Hassas Bilgi Gizleme Uygulaması
Sensitive Knowledge Hiding Application

Harun Gökçe, Osman Abul

Bilgisayar Mühendisliği Bölümü
TOBB Ekonomi ve Teknoloji Üniversitesi
hgokce, osmanabul@etu.edu.tr

Özet

Çeşitli organizasyonlar işleri ile ilgili verileri kendi veritabanlarında biriktirirler. Bunlardan bazıları topladıkları verileri olduğu gibi üçüncü parti kişilere araştırma amacıyla yayınlarlar. Fakat veriyi olduğu gibi yayınlama, yayınlanan verilerde bulunabilecek hassas bilginin açığa çıkarılma riskini de beraberinde getirir. Bu riskten kaçınmak için, yayınlanacak veri arındırılmalıdır; ancak bu şekilde yayınlanacak veri hassas bilgi içermez ve güvenle yayınlanabilir. Literatürde bu tip veri paylaşımı mahremiyet kaygılı veri yayınlama *olarak bilinir. Problemin yaygın çözümü, hassas bilgilerin belirlenmesi ve devamında yayınlanacak olan veritabanının bir dönüşüm işlemine tabi tutularak bu bilgilerin temizlenmesi biçimindedir. İşlem sonucunda elde edilen veritabanının tüm hassas bilgilerden arınmış ve aynı zamanda asıl veritabanına olabildiğince benzemesi gerekir. Literatürde, farklı tip veritabanlarından hassas bilgi gizlemeye yönelik birçok ayrı çalışma önerilmiştir. Bu çalışmamızda, farklı tip veritabanlarından hassas bilgileri gizleyebilecek bütüncül bir uygulamamızı sunuyoruz.*

Abstract

Diverse organizations store their business related data in their own databases. Some of them publish their data as is to third parties for research purposes. However, this way of data sharing has the risk of disclosing sensitive knowledge which the database may entail. For safe publishing the database must be sanitized before sharing; so that no sensitive knowledge is conveyed. The process is known as privacy-aware data publishing *in the literature. The process requires (i) identification of sensitive knowledge, and (ii) removal of them from the to-be-released database. The challenge here is to hide all sensitive knowledge, and at the same time keep the released version as similar as possible to the original one. In the literature, there exists separate proposals to hide various kinds of sensitive knowledge formats from diverse databases. In this work, we present a single unified platform for this task.*

Bu çalışma TÜBİTAK tarafından 108E016 nolu 1001 projesi kapsamında desteklenmektedir.

1. Giriş

Üçüncü parti kişilere araştırma amacıyla yayınlanan veritabanlarında, veritabanı sahibinin başkalarınca bilinmesini istemediği bilgiler yer alabilir. Hatta bu bilgilerin çoğu sakınılan kişiler tarafından, gelişmiş veri madenciliği teknikleri kullanılarak, veritabanından kolayca çıkarılabilir. Böyle bir durum, veritabanı sahipleri tarafından arzu edilmeyen bir sonuç olduğundan, veritabanı sahipleri veritabanlarını hassas bilgilerini içermeyecek bir şekilde yayınlamayı tercih ederler. Bu yayınlama yaklaşımı literatürde *mahremiyet kaygılı veri yayınlama* ismiyle yer alır ve 1991'den beri aktif bir araştırma konusudur[1].

Bir an için, bir alışveriş marketinin, markette yapılan tüm alışverişleri, her alışveriş birden fazla satılan ürün bilgisini içerecek şekilde tuttuğunu düşünelim. Bu market, uzun bir süre sonucunda oluşan tüm alışverişler veritabanını, satılan ürünler arasındaki birliktelik bağımlılığını belirlemek, hangi ürünlerin ne kadar çok ya da ne kadar az satıldığını öğrenmek gibi farklı çıkarımlar elde edecek analizler yapılabilmesi için üçüncü parti kişilere paylaşsın. Diğer yandan bu marketin sattığı bazı ürünleri kendisinin geliştirip market içerisinde uygun bir şekilde rafladığını, bu ürünlerin de diğer bazı ürünlerle çok sayıda satıldığını düşünelim. Esasında bu, o market için bir ürün satma stratejisidir ve başkalarınca öğrenilmemesi gerekir, dolayısıyla hassas bir bilgidir. Fakat bu bilgi yayınlanan alışveriş veritabanı üzerinde bir takım veri madenciliği teknikleri ile açığa çıkarılabilir bir bilgidir ve böyle bir sonuç, o market için hassas olan bu bilginin ifşâ olması demektir.

