1. Less dimension lead to less computation/training time. Some algorithms also do not perform well when we have a large dimensions. It also takes care of multidimensionality by removing redundant features.

Disadvantages of Dimensionality Reduction:

* It may lead to some amount of data loss.
* PCA tends to find linear correlations between variables, which is sometimes undesirable.
* PCA fails in cases where mean and covariance are not enough to define datasets.

2. The curse of dimensionality basically means that the error increases with the increase in the number of features. It refers to the fact that algorithms are harder to design in high dimensions and often have a running time exponential in the dimensions.

3. No, dimensionality reduction is not reversible in general.

4. Well it depends on dataset. If it is comprised of points that are perfectly aligned, PCA can reduce the dataset down to 1 dimension and preserve 95% of the variance.

5.  If I perform PCA on a 1,000-dimensional dataset, setting the explained variance ratio to 95%. In this case roughly 950 dimensions are required to preserve 95% of the variance. So the answer is, it depends on the dataset, and it could be any number between 1 and 950.

6. The following are the scenarios where the following are used:

* Vanilla PCA: the dataset fit in memory
* Incremental PCA: largest dataset that don't fit in memory, online tasks
* Randomized PCA: considerably reduce dimensionality and the dataset fit the memory.
* kernel PCA: used for nonlinear PCA

7. By doing PCA, it is a good choice for dimensionality reduction and visualization for datasets with a large number of variables.

8. Indeed, it often make any sense to chain two different dimensionality reduction algorithms. A common example is using PCA to quickly get rid of a large number of useless dimensions, then applying another much slower dimensionality reduction algorithm, such as LLE.