به نام خدا

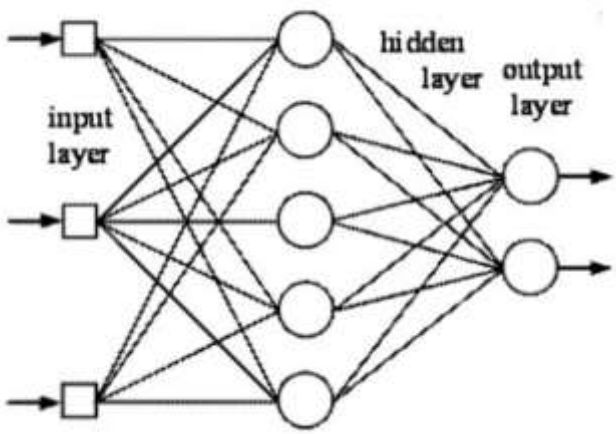
گزارش تمرین سری دوم درس شبکه عصبی

زهرا دهقانیان

۹۸۱۳۱۰۵۹

سوال اول)

شبکه عصبی پرسپترونی چند لایه، یک مدل شبکه عصبی است که قادر دسته بندی و رگرسیون روی داده های غیرخطی را انجام دهد. این شبکه توسط رزنبلت معرفی اولیه شد و قانون اموزش ان توسط وربز ارایه اولیه و توسط راملهارت بکار گرفته شد. معماری این شبکه به صورت است که از یک لایه ورودی ، چندین لایه نهان و یک لایه خروجی تشکیل شده است. تعداد نورون لایه ورودی بستگی به بعد و یا تعداد ویژگی های داده ها دارد و تعداد نورون خروجی بستگی به تعداد کلاس ها و یا صورت مورد انتظار خروجی دارد. اما تعداد نورون ها در لایه های نهان به عوامل زیادی بستگی دارد و با ازمون و خطا باید بدست بیاید. در تمام شبکه همه ی نورون های لایه قبل به نورون یک لایه متصل هستند. یک نمای کلی از این شبکه به صورت زیر است:



قانون اموزش در این شبکه قانون پس انتشار خطاست. توضیح کلی این قانون به صورت است که یک مرتبه از سمت نود های ورودی به سمت نودهای خروجی ، حاصل خروجی هر کدام از نورون ها را تولید میکنیم. سپس برای نودهای لایه اخر میزان خطا را محاسبه میکنیم و این خطا را با توجه به وزن های هر نود در یک لایه به لایه قبلی و در جهت ورودی انتشار میدهیم. پس از این دو مرحله ، با توجه به میزان خطا و مقدار خروجی هر نورون ، وزن مربتط با ان را اپدیت میکنیم و این روند را ادامه میدهیم تا خطای شبکه از حد خاصی کمتر شود و یا تغییری دیگر در شبکه حاصل نشود.

سوال دوم)

در این بخش به کمک تابع csv_reader داده هارا خواندیم و سپس به کمک تابع train_test_split داده هارا تفسیم کردیم. سپس به پیش پردازش داده هاپرداختیم. در این خصوص در بخش بعد توضیحات بیشتر داده شده است.

سوال سوم)

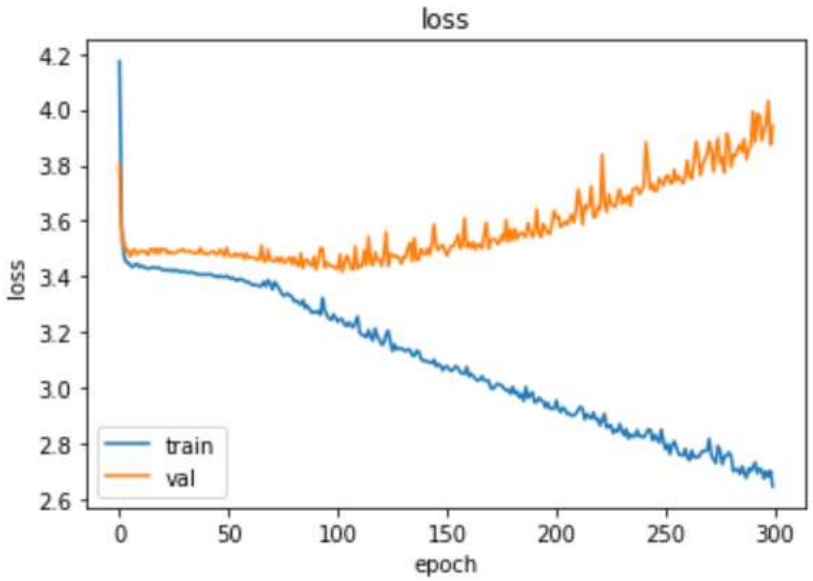
تفاوت در رگرسیون و دسته بندی در دو بخش است . تفاوت اول در بازنمایی برای کلاس هاست ؛ زمانی که در حالت رگرسیون هستیم. فاصله بین کلاس ها معنی دارد و با توجه به صحبت های استاد بهتر هم است که scale خروجی و ورودی ها یکسان باشد. پس بهترین گزینه این است که تمام داده ها را به کمک MinMaxScaler در بازه ۰ تا ۱ ببریم. اما در حالت کلاس بندی، بهتر است به کمک پیش پردازش labelEncoder که برای داده های دسته بندی تعریف شده است، استفاده کنیم

تفاوت دوم در loss function این دو شبکه است. در حالت دسته بندی ، همان طور که در کلاس حل تمرین هم بررسی شد، cross entropy تابع مناسب است اما در حالت رگرسیون میانگین مربع خطاها یا همان mse تابع مناسب می باشد.

در این مساله با توجه به این که ساختار برچسب ها از نوع پیوسته است، به نظر می رسد که این دیتاست بیشتر برای تسک های رگرسیون مناسب باشد تا تسک دسته بندی.

سوال چهارم)

حال به بررسی مقادیر بهینه برای شبکه می پردازیم. در ابتدا به کمک ۱۰۰۰ داده به یافتن مقادیر بهینه شبکه خواهیم پرداخت. ابتدا ساختار شبکه دسته بند را مورد بررسی قرار میدهیم. ابتدا برای یافتن تعداد ایپک مناسب نمودار دقت برای داده اموزش و ولیدیشن را رسم میکنیم که به صورت زیر است:



با توجه به این نمودار مقدار بهینه ایپک در حدود ۷۰ می باشد. اما زمانی که دقت را برای ۷۰ ایپک و ۲۰۰ و ۳۰۰و ۴۰۰ و ۵۰۰ ایپک در شبکه های مختلف بررسی میکنیم میبینیم که تقریبا در همه شبکه ها ۲۰۰ ایپک دقت بیشتری دارد. پس با ۲۰۰ ایپک ازمایش های زیر را انجام میدهیم.

برای بررسی تعداد لایه، سه شبکه با معماری های زیر را مورد بررسی قرار میدهیم. خروجی این سه شبکه به صورت زیر می باشد:

شبکه با معماری ۵ لایه به صورت ۸۹-۱۲۵-۲۵۰-۵۰۰-۹۰

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_61 (Dense)             (None, 500)               45500
```

```
_________________________________________________________________
dense_62 (Dense)             (None, 250)               125250
_________________________________________________________________
dense_63 (Dense)             (None, 125)               31375
_________________________________________________________________
dense_64 (Dense)             (None, 89)                11214
=================================================================
Total params: 213,339
Trainable params: 213,339
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 4.0891 - accuracy: 0.0700
```

شبکه بعدی که مورد بررسی قرار دادیم چهار لایه با معماری ۸۹-۱۲۵-۲۵۰-۹۰ بود:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_65 (Dense)             (None, 250)               22750
_________________________________________________________________
dense_66 (Dense)             (None, 125)               31375
_________________________________________________________________
dense_67 (Dense)             (None, 89)                11214
=================================================================
Total params: 65,339
Trainable params: 65,339
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 3.5013 - accuracy: 0.0800
```

شبکه بعدی معماری سه لایه به صورت ۸۹-۱۲۵-۹۰ داشت و خروجی آن به صورت زیر بوده است:

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_68 (Dense)             (None, 125)               11375
_________________________________________________________________
dense_69 (Dense)             (None, 89)                11214
=================================================================
Total params: 22,589
Trainable params: 22,589
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 3.5099 - accuracy: 0.0400
```

همانطور که می بینید بهترین دقت و کمترین loss مربوط به معماری ۴ لایه بود.

پس از این بخش به بررسی مقدار قدم مناسب برای اموزش می پردازیم. سه مقدار ۰,۱ و ۰,۰۱ و ۱ را بر روی شبکه بهینه مورد بررسی قرار میدهیم:

