In [58]:

```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
import os
```

In [59]:

```python
data=pd.read_csv("car data.csv")
data.head()
```

Out[59]:

	Car_Name	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Selling_type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

In [60]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Car_Name       301 non-null    object 
 1   Year           301 non-null    int64  
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Driven_kms     301 non-null    int64  
 5   Fuel_Type      301 non-null    object 
 6   Selling_type   301 non-null    object 
 7   Transmission   301 non-null    object 
 8   Owner          301 non-null    int64  
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

In [61]:

```
data.isna().any()
```

Out[61]:

```
Car_Name         False
Year             False
Selling_Price    False
Present_Price    False
Driven_kms       False
Fuel_Type        False
Selling_type     False
Transmission     False
Owner            False
dtype: bool
```

In [62]:

```
print(data.Fuel_Type.value_counts(),"\n")
print(data.Selling_type.value_counts(),"\n")
print(data.Transmission.value_counts())
```

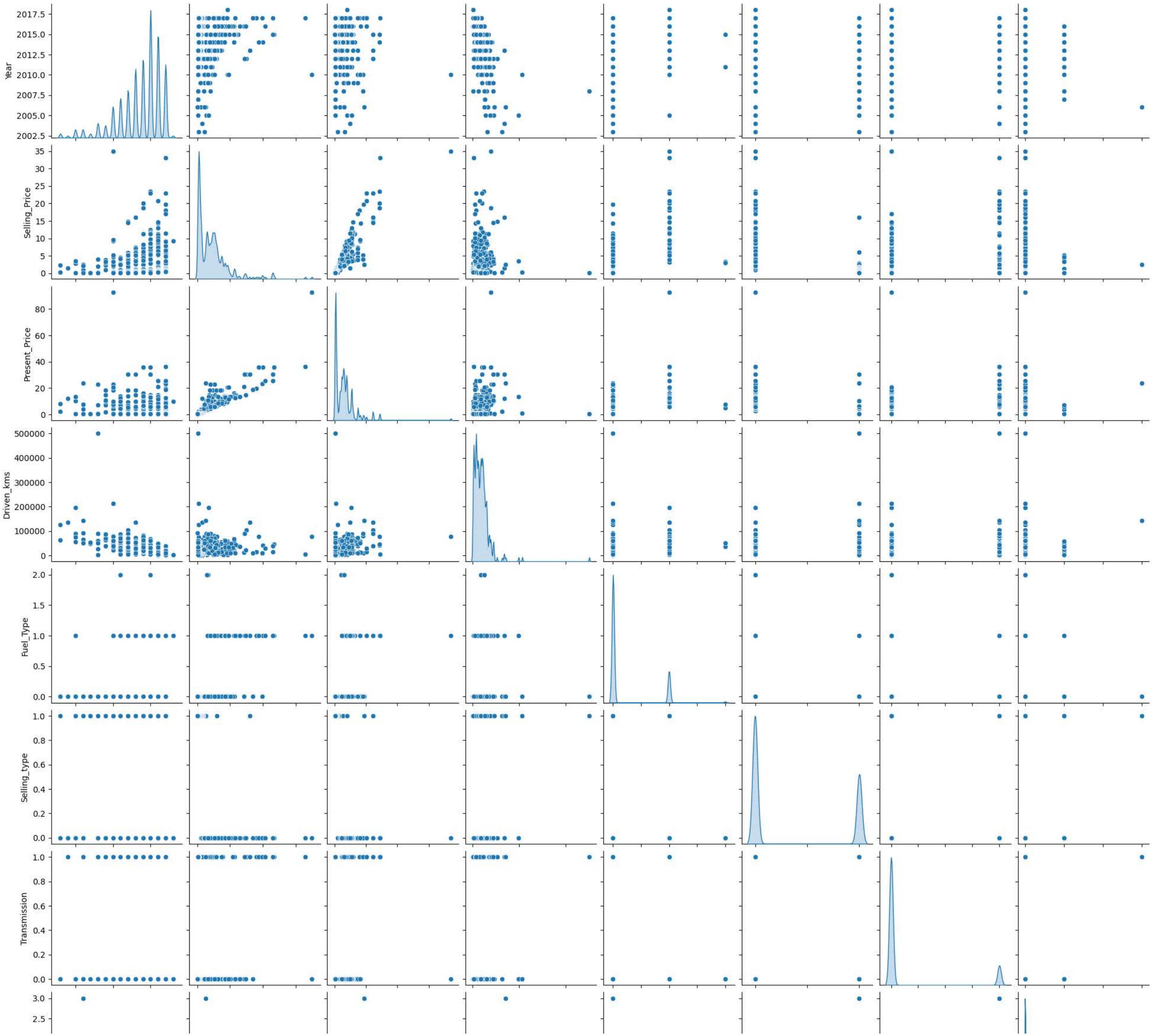
```
Petrol    239
Diesel     60
CNG         2
Name: Fuel_Type, dtype: int64 

Dealer        195
Individual    106
Name: Selling_type, dtype: int64 

Manual       261
Automatic     40
Name: Transmission, dtype: int64
```

