In [64]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.model_selection import GridSearchCV
```

In [66]:
```
# Load dataset
url = "CarPrice_Assignment.csv"
df = pd.read_csv(url)
```

In [68]:
```
# 1. Preprocessing
# Check for missing values
print(df.isnull().sum())
```

```
car_ID              0
symboling           0
CarName             0
fueltype            0
aspiration          0
doornumber          0
carbody             0
drivewheel          0
enginelocation      0
wheelbase           0
carlength           0
carwidth            0
carheight           0
curbweight          0
enginetype          0
cylindernumber      0
enginesize          0
fuelsystem          0
boreratio           0
stroke              0
compressionratio    0
horsepower          0
peakrpm             0
citympg             0
highwaympg          0
price               0
dtype: int64
```

In [70]:
```
# Handle missing data
# Filling missing numerical values with the mean
df.fillna(df.mean(), inplace=True)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[70], line 3
      1 # Handle missing data
      2 # Filling missing numerical values with the mean
----> 3 df.fillna(df.mean(), inplace=True)

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:11693, in DataFrame.mean
(self, axis, skipna, numeric_only, **kwargs)
  11685 @doc(make_doc("mean", ndim=2))
  11686 def mean(
  11687     self,
   (...)
  11691     **kwargs,
  11692 ):
> 11693     result = super().mean(axis, skipna, numeric_only, **kwargs)
  11694     if isinstance(result, Series):
  11695         result = result.__finalize__(self, method="mean")

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:12420, in NDFrame.mean
(self, axis, skipna, numeric_only, **kwargs)
  12413 def mean(
  12414     self,
  12415     axis: Axis | None = 0,
   (...)
  12418     **kwargs,
  12419 ) -> Series | float:
> 12420     return self._stat_function(
  12421         "mean", nanops.nanmean, axis, skipna, numeric_only, **kwargs
  12422     )

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:12377, in NDFrame._stat
_function(self, name, func, axis, skipna, numeric_only, **kwargs)
  12373 nv.validate_func(name, (), kwargs)
  12375 validate_bool_kwarg(skipna, "skipna", none_allowed=False)
> 12377 return self._reduce(
  12378     func, name=name, axis=axis, skipna=skipna, numeric_only=numeric_only
  12379 )

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:11562, in DataFrame._redu
ce(self, op, name, axis, skipna, numeric_only, filter_type, **kwds)
  11558     df = df.T
  11560 # After possibly _get_data and transposing, we are now in the
  11561 #  simple case where we can use BlockManager.reduce
> 11562 res = df._mgr.reduce(blk_func)
  11563 out = df._constructor_from_mgr(res, axes=res.axes).iloc[0]
  11564 if out_dtype is not None and out.dtype != "boolean":

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:1500, in Blo
ckManager.reduce(self, func)
   1498 res_blocks: list[Block] = []
   1499 for blk in self.blocks:
-> 1500     nbs = blk.reduce(func)
   1501     res_blocks.extend(nbs)
   1503 index = Index([None])  # placeholder

File ~\anaconda3\Lib\site-packages\pandas\core\internals\blocks.py:404, in Block.
reduce(self, func)
    398 @final
    399 def reduce(self, func) -> list[Block]:
```

```
    400     # We will apply the function and reshape the result into a single-row
    401     #  Block with the same mgr_locs; squeezing will be done at a higher l
evel
    402     assert self.ndim == 2
--> 404     result = func(self.values)
    406     if self.values.ndim == 1:
    407         res_values = result

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:11481, in DataFrame._redu
ce.<locals>.blk_func(values, axis)
  11479         return np.array([result])
  11480 else:
> 11481     return op(values, axis=axis, skipna=skipna, **kwds)

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:147, in bottleneck_switc
h.__call__.<locals>.f(values, axis, skipna, **kwds)
    145         result = alt(values, axis=axis, skipna=skipna, **kwds)
    146 else:
--> 147     result = alt(values, axis=axis, skipna=skipna, **kwds)
    149 return result

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:404, in _datetimelike_co
mpat.<locals>.new_func(values, axis, skipna, mask, **kwargs)
    401 if datetimelike and mask is None:
    402     mask = isna(values)
--> 404 result = func(values, axis=axis, skipna=skipna, mask=mask, **kwargs)
    406 if datetimelike:
    407     result = _wrap_results(result, orig_values.dtype, fill_value=iNaT)

