KOCAELİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BLM429 VERİ MADENCİLİĞİ DÖNEM ÖDEVİ

SINIFLANDIRMA ALGORİTMALARI ANALİZİ

İrem GÜLTEN / 170201064

PROBLEM

Proje kapsamında "Kaggle" web sitesinde bulunan "Kredi Tahmini" veri seti kullanılmıştır. Bu veri setinde bankaların müşterilerine vereceği krediler için, kredi çekmek isteyen müşterinin durum analizi yapılması esas alınmıştır. Bu durumda kredi talebinin onaylanıp onaylanmayacağını makine öğrenmesi algoritmaları ile öngörmemizi gerektiren bir sınıflandırma problemi ile karşılaşmaktayız.

VERİ SETİ ANALİZİ

Kullanılan veri setinde

- Loan_ID (Müşteri Numarası),
- Gender (Cinsiyet),
- Married (Evlilik),
- Dependents (Bağımlı Kişi Sayısı),
- Education (Eğitim),
- Self_Employed (Çalışma Şekli),
- ApplicantIncome (Gelir),
- CoapplicantIncome (Kefil Geliri),
- LoanAmount (Kredi Tutarı),
- Loan_Amount_Term (Kredi Vadesi),
- Credit_History (Kredi Geçmişi),
- Property_Area (Bölge) ,
- Loan_Status (Kredi Durumu)

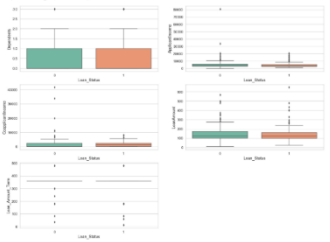
değişkenleri yer almaktadır. Değişkenlerimiz, 12 bağımsız değişken ve 1 hedef değişkeni olarak ayrılmaktadır. Değişkenlerimizin veri tiplerine baktığımızda başlangıçta object, int64 ve float64 veri tiplerinde olduklarını görülmekte fakat ön işleme aşamasında veri tiplerinde bazı değişikliklere gidilmiştir. Veri setinin analizi sırasında info() fonksiyonu ile veri seti incelenmiş, shape komutu ile sütun ve satır sayısına bakılmış, describe() fonksiyonu ile sayısal verilerin istatistiksel analizleri incelenmiştir.

ÖN İŞLEME (PREPROCESS)

Ön işleme aşamasında öncelikle veri setindeki tekrarlı satırların sayısı kontrol edilmiş ve veri satında herhangi bir tekrar eden satır bulunmadığı görülmüştür. Ardından veri setinde bulunan kayıp (Nan) hücrelerin kontrolü gerçekleşmiştir. Nan olan satırların sayısına bakıldığında, 614 kayıt olan veri setinin 149 satırının Nan değer olduğu görülmüştür. Bunun ortadan kaldırılması Nan olan hücrelerin bulunduğu satırın veri setinden atılması ya da ilgili hücrelerin sütundaki diğer hücrelerin ortalama, mod, medyan gibi istatistiksel verileriyle doldurulması ile sağlanabilmektedir. Veri setimizde öncelikle bizim için önemsiz olacağı düşünülen "Cinsiyet ve Evlilik" değişkenlerinde Nan değerler olan hücrelerin bulunduğu satırlar veri setinden kaldırılmıştır. Verilere bakıldığında "Yes,No" , "Y,N" gibi object tipli verilerin bulunduğu ve matematiksel işlemlerde bu değişkenlerin kullanıma uygun olmadığı görülmüştür. Bunun üzerine ilgili değişkenlerin değerleri replace() komutu kullanılarak maskeleme yapılıp int, float tiplerine dönüştürülmüşlerdir. Ardından bu değişkenlerdeki Nan değerler sütundaki diğer verilerden yola çıkılarak medyan değeri ile doldurulmuştur. Yapılan bu işlemler sonucunda shape komutu ile kontrol edildiğinde veri setinde 598 kayıt kaldığı görülmüştür. Replace fonksiyonunda değer ataması gerçekleştirildiğinde 0-1 ataması yapıldığında bu değerlerin float olarak tutulduğu görülmüştür ve bu daha sonra uygulanan Smote işleminde sentetik veri üretiminde sorun

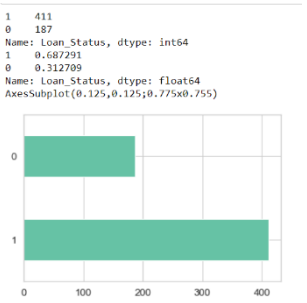
oluşturmuştur. Bunun önüne geçmek için tüm sütunlar astype() fonksiyonu kullanılarak int64 tipine dönüştürülmüştür.

Ardından veri setinde bulunan aykırı (outlier) verilerin anlaşılması için verilerin kutu grafiğinde aykırı değer analizi gerçekleştirilmiştir. Kutu grafiği gösteriminde aykırı değerler nokta ile gösterilirken veride yoğun olarak bulunan değerler kutular halinde gösterilmektedir. Aykırı değer analizine bakıldığında bazı aykırı değerlerin bulunduğu görülmüş fakat bu değerler algoritma sonucuna etki edebileceği için değerli görülüp herhangi bir işlem uygulanmamıştır.

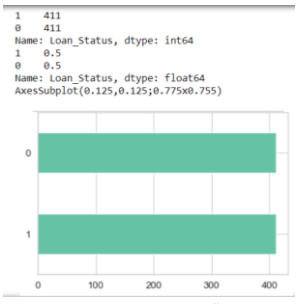


Şekil.1 Aykırı Değer Kutu Haritası

Sonrasında veri setinin dengesizliğinin kontrolü hedef sütunundaki farklı değerler için veri sayısına bakılarak yapılmıştır. Buna göre veri setindeki hedef değişkeninde 0 için 187, 1 için 411 kayıt olduğu görülmüştür. Bu değerler veri setinin dengesizliğinin göstergesidir ve bunu ortadan kaldırmak için over-sampling yöntemi uygulanmıştır. Over-sampling yöntemi, azınlık sınıfa ait verilerin sayısının ağırlıklı sınıda yaklaştırılmasıdır. Over-sampling uygulanması Smote yöntemi ile gerçekleştirilmiştir. Bu yöntemde her azınlık sınıfın örneği alınıp bu örneğe ait komşulara bakılarak sentetik örnekler oluşturulmaktadır. Smote işlemi sonrasında veri setinde 0 için 411 ve 1 için 411 olmak üzere toplam 822 kayıt olmuştur.



Şekil.2 Veri Dağılımı

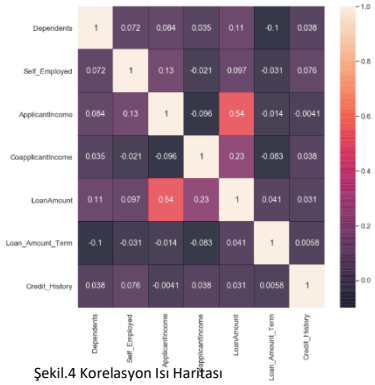


Şekil.3 SMOTE Sonrası Veri Dağılımı

Veri setinin dengesizliğinin giderilmesinin ardından bağımsız değişkenlerin birbirlerinden ne kadar etkilediklerinin anlaşılabilmesi için korelasyon analizi yapılmış ve bu analiz ısı haritası üzerinde gösterilmiştir. Korelasyon analizi değişkenler arası ilişkinin ölçümüdür. Analiz sonucunda hesaplanan değerin:

r < 0.2 ise korelasyon çok zayıf ya da yok,
0.2 < r < 0.4 ise zayıf korelasyon,
0.4 < r < 0.6 ise orta şiddette korelasyon,
0.6 < r < 0.8 ise yüksek korelasyon,
r > 0.8 ise çok yüksek korelasyon

şeklinde yorumlanması gerçekleştirilmektedir. Isı haritasında açık renkler yüksek korelasyonu göstermektedir. Veri setinin korelasyon analizi sonucunda çıkan ısı haritasına bakıldığında sadece iki bağımsız değişken arasında orta şiddette korelasyon; diğer bağımsız değişkenler arasında ise korelasyonun olmadığı ya da çok zayıf korelasyon olduğu görülmüştür. Bağımsız değişkenler aralarında düşük korelasyona sahip oldukları için ön işleme aşamasında herhangi boyut indirgeme yöntemi (PCA, LDA gibi) kullanılmamıştır.



