②非線形回帰モデル_実装演習

In [4]:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
```

・学習用データの準備

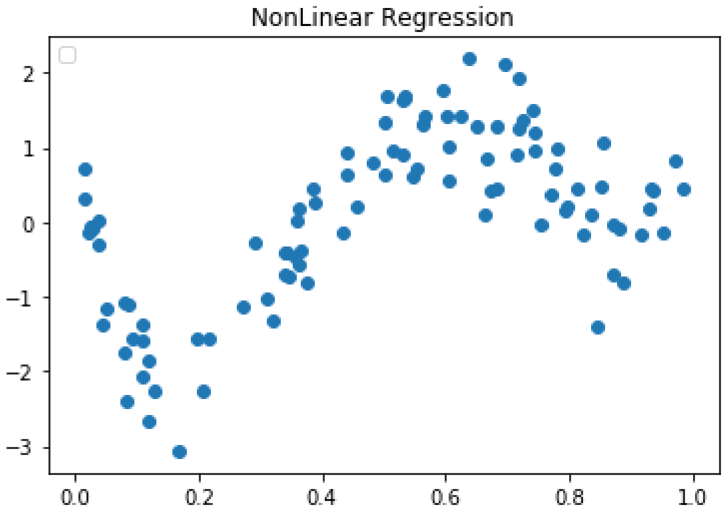
In [5]:

```
n=100

def true_func(x):
    z = 1-48*x+218*x**2-315*x**3+145*x**4
    return z

def linear_func(x):
    z = x
    return z


# 真の関数からノイズを伴うデータを生成

# 真の関数からデータ生成
data = np.random.rand(n).astype(np.float32)
data = np.sort(data)
target = true_func(data)

#  ノイズを加える
noise = 0.5 * np.random.randn(n)
target = target  + noise

# ノイズ付きデータを描画

plt.scatter(data, target)

plt.title('NonLinear Regression')
plt.legend(loc=2)
```

No handles with labels found to put in legend.

Out[5]:

<matplotlib.legend.Legend at 0x2247e8512c8>



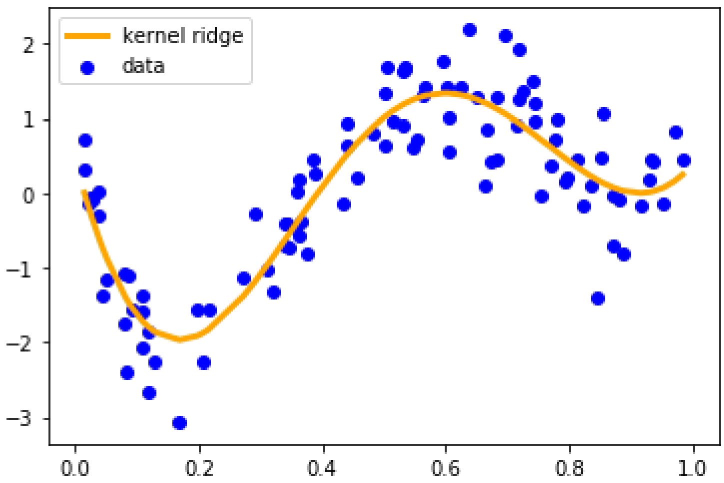
==========

1.Ridge回帰

In [10]:

```
from sklearn.kernel_ridge import KernelRidge

clf = KernelRidge(alpha=0.0002, kernel='rbf')
data = data.reshape(-1,1)
target = target.reshape(-1,1)

clf.fit(data, target)
p_kridge = clf.predict(data)


plt.scatter(data, target, color='blue', label='data')

plt.plot(data, p_kridge, color='orange', linestyle='-', linewidth=3, markersize=6, label='kernel ridge')
plt.legend()