Veritabanlarının doğrudan yayınlanması bir yanıyla farklı analiz çalışmalarına konu olduğu için yararlı, diğer yanıyla içerebileceği hassas bilgilerin açığa çıkması ihtimali nedeniyle risklidir. Bu ikilemden kaçınmanın yolu hassasiyet kaygılı veri yayınlamadır. Hassasiyet kaygılı veri yayınlama, veritabanının, içinde barındırabileceği hassas bilginin silinmesinden ya da değiştirilmesinden sonra yayınlanmasıdır. Bu şekilde, yayınlanan veritabanı herhangi bir şekilde risk taşımaz ve böylece hassas bilgilerin ifşasından kaçınılmış olur.

Yayınlanabilecek veritabanları gerçek hayatta farklı biçimde bulunabilirler. Yukarıda değinilen market ürünleri örneği öge kümeleri tipinden bir veri biçimidir. Bunun yanında herhangi bir kişinin belirli bir süre gezmesiyle, uğradığı yerlerin oluşturduğu veri dizgi tipinde bir veri biçimi; alt alta birçok kategori bulunan bir web sitesinde kullanıcıların gezinmesiyle

oluşan veri de ağaç tipinde veriler olarak örneklendirilebilirler. Farklı tipteki veritabanlarından hassas bilgi gizleme için literatürde birçok çalışma mevcuttur. Var olan bu çalışmalarda genelde tek tip bir veriden hassas bilgi gizleme problemi üzerinde yoğunlaşılmıştır. Biz ise bu çalışmamızda hassas bilgi gizleme için geliştirdiğimiz, farklı tip verilerden hassas bilgi gizleme işlemini yapabilecek, literatürdeki bazı algoritmaları da içeren bütüncül uygulamamızı tanıtacağız.

2. Tanımlamalar

Bu bölümde, sırasıyla, farklı yapıdaki verilerden hassas bilgi gizleme problemleri tanımlanmaktadır.

2.1. Öge Kümesi gizleme

$\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ şeklinde her ögenin bir sembolle gösterildiği bir öge kümesi olsun. Bir T işlemi, $\mathcal{I}$ kümesinin boş olmayan bir alt kümesidir, *mesela*, $T \in 2^{\mathcal{I}} - \emptyset$. Bir $\mathcal{D}$ veritabanı, $\mathcal{D} = \{t_1, t_2, \ldots, t_m\}$ şeklinde gösterilir ve her t_i bir işlem olmak üzere bir işlemler kolleksiyonudur.

Tanım 1 (Destek Değeri) *X diye bir öge kümesi veriliyor olsun, X'in $\mathcal{D}$ veritabanındaki* destek kümesi *X'i alt küme olarak içeren tüm işlemlerin kümesidir ve $S_{\mathcal{D}}(X)$ şeklinde gösterilir. Formal olarak, $S_{\mathcal{D}}(X) = \{T : X \subseteq T \wedge T \in \mathcal{D}\}$. X'in $\mathcal{D}$'deki* desteği, *$sup_{\mathcal{D}}(X)$ şeklinde gösterilir ve $S_{\mathcal{D}}(X)$'in boyutu kadardır,* mesela, *$sup_{\mathcal{D}}(X) = \mid S_{\mathcal{D}}(X) \mid$.*

Tanım 2 (Sık Öge Kümesi Madenciliği[2]) *σ negatif olmayan, kullanıcı tanımlı bir tamsayı olsun ve* destek eşiği *olarak belirtilsin. Bir $\mathcal{D}$ veritabanında, destek değeri σ'dan küçük olmayan öge kümelerine,* sık *öge kümeleri denir. Tüm sık öge kümelerinin kümesi şu şekilde ifade edilir: $F_{(\mathcal{D},\sigma)} = \{X : X \subseteq I, X \neq \emptyset, sup_D(X) \geq \sigma\}$. Sık öge kümesi madenciliği sağlanan $\mathcal{D}$ ve σ değerlerinden yararlanılarak $F_{(D,\sigma)}$ şeklinde gösterilir.*

Tanım 3 (Hassas Öge Kümesi Gizleme) $P_h = \{X_i \mid X_i \in 2^{\mathcal{I}} \wedge i = 1, 2, \ldots, n\}$, *$n$ tane öge kümesinden oluşan bir küme olsun ve bu küme hassas bilgi olarak addedilsin.* Hassas öge kümesi gizleme *problemi, verilen bir hassasiyet eşiği, ψ'nden yararlanılarak, $\mathcal{D}$ veritabanındaki işlemleri bir dönüşümden geçirip $\mathcal{D}'$'yi elde etmektir. Öyleki, dönüşüm sonucunda şunlar sağlanmış olmalıdır:*

