به نام خدا

گزارش پروژه درس جدا سازی کور منابع

استاد درس: آقای دکتر اخوان بهابادی

دانشجو : امیرحسین چمی

فهرست

[بررسی اولیه داده ها 3](#_Toc202840910)

[پیش‌ پردازش داده ها 6](#_Toc202840911)

[1. انتخاب کانال ها 6](#_Toc202840912)

[2. بالانس کردن تعداد trial ها 8](#_Toc202840913)

[3. فیلتر کردن سیگنال ها 8](#_Toc202840914)

[4. رفع مقادیر nan و inf 9](#_Toc202840915)

[الگوی فضایی مشترک (CSP) 10](#_Toc202840916)

[الگوریتم LDA 11](#_Toc202840917)

[الگوریتم grid search 12](#_Toc202840918)

[رویکرد طبقه بندی 12](#_Toc202840919)

[1. پیاده سازی تابع طبقه بندی با LDA 12](#_Toc202840920)

[2. پیاده سازی تابع پیش بینی 13](#_Toc202840921)

[رویکرد های آموزش مدل 14](#_Toc202840922)

[1. آموزش یک مدل برای هر subject 14](#_Toc202840923)

[نتایج آموزش 16](#_Toc202840924)

[2. آموزش یک مدل برای تمام subject ها 24](#_Toc202840925)

[نتایج آموزش 26](#_Toc202840926)

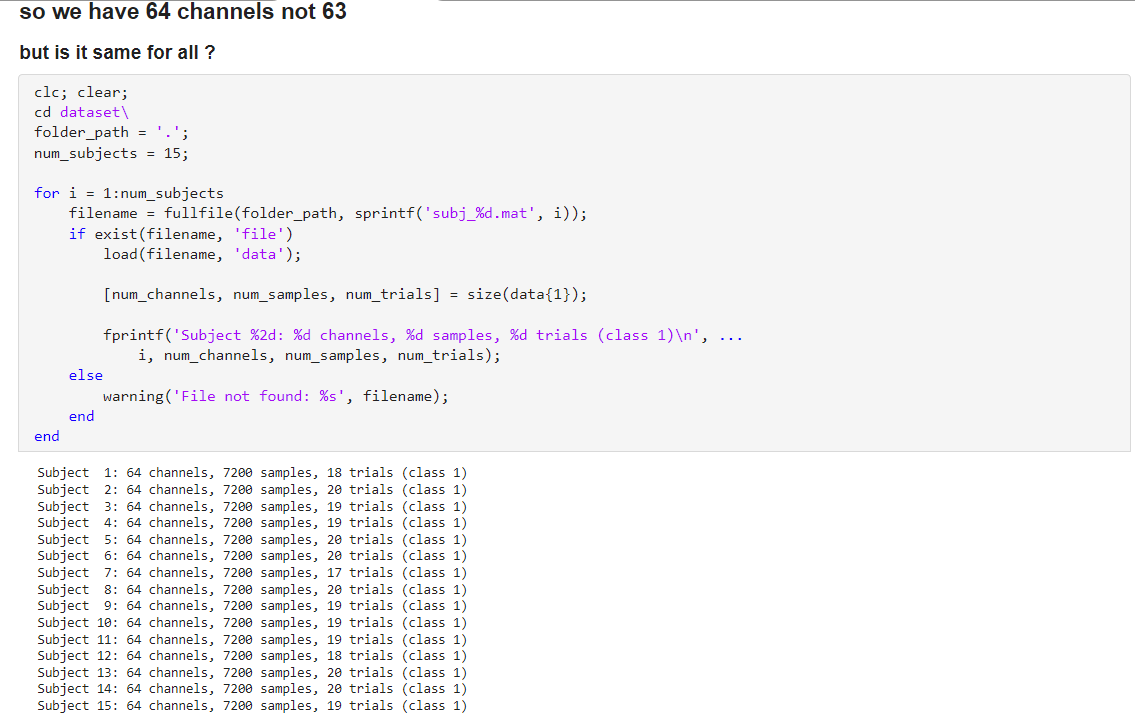
[مراجع 27](#_Toc202840927)

# بررسی اولیه داده ها

در ابتدا در فایل data\_review.mlx تمام آنچه در مورد ویژگی های دیتاست در فایل recording.pdf گفته شده بود را مورد بررسی قرار دادم. این داد ها را استثنائا در بخش live بررسی کردم زیرا می خواستم خروجی کد را در کنار آن داشته باشم. بقیه ی فایل ها به صورت .m خواهند بود.

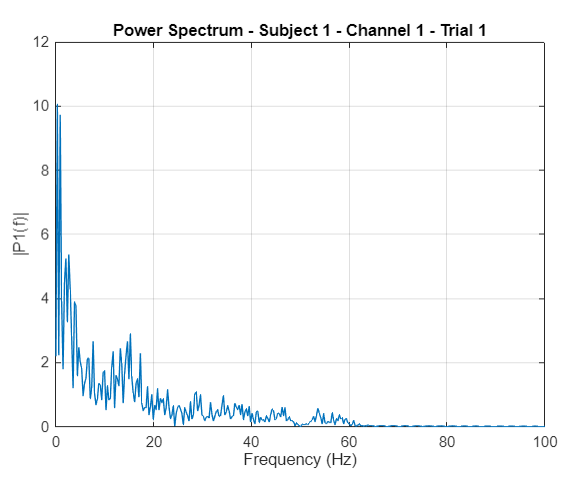


در این بخش داده های نفر اول یعنی subj\_1.m را load کرده و ساختار آن که یک سلول 4\*1 است را بررسی می کنم. علی رغم اینکه فایل Recording.pdf گفته، ما 64 کانال داریم یعنی کانال مرجع از آن حذف نشده است. حال باید بررسی کنی که آیا این مورد برای تمام subject ها برقرار است یا خیر.

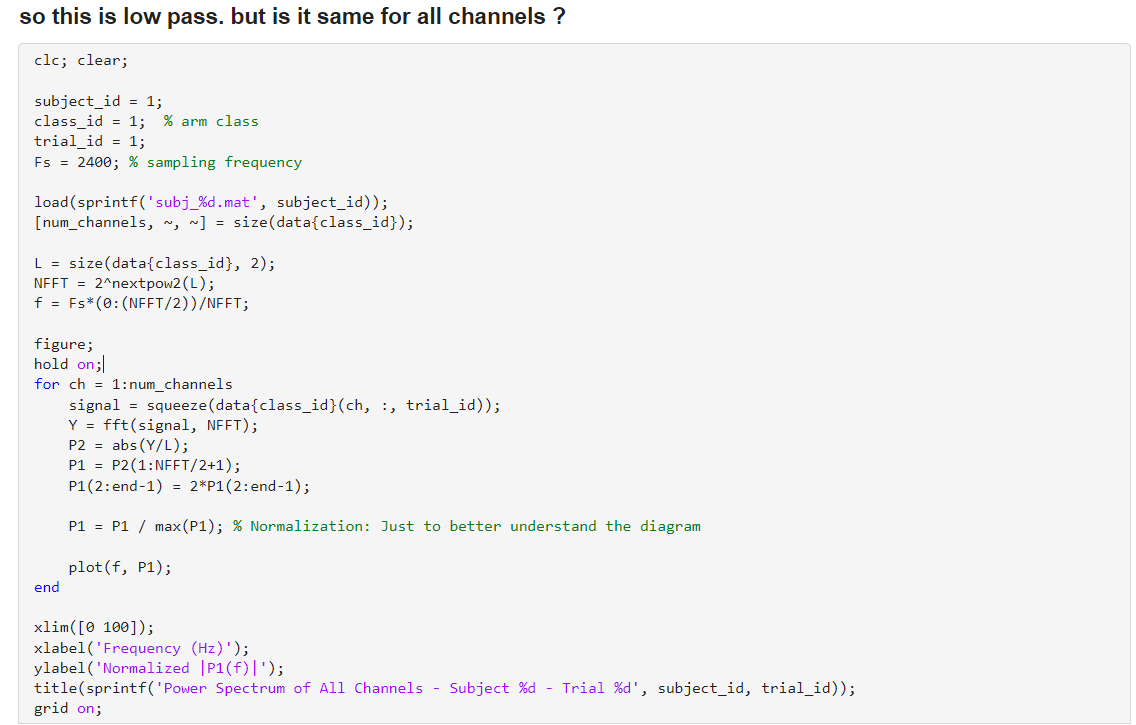


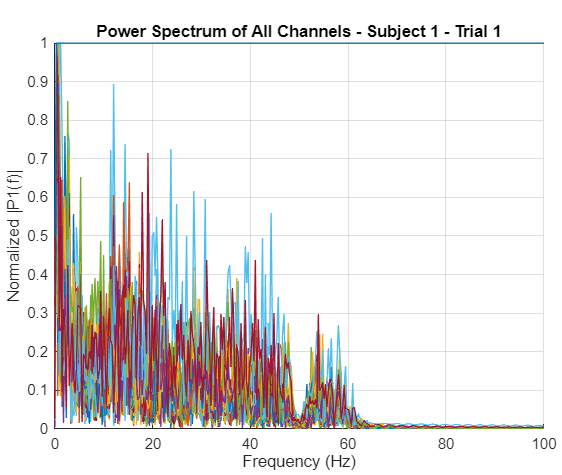
حال می خواهم طیف فرکانسی داده ها را بررسی کنم.





پس طیف داده ی trial اول از کانال اول و subj\_1 پایین گذر است. حال ببینیم این برای تمام کانال ها برقرار است؟





پس تمام کانال ها پایین گذر هستند.

در این جا کار من با فایل data\_review.mlx به پایان می رسد. موارد مهمی که فهمیدم به شرح زیر است:

* ساختار داده ها
* تعداد کانال ها
* پایین گذر بودن طیف فرکانسی

# پیش‌ پردازش داده ها

## انتخاب کانال ها

تنها مورد متفاوت با ادعای فایل Recording.pdf تعداد کانال هاست که کانال مرجع از آن حذف نشده است. در کل دو رویکرد برای این مورد وجود دارد:

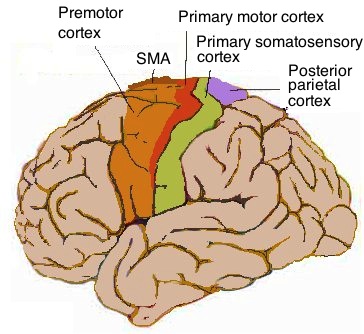
* رویکرد اول اینکه یکی از کانال ها را به عنوان مرجع انتخاب کرده و آن را از بین کانال ها حذف کنیم. انتخاب این کانال مرجع می تواند با معیار های مختلفی مثل میانگین دامنه ی آن کانال باشد.
* رویکرد دوم بر اساس یک دیدگاه متفاوت است؛ در چنین آزمایشی تمام کانال ها الزاما کاربردی و مفید نیستند پس می توانیم تنها کانال های مفید را انتخاب کرده و بقیه ی کانال ها را کنار بگذاریم. در این صورت احتمالا تعداد کانال های انتخابی از 64 تا خیلی کمتر است.

من رویکرد دوم را انتخاب می کنم. این رویکرد به دو روش قابل پیاده سازی است.

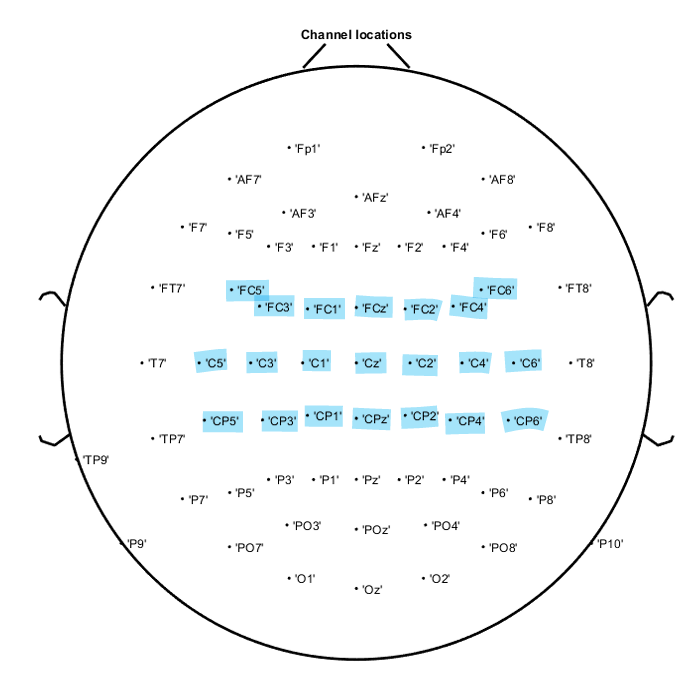
* روش اول استفاده از CSP است. با اعمال CSP روی داده ها، یک بردار از وزن ها به دست خواهیم آورد. بعضی از این وزن ها مقدار بیشتری دارند که این مقدار بیشتر نشان دهنده ی این است که کانال متناظر با آن مقدار، بخش مهم تری از مغز را شامل می شود. پس ما کانال های متناظر با این وزن ها را انتخاب می کنیم.

