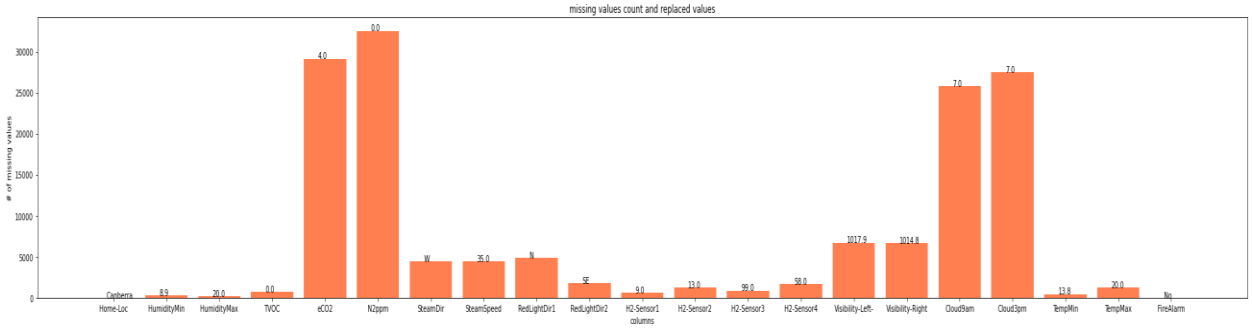
گزارش پیاده‌سازی و پاسخ به سوالات

سوال ۱:

الف) راه‌کارهای مختلفی برای پر کردن مقادیر حذف شده وجود دارد. در ستون‌های عددی می‌توان از میانگین یا مد یا میانه مقادیر آن ستون استفاده کرد و مقدار به‌دست آمده را در محل مقادیر حذف شده قرار داد. در ستون‌های دارای مقادیر غیر عددی می‌توان از مقدار مد آن ستون‌ها استفاده کرد. یکی دیگر از راه‌هایی که می‌توان مقادیر حذف شده را پر کرد آن است که نزدیک‌ترین سطری که به سطر فعلی وجود دارد را بیابیم و سپس مقادیر آن سطر را برای مقادیر حذف شده استفاده کنیم که روشی هوشمندانه‌تر است ولی پیچیدگی بیشتری دارد.

در این تمرین ابتدا سطرهایی که مقدار FireAlarm (که برچسب داده‌ها است) را ندارند حذف شده‌اند و سپس مقادیر حذف شده سایر ستون‌ها با مقدار مد برای آن ستون جایگزین شده اند.

نمودار زیر تعداد مقادیر حذف شده در هر ستون از داده‌ها و مقدار مد در آن ستون که به عنوان جایگزین استفاده شده است را نشان می‌دهد که این مقدار در بالای هر ستون از نمودار قرار گرفته است:



ب) نرمال سازی در شبکه‌های عصبی جلورو از اهمیت زیادی برخوردار است. دلیل اصلی این امر آن است که استفاده از داده‌های نرمال نشده باعث می‌شود که برخی از ویژگی‌ها دارای مقیاس بزرگتری باشند و در خروجی مدل تاثیر بیشتری بگذارند در حالی که ممکن است اهمیت آن ویژگی زیاد نباشد. همچنین عدم نرمال‌سازی داده‌ها باعث می‌شود که خروجی لایه‌های شبکه جلورو در مناطق اشباع توابع فعال‌سازی باشد و گرادیان محاسبه شده از آن کم باشد در حالی که نرمال‌سازی می‌تواند از این کار جلوگیری کند. به طور کلی هم نرمال‌سازی داده‌ها می‌تواند سرعت آموزش و همگرایی مدل را افزایش دهد.

در این تمرین نیز جهت نرمال سازی داده‌ها دو روش استاندارد و min-max پیاده‌سازی شده است.

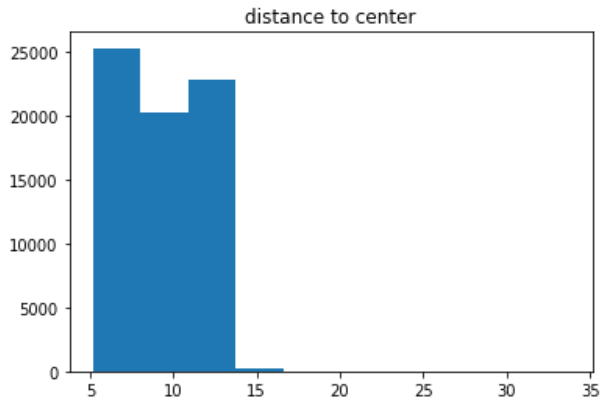
ج) در تبدیل ویژگی‌های غیر عددی به ویژگی‌های عددی روش‌هایی وجود دارد که دو مورد از آن‌ها انتساب اعدادی به مقادیر مختلف آن ویژگی عددی و همچنین one hot encoding است. در این تمرین پس از بررسی مشخص

شد که ویژگی مکانی موجود در داده‌ها دارای ۴۵ مقدار مختلف است که با توجه به تنوع حالت زیاد این ویژگی به هر یک از مقادیر این ویژگی عددی اختصاص یافته است. همچنین چند ویژگی دیگر از داده‌ها، جهت‌های جغرافیایی هستند که برای این ویژگی‌ها از one hot encoding استفاده شده است.

د) در داده‌های آموزشی، داده‌هایی که از عموم داده‌ها فاصله زیادی دارند داده پرت نامیده می‌شوند که باید قبل از آموزش مدل این داده‌ها تشخیص داده شده و حذف شوند.

روش‌های مختلفی برای حذف داده‌های پرت وجود دارد. یکی از این روش‌ها آن است که توزیع نرمال متناسب با داده‌ها یافته شود و داده‌هایی که در فاصله‌ای بیشتر از ۳ برابر انحراف معیار هستند به عنوان داده پرت شناخته شده و حذف می‌شوند. روش دیگر آن است که n نزدیک‌ترین داده به هر یک از داده‌ها را به دست آوریم و سپس بر اساس فاصله آن نزدیک‌ترین داده‌ها به هر داده امتیازی برای آن داده در نظر بگیریم که هر چه امتیاز آن داده بیشتر بود یعنی نزدیک‌ترین داده‌ها به آن داده فاصله بیشتری از آن داشته‌اند. سپس با رسم نمودار مربوط به این امتیازها می‌توان داده‌هایی که مقدار امتیاز آن‌ها خیلی بیشتر از سایر داده‌ها است را تشخیص داده که همان داده‌های پرت هستند.

