```
import numpy as np
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.preprocessing import
StandardScaler

class DynamicObjectiveSpecification:
    def __init__(self, data,
user_preferences):
        self.data = data
        self.user_preferences =
user_preferences
        self.objective_space = None

    def preprocess_data(self):
        # Preprocess raw data (e.g., scaling,
normalization)
        scaler = StandardScaler()
        scaled_data =
scaler.fit_transform(self.data)
        self.data =
pd.DataFrame(scaled_data,
```

```
columns=self.data.columns)

    def generate_objective_space(self):
        # Generate initial objective space
based on data characteristics
        kmeans = KMeans(n_clusters=3)
        kmeans.fit(self.data)
        self.objective_space =
kmeans.cluster_centers_

    def adapt_objective_space(self,
contextual_information):
        # Adapt objective space based on
contextual information (e.g., user
feedback)
        # Update objective space using
reinforcement learning, evolutionary
algorithms, or other techniques
        pass

    def select_objectives(self):
        # Select objectives based on user
preferences and task requirements
```

```
        selected_objectives = []
        for preference in
self.user_preferences:
            # Find objectives closest to user
preferences in the objective space
            objective_idx =
np.argmin(np.linalg.norm(self.objective_
space - preference, axis=1))

selected_objectives.append(self.objective
_space[objective_idx])
        return selected_objectives

    def run(self, contextual_information):
        self.preprocess_data()
        self.generate_objective_space()

self.adapt_objective_space(contextual_i
nformation)
        selected_objectives =
self.select_objectives()
        return selected_objectives
```

```
# Example usage:
data = pd.read_csv("data.csv")  # Load raw data
user_preferences = np.array([[1, 0, 0], [0, 1, 0], [0, 0, 1]])  # User preferences for objectives
contextual_information = {}  # Additional contextual information (e.g., user feedback)
dos = DynamicObjectiveSpecification(data, user_preferences)
selected_objectives = dos.run(contextual_information)
print("Selected Objectives:", selected_objectives)
```