واضحا برای پیاده سازی این روش از داده های کلاس چهارم نباید استفاده کنیم زیرا در کلاس چهارم داده های سکون اشخاص ذخیره شده اند در نتیجه بخش هایی که برای ما مهم هستند احتمالا در این کلاس تفاوت خاصی با بقیه ی بخش های مغز نخواهند داشت.

* روش دوم با توجه به ساختار مغز است. پس ابتدا نیاز است ساختار مغز را بشناسیم. در تصویر زیر بخش هایی که نام گذاری شده اند، همگی مسئول فرآیند حرکت اندام های بدن هستند. از تصمیم به حرکت تا حرکت اندام ها.[1]



پس من هم کانال هایی که به نظر متناظر با همین بخش ها هستند را انتخاب می کنم. در تصویر زیر این کانال ها مشخص هستند.



شماره ی کانال های انتخاب شده به شرح زیر است.

**11, 40, 12, 41, 13, 42, 14, 44, 16, 45, 17, 46, 18, 47, 20, 49, 21, 50, 22, 51, 23**

## بالانس کردن تعداد trial ها

از آنجا که در بررسی دیتا دیدم، تعداد trial ها برای کلاس های مختلف و برای افراد مختلف متفاوت است. به همین دلیل برای این که از سوگیری مدل در مرحله ی آموزش جلوگیری کنم، تعداد trial ها را در تمام کلاس ها یکسان و برابر با کمترین تعداد برای هر شخص می کنم.

## فیلتر کردن سیگنال ها

در بررسی اولیه در فایل data\_review.mlx دیدم که سیگنال ها همگی پایین گذر بوده و طیف آن ها تا حدود 60 هرتز را شامل می شود. به علت شرایط ضبط سیگنال های EEG مثل فرکانس برق شهر و یا عملکرد بخش های دیگر مغز، تمام این طیف فرکانسی شامل داده های مورد نظر ما نیست. پس باید آن را فیلتر کنیم. دو راه برای فیلتر کردن وجود دارد.

* رویکرد اول: استفاده از یک طیف فرکانسی ثابت برای تمامی اشخاص. از آنجا که داده های ما سیگنال های EEG هستند، و هدف ما تشخیص حرکت های مختلف شخص است، می توانیم با شناخت دقیق تر مغز فرکانس هایی را انتخاب کنیم که برای هدف ما مناسب هستند. مغز انواع مختلفی از موج را تولید می کند که دلتا[2]، تتا[3]، آلفا[4] و بتا[5] چهار موج اصلی هستند.

هر کدام از این سیگنال ها نشان دهنده یک حالت خاص هستند.

**دلتا:** دلتا سیگنال خواب عمیق است. شامل فرکانس های 1 الی 4 هرتز می شود.

**تتا:** سیگنال حالت آرامش یا خواب سبک است. شامل فرکانس های 4 الی 7 هرتز است.

**آلفا:** سیگنال حالت استراحت یا چشم بسته است. شامل فرکانس های 8 الی 12 هرتز است.

**بتا:** سیگنال فعالیت مغزی بالا و تمرکز است. شامل فرکانس های 12.5 الی 30 هرتز است.

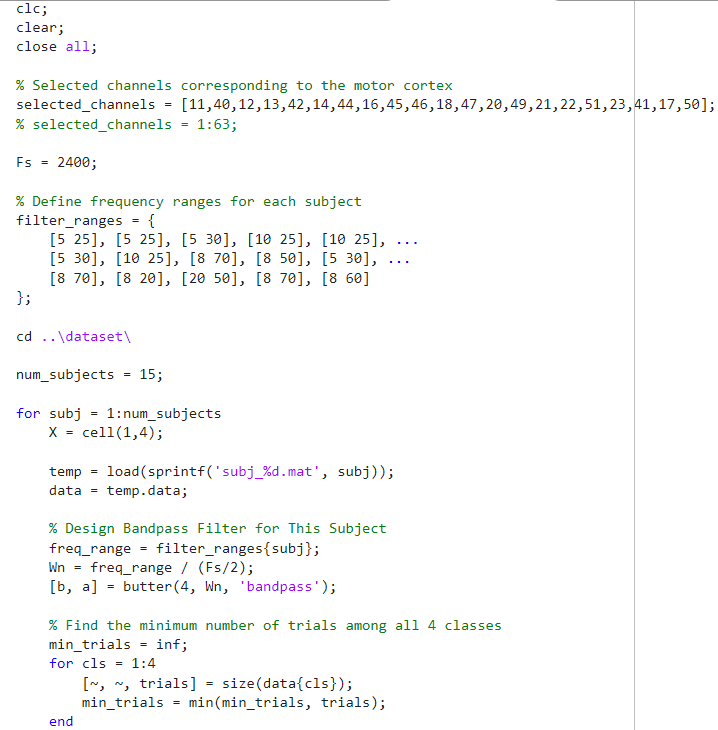
پس با توجه به چهار کلاس مورد نظر ما، بهتر است امواج آلفا و بتا را در نظر بگیریم یعنی بازه ی فرکانسی 8 الی 30 هرتز. برای اینکه از اعوجاج هم جلوگیری کنیم می توانیم کمی این بازه را بزرگ تر درنظر بگیریم. برای مثال 7 الی 35 هرتز.

* رویکرد دوم: استفاده از پهنای باند جداگانه برای هر شخص. در این روش پهنای باند برای هر شخص جداگانه تعیین شده و بر اساس این پهنای باند فیلتر طراحی می شود.

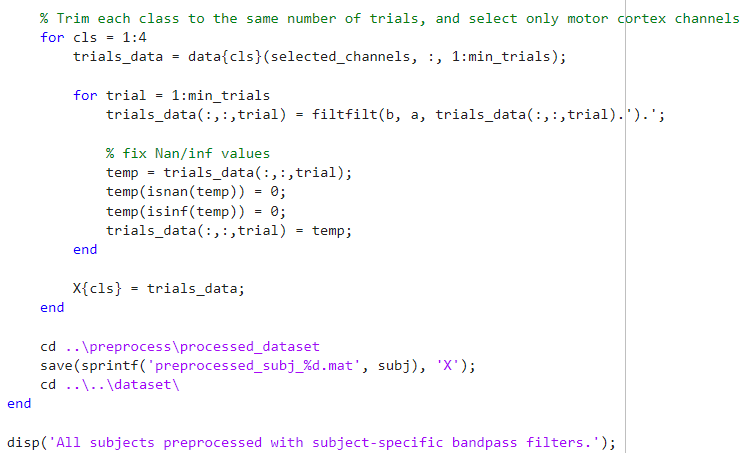
## رفع مقادیر nan و inf

یکی از مواردی که می تواند در اثر شرایط ضبط سیگنال ها رخ دهد، وجود داده های Nan یا inf است. می توانیم آن ها را صفر کنیم.

تمامی موارد مربوط به پیش پردازش که گفته شد در فایل preprocessing.m در پوشه ی preprocess پیاده سازی می شود.



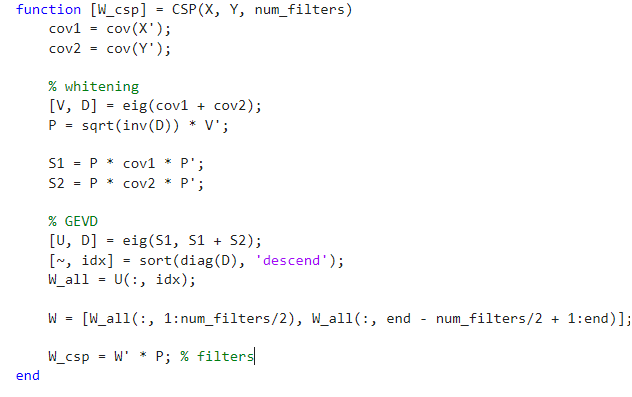
به علت طولانی بودن کد، ادامه ی آن در صفحه ی بعد.



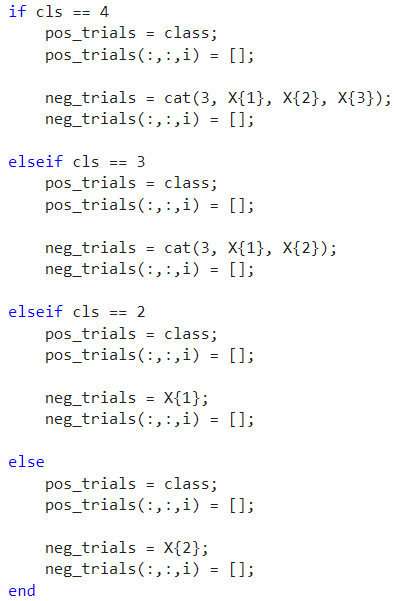
در این کد داده های هر شخص لود می شود و پس از انجام مراحل پیش پردازش ذکر شده، در پوشه ی processed\_dataset ذخیره می شود.

# الگوی فضایی مشترک (CSP)

الگوی فضایی مشترک (CSP) یک تکنیک فیلتر فضایی است که هدف آن یافتن مجموعه ای از فیلتر های فضایی است که بتواند به طور موثر بین دو کلاس سیگنال بر اساس ماتریس های کواریانس آن ها تمایز قائل شود. برای پیاده سازی آن تابع CSP را در فایل CSP.m در پوشه ی train and evaluate نوشته ام.



جلو تر خواهیم دید که رویکرد آموزش مدل را رویکرد، طبقه بندی cascade انتخاب می کنم. به همین دلیل باید داده هایی که برای آموزش مدل استفاده می شوند هم بر همین اساس تفکیک و ویژگی آن ها استخراج شود. بر همین اساس کد زیر در کد آموزش اصلی یعنی train\_and\_evaluate.m نوشته شده.



در هر مرحله با توجه به کلاسی که در آن هستیم داده ها را به دو دسته تقسیم کرده و آن ها را به تابع CSP دهیم. این کد را در بخش آموزش مدل خواهیم دید.

# الگوریتم LDA

این الگوریتم در واقع ابزاری است که برای طبقه بندی داده ها استفاده می شود. هدف آن یافتن یک ترکیب خطی از ویژگی هاست که بیشترین تماین را بین کلاس های مختلف قائل شود. می توان آن را مستقیما نوشت و یا از دستور fitcdiscr() استفاده کرد.

## الگوریتم grid search

الگوریتم [6]Grid Search یک روش جستجوی فراگیر برای پیدا کردن بهترین ترکیب از ابرپارامترها در مدل ‌های یادگیری ماشین است. در این روش، با تعریف یک شبکه (grid) از مقادیر ممکن برای هر پارامتر، همه ‌ی ترکیب ‌های ممکن بررسی می ‌شوند و عملکرد مدل برای هر ترکیب با استفاده از یک معیار ارزیابی (مثل دقت یا میانگین خطا) سنجیده می ‌شود.

برای پیاده سازی آن در پوشه ی grid search فایلی با همین نام نوشتم و الگوریتم آن را پیاده سازی کردم. تنها تفاوت کد آن با کد اصلی که در بخش آموزش مدل -رویکرد اول- خواهیم دید، این است که تعداد فیلتر ها در این کد یک range است که برای هر عدد در این بازه یکبار مدل برای هر شخص اموزش داده می شود و دقت آن اندازه گیری می شود. به دلیل اینکه کد طولانی و تکراری است، تصویر آن را قرار ندادم. برای بررسی دقیق تر به فایل آن مراجعه فرمایید.

طبق این الگوریتم تعداد فیلتر های مکانی برای هر subject به صورت زیر انتخاب شد.

Subj1: 14, subj2: 8, subj3: 14, subj4: 6, subj5: 14, subj6: 10, subj7: 14, subj8: 10 , subj9: 14, subj10: 8, subj11: 14, subj12: 12, subj13: 10, subj14: 12, subj15: 6

من یکبار با تعداد فیلتر های ثابت مدل ها را آموزش دادم و دقت های بدست آمده خوب بود. اما به نظرم راضی کننده نبود برای همین از الگوریتم grid search استفاده می کنم.

