باسمه تعالی

گزارش کار تمرین دوم درس یادگیری ماشین

استاد درس : سرکار خانم دکتر ساجدی

نام دانشجو : ایمان کیانیان

شماره دانشجویی : ۶۱۰۳۰۰۲۰۳

۱- مقدمه

در این تمرین، ابتدا با استفاده از کد های تمرین قبلی(SVM،XGBoost، Random Forest و DecisionTree) یک مسئله دسته بندی را مدل کردیم . لازم به ذکر است داده های این مسئله با داده های مسئله ی قبل متفاوت است. سپس با استفاده از روش های دسته بندی بر پایه شبکه های عصبی MLP و ELM عمل دسته بندی را روی این داده ها انجام دادیم. سپس با استفاده از یک autoencoder ، کاهش ابعاد انجام دادیم و سپس روش های دسته بندی بر پایه شبکه های عصبی که قبلا نوشته بودیم را روی داده های کاهش بعد یافته اعمال کردیم و نتایج هر بخش را مقایسه کردیم. داده های این تمرین با داده های تمرین گذشته متفاوت بود و شامل ۷۵۵ ستون و ۷۵۶ سطر هستند که بیانگر وجود ۷۵۵ فیچر و ۷۵۶ سمپل است! تعداد سمپل ها با توجه به تعداد فیچر ها به شدت کم است و حتی از حالا هم میتوان فهمید در روش هایی که پیاده خواهیم کرد به مشکل اورفیت برخورد میکنیم و هدف ما در این تمرین باید این باشد که چطور اورفیت را مهار میکنیم. همچنین یک بردار ۷۵۶ تایی برای لیبل های داده ها داریم. در ابتدا داده ها را به ۲ بخش داده های آموزشی و داده های تست تقسیم کردیم. ۹۰ درصد از داده های ما یعنی ۶۸۰ سمپل مربوط به داده های آموزشی و ۱۰ درصد داده ها، معادل با ۷۶ سمپل مربوط به داده های تست است. با استفاده از الگوریتم های ذکر شده و داده های وارد شده، مدل ها را آموزش داده و سپس تست کردیم و نتایج را به تفکیک در ادامه گزارش خواهیم دید. در تمام مدل های این درخت از random_state=0 استفاده کردیم به دلیل اینکه میخواستیم در هر بار اجرا دقت واحدی بگیریم و هیچ چیز تصادفی نباشد (البته شاید باعث جلوگیری از دریافت نتایج بهتر شود. اما چون میخواستیم نتایج حالت مقایسه ای داشته باشد این کار را انجام دادیم). در این داده ها مدل baseline که همواره عدد ۱ را تشخیص دهیم دقت برابر تقریبا ۷۸ درصد را دارد . پس در این تمرین مدلی که ۷۸ درصد دقت دارد مدل خیلی خوبی نیست! هدف ما بالابردن این دقت است.

در این تمرین ابتدا پیش پردازش هایی انجام دادیم که در ادامه ذکر میکنیم.

۲- پیش پردازش ها:

در این قسمت به پیش پردازش هایی که قبل از اجرای مدل های قسمت های بعدی اورده ایم میپردازیم. در واقع همه ی مدل های آموزش داده شده در قسمت های بعدی توسط داده های پیش پردازش شده آموزش داده شده اند. بعد از اینکه داده های آموزشی و تست را با نسبت ۹۰ به ۱۰ تقسیم کردیم. روی داده های استاندارد سازی انجام میدهیم. در واقع داده ها را با میانگین صفر و واریانس ۱ در نظر میگیریم. فرمول استاندارد سازی به صورت زیر است:

$$X_{standard} = \frac{X - \mu(X)}{\sqrt{Var(X)}}$$

همچنین روش پیش پردازش Min Max که مقدار هر ویژگی را با توجه به مقیاس گذاری روی آن ویژگی تغییر میدهد نیز بر روی داده ها اعمال کردیم. مدل هایی که ذکر شد را با داده های آموزشی fit کردیم و بعد با آن مدل داده های اموزشی و تست را transform کردیم . یعنی در مرحله پیش پردازش داده ها ، از داده های تست برای آموزش استفاده نکردیم.

۳- مدل های مربوط به تمرین قبل:

در این روش با استفاده از کتابخانه sklearn، ابتدا داده های آموزشی تمرین دوم را به مدل های ارائه شده در تمرین ۱ با همان روش و پارامتر ها دادیم. با توجه به وضعیت داده ها انتظار overfit را داریم (حتی بیشتر از تمرین ۱ چون تعداد فیچر های ما و سمپل های آموزشی ما تعداد مناسبی را ندارند و به دلیل مبحثی که در curse of dimensionality ذکر میشود، چون تعداد داده ها زیاد و فیچر ها کم است در عملکرد دسته بندی دچار مشکل میشویم و احتمالا اورفیت خواهیم داشت.). در فایل ipynb. پیوست شده همراه با این فایل توضیحات به تفکیک میتوانید دقت روی هر بخش را به صورت جزئی و دقیق تر مشاهده بفرمایید اما برای خلاصه سازی این فایل توضیحات به آوردن نتایج کلی اکتفا میکنیم. دقت مدل های تمرین ۱، بر روی داده های آموزشی جدید:

model	Value of Parameters	Accuracy	Method	Precision	Recall	f-measure
Decision Tree	*Criterion : entropy* *Others : Click*	*1.0*	Macro	*1.0*	*1.0*	*1.0*
			Weighted	*1.0*	*1.0*	*1.0*
Random Forest	*n_estimators : 100* *Criterion : entropy* *Others : Click*	*1.0*	Macro	*1.0*	*1.0*	*1.0*
			Weighted	*1.0*	*1.0*	*1.0*
XGBoost	*Objective : multi:softprob* *random_state : 0* *Others : Click*	*1.0*	Macro	*1.0*	*1.0*	*1.0*
			Weighted	*1.0*	*1.0*	*1.0*
SVM	*Kernel : linear* *Default : Click and Click*	*0.969*	Macro	*0.97*	*0.95*	*0.96*
			Weighted	*0.97*	*0.97*	*0.97*

حال دقت مدل های تمرین ۱، بر روی داده های تست جدید:

model	Value of Parameters	Accuracy	Method	Precision	Recall	f-measure
Decision Tree	!!!Overfitting!!! *Criterion : entropy* *Others : Click*	*0.7763*	Macro	*0.69*	*0.73*	*0.71*
			Weighted	*0.80*	*0.78*	*0.79*
Random Forest	*n_estimators : 100* *Criterion : entropy* *Others : Click*	*0.895*	Macro	*0.88*	*0.81*	*0.83*
			Weighted	*0.89*	*0.89*	*0.89*
XGBoost	*Objective : multi:softprob* *random_state : 0* *Others : Click*	*0.921*	Macro	*0.92*	*0.84*	*0.88*
			Weighted	*0.92*	*0.92*	*0.92*
SVM	*Kernel : linear* *Default : Click and Click*	*0.895*	Macro	*0.84*	*0.87*	*0.85*
			Weighted	*0.90*	*0.89*	*0.90*

در نتیجه مشاهده میشود در سه روش اول مشکل overfitting با شدت و ضعف (در درخت تصمیم زیاد ، random forest کمتر و xgboost کمتر) وجود دارد. شدت overfitting تقریبا در روش decision tree زیاد است. وضعیت SVM در این تمرین نسبت به تمرین گذشته بهتر است و علت این قضیه داده های ما است. باید داده ها را در هر مسئله بررسی کرد تا دید میتوان برای آن داده ها مدل های خاصی متصور شد یا خیر. البته این SVM با SVM تمرین قبل در kernel اش متفاوت است.

با استفاده از 5-fold cross validation میتوانیم نتیجه قابل اتکا تری از این نتایج بگیریم. البته مشکلی که این روش دارد کمتر شدن داده ی train و بیشتر شدن مشکل curse of dimensionality هست و این راه حلی جز کمتر کردن فیچر ها یا بیشتر کردن داده ها ندارد. در این تمرین تلاشمان پیدا کردن مجموعه فیچر هایی است که مشکل overfitting را برایمان حل کند. البته در این نتیجه گیری این قضیه اعمال نشده است و در ادامه به تست کردن چند روش برای حل مشکل overfitting میپردازیم.

دقت این مدل ها در 5-fold cross validation به صورت زیر است:

model	Accuracy (5-fold cross validation)
Decision Tree	*0.7912* *(overfitting)*
Random Forest	*0.8691*
XGBoost	*0.8824* *(**best** tradeoff between bias and variance among others)* *(**best** choice among others)*
SVM	*0.8471*

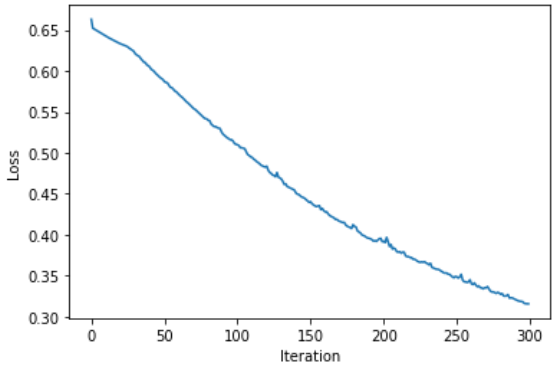
۴- مدل شبکه عصبی MLP (Multi-Layer Perceptron):

در این قسمت میخواهیم با استفاده از چندین مدل شبکه عصبی MLP داده های خودمان را آموزش دهیم. در این قسمت max_iter که حداکثر تعداد تکرار حلقه برای آموزش شبکه است را برابر مقدار ثابت ۳۰۰ در نظر میگیریم. طبیعتا هر چه مقدار max_iter بیشتر باشد احتمالا مدل بهتری خواهیم داشت. (البته ممکن است مشکل overfitting را هم ایجاد کند.) با کم گرفتن مقدار max_iter نیز احتمال وقوع underfitting بالا میرود. alpha که پارامتر regularization است را مقدار بالا 0.01 در نظر گرفته که از مشکل overfitting تا حدی جلوگیری کنیم (با تست های زیاد به این نتیجه رسیدیم. میتوانستیم بالاتر هم بگیریم ولی دقت روی تست هم پایین می آمد یعنی اجازه ی learn شدن مدل را نمی داد). برای activation function از relu استفاده کردیم و برای optimizer از adam استفاده کردیم. تست های مختلف را به ترتیب مینویسیم:

- **تست اول – مدل MLP ۳ لایه با ۱۰ نورون در لایه اول و ۵ نورون در لایه دوم :**

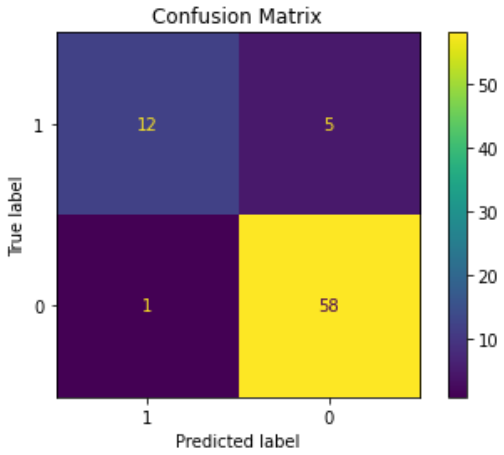
در این تست ابتدا یک مدل ۳ لایه با ۱۰ نورون در لایه اول ، ۵ نورون در لایه دوم و ۲ نورون در لایه خروجی مدل را آموزش دادیم . دقت روی داده های آموزشی و تست به صورت زیر بود:

```
Accuracy On Training Data  = 0.9264705882352942
Accuracy On Test Data  = 0.9210526315789473
```



که به نظر دقت خوبی است. با توجه به اینکه تقسیم داده ها ۹۰ به ۱۰ است شاید کمی در محاسبه دقت خطا وجود داشته باشد اما با استفاده از یک 5-fold-cross- validation برابر 0.8426 است و تا حد بیشتری میتوانیم به این نتیجه اعتماد کنیم. البته چون تعداد داده ها کم است باز هم کمی دچار افت دقت میشویم اما چاره ای نیست!

جدول confusion matrix برای داده های تست به صورت زیر خواهد بود:



و همچنین دقت های ذکر شده روی داده تست برابر زیر است:

	precision	recall	f1-score	support
Class 0	0.92	0.71	0.80	17
Class 1	0.92	0.98	0.95	59
accuracy			0.92	76
macro avg	0.92	0.84	0.88	76
weighted avg	0.92	0.92	0.92	76

با بررسی دقت های زیر که برای داده های آموزشی محاسبه شده اند میتوان فهمید که از underfitting رنج میبریم:

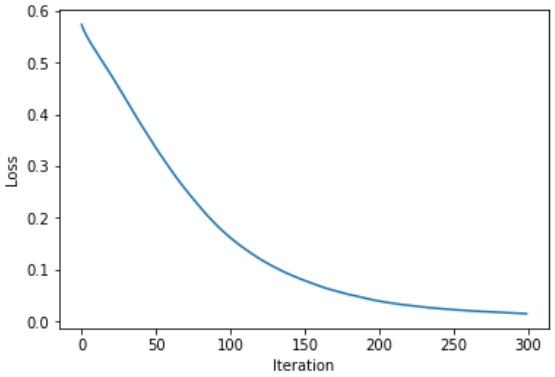
	precision	recall	f1-score	support
Class 0	1.00	0.71	0.83	175
Class 1	0.91	1.00	0.95	505
accuracy			0.93	680
macro avg	0.95	0.86	0.89	680
weighted avg	0.93	0.93	0.92	680

به دقت recall زمانی که کلاس 0 داریم توجه کنید. دقت به شدت پایین است و مناسب نیست باید به دنبال این باشیم این دقت را بالا بیاوریم.

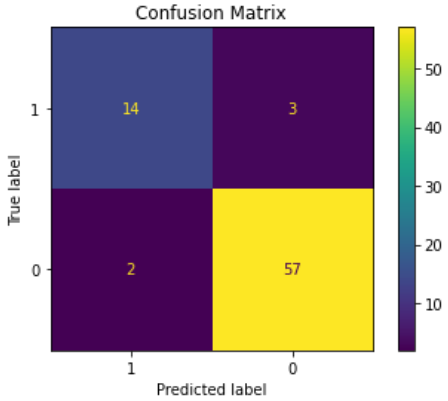
- **تست دوم – استفاده از PCA و MLP تست قبلی:**

شاید کمی روشن باشد که ما دچار اورفیت میشویم و علت این اورفیت بالا، وجود تعداد فیچر های بالا نسبت به تعداد دیتا هاست. برای مدیریت این مشکل در این آزمایش ابتدا یک PCA با ۷۰ مولفه اصلی آموزش دادیم و سپس داده های تولید شده را به MLP ذکر شده در تمرین اول داده و عمل دسته بندی را انجام دادیم. نتایج به صورت زیر است:

```
Accuracy On Training Data  = 1.0
Accuracy On Test Data  = 0.9342105263157895
```



واضح است که overfitting همچنان وجود دارد و نتیجه ای که در پایان این تست می آوریم قابل اتکا تر است (weighted و macro). نتیجه بدست آمده توسط 5-fold cross validation برابر 0.8706 است. ماتریس confusion نیز برای داده های تست به صورت زیر است:



واضح است که روی داده های تست اوضاع کمی بهتر است. حال دقت دقیق تر را روی داده های تست بدست می آوریم :

	precision	recall	f1-score	support
Class 0	0.88	0.82	0.85	17
Class 1	0.95	0.97	0.96	59
accuracy			0.93	76
macro avg	0.91	0.89	0.90	76
weighted avg	0.93	0.93	0.93	76

حال با نشان دادن دقت روی داده های آموزشی میتوان متوجه overfitting شد:

	precision	recall	f1-score	support
Class 0	1.00	1.00	1.00	175
Class 1	1.00	1.00	1.00	505
accuracy			1.00	680
macro avg	1.00	1.00	1.00	680
weighted avg	1.00	1.00	1.00	680

در بین تست هایی که انجام دادیم این تست بهترین عملکرد را در معیار های اندازه گیری دقت داشته است. (روی داده ی تست خوب عمل کرده).

- **تست سوم – استفاده از PCA قبلی و MLP با ۳ نورون در لایه اول ، ۳ نورون در لایه دوم و ۲ نورون در لایه سوم:**

در این قسمت دقت روی داده های آموزشی به صورت زیر است:

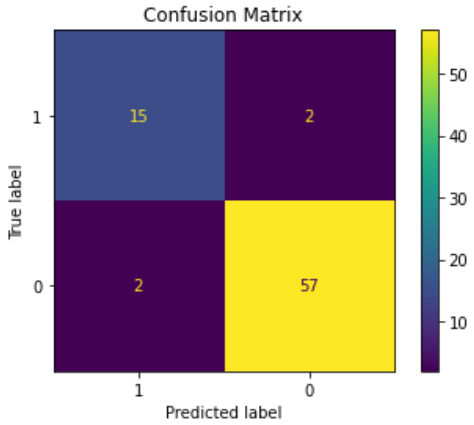
	precision	recall	f1-score	support
Class 0	0.93	0.84	0.88	175
Class 1	0.95	0.98	0.96	505
accuracy			0.94	680
macro avg	0.94	0.91	0.92	680
weighted avg	0.94	0.94	0.94	680

و دقت روی داده های تست به صورت زیر است:

	precision	recall	f1-score	support
Class 0	0.88	0.88	0.88	17
Class 1	0.97	0.97	0.97	59
accuracy			0.95	76
macro avg	0.92	0.92	0.92	76
weighted avg	0.95	0.95	0.95	76

که این یعنی این شبکه از شبکه در تست دوم بهتر است و overfitting تا حد زیادی از بین رفته است.

دقت 5-fold-cross-validation برابر 0.8485 است که کمی کمتر از تست دو است.



- **تست چهارم – استفاده از MLP دو لایه با استفاده از ۱۰ نورون در لایه اول و ۲ نورون در لایه خروجی:**

چون تعداد حالت ها و تست ها بسیار زیاد است از آوردن نتایج دقیق هر تست خودداری میکنیم. در این حالت دقت بسیار کم است و underfitting داریم اصلا MLP ما نتوانسته داده ها را مدل کند.

دقت روی داده آموزشی:

	precision	recall	f1-score	support
Class 0	0.00	0.00	0.00	175
Class 1	0.74	1.00	0.85	505
accuracy			0.74	680
macro avg	0.37	0.50	0.43	680
weighted avg	0.55	0.74	0.63	680

از آوردن بقیه ی تست ها در اینجا خودداری میکنیم. میتوانید در فایل ipynb. همراه با این فایل توضیحات، تست های بیشتری را مشاهده کنید که نتیجه گیری از تست های مختلف به اینصورت بود که ، هر چه مدل پیچیده تر باشد احتمال overfitting بالاتر میرود و هر چه مدل ساده تر باشد احتمال underfitting بالا می رود. البته نتیجه بهتری از مدل های دیگر نگرفتیم و البته فقط ایراد از مدل ها نیست ، مشکل اصلی ما در این تمرین که کمی مقایسه مدل ها را سخت میکند، تعداد کم داده هاست.

خلاصه ای از همه ی تست ها روی داده های تست بصورت زیر است:

Test Number	Accuracy	Method	Precision	Recall	f-measure
1	*0.9211*	Macro	*0.92*	*0.84*	*0.88*
		Weighted	*0.92*	*0.92*	*0.92*
2	*0.9342*	Macro	*0.91*	*0.89*	*0.90*
		Weighted	*0.93*	*0.93*	*0.93*
3	*0.9474*	Macro	*0.92*	*0.92*	*0.92*
		Weighted	*0.95*	*0.95*	*0.95*
4	*0.7763 (underfitting)*	Macro	*0.37*	*0.50*	*0.43*
		Weighted	*0.55*	*0.74*	*0.63*

۵- ELM (Extreme Learning Machine) :

در این قسمت از آنجایی که پکیج خوبی برای ELM وجود نداشت، آن را پیاده سازی کردیم. البته این ELM یک شبکه دو لایه است . ما فقط اندازه لایه اول مخفی را تایین میکنیم و لایه آخر (خروجی) به اندازه تعداد کلاس ها که ۲ تا هستند ، نورون دارد. تست های زیادی انجام شد . نکته مثبت این شبکه عصبی زمان کم آموزش دیدن آن است که خیلی خوب است.

- **تست اول – ELM با ۸۰ نورون در لایه مخفی اول :**

با انجام این تست متوجه شدیم که underfitting اتفاق می افتد. یعنی تعداد نورون ها برای مدلسازی داده ها کم است و مدل ما بسیار ساده است و قابلیت زیادی ندارد. بنابراین باید تعداد نورون هارا در تست بعدی زیاد کنیم. دقت روی داده های آموزشی:

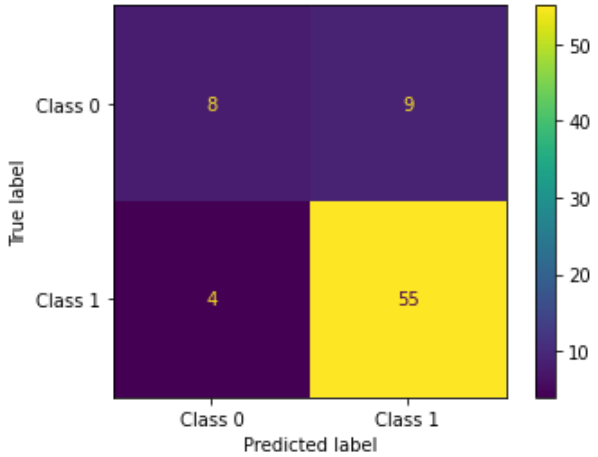
	precision	recall	f1-score	support
Class 0	0.80	0.54	0.64	175
Class 1	0.86	0.95	0.90	505
accuracy			0.85	680
macro avg	0.83	0.75	0.77	680
weighted avg	0.84	0.85	0.84	680

حال دقت روی داده های تست را مشاهده میکنیم:

	precision	recall	f1-score	support
Class 0	0.67	0.47	0.55	17
Class 1	0.86	0.93	0.89	59
accuracy			0.83	76
macro avg	0.76	0.70	0.72	76
weighted avg	0.82	0.83	0.82	76

دقت در 5-Fold Cross Validation برابر 0.8118 است.

و confusion matrix بر روی داده های تست به صورت زیر است:



واضح است که در این تست underfitting داریم (به دقت های روی داده های آموزشی دقت کنید).

- **تست دوم – ELM با ۱۰۰۰ نورون در لایه اول مخفی :**

در این تست مشاهده شد که ۱۰۰۰ نورون مدل ما را بسیار پیچیده تر از حد میکند و باعث overfitting زیادی میشود که اصلا مناسب نیست. بنابراین باید مدل را ساده تر کنیم. دقت برای داده های آموزشی:

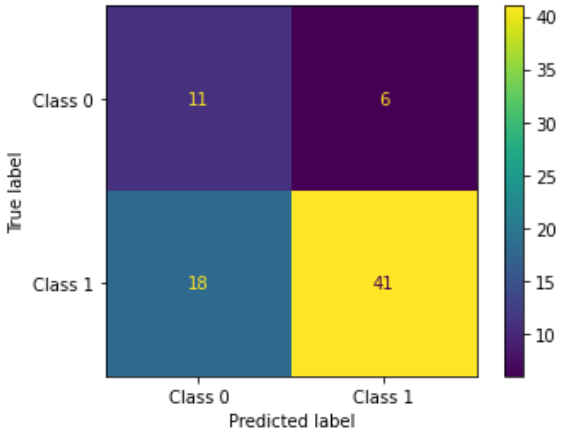
	precision	recall	f1-score	support
Class 0	1.00	1.00	1.00	175
Class 1	1.00	1.00	1.00	505
accuracy			1.00	680
macro avg	1.00	1.00	1.00	680
weighted avg	1.00	1.00	1.00	680

و دقت برای داده های تست بصورت زیر است:

	precision	recall	f1-score	support
Class 0	0.38	0.65	0.48	17
Class 1	0.87	0.69	0.77	59
accuracy			0.68	76
macro avg	0.63	0.67	0.63	76
weighted avg	0.76	0.68	0.71	76

دقت در 5-Fold Cross Validation برابر 0.6779 است.

و همینطور confusion matrix برای داده های تست:



که نشان میدهد اورفیت زیادی در این مدل وجود دارد برای همین باز هم تعداد نورون هارا در تست ۳ کمتر میکنیم و نتیجه تست را مشاهده خواهیم کرد.

- **تست سوم – ELM با ۲۱۰ نورون در لایه اول مخفی :**

در این قسمت مقدار اورفیت کم میشود در واقع اورفیت مقدار زیادی ندارد . نتیجه دقت روی داده های آموزشی:

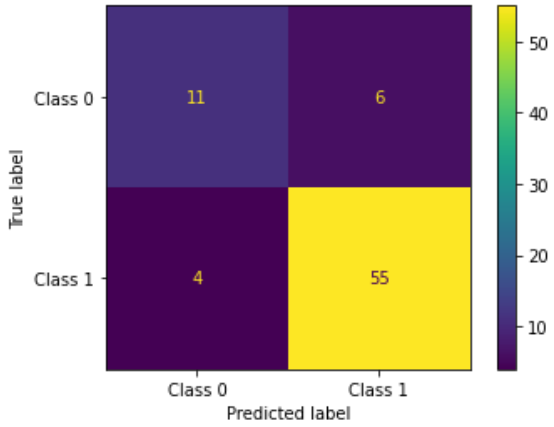
	precision	recall	f1-score	support
Class 0	0.91	0.78	0.84	175
Class 1	0.93	0.97	0.95	505
accuracy			0.92	680
macro avg	0.92	0.88	0.89	680
weighted avg	0.92	0.92	0.92	680

همچنین دقت روی داده تست:

	precision	recall	f1-score	support
Class 0	0.73	0.65	0.69	17
Class 1	0.90	0.93	0.92	59
accuracy			0.87	76
macro avg	0.82	0.79	0.80	76
weighted avg	0.86	0.87	0.87	76

و دقت در 5-Fold Cross Validation برابر 0.8368 است که خوب است.

و همینطور confusion matrix برای داده های تست:



همینطور که مشاهده شد دقت تست بالاتر آمد به دلیل اورفیت کمتر. اما نکته عجیب این است که هر چقدر از ۱۰۰۰ نورون بیشتر میشویم مقدار اورفیت کمتر میشود. یعنی بالاتر بردن تعداد نورون ها بعد از مدتی که اورفیت را ایجاد کرد دوباره باعث کمتر شدن اورفیت میشود. این نکته ی تست هایی بود که انجام داده ام. به تست های بعد دقت کنید:

- **تست چهارم – ELM با ۱۰۰۰۰ نورون در لایه اول مخفی :**

تعداد نورون های تست دوم را ۱۰ برابر کردیم! انتظار داشتیم که اورفیت مشکل بیشتری ایجاد کند چون مدل پیچیده تر و با قابلیت مدلسازی بیشتر میشود اما اینطور نشد! نتیجه مشاهدات به صورت زیر است.

دقت روی داده آموزشی:

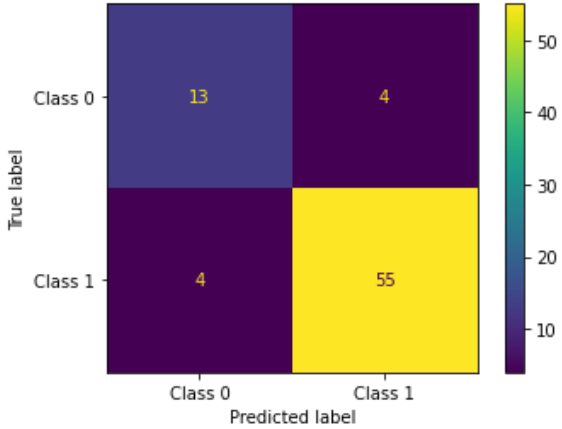
	precision	recall	f1-score	support
Class 0	1.00	1.00	1.00	175
Class 1	1.00	1.00	1.00	505
accuracy			1.00	680
macro avg	1.00	1.00	1.00	680
weighted avg	1.00	1.00	1.00	680

دقت روی داده تست:

	precision	recall	f1-score	support
Class 0	0.83	0.88	0.86	17
Class 1	0.97	0.95	0.96	59
accuracy			0.93	76
macro avg	0.90	0.92	0.91	76
weighted avg	0.94	0.93	0.93	76

دقت در 5-Fold Cross Validation برابر 0.8588 است.

همچنین confusion matrix برای داده تست بصورت زیر است:



البته این تست شانسی نبوده و در فایل پایتون پیوست شده کنار این فایل توضیحات میتوانید تست های دیگری با وضعیت مشابه ببینید (تست با ۵۰۰۰۰ نورون هم انجام شده و به همین صورت دقت بالاتر رفته است.) بهترین تستی که توانستیم در این قسمت انجام دهیم تست زیر است که از PCA بهره گرفتیم و ابتدا بعد را کم کردیم. سپس یک ELM با ۲۰۰۰ نورون مخفی را آموزش دادیم و دقت بسیار بهتری از حالتی که از ۵۰۰۰۰ نورون در تستی که در فایل پایتون وجود دارد ، بدست آوردیم.

- **تست پنجم – اعمال PCA روی داده های آموزشی و استفاده از ELM با ۲۰۰۰ نورون در لایه اول مخفی :**

در این تست ابتدا یک PCA با ۷۰ مولفه اصلی را با داده های آموزشی آموزش دادیم و سپس داده ها را کاهش بعد دادیم و بعد از آن داده های بدست آمده را به یک شبکه عصبی ELM با ۲۰۰۰ نورون دادیم . نتیجه بسیار خوبی گرفتیم که به صورت زیر است.

دقت روی داده آموزشی :

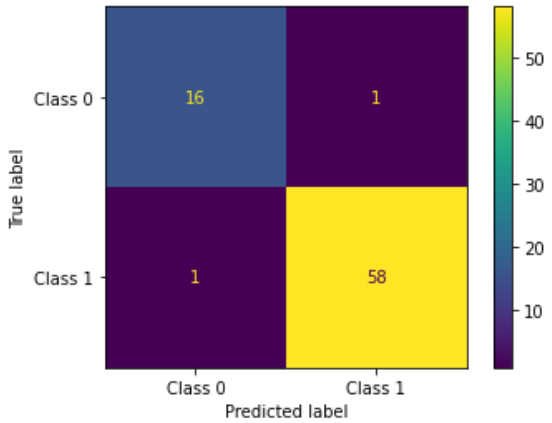
	precision	recall	f1-score	support
Class 0	1.00	1.00	1.00	175
Class 1	1.00	1.00	1.00	505
accuracy			1.00	680
macro avg	1.00	1.00	1.00	680
weighted avg	1.00	1.00	1.00	680

دقت روی داده تست:

	precision	recall	f1-score	support
Class 0	0.94	0.94	0.94	17
Class 1	0.98	0.98	0.98	59
accuracy			0.97	76
macro avg	0.96	0.96	0.96	76
weighted avg	0.97	0.97	0.97	76

دقت در `5-Fold Cross Validation` برابر `0.9103` است.

و همچنین `confusion matrix` برای داده تست بصورت زیر است:



که دقت بسیار خوبی است. کمی اورفیت وجود دارد ولی دقت خیلی خوبی بدست آمد.

