INSTITUTO TECNOLÓGICO DE BUENOS AIRES – ITBA

ESCUELA DE INGENIERÍA Y GESTIÓN

# Caylent

# Talent Acquisition hiring process

AUTOR/ES: Nasillo, Agustin Gabriel (Leg. Nº 56304)

DOCENTE/S TITULAR/ES O TUTOR/ES:

Nosetti, María Inés - mnosetti@itba.edu

González Rodríguez, Rubén Darío - rgonzalez@itba.edu.ar

Rodriguez Varela, Juan Pablo - juanrodr@itba.edu.ar

TRABAJO FINAL PRESENTADO PARA LA OBTENCIÓN DEL TÍTULO DE LICENCIADO EN ANALÍTICA EMPRESARIAL Y SOCIAL

BUENOS AIRES

SEGUNDO CUATRIMESTRE, 2021

# CAYLENT

# Talent Acquisition hiring process

## Content

# *Entrega 1*

## Objetivo del proyecto

### *¿Cuál es el problema?*

El proceso de contratación de **Caylent** tiene cómo objetivo brindar a todos los candidatos a **Engineer** y **Solutions Architect** un *assessment de prueba de trabajo* para diseñar y describir una solución que cumpla con algunos criterios específicos, al tiempo que permite un *amplio margen de diferenciación entre los candidatos*. Esta evaluación, combinada con las puntuaciones del **GMA** y los resultados de las rondas de **entrevistas técnicas**, constituye la mayoría de los datos utilizados para llegar a una decisión de contratación.

Desde un punto de vista de procesos, el flujo de selección está conformado por cinco (5) instancias, las cuales se describen a continuación:

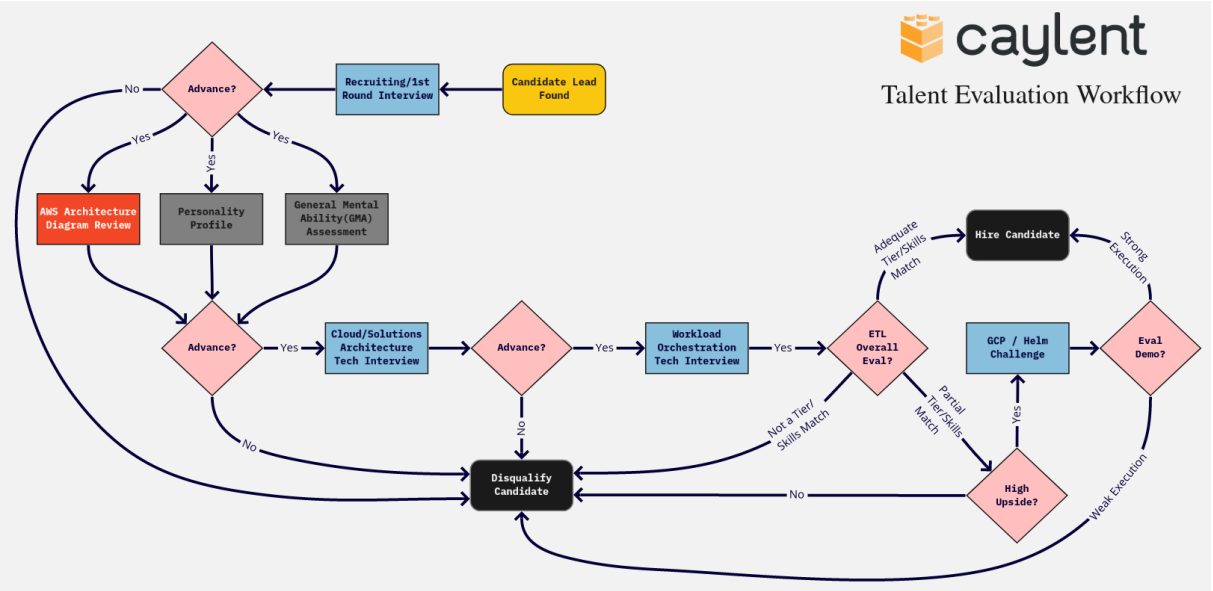



***Imagen 1:*** *Workflow de evaluación del candidato.*

Para comprender mejor las etapas que acompañan el flujo observado previamente en la **Imagen 1**, el mismo fue identificado de la siguiente manera:

1. **Recruiting / 1st Round Interview:** En esta instancia, se hace un screening por videollamada con el candidato para conocerlo, contarle acerca de la compañía y entender cuáles son sus motivaciones para un cambio laboral.

2. **General Mental Ability (GMA) assessment / Personality profile:** La capacidad mental general se define como la capacidad de comprender e

interpretar información verbal, la capacidad de percibir y procesar números e información dada en formato tabular / gráfico, la capacidad de pensar lateralmente y hacer conexiones lógicas entre diferentes conceptos, mientras que el perfil de personalidad busca categorizar al empleado para entender un poco mejor sus conductas y valores.

3. **Architecture Diagram:** Es el diseño de una arquitectura propia según requisitos mínimos, esto permite al candidato emplear sus conocimientos de la mejor manera para el diseño de un diagrama similar al que se utiliza en una propuesta técnica. De esta manera, no solo se recaban puntos acerca del conocimiento y la experiencia del mismo, sino que permite obtener información acerca de la redacción, lenguaje, estética y atención al detalle bajo una situación puntual de trabajo.

4. **Cloud / Solutions Architecture Interview:** La primera de dos entrevistas técnicas estructuras, representan un formato pautado y poseen una serie de preguntas que se realizan acerca de conocimientos de un *Cloud Service Provider*, típicamente **AWS**. El nivel de la entrevista recorre *niveles más básicos hasta preguntas de arquitectura realmente complejas*, acorde a las mejores prácticas y recomendaciones de **AWS**. A continuación, se describe una **fragmento** de la entrevista:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| EC2 | EC2 | Points | "Can you take me through the EC2 provisioning and configuration process? I'd like to understand both all the things you must configure to provision EC2 as well as all the optional items I can configure during this process." | | | | | | |
| | ☑ | 0.25 | If they understand instance types | | | | | | |
| | ☑ | 0.25 | If they understand the role of the AMI | | | | | | |
| | ☑ | 0.25 | If they understand the role of keypairs and the presence of ec2-user in default Amazon Linux AMI's and/or centos for centos or ubuntu for Ubuntu Server | | | | | | |
| | ☑ | 0.25 | If they included mention of selecting both the VPC and subnet | | | | | | |
| | ☑ | 0.25 | If they included the selection or creation of a security group | | | | | | |
| | ☑ | 0.25 | If they include EBS block storage provisioning | | | | | | |
| | ☑ | 0.25 | If they mention the EC2 instance can be associated with an IAM Role | | | | | | |
| | ☑ | 0.25 | If they understand customization via user-data, instance tagging, public/private ephemeral IPs, elastic IP association, mention placement groups, mention enabling t2/t3 unlimited, mention shared/dedicated tenancy, GPU-enabled instance types, FPGA-enabled instance types, mention the ability to tick the "request spot instance" box, mention instance type families, EBS optimized instance types, mention instance store vs EBS, different EBS storage types, or attachment of additional EBS volumes or ENI interfaces. (.25 if they catch at least 3 relevant added features of EC2) | Mentioned EIP, spot instances, user-data | | | | | |
| | Total | 2 | 2 | | | | | | |
| | S3 | Points | "Please explain your understanding of how S3 as a service is architected and some of the key advantages this provides to data housed on it?" Mentions: | Mention half or more of the following attributes (.25 for 3+ .50 for 5+) | | | | | |
| | ☑ | | Regional Service/Replicated through a single region | Range | Points | | | | |
| | ☐ | | Globally unique names | 3 | 0.25 | | | | |
| | ☑ | | Eventual consistency replication (Dec 2020 - Now Strong Consistency) | 5 | 0.5 | | | | |
| | ☑ | | It's an Object Storage Service typically used to hold files | | | | | | |
| | ☑ | | High file durability | | | | | | |
| | ☑ | | Highly scalable | | | | | | |
| | ☑ | | High service availability | | | | | | |
| | ☐ | | Support for Static Website Hosting | | | | | | |
| | ☐ | | Mentioned S3 uses public endpoints by default unless using private vpc endpoints | | | | | | |
| | Total | 0.5 | 0.5 | | | | | | |

***Imagen 2:*** *Evaluación de AWS.*

5. **Workload Orchestration Tech Interview:** Esta etapa puede contener dos aristas. La **primera**, la instancia más ligada a **DevOps**, consta de *dos entrevistas parciales:* ***Terraform*** *y* ***Kubernetes***. La **segunda**, la instancia más ligada a **Developers**, consta de *una entrevista de* ***Serverless***. Esta última fue

recientemente implementada y la elección es a discreción del candidato, por lo que se espera que la cantidad de datos de esta muestra *sea significativamente menor al resto*. La **tercera** y última, la menos común, es la de Leadership, la cual evalúa los puntos necesarios para ser un **ETL** (Engineering Team Leader). Se conservarán los datos a fin de contrastarlos con los Engineers y Solutions Architects, pero se descartan para el input del modelo. A continuación, se describe una **fragmento** de la entrevista de Terraform:

| | Points | What is the process you should follow in making Terraform changes if you want to see potential configuration drift or conflicts prior to applying your changes? |
|---|---|---|
| ☑ | 0.25 | Terraform plan with -out argument to capture the plan to a file |
| ☑ | 0.25 | Terraform apply with the path to the file created via the "terraform plan -out …" command |
| **Total** | 0.5 | 0.5 |
| | **Points** | **Can you describe how terraform organizes resources and the relationship between them?** |
| ☑ | 0.25 | Know what module a is and that is a container for multiple resources that are used together. |
| ☑ | 0.25 | Know how to call a module using source, to call a module means to include the contents of that module into the configuration with specific values for its input variables, modules are called from within other modules using module block |
| ☑ | 0.25 | Know what is commonly seen in a module: inputs (variables), outputs, and resources defined within it |
| ☑ | 0.25 | Know that by using outputs from a module as input of the other modules you can tell terraform how to make better paths in the graph, be careful with cycles. |
| ☐ | 0.25 | Using the data access method to query data from the remote state (for passing as input to other modules/resources) in order to avoid hard-coding inputs |
| **Total** | 1 | 1.25 |
| | **Points** | **Explain the different methods you can use to provide input values into the terraform configuration.** |
| ☑ | 0.25 | Variables |
| ☑ | 0.25 | Locals |
| ☐ | 0.25 | In variable definitions (.tfvars) files |
| ☐ | 0.25 | Environment variables named TF_VAR_ followed by the name of a declared variable. |
| ☑ | 0.25 | Individually, with the -var command line option. |
| **Total** | 0.75 | 1.25 |

***Imagen 3:*** *Evaluación de Terraform.*

## *¿Por qué es un problema?*

Algunas de las situaciones (generales) que se presentan hoy en día en **Caylent** causan demasiada fricción en el proceso de selección, tanto para los **managers** así cómo también para los **candidatos**, quienes muchas veces reciben una categorización por fuera de lo que es su verdadero rango. Al ser un proceso tan **desacoplado**, y pasar por tantas personas, muchas veces se pierde ese hilo conector que une todas las piezas. Por el contrario, estableciendo una *baseline en el tratamiento y procesamiento de los datos*, y por sobre todo en la *decisión final de contratación (o no) y qué tipo de puesto (y salario)* ofrecerle al candidato, se abstrae la decisión a la de un sistema.

Por tanto, podemos afirmar que se considera un problema ya que existen desviaciones en el entendimiento de los datos actualmente, y en muchas de las ocasiones pueden presentarse diferencias entre los deseos de selección del tier del equipo de **Talent** versus a lo sugerido por el **ETL**. Se espera que este proyecto disminuya o elimine esta brecha, proporcionando una mirada objetiva y holística sobre el proceso en sí.

## *¿Qué se busca del proyecto?*

Primariamente, se busca que el proyecto *solucioné, elimine, reduzca o al menos modifique* el problema descrito en la sección anterior. Al mismo tiempo en que se cuenta con este objetivo primario, se esperan alcanzar **dos hitos**, *consecuentes uno del otro.*

**El primer hito** consiste en conseguir un **output** similar al que se muestra a continuación en la **Imagen 4**, la cual contiene información que se maneja en la actualidad. La diferencia radica en que se busca salvar aquellos **nulls / missings** que se encuentran en el proceso de varios candidatos, al mismo tiempo que se busca integrar la totalidad de los **170 candidatos**, ya que hasta este momento, solo se mantiene una lista de **51**.

El **segundo** hito, por su parte, consiste en la posibilidad de *establecer un baseline acerca de la contratación o no del candidato*, abstrayendo al equipo de **Talent / ETL del Si / No,** seguido de la respuesta a la próxima pregunta: qué **Tier** / **Level** deberíamos asignar a cada candidato que acordamos contratar.

De esta manera, se abarcara una **mayor transparencia** durante el punta a punta de contratación, *abstrayendo significativamente al evaluador de los resultados*.

| Round Level | EC2 Score (Max: 2.0) | S3 Arch (Max: 0.5) | S3 Access (Max: 0.5) | S3 Classes (Max: 0.5) | S3 DR (Max: 0.5) | IAM (Max: 1.0) | AWS Net (Max: 2.0) | AWS SolArch (Max: 2.0) | S3 Blended (Max: 2.0) | Tier I (Max 3.5) | Tier II (Max 5.5) | Overall % | Diag Grade | GMA % | End Result | Eng Level | 6 month Eval | 1 year Eval |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1/2 | 1.75 | 0.50 | 0.50 | 0.25 | 0.25 | 0.60 | 1.75 | 1.25 | 1.50 | 3.00 | 3.85 | 76.11% | B | 71% | Hired | 1 | | |
| DNF | 1.50 | 0.25 | 0.00 | 0.00 | DNF | DNF | DNF | DNF | 0.25 | 1.75 | 0.00 | 19.44% | C | 86% | DQ | - | | |
| 1/2 | 2.00 | 0.25 | 0.25 | 0.00 | 0.00 | 0.00 | 2.00 | 2.00 | 0.50 | 2.50 | 4.00 | 72.22% | C+ | 44% | Shelf | | | |
| 3 | 2.00 | 0.50 | 0.50 | 0.50 | 0.50 | 0.80 | 2.00 | 2.00 | 2.00 | 3.50 | 5.30 | 97.78% | B- | 53% | Hired | 3 | | |
| 1 | 2.00 | 0.50 | 0.25 | 0.00 | 0.25 | 0.00 | 2.00 | 1.00 | 1.00 | 2.75 | 3.25 | 66.67% | B- | 53% | - | - | | |
| 3 | 2.00 | 0.50 | 0.50 | 0.50 | 0.50 | 0.90 | 2.00 | 2.00 | 2.00 | 3.50 | 5.40 | 98.89% | B+ | 32% | Hired | 3 | | |
| 2 | 2.00 | 0.00 | 0.25 | 0.25 | 0.25 | 0.40 | 1.75 | 1.50 | 0.75 | 2.50 | 3.90 | 71.11% | D+ | | DQ | - | | |
| DNF | 1.75 | 0.25 | 0.25 | 0.25 | DNF | DNF | DNF | DNF | 0.75 | 2.50 | 0.00 | 27.78% | C- | 49% | DQ | - | | |
| 2/3 | 2.00 | 0.50 | 0.50 | 0.25 | 0.25 | 0.70 | 2.00 | 2.00 | 1.50 | 3.25 | 4.95 | 91.11% | C | 86% | Offer | 3 | | |
| 3 | 2.00 | 0.50 | 0.50 | 0.25 | 0.50 | 1.00 | 2.00 | 2.00 | 1.75 | 3.25 | 5.50 | 97.22% | B | 80% | Hired | 3 | | |
| 3A | 2.00 | 0.50 | 0.50 | 0.50 | 0.50 | 0.80 | 2.00 | 2.00 | 2.00 | 3.50 | 5.30 | 97.78% | A | 88% | Hired | 3 | | |
| 2 | 2.00 | 0.50 | 0.25 | 0.50 | 0.25 | 0.20 | 2.00 | 2.00 | 1.50 | 3.25 | 4.45 | 85.56% | B | 86% | Hired | 2 | | |
| 3 | 2.00 | 0.50 | 0.50 | 0.25 | 0.25 | 1.00 | 2.00 | 2.00 | 1.50 | 3.25 | 5.25 | 94.44% | B- | 80% | Hired | ETL | | |
| 1 | 1.75 | 0.25 | 0.25 | 0.00 | 0.00 | 0.40 | 2.00 | DNF | 0.50 | 2.25 | 2.40 | 51.67% | D- | 81% | Pulled | - | | |
| DNP | DNP | DNP | DNP | DNP | DNP | DNP | DNP | DNP | DNP | DNP | DNP | DNP | A- | 19% | Hired | 2 | | |
| 2/3 | 2.00 | 0.50 | 0.50 | 0.50 | 0.50 | 0.00 | 1.75 | 2.00 | 2.00 | 3.50 | 4.25 | 86.11% | C- | 58% | Offer | 3 | | |
| 2 | 1.75 | 0.25 | 0.50 | 0.00 | 0.25 | 0.80 | 2.00 | 1.75 | 1.00 | 2.50 | 4.80 | 81.11% | DNP | 86% | Hired | 3 | | |
| DNF | 2.00 | 0.25 | 0.00 | 0.25 | DNF | DNF | DNF | DNF | 0.50 | 2.50 | 0.00 | 27.78% | B- | 86% | DQ | - | | |
| DNF | 1.00 | 0.25 | 0.00 | 0.00 | DNF | DNF | DNF | DNF | 0.25 | 1.25 | 0.00 | 13.89% | D | 32% | DQ | - | | |
| 3 | 1.75 | 0.50 | 0.50 | 0.25 | 0.50 | 0.60 | 2.00 | 2.00 | 1.75 | 3.00 | 5.10 | 90.00% | B- | 58% | Pulled | - | | |
| 2 | 2.00 | 0.50 | 0.00 | 0.25 | 0.25 | 0.25 | 1.50 | 2.00 | 1.00 | 2.75 | 4.00 | 75.00% | B | | Pulled | - | | |
| 2 | 2.00 | 0.50 | 0.50 | 0.50 | 0.25 | 0.25 | 1.50 | 1.75 | 1.75 | 3.50 | 3.75 | 80.56% | C | 22% | Pulled | - | | |

***Imagen 4:*** *Matrix esperada de resultados.*

# Medición de valor y KPIs a impactar

## *¿Cómo se medirá el éxito del proyecto?*

El éxito del proyecto será medido por **dos (2) grupos** de stakeholders principales del proyecto: la **cátedra** y el equipo interno de **Talent / ETL**.

De parte del primero, la cátedra, se obtendrá una **puntuación numérica del 1 al 10**, la cual marcará la *opinión de los expertos* en base a lo visto / analizado durante todo el proceso.

De parte del segundo, los equipos internos de **Caylent**, *se podrá obtener o no el visto bueno* en relación a los **insights** generados al final del proyecto. Si se logra hacer, al menos un cambio p*or más pequeño que sea*, se considerará que el proyecto fue exitoso. Entiéndase que el cambio tendrá que sera una mejora, o en su defecto, el descubrimiento de una oportunidad de mejora que se *encuentre por fuera de los alcances, límites y facultades de este proyecto y del ejecutor* (quien redacta esto, mi persona), ya que se podrán **señalar, sugerir o recomendar modificaciones** sobre *procesos externos no directamente relacionados con este proyecto de Ciencia de Datos*. Esto quiere decir que, se podrán encontrar oportunidades de mejora sobre la forma en que sea el primer acercamiento desde Talent, o modificar los órdenes de las entrevistas, solo por mencionar algunos.

## *¿Cómo se medirá el valor e impacto en el negocio?*

En **primer** lugar, el impacto será medido a través de la **reducción de tiempos** del proceso *(medido directamente en $)*, mientras que adicionalmente y de forma consecuente, se encuentra valor en la **automatización y abstracción** no sólo de la **decisión**, sino también de la **validación de integridad** del proceso en sí mismo.

En **segundo** lugar, se busca generar conclusiones que **validen o rechacen el proceso de selección actual**, para entender si el mismo debe ser mantenido o replanteado. Para lograr esto, se puede correr el (futuro) pipeline del modelo con los empleados actuales, lo cual permitirá contrastar al candidato **previo a la incorporación**.

En **tercer** lugar, y quizás por fuera de los límites de este proyecto, se buscará realizar una obtención de datos a través de la API de **15Five**, la cual es utilizada para revisión y evaluaciones de desempeño. De esta manera, con su **desarrollo de carrera ya dentro de la compañía**, podremos entender si determinados candidatos incrementaron su performance luego de pasar a formar parte de ella.

## *¿Qué KPIs se utilizarán y cómo se calcularán?*

Una vez **conseguidos la totalidad de los datos**, se buscará construir un modelo que supere el umbral de **aciertos del 80 / 85 %**. De forma consecuente, y solo si se cuenta con toda la información necesaria para el correcto desarrollo de este proyecto, se construyen los siguientes KPIs que se analizaran para medir la performance del proyecto:

- ***Error de Tipo I:*** a la *cantidad de candidatos descalificados contratados*
- ***Error de Tipo II:*** a la *cantidad de candidatos calificados no contratados.*
- ***Cantidad de contratados****: cantidad de candidatos contratados / total de*

*candidatos (que participan del proceso[1]).*

Vale la pena señalar, que *el riesgo de* ***contratar a una persona que no debería ser contratada*** *es notablemente superior* al de *no contratar a alguien que hubiera sido un* ***buen-fit*** *de la compañía*, por tanto, una vez llegado el momento de medir cómo performa el modelo, el **Error de Tipo 1** será de mayor importancia.

