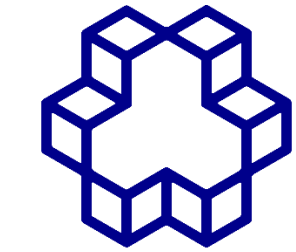
Google drive: https://drive.google.com/drive/folders/1n8uWALN39KWm7Zrq6tpUlCi1efUUQn4D?usp=sharing

Github: https://github.com/AliBagheriNejad/ML-AliYari



گزارش نیمچه پروژه ۲ درس یادگیری ماشین

علی باقری نژاد

۴۰۲۰۲۸۵۴

فهرست مطالب

فهرست شکل‌ها

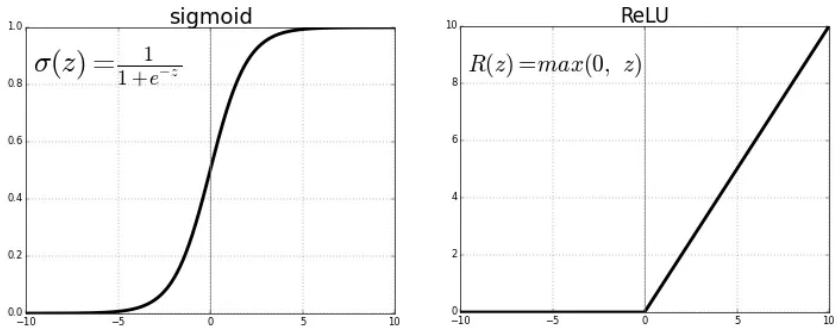
سوال ۱

۱-۱

فرمول محاسبه توابع ReLU و sigmoid به‌صورت زیر می‌باشد:

$$a_{ReLU}(z) = \max(0, z)$$

$$a_{sigmoid}(z) = \frac{1}{1 + e^{-z}}$$



شکل ۱ تصویر عملکرد توابع ReLU و Sigmoid [1]

هنگامی که داده‌ها از تابع ReLU می‌گذرند، مقادیر مثبت آن‌ها باقی می‌ماند و مقادیر منفی آن‌ها صفر می‌شود. یعنی خروجی این لایه، اعدادی غیرمنفی می‌باشند. پس از آنکه خروجی لایه قبلی (a) در وزن‌های لایه نهایی ضرب شد ($w^{[last]}, a >$)، مقادیر $Z^{[last]}$ وارد تابع فعالسازی sigmoid می شوند. در صورت طبقه بندی دوکلاسه، لایه نهایی تنها یک نورون خواهد داشت و خروجی لایه نهایی در بازه $[0,1]$ خواهد بود.

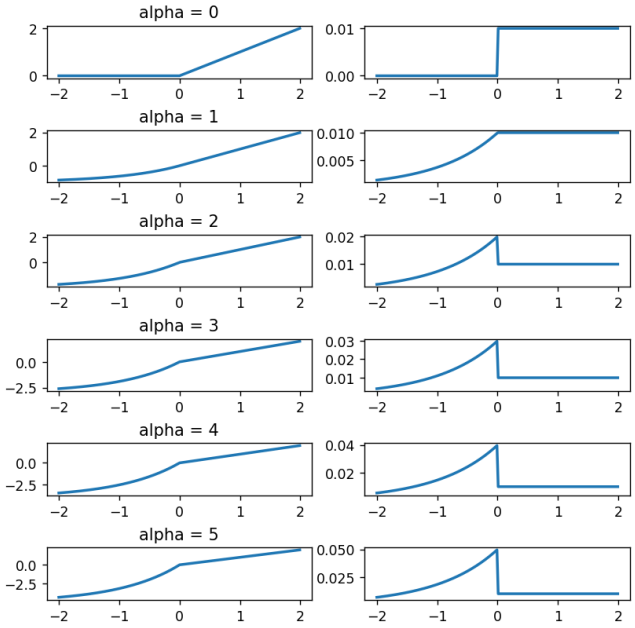
برای طبقه‌دو کلاسه نیازی به استفاده از دو نورون نیست، زیرا به اندازه آن مقداری که یک ورودی به یک کلاس تعلق ندارد، به کلاس دیگر تعلق خواهد داشت. به همین دلیلی تنها از یک نورون استفاده می کنیم تا حجم محاسبات را کاهش دهیم. یعنی درصد تعلق ورودی به یک کلاس را درنظر گرفته و هرچه که به این کلاس تعلق نداشته باشد، به دیگری تعلق دارد.

در نهایت، با مقایسه مقدار خروجی با یک threshold، تعیین می‌کنیم که ورودی به کلاس ۱ تعلق دارد یا به کلاس ۰.

۱-۲ EReLU

محاسبه گرادیان:

$$f=\begin{cases} x; x\geq 0 \\ \alpha(e^x-1); x<0 \end{cases} \Rightarrow \frac{\partial f}{\partial x}=\begin{cases} 1; x\geq 0 \\ \alpha e^x; x<0 \end{cases} \overset{\alpha=1}{\Longrightarrow} \frac{\partial f}{\partial x}=\begin{cases} 1; x\geq 0 \\ e^x; x<0 \end{cases}$$



شکل ۲عملکرد تابع EReLU به ازا مقادیر مختلف alpha

شکل ۲ تابع EReLU و مشتق آن را برای α های مختلف نمایش داده است. تصاویر سمت چپ مربوط به خـود تـابع و تصاویر سمت راست مربوط به مشتق آن می باشند. اگر مقدار $\alpha = 0$ آنگاه تابع ReLU بدست می آید. این تابع بـه ازای $\alpha =$ 1 در نقطه $x = 0$ دارای مشتق می باشد. در هر α دیگری، در نقطه $x = 0$ مشتق نداریم.

برتری نسبت به ReLU:

- قابلیت خروجی دادن برای مقادیر منفی
- نرمی بیشتر تابع بین در نقطه ۰ (در حالت $\alpha = 1$ مشتقپذیر نیز می باشد)

۱-۳ طراحی شبکه برای تشخیص ناحیه مشخص شده

برای انجام این بخش، از مدل MCP استفاده می‌کنیم.

سه خط در نظر می‌گیریم که اگر نقطه ای نسبت به این خطوط شرط های مناسب را داشته باشد؛ مثلا پایین تـر از یـک خط و بالاتر از دو خط دیگر باشد، در ناحیه هاشورخورده قرار می گیرد. از آنجایی که صرفا نیاز است یـک سـری شـروط در ایـن مسئله ارضا شود، True یا False باشد، از نورون MCP استفاده می کنیم.

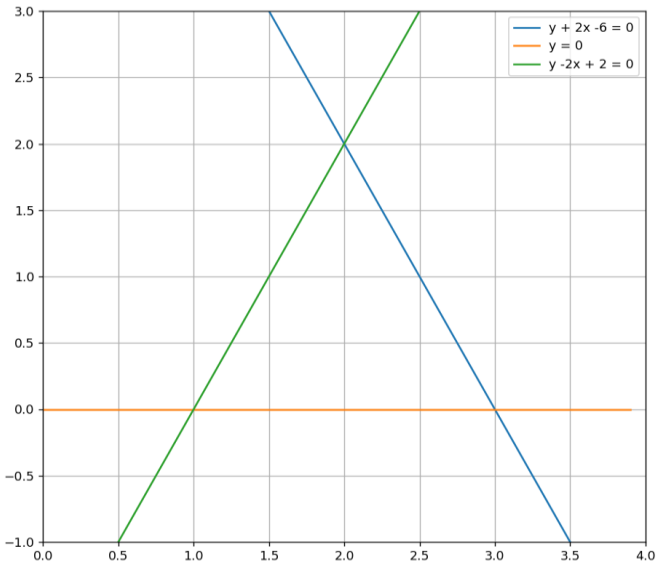
اگر قرار باشد که نقاطی که در ناحیه هاشورخورده هستند تعیین شوند، باید نسبت معادله خطوطی که ناحیه هاشورخوره را ایجاد می کنند، حالت مناسبی داشته باشند. برای درک بهتر این مطلب، تئوری حل مسئله توضیح داده‌می‌شود.

از نقاط رأس مثلث داده شده استفاده می کنیم و سه معادله خط را بدست می آوریم که از تقاطع این خطوط، مثلث داده شده ایجاد می شود:

$$\overline{AB} \overset{A}{\Rightarrow} y - 2 = -2(x - 2) \rightarrow y + 2x - 6 = 0 \qquad I$$

$$\overline{BC} \overset{B}{\Rightarrow} y - 0 = 0(x - 3) \rightarrow y = 0 \qquad II$$

$$\overline{CA} \overset{C}{\Rightarrow} y - 0 = 2(x - 1) \rightarrow y - 2x + 2 = 0 \qquad III$$



شکل ۳ خطوط ایجاد شده از معادلات خط بدست آمده

شکل ۳ خطوط ایجاد شده از معادلات خط بدست آمده را به نمایش گذاشته است. برای آن که نقاط بین این خطوط قرار گیرند، باید شروط زیر را ارضاء کنند:

- $y + 2x - 6 \leq 0 \Rightarrow -2x - y \geq -6$

- $y \geq 0$
- $y - 2x + 2 \leq 0 \Rightarrow 2x - y \geq 2$

حال بر اساس این نامعادلات بدست آمده، نورون های MCP مورد نیاز را بدست می آوریم. از آنجایی که با سه نامعادله توانستیم ناحیه خواسته شده را تعیین کنیم، بنابراین باید سه نورون MCP طراحی کرده تا استفاده از خروجی هر سه نورون، حضور نقطه در ناحیه خواسته شده را تعیین کنیم. البته باید یک نورون دیگر هم طراحی کنیم که با گرفتن خروجی ها سه نورون دیگر، نتیجه‌گیری نماید که آیا نقطه داده شده درون ناحیه تعیین شده قرار دارد یا خیر.

بنابراین، سه نورون خواهیم داشت که مختصات x و y به‌عنوان ورودی به این نورون ها داده می شوند و اگر خروجی تمامی این نورون ها True باشد، با گذشت از نورون نهایی که پس از نورون های اولیه قرار می گیرد، مقدار True یا False بودن حضور نقطه در ناحیه مشخص شده، تعیین می شود. وزن ها و آستانه های نورون ها به صورت زیر می باشند: (ورودی به صورت $u = [x \quad y]^T$ می باشند)

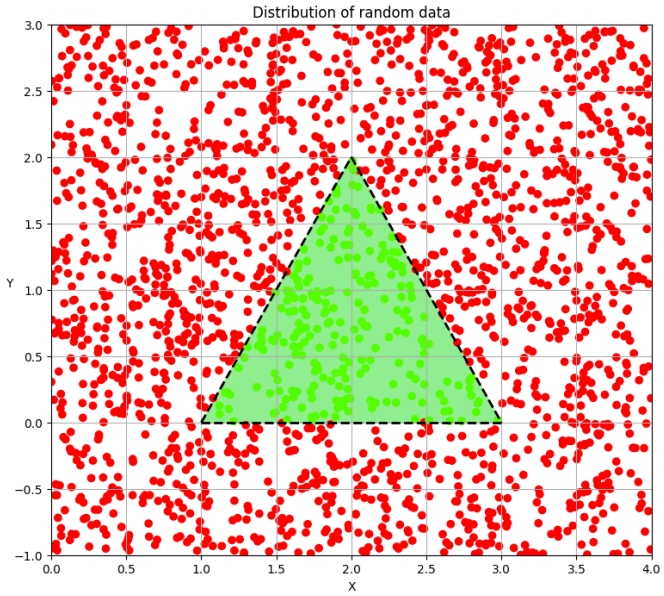
$$neuron1: W = [-2 \quad -1], \qquad threshold = -6$$
$$neuton2: W = [0 \quad 1], \qquad threshold = 0$$
$$neuron3: W = [2 \quad -1], \qquad threshold = 2$$
$$neuron4: W = [1 \quad 1 \quad 1], \qquad threshold = 3$$

خروجی نورون‌های ۱تا۳ وارد نورون ۴ام می شوند. اگر مقدار تمامی نورون های ۱تا۳ برابر ۱ (True) باشد، آنگاه مشخص می شود که نقطه داده شده در ناحیه مشخص شده قرار دارد و در غیر این صورت، اگر خروجی حداقل یکی از نورون ها ۰ باشد، نورون نهایی مقدار ۰ (False) را باز می گرداند که نشان می دهد نقطه داده شده در ناحیه مشخص شده قرار ندارد.

```
Point with coordinates of [2 1] is in the area
Point with coordinates of [2.  1.5] is in the area
Point with coordinates of [0 0] is NOT in the area
Point with coordinates of [1 1] is NOT in the area
```

شکل ۴ عملکرد مدل در برابر چند نقطه داده شده

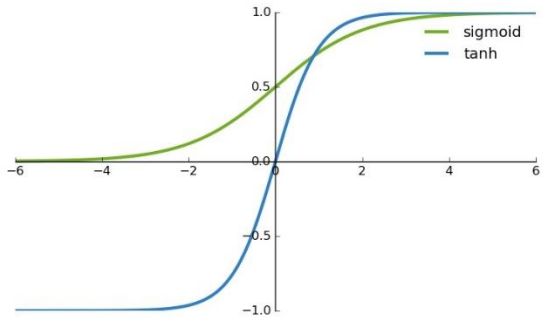
فایل Q1_1.ipynb مدل ایجاد شده از نورون‌های MCP را اجزا می کند. شکل ۴ عملکرد مدل را برای ۴ نقطه به نمایش گذاشته است که که دوتای آن ها درون ناحیه مشخص شده قرار دارند و دو تای دیگر آن ها خارج از ناحیه می باشند. همانطور که مشاهده می شود، مدل ایجاد شده توانسته است به خوبی وظیفه خواسته شده را انجام دهد.



شکل ۵ نقاط رندوم و پیشبینی مدل

شکل ۵ عملکرد مدل ایجاد شده روی داده های رندوم را به نمایش گذاشته است.

اولین تابع فعال‌سازی که از آن استفاده کردیم، تابع فعال‌ساز hard limit بود. این تابع فعال‌ساز، برای تمامی نورون ها استفاده شد. با استفاده از تابع hard limit، به عنوان خروجی صرفا قرار داشتن یا نداشتن نقطه در ناحیه خواسته شده معلوم می شود. برای قسمت بعدی از دو تابع فعال سازی sigmoid و tanh استفاده می کنیم.



شکل ۶ مقایسه عملکرد توابع فعاسازی sigmoid و tanh [1]

ابتدا از sigmoid و سپس از tanh استفاده می‌کنیم. اگر از این توابع فعاسازی استفاده کنیم، نورون ایجاد شده دیگر MCP نمی‌باشد و صرفا در حال استفاده از یک نورون عادی (پرسپترون) هستیم.

برای استفاده از توابع فعال‌سازی جدید، مقدار thresholdای که در MCP استفاده کردیم را حذف کرده و از مقدار bias برای محاسبات استفاده می‌کنیم.

استفاده از sigmoid:

همان‌طور که در شکل ۵ نمایش داده شده است، خروجی این تابع عددی بین ۰ و ۱ خواهد بود. یعنی باید وزن‌ها را طوری تعیین کنیم هر چه که عدد خروجی نورون به ۱ نزدیک باشد، نشانه حضور داشتن در ناحیه خواسته شده است و هر چه خروجی به عدد ۰ نزدیک باشد، نشانه خارج بودن از ناحیه مورد نظر است.

ازآنجایی‌که خروجی تابع یک عدد پیوسته است، باید عددی به عنوان threshold تعیین کنیم تا بتوانیم خروجی را به یک کلاس نسبت به دهیم. در حالت استفاده از تابع sigmoid، threshold را ۰.۵ درنظر می‌گیریم.

پارامترهای در نظر گرفته‌شده:

$$n_1^1: W = [-2 \quad -1], b = +6$$
$$n_2^1: W = [0 \quad 1], b = 0$$
$$n_3^1: W = [2 \quad -1], b = -2$$
$$n_1^2: W = [1 \quad 1 \quad 1], b = -1.5$$

استفاده از تابع tanh:

این تابع همانند تابع قبل عمل می‌کند با این تفاوت که خروجی نهایی آن بین ۱ و-۱ قرار دارد. یعنی اگر عدد داده شده به ۱ نزدیک بود، یعنی نقطه در ناحیه مورد نظر قرار دارد و اگر خروجی به -۱ نزدیک بود یعنی که نقطه داده‌شده در ناحیه مشخص شده قرار ندارد.

همانند تابع sigmoid، برای این تابع هم threshold صفر را انتخاب می‌کنیم.