در تست های قبل حالت های مختلف `ELM` را تست کردیم. شاید با دستکاری پارامتر ها بتوان دقت های بهتری هم بدست آورد اما در قالب این تمرین جای نمیگیرد. گزارش کلی دقت روی داده های تست این قسمت از تمرین به صورت زیر است:

Test Number	Accuracy	Method	Precision	Recall	f-measure	K-Fold Cross Validation
1	*0.8289*	Macro	*0.76*	*0.70*	*0.72*	*0.8118*
		Weighted	*0.82*	*0.83*	*0.82*	
2	*0.6842* *(Overfitting)*	Macro	*0.63*	*0.67*	*0.63*	*0.6779*
		Weighted	*0.76*	*0.68*	*0.71*	
3	*0.8684*	Macro	*0.82*	*0.79*	*0.80*	*0.8367*
		Weighted	*0.86*	*0.87*	*0.87*	
4	*0.9342*	Macro	*0.90*	*0.92*	*0.91*	*0.8588*
		Weighted	*0.94*	*0.93*	*0.93*	
5 (Excellent)	*0.9737*	Macro	*0.96*	*0.96*	*0.96*	*0.9103*
		Weighted	*0.97*	*0.97*	*0.97*	

۶- Auto Encoder در دسته بندی:

در این قسمت ابتدا با استفاده از معماری های مختلف یک auto encoder میسازیم و سپس از قسمت encoder آن برای dimensionality reduction استفاده میکنیم تا کاری شبیه به PCA را انجام دهیم. سپس داده هایی که کاهش بعد یافته اند را به یک شبکه عصبی ELM یا MLP میدهیم تا کار دسته بندی را برایمان انجام دهد و سپس گزارش هر تست را بیان میکنیم. انتظار داریم در این تمرین میزان اورفیت در دسته بندی کم باشد. تست های زیادی انجام داده ایم ولی بخش کوچکی را توضیحات این بخش می آوریم که ذکر خواهد شد. در مورد پارامتر های هر تست چون مقادیر پارامتر ها زیاد است لطفا به فایل پایتون سر بزنید و پارامتر ها را به صورت دقیق مشاهده فرمایید.

- **تست ۱ - شبکه عصبی auto encoder با ۲ لایه انکودر و ۲ لایه دیکدر:**

در این تست ابتدا یک اتو انکدر تعریف کردیم . اتو انکودر به صورت زیر است:

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 input_1 (InputLayer)         [(None, 753)]             0

 dense (Dense)                (None, 600)               452400

 dense_1 (Dense)              (None, 400)               240400

 dense_2 (Dense)              (None, 600)               240600

 dense_3 (Dense)              (None, 753)               452553

=================================================================
Total params: 1,385,953
Trainable params: 1,385,953
Non-trainable params: 0
_________________________________________________________________
```

آموزش این شبکه در ۱۰۰۰ تکرار با batch های ۳۲ تایی صورت گرفت. سپس از قسمت encoder که تا لایه سوم است (۴۰۰ نورون) استفاده کردیم و یک ELM را به آن متصل کردیم. تست های مختلفی انجام دادیم و بهترین حالت زمانی بود که ۱۰۰۰۰ نورون در لایه مخفی ELM وجود داشت. نتیجه به صورت زیر است.

دقت روی داده آموزشی:

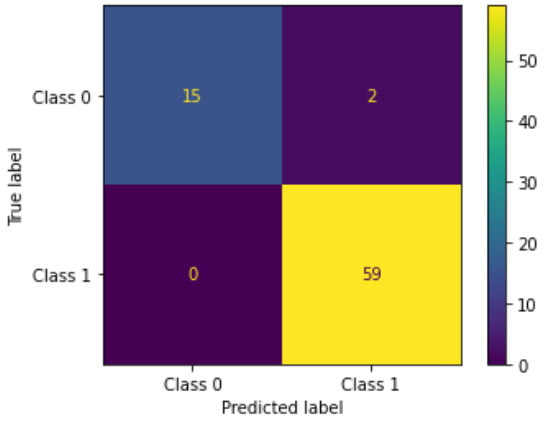
	precision	recall	f1-score	support
Class 0	1.00	1.00	1.00	175
Class 1	1.00	1.00	1.00	505
accuracy			1.00	680
macro avg	1.00	1.00	1.00	680
weighted avg	1.00	1.00	1.00	680

دقت روی داده تست:

	precision	recall	f1-score	support
Class 0	1.00	0.88	0.94	17
Class 1	0.97	1.00	0.98	59
accuracy			0.97	76
macro avg	0.98	0.94	0.96	76
weighted avg	0.97	0.97	0.97	76

دقت در 5-Fold Cross Validation برابر 0.9162 است.

و همچنین confusion matrix برای داده تست بصورت زیر است:



- **تست ۲ - شبکه عصبی auto encoder با ۸ لایه انکودر و ۸ لایه دیکدر:**

چون میخواستیم تعداد فیچر های استخراج شده از داده ها را با این روش کوچکتر کنیم (۵۰ تا) از لایه های متعدد استفاده کردیم که کم کم این اتفاق رخ دهد تا دچار خطا نشویم. سپس داده های تغییر بعد داده شده را به یک ELM با ۲۰۰۰۰ نورون در لایه مخفی دادیم اما در این حالت بهبودی در وضعیت ایجاد نشد به همین علت از آوردن جزئیات این تست در این بخش صرف نظر میکنیم و در آینده در جدول جزئیات تست را مینویسیم. فقط ساختار این شبکه عصبی را در این قسمت ذکر میکنیم:

Layer (type)	Output Shape	Param #
input_19 (InputLayer)	[(None, 753)]	0
dense_141 (Dense)	(None, 700)	527800
dense_142 (Dense)	(None, 600)	420600
dense_143 (Dense)	(None, 500)	300500
dense_144 (Dense)	(None, 400)	200400
dense_145 (Dense)	(None, 300)	120300
dense_146 (Dense)	(None, 200)	60200
dense_147 (Dense)	(None, 100)	20100
dense_148 (Dense)	(None, 50)	5050
dense_149 (Dense)	(None, 100)	5100
dense_150 (Dense)	(None, 200)	20200
dense_151 (Dense)	(None, 300)	60300
dense_152 (Dense)	(None, 400)	120400
dense_153 (Dense)	(None, 500)	200500
dense_154 (Dense)	(None, 600)	300600
dense_155 (Dense)	(None, 700)	420700
dense_156 (Dense)	(None, 753)	527853

Total params: 3,310,603
Trainable params: 3,310,603
Non-trainable params: 0

- **تست ۳ - شبکه عصبی auto encoder با ۳ لایه انکودر و ۳ لایه دیکدر:**

در این روش هم دقت بیشتری از تست ۱ حاصل نشد. بنابراین جزئیات دقت تست ها و confusion matrix را ذکر نمیکنیم و فقط به جزئیات طراحی شبکه عصبی میپردازیم. شبکه عصبی auto encoder طراحی شده در این تست به صورت زیر است:

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 input_19 (InputLayer)        [(None, 753)]             0

 dense_162 (Dense)            (None, 500)               377000

 dense_163 (Dense)            (None, 300)               150300

 dense_164 (Dense)            (None, 200)               60200

 dense_165 (Dense)            (None, 300)               60300

 dense_166 (Dense)            (None, 500)               150500

 dense_167 (Dense)            (None, 753)               377253

=================================================================
Total params: 1,175,553
Trainable params: 1,175,553
Non-trainable params: 0
_________________________________________________________________
```