- $\forall X_i \in P_h : sup_{\mathcal{D}'}(X) < \psi$.
- $\sum_{X \in (2^{\mathcal{I}} - \emptyset) \setminus \mathcal{P}_h} \mid sup_{\mathcal{D}}(X) - sup_{\mathcal{D}'}(X) \mid$ *minimum olmalıdır.*

Dönüşüm işlemi esasında veritabanının hassas bilgilerden arındırılmasıdır. Arındırma sonucunda yayınlanabilir olan $\mathcal{D}'$ veritabanı elde edilir. Arındırma işlemi, ilk olarak, tüm hassas öge kümelerinin arındırma sonucunda oluşan veritabanındaki destek değerlerinin verilen hassasiyet eşiğinden büyük olmamasını; ikinci olarak da veritabanı bütünlüğünü korumayı bir diğer ifadeyle de $\mathcal{D}$ ile $\mathcal{D}'$'nin aynılığını sağlamaya odaklanmayı gerektirir. Arındırma işlemi, literatürde, hassas öge kümelerini içeren işlemlerden bazı ögeleri silmek ya da değiştirmek suretiyle o işlemlerin hassas ögeleri artık içermemesi şeklinde yer almaktadır. Bu işleminin yan etkisi, hassas öge kümelerini gizlerken, hassas olmayan bazı öge kümlerini de gizlemesi veya destek değerlerinin düşmesine neden olmasıdır. Atallah [3] hassas öge kümelerini gizleme probleminin en iyi çözümünün bulmanın NP-Hard bir problem olduğunu ispatlamıştır. Bu nedenle araştırmacılar en iyi çözümü bulmak yerine iyi olabilecek çözümler bulmaya yoğunlaşmıştır.

acd, ad, bcd

(a) Hassas öge kümeleri

Tid	Önce	Sonra
1	*abc**d**e*	*abce*
2	***a**cd*	*cd*
3	*abdfg*	*abdfg*
4	*bcde*	*bcde*
5	*ab**d***	*ab*
6	*bcdfh*	*bcdfh*
7	*abcg*	*abcg*
8	*ac**d**e*	*ace*
9	*acdh*	*acdh*

(b) Veritabanınnın arındırmadan önceki ve sonraki hali

Şekil 1: (a) Örnek veritabanından gizlenecek hassas öge kümeleri, ve (b) $\psi = 3$ hassasiyet değerine göre; veritabanının arındırmadan önce ve sonraki hali.

Şekil 1'de [4]'den alınmış örnek bir veritabanı ve hassas öge kümeleri verilmiştir. Hassas öge kümelerinin destek değerleri hassasiyet eşiğinden ($\psi = 3$) büyüktür. Tablodan görülebileceği üzere, kalın harflerle gösterilen ögeler silinmek suretiyle hassas bilgi gizleme işlemi yapılmıştır. Gizleme sonucunda, önceden hassas olan öge kümeleri artık $\psi = 3$ hassasiyet eşiğinden daha büyük bir destek değerine sahip değildir.

2.2. Dizgi Gizleme

$\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ şeklinde boş olmayan bir öge kümesi olsun. Bir $\mathcal{U}$ dizgisi, $\mathcal{I}$ kümesinin sıralanmış bir şeklidir. Aynı öge kümesinin farklı sıralanışları, farklı dizgileri gösterir. Bir $\mathcal{D}$ veritabanı, $\mathcal{D} = \{t_1, t_2, \ldots, t_m\}$ şeklinde gösterilir ve her t_i bir dizgi olmak üzere bir dizgiler kolleksiyonudur. Tüm dizgiler kümesi Σ^* ile gösterilir. $\mathcal{U} \in \Sigma^*$ ve $\mathcal{V} \in \Sigma^*$ birer dizgi olmak üzere, eğer $\mathcal{V}$'den bazı semboller silinerek $\mathcal{U}$ elde edilebiliyorsa, $\mathcal{U}$, $\mathcal{V}$'nin bir alt dizgisidir ve bu $\mathcal{U} \sqsubseteq \mathcal{V}$ ile ifade edilir.