# Entregables y outputs del proyecto

## *Entrega final*

Debido a la magnitud y complejidad del proyecto del proyecto, se establecen los siguientes ítems cómo entregables:

1. **Matriz de resultados** del candidato, acorde a lo explicado en la sección **Objetivo del Proyecto** bajo el ítem *Que se busca del proyecto*.
2. Al menos, dos (2) scripts con los procesos de **ETL** y conversión de datos correspondientes.
    a. En conjunto, a los scripts, se entregan más de 300 archivos **.xlsx** una vez finalizado el proceso de **ETL**.
3. **Repositorio** de versionado de código.
4. **Modelo productivo** que permita la ejecución remota de nuevos candidatos.
5. **Informe** de resultados estadísticos y conclusiones. Incluye próximos pasos.

## *¿Cómo se podrá utilizar el output del proyecto a futuro?*

Debido a la **parametrización** necesaria que se debe realizar a fin de cubrir el **gap** provocado por la *anonimización de datos*, los **scripts son plenamente reutilizables** para futuros candidatos, permitiendo un re-entrenamiento del modelo a fin de suplir **data-drifts** o simplemente proveer una **mayor robustez** en el proceso de entrenamiento.

# Abordaje del problema

## *¿Qué metodologías se aplicaran para la resolución del problema?*

Si bien este enunciado ya se ha respondido, en parte a lo largo de lo recorrido en este documento, y será en otra parte, respondido al final del mismo, se opta cómo elección **metodológica al esquema de ciencia de datos**, la cual consiste en *diez etapas*. Esta metodología, contiene un marco con *una serie de propuestas secuenciales*, que facilitan el

[1] En caso de presentar un número muy bajo se deberán revisar procesos de sourcing para alinearlos mejor con los perfiles buscados (ya sea experiencia, tecnologías, etc).

trabajo de y en este campo, a su vez que incrementa las opciones de éxito de un proyecto.

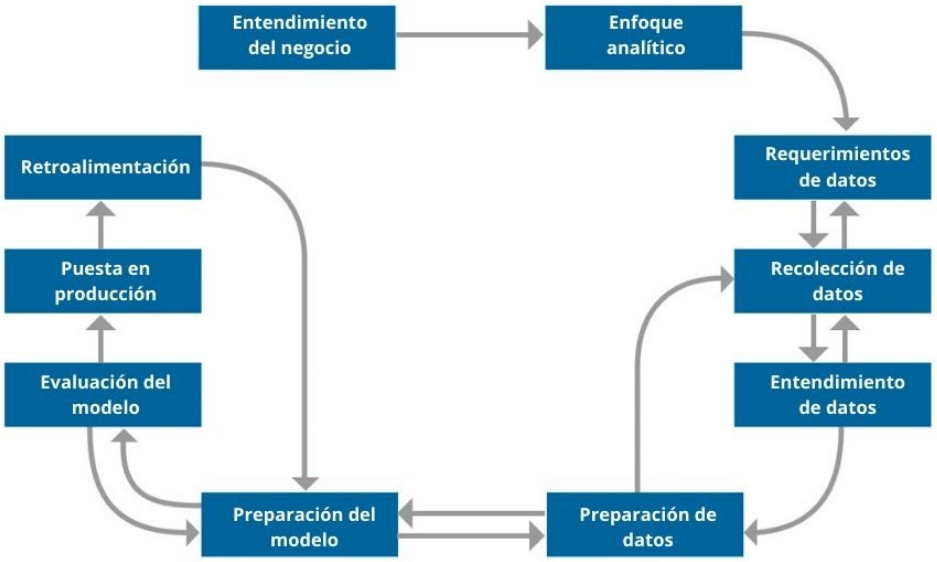



***Imagen 5:*** *Metodología de Ciencia de Datos.*

Para comprender un poco mejor el flujo, se acompaña cada etapa de una pregunta guía que modela la propuesta.

1. **Entendimiento del negocio:** *¿Qué problema se quiere resolver?*
2. **Enfoque analítico:** *¿Cómo se puede usar datos para responder la pregunta?*
3. **Requerimientos de datos:** *¿Qué datos se necesitan para responder la pregunta?*
4. **Recolección de datos:** *¿De dónde vienen los datos y cómo obtenerlos?*
5. **Entendimiento de datos:** *¿Son los datos recolectados lo suficientemente representativos para resolver el problema?*
6. **Preparación de datos:** *¿Qué trabajo extra se requiere para manipular y usar los datos?*
7. **Preparación del modelo:** *¿Cómo se puede visualizar los datos para obtener la respuesta esperada?*
8. **Evaluación del modelo:** *¿El modelo obtenido resuelve la pregunta inicial o necesita ser ajustado?*
9. **Puesta en producción:** *¿Se puede usar el modelo?*
10. **Retroalimentación:** *¿Se puede obtener retroalimentación para responder la pregunta inicial?*

Cómo se puede observar, no se encuentran menciones en una primera instancia, a ningún tipo de metodologías **Kanban** o **Agile**, ya que las mismas poseen un enfoque colaborativo orientado a equipos, y en este caso, la cantidad de integrantes del proyecto fue definida a una. Sin embargo, se espera que esta situación se revierta sola a lo largo del proceso, en cuanto y en tanto los **key-players** / **product owners** del proyecto comiencen a

tener una mayor relevancia sobre el mismo.

# Plan de Trabajo

## *¿Cómo se puede dividir el proyecto en fases?*

Para responder a esta pregunta, es preciso abstraernos a un clásico problema de Machine Learning. Las etapas que lo componen, se encuentran descritas a continuación.

1. **Recolección de datos / pre-procesamiento del dato**: es necesario realizar un pre-procesamiento por todo lo mencionado anteriormente, ya que no se encuentra con el source de datos de manera transparente, sino que la misma se encuentra ofuscada.
2. **EDA / ETL**: para entender cuales son nuestros y cómo se comportan, al mismo tiempo en que se busca entender anomalías y ver que se puede mejorar, reagrupar, modificar, quitar, o simplemente necesita ser convertido.
3. **Feature Engineering**: para poder entender qué label tiene mayor peso sobre el otro, y probar diversas metodologías y combinaciones, por ejemplo, entender si alguna pregunta/ítem del proceso de entrevistas tiene mayor / menor peso respecto de las otras.
4. **Entrenamiento del modelo**: prueba de distintas técnicas hasta obtener un resultado que cubra los requerimientos esperados.
5. **Evaluación del modelo y tunning de hyperparameters**: a fines de entender *cómo mejorar el modelo que tenemos*, cómo hacerlo performar mejor y entregar así mejores resultados.
6. **Debugeo de resultados y del ciclo completo proceso**: analizar el output del modelo y realizar las pruebas a niveles productivos, iteración del ciclo descrito en los puntos 2-5, para nuevo proceso de debugging hasta obtener resultados esperados.

## *¿Qué pasos se seguirán?*

En complemento con lo expresado en el ítem anterior, y adicional a lo ya estipulado en la sección Abordaje del problema se busca establecer una **metodología** similar a la descrita en la imagen que se encuentra a continuación. Claro está, que se plantea esta mecánica ya que el objetivo es la **puesta productiva de un modelo de machine learning**, según lo definido en la sección de Medición de valor y KPIs a impactar.

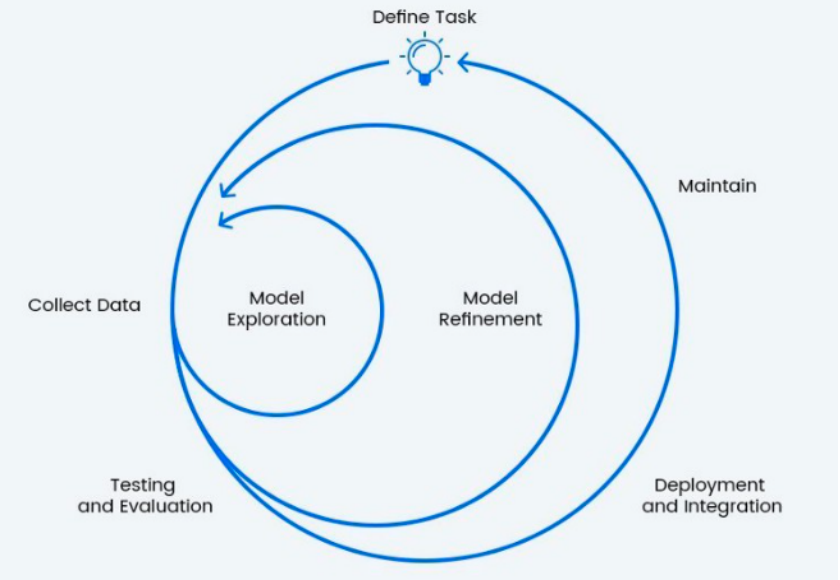



***Imagen 6:*** *Proceso iterativo de la ciencia de datos.*

## *¿Qué tiempos y riesgos se están estimando para el mismo?*

Respecto a la estimación de tiempos, se establecen como **cuatro (4) los meses** que se posee para completar esta *heroica acción*. A continuación se encuentra un **diagrama Gantt** con la *estimación de las tareas para todo el proyecto*.

| | | | | | 08/02/21 | 08/09/21 | 08/16/21 | 08/23/21 | 08/30/21 | 09/06/21 | 09/13/21 | 09/20/21 | 09/27/21 | 10/04/21 | 10/11/21 | 10/18/21 | 10/25/21 | 11/01/21 | 11/08/21 | 11/15/21 | 11/22/21 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Mes 1 | | | | Mes 2 | | | | Mes 3 | | | | | Mes 4 | | | |
| | Actividad | Inicio (semana) | Fin (semana) | Duración (semana) | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| | **Especificar requisitos y evaluar proyecto** | | | | | | | | | | | | | | | | | | | | |
| 1 | Recolección de requisitos | 1 | 1 | 1 | | | | | | | | | | | | | | | | | |
| 2 | Especificación preliminar | 2 | 2 | 1 | | | | | | | | | | | | | | | | | |
| 3 | Elaboración del presupuesto | 2 | 2 | 1 | | | | | | | | | | | | | | | | | |
| 4 | Revisión de especificaciones y presupuesto | 3 | 3 | 1 | | | | | | | | | | | | | | | | | |
| 5 | Incorporar mejoras en requisitos | 3 | 3 | 1 | | | | | | | | | | | | | | | | | |
| 6 | Elaborar el cronograma | 4 | 4 | 1 | | | | | | | | | | | | | | | | | |
| 7 | Consguir aprobación de Talent/Management | 5 | 5 | 1 | | | | | | | | | | | | | | | | | |
| 8 | Asegurar los archivos .xlsx necesarios | 4 | 5 | 2 | | | | | Entrega | | | | | | | | | | | | |
| | **Análisis & Diseño** | | | | | | | | | | | | | | | | | | | | |
| 9 | Analisis de archivos de evaluaciónes | 4 | 4 | 1 | | | | | | | | | | | | | | | | | |
| 10 | Arquitectura e ingesta (K8, AWS, TF) | 4 | 5 | 2 | | | | | | | | | | | | | | | | | |
| 11 | ETL de datos | 6 | 11 | 6 | | | | | | | | | | | | | | | | | |
| 12 | EDA del proyecto | 6 | 11 | 6 | | | | | | | | | | | | | | | | | |
| 13 | Revisión de diseño | 6 | 11 | 6 | | | | | | | | | | | | | | | | | |
| 14 | Generar documentación pertinente | 8 | 14 | 7 | | | | | | | | | | | | Preview | | Entrega | | | |
| | **Ciencia de Datos** | | | | | | | | | | | | | | | | | | | | |
| 15 | Testeo de modelos | 10 | 11 | 2 | | | | | | | | | | | | | | | | | |
| 16 | Validación de métricas de performance | 11 | 12 | 2 | | | | | | | | | | | | | | | | | |
| 17 | Orquestación de la ingesta de datos | 10 | 11 | 2 | | | | | | | | | | | | | | | | | |
| 18 | Ajustar modelo (data drift, etc) | 11 | 12 | 2 | | | | | | | | | | | | | | | | | |
| 19 | Puesta de testeo del modelo | 12 | 14 | 3 | | | | | | | | | | | | Preview | | Entrega | | | |
| | **Validación y extras** | | | | | | | | | | | | | | | | | | | | |
| 20 | Estudiar modelo desarrollado | 13 | 13 | 1 | | | | | | | | | | | | | | | | | |
| 21 | Validar output con equipo Talent | 14 | 15 | 2 | | | | | | | | | | | | | | | | | |
| 22 | Detectar y corregir fallos | 14 | 15 | 2 | | | | | | | | | | | | | | | | | |
| 23 | Documentar e informar | 15 | 17 | 3 | | | | | | | | | | | | | | | | | |
| 24 | Peer Review | 15 | 15 | 1 | | | | | | | | | | | | | | | Entrega | | |
| 25 | Informe Final | 17 | 17 | 1 | | | | | | | | | | | | | | | | | Entrega |
| | **Despliegue** | | | | | | | | | | | | | | | | | | | | |
| 26 | Establecer estrategia de despliegue | 16 | 16 | 1 | | | | | | | | | | | | | | | | | |
| 27 | Desarrollar metodología de despliegue | 16 | 17 | 2 | | | | | | | | | | | | | | | | | |
| 28 | Revision de solucion acorde a espectativas | 17 | 17 | 1 | | | | | | | | | | | | | | | | | |
| 29 | Formación a usuarios (Talent / ETL) | 17 | 17 | 1 | | | | | | | | | | | | | | | | | |
| 30 | Despliegue productivo del modelo | 17 | 17 | 1 | | | | | | | | | | | | | | | | | |

***Imagen 7:*** *Diagrama Gantt del proyecto*

Respecto a los riesgos, se diseñó la siguiente **matriz de 13 riesgos** que busca identificar aquellos potenciales riesgos que pudieran entorpecer el desarrollo de la solución. Se calcula la probabilidad del riesgo mediante la ponderación entre la **probabilidad de ocurrencia del impacto (P)** y la **severidad de dicho impacto (I)**, de tal manera que la misma queda conformada como **P x I = Riesgo**.

| REF | # | CAPÍTULO / DESCRIPCIÓN DEL RIESGO | P (Probabilidad Ocurrencia) | I (Severidad Impacto) | P x I (R) (Categoría de Riesgo) | ACCIÓN A ENCARAR | DETALLE DE ACCIÓN |
|---|---|---|---|---|---|---|---|
| 1 | | **Área operativa** | | | | | |
| | 1 | Desvinculacion de la compania | 0.20 | 0.80 | 0.16 | **Aceptar** | Entender que, ya sea por decision propia o ajena, ya no se podra contar con los insights de ciertos referentes del proceso. |
| | 2 | Falta de datos / inconsistencia | 0.40 | 0.90 | 0.36 | **Mitigar** | Evaluar la posibilidad de generar datos mediante un proceso alternativo como SMOTE o similar, a fin de cubrir los gaps que puedan encontrarse |
| | 3 | Fuga de información a mitad de proyecto | 0.10 | 0.40 | 0.04 | **Mitigar** | Validar que tanto el equipo de Talent como los ETL poseen el conocimiento necesario para oficiar de stakeholders durante el desarrollo del proyecto |
| | 4 | Falta de documentación | 0.10 | 0.30 | 0.03 | **Mitigar** | Diseñar un flujo de procesos para evitar futura deuda técnica |
| 2 | | **Implementación** | | | | | |
| | 5 | Dificultad de integracion con API existente | 0.60 | 0.50 | 0.30 | **Delegar** | Evaluar la posibilidad de contar con una persona extra de ayuda respecto a la tarea de desarrollo de la API de 15Five |
| | 6 | Modelo de Machine Learning no eficiente | 0.50 | 0.70 | 0.35 | **Mitigar** | Iterar de forma recursiva hasta encontrar un modelo con mejor accuracy y sensibilidad y especificidad |
| | 7 | Retraso en tiempo de implementación | 0.70 | 0.60 | 0.42 | **Mitigar** | Establecer un backlog y fijar los sprints del proyecto |
| 3 | | **Desarrollo** | | | | | |
| | 8 | Erronea definición del alcance del problema | 0.20 | 0.40 | 0.08 | **Mitigar** | Realizar revisiones continuas con el equipo de Talent para validar los avances. |
| | 9 | Utilización de una metodología de desarrollo inapropiada | 0.30 | 0.80 | 0.24 | **Mitigar** | Evaluar antes del comienzo del proyecto las metodologías disponibles e indagar en su utilización en otros proyectos de similar naturaleza. |
| | 10 | Fallar en la definición de futuros pasos | 0.50 | 0.70 | 0.35 | **Mitigar** | Una vez definidos los requerimientos realizar un mapeo de los puntos a desarrollar con los conocimientos requeridos. |
| 4 | | **Externos** | | | | | |
| | 11 | Cierre de Caylent | 0.10 | 0.90 | 0.09 | **Aceptar** | Considerar la posibilidad de que Caylent pierda el funding y comience su cierre de operaciones. |
| | 12 | Nueva ley, normativa o regulación, ligada a GDPR por ejemplo | 0.50 | 0.80 | 0.4 | **Aceptar** | Realizar análisis de la ley y cumplirla. |
| | 13 | Más/menos números de usuarios de lo planificado | 0.30 | 0.60 | 0.18 | **Mitigar** | Realizar una arquitectura escalable y desacoplada qué permita un crecimiento elástico de la aplicación. |

***Imagen 8:*** *Matriz de riesgos*

# Datos a utilizar

Para una primera iteración del proyecto, se espera poder ingestar cerca de **300 archivos** que corresponden a **170 candidatos** que fueron tomados o descartados a lo largo de todo el proceso. Las fuentes de datos serán **cinco** ‘ ***.xlsx** ’ con formatos diferentes *(AWS, Terraform, Kubernetes, Serverless, Leadership)* y se espera poder realizar una conversión de todas los datasources a fin de *unificarlos en una matriz general* que contenga el **hash del empleado**, y **los resultados** de las distintas evaluaciones.

Dado que la ingesta será semi-automatizada y que los datos a los cuales se acceden están ofuscados, por temas de compliance en la anonimización de los nombres de los empleados, se deberá tener mucho cuidado con el proceso de **hashing** que realizará en el script, dado que será ejecutada por HR / externo y no por el equipo que interviene en este proyecto.

# Herramientas a utilizar

El uso de las herramientas podrá variar, y se espera que varíe, a lo largo del proyecto. Sin embargo, a continuación se describen aquellas premisas o ideas primarias.

## *Programming languages*

- Python 3, en cualquiera de sus versiones 3.7 / 3.8 / 3.9.
- Bash, para las manipulaciones de archivos a nivel de S.O, entre otros.

## *Tools*

- Google Colaboratory
- Visual Studio Code
- Github
- Jira

# *Entrega 2*

## Correcciones realizadas

### *Enunciados no correlacionados*

Respecto a los enunciados *¿Por qué es un problema?, ¿Qué se busca del proyecto? y ¿Cómo se medirá el éxito del proyecto?, los cuales se encontraban desconectados,* los mismos fueron reestructurados según los comentarios y el feedback recibido.

### *Project charter*

Respecto al *Gantt*, el mismo fue actualizado y se encuentra ya *ajustado a las fechas reales* incluyendo las semanas de entrega ya completadas (bajo el flag “Entrega” dentro de la celda).

# Aspectos generales

## *Version control*

A partir de esta entrega, se podrá acceder a todos los files[2] del proyecto en el siguiente repositorio público. Cabe destacar que se aplicaron estrategias de versionados así cómo también '**buenas prácticas'** al *lockear la main branch y trabajar con pull-request*.


https://github.com/anasillo/caylent-hiring

## *Estructura de directorios*

Se dividió el proyecto en folders, correspondientes a cada entrega: Hasta el momento, los mismos son:

- **code**: contiene los archivos del código en sí mismo.
  - 0 - ETL
  - 1 - EDA
  - 2 - E-ML
- **outputs:** contiene los outputs / exports del proyecto
  - **files:** contiene los csv exportados, para una mejor lectura
    - describe
    - head
  - **graph:** contiene los gráficos obtenidos propios de la ejecución del código.
    - aws
    - general
    - k8
    - tf

## *Estructura del archivo 1 - EDA / main.ipynb*

- Packages
  - Downloading packages
  - Importing libraries

[2] Algunos archivos y líneas de código fueron levemente modificados ya que contenían información confidencial. Los originales se encuentran en el drive, bajo una restricción de accesos más apropiada. De la misma manera, varios archivos no fueron incluidos dentro del control de versiones, tales cómo las propias evaluaciones de los candidatos.

    - Reading files and setting variables
- ETL
    - Renaming columns and changing axis order
    - Reindexing columns
    - Re-group of columns for better management
    - ***Adding hired/no hired labels:*** debido a que no se contaba con los datos, es importante destacar que la lógica se realizó fue de asignación de **Hired** si el Overall Score de las evaluaciones era *igual o superior (ge / =>) al 50%.*
- EDA
    - Functions definitions
        - Disclaimer
        - Generate pie chart
        - Generate barchart
        - Generate boxplots
        - Generate missings
        - Generate columns interactions
    - Analyzing information
        - Sample information
        - Counting information
        - Information structure
        - Calculating descriptive statistics
    - ***Functions executions****:* contiene las ejecuciones de los módulos definidos en el enunciado de definición de funciones.

# EDA

## *How we started*

### Starting point

Cómo punto principal, antes de comenzar con la explicación del *Entregable #2*, es importante remarcar cómo llegaron los datos a este estadío. Para esto resulta necesario remontarnos a la **Imagen 1** y a la **Imagen 2** de la Entrega 1, que son los documentos que se utilizaron para realizar la ingesta de datos, previa a un extenso saneamiento de los mismos.

Por supuesto, hay que agregarle los de **K8**, los cuales vienen en tres versiones, por lo que hubo que realizar un mapeo de datos para unificarlos.

| | | | | |
|---|---|---|---|---|
| | | | | KUBERNETES EVALUATION |
| | ARCHITECTURE | | Points | "Which are the main components in a kubernetes architecture? |
| | | VERDADERO | | Masters and Workers nodes |
| | | FALSO | | API server |
| | | FALSO | | kube-scheduler |
| | | FALSO | | kube-controllers |
| | | FALSO | | etcd |
| | | FALSO | | kubelet |
| | | FALSO | | kube-proxy |
| | | FALSO | | container runtime |
| | | FALSO | | CoreDNS |
| | | FALSO | | CNI |
| | | FALSO | | ClusterAutoscaler |
| | | FALSO | | metrics-server |
| | | **Total** | 0 | 1 |
| | VICES | | Points | **What is a kubernetes service? What are the Kubernetes service types and when you would use each one?** |
| | | VERDADERO | | Knows what a service is in Kubernetes |
| | | FALSO | | ClusterIP |
| | | FALSO | | Headless Services |
| | | FALSO | | LoadBalancer |

***Documento 1:*** *Kubernetes versiones 0.1 y 0.2*

**Caylent Structured Interview**

| | |
|---|---|
| **Interview Round** | 2nd Technical |
| **Topic** | **KUBERNETES** |
| **Version** | v0.3 |
| **Candidate** | **CANDIDATE NAME** |
| **Workable Link** | |
| **Score** | 73/100 |

**Interview Questions**

| No. | 1 | Category | CORE CONCEPTS | Points | 18 | | Completeness | 1.00 | SCORE | 18 |
|---|---|---|---|---|---|---|---|---|---|---|

***Can you explain, as best you can, what a Kubernetes Pod is? I would like to understand the relationship between pods and containers as well as all the things I can configure for a Pod.***

| | Predefined responses | | Probing questions | Interviewer comments |
|---|---|---|---|---|
| A | VERDADERO | Explains Pods are a logical wrapper entity for one or more containers. | *How would you define a Kubernetes "Pod"? What is a "Pod", essentially?* | |
| B | VERDADERO | Mentions Pods are the smallest deployable unit that you can create and manage in Kubernetes | *Can I run a standalone container outside a Pod in Kubernetes?* | |
| | | Understands multiple containers can run inside the same Pod. Defines or provides valid examples ***of at least two (2)*** of the following: | | |

***Documento 2:*** *Kubernetes version 0.3*

### Base documents

Una vez recopilados los documentos, para cada una de las entrevistas **(AWS, K8 y TF)** y para cada uno de los candidatos, los mismos fueron mergeados en una única matriz. Aca, resulta importante mencionar que se generó una columna adicional **Overall** la cual contiene la ponderación del resto de las columnas.

El documento que se muestra a continuación, es la *base para todo el proyecto*:

| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Employee | EC2 (2) | IAM (1) | t. App. Reliability | Networking (2) | Overall | S3 Access (0.5) | S3 Arch (0.5) | S3 Classes (0.5) | S3 DR (0.5) | Configuration drif | ount and ForEac | Environment |
| 2 | 01b7a5 | 87.5 | 0.0 | 100.0 | 100.0 | 643.444 | 50.0 | 100.0 | 50.0 | 50.0 | 50.0 | 0.0 | 100.0 |
| 3 | 04bb34 | 100.0 | 40.0 | 100.0 | 100.0 | 78.056 | 50.0 | 100.0 | 100.0 | 100.0 | | | |
| 4 | 0b2569 | 87.5 | 80.0 | 100.0 | 100.0 | 693.444 | 50.0 | 50.0 | 0.0 | 50.0 | 100.0 | 100.0 | 100.0 |
| 5 | 0c59a8 | 87.5 | 40.0 | 25.0 | 87.5 | 55.278 | 100.0 | 0.0 | 100.0 | 50.0 | | | |
| 6 | 0f955f | 100.0 | 80.0 | 87.5 | 100.0 | 83.42 | 100.0 | 100.0 | 50.0 | 100.0 | | | |
| 7 | 119661 | 87.5 | 0.0 | 87.5 | 75.0 | 56.944 | 50.0 | 100.0 | 50.0 | 50.0 | | | |
| 8 | 119e1c | 100.0 | 40.0 | 75.0 | 100.0 | 64.444 | 50.0 | 100.0 | 50.0 | 50.0 | | | |

***Documento 3:*** *Matriz de resultados combinados*

## *Analyzing information*

### Sample information

A continuación se encuentra el resultado del *df.head()*.

Sample of the dataset with employee information:

| | [Candidate] ID | [Candidate] Hired | [Candidate] Overall | [AWS] EC2 | [AWS] S3: Arch | [AWS] S3: Access | [AWS] S3: Clasess | [AWS] S3: DR | [AWS] IAM | [AWS] Networking | ... | [TF] Secrets and States | [TF] Sensitive information | [TF] Best practices |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 01b7a5 | Yes | 64.3444 | 87.5 | 100.0 | 50.0 | 50.0 | 50.0 | 0.0 | 100.0 | ... | 33.33 | 75.0 | 75.0 |
| 1 | 04bb34 | Yes | 78.0560 | 100.0 | 100.0 | 50.0 | 100.0 | 100.0 | 40.0 | 100.0 | ... | NaN | NaN | NaN |
| 2 | 0b2569 | Yes | 69.3444 | 87.5 | 50.0 | 50.0 | 0.0 | 50.0 | 80.0 | 100.0 | ... | 0.00 | 75.0 | 50.0 |
| 3 | 0c59a8 | Yes | 55.2780 | 87.5 | 0.0 | 100.0 | 100.0 | 50.0 | 40.0 | 87.5 | ... | NaN | NaN | NaN |
| 4 | 0f955f | Yes | 83.4200 | 100.0 | 100.0 | 100.0 | 50.0 | 100.0 | 80.0 | 100.0 | ... | NaN | NaN | NaN |

***Imagen 9:*** *df.head() output*

Cómo el output no es óptimo a través de la consola, se optó por enviarlo directo a un csv[3] al mismo tiempo en que se le aplicaba un *df.transpose(),* de esta manera se facilita la lectura al transponer la matriz. El resultado es el siguiente:

| # | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| [Candidate] ID | 01b7a5 | 04bb34 | 0b2569 | 0c59a8 | 0f955f |
| [Candidate] Hired | Yes | Yes | Yes | Yes | Yes |
| [Candidate] Overall | 64.3444 | 78.056 | 69.344 | 55.278 | 83.42 |
| [AWS] EC2 | 87.5 | 100.0 | 87.5 | 87.5 | 100.0 |
| [AWS] S3: Arch | 100.0 | 100.0 | 50.0 | 0.0 | 100.0 |
| [AWS] S3: Access | 50.0 | 50.0 | 50.0 | 100.0 | 100.0 |

[3] Este archivo se encuentra subido al repositorio.

| | | | | | |
|---|---|---|---|---|---|
| [AWS] S3: Classes | 50.0 | 100.0 | 0.0 | 100.0 | 50.0 |
| [AWS] S3: DR | 50.0 | 100.0 | 50.0 | 50.0 | 100.0 |
| [AWS] IAM | 0.0 | 40.0 | 80.0 | 40.0 | 80.0 |
| [AWS] Networking | 100.0 | 100.0 | 100.0 | 87.5 | 100.0 |
| [AWS] Net. App. Reliability | 100.0 | 100.0 | 100.0 | 25.0 | 87.5 |
| [TF] Configuration drifts | 50.0 | | 100.0 | | |
| [TF] Use of count & for_each | 0.0 | | 100.0 | | |
| [TF] Environments | 100.0 | | 100.0 | | |
| [TF] Managing resources | 60.0 | | 80.0 | | |
| [TF] Process | 100.0 | | 100.0 | | |
| [TF] Secrets and States | 33.33 | | 0.0 | | |
| [TF] Sensitive information | 75.0 | | 75.0 | | |
| [TF] Best practices | 75.0 | | 50.0 | | |
| [TF] Managing values | 100.0 | | 80.0 | | |
| [K8] Architecture: Auto Scaling | 50.0 | | 25.0 | | 75.0 |
| [K8] Architecture: Control Plane | 50.0 | | 25.0 | | 75.0 |
| [K8] Core Concepts | 50.0 | | 100.0 | | 100.0 |
| [K8] Services & Networking: Ingress | 50.0 | | 100.0 | | 75.0 |
| [K8] Services & Networking: Service | 100.0 | | 100.0 | | 100.0 |
| [K8] Workload Management | 100.0 | | 100.0 | | 100.0 |

*Tabla 1: df.head().transpose() sample information*

### Counting information

El **dataset** contiene **141 observaciones (filas)** y **26 features (columnas)**. Es importante mencionar que muchos de los archivos se encuentran *"perdidos"*, por lo que si bien es cierto que 141 observaciones son pocas para alimentar el modelo, en un futuro si se consigue acceso a más files, el número podrá incrementarse exponencialmente.

### Information structure

A continuación se encuentra el resultado del *df.info()*. Lo más importante a mencionar es que para las categorías de **AWS no tenemos observaciones nulas** mientras que para las categorías de **Terraform** y **Kubernetes**, nos encontramos con **~100 observaciones nulas** según la tabla a continuación**.**

| # | Column | Non-Null Count | Dtype |
|---|---|---|---|
| 0 | [Candidate] ID | 141 non-null | object |
| 1 | [Candidate] Hired | 141 non-null | object |
| 2 | [Candidate] Overall | 141 non-null | float64 |
| 3 | [AWS] EC2 | 141 non-null | float64 |
| 4 | [AWS] S3: Arch | 141 non-null | float64 |
| 5 | [AWS] S3: Access | 141 non-null | float64 |

| | | | |
|---|---|---|---|
| 6 | [AWS] S3: Classes | 141 non-null | float64 |
| 7 | [AWS] S3: DR | 141 non-null | float64 |
| 8 | [AWS] IAM | 141 non-null | float64 |
| 9 | [AWS] Networking | 141 non-null | float64 |
| 10 | [AWS] Net. App. Reliability | 141 non-null | float64 |
| 11 | [TF] Configuration drifts | 36 non-null | float64 |
| 12 | [TF] Use of count & for_each | 36 non-null | float64 |
| 13 | [TF] Environments | 36 non-null | float64 |
| 14 | [TF] Managing resources | 36 non-null | float64 |
| 15 | [TF] Process | 36 non-null | float64 |
| 16 | [TF] Secrets and States | 36 non-null | float64 |
| 17 | [TF] Sensitive information | 36 non-null | float64 |
| 18 | [TF] Best practices | 36 non-null | float64 |
| 19 | [TF] Managing values | 36 non-null | float64 |
| 20 | [K8] Architecture: Auto Scaling | 37 non-null | float64 |
| 21 | [K8] Architecture: Control Plane | 37 non-null | float64 |
| 22 | [K8] Core Concepts | 37 non-null | float64 |
| 23 | [K8] Services & Networking: Ingress | 37 non-null | float64 |
| 24 | [K8] Services & Networking: Service | 37 non-null | float64 |
| 25 | [K8] Workload Management | 37 non-null | float64 |

***Tabla 2:*** *df.info() information structure*

## Calculating descriptive statistics

A continuación se encuentra el resultado del *df.describe()*.

| | [Candidate] Overall | [AWS] EC2 | [AWS] S3: Arch | [AWS] S3: Access | [AWS] S3: Clasess | [AWS] S3: DR | [AWS] IAM | [AWS] Networking | [AWS] Net. App. Reliability | [TF] Configuration drifts | ... | [TF] Secrets and States | [TF] Sensitive information |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 141.000000 | 141.000000 | 141.000000 | 141.000000 | 141.000000 | 141.000000 | 141.000000 | 141.000000 | 141.000000 | 36.000000 | ... | 36.000000 | 36.000000 |
| mean | 49.123313 | 82.978723 | 67.730496 | 52.127660 | 41.489362 | 45.035461 | 27.517730 | 71.985816 | 62.677305 | 44.444444 | ... | 25.925556 | 29.166667 |
| std | 22.354227 | 22.720968 | 37.860058 | 37.736086 | 38.710209 | 36.500961 | 32.648693 | 33.835893 | 37.350470 | 33.333333 | ... | 30.975282 | 23.528099 |
| min | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | ... | 0.000000 | 0.000000 |
| 25% | 31.250000 | 75.000000 | 50.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 50.000000 | 37.500000 | 0.000000 | ... | 0.000000 | 18.750000 |
| 50% | 51.667000 | 87.500000 | 100.000000 | 50.000000 | 50.000000 | 50.000000 | 20.000000 | 87.500000 | 75.000000 | 50.000000 | ... | 16.665000 | 25.000000 |
| 75% | 65.139000 | 100.000000 | 100.000000 | 100.000000 | 50.000000 | 50.000000 | 60.000000 | 100.000000 | 87.500000 | 50.000000 | ... | 33.330000 | 50.000000 |
| max | 90.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | ... | 100.000000 | 75.000000 |

***Imagen 10:*** *df.describe() output*

Cómo el output no es óptimo a través de la consola, se optó por enviarlo directo a un csv[4] al mismo tiempo en que se le aplicaba un *df.transpose(),* de esta manera se facilita la lectura al transponer la matriz. El resultado es el siguiente:

[4] Este archivo se encuentra subido al repositorio.

| | count | mean | std | min | 25 % | 50 % | 75 % | max |
|---|---|---|---|---|---|---|---|---|
| [Candidate] Overall | 141.0 | 49.123 | 22.354 | 0.0 | 31.25 | 51.667 | 65.139 | 90.0 |
| [AWS] EC2 | 141.0 | 82.978 | 22.720 | 0.0 | 75.0 | 87.5 | 100.0 | 100.0 |
| [AWS] S3: Arch | 141.0 | 67.730 | 37.860 | 0.0 | 50.0 | 100.0 | 100.0 | 100.0 |
| [AWS] S3: Access | 141.0 | 52.127 | 37.736 | 0.0 | 0.0 | 50.0 | 100.0 | 100.0 |
| [AWS] S3: Classes | 141.0 | 41.489 | 38.710 | 0.0 | 0.0 | 50.0 | 50.0 | 100.0 |
| [AWS] S3: DR | 141.0 | 45.035 | 36.500 | 0.0 | 0.0 | 50.0 | 50.0 | 100.0 |
| [AWS] IAM | 141.0 | 27.517 | 32.648 | 0.0 | 0.0 | 20.0 | 60.0 | 100.0 |
| [AWS] Networking | 141.0 | 71.985 | 33.835 | 0.0 | 50.0 | 87.5 | 100.0 | 100.0 |
| [AWS] Net. App. Reliability | 141.0 | 62.677 | 37.350 | 0.0 | 37.5 | 75.0 | 87.5 | 100.0 |
| [TF] Configuration drifts | 36.0 | 44.444 | 33.333 | 0.0 | 0.0 | 50.0 | 50.0 | 100.0 |
| [TF] Use of count & for_each | 36.0 | 29.166 | 29.580 | 0.0 | 0.0 | 25.0 | 50.0 | 100.0 |
| [TF] Environments | 36.0 | 76.388 | 33.243 | 0.0 | 50.0 | 100.0 | 100.0 | 100.0 |
| [TF] Managing resources | 36.0 | 68.333 | 27.202 | 0.0 | 60.0 | 80.0 | 80.0 | 100.0 |
| [TF] Process | 36.0 | 76.388 | 36.812 | 0.0 | 50.0 | 100.0 | 100.0 | 100.0 |
| [TF] Secrets and States | 36.0 | 25.925 | 30.975 | 0.0 | 0.0 | 16.665 | 33.33 | 100.0 |
| [TF] Sensitive information | 36.0 | 29.166 | 23.528 | 0.0 | 18.75 | 25.0 | 50.0 | 75.0 |
| [TF] Best practices | 36.0 | 49.305 | 27.701 | 0.0 | 25.0 | 50.0 | 75.0 | 100.0 |
| [TF] Managing values | 36.0 | 59.444 | 32.244 | 0.0 | 40.0 | 60.0 | 80.0 | 100.0 |
| [K8] Architecture: Auto Scaling | 37.0 | 31.351 | 34.754 | 0.0 | 0.0 | 25.0 | 50.0 | 100.0 |
| [K8] Architecture: Control Plane | 37.0 | 22.972 | 33.007 | 0.0 | 0.0 | 0.0 | 50.0 | 100.0 |
| [K8] Core Concepts | 37.0 | 54.922 | 35.757 | 0.0 | 50.0 | 50.0 | 100.0 | 100.0 |
| [K8] Services & Networking: Ingress | 37.0 | 35.810 | 38.43 | 0.0 | 0.0 | 50.0 | 75.0 | 100.0 |
| [K8] Services & Networking: Service | 37.0 | 73.455 | 61.431 | 0.0 | 0.0 | 50.0 | 75.0 | 75.0 |
| [K8] Workload Management | 37.0 | 57.915 | 52.103 | 0.0 | 0.0 | 25.0 | 100.0 | 100.0 |

***Tabla 3:*** *df.describe().transpose() sample information*

## *Label principal*

Cómo se mencionó ya previamente, antes de comenzar con el EDA se realizó un proceso de ETL necesario para cumplir con los objetivos planteados para esta entrega. De los ya mencionados en la sección *Estructura del archivo 1 - EDA / main.ipynb,* quizás el más relevante para volver a destacar fue el de la condición de **Hired = Yes / No** en base al **threshold del 50%**, ya que no se contaban con los datos reales.

Si bien uno podría considerar que *dos gráficos distintos que muestran lo mismo no son necesarios*, en este caso se pueden obtener dos insights distintos de cada uno de ellos. Pero lo más importante es que podemos entender que las **clases están balanceadas**, lo cual es un punto a favor ya que nos exime de tener que aplicar técnicas cómo **SMOTE**[5] o similares.

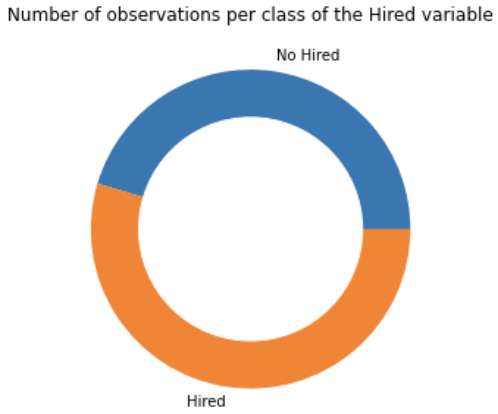



***Imagen 11:*** *Pie chart for Hired*

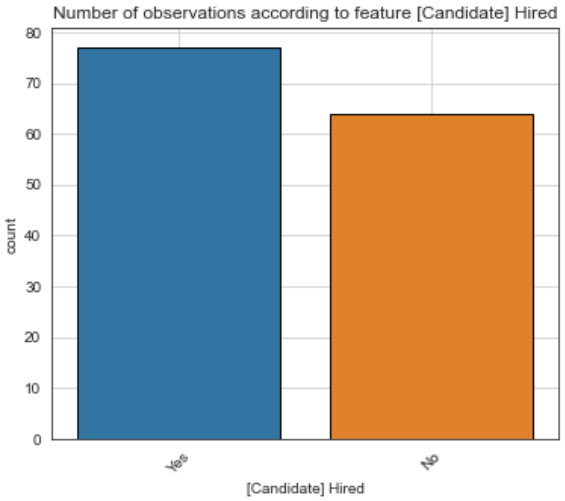



***Imagen 12:*** *Bar chart for Hired*

## *Charts de cada categoría*

### AWS

Respecto de los charts obtenidos de la categoría de **AWS**, las variaciones entre los gráficos son amplias. Podemos observar que algunos de ellos sostienen **distribuciones normales**, mientras que otras sostienen distribuciones que podrían asemejarse a una **exponencial** o **exponencial decreciente**.

[5]La aplicación de la técnica de balanceo de clases **SMOTE** (Synthetic Minority Oversampling Technique), incrementa en forma "sintética" los ejemplos de la clase minoritaria mediante algoritmos de interpolación.

Sin embargo, lo más importante a destacar es que se empleó una técnica de agrupamiento por **cuartiles**[6], ya que se entiende que una *diferencia de 44 a 46 puntos*, por poner un ejemplo, era **matemáticamente absurda** y se debe plenamente a una **conversión aritmética** similar que *no debía ser tratada de esta forma*.

Mientras que a continuación analizaremos los gráficos recién mencionados, las distribuciones reales sin modificar se encuentran en el Anexo (***Imagen Anexo 2.1 a 2.8)***, para aquel interesado que desee compararlas.

