TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

COĞRAFİ KONUM VE SENSÖR VERİLERİ İLE GÖZETİMSİZ SÜRÜCÜ PERFORMANSI SKORLAMA

YÜKSEK LİSANS TEZİ

Ozan Fırat ÖZGÜL

Elektrik ve Elektronik Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Harun Taha HAYVACI

AĞUSTOS 2018

Fen Bilimleri Enstitüsü Onayı

……..
Prof. Dr. Osman EROĞUL
Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksininlerini sağladığını onaylarım.

……
Prof. Dr. Tolga GİRİCİ
Anabilimdalı Başkanı

TOBB ETÜ, Fen Bilimleri Enstitüsü'nün 161211113 numaralı Yüksek Lisans Öğrencisi **Ozan Fırat ÖZGÜL**'ün ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı **"COĞRAFİ KONUM VE SENSÖR VERİLERİ İLE GÖZETİMSİZ SÜRÜCÜ PERFORMANSI SKORLAMA"** başlıklı tezi **08.08.2018** tarihinde aşağıda imzaları olan jüri tarafından kabul edilmiştir.

Tez Danışmanı : **Dr. Öğr. Üyesi Harun Taha HAYVACI** .....
TOBB Ekonomi ve Teknoloji Üniversitesi

Jüri Üyeleri : **Prof. Dr. Ali KARA (Başkan)** .....
Atılım Üniversitesi

Prof. Dr. Bülent TAVLI .....
TOBB Ekonomi ve Teknoloji Üniversitesi

Dr. Öğr. Üyesi Harun Taha HAYVACI .....
TOBB Ekonomi ve Teknoloji Üniversitesi

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, alıntı yapılan kaynaklara eksiksiz atıf yapıldığını, referansların tam olarak belirtildiğini ve ayrıca bu tezin TOBB ETÜ Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırlandığını bildiririm.

Ozan Fırat ÖZGÜL

ÖZET

Yüksek Lisans Tezi

COĞRAFİ KONUM VE SENSÖR VERİLERİ İLE GÖZETİMSİZ SÜRÜCÜ PERFORMANSI SKORLAMA

Ozan Fırat ÖZGÜL

TOBB Ekonomi ve Teknoloji Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik ve Elektronik Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Harun Taha HAYVACI

Tarih: Ağustos 2018

Araç sürüş performansının ölçülmesi, özellikle otomotiv ve sigorta sektörlerinde çalışan araştırmacıların ilgisini çeken, oldukça zorlu bir konudur. Bu alandaki geçmiş çalışmaların bir kolu Denetleyici Alanı Veri Yolu Ağı (CAN Bus) ve Küresel Konum Belirleme Sistemi (GPS) çıktıları, fizyolojik veriler, kamera kayıtları ve pek çok diğer veri tipini öznitelik olarak kullanarak, etiketli veri setleri üzerinde agresif/agresif olmayan, dikkatli/dikkatsiz, uykulu/uykusuz gibi davranışsal ayrımları istatistiksel olarak öğrenmeyi amaçlamışlardır. Bir diğer akımda ise, araştırmacılar sürüş davranışlarını kural-bazlı olarak değerlendirmeyi tercih etmişlerdir. Ancak, bu yaklaşımlar etiketli verinin çoğu zaman mevcut olmaması, bütün yol şartlarını temsil edebilecek kuralların öğrenilememesi ve standart bir aracın gerekli bütün sensör modalitelerine sahip olmamasından dolayı kullanışlı değillerdir. Çalışmamızda, bu problemlerin hepsinin üstesinden gelen, minimalistik bir veri üzerinde skorlama yapma kapasitesine sahip, gözetimsiz bir olasılıksal model tasarlanmıştır.

Sunulan model, sürücüleri geleneksel anomali tespiti yaklaşımlarıyla değerlendirir. Buna göre, bir sürüş tecrübesinin geçmişte görülen örnekler üzerinden hesaplanan normlara ne kadar uyumlu olduğu, onun ne kadar yüksek skorlanacağını tanımlar. Bu

normlar, diğer çalışmalardan farklı olarak, yolun tipine ve trafik akışına bağlı olarak bulunur. Takip edilen olasılıksal yaklaşım, bu sürekli değişkenlerin bileşik olasılık dağılımlarının bilinmesini gerektirmektedir; ancak bu matematiksel olarak oldukça zorlu bir problemdir. Bu işlemi kolaylaştırmak için, değişkenlerden her birini gözetimsiz öğrenme yöntemleri ile ayrıklaştırma yoluna gidilmiştir. Bu sayede, değişkenleri ayrık az sayıda küme ile temsil etmek ve bu kümeler arasındaki paylaşılan eleman sayılarını kullanarak bileşik olasılık dağılımlarını kestirmek mümkün olmuştur. Bileşik dağılım bilgisi, Birlikte Kümelenme Matrisi (BKM) adlı bir yapıda tutulmuştur ve bu matris elde edildikten sonra, skorlama sadece matris üzerindeki pozisyonu bulma problemine indirgenmiştir.

Değişkenlerin gözetimsiz modellerle ayrıklaştırılması çalışmamızın merkez noktasını oluşturmaktadır. GPS verileri kullanarak yol tiplerinin kümelenmesi ve CAN Bus kayıtlarından yola çıkarak trafik akış tipi ve sürüş stili kümelenmeleri üzerinde durulmuş, doğru öznitelik seçimi hakkında bilgiler sunulmuş ve kümelenmenin farklı ayrışım metodları ve farklı benzerlik ölçütlerinden hangileriyle daha iyi başarıldığı saptanmıştır. Bu başarım sayısal olarak sunulmuş ve kullandığımız veri setinde en başarılı olan yöntemler saptanmıştır. Ardından bu başarının arkasında yatan faktörler sorgulanmıştır. Böylece alandaki gelecek çalışmalara ışık tutacak bir çerçeve oluşturulmaya çalışılmıştır. Buna ek olarak, kümelenmenin öznitelik uzayından değil de, daha düşük boyutlu bir uzaydan yola çıkılarak yapılmasının yararları açıklanmış, bu yöntem yol tipi ve sürüş stili kümeleme aşamasından uygulanmıştır.

Değişkenlerin kümelenmeleri başarıldıktan sonra, elimizde bulunan küçük bir etiketli veri seti üzerinde skorlama işlemi yapılmıştır. Burada agresif şoförlerin, agresif olmayanlardan genellikle daha düşük skorlar alması amaçlanmış ve bu başarılmıştır. Son aşamada ise, aynı başarının literatürdeki diğer bir güçlü modelin varyasyonu ile başarılıp başarılamayacağına bakılmıştır. Bu metot, bizim skorlama yaklaşımımızın tersine, agresif ve agresif olmayan şoförler arasında herhangi bir skorlama farkı gösterememiştir.

Anahtar Kelimeler: Sürücü skorlama, Gözetimsiz öğrenme, Yapay öğrenme.

ABSTRACT

Master of Science

UNSUPERVISED DRIVER PERFORMANCE SCORING USING GEOGRAPHICAL POSITION AND SENSOR DATA

Ozan Fırat ÖZGÜL

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Electrical and Electronics Master of Science Programme

Supervisor: Asst. Prof. Harun Taha HAYVACI

Date: August 2018

Rating driving performance is a challenging topic. It attracts professionals from a variety of domains such as automotive industry and insurance companies. A great majority of the previous studies combines multiple measurement modalities such as Controller Are Network (CAN Bus) data, physiological measurements, camera reconrdings and localization estimates from Global Positioning System (GPS). One school of thought attempted to discriminate agressive/non-agressive, attentive/inattentive or drowsy/wakeful drivers through a statistical learning. Other researchers applied a rule-based approach. However, this approaches are inapplicable since labelled data for supervised learning schemes is scarce and rules that are representative for all road conditions are not feasible. Moreover, the abundance of sensor modalities in a personal vehicle is rather costly. In order to handle these problems, in this work, we propose a fully unsupervised driving style scoring mechanism operating on a minimalistic dataset.

The proposed model operates similar to conventional anomaly detecton schemes. In this setting, a driving experience is scored in proportion to its congruency to the driving norms which are obtained as the most common driving patters in the training

data. As a novelity of our work, these norms are defined considering road type and traffic flow patterns. This is applied via a probabilistic approach where joint probability densities of the variables controlling road type, traffic flow type and driving style are required. Since estimating this probability is mathematically intractable, we follow an alternative approach relaxing the probability estimation through discretization. In this context, each of these variables are clustered by unsupervised learning techniques and the joint probabilities are approximated by the number elements shared between inter-variable clusters. This probability information is stored in a special architecture which we call Co-Clustering Matrix. (CCM). Once this matrix is learnt, scoring of a new driving experience is degraded into finding its position inside the matrix.

Clustering of these variables is the central point of our work. This part includes clustering of road types through GPS recordings and traffic flow type and driving style clustering by CAN Bus data as well as the identification of the most efficient clustering methods and distance metrics. All evaluations are supported by mathematical evidences and the factors behind successful methods are discuessed. In this way, we attempt to present a framework for the prospective studies. Furthermore, we discover the efficiency of the clustering of lower dimensional representations rather than the original feature sets.

Upon obtaining successful clustering of the data from multiple views, we validate our scoring mechanism utilizing a small labelled dataset. Here, the aggressive drivers are expected to obtain significantly lower scores than their nonaggressive counterparts. This is achieved and statistically validated. Following that, we follow the same procedure for another scoring methodology and in contrast to our approach, no change is observed between scoring patterns of aggressive and nonaggressive drivers.

Keywords: Driving style scoring, Unsupervised learning, machine learning.

TEŞEKKÜR

Sonsuz destekleri için aileme, Özlem'e ve değerli yardım ve katkılarıyla beni yönlendiren hocam Harun Taha Hayvacı'ya, bu çalışmanın gerçekleşmesinde büyük katkıları olan Mehmet Ulaş Çakır ve STM A.Ş Siber Güvenlik ve Büyük Veri Direktörlüğü çalışanlarına teşekkürlerimi sunarım. Ayrıca çalışmalarımı araştırma burslu statüsünde devam ettirmeme olanak sağlayan TOBB Ekonomi ve Teknoloji Üniversitesi'ne minnetlerimi sunarım.

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

KISALTMALAR

GPS : Küresel Konum Belirleme Sistemi (Global Positioning System)
CAN Bus : Denetleyici Alanı Veri Yolu Ağı (Controller Area Network)
EOG : Elektrookülography (Electryooculography)
YSA : Yapay Sinir Ağı
EEG : Elektroansefalografi (Electroencephalography)
EKG : Eletkrokardiyografi (Electrocardiography)
EMG : Elektromiyografi (Electromyogprahy)
BIRCH : Dengeli Yinelemeli Azaltma ve Hiyerarşik Kümeleme (Balanced Iterative Reducing and Clustering Using Hierarchies)
KÖA : Küme Öznitelikleri Ağacı
DBSCAN : Yoğunluk-bazlı Gürültülü Uzaysal Kümeleme (Density-based Spatial Clustering of Applications with Noise)
BKM : Birlikte Kümelenme Matrisi
TBA : Temel Bileşenler Analizi
EUOA : En Uzun Ortak Altdizi
DZB : Dinamik Zaman Bükülmesi
OSK : Ortalama Silüet Katsayısı

SEMBOL LİSTESİ

Bu çalışmada kullanılmış olan simgeler açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklama**
ε	y-ekseninde uzaysal eşik
σ	x-ekseninde uzaysal eşik
a_i	i'inci örneğin kendi kümesindeki elemanlara olan ortalama uzaklığı
b_i	i'inci örneğin en yakın başka kümedeki elemanlara olan ortalama uzaklığı
T_i	i'inci gezinge

1. GİRİŞ

Günümüz araç trafiği, üzerindeki 1 milyardan fazla araç ile insanoğlunun ortaya getirdiği en geniş çaplı ağlardan birisidir [1]. Ülkemizde ise bu sayı yaklaşık 22 milyon olarak tespit edilmiş olup, gündelik yaşamlarımızın en önemli bileşenlerinden birisi haline gelmiştir [2]. Modern araçların sağladığı ulaşım kolaylığı ve bununla beraber gelen yaşam kalitesi artışı yadsınamaz olsa da, trafik güvenliğini sağlamak oldukça zorlu bir problemdir.

Bu kompleks ağ içerisinde güvenliği sağlamak için zaman içerisinde sayısız koruyucu ve önleyici yaklaşımlar önerilmiş olsa da, bunların pek azı ağ içerisindeki temel kontrol mekanizması olan sürücünün davranışlarını değerlendirmeye yöneliktir. Bu işlem oldukça subjektif ve kompleks olmakla beraber, araç sürücülerine dair sağlıklı davranışsal (sürüş stili) değerlendirmeler elde edilebildiği takdirde, sürüş öncesinde önleyici, sürüş esnasında ise anlık koruyucu tedbirlerin alınması mümkün hale gelecektir. Burada önemli noktalardan bir tanesi, bu sürüş stili verilerinin ne şekilde elde edileceği ve bunların iyi ve kötü sürüşe delalet edecek şekle nasıl sokulacağıdır. Kullanılabilecek modaliteler çeşitli olmasına rağmen, bunların her araçta bulunması mümkün değildir. Örneğin, trafikteki bütün araçlara çoklu kamera sistemleri, fizyolojik kayıt üniteleri ve benzerlerini kurmak oldukça masraflı olacaktır. Üstelik bu veriler elde edilse bile, iyi/kötü sürücü ayrımını yapmak kolay olmayacaktır. Bunu başarabilmek için kural-bazlı yaklaşımlar kullanılabilir; ancak bütün yol şartlarında (yol tipi, trafik akışı gibi) geçerli olacak kurallar geliştirmek kolay bir iş değildir. Bu durum araştırmacıları genelde son zamanlarda yüksek popülariteye erişen istatistiksel öğrenme-bazlı yaklaşımlara yöneltmiştir. Bu mecrada, iyi ve kötü olarak etiketlenmiş geçmişe ait veri setleri üzerinde gelişmiş sınıflandırıcılar eğitilerek, sınıfları birbirlerinden ayıran istatistiklere erişilebilir. Böylece gelecekte görülecek sürüş verileri otomatik olarak eğitilmiş model üzerinden değerlendirilebilir. Yapay öğrenme yaklaşımı matematiksel olarak oldukça makul olsa da, etiketli sürüş verisi bulmak oldukça maliyetli ve yine subjektiflik içeren bir faaliyettir.

Hem bahsi geçen problemlerin etrafından dolaşabilecek, hem de sağlıklı sürüş stili değerlendirilmesi yapabilecek bir metodoloji literatürde rastlanmamış olmakla beraber, trafik güvenliğinin sağlanması açısından oldukça faydalı bir araç olacaktır.

1.1 Tezin Amacı

Bu tez çalışmasında, asgari sayıda veri modalitesi kullanan ve etiketli veriye ihtiyaç duymadan gelişmiş istatiksel öğrenme metodları kullanarak sürüş stillerini skorlayan bir çerçeve geliştirilmiştir. Skorlama yaklaşımı tamamen olasılıksal temellidir. Öncelikle, skorlanan sürüş tecrübesi için yol şartları değerlendirilerek, geçmiş veriler üzerinden, bu şartlar altındaki 'sürüş normu' tespit edilmeye çalışılır. Yani, 'Bu şartlarda, diğer sürücüler ne şekilde sürmüş?' sorusunun cevabı aranır. Elde edilen norm kullanılarak, mevcut sürücünün ne kadar makul bir sürüş tecrübesi yaşadığı tespit edilir. Burada makullük ölçüsü, normlara uzaklıkla ters orantılı olarak belirlenir. Bu uzaklık, doğrudan skor olarak atanır.

Tezin temel amacı, bu olasılıksal skorlama yaklaşımını ve tüm alt bileşenlerini kurgulamaktır. Alt bileşenler, yol tipi tespiti için lokasyon belirten Global Positioning System (GPS) verilerinin işlenmesi ve kümelenmesi; Controller Area Network (CAN Bus) kullanılarak, trafik akış durumu ve sürüş stili belirten özniteliklerin elde edilme ve kümelenmelerini içermektedir. Elde edilen skorlar, elimizde bulunan küçük çaplı agresif/agresif olmayan sürüş verilerini içeren bir veri seti üzerinde test edilecektir. Bu karşılaştırma, agresif olan ve olmayan sürücülere verilen puanlar arasındaki farkın maksimize edilmesi üzerinden değerlendirilecek; literatürdeki bir diğer yaklaşım ile karşılaştırmalarda da bulunulacaktır.

Çalışmamızda, STM A.Ş.'nin Kasım 2017- Mart 2018 arasında İzmir, İstanbul ve Adana'da görev yapmış 21 belediye otobüsü üzerinden elde ettiği GPS ve CAN Bus verileri kullanılmıştır. Mevcut sistem tasarımı, STM A.Ş.'nın SmartFleetics platformu üzerinde kullanılmak amacıyla geliştirilmiştir. Çalışmamız tamamlandığında, belediyeler, otobüslerinin sürücülerinin anlık sürüş stili skorlarına SmartFleetics platformu üzerinden ulaşabileceklerdir.

1.2 Literatür Araştırması

Konunun subjektif doğası, literatürde pek çok farklı yaklaşımın ortaya çıkmasına neden olmuştur. Bu çalışmalarda temel amaç, iyi ve kötü, dikkatli ve dikkatsiz,

uykulu ve uykusuz şoförleri birbirinden ayırt edebilecek analitik çözümler bulabilmektir. Bu ayrımlar pek çok farklı metodoloji ve sinyal tipi ile yapılabilir. Bunların başında fizyolojik, yani vücudun biyolojik süreçlerine dair tutulmuş kayıtlar gelmektedir. Örneğin [3], [4] ve [5]'de sürücülerin göz hareketleri kullanılarak, dikkat, konsantrasyon ve uykulu olmak durumları tespit edilmeye çalışılmıştır. Bu hareketlerin tespiti, iki şekilde yapılmaktadır: (1) Bir kamera yardımıyla, sırasıyla yüz tespiti ve göz tespiti yapılmış; ardından gözlerin hareketlerine erişilebilmiştir, (2) Göz bebeği hareketine bağlı değişen göz çevresi elektrik potansiyeli kayıtlarını tutan Electrooculography (EOG) sinyalleri işlenerek göz hareketleri analiz edilebilmiştir. Bu çalışmalarda temelde sürüş kabiliyetleri test edilmemekle beraber, dikkat, konsantrasyon ve uyku durumlarının sürüşe etkisi vurgulanmış ve elde edilen metriklerin sürüş performansı açısından belirleyici olduğu savunulmuştur. Benzer bir diğer çalışmada, sürücülerin ağız hareketleri işlenerek, bir yapay sinir ağı (YSA) yardımıyla dikkat dağılımı durumu tespit edilmiştir.

Bir diğer akım ise, Electroencephalography (EEG) [6] ve Electrocardiography (ECG) [7] verileri üzerinden kişilerin yorgunluk ve dikkat dağınıklıklarını tespit etmeyi amaçlamıştır. İlişkili bir diğer çalışmada ise, sürüş esnasında EEG ve ECG sinyalleri arasındaki ilintiler saptanmış ve bu ilişkinin yorgunluk durumu ile bağlantısı gösterilmeye çalışılmıştır [8]. Bu modaliteler, öncekilerden bilgice daha zengin olmalarından dolayı, sürüş analizi açısından daha uygun görülmüşlerdir. Bu modalitelerin daha kapsamlı işlenmeleri ile heyecan, korku, tedirginlik gibi duygusal durumları ölçmek de mümkündür [9]. Ancak, sürücü skorlama amacıyla yapılmış böyle bir çalışma mevcut değildir.

Bunlara ek olarak, Surface Electromyography (sEMG) ile servikal bölgedeki kas hareketlerini analiz ederek sürüş esnasındaki konforsuzluk değerlendirmesi [10] ve yine EMG kullanarak sürücü yorgunluğu tespiti [11] gibi kas hareketi-bazlı çalışmalara da rastlanmaktadır.

Literatürün bir diğer ayağında ise, fizyolojik veriler yerine araç üzerindeki sensörlerden elde edilmiş, çoğunlukla mekanik temelli veriler üzerinden sürücü sınıflandırma (agresif/agresif olmayan) sıklıkla görülmektedir. Bu çalışmalarda kullanılan veriler genelde CAN Bus ve GPS ile sınırlıdır. Örneğin, Quintero ve arkadaşları, çalışmalarında GPS kayıtları, direksiyon açısı ve pedal kullanım verileri üzerinden bir YSA sınıflandırıcısı eğitmişler ve sürücüleri agresif ve agresif olmayan şeklinde sınıflandırmayı başarmışlardır [12]. Bu çalışmayla ilgili en büyük problem

analizlerin sadece simülasyon verisi üzerinden yapılmış olmasıdır. Bir diğer simülasyon temelli çalışmada ise, eş uzunluktaki yol parçaları üzerinde sürücülerin hız normları, ivmelenme normları ve açısal hız ölçümleri ile bunların temel istatistikleri (ortalama, standart sapma, medyan değer, dörttebirlik değerleri) hesaplanmış ve bu değerler sürücü karakteristiğini ayırt eden öznitelikler olarak kullanılmıştır [13]. Bu çalışmada, özniteliklerin birbirleriyle oldukça ilintili oldukları yönünde bir varsayımda bulunulmuştur. Bu oldukça doğaldır; şayet aracın hızı, ivmelenmesi ve motor devri gibi özellikleri aslında birbirlerine sıkıca bağlı faktörler tarafından kontrol edilmektedir. Bu gibi durumlarda orijinal öznitelik setini kullanmak yerine, boyut-indirgenmiş; ancak bilgice daha zengin temsillerden yararlanmak mümkündür [14]. Bu yaklaşım, özellikle Doğal Dil İşleme çalışmalarında yoğunlukla kullanılmakta ve kelimeler için, benzer kelimelerin kümelenmesini sağlayan yeni temsiller (örneğin word2vec [15]) elde edilebilmektedir. Bu yaklaşımı sürüş stili verisine uygulayan çalışmalarda da sıklıkla driver2vec, driving2vec gibi terimlerle karşılaşılabilir. Bu düşük boyutlu temsillerin elde edilme aşamasında kullanılabilecek pek çok yaklaşım olmakla beraber, bahsi geçen çalışmalarda genellikle otokodlayıcı olarak adlandırılan, özelleşmiş YSA mimarilerinden yararlanılmaktadır. Otokodlayıcıların şişe boğazı (bottleneck) katmanında elde edilen temsiller üzerinde sınıflandırma yapıldığında, sürüş stillerini veya sürücü tiplerini ayırt etmek mümkün olmuştur. Benzer bir diğer çalışmada ise, elde edilen temsiller üzerinden genel geçer sürüş stilleri imzaları elde edilmiş, bu imzaları içeren bir risk modelleme matrisi yardımıyla, sürücünün hangi sınıfa ait olduğu, dolayısıyla sigorta ücretlendirilmesinin nasıl olması gerektiği tespit edilmiştir [16].

