School of Electronics and Computer Science
University of Southampton

COMP6245(2020/21): Foundations of Machine Learning Lab Four

Issue	2 November 2020
Deadline	10 November 2020 (10:00 AM)

Objective

- To implement a Radial Basis Functions model on a regression problem and compare its performance with linear regression.
- To use ten-fold cross validation to quote uncertainty in empirical results when comparing the percormances of two machine learning approaches.

Radial Basis Functions (RBF)

The RBF model is given by

$$g(\boldsymbol{x}) = \sum_{j=1}^{M} \lambda_j \, \phi(||\boldsymbol{x} - \boldsymbol{m}_j|| \, / \, \sigma).$$

The model has a nonlinear part (with $\boldsymbol{m}_j$ and σ as parameters within a basis function $\phi(.)$) and a linear part with parameters λ_j. In problems where we can make sensible choices of the nonlinear part, the learning problem reduces to a linear problem in the λ_js.
Constructing an $N \times M$ *design matrix* U with terms $u_{ij} = \phi(||\boldsymbol{x}_i - \boldsymbol{x}_j|| \, / \, \sigma$, and the targets in an $N \times 1$ vector $\boldsymbol{f}$, the least squares solution to estimate λs is similar to linear regression: $\boldsymbol{l} = \left(U^t \, U\right)^{-1} U^t \, \boldsymbol{f}$, where $\boldsymbol{l}$ is an $M \times 1$ vector containing the unknown λs.

1. Study the skeleton implementation of a Gaussian RBF model given in the Appendix.
2. Make the following improvements to the given implementation:
   - Normalize each feature of the input data to have a mean of 0 and standard deviation of 1.
   - The width parameter of the basis functions σ is set to be the distance between two randomly chosen points. Could this sometimes cause an error? Change it to be the average of several pairwise distances.
   - The locations of the M basis functions $\boldsymbol{m}_j$ are set at random points in the input space. Cluster the data using K-means clustering (with $K = M$) and set the basis function locations to the cluster centres.
   - Split the data into training and test sets, estimate the model on the training set and note the test set performance.
3. Implement ten-fold cross validation, where you split the data into ten parts, train on nine tenths of the data and test on the held out tenth set, repeating the process ten times.
4. Display the distributions of test set results for the RBF and Linear Regression Models as boxplots side by side.
5. Compare your implementations with `sklearn`'s inbuilt RBF model.

Report

Describe the work you have done as a short report. Upload a *pdf* file **no longer than two pages** using the ECS handin system: `http://handin.ecs.soton.ac.uk`. Please use LaTeX to typeset your report. Please make sure your name and email are included.

Appendix: Skeleton Implementation of RBF

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

from sklearn import datasets
from sklearn.linear_model import LinearRegression

diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target

def gaussian(x, u, sigma):
    return(np.exp(-0.5 * np.linalg.norm(x-u) / sigma))

N, p = X.shape
print(N, p)

# Space for design matrix
#
M = 200
U = np.zeros((N,M))

# Basis function locations at random
#
C = np.random.randn(M,p)

# Basis function range as distance between two random data
#
x1 = X[np.floor(np.random.rand()*N).astype(int),:]
x2 = X[np.floor(np.random.rand()*N).astype(int),:]
sigma = np.linalg.norm(x1-x2)

# Construct the design matrix
#
for i in range(N):
    for j in range(M):
        U[i,j] = gaussian(X[i,:], C[j,:], sigma)

# Pseudo inverse solution for linear part
#
l = np.linalg.inv(U.T @ U) @ U.T @ y

# Predicted values on training data
#
yh = U @ l
fig, ax = plt.subplots(figsize=(3,3))
ax.scatter(y, yh, c='m', s=3)
ax.grid(True)
ax.set_title("Training Set", fontsize=14)
ax.set_xlabel("True Target", fontsize=12)
ax.set_ylabel("Prediction", fontsize=12)
```