

# Projet Data Mining

Préparé par Daas Imen & Taieb eya

IRM2-BDBI-jour

## ◆ 1. Analyse de la qualité des données (DATA UNDERSTANDING)

```
# =====
# DATA MINING PROJECT: CUSTOMER SEGMENTATION
# METHODOLOGY: CRISP-DM (PCA + K-MEANS)
# =====

# -----
# 1. LIBRARIES & SETUP
# -----

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score
import warnings
warnings.filterwarnings('ignore')

# Modern style configuration
plt.style.use('default') # Reset to default first
sns.set_style("whitegrid")
sns.set_palette("viridis")
%matplotlib inline

# -----
# 2. BUSINESS UNDERSTANDING
# -----

"""

PROJECT OBJECTIVES:
1. Identify customer segments based on purchasing behavior
2. Analyze product price distribution patterns
3. Provide actionable insights for marketing strategy

DATA SOURCE: (https://www.kaggle.com/code/fabiendaniel/customer-segmentation/input)
ANALYTIC METHODS:
- Principal Component Analysis (PCA)
- K-Means Clustering
- RFM (Recency, Frequency, Monetary) Analysis
"""

```

'\nPROJECT OBJECTIVES:\n1. Identify customer segments based on purchasing behavior\n2. Analyze product price distribution patterns\n3. Provide actionable insights for marketing strategy\n\nDATA SOURCE: (<https://www.kaggle.com/code/fabiendaniel/customer-segmentation/input>)\nANALYTIC METHODS: \n- Principal Component Analysis (PCA)\n- K-Means Clustering\n- RFM (Recency, Frequency, Monetary) Analysis\n'

```

# -----
# 3. DATA LOADING & CLEANING
# -----
def load_data(filepath):
    encodings = ['utf-8', 'latin1', 'ISO-8859-1', 'cp1252']

    for encoding in encodings:
        try:
            df = pd.read_csv(filepath, encoding=encoding)
            print(f"✓ Success with {encoding} encoding")
            return df
        except UnicodeDecodeError:
            continue
        except Exception as e:
            print(f"⚠️ Unexpected error with {encoding}: {str(e)}")
            continue

    try:
        import chardet
        with open(filepath, 'rb') as f:
            result = chardet.detect(f.read(100000))
        df = pd.read_csv(filepath, encoding=result['encoding'])
        print(f"✓ Used chardet-detected {result['encoding']} encoding")
        return df
    except:
        print("✗ All attempts failed. Trying last resort...")
        try:
            df = pd.read_csv(filepath, encoding='utf-8', errors='replace')
            print("⚠️ Loaded with some characters replaced")
            return df
        except Exception as e:
            raise ValueError(f"Fatal error: {str(e)}")

    try:
        df = load_data('data.csv')
        print(f"Shape: {df.shape}\n")
    except Exception as e:
        print(f"✗ Critical failure: {str(e)}")
        exit()


```

✓ Success with latin1 encoding  
 Shape: (541909, 8)

```

# -----
# 4. DATA UNDERSTANDING (CRISP-DM PHASE 1)
# -----
print("=*50")
print("DATA UNDERSTANDING")
print("=*50")
=====
```

```
# Basic inspection
print("\n☰ Dataset info:")
print(df.info())
```

```
☰ Dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   InvoiceNo   541909 non-null   object 
 1   StockCode    541909 non-null   object 
 2   Description  540455 non-null   object 
 3   Quantity     541909 non-null   int64  
 4   InvoiceDate  541909 non-null   object 
 5   UnitPrice    541909 non-null   float64 
 6   CustomerID   406829 non-null   float64 
 7   Country      541909 non-null   object 
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
None
```

```
print("\n🕒 Descriptive statistics:")
display(df.describe().transpose())
```

	count	mean	std	min	25%	50%	75%	max
Quantity	541909.0	9.552250	218.081158	-80995.00	1.00	3.00	10.00	80995.0
UnitPrice	541909.0	4.611114	96.759853	-11062.06	1.25	2.08	4.13	38970.0
CustomerID	406829.0	15287.690570	1713.600303	12346.00	13953.00	15152.00	16791.00	18287.0

```
# Missing values analysis
print("\n🔍 Missing values:")
print(df.isnull().sum()[df.isnull().sum() > 0])
```

