Explanatory Data Analyst Car Seling in Dekho

Oleh : Ika Lulus Yuliatin

Car Dekho is an Indian auto portal that helps its users with car research, finance, insurance, used cars, and any other aspect of car buying and selling. The company has tie-ups with many auto manufacturers, car dealers, and numerous financial institutions to facilitate the purchase of vehicles.

In this report, we will do data visualization analysis from 2 kinds of variable continue and 6 kinds of variable discrete The details of variables included in the dataset are :

1) Car Name 2) Year 3) Selling Price 4) Kms driven 5) Fuel 6) Seller type 7) Transmission 8) Owner

Source of Data : https://www.kaggle.com/datasets/akshaydattatraykhare/car-details-dataset/code (https://www.kaggle.com/datasets/akshaydattatraykhare/car-details-dataset/code)

In [69]:
```
# Common
import numpy as np
import pandas as pd

# Data Visualization
import seaborn as sns
import plotly.express as px
import matplotlib.pyplot as plt
from matplotlib import colors
```

In [2]:
```
car_df = pd.read_csv('CAR DETAILS FROM CAR DEKHO.csv')
car_df.head()
```

Out[2]:

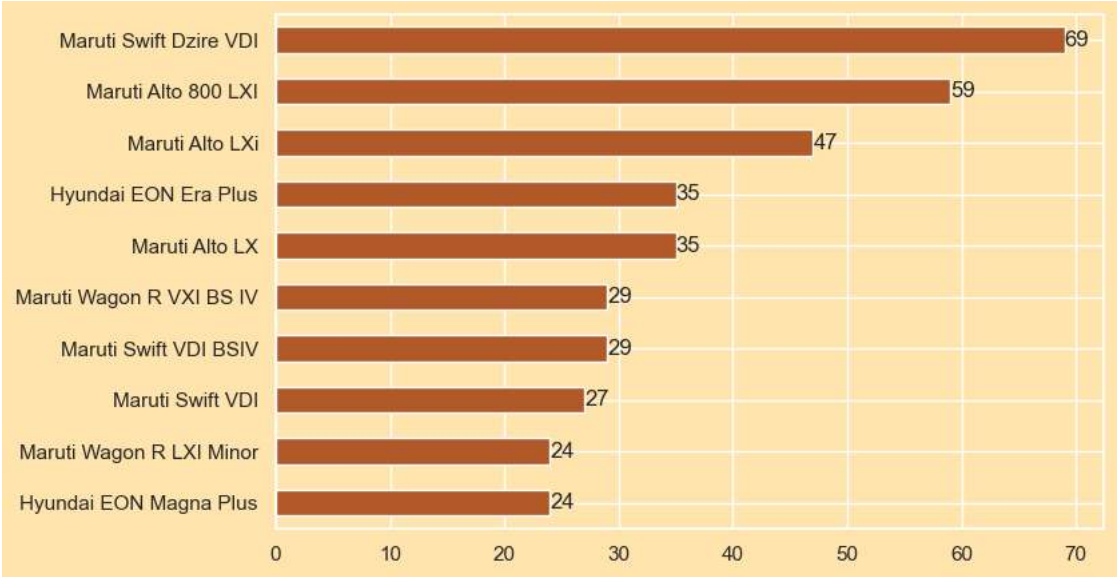
	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner
3	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner
4	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner

In [3]:
```
car_df.info()
```

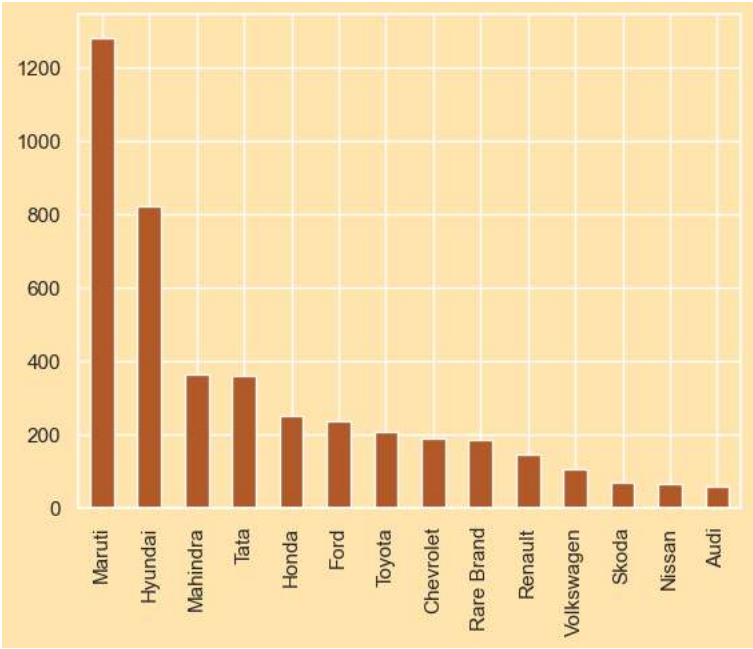
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4340 entries, 0 to 4339
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   name           4340 non-null   object
 1   year           4340 non-null   int64 
 2   selling_price  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
```

In [4]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
cmap = colors.ListedColormap(["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"])

ax = car_df['name'].value_counts().head(10).sort_values().plot(kind='barh', figsize=(8,5), cmap='Paired_r')
ax.bar_label(ax.containers[0], )
```

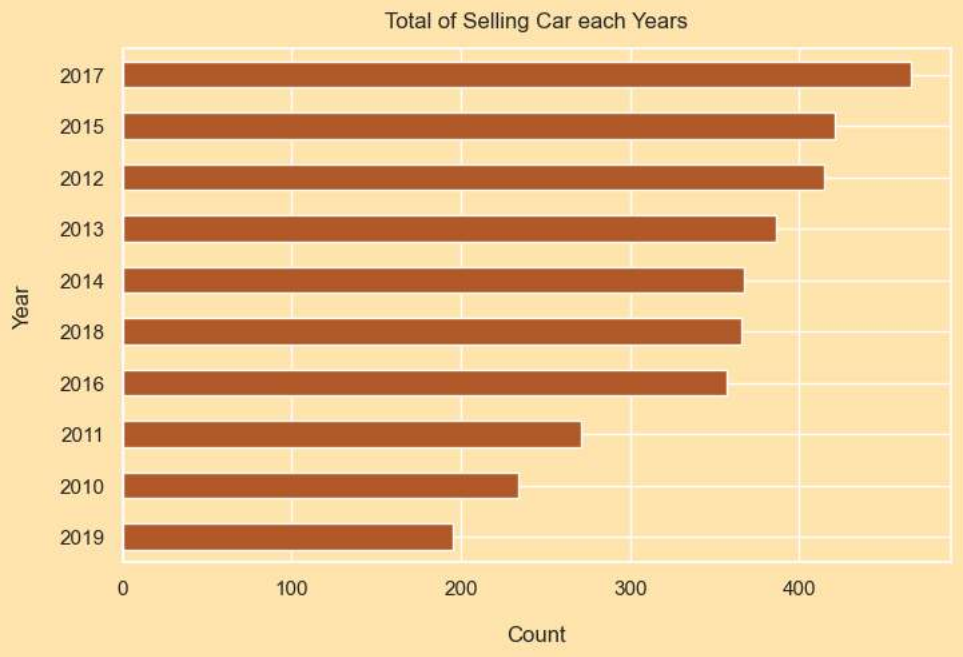
Out[4]:
```
[Text(0, 0, '24'),
 Text(0, 0, '24'),
 Text(0, 0, '27'),
 Text(0, 0, '29'),
 Text(0, 0, '29'),
 Text(0, 0, '35'),
 Text(0, 0, '35'),
 Text(0, 0, '47'),
 Text(0, 0, '59'),
 Text(0, 0, '69')]
```



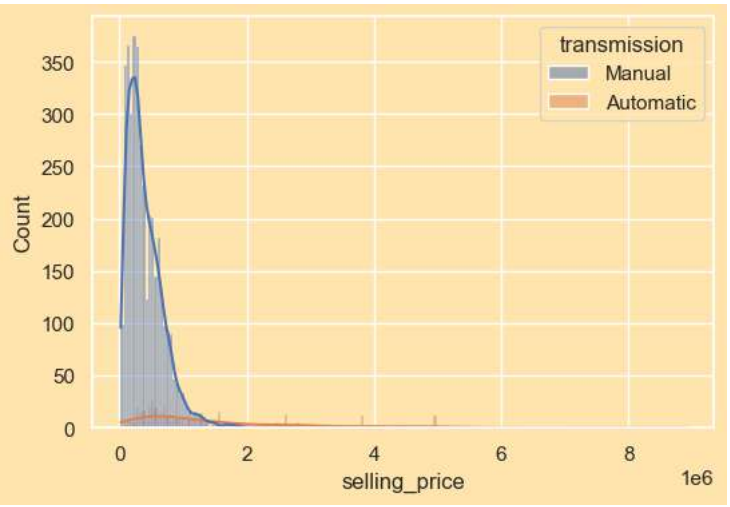
In [5]:
```
car_df["make"] = car_df.name.apply(lambda x : x.split(' ')[0])
car_df.loc[:,"make"] = car_df.make.apply(lambda x : "Rare Brand" if car_df.make.value_counts(normalize = True)[x] < 0.01  else x)
car_df.make.value_counts(normalize = False).plot(kind = "bar", cmap='Paired_r')
plt.show()
```



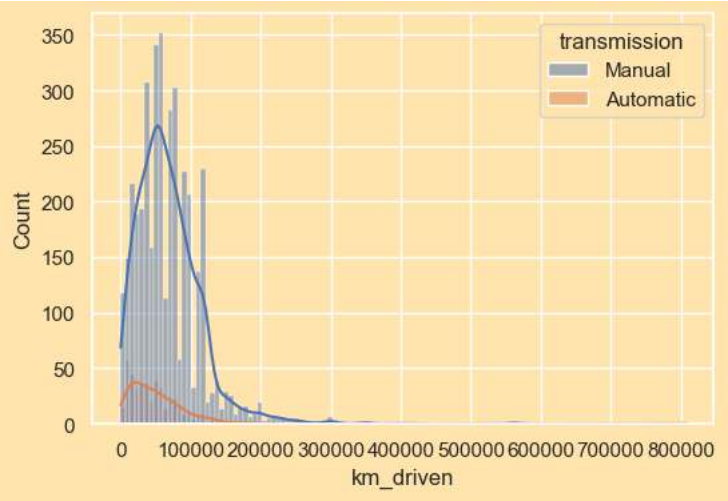
In [6]:
```
ax = car_df['year'].value_counts().head(10).sort_values().plot(kind='barh', figsize=(8,5), cmap='Paired_r')
plt.xlabel("Count", labelpad=14)
plt.ylabel("Year", labelpad=14)
plt.title("Total of Selling Car each Years", y=1.02);
```



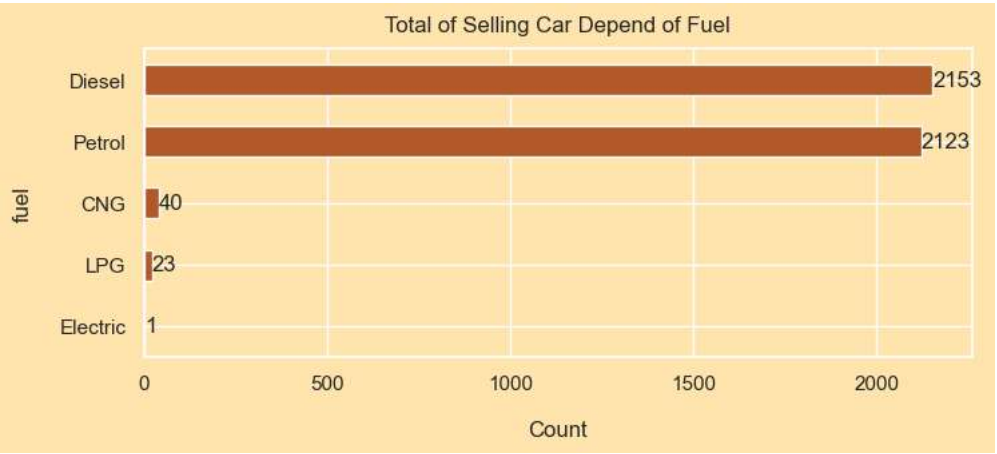
In [7]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
cmap = colors.ListedColormap(["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"])
plt.figure(figsize=(6,4))
sns.histplot(data=car_df,x='selling_price',kde=True,hue='transmission');
```



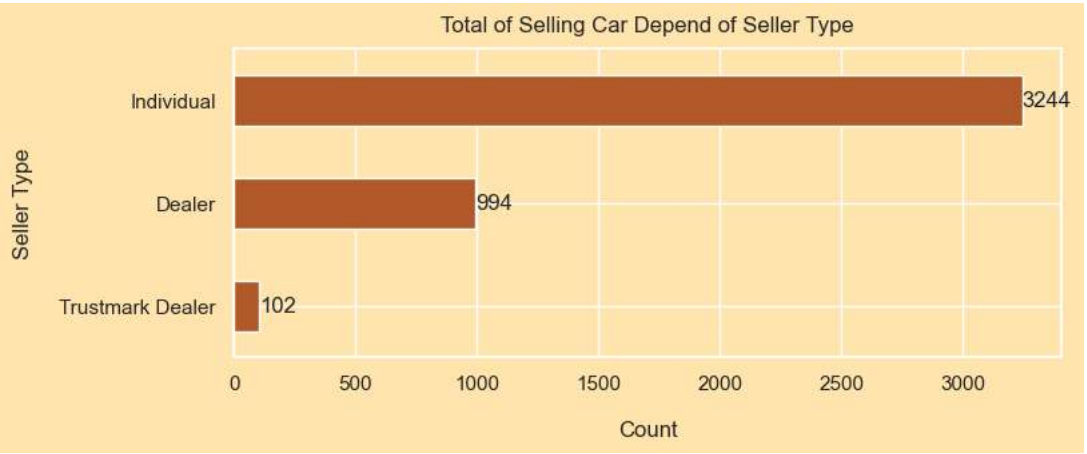
In [8]:
```
plt.figure(figsize=(6,4))
sns.histplot(data=car_df,x='km_driven',kde=True,hue='transmission');
```



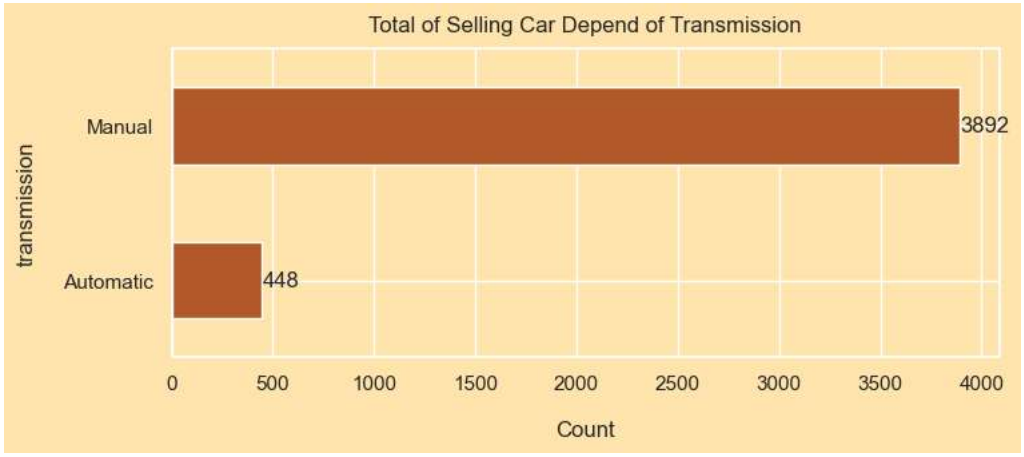
In [9]:
```
ax = car_df['fuel'].value_counts().head(10).sort_values().plot(kind='barh', figsize=(8,3), cmap='Paired_r')
ax.bar_label(ax.containers[0], )
plt.xlabel("Count", labelpad=14)
plt.ylabel("fuel", labelpad=14)
plt.title("Total of Selling Car Depend of Fuel", y=1.02);
```



In [10]:
```
ax = car_df['seller_type'].value_counts().head(10).sort_values().plot(kind='barh', figsize=(8,3), cmap='Paired_r')
ax.bar_label(ax.containers[0], )
plt.xlabel("Count", labelpad=14)
plt.ylabel("Seller Type", labelpad=14)
plt.title("Total of Selling Car Depend of Seller Type", y=1.02);
```

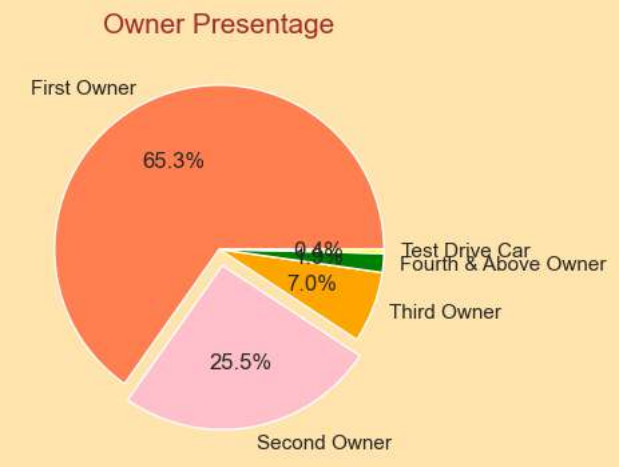


In [11]:
```
ax = car_df['transmission'].value_counts().head(10).sort_values().plot(kind='barh', figsize=(8,3), cmap='Paired_r')
ax.bar_label(ax.containers[0], )
plt.xlabel("Count", labelpad=14)
plt.ylabel("transmission", labelpad=14)
plt.title("Total of Selling Car Depend of Transmission", y=1.02);
```

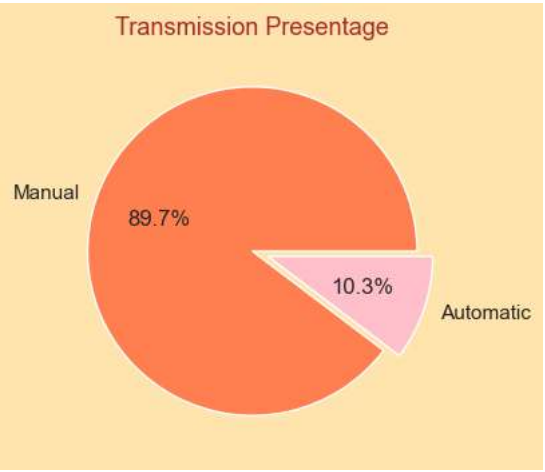


In [12]:
```
ax = car_df['owner'].value_counts().head(10).sort_values().plot(kind='barh', figsize=(8,3), cmap='Paired_r')
ax.bar_label(ax.containers[0], )
plt.xlabel("Count", labelpad=14)
plt.ylabel("Owner", labelpad=14)
plt.title("Total of Selling Car Depend of Owner", y=1.02);
```

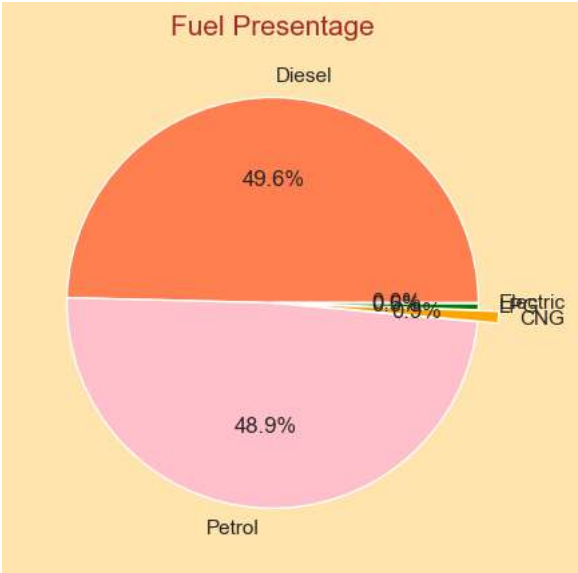
In [13]:
```
labels=car_df['owner'].value_counts().index
colors=['coral','pink','orange','green','yellow']
explode=[0,0.1,0,0,0]
values=car_df['owner'].value_counts().values

#visualization
plt.figure(figsize=(4,4))
plt.pie(values,explode=explode,labels=labels,colors=colors,autopct='%1.1f%%')
plt.title('Owner Presentage',color='brown',fontsize=15)
plt.show()
```



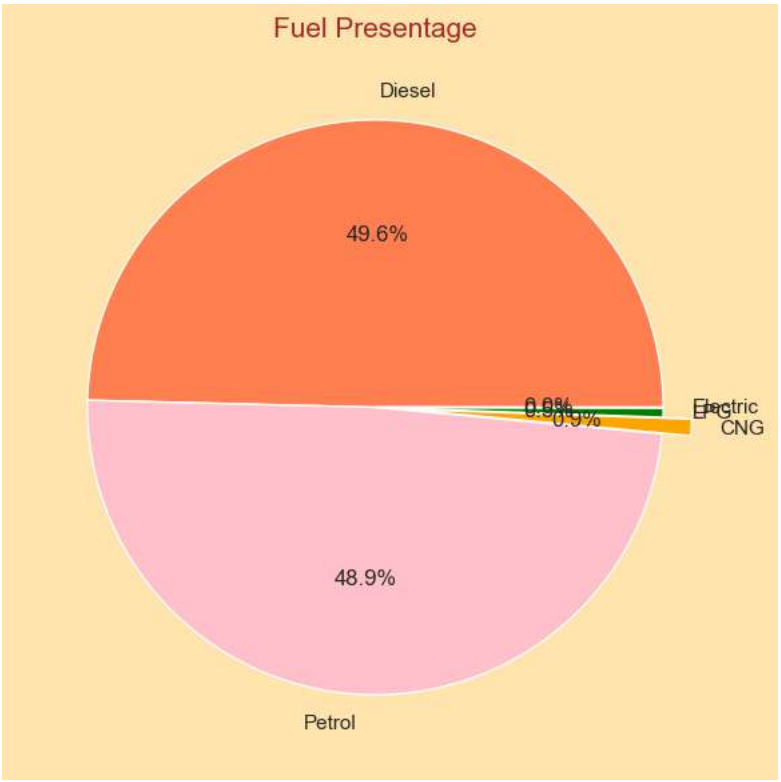
In [14]:
```
labels=car_df['transmission'].value_counts().index
colors=['coral','pink']
explode=[0,0.1]
values=car_df['transmission'].value_counts().values

#visualization
plt.figure(figsize=(4,4))
plt.pie(values,explode=explode,labels=labels,colors=colors,autopct='%1.1f%%')
plt.title('Transmission Presentage',color='brown',fontsize=13)
plt.show()
```



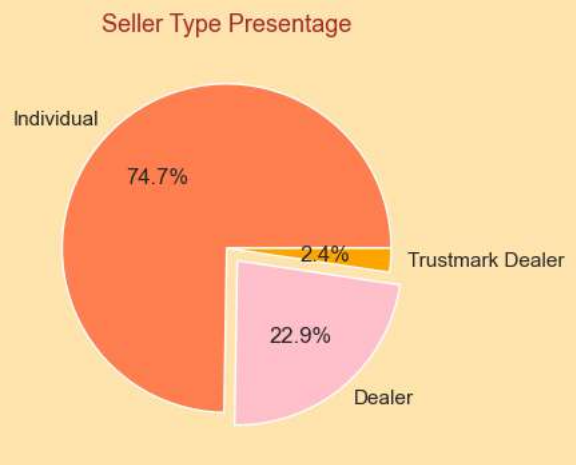
In [15]:
```
labels=car_df['fuel'].value_counts().index
colors=['coral','pink','orange','green','yellow']
explode=[0,0,0.1,0,0]
values=car_df['fuel'].value_counts().values

#visualization
plt.figure(figsize=(5,5))
plt.pie(values,explode=explode,labels=labels,colors=colors,autopct='%1.1f%%')
plt.title('Fuel Presentage',color='brown',fontsize=15)
plt.show()
```



In [16]:
```
labels=car_df['fuel'].value_counts().index
colors=['coral','pink','orange','green','yellow']
explode=[0,0,0.1,0,0]
values=car_df['fuel'].value_counts().values

#visualization
plt.figure(figsize=(7,7))
plt.pie(values,explode=explode,labels=labels,colors=colors,autopct='%1.1f%%')
plt.title('Fuel Presentage',color='brown',fontsize=15)
plt.show()
```

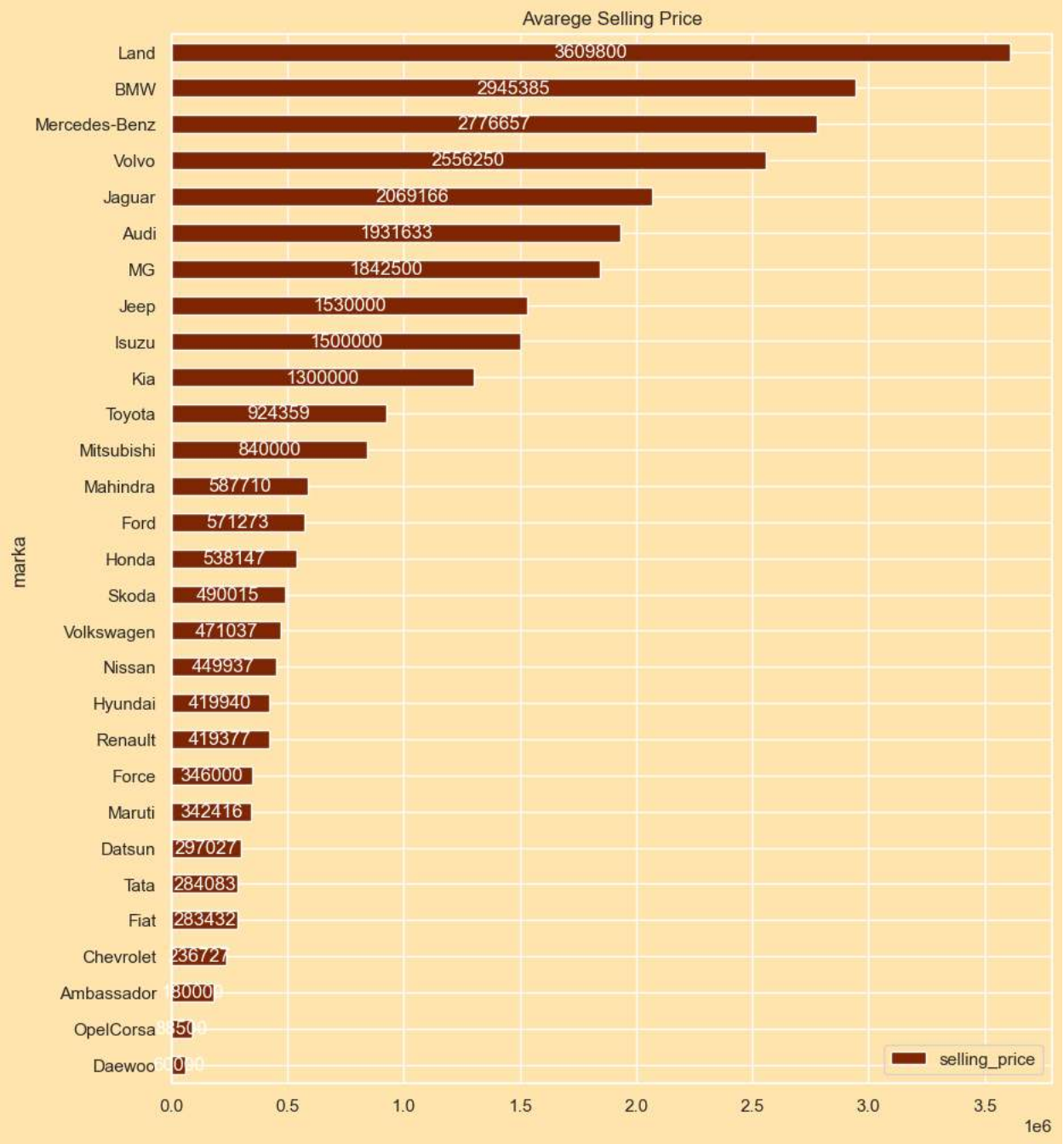


In [17]:
```
labels=car_df['seller_type'].value_counts().index
colors=['coral','pink', 'orange']
explode=[0,0.1,0]
values=car_df['seller_type'].value_counts().values

#visualization
plt.figure(figsize=(4,4))
plt.pie(values,explode=explode,labels=labels,colors=colors,autopct='%1.1f%%')
plt.title('Seller Type Presentage',color='brown',fontsize=13)
plt.show()
```



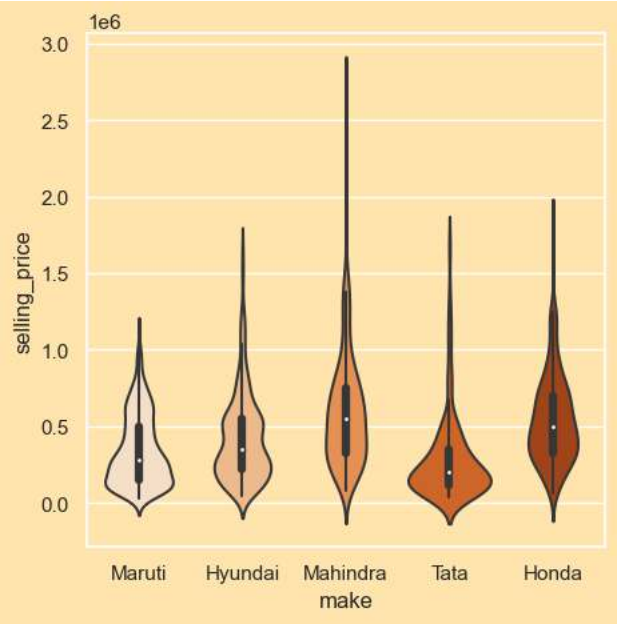
In [18]:
```
car_df['marka'] = car_df['name'].str.split(' ').str[0]
```

In [19]:
```
df_marka_price = car_df.groupby(['marka'])[['selling_price']].mean()
df_marka_price.sort_values(by='selling_price', ascending=True, inplace=True)
ax   = df_marka_price.plot(kind='barh', cmap='Oranges_r' , figsize=(10,12) ,title= 'Avarege Selling Price', )
for c in ax.containers:
        # set the bar label
    ax.bar_label(c, fmt='%.0f',label_type='center', color='w',rotation=0)
```

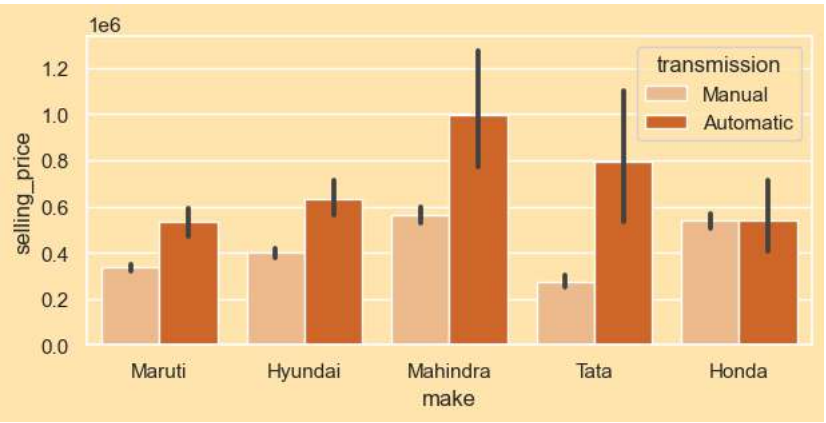


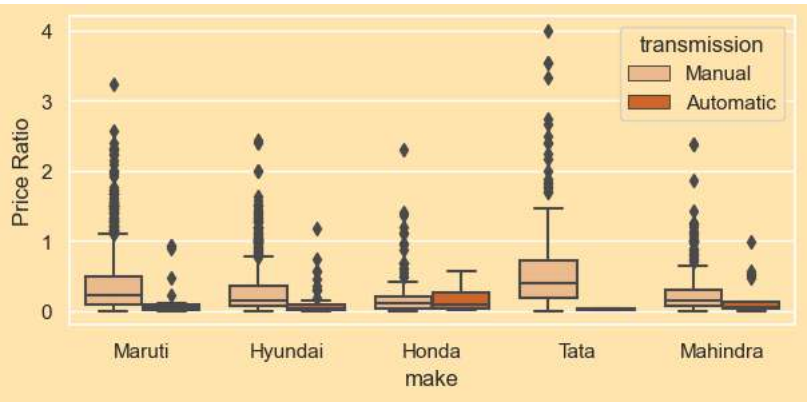
In [20]:
```
top_5 = ["Maruti", "Hyundai", "Mahindra", "Tata", "Honda"]
top_5_df = car_df[car_df.make.isin(top_5)]
fig, ax = plt.subplots(figsize = (5,5))
sns.violinplot(data = top_5_df, x = "make", y = "selling_price", order = top_5, ax = ax, palette='Oranges')
plt.show
```

Out[20]: <function matplotlib.pyplot.show(close=None, block=None)>

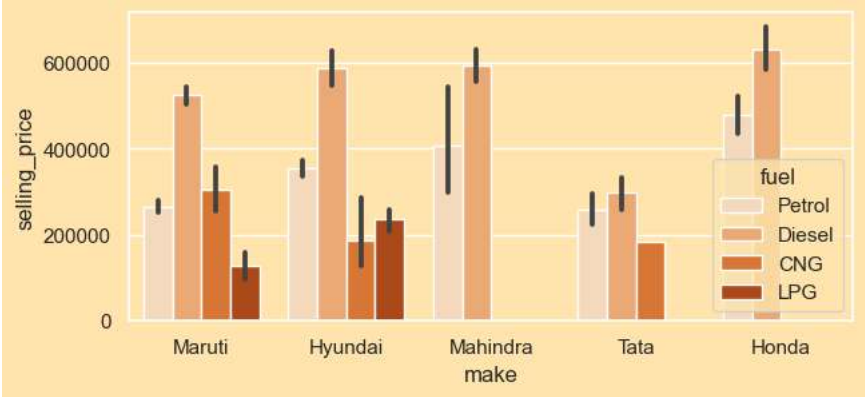


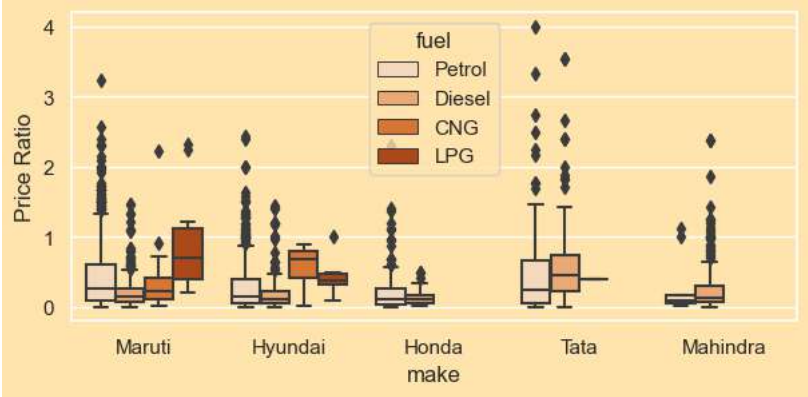
In [21]:
```
top_5 = ["Maruti", "Hyundai", "Mahindra", "Tata", "Honda"]
top_5_df = car_df[car_df.make.isin(top_5)]
fig, ax = plt.subplots(figsize = (7,3))
sns.barplot(data = top_5_df, x = "make", y = "selling_price", order = top_5, hue ="transmission", ax = ax, palette='Oranges')
plt.show

fig, ax = plt.subplots(figsize = (7,3))
car_df["Price Ratio"] = car_df.km_driven / car_df.selling_price
sns.boxplot(data = car_df[car_df.make.isin(top_5)], x = "make", y = "Price Ratio", hue ="transmission", ax = ax, palette='Oranges
plt.show()
```



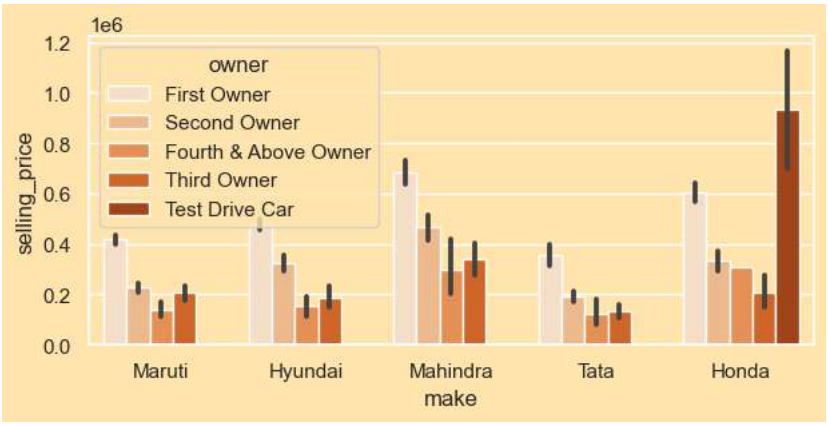


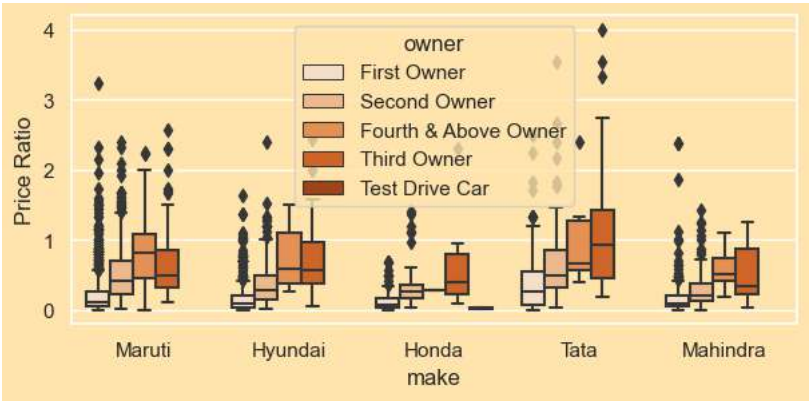
In [22]:
```
top_5 = ["Maruti", "Hyundai", "Mahindra", "Tata", "Honda"]
top_5_df = car_df[car_df.make.isin(top_5)]
fig, ax = plt.subplots(figsize = (7,3))
sns.barplot(data = top_5_df, x = "make", y = "selling_price", order = top_5, hue ="fuel", ax = ax, palette='Oranges')
plt.show
fig, ax = plt.subplots(figsize = (7,3))
car_df["Price Ratio"] = car_df.km_driven / car_df.selling_price
sns.boxplot(data = car_df[car_df.make.isin(top_5)], x = "make", y = "Price Ratio", hue ="fuel", ax = ax, palette='Oranges')
plt.show()
```



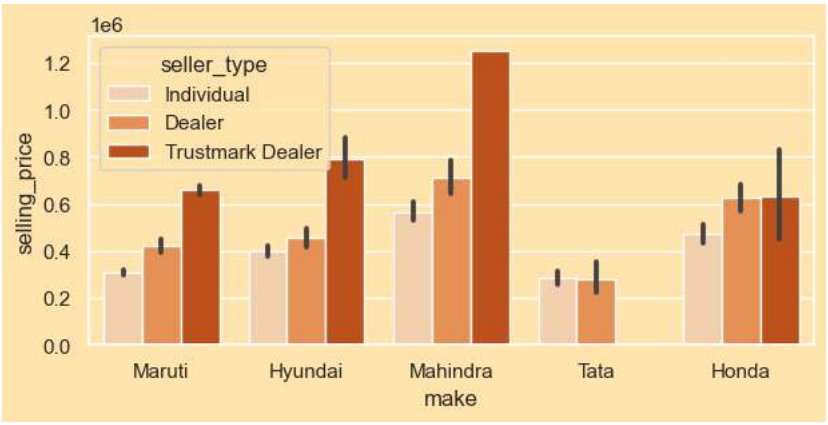


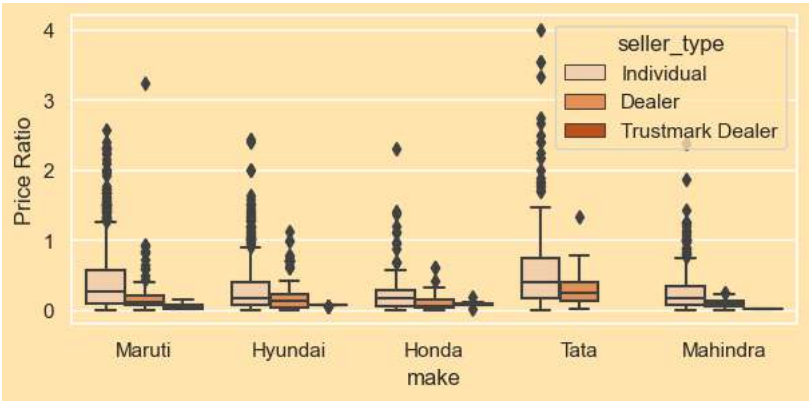
In [23]:
```
top_5 = ["Maruti", "Hyundai", "Mahindra", "Tata", "Honda"]
top_5_df = car_df[car_df.make.isin(top_5)]
fig, ax = plt.subplots(figsize = (7,3))
sns.barplot(data = top_5_df, x = "make", y = "selling_price", order = top_5, hue ="owner", ax = ax, palette='Oranges')
plt.show

fig, ax = plt.subplots(figsize = (7,3))
car_df["Price Ratio"] = car_df.km_driven / car_df.selling_price
sns.boxplot(data = car_df[car_df.make.isin(top_5)], x = "make", y = "Price Ratio", hue ="owner", ax = ax, palette='Oranges')
plt.show()
```





In [24]:
```
top_5 = ["Maruti", "Hyundai", "Mahindra", "Tata", "Honda"]
top_5_df = car_df[car_df.make.isin(top_5)]
fig, ax = plt.subplots(figsize = (7,3))
sns.barplot(data = top_5_df, x = "make", y = "selling_price", order = top_5, hue ="seller_type", ax = ax, palette='Oranges')
plt.show
fig, ax = plt.subplots(figsize = (7,3))
car_df["Price Ratio"] = car_df.km_driven / car_df.selling_price
sns.boxplot(data = car_df[car_df.make.isin(top_5)], x = "make", y = "Price Ratio", hue ="seller_type", ax = ax, palette='Oranges'
plt.show()
```



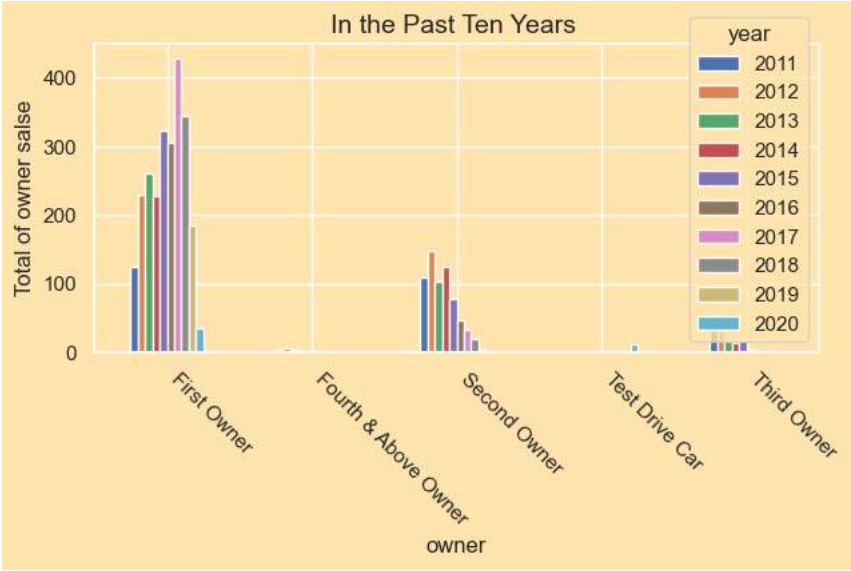


In [25]:
```
car_df.groupby("year")["owner"].value_counts().unstack().fillna(0).astype(
    "int"
).style.background_gradient(cmap="Oranges")
```

Out[25]:

owner	First Owner	Fourth & Above Owner	Second Owner	Test Drive Car	Third Owner
year					
1992	0	1	0	0	0
1995	0	0	1	0	0
1996	1	0	1	0	0
1997	1	0	1	0	1
1998	0	3	6	0	3
1999	1	3	5	0	1
2000	4	2	4	0	2
2001	5	1	6	0	8
2002	7	3	6	0	5
2003	7	3	11	0	2
2004	13	6	15	0	8
2005	33	2	36	0	14
2006	30	5	51	0	24
2007	41	9	66	0	18
2008	59	4	52	0	30
2009	77	10	74	0	32
2010	91	9	102	0	32
2011	124	4	110	0	33
2012	230	6	148	0	31
2013	260	5	103	0	18
2014	227	2	124	0	14
2015	324	1	78	0	18
2016	305	0	47	0	5
2017	428	1	34	1	2
2018	344	0	20	1	1
2019	185	0	5	3	2
2020	35	1	0	12	0

In [28]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
cmap = colors.ListedColormap(["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"])
df_owner = car_df.query('year > 2010')
df_owner = (
    df_owner.groupby("year")["owner"]
    .value_counts()
    .sort_index()
    .unstack()
)
ax = df_owner.T.plot(kind="bar", figsize=(7, 3))
ax.set_title("In the Past Ten Years ", fontsize=14)
ax.set_ylabel("Total of owner salse")
plt.xticks(rotation=-45, ha="left")
plt.show()
```



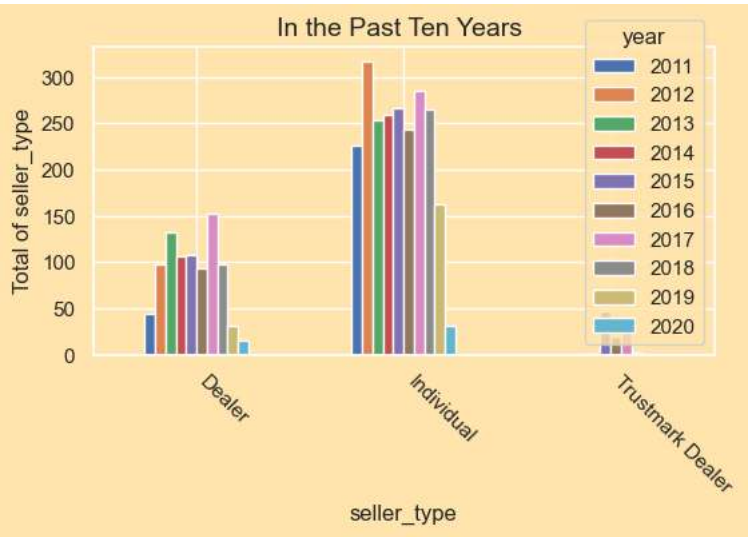
In [29]:
```
plt.figure(figsize=(7,3))
sns.lineplot(data=car_df, x="year", y="selling_price", hue="fuel", ci=False)
plt.title("Change in Price Over Years for Depend on Fuel Type")
```

Out[29]:
```
Text(0.5, 1.0, 'Change in Price Over Years for Depend on Fuel Type')
```

In [30]:
```
plt.figure(figsize=(7,3))
sns.lineplot(data=car_df, x="year", y="selling_price", hue="seller_type", ci=False)
plt.title("Change in Price Over Years Depend on Seller Typee")
```

Out[30]:
```
Text(0.5, 1.0, 'Change in Price Over Years Depend on Seller Typee')
```

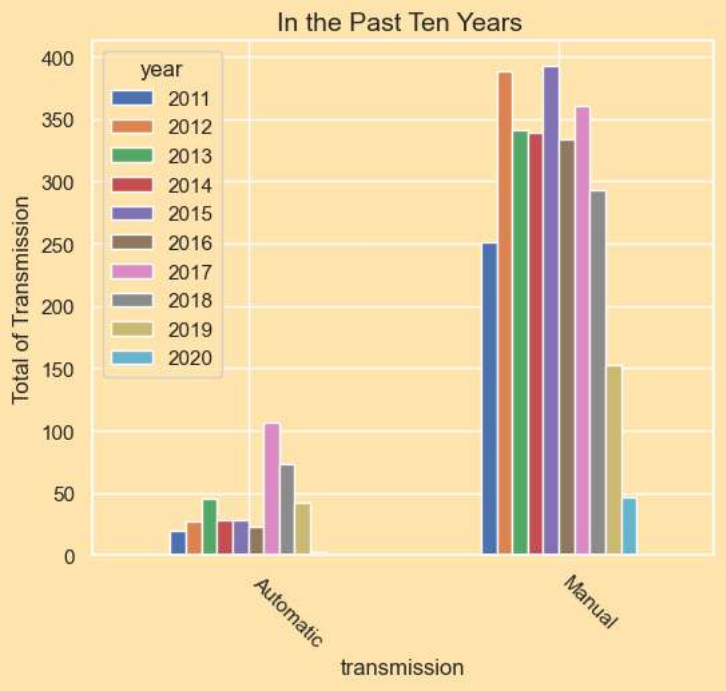
In [31]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
cmap = colors.ListedColormap(["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"])
df_owner = car_df.query('year > 2010')
df_owner = (
    df_owner.groupby("year")["seller_type"]
    .value_counts()
    .sort_index()
    .unstack()
)
ax = df_owner.T.plot(kind="bar", figsize=(6, 3))
ax.set_title("In the Past Ten Years ", fontsize=14)
ax.set_ylabel("Total of seller_type")
plt.xticks(rotation=-45, ha="left")
plt.show()
```



In [32]:
```
plt.figure(figsize=(7,3))
sns.lineplot(data=car_df, x="year", y="selling_price", hue="transmission", ci=False)
plt.title("Change in Price Over Years depend on Transmission Type")
```

Out[32]:
```
Text(0.5, 1.0, 'Change in Price Over Years depend on Transmission Type')
```

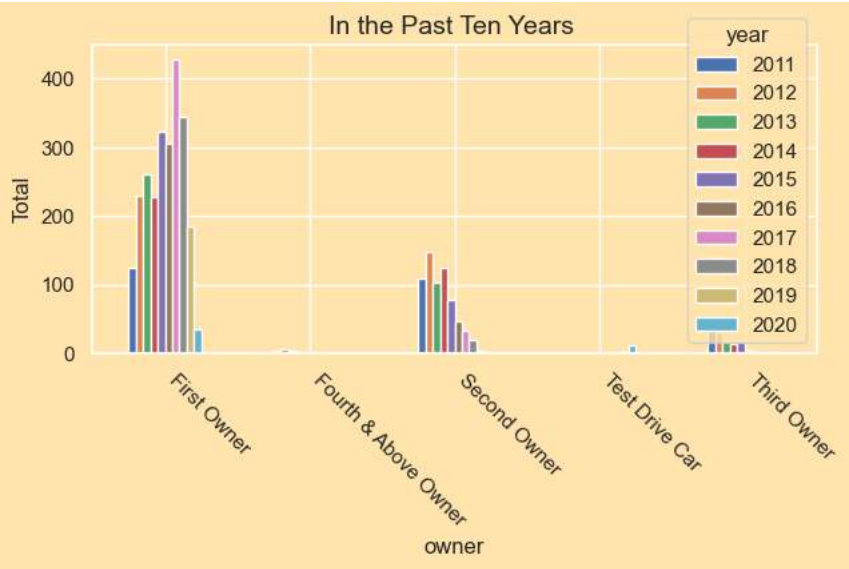
In [33]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
cmap = colors.ListedColormap(["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"])
df_owner = car_df.query('year > 2010')
df_owner = (
    df_owner.groupby("year")["transmission"]
    .value_counts()
    .sort_index()
    .unstack()
)
ax = df_owner.T.plot(kind="bar", figsize=(6, 5))
ax.set_title("In the Past Ten Years ", fontsize=14)
ax.set_ylabel("Total of Transmission")
plt.xticks(rotation=-45, ha="left")
plt.show()
```



In [34]:
```
plt.figure(figsize=(7,3))
sns.lineplot(data=car_df, x="year", y="selling_price", hue="owner", ci=False)
plt.title("change in price over years Depend on Owner")
```

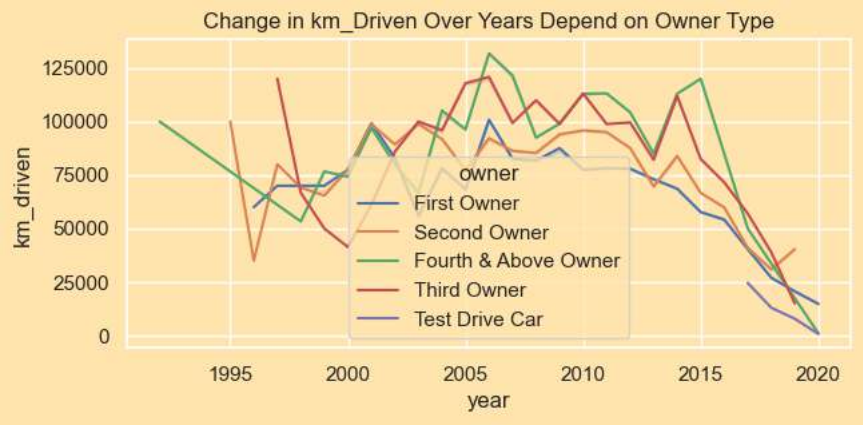
Out[34]:
```
Text(0.5, 1.0, 'change in price over years Depend on Owner')
```

In [35]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
cmap = colors.ListedColormap(["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"])
df_owner = car_df.query('year > 2010')
df_owner = (
    df_owner.groupby("year")["owner"]
    .value_counts()
    .sort_index()
    .unstack()
)
ax = df_owner.T.plot(kind="bar", figsize=(7, 3))
ax.set_title("In the Past Ten Years ", fontsize=14)
ax.set_ylabel("Total")
plt.xticks(rotation=-45, ha="left")
plt.show()
```



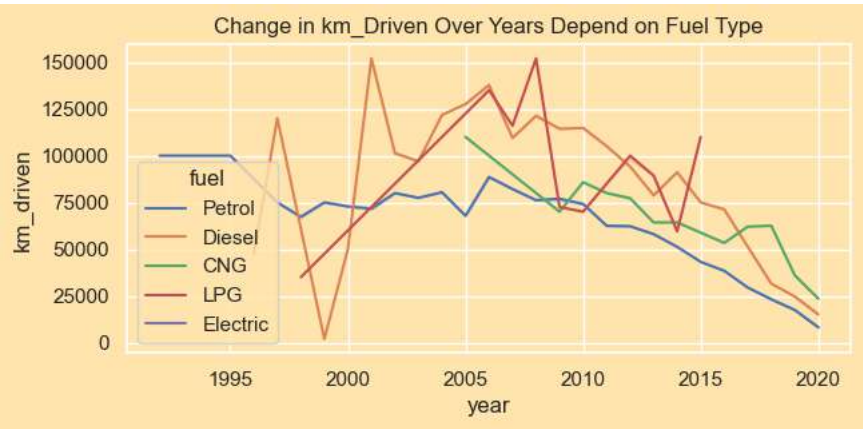
In [36]:
```
plt.figure(figsize=(7,3))
sns.lineplot(data=car_df, x="year", y="km_driven", hue="owner", ci=False)
plt.title("Change in km_Driven Over Years Depend on Owner Type")
```

Out[36]:
```
Text(0.5, 1.0, 'Change in km_Driven Over Years Depend on Owner Type')
```



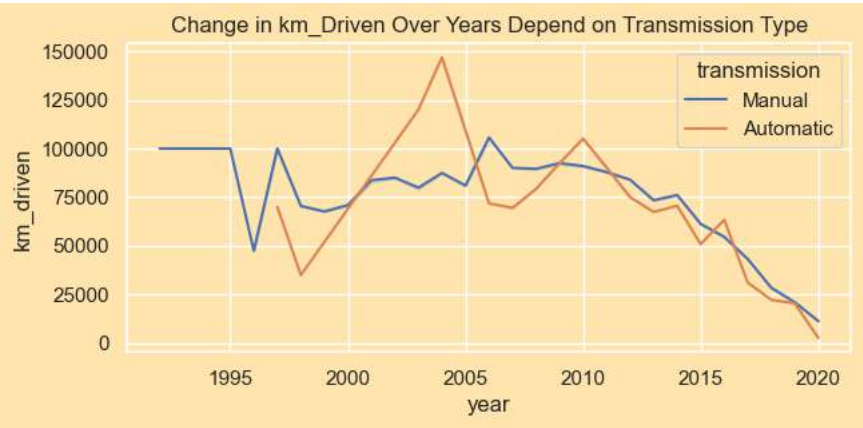
In [37]:
```
plt.figure(figsize=(7,3))
sns.lineplot(data=car_df, x="year", y="km_driven", hue="fuel", ci=False)
plt.title("Change in km_Driven Over Years Depend on Fuel Type")
```

Out[37]:
```
Text(0.5, 1.0, 'Change in km_Driven Over Years Depend on Fuel Type')
```



In [38]:
```
plt.figure(figsize=(7,3))
sns.lineplot(data=car_df, x="year", y="km_driven", hue="transmission", ci=False)
plt.title("Change in km_Driven Over Years Depend on Transmission Type")
```

Out[38]:
```
Text(0.5, 1.0, 'Change in km_Driven Over Years Depend on Transmission Type')
```



In [39]:
```
import numpy as np
import pandas as pd
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [40]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.linear_model import SGDRegressor, LinearRegression
from sklearn.metrics import mean_absolute_error
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingRegressor,RandomForestRegressor
sns.set_style("whitegrid")
```

In [41]:
```
car_df.dtypes
```

Out[41]:
```
name              object
year               int64
selling_price      int64
km_driven          int64
fuel              object
seller_type       object
transmission      object
owner             object
make              object
marka             object
Price Ratio      float64
dtype: object
```

In [42]:
```
car_df.duplicated().sum()
```

Out[42]:
```
763
```

In [43]:
```
cars_data = car_df.drop_duplicates()
cars_data.duplicated().sum()
```

Out[43]:
```
0
```

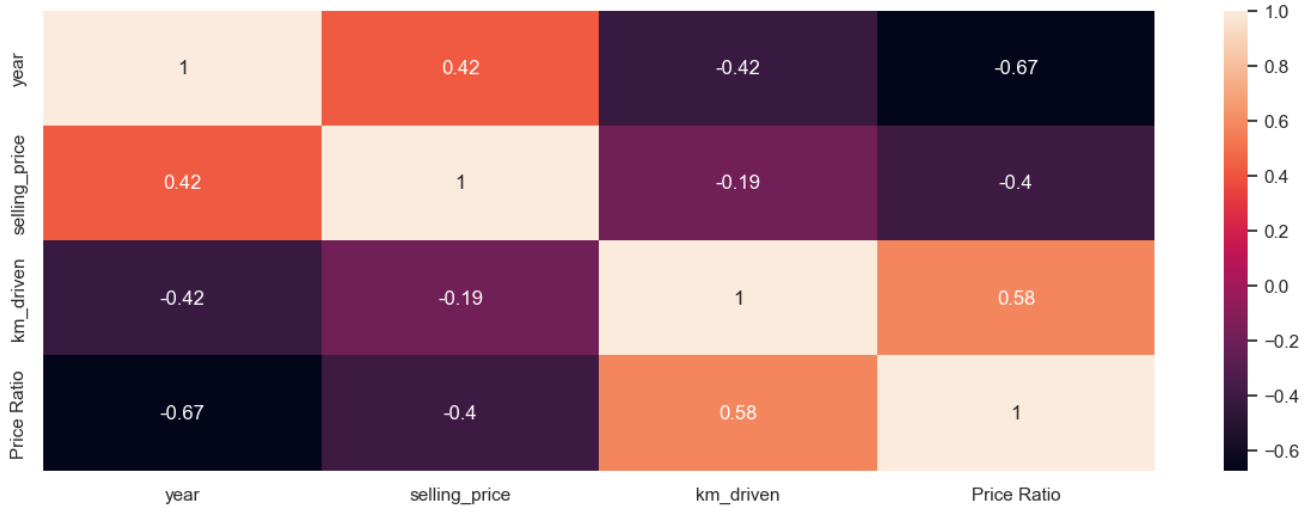
In [44]:
```
cars_data.describe().T
```

Out[44]:

	count	mean	std	min	25%	50%	75%	max
year	3577.0	2012.962538	4.251759	1992.000000	2010.000000	2013.00	2016.0	2020.0
selling_price	3577.0	473912.542074	509301.809816	20000.000000	200000.000000	350000.00	600000.0	8900000.0
km_driven	3577.0	69250.545709	47579.940016	1.000000	36000.000000	60000.00	90000.0	806599.0
Price Ratio	3577.0	0.316898	0.394270	0.000004	0.072165	0.18	0.4	4.0

In [45]:
```
corr = cars_data.corr(method="pearson").T

plt.figure(figsize=(15,5))
sns.heatmap(corr, annot=True)
```

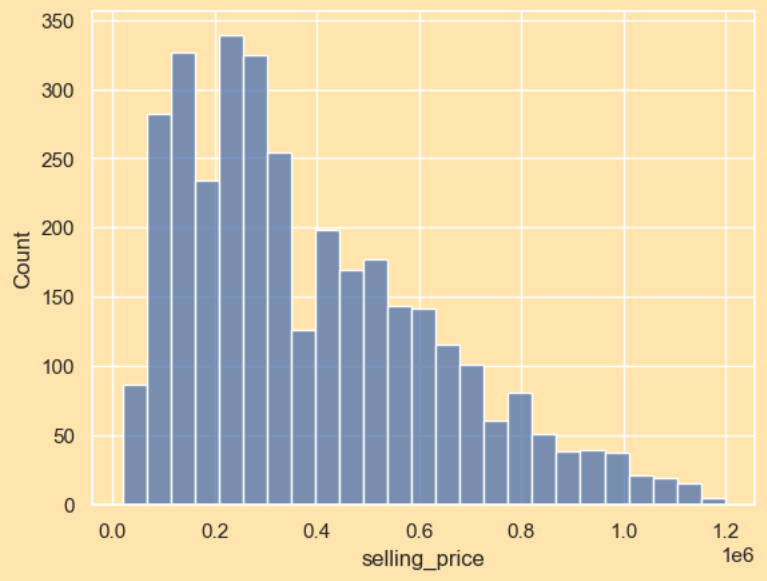
Out[45]:
```
<AxesSubplot:>
```



exploratory data analysis

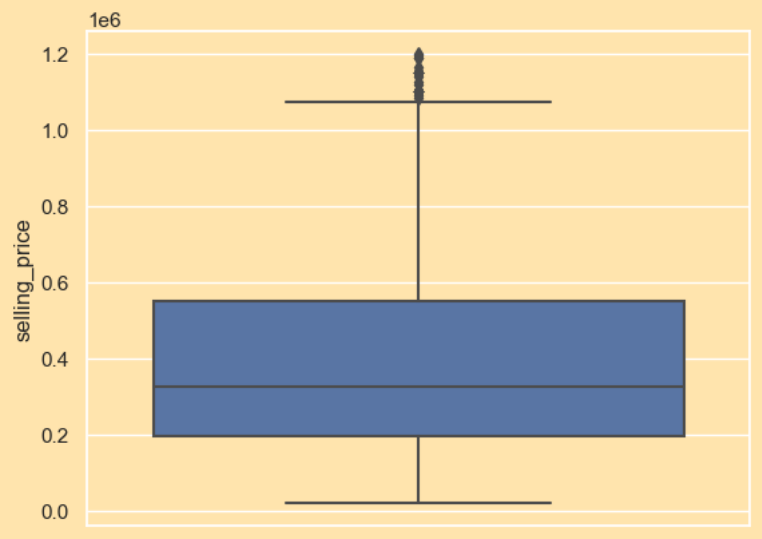
In [59]:
```
sns.set(rc={"axes.facecolor":"#ffe4ad","figure.facecolor":"#ffe4ad"})
pallet = ["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"]
cmap = colors.ListedColormap(["#682F2F", "#9E726F", "#D6B2B1", "#B9C0C9", "#9F8A78", "#F3AB60"])
sns.histplot(data=cars_data, x="selling_price")
```

Out[59]: <AxesSubplot:xlabel='selling_price', ylabel='Count'>



In [60]:
```
sns.boxplot(data=cars_data, y="selling_price")
```

Out[60]: <AxesSubplot:ylabel='selling_price'>

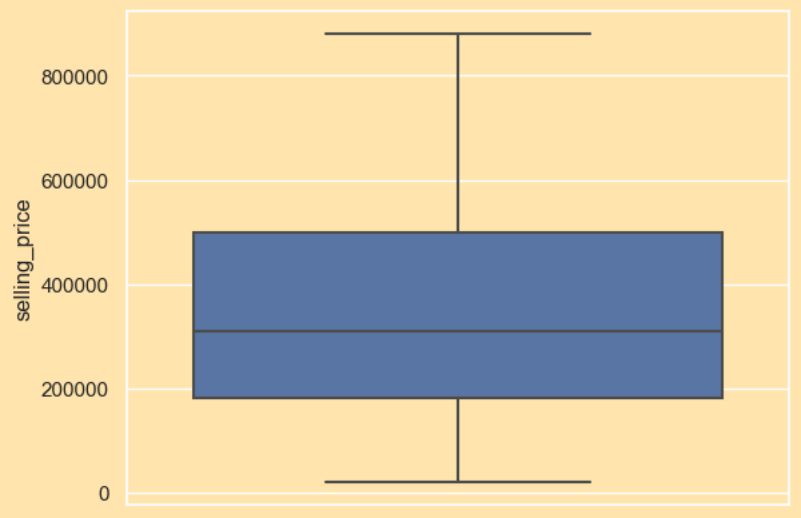


Remove Outliars

In [61]:
```
q = cars_data["selling_price"].quantile(0.95)
cars_data = cars_data[cars_data['selling_price'] < q]
```

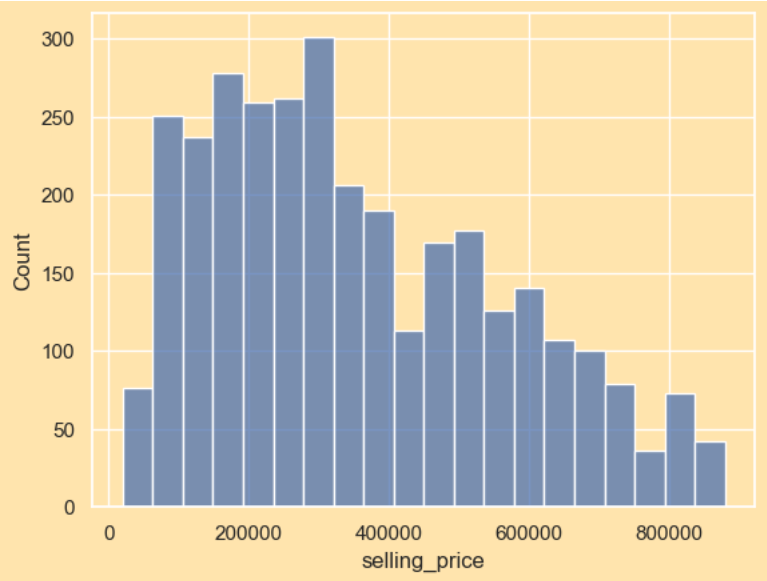
In [62]:
```
sns.boxplot(data=cars_data, y="selling_price")
```

Out[62]:
```
<AxesSubplot:ylabel='selling_price'>
```



In [63]:
```
sns.histplot(data=cars_data, x="selling_price")
```

Out[63]:
```
<AxesSubplot:xlabel='selling_price', ylabel='Count'>
```



Data Analysis

In [64]:
```
cars_data = cars_data.sort_values("selling_price", ascending=False)
```

In [65]:
```
cars_data.columns
```

Out[65]:
```
Index(['year', 'selling_price', 'km_driven', 'Price Ratio',
       'name_Ambassador CLASSIC 1500 DSL AC',
       'name_Ambassador Classic 2000 Dsz',
       'name_Ambassador Grand 1800 ISZ MPFI PW CL',
       'name_Audi A4 2.0 TDI 177 Bhp Premium Plus', 'name_Audi A6 2.7 TDI',
       'name_Audi A6 2.8 FSI',
       ...
       'marka_Maruti', 'marka_Mercedes-Benz', 'marka_Mitsubishi',
       'marka_Nissan', 'marka_OpelCorsa', 'marka_Renault', 'marka_Skoda',
       'marka_Tata', 'marka_Toyota', 'marka_Volkswagen'],
      dtype='object', length=1437)
```

In [71]:
```
cars_data = pd.get_dummies(cars_data)
```

In [72]:
```
features = cars_data.drop(['selling_price'], axis=1)
target = cars_data['selling_price']


X_train, X_test, y_train, y_test = train_test_split(features, target,
                                                    test_size=0.2,
                                                    random_state=42)
```

In [73]:
```
scaler = StandardScaler()
X_train= scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [74]:
```
models = [RandomForestRegressor(n_estimators=200, max_depth=8, min_samples_leaf=2, random_state=42),
          SGDRegressor(alpha=0.001, random_state=42),
          GradientBoostingRegressor(n_estimators=200, max_depth=8, learning_rate=0.01, random_state=42)]
maes = []

for model in models:
    model = model.fit(X_train, y_train)
    y_hat = model.predict(X_test)
    mae = mean_absolute_error(y_test, y_hat)
    maes.append(mae)
```

In [75]:
```
model_mae = pd.DataFrame(data=[models, maes], columns=["RF", "SGD", "GB"])
model_mae.loc[1, :]
```

Out[75]:
```
RF         21679.125328
SGD    470151623.572492
GB         33098.556046
Name: 1, dtype: object
```