Apuntes

Un compendio estructurado de los conceptos, métricas y preguntas clave del curso.

Módulo 1: Fundamentos de Ciencia de Datos y el Proceso de Proyecto

Áreas Clave de la Inteligencia de Negocio (Business Intelligence)

- **Data Warehousing:** Almacenamiento de datos, ya sea en servidores locales (tradicional) o en la nube (moderno), como AWS, Google Cloud, Azure.
- **Data Discovery & Visualization:** Creación de gráficos para extraer información, perfil típico del analista de negocio.
- **Data Quality Management:** Aseguramiento de la calidad y fiabilidad de los datos. Si los datos base son incorrectos, los resultados no serán fiables.
- **Self-Service BI:** Creación de dashboards o cuadros de mando interactivos para que cualquier usuario pueda explorar los datos.
- **Integración de Datos:** Unificación de datos de múltiples fuentes (internas, externas, diferentes formatos como texto, video, transacciones).
- **Data Governance:** Asegurar que el tratamiento de datos cumple con la legislación vigente (ej. RGPD en Europa) y es ético (sin sesgos).

El Proceso de un Proyecto Orientado a Datos (CRISP-DM)

- **Comprensión del Negocio:** Entender la necesidad de la empresa para establecer un objetivo claro. Sin un objetivo, el análisis se dispersa.
- **Comprensión de los Datos:** Recopilar y explorar los datos iniciales para entender qué significan y su calidad.
- **Preparación de los Datos:** Limpieza y transformación de los datos (tratar nulos, duplicados, errores). Es la fase más importante.
- **Modelado:** Aplicar algoritmos estadísticos o de Machine Learning (predicción, clasificación, etc.).

- **Evaluación:** Analizar qué tan bueno es el modelo y si cumple los objetivos del negocio.
- **Implementación (Despliegue):** Poner el modelo o los resultados a disposición de los usuarios finales. Es un proceso cíclico y con retroalimentación.

Definiciones Clave

- **Inteligencia de Negocio (BI):** Proceso de analizar datos para extraer información útil que ayude a la toma de decisiones empresariales.
- **Ciencia de Datos (Big Data):** Disciplina que estudia los datos para entender fenómenos, tomar decisiones y resolver problemas, combinando matemáticas, programación y conocimiento del dominio.
- **ETL (Extract, Transform, Load):** Proceso de **Extraer** datos de diversas fuentes, **Transformarlos** (limpiarlos, darles formato) y **Cargarlos** en un almacén de datos (Data Warehouse).
- **Empresa Data-Driven:** Organización que basa su toma de decisiones estratégicas en el análisis de datos, complementando la experiencia humana.
- **Modelo de Proceso:** El conjunto de todas las tareas a realizar en un proyecto (ej. extracción, limpieza, modelado).
- **Metodología:** El 'cómo' se realizan las tareas, siguiendo reglas y buenas prácticas.
- **Ciclo de Vida:** El orden en que se ejecutan las tareas del modelo de proceso.

Módulo 2: Tipos, Calidad y Preparación de Datos en R

Tipos de Datos

- **Datos Estructurados:** Datos que pueden organizarse en un formato de tabla (filas y columnas). Ej: Archivos Excel, CSV, bases de datos relacionales.
- **Datos No Estructurados:** Datos sin un formato predefinido, no se pueden poner fácilmente en una tabla. Ej: Textos (Word, PDF), correos, imágenes, vídeos, audio.
- **Datos Semi-estructurados:** Tienen cierta estructura pero no encajan en una tabla rígida. Ej: JSON, XML, formularios con secciones marcadas.

Propiedades de la Calidad de los Datos (Las 5 'C' + I)

- **Completitud:** Que no haya valores nulos o vacíos.

- **Credibilidad (Fiabilidad):** La fuente de los datos es confiable (ej. organismos oficiales vs. una persona anónima).
- **Precisión:** Que los datos no contengan errores (ej. un importe negativo donde no debería).
- **Consistencia:** El mismo dato debe tener el mismo valor en diferentes tablas o sistemas (ej. una persona no puede ser 'soltera' en una tabla y 'casada' en otra).
- **Interpretabilidad:** Que los nombres de las columnas y los valores sean comprensibles o estén documentados.

Tratamiento de Datos en R

- **Valores Nulos (NA):** Se pueden **eliminar las filas** (si son pocas), **eliminar la columna** (si no es relevante o tiene demasiados nulos), o **reemplazar (imputar)** con un valor como cero, la media o la mediana.
- **Outliers (Valores Anómalos):** Puntos de datos que son extraños o muy diferentes del resto (ej. una nota de 1000 en una escala de 1 a 10). Pueden ser errores o datos reales pero atípicos. La **mediana** es más robusta a outliers que la **media**.
- **Transformación de Variables:** Crear nuevas columnas o modificar existentes. Ejemplo clave: convertir una 'fecha de nacimiento' a 'edad'.
- **Normalización/Estandarización:** Poner todas las variables numéricas en una misma escala (ej. de -2 a 2) para que el algoritmo no dé más importancia a las que tienen valores más grandes. Se usa la fórmula **(valor - media) / desviación_típica**.
- **Variables Categóricas a Numéricas:** La mayoría de los algoritmos necesitan números. Las variables de texto (categóricas) se deben convertir a números. En R, se utiliza el tipo factor, que internamente asigna un número a cada categoría.

Código R relevante:

codeR

```
library(readr) # Cargar librería para leer CSV
library(dplyr) # Cargar librería para manipulación de datos
datos <- read.csv('ruta/a/tu/archivo.csv')
str(datos) # Ver la estructura y tipos de datos de las columnas
summary(datos) # Obtener estadísticas descriptivas y conteo de NAs
colSums(is.na(datos)) # Contar valores nulos por columna
datos_limpios <- na.omit(datos) # Eliminar todas las filas con
algún NA
datos$columna[is.na(datos$columna)] <- 0 # Reemplazar NAs por 0 en
una columna
datos$columna_factor <- as.factor(datos$columna_texto) # Convertir
a factor
```

Módulo 3: Estadística Descriptiva y Relaciones entre Variables

Medidas de Tendencia Central (Buscan el 'centro' de los datos)

- **Media Aritmética (mean()):** La suma de todos los valores dividida por el número total de valores. Es muy sensible a outliers.
- **Media Ponderada:** Similar a la media, pero a cada valor se le asigna un 'peso' o importancia.
- **Media Recortada (mean(…, trim=…)):** Se calcula la media después de eliminar un porcentaje de los valores más altos y más bajos, haciéndola más robusta a outliers.
- **Mediana (median()):** El valor que se encuentra en el centro de los datos cuando se ordenan de menor a mayor. Es la mejor medida de tendencia central cuando hay outliers.
- **Moda:** El valor que más se repite. Se usa principalmente para variables categóricas (texto).

Medidas de Dispersión (Miden qué tan 'esparcidos' están los datos)

- **Varianza (var()):** Mide la dispersión de los datos alrededor de la media. Un valor alto significa datos muy dispersos.
- **Desviación Típica (o Estándar) (sd()):** Es la raíz cuadrada de la varianza. Se interpreta más fácilmente porque está en las mismas unidades que los datos originales.
- **Rango:** La diferencia entre el valor máximo y el valor mínimo.
- **Cuantiles (summary()):** Valores que dividen los datos ordenados en partes iguales. Los más comunes son los **cuartiles** (dividen en 4 partes): **Q1** (25%), **Q2** (50%, es la mediana), **Q3** (75%).
- **Coeficiente de Variación:** Se calcula como (desviación_típica / media) * 100. Permite comparar la dispersión de variables con diferentes escalas. Un valor > 30% (o 0.3) se considera una dispersión significativa.

Relación entre Variables: Covarianza y Correlación

- **Covarianza (cov()):** Indica la dirección de la relación lineal entre dos variables. Su signo es lo más importante: **Positivo** (ambas variables se mueven en la misma dirección), **Negativo** (se mueven en direcciones opuestas), **Cercano a cero** (no hay relación lineal).
- **Correlación (cor()):** Mide tanto la **dirección** como la **fuerza** de la relación lineal. Es una métrica estandarizada que va de -1 a 1.

- **Interpretación del Coeficiente de Correlación (ρ):**
  - **-1:** Correlación negativa perfecta (línea recta descendente).
  - **-1 a -0.5:** Correlación negativa **fuerte**.
  - **-0.5 a 0:** Correlación negativa **débil**.
  - **0:** Sin correlación lineal.
  - **0 a 0.5:** Correlación positiva **débil**.
  - **0.5 a 1:** Correlación positiva **fuerte**.
  - **1:** Correlación positiva perfecta (línea recta ascendente).
- **¡MUY IMPORTANTE!:** Correlación no implica causalidad. Que dos variables se muevan juntas no significa que una cause a la otra.

Código R relevante:

codeR

```
media <- mean(datos$columna, na.rm = TRUE)
mediana <- median(datos$columna, na.rm = TRUE)
varianza <- var(datos$columna, na.rm = TRUE)
desv_tipica <- sd(datos$columna, na.rm = TRUE)
matriz_cor <- cor(datos_numericos, use = 'complete.obs')
matriz_cov <- cov(datos_numericos, use = 'complete.obs')
```

Módulo 4: Aprendizaje Supervisado (Predicción y Clasificación)

Conceptos Fundamentales

- **Aprendizaje Supervisado:** El algoritmo aprende de datos ‘etiquetados’, donde ya conocemos el resultado o la categoría correcta. El objetivo es predecir este resultado para datos nuevos.
- **Variable Dependiente (Objetivo o Target):** La variable que queremos predecir (la ‘Y’).
- **Variables Independientes (Predictoras o Features):** Las variables que usamos para predecir la variable dependiente (las ‘X’).
- **Entrenamiento y Testeo (Train/Test Split):** Es **crucial** dividir el dataset. Se usa una parte grande (ej. 70-80%) para **entrenar** el modelo (donde aprende las relaciones) y una parte pequeña (20-30%) para **testear** su rendimiento en datos que no ha visto antes.

- **Sesgo vs. Varianza:** Un modelo con **alto sesgo** es demasiado simple y no aprende bien (underfitting). Un modelo con **alta varianza** se ajusta demasiado a los datos de entrenamiento y no generaliza a datos nuevos (overfitting).

Código R relevante:

codeR

```
library(caret) # Librería para partición de datos y matriz de
confusión
set.seed(123) # Para reproducibilidad
trainIndex <- createDataPartition(datos$variable_objetivo, p =
0.8, list = FALSE)
train_set <- datos[trainIndex, ]
test_set <- datos[-trainIndex, ]
```

Regresión Lineal (Para predecir un valor numérico)

- **Objetivo:** Predecir una variable numérica continua (ej. precio de una casa, ventas futuras).
- **Modelo:** Busca encontrar la 'línea recta' (o hiperplano) que mejor se ajusta a los datos. La fórmula es $Y = \beta_0 + \beta_1X_1 + \beta_2X_2 + \dots$
- **Interpretación de Coeficientes (summary(modelo)):** Cada coeficiente (β) indica cuánto cambia la variable Y por cada unidad que aumenta la variable X correspondiente, manteniendo las demás constantes.
- **Métrica de Evaluación (R-cuadrado o R^2):** Indica el porcentaje de la variabilidad de la variable Y que es explicado por las variables X del modelo. Va de 0 a 1. Un valor más cercano a 1 indica un mejor ajuste.

Código R relevante:

codeR

```
modelo_lineal <- lm(variable_a_predecir ~ var_indep_1 +
var_indep_2, data = train_set)
summary(modelo_lineal) # Muestra coeficientes, R-cuadrado, p-
valores, etc.
predicciones <- predict(modelo_lineal, newdata = test_set)
```

Clasificación (Para predecir una categoría)

- **Objetivo:** Predecir a qué categoría pertenece una observación (ej. 'cliente' vs 'no cliente', 'spam' vs 'no spam').
- **Regresión Logística (glm):** Se usa para clasificación **binaria** (solo dos categorías). Predice la probabilidad de que una observación pertenezca a una de las clases.

- **Árbol de Decisión (rpart):** Crea un modelo similar a un diagrama de flujo con reglas del tipo 'si-entonces'. Es muy interpretable y puede usarse para clasificación binaria o multiclase. Sirve para ver la **importancia de las variables**.

Código R relevante:

codeR

```
# Regresión Logística (Binaria)
modelo_logit <- glm(variable_binaria ~ var_1 + var_2, data =
train_set, family = 'binomial')
# Árbol de Decisión (Binaria o Multiclase)
library(rpart)
library(rpart.plot)
modelo_arbol <- rpart(variable_a_clasificar ~ var_1 + var_2, data
= train_set, method = 'class')
rpart.plot(modelo_arbol) # Para visualizar el árbol
```

Evaluación de Modelos de Clasificación: La Matriz de Confusión

- **Matriz de Confusión:** Tabla que compara los valores reales con los valores predichos por el modelo. Es la métrica **fundamental** para evaluar un clasificador.