Tanım 4 (Dizginin Destek Değeri) *$\mathcal{U}$ diye bir dizgi veriliyor olsun, $\mathcal{U}$'in destek değeri $\mathcal{D}$ veritabanındaki dizgilerden $\mathcal{U}$ dizgisini alt dizgi olarak içeren dizgilerin sayısıdır, $sup_{\mathcal{D}}(\mathcal{U}) = \mid \{T : \mathcal{U} \sqsubseteq T \wedge T \in \mathcal{D}\} \mid$.*

Tanım 5 (Sık Dizgi Madenciliği[5]) *σ negatif olmayan, kullanıcı tanımlı bir tamsayı olsun ve* destek eşiği *olarak belirtilsin. Bir $\mathcal{D}$ veritabanında, destek değeri σ'dan küçük olmayan dizgilere,* sık *dizgiler denir. Tüm düzgelerin kümesi şu şekilde ifade edilir: $F_{(\mathcal{D},\sigma)} = \{X : X \subseteq \Sigma^* \wedge sup_D(X) \geq \sigma\}$.*

Tanım 6 (Hassas Dizgi Gizleme) $\mathcal{S}_h = \{S_1, S_2, \ldots, S_n\}$, *$n$ tane dizgiden oluşan bir sık dizgiler kümesi olsun ve hassas addedilsin.* Hassas dizgi gizleme *problemi, verilen bir hassasiyet eşiği, ψ'nden yararlanılarak, $\mathcal{D}$ veritabanını bir*

dönüşümden geçirerek $\mathcal{D}'$'yi elde etmektir. Öyleki, dönüşüm sonucunda şunlar sağlanmış olmalıdır:

- $\forall S_i \in \mathcal{S}_h : sup_{\mathcal{D}'}(S_i) < \psi$.
- $\sum_{X \in \Sigma^* \setminus \mathcal{S}_h} \mid sup_{\mathcal{D}}(S) - sup_{\mathcal{D}'}(S) \mid$ *minimum olmalıdır.*

Dönüşüm işleminin birinci maddesi dönüştürülmüş veritabanında hassas dizgilerinin destek değerlerinin belirtilmiş eşik değerinden küçük olmasını; ikinci maddesi ise hassas olmayan dizgilerin destek değerlerinin mümkün olduğunca korunmasını gerektirir. Literatürde dönüştürme işlemi, dizgileri oluşturan ögeleri, silmek değil de, ögelerin sembollerini aldıkları kümeden olmayan bir sembol ile ,bir bilinmeyen ile, değiştirmek biçiminde olmaktadır. Öge kümelerinde olduğu gibi dizgilerde de hassas bilgiyi gizleyecek en iyi çözümü bulma problemi NP-Hard bir problemdir[6].

Şekil 2'de örnek bir veritabanı ve hassas dizgiler verilmiştir. Hassas dizgilerden, $\langle ABA \rangle$, $\langle BED \rangle$ ve $\langle CA \rangle$'nın arındırılmamış veritabanındaki destek değerleri sırasıyla 4,3 ve 4 tür. $\psi = 3$ hassasiyet değerine göre bir arındırma işlemi yapılırsa, tablodaki gibi arındırılmış bir varitabanı elde edilebilir. Arındırmada koyu sembollerle gösterilen ögeler **?** ile değiştirişmiştir. Arındırma sonucunda, önceden hassas olan dizgiler $\psi = 3$ hassasiyet eşiğine göre artık hassas değildir.

ABA , BED , CA

(a) Hassas dizgiler

Tid	Önce	Sonra
1	*A**B**EAD*	*A?EAD*
2	*ABCD**A***	*ABCD?*
3	*ED*	*ED*
4	***C**EADA*	*?EADA*
5	*BECD*	*BECD*
6	*ACBA*	*ACBA*
7	*BCEAD*	*BCEAD*
8	*ABBA*	*ABBA*

(b) Veritabanınnın arındırmadan önceki ve sonraki hali

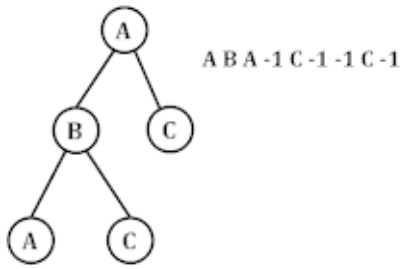
Şekil 2: (a) Örnek veritabanından gizlenecek hassas dizgiler ve (b) $\psi = 3$ hassasiyet değerine göre veritabanının arındırmadan önce ve sonraki hali.

2.3. Ağaç Gizleme

Tanım 7 (Ağaç) *Bir $\mathcal{T}$ ağacı boş olmayan ve sonlu sayıda düğüm içeren bir düğümler kümesidir, $\mathcal{T} = \{r\} \cup T_1, T_2, \ldots, T_n$ ve şu özellikleri sağlamalıdır:*

- *Belirlenmiş bir r düğümü ağacın köküdür.*
- *Diğer düğümler $n \geq 0$ kadar farklı alt kümeye bölünür, her $T_1, T_2, \ldots, T_n$ de birer ağaçtır. $T_1, T_2, \ldots, T_n$ ağaçları $\mathcal{T}$'nin alt ağaçlarıdır.*

Şekil 3'te örnek bir ağaç gösterilmiştir. Her düğüm *V* kümesinden bir etiketle etiketlenmiştir. Örnek ağaçtan da görülebileceği gibi ağaçlar etiket dizgisi olarak gösterilebilirler. Bu gösterim, kökten başlanarak ve ağaç ön sıralı gezilerek düğüm etiketlerinin bir dizgi şeklinde yazılmasıyla olur; gezme esnasında çocuk düğümden ebeveyn düğüme gidilirken *V*'den olmayan bir karakter, örneğin -1, eklenir.



Şekil 3: Örnek bir ağaç

Tanım 8 (Alt Ağaç) *Bir $\mathcal{T}$ ağacından bazı düğümlerin silinmesiyle oluşabilecek ağaçlara alt ağaç denir. Düğüm silme işlemi sonucunda, silinen düğümlerin çocukları artık silinen düğümün ebeveyninin çocuklardır. Eğer alt ağaçlar ebeveyn-çocuk ilişkisi gözetilerek oluşturuluyorsa, oluşan ağaç* birebir *alt ağaç; eğer sadece ata-torun ilişkisi korunuyorsa oluşan alt ağaç* gömülü *alt ağaç olarak adlandırılırlar.*

(A B -1 C), (B A -1 C), (A B C) Şekil 3'teki ağaçtan elde edilebilecek birebir alt ağaçlara; (A A -1 C), (A C -1 C) ise gömülü alt ağaçlara örnek olarak gösterilebilir.

Tanım 9 (Ağaçlarda Destek Değeri) *D diye ağaçlardan oluşan bir ağaçlar veritabanı olsun, X ise bir alt ağaç olsun. X'in destek değeri, X'i içeren $T_i \in D$'lerin sayısı kadardır. Formal olarak şu şekilde gösterilir: $sup_{\mathcal{D}}(X) = \mid \{T : X \dashv T \wedge T \in \mathcal{D}\} \mid$.*

Tanım 9'daki $\dashv$ simgesi, solundaki ağacın sağındaki ağacın bir alt ağacı olduğunu göstermektedir. Alt ağaçlar birebir veya gömülü olabileceği gibi sıralı ya da sırasız da olabilir. Alt ağaçlar elde etmek için düğümler silinirken geriye kalan ve birbirleriyle kardeş olan düğümlerin sırası değiştirilmiyorsa oluşan alt ağaçlar sıralı alt ağaçlar; eğer kardşlerin sırası korunmuyor ise oluşan ağaçlar sırasız alt ağaçlardır. Buna göre alt ağaçlar birebir-sıralı, birebir-sırasız, gömülü-sıralı ve gömülü-sırasız olmak üzere dört farklı biçimde olabilirler.

Tanım 10 (Alt Ağaç Madenciliği[7]) *σ negatif olmayan, kullanıcı tanımlı bir tamsayı olsun ve* destek eşiği *olarak belirtilsin. Σ^*, $\mathcal{D}$ veritabanındaki ağaçlardan elde edilebilecek tüm ağaçların kümesi olsun. $\mathcal{D}$ veritabanında, destek değeri σ'dan küçük olmayan alt ağaçlara,* sık *alt ağaçlar denir ve şu şekilde ifade edilir: $F_{(\mathcal{D},\sigma)} = \{T : T \subseteq \Sigma^*, sup_D(T) \geq \sigma\}$.*

Alt ağaç madenciliği yukarıda bahsedilen dört farklı alt ağaç biçimine göre yapılır. Birebir alt ağaç madenciliği ebeveyn-çocuk ilişkisini koruduğu için gömülü alt ağaç madenciliğine göre; sıralı alt ağaç madenciliği de sırayı korumak gerektiğinden sırasız alt ağaç madenciliğine göre daha kısıtlayıcıdır. Dolayısıyla, birebir alt ağaç madenciliğinde ortaya çıkan alt ağaçların sayısı gömülü alt ağaç madenciliğinde ortaya çıkan alt ağaçların sayısından; sıralı alt ağaç madenciliğinde ortaya çıkan alt ağaçların sayısı ise sırasız alt ağaç madenciliğinde ortaya çıkan alt ağaçların sayısından daha azdır.

Tanım 11 (Hassas Alt Ağaç Gizleme) *$\mathcal{T}_h = \{T_1, T_2, \ldots, T_n\}$, n tane alt ağaçtan oluşan bir sık alt*

ağaçlar kümesi olsun ve hassas addedilsin. Hassas alt ağaç gizleme *problemi, verilen bir hassasiyet eşiği, ψ'nden yararlanılarak, $\mathcal{D}$ veritabanını bir dönüşümden geçirip $\mathcal{D}'$'yi elde etmektir. Öyleki, dönüşüm sonucunda şunlar sağlanmış olmalıdır:*

- $\forall T_i \in \mathcal{T}_h : sup_{\mathcal{D}'}(T_i) < \psi$.
- $\sum_{X \in \Sigma^* \setminus \mathcal{T}_h} \mid sup_{\mathcal{D}}(X) - sup_{\mathcal{D}'}(X) \mid$ *minimum olmalıdır.*

Dönüşümün ilk maddesi hassas alt ağaçlarının tümünün, dönüşüm sonucunda verilen eşikten daha az destek değerlerine sahip olmasını, ikinci maddesi ise hassas olmayan alt ağaçların destek değerlerinin mümkün olduğunca korunmasını gerektirir. Hassas alt ağaçların gizlenmesi, veritabanındaki ağaçların hassas alt ağaçlarını içerip içermediğinin bulunması; eğer içeriyorsa da hassas alt ağaçlarının köklerinin eşlendiği düğümlerin etiketlerinin, ağacın etiketlerini aldığı alfabeden olmayan bir etiketle değiştirilmesiyle olur.

A A -1 B -1 , B A -1 B -1

(a) Hassas alt ağaçlar

Tid	Önce	Sonra
1	***B A*** *A -1 B -1 -1 B -1*	*B * A -1 B -1 -1 B -1*
2	***A*** *A -1 A -1 B -1*	***A*** *A -1 A -1 B -1*
3	***B*** *A -1 C -1 B -1*	***B*** *A -1 C -1 B -1*
4	***A*** *A -1* ***B*** *A -1 B -1 -1*	*A A -1 * A -1 B -1 -1*

(b) Veritabanınnın arındırmadan önceki ve sonraki hali

Şekil 4: (a) Örnek veritabanından gizlenecek hassas alt ağaçlar ve (b) $\psi = 2$ hassasiyet değerine göre; veritabanının arındırmadan önce ve sonraki hali.

Şekil 4'te örnek bir ağaç veritabanı yer almaktadır. Ağaçlar dizi halinde gösterilmişlerdir. Ayrıca destek değerleri 3 olan iki tane de hassas alt ağaç verilmiştir. Koyu olan düğümler, o düğümlerin içerdiği hassas ağaçların köküyle eşlendiğini göstermektedir. Dönüştürülmüş veritabanında ise * ile gösterilen düğümler gizlenen düğümleri belirtmektedir. Burada dikkat edilmesi gereken, 1 ve 4. ağaçların her iki hassas alt ağacı da içerdiğidir. Fakat bu iki ağaçtan sadece bir düğümün gizlenmiş olması her iki hassas alt ağacın gizlenmesi için yeterli olmuştur.

3. Uygulama

Öge kümesi, dizgi, ağaç ve çizge tipi verilerden oluşan veritabanlarından hassas bilgi gizlemek için geliştirmiz olduğumuz uygulamamız[8] Java programlama dili ile geliştirildi. Uygulamada her farklı tip veri gizleme problemi için farklı sekmeler tasarlandı. Her sekmede, gizleme işlemi için gizleme yapılacak veritabanı dosyasının yolu, gizlenecek bilgilerin dosya yolu, eşik değeri, kullanılacak gizleme algoritması ve varsa probleme özgü diğer gerekli girdiler seçilmekte ve uygulama çalıştırılmaktadır. Uygulamanın çalışması başarıyla sonlandıktan sonra, sonuç veritabanı ayrıca kaydedilmekte; farklı şekilde tanımlanıp ölçülen, problemlerlere özgü verimlilik ve etkililik değerleri ekrana yansıtılmakta ve bir dosyada saklanmaktadır.

