

دانشکده مهندسي برق و کامپیوتر

نام و نام خانوادگی :

امیر اسماعیل زاده نوبری

شماره دانشجویی

40101924

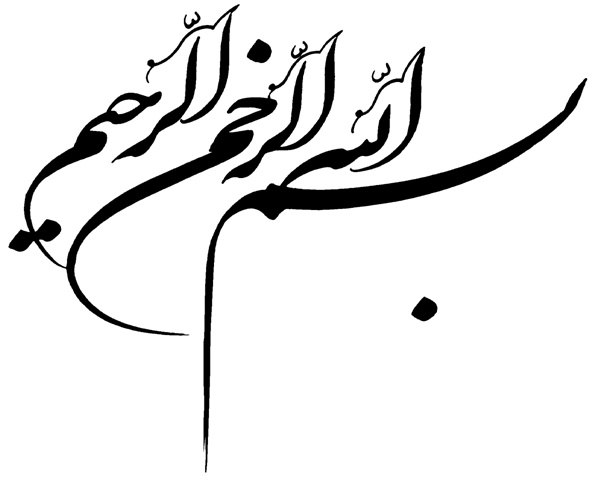
درس یادگیری ماشین

مینی پروژه دوم

استاد درس:

دکتر علیاری

بهار 1403



Contents

[سوال 1 4](#_Toc167394903)

[1 . 4](#_Toc167394904)

[2. 5](#_Toc167394905)

[3. 6](#_Toc167394906)

[ReLU: 9](#_Toc167394907)

[Sigmoid : 11](#_Toc167394908)

[سوال 2 13](#_Toc167394909)

[1 . 13](#_Toc167394910)

[2 . 16](#_Toc167394911)

[1) 16](#_Toc167394912)

[2)ReLU : 18](#_Toc167394913)

[3. 19](#_Toc167394914)

[4. 21](#_Toc167394915)

[k-fold 23](#_Toc167394916)

[Statified K-fold : 25](#_Toc167394917)

[3. 27](#_Toc167394918)

[دیتاست دارو 27](#_Toc167394919)

[توضیحات دیتاست 27](#_Toc167394920)

[1 . پیش پردازش 28](#_Toc167394921)

[2. 34](#_Toc167394922)

[3. 38](#_Toc167394923)

[AdaBoost : 39](#_Toc167394924)

[Random Forest 40](#_Toc167394925)

[نکته 40](#_Toc167394926)

[سوال 4 41](#_Toc167394927)

[Histogram 42](#_Toc167394928)

[Pairplot 43](#_Toc167394929)

[Corr 44](#_Toc167394930)

[1. 45](#_Toc167394931)

لینک GIT : <https://github.com/CAmiren/Machine_learning4022>

لینک colab :

https://drive.google.com/drive/folders/1H7JAWrO\_wJeoyRBBOWCv3oltXbbxCTD6?usp=sharing

# سوال 1

## 1 .



در یک مسئلۀ طبقه‌بندی دوکلاسه، هدف نهایی این است که احتمال تعلق یک نمونه به یکی از دو کلاس را پیش‌بینی کنیم. برای این منظور، معمولاً از یک نرون خروجی با فعال‌ساز سیگموید در لایۀ آخر استفاده می‌شود. سیگموید مقدار عددی را به یک مقدار بین ۰ و ۱ نگاشت می‌کند که می‌توان آن را به‌عنوان احتمال تفسیر کرد.

اگر دو لایۀ انتهایی شبکۀ شما به صورت فعال‌ساز ReLU و سیگموید تنظیم شده باشند:

1. فعال‌ساز ReLU (Rectified Linear Unit):

این فعال‌ساز هر مقدار منفی را به صفر نگاشت می‌کند و هر مقدار مثبت را به خود مقدار نگاشت می‌کند.

این فعال‌ساز کمک می‌کند که شبکه بتواند مقادیر منفی را به صفر تقلیل دهد و تنها مقادیر مثبت را انتقال دهد.

2. فعال‌ساز سیگموید:

این فعال‌ساز مقدار ورودی را به یک مقدار بین ۰ و ۱ نگاشت می‌کند.

این فعال‌ساز به دلیل این که مقدار خروجی آن بین ۰ و ۱ است، برای مسائل طبقه‌بندی دوکلاسه بسیار مناسب است زیرا می‌توان خروجی آن را به‌عنوان احتمال کلاس مثبت تفسیر کرد.

ترکیب فعال‌ساز ReLU و سیگموید در لایه‌های انتهایی :

لایۀ قبل از لایۀ نهایی با فعال‌ساز ReLU مقادیر ورودی منفی را به صفر تبدیل می‌کند و مقادیر ورودی مثبت را بدون تغییر انتقال می‌دهد.

لایۀ نهایی با فعال‌ساز سیگموید این مقادیر را به یک مقدار بین ۰ و ۱ نگاشت می‌کند.

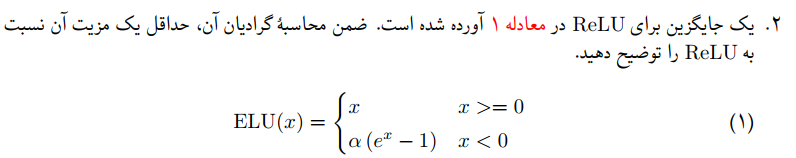
در نتیجه، مقادیر ورودی منفی که به صفر تبدیل شده‌اند، در لایۀ سیگموید خواهند داشت.

مقادیر مثبت در لایۀ سیگموید به یک مقدار بین ۰ و ۱ نگاشت خواهند شد که بستگی به مقدار خود دارند.

نتیجه‌گیری

این ترکیب به طور کلی مشکلی ایجاد نمی‌کند و شبکه به درستی می‌تواند آموزش ببیند. با این حال، مقادیر منفی که به صفر تبدیل شده‌اند، همیشه مقدار احتمال 0.5 خواهند داشت که ممکن است تفسیر خروجی را کمی پیچیده‌تر کند. به هر حال، در عمل، چنین ترکیبی به خوبی کار می‌کند و به شبکه کمک می‌کند که بتواند طبقه‌بندی دوکلاسه را به درستی انجام دهد.

## 2.



مزایای ELU نسبت به ReLU :

1)یکی از مزایای ELU نسبت به ReLU این است که مشکل "مرگ نرون‌ها" (Dead Neurons) را برطرف می‌کند. در ReLU، اگر ورودی به یک نرون منفی باشد، خروجی صفر می‌شود و گرادیان نیز صفر می‌شود. این مسئله می‌تواند باعث شود که نرون‌ها در طول آموزش به صفر قفل شوند و دیگر به‌روزرسانی نشوند، که به آن "مرگ نرون‌ها" گفته می‌شود.

با توجه به اینکه در ELU برای ورودی‌های منفی مقدار خروجی و گرادیان غیر صفر است، این فعال‌ساز از مرگ نرون‌ها جلوگیری می‌کند. حتی برای ورودی‌های منفی، خروجی غیر صفر و مشتق نسبتاً بزرگ است که باعث به‌روزرسانی وزن‌ها می‌شود.

2) یکی دیگر از مزایای ELU این است که میانگین خروجی نرون‌ها نزدیک به صفر است که می‌تواند باعث شود که فرآیند آموزش سریع‌تر و پایدارتر باشد. این ویژگی به بهبود همگرایی مدل کمک می‌کند.

## 3.



توضیح مختصر راجع به McCulloch-Pitts :

محاسبه‌گر McCulloch-Pitts، که توسط Warren McCulloch و Walter Pitts در سال ۱۹۴۳ معرفی شد، یکی از قدیمی‌ترین مدل‌های نورون مصنوعی است و پایه‌های نظریه شبکه‌های عصبی را تشکیل می‌دهد. این مدل یک نمایش ریاضی ساده از نورون‌های بیولوژیکی است و اصل اصلی پردازش اطلاعات در مغز را دربر می‌گیرد.

اجزاء یک McCulloch-Pitts-neuron:

1. ورودی‌ها (x): یک نورون مک‌کولوخ-پیتس سیگنال‌های ورودی چندگانه را دریافت می‌کند که به ترتیب به صورت نشان داده می‌شوند و هر کدام با یک وزن مرتبط است.

2. وزن‌ها (w): هر سیگنال ورودی با وزن متناظر ضرب می‌شود. وزن‌ها نشان‌دهنده قدرت یا اهمیت سیگنال‌های ورودی برای نورون هستند.

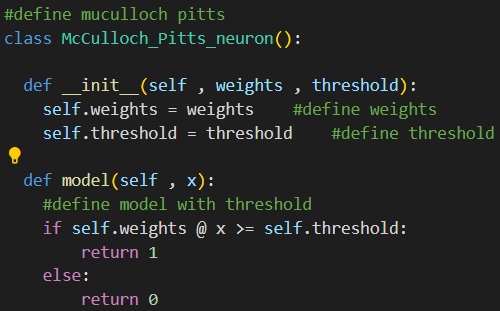
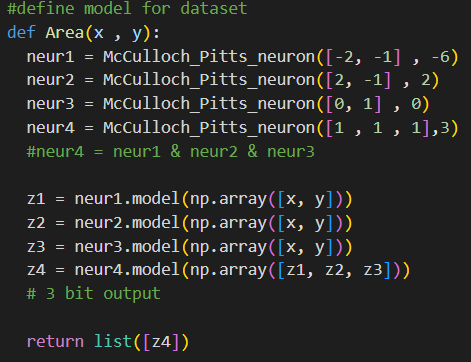
3. آستانه (θ): نورون دارای یک مقدار آستانه است. این آستانه جمع ورودی‌های وزن‌دار را محاسبه کرده و نتیجه را با آستانه مقایسه می‌کند.

4. تابع فعال‌سازی: تابع فعال‌سازی نورون یک تابع آستانه است. اگر مجموع ورودی‌های وزن‌دار بیشتر از آستانه باشد، نورون "آتش می‌زند" یا سیگنال خروجی تولید می‌کند؛ در غیر این صورت، غیرفعال می‌ماند.

نمایش ریاضی:

با دریافت ورودی‌های و وزن‌های متناظر فعال‌سازی z یک نورون McCulloch-Pitts به شکل زیر محاسبه می‌شود:

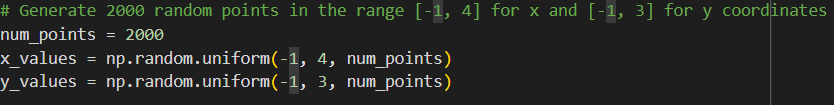
CODE :



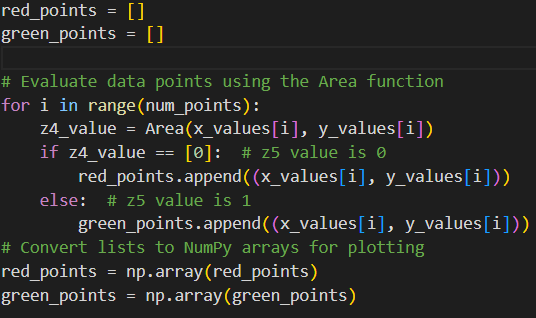
طبق شکل سوال نقاط مثلث داده شده شامل:

خطوط AB و BC و AC را طبق معادله زیر بدست می آوریم:

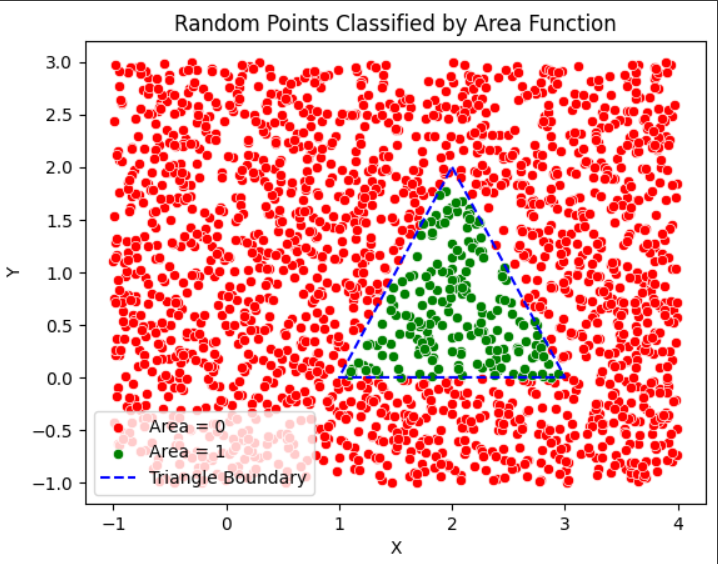
که وزن های نرون ها و آستانه را مشخص میکنند تا موقعیت نسبت به خط ها مشخص شود و نرون 4 ام مشخص می کند که نقطه ی تست شده هر سه شرط را رعایت میکند.



