

**EGE ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**YAPAY ZEKÂ YÖNTEMLERİ (3+0)**

**2022-2023 BAHAR YARIYILI**

**PROJE-1 RAPORU**

**TESLİM TARİHİ**

07/05/2023

**HAZIRLAYANLAR**

05180000063, Burak KIZILAY

05190000080, Bahri Bilgekağan TAŞ

05210000915, Abdullah Alper BAŞ

İçindekiler

[1) 1) Algoritmalar, Tanımlar, Karşılaştırma, Araştırma ve Yorum 2](#_Toc134136205)

[1.a Kalemle yazılan algoritmaların tarayıcı görüntüleri veya fotoğrafları 2](#_Toc134136206)

[1.b Tanım ve Karşılaştırmalar 2](#_Toc134136207)

[1.c Araştırma ve Yorum 2](#_Toc134136208)

[2) Problem Çözme ve Kodlama 2](#_Toc134136209)

[2.a Problemin Tanımı 3](#_Toc134136210)

[2.b Çözüm Mekanizması ve Kaynak Kod 3](#_Toc134136211)

[2.c Programın Ekran Görüntüleri 3](#_Toc134136212)

[2.d Sonuç Tablosu 3](#_Toc134136213)

[3) Genetik Algoritmalar ile Şifre Kırma 2](#_Toc134136214)

[3.a İlgili Maddede İstenenler ve Karşılaştırma (kromozom sayısının etkisi) 2](#_Toc134136215)

[3.b Kod; Çaprazlama ve Mutasyon Fonksiyonlarının Anlatımı 2](#_Toc134136216)

[3.c Çözüm Süreleri Karşılaştırması 2](#_Toc134136217)

[4) Makine Öğrenmesi 3](#_Toc134136218)

[4.a Standardization ve Normalization Farkı ve Python Örneği 3](#_Toc134136219)

[4.b Veri Setinin ve Problemin Kısa Anlatımı 3](#_Toc134136220)

[4.c İki Farklı Sınıflandırıcı için Python Kodu 3](#_Toc134136221)

[4.d İki Farklı Sınıflandırıcı için Sonuçlar: Hata Matrisleri, Tablo 3](#_Toc134136222)

[4.e Kullanıcı tarafından verilen örneğin sınıflandırma ekran görüntüsü (konsol çıktıları) 3](#_Toc134136223)

[4.f ChatGPT Kullanımı ve Yorumlama 3](#_Toc134136224)

[5) Öz değerlendirme Tablosu 3](#_Toc134136225)

# 1) 1) Algoritmalar, Tanımlar, Karşılaştırma, Araştırma ve Yorum

## 1.a Kalemle yazılan algoritmaların tarayıcı görüntüleri veya fotoğrafları

Text, letter

Description automatically generated

Text, letter

Description automatically generated

Text, letter

Description automatically generated

## 1.b Tanım ve Karşılaştırmalar

i) GPT-4

GPT'nin açılımı Generative Pre-trained Transformer olup insan benzeri, konuşmaya dayalı metinler üretmek için derin öğrenmeyi kullanan bir tür dil modelidir. Generating Pre-trained Transformer 4'ün kısaltması olan GPT-4, OpenAI tarafından geliştirilen GPT büyük dil modelleri ailesinin dördüncü iterasyonudur. Yapay zeka sohbet robotu ChatGPT'ye güç veren GPT-3 modelinin yeni sürümü olan GPT-4, önceki sürümler gibi insan benzeri metin üretmek, özetleme ve dil çevirisi gibi görevleri tamamlamak ve hatta şiir, müzik sözleri ve kurgu gibi yaratıcı yazılar üretmek için tasarlanmıştır.

İİ)AutoML

AutoML, oldukça basit bir şekilde planlamayı destekleyen ve kullanııların ince ayar yapmasına ya da tekrar modelleme yapmasına gerek duymadığı makine öğrenme modellerinin uygulanmasına ve gelişmesine olanak tanıyan otomatik özellik, algoritma seçim ve uygulama sürecidir.

Makine öğrenimi, iş kullanıcılarının yükünü hafifletmeyi sağlar, ayrıca “akıllı” bir çözümün rehberi olarak iş kullanıcılarına yardımını sunarken makine öğrenmesinin esnekliğine ve karmaşıklığına rağmen, öngörücü analitik yapmak için temel iş bilgilerini ve becerilerini kullanmalarına olanak tanır. Sistem, veri kümesini yorumlar, önemli sütunları seçer, kategorileri türleri ve diğer parametreleri analiz eder. En iyi algoritmayı ve analitik tekniği otomatik olarak uygulamak ve veri bilgileri sağlamak için akıllı makine öğrenimini kullanır.

iii) Caption Generation

Görüntü altyazısı oluşturma (İmage Caption Generation), derin öğrenmede görüntülerin doğal dil açıklamalarını oluşturmayı içeren bir görevdir. İşlem, bir görüntüyü analiz etmek ve karşılık gelen bir resim yazısı oluşturmak için bilgisayarlı görü ve doğal dil işleme tekniklerinin bir kombinasyonunu kullanmayı içerir.