```
# learning rate = 0.1
4/4 - 0s - loss: 3.5106 - accuracy: 0.0600
# learning rate = 0.01
4/4 - 0s - loss: 3.4697 - accuracy: 0.0500
# learning rate = 0.001
4/4 - 0s - loss: 3.5908 - accuracy: 0.0700
```

در اینجا بهترین مقدار = ۰,۰۰۱ می باشد که همان مقدار default برای این optimizer است.

در ادامه به بررسی تعداد نورون بهینه برای هر لایه میپردازیم. میدانیم که تعداد نورون خروجی و ورودی قابل تغییر نیس و تنها برای لایه های نهان مقادیر متفاوت زیر را امتحان میکنیم.

مدل اول که بررسی کردیم معماری ۸۹-۱۵۰-۳۰۰-۹۰ بود:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_82 (Dense)             (None, 300)               27300
_________________________________________________________________
dense_83 (Dense)             (None, 150)               45150
_________________________________________________________________
dense_84 (Dense)             (None, 89)                13439
=================================================================
Total params: 85,889
Trainable params: 85,889
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 3.6666 - accuracy: 0.0300
```

مدل بعدی مدل ۸۹-۲۵۰-۵۰۰-۹۰ بود :

```
Model: "sequential_29"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_85 (Dense)             (None, 500)               45500
_________________________________________________________________
dense_86 (Dense)             (None, 250)               125250
_________________________________________________________________
```

```
dense_87 (Dense)             (None, 89)                22339
=================================================================
Total params: 193,089
Trainable params: 193,089
Non-trainable params: 0

4/4 - 0s - loss: 3.7679 - accuracy: 0.0500
```

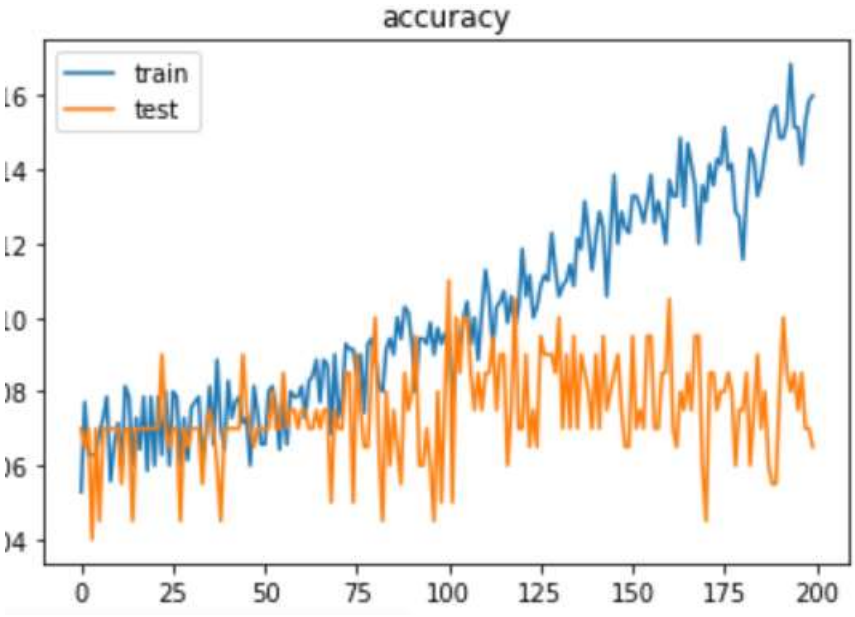
و مدل اخر مدل ۸۹-۱۰۰-۲۰۰-۹۰ بود که خروجی ان بدین ترتیب است:

```
Model: "sequential_30"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_88 (Dense)             (None, 200)               18200
_________________________________________________________________
dense_89 (Dense)             (None, 100)               20100

dense_90 (Dense)             (None, 89)                8989
=================================================================
Total params: 47,289
Trainable params: 47,289
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 3.4708 - accuracy: 0.0500
```