پارامترهای در نظر گرفته‌شده:

$$n_1^1: W = [-2 \quad -1], b = +6$$
$$n_2^1: W = [0 \quad 1], b = 0$$
$$n_3^1: W = [2 \quad -1], b = -2$$
$$n_1^2: W = [1 \quad 1 \quad 1], b = 0$$

نکته: سه نورون اول مشخص می‌کنند که آیا مختصات داده شده نسبت به خطوط موردنظر در ناحیه دلخواهی قرار دارد یا خیر. ممکن است که یک نقطه نسبت به دو خط در ناحیه مورد نظر قرار داشته باشد ولی نسبت به خط دیگر در ناحیه مناسبی نباشد؛ در این حالت احتمال داردکه خروجی آخرین نورون بیشتر از threshold قرار داشته‌باشد ولی نزدیک threshold قرار می گیرد. یعنی اگر خروجی نهایی نزدیک عدد threshold قرارداشت، شاید نتوان با اعتماد کامل درباره حضور یا عدم حضور مختصات نقطه در ناحیه نظر داد. برای درک بهتر این متلب، ۵ نقطه به صورت تصادفی انتخاب شدند و خروجی نهایی در شکل زیر به نمایش درآمده است.

```
For the point with coordinates of :[1.681 2.876]
Outcome with sigmoid AF is: 0.52
Outcome with tanh AF is: -0.15
Point with coordinates of [1.681 2.876] is NOT in the area

For the point with coordinates of :[1.453 2.206]
Outcome with sigmoid AF is: 0.58
Outcome with tanh AF is: 0.68
Point with coordinates of [1.453 2.206] is NOT in the area

For the point with coordinates of :[0.74  2.029]
Outcome with sigmoid AF is: 0.59
Outcome with tanh AF is: 0.75
Point with coordinates of [0.74  2.029] is NOT in the area

For the point with coordinates of :[2.073 1.686]
Outcome with sigmoid AF is: 0.62
Outcome with tanh AF is: 0.91
Point with coordinates of [2.073 1.686] is in the area

For the point with coordinates of :[ 0.034 -0.881]
Outcome with sigmoid AF is: 0.51
Outcome with tanh AF is: -0.45
Point with coordinates of [ 0.034 -0.881] is NOT in the area
```

شکل ۷خروجی مدل به صورت یک عدد پیوسته

در شکل بالا مشاهده می‌شود هرجایی که احتمال بالا باشد نقطه درون ناحیه قرار دارد؛ ولی اگر احتمال نزدیک به threshold باشد، نمی‌توان به صورت قطعی نظر داد. ۴ نقطه از ۵ نقطه خارج از ناحیه قرار دارند ولی مثلا حالت sigmoid آن‌ها را داخل ناحیه حساب کرده است.

برای بهبود عملکرد می‌توان وزن‌های دیده‌شده را در یک عدد ضرب کرد که به ازای ضرایب بالا، توابع را بـه تـابع hard limit تبدیل می‌کنیم و همان نورون MCP را می‌سازیم.

سؤال ۲

۲-۱ دیتاست CWRU

برای دانلود دیتاست‌های مربوط به هر کلاس، از دستور `wget` استفاده شد.

داده‌های معیوب خواسته شده به سه نوع دسته بندی می شوند:

- عیب شیار داخلی (Inner race)
- عیب ساچمه (Ball)
- عیب شیار خارجی (Outer race)

نام داده‌های مربوط به هر نوع عیب به ترتیب IR007_2، B007_2 و OR007@6_2 می باشد.

البته عیب‌های مربوط به شیار خارجی دارای سه حالت مختلف مرکز، عمود و مقایل بودند که بـرای ایـن تمـرین حالـت مرکز انتخاب شده است.

پس از دانلود داده‌ها، با استفاده از دستور `loadmat` از کتابخانه scipy، فایل ها خوانده می شـوند. هماننـد سـری قبلـی تمرین، داده ها از بخش های مختلفی تشکیل شده اند که برای ادامه روند حـل مسـئله، مـدل `DE-time` کـه در تمـامی آن هـا مشترک می باشد را انتخاب می کنیم.

```
normal data consists of ['X098_DE_time', 'X098_FE_time', 'X099_DE_time', 'X099_FE_time']
inner race fault data consista of ['X107_DE_time', 'X107_FE_time', 'X107_BA_time']
ball fault data consists of ['X120_DE_time', 'X120_FE_time', 'X120_BA_time', 'X120RPM']
outer race fault data consists of ['X132_DE_time', 'X132_FE_time', 'X132_BA_time']
```

شکل ۸ تقسیم بندی موجود برای داده ها

سپس همانند تمرین سری اول، ۲۵۰ سیگنال به اندازه ۲۰۰ از داده های اصلی جدا میکنیم.

```
From normal data, 250 signals with length of 200 are extracted
From inner fault data, 250 signals with length of 200 are extracted
From ball fault data, 250 signals with length of 200 are extracted
From outter fault data, 250 signals with length of 200 are extracted
```

شکل ۹ تعداد و اندازه سیگنال های جدا شده از داده ها

در سری قبل، تمامی ویژگی‌های داده شده را برای نمونه های سیگنال محاسبه کردیم. در این جا نیز تمامی ویژگی ها را محاسبه می نماییم.

```
Label of 'normal' data is 0
Label of 'inner fault' data is 1
Label of 'ball fault' data is 2
Label of 'outter fault' data is 3
```

	standard deviation	peak	skewness	mean	absolute mean	root mean square	square root mean	kurtosis	crest factor	clearance factor	peak to peak	shape factor	impact factor	impulse factor	label
0	0.065162	0.179826	-0.102434	0.016275	0.051197	0.067005	0.041341	0.205579	2.683775	4.349826	0.355481	1.308760	1.308760	0.317890	0
1	0.065124	0.179826	-0.101089	0.016218	0.051140	0.066955	0.041293	0.211600	2.685784	4.354910	0.355481	1.309248	1.309248	0.317125	0
2	0.065046	0.179826	-0.095520	0.016050	0.050972	0.066839	0.041131	0.224659	2.690446	4.371997	0.355481	1.311286	1.311286	0.314875	0
3	0.065087	0.179826	-0.092039	0.015959	0.051063	0.066857	0.041302	0.216531	2.689711	4.353963	0.355481	1.309313	1.309313	0.312538	0
4	0.065158	0.179826	-0.091184	0.015882	0.051140	0.066907	0.041377	0.205829	2.687688	4.346023	0.355481	1.308321	1.308321	0.310557	0

شکل ۱۰ نمایی از دیتافریم ایجاد شده

همان‌طور که در شکل ۱۰ نمایش داده شده است، برای هر نوع داده نرمال یا عیب، یک لیبل عددی تخصیص داده شده است تا بتوان محاسبات را انجام داد.

داده‌ها را به روش زیر به سه بخش آموزش، ارزیابی و اعتبارسنجی تقسیم می‌کنیم:

۱. ۱۵ درصد از کل داده ها را جدا کرده و به مجموعه ارزیابی تخصیص می‌دهیم.

۲. از میان داده‌های باقی‌مانده، ۱۵ درصد را به مجموعه داده اعتبارسنجی اختصاص داده و باقی داده‌ها را به عنوان داده آموزش استفاده می‌کنیم.

```
Size of Train data is:
X --> (722, 14)
y --> (722,)
Size of Validation data is:
X --> (128, 14)
y --> (128,)
Size of Test data is:
X --> (150, 14)
y --> (150,)
```

شکل ۱۱ تقسیم بندی داده‌ها

مجموعه‌داده اعتبارسنجی برای کارهای زیر استفاده می‌شود:

- انتخاب مدل

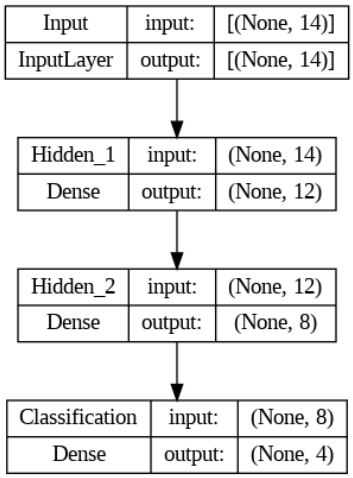
- تنظیم‌کردن hyperparameters
- جلوگیری از overfitting
- Early stopping در بحث شبکه عصبی و یادگیری عمیق

مرحله بعد از تقسیم‌بندی داده‌ها، نرمالایز کردن آن‌ها است. این کار با دستور `StandardScaler` انجام می‌شود. ابتدا پارامترهای scaler را با توجه به داده‌های آموزش بدست می‌آوریم و سپس scaler را به تمامی داده‌ها اعمال می‌کنیم.

۲-۲: ساختن شبکه MLP

برای ایجاد این شبکه عصبی، از کتابخانه tensorflow استفاده می‌کنیم. قبل از انجام هر کاری، مقدار seed کتابخانه tensorflow را برای ۵۴ تنظیم می‌کنیم تا تمامی فعالیت‌های انجام شده قابل تکرار باشند و بتوانیم نتایج را با هم مقایسه کنیم. (برای این کار از تابع `set_seed` استفاده می‌کنیم)

سپس با استفاده از توابع `Dense` و `Input`، یک مدل Sequential ایجاد می‌کنیم.

