PCA

PCA (Principal Component Analysis) o Análisis de Componentes Principales

PCA es una técnica de **reducción de dimensionalidad** utilizada en machine learning y estadística. Su objetivo es transformar un conjunto de datos con muchas variables (dimensiones) en un nuevo conjunto de datos con menos variables, pero que aún conserve la mayor parte de la información (varianza) de los datos originales.

En esencia, PCA identifica las direcciones (llamadas **componentes principales**) en las que los datos varían más y proyecta los datos en esas direcciones. Esto permite simplificar la complejidad de los datos sin perder mucha información.

¿Por qué se usa PCA?

1. **Reducción de dimensionalidad**:
   - Cuando trabajamos con datos de alta dimensionalidad (muchas características), el análisis y la visualización se vuelven difíciles.
   - PCA reduce el número de variables, lo que facilita el procesamiento y la interpretación.
2. **Eliminación de redundancia**:
   - En muchos casos, las variables en un conjunto de datos están correlacionadas (por ejemplo, altura y peso).
   - PCA transforma los datos en un nuevo conjunto de variables no correlacionadas (componentes principales).
3. **Visualización**:
   - PCA permite proyectar datos de alta dimensionalidad en 2 o 3 dimensiones para su visualización.
4. **Mejora del rendimiento**:
   - Al reducir la dimensionalidad, los algoritmos de machine learning pueden funcionar más rápido y con menos riesgo de sobreajuste (overfitting).

¿Cómo funciona PCA?

1. Centrado de los datos:

- PCA comienza centrando los datos restando la media de cada variable. Esto asegura que el primer componente principal pase por el centro de los datos.

2. Cálculo de la matriz de covarianza:

- Se calcula la matriz de covarianza de los datos centrados. Esta matriz describe cómo las variables están relacionadas entre sí.

3. Descomposición en valores propios (eigenvalues) y vectores propios (eigenvectors):

- La matriz de covarianza se descompone en sus valores propios y vectores propios.
- Los **vectores propios** son las direcciones de los componentes principales.
- Los **valores propios** indican la cantidad de varianza explicada por cada componente principal.

4. Selección de componentes principales:

- Los componentes principales se ordenan de mayor a menor según su valor propio (varianza explicada).
- Se seleccionan los primeros `n` componentes principales (por ejemplo, 2 o 3) para reducir la dimensionalidad.

5. Proyección de los datos:

- Los datos originales se proyectan en el nuevo espacio definido por los componentes principales seleccionados.

Ejemplo intuitivo

Imagina que tienes un conjunto de datos en 3D (por ejemplo, puntos en el espacio). PCA identifica las direcciones en las que los datos se extienden más:

1. El **primer componente principal** es la dirección de máxima varianza.
2. El **segundo componente principal** es la dirección de máxima varianza perpendicular al primero.
3. El **tercer componente principal** es la dirección de máxima varianza perpendicular a los dos anteriores.

Si reduces los datos a 2 componentes principales, básicamente estás "aplanando" los datos en un plano 2D que captura la mayor variación posible.

Interpretación de los componentes principales (combinación lineal)

- **Componente 1 (Comp1)**: Es la dirección en la que los datos varían más. Captura la mayor cantidad de información.
- **Componente 2 (Comp2)**: Es la segunda dirección más importante, perpendicular a la primera, y captura la siguiente mayor cantidad de información.
- Y así sucesivamente.

Cada componente principal es una combinación lineal de las variables originales. Por ejemplo:

```
Comp1 = a1 * Variable1 + a2 * Variable2 + ... + an * Variablen
Comp2 = b1 * Variable1 + b2 * Variable2 + ... + bn * Variablen
```

Limitaciones de PCA

1. **Interpretabilidad**:
   - Los componentes principales son combinaciones lineales de las variables originales, lo que puede dificultar su interpretación.
2. **Linealidad**:
   - PCA asume que las relaciones entre las variables son lineales. Si hay relaciones no lineales, PCA puede no ser efectivo.
3. **Pérdida de información**:
   - Al reducir la dimensionalidad, siempre se pierde algo de información. Es importante elegir un número adecuado de componentes principales.

