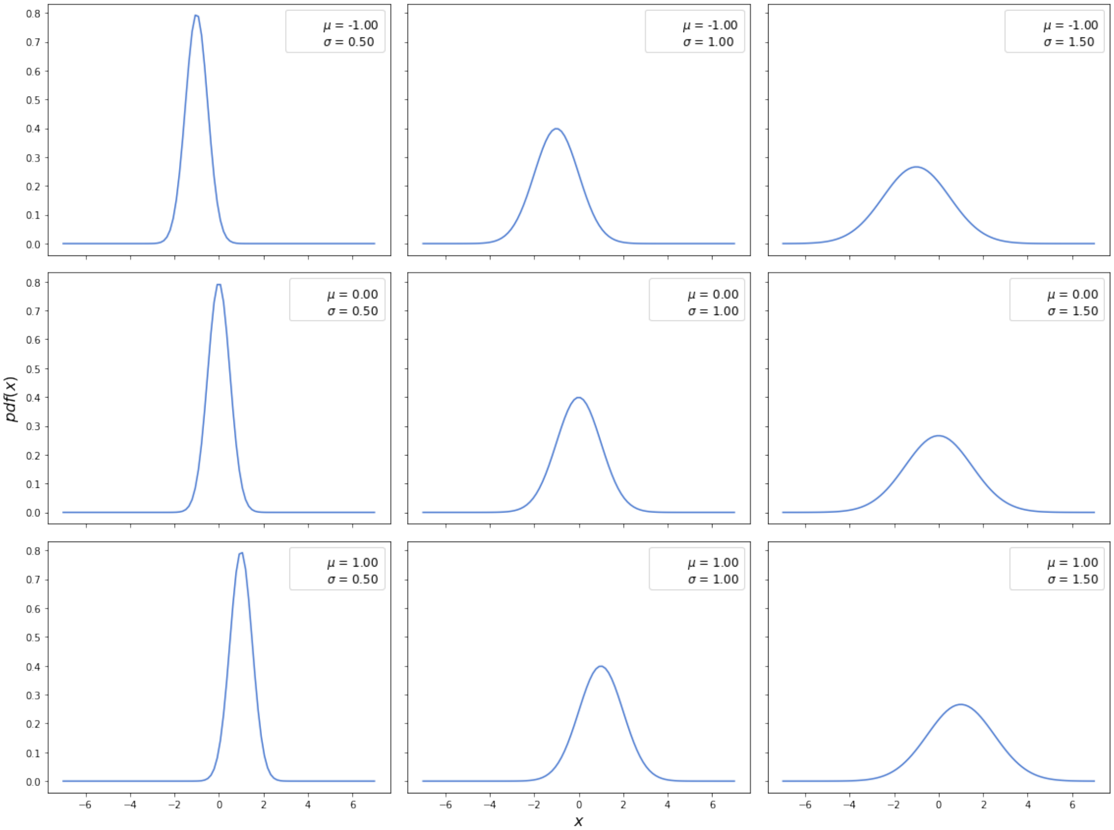
Bayesian Data Analysis Chapter 1

January 19, 2019

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from scipy import stats
        import seaborn as sns
        palette = 'muted'
        sns.set_palette(palette); sns.set_color_codes(palette)