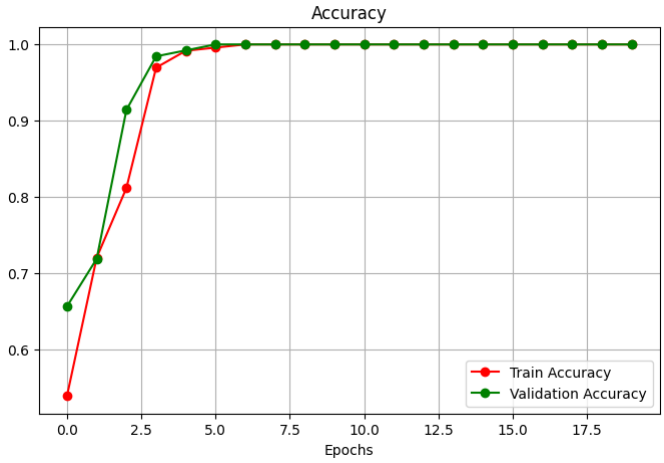


شکل ۱۲ ساختار شبکه ایجاد شده

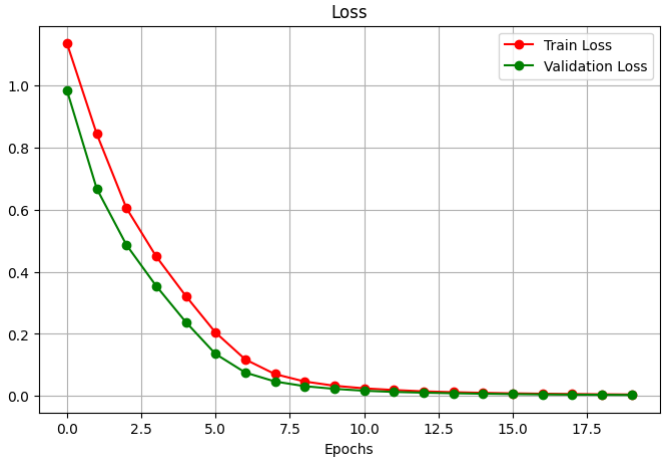
ازآنجایی‌که لیبل‌های ما (`y_train` و `y_val`) به صورت عدد بوده و one-hot-encoded نبودند، برای تابع خطا، از تابع `SparseCategoricalCrossentropy` استفاده می‌کنیم. برای Optimizer از `Adam` استفاده می‌نماییم.

مدل نهایی را با مشخصات زیر آموزش می‌دهیم:

```
x = x_train_scaled
y = y_train
batch_size = 16
epochs = 20
validation_data = (x_val_scaled,y_val)
```



شکل ۱۳ دقت بدست‌آمده حین آموزش



شکل ۱۴ خطا بدست‌آمده حین آموزش

باتوجه‌به نمودار به نمودارهای دقت و خطا، می‌بینیم که هر دو در حال بهبود هستند و خیلی سریع به بهترین حالت خود می‌رسند؛ یعنی خطا به سرعت کاهش یافته و ثابت می‌شود و دقت به به ۱۰۰ درصد رسیده و ثابت می‌ماند. در این آموزش، با ۲۰ دور آموزش دادن شبکه، مدل دچار overfit نمی‌شود. از طرفی کاملا معلوم است که مدل overmodel هم نمی‌باشد زیرا داده‌های اعتبارسنجی به خوبی طبقه‌بندی می‌شوند.

قبل از انجام classification report، باید داده‌های آموزش و اعتبارسنجی را به مدل داده تا برای آن‌ها پیش‌بینی را انجام دهد و بعد از به دست آوردن y-hat برای آن‌ها، گزارش طبقه‌بندی را انجام دهیم.

Classification report for Train data is:

	precision	recall	f1-score	support
normal	1.00	1.00	1.00	183
inner brace fault	1.00	1.00	1.00	192
ball fault	1.00	1.00	1.00	174
outter brace fault	1.00	1.00	1.00	173
accuracy			1.00	722
macro avg	1.00	1.00	1.00	722
weighted avg	1.00	1.00	1.00	722

Classification report for Validation data is:

	precision	recall	f1-score	support
normal	1.00	1.00	1.00	36
inner brace fault	1.00	1.00	1.00	22
ball fault	1.00	1.00	1.00	31
outter brace fault	1.00	1.00	1.00	39
accuracy			1.00	128
macro avg	1.00	1.00	1.00	128
weighted avg	1.00	1.00	1.00	128

شکل ۱۵ classification report برای داده‌های آموزش و اعتبارسنجی

Classification report تعداد تمامی داده‌ها و کلاس‌ها را آوردن و با معیارهای مناسبی نتایج را برای آن‌ها اعلام می‌کند. همان‌طور که از شکل ۱۵ برداشت می‌شود، برای تمامی کلاس‌ها، تمامی معیارها، هستند؛ یعنی مدل تمامی ورودی‌ها را به درستی پیش‌بینی کرده‌است و عملکرد بسیار خوبی دارد.

Classification report for Test data is:

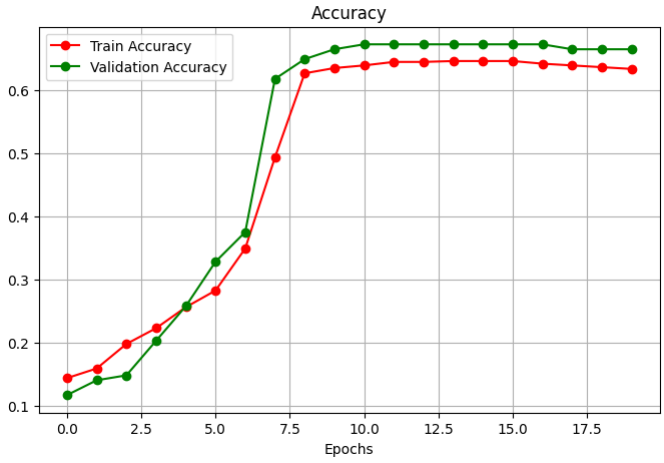
	precision	recall	f1-score	support
normal	1.00	1.00	1.00	31
inner brace fault	1.00	1.00	1.00	36
ball fault	1.00	1.00	1.00	45
outter brace fault	1.00	1.00	1.00	38
accuracy			1.00	150
macro avg	1.00	1.00	1.00	150
weighted avg	1.00	1.00	1.00	150

شکل ۱۶ classification report برای داده‌های آزمون

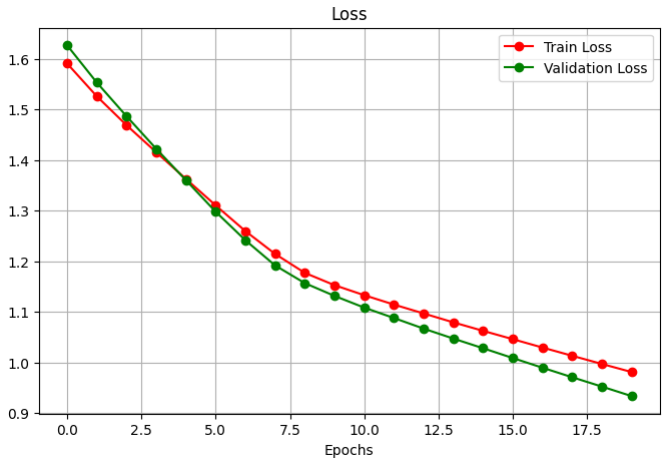
این تصویر نیز گزارش طبقه‌بندی را برای مجموعه‌داده آزمون نمایش می‌دهد.

۳-۲ آموزش با تابع هزینه و بهینه‌ساز دیگر

برای این بخش، از تابع هزینه KLDivergence برای خطا و از SGD به‌عنوان بهینه‌سازی استفاده شد. البته باید این نکته را ذکر کرد که برای استفاده از این تابع هزینه باید مقادیر target را به صورت one-hot-encode درآورد. برای این کار از تابع `to_categorical` استفاده شد.



شکل ۱۷ دقت بدست‌آمده حین آموزش



شکل ۱۸ خطا بدست‌آمده حین آموزش

همان‌طور که از شکل‌های ۱۶ و ۱۷ مشخص است، عملکرد مدل با استفاده از تابع هزینه و بهینه‌سازی جدید، باعث تضعیف مدل شده و عملکرد آن را بسیار کاهش داده‌است. برخلاف قسمت قبل، دقت به ۱۰۰ درصد نرسیده و مقدار خطا هم از ۰ فاصله دارد.

Classification report for Train data is:

	precision	recall	f1-score	support
normal	0.52	1.00	0.68	183
inner brace fault	0.96	0.52	0.67	192
ball fault	0.00	0.00	0.00	174
outter brace fault	0.65	1.00	0.79	173
accuracy			0.63	722
macro avg	0.53	0.63	0.54	722
weighted avg	0.54	0.63	0.54	722

Classification report for Validation data is:

	precision	recall	f1-score	support
normal	0.54	1.00	0.70	36
inner brace fault	1.00	0.45	0.62	22
ball fault	0.00	0.00	0.00	31
outter brace fault	0.76	1.00	0.87	39
accuracy			0.66	128
macro avg	0.58	0.61	0.55	128
weighted avg	0.56	0.66	0.57	128

شکل ۱۹ classification report برای داده‌های آموزش و اعتبارسنجی

شکل ۱۸ معیارهای مختلفی را برای سنجش عملکرد مدل ایجاد شده به ما نشان می‌دهد که همه آن‌ها مقادیر بسیار کمتری نسبت به حالت قبل نشان می‌دهند. می‌توان گفت که دقت مدل نسبت به حالت قبل به نصف رسیده‌است.

همان‌طور که از نتایج برمی‌آید، با عوض کردن تابع هزینه مدل، عملکرد مدل تغییر کرد که بر همین اساس می‌توان نتیجه گرفت که انتخاب تابع هزینه روی عملکرد مدل تاثیر دارد.

