باسمه تعالی

دانشگاه صنعتی شریف

دانشکده مهندسی برق

۲۵۷۴۲ گروه ۴ ـ سیگنال‌ها و سیستم‌ها ـ بهار ۹۸-۱۳۹۷

پروژه ـ فاز دوم

موعد تحویل: ۱۷ خرداد ۱۳۹۸، ساعت ۲۳:۵۵

نحوه‌ی تحویل:

- گزارش پروژه خود را در قالب یک فایل pdf. تحویل دهید. در گزارش لازم است تمامی خروجی‌ها و نتایج نهایی، پرسش‌های متن تمرین، و توضیح مختصری از فرآیند حل مسأله‌ی خود در هر قسمت را ذکر کنید.
- کد کامل تمرین را در قالب یک فایل m. تحویل دهید. لازم است بخش‌های مختلف پروژه در sectionهای مختلف تفکیک شوند و کد تحویلی، منظّم و دارای کامنت‌گذاری مناسب باشد. بدیهی است آپلود کردن کدی که به درستی اجرا نشود، به منزله‌ی فاقد اعتبار بودن نتایج گزارش‌شده نیز می‌باشد.
- توابعی را که (در صورت لزوم) نوشته‌اید، در فالب فایل‌های m. در کنار فایل‌های گزارش و کد اصلی پروژه، ضمیمه کنید.
- مجموعه‌ی تمامی فایل‌ها (گزارش، کد اصلی، توابع، و خروجی‌های دیگر در صورت لزوم) را در قالب یک فایل zip/.rar. ذخیره کرده و از طریق سامانه‌ی CW تحویل دهید.
- نام‌گذاری فایل‌های تحویلی را به صورت

  `ProjectPh2_StudentNumber1_StudentNumber2.pdf/.m/.zip/.rar`

  انجام دهید.

معیار نمره دهی:

- ساختار مرتب و حرفه‌ای گزارش
- استفاده از توابع و الگوریتم‌های مناسب
- پاسخ به سؤالات تئوری و توضیح روش‌های مطلوب سوال
- کد و گزارش خروجی کد برای خواسته‌های مسأله

نکات تکمیلی:

- همواره در تمامی تمارین و پروژه‌ها، تا سقف ۱۰٪ نمره اضافه برای قسمت‌های امتیازی و نیز هر گونه روش‌های ابتکاری و فرادرسی در نظر گرفته می‌شود و سقف نمره‌ی قابل کسب معادل با ۱۱۰/۱۰۰ می‌باشد.
- *شرافت انسانی ارزشی به مراتب والاتر از تعلّقات دنیوی دارد. رونویسی تمارین، زیر پا گذاشتن شرافت خویشتن است؛ به کسانی که شرافتشان را زیر پا می‌گذارند هیچ نمره‌ای تعلّق نمی‌گیرد.*

در قسمت تئوری فاز اوّل پروژه به مباحث تبدیل فوریه‌ی گسسته و همچنین نمونه‌برداری پرداختیم. در قسمت بعدی آن با سیگنال‌های EEG آشنا شدیم و نهایتاً سعی در خوشه‌بندی الکترودهای کلاه EEG داشتیم. در این فاز از پروژه، ابتدا به برّرسی نکات مربوط به طرّاحی فیلتر می‌پردازیم و نهایتاً با استفاده از روش‌های مبتنی یادگیری ماشین، سعی در پیاده‌سازی فرایندی برای تشخیص کلمات با استفاده از سیگنال‌های EEG خواهیم داشت.

طرّاحی فیلتر

فیلترهای انتخاب فرکانس[1] ابزارهایی برای حذف محتوای فرکانسی ناخواسته مانند نویز، و نگه‌داری باندهای فرکانسی مطلوب می‌باشند. برای طرّاحی و استفاده از فیلترها لازم است نکاتی را مورد توجّه قرار داد که در این بخش، به برّرسی نمونه‌هایی از مهم‌ترینِ این نکات می‌پردازیم.
اندازه و فاز، دو مشخّصه‌ی مهمّ یک فیلتر هستند. در حالت کلّی، تبدیل فوریه را می‌توانیم به صورت زیر (بر مبنای اندازه و فاز) نمایش دهیم:

$$H(j\omega) = |H(j\omega)|\, e^{j\measuredangle H(j\omega)} \qquad (۱)$$

اگر چه تعریف فوق از نظر ریاضی معتبر است، امّا ویژگی ناخوش‌آیندی دارد که به شرح زیر است:
فرض کنید $H(j\omega)$ به صورت حاصل‌ضرب دو عبارت به شکل $H(j\omega) = F(j\omega)G(j\omega)$ قابل بیان باشد که در آن، $F(j\omega)$ همواره حقیقی باشد. در این صورت بدیهی است که هرگاه $F(j\omega)$ تغییر علامت بدهد، فاز $H(j\omega)$ به اندازه‌ی π رادیان تغییر می‌کند؛ لذا همواره در محل‌های تغییر علامت $F(j\omega)$، فاز تبدیل فوریه ناپیوستگی‌هایی با اندازه‌ی π خواهد داشت. با این حال، این ناپیوستگی‌ها صرفاً ناشی از تعریف ما از فاز هستند و رفتار طبیعی فاز تبدیل را ـ که احتمالاً یک رفتار هموار و پیوسته بوده است ـ به درستی توصیف نمی‌کنند. برای رفع این مشکل، کافی است از تعریف زیر استفاده کنیم:

$$H(j\omega) = A(j\omega)e^{j\phi(j\omega)} \qquad (۲)$$

که در آن $A(j\omega)$ حقیقی است و می‌تواند مثبت یا منفی باشد. دقّت کنید که تنها تفاوت معادله‌ی ۲ با معادله‌ی ۱ آن است که در تعریف جدید، $|H(j\omega)|$ را که یک عدد حقیقی مثبت است، با $A(j\omega)$ جایگزین کرده‌ایم که یک عدد حقیقی مثبت یا منفی است. این کار، مشکل مذکور را حل کرده و اثر تغییر علامت‌های ممکن را از فاز تبدیل خارج کرده و به $A(j\omega)$ منتقل می‌کند. عبارت $\phi(j\omega)$ از معادله‌ی ۲ را *فاز تعمیم‌یافته‌ی خطّی* یا generalized linear phase می‌نامیم.
فاز فیلتر از ویژگی‌های مهمّ آن است که در نگاه اوّل شاید به آن توجّه نشود. یکی از مهم‌ترین ویژگی‌هایی که در فیلترها مطلوب است، خطّی بودن فاز تعمیم‌یافته‌ی فیلتر است. پرسش زیر، اهمیّت این موضوع را برّرسی می‌کند.

- فرض کنید فیلتری دارید که اندازه‌ی آن ثابت است. فاز این فیلتر را نیز به صورت زیر در نظر بگیرید:

$$\phi(\omega) = -\frac{\pi}{3}\mathrm{sgn}(\omega)$$

ورودی فیلتر، سیگنال زیر است:

$$x(t) = \cos(\omega_0 t) + \cos(2\omega_0 t)$$

خروجی سیستم را پیدا کنید. سیگنال ورودی از دو سیگنال تک‌فرکانس تشکیل شده بود. هر یک از این دو سیگنال، پس از عبور از این فیلتر چه مقدار در زمان تأخیر پیدا کرده‌اند؟
این بار فرض کنید فاز فیلتر خطّی باشد، یعنی

$$\phi(\omega) = -\frac{\pi}{3}\omega$$

پاسخ دو پرسش فوق را برای این حالت نیز بیان کنید. به نظر شما خطّی بودن فاز فیلتر چه اثر مثبتی دارد؟ خطّی نبودن فاز فیلتر چگونه می‌تواند باعث ایجاد اعوجاج در سیگنال خروجی شود؟

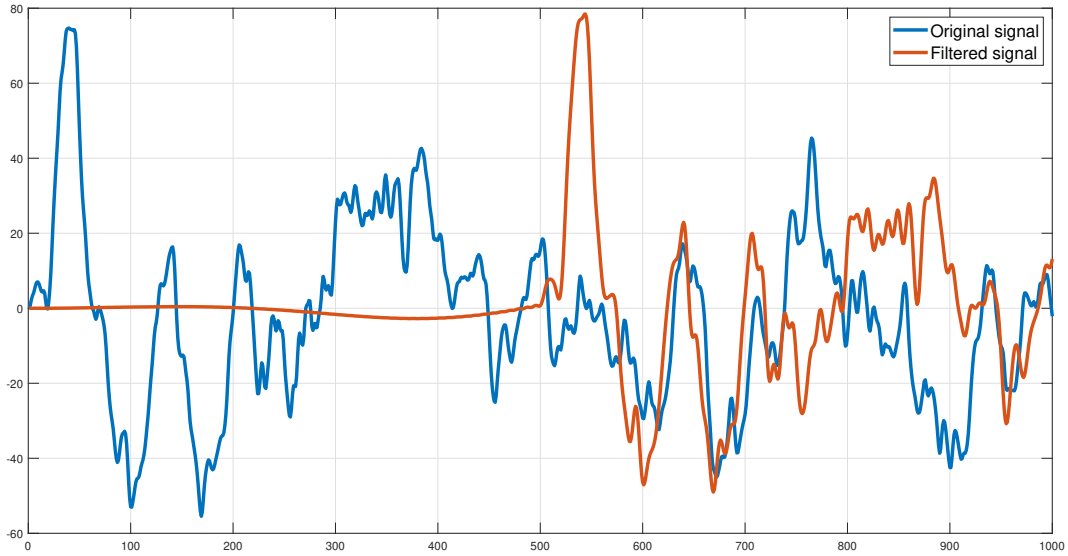
[1] frequency selective filters

با توجه به پرسش فوق، باید به این نکته پی برده باشید چرا در طرّاحی فیلتر، همواره به دنبال آن هستیم که در صورت امکان، فاز فیلتر در باندهای فرکانسی مورد توجّه ما، خطی باشد. برای این منظور متغیّری تحت عنوان group delay را به صورت زیر تعریف می‌کنیم:

$$\mathrm{gd}(\omega) = -\frac{d\phi(\omega)}{d\omega} \qquad (۳)$$

که در آن، $\phi(\omega)$ فاز تعمیم‌یافته‌ی خطی است. بُعد group delay برای سیستم‌های گسسته، تعداد نمونه یا سمپل بوده و برای سیستم‌های پیوسته، زمان (بر حسب ثانیه) است.

گزاره‌ی مهم: تأخیر گروه برای محتوای هر فرکانس، نشانگر این است که پس از عبور از فیلتر، چه مقدار تأخیر (بر حسب سمپل یا ثانیه) خواهد یافت. (به عبارت دقیق‌تر، این کمیت در حالت گسسته بدون بُعد است.) درستی گزاره‌ی فوق را برای سیگنال ورودی و دو سیستم معرّفی‌شده در پرسش قبل تحقیق کنید. اگر در فاز قبل پروژه به خروجی‌های فیلتر توجّه می‌کردید، می‌توانستید این تأخیر را مشاهده کنید. در شکل زیر نیز به وضوح می‌توان این تأخیر را برای سیگنالی که با فیلتری با $\text{group delay} = 500 \text{ samples}$ فیلتر شده است، مشاهده نمود:



شکل ۱

با توجه به نکات فوق، واضح است که حتماً باید این تأخیر را در محاسبات خود وارد کرده و اثرات آن را در نظر بگیریم، در غیر این صورت به احتمال خوبی در محاسبات خود دچار اشتباه خواهیم شد. برای این کار، ابتدا لازم است مقدار group delay را برای یک فیلتر داده‌شده به دست آوریم.

- با استفاده از روابط ۲ و ؟؟، ثابت کنید:

$$\mathrm{gd}(\omega) = \mathbf{Re}\left\{\frac{j\frac{d}{d\omega}H(\omega)}{H(\omega)}\right\} \qquad (۴)$$

- تابعی بنویسید تا به کمک معادله‌ی ۳، تأخیر گروه را برای هر فیلتر دلخواه محاسبه کند. بدیهی است مجاز به استفاده از توابع آماده‌ی متلب که مستقیماً تأخیر گروه را محاسبه می‌کنند، نیستید. نام‌گذاری تابع شما باید به صورت زیر باشد:

```
groupdelay(h,N)
```

که در آن، متغیر `N` مشخّص می‌کند که DFT مورد استفاده در محاسبه‌ی تأخیر گروه (به کمک تابع `fft`) چند نقطه‌ای باشد. هر چه این عدد بزرگتر باشد دقت محاسبات بالاتر است. (چرا؟)

- نمودار خروجی تابع بالا را، بر حسب فرکانس، برای هر فیلتری که در این تمرین استفاده خواهید کرد، در گزارش بیاورید. (برای بررسی صحّت عملکرد تابع خود، می‌توانید نتایج آن را با تابع `grpdelay` متلب مقایسه کنید.)
- با استفاده از تابع بالا، تابعی دیگر بنویسید که برای یک فیلتر و سیگنال داده‌شده، سیگنال را فیلتر کرده و تأخیر ناشی از فیلترینگ را نیز لحاظ کرده و خنثی کند. تابع را به صورت زیر نام‌گذاری کنید:

```
zphasefilter(h,x)
```

که در آن `x` سیگنال ورودی، و `h` پاسخ ضربه‌ی فیلتر است. آیا فیلتری علّی و حقیقی وجود دارد که فاز آن در تمام فرکانس‌ها صفر باشد؟

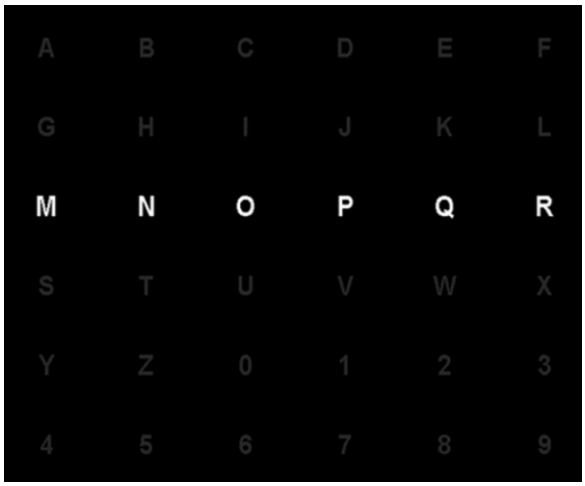
توجّه: تولباکس متلب تحت عنوان filter designer مجموعه‌ای غنی از ابزارهای طرّاحی و بررسی فیلترها است که در فاز قبلی پروژه با آن آشنا شدید. در قسمت‌های بعدی می‌توانید برای طرّاحی فیلتر از این تولباکس استفاده نموده و با استفاده از ابزارهای موجود در آن، اطلاعات زیادی در مورد فیلترها کسب کنید.

شناسایی کلمات

تمامی مراحلی که تا کنون طی شده‌اند برای آماده‌شدن داده‌ها جهت استفاده در این بخش از پروژه بوده است. در این قسمت سعی داریم که با استفاده از الگوریتم‌های مبتنی بر یادگیری ماشین و از طریق سیگنال‌های مغزی EEG، کلمه‌ای را که فرد به آن فکر می‌کند، تشخیص دهیم و یک سیستم تایپ با مغز بسازیم!

شرح آزمایش

فرد مورد آزمایش در مقابل نمایش‌گری به شکل ۲ قرار می‌گیرد. این نمایش‌گر می‌تواند همه‌ی حروف انگلیسی و اعداد از صفر تا ۹ را با آرایش 6×6 (مجموعاً ۳۶ کاراکتر) نشان دهد. هدف نهایی آن است که فرد تنها با نگاه کردن به این صفحه‌ی نمایش بتواند با یک سیستم مبتنی بر هوش مصنوعی و یادگیری ماشین ارتباط برقرار کند، به گونه‌ای که کلمه‌ای که در ذهن دارد از طریق سیستم شناسایی شده و نمایش داده شود. برای این منظور، لازم است کلمه به صورت حرف به حرف مشخّص شود. برای تعیین یک حرف مشخّص، فرد برای مدّتی به محلّ آن کاراکتر روی صفحه‌ی نمایش خیره می‌شود. در طول این زمان، همه‌ی کاراکترهای نمایش‌گر با الگوی زمانی مشخّصی روشن و خاموش می‌شوند، به گونه‌ای که در هر لحظه از زمان، ممکن است همه‌ی کاراکترها خاموش باشند، و یا بسته به نوع پیاده‌سازی آزمایش، یک یا چند کاراکتر روشن باشند. (در مورد نوع آزمایش و الگوی زمانی روشن و خاموش شدن کاراکترها در ادامه توضیح داده خواهد شد.) در تمام طول این آزمایش، سیگنال‌های مغزی فرد در حال ثبت شدن است.



شکل ۲

هدف نهایی آزمایش آن خواهد بود که با استفاده از سیگنال‌های مغزی، بتوانیم زمان‌هایی را تشخیص دهیم که حرف مدّ نظر فرد، روشن شده است. به عبارت دیگر، انتظار داریم وقتی فرد برای مدّتی به حرفی خیره شده است، در لحظاتی که آن حرف روشن می‌شود، در سیگنال مغزی دریافتی اثراتی مشاهده شود، به گونه‌ای که بتوانیم با پردازش آن‌ها لحظاتی را که کاراکتر مطلوب روی صفحه‌ی نمایش روشن شده‌اند، شناسایی کنیم. اگر در تشخیص این زمان‌ها موفّق باشیم، کار تقریباً تمام شده است، چرا که می‌توانیم با مراجعه به الگوی زمانی روشن و خاموش شدن کاراکترها (که از ابتدا توسّط ما مشخّص شده بود و اطّلاعات آن را در دست داریم)، بفهمیم کدام کاراکتر در آن لحظات روشن شده و بنابراین، حرف مدّ نظر شخص مورد آزمایش را تشخیص دهیم. با تشخیص هر حرف، فرد محلّ نگاه خود را عوض می‌کند و به سراغ حرف بعدی می‌رویم. همچنین برای آن که آزمایش در مقابل نویز و خطا مقاوم‌تر باشد، این فرایند برای هر حرف چندین بار تکرار می‌شود. یعنی به عنوان مثال در مدّت زمانی که فرد به یک کاراکتر خیره شده، فرایند روشن و خاموش شدن حروف ۱۵ بار تکرار شود. بنابراین می‌توانیم ۱۵ کاراکتر را با الگوریتم فوق تشخیص دهیم که اگر هیچ خطایی وجود نداشته باشد، باید همگی یکسان و برابر با کاراکتر هدف باشند. بدیهی است به دلیل وجود نویز و خطاهای عملیاتی، تمامی این ۱۵ حاصل یکسان نخواهند بود، امّا انتظار داریم درصد بالایی از آن‌ها یکسان بوده و نتیجه‌ی مطلوب را در اختیار ما قرار بدهند. با اتمام این فرایند، با تغییر محلّ نگاه فرد مورد آزمایش، به سراغ حرف بعدی می‌رویم.

دو پارادایم مختلف برای آزمایش (چگونگی روشن و خاموش شدن کاراکترها روی صفحه) وجود دارد که آزمایش هر یک از افراد، توسّط یکی از این دو روش انجام می‌شود.

۱. Single Character Paradigm
کاراکترها یک به یک روشن و خاموش می‌شوند؛ بنابراین در هر لحظه از زمان، حدّاکثر یک کاراکتر بر روی صفحه‌ی نمایش روشن است.

۲. Row-Column Paradigm
هر بار یک سطر یا یک ستون روشن و خاموش می‌شود و در نتیجه هر حرف از تقاطع یک سطر و یک ستون به دست می‌آید. (به عنوان مثال، شکل ۲ نمونه‌ای از این روش است.)

در توضیح دقیق‌تر آزمایش می‌توان گفت که در روش SC paradigm برای تشخیص یک حرف، تمامی حروف ۱۵ بار روشن می‌شوند و در روش RC paradigm هر سطر و هر ستون ۱۵ بار روشن می‌شوند.
شاید در نگاه اوّل به نظر برسد که روش اوّل بهتر است و نیازی به روش دوم نیست، امّا مشاهده نشان می‌دهد که ممکن است روش دوم به دقّت بالاتری دست یابد.

بررسی دیتاست و پردازش‌های اوّلیه

در دیتاست موجود می‌توان با استفاده از سطر ۱۰ام برای هر فرد، تشخیص داد که چه سطر، ستون و یا حرفی روشن شده است.

- با استفاده از داده‌های سطر ۱۰ام هر فرد، تشخیص دهید که هر فرد با کدام یکی از paradigmها (SC یا RC) آزمایش را انجام داده است و در گزارش ذکر کنید.

فرد در این آزمایش می‌خواهد یک کلمه را با استفاده از روش‌های بالا بگوید. تمامی داده‌های یادگیری (train) با استفاده از کلمه‌ی LOKUS اخذ شده‌اند. هر وقت که خود کاراکتر (SC) و یا سطر یا ستون حاوی کاراکتر (RC) روشن شود، در سطر ۱۱ام دیتاست عدد ۱ ذخیره شده‌است. با توجه به این نکات به سوال زیر پاسخ دهید:

- با توجّه به این که کلمه‌ی یادگیری LOKUS است، نحوه‌ی شماره‌گذاری سطرها و ستون‌ها را برای RC paradigm و شماره‌گذاری کاراکترها را برای SC paradigm تشخیص دهید و در گزارش کار ذکر کنید. (یعنی باید بگویید هر یک از حروف صفحه نمایش، چگونه در سطر دهم دیتاست ذکر شده است.)

توجّه: برای جلوگیری از زیاد شدن متغیّرها و شلوغی، تمامی داده‌های مربوط به هر یک از افراد را در یک استراکت تحت عنوان `subject` ذخیره نمایید و فیلدها را با نام‌گذاری مناسب اضافه کنید.

- با استفاده از داده‌های آزمایش تابعی تحت عنوان `IndexExtraction` بنویسید که با ورودی گرفتن استراکت کلی، اندیس زمان نمایش هر حرف (برای پارادایم SC) یا سطر و یا ستون (برای پارادایم RC) را برگرداند. در خروجی باید یک فیلد جدید تحت عنوان `time` به استراکت ورودی اضافه شود که در آن به تفکیک `target` یا `not-target` بودن، اندیس لحظه‌ی روشن‌شدن سطر یا ستون و یا کاراکتر متناظر برای هر دو سری داده‌ی `train` و `test` آمده‌باشد. منظور از `target` این است که در اثر این روشن شدن، همان کاراکتر موردنظر فرد روشن شده‌است. البته تفکیک بین `target` و غیر آن برای داده‌های `test` میسر نیست. در واقع قرار است تنها داده‌های غیر صفر سطر دهم را انتخاب کنید و اندیس آن‌ها را برگردانید و سپس با استفاده از سطر یازدهم هم بین `target` و `not-target` تمایز قائل شوید.

با آشنایی که اکنون از آزمایش دارید می‌توان به سراغ پیاده‌سازی الگوریتم رفت. با توجه به نکات فاز قبل و قسمت قبل، داده‌ها را فیلتر کنید و سپس با استفاده از تابع `epoching` که در فاز قبل آن را نوشتید و با استفاده از فیلد `time`، پنجره‌بندی مناسب را انجام دهید. در پنجره‌بندی می‌توانید جهت سهولت کار در ادامه، برای سری داده‌های `train`، داده‌های `target` و `non-target` را در دو ماتریس جدا از هم بگذارید.

استخراج ویژگی

در این تمرین، هدف ابتدایی، تمییز دادن داده‌های target و non-target است. برای این کار از روش‌های یادگیری ماشین بهره می‌بریم. برای استفاده از الگوریتم‌های یادگیری ماشین لازم است که ویژگی‌های متمایزکننده بین گروه‌های مختلفی که می‌خواهیم آن‌ها را از یکدیگر جدا کنیم پیدا کنیم. این کار به خودی خود کار ساده‌ای نیست و انتخاب ویژگی‌های مناسب، کار بسیار دشواری است. در واقع یافتن ویژگی‌های مناسب، تا حدّی وابسته به دانش پیشین در حوزه‌ای است که از الگوریتم‌های یادگیری در آن استفاده می‌کنیم.

یک روش که گاهی در مطالعات مبتنی بر یادگیری ماشین انجام می شود این است که در ابتدا، تعداد زیادی از ویژگی‌هایی که «به نظر» مناسب هستند را استخراج می‌کنیم سپس با استفاده از معیاری، ویژگی‌هایی را که برای تمییز دادن گروه‌های مختلف مناسب اند بیابیم. در ادامه به موضوع انتخاب ویژگی از بین تعداد زیادی ویژگی باز خواهیم گشت.

در این مرحله به استخراج ویژگی خواهیم پراخت. این قسمت از پروژه مهم‌ترین قسمت برای جواب گرفتن در الگوریتم نهایی است پس شایسته است که برای آن وقت کافی بگذارید و ویژگی‌های مختلف را که در اینجا ذکر شده و «یا نشده‌اند» را یافته و به درستی استخراج کنید. در بخش بعدی پروژه، قرار است که برخی ویژگی‌ها را از بین ویژگی‌هایی که در این بخش استخراج می کنید انتخاب کنید.

در کل می‌توان ویژگی‌ها را به سه گروه تقسیم کرد:

۱. ویژگی‌های حوزه زمان:
میانگین، واریانس، خود سیگنال زمانی، هیستوگرام دامنه، همبستگی کانال‌ها و یا سیگنال‌ها و ...

۲. ویژگی‌های حوزه فرکانس:
انرژی باندهای فرکانسی، فرکانس مد، فرکانس میانگین، فرکانس میانگین، تبدیل سینوسی و کسینوسی گسسته و ...

۳. ویژگی‌های زمان ـ فرکانس:
تبدیل‌های Wavelet، تبدیل STFT و ...

ویژگی‌های فراوان دیگری نیز وجود دارند که با جست‌جو در اینترنت می‌توانید آن‌ها را بیابید. برای استخراج ویژگی‌های بالا می‌توانید از هر تابع آماده‌ای که موجود باشد استفاده کنید.
توجّه مهم از هر ویژگی که استفاده کنید بایستی در گزارش کار آن را ذکر کنید و در مورد آن توضیح بدهید. به خصوص در مورد ویژگی‌های غیر بدیهی و ویژگی‌هایی که خود با جست‌وجو در اینترنت آن‌ها را یافته‌اید. برای مثال اگر از تبدیل Wavelet استفاده می‌کنید لازم است حتماً در مورد آن توضیحات کافی را بدهید و بگویید که چرا ممکن است مفید واقع شود. البته لازم نیست که تئوری کامل آن را توضیح دهید اما بایستی توضیحات شما کافی باشند. در صورتی که این بند را رعایت نکنید نمره‌ای به شما تعلق نخواهد گرفت.
توجّه مهم بخش‌هایی که در آن ویژگی استخراج می‌کنید را به صورت صریح و با کامنت‌گذاری مناسب از یکدیگر جدا کنید به صورتی که مشخص باشد که در هر قسمت کدام ویژگی استخراج می‌شود و به وضوح از قسمت‌های دیگر جدا باشد. عدم رعایت این بند نیز باعث از دست‌رفتن کامل نمره‌ی این بخش از تمرین خواهدشد.

در این آزمایش می‌توانیم که هر trial را به یکی از دو گروه target و non-target تقسیم کنیم. به trialهایی target می‌گوییم که در آن‌ها، حرف مورد نظرمان روشن شده است. از سیگنال زمانی مربوط به هر trial، ویژگی‌های مد نظرتان را استخراج کنید و ویژگی‌های استخراج شده را در یک ماتریس دو بعدی که هر ستون یک ویژگی و هر ردیف آن مربوط به یک trial است، قرار دهید. ویژگی‌ها را به صورت همزمان برای داده‌های train و test استخراج کنید چرا که لازم است که ستون‌های این ماتریس برای داده‌های train و test متناظراً از یک ذات باشند. برای مثال فرض کنید که شما از ویژگی میانگین استفاده کرده‌اید و برای هر trial یک میانگین بدست آورده‌اید. فرض کنید که در مجموع ده trial داریم، پس در سطر مربوط به هر trial از ماتریس مذکور، در ستون متناظر با ویژگی میانگین، عدد میانگین آن trial را قرار می‌دهیم. در این مثال، ماتریس شما ده ردیف خواهد داشت و تعداد ستون‌های آن به تعداد ویژگی‌های انتخابیتان خواهد بود.

در ماتریس ویژگی‌ها برای داده‌های test نیز باید در ستون متناظر با ویژگی میانگین، میانگین را برای trialهای داده‌های test

قرار دهیم. توجه کنید که لزومی برای برابر بودن تعداد سطرهای ماتریس ویژگی‌ها برای داده‌های test و داده‌های train وجود ندارد چرا که دلیلی وجود ندارد که تعداد trialهای دو آزمایش برابر باشند.

ویژگی میانگین از آن دسته ویژگی‌هایی است که برای هر trial یک عدد خروجی می‌دهد. برخی ویژگی ها برای هر trial چندین خروجی می‌دهند که در این‌صورت هر کدام از این خروجی‌ها را به عنوان یک ویژگی متمایز درنظر میگیریم و مراحل بالا را برای آن تکرار می‌کنیم. یک مثال ساده، ویژگی سیگنال زمانی است که در آن می‌توان هر درایه از سیگنال زمانی را به عنوان یک ویژگی در نظر گرفت (مثلاً نمونه پنجم سیگنال زمانی یک trial یک ویژگی آن trial است.)

اکنون که استخراج ویژگی پایان یافته است بایستی از بین ویژگی‌های استخراج شده، ویژگی‌های مناسب را پیدا کنیم. برای این کار می‌توان از آزمون فرضیه‌ی ANOVA استفاده کرد. اگر بخواهیم به زبان ساده آن را توضیح دهیم، این آزمون به بررسی این می‌پردازد که برای یک ویژگی خاص، میانگین این ویژگی برای گروه‌های مختلف در چه حد از یکدیگر متمایز هستند و همچنین واریانس این ویژگی در داخل یک گروه چه مقدار کم است. برای یک ویژگی هر چقدر این تمایز و فاصله‌ی بین میانگین‌ها بیشتر باشد، و واریانس در داخل هر گروه کمتر، این ویژگی جهت تمییز بین داده‌ها مناسب‌تر است. یک فرمولاسیون برای این کار را می‌توان در عبارت زیر دید:

$$J = \frac{|\mu_0 - \mu_1| + |\mu_0 - \mu_2|}{\sigma_1^2 + \sigma_2^2} \quad (۵)$$

که در عبارت بالا μ_1 و μ_2 میانگین دو گروهی هستند که بررسی می‌کنیم و μ_0 میانگین کل داده‌های همین ویژگی است (برخلاف این آزمایش که تنها دو حالت target و non-target داریم، ممکن است در آزمایش‌هایی تعداد این گروه‌ها بیشتر باشند که در این صورت بایستی دو به دو بین تمام گروه‌ها عبارت بالا را محاسبه کنیم که μ_0 میانگین تمامی گروه‌ها است). σ_1^2 برابر واریانس ویژگی در trialهای دسته‌ی اول و σ_2^2 برابر واریانس ویژگی در trialهای دسته‌ی دوم است. توجه کنید که به با توجه به رابطه‌ی فوق، برای هر ویژگی می‌توان یک مقدار J محاسبه کنید که این مقدار هر چقدر بزرگتر باشد، ویژگی مد نظر، مناسب‌تر است.

عبارت بالا تنها یک برداشت از آزمون ANOVA است و خود آزمون به صورت آماری به بررسی تفاوت میانگین‌ها می‌پردازد. برای پروژه می‌توانید عبارت ۴ را پیاده‌سازی کنید و یا می‌توانید از تابع `anova1` متلب استفاده کنید.
پس از اعمال ANOVA و یا محاسبه عبارت ۴ برای تمام ویژگی‌ها بایستی از خروجی‌های موردنظر آن‌هایی را که در تمییز بهتر هستند را انتخاب کنیم که این کار را می‌توان با استفاده از یک روش حد آستانه و انتخاب ویژگی‌هایی که بیشترین تمییز را بین داده‌های target و non-target قائل هستند انتخاب کنیم.

- علت اینکه ما از تمامی ویژگی‌های استخراج شده استفاده نمی‌کنیم و سعی در آن داریم که تعداد ویژگی‌ها را کاهش دهیم، پدیده‌ای تحت عنوان overfit شدن الگوریتم است. با مطالعه در اینترنت توضیحات کافی در مورد آن را در گزارش بیاورید.
- معیاری برای انتخاب حد آستانه‌ی ذکر شده ارائه دهید.
- با توجه به حد آستانه‌ای که در بالا انتخاب کرده‌اید، تعدادی ویژگی در خروجی به عنوان ویژگی‌های خوب انتخاب می‌شوند. کدام دسته ویژگی‌ها بیشتر در انتخاب نهایی شما به عنوان ویژگی‌های مناسب انتخاب شده‌اند؟ لازم نیست که یک به یک تمام ویژگی‌های موجود در انتخاب نهایی را ذکر کنید بلکه فقط اینکه از آن دسته از ویژگی‌ها که خود برای هر trial چندین خروجی می‌دهند، آیا ویژگی‌هایی وجود دارند که تعداد زیادی از ویژگی‌هایشان در انتخاب نهایی ظاهر شده‌باشند؟

Cross Validation

فرض کنید که شما تنها یک سری داده از آزمایشی را در دست دارید و با استفاده از آن‌ها یک مدل یادگیری ماشین را آموزش داده‌اید. اکنون شما از تمامی داده‌های موجود خود استفاده کرده‌اید. بدون داده‌گیری جدید چگونه می‌توانید از صحت کارکرد الگوریتم بالا اطمینان پیدا کنید؟ جواب این سوال واضح است! در ابتدا به جای اینکه از تمامی داده‌ها جهت آموزش الگوریتم استفاده کنیم، یک قسمت از آن را کنار می‌گذاریم و در نهایت پس از یادگیری ماشین، با داده‌های کنارگذاشته‌شده، درصد صحت را تست می‌کنیم.
برای بررسی درصد صحت کارکرد الگوریتم‌های یادگیری ماشین از تکنیکی تحت عنوان cross validation استفاده می‌شود. انواع مختلفی از این تکنیک را می‌توان تعریف کرد که یک نوع مناسب آن، Holdout Validation است که در آن مطابق توضیحات بالا، یک قسمت از داده‌های آزمایش را از ابتدا کنار می‌گذاریم و در آخر برای بررسی صحت کارکرد الگوریتم از آن‌ها استفاده می‌کنیم و در صورتی الگوریتم را بر روی داده تست (test) اجرا می‌کنیم که در این قسمت درصد صحت قابل توجهی گرفته‌باشیم.
توجه شود که اگرچه در توضیح پروژه، این قسمت بعد از قسمت **استخراج ویژگی** آمده‌است اما برای صحت کامل مراحل و جلوگیری از بایاس در آموزش الگوریتم، کنار گذاشتن چند trial را بهتر است قبل از اعمال آستانه بر خروجی‌های تست ANOVA انجام داد. اهمیت اجرای این قسمت در حالتی که آستانه شما ثابت نیست و خود به خروجی‌های تست ANOVA ربط دارد بیشتر است.

پیاده‌سازی الگوریتم یادگیری ماشین

با توجه به قسمت‌های قبل اکنون می‌توان به فرآیند یادگیری پرداخت. بدین منظور لازم است که ابتدا یک مدل را بر اساس یک الگوریتم آموزش دهیم و سپس با استفاده از آن، داده‌های جدید را لیبل بگذاریم. داده‌های آموزش هر فرد را در ماتریس `Train_Features` قرار دهید که در این ماتریس، هر ستون یک ویژگی و هر درایه، یک تحقق از آن ویژگی است. بدیهی است که ساختار این ماتریس برای داده‌های تست نیز باید دقیقا مشابه باشد یعنی به طور مثال اگر ستون اول میانگین داده‌های هر پنجره است، برای داده‌های تست نیز این ستون بایستی همین ویژگی باشد. در واقع می‌توان بدین شکل فرض کرد که ستون‌ها بردارهایی هستند که فضای شما را می‌سازند پس باید برای داده‌های تست نیز در هر بعد داده‌ی متناظر با آن بعد را قرار داد. هر سطر از ماتریس بالا، یا متناظر با فعالیت `target` است و یا `non-target`. در یک بردار، برای هر سطر که متناظر با یک سطر `target` است، ۱ و در غیر این‌صورت، صفر بگذارید. با رعایت فرمت بالا اکنون می‌توانید مدل را آموزش دهید. برای این کار می‌توانید از الگوریتم های مختلف موجود استفاده کنید. دو الگوریتم پرکاربرد SVM و LDA هستند. این دو با استفاده از توابع `fitcsvm` و `fitcdiscr` پیاده‌سازی می‌شوند. خروجی این توابع یک *مدل* آموزش دیده از همان الگوریتم است. مدل هر فرد را در استراکت کلی تحت همان فرد ذخیره کنید.
در ادامه می‌توان با استفاده از تابع `predict` داده‌های جدید را طبقه‌بندی کرد. نحوه‌ی کار با تابع `predict` راحت است و به عنوان ورود *مدل* آموزش‌یافته و ماتریس ویژگی داده‌هایی که خواستار طبقه‌بندی آن‌ها هستیم را می‌خواهد و در خروجی برچسب هر سطر را برمی‌گرداند.

- برای هر فرد مدلی بر اساس هر دو روش یادگیری ماشین را آموزش دهید.
- اگر به عنوان ورودی به تابع `predict` همان ماتریس `Train_Features` را بدهیم، انتظار دارید که چه درصد صحتی بگیرید؟ این مورد را بررسی کنید و در گزارش، خروجی که در واقعیت دریافت می‌شود را ذکر کنید. علت این امر چیست؟
- اگر در این مرحله با مقایسه صفر و یکی تشخیص target و non-target بودن درصد صحت بالایی می‌گیرید بایستی بدین توجه کنید که با توجه به اینکه تعداد trialهای non-target بسیار بیشتر از حالت target است، شما اگر خود بدین صورت بیایید و خروجی‌ها را به صورت دستی همگی را non-target تشخیص دهید، درصد صحت بالایی خواهید گرفت. پس بدین منظور لازم است که تابع هزینه شما وزن بیشتری برای تشخیص اشتباه داده‌های target به عنوان non-target داشته باشد. بدین منظور تابع هزینه خود را مناسب تعریف کنید.
- اگرچه تا کنون یک مدل را آموزش داده‌اید و بر حسب آن `target` و غیر آن را از یکدیگر تشخیص داده‌اید اما هنوز کلمه‌ای را تشخیص نداده‌اید. برای این منظور تابعی بنویسید که با ورودی گرفتن خروجی‌های فرایند `predict`، کلمه‌ای را که فرد در صدد گفتن آن بوده‌است را تشخیص بدهد.
- خروجی را برای داده‌های تست هر فرد بیابید و تشخیص بدهید که هر فرد در صدد گفتن چه کلمه‌ای بوده‌است. کلمه‌ی موردنظر در داده‌گیری test هم ۵ حرفی بوده‌است.