```
from google.colab import files
uploaded = files.upload()
```

Choose files CarPrice_Assignment.csv

- **CarPrice_Assignment.csv**(text/csv) - 26717 bytes, last modified: 21/06/2025 - 100% done

```
Saving CarPrice_Assignment.csv to CarPrice_Assignment.csv
```

```
import pandas as pd

df = pd.read_csv('CarPrice_Assignment.csv')
df.head()
```

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	en
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	

5 rows × 26 columns

```
df = pd.read_csv('CarPrice_Assignment.csv')
df.head()
```

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	en
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	

5 rows × 26 columns

```
#Data Cleaning
df.drop(['car_ID'], axis=1, inplace=True)
df['CarCompany'] = df['CarName'].apply(lambda x: x.split(' ')[0].lower())
df.drop(['CarName'], axis=1, inplace=True)
df['CarCompany'] = df['CarCompany'].replace({
    'vw': 'volkswagen',
    'vokswagen': 'volkswagen',
    'porcshce': 'porsche',
    'toyouta': 'toyota',
    'maxda': 'mazda',
    'Nissan': 'nissan'
})
df[['CarCompany']].drop_duplicates().sort_values(by='CarCompany')
```

None
Like what you see? Visit the data table notebook to learn more about interactive tables.

1 to 22 of 22 entries Filter

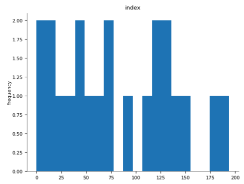
index	CarCompany
0	alfa-romero
3	audi
10	bmw
67	buick
18	chevrolet
21	dodge
30	honda
43	isuzu
47	jaguar
50	mazda
75	mercury
76	mitsubishi
89	nissan
107	peugeot
118	plymouth
125	porsche
130	renault
132	saab
138	subaru
150	toyota
182	volkswagen
194	volvo

Show 25 per page

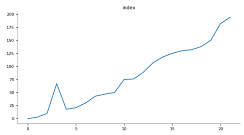
Like what you see? Visit the data table notebook to learn more about interactive tables.

```
Error: Runtime no longer has a reference to this dataframe, please re-run this cell and try again.
No charts were generated by quickchart
```

