



T.C

YILDIZ TEKNİK ÜNİVERSİTESİ
KİMYA METALURJİ FAKÜLTESİ
MATEMATİK MÜHENDİSLİĞİ

BİSİKLET PAYLAŞIM SİSTEMİ DATA ANALİZİ PROJE ÇALIŞMASI

FETHİ FIRAT TÜLÜ

13052068

VERİ MADENCİLİĞİNE GİRİŞ

Dr.Öğr.Üyesi Nilgün GÜLER BAYAZIT

ÖZET

Veri madenciliği mevcut veriyi kullanışlı bilgiye çevirme ihtiyacından ortaya çıkmış, büyük boyuttaki veri tabanlarından önceden bilinmeyen, gizli, anlamlı ve yararlı bilgilerin elde edilmesi süreci olup gelecek ile ilgili tahminde bulunmayı sağlar. Veri madenciliği sürecinin en önemli aşamaları ise verinin hazırlanması ve belirlenen amaca göre veri madenciliği algoritmalarının kullanımını içermektedir.

Bu çalışmada “Bisiklet Paylaşım Sistemi ve Hava Durumu” veri setinin içeriği, kullanılan özelliklerin anlamı, sınıf sayısı ve isimleri, her sınıfa düşen örnek sayısı gibi bilgilerin çıkarılması, veri seti içindeki özelliklerin sınıflamada ki ayırt ediciliklerine göre sıralanması ve eksik veri varsa giderilmesi yöntemlerinin araştırılması ayrıca Weka paket programı içinde yer alan Sınıflama / Kümeleme / Regresyon yöntemlerinden en başarılı olanların araştırılması, bir tanesinin detaylı incelenmesi, sınıflama başarımlarının ve sınıf karışıklık matrislerinin karşılaştırılması ayrıntılı olarak gösterilmektedir.

GİRİŞ

Bisiklet paylaşım sistemleri, üyelik, kiralama ve iade işlemlerinin tamamlandığı yeni nesil geleneksel bisiklet kiralamalarıdır. Bu sistemler sayesinde, kullanıcı belirli bir konumdan bisiklet kiralayabilir ve dönüş yapabilir. Şu anda dünya çapında 500'den fazla bisiklet paylaşım programı var. Programların sahip olduğu bisiklet sayısı ise 5000'den fazla. Bugün dünyamızda, bu sistemler çevresel ve sağlık sorunlarındaki trafikte önemli rolleri nedeniyle büyük ilgi görüyor.

Bisiklet paylaşım kiralama süreci, çevre ve mevsimsel ortamlarla oldukça ilişkilidir. Örneğin hava koşulları yağış, haftanın günü, mevsim, günün saati vb. kiralama davranışlarını etkileyebilir ayrıca sistemlerde seyahat, kalkış ve varış pozisyonu da açıkça kaydedilir. Şehirde hareketliliği algılamak için sanal bir ağ olduğu gözlemlenebilir. Bu veriler izlenerek şehirdeki olaylar tespit edilebilir.

Bu veri seti, 2011-2012 yılları arasında “Capital Bikeshare” sisteminde, ilgili hava ve mevsimsel bilgileri içeren saatlik ve günlük kiralık bisiklet sayısını içermektedir. Bu veriler 500 ayrı bisiklet paylaşım programı üzerinden Porto Üniversitesi Yapay Zekâ ve Karar Destek Laboratuvarı (LIAAD) sayesinde toplandı. Yapay Zekâ ve Karar Destek Laboratuvarı (LIAAD), verileri iki saatlik ve günlük olarak düzenledi ve daha sonra <http://www.freemeteo.com> adresinden elde edilen ilgili hava ve mevsimsel bilgileri çıkardı ve verilere ekledi ve son haline ulaştı.

Bu çalışmanın amacı bu veri setini ve Weka programını kullanarak istenilen ve öğrenilmesi beklenen bilgilerin raporunu sunmaktır.

VERİ SETİNİN İÇERİĞİ

Veri setini içeren klasör 3 adet dosya içermektedir.

- Readme.txt (Veri setinin bilgilerini içeren .txt uzantılı metin dosyası)
- day.csv (Günlük verileri içeren .csv uzantılı veri dosyası)
- hour.csv (Saatlik verileri içeren .csv uzantılı veri dosyası)

Görülebileceği üzere günlük ve saatlik olarak iki ayrı veri seti mevcut. Biz günlük olan day.csv'yi .arff uzantılı dosyaya dönüştürerek inceleyeceğiz.

ARFF-Viewer - C:\Users\Fethi Fırat\Desktop\Data Mining Homework\Bike-Sharing-Dataset (1)\day.arff