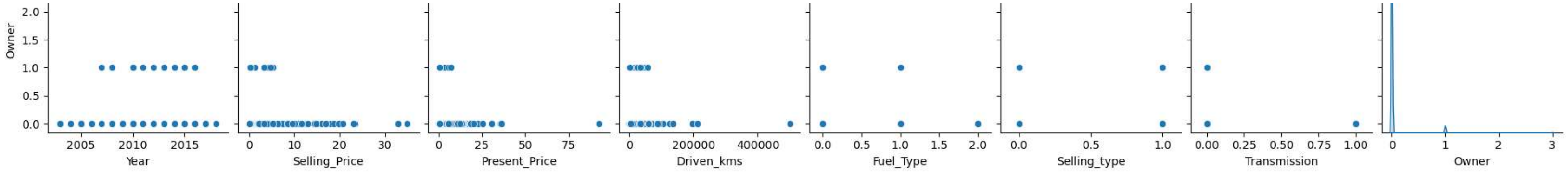
In [63]:

```
data.Fuel_Type.replace(regex={"Petrol":"0","Diesel":"1","CNG":"2"},inplace=True)
data.Selling_type.replace(regex={"Dealer":"0","Individual":"1"},inplace=True)
data.Transmission.replace(regex={"Manual":"0","Automatic":"1"},inplace=True)
data[["Fuel_Type","Selling_type","Transmission"]]=data[["Fuel_Type","Selling_type","Transmission"]].astype(in
```

In [64]:

```
sns.pairplot(data,diag_kind="kde", diag_kws=dict(shade=True, bw=.05, vertical=False))
plt.show()
```


Year
2017.5
2015.0
2012.5
2010.0
2007.5
2005.0
2002.5
Selling_Price
35
30
25
20
15
10
5
0
Present_Price
80
60
40
20
0
Driven_kms
500000
400000
300000
200000
100000
0
Fuel_Type
2.0
1.5
1.0
0.5
0.0
Selling_type
1.0
0.8
0.6
0.4
0.2
0.0
Transmission
1.0
0.8
0.6
0.4
0.2
0.0
3.0
2.5



In [65]:

```
y=data.Selling_Price
x=data.drop(["Selling_Price","Car_Name"],axis=1)
```

In [66]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
print("x train: ",x_train.shape)
print("x test: ",x_test.shape)
print("y train: ",y_train.shape)
print("y test: ",y_test.shape)
```

```
x train:  (240, 7)
x test:  (61, 7)
y train:  (240,)
y test:  (61,)
```

In [67]:

```
from sklearn.metrics import r2_score
from sklearn.model_selection import cross_val_score
```

In [68]:

```
cv=5
r_2 = []
CV = []
```

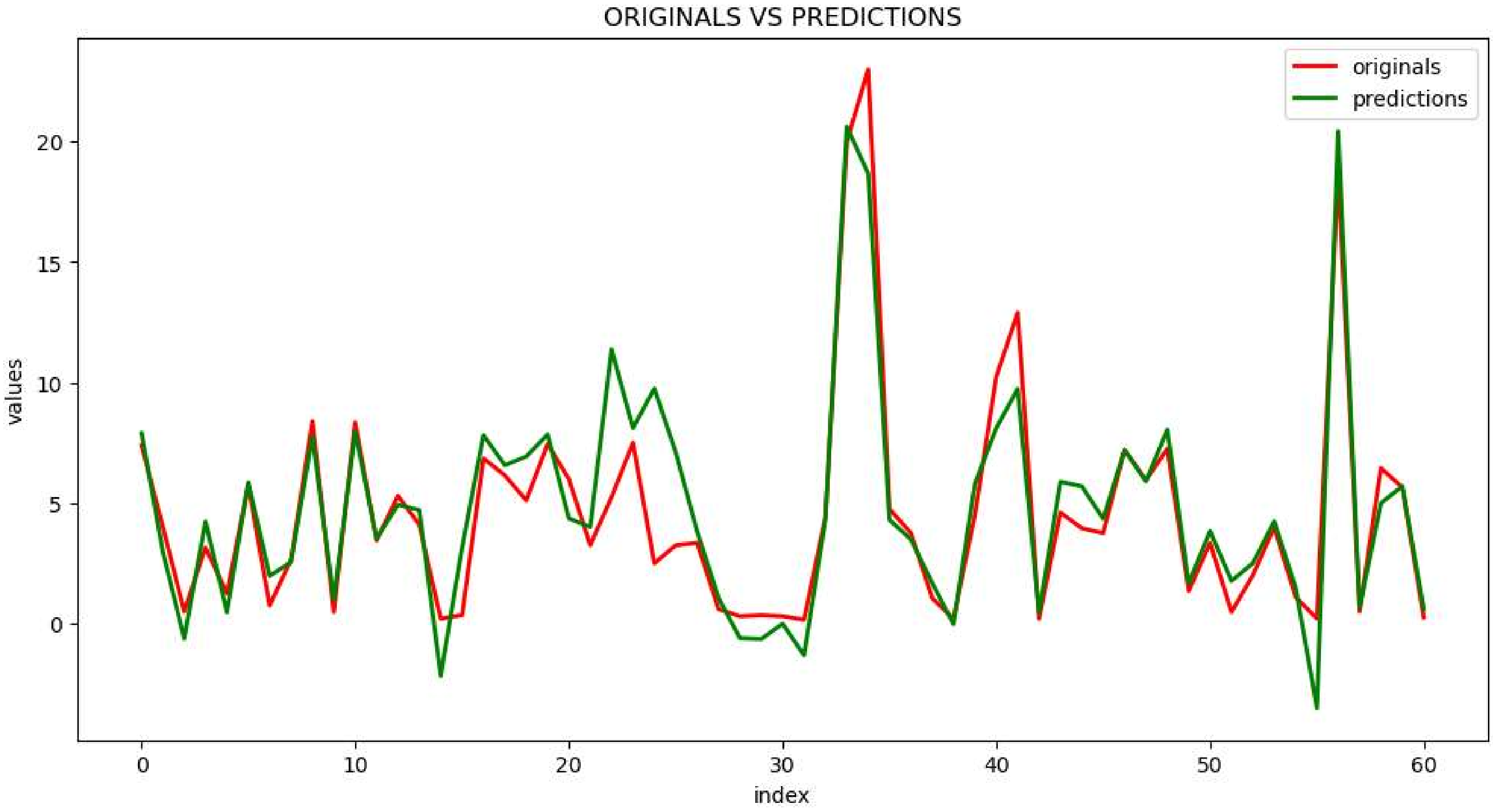
In [69]:

```
def model(algorithm,x_train_,y_train_,x_test_,y_test_):
    algorithm.fit(x_train_,y_train_)
    predicts=algorithm.predict(x_test_)
    prediction=pd.DataFrame(predicts)
    R_2=r2_score(y_test_,prediction)
    cross_val=cross_val_score(algorithm,x_train_,y_train_,cv=cv)

    r_2.append(R_2)
    CV.append(cross_val.mean())

    # Printing results
    print(algorithm,"\n")
    print("r_2 score :",R_2,"\n")
    print("CV scores:",cross_val,"\n")
    print("CV scores mean:",cross_val.mean())


    test_index=y_test_.reset_index()["Selling_Price"]
    ax=test_index.plot(label="originals",figsize=(8,3),linewidth=2,color="r")
    ax=prediction[0].plot(label = "predictions",figsize=(12,6),linewidth=2,color="g")
    plt.legend(loc='upper right')
    plt.title("ORIGINALS VS PREDICTIONS")
    plt.xlabel("index")
    plt.ylabel("values")
    plt.show()
```

In [70]:

```python
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
model(lr,x_train,y_train,x_test,y_test)
```

```
LinearRegression()

r_2 score : 0.8476231240063454

CV scores: [0.89742884 0.88694518 0.83010852 0.81465876 0.75781611]

CV scores mean: 0.8373914789815358
```


ORIGINALS VS PREDICTIONS

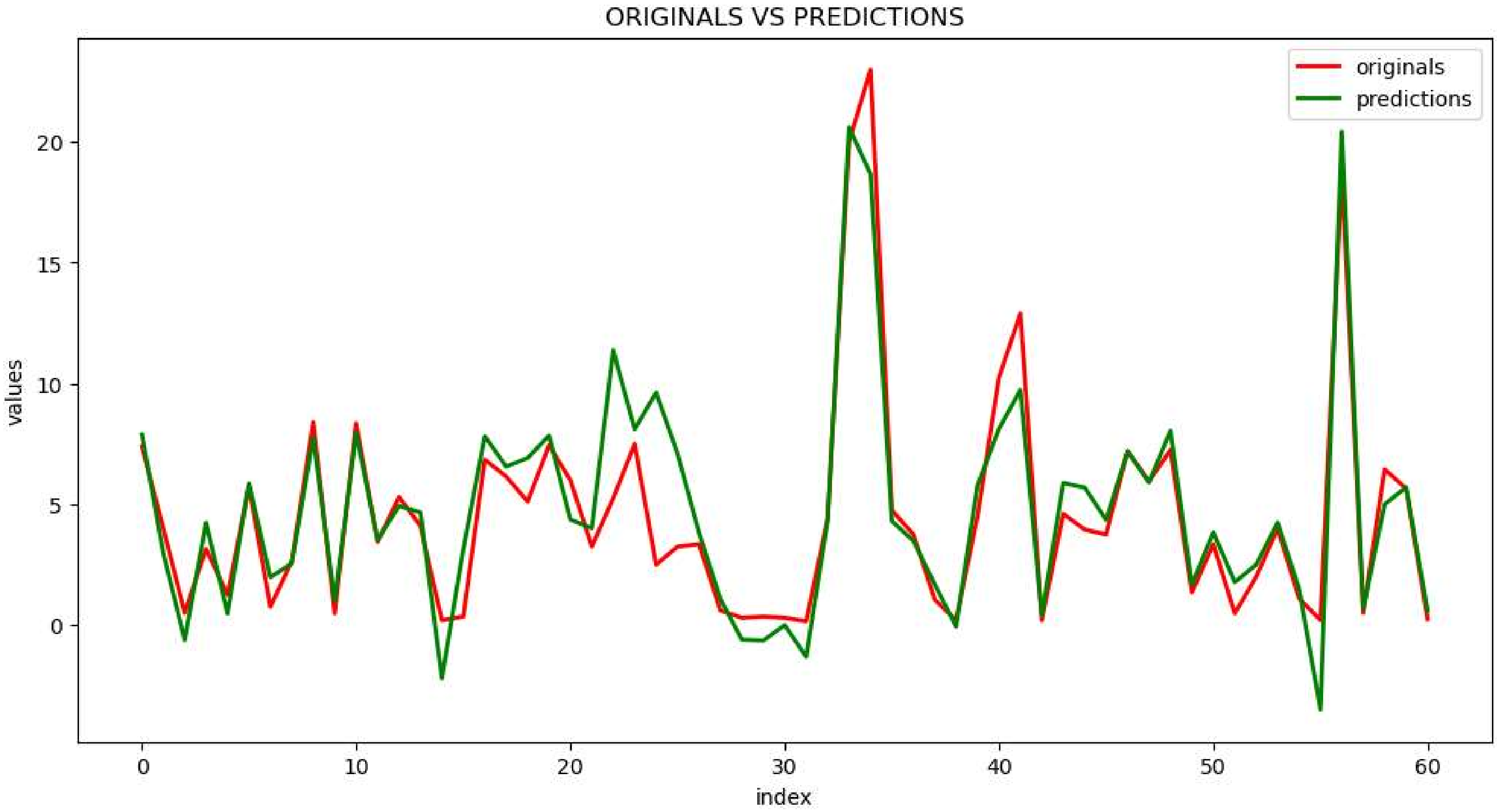
In [71]:

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

alphas = np.logspace(-3,3,num=14) # range for alpha

grid = GridSearchCV(estimator=Lasso(), param_grid=dict(alpha=alphas))
grid.fit(x_train, y_train)

print(grid.best_score_)
print(grid.best_estimator_.alpha)
```

```
0.8372790430791298
0.001
```

In [72]:
```
ls = Lasso(alpha = grid.best_estimator_.alpha)
model(ls,x_train,y_train,x_test,y_test)
```

```
Lasso(alpha=0.001)

r_2 score : 0.8491511644416914

CV scores: [0.89745474 0.88725029 0.83033475 0.81478087 0.75657455]

CV scores mean: 0.8372790430791298
```



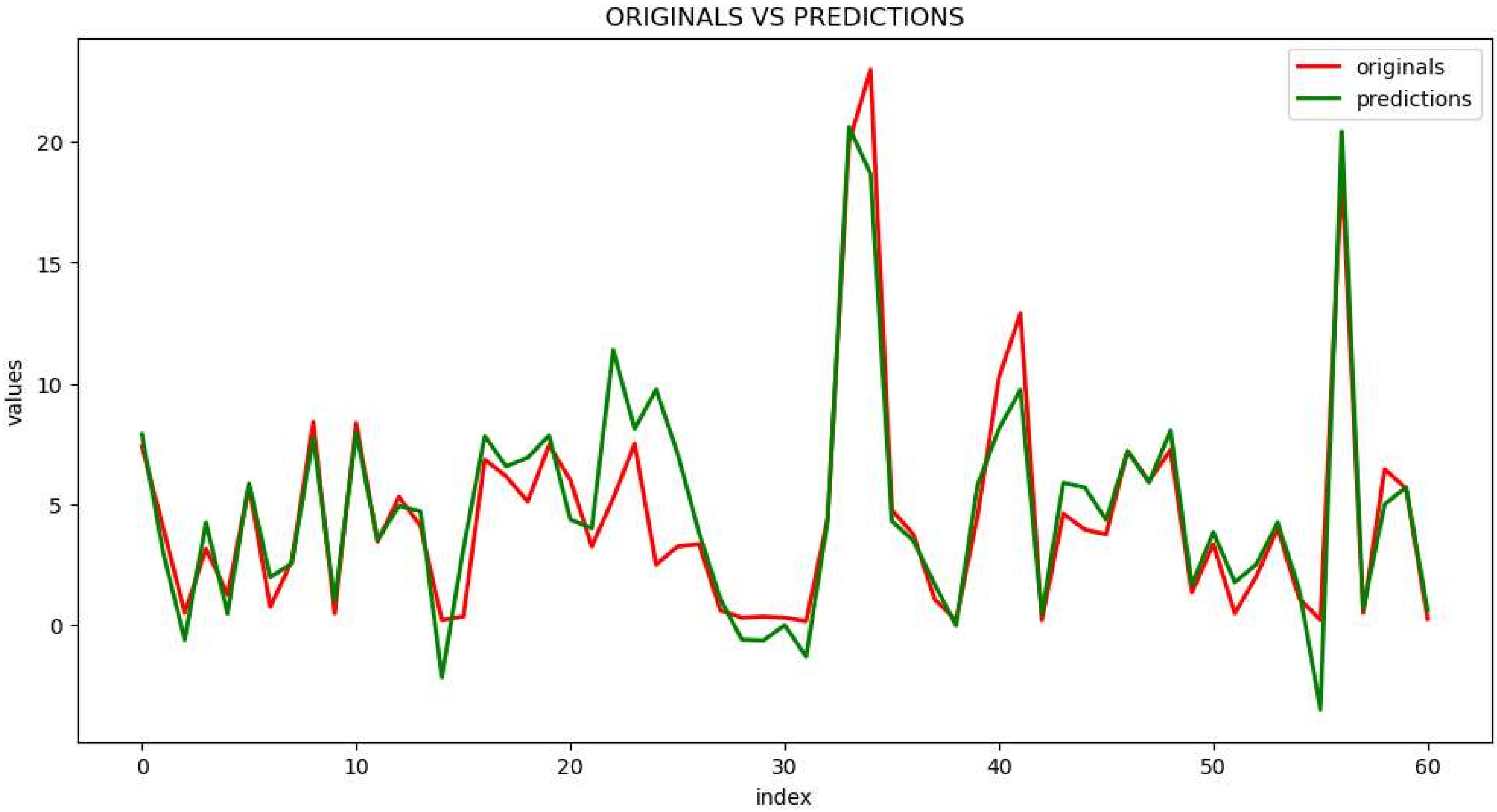
In [73]:

```
from sklearn.linear_model import Ridge

alphas = np.logspace(-3,3,num=14)

grid2 = GridSearchCV(estimator=Ridge(), param_grid=dict(alpha=alphas))
grid2.fit(x_train, y_train)

print(grid2.best_score_)
print(grid2.best_estimator_.alpha)
```

```
0.8374514034792337
0.5878016072274912
```

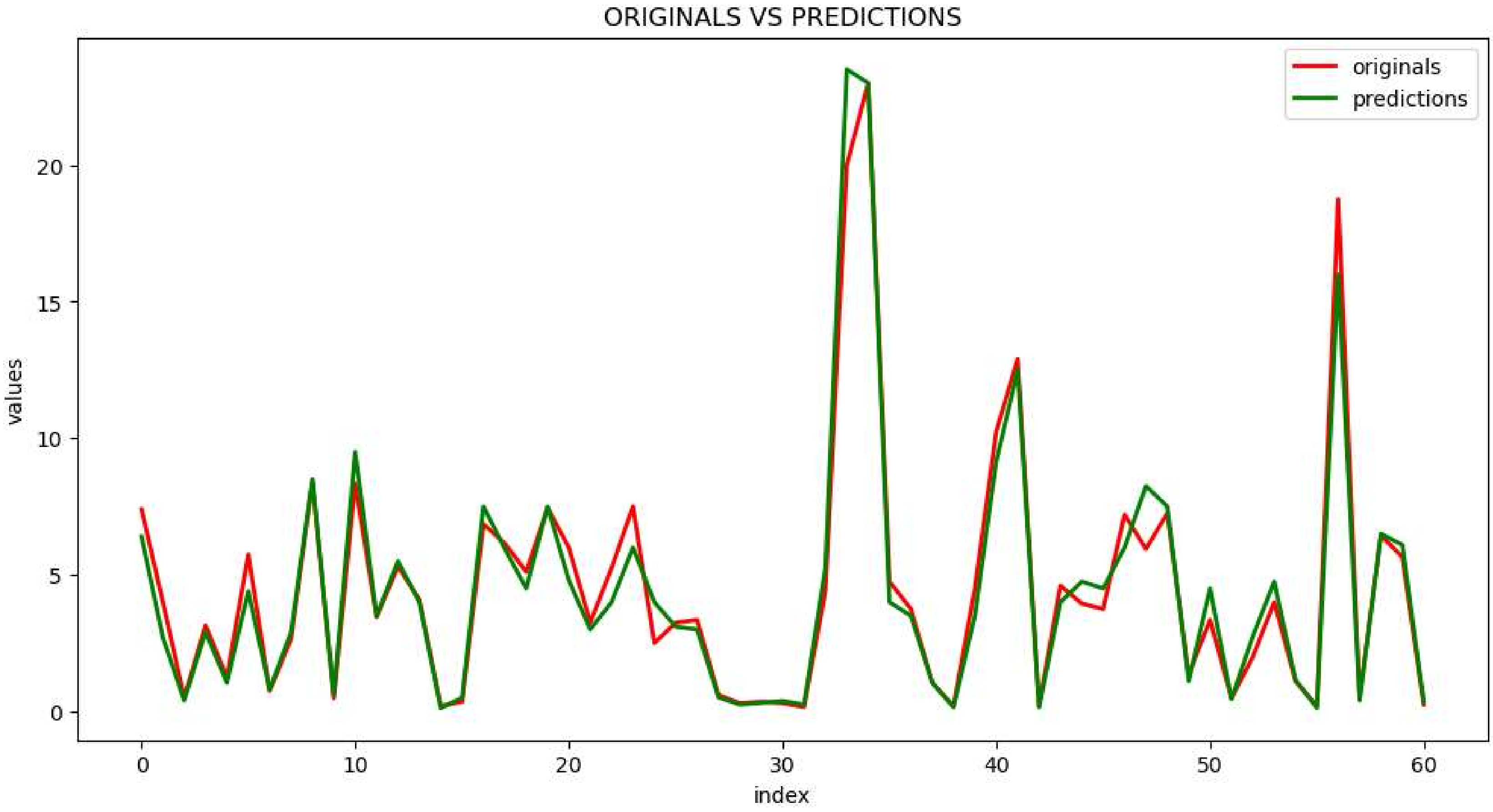
In [74]:
```
ridge = Ridge(alpha = 0.01)
model(ridge,x_train,y_train,x_test,y_test)
```

```
Ridge(alpha=0.01)

r_2 score : 0.8476639137636746

CV scores: [0.89742652 0.88696488 0.83012148 0.81467475 0.75777774]

CV scores mean: 0.8373930740726413
```



```
In [75]: from sklearn.tree import DecisionTreeRegressor
         dtr = DecisionTreeRegressor()
         model(dtr,x_train,y_train,x_test,y_test)
```

