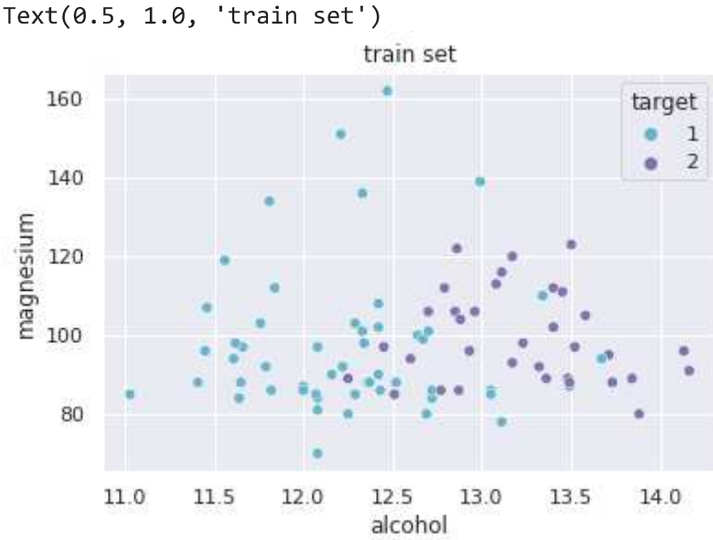
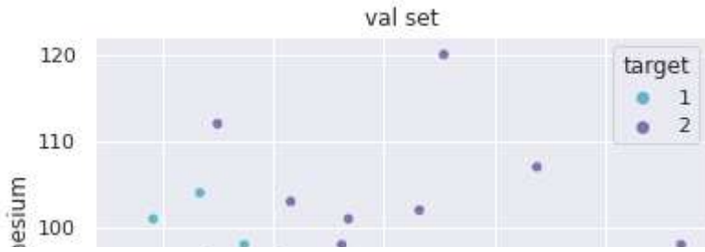
```
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
import pandas as pd

from sklearn.model_selection import train_test_split
# use seaborn plotting defaults
import seaborn as sns; sns.set()


from sklearn.datasets import load_wine
# Read the wine dataset
dataset = load_wine()
df = pd.DataFrame(data=dataset['data'], columns=dataset['feature_names'])
df = df.assign(target=pd.Series(dataset['target']).values)


# Filter the irrelevant columns
df = df[['alcohol', 'magnesium', 'target']]
# Filter the irrelevant label
df = df[df.target != 0]



from sklearn.svm import SVC


train_df, val_df = train_test_split(df, test_size=30, random_state=3)




#1.1
sns.scatterplot(data=train_df, x='alcohol', y='magnesium', hue='target',palette=['c','m'])
plt.title("train set")
```

```
Text(0.5, 1.0, 'train set')
```




```
#1.2
sns.scatterplot(data=val_df, x='alcohol', y='magnesium', hue='target',palette=['c','m'])
plt.title("val set")
```

