Changes	Time	Difficulty
Use elbow method to find the best number of cluster and get 3 for dataset to find if the customer has risk_flag or not	40 mins	4
Train the model with only 4 features using cluster of 3 and the cluster are overlapping each other too much	20 mins	2
Retrain the model using 10 features and get a lower silhoutte score of 0.15 compare to previous of 0.18	30 mins	3
Encode categorical faetures and use all 13 features to train the model , but does not work	1 hour	6
Lower iteration value to run faster and no changes to cluster	10mins	1
Change algorithm to use Elkan since the dataset is big but the result stay similar	10 mins	1
Change dataset about cancer cell but does not work since dataset only contain yes and no	1 hour	6
Train the model using wine data with default settings but use elbow method to get the cluster number which is 4	45 mins	5

In [1]:
```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import StandardScaler
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import OneHotEncoder
```

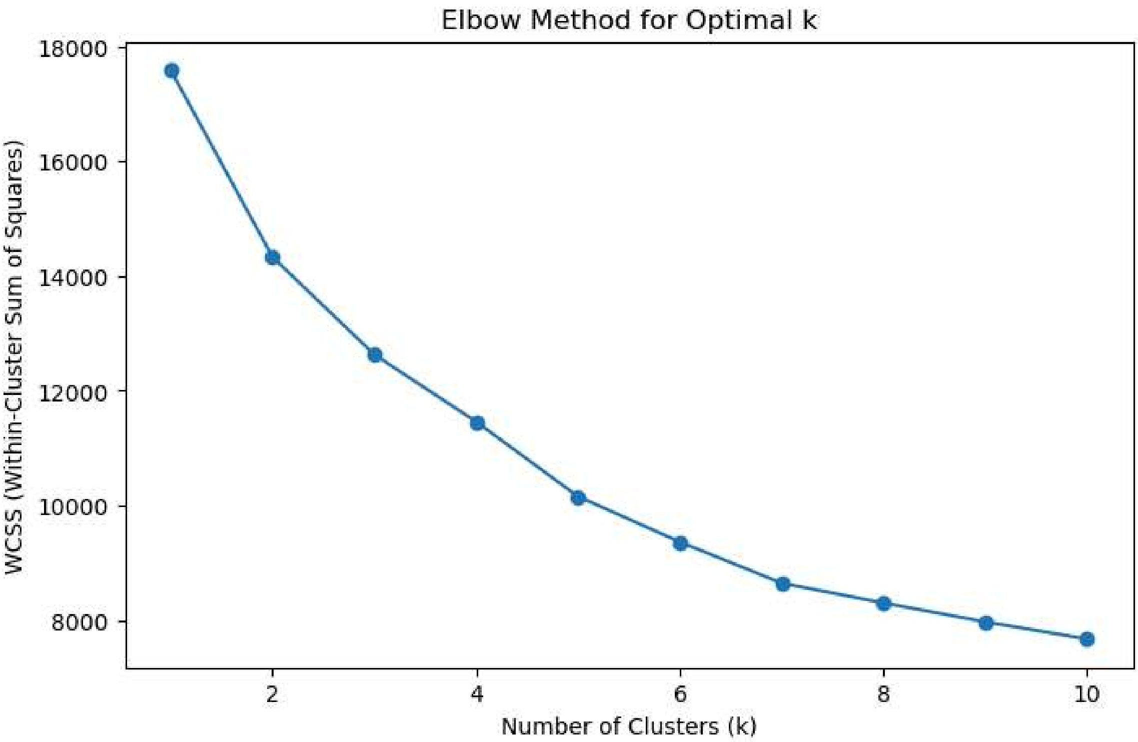
In [2]:
```
import os
os.environ["OMP_NUM_THREADS"] = "7"
```

In [3]:
```
file_path = 'wine_data.csv'
df = pd.read_csv(file_path)
```

In [4]:
```
scaler = StandardScaler()
X = df.iloc[:, :-1]
X_train_scaled = scaler.fit_transform(X)
```

In [5]:
```
wcss = []

for k in range(1, 11) :
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
    kmeans.fit(X_train_scaled)
    wcss.append(kmeans.inertia_)

plt.figure(figsize=(8, 5))
plt.plot(range(1, 11), wcss, marker='o', linestyle='-')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('WCSS (Within-Cluster Sum of Squares)')
plt.title('Elbow Method for Optimal k')
plt.show()
```

```
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMe
ans is known to have a memory leak on Windows with MKL, when there are less chunks than available thread
s. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
```

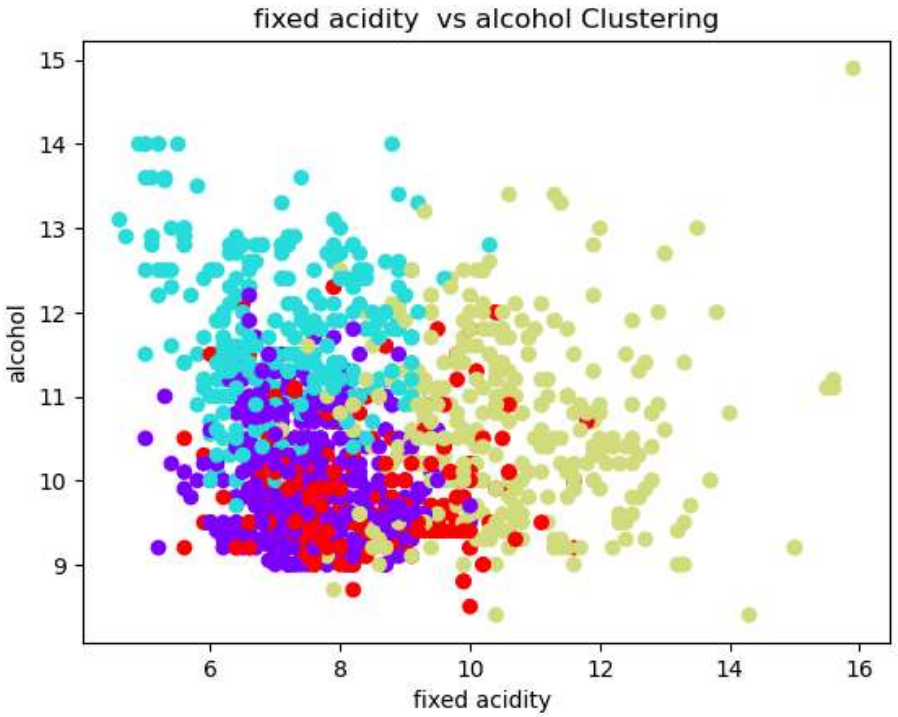


In [6]:
```
kmeans = KMeans(n_clusters=4, random_state=42, n_init=10)
kmeans.fit(X_train_scaled)

clusters = kmeans.labels_
```

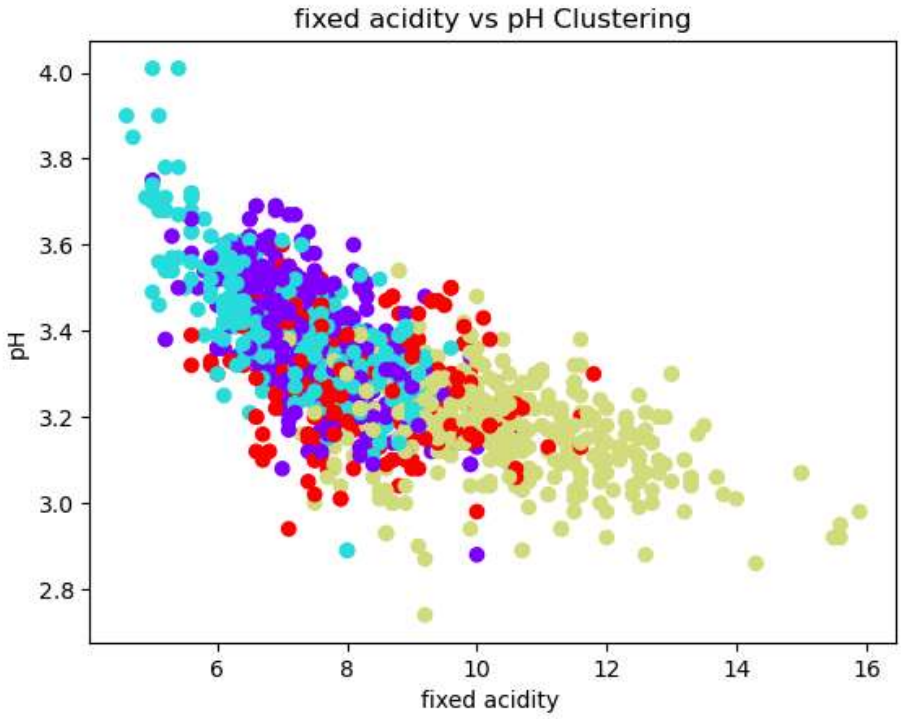
```
silhouette = silhouette_score(X_train_scaled, clusters)
print(f'Silhouette Score: {silhouette:.4f}')
```

```
C:\Users\GGMachines_Gaming\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
  warnings.warn(
Silhouette Score: 0.1716
```

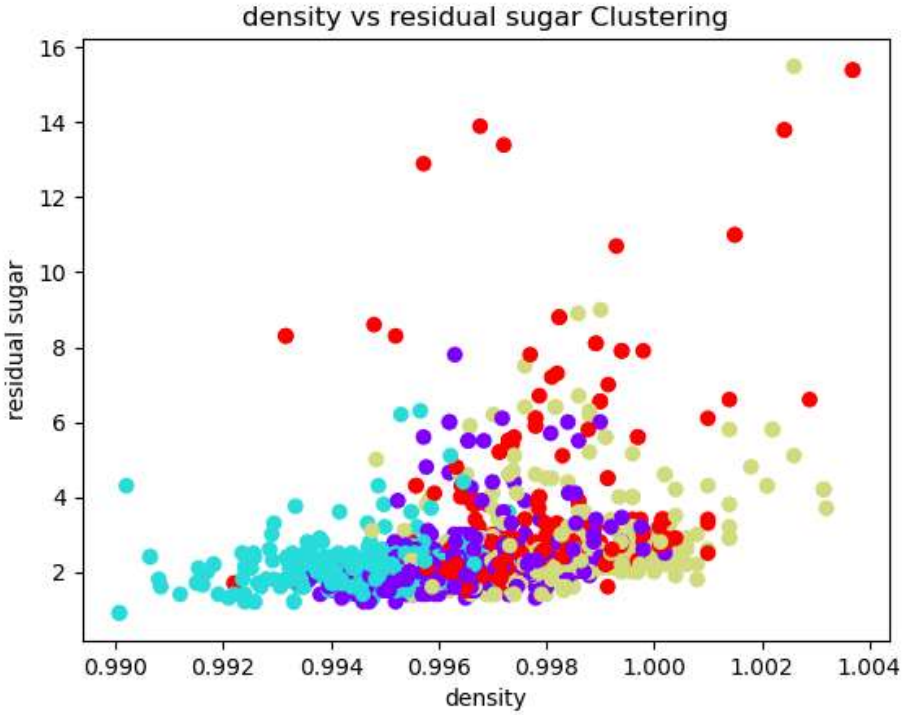
In [22]:
```
plt.scatter(df['fixed acidity'], df['alcohol'], c=clusters, cmap='rainbow')

plt.title('fixed acidity  vs alcohol Clustering')
plt.xlabel('fixed acidity ')
plt.ylabel('alcohol')

plt.show()
```



In [30]:
```
plt.scatter(df['fixed acidity'], df['pH'], c=clusters, cmap='rainbow')

plt.title('fixed acidity vs pH Clustering')
plt.xlabel('fixed acidity ')
plt.ylabel('pH')

plt.show()
```



In [32]:
```
plt.scatter(df['density'], df['residual sugar'], c=clusters, cmap='rainbow')

plt.title('density vs residual sugar Clustering')
plt.xlabel('density')
plt.ylabel('residual sugar')

plt.show()
```



==========

FAILED ATTEMPTtrain_path = 'CustomerTrainingData.csv' test_path = 'CustomerTestData.csv' df_train = pd.read_csv(train_path) df_test = pd.read_csv(test_path)df_train.info(), df_train.head()# Only take numerical categories features = ["Income", "Age", "Experience", "CURRENT_JOB_YRS", "CURRENT_HOUSE_YRS"] X_train = df_train[features] X_test = df_test[features] # features normalisation scaler = StandardScaler() X_train_scaled = scaler.fit_transform(X_train) #X_test_scaled = scaler.transform(X_test) # Visualisation sns.pairplot(df_train, vars=features, diag_kind='kde', plot_kws= {'alpha': 0.5}) plt.suptitle("Pairplot of Training Data Features", y=1.02) plt.show() wcss = [] for k in range(1, 11) : kmeans = KMeans(n_clusters=k, random_state=42, n_init=10) kmeans.fit(X_train_scaled) wcss.append(kmeans.inertia_) plt.figure(figsize=(8, 5)) plt.plot(range(1, 11), wcss, marker='o', linestyle='-') plt.xlabel('Number of Clusters (k)') plt.ylabel('WCSS (Within-Cluster Sum of Squares)') plt.title('Elbow Method for Optimal k') plt.show()kmeans = KMeans(n_clusters=3, random_state=42, n_init=20, max_iter=10, init='k-means++', algorithm='elkan') kmeans.fit(X_train_scaled) clusters = kmeans.labels_ silhouette = silhouette_score(X_train_scaled, clusters) print(f'Silhouette Score: {silhouette:.4f}')plt.scatter(df_train['Experience'], df_train['Age'], c=clusters, cmap='rainbow') plt.title('Experience vs Age Clustering') plt.xlabel('Experience ') plt.ylabel('Age') # Show the plot plt.show() categorical_features = ['Married/Single', 'House_Ownership', 'Car_Ownership','Profession'] encoder = OneHotEncoder(drop='first', sparse_output=False)

encoded_cats = encoder.fit_transform(df_train[categorical_features]) encoded_df = pd.DataFrame(encoded_cats, columns=encoder.get_feature_names_out(categorical_features)) df = df_train.drop(columns=categorical_features) df = pd.concat([df, encoded_df], axis=1) print(df.head())# Only take numerical categories features = ["Income", "Age", "Experience", "CURRENT_JOB_YRS", "CURRENT_HOUSE_YRS","Married/Single", "House_Ownership", "Car_Ownership","Profession"] X_train = df[features] X_test = df[features] scaler = StandardScaler() X_train_scaled = scaler.fit_transform(X_train) #X_test_scaled = scaler.transform(X_test) sns.pairplot(df_train, vars=features, diag_kind='kde', plot_kws={'alpha': 0.5}) plt.suptitle("Pairplot of Training Data Features", y=1.02) plt.show()wcss = [] for k in range(1, 11) : kmeans = KMeans(n_clusters=k, random_state=42, n_init=10) kmeans.fit(X_train_scaled) wcss.append(kmeans.inertia_) plt.figure(figsize=(8, 5)) plt.plot(range(1, 11), wcss, marker='o', linestyle='-') plt.xlabel('Number of Clusters (k)') plt.ylabel('WCSS (Within-Cluster Sum of Squares)') plt.title('Elbow Method for Optimal k') plt.show()kmeans = KMeans(n_clusters=7, random_state=42, n_init=10) kmeans.fit(X_train_scaled) clusters = kmeans.labels_ silhouette = silhouette_score(X_train_scaled, clusters) print(f'Silhouette Score: {silhouette:.4f}')FAILED ATTEMPT AT USING CANCER DATA DUE TO DATA BEING ONLY 1 AND 0

==========

cancer_path = 'lungCancerData.csv' df_cancer = pd.read_csv(cancer_path)# Only take numerical categories features = ['SMOKING', 'YELLOW_FINGERS', 'ANXIETY', 'PEER_PRESSURE', 'CHRONIC DISEASE', 'FATIGUE ', 'ALLERGY ', 'WHEEZING', 'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH', 'SWALLOWING DIFFICULTY', 'CHEST PAIN'] X_train = df_cancer[features] # features normalisation scaler = StandardScaler() X_train_scaled = scaler.fit_transform(X_train) wcss = [] for k in range(1, 11) : kmeans = KMeans(n_clusters=k, random_state=42, n_init=10) kmeans.fit(X_train_scaled) wcss.append(kmeans.inertia_) plt.figure(figsize=(8, 5)) plt.plot(range(1, 11), wcss, marker='o', linestyle='-') plt.xlabel('Number of Clusters (k)') plt.ylabel('WCSS (Within-Cluster Sum of Squares)') plt.title('Elbow Method for Optimal k') plt.show()kmeans = KMeans(n_clusters=4, random_state=42, n_init=10) kmeans.fit(X_train_scaled) clusters = kmeans.labels_ silhouette = silhouette_score(X_train_scaled, clusters) print(f'Silhouette Score: {silhouette:.4f}')plt.scatter(df_cancer['SMOKING'], df_cancer['SHORTNESS OF BREATH'], c=clusters, cmap='rainbow') plt.title('SMOKING vs SHORTNESS OF BREATH Clustering') plt.xlabel('SMOKING ') plt.ylabel('SHORTNESS OF BREATH') # Show the plot plt.show()plt.scatter(df_cancer["AGE"],df_cancer["SHORTNESS OF BREATH"]) plt.xlabel('AGE') plt.ylabel('SHORTNESS OF BREATH')

==========