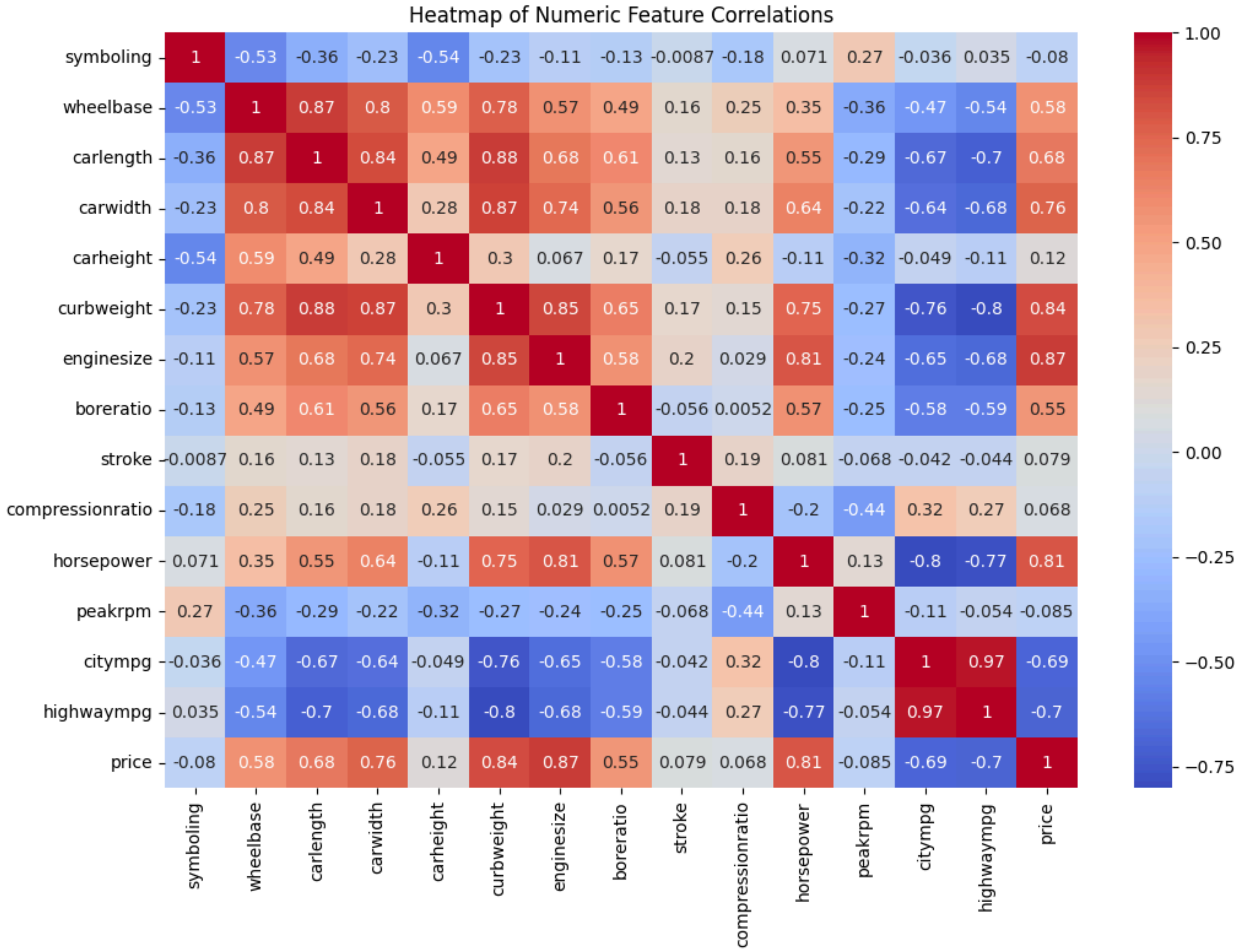
Distributions



Values




```
import matplotlib.pyplot as plt
import seaborn as sns
numeric_df = df.select_dtypes(include=['number'])
plt.figure(figsize=(12, 8))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm')
plt.title("Heatmap of Numeric Feature Correlations")
plt.show()
```




```
df.drop(['carlength', 'carwidth', 'curbweight', 'highwaympg'], axis=1, inplace=True)
df.columns
```

```
Index(['symboling', 'fueltype', 'aspiration', 'doornumber', 'carbody',
       'drivewheel', 'enginelocation', 'wheelbase', 'carheight', 'enginetype',
       'cylindernumber', 'enginesize', 'fuelsystem', 'boreratio', 'stroke',
       'compressionratio', 'horsepower', 'peakrpm', 'citympg', 'price',
       'CarCompany'],
      dtype='object')
```

```
df = pd.get_dummies(df, drop_first=True)
df.head()
```

	symboling	wheelbase	carheight	enginesize	boreratio	stroke	compressionratio	horsepower	peakrpm	citympg	...	Car
0	3	88.6	48.8	130	3.47	2.68	9.0	111	5000	21	...	
1	3	88.6	48.8	130	3.47	2.68	9.0	111	5000	21	...	
2	1	94.5	52.4	152	2.68	3.47	9.0	154	5000	19	...	
3	2	99.8	54.3	109	3.19	3.40	10.0	102	5500	24	...	
4	2	99.4	54.3	136	3.19	3.40	8.0	115	5500	18	...	

5 rows × 61 columns

```
from sklearn.model_selection import train_test_split
X = df.drop('price', axis=1)
y = df['price']
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, random_state=100)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((143, 60), (62, 60), (143,), (62,))
```

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
```

```
X_train_scaled = pd.DataFrame(scaler.fit_transform(X_train), columns=X_train.columns)
X_test_scaled = pd.DataFrame(scaler.transform(X_test), columns=X_test.columns)
X_train_scaled.head()
```

	symboling	wheelbase	carheight	enginesize	boreratio	stroke	compressionratio	horsepower	peakrpm	citympg	...	Ca
0	0.6	0.244828	0.265487	0.139623	0.230159	0.525253	0.15000	0.083333	0.551020	0.500000	...	
1	1.0	0.272414	0.212389	0.339623	1.000000	0.464646	0.15625	0.395833	0.551020	0.166667	...	
2	0.6	0.272414	0.424779	0.139623	0.444444	0.449495	0.15000	0.266667	1.000000	0.361111	...	
3	1.0	0.068966	0.088496	0.260377	0.626984	0.247475	0.12500	0.262500	0.346939	0.222222	...	
4	0.2	0.610345	0.858407	0.260377	0.746032	0.484848	0.03125	0.475000	0.387755	0.111111	...	

5 rows × 60 columns

```
from sklearn.linear_model import LinearRegression
from sklearn.feature_selection import RFE
rfe = RFE(estimator=lm, n_features_to_select=15)
rfe = rfe.fit(X_train_scaled, y_train)
selected_cols = X_train_scaled.columns[rfe.support_]
selected_cols
```

```
Index(['wheelbase', 'enginesize', 'boreratio', 'stroke', 'enginelocation_rear',
       'enginetype_rotor', 'cylindernumber_five', 'cylindernumber_four',
       'cylindernumber_three', 'cylindernumber_twelve', 'cylindernumber_two',
       'CarCompany_bmw', 'CarCompany_peugeot', 'CarCompany_porsche',
       'CarCompany_saab'],
      dtype='object')
```

```
X_train_rfe = X_train_scaled[selected_cols]
```

```
y_train = y_train.reset_index(drop=True)
```

```
import statsmodels.api as sm
X_train_sm = sm.add_constant(X_train_rfe)
model = sm.OLS(y_train, X_train_sm).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.918
Model:                            OLS   Adj. R-squared:                  0.909
Method:                 Least Squares   F-statistic:                     110.6
Date:                Sat, 21 Jun 2025   Prob (F-statistic):           2.87e-63
Time:                        08:57:54   Log-Likelihood:                -1305.5
No. Observations:                 143   AIC:                             2639.
Df Residuals:                     129   BIC:                             2681.
Df Model:                          13
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                 -1807.3445   1470.437     -1.229      0.221   -4716.641    1101.952
wheelbase              7423.9684   1434.279      5.176      0.000    4586.212    1.03e+04
enginesize             6.581e+04   5777.814     11.389      0.000    5.44e+04    7.72e+04
boreratio             -1.202e+04   2264.580     -5.306      0.000   -1.65e+04   -7536.458
stroke                -1.025e+04   1946.119     -5.267      0.000   -1.41e+04   -6400.546
enginelocation_rear    5662.5883   2928.413      1.934      0.055    -131.349    1.15e+04
enginetype_rotor       1.073e+04   1359.969      7.889      0.000    8037.985    1.34e+04
cylindernumber_five    8680.7402   1305.215      6.651      0.000    6098.340    1.13e+04
cylindernumber_four    7561.7451   1605.488      4.710      0.000    4385.248    1.07e+04
cylindernumber_three   1.304e+04   3061.069      4.260      0.000    6983.628    1.91e+04
cylindernumber_twelve -2.079e+04   3997.090     -5.201      0.000   -2.87e+04   -1.29e+04
cylindernumber_two     1.073e+04   1359.969      7.889      0.000    8037.985    1.34e+04
CarCompany_bmw         8612.3999   1101.978      7.815      0.000    6432.110    1.08e+04
CarCompany_porsche     1.019e+04   1859.922      5.478      0.000    6508.483    1.39e+04
CarCompany_saab        4230.7827   1398.202      3.026      0.003    1464.405    6997.161
==============================================================================
Omnibus:                       22.726   Durbin-Watson:                   1.913
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               35.142
Skew:                           0.808   Prob(JB):                     2.34e-08
Kurtosis:                       4.813   Cond. No.                     1.39e+17
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.75e-32. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