- **Accuracy (Precisión Global):** (Verdaderos Positivos + Verdaderos Negativos) / Total. Porcentaje de predicciones correctas en general. Puede ser engañosa si las clases están desbalanceadas.

- **Sensitividad (Recall o Tasa de Verdaderos Positivos):** Verdaderos Positivos / (VP + Falsos Negativos). De todos los que eran realmente positivos, ¿cuántos predijo correctamente el modelo? Mide la capacidad de detectar los positivos.

- **Especificidad:** Verdaderos Negativos / (VN + Falsos Positivos). De todos los que eran realmente negativos, ¿cuántos predijo correctamente? Mide la capacidad de detectar los negativos.

Código R relevante:

codeR

```
probabilidades <- predict(modelo_logit, newdata = test_set, type =
'response')
predicciones_clase <- ifelse(probabilidades > 0.5, 1, 0)
predicciones_clase_factor <- as.factor(predicciones_clase)
referencia_factor <- as.factor(test_set$variable_binaria)
confusionMatrix(data = predicciones_clase_factor, reference =
referencia_factor)
```

Módulo 5: Aprendizaje No Supervisado (Descubrimiento de Patrones)

Conceptos Fundamentales

- **Aprendizaje No Supervisado:** El algoritmo trabaja con datos 'no etiquetados', es decir, no conocemos el resultado o la categoría de antemano. El objetivo es descubrir patrones, estructuras o grupos ocultos en los datos.
- **No se divide en entrenamiento y testeo:** Como no hay una 'respuesta correcta' para comparar, el modelo se aplica a todo el conjunto de datos para encontrar la estructura inherente.

Clustering o Agrupamiento (K-Means)

- **Objetivo:** Agrupar observaciones similares en 'clústeres' o grupos. Las observaciones dentro de un mismo clúster son muy parecidas entre sí, y muy diferentes a las de otros clústeres.
- **K-Means:** Un algoritmo popular que agrupa los datos en un número **K** de clústeres predefinido.
- **¿Cómo determinar el número óptimo de clústeres (K)?:**
  - **Método del Codo (Elbow Method):** Se grafica una métrica de error (WSS) para diferentes valores de K. El 'codo' del gráfico indica el punto donde añadir más clústeres ya no mejora significativamente el modelo.
  - **Método de la Silueta:** Mide qué tan bien agrupada está cada observación.
  - **NbClust() en R:** Una función que ejecuta múltiples tests estadísticos y 'vota' por el número de clústeres más adecuado. Es el método más recomendado en el curso.
- **Interpretación de Clústeres:** Una vez formados los grupos, se analizan las características medias de cada uno para darles un significado de negocio (ej. 'clientes de alto valor', 'clientes de bajo compromiso'). Se puede usar un árbol de decisión como 'trampa' para interpretar las reglas que definen cada clúster.

Código R relevante:

codeR

```
library(NbClust) # Para determinar el número óptimo de clústeres
datos_numericos <- datos[, sapply(datos, is.numeric)] #
Seleccionar solo columnas numéricas
res_nbclust <- NbClust(data = datos_numericos, min.nc = 2, max.nc
= 8, method = 'kmeans')
# El resultado de res_nbclust sugiere el mejor número de clústeres
resultado_kmeans <- kmeans(datos_numericos, centers = 3) #
Ejecutar con K=3
print(resultado_kmeans$centers) # Ver las medias de cada clúster
```

```
para interpretarlos
table(resultado_kmeans$cluster, datos$especie_real) # Comparar con
una clasificación real si existe
```

Reducción de Dimensión (PCA)

- **Objetivo:** Reducir el número de variables (columnas) de un dataset, manteniendo la mayor cantidad de información posible. Útil para datasets con cientos de columnas.
- **Análisis de Componentes Principales (PCA):** Es la técnica más común. Transforma las variables originales en un nuevo conjunto de variables (componentes principales) que no están correlacionadas entre sí y que capturan la mayor parte de la varianza (información) de los datos.

Módulo 6: Series Temporales

Conceptos Fundamentales

- **Serie Temporal:** Una secuencia de puntos de datos medidos a intervalos de tiempo regulares (ej. ventas diarias, temperatura mensual).
- **Objetivo:** Analizar la evolución de una variable en el tiempo para entender su comportamiento y hacer predicciones futuras.
- **Componentes de una Serie Temporal:**
  - **Tendencia:** La dirección general a largo plazo de la serie (ascendente, descendente o lateral).
  - **Estacionalidad:** Patrones que se repiten en intervalos fijos y conocidos (ej. ventas de helados más altas en verano).
  - **Ciclo:** Patrones que se repiten pero en intervalos no fijos, a menudo relacionados con ciclos económicos.
  - **Componente Irregular (Ruido):** Variaciones aleatorias e impredecibles.
- **Estacionariedad:** Una propiedad crucial. Una serie es estacionaria si sus propiedades estadísticas (como la media y la varianza) no cambian con el tiempo. Muchos modelos, como ARIMA, requieren que la serie sea estacionaria.

Predicción con ARIMA en R

- **ARIMA (Autoregressive Integrated Moving Average):** Un modelo estadístico muy popular para analizar y predecir datos de series temporales.
- **auto.arima():** Una función de la librería forecast que simplifica enormemente el proceso. Encuentra automáticamente el mejor modelo ARIMA para los datos, sin necesidad de especificar manualmente sus parámetros (p, d, q).

- **Pasos para la predicción:**

    1. Convertir los datos a un objeto de serie temporal (ts()), especificando la fecha de inicio (start) y la frecuencia (frequency). frequency=12 para datos mensuales, frequency=4 para trimestrales, frequency=52 para semanales.

    2. Usar auto.arima() sobre el objeto ts para encontrar el mejor modelo.

    3. Usar forecast() sobre el modelo ARIMA resultante para predecir los próximos N periodos.

Código R relevante:

codeR

```
library(forecast) # Librería para series temporales
# Convertir a objeto de serie temporal (datos mensuales desde
Enero 2000)
serie_ts <- ts(datos$columna_valor, start = c(2000, 1), frequency
= 12)
# Encontrar el mejor modelo ARIMA automáticamente
modelo_arima <- auto.arima(serie_ts)
# Hacer una predicción para los próximos 12 meses
prediccion <- forecast(modelo_arima, h = 12)
plot(prediccion) # Graficar la serie original y la predicción con
intervalos de confianza
print(prediccion) # Ver los valores predichos
```

Preguntas de Examen Probables

Conceptos y Procedimientos Generales

- **Diferenciar entre Aprendizaje Supervisado y No Supervisado:** ¿Cuándo usarías un modelo de clasificación (ej. Regresión Logística) y cuándo uno de clustering (ej. K-Means)?

- **Tratamiento de Datos Nulos:** Dada una situación, justificar si eliminarías las filas, la columna o reemplazarías los valores nulos (y con qué valor, ej. media o mediana).

- **Importancia de la división Entrenamiento/Testeo:** ¿Por qué es necesario dividir los datos en modelos supervisados? ¿Qué problema se busca evitar (overfitting)?

- **Interpretación de Correlación:** Dada una matriz de correlación, identificar la relación más fuerte/débil, y explicar si es positiva o negativa y qué significa en el contexto del problema.

Modelos Específicos y su Interpretación

- **Regresión Lineal:** Dado el summary() de un modelo lm(), escribir la ecuación de la recta e interpretar el significado de un coeficiente específico. Explicar el R-cuadrado.

- **Regresión Logística y Árbol de Decisión:** Dada la confusionMatrix() de dos modelos, determinar cuál tiene mayor Accuracy (precisión) y explicar qué significan la Sensitivity y Specificity en el contexto del problema.

- **Clustering (K-Means):** Dado un gráfico del 'método del codo' o la salida de NbClust, determinar y justificar el número óptimo de clústeres a utilizar.

- **Interpretación de Clústeres:** Dada la salida de los centros (centers) de K-Means, describir las características de cada clúster (ej. 'Clúster 1 son flores pequeñas, Clúster 2 son flores grandes').

- **Series Temporales (ARIMA):** Dado un problema, explicar los pasos para realizar una predicción, incluyendo cómo crear el objeto ts (especificando start y frequency) y cómo usar auto.arima y forecast.

Operaciones Clave en R (Saber aplicar el código)

- **Filtrado de Datos con dplyr:** Seleccionar un subconjunto de datos que cumpla una o varias condiciones (ej. 'aguacates orgánicos vendidos en Albany').

- **Agrupación y Resumen con dplyr:** Calcular una métrica (ej. la media del precio) para diferentes grupos (ej. por cada ciudad). Uso de group_by() y summarize().

- **Creación de una nueva columna:** Usar ifelse() para crear una variable binaria (0/1) a partir de una variable categórica (ej. 'sí'/'no').