گزارش فاز دوم پروژه درس سیگنال ها و سیستم ها

دکتر کربلایی‌زاده

پرهام محمدی 96102342

علی محرابیان 96102331

طراحی فیلتر

می دانیم که در حوزه فرکانس ، تبدیل فوریه ورودی در تبدیل فوریه پاسخ ضربه سیستم ضرب می‌شود و تبدیل فوریه تابع خروجی بدست می‌آید.

پس از ورودی تبدیل فوریه می‌گیریم و در تابع تبدیل سیستم ضرب می‌کنیم:

$$X(w) = \pi\big(\delta(\omega-\omega_0)+\delta(\omega+\omega_0)\big)+\pi\big(\delta(\omega-2\omega_0)+\delta(\omega+2\omega_0)\big)$$

$$Y(\omega) = H(\omega)\times X(\omega) = \ e^{-\frac{\pi}{3}sgn(\omega)}\times X(\omega)$$

$$Y(w) = \ \pi e^{-\frac{\pi}{3}}\big(\delta(\omega-\omega_0)+\delta(\omega-2\omega_0)\big)+\pi e^{\frac{\pi}{3}}\big(\delta(\omega+\omega_0)+\delta(\omega+2\omega_0)\big)$$

$$\rightarrow y(t) = \ \frac{1}{2}\left(e^{\left(j\omega_0 t-\frac{\pi}{3}\right)}-e^{-\left(j\omega_0 t-\frac{\pi}{3}\right)}\right)+\frac{1}{2}\left(e^{\left(j2\omega_0 t-\frac{\pi}{3}\right)}-e^{-\left(j2\omega_0 t-\frac{\pi}{3}\right)}\right)$$

حال از این عبارت ، تبدیل فوریه وارون می‌گیریم:

$$\rightarrow y(t) = \cos\left(\omega_0 t-\frac{\pi}{3}\right)+\cos\left(2\omega_0 t-\frac{\pi}{3}\right)$$

$$\rightarrow y(t) = \cos\left(\omega_0\left(t-\frac{\pi}{3\omega_0}\right)\right)+\cos\left(2\omega_0\left(t-\frac{\pi}{6\omega_0}\right)\right)$$

همانطور که مشاهده می‌کنیم ، سیگنال اول به اندازه $\frac{\pi}{3\omega_0}$ و سیگنال دوم به اندازه $\frac{\pi}{6\omega_0}$ تاخیر زمانی پیدا کرده‌اند.

$$Y(\omega) = H(\omega)\times X(\omega) = \ e^{-\frac{\pi}{3}\omega}\times X(\omega)$$

طبق خاصیت شیفت زمانی در تبدیل فوریه ، خروجی همان شیفت یافته ورودی به اندازه است.

$$\rightarrow y(t) = x\left(t-\frac{\pi}{3}\right) \rightarrow y(t) = \ \cos\left(\omega_0 t-\frac{\pi}{3}\right)+\cos\left(2\omega_0 t-\frac{2\pi}{3}\right)$$

همان طور که مشاهده می‌کنیم در حالته دوم که فاز تابع تبدیل خطی است ، تاثیر فاز سیستم در ورودی ، صرفا یک شیفت زمانی است.و هنگامی که این فاز خطی باشد ، مقدار شیفت زمانی در همه‌ی فرکانس‌ها عددی ثابت است.

پس صرفا یک تاخیر در سیگنال ظاهر می‌شود که به سادگی سیگنال اولیه قابل بازسازی است.

اما اگر فاز فیلتر خطی نباشد ، مقدار شیفت زمانی در فرکانس های مختلف ، متفاوت است و فرم و شکل سیگنال فیلتر شده دچار دگرگونی شده و در آن اعوجاج پیش می‌آید.

