به نام خدا

دانشگاه صنعتی امیرکبیر

دانشکده مهندسی کامپیوتر

پاسخ تمرین سری دوم شبکه‌های عصبی

استاد:

دکتر صفابخش

دانشجو:

حلیمه رحیمی

شماره دانشجویی:

۹۹۱۳۱۰۴۳

بهار ۱۴۰۰

۱- یک پرسپترون چند لایه تشکیل شده است از چندین نورون که در لایه‌های مختلف جای گرفته‌اند و نورون‌های هر لایه به لایه‌ی پایینی و بالایی خود متصل است و بنابراین یک شبکه‌ی تماما متصل را ایجاد می‌کنند. لایه‌ها را می‌توان به سه گروه ورودی، پنهان و خروجی تقسیم کرد. نورون‌ها در لایه‌ی اول که به عنوان لایه‌ی ورودی محسوب می‌شود، تنها ورودی‌ها را می‌گیرند و خروجی هر لایه، ورودی لایه‌ی بعد خواهد بود.

به طور کلی آموزش پرسپترون چند لایه در دو فاز انجام می‌گیرد:

- فاز پیش‌رو: در این فاز وزن‌ها تغییری نمی‌کنند و ورودی‌ها پس از محاسبات انجام گرفته بر اساس وزن‌ها و توابع فعالیت لایه به لایه گذر کرده و منتهی به پیشبینی خروجی می‌شوند.
- فاز پس‌رو: در این فاز خروجی‌ها با مقدار مورد انتظار مقایسه شده و خطای حاصل لایه به لایه و در جهت مخالف گذر کرده و موجب بروزرسانی وزن‌ها می‌شوند.

هر نورون پنهان دو محاسبات انجام می‌دهد:

- محاسبه‌ی خروجی با استفاده از تشکیل تابعی غیرخطی برای نگاشت ورودی‌ها.

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) y_i(n)$$

$$y_j(n) = \varphi_j(v_j(n))$$

نورون کنونی با j و نورون پیشین با i شناخته می‌شود. n نشاندهنده‌ی نورون است و φ_j تابع فعالیت نورون کنونی.

- محاسبه برای اصلاح وزن‌ها در جهت بردار گرادیان.

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)}$$

که در آن η درجه یادگیری می‌باشد.

با قاعده‌ی زنجیره‌ای داریم:

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial v_j(n)} y_i(n)$$

که در آن

$$-\frac{\partial \mathcal{E}(n)}{\partial v_j(n)} = \varphi_j'\left(v_j(n)\right) \sum_k -\frac{\partial \mathcal{E}(n)}{\partial v_k(n)} w_{jk}(n)$$

خواهد بود.

می‌توان عملکرد نورون‌های لایه‌های پنهان را اینگونه بیان کرد که ورودی‌ها را با تبدیلی غیرخطی به یک فضای ویژگی می برند. ممکن است در این فضا بتوان داده‌ها را راحت‌تر از قبل از یکدیگر جدا کرد. همچنین از نگاهی دیگر، می‌توان اینطور بیان کرد که در هر لایه نورون‌ها خطی را به عنوان جاکننده تعیین می‌کنند. با گذر از هر لایه، امکان آن وجود دارد که به طور مثال، فضایی مانند یک مثلث تشکیل شود که دو دسته را به خوبی از یکدیگر جدا کند.

۲- در ابتدا داده‌ها را شافل و نتیجه را با پسوند csv ذخیره کردم تا مقایسه نتایج هنگام tune کردن پارامترها همواره براساس داده‌های یکسان باشد. تمامی نتایج براساس همین شافل است. داده‌ها را به ترتیب ۷۰، ۲۰ و ۱۰ به مجموعه آموزش، اعتبارسنجی و آزمون تقسیم کردم و ستون اول را به عنوان برچسب داده‌ها جدا کردم.

۳- در رابطه با رگرسیون: با توجه به آنکه ویژگی‌ها در مقیاس‌های متفاوتی هستند (به طور مثال، همانطور که در تصویر می‌بینید ویژگی ۱ در بازه‌ی بین ۱ تا ۶۲ بوده درحالیکه ویژگی ۲ بین ۳۳۷- تا ۳۸۴ می‌باشد) نیاز است که داده‌ها را نرمال یا استاندارد کنیم.

	0	1	2	3
count	515345.000000	515345.000000	515345.000000	515345.000000
mean	1998.397082	43.387126	1.289554	8.658347
std	10.931046	6.067558	51.580351	35.268585
min	1922.000000	1.749000	-337.092500	-301.005060
25%	1994.000000	39.954690	-26.059520	-11.462710
50%	2002.000000	44.258500	8.417850	10.476320
75%	2006.000000	47.833890	36.124010	29.764820
max	2011.000000	61.970140	384.065730	322.851430

8 rows × 91 columns

لازم به ذکر است که علاوه بر ویژگی‌ها، مقادیر هدف را نیز تغییر مقیاس داده‌ام (نرمال کرده و بین صفر و یک برده‌ام). این مسئله لزومی ندارد اما وجودش می‌تواند کمک- کننده باشد. با کوچک شدن مقادیر ویژگی‌ها در صورتی که مقادیر هدف بسیار بزرگ باشند، مقدار وزن‌ها افزایش پیدا می‌کند طوری که ممکن است مقدار بی‌نهایت بگیرد و مشکلات محاسباتی ایجاد کند. البته در اینجا چنین اتفاقی نمی‌افتاد ولی بهتر دیدم دلایل خود را ذکر کنم. علاوه بر این با توجه به اینکه در اینجا مقادیر هدف حداقل یک عدد با یکدیگر فاصله دارند، احتمال می‌دهم با بردن مقادیر هدف بین صفر و یک بتوان این فاصله را نسبتا بیشتر کرد.

در رابطه با تابع هزینه، لازم می‌بینم از میانگین مربعات خطا استفاده کنم. با توجه به اینکه مقادیر هدف در حداقل حالت یک عدد با هم تفاوت دارند، و معمول‌تر است سال تولید موسیقی را در حد یک سال خطا داشته باشیم تا چند سال، می‌توان اهمیت بیشتری به خطاهای بزرگتر داد و بنابراین از MSE استفاده کرد. البته این احتمال وجود دارد که برای کوچک کردن MSE وزن‌ها به گونه‌ای بروز شوند که چندین خطای کوچک یک ساله داشته باشیم. چنین رخدادی گاه در حال آموزش رخ می‌دهد و می‌توان در نتایج آن را مشاهده کرد. به طور کلی هر دو MSE و MAE تقریبا به یک شکل پیش می‌روند و کمتر رخ می‌دهد که کم شدن یکی باعث بیشتر شدن دیگری شود.

تعداد نورون‌های ورودی شبکه ۹۰ تا خواهد بود و تنها یک نورون خروجی با تابع فعالیت همانی/ خطی خواهیم داشت.

برای دسته‌بندی لازم است مقادیر هدف را منهای کمترین سال (یعنی ۱۹۲۲) کنیم و سپس به شکل one hot درآوریم. تعداد نورون‌های ورودی شبکه همچنان برابر با تعداد ویژگی‌ها و ۹۰ خواهد بود اما تعداد نورون‌های خروجی به ۹۰ تا (۱+۱۹۲۲-۲۰۱۱) تغییر خواهد کرد که تابع فعالیت softmax برای آنها انتخاب می‌شود. البته در میان داده‌ها هیچ یک مربوط به سال نمی‌باشد با این حال برای حفظ صورت مسئله و ترتیب صحیح اعداد، ۹۰ برچسب را تعیین کردم.

برای تابع هزینه از یکی از توابع مناسب دسته‌بندی چند کلاسه و به طور معمول از categorical cross-entropy استفاده می‌کنیم (با توجه به اینکه برچسب‌ها نیز به شکل categorical می‌باشند).

می‌توان دو نوع نگاه به پاسخ مسئله داشت:

با توجه به اینکه داده‌ها مربوط به موسیقی و سال تولیدشان است و مقادیر هدف در حداقل حالت تنها یک عدد با یکدیگر فاصله دارند، از طرفی سبک‌های موسیقی معمولا سال به سال تغییر نمی‌کنند بلکه دهه به دهه (یا شاید هر پنج سال) این تغییرات مشهود است، می‌توان انتظار داشت که در حد زیر ده سال (و شاید زیر پنج سال) خطا داشت. از آن جهت که در دسته‌بندی حتی یک سال خطا موجب می‌شود داده به اشتباه در دسته‌ای دیگر قرار گیرد، خطا برای دسته‌بندی بالا خواهد بود. در حالیکه در رگرسیون خطای یک ساله قابل پذیرش است. با این نگاه، رگرسیون جواب بهتری دارد.