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:720, in nanmean(values,
axis, skipna, mask)
    718 count = _get_counts(values.shape, mask, axis, dtype=dtype_count)
    719 the_sum = values.sum(axis, dtype=dtype_sum)
--> 720 the_sum = _ensure_numeric(the_sum)
    722 if axis is not None and getattr(the_sum, "ndim", False):
    723     count = cast(np.ndarray, count)

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:1686, in _ensure_numeric
(x)
   1683 inferred = lib.infer_dtype(x)
   1684 if inferred in ["string", "mixed"]:
   1685     # GH#44008, GH#36703 avoid casting e.g. strings to numeric
-> 1686     raise TypeError(f"Could not convert {x} to numeric")
   1687 try:
   1688     x = x.astype(np.complex128)

TypeError: Could not convert ['alfa-romero giuliaalfa-romero stelvioalfa-romero Q
uadrifoglioaudi 100 lsaudi 100lsaudi foxaudi 100lsaudi 5000audi 4000audi 5000s (d
iesel)bmw 320ibmw 320ibmw x1bmw x3bmw z4bmw x4bmw x5bmw x3chevrolet impalachevrol
et monte carlochevrolet vega 2300dodge rampagedodge challenger sedodge d200dodge
monaco (sw)dodge colt hardtopdodge colt (sw)dodge coronet customdodge dart custom
dodge coronet custom (sw)honda civichonda civic cvcchonda civichonda accord cvcch
onda civic cvcchonda accord lxhonda civic 1500 glhonda accordhonda civic 1300hond
a preludehonda accordhonda civichonda civic (auto)isuzu MU-Xisuzu D-Max isuzu D-M
ax V-Crossisuzu D-Max jaguar xjjaguar xfjaguar xkmaxda rx3maxda glc deluxemazda r
x2 coupemazda rx-4mazda glc deluxemazda 626mazda glcmazda rx-7 gsmazda glc 4mazda
626mazda glc custom lmazda glc custommazda rx-4mazda glc deluxemazda 626mazda glc
mazda rx-7 gsbuick electra 225 custombuick century luxus (sw)buick centurybuick s
kyhawkbuick opel isuzu deluxebuick skylarkbuick century specialbuick regal sport
coupe (turbo)mercury cougarmitsubishi miragemitsubishi lancermitsubishi outlander
```

