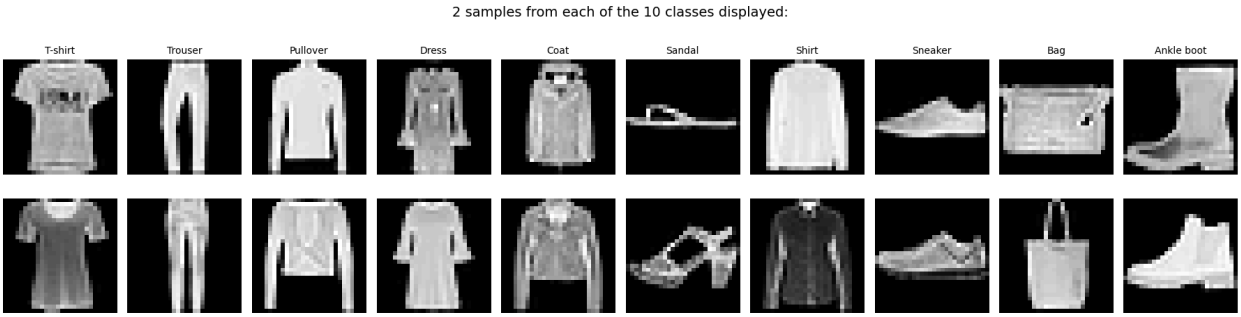
گزارش تمرین دوم یادگیری ماشین - مهدیس رحمانی - 40313141027

۱. معرفی مجموعه داده Fashion-MNIST

مجموعه داده‌ی Fashion-MNIST شامل ۷۰٬۰۰۰ تصویر خاکستری ۲۸×۲۸ پیکسلی از لباس‌های مختلف است. این تصاویر در ۱۰ دسته‌ی مختلف قرار دارند: تی‌شرت/تاپ، شلوار، پلیور، پیراهن، کت، صندل، پیراهن مردانه، کتانی، کیف، و بوت. این مجموعه جایگزین مناسبی برای داده‌ی MNIST کلاسیک که متشکل از اعداد دستنویس است بوده و برای دسته‌بندی تصویر بسیار پرکاربرد است.

در قسمتی از کد، پس از ایمپورت کردن و لود دیتاست، دو نمونه از داده های هر کلاس برای درک بهتر، مصورسازی و نمایش داده شده اند:



۲. نرمال‌سازی داده‌ها

در ابتدا داده‌ها با استفاده از تابع fetch_openml از scikit-learn بارگذاری شدند.

سپس داده‌ها نرمال‌سازی شدند تا مقادیر پیکسل‌ها از بازه‌ی ۰ تا ۲۵۵ به بازه‌ی ۰ تا ۱ تبدیل شوند. این کار باعث افزایش سرعت همگرایی مدل و کاهش تأثیر مقیاس ویژگی‌ها می‌شود. سپس داده‌ها به نسبت ۸۰٪ آموزش و ۲۰٪ تست تقسیم شدند.

۳. آموزش مدل Logistic Regression

مدل LogisticRegression با حل‌کننده‌ی liblinear، آموزش داده شد. این مدل روی داده‌های نرمال‌شده آموزش دید و دقت آن روی هر دو بخش آموزش و تست محاسبه شد. دقت اولیه‌ی مدل در سطح قابل قبولی قرار داشت.

دقت آموزش: Train Accuracy: %81.19 و دقت تست: Test Accuracy: %80.66

۴. تحلیل دقت ها

اختلاف کم بین دقت آموزش و تست نشان میدهد مدل دچار overfitting نیست و توانایی تعمیم خوبی روی داده های دیده نشده دارد. دقت کلی با توجه به سادگی مدل و ساختار دادگان قابل قبول است.

۵. پارامترهای مدل رگرسیون لجستیک که در بالا مشخص شده اند و مقادیر ممکن برای هر پارامتر را معرفی کنید.

برای اختصار و خوانش بهتر، پارامتر ها به همراه توضیحات لازم در جدول زیر آورده شده اند:

پارامتر (نام)	مقدار در کد	نوع داده	مقادیر ممکن	توضیح فارسی
`solver`	`'liblinear'`	رشته (str)	`liblinear'`, `'lbfgs'`, `'newton- '` `' cg'`, `'sag'`, `'saga`	الگوریتم بهینه‌سازی برای حل مسأله یادگیری. `liblinear` برای داده‌های کوچک یا sparse مناسب است و فقط از `'penalty='l1` پشتیبانی می‌کند. یا `'l2'`
`C`	`0.001`	عدد مثبت (float)	هر عدد مثبت (مثلاً `0.001`, `0.01`, `1`, `10`, ...)	معکوس قدرت نظم‌دهی (regularization). مقدار کمتر باعث نظم‌دهی بیشتر و جلوگیری از بیش‌برازش می‌شود.
`multi_class`	`'auto'`	رشته (str)	`' auto'`, `'ovr'`, `'multinomial '`	استراتژی برای طبقه‌بندی چندکلاسه. `'auto'` به‌طور خودکار بهترین گزینه را بر اساس `solver` انتخاب می‌کند.
`random_state`	`0`	عدد صحیح (int)	عدد صحیح دلخواه (مثلاً `0`, `42`, ...) یا `None`	برای کنترل تصادفی بودن در آموزش مدل. مقدار ثابت باعث بازتولیدپذیری نتایج در اجراهای مختلف می‌شود.
`penalty`	`'l2'`	رشته (str)	`l1'`, `'l2'`, `'elasticnet'`, `''none'` (بسته به `solver`)	نوع جریمه نظم‌دهی. `'l2'` باعث کوچک شدن ضرایب می‌شود. برای `liblinear`، فقط `'l1'` و `'l2'` مجازند.
`max_iter`	`1000`	عدد صحیح (int)	هر عدد صحیح مثبت (مثلاً `100`, `500`, `1000`, ...)	حداکثر تعداد تکرارهای الگوریتم بهینه‌سازی برای رسیدن به همگرایی. اگر مدل همگرا نشد، این مقدار را افزایش دهید.

۶. برای پارامتر C مقادیر دیگری را در بخش ۴ پیاده سازی مورد بررسی قرار دادیم. کدام مقدار دقت بیشتری در حالت ارزیابی دارد؟ تحلیل کنید که چرا این مقدار از بقیه مقادیر بهتر عمل کرده است.

در طی اجرای Grid Search برای مدل Logistic Regression، پارامتر C روی چند مقدار تست شد. بر اساس نتایج به‌دست‌آمده:

- **بهترین مقدار C برابر 0.3** گزارش شده است.
- دقت متقابل (Cross-Validation Accuracy) برای این مقدار: **0.8525**

- دقت نهایی مدل روی داده تست: **0.8536**

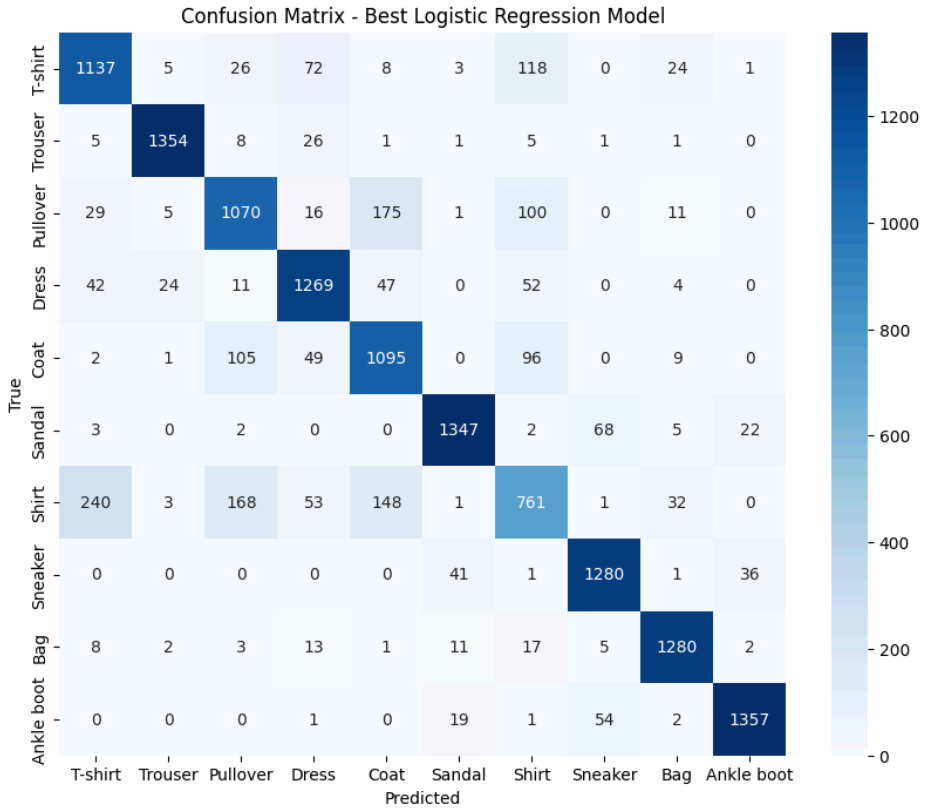
تحلیل:

پارامتر C معکوس شدت regularization است؛ بنابراین:

- **مقدار بسیار کوچک (مثلاً 0.001)** باعث نظم‌دهی شدید (strong regularization) می‌شود. این موضوع ممکن است منجر به **bias** بالا و **underfitting** شود.
- **مقدار بزرگ (مثلاً 10)** نظم‌دهی را ضعیف می‌کند که می‌تواند به **variance** بالا و **overfitting** منجر شود.

مقدار **0.3** به‌نظر می‌رسد **تعادل مناسبی بین bias** و **variance** برقرار کرده و اجازه داده مدل هم پیچیدگی لازم را حفظ کند و هم تعمیم‌پذیر بماند

۷.ماتریس درهم ریختگی (Matrix Confusion)پیش بینی مدل در بخش ۵ پیاده سازی را تحلیل نمایید.



ماتریس درهم‌ریختگی نشان می‌دهد مدل در **طبقه‌بندی برخی کلاس‌ها عملکرد بسیار خوب** و در برخی دیگر **ضعف نسبی** دارد. کلاس‌هایی مثل **Sandal، Sneaker** و **Ankle boot** با دقت بسیار بالا (بیش از 1300 مورد صحیح از 1400) پیش‌بینی شده‌اند. کلاس **Trouser** نیز با دقت بالا تشخیص داده شده است. بیشترین خطا در تشخیص **Shirt** دیده می‌شود: Shirt با **T-shirt (240 مورد**، **Coat (148** ،**Pullover (168)** اشتباه گرفته شده است. علت اصلی: **شباهت ظاهری** در ویژگی‌های تصویری این کلاس‌ها و **محدودیت مدل‌های خطی** در تفکیک فضاهای پیچیده ویژگی. این نشان می‌دهد که مدل Logistic Regression در تفکیک کلاس‌هایی با ویژگی‌های هم‌پوشان، **توانمندی محدودی** دارد.

۸. دقت آموزش و تست مدل SGDClassifier در بخش ۶ پیاده سازی را گزارش نمایید.

مدل `SGDClassifier` نتایج زیر را تولید کرده است:

- **Train Accuracy:** 0.8486
- **Test Accuracy:** 0.8369

اختلاف جزئی بین دقت آموزش و تست نشان می‌دهد که مدل به‌خوبی تعمیم یافته و **نه overfit شده** و **نه underfit**. مدل با داده‌های بزرگ سازگار است و در زمان کم آموزش می‌بیند، اما دقت آن کمی کمتر از مدل بهینه Logistic Regression است.

9. کدام مدل دسته بندی بهتری انجام داده است؟ علت آن چیست؟

- مدل **Logistic Regression** با **C=0.3** عملکرد بهتری دارد.
- علت اصلی:
  1. حل دقیق‌تر مسأله بهینه‌سازی با `liblinear`
  2. انتخاب بهینه پارامتر با **Grid Search**
  3. مدل دارای regularization مناسب و حل همگرا

در حالی که **SGDClassifier** سریع‌تر آموزش می‌بیند، اما به علت طبیعت تصادفی (stochastic)، نیازمند تنظیم دقیق نرخ یادگیری و epochها است و دقت نهایی کمی کمتر است.

پیشنهاد:

میتوانیم برای دستیابی به نتایج بهتر، از شبکه های عصبی کانولوشن استفاده کنیم چرا که با عکس و **spatial information** سروکار داریم و برداری کردن، اطلاعات همسایگی پیکسل ها را از بین میبرد. همینطور میتوان با عکس هایی با ابعاد بزرگتر و همینطور دارای 3 کانال **RGB** استفاده کنیم.