Classification report for Test data is:

	precision	recall	f1-score	support
normal	0.43	1.00	0.60	31
inner brace fault	0.00	0.00	0.00	36
ball fault	1.00	0.09	0.16	45
outter brace fault	0.51	1.00	0.68	38
accuracy			0.49	150
macro avg	0.49	0.52	0.36	150
weighted avg	0.52	0.49	0.35	150

شکل ۲۰ classification report برای داده‌های آزمون

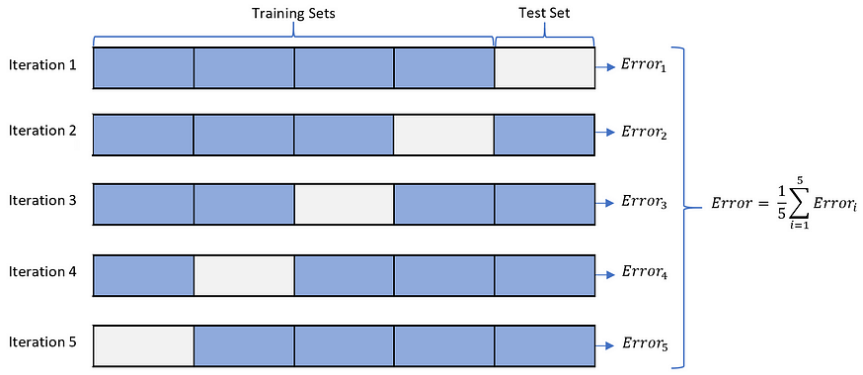
این تصویر عملکرد مدل دوم را برای داده‌های آزمون نمایش می‌دهد.

۲-۴ K-fold cros-validation & Stratified K-fold cross-validation

ابتدا روش K-fold cross-validation را توضیح می‌دهیم و ازآنجایی‌که روش stratifiled K-fold Cross-validation شباهت زیادی به روش قبلی دارد، بخش متفاوت آن را توضیح می‌دهیم.

K-Fold Cross-validation

در این روش، کل دیتاست به K بخش مساوی تقسیم می‌شود. آنگاه به تعداد K بار مدل آموزش می‌دهیم. در هر بار آموزش، یکی از foldها را به‌عنوان مجموعه ارزیابی (یا اعتبارسنجی) درنظر می‌گیریم و با استفاده از دیگر داده‌ها مدل را آموزش می‌دهیم. یعنی در هر دور آموزش، K-1 مجموعه به عنوان داده آموزش استفاده می‌شود.



شکل ۲۱ نمایش روش k-fold cross-validation [2]

شکل ۲۱ روش k-fold cross-validation را نمایش داده‌است. در هر دور یک fold را به عنوان آزمون انتخاب کرده و دیگران را به عنوان آموزش استفاده می‌کنیم.

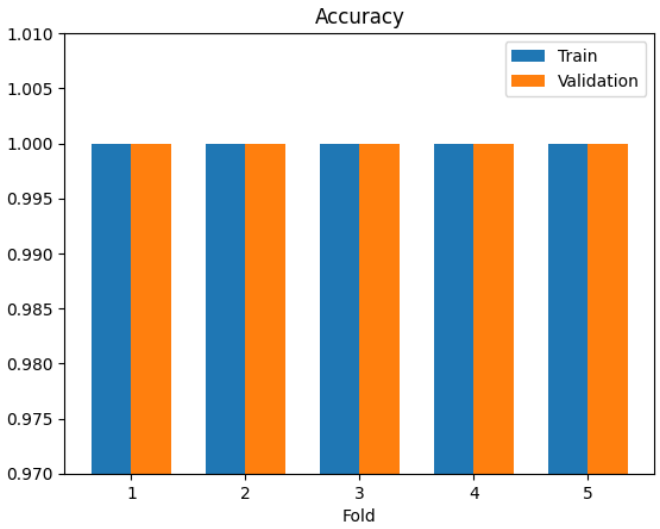
اگر قصد داشته‌ایم عملکرد مدل‌هایی با ساختارهای مختلف را مقایسه کنیم، از تمامی خطاهای بدست‌آمده با استفاده از این روش، میانگین می‌گیریم. اما برای انتخاب مدل نهایی یکی از k مدل‌های آموزش داده شده را با هم مقایسه می‌کنیم.

Stratified K-fold cross-validation

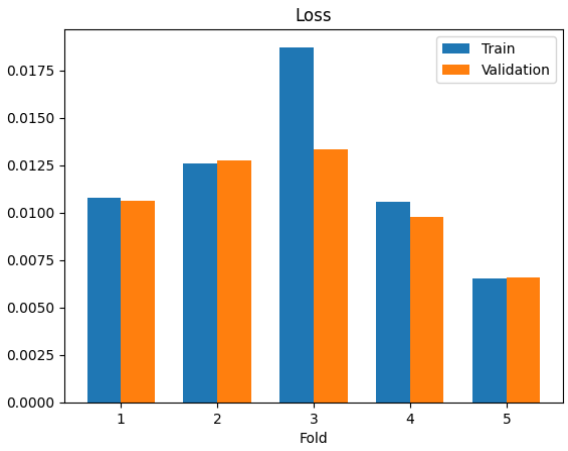
تفاوتی که این روش با روش قبلی دارد این است که در اینجا، برای هر fold ایجاد شده، تلاش می‌شود که پخش داده موجود، مانند پخش داده کل باشد. یعنی نسبت هر کلاس از داده در هر fold ایجاد شده، با نسبت پخش کلاس‌ها در کل داده یکسان است.

برای آموزش مدل از روش Stratified K-fold cross-validaton استفاده می‌کنیم. به‌جای استفاده از کل داده، صرفا از مجموعه داده آموزش برای این بخش استفاده می‌کنیم. دلیل استفاده از این روش این است که پخش داده‌ها را در تمامی foldها را یکسان درنظر می‌گیرد. با استفاده از دستور `StratifedKFold` از کتابخانه scikit-learn، یک شئ تعریف می‌کنیم

و با متد `split()`. ایندکس‌های fold ها را استخراج می‌کنیم تا از آن‌ها در حلقه آموزش استفاده کنیم. در هر دور حلقه آموزش، مدلی آموزش داده‌نشده ایجاد می‌شود و با استفاده از دسته داده‌های ایجاد شده آموزش داده‌می‌شود. درنهایت برای هر fold ایجادشده، یک دقت و خطا نهایی را گزارش می‌دهیم.



شکل ۲۲ نمایش دقت بدست‌آمده برای foldهای مختلف



شکل ۲۳ نمایش خطا بدست‌آمده برای foldهای مختلف

اگر بخواهیم مقادیر دقت را باهم مقایسه کنیم، هیچ تفاوتی وجود ندارد. اما بادقت در تصویر ۲۳، پی می‌بریم که برای حالت ۵ام کمترین میزان خطا برای هم داده آموزش و هم داده اعتبارسنجی (این مجموعه داده، با داده اعتبارسنجی که اول کار بوجود آمد متفاوت است) خواهیم داشت.

Classification report for Test data is:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	31
1	1.00	1.00	1.00	36
2	1.00	1.00	1.00	45
3	1.00	1.00	1.00	38
accuracy			1.00	150
macro avg	1.00	1.00	1.00	150
weighted avg	1.00	1.00	1.00	150

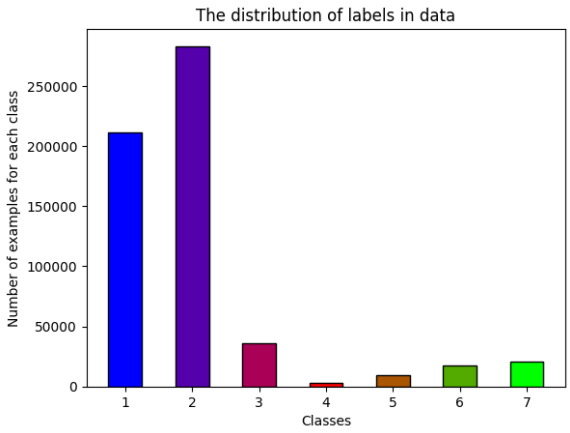
شکل ۲۴ classification report برای داده‌های آزمون

بعد از آموزش مدل‌ها، عملکرد بهترین مدل روی مجموعه‌داده آزمون که اول کار ایجاد شد تست شد و نتایج آن در تصویر ۲۴ به نمایش درآمده‌است.

نکته: random_state ای که برای تابع StratifiedKFold تعیین می‌کنیم، هیچ تاثیری روی تکرارپذیری تقسیم‌بندی داده‌ها نداشت، به همین دلیل تکرار نتایج یکسان ممکن نیست.