        figureSize = (16,12)
```

0.1 Normal distribution (iid variable)

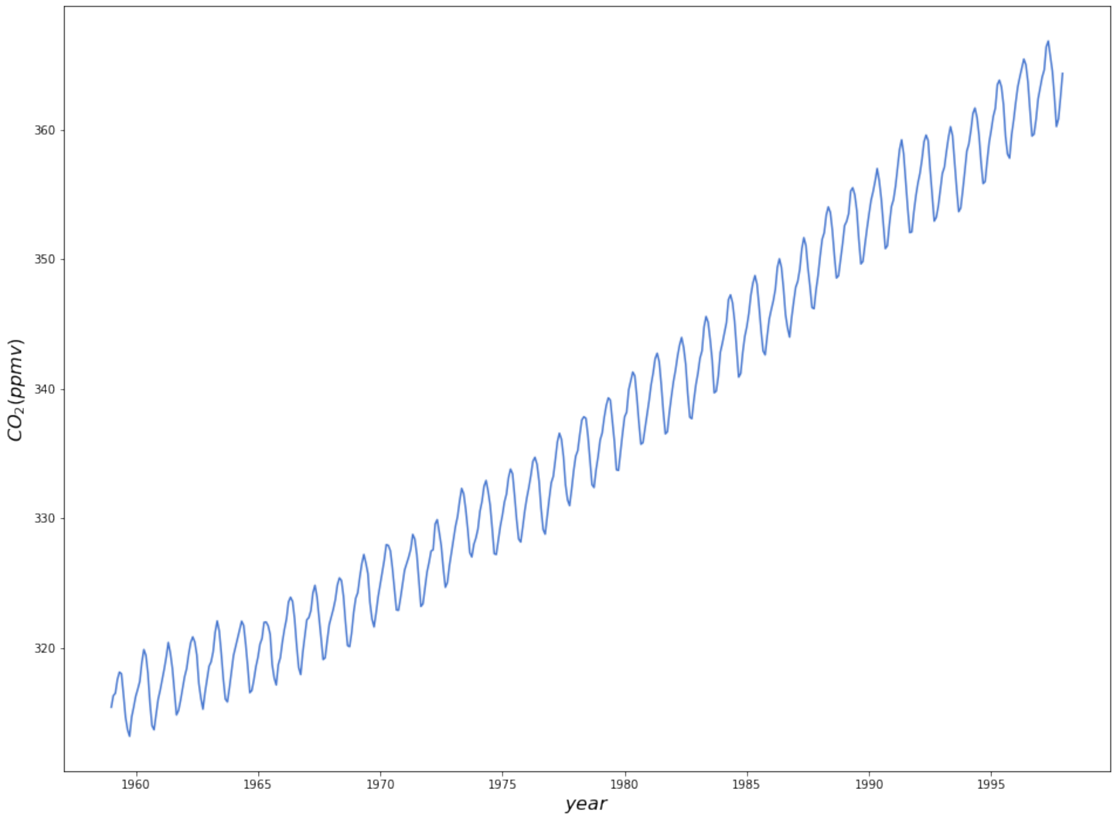
```
In [2]: mu_params = [-1, 0, 1]
        sd_params = [0.5, 1, 1.5]
        x = np.linspace(-7, 7, 100)
        f, ax = plt.subplots(len(mu_params), len(sd_params), sharex=True, sharey=True,figsize=
        for i in range(3):
            for j in range(3):
                mu = mu_params[i]
                sd = sd_params[j]
                y = stats.norm(mu, sd).pdf(x)
                ax[i,j].plot(x, y)
                ax[i,j].plot(0, 0,
                label="$\\mu$ = {:3.2f}\n$\\sigma$ = {:3.2f}".format(mu, sd), alpha=0)
                ax[i,j].legend(fontsize=12)
        ax[2,1].set_xlabel('$x$', fontsize=16)
        ax[1,0].set_ylabel('$pdf(x)$', fontsize=16)
        plt.tight_layout()
```



0.2 Non iid variable