Düşük boyutlu temsillerin başrolünde olduğu farklı bir çalışma ise çeşitli sürüş stillerinin, kamera ile elde edilmiş yol örüntüsü üzerinde farklı renklerle görselleştirildiği bir çalışmadır [17]. Burada yazarlar, önceki çalışmalardaki gibi, araçtan toplanan CAN Bus verilerinin düşük boyutta davranışsal bilgileri daha iyi temsil ettiklerini hipotez ederek, bir Derin Seyrek Otokodlayıcı yardımıyla bu temsillere erişmişler ve farklı sürüş davranışlarını 2 boyutlu yol haritası üzerinde farklı renklerle görselleştirmeyi başarmışlardır. Burada sürüş stilleri, ileri gidiş/dönüşler/yüksek pozitif ve negatif ivmeli hareket gibi farklı sınıfları içermektedir.

Diğerlerinden farklı olarak, [18], [19], [20] gibi çalışmalarda ise bir kontrol teorisi yaklaşımı benimsenmiş ve sürüş stili, sürücü-araç-yol kapalı döngüsü üzerinde dinamik simülasyonlar temelinde analiz edilmiştir. Bu çalışmalarda, belirtilen kapalı döngü sistemin optimal tasarımın tespiti amaçlanmıştır.

Literatürdeki bir diğer yaklaşım ise, sürüş esnasında elde edilmiş çeşitli sensör verilerini kullanarak, o esnada aracı kullanan kişinin kimliğini saptamaktır. Sürücü tespitinin başarıyla uygulanması; pek çok farklı uygulamaya kapı açmaktadır: (1) Aracın sahibi dışındaki kişiler tarafından kullanıldığı durumlarda, güvenlik uyarılarında bulunmak, (2) Kişiye özel sürüş asistan tasarımı, (3) sürücünün kendi normlarından sapmasına endeksli anlık uykusuzluk/dikkatsizlik/sarhoşluk uyarıları oluşturulması. Bu geniş uygulama alanları, otomotiv endüstrisi ve ilişkili araştırmacıları cezbetmekte, dolayısıyla bu konuya adanmış çalışmaların sayısı gün geçtikçe artmaktadır [21].

Literatürdeki sürüş stili çalışmalarındaki en büyük eksik, GPS-bazlı yol tipi verisinin neredeyse hiç dikkate alınmamasıdır. Bu, yapılan çalışmaların genellikle yol tipinden bağımsız olmasına ve buna bağlı değişen sürüş stili değişikliklerinin sürüş stili değişikliği olarak algılanmasına neden olmaktadır. Yaptığımız detaylı literatür taramasının sonucunda, yol tipini dikkate alan sadece bir çalışmayla karşılaştık. Bu çalışmada, GPS-bazlı gezinge verileri karşılaştırılarak, normal ve anormal sürüş ayrımı yapılmaya çalışılmıştır [22]. Burada gezinge verileri, aracın uzaydaki pozisyonlarının zamansal olarak temsil edilmeleriyle oluşturulmuştur. Bu çalışmadaki en önemli eksik ise, gezinge örüntüleri dışında herhangi bir veriden yararlanılmamış olmasıdır.

2. ÖNERİLEN SÜRÜŞ STİLİ SKORLAMA METODOLOJİSİ

2.1 Amaç

Bu bölümde, çalışmamız sonucu ortaya konulan özgün sürüş stili skorlama metodolojisi ayrıntılı bir biçimde ortaya konulacaktır. Burada, genel motivasyon ve matematiksel altyapının okuyucuyu sunulmasının ardından, metodolojinin alt bileşenleri tek tek incelenecektir.

2.2 Kullanılan Veri Seti

Çalışmamızda, STM A.Ş. tarafından İnTelA projesi kapsamında; İstanbul, Ankara ve İzmir'de görev yapmakta olan toplam 21 belediye otobüsünden toplanmış CAN Bus ve GPS verileri kullanılmıştır. Veriler Kasım-Aralık 2017 tarihleri aralığında tutulmuş ve örnekleme frekansı 1 Hz değerine sabitlenmiştir.

GPS verileri, enlem ve boylam açılarını içermektedir. Çalışmamızda, yolların iniş-çıkış karakteristiklerinin de öneme sahip olması nedeniyle, anlık rakım bilgisine de ihtiyaç duyulmuştur. Bunu elde etmek için Google Elevation API aracından yararlanılmıştır. Bu şekilde, bir aracın saniye başına bulunduğu üç-boyutlu pozisyon elde edilebilmiştir.

CAN Bus verileri ise 51 alandan oluşan büyük veri karakteristiğine sahiptir. Bu öznitelikler arasında çalışmamızla ilişkili bulunanlar, motor yükü, motor hızı gibi motor ile ilişkili girdiler; hız/ivme bilgileri; yakıt tüketim verileri; araç sıvı sıcaklıkları; gaz/fren pedal basış açıları olarak listelenebilir.

2.3 Olasılıksal Skorlama Yaklaşımı

Literatürde, özellikle sürücü değerlendirme, agresif/agresif olmayan şoför sınıflandırması, şoför tanıma uygulamaları ve uykusuzluk/dikkatsizlik tespiti gibi konularda umut veren sonuçlara ulaşılmış olsa da, neredeyse hepsi şu problemlerden muzdarip olmaktadırlar:

1) Skorlama ya statik kural-bazlı ya da gözetimli öğrenme (supervised learning) şeklindedir. Bu durum; statik kuralların bütün yol şartları için genelleştirilemeyeceğinden ve gözetimli öğrenme için gerekli olan etiketli verinin zor bulunur olmasından dolayı sağlıklı değildir.
2) Geçmiş çalışmalar, çok sayıda farklı tip sensör verisinden yararlanmışlar. Ancak araçları her tip fizyolojik, optik, mekanik ve elektronik kayıt üniteleriyle donatmak oldukça maliyetli olacaktır.
3) Farklı yol tiplerinin ve trafik akış örüntülerinin sürüşe etkisi modellenmemiştir.

Aynı anda gerçeklenebilir, makul ve gerçekçi skorlama yapabilen bir metodolojinin tasarlanması, bütün bu problemlerin aşılmasına bağlıdır. Bu problemlerin hepsini aşabilecek bir metodlojinin sahip olması gereken temel özellikler:

1) Herhangi bir etiketli veri ya da kurala ihtiyaç duymadan, veri üzerinden gözetimsiz (unsupervised) olarak iyi/kötü sürüş normları öğrenebilmek,
2) Her araçta bulunabilecek minimalistik bir veri seti kullanmak.
3) Skorlama yaklaşımını yol tipi/geometrisine bağımlı bir hale getirmek.

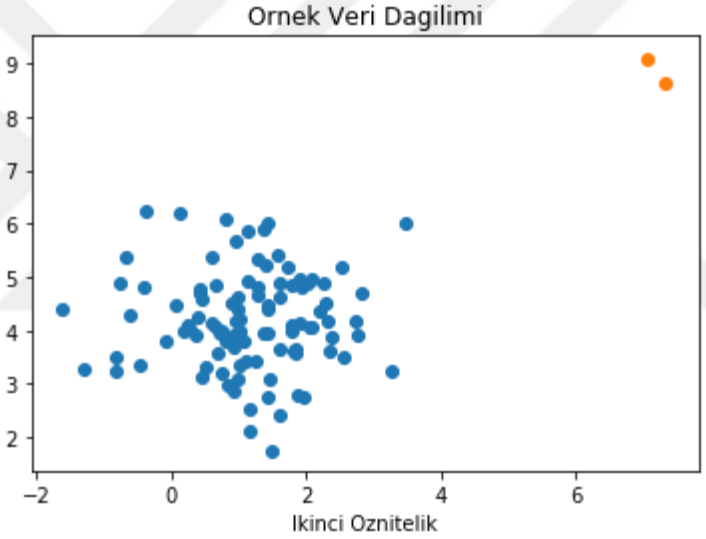
Çalışmamızda, bütün bu şartları sağlayan, dolayısıyla sık karşılaşılan problemlerin çevresinden dolaşan bir metodoloji tasarlamayı amaç edindik. Bu metodoloji, sadece her araçta bulunan GPS ve CAN Bus verilerinden yararlanarak, tamamen gözetimsiz olarak faaliyet göstermektedir. Çalışmamızın temeli, doğru sürüş tipinin yol geometrisi ve trafik akış şemasına bağlı olduğu varsayımıdır. Örneğin CAN Bus üzerinde yer alan gaz pedalı basış bilgilerini ele alalım. Bu veri alanından daha önce bahsedilen geçmiş çalışmalarda sıklıkla yararlanılmıştır. Sabit trafik akış şartlarında, gaz pedalına agresif basış, yokuş yukarı bir yolda oldukça normalken; yokuş aşağı giderken çok tehlikelidir. Aynı durum farklı trafik akış durumları için de düşünülebilir. Diğer bir deyişle, sürüş normları lokal şartlar tarafından belirlenir ve makul bir değerlendirme mekanizmasının bu değişken normları öğrenerek, normlara uygun sürüş tecrübelerine daha yüksek skorlar vermesi gerekmektedir. Peki, bu skorlar nasıl verilecektir?

2.4 Anomali Tespiti

Anomaliler, veri içerisindeki, iyi tanımlanmış normal davranış ile uyuşmayan örneklerdir. Bunlar, uygulamaya bağlı olarak, kötü niyetli internet aktivitesi, bozuk

sensör verisi, arızalanmış cihaz kayıtları olabilir. Bu tip verilerin tespiti sistem optimizasyonu açısından yararlı olabileceği gibi, normlardan sapan davranışların raporlanması siber saldırı tespiti gibi konularda başlı başına bir uygulama alanıdır [23].

Şekil 2.1'de örnek bir veri saçılımı görünmektedir. Eksenler farklı öznitelikleri, noktalar ise örnekleri temsil etmektedir. Burada mavi renk ile gösterilmiş noktalar, birbirlerine yakın, bir veri bulutu oluşturan örneklerdir. Buna karşın, turuncu örnekler ise diğerlerinden oldukça uzağa düşen örneklerdir. Burada elimizde normal davranışın ne olduğuna dair herhangi bir ön bilgi bulunmamasına rağmen, çok sayıda örneğin bir bulut oluşturuyor olması bu bulutun normal davranışı temsil ettiğini düşünmemeizi sağlamaktadır. Bu durumda, diğerlerden uzağa düşen örnekler normal davranışa uymayan olarak etiketlenebilirler. Bunlara anomali denilir.



Şekil 2.1: Örnek bir veri dağılımı.

Bu yaklaşımdaki en önemli varsayım, normal davranışın çok sayıda örnek tarafından paylaşılan, yani çok sık tecrübe edilen durumlar olduğudur. Bu yaklaşım sayesinde, içeriksel olarak normal olma durumuyla ilgilenmeye gerek kalmadan, sadece örnek-bazlı düşünerek normal/anormal ayrımı yapmak mümkündür. Peki bu yaklaşım şoför davranış skorlama ile nasıl ilişkilendirilebilir?

Daha önce tartışıldığı gibi iyi/kötü sürüş tecrübelerini belirli kurallara bağlamak, bu kuralların tanımlanmasının zorluğundan dolayı kullanışlı değildir. Ancak, bunun yerine örnekler arası benzerliklerden yararlanan tek bir kural kullanabiliriz. Bu kural, anomali tespitinde olduğu gibi, diğer noktalara uzaklık-bazlı olabilir. Yani, bir sürüş, geçmişteki diğer sürüş tecrübeleri tarafından tanımlanan normlara ne kadar

uyumluysa, o kadar başarılı; ne kadar uyumsuz ise o kadar başarısızdır. Bu uyum örnekler arasında mesafe ölçümleri yapılarak kolaylıkla elde edilebilir. Üstelik mesafe ölçümleri sürekli bir ağ üzerinde yapıldığı takdirde, bu ölçümler doğrudan skor olarak da kullanılabilir. Örneğin, diğer bütün noktaların oluşturduğu bulutun merkezine *x* birim uzaklıkta bulunan bir nokta, eğer bu metriğin alabileceği değer üzerinde birim normalizasyon uygulanırsa, *1/x* şeklinde skorlanabilir. Bu skor anomali derecesini belirtmekte olup, bulut üzerindeki noktalarda bire yakın olacaktır. Bu durumda, diğer şoförlere en uzak şoför en düşük skoru alacak ve sürüş skorlama sisteminin en önemli isterlerinden bir tanesi karşılanmış olacaktır.

Fakat burada iki tip problem hala çözümsüzdür. Bunlardan ilki, sürüş tecrübelerinin ne şekilde tanımlandığı, yani bir sürücüyü hangi özelliklerin temsil ettiğidir. Bu durum standart bir öznitelik çıkarma problemidir ve alan bilgisi kullanılarak çözümlenebilir. İkinci problem ise, çoğu zaman elimizde normal örneklerin kümelendiği bir veri bulutu olmaması, aksine çok sayıda bulutun bulunmasıdır. Bunun nedeni, daha önce de belirtildiği gibi, şartlara bağlı olarak birden fazla sürüş normu olmasıdır. Dolayısıyla, bir sürüş tecrübesini hangi şartlar üzerinde norm olarak tanımlanmış veri bulutuna dahil olması gerektiğine, yani skorun hangi merkeze göre belirleneceğine karar vermek gerekmektedir. Bu aşamada en sağlıklı yöntem, gelecek bölümde tartışılacak olan olasılıksal skorlamadır.

2.5 Sürüş Stili Skorlama

Önceki bölümde, bir örneğin örnekler bulutuna uzaklığının aslında skorlama için kullanılabileceğini gördük. Burada salt bir uzaklık ölçüsü yerine, olasılıksal bir yaklaşım da benimsenebilir. Bir diğer deyişle, bir örneğin bir gruba uzaklığı, aslında o gruba ait olma olasılığıyla ters orantılıdır. Skorlama yaklaşımında da mesafe ile ters orantılılık kabul edildiğine göre, aslında bu olasılık doğrudan skor olarak kullanılabilir. Kısacası, bir sürüş stilinin ne kadar olası olduğu bilgisinden, doğrudan bir skor olarak yararlanılabilir (Eşitlik (2.1)).

$$\text{Skor} \sim p(\text{Sürüş Stili}) \tag{2.1}$$

Bu olasılıksal yaklaşımın en büyük avantajı, trafik durumuyla ilgili değişkenlerin de hesaba katılabilmesidir. Örneğin D, T ve F sürekli rasgele değişkenlerinin sırasıyla sürüş stili, yol tipi ve trafik akış özelliklerini temsil ettiğini düşünelim. Bu durumda, Eşitlik (2.1) tekrar düzenlenerek, Eşitlik (2.2) haline sokulabilir.

$$\text{Skor} \sim p(D|T,F) \tag{2.2}$$

Eşitlik (2.2), sürüş stilinin yol tipi ve trafik akışına koşullandırılmış olduğunu göstermektedir. Yani, belirli bir yol tipi ve trafik akışını gözlemledikten sonra, o şartlar altındaki sürüş normu bulunur ve sürüş skoru bu norm üzerinden verilir. Böylece hem olasılıksal skorlama yaklaşımı korunmuş, hem de çoklu-norm isteri yerine getirilmiş olur. Bir diğer iyi haber ise Eşitlik (2.2)'nin olasılıksal zincir kuralına tabi tutulması ile, Eşitlik (2.3)'de görülen, sadece bileşik olasılık içeren bir denklem elde edilebilmesidir.

$$p(D|T,F)= p(D,T,F)/P(T,F) \tag{2.3}$$

Eşitlik (2.3)'e göre, bir sürüş skorlama, sürüş stilleri, yol tipleri ve trafik akış şekillerinin birlikte gerçekleşme olasılıkları bilgisiyle hesaplanabilir. Burada temel problem, bileşik olasılıkların kestirilmesinin zorluğudur. Bunlar, Karışık Gaussian Modeli gibi basit model varsayımları altında bile kapalı yapıda çözümlere sahip değillerdir. Bundan dolayı, Eşitlik (2.3)'ü doğrudan çözmek mümkün görünmemektedir.

Çalışmamızda, Eşitlik (2.3)'ün sonucunu kestirebilmek için, rasgele değişkenleri ayrıklaştırma yoluna gittik. Böylece, her bir değişkenin yalnızca sonlu sayıda sınıfa ait olabilmesi sağlanmış ve bileşik olasılıkların kestirilmesi mümkün hale gelmiştir. Bu işlemin arkasındaki temel motivasyon, ayrık rasgele değişkenlerin bileşik olasılıkların En Büyük Olabilirlik Kestiriminin, değişkenlerarası bir çok-boyutlu histogram kullanılarak hesaplanabilir olmasıdır. Bir diğer deyişle, iki veya daha fazla değişkenin bileşik olasılıkları, bu değişkenlerin sahip oldukları ayrık sınıfların, ne kadar çok geçmiş örnek tarafından paylaşıldığı bilgisi kullanılarak elde edilebilir. Bu noktada cevap verilmesi gereken soru ise, bu ayrıklaştırmanın nasıl yapılacağıdır.

2.6 Kümeleme ve Değişken Ayrıklaştırma

Kümeleme, verinin gözetimsiz olarak, içerdiği örüntüler göz önünde bulundurularak farklı kümelere ayrılması anlamına gelmektir. Gözetimli öğrenme yaklaşımlarının aksine, kümeleme algoritmaları etiketli veriye ihtiyaç duymazlar. Bu onları etiketli verinin maliyetli olduğu durumlarda çok etkili bir araç haline getirir.

Tıpkı farklı gözetimli öğrenme algoritmalarının veriyi sınıflandırmada farklı yollar izlemeleri gibi, her bir kümeleme algoritması da, veriyi farklı yaklaşımlarla kümeler.

Bunlar her bir kümenin iç varyasyonlarını minimize etmek amaçlı olabileceği gibi, uzaydaki örnek yoğunluğu üzerinden de olabilir. Bu kümeleme yaklaşımlarından, en önemli dört tanesi takip eden bölümde bölümde işlenmektedir.

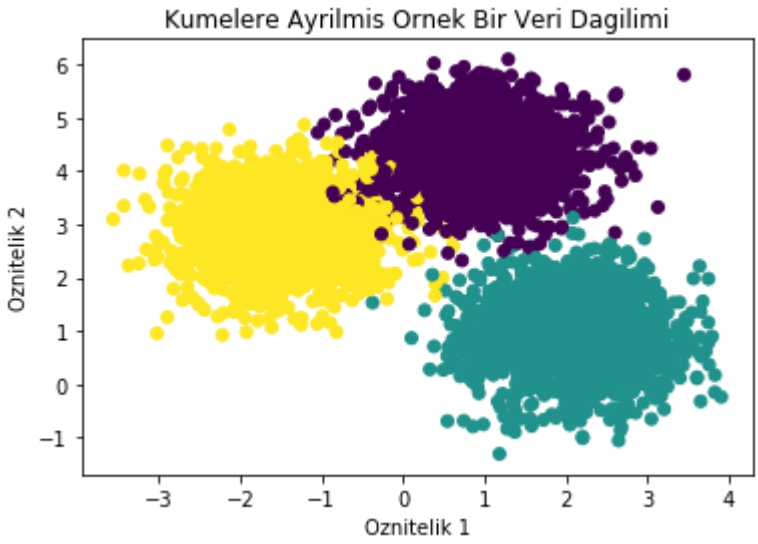
2.6.1 Ayrıştırma-bazlı modeller

Ayrıştırma-bazlı Modeller, N elemanlı bir veri setini, kullanıcı tarafından belirlenmiş k adet alt parçaya bölmeye çalışır ($k \leq N$). Bu alt parçaların her birine bir küme denir ve her küme en az bir örnek içermek zorundadır. Tek seviyeli ayrıştırma yapan bu modeller çoğunlukla kümeler arası maksimum ayrışımı amaçlarlar. Bu yaklaşımda her eleman yalnızca bir kümeye üye olabilmesine rağmen, bulanık kümeleme gibi bazı yöntemlerde, çoklu üyelik mümkündür.

Ayrıştırma-bazlı modeller, ayrışımları genellikle uzaklık-bazlı tanımlarlar. Bu modellerin eğitimleri ise yinelemeli olarak yapılır. Öncelikle, her bir nokta rastgele bir kümeye atanır. İkinci aşamada, her bir kümedeki örneklerin birbirlerine ne kadar benzedikleri üzerinden bir küme içi değişkenlik metriği hesaplanır ve küme üyelikleri bu metriği azaltacak şekilde güncellenir. Bu işlem, güncellenen eleman sayısı belirli bir eşik değerinin altında kalana kadar devam eder. Elde edilen çözüm optimaldir.

Ayrıştırma-bazlı Modeller arasında en popüleri, K-MEANS algoritmasıdır. K-MEANS, örnekler arası mesafeyi Öklid uzunluğu kullanarak hesaplar ve kullanıcı tarafından belirlenmiş maksimum yinelenme sayısı içerisinde küme-içi Öklid mesafelerini minimize edecek küme üyelik bilgisini elde eder. Bununla beraber, toplam küme sayısı da algoritmaya girdi olarak sunulmalıdır.

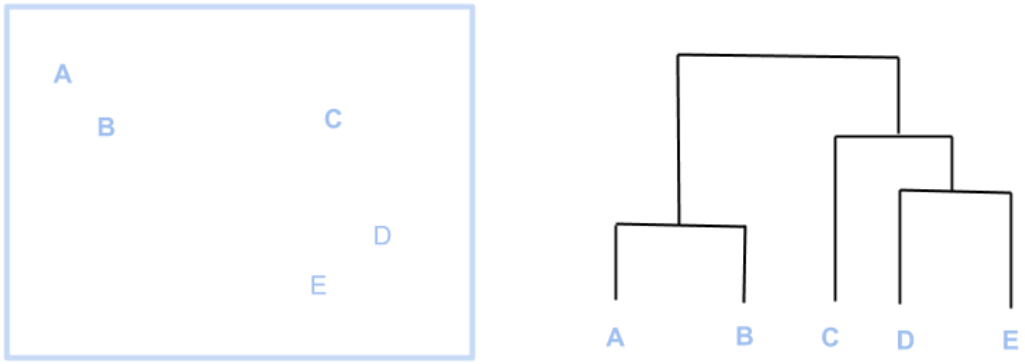
K-MEANS, küresel veya konveks kümeleri ayrıştırmakta çok başarılıdır. Şekil 2.2'de Gaussian damlalar halinde dağılmış bir veri bulutunun, K-MEANS kullanılarak kümelenmesi bunu açıkça göstermektedir. Buna karşın, K-MEANS gürültüden en çok etkilenen kümeleme algoritmalarından biridir. Bu nedenle, K-MEANS ile kümeleme yapmadan önce, veri dağılımı örüntüsünün konveks karakteristik taşıyıp taşımadığı ve dağılımın ne kadar gürültülü olduğuna dair gözlemlerde bulunmak gerekmektedir [24].



Şekil 2.2: K-MEANS algoritması kullanılarak 3 kümeye ayrştırılmış bir veri seti.

2.6.2 Hiyerarşi-bazlı modeller

Bir diğer kümeleme yaklaşımı, verilerin hiyerarşik olarak çözümlenmelerini içerir. Hiyerarşik Yığınlayıcı yöntemler, en küçük veri ünitelerinden başlayarak, birbirlerine daha yakın olan örneklerin bir araya getirilmelerini amaçlar. Bir diğer deyişle, işlemin başlangıcında, her örnek farklı bir küme olarak kabul edilir. Örnekler biraraya getirildikçe, küme sayısı düşer. Bu işlem, bütün örnekler tek bir büyük kümeye dahil edilene kadar devam eder. Şekil 2.3 bu yaklaşımın bir örneğini sunmaktadır.



(a) (b)

Şekil 2.3: (a) Örnek bir veri dağılımı (b) Bu dağılım için çizilmiş bir dendrogram.

Şekil 2.3'de; A,B,C,D,E ve F örnekleri hiyerarşik olarak kümelenmektedir. Dendrogram, E ve F örneklerinin en yakın olduğunu göstermektedir. Bu küme, bir üst seviyede D, daha sonra da D örneği ile birleşmektedir. En son noktada ise, A ve B'nin oluşturduğu küme bu bütüne dahil olmaktadır. Hiyerarşik yöntemler, mesafeyi doğrudan mesafe metrikleri cinsinden alabileceği gibi, ilinti gibi herhangi bir benzerlik metriği ile ters orantılı olan herhangi bir örnekler-arası ilişki formülasyonunu da kabul edebilirler.

Hiyerarşik yöntemlerin en büyük avantajları, kullanıcıdan küme sayısı girdisini beklememeleridir. Kullanıcı, Şekil 2.3'de görselleştirilen dendrogramı istediği seviyede keserek, farklı sayıda küme sayısı elde edebilir. Buna ek olarak, hiyerarşik modeller, herhangi bir veri dağılım şeklini tercih etmez, her tip dağılım morfolojisi üzerinde çalışırlar.

Çalışmamızda, hiyerarşik modeller arasından Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) modelini tercih ettik. BIRCH diğer hiyerarşik modellerden farklı bir yol izleyerek kümeleme yapar. BIRCH operasyon şemasında kümeler küçük bir özet istatistik seti yardımıyla temsil edilir. Bu istatistikler, kümenin içerisinde ne kadar örnek olduğunu, bunların ortalama koordinatlarının ne olduğunu ve bu ortalama koordinat etrafındaki saçılımın ne kadar güçlü olduğunu temsil ederler. Bunlara sırasıyla, merkez, yarıçap ve çap isimleri verilir. Bu istatistiklerden oluşan üniteler, Küme Öznitelikleri Ağacı (KÖA) adlı bir yapı içerisinde tutulur. Bu ağacın yaprak düğümleri bulunmak istenen kümeleri içerirken, yaprak olmayan düğümler ise, çok sayıda alt küme içerirler. Yaprak olmayan düğümlerde, kümeler merkez, yarıçap ve çap değerleri toplanmak suretiyle birleştirilirler. Her yeni gelen veri, bu ağaç üzerinde kendisi ile uyuşan yolu izleyerek, nihai bir kümeye ulaşır. Bir yaprak düğümü olmayan ünitede kaç alt ünitenin olabileceği ve bir küme içerisinde dağılımın ne kadar geniş olabileceği sistem parametreleri ile belirlenir. BIRCH, hem gürültülü durumlarda bile başarılı kümele yapabilmesi, hem de büyük veri üzerinde hızı çalışabilmesi nedeniyle tercih sebebi olmaktadır. Buna ek olarak, K-MEANS gibi BIRCH de konveks kümeler için daha iyi çalışmaktadır [25].

2.6.3 Yoğunluk-bazlı modeller

Önceki kümeleme yaklaşımları örnekler arasındaki uzaklık veya benzerlikleri temel almaktaydılar. Bunlar konveks/küresel şekildeki kümelerde başarı sağlamış olsalar da, rasgele şekiller üzerinde aynı performansı gösteremez.

Yoğunluk-bazlı metotlar, örnekler-arası uzaklık/benzerlik ilişkilerinden sıyrılarak, gelişigüzel şekilde kümeler tespit edebilmeyi amaçlarlar. Bunu yaparken, uzaydaki örnek yoğunluğundan yararlanırlar. Buna göre, bir küme etrafındaki örnek yoğunluğu bir eşik değerinin altına inene kadar genişlemeye devam eder. Bir diğer deyişle, bir küme içerisindeki her bir noktanın çevresindeki sabit bir alanda, en az belirli bir sayıda örnek bulunmalıdır. Böylece, kümeler örneklerin yoğun olduğu alanlarda serbestçe genişleyebilir ve rastgele şekiller alabilirler [26].

Density-based Spatial Clustering of Applications with Noise (DBSCAN), yoğunluk-bazlı kümeleme yöntemleri arasında literatürde en başarılı bulunanlardan biridir. DBSCAN, her bir noktayı; çekirdek, erişilebilir veya aykırı değer olarak sınıflandırmaya çalışır. Buna göre, eğer bir noktanın çevresindeki belirli bir alanda, en az önceden belirtilmiş sayıda örnek varsa, bu nokta bir çekirdektir. Çekirdeğin çevresindeki örnekler ise erişilebilir örnekler olarak etiketlenir. En az bir çekirdek, erişebildiği örneklerle beraber bir küme tanımlar. Bu şekilde kümeleme başarılabilir. Ancak bu şema içerisinde, hiç bir çekirdek tarafından erişilemeyen örnekler de var olabilir. DBSCAN, bu örnekleri aykırı değer olarak etiketler. Böylece, gürültülü örneklerin kümeleme metodolojisini etkilemesine izin verilmez.

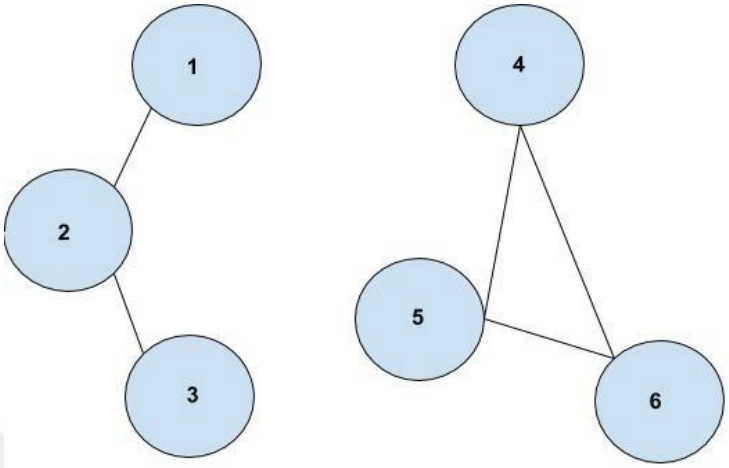
DBSCAN, her türlü morfolojide, gürültüye karşı kümeleme yapabilmektedir. Bu özelliği sayesinde literatürde sıkça yer bulmaktadır.

2.6.4 Çizge Ayrıştırma-bazlı modeller

Çizge Ayrıştırma-bazlı Modeller, aslında standart Ayrıştırma-bazlı Modeller sınıfına ait olsalar da, bu sınıf ile aralarındaki keskin farklar nedeniyle farklı bir bölümde işlenmişlerdir.

Çizgeler, farklı örnekler arasındaki ilişkileri görselleştiren araçlardır. Şekil 2.4 örnek bir çizge sunmaktadır. Bir çizgede, her bir örnek bir köşe olarak tanımlanırken, köşeleri bağlayan ünitelere de kenar adı verilir. İki köşe arasında bir kenar çizilmesi, bu iki köşenin (örneğin) birbirleriyle ilişkili olduğu anlamına gelmektedir. Bu ilişki

pek çok şekilde tanımlanabileceği gibi, genellikle eşiklenmiş benzerlik değerlerinden yararlanılmaktadır.



Şekil 2.4: Örnek bir çizge.

Şekil 2.4'de sunulan çizge açık bir şekilde; 1. örneğin 2. örnek ile ve 2. örneğin 3. örnek ile benzer olduğunu göstermektedir. Bu durum 1,2 ve 5 numaralı örnekler için de geçerlidir. Bu çizge,birbirleriyle bağlı olma durumuna göre rahatlıkla {1,2,3} ve {4,5,6} şeklinde iki kümeye ayrılabilir. Çizge-bazlı ayrıştırmanın temel çalışma mantığı budur.

Spektral Kümeleme algoritması, öncelikle örnekleri bir çizge halinde tutar. Aralarından sıfırdan büyük benzerlik olan bütün örnekler arasına kenarlar yerleştirir. Bu kenarlara, benzerlikle orantılı bir ağırlık atanır. Bu orantı genellikle bir radyal bazlı fonksiyon vasıtasıyla sağlanır. Böylece birbirlerine daha çok benzeyen örnekler arasındaki bağlantı, daha az benzeyenlerden daha kuvvetli olacaktır. Ardından, çizge öyle bir şekilde ayrıştırılır ki, aynı küme içerisinde kalan örnekler arasındaki ağırlıklar mümkün olduğunca yüksek olurken, farklı gruplar arası bağlantılar ise bir o kadar zayıf olur. Bu şekilde, daha benzer örnekleri aynı kümede tutmak mümkün olur [27].

2.6.5 Birlikte kümelenme matrisi

Önceki bölümlerdeki güçlü kümeleme yaklaşımlarını arkamıza aldıktan sonra, sürüş stili, yol tipi ve trafik akışını temsil eden rasgele değişkenleri kümeleme yoluyla

ayrıklaştırabiliriz. Bu şekilde, Eşitlik (2.3)'de tanımlanan olasılıksal skoru elde etmek mümkün olacaktır.

Ayrıklaştırma işlemi, her değişken için (D,T ve F) ayrı ayrı yapılmalıdır. Bu da, aynı örneğin; D, T ve F kümeleme için farklı özniteliklerle temsil edilmesini gerektirmektedir. Bir diğer deyişle, bir sürüş verisini, sürüş stili, yol tipi ve trafik açısı bilgilerini yansıtacak şekilde temsil edebilmemiz gerekmektedir. Bu özniteliklerin neler olabileceği ve nasıl seçildikleri ilerleyen kısımlarda detaylı incelenecektir. Bu tip kümeleme yaklaşımı ise, literatürde çok-bakışlı kümeleme olarak adlandırılır. Bu bölümde, değişkenlerin ne şekilde kümeleneceği konusu işlenmeyecek; ancak bu kümelenmenin hali hazırda başarılmış olduğu varsayılacaktır.

D, T ve F alanında kümelemeler başarıldıktan sonra, sırasıyla C_D, C_T ve C_F, yani her alandaki kümeleme bilgisini elde edebiliriz. Burada alan başına düşen küme sayısı farklı olabilmektedir. Bu konuda ilerleyen bölümlerde daha geniş bilgiler verilecektir.

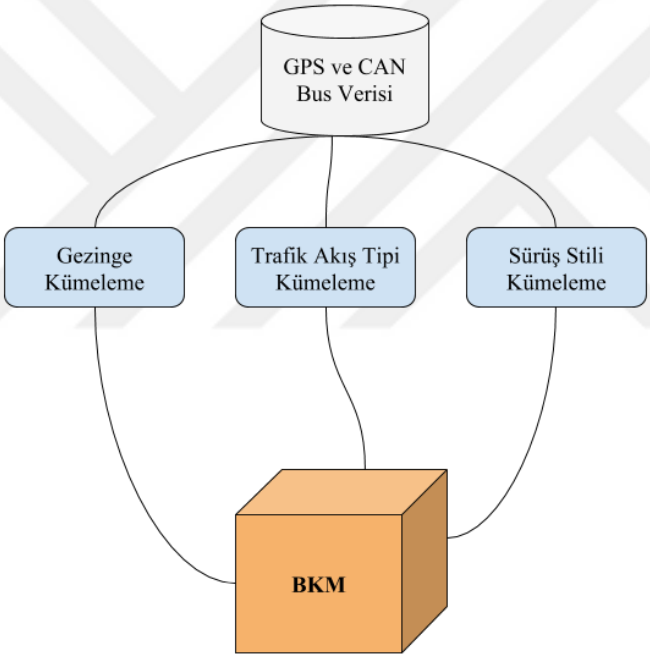
D,T ve F arasındaki bileşik olasılık bilgisi, daha önce de belirtildiği gibi, bu kümeler arasında paylaşılan eleman sayıları kullanılarak başarılabilir. Çalışmamızda, bu bilgiyi Birlikte Kümelenme Matrisi (BKM) adlı bir yapı kullanarak elde ettik. Bu yapı, her boyutunda, farklı bir alandaki kümeleri (D,T ve F) içeren 3-boyutlu bir matristen başka bir şey değildir. BKM, Eşitlik (2.4)'de tanımlanmıştır.

$$BKM(i,j,k) = \frac{C_T^i \cap C_F^j \cap C_D^k}{N} \tag{2.4}$$

Eşitlik (2.4)'de, C_D^i, C_T^j and C_F^k sırasıyla i'inci sürüş stili kümesi, j'inci yol tipi kümesi ve k'ıncı trafik akış şekli kümesini temsil etmektedir. Bu durumda, *BKM(i,j,k)*, geçmişte görülen örnekler arasından kaç tanesinin C_D^i, C_T^j and C_F^k kümelerinde aynı anda yer aldığını, toplam eleman sayısıyla (N) normalize ederek kaydetmektedir. Bu değer, aslında C_D^i, C_T^j and C_F^k kümelerinin birlikte ortaya çıkma ihtimalini, yani bileşik olasılığını tutmaktadır. Bu bilgi ışığında, Eşitlik (2.3)'de tanımlanan, yol tipi ve trafik akışına koşullandırılmış sürüş skorlama yaklaşımı, BKM yapısı cinsinden yazılabilir. Eşitlik (2.5) bunu belirtmektedir:

$$Skor(i,j,k) = \frac{BKM(i,j,k)}{\sum_{i} BKM(i,j,k)} \tag{2.5}$$

Eşitlik (2.5), BKM'nin doğrudan skorlama için nasıl kullanabileceğini göstermektedir. Şekil 2.5, BKM'nin nasıl yaratılması gerektiğini görselleştirmektedir. Buna göre, geçmiş GPS ve CAN Bus kayıtlarından oluşan veri seti tüm açılardan kümelenmiş, daha sonra kümeler arası paylaşılan eleman sayısı bilgisi kullanılarak, ilgili BKM endeksleri doldurulmuştur. Sisteme yeni bir örnek geldiğinde, bu örneğin BKM'de denk geldiği pozisyon kolaylıkla saptanabilir ve skorlama yapılabilir.



Şekil 2.5: BKM'nin üretimi.

Bulunduğumuz noktada skorlama hakkında son bir problem daha bulunmaktadır. Bu da, sürüşlerin ne şekilde temsil edileceğidir. Çalışmamızda, sürücüleri değerlendirme ziyade, kilometre-bazında tecrübe edilen sürüşü skorlamak daha doğru bulunmuştur. Bunun nedeni, özellikle çalışmamıza konu olan halk otobüslerinde, o an aracı kullanan şoföre dair bir bilgi olmaması ve şoför değişikliklerinin de CAN Bus verisine yansımamasıdır. Bu bilginin eksikliği nedeniyle, kişi bazında çalışılmamıştır ve şoförden bağımsız, gözlenen kilometre-başı sürüş değerlendirmesi esas alınmıştır. Bunun için, aracın hareket ettiği her bir kilometre için GPS verileri ve CAN Bus

verileri çıkartılmış, GPS verileri kullanılarak yol tipine kümelemesi, CAN Bus ile iser sürüş tipi ve trafik akış tipi öznitelikleri elde edilmiştir. Ardından, daha önce açıklandığı gibi, BKM kullanılarak skorlama yapılabilmiştir. Bu işlem her kilometre için tekrarlanmıştır. Bu işlem yapılırken, aracın çok yavaş veya çok kesikli hareket ettiği aralıklar dışarıda tutulmuş, böylece aykırı örneklerin analize zarar vermesinin önüne geçilmiştir.

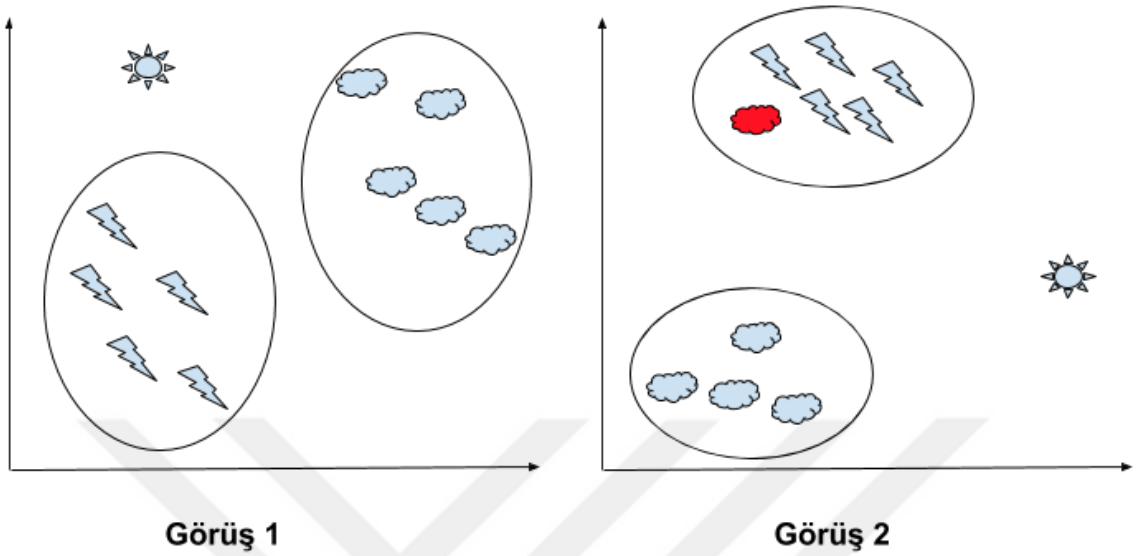
2.6.6 Benzer yaklaşımlar

Verinin D, T ve F alanında nasıl kümelendiğinin açıklandığı 3. bölüme geçmeden önce, skorlama yapmak için kullandığımız olasılıksal yaklaşıma benzer çok-bakışlı kümeleme-temelli modellerin literatürdeki kullanımlarından bahsetmenin yararlı olacağı kanaatindeyiz.

Çok-görüşlü öğrenme yaklaşımları literatürde genellikle, birbirleriyle karşılaştırılması anlamsal olarak makul olmayan öznitelik alt setleri üzerinde çalışıldığı zaman uygulanır. Örneğin bir objeye dair ses ve görüntü bilgiları aynı anda mevcutsa, bunları aynı öznitelik setinde tutmak çok akıllıca olmayacaktır. Şayet, öğrenme modelleri bir şekilde hem görüntüdeki piksel değerlerini, hem de ses sinyalinin genlik değerlerini kombine etmeye çalışacaktır. Bu yaklaşım fiziksel olarak mantıklı değildir. Bunun önüne geçmek için, birbirleriyle fiziksel olarak uyuşmayan modalitelere ait öznitelikler farklı öznitelik setlerinde tutularak öğrenme başarılabilir. Elde edilen birden fazla sonuç, göreve bağlı olarak ortalama alınarak (regresyon) yada çoğunlukla oylama (sınıflandırma) yapılarak kombine edilebilir.

Bu yaklaşım, anomali tespitinde de kullanılabilir. Özellikle, tek görüşten ortaya çıkartılamayan anomali örüntüleri, iki veya daha fazla görüş kombine edildiğinde rahatlıkla tespit edilebilir. Şekil 2.6, bu duruma bir örnek sunmaktadır. Burada, aynı veri örnekleri, iki farklı görüşten kümelenmişlerdir. Sadece birinci görüş kullanıldığında, veri üzerinde birbirinden ayrık, ⚡ ve ☁ sembollerinden oluşan iki küme olduğu ve ☼ objesinin bu sınıfların dışında kalan bir aykırı değer olduğu düşünülecektir. Ancak analize ikinci bir görüş eklersek, ☼ objeninin yine iki kümenin dışında, ikisine de benzer uzaklıkta olduğunu tespit edebiliriz. Bu durum bize, ☼ objesinin aslında bir aykırı bir değer değil de, yalnızca az sayıda gözlemlenmiş farklı bir sınıfa ait bir örnek olabileceğini göstermektedir. Buna ek olarak, önceki görüşte herhangi bir problem görünmeyen bir ☁ objesi (kırmızı ile

renklendirilmiş), ikinci görüşte kendi sınıfından sapıp, ⚡ objeleri ile bir arada kümelenmiştir. Bu durum, bu objenin anormal bir davranışa sahip olduğunu göstermektedir.



Şekil 2.6: İki görüşlü anomali tespiti.

Çok-görüşlü anomali tespiti, bu şekilde, farklı objelerin, görüşler arasında ne şekilde davranış değiştirdiğini inceleyerek, anormal durumları tahlil etmeye çalışan bir metodolojiler bütünüdür. Literatürde, bu yaklaşımın pek çok örneğine rastlanabilir [28], [29]. Bizim çalışmamızda da aslında yapılan oldukça benzerdir. Şayet, temel beklentimiz, aynı trafik akış ve yol tipi kümelerine giren bir sürüş tecrübesinin, sürüş stili açısından da aynı kümelere girmesidir. Bu şekilde hem daha önce açıklanan, koşullara bağlı değişen sürüş normu beklentisini karşılamak, hem de özellikle birbirlerinden farklı doğalara sahip CAN Bus ve GPS verilerini farklı öznitelik setlerinde tutmak mümkün olmuştur.

3. VERİNİN FARKLI AÇILARDAN KÜMELENMESİ

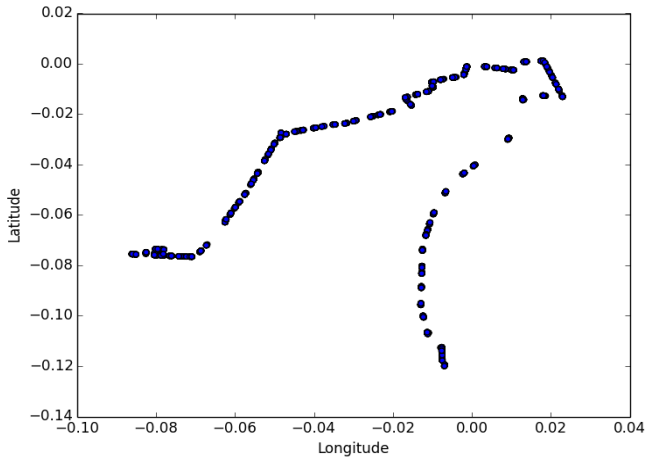
3.1 Bölüm İçeriği ve Amacı

Bu bölümde, verinin yol tipleri, trafik akış tipleri ve sürüş stilleri temsil edilecek şekilde işlenmesi ve değişkenlerin çeşitli yöntemler kullanılarak ayrıklaştırılması işlemi okuyuculara sunulacaktır.

3.2 Yol Tipi Kümeleme

Son yıllarda GPS verisi toplama özelliğine sahip cihazların maliyetlerinin azalmasıyla ciddi bir lokalizasyon verisi birikimi sağlamıştır. Bu zengin veri havuzu, aktivite tanıma, gezi tavsiyesi, lokasyon bazlı reklamcılık gibi uygulamalarda kullanılmaktadır. Gezinge kümeleme metodları, bizim çalışmamızın da en önemli bileşenlerinden olmakla birlikte, taşıdığı zengin bilgi haznesi nedeniyle, davranışsal çıkarımlarda bulunmak isteyen araştırmacıların en ilgi duyduğu konulardan biridir [30].

Gezinge, lokasyon bilgisinin zaman içerisindeki evrimini gösteren bir zaman serisidir. Burada lokasyon bilgileri, genellikle GPS kayıtlarıdır. Şekil 3.1'de GPS verisi üzerinden elde edilmiş bir gezinge görünmektedir. Burada her bir nokta, kayıt alınan farklı bir noktayı temsil etmektedir.



Şekil 3.1: GPS'den elde edilmiş iki-boyutlu bir gezinge.

Gezinge verisi, başta gürültü olmak üzere, pek çok problemlerden muzdariptir. Gelecek bölümde, çalışmamızda gezinge verilerine dair yaşadığımız problemler ve bunları nasıl çözdüğümüz işlenecektir.

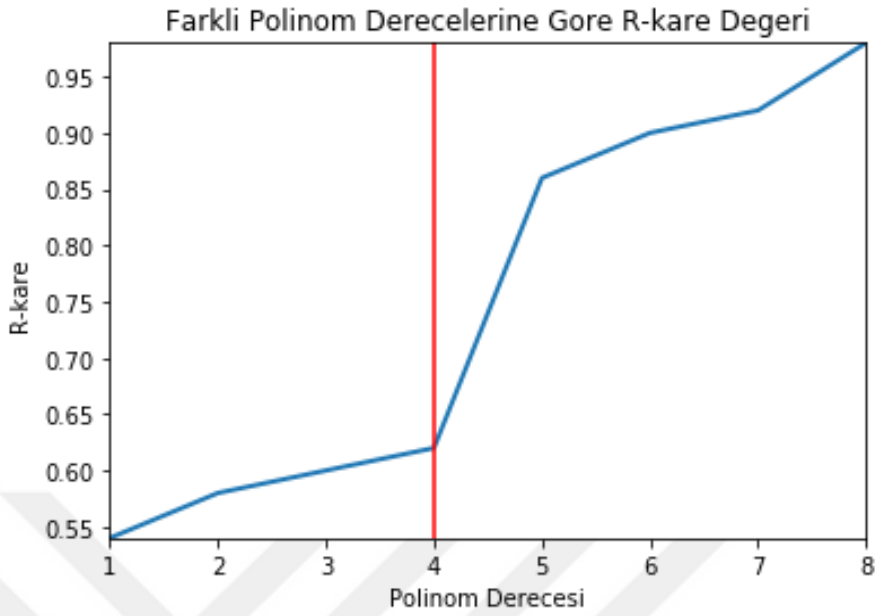
3.2.1 Gezinge verilerinin ön işlenmesi

GPS, dolayısıyla gezinge verilerindeki en büyük problem, sinyalin çoğu zaman yüksek gürültüden muzdarip olmasıdır. Bu durum GPS alıcısının uzaydaki konumu ile ilişkilidir. Örneğin, yüksek binalarla çevrili bir bölgede, elimizdeki GPS kayıtlarında birkaç yüz metreye kadar sapmalar ile karşılaşabiliriz. Aynı şekilde, kırsal bölgelerde de GPS sinyalinin zayıflığı nedeniyle gürültü etkisinin kuvvetlendiği görülmüştür. Benzer durum, kullanılan alıcı özellikleri ve coğrafi özellikler tarafından da yaratılabilir. Bu gürültü, şayet elenmediği takdirde, ardından gelecek adımlardan sağlıklı sonuçlar almayı engelleyecektir. Çalışmamızda lokasyon verisinin yol tipi kümelemesi açısından önemini düşünürsek, bu gürültüden kurtulmanın oldukça kritik olduğu görülecektir.

Gezinge verilerinde veya daha geniş anlamda GPS sinyallerinde gürültüden kurtulmak için uygulanabilecek çok sayıda strateji vardır. Bunlardan ilki, elimizdeki gezinge verisini bir hareketli ortalama süzgecinden geçirmektir. Örneğin verinin sabit uzunlukta bir kayan pencere içerisindeki ortalama değerini kullanarak, gezinge verileri yumuşatılabilir. Eğer GPS verisindeki zıplamalar çok güçlüyse, ortalama değer yerine, bu tip durumlarda daha iyi çalışan medyan süzgeçler de kullanılabilir. Eğer gezingeler az sayıda örnek içeriyorlarsa, Kalman süzgeci gibi, hareket bilgisini de hesaba katan daha kompleks yöntemlere başvurulabilir.

Çalışmamızda bahsi geçen yöntemler yerine, en küçük kareler-bazlı bir yaklaşımı benimsedik. Buna göre bir gezinge, kendisi üzerine en küçük ortalama karesel hatayı verecek şekilde oturtulmuş yüksek dereceden bir polinom ile temsil edilebilir. Bu polinomun derecesi arttırıldıkça, ortalama karesel hata sıfıra yakınsayacaktır; ancak temel amaç bu işlemi en küçük dereceli polinom ile yaparak, veri içerisindeki gürültüleri modellemeden, sadece genel trendi yakalamaktır. Burada önemli noktalardan bir tanesi, bu polinomun optimal derecesini tespit edebilmektir. Bunun için bir altın standart olmamasına rağmen, R-kare metriği kullanılarak bu işlem yapılabilir. R-kare, elimizdeki polinomun verideki toplam varyansın ne kadarını modelleyebildiğini gösterir. R-kare değeri en fazla 1; varyansın negatif olmaması nedeniyle en az da 0 olabilir. Veri setindeki bütün gezinge verilerine çeşitli derecede

polinomlar oturtularak R-kare değerlerini hesaplanmış ve Şekil 3.2'deki sonuçlar elde edilmiştir.



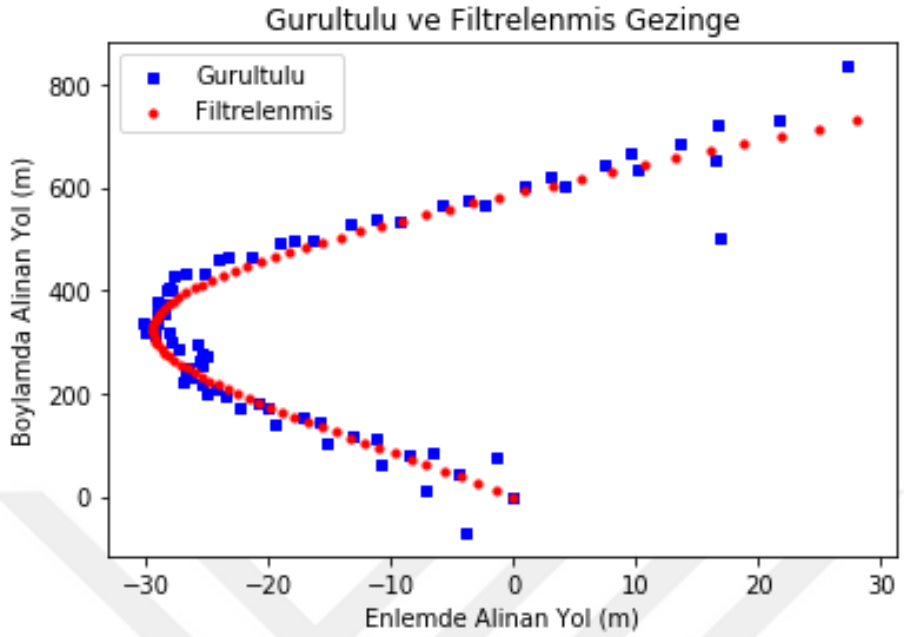
Şekil 3.2: Farklı polinom derecelerine göre R-kare değerleri ve eşik değeri (kırmızı çizgi).

Şekil 3.2'de görüleceği üzere, 1. dereceden bir polinom için yaklaşık 0.55'den başlayan R-kare değeri, 8. dereceden bir polinom kullanıldığında, 0.97 seviyesine çıkmıştır. Burada, R-kare değerinin zıplama yaptığı nokta, yani 4. derece, optimal değer olarak değerlendirilmiştir. Bunun üzerindeki dereceler için, elde edilen gezinge temsilinin gürültüyü modellediği ve bu nedenle dikkate alınmaması gerektiği kanaatine varılmıştır. İlerleyen aşamalarda orjinal gezingeler yerine, sadece bu temsil gezingeler kullanılmıştır. Aşağıda, gürültülü bir gezinge ve aynı gezingenin 4. dereceden bir polinom ile yumuşatılmış hali gösterilmektedir.

Şekil 3.3'un açıkça gösterdiği üzere, gezingenin genel geometrisi korunmuş olmakla beraber, enlemde 20. metre, boylamda ise yaklaşık 500. metrede bulunan aykırı değerler elenmiştir. Bu sonuç, metodolojimizin olması gerektiği gibi işlediğini göstermektedir.

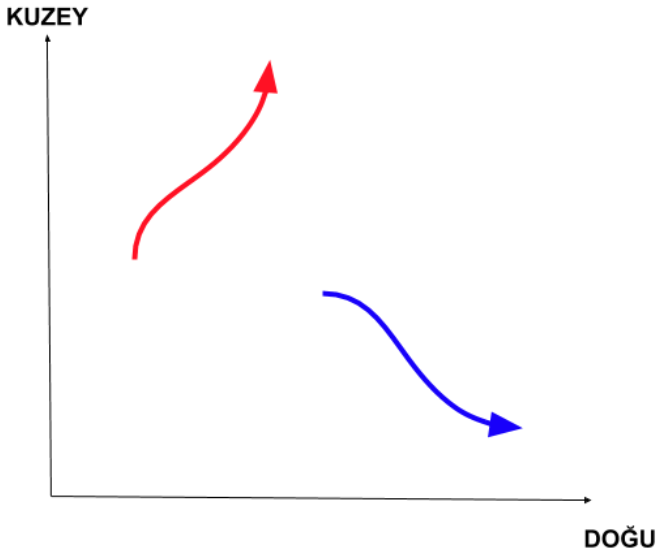
Gezingelerin kümelenmesinin önündeki bir diğer engel ise, uzaydaki hizasızlık durumudur. Bu durum Şekil 3.4'da kolayca görülebilir. Bu şemada, kırmızı ve mavi renkler ile boyanmış gezingeler özdeştir. Ancak, başlangıç noktalarının ve uzaydaki yönelimlerinin farklı olması nedeniyle, kartezyen sistemde oldukça farklı şekilde

temsil edilmektedirler. Bu nedenle, bu iki gezinge arasındaki mesafe oldukça geniş olacak ve aynı kümeye girmeleri ihtimali azalacaktır.



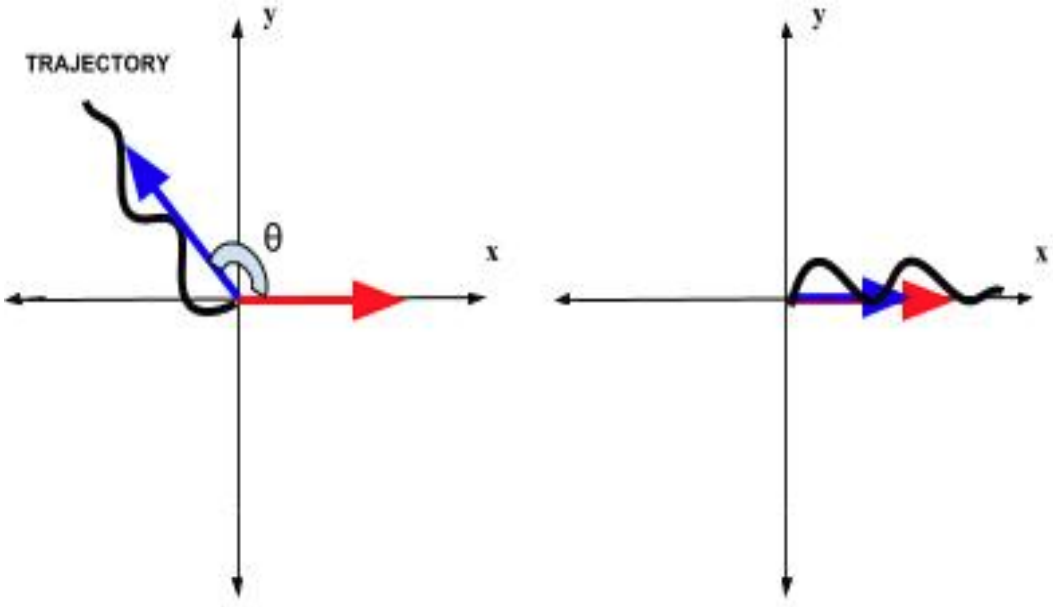
Şekil 3.3: Gürültülü ve filtrelenmiş gezinge örneği.

Benzer gezingelerin uzaydaki konumlarından bağımsız olarak aynı kümelere düşebilmelerini sağlayabilmek için, kümeleme işleminden önce, bir hizalama ön işlemesi yapılmalıdır. Bu hizalama, yapılan doğru hiza tanımına göre pek çok şekilde başarılabilir. Örneğin, iki gezingenin ilk birkaç noktasının aralarındaki mesafenin minimize edilecek şekilde üst üste getirilmesi, hem başlangıç noktası hem de uzaydaki yönelim problemini çözebilir. Ancak, az sayıda nokta üzerinden yapılan işlemler genellikle gürültüye karşı savunmasızdır. Bunun yerine, hizalama esnasında bütün gezinge endekslerini hesaba katacak bir yaklaşımın çok daha başarılı olacağı düşünülmüştür. Bu noktada, medikal görüntülemede sıklıkla uygulanan Temel Bileşen Analizi (TBA) temelli katı gövde hizalaması uygulanması uygun görülmüştür [31]. Bu teknik, objelerin (gezingelerin) en güçlü temsil edildikleri eksenler üzerinden hizalanmasını tercih etmektedir. Bir diğer deyişle, gezingeler, üzerlerindeki varyansın en yüksek olduğu yönde hizalanırlar. TBA, veri oto-korelasyon matrisinin özvektörlerini hesaplayarak, veri içindeki varyansın içerildiği, dik eksenleri bulmak için özelleşmiş bir metoddur. Bu eksenlerin her birine temel bileşen (TB) denir ve bunlar en çok varyans içerenden en az içerene doğru sıralanırlar. Örneğin, bir gezinge verisi için TB1, varyansın en güçlü olduğu istikamettir.



Şekil 3.4: Uzayda hizasızlık.

Buna göre, TBA kullanarak, gezingelerin en yüksek enerji içerdiği istikametler şu şekilde bulunabilir: (1) Tüm gezingelerin başlangıç noktası (0,0) olarak sabitlenir, (2) Gezingelerin hepsinin üzerinde toplanacağı bir referans vektörü bulunur, (3) Her bir gezinge için TB1 hesaplanır, (4) Bu TB1'lerin referans vektörü ile arasındaki açı bulunur ve (5) Gezingelerin her biri bu açı kadar döndürülerek referans üzerine yansıtılır. Bu işlem aşağıdaki resimde görselleştirilmiştir (Şekil 3.5).



Şekil 3.5: Gezingeye (siyah eğri) ait TB1 (mavi vektör) vektörünün referans (kırmızı vektör) üzerine doğru döndürülmesi.

Bu rotasyon işleminin basamakları detaylı incelenecek olursa:

1) TB1 ve referans vektör arasındaki açı, θ, bu iki vektörün arasındaki kosinüs uzaklığının bulunması ve ardından ark kosinüs fonksiyonu ile bu uzaklığın açıya çevrilmesi ile bulunabilir. Burada dikkat edilmesi gereken nokta, TB1'in aynı eksenin iki tarafına doğru da olabileceğidir. Örnek vermek gerekirse, eğer TB1 ekseni y=x ise, elde edilecek sonuç y=x veya y=-x olabilir. Bu nedenle işlemler esnasında iki durum da göz önünde bulundurulmalı ve doğru eksen tespit edilmelidir. Çalışmamızda, tüm işlemler bu iki yön için de yapılmış ve sonuçlar elde edilmiş, bunlar arasından referans vektörü ile en çok ilintiye sahip olan doğru kabul edilmiştir.
2) θ açısı bulunduktan sonra, rotasyon matrisi Eşitlik (3.1)'de gösterildiği şekilde bulunmuştur [32]. Gezingenin rotasyonu, gezinge ile bu matrisin çarpımı şeklinde tanımlanabilir.

$$R = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{3.1}$$

Gezingelerin ön işlenmesi tamamlandıktan sonra, kümeleme aşamasında geçilebilir.

3.2.2 Hizalanmış gezingelerin kümelenmesi

TBA prosedürü ile bütün gezinge verileri hizalanmıştır. Şekil 2.12, gezingelerin, bu işlemden önce ve sonraki pozisyonlarını görselleştirmektedir.