حال با دستور بالا 2000 نقطه با توجه به محدوده خواسته شده سوال تولید می کنیم .



نقاط درون مثلث با value 1 و رنگ سبز ونقاط بیرون با value 0 و رنگ قرمز نمایش میدهیم:

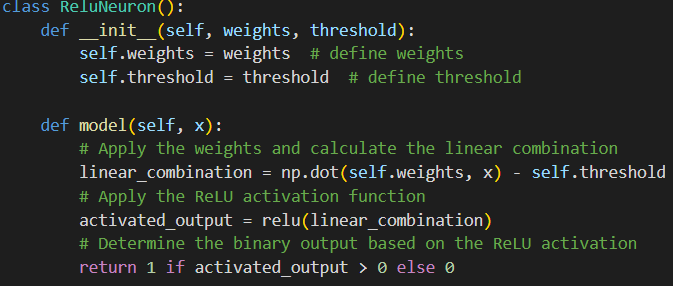


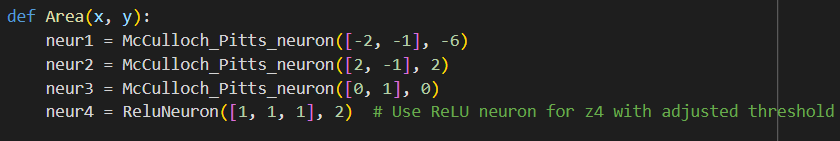
اضافه کردن تابع فعال ساز به مرحله تصمیم گیری :

ما تابع فعال‌سازی ReLU و sigmoid را برای تابع فعالساز انتخاب کردیم

### ReLU:







تابع فعالساز ReLU برای ورودی‌های منفی 0 را برمی‌گرداند و برای ورودی‌های غیر منفی، خود ورودی را برمی‌گرداند.

یک کلاس به نام ReluNeuron ایجاد کردیم تا نورون نهایی با استفاده از فعال‌سازی ReLU را نشان دهد.

در تابع Area، پیاده‌سازی قبلی نورون نهایی را با ReluNeuron جایگزین کردیم.

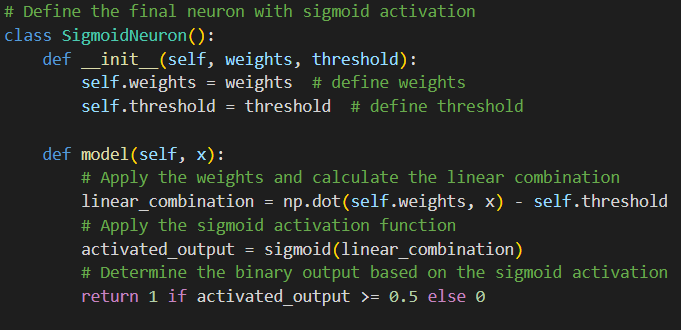
تنظیماتی انجام دادیم تا اطمینان حاصل شود که طبقه‌بندی به درستی انجام می‌شود، از جمله تنظیم آستانه برای نورون نهایی به 2.

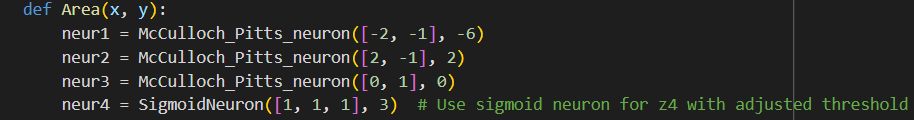
کلاس McCulloch\_Pitts\_neuron نورون‌هایی را نشان می‌دهد که با منطق مبتنی بر آستانه کار می‌کنند.

روش model ترکیب خطی ورودی‌ها را محاسبه کرده و آستانه را اعمال می‌کند تا خروجی دودویی تولید کند.

فعال‌سازی ReLU به صورت خارجی، در تابع Area، برای فرآیند تصمیم‌گیری نهایی اعمال می‌شود.

### Sigmoid :





تابع فعالساز Sigmoid یک تابع غیر خطی است و مقدار خروجی آن بین 0 و 1 قرار می‌گیرد.

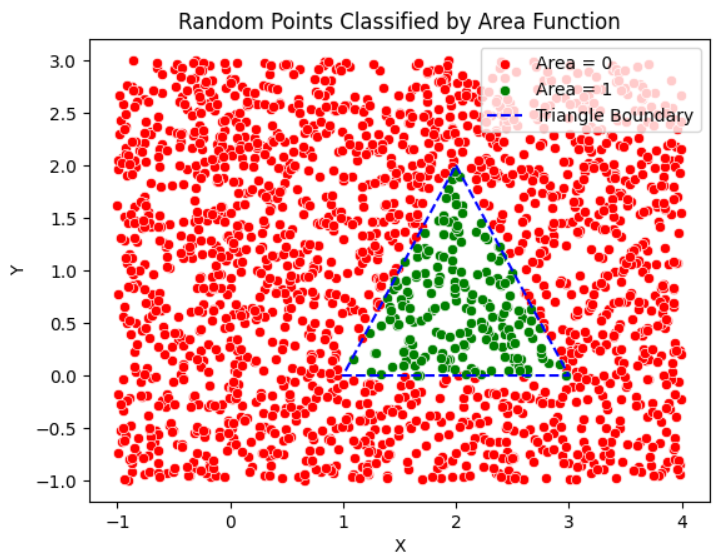
مانند قبل بجای Relu از Sigmoid استفاده می کنیم .

تنظیماتی انجام دادیم تا اطمینان حاصل شود که طبقه‌بندی به درستی انجام می‌شود، از جمله تنظیم آستانه برای نورون نهایی به 3.

کلاس McCulloch\_Pitts\_neuron نورون‌هایی را نشان می‌دهد که با منطق مبتنی بر آستانه کار می‌کنند.

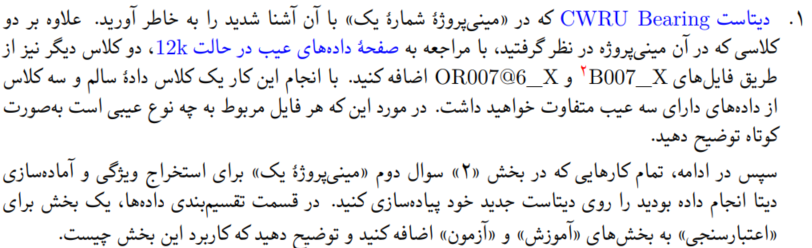
روش model ترکیب خطی ورودی‌ها را محاسبه کرده و آستانه را اعمال می‌کند تا خروجی دودویی تولید کند.

فعال‌سازی Sigmoid به صورت خارجی، در تابع Area، برای فرآیند تصمیم‌گیری نهایی اعمال می‌شود.



# سوال 2

## 1 .



داده‌های خرابی را برای بلبرینگ سراندار از یک مجموعه داده استخراج کرده‌اید که با نرخ نمونه‌برداری ۱۲ هزار نمونه در ثانیه جمع‌آوری شده است. داده‌های خاص خرابی که شما استخراج کرده‌اید عبارتند از:

داده خرابی محفظه بیرونی:

1. موقعیت نسبت به منطقه بارگذاری: متمرکز

منطقه بارگذاری متمرکز در ۶:۰۰: این نشان می‌دهد که خرابی در محفظه بیرونی بلبرینگ قرار دارد و در موقعیت ۶:۰۰ نسبت به منطقه بارگذاری متمرکز است. این موقعیت می‌تواند بر روی نحوه‌ای که خرابی بر داده‌های ارتعاشی تأثیر می‌گذارد، تأثیرگذار باشد.

1. داده خرابی توپ:

این نشان می‌دهد که خرابی در یکی از عناصر چرخان (توپ‌ها) بلبرینگ وجود دارد.

1. داده خرابی محفظه داخلی:

این نشان می‌دهد که خرابی در محفظه داخلی بلبرینگ وجود دارد.

درک مکان‌های خرابی:

خرابی محفظه بیرونی: خرابی‌ها در محفظه بیرونی ثابت هستند و حرکت نسبی بین محفظه بیرونی و عناصر چرخان منجر به ضربات دوره‌ای می‌شود که الگوهای ارتعاشی مشخصی را ایجاد می‌کند.

خرابی توپ: خرابی‌ها در عناصر چرخان (توپ‌ها) به ضربات منجر می‌شوند زیرا توپ‌ها بر روی محفظه‌های داخلی و بیرونی حرکت می‌کنند و الگوهای ارتعاشی خاصی تولید می‌کنند.

خرابی محفظه داخلی: خرابی‌ها در محفظه داخلی تحت تأثیر سرعت چرخشی شافت قرار دارند. هنگامی که محفظه داخلی دوران می‌کند، باعث ضربات دوره‌ای با عناصر چرخان می‌شود و الگوهای ارتعاشی منحصر به فردی تولید می‌کند.

فایل‌های داده به فرمت متلب است. هر فایل شامل داده‌های ارتعاشات سمت فن و درایو همراه با سرعت چرخش موتور است. برای تمام فایل‌ها، مورد زیر در نام متغیر نشان می‌دهد:

DE - داده‌های شتاب‌سنج سمت درایو

FE - داده‌های شتاب‌سنج سمت فن

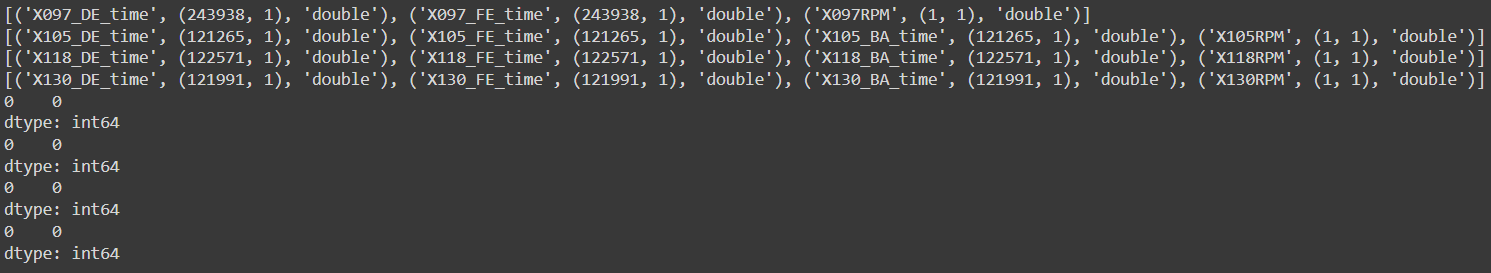
BA - داده‌های شتاب‌سنج پایه

time - داده‌های سری زمانی

RPM - دور در دقیقه در طول آزمایش

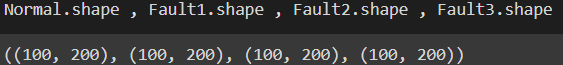
داده‌های پایه نرمال ، داده‌های عیب بلبرینگ سمت درایو با نرخ 12k ، داده‌های عیب بلبرینگ ، سمت درایو با نرخ 48k ، داده‌های عیب بلبرینگ سمت فن

با توجه به خواسته های سوال از داده های نرمال با سرعت (1797rmp) وداده های فالت با قطر‌های ۰.۰۰۷ اینچ مسیر محفظه داخلی،خارجی و توپ استفاده میکنیم .



در اینجا فیچر های مرتبط نوشته شده و بنا بر خواسته ی مساله ما فقط ازDE\_time استفاده کردیم.

