Machine Learning Homework 3 Report

Kasturi Chandra Shekhar
UTA-ID:1001825454

The University of Texas at Arlington
Computer and Information Sciences

CSE 6363 Machine Learning
Prof.Won Hwa Kim

Late Submission
Using one of my Late submission days

Problem 1

Kernel Density Estimation

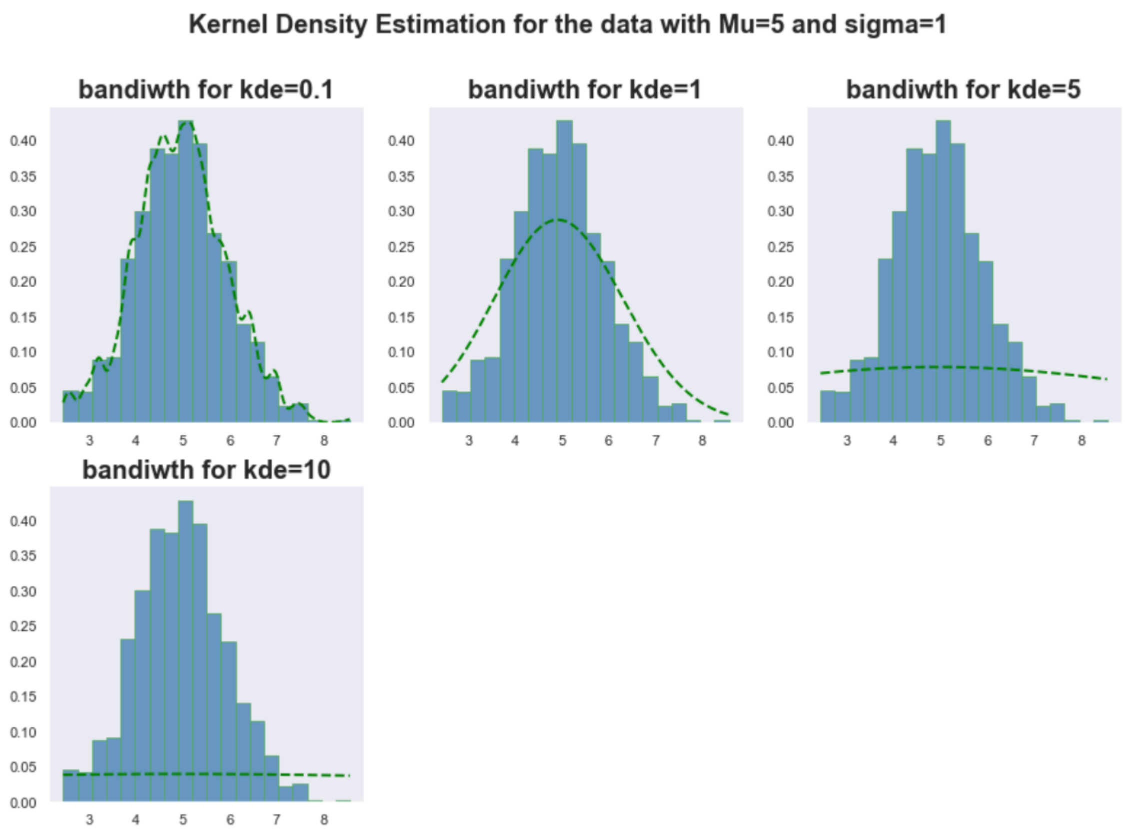
Kernel density estimation (KDE) is a non-parametric way to estimate the probability density function of a random variable. Kernel density estimation is a fundamental data smoothing problem.Given a univariate independent and identically distributed sample drawn from some distribution with an unknown density f at any given point x. We are interested in estimating the shape of this function f. Its kernel density estimator is

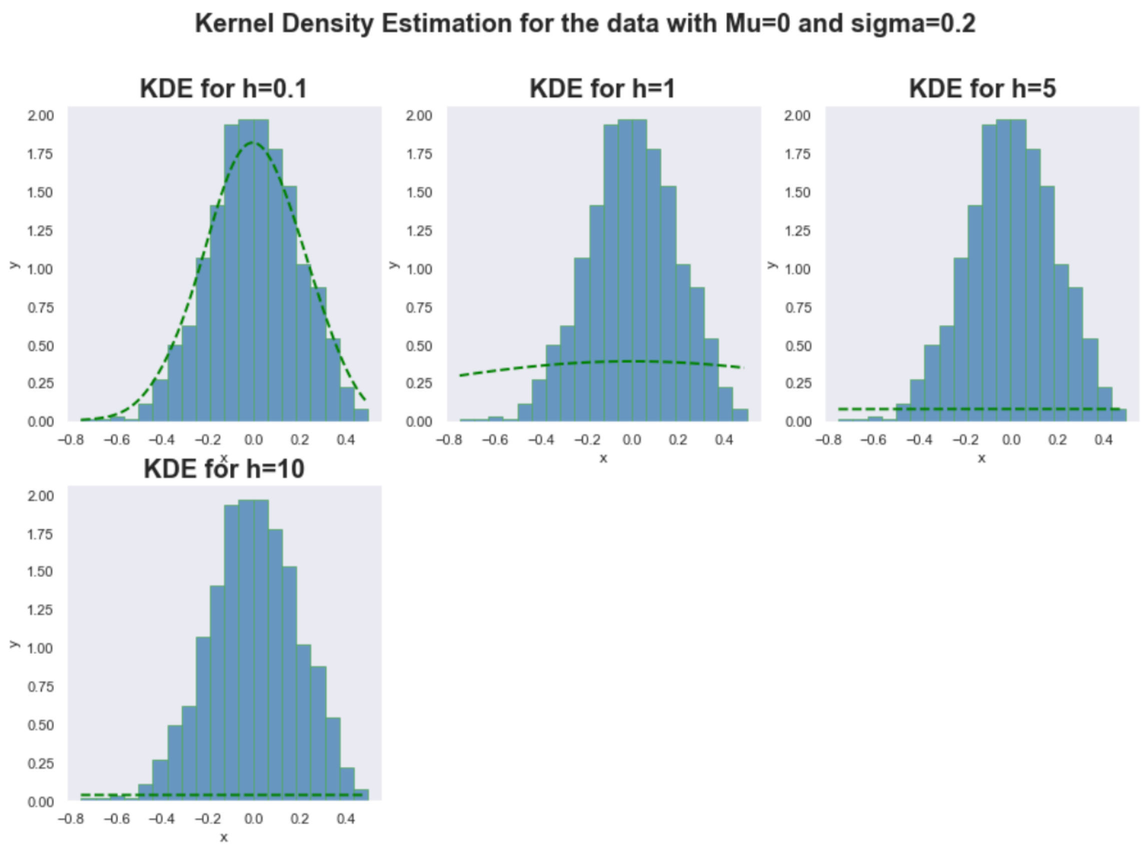
$$\widehat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\Big(\frac{x - x_i}{h}\Big),$$

where K is the kernel — a non-negative function — and h $>$ 0 is a smoothing parameter called the bandwidth. A kernel with subscript h is called the scaled kernel and defined as Kh(x) = 1/h K(x/h). Intuitively one wants to choose h as small as the data will allow; however, there is always a trade-off between the bias of the estimator and its variance.

The implementation of the function $mkde(X, smoothing_function, h = bandwidth)$ involves three parameters to call.The first parameter is the generated univariate data using the values of $mu = 5andsigma = 1$, the second parameter is the kernel smoothing function ,the third parameter is bandwidth values=0.1,1,5,10 that affects how “smooth” the resulting curve is.The KDE is calculated by weighting the distances of all the data points we’ve seen for each location if we’ve seen more points nearby, the estimate is higher, indicating that probability of seeing a point at that location and a gaussian curve is used to indicate how the point distances are weighted which is calculated using the $gaussianpdf(X_gauss, h = bandwidth)$, and is called the kernel function. The points are colored according to this function.Changing the bandwidth changes the shape of the kernel: a lower bandwidth means only points very close to the current position are given any weight, which leads to the estimate looking squiggly; a higher bandwidth means a shallow kernel where distant points can contribute.The density of the generated is represented using the histograms and the gaussian curves is represented using green dashed line.

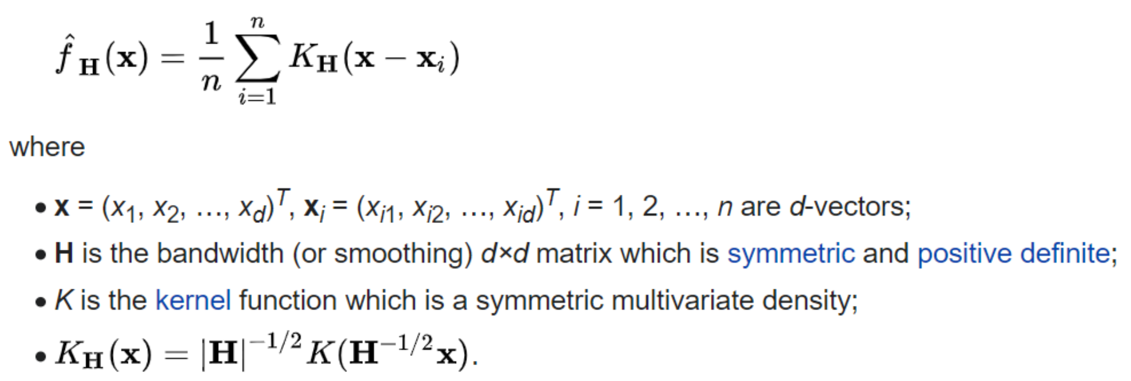
The kernel density estimations for two different datasets with the given bandwidths are shown in the figures.As stated earlier the density curves for the lower bandwidths are shabbier when compared to the higher bandwidths because a small bandwidth(h=0.1) only considers the closest values so the estimation is close to the data. A large bandwidth(h=5) considers more points and gives a smoother estimation.


Kernel Density Estimation for the data with Mu=5 and sigma=1
bandiwth for kde=0.1
bandiwth for kde=1
bandiwth for kde=5
bandiwth for kde=10


Kernel Density Estimation for the data with Mu=0 and sigma=0.2
KDE for h=0.1
KDE for h=1
KDE for h=5
KDE for h=10
y
x

Kernel Density Estimation for 2D Gaussian Random data

For the multivariate kernel denisty estimation the goal of density estimation is to take a finite sample of data and to make inferences about the underlying probability density function everywhere, including where no data are observed. In kernel density estimation, the contribution of each data point is smoothed out from a single point into a region of space surrounding it. Aggregating the individually smoothed contributions gives an overall picture of the structure of the data and its density function. In the details to follow, we show that this approach leads to a reasonable estimate of the underlying density function.Let x1, x2, ..., xn be a sample of d-variate random vectors drawn from a common distribution described by the density function f. The kernel density estimate is defined to be



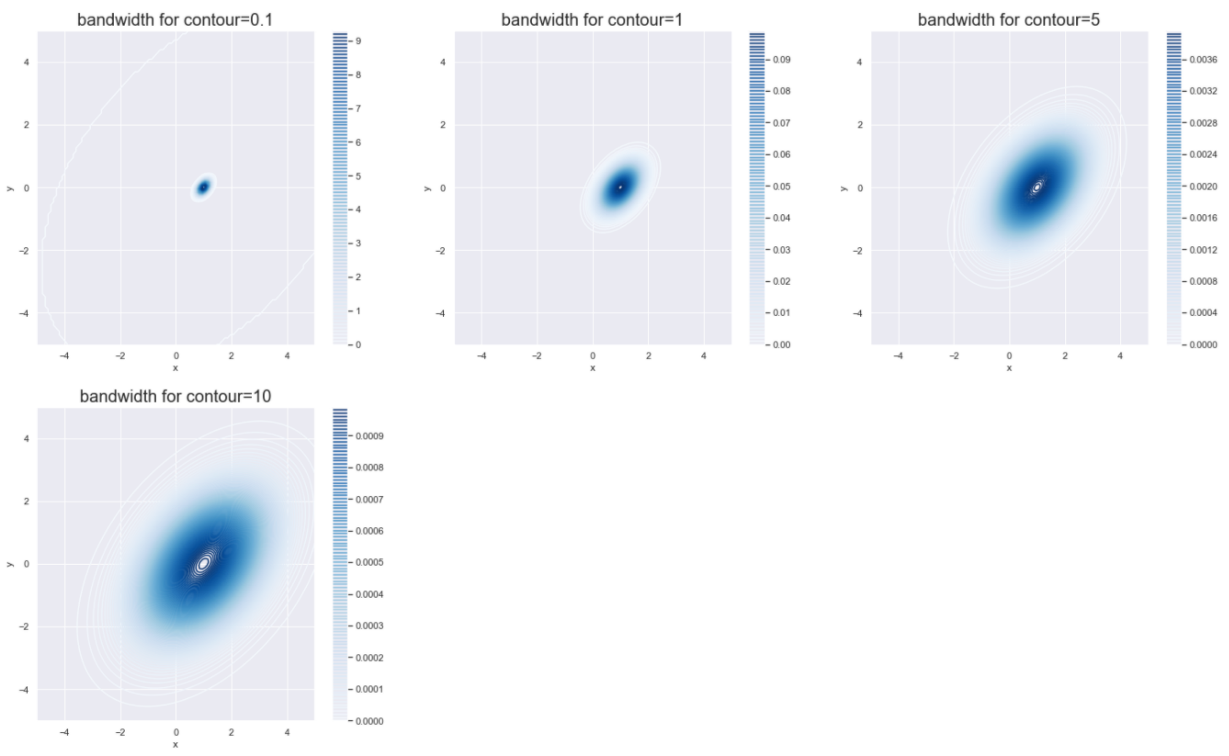
$$\hat{f}_{\mathbf{H}}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i)$$

where

- $\mathbf{x} = (x_1, x_2, \ldots, x_d)^T$, $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id})^T$, $i = 1, 2, \ldots, n$ are d-vectors;
- **H** is the bandwidth (or smoothing) $d \times d$ matrix which is symmetric and positive definite;
- K is the kernel function which is a symmetric multivariate density;
- $K_{\mathbf{H}}(\mathbf{x}) = |\mathbf{H}|^{-1/2} K(\mathbf{H}^{-1/2}\mathbf{x})$.

The implementation of the bi-variate kernel density function is done using $mykde2d(x, d, mu, cov, h)$.The smoothing function is used to generate a contour for the 2-Dimensional representation of the kernel density estimation.The data is generated with the given values of mu and sigma with N=500 each and the two datasets are concatenated to produce the dataset X which is used in the function call if the mykde2d().The contours obtained are as follows when using the function contourkde(mu, cov, d,h) for the different bandwidths representing that with higher bandwidths the expanse of the effect of data close to the actual data is more when compared to a lower bandwidth.

Contour plots for the kde-2D



Problem 2

Principal Component Analysis - Dimensionality Reduction

In machine learning, **"dimensionality"** simply refers to the number of features or input variables in your dataset.While the performance of any machine learning model increases if we add additional features/dimensions, at some point a further insertion leads to performance degradation that is when the number of features is very large commensurate with the number of observations in your dataset, several linear algorithms strive hard to train efficient models. This is called the **"Curse of Dimensionality"**.Dimensionality reduction is a set of techniques that studies how to reduce the size of data by preserving the peculiar features by eliminating the curse of dimensionality playing an important role in the classification and clustering problems.

Principal Component Analysis is a linear technique for dimensionality reduction performing a linear mapping of data to a lower-dimensional space by maximizing the variance of the data in the lower-dimension.It is obtained by computing the covariance and the eigenvectors of this covariance matrix.Later these largest eigen-vectors which correspond to the eigen-values(principal components) can be used to construct the large fraction of the variance of the original data.

Principal Components Analysis (PCA)

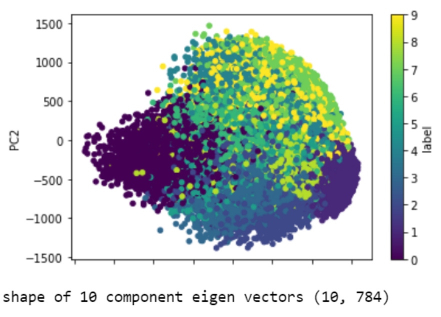
1. Compute the sample covariance matrix $\widehat{\Sigma} = n^{-1}\sum_{i=1}^{n}(X_i - \overline{X}_n)(X_i - \overline{X}_n)^T$.
2. Compute the eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots$ and eigenvectors $e_1, e_2, \ldots,$ of $\widehat{\Sigma}$.
3. Choose a dimension k.
4. Define the dimension reduced data $Z_i = T_k(X_i) = \overline{X} + \sum_{j=1}^{k}\beta_{ij}e_j$ where $\beta_{ij} = \langle X_i - \overline{X}, e_j\rangle$.

In this assignment Principal Component Analysis is implemented using $myPCA(X, k)$ where X represents the MNIST data generated and k representing the number of principal components to be returned by the function.The function

$$myPCA(X, k)$$

is implemented by calculating the mean centered data thereby calculating the covariance of the reduced mean matrix of the MNIST data and computing the Eigenvalues, Eigenvectors for the covariance matrix.The obtained Eigenvalues and Eigenvectors are sorted to obtain the eigenvalues which maximise the variance in the lower dimensional space.Returning the first k principal component scores.Finally, transform the data by having a dot product between the Transpose of the Eigenvector subset and the Transpose of the mean-centered data. By transposing the outcome of the dot product, the result we get is the data reduced to lower dimensions from higher dimensions.The PCs are formed in such a way that the first Principal Component (PC1) explains more variance in original data compared to PC2. Likewise, PC2 explains more than PC3, and so on

The scatter plot obtained by projecting the data onto 2-D space with both the axes as PC scores.There is a huge degree of overlap, the points belonging to same category are not distinctly clustered and region bound. This proves that the data captured in the first two PCs is not informative enough to discriminate the categories from each other and requires more PC's to discriminate the data efficiently.



```
shape of 10 component eigen vectors (10, 784)
```

Figure 1: Scatter plot using first two PC's

The 10 PC visiualisation is obtained by sending the MNIST data to $myPCA(X, k) with k = 10$.The visualization of 10 PC's as 28*28 image obtained interesting shapes by which we can recognise the class-label or the hand-written digits.By reducing the dimensionality of the original data the important features which are peculiar to each class label is identfied and learned to recognise the class-label efficiently.

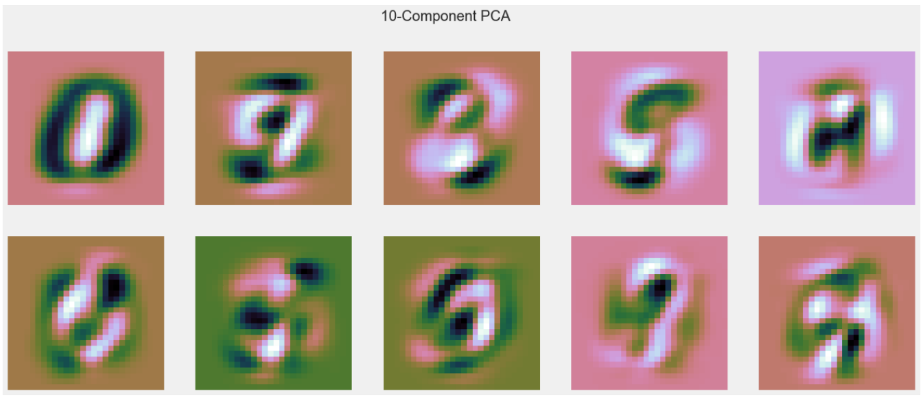


Figure 2: 10 Component PCA as 28*28 image

Later we trained the raw MNIST data using Sequential model of the Logistic Regression using softmax as the activation function.The input to this sequential model of the MNIST data has undergone reshaping,separation as test and training data.Then this reshaped data is sent to the sequential classfier with ten output nodes the shape of the output tensor is (10,784) each node representing their respective class-labels.Then the raw MNIST data is again sent to $myPCA(X, k)$ with k value as 30 and again trained with the classifier defined above.The training time of both classifiers are calculated and the training time for training the 30 PC data is found to be faster.However, the accuracy obtained with this training was not quite impressive when compared to training the raw data with the Linear Regression.The performance on the testing data was found to be worse when compared to the training data (drop of accuracy from 92 to 90 percent).The visualisation on the trained classifier of the Logistic regression is shown in Fig:3.

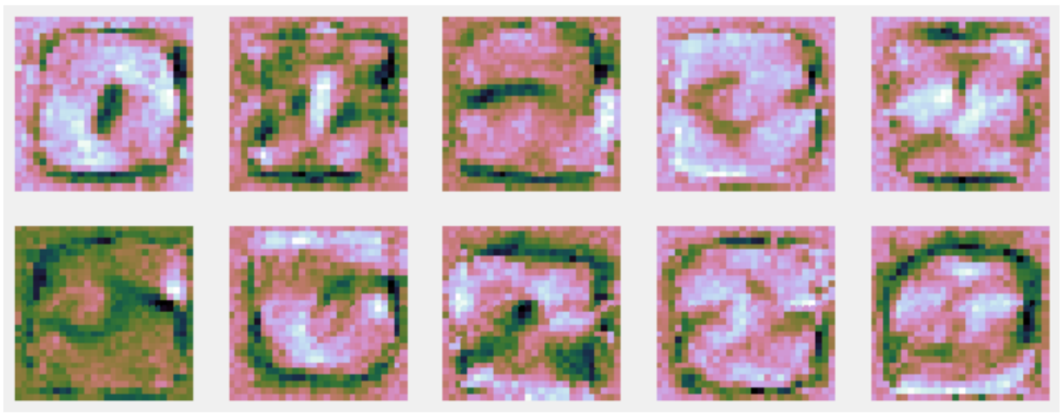


Figure 3: Visualization of Pre-trained Logistic regression

The output for the pre-trained Logistic regression is obtained by visualizing it as 28*28 pixel image.It is clearly visible that it represents the class-labels of the MNIST data.By reducing the dimenisonality the important features of the data are learned by the classifier to differentiate between different representations of the class-labels increasing its performance in efficiently identifying the digits of the MNIST data.

References

Problem 1

- `https://www.homeworkhelponline.net/blog/math/tutorial-kde`
- `https://stackabuse.com/kernel-density-estimation-in-python-using-scikit-learn/`

Problem 2

- `https://www.askpython.com/python/examples/principal-component-analysis`
- `https://mylearningsinaiml.wordpress.com/2018/09/10/pca-visualization-mnist-data/`