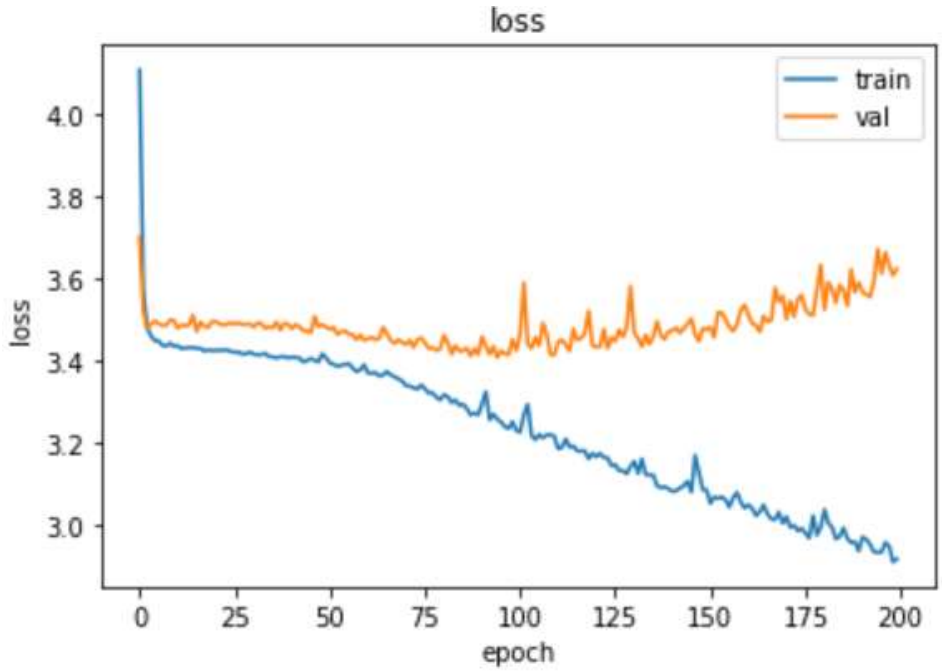
به نظر می رسد با این اوصاف بهترین دسته بندی که یافتیم ، یک دسته بند با دو لایه نهان با معماری ۸۹-۱۲۵-۲۵۰-۹۰ با نرخ یادگیری ۰,۰۰۱ بوده است که خروجی آن بدین ترتیب است :

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_65 (Dense)             (None, 250)               22750
_________________________________________________________________
dense_66 (Dense)             (None, 125)               31375

dense_67 (Dense)             (None, 89)                11214
=================================================================
Total params: 65,339
Trainable params: 65,339
Non-trainable params: 0

_________________________________________________________________
4/4 - 0s - loss: 3.5013 - accuracy: 0.0800
```

در نهایت نمودار دقت و تابع loss به صورت زیر است :



و نمودار تابع ضرر به صورت زیر است:



حال به بررسی مقادیر بهینه برای شبکه رگرسیون می پردازیم.

ابتدا به بررسی تعداد لایه های مناسب می پردازیم. ابتدا به بررسی معماری ۵ لایه می پردازیم :

```
Model: "sequential_48"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_143 (Dense)            (None, 500)               45500
_________________________________________________________________
dense_144 (Dense)            (None, 250)               125250
_________________________________________________________________
dense_145 (Dense)            (None, 125)               31375
_________________________________________________________________
dense_146 (Dense)            (None, 1)                 126
=================================================================
Total params: 202,251
Trainable params: 202,251
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 6.6993 - mean_squared_error: 83.3077
```

معماری ۴ لایه عملکردی به صورت زیر دارد:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_147 (Dense)            (None, 250)               22750
_________________________________________________________________
dense_148 (Dense)            (None, 125)               31375
_________________________________________________________________
dense_149 (Dense)            (None, 1)                 126
=================================================================
Total params: 54,251
Trainable params: 54,251
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 6.7653 - mean_squared_error: 86.2693
```

و خروجی معماری ۳ لایه به صورت زیر خواهد بود:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_150 (Dense)            (None, 125)               11375
_________________________________________________________________
dense_151 (Dense)            (None, 1)                 126
```

```
=================================================================
Total params: 11,501
Trainable params: 11,501
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 7.1282 - mean_squared_error: 103.3775
```

به نظر می رسد در اینجا معماری ۵ لایه عملکرد بهتری دارد.

حال به بررسی نرخ آموزش مناسب می پردازیم. سه نرخ ۰,۱ و ۰,۰۱ و ۰,۰۰۱ را بررسی میکنیم:

```
# learning rate = 0.1
4/4 - 0s - loss: 7.0448 - mean_squared_error: 102.8743
# learning rate = 0.01
4/4 - 0s - loss: 6.9329 - mean_squared_error: 92.9388
# learning rate = 0.001
4/4 - 0s - loss: 6.7801 - mean_squared_error: 88.3100
# learning rate = 0.0001
4/4 - 0s - loss: 6.9521 - mean_squared_error: 89.9774
```

به نظر می رسد در اینجا نیز نرخ ۰/۰۰۱ نرخ مناسبی می باشد.

حال به بررسی تعداد نورون مناسب برای لایه های نهان با معماری ۵ لایه می پردازیم.

معماری اول مورد بررسی معماری ۹۰-۱۲۵-۸۰-۴۰-۱ می باشد:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_143 (Dense)            (None, 500)               45500
_________________________________________________________________
dense_144 (Dense)            (None, 250)               125250
_________________________________________________________________
dense_145 (Dense)            (None, 125)               31375
_________________________________________________________________
dense_146 (Dense)            (None, 1)                 126
=================================================================
Total params: 202,251
Trainable params: 202,251
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 6.6993 - mean_squared_error: 83.3077
```

معماری بعدی مورد بررسی ۹۰-۲۰۰-۱۰۰-۵۰-۱ می باشد :

```
Model: "sequential_56"
_________________________________________________________________
```

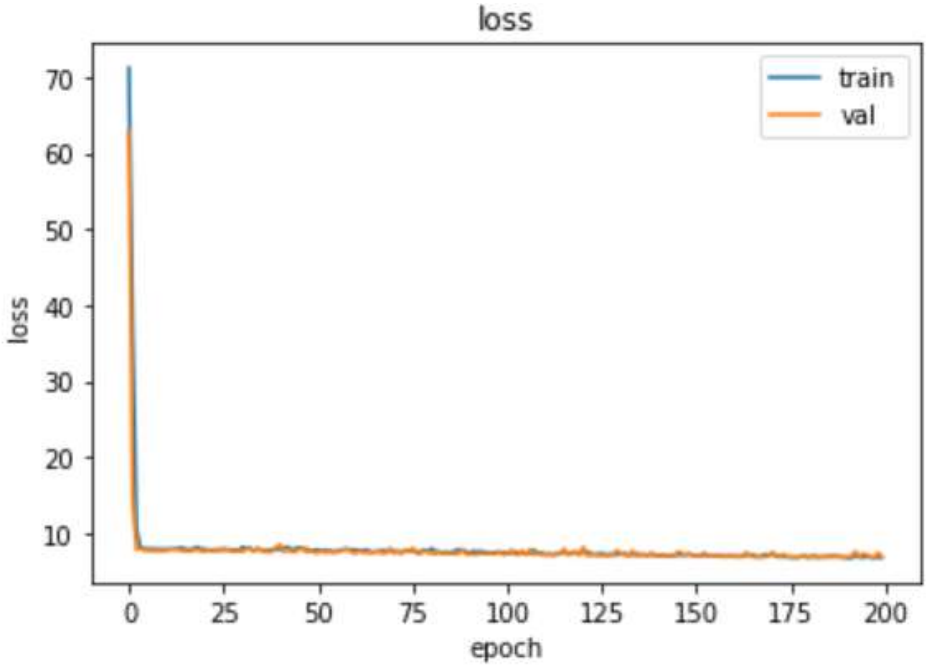
```
Layer (type)                 Output Shape              Param #
=================================================================
dense_171 (Dense)            (None, 200)               18200
_________________________________________________________________
dense_172 (Dense)            (None, 100)               20100
_________________________________________________________________
dense_173 (Dense)            (None, 50)                5050
_________________________________________________________________
dense_174 (Dense)            (None, 1)                 51
=================================================================
Total params: 43,401
Trainable params: 43,401
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 6.7064 - mean_squared_error: 81.8725
```

و معماری بعدی مورد بررسی ۹۰-۶۰-۳۰-۱۵-۱ است:

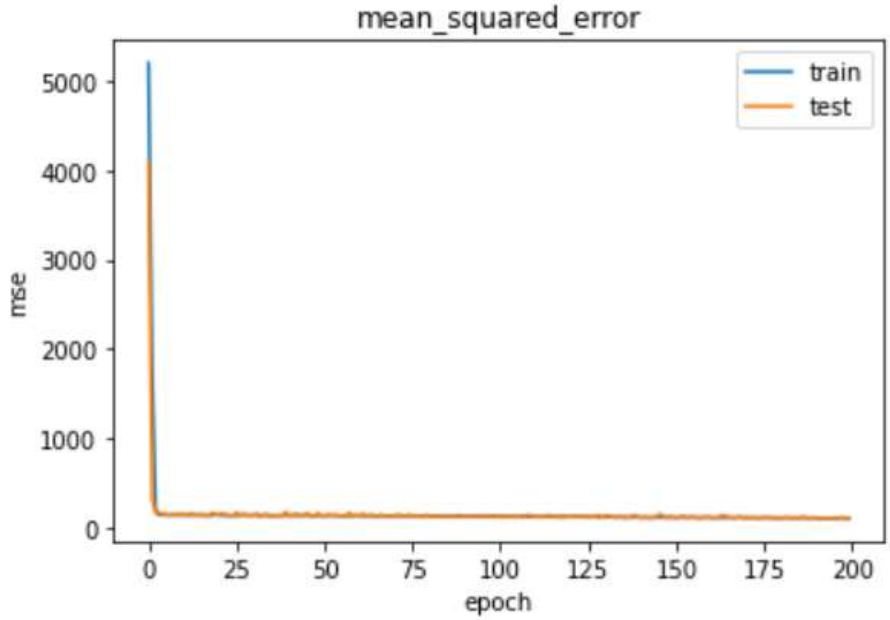
```
Model: "sequential_57"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_175 (Dense)            (None, 60)                5460
_________________________________________________________________
dense_176 (Dense)            (None, 30)                1830
_________________________________________________________________
dense_177 (Dense)            (None, 15)                465
_________________________________________________________________
dense_178 (Dense)            (None, 1)                 16
=================================================================
Total params: 7,771
Trainable params: 7,771
Non-trainable params: 0
_________________________________________________________________
4/4 - 0s - loss: 6.9577 - mean_squared_error: 91.6627
```

همانطور که مشخص است، بهترین شبکه ای که برای رگرسیون یافتیم شبکه ۵ لایه با نرخ یادگیری ۰,۰۰۱ و معماری ۹۰-۲۰۰-۱۰۰-۵۰-۱ بوده است. نمودار تابع ضرر و mse برای این شبکه به صورت زیر است :

تابع ضرر :



خطای میانگین مجموع مربعات :



در نهایت باید عملکرد کلی این دو شبکه بهینه یافته شده را با هم مقایسه کنیم و برای این کار ابتدا شبکه هارا بر روی تمامی داده ها اموزش میدهیم و سپس از شبکه رگرسیون نیز معیار دقت و از شبکه دسته بندی معیار mse را میخواهیم تا این دو روش را با هم مقایسه کنیم.

خروجی شبکه دسته بند :

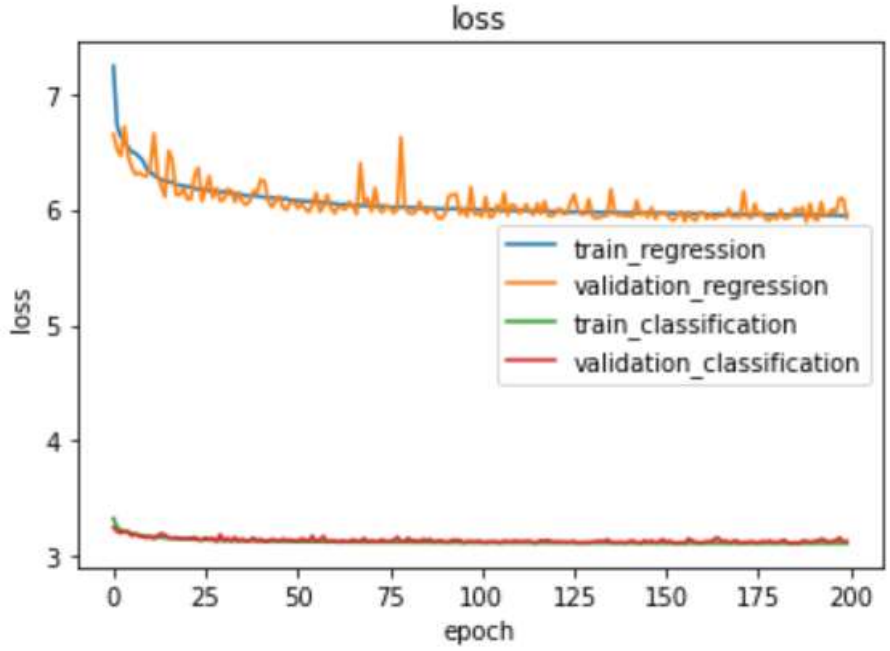
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 250)               22750
_________________________________________________________________
dense_4 (Dense)              (None, 125)               31375
_________________________________________________________________
dense_5 (Dense)              (None, 89)                11214
=================================================================
Total params: 65,339
Trainable params: 65,339
Non-trainable params: 0

1611/1611 - 1s - loss: 3.1350 - mean_squared_error: 5803.6841 - accuracy:
0.0909
```

خروجی شبکه رگرسیون:

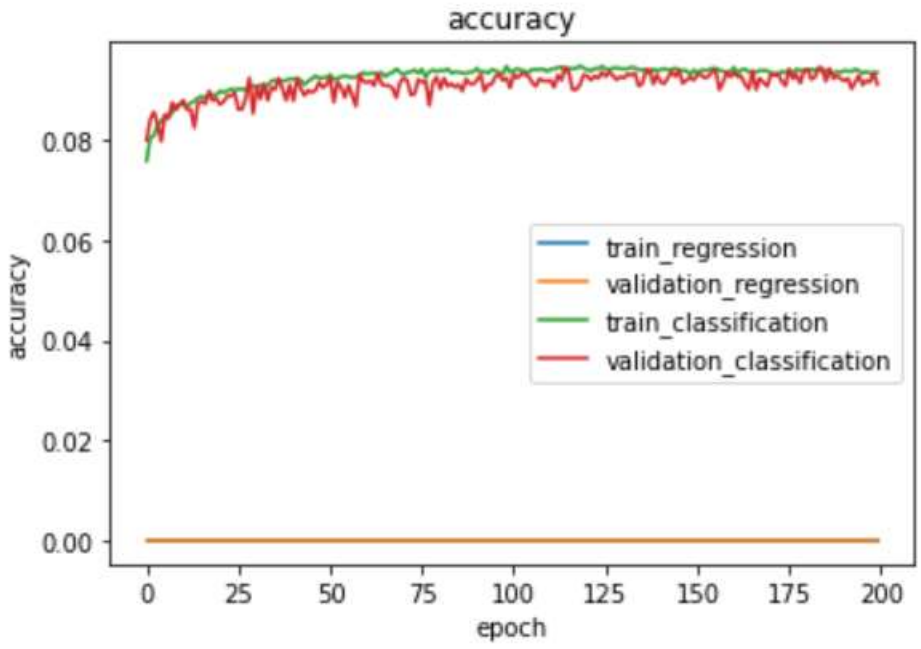
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_6 (Dense)              (None, 200)               18200
_________________________________________________________________
dense_7 (Dense)              (None, 100)               20100
_________________________________________________________________
dense_8 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_9 (Dense)              (None, 1)                 51
=================================================================
Total params: 43,401
Trainable params: 43,401
Non-trainable params: 0
_________________________________________________________________
1611/1611 - 1s - loss: 5.9139 - mean_squared_error: 86.9185 - accuracy:
0.0000e+00
```

نمودار خطای mse به صورت زیر است:

نمودار تابع ضرر به صورت زیر است :



نمودار دقت به صورت زیر است:



به نظر می رسد برخلاف انتظار عملکرد کلی برای شبکه دسته بندی بهتر بوده است.

***نکته ای که در پایان به نظرم قابل توجه است این است که دقت کلی هیچ کدام از این مدل ها به ۱۰٪ نرسیده و این به هیچ وجه نتیجه قابل قبولی نیست و به نظر هیچ کدام از این شبکه ها روی این داده ها درست عمل نکرده اند. علت این موضوع میتواند ضعف قدرت پردازشی ما و یا این باشد که در این جا ما ۹۰ ویژگی میگیریم و میخواهیم به کمک آن نوع کلاس این داده از میان ۹۰ کلاس را مشخص کنیم که به نظر این تعداد ویژگی متناسب با تعداد کلاس ها نمی باشد. همین طور ممکن است که به طور کلی این نوع شبکه برای این مجموعه داده مناسب نبوده باشد و یا این مجموعه داده نویز زیاد داشته که سبب این نتیجه شده است.