve Francis*, 543-555.

İETT. (2022). *Bunları Biliyor Musunuz?* https://iett.istanbul/: Erişim Tarihi: 20 Mart 2021. https://iett.istanbul/icerik/bunlari-biliyor-musunuz adresinden alındı

Kavurucu, Y. (2009). *An ILP-Based Concept Discovery System For Multi-Relational Data Mining* (Yayın No. 255312). [Yayınlanmış doktora tezi, Orta Doğu Teknik Üniversitesi].: https://tez.yok.gov.tr/UlusalTezMerkezi/tezSorguSonucYeni.jsp adresinden alındı

K-Means Algorithm for Cluster Analysis. (2018). juniata: Erişim Tarihi:10 Haziran 2021. https://jcsites.juniata.edu/faculty/rhodes/ids/kmeans.html adresinden alındı

Lutins, E. (2017). *DBSCAN: What is it? When to Use it? How to use it*. medium: Erişim Tarihi:14 Temmuz 2021. https://elutins.medium.com/dbscan-what-is-it-when-to-use-it-how-to-use-it-8bd506293818 adresinden alındı

Md Noor, R., Yik, N., Kolandaisamy, R., ve Ahmedy, I. (2020, 02 15). Predict Arrival Time By Using Machine Learning Algorithm. Erişim Tarihi:14 Temmuz 2021. *Preprints*, 1-18. preprints. adresinden alındı

Oğuz, P. D. (2018). *Ders Notları MA 6*. https://acikders.ankara.edu.tr/: Erişim Tarihi:14 Temmuz 2021. https://acikders.ankara.edu.tr/pluginfile.php/171125/mod_resource/content/1/6%20K%C3%BCmeleme.pdf adresinden alındı

Reich, T., Budka, M., ve Hulbert, D. (2021). Bus journey simulation to develop public transport predictive algorithms. *Elsevier*, 1-10.

Sever, S. Z. (2010, 266135). *Yoğunluk Tabanlı Kümeleme Metodları Kullanılarak Paralel Veri Madenciliği Gerçekleştirilmesi (Yayın no. 266135).* [Yayınlanmış yüksek lisans tezi, Maltepe Üniversitesi].: https://tez.yok.gov.tr/UlusalTezMerkezi/tezSorguSonucYeni.jsp adresinden alındı

Taşyürek, M., ve Çelik, M. (2020). Akıllı Durak Sistemindeki Araç Seyahat Sürelerinin Birleşik Yapay Sinir Ağları Kullanarak Tahmini. *Avrupa Bilim ve Teknoloji Dergisi Özel Sayı*, 72-79.

Tiryaki, M. M. (2015). *Akıllı Ulaşım İçin Veri Madenciliğine Dayalı Tahmin Sistemi* (Yayın no. 394516). [Yayınlanmış yüksek lisans tezi, İstanbul Üniversitesi].: https://tez.yok.gov.tr/UlusalTezMerkezi/tezSorguSonucYeni.jsp adresinden alındı

Ulgen, K. (2017). *Makine Öğrenimi Bölüm-3 (Yapay Sinir Ağları)*. medium: Erişim Tarihi: 10 Haziran 2021. https://medium.com/@k.ulgen90/makine-%C3%B6%C4%9Frenimi-b%C3%B6l%C3%BCm-3-4b160df1f4c8 adresinden alındı

Xu, H., ve Ying, J. (2017). Bus arrival time prediction with real-time and historic data. *Springer Science+Business Media*, 3099-3106.

Zhang, X., Lauber, L., Liu, H., Shi, J., Xie, M., ve Pan, Y. (2022). Travel time prediction of urban public transportation based on detection of single routes. *PLOS ONE*, 1-23.

ÖZGEÇMİŞ

Kadir KUŞTUL

Eğitim

Derece	Yıl Üniversite, Enstitü, Anabilim/ Anasanat Dalı
Y. Lisans	2009 Haliç Üniversitesi Fen Bilimleri/Yönetim Bilişim Sistemleri
Lisans	2006 Haliç Üniversitesi Mühendislik Fakültesi/Bilgisayar Mühendisliği
ÖnLisans	2001 Selçuk Üniversitesi Akören Ali Rıza Ercan Meslek Yüksek Okulu Bilgisayar Programcılığı

İş/İstihdam

Yıl	Kurum
2016-Devam	İETT
2014-2016	Belbim/İETT
2010-2019	İSMEK (Yönetici)
2001–2009	İSMEK