Decision Theory and Bayesian Analysis Project: A Gentle Introduction to Dirichlet Process with applications in *R*

Enes Dilber

January 5, 2018

Abstract

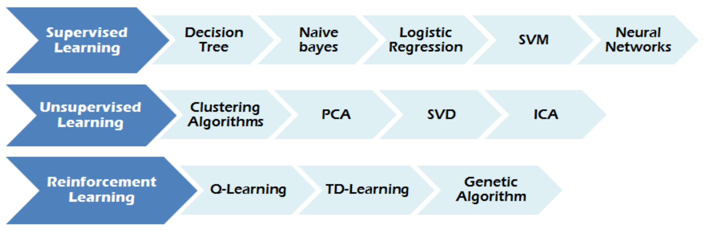
This project document aims to give a high level introduction to Dirichlet Process and its applications on Non-parametric Bayesian methods. In preliminaries section we gave brief introduction about the concepts that are used in Dirichlet Process and Bayesian Modelling. In section 2. we gave an introduction to Dirichlet Process with *R* illustrations. Since there are many methods in Dirichlet Process we only on covered: GEM, Stick Breaking and Chinese Restaurant Process. In section 3, we provided application of Dirichlet Process in Gaussian Mixture Models by finding parameter estimations using Gibbs Sampling. We also briefly mentioned further research topics in the area.

1 Preliminaries

1.1 Unsupervised Learning

There are several tasks in Machine Learning. One can see an illustration of tasks in Machine Learning, Figure 1. I am going to give Murphy's unsupervised learning definition.

Figure 1: An Illustration of Categories of Machine Learning [1]



Unsupervised learning, where we are just given output data, without any inputs. The goal is to discover "interesting structure" in the data; this is sometimes called knowledge discovery. Unlike supervised learning, we are not told what the desired output is for each input. Instead, we will formalize *our task as one of density estimation, that is, we want to build models of the form $p(x_i|\theta)$. There are two differences from the supervised case. First, we have written $p(x_i|\theta)$ instead of $p(y_i|x_i,\theta)$; that is, supervised learning is conditional density estimation, whereas unsupervised learning is unconditional density estimation. Second, x_i is a vector of features, so we need to create multivariate probability models*[2]. In general we have univariate response in supervised learning. For example in a classification problem: "what is the correct label of the input?", in a regression problem: "What is the estimated (or predicted) value of the response". In other words, for supervised learning we are trying to train our model with some input. On the other hand, in unsupervised learning there is no such variable as input, all variables are output, and we are searching underlying structure of the data. Therefore the model itself is a density estimation. Therefore unsupervised learning models usually called as generative models.

1.2 Clustering

We can call clustering as *unsupervised classification.* Because, we do not have any labels. The question is "How could we classify data?". Earlier we mentioned unsupervised learning and discussed searching underlying structure of data. In some context, we can say clustering methods trying to fit a mixture distribution to data and specific distributions at the mixture are our labels.

Example 1.1. We generated random samples from a mixture distribution of four *Bivariate Normal* distributions with mixing proportion $p_i = 0.25,\ i = 1, 2, 3, 4$. Their mean parameters are as follows: $\mu_1 = (2, 5), \mu_2 = (5, 1), \mu_3 = (0, 1), \mu_4 = (8, 2)$. All variance parameters are equal to identity matrix. We can see at Figure 2, the several fits for different k's.

The two question arises with this example. First one is: *What is the method for measuring similarity*? In this example it is quite easy $L2$ or $L1$ distances would be sufficient. But, what if our vectors are images, words, genes or even functions. The problem getting deeper when we try applying it to real life. Second problem is *What is the optimum number of clusters?.* Although, there are several methods to get a optimum value, it is hard to decide even in this synthetic data. Now imagine higher dimensions and another spaces. Task is getting harder, even impossible to keep track.

1.3 Beta, Categorical and Multinomial Distributions

1.3.1 Beta Distributions

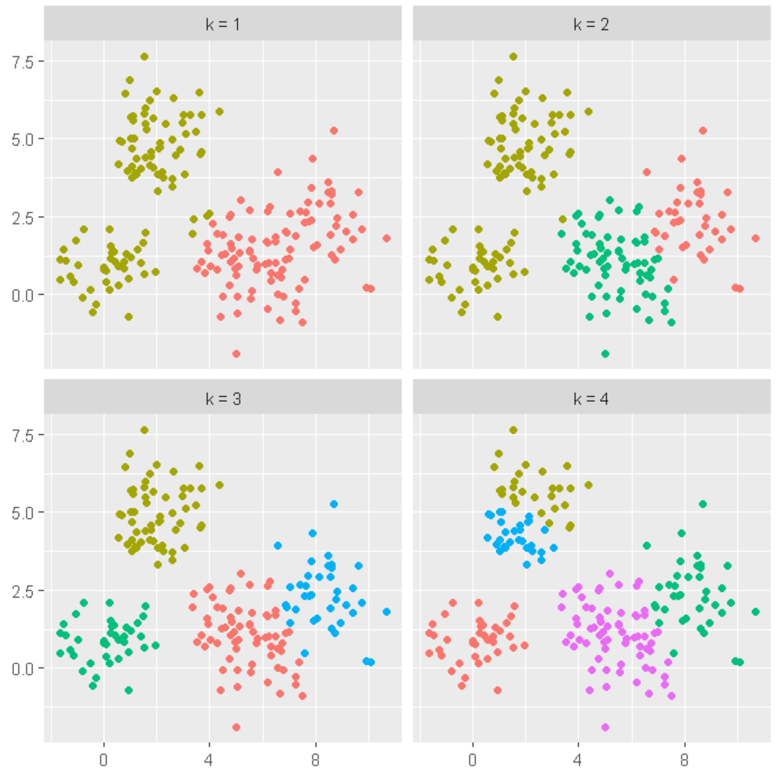
In Bayesian Analysis *Beta* distribution mostly considered as prior distribution for a probability value. Since its support $x \in (0, 1)$, it is easily be manipulated and make it to give several prior information about proportion value.