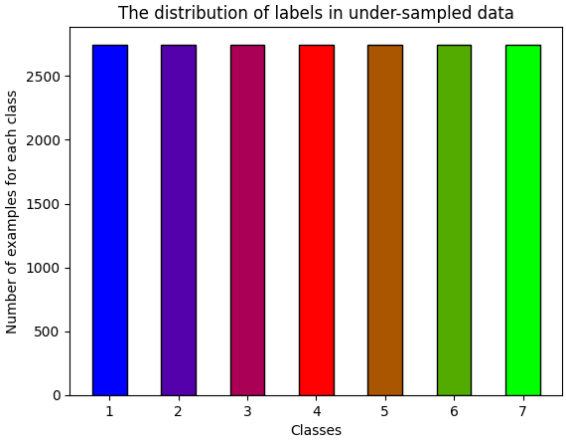
سؤال ۳

برای انجام این سؤال از داده‌های پوشش جنگلی استفاده کردیم. این داده‌ها در کتابخانه scikit-learn موجود هستند و برای دانلود این داده‌ها از دستور `fetch-covtype` استفاده شد.



شکل ۲۵ پخش داده‌ها در دیتاست پوشش جنگلی

باتوجه‌به شکل ۲۵، پخش داده‌های این دیتاست بسیار نامتعادل می‌باشد. اگر این عدم تعادل در داده آموزش نیز باشد، مدل ایجاد شده، مدلی مناسب نخواهد بود و دارای bias نسبت به کلاسی است که بیشترین تعداد داده را دارد. برای متعادل کردن این دیتاست از روش Under-sampling استفاده می‌کنیم.



شکل ۲۶ پخش داده‌ها بعد از under-sampling

شکل ۲۶ نشان می‌دهد که بعد از under-sampling تعداد داده‌ها در کلاس‌های مختلف یکسان شده و دیتاست متعادل شده‌است. اما under-sampling باعث کاهش تعداد داده‌ها می‌شود. با مقایسه تصاویر ۲۵ و ۲۶ می‌توانیم به این موضوع پی ببریم.

۳-۱ تقسیم‌بندی داده

برای تقسیم‌بندی داده‌ها از روش random sampling استفاده می‌کنیم. تابع تقسیم داده‌ها بر اساس این روش، `train_test_split` می‌باشد. در این روش داده‌های آموزش به صورت کاملا تصادفی انتخاب می‌شوند. در این بخش، ۱۵درصد داده به آزمون اختصاص داده‌شد و بقیه داده آموزش.

به جز random sampling، دو استراتژی دیگر نیز برای انتخاب داده آموزش و آزمون نیز وجود دارد:

- Cross Validation

در این روش دیتاست به بخش‌های مختلف تقسیم می‌شود و به تعداد بخش‌های تقسیم شده، مدل آموزش داده می‌شود و بهترین آن مدل‌ها انتخاب می‌شود.

اگر از این روش برای مقایسه الگوریتم‌های مختلف یا مدل‌هایی با فراپارامترهای مختلف استفاده شود، از خطا تمامی مدل‌ها میانگین گرفته می‌شود و الگوریتم‌ها با استفاده از این میانگین با هم مقایسه می‌شوند.

- Bootstrap

مبنای این روش‌ها انتخاب زیرمجموعه‌هایی از دیتاست اصلی با قابلیت جای‌گذاری داده‌ها می‌باشد. یعنی از یک دیتاست، یک زیردیتاست انتخاب می‌کنیم که برای انتخاب داده برای هر کدام از آن داده‌ها، انتخاب با جایگذاری داریم.

البته باید این نکته را ذکر کرد که عموماً این روش‌ها زمانی استفاده می‌شوند که تعداد داده‌های موجود برای آموزش کم باشد. در این حالت که تعداد داده‌ها، بعد از under-sampling، به اندازه مناسبی می‌باشد (برای هر کلاس ۲۷۰۰ داده)، نیازی احساس نمی‌شود که از دو روش دوم استفاده کنیم.

نکته: در بخش آموزش‌دادن مدل، چند روش باهم مقایسه شدند و از بهترین آن‌ها برای مراحل بعدی استفاده شد.

برای آموزش یک مدل درخت تصمیم‌گیری از سه استراتژی استفاده شد:

۱. آموزش داده‌های under sapmle شده
۲. آموزش داده‌ها بدون هیچ کار و عملیات اضافه‌ای
۳. آموزش داده‌ها با وزن دادن به الگوریتم (وزن هر کلاس $w_{class} = \frac{\#all\ data}{\#\ data\ within\ the\ class}$)
۴. آموزش داده‌ها با روش 5-fold Cross-validation
۵. آموزش داده‌ها با روش Stratified 5-fold Cross-validaiton

در ادامه به بررسی نتایج هرکدام از این استراتژی‌ها می‌پردازیم.

```
Model score for under-sampled data is: 0.7945

Model score for normal data is: 0.9421

The weight for each class is : [  3   2  16 212  61  33  28]
Model score for weighted data is: 0.8878

Model score for 5-fold CV is : 0.5580 --> (mean value)

Model score for Stratified 5-fold CV is 0.9345 --> (mean value)
```

شکل ۲۷ عملکرد استراتژی‌ها

For Normal model

	precision	recall	f1-score	support
1	0.94	0.94	0.94	31762
2	0.95	0.95	0.95	42618
3	0.93	0.94	0.94	5380
4	0.84	0.85	0.85	413
5	0.84	0.87	0.85	1477
6	0.88	0.89	0.89	2538
7	0.95	0.95	0.95	2964
accuracy			0.94	87152
macro avg	0.91	0.91	0.91	87152
weighted avg	0.94	0.94	0.94	87152

For Weighted model

	precision	recall	f1-score	support
1	0.94	0.94	0.94	31762
2	0.95	0.95	0.95	42618
3	0.92	0.92	0.92	5380
4	0.82	0.82	0.82	413
5	0.84	0.80	0.82	1477
6	0.88	0.86	0.87	2538
7	0.95	0.93	0.94	2964
accuracy			0.94	87152
macro avg	0.90	0.89	0.89	87152
weighted avg	0.94	0.94	0.94	87152

For Stratified 5-fold model

	precision	recall	f1-score	support
1	0.93	0.93	0.93	36016
2	0.94	0.95	0.94	48136
3	0.93	0.93	0.93	6075
4	0.84	0.82	0.83	467
5	0.83	0.82	0.82	1603
6	0.87	0.87	0.87	2966
7	0.95	0.94	0.94	3509
accuracy			0.94	98772
macro avg	0.90	0.89	0.90	98772
weighted avg	0.94	0.94	0.94	98772

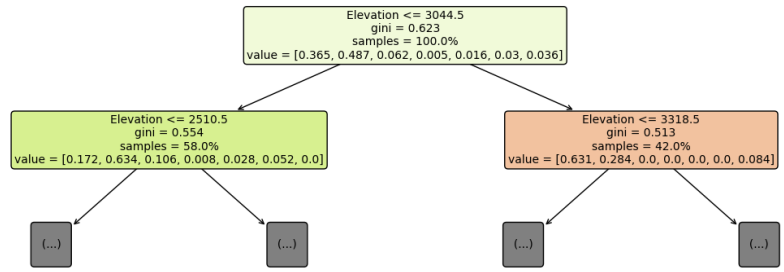
For Under sampled model

	precision	recall	f1-score	support
1	0.69	0.64	0.66	402
2	0.63	0.62	0.63	422
3	0.78	0.77	0.77	416
4	0.93	0.92	0.92	404
5	0.86	0.90	0.88	408
6	0.75	0.79	0.77	403
7	0.91	0.93	0.92	430
accuracy			0.79	2885
macro avg	0.79	0.79	0.79	2885
weighted avg	0.79	0.79	0.79	2885

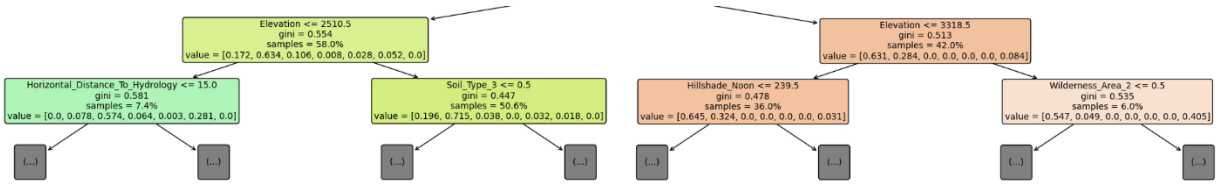
شکل ۲۸ classification report برای استراتژی‌های ۱تا۳ و ۵

شکل ۲۸ classification reportها به‌ازای استراتژی‌های ۱ تا ۳ را به نمایش گذاشته‌است. با این که عملکرد مـدل‌ها به برای کلاس‌های مختلف، متفاوت است و هیچ مدلی به طور کامل از بقیه عملکرد بهتری ندارد، ولی به صـورت کلـی می‌تـوانیم استراتژی دوم را انتخاب کنیم زیرا عمکرد کلی بهتری نسبت به دیگر استراتژی‌ها دارد.

بنابراین، برای ادامه محاسبات، به‌عنوان داده ورودی، از تمامی داده‌ها بدون هیچ وزنی استفاده می‌کنیم.



شکل ۲۹ تقسیم بندی برای اولی لایه



شکل ۳۰تقسیم‌بندی برای لایه دوم

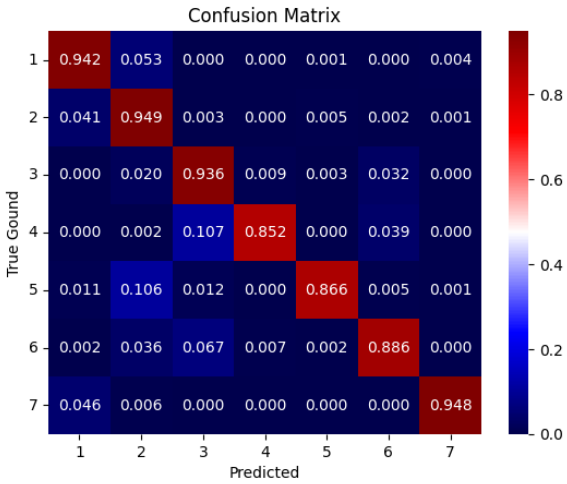
در تصاویر ۲۹ و ۳۰ تقسیم‌بندی لایه‌های مختلف را به نمایش گذاشته است. در هر بلوک اطلاعاتی مانند نام ویژگی، مقدار threshold، مقدار gini و تعداد نمونه‌های موجود در آن گره به نمایش درآمده‌اند.

این درخت آموزش‌داده‌شده تا ۴۳ لایه جداسازی کرده و تعداد ۲ نمونه در هر یک از برگ‌های آن موجود است. از این اطلاعات می‌توان چند نکته را دریافت:

- به دلیل زیاد بودن لایه‌های این درخت، نمی‌توان تمامی گره ها و برگ‌های آن را به صورت تصویر یا متن به نمایش درآورد.
- این مدل به‌شدت دچار overfitting شده‌است.

۳-۲ ماتریس درهم‌ریختگی و معیارها

ماتریس درهم‌ریختگی درخت تصمیم‌گیری به صورت زیر می‌باشد.

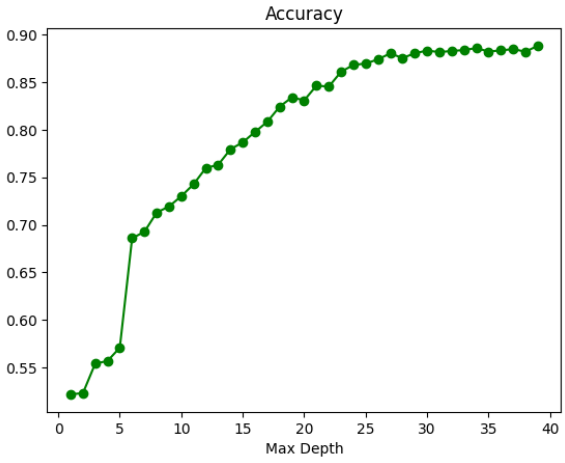


شکل ۳۱ ماتریس درهم‌ریختگی درخت‌تصمیم‌گیری

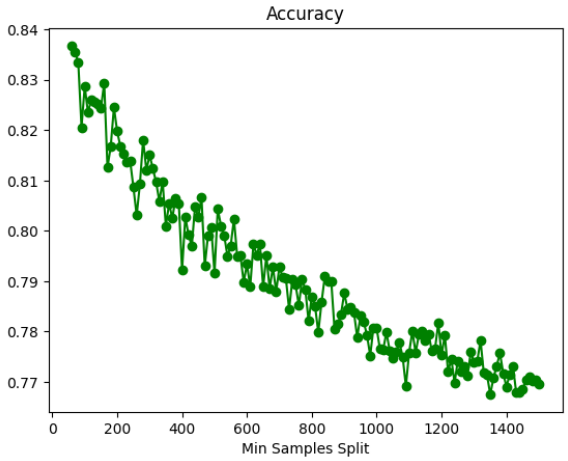
شکل ۳۱ ، ماتریس درهم‌ریختگی را برای داده‌های آزمون به نمایش گذاشته‌است.

جدول ۱ معیارهای محاسبه‌شده برای مدل

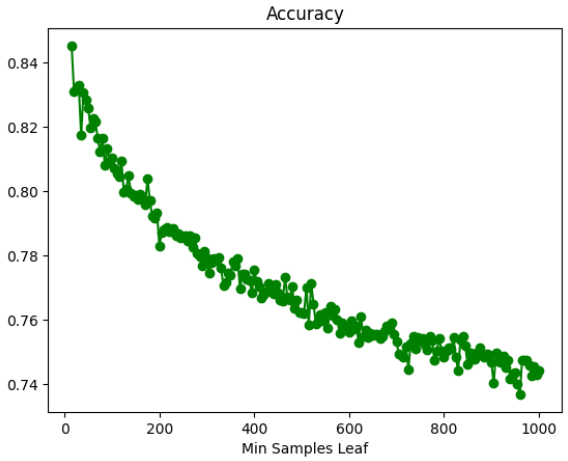
class	precision	recall	f1_score
1	0.9407	0.9422	0.9415
2	0.9517	0.9492	0.9505
3	0.9345	0.9359	0.9352
4	0.8441	0.8523	0.8482
5	0.8376	0.8659	0.8515
6	0.8847	0.8861	0.8854
7	0.9487	0.9477	0.9482



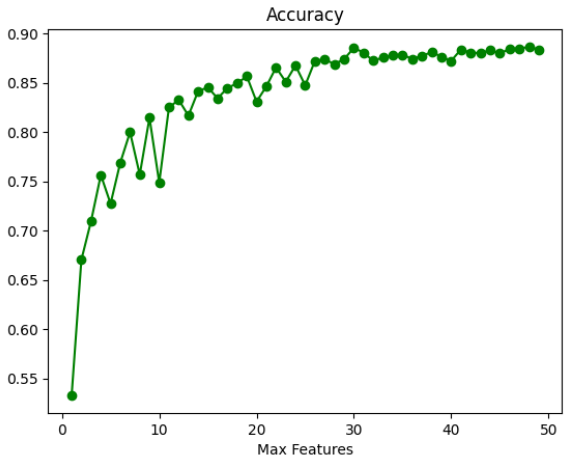
شکل ۳۲ دقت بر اساس max depth



شکل ۳۳دقت براساس *min samples split*



شکل ۳۴ دقت براساس *min samples leaf*



شکل ۳۵ دقت براساس *max features*

باتوجه‌به شکل‌های ۳۳ و ۳۴، می‌توانیم ببینیم که هر چه مدل را محدودتر می‌کنیم،درحال جلـوگیری از overfitting هستیم. اما از طرفی با توجه به شکل ۳۲، هرچه max depth را بیشتر می‌کنیم، دقت مدل روی داده آزمون بیشتر می‌شود.

با استفاده از نمودارهای عملکرد مدل برای مقادیر مختلف hyperparameterها که در نمودارها نمایش‌داده‌شده اسـت، بازه‌هایی برای این hyperparmametrها درنظر گرفته و با روش GridSearch بهترین آن‌ها را انتخاب می‌کنیم.

```
The best score on Test data is :0.9239
The best parameters are :{'max_depth': 36, 'max_features': 45, 'min_samples_leaf': 2}
```

شکل ۳۶ عملکرد مدل برای حالت gridsearch

۳-۳ adaboost & randomforest

روش‌هـــای AdaBoost و RandomForest از دســـته روش‌هـــای ensemble learning هســـتند. روش‌هـــای ensemble learning با استفاده از ایجاد زیرمجموعه‌هایی از مجموعه داده اصلی، با جایگشت داده، مدل‌های مختلف و مستقل ازهم‌ای تولید می‌کنند که باعث ایجاد خاصیت‌های زیر می‌شوند:

- کاهش واریانس (جلوگیری از overfitting): استفاده از نتیجه چندین مدل باعث می‌شود که عملکرد کلـی در جهت کاهش تاثیر نویز کمتر شود.
- Generalization: ایجاد مدل‌های مختلف باعث می‌شود که ابعاد مختلفی از توزیع آمـاری موجـود در داده را بیابیم.
- تأثیر کمتر نویز روی مدل: از آنجایی کـه مـدل‌ها روی زیرمجموعـه‌ها آمـوزش داده می‌شـوند، بـا درنظرگـرفتن عملکرد تمامی مدل‌ها، تاثیر نویز کاهش پیدامی‌کند..

البته که منفعت‌های روش‌های ensemble learning به اینجا ختم نمی‌شود و فواید دیگری هم دارد.

برای این بخش از الگوریتم جنگل تصادفی استفاده می‌کنیم و نتایج را با مدل بخش قبل مقایسه می‌نماییم.

```
The score of random forest model on test data is 0.9557

Classification report for the random forest model is
              precision    recall  f1-score   support

           1       0.97      0.94      0.95     31762
           2       0.95      0.98      0.96     42618
           3       0.95      0.96      0.95      5380
           4       0.90      0.87      0.89       413
           5       0.93      0.78      0.85      1477
           6       0.93      0.90      0.91      2538
           7       0.97      0.95      0.96      2964

    accuracy                           0.96     87152
   macro avg       0.94      0.91      0.93     87152
weighted avg       0.96      0.96      0.96     87152
```

شکل ۳۷ classification report مدل جنگل تصادفی

همان‌طور که در شکل ۳۶ مشاهده می‌شود، عملکرد نهایی مدل جنگل تصادفی نسبت به درخت تصمیم‌گیری، بهبود یافته است.

