ASSIGNMENT 03

Username on miner: shirdisai
Name: Varshaa Shree Bhuvanendar

Iris:
Rank: 15(At the time of making report)
V-Score: 0.95

Image:
Rank: 57(At the time of making report)
V-Score: 0.77

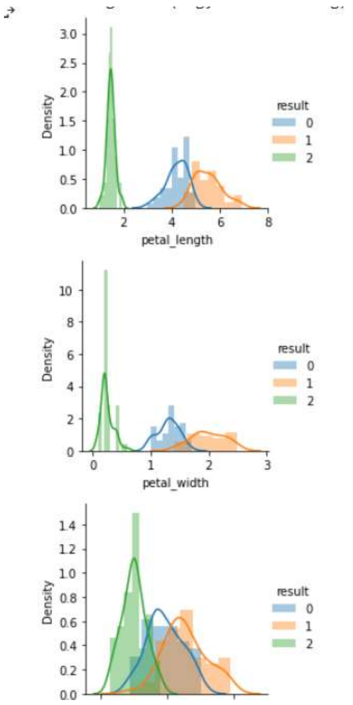
PART 1:

RESULTS FROM IRIS DATASET:

PREPROCESSING:

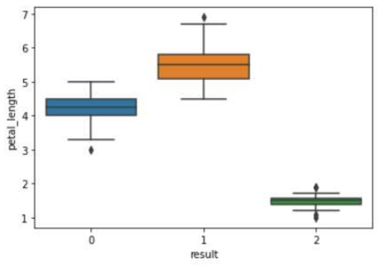
Imported data and split based on \n assigned column headings passed it to kmeans

GRAPHS:



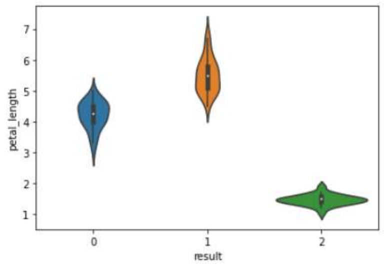
BOX PLOT:

```
sns.boxplot(x="result",y="petal_length",data=df)
plt.show()
```



VIOLIN PLOT:

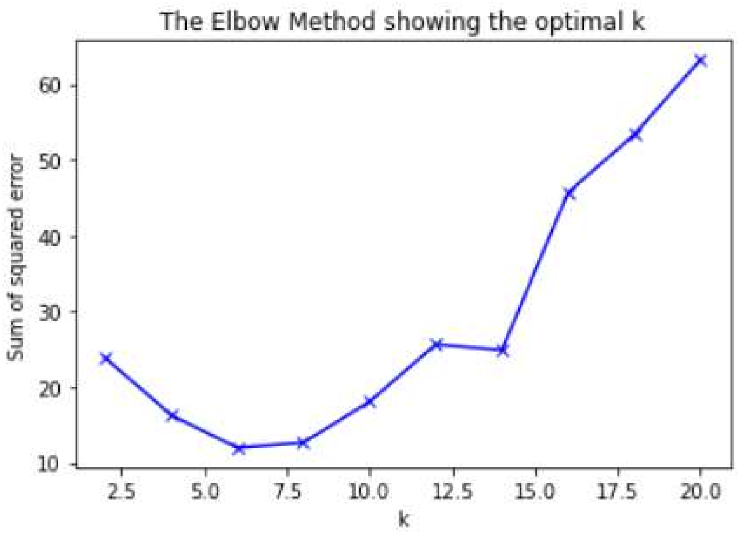
```
sns.violinplot(x="result",y="petal_length",data=df)
plt.show()
```



PART 2:

1) Implement choice of metric and plot metric on y axis and k on x axis

I used **sum of squared error** for validation to choose right k value from [2,20] with a gap of 2

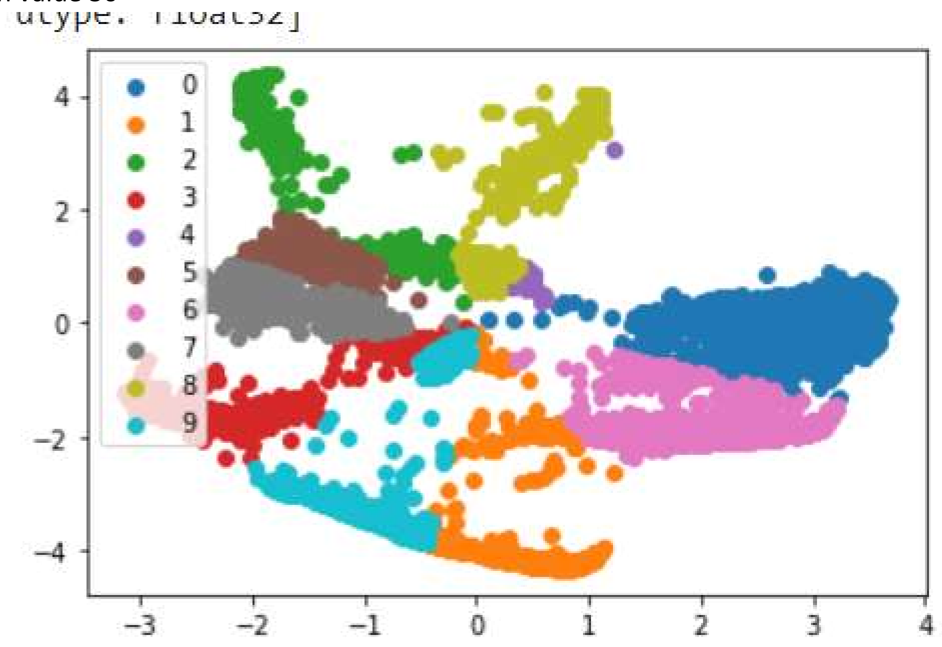


Here we can see the elbow.

2) You Approach

PSEUDO CODE:

```
•  Import dataset
•  Apply dimensionality reduction technique:
   1) Why we need dimensionality reduction?
      We have 783 columns it would be highly impossible to visualize such a data with so
      many dimension so we are reducing first to 72 using dimensionality reduction
      technique PCA then using dimensionality reduction technique TSNE to 2.
•  Next we implement KMeans
   1) Choose random centroids
   2) Using cosine for distance similarity
   3) In each iteration finding distance of centroid to each point and assigning each point
      to its least distance centroid
   4) We run same for 50 iterations.
      How I decided on iteration?
      Plotted the graph for different iteration values found it to remain same around
      iteration value 50
```



```
   5) Plotted clusters
   6) Plotted elbow curve for different k values
```

Dimensionality Reduction Techniques used:

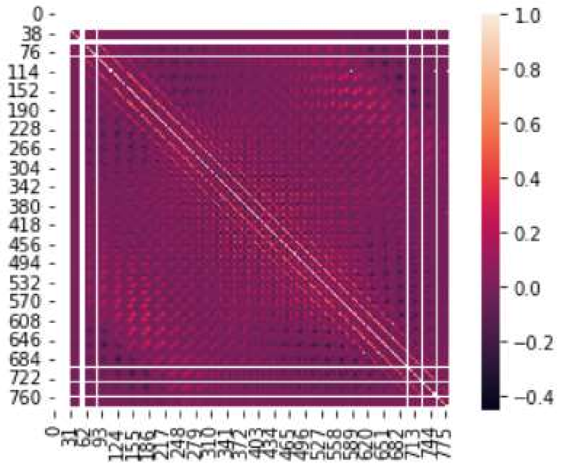
PCA(Principal component Analysis):

PCA is dimensionality reduction technique that is computation with many columns is very costly. To reduce computation time, we use the concept of dimensionality reduction. Where we take all columns in our case 783 and reduce them to 72.

Forms eigen vectors from covariance matrix . The new features are independent of each other.

```
[51]
import seaborn as sns
col = df[df.columns] #Subsetting the data
cor = col.corr() #Calculate the correlation of the above variables
sns.heatmap(cor, square = True) #Plot the correlation as heat mapdf
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa0fdf567d0>
```



TSNE:

Works on non-linear dimensionality.

STEPS:

Calculates probability in higher dimensional space and probability of similarity in lower dimensional space

Then it tries to reduce the difference in conditional probability in the higher and lower dimensional space. To minimize it uses gradient descent method

Conclusion:

Obtained an V-Score of 77 in part2 and V-score of 95 in part1 using kmeans clustering. What I learned is by doing dimensionality reduction improved V-Score