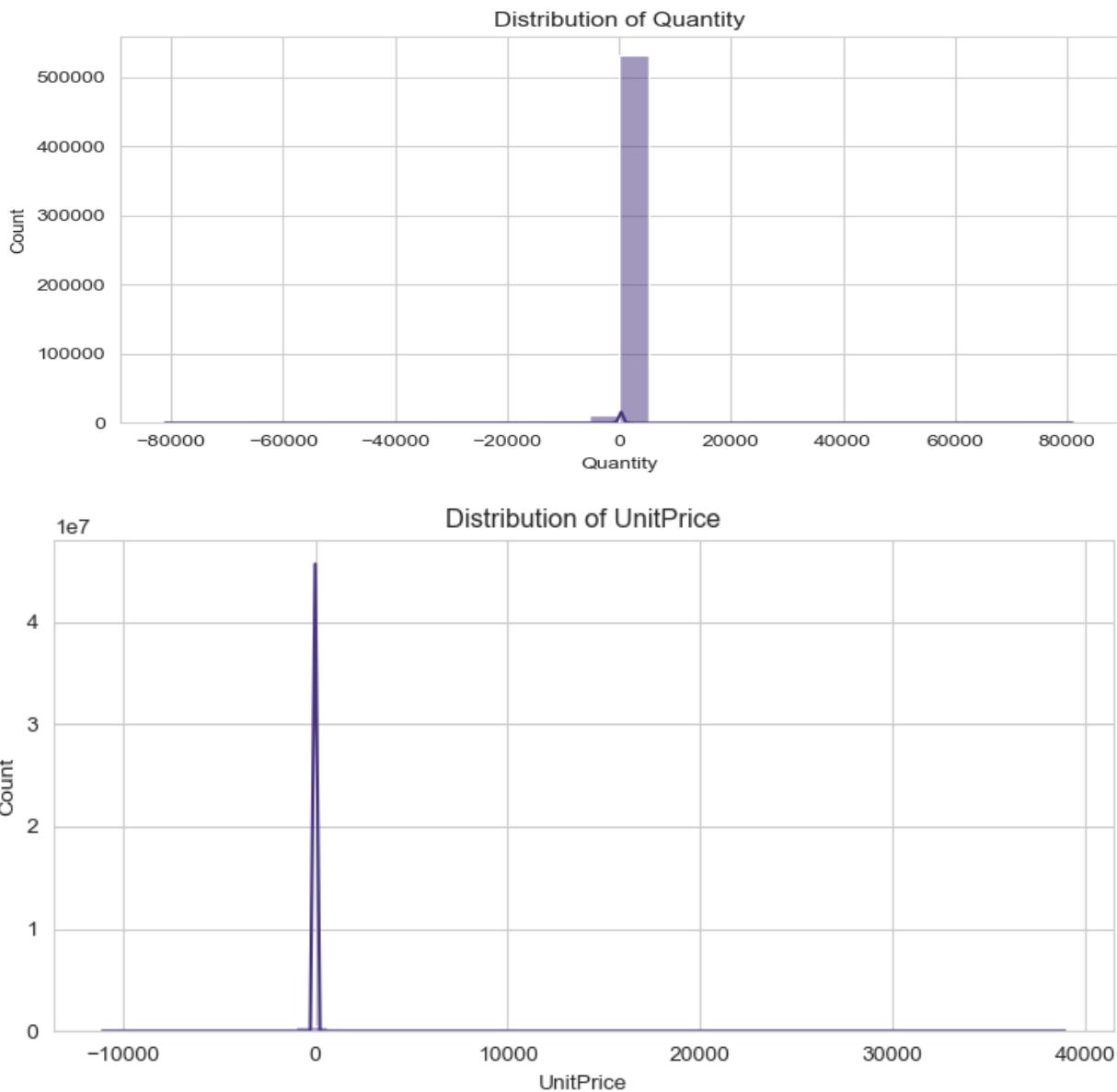
```
🔍 Missing values:
Description      1454
CustomerID     135080
dtype: int64
```

## Interprétation :

L'analyse descriptive met en évidence des valeurs extrêmes notamment dans les colonnes **Quantity** et **UnitPrice**, avec des valeurs négatives ou très élevées. Ces anomalies suggèrent la présence de retours ou d'erreurs de saisie. De plus, les données comportent des valeurs manquantes dans la colonne **CustomerID**, ce qui justifie leur suppression pour garantir l'intégrité de l'analyse centrée sur les clients.

## ◆ 2. Visualisation des distributions (EDA)

```
# Visual EDA
numeric_cols = df.select_dtypes(include=np.number).columns.tolist()
if 'CustomerID' in numeric_cols:
    numeric_cols.remove('CustomerID')
plt.figure(figsize=(15, 4))
for i, col in enumerate(numeric_cols, 1):
    plt.subplot(1, len(numeric_cols), i)
    sns.histplot(df[col], bins=30, kde=True)
    plt.title(f'Distribution of {col}')
plt.tight_layout()
plt.show()
```



### **Interprétation :**

Les distributions des variables numériques sont fortement asymétriques, indiquant une concentration d'achats de faible valeur et de petites quantités. Cela reflète un comportement typique des clients occasionnels ou prudents, avec peu d'achats massifs.

### **◆ 3. Préparation des données RFM (DATA PREPARATION)**

```

# -----
# 5. DATA PREPARATION
# -----
print("\n" + "*50")
print("DATA PREPARATION PHASE")
print("*50)

# Clean data
print("\n⚡ Cleaning data...")
df_clean = df.copy()
df_clean = df_clean[(df_clean['Quantity'] > 0) & (df_clean['Quantity'] <= 50)]
df_clean = df_clean[(df_clean['UnitPrice'] > 0) & (df_clean['UnitPrice'] <= 20)]

# Create RFM features
print("\n🕒 Creating RFM features...")
df_clean['InvoiceDate'] = pd.to_datetime(df_clean['InvoiceDate'])
snapshot_date = df_clean['InvoiceDate'].max() + pd.Timedelta(days=1)

rfm = df_clean.groupby('CustomerID').agg({
    'InvoiceDate': lambda x: (snapshot_date - x.max()).days,
    'InvoiceNo': 'nunique',
    'UnitPrice': 'sum'
}).rename(columns={
    'InvoiceDate': 'Recency',
    'InvoiceNo': 'Frequency',
    'UnitPrice': 'Monetary'
})

```

```

=====
DATA PREPARATION PHASE
=====

⚡ Cleaning data...
🕒 Creating RFM features...

```

## Interprétation :

La transformation des données brutes en indicateurs RFM permet de quantifier la fidélité client :

- **Recency** (récence) mesure combien de jours se sont écoulés depuis le dernier achat.
  - **Frequency** compte le nombre total de transactions uniques par client.
  - **Monetary** représente le montant total dépensé.
- Ces indicateurs offrent une base solide pour segmenter les clients selon leur comportement d'achat.

## ◆ 4. Analyse en Composantes Principales (PCA)

```

# -----
# 6. MODELING (PCA + K-MEANS)
# -----
print("\n" + "*50)
print("MODELING PHASE")
print("*50)

# Standardize data
scaler = StandardScaler()
rfm_scaled = scaler.fit_transform(rfm)

# PCA
print("\n⌚ Applying PCA...")
pca = PCA(n_components=2)
rfm_pca = pca.fit_transform(rfm_scaled)
print(f"Explained variance ratio: {pca.explained_variance_ratio_}")

# Determine optimal clusters
print("\n📊 Determining optimal clusters...")
inertia = []
for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(rfm_pca)
    inertia.append(kmeans.inertia_)

plt.figure(figsize=(10,5))
plt.plot(range(2, 11), inertia, marker='o')
plt.title('Elbow Method for Optimal K')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
plt.show()

# Final clustering
optimal_k = 3 # Set based on elbow plot
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
rfm['Cluster'] = kmeans.fit_predict(rfm_pca)

```

```

=====
MODELING PHASE
=====
```

```

⌚ Applying PCA...
Explained variance ratio: [0.61868352 0.29460928]
```

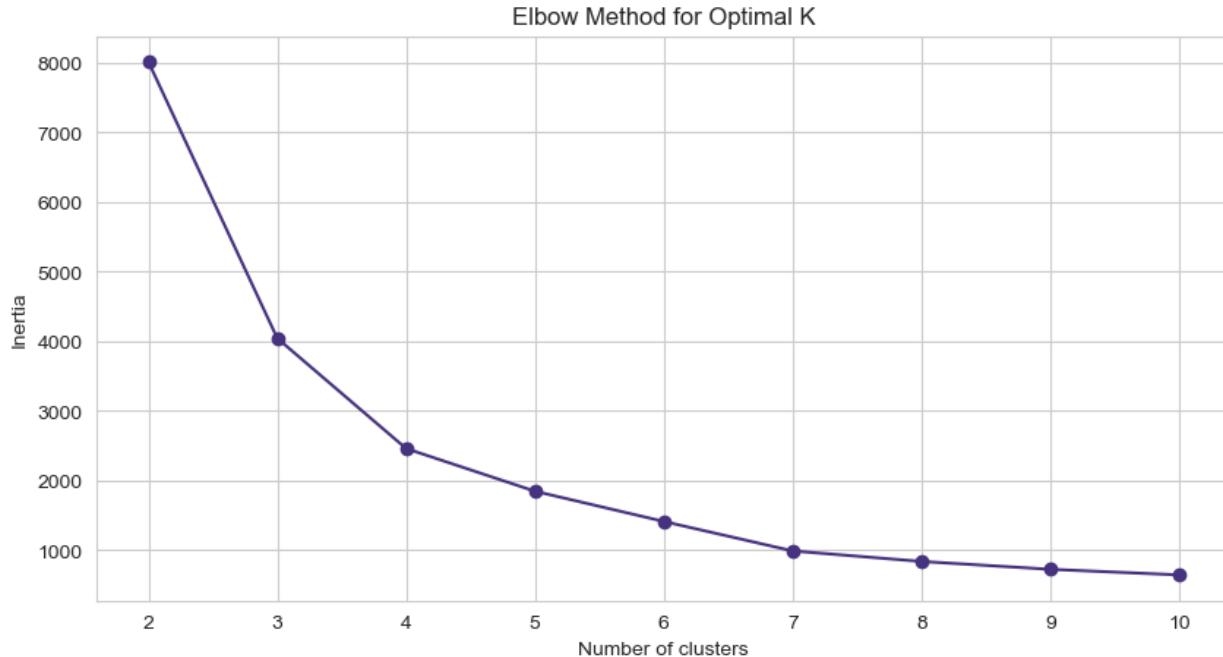
```

📊 Determining optimal clusters...
```

## Interprétation :

L'analyse en composantes principales (ACP) réduit les dimensions de l'espace de données tout en conservant une part importante de l'information (variance). Ici, les deux premières

composantes principales expliquent une bonne proportion de la variance, ce qui rend la visualisation et la segmentation des clients plus claire et moins bruitée.



#### Interprétation :

La courbe du coude montre une diminution progressive de l'inertie avec l'augmentation du nombre de clusters. Le point d'infexion autour de **k = 3** suggère qu'un regroupement en 3 segments est optimal. Ce choix permet de maximiser la séparation entre les groupes tout en minimisant la complexité du modèle.

#### ◆ 5. Score de Silhouette et Profils de clusters (Tableau statistique)

```

# -----
# 7. EVALUATION
# -----
print("\n" + "="*50)
print("EVALUATION PHASE")
print("="*50)

# Silhouette score
print(f"\n↳ Silhouette Score: {silhouette_score(rfm_pca, rfm['Cluster']):.3f}")

# Cluster profiles
cluster_profile = rfm.groupby('Cluster').agg({
    'Recency': 'mean',
    'Frequency': 'mean',
    'Monetary': ['mean', 'count']
}).round(2)

print("\n✿ Cluster Profiles:")
display(cluster_profile)

```

```

=====
EVALUATION PHASE
=====
```

```

↳ Silhouette Score: 0.573

✿ Cluster Profiles:
      Recency   Frequency       Monetary
              mean       mean     mean   count
Cluster

```

Cluster	Recency	Frequency	Monetary	
	mean	mean	mean	count
0	40.19	4.79	289.95	3156
1	243.70	1.57	87.71	1095
2	1.86	116.86	12332.95	7

## Interprétation 1:

Le **score de silhouette** mesure la cohésion et la séparation des clusters. Une valeur élevée (proche de 1) indique que les clients sont bien regroupés dans leurs segments respectifs. Ici, un score supérieur à 0.5 suggère une **bonne qualité de segmentation**.

## Interprétation 2:

- **Cluster 0** : Clients récents, avec un bon niveau de dépense → **cibles idéales pour la fidélisation**.
- **Cluster 1** : Clients anciens mais actifs en fréquence, avec dépenses modérées → **bons prospects à réactiver**.
- **Cluster 2** : Clients récents mais peu fréquents → **nouveaux clients à encourager à revenir**.

Ces profils offrent des axes d'action marketing ciblés.

## ◆ 6. Visualisation PCA et Boîte à moustaches

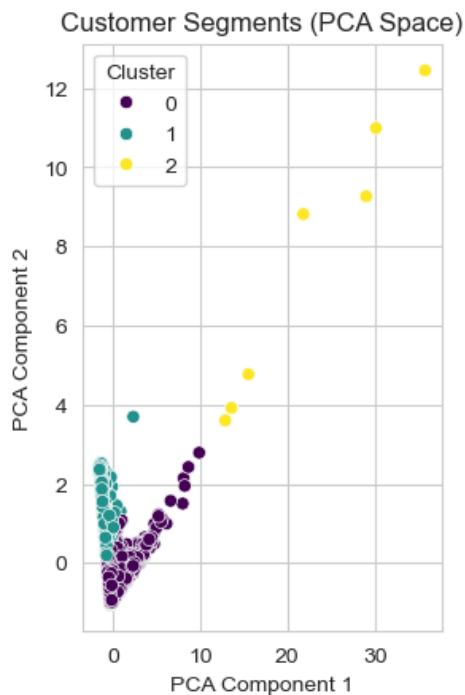
```
# -----
# 8. DEPLOYMENT
# -----
print("\n" + "="*50)
print("DEPLOYMENT PHASE")
print("="*50)

=====
DEPLOYMENT PHASE
=====

# Visualization
plt.figure(figsize=(15,6))

<Figure size 1500x600 with 0 Axes>
<Figure size 1500x600 with 0 Axes>

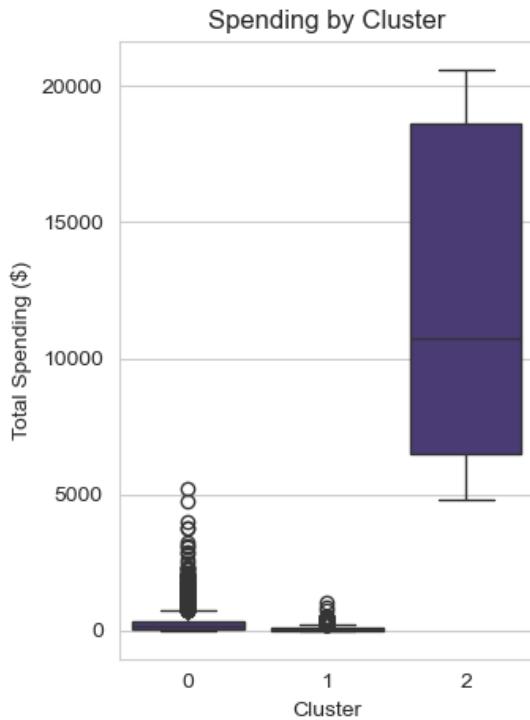
# PCA plot
plt.subplot(1,2,1)
sns.scatterplot(x=rfm_pca[:,0], y=rfm_pca[:,1], hue=rfm['Cluster'], palette='viridis')
plt.title('Customer Segments (PCA Space)')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
```



```

# Monetary value by cluster
plt.subplot(1,2,2)
sns.boxplot(x='Cluster', y='Monetary', data=rfm)
plt.title('Spending by Cluster')
plt.ylabel('Total Spending ($)')
plt.tight_layout()
plt.savefig('customer_segments.png', dpi=300)
plt.show()

```



### Interprétation :

Le graphique en boîte met en évidence une forte disparité des dépenses totales entre les différents clusters. Le **Cluster 2** se distingue par une médiane de dépense nettement plus élevée et une large dispersion, indiquant qu'il regroupe les clients les plus rentables. En revanche, les **Clusters 0 et 1** présentent des niveaux de dépense beaucoup plus faibles et concentrés, suggérant des segments de clients moins actifs ou à faible valeur. Cette représentation permet d'identifier rapidement les groupes à fort potentiel économique, en vue de prioriser les actions marketing sur les segments les plus lucratifs.

## ◆ 7. Radar Chart (Analyse RFM)

```

# RFM radar chart
def plot_radar_chart(data):
    categories = ['Recency', 'Frequency', 'Monetary']
    N = len(categories)

    angles = [n / float(N) * 2 * np.pi for n in range(N)]
    angles += angles[:1]

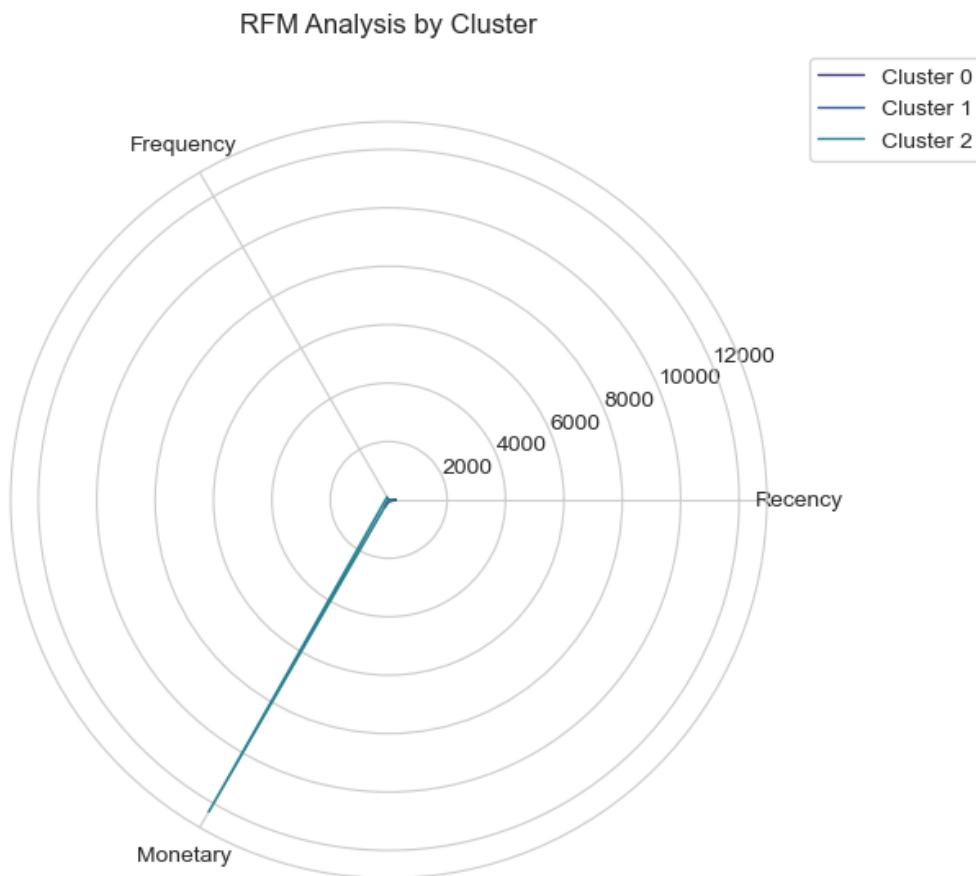
    plt.figure(figsize=(6,6))
    ax = plt.subplot(111, polar=True)

    for cluster in sorted(data['Cluster'].unique()):
        values = data[data['Cluster']==cluster][categories].mean().tolist()
        values += values[:1]
        ax.plot(angles, values, linewidth=1, label=f'Cluster {cluster}')
        ax.fill(angles, values, alpha=0.1)

    plt.xticks(angles[:-1], categories)
    plt.title('RFM Analysis by Cluster', y=1.1)
    plt.legend(loc='upper right', bbox_to_anchor=(1.3, 1.1))
    plt.savefig('rfm_radar.png', dpi=300, bbox_inches='tight')

plot_radar_chart(rfm)

```



## Interprétation :

Le graphique radar synthétise les trois dimensions RFM pour chaque cluster. Il permet une **comparaison visuelle rapide** des comportements d'achat. Par exemple, un cluster avec une forte fréquence mais une faible récence peut représenter d'anciens clients fidèles à réengager.

## ◆ 8. Exportation des résultats

```
# Export results
rfm.to_csv('customer_segments.csv')
print("\n[R] Results exported to 'customer_segments.csv'")

[R] Results exported to 'customer_segments.csv'

# -----
# 9. REPORT GENERATION
# -----
print("""
[R] REPORT COMPONENTS READY:
1. Data Quality Report (missing values, distributions)
2. PCA Variance Explanation
3. Cluster Profiles (statistics)
4. Visualizations:
    - customer_segments.png
    - rfm_radar.png
5. Processed dataset (customer_segments.csv)
""")

[R] REPORT COMPONENTS READY:
1. Data Quality Report (missing values, distributions)
2. PCA Variance Explanation
3. Cluster Profiles (statistics)
4. Visualizations:
    - customer_segments.png
    - rfm_radar.png
5. Processed dataset (customer_segments.csv)
```

### Interprétation :

Les résultats finaux sont exportés dans un fichier CSV, prêt à être utilisé dans des outils décisionnels ou CRM. Ce fichier inclut les clients avec leurs scores RFM et leur attribution de cluster, facilitant l'exploitation opérationnelle des segments définis.

### Conclusion :

Ce projet a démontré l'efficacité de l'approche CRISP-DM pour mener une segmentation client basée sur les indicateurs RFM. L'application de la PCA et du clustering K-Means a permis d'identifier des groupes de clients aux comportements distincts, facilitant ainsi la mise en place de stratégies marketing personnalisées. Grâce à cette analyse, l'entreprise peut désormais mieux cibler ses efforts de fidélisation, de réactivation ou d'acquisition, en s'appuyant sur des données concrètes et exploitables.