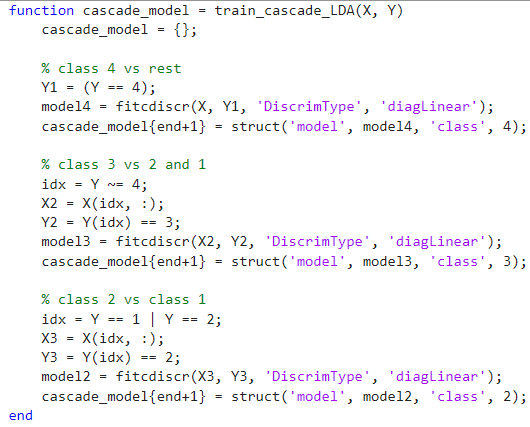
# رویکرد طبقه بندی

## پیاده سازی تابع طبقه بندی با LDA

از بین روش های ممکن برای طبقه بندی چهار کلاس مختلف که در کلاس به آن اشاره شد، من از روش پیشنهادی فایل Report.pdf استفاده می کنم. این رویکرد طبقه بندی به صورت آبشاری(cascade) است.[7]

در این روش ابتدا کلاسی که می دانیم تشخیص آن نسبت به بقیه ساده تر است را انتخاب می کنیم. سپس طبقه بند اول را برای تشخیص بین این کلاس (در اینجا کلاس چهارم) و سه کلاس دیگر آموزش می دهیم. و همین فرآیند را برای سه کلاس دیگر تکرار می کنیم.( اطلاعات کافی در کلاس در این مورد ارائه شد. برای آشنایی بیشتر به مرجع شماره 6 مراجعه نمایید.)

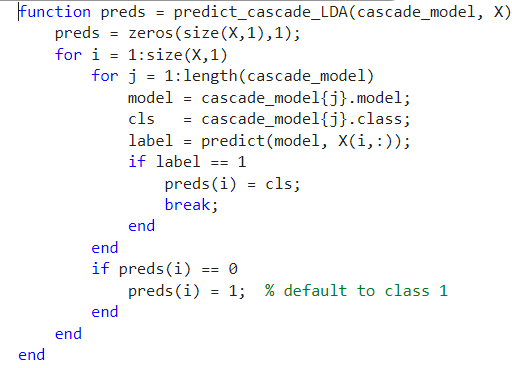
تابع آن را در فایل train\_cascade\_LDA.m در پوشه ی train and evaluate پیاده سازی کردم.



## پیاده سازی تابع پیش بینی

از آنجا که از طبقه بندی آبشاری استفاده شد، باید تابع پیش بینی هم مطابق با آن باشد. در این پیش بینی هر داده ابتدا با استفاده از یک مدل آموزش داده، پیش بینی می شود. اگر پیش بینی آن مدل 1 منطقی بود پس به آن کلاس مربوط است و اگر 0 منطقی بود به کلاس بعدی می رویم و این روال را تا طبقه بند سوم ادامه می دهیم و اگر داده ای در پیش بینی طبقه بند سوم هم 0 منطقی بود پس طبق فرض باید آن داده برای کلاس 1 باشد.

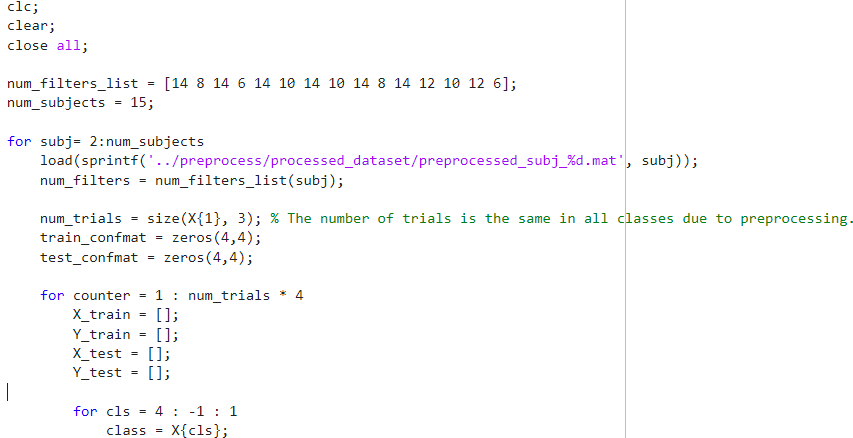
تابع آن را در فایل predict\_cascade\_LDA.m و در پوشه ی train and evaluate نوشتم.



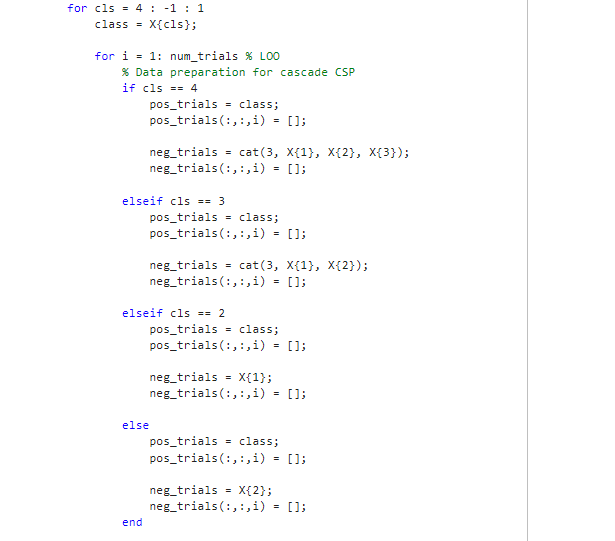
# رویکرد های آموزش مدل

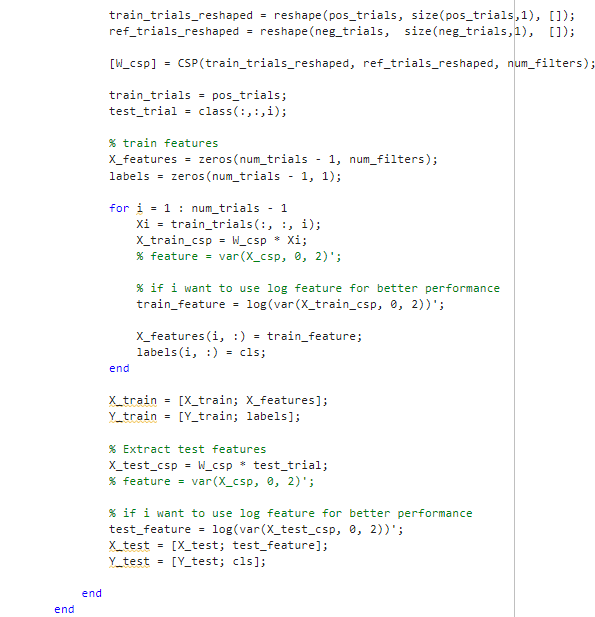
## آموزش یک مدل برای هر subject

مطابق کدی که در فایل train\_one\_model\_per\_subj.m در پوشه ی train and evaluate نوشتم، داده های هر شخص لود شده و پس از اعمال روش Leave One Out آموزش مدل شروع شده و در انتها مدل آموزش دیده ارزیابی شده، سپس مدل آموزش دیده برای هر شخص در مسیر models/one model per subj ذخیره می شود و ماتریس کانفیوژن اموزش و تست در یک fig رسم شده و این ماتریس های کانفیوژن با دو فرمت .jpg و .fig در مسیر confusion matrix/cm model per subj ذخیره می شود.

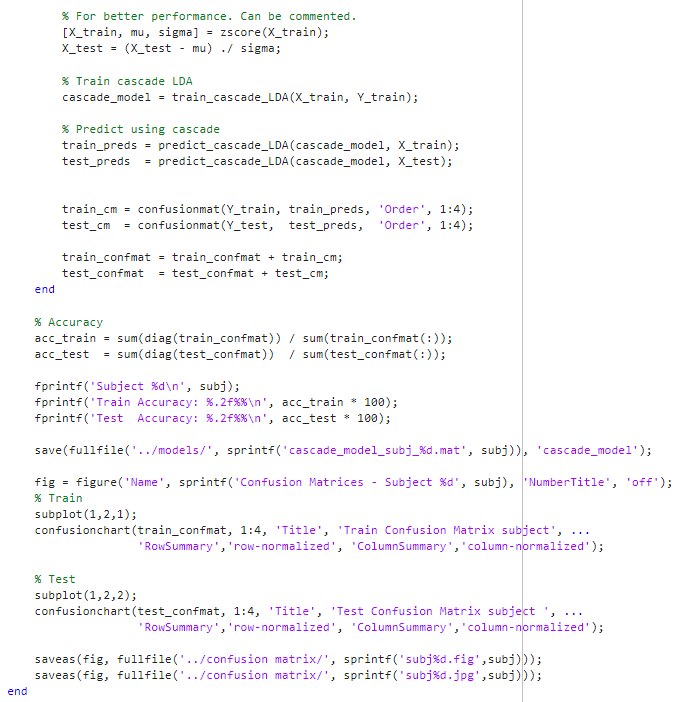


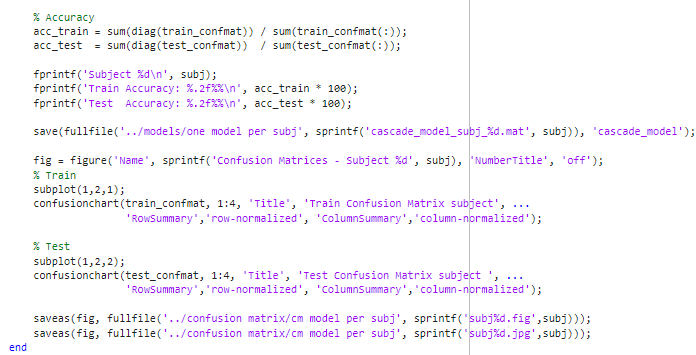
ادامه کد در تصویر بعدی.





در اینجا الگوریتم LOO و CSP برای جداسازی داده ها و استخراج ویژگی های آن ها به پایان می رسد. ادامه ی کد آموزش مدل و ارزیابی آن است.





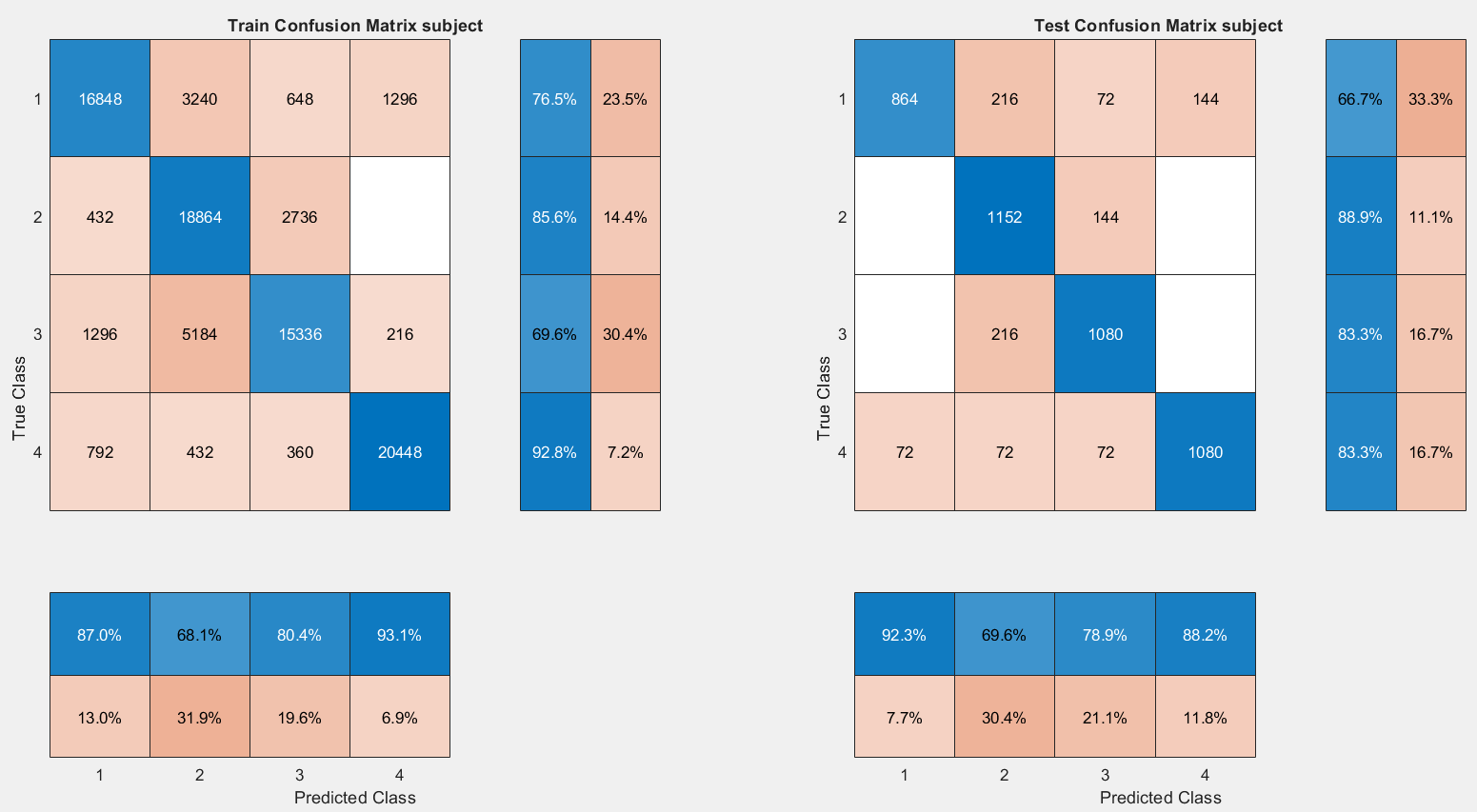
## نتایج آموزش

ماتریس کانفیوژن اموزش و تست هر شخص به صورت زیر است.

Subject 1

Train Accuracy: 81.13%

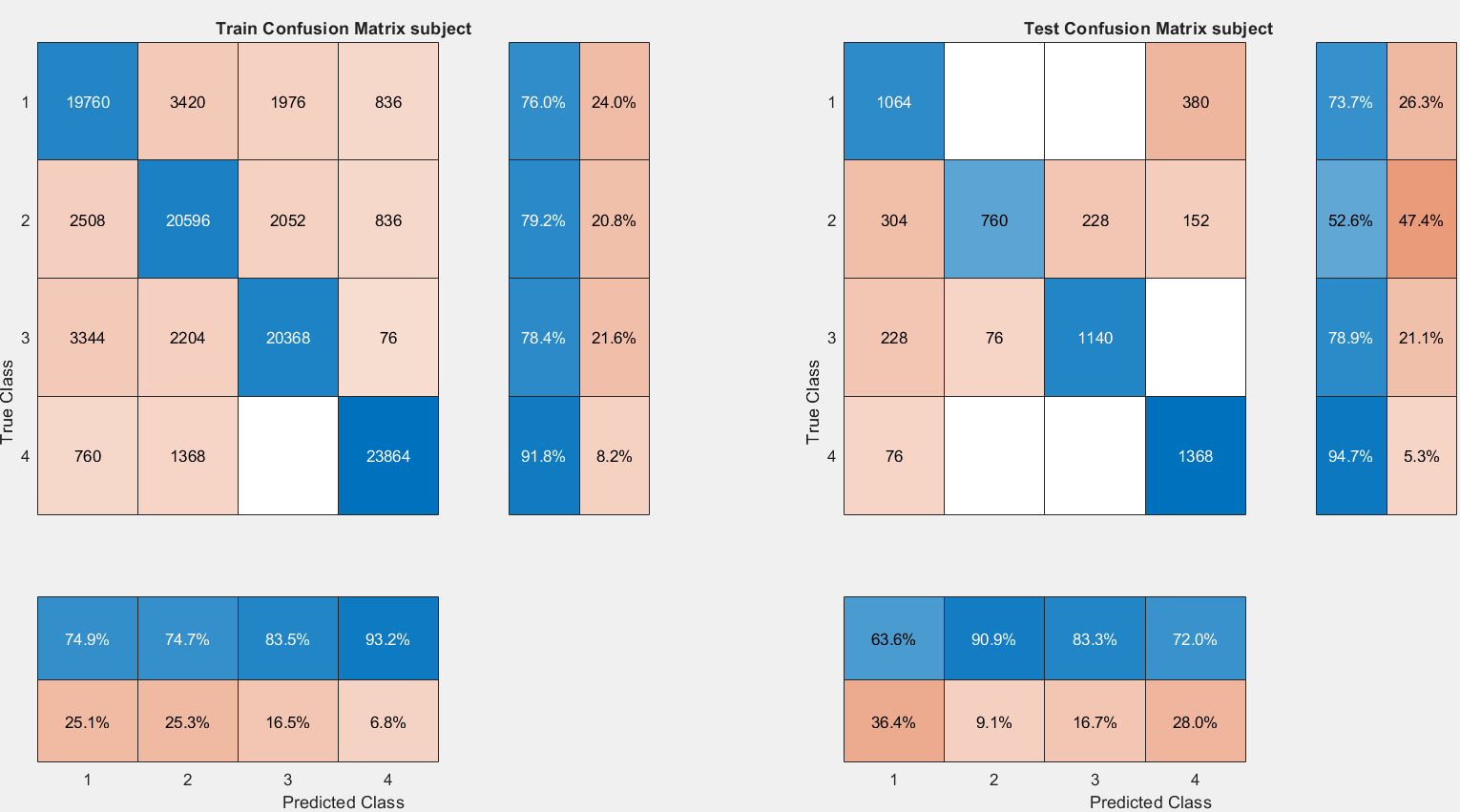
Test Accuracy: 80.56%



Subject 2

Train Accuracy: 81.36%

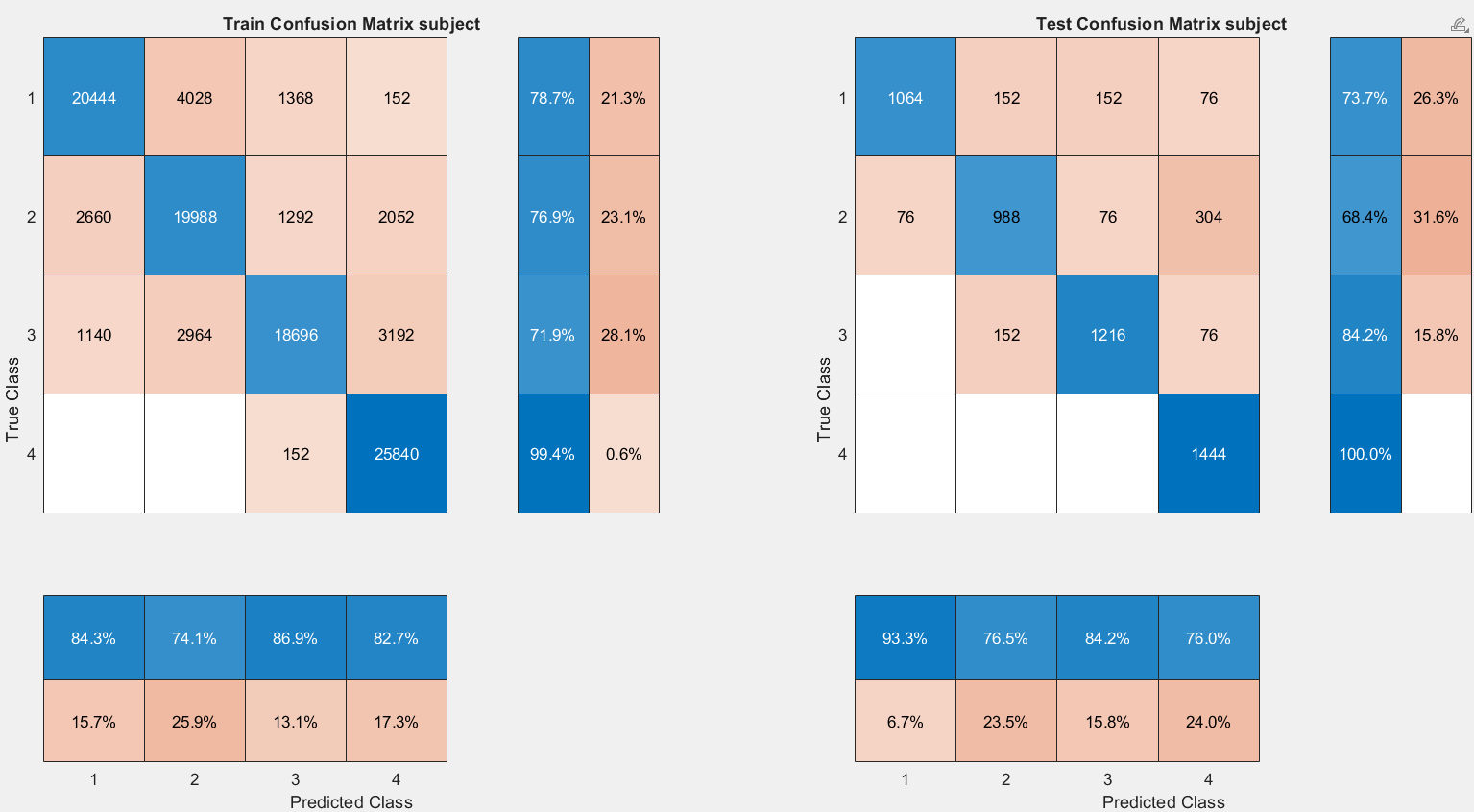
Test Accuracy: 75.00%



Subject 3

Train Accuracy: 81.73%

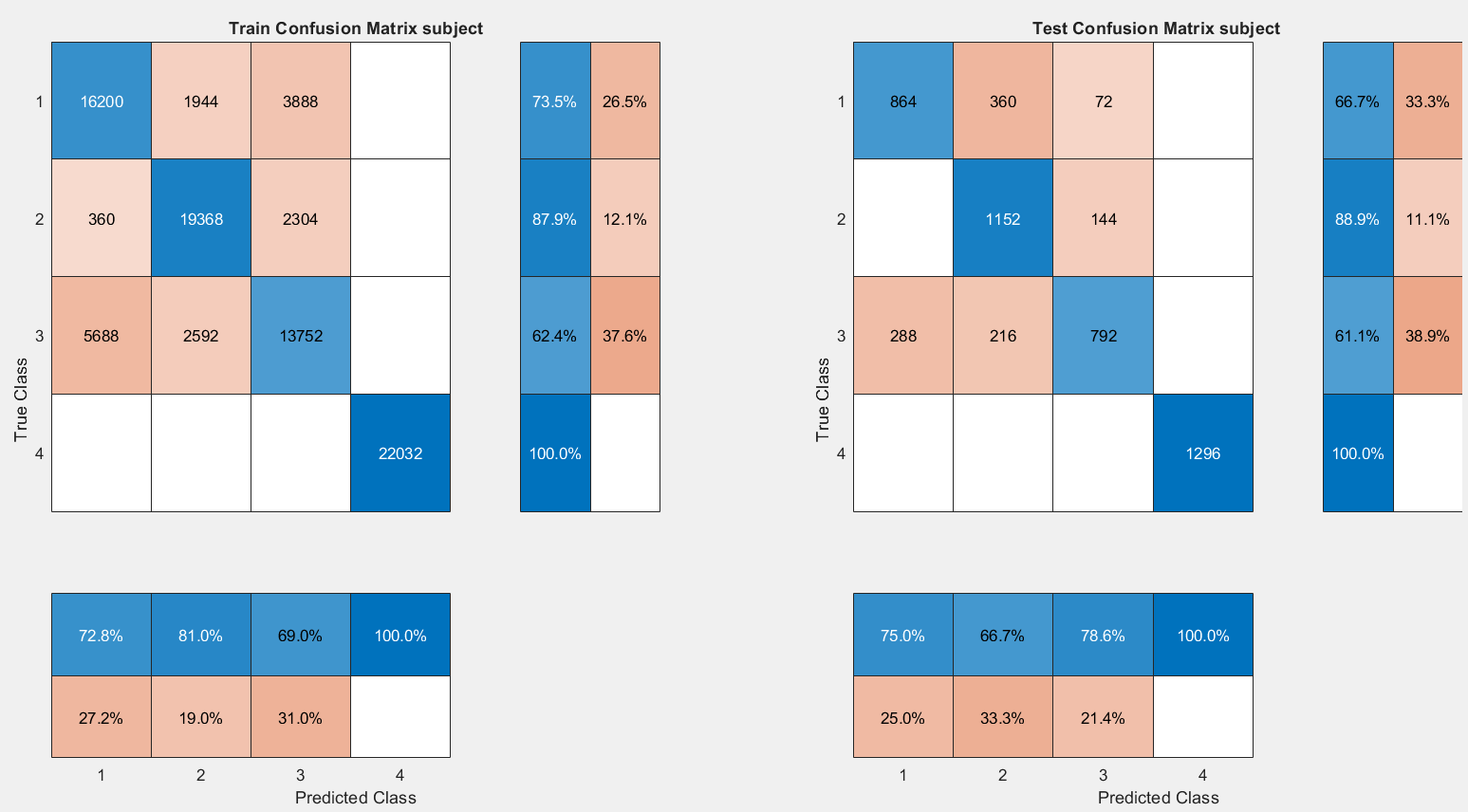
Test Accuracy: 81.58%



Subject 4

Train Accuracy: 80.96%

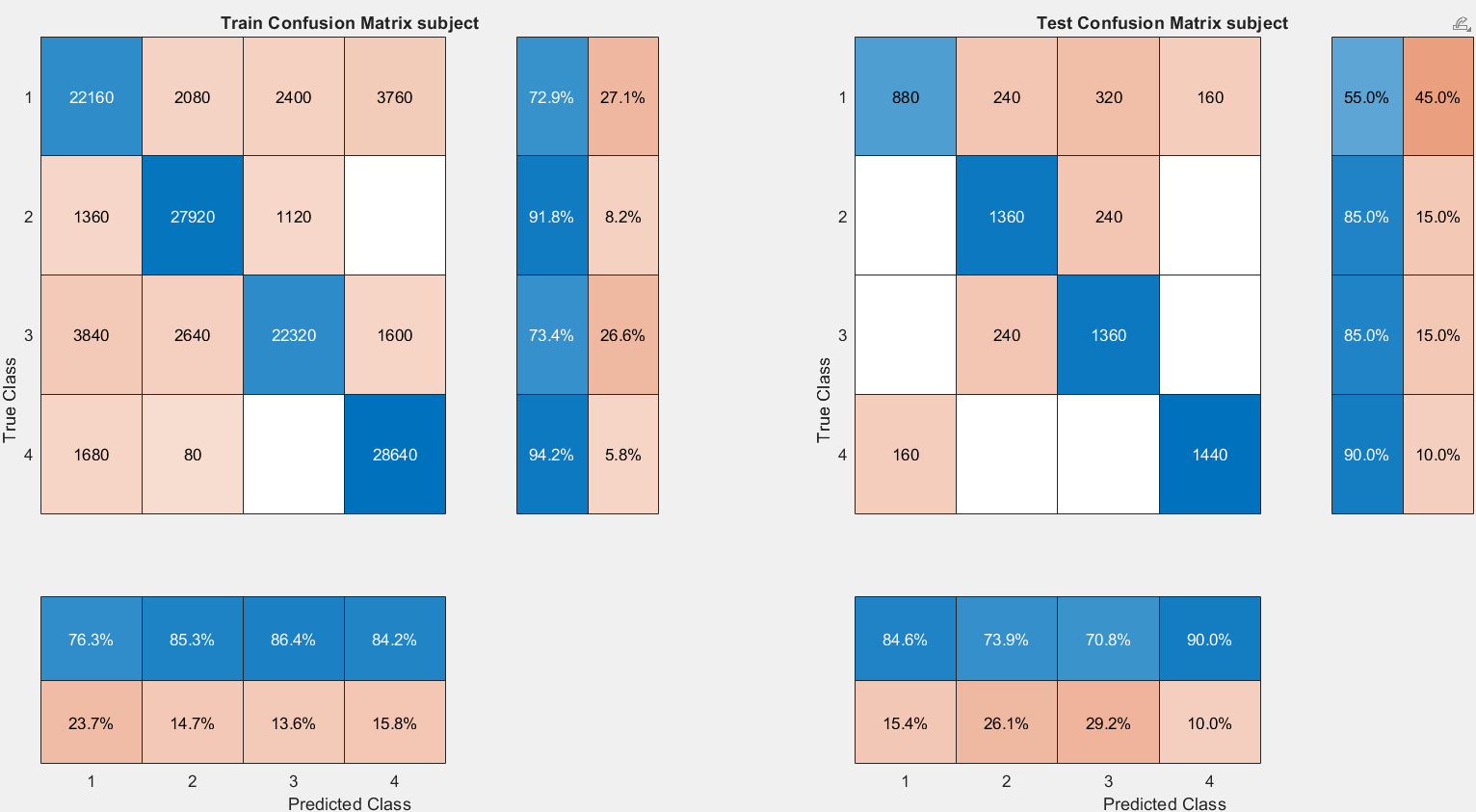
Test Accuracy: 79.17%



Subject 5

Train Accuracy: 83.09%

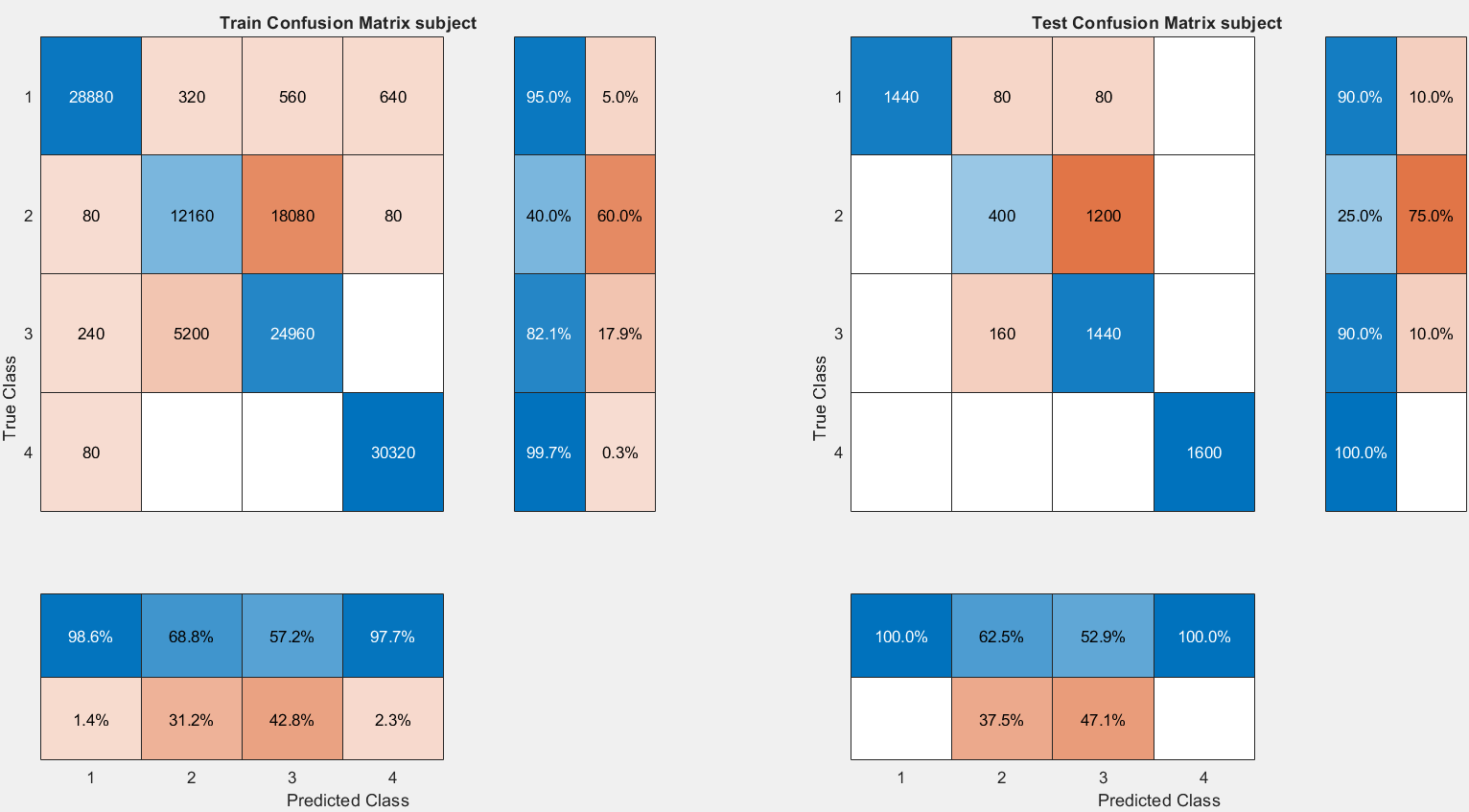
Test Accuracy: 78.75%



Subject 6

Train Accuracy: 79.21%

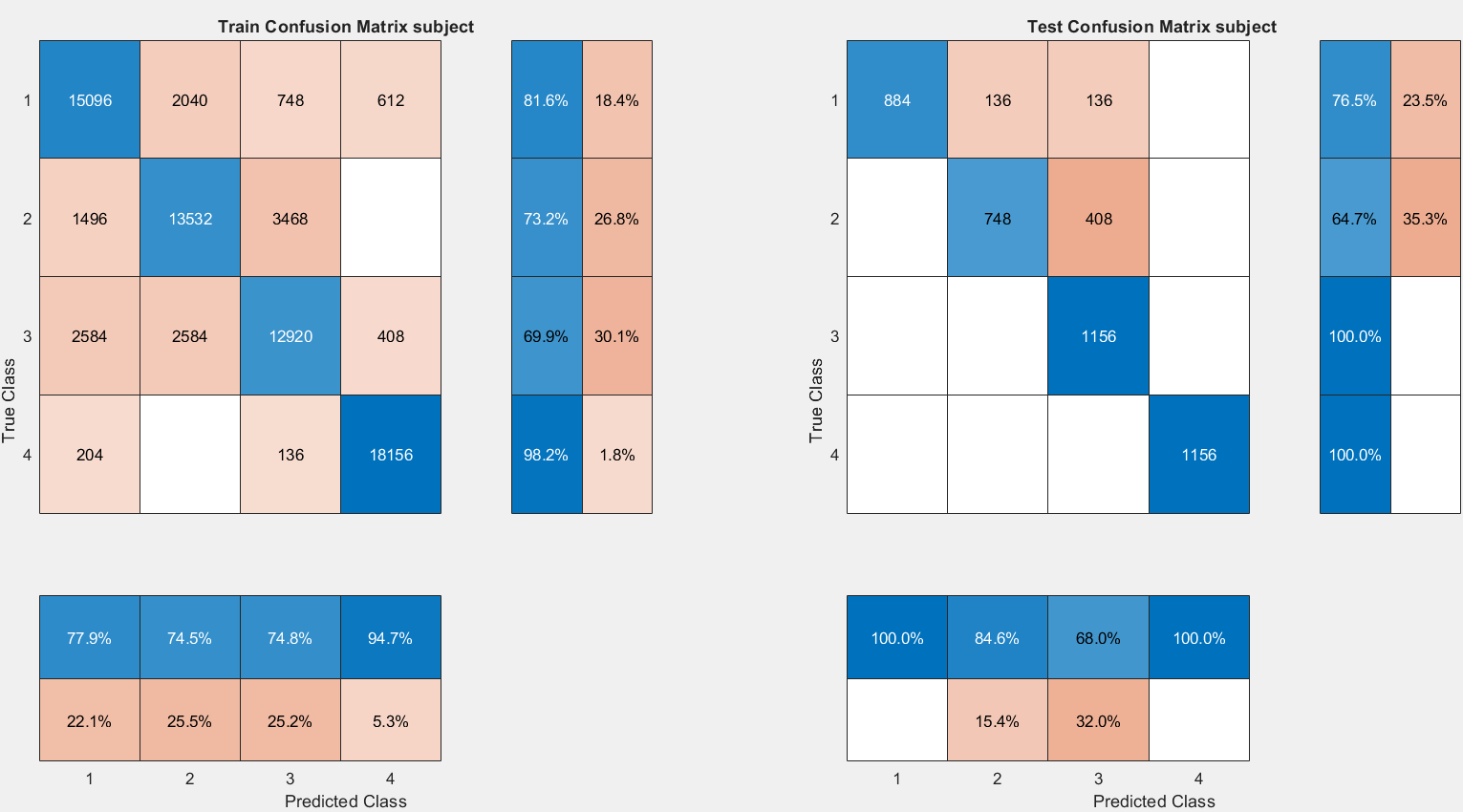
Test Accuracy: 76.25%



Subject 7

Train Accuracy: 80.70%

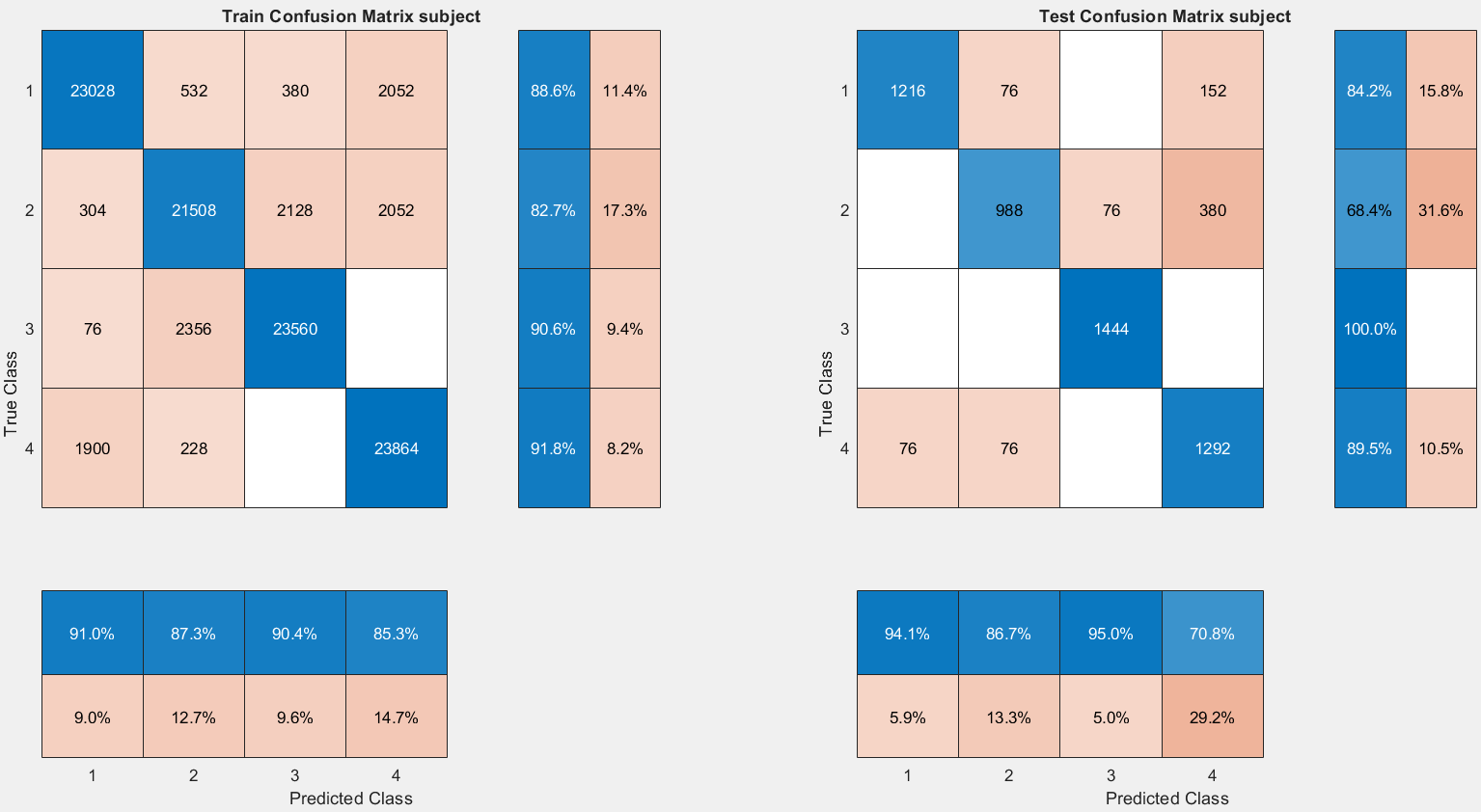
Test Accuracy: 85.29%



Subject 8

Train Accuracy: 88.45%

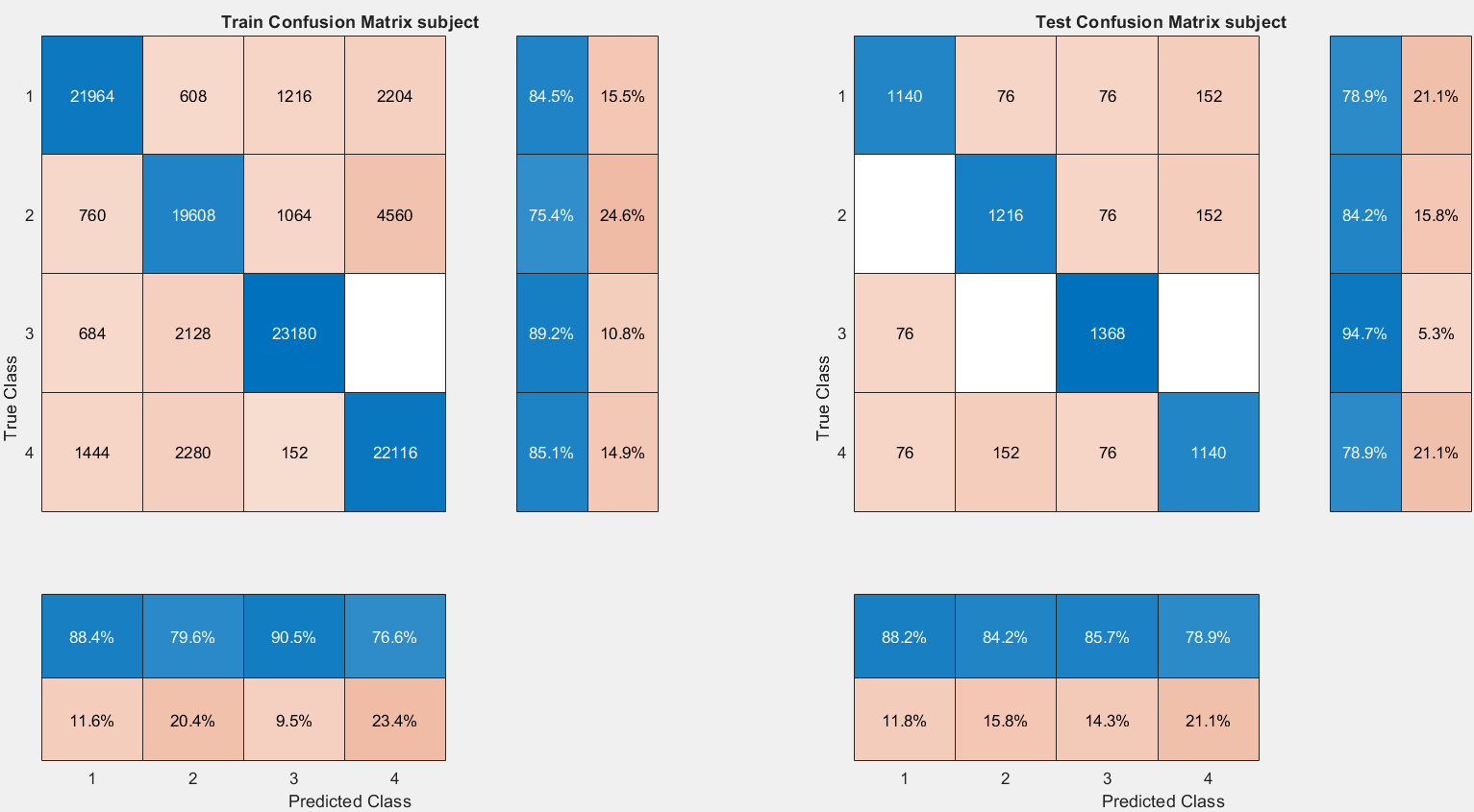
Test Accuracy: 85.53%



Subject 9

Train Accuracy: 83.55%

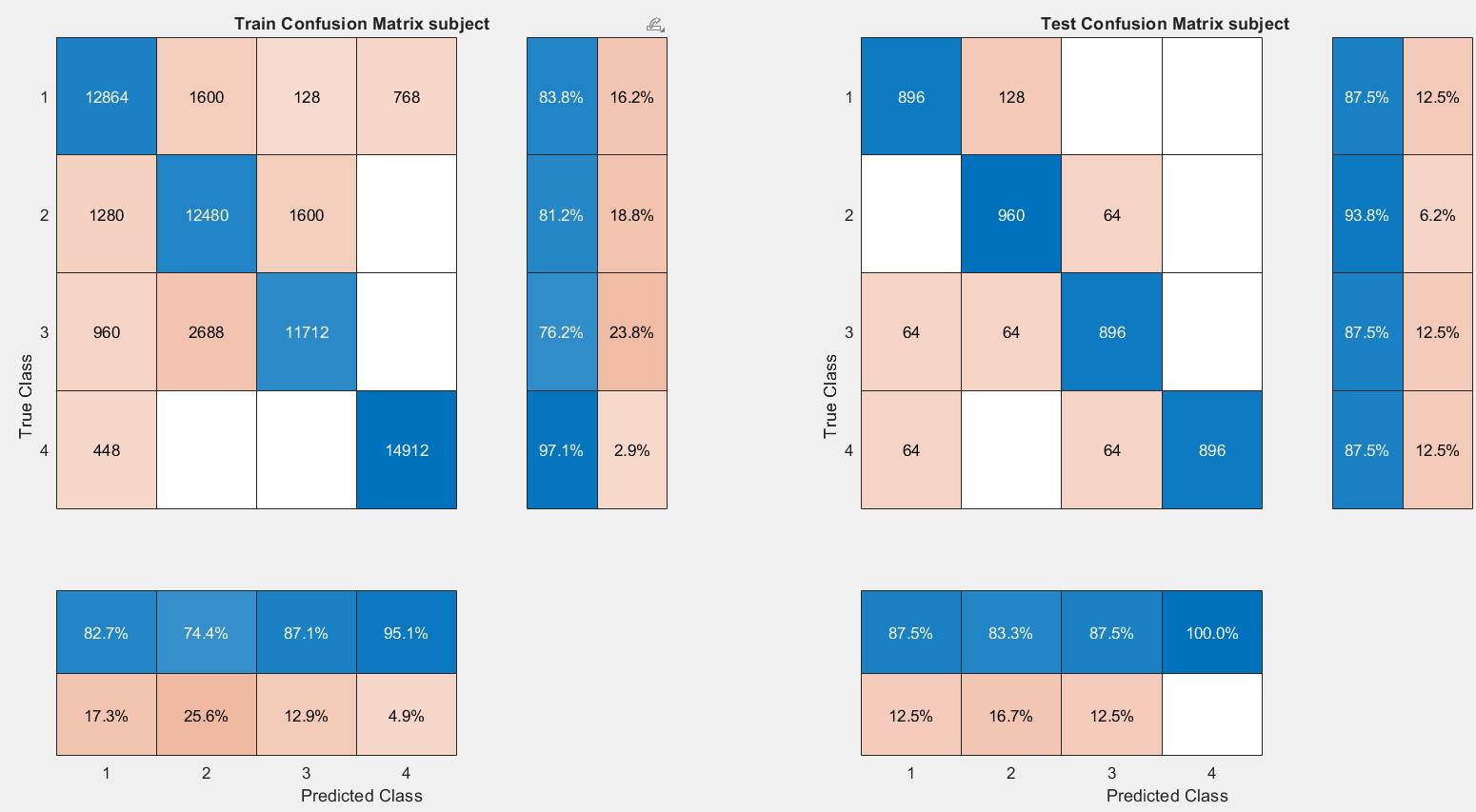
Test Accuracy: 84.21%



Subject 10

Train Accuracy: 84.58%

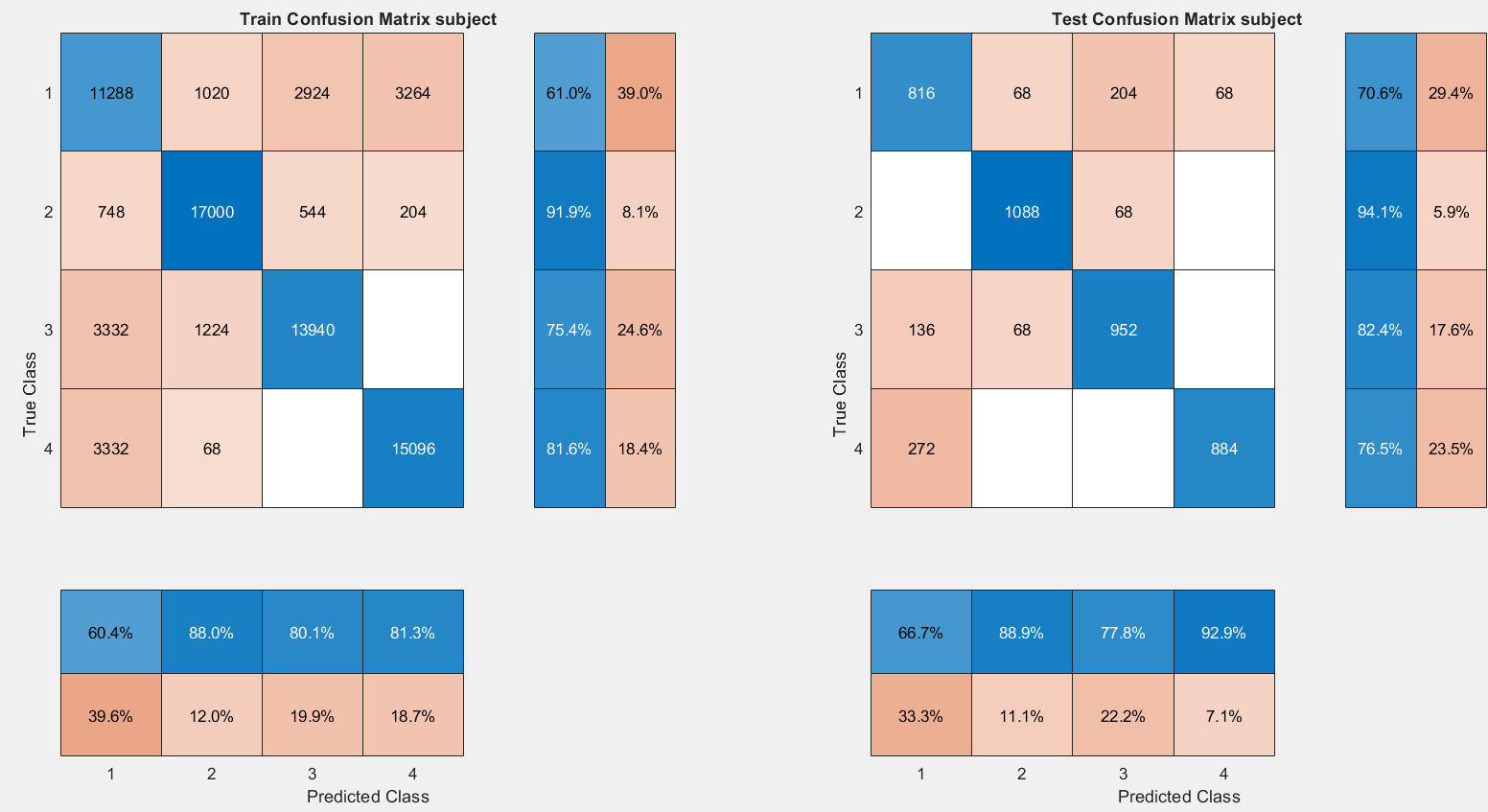
Test Accuracy: 89.06%



Subject 11

Train Accuracy: 77.48%

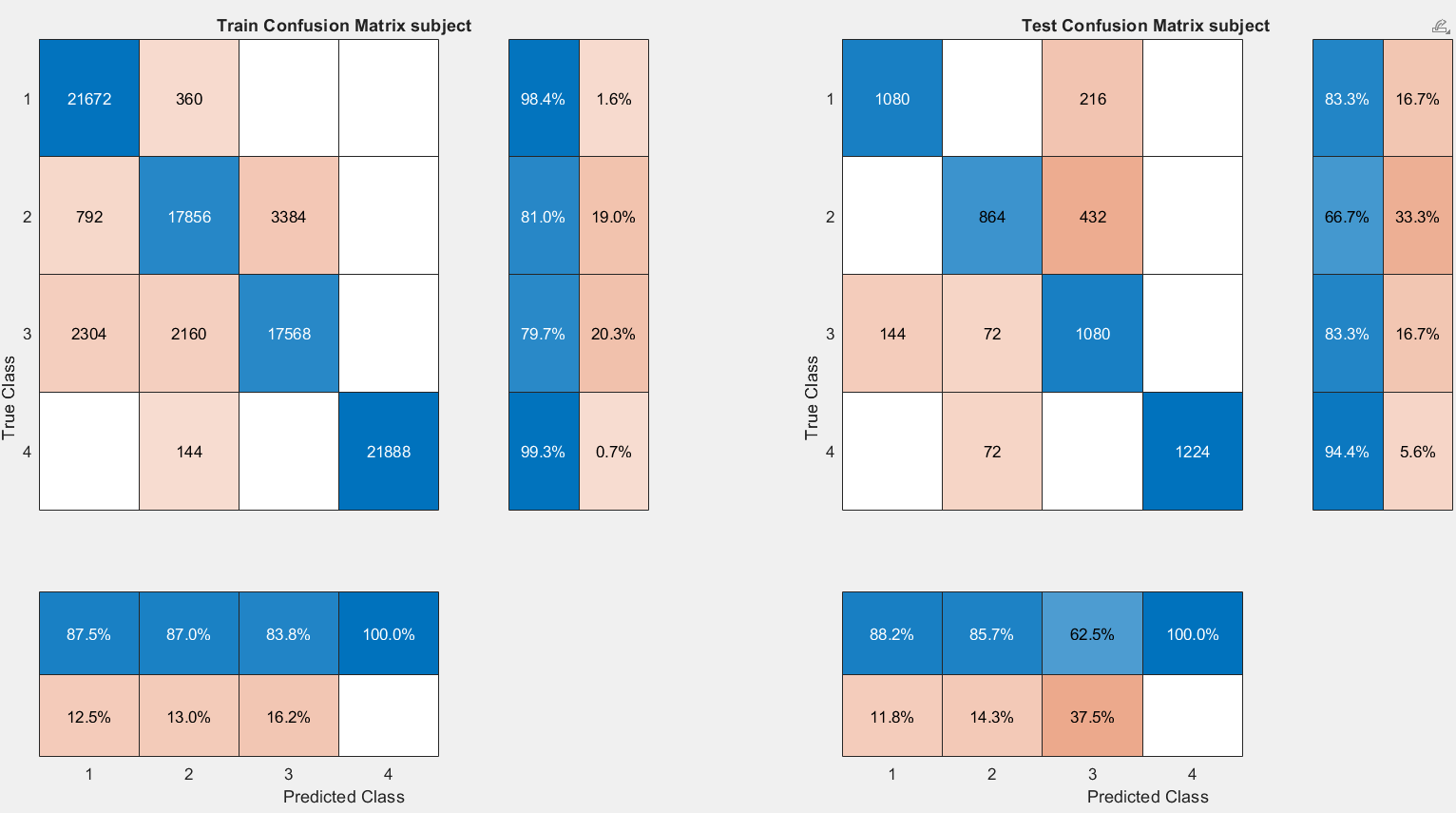
Test Accuracy: 80.88%



Subject 12

Train Accuracy: 89.62%

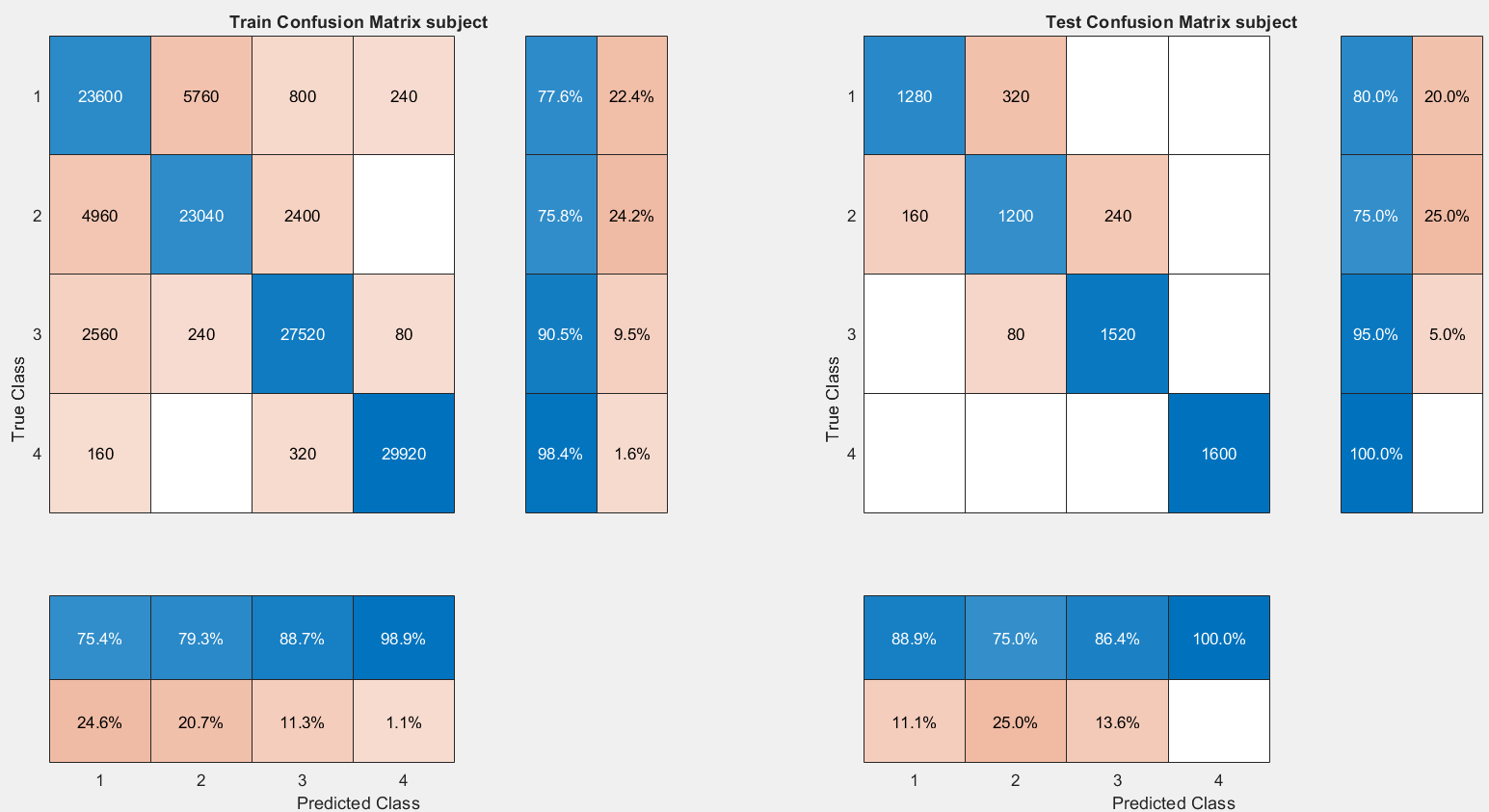
Test Accuracy: 81.94%



Subject 13

Train Accuracy: 85.59%

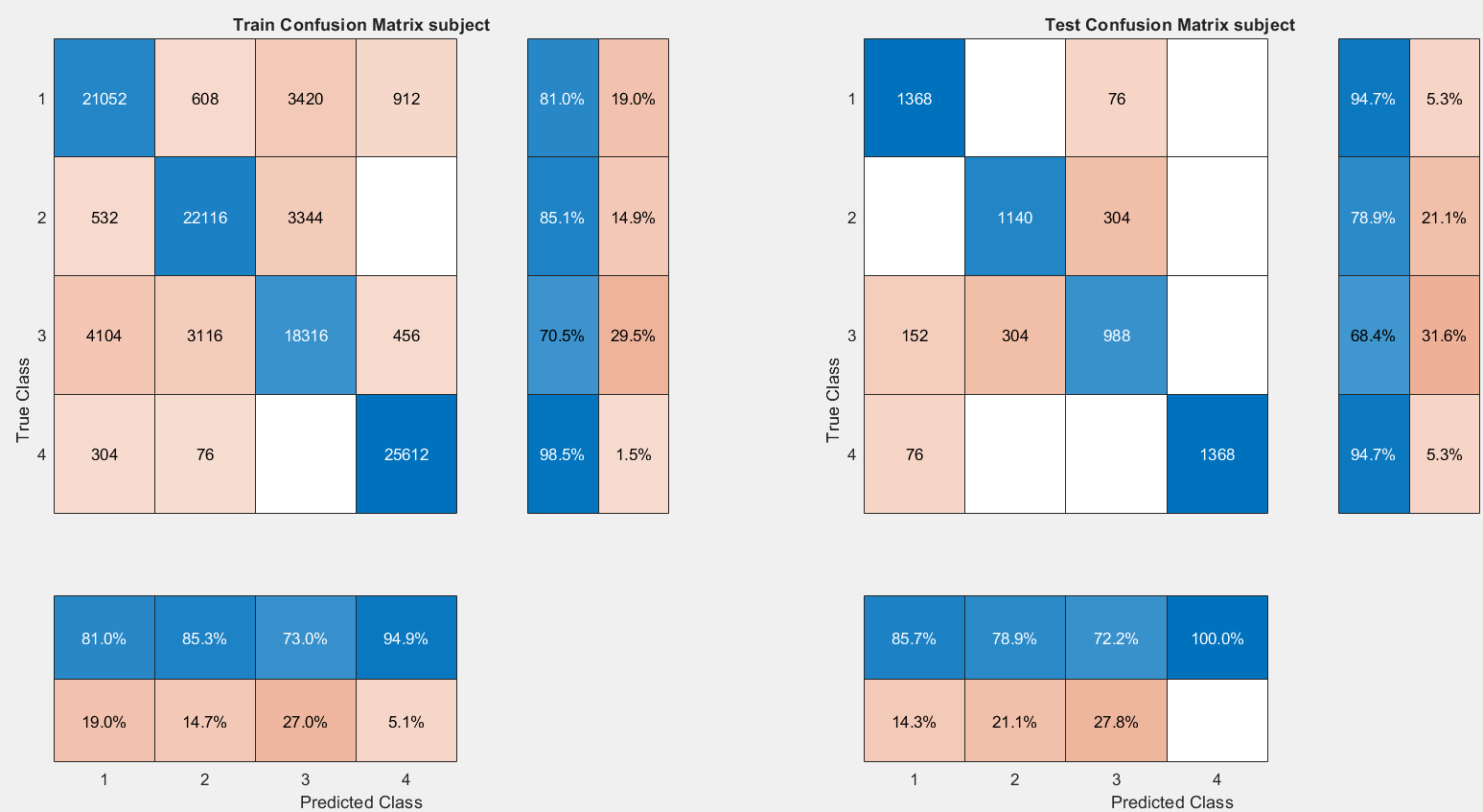