برای تشخیص داده‌های پرت با استفاده از k-means به این صورت عمل می‌کنیم که ابتدا با توجه به آن که برچسب به صورت صفر و یکی است خوشه‌بندی به دو خوشه را انجام می‌دهیم و سپس فاصله هر داده تا مرکز خوشه مربوط به خود را محاسبه می‌کنیم و نمودار هیستوگرام این فاصله‌ها را نمایش می‌دهیم که نتیجه آن به صورت زیر است:



همان‌طور که دیده می‌شود فاصله داده‌ها تا مرکز خوشه مربوط به خودشان بیشتر در بازه ۵ تا ۱۵ است ولی تعداد معدودی از داده‌ها دارای فاصله‌ای بیشتر از ۱۵ هستند که همان داده‌های پرت می‌باشند و باید آن‌ها را حذف کرد. که در اثر این عمل تعداد ۱۲۱ داده حذف می‌شود.

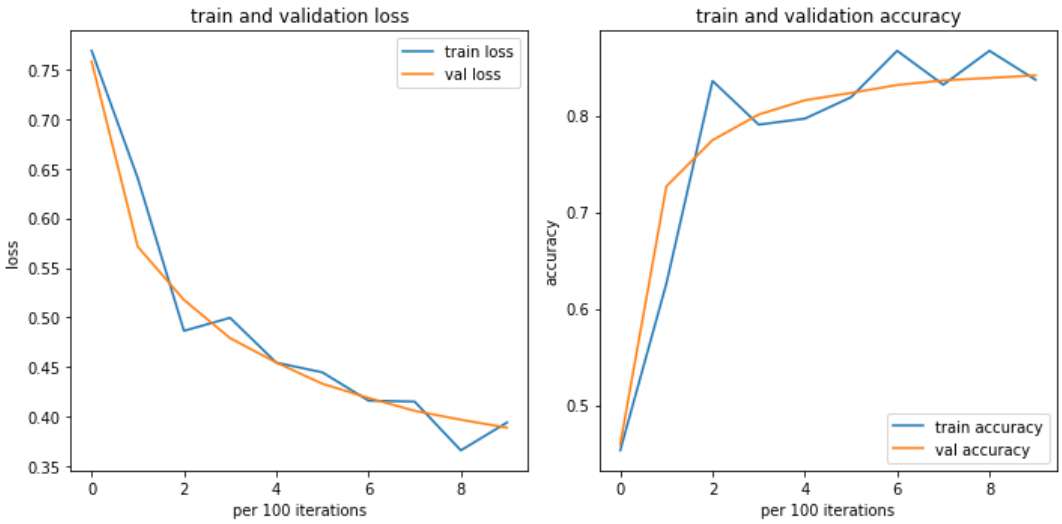
ه) داده‌ها به صورت تصادفی به سه قسمت تقسیم شده‌اند که ۷۰ درصد از داده‌ها مجموعه آموزشی و ۱۵ درصد مجموعه ارزیابی و ۱۵ درصد مجموعه آزمون می‌باشند.

سوال ۲:

جهت تشخیص جداپذیر خطی بودن داده‌ها با استفاده از شبکه عصبی جلورو به این صورت عمل می‌شود که یک شبکه جلورو تک لایه و بدون تابع فعال‌سازی استفاده می‌کنیم و اگر داده‌ها خطی جداپذیر باشند باید دقت حاصل از این شبکه تقریبا برابر ۱۰۰ باشد.

معماری پیاده‌سازی شده به این صورت است که تنها یک لایه وجود دارد که ورودی را به یک نورون خروجی متصل می‌کند و هیچ تابع فعال‌سازی استفاده نشده است. این شبکه قادر خواهد بود یک جدا کننده خطی در فضای داده‌ها بسازد و اگر داده‌ها خطی جداپذیر باشند آن‌ها را از هم جدا کند.

نمودار تغییرات loss و accuracy پس از ۵ ایپاک به صورت زیر است که در آن از batch size برابر ۱۲۸ استفاده شده و هر ۲۰۰ قدم مقدار loss و accuracy ذخیره شده است:



همان‌طور دیده می‌شود مقدار accuracy به دست آمده برابر ۰.۸۴۲ است که نشان دهنده آن است که داده‌های موجود به صورت خطی جداپذیر نیستند.

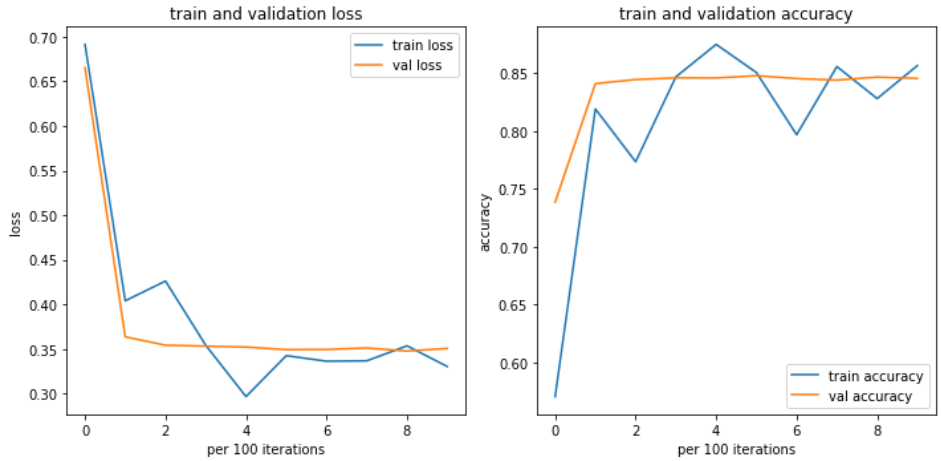
سوال ۳:

برای رسیدن به تعداد لایه‌ها و نورون‌های بهینه در شبکه جلورو حالت‌های مختلف را بررسی کرده‌ایم. حالت‌های مختلف در نظر گرفته شده و برای هر یک مدل را تا ۳ ایپاک آموزش داده‌ایم که نتایج حاصل از آن‌ها بر روی داده آموزشی و ارزیابی به صورت زیر است که در آن بهترین مقادیر با رنگ متفاوت نشان داده شده‌اند:

نورون‌های هر لایه	خطا در آموزش	صحت در آموزش	خطا در ارزیابی	صحت در ارزیابی
[8, 1]	۰.۳۶۴	۰.۸۴۲	۰.۳۶۲	۰.۸۴۴
[8, 8, 1]	۰.۳۶۲	۰.۸۴۲	۰.۳۶	۰.۸۴۲
[8, 8, 8, 1]	۰.۳۷۱	۰.۸۴	۰.۳۶۹	۰.۸۴۳
[64, 1]	۰.۳۵۲	۰.۸۴۷	۰.۳۵۴	۰.۸۴۵
[64, 64, 1]	۰.۳۴۹	۰.۸۴۸	۰.۳۵۱	۰.۸۴۵
[64, 64, 64, 1]	۰.۳۴۹	۰.۸۴۹	۰.۳۵۳	**۰.۸۴۷**
[128, 1]	۰.۳۴۸	۰.۸۴۸	۰.۳۵۲	۰.۸۴۶
[128, 128, 1]	۰.۳۴۵	۰.۸۵۱	**۰.۳۴۹**	۰.۸۴۶
[128, 128, 128, 1]	**۰.۳۴۴**	**۰.۸۵۲**	۰.۳۵۱	۰.۸۴۶
[64, 128, 1]	۰.۳۴۸	۰.۸۴۹	۰.۳۵۲	۰.۸۴۴
[128, 64, 1]	۰.۳۴۶	۰.۸۵	۰.۳۵۳	۰.۸۴۶
[128, 64, 64, 1]	۰.۳۴۶	۰.۸۵	۰.۳۵	۰.۸۴۵
[64, 128, 64, 1]	۰.۳۴۸	۰.۸۴۹	۰.۳۵۲	**۰.۸۴۷**
[64, 64, 128, 1]	۰.۳۴۷	۰.۸۵	۰.۳۵۱	**۰.۸۴۷**
[128, 128, 64, 1]	۰.۳۴۵	۰.۸۵	۰.۳۵	۰.۸۴۶
[128, 64, 128, 1]	۰.۳۴۵	۰.۸۵	۰.۳۵۳	۰.۸۴۴
[64, 128, 128, 1]	۰.۳۴۵	۰.۸۵	۰.۳۵۱	۰.۸۴۶
[64, 8, 1]	۰.۳۵	۰.۸۴۹	۰.۳۵۱	**۰.۸۴۷**
[8, 64, 1]	۰.۳۶۲	۰.۸۴۳	۰.۳۵۸	۰.۸۴۴
[64, 8, 8, 1]	۰.۳۵۱	۰.۸۴۹	۰.۳۵۲	۰.۸۴۶
[8, 64, 8, 1]	۰.۳۶۱	۰.۸۴۴	۰.۳۶۱	۰.۸۴۳
[8, 8, 64, 1]	۰.۳۶۲	۰.۸۴۳	۰.۳۶۱	۰.۸۴۲

همان‌طور که دیده می‌شود کمترین خطای ارزیابی در شبکه‌ای با تعداد نورون‌های ۱۲۸و ۱۲۸ و ۱ است که مقدار صحت ارزیابی و همچنین خطا و صحت آموزشی هم برای این مقدار تقریبا برابر بهترین مقادیر است که در نتیجه

این معماری به عنوان بهترین معماری در نظر گرفته می‌شود. پس از آموزش این معماری تا ۵ ایپاک نمودار تغییرات خطا و صحت در آموزش و ارزیابی به صورت زیر است:

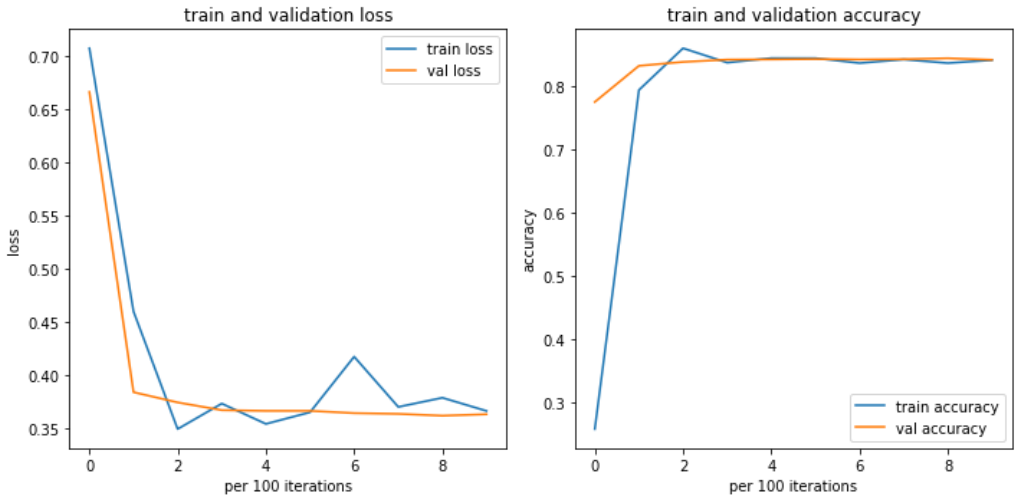


سوال ۴:

به طور کلی بدون استفاده از توابع فعال‌سازی، ترکیب چند لایه خطی باز هم خطی است و برای تولید یک جداکننده غیرخطی به این توابع نیاز داریم. نوع توابع فعال‌سازی نیز از اهمیت زیادی برخوردار است چرا که استفاده از توابع مختلف باعث می‌شود نوع ورودی لایه بعدی به کلی تغییر کند و این مسئله در نحوه آموزش مدل تغییرات جدی ایجاد می‌کند.

