ماشین بردار پشتیبان

همان طور که در صورت سوال گفته شد ، از ابزار ها و کتاب خانه های آماده svm استفاده میکنیم.

در قسمت اول فقط سعی میکنیم که دو نوع نقاط که با رنگ های آبی و قرمز مشخص کردیم را در صفحه خودمان از هم جدا کنیم. (همان طور که گفته شده نقاط را در دو کلاس فرضی دسته بندی کردیم)

یعنی ابتدا نقاط را تعریف کرده و دسته بندی میکنیم سپس آنها را در صفحه از هم جدا میکنیم.

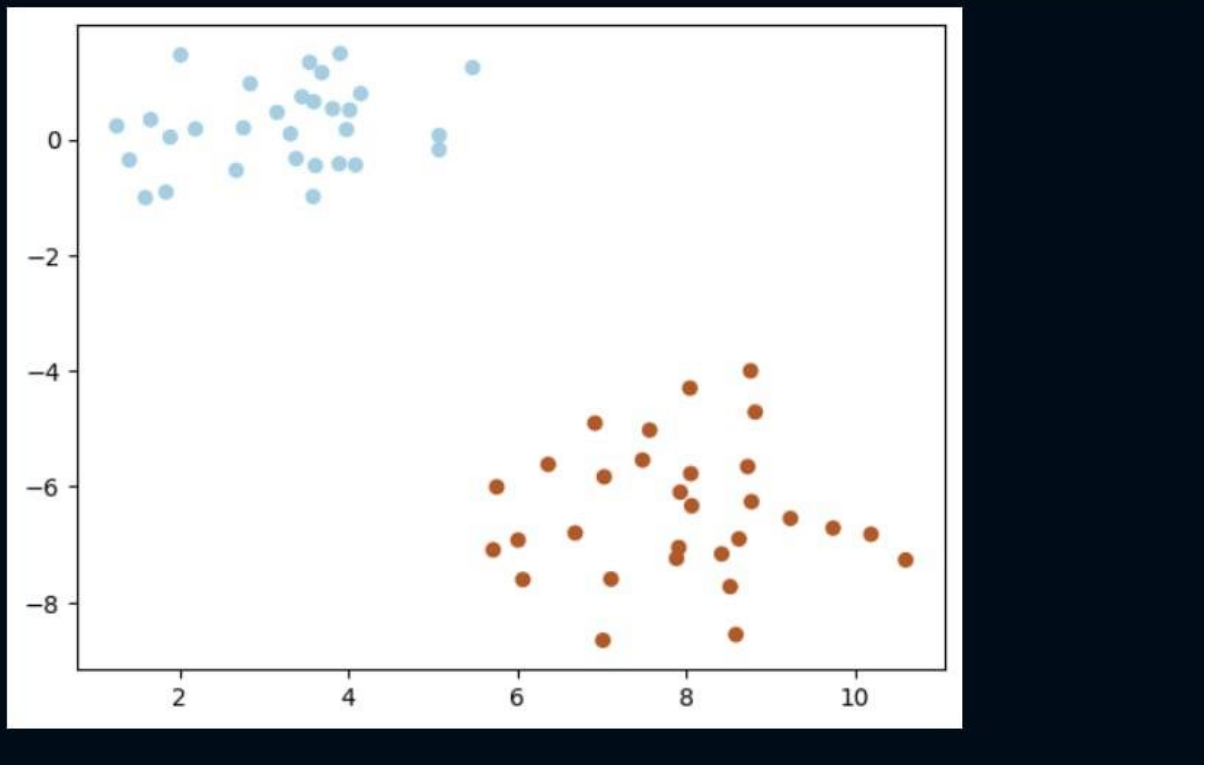
```python
x, y = make_blobs(n_samples=60, centers=2, random_state=18) #centers = anva , n_samples = tedad nemune

m_svm = svm.SVC(kernel='linear', C=1)
m_svm.fit(x, y)

plt.scatter(x[:, 0], x[:, 1], c=y, s=30, cmap=plt.cm.Paired)
plt.show()
```

✓ 11.9s Python

22] ✓ 11.9s



سپس خط جدا کننده که svm یافته است را رسم میکنیم و در ادامه خط مربوط به margin را هم نمایش میدهیم.

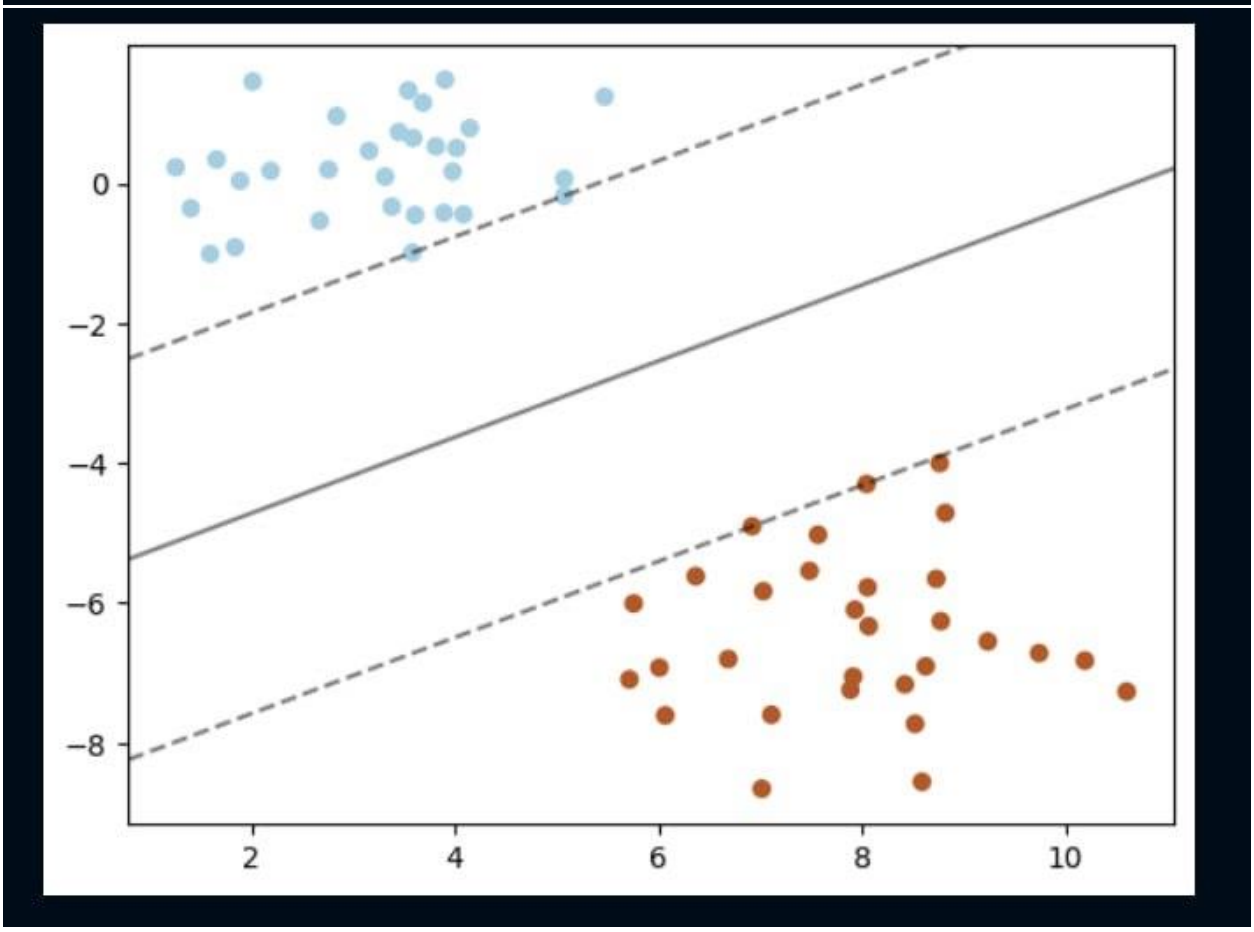
```python
svm_me = svm.SVC(kernel='linear', C=1000)
svm_me.fit(x, y)
plt.scatter(x[:, 0], x[:, 1], c=y, s=30, cmap=plt.cm.Paired)

ax = plt.gca()
xlim = ax.get_xlim()
ylim = ax.get_ylim()
xx = np.linspace(xlim[0], xlim[1], 30)
yy = np.linspace(ylim[0], ylim[1], 30)
YY, XX = np.meshgrid(yy, xx)
xy = np.vstack([XX.ravel(), YY.ravel()]).T
Z = svm_me.decision_function(xy).reshape(XX.shape)

ax.contour(XX, YY, Z, colors='k', levels = [-1,0,1], alpha=0.5
           , linestyles=['--', '-', '--'])

ax.scatter(svm_me.support_vectors_[:, 0], svm_me.support_vectors_[:, 1],
           s = 100, linewidth=1, facecolors='none')
plt.show()
```

3] ✓ 2.1s



در بخش دوم ابتدا یک پایگاه داده برای خودمان تعریف میکنیم. (در اینجا پایگاه داده ما دو نوع از عکس هایی است که در فایل زیپ مربوط به صورت سوال بودند).

در ادامه در کدی که داریم با imread عکس ها را میخوانیم. چون اندازه عکس ها یکسان است نیازی به کد resize نیست). بعد بر روی همین عکس ها یک flatten میزنیم. در ادامه مشاهده میکنیم که عکس های ما به لیستی از دیتا ها تبدیل شده اند.

در اینجا ما فقط از عکس هایی با شماره های صفر و ۲ استفاده کردیم برای همین یک if تعریف کرده و میبینیم اگر عکس ما با ۰_ آغاز میشد که آنرا در آرایه ما برابر صفر قرار دهد و اگر با ۲_ شروع میشد در آرایه ما آن را برابر ۲ قرار دهد (در انتها کد این آرایه را نمایش میدهیم)

```
path = "img"
contents = os.listdir(path)
array_of_numbers = []
value_of_numbers = []
for i in range(len(contents)):
    mem = imread(os.path.join('img',f"{contents[i]}"))
    flat = mem.flatten()
    array_of_numbers.append(flat)
    if '0_' in contents[i] :
      value_of_numbers.append(0)
    elif '2_' in contents[i] :
      value_of_numbers.append(2)

#print(array_of_numbers)
arr = np.array(array_of_numbers)
print(arr.shape)
print(value_of_numbers)
```

[24] ✓ 3.9s

در انتها تعداد عکس ها که برابر ۱۶۸ است و تعداد پیکسل عکس ها که برابر ۲۵۶ است را نمایش میدهیم

```
(168, 256)
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

یک نکته دیگر این که در زمانی که قرار است کد ما عکس ها را تشخیص دهد که ۰ است یا ۲ پیکسل های صفید را با عدد ۰ و پیکسل های مشکی را با عدد ۱ برای خودش مشخص میکند.