حال درستی گزاره گفته شده را برای دو سیستم معرفی شده در پرسش قبل ، تحقیق می‌کنیم :

$$gd(\omega) = -\frac{d\emptyset(\omega)}{d\omega} = -\frac{2\pi}{3}\delta(\omega)$$

برای سیستم اول , فاز سیستم ثابت است.تاخیر گروه برای محاسبه میزان تاخیر در هر محتوای فرکانسی ، فاز تابع تبدیل را با یک خط تقریب می‌زند.بنابراین تاخیر گروه به نوعی تاخیر پوش سیگنال می‌باشد. و چون در سیستم اول پوش سیگنال ثابت است ، تاخیر پوش نداریم. و همان میزان تاخیری که با اثر دادن تابع تبدیل سیستم در حوزه فرکانس بدست آمد نشان دهنده اثر تاخیر گروه دد فازهای غیرخطی می‌باشد.

$$gd(\omega) = -\frac{d\emptyset(\omega)}{d\omega} = -\frac{\pi}{3}$$

همان طور که مشاهده می‌کنیم در سیستم دوم که فاز سیستم خطی است ، تاخیر گروه در همه‌ی فرکانس‌ها ، همان شیب خط موردنظر یعنی می‌باشد و همان طور که بررسی کردیم میزان شیفت سیگنال زمانی نیز مقدار ثابت در همه‌ی فرکانس‌ها بود.

پس در سیستم های با فاز خطی ، تاخیر گروه در همه‌ی فرکانس ها یک عدد ثابت و میزان شیفت سیگنال زمانی در همه‌ی فرکانس‌ها آن عدد ثابت می‌باشد.

حال به اثبات رابطه‌ی $gd(w) = Re\left\{\frac{j\frac{d}{dw}H(w)}{H(w)}\right\}$ می‌پردازیم :

$$Re\left\{j\frac{d}{dw}H(w)\right\} = Re\left\{j\frac{d}{dw}\left(A(w)e^{j\emptyset(w)}\right)\right\} =$$

$$Re\left\{j\left(A'(w)e^{j\emptyset(w)} + A(w)j\emptyset'(w)e^{j\emptyset(w)}\right)\right\} = Re\left\{-A(w)\emptyset'(w)e^{j\emptyset(w)} + jA'(w)\right\}$$

$$= \frac{-A(w)\emptyset'(w)e^{j\emptyset w}}{A(w)e^{j\emptyset w}} = -\emptyset'(w) \rightarrow \quad gd(w) = Re\left\{j\frac{d}{dw}H(w)\right\}$$

برای پیاده‌سازی تابع groupdelay ، از خاصیت مشتق در حوزه فرکانس استفاده می‌کنیم :

$$n \times h[n] \quad \leftrightarrow \quad j\frac{d}{d\omega}H(\omega)$$

در نتیجه برای ساخت صورت عبارت 4 ، از پاسخ ضربه زمانی با دستور fft تبدیل فوریه می‌گیریم و برای ساخت مخرج از n×h[n] تبدیل فوریه می‌گیریم.

سپس دو بردار را به صورت elementwise بر هم تقسیم می‌کنیم و قسمت حقیقی آن‌را بدست می‌آوریم که همان تاخیر گروه می‌باشد.

در این دو fft ، DFT که محاسبه می‌کنیم ، N نقطه‌ای در نظر می‌گیریم.

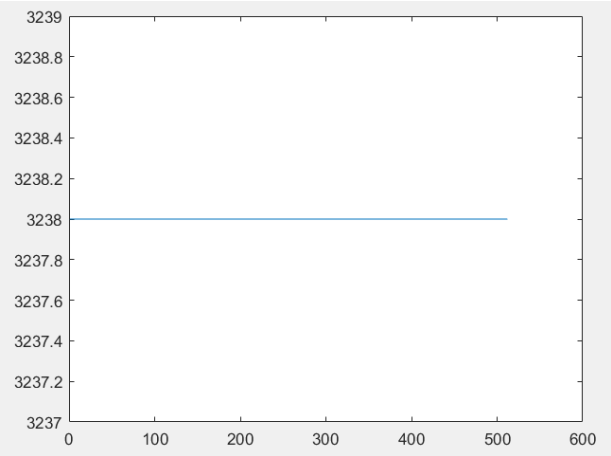
هر چه N بزرگتر باشد ، دقت محاسبات بیشتر است زیرا هر چه تعداد نقاط تابع تبدیل سیستم بیشتر باشد ، فیلتر به حالت ایده‌آل نزدیک‌تر می‌شود.

همچنین اگر N از تعداد نمونه‌ها کمتر باشد ، خروجی نهایی نادرست می‌باشد و به اشتباه بازسازی می‌شود.

