Firas ayoub 308185313

Shirin 311382840

For feature points we used:

sift.detectandcompute(im,None)

for clustering we used:

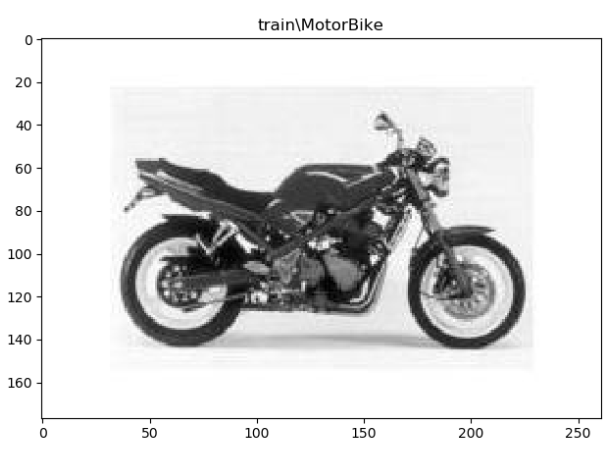
Kmeans(#clusters=50,tol=0.0001)

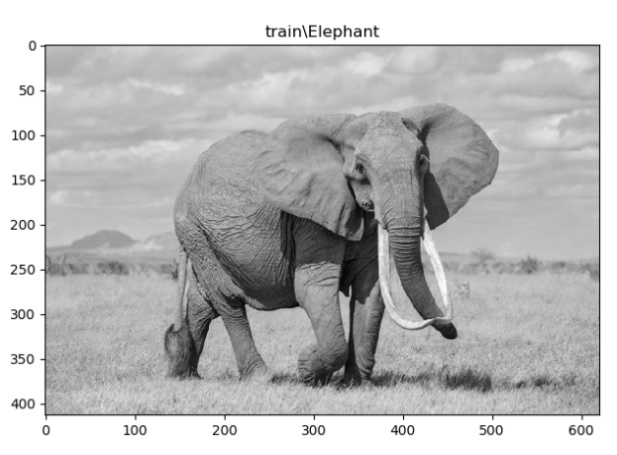
For Classifier we used:

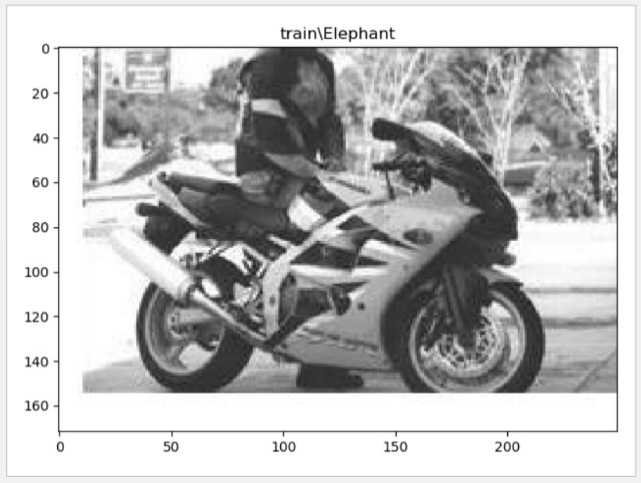
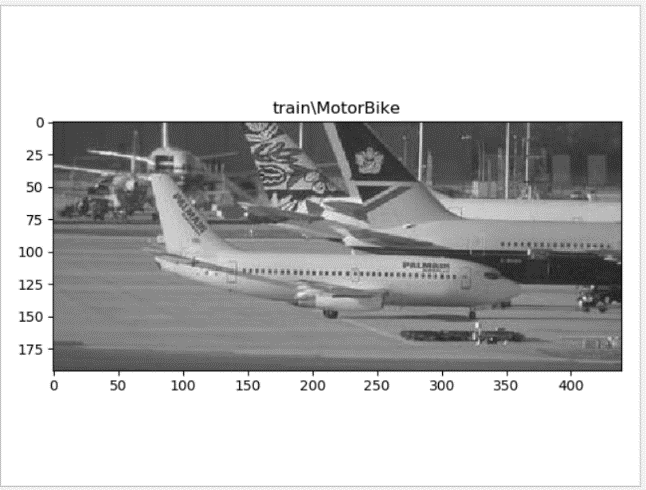
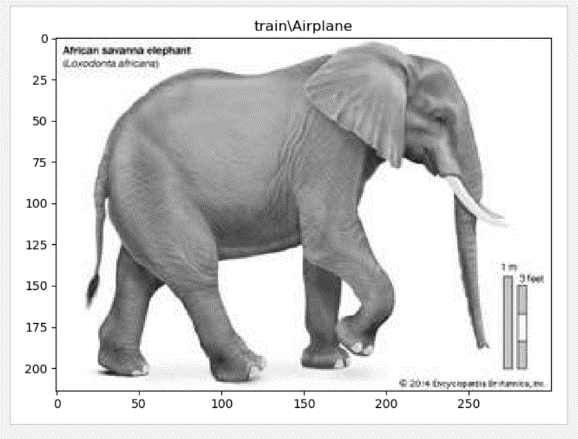
svm.LinearSVC()

(we changed the tol in the Kmeans instead of the svm)

Correct :



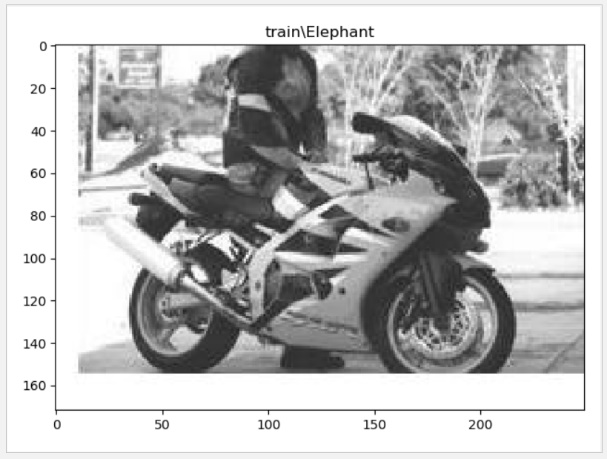
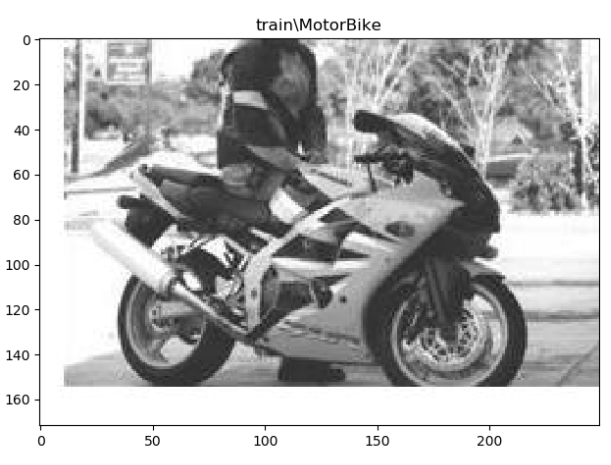


False classifications using different #clusters or different tol: 

To improve the system we tried two different ways expanding the vocabulary and changing the distance between features in the clustering but we quickly realized that we cant over do it meaning if you keep on increasing your vocabulary or minimizing the distance at some point you lose accuracy that’s why we chose #clust=50 and tol= 10^-3

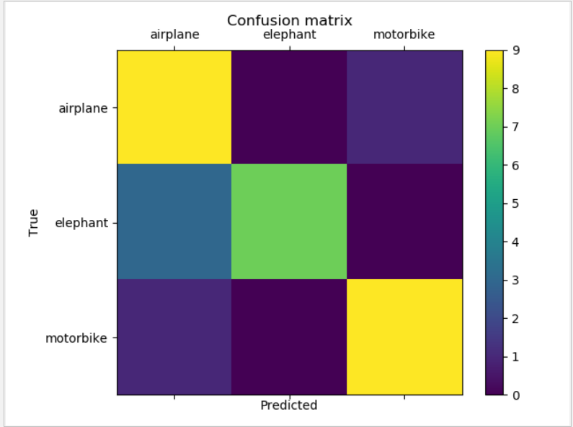
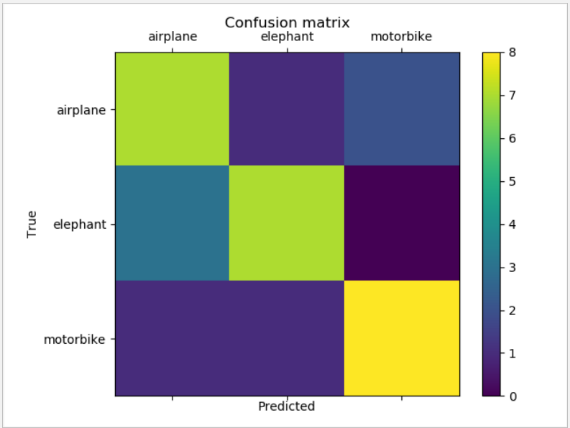
Some examples for different confusion matrices and improved predections.

Example for two different tol with two different results:

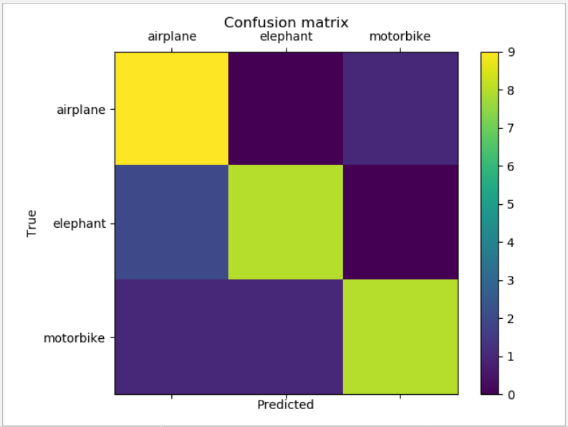
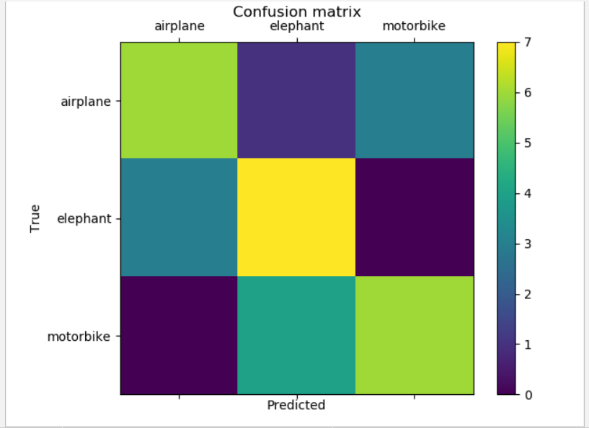


Confusion matrices for different # of clusters:

K=100 k=50

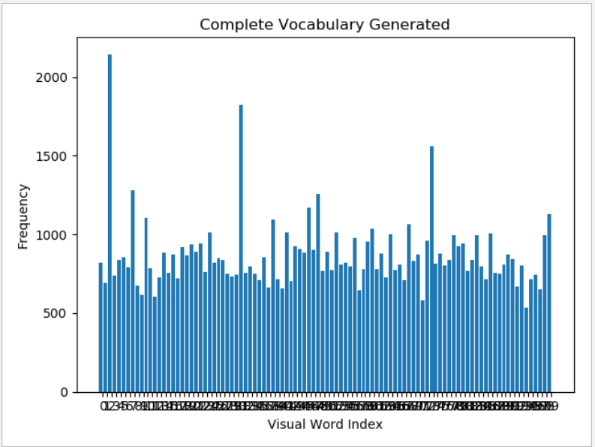
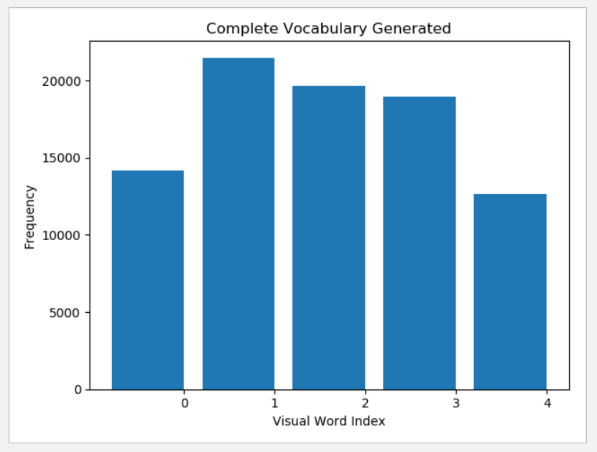
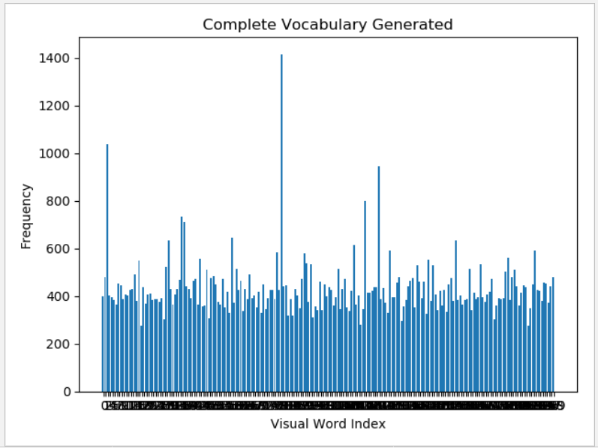
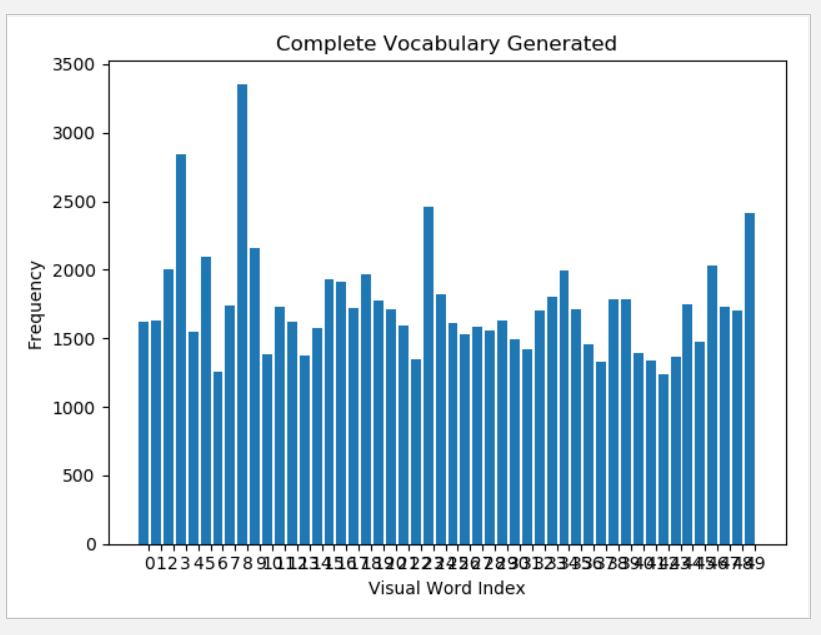


K=5 K=200



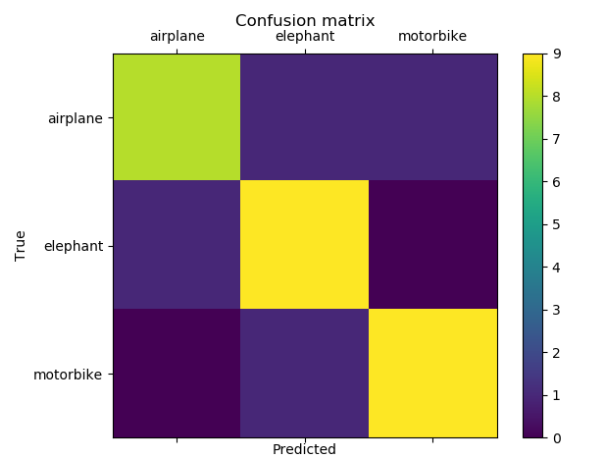
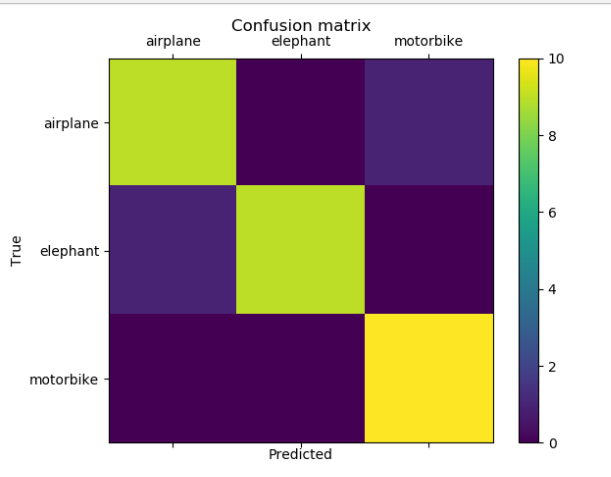
Frequencies histogram for each k :

K=50

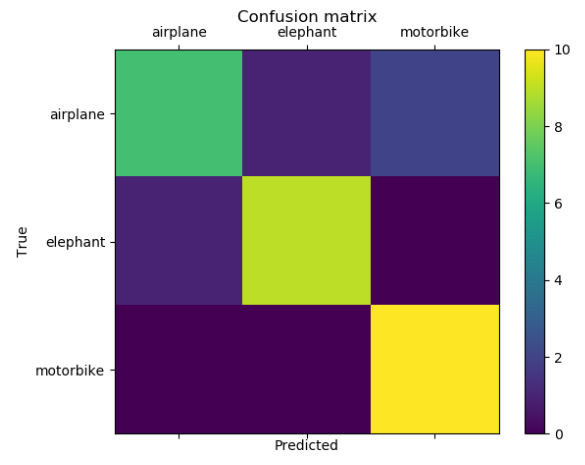
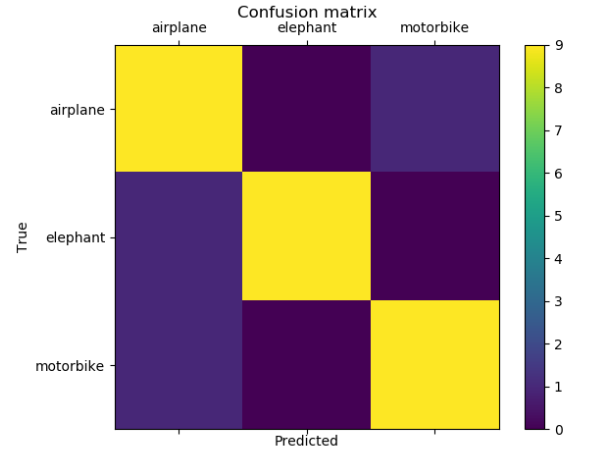


Confusion matrices for different tol while # cluster =50:

Tol= standard tol= 0.1



Tol= 10^-5 tol=10^-6



f) adding more classes to the train model wouldn’t consistently improve nor decrease the accuracy since its very dependent on both the vocabulary that we already have and the classes were adding it either we add a word that is way far from all the other classes and then probably nothing will change or we add a class that is very similar to one that we already have and then we will reduce the accuracy or that we get a class that will help in dividing to classes that were already being confused with each other.