In [1]:
```
import pandas as pd
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import ConfusionMatrixDisplay,accuracy_score
from sklearn.metrics import classification_report
import numpy as nm
import matplotlib.pyplot as mtp
from sklearn import svm
```

In [3]:
```
df=pd.read_csv('email.csv')
df.head(7)
```

Out[3]:

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastr
0	Email 1	0	0	1	0	0	0	2	0	0	...	0	0	0	0	
1	Email 2	8	13	24	6	6	2	102	1	27	...	0	0	0	0	
2	Email 3	0	0	1	0	0	0	8	0	0	...	0	0	0	0	
3	Email 4	0	5	22	0	5	1	51	2	10	...	0	0	0	0	
4	Email 5	7	6	17	1	5	2	57	0	9	...	0	0	0	0	
5	Email 6	4	5	1	4	2	3	45	1	0	...	0	0	0	0	
6	Email 7	5	3	1	3	2	1	37	0	0	...	0	0	0	0	

7 rows × 3002 columns

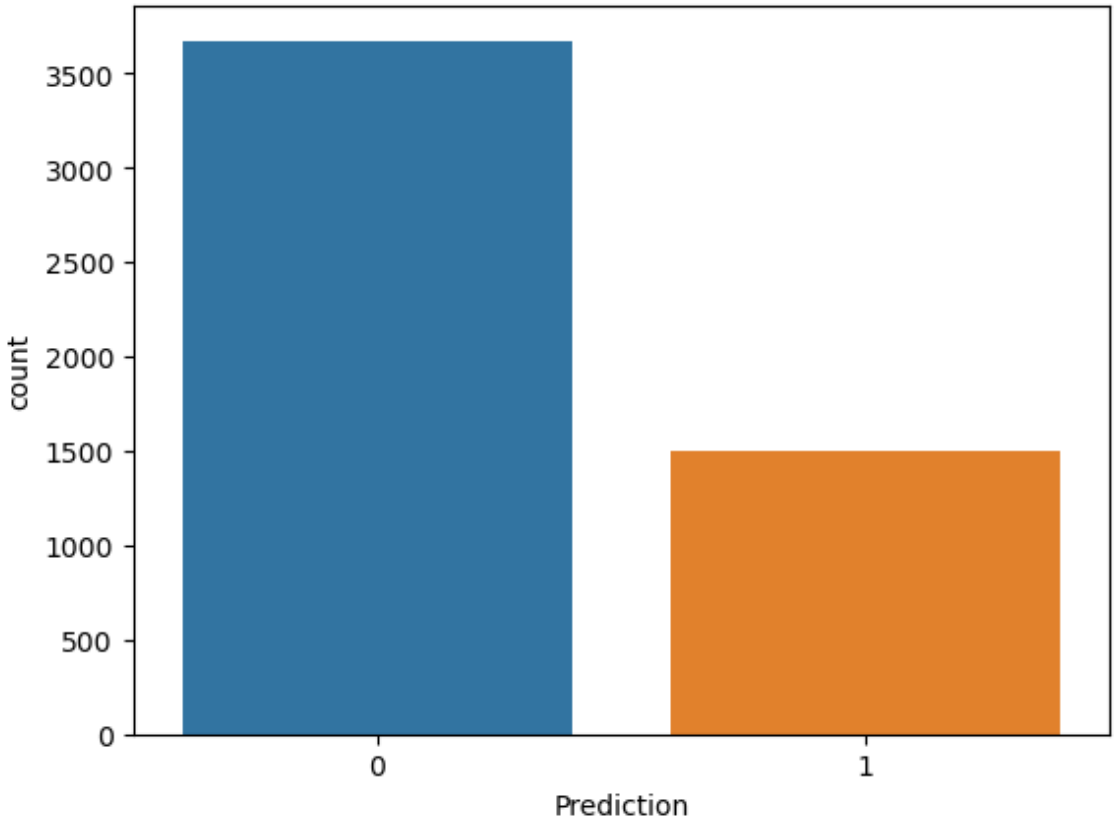
In [4]:
```
df.shape
```

Out[4]:
```
(5172, 3002)
```

In [5]:
```
x=df.drop(['Email No.','Prediction'],axis=1)
y=df['Prediction']
```

In [6]:
```
sns.countplot(x=y)
```

Out[6]:
```
<AxesSubplot:xlabel='Prediction', ylabel='count'>
```



In [7]:
```
y.value_counts()
```

Out[7]:
```
0    3672
1    1500
Name: Prediction, dtype: int64
```

In [8]:
```
scaler=MinMaxScaler();
result=scaler.fit_transform(x)
```

In [9]:
```
X_train,X_test,Y_train,Y_test=train_test_split(result,y,random_state=0,test
```

K Nearest Neighbor

In [10]:
```
knn=KNeighborsClassifier(n_neighbors=5)
```

In [11]:
```
knn.fit(X_train,Y_train)
```

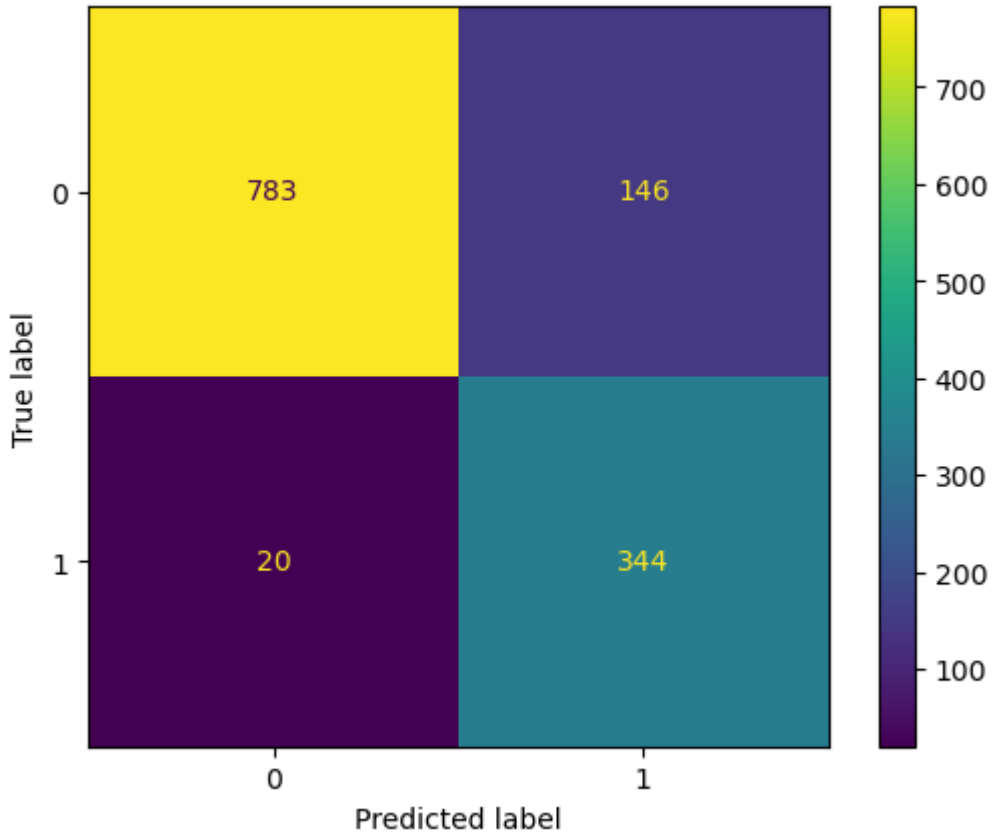
Out[11]:
```
KNeighborsClassifier()
```

In [12]:
```
ypred=knn.predict(X_test)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\neighbors\_classificati
on.py:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `
kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value
of `keepdims` will become False, the `axis` over which the statistic is ta
ken will be eliminated, and the value None will no longer be accepted. Set
`keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [13]:
```
ConfusionMatrixDisplay.from_predictions(Y_test, ypred)
```

Out[13]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1f3acb
697c0>
```



In [14]:
```
accuracy_score(Y_test, ypred)
```

Out[14]:
```
0.871616395978345
```

In [15]:
```
print(classification_report(Y_test, ypred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.84      0.90       929
           1       0.70      0.95      0.81       364

    accuracy                           0.87      1293
   macro avg       0.84      0.89      0.85      1293
weighted avg       0.90      0.87      0.88      1293
```

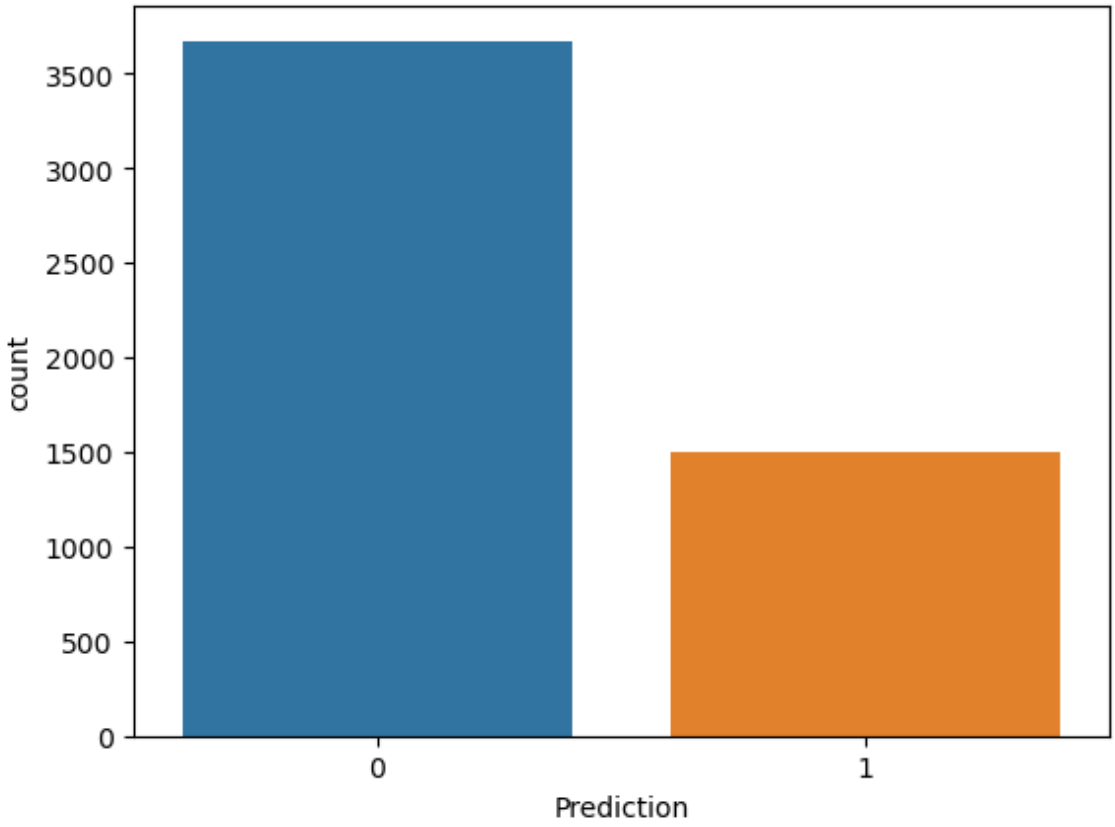
Support Vector Machine

In [19]:
```
print(df.shape)
```

```
(5172, 3002)
```

In [40]:
```
x = df.drop(['Email No.', 'Prediction'], axis=1)
y = df['Prediction']
```

In [21]:
```
sns.countplot(x=y)
plt.show()
```



In [41]:
```
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(x)
```

In [42]:
```
feature1_index = 0
feature2_index = 1
```

In [43]:
```
X_selected_features = X_scaled[:, [feature1_index, feature2_index]]
```

In [44]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X_selected_features, y,
```

In [71]:
```
svm_classifier = svm.SVC(kernel='linear')  # You can choose a different ker
```

In [72]:
```
svm_classifier.fit(X_train, Y_train)
```

Out[72]:
```
SVC(kernel='linear')
```

In [73]:
```
y_pred_svm = svm_classifier.predict(X_test)
```

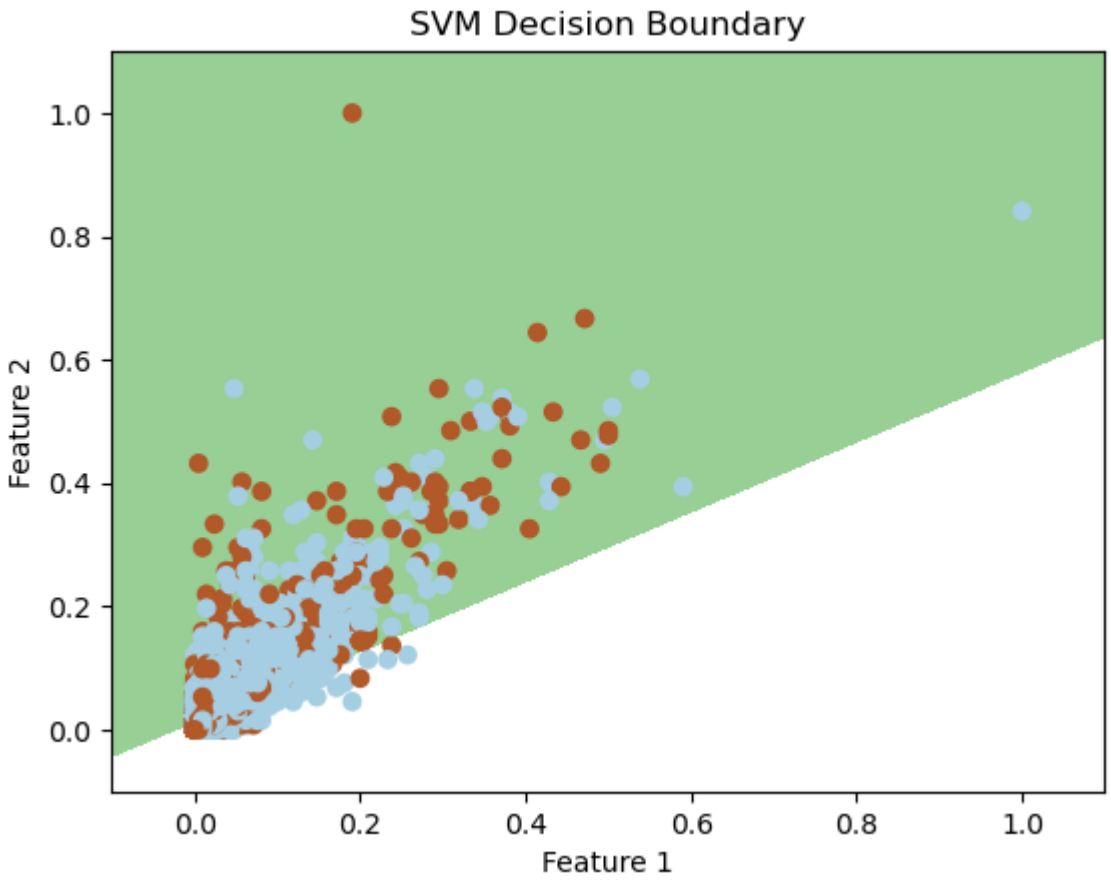
In [74]:
```
confusion_matrix_svm = confusion_matrix(Y_test, y_pred_svm)
print("Confusion Matrix:")
print(confusion_matrix_svm)
```

```
Confusion Matrix:
[[929   0]
 [364   0]]
```

In [75]:
```
x_min, x_max = X_selected_features[:, 0].min() - 0.1, X_selected_features[:
y_min, y_max = X_selected_features[:, 1].min() - 0.1, X_selected_features[:
xx, yy = np.meshgrid(np.linspace(x_min, x_max, 500), np.linspace(y_min, y_m
```

In [76]:
```
Z = svm_classifier.decision_function(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
```

In [79]:
```
plt.contourf(xx, yy, Z, levels=[-1, 0, 1], alpha=0.5, cmap=plt.cm.Paired)
plt.scatter(X_selected_features[:, 0], X_selected_features[:, 1], c=y, cmap
plt.xlabel(f'Feature {feature1_index + 1}')
plt.ylabel(f'Feature {feature2_index + 1}')
plt.title('SVM Decision Boundary')
plt.show()
```



In [78]:
```
accuracy_svm = accuracy_score(Y_test, y_pred_svm)
print(f"Accuracy Score: {accuracy_svm:.2f}")

classification_rep_svm = classification_report(Y_test, y_pred_svm)
print("Classification Report:")
print(classification_rep_svm)
```

```
Accuracy Score: 0.72
Classification Report:
              precision    recall  f1-score   support

           0       0.72      1.00      0.84       929
           1       0.00      0.00      0.00       364

    accuracy                           0.72      1293
   macro avg       0.36      0.50      0.42      1293
weighted avg       0.52      0.72      0.60      1293


C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\_classificatio
n.py:1318: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\_classificatio
n.py:1318: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\_classificatio
n.py:1318: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```