Ridge & Lasso Regression

In [20]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn.linear_model import Lasso, Ridge
```

In [41]:

```
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
from sklearn.datasets import fetch_california_housing
```

In [3]:

```
df = fetch_california_housing()
df
```

Out[3]:
```
{'data': array([[   8.3252    ,   41.        ,    6.98412698, ...,    2.55555556,
           37.88      , -122.23      ],
        [   8.3014    ,   21.        ,    6.23813708, ...,    2.10984183,
           37.86      , -122.22      ],
        [   7.2574    ,   52.        ,    8.28813559, ...,    2.80225989,
           37.85      , -122.24      ],
        ...,
        [   1.7       ,   17.        ,    5.20554273, ...,    2.3256351 ,
           39.43      , -121.22      ],
        [   1.8672    ,   18.        ,    5.32951289, ...,    2.12320917,
           39.43      , -121.32      ],
        [   2.3886    ,   16.        ,    5.25471698, ...,    2.61698113,
           39.37      , -121.24      ]]),
 'target': array([4.526, 3.585, 3.521, ..., 0.923, 0.847, 0.894]),
 'frame': None,
 'target_names': ['MedHouseVal'],
 'feature_names': ['MedInc',
  'HouseAge',
  'AveRooms',
  'AveBedrms',
  'Population',
  'AveOccup',
  'Latitude',
  'Longitude'],
 'DESCR': '.. _california_housing_dataset:\n\nCalifornia Housing dataset\n--------
------------------\n\n**Data Set Characteristics:**\n\n:Number of Instances: 20640
\n\n:Number of Attributes: 8 numeric, predictive attributes and the target\n\n:Att
ribute Information:\n    - MedInc        median income in block group\n    - House
Age      median house age in block group\n    - AveRooms      average number of ro
oms per household\n    - AveBedrms     average number of bedrooms per household\n
- Population    block group population\n    - AveOccup      average number of hous
ehold members\n    - Latitude      block group latitude\n    - Longitude     block
group longitude\n\n:Missing Attribute Values: None\n\nThis dataset was obtained fr
om the StatLib repository.\nhttps://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housin
g.html\n\nThe target variable is the median house value for California district
s,\nexpressed in hundreds of thousands of dollars ($100,000).\n\nThis dataset was
derived from the 1990 U.S. census, using one row per census\nblock group. A block
group is the smallest geographical unit for which the U.S.\nCensus Bureau publishe
s sample data (a block group typically has a population\nof 600 to 3,000 peopl
e).\n\nA household is a group of people residing within a home. Since the average
\nnumber of rooms and bedrooms in this dataset are provided per household, these\n
columns may take surprisingly large values for block groups with few households\na
nd many empty houses, such as vacation resorts.\n\nIt can be downloaded/loaded usi
ng the\n:func:`sklearn.datasets.fetch_california_housing` function.\n\n.. rubric::
References\n\n- Pace, R. Kelley and Ronald Barry, Sparse Spatial Autoregression
s,\n  Statistics and Probability Letters, 33:291-297, 1997.\n'}
```

In [4]:
```
housing = fetch_california_housing(as_frame=True)
```

In [5]:
```
df = housing.frame
```

In [6]:
```
x = housing.data      #independent Variable
y = housing.target    #dependent variable
```

In [7]:
```
df.head()
```

Out[7]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedH
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	

In [8]:

```
df.shape
```

Out[8]:

```
(20640, 9)
```

In [9]:

```
df.head()
```

Out[9]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedH
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	

In [10]:

```
X= df.iloc[:,:-1]
Y=df.iloc[:,-1]
```

Linear Regression

In [11]:

```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression

lin_regressor=LinearRegression()
mse=cross_val_score(lin_regressor,X,Y,scoring='neg_mean_squared_error',cv=5)
mean_mse=np.mean(mse)
print(mean_mse)
```

```
-0.558290171768654
```

Ridge Regression

In [12]:

```
 # FIND THE VALUES OF LAMBDA & LAMBDA USED TO FIND A CROSS VALIDATION
```

In [13]:

```
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV

ridge=Ridge()
parameters={'alpha':[1e-15,1e-10,1e-8,1e-3,1e-2,1,5,10,20,30,35,40,45,50,55,100]}
ridge_regressor=GridSearchCV(ridge,parameters,scoring='neg_mean_squared_error',cv=5
ridge_regressor.fit(X,Y)
```

Out[13]:

```
▸ GridSearchCV
▸ best_estimator_:
  Ridge
  ▸ Ridge
```

In [14]:

```python
print(ridge_regressor.best_params_)
print(ridge_regressor.best_score_)# Mean Squared Error
```

```
{'alpha': 55}
-0.5579444917053032
```

Lasso Regression

In [15]:

```python
from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV

lasso=Lasso()
parameters={'alpha':[1e-15,1e-10,1e-8,1e-3,1e-2,1,5,10,20,30,35,40,45,50,55,100]}
Lasso_regressor=GridSearchCV(ridge,parameters,scoring='neg_mean_squared_error',cv=5
Lasso_regressor.fit(X,Y)
```

Out[15]:

```
▸ GridSearchCV
▸ best_estimator_:
  Ridge
  ▸ Ridge
```

In [16]:

```python
print(Lasso_regressor.best_params_)
print(Lasso_regressor.best_score_)
```

```
{'alpha': 55}
-0.5579444917053032
```

In [26]:

```python
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.3, random_stat
from sklearn.linear_model import Lasso, Ridge
from sklearn.metrics import mean_squared_error, r2_score
```

In [32]:

```python
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Lasso, Ridge
import pandas as pd

# Split dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta

# Step 1: Create model
lasso_regressor = Lasso(alpha=0.1)
ridge_regressor = Ridge(alpha=1.0)

# Step 2: Train model
lasso_regressor.fit(X_train, y_train)
```

```
ridge_regressor.fit(X_train, y_train)

# Step 3: Predict
prediction_lasso = lasso_regressor.predict(X_test)
prediction_ridge = ridge_regressor.predict(X_test)
```

In [33]:

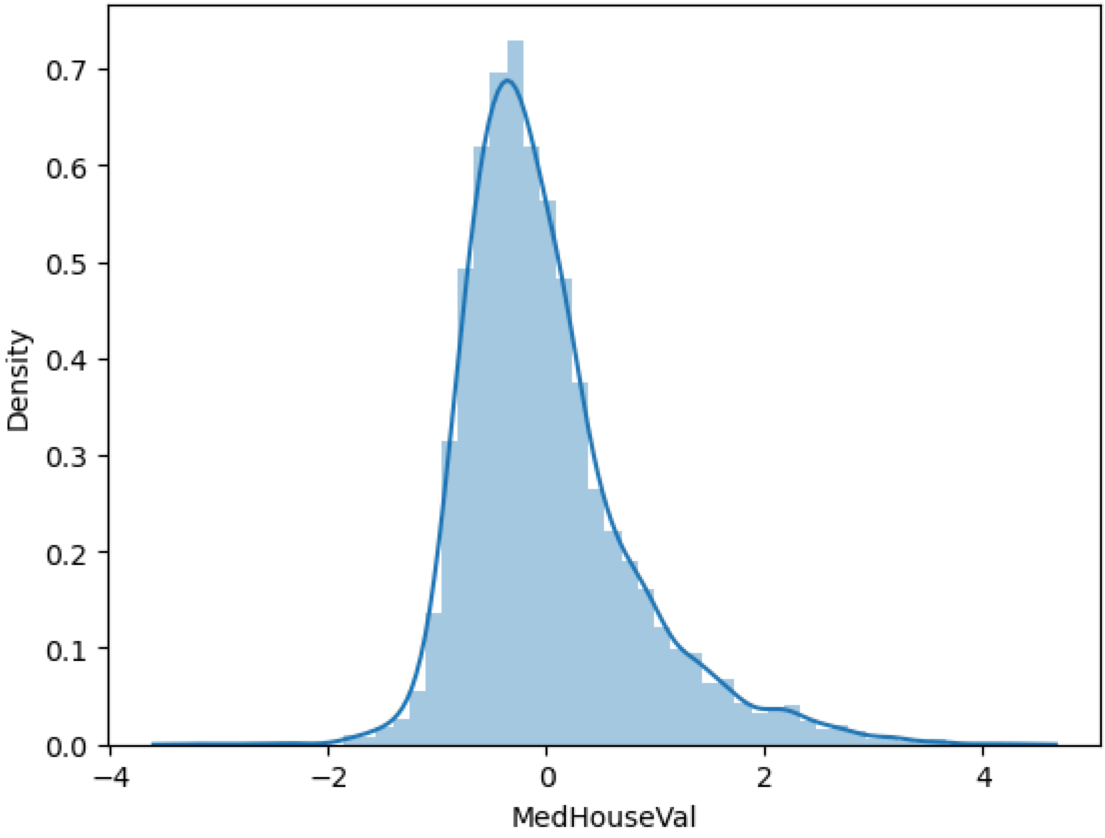
```
prediction_lasso=lasso_regressor.predict(X_test)
prediction_ridge=ridge_regressor.predict(X_test)
```

In [42]:

```
sns.distplot(Y_test-prediction_lasso)
```

Out[42]:

```
<Axes: xlabel='MedHouseVal', ylabel='Density'>
```

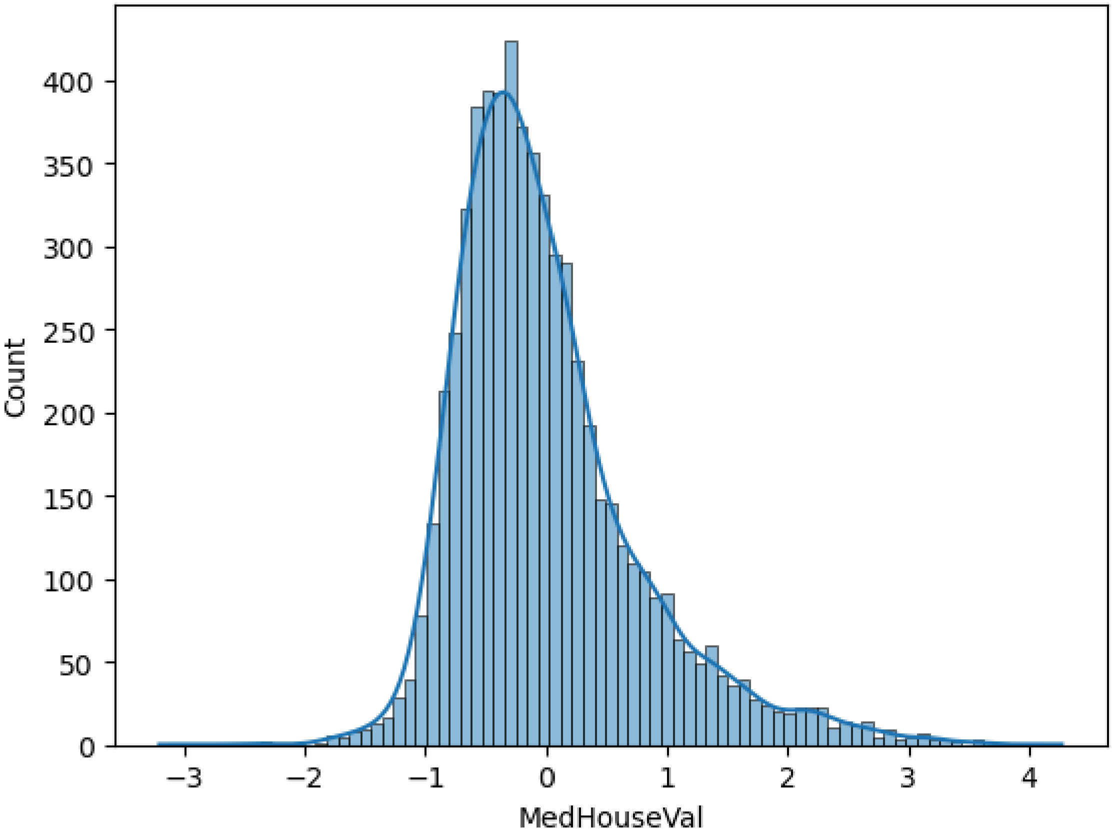


In [39]:

```
sns.histplot(Y_test - prediction_lasso, kde=True)
```

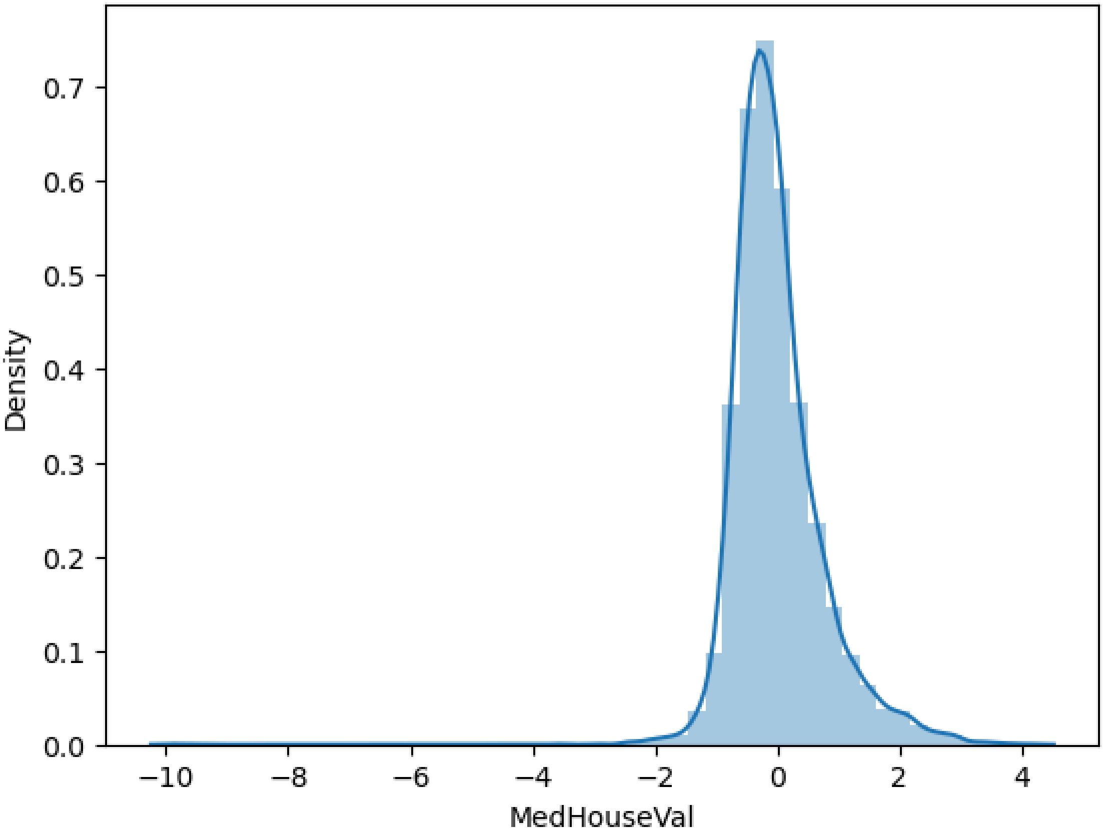
Out[39]:

```
<Axes: xlabel='MedHouseVal', ylabel='Count'>
```



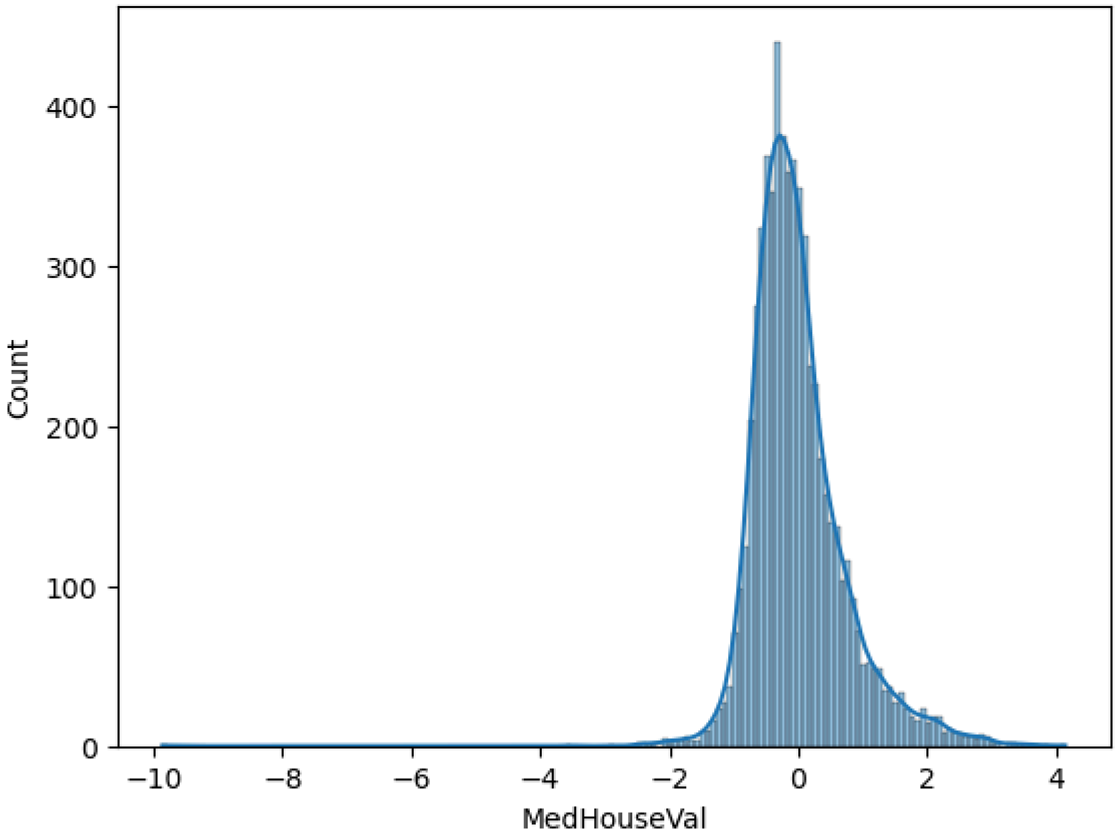
In [49]:
```
sns.distplot(Y_test-prediction_ridge)
```

Out[49]:
```
<Axes: xlabel='MedHouseVal', ylabel='Density'>
```



In [48]:
```
sns.histplot(Y_test - prediction_ridge, kde=True)
```

Out[48]:
```
<Axes: xlabel='MedHouseVal', ylabel='Count'>
```


400
300
200
100
0
Count
−10
−8
−6
−4
−2
0
2
4
MedHouseVal