Beta pdf: $f(x;\alpha,\beta) = x^{\alpha-1}(1-x)^{\beta-1}\ \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\ \mathbb{I}(\alpha > 0)\mathbb{I}(\beta > 0)\mathbb{I}(x \in (0, 1))$

1.3.2 Categorical and Multinomial Distributions

The relationship between these two distributions are similar to *Bernoulli* and *Binomial*. These distributions are the tool that how we model clusters. Each observation will have a probability value to belong a *category*. In our case *category* will be *cluster*.

Figure 2: k-mean fits to Gaussian Mixture (repeated for several k's)



Categorical pmf: $f(x;p_1,\dots p_n) = p_1^{\mathbb{I}(x=1)} \dots p_n^{\mathbb{I}(x=n)} \; \mathbb{I}(\sum_{i=1}^n p_i = 1)$

Multinomial pmf: $f(x;n,p_1,\dots p_n) = \frac{n!}{x_1!\dots x_n!} \; p_1^{x_1} \dots p_n^{x_n} \; \mathbb{I}(\sum_{i=1}^n p_i = 1)$

1.4 Exchangeability and de Finetti's Theorem

Definition 1.1 (Exchangeability)**.** An infinite sequence $\{X_i\}_{i=1}^{\infty}$ of random variables is exchangeble if $\forall n \;= 1,2,\dots$

$$X_1,\dots X_n \overset{d}{=} X_{\pi(1)} \dots X_{\pi(n)} \quad \forall \pi \in S(n)$$

where $S(n)$ is the symmetric group, the group of permutations. Also note that *iid* $\implies$ *exchangeable* but *exchangeable* $\not\Longrightarrow$ *iid*[3]

Theorem 1.1 (De Finetti, 1931)**.** Let $x_1, x_2, x_3, \dots$ be an infinite sequence of exchangeable binary random variables. Then their joint distribution is characterized by a distribution $f(\theta)$ for a parameter $\theta \in [0,1]$ such that the x_i's are independent of θ with $P(x_i = 1 \mid \theta) = 0$.

De Finetti's theorem states that if we have a sequence of binary variables, so that each x_i takes the value 1 (success) or 0 (failure), then we can represent them as independent Bernouilli trials with probability θ of success in each trial.

1.5 Conjugate Prior

In Bayesian probability theory, if the posterior distributions $p(\theta|x)$ are in the same family as the prior probability distribution $p(\theta)$, the prior and posterior are then called conjugate distributions, and the prior is called a conjugate prior for the likelihood function[4].

One of the most common conjugates in Bayesian analysis is $Beta - Binomial$. Quite similar to that, now we have $Dirichlet - Multinomial$ and $Dirichlet - Categorical$.

1.6 Bayesian Nonparametric Models

- **Bayesian**:
  $P(parameters|data) \propto P(data|parameters)P(parameters)$
- **Non-parametric**:
  Not-finite parameters, unbounded/growing/infinite number of parameters

In Clustering problem that we mentioned earlier, or in a factor analysis; generally the question is How many *Mixture model / Factors* should I assign? The classical approach consist two component: $goodness\ of\ fit$ and *model complexity*. In Bayesian Nonparametric sense, rather than comparing models that vary in complexity, it defines the fit as a single model that can adapt its complexity to the data. Furthermore, Bayesian Nonparametric models allow the complexity to grow as more data are observed, such as when using a model to perform prediction. We provide a brief summary of NPB models that Gershman et al.[5] mentions.

1.6.1 Mixture models and clustering

In a clustering problem, algorithms like k-mean and Gaussian Mixture Models requires k, number of clusters. In BNP setting, there is no need to define the number of clusters. Bayesian mixture models contain a prior over the mixing distribution $P(c)$, and a prior over the cluster parameters. θ G_0. In a Gaussian mixture, for example, it is computationally convenient to choose the cluster parameter prior to be Gaussian. A convenient choice for the distribution on the mixing distribution is a Dirichlet (we will later discuss Dirichlet distribution). The meaning of convenience is actually conjugate prior. This generative process defines a joint distribution over the observations, cluster assignments, and cluster parameters,

$$P(y, c, \theta) = \prod_{k=1}^{K} G_0(\theta_k) \prod_{n=1}^{N} F(y_n|\theta_{c_n} P(c_n))$$

where the observations are $y = \{y_1, \ldots, y_N\}$, the cluster assignments are $c = \{c_1, \ldots, c_N\}$, and the cluster parameters are $\theta = \{\theta_1, \ldots, \theta_N\}$. The product over n follows from assuming that each observation is conditionally independent given its latent cluster assignment and the cluster parameters. Given a data set, we are usually interested in the cluster assignments, i.e., a grouping of the data, We can use Bayes rule to calculate the posterior probability of assignments given the data.

$$P(c|y) \propto P(y|c)P(c)$$

1.6.2 Latent factor models and dimensionality reduction

The most popular of these models—factor analysis (FA), principal component analysis (PCA) and independent component analysis (ICA)—all assume that the number of factors K is known. The Bayesian nonparametric variant of latent factor models allows the number of factors to grow as more data are observed. As with the BNP mixture model, the posterior distribution provides both the properties of the latent factors and how many are exhibited in the data.

For the scope of this project, we will limit our context with clustering.

2 Introduction to Dirichlet Process

2.1 Generative Model

Since our task is an unsupervised learning, we could not directly apply the bayes rule and get inference about parameters. Therefore we use generative models. Generative models are specifying a joint probability distribution over observation and label values. A conditional distribution can be formed from a generative model through Bayes' rule.

Example 2.1. Suppose a clustering task where;

- $k = 2$ (2 clusters)
- Prior of each cluster means: $\mu_k \overset{iid}{\sim} N(\mu_0, \Sigma_0)$
- Prior percentage of points from first cluster $p_1 \sim Beta(a_1, a_2)$, $p_2 = 1 - p_1$
- Assignment of each data point to *cluster* 1 or 2 is $z_n \overset{iid}{\sim} Categorical(p_1, p_2)$
- Distribution of each data point $x_n \overset{iid}{\sim} N(\mu_{z_n}, \Sigma)$

Our interest is this generative model $P(z|x_i, \boldsymbol{\theta}) = P(x_i|z, \boldsymbol{\theta})P(z|\boldsymbol{\theta})$ In our case:

- $P(x_i|z, \boldsymbol{\theta}) \sim N(\mu_{z_n}, \Sigma)$
- $P(z|\boldsymbol{\theta}) \sim Categorical(p_1, p_2) \times \ Beta(a_1, a_2) \times N(\mu_0, \Sigma_0)$

In later chapters, we are going to discuss the conjugacy of this model.

2.2 Dirichlet Distribution

In previous section we focus on a Gaussian Mixture Model with 2 components. Now consider more than 2 clusters. For this task *Beta* is useless. Luckily, we could extend *Beta* distribution with *Dirichlet* distribution. It is basically the multivariate version of *Beta*. We want to use Dirichlet distribution because it is the conjugate prior of the categorical distribution and multinomial distribution. Similar to *Beta*, its parameters are weights for expected probabilities.

Example 2.2. Let $X \sim Beta(3,7)$, then expected proportion would be $p_1 = 0.3$ and $p_2 = 0.7$. Similiarly if $Y \sim Dirichlet(3,2,5)$, expected proportion would be $p_1 = 0.3$, $p_2 = 0.2$ and $p_3 = 0.5$

```
n = 10000
X = rbeta(n, 3, 7)
X = cbind(X, 1-X)
round(colMeans(X), 3)

Y = rdirichlet(n, c(3,2,5))
round(colMeans(Y), 3)
```

Output:
Beta(3,7):
0.301 0.699
Dirichlet(3, 2, 5):
0.299 0.201 0.5

One could realize that mean of the random samples are proportional to its corresponding parameter $(E(X_i) = \alpha_i / \sum \alpha_k)$, in both distributions. This is very useful when we imagine this proportions as mixing parameter in mixture models. Without further explanation, let us give probability density function of *Dirichlet* distribution.

Dirichlet pdf: $f(\boldsymbol{x}; k, \boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma(\prod_{i=1}^{K} \alpha_i)} \prod_{i=1}^{K} x_i^{\alpha-1} \ \mathbb{I}(\alpha_i > 0)\mathbb{I}(x_i \in (0,1))$
One can observe that it is quite similar to *Beta*. And we know *Beta* is conjugate prior of *Binomial / Bernoulli*. As intuition, one could realize $Beta - Binomial$ Conjugacy could be generalized to $Dirichlet - Multinomial$.

Effect of parameter k is obvious, it defines the dimension of multivariate distribution. On the other hand, effect of α is quite crucial for our later analysis. In Example 2.2, we actually calculated expectation of *Dirichlet* by Monte Carlo. Now our attention is at $V[X_i]$ and how variation effect *Dirichlet* samples. All in all we are interested *Dirichlet* samples, because we think it as knowledge about cluster proportion. We could give variance formula of *Dirichlet*, but to illustrate its effect on clustering assignment is our main interest. Therefore we refer Example 2.3 to more understand how change in α is going to effect our prior.

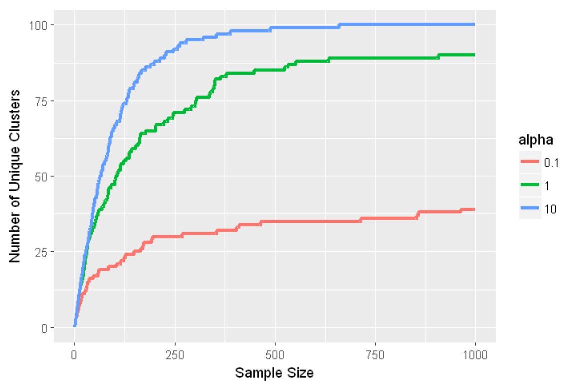
Example 2.3. We are going to sample from $Dir(\alpha_1 \dots \alpha_{100})$ while $\alpha_i = \alpha_j, \ \ \forall i, j$. Then we fix those probabilities as index selection probabilities, and select 1000 indexes. This selection is generalized form of z_n selection at 2.2, index is used as synonym for cluster number in our context. *Generate_Dir* is random sampler for *Dirichlet* and *Generate_Cat* is random sampler for *Categorical*. Here is the algorithm:

```
while a in (0.1, 1, 10) do
    $\pi_1, \dots \pi_{100} = Generate\_Dir(size = 1, k = 100, \boldsymbol{\alpha} = a)$;
    $z_1, \dots z_{1000} = Generate\_Cat(size = 1000, \boldsymbol{p} = \pi_1, \dots \pi_{100})$;
    while i in 1 ... n do
        plot(i, size(unique($z_1, \dots, z_i$)))
    end
end
```

```
n = 1000
k = 100
X = sapply(c(0.1, 1, 10), function(x) RDr(n, k, x))
df = data.frame(alpha = as.factor(rep(c("0.1", "1", "10"), each =  n)),
                x = rep(1:n, by = 3),
                y = as.vector(X))

ggplot(df, aes(x, y)) + geom_line(aes(colour = alpha), size=1.4)+
    xlab("Number of Unique Selection") + ylab("Number of Selection") +
    xlim(c(0, n)) + ylim(c(0, k))
```

In Figure 3, for a universal $\boldsymbol{\alpha}$ parameter, we plotted *Sample size* vs *Unique clusters*. It is clear that, when alpha decreases the we get more extreme values for probabilities. Therefore the selection of some indexes got high probabilities some of them got very low probabilities. Takeaway from this example is: Although there is a probability to choose every cluster, for very small $\alpha's$ we ended up with less clusters. We could call α as *concentration* parameter.

Figure 3: Output of Example 2.3



2.3 Stick Breaking

The interest of Bayesian Nonparametric is *not-fixed* number of clusters. But until this chapter, we gave examples of fixed clusters and focused on α parameter. Suppose we want an infinite number of clusters. This means $k = \infty$, but with this setting it is impossible to sample from *Dirichlet* with $k = \infty$. However we could use the relation of *Beta* and *Dirichlet* to sample from $Dirichlet(k = \infty, \boldsymbol{\alpha})$. We showed that, the α parameter vector $\propto E(\boldsymbol{\pi})$. And by using these two facts. We could define *Stick Breaking* process.

Definition 2.1. Stick Breaking

$$Let\ \pi_{1:k} \sim Dirichlet(\alpha_{1:k})$$

is equivalent to:

$$V_1 \sim Beta(a_1, \sum_{k=2}^{K} a_k) \qquad \pi_1 = V_1$$

$$V_2 \sim Beta(a_2, \sum_{k=3}^{K} a_k) \qquad \pi_2 = (1 - \pi_1)V_2$$

$$\vdots$$

$$V_{k-1} \sim Beta(a_{k-1}, a_k) \qquad \pi_{k-1} = (1 - \sum_{i=1}^{k-2} \pi_i)V_{k-1}$$

$$\pi_k = 1 - \sum_{i=1}^{k-1} \pi_i$$

We break a stick with length 1 into infinite pieces. For example; define $\alpha_i = 10\ (1/2)^{i-1}$ as a geometric series. In this setup, we got a decaying trend over index's. So problem is, what is the right way to setup our process. This brings us the next chapter.

2.4 GEM Distribution

Named by Ewens [6] after Griths, Engen and McCloskey. It is a special case of Stick Breaking process. Stick Breaking method that we mentioned earlier will definitely gave the solution but keep adding $a's$ waste of time. Instead this objective could be obtained by a simpler equation by seting $a_i = 1$ and $b_i = \alpha$

$$V_1 \sim Beta(a_1, b_1) \qquad p_1 = V_1$$

$$V_2 \sim Beta(a_2, b_2) \qquad p_2 = (1 - p_1)V_2$$

$$\vdots$$

$$V_k \sim Beta(a_k, b_k) \qquad p_k = (1 - \sum_{i=1}^{k-1} p_i)V_{k-1}$$

This violates the structure of Stick Breaking, however we are still breaking stick into pieces but this time it guarantees an exponential decay. How this works is beyond the scope of this project. One could gave adequate information from Ewens et al. [6].

GEM distribution is a multivariate continuous variable with support space $\mathbb{R}^\infty$ and summation of its support is equal to one. In other words, it's a generalized version of *Dirichlet* Distribution. It only have 1 parameter α which, similar to *Beta* and *Dirichlet*, controls the variation among supports. Let us illustrate what we mean by *variation*.

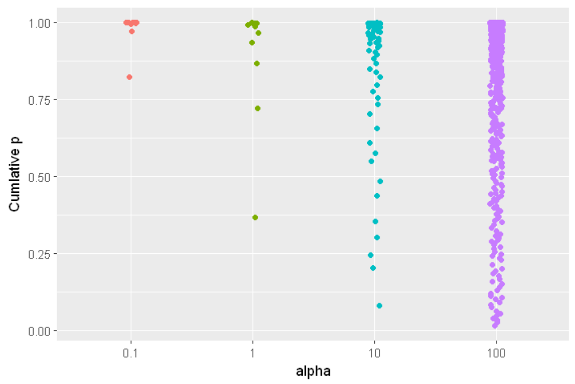
Example 2.4. We are going to sample from $GEM(\alpha)$ where α would take values $(0.1, 1, 10, 100)$ then we plot the *Stick Breaking* process.

```r
alphas = c(0.1, 1, 10, 100)
X = lapply(alphas, function(x) colMeans(rGEM(10, x)))

label = c()
probs = c()
for(i in 1:length(alphas)){
    n = length(X[[i]])
    c_label = rep(as.character(alphas[i]), n)
    label = c(label, c_label)

    c_probs = X[[i]]
    for(j in 2:n){
        c_probs[j] = c_probs[j-1] + c_probs[j]
    }
    probs = c(probs, c_probs)
}

df = data.frame(l = label, p = probs)
ggplot(df, aes(l, p)) + theme(legend.position = "none")+
  geom_jitter(width = 0.05, aes(colour = l)) + xlab("alpha")
```

For all α values, support size is infinite. Since random samples from GEM would be our cluster proportion priors, we could refer them as π_i's. In Figure 4, for all α values i goes from 1 to ∞ however, for small α's the first proportion is large and it gets smaller so fast, for $i > 10$ it gets values less than 10^{-10}. In Figure 4, for $\alpha = 1$, the sequence as follows: $\pi_1 = 0.37, \pi_2 = 0.72 - (0.37) = 0.35, \pi_3 = 0.86 - (0.72) = 0.14$ and so on. So theoretical we have infinite support, but in practice we only deal with a small number of vales. When α gets larger, we got so many π_i's in practice. If we consider them as mixing proportion, or prior cluster probability, we get more and more cluster when we increase α.

One could realize now, the previous π_i effects π_{i+1}, so it is a *Stochastic Process* more than a distribution.

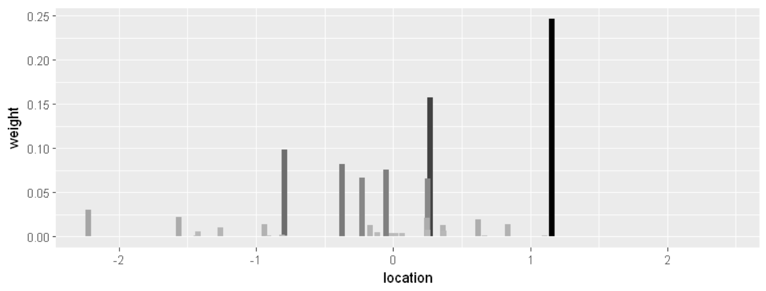
Figure 4: Output of Example 2.4



2.5 Dirichlet -*Stochastic*- Process

So far, we generate an $infinite$ sequence of probabilities such that $\sum_{i=1}^{\infty} \pi_i = 1$ from $GEM(\alpha)$. And suppose we sampled infinite number of points corresponding to each π_i from a distribution G_0, call them θ_i. This is "draw" from *Dirichlet Process* with *weights* $\boldsymbol{\pi}$ and *location* $\boldsymbol{\theta}$. Let us make it clear with an example. Suppose $G_0 = N(0,1)$ and $alpha = 5$. A sample from $DP(5, N(0,1))$ would look like Figure 5. Therefore if we continue sampling, density of it would have a similar -but discrete-shape to $N(0,1)$. With each sampling we obtain a different shape than rather Figure 5. So it is random. Although we could not see majority of weights in the figure, -theoretically- we have infinitely many of them.

Figure 5: A random sample from $DP(5, N(0,1))$



So far, we sampled from a *Dirichlet Process.* Now let us mention its distribution more deeply.

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$$

$$G \sim DP(\alpha, G_0)$$

- G is a random measure
- ϕ's are random samples from G_0
- π's are random samples from GEM

We will define Ω which is the underlying space of ϕ_k. So in our previous example it was R. For a fixed G, random samples are a measure. Suppose $A \subset \Omega$. We could obtain measure of $G(A)$ for a particular G.

$$G(A) = \sum_{k} \pi_k \delta_{\phi_k}(A)$$
$$= \sum_{k:\phi_k \in A} \pi_k$$

Therefore, for a fixed G we can get a measure. For several G's we get random measures. Which is index collection of random variables. In classical stochastic process concept *time* would be the index value. For example in *Brownian Motion*, each X_t is a random variable. For each run, X_t takes different values according to its *Gaussian* distribution. *Dirichlet* also a stochastic process but its index set is not time. For a fixed $A \subset \Omega$, G(A) is random. If we fixed A and draw $G \sim DP(\alpha, G_0)$, in each draw we get different amount of weights drop the A. A is unbounded, infinite index set. Therefore, index set is subsets of underlying set. Hence we could say, like X_t in *Brownian Motion*, each $G(A)$ have a distribution.

For the scope of this project, we are not going to give more details about $\sigma - algebra$ and topology perspectives of *Dirichlet Process.* We could refer Bayesian Nonparametrics book [7], which gives complete background and most common application areas of BNP.

3 Dirichlet Process Gaussian Mixture Models

We gave definition of *Dirichlet Process* and intuition behind it. In this section we focus on our clustering problem. First of all it is an infinite Gaussian Process, which means we will not limit the number of components in the mixture.

3.1 Generating data points by Dirichlet Process

Remember the setting at Example 2.1. Now the case is:

- $k = \infty$ (infinitely many clusters)
- Prior of each cluster means: $\mu_k \overset{iid}{\sim} N(\mu_0, \Sigma)$
- Expected percentage of points from k^{th} cluster $(\pi_1, \dots, \pi_k, \dots, \pi_\infty) \sim GEM(\alpha)$
- Assignment of each data point to clusters $z_n \overset{iid}{\sim} Categorical(\boldsymbol{\pi})$
- Distribution of each data point $x_n \overset{iid}{\sim} N(\mu_{z_n}, \Sigma_0)$

Here is our selection sequence:

1. Sample and fix selection probabilities $\boldsymbol{\pi} \sim GEM(\alpha)$
2. Generate infinitely many μ's where $\mu_k \sim N(\mu_0, \Sigma)$
3. Select N points z_n where $z_n \sim Categorical(\boldsymbol{\pi})$
4. Sample N points x_n from $N(\mu_{z_n}, \Sigma_0)$

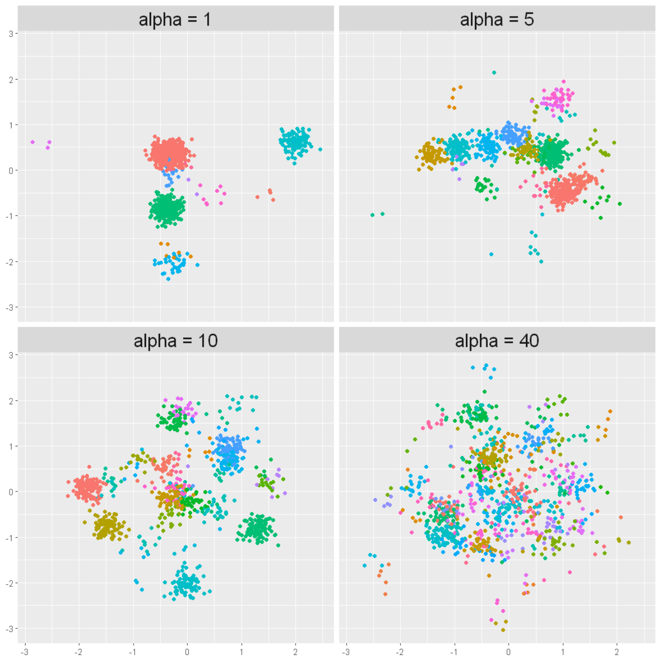
The equivalent way to draw GEM and draw the cluster locations μ_k is to draw from *Dirichlet Process*. So it gives two things, *locations* of atoms. So we could just write $\mu_n \overset{iid}{\sim} G$ and sample this $x_n \overset{indep}{\sim} N(\mu_n, \Sigma_0)$. x_n's are not identical, they are coming from different locations, but they are independent from each other. So ordering is not important, this is called *exchangebility*.

Example 3.1. In this example, we are going to sample data points by using *Dirichlet Process*. In our example, $G_0 = N_2(\mu = (0,0), \Sigma = 2)$ and suppose Σ_0 is known and equal to 0.1 Let us generate 1000 data points by *Dirichlet Process*. We would run process for several α's
.

```r
n = 1000; mu = c(0,0); Sigma = diag(2);

a = c(1, 5, 10, 40)

df = data.frame(0, 0, 0, 0)
colnames(df) = c("X1", "X2", "c", "alpha")

for(i in 1:length(a)){
    alpha = a[i]
    p = rGEM(1, alpha, mySeed = 108)
    z = rcat(n, p[1:length(p)])
    m = rmvnorm(length(p), mu, Sigma)

    x = matrix(0, n, 2)
    for(i in 1:n){
        x[i,] = rmvnorm(1, m[z[i],], Sigma*0.02)
    }
    x = as.data.frame(x)
    x$c = as.factor(z)
    x$alpha = paste("alpha =", alpha)
    colnames(x) = c("X1", "X2", "c", "alpha")
    df = rbind(df, x)
}
df = df[-1,]
df$alpha = as.factor(df$alpha)
df$alpha = factor(df$alpha, levels(df$alpha)[c(1,4,2,3)])

ggplot(df, aes(X1, X2, c)) + geom_point(aes(color = c)) + theme(legend.position = "none")+
facet_wrap(~as.factor(alpha), ncol=2) + xlab("") + ylab("")
```

Since we focus on generative models, we try to generate data points. We believe, in a specific clustering problem, we could find underlying cluster allocation distribution with this process. In Figure 6, we could observe the change of behaviour in DP with a $Bi-variate\ Gaussian$ base for several α values. Since we are interested in a Bayesian Setting, later we update our prior generative model with data. Like in all Bayesian methods, our posterior will be a weighted average of prior and posterior. The generated points are like prior locations of data points and we try to find a relation between those generations and data, therefore we update our information.

Figure 6: Output of Example 3.1



3.2 Chinese Restaurant Process

It is again a Stochastic Process, a special case of *Dirichlet Process* but its index set is the permutations of customers. Lets illustrate this with the famous example. Imagine a Chinese restaurant, where there are infinitely many tables. Each customer who enters the restaurant, either sits an existing table or starts a new one. For example, first customer comes in and starts a new table, because there is no open table. Then second customer comes, s/he either sits the fist customers table or starts a new one. Process continues like this. Here is the rule:

$$P(customer\ n+1\ joins\ table\ c \mid \pi_{[n]}) = \frac{|c|}{\alpha+n}$$

Where, c is existing tables, $|c|$ is the size of the table. α is the probability of opening a new table, n is the number of customers in total. Suppose a configuration like this:

$$\pi_{[10]} = \{\{3,5\},\{1,2,9,10\},\{4,6,7\},\{8\}\}$$

And customer 11 enters the restaurant, s/he will sit table 1, 2, 3, 4 with probability $2/(10+\alpha), 4/(10+\alpha), 3/(10+\alpha), 1/(10+\alpha)$, respectively. And s/he opens a new table with probability $\alpha/(1+\alpha)$. Probability of starting your own table goes down. In addition large tables become larger due to *the selection probability* $\propto$ *size of the table.*

Example 3.2. Excangibility of Chinese Restaurant Process Suppose a size six configuration:

$$\pi_{[6]} = \{\{1,2,5\},\{3,4\},\{6\}\}$$

Let us denote k^{th} person joins table c with $k \to c$ and s/he opens a new table $k \to T$.

$$P(\pi_{[6]}) = \underbrace{\frac{\alpha}{(\alpha)}}_{1\to T} \underbrace{\frac{1}{(1+\alpha)}}_{2\to 1} \underbrace{\frac{\alpha}{(2+\alpha)}}_{3\to T} \underbrace{\frac{1}{(3+\alpha)}}_{4\to 2} \underbrace{\frac{2}{(4+\alpha)}}_{5\to 1} \underbrace{\frac{\alpha}{(5+\alpha)}}_{6\to T} = \frac{\alpha^3}{\alpha^{(6)}} \prod_{c\in\pi_{[6]}} (|c|-1)!$$

This formula is brings us to Exchangeable Partition Probability Function (EPPF). Ordering does not effect the partition. For instance, $\pi_{[6]} = \{\{4,1\},\{1,5,2\},\{3\}\}$ has the same probability with our example. So we could generalize the formula as this:

$$P(\pi_{[N]}) = \frac{\alpha^n}{\alpha^{(N)}} \prod_{c \in \pi_{[N]}} (|c|-1)!$$

It is the probability distribution of the object. It will be our prior at the modeling part.

Important thing about Example 3.2 is CRP derived probability of observation in cluster is exchangeable. It will be used at Gibbs Sampling part. We will delete an observation in table and add them to process like they are the one who comes to restaurant last. We can always pretend some specific customer ω is the last customer and calculate $P(\pi_{[N]}|\pi_{[N],-\omega})$.

3.3 Gibbs Sampling for Gaussian Mixture Model with CRP Prior

Let us construct our GMM with CRP prior. Here is the predictive model for cluster assignment.

$$\underbrace{P(z_i = k|z_{-i}, \boldsymbol{X}, \alpha, \beta)}_{\text{Posterior}} \propto \underbrace{P(z_i = k|z_{-i}, \alpha)}_{\text{Prior}} \underbrace{P(x_i|\boldsymbol{X}_{-i}, z_i = k, z_{-i}|\beta)P(\beta|M_N, \kappa_N, \eta_N.S_N)}_{\text{Likelihood}}$$

As one can see, it is a hierarchical model that relies on seven hyper parameters: $\alpha, \beta = (\mu, \Sigma), M_N, \kappa_N, \eta_N.S_N$. There are also μ_k and Σ_k, they are learnable parameters. However, we do not need them in Gibbs Sampling for posterior predictive.

3.3.1 Prior

Prior is probability distribution of which table the costumer i is going to sit.

$$P(z_i = k|z_{-i}, \alpha) = \begin{cases} \frac{N_{k,-i}}{\alpha+N-1} & if\ k\ is\ existing\ table \\ \frac{\alpha}{\alpha+N-1} & if\ k\ is\ a\ new\ table \end{cases}$$

This is exactly the same case with Example 3.2, with a slight different notation. z_{-i} implies all table assignments except i^{th} customer. $N_{k,-i}$: size of the table k without i^{th}. The rest is the same. Of course with each a new table opening, we update the size of the cluster. That freedom enable us to fit the model to data with an adaptive sized cluster.

3.3.2 Likelihood

Let us start with β. It is the set of parameters that are priors on table parameters. In other word, we have distributions over cluster centers. Location is distributed as $Normal$, and those clusters also have precision, which are distributed as $Inverse\ Wishart$. Together, their joint distribution is $Normal\ Inverse\ Wishart$. Therefore $\beta = (\boldsymbol{\mu}, \boldsymbol{\Sigma}) \sim NIW_D(M_N, \kappa_N, \eta_N.S_N)$. D is the dimension of the NIW.

Now, we could begin *likelihood* simplification. Here is our likelihood.

$$P(x_i|\boldsymbol{X}_{-i}, z_i = k, z_{-i}, \beta)$$

We could denote the set $(\boldsymbol{X}_{-i}, z_i = k, z_{-i}, \beta)$ as $(\boldsymbol{X}_{(k,-i)}, \beta)$. Both of them indicate that: Intersection of the set of customers in table k without i^{th} person and β. Now with a given set of β, the likelihood illustrate this: what is the probability of this particular customer to sit a table with the other people who sits that table. Again *Chinese Restaurant Process* is a great analogy for this clustering task. Because imagine the locals in that area. Some of them know each other and some of them do not. So we are not just randomly assign them to a table or opens a new one, but also consider the people who sits at that table. We could rewrite the likelihood:

$$P(x_i|\boldsymbol{X}_{(k,-i)}, \beta) = \frac{P(\boldsymbol{X}_k|\beta)}{P(\boldsymbol{X}_{(k,-i)}|\beta)}$$

This is a likelihood ratio and it gives posterior predictive. It is $multivariate\ student's\ t$ with parameters

$$\big(\mu_N, \frac{\kappa_N + 1}{\kappa(\eta_N - D + 1)} S_N, \eta_N - D + 1\big)$$

Beside the cluster predictive assignment, we need to update μ and Σ in every cluster iteration. Because they are unique to the clusters and the data points. In a Gibbs Sampling iteration for a single data point:

$$P(\mu, \Sigma|y_i) = \prod_{\forall i\ s.t.\ c_i = k} P(y_i|\mu, \Sigma) P(\mu, \Sigma)$$

Since $\mu, \Sigma|y_i \sim N(\mu, \Sigma)$ and $\mu, \Sigma \sim NIW(M_N, \kappa_N, \eta_N.S_N)$ they are conjugate, and resulting distribution again NIW. Updates are:

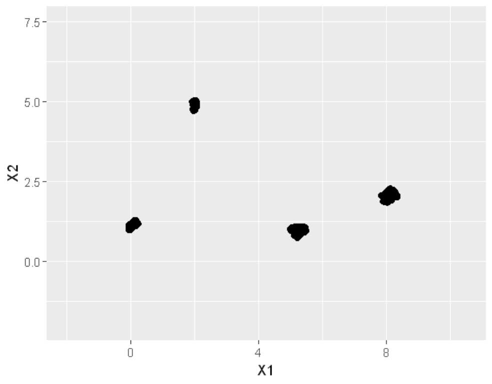
$$M_N = \frac{\kappa_n M_N + N\bar{y}}{\kappa_N + N}$$
$$\kappa_N = \kappa_N + N$$
$$\eta_N = \eta_N + N$$

One could get more detailed derivation in here [8].

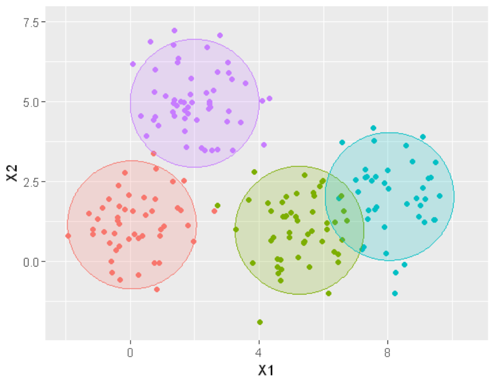
Example 3.3. We are going to construct a simpler example. Suppose we have a 2-dimensional sample with known $\Sigma =$ I. Which means, we fixed covariance matrix of each cluster. Suppose cluster means coming from a $N_2(0, 9\text{I})$. And let $\alpha = 0.01$. We use Gibbs Sampling:

$$\underbrace{P(z_i = k|z_{-i}, \boldsymbol{X}, \alpha, \mu_k)}_{\text{Posterior}} \propto \underbrace{P(z_i = k|z_{-i}, \alpha)}_{\text{Prior}} \underbrace{P(x_i|\boldsymbol{X}_{-i}, z_i = k, z_{-i}|\mu_k) P(\mu_k|\mu_0 = 0, \Sigma_0 = 9\text{I})}_{\text{Likelihood}}$$

and we update cluster mean location by using similar equation with *likelihood* section. After burn-in period we draw the location values at Figure 7a. In Figure 7b we could the see the fit of the model. We draw the circles, their centers are the averages of locations that are sampled by Gibbs Sampling. Circle radius= 2Σ.



(a) Gibbs Sampling values of Cluster Means



(b) Cluster Fits of CRP GMM

Figure 7: Plot of Gibbs Sampling Cluster Algorithm

```r
x = gen_BVN(n, Mu, mixp = c(0.25, 0.25, 0.25, 0.25), mySeed = 1) #sample our syntetic data

myX = data.frame(X1= x[,1],X2 = x[,2])

# run a CRP Gibbs sampler with 100 iterations and initialized all data points in the same cluster
GMMfit = ex6_crp_gibbs(data=data.matrix(x), sd=1, sd0 = 3 ,initz=rep(1,n), MI = 100, mySeed = 1)
df = GMMfit

df = data.frame(df[-(1:15000),]) #discard burn-in

ggplot(data = df, aes(X3, X4))+geom_point() +coord_fixed() + xlab("X1") + ylab("X2") +
  xlim(c(-2, 10.5)) +ylim(c(-2, 7.5))

#below code is just for illustration purposes
k = 4
kk = kmeans(df[,c(3,4)], k, nstart = 4) #to find mean of each cluster, we cluster gibbs returns
df$clusters = kk$cluster
points = data.frame(cbind(df$X2, dummy.code(df$clusters)))
points = aggregate(points[,-1], list(points$V1), mean)
myX$cluster = as.factor(apply(points[,-1], 1,  which.max))

circles = data.frame( #draw standard deviation circles
  x0 = kk$centers[,1],
  y0 =  kk$centers[,2],
  r = rep(2, k),
  l = as.factor(1:k)
)

options(repr.plot.width=5.2, repr.plot.height=4)
ggplot() + geom_point(data = myX, aes(X1, X2, color = cluster)) + theme(legend.position = "none") +
  geom_circle(data =circles, aes(x0 = x0, y0 =y0, r = r, fill = l, color = l), alpha = 0.2) +
  xlim(c(-2, 10.5)) +ylim(c(-2, 7.5))
```

For more realistic fits, we could put a prior distribution over Σ. Even, one can add a prior distribution for α parameter [9]. Also Gibbs sampling is not the only solution for inference. For example, in the case of we could not find a non-conjugate prior, metropolis-hasting algorithm could be the answer. In addition one could fit its model by using *Variational Inference* [10]. Also it has a sckit-learn implementation in python [11].

4 Discussion

We tried to explain the basic intuition behind Bayesian Non-parametrics with toy examples. Although we only covered Gaussian Mixture models, the application of Dirichlet Process is not limited by GMM's. It is also powerful for many areas, because it enables the researcher to reshape the process for a specific type of data. It is a distribution over distributions. For example in topic modeling, they use word distributions. The famous Latent Dirichlet Allocation for topic discovery paper has published by David Blei, Andrew Ng, and Michael I Jordan in 2003 [12], basically does that. Here are some recent research areas that uses Dirichlet Process:

- Natural Language Processing: *Topic Modelling*
- Graph Theory: *Social Networks*
- Bionformatics: *Gene Expression Clustering*
- Biostatistics: *Modeling Biodiversity*
- Bayesian Networks: *Hidden Markov Models*

5 Acknowledgements

In addition to references, this project done by the help of Michael I. Jordan and Tamara Broderick's seminar. The name of the seminar is Nonparametric Bayesian Methods: Models, Algorithms, and Applications [13]. It made me familiar with the subject.

Also one could reach the R notebook of the project at enesdilber.github.io/565.html

References

[1] THE UPX ACADEMY. Types of machine learning, 2017. URL `https://upxacademy.com/wp-content/uploads/2016/06/Machine-Learning.png`. [Online; accessed December 31, 2017].

[2] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[3] Sayan Mukherjee. Lecture notes in exchangeability and de finetti representation, 2009.

[4] Robert Schlaifer Howard Raiffa. Applied statistical decision theory. *Journal of the American Statistical Association*, 57(297):199–202, 1962.

[5] Samuel J. Gershman and David M. Blei. A tutorial on bayesian nonparametric models. *Journal of Mathematical Psychology*, 56(1):1–12, 2012.

[6] W. J. Ewens. *Population Genetics Theory - The Past and the Future*, pages 177–227. Springer Netherlands, 1990. ISBN 978-94-009-0513-9.

[7] *Bayesian Nonparametrics*. Cambridge University Press, 2010.

[8] Herman Kamper. Gibbs sampling for fitting finite and infinite gaussian mixture models, 2013.

[9] Michael D Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588, 1995.

[10] David M Blei, Michael I Jordan, et al. Variational inference for dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143, 2006.

[11] Andreas Mueller. Dpgmm classifier: Infinite gaussian mixtures, 2015. URL `http://scikit-learn.sourceforge.net/dev/modules/generated/sklearn.mixture.DPGMM.html`.

[12] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[13] Michael I. Jordan Tamara Broderick. Nonparametric bayesian methods: Models, algorithms, and applications, 2017. URL `https://simons.berkeley.edu/talks/nonparametric-bayesian-methods`. [Online; accessed January 4, 2018].