**ML – 3DMM (3D morphable models)**

3DMM are powerful statistical models of 3D facial shape and texture, we will talk about the PCA method of 3DMM

3DMM are based on two concepts:

* All faces are in **dense point-to-point** correspondence, this is to define linear combination of faces in a meaningful way
* Separation of facial shape and color to **work independently from illumination and camera parameters**

This evolved from the **eigenfaces** approach, Eigenfaces treated images of faces as a vector space and performed a principal component analysis, with the eigenvectors representing the main modes of variation in that space

This had limitation in terms of pose and illumination

3DMM parametrize surfaces by using interpolation and PCA is applied to estimate the prior distribution in the space of coloured 3D scans; then at test time you take a plausible sample that can be rendered similarly to the input image

**Remark:** All of machine learning is about extrapolating patterns

Basics definitions for ML:

* **Dimension** usually refers to the **number** of attributes, although it can also be used in form of "second dimension of the data vector is person age", but it is rather rare - in most cases dimension is "number of attributes"

Groups of Dimensions create datapoints

* **Attribute** is one particular "type of data" in your points, so each observation/datapoint (like personal record) contains many different attributes (like person weight, height, age, etc.)
* **Feature** may have multiple meanings depending on context:
  + It sometimes refers to attribute
  + It sometimes refers to the internal representation of the data generated by particular learning model, for example - neural networks extract **features** which are combinations of the attributes or other features
  + It sometimes refers to the hypothetical representation of the data induced by the kernel method (in Kernel PCA, Kernel k-means, SVM)

**The curse of dimensionality**

This term describes the explosive nature of increasing data dimension, resulting in an exponential increase in computational cost and in needed datapoints (and in training set size)

As the number of features increase, the chance of **overfitting** becomes more and more a reality, as the model will become more and more dependent on the data it was trained on

Dimensionality reduction can be done manually or programmatically; they are both based on the **variance** of the data (see PCA in previous document – rotate and project data along the direction of increasing variance)