print(clf.score(data, target))
```

0.8077326696710864



・alphaにより、正則化による各特徴量の影響度を制御　→Ridge回帰はL2正則化（係数のオーダーを0に近づける）

==========

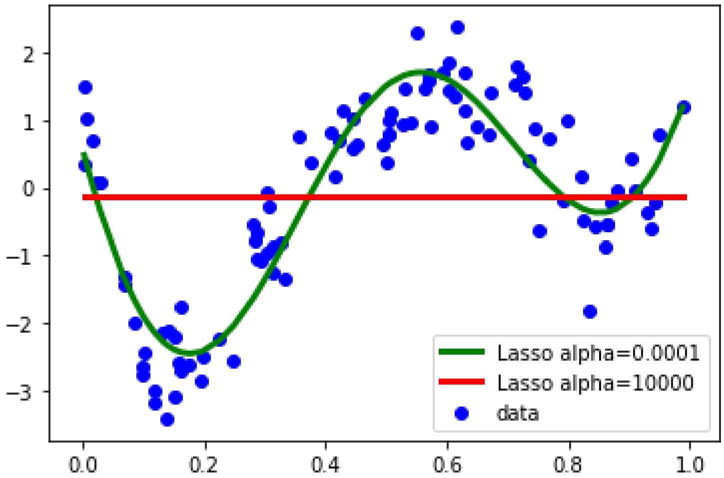
2.Lasso回帰

In [42]:

```
from sklearn.metrics.pairwise import rbf_kernel
from sklearn.linear_model import Lasso

kx = rbf_kernel(X=data, Y=data, gamma=5)
#KX = rbf_kernel(X, x)

#lasso_clf = LinearRegression()
lasso_clf = Lasso(alpha=0.0001, max_iter=1000)
lasso2_clf = Lasso(alpha=10000, max_iter=1000)

lasso_clf.fit(kx, target)
p_lasso = lasso_clf.predict(kx)

lasso2_clf.fit(kx, target)
p_lasso2 = lasso2_clf.predict(kx)

plt.scatter(data, target, color='blue', label='data')

plt.plot(data, p_lasso, color='green', linestyle='-', linewidth=3, markersize=3, label='Lasso al
pha=0.0001')
plt.plot(data, p_lasso2, color='red', linestyle='-', linewidth=3, markersize=3, label='Lasso alp
ha=10000')
plt.legend()

print(lasso_clf.score(kx, target))
```

```
C:¥Users¥Kadoya Toshiki¥anaconda3¥lib¥site-packages¥sklearn¥linear_model¥_coordina
te_descent.py:476: ConvergenceWarning: Objective did not converge. You might want
to increase the number of iterations. Duality gap: 15.826719284057617, tolerance:
0.021877393126487732
  positive)
```

```
0.8321755328782781
```



・alphaにより、正則化による各特徴量の影響度を制御　→Lasso回帰はL1正則化（係数を0にすべきものは0にする）

In [43]:

```
print(lasso_clf.coef_)
```

```
[-3.98471117e+00 -1.67692062e-02 -1.72898933e-01 -9.34408486e-01
 -6.56927526e-01 -6.45445883e-01 -3.63020134e+00 -9.24276263e-02
 -1.49403429e+00 -1.18890727e+00 -3.77671309e-02 -3.21438909e-01
 -1.11430216e+00 -1.80252299e-01 -1.07315016e+00 -3.77791435e-01
 -2.44704336e-01 -7.52643108e-01 -3.36287133e-02 -5.68750679e-01
 -2.06891254e-01 -1.23310603e-01 -9.87765670e-01 -1.29047227e+00
 -2.62833893e-01 -2.01577497e+00 -1.54196441e+00 -1.99464178e+00
 -1.92237452e-01 -1.41402900e-01 -1.41018823e-01 -2.00207323e-01
 -6.38611794e-01 -7.40003437e-02 -1.28243551e-01 -3.00799757e-01
 -8.22383761e-02 -6.25979364e-01 -3.15263629e-01 -8.16765308e-01
 -6.75999045e-01 -5.42335570e-01 -1.11063197e-01 -7.69970641e-02
 -1.09946765e-01 -2.33010411e-01 -2.69265100e-03 -1.05610646e-01
 -7.01718628e-02  2.03505322e-01  3.19895655e-01  4.70564440e-02
  8.00441653e-02  5.90136554e-03  2.02856153e-01  1.02652299e+00
  1.65248066e-01  5.44442534e-01  4.53474283e-01  6.56103969e-01
  2.23822400e-01  8.70606080e-02  1.73243165e-01  7.80304670e-01
  3.40083808e-01  7.65337870e-02  2.71116644e-01  5.97040448e-03
  2.69377142e-01  3.80235836e-02  1.57051682e-02  3.25394154e-01
  1.80170164e-01  2.59489715e-02  2.81355024e-04 -1.42289519e-01
 -2.06858799e-01 -2.20985875e-01 -3.07191372e-01 -3.99078846e-01
 -4.90014941e-01 -1.01162386e+00 -1.40955353e+00 -5.93343616e-01
 -2.03063273e+00 -3.18751186e-01 -8.93124878e-01 -9.69373465e-01
 -1.75127470e+00 -1.37798712e-01 -1.85747772e-01 -6.44570470e-01
 -9.87000883e-01 -3.10650396e+00 -7.54872501e-01 -2.50318146e+00
 -1.14751112e+00 -8.13636422e-01 -1.06570303e+00 -6.20502663e+00]
```

In [44]:

```
print(lasso2_clf.coef_)
```

```
[-0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.
 -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.  -0.
 -0.  -0.  -0.  -0.  -0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
  0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
  0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
  0.   0.   0.   0.   0.   0.   0.   0.   0.   0. ]
```

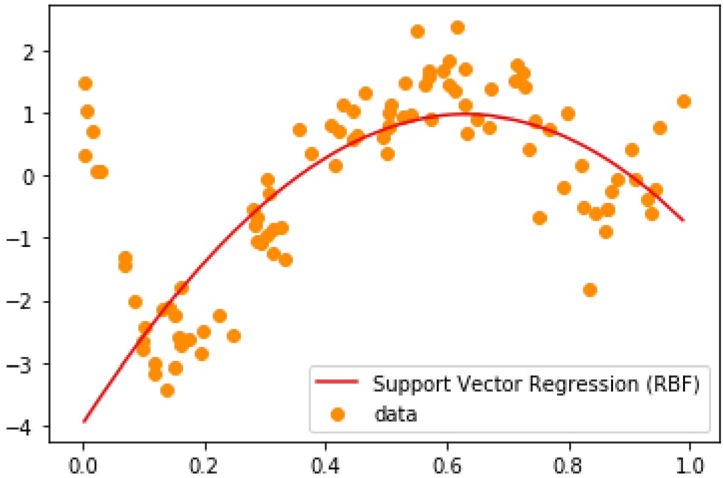
==========

3.サポートベクター回帰

In [34]:

```
from sklearn import model_selection, preprocessing, linear_model, svm

# SVR-rbf
clf_svr = svm.SVR(kernel='rbf', C=1e3, gamma=0.1, epsilon=0.1)

clf_svr.fit(data, target)
y_rbf = clf_svr.fit(data, target).predict(data)


plt.scatter(data, target, color='darkorange', label='data')
plt.plot(data, y_rbf, color='red', label='Support Vector Regression (RBF)')
plt.legend()
plt.show()

print(clf_svr.score(data, target))
```

```
C:¥Users¥Kadoya Toshiki¥anaconda3¥lib¥site-packages¥sklearn¥utils¥validation.py:76
0: DataConversionWarning: A column-vector y was passed when a 1d array was expecte
d. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:¥Users¥Kadoya Toshiki¥anaconda3¥lib¥site-packages¥sklearn¥utils¥validation.py:76
0: DataConversionWarning: A column-vector y was passed when a 1d array was expecte
d. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```



0.20650204236340042

==========

・非線形回帰モデルによる予測を実施。 ・非線形分布のデータを作成し、それに当てはまるモデルを構築。 ・Ridge回帰、Lasso回帰、サポートベクター回帰を使用。 ・Ridge回帰 →L2正則化・・・各特徴量の係数(重み)のオーダーを0に近づける ・Lasso回帰 →L1正則化・・・係数を0にすべきものは0にする→特徴量の削減に寄与 ・ハイパーパラメータの設定値により、結果が変化する。最適なパラメータ値を求める必要がある。

In [ ]: