

Actividad 1

Python y librerías numéricas

Profesor: Pablo Estévez

Auxiliar: Pablo Cornejo

Ayudantes: Diego Castillo, Andrés González, Sebastián Guzmán, Francisco Soto

Instrucciones Generales

- La entrega de esta actividad se realizará enviando un script de código por pregunta (todas las subpreguntas se realizan dentro del mismo script) cuyo nombre debe corresponder a la pregunta: **P1.py**, **P2.py**, etc. En los casos en los que se pidan gráficos, se debe entregar además un archivo **.png** cuyo nombre debe ser la pregunta seguida de un guión y la subpregunta. Por ejemplo, si se está respondiendo la pregunta 1, subpregunta 2, entonces se debe entregar un archivo **P1-2.png** y en el código se debe mostrar cómo se guardó este archivo (función **savefig** en **matplotlib**).
- **Importante:** Todos los gráficos realizados **deben** tener una etiqueta en el eje X y en el eje Y, además de un título. Las etiquetas y los títulos son libres a ser determinados por cada uno, sin embargo, deben estar relacionados con el contexto del gráfico.
- Por favor comentar los códigos. Si hay subsecciones, se recomienda comentar los lugares donde comienza cada subsección.
- Utilicen nombres significativos para variables, funciones y clases. Esto les ayudará principalmente a ustedes cuando lean sus códigos en un futuro y también al resto de la gente con la que trabajen luego.

Instrucciones Específicas

- Se darán puntos extra si en los scripts se sigue la recomendación entregada como buena práctica en el PPT del auxiliar 1.
- Dado que la pregunta 2 depende de la pregunta 1 pero se entregan en scripts separados, pueden copiar y pegar la parte en la que generan el dataset en la pregunta 1 (sin las partes donde grafican y guardan gráficos).

Preliminar: Instalación de Python y creación de ambientes virtuales

Instale Python en su computador, ya sea utilizando la [página principal](#) o utilizando Anaconda (en este último caso se recomienda la instalación de [miniconda](#)). Luego, cree un ambiente virtual: Si no está utilizando `conda`, puede utilizar el módulo `venv`. Si está utilizando `conda`, entonces puede [crear ambientes virtuales directamente con el comando conda](#). Una vez creado el ambiente virtual, instale las librerías `numpy`, `matplotlib`, `pandas`, `seaborn` y `sklearn`.

Material complementario

- [Artículo interesante sobre ambientes virtuales](#)

P1: Gráfico de distribuciones

1. **Generación de Muestras:** Genere una matriz con 1000 muestras generadas de una distribución [normal bivariada](#) con medias $[-1, -4]$ y matriz de covarianza:

$$\begin{pmatrix} 1 & 0.6 \\ 0.6 & 1 \end{pmatrix}$$

Realice lo mismo en una matriz aparte, cambiando las medias a $[3, -7]$ y utilizando la misma matriz de covarianza anterior. Debería tener ahora dos matrices de 1000×2 cada una. Verifique que la media de cada vector sea correcta utilizando las respectivas funciones de `numpy` aplicadas sobre la dimensión correcta. Por ejemplo, si calcula la media sobre todas las muestras en el primer vector, debería obtener un array con dos valores cercanos a las medias originales.

Hint: En la función `multivariate_normal` de `numpy`, puede definir el número de muestras mediante el argumento `size`. Por ejemplo, `size=10` le entregará 10 muestras de la distribución.

2. **Histograma Bidimensional:** Muestre un [histograma bidimensional](#) de cada muestreo por separado. Verifique visualmente que las medias se encuentran cercanas a las definidas.
3. **Concatenación y Creación de DataFrame:** [Concatene](#) ambas matrices en una sola de tamaño 2000×2 y cree un [DataFrame](#) de `pandas` utilizando la nueva matriz. Llame a la primera y segunda columna: 'Columna 1' y 'Columna 2' respectivamente. Luego, cree una nueva columna llamada 'Clase' tal que los primeros 1000 elementos contengan sólo ceros y los últimos 1000 contengan sólo unos. Para esto pueden ser útiles las funciones de `numpy` [zeros](#) y [ones](#), además de la función para concatenar entregada anteriormente. Muestre el `DataFrame` y verifique que la forma del `DataFrame` sea de 2000×3 (ya que ahora se tiene una nueva columna con las clases).

Este **DataFrame** representa una base de datos con dos características (las dos columnas muestreadas por ustedes en el paso 1), tal que cada dato corresponde a una clase (0 o 1).

4. **Histograma Bidimensional con Seaborn:** Utilizando el **DataFrame** anterior, genere un [histograma bidimensional](#) de ambas columnas con **seaborn** tal que se coloreen de forma distinta los datos asignados a la clase 0 y a la clase 1 (puede ser útil ver la sección de **seaborn** del auxiliar 1). Observe ambas distribuciones.

P2: Clasificador Simple

En esta sección se realizará una clasificación similar a la realizada en el auxiliar 1 en el notebook de librerías numéricas. Consideraremos una forma simple de un clasificador, donde se decidirá si un ejemplo pertenece a una clase si una de sus características está sobre o bajo un umbral. Para esto utilizaremos los datos anteriores, pero como pueden haber visto, tenemos dos columnas. Dado que queremos utilizar un solo umbral, realizaremos primero un preprocesamiento de los datos.

En la pregunta 1.4 pueden haber observado que ambas clases no son fácilmente separables utilizando una sola columna, ya que la línea que las separa es diagonal. Sin embargo, si logramos rotar estos datos, también podemos rotar esta línea imaginaria que permite separar ambas clases, tal que ésta quede de forma perpendicular a una de las características (similar a un umbral).

1. **Aplicación de la Función de Rotación:** Copie la función auxiliar entregada en el archivo **rotacion.py** dentro de su script definido para esta pregunta. Lea la documentación incluida en esta función para aprender a utilizarla. Luego, aplique esta función a los datos de la pregunta 1 probando distintos ángulos hasta que visualmente verifique que la primera columna de los datos es suficiente para separar las clases utilizando un umbral.
2. **Reducción de Dimensionalidad:** Elimine la columna restante del **DataFrame** de **pandas**, dejando un **DataFrame** con una sola columna de características y una columna con las clases. Grafique un histograma unidimensional coloreado según la clase. Luego, transforme este **DataFrame** en un array de **numpy** llamado **X** con la columna del **DataFrame** que representa las características de los datos y un array de **numpy** llamado **y** con las clases. Esta forma de representar los datos es muy común, en especial cuando se utiliza la librería **sklearn**.
3. **Clasificación por Umbral:** En base al histograma de la pregunta anterior, determine visualmente un umbral adecuado para separar ambas clases. Utilizando un clasificador por umbral similar al mostrado en el auxiliar 1, genere un vector con los resultados de su clasificador aplicado a los datos.
4. **Evaluación del Clasificador:** Grafique una matriz de confusión **normalizada** en base a las clases reales y a las clases predichas en el punto anterior. Comente.