Dataset:

Car Details from CarDekho (Used Car Dataset)

Goal:

Building predictive and analytical models for used car pricing and market trends

Research Questions:

1.Can we predict the selling price of a used car based on its features (year, km driven, fuel type, transmission, etc.)? Use: Helps car dealers, buyers, and sellers estimate fair market prices and make informed decisions while buying or selling used cars.

2.Can we classify whether a car's resale value is high, medium, or low using its attributes (brand, age, mileage, etc.)? Use: Supports customers in understanding car depreciation rates and assists businesses in segmenting cars for pricing strategy and loan evaluation.

3.What are the key factors influencing car prices across different fuel types and transmission modes (manual vs automatic)? Use: Provides insights for manufacturers, dealers, and policymakers into consumer preferences and market demand trends

```
# Mount the data
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
#importing Numpy and pandas
import numpy  as np
import pandas as pd

#Reading csv file from drive
# read the data
df=pd.read_csv('/content/gdrive/My Drive/Colab Notebooks/CAR DETAILS FROM CAR DEKHO.csv')
#Shape of the data
print("Shape of data :")
df.shape
```

```
Shape of data :
(4340, 8)
```

```
# Head of the data
df.head(3)
```

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner

Next steps: Generate code with df New interactive sheet

```
# Tail of the data
df.tail(3)
```

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
4337	Maruti 800 AC BSIII	2009	110000	83000	Petrol	Individual	Manual	Second Owner
4338	Hyundai Creta 1.6 CRDi SX Option	2016	865000	90000	Diesel	Individual	Manual	First Owner
4339	Renault KWID RXT	2016	225000	40000	Petrol	Individual	Manual	First Owner

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4340 entries, 0 to 4339
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   name           4340 non-null   object
 1   year           4340 non-null   int64 
 2   selling_price  4340 non-null   int64 
 3   km_driven      4340 non-null   int64 
 4   fuel           4340 non-null   object
 5   seller_type    4340 non-null   object
 6   transmission   4340 non-null   object
 7   owner          4340 non-null   object
dtypes: int64(3), object(5)
memory usage: 271.4+ KB
```

```
df.describe()
```

	year	selling_price	km_driven	high
count	4340.000000	4.340000e+03	4340.000000	0.0
mean	2013.090783	5.041273e+05	66215.777419	NaN
std	4.215344	5.785487e+05	46644.102194	NaN
min	1992.000000	2.000000e+04	1.000000	NaN
25%	2011.000000	2.087498e+05	35000.000000	NaN
50%	2014.000000	3.500000e+05	60000.000000	NaN
75%	2016.000000	6.000000e+05	90000.000000	NaN
max	2020.000000	8.900000e+06	806599.000000	NaN

```
df.isnull().values.any()
```

```
np.True_
```

```
# Count Null per column
df.isnull().values.any()
```

```
np.True_
```

```
# set random seed for reproducibility
np.random.seed(42)
# Choose 5 Randomrow indicies
rows_to_null = np.random.choice(df.index, size=5 , replace=False)
# Replace Parameter
df.loc[rows_to_null,"high"] = np.nan

# Save modified dataset
df.to_csv("CAR DETAILS FROM CAR DEKHO.csv",index = False)

# check if there is any null
df.isnull().values.any()
```

```
np.True_
```

```
# count null per column
```

```
df.isnull().values.sum()
df[df["high"].isnull()]
```

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner	high
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner	NaN
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner	NaN
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner	NaN
3	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner	NaN
4	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner	NaN
...	...	...	...	...	...	...	...	...	...
4335	Hyundai i20 Magna 1.4 CRDi (Diesel)	2014	409999	80000	Diesel	Individual	Manual	Second Owner	NaN
4336	Hyundai i20 Magna 1.4 CRDi	2014	409999	80000	Diesel	Individual	Manual	Second Owner	NaN
4337	Maruti 800 AC BSIII	2009	110000	83000	Petrol	Individual	Manual	Second Owner	NaN
4338	Hyundai Creta 1.6 CRDi SX Option	2016	865000	90000	Diesel	Individual	Manual	First Owner	NaN
4339	Renault KWID RXT	2016	225000	40000	Petrol	Individual	Manual	First Owner	NaN

4340 rows × 9 columns

```
df['high'] = df['high'].replace('Unkown',np.nan)
print(df.isnull().values.sum())
```

```
4340
```

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

	0
name	0
year	0
selling_price	0
km_driven	0
fuel	0
seller_type	0
transmission	0
owner	0
high	4340

dtype: int64

```
#Importing libraries for data visualization
import matplotlib.pyplot as plt
import seaborn as sns
```

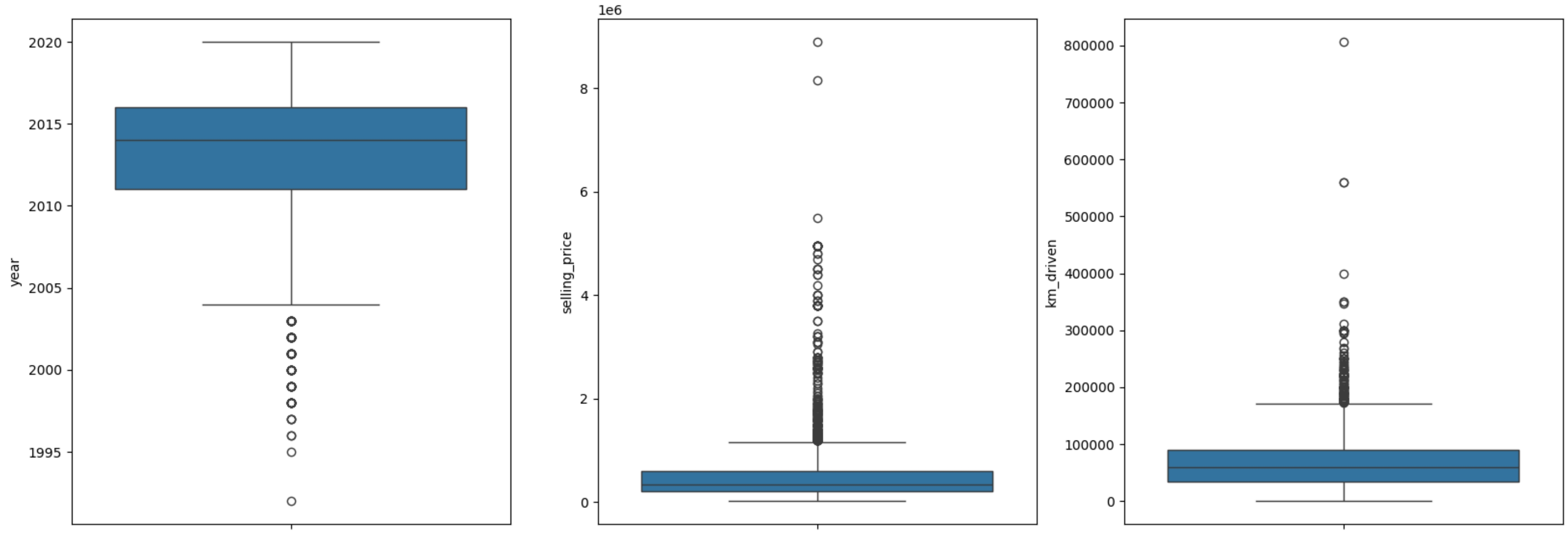
```
# plotting  boxplots
# here there are 3 columns of numeric values , so we'll plot these three columns
plt.figure(figsize = (20,15))

plt.subplot(2,3,1)
sns.boxplot(df['year'])

plt.subplot(2,3,2)
sns.boxplot(df['selling_price'])

plt.subplot(2,3,3)
sns.boxplot(df['km_driven'])
```

```
<Axes: ylabel='km_driven'>
```



```
import random

col = 'selling_price'
# dEFINE OUTLIER values (extremly low an dextrem,ly high or unsual values eg:negative values  )
outlier_values = [5_000_000,-5,50,4_555_000.10]

# Randomly pick some indices to replace
random_indices = np.random.choice(df.index,size=10,replace=False)

# Replace selected rows with random outlier values
for i in random_indices :
  df.at[i,col] = random.choice(outlier_values)

print("Injected outliers at rows :",random_indices)
print(df.loc[random_indices, col])
```

```
Injected outliers at rows : [1142 1417 3369 1287 1701 1758 1233 3000 2597 2389]
1142    4555000.1
1417         50.0
3369    5000000.0
1287         -5.0
1701    4555000.1
1758    5000000.0
1233    5000000.0
```

```
3000    5000000.0
2597         -5.0
2389    5000000.0
Name: selling_price, dtype: float64
/tmp/ipython-input-1169322841.py:12: FutureWarning: Setting an item of incompatible dtype is deprecated and will raise an error in a fu
  df.at[i,col] = random.choice(outlier_values)
```

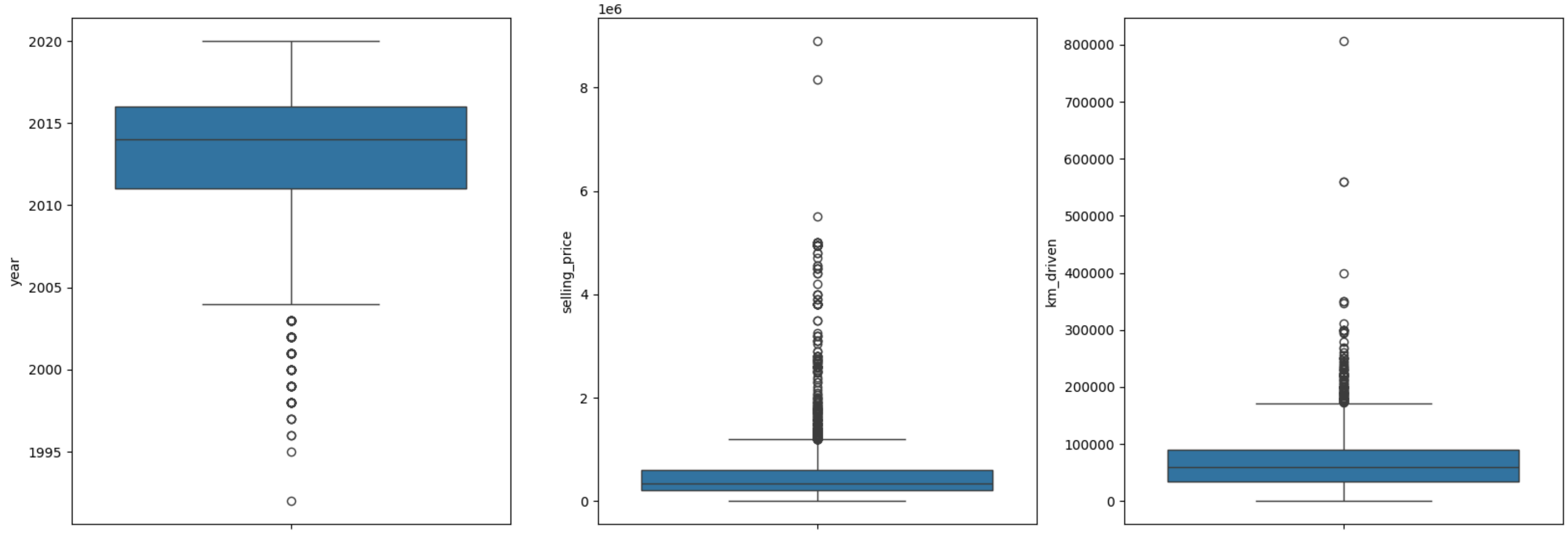
```
plt.figure(figsize = (20,15))

plt.subplot(2,3,1)
sns.boxplot(df['year'])

plt.subplot(2,3,2)
sns.boxplot(df['selling_price'])

plt.subplot(2,3,3)
sns.boxplot(df['km_driven'])
```

```
<Axes: ylabel='km_driven'>
```



```
# Detect duplicate rows
duplicates = df[df.duplicated()]
print("Number of duplicated rows :",len(duplicates))
print(duplicates.head())
```

```
Number of duplicated rows : 761
                        name  year  selling_price  km_driven    fuel  \
13              Maruti 800 AC  2007        60000.0      70000  Petrol
14  Maruti Wagon R LXI Minor  2007       135000.0      50000  Petrol
15      Hyundai Verna 1.6 SX  2012       600000.0     100000  Diesel
16    Datsun RediGO T Option  2017       250000.0      46000  Petrol
17     Honda Amaze VX i-DTEC  2014       450000.0     141000  Diesel
```