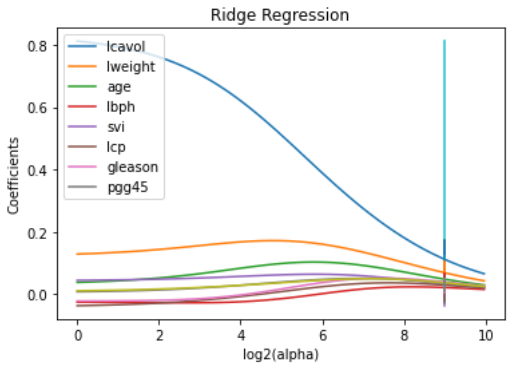
```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Ridge, RidgeCV
from sklearn.kernel_ridge import KernelRidge
import matplotlib.pyplot as plt


train_data = np.loadtxt('/content/prostate.training.txt', skiprows=1)
X_train= train_data[:, :-1]
y_train = train_data[:, -1]

# Read testing data from text file
test_data = np.loadtxt('/content/prostate.testing.txt',skiprows=1)
X_test = test_data[:, :-1]
y_test = test_data[:, -1]



# Method I: Ridge regression with cross-validation
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
alphas = np.logspace(0, 3, 100)
ridge_cv = RidgeCV(alphas=alphas)
ridge_cv.fit(X_train_scaled, y_train)

coefs_ridge = []
df_ridge = []

for alpha in alphas:
    # Method I
    ridge = Ridge(alpha=alpha)
    ridge.fit(X_train_scaled, y_train)
    coefs_ridge.append(ridge.coef_)
    df_ridge.append(np.sum(ridge.coef_ != 0))


# Convert to arrays and plot
coefs_ridge = np.array(coefs_ridge)
df_ridge = np.array(df_ridge)

plt.figure()
plt.plot(np.log2(alphas),coefs_ridge)
plt.plot(df_ridge,coefs_ridge)
plt.xlabel('log2(alpha)')
plt.ylabel('Coefficients')
plt.title('Ridge Regression')
plt.legend(['lcavol', 'lweight', 'age', 'lbph', 'svi', 'lcp', 'gleason', 'pgg45'], loc='upper left')
```

```
<matplotlib.legend.Legend at 0x7f16bc0ea910>
```




```
from sklearn.metrics import mean_squared_error

# Define lambdas
lambdas = np.arange(0, 101, 10)π

# Compute mean squared error for Ridge regression
mse_ridge = []
for l in lambdas:
    ridge = Ridge(alpha=l)
    ridge.fit(X_train_scaled, y_train)
    X_test_scaled = scaler.transform(X_test)
    y_pred = ridge.predict(X_test_scaled)
    mse = mean_squared_error(y_test, y_pred)
```

```
    mse_ridge.append(mse)

# Compute mean squared error for Kernel Ridge regression
mse_kernel_ridge = []
for l in lambdas:
    kernel_ridge = KernelRidge(alpha=l, kernel='rbf', gamma=1.0/X_train.shape[1])
    kernel_ridge.fit(X_train_scaled, y_train)
    X_test_scaled = scaler.transform(X_test)
    y_pred = kernel_ridge.predict(X_test_scaled)
    mse = mean_squared_error(y_test, y_pred)
    mse_kernel_ridge.append(mse)

# Display results in a table
results = pd.DataFrame({'Lambda': lambdas, 'Ridge Regression': mse_ridge, 'Kernel Ridge Regression': mse_kernel_ridge})
print(results)
```

```
    Lambda  Ridge Regression  Kernel Ridge Regression
0        0          0.627530                 4.653667
1       10          0.805692                12.205825
2       20          0.968397                14.708559
3       30          1.116113                16.049389
4       40          1.250760                16.895016
5       50          1.374258                17.479965
6       60          1.488240                17.909752
7       70          1.594043                18.239341
8       80          1.692755                18.500329
9       90          1.785260                18.712225
10     100          1.872285                18.887751
```

```
import matplotlib.pyplot as plt

# Define lambda values
lambdas_method1 = np.arange(0, 10001, 1000)
lambdas_method2 = np.arange(0, 101, 10)

# Define lists to store coefficients and df
coefs_ridge = []
df_ridge = []
coefs_kernel_ridge = []
df_kernel_ridge = []

# Loop over lambda values for method 1
for alpha in lambdas_method1:
    # Ridge Regression
    ridge = Ridge(alpha=alpha)
    ridge.fit(X_train_scaled, y_train)
    coefs_ridge.append(ridge.coef_)
    df_ridge.append(np.sum(ridge.coef_ != 0))

# Loop over lambda values for method 2
for alpha in lambdas_method2:
    # Generalized Ridge Regression with Gaussian kernel
    kernel_ridge = KernelRidge(alpha=alpha, kernel='rbf', gamma=1.0/X_train.shape[1])
    kernel_ridge.fit(X_train_scaled, y_train)
    coefs_kernel_ridge.append(kernel_ridge.dual_coef_.dot(kernel_ridge.X_fit_))
    df_kernel_ridge.append(np.sum(kernel_ridge.dual_coef_ != 0))


# xa = np.arange(0, 5000, 1000)
# Plot the coefficients versus the effective degrees of freedom for both methods
plt.figure(figsize=(10, 6))
plt.plot(lambdas_method1, coefs_ridge)

plt.xlabel('Effective Degrees of Freedom')
plt.ylabel('Coefficients')
plt.title('Ridge Regression')
plt.legend(['lcavol', 'lweight', 'age', 'lbph', 'svi', 'lcp', 'gleason', 'pgg45'], loc='upper left')
plt.figure(figsize=(10, 6))
plt.plot(lambdas_method2, coefs_kernel_ridge)
plt.xlabel('Effective Degrees of Freedom')
plt.ylabel('Coefficients')
plt.title('Generalized Ridge Regression with Gaussian kernel')
plt.legend(['lcavol', 'lweight', 'age', 'lbph', 'svi', 'lcp', 'gleason', 'pgg45'], loc='upper left')
plt.show()
```