Görüntü alt yazısı oluşturmaya yönelik en yaygın yaklaşım, bir kodlayıcı-kod çözücü (encoder-decoder) mimarisine dayalıdır. Bu mimaride, görüntü önce üst düzey görsel özellikleri çıkarmak için evrişimli bir sinir ağı (CNN) aracılığıyla beslenir. Ortaya çıkan özellik vektörü daha sonra, karşılık gelen altyazıyı kelime kelime oluşturan uzun kısa süreli bellek (LSTM) ağı veya kapılı tekrarlayan birim (GRU) ağı gibi tekrarlayan sinir ağına (RNN) beslenir.

iv)Self-attention GAN

Self-attention GAN (Generative Adversarial Network), self-attention (öz-dikkat) mekanizmalarını jeneratör mimarisine dahil eden bir GAN türüdür. Bu mimari, Zhang ve diğerleri tarafından " Self-Attention Generative Adversarial Networks" makalesinde tanıtıldı (2019).

Self-attention GAN'larındaki öz-dikkat mekanizması, üreticinin özellik haritasındaki her piksel için bir dikkat matrisi hesaplayarak çalışır. Bu dikkat matrisi, son görüntüyü oluşturmak için özellik haritasındaki farklı piksellerin önemini yansıtır. Dikkat matrisi daha sonra, oluşturucunun bir sonraki katmanına girdi olarak kullanılan özellik haritasının ağırlıklı bir toplamını hesaplamak için kullanılır.

v) Random Forest vs Ensemble Learning

Ensemble Learning yani Topluluk Öğrenmesi, birden fazla modelin aynı sorunu çözmek için eğitilmesi ve bunun için birlikte çalışmalarına denir. Modeller sorunu ayrı ayrı çözer ve ortalama sonuç değerlendirilir.

Random Forest ise bir Topluluk Öğrenme yöntemidir. Karar Ağacının bir Ensemble Modelidir. Yani sınıflandırma işleminde birden fazla Karar Ağacı üretir. Random Forest, Ensemble olduğu için tek sınıflayıcılardan çok daha iyi sonuç üretir.

[1] <https://toptalent.co/gpt-4-nedir-gpt-4-nasil-kullanilir>

[2] <https://basakbuluz.medium.com/makale-okumaları-volm-3-a8c8bc94456f>

[3] <https://www.akademi40.org/otomatik-makine-ogrenimi-automl-nedir>

[4] https://www.youtube.com/watch?v=OVeGatovZ7Y

## 1.c Araştırma ve Yorum

Text, letter

Description automatically generated

# 2) Problem Çözme ve Kodlama

## 2.a Problemin Tanımı

8 Vezir Problemi ilk olarak 1848 yılında profesyonel satranç oyuncusu olan Max Bezzel tarafından ortaya atılmış, yıllar içerisinde önemli matematikçilerden Gauss ve Georg Cantor’un da incelediği bu problemin ilk çözümü 1950 yılında Franz Nauck tarafından sunulmuştur. Ayrıca Franz Nacuk, problem NxN’lik tahta üzerinde genelleştirerek “N Vezir Problemi” haline getirmiştir.

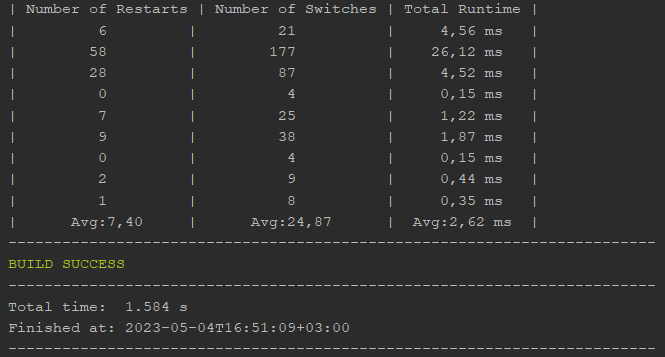
8 Vezir problemindeki amaç, 8x8 (8 satır, 8 sütun) satranç tahtası üzerinde 8 vezirin birbirini tehdit etmeyecek şekilde konumlandırılmasıdır. (Aynı satır, aynı sütun veya aynı çapraz sırada iki ve ikiden fazla vezir bulunamaz.)

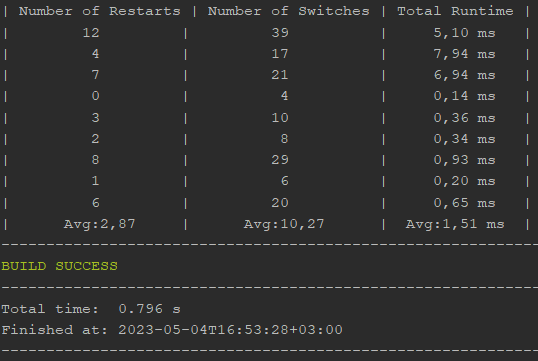
## 2.b Çözüm Mekanizması ve Kaynak Kod

8 vezirin her biri bir sütuna yerleştirildiğinde program daha optimal bir şekilde çalışacağı için bu yöntem tercih edildi ve her bir vezirin yeri, int tipinde verileri tutan tek boyutlu bir diziye yerleştirildi.