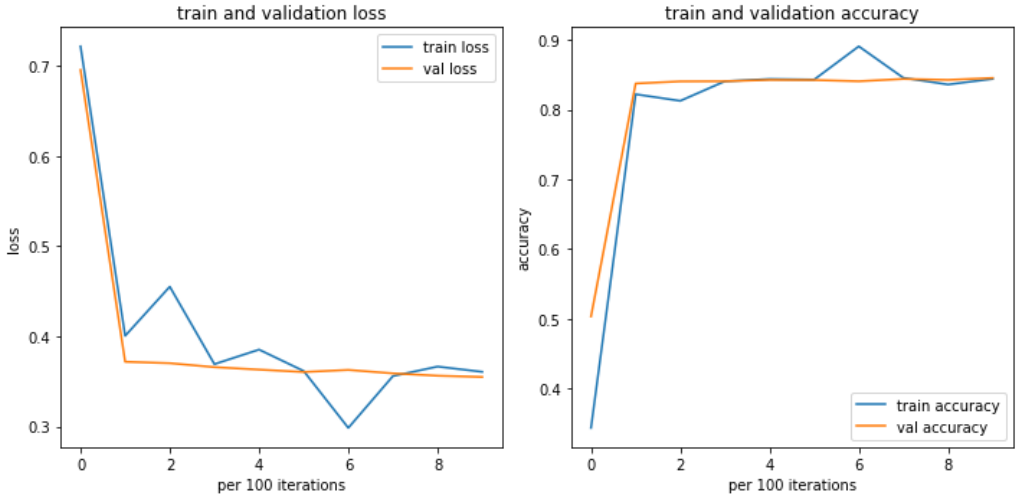
در مقایسه توابع مختلف چهار تابع فعال‌سازی سیگموید، تانژانت هایپربولیک، رلو و لیکی‌رلو در نظر گرفته‌شده‌اند و تعداد لایه‌ها و نورون‌های شبکه جلورو برای همه یکسان در نظر گرفته شده است. نمودار تغییرات loss و accuracy به صورت زیر است:

تابع فعال‌سازی	تغییرات loss و accuracy



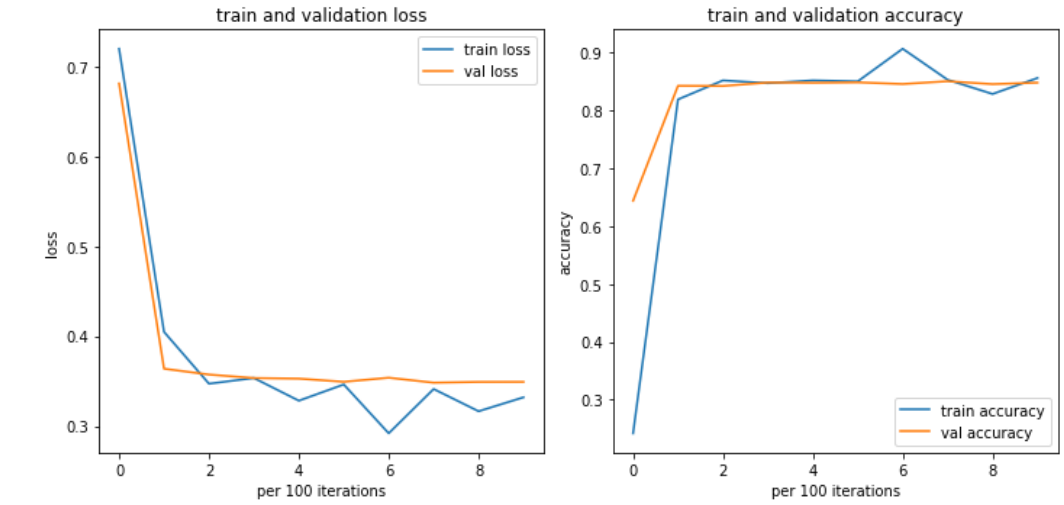
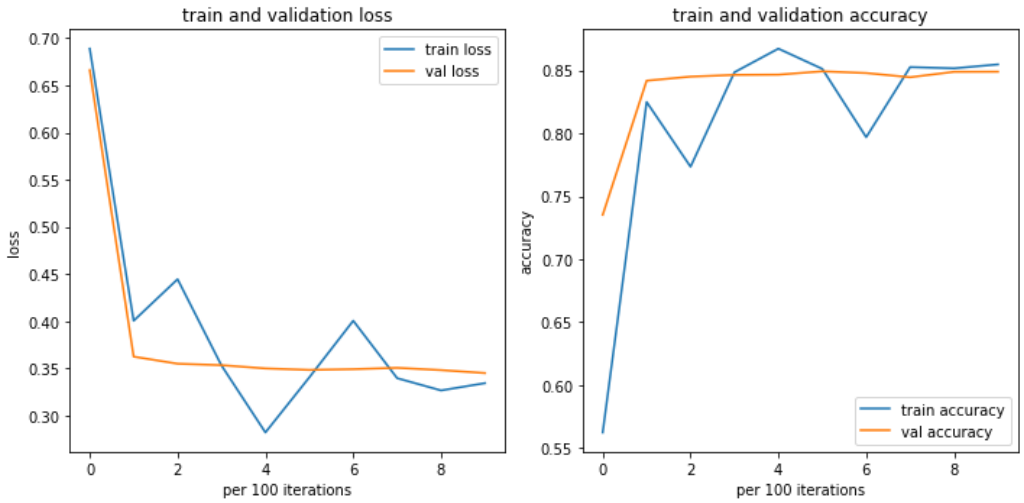
Sigmoid

```
epoch 3:train loss=0.367, train acc=0.842, val loss=0.364, val acc=0.842
```



