FACULTAD DE INGENIERÍA UBA

TALLER DE PROCESAMIENTO DE SEÑALES

Guía de Trabajos Prácticos

Versión 1.0

Primer cuatrimestre 2024

Guía 1

1.1 Sin utilizar loops (for/while) convertir a escala de grises la imagen `pikachu_vs_charmander.jpeg` implementando las siguientes técnicas:

(**a**) $\frac{\text{mín}(R,G,B)+\text{máx}(R,G,B)}{2}$

(**b**) $\frac{R+G+B}{3}$

(**c**) $0.3R + 0.59G + 0.11B$ (utilice el comando @)

: Utilice las funciones `imread` e `imshow` (matplotlib).

1.2 Sin utilizar loops (for/while), utilizando indexación edite la imagen `AFALogo.bmp` para

(**a**) Cortar las letras dentro del logo.

(**b**) Cortar las estrellas y tranpsonerlas.

(**c**) Generar una mascara separando el color de fondo del logotipo.

(**d**) Cambiar el color de fondo de blanco a negro.

(**e**) Espejar la imagen (izquierda a derecha).

(**f**) Dibujar una grilla sobre la imagen cada 4 píxeles. : Utilizar strides.

(**g**) Agregar la 3era estrella.

1.3 Sea la función de densidad de probabilidad

$$p_{XY}(x,y) = \frac{3}{4}\mathbb{1}\left\{0 < y < 1 + x^2, 0 < x < 1\right\}$$

: Se recomienda resolver las integrales con un software.

(**a**) Calcular y graficar en una misma figura el soporte, la esperanza condicional $\mathbb{E}[Y|X = x]$ y la recta de regresión.

(**b**) Calcular el error bayesiano.

1.4 Una conocida cadena de comida rápida desea predecir la ganancia de una sucursal en función de la cantidad de habitantes de la ciudad para decidir si conviene abrirla o no. El archivo `mc.txt` contiene la base de datos a utilizar. La primera columna es la población de la ciudad (de a 10.000 personas) y la segunda es la ganancia (de a $USD 10.000). Los valores negativos indican pérdidas.

(**a**) Implemente su propio código, utilizando matrices, para realizar una regresión lineal que minimice el error cuadrático medio. ¿Cuanto vale dicho error?

(**b**) Visualizar los datos con `scatter` (matplotlib) y superponer la recta de regresión estimada sobre ellos.

(**c**) Diseñe una grilla de puntos que le permita graficar la función costo en un gráfico 3d utilizando `plot_surface` de matplotlib.

(**d**) Predecir la ganancia de una ciudad de 35.000 habitantes.

1.5 Se desea analizar los vientos que ocurren en un parque eólico. El archivo `molinos.csv` contiene datos de potencias acumuladas por un parque eólico para los diferentes vientos. La columna `Velocity` contiene el módulo de la velocidad del viento en ese instante y la comulna `Direction` el ángulo de la velocidad medido en sentido horario ubicando el cero en vientos que provienen del norte. Finalmente las columnas `P` contiene las potencias acumuladas por cada molino.

(**a**) Las potencias negativas son errores de medición. Reemplazar todos estos valores con el valor medio de los valores restantes.
: El comando `SimpleImputer` (sklearn) puede ser util.

(**b**) Expresar la velocidad en coordenadas cartesianas.

(**c**) Entrenar un regresor lineal que estime la velocidad del viento (dos dimensiones cartesianas) en función de las potencias.
: El comando `MultiOutputRegressor` (sklearn) puede ser de gran utilidad.

1.6 Una inmobiliaria desea automatizar la tarea de tazar terrenos. El archivo `inmobiliaria.csv` contiene la base de datos de casas en California.

(**a**) Explorar los datos usando `read_csv` (pandas). Indicar cantidad de muestras, nombre y tipo de dato de cada *feature*.

(**b**) Indicar las frecuencias de las variables categóricas.

(**c**) Para analizar las variables numéricas utilice el comando `pairplot` (seaborn). Explique que representan los gráficos.

(**d**) Utilice el comando `SimpleImputer` (sklearn) para completar los valores faltantes con los más frecuentes.

(**e**) Utilice el comando `get_dummies` (pandas) para codifique las variables categóricas como *one-hot*.

(**f**) Utilice el comando `train_test_split` (sklearn) para definir dos conjuntos con las proporciones 75 % y 25 %. Grafique los histogramas de ambos conjuntos (superpuestos) de la mediana del valor de las propiedades.

(**g**) Utilice el comando `StandardScaler` (sklearn) para normalizar cada variable numérica. Utilice el conjunto de entrenamiento para fijar la normalización y apliquela a ambos conjuntos.

(**h**) Realizar una regresión lineal para predecir la mediana del valor de la propiedad en función del resto de las variables. Indicar el ECM de entrenamiento y testeo.

1.7 Hallar una solución matricial al problema de regresión lineal sin sesgo y con regularización L2. ¿A que se aproxima la solución si el el algoritmo está muy regularizado?

1.8 Se desea estimar la cantidad de agua que fluye por una presa a partir de la variación del nivel de agua. El archivo `represa.csv` contiene los datos a utilizar, definiendo los conjuntos de entrenamiento, validación y testeo.

(**a**) Visualice el dataset de entrenamiento a partir de un gráfico `scatter`.

(**b**) Realice una regresión lineal utilizando `LinearRegression` de sklearn. Grafique la recta de regresión estimada sobre la el `scatter`.

(**c**) Realice una regresión polinómica de orden 8 sin regularización. Grafique la función de regresión estimada sobre el `scatter`.

(**d**) Utilizando `sklearn.linear_model.Ridge`, repetir el inciso anterior regularizando con $\lambda = 1$ y $\lambda = 100$.

(**e**) Graficar el error cuadrático medio en función del hiperparámetro de regularización λ para el conjunto de entrenamiento y validación. ¿Que valor minimiza el error de validación?

(**f**) Calcular el error cuadrático medio de testeo para el hiperparámetro elegido en el inciso anterior.

Guía 2

2.1 Sea $Y \sim \text{Ber}(3/4)$, $X|Y = 0 \sim \mathcal{N}(0, 4)$ y $X|Y = 1 \sim \mathcal{N}(0, 1)$. Hallar $P(y = 1|x)$ y graficarlo sobre la densidad de X. Además computar el error bayesiano, el error de un *clasificador al azar* y el error del *clasificador dummy*.

: Se recomienda resolver las integrales y graficar con un software.

2.2 Encontrar el clasificador óptimo pero permitiendo decisiones aleatorias (no solamente determinísticas).

: Suponga que $\hat{Y}|X = x \sim Q(\cdot|x)$ tal que la verdadera Y e $\hat{Y}$ son independientes cuando $X = x$ (porque la única dependencia entre ambas pasa por X). Encontrar la $Q(\hat{y}|x)$ que minimiza la probabilidad de error $\mathbf{P}(Y \neq \hat{Y})$.

2.3 Sean p y q dos distribuciones Bernoulli de parámetros $\frac{1}{2}$ y $\frac{1}{4}$ respectivamente. Calcular $\text{KL}(p\|q)$ y $\text{KL}(q\|p)$.

2.4 Hallar la distribución de máxima entropía de:

(**a**) (Entropía discreta) Una variable aleatoria discreta de k átomos.

(**b**) (Entropía diferencial) Una variable aleatoria continua con varianza σ^2.

: Analizar $\text{KL}(p\|q)$, donde q es una distribución uniforme discreta de k átomos y una normal de varianza σ^2 en cada caso.

2.5 Sea $p = \sigma(z)$ la función sigmoide.

(**a**) Calcular la función inversa $\sigma^{-1}(p)$ con $p \in (0, 1)$.

(**b**) Calcular la derivada $\sigma'(z)$. Encontrar sus valores mínimo y su máximo, y los puntos donde los alcanza.

(**c**) Escribir la derivada en función de p.

2.6 Un profesor desea estimar si un alumno va a aprobar o no la materia en base a la nota de dos parcialitos. El archivo `parcialitos.txt` contiene una base de datos con las notas de cada estudiante en los parcialitos y si, efectivamente, aprobó o no la materia (1 es aprobar).

(**a**) Hallar una expresión analítica para la función costo y su correspondiente gradiente.

(**b**) Realizar una regresión logística utilizando gradiente descendente y graficar la función costo en función de las iteraciones del entrenamiento.

(**c**) Graficar la frontera de decisión sobre un `scatter`. Prediga si un estudiante con notas 45 y 85 aprobaría la materia.

(**d**) Realizar una regresión logística utilizando `LogisticRegression` (sklearn) y graficar la frontera de decisión sobre un `scatter`.

(**e**) Graficar la curva ROC del clasificador e indicar el punto correspondiente a la decisión tomada anteriormente y el EER.

2.7 El gerente de producción de una fábrica de circuitos integrados desea predecir si un determinado integrado pasará el control de calidad. El archivo `microchips.txt` posee datos de la evaluación de dos pruebas diagnóstico de diferentes integrados, y una tercer columna que indica si pasaron el mencionado control (1 es pasar la inspección).

(**a**) Construir un mapa polinómico hasta orden 6 inclusive. ¿Como puede relacionar la cantidad de parámetros con el grado del polinomio y la cantidad de *features*?

(**b**) Realizar una regresión logística utilizando `LogisticRegression` (sklearn) y graficar la frontera de decisión sobre un `scatter` sin regularización.

(**c**) Realizar una regresión logística y graficar la frontera de decisión sobre un `scatter` con regularización L2 y $\lambda = 1000$.

(**d**) Realizar una regresión logística y graficar la frontera de decisión sobre un `scatter` con regularización L2 y $\lambda = 1$.

: Funciones como `meshgrid` (numpy) y `contour` (matplotlib) pueden ser útiles para graficar las fronteras.

2.8 La base de datos MNIST posee imágenes de los dígitos manuscritos (del 0 al 9). Se desea entrenar un clasificador que a partir de una imagen prediga que dígito aparece en ella.

(**a**) Cargar la base de datos utilizando `tensorflow.keras.datasets.mnist.load_data`. Utilizando `imshow` (matplotlib) represente 10 muestras del conjunto de testeo elegidas al azar.

(**b**) Realizar una regresión logística e indicar el *accuracy* de entrenamiento y testeo.

(**c**) Utilizando `ConfusionMatrixDisplay` (sklearn) represente la matriz de confusión normalizada (testeo) para mostrar la probabilidad de cada predicción para cada clase con 3 decimales.

2.9 Se denomina formante a las frecuencias donde se dan los picos de intensidad en el espectro de un sonido. El archivo `formantes.txt` contiene ejemplos de los 3 primeros formantes del sonido de las vocales /a/, /o/ y /u/. Utilizando solamente los dos primeros formantes:

(**a**) Graficar las muestras en un `scatter`.

(**b**) Superponer a la gráfica anterior las medias y las covarianzas de cada gaussiana (una curva de nivel) del modelo de LDA.

(**c**) Implementar un algoritmo de LDA para clasificar los formantes.

(**d**) Graficar la predicción de las muestras y la frontera de decisión.

(**e**) Generar 50 muestras sintéticas y graficarlas junto a las fronteras.

: Funciones como `random.choice` y `random.multivariate_normal` (numpy) pueden ser útiles.

2.10 [ver **Ejercicio 2.7**] La fábrica de circuitos integrados desea predecir si un determinado integrado pasará el control de calidad a partir del archivo `microchips.txt`.

(**a**) Graficar la frontera de decisión de un algoritmo 1NN sobre el `scatter` de la base de datos. ¿Que puede decir del error de entrenamiento?

(**b**) Repetir para un 7NN. Relacionar el valor de K con los conceptos de *overfitting* y regularización.

(**c**) Graficar $\hat{P}(1|x)$ para un algoritmo 1NN y 7NN entrenados solamente con la primera de las pruebas diagnóstico.

: La función `argsort` (numpy) puede ser útil.

2.11 El archivo `ejs_svm.pkl` contiene un par de bases de datos. Utilizando la base de datos *1er Dataset*:

(**a**) Implementar una clasificación SVM utilizando `solve_qp` (qpsolvers). Graficar la frontera de decisión y las rectas de vectores soportes sobre un `scatter`.

(**b**) Repetir el inciso anterior relajando los márgenes (utilizando $C = 1$).

2.12 El archivo `ejs_svm.pkl` contiene un par de bases de datos. Utilizando la base de datos *2do Dataset*, implementar una clasificación SVM con Kernel gaussiano ($\gamma = 50$) utilizando `svm.SVC` (sklearn) con $C = 1$. Graficar la frontera de decisión sobre un `scatter`.

2.13 La cromatografía de ultra alta performance acoplada a espectrometría de masas de alta resolución permite el diagnóstico del cáncer de próstata. El archivo `prostata.csv` posee datos de la abundancia de concentración de diferentes compuestos químicos y el resultado del diagnóstico: sano, cáncer, benigno y post-cirugía. Se desea predecir el diagnóstico en función del resto de los indicadores.

(**a**) Definir el conjunto de entrenamiento utilizando las muestras con étiquetas válidas. Con las muestras no etiquetadas armar un segundo conjunto de datos.

(**b**) Utilizando `cost_complexity_pruning_path` (sklearn) y utilizando la entropía como impureza, calcular todos los α relevantes para la poda de un árbol de decisión.

(**c**) Utilizando `GridSearchCV` (sklearn) optimizar el valor de α para un 5-fold, utilizando como métrica la F_1 macro. Graficar los valores de F_1 cross-validada en función de α.

(**d**) Utilizando `plot_tree` (sklearn) graficar el árbol podado.

(**e**) Encontrar los 5 *features* más relevantes según la *Gini importance*.

(**f**) Clasificar las muestras del conjunto que no posee etiquetas. Comparar las proporciones de las etiquetas de entrenamiento, las predicciones de entrenamiento y las predicciones del conjunto sin etiquetar.

2.14 [ver **Ejercicio 2.8**] La base de datos FASHION-MNIST posee la mismas características que la MNIST pero para clasificar 10 tipos de ropa. Se desea entrenar un clasificador que a partir de una imagen prediga que dígito aparece en ella.

(**a**) Cargar la base de datos utilizando `tensorflow.keras.datasets.fashion_mnist.load_data`. Utilizando `imshow` (matplotlib) represente 10 muestras del conjunto de testeo elegidas al azar.

(**b**) Utilizando `RandomForestClassifier` (sklearn), entrenar un bosque aleatorio de 100 árboles con impureza *Gini*. Indicar el *accuracy* de entrenamiento y testeo.

(**c**) Utilizando `ConfusionMatrixDisplay` (sklearn) represente la matriz de confusión normalizada (testeo) para mostrar la probabilidad de cada predicción para cada clase con 3 decimales.

(**d**) Graficar en una imagen los 100 píxeles más relevantes según la *Gini importance*.

Guía 3

A. Aprendizaje No Supervisado

3.1 Sea (X, Y) un vector aleatorio con densidad de probabilidad conjunta

$$p_{XY}(x, y) = \frac{e^{-\left(2x+\frac{y}{4x+2}\right)}}{2x+1} \cdot \mathbb{1}\{x > 0, y > 0\}.$$

Encontrar un mecanismo simple que permita generar una de las variables aleatorias en función de la otra y un ruido independiente. Sugiera que variable posiblemente sea la causa y cual el efecto.

: Notar que si $T \sim \mathcal{E}(\lambda)$, entonces $kT \sim \mathcal{E}(\lambda/k)$ con $k > 0$.

3.2 [ver **Ejercicio 2.13**] Utilizando todos los indicadores de la abundancia de concentración de los diferentes compuestos químicos de la base de datos `prostata.csv`, entrenar un algoritmo de PCA.

(**a**) Utilizando `linalg.eig` (numpy), encontrar los autovectores y autovalores. Graficar el porcentaje de energía en función del número de componentes principales.

(**b**) Graficar el error cuadrático medio en función del número de componentes principales. : Sea cuidadoso y no repita cuentas en los loops.

(**c**) Graficar un `scatter` de las dos primeras componentes principales, indicando en diferentes colores las clases originales. Tomar los NaN como una clase distinta.

3.3 [ver **Ejercicio 2.14**] Utilizando la base de datos FASHION-MNIST, se desea entrenar un algoritmo de PCA.

(**a**) Utilizando `decomposition.PCA` (sklearn), calcular y graficar el porcentaje de energía en función del número de componentes principales.

(**b**) Graficar el error cuadrático medio de testeo en función del número de componentes principales.

: Hay que ser cuidadoso con la relación de compromiso entre tiempo de cómputo y memoria RAM. Un buen *tradeoff* puede ser computar la reconstrucción cada 10 componentes principales (1, 11, 21, 31, etc).

(**c**) Graficar imágenes reconstruidas utilizando 1, 81 y 781 componentes principales.

(**d**) Se desea evaluar el desempeño del algoritmo de PCA como detector de anomalías. Para ello, construir una base de datos combinando el conjunto de datos de testeo con el conjunto de datos de testeo de la base de datos MNIST (dígitos).

(**e**) Diseñar un detector de anomalías comparando el error cuadrático contra un umbral. Graficar la curva ROC y marcar el *equal error rate* para 1, 80 y 784 componentes principales. Interpretar resultados.

3.4 [ver **Ejercicio 2.9**] Utilizando los dos primeros formantes de la base de datos `formantes.txt`:

(**a**) Implementar K-means para 3 clusters. Utilizar, como condición de parada, tanto cantidad de iteraciones como convergencia.

(**b**) Graficar un `scatter` de la clasificación final de los datos de entrenamiento, resaltando los centroides.

(**c**) Graficar las fronteras de decisión, superpuestos a un `scatter` con las verdaderas etiquetas.

3.5 Se desea comprimir la imagen `pikachu_vs_charmander.jpeg` a 16 colores, utilizando `cluster.KMeans` (sklearn).

(**a**) Tomando cada pixel como muestras diferentes, implementar un K-means de 16 clusters.

(**b**) Utilizar los centroides como diccionario, para convertir cada pixel en un centroide (utilizando el algoritmo previamente entrenado). Utilizar `imshow` (matplotlib) para graficar la imagen ya codificada.

(**c**) Calcular la cantidad de bits necesarios para guardar la imagen antes y después de comprimirla (teniendo en cuenta el etiquetado y los centroides).

3.6 Los habitantes de *Smallville* pueden ser considerados *trabajador registrado*, *trabajador informal* o *desempleado* con probabilidades $\frac{\theta}{2}$, $\frac{1-\theta}{2}$ y $\frac{1}{2}$ respectivamente, donde $0 \leqslant \theta \leqslant 1$. El municipio posee 10.000 habitantes y cuenta con 4.000 trabajadores registrados.

(**a**) Estimar θ por máxima verosimilitud.

(**b**) Deducir matemáticamente una recursión, vía algoritmo EM, que permita estimar θ.

(**c**) Se denomina *puntos fijos* a los valores de θ que no varían al iterar un paso del algoritmo. Encontrar los puntos fijos del problema de recursión definido por EM.

(**d**) ¿A que punto converge el problema si $\theta_0 = 0.99$?

3.7 Se desea utilizar el algoritmo EM para aproximar la distribución de una variable aleatoria a una mezcla de gaussianas.

(**a**) Generar 100 muestras de una mezcla de gaussianas con pesos $0.1, 0.4, 0.2, 0.3$, medias $-4, 0, 4, 5$ y varianzas $1, 1.96, 1.44, 1$ respectivamente.

(**b**) Implementar el algoritmo EM para entrenar una mezcla de 6 gaussianas a partir de los datos generados.

(**c**) Repetir el inciso anterior utilizando `GaussianMixture` (sklearn).

(**d**) Graficar las dos densidades de probabilidad aprendidas y la verdadera en un mismo gráfico.

3.8 La base de datos `fetch_olivetti_faces` (sklearn) contiene 400 imágenes de rostros.

(**a**) Elegir 6 imágenes al azar y graficarlas.

(**b**) Encontrar las relaciones matemáticas que definen el algoritmo EM. Puede deducirlas (recomendado) o buscarlas en la bibliografía. En caso de elegir la segunda, explicar detalladamente como se implementaría.

(**c**) Utilizando *FactorAnalysis* (sklearn) reducir la dimensión a un espacio latente normal de dimensión 2. Inicializar la matriz Ψ con la diagonal de la matriz de covarianza y utilizar la implementación *lapack*.

(**d**) Se desea explorar el manifold del algoritmo entrenado previamente. Defina una grilla regular de 10×10 entre $[-2, 2]$. Cada punto de esa grilla (100 en total) debe ser reconstruido en una imagen y mostrar los 100 rostros reconstruidos en una grilla representativa del espacio latente.

B. Extracción de Features en aplicaciones específicas

3.9 ⌨ La base de datos 20 NewsGroups posee textos sobre 20 tópicos diferentes.

(**a**) Utilizando `fetch_20newsgroups` (sklearn) cargar los conjuntos de entrenamiento y testeo omitiendo *headers, footers y quotes* de las categorías `alt.atheism`, `talk.religion.misc`, `comp.graphics`, `sci.space`.

(**b**) Utilizando `TfidfVectorizer` (sklearn) pre-procesar los datos.

: Se recomienda convertir todo a minúsculas, utilizar como *Stop Words* las estándar del idioma inglés, descartar el 80 % de las palabras más frecuentes y utilizar la transformación tf-idf.

(**c**) Utilizando `LogisticRegression`(sklearn), entrenar un clasificador logístico y evaluar el *accuracy* con el conjunto de testeo.

3.10 ⌨ Se desea estudiar relaciones entre píses y sus capitales. El archivo `country-list.csv` contiene la información correspondiente.

(**a**) Descargar las representaciones pre-entrenadas *FastText* en idioma inglés.

: Con el siguiente código shell puede descargar el modelo:

```
wget https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.en.300.bin.gz
gzip -d cc.en.300.bin.gz
```

(**b**) Utilizando `FastText` (pyfasttext), cargar las representaciones pre-entrenadas.

(**c**) Utilizando `most_similar` (pyfasttext), armar una función que conteste a la pregunta *A es a B como C es a ...* dando 3 opciones posibles.

: Ej. *France* es a *Paris* como *Brazil* es a ... (Brasilia).

(**d**) Combinar los países y las capitales de `country-list.csv` para entrenar un algoritmo PCA que reduzca la dimensión a 2. Graficar las representaciones reducidas de *Italy, Rome, France, Paris, Germany* y *Berlin* uniendo los países con sus capitales.

3.11 ⌨ Se desea incursionar en la temática de clasificación de género musical.

(**a**) Utilizando `load` (librosa) cargar los primeros 120 segundos del archivo `mi_perro_dinamita.mp3`. Graficar la señal temporal en función del tiempo (en segundos). Repetir con los primeros 120 segundos del archivo `exclusive.mp3`.

(**b**) Reproducir los audio utilizando `Audio` (IPython).

(**c**) Extraer los 12 primeros MFCC de cada señal utlizando `mfcc` (librosa). Concatenarlos y normalizar en media y varianza utilizando `StandardScaler` (sklearn).

(**d**) Generar etiquetas que indiquen de que archivo provenía cada frame e implementar una clasificación SVM con Kernel gaussiano utilizando `svm.SVC` (sklearn) con $C = 1$.

(**e**) Construir una base de datos de testeo con los siguientes 30 segundos de cada audio. Evaluar el *accuracy* del clasificador en la nueva base de datos.

(**f**) Repetir el ejercicio agregando los coeficientes Δ y $\Delta\Delta$ utilizando `delta` (librosa).

Guía 4

4.1 Lucas dispara a un blanco y el disparo impacta en un punto aleatorio $(X, 0)$ con X (en decímetros) una variable aleatoria con distribución normal de media nula y varianza $1/\tau$, donde τ representa la precisión de Lucas. A priori la precisión τ tiene una distribución chi-cuadrado de 8 grados de libertad. Lucas tiro 10 veces al blanco y observó $\sum_{i=1}^{10} x_i^2 = 17$. En virtud a la información muestral,

(**a**) Hallar la distribución *a posteriori*.

(**b**) Hallar la distribución predictiva. : Mirar con cariño la distribución t-student.

(**c**) Estimar la probabilidad de que su próximo tiro se aleje del centro en menos de 2.1 decímetros. : Se recomienda calcular el cuantil con un software.

4.2 [ver **Ejercicio 2.9**] Utilizando los dos primeros formantes de la base de datos `formantes.txt`:

(**a**) Utilice el comando `train_test_split` (sklearn) para definir dos conjuntos con las proporciones 80 % y 20 %.

(**b**) Graficar las muestras en un `scatter`, resaltando las medias y las covarianzas de cada gaussiana (una curva de nivel) del modelo GNB.

(**c**) Repetir el inciso anterior para LDA y QDA.

(**d**) Clasificar el conjunto de testeo con cada método. Indicar la probabilidad de error.

4.3 [ver **Ejercicio 3.9**] Utilizando `fetch_20newsgroups` (sklearn):

(**a**) Cargar los conjuntos de entrenamiento y testeo omitiendo *headers, footers y quotes*.

(**b**) Utilizando `CountVectorizer` (sklearn) pre-procesar los datos.
: Se recomienda convertir todo a minúsculas, utilizar como *Stop Words* las están-dar del idioma inglés, descartar el 95 % de las palabras más frecuentes y descartar las palabras vistas 1 sola vez.

(**c**) Implementar un código *Multinomial Naive Bayes* y evaluar el *accuracy* con el conjunto de testeo.

4.4 [ver **Ejercicio 3.7**] Utilizando los datos del ejercicio **Ejercicio 3.7**, implementar un Variational Bayes Gaussiano. Suponer *a priori* $m = 0$, $\delta = \nu = \beta = 0.05$ y $\alpha = (1, 1, 1, 1, 1, 1)$, y utilizar el algoritmo EM para inicializar las probabilidades.

(**a**) Con la distribución *a posteriori* generar 3 muestras de parámetros y graficar la densidad de $X|\mu, \lambda, \pi$ para cada uno de esos conjuntos de parámetros. Comparar con la densidad verdadera y con la estimada por el algoritmo EM.

(**b**) Graficar la densidad *predictiva*. Comparar con la densidad verdadera y con la estimada por el algoritmo EM.

4.5 El método de Euler es un método numérico para la resolución de ecuaciones diferenciales ordinarias. Sea la ecuación $y'(t) = f(t, y(t))$ con $y(t_0) = y_0$, el método consiste en discretizar $t_k = t_0 + hk$ (con h pequeño) para luego calcular $y_{k+1} = y_k + hf(t_k, y_k)$ y aproximar $y(t_{k+1}) \approx y_{k+1}$.

(**a**) Utilizar el método de Euler para aproximar $y(4)$ para el problema $y' = y$, $y(0) = 1$ con diferentes valores de h. Graficar el error en función de h. : Deberá resolver la ecuación diferencial (analíticamente) para calcular el error.

(**b**) Se desea utilizar el método de Euler para aproximar $\int_a^b g(t)dt$. ¿Que función $f(\cdot, \cdot)$ utilizaría? Si $m = \frac{a}{h}$ y $n = \frac{b}{h}$ son naturales, hallar una expresión analítica para $y_n - y_m$. Interpretar gráficamente.

(**c**) Estimar por el método de Euler $\int_{-1}^{2} \frac{e^{-t^2/2}}{\sqrt{2\pi}} dt$ a partir de 12 puntos equiespaciados. Comparar el error con una estimación Monte-Carlo de 12 puntos:

- Si las variables aleatorias se sortean de forma uniforme.
- Si las variables aleatorias se sortean de forma normal.

4.6 [ver **Ejercicio 4.4**] Utilizando los datos del ejercicio **Ejercicio 3.7**, implementar un *Markov Chain Monte-Carlo* utilizando pymc. Suponer *a priori* $m = 0$, $\delta = \nu = \beta = 0.05$ y $\alpha = (1, 1, 1, 1, 1, 1)$, y simular 3 cadenas de 1000 muestras cada una. Graficar la densidad *predictiva* para cada cadena. Comparar con la densidad verdadera, con la estimada por el algoritmo EM con la estimada utilizando Bayes Variacional.

4.7 Se desean modelar los resultados de un partido de futbol. Para esto, se propone asumir que la cantidad de goles que marca un equipo es una función de su potencia ofensiva combinada con la fuerza defensiva del rival. También se debe tener en cuenta la ventaja de la localía. El modelo se verá así:

- potencial local = ventaja localía + ataque local + defensa visitante
- potencial visitante = ataque visitante + defensa local

Notar que mientras tener alto ataque es positivo, tener alta defensa es negativo. Cada equipo tendrá su propio *rating* de ataque y defensa, pero estás dependerán de una distribución común (si bien cada equipo de fútbol es claramente diferente entre sí, todos juegan en la misma liga). Vamos a suponer que el ataque y defensa (inicial) de cada equipo son variables aleatorias normales de media nula. *A priori*, la precisión (inversa de la varianza) de ataque y la de defensa (comunes a todos los equipos) serán variables aleatorias gamma de parámetros 0.1 y 0.1.

Dado que existen innumerables combinaciones de parámetros que potencialmente podrían darnos los mismos resultados, se propone restar las calificaciones medias (es decir, el promedio de todos los equipos) tanto de ataque como defensa para garantizar la identificabilidad. Esto obliga al modelo a brindarnos un resultado reproducible y mantiene los parámetros en un rango realista. De esta manera se construye el ataque y defensa final de cada equipo.

Finalmente, una vez definidos los potenciales tanto del local como del visitante, es necesario modelar la cantidad de goles de cada equipo. La cantidad de goles de cada equipo se supondrán Poisson de media $e^{\text{potencial}}$ (para que sean positivos).

(**a**) La base de datos `liga_arg.csv` contiene todos los resultados del futbol argentino desde el profesionalismo. Armar un conjunto de datos de entrenamiento con todas las competiciones iniciadas en el 2020 o posterior.

(**b**) Construir la arquitectura descripta. Utilizando `model_to_graphviz` (pymc) mostrar el grafo del modelo.

(**c**) Entrenar el modelo bayesiano con los datos del inciso (**a**). Utilizando `plot_posterior` (pymc) graficar la distribución a posteriori de la ventaja de localía.

(**d**) Reportar los 5 equipos con mejores ataques esperados. Repetir con los de mejores defensas.

(**e**) Diseñar una función que, dado un equipo local y uno visitante, estime la cantidad de goles esperados por cada equipo en un partido. Estimar el valor esperado del resultado global en un cruce ida y vuelta entre River y Boca (dos partidos, uno local River y el otro local Boca). : Si $X|A = a \sim \text{Poi}(e^a)$, entonces $\mathbb{E}[X] = \mathbb{E}[e^A]$.

(**f**) Diseñar una función que, dado un equipo local y uno visitante, estime la probabilidad que gane el local, empate y gane el visitante. Aplicarla a la última fecha jugada del campeonato local (solo tener en cuenta los partidos donde ambos equipos aparezcan en el conjunto de datos de entrenamiento) y reportar el porcentaje de acierto. : Si $X|A = a \sim \text{Poi}(e^a)$ y $Y|B = b \sim \text{Poi}(e^b)$, entonces $\mathbb{P}(X > Y) = \sum_{i=0}^{\infty} \sum_{j=0}^{i-1} \mathbb{E}[\mathbb{P}(X = i, Y = j|A = a, B = b)]$. Para su implementación numérica asuma que la máxima cantidad de goles por un equipo es 10.

BIBLIOGRAFÍA SUGERIDA

1. "Pattern Recognition and Machine Learning", C. Bishop.
2. "The Elements of Statistical Learning: Data Mining, Inference, and Prediction", J. Hastie, T. Tibshirani, R. Friedman.
3. "Machine Learning: A Probabilistic Perspective", K. Murphy.
4. "Introduction to Machine Learning with Python: A Guide for Data Scientists", A. Müller, S. Guido.
5. "Bayesian Methods for Hackers: Probabilistic Programming and Bayesian Inference", C. Davidson-Pilon.
6. "Pattern Classification", R. Duda, P. Hart, D. Stork.
7. "Deep Learning", I. Goodfellow, Y. Bengio, A. Courville.
8. "Elements of Information Theory", T. Cover, J. Thomas.
9. "Elements of Causal Inference: Foundations and Learning Algorithms", J. Peters, D. Janzing, B. Schölkopf.
10. "Foundations of Machine Learning", M. Mohri, A. Rostamizadeh, A. Talwalkar.
11. "Data Analysis: A Bayesian Tutorial", D. Sivia and J.Skilling.
12. "The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation", C. Robert.