تابعی تعریف می کنیم که دیتارا گرفته آن را بر بزند و نمونه با طول N جدا کند . سپس این تابع را به دیتاست اعمال می کنیم و از برای هرکلاس یک ماتریس 100\*200 تشکیل میدهیم .



فیچر های ذکر شده در کتابخانه های numpy و scypy موجود هستند آنها را درون یک کلاس تعریف میکنیم. و به ماتریس های قبل اعمال می کنیم حال یک داده با 14 feature و 400 نمونه داریم .همچنین داده ها را با تعریف تابعی لیبل می زنیم:



لیبل ها به شکل زیر در میآیند :



داده ها را با استفاده از کتاب خانه sklearn به آموزش ، تست و ارزیابی تقسیم می کنیم و بر میزنیم.

و همچنین آن را استاندارد سازی میکنیم.

در قسمت تقسیم بندی داده‌ها، اضافه کردن ارزیابی به آموزش و تست می‌تواند بسیار مفید باشد. این بخش ارزیابی به عنوان مرحله‌ای میانی بین آموزش و تست عمل می‌کند و کاربردهای مختلفی دارد:

1. ارزیابی عملکرد مدل:

در مرحله آموزش، مدل با استفاده از داده‌های آموزش آموزش داده می‌شود. سپس، برای ارزیابی عملکرد مدل و جلوگیری از بیش‌برازش یا کم‌برازش، می‌توان از داده‌های ارزیابی استفاده کرد. این داده‌ها به عنوان یک مجموعه میانی برای ارزیابی دقت و کارایی مدل در حالتی که با داده‌های آموزش کار نکرده است، استفاده می‌شوند.

2. انتخاب مدل بهتر:

از داده‌های ارزیابی می‌توان برای انتخاب بهترین مدل از بین چندین مدل آموزش داده شده استفاده کرد. با ارزیابی عملکرد هر مدل با استفاده از داده‌های اعتبارسنجی، می‌توان مدلی را که بهترین عملکرد را دارد، انتخاب کرد.

3. تنظیم پارامترها:

برخی از مدل‌ها دارای پارامترهایی هستند که نیاز به تنظیم دقیق دارند. از داده‌های ارزیابی می‌توان برای تنظیم بهینه این پارامترها با استفاده از تکنیک‌هایی مانند ارزیابی متقابل (cross-validation) استفاده کرد.

4. ارزیابی مدل:

استفاده از داده‌های ارزیابی به ما اجازه می‌دهد تا مطمئن شویم که مدل آموزش داده شده قادر به عملکرد مناسب در داده‌های جدید (تست) خواهد بود. این ارزیابی اطمینان ما را از کارایی عملیاتی مدل در موارد واقعی افزایش می‌دهد.

## 2 .



برای استفاده از شبکه عصبی از کتاب خانه Tensorflow.keras استفاده می کنیم .

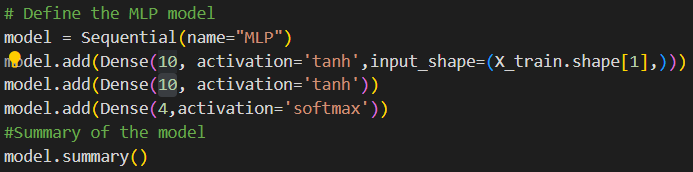
MLP :

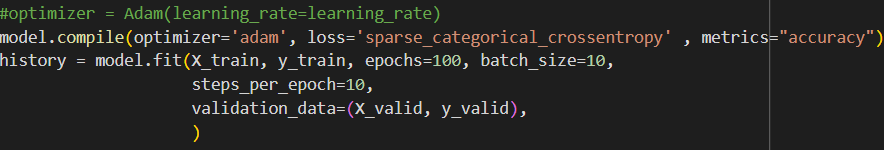
برای این بخش با 2 تابع فعال ساز 2 تابع اتلاف و2 بهینه ساز مختلف استفاده می کنیم

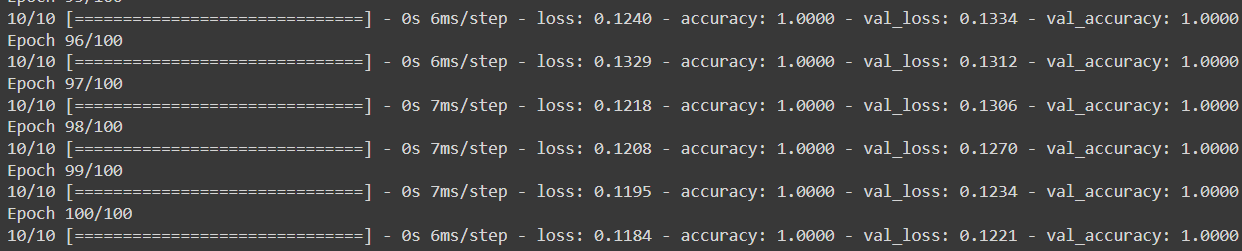
### 1)

از MLP دولایه با تابع فعال ساز tanh برای دو لایه پنهان و softmax برای لایه خروجی

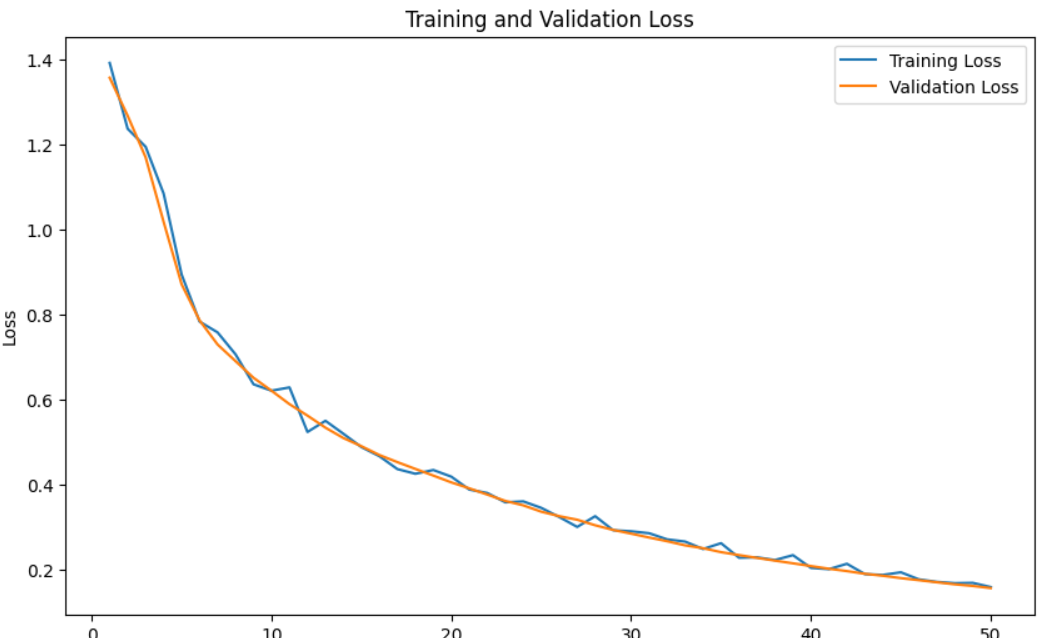
استفاده کرده ایم . همچنین تابع اتلاف Sparse-categorical-cross-entropy که برای لیبل های integer استفاده می شود و بهینه ساز ‘adam’ استفاده کرده ایم:

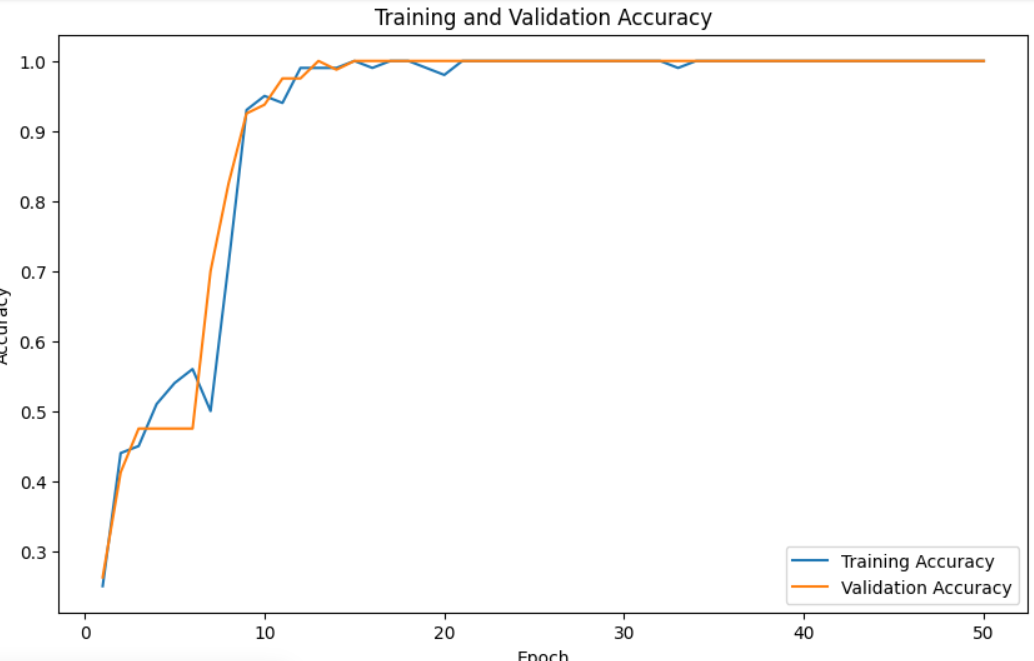






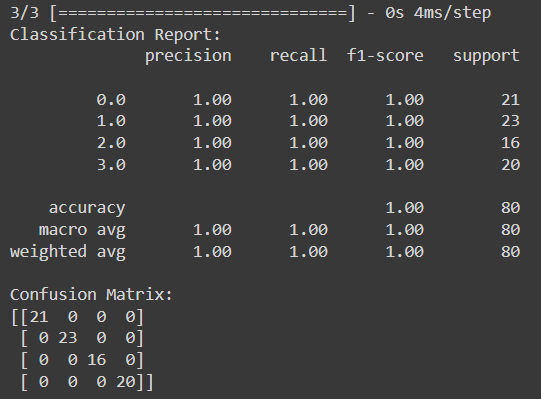
نمودار Loss و Accuracy :





همانطور که معلوم است دقت برای داده های ارزیابی و آموزش 100 درصد درآمده و اتلاف 0.157 و 0.159 شده است که بسیار مطلوب است .

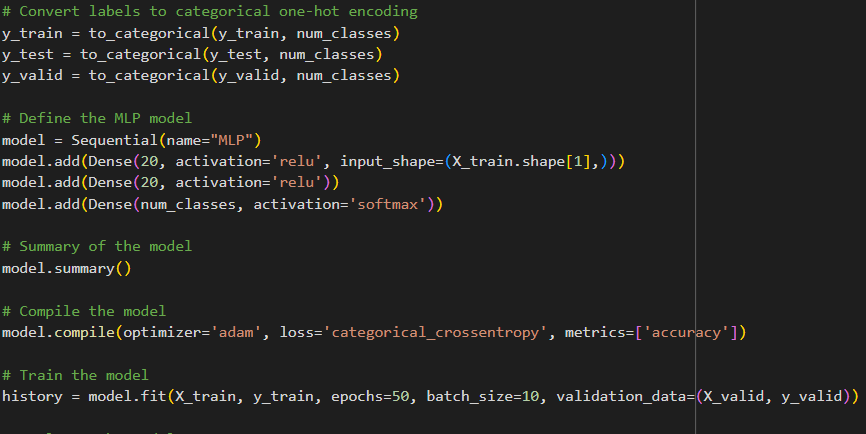
همچنین classification\_report و confusion\_matrix به صورت زیر درآمده اند:

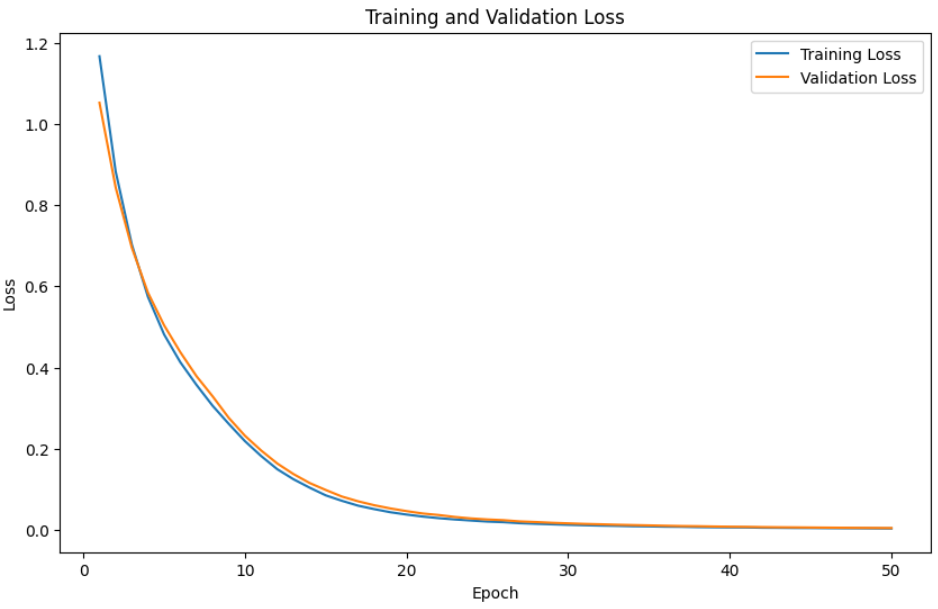


دقت با معیار های مختلف سنجیده شده که می بینیم در همه 100% شده اند و همچنین با توجه به confusion\_matix میبینیم که هیچ داده ای missclassified نشده و تماما در قطر اصلی هستند.

