Gaussian Mixtures

Abdullah Zafar
DEEP LEARNING 1
Hochschule Hamm-Lippstadt
Lippstadt, Germany
abdullah.zafar@stud.hshl.de

***Abstract*—This paper initials discusses the mathematical Basics of Gaussian Mixtures and how Gaussian Mixtures are proven mathematically. The equation used to describe Gaussian Mixtures and a brief summary describing all the variables that are mentioned in the equation. Furthermore, Expectation-Maximization in Gaussian mixture models are mentioned and how are they useful and their implementations in the given scenario. Drawbacks of K means Clustering furthermore emphasis on the important Gaussian Mixture Models and how they fulfill these gaps created by K means Clustering. Bayesian Gaussian Mixture Models and Anomaly detection using gaussian mixture models are mentioned in good detail aswell, as these are some important topics to be covered along Gaussian Mixtures.**

Mathematical Equations and usage is also discussed along the paper. Graphical images are used repeatedly to better explain the topic as Gaussian Mixtures rely heavily on graphical representation.
Implementing an example with python framework and applications are also discussed .Lastly, The paper includes Gaussian Mixture Model implementations and how they are implemented in real world. Summary and Conclusion is a brief look at the topic that are discussed in the paper.

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Machine Learning includes two methods of learning, supervised and unsupervised. The main distinguishing factor between the both includes the method it uses to deal with data. Clustering is a unsupervised method where clusters area made around the data set that share the same characteristics.K-means is a popular method of clustering but a major drawbacks associated towards it is that it does not specify that how much a data point is associated with a specific cluster. This is commonly known as hard Clustering. Gaussian Mixture Models empowers us to implement soft clustering and get a better representation.A Gaussian mixture model is a probabilistic model that assumes all the given data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters and tends to group the given points to a single distribution that it belongs to. The Gaussian mixtures implement the expectation-maximization algorithm for fitting mixture-of-Gaussian models. Gaussian Mixtures further compute Bayesian Information Criterion to asses the number of clusters in a given data. Gaussian Mixture fit method enables the learning of a Gaussian Mixture Model from a given train data. Each sample is then assigned to a Gaussian using Gaussian Mixture predict method.

II. Gaussian Mixtures mathematical basics

Inability of Gaussian Distribution to model real data sets overshadow its important analytical properties. Super position of several Gaussians always gives a better representation of the data available. Gaussian mixture models enables us to form superpositions and can be formulated as probabilistic models known as mixture distributions.

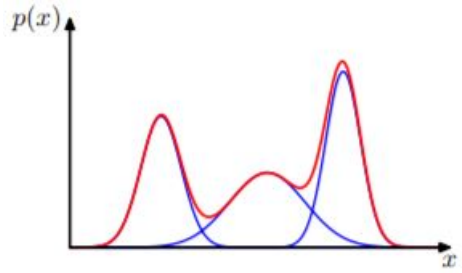


Fig. 1. Demonstrates 3 Gaussian represented in blue and their sum in red

Figure 1.0 represents how Gaussian Mixtures enables us to sum an adequate number of Gaussians for better data representation.

By utilizing an adequate number of Gaussians, and by changing their methods and covariances just as the coefficients in the linear combination, practically any continuous density can be approximated to arbitrary precision. We in this manner consider a superposition of K Gaussian densities of the form, which are called Gaussian Mixtures.

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Each Gaussian density $N(x\|\mu_k, \sum_k)$ is called component of the mixture and has its own mean μ_k and covariance $\sum_k$.

K represents the number of clusters in a given data set and is varied according to the number of clusters. Each Gaussian k in the mixture includes the following parameters:

- μ that defines the center.
- covariance that is represented using $\sum$.
- A mixing probability π that defines how big or small the Gaussian function will be.

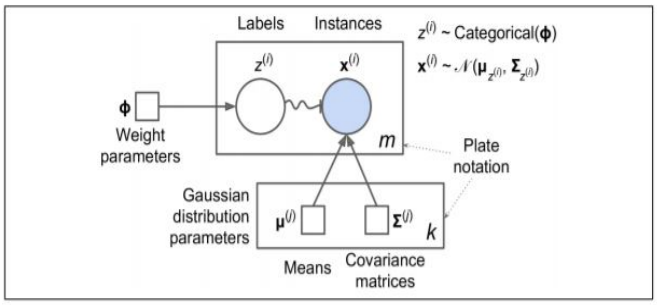


Fig. 2.

Figure 2.0 represents the structure of the conditional dependencies between random variables.

Random variables are represented by circles. The squares address fixed qualities (i.e., boundaries of the model). The big square shapes are called plates: they show that their substance is repeated a few times.

III. Anomaly detection using Gaussian mixture models

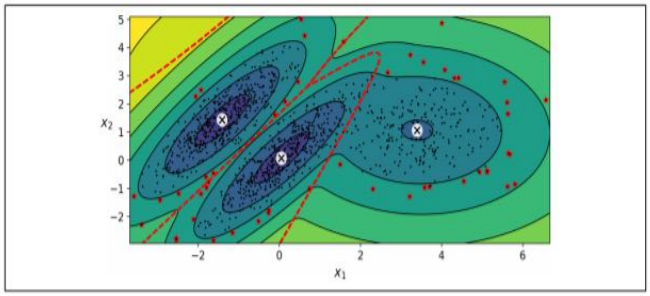


Fig. 3. Anomalies are represented as stars

Anomaly Detections is the process of locating the instances that oppose the convention in a very strong manner. These instances are commonly referred as anomalies or outliers, and the remaining ones are referred as inliers. Anomaly detection gets very handy when they are applied to real life scenarios. Few of the examples include, detecting fraud, detecting defective products in manufacturing or to obsolete dataset another feeding another model, which can improve the performance of the resulting models by many steps. Detecting anomalies in Gaussian mixture model is not a hefty task because any instance being located in a low density region can be referred as an anomaly.

Novelty detection differs from anomaly detection in such a way that in novelty detection, the algorithm is assumed to be trained on a ‘’clean” dataset, free of outliers, whereas anomaly detection does not make that assumption.

Gaussian mixture models tries to accommodate all the available data, including outliers, if outliers are more in number, the ‘normality’ gets effected and some outliers may be now considered as normal. If this happens, you can try to fit the model once, use it to detect and remove the most extreme outliers, then fit the model again on the cleaned up dataset. Another approach is to use robust covariance estimation methods (see the EllipticEnvelope class).

IV. Bayesian Gaussian Mixture Models

Selecting number of clusters can be a long task but it is possible to use Bayesian Gaussian mixture class which is in fact capable of giving weights equal or close to zero to the clusters those are unnecessary.

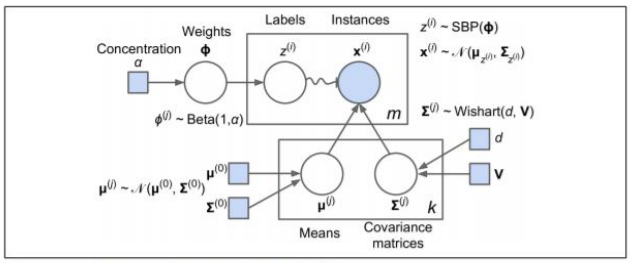


Figure 9-22. Bayesian Gaussian mixture model

Fig. 4. Anomalies are represented as stars

In figure 4.0 , the cluster parameters (including the weights, means and covariance matrices) are not treated as fixed model parameters anymore, but as latent random variables.

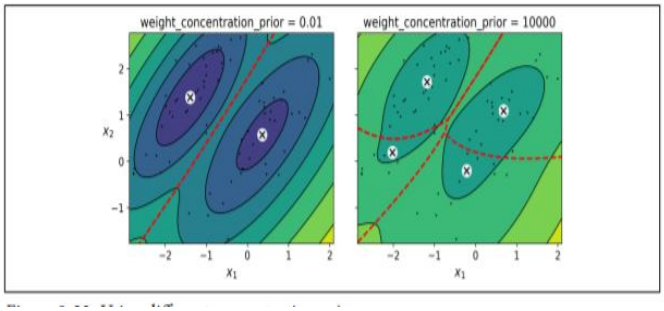


Fig. 5. Anomalies are represented as stars

V. Expectation-Maximization

Expectation maximization is a method that is used to estimate the parameters of the mixture model. Models are typically learned by maximum likelihood estimation, which tend to seek the maximum probability of the given data model parameters. It is usually analytically impossible to find maximum likelihood solution for mixture models by differential of log likelihood and solving for 00.

Expectation maximization is an iterative algorithmic method that tends to increase with each subsequent iteration. Generally two steps are involved in calculating Expectation maximization for Gaussian Mixture Models.

The first step involves the calculation of the expectation of

the component assignments for each data point given the model parameter such as ϕ μ and σ.

The second step is generally referred as maximization and it considers calculating the maximum expectation that was previously calculated in the first step with relation to the model parameters and this part consists of updating the values of ϕ μ and σ. As mentioned before, this method is an iterative method, so iterations take place until and unless the algorithm converges, hence giving the maximum likelihood estimate. Altering between the values that are known to be fixed or assumed, maximum likelihood successfully estimates the values those are non-fixed and they can hence be calculated in an efficient manner.

VI. Bayes' theorem

Bayes's theorem can be used along estimated model paraments to estimate the posteriori component assignment probability. Learning the fact that an instance belongs to a specific distribution provides an important tool for learning clusters. Clustering has a lot of implementations in machine learning and vary from tissue differentiation in medical imaging to customer segmentation in market research.

$$p(C_i \mid x) = \frac{p(x, C_i)}{p(x)} = \frac{p(C_i)p(x \mid C_i)}{\sum_{j=1}^{K} p(C_j)p(x \mid C_j)} = \frac{\phi_i \mathcal{N}(x \mid \mu_i, \sigma_i)}{\sum_{j=1}^{K} \phi_j \mathcal{N}(x \mid \mu_j, \sigma_j)},$$

VII. Applications of Gaussian Mixtures

Gaussian Mixture Models are extensively used in the following aspects:

- Speech recognition.
- Object tracking with multiple objects.

VIII. How Gaussian Mixture Models differs from K-means

K-Means Clustering is a method that divides sample data into K clusters depending on features. Clustering is accomplished by shortening the distance between the sample and the cluster's center.

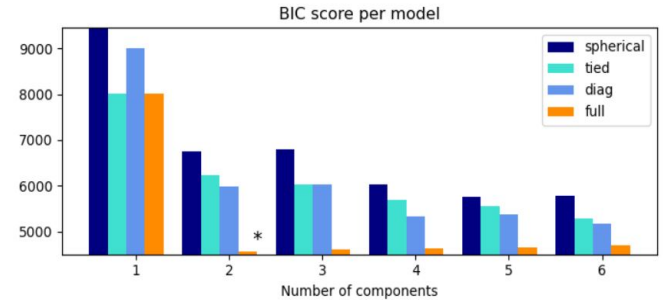
Hard clustering is a more popular term for this. Lesser processing time, better for high-dimensional data, and it's a lot easier to install are just a few of the benefits connected with K means. Hard clustering has a number of drawbacks, including the possibility of erroneous data and the fact that, unlike Gaussian mixtures, it does not function well with complex geometrically formed data.

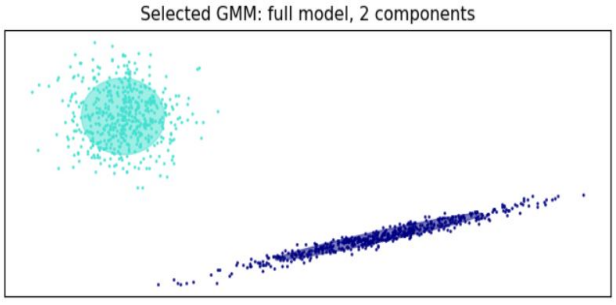
K-Means clustering and Gaussian Mixtures have no logical relationship. The data will cluster into roughly spherical clumps centered on the means of each mixture component if it fits a spherical Gaussian mixture model effectively. K-means clustering excels at the following types of data: Clusters that individually match to a mixture component will be found, with cluster centers near the mixture.
However, you can use k-means clustering without making any assumptions about the data's origins. It can be used to split up data into useful and reasonably homogeneous bits in the same way that other clustering algorithms may, with no presumption that those bits are genuine objects (eg, for market segmentation). You can prove what k-means estimates are without assuming mixed models.

Gaussian mixture models can be fitted using maximum likelihood, which is a different estimator and methodology than k-means. Alternatively, Bayesian estimators and their accompanying methodologies can be used.
Spherical Gaussian mixture models and k-means clustering are very comparable in several ways. They're not only distinct, but they're also different kinds of things.

IX. Gaussian Mixture Model Selection(Implemented Example)

This example demonstrates how Gaussian Mixture Models can be used to pick models using information-theoretic criteria (BIC). The type of covariance and the number of components in the model are also factors to consider when choosing a model. In that situation, AIC also produces the correct answer (which isn't given to save time), but BIC is more suited if the issue is determining the correct model.





```
print(__doc__)

# Number of samples per component
n_samples = 500

# Generate random sample, two components
np.random.seed(0)
C = np.array([[0., -0.1], [1.7, .4]])
X = np.r [np.dot(np.random.randn(n_samples, 2), C),
          .7 * np.random.randn(n_samples, 2) + np.array([-6, 3])]

lowest_bic = np.infty
bic = []
n_components_range = range(1, 7)
cv_types = ['spherical', 'tied', 'diag', 'full']
for cv_type in cv_types:
    for n_components in n_components_range:
        # Fit a Gaussian mixture with EM
        gmm = mixture.GaussianMixture(n_components=n_components,
                                      covariance_type=cv_type)
        gmm.fit(X)
        bic.append(gmm.bic(X))
        if bic[-1] < lowest_bic:
            lowest_bic = bic[-1]
            best_gmm = gmm

bic = np.array(bic)
color_iter = itertools.cycle(['navy', 'turquoise', 'cornflowerblue',
                              'darkorange'])
clf = best_gmm
bars = []

# Plot the BIC scores
plt.figure(figsize=(8, 6))
spl = plt.subplot(2, 1, 1)
for i, (cv_type, color) in enumerate(zip(cv_types, color_iter)):
    xpos = np.array(n_components_range) + .2 * (i - 2)
    bars.append(plt.bar(xpos, bic[i * len(n_components_range):
                                  (i + 1) * len(n_components_range)],
                        width=.2, color=color))
plt.xticks(n_components_range)
plt.ylim([bic.min() * 1.01 - .01 * bic.max(), bic.max()])
plt.title('BIC score per model')
xpos = np.mod(bic.argmin(), len(n_components_range)) + .65 +\
    .2 * np.floor(bic.argmin() / len(n_components_range))
plt.text(xpos, bic.min() * 0.97 + .03 * bic.max(), '*', fontsize=14)
spl.set_xlabel('Number of components')
spl.legend([b[0] for b in bars], cv_types)

# Plot the winner
splot = plt.subplot(2, 1, 2)
Y_ = clf.predict(X)
for i, (mean, cov, color) in enumerate(zip(clf.means_, clf.covariances_,
                                           color_iter)):
    v, w = linalg.eigh(cov)
    if not np.any(Y_ == i):
        continue
    plt.scatter(X[Y_ == i, 0], X[Y_ == i, 1], .8, color=color)

    # Plot an ellipse to show the Gaussian component
    angle = np.arctan2(w[0][1], w[0][0])
    angle = 180. * angle / np.pi  # convert to degrees
    v = 2. * np.sqrt(2.) * np.sqrt(v)
    ell = mpl.patches.Ellipse(mean, v[0], v[1], 180. + angle, color=color)
    ell.set_clip_box(splot.bbox)
    ell.set_alpha(.5)
    splot.add_artist(ell)

plt.xticks(())
plt.yticks(())
plt.title(f'Selected GMM: {best_gmm.covariance_type} model, '
          f'{best_gmm.n_components} components')
plt.subplots_adjust(hspace=.35, bottom=.02)
plt.show()
```

X. Summary and conclusion

- Géron, A., 2019. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media.
- Bishop, C.M., 2006. Pattern recognition and machine learning. springer.
- Murphy, K.P., 2012. Machine learning, a probabilistic perspective. MIT press.

Machine Learning$_A Probabilistic Perspective$