تمرین شماره ۱
درس یادگیری ماشین
زهرا بیات۴۰۲۰۳۱۸۴

لینک کولب تمرین یک :

https://colab.research.google.com/drive/11vvqRppKGJ7-GGqYuWcbi4CxjU6rm1lj?usp=sharing

لینک مخزن گیت هاب پروژه های درس:

https://github.com/zbyt9406/ml-2025.git

سوال اول

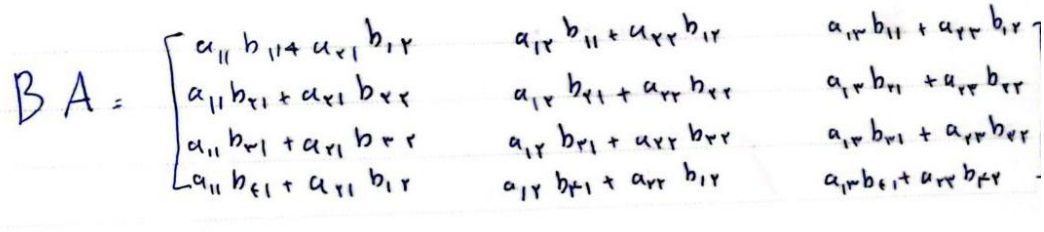
I ضرب و ابعاد

الف وب/

درماتریس A_{mn}، m تعداد سطرها و n تعداد ستون هاست در نتیجه ماتریس A_{2*3} و ماتریس B_{4*2} است.

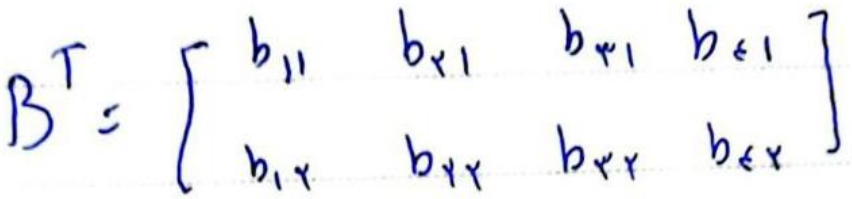
ماتریس حاصل از ضرب دو ماتریس شامل تعداد سطر هایی به اندازه ماتریس اول و تعداد ستون هایی به اندازه ماتریس دوم می باشد و در صورتی وجود دارد که تعداد ستون های ماتریس اول برابر تعداد سطرهای ماتریس دوم باشد . در نتیجه ابعاد و حاصل ماتریس های داده شده به صورت زیر است :

BA→ 4*3



ترانهاده ماتریس B یعنی جای سطر ها و ستون های آن جابه جا شده است پس:

BT→ 2*4



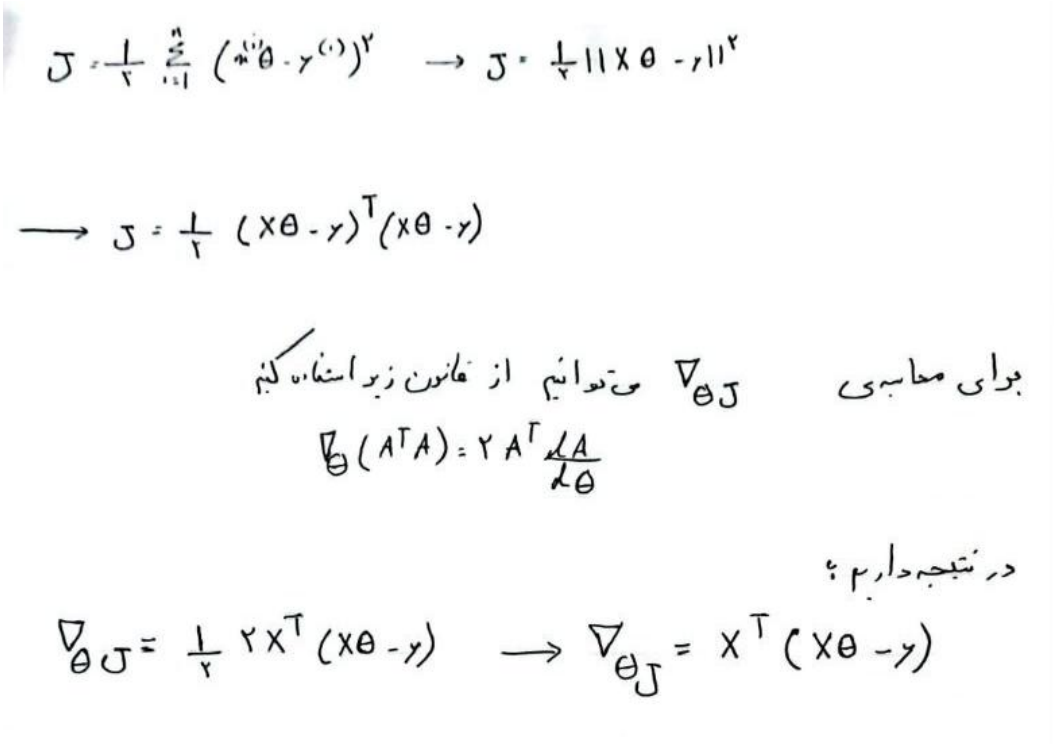
وجود ندارد→BTA

وجود ندارد→ ATB

II ضرب و ابعاد، دشوارتر

آ. حاصل ماتریس $x^i\theta$ یک عدد است پس یک بعد دارد.و حاصل Xθ دارای n بعد می باشد

ب/



سوال ۲

ا مقدمه

اصل بیز یک اصل بنیادی در نظریهٔ احتمال و آمار است که نشان می‌دهد چگونه می‌توان احتمال یک رویداد را پس از مشاهده یا کسب اطلاعات جدید به‌روزرسانی کرد. این قضیه به صورت زیر تعریف میشود:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

که:

$$P(B) = P(B|A)P(A)+P(B|A^C)P(A^C)$$

IIسناریو

آ

اگر A را رویداد اینکه فرد مراجعه کننده بیمار باشد در نظر بگیریم و A^C رویداد اینکه فرد مراجعه کننده سالم باشد و B را رویداد اینکه نتیجه تست مثبت باشد و B^C رویداد اینکه نتیجه تست منفی باشد در نظر بگیریم داریم:

P(B|A) = 99/100 ,P(A)=1/10000 , P($B^C|A^C) = 99/100$

طبق روابط قانون بیز داریم:

P(B)= $\frac{99}{100}\times\frac{1}{10000}$ +(1-$\frac{99}{100}$)(1-$\frac{1}{10000}$)=0.010098

P(A|B)=$\frac{.99\times.00001}{.010098}$ × 100=0.980392درصد

ب/

در این حالت براساس قاعده بیز و نتیجه تست قبلی داریم:

P(A|B,B) = $\frac{P(B|A)P(A|B)}{P(A|B)P(B|A)+P(B|A^C)(1-P(A|B)}$ =$\frac{0.9999\times0.00980392}{.9999\times.00980392+0.0001\times(1-0.00980392)}$= 0.989999998 → 98.9999 درصد

ج/

احتمال داشتن سرطان به صورت زیر بدست می آید:

P(A|B,B, B^C) = $\frac{P(B^C|A)P(A|B,B)}{P(A|B,B)P(B^C|A)+P(B^C|A^C)(1-P(A|B,B)} = \frac{01-0.999999)\times0.00989999998}{1-.999999\times.00989999998+0.999999\times(1-0.00989999998)}$

≈.0000989 → ≈0.01 %

سوال ۳

CWRU Dataset I

آ/

۱.پسوند این فایل mat. می باشد که نشان دهنده ی این است که این یک فایل متلب است .این مدل فایل‌ها معمولاً حاوی آرایه‌های عددی (مانند آرایه‌های ۱ بعدی، ۲ بعدی یا چند بعدی) هستند، اما می‌توانند شامل ساختارها، آرایه‌های سلولی (مشابه لیست‌های پایتون) یا انواع داده‌های پیچیده‌ای باشند که از MATLAB ذخیره شده‌اند.

۲.برای خواندن این فایل میتوان از کتابخانه scipy استفاده کرد . برای خواند این فایل میتوان کد زیر را مورد استفاده قرار داد:

```
import scipy.io

data = scipy.io.loadmat('122.mat')

print(data.keys())
```

این فایل را خوانده و در متغییری به نام mat_data ذخیره میکنیم . این داده ها شامل آرایه های عددی(numpy.ndarray) میباشد که با خواندن آن با کد قسمت قبل یک Python dictionary خواهیم داشت.

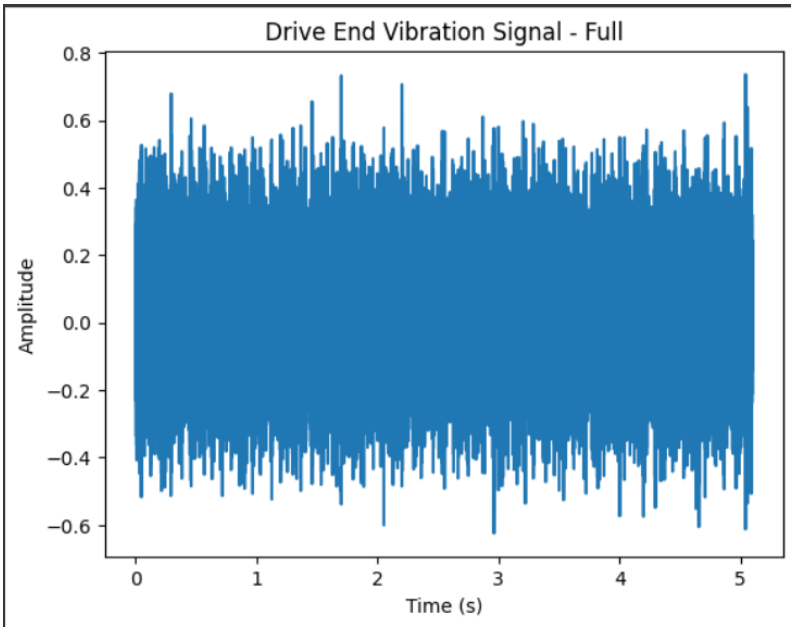
اجزای مهم این فایل شامل : '__header__', '__version__', '__globals__', 'X122_DE_time', 'X122_FE_time' 'X122RPM' می باشد . که میتوان با کد زیر به آن دست یافت

```
print(mat_data.keys())
```

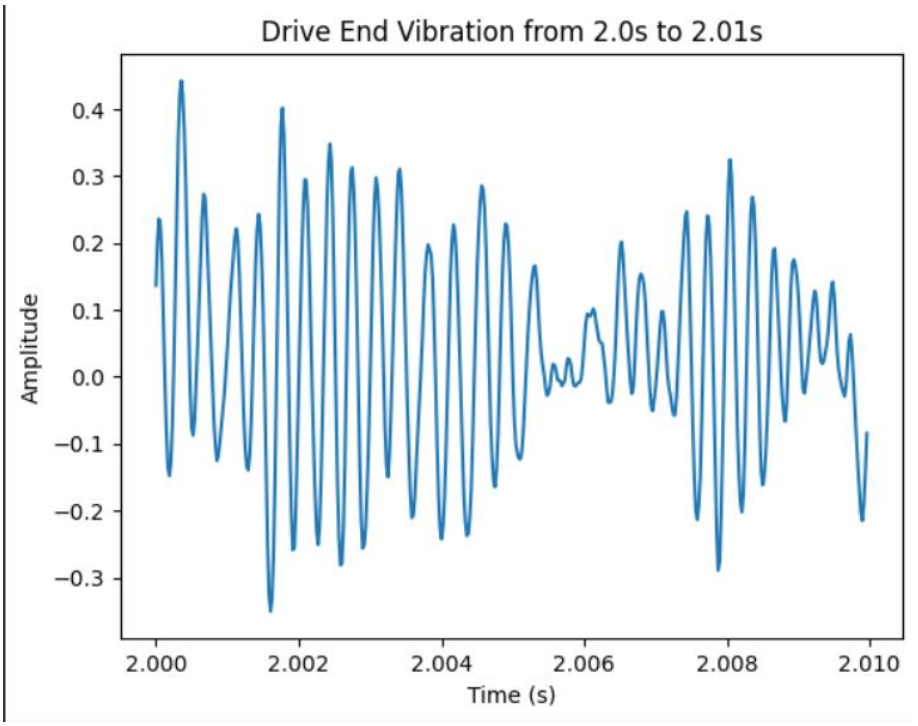
۳.از میان سیگنال ها سیگنال X122_DE_time که دارای دیتاتایپ float64، تایپ <class 'numpy.ndarray> و ابعاد ۱×۲۴۴۷۳۹ می باشد انتخاب میکنیم و در متغییر DE_raw ذخیره میکنیم سپس با استفاده از دستور squeeze یک بعد آن را کاهش میدهیم تا ۲۴۴۷۳۹ سمپل را داشته باشیم و آن ها را در DE_signal ذخیره میکنیم .

ب /

۱. با توجه به فرکانس نمونه برداری، که نشان دهنده تعداد نمونه های گرفته شده در یک ثانیه می باشد ، و با دانستن تعداد کل نمونه های این سیگنال(۲۴۴۷۳۹) میتوان زمان ثبت این نمونه ها را بدست آورد. کل مدت زمان ثبت این سیگنال حدودا ۵/۹۸ ثانیه میباشد.این زمان ها را در آرایه ای به نام time_array ذخیره میکنیم . سپس این سیگنال را در کل بازه ی زمانی آن نمایش میدهیم که به صورت زیر میباشد:



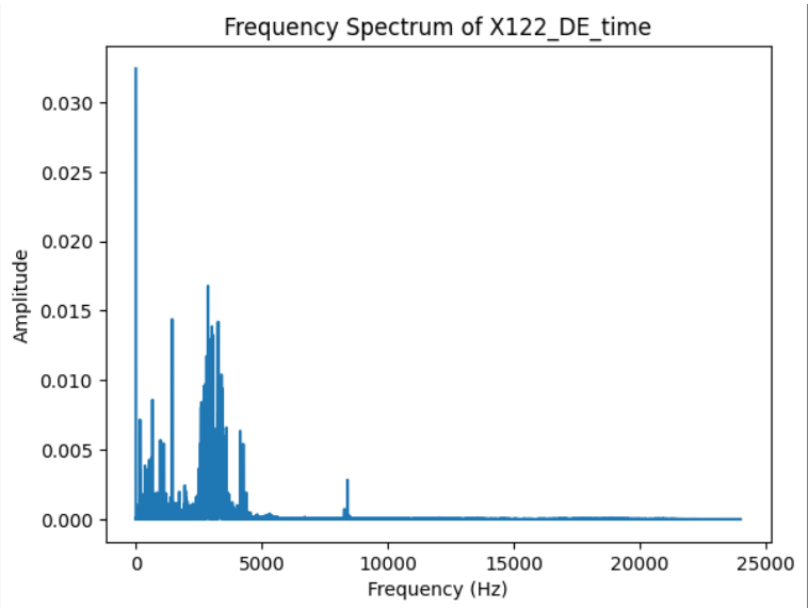
۲. برای نمایش سیگنال در این بازه زمانی ابتدا با ضرب دو مقدار ۲ و ۲/۰۱ در فرکانس نمونه برداری شماره ی ایندکس نمونه های مربوط به این دو زمان را بدست آورده سپس سیگنال و زمان متناظر با آن را بر اساس شماره ایندکس به دست آمده دراین بازه در دو آرایه به نام های time_slice و signal_slice و نمایش میدهیم .



ج/

۱. در این قسمت ابتدا یک تابع به نام plot_frequency_spectrum تعریف میکنیم در این تابع از سیگنال با استفاده از دستور np.fft.rfft، FFT سیگنال را به دست می آوریم .استفاده از این دستور بجای np.fft.fft() سبب میشود تا تنها مولفه های فرکانسی غیر صفر بدست آید اندازه مقادیر بدست آمده را با دستور abs() می یابیم (برای نرمالایز کردن مقادیر آن ها را بر تعداد نمونه ها تقسیم میکنیم)

محور افقی و فرکانس های متناظر با مقادیر fft بدست آمده را با استفاده از دستور np.fft.rfftfreq(N, d=1/fs) (d همان گام زمانی است) بدست آورده و در نهایت طیف فرکانسی را رسم می نماییم



۲. در تجزیه و تحلیل سیگنال در حوزه فرکانس، «فرکانس غالب (Dominant Frequency) «به فرکانسی گفته می‌شود که در طیف فرکانسی، دارای بالاترین دامنه (Amplitude) یا توان (Power) است. به بیان دیگر، وقتی سیگنال را تبدیل به حوزه فرکانس می‌کنیم و به نمودار آن نگاه می‌کنیم، فرکانس غالب معمولاً بیشترین مقدار قله را در آن طیف دارد و نشان می‌دهد سیگنال تمایل دارد با چه بسامدی (در میان سایر بسامدهای موجود) بیشترین انرژی یا دامنه را داشته باشد. در این سوال برای بدست آوردن ماکزیمم فرکانس میتوان ابتدا اندیس بیشترین فرکانس در طیف فرکانسی را بدست آورد و سپس بوسیله همین ایندکس مقدار متناظر آن را از آرایه مربوط به مقدارفرکانس(نمودار افقی قسمت قبل) استخراج کرد. همانطور که در تصویر طیف فرکانس سیگنال مشخص است فرکانس غالب این سیگنال ، فرکانس صفر یا همان مقدار DC سیگنال است.

د/

۱. ابتدا مقدار قطعات و میزان overlap دلخواه را در دو متغییر ذخیره میکنیم (در اینجا overlapرا ۶۴ در نظر گرفته ایم) سپس یک گام به اندازه مقدار ۱۲۸ منهای مقدار overlap تعریف میکنیم در یک حلقه while با این گام حرکت کرده و سیگنال را به قطعاتی با طول ۱۲۸ تقسیم کرده و در یک لیست قرار میدهیم در پایان هر تکرار از حلقه اولین شماره نمونه قطعه بعدی را به روز رسانی میکنیم. این کار را تا زمانی که حاصل جمع مقدار شروع قطعه ۱۲۸(طول قطعات) کمتر از طول سیگنال باشد ادامه میدهیم. در زیر تعداد قطعات ،طول آنها و ابعاد ارایه ی شامل این قطعات نشان داده شده است.

Number of chunks: 3823

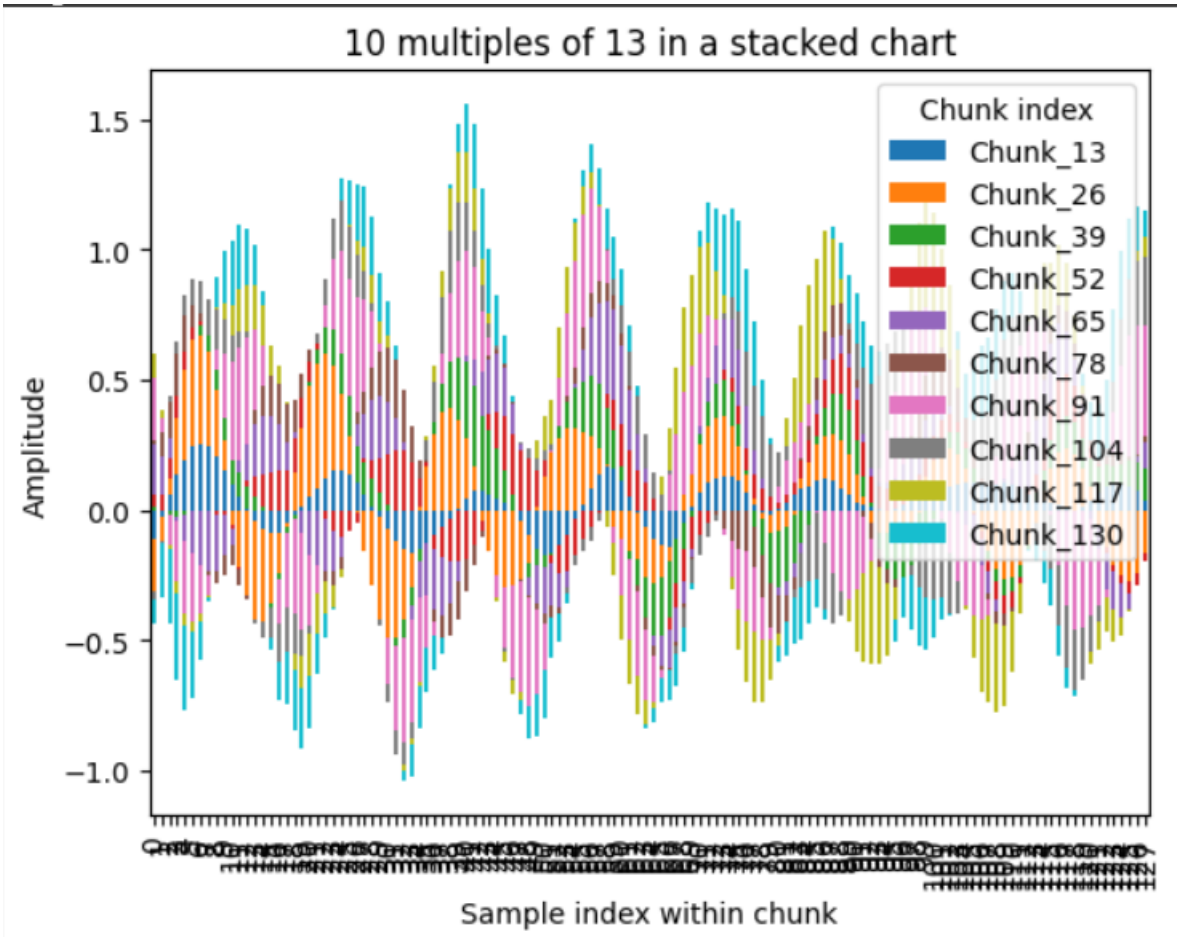
Chunk size: 128

chunks_array shape: (3823, 128)

ه/

همانطور که گفتیم در قسمت قبل هر قطعه را در لیست chunks ذخیره کرده ایم. با استفاده از دستور np.stack(chunks, axis=0)

ارایه های ۱ بعدی لیست chunks به صورت عمودی رو هم قرار داده و یک 2D NumPy array با ابعاد (3816, 128) میسازیم.سپس با استفاده از دستور pd.DataFrame(...) یک DataFrame میسازیم که ردیف های آن قطعه های سیگنال و ستون های آن شماره ایندکس های نمونه های متناظر با این قطعات میباشد. یعنی ابعاد دیتا فریم (3816, 128) متناظر با ۳۸۱۶ سطر و ۱۲۸ ستون است.سپس یک حلقه for با ۱۰ تکرار تعریف میکنیم که ردیف های با شماره ایندکس که مضربی از ۱۳ باشد را پیدا میکند. سپس ردیف متناظر با این ایندکس ها را بوسیبه ی دستور df_chunks.iloc[[idx]] گرفته و در متغییری به نام row_df ذخیره میکنیم . و این ۱۰ ردیف های استخراج شده را در لیست ای با نام df_m13_list ذخیره می نماییم. در انتها با استفاده از دستور pd.concat(df_m13_list, ignore_index=True) تمام ردیف ها را به هم میچسبانیم و در یک دیتا فریم جدید به نام df_m13 ذخیره میکنیم و برای نمایش صحیح از آن transpose میگیریم . و در نهایت این قطعه های سیگنال که دارای ایندکس از مضارب ۱۳می باشد را نمایش میدهیم که به صورت زیر میباشد .



و/

در این قسمت یک تابع با نام calculate_features تعریف کرده و میانگین ، انحراف معیار و ریشه ی میانگین مربعات را با استفاده از دستورات np.mean(signal) و np.std(signal) و np.sqrt(np.mean(signal**2)) به ترتیب پیدا سازی کرده ایم سپس با استفاده از نمونه های سیگنال و این تابع مقادیر را بدست آورده و در یک فایل csv ذخیره کرده ایم.

	A	B	C
1	mean	std	rms
2	0.032419	0.14588	0.149439
3			
4			

Iris Dataset II

آ/

۱.

دیتاست **Iris** یکی از معروف ترین و پراستفاده ترین مجموعه داده‌ها در یادگیری ماشین و آمار است. این دیتاست اولین بار توسط **رونالد فیشر** معرفی شد و برای مسائل طبقه بندی (Classification) بسیار پر کاربرد است. این دیتاست یک فایل CSV است که دارای ۱۵۰ نمونه از گلهای زنبق است (ردیف ها) . هر نمونه دارای ۴ ویژگی(ستون عددی) یعنی طول کاسبرگ(sepal length) ، عرض کاسبرگ(sepal width) ، طول گلبرگ (petal length) و عرض گلبرگ(petal width) میباشد. در این دیتاست سه کلاس مختلف(سه نوع گل زنبق) وجود دارد که هر نمونه به یکی از این سه کلاس تعلق دارد(Setosa، Versicolor، Virginica) هرکلاس دارای ۵۰ نمونه میباشد.

۲.با استفاده از sckit_learn و دستور iris = load_iris() میتوانیم این دیتاست را فرخوانی کنیم . سپس قسمت های مختلف دیتاست ، نام ویژگی های ، نام کلاس ها یا همان تارگت و ابعاد داده های این دیتاست و ابعاد تارگت را استخراج میکنیم که به صورت زیر میباشد :

```
Keys of iris dataset:
 dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])

Feature names:
 ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']

Target names:
 ['setosa' 'versicolor' 'virginica']

Shape of data: (150, 4)
Shape of target: (150,)
```

۳.

برای تقسیم داده ها به دو قسمت تست و ترین ابتدا تابع **train_test_split**را از ماژول **sklearn.model_selection**فراخوانی میکنیم.این تابع برای **تقسیم مجموعه داده** به دو بخش **داده‌های آموزش (Training Set)** و **داده‌های آزمایش (Test Set)** استفاده می‌شود. سپس ویژگیهای دیتاست به متغیر x و برچسب یا همان کلاس ها را به متغییر yنسبت میدهیم با استفاده از تابع فراخوانی شده داده های درون x و y را با نسبت ۲۰ درصد برای تست و ۸۰ درصد برای آموزش تقسیم میکنیم . این عملیات به صورت رندوم از در میان داده ها انجام میشود

random_state=42 این خط از کد باعث می‌شود که هر بار که کد اجرا می‌شود، **تقسیم داده‌ها یکسان** باشد (نتایج قابل تکرار باشند).

۴.

در این قسمت ابتدا برای نام ستون هایی که برای هر دیتافریم میخواهیم ایجاد کنیم .نام ویژگی های این دیتاست را دریافت میکنیم و آن ها را در یک لیست به نام column_names ذخیره میکنیم .(این لیست شامل ۴ ویژگی است: 'sepal width (cm)', 'sepal length (cm)', 'petal length (cm)', 'petal width (cm)') سپس دو دیتا فریم مجزا برای داده های تست و آموزش با نام های train_df و test_df ایجاد میکنیم . دیتا فریم آموزش شامل ۱۲۰ داده و ۴ ستون ویژگی است (train_df = pd.DataFrame(X_train, columns=column_names) یک ستون برای نمایش کلاس هر نمونه و یک ستون برای نمایش اینکه نمونه جزو داده های تست است یا آموزش(در اینجا داده های آموزش) ایجاد میکنیم(این مرحله در قسمت بعدی سوال کاربرد دارد). این کار را برای دیتا فریم

تست که شامل ۳۰ نمونه است هم انجام میدهیم .پنج ردیف اول این دو دیتا فریم به صورت زیر می باشد. (توجه کنید که تارگت یا همان کلاس نمونه ها با اعداد۰،۱و۲ نمایش داده شده است)

```
Training DataFrame (first 5 rows):
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                4.6               3.6                1.0               0.2
1                5.7               4.4                1.5               0.4
2                6.7               3.1                4.4               1.4
3                4.8               3.4                1.6               0.2
4                4.4               3.2                1.3               0.2

   target dataset
0       0   train
1       0   train
2       1   train
3       0   train
4       0   train

Test DataFrame (first 5 rows):
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                6.1               2.8                4.7               1.2
1                5.7               3.8                1.7               0.3
2                7.7               2.6                6.9               2.3
3                6.0               2.9                4.5               1.5
4                6.8               2.8                4.8               1.4

   target dataset
0       1    test
1       0    test
2       2    test
3       1    test
4       1    test
```

۵.

با استفاده از دستور concat به صورت این خط کد :

```
combined_df = pd.concat([train_df, test_df], ignore_index=True)
```

این دو دیتا فریم را ادغام میکنیم

۵ ردیف اول به صورت زیر میباشد :

```
Combined DataFrame (last 5 rows):
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
145                6.7               3.0                5.2               2.3
146                6.7               2.5                5.8               1.8
147                6.8               3.2                5.9               2.3
148                4.8               3.0                1.4               0.3
149                4.8               3.1                1.6               0.2

     target dataset
145       2    test
146       2    test
147       2    test
148       0    test
149       0    test
```

ب/

۱. در این قسمت دو ویژگی طول کاسبرگ(feature index 0) و عرض کاسبرگ (feature index 1) انتخاب کرده ایم.

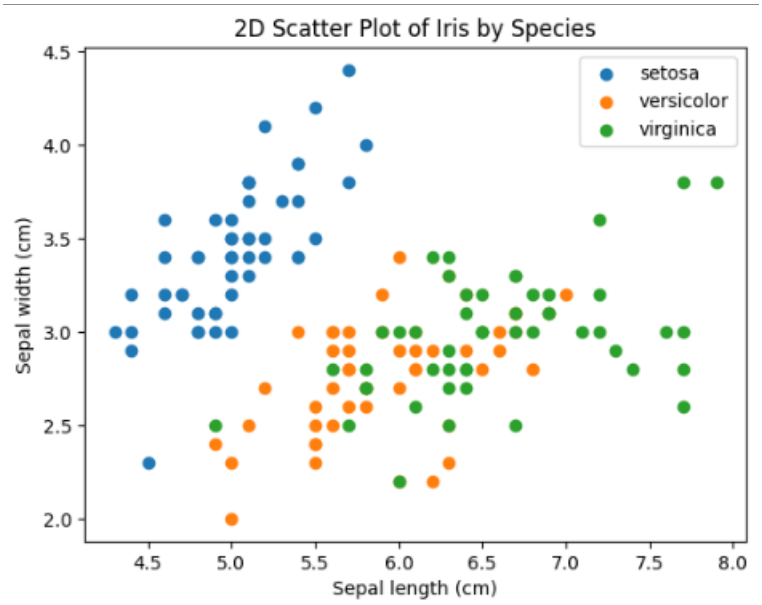
ابتدا با استفاده از کد زیر مقداری عددی سه کلاس(برچسب نمونه ها) را استخراج کرده و سپس یک لیست با نام species_labels از مقادیر یکتای آن ها ایجاد میکنیم(یعنی مقادیر ۰و۱و۲در این لیست قرار میگیرند که نشان دهنده همان سه گونه زنبق موجود در دیتاست است)

```
species_labels = np.unique(combined_df["target"])
```

برای رسم نمودار پراکندگی حلقه for ایی روی هرکلاس **(0, 1, 2)**تکرار می‌شود و با استفاده از کد زیر داده‌های مربوط به آن گونه‌ی خاص را از دیتافریم ترکیبی تست و آموزش جدا می‌کند.

```
subset = combined_df[combined_df["target"] == species]
```

در نهایت با استفاده از دستور plt.scatter نمونه ها بر اساس دو ویژگی طول کاسبرگ (محور افقی) و عرض کاسبرگ(محور عمودی) نمایش داده میشود. هر گونه زنبق، نقاط پراکندگی خود را در نمودار با رنگ و برچسب مخصوص خود خواهد داشت.

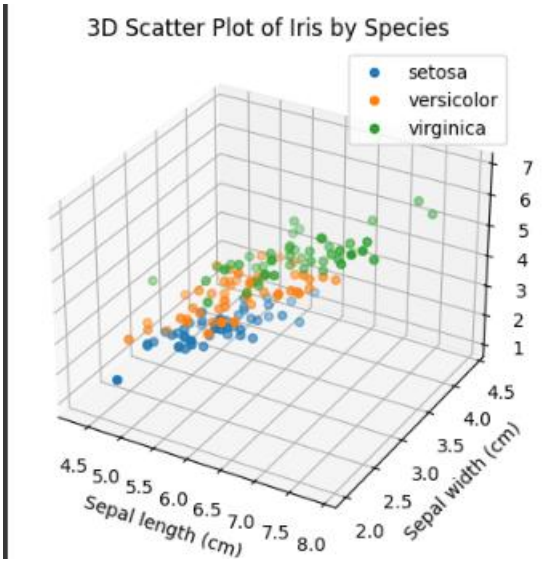


۲.

برای رسم نمودار ۳ بعدی ۳ ویژگی طول کاسبرگ(feature index 0) ، عرض کاسبرگ (feature index 1) و طول گلبرگ (feature index 2) انتخاب کرده ایم.

برای رسم نمودار سه بعدی از ابزار mpl_toolkits.mplot3d در Matplotlib استفاده میکنیم (فرخوانی در خط اول کد این قسمت). **Axes3D** کلاس اصلی برای **افزودن محور سه‌بعدی** (3D axes) به نمودار است.

به مانند قسمت قبل حلقه ای روی هرکلاس تکرار میشود و داده های مربوط به آن کلاس را از دیتا فریم ترکیبی تست و آموزش جدا میکند و در نهایت هر گونه زنبق با رنگ و نماد مخصوص خود در فضای سه‌بعدی نمایش داده می‌شود.



.۳

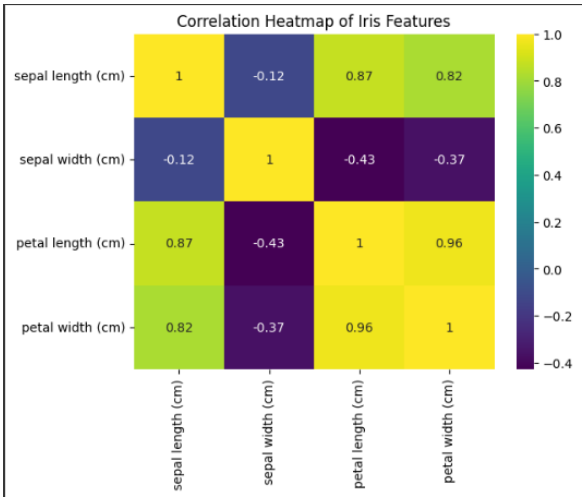
در این قسمت نقشه حرارتی (Heatmap) از همبستگی ویژگی‌های دیتاست Irisرسم کردیم. این نقشه حرارتی به ما نشان می‌دهد که چگونه ویژگی‌های مختلف با یکدیگر همبستگی دارند. ابتدا کتابخانه seaborn را با کد import seaborn as sns فراخوانی میکنیم. این یک کتابخانه برای بصری‌سازی داده‌ها است که مبتنی بر Matplotlibکار می‌کند.

سپس با کد corr_matrix = combined_df[feature_names].corr() چهار ویژگی را از دیتافریم combined_dfانتخاب کزده و یک ماتریس همبستگی ۴×۴ را بر اساس آنها ایجاد میکنیم. مقدار هر خانه از این ماتریس بین ۱- تا ۱ است.

(۱ یعنی:همبستگی مثبت کامل (یعنی دو ویژگی باهم رشد می‌کنند) ۱- یعنی : همبستگی منفی کامل (یعنی یکی افزایش پیدا کند، دیگری کاهش می‌یابد) و ۰یعنی بدون همبستگی)

سپس با دستور sns.heatmap(corr_matrix, annot=True, cmap='viridis') این ماتریس را نمایش میدهیم .در این ماتریس رنگ‌های روشن‌تر (زرد) نشان دهنده ی همبستگی مثبت قوی و رنگ‌های تیره‌تر (بنفش/سبز تیره) نشان دهنده ی همبستگی منفی یا ضعیف می باشد.

ماتریس همبستگی ویژگی ها به صورت زیر می باشد:



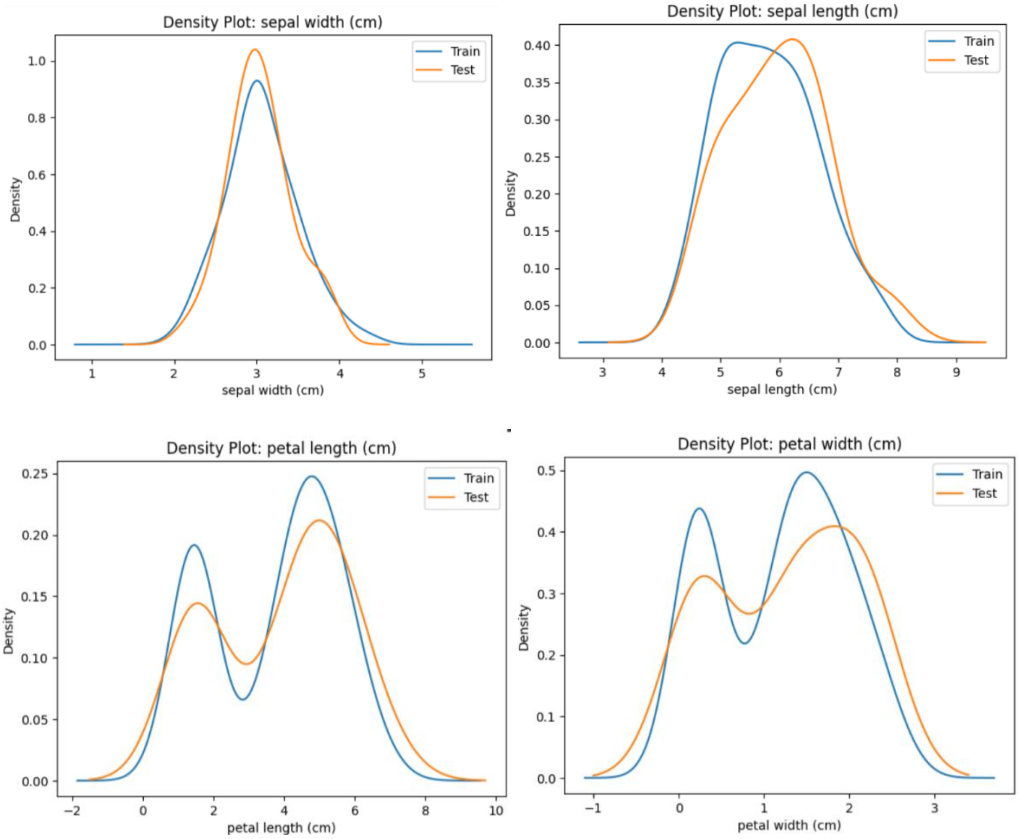
.٤

برای نمایش نمودار چگالی احتمال برای هریک از ویژگی ها ، ابتدا نام ویژگی ها را با استفاده از کد feature_names استخراج کرده و در لیستی به نام features قرار میدهیم . سپس در یک حلقه for بر روی ویژگی ها و با استفاده از دو خط کد زیر پس از گرفتن مقادیر داده ها در دیتافریم های آموزش و تست ، نمودار چگالی احتمال آن ها را برای هر یک از ویژگی ها رسم میکند.

```
train_df[feature].plot(kind='kde', label='Train')

test_df[feature].plot(kind='kde', label='Test')
```

و در نهایت با نمایش این نمودار ها میتوان نمودار توابع چگالی احتمال را برای ٤ ویژگی موجود در دیتافریم های تست و آموزش به صورت زیر نشان داد:



ج/

برای انجام این قسمت ازسوال از ویژگی طول کاسبرگ استفاده شده است . ابتدا به دیتافریم ترکیبی تست و آموزش یک ستون جدید به نام sepal_length_discrete اضافه میکنیم . سپس با استفاده از کد ["sepal length (cm)"]combined_df مقدار طول کاسبرگ برای تمام نمونه‌های دیتافریم گرفته می شود . از تابع pd.cut برای دسته‌بندی (Binning) مقادیر عددی به گروه‌های گسسته استفاده می‌شود . برای این منظوز مقدار sepal length (cm)را به سه دسته (Short, Medium, Long) تقسیم می‌کند. این تقسیم بندی بر اساس تعریف پارامتر bins=[0, 5.0, 6.0, np.inf] انجام میشود .اگر طول کاسبرگ نمونه ای بین ۰ تا ۵ باشد در دسته کوتاه با برچسب short ، اگر بین ۵ تا ۶ باشد در دسته متوسط با برچسب medium و اگر بیشتر از ۶ باشد در دسته بلند با

برچسب long قرار میگیرد. این برچسب ها برای هر نمونه در ستون ایجاد شده sepal_length_discrete قرار میگیرد. به صورت کلی کد این قسمت به صورت زیر می باشد :

```
combined_df["sepal_length_discrete"] = pd.cut(
   combined_df["sepal length (cm)"],
   bins=[0, 5.0, 6.0, np.inf],
   labels=["short", "medium", "long"]
)
```

نمایش ۱۰ سطر اول از دیتافریم ترکیبی پس از انجام عملیات بالا به صورت زیر می باشد:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	\
0	4.6	3.6	1.0	0.2	
1	5.7	4.4	1.5	0.4	
2	6.7	3.1	4.4	1.4	
3	4.8	3.4	1.6	0.2	
4	4.4	3.2	1.3	0.2	
5	6.3	2.5	5.0	1.9	
6	6.4	3.2	4.5	1.5	
7	5.2	3.5	1.5	0.2	
8	5.0	3.6	1.4	0.2	
9	5.2	4.1	1.5	0.1	

	target	dataset	sepal_length_discrete
0	0	train	short
1	0	train	medium
2	1	train	long
3	0	train	short
4	0	train	short
5	2	train	long
6	1	train	long
7	0	train	medium
8	0	train	short
9	0	train	medium

د/

در این بخش ابتدا بویسله کد combined_df["target"] == [که میتواند برابر با هریک از مقادیر ۰،۱ ، ۲ باشد نمونه های دارای برچیب کلاس های مختلف را برای انجام کار آماری از دیتا فریم ترکیبی جدا میکنیم(برای مثال نمونه های دارای برچسب ۰ با این خط کد فیلتر میشوند : combined_df[combined_df["target"] == 0])

همانطور که پیشتر گفته شد در ستون target در دیتا فریم ترکیبی کلاس نمونه ها داده شده است ، کلاس ۰ برای نوع Setosa ، ۱ برای نوع Versicolor و ۲ برای نوع Virginica می باشد . پس از انتخاب کلاس مورد نظر ، نمونه های مربوط به آن در دیتا فریم setosa_df ذخیره میشود. با استفاده از دستور describe() و اعمال روی دیتا فریم setosa_df یک خلاصه آماری ازهریک از ویژگی‌های عددی مربوط به نمونه های کلاس انتخاب شده ایجاد شده است.

این اطلاعات آماری شامل ماکزیمم ،میانگین ،انحراف معیار ، چارک‌ها شامل : صدک ۲۵ ام ، صدک ۵۰ ام (میانه) ، صدک ۷۵ ام و تعداد کل نمونه ها می باشد.

برای مثال ویژگی های آماری مربوط به داده های کلاس صفر و یک در دو شکل زیر نشان داده شده است:

کلاس ۰ :

	sepal length (cm)	sepal width (cm)	petal length (cm)	\
count	50.000000	50.000000	50.000000	
mean	5.936000	2.770000	4.260000	
std	0.516171	0.313798	0.469911	
min	4.900000	2.000000	3.000000	
25%	5.600000	2.525000	4.000000	
50%	5.900000	2.800000	4.350000	
75%	6.300000	3.000000	4.600000	
max	7.000000	3.400000	5.100000	

	petal width (cm)	target
count	50.000000	50.0
mean	1.326000	1.0
std	0.197753	0.0
min	1.000000	1.0
25%	1.200000	1.0
50%	1.300000	1.0
75%	1.500000	1.0
max	1.800000	1.0

کلاس ۱ :

	sepal length (cm)	sepal width (cm)	petal length (cm)	\
count	50.000000	50.000000	50.000000	
mean	5.936000	2.770000	4.260000	
std	0.516171	0.313798	0.469911	
min	4.900000	2.000000	3.000000	
25%	5.600000	2.525000	4.000000	
50%	5.900000	2.800000	4.350000	
75%	6.300000	3.000000	4.600000	
max	7.000000	3.400000	5.100000	

	petal width (cm)	target
count	50.000000	50.0
mean	1.326000	1.0
std	0.197753	0.0
min	1.000000	1.0
25%	1.200000	1.0
50%	1.300000	1.0
75%	1.500000	1.0
max	1.800000	1.0