In [1]:
```
#Import necessary libraries

import pandas as pd
import numpy as np
import mglearn
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

Linear Regression

In [2]:
```
#Load the dataset

df = pd.read_csv('Advertising Budget and Sales.csv')
df = df.drop(df.columns[0], axis=1)
```

In [3]:
```
df.head()
```

Out[3]:

	TV Ad Budget ($)	Radio Ad Budget ($)	Newspaper Ad Budget ($)	Sales ($)
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9

In [4]:
```
df = df.rename(columns={
    'TV Ad Budget ($)' : 'TV_ads_budget',
    'Radio Ad Budget ($)' : 'Ra_ads_budget',
    'Newspaper Ad Budget ($)' : 'Ne_ads_budget',
    'Sales ($)' : 'Sales'
})
```

In [5]:
```
#Data preprocessing

X = df[['TV_ads_budget']]  # Independent variable(s)
y = df['Sales']    # Dependent variable (target)
```

In [6]:
```
#Splitting 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_state=4
```

In [7]:
```
#Fitting the Linear Regression model
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

```python
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[7]: LinearRegression()

In [8]:

```python
#Making predictions and evaluating the model

y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

In [9]:

```python
print(f"Mean Squared Error: {mse:.3f}")
print(f"R² Score: {r2:.3f}")
```

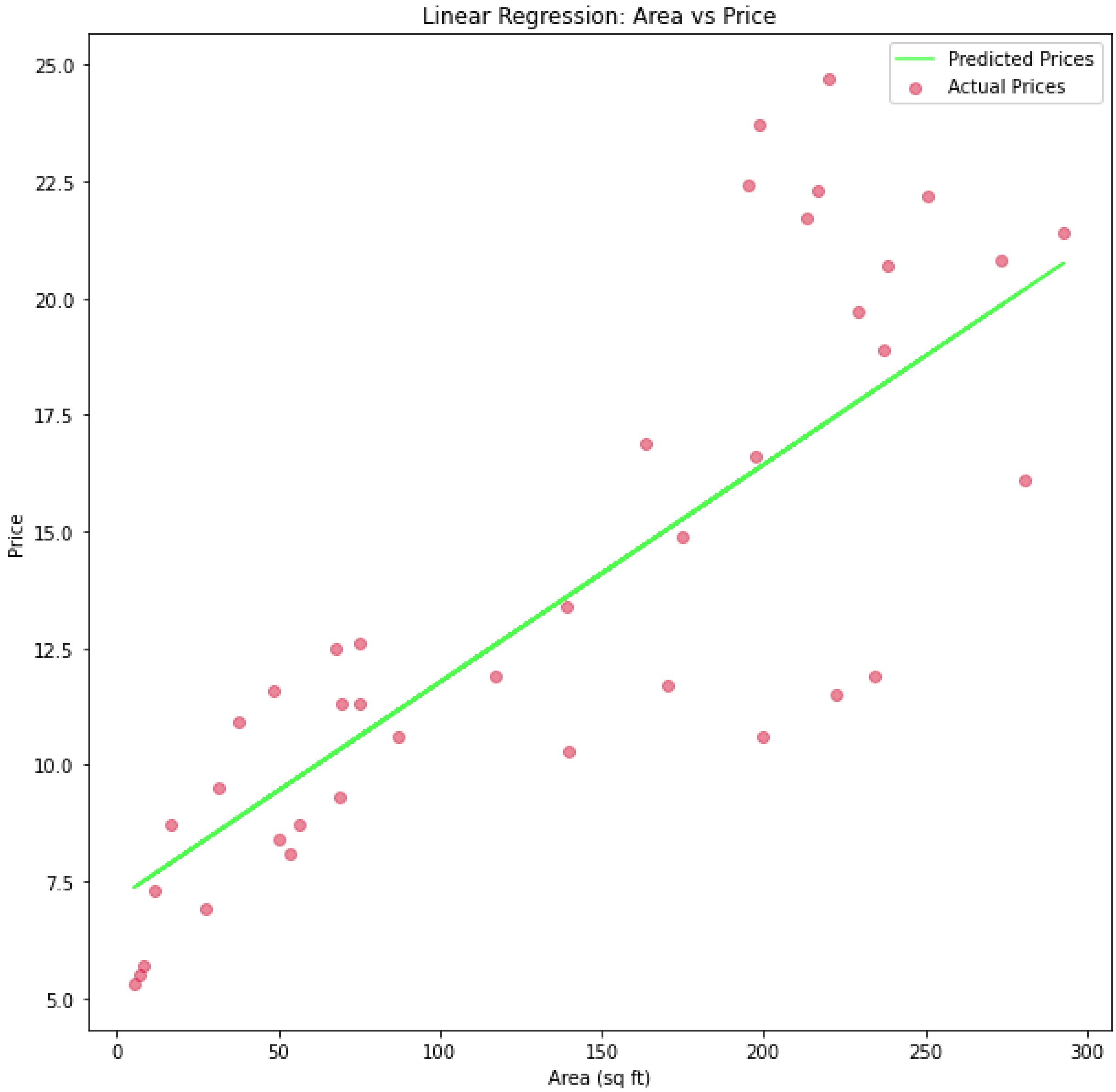
```
Mean Squared Error: 10.205
R² Score: 0.677
```

In [10]:

```python
#Visualizing the Results

plt.figure(figsize=(10,10))
plt.scatter(X_test, y_test, color='#DC143C', label='Actual Prices', alpha = 0.5)
plt.plot(X_test, y_pred, color='#00FF00', label='Predicted Prices', alpha = 0.7)
plt.xlabel('Area (sq ft)')
plt.ylabel('Price')
plt.title('Linear Regression: Area vs Price')
plt.legend()
plt.show()
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js



Multiple Linear Regression

In [11]:

```
#Define features and target variable

X = df[['TV_ads_budget', 'Ra_ads_budget', 'Ne_ads_budget']]  # Independent variable(s)
y = df['Sales']    # Dependent variable (target)
```

In [12]:

```
#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_state=4
```

In [13]:

```
#Fit a multiple linear regression model

model = LinearRegression()
model.fit(X_train, y_train)
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Out[13]: LinearRegression()

In [14]:
```
#Make predictions

y_pred = model.predict(X_test)
```

In [15]:
```
#Evaluate the model

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

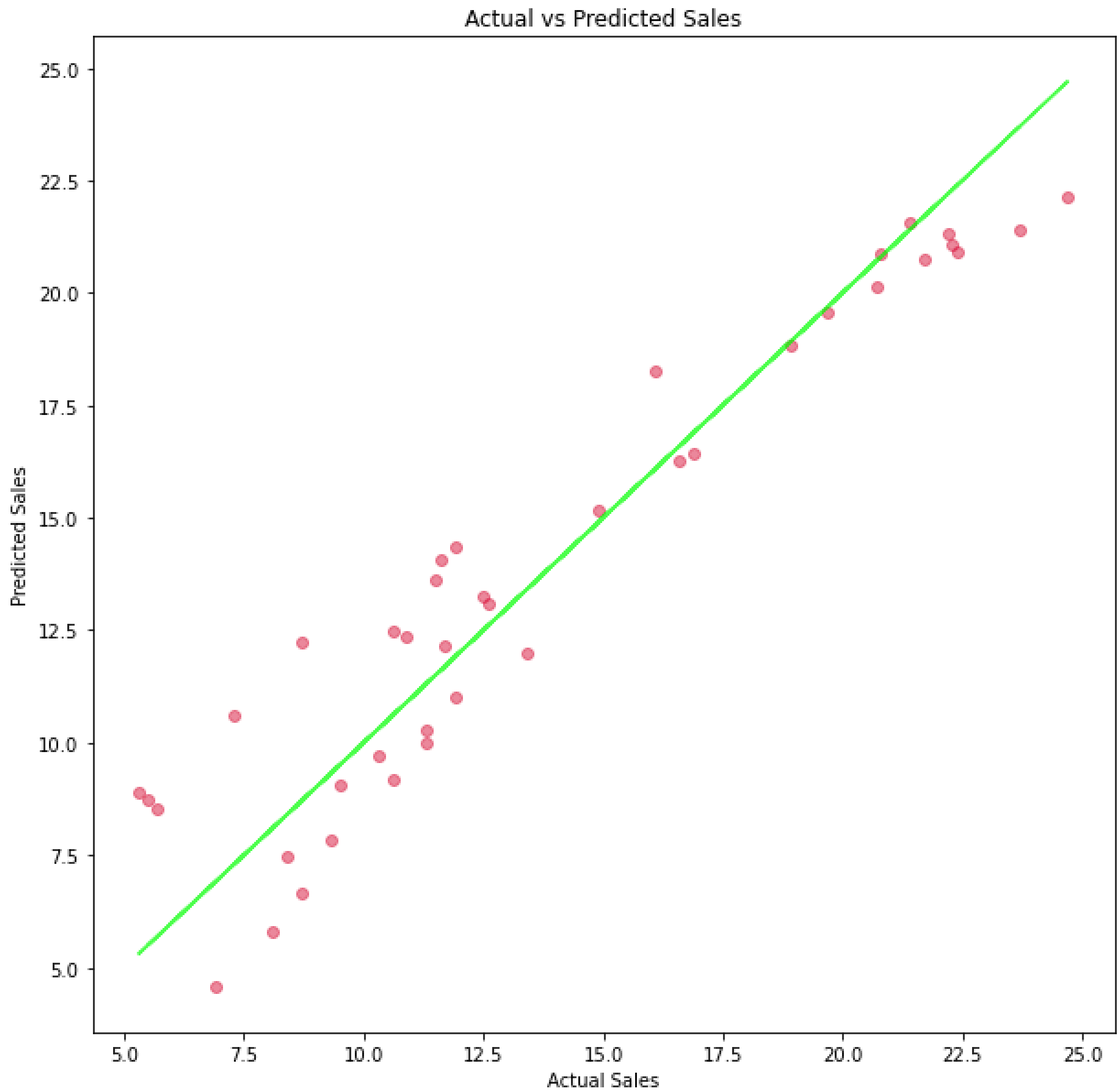
In [16]:
```
print(f"Mean Squared Error: {mse:.3f}")
print(f"R² Score: {r2:.3f}")
```

```
Mean Squared Error: 3.174
R² Score: 0.899
```

In [17]:
```
# Visualizing Actual vs Predicted

plt.figure(figsize=(10, 10))
