

在使用 python 时，我们经常会用到许多工具库，它们提供了较为方便的函数调用。但是仍然会有一些情况，例如数据类型或格式不符合函数要求，参数存在差异等，使得调用前需要对数据进行额外处理。本次作业要求基于 matplotlib, wordcloud, PIL, imageio 等绘图库的绘制函数，设计并实现适配器抽象类和不同的适配类，以实现不同类型数据的多样化可视，并在不同类型的数据上进行充分测试。具体要求如下：

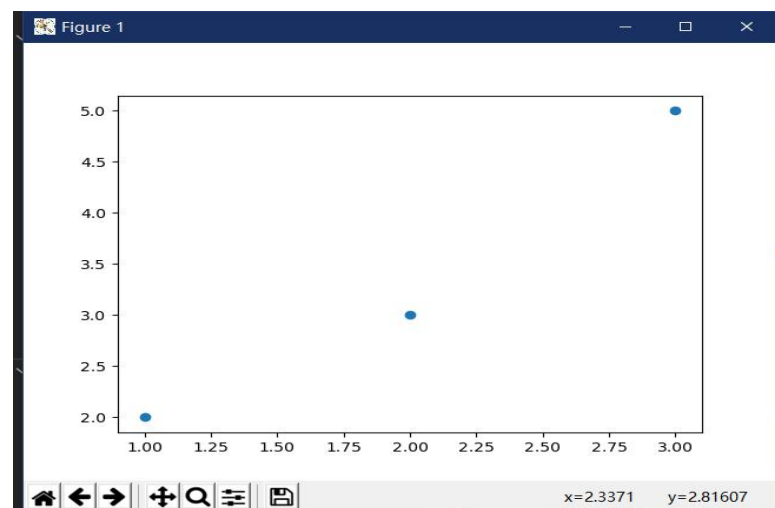
1. 要求设计抽象类 Plotter，至少包含抽象方法 plot(data, *args, **kwargs)方法，以期通过不同子类的具体实现来支持多类型数据的绘制，至少包括数值型数据，文本，图片等。

```
class Plotter(abc.ABC):  
    @abc.abstractmethod  
    def plot(self, data, *args, **kwargs):  
        pass
```

2. 实现类 PointPlotter，实现数据点型数据的绘制，即输入数据为[(x,y)...]型，每个元素为一个 Point 类的实例。

```
class PointPlotter(Plotter):  
  
    def plot(self, data, *args, **kwargs):  
        x=[]  
        y=[]  
        for i in data:  
            x.append(i[0])  
            y.append(i[1])  
        x=np.array(x)  
        y=np.array(y)  
        plt.scatter(x,y)  
        plt.show()
```