File Edit View

day.arff

Relation: day

No.	1: instant	2: dteday	3: season	4: yr	5: mnth	6: holiday	7: weekday	8: workingday	9: weathersit	10: temp	11: atemp	12: hum	13: windspeed	14: casual	15: registered	16: cnt
	Numeric	Nominal	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric
1	1.0	2011-...	1.0	0.0	1.0	0.0	6.0	0.0	2.0	0.344...	0.3636...	0.805...	0.160446	331.0	654.0	985.0
2	2.0	2011-...	1.0	0.0	1.0	0.0	0.0	0.0	2.0	0.363...	0.3537...	0.696...	0.248539	131.0	670.0	801.0
3	3.0	2011-...	1.0	0.0	1.0	0.0	1.0	1.0	1.0	0.196...	0.1894...	0.437...	0.248309	120.0	1229.0	1349.0
4	4.0	2011-...	1.0	0.0	1.0	0.0	2.0	1.0	1.0	0.2	0.2121...	0.590...	0.160296	108.0	1454.0	1562.0
5	5.0	2011-...	1.0	0.0	1.0	0.0	3.0	1.0	1.0	0.226...	0.22927	0.436...	0.1869	82.0	1518.0	1600.0
6	6.0	2011-...	1.0	0.0	1.0	0.0	4.0	1.0	1.0	0.204...	0.2332...	0.518...	0.089565	88.0	1518.0	1606.0
7	7.0	2011-...	1.0	0.0	1.0	0.0	5.0	1.0	2.0	0.196...	0.2088...	0.498...	0.168726	148.0	1362.0	1510.0
8	8.0	2011-...	1.0	0.0	1.0	0.0	6.0	0.0	2.0	0.165	0.1622...	0.535...	0.266804	68.0	891.0	959.0
9	9.0	2011-...	1.0	0.0	1.0	0.0	0.0	0.0	1.0	0.138...	0.1161...	0.434...	0.36195	54.0	768.0	822.0
10	10.0	2011-...	1.0	0.0	1.0	0.0	1.0	1.0	1.0	0.150...	0.1508...	0.482...	0.223267	41.0	1280.0	1321.0
11	11.0	2011-...	1.0	0.0	1.0	0.0	2.0	1.0	2.0	0.169...	0.1914...	0.686...	0.122132	43.0	1220.0	1263.0
12	12.0	2011-...	1.0	0.0	1.0	0.0	3.0	1.0	1.0	0.172...	0.1604...	0.599...	0.304627	25.0	1137.0	1162.0
13	13.0	2011-...	1.0	0.0	1.0	0.0	4.0	1.0	1.0	0.165	0.1508...	0.470...	0.301	38.0	1368.0	1406.0
14	14.0	2011-...	1.0	0.0	1.0	0.0	5.0	1.0	1.0	0.160...	0.1884...	0.537...	0.126548	54.0	1367.0	1421.0
15	15.0	2011-...	1.0	0.0	1.0	0.0	6.0	0.0	2.0	0.233...	0.2481...	0.498...	0.157963	222.0	1026.0	1248.0
16	16.0	2011-...	1.0	0.0	1.0	0.0	0.0	0.0	1.0	0.231...	0.2342...	0.483...	0.188433	251.0	953.0	1204.0
17	17.0	2011-...	1.0	0.0	1.0	1.0	1.0	0.0	2.0	0.175...	0.1767...	0.5375	0.194017	117.0	883.0	1000.0
18	18.0	2011-...	1.0	0.0	1.0	0.0	2.0	1.0	2.0	0.216...	0.2323...	0.861...	0.146775	9.0	674.0	683.0
19	19.0	2011-...	1.0	0.0	1.0	0.0	3.0	1.0	2.0	0.292...	0.2984...	0.741...	0.208317	78.0	1572.0	1650.0
20	20.0	2011-...	1.0	0.0	1.0	0.0	4.0	1.0	2.0	0.261...	0.25505	0.538...	0.195904	83.0	1844.0	1927.0
21	21.0	2011-...	1.0	0.0	1.0	0.0	5.0	1.0	1.0	0.1775	0.1578...	0.457...	0.353242	75.0	1468.0	1543.0
22	22.0	2011-...	1.0	0.0	1.0	0.0	6.0	0.0	1.0	0.059...	0.07907	0.4	0.17197	93.0	888.0	981.0
23	23.0	2011-...	1.0	0.0	1.0	0.0	0.0	0.0	1.0	0.096...	0.0988...	0.436...	0.2466	150.0	836.0	986.0
24	24.0	2011-...	1.0	0.0	1.0	0.0	1.0	1.0	1.0	0.097...	0.11793	0.491...	0.15833	86.0	1330.0	1416.0
25	25.0	2011-...	1.0	0.0	1.0	0.0	2.0	1.0	2.0	0.223...	0.2345...	0.616...	0.129796	186.0	1799.0	1985.0
26	26.0	2011-...	1.0	0.0	1.0	0.0	3.0	1.0	3.0	0.2175	0.2036	0.8625	0.29385	34.0	472.0	506.0
27	27.0	2011-...	1.0	0.0	1.0	0.0	4.0	1.0	1.0	0.195	0.2197	0.6875	0.113837	15.0	416.0	431.0
28	28.0	2011-...	1.0	0.0	1.0	0.0	5.0	1.0	2.0	0.203...	0.2233...	0.793...	0.1233	38.0	1129.0	1167.0
29	29.0	2011-...	1.0	0.0	1.0	0.0	6.0	0.0	1.0	0.196...	0.2121...	0.651...	0.145365	123.0	975.0	1098.0
30	30.0	2011-...	1.0	0.0	1.0	0.0	0.0	0.0	1.0	0.216...	0.2503...	0.722...	0.073983	140.0	956.0	1096.0
31	31.0	2011-...	1.0	0.0	1.0	0.0	1.0	1.0	2.0	0.180...	0.18625	0.603...	0.187192	42.0	1459.0	1501.0
32	32.0	2011-...	1.0	0.0	2.0	0.0	2.0	1.0	2.0	0.192...	0.23453	0.829...	0.053213	47.0	1313.0	1360.0
33	33.0	2011-...	1.0	0.0	2.0	0.0	3.0	1.0	2.0	0.26	0.2544...	0.775...	0.264308	72.0	1454.0	1526.0
34	34.0	2011-...	1.0	0.0	2.0	0.0	4.0	1.0	1.0	0.186...	0.1778...	0.437...	0.277752	61.0	1489.0	1550.0
35	35.0	2011-...	1.0	0.0	2.0	0.0	5.0	1.0	2.0	0.211...	0.2285...	0.585...	0.127839	88.0	1620.0	1708.0
36	36.0	2011-...	1.0	0.0	2.0	0.0	6.0	0.0	2.0	0.233...	0.2430...	0.929...	0.161079	100.0	905.0	1005.0
37	37.0	2011-...	1.0	0.0	2.0	0.0	0.0	0.0	1.0	0.285...	0.2916...	0.568...	0.1418	354.0	1269.0	1623.0
38	38.0	2011-...	1.0	0.0	2.0	0.0	1.0	1.0	1.0	0.271...	0.3036...	0.738...	0.045408	120.0	1592.0	1712.0
39	39.0	2011-...	1.0	0.0	2.0	0.0	2.0	1.0	1.0	0.220...	0.1982...	0.537...	0.36195	64.0	1466.0	1530.0
40	40.0	2011-...	1.0	0.0	2.0	0.0	3.0	1.0	2.0	0.134...	0.1442...	0.494...	0.188839	53.0	1552.0	1605.0
41	41.0	2011-...	1.0	0.0	2.0	0.0	4.0	1.0	1.0	0.144...	0.1495...	0.437...	0.221935	47.0	1491.0	1538.0
42	42.0	2011-...	1.0	0.0	2.0	0.0	5.0	1.0	1.0	0.189...	0.2135...	0.506...	0.10855	149.0	1597.0	1746.0
43	43.0	2011-...	1.0	0.0	2.0	0.0	6.0	0.0	1.0	0.2225	0.2329...	0.544...	0.203367	288.0	1184.0	1472.0
44	44.0	2011-...	1.0	0.0	2.0	0.0	0.0	0.0	1.0	0.316...	0.3241...	0.457...	0.260883	397.0	1192.0	1589.0
45	45.0	2011-...	1.0	0.0	2.0	0.0	1.0	1.0	1.0	0.415	0.39835	0.375...	0.417908	208.0	1705.0	1913.0
46	46.0	2011-...	1.0	0.0	2.0	0.0	2.0	1.0	1.0	0.266...	0.2542...	0.314...	0.291374	140.0	1675.0	1815.0
47	47.0	2011-...	1.0	0.0	2.0	0.0	3.0	1.0	1.0	0.318...	0.3162	0.423...	0.251791	218.0	1897.0	2115.0
48	48.0	2011-...	1.0	0.0	2.0	0.0	4.0	1.0	1.0	0.435...	0.4286...	0.505	0.230104	259.0	2216.0	2475.0
49	49.0	2011-...	1.0	0.0	2.0	0.0	5.0	1.0	1.0	0.521...	0.5119...	0.516...	0.264925	579.0	2348.0	2927.0
50	50.0	2011-...	1.0	0.0	2.0	0.0	6.0	0.0	1.0	0.399...	0.3914...	0.187...	0.507463	532.0	1103.0	1635.0
51	51.0	2011-...	1.0	0.0	2.0	0.0	0.0	0.0	1.0	0.285...	0.27733	0.407...	0.223235	639.0	1173.0	1812.0
52	52.0	2011-...	1.0	0.0	2.0	1.0	1.0	0.0	2.0	0.303...	0.2840...	0.605	0.307846	195.0	912.0	1107.0
53	53.0	2011-...	1.0	0.0	2.0	0.0	2.0	1.0	1.0	0.182...	0.1860...	0.577...	0.195683	74.0	1376.0	1450.0
54	54.0	2011-...	1.0	0.0	2.0	0.0	3.0	1.0	1.0	0.221...	0.2457...	0.423...	0.094113	139.0	1778.0	1917.0
55	55.0	2011-...	1.0	0.0	2.0	0.0	4.0	1.0	2.0	0.295...	0.2891...	0.697...	0.250496	100.0	1707.0	1807.0
56	56.0	2011-...	1.0	0.0	2.0	0.0	5.0	1.0	2.0	0.364...	0.3504...	0.712...	0.346539	120.0	1341.0	1461.0

Veri Setlerinin Değişkenleri (Özellikleri, Attributes);

hour.csv için;

Relation: hour – Attributes : 17 – Instances: 17379 – Sum of Weights: 17379

day.csv için;

Relation: hour – Attributes : 16* – Instances: 731 – Sum of Weights: 731

- **instant**: Kayıt Numarası (Type: Numerical – Missing: 0 – Distinct: 17379 – Unique: 17379)

- **dteday** : Gün (Type: Nominal – Missing: 0 – Distinct: 731 – Unique: 1)

- **season** : Mevsim (Type: Nominal – Missing: 0 – Distinct: 731 – Unique: 1)

(1: İlkbahar, 2: Yaz, 3: Sonbahar, 4: Kış)

- **yr** : Yıl (Type: Numeric – Missing: 0 – Distinct: 2 – Unique: 0)

(0: 2011, 1:2012)

- **mnth** : Ay (Type: Numeric – Missing: 0 – Distinct: 12 – Unique: 0)

(1'den 12'ye kadar)

- **hr** : Saat (Type: Numeric – Missing: 0 – Distinct: 24 – Unique: 0)*

(0'dan 23'e kadar)

- **holiday** : Gün tatil ya da değil. (Type: Numeric – Missing: 0 – Distinct: 2 – Unique: 0)

(kaynak: <http://dchr.dc.gov/page/holiday-schedule>)

- **weekday** : Haftanın günü. (Type: Numeric – Missing: 0 – Distinct: 7 – Unique: 0)

- **workingday** : Çalışma günü. (Type: Numeric – Missing: 0 – Distinct: 2 – Unique: 0)

(Eğer gün hafta sonu ya da tatil değil ise 1, öyle ise 0)

+ **weathersit** : Hava Durumu (Type: Numeric – Missing: 0 – Distinct: 4 – Unique: 0)

- 1: Clear, Few clouds, Partly cloudy, Partly cloudy

- 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist

- 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds

- 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

- **temp** : Sıcaklık (Type: Numeric – Missing: 0 – Distinct: 50 – Unique: 2)

(Celsius, değerler 41'e bölünür. (max))

- **atemp**: Hissedilen sıcaklık (Type: Numeric – Missing: 0 – Distinct: 65 – Unique: 2)

(Celsius. Değerler 50'ye bölünür. (max))

- **hum**: Nem (Type: Numeric – Missing: 0 – Distinct: 89 – Unique: 6)

(Değerler 100'e bölünür. (max))

- **windspeed**: Rüzgar hızı. (Type: Numeric – Missing: 0 – Distinct: 30 – Unique: 2)

(Değerler 67'ye bölünür. (max))

- **casual**: Normal kullanıcı sayısı. (Type: Numeric – Missing: 0 – Distinct: 322 – Unique: 44)

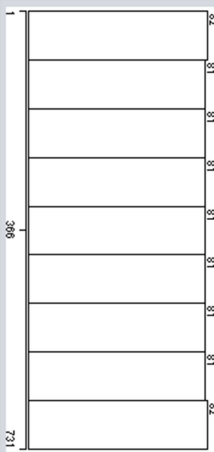
- **registered**: Kayıtlı kullanıcı sayısı. (Type: Numeric – Missing: 0 – Distinct: 776 – Unique: 91)

- **cnt**: Hem normal hem de kayıtlı olmak üzere toplam kiralık bisikletlerin sayısı.

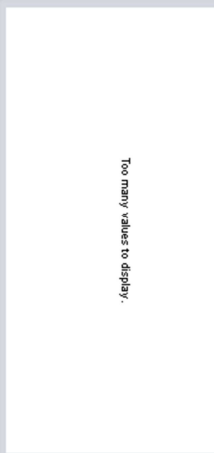
(Type: Numeric – Missing: 0 – Distinct: 869 – Unique: 91)

*hour.csv ve day.csv öğelerinin ikisi de aynı değişkenlere sahiptir. Yalnızca hour.csv de ek olarak hr değişkeni mevcuttur. hr değişkeni day.csv de bulunmaz.

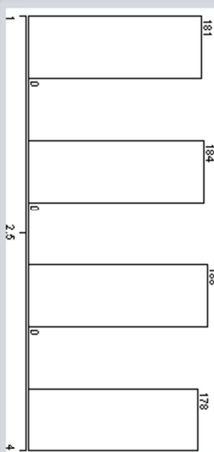
instant



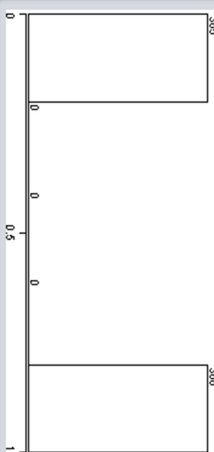
dayofweek



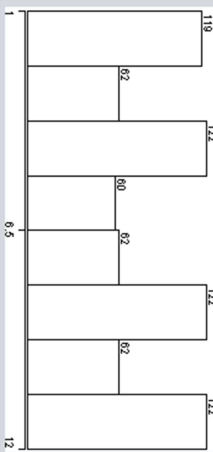
season



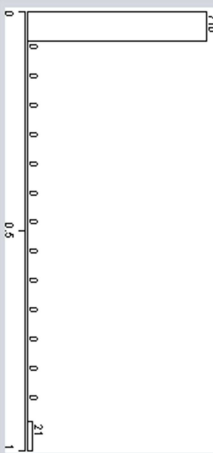
yr



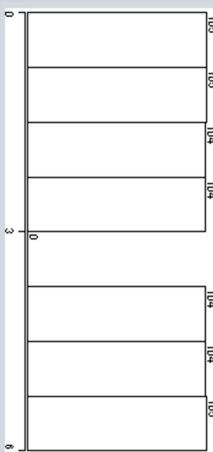
month



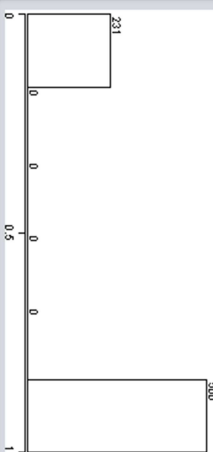
holiday



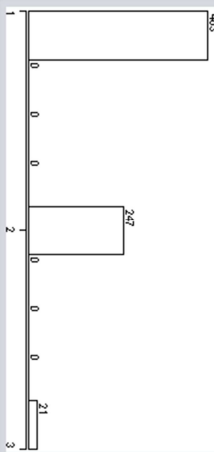
weekday



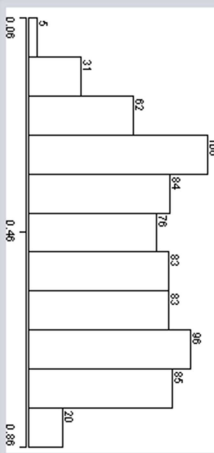
workingday



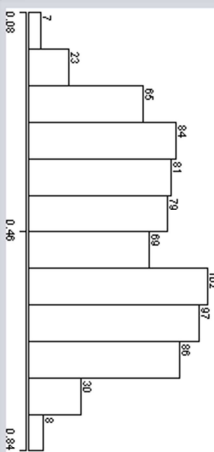
weather_sit



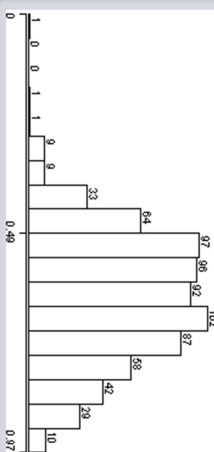
temp



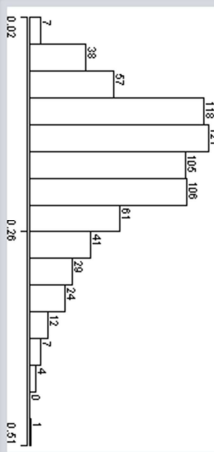
atemp



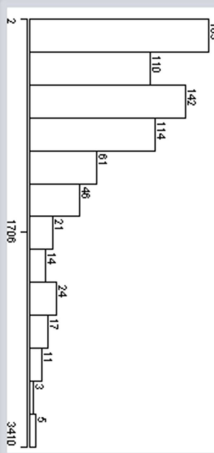
hum



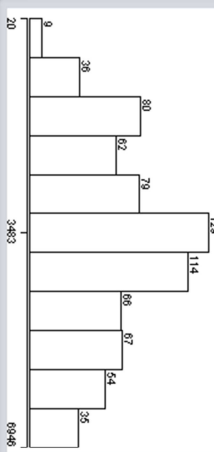
windspeed



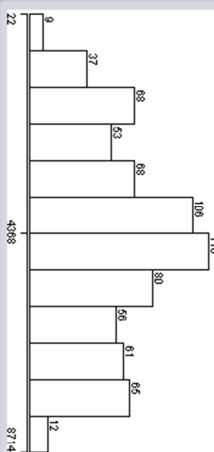
casual



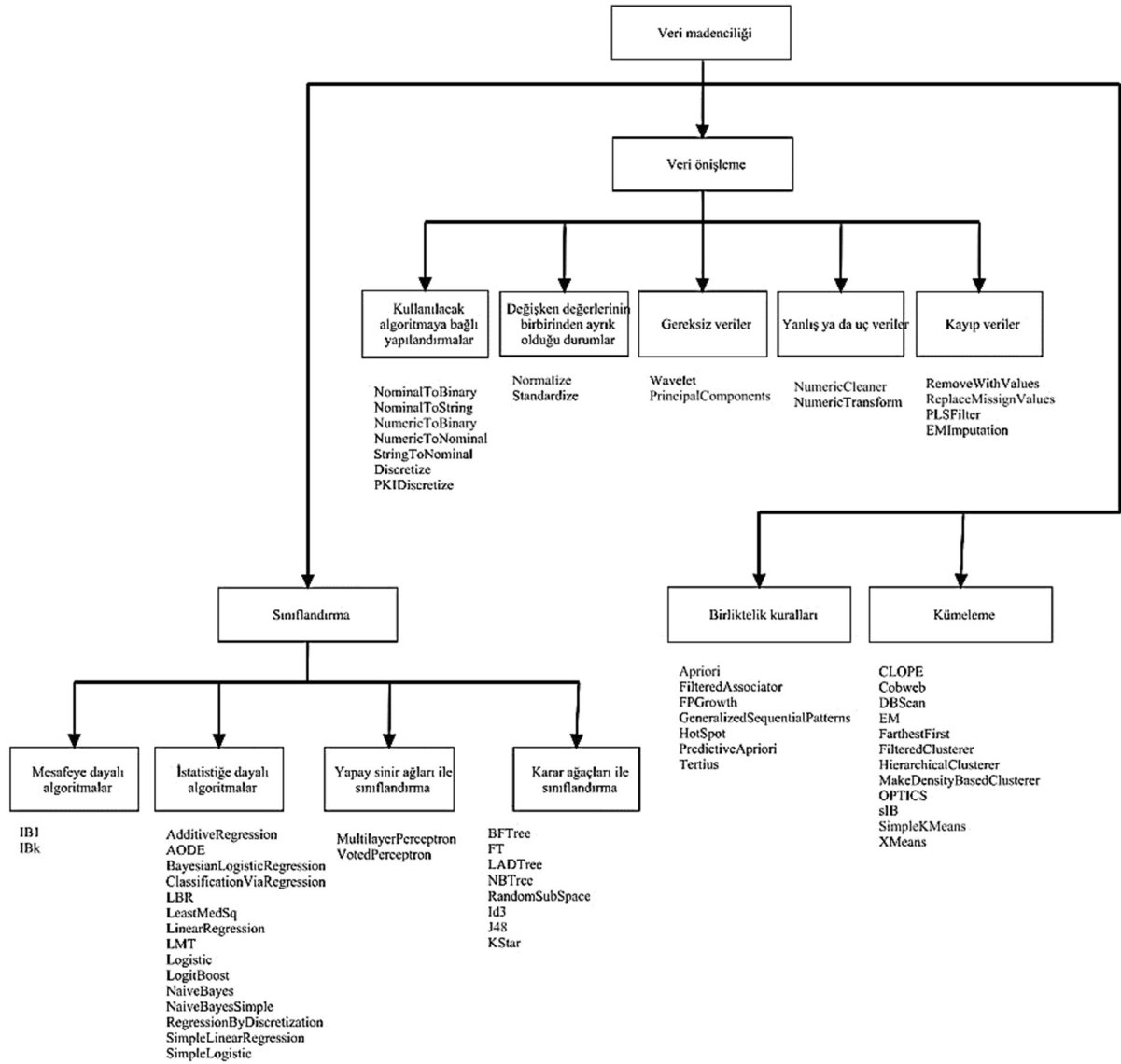
registered



cnt



WEKA PAKET PROGRAMININ HİYERARŞİK YAPISI



VERİ ÖNİŞLEME

Kayıp Veriler;

Kayıp verilerin yaratacağı sorunları ortadan kaldırmak için kullanılan yöntemlere örnek olarak Replace Missing Values modülü ele alınmıştır. Bu yöntemle veri tabanındaki kayıp değerler, ait oldukları niteliğin diğer değerlerinin ortalaması ya da moduyla değiştirilmektedir.

Yanlış ya da Aşırı Uç Veriler;

Bu tür veriler için ise Numeric Cleaner modülü ele alınmıştır. Bu yöntem çok büyük, çok küçük ya da belli bir değere çok yakın değerlerin veri tabanından silinerek bu değerlerin yerine önceden belirlenmiş başka bir değer atanmasını içerir.

Gereksiz Veriler;

Örneğin aynı veri tabanı içinde hem kayıt numarası hem de rüzgar hızı bilgisinin verilmesi durumunda oluşan gereksiz verilerin bilgisayar çalışma zamanını ve sonuçların kalitesini etkilememesi amacıyla Principal Components modülü kullanılarak veri boyutu azaltılır.

Değişken Değerlerinin Birbirinden Ayrık Olduğu Durumlar;

Değişken değerlerinin birbirinden ayrık olduğu durumlar için WEKA'nın Normalize modülünden faydalanılır. Bu yöntemle nümerik değerler birbirine yaklaştırılarak normalleştirilmektedir.

Kullanılacak Algoritmaya Bağlı Yapılandırmalar;

Veri önışlemeye gereksinim duyan diğere bir durum olan algoritmanın sebep olduđu kısıtlar için ise veri tabanındaki nümerik değerlerin 0-1 değerlere dönüştürülmesi esasına dayanan NumericToBinary modülü kullanılır.

Bize verilen veri setinin şanslıyız ki bu önışleme işlemlerinin hiçbirini kullanmadık. Çünkü veri setinde kayıp (missing) veri, yanlış ya da aşırı uç veriler, gereksiz veriler, değışkelerin birbirinden ayrık olduđu durumlar ya da diğere yapılandırmaların hiçbirine gerek yoktur. Dolayısıyla veri seti gayet düzgün ve sınıflandırılma aşamasına geçmeye hazırdır.

SINIFLANDIRMA

Algoritmalar	Doğru sınıflandırılan Örnek	Korelasyon Katsayısı	Ortalama Mutlak Hata	Ortalama Hata Karekök	Görelî Mutlak Hata %	Görelî Hata Karekök %
GaussianProcesses	731	0,949	486,5526	623,3823	30,7373	32,1781
LinearRegression	731	0,5804	1384,3005	1729,9891	87,4512	89,2994
SMOreg	731	0,9989	74,6493	100,1065	4,7159	5,1673
IBk	731	0,9299	498,7415	723,3457	31,5073	37,338
Kstar	731	0,9872	227,7777	309,9778	14,3895	16,0006
LWL	731	0,8525	845,0174	1022,7757	53,3828	52,7941
AdditiveRegression	731	0,9649	425,8111	516,4437	26,9	26,658
Bagging	731	-0,0309	1582,0413	1936,687	99,9432	99,9689
CVParameterSelection	731	-0,0787	1582,9401	1937,2902	100	100
MultiScheme	731	-0,0787	1582,9401	1937,2902	100	100
RandomCommittee	731	0,9432	737,6976	934,2476	46,603	48,2245
RandomizableFilteredClassifier	731	0,8996	585,725	859,9494	37,0024	44,3893
RandomSubSpace	731	0,9197	987,2253	1249,6582	62,3666	64,5055
RegressionByDiscretization	731	0,986	252,4994	322,5602	15,9513	16,6501
Stacking	731	-0,0787	1582,9401	1937,2902	100	100
Vote	731	-0,0787	1582,9401	1937,2902	100	100
WeightInstancesHandlerWrapper	731	-0,0787	1582,9401	1937,2902	100	100
InputMappedClassifier	731	-0,0787	1582,9401	1937,2902	100	100
DecisionTable	731	0,984	208,5049	345,4564	13,172	17,8319
M5Rules	731	-0,0195	1582,2911	1939,5994	99,959	100,1192
ZeroR	731	-0,0787	1582,9401	1937,2902	100	100
DecisionStump	731	0,7448	1108,0987	1293,1265	70,0026	66,7492
M5P	731	-0,0195	1582,2911	1939,5994	99,959	100,1192
RandomForest	731	0,9692	739,382	937,6852	46,7094	48,4019
RandomTree	731	0,8685	719,074	960,8062	45,4265	49,5954
REPTree	731	-0,0787	1582,9401	1937,2902	100	100