### 2)ReLU :

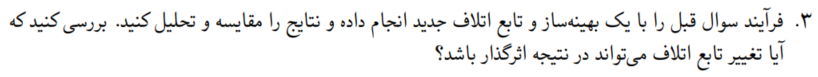
با استفاده از 2 لایه پنهان با فعالساز ‘ReLU’ و تابع اتلاف ‘categorical\_crossentropy’ برای لیبل های ‘One-hot’ شده و بهینه ساز ‘adam’ استفاده میکنیم:



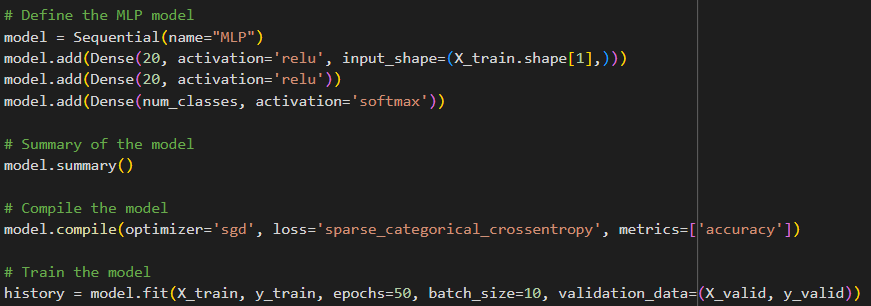


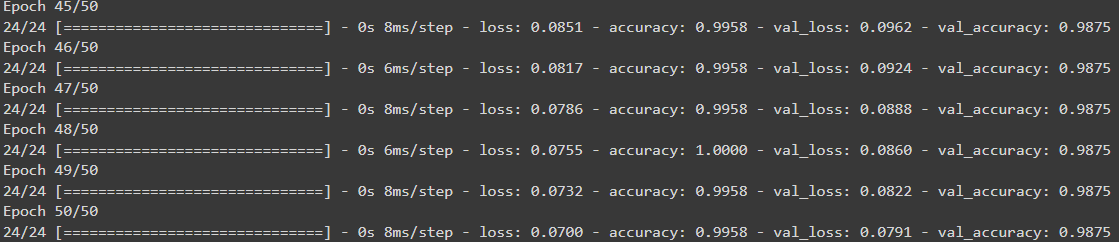


## 3.

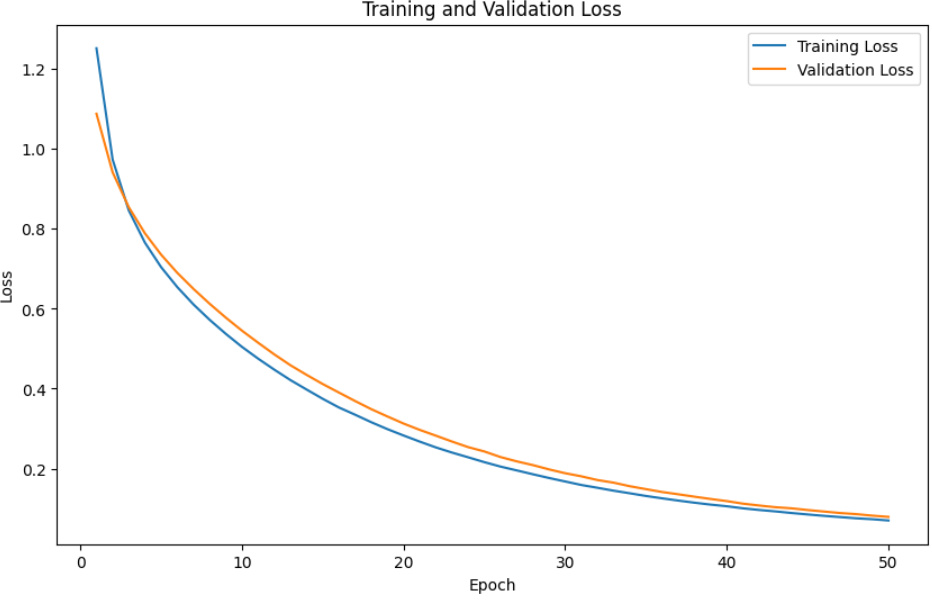


حال از تابع اتلاف ‘sparse\_categorical\_crossentropy’ و بهینه ساز ‘sgd’ استفاده میکنیم:





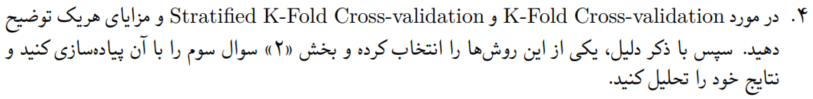






انتخاب تابع اتلاف و بهینه ساز مناسب بسیار مهم می باشد .

## 4.



K-Fold Cross-Validation :

K-Fold Cross-Validation یک روش نمونه‌گیری مجدد است که برای ارزیابی مدل‌های یادگیری ماشین بر روی مجموعه داده‌های محدود استفاده می‌شود. این روش به پارامتری به نام K وابسته است که به تعداد بخش‌هایی که مجموعه داده‌ها به آن تقسیم می‌شود اشاره دارد.

روش کار به این صورت است:

1. مجموعه داده‌ها به صورت تصادفی مرتب می‌شوند.

2. مجموعه داده‌ها به K بخش (یا fold) تقسیم می‌شوند.

3. برای هر بخش منحصر به فرد:

- آن بخش را به عنوان مجموعه داده تست انتخاب کنید.

- بقیه بخش‌ها را به عنوان مجموعه داده آموزشی انتخاب کنید.

- مدل را بر روی مجموعه آموزشی آموزش داده و بر روی مجموعه تست ارزیابی کنید.

- نمره ارزیابی را نگه داشته و مدل را کنار بگذارید.

4. میانگین نمرات ارزیابی را به عنوان معیار نهایی عملکرد مدل استفاده کنید.

:Stratified K-Fold Cross-Validation

Stratified K-Fold Cross-Validation یک نوع خاص از K-Fold Cross-Validation است که تضمین می‌کند هر بخش نمایانگر نسبت صحیحی از هر کلاس هدف باشد. این روش به ویژه در مسائل طبقه‌بندی با توزیع نامتعادل کلاس‌ها مفید است.

روش کار به این صورت است:

1. مجموعه داده‌ها به صورت تصادفی مرتب می‌شوند.

2. مجموعه داده‌ها به K بخش تقسیم می‌شوند، اما این تقسیم به صورت لایه‌ای انجام می‌شود تا هر بخش تقریباً درصدی از نمونه‌های هر کلاس هدف را داشته باشد که با مجموعه کامل برابر است.

3. برای هر بخش منحصر به فرد:

- آن بخش را به عنوان مجموعه داده تست انتخاب کنید.

- بقیه بخش‌ها را به عنوان مجموعه داده آموزشی انتخاب کنید.

- مدل را بر روی مجموعه آموزشی آموزش داده و بر روی مجموعه تست ارزیابی کنید.

- نمره ارزیابی را نگه داشته و مدل را کنار بگذارید.

4. میانگین نمرات ارزیابی را به عنوان معیار نهایی عملکرد مدل استفاده کنید.

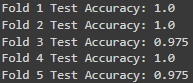
انتخاب بین K-Fold و Stratified K-Fold Cross-Validation

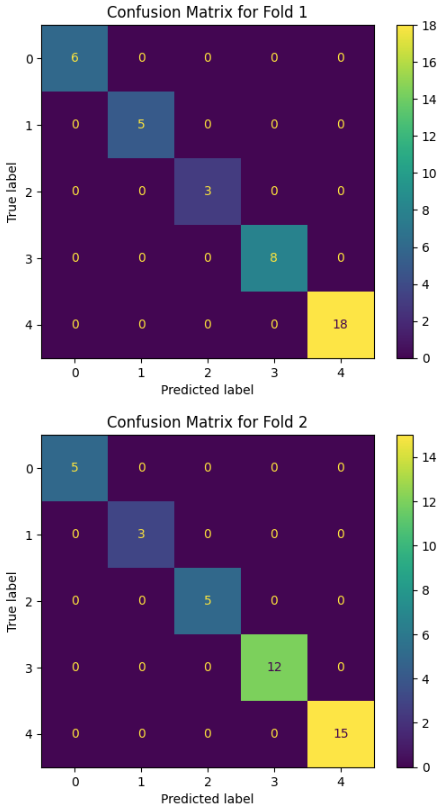
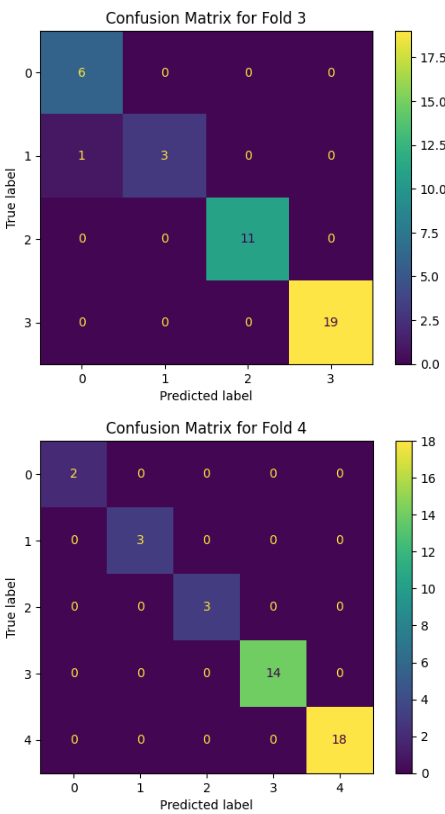
در مسائل طبقه‌بندی، به ویژه زمانی که کلاس‌ها نامتعادل هستند، بهتر است از Stratified K-Fold Cross-Validation استفاده شود. Stratified K-Fold تضمین می‌کند که هر بخش نمایانگر توزیع کلاس‌های داده باشد، که این کار ارزیابی عملکرد مدل را دقیق‌تر و پایدارتر باشد.

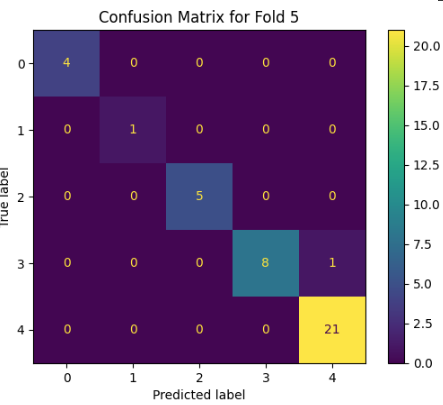
با استفاده از Stratified K-Fold Cross-Validation، اطمینان حاصل می‌شود که ارزیابی مدل شما قوی‌تر و کمتر در معرض تغییرات ناشی از عدم تعادل کلاس‌ها قرار می‌گیرد، که منجر به ارزیابی عملکرد قابل اطمینان‌تری می‌شود.

