به نام خدا

گزارش فاز دوم

پروژه هوش محاسباتی

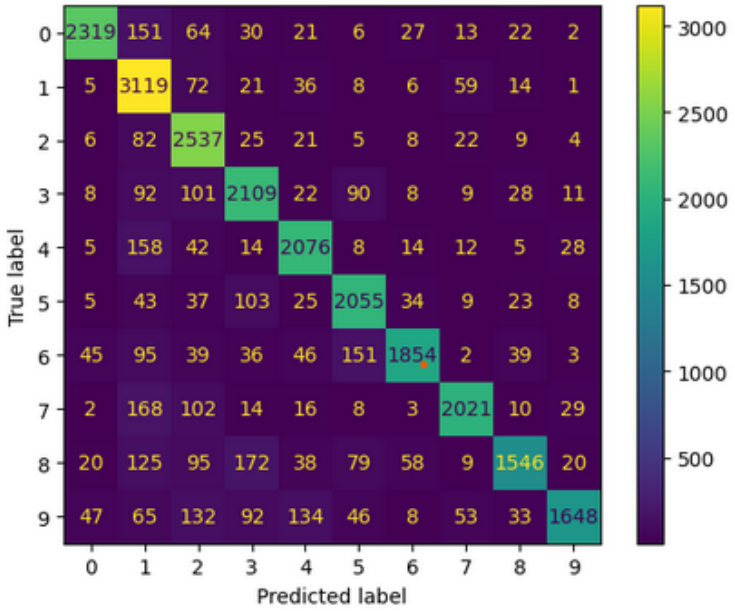
پوریا ناظمی و طهورا سعیدی

بخش اول

در بخش اول به دنبال این هستیم که با مدل‌سازی مناسب بدون توجه به domainها داده‌ها را بر حسب عددی که در بر دارند طبقه‌بندی کنیم. به این منظور باید با توجه به صورت پروژه پارامترهای مناسب مدل random forest را با توجه به داده‌ها پیدا کنیم. به این منظور برای hyper tuning پارامترهای مدل از grid search استفاده کردیم و برای دو پارامتر اصلی n estimators و max depth مجموعه‌ای از مقادیر را در نظر گرفتیم و با تابع grid search cv با cv=3 بهترین ترکیب این مقادیر را پیدا کردیم. max depth برابر 10 و n estimators برابر 200 شده است. با این دو پارامتر مدل را بر روی داده‌ها fit کردیم. نتایج حاصل بر روی داده‌های تست به شرح زیر است:

```
cm_digits = confusion_matrix(digits_test, predicted_number)
ConfusionMatrixDisplay(confusion_matrix=cm_digits, display_labels=digit_classifier.classes_).plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f8fc19e7610>
```




```
recall_score(digits_test, predicted_number, average="micro")
```

```
0.85136
```

recall score به دست آمده از این طبقه‌بندی 0.85 می‌باشد و ماتریس گمراهی نیز ترسیم شده است.

بخش دوم

در بخش دوم ابتدا باید داده‌ها را بر حسب domain طبقه‌بندی کنیم. مشابه بخش قبل از همان روش برای hyper tuning استفاده می‌کنیم. پارامترهای به دست آمده مطلوب برای این مدل‌سازی به شرح زیر است:

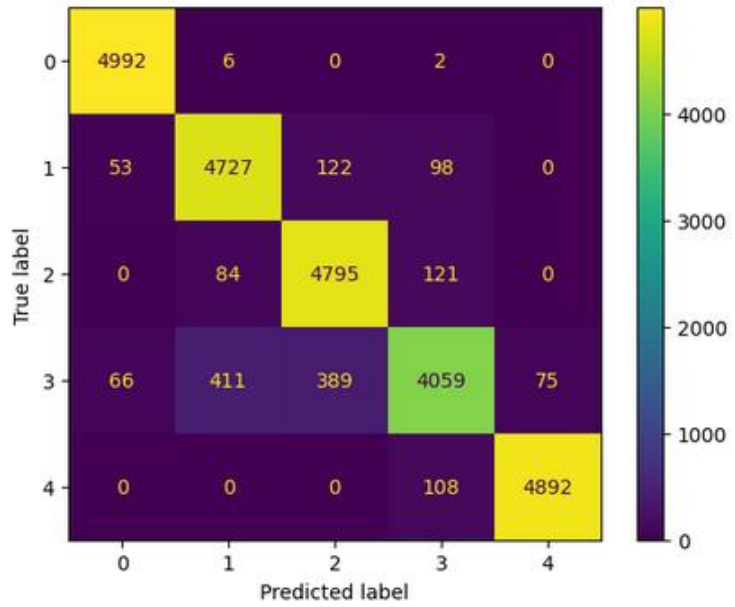
n estimator = 100

max depth = 20

مدل را با این پارامترها می‌سازیم و بر روی داده‌های train آن را fit می‌کنیم. ارزیابی انجام شده بر روی داده‌های تست به شرح زیر است:

```
[17] cm_domains = confusion_matrix(domains_test, predicted_domain)
     ConfusionMatrixDisplay(confusion_matrix=cm_domains, display_labels=domain_classifier.classes_).plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f8fb28b21d0>
```




```
[18] recall_score(domains_test, predicted_domain, average="micro")
```

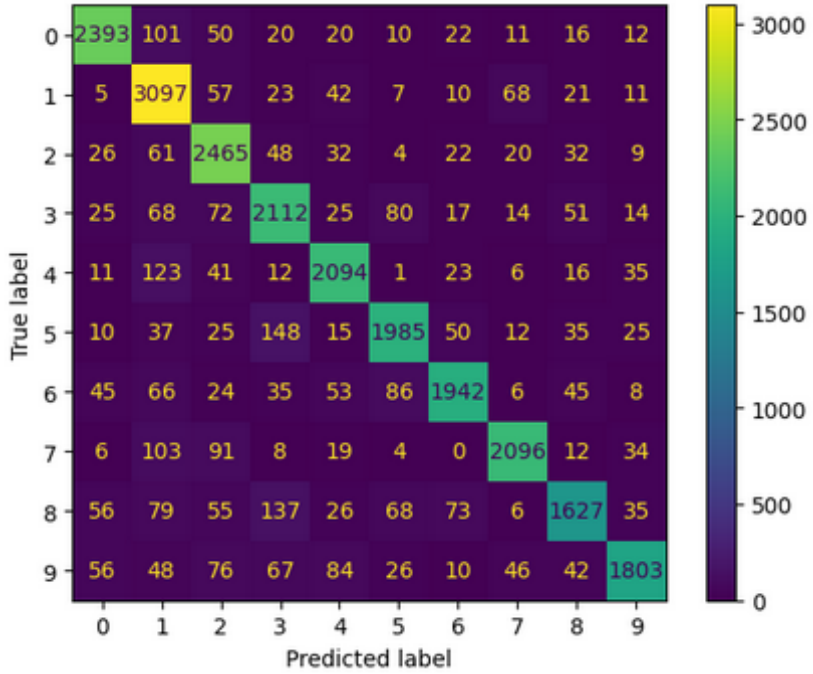
```
0.9386
```

حال باید مدل‌سازی طبقه‌بندی اعداد را برای داده‌های هر domain به طور جداگانه انجام دهیم؛ پس بر روی داده‌های هر domain مدل random forest را به منظور طبقه‌بندی اعداد با پارامترهای مناسب اعمال می‌کنیم. اکنون با توجه به وجود ۵ نوع domain مختلف، ۵ مدل مختلف داریم. داده تستی که وارد می‌شود باید ابتدا توسط مدل اول تعیین domain شود. سپس داده تست با توجه به مدل train شده برای تشخیص اعداد برای آن domain بر اساس عدد طبقه بندی شود. نتیجه ارزیابی حاصل از این بخش نیز به شرح زیر است:

```
cm_pipe_digits = confusion_matrix(digits_test, pipe_df["digit"])
ConfusionMatrixDisplay(confusion_matrix= cm_pipe_digits, display_labels=model.classes_).plot()

print(recall_score(digits_test, pipe_df["digit"], average="micro"))
```

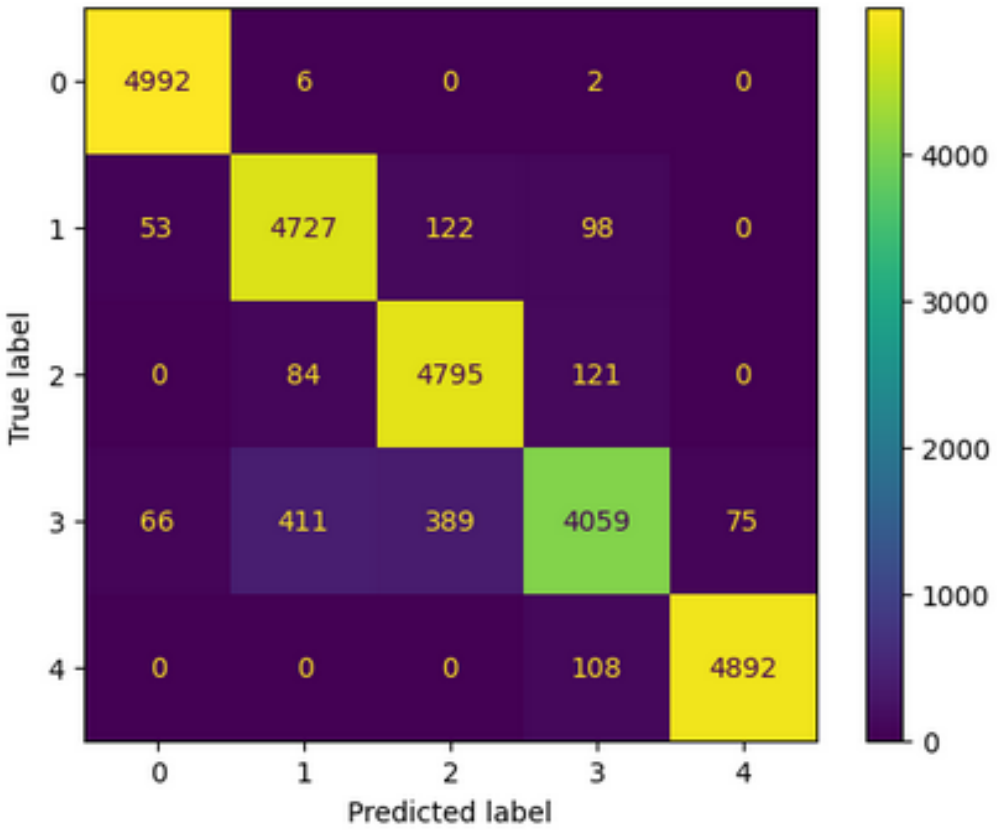
```
0.86456
```



همان طور که مشاهده می‌شود recall score به دست آمده برای این بخش 0.8645 است. این مورد نشان می‌دهد این رویکرد (ابتدا طبقه‌بندی domain و سپس دادن داده به مدل خاص همان domain) می‌تواند نتایج ارزیابی ما را بهبود بخشد.

بخش سوم

در بخش سوم بار دیگر ماتریس گمراهی طبقه‌بندی domainها را مشاهده می‌کنیم.



همان طور که مشاهده می‌شود در تعدادی از داده‌ها تشخیص domain اشتباه بوده است. و این مورد روی تشخیص درست طبقه‌بندی اعداد هر domain نیز تاثیر می‌گذارد. نکته‌ای که وجود دارد این است که داده‌هایی از یک domain که به اشتباه در domain دیگر قرار گرفته‌اند، تا حدودی به یکدیگر شبیه بوده‌اند.

بنابراین با توجه به موارد گفته شده و صورت پروژه، تعدادی از داده‌هایی را که از دیگر domainها در یک domain قرار گرفته‌اند، با نسبتی به این domain اضافه می‌کنیم. اما نکته‌ای که وجود دارد این است که چه نسبتی از داده‌های دیگر را به domainای بیاوریم که بخشی از داده‌های آن در domainهای دیگر قرار گرفته است. برای مثال با توجه به ماتریس گمراهی بالا از داده‌های domain شماره ۲ تعداد ۸۴ عدد در domain شماره ۱ و تعداد ۱۲۱ عدد در domain شماره ۳ قرار گرفته‌اند. برای این که مدل tarin شده بر روی داده‌های domain بتواند داده‌های متعلق به طبقه‌های دیگر که شباهت به داده‌های همان domain داشته‌اند را نیز به درستی از جهت عدد طبقه بندی کند باید این داده‌ها را نیز ببیند. پس درصدی از domainهای دیگر که بخشی از داده‌های آن‌ها شباهت داشته است را به این domain می‌آوریم. نسبتی که ما در نظر گرفته‌ایم به این شکل است که تعداد کل داده‌های در domain را حدود ۵۰۰۰ عدد فرض کرده‌ایم. برای مثال اگر ۴۵ داده از یک domain به اشتباه در کلاس دیگری قرار گرفته است، ۴۵/۵۰۰۰ داده‌های domain دیگر را به این domain می‌آوریم.

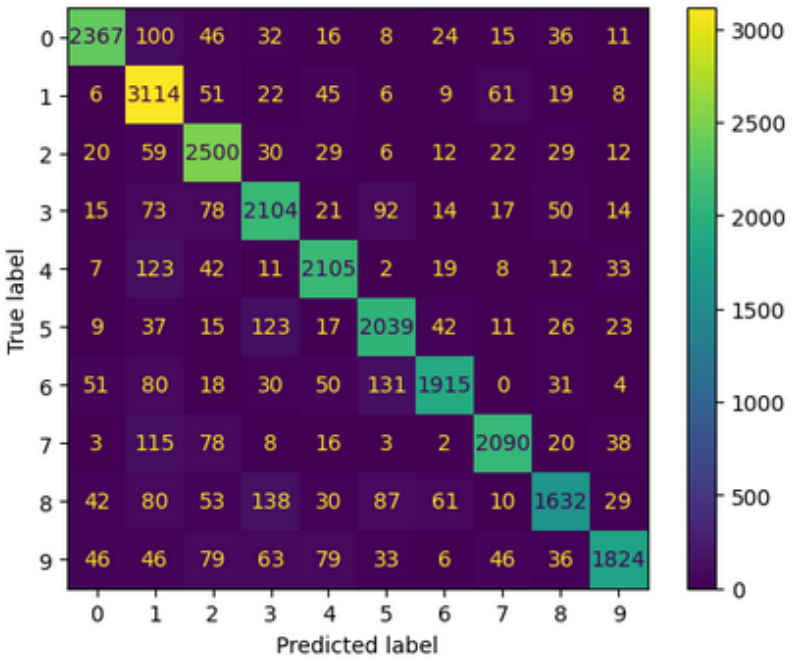
در نهایت data frameای تشکیل می‌شود که در هر domain آن بخشی از داده‌های domainهای دیگر نیز قرار گرفته است. دقت شود که میزان داده‌های اضافه شده از هر domain با توجه به نسبت گفته شده تعیین شده است و ممکن است برای هر domain متفاوت باشد. اکنون که data frame را تشکیل داده‌ایم با داده‌های هر domain که ممکن است در برگیرنده بخشی از داده‌های domainهای دیگر نیز باشد، مدل‌سازی طبقه‌بندی اعداد را با پارامترهای مناسب انجام می‌دهیم. اکنون برای تشخیص اعداد، ۵ مدل جدید داریم.

در نهایت با مدل‌های جدید، داده‌های تست را به منظور ارزیابی رویکرد به کار گرفته شده، به مدل می‌دهیم. نتایج به شرح زیر است:

```
cm_pipe_digits = confusion_matrix(digits_test, pipe_df["digit"])
ConfusionMatrixDisplay(confusion_matrix= cm_pipe_digits, display_labels=model.classes_).plot()

print(recall_score(digits_test, pipe_df["digit"], average="micro"))
```

```
0.8676
```



همان طور که مشاهده می‌شود مقدار recall score نسبت به حالت قبل کمی افزایش یافته است.

جمع بندی نهایی:

در ابتدا بدون توجه به domain سعی کردیم داده‌ها را بر اساس اعداد طبقه‌بندی کنیم. در این حالت تاثیر domain لحاظ نشده بود و از این رو امتیاز به دست آمده، مقداری کمتر بود. در گام بعد ابتدا داده‌ها را بر اساس domain جدا کردیم و برای هر domain مدل جداگانه‌ای برای اعداد در نظر گرفتیم. همان طور که مشاهده شد نتیجه مقداری بهبود یافت. در گام آخر نکته‌ای که وجود داشت این بود که ممکن بود در بخش تعیین domain خطاهای موجود باعث شود مدل‌سازی اعداد نیز تحت تاثیر قرار گرفته و روی مقدار امتیاز به دست آمده موثر باشد. به این منظور به هر domain بخشی از داده‌های domain دیگر را به نسبتی که پیشتر به آن اشاره شد، اضافه کردیم. نتایج حاصل نشان داد که با این روش، نتیجه طبقه‌بندی اعداد مقداری بهتر از حالت قبل شده است.