Model Training Documentation

In this project, we trained a K-Means clustering model to segment customers based on their features.

What is K-Means Clustering?

K-Means clustering is an unsupervised machine learning algorithm used to partition a dataset into K distinct, non-overlapping subsets (clusters). The algorithm aims to minimize the within-cluster sum of squares (inertia), which measures the variance within each cluster.

Steps Involved in K-Means Clustering:

1. Initialization: Select K initial centroids randomly from the dataset.

2. Assignment: Assign each data point to the nearest centroid, forming K clusters.

3. Update: Recalculate the centroids of the clusters by taking the mean of all data points in each cluster.

4. Repeat: Repeat the assignment and update steps until the centroids no longer change or the maximum number of iterations is reached.

The goal is to minimize the distance between data points within the same cluster while maximizing the distance between clusters.

Model Training Steps:

1. Data Loading: The preprocessed dataset was loaded.

2. Feature Selection: Features for clustering were selected by excluding the target variable 'Churn_Yes'.

3. Feature Scaling: The features were scaled using the StandardScaler to standardize the data.

4. Model Training: The K-Means algorithm was used to train the model with the optimal number of clusters (3), as determined by the Elbow method.

5. Model Saving: The trained K-Means model was saved using joblib.

6. Results Saving: The clustering results were saved to a CSV file. The trained model can be used to predict cluster labels for new data points, helping in understanding customer segments and devising targeted strategies.

Code Snippet for Training the Model:

```
import joblib

import pandas as pd

from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

# Load the dataset
url = 'https://drive.google.com/uc?id=1qBkAiPPQ9bTiaY6PcDq7Tmp8DQsCyyrZ'
df = pd.read_csv(url)

# Convert categorical variables to numeric
df_encoded = pd.get_dummies(df)

target_column = 'Churn_No'

# Separate features and target variable
X = df_encoded.drop([target_column, 'Churn_Yes'], axis=1)  # Drop the target variables

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train the clustering model (using 3 clusters as an example)
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(X_scaled)

# Save the trained model
joblib.dump(kmeans, 'kmeans_model.pkl')

print("Clustering model trained and saved as 'kmeans_model.pkl'")
```