T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİTABANI SİSTEMLERİNDE SORGU OPTİMİZASYONLARININ VERİ ANALİZ TEKNİKLERİYLE GELİŞTİRİLMESİ

AYŞE ÖNCÜ UYSAL

DOKTORA TEZİ
MATEMATİK MÜHENDİSLİĞİ ANABİLİM DALI
MATEMATİK MÜHENDİSLİĞİ PROGRAMI

DANIŞMAN
DOÇ. DR. AYLA ŞAYLI

İSTANBUL, 2011

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

VERİ TABANI SİSTEMLERİNDE SORGU OPTİMİZASYONLARININ VERİ ANALİZ TEKNİKLERİYLE GELİŞTİRİLMESİ

Ayşe Öncü UYSAL tarafından hazırlanan tez çalışması 02.08.2011 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Matematik Mühendisliği Anabilim Dalı'nda **DOKTORA TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Doç. Dr. Ayla ŞAYLI

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Prof. Dr. Behiç ÇAĞAL

İstanbul Kültür Üniversitesi _____

Prof. Dr. İrfan ŞİAP

Yıldız Teknik Üniversitesi _____

Doç. Dr. Hülya ŞAHİNTÜRK

Yıldız Teknik Üniversitesi _____

Doç. Dr. Yaşar SÖZEN

Fatih Üniversitesi _____

Doç. Dr. Ayla ŞAYLI

Yıldız Teknik Üniversitesi _____

ÖNSÖZ

Bu tez çalışmasında, gün geçtikçe önemi artan veritabanlarında sorgu çalışma proseslerini optimize etmek amacıyla anlamsal sorgu optimizasyonu araştırılmıştır. Anlamsal Sorgu Optimizasyonunun Otomatik Kural Öğrenme modülünün iyileştirilmesi için veri analiz metotlarına dayalı yeni bir metot önerilmiştir.

Bu tezin hazırlanmasında yardımlarını ve desteğini esirgemeyen değerli hocamız Sayın Doç. Dr. Ayla Şaylı'ya çok teşekkür ederim.

Mayıs, 2011

Ayşe Öncü UYSAL

İÇİNDEKİLER

Sayfa

b_0	Tahmin edilen regresyon kesim noktası
$b_1. b_i. . b_k$	Tahmin edilen eğim katsayıları
R	Çoklu korelasyon katsayısı
R^2	Çoklu belirlilik katsayısı
$\hat{Y}$	Tahmin edilen Y değeri
@	Karşılaştırma operatör göstergesi
-->	Kurallarda sol ve sağ tarafların ayrıldığı gösterge(Gerektirme Operatörü)

KISALTMA LİSTESİ

ANSI	American National Standards Institute(Amerikan Ulusal Standartlar Enstitüsü)
DDL	Data Definition Language (Veri Tanımlama Dili)
DML	Data Manipulation Language(Veri İşleme Dili)
DMRA	Dynamic Multiple Regression Analysis (Dinamik Çoklu Regresyon Analizi)
ISO	International Organization for Standardization (Uluslararası Standardizasyon Organizasyonu)
RDBMS	Relational Database Management System(İlişkisel Veritabanı Yönetim Sistemi)
SQL	Structured Query Language (Yapısal Sorgu Dili)
SQO	Semantic Query Optimization (Anlamsal Sorgu Optimizasyonu)
VIF	Variance Inflation Factor(Varyans Büyütme Faktörü)
VDL	View Definition Language (Görünüm Tanımlama Dili)

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ÖZET

VERİTABANI SİSTEMLERİNDE SORGU OPTİMİZASYONLARININ VERİ ANALİZ TEKNİKLERİYLE GELİŞTİRİLMESİ

Ayşe Öncü UYSAL

Matematik Mühendisliği Anabilim Dalı

Doktora Tezi

Tez Danışmanı: Doç. Dr. Ayla ŞAYLI

Bilgisayarlar eğitim, sağlık, finans ve bunun gibi birçok alanda kullanılmaktadır. Bu kullanım sonucunda, insanların ihtiyaçlarını daha iyi karşılayabilmek için daha iyi servislerin kullanımı zorunlu hale gelmiştir. Günümüzde bu servisler 7 gün 24 saat hizmet vermektedirler. Bu, gün geçtikçe büyüyen birçok verinin oluşmasına sebep olmuştur ve bu verinin depolanması, dizaynı, yedeklenmesi, bakımı ve yönetilmesi ihtiyacı doğmuştur.

Bu sebeple 1970 yılında İlişkisel Veritabanı Yönetim Sistemleri (RDBMS) ilk olarak Codd tarafından geliştirilmiştir. Sistemin endüstride ticari hale gelmesi oldukça zaman almış ancak bu aşamadan sonra İlişkisel Veritabanı Yönetim Sistemlerini kullanan uygulamalar, en çok aranılan uygulamalar haline almıştır. RDBMS'nin birçok kullanıcı tarafından kullanılması neticesinde, veritabanı boyutları, kayıt sayıları ve bununla beraber sorgu sayıları da önemli bir artış göstermiştir. RDBMS'nin artış gösteren doğası, sorgu çalıştırma proseslerinde probleme yol açmış ve bu sebeple bir çok araştırmacı sorguları daha hızlı çalıştırabilmenin yollarını bulmak için araştırmalara yönelmiştir.

Anlamsal sorgu optimizasyonu, sorguları verildiği halinden daha hızlı çalıştırabilmek için geliştirilmiş bir yaklaşımdır. Bu optimizasyon, kuralları kullanarak sorguları verildiği halinden daha hızlı çalıştırabilmek için optimum sorguyu bulma esasına dayanır.

Kurallar daha önceki sorgular kullanılarak öğrenilebilir. Yaklaşım dört ana modüle sahiptir, bunlar “Sorgu Gösterimi”, ”Sorgu Optimizasyonu” ,”Otomatik Kural Öğrenme” ve “Kuralların Bakımı” modülleridir. Bu tez çalışmasında “Otomatik Kural Öğrenme” modülü üzerine odaklanılmıştır ve kural öğrenmek için veri analizlerine dayalı yeni bir dinamik metot önerilmiştir.

Kural öğrenme için yazılım geliştirilmiş ve önerilen metodun hesaplanan çıktıları gösterilmiştir. Bu tez çalışmasında geliştirilen yazılım ayrıca mühendislikte veri analiz çalışmaları için de kullanılmıştır. Yine bu çalışma göstermiştir ki, kural öğrenme için geliştirilen sistem yararlı kuralların öğrenilmesiyle beraber daha verimli bir sistem oluşturulmasına yardımcı olmuştur.

Anahtar Kelimeler: Veritabanı, SQL, Anlamsal Sorgu Optimizasyonu, Veri Analizi, Çoklu Regresyon Analizi

ABSTRACT

DEVELOPMENT OF QUERY OPTIMIZATION WITH DATA ANALYSIS TECHNIQUES IN DATABASE SYSTEMS

Ayşe Öncü UYSAL

Department of Mathematical Engineering

PhD. Thesis

Advisor: Assoc. Prof. Dr. Ayla ŞAYLI

Computers are being used in many different purposes such as education, health, finance and so on. The need of this usage becomes compulsory for better services in order to satisfy people's needs, especially nowadays it is necessary to have these services in 7 days at 24 hours. Doing this produces a lot of data. For this purpose, this data has to be stored, retrieved and managed in the running time as well as backed up, designed, maintained.

For this purpose, in 1970, the Relational Database Management Systems (RDBMSs) has developed firstly by Codd. It has taken many years to be commercial in industry. But when it was, software applications based on RDBMSs was the most wanted applications which was capable of using the data whenever required. In time, the use of the RDBMSs by many users caused the huge database size and the number of records increased as well as the number of queries. Although the RDBMSs has developed in many aspects, this increasing nature of the RDBMSs remained to cause problems for query processing and many researchers aimed to work on how to execute queries faster than the current executions.

Semantic query optimization is studied to run queries better than their given forms. This optimization is relatively a new approach to execute the optimum query instead of the user given query using the rules. These rules can be learned from the past

queries. The approach has four modules which are “Query Representation”, “Query Optimization”, “Automatic Rule Derivation” and “Rule Maintenance”. In this thesis, we focused on the automatic rule derivation and proposed a new dynamic method to learn new rules using data analysis, mainly multiple regression analysis, stepwise regression analysis and standardized regression coefficients from statistics.

A software application of the rule derivation is implemented and the computational results show that the proposed method can derive new rules. This thesis also can be used for the data analysis in engineering. The rule learning method of the implemented system can derive the useful rules efficiently.

Key words: Database, SQL, Semantic Query Optimization, Data Analysis, Multiple Regression Analysis

BÖLÜM 1

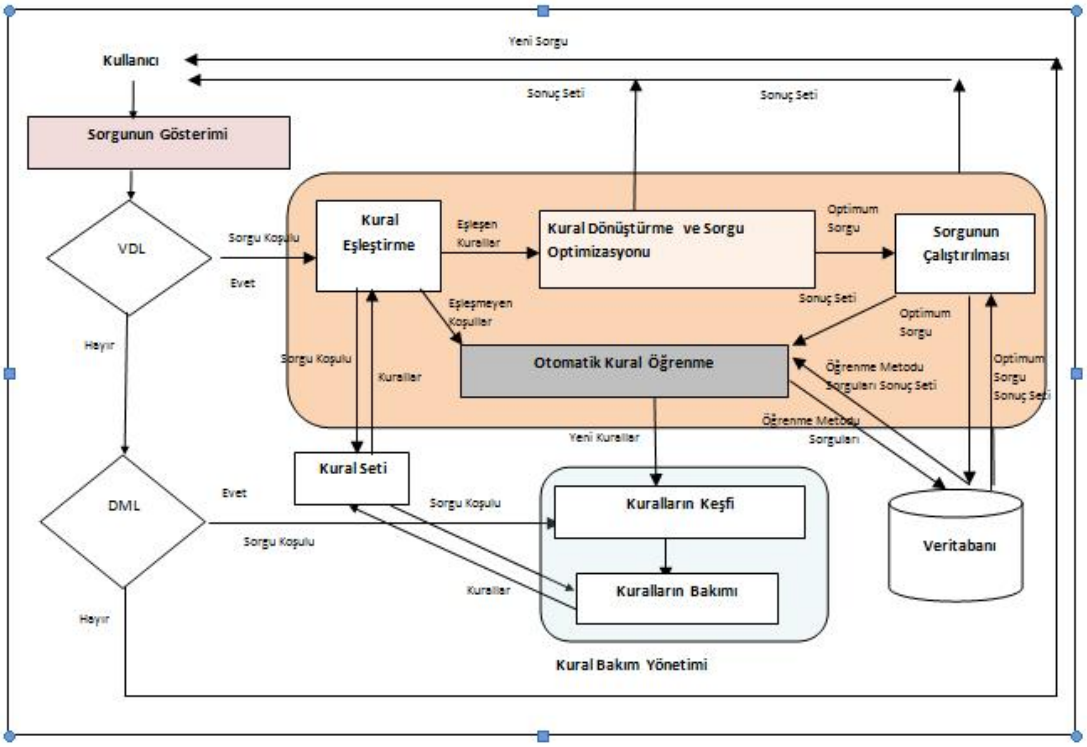
GİRİŞ

Günümüzde bilgisayar teknolojileri her alanda kullanılmaktadır. Özellikle yedi gün yirmi dört saat hizmete açık olan eğitim, sağlık, güvenlik gibi alanlarda kullanılan otomasyon sistemlerinde verilerinin depolanması, sorgulanması ve güvenliği oldukça önemlidir. Bu kullanımlar uygulanan alanlar ne olursa olsun veritabanlarına olan ihtiyacı artırmıştır. Veritabanları çok kullanıcı tarafından ortak kullanılmaları ve hizmette sınır tanınmaması nedeni ile hızla büyüyen bir doğaya sahiptir. 1991 yılında yapılan bir çalışmaya göre veritabanının büyüme hızının her yirmi ayda iki katı olduğu yayınlanmıştır (FRAWLEY vd. [1]). Günümüzde ise web tabanlı sistemlerin artmasıyla da beraber büyüme hızının tahmini değeri saptanamamaktadır.

Veritabanlarının büyümesiyle, veritabanlarını kullanan programların veritabanından bilgi alım süreci daha çok zaman almaya başlamış dolayısıyla veritabanlarında bilgi keşfi oldukça önemli bir hal almıştır. Bu alandaki yakın tarihli araştırmacıların çalışmaları göz önüne alındığında var olan veritabanlarının içerisindeki verilerin analiz edilmesi ve bu analizlerin sonuçlarına göre ileriye dönük yeni kullanıcı taleplerinin gözden geçirilmesi gerekmektedir. Her kullanıcının talebi karşılandıktan sonra talebi ve talebinin sonucu incelenerek daha sonra gelecek kullanıcıların taleplerini daha akıllı bir şekilde karşılayabilmenin yolları araştırılmaktadır. Bu amaçla araştırmamız içerisinde ana hedefimiz veritabanı içerisindeki verilerin analiz edilmesi ve kullanıcı taleplerinin sonuçlarının en kısa zamanda bulunmasıdır. Veri analizi için pek çok araştırmacı istatistik ve olasılık metotlarının kullanılmasını önermiştir ancak bilgisayar ortamında sorgu tabanlı olarak gerçekleştirmemiştir. Bu sebeple bu çalışmada istatistik metotlar araştırılmış ve kullanılmıştır (Armutlu, Çil, Hamburg, [2],[3],[4]).

1.1 Literatür Özeti

Veritabanı yönetim sistemlerinde kullanıcı sorgularının çalıştırılmasındaki en önemli kısım sorguların optimizasyonu konusudur. Ticari olarak satışa hazır sunulan veritabanı yönetim sistemleri de göz önüne alındığında sorguların optimizasyonu üç yaklaşımla yapılmıştır. Bunlar "Sezgisel Tabanlı Sorgu Optimizasyonu", "Sistematik Tabanlı Sorgu Optimizasyonu", "Anlamsal Sorgu Optimizasyonu" dur. İlk iki yaklaşım ticari uygulamalarda mevcut olup anlamsal sorgu optimizasyonu diğerlerine kıyasla daha çok akademik araştırma ve geliştirme aşamasındadır ([5-30]). Tezin kapsamında incelenen ve çalışılan yaklaşım anlamsal sorgu optimizasyonudur. SQO'nun(Anlamsal Sorgu Optimizasyonu) amacı orijinal sorguyu, aynı sonuç setini döndüren farklı bir forma dönüştürerek daha hızlı ve verimli sonuç alınmasını sağlamaktır. SQO dört ana modüle sahiptir. Bunlar "Sorgu Gösterimi", "Sorgu Optimizasyonu", "Otomatik Kural Öğrenme" ve "Kuralların Bakımı" modülleridir. Aşağıdaki Şekil 1.1'de SQO'nun çalışma metodolojisi gösterilmiştir



Şekil 1.1 Anlamsal Sorgu Optimizasyonunun Çalışma Metodolojisi

Şekil 1.1'de gösterildiği üzere SQO'ya giren sorgu, SQL sorgulama diliyle ifade edilen kullanıcının sorgusu sorgu optimizatörünün kullandığı iç sorgu diline (ağaç veya cebirsel forma) dönüştürülmelidir. Buna "Sorgunun Gösterimi" adı verilmiştir. Eğer sorgu görüntü tanımlama dili (SELECT) ise kural seti bu sorgu ile ilgili tüm kuralları bulmak amacı ile (orijinal veritabanına erişmeden) taranır. İlişkili kurallar bulunduktan sonra anlamsal dönüştürme işlemi yapılır. Dönüştürme işleminin ardından en uygun (maliyeti en az olan) sorgu, optimum sorgu olarak belirlenir. Optimum sorgu çalıştırılır. Eğer gelen sorguya uygun herhangi bir eşleşme(uygun kural) bulunamadıysa, sorgu daha sonraki sorgularda kullanılmak üzere yeni kuralların türetilmesi için kullanılabilir. Bu durumda koşullar "Otomatik Kural Öğrenme" modülüne girer. Eğer gelen komut veri güncelleme dili (INSERT, UPDATE, DELETE) ise, kuralların doğruluğunu korumak amacıyla "Kural Bakımı" modülü devreye girer.

Sorgu tabanlı kural öğrenme metotları genel olarak bilimsel literatürde iki yaklaşım içerisinde tanımlanmıştır. Bu yaklaşımlar "Sezgiye Dayalı" ve "Veriye Dayalı" olarak iki alt başlıkta isimlendirilebilir. Sezgiye dayalı olarak yapılan öğrenme metotları en fazla Siegel tarafından gerçekleştirilmiştir (Siegel, Siegel vd., [18], [19], [23], [24]). Veriye dayalı olarak yapılan öğrenme metotları ise daha çok Piatetsky-Shapiro ve Knoblock tarafından ortaya atılmıştır (Hsu ve Knoblock, Piatetsky-Shapiro, Arens ve Knoblock, Arens vd., [13-15], [25-28]). Bu tez içeriğinde önerilen istatistiksel kural öğrenme metodu Siegel metodu ile karşılaştırıldığından dolayı Siegel'in metodu Bölüm 3.1'de detaylı olarak açıklanmaktadır. Bölüm 3.2 'de Knoblock tarafından ortaya atılan öğrenme metodu tanımlanmıştır. İstatistik ve olasılığa dayalı kural öğrenme metotlarının çeşitli yayınlarda [Hsu ve Knoblock ,Siegel, Siegel vd.,Piatetsky-Shapiro, Arens ve Knoblock, Arens vd., [13-15],[18],[19],[23-28]) önerilmesi ve adres edilmiş olmasına rağmen belirli bir metot ve bu metodun bilgisayarlı sonuçları açıklanmamıştır.

1.2 Tezin Amacı

Bu tezde SQO'nun modülleri incelenmiş ve "Otomatik Kural Öğrenme" modülü için veri analiz teknikleri kullanılarak iyileştirme yapılması amaçlanmıştır. Araştırma içerisinde ana hedef veritabanı içerisindeki verileri analiz ederek ve bu analizlerden elde edilen sonuçları sorguların optimizasyonunda kullanarak, kullanıcıya daha hızlı cevap veren sistemin oluşturulmasını sağlamaktır. Veritabanları genel yapısı ve SQO ile ilgili detaylı inceleme Bölüm 2 ve Bölüm 3 'de verilecektir. Kullanılan veri analiz metotları ile ilgili inceleme Bölüm 4'de verilecektir.

1.3 Hipotez

Bir veritabanı üzerinde çalıştırılan sorgu ele alınarak yapılan kural öğrenmeleri sonucunda elde edilen kuralların sisteme girilecek yeni sorgular için kullanılması anlamsal sorgu optimizasyonunun temelini oluşturmaktadır. Bu kullanım ile sistem daha akıllı bir sisteme dönüştürülmekte ve gerekmedikçe veritabanına giriş yapılmadan öğrenilen kurallardan sorgunun sonucu bulunma ihtimali araştırılmaktadır. Eğer bu ihtimal yoksa bu durumda bile kuralın sisteme sağlayacağı bilgi dahilinde daha bilinçli bir sorgu optimizasyonu yapılması sağlanabilir. Yani sistem kendi içerisinde kendinden kendine öğrenme yapabilmektedir. Kurallardan elde edilecek anahtarlara ve indeks yapılarına ait sütunlar içeren kuralların kullanımları sistemin zamandan tasarruf etmesini sağlar. Ancak daha önce açıklanan öğrenme metotlarına göre yapılan öğrenmeler sonucunda elde edilen kural sayısı oldukça fazladır. Tüm kuralların kullanılması veya bazılarının seçilerek kullanılması sonucu elde edilen ipuçlarına göre her kural aynı yararlılığa sahip olmamaktadır. Bu nedenle mümkünse bu tip kuralların hiç öğrenilmemesi ya da bu tip kuralların seçilerek kullanılması gerekmektedir. Bu bağlamda ilk husus hedeflenerek kuralın faydasının ölçülmesinde veri analiz tekniklerinin kullanılması önerilmiştir ancak bu konuda çalışma gerçekleştirilmemiştir. Veri analiz teknikleri ve özellikle çoklu regresyon analizleri kullanılarak oluşturulacak öğrenme metodunu içeren kural öğrenme ve kullanma sistemi, sorguların çalışmasını daha verimli bir hale getirebilir. Ayrıca bu sistem zaman kazancını maksimum yaparak, sorgunun çalışma zamanını minimize edebilir.

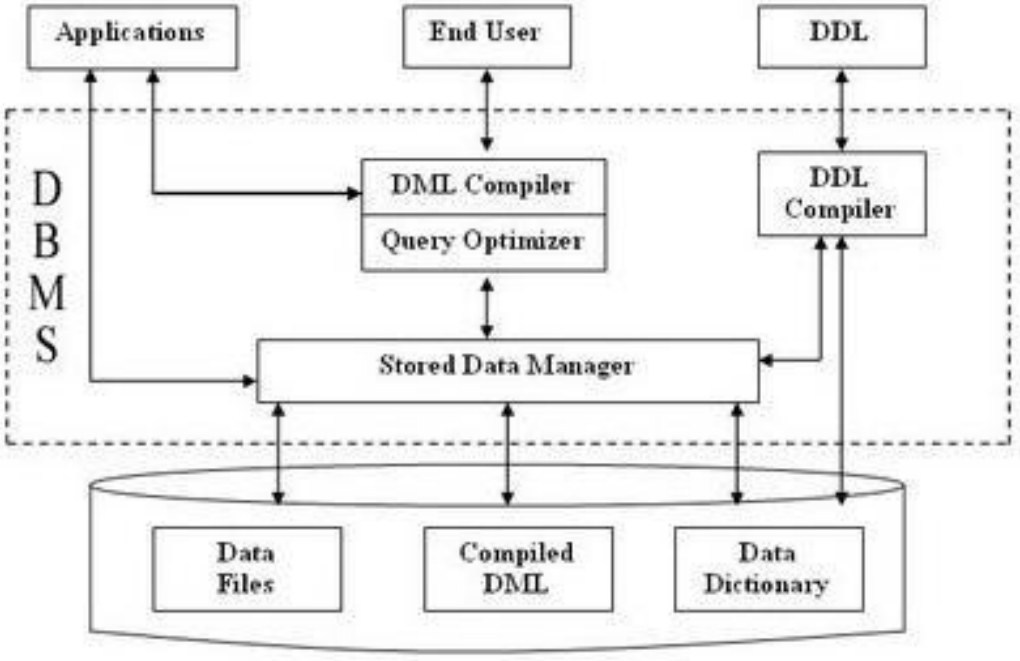
Bu doğrultuda oluşturulan kural öğrenme sistemi öğrenmeye başlamadan önce anlamlı bir şekilde sütun sayısını azaltmalıdır. İstatistiksel analizler yardımıyla kural öğrenmek istediğimiz sütun için anlamı olan başka sütunları tespit etmek daha anlamsız olan sütunları ise elemek mümkündür. Sütunların azalması ya da bir başka deyişle eleme yapılması kural sayısının azalmasına neden olacaktır. Daha az kural üreterek çalışan öğrenen sistem daha hızlı çalışacaktır. Kural seti anlamsız verilerin neden olduğu gereksiz şişmelerden arınacaktır.

Bu tezde çoklu regresyon analizine, geriye doğru eleme yöntemi ve standartlaştırılmış regresyon katsayılarına dayalı (Armutlu, Çil, Hamburg, [2],[3],[4]) dinamik bir kural öğrenme metodu oluşturulmuştur. Önerilen metodu kullanarak öğrenmeyi sağlayan bir otomasyon yazılımı oluşturulmuştur. Yazılımın sonuçları Bölüm 5'de anlatılmıştır.

BÖLÜM 2

İLİŞKİSEL VERİTABANI YÖNETİM SİSTEMLERİ

İlişkisel veritabanı yönetim sistemleri (RDBMS) verilerin ilişkisel olarak depolandığı, çok kullanıcı tarafından farklı amaçlar için yararlanıldığı ve verilerin yönetildiği yazılımcılara yönelik geliştirilmiş profesyonel yazılım sistemleridir. Bu sistem Şekil 2.1'de ki bir mimari yapıya sahiptir.



Şekil 2.1 Veritabanı Genel Mimarisi

Bu bölümde RDBMS sistemlerinde kullanılan temel kavramlar ve bu sistemlerdeki anlamsal sorgu optimizasyonu alt başlıklar halinde açıklanacaktır.

2.1 Temel Kavramlar

Bu bölümde ilişkisel veri yapılarından ve ilişkisel veri yapılarının bazı genel kavramları tanıtılacaktır. İlişkisel Veritabanları ile ilgili bütün kavramlar açıklanmamıştır sadece bu tezin ana konusuna giriş yapmak için gerekli olanlar seçilerek özet olarak aktarılmaya çalışılmıştır([35],[36],[37]). Ayrıca seçim esnasında anlatılan kavramların tez içerisindeki kullanım sıklığı da dikkate alınmıştır. Bu tezde veritabanı olarak SQL Server kullanılmıştır.

2.1.1 Veritabanı Tasarım Evreleri

Veritabanlarının 4 önemli evresi bulunmaktadır. Bunlar:

- Kavramsal Evre
- Mantıksal Evre
- Uygulama Evresi
- Fiziksel Evre

2.1.1.1 Kavramsal Evre

Bu evre modelin analiz ve test evresidir. Evrenin temelini iki önemli unsur oluşturmaktadır. Bu unsurlar aşağıda verilmiştir:

- Varlık kümelerini ve aralarındaki ilişkiyi keşfetmek
- İş kurallarını belirlemek, ve sistem kapsamının keşif ve dokümantasyonu

Varlıklar iş süreçlerine temel oluşturan nesneler olarak düşünülebilir.

Örneğin:

“İnsanlar ay sonunda maaş alırlar “ sürecindeki insan ve maaş bu sürecin varlıklarıdır.

İş kuralları sistemin işleyişini düzenleyen kurallardır.

Örneğin:

“Her çalışanın TC Kimlik numarası mutlaka sisteme kaydedilmelidir”, ”Telefon numaraları 12 karakterden oluşmalıdır” gibi kurallara “İş Kuralları” denir. Bunların

hangilerinin esnek olacağına, hangilerinin veritabanı, hangilerinin kullanıcı arayüzü tarafında kısıtlanacağına karar verilmelidir.

2.1.1.2 Mantıksal Evre

Kavramsal evrede yapılan işin mükemmelleştirilerek tamamlanma aşamasıdır. Bu bölümün çıktısı için "Sistemin Detaylı Proje Taslağı" denir. Bu aşamada kavramsal evre tümüyle gözden geçirilir ve bunun sonucunda tüm varlık setlerinin tanımlaması yapılır. Tanımlanan bütün varlıkların özellikleri ve bu özelliklerin en uygun veri tipi belirlenir. Özellikler arasından anahtarlık koşulunu sağlayanlar belirlenir. Normalizasyon kuralları uygulanır. İlişkiler ve ilişki kuralları belirlenerek veri modeli oluşturulur.

2.1.1.3 Uygulama Evresi

Bu evre mantıksal evrede gerçekleştirilen tasarımın veritabanı üzerinde uygulama aşamasıdır. Uygulama evresinde data tipleri seçilir, tablolar oluşturulur, kısıtlamalar oluşturulur, tetikleyiciler yazılır, vb. Bu evrede, daha önceden geliştirilmiş tasarım üzerinde yeni birtakım düzenlemeler yapılabilir. Yine bu evrede keşfedilen iş kurallarına göre kısıtlamalar, tetikleyiciler, prosedürler oluşturulur. Veri güvenliği de tasarımın bu evresinde sağlanır.

2.1.1.4 Fiziksel Evre

Veri erişiminin optimize edilme evresidir. Örneğin verinin fiziksel disk üzerinde erişimin optimize edilme amacı ile paylaştırılması işi bu evrede gerçekleştirilir. Bu evrede amaç performansı artırmak ve bunu yaparken de mantıksal tasarım tarafında hiçbir şeyi bozmuyor olmaktır.

2.1.2 İlişkisel Veri Yapıları

Bu bölümde aşağıdaki veritabanı kavram ve yapıları tanıtılacaktır:

- Veritabanı ve Şema
- Tablolar, Satırlar ve Kolonlar
- Anahtarlar

- Metadata
- Olmayan Değerler (NULL değerler)

2.1.2.1 Veritabanı ve Şema

Veritabanı, veri veya bilgilerin saklı tutulduğu ve kullanıcılara bu bilgiye erişim imkanı sağlayan sistemdir. Veritabanının elektronik bir formda olma zorunluluğu yoktur. Kütüphanedeki kart koleksiyonu da veritabanına bir örnek teşkil etmektedir. Veritabanının elektronik formda olması veriye hızlı ve kolay erişim imkanı sağlar. SQL Server veritabanı sunucusunda birden fazla veritabanı olabilir. Bu veritabanlarıherbiri ayrı bir amaç için oluşturulmuş koleksiyonlar topluluğu olarak düşünülebilir.

Veritabanının bir alt kırılımına şema adı verilir. Veritabanı nesnelerini sahiplerine ve/veya temalarına göre gruplandırmak gerekebilir. İşte bu amaçla şemalar kullanılır. SQL Server veritabanında bir objeye erişebilmek için veritabanı ismi ve şema ismi kullanılır.

Örneğin veritabanında bir tabloya erişmek isteniyorsa, bu aşağıda gösterilen yapıda olmalıdır:

VeritabanıAdı.ŞemaAdı.TabloAdı

2.1.2.2 Tablolar, Satırlar, Kolonlar

Tablolar bilgiyi depolamak amacıyla kullanılan nesnelerdir. SQL Server'da depolamanın temel birimi tablolardır. SQL Server tabloları, veritabanında yaratılır ve devamlılıkları burada sağlanır. Bir SQL Server tablosu satır ve sütunlardan oluşur. Bir tablo içindeki satırlar verilerin eksiksiz bir kaydını temsil eder.

Çizelge 2.1' deki gösterilen yapı satır ve sütunlardan oluşan bir tabloya örnek verilebilir.

Çizelge 2.1 Örnek Tablo, Gemiler

ID	Heave	Pitch	VACC	LBP	BWL	D
1	0,0921	0,0433	3,7331	21,375	6,74	4

2	0,0604	0,0274	3,4545	21,375	6,74	4
3	0,0627	0,0243	4,3801	21,375	6,74	4
4	0,0235	0,0135	2,3836	21,375	6,74	4
5	0,0033	0,0075	1,762	21,375	6,74	4
6	0,0093	0,0053	1,7244	21,375	6,74	4
7	0,0138	0,004	1,7834	21,375	6,74	4
8	0,0764	0,0438	3,6327	21,375	6,74	4

Tablo gemilere ait bilgilerin depolandığı bir nesnedir. Tabloda her bir kolon geminin bir özelliğini temsil etmektedir. Her satır ise spesifik bir geminin kaydıdır.

Tablolar, satırlar ve kolonlar farklı veritabanı tasarım evrelerinde farklı isimlendirilebilir.

Bunlar Çizelge 2.2 ’de özetlenmiştir.

Çizelge 2.2 Tablolar, Kolonlar, Satırlar

	Tablo	*Kolonlar*	*Satırlar*
Mantıksal/Kavramsam Evre	Varlık	Özellik	Örnek
Uygulama Evresi	Tablo	Kolon	Satır
Fiziksel	Dosya	Alan	Kayıt

2.1.2.3 Anahtarlar

İlişkisel teoriye göre birbirinin aynısı iki satır olmamalıdır. Her tabloda en az bir anahtar(anahtar bir veya birden çok kolonun biraraya gelmesinden oluşabilir) olmalı ve

bu anahtar tablonun bir satırını temsil etmelidir. RDBMS araçları anahtar değerin tablo içerisinde tekrarlanmasını engellemektedir(tekliği, NOT NULL ve sıralılığı garanti eder).

Çizelge 2.3 Örnek Tablo, Gemiler

ID	Heave
1	0,0921
2	0,0604

Gemiler tablosuna

“Insert into Gemiler(ID, Heave) values(1,0.0921) ”

Komutu ile bir satır ekleyecek olunursa, bu bilgiye tekil olarak erişme yolu yoktur, çünkü tabloda aynı bilgiyi içeren bir satır daha vardır. Eğer daha önce ID kolonu anahtar olarak tanımlanmış olsaydı, yukarıdaki sorgu çalıştığında satır ekleme işlemi başarısızlıkla sonuçlanacak ve satır tekliği her zaman korunmuş olacaktı.

Birincil Anahtarlar: Varlığın birincil belirteci olarak birincil anahtarlar (Primary Key-PK) ’ kullanılır. Ancak bir tabloda anahtar görevini görecek birden çok özellik olabilir. Bu durumda ilk olarak bu anahtarların arasından bir birincil anahtar seçilir

Örneğin bir tabloda kişinin hem TC Kimlik Numarası hem de SSK numarası mevcut ise bunlardan birini birincil anahtar olarak belirlemek mümkündür.

Birincil anahtar bileşkesi bir tabloda ancak bir kere bulunabilir ve tekrarlanamaz. Birincil anahtar bileşkesindeki tüm alanlar “not null” yani içerisinde veri olması zorunlu sütunlar olmak zorundadır. Her tabloda ancak bir tane birincil anahtar bileşkesi olabilir. Fakat tekilliğin sağlanması için bazı durumlarda tablonun birden fazla sütunu birleştirilerek, bileşim birincil anahtar olarak tanımlanır. Bu tip bileşimli ana anahtarlar “Bileşik Ana Anahtar” olarak da isimlendirilir.

Yabancı Anahtarlar: Kaynak ve detay tablolarının kayıtları arasındaki ilişkilerin oluşturabilmesi için kaynak tablonun ana anahtarı yararlanıcı tablosuna gömülür; gömülen bu anahtara "Yabancı Anahtar" adı verilir. Bir başka deyişle detay tablonun yabancı anahtarla kaynak tabloyu referans etmesidir. Yabancı Anahtarı içeren detay tablosu referans eden, kaynak tablosu ise referans edilendir.

Tekil Anahtarlar: Tablodaki bir veya birkaç sütunun birleşmesiyle oluşturulan tekil anahtardır. Tekil anahtar bileşkesi tabloda en az bir kere birincil anahtar üzerinde bulunabilir. Fakat diğer sütunlarda da tekillik şartı verilebilir. Aynı alanda tekillik bir defa verilir, tekrarlamamayı sağlar.

2.1.2.4 Metadata

Metadata, depolan veriyi anlatmak için depolanan verilerdir. Yetkisi olan kullanıcılar, asıl veriye ulaşmak için kullandıkları sorgulama dilini kullanarak Metadata'ya da ulaşabilirler. İlişkisel teoriye göre veri iki kısımdan oluşmaktadır.

Başlık: Kolon isimleri, kolon veri tipleri

Gövde: Tabloyu oluşturan satırlar

SQL Server veritabanında başlık bilgisinin tutulduğu tablolar mevcuttur.

2.1.2.5 Eksik Değerler (NULL Değerler)

İlişkisel teoriye göre değeri olmayan veriye "Null" adı verilir. Null değerler (0 veya boşluk değil), veri tipinden bağımsız olarak değeri bilinmeyen bilgiyi temsil etmektedir.

Null değerine ilişkin bir takım özellikler mevcuttur.

- Null değer ile birleştirilmiş herhangi bir sütun Null sonucu döndürür
- Bütün matematiksel işlemlerle Null değer ile herhangi bir birleşme Null sonucunu döndürür
- Mantıksal karşılaştırmalara Null değer dahil olunca sonuç yanıltıcı olabilir

2.1.2.6 Varlıklar Arasındaki İlişkiler

Varlıklar arasındaki bağlantıya "İlişki" adı verilir. Varlıklar arasında ilişki kavramı olmasaydı, tüm veriyi aynı varlık altında toplamak gerekecek, bu da bir grup verinin gereksizce tekrarlanıyor olmasına sebep olacaktı. Varlıklar arasındaki ilişki yabancı anahtarlar sayesinde sağlanır.

Çizelge 2.4 Örnek Tablo, Personel

Personel ID	**Adı Soyadı**	**Departmanı**
1	Ayşe Öncü UYSAL	1
2	Özgür Arslan	2
3	Ahmet Aktaş	1

Çizelge 2.5 Örnek Tablo, Departman

Departman ID	**Departman Adı**
1	Bilgi İşlem
2	İnsan Kaynakları
3	Muhasebe

Yukarıda verilen çizelgelerdeki "Departman" tablosu ana tablo, "Personel" tablosu detay tablodur. İki tabloyu birbirine "Personel" tablosundaki yabancı anahtar ve "Departman" tablosunda ana anahtar olan "DepartmanID" niteliği bağlamaktadır.

Bir varlıkla ilişkili olduğu diğer varlığın kayıtları arasındaki kurulan ilişki sayısına "Eşleme Sayısı" adı verilir. Eşleşmeler tiplerine göre gruplandırılabilir. A ve B isimli 2 varlık kümesi olduğu düşünürse, A ve B kümeleri arasındaki olası ilişkilerin tipi aşağıdaki gibi özetlenebilir.

Bire-bire ilişki: A varlık kümesi içindeki bir kayıt, B kümesi içindeki sadece bir kayıt ile ilişkili ise "Bire Bire" ilişki tipi söz konusudur.

Birden-Çoğa/ Çoktan-Bire: A kümesi içindeki bir kayıt B kümesi içindeki birden fazla kayıt ile ilişkili ise, bu ilişkiye "Birden-Çoğa/ Çoktan-Bire" ilişki adı verilir. Varlıklar aksi

yönde düşünülürse, B kümesindeki birden fazla kayıt, A kümesindeki sadece bir kayıt ile eşleşebilir. Bu nedenle birde-çoğa veya çoktan-bire aynı tip olarak adlandırılır.

Çoktan-Çoğa: A varlık kümesindeki birden fazla kayıt, B kümesindeki birden fazla kayıt ile ilişkili ise ve bu B kümesindeki birden fazla kayıt A kümesindeki birden fazla kayıt ile ilişkili ise bu eşleşmeye "Çoktan-Çoğa" ilişki tipi adı verilir.

2.1.3 SQL Sorgulama Dili

SQL ilişkisel veritabanlarında veriyi yönetmek için tasarlanan ve geliştirilen ilişkisel bir dildir. ANSI (American National Standards Institute) ve ISO (International Organization for Standardization), SQL dilini standartlaştırmak için birçok çalışma yapmıştır. SQL tüm veritabanı ürünlerinde kullanılmaktadır (SQL Server-T-SQL, Oracle-PL/SQL, vb.). SQL sorgulama dili dört ana bölümde incelenebilir. Bunlar "Veri tanımlama Dili / DDL" (Data Definition Language), "Veri Güncelleme Dili / DML" (Data Manipulation Language), Veri Görüntüleme Dili / VDL" (View Definition Language) ve "Veri Depolama Dili / DSL" (Data Storage Language) dilleridir.

2.1.3.1 Veri Tanımlama Dili

Veri tanımlama dili yani DDL, veritabanı yapısını(veya diğer adıyla metadata'yı) tanımlamak için kullanılan SQL ifadelerine verilen isimdir. Kullanılan en temel komutları

- CREATE DATABASE
- DROP DATABASE
- CREATE TABLE
- DROP TABLE
- ALTER TABLE

Olup bu komutlar kısaca tanımlanmak istenirse:

CREATE DATABASE: Veritabanı sunucusunda veritabanı yaratmak için bu komut kullanılır. Komutun genel formatı şu şekildedir:

CREATE DATABASE<veritabanı adı> ;

DROP DATABASE: Veritabanı sunucusunda yaratılan veritabanını tamamen kaldırmak için bu komut kullanılır. Komutun genel formatı şu şekildedir:

DROP DATABASE<veritabanı adı>;

CREATE TABLE: Yaratılan veritabanı içerisinde tablo yaratmak için bu komut kullanılır.

Komutun genel formatı şu şekildedir:

CREATE TABLE GEMİ (...);

Parantez içinde kolon tanımlamaları olmalıdır. Kolon tanımlamaları için kullanılan ifade aşağıdaki şekildedir:

ID int

Bu komut ID isimli "integer" veri tipinde bir kolon tanımlar. Kolonlar komut içerisinde birbirlerinden virgülle ayrılırlar. Gemi tablosunun CREATE TABLE komutu aşağıdaki gibidir.

CREATE TABLE [dbo].[Gemiler](

[ID] [int] **IDENTITY**(1,1)NOTNULL,

[LamL] [float] NULL,

[Fn] [float] NULL,

[fn2] [float] NULL,

[Heave] [float] NULL,

[Pitch] [float] NULL,

[VACC] [float] NULL,

[LBP] [float] NULL,

[BWL] [float] NULL,

[D] [float] NULL,

[T] [float] NULL,

[LCB] [float] NULL,

[LCF] [float] NULL,

[BMT] [float] NULL,

[BML] [float] NULL,

[CWP] [float] NULL,

[CP] [float] NULL,

[CM] [float] NULL,

[CB] [float] NULL,

[CVP] [float] NULL,

CONSTRAINT [PK_Gemiler] **PRIMARY KEY CLUSTERED**

(

[ID] **ASC**

)**WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE =**

OFF, IGNORE_DUP_KEY=**OFF, ALLOW_ROW_LOCKS = ON,**

ALLOW_PAGE_LOCKS = ON)**ON** [PRIMARY]

)**ON** [PRIMARY]

DROP TABLE: Veritabanı içinde yaratılan tabloyu kaldırmak için bu konut kullanılır. Komutun genel formatı şu şekildedir:

DROP TABLE<tablo adı>

ALTER TABLE: Bu komut veritabanı içerisinde yaratılan bir tablonun yapısıyla ilgili değişiklik yapmak için kullanılır. Tabloya kolon eklemek veya silmek, ya da var olan kolonun veri tipini değiştirmek için kullanılır.

Kolon eklemek için genel format:

ALTER TABLE<tablo adı>**ADD**<alan adı><veri tipi>

Kolon kaldırmak için genel format:

ALTER TABLE<tablo adı>**DROP**<alan adı>

Kolon üzerinde değişiklik yapmak için genel format:

ALTER TABLE<tablo adı>**MODIFY** <alan adı><yeni veri tipi>

2.1.3.2 Veri İşleme Dili

Veri işleme dili yani DML, veritabanı içerisindeki veriyi güncellemek için kullanılan SQL ifadelerine verilen isimdir. Burada güncellemekten kasıt, veriyi değiştirmek, silmek ve eklemek gibi işlemlerin gerçekleştirilmesidir.

Kullanılan en temel DML ifadeleri:

- İNSERT
- UPDATE
- DELETE

INSERT: Tabloya veri eklemek için kullanılır. Genel yazım formatı şu şekildedir:

INSERT INTO<tablo adı> (<kolon1>,<kolon2>,<kolon3>,...)

VALUES (<kolon4>,<kolon5>,<kolon6>);

UPDATE: Tablolardaki verileri güncelleştirmek için UPDATE komutu kullanılır:

UPDATE<tablo adı>

SET<kolon1> = <değer1>, <kolon2> = <değer2>, ...

WHERE<kriter>

DELETE: Tablolardaki verileri silmek için DELETE komutu kullanılır:

DELETE FROM<tablo adı>

WHERE <kriter>

2.1.3.3 Veri Görüntüleme Dili

Veri Görüntüleme Dili yani VDL, veritabanı içerisindeki veriyi görüntülemek için kullanılan SQL ifadelerine verilen isimdir.

Kullanılan en temel VDL ifadeleri

- SELECT

SELECT: Tablo veya tablolardaki verileri getirmek için kullanılır. Genel yazım formatı şu şekildedir:

SELECT <kolon listesi>

FROM<tablo listesi>

WHERE<arama koşulları>

Yukarıda belirtilen VDL komutları tek başlarına kullanılmazlar, komutların işlevlerini yerine getirmek üzere bazı yardımcı deyimlerden yararlanılır. Bu deyimlerden bazılarını özetlenecek olursa:

FROM: FROM cümleciği SELECT sorgusunu karşılayacak olan sütunların hangi tablolardan alınacağını belirler. SELECT deyimi içerisinde tablolara, görünümlere, türetilmiş tablolara ve tablo değişkenlerine her başvurulduğunda FROM anahtar sözcüğüne ihtiyaç duyulur.

WHERE: WHERE anahtar bir koşul belirterek sadece o koşula uyan kayıtların getirilmesini sağlar. Sorguda kayıtların verilen koşul veya koşullarla sınırlandırılarak getirilmesidir. Verilen koşul doğru ise verileri getirir. Eğer bir koşul verilmezse tablo veya görüntü üzerinde tüm kayıtlar getirilir.

GROUP BY: GROUP BY kayıtların gruplamasını sağlar. Sonuç kümesini alır ve onu gruplandırır. Gruplama yapılırken grup içerisinde istenilen hesaplamalar yapılabilir.

HAVING: Sorguda gruplanmış kayıtların istenilen şekilde sınırlandırılmasıdır. İstenen koşul doğru ise verileri getirir. Buradaki amaç gruplamada ki hesaplamalar üzerinde

sınırlama yapmaktır. Eğer bir koşul verilmezse tablo veya görüntü üzerinde tüm kayıtlar gruplanarak getirilir.

SQL bir dil olduğuna göre, doğal olarak bazı işleçlerinin kullanılmasına olanak sağlamalıdır. SQL ile kullanılacak mantıksal ve karşılaştırma işleçleri AND, OR ve NOT biçimindedir.

AND: Seçme işleminin iki ayrı koşulun birlikte gerçekleşmesi durumunda yapılacaktır.

OR: Koşulların biri gerçekleştiğinde belirtilen seçme işlemi yapılacaktır.

NOT: Koşulun gerçekleşmemesi durumunda yapılacak seçme işlemini tanımlar.

SQL'de verilen koşullar içerisinde aşağıda belirtilen karşılaştırma operatörlerinden de yararlanılır. Bunlar Çizelge 2.6 'da tanımlanmıştır.

Çizelge 2.6 SQL Karşılaştırma Operatörleri

<	*Küçüktür*
<=	Küçük eşittir
>	Büyüktür
>=	Büyük eşittir
<>	Eşit değildir
BETWEEN	İki değer arasında
LIKE	Belirtilen değere benzerlik
IN	Belitilen değerin içinde geçme

2.1.4 Fonksiyonel Bağımlılık

Bir niteliğin değeri bir ya da başka bir nitelik (veya nitelik kümesi) tarafından ifade edilebiliyor ise buna fonksiyonel bağımlılık denir. A,B ilişkisinde her bir A değeri bir B değerine işaret ediyor ise "B A'ya fonksiyonel olarak bağımlıdır" denilebilir. A→B ilişkisinde, A B'yi fonksiyonel olarak tanımlar. Fonksiyonel bağımlılık %100 güven esasına dayanır.

Eğer güven %100 değil ise buna "Yaklaşık Bağımlılık" adı verilir. Yaklaşık bağımlılıkta, bazı kayıtlar belirtilen bağımlılığı bozacak istisnai durumlar içerir.

2.1.5 Bütünlük Kısıtları

Bütünlük kısıtları, verilerin bir nitelik için kabul edilebilir olmasını zorunlu tutar. Bütünlük kısıtları önceden tanımlanmış kurallar gibidir. Veri bütünlüğü kontrolü veritabanında depolanan verinin doğruluğunu, tutarlılığını kontrol altına almak amacıyla yapılır. Veri bütünlüğü SQL Server tarafında veya UI tarafında sağlanabilir. Ancak en çok kullanılan yöntem kısıtlar ve tetikleyicilerdir. Kısıtlar tabloya eklenen kurallardır. Otomatik olarak RDBMS tarafından uygulanırlar.

2.1.6 Veritabanlarında Birliktelik Kuralları

Birliktelik analizi, veritabanı içerisinde birlikte görülen kümelerden birliktelik kuralları keşfetme işlemine verilen isimdir. Birliktelik kuralları, geçmiş deneyimlerden yararlanarak, yeni bilgi keşfi ve bilgi analizi açısından oldukça önemli bir veri madenciliği metodudur. En çok pazar-sepet analizinde kullanılmaktadır. Metodun temeli olayların birlikte gerçekleşme durumlarını çözümleme esasına dayanmaktadır.

Birliktelik kuralları şu şekilde ifade edilebilir:

$$A \rightarrow B,$$

$$A1, A2,.....,Am \rightarrow B1, B2, ..., Bn$$

A→B, veritabanı içerisinde A'yı sağlayan satırlar, B'yi de sağlar anlamını çıkarabiliriz.

2.1.6.1 Destek-Güven Mekanizması:

Birliktelik kuralları oluşturulurken Destek-Güven mekanizması kullanılmaktadır.

Destek-Güven mekanizması şu şekilde özetlenebilir:

1.Kurallar oluşturulurken "Destek" değeri belirlenen en küçük eşik-destek değerinden büyük veya eşit olmalıdır:

Destek>=Minimum(Destek-Eşik değeri)

2.Kurallar oluşturulurken "Güven" değeri belirlenen en küçük güven-eşik değerinden büyük veya eşit olmalıdır:

Güven>=Minimum(Güven-Eşik değeri)

“Destek” ve “Güven” değerlerinin hesaplanış şekilleri ise şu şekildedir:

$$\text{Destek}(A \rightarrow B) = \text{Toplam Sayı}(AUB)/N$$

Buradaki Toplam Sayı(AUB), A ve B’nin birlikte gerçekleşme sayısını gösterirken N kayıt sayısını göstermektedir.

$$\text{Güven}(A \rightarrow B) = \text{Toplam Sayı}(AUB)/A$$

Burada Toplam Sayı(AUB), A ve B’nin birlikte gerçekleşme sayısını gösterirken, A’nın gerçekleşme sayısını göstermektedir

Örnek: Bir personel veritabanında

Yaş (X, “40..50”) ^ Gelir (X,”>5000 TL”) → unvanı (X, ”Müdür”)

Birliktelik kuralı (%2 Destek,%60 Güven sınırları içerinde gerçekleşmektedir)

Kuralın anlamı, veritabanı içerisinde yaşı 40 ile 50 arasında olan ve geliri 5000 TL’nin üzerinde olan kişilerin unvanı çoğunlukla müdür olmaktadır.

Burada destek değeri olan %2:

Yaş (X, “40..50”) ^ Gelir (X,”>5000 TL”) → unvanı (X, ”Müdür”)

Kişilerin veritabanında görülme yüzdesi yani toplam kayıt sayısına oranının yüzde ile ifadesidir.

Güven değeri olan %60:

Yaş (X, “40..50”) ^ Gelir (X,”>5000 TL”) → Unvanı (X, ”Müdür”) olan kayıt sayısının, Yaş (X, “40..50”) ^ Gelir (X,”>5000 TL”) olan kayıt sayısına oranının yüzde ile ifadesidir.

Birliktelik kurallarının gücünü hesaplanan “Destek” ve “Güven” değerleri belirlemektedir. Bu değerler ne kadar yüksekse birliktelik kurallarının o kadar güçlü olduğuna karar verilir.

Kural öğrenimine ilişkin farklı yaklaşımlar mevcuttur, bunlardan en sık kullanılanı Apriori algoritmasıdır. Bu teknik “Yaygın bir nesne kümesinin tüm altkümeleri de yaygın olmalıdır” kuralına dayanmaktadır. Bu algoritmada temel yaklaşım “Eğer k-öğe kümesi minimum destek metriğini sağlıyorsa bu kümenin alt kümeleri de minimum destek metriğini sağlar” şeklindedir.

2.2 Anlamsal Sorgu Optimizasyonu

Veritabanlarının büyümesiyle, veritabanlarını kullanan programların veritabanından bilgi alım süreci daha çok zaman almaya başlamış dolayısıyla veritabanlarında bilgi keşfi oldukça önemli bir hal almıştır. Keşfedilen bilgiler veritabanı performansını artırmak için kullanılabilir, bu kullanımda veritabanı sistemlerinin daha akıllı hale getirebilir.

SQO'nun amacı orijinal sorguyu aynı sonuç setini döndüren farklı bir forma dönüştürerek daha hızlı ve verimli sonuç alınmasını sağlamaktır.

SQO'yu dört ana modülde inceleyebiliriz, bunlar

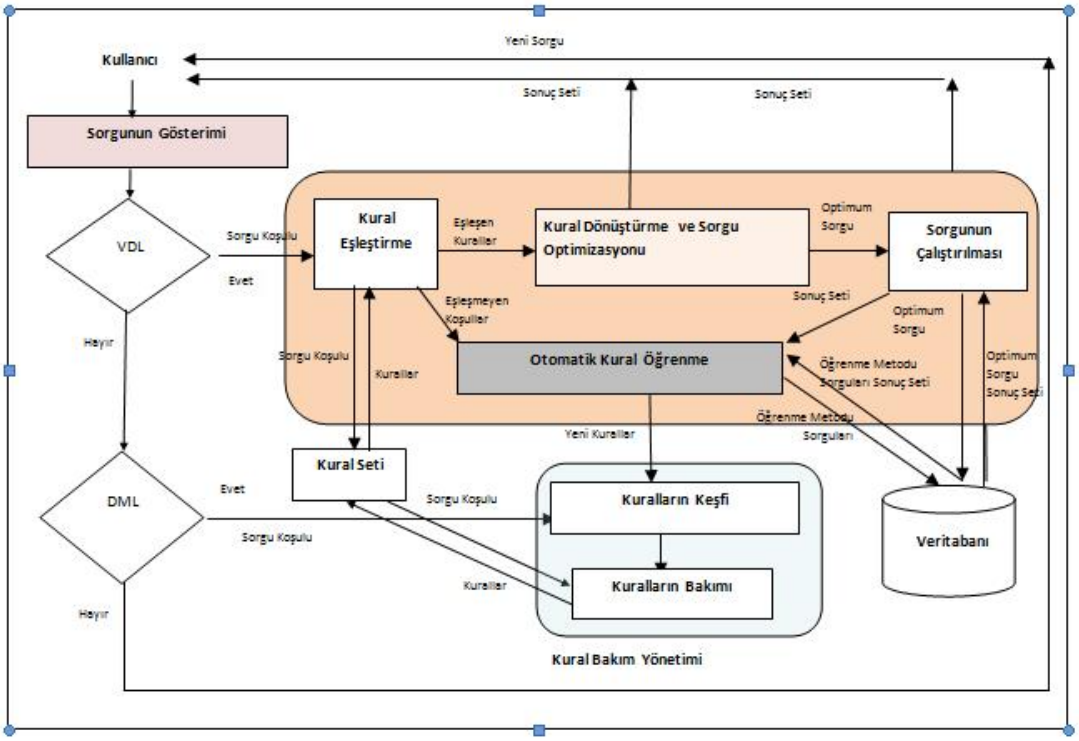
- Sorgu Gösterimi
- Sorgu Optimizasyonu
- Otomatik Kural Öğrenme
- Kuralların Bakımı

Bütün sistemin oluşturulabilmesi için bu dört modül birlikte düşünülmelidir. Ancak her modül için ayrı metotlar mevcuttur, her bölüm ayrı bir araştırma konusu olabilir. Bu tez çalışmasında yoğun olarak otomatik kural öğrenme modülü için iyileştirme çalışmaları amaçlanmıştır, gerçekleştirilen çalışmalar Bölüm 5 'de detaylı olarak anlatılacaktır. Bu bölümde SQO yaklaşımının ne demek olduğu, yaklaşımın modülleri, metotları ve konuyla ilgili daha önce gerçekleştirilen çalışmalar kısaca anlatılacaktır.

2.2.1 Anlamsal Sorgu Optimizasyonuna Genel Bakış

SQO, anlamsal bilgiyi kullanarak, kullanıcının verdiği sorguya alternatif bir sorguyu oluşturarak aynı sonucu veren ve daha verimli çalışan bu alternatif sorguyu çalıştıran sorgu optimizasyonu yaklaşımıdır.

Aşağıdaki şekilde SQO'nun genel olarak modülleriyle birlikte bir resimsel gösterimi verilmiştir.



Şekil 2.2 Anlamsal Sorgu Optimizasyonuna Genel Gösterimi

Şekil 2.2'de gösterildiği üzere SQO'ya giren sorgu, SQL sorgulama diliyle ifade edilen kullanıcının sorgusu sorgu optimizatörün kullandığı iç sorgu diline (ağaç veya cebirsel forma) dönüştürülmelidir. Buna "Sorgunun Gösterimi" adı verilmiştir. Eğer sorgu görüntü tanımlama dili (SELECT komutuysa) ise kural seti bu sorgu ile ilgili tüm kuralları bulmak amacı ile (orijinal veritabanına erişmeden) taranır. İlişkili kurallar bulunduktan sonra anlamsal dönüştürme işlemi yapılır. Dönüştürme işleminin ardından en uygun (maliyeti en az olan) sorgu, optimum sorgu olarak belirlenir. Optimum sorgu çalıştırılır. Eğer gelen sorguya uygun herhangi bir eşleşme(uygun kural) bulunamadıysa, sorgu daha sonraki sorgularda kullanılmak üzere yeni kuralların türetilmesi için kullanılabilir. Bu durumda koşullar "Otomatik Kural Öğrenme" modülüne girer. Eğer gelen komut veri güncelleme dili (INSERT, UPDATE, DELETE komutuysa) ise, kuralların doğruluğunu korumak amacıyla "Kural Bakımı" modülü devreye girer.

2.2.2 Kurallar

SQO yaklaşımındaki kuralları, geçmiş sorgulardan elde edilen koşul veya kısıtlar kullanılarak üretilen kurallar olarak düşünebiliriz. Bu tezde bahsedilen kurallar Bölüm

2.1.6'de ayrıntılı olarak bahsedilen birliktelik kurallarıdır. Bölüm 2.1.5' de ayrıntılı olarak bütünlük kısıtlarından bahsedilmişti yine de kısaca değinmek gerekirse, bütünlük kısıtları, verilerin bir nitelik için kabul edilebilir olmasını zorunlu tutar. Bütünlük kısıtları önceden tanımlanmış kurallar gibidir. Kurallardan farkı veri bütünlüğünü korumazlar, sadece veritabanı ortamını karakterize ederler. Çoğu SQO araştırmacısı araştırmalarında kurallar yerine bütünlük kısıtlarını kullanmışlardır.

Araştırmacılar verilen sorguya ilişkin sonucun bulunabilmesi için SQO yaklaşımındaki kuralları örnek göstermişlerdir([5-17]).

Kuralların gösteriminden bahsedilecek olursa, bir veritabanındaki iki ilişki ele alınsın ve bunlar R(x, y, z) ve S(w, v, y) olsun. Burada x, y, z R'nin özellikleri ve w, v, y ise S'nin özellikleridir. Buradan $R.x \oplus x1 \rightarrow S.w \otimes w1$ formunda bir kural elde edebilir. Burada x1, x'e ait herhangi bir değer, w1 ise w'ye ait herhangi bir değerdir. $\oplus$ ve $\otimes$ operatörleri ise (<,<=,>,>=,=,!=) değerlerinden herhangi birini ifade ederler. Kuralın sol tarafı sağlandığında, sağ tarafı da doğru olmalıdır. Genel ifadeyle sol taraf "Öncül Koşul", sağ taraf ise "Sonuç Koşul" olarak adlandırılır. x öncül özellik, x1 öncül değer ve $\oplus$ öncül operatördür. Aynı şekilde w sonuç özellik, w1 sonuç değer ve $\otimes$ sonuç operatördür. Bu şekilde gösterilen kurallara "basit kural" adı verilmiştir. Bu tez çalışmasında basit kurallar kullanılmıştır. Basit kurallarda, öncül ve sonuç koşullar yalnız birer koşul içermektedir. Eğer birden fazla koşul olsaydı, bu koşullar birbirleriyle 'Λ' ve '$\vee$' operatörleriyle birleştirilirlerdi ve kompleks kurallar oluşurdu.

2.2.3 Anlamsal Olarak Denk Sorgular

Anlamsal olarak denk sorgular SQO'nun kalbi olarak adlandırılır (King [6], [7]). SQO'nun ana fikri, verilen sorguyu anlamsal olarak dönüştürerek alternatif sorgular oluşturmak ve bu sorgulardan birini optimum sorgu seçerek verilen sorguyla aynı sonuç setini elde etmektir. Bu dönüşüm, verilen sorgunun arama alanı genişletilerek alternatif sorgular içerisinde en az maliyetli sorguyu bulmak için fırsat yaratır.

Örneğin veritabanında Personel tablosuna ait aşağıdaki iki kurala sahip olunsun.

DepartmanAdi = "Bilgi İşlem"→Maas > 3000

DepartmanAdi = "Bilgi İşlem"→DepartmanKodu = '004'

İlgili veritabanına DepartmanAdi “Bilgi İşlem” olan kayıtları aramak için aşağıdaki sorgu gönderilirsin:

SELECT * FROM PERSONEL WHERE DepartmanAdi = ’Bilgi İşlem’;

Yukarıda verilen iki kural kullanılarak bu sorgu için anlamsal olarak denk 3 farklı alternatif sorgu oluşturulabilir. Bunlar,

Alternatif Sorgu 1:

SELECT * FROM

PERSONEL

WHERE DepartmanAdi = ’Bilgi İşlem’ and Maas > 3000;

Alternatif Sorgu 2:

SELECT *

FROM PERSONEL

WHERE DepartmanAdi = ’Bilgi İşlem’ and DepartmanAdi = ’004’;

Alternatif Sorgu 3:

SELECT *

FROM PERSONEL

WHERE DepartmanAdi = ’Bilgi İşlem’ and Maas > 3000 and DepartmanKodu = ’004’;

Yukarıdaki üç sorgu yazınsal olarak farklı ancak anlamsal olarak denk sorgulardır, yani aynı sonuç kümesini döndürmektedirler.

Anlamsal olarak ve mantıksal olarak denk sorgular arasındaki farkın altını çizmek gerekir. Mantıksal denklik mantıksal bir takım eşitlikler sonucu oluşur, ancak anlamsal denklik o andaki veritabanının durumuna göre şekillenir. Bu durumda denilebilir ki mantıksal olarak denk sorgular anlamsal olarak da denktir ancak her anlamsal olarak denk olan sorgu mantıksal olarak denk olmayabilir.

2.2.4 Anlamsal Sorgu Optimizasyonunun Ana Modülleri

2.2.4.1 Sorgu Gösterimi

Bir sorgu kullanıcı tarafından veritabanına gönderilirken onun söz konusu veritabanı tarafından anlaşılabilir olması için bir dil ile ifade edilmesi gerekir, bu dil "Yapısal Sorgu Dili (Structured Query Language) / SQL" olarak adlandırılmıştır. Bu tezde Bölüm 2.1 'de detaylı olarak bahsedilen SQL sorgulama dili kullanılmıştır.

2.2.4.2 Sorgu Optimizasyonu

Bir sorguyu alternatif sorgular oluşturacak şekilde dönüştürebilmek için kullanılan kural setinde bir çok kural vardır. Bu kuralların hepsi her zaman yararlı olmayabilir. Bazıları sadece bazı sorgular için yararlı olurken bazıları da ya hiç yararlı olmayabilir ya da bazılarına göre daha az yararlı olabilir. Bu sebeple kuralların türetilmesinin ardında veya öncesinde onların nasıl ve hangisinin kullanılacağına karar verilmesi gerekir. Bu bölümde sorgu optimizasyonu üç ana bölüm altında incelenerek anlatılacaktır: "Uygun Kuralları Belirleme", "Anlamsal Dönüştürme" ve "Optimum Sorgu Seçimi".

Uygun Kuralları Belirleme: SQO'nun bu bölümü sorguda verilen koşula göre kural setindeki kuralları bulmak üzerinedir. Sorgu koşulu kural tablosundaki öncül koşullarla karşılaştırılır. Eğer bir eşleşme var ise kural, sorgu prosesinde ve optimizasyonunda kullanılır. Kural eşleştirme aşamasında iki özel durum oluşabilir. Bunlardan biri "Sorgunun çürütülmesi" diğeri ise "Sorgunun cevabının kural seti içerisinde bulunması" durumlarıdır. Bu durumların kısa açıklaması aşağıdaki gibi yapılabilir:

Sorgunun çürütülmesi: Bu durumda eşleşen kuralların bulunması aşamasında, sorguyu çürüten bir kurala rastlanır. Bu oldukça önemlidir çünkü bu durumda sorgunun proses edilmesine gerek kalmamış olur. Sorgu çürütülmesi bir çok SQO araştırmacısı tarafından tanımlanmış ve nasıl olacağı gösterilmiştir (Hsu ve Knoblock, Lowden ve diğerleri, Siegel, Sayli ve Lowden [13-21]).

Örneğin veritabanına aşağıdaki sorgu gönderilecek olursa:

```
SELECT *
FROM PERSONEL
```

WHERE DepartmanAdi =’Bilgi İşlem’ and DepartmanKodu =’001’;

Yukarıda verilen sorgunun şartları için Bölüm 3.3 ‘te belirttiğimiz kurallar hala geçerli olsun. Bu sorgu,

DepartmanAdi = ‘Bilgi İşlem’ →DepartmanKodu = ’004’

kuralı tarafından çürütülür. Bu da demektir ki sorgu sonuç döndürmez, veya başka bir deyişle sonuç “NULL” döndürür. SQO’nun veritabanına erişmesine gerek kalmadan işlem sonlandırılmış olur.

Sorgunun cevabının kural seti içerisinde bulunması: Bu durumda sorgunun cevabı direkt olarak kural setindeki herhangi bir kuraldan yararlanarak bulunur. Örneğin veritabanına aşağıdaki sorgunun gönderildiği düşünülürse:

SELECT DepartmanKodu

FROM PERSONEL

WHERE DepartmanAdi = ’Bilgi İşlem’;

Kural setinden departmanı “Bilgi İşlem” olan değerin Departman Kodu=’004’ bilgisine ulaşılır. Dolayısıyla yine SQO’nun veritabanına erişmesine gerek kalmadan işlem sonlandırılmış olur.

Anlamsal Dönüştürme: Anlamsal dönüştürme kuralları kullanarak sorguyu alternatif sorguya dönüştürme işlemine denir. Bu işlem esnasında 2 metot kullanılabilir. Bunlar “Koşul Eklenmesi” ve “Koşul Çıkarılması” olarak adlandırılır:

Koşul Eklenmesi: Kural tablosunda eşleşen bir kural bulunduğunda, sonuç koşullar orijinal sorguya eklenebilir.

Koşul Çıkarılması: Kural tablosunda eşleşen bir kural bulunduğunda, bulunan sonuç eklendikten sonra sorgudaki koşul sorgudan çıkarılabilir.

Örneğin, verilen sorgu

SELECT *

FROM PERSONEL

WHERE DepartmanAdi =’Bilgi İşlem’ and DepartmanKodu = ’001’;

olsun. Koşul eklenmesine örnek verecek olunursa, daha önceki bölümde verilen kuralın sonuç kısmı sorguya eklenebilir:

SELECT *

FROM PERSONEL

WHERE DepartmanAdi = ’Bilgi İşlem’ and DepartmanKodu =’001’ and MAAS > 3000;

Koşul çıkarılmasına örnek olarak da daha önceki bölümde verilen kurala istinaden “DepartmanKodu = ’001’ ” ve “MAAS > 3000” koşulları sorgudan çıkarılabilir.

SELECT *

FROM PERSONEL

WHERE DepartmanAdi = ’Bilgi İşlem’;

Sorgu Seçimi: Burada amaç en uygun maliyetli sorguyu seçmektir. Burada farklı bazı sezgisel yaklaşımlar ve maliyet hesaplama teknikleri söz konusudur. İlk olarak sezgisel yaklaşım tanımlanırsa, bu yaklaşımda kural setindeki bütün kuralları kullanmak her zaman için pratik sonuç vermeyebilir. Bu durumda kural setine bazı limitler koymakta fayda vardır. Örneğin, bir sorguya indeks özelliği taşıyan bir koşulun ilave edilmesi, daha az maliyetli ve daha verimli çalışan bir sorgu yaratabilir. SQO’da genel olarak üç sezgisel yaklaşımdan bahsedilebilir:

İndeks Ekleme: Eğer bir ilişkide A özelliğine ilişkin kural aranıyorsa ve bu ilişki de B özelliği indeks ise, kural tablosunda sonuç koşulu B olan kurallar aranır.

Tarama Azaltılması: Eğer R ilişkisindeki tek koşul JOIN koşulu ise ve R ilişkisinin özellikleri indeks içermiyor ise bu ilişki üzerinden seçim koşulu aranabilir. Yeni seçim koşulu JOIN ifadesindeki seçimden çıkarılabilir.

Seçim Azaltılması: Eğer A özelliği üzerindeki koşul, B özelliği üzerindeki koşul ile ifade edilebiliyor ise ve A indeks değil ise A özelliği sorgudan çıkarılabilir.

Maliyet tahmini yaklaşımına göre seçim, bütün alternatif sorgular belirlendikten sonra en az maliyetli olanı seçme esasına dayanır. Bu problem sorguların maliyetlerini tahmin etmek suretiyle ve bu tahmine istinaden seçim yapmak suretiyle çözülür. Siegel, bu maliyetin geçmiş deneyimlerden yararlanarak hesaplanmasını önerir (Siegel [18],[19]).

2.2.4.3 Otomatik Kural Öğrenme

SQO yaklaşımı daha iyi performans sağlamak için kuralları kullanır. Ancak bu kuralların daha önceden öğrenilmeleri gerekmektedir. Bu durumda bu kuralların nasıl öğrenildiğini incelemek gerekir. Temel düşünce, kuralları otomatik olarak veritabanına gönderilen sorguya göre öğrenmektir. Açıktır ki, SQO yaklaşımının performansını kural öğrenme belirlemektedir. Gelen sorgu sonucunda, bazı durumlarda hiçbir kural öğrenilmeyebilir, bazı durumlarda ise öğrenilen kurallar daha sonraki sorguların performansı açısından oldukça büyük bir önem arz edebilir. Kural türetilmesi SQO'nun en önemli bileşenlerinden birisidir. Bu tezde yapılan çalışmanın ana amaçlarından biri de kural öğrenme aşamasında iyileşme yapmaktır. Bu sebeple bazı istatistiksel yöntemler kullanılmış ve kural öğrenme aşaması optimize edilmeye çalışılmıştır. Kullanılan yöntemler ve sonuçları Bölüm 4'ün konusunu oluşturmaktadır ve öne sürülen öğrenme metodu Bölüm 5'de detaylı olarak anlatılacaktır.

2.2.4.4 Kuralların Bakımı

Otomatik kural öğrenme metotlarından herhangi birini kullanarak kural öğrenmesi yapılırken zaman içerisinde öğrenilen kuralların yeni öğrenilen kural öğrenmeleri ile bir geçerlilik değerlendirilmesinin yapılması gereklidir. Bu değerlendirmede bazen kuralların sol ve sağ taraflarında kullanılan sınır değerleri güncellenmeli, bazen de geçersizlik oluşması durumunda doğruluk tespiti yapılarak kural setinden silinmesi sağlanmalıdır. Başka bir deyişle kural setinin sürekli olarak bakımı yapılmalıdır. Bu amaç için çeşitli araştırmacıların ortaya attığı bazı çalışmalar mevcuttur (Hsu and Knoblock, Lowden ve Lim, Schkolnick and Tiberio [13], [14], [15], [17], [22]). Schkolnick ve Tiberio kuralların bakımını veritabanına giriş yapma ve arama maliyeti hesabına dayalı olarak fiziksel kaç bloğa giriş yapılacağını hesaplama üzerine bir metot geliştirmiştir. Hsu and Knoblock ise kuralların ne kadar faydası olduğunu kuralların kullanım sıklığını

istatistiksel olarak derleme ile açıklamaya çalışmıştır. Bu çalışmada nasıl yapılacağı belirtilmesine rağmen metodun bilgisayar uygulaması gerçekleştirilmemiştir. Lowden ve Lim, kuralların sol ve sağ taraflarının sınır değerlerinin güncellenmesi için altı algoritma geliştirmiştir. Geliştirilen metotlarda pek çok hususa dikkat edilmektedir ancak yine de bazı problemler mevcuttur. Bu modül doktora çalışmasında amaç olarak ele alınmamıştır.

BÖLÜM 3

ANLAMSAL SORGU OPTİMİZASYONUNDA KURAL ÖĞRENME METODLARI

Bölüm 2.2 de kısaca tanıtılan sorgu tabanlı kural öğrenmeleri metotları genel olarak bilimsel literatürde iki yaklaşım içerisinde tanımlanmıştır. Bu yaklaşımlar "Sezgiye Dayalı" ve "Veriye Dayalı" olarak iki alt başlıkta isimlendirilebilir. Sezgiye dayalı olarak yapılan öğrenme metotları en fazla Siegel tarafından gerçekleştirilmiştir (Siegel, Siegel vd., [18], [19], [23], [24]). Veriye dayalı olarak yapılan öğrenme metotları ise daha çok Piatetsky-Shapiro ve Knoblock tarafından ortaya atılmıştır (Hsu ve Knoblock, Piatetsky-Shapiro, Arens ve Knoblock, Arens vd., [13-15], [25-28]). Bu tez içeriğinde önerilen istatistiksel kural öğrenme metodu Siegel metodu ile karşılaştırıldığından dolayı Siegel'in metodu Bölüm 3.1'de detaylı olarak açıklanmaktadır. Bölüm 3.2 'de Knoblock tarafından ortaya atılan öğrenme metodu tanımlanmıştır. İstatistik ve olasılığa dayalı kural öğrenme metotlarının çeşitli yayınlarda [Hsu ve Knoblock ,Siegel, Siegel vd.,Piatetsky-Shapiro, Arens ve Knoblock, Arens vd., [13-15],[18],[19],[23-28]) önerilmesi ve adres edilmiş olmasına rağmen belirli bir metot ve bu metodun bilgisayarlı sonuçları açıklanmamıştır. Bu nedenle tez çalışmamızda istatistiğe dayalı öğrenmeler amaç edinilmiştir. Önerilen metot Bölüm 5 'de açıklanmaktadır.

3.1 Siegel Yöntemi

(Siegel, Siegel vd., [18], [19], [23], [24]) tarafından, SQO yaklaşımına otomatik kural türetme için temel bir yöntem sunulmuştur. Bu metot iki bölümlü bir proses olarak düşünülebilir. Birinci bölüm değerli (sisteme daha çok katkı sağlayan) kuralların karakteristiğini tanımlamak, ikinci bölüm ise bu karakteristiğe uygun kuralların

türetilmesidir. Bu iki bölüm SQO yaklaşımında yararlı kuralların bulunmasına öncülük eder. Önerilen kurallar kavramı bu bölümde kendini göstermektedir. Önerilen kuralın öncül koşulu sorgunun herhangi bir durumunu alarak oluşturulur. Bu metot dört bölüm halinde incelenebilir. Bunlar, "Kural Karakteristiklerini Tanımlamak", "Önerilen Kuralları Seçmek", "Sorgu Yaratmak" ve "Kural Yönetimi" modülleridir.

3.1.1 Kural Karakteristiklerini Tanımlamak

Kural karakteristiklerini tanımlama yöntemin ilk modülüdür. Bu modülün amacı, işe yarar kurallar türetmek için sorgu iyileştiricisi tarafından kural türetme işleminin kontrol edilmesidir. Kuralları var olan kurallardan veya doğrudan veritabanından türetebilmek mümkündür. Yine de, eğer türetme işlemi sorgu iyileştiricisine bağlı değilse bu işlem SQO yaklaşımı için gereksiz kuralların bulunmasına neden olabilir. Bu modülde, SQO yaklaşımı beklenen maliyet tasarrufu ile birlikte bütün uyumsuz önerilen kuralları, kural türetme modülüne gönderir ve sonra bu modül tarafından incelenen önerilen kurallar, hesaplanan potansiyel maliyet tasarrufları ile birlikte önerilen kural listesine girer. Eğer bu kural listede mevcutsa birikmiş potansiyel maliyet tasarrufu bu önerilen kural için hesaplanır. Bundan sonra, belli bir eşiğin altındaki birikmiş potansiyel maliyet tasarruflu önerilen kurallar, önerilen kural listesinden silinmelidir. Aksi takdirde, bu önerilen kurallar yeni kurallar gibi algılanır .

3.1.2 Önerilen Kuralların Seçimi

Yöntemin ikinci modülü önerilen kuralları seçmektir. Bu modül türetilebilen en iyi kuralları belirlemede kullanılır. Önerilen kuralların belirlenmesi uzmanlar için çok zor bir iştir. Bu sebeple, gittikçe artan beklenen maliyet tasarruflarını hesaplama, önceki tecrübelere dayandırılır. Potansiyel değeri yeterince yüksek olan uyumlu önerilen kurallardan biri listede yer aldığında, önerilen kurallar işlem için seçilebilir. Bu işlem boyunca, daha düşük potansiyel değeri var olan kurallar, kurallar kümesinden silinecektir. Bu işlem kurallar kümesinin büyüklüğü ve kalitesinin belirlenmesinde kullanılabilir. Aynı amaçla, kural kümesinin en fazla değerine sınır koymak ve bir kuralın en az değerine sınır koymak da yararlı olabilir. Eğer kurallar bu en az değerin altında bir değere sahipse kurallar kümesinden silinecektir.

3.1.3 Sorgu Üretme

Sorgu üretme yöntemin üçüncü modülüdür. Bu modülün girdisini, ikinci modül tarafından belirlenen "Önerilen Kural" oluşturur. Temel olarak, modül önerilen kuralı kullanarak bir sorgu şablonu üretir. Bundan sonra veritabanı üzerinde sorgu çalıştırılır. Bu işlem boyunca ortaya çıkabilecek birkaç durum söz konusudur:

i. Eğer önerilen kural bir koşulun çıkarılması ile bulunduysa sonucu ile ilgili bir koşul olacaktır. Bu koşul üretilen sorguda kullanılmaz.

ii. Eğer önerilen kuralda, iki nesne ilişkisi arasında bir bağlantı her hangi bir sütun ile geliyorsa, bu ilişki koşulu kural oluşturulmasında kullanılmaz.

3.1.4 Kural Yönetimi

Kural yönetimi yöntemin son modülüdür. Bu modül, kurallar kümesine yeni bir kuralın eklenip eklenmeyeceğine karar verir. Modülde, türetilmiş kural, üretilen sorgunun cevabından ve önerilen kuraldan oluşturulur. Eğer cevap hükümsüzse, yeni kural olamaz ve önerilen kural çıkarılır. Eğer önerilen kural bir koşul çıkarılması ile oluşuyorsa, önerilen kuralın sonucu sorgunun cevabına göre kontrol edilmelidir. Bu durumda türetilmiş kuralın sonucu sorgunun olabilecek tüm cevaplarını kullanarak bulunabilir. Bu yöntem, olası tüm kuralların türetilmesinde kullanılabilir. Yine de sadece gerekli kuralları bulabilmek, otomatik kural türetmedeki en önemli faktördür.

3.2 Knoblock Yöntemi

Öğrenme sistemlerindeki asıl amaç, veritabanındaki veri değerlerinin modellerine ve bunların önemlerine göre kurallar türetmek, veritabanında verilen bir sorguda veritabanına mümkünse giriş yapmadan, mümkün değilse en kısa sürede sorgunun sonucunu bulmaktır. Knoblock yöntemi bu amaçla tercih edilen diğer bir öğrenme yöntemidir.

(Piatetsky-Shapiro [25]) tarafından yayınlanan, veritabanlarında bilgi keşfine dayalı araştırma, bir yarı uygun algoritma olan KID3 kullanılarak kuralları öğrenmeyi anlatır. Bilgi keşfi, kural-yarar ölçümleri ve tüm veri kümelerindeki örnek türetme kurallarının

doğruluğu gibi diğer konuları da içerir. Bu konular, istatistiksel yöntemler kullanılarak tanımlanmışlardır. Kesin kurallar ve sağlam kurallar olarak iki gruba ayrılırlar. Kesin kurallar veri tabanında her zaman doğrudurlar ve SQO yaklaşımı için kullanılırlar, sağlam kurallar hemen hemen her zaman doğrudurlar. Bu yüzden kesin kuralların keşfi için KID3 algoritması kullanılır. Bir kesin kuralı öğrenmek için, önceki duruma uyan kayıtların aynı zamanda sonuç durumuna uyup uymadığı kontrol edilmelidir. KID3 algoritması A @ a → B @ b şeklindeki kural için kullanılabilir. Burada 'a' ve 'b' sabitler ve @ karşılaştırma operatörlerinden { <, <=, >, >=, =, != } biridir. Algoritmanın ana fikri, her bir kaydı A ile değiştirmektir. Her bir değişen hücre, tüm kayıtların bir özetini saklamak için kullanılır. Eğer bir kayıt kullanılan bir hücreye dönüştürülürse, hücre özeti, özet ve kayıt arasındaki karşılaştırmaya göre güncellenir. Algoritma aşağıdaki gibidir.

Procedure KID3 (A, file)

```
For Tuple in File
Get Cell corresponding To Tuple.A
If empty Cell Then
Cell.Summary = Tuple, Cell.Count = 1;;Hücre başlangıç durumuna gelir
Else
Cell.Count = Cell.Count + 1;; Kayıt değiştirme
For field C in Cell.Summary
        When Cell.Summary.C != NIL ;; Hücre özeti hükümsüzleştirilir.
        Do<sütun tipi> Summary-Update(Cell.Summary.C, Tuple.C)
        End for
    End If
End For
```

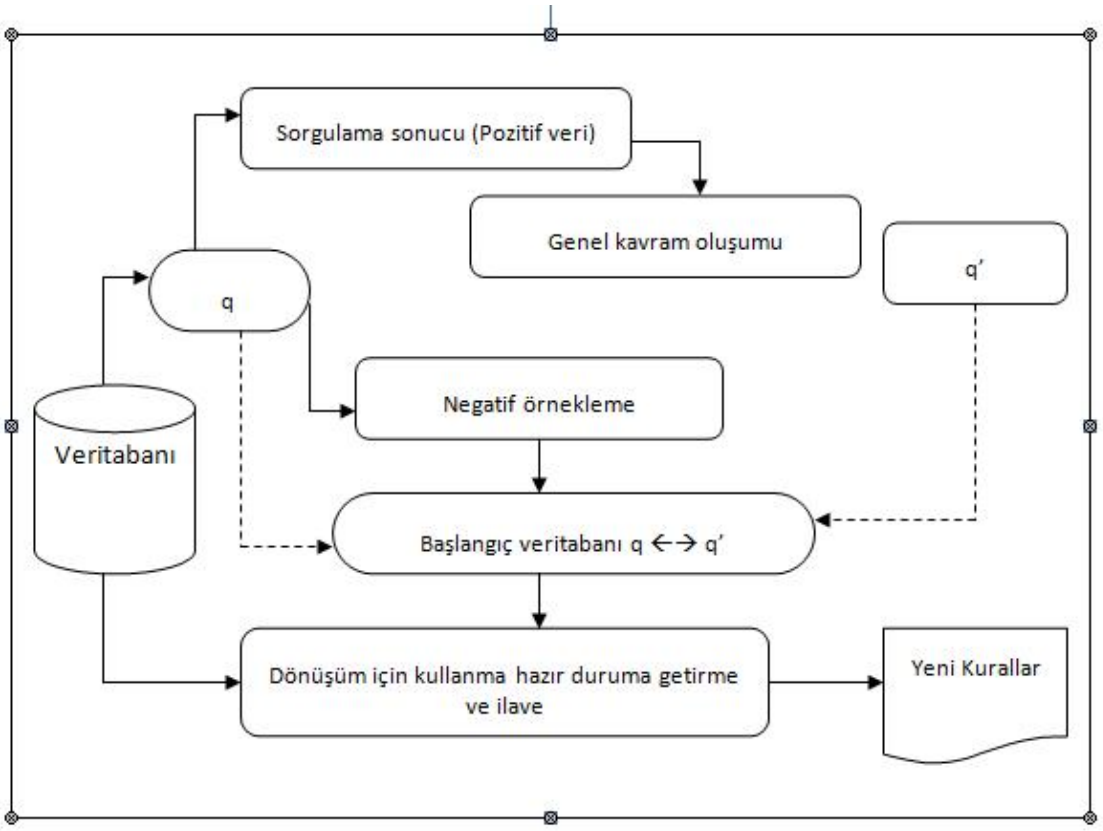
KID3'deki 'Summary-Update(hücre, kayıt)' prosedürü kayıttaki C değeriyle karşılaştırarak o anki hücre özetini bir C alanı için değiştirir. Eğer alanın özeti çok genelse, özet değeri hükümsüzleştirilir ki bu da alanın öğrenmede kullanılmayacağı anlamına gelir. Dizi alanının tipi için, hücre özeti K değerlerine sahiptir. Burada K, alanın

alabileceği değişik değerlerin en fazla sayısıdır. Bu maksimum sayıdan daha fazla K değeri varsa, özet hükümsüzleştirilir. Olağan değerler ve onların frekansları, kompleks özetler için K değerlerinin seçiminde kullanılabilir. 'NUMBER' alan tipi için, hücre, alanın en az ve en fazla değerlerine sahiptir. Kompleks özetler için ortalama değer ve standart sapma gibi değerler elde etmek mümkündür.

Bu aşamada, özetleri kullanarak kuralların nasıl yorumlandığını açıklamak gerekir. Bu hücre sayısı ve her bir hücre için A = a değerli tüm kayıtlarının özetleri kullanılarak yapılır. Her bir geçerli özet alanı C için, alanın tipini bulmak için bir kontrol yapılır. Eğer tip 'CHAR' veya 'VARCHAR2' ise C alanı şöyle yorumlanabilir. C = c1V.....Vck, burada c1,.....,ck ise C alanının değerleridir. Eğer tip 'NUMBER' ise, C alanı şöyle yorumlanabilir. c1 <= C <= c2, burada c1 C'nin en az c2 ise C'nin en fazla değeridir.

Bu algoritmayı değişik tipteki kurallar için genişletmek mümkündür. Örneğin önceki durum a1 <= A <= a2 şeklinde ise yine algoritma uygun kayıt sayısı veya veri dizinleme yapısı seçiciliği temeline dayanmaz. Yapılan tek kontrol, A üzerindeki durumla uyumlu tüm kayıtların, C üzerindeki duruma da uyup uymadığının bulunmasıdır. Bu yüzden tüm kuralların veritabanındaki etkilerini kontrol etmek için bir yoldur. Yani bazı kurallar, kurallar kümesinde tutulmaya değer olmayabilir. Bu gelecekte çok büyük bir kurallar kümesine sahip olmamıza ve verimsiz bir sorgulama işlemine yol açar. Her ne kadar bu araştırma, kuralları analiz etme için kural yararı ölçümlerini verse de, öğrenme işleminde kurallar türetme aşamasında daha seçici olması gereklidir.

Knoblock yöntemi ise, veritabanındaki asıl veri örneklerini kontrol ederek, verilen bir sorgunun durumlarını kullanır. Bunu, aday durumlar üzerinde dizinleme gibi özel veri yapılarını içeren en çok istenilen kuralları belirlemek için yapar. Yöntem, verilen bir sorgunun herhangi bir durumu için veritabanından gelen pozitif ve negatif örneklerle, bu durumun SQO yaklaşımında kaç kere oluştuğu temeline dayanır. Yöntem diğerlerine göre daha seçicidir ve buda en büyük avantajıdır.



Şekil 3.1 Veritabanının Genelini Öğrenme Yöntemi

Şekil 3.1 'deki Knoblock sistemi iki bölümlü bir işlem olarak verilmiştir. İlk bölümde alternatif bir sorgu q', tümevarımlı öğrenme algoritmasıyla veritabanından oluşturulur. Bu alternatif sorgu verilen sorguyla aynı kayıtları gösterir yani anlamsal olarak aynıdır. Ama verilen sorgudan daha verimli ve ucuz bir şekilde işlenebilir. Bu algoritma 3.2.1'de anlatılmaktadır.

3.2.1 Alternatif Sorgu için Tümevarımla Öğrenme Algoritması

P pozitif veri, N negatif veri, S ise verilen veritabanı şemasıdır.

Adım 1: q' hükümsüzleştirilir.

Adım 2: Her bir sütunun (A), değer aralığı R ve P arasında olduğu bulunur.

Adım 3: Her bir sütunun kazancı: gain(x) ve maliyeti: cost(x) hesaplanır.

Adım 4: Eğer kazanç < 0 ise kullanılmaz.

Adım 5: Kazanç / Maliyet oranı en yüksek olan sütunu seçilir.

Adım 6 : q’ = q U {x}; A = A – {x}; N = N – {n|Her n, x’ler hariç n’lerin toplamı N’den}

Adım 7 : Eğer N = Hükümsüz ise geriye q’yü döndür.

değilse Adım 3.’e git.

Where gain(x) = {n|Her n, n’lerin toplamı N ve x’ler elenir.}

cost(x) = E-cost * C-cost

= Değişen cost(x)*(P+N-gain(x)), eğer sütun indeks üzerinde ise

= Değişen cost(x)*(P+N), eğer sütun indeks üzerinde değilse

Bir koşul için kazanç, gain(x) fonksiyonu, durum tarafından elenen örneklerin sayısıdır. Durumun maliyeti, değer maliyeti, E-cost, ile değerlendirilecek örnek sayısının, C-cost, çarpımıyla bulunabilir. E-cost, değerin maliyetidir ve bir örneğin duruma uygun olup olmadığını belirler. Bu maliyet, verilen durum için karşılaştırma koşullarının sayısıyla, her bir karşılaştırmanın maliyetinin çarpımıyla bulunur. Karşılaştırma maliyeti, koşul veri tipine bağlıdır. İndeks tipi için, maliyet veri şemasındaki durum sütunun uzunluğunun tanımıdır. Bu sayı tipi için 2’dir. C-cost, koşulun indekslenmiş bir sütun temelli olup olmadığıyla ilgilidir. Eğer koşul indekslenmiş bir sütun temelliyse, C-cost koşulu karşılayan örneklerin sayısıdır. Yoksa, veritabanındaki örneklerin toplam sayısıdır. Bunun sebebi, veritabanındaki tüm örneklerin koşula değer biçmek için kontrol edilmesindendir.

İkinci bölümde, başlangıç veritabanı iki aşamada işleme konur. Bu aşamalar var olan kuralları formüle etmek için yapılır ki böylece kurallar verilen sorgunun yeniden formüle edilmesinde kullanılabilirler, q → q’: **dönüşüm ve yarar**. Dönüşüm aşamasında, veritabanı başlangıç geneli bir kriteri karşılamak için dönüştürülür, bu kriter kuralın sonuç tarafı ile orantılı bir aralığa sahip olmalıdır. Yarar aşamasında dönüştürülen kurallar başka bir kriteri karşılamak için basitleştirilir, bu kriter ‘Kuralın önceki durum tarafının olabildiğince kısa olmasıdır.’ Daha da fazlası önceki durum tarafını basitleştirmek için, önemsiz koşullar elenir ve bunun için bir algoritma kullanılır. Algoritmayla sonuç tarafındaki negatif örnekler aranır ve sonra eğer koşul sonuç tarafında en çok örneğe sahipse koşulu önceki durum tarafından seçer.

Tümevarımlı öğrenme algoritmasından görüleceği gibi, yöntem sıkça uygulanan ve düşük maliyetli yararlı kuralları öğrenmek için kullanılabilir. Öğrenme işlemi her ne

kadar iyi tanımlanmış olsa da, sistem SQO yaklaşımının kural bakımı ve maliyet değeriyle sorgu iyileştirme gibi diğer bileşenlerine daha az hitap eder.

SQO yaklaşımı için Knoblock'un yöntemine benzer olarak (Lowden ve diğerleri, Lowden ve Lim [16],[17]) tarafından anlamsal kurallar türetmek için ARDOR ortaya atılmıştır. ARDOR öğrenme zamanını azaltan örnek sorgular temeline dayanır. Bu yöntem sistem kullanıcılarının herhangi bir denetim ve müdahalesine gerek duymaz. Kural türetme işlemine bir ilavesi vardır. Bu ilave, yeni bir kural türetildiğinde, kurallar kümesine eklenmeden önce yeni kuralla aynı önceki durum ve sonuca sahip herhangi başka bir kuralın olup olmadığının kontrol edilmesidir. Eğer böyle bir kural varsa yeni kural ve var olan kural önceki koşul ve sonuç koşulunun sınırlarını değiştirmek için kıyaslanırlar. Bu sonuç kuralının düzgün sınırları olmasını sağlar. Yapılan inceleme aynı sonuçla biterse, fakat önceki koşul farklıysa ve var olan kuralın önceki koşulu yeni kuralın önceki koşulu kullanılarak genişletilebiliyorsa, yeni kural var olan kuralın yerini alır. Diğer bir durumda, eğer inceleme aynı önceki koşullarla fakat farklı sonuçlarla biterse ve var olan kuralın sonucunu yeni kuralın sonucuyla sınırlamak mümkünse, yeni kural var olan kuralın yerini alır. Örneğin iki var olan kural {R1, R2} ve iki türetilmiş kural {R1a, R1b} olsun.

R1) project >= 12 --> dcode = 'SALE' R2) dcode = 'SALE' --> project >= 12

R1a) project > 15 --> dcode = 'SALE' R1b) project >= 10 --> dcode = 'SALE'

R1a ve R1b, yeni kurallarıyla karşılaştırarak, R1'in sınırını değiştirecek bir kontrolün yapılmasına gerek duyulduğu görülebilir, çünkü bu kurallar aynı kurallar sınıfına dahildirler. R1 kuralının önceki koşulunu R1a'ya değiştirirsek, kuralın sınırı daha küçülür. Bu yüzden, R1a kuralı pas geçilir. Ama R1'in önceki durumun sınırını R1b'ye değiştirmek sınırı genişletir. Bu yüzden, kural R1b, R1'in yerini alır.

İki türetilmiş yeni kural {R2a, R2b} düşünelim:

R2a) dcode = 'SALE' --> project > 15 R2b) dcode = 'SALE' --> project >= 10

Bu kuralın R2 kuralıyla aynı olduğu görülebilir. Sonuçlar açısından, R2b kuralı pas geçilir ve R2a kuralı R2'nin yerini almak için seçilir, böylece sonucun sınırı en küçük değerine düşürülür. (project => 12'den project >= 15'ye kadar)

3.3 Öğrenme Metotlarının Avantajları ve Dezavantajları

Siegel yönteminin temel yaklaşımı (Siegel [16], [17]) kural türetme, sorgu dönüşümü ve düşük maliyetli sorgu işlemini bulmaktır. Bu yöntemden, önerilen kuralları kullanarak bir çok dönüşüm türetmek mümkündür ve bunların bir kısmı sorgu iyileştirme işlemi için kullanışlı olmayabilir. Bunun için kurallar kümesinin büyüklüğünü sınırlamak çok önemlidir. Başka bir nokta ise yöntem gerçek verilere değil gerçek verilerden oluşturulan verilere bağlıdır. Bu yüzden, yöntem veri şablonlarını kullanarak geliştirilmelidir. Böylece veritabanındaki gerçek veri değerlerinin üzerinde, kuralların etkileri belirlenebilir. Bu yüzden, bu tarz sorunlarla başa çıkabilmesi için temel yöntemi geliştirmek önemlidir. Bu geliştirmeler Siegel'in aynı araştırmasında üç farklı alanda özetlemiştir. Bunlar "Kullanıcıya Özel Alan Bilgisi", "Sezgisel Bilgi" ve" İstatistiksel artış" alanlarıdır.

Kullanıcıya Özel Alan Bilgisi: Otomatik kural türetme sabit bir işlem değildir, başka bir deyişle, çeşitli anlamsal bilgi ve istatistiklere bağlıdır. Temel yöntemde bilginin kullanımı tek bir alan tarafından sınırlandırılmıştır. Yine de, kullanıcıya özel alan ilgisi tarafından analiz edilen ve kullanılan bilgi için KDD ile ortaya atılan birkaç yöntem vardır. Bu alan bilgisi kural sınıflarının bir belirtisi olarak görülebilir ve bu belirtiye göre hangi kural sınıfının anlamlı ilişkiler içerdiğini belirlemek mümkündür.

Sezgisel Bilgi: Temel yöntemi desteklemek amacıyla sezgisel bilgi SQO yaklaşımı için çok önemlidir. Yine de bu bilginin kullanımı da kurallar için sınırlandırılmıştır. Bu kurallar sadece değere bağlıdır. Başka bir deyişle bağımlılık yoktur. Eğer ilişkilerin bağımlılıklarını kapsayan yeni bir sezgi tanımlamak mümkün olursa, temel yöntem gerçekte kullanılan sorgular için daha verimli ve daha uygulanabilir olacaktır. Sezgi, keşfedilmiş sezgilerden (İndeks tanıtma, tarama azaltması gibi) türetilemeyen farklı kural tiplerini bulmak için kullanılabilirler. Eklenen sezgiler, referans kural sütunları,

dinamik kural sütunları, statik kurallar (Özellikle ilişkinin anahtar sütunu için) ve kategori sütunları gibi yararlı kuralların türetilebilmesi için ilişkilerde ne tip sütunların olması gerektiğini keşfedecek şekilde geliştirilmelidir.

İstatistiksel Artış: İstatiksel artış, geçmiş sorguların işleme konulması ile toplanabilen istatistikleri (geçmiş bilgileri) kullanan kural türetme işleminin kalitesini artırmak için kullanılır. Siegel yöntemi, istatistiklerin elde edilmesi durumunda kural türetme yöntemindeki önerilen kurallar için potansiyel maliyet kazancının değerlendirilmesinde 3 faktörün belirlenmesi için bu istatistiklerin kullanılabileceğini işaret eder. Bu üç faktör eski türetmelerin performansı ile her hangi bir bilgiye sahip olmadan sadece şimdiki sorguyu kullanarak bu faktörlerin yöntemle keşfedilememesidir. Türetilebilme faktörü, kural sınıfındaki sütunların ilişkisine göre yapılır. Bakım faktörü kural kümesindeki bir kuralın bakımının ve bozuklukların kontrol edilmesinin maliyetiyle ilgilenir. Bu faktör sonradan kural türetişinin yararlılığı hakkında bir karar vermek için kullanılabilir. Önceki seçicilik faktörü, önerilen kural için önceki koşulunun seçiciliği ile alakalıdır. Eğer önceki koşul çok genişse, önerilen kuraldan gerçekten yararlı yeni bir kural türetmek pek mümkün değildir. Görüldüğü gibi, bu faktörlerin uzmanlar tarafından belirlenebilmesi çok kolay değildir. Çünkü bu faktörlerde işlenmiş sorguların geçmiş performanslarına ve bunların istatistiklerine bağlıdır.

KDD'de veri bağımlılıkları temelli yöntemleri kullanarak yeni kuralları öğrenmek mümkündür. Bu yöntemlerdeki ana fikir, istatistikler ve alan bilgisi aracılığıyla ilişkiler arasındaki veri bağımlılığını ölçmektir. Böylece sağlam kural sınıfları bulabilmek için gereksiz sütunların elenmesi sağlanır. Genellikle yöntemler tek bir kuralın yerine, bir kural sınıfında olmasının test edilmesi temeline dayanır. Çünkü bazı durumlarda kurallar aynı kural sınıfında olmalarına rağmen bir kural diğerlerinden daha verimli olabilir. Bu sebeple kural kümesinin verimli ve verimsiz kurallarla çok fazla büyümesi gerçekleşebilir. Buda SQO'nun performansını azaltır.

(Siegel vd. [23], [24]) Siegel tarafından sunulan, yöntemlere karşılık, (Hsu ve Knoblock [13], [14], [15]) tarafından sunulan, Knoblock temelli yöntem daha verimli kullanılabilir. Çünkü her hangi bir sezgi ile sınırlandırılmamıştır. Yöntem birkaç yönden çok umut vericidir. Bunlardan biri, verilen sorgu yeniden formüle edilir ve sonra yeni kurallar

türetmek için kullanılırlar. Bir başka umut verici yönü ise, veri örnekleri verilen sorgunun koşulları için bu yöntemle kontrol edilir ve her bir koşul için koşulla karşılanan pozitif örnekleri bulup bu örnekler aday koşulların yapımında kullanılır. Sonra negatif örnekleri elemek için en güçlü aday koşulları seçer. Bu eleme, veri tipleri, uzunlukları, dizin yapıları ve bunların oluşum sayılarına bağımlıdır.

(Lowden vd., Lowden ve Lim [16],[17]) tarafından sunulan SQO sistemi, birkaç eklemeyle Knoblock yöntemi' ne benzer bir öğrenme işlemi temeline dayanır.

BÖLÜM 4

DİNAMİK VERİTABANLARINA DAYALI ÇOKLU REGRESYON ANALİZİ

Veritabanlarında saklanan veri istatistiki açıdan oldukça önemlidir. Bu sistemler sayesinde daha çok veri üzerinde uygulanacak olan istatistiksel metotlar, verilerin içerisinde direkt görülemeyen ilişkilerin tespiti ile daha anlamlı bilgilerin bulunmasına imkan sağlamaktadır. Bu amaçla "Çoklu Regresyon Analizi" kullanılarak akıllı bir sistem oluşturulabilir. Bu sistem çeşitli mühendislik alanlarındaki modellemelerde kullanılabileceği gibi veritabanı alanında sorguların optimizasyonu için yeni kuralların öğrenilmesinde kullanılabilir. Bu bölümde öncelikle oluşturulan yazılım sisteminin temelini oluşturan "Çoklu Regresyon Analizi" genel hatlarıyla Bölüm 4.1'de açıklanmıştır. Yazılımın Gemi İnşaatı Mühendisliği alanında gemilerin başlangıç tasarımı için yapılan modellemelerdeki uygulaması Bölüm 4.2'de anlatılmıştır.

4.1 Çoklu Regresyon Analizi

4.1.1 Korelasyon Analizi

İki veya daha çok değişken arasındaki ilişkiye "Korelasyon" adı verilir, korelasyon analizi değişkenler arasındaki ilişkinin gücünü ve yönünü belirlemek amacı ile yapılır. Değişkenler arasındaki ilişkinin derecesini gösteren katsayıya "Korelasyon Katsayısı" adı verilir. "R" sembolü ile ifade edilir. Korelasyon katsayısı daima +1 ile -1 arasındadır. Eğer ilişkinin yönü aynı ise katsayı pozitif, ters yönde ise katsayı negatif çıkar. R değerinin işareti ilişkinin yönünü, mutlak değeri ise ilişkinin gücünü göstermektedir.

$|R| = 1$ ise tam ilişki

$|R| = 0$ ise ilişki yok

$|R|$ değeri 1'e yaklaştıkça ilişkinin kuvveti artar, 0'a yaklaştıkça ilişkinin kuvveti azalır.

4.1.2 Regresyon Analizi

Regresyon değişkenler arasındaki ilişkiyi incelemek amacıyla kullanılan bir analiz yöntemidir. Değişkenler arasında korelasyon bulunduğunda bu ilişki denklem ile ifade edilebilir. Bu denkleme "Regresyon Denklemi" adı verilir.

Eğer regresyon modelinde bir bağımlı değişken ve bir bağımsız değişken bulunuyorsa, burada yapılan analize "Tek Değişkenli Regresyon Analizi" denir. Eğer regresyon modeli bir bağımlı değişken ve birden çok bağımsız değişken içeriyorsa, yapılan analize "Çoklu Regresyon Analizi" adı verilir.

Çoklu regresyon modelleri doğrusal(lineer) ve eğrisel(lineer olmayan) olarak ikiye ayrılır. Bu çalışmada lineer regresyon modeli esas alınmıştır, araştırma ve çalışmalar bu kapsamda gerçekleştirilmiştir.

4.1.3 Çoklu Regresyon Modeli

$\hat{Y}$,tahmin edilen Y değeri; b_0 , tahmin edilen regresyon kesim noktası; $b_1.b_2 \ldots . b_k$, tahmin edilen eğim katsayıları ve $X_1, X_2, \ldots, X_k$, bağımsız değişkenleri göstermek üzere çoklu regresyon modeli aşağıdaki şekilde ifade edilebilir:

$$\hat{Y} = b_0 + b_1 X_1 + b_2 X_2 + b_3 X_3 + \ldots . + b_k X_k \qquad (4.1)$$

Buradaki katsayılar örnek veri kullanılarak hesaplanır.

4.1.4 Çoklu Regresyon Katsayılarının Hesaplanması

Çoklu regresyon modelleri en küçük kareler yöntemi kullanılarak çözümlenebilmektedir. Bilinmeyen sayısı ve denklem sayısı arttıkça parametrelerin kestirimi zorlaşmaktadır. Parametrelerin kestirimi için matrisleri kullanmakla genel bir formüle ulaşılabilir.

Buna göre bağımsız değişken için gözlenen değerleri X, bağımlı değişken için gözlenen değerleri Y matrisleri ve bilinmeyenleri b matrisi olarak düşünüldüğünde X, Y ve b matrisleri aşağıdaki gibi olur:

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix} \quad , \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad , \quad b = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_k \end{bmatrix}$$

b matrisini hesaplanışı şu şekilde olacaktır.

$$b = (X^T X)^{-1}(X^T Y) \tag{4.2}$$

b matrisinin elemanları çoklu regresyon katsayılarıdır.

Hesaplanan bu katsayılar sayesinde regresyon denklemi elde edilmiş olur.

4.1.5 Çoklu Belirlilik Katsayısı

Bağımlı değişken Y'deki toplam varyasyonun tüm X değişkenleri tarafından açıklanan oranını ifade eder. R^2, 0 ve +1 aralığındadır.

$$SST = \sum_{i=1}^{n}(y_i - \bar{y})^2 \tag{4.3}$$

$$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2 \tag{4.4}$$

$$\sum_{i=1}^{n}(y_i - \bar{y})^2 = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2 + \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4.5}$$

Olmak üzere R^2 hesaplanışı şu şekildedir,

$$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST} \tag{4.6}$$

4.1.6 Çoklu Korelasyon Katsayısı

Determinasyon katsayısının pozitif karekökü çoklu korelasyon katsayısıdır. Çoklu korelasyon katsayısı bağımsız değişkenlerle bağımlı değişkenler arasındaki doğrusal ilişkinin derecesini verir.

$$R = \sqrt{R^2} \tag{4.7}$$

4.1.7 Düzeltilmiş Belirlilik Katsayısı (adj R^2)

R^2 modele yeni bağımsız değişkenler eklendikçe yükselme eğilimi göstermektedir. Düzeltilmiş R^2 bağımlı değişken Y'deki toplam varyasyonun tüm X değişkenleri tarafından açıklanan oranını modeldekullanılan X değişken sayısı ile ayarlayarak ifade eder.

$$adjR^2 = (1 - \left[(1 - R^2)(\frac{n-1}{n-k-1})\right] \quad (4.8)$$

Önemli olmayan bağımsız değişkenin modelde fazladan kullanımını cezalandırmak amacı ile kullanılır. Düzeltilmiş belirlilik katsayısı R^2 'den daha küçüktür.

4.1.8 Standartlaştırılmış Regresyon Katsayıları

Regresyon denkleminde değişkenlerin önündeki katsayılar, o değişkenin bağımlı değişken üzerinde etkisi hakkında bilgi sahibi yapmayabilir. Standartlaştırılmış regresyon (tekbiçimleştirilmiş) katsayıları boyutsuz olup önündeki değişkenin bağımlı değişkenler üzerinde göreli etkisini gösterir (Armutlu 2000). Tekbiçimleşmiş modeli ikiden fazla değişken için genelleştirecek olursa

$$S_{kj} = \sum_{i=1}^{n}(x_k - \bar{x}_k)(x_{ij} - \bar{x}_j) \quad (4.9)$$

olmal üzere x_k ve x_j arasındaki doğrusal korelasyon

$$r_{kj} = S_{kj}/(\sqrt{S_{kk}S_{JJ}}) \quad (4.10)$$

ve bu korelasyonların oluşturduğu korelasyon matrisi

$$R = \begin{bmatrix} 1 & r_{12} & r_{13} & \cdots & r_{1k} \\ r_{12} & 1 & r_{23} & \cdots & r_{2k} \\ r_{13} & r_{23} & 1 & & r_{3k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{1k} & r_{2k} & r_{3k} & \cdots & 1 \end{bmatrix}$$

olur. Benzer şekilde k adet bağımsız değişkenin her biriyle bağımlı değişken y arasındaki basit doğrusal korelasyonun oluşturduğu sütun matrisi

$$b = \begin{bmatrix} r_{1y} \\ r_{2y} \\ \vdots \\ r_{ky} \end{bmatrix}$$

olur. Bu iki matrisin yardımı ile k adet α_j parametresinin EKK (En Küçük Kareler Yöntemi) yörtemi ile kestirimi

$$a = R^{-1}g \tag{4.11}$$

eşitliğinden bulunur. Bu kestirimin ardından eğer istenirse orijinal modeldeki katsayı kestirimlerine

$$b_j = a_j \left(\frac{S_{YY}}{S_{JJ}}\right)^{1/2} \quad , j = 1,2, \dots, k \tag{4.12}$$

formülü ile ulaşılır.

4.1.9 Çoklu Regresyon Modelinin Standart Sapması

Çoklu regresyon modelinin standart sapması

$$S_E = \sqrt{\frac{SSE}{n-k-1}} = \sqrt{MSE} \tag{4.13}$$

formülüyle hesaplanır.

4.1.10 Çoklu Regresyon Modelinin Genel Anlamlılık Testi

Çoklu regresyon analizinde modelin anlamlılığını ölçmek için F testi kullanılır. F testi bağımsız değişkenler X ile bağımlı değişken Y arasında ilişki olup olmadığını test eder.

F testi yapabilmek için 2 hipotez önerilir.

Hipotez:

H_0: $\beta_1 = \beta_2 = \cdots = \beta_k = 0$: (lineer ilişki mevcut değil)

H_1 en az bir $\beta_i \neq 0$ (en azından bir değişken Y'i etkilemektedir)

Test sonucunda bir güvenirlik sınırı belirlenir, α ve çıkan F değeri F dağılım tablosunda belirlenen α değeri için tablo değeri ile karşılaştırılır. Karşılaştırma sonucuna göre Ho hipotezi ret veya kabul edilir. Ho hipotezinin reddedilmesi bağımsız değişkenlerden en az birinin modelde bulunması gerektiğine işaret eder.

F oranını hesaplamak için MSR ve MSE değerleri bilinmelidir.

$$F = \frac{MSR}{MSE} \tag{4.14}$$

Buradaki $MSR = \frac{SSR}{k}$ (4.15)

ve $MSE = \frac{SSE}{n-k-1}$ (4.16)

4.1.11 Çoklu Regresyon Modeli Değişkenlerinin Anlamlılık Testi

t-testi her bir değişkenin eğimlerinin anlamlılıklarını test etmede kullanılır. Y ve X_i değişkeni arasında ilişki olup olmadığını gösterir.

Hipotez:

H_0: $\beta_1 = \beta_2 = \cdots = \beta_k = 0$: (lineer ilişki mevcut değil)

H_1 en az bir $\beta_i \neq 0$ (en azından bir değişken Y ve X_i değişkeni arasında lineer ilişki mevcuttur)

t istatistiği:

$$\frac{b_i - 0}{s_{bi}} \tag{4.17}$$

formülüyle hesaplanır. Burada $Serbestlik\ Derecesi = df = n - k - 1$ ' dir ve

$SE = S_{bi} = \sqrt{\left[\sum \frac{(y_i - \hat{y}_i)^2}{df}\right]} / \sqrt{[\sum (x_i - x)^2]}$ 'dir. (4.18)

4.1.12 Kısmi Belirlilik Katsayısı

Diğer bağımsız değişkenler sabit tutulurken X_j değişkeni tarafından açıklanan değişim oranını ifade eder.

4.1.13 Kukla değişken kullanımı

Regresyon modelinde bağımsız değişkenlerin 2 veya daha fazla seviyede kategorize edilerek gösterimine "Kukla Değişken Kullanımı" denir. Örneğin erkek/kadın, genç/yaşlı, evet/hayır. Bu değişkenler (0-1) olarak kodlanmak suretiyle model içerisinde kullanılabilir. Seviye sayısının 2'den fazla olduğu durumda kukla değişken sayısı seviye sayısı-1 olmalıdır

4.1.14 Çoklu Doğrusal Bağlantı (MultiCollinearity)

Bağımsız değişkenler arasında yüksek korelasyon olma durumuna "Çoklu Doğrusal Bağlantı" adı verilir. Bu da demektir ki, aralarında yüksek korelasyon bulunan bağımsız değişkenler modele benzer bilgiyle katkı sağlamaktadır. Bu da istikrarsız katsayılar bulunmasına, regresyon katsayılarının beklenen işaretleri vermemesine sebep olur.

Yüksek Çoklu Doğrusal Bağlantı Göstergeleri:

- Modele yeni bir değişken eklenmesi durumunda katsayıların büyük oranda değişmesi
- Katsayıların işaretlerinin yanlış olması
- Modele yeni değişken eklendikçe model hatasının artış göstermesi
- Modelde daha önce anlamlı olan değişkenin yeni bir değişken katılması durumunda anlamsız olması

VIF_J (Variance Inflation Factor) çoklu doğrusal bağlantı ölçmede kullanılır:

$$VIF_J = \frac{1}{(1-R_J)^2} \quad (4.19)$$

Burada R_j, X_j ve diğer bütün x'lerle beraber çoklu belirlilik katsayısıdır.

Eğer $VIF_J > 10$, ise X_j ile diğer bağımsız değişkenler arasında yüksek doğrusal korelasyon vardır.

4.1.15 Regresyon Modelinin Oluşturulması

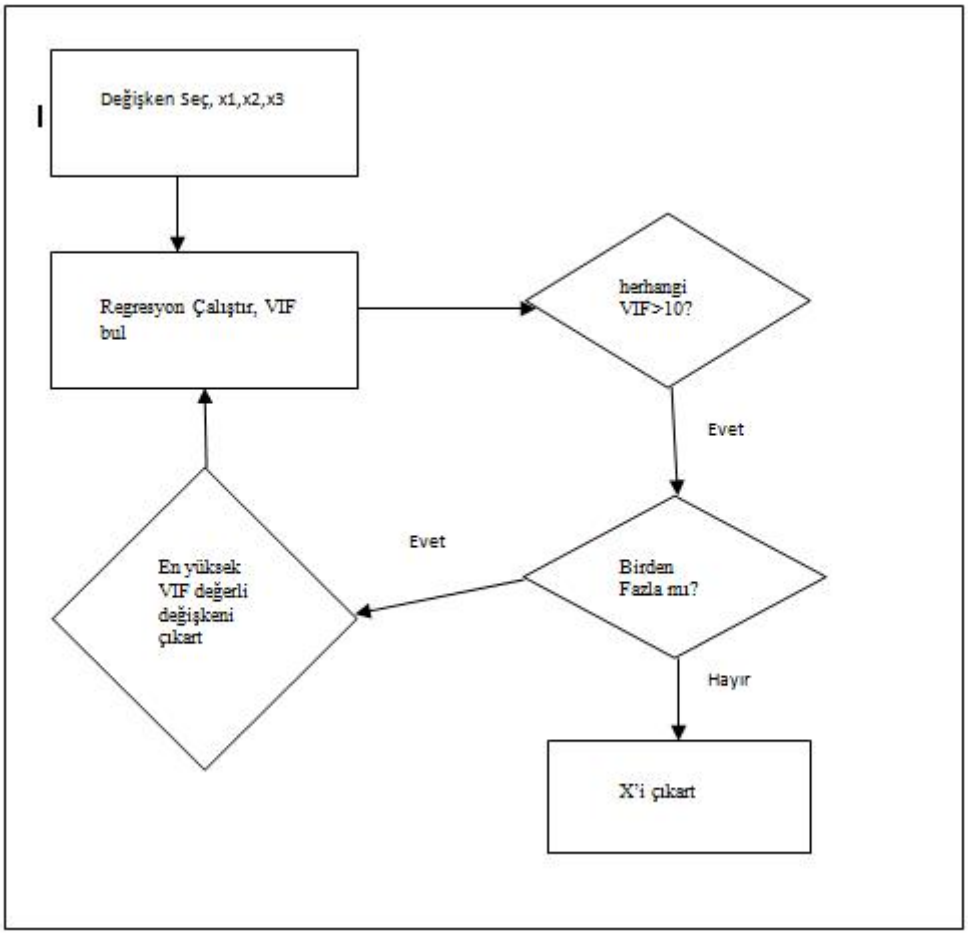
Regresyon modeli oluştururken amaç, modeli en iyi ifade edecek bağımsız değişken kümesini kullanıyor olmaktır. Bu amaçla en iyi grubu oluşturabilmek için gereksiz olan bağımsız değişkenlerin modelden çıkarılması gerekir. Çıkarma işlemi için ileriye doğru seçim veya geriye doğru eleme metotları kullanılabilir.

4.1.16 Geriye Doğru Eleme Metodu

1. Model içerisindeki tüm değişkenler için VIF değeri hesaplanır
2. VIF>10 olan değişkenler belirlenir
3. Eğer VIF>10 ise değişken modelden çıkarılır. Eğer birden fazla değişken VIF>10 ise VIF değeri en büyük olan değişkeni modelden çıkar. 1.adıma geri dönülür. Eğer hiçbir değişken VIF>10 değil ise işlem durdurulur.

İşlemi çalıştırmaya en iyi alt grubu bulmak amacıyla devam edilebilir ancak bu tez çalışmasında geriye doğru eleme metodu yukarıdaki üç adımı kullanmak suretiyle gerçekleştirilmiştir.

Aşağıdaki şekilde geriye doğru eleme metodu şekil olarak gösterilmiştir.



Şekil 4.1 Geriye Doğru Eleme Yöntemi

4.1.17 İleriye Doğru Seçim Metodu

Geriye doğru seçim metodunun tam tersi olarak düşünülebilir. Boş bir modelle başlanır ve durma noktasına kadar değişkenler modele birer birer eklenir. Algoritmasının adımları aşağıda verilmiştir:

1. Model içerisindeki tüm değişkenler için VIF değeri hesaplanır.
2. En küçük VIF değerine sahip olan değişken modele dahil edilir.
3. Geriye kalan değişkenler için yeniden VIF değeri hesaplanır.
4. Eğer tüm değişkenlerin VIF<10 ise işlem durdurulur; aksi durumda 2.adıma dönülür. Eğer hiçbir değişken VIF>10 değil ise işlem durdurulur.

4.1.18 Örnek Uygulama

Aşağıda örnek için kullanılan 39 satırdan ve 13 parametreden oluşan Gemi Parametreler'i tablosunun ilk 10 satırı Çizelge 4.1'de verilmiştir. Örnek model, 39 satırlık tablo kullanılarak oluşturulmuştur ve parametre kestirimleri buna göre gerçekleştirilmiştir. 39 satırın tamamı EK-A'da yer almaktadır.

Çizelge 4.1 Gemi Parametreleri

ID	LBP	BWL	D	T	LCB	LCF	BMT	BML	CWP	CP	CM	CB	CVP
1	21,375	6,74	4	2,61	9,8987	8,3831	2,4823	24,6739	0,8569	0,6257	0,6708	0,4197	0,4898
2	21,375	6,74	4	2,742	9,7545	8,3947	2,3538	23,247	0,8741	0,6417	0,6874	0,4411	0,5046
3	21,375	6,74	4	2,368	10,2311	8,5496	2,6462	26,8821	0,8067	0,5922	0,6369	0,3772	0,4676
4	25,74	7	3,75	2,646	12,5654	10,4393	2,0249	31,3183	0,8408	0,5861	0,8237	0,4828	0,5742
5	25,74	7	3,75	2,755	12,4155	10,2824	1,9827	31,0618	0,864	0,5988	0,8307	0,4974	0,5757
6	25,74	7	3,75	2,528	12,7274	10,9708	2,063	28,9697	0,7958	0,5725	0,8155	0,4669	0,5867
7	25	7,2	3,625	2,28	12,3395	10,893	2,3498	31,7088	0,8322	0,6317	0,804	0,5079	0,6103
8	25	7,2	3,625	2,41	12,2036	10,6626	2,2716	30,2129	0,8525	0,6457	0,8146	0,526	0,617
9	25	7,2	3,625	2,119	12,4753	11,693	2,4164	28,8742	0,7703	0,6152	0,7892	0,4855	0,6303
10	26,35	7,5	3,55	2,574	12,8849	11,2478	2,237	28,8353	0,8133	0,6238	0,8102	0,5054	0,6214

Model planında bağımlı değişken LBP, bağımsız değişkenler ise diğerleri olarak belirlenirse, bu durumda parametre kestirimin ardından

$$LBP = b_0 + BWL * b_1 + D * b_2 + T * b_3 + LCB * b_4 + LCF * b_5 + BMT * b_6 + BML * b_7 + CWP * b_8 + CP * b_9 + CM * b_{10} + CB * b_{11} + CVP * b_{12} + e$$

Modelini oluşturabilmek için $b_0, b_1, \ldots b_{12}$ parametrelerinin kestirimini yapmak yani b katsayılar matrisini hesaplamak gerekir. Sonuçlar için şekilde gösterilen program geliştirilmiş ve kullanılmıştır. Gemi Parametreleri modeli için hesaplanan katsayılar tabloda gösterilmiştir.

Çizelge 4.2 Model 1 için Hesaplanan Regresyon Katsayıları

b_0	12,14627
b_1	-0,31246
b_2	-0,47281
b_3	5,770053
b_4	-0,45897
b_5	0,960367
b_6	0,766683
b_7	0,409343
b_8	42,87784
b_9	-94,9722
b_{10}	-71,6552
b_{11}	16,63614
b_{12}	94,5443

Gemi Parametreleri tablosu için hesaplanan çoklu belirlilik katsayısı, çoklu korelasyon katsayısı ve düzeltilmiş belirlilik katsayısı ise aşağıda gösterildiği gibidir.

Çizelge 4.3 Model 1 için Hesaplanan Rsq, R, Adjusted Rsq

R^2	0,997
R	0,998
$Adjusted\ R^2$	0,995

Yine modelin anlamlılığını yani F oranını hesaplamak için gerekli olan MSR ve MSE katsayıları ve MSR/MSE sonucu elde edilen F oranı aşağıdaki gibidir.

Çizelge 4.4 Model 1 için Hesaplanan F oranı

MSR	32,29
MSE	0,04109
F Oranı	785,836

Model için hesaplanan standartlaştırılmış katsayılar ise aşağıdaki gibidir. Standartlaştırılmış katsayıların önündeki bağımsız değişkeni açıklamakta göreli etkisi olduğundan bahsedilmişti. Ancak model içinde yüksek Çoklu Doğrusal Bağlantı 'ya sahip değişkenler olduğundan modele değişken eklenip çıkarılması durumunda katsayıların ve katsayı işaretlerinin değişkenlik göstermesinden dolayı buradaki katsayıların yanıltıcı olduğunu söylemek çok da yanlış olmaz.

Çizelge 4.5 Model 1 için Hesaplanan Standartlaştırılmış Katsayılar

b_1	-0,10303
b_2	-0,08239
b_3	0,440641
b_4	-0,20651
b_5	0,415171
b_6	0,108083
b_7	0,617959
b_8	0,638844
b_9	-1,2368
b_{10}	-1,76955
b_{11}	0,327651
b_{12}	1,882797

Sonuçları bulmak için kullandığımız bizim tarafımızdan geliştirilen analiz aracının söz konusu model için çalıştırılmış ekran görüntüsü yukarıdaki şekilden görülebilir. Model için çoklu doğrusal bağlantılı değişkenleri eleme işlemini bölüm 1.14.1'deki adımları izlemek suretiyle gerçekleştirirsek modelden BWL, LCB, LCF, CB, CM bağımsız değişkenleri modelden yüksek çoklu doğrusal bağlantıya sahip oldukları için elenirler. D, T, BMT, BML, CWP, CP, CVP katsayıları ise modelde kalır. Yeni model

$$LBP = b_0 + D * b_1 + T * b_2 + BMT * b_3 + BML * b_4 + CWP * b_5 + CP * b_6 + CVP * b_7 + e$$

şeklinde olur ve yeni model için hesaplanan katsayılar Çizelge 4.6'de verilmiştir.

Çizelge 4.6 Model 2 için Hesaplanan Katsayılar

β_0	6,456679
β_1	0,407207
β_2	3,735485
β_3	-0,38465
β_4	0,413615
β_5	-15,2526
β_6	-7,06068
β_7	22,63472

Yine söz konusu modele ait hesaplanan çoklu belirlilik katsayısı, çoklu korelasyon katsayısı ve düzeltilmiş belirlilik katsayısı ise aşağıda gösterildiği gibidir.

Çizelge 4.7 Model 2 için Hesaplanan Rsq, R, Adjusted Rsq

R^2	0,9833
R	0,9916
$Adjusted\ R^2$	0,979

Yine modelin anlamlılığını yani F oranını hesaplamak için gerekli olan MSR ve MSE katsayıları ve MSR/MSE sonucu elde edilen F oranı aşağıdaki gibidir.

Çizelge 4.8 Model 2 için Hesaplanan F oranı

MSR	54,59
$\frac{MSR}{MSE}$	0,208
F Oranı	262,32

Yukarıdaki sonuçlardan da anlaşıldığı üzere modelden beş değişken çıkarıldığı halde çoklu belirlilik katsayısı yüksek ölçüde değişim göstermemiş olduğunu 1'e yakınlığını koruduğunu ve halen modeli açıklamakta 0,9833 oranında başarılı olduğunu gözlemlenebilir. Yeni model için standartlaştırılmış katsayılar Çizelge 4.9'da verilmiştir.

Çizelge 4.9 Model 2 için Hesaplanan Standartlaştırılmış Katsayılar

b_1	0,070958
b_2	0,285267
b_3	-0,05423
b_4	0,624409
b_5	-0,22725
b_6	-0,09195
b_7	0,450758

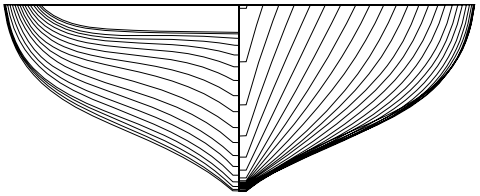
Modelden rastgele seçilen bir değişkeni (buradaki örnekte “D” değişkenini çıkarılmıştır) çıkarıp standartlaştırılmış katsayılar tekrar hesaplandığında katsayılarda önemli ölçüde değişim olmadığı ve katsayıların işaretlerinin aynı kaldığı görülebilir. Bu da gösteriyor ki Çoklu Doğrusal Bağlantı içeren değişkenlerin modelden çıkarılmasının ardından model katsayılarının yanıltıcı etkisi ortadan kalkmaktadır.

Çizelge 4.10 Model 3 için Hesaplanan Standartlaştırılmış Katsayılar

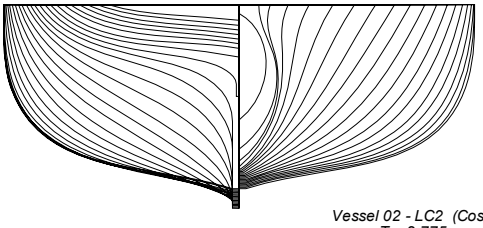
b_1	0,323445
b_2	-0,00392
b_3	0,64017
b_4	-0,24256
b_5	-0,0917
b_6	0,453146

4.2 Başlangıç Dizayn Aşamasında Balıkçı Gemilerinin Gemi Hareketlerinin Tekne Form Parametreleri Kullanarak Çoklu Regresyon Analizine Dayalı Modellenmesi

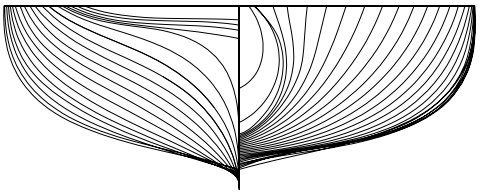
Bu bölüm kapsamında “Başlangıç Dizayn Aşamasında Balıkçı Gemilerinin Gemi Hareketlerinin Tekne Form Parametreleri Kullanarak Çoklu Regresyon Analizine Dayalı Modellenmesi” yapılmıştır. Modelleme aşamasında kullanılan veritabanı balıkçı gemilerine aittir. Şekil 4.2’ de on üç farklı balıkçı gemisinin geometrisi verilmektedir. Bu gemilerin ana ve yardımcı parametrelerinin tanımları EK-B ’da, parametrelerin değerleri ise Çizelge 4.11’de aşağıda verilmektedir. İlgili veritabanında göz önüne alınan bu gemilerin üç ayrı yükteki üç ayrı hareketi incelenmiştir.



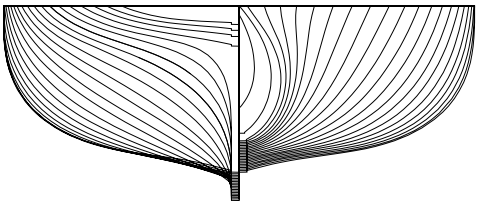
Vessel 01 - LC2 (Dinko)
T = 2.775 m



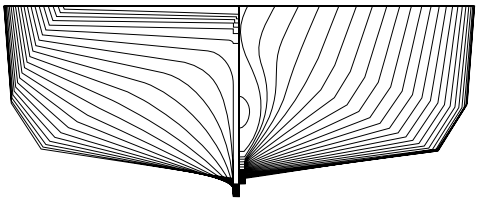
Vessel 02 - LC2 (Cost08)
T = 2.775 m



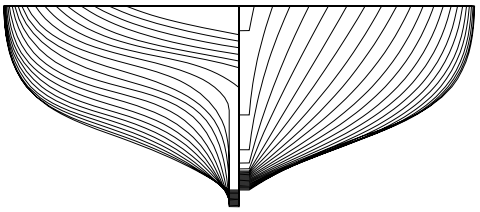
Vessel 03 - LC2 (Flori)
T = 2.410 m



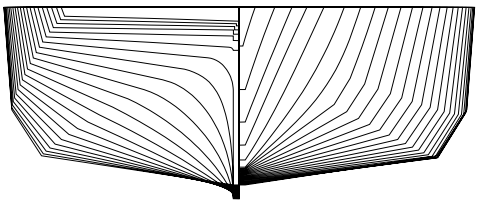
Vessel 04 - LC2 (Gemma)
T = 2.647 m



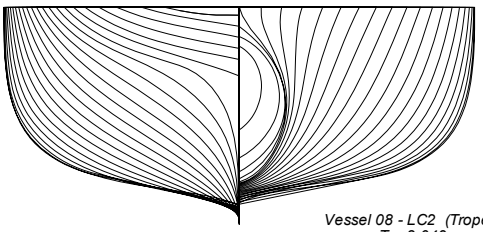
Vessel 05 - LC2 (Genova)
T = 2.810 m



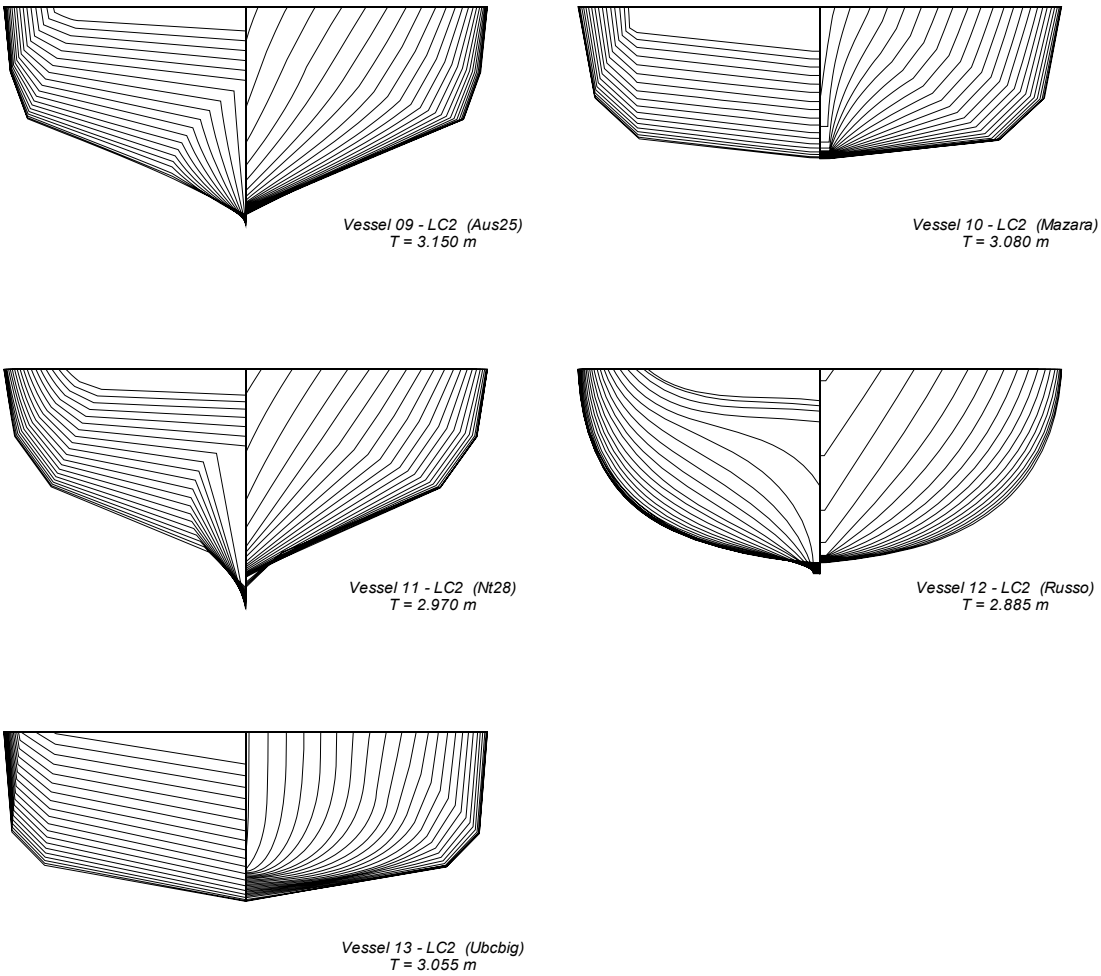
Vessel 06 - LC2 (Greben)
T = 2.687 m



Vessel 07 - LC2 (Ligny)
T = 2.906 m



Vessel 08 - LC2 (Tropesca)
T = 3.049 m



Şekil 4.2 Gemi Geometrileri

Çizelge 4.11 Gemi Parametreleri

Vessel	L (m)	L/B (-)	B/T (-)	$L/\nabla^{1/3}$ (-)	C_{WP} (-)	C_{VP} (-)	C_{WPA} (-)	C_{WPF} (-)	C_{VPA} (-)	C_{VPF} (-)	LCF/L (-)	LCB/L (-)
V_011	21.38	3.171	2.582	3.955	0.857	0.490	0.965	0.621	0.513	0.583	0.392	0.463
V_012	21.38	3.171	2.458	3.827	0.874	0.505	0.979	0.635	0.535	0.589	0.393	0.456
V_013	21.38	3.171	2.846	4.234	0.807	0.468	0.909	0.591	0.479	0.570	0.400	0.479
V_021	25.74	3.677	2.646	4.200	0.841	0.574	0.846	0.663	0.608	0.670	0.406	0.488
V_022	25.74	3.677	2.541	4.103	0.864	0.576	0.871	0.673	0.613	0.701	0.399	0.482
V_023	25.74	3.677	2.769	4.312	0.796	0.587	0.804	0.654	0.611	0.696	0.426	0.494
V_031	25.00	3.472	3.158	4.216	0.832	0.610	0.834	0.698	0.635	0.689	0.436	0.494
V_032	25.00	3.472	2.988	4.091	0.853	0.617	0.868	0.703	0.638	0.701	0.427	0.488

V_033	25.00	3.472	3.398	4.386	0.770	0.630	0.744	0.690	0.675	0.674	0.468	0.499
V_041	26.35	3.513	2.914	4.144	0.813	0.621	0.852	0.666	0.626	0.715	0.427	0.489
V_042	26.35	3.513	2.833	4.083	0.823	0.625	0.868	0.668	0.628	0.720	0.423	0.486
V_043	26.35	3.513	3.158	4.327	0.771	0.624	0.789	0.659	0.635	0.698	0.447	0.497
V_051	25.00	3.125	2.835	3.692	0.875	0.628	0.882	0.668	0.710	0.757	0.397	0.472
V_052	25.00	3.125	2.752	3.634	0.890	0.629	0.890	0.674	0.719	0.763	0.393	0.468
V_053	25.00	3.125	3.000	3.802	0.819	0.651	0.800	0.658	0.753	0.760	0.423	0.477
V_061	20.50	2.941	2.766	3.852	0.804	0.521	0.845	0.559	0.511	0.614	0.435	0.492
V_062	20.50	2.941	2.585	3.691	0.834	0.533	0.879	0.574	0.530	0.622	0.429	0.484
V_063	20.50	2.941	3.066	4.117	0.742	0.512	0.769	0.536	0.495	0.597	0.451	0.503
V_071	25.00	3.125	2.835	3.631	0.898	0.644	0.891	0.690	0.719	0.746	0.398	0.467
V_072	25.00	3.125	2.753	3.576	0.903	0.651	0.894	0.696	0.732	0.747	0.398	0.463
V_073	25.00	3.125	3.001	3.739	0.860	0.652	0.852	0.679	0.723	0.743	0.413	0.473
V_081	27.25	3.733	2.547	4.118	0.782	0.650	0.799	0.669	0.661	0.747	0.440	0.501
V_082	27.25	3.733	2.394	3.989	0.818	0.642	0.836	0.679	0.657	0.750	0.426	0.495
V_083	27.25	3.733	2.700	4.243	0.753	0.654	0.746	0.661	0.674	0.740	0.452	0.507
V_091	21.00	2.770	2.627	3.514	0.861	0.540	0.844	0.672	0.515	0.652	0.424	0.481
V_092	21.00	2.770	2.406	3.332	0.887	0.563	0.881	0.697	0.556	0.663	0.424	0.472
V_093	21.00	2.770	2.756	3.619	0.831	0.537	0.816	0.660	0.572	0.643	0.433	0.486
V_101	30.80	2.962	3.870	4.061	0.766	0.662	0.873	0.544	0.718	0.695	0.388	0.424
V_102	30.80	2.962	3.402	3.811	0.783	0.688	0.884	0.567	0.759	0.688	0.392	0.418
V_103	30.80	2.962	4.132	4.199	0.756	0.648	0.862	0.533	0.700	0.689	0.386	0.428
V_111	20.00	3.061	2.633	3.967	0.799	0.495	0.734	0.666	0.495	0.509	0.428	0.486
V_112	20.00	3.061	2.455	3.787	0.824	0.514	0.754	0.691	0.527	0.520	0.430	0.478
V_113	20.00	3.061	2.901	4.239	0.738	0.484	0.667	0.636	0.474	0.487	0.447	0.497
V_121	27.30	4.015	2.729	4.275	0.884	0.636	0.831	0.784	0.663	0.651	0.447	0.485
V_122	27.30	4.015	2.484	4.072	0.915	0.648	0.866	0.806	0.679	0.663	0.444	0.480
V_123	27.30	4.015	2.941	4.447	0.841	0.642	0.785	0.765	0.665	0.642	0.461	0.490
V_131	28.00	3.060	3.386	3.825	0.854	0.663	0.903	0.699	0.658	0.770	0.428	0.488
V_132	28.00	3.060	2.995	3.599	0.885	0.680	0.939	0.707	0.685	0.788	0.416	0.477

V_133	28.00	3.060	3.704	4.003	0.823	0.657	0.859	0.694	0.650	0.755	0.442	0.496
min	30.80	4.015	4.132	4.447	0.915	0.688	0.979	0.806	0.759	0.788	0.468	0.507
max	20.00	2.770	2.394	3.332	0.738	0.468	0.667	0.533	0.474	0.487	0.386	0.418

Yukarıda tanımı yapılan gemilerin dokuz farklı dalga boyu ve yedi faklı hızdaki üç hareketini içeren bir veritabanı için çoklu regresyon analizi gerçekleyen bir yazılım sistemi geliştirilmiştir. Bu sistemin arayüzleri ve detayları (Sayli vd. [29]) yayınlarından izlenebilir. Bu yayınlarda da belirtildiği üzere yapılan çeşitli analizlerden sonra dört seviyedeki modellere Çizelge 4.12 de gösterilen parametrelere dayalı oluşturulmuştur.

Çizelge 4.12 Gemi Parametreleri için Modeller

Model	Adı	Gemi Ana Parametreleri	Diğer Parametreleri	Hız
I	Basit	$L/\nabla^{1/3}$, L/B , B/T		Fn , Fn^2
II	Orta	$L/\nabla^{1/3}$, L/B , B/T	C_{WP}, C_{VP}	Fn , Fn^2
III	İleri 1	$L/\nabla^{1/3}$, L/B , B/T	C_{WP}, C_{VP}, LCF/L , LCB/L	Fn , Fn^2
IV	İleri 2	$L/\nabla^{1/3}$, L/B , B/T	C_{WPA}, C_{WPF}, C_{VPA}, C_{VPF}	Fn , Fn^2

Geliştirilen çoklu regresyon analizi modelleri (İnme kalkma hareketi, yalpalama hareketi ve dikey hareket için) aşağıda gösterilmiştir:

I) Basit Model

Aşağıda sırasıyla İnme kalkma hareketi, yalpalama hareketi ve dikey hareket için hesaplanan denklemler verilmektedir.

$$\frac{z}{a} = A_o + A_1\frac{L}{\nabla^{1/3}} + A_2\frac{L}{B} + A_3\frac{B}{T} + A_4 Fn + A_5 Fn^2$$

$$\frac{\theta}{\alpha} = B_o + B_1\frac{L}{\nabla^{1/3}} + B_2\frac{L}{B} + B_3\frac{B}{T} + B_4 Fn + B_5 Fn^2 \qquad (1)$$

$$\frac{a_v L}{ga} = C_o + C_1\frac{L}{\nabla^{1/3}} + C_2\frac{L}{B} + C_3\frac{B}{T} + C_4 Fn + C_5 Fn^2$$

II) Orta Model

Aşağıda sırasıyla İnme kalkma hareketi, yalpalama hareketi ve dikey hareket için hesaplanan denklemler verilmektedir.

$$\frac{z}{a} = A_o + A_1 \frac{L}{\nabla^{1/3}} + A_2 \frac{L}{B} + A_3 \frac{B}{T} + A_4\ C_{WP} + A_5 C_{PV} + A_6 Fn + A_7 Fn^2$$

$$\frac{\theta}{\alpha} = B_o + B_1 \frac{L}{\nabla^{1/3}} + B_2 \frac{L}{B} + B_3 \frac{B}{T} + B_4 C_{WP} + B_5 C_{PV} + B_6 Fn + B_7 Fn^2 \qquad (2)$$

$$\frac{a_v L}{ga} = C_o + C_1 \frac{L}{B} + C_2 \frac{B}{T} + C_3 \frac{L}{T} + C_4 C_B + C_5 Fn + C_6 Fn^2$$

III) İleri 1 Model

Aşağıda sırasıyla İnme kalkma hareketi, yalpalama hareketi ve dikey hareket için hesaplanan denklemler verilmektedir.

$$\frac{z}{a} = A_o + A_1 \frac{L}{\nabla^{1/3}} + A_2 \frac{L}{B} + A_3 \frac{B}{T} + A_4\ C_{WP} + A_5 C_{PV} + A_6 LCF/L + A_7 LCB/L + A_8 Fn + A_9 Fn^2$$

$$\frac{\theta}{a} = B_o + B_1 \frac{L}{\nabla^{1/3}} + B_2 \frac{L}{B} + B_3 \frac{B}{T} + B_4\ C_{WP} + B_5 C_{PV} + B_6 LCF/L + B_7 LCB/L + B_8 Fn + B_9 Fn^2 \qquad (3)$$

$$\frac{a_v L}{ga} = C_o + C_1 \frac{L}{\nabla^{1/3}} + C_2 \frac{L}{B} + C_3 \frac{B}{T} + C_4\ C_{WP} + C_5 C_{PV} + C_6 LCF/L + C_7 LCB/L + C_8 Fn + C_9 Fn^2$$

IV) İleri 2 Model

Aşağıda sırasıyla İnme kalkma hareketi, yalpalama hareketi ve dikey hareket için hesaplanan denklemler verilmektedir.

$$\frac{z}{a} = A_o + A_1 \frac{L}{\nabla^{1/3}} + A_2 \frac{L}{B} + A_3 \frac{B}{T} + A_4\ C_{WPA} + A_5 C_{WPF} + A_6 C_{VPA} + A_7 C_{VPF} + A_8 Fn + A_9 Fn^2$$

$$\frac{\theta}{a} = B_o + B_1 \frac{L}{\nabla^{1/3}} + B_2 \frac{L}{B} + B_3 \frac{B}{T} + B_4\ C_{WPA} + B_5 C_{WPF} + B_6 C_{VPA} + B_7 C_{VPF} + B_8 Fn + B_9 Fn^2 \qquad (4)$$

$$\frac{A_v L}{ga} = C_o + C_1 \frac{L}{\nabla^{1/3}} + C_2 \frac{L}{B} + C_3 \frac{B}{T} + C_4\ C_{WPA} + C_5 C_{WPF} + C_6 C_{VPA} + C_7 C_{VPF} + C_8 Fn + C_9 Fn^2$$

Basit model için hesaplanan regresyon katsayıları ve çoklu belirginlik katsayıları tabloda verilmiştir. Diğer modeller için hesaplanan katsayılar EK B'de bulunmaktadır.

Çizelge 4.13 İnme Kalkma Hareketi için Hesaplanan Katsayılar

$$\frac{z}{a} = A_o + A_1 \frac{L}{\nabla^{1/3}} + A_2 \frac{L}{B} + A_3 \frac{B}{T} + A_4 Fn + A_5 Fn^2$$

λ/L	A_0	A_1	A_2	A_3	A_4	A_5	R-sq
0.50	-0.0487	0.0555	-0.0123	-0.0156	-0.5502	1.0330	0.6838
0.75	0.1354	-0.0618	0.0583	0.0441	-1.4327	3.0426	0.8495
1.00	-0.5461	0.3678	-0.1107	-0.0962	-0.0872	-2.3763	0.7565
1.25	-1.1381	0.5969	-0.0999	-0.1722	3.6513	-12.9446	0.8227
1.50	-0.8469	0.4967	-0.0443	-0.1605	4.4911	-9.7896	0.7724
1.75	0.2402	0.1930	0.0053	-0.1299	2.2373	2.0216	0.8949
2.00	0.9499	0.0168	0.0151	-0.1103	0.7934	6.2817	0.9205
2.50	1.0372	0.0112	-0.0049	-0.0769	0.4371	3.3637	0.9022
3.00	0.9891	0.0159	-0.0052	-0.0496	0.4236	1.2317	0.8896

Çizelge 4.14 Yalpalama Hareketi için Hesaplanan Katsayılar

$$\frac{\theta}{\alpha} = B_o + B_1 \frac{L}{\nabla^{1/3}} + B_2 \frac{L}{B} + B_3 \frac{B}{T} + B_4 Fn + B_5 Fn^2$$

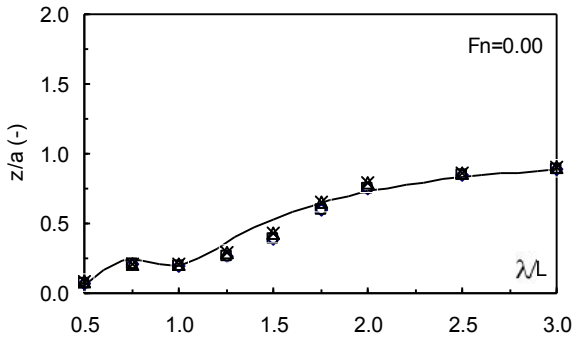
λ/L	B_0	B_1	B_2	B_3	B_4	B_5	R-sq
0.50	0.0228	-0.0051	0.0097	0.0049	-0.2979	0.5015	0.8752
0.75	-0.0614	0.0954	-0.0407	-0.0204	-0.7397	1.3033	0.8052
1.00	-0.1758	0.1944	-0.0302	-0.0171	-1.0827	-0.3594	0.9368
1.25	-0.2990	0.2487	-0.0301	0.0159	0.9213	-6.3529	0.8553
1.50	-0.0236	0.2092	-0.0619	0.0451	1.9392	-5.4877	0.5522
1.75	0.6932	0.0580	-0.0757	0.0498	1.1895	0.7746	0.8581
2.00	1.1157	-0.0177	-0.0784	0.0299	0.7191	3.1494	0.9029
2.50	1.1391	0.0044	-0.0784	0.0095	0.6535	2.3557	0.9203
3.00	1.1094	0.0155	-0.0721	0.0069	0.5298	2.0278	0.9336

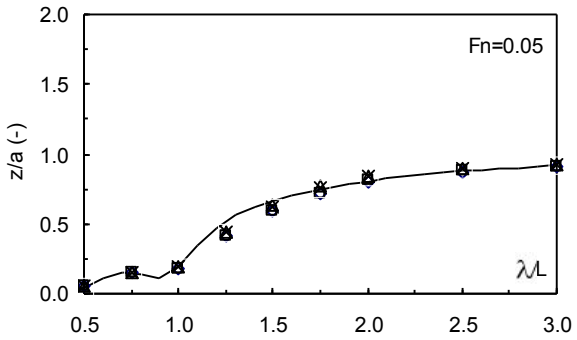
Çizelge 4.15 Dikey Hareket için Hesaplanan Katsayılar

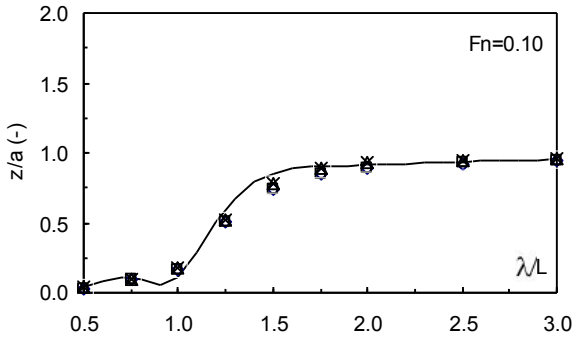
$$\frac{a_v L}{ga} = C_o + C_1 \frac{L}{\nabla^{1/3}} + C_2 \frac{L}{B} + C_3 \frac{B}{T} + C_4 Fn + C_5 Fn^2$$

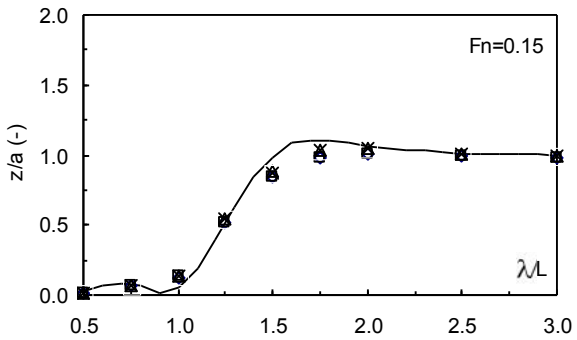
λ/L	C_0	C_1	C_2	C_3	C_4	C_5	R-sq
0.50	-1.6371	-0.1706	1.6461	0.3207	-0.4979	-26.5440	0.5258
0.75	-1.3229	4.1800	-1.6659	-1.9544	-6.8836	6.1558	0.5030
1.00	-14.5208	6.8680	-0.2903	-1.6383	31.6532	-119.7730	0.7518
1.25	-17.0380	5.5094	0.8831	-0.4738	61.2818	-131.1120	0.7290
1.50	-7.4445	2.5627	0.7124	0.1890	38.5132	22.3118	0.8829
1.75	5.1534	-0.6110	0.4263	0.3692	10.1442	124.8288	0.9667
2.00	8.0496	-1.0829	0.0775	0.1729	5.4764	103.1251	0.9650
2.50	4.4491	-0.1610	-0.1656	0.0091	8.3733	38.5484	0.9755
3.00	2.9537	0.0055	-0.1248	0.0052	7.2054	19.8967	0.9834

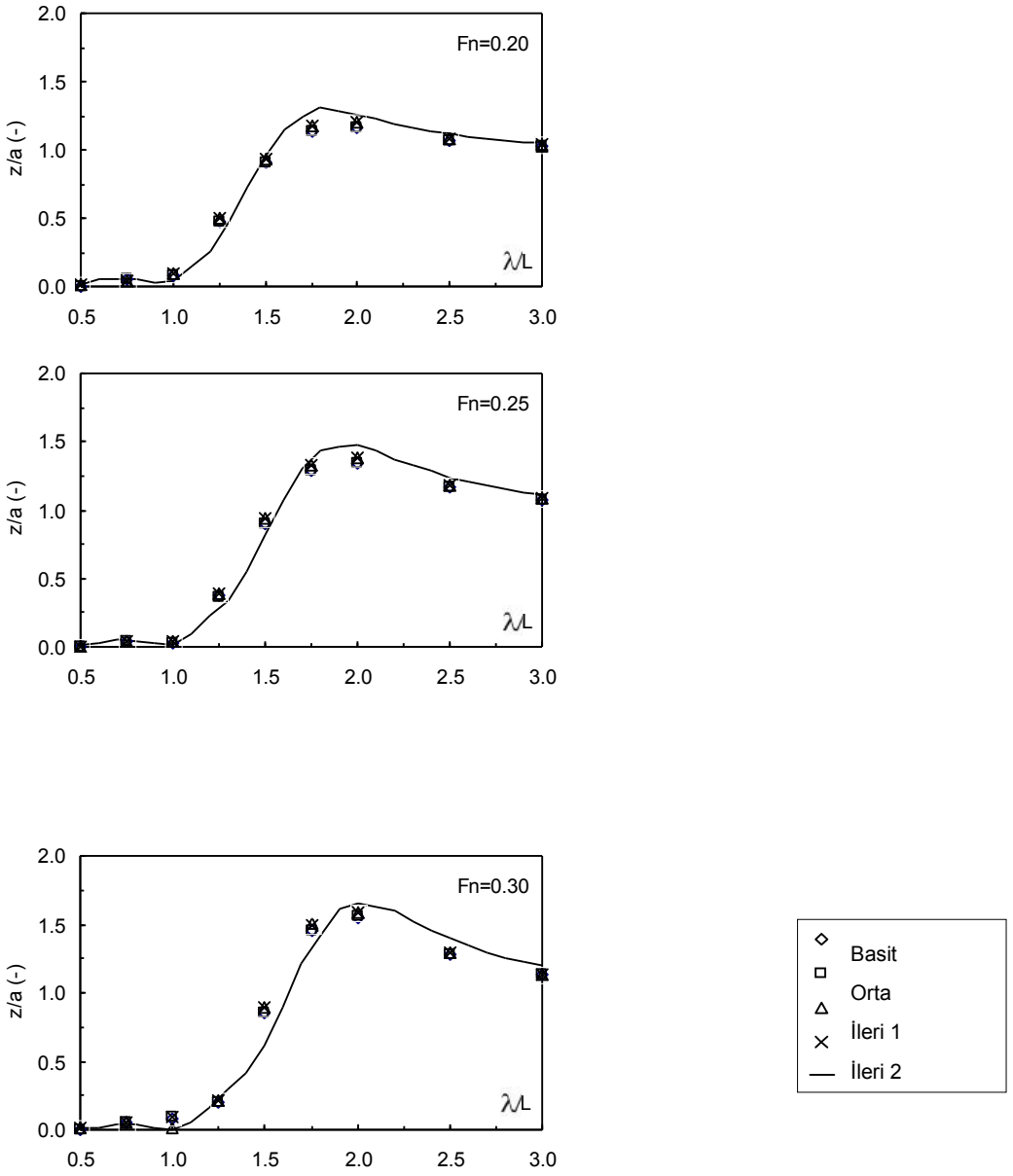
Vessel V_051 gemisinin inme kalkma hareketi ele alınarak her hız için veritabanında bulunan gerçek değerler ile modellerden elde edilen tahmini değerler karşılaştırılmıştır ve sonuçlar Şekil 4.3 grafik olarak verilmiştir.


Fn=0.00
z/a (-)
λ/L
2.0
1.5
1.0
0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Fn=0.05
z/a (-)
λ/L
2.0
1.5
1.0
0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Fn=0.10
z/a (-)
λ/L
2.0
1.5
1.0
0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Fn=0.15
z/a (-)
λ/L
2.0
1.5
1.0
0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0

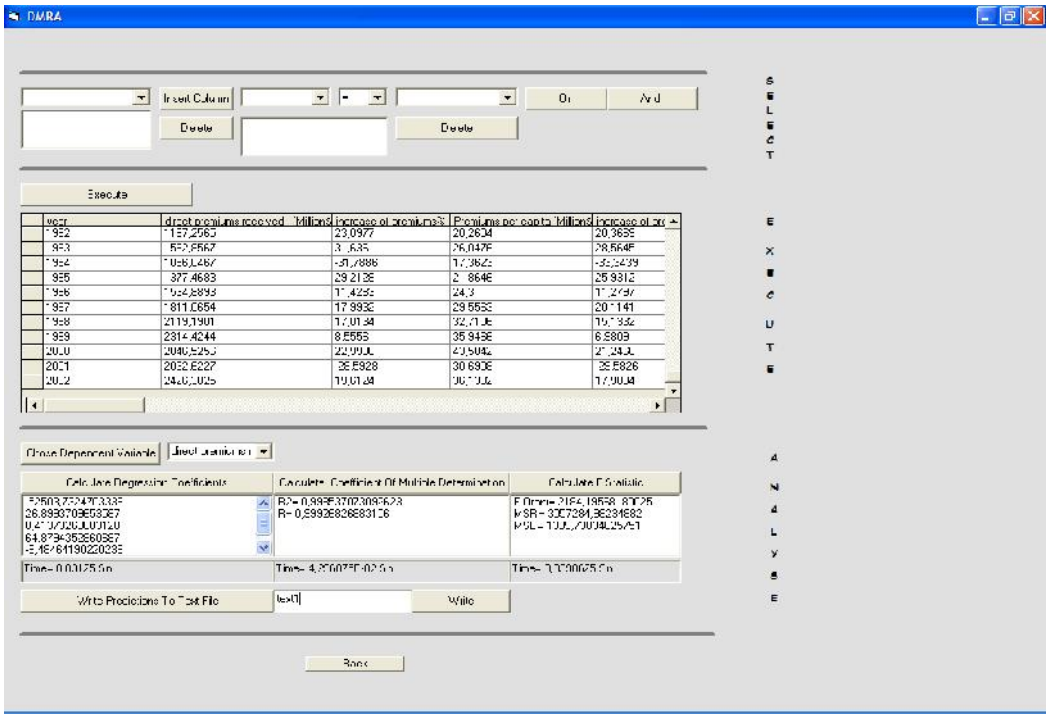


Şekil 4.3 Gemi Modelleri için Gerçekleşen ve Tahmin Edilen Değerlerin Kıyaslaması

4.3 Çoklu Regresyon Analizlerini Kullanarak Kural Öğrenen Bilgi Sistemi

Bu çalışmada, çoklu regresyon analizine dayanan bir dinamik bilgi öğrenme sistemi oluşturulmuştur. Sistem kullanıcının verdiği her sorgu için sonuç tablosundaki verileri çoklu doğrusal regresyonla analiz ederek ilişkili olanlar arasında bulunan bilgileri "kurallar" adı altında çıkarımlar yapar, daha sonra da bu kuralları gelecek olan yeni sorguların çalıştırılmasında kullanır.

Geliştirilen Bilgi Sistemi Şekil 4.4' de verilmiştir.



Şekil 4.4 DMRA Programı Arayüzü

Sistem EPISODE veritabanı (33.873 kayıtlı)üzerinde çalıştırılmıştır. Bu veritabanı İngiltere Sağlık Dairesi'nin 'Hospital Episode Statistics - 1998/2003' web sayfasından alınmıştır:

http://www.dh.gov.uk/PublicationsAndStatistics/Statistics/HospitalEpisodeStatistics/HESFreeData.

Sistem üzerinde pek çok sorgu çalıştırılmış ve sonuçlar analiz edilerek (Saylı vd. [30]) yayınlanmıştır.

BÖLÜM5

VERİ ANALİZİ KULLANILARAK KURALLARIN ÖĞRENİLMESİ

Bir veritabanı üzerinde çalıştırılan sorgu ele alınarak yapılan kural öğrenmeleri sonucunda elde edilen kuralların sisteme girilecek yeni sorgular için kullanılması anlamsal sorgu optimizasyonunun temelini oluşturmaktadır. Bu kullanım ile sistem daha akıllı bir sisteme dönüştürülmekte ve gerekmedikçe veri tabanına giriş yapılmadan öğrenilen kurallardan sorgunun sonucunun bulunma ihtimali araştırılmaktadır, eğer bu ihtimal yoksa bu durumda bile kuralın sisteme sağlayacağı bilgi dahilinde daha bilinçli bir sorgu optimizasyonunun yapılması sağlanabilir. Başka bir deyişle sistem kendi içerisinde kendinden kendine öğrenme yapabilmektedir. Kurallardan elde edilecek anahtarlara ve indeks yapılarına ait sütunlar içeren öğrenilen kuralların kullanımları ile sistemin zamandan tasarruf etmesi sağlanabilir. Ancak daha önce açıklanan öğrenme metotlarına göre yapılan öğrenmeler sonucunda elde edilen kural sayısı hızla artmaktadır. Tüm kuralların kullanılması veya bazılarının seçilerek kullanılması sonucu elde edilen ipuçlarına göre her kural aynı yararlılığa sahip olmamaktadır. Bu nedenle mümkünse bu tip kuralların hiç öğrenilmemesi ya da bu tip kuralların seçilerek kullanılması gerekmektedir. Bu bağlamda ilk husus hedeflenerek kuralların faydasının ölçülmesinde veri analiz tekniklerinin kullanılması önerilmiştir. Veri analiz teknikleri ve özellikle çoklu regresyon analizleri kullanılarak oluşturulacak bir öğrenme metodunu içeren kural öğrenme ve kullanma sistemi sorguların çalışmasını daha verimli bir hale getirebilir. Ayrıca bu sistem zaman kazancını maksimum yaparak, sorgunun çalışma zamanını minimize edebilir.

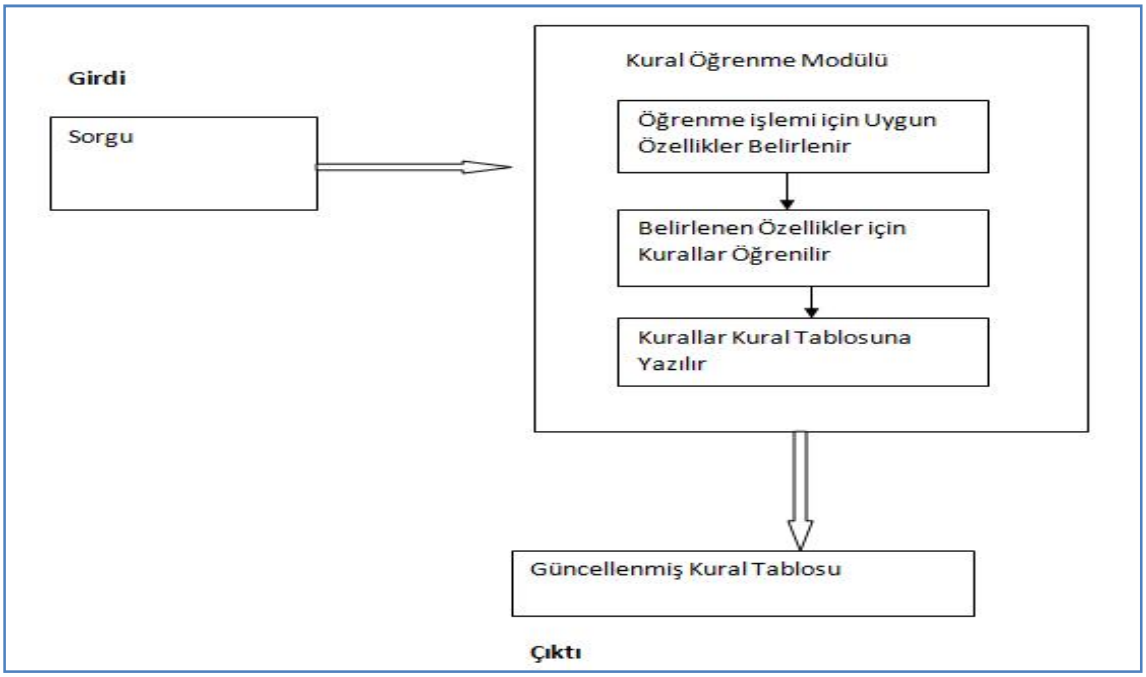
Bu doğrultuda oluşturulan kural öğrenme sistemi öğrenmeye başlamadan önce anlamlı bir şekilde sütun sayısını azaltmalıdır. İstatistiksel analizler yardımıyla kural öğrenmek

istenilen sütun için anlamı olan başka sütunları tespit etmek daha anlamsız olan sütunları ise elemek mümkündür. Sütunların azalması yada bir başka deyişle eliminasyon yapılması kural sayısının azalmasına neden olacaktır. Daha az kural üreterek çalışan öğrenen sistem daha hızlı çalışacaktır. Kural kümesi anlamsız verilerin neden olduğu gereksiz şişmelerden arınacaktır.

Çoklu regresyon analizine, geriye doğru eleme yöntemi ve standartlaştırılmış regresyon katsayılarına dayalı dinamik bir kural öğrenme metodu oluşturulmuştur. Bu metodun akış diyagramı Bölüm 5.1'de şekilsel olarak tanımlanmıştır.

5.1 Dinamik Kural Öğrenme Metodu

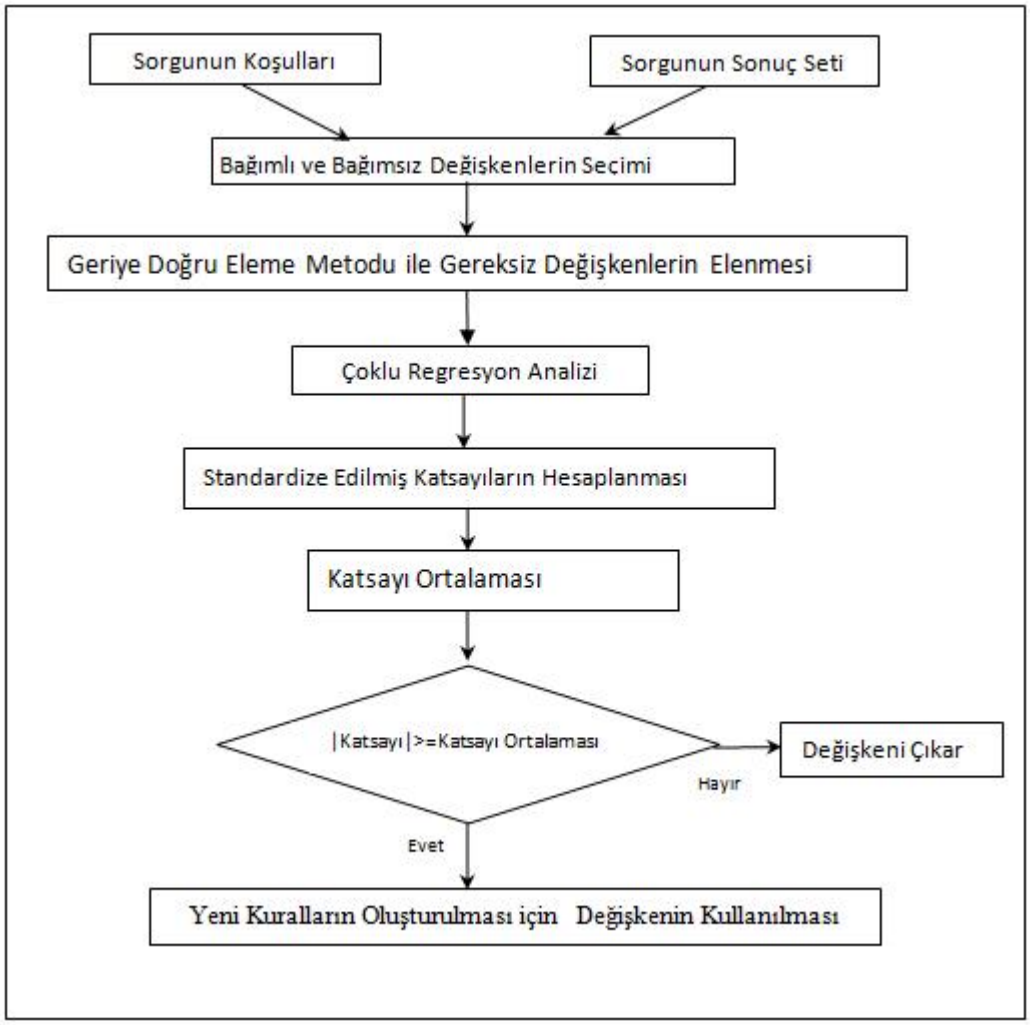
Dinamik Kural Öğrenme Modülü şekil 5.1'de gösterilmiştir. Dinamik kural öğrenme modülü sorgunun sisteme girmesi ile başlar, daha sonra uygun model kurularak öğrenme işlemi için optimum özellikler belirlenir. Bu aşamada gerçekleştirilen işlemler ve kullanılan teknikler, Bölüm 5.1.1'de detaylı olarak anlatılmıştır. Uygun özellikleri belirledikten sonraki aşama ise kural öğrenme aşamasıdır. Bu aşamanın adımları Bölüm 5.1.2'de anlatılmıştır. Kurallar öğrenildikten sonra, öğrenilen kurallar kural tablosuna eklenir. Kural tablosu ve özellikleri Bölüm 5.1.3'de anlatılmıştır. Sistemin çıktısını güncellenmiş kural tablosu oluşturmaktadır.



Şekil 5.1 Dinamik Kural Öğrenme Modülü

5.1.1 Kural Öğrenmede Kullanılacak Değişkenlerin Belirlenmesi

Kural öğrenme metodu veritabanına sorgunun girişi ile başlar, eğer sorgu koşulları ile eşleşen kural var ise sorgunun optimizasyonu yapılarak sorgunun sonuç kümesi kullanıcıya döndürülür, eşleşen kural bulunmuyorsa sorgunun koşulu yeni kuralların öğrenilmesi için SQO'nun otomatik kural öğrenme modülüne gönderilir. Bu noktada "Dinamik Kural Öğrenme Metodu" devreye girer. Dinamik kural öğrenme metodunun adımları Şekil 5.2' de şekilsel olarak ifade edilmiştir.



Şekil 5.2 Dinamik Kural Öğrenme Metodu Adımları

Metotta önemli olan nokta sorgunun koşulu için, veritabanında bulunan hangi özelliklere dayanarak kural öğrenme işleminin yapılacağıdır. Şu şekilde düşünülürse, 200 özellikten oluşan bir tablo da, sorgu koşulu için kural öğrenmek istendiğinde hangi özellikler kullanılmalı ki öğrenilen kurallar yararlı olsun ve bundan sonra sisteme giren koşullarla eşleştiğinde kullanıcıya optimum sonucu veren sorgunun yaratılması için kullanılabilsin. Bu noktada gereksiz kuralları öğrenmemek oldukça önemlidir, çünkü gereksiz kurallar kural tablosunun boyutunun büyümesine neden olacaktır ve bu büyüme sebebiyle yararlı kuralların sorgunun optimize edilmesine olan katkısı küçülecektir. Yine bu büyüme sorgu koşuluyla ilişkili kural eşleştirme işleminin yavaşlamasına sebep olacaktır. Dolayısıyla kural tablosu bakım maliyetleri artacaktır. Bu bağlamda kural öğrenme aşamasında hangi özellikler için kural öğrenileceğini

belirlemek önemli bir husustur. Bu sebeple sisteme giren sorgu koşulu ve sonuç kümesi kullanılarak sistemin bağımlı ve bağımsız değişkenleri belirlenir ve buna istinaden veri analizi gerçekleştirilir.

Sorgu koşulu bu modülün bağımlı değişkeni, tablonun diğer özellikleri ise bağımsız değişkenler olarak düşünülür. Örneğin verilen sorgu ve tablo aşağıdaki şekilde olsun.

Sorgu: SELECT * FROM TABLO WHERE KOŞUL = 'a' ve

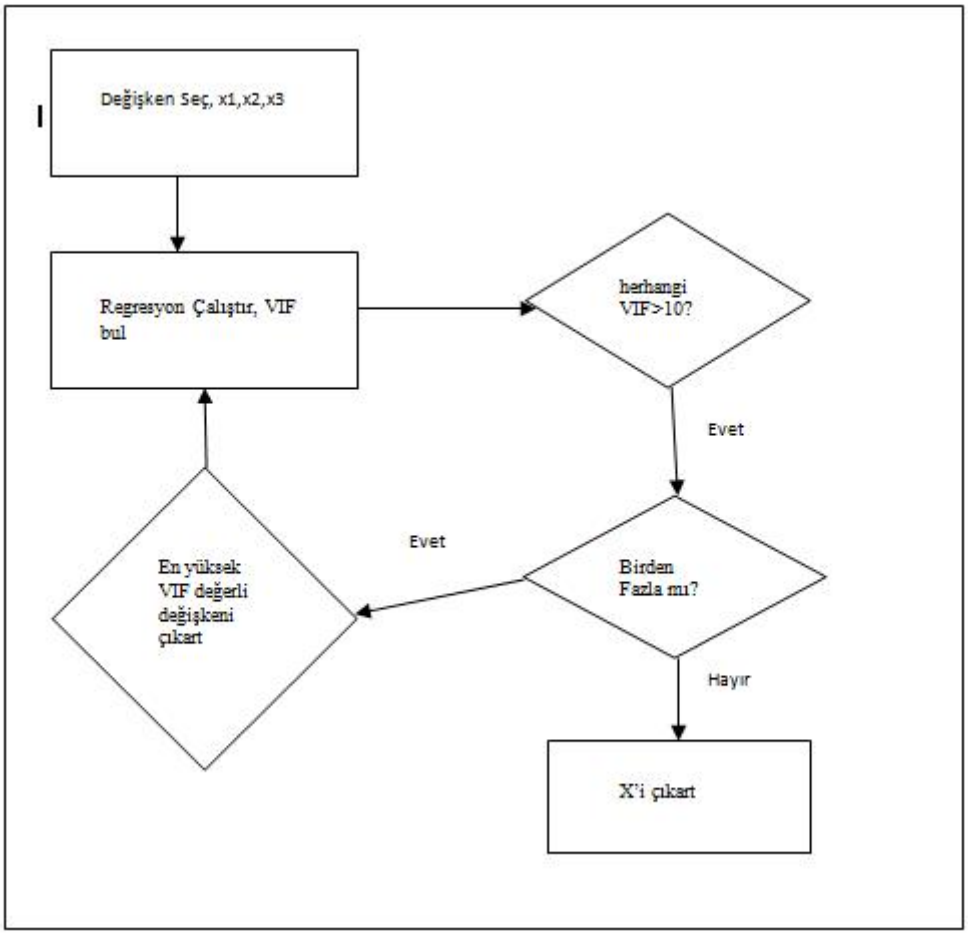
Tablo:

a	B	c	D	.....	z
				.....	
				.....	
				.....	
				.....	
				.....	

Sorgu Koşulu = Bağımlı Değişken: a

Tablo Özellikleri= Bağımsız Değişkenler: b, c, d, e...z

olarak alınacaktır. Modelde bağımlı ve bağımsız değişkenler belirledikten sonra sistem Çoklu Doğrusal Bağlantı içeren değişkenleri modelden çıkarmak üzere çalışır. Daha önce Bölüm 4'de geriye doğru eleme metodu ve adımları açıklanmıştı. Şekil 5.3'de modelin adımları verilmiştir.



Şekil 5.3 Geriye Doğru Eleme Metodu

Burada yapılan işlem bağımsız değişkenler arasından yüksek Çoklu Doğrusal Bağlantı' ya sahip olan değişkenleri modelden çıkarmaktır. Çünkü yüksek Çoklu Doğrusal Bağlantı' ya sahip özelliklerin modele olan katkıları hiç olmasa bile oldukça küçüktür.

Bu elemenin ardından model için çoklu regresyon analizi çalıştırılır ve standartlaştırılmış regresyon katsayıları hesaplanır.

Regresyon Katsayıları:

$$a = \beta_0 + \beta_1 b + \beta_2 c + \cdots \beta_{29} z$$

Standartlaştırılmış Katsayılar (değişken dönüşümünden sonra)

$$a = \alpha_1 b + \alpha_2 c + \cdots \alpha_{29} z$$

Standartlaştırılmış katsayılar önlerindeki değişkenin modele olan katkısını göreli olarak ifade ederler.

Burada amaç modele en çok katkı sağlayan değişkenleri belirlemektir. Bu bağlamda katsayıların mutlak değerlerinin ortalaması alınır, ortalamadan büyük olan katsayıların önündeki değişkenler sistemde tutulurken, küçük olanlar elenir.

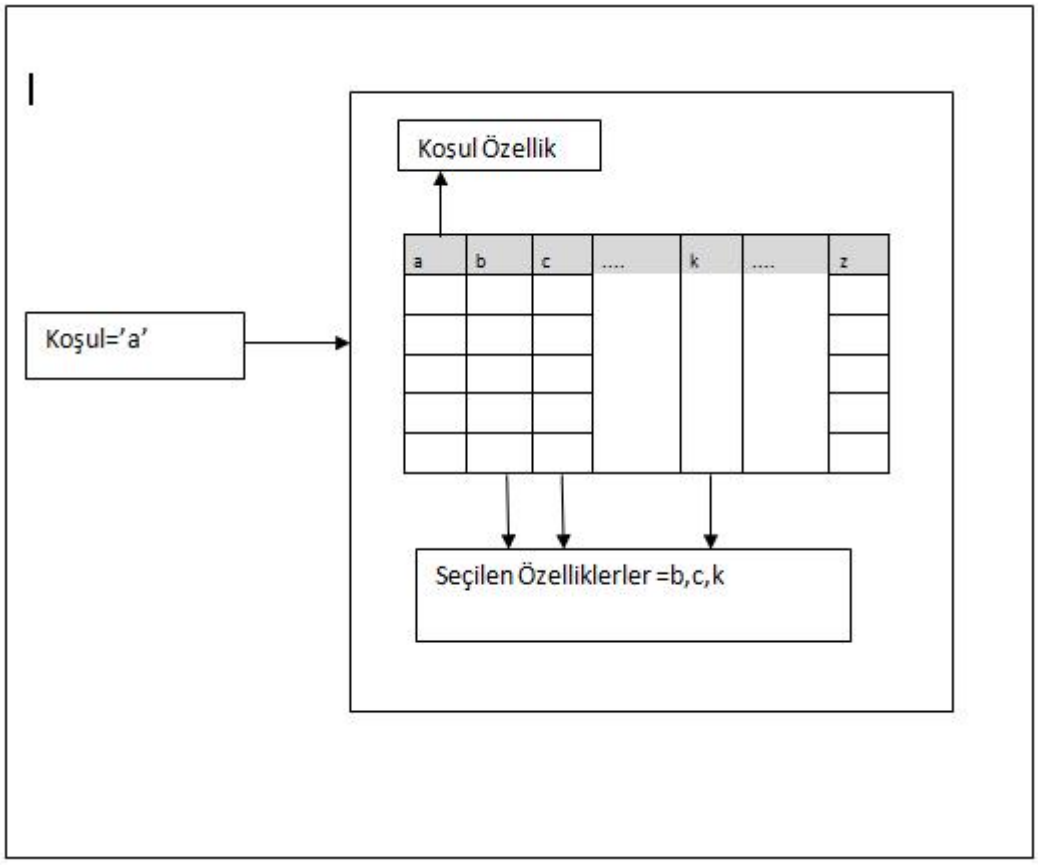
Burada gözden kaçırılmaması gereken nokta tabloya ait anahtar ve indeks değerine sahip özellikler var ise bunların modelde tutulması gerekir. Yani değişken eleme işleminde değişkenin "anahtar" veya "indeks" olup olmadığı kontrol edilir. Değişken bu özelliklere sahip değilse eleme işlemine tabi tutulur.

Değişkenlerin eleme işleminden sonra modelde kalan değişkenler otomatik kural öğrenme işlemi için kullanılırlar.

Örneğin a koşulu için, a, b, c,...z özelliklerinden oluşan tablomuzda veri analiz modülünü çalıştırdıktan sonra, elemenin ardından kalan değişkenler b, c, k olsun. Bu durumda kural öğrenme işlemi sadece b, c, k özellikleri için yapılacaktır.

5.1.2 Belirlenen Özellikler için Kuralın Öğrenme Algoritması

Sorgu koşulu için hangi özellikler için kural öğrenileceğini belirledikten sonraki aşama kuralları öğrenme aşamasıdır .Koşul='a' olduğunu ve a, b, c,..., z özelliklerine sahip tablo için elemeden sonra kalan, yani kural öğrenme işlemi için seçilen değişken b, c, k olduğu varsayılsın. Bu durumda Koşul='a' için, b, c, k özellikleri için kural öğrenme işlemi gerçekleştirilmelidir.



Şekil 5.4 Kural Öğrenme

Kural öğrenme işlemi sınır değerlere bakarak gerçekleştirilir. Kural öğrenme işlemin algoritması aşağıdaki gibi tanımlanabilir.

Belirlenen Özellikler için Kural Öğrenme Algoritması:

Adım 1: Sorgu Koşulu ile karşılaştırmak üzere özellik seçilir.

Adım 2: Sorgu Koşulu seçilen özellikler "=" operatörü ile karşılaştırılır.

Adım 3: Karşılaştırma sonucunda değer bulunur ise yani sonuç kümesi boş değil ise bulunan değer için kural oluşturma işlemi için Adım 5 'e geçilir.

Adım 4: "=" operatörü ile karşılaştırma işleminin sonuç kümesi boş ise, koşul ">=","<=" operatörleri kullanılarak seçilen özellik ile karşılaştırılır. Bulunan değerler için kural oluşturma işlemi için Adım 5'e geçilir.

Adım 5: Kural oluşturulur ve kural tablosuna eklenir.

Adım 6: Bütün özellikler için Adımlar uygulandıysa proses sonlandırılır, değil ise Adım 1'e dönülür.

Örneğin aşağıdaki örnek tablo üzerinde a=5 koşulu için, b, c, k özellikleri ile kural öğrenme işlemi gerçekleştirilirse

Çizelge 5.1 Örnek Tablo

a	b	c	k
5	3	8	3
5	3	9	7
5	3	10	8
5	3	11	67
5	3	12	67
5	3	8	45

öğrenilen kurallar şu şekilde olur:

a=5 → b = 3

a=5 → c >= 8

a=5 → c <= 12

a=5 →k >= 3

a=5 →k <=67

5.1.3 Kural Tablosu

Bir önceki bölümde öğrenilen kuralların kural tablosuna yazıldığı belirtilmişti. Aşağıda kural tablosunu oluşturan sütunların anlamları Çizelge 5.2'de verilmiştir.

Çizelge 5.2 Kural Tablosu Özellikleri

Adı	**Tipi**	**Açıklaması**
Antatt	varchar	Kural tablosunun sol tarafı-adı
Antop	varchar	Kural tablosunun sol tarafı-operatörü
Antvalue	float	Kural tablosunun sol tarafı-değeri
Antindex	bit	Kural tablosunun sol tarafı-dizin(indeks)
Antkey	bit	Kural tablosunun sol tarafı-anahtar(key)
Antrn	bigint	Kural tablosunun sol tarafı-koşulu sağlayan değer sayısı

Consatt	varchar	Kural tablosunun sağ tarafı-adı
Consop	varchar	Kural tablosunun sağ tarafı-operatörü
Consvalue	float	Kural tablosunun sağ tarafı-değeri
Consindex	bit	Kural tablosunun sağ tarafı-dizin(indeks)
Conskey	bit	Kural tablosunun sağ tarafı-anahtar(key)
Consrn	bigint	Kural tablosunun sağ tarafı-koşulu sağlayan değer sayısı
Rlt	float	Kural öğrenme zamanı
TableName	varchar	Kuralın öğrenildiği tablo

a koşulu için, b ,c, k özellikleri ile kural öğrenme işleminin ardından kural tablosu aşağıdaki

Çizelge 5.3 Kural Tablosu Örnek

Ant Att	AntOp	Ant Value	Ant Index	Ant Key	AntRn	Cons Att	ConsOp	Cons Value	Cons Index	Cons Key	ConsRn	Rlt
A	=	5	0	0	6	b	=	3	0	0	6	0,1
A	=	5	0	0	6	c	>=	8	0	0	6	0,1
A	=	5	0	0	6	c	<=	12	0	0	6	0,1
A	=	5	0	0	6	k	>=	3	0	0	6	0,1
A	=	5	0	0	6	k	<=	67	0	0	6	0,1

gibi oluşur.

5.2 Dinamik Kural Öğrenme Metodu Algoritması ve Örnek Uygulama

Dinamik kural öğrenme metodu üç aşamadan oluşmaktadır. Bunlar

Öğrenme için uygun özellikleri belirleme: Bu aşamanın detayları ve akışı Bölüm 5.1.1'de anlatılmıştır.

Belirlenen özellikler için kural öğrenme: Bu aşamanın detayları ve adımları Bölüm 5.1.2'de anlatılmıştır.

Öğrenilen kuralların kural tablosuna yazılması: Kural tablosu ve özellikleri Bölüm 5.1.3'de anlatılmıştır.

Sistem sorgunun veritabanına gönderilmesi ile başlar ve öğrenilen kuralların kural tablosuna yazılması ile sona erer. Bölüm 5.1'de detaylı olarak açıklanan aşamaları kısaca özetleyecek olursak

1. Sorgu sisteme gönderilir ve sonuç seti elde edilir.
2. Sonuç setinden bağımlı ve bağımsız değişkenler belirlenir ve tablo veri analiz teknikleri kullanılarak (çoklu regresyon analizi, geriye doğru eleme ve standartlaştırılmış katsayılar) analiz edilir.
3. Yapılan analizlerin ardından sistem için değerli değişkenler belirlenir.
4. Değerli değişkenler sistemde tutulur, diğerleri elenir.
5. Belirlenen değişkenler için kural öğrenme işlemi gerçekleştirilir.
6. Öğrenilen kurallar kural tablosuna yazılır.

Aşağıda bu metodu kullanarak gerçekleştirilmiş örnek uygulama verilmiştir.

Örnek Uygulama:

Örnek için 39 veriden oluşan gemi parametreleri(GemiParametreleri1) tablosu kullanılmıştır. Tablo EK A' da verilmiştir.

Gemi parametreleri tablosuna aşağıdaki sorgu gönderilmiştir.

" Select * from GemiParametreleri1 where LBP='21.375' "

Bu noktada, sistem oluşturulduğunda sistemin

Bağımlı Değişkeni: LBP

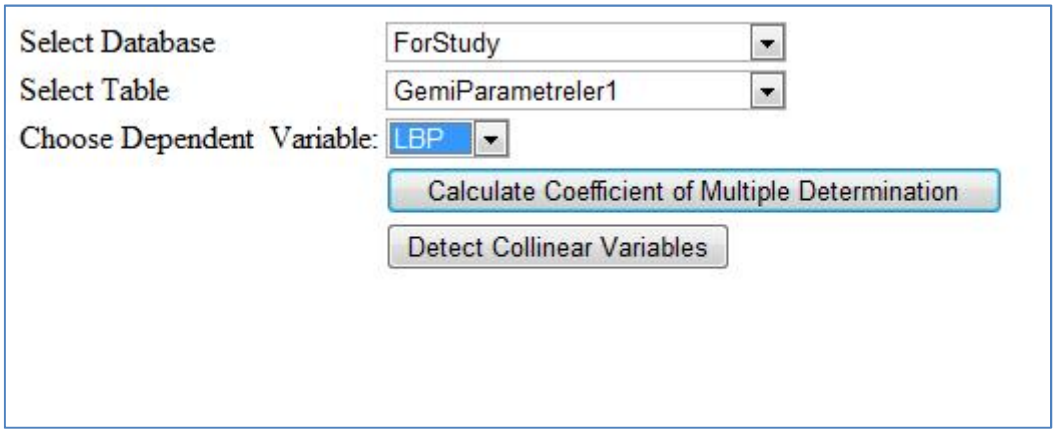
ve

Bağımsız Değişkenleri: BWL,D ,T ,LCB,LCF ,BMT ,BML ,CWP ,CP ,CM ,CB ,CVP

Olur.

Burada önemli nokta GemiParametreleri1 tablosunda söz konusu bağımlı değişken LBP olduğu durumda bu bağımlı değişkeni en iyi açıklayan bağımsız değişkenleri saptamaktır. Bu belirlemeyi yapabilmek için veri analiz modülü devreye girer.

İlk aşama geriye doğru eleme yöntemiyle çoklu doğrusal bağlantıya sahip değişkenleri modelden elemektir. Bu bağlamda, bu işlem için yazılan program çalıştırılır ve eleme işlemi gerçekleştirilir. Söz konusu sistemin ekran görüntüleri Şekil 5.5'de verilmektedir.



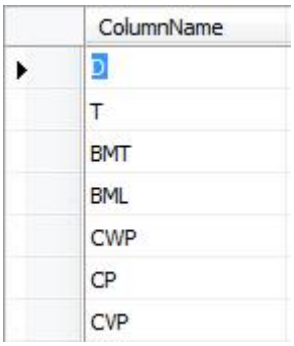
Şekil 5.5 Geriye Doğru Eleme Metodu için Geliştirilen Programın Ekran Görüntüsü

Model için yapılan son iterasyondan önce hesaplanan VIF değerleri Şekil 5.6'da görülebilir.

	ColumnName	VIF
▶	D	9,26130718811...
	T	4,85665079237...
	LCF	10,7876898912...
	BMT	6,44421185809...
	BML	5,25949558507...
	CWP	4,01669810053...
	CP	2,04554342544...
	CVP	7,69710275938...

Şekil 5.6 Geriye Doğru Eleme için Hesaplanan VIF Değerleri

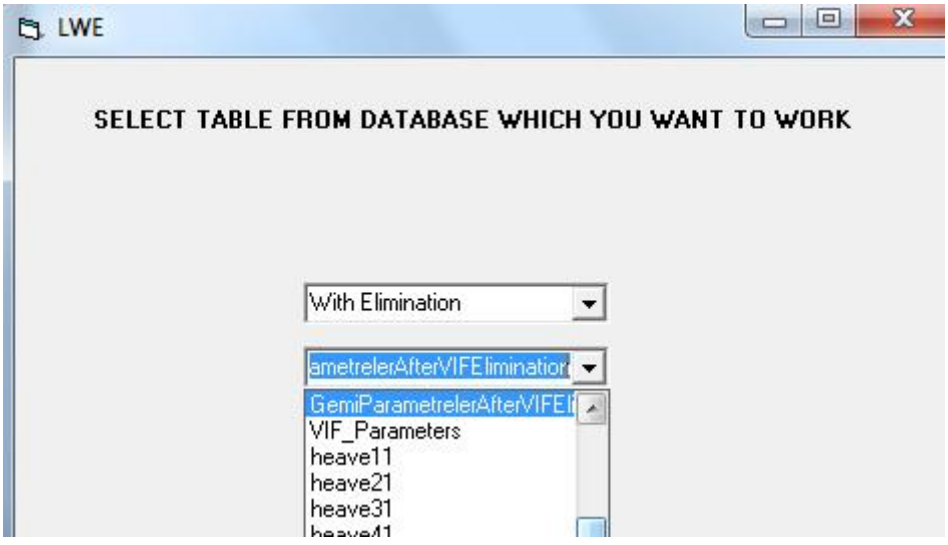
VIF değerlerinden de anlaşılacağı üzere modelden son elenen değişken LCF değişkenidir. Elemenin ardından kalan sütunlar aşağıda verilmiştir.



Şekil 5.7 Geriye Doğru Elemeden Sonra Sonuç Tablosu

Yani BWL, LCB, LCF, CM, CB değişkenleri modelden elenirken D, T, BMT, BML, CWP, CP, CVP değişkenleri modelde kalır.

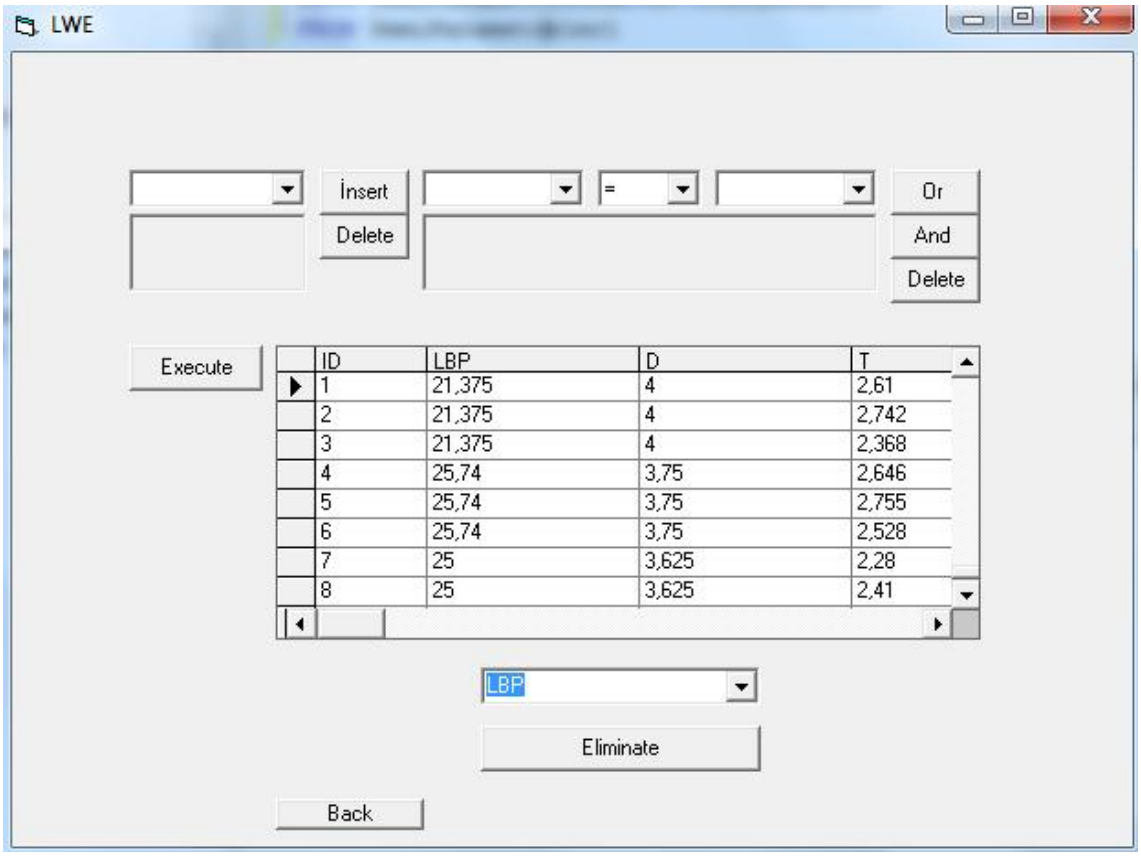
Yüksek doğrusal bağlantıya sahip değişkenleri modelden eledikten sonraki aşama, bağımlı değişken LBP için en anlamlı değişkeni bulma aşamasıdır. Bu amaçla standartlaştırılmış katsayılar hesaplanır. Katsayı ortalamaları alınır ve katsayı ortalamasından yüksek katsayıya sahip değişkenler bağımlı değişken LBP için en anlamlı değişken olarak belirlenerek modelde kalır. Bu hesaplamayı yapmak için geliştirilen program, ekran görüntüleri ve sonuçları aşağıda verilmiştir.



Şekil 5.8 Standart Katsayılarla Eleme için Metot ve Tablo Seçim Ekranı

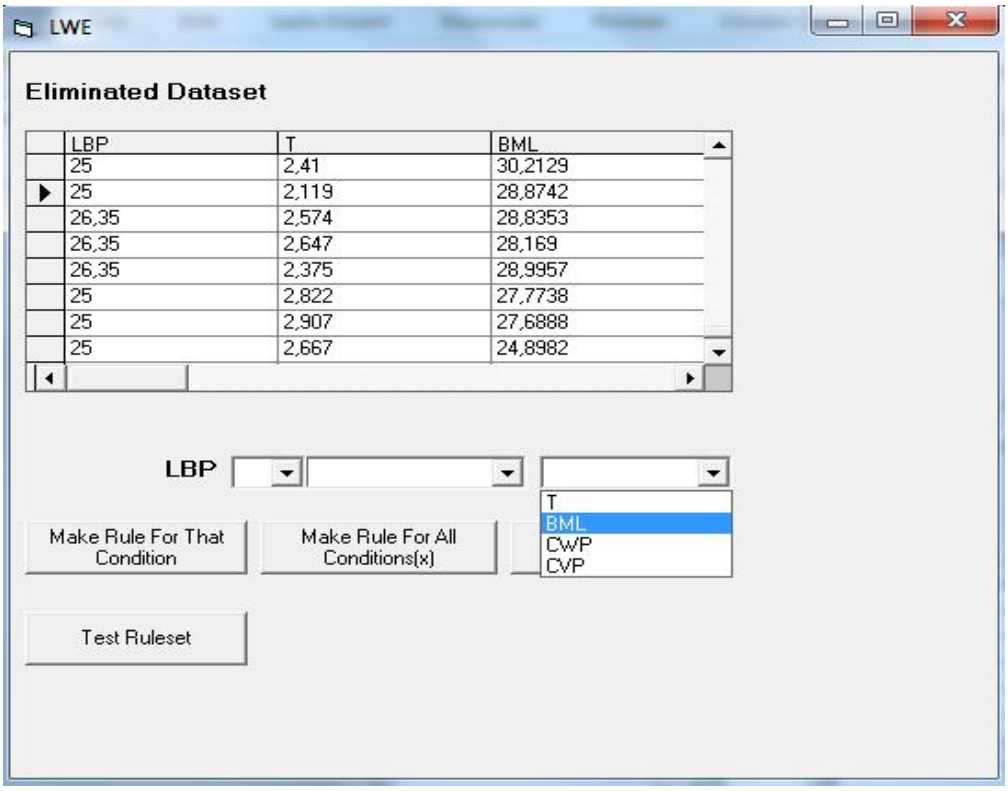
İlk olarak hangi tablo üzerinde çalışılacağı belirlenir. Daha sonraki aşama hangi sütun için standart katsayılar kullanılarak eleme yapılacağının belirlenmesidir. Bu örnekte

LBP değişkenini bağımlı değişken olarak belirlenmişti. LBP değişkeni için eleme işlemi gerçekleştirilir.



Şekil 5.9 Standartlaştırılmış Katsayılarla Eleme için Bağımlı Değişken Seçim Ekranı

Aşağıda Şekil 5.10'da eleme sonrası kalan değişkenler görülmektedir.



Şekil 5.10 Standartlaştırılmış Katsayılarla Eleme Sonrası ve Kural Öğrenme

Standart katsayılarla eleme işleminin ardından modelde T,BML,CWP ve CVP değişkenleri kalırken D, BMT ve CP sütunları elenir.

Bu durumda bu algoritmaya göre kural öğrenme T,BML,CWP,CVP ve ID sütunları için yapılmalıdır. (ID sütunu anahtar olduğu için eleme işlemine tabi tutulmaz).

Veri analizinin ardındandan bir sonraki aşama sorgunun sonuç setine göre kural öğrenme aşamasıdır. Kural öğrenme algoritması Bölüm 5.1.2'de anlatılmıştı. Öğrenilen kurallar Çizelge 5.4'de verilmektedir.

Çizelge 5.4 LBP Koşulu için Kural Tablosu

AntAtt	AntOp	AntValue	AntIndex	AntKey	AntRn	ConsAtt	ConsOp	ConsValue	ConsIndex	ConsKey	ConsRn	Duration
LBP	=	21.375	0	0	3	ID	<=	3	0	1	3	7
LBP	=	21.375	0	0	3	ID	>=	1	0	1	39	8
LBP	=	21.375	0	0	3	T	<=	2.742	0	0	27	6
LBP	=	21.375	0	0	3	T	>=	2.368	0	0	34	8
LBP	=	21.375	0	0	3	BML	<=	26.8821	0	0	16	7
LBP	=	21.375	0	0	3	BML	>=	23.247	0	0	32	9
LBP	=	21.375	0	0	3	CWP	<=	0.8741	0	0	31	5
LBP	=	21.375	0	0	3	CWP	>=	0.8067	0	0	27	8
LBP	=	21.375	0	0	3	CVP	<=	0.5046	0	0	5	6
LBP	=	21.375	0	0	3	CVP	>=	0.4676	0	0	39	10

5.3 Dinamik Kural Öğrenme Metodu Kullanılarak Elde Edilen Sonuçlar

Bölüm 5.1 de anlatılan dinamik kural öğrenme metodu kullanılarak "EPISODE" veritabanı üzerinde öğrenme gerçekleştirilmiş ve sonuçların kıyaslaması yapılmıştır. Bu veritabanı İngiltere Sağlık Dairesi'nin 'Hospital Episode Statistics - 1998/2003' web sayfasından alınmıştır:

http://www.dh.gov.uk/PublicationsAndStatistics/Statistics/HospitalEpisodeStatistics/HESFreeData

Episode tablosunun ilk 10 satırlık bölümü çizelge 5.5' de verilmektedir.

Çizelge 5.5 Episode Tablosu (10 Satır)

ID	EPISODES	ADMISSIONS	MALE	EMERGENCY	WAITING LIST	MEAN WAITIG	MEDIAN WAITING	MEAN LENGTH	MEDIAN LENGTH	MEAN AGE	AGE 0 14	AGE 15 59	AGE 60 74	AGE 75	DAY CASE	BED DAYS
1	140	118	635	136	6	5	1	6,7	4	36	414	570	202	217	2	8100,2
2	87	64	42	78	4	11	11	14,3	10	49	11	32	23	20	1	945,7
3	23	18	14	18	3	47	47	12,1	8	32	10	10	2	1	0	226
4	22	20	16	20	2	77,5	77	5,7	4	23	8	11	1	2	0	112,8
5	275	225	125	266	4	11,7	13	7,1	5	41	49	131	39	56	1	1674,3
6	128	114	50	116	2	13	13	6,6	4	26	62	42	14	10	0	706,5
7	227	198	112	197	262	53	35	5,4	4	43	334	113	422	381	248	9418,9
8	18	13	8	15	3	18,3	5	13,6	10	42	0	9	0	9	1	179,5
9	195	132	662	179	35	16,8	7	21,8	14	74	13	230	491	122	6	29697,
10	268	255	126	388	215	51,2	36	10,3	5	53	92	147	771	344	223	3242,2

Tablo sütunları ve veri tipleri Çizelge 5.6' da verilmektedir.

Çizelge 5.6 Episode Tablosu Veri Tipleri

[ID]	[int]	Primary Key
[EPISODES]	[float]	
[ADMISSIONS]	[float]	
[MALE]	[float]	
[EMERGENCY]	[float]	
[WAITING_LIST]	[float]	
[MEAN_WAITING]	[float]	
[MEDIAN_WAITING]	[float]	
[MEAN_LENGTH]	[float]	
[MEDIAN_LENGTH]	[float]	
[MEAN_AGE]	[float]	
[AGE_0_14]	[float]	
[AGE_15_59]	[float]	
[AGE_60_74]	[float]	
[AGE_75]	[float]	
[DAY_CASE]	[float]	
[BED_DAYS]	[float]	

Episode tablosuna, öğrenme için koşullar gönderilmek suretiyle öğrenme gerçekleştirilmiştir. Öğrenme ilk olarak gönderilen koşulun tüm özellikler için gerçekleştirilmiş ve sonuçlar veritabanına yazılmıştır. Daha sonra Bölüm 5.1' de anlatılan yöntem kullanılarak özellik eleme işlemi gerçekleştirilmiş, öğrenme seçilen özellikler kullanılarak yapılmış ve sonuçlar veritabanına yazılmıştır. Bölüm 5.2.1' de Dinamik Öğrenme Metoduna örnek bir uygulama verilmiştir. Bölüm 5.2.2'de Siegel Metodu ve Dinamik Öğrenme Metodu kural sayısı ve öğrenme sürelerine göre karşılaştırılmıştır.

5.3.1 Dinamik Kural Öğrenme Metodunun Örnek Uygulaması

Çizelge 5.7'de veritabanına gönderilen

"SELECT * FROM EPISODE WHERE EPISODES=1384", koşulu için özellik eleme işlemi yapılmaksızın öğrenme sonuçları görülebilir.

Çizelge 5.7 Özellik Eleme İşlemi Yapılmaksızın Öğrenilen Kurallar

ID	AntAtt	Ant Op	AntValue	AntIndex	AntKey	AntRn	ConsAtt	ConsOp	ConsValue	ConsIndex	ConsKey	ConsRn	Rlt
1	EPISODES	=	1384	0	0	4	ID	<=	27107	0	1	21109	118
2	EPISODES	=	1384	0	0	4	ID	>=	9027	0	1	26894	135
3	EPISODES	=	1384	0	0	4	ADMISSIONS	<=	1369	0	0	28558	48
4	EPISODES	=	1384	0	0	4	ADMISSIONS	>=	1226	0	0	5718	66
5	EPISODES	=	1384	0	0	4	MALE	<=	863	0	0	29710	38
6	EPISODES	=	1384	0	0	4	MALE	>=	52	0	0	16630	66
7	EPISODES	=	1384	0	0	4	EMERGENCY	<=	1077	0	0	31235	32
8	EPISODES	=	1384	0	0	4	EMERGENCY	>=	49	0	0	13710	50
9	EPISODES	=	1384	0	0	4	WAITING_LIST	<=	734	0	0	30085	30
10	EPISODES	=	1384	0	0	4	WAITING_LIST	>=	115	0	0	10420	49
11	EPISODES	=	1384	0	0	4	MEAN_WAITING	<=	40	0	0	16371	36
12	EPISODES	=	1384	0	0	4	MEAN_WAITING	>=	14	0	0	26411	56
13	EPISODES	=	1384	0	0	4	MEDIAN_WAITING	<=	21	0	0	16164	30
14	EPISODES	=	1384	0	0	4	MEDIAN_WAITING	>=	6	0	0	27422	50
15	EPISODES	=	1384	0	0	4	MEAN_LENGTH	<=	40551	0	0	22871	31
16	EPISODES	=	1384	0	0	4	MEAN_LENGTH	>=	40758	0	0	20180	49
17	EPISODES	=	1384	0	0	4	MEDIAN_LENGTH	<=	4	0	0	23421	30
18	EPISODES	=	1384	0	0	4	MEDIAN_LENGTH	>=	2	0	0	23630	49
19	EPISODES	=	1384	0	0	4	MEAN_AGE	<=	59	0	0	27079	30
20	EPISODES	=	1384	0	0	4	MEAN_AGE	>=	9	0	0	31159	50
21	EPISODES	=	1384	0	0	4	AGE_0_14	<=	1157	0	0	33024	31
22	EPISODES	=	1384	0	0	4	AGE_0_14	>=	0	0	0	33873	50
23	EPISODES	=	1384	0	0	4	AGE_15_59	<=	941	0	0	29624	32
24	EPISODES	=	1384	0	0	4	AGE_15_59	>=	150	0	0	11600	49
25	EPISODES	=	1384	0	0	4	AGE_60_74	<=	481	1	0	30576	31
26	EPISODES	=	1384	0	0	4	AGE_60_74	>=	61	1	0	10041	53
27	EPISODES	=	1384	0	0	4	AGE_75	<=	175	0	0	28816	31
28	EPISODES	=	1384	0	0	4	AGE_75	>=	15	0	0	13975	49
29	EPISODES	=	1384	0	0	4	DAY_CASE	<=	872	0	0	31210	32
30	EPISODES	=	1384	0	0	4	DAY_CASE	>=	74	0	0	9498	49
31	EPISODES	=	1384	0	0	4	BED_DAYS	<=	5248.6	0	0	28832	37
32	EPISODES	=	1384	0	0	4	BED_DAYS	>=	2968.5	0	0	7155	58

“EPISODES” koşulu için özellik eleme işlemi önce geriye doğru eleme daha sonra standart katsayılarla eleme işlemi yapılarak gerçekleştirilmiştir. Elemeden sonra kalan sütunlar Çizelge 5.8’ de gösterilmektedir. Diğer koşullar için yapılan eleme işleminin

sonuçları EK C'de verilmektedir. (Bölüm 5.1' de de belirtildiği üzere anahtar ve indeks değerine sahip özellikler eleme işlemine tabi tutulmaz)

Çizelge 5.8 EPISODES Koşulu için Eleme Sonrası Kalan Sütunlar

EPISODES -->	**Geriye Doğru Elemeden Sonra Kalan Değişkenler**	**Standart Katsayılarla Elemeden Sonra Kalan değişkenler**
	ID	*ID*
	EMERGENCY	AGE_0_14
	WAITING_LIST	AGE_15_59
	MEAN_WAITING	AGE_60_74
	MEDIAN_WAITING	AGE_75
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

Özellik eleme işlemi yapıldıktan sonra seçilen özelliklere yapılan öğrenme işlemi sonuçları ise çizelge 5.9' da görülebilir.

Çizelge 5.9 Özellik Eleme İşlemi Yapılarak Öğrenilen Kurallar

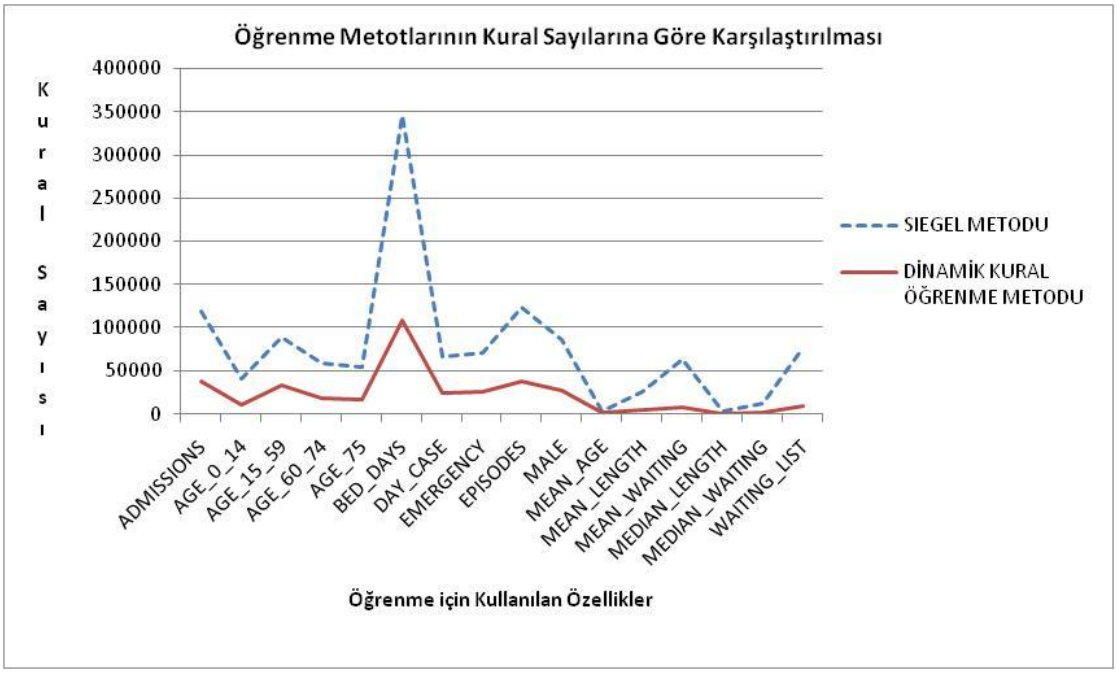
ID	Ant Att	Ant Op	Ant Value	Ant Index	Ant Key	Ant Rn	Cons Att	Cons Op	Cons Value	Cons Index	Cons Key	Cons Rn	Rlt
1	EPISODES	=	1384	0	0	4	ID	<=	27107	0	1	21109	118
2	EPISODES	=	1384	0	0	4	ID	>=	9027	0	1	26894	135
21	EPISODES	=	1384	0	0	4	AGE_0_14	<=	1157	0	0	33024	31
22	EPISODES	=	1384	0	0	4	AGE_0_14	>=	0	0	0	33873	50
23	EPISODES	=	1384	0	0	4	AGE_15_59	<=	941	0	0	29624	32
24	EPISODES	=	1384	0	0	4	AGE_15_59	>=	150	0	0	11600	49
25	EPISODES	=	1384	0	0	4	AGE_60_74	<=	481	1	0	30576	31
26	EPISODES	=	1384	0	0	4	AGE_60_74	>=	61	1	0	10041	53
27	EPISODES	=	1384	0	0	4	AGE_75	<=	175	0	0	28816	31
28	EPISODES	=	1384	0	0	4	AGE_75	>=	15	0	0	13975	49

5.3.2 Öğrenme Metotlarının Kural Sayısı ve Öğrenme Sürelerine Göre Karşılaştırılması

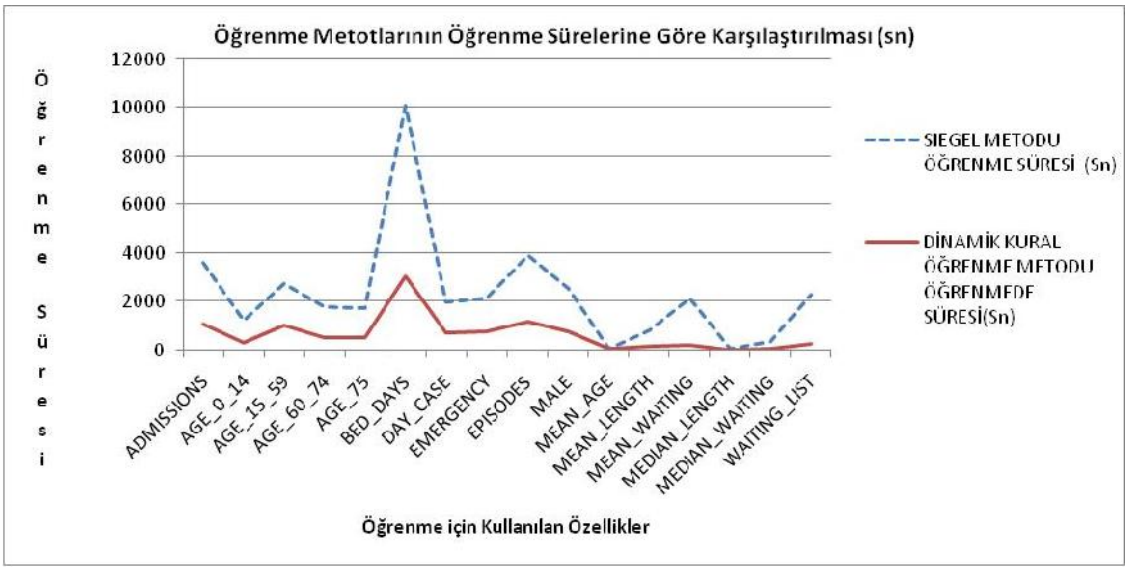
Öğrenme işlemi 55635 sorgunun sisteme gönderilmesiyle gerçekleştirilmiştir. Çizelge 5.10'da Siegel Metodu ve Dinamik Kural Öğrenme Metoduna göre çalıştırılan sorgular için öğrenilen kural sayıları ve öğrenme zamanları verilmiştir. Sonuçların kıyaslaması Çizelge 5.10' dan ve grafiksel olarak Şekil 5.11 ve Şekil 5.12'den yapılabilir.

Çizelge 5.10 Öğrenilen Kural Sayıları ve Öğrenme Süreleri

	SIEGEL METODU	SIEGEL METODU ÖĞRENME SÜRESİ (Sn)	DİNAMİK KURAL ÖĞRENME METODU	DİNAMİK KURAL ÖĞRENME METODU ÖĞRENME SÜRESİ (Sn)
ADMISSIONS	118482	3598,862	37065	1089,541
AGE_0_14	40297	1250,047	10093	299,588
AGE_15_59	89165	2755,12	33501	1009,001
AGE_60_74	58336	1782,138	18242	539,099
AGE_75	53199	1757,934	16644	530,612
BED_DAYS	345866	10074,755	108067	3049,05
DAY_CASE	66104	2009,022	24837	735,377
EMERGENCY	70490	2141,806	26460	786,038
EPISODES	123084	3899,188	38493	1183,223
MALE	84886	2563,96	26556	776,4
MEAN_AGE	2998	105,641	1506	45,049
MEAN_LENGTH	25624	825,673	4817	141,904
MEAN_WAITING	63478	2166,378	7934	233,571
MEDIAN_LENGTH	2755	88,84	348	9,702
MEDIAN_WAITING	12082	412,921	1506	45,049
WAITING_LIST	76378	2316,263	9565	256,024
TOPLAM	***1233224***	***37748,548***	***365634***	***10729,228***



Şekil 5.11 Öğrenme Metotlarının Kural Sayılarına Göre Karşılaştırması



Şekil 5.12 Öğrenme Metotlarının Öğrenme Sürelerine Göre Karşılaştırılması

Yapılan öğrenmelere sonucunda öğrenmenin öğrenilen özellikteki farklı değerlerin sayısı ve farklı değerin gerçeklediği kayıt sayısı ile bağlantılı olduğu tespit edilmiştir. Örneğin Şekil 5.12'de "Bed_Days" en fazla farklı değer sayısı ve farklı değerin gerçekleştiği kayıt sayısına sahiptir.

Ayrıca öğrenmeden elde edilen kurallar içerisinde gözlenen diğer bir husus da kuralın içeriğinden kaynaklanmaktadır. Bu husus dört ayrı grupta incelenmiştir:

1) Kuralın sağ tarafı anahtar özellik üzerinde eşitlik içeriyorsa
2) Kuralın sağ tarafı anahtar özellik üzerinde aralık içeriyorsa
3) Kuralın sağ tarafı anahtar olmayan özellik üzerinde eşitlik içeriyorsa
4) Kuralın sağ tarafı anahtar olmayan özellik üzerinde aralık içeriyorsa

Bu gruplara ait öğrenilen kurallar, öğrenme süreleri ve kural başına düşen öğrenme süresi Çizelge 5.11'de verilmiştir.

Çizelge 5.11 Anahtar Olan ve Olmayan Alanlar Üzerindeki Kural Başına Öğrenme Süreleri

	Kural Sayısı	Öğrenme Süresi (sn)	Öğrenme Süresi/Kural Sayısı
Sağ Anahtarda Eşitlik	*33999*	*600,91*	*0,0177*
Sağ Anahtar da Aralık	43272	1179,228	*0,0273*
Sağ Anahtar Olmayanda eşitlik	513097	12473,109	*0,0243*
Sağ Anahtar Olmayanda Aralık	642856	23495,301	*0,0365*

Çizelge 5.11'den görülebileceği üzere birinci grup en az, dördüncü grup ise en çok birim öğrenme süresine sahiptir.

5.4 Dinamik Kuralların Kullanılması ile Elde Edilen Sorgu Optimizasyon Sonuçları

Bu kısımda öğrenilen kuralların kullanılması esnasında sorguların çalışmasına yaptığı katkı veya başka bir deyişle yarar hesaplanmaya çalışılmıştır. Bu amaçla 878 sorgu çalıştırılmıştır.