```
In [3]: data = np.genfromtxt('mauna_loa_CO2.csv', delimiter=',')
        f, ax = plt.subplots(figsize=figureSize)
        ax.plot(data[:,0], data[:,1])
        ax.set_xlabel('$year$', fontsize=16)
        ax.set_ylabel('$CO_2 (ppmv)$', fontsize=16)
```

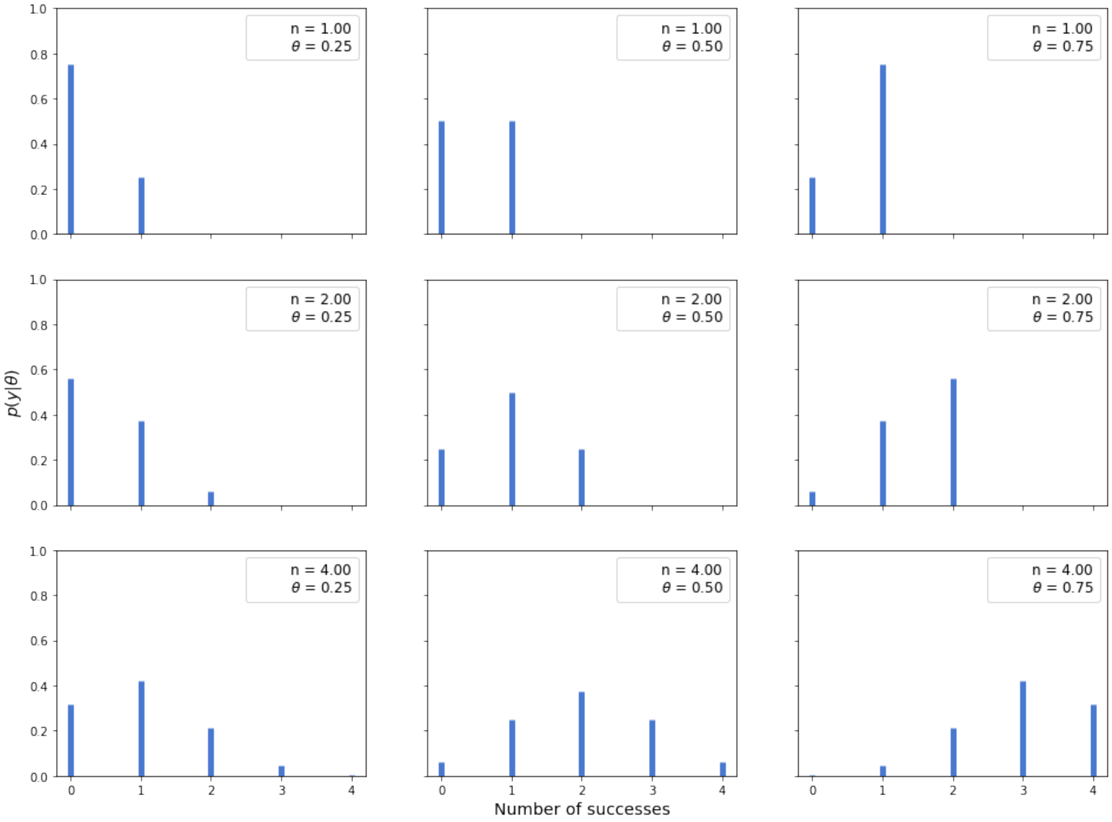
```
Out[3]: Text(0, 0.5, '$CO_2 (ppmv)$')
```



```
In [4]: # n_params ->
        n_params = [1, 2, 4]
        # p_parameters -> probability of sucess
        p_params = [0.25, 0.5, 0.75]
        # x-> number of sucesses
        x = np.arange(0, max(n_params)+1)
        f, ax = plt.subplots(len(n_params), len(p_params), sharex=True,
          sharey=True, figsize=figureSize)
        for i in range(3):
            for j in range(3):
                n = n_params[i]
                p = p_params[j]
                # probability of number of sucesses x, with n number of tosses and p probabili
                y = stats.binom(n=n, p=p).pmf(x)
                #matplotlib.pyplot.vlines(x, ymin, ymax)
                ax[i,j].vlines(x, 0, y, colors='b', lw=5)
                ax[i,j].set_ylim(0, 1)
                ax[i,j].plot(0, 0, label="n = {:3.2f}\n$\\theta$ = {:3.2f}".format(n, p), alpha
                ax[i,j].legend(fontsize=12)
        ax[2,1].set_xlabel('Number of successes', fontsize=14)
        ax[1,0].set_ylabel('$p(y|\\theta)$', fontsize=14)
```

```
ax[0,0].set_xticks(x)
```

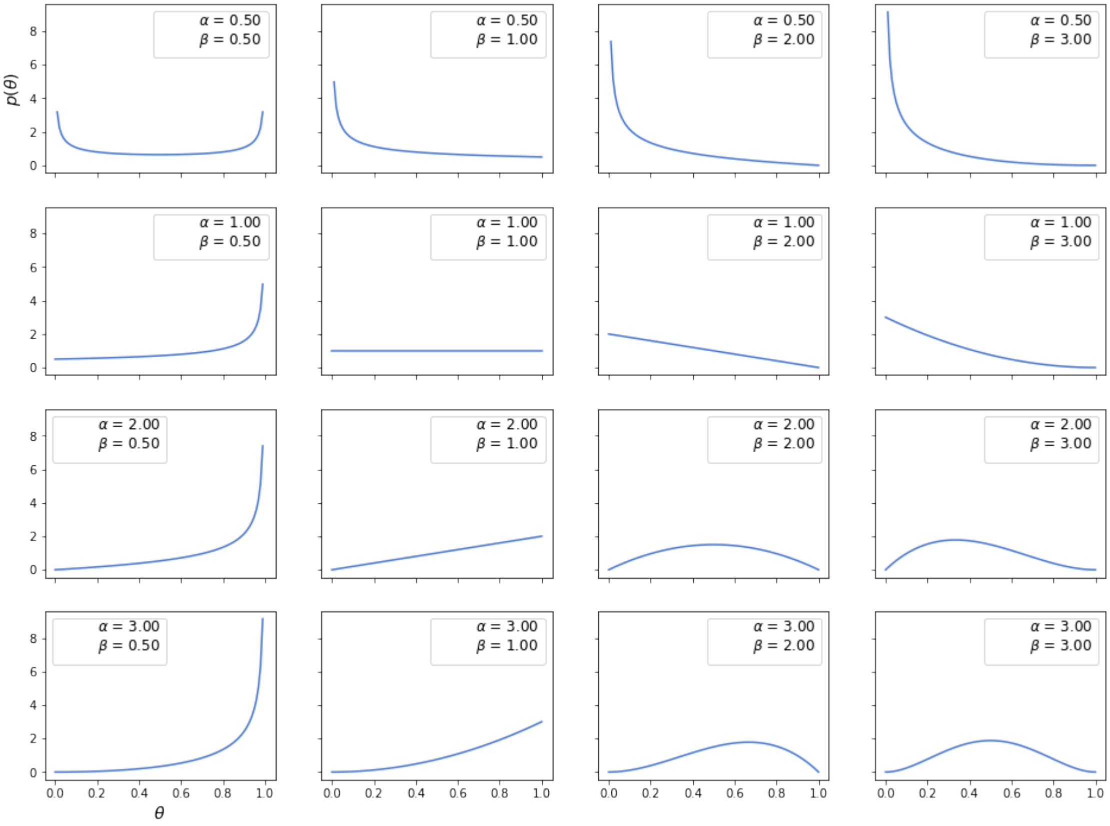
```
Out[4]: [<matplotlib.axis.XTick at 0x7f19b8eeefd0>,
         <matplotlib.axis.XTick at 0x7f19b8eee630>,
         <matplotlib.axis.XTick at 0x7f19b8eee8d0>,
         <matplotlib.axis.XTick at 0x7f19b2e474e0>,
         <matplotlib.axis.XTick at 0x7f19b2e47978>]
```



0.3 Coin toss experiments

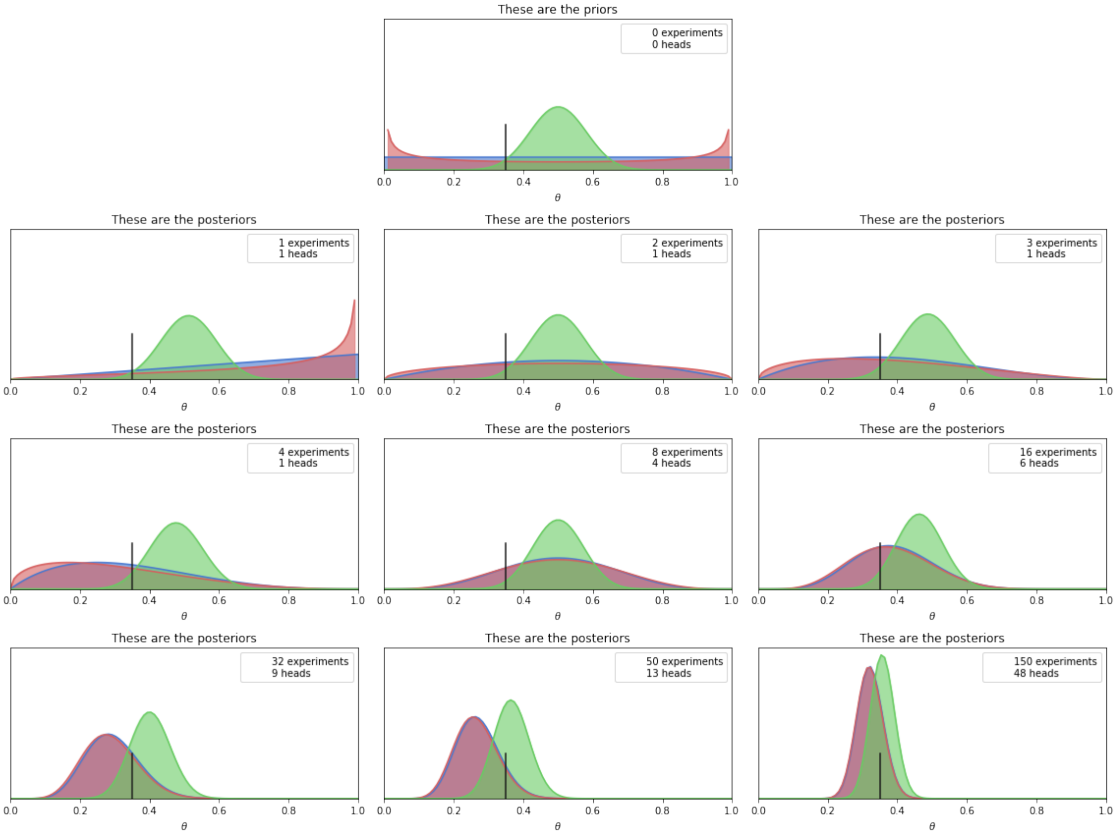
```
In [5]: params = [0.5, 1, 2, 3]
        x = np.linspace(0, 1, 100)
        f, ax = plt.subplots(len(params), len(params), sharex=True,
          sharey=True, figsize=figureSize)
        for i in range(4):
            for j in range(4):
                a = params[i]
                b = params[j]
                y = stats.beta(a, b).pdf(x)
                ax[i,j].plot(x, y)
                ax[i,j].plot(0, 0, label="$\\alpha$ = {:3.2f}\n$\\beta$ = {:3.2f}".format(a, b
                ax[i,j].legend(fontsize=12)
        ax[3,0].set_xlabel('$\\theta$', fontsize=14)
        ax[0,0].set_ylabel('$p(\\theta)$', fontsize=14)
```

```
Out[5]: Text(0, 0.5, '$p(\\theta)$')
```



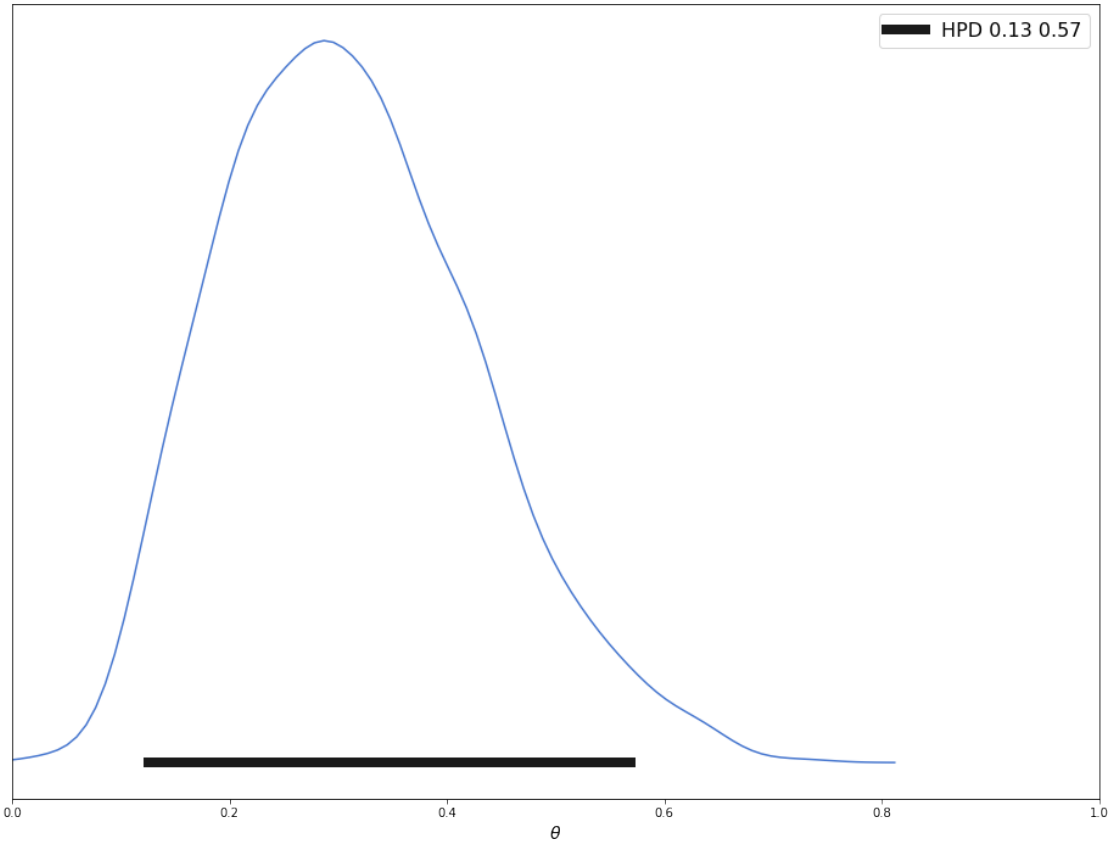
0.4 Priors and posteriors