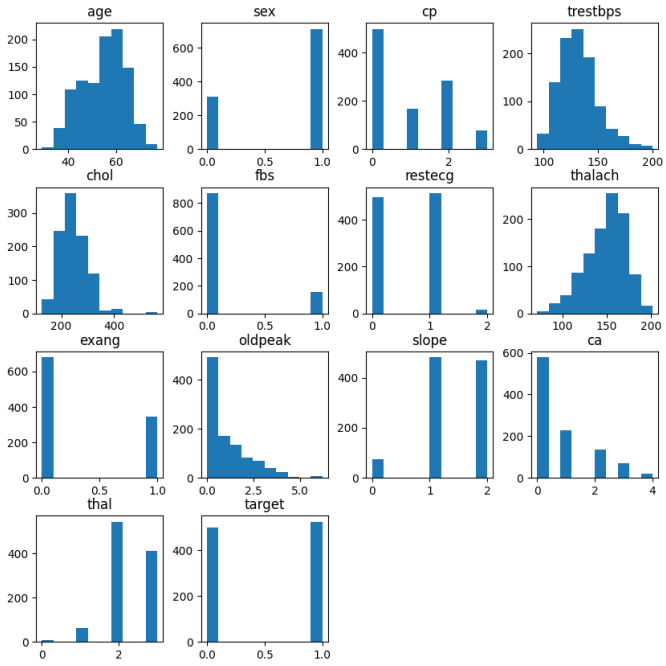
پارامتر تعیین شده برای این بخش، `max_depth=40` بود.

سؤال ۴

در اولین قدم دیتاست را در محیط colab دانلود می‌کنیم. با استفاده از کتابخانه pandas، فایل csv. را به‌صورت یک DataFrame می‌خوانیم.

```
There are 1025 examples in the dataset

Each example has 13 features
Name of the features are:
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal'],
      dtype='object')

Target classes are [0 1]
```

شکل ۳۸ برخی از ویژگی‌های دیتاست



شکل ۳۹ هیستوگرام ویژگی‌های موجود در دیتاست

از dataframe ایجادشده، داده‌های ورودی و خروجی را استخراج نموده و آن ها را ذخیره می‌نماییم.

```
The shape of input array is (1025, 13)
The shape of output array is (1025,)
```

شکل ۴۰ ابعاد آرایه‌های ورودی و خروجی

سپس داده‌ها را به نسبت ۸۵ به ۱۵ به دسته‌های آموزش و آزمون تقسیم می‌کنیم.

```
Shape of train data is (871, 13)
Shape of test data is (154, 13)
Shape of train target array is (871,)
Shape of test target array is (154,)
```

شکل ۴۱ ابعاد آرایه‌ها بعد از تقسیم‌بندی داده

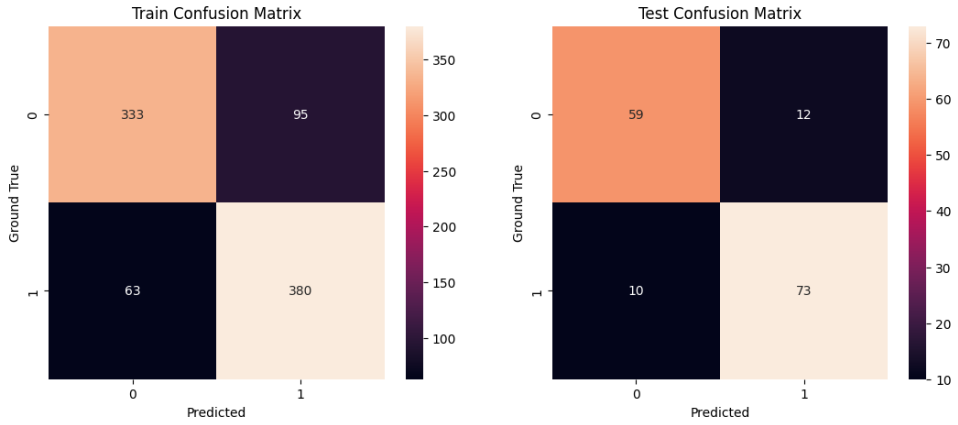
در قدم بعدی با استفاده از StandardScaler، داده‌های ورودی را نرمالیزه می‌نماییم.

سپس از کتابخانه scikit-learn، با استفاده از تابع `GaussianNB` یک مدل bayes را آموزش می‌دهیم.

```
The score on train data is 0.8186
The score on test data is 0.8571

Classsification report on train data is:

              precision    recall  f1-score   support

           0       0.84      0.78      0.81       428
           1       0.80      0.86      0.83       443

    accuracy                           0.82       871
   macro avg       0.82      0.82      0.82       871
weighted avg       0.82      0.82      0.82       871


Classsification report on test data is:

              precision    recall  f1-score   support

           0       0.86      0.83      0.84        71
           1       0.86      0.88      0.87        83

    accuracy                           0.86       154
   macro avg       0.86      0.86      0.86       154
weighted avg       0.86      0.86      0.86       154
```

شکل ۴۲ نتایج مدل bayes آموزش داده‌شده

شکل ۴۲ نتایج مدل آموزش داده‌شده را به نمایش گذاشته‌است. این نتایج شامل دقت مدل روی دیتا آموزش و آزمون بوده و classification report های مربوط به هر داده را نشان می‌دهد. با توجه به مقادیر f1-score، می‌توان بیان کرد که عملکرد مدل برای طبقه‌بندی هر کلاس مشابه است ولی عملکرد خیلی خوبی از خود نشان نمی‌دهد.



شکل ۴۳ ماتریس درهم‌ریختگی مدل برای داده‌های آزمون و آموزش

در شکل ۴۳ ماتریس پراکندگی برای داده‌های آموزش و آزمون نمایش داده‌شده است. در این ماتریس درهم‌ریختگی، تعداد موارد نشان داده شده است. با توجه به موارد false، می‌توان گفت که مدل عملکرد قابل قبولی ندارد.

درکل می‌توان گفت، عملکرد نه‌چندان خوب مدل می‌تواند ناشی از این باشد که پخش داده‌ها و ویژگی‌های این دیتاست، به‌صورت گوسی نبود. (از تصویر ۳۹ این ادعا اثبات می‌شود)

تفاوت مدل Micro و Macro:

Macro:

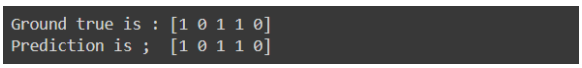
- برای هر کلاس، معیارها به طور مستقل محاسبه می‌شود.
- سپس، میانگین این معیارها در تمام کلاس‌ها محاسبه شده و به هر کلاس وزنی مساوی تعلق می‌گیرد.
- Micro average با همه کلاس‌ها بدون توجه به توزیع کلاس یا اندازه آنها به یک اندازه رفتار می‌کند.
- زمانی مناسب است که همه کلاس‌ها به یک اندازه مهم هستند و می‌خواهیم عملکرد کلی را بدون درنظرگـرفتن عدم تعادل کلاس ارزیابی کنید.

Micro:

- مجموع مثبت‌های درست، منفی‌های کاذب و مثبت‌های کاذب را در همه کلاس‌ها می‌شمارد.
- سپس، معیار با استفاده از این شمارش‌های کلی محاسبه می‌شود.
- میانگین‌گیری میکرو بر عملکرد کلی با درنظرگرفتن مشارکت‌های کل همه طبقات تأکید می‌کند.

زمانی مناسب است که می‌خواهیم عملکرد کلی را ارزیابی کنیم و نگران عدم تعادل طبقاتی هستیم.

مقایسه نتایج رندوم داده آزمون:



شکل ۴۴ مقایسه عملکرد مدل

شکل ۴۴ خروجی مدل برای ۵ داده تصادفی را با لیبل حقیقی آن‌ها مقایسه کرده‌است. در این حالت تمامی داده‌ها به صورت درست پیش‌بینی شده‌اند.

```
The inut data is:
[[ 58.    1.    2.  140.  211.    1.    0.  165.    0.    0.    2.    0.
    2. ]
 [ 63.    0.    0.  108.  269.    0.    1.  169.    1.    1.8   1.    2.
    2. ]
 [ 54.    1.    0.  140.  239.    0.    1.  160.    0.    1.2   2.    0.
    2. ]
 [ 49.    0.    1.  134.  271.    0.    1.  162.    0.    0.    1.    0.
    2. ]
 [ 50.    1.    0.  150.  243.    0.    0.  128.    0.    2.6   1.    0.
    3. ]]
```

شکل ۴۵ input مقادیر انتخاب شده