Şekil.4 Korelasyon Isı Haritası

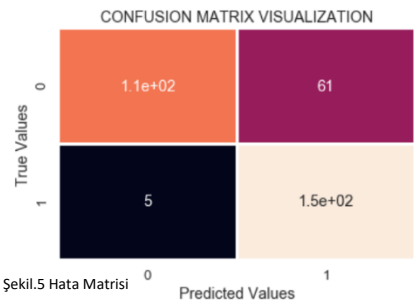
Son olarak bağımsız değişkenlere standardizasyon işlemi uygulanmıştır. Standardizasyon işleminde değişkenlerin standart bir normal dağılım özelliklerine sahip olacak şekilde yeniden ölçeklendirilmesidir. Ardından veri seti train(eğitim) seti ve test seti olarak ayrılmıştır. Bu ayrım yapılırken test_size ve random_state parametreleri farklı değerlerle denenerek algoritmalar sonucunda en yüksek accuracy (doğruluk) değerini veren değerler tercih edilmiştir. Buna göre test_size değeri 0.4 olarak belirlenirken, random_state değeri 120 olarak belirlenmiş ve sınıflandırma modellerinin uygulanmasına geçilmiştir.

SINIFLANDIRMA MODELLERİNİN UYGULANMASI

Ön işleme aşamasının ardından sınıflandırma algoritmalarının kullanımına geçilmiştir. Projede sınıflandırma algoritmalarından SVM (Support Vector Machine), NB (Naive Bayes), KNN (K-Nearest Neighbor) ve Random Forest algoritmaları kullanılmıştır. Algoritmaların uygulanması ile birlikte algoritmalara dair hata(confusion) matrisleri hesaplanıp bu matrisler ısı haritası üzerinde gösterilmiş ve makine öğrenmesi kütüphanesi Sklearn'e ait classification_report() fonksiyonu ile ilgili algoritmaya ait precision (kesinlik), recall(hassaslık), f1-score ve support değerleri tabloda gösterilmiştir.

Öncelikle SVM (Support Vector Machine) sınıflandırma algoritması uygulanmıştır. Bu algoritme iki sınıfı ayıran en iyi çizgiyi bulmaya çalışmaktadır. Algoritma, çizilecek doğruyu iki sınıfın da elemanlarına en uzak yerde geçecek şekilde ayarlamaktadır. Modelin uygulanmasından önce modele verilecek parametreler üzerinde denemeler yapılarak tespit edilmiştir. Veri setinin sınıflandırılmasında en yüksek accuracy(doğruluk) değerini veren parametreler C= "3" , kernel = "rbf" olarak tespit edilmiştir. Model sonucunda çıkan hata(confusion) matrisine bakıldığında 0 olarak sınıflandırılması gereken 171 verinin 110'unun doğru, 61'inin yanlış sınıflandırıldığı; 1 olarak sınıflandırılması gereken 158 verinin 153'ünün doğru 5'inin yanlış sınıflandırıldığı görülmüştür. Veri setine SVM algoritması uygulanması sonucunda accuracy (doğruluk) değeri 0.7994 olarak hesaplanmıştır.

```
[[110  61]
 [  5 153]]
```



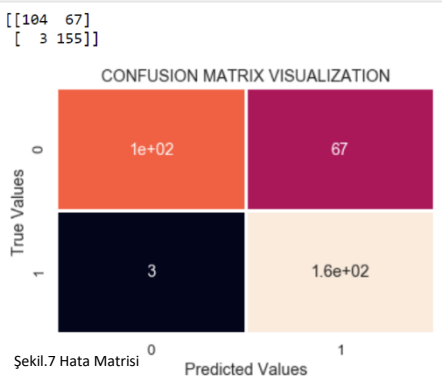
Şekil.5 Hata Matrisi

	precision	recall	f1-score	support
0	0.96	0.64	0.77	171
1	0.71	0.97	0.82	158
accuracy			0.80	329
macro avg	0.84	0.81	0.80	329
weighted avg	0.84	0.80	0.79	329

Şekil.6 Sınıflandırma Raporu

Ardından NB (Naive Bayes) sınıflandırma algoritması uygulanmıştır. Naive Bayes sınıflandırıcısının temeli Bayes teoremine dayanmaktadır. Algoritma, bir eleman için her durumun olasılığını hesaplayıp olasılık değeri en yüksek olana göre sınıflandırma gerçekleştirmektedir. Model sonucunda çıkan hata(confusion) matrisine bakıldığında 0 olarak sınıflandırılması gereken 171 verinin 104'ünün doğru, 67'sinin yanlış sınıflandırıldığı; 1 olarak sınıflandırılması gereken 158 155'inin doğru, 3'ünün yanlış sınıflandırıldığı görülmüştür. Veri setine NB algoritması uygulanması sonucunda accuracy (doğruluk) değeri 0.7872 olarak hesaplanmıştır.

```
[[104  67]
 [  3 155]]
```



Şekil.7 Hata Matrisi

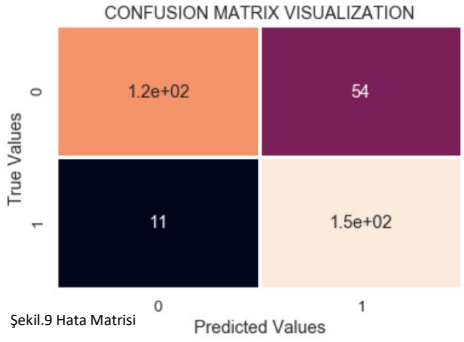
	precision	recall	f1-score	support
0	0.97	0.61	0.75	171
1	0.70	0.98	0.82	158
accuracy			0.79	329
macro avg	0.84	0.79	0.78	329
weighted avg	0.84	0.79	0.78	329

Şekil.8 Sınıflandırma Raporu

Ardından KNN (K-Nearest Neighbor) algoritması uygulanmıştır. Algoritma çalıştırılırken bir K değeri (bakılacak eleman sayısı) belirlenmektedir. Bir değer geldiğinde en yakın K kadar eleman alınarak gelen değerle arasındaki ilişkileri (uzaklıkları) hesaplanmaktadır ve buna

göre sınıflandırma yapılmaktadır. Modelin gerçekleştirilmesi default olarak 5 komşuya bakılarak gerçekleştirilmektedir. Farklı değerlere bakıldığında en yüksek accuracy (doğruluk) değerini 7 komşunun kontrolü ile verdiği görülmüştür. Model sonucunda çıkan hata(confusion) matrisine bakıldığında 0 olarak sınıflandırılması gereken 171 verinin 117'sinin doğru, 54'ünün yanlış sınıflandırıldığı; 1 olarak sınıflandırılması gereken 158 verinin 147'sinin doğru 11'inin yanlış sınıflandırıldığı görülmüştür. Veri setine KNN algoritması uygulanması sonucunda accuracy (doğruluk) değeri 0.8024 olarak hesaplanmıştır.

```
[[117  54]
 [ 11 147]]
```



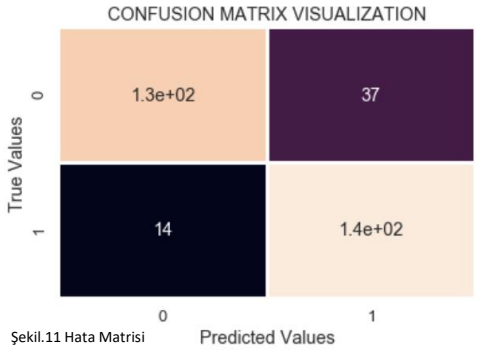
Şekil.9 Hata Matrisi

	precision	recall	f1-score	support
0	0.91	0.68	0.78	171
1	0.73	0.93	0.82	158
accuracy			0.80	329
macro avg	0.82	0.81	0.80	329
weighted avg	0.83	0.80	0.80	329

Şekil.10 Sınıflandırma Raporu

Son olarak Random Forest algoritması uygulanmıştır. Algoritmanın çalışma mantığına bakıldığında, birden fazla karar ağacının oluşturulup daha doğru ve istikrarlı bir tahmin elde etmek için onların birleştirildiğini görmekteyiz. Ağaç sayısı arttıkça elde edilecek sonuçların kesinliği de artmaktadır. Modelin gerçekleştirilmesi sırasında parametreler farklı değerler ile denenmiş ve en yüksek accuracy(doğruluk) değeri veren değerler seçilmiştir. Model sonucunda çıkan hata(confusion) matrisine bakıldığında 0 olarak sınıflandırılması gereken 171 verinin 134'ünün doğru, 37'inin yanlış sınıflandırıldığı; 1 olarak sınıflandırılması gereken 158 verinin 144'ünün doğru 14'ünün yanlış sınıflandırıldığı görülmüştür. Veri setine Random Forest algoritması uygulanması sonucunda accuracy (doğruluk) değeri 0.8450 olarak hesaplanmıştır.

```
[[134  37]
 [ 14 144]]
```



Şekil.11 Hata Matrisi

	precision	recall	f1-score	support
0	0.91	0.78	0.84	171
1	0.80	0.91	0.85	158
accuracy			0.84	329
macro avg	0.85	0.85	0.84	329
weighted avg	0.85	0.84	0.84	329

Şekil.12 Sınıflandırma Raporu

DENEYSEL SONUÇ

Veri madenciliği dönem projesinin sonucunda kredi çekmek isteyen müşterilerin ilgili bilgilerinin kullanılarak kredi başvurusunun onaylanıp onaylanamayacağını önceden kestirmenin mümkün olduğunu görmüş bulunmaktayız. Projenin sonunda 4 farklı sınıflandırma algoritması kullanılarak yapılan sınıflandırma analizlerinin accuracy (doğruluk) değerleri karşılaştırılmıştır. Bu karşılaştırmaya göre Random Forest algoritmasının %85 ile en yüksek accuracy (doğruluk) değerine sahip olduğunu görmekteyiz. En yüksek doğruluk değerine sahip bu algoritmanın seçilen "Kredi Tahmini" veri setinin analizi için en uygun algoritma olduğu sonucuna varılmıştır.

	Model	Accuracy
0	SVM	0.799392
1	NB	0.787234
2	KNN	0.802432
3	RF	0.844985

Şekil.12 Algoritma Karşılaştırılması

KAYNAKÇA

Browlee, J. (2020). *SMOTE for Imbalanced Classification with Python*. Machine Learning Mastery: https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/ adresinden alındı

CANGUVEN, G. (2019). *DENGESİZ VERİ KÜMELERİ İLE MAKİNE ÖĞRENMESİ.* MEDIUM: https://medium.com/@g.canguven11/dengesiz-veri-kümeleri-ile-makine-öğrenmesi-63bbac5f6869 adresinden alındı

CİRİT, A. (2020). *Telco Churn Turkish %81,3 accuaracy*. KAGGLE: https://www.kaggle.com/ahmetbugrahancirit/telco-churn-turkish-81-3-accuaracy adresinden alındı

ÇEBİ, C. (2020). *Rastgele Orman Algoritması.* MEDIUM: https://medium.com/@cemthecebi/rastgele-orman-algoritmas%C4%B1-1600ca4f4784 adresinden alındı

ERDEN, C. (2020). *Diyabet Hastalığı Tahmini için Bir Veri Madenciliği Uygulaması*. YOUTUBE: https://www.youtube.com/watch?v=OA3wQVMIoDI adresinden alındı

ERDEN, C. (2020). *Veri Madenciliği Final Sunumu*. YOUTUBE: https://www.youtube.com/watch?v=J-dSw5ADy8c&list=PLNVCj-z_HS4YpUuNJ-XbPrfeThCF-ROeb&index=26 adresinden alındı

github. (2018). *Kredi Tahmini*. GITHUB: https://github.com/ColeCola/KrediTahmini/blob/master/train_kredi_tahmini.csv adresinden alındı

github. (2019). *Data Preprocessing Project - Imbalanced Classes Problem*. GITHUB: https://github.com/pb111/Data-Preprocessing-Project-Imbalanced-Classes-Problem/blob/master/Data%20Preprocessing%20Project%20-%20Imbalanced%20Classes%20Problem.ipynb adresinden alındı

HATİPOĞLU, E. (2018). *Machine Learning — Classification — K-NN (K En Yakın Komşu).* MEDIUM: https://medium.com/@ekrem.hatipoglu/machine-learning-classification-k-nn-k-en-yak%C4%B1n-kom%C5%9Fu-part-9-6f18cd6185d adresinden alındı

HATİPOĞLU, E. (2018). *Machine Learning — Classification — Naive Bayes — Kernel Trick.* MEDIUM: https://medium.com/@ekrem.hatipoglu/machine-learning-classification-naive-bayes-part-11-4a10cd3452b4 adresinden alındı

KARADERİLİ, Ş. (2018). *Hata Matrisini Anlamak.* MEDIUM: https://medium.com/@sengul_krdrl/hata-matrisini-anlamak-7035b7921c0f adresinden alındı

NAYANSAK, M. (2020). *Python ile Makine Öğrenmesi*. GITHUB: https://github.com/MustafaNayansak/Python-ile-Veri-Bilimi-Uygulamalar- adresinden alındı

Sefa. (2019). *Python ile Veri Bilimi ve Makine Ogrenmesi*. GITHUB: https://github.com/Sefa314159/Python-ile-Veri-Bilimi-ve-Makine-Ogrenmesi/tree/master/Python-ile-VeriVeri-Bilimi-ve-Makine-Ogrenmesi/tree/master/3.%20Sınıflandırma%20Problemleri adresinden alındı