(Tablo:1)

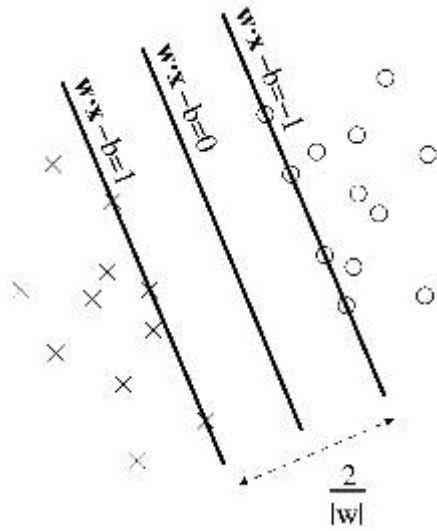
Tablo 1'deki sonuçlar WEKA programı ile elde edilmiş sonuçlardır. Yapılan uygulama çalışmasında sınıflandırma analizine ait pek çok algoritma denenmiş, hepsinin dereceleri tabloda verilmiştir. Tüm bu algoritmalar denenirken WEKA programında "Test Options" bölümünde Cross – Validation Folds 10 seçeneği seçildi. Yani veri seti 10 parçaya bölünür, 9 tanesi eğitim 1 tanesi test için kullandık. Bizim değişken tiplerimiz numeric olduğu için Kappa Statistics, Confusion Matrix gibi sonuçlar elde edilemiyor. Bazı algoritmaların oranları eşit çıkmasına karşın tabloda yer verdim.

Her algoritma da doğru sınıflandırılan örnek eşit. O nedenle Korelasyon Katsayısı en yüksek olan SMOreg algoritmasını seçerek incelemeye karar verdim.

SUPPORT VECTOR MACHINE (SVM) İLE SINIFLANDIRMA

Sınıflandırma (Classification) konusunda kullanılan oldukça etkili ve basit yöntemlerden birisidir. Sınıflandırma için bir düzlemde bulunan iki grup arasında bir sınır çizilerek iki grubu ayırmak mümkündür. Bu sınırın çizileceği yer ise iki grubun da üyelerine en uzak olan yer olmalıdır. İşte SVM bu sınırın nasıl çizileceğini belirler.

Bu işlemin yapılması için iki gruba da yakın ve birbirine paralel iki sınır çizgisi çizilir ve bu sınır çizgileri birbirine yaklaştırılarak ortak sınır çizgisi üretilir. Örneğin aşağıdaki şekildeki iki grubu ele alalım:



Bu şekilde iki grup iki boyutlu bir düzlem üzerinde gösterilmiştir. Bu düzlemi ve boyutları birer özellik olarak düşünmek mümkündür. Yani basit anlamda sisteme giren her girdinin (input) bir özellik çıkarımı (feature extraction) yapılmış ve sonuçta bu iki boyutlu düzlemde her girdiyi gösteren farklı bir nokta elde edilmiştir. Bu noktaların sınıflandırılması demek, çıkarılmış olan özelliklere göre girdilerin sınıflanması demektir.

Yukarıda her iki sınıf arasında oluşan aralığa tolerans (offset) demek mümkündür. Bu düzlemdeki her bir noktanın tanımı aşağıdaki gösterim ile yapılabilir:

$$\mathcal{D} = \{(\mathbf{x}_i, c_i) \mid \mathbf{x}_i \in \mathbb{R}^p, c_i \in \{-1, 1\}\}_{i=1}^n$$

Yukarıdaki gösterimi şu şekilde okumak mümkündür. Her x, c ikilisi için X vektör uzayımızdaki bir nokta ve c ise bu noktanın -1 veya $+1$ olduğunu gösteren değeridir. Bu noktalar kümesi $i=1$ 'den n 'e kadar gitmektedir.

Yani bu gösterim bir önceki şekilde olan noktaları ifade etmektedir.

Bu gösterimin bir aşırı düzlem (hyperplane) üzerinde olduğunu düşünersek. Bu gösterimdeki her noktanın:

$$w\mathbf{x} - b = 0$$

denklemleri ile ifade edilebilir. Buradaki w aşırıdüzleme dik olan normal vektörü ve x noktanın değişen parametresi ve b ise kayma oranıdır. Bu denklemleri klasik $ax+b$ doğru denklemlerine benzetmek mümkündür.

Yine yukarıdaki denklemlere göre $b/\|w\|$ değeri bize iki grup arasındaki mesafe farkını verir. Bu mesafe farkına daha önce tolerans (offset) ismini de vermiştik. Bu mesafe farkı denklemlere göre mesafeyi en yüksek değere çıkarmak için yukarıdaki ilk şekilde gösterilen 0, -1 ve +1 değerlerine sahip 3 doğruyu veren denklemlerde $2/\|w\|$ formülü kullanılmıştır. Yani doğrular arası mesafe 2 birim olarak belirlenmiştir.

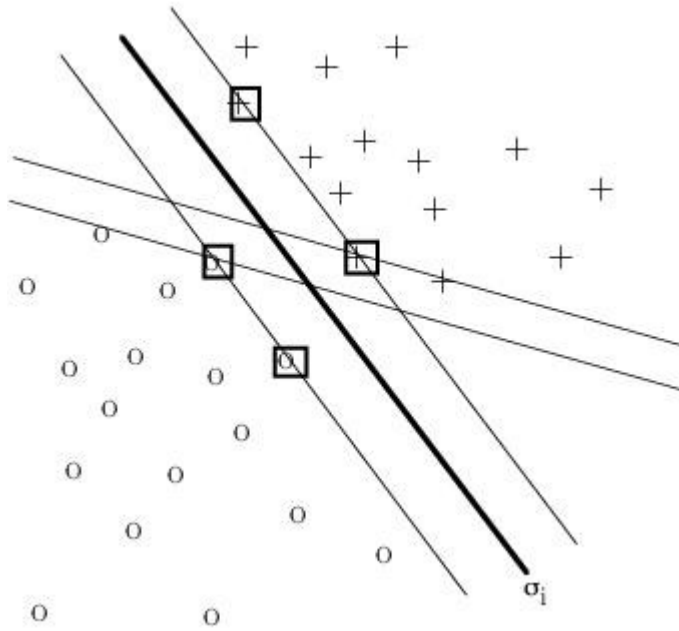
Bu denklemlere göre elde edilen iki doğru denklemi:

$$wx - b = -1$$

$$wx + b = 1$$

olarak bulunmuştur. Aslında bu denklemler doğruların kaydırılması sonucunda elde edilen en yüksek değerlerin bulunması işleminin bir sonucudur. Aynı zamanda bu denklemlerle problemin doğrusal ayrılabilir (linearly separable) olduğu da kabul edilmiş olur.

Tahmin edileceği üzere iki grup arasındaki aşırıdüzlemin (hyperplane) tek yönlü olması mümkün değildir. Aşağıda bu duruma bir örnek gösterilmiştir:



Yukarıdaki şekilde iki farklı hiperdüzlem (aşırı düzlem) olasılığı bulunmasına karşılık SVM yönteminde bu olasılıklardan en büyük toleransa (offset) sahip olanı alınır.

WEKA PAKET PROGRAMI ÇIKTISI

Number of kernel evaluations: 267546 (91.412% cached)

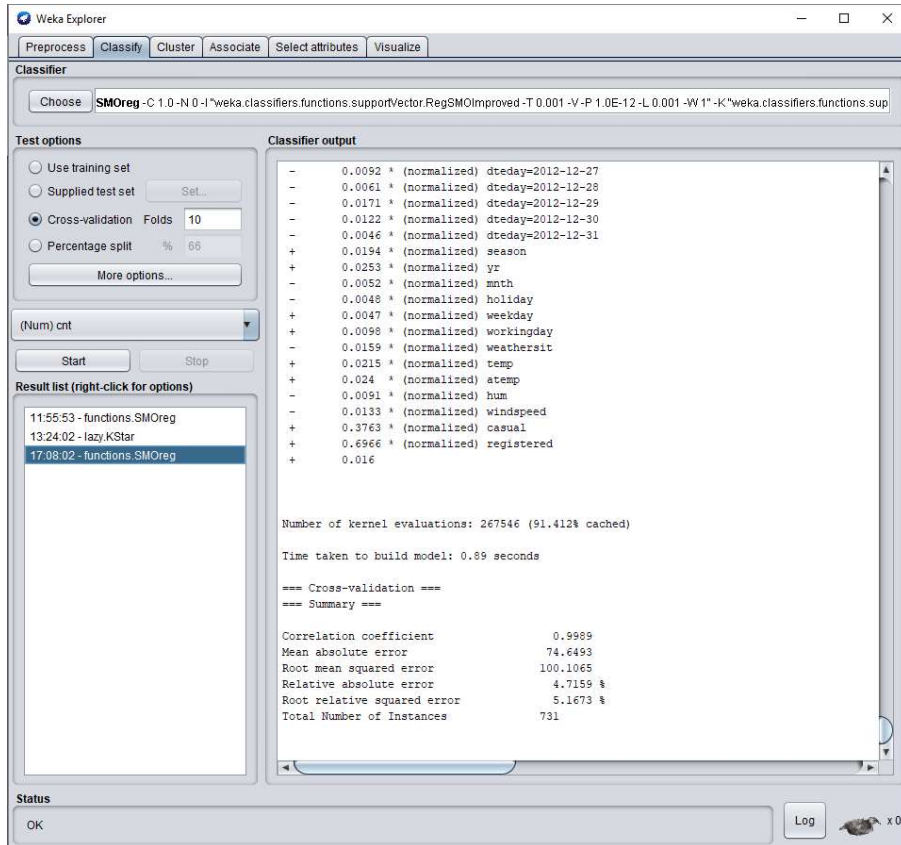
Time taken to build model: 1.28 seconds

=== Cross-validation ===

=== Summary ===

Correlation coefficient	0.9989
Mean absolute error	74.6493
Root mean squared error	100.1065
Relative absolute error	4.7159 %
Root relative squared error	5.1673 %
Total Number of Instances	731

Diğer tüm çıktıların yazılı halini [sınıflandırma.txt](#)'de bulabilirsiniz.



The screenshot shows the Weka Explorer interface. The 'Classifier' tab is active, displaying the 'SMOreg' classifier. The 'Test options' section shows 'Cross-validation' selected with 'Folds' set to 10. The 'Classifier output' pane displays a list of features with their normalized coefficients and a summary of the model's performance metrics, including the number of kernel evaluations, time taken to build the model, and the cross-validation summary.

```
Classifier
Choose SMOreg -C 1.0 -N 0 -I "weka.classifiers.functions.supportVector.RegSMOImproved -T 0.001 -V -P 1.0E-12 -L 0.001 -W 1" -K "weka.classifiers.functions.svm

Test options
Use training set
Supplied test set
Cross-validation Folds 10
Percentage split % 66
More options...

(Num) cnt
Start Stop

Result list (right-click for options)
11:55:53 - functions.SMOreg
13:24:02 - lazy.KStar
17:08:02 - functions.SMOreg

Classifier output
- 0.0092 * (normalized) gteday=2012-12-27
- 0.0061 * (normalized) gteday=2012-12-28
- 0.0171 * (normalized) gteday=2012-12-29
- 0.0122 * (normalized) gteday=2012-12-30
- 0.0046 * (normalized) gteday=2012-12-31
+ 0.0194 * (normalized) season
+ 0.0253 * (normalized) yr
- 0.0052 * (normalized) mnth
- 0.0048 * (normalized) holiday
+ 0.0047 * (normalized) weekday
+ 0.0098 * (normalized) workingday
- 0.0159 * (normalized) weathersit
+ 0.0215 * (normalized) temp
+ 0.024 * (normalized) atemp
- 0.0091 * (normalized) hum
- 0.0133 * (normalized) windspeed
+ 0.3763 * (normalized) casual
+ 0.6966 * (normalized) registered
+ 0.016

Number of kernel evaluations: 267546 (91.412% cached)

Time taken to build model: 0.89 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient      0.9989
Mean absolute error         74.6493
Root mean squared error     100.1065
Relative absolute error     4.7159 %
Root relative squared error  5.1673 %
Total Number of Instances   731
```

KÜMELEME

Kümeleme, verilerin birbirlerine olan benzerliklerine göre gruplandırılmasına dayanır. Sınıflandırmada olduğu gibi bir gruplandırma söz konusu olsa da sınıflandırmadan farklı olarak sınıflar önceden belli değildir. Kümeleme algoritmaları, küme oluşturma stratejisine ve kullanılan veri türüne göre farklılık gösterirler. Kümeleme yöntemlerinin çoğu veriler arasındaki uzaklıkları yani veriler arasındaki benzerlik ya da farklılıkları kullanırlar.

Hiyerarşik Algoritmalar

Veri tabanındaki her noktanın birleşiminden oluşan bir kümenin aşamalı olarak alt kümelere ayrılması (bölünür kümeleme algoritmaları) ya da veri tabanındaki her noktanın ayrı bir küme olarak ele alınarak bu kümelerin birleştirilmesiyle ayrı kümelere ulaşılması (toplaşım kümeleme algoritmaları) esasına dayanır. Kullanımı kolay ve hemen hemen tüm veri tiplerine uygulanabilen esnek bir yapıya sahiptir.

SLINK (En yakın komşu algoritması): Her bir verinin ayrı bir küme olarak ele alındığı ve aşamalı olarak bu kümelerin birleştirildiği bir yapıya sahiptir. Bu algoritmada iki kümenin birbirine olan uzaklığı, o kümelerdeki birbirine en yakın verilerin birbirine olan uzaklığı olarak kabul edilir. Eğer eldeki uzaklık verisi belli bir eşik değerini geçiyorsa kümeler birleştirilir.

CURE (Temsilciler kullanarak kümeleme): Veri tabanı içinde diğer verilerden uzakta bulunan ve sayıları az olup aslında hiçbir kümeye ait olmaması gereken uç verilerin kümeleme kalitesini etkilememesi amacıyla geliştirilmiş bir algoritmadır. En yakın komşu algoritmasındaki toplaşım ve yakınlık prensibine dayanır.

EN UZAK KOMŞU ALGORİTMASI: En yakın komşu algoritmasından farklı olarak iki kümenin birbirine olan uzaklığı, kümelerdeki birbirine en uzak verilerin arasındaki uzaklıkla belirlenir.

CHAMELEON: İki kümenin birbirine olan uzaklığının yanı sıra birbirine olan benzerlikleri bilgisini de kullanır. İki kümenin birleştirilmesi esnasında, kümelerin birbirine olan benzerliği ve yakınlığı ile bu kümelerin kendi iç benzerlikleri ve yakınlıkları karşılaştırılır. Böylece daha kaliteli ve homojen kümeler elde edilir. En yakın komşu algoritmasındaki toplaşım ve yakınlık prensibine dayanır.

BIRCH (Hiyerarşi kullanarak dengelenmiş iteratif azaltma ve kümeleme): Temel olarak gürültülü verilerin kontrol edilmesi amacıyla büyük boyutlu veri tabanlarının kümelenebilmesi için geliştirilmiştir. En uzak komşu algoritmasında olduğu gibi bölünür bir yapıya sahip olan algoritma, sadece sayısal verilere uygulanabilmektedir.

Bölümlemeli Algoritmalar

Kümeler arasındaki minimum ya da maksimum uzaklığın, kümelerin iç benzerlik kriterlerinin ve küme sayısının kullanıcı tarafından belirlendiği algoritmalarıdır. Hiyerarşik algoritmalarından daha hızlı çalışan bölümlemeli algoritmalar, bu özelliklerinden dolayı büyük veritabanlarının kümelenmesi için daha uygundur.

K-ORTALAMA ALGORİTMASI: Verilerin kümelerin ortalamalarına göre önceden belirlenmiş K adet kümeye ayrılması esasına dayanan bir algoritmadır. Toplam ortalama hatanın minimize edilmesini amaçlayan algoritma sadece sayısal verilerde kullanılabilir. Bu alandaki algoritmaların çoğu k-ortalama algoritmasının geliştirilmesiyle ortaya çıkmıştır.

K-MEDOID ALGORİTMASI: Sayısı önceden belirlenmiş K adet kümenin her biri için K adet medoid belirlenmesi ile başlayan algoritma veri tabanındaki diğer verilerin kendilerine en çok benzeyen medoidlerin etrafına toplanması esasına dayanır. Medoid ise kümenin merkezine yakın uzaklıkta bulunan noktayı temsil etmektedir.

CLARA ALGORİTMASI (Geniş uygulamaların kümelenmesi): K-MEDOID algoritmasından farklı olarak tüm veri tabanını tarayarak medoid noktaları belirlemek yerine veri tabanından rastgele oluşturulan bir örnek küme üzerinde benzer şekilde çalışır. İki algoritma karşılaştırıldığında CLARA algoritmasının büyük boyutlu veri tabanları için daha güvenli olduğu ve daha kısa süre içinde kümeleme yapabildiği belirtilmiştir.

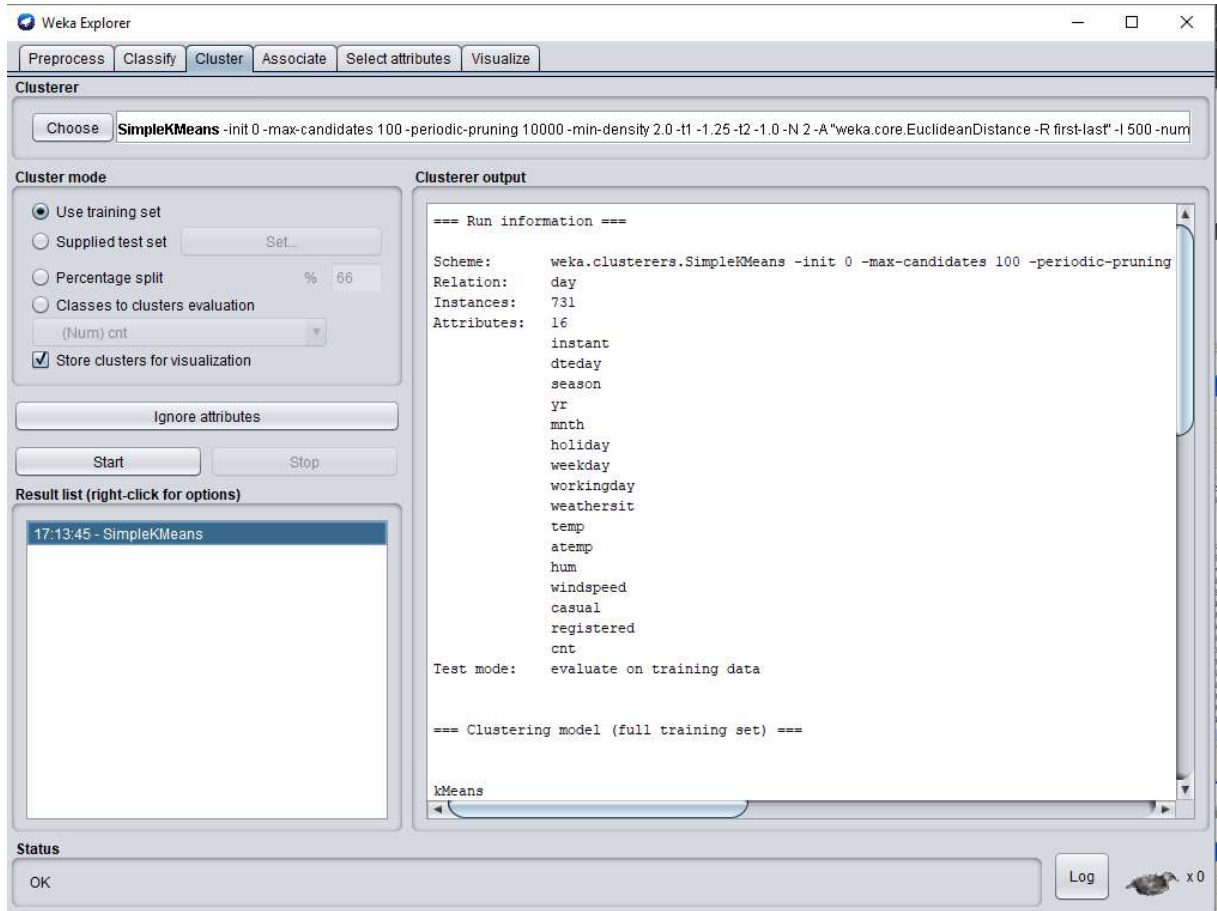
CLARANS ALGORİTMASI (Rastgele aramaya dayalı geniş uygulamaları kümeleme): k-medoid ve CLARA algoritmalarının geliştirilmiş bir halini barındıran CLARANS algoritması şebeke diyagramından yararlanan bir yapıya sahiptir. CLARA algoritmasına benzer olarak bütün veri tabanı taranmazken yapılan örnekleme dinamik bir yapıya sahiptir.

Yoğunluğa Dayalı Algoritmalar

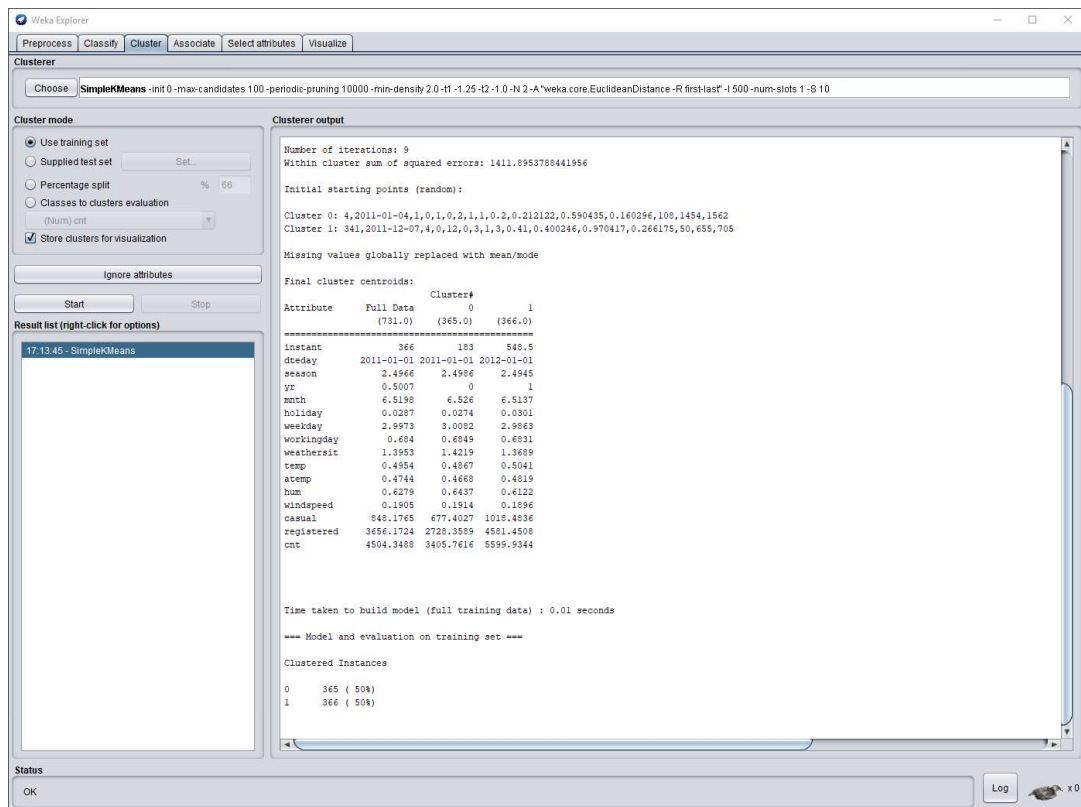
Dağılmış verilere sahip veri tabanlarının sadece uzaklığı temel alan bölümlemeli algoritmalar ile kümelenmesi oldukça güçtür. Çünkü hiçbir kümeye dâhil olmayan uç noktalar içeren bu dağılmış veri tabanlarının bölümlemeli algoritmalar ile kümelenmesi neticesinde doğru kümeler ortaya çıkmayacaktır. Bu durumda birlikte bir yoğunluk oluşturan verilerin aynı kümeye alınmasına dayanan yoğunluğa dayalı algoritmalar kullanılmalıdır. Bu tür algoritmalara örnek olarak DBSCAN, OPTICS ve DENCLUE algoritmaları verilebilir.

Grid Temelli Algoritmalar

Büyük boyuttaki veri tabanlarının kümelenmesinde numaralandırılmış çizgilerden oluşan hücresel yapıları kullanan algoritmalarıdır. Bu algoritmalara örnek olarak ise bölgenin dikdörtgen hücrelere bölünerek hiyerarşik bir yapının kullanıldığı STING algoritması, değişik şekillerde kümeler sunabilen ve hassas kümeleme kabiliyeti olan dalga kümeleme algoritması ve hem yoğunluğa hem de grid yapısına sahip CLIQUE algoritması verilebilir.

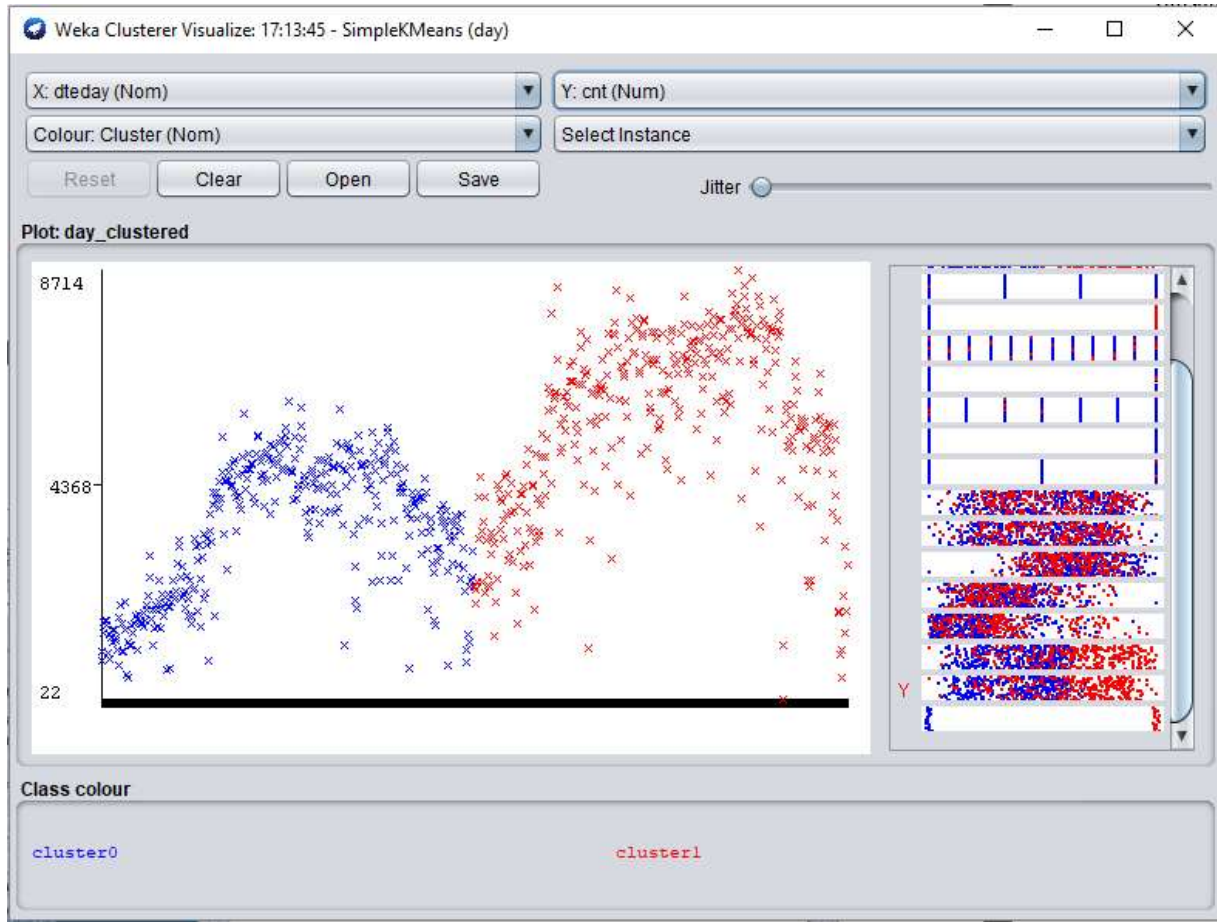


SimpleKMeans Kümeleme Modülünün ile elde Edinilen Kümeleme Bilgileri



SimpleKMeans Modülünün Kullanımı Neticesindeki Kümeleme Yüzdeleri

SimpleKMeans kümeleme modülü ile elde edilen tüm çıktıları [kümeleme.txt](#) içerisinde bulabilirsiniz.



Yukarıda görüldüğü üzere veri setimiz cluster0 ve cluster1 olarak iki kümeye ayrıldı. Bu ikiye ayırma işlemi yıl olarak ele alındı. Mavi renk 2011 yılındaki günleri, kırmızı renk ise 2012 yılındaki günleri temsil ediyor.

WEKA paket programında X ve Y eksenindeki değerler ile oynayarak karşımıza gelen grafikleri yorumlayabiliriz. Bu sayede bu veri setinde bulunan sıcaklık, rüzgâr hızı, iş günleri vs değişkenler ile o günler kiralanan bisiklet sayısının değişkenliğini gözlemleyebiliriz.

SONUÇ

Bu çalışmada veri madenciliğinin temel aşamaları olan veri ön işleme ve veri madenciliği algoritmalarının kullanımı ele alınmıştır. İlgili süreçlerle ilgili temel bilgiler verilmiş, kullanılacak yöntemler tanıtılmış ve örnek bir veri tabanı üzerinde seçilen yöntemlerin uygulaması WEKA yazılımı kullanılarak gerçekleştirilmiştir.

KAYNAKÇA

https://rpubs.com/aashish/bikesharing_EDA

<http://embk.mmoizmir.org/wp-content/uploads/2016/05/emyk18.pdf>

<http://dergipark.gov.tr/download/article-file/55797>

<https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset>

<https://www.youtube.com/watch?v=VQLAUXfn0Lc>

https://www.researchgate.net/publication/286048668_Veri_Madenciligi_Kumeleme_ve_Siniflama_Algoritmaları

<http://bilgisayarkavramlari.sadievrenseker.com/2011/09/19/weka-ile-svm/>

<https://www.cs.waikato.ac.nz/ml/weka/mooc/advanceddataminingwithweka/slides/Class3-AdvancedDataMiningWithWeka-2016.pdf>

<https://machinelearningmastery.com/use-regression-machine-learning-algorithms-weka/>

https://www.youtube.com/watch?time_continue=806&v=5s8IgMfH698

<https://sadanand-singh.github.io/posts/svmmodels/>

<https://www.slideshare.net/oguzhantas/destek-vektor-makinelari-support-vector-machine>

<http://cagriemreakin.com/veri-bilimi/support-vector-machine-svm-8.html>

<https://zeynepozturkk.wordpress.com/2018/07/30/weka-ile-veri-madenciligi-4-kumelemeclustering/>

<https://kodcu.com/2013/02/java-weka-k-means-clustering-veri-kumeleme/>

<http://bilgisayarkavramlari.sadievrenseker.com/2008/12/15/k-ortalama-algoritmasi-k-means-algorithm/>