دانشکده مهندسی برق

# تکلیف کامپیوتری درس بازشناسی آماری الگو، سری دوم

**استاد**

دکتر ابریشمی مقدم

نیمسال اول ۱۴۰۵-۱۴۰۴

**مهلت تحویل: ۳۰ آبان ۱۴۰۴**

**با سلام و آرزوی شادی، موفقیت و سلامتی؛**

**لطفا در تحویل پاسخ‌های خود موارد زیر را مدنظر داشته باشید:**

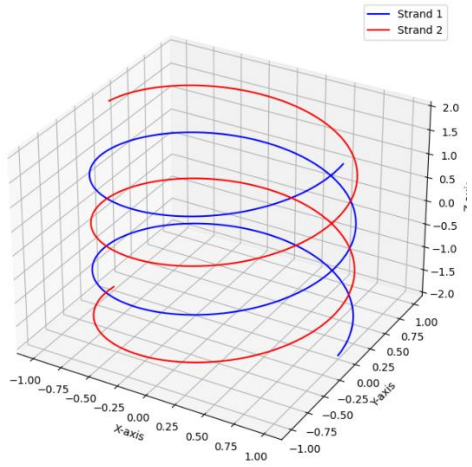
- فقط برنامه‌هایی که به زبان ترجیحا پایتون و یا متلب باشند قابل قبول خواهند بود.
- تحویل همزمان گزارش و کدها الزامی است.
- گزارش باید شامل خروجی‌های کدهای نوشته شده باشد، که موارد خواسته شده در سوالات هستند و سایر توضیحات خواسته شده دیگر در متن سوالات نیز پاسخ داده شود (از آوردن کد در گزارش خودداری کنید).
- لطفا کدهای برنامه به صورت ماجولار و همراه با توضیحات کافی باشند؛ طوری که بخش‌های مختلف برنامه کاملا قابل تفکیک بوده و اجرا و ارزیابی هر بخش توسط کاربر به آسانی و بدون نیاز به ورود به جزییات برنامه میسر باشد.
- فایل تحویلی پاسخ شما باید تنها یک فایل زیپ، تحت عنوان "SPR_CHW2_Student ID" محتوی دو پوشه باشد. گزارش خود را در پوشه اول با عنوان "Report" و کدهای خود را ترجیحا به فرمت Jupyter Notebook در پوشه "Codes" قرار دهید.
- با این که همکاری، مشورت، و استفاده از ابزارهای کمکی در حل سوالات پیشنهاد می‌شود، حتما به صورت مستقل به نوشتن کدها و گزارش بپردازید.
- ممکن است از دانشجویی خواسته شود در زمانی که تعیین خواهد شد جزئیات کدش را در جلسه‌ای مجازی توضیح دهد، نتایج را تحلیل کند و حتی تغییراتی در پارامترهای کد اعمال کند. در صورتی که دانشجویی تمارین را تحویل داده باشد اما نتواند کد خود را توضیح دهد و یا تغییراتی روی آن اعمال کند، و یا اینکه کد یا گزارش تحویلی به پاسخ دیگران شباهت غیرمنطقی داشته باشد، نمره تمرین صفر لحاظ شده و نمره‌ای منفی هم لحاظ خواهد شد.
- در صورت وجود هرگونه سوال یا ابهام، مشکل مربوطه را با آی دی تلگرام زیر در میان گذارید:

@omid_Emaa

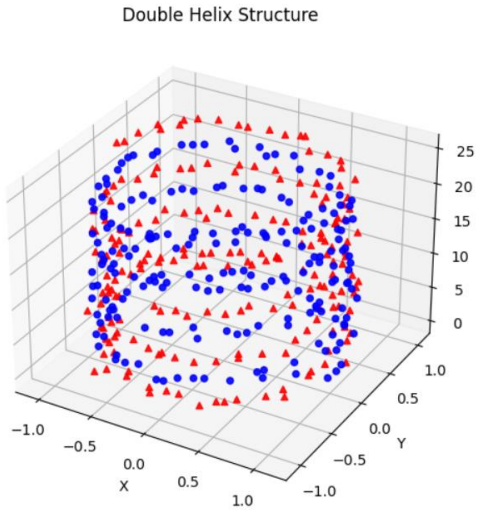
در این تمرین قصد داریم به پیاده‌سازی الگوریتم KPCA بپردازیم.

## بخش اول: (۵۰ امتیاز)

۱: ابتدا یک ساختار مارپیچ دوگانه تولید کنید.




میتوانید از کدهای پایین استفاده کنید و نتیجه می‌بایست شکلی مشابه زیر باشد:



## Python:

```
import numpy as np
theta = np.linspace(0, 8 * np.pi, 200)
x1 = np.cos(theta) + np.random.normal(0, 0.05, len(theta))
y1 = np.sin(theta) + np.random.normal(0, 0.05, len(theta))
z1 = theta + np.random.normal(0, 0.1, len(theta))

x2 = -np.cos(theta) + np.random.normal(0, 0.05, len(theta))
y2 = -np.sin(theta) + np.random.normal(0, 0.05, len(theta))
z2 = theta + np.random.normal(0, 0.1, len(theta))

fig = plt.figure(figsize=(6, 6))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x1, y1, z1, color='blue', alpha=0.9)
ax.scatter(x2, y2, z2, color='red', alpha=0.9)
plt.show()
```

## MATLAB:

```
theta = linspace(0, 8 * pi, 200);
x1 = cos(theta) + randn(1, length(theta)) * 0.05;
y1 = sin(theta) + randn(1, length(theta)) * 0.05;
z1 = theta + randn(1, length(theta)) * 0.1;

x2 = -cos(theta) + randn(1, length(theta)) * 0.05;
y2 = -sin(theta) + randn(1, length(theta)) * 0.05;
z2 = theta + randn(1, length(theta)) * 0.1;

figure;
scatter3(x1, y1, z1, 36, 'b', 'MarkerFaceAlpha', 0.9);
```

```
hold on;
scatter3(x2, y2, z2, 36, 'r', 'MarkerFaceAlpha', 0.9);
view(3);
hold off;
```

۲. با استفاده از الگوریتم PCA، که در تمرین سری اول با آن آشنا شدید، مجموعه داده را بدون کاهش بعد به فضای ویژگی جدید ببرید. (در این قسمت می‌توانید از توابع آماده استفاده کنید.)

۳. حال با استفاده از الگوریتم KPCA، مجموعه داده را یک بار با استفاده از تابع آماده KernelPCA موجود در کتابخانه sklearn، به فضای ویژگی دوبعدی جدید ببرید و مجموعه داده را به تفکیک کلاس در فضای جدید نمایش دهید.

۴. از میان کرنل‌هایی که در کلاس با آنها آشنا شده‌اید، کدام کرنل عملکرد بهتری دارد و چرا؟ (چندجمله‌ای، rbf، تانژانت هایپربولیک)

می‌توانید مشخصات کرنل‌ها را نیز تغییر دهید و عملکرد آنها را با هم مقایسه کنید (مثلا پارامتر گاما در کرنل RBF).

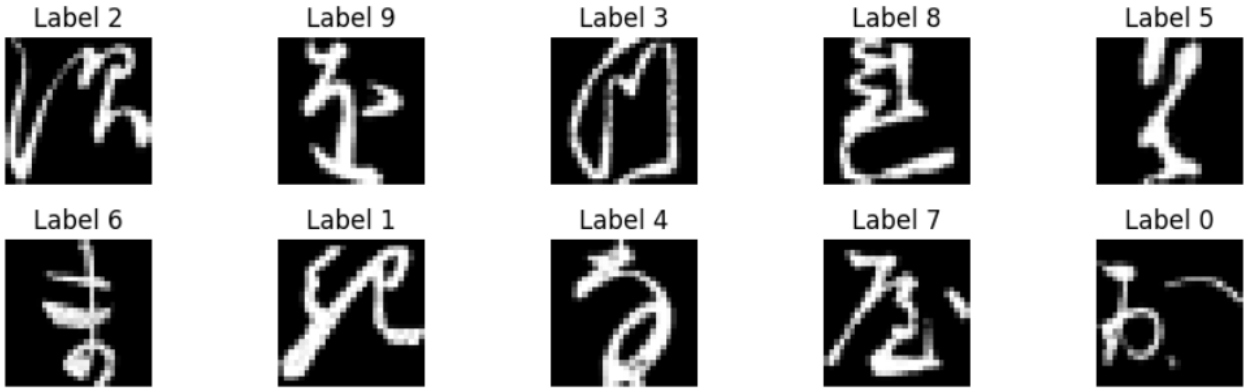
۵. مجموعه داده را، بار دیگر بدون استفاده از توابع آماده به فضای ویژگی دو بعدی جدید ببرید و به تفکیک کلاس در فضای جدید نمایش دهید.

**بخش دوم: (۷۰ امتیاز)**

در ادامه به بررسی کاربرد الگوریتم‌های PCA و KPCA در زمینه کاهش نویز در تصاویر خواهیم پرداخت. مجموعه داده‌ای که در این تمرین از آن استفاده خواهیم کرد، Kuzushiji-MNIST (KMNIST) نام دارد که شامل ۷۰۰۰۰ تصویر سیاه و سفید از حروف خوشنویسی ژاپنی (Kuzushiji) است. هر تصویر اندازه‌ای برابر با ۲۸ در ۲۸ پیکسل دارد و داده‌ها در ده دسته مختلف طبقه‌بندی شده‌اند.

لینک راهنمایی استفاده و توضیحات داده

۶. یک نمونه تصویر به ازای هر برچسب را مانند تصویر زیر نمایش دهید.




۷. هر تصویر که آرایه‌ای با ابعاد ۲۸*۲۸ است را به برداری با اندازه ۷۸۴ تبدیل کنید.

۸. ابعاد ویژگی مجموعه داده را با استفاده از الگوریتم PCA کاهش (۴۰۰ بعد) دهید. سپس تصاویر را تنها با استفاده از این مؤلفه‌ها بازسازی کنید. تصاویر بازسازی‌شده را نمایش دهید.

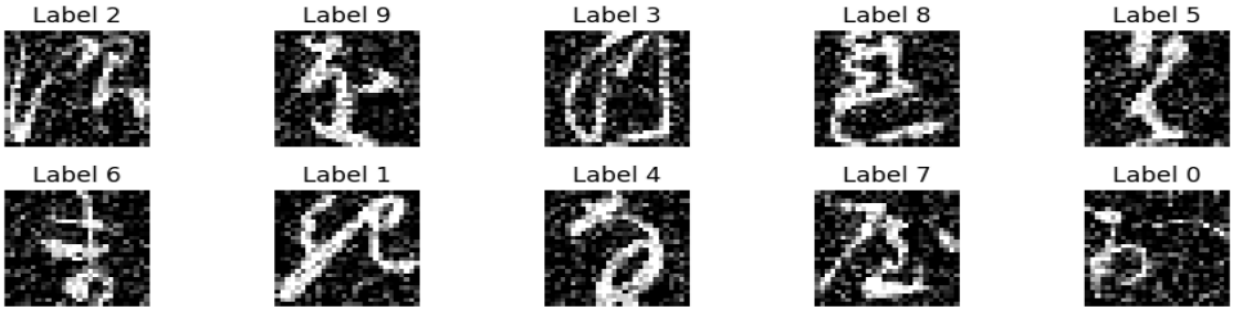
۹. ابعاد ویژگی مجموعه داده را با استفاده از الگوریتم KPCA کاهش (۴۰۰ بعد) دهید. سپس تصاویر را تنها با استفاده از این مؤلفه‌ها بازسازی کنید. تصاویر بازسازی شده را نمایش دهید.

(gamma = 1e-3, kernel = ‘rbf’)

: PCA

: KPCA

۱۰. به داده‌ها نویز گوسی با مقیاس ۰,۲۵ اضافه کنید. توجه کنید که افزودن نویز گوسی نباید موجب خارج شدن سطح خاکستری هر پیکسل از محدوده [۰،۱] بشود. به عبارت دیگر چنانچه افزودن نویز به مقدار هر پیکسل، مقدار آن را از محدوده مجاز خارج کند، با بکارگیری عملگر اشباع، باعث شوید تا حاصل جمع در محدوده مجاز قرار گیرد. حال، مشابه ۶، یک نمونه تصویر به ازای هر برچسب را مانند تصویر زیر نمایش دهید. می‌توانید از تابع آماده random.normal موجود در کتابخانه numpy استفاده کنید.




۱۰. مرحله ۸ را برای داده‌های نویزی تکرار کنید.

۱۱. مرحله ۹ را برای داده‌های نویزی تکرار کنید و نتایج به دست آمده را در حالت استفاده از الگوریتم PCA و KPCA با یکدیگر مقایسه کنید.

"موفق باشید"