Assignment-5
Siddharth Saklecha
201505570

1.DATASET-IRIS-DATA

1.1 Algorithm: Principal Component Analysis(PCA)

a) Pc1 vs Pc2

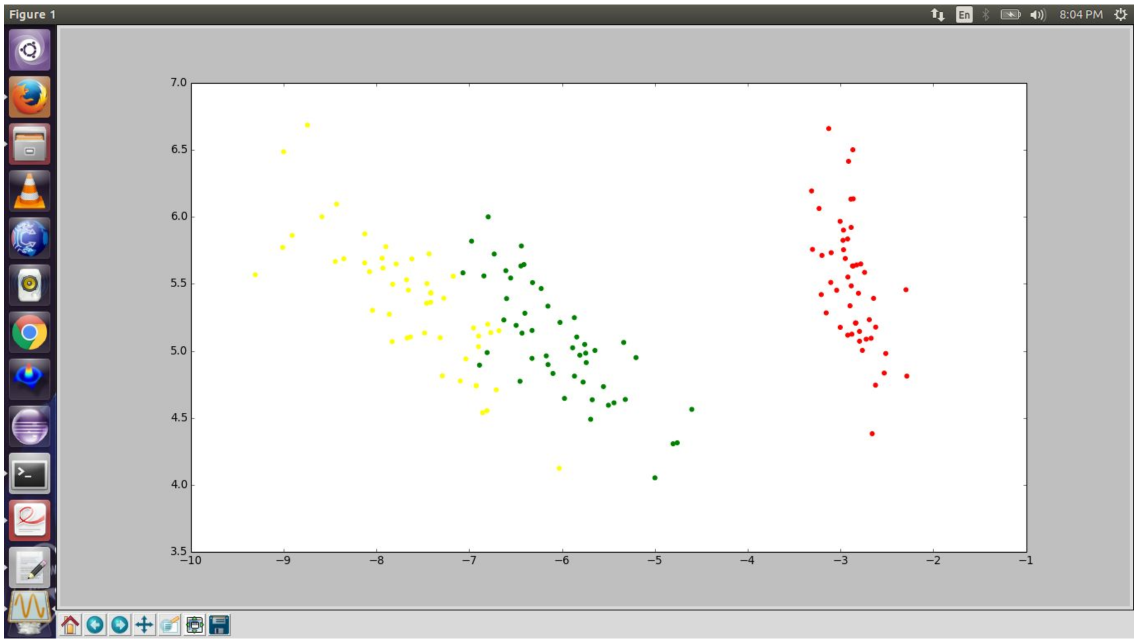
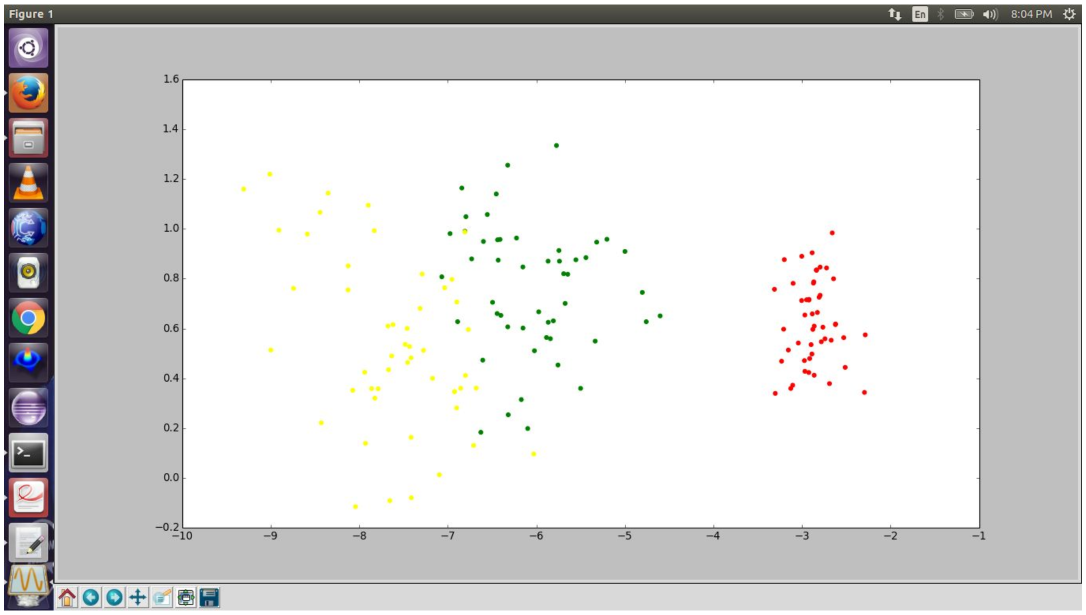


Figure 1:PC1 vs PC2

b) Pc1 vs Pc3

Figure 2:PC1 vs PC3



c)Pc2 vs Pc3

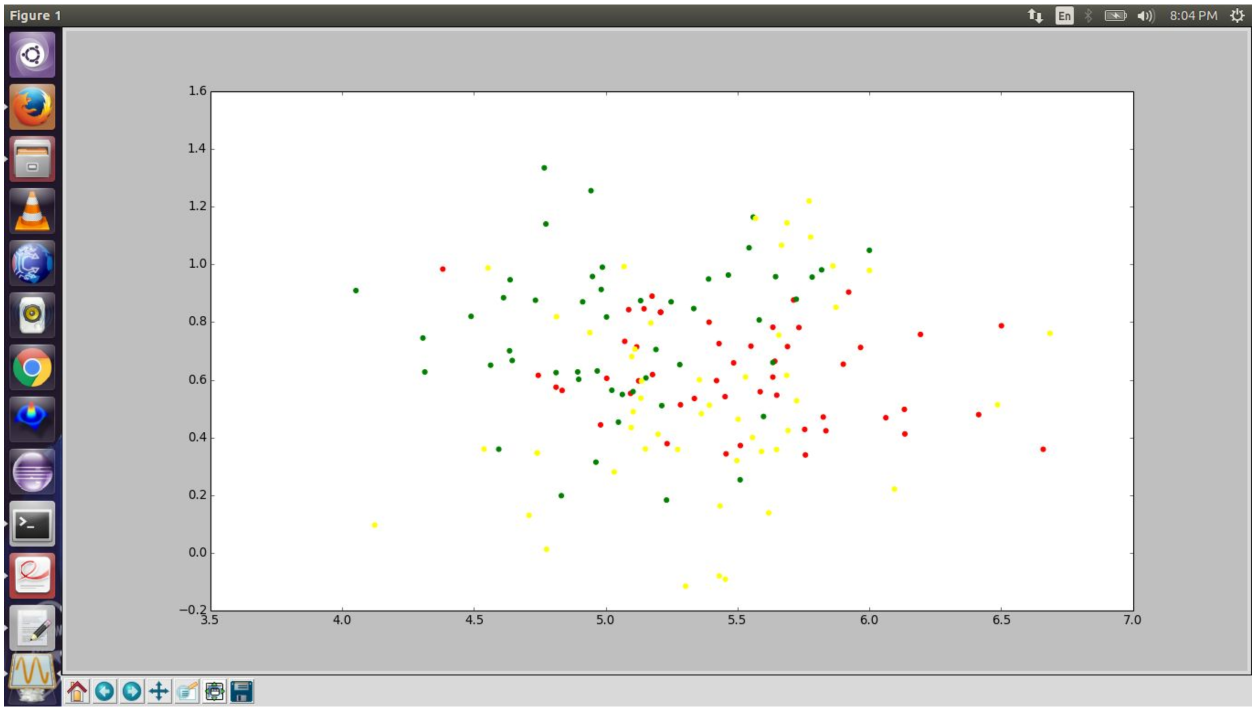


Figure 3:Pca2 vs Pca3

1.2 Algorithm: Fisher's Linear Discriminant Analysis (LDA)

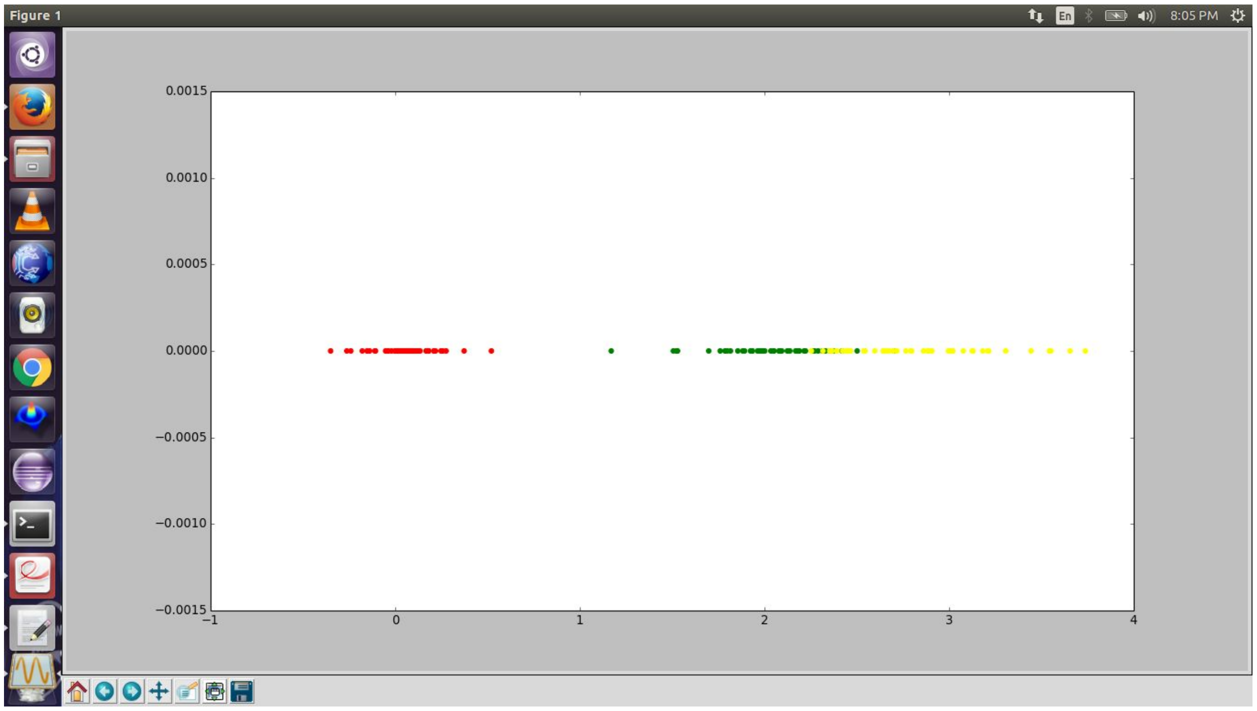


Figure 4:Lda on iris-data

2.DATASET: ARCENE-DATA

2.1 Algorithm:Principal Component Analysis(PCA)

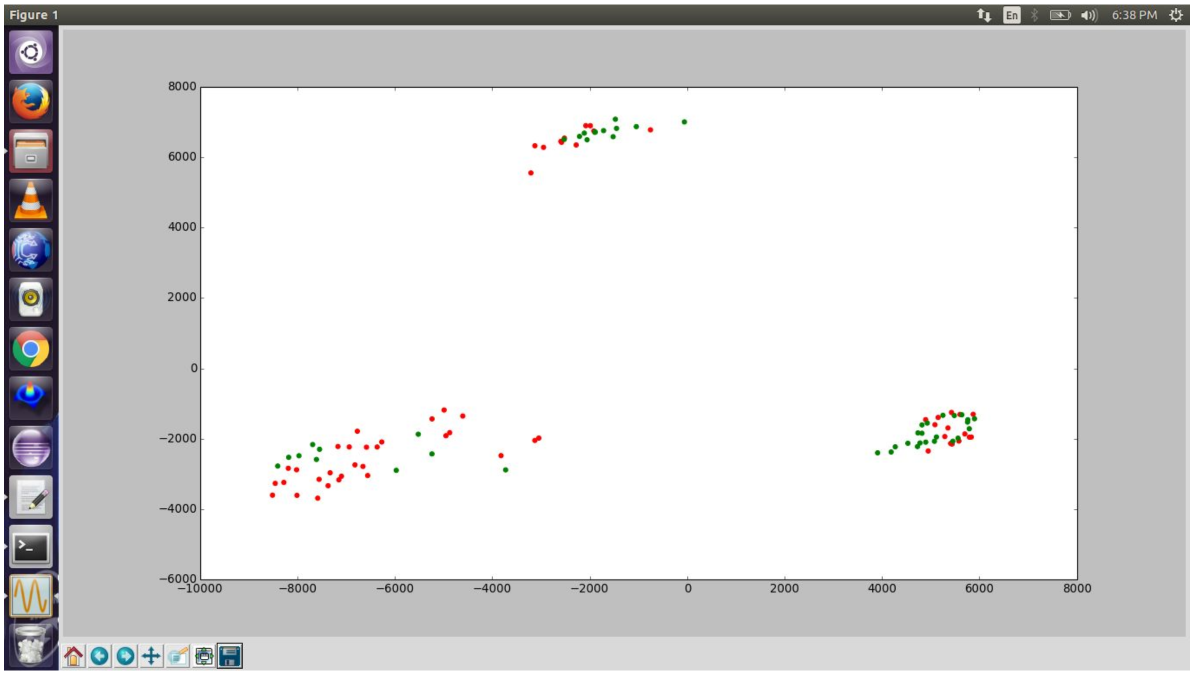


Figure 5:PCA(pc1 vs pc2) on Arcene-data

2.2 Algorithm:Fisher's Linear Discriminant Analysis (LDA)

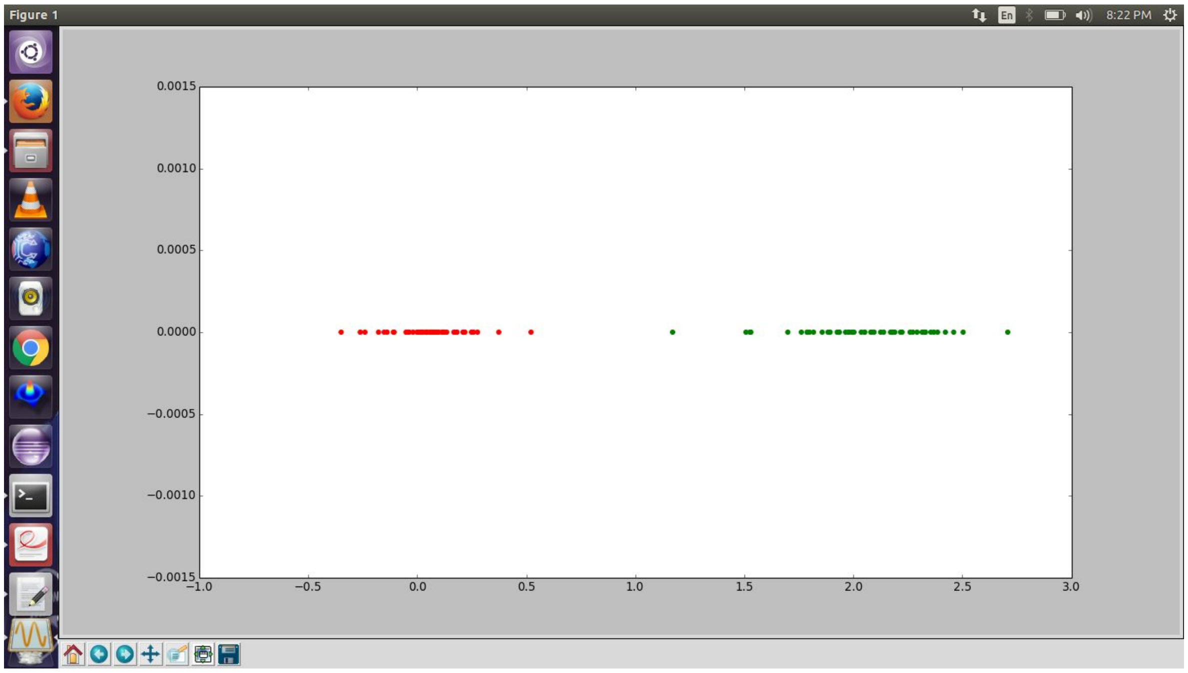
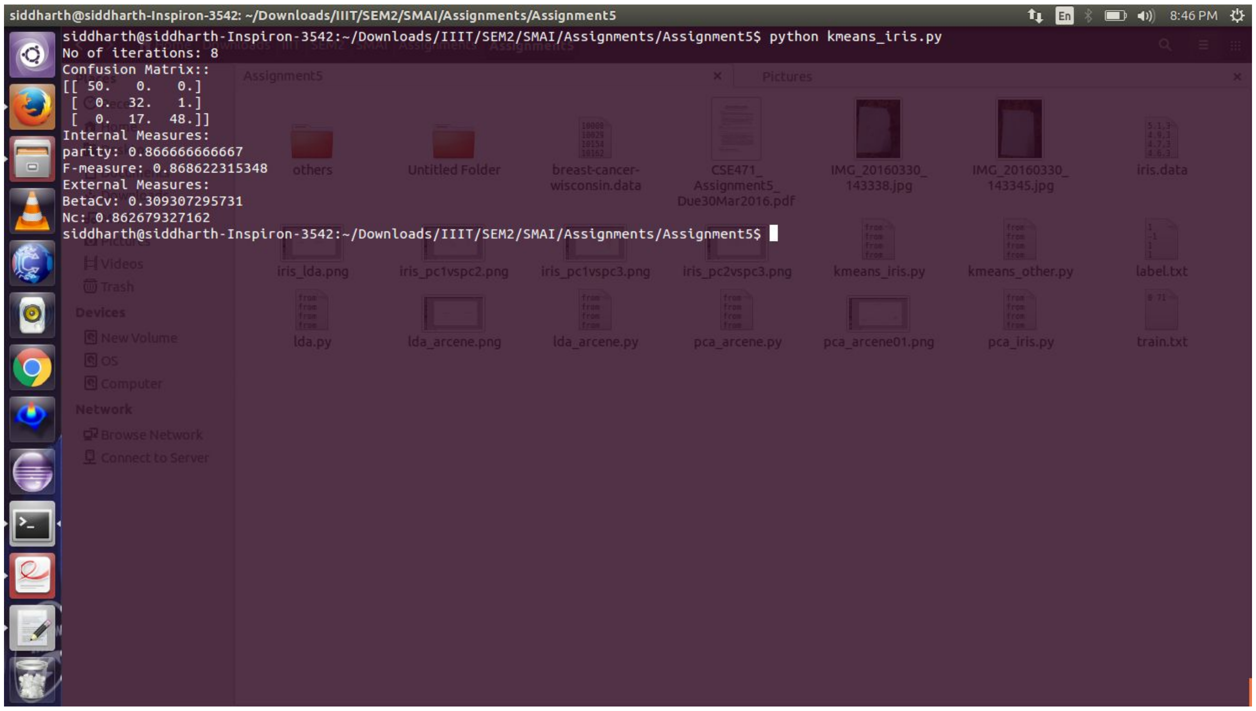


Figure 6:Lda on iris-data

Principal Component Analysis (PCA) applied to this data identifies the combination of attributes (principal components, or directions in the feature space) that account for the most variance in the data.
Linear Discriminant Analysis (LDA) tries to identify attributes that account for the most variance *between classes*. In particular, LDA, in contrast to PCA, is a supervised method, using known class labels.

3.DATASET: IRIS-DATA

3.1 Algorithm:K-means



```
siddharth@siddharth-Inspiron-3542:~/Downloads/IIIT/SEM2/SMAI/Assignments/Assignment5$ python kmeans_iris.py
No of iterations: 8
Confusion Matrix::
[[ 50.   0.   0.]
 [  0.  32.   1.]
 [  0.  17.  48.]]
Internal Measures:
parity: 0.866666666667
F-measure: 0.868622315348
External Measures:
BetaCv: 0.309307295731
Nc: 0.862679327162
siddharth@siddharth-Inspiron-3542:~/Downloads/IIIT/SEM2/SMAI/Assignments/Assignment5$
```

Figure 7:K-means on iris-data

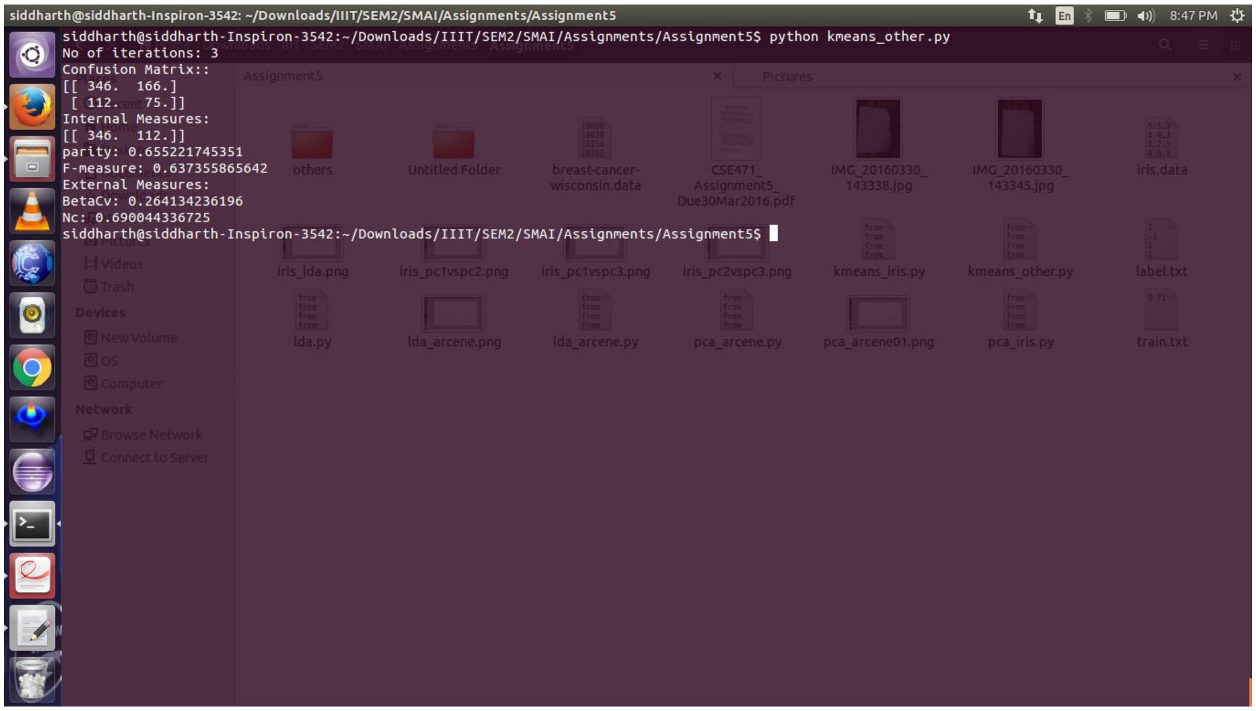
No of iterations: 8

Confusion Matrix:

	Class1	class2	Class3
Cluster 1	50	0	0
Cluster 2	0	32	1
Cluster 3	0	17	48

Validation:
Internal Measures:
Parity: 0.866666666667
F-measure: 0.868622315348
External Measures:
BetaCv: 0.309307295731
Nc: 0.862679327162

4.DATASET: breast-cancer-wisconsin.data
4.1 Algorithm:K-means



```
siddharth@siddharth-Inspiron-3542:~/Downloads/IIIT/SEM2/SMAI/Assignments/Assignment5$ python kmeans_other.py
No of iterations: 3
Confusion Matrix::
[[ 346.  166.]
 [ 112.   75.]]
Internal Measures:
[[ 346.  112.]]
parity: 0.655221745351
F-measure: 0.637355865642
External Measures:
BetaCv: 0.264134236196
Nc: 0.690044336725
siddharth@siddharth-Inspiron-3542:~/Downloads/IIIT/SEM2/SMAI/Assignments/Assignment5$
```

Figure 8:Kmeans on breast-cancer-wisconsin.data

No of iterations: 3
Confusion Matrix::

	Class1	class2
Cluster 1	346	166
Cluster 2	112	75

Internal Measures:
Parity: 0.655221745351
F-measure: 0.637355865642
External Measures:
BetaCv: 0.264134236196
Nc: 0.690044336725

Q1.Compare and contrast K-Means, K-median and K-medoid approaches
Ans.
From the perspective of algorithm steps, **the difference is :**

1.When computing the center of each cluster, **K-means** method will take the average(mean) of samples in each cluster.
2.While **K-medians** method choose to take the median of samples instead of mean.
3.And **K-medoids** method is similar to K-means but a slight difference is that center is chosen from the points itself.
For example, if {(0.9,3), (1,2), (2,1.9)} form a cluster, the center computed from K-means is $\frac{1}{n_c}\sum_{i=1}^{n_c} x_i=(1.3,2.3)$. While the center computed from K-medians is $median=(1,1.9)$.
While the center computed from K-medoid is (1,2).

From the perspective of optimization function,the difference is:

1.K-medians will minimize a *L1* norm error function. I.e. least absolute deviations (LAD), least absolute errors (LAE). It is basically minimizing the sum of the absolute differences between the target value and the estimated values.
2.While **K-means** will minimize a *L2* norm error function.L2-norm is also known as least squares. It is basically minimizing the sum of the square of the differences **(S)** between the target value (**Y_i**) and the estimated values
In practice, K-means is easily affected by outliers. K-medians is robust to outliers and results in compact clusters.Whereas K-medoid is more flexible and robust but it is more expensive.

Q2.Difference between using Covariance matrix versus Correlation matrix for doing PCA on data?
Ans.
The covariance matrix is used when the variable scales are similar and the correlation matrix when variables are on different scales.Using the correlation matrix *standardises* the data.
In general they give different results. Especially when the scales are different.
Use the correlation matrix R when within-variable range and scale widely differs, and use the covariance matrix C to preserve variance if the range and scale of variables is similar or in the same units of measure.

One can also say that use covariance matrix when variance of the original variable is important, and use correlation when it is not.

Q3.Kernelizing PCA.
Ans.
The "kernel trick": Given any algorithm that depends only on inner products between samples, the algorithm can be computed implicitly in any space F (Fis a Hilbert space) for which k can be defined, which means, for example, that linear algorithms can be mapped to one of a very rich set of possible nonlinear versions by simple choice of the function k.

PCA is a linear method, in the sense that the reduced dimension representation is generated by linear projections.Kernel PCA applies the kernel trick to create a nonlinear version of PCA in sample space by performing ordinary PCA in F.
It's striking that, since projections are being performed in a space whose dimension can be much larger than d, the number of useful such projections can actually exceed d (although the hope for those doing dimension reduction is that a number d'<<d of projections will suffice). It is not immediately obvious that PCA is eligible for the kernel trick, since in PCA the data appears in expectations over products of individual components of vectors, not over inner products between the vectors.
The eigenvectors of the covariance matrix in F lie in the span of the (centered) mapped data, and second, that therefore no information in the eigenvalue equation is lost if the equation is replaced by m equations, formed by taking the inner product of each side of the eigenvalue equation with each (centered) mapped data point.

Kernel PCA may be viewed as a way of putting more effort into the up-front computation of features, rather than on the classifier or regression algorithm. Kernel PCA followed by a linear SVM on a pattern recognition problem gives similar results to using a nonlinear SVM using the same kernel.