Actividad 3: Python básico

Máster de Estadística Aplicada

UNIVERSIDAD DE GRANADA

Javier Arnedo. arnedo@ugr.es.
Departamento de Estadística e Investigación Operativa.
Curso 2025-2026.

Ejercicios prácticos.

La actividad consiste en la realización de 3 ejercicios prácticos, para lo cual tendrás que crear un documento Quarto en el que incluyas el código Python necesario para resolverlos y renderizarlo en un documento PDF de no más de 12 páginas. Tendrás que subir a Prado tanto el documento Quarto como el PDF resultante.

Esta actividad es obligatoria como parte de la evaluación ordinaria y supondrá el 40 % de la calificación final. Algunas notas generales:

- La actividad es individual y debe ser original.
- Cada estudiante deberá subir la solución en la tarea habilitada en PRADO respetando los plazos establecidos.
- No se admitirán entregas fuera de plazo.
- La solución consistirá en dos ficheros: un documento dinámico en formato pdf donde se describa la solución, mostrando adecuadamente el código y los resultados (que deben estar integrados en el documento de manera automática usando para ello Quarto); y el fichero fuente (formato qmd) a partir del cual se ha generado dicho pdf.
- El fichero PDF resultante no podrá sobrepasar las 12 páginas de longitud.
- Criterios de evaluación: (i) originalidad; (ii) completitud y resolución correcta; (iii) detalles en la descripción del procedimiento de resolución, en los comentarios y las conclusiones; (iv) presentación del documento.

Ejercicio 1.

Sin utilizar paquetes adicionales, resuelve los siguientes ejercicios y comenta las posibles diferencias de implementación respecto a R:

Ejemplo: Asigna una variable con el valor 5 y muestra el resultado:

```
x = 5
print(x)
```

```
>>> 5
```

En Python utilizamos el signo `=` para asignar valores a las variables, mientras que en R podemos utilizar tanto el signo `<-` como el signo `=`.

En Python utilizamos la función `print()` para mostrar el resultado por pantalla si queremos que funcione en cualquier entorno, porque la autoimpresión solo funciona en un entorno interactivo, mientras que en R simplemente escribimos el nombre de la variable para autoimprimir.

1.1. Escribe un programa que muestre los números del 1 al 10.

1.2. Escribe un programa que almacene en una variable una lista con los números del 1 al 10 y los muestre por pantalla en orden inverso.

1.3. Escribe un programa que muestre los números pares del 1 al 10.

1.4. Escribe un programa que cree un diccionario con las claves "nombre", "edad" y "ciudad" y los valores "Juan", 25 y "Granada", respectivamente, y muestre por pantalla el valor correspondiente a la clave "ciudad".

1.5. Escribe una función que reciba un número entero y devuelva su cuadrado. Utilízala para calcular el cuadrado de algún número y mostrarlo por pantalla.

1.6. Escribe una función que reciba una cadena de texto e imprima un mensaje si es un palíndromo. Utiliza dicha función para comprobar una palabra o frase que sea palíndromo y otra que no lo sea.

Ejercicio 2.

En este ejercicio tendrás que escribir un programa en Python que realice la implementación de la técnica de estadística multivariante de reducción de la dimensionalidad llamada Análisis de Componentes Principales (PCA). Para ello, tendrás que utilizar los paquetes NumPy, Pandas, Matplotlib y SciKit-learn estudiados en la asignatura.

Análisis de Componentes Principales (PCA) es una técnica de reducción de la dimensionalidad que se utiliza para reducir la cantidad de variables en un conjunto de datos. PCA transforma las variables originales en un nuevo conjunto de variables no correlacionadas entre sí llamadas componentes principales. Los componentes principales se ordenan de mayor a menor varianza, de modo que el primer componente principal captura la mayor cantidad de varianza en los datos, el segundo componente principal captura la segunda mayor cantidad de varianza, y así sucesivamente. PCA se utiliza para visualizar datos en un espacio de menor dimensión, para reducir el ruido en los datos, para eliminar la multicolinealidad entre las variables, para identificar patrones en los datos, etc. Puedes aprender más sobre PCA en el libro "An Introduction to Statistical Learning" de James, Witten, Hastie y Tibshirani disponible en el siguiente enlace: https://www.statlearning.com/.

Para implementar PCA en Python, tenemos dos opciones, directamente utilizar la función PCA del paquete SciKit-learn, o hacerlo paso por paso realizando los cálculos necesarios con NumPy. En este ejercicio, tendrás que hacerlo en primer lugar paso por paso y posteriormente con SciKit-learn para comprobar que se obtienen los mismos resultados.

PCA paso a paso:

1. Carga el conjunto de datos en formato CSV que podrás encontar en la actividad de Prado utilizando Pandas.
2. Elimina las filas que contengan valores faltantes.
3. Estandariza los datos para que cada columna tenga media 0 y desviación estándar 1, esto se consigue restando a cada valor la media de su columna y dividiendo por la desviación estándar de su columna. No es necesario estandarizar siempre, pero es recomendable en muchos casos para que todas las variables tengan la misma importancia. Esto es equivalente a no estandarizar los datos, pero utilizar la matriz de correlación en vez de la de covarianza (en el paso 4) al hacer PCA.
4. Calcula la matriz de covarianza de los datos estandarizados.
5. Calcula los vectores y valores propios de la matriz de covarianza. Los vectores propios son los componentes principales y los valores propios son la cantidad de varianza que explican los componentes principales.
6. Selecciona los dos primeros componentes principales que explican la mayor cantidad de varianza en los datos. Para ello tendrás que seleccionar los dos primeros vectores propios correspondientes a los dos mayores valores propios.
7. Proyecta los datos en el nuevo espacio. Para ello tendrás que multiplicar la matriz de datos estandarizados por la matriz de los vectores propios obtenida en los pasos anteriores. Esto se puede hacer utilizando el producto matricial.
8. Visualiza los datos en el nuevo espacio de menor dimensión utilizando un diagrama de dispersión del espacio de los dos primeros componentes principales.

PCA con SciKit-learn:

1. Carga el conjunto de datos en formato CSV que podrás encontar en la actividad de Prado utilizando Pandas.
2. Elimina las filas que contengan valores faltantes.
3. Estandariza los datos para que tengan media 0 y desviación estándar 1 al igual que en el caso paso a paso.
4. Crea el modelo de PCA utilizando la función `PCA()` del submódulo de SciKit-learn `sklearn.decomposition`. Puedes encontrar más información sobre esta función en: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

5. Ajusta el modelo de PCA a los datos estandarizados utilizando el método `fit()` del modelo con los datos estandarizados como parámetro.
6. Transforma los datos originales en el nuevo espacio de menor dimensión utilizando el método `transform()` del modelo con los datos estandarizados como parámetro, con la siguiente orden:

```
X_nuevo = modelo.transform(X_estandarizado)
```

7. Visualiza los datos en el nuevo espacio de menor dimensión utilizando un diagrama de dispersión. X_nuevo será una matriz donde cada columna corresponde al valor obtenido para cada componente principal y cada fila corresponde a cada una de las observaciones originales.

*Nota: El signo de los componentes principales puede variar dependiendo de la implementación, ya que serían equivalentes siempre que su valor absoluto fuese el mismo. Por ello, es posible que los resultados no sean exactamente iguales, si no que tengan alguna componente invertida.

Ejercicio 3.

Busca un paquete de Python que esté relacionado con el análisis de datos y que no haya sido explicado en la asignatura y realiza una descripción de sus funcionalidades incluyendo algunos ejemplos de uso.