```
mitsubishi g4mitsubishi mirage g4mitsubishi g4mitsubishi outlandermitsubishi g4mi
tsubishi mirage g4mitsubishi monteromitsubishi pajeromitsubishi outlandermitsubis
hi mirage g4Nissan versanissan gt-rnissan roguenissan lationissan titannissan lea
fnissan jukenissan lationissan notenissan clippernissan roguenissan nv200nissan d
ayznissan fuganissan ottinissan teananissan kicksnissan clipperpeugeot 504peugeot
304peugeot 504 (sw)peugeot 504peugeot 504peugeot 604slpeugeot 504peugeot 505s tur
bo dieselpeugeot 504peugeot 504peugeot 604slplymouth fury iiiplymouth cricketplym
outh fury iiiplymouth satellite custom (sw)plymouth fury gran sedanplymouth valia
ntplymouth dusterporsche macanporcshce panameraporsche cayenneporsche boxterporsc
he cayennerenault 12tlrenault 5 gtlsaab 99esaab 99lesaab 99lesaab 99glesaab 99gle
saab 99esubarusubaru dlsubaru dlsubarusubaru brzsubaru bajasubaru r1subaru r2suba
ru treziasubaru tribecasubaru dlsubaru dltoyota corona mark iitoyota coronatoyota
corolla 1200toyota corona hardtoptoyota corolla 1600 (sw)toyota carinatoyota mark
iitoyota corolla 1200toyota coronatoyota corollatoyota coronatoyota corollatoyota
mark iitoyota corolla liftbacktoyota coronatoyota celica gt liftbacktoyota coroll
a terceltoyota corona liftbacktoyota corollatoyota starlettoyota terceltoyota cor
ollatoyota cressidatoyota corollatoyota celica gttoyota coronatoyota corollatoyot
a mark iitoyota corolla liftbacktoyota coronatoyota starlettoyouta tercelvokswage
n rabbitvolkswagen 1131 deluxe sedanvolkswagen model 111volkswagen type 3volkswag
en 411 (sw)volkswagen super beetlevolkswagen dashervw dashervw rabbitvolkswagen r
abbitvolkswagen rabbit customvolkswagen dashervolvo 145e (sw)volvo 144eavolvo 244
dlvolvo 245volvo 264glvolvo dieselvolvo 145e (sw)volvo 144eavolvo 244dlvolvo 246v
olvo 264gl'
 'gasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasg
asgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasg
asgasgasgasgasgasgasgasgasgasdieselgasgasdieseldieseldieseldieseldieselgasgasgasg
asgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasdieselgasgasgasgasgasgasgasgasgasg
asgasgasgasgasgasgasgasdieselgasdieselgasdieselgasdieselgasdieselgasgasgasgasgasg
asgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasgasg
asgasgasgasgasgasgasgasgasdieseldieselgasgasgasgasgasgasgasgasgasgasgasgasgasgasd
ieselgasgasgasgasgasgasgasdieselgasdieselgasgasdieselgasgasgasgasdieselgasgasgasg
asgasgasgasgasgasgasdieselgas'
 'stdstdstdstdstdstdstdstdstdturboturbostdstdstdstdstdstdstdstdstdstdstdstdturbos
tdstdstdturbostdturbostdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstd
stdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdturboturboturboturbostdstdstds
tdturbostdstdstdturboturbostdturboturboturbostdstdturbostdstdstdstdstdstdstdstdst
dstdstdstdstdstdstdstdstdturbostdstdturbostdturbostdturbostdturbostdturboturbostd
turbostdstdstdstdturbostdstdstdstdstdstdstdstdstdstdturboturbostdstdstdstdstds
tdstdturbostdstdstdturbostdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstdstd
stdstdstdstdstdturbostdstdstdstdstdstdstdstdstdstdturbostdstdstdstdturbostd
stdstdstdstdturboturbostdturbostdturboturbo'
 'twotwotwofourfourtwofourfourfourtwotwofourtwofourfourfourtwofourtwotwofourtwotw
otwofourfourfourtwofourtwotwotwotwotwotwofourfourtwotwofourfourfourtwofourtwofour
twofourfourtwotwotwotwofourfourtwotwotwotwotwofourtwofourfourfourfourfourfourfour
twofourfourtwofourtwotwotwotwotwotwotwotwotwotwofourfourfourfourtwotwotwofourf
ourtwotwofourfourtwofourfourfourfourfourtwotwotwofourfourfourfourfourfourfourfour
fourfourfourtwotwofourfourfourfourtwotwotwotwotwotwofourtwotwofourtwofourtwofourt
wotwotwofourfourfourfourfourfourfourfourfourtwotwofourfourfourfourfourfourfourfou
rfourfourfourtwotwotwotwotwotwotwotwotwofourfourfourfourfourtwotwofourfourtwot
wofourfourfourfourfourtwotwofourfourfourfourfourfourfourfourfourfourfourfourfourf
our'
 'convertibleconvertiblehatchbacksedansedansedansedanwagonsedanhatchbacksedanseda
nsedansedansedansedansedanhatchbackhatchbacksedanhatchbackhatchbackhatchback
hatchbacksedansedansedanwagonhatchbackhatchbackhatchbackhatchbackhatchbackhatchba
cksedanwagonhatchbackhatchbacksedansedansedansedansedansedanhatchbacksedanse
dansedanhatchbackhatchbackhatchbacksedansedanhatchbackhatchbackhatchbackhatchback
hatchbacksedanhatchbacksedansedanhatchbacksedansedansedanwagonhardtopsedansedanco
nvertiblesedanhardtophatchbackhatchbackhatchbackhatchbackhatchbackhatchbackhatchb
ackhatchbackhatchbackhatchbacksedansedansedansedansedansedansedanwagonsedanh
atchbacksedanwagonhardtophatchbacksedansedanwagonsedanhatchbackhatchbackhatchback
```

```
sedansedanwagonwagonsedansedanwagonwagonsedansedansedanhatchbackhatchbackhatchbac
ksedansedanwagonhatchbackhatchbackhardtophardtopconvertiblehatchbackwagonhatchbac
khatchbacksedanhatchbacksedanhatchbacksedanhatchbackhatchbackhatchbacksedansedans
edansedansedanwagonwagonwagonwagonhatchbackhatchbackhatchbackwagonwagonwagonsedan
hatchbacksedanhatchbacksedanhatchbacksedansedanhatchbacksedanhatchbackhardtophard
tophatchbackhardtophatchbackconvertiblesedansedanhatchbacksedanhatchbackhatchback
hatchbacksedanwagonsedansedansedansedansedansedansedanconvertiblehatchbacksedanse
danwagonsedanwagonsedanwagonsedanwagonsedansedansedansedansedan'
 'rwdrwdrwdfwd4wdfwdfwdfwdfwd4wdrwdrwdrwdrwdrwdrwdrwdrwdfwdfwdfwdfwdfwdfwdfwdfwdf
wdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdrwdfwdfwdrwdrwdrwdrwdfwdfwdfwdf
wdfwdrwdrwdrwdrwdfwdfwdfwdfwdfwdfwdrwdrwdrwdrwdrwdrwdrwdrwdrwdrwdrwdfwdfwdfwdfwdf
wdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdfwdrwdrwdrwdr
wdrwdrwdrwdrwdrwdrwdrwdrwdrwdfwdfwdfwdfwdfwdfwdrwdrwdrwdrwdrwdrwdfwdfwdfwdfwdf
wdfwdfwdfwdfwdfwd4wdfwdfwdfwd4wd4wdfwdfwd4wd4wdfwdfwdfwd4wd4wdfwdfwdfwdfwdfwdf
wdfwdrwdrwdrwdrwdrwdrwdrwdrwdrwdfwdfwdfwdfwdfwdrwdrwdrwdrwdfwdfwdfwdfwdfwdf
wdfwdfwdfwdfwdfwdrwdrwdrwdrwdrwdrwdrwdrwdrwdrwd'
 'frontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfron
tfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfront
frontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontf
rontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfr
ontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfro
ntfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfron
tfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfront
frontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontrearrearrearfron
tfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfront
frontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontf
rontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfr
ontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfrontfro
ntfrontfrontfrontfrontfrontfrontfrontfrontfrontfront'
 'dohcdohcohcvohcohcohcohcohcohcohcohcohcohcohcohcohcohcohclohcohcohcohcohcohcohc
ohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcdohcdohcohcvohcohc
ohcohcohcrotorrotorrotorrotorohcohcohcohcohcohcohcohcohcohcohcohcohcvohcvohcvohcv
ohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohc
vohcvohcvohcvohcvohcvllllllllllohcohcohcohcohcohcohcohcfohcfohcfdohcvohcohcoh
cohcohcohcdohcdohcohcfohcfohcfohcfohcfohcfohcfohcfohcfohcfohcfohcfohcohcohcohcohc
ohcohcohcohcohcohcohcohcohcohcdohcdohcohcohcohcohcohcohcohcohcohcohcohcdohcdohcdo
hcdohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcohcvohcohc'
 'fourfoursixfourfivefivefivefivefivefivefourfoursixsixsixsixsixsixthreefourfourf
ourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfo
urfourfourfourfourfoursixsixtwelvefourfourfourfourfourtwotwotwotwofourfourfourfou
rfourfourfourfourfivefivefivefiveeighteighteighteightfourfourfourfourfourfourfour
fourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfoursixsi
xsixsixsixsixfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfour
fourfoursixsixsixeightfourfourfourfourfourfourfourfourfourfourfourfourfourfourfou
rfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfourfour
fourfourfourfourfourfourfourfourfourfourfourfourfoursixsixsixsixfourfourfourfourf
ourfourfourfourfourfivefourfourfourfourfourfourfourfourfourfoursixsixfour'
 'mpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfi2bbl2bb
l2bbl2bbl2bblmpfi2bbl2bbl2bblmpfi2bblmfi1bbl1bbl1bbl1bbl1bbl1bbl1bbl1bbl1bbl1bbl1
bblmpfi2bbl2bbl2bbl2bblspfimpfimpfimpfi2bbl2bbl2bbl2bbl2bbl4bbl4bbl4bblmpfi2bbl2b
bl2bbl2bblidi2bblmpfiidiidiidiidiidimpfimpfimpfimpfimpfi2bbl2bbl2bblspdispdi2bbls
pdispdispdi2bbl2bblspdispdi2bblidi2bbl2bbl2bbl2bbl2bbl2bbl2bbl2bbl2bbl2bblmpfimpf
impfimpfimpfimpfimpfiidimpfiidimpfiidimpfiidimpfiidimpfi2bblspdi2bbl2bbl2bbl2bbls
pdimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfi2bbl2bbl2bbl2bbl2bblmpfi2b
blmpfi2bblmpfi2bblmpfi2bbl2bbl2bbl2bbl2bbl2bbl2bbl2bblidiidi2bbl2bbl2bbl2bbl2bblm
pfimpfimpfimpfimpfimpfimpfimpfimpfiidimpfimpfimpfimpfimpfimpfimpfiidimpfiidimpfim
pfiidimpfimpfimpfimpfiidimpfimpfimpfimpfimpfimpfimpfimpfimpfimpfiidimpfi'] to num
eric
```

In [72]:
```
# Convert categorical columns (if any) to numerical using one-hot encoding or la
df = pd.get_dummies(df, drop_first=True)
```

In [74]:
```
# Feature and Target separation
X = df.drop('price', axis=1)  # assuming 'price' is the target variable
y = df['price']
```

In [76]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [78]:
```
# Feature scaling (Standardization)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [80]:
```
# 2. Model Implementation

# Linear Regression
lr = LinearRegression()
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)
```