Derste anlatılan Hill Climbing Algoritmasının üzerinde herhangi bir değişiklik yapılmadı. Sorunun açıklamasında belirtildiği gibi lokal maksimum noktalarında takıldığı durumlarda random restart yaparak problemin çözümünün devam etmesi sağlandı. Java'nın içinde olan birkaç adet kütüphane hariç kütüphane kullanılmadı.

## 2.c Programın Ekran Görüntüleri

1 - 

2 - 

## 2.d Sonuç Tablosu

Bu bölümde istenen tüm bilgiler 2.3 Programın Ekran Görüntüleri’nde yer almaktadır.

# 3) Genetik Algoritmalar ile Şifre Kırma

## 3.a İlgili Maddede İstenenler ve Karşılaştırma (kromozom sayısının etkisi)

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

*Figure 1: 50 Kromozom ile Problemi Çözdürme – 1*

*metin içeren bir resim

Açıklama otomatik olarak oluşturuldu*

*Figure 2: 50 Kromozom ile Problemi Çözdürme – 2*

*metin içeren bir resim

Açıklama otomatik olarak oluşturuldu*

*Figure 3: 50 Kromozom ile Problemi Çözdürme – 3*

*metin içeren bir resim

Açıklama otomatik olarak oluşturuldu*

*Figure 4: 100 Kromozom ile Problemi Çözdürme – 1*

*metin içeren bir resim

Açıklama otomatik olarak oluşturuldu*

*Figure 5: 100 Kromozom ile Problemi Çözdürme – 2*

*metin içeren bir resim

Açıklama otomatik olarak oluşturuldu*

*Figure 6: 100 Kromozom ile Problemi Çözdürme – 3*

*metin içeren bir resim

Açıklama otomatik olarak oluşturuldu*

*Figure 7: 150 Kromozom ile Problemi Çözdürme – 1*

*metin içeren bir resim

Açıklama otomatik olarak oluşturuldu*

*Figure 8: 150 Kromozom ile Problemi Çözdürme – 2*

*metin içeren bir resim

Açıklama otomatik olarak oluşturuldu*

*Figure 9: 150 Kromozom ile Problemi Çözdürme – 3*

|  |  |  |
| --- | --- | --- |
| Kromozom Sayısı | Nesil Ortalaması | Süre Ortalaması |
| 50 | 1606 | 275 |
| 100 | 1002 | 276 |
| 150 | 631 | 280 |

Kromozom sayısı arttıkça daha fazla işlem yapılacağı için süre artar fakat daha fazla çocuk olacağı için öncekinden daha iyi bir çocuk oluşma olasılığı artar. Bu da daha az nesilde başarıya ulaşmayı sağlar.

## 3.b Kod; Çaprazlama ve Mutasyon Fonksiyonlarının Anlatımı

Çaprazlama fonksiyonumuzun adı “breed”. Parametre olarak char array tipinde 2 parent kullanarak char array tipinde 1 child döndürüyor. Bunu yaparken Two Point Crossover kullanıyoruz. Çaprazlamanın gerçekleşeceği aralığı 2 adet random int değeri seçerek belirliyoruz. Bu iki random değerden küçük olan başlangıç noktası, büyük olan da bitiş noktasıdır. Bu iki nokta arasında kalan genleri parent2’den, dışında kalan genleri de parent1’den alıyoruz. Bunun sonucuda yeni child oluşmuş oluyor.

Mutasyon fonksiyonumuzun adı “mutation”. Parametre olarak iki boyutlu bir char array olan child\_list kullanıyor. Bir for döngüsü ile yeni jenerasyondaki her kromozomu dönüyor ve bunun sonucunda her kromozom %10 ihtimalle mutasyona uğruyor. Mutasyon sırasında da kromozom içindeki rastgele bir gen rastgele bir harfe dönüşüyor.

**KOD**

package com.mycompany.soru3;

/\*\*

\*

\* @author burak

\*/

import java.util.Arrays;

import java.util.Locale;

import java.util.Random;

public class Soru3 {

public static char get\_random\_char(){

int leftLimit = 32; // "space"

int rightLimit = 122; // letter 'z'

int targetStringLength = 1;

//System.out.println(i);

Random random = new Random();

String generatedString = random.ints(leftLimit, rightLimit + 1)

.limit(targetStringLength)

.collect(StringBuilder::new, StringBuilder::appendCodePoint, StringBuilder::append)

.toString();

if (random.nextBoolean()){

return generatedString.toUpperCase().charAt(0);

}else{

return generatedString.charAt(0);

}

}

public static char[] get\_random\_char\_arr(int length){

char[] arr = new char[length];

for (int i = 0; i < length; i++) {

arr[i] = get\_random\_char();

}

return arr;

}

public static void print\_2d\_char\_arr\_with\_score(char[][] arr, double[] scores){

for (int i = 0; i < arr.length; i++) {

for (int j = 0; j < arr[0].length; j++) {

System.out.print(arr[i][j]);

}

System.out.println(scores[i]);

}

System.out.println("------------------------------");

}

public static char[] get\_copy\_element\_RS(char[][] generation){

//rank selection

//assume that array is revers sorted according to their fitness score

int totalsum = (generation.length) \* (generation.length + 1) / 2;

Random rnd = new Random();

int random\_number = rnd.nextInt(totalsum);

int partial\_sum = 1;

for (int i = generation.length-1; i >= 0; i--) {

partial\_sum += i;

if (partial\_sum>=random\_number){

return generation[i];

}

}

return generation[generation.length-1];

}

public static char[][] get\_new\_generation(char[][] parents, int elite\_size, int population\_size){

char[][] children = new char[population\_size][parents[0].length];

char[][] reversed\_parents = new char[parents.length][parents[0].length];

for (int i = reversed\_parents.length-1; i >= 0; i--) {

reversed\_parents[i] = parents[parents.length-i-1];

}

int index = 0;

for (index = 0; index < elite\_size; index++) {

children[index] = reversed\_parents[index];

}

for (; index < population\_size; index++) {

char[] parent1 = get\_copy\_element\_RS(parents);

char[] parent2 = get\_copy\_element\_RS(parents);

/\*while (Arrays.equals(parent1, parent2)){

parent2 = get\_copy\_element\_RS(parents);

}\*/

children[index] = breed(parent1,parent2);

}

return children;

}

private static char[] breed(char[] parent1, char[] parent2) {

char[] child = new char[parent1.length];

Random rnd = new Random();

int a = rnd.nextInt(child.length);

int b = rnd.nextInt(child.length);

int start = Math.min(a,b);

int end = Math.max(a,b);

for (int i = 0; i < child.length; i++) {

if(i < start || i>end){

child[i] = parent1[i];

}else{

child[i] = parent2[i];

}

}

return child;

}

public static void calc\_fit\_score(char[][] population, double[] scores, char[] pass) {

for (int i = 0; i < population.length; i++) {

double matches = 0;

for (int j = 0; j < pass.length; j++) {

if (population[i][j] == pass[j]){

matches++;

}

}

scores[i] = ((double) matches / pass.length) \* 100;

}

quickSort(population,scores,0,population.length-1);

}

public static void quickSort(char[][] gen,double[] fit\_score,int low, int high) {

if (low < high)

{

int pi = partition(gen,fit\_score,low, high);

quickSort(gen,fit\_score,low, pi - 1);

quickSort(gen,fit\_score,pi + 1, high);

}

}

public static int partition(char[][] gen,double[] fit\_score,int low, int high) {

double pivot = fit\_score[high];

int i = (low-1);

for (int j=low; j<high; j++)

{

if (fit\_score[j] <= pivot)

{

i++;

double temp = fit\_score[i];

char[] temp2 = gen[i];

fit\_score[i] = fit\_score[j];

gen[i] = gen[j];

fit\_score[j] = temp;

gen[j] = temp2;

}

}

double temp = fit\_score[i+1];

char[] temp2 = gen[i+1];

fit\_score[i+1] = fit\_score[high];

gen[i+1] = gen[high];

fit\_score[high] = temp;

gen[high] = temp2;

return i+1;

}

public static void mutation(char[][] child\_list){

Random rnd = new Random();

for (int i = 0; i < child\_list.length; i++) {

if(rnd.nextInt(10)<1){

child\_list[i][rnd.nextInt(child\_list[i].length)] = get\_random\_char();

}

}

}

public static void main(String[] args) {

Random rnd = new Random();

int population\_size = 150;

int number\_of\_parents = population\_size / 5;

int number\_of\_elites = population\_size / 50;

//int generation\_count = 10;

String pass\_code = "ChatGPT and GPT-4";

int pass\_length = pass\_code.length();

char[][] prev\_gen = new char[population\_size][pass\_length];

for (int i = 0; i < population\_size; i++) {

prev\_gen[i] = get\_random\_char\_arr(pass\_length); //first generation

}

char[][] parents = new char[number\_of\_parents][pass\_length];

double[] scores = new double[population\_size];

int count = 1;

long startTime = System.currentTimeMillis();

while (scores[population\_size-1] != 100 && count<10000) {

calc\_fit\_score(prev\_gen,scores,pass\_code.toCharArray());

quickSort(prev\_gen,scores,0,prev\_gen.length-1);

//print\_2d\_char\_arr\_with\_score(prev\_gen,scores);

System.out.println("Best score in generation " + count + ": " + scores[population\_size-1]);

System.out.println(String.valueOf(prev\_gen[population\_size-1]));

for (int i = 0; i < number\_of\_parents; i++) {

parents[i] = get\_copy\_element\_RS(prev\_gen);

}

double[] parent\_fit = new double[number\_of\_parents];

calc\_fit\_score(parents,parent\_fit, pass\_code.toCharArray());

//quickSort(parents,parent\_fit,0,parents.length-1);

char[][] next\_gen = get\_new\_generation(parents,number\_of\_elites,population\_size);

mutation(next\_gen);

calc\_fit\_score(next\_gen,scores,pass\_code.toCharArray());

//print\_2d\_char\_arr(next\_gen,scores);

for (int i = 0; i < prev\_gen.length; i++) {

prev\_gen[i] = Arrays.copyOf(next\_gen[i], next\_gen[i].length);

}

count++;

}

long endTime = System.currentTimeMillis();

System.out.println("--------------");

if(scores[population\_size-1] == 100){

System.out.println("Result found after " + count + " generation.");

System.out.println(String.valueOf(prev\_gen[population\_size-1]));

System.out.println("That took " + (endTime - startTime) + " milliseconds");

}else{

System.out.println("After " + count + " generation result couldn't been found. Best score is: " + scores[population\_size-1]);

System.out.println("That took " + (endTime - startTime) + " milliseconds");

}

//print\_2d\_char\_arr(temp);

}

}

## 3.c Çözüm Süreleri Karşılaştırması

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu metin içeren bir resim

Açıklama otomatik olarak oluşturuldu metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

*Figure 10: 50 Kromozom ile Problemi Çözdürme “ChatGPT”*

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu metin içeren bir resim

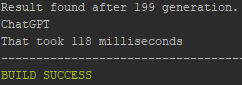
Açıklama otomatik olarak oluşturuldu metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

*Figure 11: 100 Kromozom ile Problemi Çözdürme “ChatGPT”*

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu metin içeren bir resim

Açıklama otomatik olarak oluşturuldu 

*Figure 12: 150 Kromozom ile Problemi Çözdürme “ChatGPT”*

|  |  |  |
| --- | --- | --- |
| Kromozom Sayısı | Nesil Ortalaması | Süre Ortalaması |
| 50 | 807 | 164 |
| 100 | 201 | 84 |
| 150 | 193 | 121 |

Hemen hemen yarıya düşmüş bir nesil ortalaması ve süre ortalaması görmekteyiz. Bunun sebebi de haliyle kullandığımız kelimenin hemen hemen yarıya düşmesinden kaynaklanmaktadır.

# 4) Makine Öğrenmesi

## 4.a Standardization ve Normalization Farkı ve Python Örneği

Hocam Standardization ve Normalization farkını az çok anladım fakat proje dosyasını yanlış okuduğum için normalizasyon (LabelEncoder) ile yapmış bulundum. Sonrasında da baya bir uğraş gösterdim fakat Standardization ile yapacak şekilde çeviremedim. LabelEncoder’a ek olarak OneHotEncoder ve StandardScaler birlikte kullanarak çevirebilme ihtimalim vardı fakat dediğim gibi başaramadım hocam. --- EK NOT: Hocam sonradan sırf bulunsun diye bir tane de random yaratılmış bir datasetiyle standardizasyon örneği yapmaya çalıştım elimden bu kadarı geldi. Bu sonradan denediğim standardization’ı da “soru4” klasörünün içindeki “Standardization baştan deneme” adlı klasörde bulabilirsiniz. Ama tabii ki bu projede soru 4 için uyguladığım esas çözüm “soru4” kalsöründeki “soru4” klasöründedir. --- Sorunun cevabına geçecek olursak:

Standardizasyon ve normalizasyon, bir modeli eğitmeden önce girdi verilerini dönüştürmek için makine öğreniminde kullanılan iki yaygın ön işleme (preprocessing) tekniğidir. İkisi arasındaki temel fark, verileri ölçeklendirme biçimleridir.

Z-skoru normalizasyonu olarak da bilinen standardizasyon, verileri sıfır ortalama (zero mean) ve birim varyansa (unit variance) sahip olacak şekilde ölçeklendirir. Bu, dönüştürülen verilerin ortalamasının 0 ve standart sapmasının 1 olacağı anlamına gelir. Standardizasyon, özelliklerin farklı ölçeklere sahip olduğu veya verilerin aykırı değerlere sahip olduğu durumlarda yararlıdır.

Öte yandan normalizasyon, verileri sabit bir aralıkta, genellikle 0 ile 1 arasında ölçeklendirir. Bu teknik, giriş özelliklerinin farklı ölçekleri olduğu ama aralarındaki göreli farkları korumak istediğiniz durumlarda kullanışlıdır.

## 4.b Veri Setinin ve Problemin Kısa Anlatımı

Kullandığımız veriseti bir mantar verisetidir. Veriler Kuzey Amerika mantarlarından oluşmaktadır. 8124 veri, 22 adet öznitelik ve 2 adet sınıf bulunmaktadır. Öznitelikler şunlardır:

* cap-shape: bell=b, conical=c, convex=x, flat=f, knobbed=k, sunken=s
* cap-surface: fibrous=f, grooves=g, scaly=y, smooth=s
* cap-color: brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y
* bruises: bruises=t, no=f
* odor: almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s
* gill-attachment: attached=a, descending=d, free=f, notched=n
* gill-spacing: close=c, crowded=w, distant=d
* gill-size: broad=b, narrow=n
* gill-color: black=k, brown=n, buff=b, chocolate=h, gray=g, green=r, orange=o, pink=p, purple=u, red=e, white=w, yellow=y
* stalk-shape: enlarging=e, tapering=t
* stalk-root: bulbous=b, club=c, cup=u, equal=e, rhizomorphs=z, rooted=r, missing=?
* stalk-surface-above-ring: fibrous=f, scaly=y, silky=k, smooth=s
* stalk-surface-below-ring: fibrous=f, scaly=y, silky=k, smooth=s
* stalk-color-above-ring: brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
* stalk-color-below-ring: brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
* veil-type: partial=p, universal=u
* veil-color: brown=n, orange=o, white=w, yellow=y
* ring-number: none=n, one=o, two=t
* ring-type: cobwebby=c, evanescent=e, flaring=f, large=l, none=n, pendant=p, sheathing=s, zone=z
* spore-print-color: black=k, brown=n, buff=b, chocolate=h, green=r, orange=o, purple=u, white=w, yellow=y
* population: abundant=a, clustered=c, numerous=n, scattered=s, several=v, solitary=y
* habitat: grasses=g, leaves=l, meadows=m, paths=p, urban=u, waste=w, woods=d

Sınıflar ise şunlardır:

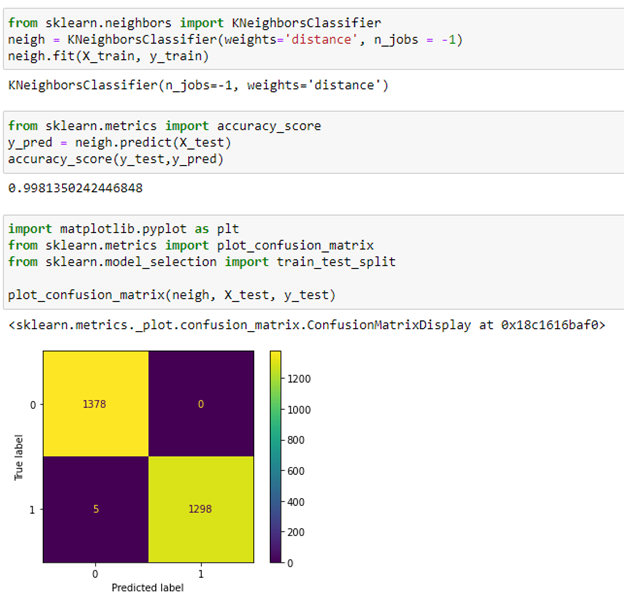
* edible=e
* poisonous=p

Programın amacı ise dış görünüş özellikleri bilinen bir mantarın yenilebilir(edible) veya zehirli(poisonous) olup olmadığını tahmin etmektir.

## 4.c İki Farklı Sınıflandırıcı için Python Kodu

Model eğitimlerine geçmeden önce label encoding yaptık. Label encoding yaparak korelasyon ısı haritası oluşturduk ve bu korelasyon ısı haritası sonucu hedef sınıfımızda düşük korelasyona sahip sütunları veri setinden düşürdük. Bunu da df\_corr isimli bir dataframe’e kaydettik.

Model için kNN ve SVC algoritmalarını kullandık. Modelde kullanacağımız verileri seçmek için de sklearn’ün train\_test\_split kütüphanesini kullandık.



*Figure 13: kNN algoritması için hata matrisi*



*Figure 14: SVC algoritması için hata matrisi*

10 fold cross validation yerine learning curve kullandık. (Bu konu hakkında Aybars Hoca’dan gerekli izin mail aracılığıyla alınmıştır.) Bunu kullanmamızın sebebi hem kendi içinde cross validation olması hem de overfitting veya underfitting olup olmadığını bize göstermesiydi. Sonucuna baktığımızda da ikisiyle de karşılaşmadığımızı görebiliyoruz.

import matplotlib.pyplot as plt

from sklearn.naive\_bayes import GaussianNB

from sklearn.svm import SVC

from sklearn.model\_selection import learning\_curve

from sklearn.model\_selection import ShuffleSplit

def plot\_learning\_curve(estimator, title, X, y, axes=None, ylim=None, cv=None,

n\_jobs=None, train\_sizes=np.linspace(.1, 1.0, 5)):

"""

Generate 3 plots: the test and training learning curve, the training

samples vs fit times curve, the fit times vs score curve.

"""

if axes is None:

\_, axes = plt.subplots(1, 3, figsize=(20, 5))

axes[0].set\_title(title)

if ylim is not None:

axes[0].set\_ylim(\*ylim)

axes[0].set\_xlabel("Training examples")

axes[0].set\_ylabel("Score")

train\_sizes, train\_scores, test\_scores, fit\_times, \_ = \

learning\_curve(estimator, X, y, cv=cv, n\_jobs=n\_jobs,

train\_sizes=train\_sizes,

return\_times=True)

train\_scores\_mean = np.mean(train\_scores, axis=1)

train\_scores\_std = np.std(train\_scores, axis=1)

test\_scores\_mean = np.mean(test\_scores, axis=1)

test\_scores\_std = np.std(test\_scores, axis=1)

fit\_times\_mean = np.mean(fit\_times, axis=1)

fit\_times\_std = np.std(fit\_times, axis=1)

# Plot learning curve

axes[0].grid()

axes[0].fill\_between(train\_sizes, train\_scores\_mean - train\_scores\_std,

train\_scores\_mean + train\_scores\_std, alpha=0.1,

color="r")

axes[0].fill\_between(train\_sizes, test\_scores\_mean - test\_scores\_std,

test\_scores\_mean + test\_scores\_std, alpha=0.1,

color="g")

axes[0].plot(train\_sizes, train\_scores\_mean, 'o-', color="r",

label="Training score")

axes[0].plot(train\_sizes, test\_scores\_mean, 'o-', color="g",

label="Cross-validation score")

axes[0].legend(loc="best")

# Plot n\_samples vs fit\_times

axes[1].grid()

axes[1].plot(train\_sizes, fit\_times\_mean, 'o-')

axes[1].fill\_between(train\_sizes, fit\_times\_mean - fit\_times\_std,

fit\_times\_mean + fit\_times\_std, alpha=0.1)

axes[1].set\_xlabel("Training examples")

axes[1].set\_ylabel("fit\_times")

axes[1].set\_title("Scalability of the model")

# Plot fit\_time vs score

axes[2].grid()

axes[2].plot(fit\_times\_mean, test\_scores\_mean, 'o-')

axes[2].fill\_between(fit\_times\_mean, test\_scores\_mean - test\_scores\_std,

test\_scores\_mean + test\_scores\_std, alpha=0.1)

axes[2].set\_xlabel("fit\_times")

axes[2].set\_ylabel("Score")

axes[2].set\_title("Performance of the model")

return plt

fig, axes = plt.subplots(3, 2, figsize=(10, 15))

title = "Learning Curves, Knn"

# Cross validation with 100 iterations to get smoother mean test and train

# score curves, each time with 20% data randomly selected as a validation set.

cv = ShuffleSplit(n\_splits=100, test\_size=0.2, random\_state=0)

estimator = KNeighborsClassifier()

plot\_learning\_curve(estimator, title, X\_train, y\_train, axes=axes[:, 0], ylim=(0.7, 1.01),

cv=cv, n\_jobs=-1)

title = r"Learning Curves (SVC, RBF kernel, $\gamma=0.001$)"

# SVC is more expensive so we do a lower number of CV iterations:

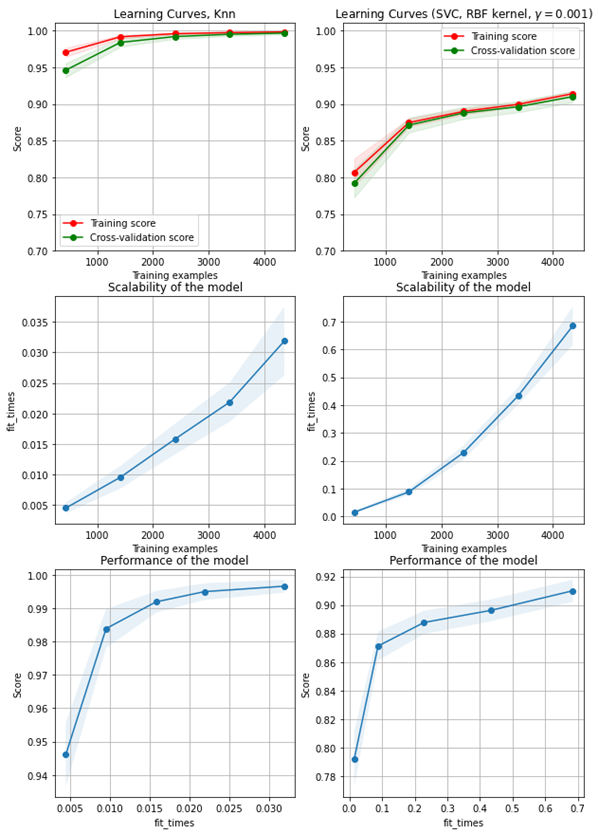
cv = ShuffleSplit(n\_splits=10, test\_size=0.2, random\_state=0)

estimator = SVC(gamma=0.001)

plot\_learning\_curve(estimator, title, X\_train, y\_train, axes=axes[:, 1], ylim=(0.7, 1.01),

cv=cv, n\_jobs=-1)

plt.show()

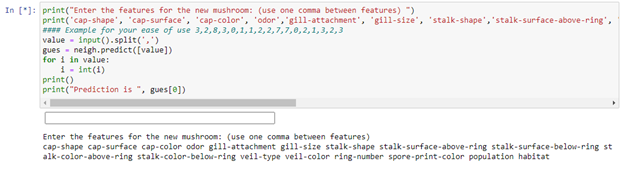


*Figure 15: Knn ve SVC için öğrenme eğrileri*

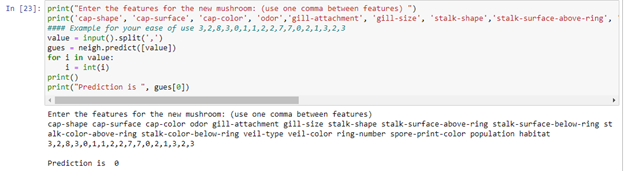
## 4.d İki Farklı Sınıflandırıcı için Sonuçlar: Hata Matrisleri, Tablo

Bu başlığı fark etmemiştim hocam, kodlarla birlikte hepsi 4.c maddesinde gösterilmiştir.