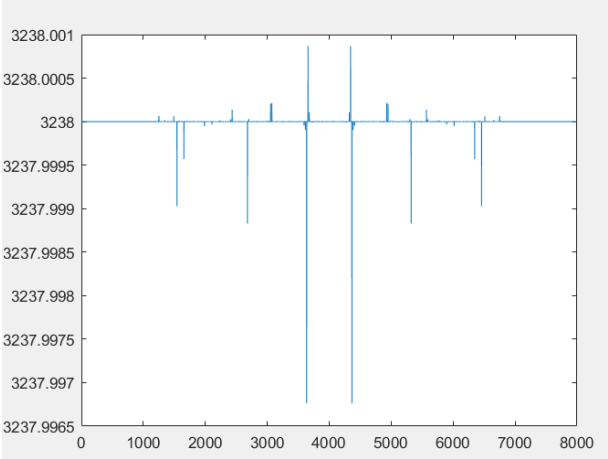
قطه کد مربوطه به صورت زیر است :

```
function gd = groupdelay(h,N)
n = [0:length(h)-1];
h2 = n .* h;
gd = real(fft(h2,N) ./ fft(h,N));
```

Groupdelay را بوسیله تابعی که پیاده‌سازی کردیم و همچنین تابع grpdelay در متلب بدست آوردیم :
همان طور که مشاهده می‌شود ، تابع پیاده‌سازی شده به خوبی تاخیر گروه را بدست آورده است.



With grpdelay



With groupdelay

(Our function)

برای استفاده از تابع پیاده‌سازی شده ، باید 3 مسئله را در نظرگرفت :

1 - داده ها NaN نباشند.

2 - داده‌ها Inf+ یا Inf- نباشند.

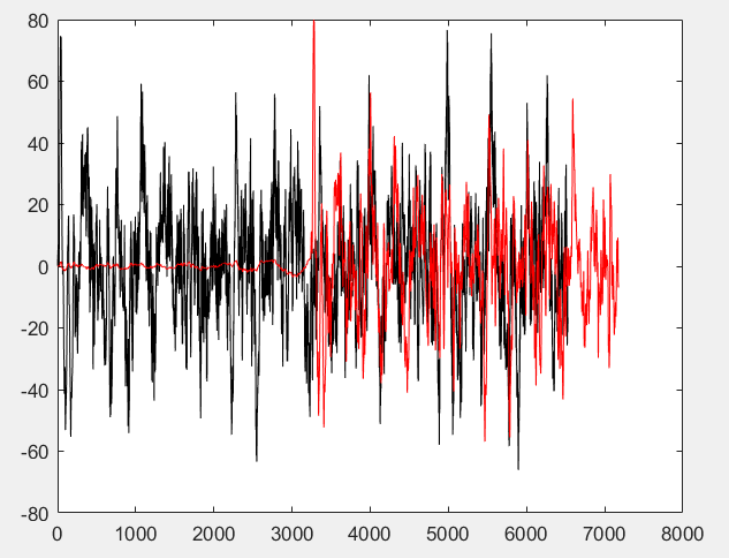
3 - داده از یک حدی مانند 10^5 بیشتر نباشند.

در صورتی که یکی از درایه های بردار gd یکی از ویژگی‌های بالا را نداشته باشد ، باید آنرا با یکی از درایه های مناسب در gd جایگزین کنیم.

در تابع zphasefilter ، ابتدا به کمک دستور filter و پاسخ ضربه داده شده ، سیگنال فیلتر شده را بدست می‌آوریم.

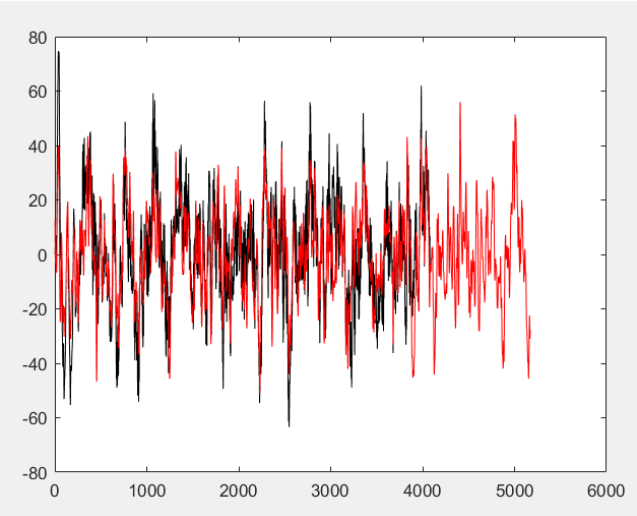
```
yy = filter(h,1,x);
```

در این سیگنال همان طور که مشاهده می‌شود ، تاخیر ایجاد می‌شود.



حال برای رفع این تاخیر ایجاد شده ، ابتدا پاسخ ضربه فیلتر را به تابع groupdelay داده و تاخیر گروه را بدست می‌آوریم.چون مقدار آن تقریبا ثابت است ، میانگین آن‌را بعنوان مقدار شیف زمانی در نظر می‌گیریم و با استفاده از جمله زیر ، این تاخیر را برطرف می‌کنیم :

```
y = yy(gd_valid + 1 : end);
```



سیگنال فیلترشده به درستی تاخیر آن برطرف شده است :

همان‌طور که می‌دانیم در سیگنال‌های حقیقی قسمت موهومی تبدیل فوریه ناشی از قسمت فرد سیگنال و قسمت حقیقی تبدیل فوریه ناشی از قسمت زوج سیگنال است.

پس اگر تبدیل فوریه یک سیگنال حقیقی , حقیقی باشد ، قسمت فرد آن صفر است و سیگنال زوج می‌باشد.

چون سیگنال علی است به ازای $t < 0$ صفر است و چون زوج است ، به ازای $t > 0$ نیز صفر است.

پس فقط می‌تواند در صفر مقدار داشته باشد و از جنس $\delta(t)$ باشد و فقط یک سیستم از نوع gain می‌تواند اینگونه باشد.

بررسی دیتاست و پردازش‌های اولیه

چون جدول کاراکترها یک جدول 6×6 است ، در روش SC ، 36 حالت مختلف و در روش RC ، 12 حالت مختلف (6 سطر و 6 ستون) وجود دارد.

حال کافی است سطر 10 هر یک از DataSet هارا بررسی کنیم ، اگر مقادیر غیر صفر از 1 تا 12 بود آزمایش به صورت RC و اگر از 1 تا 36 بود ، آزمایش به صورت SC انجام شده است.

نتیجه آزمایش به صورت زیر بدست آمد :

Dataset 9	Dataset 8	Dataset 7	Dataset 6	Dataset 5	Dataset 4	Dataset 3	Dataset 2	Dataset 1
RC	RC	RC	RC	RC	RC	RC	SC	SC

شخص می‌خواهد 5 حرف را بگوید.در روش SC هر کاراکتر 15 بار و در روش RC هر سطر یا ستون 15 بار روشن می‌شود.در حروف Target ، سطر 11ام یک می‌شود. یعنی ابتدا هنگام روشن شدن کاراکتر مربوط به حرف Target در صفحه موردنظر ، سطر 11ام یک می‌شود.

در روش RC ، در هریک از 5 دفعه ، در هنگام روشن شدن سطر و ستون مربوط به حرف دلخواه ، Target

می‌شوند یعنی سطر 11ام در این مکان‌ها 1 می‌شود.مثلا در این مثال ، در سری اول ، هنگام تحریک سطر و همچنین تحریک ستون مربوط به حرف L ، سطر 11 یک می‌شود.

پس اینکه هرکدام از حروف مربوط به کلمه LUKAS مربوط به کدام کاراکتر و کدام سطر و ستون است ، استخراج می‌شود.

بر مبنای این شماره‌های کاراکترها و سطر و ستون‌ها به شرح زیر بدست می‌آید :

شماره سطر و ستون‌ها	1	2	3	4	5	6
7	1	2	3	4	5	6
8	7	8	9	10	11	12
9	13	14	15	16	17	18
10	19	20	21	22	23	24
11	25	26	27	28	29	30
12	31	32	33	34	35	36

همان طور که مشاهده می‌شود ، کاراکترها از چپ به راست و از بالا به پایین شماره‌گذاری می‌شوند.

شماره ستون‌ها به ترتیب از چپ به راست 1 تا 6 و شماره سطرها به ترتیب از بالا به پایین 7 تا 12 است.

در این مرحله به epoch کردن داده‌ها و فیلتر کردن آن‌ها می‌پردازیم.

برای فیلتر کردن داده‌ها ابتدا به کمک filterDesigner یک فیلتر میان‌گذر که فرکانس‌های بین 0.5 تا 39.5 هرتز را عبور می‌دهد ، می‌سازیم.سپس برای اینکه این فیلتر بر روی سیگنال تاخیر گروه ایجاد نکند ، فیلتر را با دستور filtfilt به تمامی سیگنال‌های subjectdata اعمال می‌کنیم.

برای اینکه داده‌ها را epoch کنیم در تابع IndexExtraction اطلاعات مربوط به آن‌ها از جمله target و non-target برای trainها و time را تحت عنوان یک field در یک struct ذخیره می‌کنیم.

استخراج ویژگی‌ها

در این بخش ، ویژگی‌هایی از سیگنال EEG که ممکن است برای طبقه‌بندی دیتا مفید باشد را بدست آوردیم :

- واریانس
- فرکانس میانگین

با استفاده از تابع meanfreq محاسبه شده است.

- فرکانس میانه

با استفاده از تابع medfreq محاسبه شده است.

- .ضرایب تبدیل گسسته سینوسی داده های هر کانال (DST)

$$y(k) = \sum_{n=1}^{N} x(n)\sin\left(\pi \frac{kn}{N+1}\right), \quad k = 1, \ldots, N.$$

- .ضرایب تبدیل گسسته کسینوسی دادههای هر کانال(DCT)

$$y(k) = w(k)\sum_{n=1}^{N} x(n)\cos\left(\frac{\pi}{2N}(2n-1)(k-1)\right), \quad k = 1, 2, \ldots, N,$$

where

$$w(k) = \begin{cases} \frac{1}{\sqrt{N}}, & k = 1, \\ \sqrt{\frac{2}{N}}, & 2 \le k \le N, \end{cases}$$

برای محاسبه این دو ویژگی ، از توابع آماده dst و sct استفاده می‌کنیم.

- .انرژی سیگنال در 4 باند فرکانسی

انرژی موجود در 4 باند فرکانسی متعارف را (با تعاریف ارائه شده در صورت تمرین) محاسبه کردیم. برای این کار، ابتداسیگنالها را فیلتر کردیم و باندهای فرکانسی را به تفکیک به دست آوردیم، سپس انرژی موجود در هر کدام از سیگنالهای فیلترشده را محاسبه کردیم.

برای محاسبه این ویژگی ، از تابع Freqband که ضمیمه شده است ، استفاده می‌کنیم.

- تبدیل فوریه زمان کوتاه ((Short-Time Fourier Transform - STFT))

تبدیل فوریه زمان کوتاه، ابزاری است برای برقراری ارتباط بین حوزه زمان و فرکانسکه در آن، محتوای فرکانسی را در طول زمان بررسی میکنیم. در واقع، این تبدیل، سیگنال را به پنجره های زمانی کوچک تقسیم میکند و در هر کدام از این پنجرههای زمانی، به محاسبه تبدیل فوریه میپردازد. بنابراین، میتوان تغییرات انرژی در فرکانسهای مختلف را در طول زمان بررسی کرد.

برای محاسبهی STFT در متلب، از تابع spectrogram استفاده میشود. در این تابع، پنجره زمانی و میزان همپوشانی پنجره های زمانی، و نیز تعداد نقاط مورد استفاده در محاسبه fft را مشخص میکنیم.

ضمنا به دلیل این که توابع مربوط به طبقه بندی موجود در متلب (svm)قادر به پردازش دادههای مختلط نبودند، (بدیهی استSTFT مقادیری مختلط دارد) و نیز به دلیل این که مقدار انرژی موجود در فرکانسهای مختلف برای ما مورد توجه است،اندازه تبدیل فوریه زمان کوتاه را به عنوان ویژگی در نظر گرفتیم.

- تبدیل Wavelet

در برابر STFTکه دارای پنجره طول ثابت است موجک دارای پنجره طول متغیر است

$$wt(s,\tau) = \langle x, \psi_{s,\tau} \rangle = \frac{1}{\sqrt{s}} \int_{-\infty}^{\infty} x(t)\psi^* \left(\frac{t-\tau}{s}\right) dt$$

با استفاده از یک پنجره با طول متغیر می توان بر مشکل از پیش تعیین کردن رزولوشن غلبه کرد.
پنجره‌های با طول متغیر برای فرکانس‌هاس مختلف استفاده می‌شوند.
آنالیز فرکانس‌های بالا : استفاده از پنجره‌های باریکتر برای رزولوشن زمانی بهتر
آنالیز فرکانس‌های پایین : استفاده از پنجره‌های عریض برای رزولوشن فرکانسی بهتر

در مقایسه با تبدیل فوریه می‌توان گفت که تبدیل wavelet دارای خصوصیت محلی‌سازی بسیار خوبی است. بطور مثال تبدیل فوریه یک پیک تیز دارای تعداد زیادی ضریب است، چرا که توابع

پایه تبدیل فوریه توابع سینوسی و کسینوسی هستند که دامنه آن‌ها در کل بازه ثابت است، در حالی که توابع Wavelet توابعی هستند که بیشتر انرژی آن‌ها در بازه کوچکی متمرکز شده‌است و به سرعت میرا می‌شوند. بنابراین با انتخاب مناسب موجک‌های مادر می‌توان فشرده‌سازی بهتری در مقایسه با تبدیل فوریه انجام داد.

- سیگنال زمانی کانالهای مختلف

در ابتدا ویژگی‌های مذکور را در متلب پیاده سازی کردیم و برای هر trial آنرا بدست آوردیم.در یک ماتریس ، بردار ویژگیهای هر trial را به ترتیب زمانی در یک سطر قرار دادیم.و به این ترتیب ماتریس ویژگی را هم برای train و هم برای test بدست می‌آوریم :

در قسمت بعدی ، باید J_Value را به ازای تمام ویژگی‌ها محاسبه کنیم.

در انتهای قسمت گفته شده است که عملیات باید قبل از محاسبه انجام شود.در نتیجه ابتدا از ماتریس ویژگی‌هارا در نظر می‌گیریم.سپس سطرهای مربوط به trial های target و trial های non-tarhet را جدا می‌کنیم.سپس به ازای هر ویژگی میانگین و واریانس آن را در سطرهای target و سطرهای non-target به صورت جدا محاسبه می‌کنیم.همچنین میانگین آن ویژگی را در همه سطرها نیز محاسبه می‌کنیم.

سپس J-value را به کمک رابطه‌ی زیر محاسبه می‌کنیم:

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

این عدد ، معیار خوبی برای تعیین این است که سطرهای target و non-target چه قدر از نظر این ویژگی متفاوتند و هر چه مقدار بزرگتری داشته باشد ، این ویژگی بهتر است و به خوبی داده های target و non-target را از هم جدا می‌کند.

میانگین این ویژگی در تمام سطرها	$\mu_0 =$
میانگین این ویژگی در سطرهای Target	$\mu_1 =$
میانگین این ویژگی در سطرهای Non-Target	$\mu_2 =$
واریانس این ویژگی در سطرهای Target	$\sigma_1^2 =$
واریانس این ویژگی در سطرهای Non-Target	$\sigma_2^2 =$

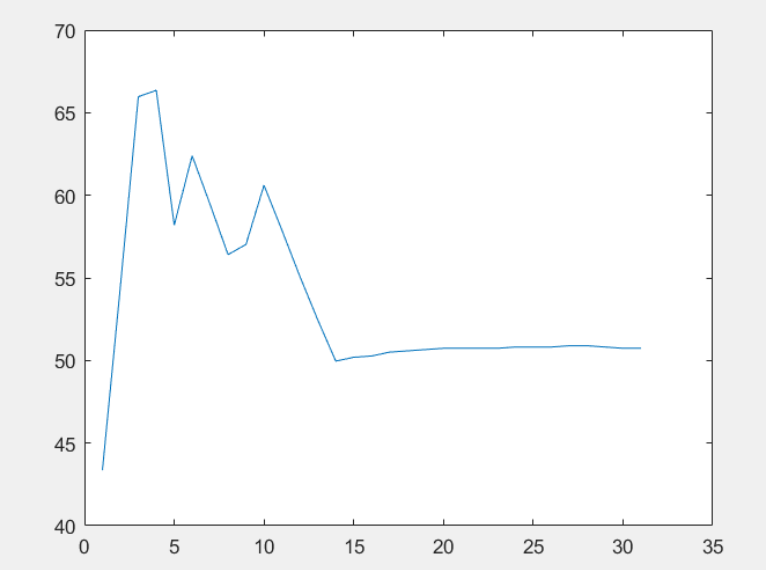
: OverFitting

در ابتدا یکبار الگوریتم یادگیری ماشین را به وسیله کل ماتریس train پیاده‌سازی کردیم و تمام ویژگی‌ها را حفظ کردیم.با وجود دقت بسیار بالا در حدود 97 درصد اما نتیجه برای ماتریس test ، بسیار بد است و این پدیده که به واسطه‌ی تعداد زیاد ویژگی‌ها رخ می‌دهد overfittng نام دارد.

برای رفع این مشکل با روش cross validation تعدادی از ویژگی‌ها با J-value کمتر از یک آستانه‌ی معین را حذف می‌کنیم.

Cross Validation

برای حذف تعداد خوبی از ویژگی ها از روش cross validation استفاده می کنیم. این روش به این صورت است که ابتدا یک آستانه برای j value در نظر می گیریم سپس تعدادی از trial ها مثلا دو سوم از ماتریس train را در نظر می گیریم و الگوریتم یادگیری ماشین را بر مبنای آن و ویژگی های جدید پیاده سازی می کنیم و نتیجه را برای قسمت validation ماتریس train تست می کنیم و مشاهده می کنیم که چند درصد خطا داریم در محاسبه خطا وزن خطای target را 35 برابر در نظر می گیریم سپس در یک حلقه for آستانه j value را مرتبا تغییر داده و در آن آستانه ای که درصد خطا کمینه می شود آن آستانه را به عنوان آستانه ای که در الگوریتم استفاده می شود در نظر می گیریم پس از انجام این حلقه ی for تعداد زیادی از ویژگی های با j value کم حذف می شوند و مساله overfitting رفع می شود .



نمودار بالا نشان می‌دهد که اگر آستانه J-value را (برای حذف ویژگی‌ها) از خیلی زیاد به سمت مقادیر کمتر تغییر دهیم ، درصد دقت ابتدا زیاد شده و سپس کم می‌شود.

پیاده سازی الگوریتم یادگیری ماشین

حال ویژگی های بهینه را بدست اوردیم trial های train را که ویژگی های بهینه آن ها بدست اورده شده است به همراه ماتریس label ها (Y) به تابع fitcsvm می دهیم سپس مدلی که در خروجی می دهد به همراه ماتریس ویژگی های تست را به تابع predict می دهیم و در خروجی label های target

های ان را بدست می اوریم.

حال باید 5 حرف گفته شده توسط فرد را از این label بدست آمده استخراج کنیم به این صورت که یک بردار با دوسطر و 36 ستون در نظر می گیریم درایه سطر اول و ستون i ام نشان می دهد که حرف i ام چند بار در صفحه روشن شده است درایه سطر دوم و ستون i ام نشان می دهد که حرف i ام چند بار target شده است پس از آن که تمام درایه های سطر اول 15 شدند یا به عبارتی trail های مربوط به ان حرف کامل شد چک میکنیم که تعداد target های کدام حرف بیشتر است وآن حرف را به عنوان حرف گفته شده در نظر می گیریم و ماتریس 2 در 36 را ریست کرده و به سراغ حرف بعدی می رویم

در نهایت 5 حرفی را که به صورت یک عدد بین یک تا 36 بدست اوردیم از جدول استخراج می کنیم که هر عدد نمایشگر چه حرفی است و کلمه ی 5 حرفی خود را استخراج می کنیم

در paradigm به صورت RC همین روش را به کار می گیریم و تنها تفاوت این است که در هر حرف ماکسیمم target سطر و ماکسیمم target ستون را بدست اورده و محل تلاقی ان حرف مورد نظر است

توضیحات مربوط به پروژه :

برای run کردن code ابتدا دو فایل فیلتر به نام‌های filter_1 و lp_filter را import کنید.

سپس در کد section1 index و section2 را کنید.

در نهایت کد های Project_phase2_96102342(2) را به ترتیب run کنید.

اصلاحات موردنظر شما مبنی براینکه Cross-validation مقدم بر محاسبه J-value انجام می‌شود یعنی J-value را از روی سطرهای training (نه validation) محاسبه می‌کنیم ، انجام شد.