In [1]:
```
#Load Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:
```
#Load Dataset
rice=pd.read_csv('./ricedataset.csv')
rice.head()
```

Out[2]:

	AREA	PERIMETER	MAJORAXIS	MINORAXIS	ECCENTRICITY	CONVEX_AREA	EXTENT	CLASS
0	15231	525.578979	229.749878	85.093788	0.928882	15617	0.572896	Cammeo
1	14656	494.311005	206.020065	91.730972	0.895405	15072	0.615436	Cammeo
2	14634	501.122009	214.106781	87.768288	0.912118	14954	0.693259	Cammeo
3	13176	458.342987	193.337387	87.448395	0.891861	13368	0.640669	Cammeo
4	14688	507.166992	211.743378	89.312454	0.906691	15262	0.646024	Cammeo

In [3]:
```
rice.describe()
```

Out[3]:

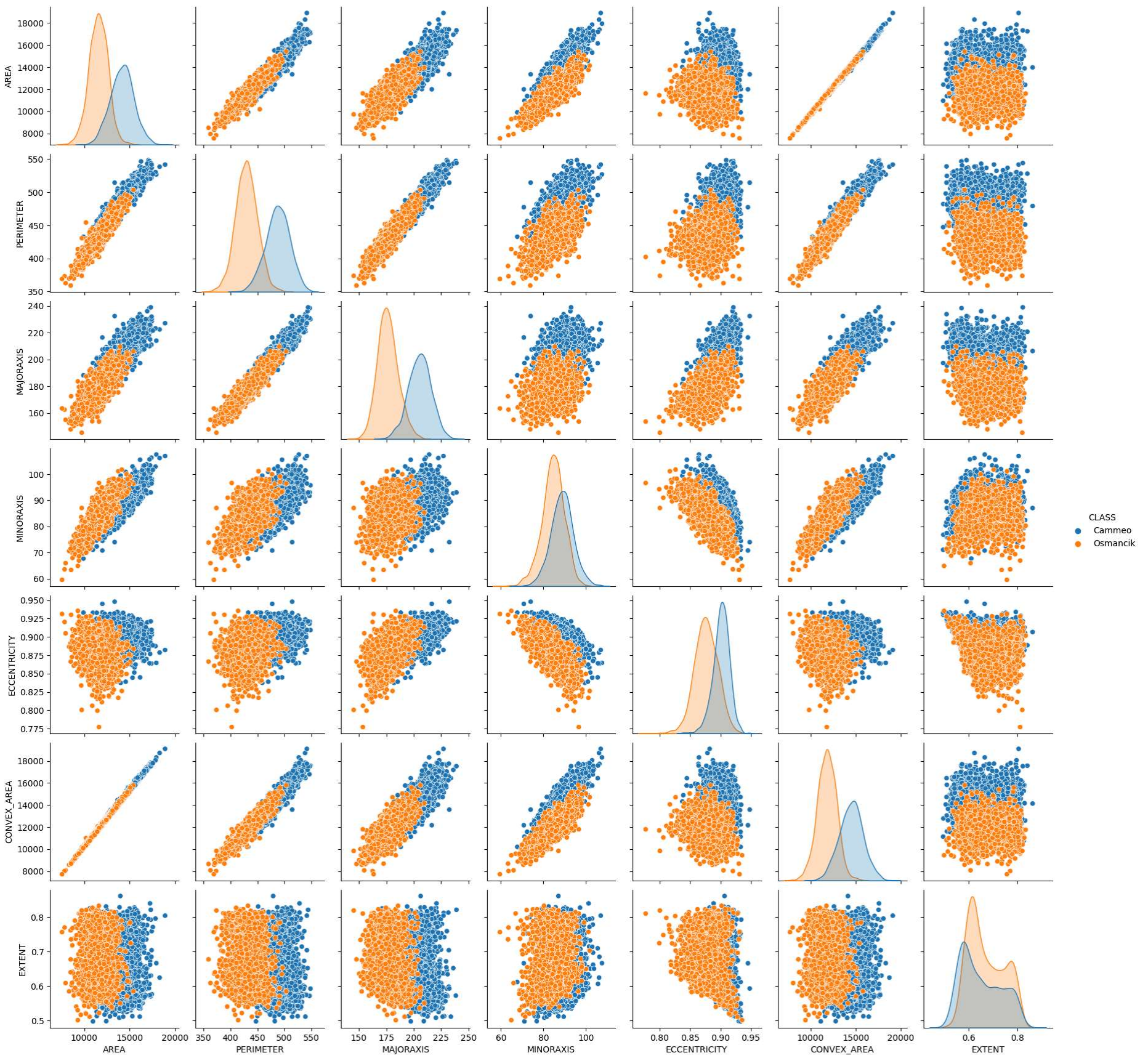
	AREA	PERIMETER	MAJORAXIS	MINORAXIS	ECCENTRICITY	CONVEX_AREA	EXTENT
count	3810.000000	3810.000000	3810.000000	3810.000000	3810.000000	3810.000000	3810.000000
mean	12667.727559	454.239180	188.776222	86.313750	0.886871	12952.496850	0.661934
std	1732.367706	35.597081	17.448679	5.729817	0.020818	1776.972042	0.077239
min	7551.000000	359.100006	145.264465	59.532406	0.777233	7723.000000	0.497413
25%	11370.500000	426.144752	174.353855	82.731695	0.872402	11626.250000	0.598862
50%	12421.500000	448.852493	185.810059	86.434647	0.889050	12706.500000	0.645361
75%	13950.000000	483.683746	203.550438	90.143677	0.902588	14284.000000	0.726562
max	18913.000000	548.445984	239.010498	107.542450	0.948007	19099.000000	0.861050

In [4]:
```
#creation of new dataframe for mapping class with integer values for correlation with independent variables.
df = rice.copy()
df['CLASS'] = df['CLASS'].map({'Cammeo':1,'Osmancik':0})
fig = plt.figure(figsize=(15,5))
sns.heatmap(df.corr(),annot=True,cmap="Blues")
```

Out[4]:
```
<AxesSubplot: >
```

In [5]:
```
# Pairplot with hue=species
sns.pairplot(rice, hue ='CLASS')
```

Out[5]:
```
<seaborn.axisgrid.PairGrid at 0x15d3debfee0>
```

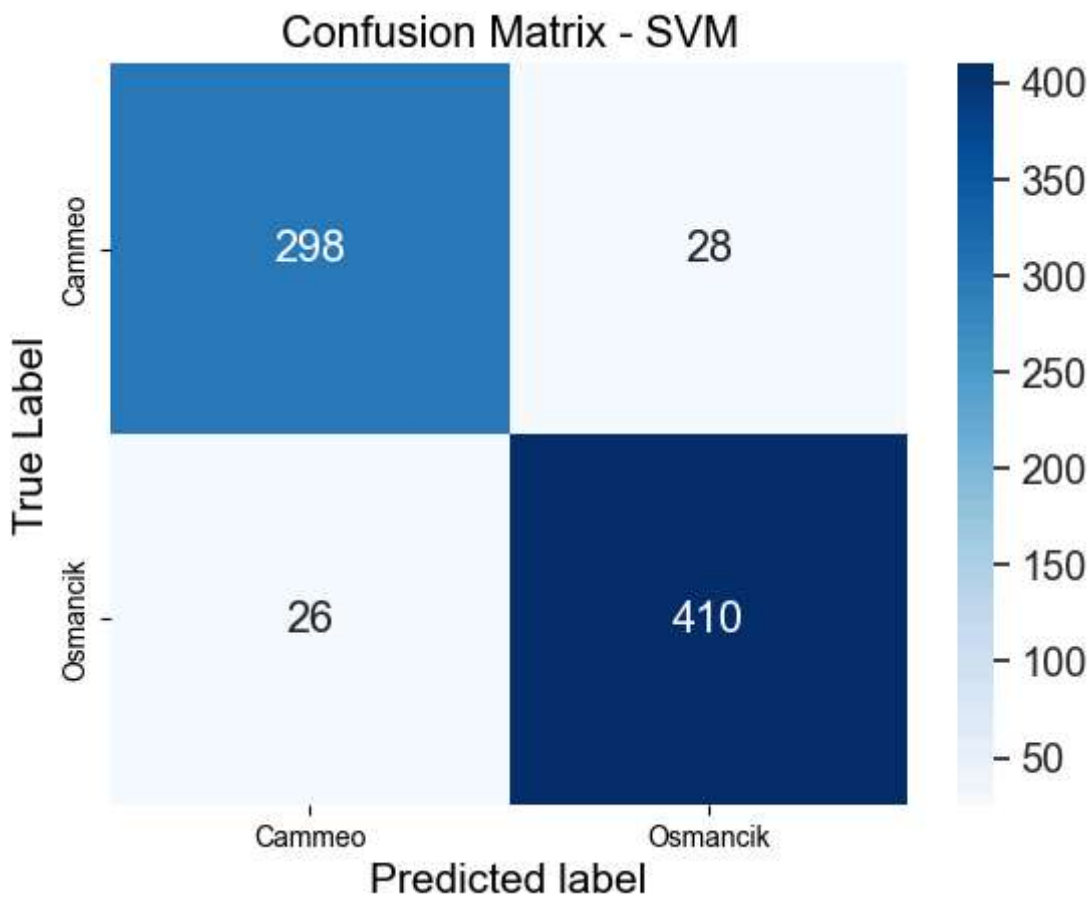


In [6]:
```python
#Create x and y variables
x=rice.drop('CLASS', axis=1).to_numpy()
y=rice['CLASS'].to_numpy()

#Create Training and Test Datasets
from sklearn.model_selection import train_test_split
x_train, x_test,y_train, y_test = train_test_split(x, y, stratify=y,test_size=0.2,random_state=100)

#Scale the Data
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train2 = sc.fit_transform(x_train)
x_test2 = sc.transform(x_test)
```

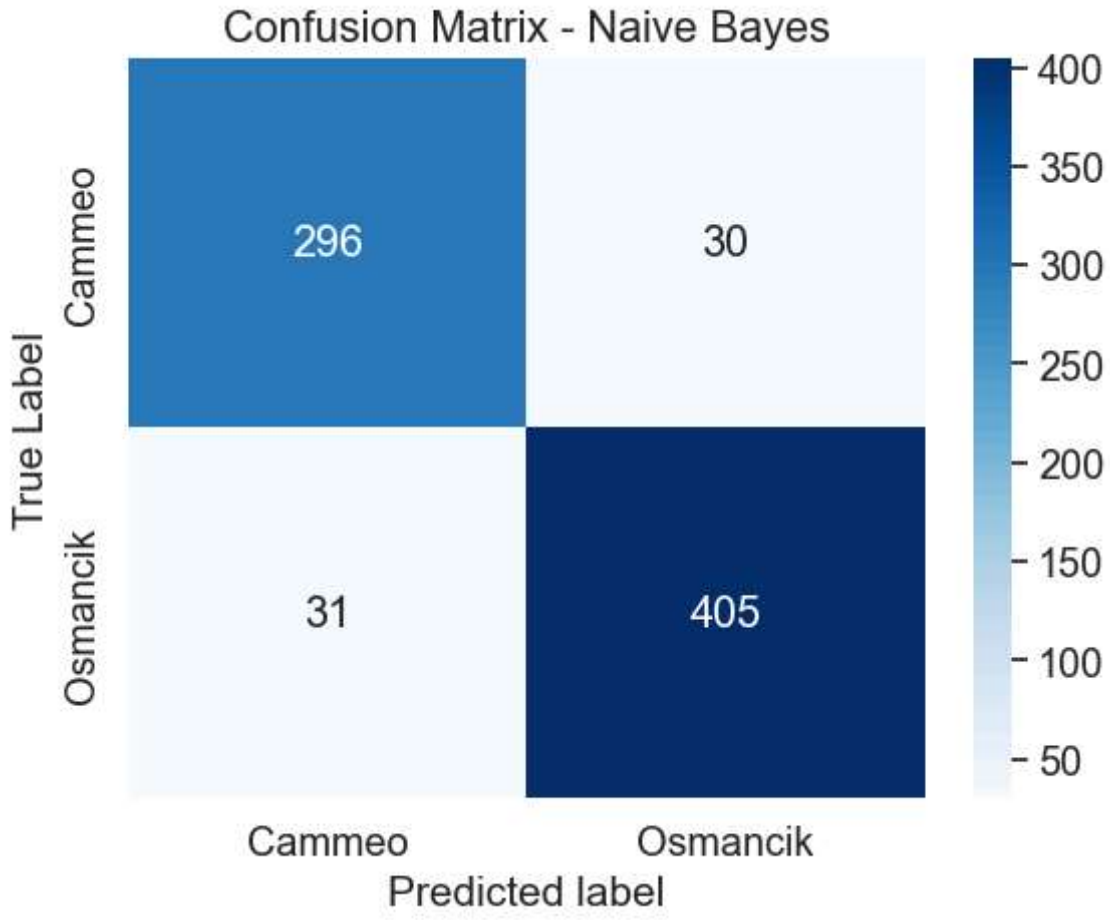
In [7]:
```python
#Script for SVM and NB
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report, confusion_matrix

for name,method in [('SVM', SVC(random_state=100)),
                    ('Naive Bayes',GaussianNB())]:
    method.fit(x_train2,y_train)
    predict = method.predict(x_test2)
    target_names=['Cammeo','Osmancik']
    print('\nEstimator: {}'.format(name))
    print(confusion_matrix(y_test,predict))
    print(classification_report(y_test,predict,target_names=target_names))
    matrix_df = pd.DataFrame(confusion_matrix(y_test,predict))
    #plot the result
    ax = plt.axes()
    sns.set(font_scale=1.3)
    plt.figure(figsize=(10,7))
    sns.heatmap(matrix_df, annot=True, fmt="g", ax=ax, cmap="Blues")
    #set axis titles
    ax.set_title('Confusion Matrix - ' + (name))
    ax.set_xlabel("Predicted label", fontsize =15)
    ax.xaxis.set_ticklabels(target_names)
    ax.set_ylabel("True Label", fontsize=15)
```

```
    ax.yaxis.set_ticklabels(target_names)
    plt.show()
```

```
Estimator: SVM
[[298  28]
 [ 26 410]]
              precision    recall  f1-score   support

      Cammeo       0.92      0.91      0.92       326
    Osmancik       0.94      0.94      0.94       436

    accuracy                           0.93       762
   macro avg       0.93      0.93      0.93       762
weighted avg       0.93      0.93      0.93       762
```




```
<Figure size 1000x700 with 0 Axes>
Estimator: Naive Bayes
[[296  30]
 [ 31 405]]
              precision    recall  f1-score   support

      Cammeo       0.91      0.91      0.91       326
    Osmancik       0.93      0.93      0.93       436

    accuracy                           0.92       762
   macro avg       0.92      0.92      0.92       762
weighted avg       0.92      0.92      0.92       762
```

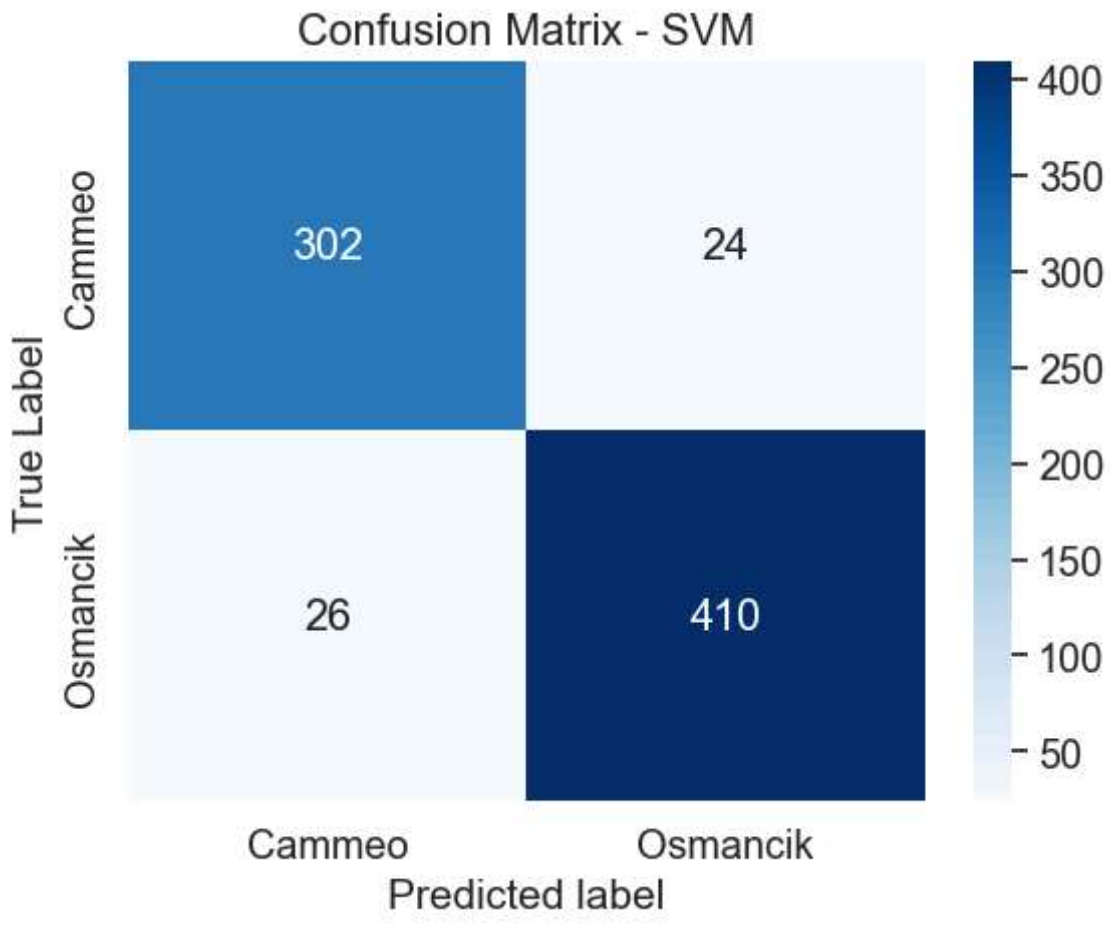



```
<Figure size 1000x700 with 0 Axes>
```

In [8]:
```
#Script for SVM with kernel as linear
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix

for name,method in [('SVM', SVC(kernel='linear', random_state=100))]:
```

```
    method.fit(x_train2,y_train)
    predict = method.predict(x_test2)
    target_names=['Cammeo','Osmancik']
    print('\nEstimator: {}'.format(name))
    print(confusion_matrix(y_test,predict))
    print(classification_report(y_test,predict,target_names=target_names))
    matrix_df = pd.DataFrame(confusion_matrix(y_test,predict))
    #plot the result
    ax = plt.axes()
    sns.set(font_scale=1.3)
    plt.figure(figsize=(10,7))
    sns.heatmap(matrix_df, annot=True, fmt="g", ax=ax, cmap="Blues")
    #set axis titles
    ax.set_title('Confusion Matrix - ' + (name))
    ax.set_xlabel("Predicted label", fontsize =15)
    ax.xaxis.set_ticklabels(target_names)
    ax.set_ylabel("True Label", fontsize=15)
    ax.yaxis.set_ticklabels(target_names)
    plt.show()
```

```
Estimator: SVM
[[302  24]
 [ 26 410]]
              precision    recall  f1-score   support

     Cammeo       0.92      0.93      0.92       326
   Osmancik       0.94      0.94      0.94       436

   accuracy                           0.93       762
  macro avg       0.93      0.93      0.93       762
weighted avg       0.93      0.93      0.93       762
```




```
<Figure size 1000x700 with 0 Axes>
```

In [ ]: