

به نام خداوند بخشنده‌ی مهربان

## گزارش تمرین عملی سوم هوش مصنوعی

### ماشین بردار پشتیبان (SVM)

نیم‌سال اول ۱۴۰۰-۱۴۰۱

سید پارسا نشایی - ۹۸۱۰۶۱۳۴

دانشگاه صنعتی شریف  
دانشکده‌ی مهندسی کامپیوتر

#### مقدمه

در این تمرین عملی، به استفاده از SVM برای حل مسائل داده شده در صورت تمرین پرداخته شده است. برنامه‌ها به زبان Python نوشته شده‌اند. در نهایت، نتایج تحلیل و بررسی شده و چالش‌ها بیان شده‌اند. کتابخانه‌های استفاده شده، numpy برای عملیات مقدماتی برداری ریاضی (به اسم np در کد وارد شده) و نیز matplotlib.pyplot برای رسم نمودار (به اسم plt در کد وارد شده) هستند. افزون بر این دو کتابخانه، از keras برای دریافت عکس‌های MNIST و نیز از sklearn برای محاسبات SVM استفاده شده است.

#### بخش اول

فایل این بخش، ۱ است.

در این بخش، چند مسئله‌ی ۲ کلاسه با اشکال مختلف توابع داخلي **sklearn** طراحی شده و سپس الگوريتم **SVM** اقدام به تميز دادن کلاس‌ها از يك ديگر كرده است. در نهايـت، خطوط حاصل برای خط جداـنـده و **margin** ها در هر بخش رسم شده‌اند.

بخش اصلی کد، سه تابع **run\_rbf**، **run\_linear** و **run\_polynomial** است که الگوريتم **SVM** را با هسته (کرنل)‌های به ترتیب **rbf**، خطی (عادی) و چندجمله‌ای اجرا می‌کنند. در اجرا با کرنل چندجمله‌ای، تابع دو پارامتر «درجه» و «مقدار ثابت» را نیز برای چندجمله‌ای دریافت خواهد کرد. تمام این توابع یک ورودی **k** هم دارند که **k-fold** بودن **classifier** برگردانده شده و قبل از خروج از تابع، امتیاز کسب شده روی داده‌های تست چاپ می‌شود تا بتوان از لحاظ کمی کرنل‌ها را با یک‌دیگر مقایسه کرد.

تابع **plot** نیز وظیفه‌ی رسم خطوط و نقاط شکل‌ها را به عهده دارد.

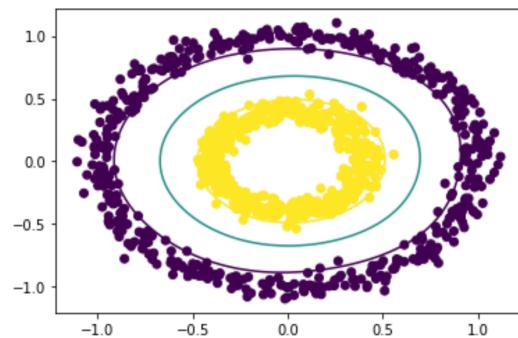
داده‌های اصلی داده شده به شرح زیرند:

- حلقه‌های تقریباً متعددالمرکز، به طوری که تمام داده‌ها از یک کلاس درون حلقه‌ی حاصل از داده‌های کلاس دیگر باشند. این داده‌ها به کمک متدهای **make\_circles** از بخش **datasets** پکیج **sklearn** قابل ساختن هستند.
- ورودی‌های این متدهای عبارت‌اند از تعداد نمونه‌ها، میزان نویز داده‌ها، ضریب اندازه‌ی دو حلقه نسبت به هم و همچنین پارامتر **random\_state** که برابر عددی ثابت تنظیم شده تا در اجراهای متوالی برنامه، جواب یکسانی دریافت شده و بتوان کد را به راحتی **debug** کرد.
- دو ناحیه‌ی از هم جدا که به کمک متدهای **make\_blobs** از بخش **datasets** پکیج **sklearn** قابل ساختن هستند.
- ورودی‌های این متدهای عبارت‌اند از تعداد نمونه‌ها، میزان نویز داده‌ها، انحراف معیار داده‌ها از مرکز ناحیه‌شان، تعداد ناحیه‌ها (مرکزهای تجمع داده‌ها) و همچنین پارامتر **random\_state** که برابر عددی ثابت تنظیم شده تا در اجراهای متوالی برنامه، جواب یکسانی دریافت شده و بتوان کد را به راحتی **debug** کرد.
- دو ناحیه‌ی هلالی شکل که در درون یک‌دیگر فرو رفت‌هایند که به کمک متدهای **make\_moons** از بخش **datasets** پکیج **sklearn** قابل ساختن هستند.

- ورودی‌های این متدها عبارت‌اند از تعداد نمونه‌ها، میزان نویز داده‌ها و همچنین پارامتر `random_state` که برابر عددی ثابت تنظیم شده تا در اجراهای متوالی برنامه، جواب یکسانی دریافت شده و بتوان کد را به راحتی `debug` کرد.

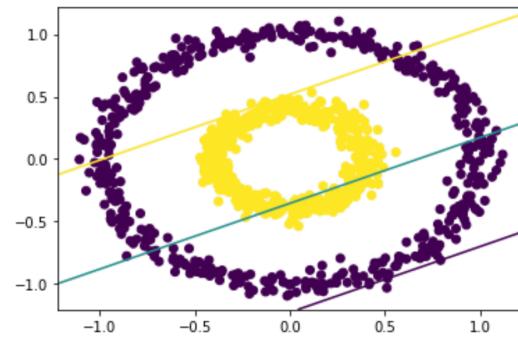
## مشاهده، تجزیه و تحلیل نتایج

- حلقه‌های تو در تو با ۱۰۰۰ نمونه و نویز ۰.۰۵ با کرنل `rbf`:



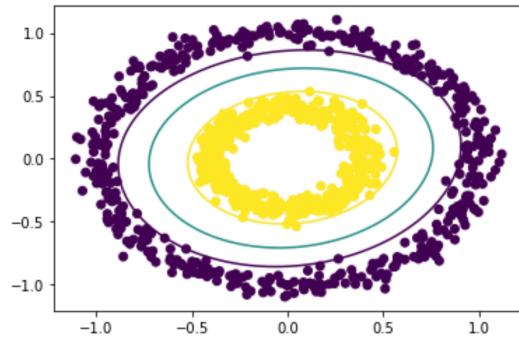
امتیاز تست برابر ۱ شده است که به آن معناست که خط (منحنی) جداساز کاملاً دقیق بوده و داده‌ها به خوبی از یکدیگر جدا شده‌اند.

- حلقه‌های تو در تو با ۱۰۰۰ نمونه و نویز ۰.۰۵ با کرنل خطی:



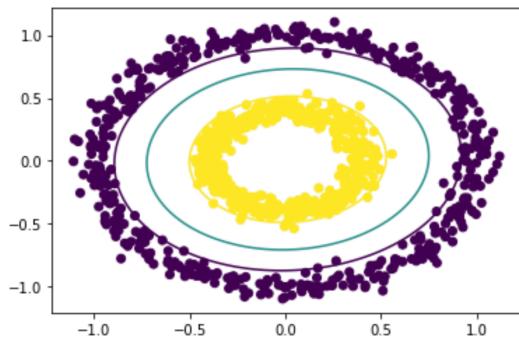
بر اساس آن‌چه از کلاس درس می‌دانیم، کرنل خطی قادر به جدا سازی داده‌های به فرم حلقه‌های تو در تو نیست که در شکل فوق نیز خطای آن واضح است. امتیاز تست برابر ۰.۶ شده است که عملاً نزدیک به نصف است و به معنای آن است که در نزدیک به نصف موقع (به عبارت دیگر تقریباً معادل انتخاب تصادفی)، جواب غلط خواهد بود.

- حلقه‌های تو در تو با ۱۰۰۰ نمونه و نویز ۰.۰۵ با کرنل چندجمله‌ای درجه ۳ با ثابت ۱:



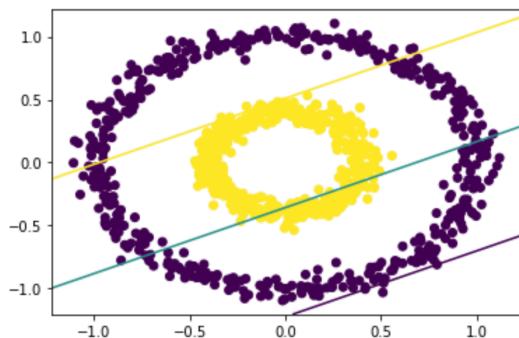
امتیاز تست برابر ۱ شده است که به آن معناست که خط جداساز کاملا دقیق بوده و دادهها به خوبی از یکدیگر جدا شده‌اند.

- حلقه‌های تو در تو با ۱۰۰۰ نمونه و نویز ۰.۰۵ با کرنل چندجمله‌ای درجه ۲ با ثابت ۱:



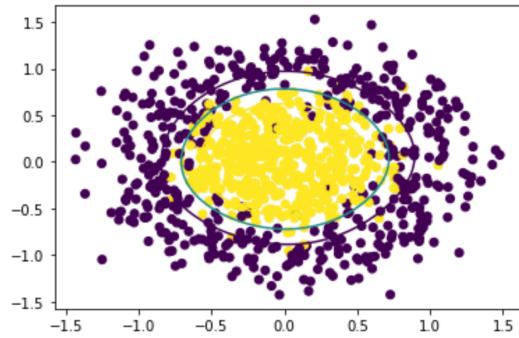
امتیاز تست برابر ۱ شده است که به آن معناست که خط جداساز کاملا دقیق بوده و دادهها به خوبی از یکدیگر جدا شده‌اند.

- حلقه‌های تو در تو با ۱۰۰۰ نمونه و نویز ۰.۰۵ با کرنل چندجمله‌ای درجه ۱ (عملای خطی) با ثابت ۱:



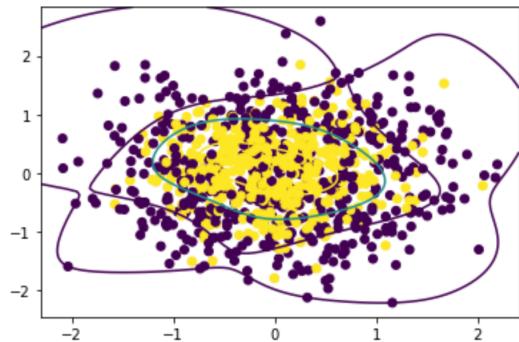
بر اساس آن‌چه از کلاس درس می‌دانیم، کرنل خطی قادر به جدا سازی داده‌های به فرم حلقه‌های تو در تو نیست که در شکل فوق نیز خطای آن واضح است. امتیاز تست برابر ۸.۶۰ شده است که عملای نزدیک به نصف است و به معنای آن است که در نزدیک به نصف موقع (به عبارت دیگر تقریباً معادل انتخاب تصادفی)، جواب غلط خواهد بود.

- حلقه‌های تو در تو با ۱۰۰۰ نمونه و نویز ۰.۲ با کرنل `rbf`:



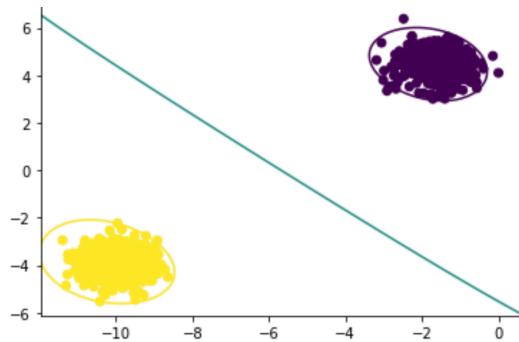
با وجود افزایش نویز و اختلاط نقاط، همچنان خط خوبی به دست آمده و امتیاز تست برابر  $0.94$  شده است که به آن معناست که منحنی جداساز تقریبا دقیق بوده و داده‌ها به خوبی از یکدیگر جدا شده‌اند.

- حلقه‌های تو در تو با  $1000$  نمونه و نویز  $0.5$  با کرنل  $\text{rbf}$ :



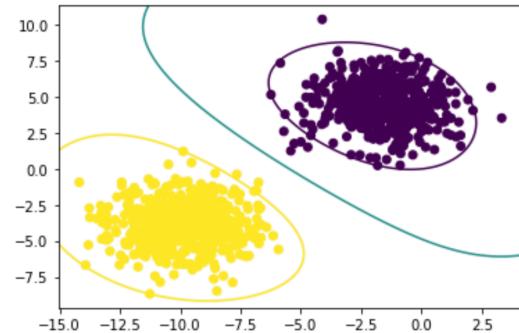
این بار میزان نویز به حدی بالا بوده که الگوریتم دچار خطا شده، خط‌های نامعقولی خروجی داده و امتیاز تست نیز به  $0.7$  رسیده که پایین‌تر از بخش قبلی است.