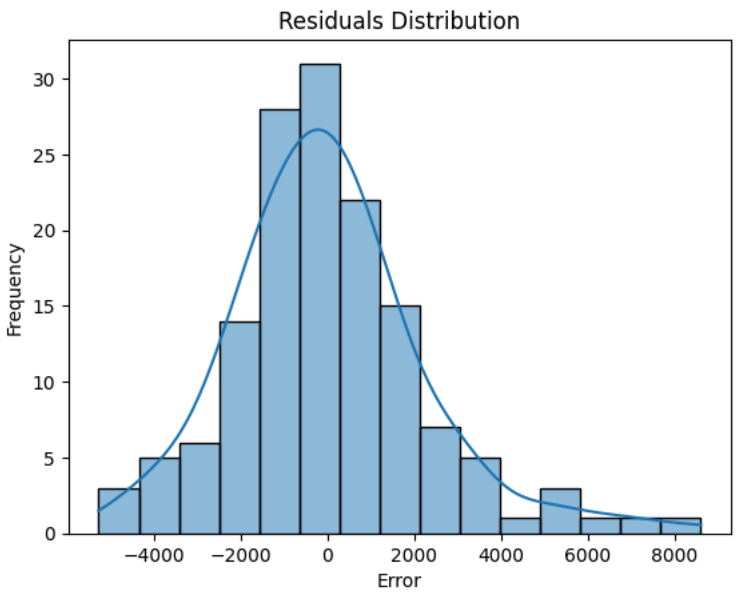
```
X_train_rfe = X_train_rfe.drop('CarCompany_peugeot', axis=1)
```

```
X_train_sm = sm.add_constant(X_train_rfe)
model = sm.OLS(y_train, X_train_sm).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.918
Model:                            OLS   Adj. R-squared:                  0.909
Method:                 Least Squares   F-statistic:                     110.6
Date:                Sat, 21 Jun 2025   Prob (F-statistic):           2.87e-63
Time:                        08:58:13   Log-Likelihood:                -1305.5
No. Observations:                 143   AIC:                             2639.
Df Residuals:                     129   BIC:                             2681.
Df Model:                          13
Covariance Type:            nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
const               -1807.3445   1470.437     -1.229      0.221   -4716.641    1101.952
wheelbase            7423.9684   1434.279      5.176      0.000    4586.212    1.03e+04
enginesize           6.581e+04   5777.814     11.389      0.000    5.44e+04    7.72e+04
boreratio           -1.202e+04   2264.580     -5.306      0.000   -1.65e+04   -7536.458
stroke              -1.025e+04   1946.119     -5.267      0.000   -1.41e+04   -6400.546
enginelocation_rear  5662.5883   2928.413      1.934      0.055    -131.349    1.15e+04
enginetype_rotor     1.073e+04   1359.969      7.889      0.000    8037.985    1.34e+04
cylindernumber_five  8680.7402   1305.215      6.651      0.000    6098.340    1.13e+04
cylindernumber_four  7561.7451   1605.488      4.710      0.000    4385.248    1.07e+04
cylindernumber_three 1.304e+04   3061.069      4.260      0.000    6983.628    1.91e+04
cylindernumber_twelve -2.079e+04  3997.090     -5.201      0.000   -2.87e+04   -1.29e+04
cylindernumber_two   1.073e+04   1359.969      7.889      0.000    8037.985    1.34e+04
CarCompany_bmw       8612.3999   1101.978      7.815      0.000    6432.110    1.08e+04
CarCompany_porsche   1.019e+04   1859.922      5.478      0.000    6508.483    1.39e+04
CarCompany_saab      4230.7827   1398.202      3.026      0.003    1464.405    6997.161
==============================================================================
Omnibus:                       22.726   Durbin-Watson:                   1.913
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               35.142
Skew:                           0.808   Prob(JB):                     2.34e-08
Kurtosis:                       4.813   Cond. No.                     1.39e+17
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.75e-32. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

```
y_train_pred = model.predict(X_train_sm)
residuals = y_train - y_train_pred
import seaborn as sns
import matplotlib.pyplot as plt
sns.histplot(residuals, kde=True)
plt.title("Residuals Distribution")
plt.xlabel("Error")
plt.ylabel("Frequency")
plt.show()
```




```
selected_cols = X_train_rfe.columns
```