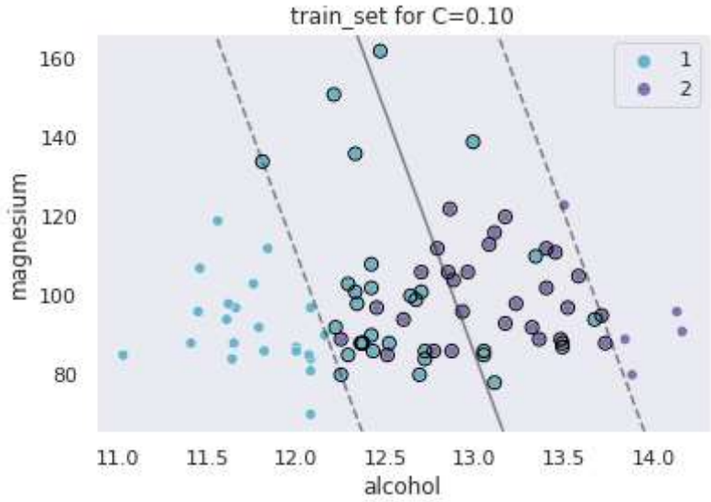
```
Text(0.5, 1.0, 'val set')
```

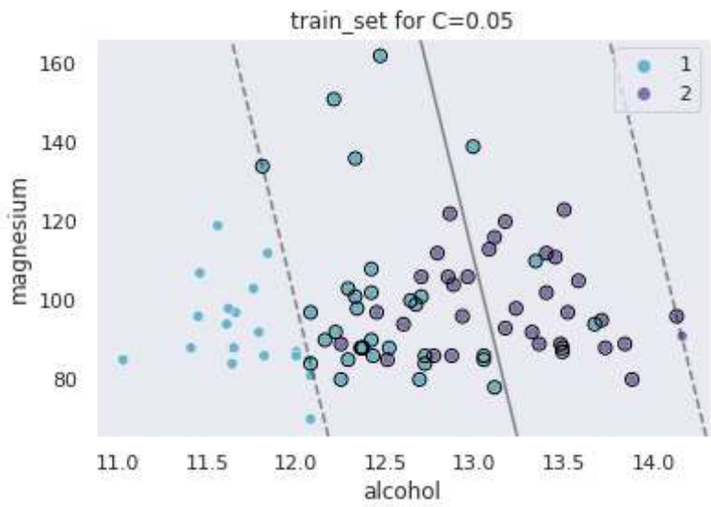


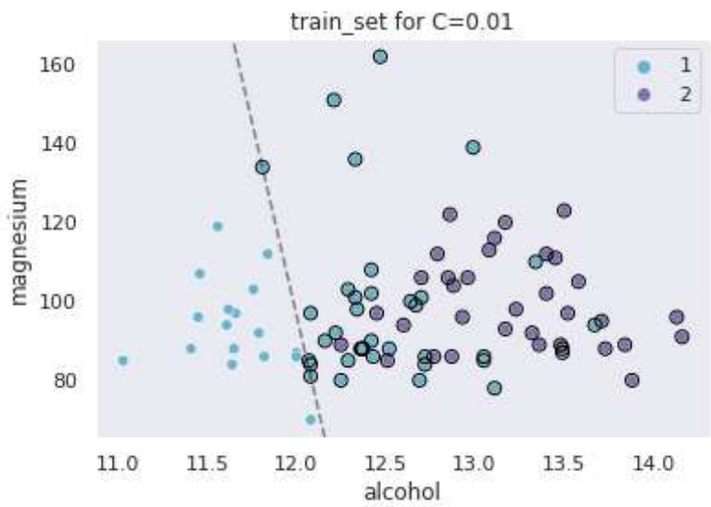
מכיוון שהנתונים שלנו לא פריד לינארי ולכן אם נפעיל את האלגורתים -1 לא נוכל להפעיל האלגורתים hard-SVM לא נקבל פתרון


```
#2
from sklearn.svm import SVC

def plot_svc_decision_function(model, ax=None, plot_support=True):
    """Plot the decision function for a 2D SVC"""
    if ax is None:
        ax = plt.gca()
    xlim = ax.get_xlim()
    ylim = ax.get_ylim()

    # create grid to evaluate model
    x = np.linspace(xlim[0], xlim[1], 30)
    y = np.linspace(ylim[0], ylim[1], 30)
    Y, X = np.meshgrid(y, x)
    xy = np.vstack([X.ravel(), Y.ravel()]).T
    P = model.decision_function(xy).reshape(X.shape)

    # plot decision boundary and margins
    ax.contour(X, Y, P, colors='k',
               levels=[-1, 0, 1], alpha=0.5,
               linestyles=['--', '-', '--'])

    # plot support vectors
    if plot_support:
        ax.scatter(model.support_vectors_[:, 0],
                   model.support_vectors_[:, 1],
                   s=50, linewidth=1, facecolors='none', edgecolor='black');
    ax.set_xlim(xlim)
    ax.set_ylim(ylim)
```

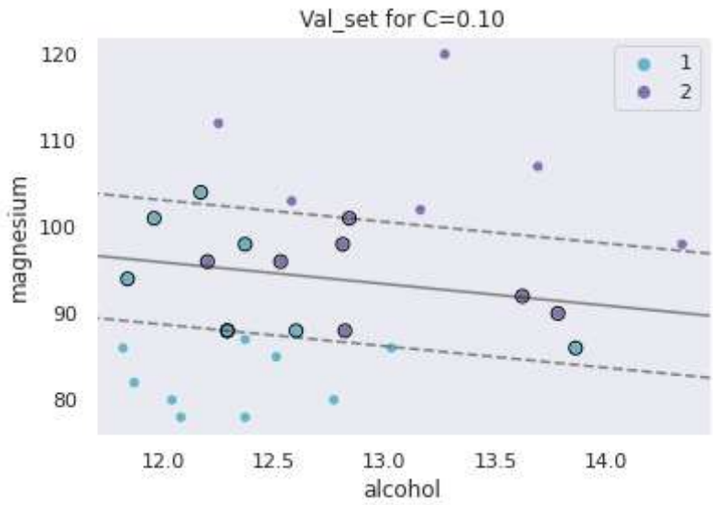
```
#2.1
C=[0.1, 0.05, 0.01]
for c in C:
  model = SVC(kernel='linear',C=c)

  sns.scatterplot(data=train_df, x="alcohol", y="magnesium", hue="target",palette=['c','m'])
  X_t=train_df.to_numpy()
  y_t=X_t[:,-1]
  X_t=np.delete(X_t,2,1)
  model.fit(X_t,y_t)

  plot_svc_decision_function(model)
  plt.legend()
  s1= "train_set for C=%.2f"
  plt.title(s1%c)
  plt.grid()
  plt.show()
```

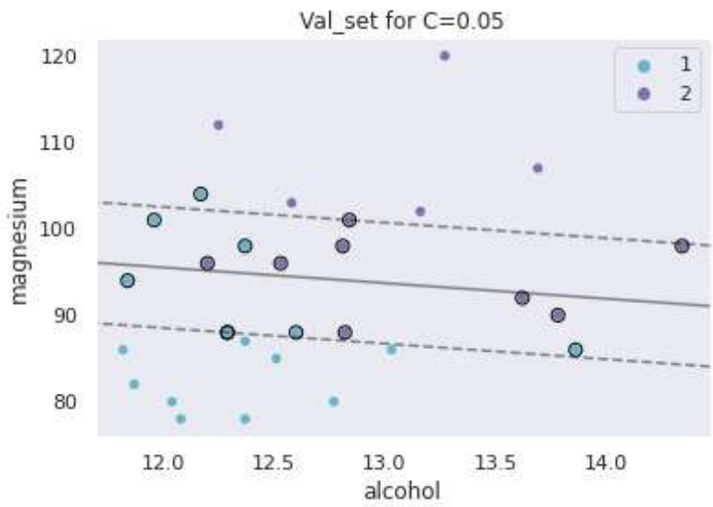


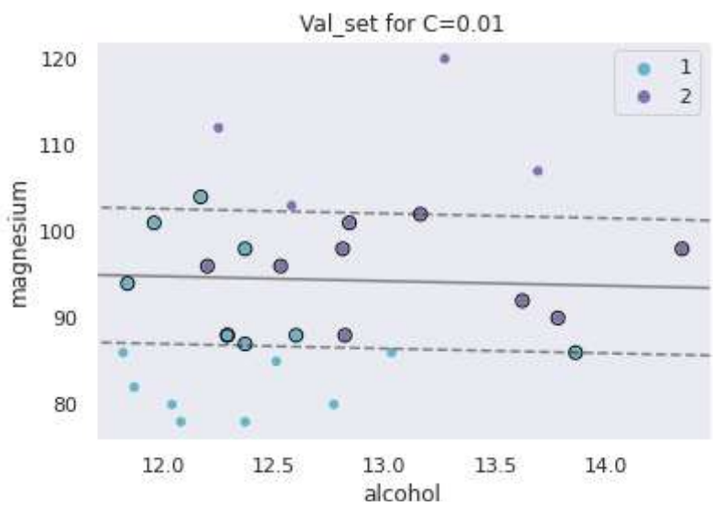





```
#2.2
C=[0.1, 0.05, 0.01]
for c in C:
  model = SVC(kernel='linear',C=c)

  sns.scatterplot(data=val_df, x="alcohol", y="magnesium", hue="target",palette=['c','m'])
  X_V=val_df.to_numpy()
  y_V=X_V[:,-1]
  X_V=np.delete(X_V,2,1)
  model.fit(X_V,y_V)

  plot_svc_decision_function(model)
  plt.legend()
  s1= "Val_set for C=%.2f"
  plt.title(s1%c)
  plt.grid()
  plt.show()
```



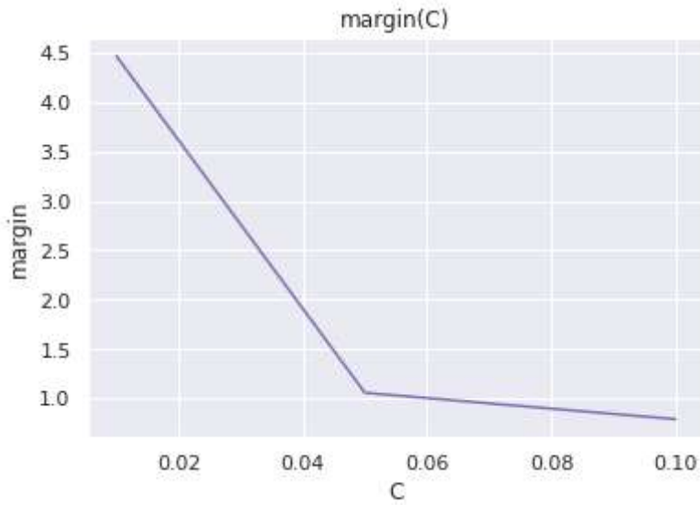




Double-click (or enter) to edit

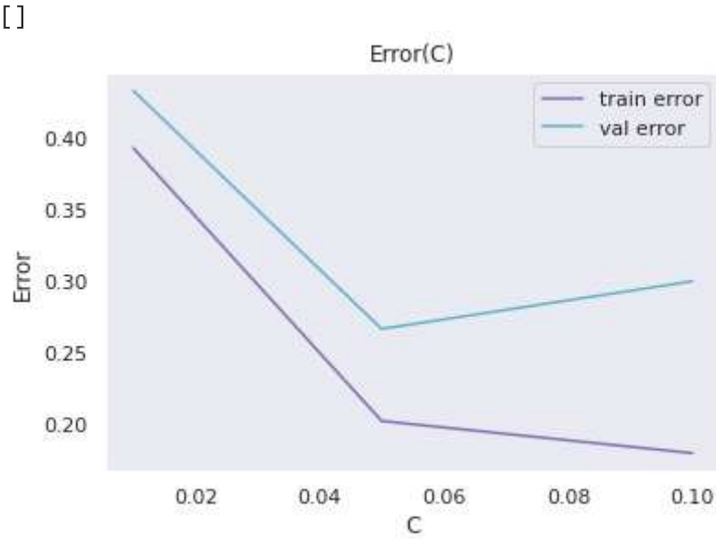
```
#4
C=[0.1, 0.05, 0.01]
train=[]
val=[]
margin=[]
for c in C:
    model=SVC(kernel='linear',C=c)
    model.fit(train_df[['alcohol','magnesium']],train_df['target'])
    train.append(model.score(train_df[['alcohol','magnesium']],train_df['target']))
    margin.append(1/(np.linalg.norm(model.coef_)))
fig,ax=plt.subplots()

plt.title('margin(C)')
plt.xlabel('C')
plt.ylabel('margin')
ax.plot(C,margin,c='m')
```

```
plt.grid
plt.show()
```




```
#5

train_error=[]
val_error=[]
for c in C:
    model=SVC(kernel='linear',C=c)
    model.fit(train_df[['alcohol','magnesium']],train_df['target'])
    train_error.append(1-model.score(train_df[['alcohol','magnesium']],train_df['target']))
    val_error.append(1-model.score(val_df[['alcohol','magnesium']],val_df['target']))
fig,ax=plt.subplots()
plt.title('Error(C)')
plt.xlabel('C')
plt.ylabel("Error")
ax.plot(C,train_error,c='m',label='train error')
ax.plot(C,val_error,c='c' ,label='val error')
plt.grid()
plt.legend()
plt.plot()
```

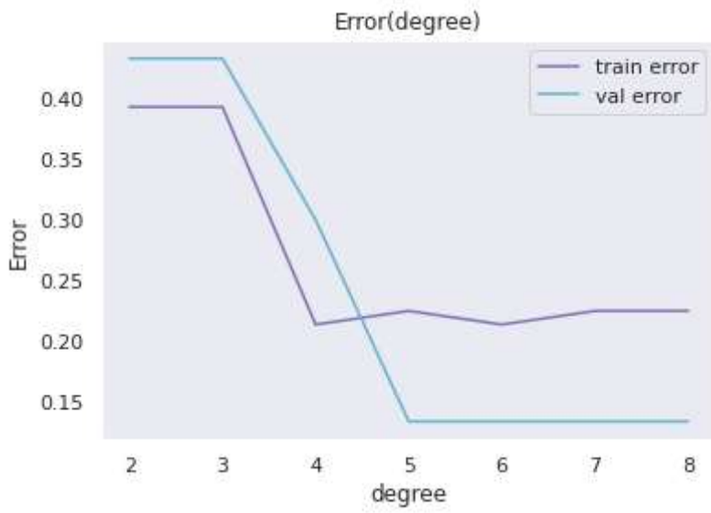
```
[]
```



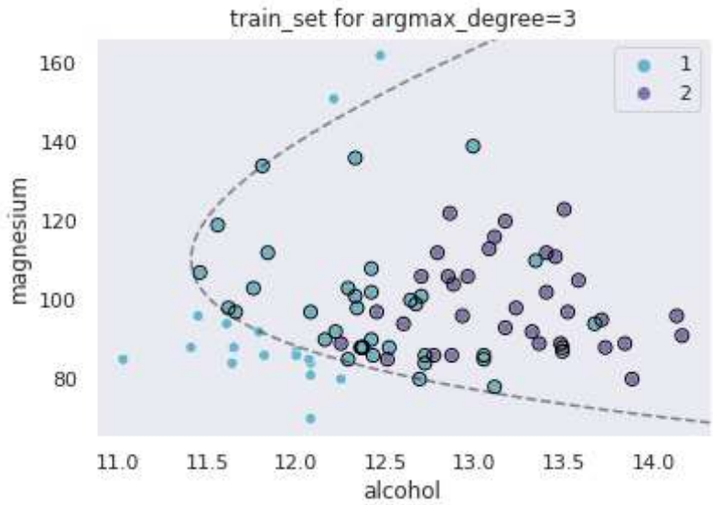

```
#6
degree=[2,3,4,5,6,7,8]
train_error=[]
val_error=[]
for d in degree:
    model=SVC(kernel='poly',C=1,degree=d)
    model.fit(train_df[['alcohol','magnesium']],train_df['target'])
    train_error.append(1-model.score(train_df[['alcohol','magnesium']],train_df['target']))
    val_error.append(1-model.score(val_df[['alcohol','magnesium']],val_df['target']))
```

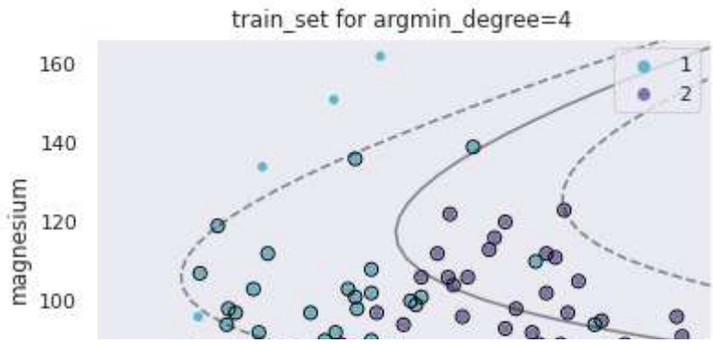
```
fig,ax=plt.subplots()
plt.title('Error(degree)')
plt.xlabel('degree')
plt.ylabel("Error")
ax.plot(degree,train_error,c='m',label='train error')
ax.plot(degree,val_error,c='c' ,label='val error')
plt.grid()
plt.legend()
plt.plot()
```

```
[]
```

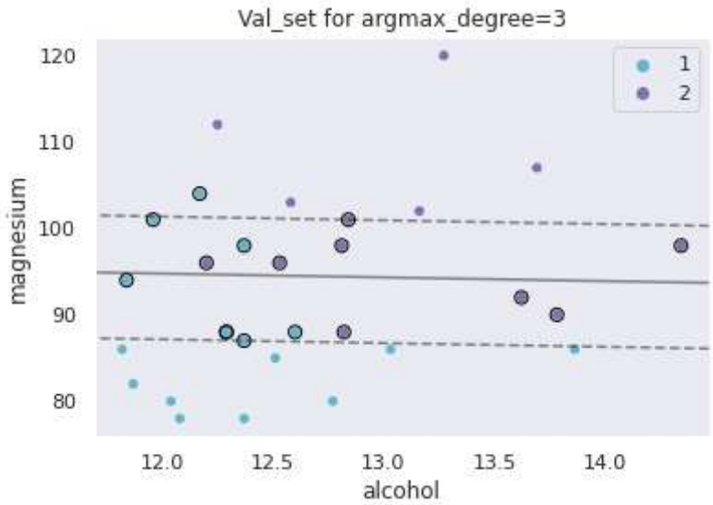


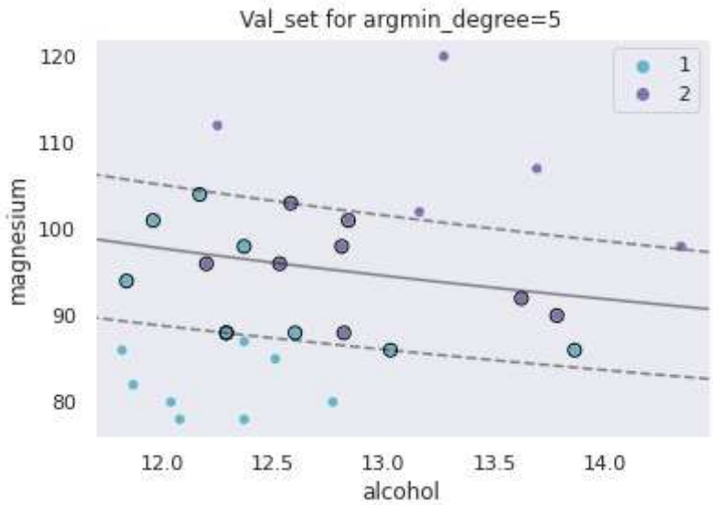

```
#7.1
model = SVC(kernel='poly',degree=3,C=1)

sns.scatterplot(data=train_df, x="alcohol", y="magnesium", hue="target",palette=['c','m'])
X_t=train_df.to_numpy()
y_t=X_t[:,-1]
X_t=np.delete(X_t,2,1)
model.fit(X_t,y_t)

plot_svc_decision_function(model)
plt.legend()

plt.title('train_set for argmax_degree=3')
plt.grid()
plt.show()

#7
model = SVC(kernel='poly',degree=4,C=1)

sns.scatterplot(data=train_df, x="alcohol", y="magnesium", hue="target",palette=['c','m'])
X_t=train_df.to_numpy()
y_t=X_t[:,-1]
X_t=np.delete(X_t,2,1)
model.fit(X_t,y_t)

plot_svc_decision_function(model)
plt.legend()

plt.title('train_set for argmin_degree=4')
plt.grid()
plt.show()
```






```
#7.2
#7
model = SVC(kernel='poly',degree=3,C=1)

sns.scatterplot(data=val_df, x="alcohol", y="magnesium", hue="target",palette=['c','m'])
X_V=val_df.to_numpy()
y_V=X_V[:,-1]
X_V=np.delete(X_V,2,1)
model.fit(X_V,y_V)

plot_svc_decision_function(model)
plt.legend()

plt.title('Val_set for argmax_degree=3')
plt.grid()
plt.show()

model2 = SVC(kernel='poly',degree=5,C=1)

sns.scatterplot(data=val_df, x="alcohol", y="magnesium", hue="target",palette=['c','m'])
X_V=val_df.to_numpy()
y_V=X_V[:,-1]
X_V=np.delete(X_V,2,1)
model2.fit(X_V,y_V)

plot_svc_decision_function(model2)
plt.legend()

plt.title('Val_set for argmin_degree=5')
plt.grid()
plt.show()
```


Val_set for argmax_degree=3
magnesium
alcohol
1
2
120
110
100
90
80
12.0
12.5
13.0
13.5
14.0


Val_set for argmin_degree=5
magnesium
alcohol
1
2
120
110
100
90
80
12.0
12.5
13.0
13.5
14.0