LA03_Ex2_KDE

April 28, 2018

1 Team Members

1.1 RaviKiran Bhat

1.2 Rubanraj Ravichandran

1.3 Mohammad Wasil

1.4 Ramesh Kumar

2 Task1

2.1 Compare the outcomes of different implementations of KDEs.

There are several options available for computing KDE in Python. - SciPy: gaussian_kde. - Statsmodels: KDEUnivariate and KDEMultivariate. - Scikit-learn: KernelDensity.

2.2 1). Generate synthethic data and plot them

Generate synthetic dataset the distribution of which can be presented as a combination of three Gausian distributions with the following parameters: μ_1=1, σ_1=1 and μ_2=8, σ_2=2 and μ_2=14, σ_2=1.5. Generate 1000 samples from the distribution. Plot the pdf of this distribution and the generated samples. 3) Use the generated samples to perform - (i) KDE with Scipy, - (ii) Univariate KDE with Statsmodels, - (iii) Multivariate KDE with Statsmodels as well as - (iv) KDE with Scikit-learn. 4) Plot all four distributions on one figure.

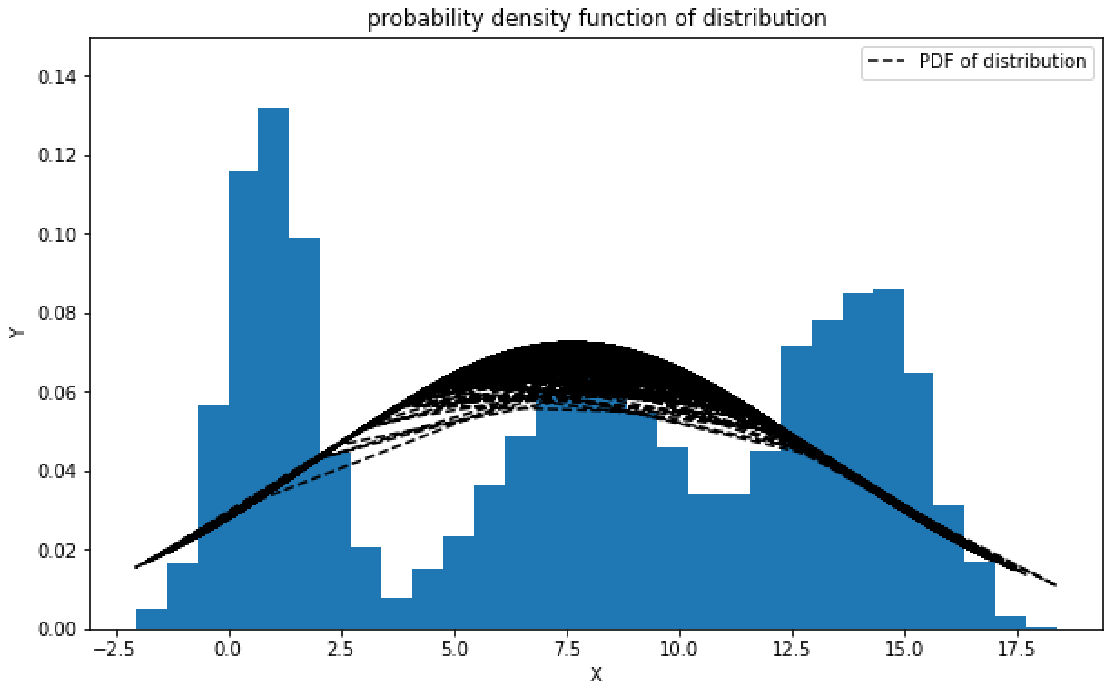
```
In [1]: import numpy as np
        from scipy.stats import norm
        import pandas as pd
        import matplotlib.pyplot as plt
        from scipy import stats
        from statsmodels.nonparametric.kde import KDEUnivariate
        from statsmodels.nonparametric.kernel_density import KDEMultivariate
        from sklearn.neighbors import KernelDensity


        %matplotlib inline
```

```
In [2]: np.random.seed(0)
        gaussian1 = 1 + 1 * np.random.randn(1000)
```

```
gaussian2 = 8 + 2 * np.random.randn(1000)
gaussian3 = 14 + 1.5 * np.random.randn(1000)
gaussian_mixture = np.hstack([gaussian1, gaussian2, gaussian3])

df = pd.DataFrame(gaussian_mixture, columns=['data'])
# parametric fit: assume normal distribution
param_density = stats.norm.pdf(gaussian_mixture, np.mean(gaussian_mixture), np.std(gauss

fig, ax = plt.subplots(figsize=(10, 6))
ax.hist(df.values, bins=30, normed=True)
ax.plot(df, param_density, 'k--', label='PDF of distribution')
ax.set_ylim([0, 0.15])
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_title("probability density function of distribution")
ax.legend(loc='best')
```

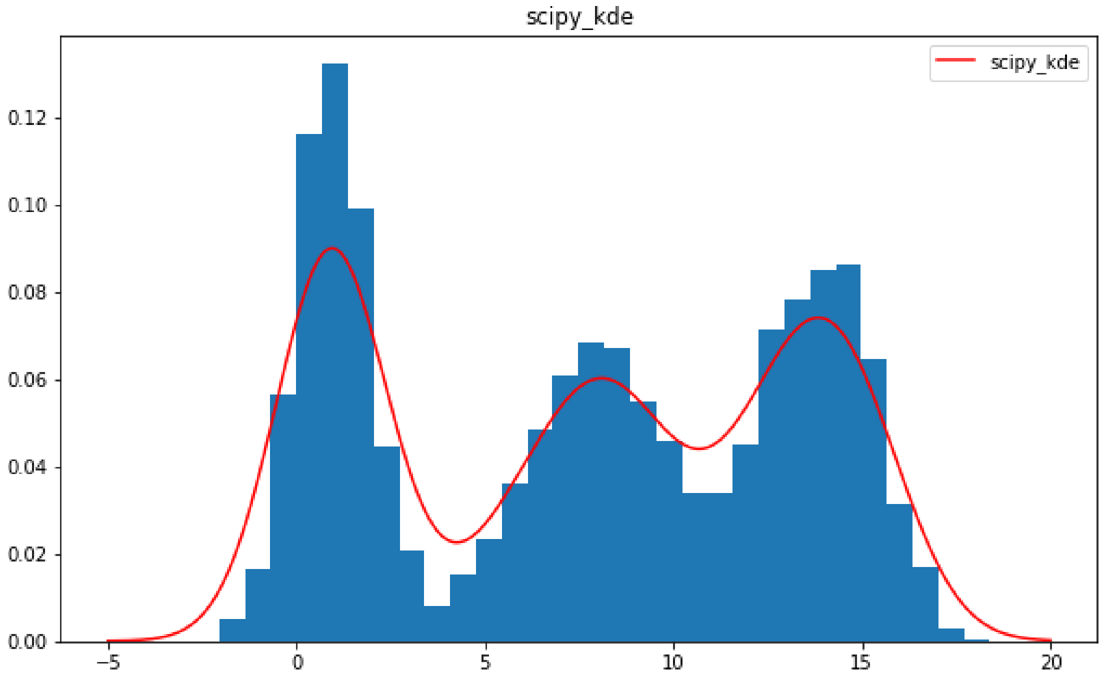
```
/home/ramesh/anaconda2/lib/python2.7/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: T
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```

```
Out[2]: <matplotlib.legend.Legend at 0x7f3921c5f0d0>
```



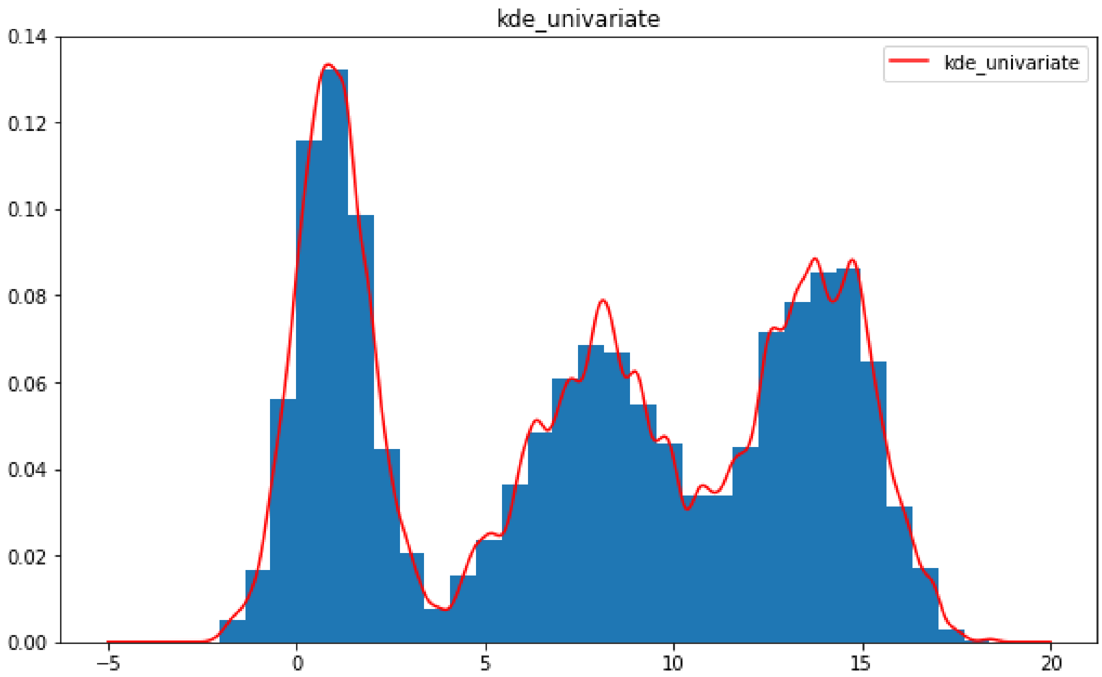
```
In [3]: # non-parametric pdf
        fig, ax = plt.subplots(figsize=(10, 6))
        ax.hist(df.values, bins=30, normed=True)

        scipy_kde = stats.gaussian_kde(df.values.ravel())
        x = np.linspace(-5, 20, 3000)
        scipy_kde = scipy_kde(x)
        ax.plot(x, scipy_kde, 'r-', label='scipy_kde')
        ax.legend(loc='best')
        plt.title('scipy_kde')
```

```
Out[3]: Text(0.5,1,'scipy_kde')
```



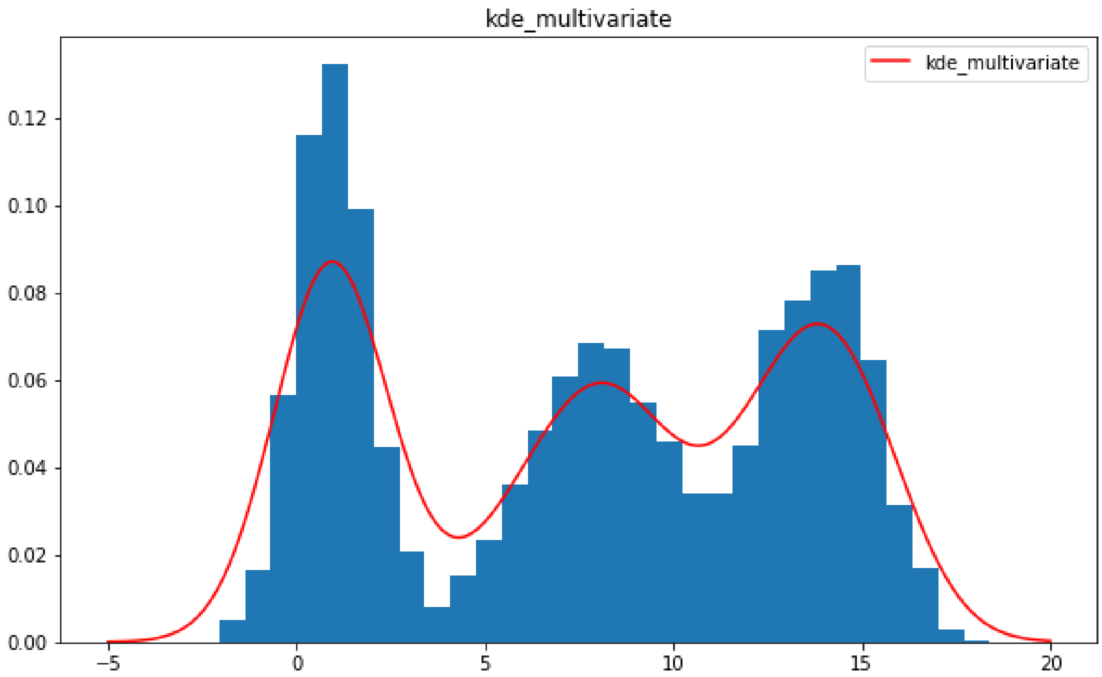
```
In [4]: fig, ax = plt.subplots(figsize=(10, 6))
        ax.hist(df.values, bins=30, normed=True)

        kde_univariate = KDEUnivariate(df.values.ravel())
        kde_univariate.fit(bw=0.2)
        x = np.linspace(-5, 20, 3000)
        pdf = kde_univariate.evaluate(x)
        ax.plot(x, pdf, color='red',label='kde_univariate')
        ax.legend(loc='best')
        plt.title('kde_univariate')
```

```
Out[4]: Text(0.5,1,'kde_univariate')
```

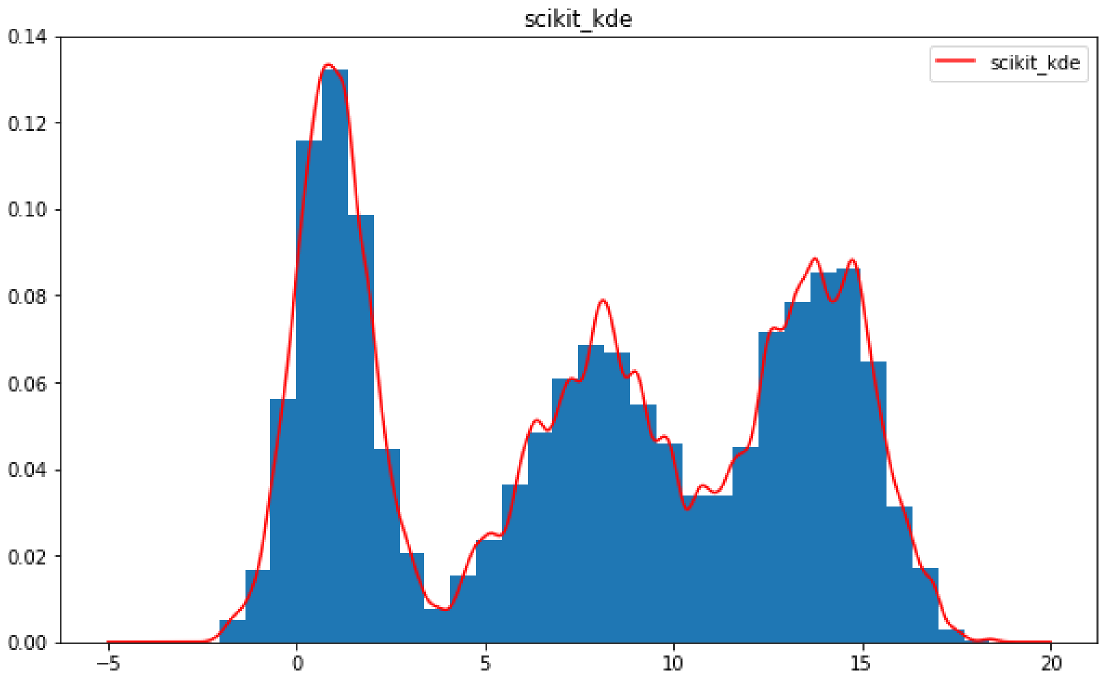


```
In [5]: fig, ax = plt.subplots(figsize=(10, 6))
        ax.hist(df.values, bins=30, normed=True)

        kde = KDEMultivariate(df.values,var_type='c')
        x_grid = np.linspace(-5, 20, 3000)
        kde_multivariate = kde.pdf(x_grid)
        plt.plot(x_grid, kde_multivariate, color='red',label='kde_multivariate')
        ax.legend(loc='best')
        plt.title('kde_multivariate')
```

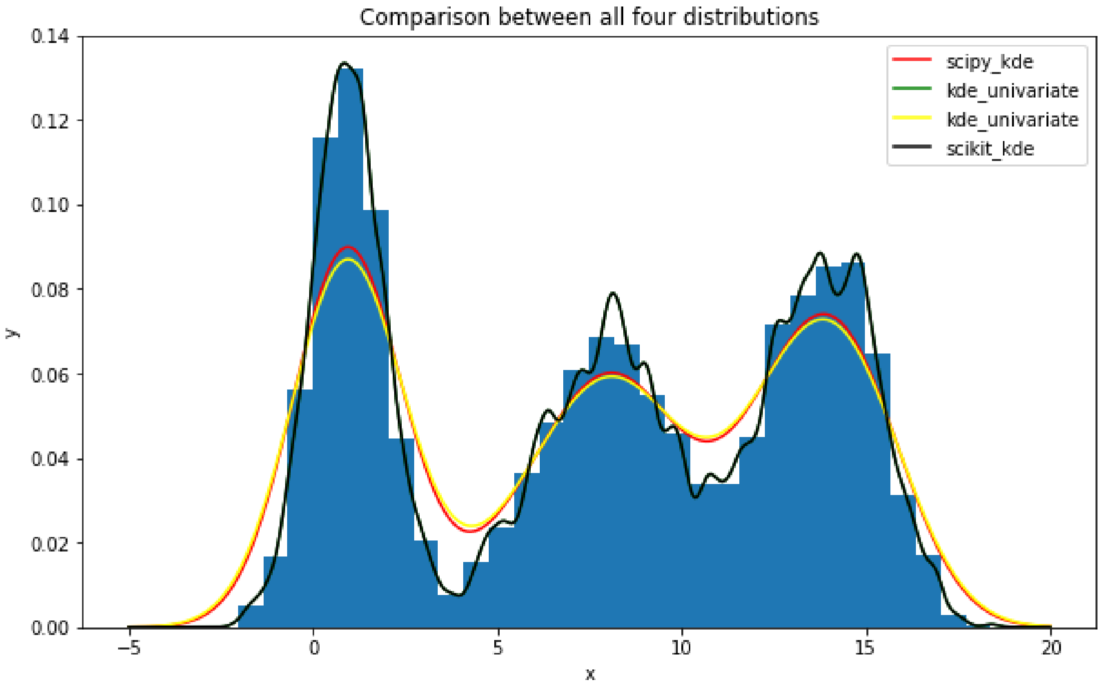
```
Out[5]: Text(0.5,1,'kde_multivariate')
```



```
In [6]: # KDE Scikit learn
        fig, ax = plt.subplots(figsize=(10, 6))
        ax.hist(df.values, bins=30, normed=True)

        scikit_kde = KernelDensity(kernel='gaussian', bandwidth=0.2).fit(df.values.ravel()[:, np
        x = np.linspace(-5, 20, 3000)
        log_dens = scikit_kde.score_samples(x[:, np.newaxis])
        ax.plot(x, np.exp(log_dens),label="scikit_kde",color='red')
        ax.legend(loc='best')
        plt.title('scikit_kde')

Out[6]: Text(0.5,1,'scikit_kde')
```



```
In [10]: # Plot all four distributions on one figure.
         fig, ax = plt.subplots(figsize=(10, 6))
         ax.hist(df.values, bins=30, normed=True)
         ax.plot(x, scipy_kde, 'r-', label='scipy_kde')

         ax.plot(x, pdf, color='green',label='kde_univariate')

         ax.plot(x, kde_multivariate, color='yellow',label='kde_univariate')

         ax.plot(x, np.exp(log_dens),label="scikit_kde",color='black')
         ax.legend(loc='best')

         ax.set_title("Comparison between all four distributions")
         ax.set_xlabel("x")
         ax.set_ylabel("y")

Out[10]: Text(0,0.5,'y')
```


Comparison between all four distributions
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
y
−5
0
5
10
15
20
x
scipy_kde
kde_univariate
kde_univariate
scikit_kde