Tanh

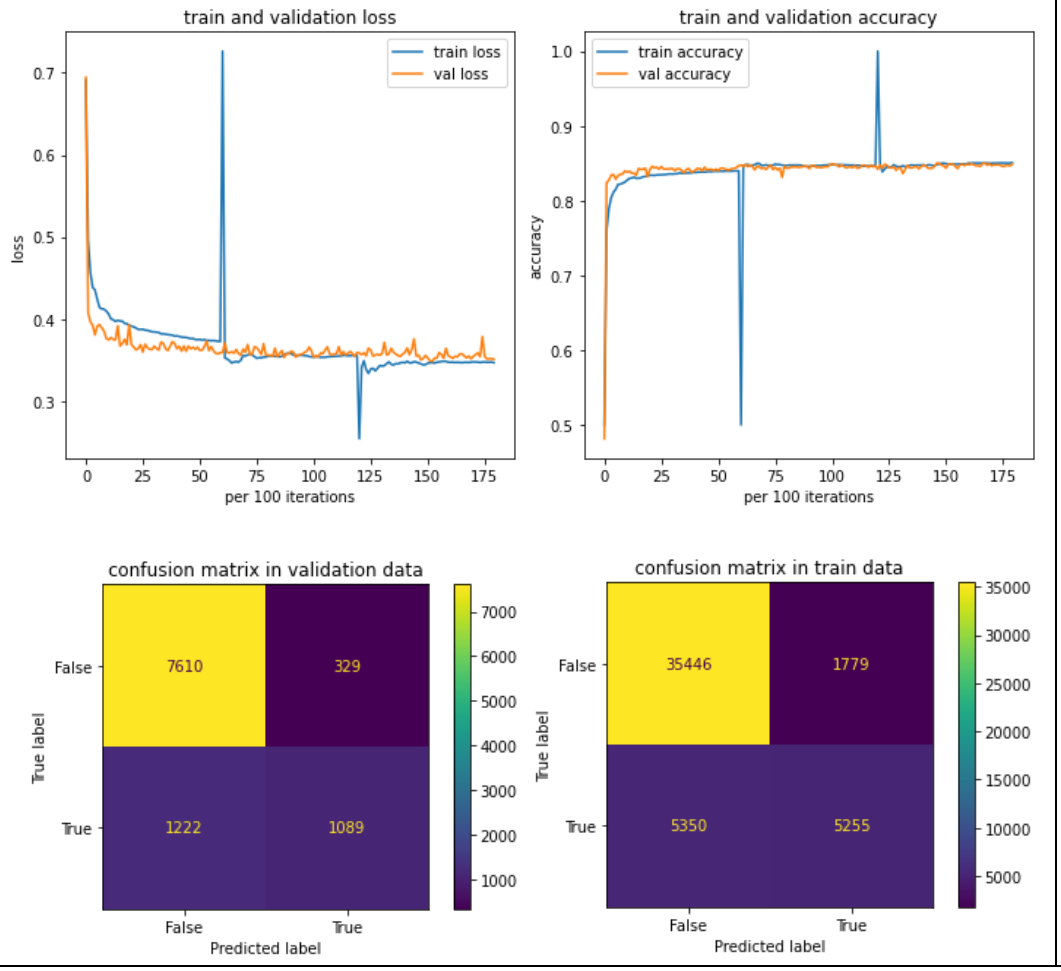
```
epoch 4: train loss=0.354, train acc=0.847, val loss=0.355, val acc=0.845
```

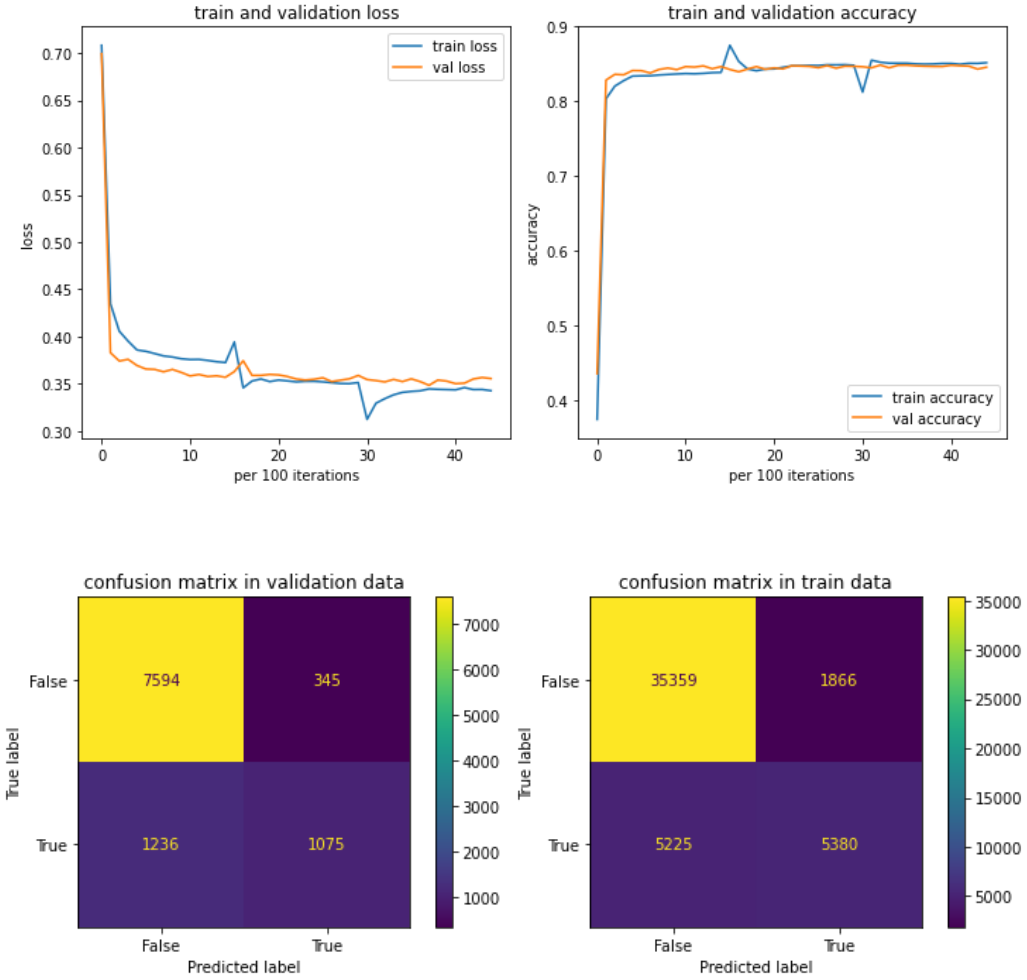
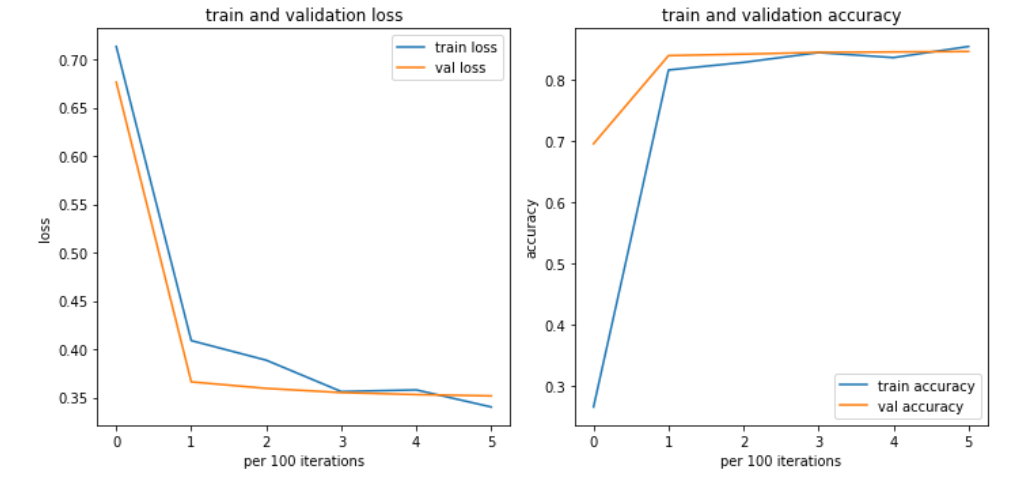
 epoch 3:train loss=0.34, train acc=0.853, val loss=0.348, val acc=0.85	ReLU
 epoch 4:train loss=0.333, train acc=0.856, val loss=0.345, val acc=0.849	Leaky ReLU