Çizelge 5.12'de orijinal sorgu, optimum sorgu 1(Siegel), optimum sorgu 2(Dinamik Öğrenme Metodu) çalışmalarındaki toplam sürelerin karşılaştırılması verilmiştir

Çizelge 5.12 Orijinal Sorgu, Optimum Sorgu 1(Siegel), Optimum Sorgu 2(Dinamik Öğrenme Metodu) Çalışmalarındaki Toplam Sürelerin Karşılaştırılması

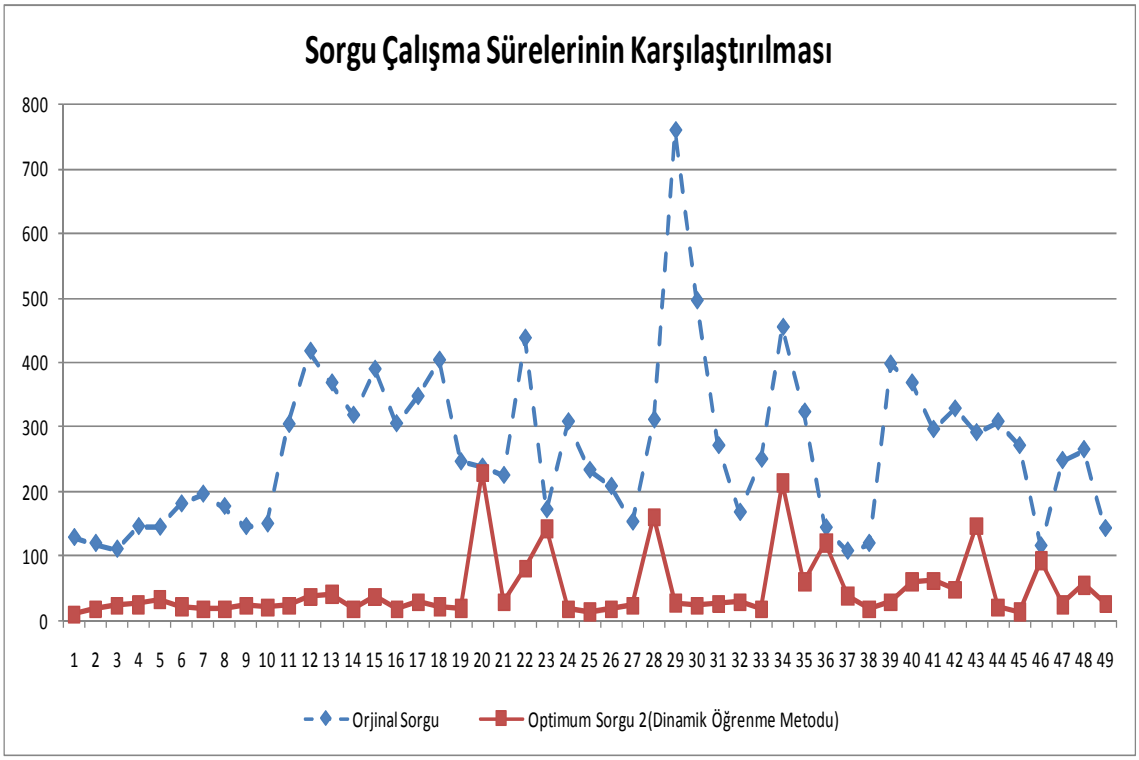
Toplam Süre (ms)		
Orijinal Sorgu	**Optimum Sorgu 1(Siegel)**	**Optimum Sorgu 2 (Dinamik Öğrenme Metodu)**
160956	115016	49446
Ortalama Süre (ms)		
Orijinal Sorgu	**Optimum Sorgu 1(Siegel)**	**Optimum Sorgu 2(Dinamik Öğrenme Metodu)**
183,32	131,00	56,32

Çizelge 5.12'den de görüleceği üzere Dinamik Öğrenme Metodu ile öğrenilen kuralların kullanımı daha faydalıdır. Bu faydanın içeriği yapılan testlerde incelendiğinde bulunan sonuçlar çizelge 5.13'de verilmiştir. Çizelgeden de görülebileceği üzere en az çalışma sürelerine sahip olan sorgularda Dinamik Öğrenme Metodu ile elde edilen anahtar da eşitlik içeren kuralların kullanılması olmuştur.

Çizelge 5.13 Orijinal Sorgu, Optimum Sorgu 1(Siegel), Optimum Sorgu 2(Dinamik Öğrenme Metodu) Çalışmalarındaki Toplam Sürelerin Karşılaştırılma Detayları

Kural Grubu	**Optimum Sorgu 1(Siegel) (ms)**	**Optimum Sorgu 2 (Dinamik Öğrenme Metodu) (ms)**
Anahtarda Eşitlik	100,14	36,07
Anahtarda Aralık	234,95	124,52
None Değerde Eşitlik	347,79	269,54
None Değerde Aralık	364,69	278,06

Sorgu bazında daha detaylı bilgi vermek amacıyla aşağıda rastgele seçilen 50 sorgu (EK D'de) ele alınmıştır. Bu sorguların orijinal sorgu çalışma süreleri ve Dinamik Öğrenme Metodu ile öğrenilen kurallar kullanılarak elde edilen optimum sorguların çalışma süreleri Şekil 5.13' de grafiksel olarak mukayese edilmiştir.



Şekil 5.13 Orijinal Sorgu ve Dinamik Öğrenme Metodu Kullanılarak Çalıştırılan Sorguların Kıyaslanması

BÖLÜM 6

SONUÇ VE ÖNERİLER

Bu tez de önerilen “Dinamik Öğrenme Metodu” Çizelge 5.9’ dan da görüleceği üzere veri analizleri istatistiksel olarak yapıldığında öğrenilen kuralların sayısını büyük oranda sınırlandırmıştır. Sorguların çalıştırılması esnasında sınırlı sayıda öğrenilen bu kuralların kullanılması anlamsal sorgu optimizasyonu sisteminin yararını azaltmamıştır. Dolayısıyla buradan veritabanı sistemlerinde tutulan gizli bilgilerin ortaya çıkarılması ve bu bilgilerin kullanılmasının sorguların çalıştırılması açısından önemi açıkça ortaya konulmuş ve bu tez amacına ulaşmıştır.

İleriye dönük çalışmalar için ([31-37]) de belirtilen noktalar ve kaynaklardan da yararlanılarak sorgu optimizasyonunda farklı öğrenme metotları geliştirilebileceği gibi farklı veri analiz metotlarına dayalı yeni metotlar da oluşturulabilir. Örneğin veri madenciliğinde kullanılan KNN algoritması değişik şekilde düzenlenerek sorgu tabanlı öğrenmeler için kullanılmasına dair çeşitli görüşlerimiz bulunmaktadır ve gelecekte olabilirliği üzerinde araştırma ve geliştirme çalışması yapmayı hedeflemiş bulunmaktayız.

Ayrıca bu tez kapsamında olmayan SQO’nun diğer modülleri içinde veri analiz metotları araştırılabilir ve yeni metotlar geliştirilebilir ya da var olan metotlar iyileştirilebilir.

En geniş anlamda düşünüldüğünde ise bu tez çalışması esnasında veri analiz metotlarının RDBMS’nin sadece sorguların çalıştırılması alanında değil, diğer alanlarında da uygulanabilirliğine yönelik çalışılmaların yapılmasının yararlı olacağı görüşüne varılmıştır.

Gerçekleştirilen yazılım sisteminin istatistiğe dayalı veri analizi yapmak için her mühendislik alanında kullanılması yararlı olacağı kanısındayız. Bu doğrultuda ilk olarak gemi inşaatı alanında gemilerin başlangıç tasarımlarına uygulaması yapılmış ve oldukça iyi sonuçlar alınmış ve (Sayli ve diğerleri [29])'da yayınlanmıştır. Diğer mühendislik alanlarında da gemi hareketlerinde olduğu gibi yeterli veriye sahip olabilmemiz durumunda sistemden dikkate değer ve tutarlı sonuçlar elde edilebilir.

Sonuç olarak elektronik olarak elimizde RDBMS içerisinde olan tüm bilgiler açığa çıkartılarak, ilgili veri tabanı üzerinde gerçekleştirilmesi istenen tüm işlemlerde bu bilgilerin kullanımı sağlamalı ve geçmiş tecrübelerden yararlanılmalıdır ki bu daha akıllı sistemlerin oluşturulmasını sağlayacaktır.

KAYNAKLAR

[1] Frawley, W.J., Piatetsky-Shapiro, G., Matheus ve C.J., (1991). "Knowledge discovery in databases: an overview.", Knowledge Discovery in Databases, 1-27

[2] Armutlu, İ.H., (2000). İşletmelerde Uygulamalı İstatistik.

[3] Çil, B., (2002). İstatistik.

[4] Hamburg, M., (1974). Statistical Analysis For Decision Making

[5] Hammer M. ve Zdondik S.B., (1980). "Knowledge-based query processing.", In Proceeding of the 6 th VLDB Conference, 1980, Montreal, Canada, 137-146

[6] King, J.J., (1981). Query optimisation through semantic reasoning, Doktora Tezi, Stanford University, USA.

[7] King, J.J., (1981). "A system for semantic query optimisation in relational databases.", In Proceeding of the 7th VLDB Conference, Sept. 1981, 510-517.

[8] Shenoy, S.T. ve Ozsoyoglu, Z.M., (1987). "A system for semantic query optimisation.", In Proceedings of the 1987 ACM-SIGMOD International Conference on Management of Data, May 1987, San Francisco, 181-195

[9] Shenoy, S.T. ve Ozsoyoglu Z.M., (1989). "Design and implementation of semantic query optimiser.", IEEE Transactions on Knowledge and Data Engineering, 1(3):344-361

[10] Chakravarthy, U.S., Fishman, D.H. ve Minker J.,(1986). "Semantic query optimization in expert systems and database systems.", In Expert Database Systems, L. Kerschberg, Ed., Benjamin/Cummings, Inc., 659-675

[11] Chakravarthy, U.S., Grant, J. ve Minker J., (1987). "Semantic query optimization: additional constraints and control strategies.", In Expert Database Systems, L. Kerschberg, Ed., Benjamin/Cummings, Inc., 345-379.

[12] Chakravarthy, U.S., Grant, J. ve Minker J., (1990). "Logic-based approach to semantic query optimization.", ACM Transactions on Database Systems, 15(2):162-207

[13] Hsu, C. ve Knoblock, C.A., (1993). "Learning database abstractions for query reformulation", Knowledge Discovery in Databases Workshop, 1993, 276-290

[14] Hsu, C. ve Knoblock, C.A., (1993). “Reformulating query plans for multidatabase systems.”, In Proceeding of the Second International Conference of Information and Knowledge Management, 1993, Washington, D.C., USA.

[15] Hsu, C. ve Knoblock, C.A., (1994). “Rule induction for semantic query optimisation.”, In Proceedings of the Eleventh International Conference on Machine Learning, 1994, 112-120

[16] Lowden, B.G.T., Robinson, J. ve LIM, K.Y., (1995). “A semantic query optimiser using automatic rule derivation.”, Proc. 5th Annual Workshop on Information Technologies and Systems, December1995, Netherlands, 68-76.

[17] Lowden, B.G.T. ve Lim K.Y., (1995). “A data-driven semantic optimiser.”, In Proceeding of ISCISX Conference, 1995, Turkey.

[18] Siegel, M.D., (1988). “Automatic rule derivation for semantic query optimisation.”, In 2nd International Conference on Expert Database Systems, Apr. 1988, 669-698.

[19] Siegel, M.D., (1989). “Automatic rule derivation for semantic query optimisation.”, Doktora Tezi, Boston University, USA.

[20] Sayli, A. ve Lowden B., (1997). “A fast transformation method to Semantic Query Optimisation.”, International Database Engineering and Applications Symposium, 1997, Concordia University, Montreal, Canada,

[21] Sayli, A. ve Lowden, B, (1997). ”Reducing rule effectiveness in SQ Transformation.”, 10th International Symposium on Methodologies for Intelligent Systems, 1997, North Carolina, USA.

[22] Schkolnick ve M., Tiberio P., (1985). “Estimating the cost of updates in a relational database.”, ACM Transactions on Database Systems, 10(2):163-179

[23] Siegel, M.D., Sciore, E. ve Salveter, S., (1991). “Rule discovery for query optimisation.”, Knowledge Discovery in Database, Ed., The AAAI Press, 1991, 411-427.

[24] Siegel, M.D., Sciore, E. ve Salveter S., (1992). ”A method for automatic rule derivation to support semantic query optimisation.”, ACM Transactions on Database Systems, 17(4):563-600

[25] Piatetsky-Shapiro, G., (1991). “Discovery, analysis and presentation of strong rules.”, Knowledge Discovery in Database, The AAAI Press, 1991, 229-248

[26] Arens, Y. ve Knoblock, A., (1992). “Planning and reformulating queries for semantically-modeled multidatabase systems.”, In Proceedings of the First International Conference on Information and Knowledge Management, 1992, Baltimore, MD.

[27] Arens, Y., Chee, C., Hsu, C. ve Knoblock, C., A., (1993). “Retrieving and integrating data from multiple information sources.” International Journal on Intelligent and Cooperative Information Systems, 2(2):127-158.

[28] Arens, Y., Chee, C., Hsu, C., In, H. ve Knoblock, C.A., (1994). ”Query processing in information mediator.”, In Proceeding in an Information Mediator, 1994, 1-12.

[29] Sayli, A., Alkan, A.D., Nabergoj, R. ve Uysal, A.O., (2007). “Seakeeping Assessmentof Fishing Vessels in Conceptual Design Stage ”, Int. Journal of Ocean Engineering, Elsevier Pub., 34(5-6):724-738.

[30] Sayli, A. ve Uysal, A.O., (2008). “A Dynamic Self-Learning Method for Semantic Query Optimisation”,Int. Journal of Technology, Policy and Management, 8(2):126-147.

[31] Marchi, F., Lopes, S. ve Petit, J.M., (2002). “Efficient Algorithms for Mining Inclusion Dependencies.”, In Proceeding of International Conference on Extending Database Technology, 2002, 464-476

[32] Pudi, V. ve Haritsa, J.R., (2003). “Generalized Closed Item Sets for Association Rule Mining.” Proceeding of 19th International Conference on Data Engineering, 2003, 714-716

[33] Bakus, J. ve Kamel, M.S., (2006). ”Higher Order Feature Selection for Text Classification.”, Knowledge and Information System, 2006, 468-491

[34] Tzitzikas, Y. ve Analyti, A., (2006). ”Mining the Meaningful Term Conjunctions from Materialised faceted Taxonomies”, Algorithms and Complexity. Knowledge and Information System, 2006, 430-467

[35] Elmasri ve Navathe, (2011). Fundemantels Of Database Sytems, 6. Baskı

[36] Hoffer, J.A., Ramesh, V. ve Topi, H., (2011). Modern Database Management,10. Baskı

[37] Jiawei, H. ve Kamber, M., (2000). Data Mining: Concepts and Techniques

EK-A

GEMİ PARAMETRELERİ TABLOSU

ID	LBP	BWL	D	T	LCB	LCF	BMT	BML	CWP	CP	CM	CB	CVP
1	21,375	6,74	4	2,61	9,8987	8,3831	2,4823	24,6739	0,8569	0,6257	0,6708	0,4197	0,4898
2	21,375	6,74	4	2,742	9,7545	8,3947	2,3538	23,247	0,8741	0,6417	0,6874	0,4411	0,5046
3	21,375	6,74	4	2,368	10,2311	8,5496	2,6462	26,8821	0,8067	0,5922	0,6369	0,3772	0,4676
4	25,74	7	3,75	2,646	12,5654	10,4393	2,0249	31,3183	0,8408	0,5861	0,8237	0,4828	0,5742
5	25,74	7	3,75	2,755	12,4155	10,2824	1,9827	31,0618	0,864	0,5988	0,8307	0,4974	0,5757
6	25,74	7	3,75	2,528	12,7274	10,9708	2,063	28,9697	0,7958	0,5725	0,8155	0,4669	0,5867
7	25	7,2	3,625	2,28	12,3395	10,893	2,3498	31,7088	0,8322	0,6317	0,804	0,5079	0,6103
8	25	7,2	3,625	2,41	12,2036	10,6626	2,2716	30,2129	0,8525	0,6457	0,8146	0,526	0,617
9	25	7,2	3,625	2,119	12,4753	11,693	2,4164	28,8742	0,7703	0,6152	0,7892	0,4855	0,6303
10	26,35	7,5	3,55	2,574	12,8849	11,2478	2,237	28,8353	0,8133	0,6238	0,8102	0,5054	0,6214
11	26,35	7,5	3,55	2,647	12,8109	11,1546	2,1938	28,169	0,823	0,6303	0,8155	0,514	0,6245
12	26,35	7,5	3,55	2,375	13,0837	11,7668	2,3437	28,9957	0,7708	0,6057	0,7943	0,4811	0,6242
13	25	8	4,05	2,822	11,7885	9,9261	2,2573	27,7738	0,8754	0,6676	0,8239	0,55	0,6283
14	25	8	4,05	2,907	11,698	9,815	2,2084	27,6888	0,8898	0,676	0,8281	0,5598	0,6291
15	25	8	4,05	2,667	11,9284	10,5663	2,3282	24,8982	0,8189	0,6533	0,8156	0,5328	0,6506
16	20,5	6,97	4	2,52	10,0765	8,9094	2,319	21,0793	0,804	0,6081	0,6884	0,4186	0,5206
17	20,5	6,97	4	2,696	9,9283	8,7911	2,1703	20,1951	0,834	0,628	0,7083	0,4448	0,5333
18	20,5	6,97	4	2,273	10,3035	9,2492	2,5284	21,0063	0,7423	0,5796	0,6556	0,38	0,5119
19	25	8	4,05	2,822	11,6633	9,9385	2,2081	28,6238	0,8979	0,6757	0,8562	0,5785	0,6443
20	25	8	4,05	2,906	11,5873	9,9551	2,1365	27,6864	0,9032	0,684	0,8595	0,5879	0,6509
21	25	8	4,05	2,666	11,8136	10,3136	2,29	27,6186	0,8595	0,6598	0,8497	0,5606	0,6522
22	27,25	7,3	3,9	2,866	13,6641	11,9851	1,7541	25,3548	0,782	0,612	0,8302	0,5081	0,6497
23	27,25	7,3	3,9	3,049	13,4927	11,6011	1,689	26,0196	0,8184	0,6256	0,8403	0,5257	0,6424
24	27,25	7,3	3,9	2,704	13,8045	12,3184	1,8206	25,2461	0,7534	0,6007	0,82	0,4926	0,6538
25	21	7,58	3,982	2,885	10,104	8,9083	2,4023	20,7611	0,8605	0,6795	0,6842	0,4649	0,5403
26	21	7,58	3,982	3,15	9,9223	8,902	2,1644	19,0374	0,8873	0,7054	0,7081	0,4995	0,5629
27	21	7,58	3,982	2,75	10,205	9,1027	2,5201	20,5826	0,8312	0,6655	0,6708	0,4464	0,5371
28	30,8	10,4	5,6	2,687	13,0722	11,936	3,462	35,9909	0,7656	0,6645	0,7625	0,5067	0,6618
29	30,8	10,4	5,6	3,057	12,8842	12,0674	3,0108	30,8772	0,7831	0,6907	0,7805	0,5391	0,6884
30	30,8	10,4	5,6	2,517	13,1945	11,8899	3,7203	38,8082	0,7559	0,65	0,7532	0,4896	0,6477

31	20	6,534	3,2	2,482	9,7115	8,559	1,9253	24,0882	0,7988	0,6585	0,6	0,3951	0,4946
32	20	6,534	3,2	2,662	9,5645	8,593	1,7896	22,318	0,8237	0,6815	0,6211	0,4233	0,5139
33	20	6,534	3,2	2,252	9,9345	8,9364	2,0998	23,9574	0,7379	0,6271	0,5691	0,3569	0,4837
34	27,3	6,8	3,75	2,492	13,2533	12,2044	1,9381	35,3149	0,8844	0,719	0,7828	0,5628	0,6364
35	27,3	6,8	3,75	2,737	13,1006	12,1295	1,7958	32,8998	0,915	0,7396	0,8016	0,5929	0,648
36	27,3	6,8	3,75	2,312	13,3698	12,5808	2,0294	34,7235	0,8405	0,7031	0,7669	0,5392	0,6415
37	28	9,15	4,5	2,702	13,6737	11,9819	2,9783	32,6828	0,8544	0,6909	0,82	0,5665	0,663
38	28	9,15	4,5	3,055	13,3586	11,6589	2,6635	29,4934	0,8846	0,7188	0,8369	0,6016	0,6801
39	28	9,15	4,5	2,47	13,8937	12,3658	3,2347	34,0685	0,8232	0,6705	0,8067	0,5409	0,6571

EK-B

GEMİLERİN GEOMETRİK PARAMETRE TANIMLARI

L : Teknenin tam boyu.

B : Gemi gövdesinin orta kesitindeki genişliği.

T : Teknenin su-çekimi (Yüzdüğü su seviyesinin tekne dibinden mesafesi).

D : Teknenin derinliği (Tekne dibinden güverte hattına kadar olan mesafe).

Ñ : Teknenin su altında kalan hacmi (Deplasman).

L/ Ñ(1/3): L'nin Hacmin küp-köküne bölümü (metre/metre), Boy / Hacim.

CB= Ñ **/ (LWL. B.T)** Geminin su altındaki hacminin (Su hattı Boyu * Genişlik * Su-çekimi) küp şeklindeki kutuya oranıdır. Yani geminin dolgunluk veya narinlik katsayısını verir.

CWP: Su hattı kesim yüzey alanının LWL*B boyutlarındaki dikdörtgen alanına oranı olan 'Su Hattı dolgunluk katsayısı' dır (Waterplane area coefficent).

$$CWP = AWP / (LWL* B)$$

CP: Prizmatik katsayıdır. Gemi su hattı hacminin, tabanı orta kesit alanı A olan ve yüksekliği **LWL** (su hattını kesen boy) ve genişliği B olan prizmanın hacmine oranıdır. Yani prizmatik katsayı CP, gemi deplasman hacminin, gemi boyunda ve kesiti geminin orta kesit alanına sahip olan prizmatik bir cismin hacmine oranıdır. **Cp =** Ñ**/ (LWL.A)** "Longitudinal Prismatic Coefficient" denilir.

CVP: Geminin su altındaki hacminin (Su hattı kesişim yüzeyinin alanı * Su-çekimi) prizma hacmine oranıdır. Yani Ñ 'nın tabanı Su hattı kesişim yüzeyi (AWL) ve yüksekliği T olan prizmaya oranı olan bir dolgunluk katsayısıdır. Vertical prismatic coefficient'dir. CVP **=** Ñ / (AWL*T)

CWPF , CWPA , CVPF, CVPA

F: Forward yani baş taraf.

A: Aft yani arka taraftır.

Gemiyi ortadan ikiye ayrıldığı varsayılırsa, yani arka taraf ve baş taraf iki ayrı gemi iseler Su hattı Alan Katsayısı **CWP** ve Düşey Prizmatik Katsayı **CVP,** bu kesilen iki parça için ayrı ayrı hesaplanırsa CWP baş, CWP arka ve CVP baş, CVP arka elde edilir. Yani baş ve arka parçalara ait hesaplanmış olan daha belirleyici dolgunluk katsayılarıdır.

LCF: "Longitudinal Center of Flotation". Yüzme merkezinin boyuna konumu denir. Su hattı kesit yüzey alan merkezinin arka dikeye olan boyuna mesafesidir.

LCB: "Longitudinal Center of Buoyancy". Su altındaki hacmin merkezi olan B noktasının arka dikeye olan boyuna mesafesidir.

EK-C

ÖZELLİK ELEME İŞLEMİ SONUÇLARI

EPISODES	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	AGE_0_14
	WAITING_LIST	AGE_15_59
	MEAN_WAITING	AGE_60_74
	MEDIAN_WAITING	AGE_75
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

ADMISSIONS	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	AGE_0_14
	WAITING_LIST	AGE_15_59
	MEAN_WAITING	AGE_60_74
	MEDIAN_WAITING	AGE_75
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

MALE	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	EMERGENCY
	WAITING_LIST	WAITING_LIST
	MEAN_WAITING	AGE_0_14
	MEDIAN_WAITING	AGE_60_74
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

EMERGENCY	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	WAITING_LIST	WAITING_LIST
	MEAN_WAITING	AGE_15_59
	MEDIAN_WAITING	AGE_60_74
	MEAN_LENGTH	AGE_75
	MEDIAN_LENGTH	DAY_CASE
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

WAITING_LIST	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	DAY_CASE
	MEAN_WAITING	
	MEDIAN_WAITING	
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

MEAN_WAITING	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	MEDIAN_WAITING
	WAITING_LIST	
	MEDIAN_WAITING	
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

MEDIAN_WAITING	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	MEAN_WAITING
	WAITING_LIST	
	MEAN_WAITING	
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

MEAN_LENGTH	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	MEDIAN_LENGTH
	WAITING_LIST	BED_DAYS
	MEAN_WAITING	
	MEDIAN_WAITING	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

MEDIAN _LENGTH	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	MALE	MEAN_LENGTH
	EMERGENCY	
	WAITING_LIST	
	MEAN_WAITING	
	MEDIAN_WAITING	
	MEAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

MEAN_AGE	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	EMERGENCY
	WAITING_LIST	MEAN_WAITING
	MEAN_WAITING	MEDIAN_WAITING
	MEDIAN_WAITING	AGE_60_74
	MEAN_LENGTH	AGE_75
	MEDIAN_LENGTH	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

AGE_0_14	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	MALE	MALE
	EMERGENCY	WAITING_LIST
	WAITING_LIST	AGE_60_74
	MEAN_WAITING	
	MEDIAN_WAITING	
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

AGE_15_59	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	EMERGENCY
	WAITING_LIST	WAITING_LIST
	MEAN_WAITING	AGE_75
	MEDIAN_WAITING	DAY_CASE
	MEAN_LENGTH	BED_DAYS
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_60_74	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

AGE_60_74	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	MALE	MALE
	EMERGENCY	AGE_0_14
	WAITING_LIST	AGE_75
	MEAN_WAITING	DAY_CASE
	MEDIAN_WAITING	
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_75	
	DAY_CASE	
	BED_DAYS	

AGE_75	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	EMERGENCY
	WAITING_LIST	AGE_60_74
	MEAN_WAITING	DAY_CASE
	MEDIAN_WAITING	BED_DAYS
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	DAY_CASE	
	BED_DAYS	

DAY_CASE	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	EMERGENCY
	WAITING_LIST	WAITING_LIST
	MEAN_WAITING	AGE_15_59
	MEDIAN_WAITING	AGE_75
	MEAN_LENGTH	BED_DAYS
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	BED_DAYS	

BED_DAYS	Geriye Doğru Elemeden Sonra Kalan Değişkenler	Standart Katsayılarla Elemeden sonra kalan değişkenler
	ID	ID
	EMERGENCY	AGE_0_14
	WAITING_LIST	AGE_15_59
	MEAN_WAITING	AGE_75
	MEDIAN_WAITING	DAY_CASE
	MEAN_LENGTH	
	MEDIAN_LENGTH	
	MEAN_AGE	
	AGE_0_14	
	AGE_15_59	
	AGE_60_74	
	AGE_75	
	DAY_CASE	

EK-D

RASTGELE SEÇİLEN SORGULAR

Aşağıda rastgele seçilen 50 sorgu tablo halinde verilmiştir.

Orijinal Sogu	Orijinal Sorgu Çalışma Süresi	Optimize Edilmiş Sorgu Çalışma Süresi
select * from EPISODE4 where EPISODES=6292	129	11
select * from EPISODE4 where EPISODES=1558	120	19
select * from EPISODE4 where EPISODES=27568	111	25
select * from EPISODE4 where EPISODES=2785	146	26
select * from EPISODE4 where EPISODES=9253	145	34
select * from EPISODE4 where EPISODES=3847	181	23
select * from EPISODE4 where EPISODES=10916	196	19
select * from EPISODE4 where EPISODES=10428	177	18
select * from EPISODE4 where EPISODES=8560	146	24
select * from EPISODE4 where EPISODES=918	150	22
select * from EPISODE4 where EPISODES=8995	304	25
select * from EPISODE4 where EPISODES=2325	417	37
select * from EPISODE4 where EPISODES=2322	368	42
select * from EPISODE4 where EPISODES=7034	318	19
select * from EPISODE4 where EPISODES=9609	389	38
select * from EPISODE4 where EPISODES=43407	305	18
select * from EPISODE4 where EPISODES=2828	347	30
select * from EPISODE4 where EPISODES=2243	403	23
select * from EPISODE4 where EPISODES=3777	246	20
select * from EPISODE4 where EPISODES=3141	238	229
select * from EPISODE4 where EPISODES=8306	225	29
select * from EPISODE4 where EPISODES=30850	437	81

select * from EPISODE4 where EPISODES=1130	172	143
select * from EPISODE4 where EPISODES=7593	308	18
select * from EPISODE4 where EPISODES=5379	233	15
select * from EPISODE4 where EPISODES=9649	208	19
select * from EPISODE4 where EPISODES=18561	153	25
select * from EPISODE4 where EPISODES=574	311	161
select * from EPISODE4 where EPISODES=7778	758	28
select * from EPISODE4 where EPISODES=1690	495	24
select * from EPISODE4 where EPISODES=4064	271	27
select * from EPISODE4 where EPISODES=6178	168	30
select * from EPISODE4 where EPISODES=3524	250	19
select * from EPISODE4 where EPISODES=505	454	214
select * from EPISODE4 where EPISODES=2773	323	61
select * from EPISODE4 where EPISODES=2803	144	121
select * from EPISODE4 where EPISODES=28394	108	39
select * from EPISODE4 where EPISODES=7368	120	19
select * from EPISODE4 where EPISODES=5340	397	30
select * from EPISODE4 where EPISODES=8192	368	61
select * from EPISODE4 where EPISODES=1388	296	62
select * from EPISODE4 where EPISODES=3186	328	49
select * from EPISODE4 where EPISODES=928	291	147
select * from EPISODE4 where EPISODES=2369	308	21
select * from EPISODE4 where EPISODES=3383	271	15
select * from EPISODE4 where EPISODES=5904	116	94
select * from EPISODE4 where EPISODES=14695	248	26
select * from EPISODE4 where EPISODES=2617	265	56
select * from EPISODE4 where EPISODES=11058	143	27

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı **:** Ayşe Öncü UYSAL

Doğum Tarihi ve Yeri **:** 14.11.1978 – Denizli

Yabancı Dili **:** İngilizce

E-posta **:** oncuuysal@yahoo.com

ÖĞRENİM DURUMU

Derece	**Alan**	**Okul/Üniversite**	**Mezuniyet Yılı**
Y. Lisans	Matematik Mühendisliği	Yıldız Teknik Üniversitesi	2004
Lisans	Matematik Mühendisliği	Yıldız Teknik Üniversitesi	2001
Lise	Fen-Matematik	Özel Işık Lisesi	1996

YAYINLARI

Makale

1. Saylı, A., Alkan, A.D., Nabergoj, R. ve **Uysal, A.O.,** (2007). "Seakeeping Assessment of Fishing Vessels in Conceptual Design Stage ", Int. Journal of Ocean Engineering, Elsevier Pub., Vol. 34, Nos 5-6, 724-738.

2. Saylı, A., **Uysal, A.O**, (2008). "A Dynamic Self-Learning Method for Semantic Query Optimisation",Int. Journal of Technology, Policy and Management, Vol. 8, No 2, 126-147.