1. (a)
   - Setting the number of points apriori: K-Means is sensitive to this factor because it assumes there are K clusters in the data set where K is determined apriori. K-Means does not attempt to find the inherent organization of the data but simply finding the most probable clusters according to K.
   - Anisotropy: K-Means is sensitive to anisotropy in data because it assumes the data is spherical/convex. In the K-Means model, it tries to fit data around K centers which minimizes the within cluster sum of squared criterion. If the data in an intuitive cluster is distributed anisotropicly, K-Means is likely to group a part of the data with data from other intuitive cluster to minimize within cluster sum of square instead of preserving the intuitive non-sperical shape.
   - Unequal variance: K-Means assumes the data is separated in K groups of equal variance. In each iteration, K-Means reduces the difference in variance between K clusters and converges until the variance between groups are the same. Thus if two intuitive clusters have different variance, K-Means will compensate by assigning points from other intuitive clusters to each until the variance converges, producing an unsatisfactory cluster assignment.
   - Unequal cluster sizes: K-Means is affected because it tries to minimized the within-cluster sum of squares, which natually put more weight to clusters with more data. Consider a dataset which contains two intuitive clusters K, J, where the number of data points in each cluster $k << j$ and the average squared distance to cluster center is the same d_{avg}. Then the within-cluster sum of squares $k * d_{avg} << j * d_{avg}$. K-Means will assign points from intuitive cluster J to K in order to balance the within-cluster sum of squares between them, thus producing an undesirable cluster assignment. In the example given by the code, the within-cluster sum of squares are incidentally equal between three intuitive clusters although they have different number of data points, thus K-Means is able to produce a desirable result under such specific condition.

   (b) As we can see in Figure 1, some points that belongs to the green cluster in the original cluster assignment is marked as red after feature scaling and vice versa. Feature scaling affects the K-Means because it affects the variance and the isotropy of each cluster. In the lower graph of Figure 1, the clusters are less spherical compare to the upper graph, thus K-Means moves points between the red and green clusters to balance the variance between them. Please refer to *simple_feature_scaling.py* for code.

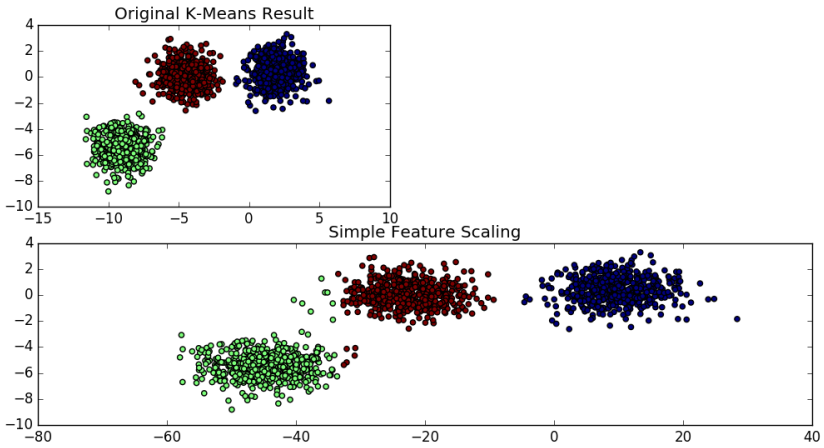

Figure 1: Before and after increasing the x-feature by a factor of 5.

   (c)

   (d) We can preprocess the data by normalization. Normalize the mean and scale each feature to unit variance.

(e) As shown in the upper right graph in Figure 2, GMM mitigates the effect of anistropicy by allowing custom covariance matrices for each Gaussian distribution. As shown in the bottom-left graph in Figure 2, GMM mitigates the effect of unequal variance because it allows each cluster(Gaussian distribution) to have different variances. Although cannot be shown through this dataset, GMM also mitigates the effect of unevenly sized blobs because it allows different covariance matrices for each cluster. Since GMM requires a input of K mixtures apriori, it does not mitigate the effect of setting the number of clusters apriori in K-Means.

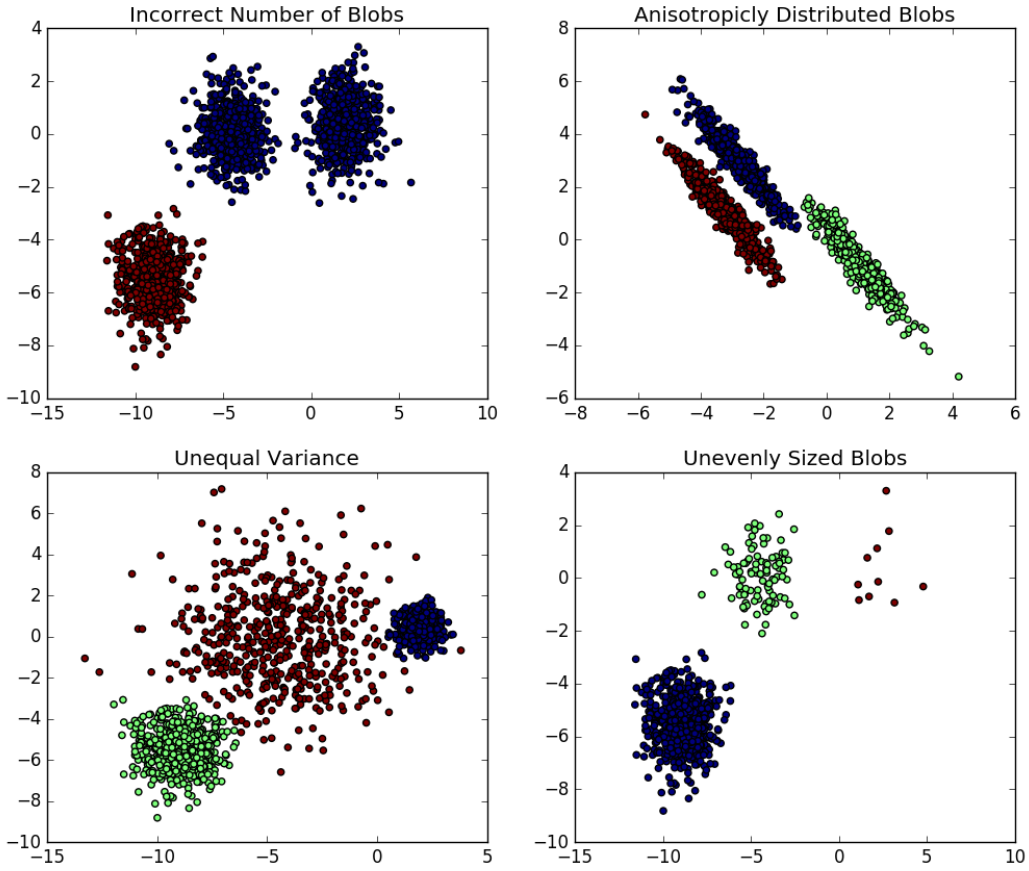

Figure 2: Clustering using GMM.

2. (a)
   (b)
   (c)
   (d)
   (e)
   (f)
3. (a)
   (b)