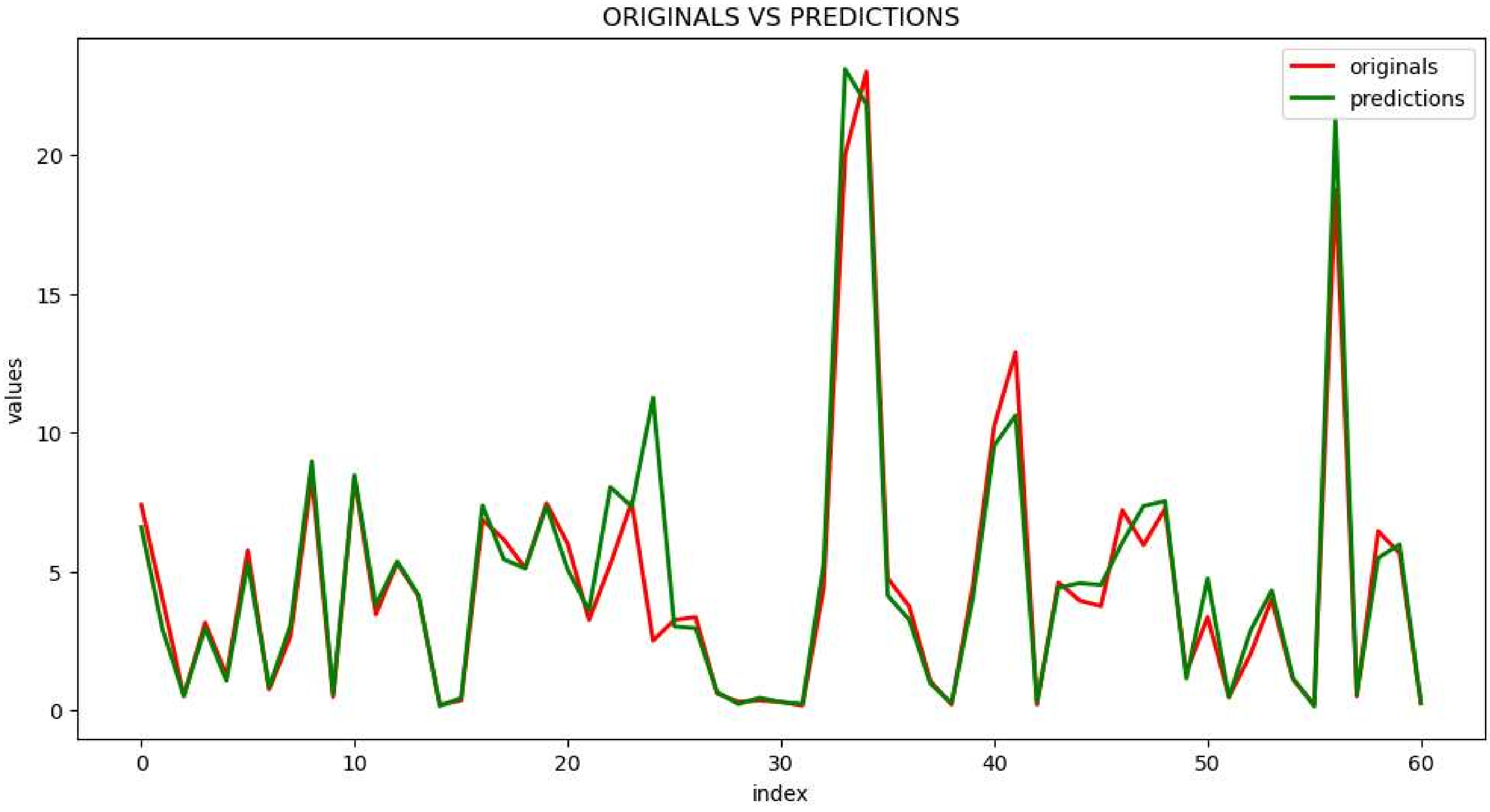
```
DecisionTreeRegressor()

r_2 score : 0.9626763268788968

CV scores: [0.92206829 0.84031671 0.83246236 0.90260766 0.70998591]

CV scores mean: 0.8414881860267462
```



In [76]:

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(n_estimators = 100, random_state = 42)
model(rf,x_train,y_train,x_test,y_test)
```

```
RandomForestRegressor(random_state=42)

r_2 score : 0.9076068142402645

CV scores: [0.93398423 0.96665965 0.85051196 0.93912002 0.7208277 ]

CV scores mean: 0.8822207131397913
```



In [78]:

```
Model = ["LinearRegression", "Lasso", "Ridge", "DecisionTreeRegressor", "RandomForestRegressor"]
results = pd.DataFrame({'Model': Model, 'R Squared': r_2, 'CV score mean': CV})

print(results)
```

```
                   Model  R Squared  CV score mean
0       LinearRegression   0.847623       0.837391
1                  Lasso   0.849151       0.837279
2                  Ridge   0.847664       0.837393
3  DecisionTreeRegressor   0.962676       0.841488
4  RandomForestRegressor   0.907607       0.882221
```

In [ ]:

In [ ]: