AIM :- Implement K-Means Clustering Algorithm

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
df=pd.read_csv("/cars.csv")
```

```
df.shape
```

```
(38531, 30)
```

```
df.head()
```

	manufacturer_name	model_name	transmission	color	odometer_value	year_produced	engine_fuel	engine_has_gas	engine_type
0	Subaru	Outback	automatic	silver	190000	2010	gasoline	False	gasoline
1	Subaru	Outback	automatic	blue	290000	2002	gasoline	False	gasoline
2	Subaru	Forester	automatic	red	402000	2001	gasoline	False	gasoline
3	Subaru	Impreza	mechanical	blue	10000	1999	gasoline	False	gasoline
4	Subaru	Legacy	automatic	black	280000	2001	gasoline	False	gasoline

5 rows × 30 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38531 entries, 0 to 38530
Data columns (total 30 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   manufacturer_name  38531 non-null  object 
 1   model_name         38531 non-null  object 
 2   transmission       38531 non-null  object 
 3   color              38531 non-null  object 
 4   odometer_value     38531 non-null  int64  
 5   year_produced      38531 non-null  int64  
 6   engine_fuel        38531 non-null  object 
 7   engine_has_gas     38531 non-null  bool   
 8   engine_type        38531 non-null  object 
 9   engine_capacity    38521 non-null  float64
 10  body_type          38531 non-null  object 
 11  has_warranty       38531 non-null  bool   
 12  state              38531 non-null  object 
 13  drivetrain         38531 non-null  object 
 14  price_usd          38531 non-null  float64
 15  is_exchangeable    38531 non-null  bool   
 16  location_region    38531 non-null  object 
 17  number_of_photos   38531 non-null  int64  
 18  up_counter         38531 non-null  int64  
 19  feature_0          38531 non-null  bool   
 20  feature_1          38531 non-null  bool   
 21  feature_2          38531 non-null  bool   
 22  feature_3          38531 non-null  bool   
 23  feature_4          38531 non-null  bool   
 24  feature_5          38531 non-null  bool   
 25  feature_6          38531 non-null  bool   
 26  feature_7          38531 non-null  bool   
 27  feature_8          38531 non-null  bool   
 28  feature_9          38531 non-null  bool   
 29  duration_listed    38531 non-null  int64  
dtypes: bool(13), float64(2), int64(5), object(10)
memory usage: 5.5+ MB
```

```
df.describe()
```

	odometer_value	year_produced	engine_capacity	price_usd	number_of_photos	up_counter	duration_listed
count	38531.000000	38531.000000	38521.000000	38531.000000	38531.000000	38531.000000	38531.000000
mean	248864.638447	2002.943734	2.055161	6639.971021	9.649062	16.306091	80.577249
std	136072.376530	8.065731	0.671178	6428.152018	6.093217	43.286933	112.826569
min	0.000000	1942.000000	0.200000	1.000000	1.000000	1.000000	0.000000
25%	158000.000000	1998.000000	1.600000	2100.000000	5.000000	2.000000	23.000000
50%	250000.000000	2003.000000	2.000000	4800.000000	8.000000	5.000000	59.000000
75%	325000.000000	2009.000000	2.300000	8990.000000	12.000000	16.000000	91.000000
max	1000000.000000	2019.000000	8.000000	50000.000000	86.000000	1861.000000	2232.000000

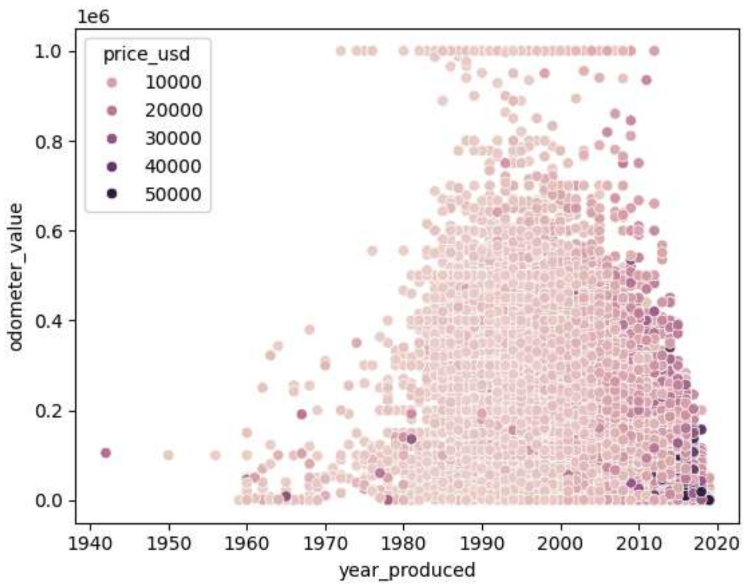
```
df.isnull().sum()
```

	0
manufacturer_name	0
model_name	0
transmission	0
color	0
odometer_value	0
year_produced	0
engine_fuel	0
engine_has_gas	0
engine_type	0
engine_capacity	10
body_type	0
has_warranty	0
state	0
drivetrain	0
price_usd	0
is_exchangeable	0
location_region	0
number_of_photos	0
up_counter	0
feature_0	0
feature_1	0
feature_2	0
feature_3	0
feature_4	0
feature_5	0
feature_6	0
feature_7	0
feature_8	0
feature_9	0
duration_listed	0

dtype: int64

```
sns.scatterplot(data=df,x='year_produced',y='odometer_value',hue='price_usd')
```

```
<Axes: xlabel='year_produced', ylabel='odometer_value'>
```



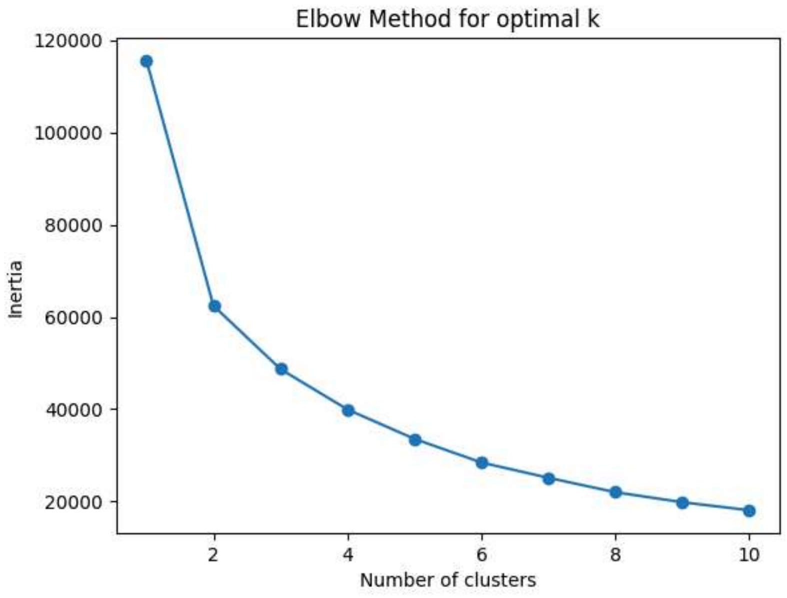
```
from sklearn.preprocessing import StandardScaler
#normalize the data
scaler=StandardScaler()
scaler.fit(df[['odometer_value','year_produced','price_usd']]) # intro to data for scalar
df_normalized=scaler.transform(df[['odometer_value','year_produced','price_usd']]) # makes it in one scale
print(df_normalized[:5])
```

```
[[-0.43260362  0.87485665  0.66272301]
 [ 0.30230894 -0.11700687 -0.25512656]
 [ 1.125411   -0.24098981 -0.59737555]
 [-1.75544621 -0.48895569  0.52255649]
 [ 0.22881768 -0.24098981 -0.70096654]]
```

```
from sklearn.cluster import KMeans
```

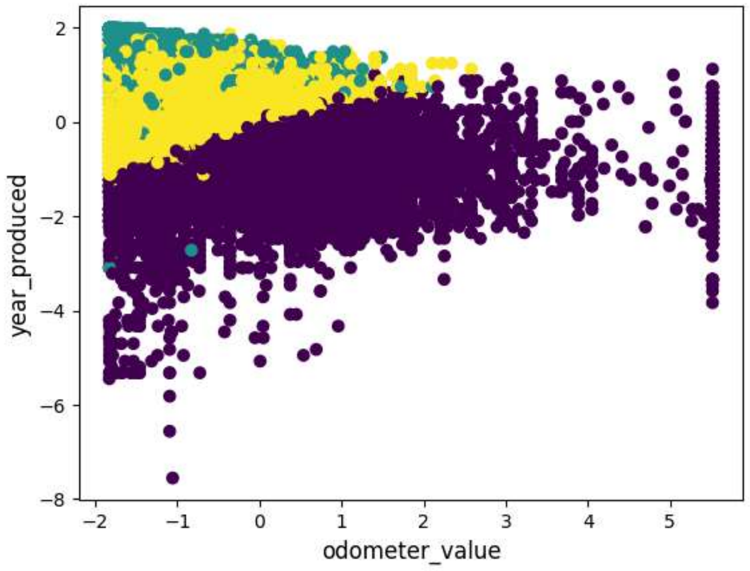
```
#calculate the within-clustera sum of square across different clusters counts
inertia=[]
for i in range(1,11):
  kmeans=KMeans(n_clusters=i,random_state=42,n_init='auto')
  kmeans.fit(df_normalized)
  inertia.append(kmeans.inertia_) # plot the elbow graph
```

```
plt.plot(range(1,11),inertia,marker='o')
plt.title('Elbow Method for optimal k')
plt.xlabel("Number of clusters")
plt.ylabel("Inertia")
plt.show()
```



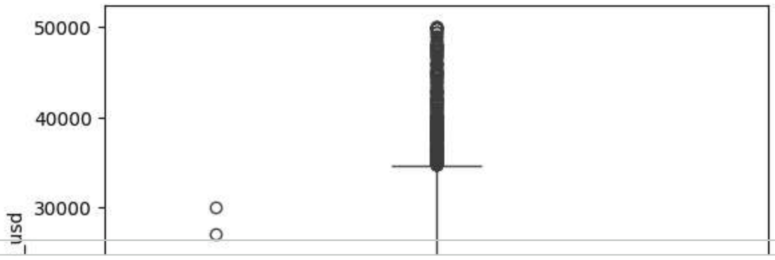
```
#Elbow is at three clusters
kmeans=KMeans(n_clusters=3,random_state=42,n_init='auto')
clusters=kmeans.fit_predict(df_normalized)
```

```
#choose two dimensions to plot ( odometer_value and year_produced)
plt.scatter(df_normalized[:,0],df_normalized[:,1],c=clusters,cmap='viridis',marker='o')# 0- refers to odometer and 1- year
plt.xlabel('odometer_value',fontsize=12)
plt.ylabel('year_produced',fontsize=12)
plt.show()
```

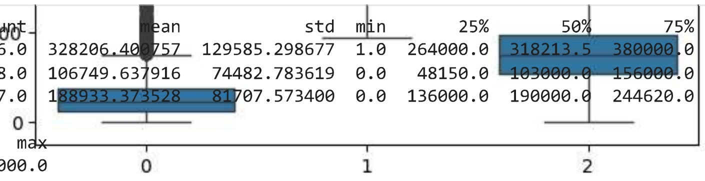


```
sns.boxplot(x=clusters,y=df['price_usd'])
summary_stats=df.groupby(clusters)['price_usd'].describe()
print(summary_stats)
```

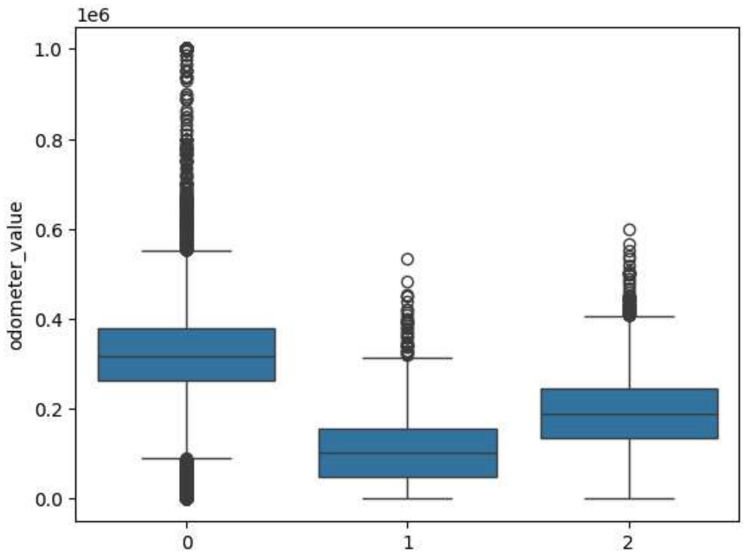
```
     count          mean          std      min       25%      50%       75%  \
0  19276.0   2723.463427  1985.037525     1.00   1249.75   2300.0   3782.31
1   4568.0  19905.578398  7566.002147  9437.54  14746.28  17500.0  22753.48
2  14687.0   7654.290208  3088.330504     1.00   5500.00   7500.0   9700.00

       max
0  30000.0
1  50000.0
2  18500.0
```



```
sns.boxplot(x=clusters,y=df['odometer_value'])
summary_stats=df.groupby(clusters)['odometer_value'].describe()
print(summary_stats)
```

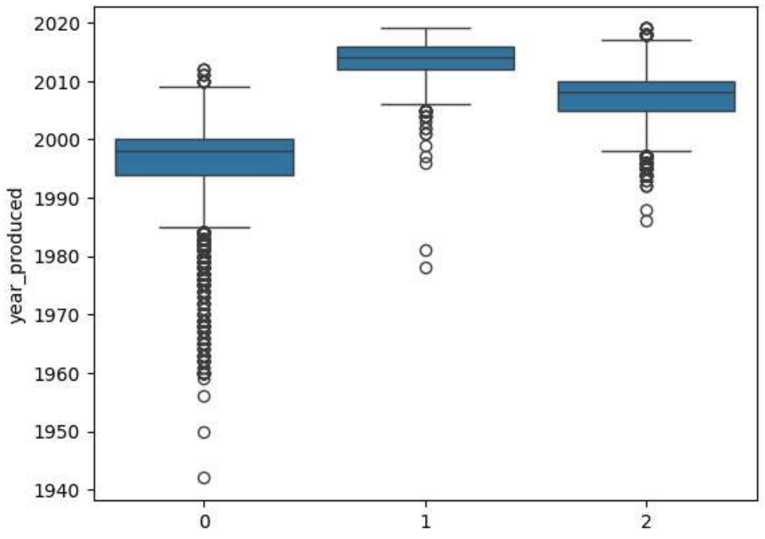


```
     count           mean            std  min       25%       50%       75%  \
0  19276.0  328206.400757  129585.298677  1.0  264000.0  318213.5  380000.0
1   4568.0  106749.637916   74482.783619  0.0   48150.0  103000.0  156000.0
2  14687.0  188933.373528   81707.573400  0.0  136000.0  190000.0  244620.0

         max
0  1000000.0
1   535000.0
2   600000.0
```



```
#CLUSTER ANALYSIS
sns.boxplot(x=clusters,y=df['year_produced'])
summary_stats=df.groupby(clusters)['year_produced'].describe()
print(summary_stats)
```

```
     count         mean       std     min     25%     50%     75%     max
0  19276.0  1996.665335  5.411023  1942.0  1994.0  1998.0  2000.0  2012.0
1   4568.0  2014.069834  3.211429  1978.0  2012.0  2014.0  2016.0  2019.0
2  14687.0  2007.723361  3.994489  1986.0  2005.0  2008.0  2010.0  2019.0
```



```
#adding clusters column to the dataframe
df['cluster']=clusters
#now lets see the first five instances of the updated dataset
print(df.head(10))
```

```
  manufacturer_name model_name transmission   color  odometer_value  \
0            Subaru    Outback    automatic  silver          190000   
1            Subaru    Outback    automatic    blue          290000   
2            Subaru   Forester    automatic     red          402000   
3            Subaru    Impreza   mechanical    blue           10000   
4            Subaru     Legacy    automatic   black          280000   
5            Subaru    Outback    automatic  silver          132449   
6            Subaru   Forester    automatic   black          318280   
7            Subaru     Legacy    automatic  silver          350000   
8            Subaru    Outback    automatic    grey          179000   
9            Subaru   Forester    automatic  silver          571317   

   year_produced engine_fuel  engine_has_gas engine_type  engine_capacity  \
0           2010    gasoline           False    gasoline              2.5   
1           2002    gasoline           False    gasoline              3.0   
2           2001    gasoline           False    gasoline              2.5   
3           1999    gasoline           False    gasoline              3.0   
4           2001    gasoline           False    gasoline              2.5   
5           2011    gasoline           False    gasoline              2.5   
6           1998    gasoline           False    gasoline              2.5   
7           2004    gasoline           False    gasoline              2.5   
8           2010    gasoline           False    gasoline              2.5   
9           1999    gasoline           False    gasoline              2.5   

   ... feature_2  feature_3 feature_4 feature_5  feature_6  feature_7  \
0  ...      True       True     False      True      False       True   
1  ...     False      False      True      True      False      False   
2  ...     False      False     False     False      False      False   
3  ...     False      False     False     False      False      False   
4  ...     False       True      True     False      False      False   
5  ...     False      False     False      True      False       True   
6  ...     False      False      True      True      False      False   
7  ...      True      False     False     False      False      False   
8  ...      True       True      True      True       True       True   
9  ...      True      False     False      True      False      False   

  feature_8  feature_9  duration_listed  cluster  
0      True       True               16        2  
1     False       True               83        0  
2      True       True              151        0  
3     False      False               86        2  
4     False       True                7        0  
5      True       True               67        1  
6      True       True              307        0  
7     False       True               73        0  
8      True       True               87        2  
9     False       True               43        0  

[10 rows x 31 columns]
```

```
#load the original dataset(copy)
df_ori=pd.read_csv("/cars.csv")
df_ori.head() # without the clusters column
```

	manufacturer_name	model_name	transmission	color	odometer_value	year_produced	engine_fuel	engine_has_gas	engine_type	e
0	Subaru	Outback	automatic	silver	190000	2010	gasoline	False	gasoline	
1	Subaru	Outback	automatic	blue	290000	2002	gasoline	False	gasoline	
2	Subaru	Forester	automatic	red	402000	2001	gasoline	False	gasoline	
3	Subaru	Impreza	mechanical	blue	10000	1999	gasoline	False	gasoline	
4	Subaru	Legacy	automatic	black	280000	2001	gasoline	False	gasoline	

5 rows × 30 columns

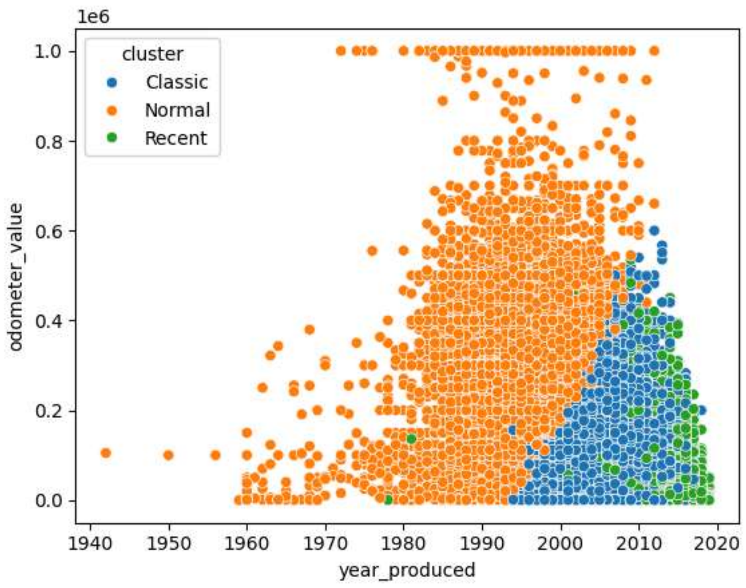
```
#Map the cluster number to a meaning
clusters_names_map={
  0:'Normal',
  1:'Recent',
  2:'Classic'
}

#create a new column with the cluster name
df_ori['cluster']=pd.Series(clusters).map(clusters_names_map)
#printing the new datatframe
df_ori.head(3)
```

	manufacturer_name	model_name	transmission	color	odometer_value	year_produced	engine_fuel	engine_has_gas	engine_type	e
0	Subaru	Outback	automatic	silver	190000	2010	gasoline	False	gasoline	
1	Subaru	Outback	automatic	blue	290000	2002	gasoline	False	gasoline	
2	Subaru	Forester	automatic	red	402000	2001	gasoline	False	gasoline	

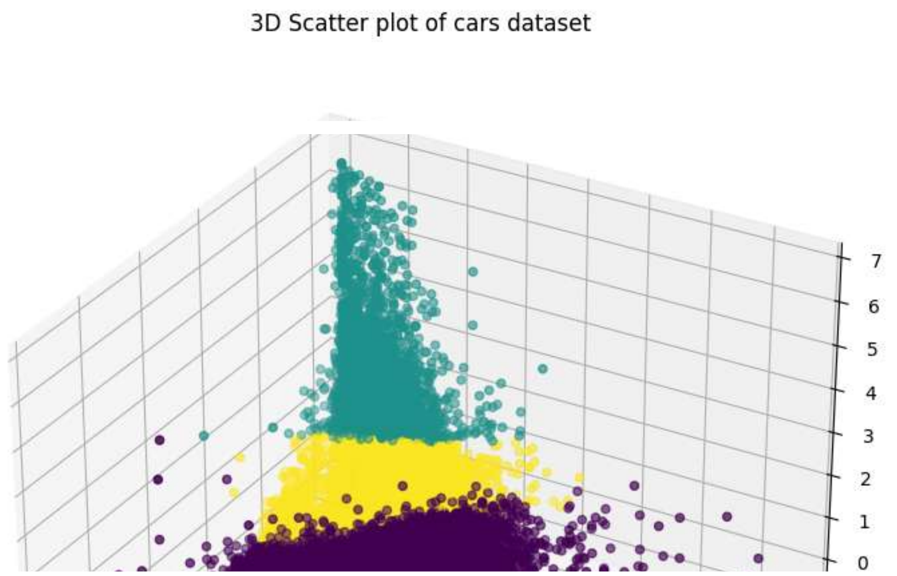
3 rows × 31 columns

```
#plot tye new dataset we made with clusters
sns.scatterplot(data=df_ori,x='year_produced',y='odometer_value',hue='cluster')
```

```
<Axes: xlabel='year_produced', ylabel='odometer_value'>
```



```
fig=plt.figure(figsize=(12,9))
ax=fig.add_subplot(111,projection='3d')

#scatterplot using the first three features of the cars dataset
ax.scatter(df_normalized[:,0], #odometer_value
           df_normalized[:,1], # year_produced
           df_normalized[:,2], # price_usd
           c=clusters, #use cluster label as color encoding
           cmap='viridis',
             marker='o')

#set labels according to the features we used
ax.set_xlabel('odometer_value')
ax.set_ylabel('year_produced')
ax.set_zlabel('price_usd')
#title of the plot
plt.title('3D Scatter plot of cars dataset')
```


3D Scatter plot of cars dataset