Test Accuracy: 87.50%



Subject 14

Train Accuracy: 83.77%

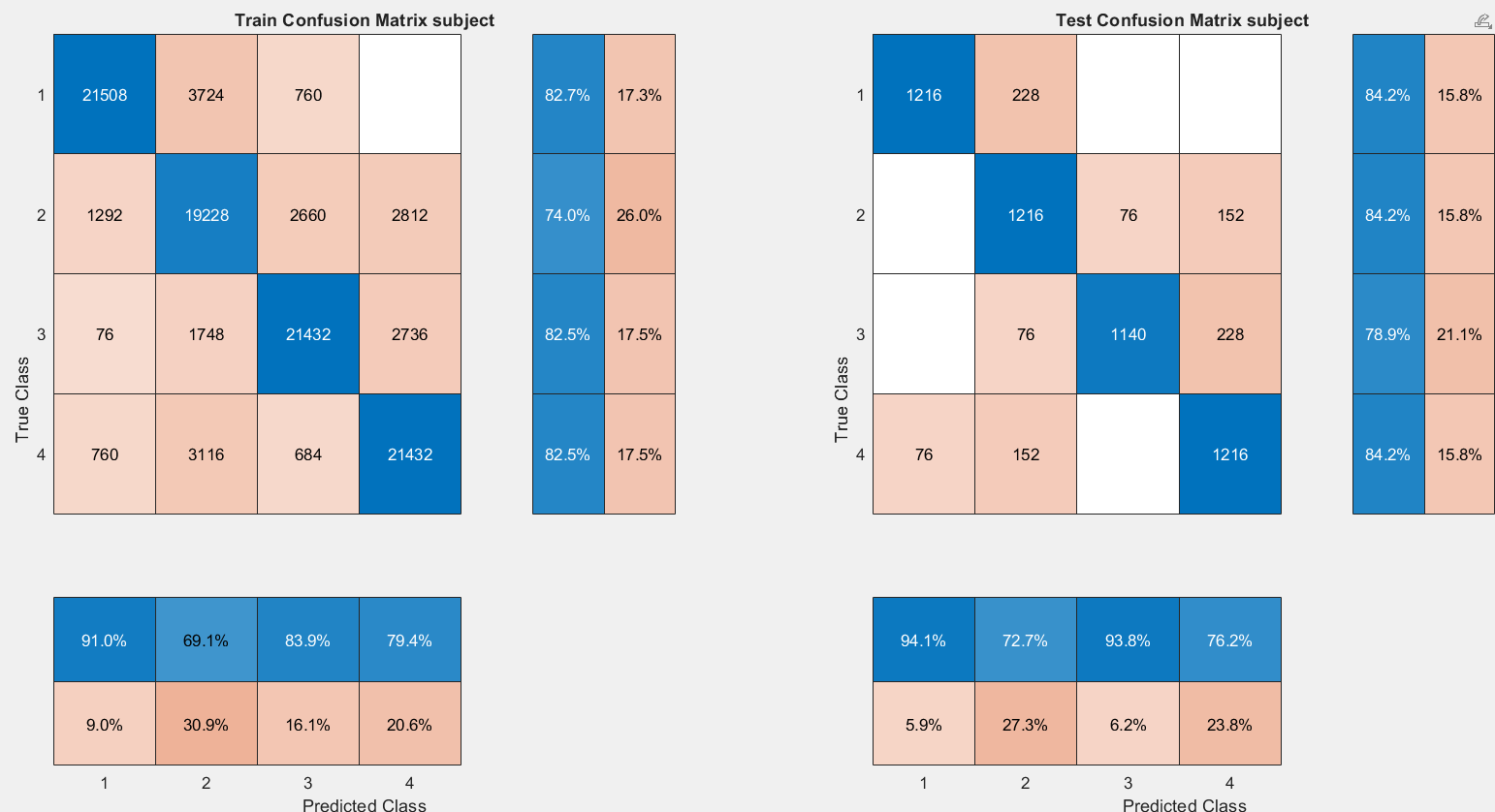
Test Accuracy: 84.21%



Subject 15

Train Accuracy: 80.41%

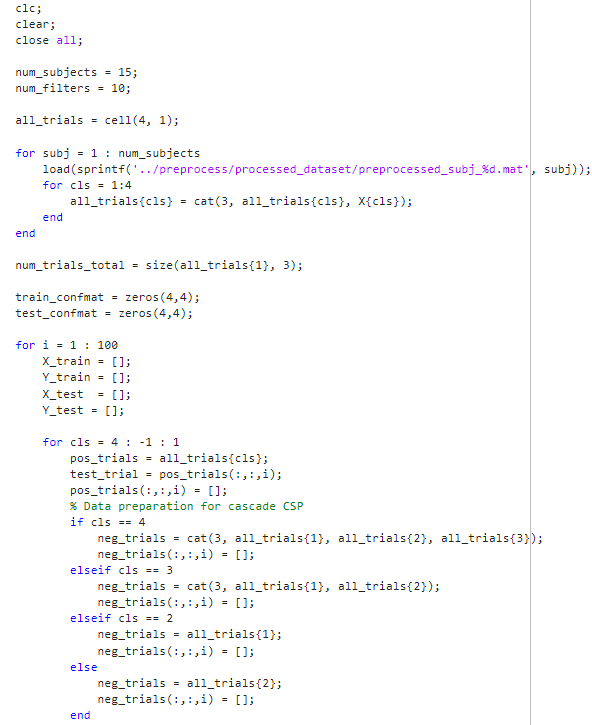
Test Accuracy: 82.89%

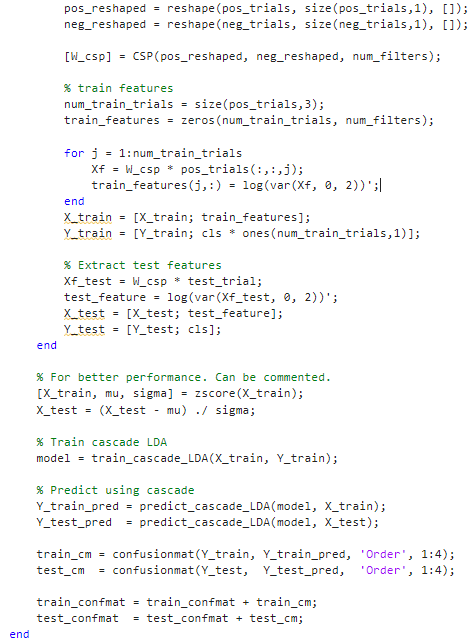


## آموزش یک مدل برای تمام subject ها

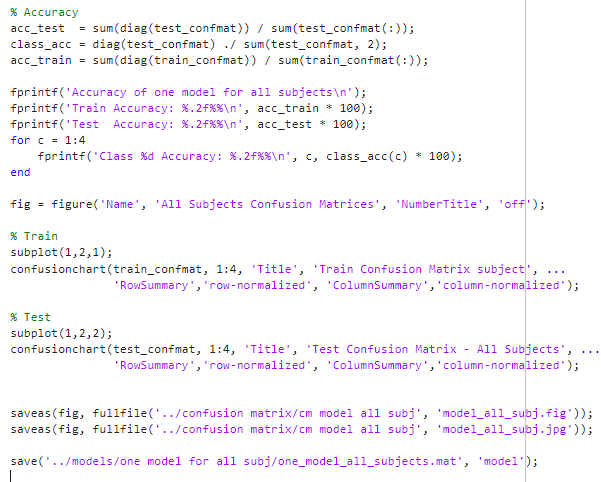
مطابق کدی که در فایل train\_model\_all\_subj.m در پوشه ی train and evaluate نوشتم، در این رویکرد داده های تمام subject ها را برای آموزش فقط یک مدل با سه طبقه بند به صورت آبشاری استفاده می کنم. نهایتا ماتریس کانفیوژن آن در مسیر confusion matrix/cm model all subj و مدل آن آموزش دیده در مسیر model/one model for all subj ذخیره می شوند.

کد این بخش شبیه کد رویکرد قبل است تنها تفاوت آن در این است که داده های هر subj در به ازای هر کلاس در یک cell جدید ریخته می شوند و تمامی مراحل از جمله LOO، CSP و LDA روی این داده های جمع شده اجرا می شود.





ادامه کد در تصویر بعدی.



## نتایج آموزش

ماتریس کانفیوژن آموزش و تست و همچنین دقت مدل روی داده های تست برای هر کلاس به شرح زیر است.

Accuracy of one model for all subjects

Train Accuracy: 74.69%

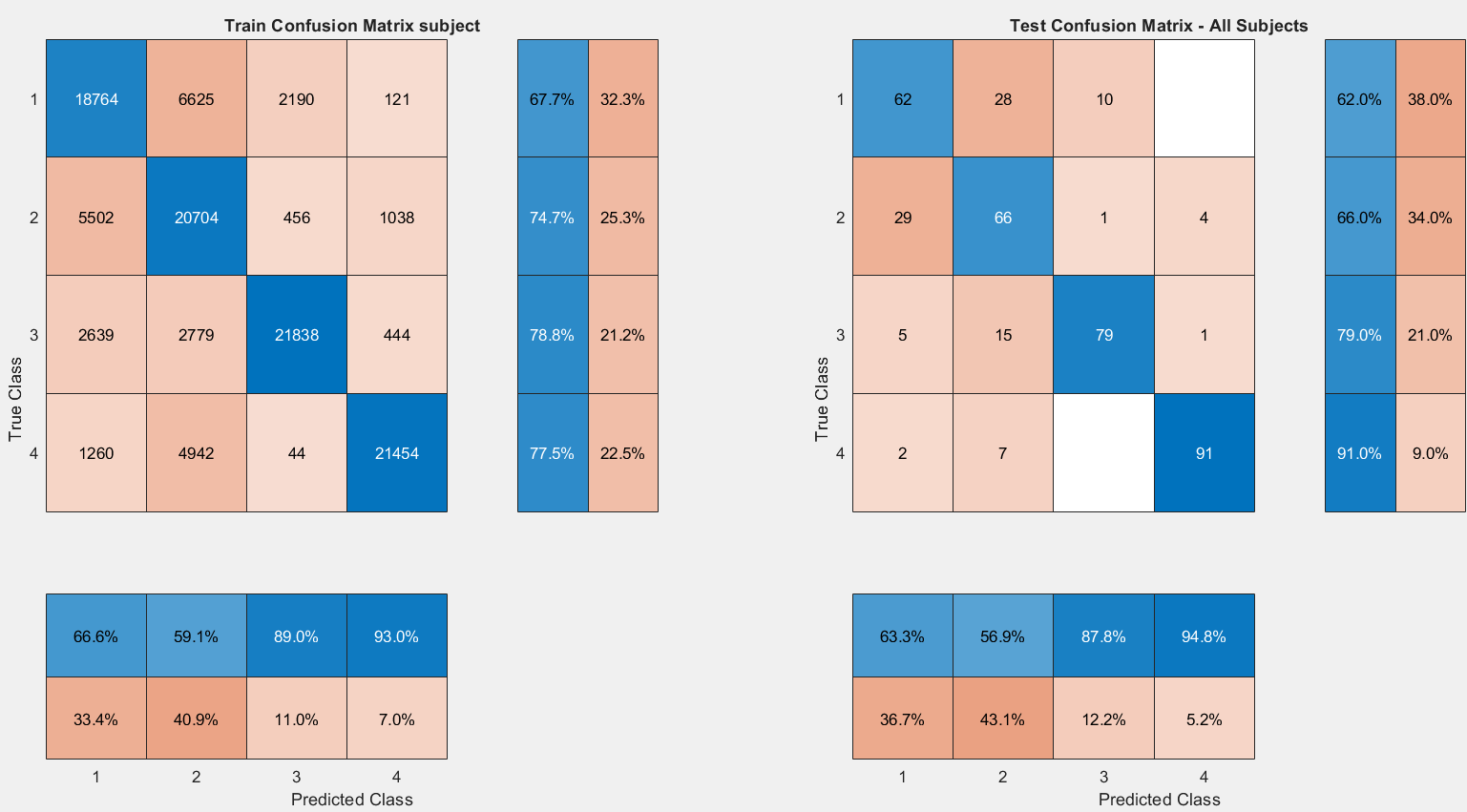
Test Accuracy: 74.50%

Class 1 Accuracy: 62.00%

Class 2 Accuracy: 66.00%

Class 3 Accuracy: 79.00%

Class 4 Accuracy: 91.00%



همان طور که می بینیم دقت مدل برای هر کلاس نزدیک آن اعدادی شد که طبق فایل Report.pdf انتظار می رفت.

# مراجع

[1] [Motor cortex - Wikipedia](https://en.wikipedia.org/wiki/Motor_cortex)

[2] [Delta wave - Wikipedia](https://en.wikipedia.org/wiki/Delta_wave)

[3] [Theta wave - Wikipedia](https://en.wikipedia.org/wiki/Theta_wave)

[4] [Alpha wave - Wikipedia](https://en.wikipedia.org/wiki/Alpha_wave)

[5] [Beta wave - Wikipedia](https://en.wikipedia.org/wiki/Beta_wave)

[6] [Hyperparameter optimization - Wikipedia](https://en.wikipedia.org/wiki/Hyperparameter_optimization)

[7] [Cascading classifiers - Wikipedia](https://en.wikipedia.org/wiki/Cascading_classifiers)