In [82]:
```
# Decision Tree Regressor
dt = DecisionTreeRegressor(random_state=42)
dt.fit(X_train, y_train)
y_pred_dt = dt.predict(X_test)
```

In [84]:
```
# Random Forest Regressor
rf = RandomForestRegressor(random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)
```

In [86]:
```
# Gradient Boosting Regressor
gb = GradientBoostingRegressor(random_state=42)
gb.fit(X_train, y_train)
y_pred_gb = gb.predict(X_test)
```

In [88]:
```
# Support Vector Regressor (SVR)
svr = SVR()
svr.fit(X_train, y_train)
y_pred_svr = svr.predict(X_test)
```

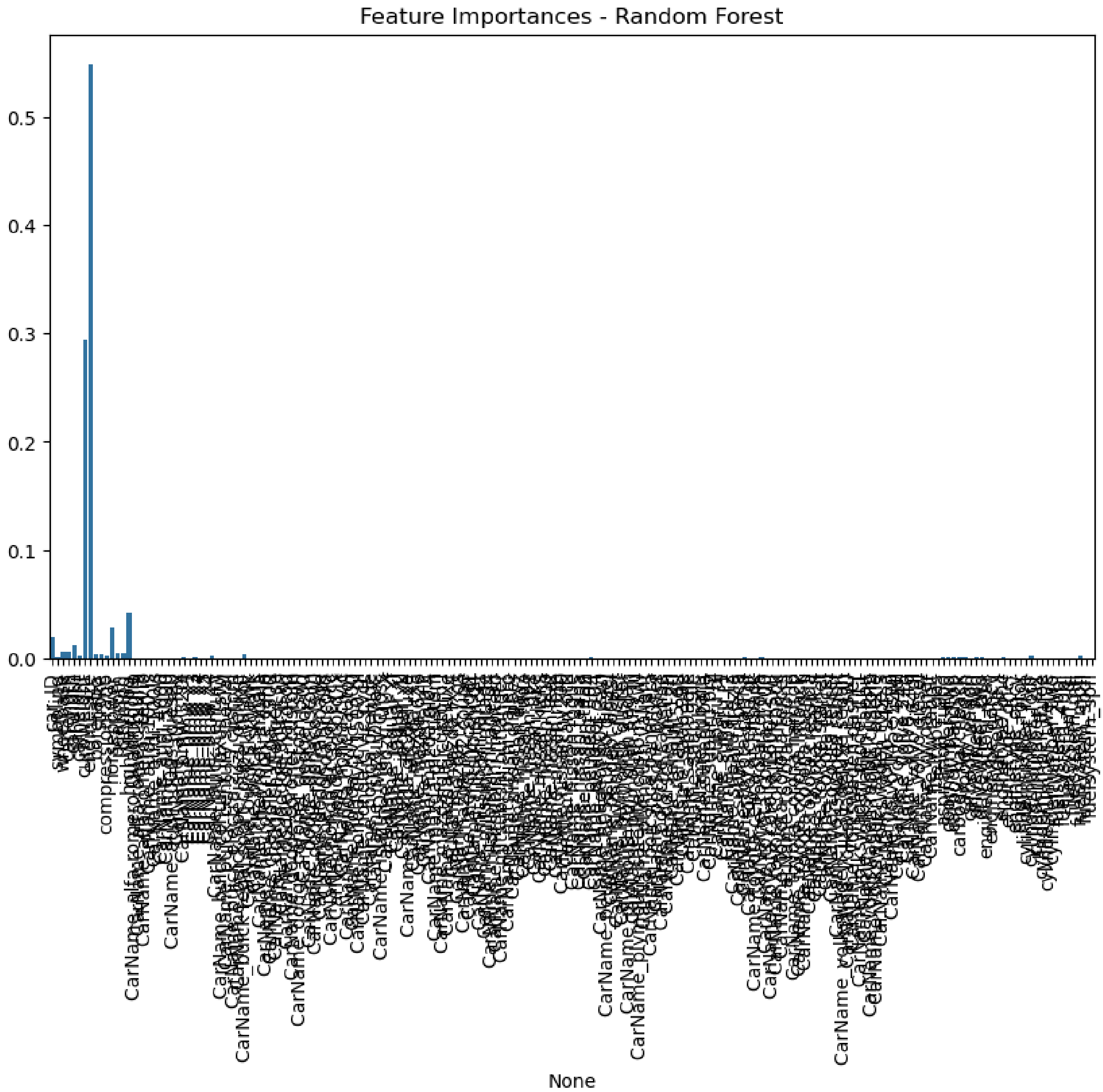
In [90]:
```
# 3. Model Evaluation

# Evaluate each model
def evaluate_model(model_name, y_true, y_pred):
    r2 = r2_score(y_true, y_pred)
    mse = mean_squared_error(y_true, y_pred)
    mae = mean_absolute_error(y_true, y_pred)
    print(f"Model: {model_name}")
    print(f"R-squared: {r2:.4f}")
    print(f"MSE: {mse:.4f}")
    print(f"MAE: {mae:.4f}\n")

evaluate_model('Linear Regression', y_test, y_pred_lr)
evaluate_model('Decision Tree Regressor', y_test, y_pred_dt)
evaluate_model('Random Forest Regressor', y_test, y_pred_rf)
```

```
evaluate_model('Gradient Boosting Regressor', y_test, y_pred_gb)
evaluate_model('Support Vector Regressor', y_test, y_pred_svr)
```