运行结果：

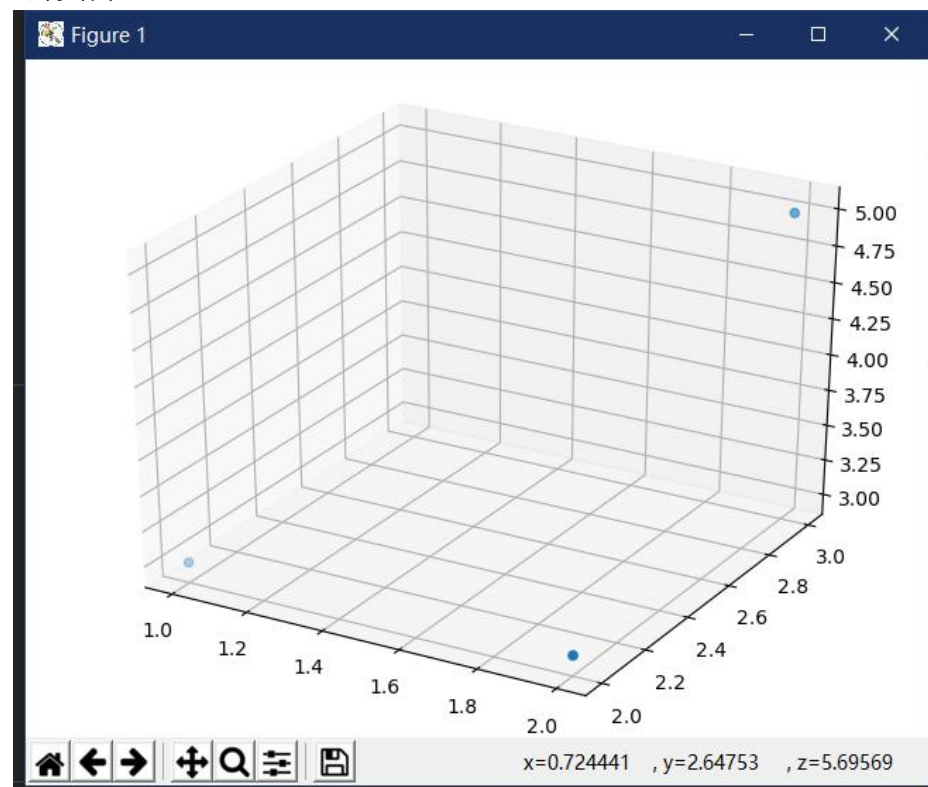


3. 实现类 `ArrayPlotter`, 实现多维数组型数据的绘制, 即输入数据可能是 `[[x1,x2...],[y1,y2...]]` 或者 `[[x1,x2...],[y1,y2...],[z1,z2...]]`。

```
class ArrayPlotter(Plotter):
    def plot(self, data, *args, **kwargs):
        fig = plt.figure()
        v=len(data)
        if v == 2:
            ax=plt.axes()
            x=np.array(data[0])
            y=np.array(data[1])
            ax.scatter(x,y)
            plt.show()

            elif v == 3:
                ax = Axes3D(fig)
                x=np.array(data[0])
                y=np.array(data[1])
                z=np.array(data[2])
                ax.scatter3D(x,y,z)
                plt.show()
```

运行结果:



4. 实现类 `TextPlotter`，实现文本型数据的绘制，即输入数据为一段或多段文本，应进行切词，关键词选择（根据频率或 `tf-idf`），继而生成词云。

```

class TextPlotter(Plotter):
    def stopwordslist(self):
        self.stopwords=[]
        with open(stopwords_file,"r",encoding='utf-8') as sw:
            for line in sw.readlines():
                self.stopwords.append(line.strip())

    def word_depart(self,filepath):
        jieba.load_userdict(r"C:\Users\86186\AppData\Local\Programs\Python\Python37\Lib\site-packages\jieba\dict1.txt")

        res={}
        self.stopwordslist()
        stopwords=self.stopwords
        with open(filepath,"r",encoding='utf-8') as f:
            pbar=tqdm(total=1000)
            for line in f.readlines():
                data1=jieba.lcut(line.strip())
                for i in data1:
                    if i not in stopwords:
                        if i in res.keys():
                            res[i] += 1
                        else:
                            res[i] = 1
                pbar.update(1)
            pbar.close()
        self.depart_res=collections.OrderedDict(sorted(res.items(),key=lambda dc:dc[1],reverse=True))

    def plot(self, data, *args, **kwargs):
        self.word_depart(data)
        wc = wordcloud.WordCloud(
            font_path='C:/Windows/Fonts/simhei.ttf',
            max_words=150,
            max_font_size=100,
            background_color='white'
        )
        wc.generate_from_frequencies(self.depart_res)
        plt.imshow(wc)
        plt.axis('off')
        plt.show()

```

```
Building prefix dict from the default dictionary ...
Loading model from cache C:\Users\86186\AppData\Local\Temp\jieba.cache
Loading model cost 0.624 seconds.
Prefix dict has been built successfully.
```

5. 实现类 ImagePlotter, 实现图片型数据的绘制, 即输入数据为图片的路径或者图片内容 (可以是多张图片), 呈现图片并按某种布局组织 (如 2x2 等)。

```
class ImagePlotter(Plotter):
    def __init__(self):
        self.info={}

        self.info["images_list"]=[]

    def load_images(self,path,**kwargs):
        self.info["path"]=path
        type_list=list(kwargs["ptype"])
        if os.path.isfile(self.info["path"]):
            a,b=os.path.split(self.info["path"])
            m,n=os.path.splitext(b)
            if n in type_list:
                self.info["images_list"].append(Image.open(self.info["path"]))
                self.info["image_type"]=b

        elif os.path.isdir(self.info["path"]):
            self.info["images_type"]=[]
            for dirpath,dirnames,files in os.walk(self.info["path"]):
                for file in files:
                    f,e = os.path.splitext(file)
                    if e in type_list:
                        self.info["images_list"].append(Image.open(os.path.join(self.info["path"],file)))
                        self.info["images_type"].append(file)
```

```
def plot(self, data, *args, **kwargs):
    self.load_images(data,**kwargs)
    if os.path.isfile(self.info["path"]):
        self.info["images_list"][0].show()
    else:
        max=len(self.info["images_list"])
        m=args[0]
        n=args[1]
        if max > m*n:
            for i in range(1,m*n+1):
                plt.subplot(m,n,i)
                plt.imshow(self.info["images_list"][i-1])
                plt.xticks([])
                plt.yticks([])
                plt.axis('off')
            plt.show()
        else:
            for i in range(1,max+1):
                plt.subplot(m,n,i)
                plt.imshow(self.info["images_list"][i-1])
                plt.xticks([])
                plt.yticks([])
                plt.axis('off')
            plt.show()
```


运行结果:



6. 实现类 GifPlotter, 支持一组图片序列的可视化 (通过文件路径或图片内容输入), 但输出是 gif 格式的动态图。

```
class GifPlotter(Plotter):
    def plot(self, data, *args, **kwargs):
        files=os.listdir(data)
        files.sort(key=lambda x:int(x[3:5]))
        length=len(files)
        frames=[]
        for i in range(length):
            frames.append(imageio.imread(os.path.join(data,files[i])))
        imageio.mimsave(args[0], frames, 'GIF', duration =args[1])
```

运行结果:

 result.gif	2021/12/4 13:20	GIF 文件	5,347 KB
--	-----------------	--------	----------

见具体文件

附加 9: 在 6 中, 如果输入是一段落视频的话, 能否通过帧采样, 将视频绘制为 gif 并输出为微信表情包? (了解 cv2)

利用 cv2 对视频进行逐帧读取, 按照需要的间隔对视频进行保存

```
class video_to_gif_Plotter(Plotter):
    def cv_imwrite(self,file_path, frame):
        cv2.imencode('.jpg', frame)[1].tofile(file_path)

    def plot(self, data, *args, **kwargs):
        vc = cv2.VideoCapture(data)
        n = 1

        if vc.isOpened():
            rval, frame = vc.read()
        else:
            rval = False

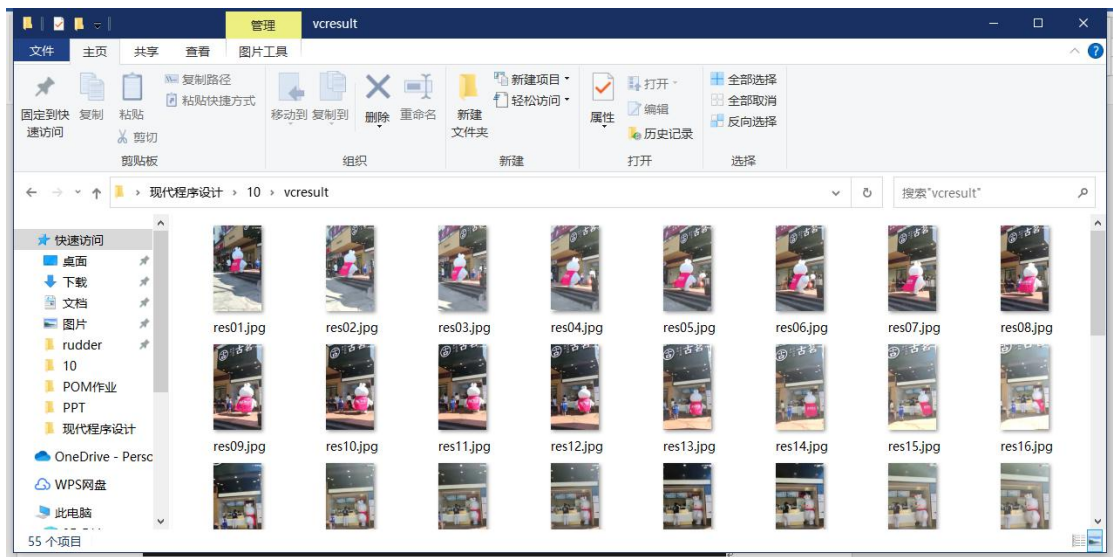
        timeF = args[0]

        i = 0
        while(rval):
            if (n % timeF == 0):
                i += 1
                self.cv_imwrite(os.path.join(vcreresultpath,"res{:0=2}.jpg".format(i)),frame)
                n = n + 1
                rval, frame = vc.read()
            vc.release()

        duration=timeF/60
        p=GifPlotter()
        p.plot(vcreresultpath,"result.gif",duration)
```

运行结果:

截取的内容



生成的 gif 动画表情:

