Material complementario

In [1]:

```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
```

Ejemplo de aprendizaje no supervisado: dimensionalidad de Iris

Como ejemplo de un problema de aprendizaje no supervisado, consideremos la reducción de dimensionalidad de los datos de Iris para visualizarlo más fácilmente. Recuerde que los datos de Iris son cuatridimensionales: hay cuatro características registradas para cada muestra.

La tarea de reducción de dimensionalidad es determinar si existe una representación adecuada con menos dimensiones que conserve las características esenciales de los datos. A menudo, la reducción de dimensionalidad se utiliza como una ayuda para visualizar datos: ¡después de todo, es mucho más fácil mostrar los datos en dos dimensiones que en cuatro dimensiones o más!

Aqui usaremos el análisis de componentes principales (PCA), que es una técnica de reducción de dimensionalidad lineal muy veloz. Solicitaremos al modelo que retorne dos componentes, es decir, una representación bi-dimensional de los datos.

Siguiendo la secuencia de pasos bosquejada antes, tenemos:

In [2]:

```
import seaborn as sns
iris = sns.load_dataset('iris')
X_iris = iris.drop('species', axis=1)
y_iris = iris['species']
```

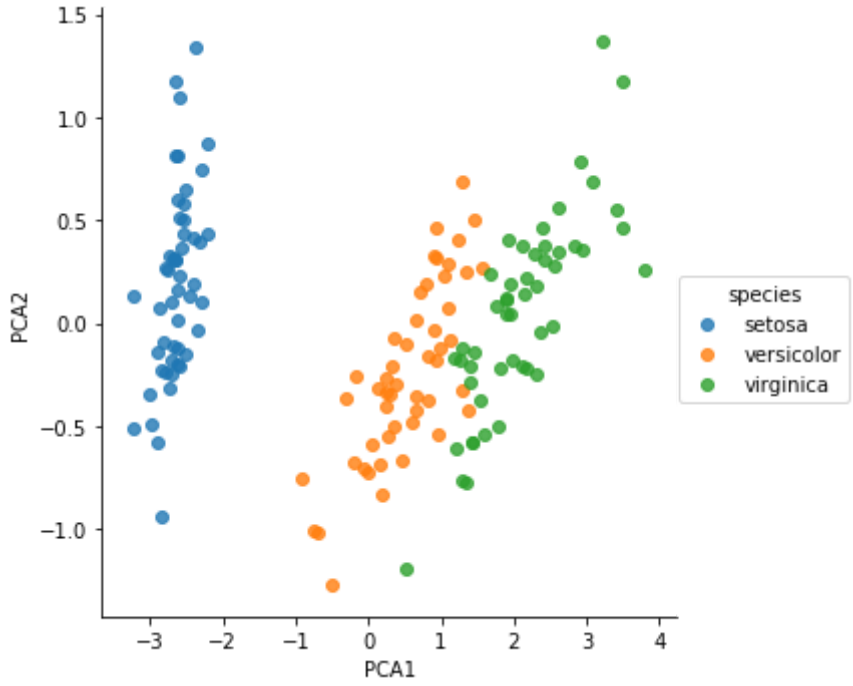
In [3]:

```
from sklearn.decomposition import PCA  # 1. Elegir la clase de modelo
model = PCA(n_components=2)            # 2. Instanciar el modelo con hiperparametros
model.fit(X_iris)                      # 3. Ajustar el modelo a los datos (aprender). N
otar que y no es  specificado!
X_2D = model.transform(X_iris)         # 4. Transformar los datos a dos dimensiones
```

Ahora grafiquemos los resultados. Una forma rápida de hacer esto es insertar los resultados en el `DataFrame` de Iris original, y usar el `lmplot` de Seaborn para mostrar los resultados:

In [4]:

```
iris['PCA1'] = X_2D[:, 0]
iris['PCA2'] = X_2D[:, 1]
sns.lmplot("PCA1", "PCA2", hue='species', data=iris, fit_reg=False);
```



Vemos en la representación bi-dimensional, que las especies están bastante bien separadas, aun cuando el algoritmo PCA no tuvo conocimiento de las etiquetas de las especies!! Eso nos indica que una clasificación relativamente directa probablemente será efectiva sobre el conjunto de datos, como vimos antes.

Aprendizaje no supervisado: agrupamiento (clustering) de Iris

Veamos ahora como hacer agrupamientos (grupos) con los datos de Iris. Un algoritmo de agrupamiento intenta encontrar grupos distintos dentro de los datos sin ninguna referencia a ninguna clase o etiqueta. Aquí usaremos un método de agrupamiento poderosos conocido como modelo de mixturas Gaussianas (en inglés, Gaussian mixture model (GMM)). Un GMM intenta modelar los datos como una colección de gráficas de Gauss

Podemos aprender un GMM de la siguiente manera:

In [5]:

```
from sklearn.mixture import GaussianMixture      # 1. Elegir la clase de modelo
model = GaussianMixture(n_components=3,
            covariance_type='full')  # 2. Instanciar el modelo con hiperparametros
model.fit(X_iris)                    # 3. Ajustar el modelo a los datos (aprender). Not
ar que y no es specificado!
y_gmm = model.predict(X_iris)        # 4. Determinar las etiquetas de los clusters/grup
os
```

Como antes, agregaremos la etiqueta del grupo al `DataFrame` de Iris and usaremos Seaborn para graficar los resultados:

In [6]:

```
iris['cluster'] = y_gmm
sns.lmplot("PCA1", "PCA2", data=iris, hue='species',
           col='cluster', fit_reg=False);
```