3.1. Kullanılan Algoritmlar

3.1.1. Öge Kümesi Gizleme

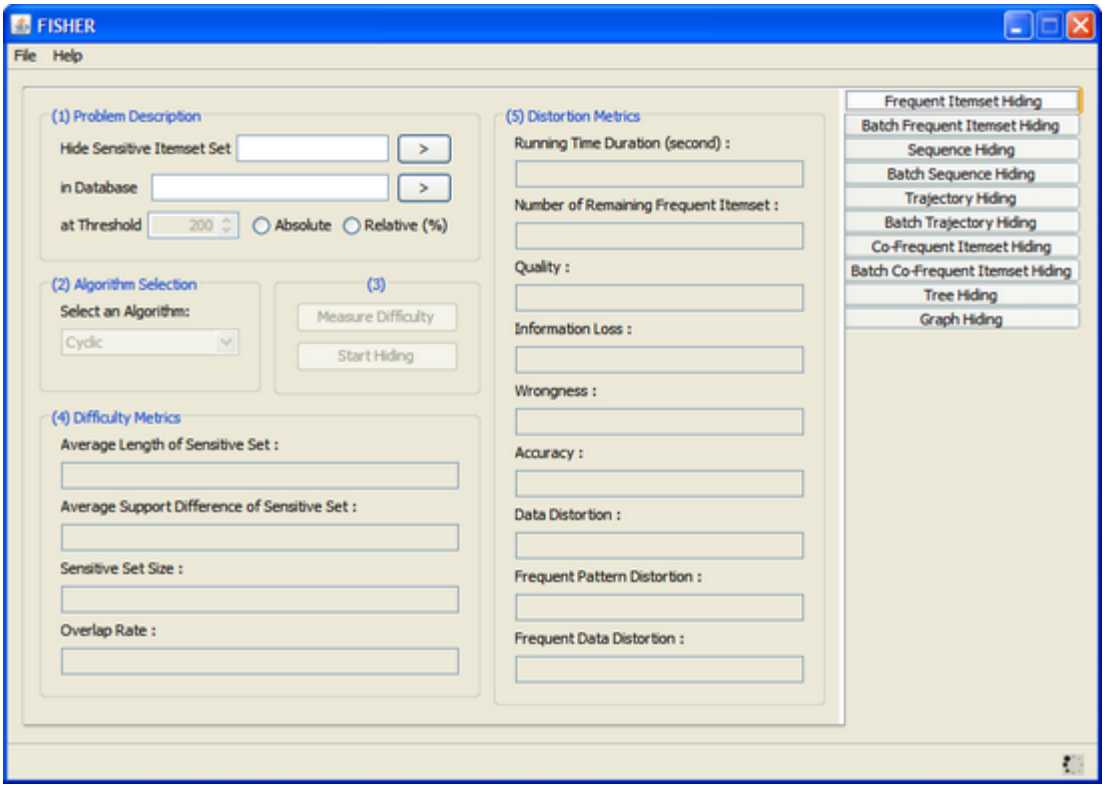
Sık öge kümlerinde hassas bilgi için uygulamamızda *Cyclic Hider*[3], *BorderBased Algorithm*[4], *FHSFI Hider*[9] ve ayrıca *Balanced Hider* isimli kendi geliştirdiğimiz algoritmayı kullandık. *Balanced Hider* dışındaki algoritmalara kısıtlı yer nedeniyle burada yer verilmemiştir, sadece kendi geliştirdiğimiz *Balanced Hider* algoritmasına değinilmektedir. *Balanced Hider* algoritmasındaki amaç, herhangi bir işlemde ($T \in \mathcal{D}$) herhangi bir öğenin değiştirilmesiyle birçok sık öge kümesinin destek değerini azaltmak ve etkili çalışmaktır. Bunun için $\mathcal{D}$ veritabanındaki tüm işlemler içerdikleri öge sayılarına göre artan sırada sıralanırlar. Bunun sebebi içerdiği öge sayısı az olan bir işlemin daha az sayıda sık öge kümesi içermesinin daha olası olmasıdır. Bu, o işlemde yapılacak bir silme işleminin hassas olmayan diğer sık öge kümelerinin destek değerlerini düşürmekten kaçınmayı sağlar. Algoritmada, her bir hassas öge kümesi için sıralanmış işlemler sırayla gezilir. Her işlem için ise hassas öge kümeleri sırayla gezilir. Gezilen işlem eğer o anki hassas öge kümesini içeriyorsa, o işlemin ayrıca diğer öge kümelerinden hangilerini içerdiği ve hassas öge kümesinin hangi ögesinin o anki işlemce içerilen diğer öge kümeleri içinde fazla sayıda bulunduğu tespit edilir. Tespit edilen bu öge silinerek hem o anki öge kümesinin destek değeri bir azaltılmış olur hem de diğer bazı hassas öge kümelerinin destek değerleri azaltılmış olabilir.

3.1.2. Dizgi Gizleme

Dizgi gizleme sekmesinde [6]'daki dizgi gizleme algoritmasına yer verilmiştir. Algoritmaya göre ilk önce eşleme kümeleri bulunmaktadır. Bir eşleme, bir hassas dizgiyi alt dizgi olarak içeren bir işlemin, hassas dizgi ile eşleşen ögelerin indekslerinnin sıralısıdır. Her bir işlemin eşleme kümeleri bulunur ve işlemler eşleme kümelerinin sayısına göre artan sayıda sıralanırlar. Hassas dizgilerin destek değerini, verilen hassasiyet eşiğinin altına düşmesini sağlayacak kadar işlem seçilir ve bu işlemdeki ilgili ögeler bir bilinmeyen karakterle değiştirilerek, o işlemin hassas dizgileri içermemesi sağlanır. Bu şekilde, en sonda, hassas dizgileri içeren işlemlerin sayısı verilen hassasiyet eşiğinden azdır, böylece gizleme işlemi başarılmış olur.

3.1.3. Ağaç Gizleme

Ağaçlardan hassas alt ağaç gizlemek için, gizlenecek alt ağacın ağaçlar içinde olup olmadığının tespit edilmesi gerekir. Uygulamamızda, dinamik programlama yöntemiyle ağaçların verilen alt ağacı içerip içermediğini test edip, içeriyorsa alt ağacın kökünün eşlendiği düğümü gizleyen bir algoritma kullandık. Bunun için veri ağacı ve alt ağaç ayrı ayrı sonsıralı gezilir; düğümlerin eşlenip eşlenmedikleri, eşleniyorlarsa, çocuklarından oluşan alt ağaçların eşlenme sayılarına bakılır. Eşlemenin sıralı ya da sırasız oluşuna göre çocuklarının eşlenme sayısı kombinasyon ya da permütasyonla hesaplanır ve ilgili düğümlerin kaç farklı şekilde eşlendikleri bir eşleme tablosuna sonradan okunmak üzere kaydedilir. Bu şekilde veri ağacının, alt ağaçı kaç farklı sayıda içerdiği tespit edilebilir. Alt ağacın kökünün eşlendiği veri ağacı düğümlerinin etiketi alfabeden olmayan, bir bilinmeyen karakter (*) ile değiştirilerek



Şekil 5: Uygulama Arayüzü. Uygulamada sık öge kümesi gizleme, dizgi gizleme, iz gizleme, ağaç gizleme ve çizge gizleme için sekmeler yer almaktadır.

o veri ağacının alt ağacı içermemesi sağlanır, bu da alt ağacın bir veri ağacından gizlenmesi demektir. Yeterince veri ağacında yapılan alt ağaç gizleme işlemi, o alt ağacın destek değerinin hassasiyet eşiğinden küçük olmasını sağlar, böylece alt ağaç gizleme işlemi başarılmış olur.

4. Sonuç

Kurumlar topladıkları veritabanlarını istatistiksel ve bilimsel araştırmalar için yayınlama eğilimindedirler. Fakat, yayınlanan veritabanları hassas bilgiler içerebileceğinden, veritabanlarının yayınlanmadan arındırılması gereklidir. Bu çalışmamızda literatürde ayrı ayrı bulunan bir takım veri gizleme yöntemlerini geliştirdiğimiz bazı yöntemlerle birleştirip oluşturduğumuz bütüncül uygulamamızı tanıttık. Uygulamaya, ağaç gizlemede daha etkin algoritmalar ve çizge tipi verilerde hassas bilgi gizlemeye yönelik algoritmalar geliştirilip eklenecektir.

5. Kaynaklar

[1] D. E. O'Leary, "Knowledge discovery as a threat to database security.," in *Knowledge Discovery in Databases* (G. Piatetsky-Shapiro and W. J. Frawley, eds.), pp. 507–516, AAAI/MIT Press, 1991.

[2] R. Agrawal, T. Imielienski, and A. Swami, "Mining association rules between sets of items in large databases," in *SIGMOD '93*, pp. 207–216, 1993.

[3] M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim, and V. S. Verykios, "Disclosure limitation of sensitive rules," in *KDEX'99*, pp. 45–52, 1999.

[4] X. Sun and P. S. Yu, "A border-based approach for hiding sensitive frequent itemsets," in *ICDM'05*, pp. 426–433, 2005.

[5] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Eleventh International Conference on Data Engineering (ICDE'95)*, (Taipei, Taiwan), pp. 3–14, 1995.

[6] O. Abul, M. Atzori, F. Bonchi, and F. Giannotti, "Hiding sequences," in *Third ICDE Int. Workshop on Privacy Data Management (PDM'07), in conjunction with ICDE'07*.

[7] M. J. Zaki, "Efficiently mining frequent trees in a forest: Algorithms and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 8, pp. 1021–1035, 2005.

[8] O. Abul, H. Gökçe, and Y. Sengez, "Frequent itemsets hiding: A performance evaluation framework," in *ISCIS*, pp. 668–673, 2009.

[9] C.-C. Weng, S.-T. Chen, and Y.-C. Chang, "A novel algorithm for hiding sensitive frequent itemsets," in *8th Int. Symposium on Advanced Intelligent Systems (ISIS'07)*, 2007.