```
In [6]: import matplotlib

        theta_real = 0.35
        trials = [0, 1, 2, 3, 4, 8, 16, 32, 50, 150]
        data = [0, 1, 1, 1, 1, 4, 6, 9, 13, 48]

        beta_params = [(1, 1), (0.5, 0.5), (20, 20)]
        dist = stats.beta
        x = np.linspace(0, 1, 100)

        for idx, N in enumerate(trials):
            if idx == 0:
                plt.subplot(4,3, 2)
                plt.title('These are the priors')
            else:
                plt.subplot(4,3, idx+3)
                plt.title('These are the posteriors')
            y = data[idx]
            for (a_prior, b_prior), c in zip(beta_params, ('b', 'r', 'g')):
                p_theta_given_y = dist.pdf(x, a_prior + y, b_prior + N - y)
                plt.plot(x, p_theta_given_y, c)
                plt.fill_between(x, 0, p_theta_given_y, color=c, alpha=0.6)


            plt.axvline(theta_real, ymax=0.3, color='k')
            plt.plot(0, 0, label="{:d} experiments\n{:d} heads".format(N, y), alpha=0)
            plt.xlim(0,1)
            plt.ylim(0,12)
            plt.xlabel(r"$\theta$")
            plt.legend()
            plt.gca().axes.get_yaxis().set_visible(False)

        fig = matplotlib.pyplot.gcf()
        fig.set_size_inches(figureSize[0],figureSize[1])
        plt.tight_layout()
```



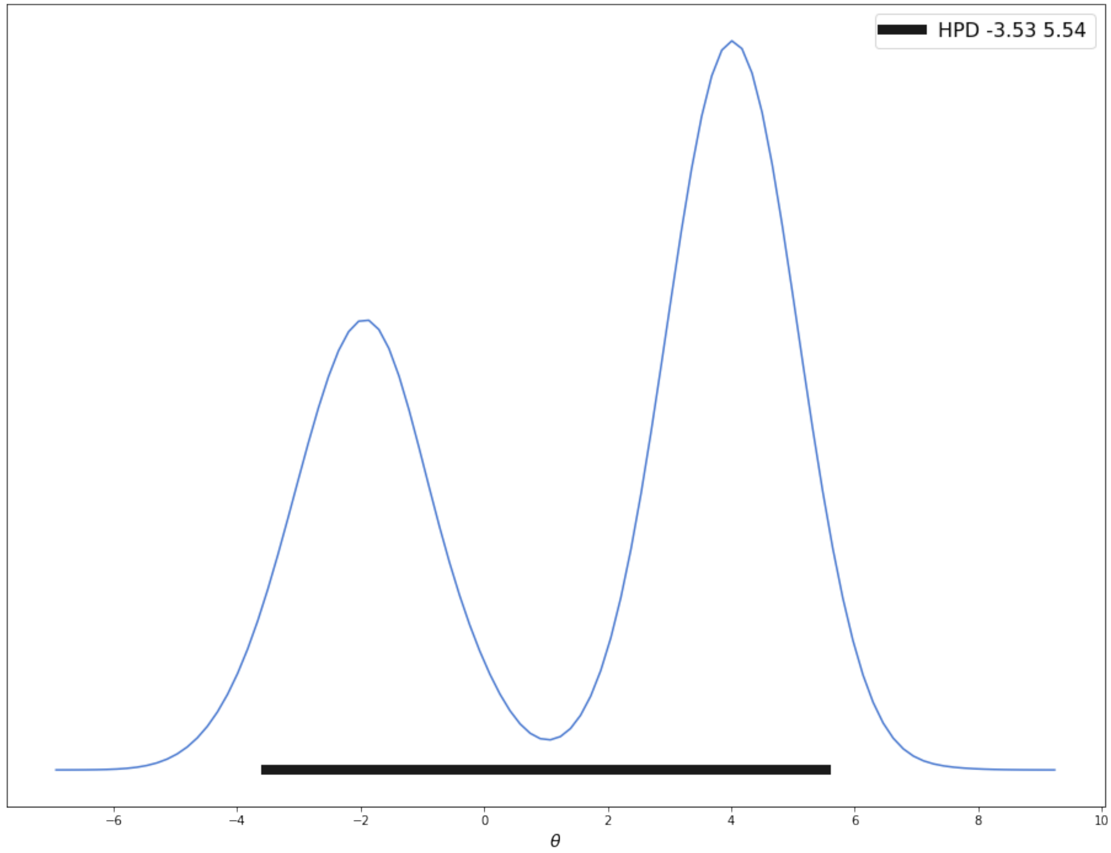
0.5 Naive calculation of HPD

```
In [7]: def naive_hpd(post):
            sns.kdeplot(post)
            HPD = np.percentile(post, [2.5, 97.5])
            plt.plot(HPD, [0, 0], label='HPD {:.2f} {:.2f}'.format(*HPD),
              linewidth=8, color='k')
            plt.legend(fontsize=16);
            plt.xlabel(r"$\theta$", fontsize=14)
            plt.gca().axes.get_yaxis().set_ticks([])


        np.random.seed(1)
        post = stats.beta.rvs(5, 11, size=1000)
        naive_hpd(post)
        plt.xlim(0, 1)
        fig = matplotlib.pyplot.gcf()
        fig.set_size_inches(figureSize[0],figureSize[1])
```



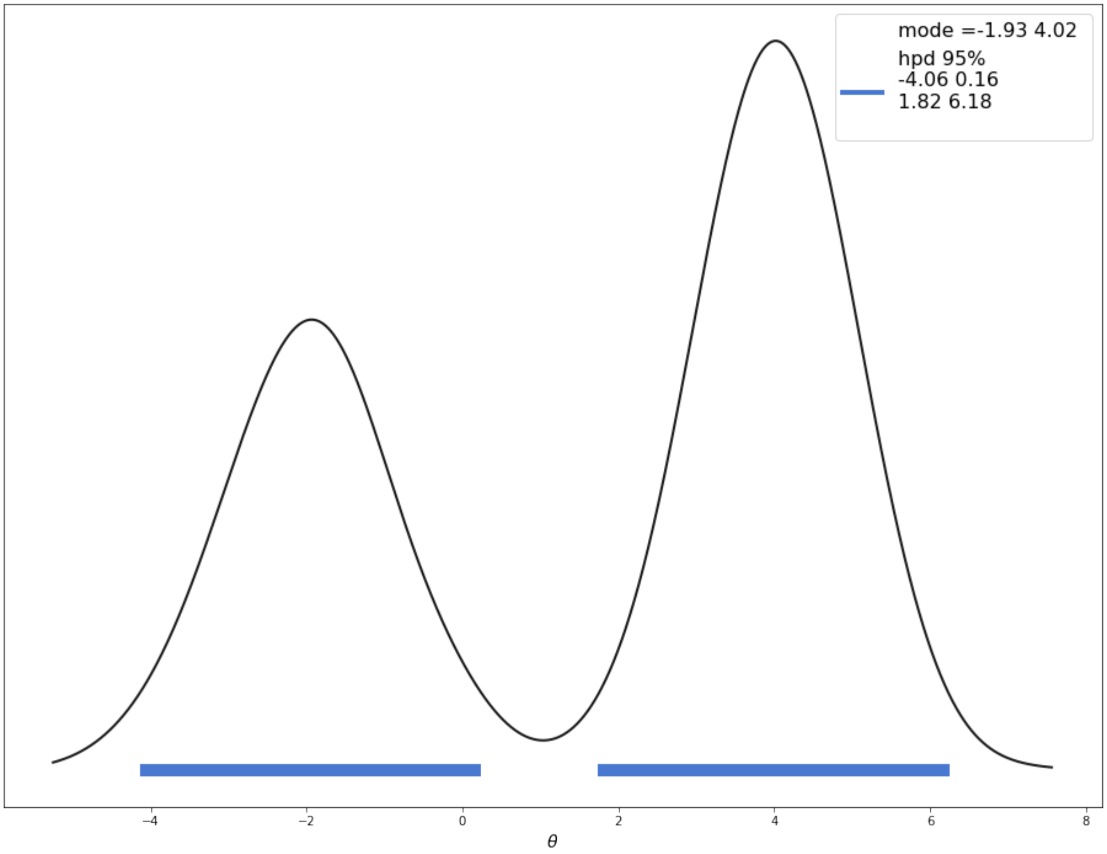
0.6 Naive prediction of HPD

```
In [8]: np.random.seed(1)
        gauss_a = stats.norm.rvs(loc=4, scale=0.9, size=3000)
        gauss_b = stats.norm.rvs(loc=-2, scale=1, size=2000)
        mix_norm = np.concatenate((gauss_a, gauss_b))

        naive_hpd(mix_norm)
        fig = matplotlib.pyplot.gcf()
        fig.set_size_inches(figureSize[0], figureSize[1])
```



0.7 Correct prediction of HPD

```
In [9]: from plot_post import plot_post
        plot_post(mix_norm, roundto=2, alpha=0.05)
        plt.legend(loc=0, fontsize=16)
        plt.xlabel(r"$\theta$", fontsize=14)
        fig = matplotlib.pyplot.gcf()
        fig.set_size_inches(figureSize[0], figureSize[1])
```


mode =-1.93 4.02
hpd 95%
-4.06 0.16
1.82 6.18
−4
−2
0
2
4
6
8
θ