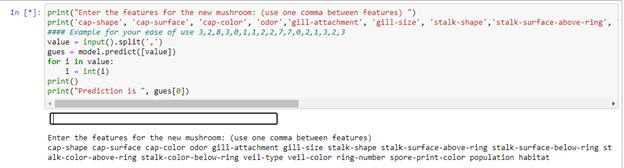
## 4.e Kullanıcı tarafından verilen örneğin sınıflandırma ekran görüntüsü (konsol çıktıları)



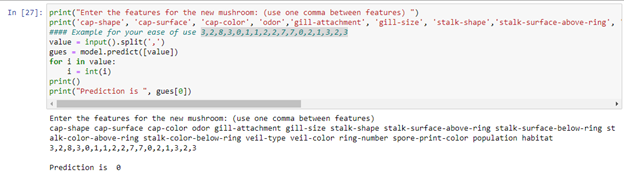
*Figure 16: Kullanıcı tarafından verilen örneğin kNN ile sınıflandırılması – 1*



*Figure 17: Kullanıcı tarafından verilen örneğin kNN ile sınıflandırılması – 2*



*Figure 18: Kullanıcı tarafından verilen örneğin SVC ile sınıflandırılması – 1*



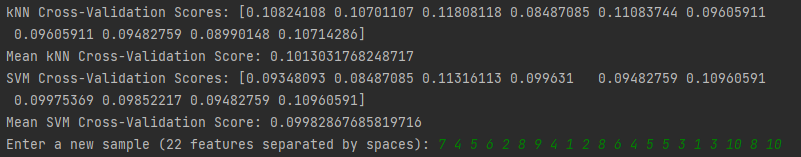
*Figure 19: Kullanıcı tarafından verilen örneğin SVC ile sınıflandırılması – 2*

## 4.f ChatGPT Kullanımı ve Yorumlama

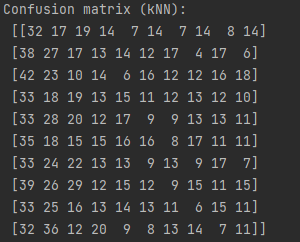
ChatGPT’ye istediklerimi hemen hemen yapabilen bir program yaptırdım. Çözümümü “soru4” klasöründeki “Standardization ChatGPTye yaptırma” klasörü içinde bulabilirsiniz. Yorumlayacak olursak eğer, verileri tabloları vs. görselleştirme anlamında bir katkı sağlayamadı ama yine de bir çözüm sundu.

Her maddeyi eklerken bir önceki maddedeki bir koşulu unuttu, atladı veya eksiltip değiştirdi gibi sorunlarla karşılaştım. Bu konuda hata yaptığını belirterek uyardığımda özür dileyip bunu düzeltti kimi zamansa tam anlamda düzeltemedi. En sonunda temel düzeyde bir program oluşturdu diyebilirim.

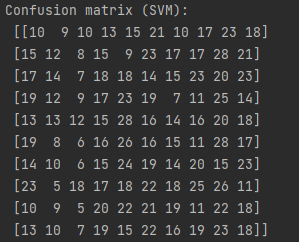
Aşağıda program çalıştırıldığında döndürdüğü çıktıların ekran görüntülerini bulabilirsiniz.



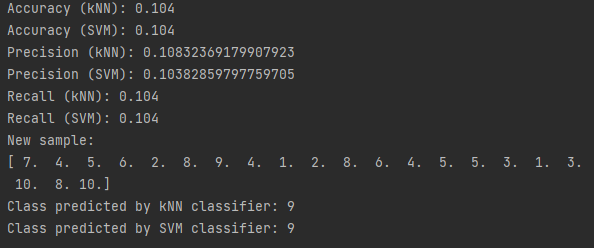
*Figure 20: ChatGPT ile Standardization uygulaması - 1*



*Figure 21: ChatGPT ile Standardization uygulaması - 2*

**

*Figure 22: ChatGPT ile Standardization uygulaması - 3*

**

*Figure 23: ChatGPT ile Standardization uygulaması - 4*

# 5) Öz değerlendirme Tablosu

**Açıklama kısmında yapıldı, yapılmadı bilgisi ve hangi maddelerin nasıl yapıldığı veya neden yapılamadığı kısaca yazılmalıdır.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **İstenen Özellik** | **Var** | **Açıklama** | **Tahmini Not** |
| **1a** | Algoritmalar + Karmaşıklıklar (10) | ✓ |  | 10 |
| **1b** | Tanım ve Karşılaştırmalar (10) | ✓ |  | 10 |
| **1c** | Araştırma ve Yorum (10) | ✓ |  | 10 |
| **2** | Problem Çözme ve Kodlama (10) | ✓ |  | 10 |
| **3** | Genetik Algoritmalar ile Şifre Kırma (15) | ✓ |  | 15 |
| **4** | Makine Öğrenmesi (25) | ✓ | Aslında puan alınacak maddeleri doğru şekilde yaptım fakat Standardization yerine Normalization içeren bir çözüm olduğu için tahmini bi puan kırışı yaptım. | 18 |
|  | Rapor (20) | ✓ |  | 20 |
| **100 üzerinden Toplam Not:** | | | | 93 |