Abhijit Anand Deshpande

UTA ID: 1001677938

Machine Learning 6363

Assignment No 3

Problem 1

Kernel Density Function

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 where inferences about the population are made, based on a finite data sample. In some fields such as signal processing and econometrics it is also termed the Parzen–Rosenblatt window method, after Emanuel Parzen and Murray Rosenblatt, who are usually credited with independently creating it in its current form.[1][2] One of the famous applications of kernel density estimation is in estimating the class-conditional marginal densities of data when using a naive Bayes classifier, which can improve its prediction accuracy.

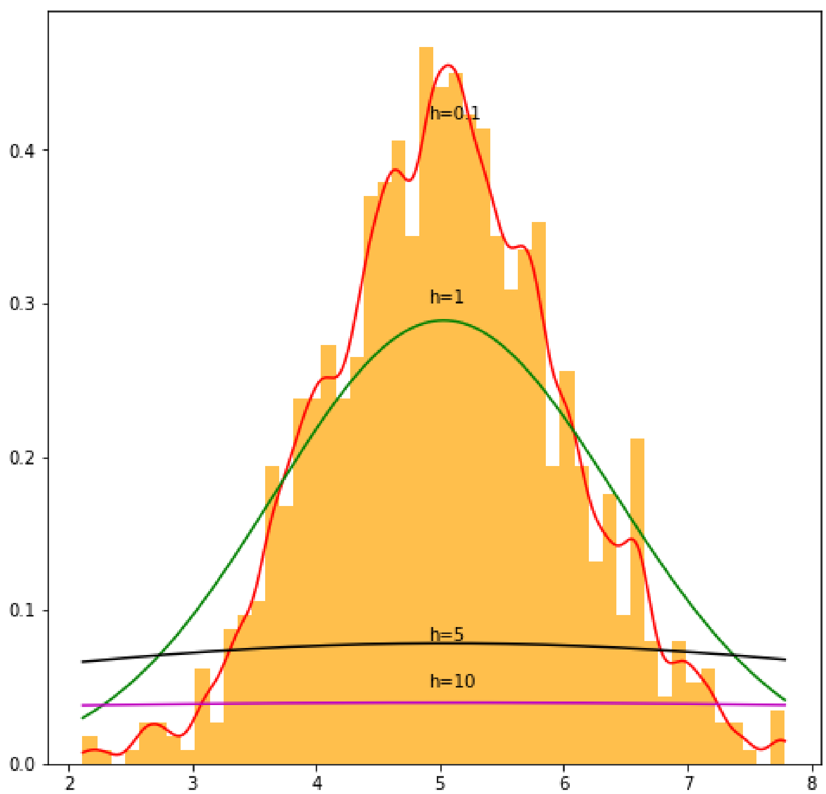
1 (a)

Gaussian random data with μ1= 5 andσ1= 1

mykde function returns the kernel densities for data with μ1= 5 andσ1= 1

for h= [0.1,1,5,10]

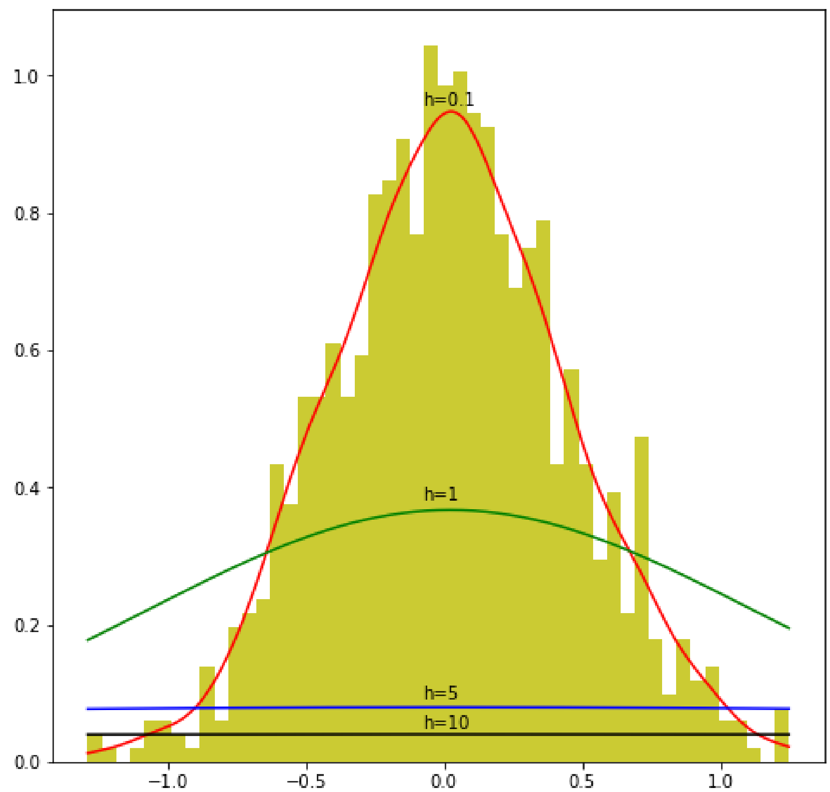
KDE is



1 (b)

Gaussian random data with μ2= 0   andσ2= 0.2

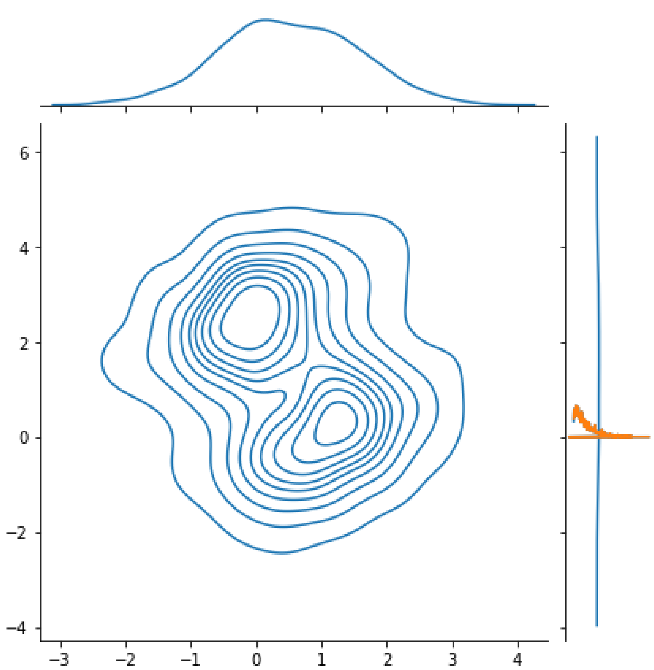
for h= [0.1,1,5,10]



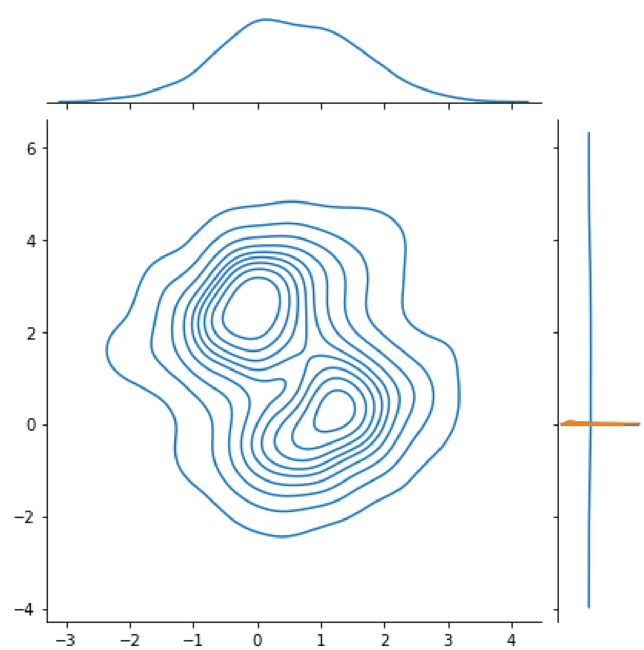
1 (2)

μ1= [1, 0], μ2= [0, 2.5], Σ1= [[0.9 0.4],[0.4 0.9]], Σ2=[[0.9 0.4],[0.4 0.9]]

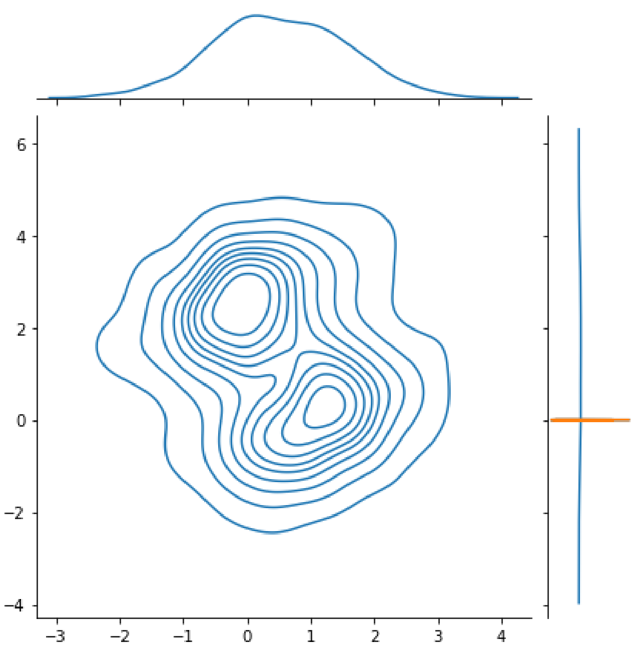
For h=0.1



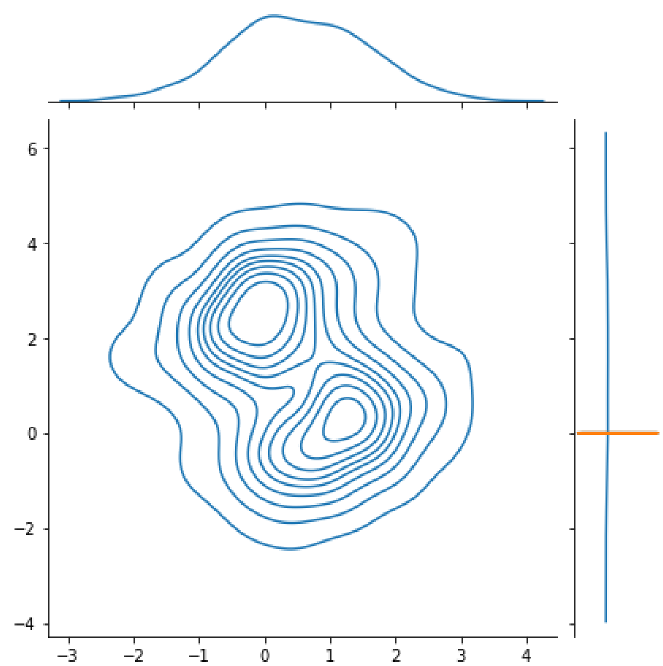
For h=1



For h=5



For h=10



Problem 2

What is PCA?

The principal components of a collection of points in a real *p*-space are a sequence of direction vectors, where the vector is the direction of a line that best fits the data while being orthogonal to the first vectors. Here, a best-fitting line is defined as one that minimizes the average squared distance from the points to the line. These directions constitute an orthonormal basis in which different individual dimensions of the data are linearly uncorrelated. Principal component analysis (PCA) is the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest.

PCA is used in exploratory data analysis and for making predictive models. It is commonly used for dimensionality reduction by projecting each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible. The first principal component can equivalently be defined as a direction that maximizes the variance of the projected data. The principal component can be taken as a direction orthogonal to the first principal components that maximizes the variance of the projected data.

P (1)

When mypca function used on 2014_Financial_Data, it took total 84 ms to reduce the data from 222D to 20D.

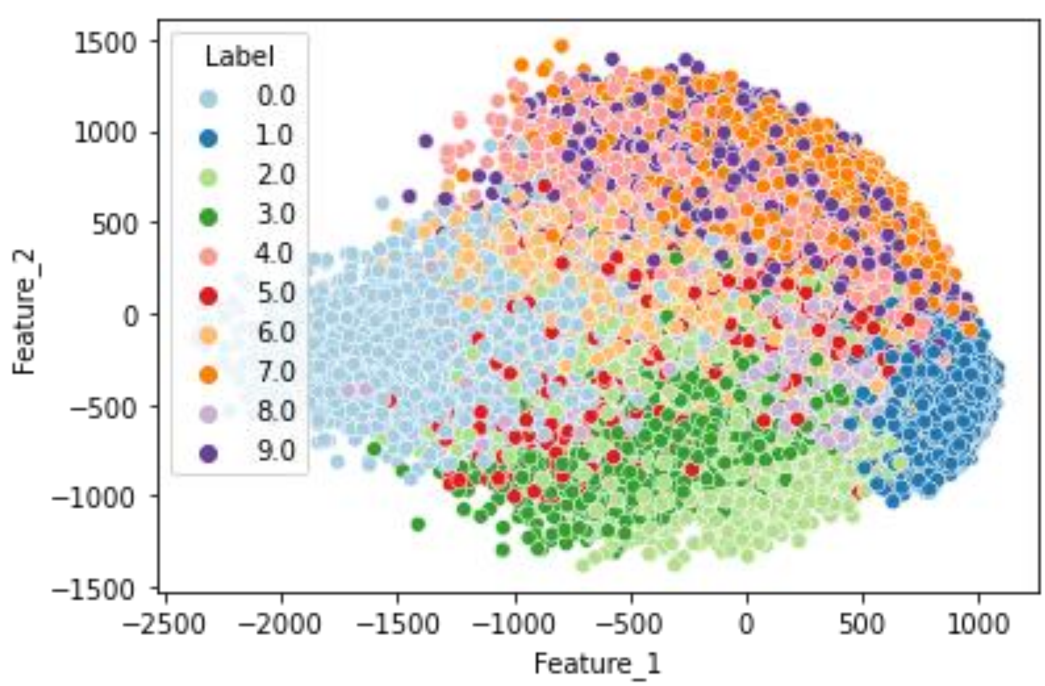
P (2)

Using the trick to reduce the time, same function used on mnist_train dataset to reduce data from 784D to 20D and it took 4.28 s which is quite big compare to previous dataset as it has 60000 samples and 784 features.

2 (3)

When mnist_train dataset reduced to 2D and plot scatter plot of it, it shows the following figure.

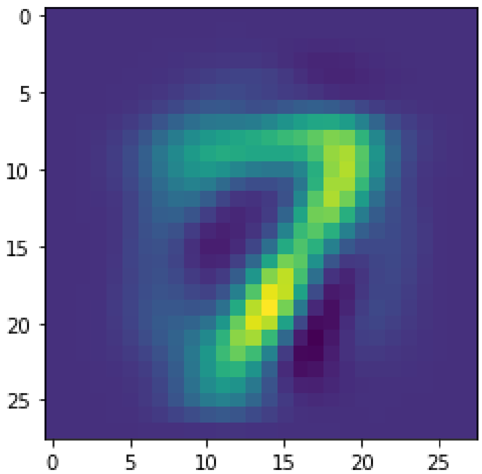
It shows the labels from 0 to 9 in 2D.

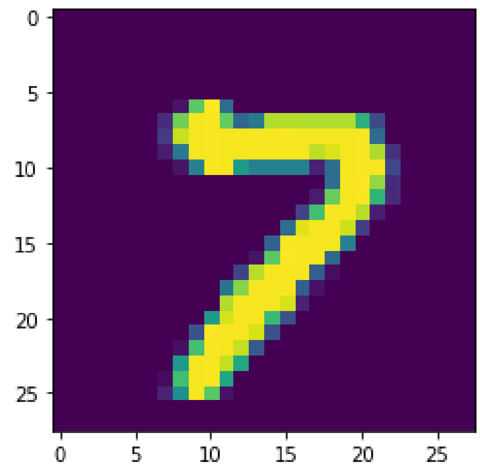


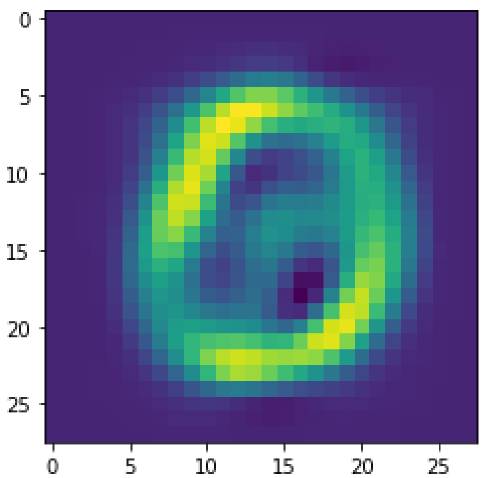
2 (4)

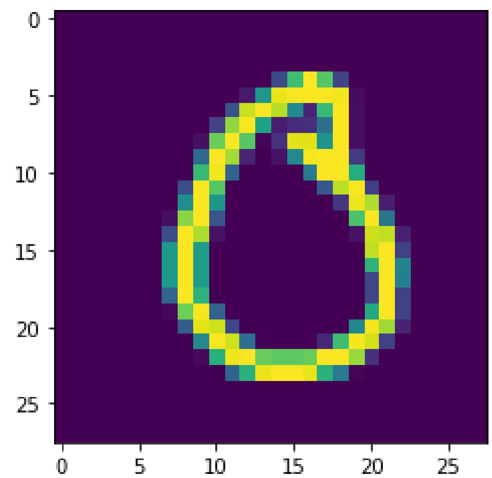
mnist_train dataset reduced to 10D

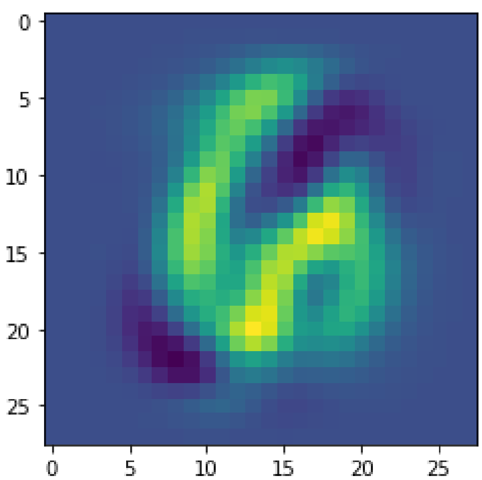
Reconstructed Image with 10D | **Original Image with 784D**

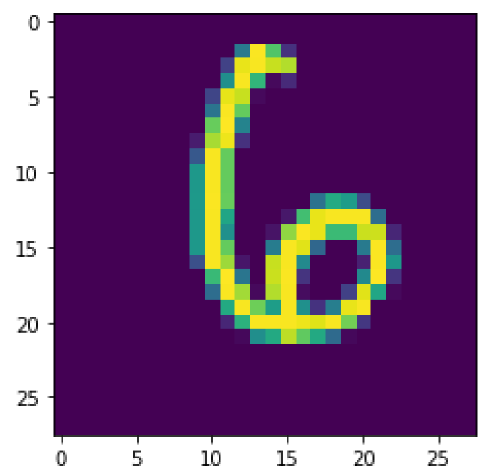










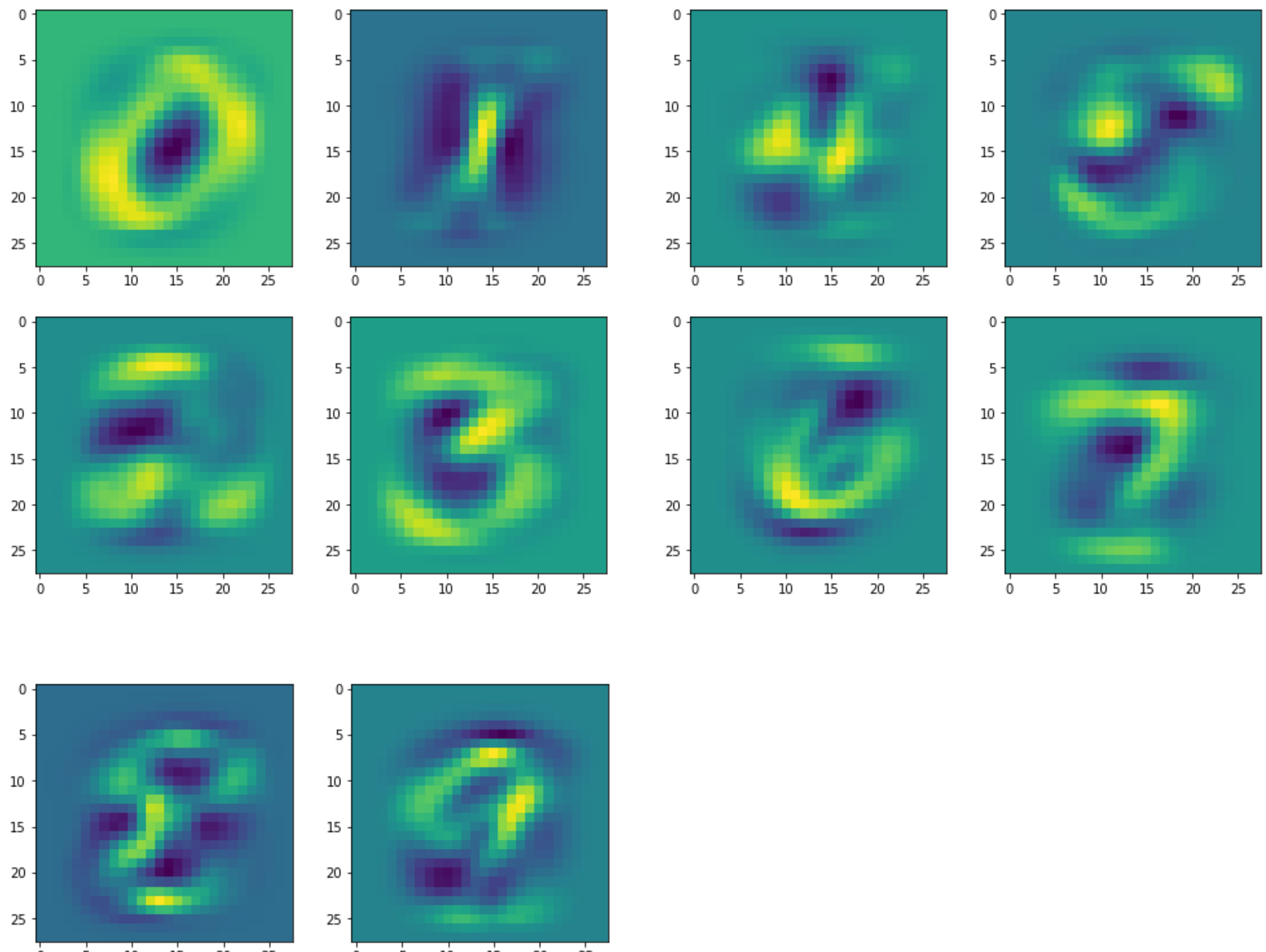


2 (5)

Using Multiple Logistic Regression on MNIST dataset with all features, it took Training Time: 2.9
53159809112549 s to train the data while when used reduced data with 30D, it used Training Ti
me:
0.4307129383087158 s to train the dataset. It shows that algorithm took very less time with red
uced
dimensions to train the data and it performed well on reduced data as well with around same ac
curacy.

2 (6)

Logistic Regression model using the raw data. Each output node has 784 weights associated
with it. The coefficients are reshaped to 28×28. These images are quite similar to reconstructed
images with 10D. Neural networks learn weights from given dataset. This looks more accurate a
nd can be recognized easily that p 2 (4).



References

1. **file:///Users/laptopuser/Downloads/Hand-on-ML.pdf**

2. **https://towardsdatascience.com/simple-example-of-2d-density-plots-in-python-83b83b934f67**

3. **https://medium.com/@mukul.mschauhan/data-visualisation-using-seaborn-464b7c0e5122**

4. **https://medium.com/analytics-vidhya/kernel-density-estimation-kernel-construction-and-bandwidth-optimization-using-maximum-b1dfce127073**

5. **https://rstudio-pubs-static.s3.amazonaws.com/238698_f5c485e2a4f2441dbc9a52ebda0fe8c0.html**