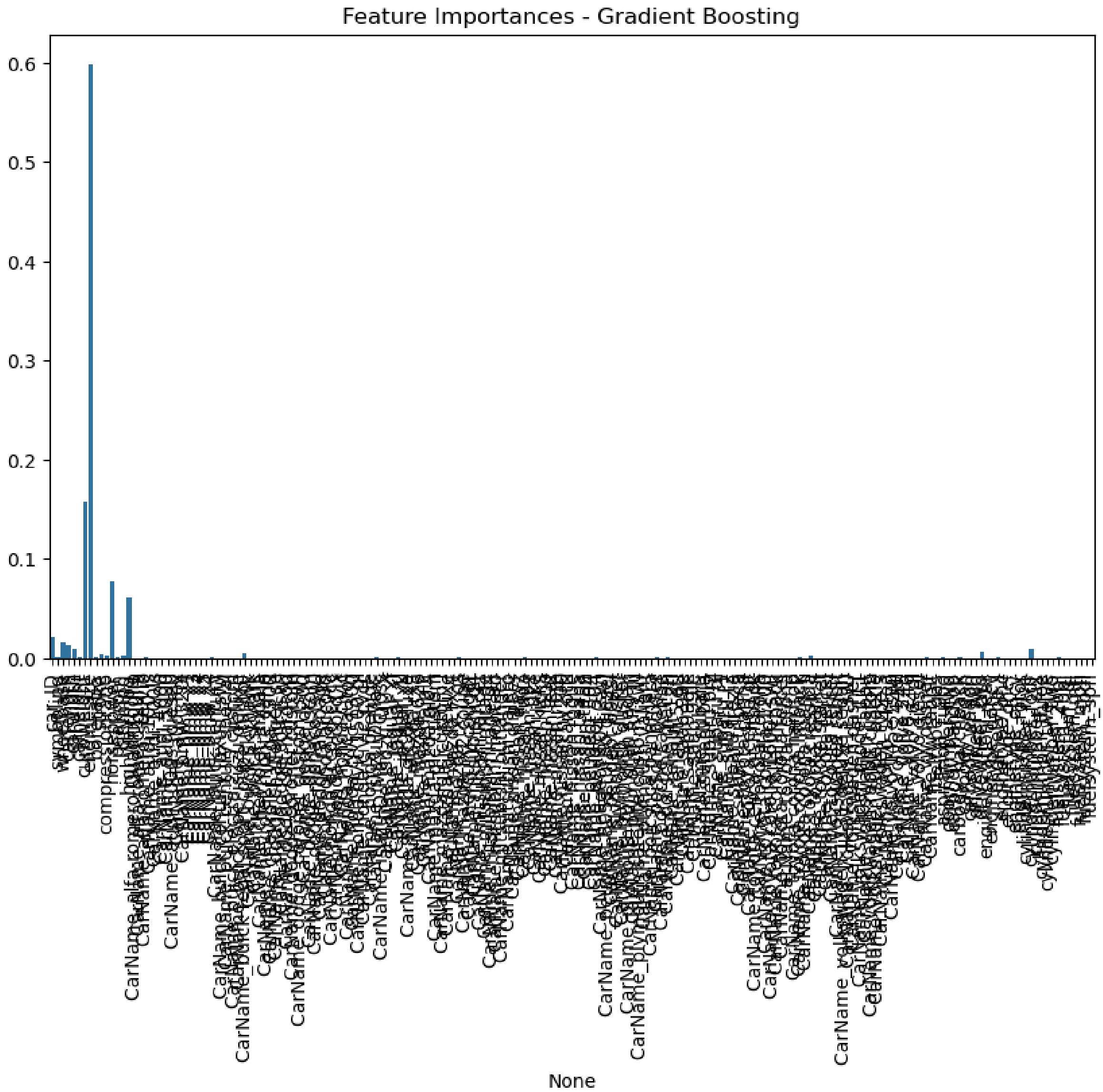
```
Model: Linear Regression
R-squared: -3338166280152721685217280.0000
MSE: 26352826852014566881123868082176.0000
MAE: 2482893968886244.0000

Model: Decision Tree Regressor
R-squared: 0.8666
MSE: 10532678.5297
MAE: 2098.3090

Model: Random Forest Regressor
R-squared: 0.9537
MSE: 3652007.2005
MAE: 1378.8925

Model: Gradient Boosting Regressor
R-squared: 0.9316
MSE: 5402849.3765
MAE: 1685.6164

Model: Support Vector Regressor
R-squared: -0.1017
MSE: 86973995.1459
MAE: 5705.0610
```

In [92]:
```
# 4. Feature Importance Analysis
# Random Forest and Gradient Boosting provide feature importances
feature_importances_rf = rf.feature_importances_
feature_importances_gb = gb.feature_importances_
```

In [56]:
```
# Plot feature importances for Random Forest
plt.figure(figsize=(10, 6))
sns.barplot(x=X.columns, y=feature_importances_rf)
plt.title("Feature Importances - Random Forest")
plt.xticks(rotation=90)
plt.show()
```



In [58]:
```
# Plot feature importances for Gradient Boosting
plt.figure(figsize=(10, 6))
sns.barplot(x=X.columns, y=feature_importances_gb)
plt.title("Feature Importances - Gradient Boosting")
plt.xticks(rotation=90)
plt.show()
```



In [62]:

```python
# 5. Hyperparameter Tuning (Example for Random Forest)
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30],
    'min_samples_split': [2, 5, 10]
}
grid_search = GridSearchCV(RandomForestRegressor(random_state=42), param_grid, c
grid_search.fit(X_train, y_train)

# Best parameters
print(f"Best Parameters: {grid_search.best_params_}")

# Evaluate the tuned model
best_rf = grid_search.best_estimator_
y_pred_best_rf = best_rf.predict(X_test)
evaluate_model('Tuned Random Forest', y_test, y_pred_best_rf)
```

```
Best Parameters: {'max_depth': 10, 'min_samples_split': 2, 'n_estimators': 100}
Model: Tuned Random Forest
R-squared: 0.9528
MSE: 3722559.9635
MAE: 1380.4867
```

In [ ]: