Libraries Syntax Help File

Pandas

```
# Importing Pandas
import pandas as pd

# Reading a CSV file
df = pd.read_csv('path_to/your_file.csv')

# Displaying the first 5 rows
print(df.head())

# Grouping data
grouped = df.groupby('column_name')
grouped_sum = grouped.sum()

# Handling missing data by filling with mean
df_filled = df.apply(lambda col: col.fillna(col.mean()), axis=0)
```

Seaborn

```
# Importing Seaborn
import seaborn as sns

# Creating a bar plot
sns.barplot(x='column_x', y='column_y', data=df)
plt.title('Title')
plt.show()
```

Matplotlib

```
# Importing Matplotlib
import matplotlib.pyplot as plt

# Plotting a scatter plot
plt.scatter(x, y, c=labels, cmap='viridis')
```

```
plt.title('Title')
plt.xlabel('X-axis label')
plt.ylabel('Y-axis label')
plt.show()

# Plotting a confusion matrix
from sklearn.metrics import ConfusionMatrixDisplay
cm = confusion_matrix(y_true, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot(cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.show()
```

Scikit-learn

```
# Importing Scikit-learn datasets
from sklearn.datasets import load_iris, load_digits, load_wine, load_breast_cancer

# Splitting data into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Training a Support Vector Machine (SVM) classifier
from sklearn.svm import SVC
svm_clf = SVC()
svm_clf.fit(X_train, y_train)
y_pred = svm_clf.predict(X_test)

# Evaluating model accuracy
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

# Training a Gradient Boosting Classifier
from sklearn.ensemble import GradientBoostingClassifier
gb_clf = GradientBoostingClassifier(n_estimators=100, random_state=42)
gb_clf.fit(X_train, y_train)

# Performing cross-validation
from sklearn.model_selection import cross_val_score
scores = cross_val_score(gb_clf, X, y, cv=5)
```

```
print(f"CV Accuracy: {scores.mean() * 100:.2f}%")

# K-means Clustering
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(X)
labels = kmeans.labels_

# DBSCAN Clustering
from sklearn.cluster import DBSCAN
dbscan = DBSCAN(eps=0.5, min_samples=5)
dbscan.fit(X)
labels = dbscan.labels_
```