به نام خدا

آشنایی و کار با شبکه های عصبی مصنوعی زهرا علی زاده 400522058

بخش اول :

تفاوت در دقت و عملکرد مدل یادگیری درخت تصمیم و مدل یادگیری شبکه عصبی مصنوعی :

دقت داده های آزمایشی	دقت داده های آموزشی	
0.9349	0.9578	درخت تصمیم گیری
0.94711	0.984143	شبکه عصبی مصنوعی

نکته :

دقت ذکر شده برای مدل شبکه عصبی میانگین حاصل ازنتایج 4-fold cross validation است.

همانطور که از جدول بالا مشخص است میان دقت دو مدل هم در بخش آموزش و هم در بخش آزمایش تفاوتی درحدود 4 درصد مشاهده میشود که علت آن قدرت مدل یادگیری شبکه عصبی در زمانیکه داده زیاد است میباشد. مزیت دیگر مدل شبکه عصبی دوری از overfitting در عین_متوسط رو به بزرگ_ شدن شبکه است، و همچنین میتواند روابط پیچیده تر را راحت تر از مدل درخت تصمیم گیری کشف کند.

بخش دوم :

تخمین تابع :

در این بخش تعدادی نقاط اموزشی از یک تابع به مدل یادگیری داده میشود و همیچنین نقاط تست و در نهایت میزان واقعی خروجی تابع برای نقاط آزمایشی و همچنین پیش بینی مدل برای این نقاط در یک نمودار به نمایش در میاید .

معیار های بررسی شده در این بخش :

1- تعداد نقاط ورودی
2- پیچیدگی تابع
3- تعداد لایه ها
4- بازه داده های ورودی

5- تعداد چرخه های شبکه

یکی از نتایج مدل یادگیری در نمودار زیر آورده شده است.

بررسی نتیجه :

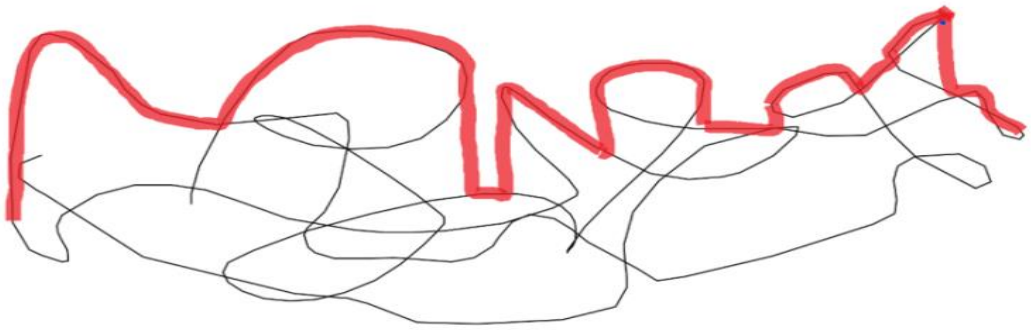
عوامل موثر در تخمین تابع به شرح زیر است :

1- پیچیدگی تابع : در میان توابعی که تعریف شد، هرچه نمودار تابع پیچیده تر شد میزان دقت تخمین شبه با توجه به میزان "پرش های ناگهانی" های موجود دچار نوسان و در اکثر موارد کاهش دقت میشد.
2- تعداد لایه ها و نورون های هر لایه : واضح هست که با افزایش تعداد لایه ها و نورون های موجود در هر لایه میزان دقت شبک افزایش پیدا میکند و میزان کاهش دقتی که توسط تابع پیچیده در نتایج مشاهده میشود را بسیار کم و نزدیک به صفر میکند.
3- وسعت داده ورودی : هرچه که داده های ورودی به مدل در دامنه وسیع تر موجود باشند خب دقت بالاتری را شاهد خواهیم بود زیرا مدل میتواند داده های تست بخصوص x هایی که در معادلاتتون داده میشود را لازم داریم .

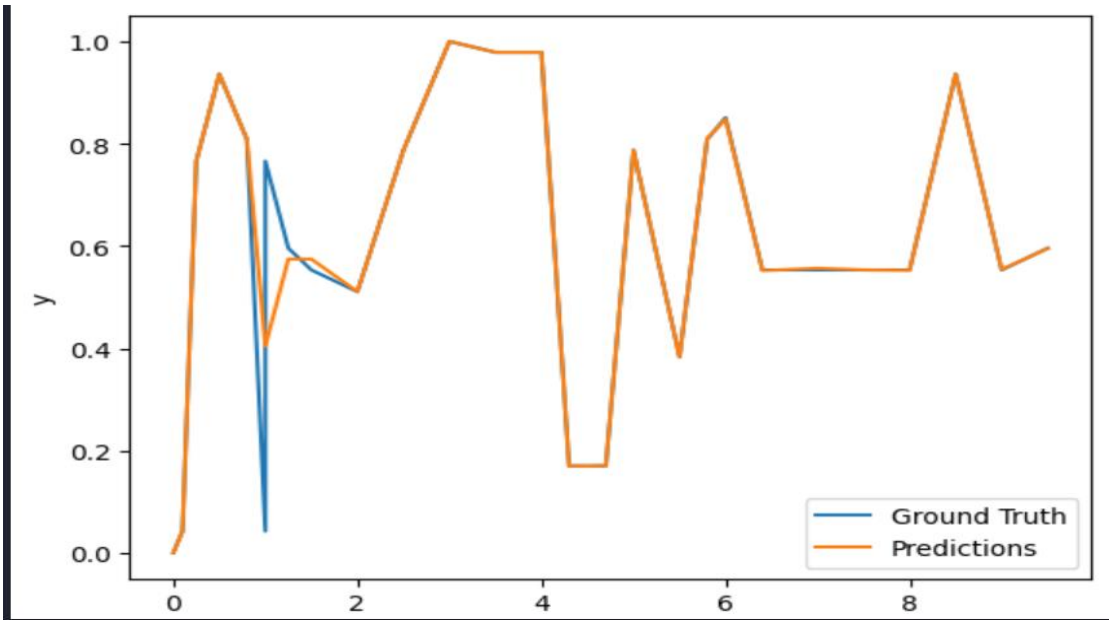
بخش سوم :

در این بخش همانند بخش قبلی تخمین تابع را داریم ولی تفاوت آن در داده ورودی است که در این بخش دارای Guassian noise شده است و هدف تخیمن صحیح داده با وجود نویز میباشد . کتتابخونه مورد استفاده در این بخش مدل sklearnمیباشد ک. برای رفع اثر این مقدار نویز دهی بایسیت شبکه را بزرگتر و تعداد نورون ها در هر لاهایه افزایش یاد .

بخش چهارم : تخمین تابع حاصل از خط خطی 😊

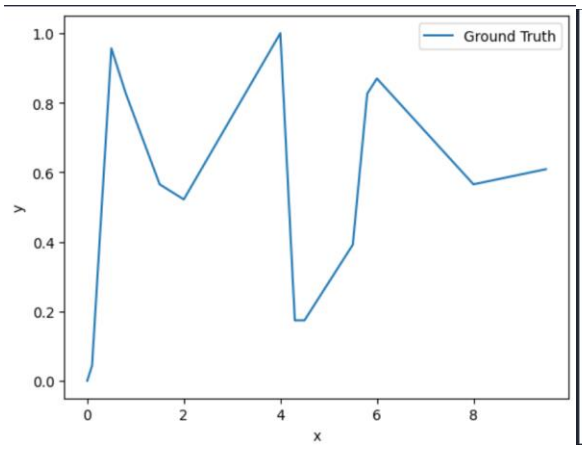


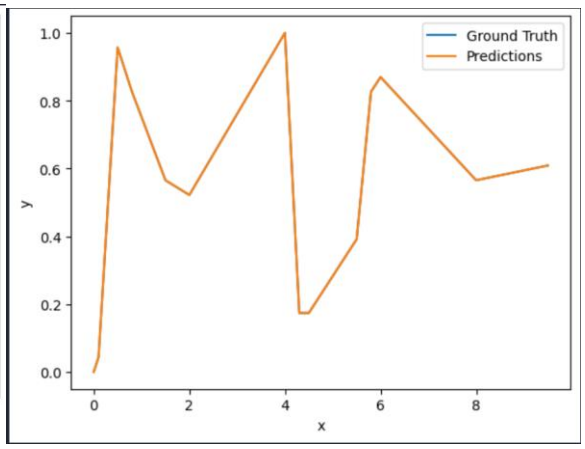
نمودار حاصل از خط خطی بالا به عنوان تابع ورودی در نظر گرفته شد که تابعی نسبتا پیچیده است.
برای اینکه تابع مناسب ورودی دادن به شبکه باشد تعداد نقاط فرضی به همراه خروجی آن به صورت زیر در نظر گرفته شد به عنوان ورودی به شبکه داده شد.



در این بخش چون تابع مشخصی موجود نبود بایستی در تعداد دوره های نسبتا زیاد مدل را آموزش داد تا بتواند خود تابع اصلی را پیش بینی کند (حدودا 80000) . همچنین در نقاط پرش ناگهانی که در نمودار بالا مشخص است پیش بینی تابع میزانی از تابع اصلی دور است چون مدل به دنبال پیدا کردن خطی است که از دو نقطه اطراف "پرش ناگهانی" عبور کند پس در این نقطه توابع واقعی و پیش بینی همپوشانی ندارند.

در تابع پایین کاملا دو تابع پیش بینی شده و تابع واقعی با یکدیگر همپوشانی دارد چونکه از پیچیدگی تابع کاسته شده و دارای پرش ناگهانی نیست .





پارت پنجم :

در این بخش از دیتاست MNIST که شامل ارقام دست نویس 0 تا 9 است، استفاده شد . هدف ایجاد یک شبکه با قابلیت classify کردن داده ها است.

مراحل اینجاد شبکه و تست آن :

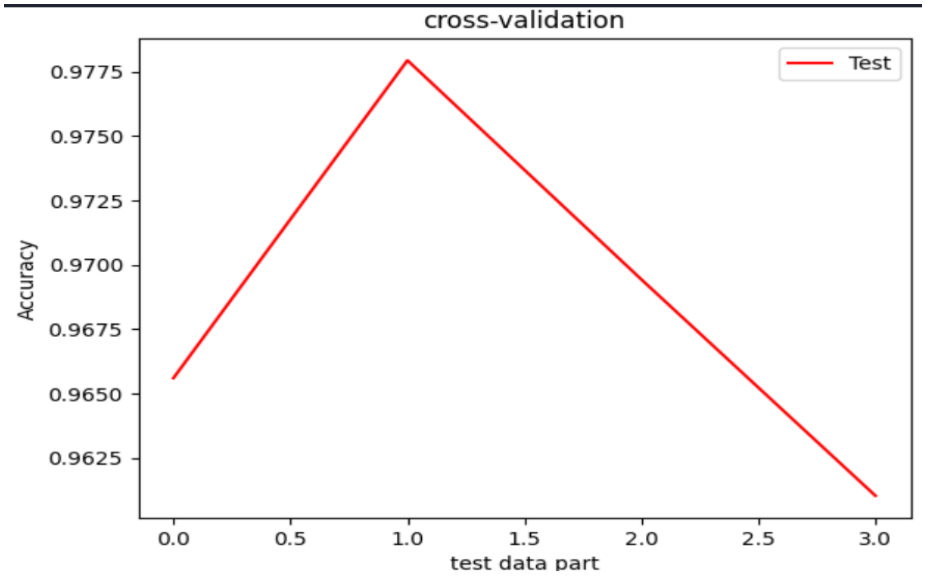
1- کتابخانه مورد استفاده : کتابخانه tensorflow مورد استفاده قرار گرفته شد.
2- لود دیتا :
کتابخانه tensorflow دارای دیتا ست ذکر شده میباشد و برای لود دیتای تست و ترین از تابع load موجود برای این دیتاست استفاده شد. 10000 داده تست و 60000 داده آموزشی .
3- ویژگی های دیتاست :
این دیتا ست حاوی 10 کلاس است که هرکلاس با لیبل رقم آن کلاس برچسب زده شده.
عکس های موجود در این دیتاست دارای سایز 28x28 هستند که مبنی بر آن است دیتا ست ساده ای میباشد و کامپوننت های مجزا در این دیتا ست کم است .
4- شبکه عصبی :
شبکه ایجاد شده دارای 4 لایه dense است که تابع فعال ساز آن "relu" و "tanh" میباشد و لایه اخر دارای تابع فعال ساز "softmax" میباشد .Dense یک تابع از کتابخانه TF میباشد که این تابع برای ایجاد یک لایه Fully Connected در شبکه عصبی استفاده میشود که دارای تابع های فعال ساز ذکر شده و تعداد واحد مشخص میتواند باشد.

ابتدا با دسته بندی دیتا لود شده توسط TF مدل را train و test میکنیم :

Train accuracy : 0.979

Test accuracy : 0.9667

سپس از تکنیک 4-fold cross validation برای تقسیم داده ها و تست آن استفاده میکنیم .
نتیجه حاصل شده به صورت میانگین : 0.9685



همانطور که مشخص است در حالت که یک چهارمِ دوم دیتا به تست اختصاص داده شود بیشترین مقدار دقت را خواهیم داشت، که خود نمایانگر اثر تقسیم داده ها بر دقت نهایی است .

پارت ششم :

در این بخش قرار است مدل شبکه عصبی را در رفع نویز تصاویر ارقام بخش قبل تست کنیم و چک کنیم که آیا مدل توانایی اینکار را دارد ؟ با چه لایه هایی ؟

پاسخ "بله" است !!

مراحل طی شده :

1- نویزی کردن عکس ها :
این عمل با استفاده تکنیک Guassian noise انجام پذیرفت که باعث میشود به پیکسل های عکس به صورت رندوم در بازه و میزان معین شده یک noise اضافه شود. عکس های زیر درحالت عادی و نویزی نمایش داده شده اند .

مقدار نویزی که در عکس بالا مشاهده میشود، متوسط است .

2- ایجاد شبکه عصبی : شبکه عصبی مورد استفاده در این بخش یک Convolutional NueralNetwork میباشد که مشابه بخش قبل با استفاده از کتابخانه Tensor Flow ایجاد شده است. اما به علت هدف متفاوت، لایه های مورد استفاده در این بخش متفاوت است.
تفسیر لایه ها :

لایه های مورد استفاده در این شبکه شامل convolution 2d به همراه pooling های متعدد بر روی داده عکس ها برای از بین بردن نویز و جدا کردن کامپوننت اصلی تصویر که همان رقم موجود است، میباشد . تابع فعال ساز استفاده شده نیز "rlue" میباشد . که شامل دو مرحله میباشد :

1- Encoding

2- Decoding

داده های عکس

3- آموزش شبکه : داده آموزشی شبکه عکس های نویز دار داده آموزشی میباشد و خروجی داده شده به شبکه عکس اورجینال داده آموزشی است .

4- تست شبکه : داده آزمایشی شبکه عکس های نویزدار آزمایشی میباشد که خروجی عکس رفع نویز شده میباشد .

تست شبکه در نویز با سه سطح کم ، متوسط و زیاد انجام شد.

با توجه به نمودار زیر با افزایش مقدار نویز باز هم شبکه قادر به رفع نویز خواهد بود اما دقت آن کاهش میباید .