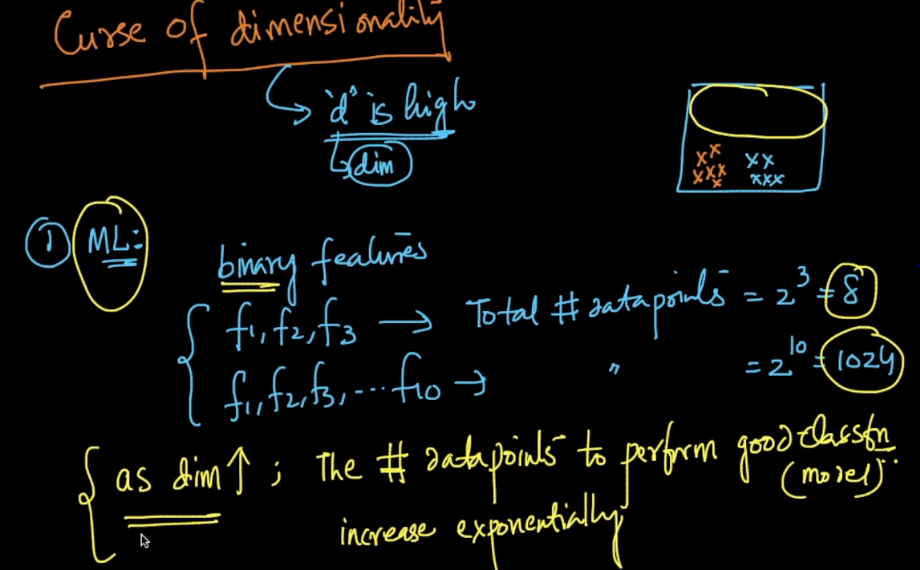
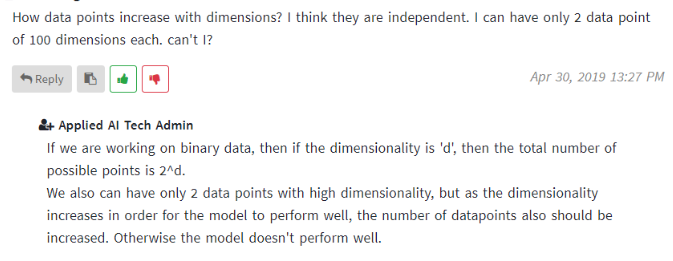
**Curse of dimensionality:**

Let’s say we’ve binary features, so if there are 3 features then there may 8 datapoints, and if there are 10 features then there may be 1024 datapoints, that means datapoints are increasing exponentially.

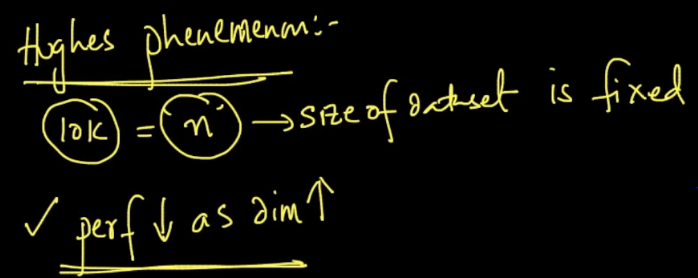
If features are not binary then in such case also datapoints increases exponentially.





**1st Problem:**

If we have fixed datapoints, and if no of dimension increases then performance decreases. This phenomenon is called Hughes phenomenon.



**2nd Problem: Distance functions (Euclidean dist).**

Let’s say in 1D we have n-random pts. Now we calculate dist-min(distance of a point which is most nearest to xi), and dist-max(distance of a point which is most farthest to xi).

Now the equation given in 2nd image will be significantly greater for 1,2 or 3 dimensions. But as dimensions increases that equation will tend to become 0.

This ratio or equation intuitively captures how far is the farthest random point as compared to the nearest random point in a d-dim space. As this ratio tends to zero, it implies that the farthest and nearest points are very close to each other.

Now when that can becomes, when dist-min becomes equal to dist-max, that means all the points are equidistant.

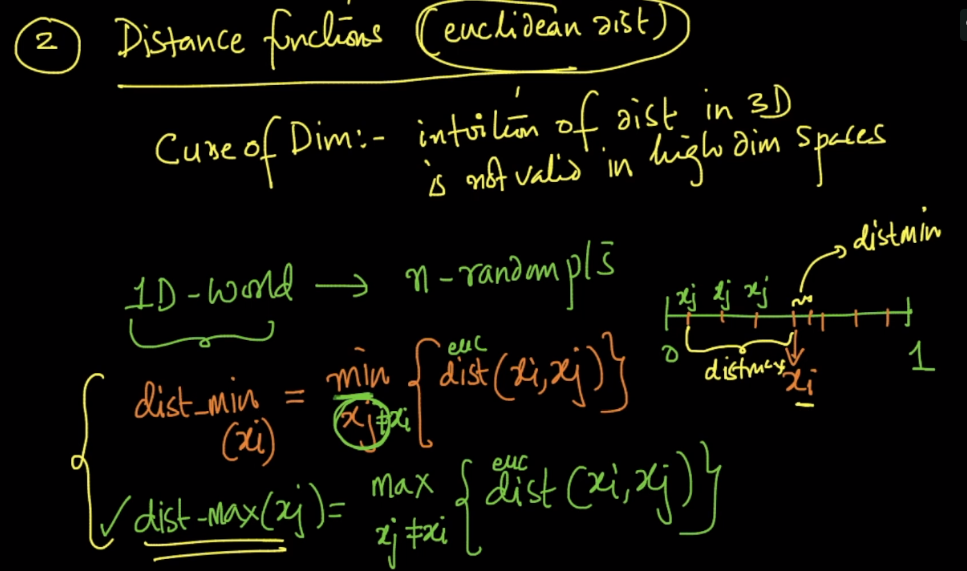
So if all the points become equidistant then how would we able to get results or patterns from them.

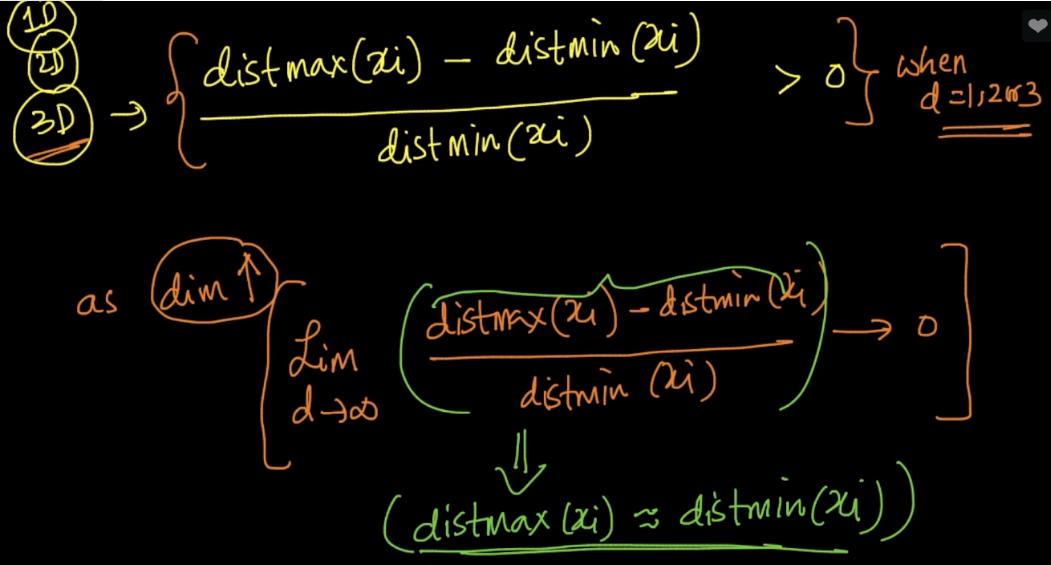
But remember the assumptions is that all points are randomly taken. But it’s observed even if data is not random, model does not work well for high dimensions in some cases.

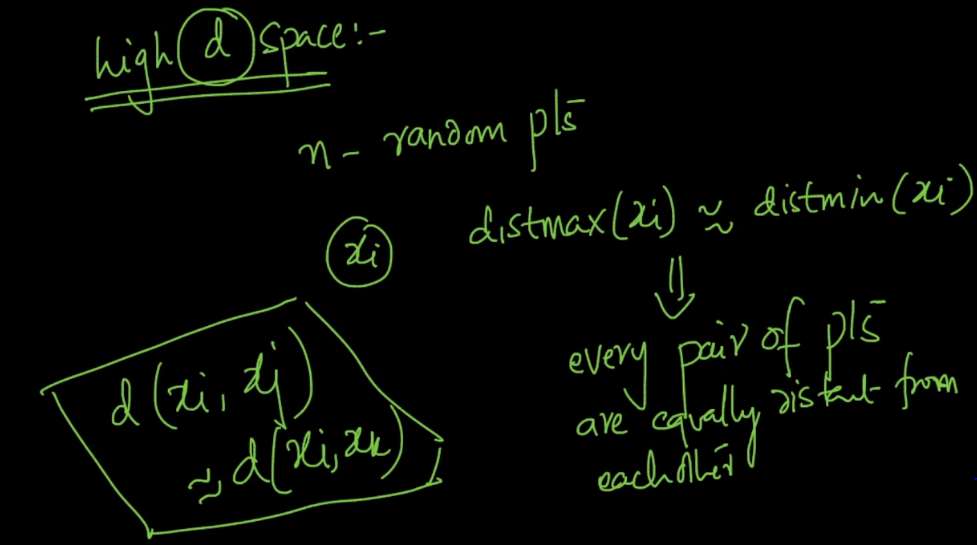
So if we use **K-NN** using **Euclidean distance,** then in high dimensions, since all points become equidistant, then it doen’t make any logical sense. Therefore KNN with Euclidean distance doen’t work well in high distance.

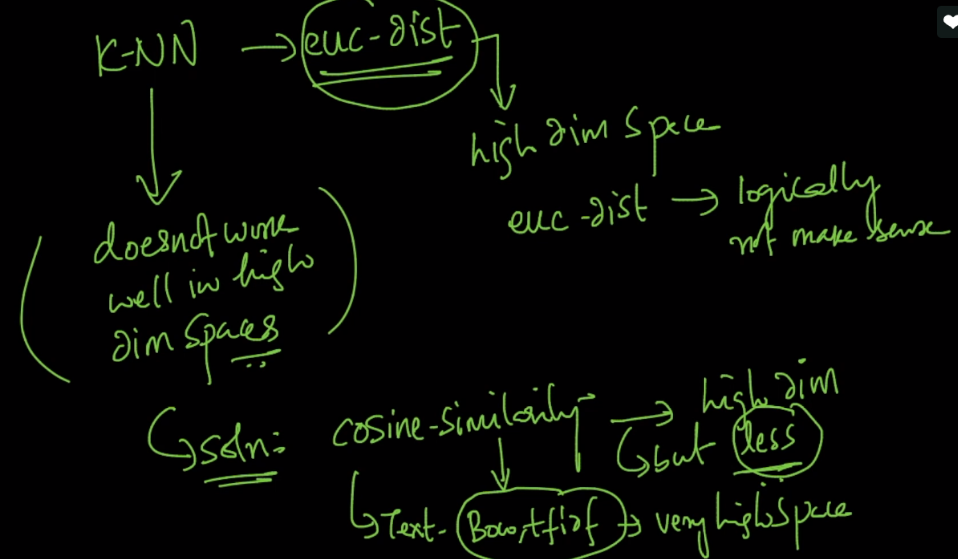
Hence for text processing, since there are very large no. of dimensions, we use knn with cosine distance.

**Note:** If there is high dimensions and all the features are discorrelated(means there is very low correlation between them), then in such case ML models works well, even if we take all dimension.





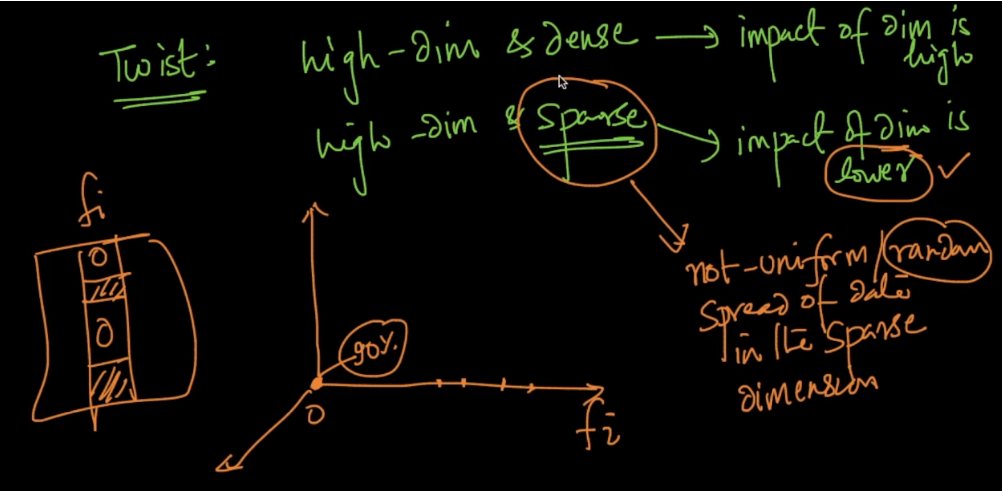




**Twist for high dimension:**

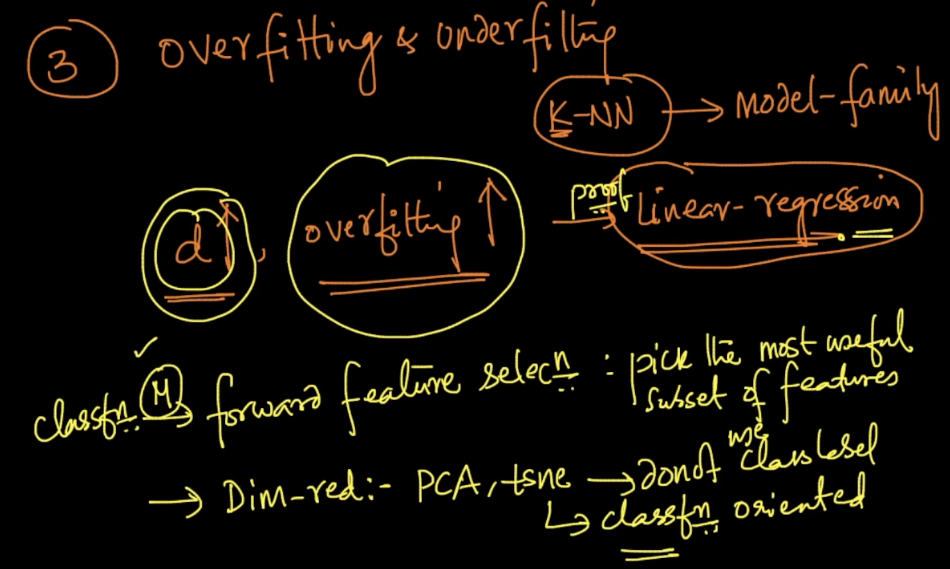
If there are high dimensions, and data matrix is also dense, then impact of dimensionality curse will be high.

If there are high dimensions, and data matrix is also sparse, then impact of dimensionality curse will be low. Because sparse is not random spread of data



**3rd Problem:**

As dimensions increases, overfitting also increases.



So what we can do is we can use forward feature selection selection or can reduce dimensionality using PCA, tsne.

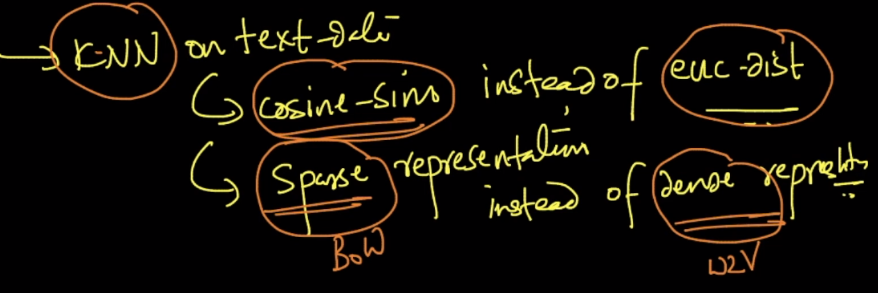
But PCA and t-SNE doen’t use class labels or it doesn’t consider class label while reducing dimensionilty. That means it is not classification oriented.

So in classification, it’s better to use Forward feature selection.

**K-NN on test data:**

So whenever we are applying K-nn on test data we should use.

* Cosine –simi instead of ecu-dist.
* Sparse representation(BOW) instead of dense(w2v) representation, but must check for each whichever performs good choose that.



<http://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/>