

#Importing libraries and modules

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF
from sklearn.naive_bayes import GaussianNB
from sklearn import svm
```

```
from sklearn.metrics import mean_squared_error
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn.metrics import roc_auc_score
from sklearn.metrics import RocCurveDisplay
```

```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import LabelBinarizer
from sklearn.metrics import balanced_accuracy_score
from sklearn.metrics import multilabel_confusion_matrix
```

#Read the desired database from an Excel spreadsheet

#The file should contain two tabs labeled Ref and Exp

#Ref contains the reference data, for instance:

#

		TiO2	Al2O3	CaO	MgO	MnO	K2O	Na2O	P2O5	
Group	sample	/SiO2	/SiO2	/SiO2	/SiO2	/SiO2	/SiO2	/SiO2	/SiO2	
	1	1	0.043	0.300	0.215	0.203	0.002	0.032	0.058	0.010
	1	2

#Exp contains the archaeological sample analyses, for instance

		TiO2	Al2O3	CaO	MgO	MnO	K2O	Na2O	P2O5
Group	sample	/SiO2	/SiO2	/SiO2	/SiO2	/SiO2	/SiO2	/SiO2	/SiO2
	Z1a	0.055	0.349	0.249	0.117	0.003	0.021	0.073	0.010
	Z1b

```
df_data = pd.read_excel("NAME.xlsx", sheet_name="Ref")
```

```
df_data_X = pd.read_excel("NAME.xlsx", sheet_name="Exp")
```

#Depending on the number of variables select or adapt the following:

#Option 1, for as many variables as in Database90

```
X_clas = df_data[["TiO2", "Al2O3", "CaO", "MgO", "MnO", "K2O", "Na2O", "P2O5"]]
```

```
y_clas = df_data["group"]
```

```
X_clasX = df_data_X[["TiO2", "Al2O3", "CaO", "MgO", "MnO", "K2O", "Na2O", "P2O5"]]
```

#Option 2, for as many variables as in Database60

```
X_clas = df_data[["TiO2", "Al2O3", "Cr", "Fe2O3", "CaO", "MgO", "MnO", "K2O", "Na2O", "P2O5", "V", "Ni", "Rb", "Sr", "Y", "Zr", "Nb"]]
```

```
y_clas = df_data["group"]
```

```
X_clasX = df_data_X[["TiO2", "Al2O3", "Cr", "Fe2O3", "CaO", "MgO", "MnO", "K2O", "Na2O", "P2O5", "V", "Ni", "Rb", "Sr", "Y", "Zr", "Nb"]]
```

#Option 3, for as many variables as in Database30

```
X_clas = df_data[["TiO2", "Al2O3", "Cr", "Fe2O3", "CaO", "MgO", "MnO", "K2O", "Na2O", "P2O5", "V", "Ni", "Cu", "Zn", "Rb", "Sr", "Y", "Zr",  
"Nb"]]
```

```
y_clas = df_data["group"]
```

```
X_clasX = df_data_X[["TiO2", "Al2O3", "Cr", "Fe2O3", "CaO", "MgO", "MnO", "K2O", "Na2O", "P2O5", "V", "Ni", "Cu", "Zn", "Rb", "Sr", "Y", "Zr",  
"Nb"]]
```

#Create a train-test split

```
X_clas_train, X_clas_test, y_clasd_train, y_clasd_test = train_test_split(X_clas, y_clasd)
```

#Select the desired model:

#Option 1, GB

```
model = GradientBoostingClassifier(n_estimators=80).fit(X_clas_train, y_clasd_train)
```

#Option 2, GPC

```
kernel = 1.0 * RBF(1.0)
```

```
model = GaussianProcessClassifier(kernel=kernel,random_state=0).fit(X_clas_train, y_clasd_train)
```

```
model.score(X_clas_train, y_clasd_train)
```

#Option 3, GNB

```
model = GaussianNB().fit(X_clas_train, y_clasd_train)
```

#Option 4, LSVM

```
model = svm.SVC(kernel="linear").fit(X_clas_train, y_clasd_train)
```

#Testing the trained model

```
preds_clas_test = model.predict(X_clas_test)
```

```
accuracy_score(y_clasd_train, preds_clas_train)
```

```
cm = confusion_matrix(y_clasd_test, preds_clas_test)
```

```
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
```

```
disp.plot()
```

#Predicting the class for archaeological data

```
preds_clas = model.predict(X_clasX)
```

```
le.inverse_transform(preds_clas)
```