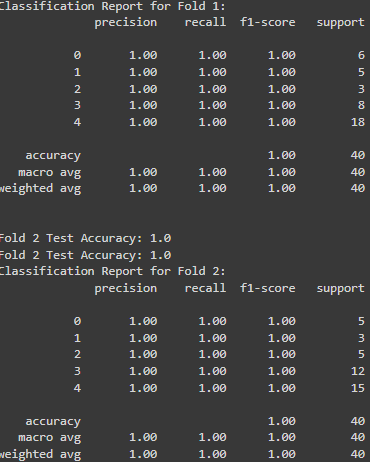
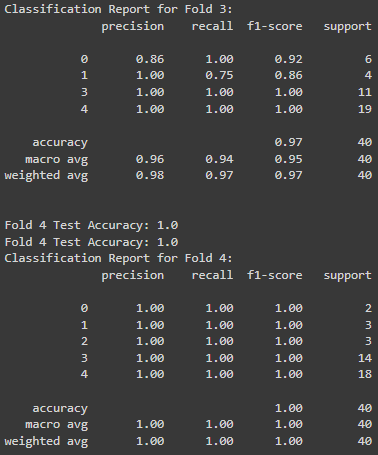
برای دیتاست سوال 3 k\_fold و stratified k-fold زده شده:

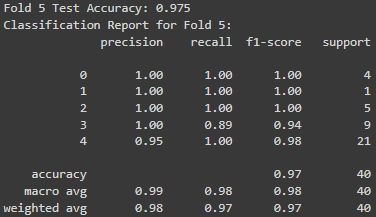
### k-fold







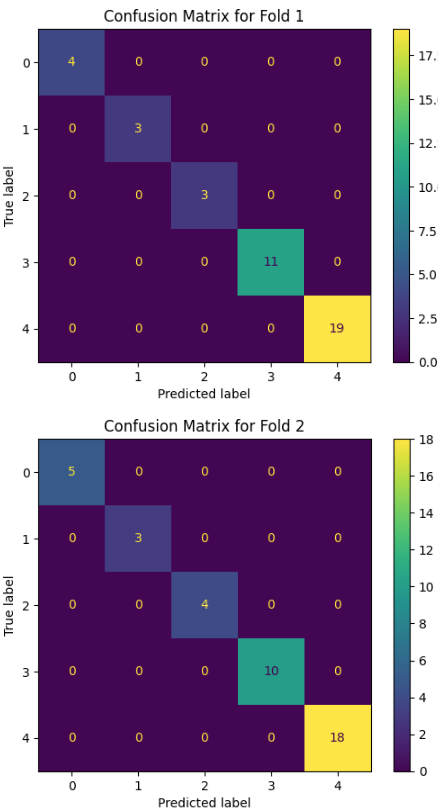
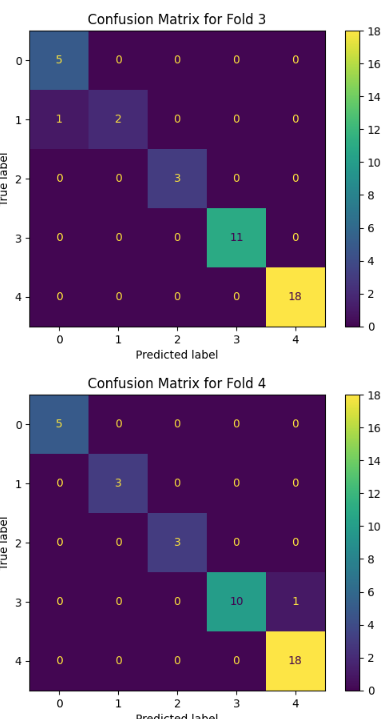


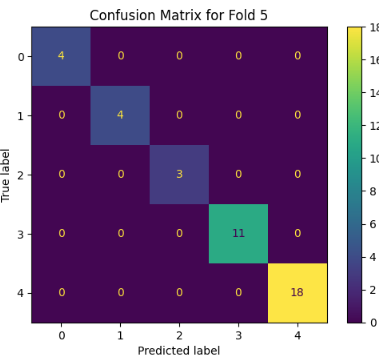


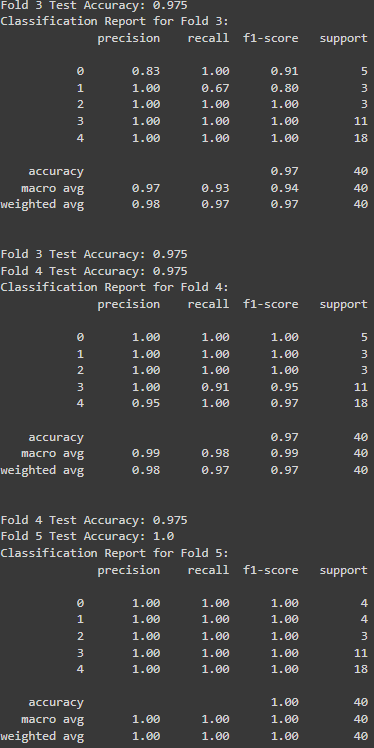
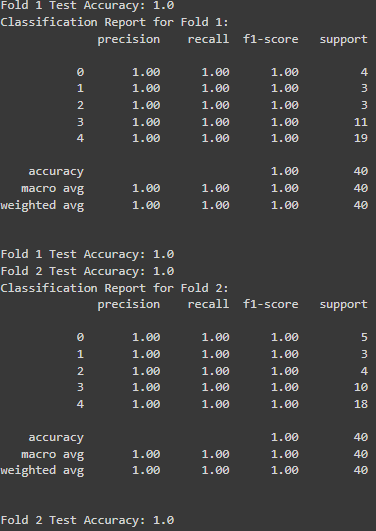


در نتایج آورده شده دقت میانگین نزدیک به 100% است که بسیار مطلوب است.

### Statified K-fold :









در نتایج آورده شده دقت میانگین نزدیک به 100% است که بسیار مطلوب است.

# 3.

در اینجا دو دیتاست طبقه بندی جنگلی و دارو معرفی شده اند

## دیتاست دارو

### توضیحات دیتاست

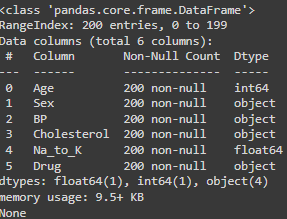
فرض کنید شما یک پژوهشگر پزشکی هستید که در حال جمع‌آوری داده‌ها برای یک مطالعه می‌باشد. شما داده‌هایی درباره‌ی مجموعه‌ای از بیماران جمع‌آوری کرده‌اید که همه‌ی آن‌ها از یک بیماری یکسان رنج می‌بردند. در طول دوره درمان، هر بیمار به یکی از ۵ دارو پاسخ داده است: داروی A، داروی B، داروی C، داروی X و Y.

وظیفه‌ی ما این است که مدلی بسازیم تا مشخص کند کدام دارو ممکن است برای یک بیمار آینده‌ای که به همان بیماری مبتلا است مناسب باشد. ویژگی‌های این مجموعه داده شامل سن، جنسیت،نسبت سدیم-پتاسیم، فشار خون و کلسترول بیماران است و هدف (متغیر خروجی) دارویی است که هر بیمار به آن پاسخ داده است.

این یک نمونه از طبقه‌بندی چندکلاسه است و ما می‌توانیم از بخش آموزشی مجموعه داده برای ساخت یک مدل درخت تصمیم‌گیری استفاده کنیم و سپس از آن برای پیش‌بینی کلاس یک بیمار ناشناخته یا تجویز دارو به یک بیمار جدید استفاده کنیم.

ویژگی ها:



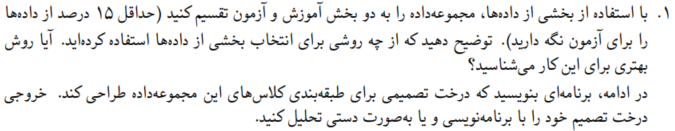


داده های null نداریم و نوع دیتاهای مختلف آورده شده که مشاهده می کنیم داده سن از نوع int64 داده نسبت سدیم-پتاسیم از نوع float64 و بقیه object هستند.تعداد ارزش های unique داده های object به گونه زیر می باشد:



داده Drug خروجی یا target ما می باشد که همانطور که میبینیم 5 value مختلف می تواند داشته باشد ، یعنی 5 کلاس داریم.

### 1 . پیش پردازش



داده های از نوع متغیرهای دسته‌بندی Sex (جنسیت)، BP (فشار خون)، Cholesterol (کلسترول) و Drug (دارو) را با استفاده از LabelEncoder به مقادیر عددی تبدیل می‌کنیم.

ویژگی‌ها (Sex, BP, Cholesterol, Na\_to\_K, Age) را به عنوان X و برچسب‌ها (Drug) را به عنوان y تعریف می‌کند.

داده‌ها را به مجموعه‌های آموزشی (70٪) و آزمایشی (30٪) تقسیم می‌کنیم.



و سپس آن هارا نرمال سازی کردیم.



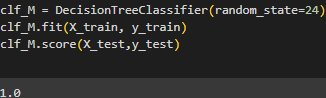
برای تقسیم مجموعه داده به دو بخش آموزش و آزمون، از روش‌های مختلفی می‌توان استفاده کرد. یکی از رایج‌ترین روش‌ها، استفاده از تابع train\_test\_split از کتابخانه sklearn است که به راحتی داده‌ها را به دو بخش تقسیم می‌کند. این تابع به طور تصادفی داده‌ها را تقسیم می‌کند، که به کاهش بایاس کمک می‌کند.در این مثال، از ۳۰٪ داده‌ها برای آزمون و ۷۰٪ برای آموزش استفاده شده است. این نسبت معمولاً به عنوان یک استاندارد خوب شناخته می‌شود، اما برای مسائل خاص، می‌توان نسبت‌های متفاوتی استفاده کرد.با استفاده از پارامتر random\_state، می‌توانیم تقسیم داده‌ها را بازتولید کنیم، به طوری که در هر اجرا همان نتایج را بدست آوریم.

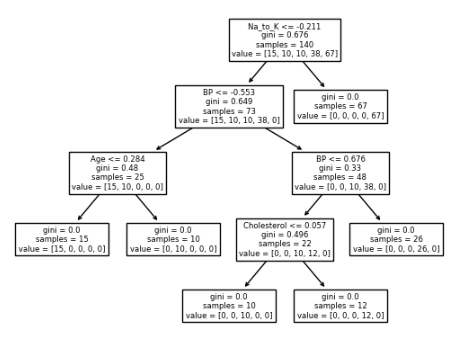
روش‌های جایگزین:

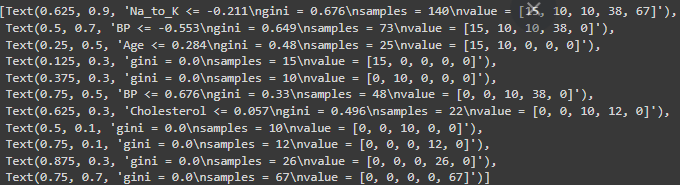
K-Fold Cross-Validation: در این روش، داده‌ها به K بخش تقسیم می‌شوند و مدل K بار آموزش داده می‌شود، هر بار یک بخش به عنوان مجموعه آزمون و بقیه به عنوان مجموعه آموزش در نظر گرفته می‌شوند. این روش دقت مدل را بهبود می‌بخشد زیرا از تمام داده‌ها برای آموزش و آزمون استفاده می‌کند.