اما اگر مسئله را اینطور بدانیم که حتما باید سال را درست تشخیص داد، پس نتایج رگرسیون را به شکلی درمی‌آوریم که بتوان جدول درهم‌ریختگی داشت و دقت به دست آورد. در این صورت دسته‌بند عملکرد بهتری خواهد داشت؛ چرا که اصلا برای همین طراحی شده است. در حالیکه رگرسیون تنها سعی در کم کردن فاصله‌ی پیشبنی با مقدار هدف دارد و می‌تواند بپذیرد که بسیاری از سال‌ها با فاصله‌ی یک ساله پیشبینی شوند.

بنابراین به طور کلی اگر مهم باشد که سال را به درستی تشخیص دهیم، دسته‌بند نتیجه بهتری خواهد داشت.

۴- در ابتدا باید بیان کنم که شرط خاتمه‌ی آموزش را برای رگرسیون کمتر نشدن خطای MSE مجموعه اعتبارسنجی، و برای دسته‌بندی بیشتر نشدن دقت مجموعه اعتبارسنجی تا ۵ ایپاک متوالی قرار دادم. آنچه در جداول می‌بینید ۵ ایپاک کمتر از مقدار ایپاکی است که متوقف شده است. علاوه بر این Tune کردن پارامترها را روی کل داده‌ها انجام دادم. هر دو کد تحویل داده شده، مربوط به بهترین مدل هستند.

در همه‌ی آزمایشات از بهینه ساز Adam استفاده کردم. تابع فعالیت لایه‌های پنهان Relu می‌باشد.

جدول تغییر پارامترها برای دسته‌بندی را در زیر مشاهده می‌کنید. بهترین نتیجه برای دقت مجموعه آموزش و اعتبارسنجی با رنگ قرمز مشخص شده است.

Trial #	# of hidden layers	Neuron in each layer	Learning Rate	Epoch	Train Accuracy	Val Accuracy
1	1	1024	0.001	10	0.1326	0.1036
2	1	512	0.001	14	0.1278	0.1018
3	1	256	0.001	5	0.1057	0.1004
4	1	128	0.001	4	0.0989	0.0977
5	2	256,128	0.001	5	0.1034	0.1006

6	2	128,256	0.001	4	0.1059	0.1015
7	2	128,128	0.001	5	0.1045	0.1001
8	2	128,512	0.001	4	0.1063	0.1035
9	3	128,256,128	0.001	9	0.1095	0.1019
10	3	64,128,64	0.001	7	0.1026	0.1006
11	3	128,256,512	0.001	8	0.1081	0.1029
12	2	128,512	0.0005	14	0.1553	0.1074
13	2	128,512	0.0003	10	0.1476	0.1097
14	2	128,512	Adaptive, init=0.0005 After 5 epochs, decay: Init/(epoch-4)	13	0.1625	0.1112

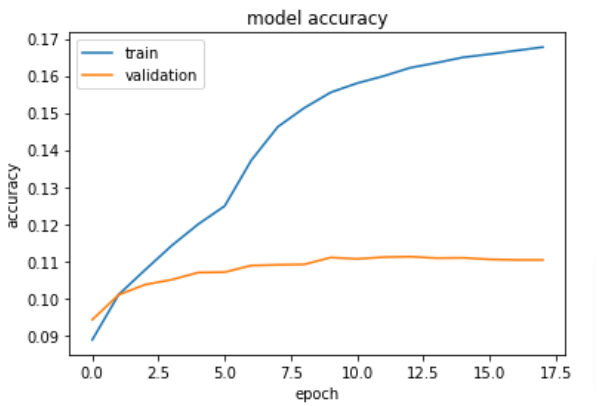
مشاهده می‌شود که در صورت استفاده از یک لایه‌ی مخفی، هرچه تعداد نورون‌ها بیشتر باشد، نتیجه بهتر خواهد بود. البته انتظار می‌رود با بیشتر شدن نورون‌ها کم کم به سمت بیش‌برازش پیش برویم. در صورت داشتن دو لایه، مشاهده می‌شود با بیشتر بودن تعداد نورون‌های لایه‌ی دوم از نورون‌های لایه‌ی اول، نتیجه‌ی بهتری حاصل می‌شود. علت این امر می‌تواند داشتن چندین دسته باشد که با لایه‌ی دوم نورون‌ها یا بیشتر کردن نورون‌ها در شبکه‌ی تک لایه می‌توان فضا را با چیزی بهتر از خطوط غیر متصل به یکدیگر جدا کرد (مثلا یک مثلث تشکیل شود). علاوه بر این تعداد پارامترهای بیشتر نتیجه بهتری را داشته اما باید به این مسئله توجه داشت که در صورتی که تعداد پارامترها بسیار بیشتر از تعداد داده‌ها باشد، احتمال کم برازش شدن شبکه وجود دارد.

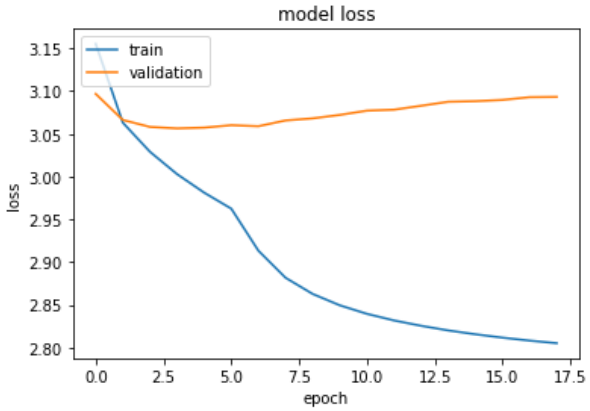
کاملا مشهود است که داشتن درجه یادگیری تطابقی نتیجه بهتری نسبت به درجه‌ی ثابت داشته است. همچنین علت آنکه در آزمایش ۱۳ مقدار درجه یادگیری را نسبت به قبل کم کردم به دلیل این بود که گمان کردم احتمال دارد درجه یادگیری بزرگتر از مقدار مناسب آن باشد و در حال پریدن از یکی از مینیموم‌های محلی است که اتفاقا احتمال دارد مینیموم کلی باشد.

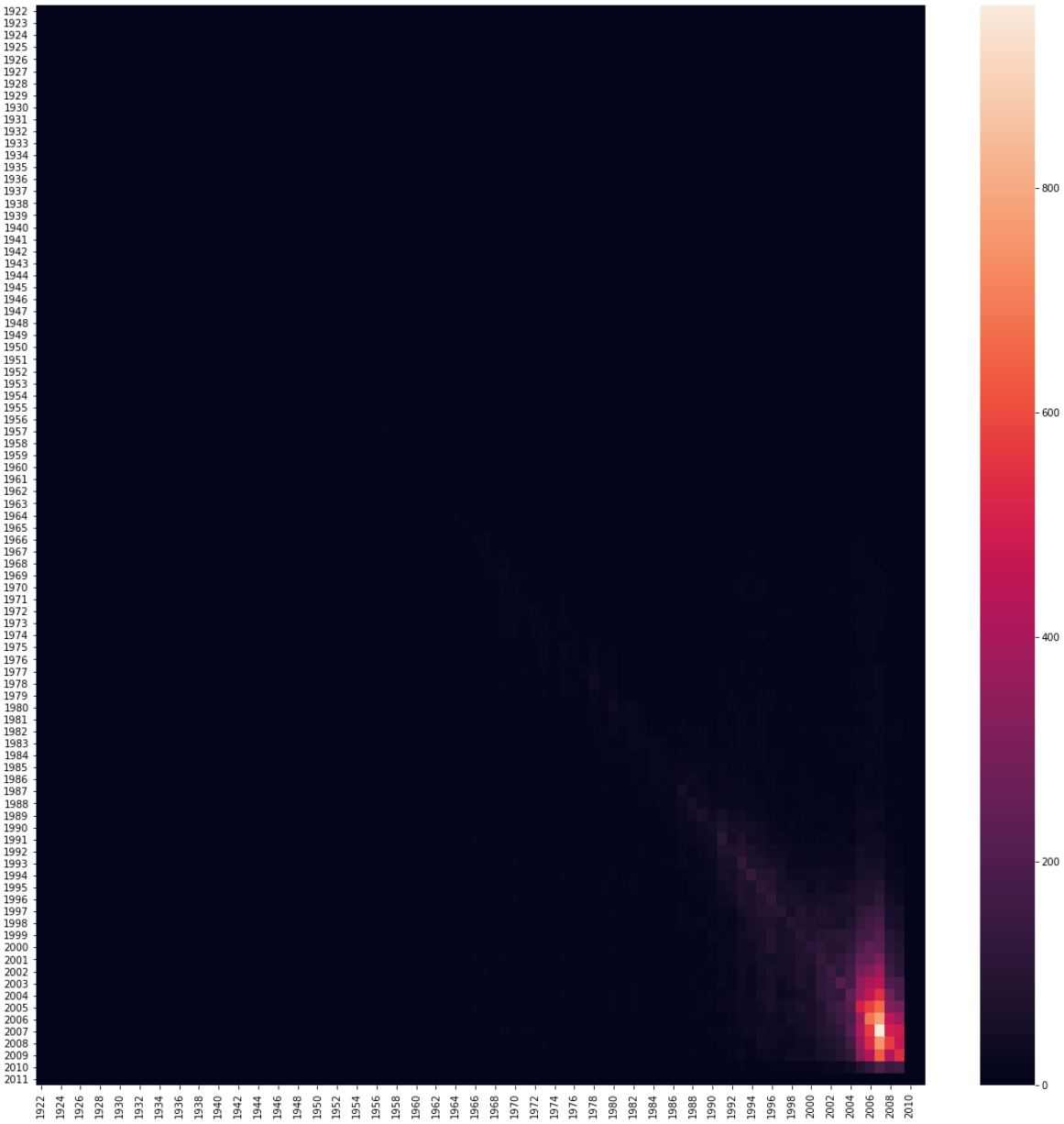
مدل دو لایه‌ی آزمایش ۸ تعداد پارامترهای کمتری نسبت به مدل سه لایه‌ی آزمایش ۱۱ دارد، درحالیکه نتیجه‌ی بهتری برای مجموعه اعتبارسنجی و نتیجه‌ی کمتری برای مجموعه آموزش دارد. احتمال دارد این مسئله به دلیل بیش برازش باشد.

بهترین مدل در آزمایش ۱۴ آمده است. نتایج این مدل را برای مجموعه آزمون مشاهده می‌کنید:

# of hidden layers	**Neuron in each layer**	**Learning Rate**	**Epoch**	**Train Accuracy**	**Val Accuracy**	**Test Accuracy**
2	128,512	Adaptive, init=0.0005 After 5 epochs, decay: Init/(epoch-4)	13	0.1625	0.1112	0.1116


model accuracy
train
validation
accuracy
epoch
0.17
0.16
0.15
0.14
0.13
0.12
0.11
0.10
0.09
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


model loss
train
validation
loss
epoch
3.15
3.10
3.05
3.00
2.95
2.90
2.85
2.80
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


1922
1924
1926
1928
1930
1932
1934
1936
1938
1940
1942
1944
1946
1948
1950
1952
1954
1956
1958
1960
1962
1964
1966
1968
1970
1972
1974
1976
1978
1980
1982
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
2006
2008
2010
2011
800
600
400
200
0

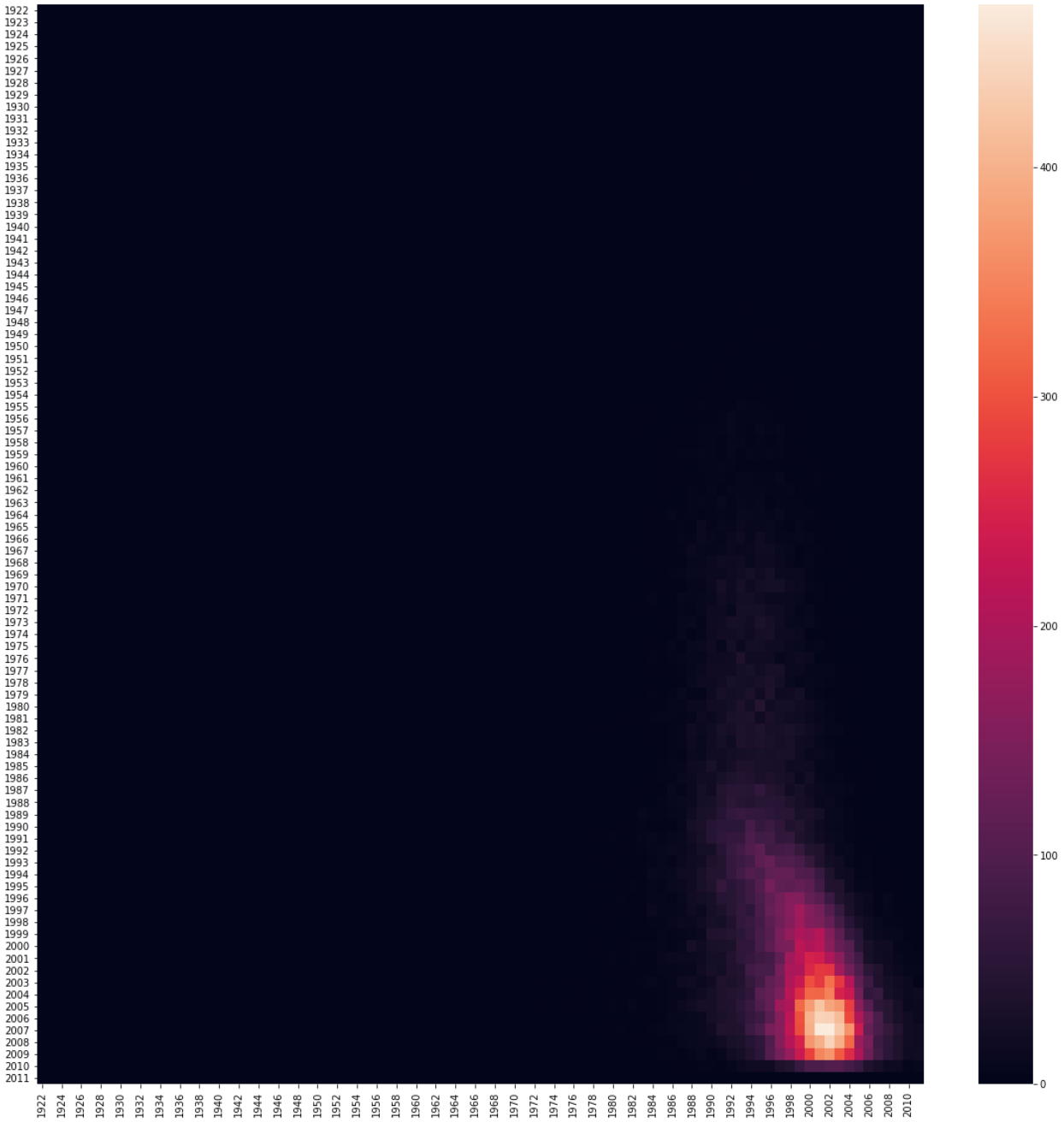
جدول تغییر پارامترها برای رگرسیون را در زیر مشاهده می‌کنید. بهترین نتیجه برای دقت مجموعه آموزش و اعتبارسنجی با رنگ قرمز مشخص شده است.

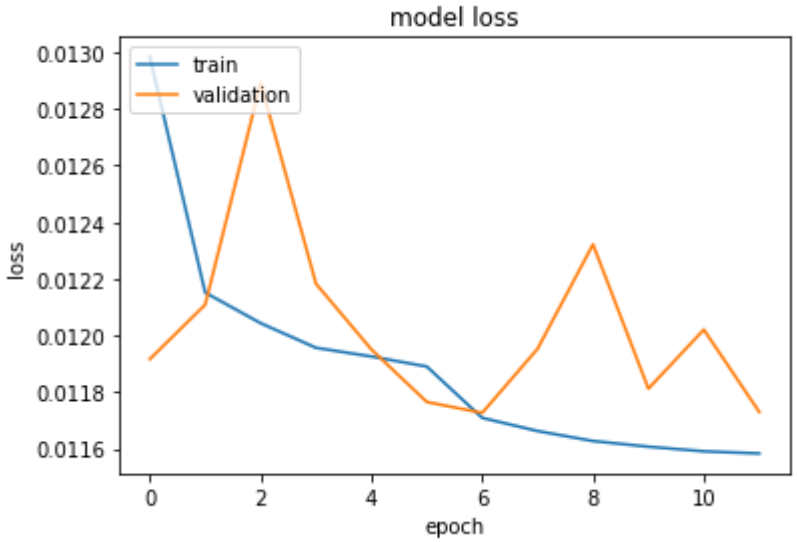
Trial #	# of hidden layers	Neuron in each layer	Learning Rate	Epoch	Train MSE error	Val MSE error
1	1	1024	0.001	12	0.0119	0.0117
2	1	512	0.001	2	0.0125	0.0119
3	1	128	0.001	4	0.0120	0.0118
4	1	64	0.001	8	0.0118	0.0117
5	2	256,128	0.001	17	0.0117	0.0117
6	2	128,256	0.001	11	0.0117	0.0117
7	2	128,128	0.001	5	0.0120	0.0118
8	2	128,512	0.001	18	0.0118	0.0117
9	3	128,256,128	0.001	8	0.0118	0.0117
10	3	64,128,64	0.001	6	0.0120	0.0118
11	3	128,256,512	0.001	4	0.0123	0.0117
12	2	256,128	0.0002	8	0.0117	0.0117
13	2	256,128	Adaptive, init=0.0002 After 5 epochs, decay: Init/(epoch-4)	7	0.0116	0.0117

متاسفانه برای رگرسیون به علت حواسپرتی تابع فعالیت یادم رفت بگذارم. نزدیک به زمان تحویل متوجه شدم. نتایج را همانگونه که بود می‌گذارم و نتیجه‌ی مدل نهایی همراه با تابع فعالیت Relu را نیز می‌گذارم. نتایج با استفاده از این تابع فعالیت بهتر می‌شود اما آموزش دیرتر متوقف می‌شود؛ به این علت که برخی وزن‌ها صفر می‌شوند. این اتفاق در دسته‌بندی نمی‌افتد بخاطر اینکه در آنجا به دنبال احتمال هر کلاس برای نورون‌های خروجی از تابع softmax استفاده کردیم. در اینجا اما نورون خروجی تابع همانی دارد که موجب می‌شود مقادیر همانی که محاسبه می‌شوند باقی بمانند. علت یکسان شدن نتایجی که مشاهده می‌شود نیز ممکن است نبودن تابع فعالیت باشد؛ چرا که دیگر مقادیر منفی وزن‌ها باقی می‌مانند و به همان میزانی که مقادیر مثبت بزرگتر یا کوچکتر شوند، مقادیر منفی نیز بزرگ یا کوچک خواهند شد و در نتیجه‌ی محاسباتی که صورت می‌گیرد وزن‌ها و در نهایت نتایج نیز تغییر چندانی نخواهند داشت.

# of hidden layers	Neuron in each layer	Learning Rate	Epoch	Train MSE error	Val MSE error	Test Accuracy
2	256,128	Adaptive, init=0.0002 After 5 epochs, decay: Init/(epoch-4)	7	0.0116	0.0117	0.055049
2 (Relu activation function) *	256,128	Adaptive, init=0.0002 After 5 epochs, decay: Init/(epoch-4)	33	0.0101	0.0103	0.0606

به نظر می‌رسد آنچه در مورد دسته‌بندی گفتیم در اینجا نیز تا حدودی صادق است. در شبکه‌های تک لایه، افزایش تعداد نورون‌ها پاسخ بهتری داده است. تفاوت اصلی در این است که با داشتن دو لایه، تعداد نورون‌های لایه‌ی اول بیشتر از لایه‌ی دوم در نظر گرفته شود نتیجه بهتری می‌دهد. این احتمال وجود دارد که این مسئله به علت تعداد نورون‌های خروجی باشد.




model loss
train
validation
loss
epoch
0.0130
0.0128
0.0126
0.0124
0.0122
0.0120
0.0118
0.0116
0
2
4
6
8
10