plt.scatter(y_test, y_pred, color='#DC143C', alpha = 0.5)
plt.plot(y_test, y_test, color='#00FF00', alpha = 0.7)
plt.xlabel('Actual Sales')
plt.ylabel('Predicted Sales')
plt.title('Actual vs Predicted Sales')
plt.show()
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js



Multiple Linear Regression (L2 & L1)

In [18]:

```
from sklearn.linear_model import Ridge, Lasso
from sklearn.model_selection import train_test_split
```

In [19]:

```
#Initialize Ridge and Lasso regression models

ridge_model = Ridge(alpha=100)
lasso_model = Lasso(alpha=1)
```

In [20]:

```
#Fit both models to the training data

ridge_model.fit(X_train, y_train)
lasso_model.fit(X_train, y_train)
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

In [21]:
```
#Make predictions with both models

ridge_pred = ridge_model.predict(X_test)
lasso_pred = lasso_model.predict(X_test)
```

In [22]:
```
#Evaluate both models using Mean Squared Error and R² Score

ridge_mse = mean_squared_error(y_test, ridge_pred)
lasso_mse = mean_squared_error(y_test, lasso_pred)

ridge_r2 = r2_score(y_test, ridge_pred)
lasso_r2 = r2_score(y_test, lasso_pred)
```

In [23]:
```
print(f"Ridge Regression - Mean Squared Error: {ridge_mse:.3f}, R² Score: {ridge_r2:.3f
print(f"Lasso Regression - Mean Squared Error: {lasso_mse:.3f}, R² Score: {lasso_r2:.3f
```

```
Ridge Regression - Mean Squared Error: 3.174, R² Score: 0.899
Lasso Regression - Mean Squared Error: 3.144, R² Score: 0.900
```

KNN Regression

In [24]:
```
from sklearn.neighbors import KNeighborsRegressor
```

In [25]:
```
#Define features and target variable

X = df[['TV_ads_budget', 'Ra_ads_budget', 'Ne_ads_budget']]  # Independent variable(s)
y = df['Sales']    # Dependent variable (target)
```

In [26]:
```
#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_state=4
```

In [27]:
```
#Fit a KNN regression model

k = 5
knn_model = KNeighborsRegressor(n_neighbors=k)
knn_model.fit(X_train, y_train)
```

Out[27]:
```
KNeighborsRegressor()
```

In [28]:
```
#Make predictions

y_pred = knn_model.predict(X_test)
```

In [29]:
```
#Evaluate the model

ed)
r2 = r2_score(y_test, y_pred)
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

In [30]:
```
print(f"Mean Squared Error: {mse:.3f}")
print(f"R² Score: {r2:.3f}")
```

```
Mean Squared Error: 2.821
R² Score: 0.911
```

Decision Tree Regression

In [31]:
```
from sklearn.tree import DecisionTreeRegressor
```

In [32]:
```
X = df[['TV_ads_budget', 'Ra_ads_budget', 'Ne_ads_budget']]  # Independent variable(s)
y = df['Sales']    # Dependent variable (target)
```

In [33]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4
```

In [34]:
```
model = DecisionTreeRegressor(random_state=42)
model.fit(X_train, y_train)
```

Out[34]:
```
DecisionTreeRegressor(random_state=42)
```

In [35]:
```
y_pred = model.predict(X_test)
```

In [36]:
```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

In [37]:
```
print(f"Mean Squared Error: {mse:.3f}")
print(f"R² Score: {r2:.3f}")
```

```
Mean Squared Error: 2.175
R² Score: 0.931
```

Random Forest Regression

In [38]:
```
from sklearn.ensemble import RandomForestRegressor
```

In [39]:
```
X = df[['TV_ads_budget', 'Ra_ads_budget', 'Ne_ads_budget']]  # Independent variable(s)
y = df['Sales']    # Dependent variable (target)
```

In [40]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4
```

In [41]:
```
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Out[41]:
```
RandomForestRegressor(random_state=42)
```

In [42]:
```
y_pred = rf_model.predict(X_test)
```

In [43]:
```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

In [44]:
```
print(f"Mean Squared Error: {mse:.3f}")
print(f"R² Score: {r2:.3f}")
```

```
Mean Squared Error: 0.591
R² Score: 0.981
```

Summary

In [45]:
```
summary = {
    'Linear Regression' : [10.205, 0.677],
    'Multiple Linear Regression' : [3.174, 0.899],
    'Ridge Multiple Linear Regression' : [3.174, 0.899],
    'Lasso Multiple Linear Regression' : [3.144, 0.900],
    'KNN Multiple Regression' : [2.821, 0.911],
    'Decision Tree Regression' : [2.175, 0.931],
    'Random Forest Regression' : [0.591, 0.981]
}
```

In [46]:
```
df_summary = pd.DataFrame.from_dict(summary, orient='index', columns=['MSE','R² Score']
```

In [47]:
```
df_summary.reset_index(inplace=True)
df_summary.rename(columns={'index': 'Model'})
```

Out[47]:

	Model	MSE	R^2 Score
0	Linear Regression	10.205	0.677
1	Multiple Linear Regression	3.174	0.899
2	Ridge Multiple Linear Regression	3.174	0.899
3	Lasso Multiple Linear Regression	3.144	0.900
4	KNN Multiple Regression	2.821	0.911
5	Decision Tree Regression	2.175	0.931
6	Random Forest Regression	0.591	0.981

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js