- ناحیه‌های جدا با  $1000$  نمونه و انحراف از مرکز  $0.5$  با کرنل  $\text{rbf}$ :



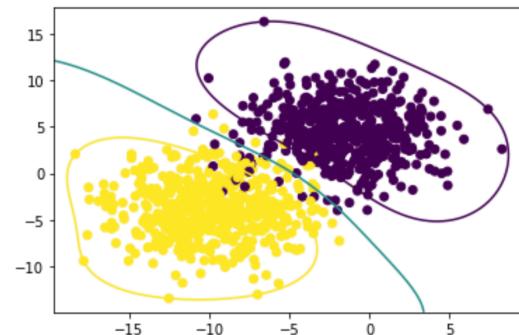
امتیاز تست برابر  $1$  شده است که به آن معناست که خط جداساز کاملا دقیق بوده و داده‌ها به خوبی از یکدیگر جدا شده‌اند.

- ناحیه‌های جدا با  $1000$  نمونه و انحراف از مرکز  $1.5$  با کرنل  $\text{rbf}$ :



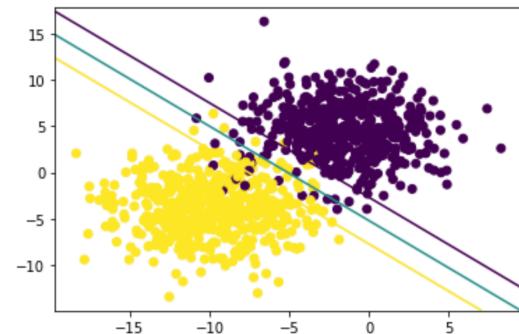
امتیاز تست برابر  $1$  شده است که به آن معناست که خط جداساز کاملاً دقیق بوده و داده‌ها به خوبی از یک‌دیگر جدا شده‌اند.

- ناحیه‌های جدا با  $1000$  نمونه و انحراف از مرکز  $3$  با کرنل  $\text{rbf}$ :



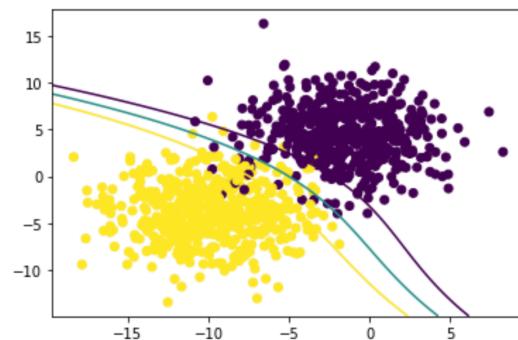
به دلیل مقداری اختلاط نقاط، امتیاز تست برابر  $0.992$  شده است ولی البته همچنان خط جداساز نسبتاً دقیق است و داده‌ها به خوبی از یک‌دیگر جدا شده‌اند.

- ناحیه‌های جدا با  $1000$  نمونه و انحراف از مرکز  $3$  با کرنل خطی:



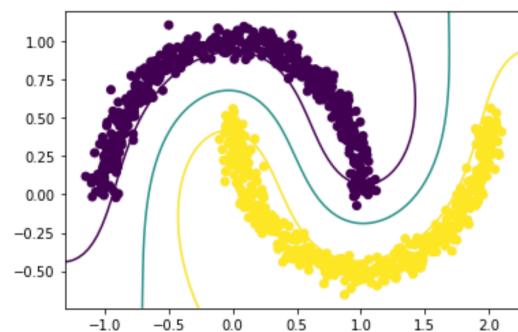
به دلیل مقداری اختلاط نقاط، امتیاز تست برابر  $0.992$  شده است ولی البته همچنان خط جداساز نسبتاً دقیق است و داده‌ها به خوبی از یکدیگر جدا شده‌اند. با این‌که کرنل خطی در ظاهر ضعیفتر از `rbf` است، ولی در این داده‌ها چون ماهیتاً دو ناحیه‌ی جدا از هم بوده و در نتیجه منطبقاً به صورت خطی جذاب‌تر هستند، نتیجه‌ی یکسانی با بخش قبل حاصل شده است.

- ناحیه‌های جدا با  $1000$  نمونه و انحراف از مرکز  $3$  با کرنل چندجمله‌ای با درجه  $3$  و ثابت  $1$ :



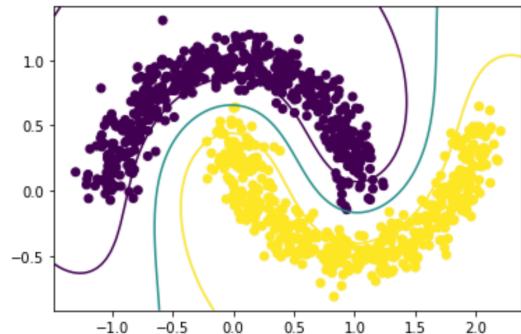
کرنل چندجمله‌ای نیز در این داده‌های به خصوص به دلیل ماهیت جذاب‌تر بودن خطی آن‌ها مانند `rbf` و خطی عمل می‌کند و امتیاز تست برابر  $0.992$  شده است.

- هلال‌های درون هم با  $1000$  نمونه و نویز  $0.05$  با کرنل `rbf`:



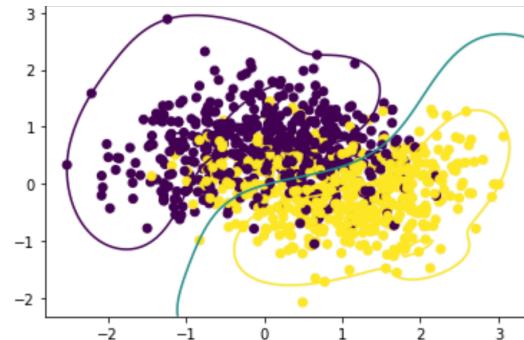
امتیاز تست برابر  $1$  شده است که به آن معناست که خط (منحنی) جداساز کاملاً دقیق بوده و داده‌ها به خوبی از یکدیگر جدا شده‌اند.

- هلال‌های درون هم با  $1000$  نمونه و نویز  $0.1$  با کرنل `rbf`:



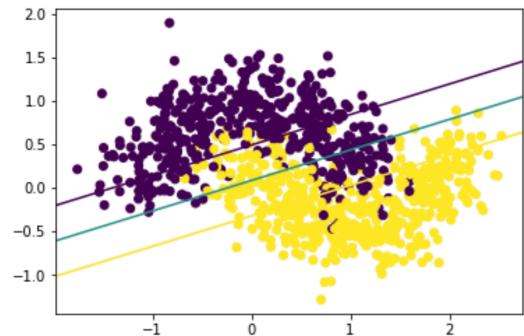
امتیاز تست برابر ۱ شده است که به آن معناست که خط (منحنی) جداساز کاملا دقیق بوده و داده‌ها به خوبی از یکدیگر جدا شده‌اند.

- هلال‌های درون هم با ۱۰۰۰ نمونه و نویز ۰.۵ با کرنل `:rbf`



به دلیل اختلاط نقاط، امتیاز تست برابر ۰.۸۱۲ شده است، اما همچنان خط جداساز توانسته بخش‌های عمدتاً زرد و عمدتاً بنفش را از یکدیگر جدا کند که به آن معناست که خط (منحنی) جداساز تا حد نسبتاً مناسبی، دقیق بوده و داده‌ها به مقدار نسبتاً مناسبی از یکدیگر جدا شده‌اند.

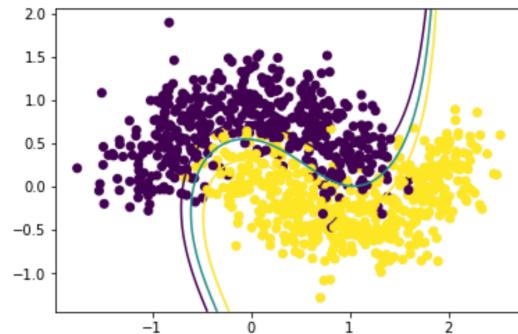
- هلال‌های درون هم با ۱۰۰۰ نمونه و نویز ۰.۲۵ با کرنل خطی:



به دلیل اختلاط نقاط و دقیقاً خطی نبودن مرز دو کلاس، امتیاز تست برابر ۰.۸۶۴ شده است، اما همچنان خط جداساز توانسته بخش‌های عمدتاً زرد و عمدتاً بنفش را از یکدیگر جدا کند که به آن معناست که خط (منحنی) جداساز تا حد نسبتاً

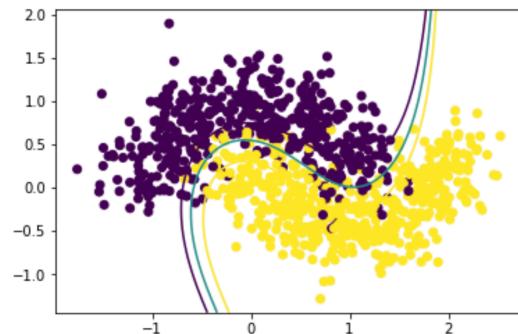
مناسبی، دقیق بوده و دادهها به مقدار نسبتاً مناسبی از یکدیگر جدا شده‌اند، گرچه احتمالاً کرنل  $rbf$  و یا چندجمله‌ای در این مثال، ما را به جواب بهتری می‌رساند.

• هلال‌های درون هم با ۱۰۰۰ نمونه و نویز ۰.۲۵ با کرنل چندجمله‌ای با درجه ۳ و ثابت ۱:



برخلاف کرنل خطی و به دلیل ماهیت شکل هلالی، کرنل چندجمله‌ای توانسته جداسازی بهتری (ولی طبعاً نه صد درصد) انجام دهد و امتیاز تست برابر ۰.۹۴۴ شده است که به آن معناست که خط (منحنی) جداساز تا حد زیادی دقیق بوده و دادهها به مقدار قابل قبولی از یکدیگر جدا شده‌اند.

• هلال‌های درون هم با ۱۰۰۰ نمونه و نویز ۰.۲۵ با کرنل چندجمله‌ای با درجه ۶ و ثابت ۱:



این قسمت مشابه قسمت قبل بوده، ولی درجه کرنل زیاد شده است که این میزان پیچیدگی برای جداسازی نقاط لازم نبوده و اتفاقاً افزایش پیچیدگی به کاهش اندک امتیاز تست به ۰.۹۴ انجامیده است؛ اما در کل همچنان امتیاز بالایی محسوب می‌شود که به آن معناست که خط (منحنی) جداساز تا حد زیادی دقیق بوده و دادهها به مقدار قابل قبولی از یکدیگر جدا شده‌اند.

به طور کلی نتیجه می‌گیریم که با زیاد بودن پیچیدگی داده‌ها بهتر است به سمت کرنل  $rbf$  برویم که بتواند جداسازی بهتری نسبت به کرنل خطی در داده‌هایی که لزوماً به خوبی جدایی‌پذیر از هم نیستند، انجام دهد و نیز پارامتر درجه‌ی چندجمله‌ای را تنها هنگامی بالا ببریم که پیچیدگی داده‌ها زیاد باشد، وگرنه امتیاز تست کمتر می‌شود.

## بخش دوم

در این بخش، دسته‌بندی پایگاه داده **MNIST** (ارقام انگلیسی با دستخط‌های مختلف) به کمک **SVM** انجام گرفته است. کد توابع **run** مانند بخش قبل است و البته امتیاز هم **train** و هم **test** را چاپ می‌کنند.

در ابتدا داده‌ها را به کمک **keras** لود کرده و سپس برای بین صفر و یک کردن مقادیر پیکسل‌ها، آن‌ها را بر ۲۵۵ تقسیم می‌کنیم. سپس تمام عکس‌ها را به یکدیگر پیوند زده، به شکل یک بردار **T** بعدی درآورده و هر سه دسته‌بندی کننده‌ی با کرنل خطی و **rbf** و چندجمله‌ای را اجرا می‌کنیم.

### مشاهده، تجزیه و تحلیل نتایج

- دسته‌بندی کننده‌ی با کرنل خطی، به امتیاز  $0.979228571428571$  روی داده‌های آزمایشی می‌رسد. (احتمالاً کمی بیش برآش رخ داده است)
- دسته‌بندی کننده‌ی با کرنل **rbf**، به امتیاز  $0.9891142857142857$  روی داده‌های آموزشی و  $0.9742$  روی داده‌های آزمایشی می‌رسد.
- دسته‌بندی کننده‌ی با کرنل چندجمله‌ای با درجه  $3$  و ثابت  $1$ ، به امتیاز  $0.9989142857142858$  روی داده‌های آموزشی و  $0.9761428571428571$  روی داده‌های آزمایشی می‌رسد.

نتیجه می‌گیریم که کرنل **rbf** و چندجمله‌ای عملکرد بهتری نسبت به کرنل خطی دارد؛ کرنل **rbf** پس از چندجمله‌ای و در انتهای نیز کرنل خطی قرار دارد. (البته اختلاف کرنل چندجمله‌ای و **rbf** ناچیز و در عمل بی‌اهمیت است) دقت این روش، بسیار بالا به دست آمده است، اما زمان اجرای این روش (احتمالاً به دلیل بهره نبردن **sklearn** از برخی از بهینه‌سازی‌های انجام شده در **GPU**) بسیار بالا بوده است (حدوداً  $9$  دقیقه برای کرنل خطی، حدوداً  $9$  دقیقه برای کرنل چندجمله‌ای و حدوداً  $20$  دقیقه برای کرنل **rbf**) که در نتیجه در عمل در بسیاری از موارد، ممکن است به دلیل زمان بالایی که این روش می‌گیرد، تصمیم گرفته شود که از کتابخانه‌های شبکه عصبی مانند **TensorFlow** به جای استفاده از روش **SVM** درون **sklearn** بهره گرفته شود.

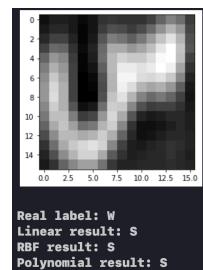
## بخش سوم

در این بخش، دسته‌بندی پایگاه داده ارقام پلاک انجام گرفته است. کد توابع `run` مانند بخش قبل است و البته امتیاز هم `train` و هم `test` را چاپ می‌کنند. در ابتدا، فایل‌های داده شده خوانده شده و اطلاعات عکس‌های آن‌ها در آرایه پس از تقسیم مقادیر پیکسل‌ها به ۲۵۵ ذخیره می‌شود (به کمک `matplotlib`). سپس، داده‌ها به شکل بردار `n` بعدی درآمده و به توابع `run` با کرنل‌های مختلف داده می‌شوند. در نهایت، یک نمونه از حروفی که به تصادف دسته‌بندی شده‌اند آورده شده است.

### مشاهده، تجزیه و تحلیل نتایج

- دسته‌بندی کننده‌ی با کرنل خطی، به امتیاز ۰.۹۷۴۶۶۶۶۶۶۶۶۶۶۶۷ روی داده‌های آموزشی و ۰.۹۶۹۳۲۲۲۲۲۲۲۲۲۴ روی داده‌های آزمایشی می‌رسد.
- دسته‌بندی کننده‌ی با کرنل `rbf`، به امتیاز ۰.۹۶۸ روی داده‌های آموزشی و ۰.۹۵۳۲۳۲۳۲۳۲۳۲۳۲۳۴ روی داده‌های آزمایشی می‌رسد.
- دسته‌بندی کننده‌ی با کرنل چندجمله‌ای با درجه ۳ و ثابت ۱، به امتیاز ۰.۹۸۶۶۶۶۶۶۶۶۶۶۶۶۶۷ روی داده‌های آموزشی و ۰.۹۶۸ روی داده‌های آزمایشی می‌رسد.

این بار کرنل `rbf` کمترین امتیاز را گرفته است و کرنل چندجمله‌ای رتبه‌ی دوم و کرنل خطی بهترین امتیاز را روی داده‌های آموزشی داشته است! معنای این مشاهده می‌تواند آن باشد که داده‌ها در فضای تشکیل شده به صورت خطی جدایذیر بوده‌اند. یک نمونه از داده‌های به اشتباہ تشخیص داده شده، مورد زیر بوده که تشخیص آن برای انسان نیز دشوار است:



## چالش‌های مسیر

اصلی‌ترین چالشی که در مسیر انجام این تمرین وجود داشت، زمان زیاد مورد نیاز برای الگوریتم `SVM` روی `MNIST` بود که دیباگ را دشوار می‌کرد. چالش دیگر، چگونگی ساختن داده‌های چند کلاسه به کمک متدهای خود کتابخانه‌ها بود که پاسخ به آن، متدهای `*_make` بودند.