در این قسمت به بررسی تاثیر auto encoder بر روی عمل دسته بندی پرداختیم. حال در جدول زیر دقت را در ۳ تست انجام شده بر روی داده های تست مشاهده میکنید. لازم به ذکر است لزوما این مشاهدات بهترین مشاهدات قابل ارائه نیست اما مشاهداتی است که درون این تمرین تست شد.

Test Number	Accuracy	Method	Precision	Recall	f-measure	K-Fold Cross Validation
1 (Best)	*0.9737*	Macro	*0.98*	*0.94*	*0.96*	*0.9162*
		Weighted	*0.97*	*0.97*	*0.97*	
2	*0.9211*	Macro	*0.88*	*0.91*	*0.89*	*0.8382*
		Weighted	*0.93*	*0.92*	*0.93*	
3	*0.9079*	Macro	*0.87*	*0.86*	*0.86*	*0.8767*
		Weighted	*0.91*	*0.91*	*0.91*	

۷- نتیجه گیری

در این قسمت نتیجه گیری های کلی از این تمرین را ذکر می کنیم. چون خواستگاه مقاله ذکر شده در صورت تمرین ، با خواستگاه تمرین ۲ این درس متفاوت و کار آن مقاله بررسی تاثیر فیچر های خاصی بر روی نتیجه دسته بندی است و ما به طور کلی میخواهیم این عمل را انجام دهیم و روش های متنوع را تست کنیم، بنابراین مقایسه نتایج مقاله و تمرین شماره دو fair یا عادلانه نیست. به طور دقیق تر در مقاله ۵۰ فیچر خاص را انتخاب کرده است و عمل دسته بندی را انجام داده و از Leave one out cross validation استفاده کرده است. بنابراین از مقایسه صرف نظر کرده و فقط نتایج بدست آمده را ارائه میکنیم. دقت کنید در روش های مرتبط با شبکه های عصبی فقط شماره ی بهترین تست آورده شده است و همه ی تست های آن بخش را میتوانید در بخش مربوطه دنبال کنید.

Method	Test Number	Accuracy	Method	Precision	Recall	f-measure	K-Fold Cross Validation
Decision Tree	1	0.7763	Macro	0.69	0.73	0.71	0.7912
			Weighted	0.80	0.78	0.79	
Random Forest	1	0.8947	Macro	0.88	0.81	0.83	0.8691
			Weighted	0.89	0.89	0.89	
XGBoost	1	0.9211	Macro	0.92	0.84	0.88	0.8824
			Weighted	0.92	0.92	0.92	
SVM	1	0.8947	Macro	0.84	0.87	0.85	0.8471
			Weighted	0.90	0.89	0.90	
MLP	3	0.9474	Macro	0.92	0.92	0.92	0.8485
			Weighted	0.95	0.95	0.95	
ELM	5	0.9737	Macro	0.96	0.96	0.96	0.9103
			Weighted	0.97	0.97	0.97	
Auto Encoder	1	0.9737	Macro	0.98	0.94	0.96	0.9162
			Weighted	0.97	0.97	0.97	

پایان.