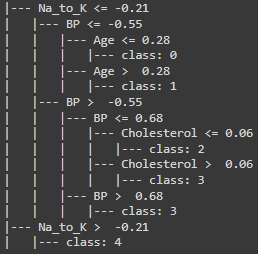
Stratified Sampling: در صورتی که داده‌ها دارای کلاس‌های نامتوازن باشند، بهتر است از نمونه‌برداری طبقه‌ای استفاده کنیم تا نسبت کلاس‌ها در مجموعه‌های آموزش و آزمون مشابه باشد. این کار می‌تواند با استفاده از پارامتر stratify در train\_test\_split انجام شود.

حال با دستور زیر طبقه بند درخت تصمیم را تعریف می کنیم:









1. Root node :



با توجه به خروجی های آورده شده بهترین feature انتخاب شده برای گره اول (root node) ویژگی Na to K است که با شرط Na\_to\_K <= -0.211 به دو دسته تقسیم می شود

معیار Gini Impurity است که برای گره اول عدد 0.676 است که نشان دهنده ناخالصی بالا می باشد مطلوب است که این عدد به صفر برسانیم.

هم چنین 140نمونه در این گره وجود دارند و توزیع کلاس این گره [15, 10, 10, 38, 67] می باشد.

1. First Level Split (Left Branch) :



این تقسیم مربوط به ویژگی فشار خون (bp) است که با شرط BP <= -0.553 و معیار ناخالصی Gini 0.649 و تعداد نمونه و توزیع کلاس [15, 10, 10, 38, 0] انجام شده.

1. Second Level Split (Left-Left Branch) :



این تقسیم که مربوط با ویژگی سن و شرط Age <= 0.284 تقسیم داده کرده .تعداد نمونه ها در این گره 25 و میار ناخالصی Gini برابر 0.48 می باشد که به معنی میزان ناخالصی بالا می باشد و تقسیم کلاس [15, 10, 0, 0, 0] می باشد .

1. Third Level Split (Left-Left-Left Branch) :



این تقسیم معیار ناخالصی Gini را به صفرمیرساند یعنی خلوص کلاس کامل است و وکلاس اول را مجزا میکند .تعداد داده ها برابر15 وکلاس به شکل [15, 0, 0, 0, 0] می باشد.

1. Third Level Split (Left-Left-Right Branch) :



همچنین در این تقسیم کلاس دوم مجزا میشود تعداد نمونه ها 10 و تقسیم کلاس [0, 10, 0, 0, 0] او خلوص کلاس کامل است .

1. Second Level Split (Left-Right Branch) :



در این تقسیم با شرط BP <= 0.676 و معیار Gini 0.33 تعداد48 داده با تقسیم کلاس

[0, 0, 10, 38, 0] داریم .

1. Third Level Split (Left-Right-Left Branch):



در این تقسیم با شرط Cholesterol <= 0.676 و معیار Gini 0.496 تعداد 22 داده با تقسیم کلاس [0, 0, 10, 12, 0]داریم .

1. Fourth Level Split (Left-Right-Left-Left Branch) :



کلاس سوم در اینجا مجزا می شود. تعداد نمونه ها 10 وتقسیم کلاس به شکل [0, 0, 10, 0, 0] و خلوص کامل کلاس .

1. Fourth Level Split (Left-Right-Left-Right Branch) :



همچنین تمام نمونه ها با تعداد12 به کلاس چهارم با معیار gini 0 توزیع کلاس

[0, 0, 0, 12, 0] و خلوص کامل کلاس .

1. Third Level Split (Left-Right-Right Branch) :



همچنین کلاس چهارم در اینجا کامل شده با تعداد نمونه 26 و توزیع کلاس به شکل

[0, 0, 0, 26, 0] و خلوص کامل کلاس .

1. First Level Split (Right Branch) :



در این گره کلاس پنجم مجزا شده با تعداد داده 67 و شکل توزیع کلاس [0, 0, 0, 0, 67] و خلوص کامل کلاس .

توزیع کلاس در گره‌های برگ:

کلاس 0 (داروی A): عمدتاً در گره‌هایی با شرایط Na\_to\_K <= -0.211، BP <= -0.553، و Age <= 0.284 مشاهده می‌شود.

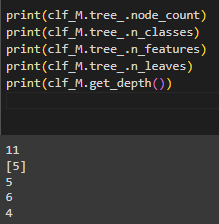
کلاس 1 (داروی B): در گره‌هایی با شرایط Na\_to\_K <= -0.211، BP <= -0.553، و Age > 0.284 دیده می‌شود.

کلاس 2 (داروی C): در گره‌هایی با شرایط Na\_to\_K <= -0.211، BP > -0.553، و Cholesterol <= 0.057 مشاهده می‌شود.

کلاس 3 (داروی X): در گره‌هایی با شرایط Na\_to\_K <= -0.211، BP > -0.553، و Cholesterol > 0.057 مشاهده می‌شود.

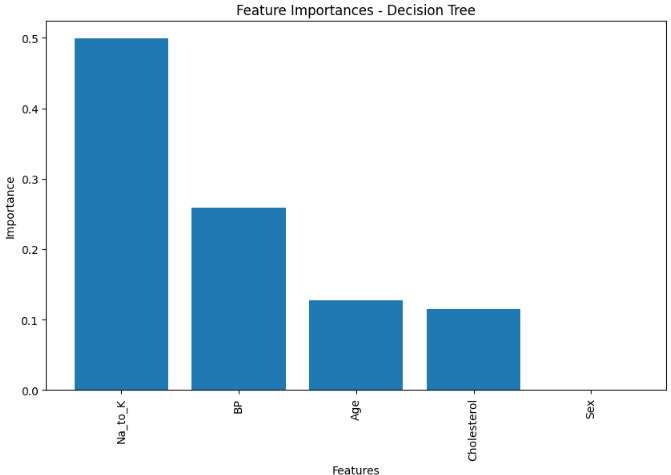
کلاس 4 (داروی Y): در گره‌هایی با شرایط Na\_to\_K > -0.211 مشاهده می‌شود.

این تحلیل نشان می‌دهد که مدل چگونه پیش‌بینی‌ها را انجام می‌دهد و کدام ویژگی‌ها بیشترین تأثیر را در تعیین کلاس دارند. حضور گره‌های برگ خالص نشان می‌دهد که مدل برای برخی کلاس‌ها به خوبی عمل می‌کند، در حالی که ناخالصی جینی بالا در برخی گره‌ها نشان‌دهنده نواحی است که مدل ممکن است نیاز به بهبود بیشتر داشته باشد یا کلاس‌ها بیشتر مخلوط باشند.

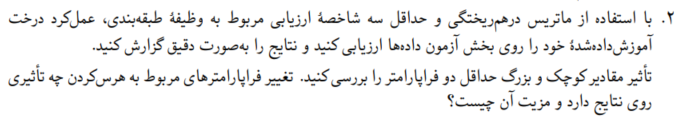


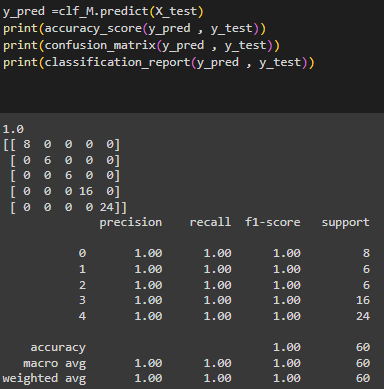
تعداد گره ها 11 ، تعداد کلاس ها 5 تعداد ویژگی ها 6 تعداد برگ ها 6 و عمق (سطح) 4 است .

همچنین اهمیت ویژگی ها به ترتیب:



## 2.





میبینیم که confusion matrix قطری شده و هیچ داده ای misclassified نشده و همچنین با classification\_report می بینیم که با معیار ارزیابی f1\_score و recall و precision وaccuracy دقت 100% شده.

تغییر 2 فرا پارامتر (max\_depth) و (min\_sample\_leaf)

**1. حداکثر عمق درخت (Max Depth):**

الف. عمق کم (مقدار کوچک):

مزایا:

جلوگیری از بیش‌برازش (Overfitting): وقتی عمق درخت کوچک است، مدل به خوبی تعمیم پیدا می‌کند و از یادگیری بیش از حد جزئیات داده‌های آموزش جلوگیری می‌شود.

سرعت بیشتر: مدل سریع‌تر آموزش می‌بیند و زمان پیش‌بینی نیز کاهش می‌یابد.

معایب:

کمتر دقیق: ممکن است مدل نتواند الگوهای پیچیده در داده‌ها را به درستی یاد بگیرد و دقت کمتری داشته باشد.

ب. عمق زیاد (مقدار بزرگ):

مزایا:

دقت بیشتر: مدل می‌تواند الگوهای پیچیده‌تری را یاد بگیرد و دقت بالاتری در داده‌های آموزش داشته باشد.

معایب:

بیش‌برازش: مدل ممکن است بیش از حد به داده‌های آموزش حساس شود و در داده‌های جدید عملکرد خوبی نداشته باشد.

افزایش زمان محاسبه: زمان آموزش و پیش‌بینی افزایش می‌یابد.

**2. حداقل نمونه‌ها در یک برگ (Min Samples Leaf):**

الف. مقدار کم:

مزایا:

دقت بیشتر: مدل می‌تواند جزئیات بیشتری از داده‌ها را یاد بگیرد و دقت بالاتری داشته باشد.

معایب:

بیش‌برازش: مدل ممکن است بیش از حد به داده‌های آموزش حساس شود و عملکرد خوبی در داده‌های جدید نداشته باشد.

پیچیدگی بیشتر: ساختار درخت پیچیده‌تر می‌شود و زمان آموزش و پیش‌بینی افزایش می‌یابد.

ب. مقدار زیاد:

مزایا:

جلوگیری از بیش‌برازش: با افزایش حداقل نمونه‌ها در یک برگ، مدل تعمیم‌پذیری بهتری پیدا می‌کند.

کاهش پیچیدگی: ساختار درخت ساده‌تر می‌شود و زمان آموزش و پیش‌بینی کاهش می‌یابد.

معایب:

کمتر دقیق: مدل ممکن است نتواند الگوهای جزئی‌تر را یاد بگیرد و دقت کمتری داشته باشد.

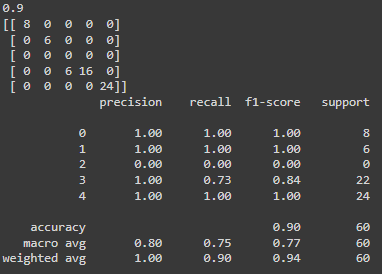
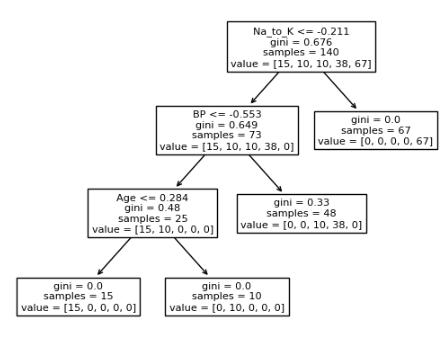
3 **.تغییر فراپارامترهای مربوط به هرس کردن و تأثیر آنها**

***ccp\_alpha***

مقداردهی به ccp\_alpha به مدل این امکان را می‌دهد که تعادل مناسبی بین سادگی و دقت را برقرار کند. مقدار بزرگتر ccp\_alpha منجر به هرس کردن بیشتر و یک درخت ساده‌تر می‌شود که ممکن است دقت کمتری داشته باشد اما از بیش‌برازش جلوگیری می‌کند. از طرف دیگر، مقدار کمتر ccp\_alpha منجر به حفظ ساختار بیشتر درخت و افزایش دقت روی داده‌های آموزش می‌شود، اما این امر ممکن است باعث بیش‌برازش شود.

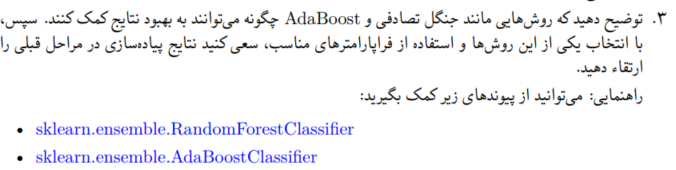
با تغییر فراپارامتر های ccp\_alpha و max\_depth داریم:





می بینیم که دقت پایین آمده و کلاس سوم هرس شده است و کلاس بندی نشده است.

## 3.



روش‌هایی مانند جنگل تصادفی و AdaBoost می‌توانند بهبود نتایج در مسائل یادگیری ماشینی کمک کنند به دلایل زیر:

۱. جنگل تصادفی (Random Forest):

- تنوع بالا: جنگل تصادفی از چندین درخت تصمیم تشکیل شده است که هر کدام به صورت مستقل و به‌طور تصادفی از داده‌ها به‌طور تکراری نمونه‌گیری می‌کنند و درخت‌های مختلفی را تولید می‌کنند. این تنوع باعث می‌شود که مدل از افرازها و الگوهای مختلفی در داده استفاده کند.

- کاهش بیش‌برازش: به دلیل تنوع بالا و تصادفی‌سازی در فرآیند آموزش، جنگل تصادفی به خوبی از بیش‌برازش جلوگیری می‌کند و دارای عملکرد خوبی در داده‌های جدید است.

- انعطاف‌پذیری: جنگل تصادفی می‌تواند با مقادیر پیش‌فرض خوبی به خوبی کار کند و نیازی به تنظیم پارامترهای پیچیده ندارد.

- قابلیت استفاده از ویژگی‌های مختلف: این روش به راحتی می‌تواند با ویژگی‌های مختلف ساختاری و غیرساختاری سازگار باشد و از هر نوع داده‌ای استفاده کند.

۲. AdaBoost (Adaptive Boosting):

- توجه به نمونه‌های دشوار: AdaBoost با تمرکز بر نمونه‌های دشوار و نادرست در فرآیند آموزش، تلاش می‌کند تا در هر مرحله مدل را بهبود دهد و از بیش‌برازش جلوگیری کند.

- اجتماع یادگیری: در هر مرحله، AdaBoost یک مدل ضعیف جدید را به مدل قبلی اضافه می‌کند و تلاش می‌کند با ترکیب این مدل‌های ضعیف، یک مدل قوی و کارآمد بسازد.

- تعمیم‌پذیری بالا: AdaBoost دارای قابلیت تعمیم‌پذیری بالاست و معمولاً به خوبی در مسائل دسته‌بندی با تعداد کمی از داده‌ها عمل می‌کند.

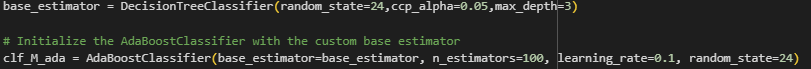
- استفاده از توابع هدف متفاوت: این روش می‌تواند با استفاده از توابع هدف متفاوتی مانند Entropy، Gini Index یا میزان اشتباه دسته‌بندی کند و به دسته‌بندی بهتری دست یابد.

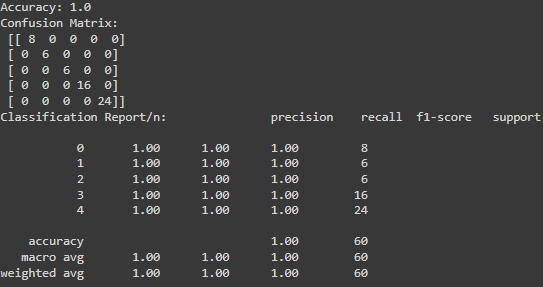
به طور کلی، جنگل تصادفی و AdaBoost از دو روش قدرتمند و مؤثر در یادگیری ماشینی هستند که بهبود نتایج و عملکرد مدل‌های پیش‌بینی را تسهیل می‌کنند.

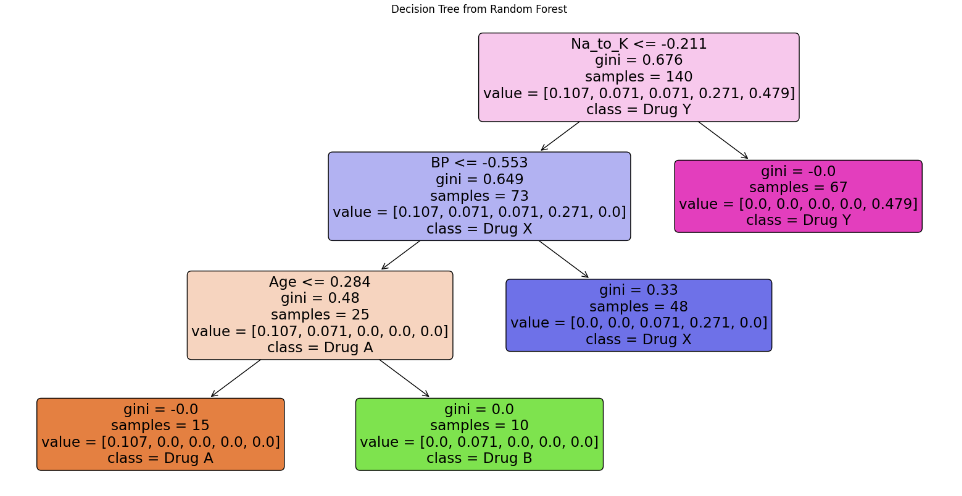
### AdaBoost :

اگر داده های قبل با hyper parameter های تغییر داده که دقت 90% داده بود و نتوانسته بود کلاس 3 را تشخیص بدهد را به adaboost بدهیم داریم:

لازم به ذکر هست که مادقت 100% بدون تغییر hyper\_parameter ها در بخش 1 گرفتیم.

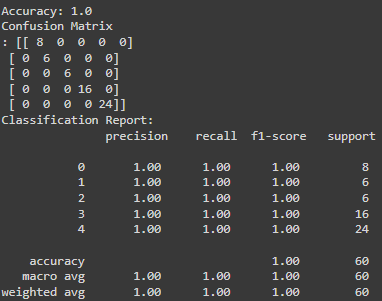


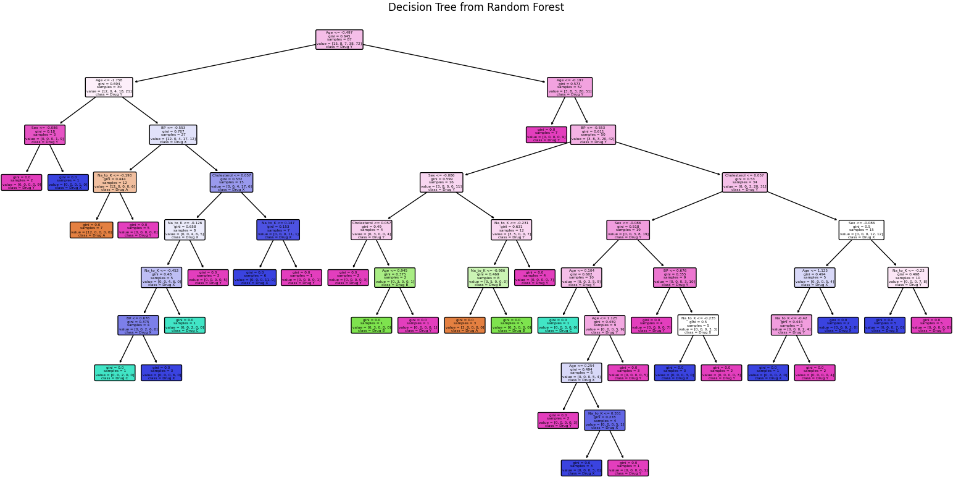




### Random Forest

همچنین برای Random Forest داریم :

****

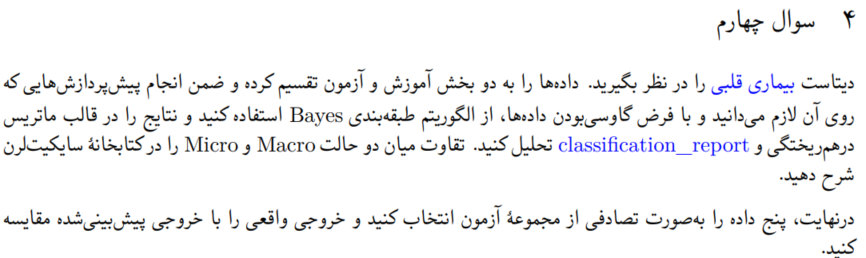


که تماما دقت 100% داشته اند

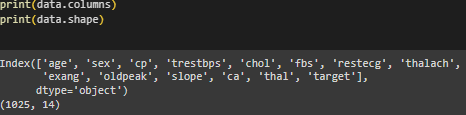
### نکته

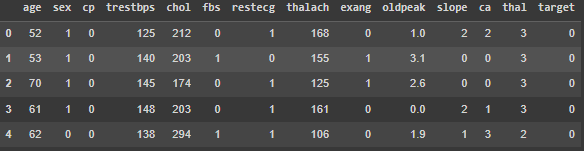
لازم به ذکر است که این دیتاست دارو با تعداد ویژگی و نمونه کمتر راحت تر حل شد در فایل کد گوگل کلب دیتاست دوم نیز حل شده ولی برای سادگی در گزارش فقط دیتاست اول را آوردم.

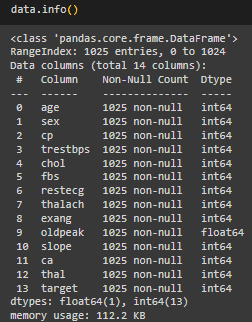
# سوال 4



داده ها را دریافت کرده و بررسی می کنیم:

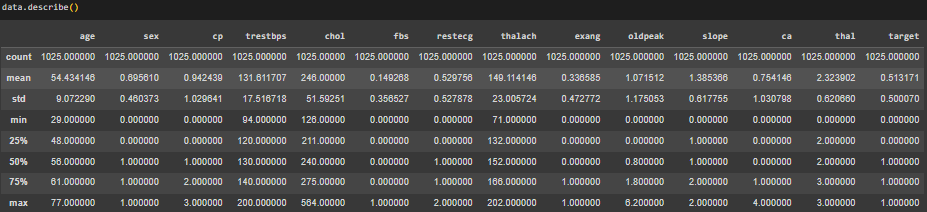






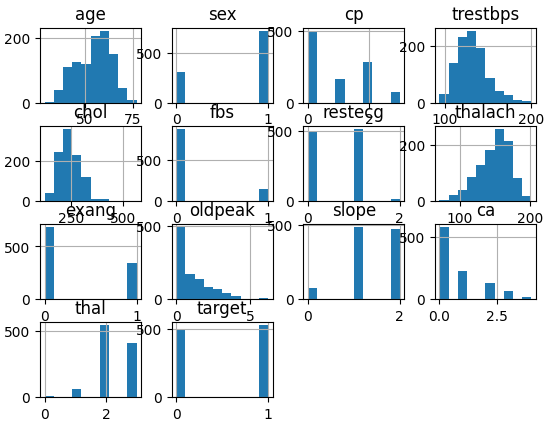
1025 نمونه وتعداد 13 ویژگی داریم که یکی از آن ها هدف یا ‘target’ هست.

از این 14 داده 13 تای آنها integer و یک float داریم.همچنین داده null ای در دیتاسست موجود نیست.