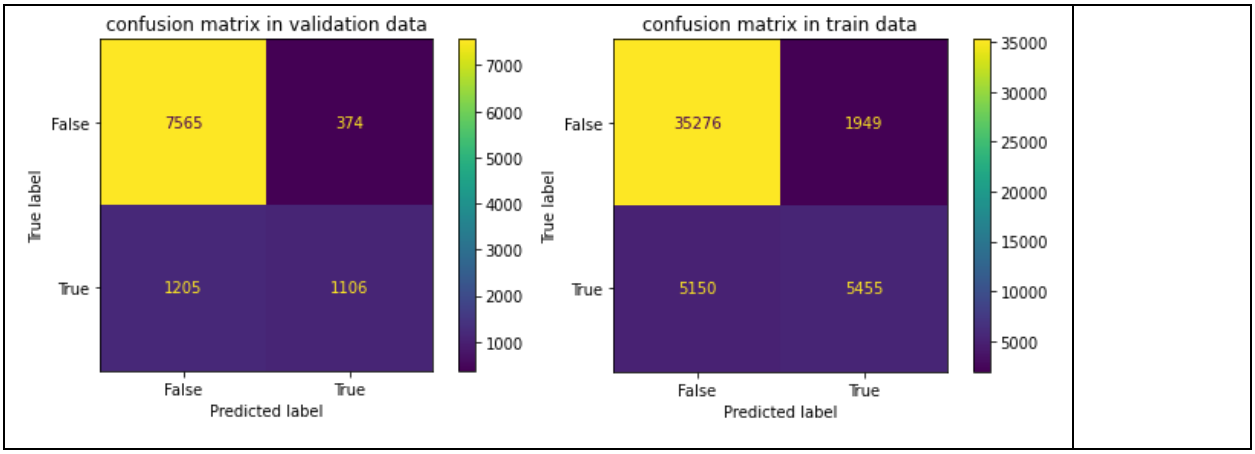
همان‌طور که دیده می‌شود مقدار خطا و صحت در اثر استفاده از توابع فعال‌سازی متفاوت مقادیر مختلفی دارند و همچنین تغییرات نمودارهای مربوط به آن‌ها نیز متفاوت است. همان‌طور که از نتایج قابل برداشت است استفاده از تابع ReLU توانسته به نتایج بهتری برسد.

سوال ۵:

برای بررسی مقادیر مختلف برای batch size معماری به دست آمده از قسمت‌های قبلی را ثابت می‌گیریم و مقدار batch size را تغییر می‌دهیم و هر بار نمودارهای خطا و صحت و همچنین ماتریس درهم‌ریختگی را نشان می‌دهیم. نتیجه این فرایند به صورت زیر است:

Batch sized	خروجی
۴	

 train and validation loss train loss val loss loss per 100 iterations train and validation accuracy accuracy train accuracy val accuracy per 100 iterations confusion matrix in validation data False True True label 7594 345 1236 1075 False True Predicted label confusion matrix in train data False True True label 35359 1866 5225 5380 False True Predicted label	۱۶
 train and validation loss train loss val loss loss per 100 iterations train and validation accuracy accuracy train accuracy val accuracy per 100 iterations	۱۲۸



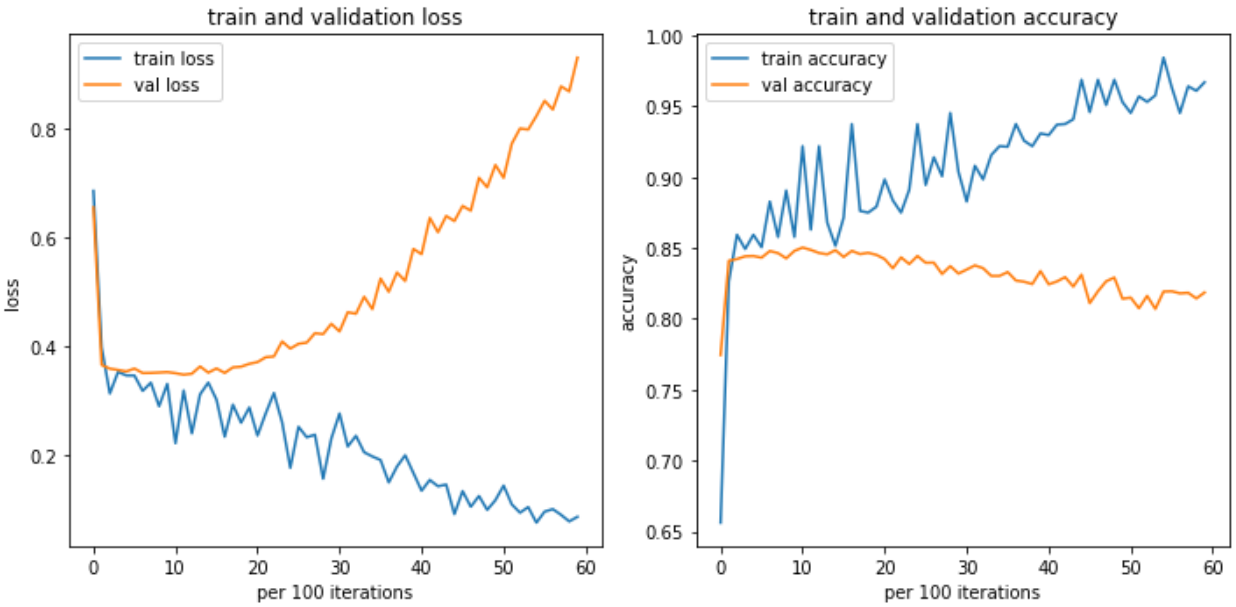
همان‌طور که در نتایج به دست آمده دیده می‌شود در اثر استفاده از batch size کوچک‌تر تغییرات loss بیشتر است و نمودار آن نویزی‌تر حرکت داشته است و در نتیجه دیرتر همگرا شده است. علاوه بر این با توجه به ماتریس‌های درهم‌ریختگی به دست آمده در هر حالت دیده می‌شود که تعداد مقادیر در هر قسمت از ماتریس متفاوت است که نشان دهنده آن است که استفاده از batch size های مختلف توانسته ما را به صحت‌های مختلفی برساند. دلیل این امر آن است که تغییرات وزن‌های مدل به ازای تعداد متفاوتی از داده‌ها انجام می‌شود.

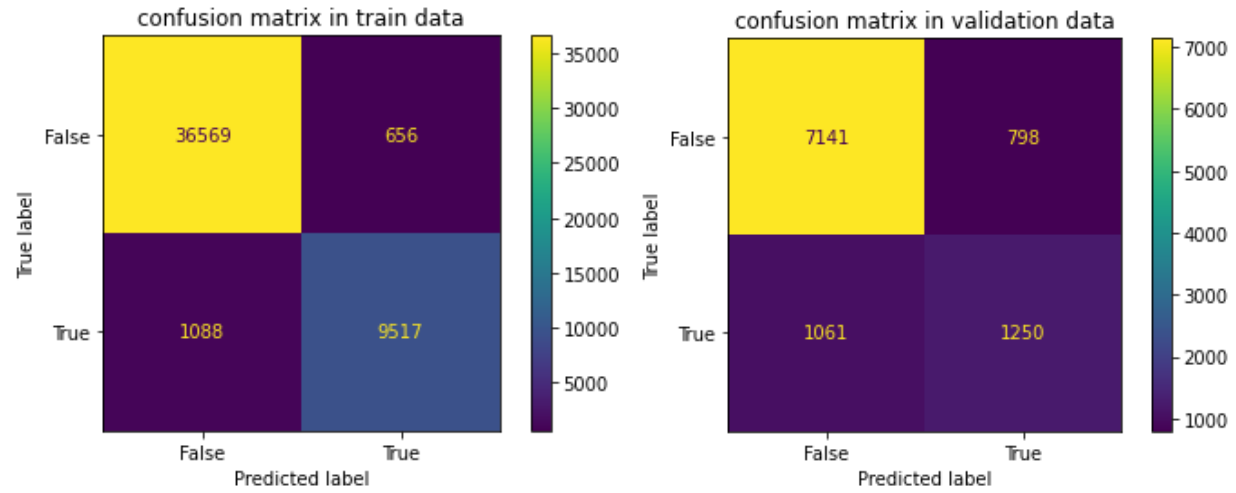
سوال ۶:

در اثر بیش‌برازش در شبکه جلورو وزن‌ها به گونه‌ای آموزش می‌بینند که بتوانند داده‌های آموزشی را به خوبی از هم جدا کنند ولی بر روی داده‌های جدید دقت خوبی نخواهند داشت. این اتفاق وقتی می‌افتد که صحت مدل بر روی داده‌های آموزشی بسیار بالا باشد ولی بر روی داده‌های ارزیابی صحت کمی را به دست می‌دهد. برای جلوگیری از بیش‌برازش در مدل باید از تعداد لایه‌ها و همچنین تعداد نورون‌های مناسبی استفاده کرد چرا که استفاده از تعداد بالای لایه با نورون‌های زیاد باعث می‌شود مدل مرزهایی را بسازد که تنها برای جداسازی داده‌های آموزشی مناسب هستند. همچنین آموزش مدل با تعداد ایپاک بالا هم می‌تواند باعث بیش‌برازش مدل بر روی داده آموزشی شود که باید تعداد ایپاک را به گونه‌ای تعیین کرد که خطا ارزیابی کمترین مقدار خود باشد که برای این کار می‌توان از early stopping استفاده نمود.

برای ایجاد بیش‌برازش در آموزش مدل از معماری با لایه‌های ۱۲۸، ۲۵۶، ۱۲۸، ۱ استفاده شده است که دارای تعداد زیادی پارامتر می‌باشد. همچنین تعداد گام‌های آموزش ۳۰ قرار داده شده است که بسیار بیشتر از حد مورد

نیاز است. نمودار تغییرات خطا و صحت در آموزش و ارزیابی و همچنین ماتریس‌های درهم‌ریختیگی آن‌ها به صورت زیر است:





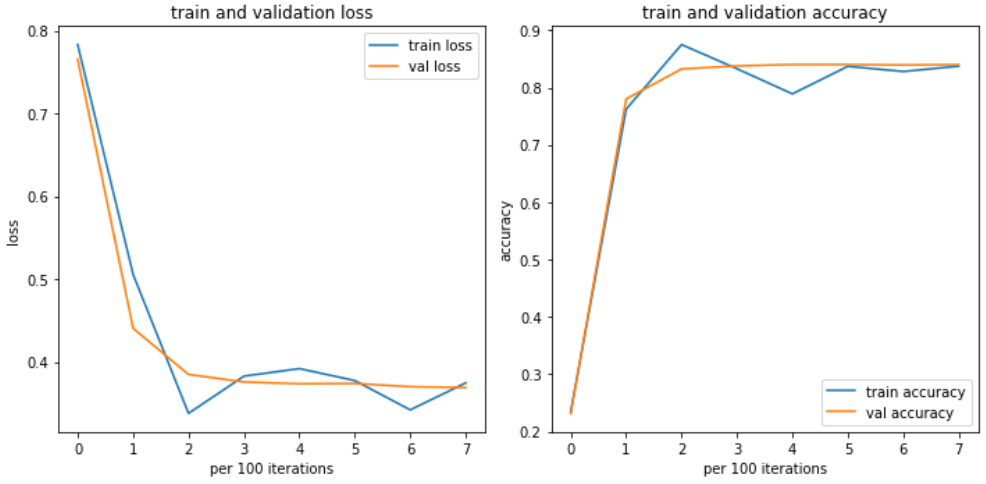
همان‌طور که در نمودار خطا و صحت دیده می‌شود ابتدا خطا در آموزش و ارزیابی کاهشی بوده است ولی پس از گذشت مدتی با تکرار بیش از حد آموزش روی داده‌های آموزشی، مدل دچار بیش‌برازش می‌شود و مقدار خطا داده‌های آموزشی بسیار کاهش یافته و در داده‌های ارزیابی افزایش یافته است. همچنین در ماتریس درهم‌ریختگی به دست آمده دیده می‌شود که در داده‌های آموزشی تعداد بسیار بالایی از داده‌ها به درستی دسته‌بندی شده‌اند و مقدار صحت و دقت مدل بالا بوده ولی در داده‌های ارزیابی درصد زیادی از داده‌ها به اشتباه دسته‌بندی شده‌اند و صحت و دقت مدل بسیار کم است.

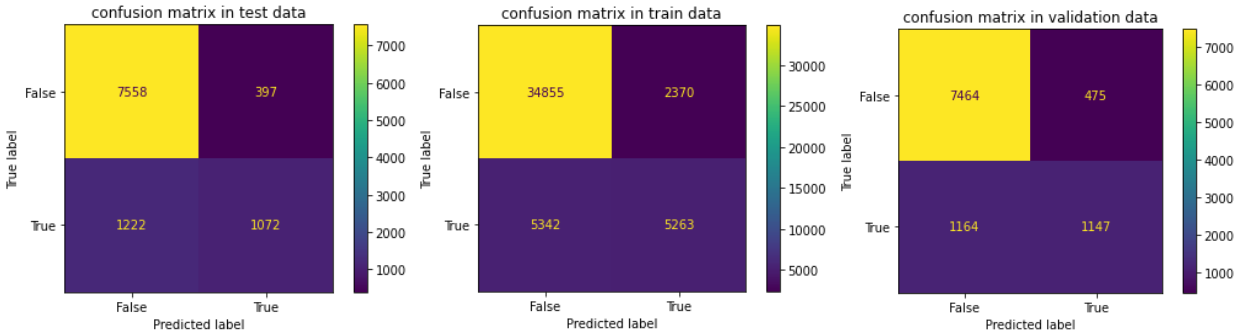
سوال ۷:

تعمیم پذیری در شبکه جلورو به معنای آن است که مدل آموزش دیده بتواند بر روی داده‌های جدیدی که تا به حال ندیده است هم دقت خوبی داشته باشد. اهمیت تعمیم پذیری در شبکه‌های جلورو در آن است که در اثر ورود داده جدید مطمئن باشیم مدل توانایی آن را دارد که با دقت مناسبی آن را دسته‌بندی کند.

در پیاده‌سازی شبکه تعمیم‌پذیر باید از استفاده از تعداد لایه‌های زیاد که دارای تعداد نورون‌های زیادی هستند اجتناب کرد. همچنین می‌توان با استفاده از لایه‌های drop out به تعمیم‌پذیری مدل کمک کرد چون این لایه‌ها باعث می‌شوند برخی از وزن‌ها حذف شوند و لایه‌های بعدی تنها با داشتن بقیه وزن‌ها پیش‌بینی لازم را انجام دهند که این عمل به توانایی مدل در برخورد با داده‌های جدید را افزایش می‌دهد. همچنین برای بهبود تعمیم‌پذیری مدل می‌توان از داده‌های آموزشی متنوع‌تری استفاده کرد تا مدل حالت‌های مختلف داده‌ها را دیده باشد و از از داده‌های تکراری و غیربالانس جلوگیری کرد.

در پیاده‌سازی انجام شده از لایه‌های ۶۴، ۶۴، ۱ استفاده شده است و بین هر دو لایه یک لایه drop out با مقدار احتمال ۰.۱ قرار داده شده است. پس از تنها ۴ ایپاک آموزش نمودار خطا و صحت و همچنین ماتریس درهم‌ریختگی برای آموزش و ارزیابی و آزمون به صورت زیر است:





```
test loss: 0.379
test accuracy: 0.842
```

همان‌طور که در نمودارهای تغییرات خطا و صحت دیده می‌شود مقدار صحت مدل در زمان آموزش کمتر از این مقدار در زمان ارزیابی و آزمون است که دلیل این امر تعمیم‌پذیری مدل است. همچنین با مشاهده ماتریس‌های درهم‌ریختگی مشاهده می‌شود که مدل بر روی داده‌های آزمون توانسته با صحت و دقت خوبی داده‌ها را دسته‌بندی کند.