

DATA SCIENCE APLICADO A NEGOCIOS

6 CASOS PRACTICOS DEL MUNDO REAL



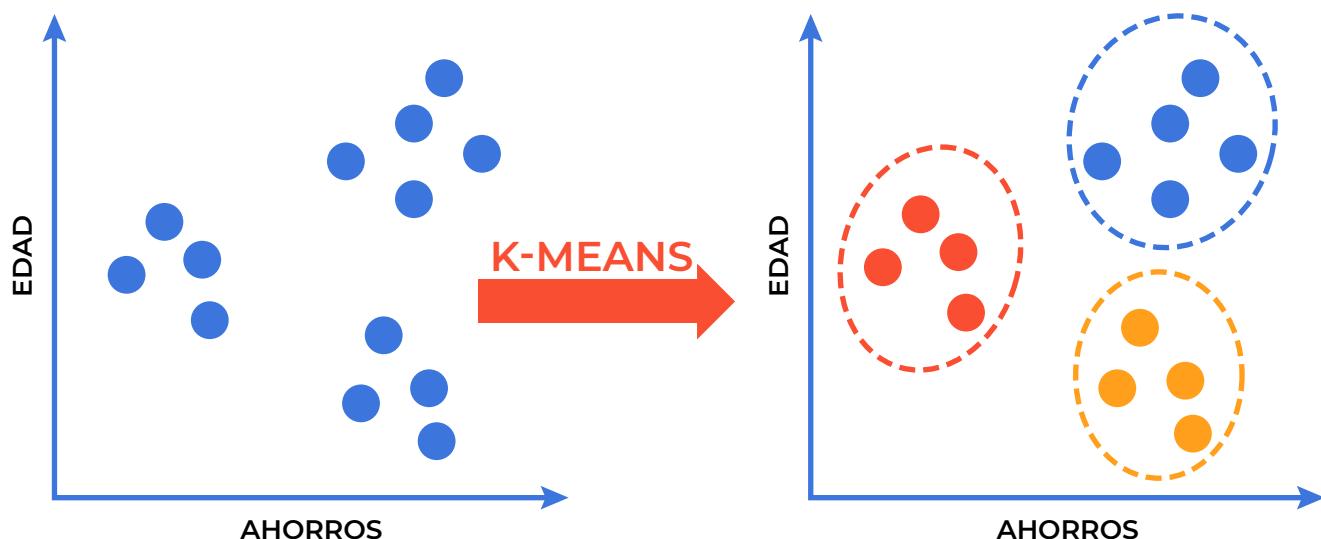
Crea 6 proyectos empresariales prácticos del mundo real y aproveche el poder de la ciencia de datos y el aprendizaje automático para resolver problemas empresariales prácticos del mundo real.

Con Juan Gabriel Gomila
y el equipo de SuperDataScience

1. APRENDIZAJE NO SUPERVISADO (CLUSTERING) K-MEANS

A. CONCEPTO

- K-means es un algoritmo no supervisado de clustering.
- K-means trabaja agrupando puntos de datos (llamados clusters) en un modo no supervisado.
- El algoritmo agrupa observaciones con atributos de valores similares conjuntamente, y para ello mide la distancia Euclídea entre dichos puntos.



B. PASOS DEL ALGORITMO K-MEANS

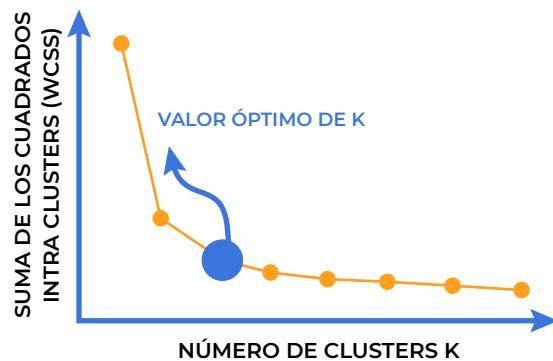
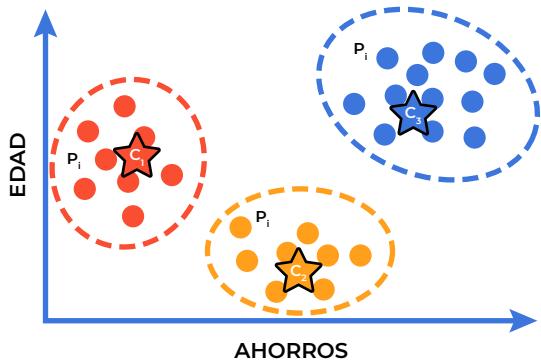
1. Elegir el número óptimo de clusters, K.
2. Seleccionar aleatoriamente K puntos que serán los centros de gravedad (centroídes) de cada cluster.
3. Asignar cada punto del conjunto de datos al centroide del cluster más cercano y crear así un total de K clusters de puntos.
4. Calcular el nuevo centroide de cada uno de los clusters.
5. Reasignar cada punto del conjunto de datos al centroide del cluster más cercano.
6. Ir al paso 4 y repetir.

C. CÓMO ELEGIR EL NÚMERO ÓPTIMO DE CLUSTERS, K?

- Calcular la suma de los cuadrados intra clusters (WCSS) para varios valores de K (número de clusters).
- Representar gráficamente WCSS en función de K y elegir el codo de la curva como el valor óptimo de clusters a utilizar.

Suma de los cuadrados intra clusters (WCSS)

$$\begin{aligned} &= \sum_{P_i \text{ en Cluster 1}} \text{distancia}(P_i, C_1)^2 \\ &+ \sum_{P_i \text{ en Cluster 2}} \text{distancia}(P_i, C_2)^2 + \sum_{P_i \text{ en Cluster 3}} \text{distancia}(P_i, C_3)^2 \end{aligned}$$



D. ¿CÓMO IMPLEMENTAR K-MEANS UTILIZANDO SCIKIT-LEARN?

```
>> k = 4 #especificar el número de clusters
>> kmeans = KMeans(k)
>> kmeans.fit(X_train)
>> labels = kmeans.labels_
```



2. PRONÓSTICO DE SERIES TEMPORALES

A. CONCEPTO

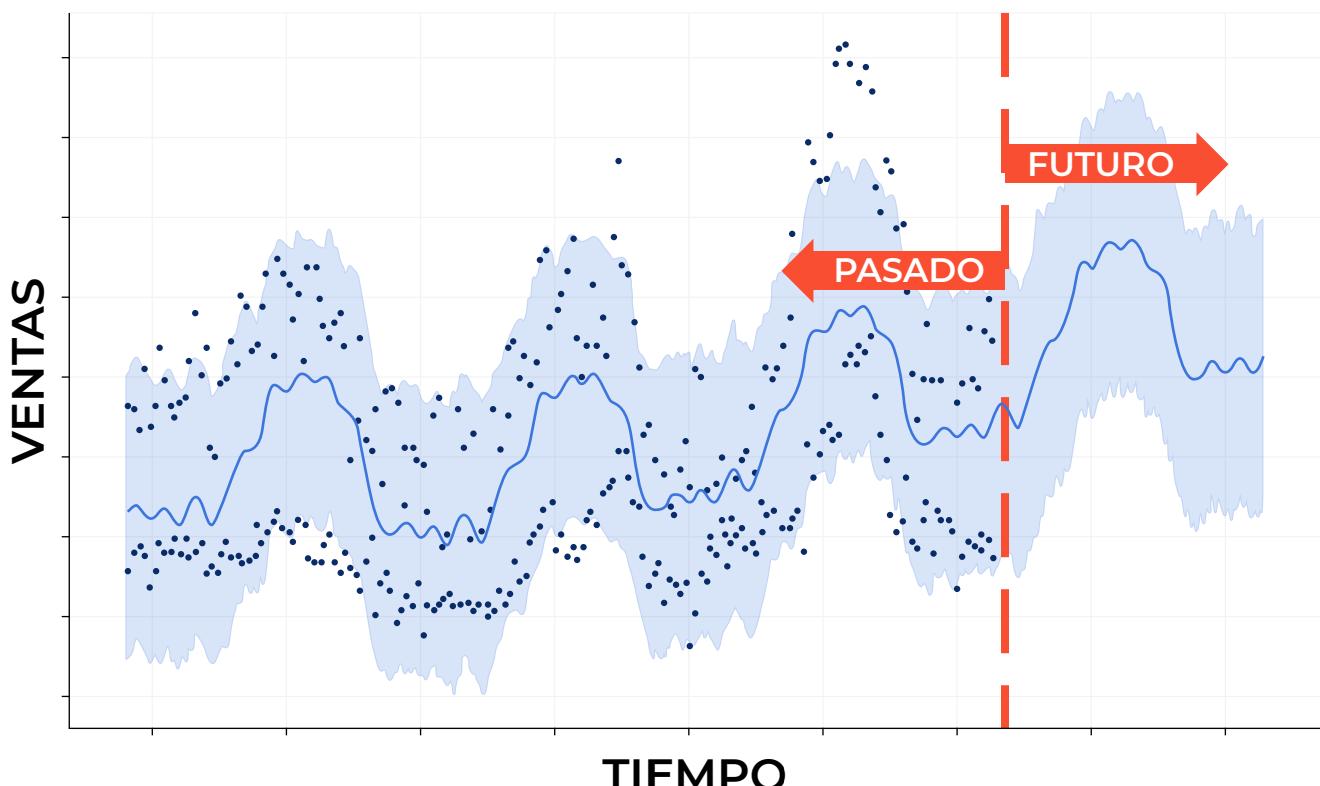
- Los modelos predictivos intentan pronosticar las ventas futuras basándose en datos históricos, al tiempo que tienen en cuenta los efectos de la estacionalidad, la demanda, las vacaciones, las promociones y la competencia.
- Facebook Prophet es un software de código abierto lanzado por el equipo Core Data Science de Facebook. Prophet es un procedimiento para pronosticar datos de series de tiempo basado en un modelo aditivo donde las tendencias no lineales se ajustan a la estacionalidad anual, semanal y diaria, más los efectos de las vacaciones.



- Prophet implementa un modelo de regresión aditiva con cuatro elementos:
 1. Prophet, lineal por partes, detecta automáticamente los puntos de cambio en los datos e identifica cualquier cambio en las tendencias.
 2. Una componente estacional anual modelada utilizando series de Fourier.
 3. Un componente estacional semanal.
 4. Una lista de días festivos que se puede proporcionar manualmente.
- El modelo de regresión aditiva toma la forma:

$$Y = \beta_0 + \sum_{j=1}^p f_j(X_j) + \epsilon$$

Las funciones $f_j(x_j)$ son funciones de suavizado desconocidas que se ajustan desde los datos



B. FACEBOOK PROPHET

```
>> from fbprophet import Prophet  
>> data_df = data_df.rename(columns = {'Date': 'ds', 'Sales': 'y'})  
>> m = Prophet()  
>> m.fit(data_df)  
>> future = m.make_future_dataframe(periods = 720)  
>> forecast = m.predict(future)  
>> figure = m.plot(forecast)
```



3. TAREAS DE REGRESIÓN

A. CONCEPTO

- La regresión se utiliza para predecir el valor de una variable Y basándose en otra variable X.
- X se llama variable independiente e Y se llama variable dependiente.
- La regresión lineal simple es un modelo estadístico que examina la relación lineal entre dos variables únicamente.
- La regresión lineal múltiple examina la relación entre más de dos variables.
- Cada variable independiente tiene su propio coeficiente correspondiente.

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + \dots + b_n * x_n$$

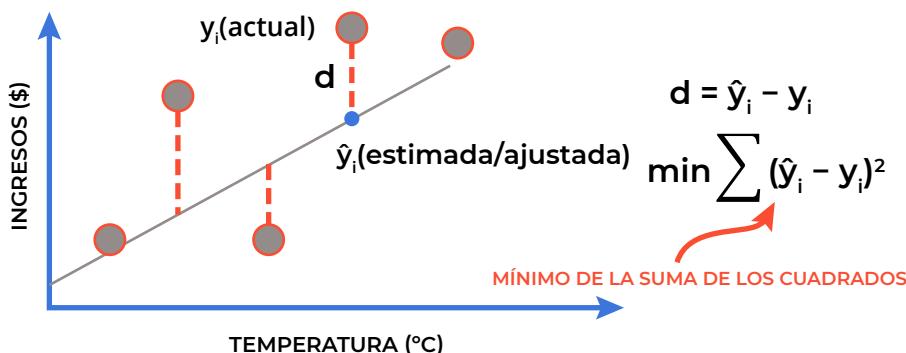
VARIABLE DEPENDIENTE

VARIABLES INDEPENDIENTES



B. LA TÉCNICA DE LOS MÍNIMOS CUADRADOS

- El ajuste por mínimos cuadrados es una forma de encontrar la curva o línea de mejor ajuste para un conjunto de puntos.
- La suma de los cuadrados de las compensaciones (residuales) se usa para estimar la curva o línea de mejor ajuste.
- Se utiliza el método de mínimos cuadrados para obtener los coeficientes m y b .



C. IMPLEMENTAR LA REGRESIÓN LINEAL MÚLTIPLE CON SCIKIT-LEARN

```
>> from sklearn.linear_model import LinearRegression  
>> regressor = LinearRegression(fit_intercept =True)  
>> regressor.fit(X_train,y_train)  
>> print('Linear Model Coefficient (m): ', regressor.coef_)  
>> print('Linear Model Coefficient (b): ', regressor.intercept_)
```

D. EVALUAR EL MODELO (HACER PREDICCIONES USANDO UN MODELO ENTRENADO)

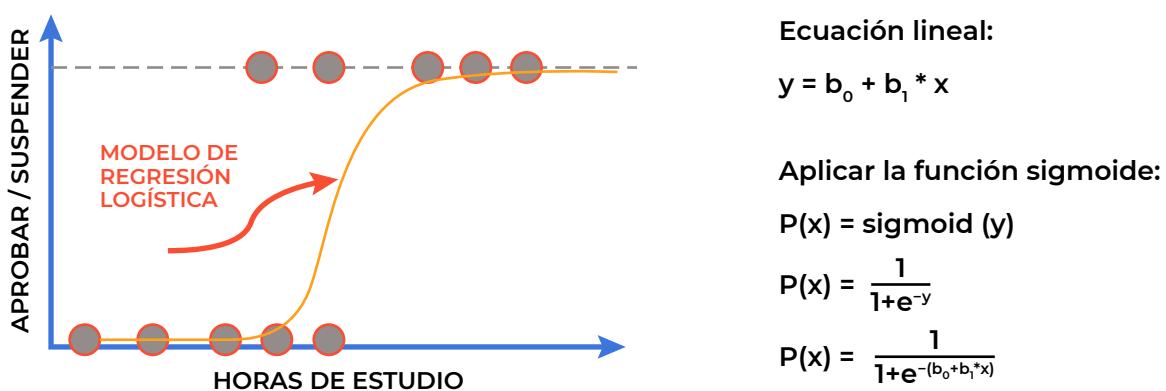
```
>> y_predict = regressor.predict( X_test)  
>> y_predict
```

4. TAREAS DE CLASIFICACIÓN

4.1. CLASIFICACIÓN UTILIZANDO REGRESIÓN LOGÍSTICA

A. EL CONCEPTO DE REGRESIÓN LOGÍSTICA

- El algoritmo de regresión logística funciona implementando en primer lugar una ecuación lineal con predictores independientes para predecir un valor.
- Este valor luego se convierte en una probabilidad que podría oscilar entre los valores 0 y 1.
- La regresión logística se puede utilizar como técnica de clasificación estableciendo un valor de umbral (Ej.: 0,5) para predecir salidas binarias con dos valores posibles etiquetados como "0" o "1".
- Por lo tanto, la salida del modelo logístico puede ser una de dos clases: aprobar /suspender, ganar / perder, sano / enfermo.



B. REGRESIÓN LOGÍSTICA CON SCI-KIT LEARN

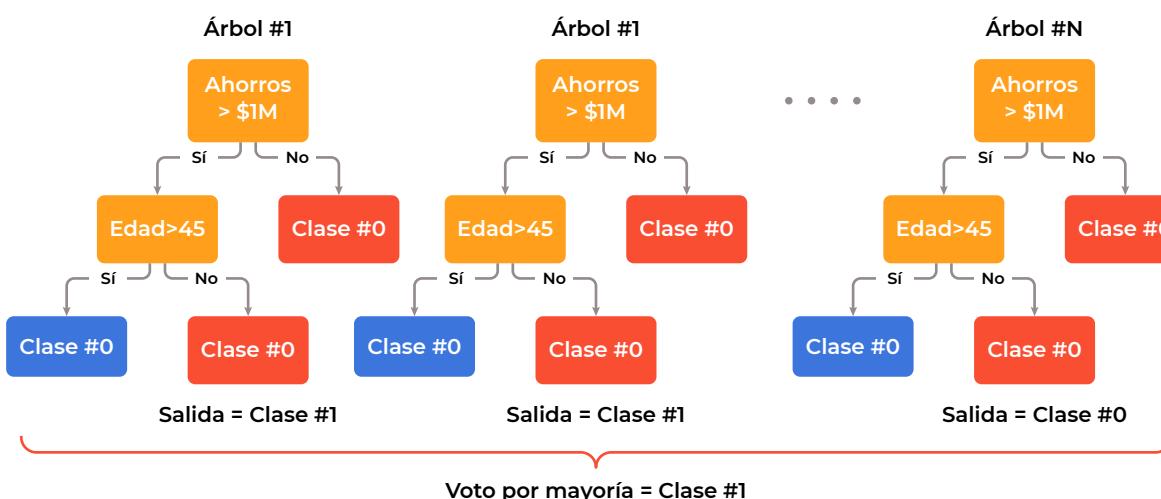
```
>> from sklearn.linear_model import LogisticRegression  
>> Logistic_Regressor = LogisticRegression(random_state = 0)
```

>> Logistic_Regressor.fit(X_train, y_train)

4.2. CLASIFICACIÓN UTILIZANDO CLASIFICACIÓN CON BOSQUES ALEATORIOS

A. EL CONCEPTO DEL CLASIFICADOR DE BOSQUES ALEATORIOS

- El clasificador de bosque aleatorio es un tipo de algoritmo de ensamblado.
- Crea un conjunto de árboles de decisión a partir de un subconjunto seleccionado al azar del conjunto de entrenamiento.
- Luego combina el voto de diferentes árboles de decisión para decidir la clase final del objeto de prueba.
- Supera los problemas con árboles de decisión únicos al reducir el efecto del ruido.
- Supera el problema del sobreajuste tomando el promedio de todas las predicciones, cancelando posibles sesgos.



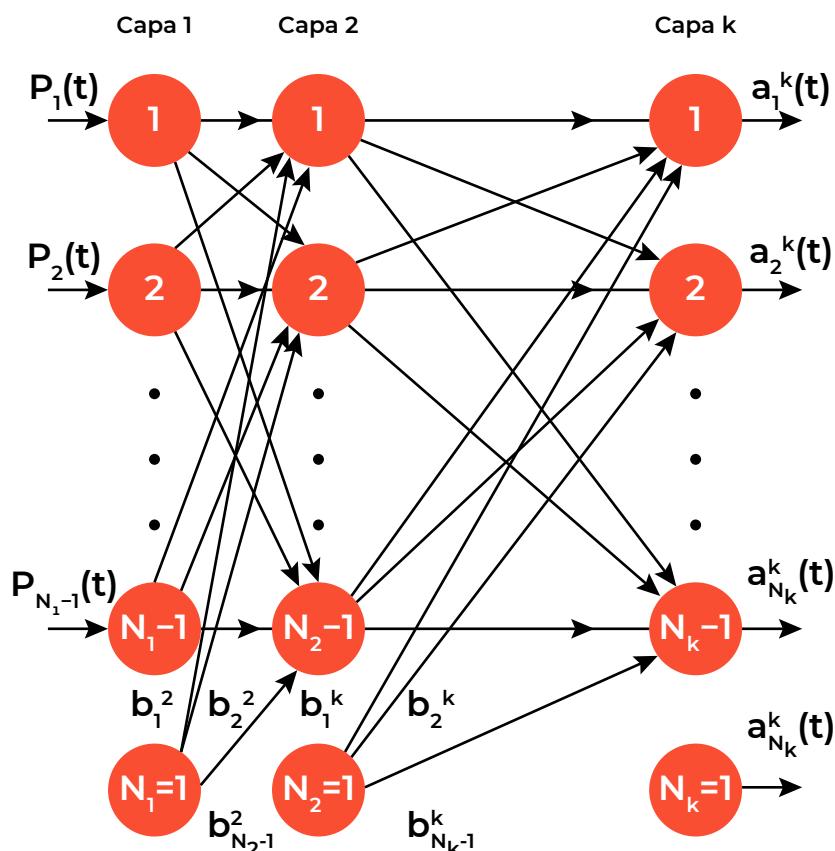
B. CLASIFICADOR DE BOSQUES ALEATORIOS EN SCI-KIT LEARN

```
>> from import sklearn.ensemble RandomForestClassifier  
>> RandomForest = RandomForestClassifier(n_estimators=250)  
>> RandomForest.fit(X_train, y_train)
```

5. DEEP LEARNING Y VISIÓN POR ORDENADOR

A. EL CONCEPTO DE REDES NEURONALES ARTIFICIALES

- Las redes neuronales artificiales son modelos de procesamiento de información inspirados en el cerebro humano. Las RNAs se construyen en capas donde las entradas se propagan comenzando desde la capa de entrada a través de las capas ocultas y finalmente hasta la salida.



El entrenamiento de redes se realiza optimizando la matriz de pesos que se describe a continuación:

$$\begin{bmatrix} W_{11} & W_{12} & & W_{1, N_1} \\ W_{21} & W_{22} & \cdots & W_{2, N_1} \\ \vdots & \ddots & \ddots & \vdots \\ W_{m-1, 1} & W_{m-1, 2} & \cdots & W_{m-1, N_1} \\ H_{m, 1} & W_{m, 2} & \cdots & W_{m, N_1} \end{bmatrix}$$

B. CONSTRUIR UNA RNA UTILIZANDO KERAS

```
>> import tensorflow.keras  
>> from keras.models import Sequential  
>> from keras.layers import Dense  
>> from sklearn.preprocessing import MinMaxScaler  
>> model = Sequential()  
>> model.add(Dense(25, input_dim=5, activation='relu'))  
>> model.add(Dense(25, activation='relu'))  
>> model.add(Dense(1, activation='linear'))  
>> model.summary()
```

C. ENTRENAR UNA RNA UTILIZANDO KERAS

```
>> model.compile(optimizer='adam', loss='mean_squared_error')  
>> epochs_hist = model.fit(X_train, y_train, epochs=20,  
batch_size=25, validation_split=0.2)
```

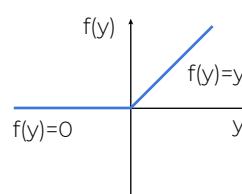
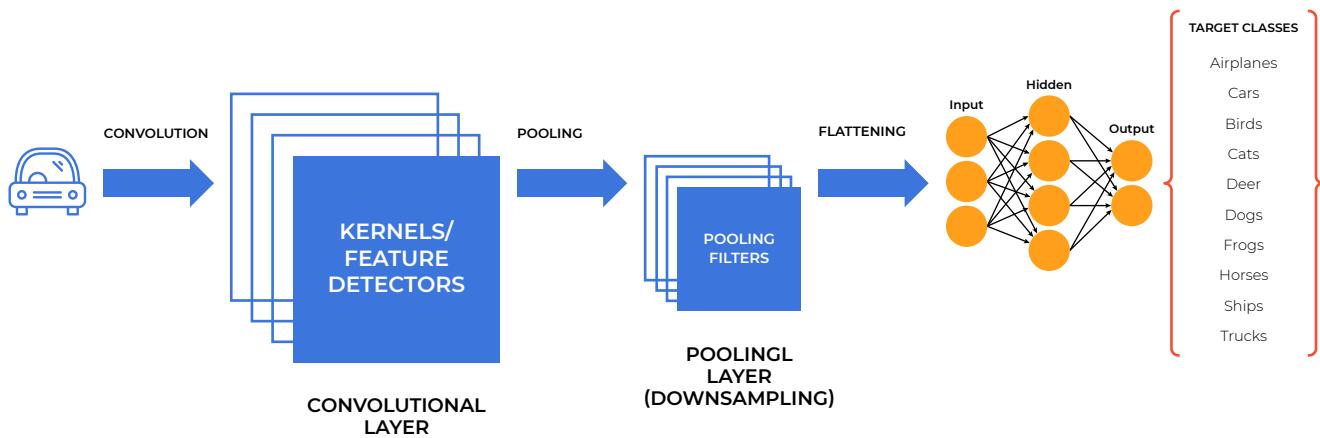


D. EVALUAR UN MODELO DE RNA UTILIZANDO KERAS

```
>> X_Testing = np.array([[input #1, input #2, input #3,.., input  
#n]])  
>> y_predict = model.predict(X_Testing)
```

E. CÓMO CONSTRUIR UNA RED NEURONAL CONVOLUCIONAL (RNC)

- Una RNC es un tipo especial de red neuronal típicamente utilizada en el problema de clasificación de imágenes.
- Las RNC están formadas por (1) capas convolucionales (núcleos y detectores de características), (2) funciones de activación (RELU), (3) capas de agrupación (agrupación máxima o agrupación promedio) y (4) capas completamente conectadas (una red de perceptrón multicapa).



F. CONSTRUIR UNA RNC UTILIZANDO KERAS

```
>> from keras.models import Sequential  
>> from keras.layers import Conv2D, MaxPooling2D,  
    AveragePooling2D, Dense, Flatten, Dropout  
>> from keras.optimizers import Adam  
>> from keras.callbacks import TensorBoard  
  
>> cnn_model = Sequential()  
>> cnn_model.add(Conv2D(filters = 32, kernel_size=(3,3),  
    activation = 'relu', input_shape = (32,32,3)))  
>> cnn_model.add(Conv2D(filters = 32, kernel_size=(3,3),  
    activation = 'relu'))  
>> cnn_model.add(MaxPooling2D(2,2))  
>> cnn_model.add(Dropout(0.3))  
>> cnn_model.add(Flatten())  
>> cnn_model.add(Dense(units = 512, activation = 'relu'))  
>> cnn_model.add(Dense(units = 10, activation = 'softmax'))
```

6. PROCESAMIENTO DE LOS LENGUAJES NATURALES (TOKENIZACIÓN)

A. CONCEPTO

- La tokenización es un procedimiento común en el procesamiento del lenguaje natural. La tokenización funciona dividiendo una oración en un conjunto de palabras. Luego, estas palabras se utilizan para entrenar un modelo de aprendizaje automático para realizar una determinada tarea.

B. TOKENIZACIÓN UTILIZANDO SCI-KIT LEARN

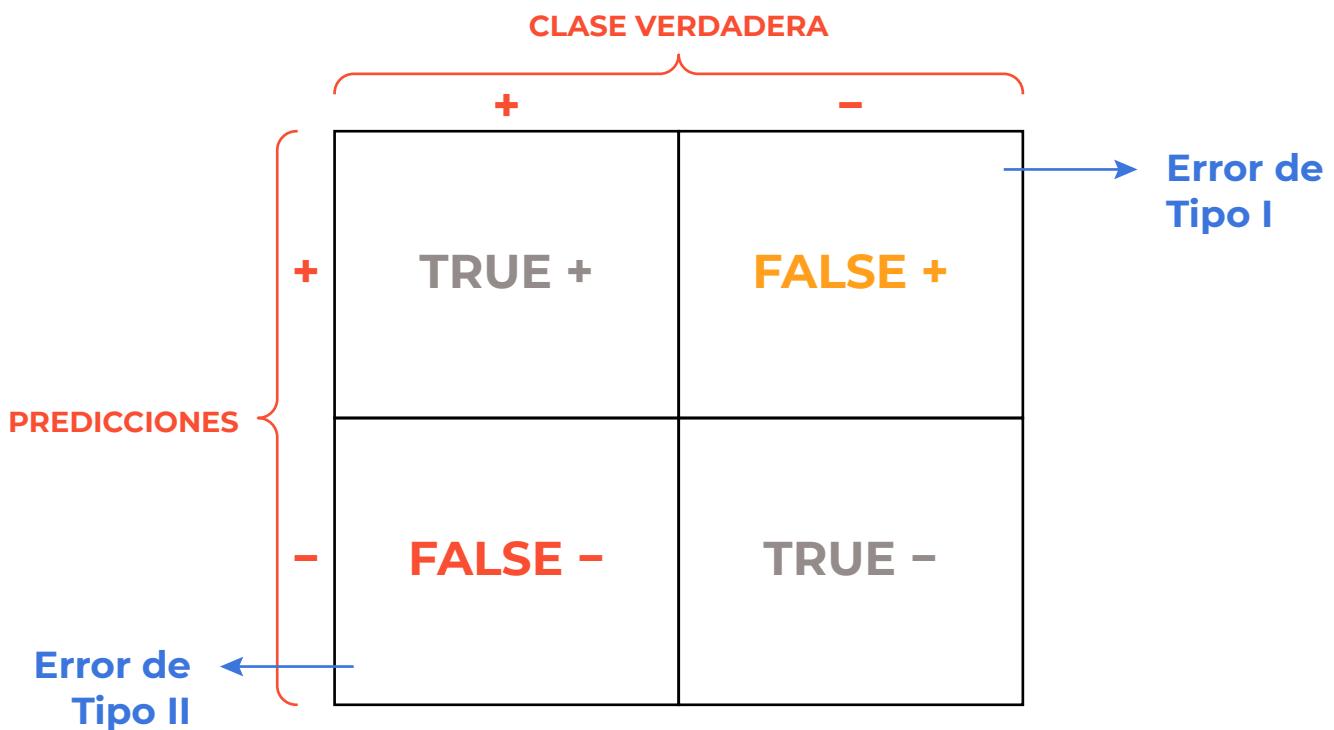
```
>> from sklearn.feature_extraction.text import CountVectorizer  
>> vectorizer = CountVectorizer()  
>> output = vectorizer.fit_transform(data_df])
```

7. EVALUACIÓN DE LOS MODELOS DE CLASIFICACIÓN (MATRIZ DE CONFUSIÓN / INFORME DE CLASIFICACIÓN)

A. EL CONCEPTO DE MATRIZ DE CONFUSIÓN

Una matriz de confusión se utiliza para describir la efectividad de un modelo de clasificación:

- Verdaderos positivos (TP): casos en los que el clasificador predijo VERDADERO (tiene una enfermedad) y la clase correcta fue VERDADERO (el paciente tiene una enfermedad).
- Negativos verdaderos (TN): casos en los que el modelo predijo FALSO (sin enfermedad) y la clase correcta fue FALSO (el paciente no tiene enfermedad).
- Falsos positivos (FP) (error de tipo I): el clasificador predijo VERDADERO, pero la clase correcta fue FALSO (el paciente no tiene enfermedad).
- Falsos negativos (FN) (error de tipo II): el clasificador predijo FALSO (el paciente no tiene la enfermedad), pero en realidad tiene la enfermedad.



- Precisión de la clasificación = $(TP+TN) / (TP + TN + FP + FN)$
- Error de clasificación (ratio de error) = $(FP + FN) / (TP + TN + FP + FN)$
- Precisión = TP/Total TRUE Predictions = TP/ (TP+FP) (Cuando el modelo predice TRUE, qué tan probable es que acierte?)
- Recall = TP/ Actual TRUE = TP/ (TP+FN) (Cuando la clase verdadera es TRUE, qué tan probable es que el modelo acierte?)

B. MATRIZ DE CONFUSIÓN EN SCI-KIT LEARN

```
>> from sklearn.metrics import classification_report,
confusion_matrix

>> y_predict_test = classifier.predict(X_test)

>> cm = confusion_matrix(y_test, y_predict_test)

>> sns.heatmap(cm, annot=True)
```

C. REPORTE DE CLASIFICACIÓN

```
>> from sklearn.metrics import classification_report  
>> print(classification_report(y_test, y_pred))
```

8. MÉTRICAS DE REGRESIÓN EN MODELO DE MACHINE LEARNING

A. ERROR ABSOLUTO PROMEDIO (MAE)

- El error absoluto medio (MAE) se obtiene calculando la diferencia absoluta entre las predicciones del modelo y los valores reales.
- El MAE es una medida de la magnitud promedio del error generado por el modelo de regresión.
- El error absoluto medio (MAE) se calcula de la siguiente manera:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

- El MAE se calcula con los siguientes pasos:
 1. Calcula el residuo (el error de predicción) para cada punto de datos.
 2. Calcula el valor absoluto (para deshacerse del signo).
 3. Calcula el promedio de todos los residuos.
- Si el MAE es cero, esto indica que las predicciones del modelo son perfectas.

B. ERROR CUADRÁTICO MEDIO (MSE)

- El error cuadrático medio (MSE) es muy similar al error absoluto medio (MAE), pero en lugar de utilizar valores absolutos, se calculan los cuadrados de la diferencia entre las predicciones del modelo y el conjunto de datos de entrenamiento (los valores reales).
- Los valores de MSE son generalmente grandes en comparación con el MAE ya que los residuos se están elevando al cuadrado.
- En caso de datos atípicos, MSE se volverá mucho más grande en comparación con MAE.
- En MSE, el error aumenta de forma cuadrática mientras que el error aumenta de manera proporcional en MAE.
- El MSE se calcula de la siguiente manera:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

- El MAE se calcula siguiendo los siguientes pasos:
 1. Calcula el residuo (el error de predicción) para cada punto de datos.
 2. Calcula el valor al cuadrado de cada residuo.
 3. Calcula el promedio de todos los residuos.



C. LA RAÍZ CUADRADA DEL ERROR CUADRÁTICO MEDIO (RMSE)

- La raíz cuadrada del error cuadrático medio (RMSE) representa la desviación estándar de los residuos (es decir, las diferencias entre las predicciones del modelo y los valores verdaderos (datos de entrenamiento)).
- RMSE se puede interpretar fácilmente en comparación con MSE porque las unidades RMSE coinciden con las unidades de salida.
- RMSE proporciona una estimación de la magnitud de la dispersión de los residuos.
- El MSE se calcula de la siguiente manera:

$$\text{MSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

- El RMSE se calcula siguiendo estos pasos:
 1. Calcula el residuo para cada punto de datos.
 2. Calcula el valor al cuadrado de los residuos.
 3. Calcula el promedio de los residuos al cuadrado.
 4. Obtén la raíz cuadrada del resultado anterior.



D. ERROR DE PORCENTAJE ABSOLUTO MEDIO (MAPE)

- Los valores de MAE pueden oscilar entre 0 y infinito, lo que dificulta la interpretación del resultado en comparación con los datos de entrenamiento.
- El error de porcentaje absoluto medio (MAPE) es el equivalente a MAE pero proporciona el error en forma de porcentaje y, por lo tanto, supera las limitaciones de MAE.
- MAPE puede presentar algunas limitaciones si el valor del punto de datos es cero (ya que hay una operación de división involucrada).
- El MAPE se calcula de la siguiente manera:

$$\text{MAPE} = \frac{100\%}{n} \sum_{i=1}^n |(y_i - \hat{y}_i)/y_i|$$

E. ERROR DE PORCENTAJE MEDIO (MPE)

- MPE es similar a MAPE pero sin la operación de valor absoluto.
- MPE es útil para proporcionar una idea de cuántos errores positivos (por exceso) hay en la predicción comparado con los negativos (por defecto).
- El MPE se calcula de la siguiente manera:

$$\text{MPE} = \frac{100\%}{n} \sum_{i=1}^n (y_i - \hat{y}_i)/y_i$$