### Histogram

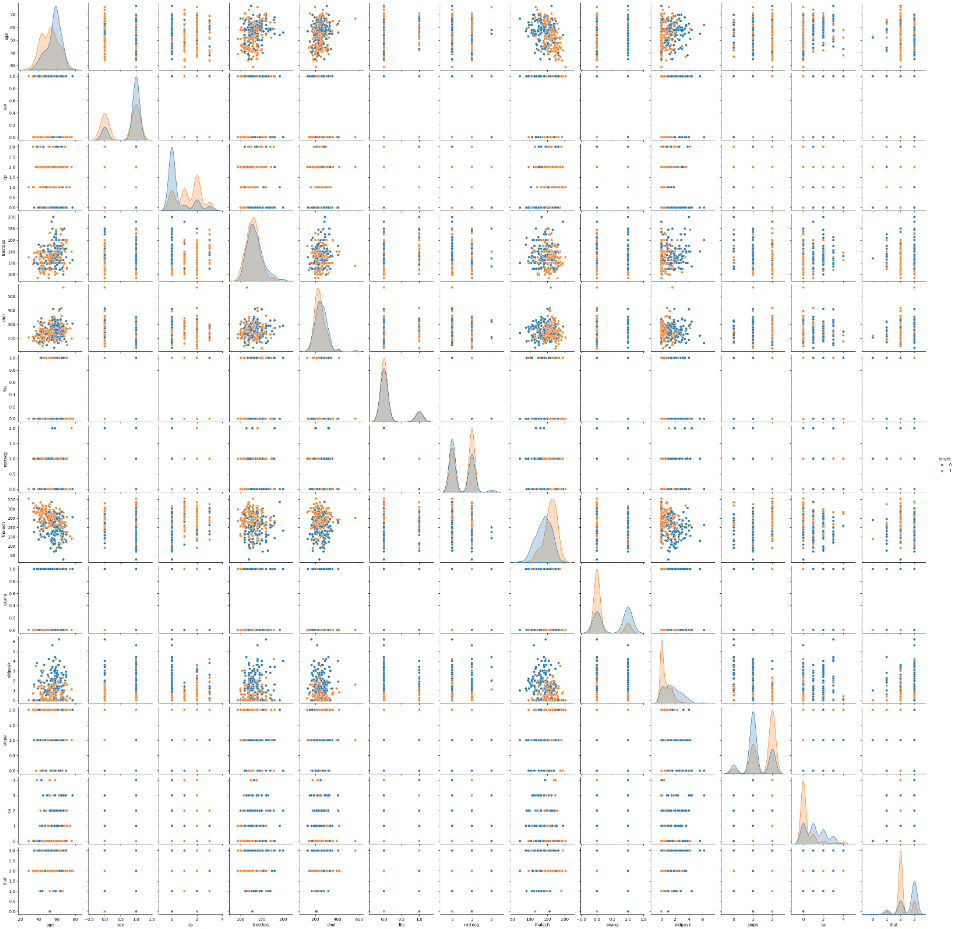
نمودار هیستوگرام را رسم میکنیم



هر هیستوگرام نماینده توزیع مقادیر برای یک ویژگی خاص در مجموعه داده است. می‌توانیم از این هیستوگرام‌ها برای به دست آوردن بینش‌ها درباره توزیع داده‌های خود و شناسایی الگوها یا نقصان‌های موجود استفاده کنید.

### Pairplot

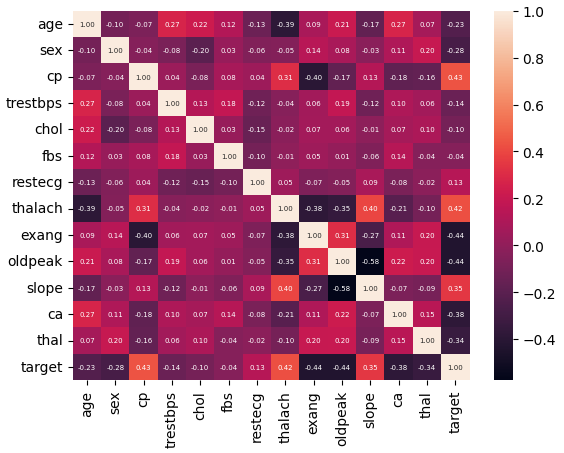
همچنین با تابع sns.pairplot() از کتابخانه Seaborn یک شبکه از نمودارهای جفتی برای هر متغیر در مجموعه داده شما ایجاد می‌کنیم:

  
 این تابع همچنین شامل نمودارهای پراکندگی برای روابط مشترک و هیستوگرام‌ها برای توزیع‌های یک متغیر در قطر است.

پارامتر hue برای مشخص کردن یک متغیر دسته‌ای استفاده می‌شود که برای رنگ‌آمیزی نقاط داده در نمودار استفاده می‌شود و اطلاعات اضافی درباره روابط بین متغیرها را فراهم می‌کند.

بنابراین، sns.pairplot(data, hue='target') یک شبکه از نمودارهای جفتی ایجاد می‌کند که نقاط داده بر اساس متغیر "هدف" رنگ آمیزی شده‌اند. این به شما امکان می‌دهد که روابط بین متغیرها را به صورت تصویری بررسی کنید در حالی که هدف را در نظر می‌گیرید. نمودار های مربوطه در colab آمده است.

### Corr



با تحلیل ضرایب همبستگی (correlation coefficients) می‌توانیم برخی از روابط میان ویژگی‌ها و متغیر هدف را مشاهده کنیم. برای مثال:

CP (نوع درد قفسه سینه): همبستگی مثبت و معنی‌دار با متغیر هدف (حدود 0.43). این نشان می‌دهد که نوع خاصی از درد قفسه سینه ممکن است با ابتلای به بیماری قلبی مرتبط باشد.

Thalach (ضربان قلب حداکثر): نیز همبستگی مثبت و معنی‌داری با متغیر هدف دارد (حدود 0.42). این نشان می‌دهد که فرکانس ضربان قلب در حداکثر ورزش ممکن است به عنوان یک عامل مهم برای تشخیص بیماری قلبی مورد استفاده قرار گیرد.

ویژگی‌های منفیاً مرتبط با متغیر هدف:

Exang (آنژین ناشی از فعالیت): همبستگی منفی و معنی‌داری با متغیر هدف دارد (حدود -0.44). این نشان می‌دهد که آنژین ناشی از فعالیت ممکن است به عنوان یک علامت برای بیماری قلبی در نظر گرفته شود.

Oldpeak (کاهش ST): همچنین همبستگی منفی و معنی‌داری با متغیر هدف دارد (حدود -0.44). این نشان می‌دهد که کاهش آمپلیتود ST پس از ورزش ممکن است به عنوان یک علامت برای بیماری قلبی در نظر گرفته شود.

ویژگی‌های کم‌تاثیر:

میان برخی از ویژگی‌ها و متغیر هدف، همبستگی ضعیف و یا نامعنی وجود دارد. به عنوان مثال، بین سن (Age) و متغیر هدف، همبستگی معنی‌داری وجود ندارد (حدود -0.23).

ویژگی‌های تکراری:

برخی از ویژگی‌ها با یکدیگر همبستگی معنی‌داری دارند که می‌تواند نشان دهنده وجود ویژگی‌های تکراری باشد. برای مثال، همبستگی معنی‌داری بین سن و فشار خون در استراحت (Trestbps) وجود دارد (حدود 0.27).

این تحلیل‌ها می‌تواند به ما کمک کند تا ویژگی‌های مهم‌تر را برای پیش‌بینی بیماری‌های قلبی شناسایی کرده و به عنوان ورودی‌های اصلی برای مدل‌های طبقه‌بندی مورد استفاده قرار دهیم.

باشد. در این تحلیل، به بررسی اطلاعات بیشتری از هر یک از ویژگی‌ها می‌پردازیم:

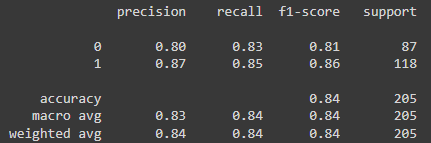
این تحلیل ما را قادر می‌سازد تا ویژگی‌های مهم و تاثیرگذار در تشخیص بیماری‌های قلبی را شناسایی کرده و آن‌ها را به عنوان ورودی‌های اصلی برای مدل‌های طبقه‌بندی مورد استفاده قرار دهیم.

## 1.

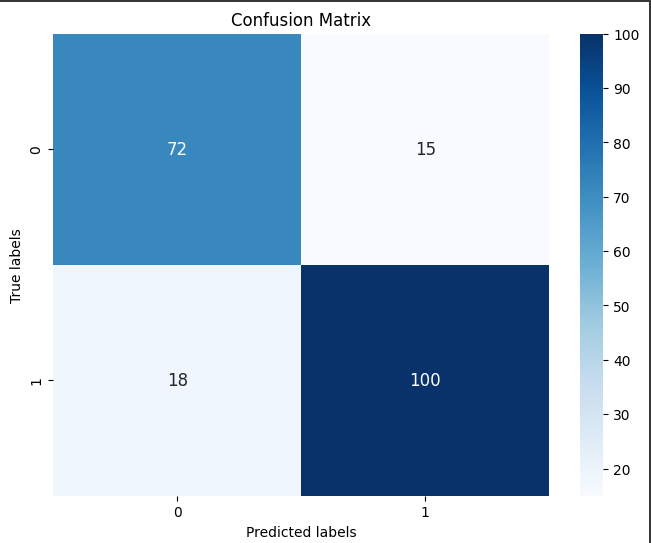
داده ها را با درصد تقسیم 20% به آموزش و تست تقسیم می کنیم .



با GaussianNB() طبقه بند Bayes با توزیع گوسی را تعریف و به داده های آموزش ، فیت می کنیم:  
ماتریس درهمریختگی و classification\_report :







داده ها با دقت بالای 80 درصد با معیار های مختلف نتیجه داده اند.

در زمینه گزارش‌های طبقه‌بندی، اصطلاحات "میکرو" و "ماکرو" به روش‌های متفاوتی که برای محاسبه معیارهایی مانند percision، recall و F1-score استفاده می‌شود، اشاره دارند. در ادامه تفاوت بین این دو روش را بیان می‌کنم:

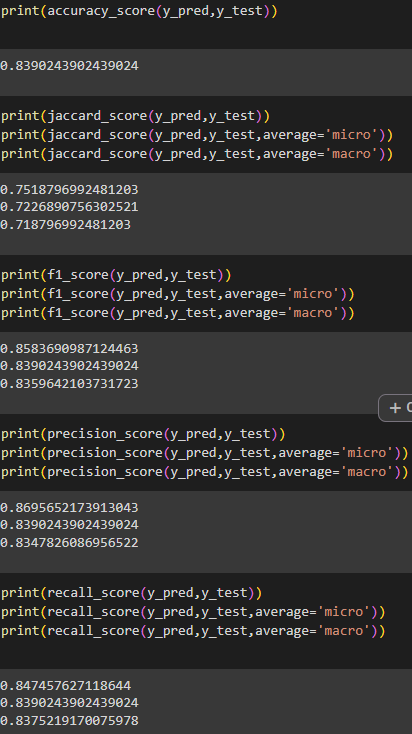
1. micro average:

به هر زوج نمونه-کلاس یک مشارکت مساوی در معیار کلی می‌دهد (به جز نتیجه‌ی وزن نمونه). به جای جمع کردن معیار برای هر کلاس، این معیار‌ها را به یکدیگر تقسیم می‌کند تا یک نسبت کلی محاسبه شود. میانگین میکرو ممکن است در تنظیمات چند برچسبی، از جمله طبقه‌بندی چند کلاسه که یک کلاس اکثریت باید نادیده گرفته شود، ترجیح داده شود.

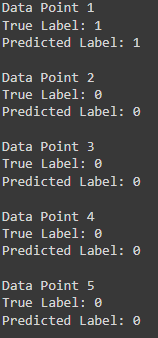
2. macro average :

میانگین معیارهای دودویی را محاسبه می‌کند و وزن مساوی به هر کلاس می‌دهد. در مسائلی که کلاس‌های با ‌فراوانی کم همچنان مهم هستند، میانگین ماکرو می‌تواند روشی برای برجسته کردن عملکرد آن‌ها باشد. از طرف دیگر، فرضیه‌ی اینکه تمامی کلاس‌ها به یک اندازه مهم هستند اغلب صحیح نیست، بنابراین میانگین ماکرو به طور زیادی بر عملکرد معمولاً پایین کلاس‌های با‌فراوانی کم تأثیر خواهد گذاشت.

به طور خلاصه، تفاوت اصلی در چگونگی محاسبه معیارها است: میکرو همه نمونه‌ها را به طور مساوی در نظر می‌گیرد، در حالی که ماکرو-میانگین همه کلاس‌ها را به طور مساوی در نظر می‌گیرد. انتخاب بین میکرو-میانگین و ماکرو-میانگین به توجه به اهداف ارزیابی خاص و ویژگی‌های مجموعه داده وابسته است.



حال 5 داده رندوم از مجموعه تست انتخاب شده تخمین زده و با مقدار واقعی مقایسه میشود:



مشاهده می شود که تمام دادخ ها به درستی تخمین زده شدند.