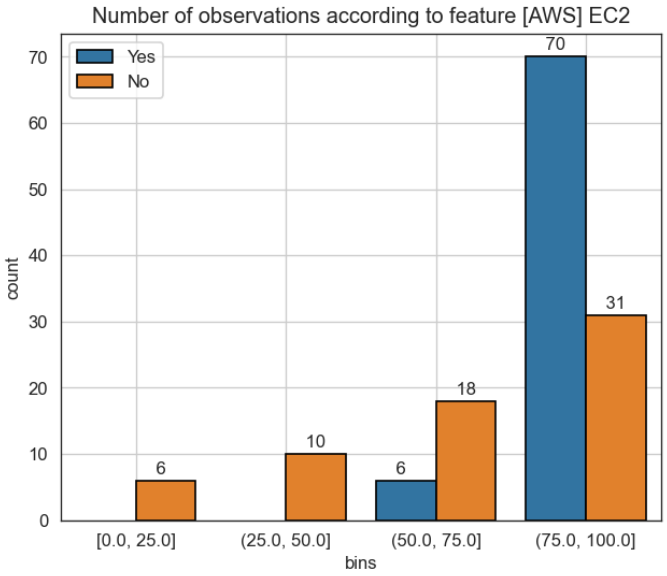



***Imagen 13:*** *AWS EC2*

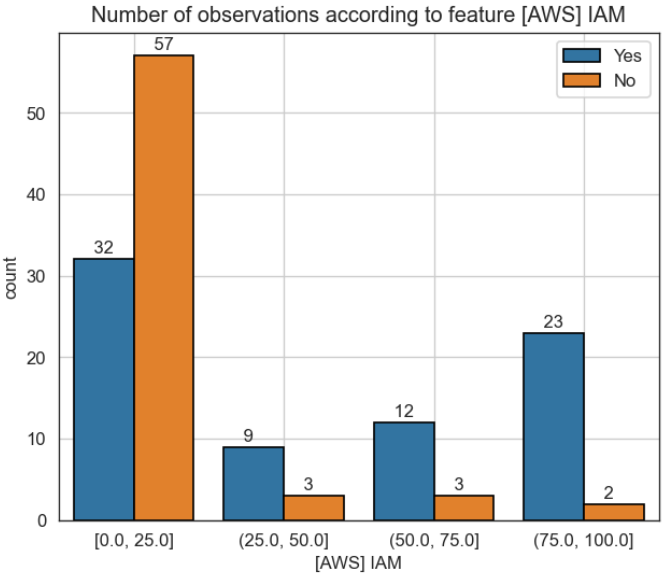



***Imagen 14:*** *AWS IAM*

Para **EC2**, los resultados parecen ser bastante consistentes, si el resultado obtenido es igual o menor al **50%**[7]**,** el candidato es descartado y no avanza en el proceso de selección. Sin embargo, a medida que avanzamos en la escala observamos que incluso se han descartado candidatos con la **puntuación máxima (100)**, por lo que pareciera que estamos ante una *variable decisiva para los niveles bajos*, no obstante, es indistinta una vez superado el umbral mencionado.

[6] Los cuartiles son valores que dividen una muestra de datos en cuatro partes iguales. De esta forma, se puede evaluar rápidamente la dispersión y la tendencia central de un conjunto de datos, que son los pasos iniciales importantes para comprender sus datos. 25% de los datos es menor que o igual a este valor.

[7] La variable de corte se daba en el **62.5%** para ser exactos, pero cómo se ha mencionado antes, se ha agrupado por cuartiles y eso hace que el target caiga dentro del grupo **(50, 75]**

Consecuentemente, se produce el mismo fenómeno para **IAM** pero a la inversa, ya que el **puntaje cero (0)** es quien concentra la mayor cantidad de candidatos, con *32 contratados y casi 57 para no contratados*. De esta manera, se entiende que *no importa si no obtiene buenas calificaciones* en este ítem, por lo que podríamos concluir que *no es una variable decisiva*.

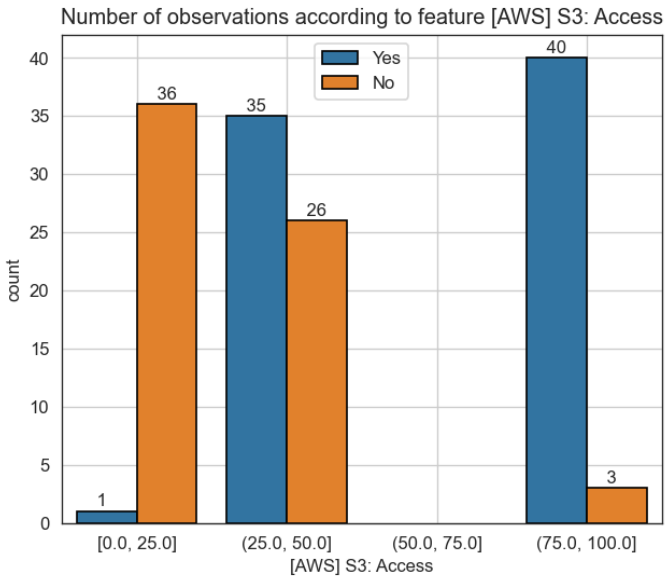



***Imagen 15:*** *AWS S3: Access*

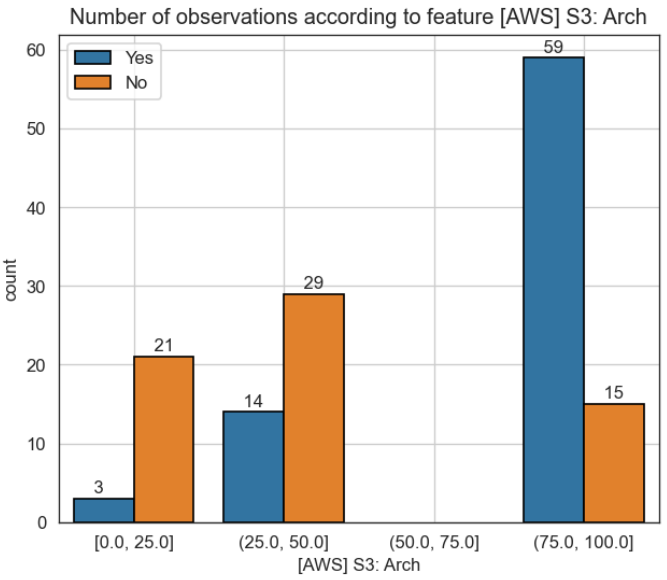



***Imagen 16:*** *AWS S3: Architecture*

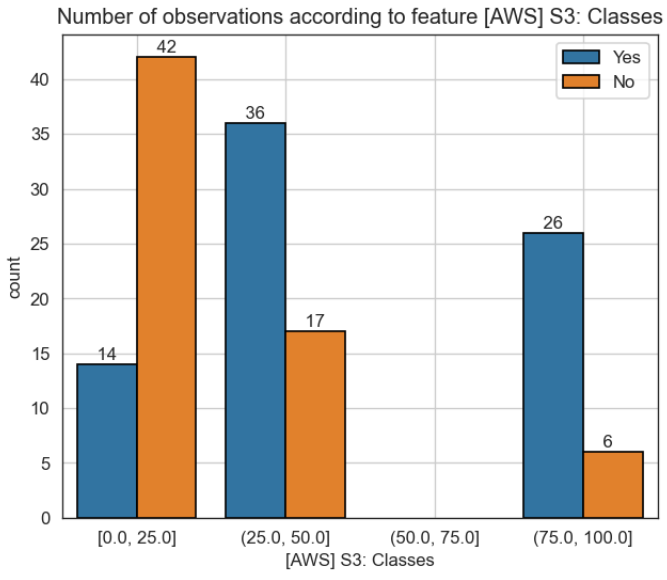



***Imagen 17:*** *AWS S3: Classes*

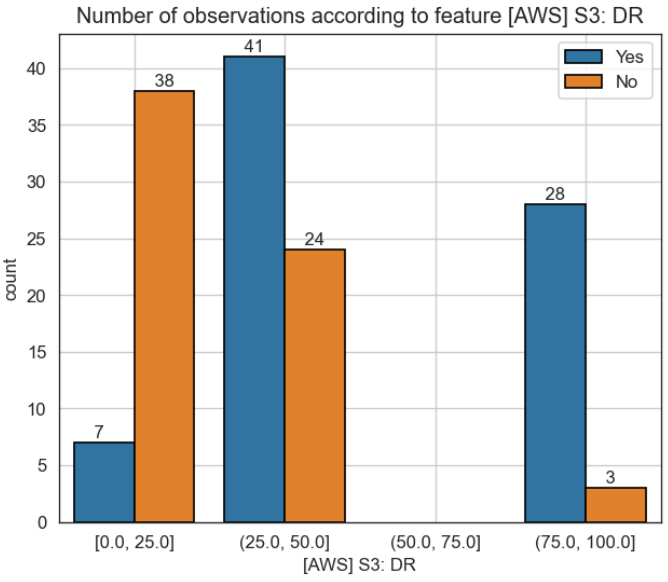



***Imagen 18:*** *AWS S3: DR*

Para los resultados obtenidos en **S3**, lo más destacable es que no existen candidatos sobre el umbral **(50, 75]**. También, resulta importante destacar que para los casos de **S3 Access** y **S3 Architecture**, podemos inferir que se sigue una distribución lógica: a medida que mayor puntaje se obtiene mayores son las chances de ser contratado. Particularmente, donde más se ve esto reflejado es en **S3 Access**, ya que *prácticamente no contiene contratados (a excepción de sólo uno)* en **puntaje cero (0)** mientras al mismo tiempo que *prácticamente todos los que obtienen un* **puntaje máximo (100)** *son contratados (a excepción de tres),* por tal motivo podríamos afirmar que estamos ante otra *variable decisiva*.

Respecto a los casos de **S3 Classes y S3 DR**, podemos inferir que la clase **No Hired** sigue una distribución exponencial negativa, mientras que la clase **Hired** se basa en una distribución normal (*probablemente con* $\mu = 50.0$ *y* $\sigma \varepsilon (0, 1)$, se debe calcular).

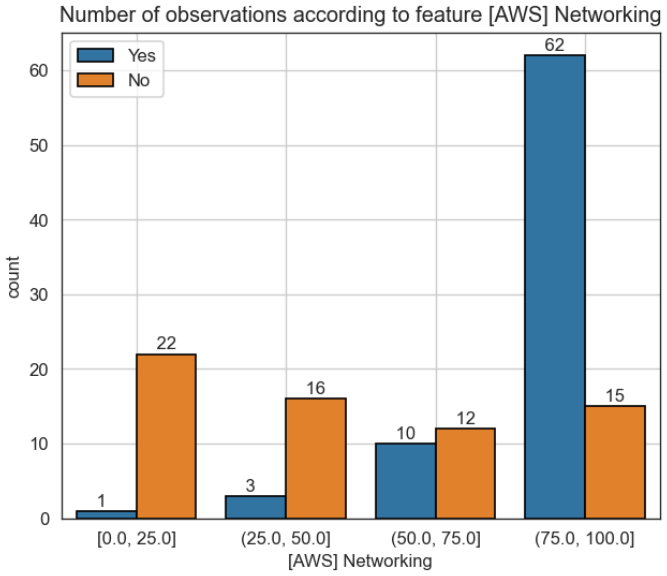



***Imagen 19:*** *AWS Networking*

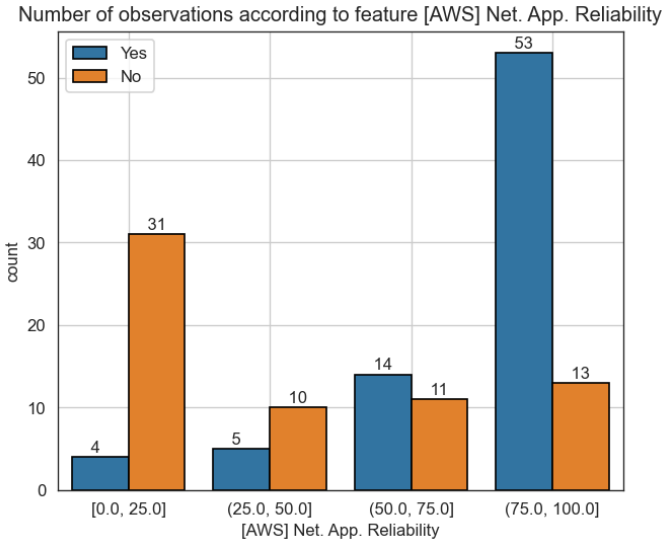



***Imagen 20:*** *AWS Networking Application Reliability*

Finalmente, para los casos de **AWS Networking** y **AWS Networking Application Reliability** observamos similitudes en las distribuciones de ambos casos, para ambos labels. Lo más sorprendente resulta ver que existe una concentración de **Hired** en el último grupo, sin embargo, hasta el momento, *no podemos considerar que sean variables decisivas* respecto a las distribuciones que mantienen.

### Kubernetes y Terraform

Para los charts de las categorías de **Kubernetes** y **Terraform**, empezamos a observar distribuciones menos uniformes ya que ahora nos encontramos en presencia de varios **Missings fields**, por lo que se dificulta realmente modelar una función que se ajuste a los valores en estos casos. Habiendo dicho esto, y teniendo en cuenta que *los gráficos aportan confusión en lugar de traer valor al análisis*, solamente a continuación veremos **dos** de ellos, a modo ilustrativo para entender la dispersión de los valores. El resto de los gráficos se encuentran en el Anexo (***Imagen Anexo 2.9 a 4.3)***.

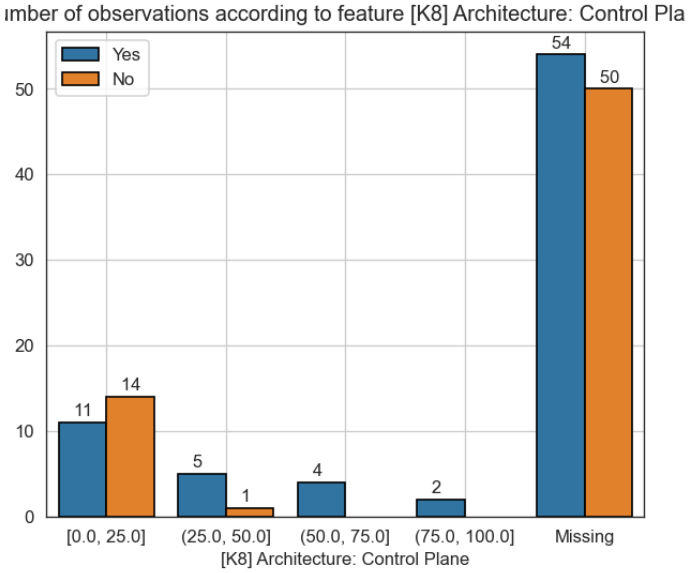



***Imagen 21:*** *K8 Architecture: Control Plane*

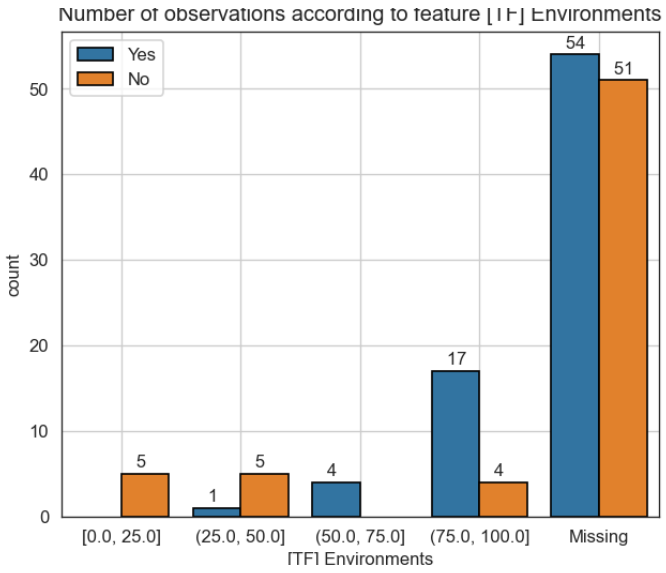



***Imagen 22:*** *TF environments*

Cómo ya se ha mencionado antes, la cantidad de Missings ante la que nos encontramos impide generar insights significativos, sin embargo, resulta curioso el ver que para el caso de **K8 Architecture: Control Plane** no se encuentran candidatos descartados una vez superado el *umbral del 50.0%*, mientras que para **TF environments** sucede exactamente lo opuesto y no se contratan candidatos que no consigan al menos alcanzar el *nivel del 50.0%.*

No obstante, más allá del análisis realizado, y por más que podamos encontrar tendencias y patrones, para ninguno de los casos pareciera ser una *variable decisiva.*

## *Charts generales*

### Boxplots

Tomando ventaja de que se contaban con la mayoría de features numéricas, se graficaron **boxplots** para representar gráficamente características principales de los datos

(posición, dispersión, asimetría, …) e identificar la presencia de valores atípicos. En este caso, se pueden observar los valores de la muestra **sin normalizar** (*arriba*) y los valores **normalizados** (*debajo*). Cómo se puede observar a continuación, en general, **no se observan outliers ni valores atípicos**, ya que los *rangos van de 0 (mínimo score) a 100 (máximo score).*

Sin embargo, hay algunas **excepciones** cómo *[AWS] EC2, [TF] Managing resources y [TF] Secrets and States* que presentan algunos outliers de lo más alejados. No obstante, estamos hablando de evaluaciones técnicas y **no sería coherente removerlos**, ya que tanto el *cero cómo el cien son valores válidos* dentro del proceso.

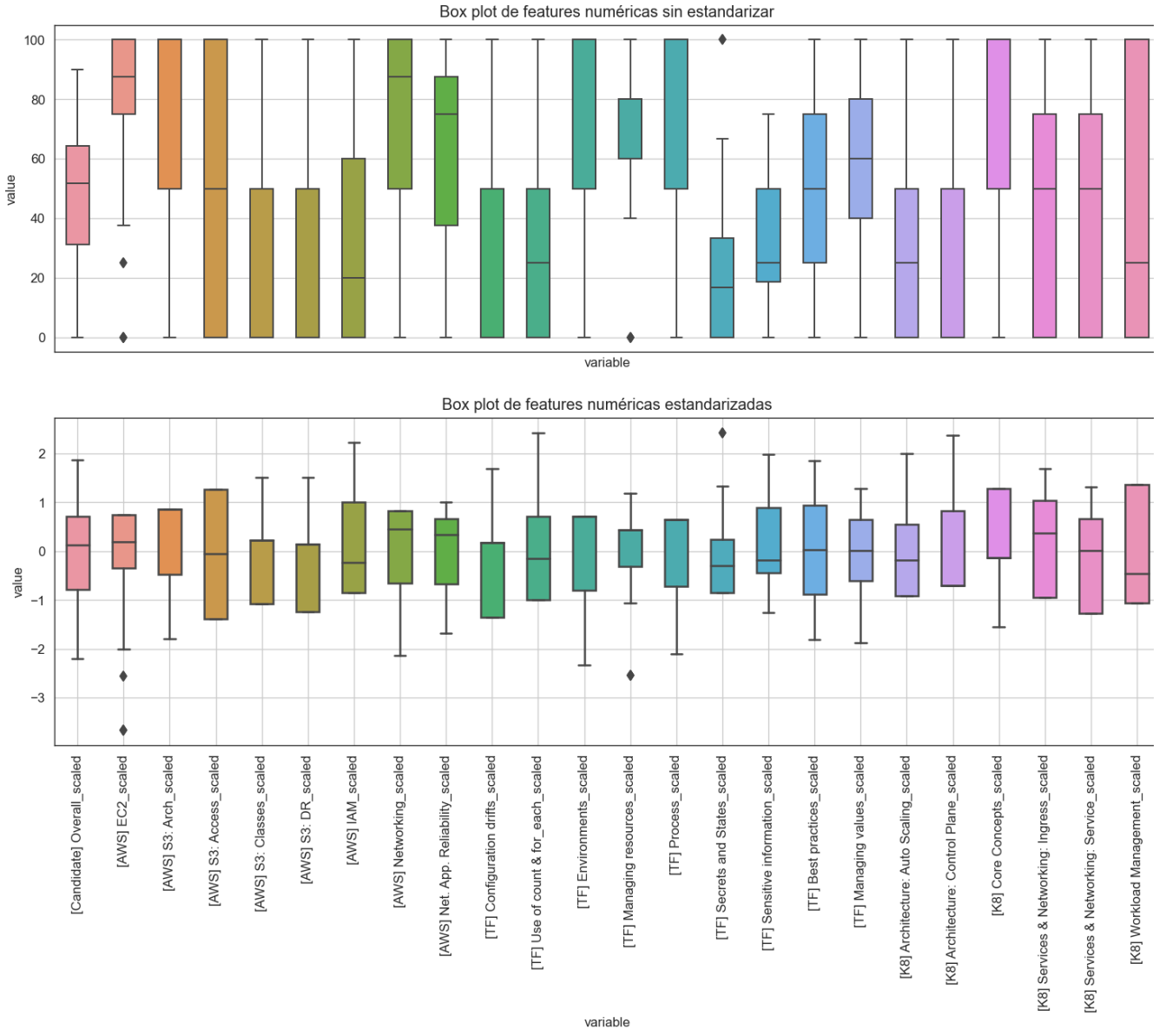



***Imagen 23:*** *Boxplots para features numéricos*

## Missing map

Respecto del missing map, la imagen no hace más que reforzar lo que ya se venía hablando anteriormente: el *dataset posee una cantidad de nulls / missings bastante grande*, a excepción de las categorías de **AWS**, claro está, las cuales no presentan ningún valor ausente.

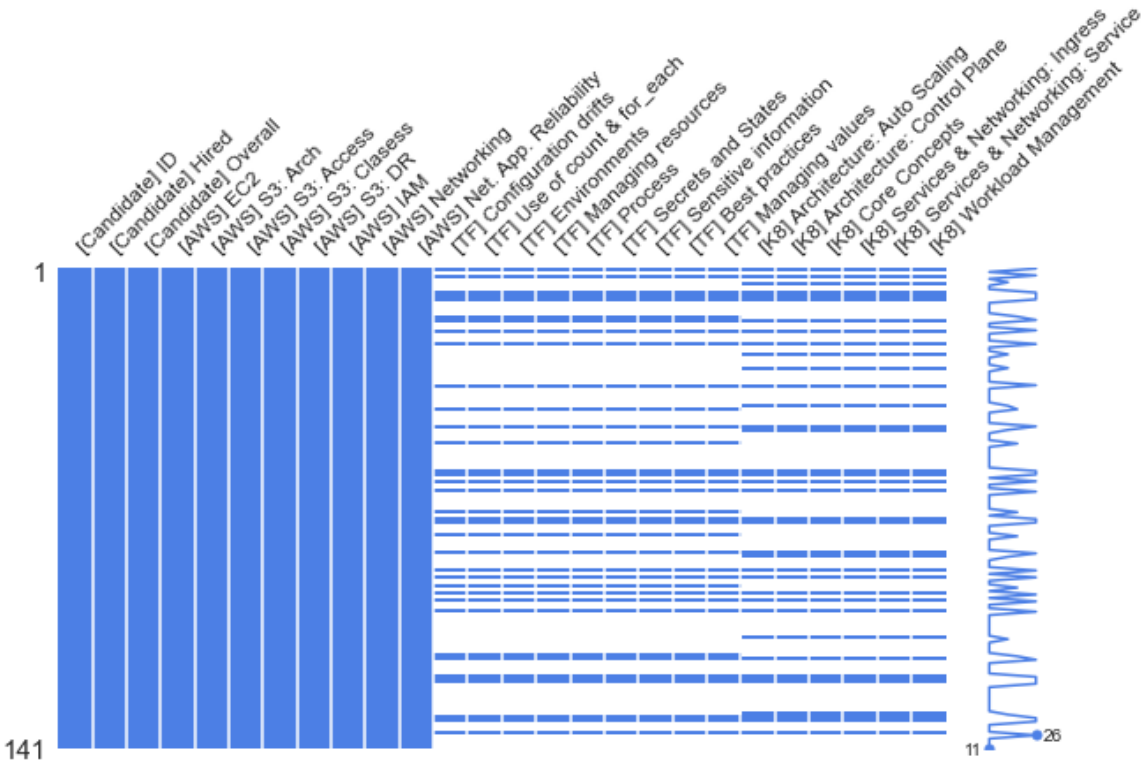



***Imagen 24:*** *Missing map*

## Correlation analysis

Respecto del análisis de correlación, se aplicó *heatmap* para validar las relaciones entre variables y así poder entender si algún feature se encontraba **correlacionado** con otro.

Se probaron distintas técnicas de validación de correlación, tales cómo las fórmulas de **Pearson**, **Kendall**, **Spearman**, siendo esta última la que se muestra a continuación. Dado que los resultados fueron muy similares, las dos primeras se encuentran en el Anexo.

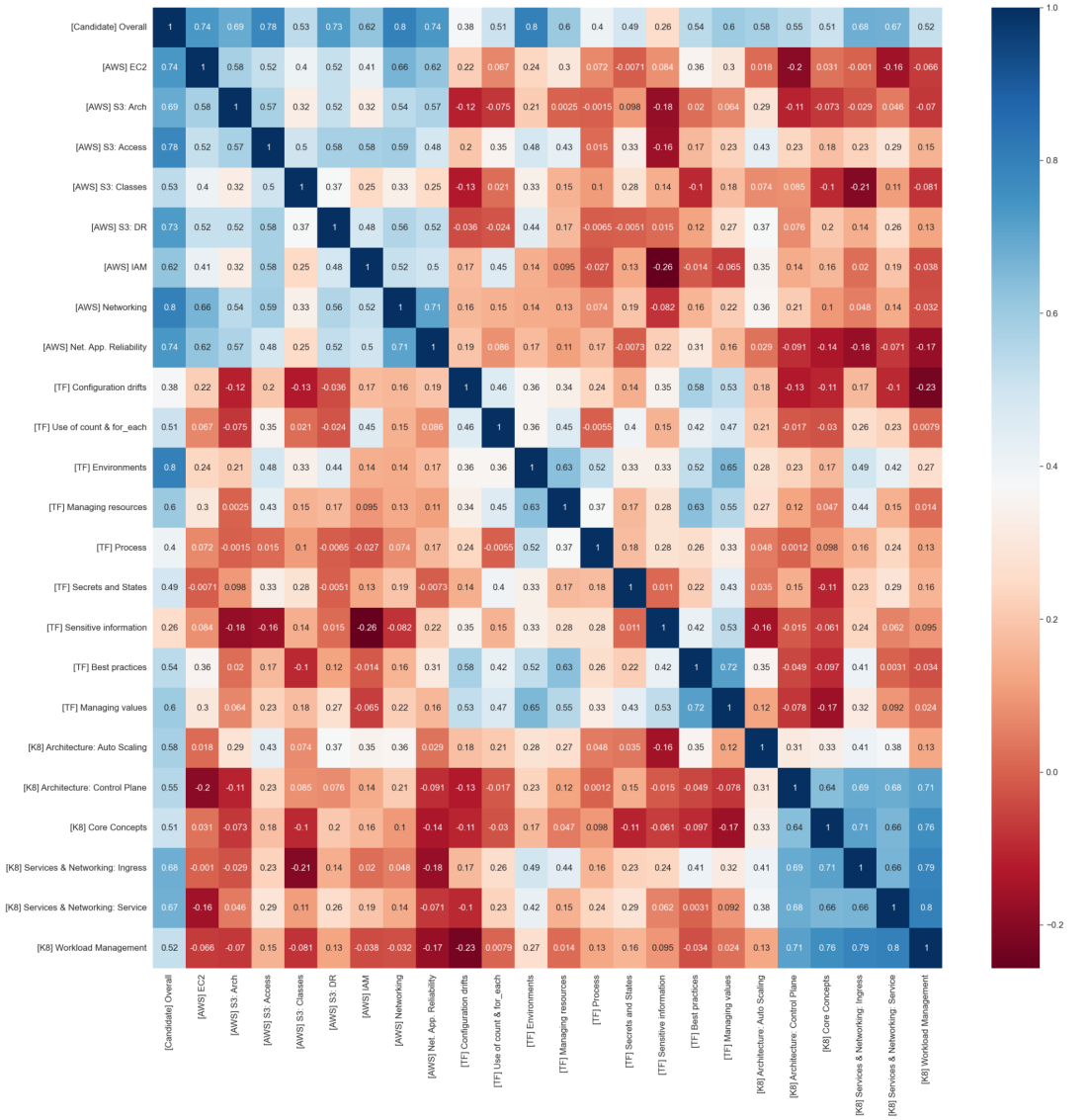



***Imagen 25:*** *Spearman correlation analysis*

Sin embargo, sí resulta importante mencionar que hay **algunos pares de features** que el mapa muestra una **leve o mediana correlación**, a continuación se enumeran los más relevantes a criterio del redactor de este informe:

El **primero**, el par de *[K8] Workload Management y [TF] Configuration drifts*. No podemos afirmar que esto es meramente una *coincidencia* ya que esta **correlación no tiene causalidad** alguna al tratarse de categorías diferentes y preguntas totalmente no-relaciones entre sí.

El **segundo**, el par de *[AWS] IAM y [TF] Sensitive information* sí podríamos afirmar que existe una **correlación lógica** (mas no tanto una **causalidad**)*, ya que ambos ítems refieren al manejo de usuarios e información confidencial (*gestión de usuarios, grupos, etc para

[AWS] IAM; y manejo de secretos, encriptación y demás para el caso de [TF]). Nuevamente, *si bien no es coincidencia, considerarlos correlacionados sería un error*.

El **tercero**, el bloque de las seis preguntas de *[K8]* interrelacionadas entre sí, tiene sentido que se encuentren correlacionadas al tratarse de un mismo tema. Sin embargo, son *preguntas que no tienen punto de contacto entre sí y* la correlación quizás pueda estar dada por una cuestión más binaria (**el candidato conoce o simplemente no conoce del tema**), sin contar con grises o puntos intermedios, dada la **complejidad** del tema.

El **cuarto**, el bloque de las ocho preguntas de *[AWS]* interrelacionadas entre sí, *aunque menos que en el punto tres,* también tiene sentido que se encuentren correlacionadas al tratarse de un mismo tema. Sin embargo en este caso, y a diferencia de K8, las preguntas (y sobre todo, los *exámenes de certificación oficial*) de **AWS** suelen ser **integradoras** por lo que es común tener que saber de un tema para operar en otro.

El **quinto**, el bloque de las nueve preguntas de *[TF]* interrelacionadas entre sí, *aunque en muchísima menor escala que los puntos tres y cuatro,* también tiene sentido que se encuentren correlacionadas al tratarse de un mismo tema. **Terraform** *(HCL)* es un **lenguaje** de alto nivel que permite definir **infraestructura cómo código**, por lo que desde un punto de vista metodológico, se aprende o simplemente no se conoce de él. En este sentido, podemos encontrar que es similar a *Kubernetes* pero en una escala de menor dificultad.

**Feature interaction**

Respecto a la interacción de features entre sí, lo primero a mencionar es que la imagen se encuentra *truncada por motivos de espacio y visualización*. En esta ocasión, sólo haremos mención a **cuatro interacciones** *(las cuatro primeras)*. Para ver la completa (los 26 features), se recomienda acceder a través del repositorio.

Cómo se puede observar en la imagen, el **label verde** hace referencia a **Hired = Yes** mientras que el **label naranja** hace referencia a **Hired = No.** Entendiendo esto, se pueden observar las relaciones entre features con centro en nuestro label principal (Hired).

Cómo regla general, se respeta la lógica. Si miramos las **distribuciones** podemos claramente observar que *a mayor puntaje, mayores son las chances de ser contratado*. Sin embargo, este punto se pone en duda en *[AWS] S3 Access* y se cuestiona en *[AWS] S3 Classes.*

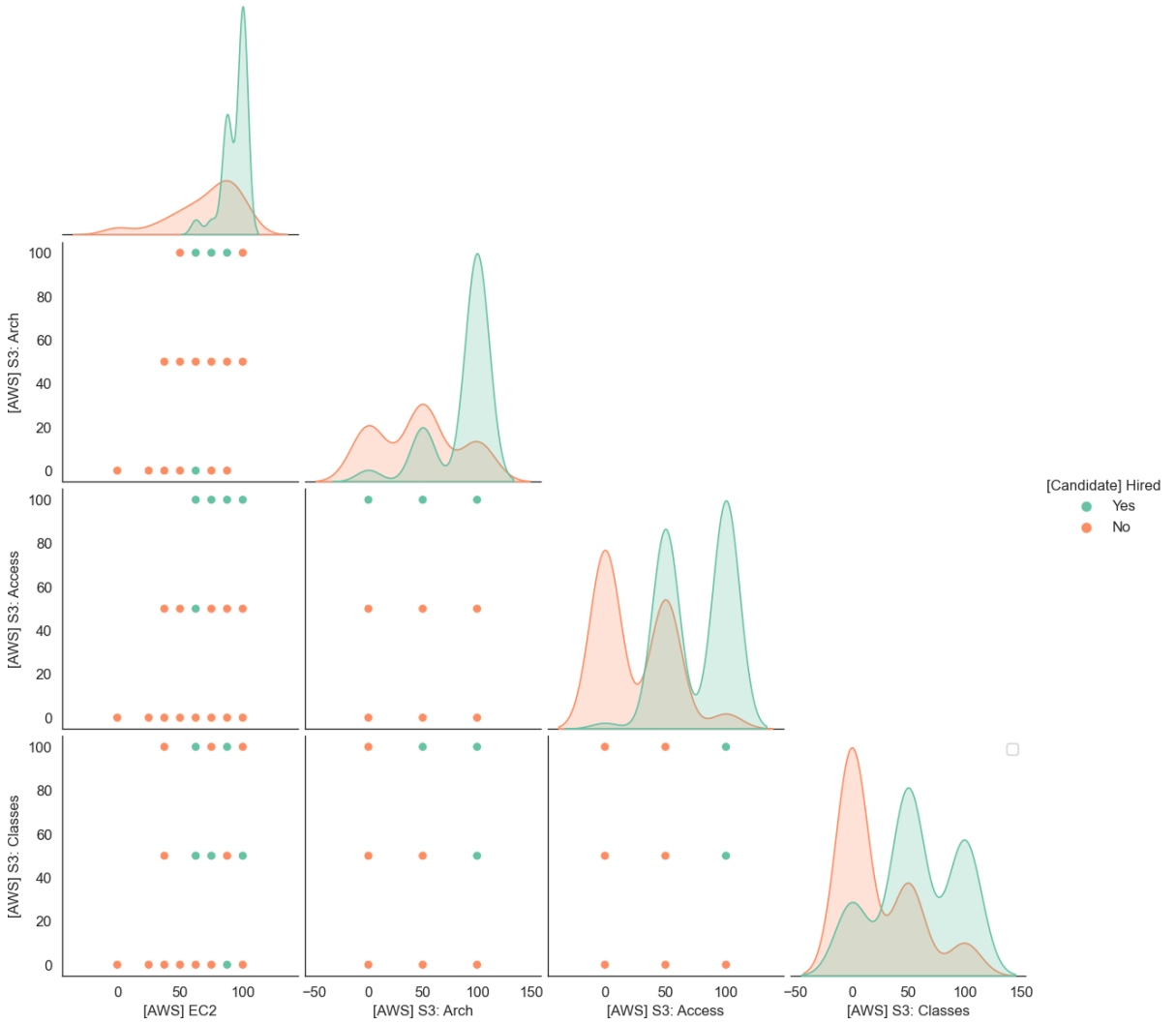



***Imagen 26:*** *Columns and features interactions*

Respecto de los cruces entre features, a continuación se detallan algunas conclusiones que resultan de interés:

- Para *[AWS] EC2* y *[AWS] S3 Arch*, los **Hired** se ubican en la mejor posición **(100%)** de *[AWS] EC2* mientras que para [AWS] *S3 Arch* solo necesitan estar por encima del **50%** aproximadamente. Solo hay una excepción en el cero **(0%)** de *[AWS] EC2*.
- Para *[AWS] EC2* y *[AWS] S3 Access*, se repite el patrón descrito anteriormente con la modificación de que la excepción se encuentra en el umbral medio **(50%).**
- Para *[AWS] EC2* y *[AWS] S3 Classes,* ya la relación se distorsiona y no hay una relación clara que se pueda identificar.
- No parece haber patrones claros para las combinaciones de *[AWS] S3 Arch, [AWS] S3 Access y [AWS] S3 Classes,* aunque suelen situarse en los extremos superiores y derechos. Esto nos indica que posiblemente las variables estén más relacionadas al tratarse de un mismo tema.

# *Entrega 3*

## Correcciones realizadas

Se agregaron los items How we started y Base documents para complementar sobre lo que no se había hablado respecto al desarrollo ETL previo del proyecto.

También se corrigieron los gráficos, explicaron en busca de generar insights. y finalmente se removieron los sobrantes, enviándolos al Anexo.

Se agregaron las métricas para el Random Forest, y se explicaron algunas decisiones, cómo la elección del modelo KNN.

Actualizado Project Charter con las fechas.

# Aspectos generales

## *Estructura del archivo 2 - E-ML / main.ipynb*

- Packages
  - → Downloading packages
  - → Importing libraries
  - → Reading files and setting variables
- Experiments Tracking
  - → Functions definitions
    - * Disclaimer
    - * Log MLFlow experiment
    - * Calculate performance
    - * Splitting feature
  - → Model definitions:
    - * Feature importances
    - * Plot Feature importances
    - * Plot Decision tree
    - * Grid Search Random Forest
    - * Grid Search KNN
    - * Experiment N°1: The naive model
    - * Experiment N°2: The delusional model
    - * Experiment N°3: KNN model with only numerical features
    - * Experiment N°4: Xgboost model
    - * Experiment N°5: Random Forest
  - → ***Functions executions****:* contiene las ejecuciones de los módulos definidos en el enunciado de definición de funciones.
  - → ***Model executions****:* contiene las ejecuciones de los módulos definidos en el enunciado de definición de módulos.

# Experiment tracking

## *General*

Antes de continuar con el análisis, resulta importante entender que el problema debe ser atacado mediante una metodología sistémica, la cual se describe a continuación.

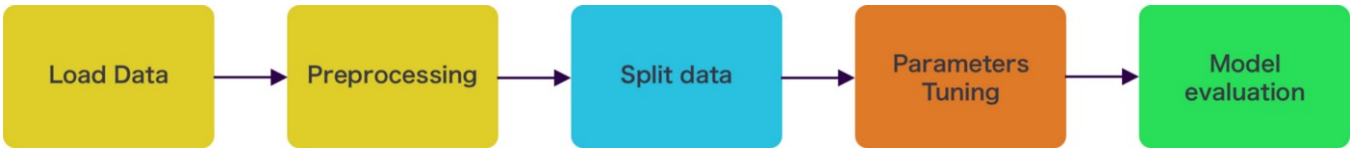



***Imagen 27:*** *flujo del proceso a desarrollar para cada modelo*

Particularmente para cada etapa, se realizarán las siguientes acciones (si aplican):

- **Load data:** lectura del archivo csv en un pandas dataframe.
- **Preprocessing:** eliminación de las columnas innecesarias, saneamiento de datos y reemplazo de datos missing o incompletos.
- **Split data:** dado el dataset, se generará el conjunto de entrenamiento y de testeo.
- **Parameters tuning:** mediante un **GridSearch**, se aplicará una combinación de *hyperparameters,* a fin de obtener los parámetros óptimos para el **Decision Tree, el Random Forest y el KNN**, seguido de un cross-validation con **kFolds = 10** para observar el comportamiento del modelo con los parámetros óptimos obtenidos.
- **Model evaluation:** dados los parámetros óptimos obtenidos, el modelo se probará utilizando los datos de prueba para calcular la precisión.

Analizando cada etapa en particular, y entendiendo que se aplica una metodología recursiva *(es decir, que el proceso es iterativo)*, el flujo completo queda determinado de la siguiente manera:

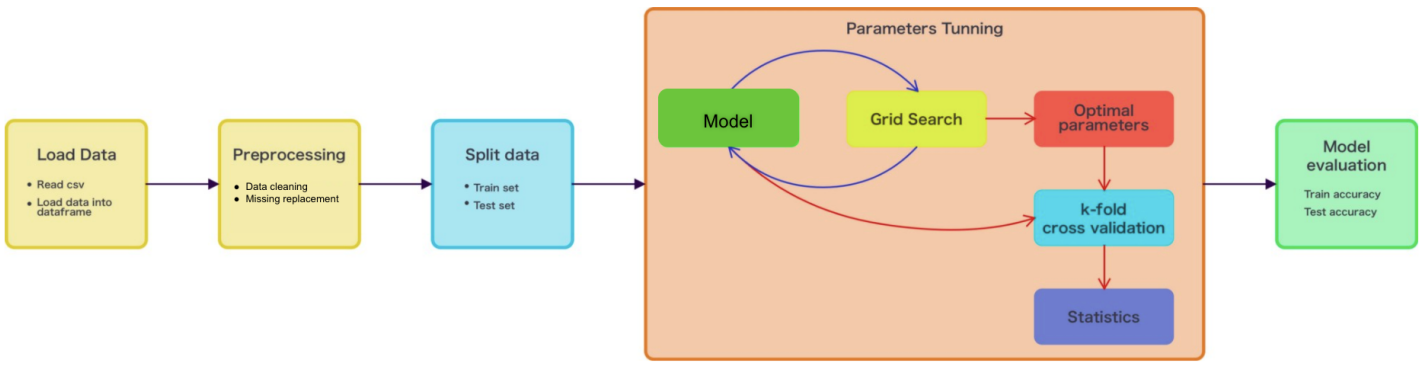



***Imagen 28:*** *flujo del proceso a desarrollar para Decision Tree y KNN.*

## Label principal

Naturalmente, se contaba con todo el pre-procesamiento realizado para las entregas previas, sea ya esto desde la carga de datos, la limpieza de los mismos, así cómo también el proceso de **ETL** realizado en la *Entrega 2 correspondiente al EDA el cual incluye la asignación del label mediante una decisión lógica.*

```python
df['Hired'] = ['Yes' if x >= 50 else 'No' for x in df['[Candidate] Overall']]
df.insert(loc = 1, column = '[Candidate] Hired', value = df['Hired'])
df.drop(['Hired'], axis=1, inplace=True)
```

✓ 0.3s Python

***Código 1:*** *definición del label principal*

## Missing values

Consecuentemente, aquellos valores que se encontraban incompletos *(***Nulls, NaN***) fueron rellenados con cero (0),* ya que **no otorgar puntaje en caso de merecerlo** *parecía una decisión más lógica que* **otorgar puntaje alguno en caso de no merecerlo**.

```python
def preparing_dataset (df, columns = [df.columns[0], df.columns[2]], feature = df.columns[1]):
    # Removing employee identification features
    df.drop(columns=columns, inplace=True)

    # Filling null values
    df.fillna(0, inplace = True)
    # df.round(decimals = 3)

    # Separating input features and target variable
    y = df[feature]
    X = df.drop(columns=feature)

    # Encoding target variable
    y = LabelEncoder().fit_transform(y)

    # Splitting dataset into train and test sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y)

    return X_train, X_test, y_train, y_test, X, y
```

Python

***Código 2:*** *dropeo del label y obtención de los conjuntos de entrenamiento y de testeo.*

## Saneamiento

Cómo se puede observar previamente en el ***Código 2***, se tomó la decisión de *dropear el feature* **'[Candidate] Overall'** (*df.columns[2]*) ya que el mismo contenía información que se encontraba **correlacionada** con el resto de los features, al ser una variable matemáticamente calculada en base al resto.

Sin embargo, la columna correspondiente al identificador del empleado '**[Candidate] ID'** *(df.columns[1]),* también fue eliminada, ya que no aportaba valor alguno al modelo, sino por el contrario, lo *'ensuciaba'*.

## *Modelos*

Para este proyecto se implementaron varios modelos, a fines de obtener un contraste entre los resultados obtenidos para cada uno de ellos. Los modelos ejecutados fueron los siguientes:

- Naive Bayes classifier
- Delusion
- KNN
- XGBoost
- Random Forest
- Decision Tree

A continuación, una breve descripción de cada modelo y los resultados obtenidos.

### Naive Bayes classifier y Delusion

Los modelos **Naive Bayes** y **Delusion** fueron descartados inmediatamente, debido a su bajísima performance. Si observamos con atención, vemos que los mismos se **encuentran bajo o cerca del umbral del 50%**, es decir, tirar una moneda arrojaría aproximadamente los mismos resultados.

| exp_id | model | metric train_acc | metrics test_acc | metrics test_f1 | metrics train_f1 |
|---|---|---|---|---|---|
| **0** | Naive Model | 0.4553 | 0.4482 | 0.0 | 0.0 |
| **1** | Delusional Model | 0.5446 | 0.5517 | 0.7111 | 0.7052 |

***Tabla 4:*** *resultados para Naive Bayes y Delusion.*

### XGBoost

El modelo **XGBoost** no pudo ser logueado a través del **MLFlow**, ya que al hacerlo mata el *kernel de ipython* (ver Imagen Anexo 1) y es imposible avanzar desde allí. Sin embargo, se encontró una solución alternativa al correrlo desde Google Colab (ver Imagen Anexo 2), con un leve refactor a los labels.

| exp_id | model | metric train_acc | metrics test_acc | metrics test_f1 | metrics train_f1 |
|---|---|---|---|---|---|
| **2** | XGBoost | 0.974767 | 0.947115 | - | - |

***Tabla 5:*** *resultados para XGBoost.*

Por supuesto, si hubiéramos podido correr el **XGBoost correctamente**, seteando corridas ilimitadas, *eventualmente llegaríamos o incluso superarían el umbral del modelo ganador* (del cual se hablará más adelante), ya que en definitiva, no deja de ser una combinación de otros modelos.

**Random Forest y Decision Tree**

El modelo de **Random Forest** y el **Decision Tree**, tal cómo se mencionó anteriormente, fueron testeados con los parámetros óptimos obtenidos tras la corrida del **GridSearch**. Para garantizar una amplitud de posibilidades, se testearon las posibilidades que se describen a continuación. La columna *valor óptimo* indica cuál fue la variante que mejor performó para ese **parámetro**.

| parameter | values | valor óptimo |
|---|---|---|
| bootstrap | True | True |
| n_estimators | [200, 300, 400, 500] | 500 |
| max_features | ['auto', 'sqrt', 'log2'] | auto |
| max_depth | range ( 1, 10, 1) | 6 |
| min_samples_leaf | range ( 1, 10, 1) | 4 |
| min_samples_split | range ( 1, 10, 1) | 8 |
| criterion | ['gini', 'entropy'] | gini |

***Tabla 6:*** *hyperparameter tuning para Random Forest y Decision Tree*

Los resultados arrojados son los siguientes:

| exp_id | model | metric train_acc | metrics test_acc | metrics test_f1 | metrics train_f1 |
|---|---|---|---|---|---|
| 3 | Random Forest | 0.9375 | 0.9310 | 0.9375 | 0.9440 |
| 4 | Decision Tree | 0.8928 | 0.7931 | 0.7999 | 0.9000 |

***Tabla 6.1:*** *resultados para Random Forest y Decision Tree.*

Y los resultados de la matriz de confusión se muestran a continuación:

| exp_id | model | | | True Class | |
|---|---|---|---|---|---|
| | | | | Positive | Negative |
| 3 | Random Forest | Predicted Class | Positive | 12 | 1 |
| | | | Negative | 1 | 15 |
| 4 | Decision Tree | | Positive | 11 | 2 |
| | | | Negative | 4 | 12 |

***Tabla 6.2:*** *Matriz de confusión para Random Forest y Decision Tree.*

Si hacemos un doble click sobre la matriz, y recordando los **KPIs** definidos inicialmente, podemos sacar las siguientes conclusiones:

- Para el **Random Forest**, tanto los **Falsos Positivos** ***(Error Tipo I)*** cómo los **Falsos Negativos** ***(Error Tipo II)*** están empatados en uno (1). Dada esta circunstancia, concluimos que este modelo no permite, a priori, una ponderación según lo establecido en los lineamientos (KPIs) del proyecto.
- Para el **Decision Tree**, los **Falsos Positivos** ***(Error Tipo I)*** son notablemente más altos (el doble) que los **Falsos Negativos** ***(Error Tipo II).*** Si bien el objetivo del Decision Tree es explicar los niveles de corte del modelo, si tomamos en cuenta su baja performance y las predicciones arrojadas, este modelo queda descartado.

**KNN**

Antes de comenzar con el análisis de este ítem, resulta preciso remarcar que *hasta este momento se han descartado todos los modelos* (por diversos motivos ya explicados), a excepción del **Random Forest**. La pregunta que acontece, por consiguiente es, *¿cuál será el modelo que mejor se ajuste a nuestros datos? ¿Será* ***Random Forest*** *o* ***KNN****?*

Habiendo dicho eso, el modelo de **KNN**, tal cómo se mencionó anteriormente, también fue testeado con los parámetros óptimos obtenidos tras la corrida del **GridSearch**. Para garantizar una amplitud de posibilidades, se testearon las posibilidades que se describen a continuación. La columna *valor óptimo* indica cuál fue la variante que mejor performó para ese **parámetro**.

| parameter | values | valor óptimo |
|---|---|---|
| algorithm | ['auto', 'ball_tree', 'kd_tree', 'brute'] | auto |
| leaf_size | range ( 1, 30, 1) | 1 |
| n_neighbors | range ( 1, 30, 1) | 12 |
| p | [1, 2] | 2 |

***Tabla 7:*** *hyperparameter tuning para KNN*

Los resultados arrojados son los siguientes:

| exp_id | model | metric train_acc | metrics test_acc | metrics test_f1 | metrics train_f1 |
|---|---|---|---|---|---|
| **3** | KNN | 0.9017 | 0.9655 | 0.9677 | 0.9133 |

***Tabla 7.1:*** *resultados para KNN.*

Y los resultados de la matriz de confusión se muestran a continuación:

| exp_id | model | | | True Class | |
|---|---|---|---|---|---|
| | | | | Positive | Negative |
| 3 | KNN | Predicted Class | Positive | 13 | 0 |
| | | | Negative | 1 | 15 |

***Tabla 7.2:*** *Matriz de confusión para KNN.*

Si hacemos un doble click sobre la matriz, y recordando los **KPIs** definidos inicialmente, podemos sacar las siguientes conclusiones:

- En primer lugar, las métricas del **KNN** de **accuracy** cómo **F1** son superiores al **Random Forest.** Sin embargo, esto no es suficiente para determinar la validez de un modelo por sobre el otro.
- No obstante, dado que en segundo lugar también se ha conseguido disminuir a cero (0) la cantidad de **Falsos Negativos *(Error Tipo II),*** podemos añadir otro punto a favor de este modelo. Esto significa que *todos los candidatos que deberían ser contratados, son efectivamente contratados.*
- Asimismo, los ***Falsos Positivos (Error Tipo 1),*** se encuentran empatados con el **Random Forest**, en una unidad (1).
- Finalmente, y haciendo eco del último **KPI** definido: ***Cantidad de contratados***, se puede observar que se ha logrado contratar a una mayor cantidad de candidatos. Puede que a futuro se demuestre que el modelo se equivocó, o puede que se refuerce la idea de que no tenía que haber sido contratado. De cualquier manera, **las decisiones del modelo están basadas en su entrenamiento**, y como su entrenamiento parte de una **conducta social humana**, quizás debamos trabajar internamente para revaluar las decisiones tomadas hasta el momento.

Habiendo comparado todos los modelos, se decide que aquel que mejor se ajusta para la resolución de este problema es el **KNN**, por lo expresado anteriormente. No obstante, se dejan abiertas las puertas a probar más posibilidades en un futuro no tan lejano.

Cómo última mención, cabe destacar que los **GridSearch** ejecutados tanto para los modelos de **Random Forest** cómo para **KNN** son amplios, se han tenido que dejar posibilidades afuera. Debido al criterio de **parsimonia**[8]**,** resultó imposible aplicar todas las combinaciones matemáticas, por limitaciones computacionales y también de tiempos.

[8] El criterio de **Parsimonia** se utiliza para determinar si un algoritmo o unidad de medida de simplicidad en un contexto empírico particular es mejor que otro, debido a simplificaciones de uso de recursos computacionales, muestrales, entre otros.

# Feature engineering

Si bien se ha manejado una metodología que permitió arribar a un resultado *'satisfactorio',* la misma no es suficiente ya que carece de carácter explicativo. Por tal motivo, se recurrieron a dos técnicas, detalladas a continuación, para poder entender el comportamiento de los modelos y así poder arribar a mejores insights.

## *Feature Importance*

Cómo primera herramienta, se realizó un **Feature Importance** para entender cuáles eran los features que poseen mayor peso / ponderación dentro del modelo.

Para generar validez al modelo, el Feature fue ejecutado con los parámetros obtenidos del **GridSearch** del **Random Forest,** *que si bien no fue el mejor de los modelos, su performance fue bastante satisfactoria*.

De esta manera, se logró **establecer una trazabilidad** *que garantizaba coherencia durante* el proceso al mismo tiempo que *validaba la ejecución de esta técnica* con **parámetros óptimos**, reforzando el nivel de confianza obtenido por la métrica de accuracy.

Los resultados[9] arrojaron la siguiente tabla. Sin embargo, es importante mencionar que en función de atomizar el espacio, solo se muestran los **diez features más importantes** (aunque si se desea un panorama completo, se puede observar en la ***Imagen 29:*** *feature importances graficados*, que se encuentra debajo).

| # | Feature | Importance |
|---|---|---|
| 1 | [AWS] Networking | 20.07 |
| 2 | [AWS] S3: Access | 13.89 |
| 3 | [AWS] Net. App. Reliability | 12.61 |
| 4 | [AWS] S3: Arch | 12.40 |
| 5 | [AWS] EC2 | 8.50 |
| 6 | [AWS] S3: DR | 6.99 |
| 7 | [AWS] S3: Classes | 6.90 |
| 8 | [AWS] IAM | 6.42 |
| 9 | [K8] Workload Management | 1.33 |
| 10 | [K8] Services & Networking: Service | 1.30 |
| - | **Accuracy** | 0.9310 |

***Tabla 5:*** *Feature Importance.*

[9] Cabe destacar que los resultados variaron levemente corrida a corrida, pero no modificaban la posición dentro del esquema. Esto quiere decir que podíamos obtener valores ligeramente diferentes pero la importancia se mantenía intacta.

El mayor análisis que se puede obtener de esto, es la ponderación asignada a una categoría cómo **Networking**, la cual es transversal a todas las áreas deriva en lo que internamente conocemos cómo **cross-cutting[10] concern**. Si bien es cierto que, podemos afirmar que conocer de *esta área resulta transcendental on a daily basis,* la diferencia que mantiene con el resto de las categorías podría resaltar **cuáles debieran ser las preguntas más importantes** para la toma de decisión durante el proceso.

No obstante, los números tal cómo se presentan podrían no reflejar la variación, por tal motivo se presenta la información completa en la siguiente gráfica. De mayor a menor, los features más importantes para el modelo. Se retoma y valida este análisis en la sección siguiente, de **Decision Tree,** *cuya performance ya fue explicado previamente en la sección de* [*Modelos*](#).

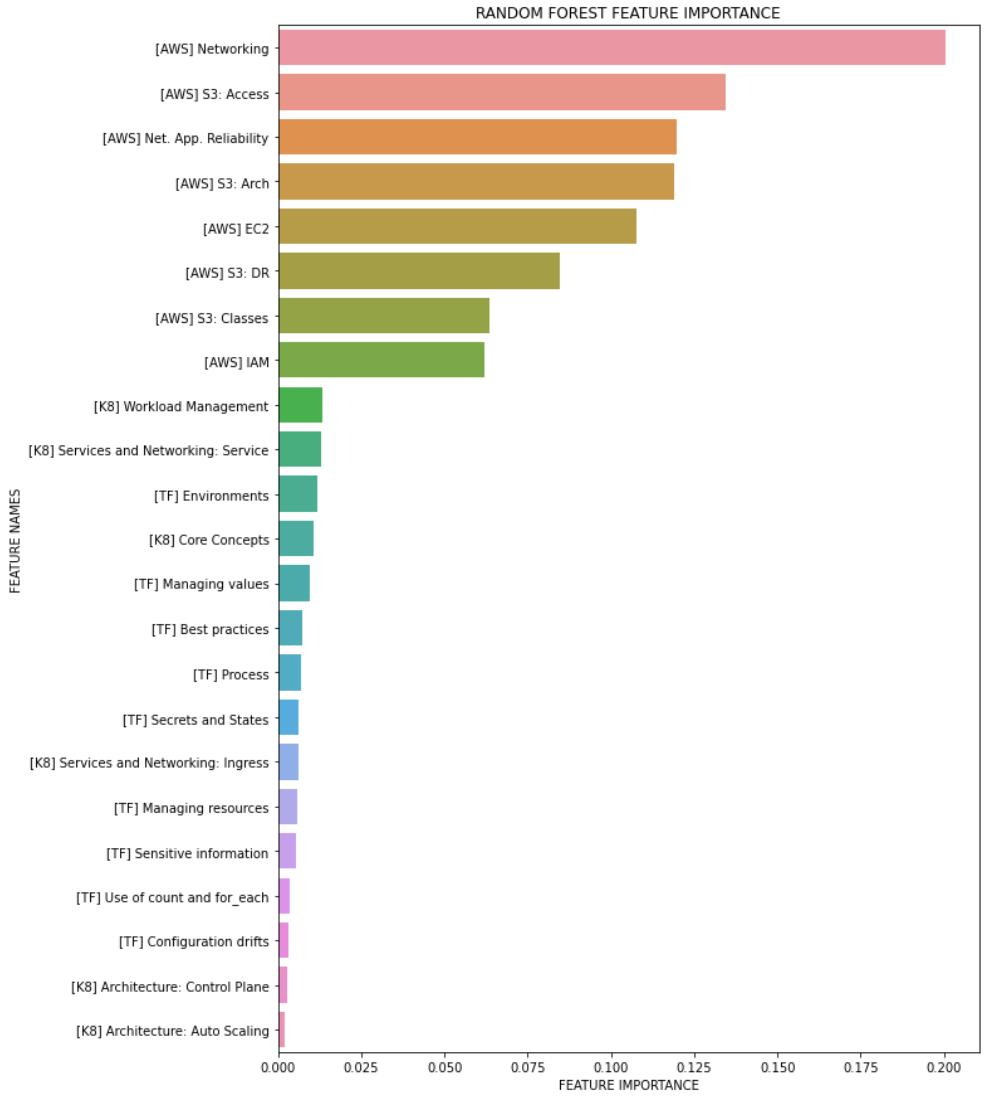



***Imagen 29:*** *Feature Importances graficados.*

[10] Las cross-cutting concerns son partes de un programa que dependen o deben afectar a muchas otras partes del sistema. Forman la base para el desarrollo de software y arquitecturas.

Habiendo gráfico el *Feature Importance*, podemos dividir los features en los siguientes cuatro rangos:

1. El **primer rango**, el mayor, correspondiente a **[AWS] Networking**, es el que mayor **pondera** por una diferencia de más de **seis (6) puntos porcentuales**. Cómo se ha mencionado anteriormente, es la mayor variable a tomar en cuenta.
2. El **segundo** rango, correspondiente a *[AWS] S3 Access, [AWS] Net. App. Reliability, [AWS] S3 Arch y [AWS] EC2*, es un grupo de las categorías que, en el orden en el cual fueron descritas, le siguen la marcha al ítem de **Networking**. Estas, deberán también ser tenidas en cuenta.
3. El **tercer** rango, compuesto por los últimos del podio, consta de *[AWS] S3: DR, [K8] Workload Management y [K8] Services & Networking: Service.* Aquí, la primera gran **sorpresa**: los ítems de **Kubernetes** lograron **equiparar** a los valores de **AWS**, aun estando en notable diferencia de observaciones (rows). Esto fuerza la idea de que para entrar **Caylent**, se necesita saber de **AWS** pero también de **K8**, al menos, en una primera instancia.
4. El **cuarto** y último rango, *la cola del resto de features,* los cuales podrían ser descartados ya que **no son para nada representativos** para el modelo. Esto, dicho de otra manera, *significa que las preguntas que se realizan, no resultan de relevancia para determinar la contratación de un cándido, o no.*

## *Decision Tree*

Consecuentemente, cómo segunda herramienta, se realizó un **Decision Tree** para poder entender cuáles y cómo eran las lógicas de ponderación y corte del modelo. A continuación se describe el correspondiente a **gini**, ya que fue el **hyperparameter** que mejor performó.

Cabe destacar que se realizaron varias iteraciones modificando los niveles y argumentos, incluso utilizando el criterio de **entropy**, pero a fines prácticos los resultados eran similares y/o incorrectos, al no estar validados por el **GridSearch** que ya se ha mencionado.

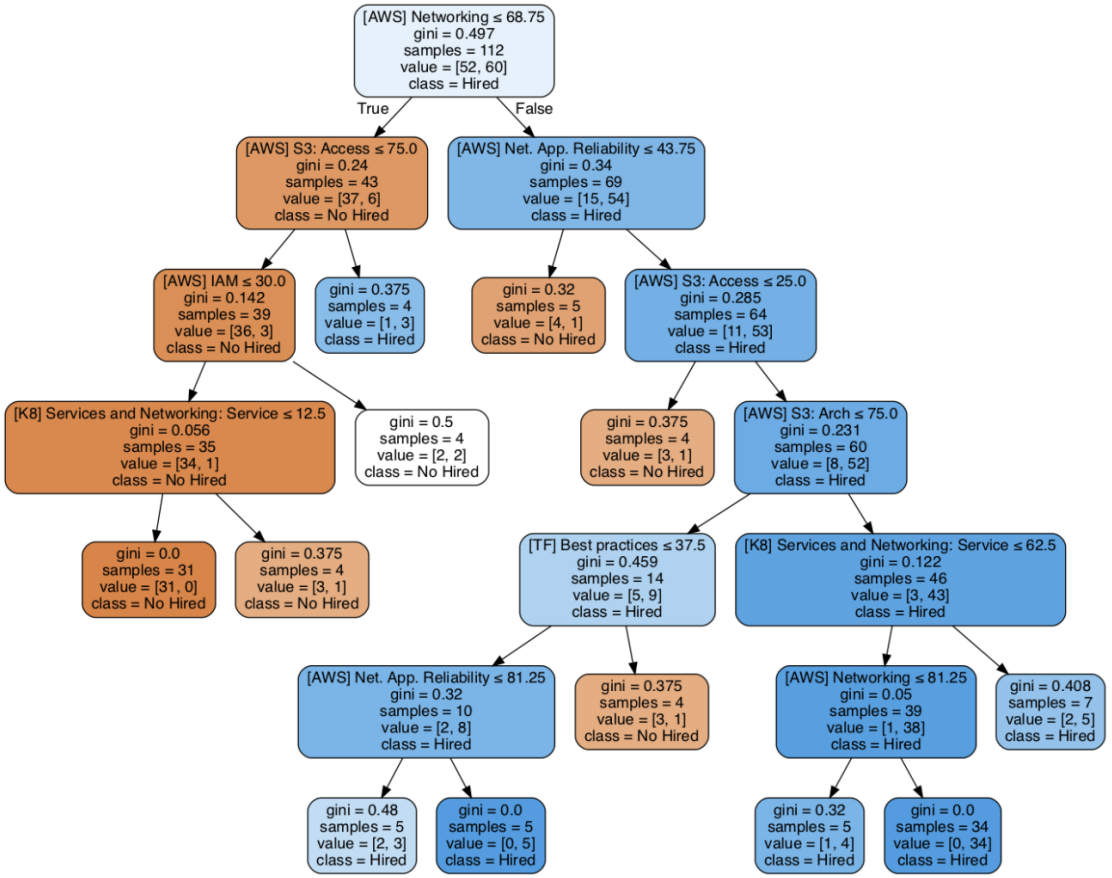



***Imagen 30:*** *Decision Tree para el criterio* ***gini***

Algo para remarcar sobre este análisis, tomando el criterio de **gini**, tanto las clases **Hire** como **No Hire** quedan *puras* en sus niveles más bajos, esto significa que *no hay otra agrupación posible*. Así mismo, en sus niveles más altos, la agrupación tiende a centrarse en el 0.4 y 0.5, esto significa que *las clases y los cortes están balanceados*. Habiendo dicho esto, podemos considerar que tanto la precisión cómo la exactitud del modelo, son bastante elevadas.

Mirando los cortes, podemos ver que si el candidato obtiene una nota igual o superior al ~ **69%** en *[AWS] Networking*, el mismo es **contratado**. Las excepciones a esto son que:

- Obtenga una nota menor al ~ **43%** en *[AWS] Net. App. Reliability*, o bien,
- Obtenga una nota menor al **25%** en *[AWS] S3 Access*.

Por el contrario, si el candidato obtiene una nota igual o inferior al ~ **69%** en *[AWS] Networking*, el mismo es **descartado**. Las excepciones a esto son que:

- Obtenga una nota mayor al **75%** en *[AWS] S3 Access*.

# Conclusiones

Independientemente de los resultados arrojados por los modelos, resulta fundamental volver a remarcar algunos puntos:

El **primero** es entender que los resultados **nunca serán óptimos**, ya que el *label principal fue seteado bajo una lógica matemática*, ya que no se contaba con los datos. Si hubiéramos contado con la información, podríamos haber arribado a conclusiones más interesantes, o al menos poder replantearnos más situaciones y evaluar mayores problemáticas.

El **segundo,** es que se estuvo muy cerca de lograr un *modelo "perfecto"*. Si bien no existen los modelos perfectos, entiéndase por perfecto en este caso que cumpla con todos nuestros KPIs originalmente definidos. Quizás, una vez madurado el modelo con más re-entrenamientos, podremos lograr eso.

El **tercero** es que, los márgenes de error devueltos por el modelo de **KNN** son extremadamente bajos, **1 error sobre 29**, lo cual da un **error menor al 4%** para ser precisos**.** Por ser *una primera iteración*, por *tratarse de una conducta social*, y por *haber trabajado con la calidad de datos que se obtuvo*, podemos afirmar sin lugar a duda que **el resultado es fantástico.**

El **cuarto** consta en que la *validación de resultados es positiva*. Esto quiere decir que, tanto los resultados obtenidos por el *Feature Importance* cómo por el *Decision Tree* son congruentes con las decisiones tomadas en **Caylent.** Y no es poca cosa, que estas técnicas provienen con una validación de **hyperparameters**, por lo cual esto refuerza aún más la credibilidad del análisis.

El **quinto**, es que se pudieron establecer los **happy** y **unhappy paths** del proceso de selección mediante el **Decision Tree**. Esto ayuda a marcar una orientación de cuáles son las preguntas que más nos hacen sentido y que valoramos realmente de un candidato.

El **sexto** y último, pero no menos importante, es que aún incluso a una cantidad reducida de observaciones sobre las categorías de **Kubernetes** y **Terraform,** algunos valores se ponderaron significamente, tal es el caso de **[K8] Services & Networking: Service** principalmente en el *Decision Tree* pero también en el *Feature Importance*.

# Proximos pasos

Cómo próximos pasos, podemos definir las siguientes acciones que podrán ser tenidas en cuenta en un futuro:

1. Iterar con el equipo de **Talent / Management** para validar estos resultados. En esta instancia, y siguiendo el feedback del peer review, se buscará atacar los siguientes ítems:
    a. Intentar dimensionar económicamente el tamaño del problema y cuanto ayuda está solución a reducirlo.
    b. Investigar qué otros beneficios trae resolver este problema.
2. Conseguir los labels reales y realizar una nueva corrida de los modelos. En tal caso, se podrá incluir los *Tiers (I, II, III)* de selección del candidato.
3. Conseguir los **missings** faltantes.
4. Llevar a producción el modelo una vez habiendo iterado con los respectivos stakeholders.

# *Entrega 4*

## Correcciones realizadas

### Generales

- Se actualizó el Project Charter con las fechas.
- Se siguieron las recomendaciones de la **cátedra**, así cómo también los del **peer review** (la mayoría, aquellos que eran factibles) de realizarse.
- Se alinearon los entregables con los resultados finales.
- Se preparó una sesión de validación con el equipo de **Talent/HR** para iterar sobre resultados y posibles próximos pasos.
- Se agregaron más Próximos Pasos.

### Respecto a la Entrega 1

- Se reformuló el enunciado de ¿Cómo se medirá el valor e impacto en el negocio?, dando una mayor profundidad al agregar más ítems y al ajustarlo a la realidad actual
- Se reformuló el enunciado de ¿Qué KPIs se utilizarán y cómo se calcularán?, para que se entienda mejor qué **KPIs** se consideran importantes y cómo se emplean finalmente en la puesta del modelo.

### Respecto a la Entrega 2

- Se profundizó sobre el gráfico de boxplots para detección de outliers.
- Se profundizó sobre la tabla de correlaciones, añadiendo varios pares adicionales, así cómo los bloques completos de cada examen.
- Se profundizó sobre el gráfico de interrelación entre variables.

### Respecto a la Entrega 3

- Se agregó una descripción del proceso más detallada, explicando el flujo de manera recursiva y no secuencial.
- Se dividieron los modelos por enunciados y se explicó que se buscaba o que se obtuvo de cada uno de ellos (antes estaban todos los resultados en una tabla).
- Se realizó un tuneo de **hyperparametros** para modelos de **Random Forest** y **KNN** mediante un **GridSearch**, buscando los parámetros óptimos.
- Se corrió el modelo **Decision Tree** y el **Feature Importance** con los parámetros óptimos obtenidos en el best parameter del RF.
- Se profundizó en las descripciones del **Feature Importance** y del **Decision Tree**.
- Se contrastaron los resultados de los modelos contra los **KPIs** definidos en la primera

instancia.

- Se removieron del informe los modelos redundantes. Asimismo, se removieron también los resultados extras del Anexo, que no aportan nada sino por el contrario traen confusión.
- Se removieron los gráficos de los **Decision Tree** adicionales que se encontraban en el Anexo, ya que no poseen una técnica validada.
- Se profundizó en la justificación de selección de modelos, mediante un exhaustivo análisis para los modelos de Random Forest y KNN, y cómo se arriba a la conclusión de esa selección.

# Anexo

## *Entrega 2*

El resto de las distribuciones de **AWS**, aquellas que no fueron agrupadas, se describen a continuación:

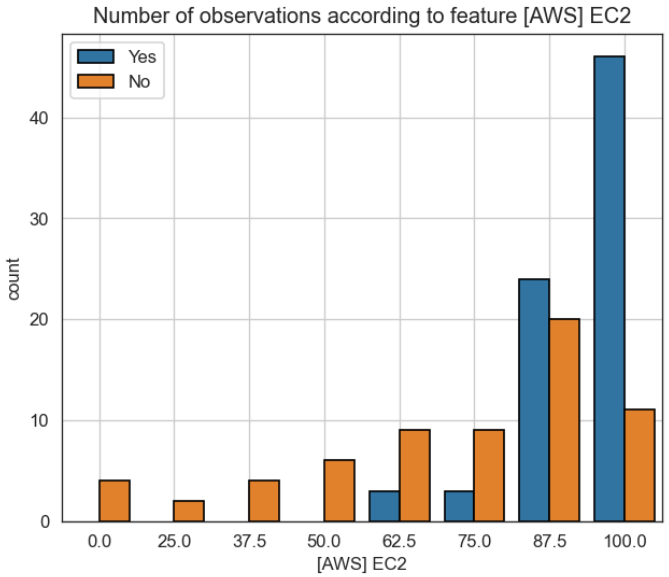



***Imagen Anexo 2.1:*** *AWS EC2*

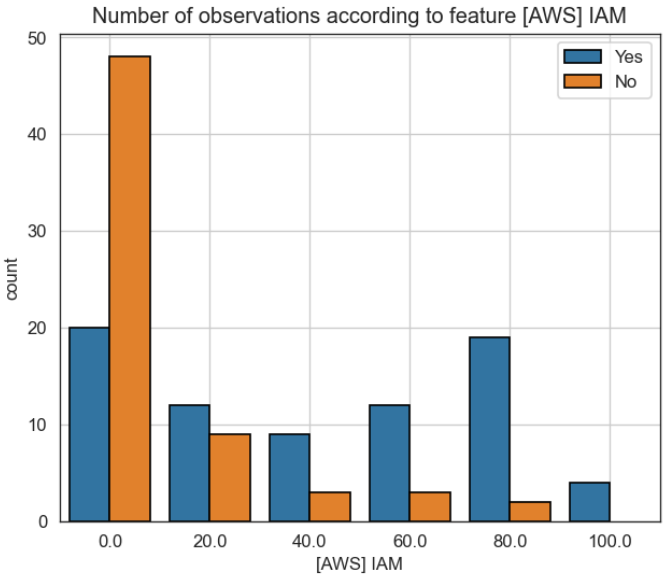



***Imagen Anexo 2.2:*** *AWS IAM*

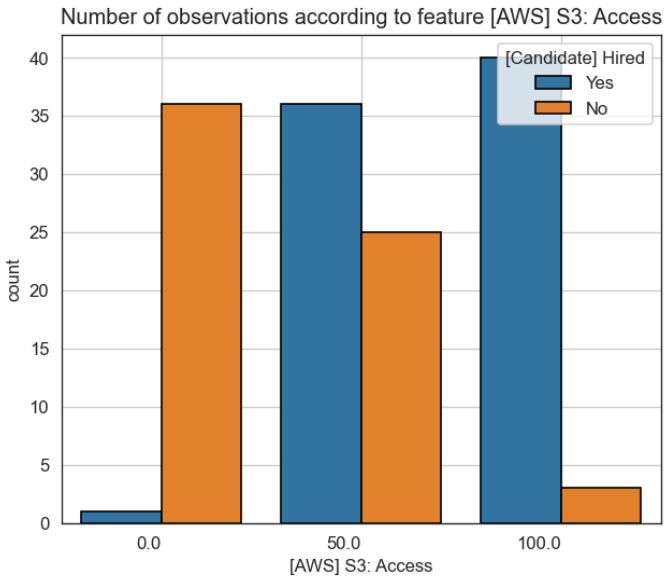



***Imagen Anexo 2.3:*** *AWS S3: Access*

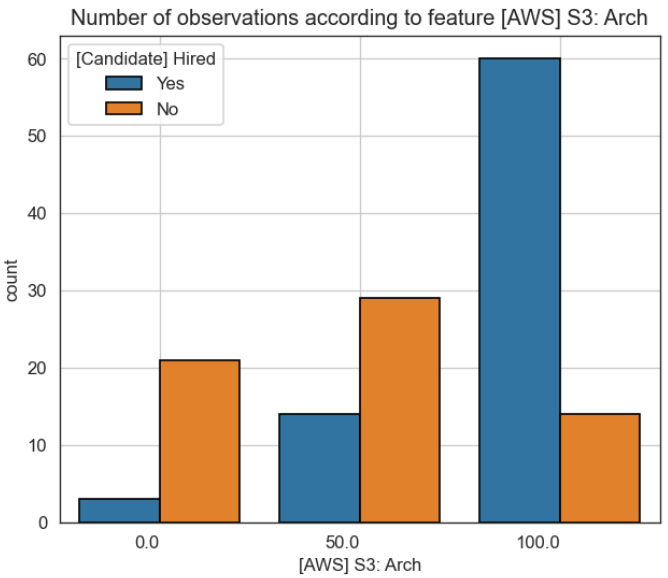



***Imagen Anexo 2.4:*** *AWS S3: Architecture*

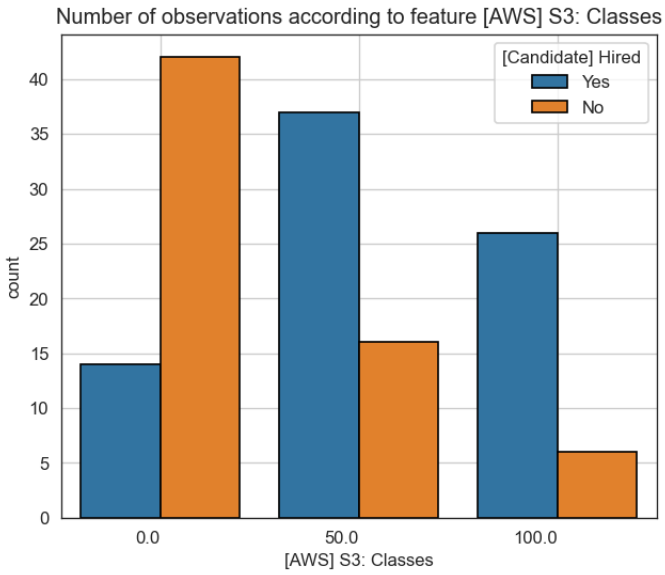



***Imagen Anexo 2.5:*** *AWS S3: Classes*

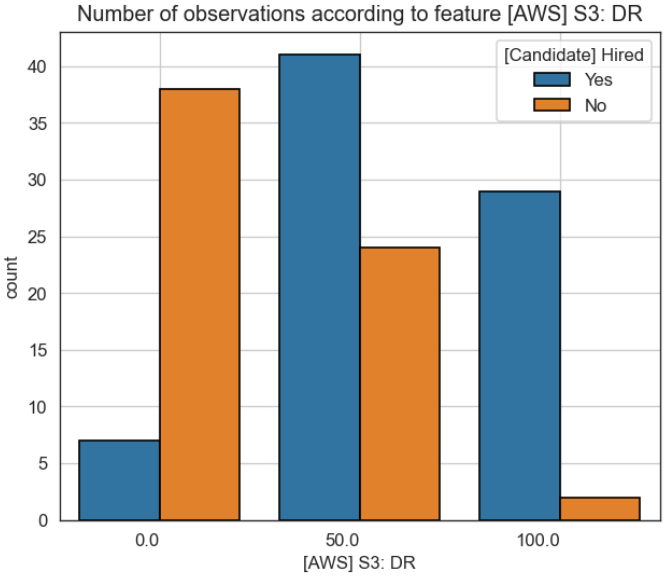



***Imagen Anexo 2.6:*** *AWS S3: DR*

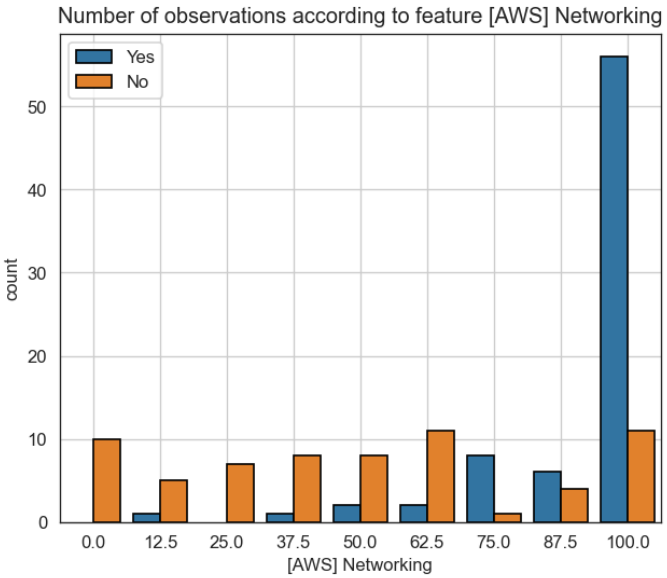



***Imagen Anexo 2.7:*** *AWS Networking*

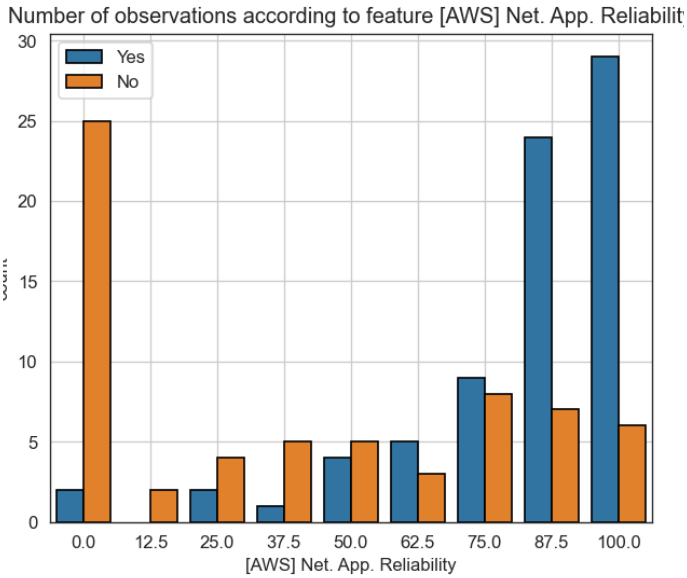



***Imagen Anexo 2.8:*** *AWS Networking Application Reliability*

El resto de las distribuciones de **Kubernetes [K8]** y **Terraform [TF]** se describen a continuación:

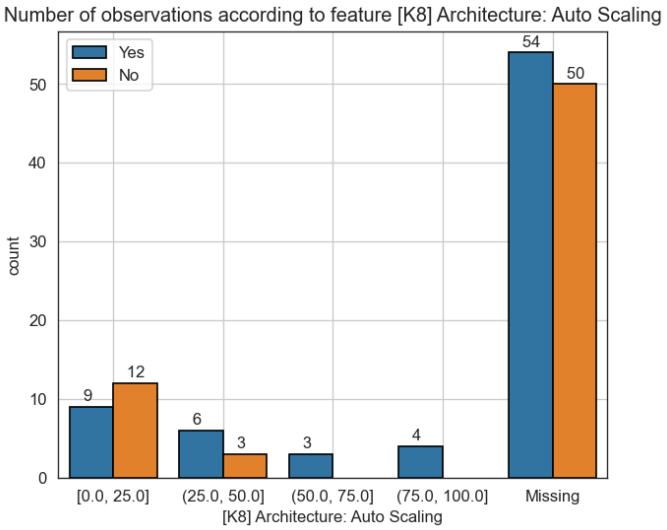



***Imagen Anexo 2.9:*** *K8 Architecture: Auto Scaling*

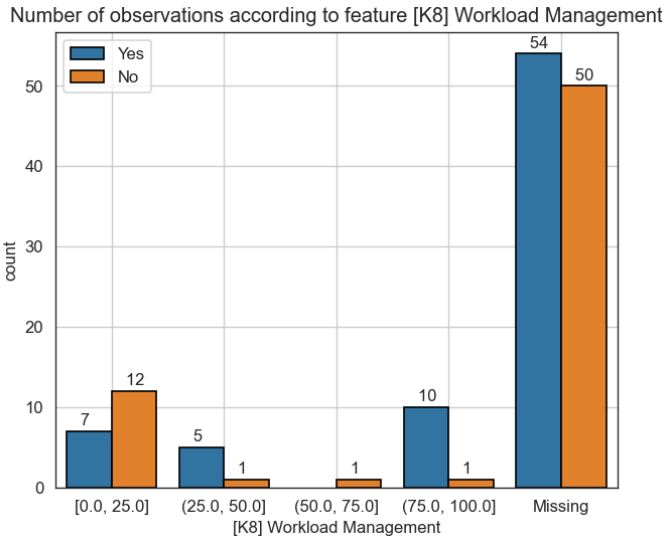



***Imagen Anexo 3.0:*** *K8 Workload Management*

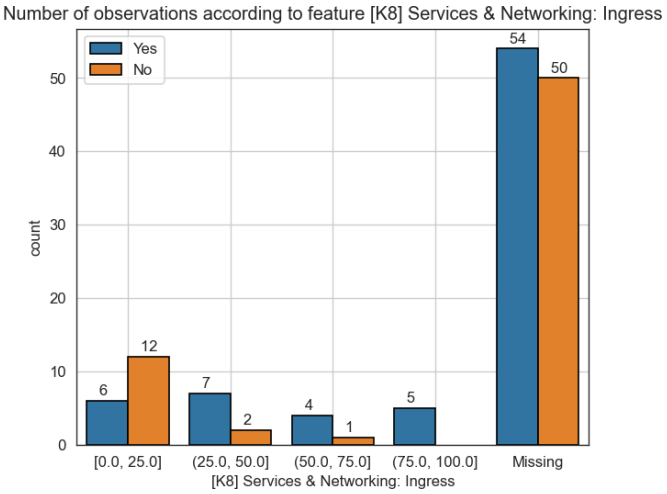



***Imagen Anexo 3.1:*** *K8 Services & Networking: Ingress*

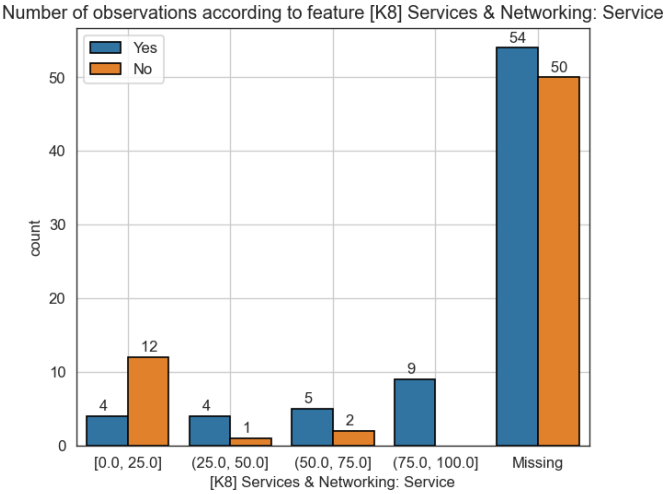



***Imagen Anexo 3.2:*** *K8 Services & Networking: Service*

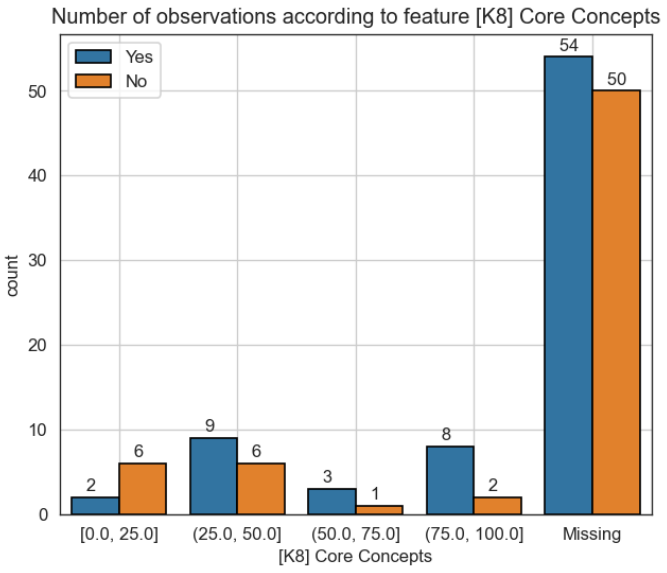



***Imagen Anexo 3.3:*** *K8 Core Concepts*

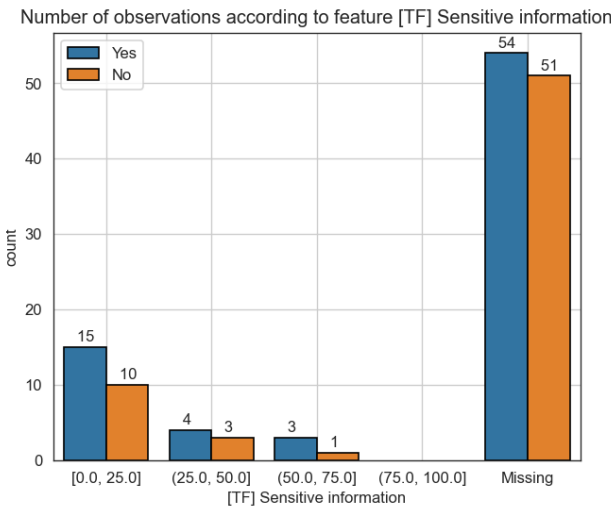



***Imagen Anexo 4.2:*** *Terraform sensitive information*

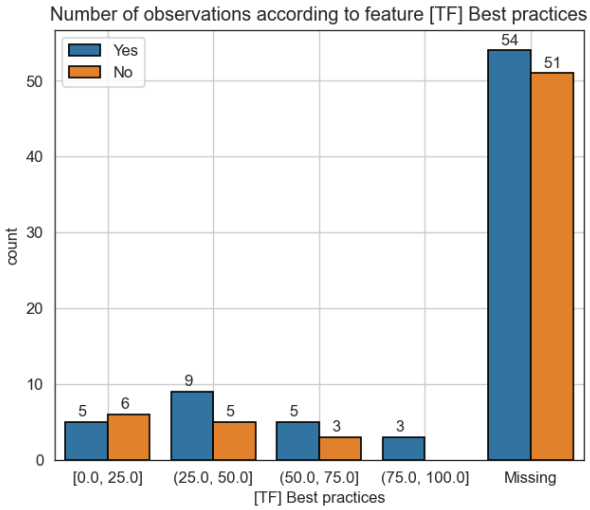



***Imagen Anexo 3.5:*** *Terraform best practices*

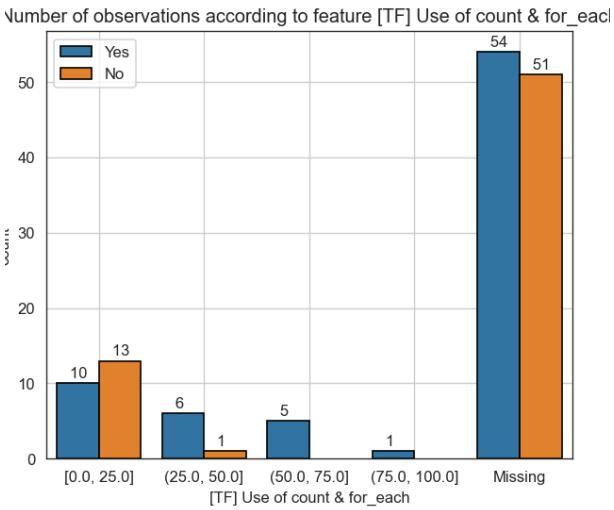



***Imagen Anexo 3.6:*** *Terraform use of count & for_each*

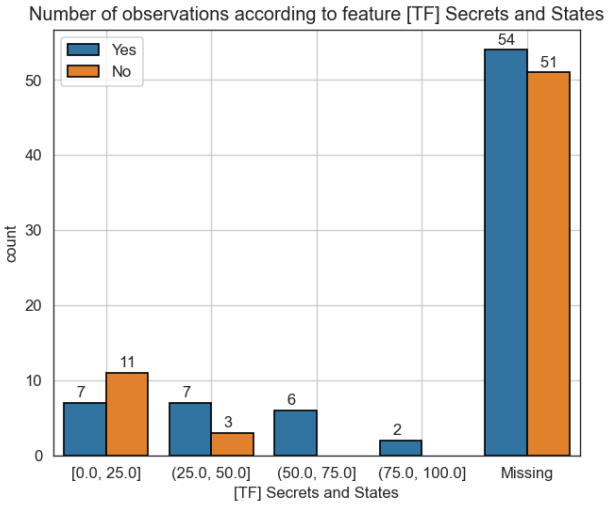



***Imagen 3.7:*** *Terraform secrets and states*

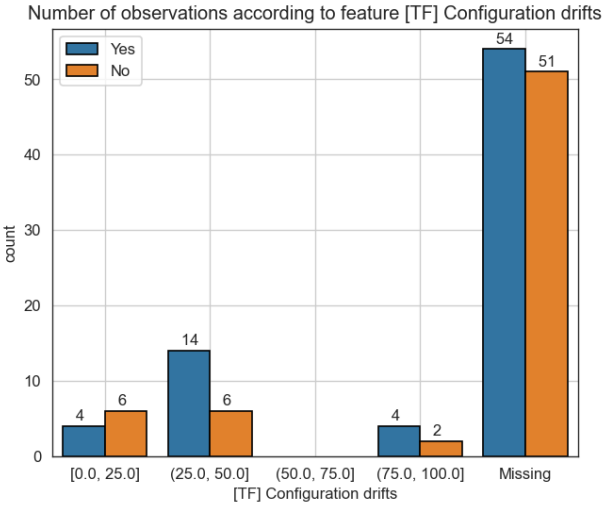
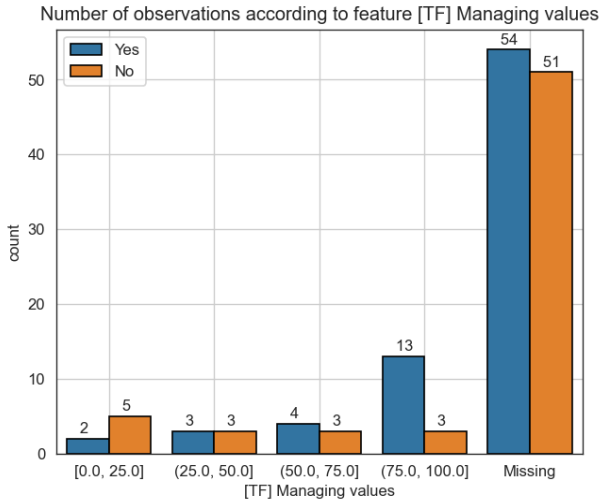



***Imagen Anexo 3.8:*** *Terraform configuration drifts*



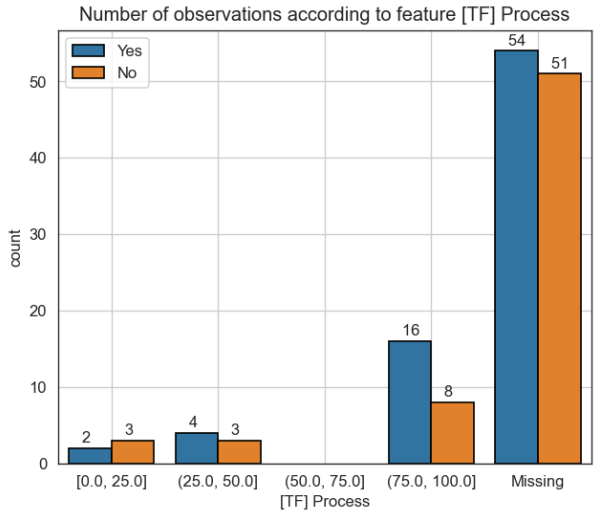



***Imagen Anexo 3.9:*** *Terraform managing values*

***Imagen Anexo 4.0:*** *Terraform process*

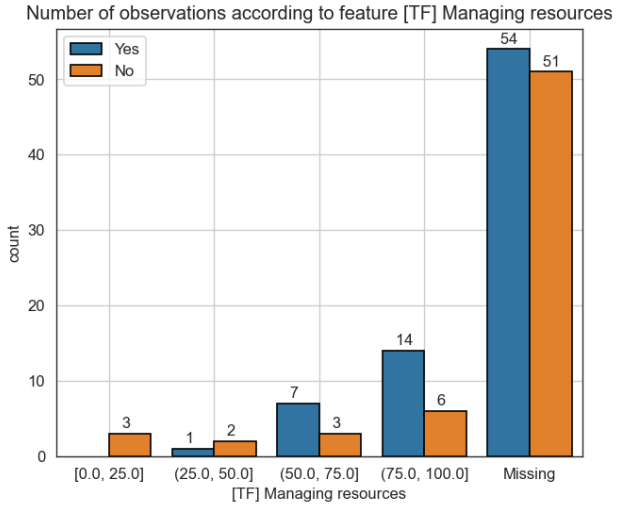



***Imagen Anexo 4.1:*** *Terraform managing resources*

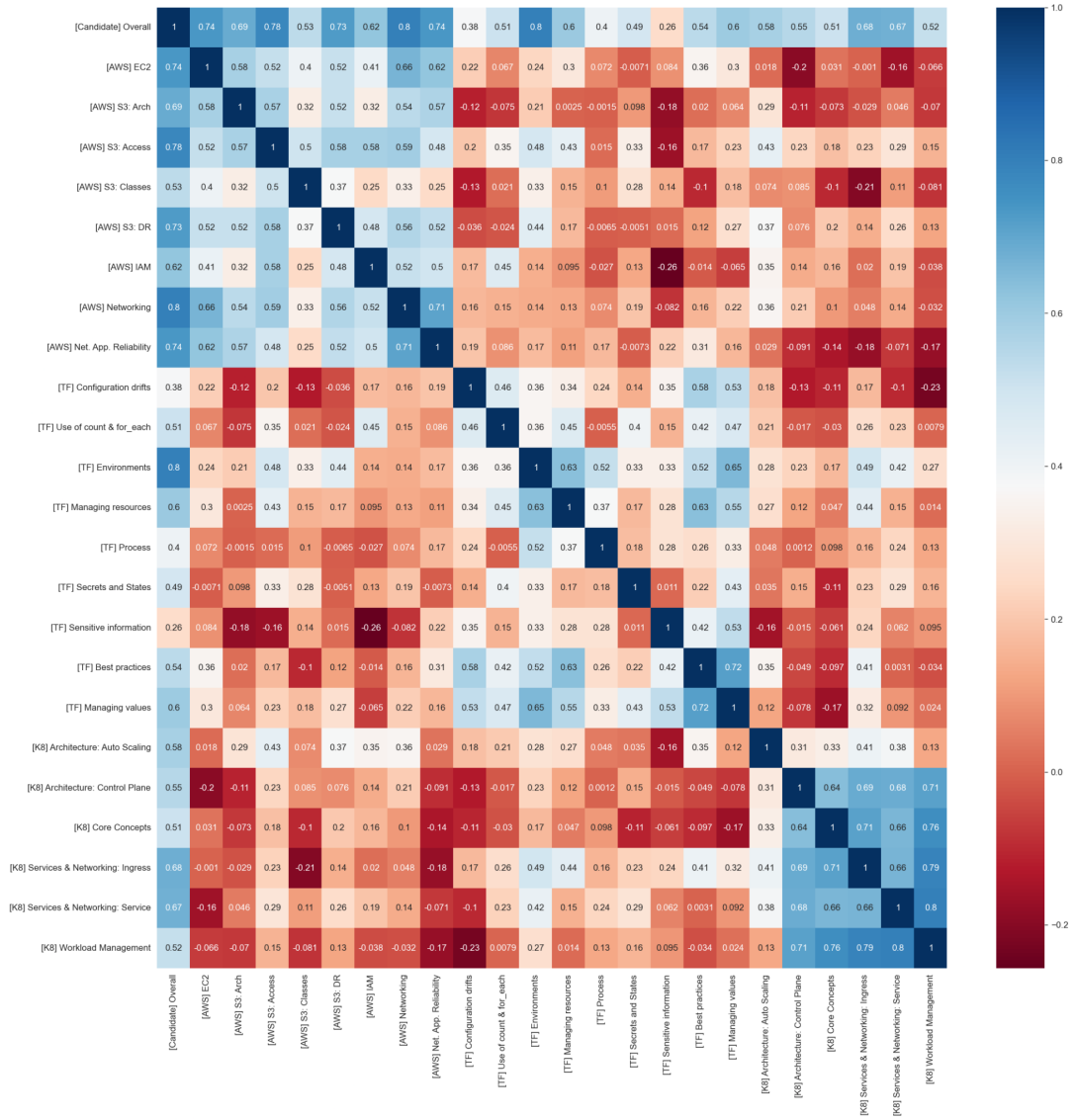



***Imagen 4.2:*** *Pearson correlation analysis*

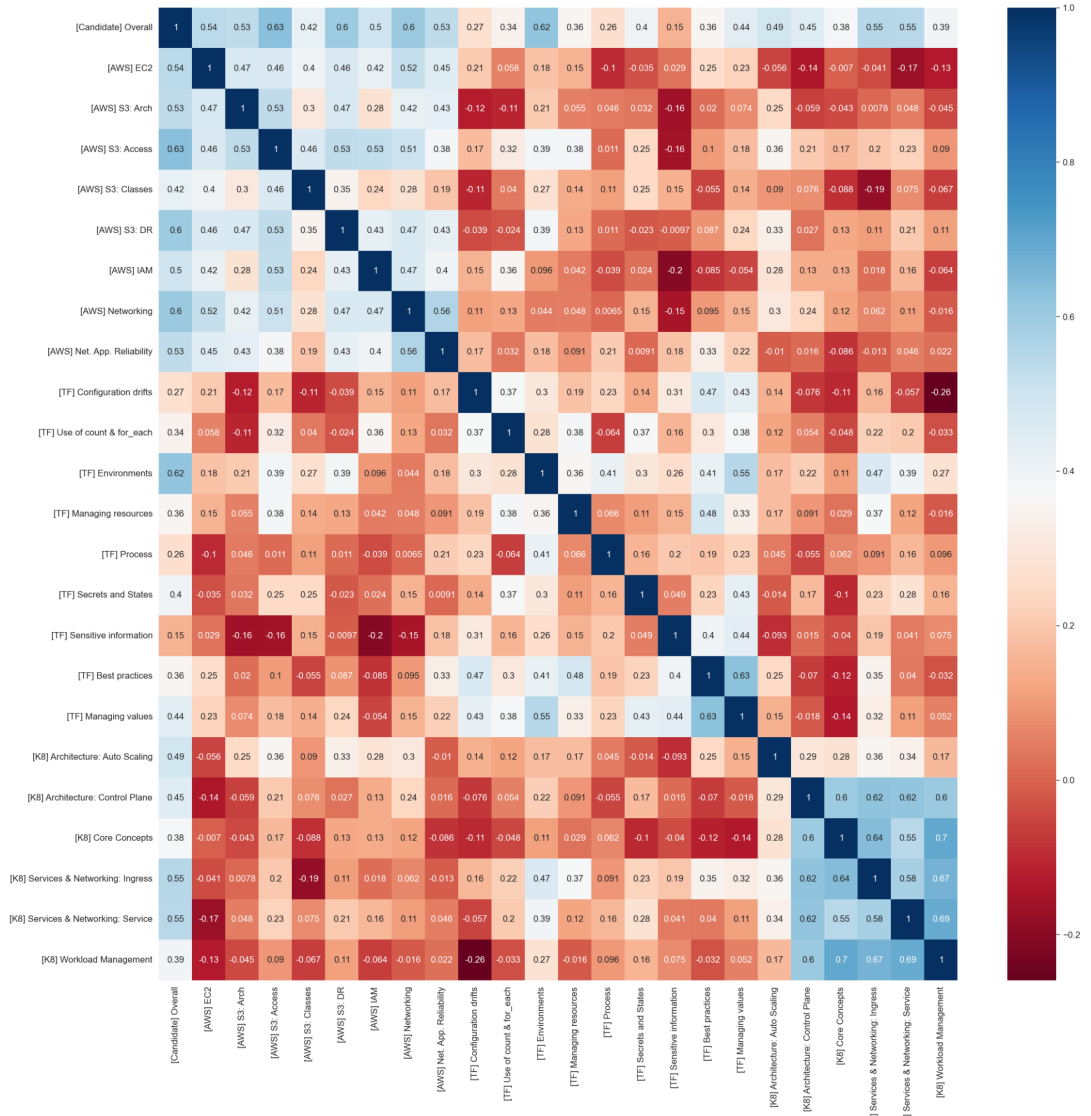



***Imagen 4.3:*** *Kendall correlation analysis*

## Entrega 3

```
Error: Kernel is dead
at g._sendKernelShellControl (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/extensio
n.js:52:1006305)
at g.sendShellMessage (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/extension.js:52:
1006074)
at g.requestExecute (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/extension.js:52:10
08616)
at d.requestExecute (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/extension.js:37:32
8037)
at S.requestExecute (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/extension.js:32:19
306)
at w.executeCodeCell (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/extension.js:52:3
00924)
at w.execute (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/extension.js:52:300551)
at w.start (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/extension.js:52:296215)
at processTicksAndRejections (internal/process/task_queues.js:93:5)
at async t.CellExecutionQueue.executeQueuedCells (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/
out/client/extension.js:52:310950)
at async t.CellExecutionQueue.start (/Users/agustin/.vscode/extensions/ms-toolsai.jupyter-2021.9.1101343141/out/client/ex
tension.js:52:310490)
```

***Imagen Anexo 4.4:*** *mensaje de error al ejecutar XGBoost*

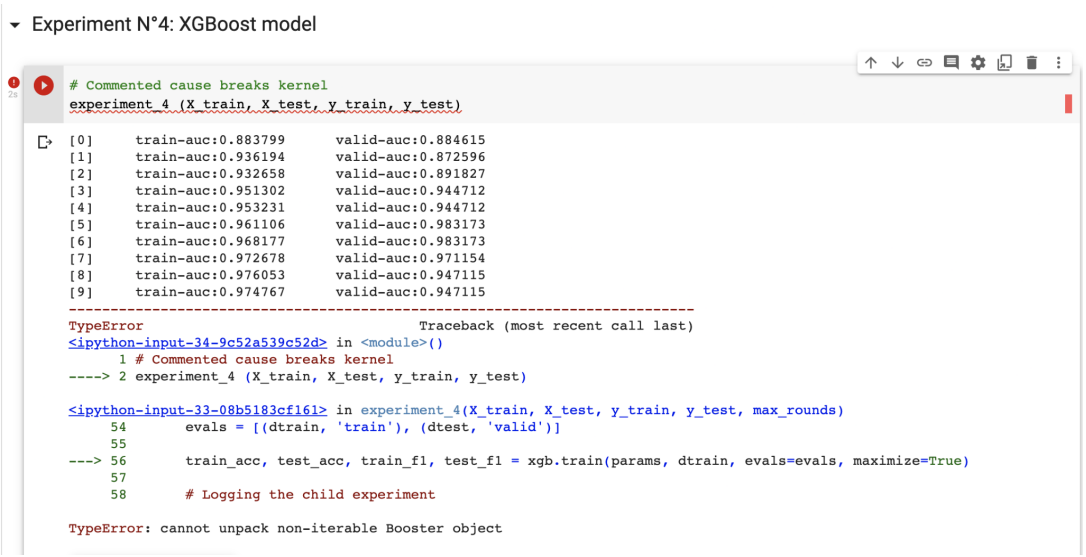



***Imagen Anexo 4.5:*** *workaround en Google Colab*