Comprobación de la varianza perdida

Para saber cuánta varianza se ha perdido al aplicar PCA, puedes utilizar la propiedad `explained_variance_ratio_` que proporciona el objeto `PCA` de scikit-learn. Esta propiedad te indica la proporción de la varianza total que es capturada por cada uno de los componentes principales seleccionados.

Pasos para calcular la varianza perdida:

1. **Ajustar el modelo PCA**:
   Después de ajustar el modelo PCA con `fit()`, puedes acceder a `explained_variance_ratio_`.

   ```
   from sklearn.decomposition import PCA

   pca = PCA(n_components=2)  # Reducir a 2 componentes principales
   pca.fit(data_X_norm)       # Ajustar el modelo a los datos normalizados
   ```

2. **Obtener la varianza explicada**:
   La propiedad `explained_variance_ratio_` devuelve un array con la proporción de varianza explicada por cada componente principal.

   ```
   explained_variance = pca.explained_variance_ratio_
   print("Varianza explicada por cada componente:", explained_variance)
   ```

   Por ejemplo, si obtienes:

   ```
   Varianza explicada por cada componente: [0.75, 0.15]
   ```

   Esto significa que:

   - El primer componente principal captura el 75% de la varianza total.
   - El segundo componente principal captura el 15% de la varianza total.

3. **Calcular la varianza total explicada**:
   Suma las proporciones de varianza explicada por los componentes seleccionados.

```
total_variance_explained = sum(explained_variance)
print("Varianza total explicada:", total_variance_explained)
```

En el ejemplo anterior:

```
Varianza total explicada: 0.90 (90%)
```

4. **Calcular la varianza perdida**:
   La varianza perdida es simplemente la diferencia entre el 100% de la varianza total y la varianza explicada por los componentes selelectados.

   ```
   variance_lost = 1 - total_variance_explained
   print("Varianza perdida:", variance_lost)
   ```

   En el ejemplo:

   ```
   Varianza perdida: 0.10 (10%)
   ```

Ejemplo completo

```
from sklearn.decomposition import PCA

# Aplicar PCA
pca = PCA(n_components=2)
pca.fit(data_X_norm)

# Varianza explicada por cada componente
explained_variance = pca.explained_variance_ratio_
print("Varianza explicada por cada componente:", explained_variance)

# Varianza total explicada
total_variance_explained = sum(explained_variance)
print("Varianza total explicada:", total_variance_explained)

# Varianza perdida
variance_lost = 1 - total_variance_explained
print("Varianza perdida:", variance_lost)
```

Interpretación de los resultados

- Si la **varianza total explicada** es cercana a 1 (o 100%), significa que los componentes principales seleccionados capturan casi toda la información de los datos originales.
- Si la **varianza perdida** es alta, significa que se ha descartado mucha información al reducir la dimensionalidad. En este caso, podrías considerar aumentar el número de componentes principales (`n_components`).

Gráfico de varianza acumulada

Para visualizar cómo aumenta la varianza explicada a medida que añades más componentes principales, puedes usar un gráfico de varianza acumulada:

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

# Aplicar PCA sin reducir dimensiones
pca = PCA().fit(data_X_norm)

# Varianza explicada acumulada
cumulative_variance = np.cumsum(pca.explained_variance_ratio_)

# Gráfico
plt.plot(cumulative_variance, marker='o')
plt.xlabel('Número de componentes principales')
plt.ylabel('Varianza explicada acumulada')
plt.title('Varianza explicada por los componentes principales')
plt.grid()
plt.show()
```

Este gráfico te ayudará a decidir cuántos componentes principales seleccionar. Por ejemplo, si el gráfico muestra que el 95% de la varianza se captura con 5 componentes, puedes reducir los datos a esas 5 dimensiones sin perder mucha información.