(a) (b)

Şekil 3.6: (a) Herhangi bir işleme tabi tutulmamış gezinge verileri, (b) TBA ile x ekseni üzerine doğru döndürülmüş gezinge verileri.

Hizalanmış gezingeler, bir sonraki aşamada birbirlerine benzeyen yol tiplerini saptayabilmek amacıyla kümeleme işlemine tabi tutulmuştur. Bu noktada, Bölüm

2.6'da detaylı açıklanmış kümeleme modellerinden yararlanılmıştır (KMEANS, BIRCH, DBSCAN, Spektral Kümeleme). Ancak, kümeleme işlemini gerçekleştirmek için hala eksik olan bir bilgi bulunmaktadır: Benzerlik metriği!

Kümelemenin temelinde yatan, birbirlerine benzeyen örneklerin bir araya toplanması, etkili bir mesafe ölçütü ihtiyacını beraberinde getirmektedir. Literatürde, bu işlem için pek çok farklı metrik kullanılmaktadır [33]. Bu ölçütlerden en sık kullanılanları aşağıdaki gibidir.

3.2.2.1 Öklid mesafesi

Öklid Mesafesi, kümeleme problemlerinde en sık kullanılan uzaklık ölçütüdür. İki eş-örnekli gezenge T_i ve T_j arasındaki Öklid Mesafesi, Eşitlik (3.2) ile bulunabilir.

$$D_E(T_i, T_j) = \frac{1}{N}\sum_{n=1}^{N} \left|T_i^n - T_j^n\right| \tag{3.2}$$

Burada gezingelerin sahip olduğu toplam eleman sayısı N, ve|. | vektör normudur.

3.2.2.2 Hausdorf mesafesi

Hausdorff Mesafesi, nokta-bazlı bir uzaklık metriğidir. Hausdorff, iki gezinge arasındaki en ekstrem durumlar üzerinden mesafe hesaplar. Bunu, bir gezinge üzerindeki her nokta için, diğer gezinge üzerindeki en uzak noktanın mesafesini bulur. Eğer gezingeler benzer ise, bu değerin belirli bir seviyenin üzerine çıkamaması beklenir. Haussdorf mesafesi Eşitlik (3.3)'de gösterildiği şekilde bulunur [34], [35].

$$\begin{aligned} D_H(T_i, T_j) &= max\left(H(T_i, T_j), H(T_j, T_i)\right) \\ H(T_i, T_j) &= \max_{p \in T_i}\left(\min_{q \in T_j} dist(p, q)\right) \end{aligned} \tag{3.3}$$

3.2.2.3 En uzun ortak altdizi mesafesi

En Uzun Ortak Altdizi (EUOA) Mesafesi bir uzaklık ölçütünden ziyade, bir benzerlik metriğidir. EUOA'nın temel varsayımı, iki gezingenin benzerliği, bir bütün olarak değil, paylaştıkları ortak örüntülerin üzerinden tanımlanmaları gerektiğidir. Eğer iki gezinge birbirlerine gerçekten de benzerlerse, bu paylaşılan altdizi uzunluğu artacak ve özdeşlik durumu varsa bu gezinge uzunluğuna eşit olacaktır.

EUOA mesafesi, altdizi benzerliklerini hesaplarken ılıman davranır. Bir gezinge üzerindeki nokta, diğer gezingedeki aynı zamanı paylaştığı nokta ile birebir eşleşmek durumunda değildir. Bu eşleşme uzayda, kullanıcı tarafından belirlenen bir marjin

içerisinde olduğu takdirde, yeterli olmaktadır. İki gezinge arasındaki, ardışık olarak, diğer gezinge tarafından paylaşılan (eşleşme olan) maksimum nokta sayısı bulunur ve bu değer EUOA Benzerliği olarak hesaplanır. Eşitlik (3.4)'de bu benzerlik formülize edilmiştir [35].

$$EUOA(T_i, T_j) = \begin{cases} EUOA(T_{i-1}, T_{j-1}) + 1 \\ \Delta_X \leq \sigma, \Delta_Y \leq \epsilon \end{cases} \quad (3.4)$$

Burada ε, y-alanındaki; σ ise x-alanındaki marjini belirtmektedir.

EUOA, daha önce belirtildiği gibi bir uzaklık mesafesi değil, bir benzerlik ölçütüdür. Yani, iki örnek arasındaki benzerlik yüksekken uzaklık ölçütleri 0'a yakın bir değer verirken, EUOA ise maksimum değerine yakın bir sonuç verecektir. Benzerlik ölçütünü, mesafe değerine çevirmek için pek çok farklı yaklaşım bulunmakla beraber, çalışmamızda *Mesafe=Maksimum Değer-Benzerlik* şeklinde basit bir yaklaşımda bulunmayı tercih ediyoruz.

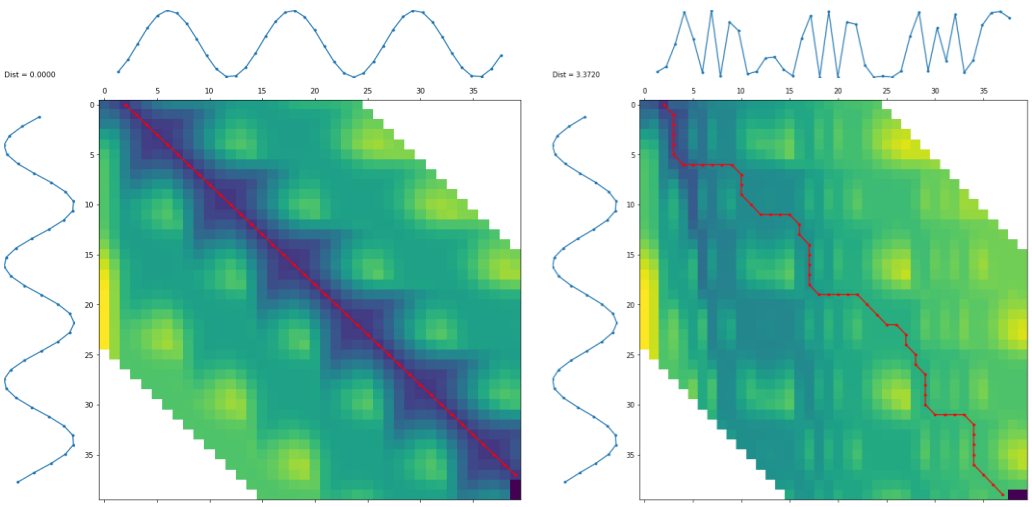
3.2.2.4 Dinamik zaman bükülmesi mesafesi

Dinamik Zaman Bükülmesi (DZB), zaman serileri arasındaki benzerliği ölçmek ve zamanda deforme olmuş serileri eşleştirmek için kullanılan, doğrusal olmayan yaklaşımlar içeren bir tekniktir. DZB metodolojisinde, öncelikle *NXN bir* bükme matrisi bulunur. Bu matris, iki gezingenin bütün nokta kombinasyonları arasındaki Öklid mesafelerini tutmaktadır. Bu matrisi, (0,0) noktasından başlayarak, (N,N) noktasında tamamlayan bütün yollar hesaplanır. Bunlar arasından en az maliyete sahip olan yol Dinamik Zaman Bükmesi yolu olarak bulunur. Şekil 2.13 bunu betimlemektedir [36].

DZB, Şekil 3.7'de görüldüğü üzere, optimal yani en az maliyetli yolu bulmak için, zaman serilerini manipüle etmeyi öngörür. En düşük maliyetli yolun maliyeti, DZB Mesafesi olarak bulunur. Matematiksel olarak yapılan işlem Eşitlik (3.5)'de gösterilmektedir.

$$C_p(T_i, T_j) = \sum_{k=1}^{N} C(T_i^k, T_j^k) \quad (3.5)$$

$$DTW(T_i, T_j) = min\,(C_p(T_i, T_j))$$



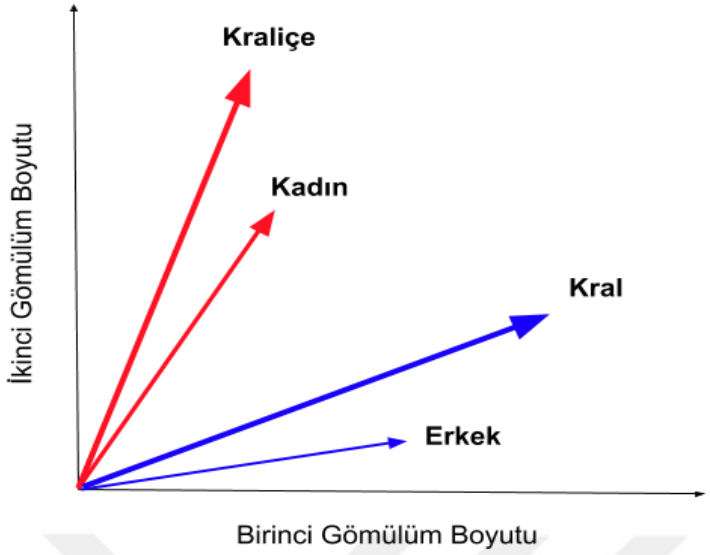
(a) (b)

Şekil 3.7: (a) Özdeş iki gezinge için, (b) birbirlerine benzemeyen iki gezinge için benzerlik matrisleri.

Çalışmamızda, bu uzaklık ölçütlerinden herhangi birini seçmek yerine, her birini ayrı ayrı kullanmayı tercih ettik. Buna ek olarak, her aşamada kullanılan kümeleme metodlarını da değiştirdik. Böylece 4 kümeleme metodu ve 4 uzaklık ölçütünün her bir kombinasyonu için kümeleme yapılmış oldu. Bunlar arasından en başarılı kombinasyon seçilerek, kullandığımız veri setinde en iyi çalışan ayarları bulmak hem de bunları başarılı yapan faktörleri incelemek mümkün hale gelmiştir. Bu sonuçlar ve çıkarımlar gelecek bölümde irdelenecektir.

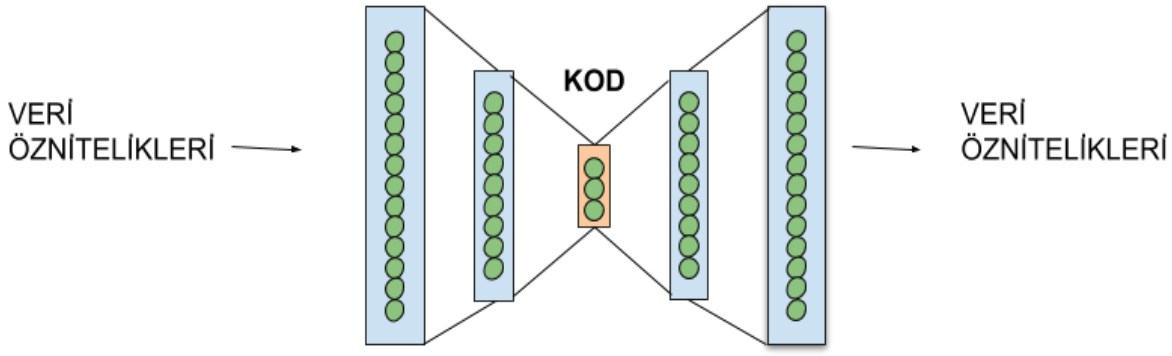
3.2.3 Gezingelerin düşük boyutlu temsiller haline getirilmesi

Bir verinin düşük-boyutlu temsili, ya da gömülümü, hem verinin düşük boyutta muhafaza edilmesini, hem de anlamsal olarak zenginleşmesini sağlayabilir. Bu temsiller, özellikle Doğal Dil İşleme alanında sıklıkla kullanılmaktadır. Burada, kelimelerin işlenmemiş temsilleri herhangi bir anlamsallık taşımazken, kelime gömülümleri elde edildiğinde, anlamsal olarak benzer olan kelimelerin gömülümlerinin de daha ilinti olduğu görülmektedir [37]. Bu durum Şekil 3.8'de gösterilmektedir. 'Kraliçe' ve 'Kadın' kelimeleri birbirlerine oldukça yakınken; aynı durum 'Kral' ve 'Erkek' için de görülmektedir.



Şekil 3.8: Kelimelerin düşük boyutlu gömülümleri.

Bu tip temsilleri yaratabilmemizi sağlayan pek çok teknik bulunmaktadır. Bunların arasından otokodlayıcı adı verilen özelleşmiş bir YSA mimarisi en sık kullanılan yöntemlerden biridir. Bir otokodlayıcı, girdi olarak aldığı veriyi, çıktıda tekrar üretmeye çalışır. Bu işlem gereksiz gibi görünse de, otokodlayıcının düşük sinir ünitesi içeren ara katmanlarında verinin anlamca zengin gömülümleri elde edilebilir [38]. Örnek bir otokodlayıcı mimarisi çizecek olursak (Şekil 3.9):



Şekil 3.9: Bir otokodlayıcı mimarisi.

Otokodlayıcının her bir ünitesi, bir sinir hücresi (nöron) olarak adlandırılır. Bu nöronlar, girdilerinin doğrusal bir işlemden geçirdikten sonra, bu işlemin sonucunu

sigmoid gibi doğrusal olmayan bir fonksiyondan geçirirler. Bir sonraki aşağada, önceki tabakadaki nöronlardan alınan çıktılar tekrar aynı işleme tabi tutulur. Böylece, oldukça kompleks, doğrusal olmayan fonksiyonlar modellenebilir. Otokodlayıcılar, diğer YSA mimarileri gibi, geri yayılım isimli bir gradyan-bazlı optimizasyon algoritması ile eğitilirler.

Veri kümeleme esnasında orijinal veri temsilleri yerine gömülümlerini kullanmak çoğu durumda daha başarılı sonuçlar vermiştir [39]. Bu yaklaşım, literatürde pek çok problemde kullanılmış olsa da, henüz gezinge verileri üzerinde uygulanmamıştır. Bu nedenle, çalışmamızın bir diğer ayağında, gezingeleri daha önce bahsi geçen yöntemler dışında, bir de gömülümler vasıtasıyla kümelemeye çalıştık. Bu noktada vanilya otokodlayıcı denilen (Şekil 3.9'de sunulan mimari) geleneksel yapılar yerine, zamansal ilişkileri modelleme çalışan Uzun Kısa Dönem Hafıza Ağları kullanılmıştır. Bu ağlar, geçmişten gelen verinin ne kadarını tutmak, ne kadarını atmak ve o an içerisinde gelen verinin ne kadarını bu geçmiş veriye eklemek gerektiğine karar verme kabiliyeti olan özel ünitelerle donatılmıştır [40].

Gömülüm-bazlı kümeleme esnasında dikkat edilmesi gereken en önemli husus ise yine uzaklık metrikleridir. Gezinge gömülümleri, Gömülümler Öklid uzayında tanımlandığı için, gezinge kümeleme için daha önce tanımlanmış metrikleri kullanmak gereksizdir. Bunun yerine, gömülümler için sıkça kullanılan iki metrik, Öklid Uzaklığı ve Kosinüs Uzaklığı kullanılmıştır. Bunlardan ilki önceki bölümde tanımlandığı şekilde uygulanmış olmakla beraber, Kosinüs Uzaklığı, iki vektör arasındaki açının kosinüs değerinin 1'den çıkartılması ile elde edilmektedir.

3.3 Trafik Akış Tipi Kümeleme

Trafiğin akış şeklinin tespit edilmesi, skorlama mekanizması için hayati öneme sahiptir. Şayet, trafiğin akış paternini anlamak, sürüş normlarının tanımlanmasının ön şartıdır. Yoğun seyreden bir trafikteki gaz-pedal hareketleri, doğal olarak akıcı trafiktekilerinden çok daha farklı olacaktır. Dolayısıyla, bu akış şemasının tespiti başarıyla tamamlanmalıdır. Literatürde bu anlamda az sayıda da olsa yayınlar bulunmaktadır [41]. Çalışmamızda, trafik yoğunluğunu kümeleyebilmek adına, üç farklı öznitelik kullandık. Bunlar;

- Dur/Kalk Sayısı

- Aracın zamanının yüzde kaçında hareket etmemesi (motor açık)
- Hız istatistikleri (ortalama ve standart sapma)

Bu öznitelikler, çok geniş bir öznitelik havuzu içerisinden trafik durumu bilgisiyle en ilintili olanlar cinsinden seçilmiştir. Seçim aşamasında, elimizdeki küçük bir etiketli veri seti üzerinde bulunan yoğun trafik/hafif trafik girdisi esas alınmıştır. Bu etiketler bir zaman serisi haline getirilmiş ve her bir öznitelik vektörü ile ilintisine (Pearson korelasyon) bakılmıştır. Bunlar arasından en yüksek ilintiye sahip özniteliklerin yüzde beşi Tablo 1'de gösterilmektedir.

Çizelge 3.1: Trafik akış tipi kümelemesinde kullanılan öznitelikler.

Öznitelik	**Dur-Kalk**	**Hareketsizlik Oranı**	**Hız Ortalaması**	**Hız Standart Sapması**
Korelasyon	0.523	0.460	0.437	0.392

Bu öznitelikler, CAN Bus verileri işlenerek elde edilmiş, bu işlem her bir gezinge parçası için tekrarlanmıştır. Kümeleme aşamasında ise, KMEANS algoritması kullanılmıştır.

3.4 Sürüş Tipi Kümeleme

Sürüş stili karakteristiğini ortaya koyan özniteliklere karar vermek oldukça zorlu bir süreçtir. Bu alandaki geçmiş çalışmalar genellikle, sabit bir GPS porsiyonu içerisinde, hız, ivme ve hızın yüksek dereceden türevleri ve bunların istatistiklerini (ortalama değer, standart sapma, kartiller gibi), bazen de yakıt tüketimi, motor devri istatistiklerini kullanmaktadırlar. Bu özniteliklerin neredeyse hepsinin benzer mekanizmalar tarafından üretildiğini düşünmek oldukça doğaldır. Şayet motor devri bilgisi; yakıt tüketimini, aracın hızını ve ivmesini doğrudan etkileyecektir. Öznitelikler arası ilintinin çok güçlü olduğu bu tip durumlarda, orijinal öznitelik setini kullanmak yerine, daha önceden de bahsedilen düşük boyutlu gömülümleri kullanmak, hem bilgi olarak daha zengin hem de daha küçük veri temsilleriyle çalışmayı sağlayacaktır. Bu durum, araştırmacıları, sürüş stili özniteliklerini bir otokodlayıcı yardımıyla sıkıştırarak kullanmaya teşvik etmiştir.

Çalışmamızda, biz de aynı yaklaşımı benimsedik; ancak öznitelik olarak öncekilerden biraz daha farklı bir setten yararlandık. Bu seti elde etmek için Bölüm

3.3'deki gibi etiketli küçük veri setinden yararlandık. Bu veri seti 10 adet şoförün agresif ve agresif olmayan sürüş tecrübelerinden oluşmaktadır. Etiketleme alan uzmanları tarafından sahada yapılmıştır. Her şoförden, 10 dakika boyunca mümkün olduğunca agresif, diğer 10 dakika boyunca ise agresif olmayan şekilde sürmeleri istenmiştir. Bu toplam 200 dakikalık verinin, %60'ı sürüş stili ile ilişkili özniteliklerin saptanması için ayrılmış, geri kalan bölüm ise sonuçların doğrulanması amacıyla saklanmıştır. Trafik akış tipi öznitelik çıkarma aşamasında olduğu gibi, burada da çok sayıda aday öznitelik arasından, hedef değişkenle en ilintili olanlar saptanmıştır. Bu çalışma, aralarında daha önce literatürde kullanılmayan bazı özelliklerin de bulunduğu, şu öznitelik setini ortaya koymuştur:

- Gaz/Fren Pedal Pozisyon Zaman Serilerinin Türevlerinin İstatistikleri: Her bir sürüş tecrübesi için gaz ve fren pedal pozisyonu sinyallerinin türevleri alınmış; ortalama değer, standart sapma ve %25, %50 ve %75'lik kartil değerleri seçilmiştir. Bu değer, sürücünün araç üzerindeki kontrolünün ne kadar sert/yumuşak olacağına dair bilgi taşımaktadır.
- Motor Yükü İstatistikleri: CANBus verisi içerisinde yer alan motor yükü zaman serilerinin %25, %50 ve %75 kartilleri seçilmiştir. Bu, doğrudan sürücü ile ilgili olmamakla birlikte, aracın ne kadar makul şekilde kullanıldığını göstermektedir.
- Pedal Tekmeleme Sayısı: Sürücünün Gaz Pedalına tekme atma sayısı. Pedal tekmelemek kişisel araçlar da pek görülmese de, otobüslerde karşılaşılan bir durumdur ve sürücünün agresiflik seviyesinin bir ölçüsü olarak görülmektedir.
- İvmelenme Momentleri: CANBus üzerinden hız zaman serisi elde edildikten sonra, bunun ilk türevi alınarak ivmelenme zaman serisine ulaşılmıştır. Ardından, bu sinyalin 5. merkezi momentine kadar bütün momentleri hesaplanmış ve öznitelik vektörüne katılmıştır. Bu istatistikler, sürücünün hızlanma/yavaşlama karakteristiğini göstermektedir.

Bu işlemler, her bir N kilometrelik yol parçacığında sürücünün 19 öznitelik ile temsil edilmesini sağlaşılmıştır. Bu öznitelik vektörü, bir otokodlayıcı yardımıyla 6-boyutlu bir gömülüm haline getirilmiştir. Bu sıkıştırılmış boyut, eğitim verisi üzerinde seçilen bir alt set üzerinde denenmiş farklı gömülüm boyutu kombinasyonları arasından, en başarılı olanı olduğu için seçilmiştir. Bunu takip eden adımda, sürüş

stili gömülümleri Öklid mesafesi kullanan bir K-MEANS modeli ile kümelenmişlerdir.

4. SONUÇLAR VE TARTIŞMALAR

4.1 Bölüm İçeriği ve Amacı

Bu bölümde, öncelikle verinin farklı görüşlerden kümelenmesinin ne kadar başarıyla yapılabildiğini sunulacaktır. Bu noktada, farklı kümeleme metotları ve uzaklık metriklerinin başarıları tartışılacak, bunların verinin doğasına uygunlukları incelenecektir. Son olarak da, bütün skorlama metodolojisi bir test verisi üzerinde değerlendirilecektir.

4.2 Ortalama Silüet Katsayısı

Çalışmamız, hem kümeleme metodlarının hem de bu metotların yararlandıkları mesafe ölçütlerinin karşılaştırılmasını içermektedir. Ancak; literatürde gözetimsiz öğrenme algoritmalarının değerlendirilmesi için genel geçer bir yöntem bulunmamaktadır. Birçok çalışmada, etiketli verinin mevcut olması sayesinde, öğrenim gözetimsiz yapılmış olsa da, sonuçları gözetimli öğrenim metrikleri ile (hassasiyet) değerlendirmek mümkündür. Fakat, etiketli verinin bulunmadığı bu çalışmada, etiketsiz veriler üzerinde çalışabilen, örneklerin uzaydaki dağılımları üzerinden karar veren bir ölçüte ihtiyaç vardır. Bunu başarabilen az sayıdaki metoddan birisi Ortalama Silüet Katsayısıdır (OSK) [42]. OSK, Eşitlik (4.1)'de sunulmuştur.

$$OSK = \sum_{i} \frac{b_i - a_i}{max(a_i, b_i)} \tag{4.1}$$

Eşitlik (4.1)'de *i* toplam örnek sayısı; a_i, i'inci örneğin kendi kümesindeki diğer örneklere olan ortalama uzaklığı; b_i ise, i'inci örneğin kendisine en yakın rakip kümenin elemanlarına olan ortalama uzaklığıdır. OSK, tüm örnekler için, bu iki mesafe arasındaki ayrımı gösterir. Buna göre, OSK örnekler kendi kümeleriyle uyumlu hale geldikçe ve diğer kümelerden uzaklaştıkça, daha yüksek değer almaktadır. Bir uç durum olarak, OSK'nın 1 olması, tüm örneklerin kendi

kümelerinin ağırlık merkezinde yer alması, 0 olması ise, tüm örneklerin kümeler arası sınır hatlarında yer almalarının göstergesidir. Buradaki temel beklenti, bir örneğin kendi kümesiyle mümkün olduğunca uyumlu, çevredeki diğer kümelerle ile mümkün olduğunca farklı olmasıdır.
Çalışmamızda, kümeleme modeli ve mesafe ölçütü kombinasyonlarının başarılarını OSK kullanarak ölçeceğiz.

4.3 Gezinge Kümeleme Sonuçları

Veri seti, 2.5 kilometrelik parçalara ayrılmış, her parça için tanımlı GPS verileri kullanılarak gezingeler elde edilmiş ve bunlar daha önce belirtildiği şekilde, gürültüden temizlenmiş ve x-ekseni yönüne doğru döndürülmüştür. Bu şekilde kümelemeye hazır hale getirilmiş gezinge veri seti, 4 farklı kümeleme metodundan ve aynı sayıda farklı uzaklık metriğinden faydalanılarak kümelenmişlerdir. Bu kümeleme sonucunda başarı OSK değerleri ile ölçülmüş, bulunan sonuçlar Çizelge 4.1'de sunulmuştur.

Çizelge 4.1: Gezingeler üzerinde farklı kümeleme metodu ve mesafe ölçütü kombinasyonlarının OSK cinsinden başarıları.

Ölçüt/Model	**K-MEANS**	**BIRCH**	**DBSCAN**	**Spektral**
Öklit	0.206	0.195	-0.092	0.003
Hausdorff	0.096	0.201	-0.112	-0.004
EUOA	0.233	0.224	-0.050	0.075
DZB	**0.251**	0.222	-0.053	0.101

Çizelge 4.1 üzerinden gezinge kümeleme hakkında pek çok çıkarımda bulunabilir. Bunlardan ilki, K-MEANS ve BIRCH algoritmalarının başarılarıdır. Bu iki model de, konveks ve küresel kümeleme üzerine uzmanlaşmışken, rakipleri DBSCAN ve Spektral Kümeleme ise, herhangi bir şekle bağlı kalmadan, veri üzerindeki lokal örüntüler üzerinden hareket etmektedirler. DBSCAN ve Spektral Kümeleme algoritmaları, mevcut veri seti üzerinde, negatif veya sıfıra çok yakın OSK değerlerine sahip olmuşlardır ve bu, örneklerin çoğunun ya yanlış kümenin içerisinde

ya da yanlış küme ile doğru kümenin sınırında yer aldıklarını göstermektedir. Bu durumun arkasında yatan sebep, verinin homojen dağılımı olabilir. Homojen dağılım, gezinge tipleri arasındaki sınırların muğlak olmasından kaynaklıdır. Örneğin, sola sert bir dönüş gösteren bir gezinge, düz seyreden bir gezinge ile aslında bağlıdır; çünkü o aralıkta pek çok farklı gezinge yer alacaktır. Bu nedenle, lokal özellikler kümelemeyi yanlış yönlendireceklerdir. K-MEANS ve BIRCH, bu lokal özellikleri dikkate almadıkları için, daha başarılı olmuşlardır.

Çizelge 4.1'in sonuçlarının işaret ettiği bir diğer önemli nokta ise, EUOA ve DZB mesafe ökçütlerinin yakaladıkları yüksek başarıdır. Bu metrikler, nokta-bazlı çalışır ve her bir noktayı, diğer gezingedeki çok sayıda nokta ile eşleştirmeyi amaçlarlar. Bu sayede elde ettikleri değerler, Öklid gibi ölçütlere göre daha ılıman olacaktır. Buna ek olarak, DZB'nin gezingeler arasındaki ilişkiyi doğrusal olmayan bir şekilde modelleyebilmesi, onun başarını daha da yukarıya taşımıştır.

Çalışmamız, literatürde de sıklıkla işaret edildiği gibi, DZB ve EUOA ölçütlerinin ve K-MEANS ve BIRCH gibi kümeleme modellerinin, gezinge kümeleme üzerinde daha iyi çalıştıklarını göstermektedir. Aynı metodolojiyi kullanarak, gezinge gömülümlerinin daha iyi kümelenip kümelenmediğini görmek de mümkündür. Çizelge 4.2, GPS-bazlı gezinge gömülümlerinin yerine 10-boyutlu gömülümlerin kümelenmesi halinde karşılaşılacak OSK sonuçlarını sunmaktadır.

Çizelge 4.2: Gezinge gömülümleri üzerinde farklı kümeleme metodu ve mesafe ölçütü kombinasyonlarının OSK cinsinden başarıları.

Ölçüt/Model	**K-MEANS**	**BIRCH**	**DBSCAN**	**Spektral**
Öklit	**0.294**	0.266	0.064	-0.420
Kosinüs	0.197	0.141	0.045	-0.294

Çizelge 4.2'de görülebileceği üzere, gezinge gömülümleri, işlenmemiş öznitelik vektörlerine kıyasla daha başarılı bir şekilde kümelenebilmişlerdir. Bu durumun, otokodlayıcılarının veri içerisindeki örüntüleri genelleyebilme konusundaki başarılarından kaynaklandığı düşünülmüştür. Şayet otokodlayıcı, veriyi kendi boyutundan çok daha küçük boyutlu bir temsilden yaratmakla yükümlüdür. Bu işlemi yaparken, verinin içerisindeki en önemli örüntüleri saklayıp, detay bilgi ve gürültü

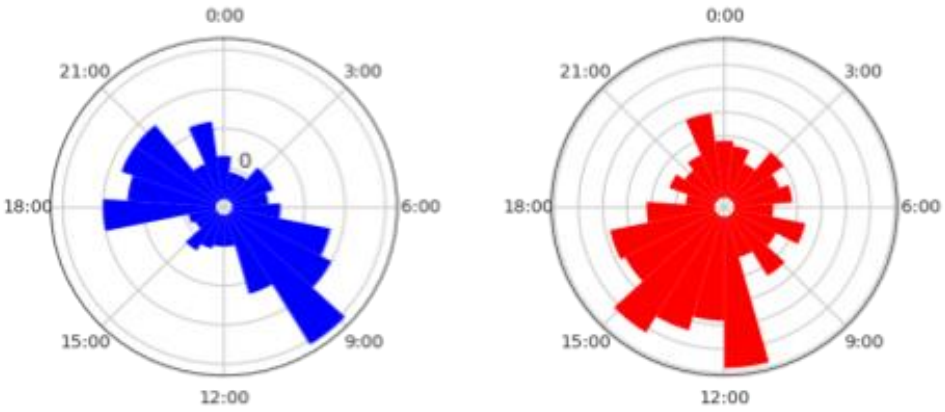
gibi varyasyon kaynaklarını çöpe atar. Bu kısa ve öz temsiller de örnekler-arası mesafeler ölçülürken, bu işleme tabi tutulmamış öznitelik vektörlerinden daha iyi performans gösterebilmektedir. Bu durum pek çok alanda kanıtlanmıştır. Çalışmamızda, şu ana kadar ilk kez gezinge gömülümlerinin, gezinge kümelemede başarıyı arttırdığı gösterilmiştir. Bu konuyu daha geniş bir çalışmada ele alarak, gezingelerin neden daha düşük boyutta daha iyi temsil edilebildiklerini açıklamayı planlıyoruz. Şu an için bu konuda daha derine inmiyor, yüksek başarısından dolayı çalışmamızın geri kalanına gezinge gömülümleri ile devam ediyoruz.

4.4 Trafik Akış Tipi Kümeleme Sonuçları

Trafik akış kümelesi aşamasındaki başarımızı ölçmek için diğerlerinden farklı bir yol izledik. Elimizdeki veri setini trafik akış tipine göre kümeledikten sonra, ortaya çıkan her bir kümenin, zamansal histogramını çizdirdik. Zamansal histogram, her bir örneğin ortaya çıktığı zaman dilimiyle eşleştirilmesi ile elde edilebilmektedir. Bu yaklaşım sayesinde, her bir kümedeki trafik akış örneklerinin genellikle hangi saat aralıklarını tercih ettiklerini gözlemlemek mümkün hale gelmiştir.

Bu aşamada, her bir örnek, önceden tanımlanan trafik akışı ile ilişkili öznitelikler tarafından temsil edilmiş, bu öznitelik seti üzerinde K-MEANS algoritması kullanılarak kümeleme yapılmıştır. Optimal K değeri seçimi, OKS-bazlı çapraz doğrulama ile başarılmıştır ve 3 olarak bulunmuştur. Şekil 4.1, elde edilen 3 küme arasından, bizim tarafımızdan seçilen 2 tanesinin zamansal histogramını belirtmektedir.

Şekil 4.1'de sunulan 2 kümenin zamansal histogramları, oldukça önemli bilgiler taşımaktadırlar. Örneğin, A kümesine ait örnekler yoğunlukla, sabah 6:00-10:00 saatleri arasında ve öğleden sonra 17:00-21:00 saatleri arasında yaşanmışlardır. Bu saatler, evrensel olarak işe-okula gidiş/dönüş saatleridir. Bunun aksine, B kümesi ise, çoğunlukla 11:00-17:00 arası örnekleri, yani gün içi trafik yoğunluğunu içermektedir. Bunun dışında, iki küme için de, diğer bütün saat aralıklarında görülmüş örnekler de bulunmaktadır. Bu durum, lokasyon özelinde farklı tip trafik akış şemalarının ortaya çıkabilmelerini göstermekte ve tarafımızdan makul olarak kabul edilmektedir. Kümeleme işleminde elde ettiğimiz üçüncü küme, bütün saat aralıklarında yüksek aktivite gösteren, zamansal olarak düzgün dağılıma yakınsayan bir karakteristik göstermiştir. Bu nedenle, bu kısımda incelenmemiştir.



(a) (b)

Şekil 4.1: (a) A kümesi için, (b) B kümesi için zamansal dağılımlar.

Trafik akış şeması kümelemedeki başarımızı gözetimli olarak da test etmek mümkündür. Bunu yaparken, elimizdeki küçük çaplı etiketli veri setinden yararlanabiliriz. Toplam yapılan doğru tahminlerin toplam sayısına ortanla yüzdesi olarak tanımalanan Hassasiyet ölçütü kullanıldığında, yoğun trafikten alınmış örneklerin %79,19 başarıyla A kümesine, hafif trafikle ilişkili örnekerin ise %61,55 oranında B kümesine atandıkları saptanmıştır. Bu durum, kümelemenin başarılı bir şekilde yapıldığının doğrudan göstergesidir.

4.5 Sürüş Tipi Kümeleme Sonuçları

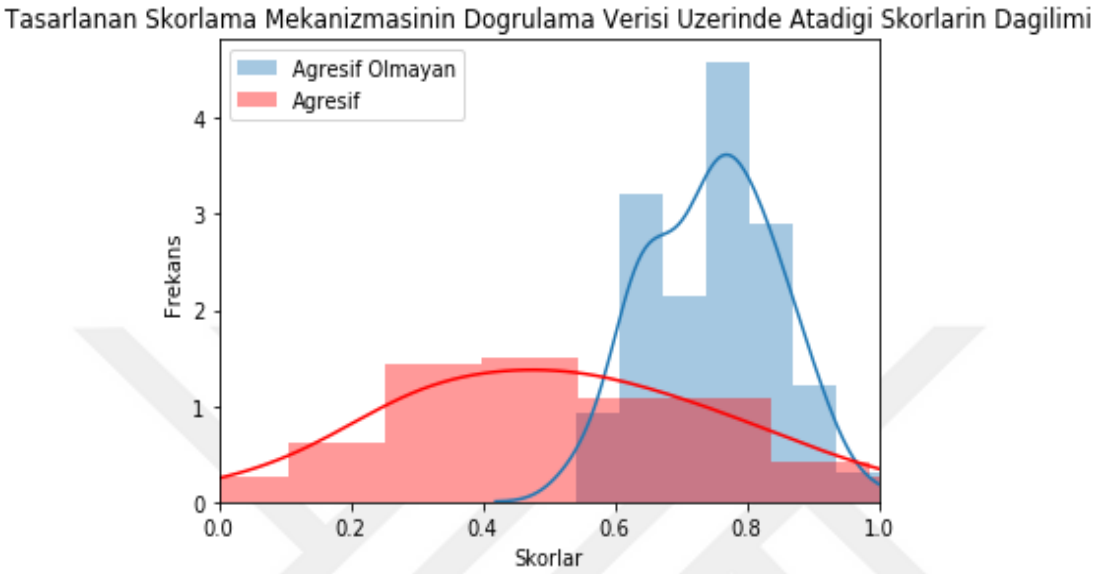
Etiketli verinin mevcut olmaması nedeniyle, sürücü stili kümeleme konusunda herhangi bir çalışma yapılmamış, modelimizin sürüş stillerini kümeleme başarısı genel skorlama yeteneği üzerinden değerlendirilmiştir.

4.6 Sürüş Tipi Skorlama Sonuçları

Skorlama yaklaşımını doğrulamak oldukça zorlu bir iştir. Şayet, elimizde skor açısından etiketlenmiş bir veri seti bulunmamaktadır. Bu noktada, daha önceden incelediğimiz, agresif/agresif olmayan sürüş şeklinde etiketlenmiş yararlanmak mümkündür. Şayet, skorlama mekanizmamız doğru çalışıyor ise, agresif etiketli örneklere, agresif olmayanlara kıyasla istatistiksel olarak gözlenebilir ölçüde düşük

skor vermelidir. Eğer bu ilişki gözlenebilirse, skorlama işlevinin makul olduğu sonucuna varmak mümkün olacaktır.

Etiketli veri üzerinde skorlama yapıldığında, sınıflar arası skor dağılımların Şekil 4.2'deki gibi olduğunu gözlenmiştir.

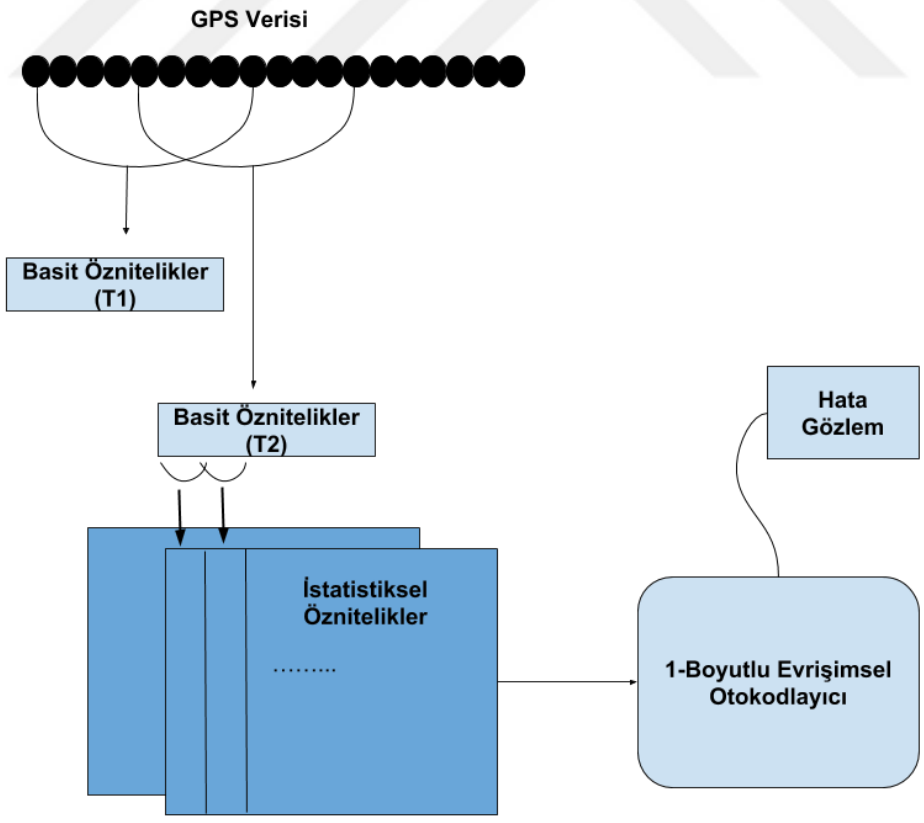


Şekil 4.2: Agresif ve agresif olmayan sürüş örnekleri için skor dağılımları.

Şekil 4.2'nin işaret ettiği gibi, agresif olmayan sürüş sınıfındaki örnekler beklendiği gibi genellikle oldukça yüksek skorlar almışlardır. Bu beklenen bir durumdur, şayet kurgulanan skorlama mekanizması çoğunluk-bazlı hareket etmekte ve bundan dolayı örneklerin çoğuna yüksek skor vermektedir. Bir diğer deyişle, yüksek puanlar pasif olarak verilmektedir. Bir sürücünün düşük puan alması için, normaları özellikle çiğnemesi gerekmektedir.

Agresif örnekler, çoğunluğu 0.3-0.8 aralığında olmak üzere, geniş bir skalada puan almışlardır. Bunun nedeni, agresiflik kavramının bütün zamana dağılamamasıdır. Agresif sürücüler de yolun bazı kısımlarını agresif olmayanlar gibi sürebilmektedir. Ancak gruplar arasında fark gözle görünür bir biçimdedir. Bu durum istatistiksel olarak da doğrulanabilir. İki sınıfın skor dağılımları da Gaussian karakteristiktedir. Agresif sınıf için ortalama skor 0.528; agresif olmayan sınıf için ise 0.794 olarak saptanmıştır. Sınıflar arası varyans farklılığı hesaba katılarak Welch's t-test kullanılmış ve sınıf ortalamalarının farkının istatistiksel olarak önemli olduğu gözlenmiştir ($p<0.001$).

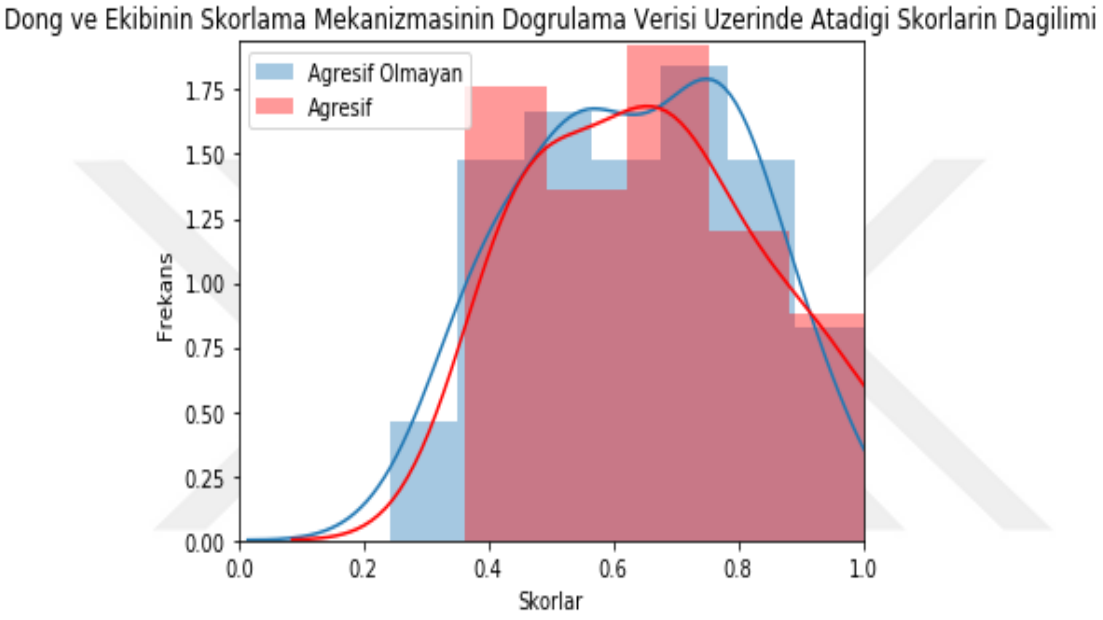
Sınıflar arası skor-bazlı ayrışım yaklaşımı ile skorlama mekanimamızın agresif ile agresif olmayan şoförleri ayırt edebildiğini tespit ettikten sonra, bunun diğer modellerden daha iyi çalışıp çalışmadığını görmek istedik. Bu aşamada, [13]'de kullanılan yaklaşım ile çalışmamızda sunulan skorlama mekanizmalarını karşılaştırmanın isabetli olacağını düşünüyoruz. Dong ve ekibi, çalışmalarında sabit uzunlukta, örtüşen GPS pencelerinde her bir nokta için anlık hız, anlık ivme ve önceki noktaya kıyasla ivmedeki değişim özniteliklerini hesaplamışlardır. Yazarlar, bunları basit öznitelikler olarak tanımlamışlardır. Ardından, bu özniteliklerin her biri için ortalama değer, standart sapma, minimum ve maksimum değerler ile %25, %50 ve %75 kartiller hesaplanmış, böylece zamanda her bir örnek için 28 adet öznitelik elde edilmiştir. Satırlarında bu öznitelik setinin elemanları, sütunlarında ise zaman endeksleri bulunan yeni yapılara ise istatistiksel öznitelikler denmiştir. Bu yapı 1-boyutlu Evrişimsel Otokodlayıcı yardımıyla tekrar üretilmiş ve üretim hatası gözlenmiştir. Üretim hatasının verinin kendi varyansına oranı birden çıkarıldıktan sonra, GPS penceresindeki davranışın geçmiş verilerle ne kadar ilişkili olduğu saptanabilir. Bu yapı Şekil 4.3'de sunulmaktadır.



Şekil 4.3: Rakip skorlama şeması [13].

Burada dikkat edilmesi gereken nokta, Dong ve ekibinin kendi çalışmalarında bu yaklaşım ile etiketli bir veri setinde şoför tanıma yapmakta olmalarıdır. Bu nedenle, yukarıda belirtilen yaklaşımdan farklı olarak Evrişimsel YSA aşaması bir etiketli öğrenme problemidir. Çalışmamızda ise, kendi skorlama senaryomuza uyulması amacıyla, bu yapı bir otokodlayıcı ile değiştirilmiş ve hata gözlemi üzerinden skorlama yapılmıştır.

Dong'un önerdiği model üzerinde çalışmamızdaki gibi skorlama çalışması yapılmış ve elde edilen sonuçlar Şekil 4.4'de sunulmuştur.



Şekil 4.4: Rakip skorlama şemasının doğrulama verisi üzerindeki skorlama dağılımı [13].

Şekil 4.4'de görüleceği gibi, skorlama Dong'un algoritması ile yapıldığında, iki sınıf için de benzer skor dağılımları görülmüştür. Agresif sürüş skor ortalaması 0.612 iken bu değer agresif olmayan sürüş tecrübeleri için 0.635 olarak saptanmıştır. Ancak, dağılımlar arasında istatistiksel bir fark bulunamamıştır ($p=0.2$). Bu durum, Dong'un yaklaşımının skorlama anlamında, bizim skorlama metodolojimizden daha başarısız olduğunu ortaya koymuştur.

5. SONUÇ VE ÖNERİLER

Çalışmamızda; İzmir, İstanbul ve Adana'da görev yapmakta olan 21 belediye otobüsünden 5 aylık bir süreçte toplanmış CAN Bus ve GPS verisinden yararlanılarak, sabit yol aralıklarında araç sürücüsünü skorlamak amaçlanmıştır. Kural bazlı yöntemlerin değişen şartlara adapte olmakta zorlanması, yapay öğrenme metotlarının ihtiyaç duyduğu etiketli verinin de mevcut olmaması nedeniyle, özgün bir olasılıksal skorlama yöntemi geliştirmenin daha sağlıklı olacağı sonucuna varılmıştır. Bu yöntem, geleneksel anomali tespiti yaklaşımını takip ederek, geçmiş veriler tarafında sürüş normları öğrenmeyi ve yeni gelen örnekleri bu norma uzaklıklarıyla ters orantılı olarak skorlamayı önermektedir. Yine literatürdeki yaklaşımlardan farklı olarak bu normlar yol tipi (geometrisi) ve trafik akış karakteristikleri dikkate alınarak oluşturulmuştur.

Skorlama mekanizmamız, yol tipi, trafik akış tipi ve sürüş stili rastgele değişkenlerinin bileşik olasılık dağılımlarının bilinmesini gerektirmektedir. Bu sürekli değişkenleri modellemek oldukça zor olduğu için, her birini kümeleme yöntemiyle ayrıklaştırma yoluna gidilmiştir. Bu durumda, kümeler arasındaki kesişen elemanların sayısı, bileşik olasılıkların optimal kestirimi haline gelmiştir. Bu bileşik olasılık bilgilerinin tutulduğu yapıya BKM adı verilmiştir. Bu yapı eğitim verisi üzerinden öğrenildikten sonra, yeni gelen bir örnek, bu matristeki pozisyonuna bakılarak kolaylıkla skorlanabilmektedir. Değişkenleri ayrıklaştırma stratejisi skorlamayı kolaylaştırmakla beraber, bu ayrıklaştırma işleminin nasıl yapılması gerektiği sorusunu ortaya çıkartmıştır. Dolayısıyla, çalışmamızın merkezi, bu soruyu nasıl cevaplayacağımız olmuştur.

Ayrıklaştırma işlemi, veriyi kümelemeyi amaçlayan gözetimsiz öğrenim araçları ile yapılabilir. Bu noktada, dört farklı tip kümeleme yaklaşımından bahsedilmiştir. Bunlar sırasıyla, Ayrıştırma-bazlı Modeller, Hiyerarşik Modeller, Yoğunluk-bazlı Modeller ve Çizge Ayrıştırma-bazlı Modellerdir. Her biri veri dağılımına farklı bir açıdan yaklaşan bu modeller ile kümeleme işlemi başarılabilir. Bunları kullanarak

öncelikle gezinge yani yol tipi kümeleme yapılmıştır. Bu aşamada, gezingeler arası benzerlikleri tanımlayan dört farklı tip uzaklık ölçütünden yararlanılmıştır. Farklı uzaklık ölçütleri ve kümeleme yöntemleri kombinasyonlarından en başarıları saptanmıştır. Bunlar K-MEANS ve BIRCH algoritmaları ve DZB ve EUOA uzaklık metrikler olmuştur. Literatürde gezinge kümeleme konusunda kesin bir metodoloji önerilmediği için, sonuçlarımız literatüre katkı açısından oldukça önemlidir. Bu sonuçlar ışığında, gezinge kümeleme konusundaki çalışmamızı bir adım ileriye götürerek, gezingelerin düşük boyutlu gömülümler halinde daha iyi kümelenip kümelenmediklerini inceledik. Gezinge verileri bir otokodlayıcı yardımıyla hacimce küçük; ancak bilgi açısından daha zengin temsiller haline getirilmiş, kümeleme bu temsiller üzerinde yapılmıştır. Literatürde ilk olmak üzere, bu temsillerin daha iyi kümelendikleri saptanmıştır. Bu işlemin daha önce başarılamamış olması, çalışmamızdaki gezinge hizalama adımının çoğu çalışmada es geçilmesi olarak yorumlanmıştır. Bu metodolojinin bütün gezinge kümeleme çalışmalarında altın standart olarak kabul edilmesi önerilerek gelecek adımlara geçilmiştir.

Trafik akış tipi kümeleme aşamasında, bu işlem için özelleşmiş, trafik akışıyla ilişkili bir öznitelik seti elde edilmiştir. Bu işlem, elimizdeki ufak etiketli bir veri setinden yararlanılarak yapılmıştır. Bu alanda kümelemenin doğrulanması, elde edilen kümelerin zamansal özellikleri ile görsel olarak yapılmıştır. Şayet, ortaya çıkan 3 kümeden bir tanesinin işe gidiş/dönüş saatlerinde, bir diğerinin ise gündüz saatlerinde yoğunlaştığı görülmüştür. Bu durum, trafik akış tipi kümeleme yönteminin anlamsal olarak doğru sonuçlar verdiğini göstermektedir. Bu nedenle, yaklamışımız doğru kabul edilmiş ve analizlere dahil edilmiştir.

Aynı işlem sürüş stili için de yapıldıktan sonra verilen skorların ne kadar makul olduğu test edilmiştir. Bu işlem için, sahadan toplanmış ve agresif/agresif olmayan şeklinde sınıflandırılmış sürüş verilerinden yararlanılmıştır. Önerilen skorlama mekanizmasının, agresif şoförlere diğerlerinden daha düşük puanlar vermesi gerektiği öngörülmüş, bu durum istatistiksel olarak doğrulanmıştır. Şayet, agresif şoförler, agresif olmayanlara kıyasla istatistiksel olarak önemli ölçüde düşük puanlar almışlardır. Bu durum skorlama yaklaşımının doğru olduğunu gösterse de, bunu diğer metotlardan iyi yapıp yapmadığını görmenin de yararlı olacağını düşünerek literatürde bu anlamda en göze çarpan yaklaşımlardan bir tanesiyle [13], bu skorlama işlemi tekrarlanmıştır. Bu derin öğrenme bazlı yaklaşım, iki sınıf arasında sürüş

skoru açısında istatistiksel bir fark yaratmayı başaramamış, bu durumun farklı yol/trafik şartlarının modellenmemiş olmasından kaynaklandığı düşünülmüştür. Bu sonuçlar, geliştirdiğimiz algoritmanın hem agresif ve agresif olmayan şoförleri ayırt edebildiği, hem de bunu diğer modellerden daha iyi yaptığını göstermektedir.

Elde edilen olumlu sonuçlara rağmen, elimizdeki etiketli veri setlerinin kısıtlı olmasından dolayı, vardığımız sonuçları daha kapsamlı bir şekilde değerlendiremiyoruz. Bunu sağlayabilmek için, uzmanlar tarafından etiketlenmiş sürüş skoru içeren veri setleri toplamayı planlıyoruz. Bunun dışında, tasarımımızın bel kemiğini oluşturan trafik akışı ve yol tipi kümeleme yöntemlerinin de test edillebilmesi için etiketli veri gerekmektedir. Bu alanda yapılacak bir çalışma, benzer tip yol ve trafik akış şemalarının ne başarıda tespit edilebildiğini gösterecektir. Son olarak, kullanılan veri setinin zenginleştirilmesi ile daha geniş çıkarımlarda bulunmak mümkün olacaktır. Örneğin, eğer fizyolojik kayıtlar sağlanırsa, sadece sürüş tipi değil, sürücülerin bilişsel süreçlerini de analiz etmek mümkün olacaktır. Benzer durum, direksiyon açısı, açısal hız ve motora dair daha bilgilendirici parametreler gibi mekanik konular için de söylenebilir. Yakın gelecekte, modelimizi bu farklı veri kaynakları ile güçlendirmek amacındayız.

KAYNAKLAR

[1] **F. Qu, F. Y. Wang, and L. Yang,** (2009**,** "Intelligent transportation spaces: Vehicles, traffic, communications, and beyond," in IEEE Communications Magazine, 2010.

[2] **Türkiye İstatistik Kurumu,** Motorlu Kara Taşıtları Haber Bülteni [Web]**,** http://www.tuik.gov.tr/PreHaberBultenleri.do?id=27646, kaynağından alındığı tarih: 01.08.2018.

[3] **S. Fazli, P. Esfehani,** "Tracking Eye State for Fatigue Detection", International Conference on Advances in Computer and Electrical Engineering (ICACEE 2012), pp. 17-20, 2012.

[4] **Y. Du, P. Ma, X. Su, and Y. Zhang,** "Driver Fatigue Detection based on Eye State Analysis," Proceedings of the 11th Joint Conference on Information Sciences (JCIS), 2008.

[5] **L. M. Bergasa and J. Nuevo,** "Real-time system for monitoring driver vigilance," in IEEE International Symposium on Industrial Electronics, 2005, vol. III, pp. 1303–1308.

[6] **Hamzah S. AlZu'bi , Waleed Al-Nuaimy , Nayel S. Al-Zubi,** "EEG-based Driver Fatigue Detection", Proceedings of the 2013 Sixth International Conference on Developments in eSystems Engineering, p.111-114, December 16-18, 2013.

[7] **S.-J. Jung, H.-S. Shin and W.-Y. Chung,** "Driver fatigue and drowsiness monitoring system with embedded electrocardiogram sensor on steering wheel," IET Intell. Transp. Syst., 2014.

[8] **X. H. Zhao, R. X. Fang, J. Rong et al.,** "Experiment study on comprehensive evaluation method of driving fatigue based on physiological signals," Journal of Beijing University of Technology, vol. 37, no. 10, 2011.

[9] **K. Ito, Y. Harada, T. Tani, Y. Hasegawa, H. Nakatsuji, Y. Tate, H. Seto, T. Aikawa, N. Nakayama, M. Ohkura,** "Evaluation of feelings of excitement caused by auditory stimulus in driving simulator using biosignals", *Proceedings of the 7th AHFE International Conference on Affective and Pleasurable Design (AHFE 2016),* vol. 483, pp. 231-240, 2016.

[10] **W. El Falou, J. Duchêne, M. Grabisch, D. Hewson, Y. Langeron, and F. Lino,** "Evaluation of driver discomfort during long-duration car driving," Appl. Ergon., 2003.

[11] **L. Wang, H. Wang, and X. Jiang,** "A new method to detect driver fatigue based on emg and ecg collected by portable non-contact sensors," Promet - Traffic - Traffico, 2017.

[12] **M. Quintero, G. Christian, J. O. Lopez, and A. C. C. Pinilla,** "Driver behavior classification model based on an intelligent driving diagnosis system," in Proceedings of the 15th International IEEE Conference on Intelligent Transportation Systems (ITSC '12), pp. 894–899, Anchorage, Alaska, USA, September 2012.

[13] **W. Dong, J. Li, R. Yao, C. Li, T. Yuan, L. Wang,** "Characterizing Driving Styles with Deep Learning", arXiv preprint arXiv:1607.03611, 2016.

[14] **H. Liu, T. Taniguchi, Y. Tanaka, K. Takenaka, and T. Bando,** "Essential Feature Extraction of Driving Behavior Using a Deep Learning Method," Intell. Veh. Symp. (IV), 2015 IEEE, 2015.

[15] **Y. Goldberg et al.,** "word2vec Parameter Learning Explained Continuous Bag-of-Word Model," Proc. Int. Conf. Learn. Represent. (ICLR 2013), 2014. [15] **Y. Goldberg et al.,** "word2vec Parameter Learning Explained Continuous Bag-of-Word Model," Proc. Int. Conf. Learn. Represent. (ICLR 2013), 2014.

[16] **M. Siaminamini, M. Naderpour, J. Lu,** "Generating a Risk Profile for Car Insurance Policyholders: A Deep Learning Conceptual Model", Australasian Conference on Information Systems, 2015.

[17] **H. Liu, S. Member, T. Taniguchi, and Y. Tanaka,** "Visualization of Driving Behavior Based on Hidden Feature Extraction by Using Deep Learning," IEEE Trans. Intell. Transp. Syst., 2017.

[18] **Y. Gu, and Q. Yu.,** "Driver-Vehicle-Environment Closed-Loop Simulation of Handling Stability Using Fuzzy Control Theory". Proceedings of the 13th IAVSD Symposium, Chendu, China, 1993. pp. 172-181.

[19] **J. Lu, D. Filev, K. P. Asante, F. Tseng, and I. V. Kolmanovsky,** "From vehicle stability control to intelligent personal minder: Real-time vehicle handling limit warning and driver style characterization," in 2009 IEEE Workshop on Computational Intelligence in Vehicles and Vehicular Systems, CIVVS 2009 - Proceedings, 2009.

[20] **A. Y. Ungoren and H. Peng,** "An adaptive lateral preview driver model," Veh. Syst. Dyn., 2005.

[21] **N. Lin, C. Zong, M. Tomizuka, P. Song, Z. Zhang, and G. Li,** "An overview on study of identification of driver behavior characteristics for automotive control," Mathematical Problems in Engineering. 2014.

[22] **T. H. Huang, V. Nikulin, and L. B. Chen,** "Detection of abnormalities in driving style based on moving object trajectories without labels", in Proceedings - 2016 5th IIAI International Congress on Advanced Applied Informatics, IIAI-AAI 2016.

[23] **V. Chandola, A. Banerjee, and V. Kumar,** "Anomaly detection: A survey" ACM Comput. Surv., 2009.

[24] **A. K. Jain, M. N. Murty, and P. J. Flynn,** "Data clustering: a review", ACM Comput. Surv., 1999.

[25] **T. Zhang, R. Ramakrishnan, and M. Livny,** "BIRCH: An Efficient Data Clustering Databases Method for Very Large", ACM SIGMOD Int. Conf. Manag. Data, 1996.

[26] **H. Jiawei, M. Kamber, J. Han, M. Kamber, and J. Pei,** "Data Mining: Concepts and Techniques", 2012.

[27] **U. Von Luxburg,** "A tutorial on spectral clustering", Stat. Comput., 2007.

[28] **A. M. Alvarez, M. Yamada, A. Kimura, and T. Iwata,** "Clustering-based anomaly detection in multi-view data", CIKM, 2013.

[29] **J. Gao, W. Fan, D. Turaga, S. Parthasarathy, and J. Han,** "A spectral framework for detecting inconsistency across multi-source object relationships", in Proceedings - IEEE International Conference on Data Mining, ICDM, 2011.

[30] **G. Yuan, P. Sun, J. Zhao, D. Li, and C. Wang,** "A review of moving object trajectory clustering algorithms" ,Artif. Intell. Rev., 2017.

[31] **M. B. Stegmann, D. D. Gomez, R. P. Plads, and D. K. Lyngby,** "A Brief Introduction to Statistical Shape Analysis", Analysis, 2002.

[32] **G. B. Arfken, H. J. Weber, and F. E. Harris,** Mathematical Methods for Physicists. 2013.

[33] **G. Yuan, P. Sun, J. Zhao, D. Li, and C. Wang,** "A review of moving object trajectory clustering algorithms", Artif. Intell. Rev., 2017.

[34] **J. Chen, R. Wang, L. Liu, and J. Song,** "Clustering of trajectories based on Hausdorff distance" , in 2011 International Conference on Electronics, Communications and Control, ICECC 2011 – Proceedings, 2011.

[35] **Ronald I. Greenberg,** "Computing the number of longest common subsequences", arXiv:cs/0301034v1, 2003.

[36] **M. Müller,** "Dynamic Time Warping", in Information Retrieval for Music and Motion, 2007.

[37] **J. Turian, L. Ratinov, Y. Bengio,** "Word representations: A simple and general method for semi-supervised learning", Proc. 48th Annu. Meeeting Assoc. Comput. Linguistics, pp. 384-394, 2010.

[38] **I. Goodfellow, Y. Bengio, A. Courville,** Deep Learning., Cambridge, MA, USA: MIT Press, 2016, [online] Available: http://www.deeplearningbook.org.

[39] **J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe,** "Deep clustering: Discriminative embeddings for segmentation and separation", in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2016.

[40] **S. Hochreiter and J. Schmidhuber,** "Long Short-Term Memory", Neural Comput. 1997.

[41] **J. Erman and M. Arlitt,** "Traffic classification using clustering algorithms", 2006 SIGCOMM Work., 2006.

[42] **R. Tibshirani, G. Walther, and T. Hastie,** "Estimating the number of cluster in a data set via the gap statistic", J. R. Stat. Soc. Ser. B Stat. Methodol, 2001.

EKLER

EK 1: Python Kodları

EK 1

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
from __future__ import division
from datetime import datetime
from mpl_toolkits.mplot3d import axes3d
from urllib import urlopen
import json
from elevationapi import Elevation
from geopy.distance import vincenty
from scipy import signal
from sklearn.decomposition import PCA
from numpy.linalg import norm
from scipy.spatial.distance import directed_hausdorff,euclidean
from fastdtw import fastdtw
import math

def time_diff(a,b):
    c = a - b
    return (c.days * 86400 + c.seconds)

from mpl_toolkits.mplot3d import axes3d

def visualize_patches(x):
    threed_path=np.zeros((x.shape[0],x.shape[1]))#[Latitude,Longitude,Elevation]

    for i in range(x.shape[0]): #Use only this many points.
        term=x.iloc[i,:]
        threed_path[i,:]=np.array([term['Latitude'],term['Longitude'],term['Elevasyon']])

    fig = plt.figure()
    ax = plt.axes(projection='3d')
    ax.scatter(threed_path[:,0],threed_path[:,1],threed_path[:,2])
    ax.set_zlabel('Rakim')

def get_polynomial_reg(data,deg):
    x = np.arange(data.shape[0])
    y = data
    coeffs=np.polyfit(x, y, deg)
    p = np.poly1d(coeffs)

    return coeffs,(p(x))

def get_euc_dist(a,b):
    return np.linalg.norm(a-b)

def transform_to_sample(x):
    #Bir yolculuğu alır, tek bir sample'ın öznitelikleri haline getirir.
    term=x.values.astype('float32')
    term-=term[0,:]
    return np.reshape(term,[1,term.shape[0]*term.shape[1]])

def inverse_transform_to_sample(x):
    #Bir sample'ı tekrar bir yolculuk haline getirir! Görselleştişrme için gerekli

    return pd.DataFrame(data=np.reshape(x,[len(x)/3,3]),columns=['Latitude','Longitude','Elevasyon'])

def find_distance(x,ref):
    #x= 3-D vector!
    x=np.asarray(x)
    ref=np.asarray(ref)
    diff=(x-ref)*111*1000 #in meters!

# Functions for Alignment!
### ALIGN KISMI ###
def unit_vector(vector):
    """ Returns the unit vector of the vector.  """
    return vector / np.linalg.norm(vector)
```

```
def angle_between(v1,v2):
    v1_u = unit_vector(v1)
    v2_u = unit_vector(v2)

    val=np.arccos(np.clip(np.dot(v1_u, v2_u), -1.0, 1.0))

    if(val>=0):
        return val
    else:
        return np.pi+abs(val)

def angle_ozan(v1,v2):
    x1=v1[0]
    y1=v1[1]
    x2=v2[0]
    y2=v2[1]

    val=math.atan2(x1*y2-y1*x2,x1*x2+y1*y2)

    return val

def angle_new(v1,v2):
    return math.atan2(abs(np.cross(v1,v2)), np.dot(v1,v2))

def euc_dist(a1,a2):
    return ((a1-a2)**2).sum()

def negative_angles(angle):
    if(angle<0):
        out=2*np.pi-abs(angle)
    else:
        out=angle

    return out

def get_pc1(x):
    x=x-x.mean(axis=0)
    C=np.cov(x.T)
    vals,vecs=np.linalg.eig(C)
    return vecs[:,vals==vals.max()]


def choose_rotation(x1,x2,alpha1,alpha2):
    x1=unit_vector(x1)
    x2=unit_vector(x2)
    R1=np.array([[np.cos(alpha1),-np.sin(alpha1)],[np.sin(alpha1),np.cos(alpha1)]])
    R2=np.array([[np.cos(alpha2),-np.sin(alpha2)],[np.sin(alpha2),np.cos(alpha2)]])
    proj1=np.dot(x2,R1)
    proj2=np.dot(x2,R2)
    dist1=euc_dist(x1,proj1)
    dist2=euc_dist(x1,proj2)
    if(dist1<=dist2):
        out=R1
    else:
        out=R2
    return out

def align_with_PCA(X):
    #input  = X is a (nx2 matrix, a N-length 2D trajectory data)
    #output = X matrix as rotated onto reference!

    #Create reference signal, a straight trajectory on x axis!
    N=X.shape[0]
    sig=np.arange(0,1,1/N)
    sig=sig.reshape(N,1)
    vec0=np.zeros((N,1))
    ref=np.hstack((sig,vec0))#+0.001*np.random.rand(N,2)  #REF

    x1=unit_vector(ref)
    x2=unit_vector(X)

    pca1=[]
    pca1=PCA(n_components=2)
    pca2=[]
    pca2=PCA(n_components=2)
```

```
    #Fit!
    pca1.fit(x1)
    pca2.fit(x2)

    c1=np.array([1,0])              #set constant!
    #c2_opt1=pca2.components_[:,0]    #kendisi
    #c2_opt2=-pca2.components_[:,0]   #negatifi

    c2_opt1=get_pc1(x2)
    c2_opt2=-get_pc1(x2)

    c2_opt1=unit_vector(c2_opt1)
    c2_opt2=unit_vector(c2_opt2)

    alpha_opt1=(angle_ozan(c1,c2_opt1))
    alpha_opt2=(angle_ozan(c1,c2_opt2))

    print('OPT1: '+str(negative_angles(alpha_opt1)*(180/np.pi))+' OPT2: '+str(negative_angles(alpha_opt2)*(180/np.pi)))
    rotation=choose_rotation(x1,x2,alpha_opt1,alpha_opt2)

    x2=np.dot(x2,rotation)
    return x2

def number_of_stops(X):
    data=(X['ahz'])
    count=0
    for i in range(1,data.shape[0]):
        if(data.iloc[i-1]>0 and data.iloc[i]==0):
            count+=1
    return count

# Bu değişecek, şu anki sadece bir deneme!
def get_driving_behavior(x):
    #x => input CANBUS data matrix!
    out=np.array([])# an empty behvavior vector
    #Feature 1: Motor Devri istatistikleri ('mtd' sütunu)
    mtd_vec=x['mtd']
    feat1=np.array([mtd_vec.mean(),mtd_vec.std()])
    out=np.append(out,feat1)
    #Feature 2: Gaz Pedal İstatistikleri ('gpp' sütunu)
    gpp_vec=x['gpp']
    feat2=np.array([gpp_vec.mean(),gpp_vec.std()])
    out=np.append(out,feat2)
    #Feature 3: Gaz Pedal Diff İstatistikleri
    gpp_diff_vec=np.diff(x['gpp'])
    feat3=np.array([gpp_vec.mean(),gpp_vec.std()])
    out=np.append(out,feat3)
    #Feature 4: İvme İstatistikleri
    ahz_vec=np.diff(x['ahz'])
    feat4=np.array([ahz_vec.mean(),ahz_vec.std()])
    out=np.append(out,feat4)
    #Feature 5: Fren Pedal İstatistikleri ('gpp' sütunu)
    fpp_vec=x['fpp']
    feat5=np.array([fpp_vec.mean(),fpp_vec.std()])
    out=np.append(out,feat5)
    #Feature6: Motor Yükü İstatistikleri
    mty_vec=x['mty']
    feat6=np.array([mty_vec.mean(),mty_vec.std()])
    out=np.append(out,feat6)
    return out

def get_traffic(x):
    #x => input CANBUS data matrix!
    out=np.array([])# an empty behvavior vector
    #Feature 1: Hız İstatistikleri ('mtd' sütunu)
    ahz_vec=x['ahz']
    feat1=np.array([ahz_vec.mean(),ahz_vec.std()])
    out=np.append(out,feat1)
    #Feature 2: Toplam Rölantide Geçen Zaman
    trc_vec=x['trc']
    feat2=np.array([trc_vec.iloc[-1]-trc_vec.iloc[0]])
    out=np.append(out,feat2)
    #Feature 3: Motor Devri
    mtd_vec=x['mtd']
    feat3=np.array([mtd_vec.mean(),mtd_vec.std()])
    out=np.append(out,feat3)
```

```
    #Feature 4: Rölantide Yakıt Tüketimi
    try_vec=x['try']
    feat4=np.array([try_vec.mean(),try_vec.std()])
    out=np.append(out,feat4)
    #Dur/Kalk Sayısı
    feat5=number_of_stops(x)
    out=np.append(out,feat5)

    return out

data=pd.read_csv("temsa12-19NOV.csv")
araclar=data['vin'].unique()
araclar
#df.assign(ln_A = lambda x: np.log(x.A))
fmt='%Y-%m-%d %H:%M:%S'

for i in araclar:
    tek_arac=data[data['vin']==i]
    tek_arac['Tarih_stamp']=tek_arac.apply(lambda row: row['gpt'].replace('T',' '),axis=1)
    tek_arac['Tarih_stamp']=tek_arac.apply(lambda row: row['Tarih_stamp'].replace('Z',''),axis=1)
    tek_arac['Tarih_stamp']=tek_arac.apply(lambda row: row['Tarih_stamp'].split('.')[0],axis=1)
    tek_arac['Tarih_stamp']=tek_arac.apply(lambda row: datetime.strptime(row['Tarih_stamp'],fmt),axis=1)
    ref=tek_arac['Tarih_stamp'][0]
    tek_arac['Differential Time']=tek_arac.apply(lambda row: time_diff(row['Tarih_stamp'],ref) ,axis=1)
    tek_arac['lat_lon']=tek_arac['lat_lon'].replace(np.nan,'0,0')
    tek_arac['Latitude']  = tek_arac.apply(lambda row: row['lat_lon'].split(',')[0],axis=1)
    tek_arac['Longitude'] = tek_arac.apply(lambda row: row['lat_lon'].split(',')[1],axis=1)

    tek_arac['Latitude']=tek_arac['Latitude'].apply(lambda row: float(row),axis=1)
    tek_arac['Longitude']=tek_arac['Longitude'].apply(lambda row: float(row),axis=1)


    #SON EKLEME ()
    tek_arac['Gun']=tek_arac.apply(lambda row:row['Tarih_stamp'].day,axis=1)
    #tek_arac['Position']= tek_arac.apply(lambda row: np.append(float(row['Longitude']),float(row['Latitude'])),axis=1)
    break

    #Tek çalıştırma için bakalım
zaman=tek_arac['Differential Time'].values
tau=np.diff(zaman)
single=[]
thr=10*60 #bu değer değişebilir! (Şu an 10 dakka beklemeyi alıyorum!)(Duraklar arası mesafe diyelim!)
limit=2*60 #2 dakkadan düşük intervalleri takma!
#sirali=sorted(tau[tau>thr])
ind=np.where(tau>thr)[0]
e = Elevation()
a=[]

#Get active intervals!
for counter,k in enumerate(ind):
    if(counter==0):
        if(k>limit):
            a.append([0,k])
            #print a
    else:
        if((k-ind[counter-1])>limit):
            a.append([ind[counter-1],k])
            #print a

#Initialize trajectory matrix!  (Do the same for the driving matrix as well!)
trajectories   =   np.array([])
driving_styles =   np.array([])
traffic        =   np.array([])

for element in a:
    ride=tek_arac[tek_arac['Differential Time']>=element[0]]
    ride=ride[ride['Differential Time']<element[1]]

    #e = Elevation()
    for counter,p in enumerate(ride.index):
        if(counter%30==0):
            temp=e.getElevation((float(ride.ix[p,'Latitude']),float(ride.ix[p,'Longitude'])))
            print temp
            ride.ix[p,'Elevation']=temp
            last=temp
        else:
```

```
            ride.ix[p,'Elevation']=last
        #break
        # Initializations
        ride['Latitude']=ride['Latitude'].astype('float32')
        ride['Longitude']=ride['Longitude'].astype('float32')
        ##############SOME INITIALIZATIONS###############

        ride['Longitude_diff']=np.append(0,np.diff(ride['Longitude'])*111*1000)
        ride['Latitude_diff']=np.append(0,np.diff(ride['Latitude'])*111*1000)
        ride['Elevation']=np.append(0,np.diff(ride['Elevation']))

        #ride['diff_dist']=np.sqrt(ride['Latitude_diff']**2+ride['Longitude_diff']**2+ride['Elevation']**2)
        ride['diff_dist']=np.append(0,np.diff(ride['tkm'])*1000)
        ##############################
        road_len    =   750 #yol uzunluğu (metre cinsinden)
        sample_size =   75  #her bir trajectory kaç eleman ile temsil edilecek?
        count       =   0
        temp        =   pd.DataFrame(data=None)
        #temp       =   pd.DataFrame()
        ind         =   0
        deg         =   3   #polynomial degree!

        for q in ride.index:
            count+=ride.loc[q,]['diff_dist']
            temp=temp.append(ride.loc[q,])
            #temp=pd.concat([temp,ride.loc[i,]],axis=1)
            #np.(temp,ride.loc[i,])
            if(count>road_len):

                lat=temp['Latitude']
                lon=temp['Longitude']
                elv=temp['Elevation']

                lat_resampled=signal.resample(lat.values,sample_size,)
                lon_resampled=signal.resample(lon.values,sample_size,)
                elv_resampled=signal.resample(elv.values,sample_size,)

                c_lat,lat_resampled=get_polynomial_reg(lat_resampled,deg)
                c_lon,lon_resampled=get_polynomial_reg(lon_resampled,deg)
                c_elv,elv_resampled=get_polynomial_reg(elv_resampled,deg)

                #Normalization!
                lat_resampled=(lat_resampled-lat_resampled[0])*111*1000
                lon_resampled=(lon_resampled-lon_resampled[0])*111*1000
                elv_resampled=(elv_resampled-elv_resampled[0])

                obs=np.concatenate((lat_resampled,lon_resampled,elv_resampled))

                #Update Trajectrory Data and Driving Data
                if not trajectories.any():#Eğer daha boşsa
                    trajectories=obs
                    driving_styles=get_driving_behavior(temp)
                    traffic=get_traffic(temp)
                    print('Boş')
                else:
                    trajectories=np.vstack((trajectories,obs))
                    driving_styles=np.vstack((driving_styles,get_driving_behavior(temp)))
                    traffic=np.vstack((traffic,get_traffic(temp)))
                    print('Dolu')

                #Repeating
                count=0
                temp=pd.DataFrame(data=None)
            trajectories_new=trajectories.copy()
    for i in range(trajectories.shape[0]):#range(trajectories.shape[0]):
        lat_x=trajectories[i,0:sample_size]
        lon_x=trajectories[i,sample_size:sample_size*2]

        #lat_x=lat_x-lat_x.mean()
        #lon_x=lon_x-lon_x.mean()

        #lat_x=(lat_x-lat_x[0])#/lat_x.max()
        #lon_x=(lon_x-lon_x[0])#/lon_x.max()

        sinyal=np.vstack([lat_x,lon_x]).T
        u=[]
```

```
    u=align_with_PCA(sinyal)
    #u=u-u[0,]
    plt.figure()
    plt.scatter(u[:,0],u[:,1])
    trajectories_new[i,0:sample_size]=u[:,0]
    trajectories_new[i,sample_size:2*sample_size]=u[:,1]
    trajectories_new[i,2*sample_size:3*sample_size]-=trajectories_new[i,2*sample_size]

from sklearn.cluster import KMeans,SpectralClustering,DBSCAN

cluster1=KMeans(n_clusters=4)
cluster2=KMeans(n_clusters=4)

#cluster1.fit(driving_styles)
pred_driving=cluster1.fit_predict(driving_styles)
#cluster2.fit(trajectories_new)
pred_trajectory=cluster2.fit_predict(trajectories_new)

cooccurence=np.zeros((np.unique(pred_trajectory).shape[0],np.unique(pred_driving).shape[0]))
for i in range(pred_driving.shape[0]):
        cooccurence[pred_trajectory[i],pred_driving[i]]+=1

#normalize no-occurence
for i in range(cooccurence.shape[0]):
    cooccurence[i,]/=sum(cooccurence[i,])

plt.imshow((cooccurence))

#Bu sırayı koru (makaledeki sıra.)
def dist_euclidean(x,y,dim):
    N=int(x.shape[0]/dim)#length
    x=np.reshape(x,[N,dim],'F')
    y=np.reshape(y,[N,dim],'F')
    return np.sqrt(((x-y)**2).sum(axis=1)).mean()

def dist_euclidean_with_PCA(x,y,dim):
    #NORMALIZATION?
    N=int(x.shape[0]/dim)#length
    x=np.reshape(x,[N,dim],'F')
    y=np.reshape(y,[N,dim],'F')

    pca_x=PCA(n_components=1)
    pca_y=PCA(n_components=1)
    pca_x.fit(x)
    pca_y.fit(y)
    return np.sqrt(((pca_x.components_[:,0]-pca_y.components_[:,0])**2).sum())

def dist_hausdorff(x,y,dim):
    N=int(x.shape[0]/dim)#length
    #NORMALIZATION?
    x=np.reshape(x,[N,dim],'F')
    y=np.reshape(y,[N,dim],'F')
    return directed_hausdorff(x,y)[0]

def dist_LCSS(x,y,dim,delta,eps):
    #Let them be fixed!
    N=int(x.shape[0]/dim)#length
    out=np.zeros((N,dim))
    for i in range(N):
        for d in range(dim):#boyutlarda dolaş
            sig1=x[d*N:(d+1)*N][i]
            sig2=y[d*N:(d+1)*N]

            interval=sig2[np.max([0,i-delta]):np.min([N-1,i+delta])+1]

            #case=interval[(interval>=(sig1-eps)).any() and (interval<=(sig1+eps)).any()]
            case1=interval[interval>=(sig1-eps)]
            case2=case1[case1<=(sig1+eps)]
            case=case2

            if(len(case)>0):
                #print('OLDU!')
                out[i,d]=1
    result=out[:,0]*out[:,1]*out[:,2]
    #Get the longest sequence!
    state='off'
```

```
    count=0
    max_count=0
    for i in range(result.shape[0]):
        if(state=='off' and result[i]==1):
            state='on'
            count+=1
            if(count>max_count):
                max_count=count
        elif(state=='on' and result[i]==1):
            count+=1
            if(count>max_count):
                max_count=count
        elif(state=='on' and result[i]==0):
            state='off'
            count=0

    return 1/max_count

def dist_dtw(x,y,dim):
    N=int(x.shape[0]/3)#length
    x=np.reshape(x,[N,dim],'F')
    y=np.reshape(y,[N,dim],'F')
    return fastdtw(x,y, dist=euclidean)[0]

dist_dtw(trajectories_new[0,:],trajectories_new[1,:])

#BENCE BURADA PROBLEM VAR###


from sklearn.cluster import KMeans,SpectralClustering,DBSCAN

n_clusters=3
cluster1=KMeans(n_clusters)
cluster2=KMeans(n_clusters,precompute_distances=True)


samp=trajectories_new.shape[0]
z_traj=np.zeros((samp,samp))
for i in range(samp):
    for j in range(samp):
        z_traj[i,j]=dist_euclidean_with_PCA(trajectories_new[i,],trajectories_new[j,])
#cluster1.fit(driving_styles)
pred_driving=cluster1.fit_predict(driving_styles)
#cluster2.fit(trajectories_new)
pred_trajectory=cluster2.fit_predict(z_traj)

#cooccurence=np.zeros((np.unique(pred_trajectory).shape[0],np.unique(pred_driving).shape[0]))
cooccurence=np.zeros((n_clusters,n_clusters))
for i in range(pred_driving.shape[0]):
        cooccurence[pred_trajectory[i],pred_driving[i]]+=1

#normalize no-occurence
for i in range(cooccurence.shape[0]):
    cooccurence[i,]/=sum(cooccurence[i,])

plt.imshow((cooccurence))


#BENCE BURADA PROBLEM VAR###
### Trafikle Beraber###


from sklearn.cluster import KMeans,SpectralClustering,DBSCAN

n_clusters_trajectory=10
n_clusters_driving=5
n_clusters_traffic=2
dim=3
cluster1=KMeans(n_clusters=n_clusters_driving)
cluster2=SpectralClustering(n_clusters=n_clusters_trajectory,affinity='precomputed')
#cluster2=KMeans(n_clusters=n_clusters_trajectory,precompute_distances=True)
cluster3=KMeans(n_clusters=n_clusters_traffic)
samp=trajectories_new.shape[0]
z_traj=np.zeros((samp,samp))
for i in range(samp):
    for j in range(samp):
```

```
        z_traj[i,j]=dist_LCSS(abs(trajectories_new[i,]),abs(trajectories_new[j,]),dim,10,0.1)
#cluster1.fit(driving_styles)
pred_driving=cluster1.fit_predict(driving_styles)
#cluster2.fit(trajectories_new)
pred_trajectory=cluster2.fit_predict(z_traj)
pred_traffic=cluster3.fit_predict(traffic)

#cooccurence=np.zeros((np.unique(pred_trajectory).shape[0],np.unique(pred_driving).shape[0]))
cooccurence=np.zeros((n_clusters_trajectory,n_clusters_driving,n_clusters_traffic))
for i in range(pred_driving.shape[0]):
            cooccurence[pred_trajectory[i],pred_driving[i],pred_traffic[i]]+=1

cooccurence+=0.01*np.random.rand()
#normalize no-occurence
for i in range(cooccurence.shape[0]):
    cooccurence[i,]/=sum(cooccurence[i,])
#plt.imshow((cooccurence))
for i in range(cooccurence.shape[2]):
    plt.figure()
    plt.imshow(cooccurence[:,:,i])
    print cooccurence[:,:,i]
```

ÖZGEÇMİŞ

Ad-Soyad : **Ozan Fırat ÖZGÜL**

Uyruğu : **T.C.**

Doğum Tarihi ve Yeri : **27.06.1989 Güzelyurt/K.K.T.C.**

E-posta : **ofirat.ozgul@stm.com.tr**

ÖĞRENİM DURUMU:

- **Lisans** : 2012, Bilkent Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Mühendisliği
- **Yüksek lisans** : 2015, KU Leuven, Mühendislik Fakültesi, Biyomedikal Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

Yıl	Yer	Görev
2017-	**STM A.Ş.**	**Veri Bilimci**
2017-	**CBML (TOBB ETÜ)**	**Araştırmacı**

YABANCI DİL: İngilizce (TOEFL iBT: 105/120)

TEZDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

- **Özgül OF,** Çakır MU, Tan M, Amasyali MF, Hayvaci HT, "A Fully Unsupervised Framework for Scoring Driving Styles", IEEE 9th International Conference on Intelligent Systems, 2018.

DİĞER YAYINLAR, SUNUMLAR VE PATENTLER:

- Tan, Mehmet, Özgül, Ozan Fırat, Bardak, Batuhan, Ekşioğlu, Işıksu "Drug response prediction by ensemble learning and drug-induced gene expression signatures." *arXiv preprint arXiv:1802.03800* (2018). (Elsevier Genomics dergisinden kabul almıştır.)
- Özgül, Ozan Fırat, Batuhan Bardak, and Mehmet Tan. "Predicting drug activity by image encoded gene expression profiles." *2018 26th Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2018.