Day 11 – K-Means Clustering (Unsupervised Learning)

📓 Today's Highlights

In this session, we explored **K-Means Clustering**, a core concept in **unsupervised learning**. We applied it to a real-world social media dataset to identify patterns in post engagement and behavior.

□ What is K-Means Clustering?

- K-Means is an **unsupervised machine learning algorithm** used for **clustering** unlabeled data into *k* distinct groups.
- The algorithm partitions data based on feature similarity by:
    1. Choosing *k* initial centroids
    2. Assigning data points to the nearest centroid
    3. Recomputing centroids based on cluster assignments
    4. Repeating until convergence

📁 Dataset Used: `Live.csv`

The dataset contained user engagement metrics from a social media platform, including:

- Likes
- Shares
- Comments
- Type of post (`status_type`)
- Total reactions

Initial cleanup involved:

- Dropping unnecessary columns: `Column1`, `Column2`, `Column3`, `Column4`, `status_id`, `status_published`
- Encoding the categorical `status_type` column using **Label Encoding**

🔍 Preprocessing Steps:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, MinMaxScaler

# Load data
df = pd.read_csv("Live.csv")

# Drop unused columns
```

```
df.drop(['Column1', 'Column2', 'Column3', 'Column4', 'status_id',
'status_published'], axis=1, inplace=True)

# Label encode the target column
le = LabelEncoder()
df['status_type'] = le.fit_transform(df['status_type'])

# Feature scaling
scaler = MinMaxScaler()
X = scaler.fit_transform(df)
```

📊 Applying K-Means Clustering

```
from sklearn.cluster import KMeans

# Train KMeans model
kmeans = KMeans(n_clusters=2, random_state=0)
kmeans.fit(X)

# Predictions
labels = kmeans.labels_

# Evaluate clustering accuracy
correct_labels = sum(le.transform(df['status_type']) == labels)
accuracy = correct_labels / float(len(df))
print(f"Accuracy score: {accuracy:.2f}")
```

🔄 Elbow Method to Find Optimal K

```
cs = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, random_state=0)
    kmeans.fit(X)
    cs.append(kmeans.inertia_)

# Plot elbow graph
plt.plot(range(1, 11), cs)
plt.xlabel("Number of Clusters")
plt.ylabel("Inertia")
plt.title("Elbow Method for Optimal K")
plt.show()
```

✅ Output & Observations

```
Result: 604 out of 1000 samples were correctly labeled.
Accuracy score: 0.60
```

- The elbow plot helps determine the ideal number of clusters (`k`).
- Even though accuracy is not the perfect evaluation metric for unsupervised learning, it gives insight when true labels are known.
- K-Means was effective in dividing posts into behavior-based clusters.

📌 Conclusion

- Understood and implemented **K-Means clustering** on scaled numeric data.
- Preprocessed and visualized data using pandas, seaborn, and matplotlib.
- Learned how clustering differs from classification — no labels are provided, patterns are identified based on distances.
- Used the **elbow method** to determine the right number of clusters.