

Analítica prescriptiva y desarrollo en equipo

Programa en Analítica

Curso *Capstone - Inteligencia Artificial & Deep Learning*

Analítica Prescriptiva

Educación continua | Universidad de los Andes
octubre 6 – noviembre 24
2021

Hoy

De qué vamos a hablar

1. Analítica prescriptiva:

*Los fundamentos conceptuales,
optimización, herramientas en R*

2. Desarrollo de software en equipo:

*Casos Agile y Scrum, ¿qué particularidades
tiene un proyecto de programación?*

3. Unos minutos para conversar con sus equipos

*¿Qué prácticas podemos incluir? Charlemos
sobre nuestro reto. (10 minutos)*



Photo by [nappy](#) from [Pexels](#)

Analítica prescriptiva

Considerando decisiones,
conceptualmente



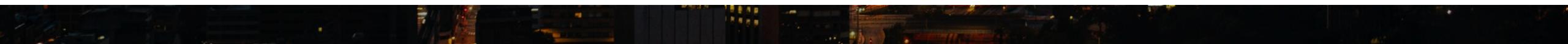
Photo by [Matilda Wormwood](#) from [Pexels](#)

Cada análisis tiene un producto distinto

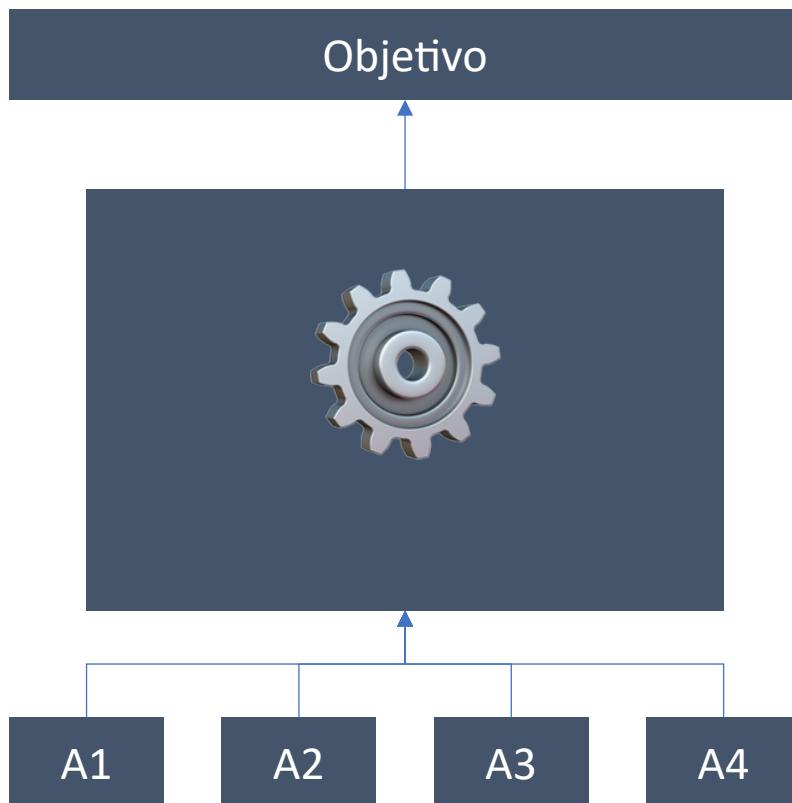
Estadística descriptiva un informe de aprendizajes

Estadística predictiva una herramienta utilizable, o una predicción

Estadística prescriptiva una recomendación de curso de acción



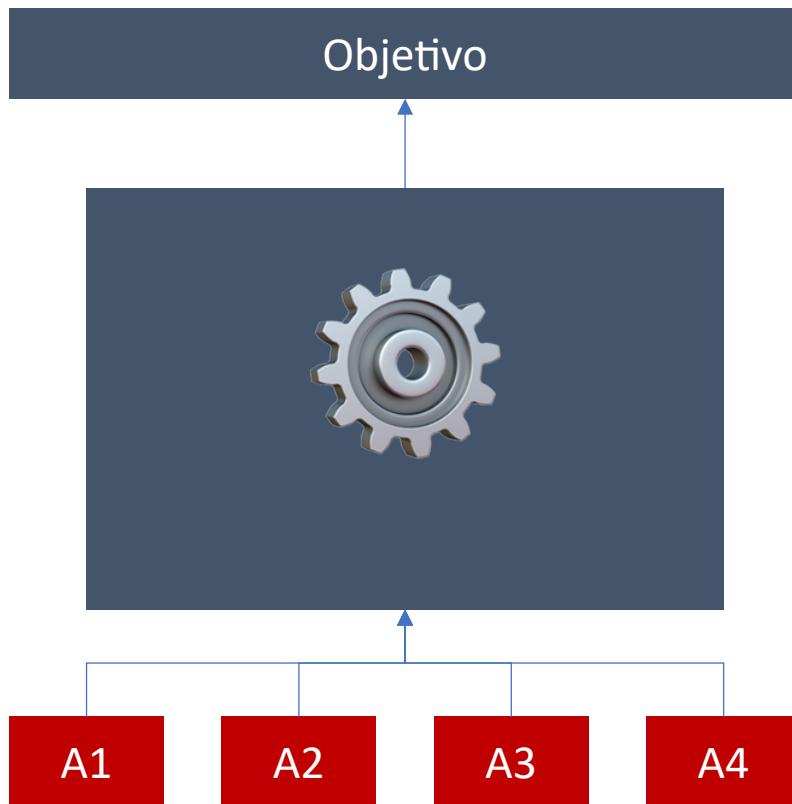
Los elementos



Un sistema de analítica prescriptiva se construye con tres componentes básicos:

- **Acciones**
- **Objetivo**
- **Modelo**

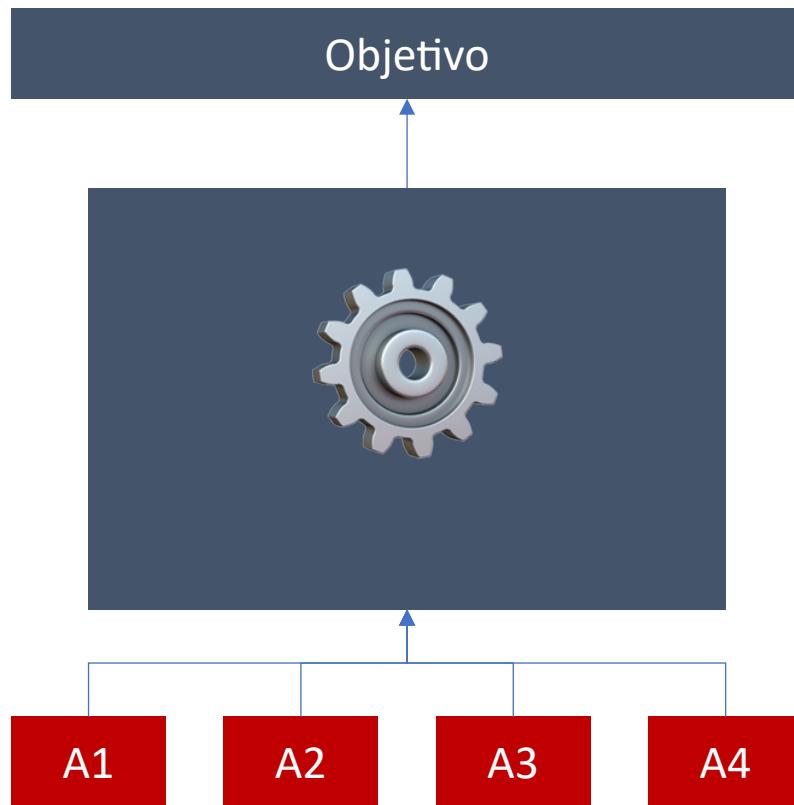
Las acciones



Hay dos tipos de acciones que se pueden tomar:

- Una opción entre varias: *¿Qué persona dirigir a consulta? ¿Qué ruta aérea utilizar para este destino?*
- Una opción entre infinitas: *¿Qué precio asignar? ¿Qué cantidad producir?*

Las acciones

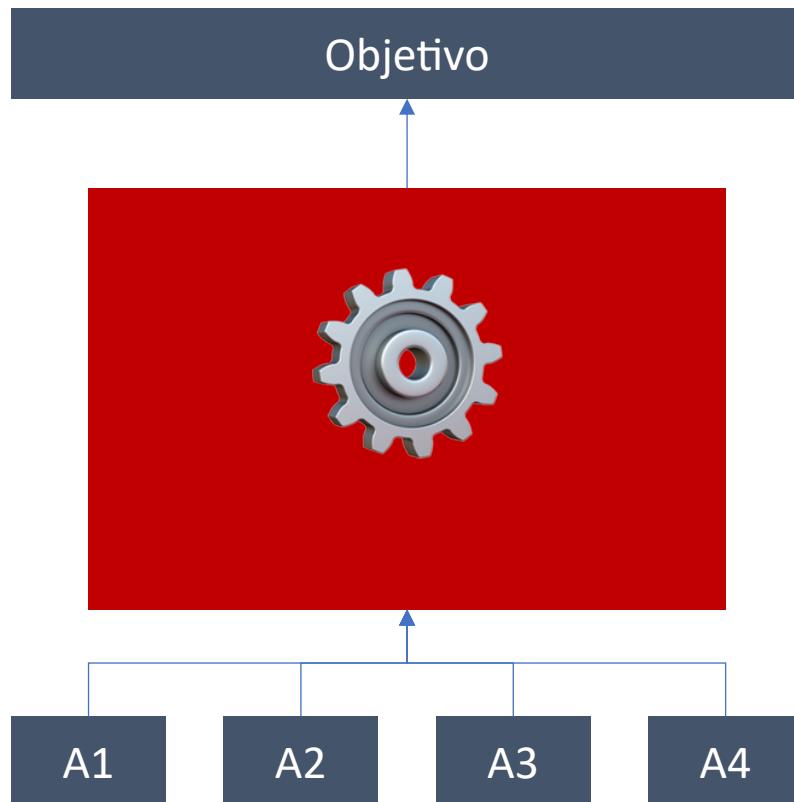


Esto nos va comenzar a determinar qué tipo de optimización utilizar

Optimización continua.

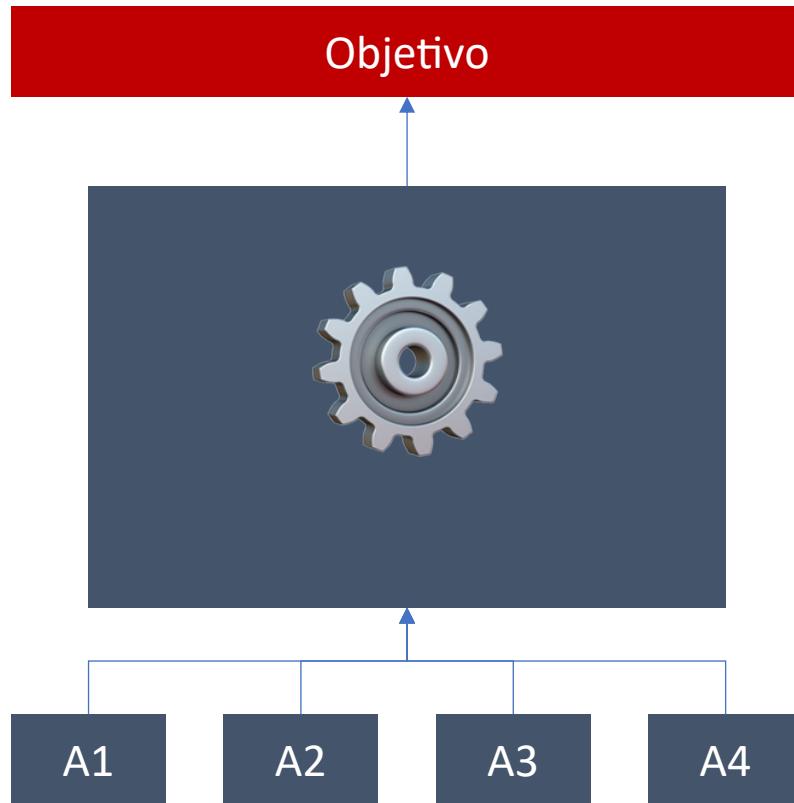
Optimización discreta.

El modelo



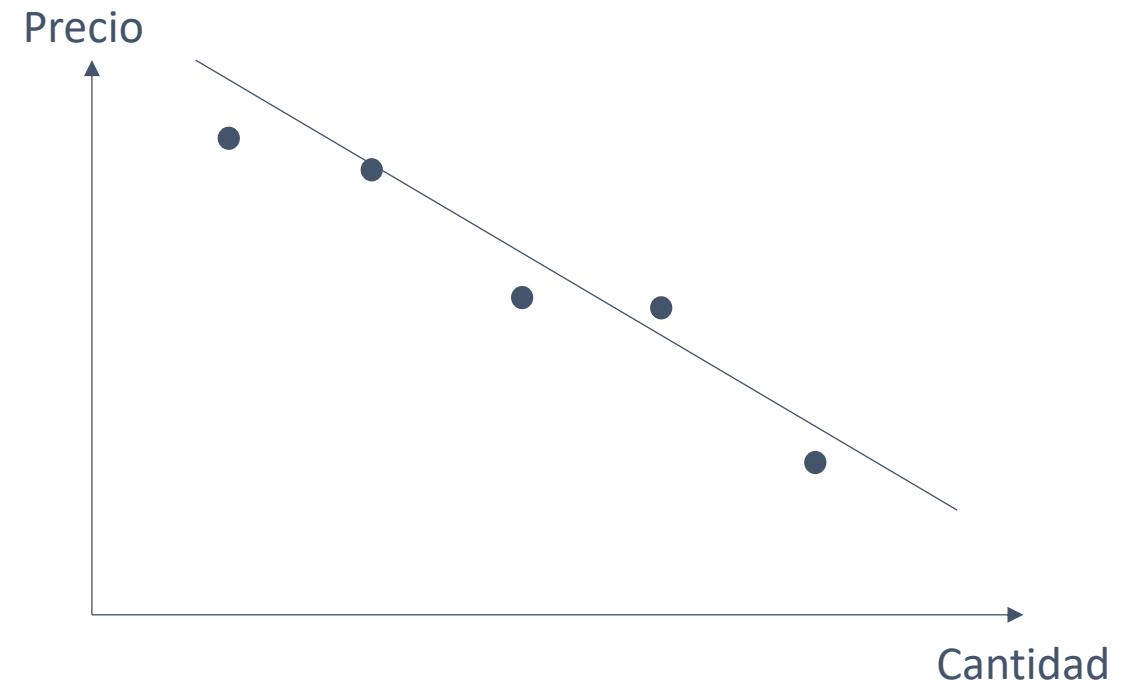
- Para que un modelo sea razonablemente útil es importante ejecutar unas primeras etapas de:
 - Analítica descriptiva: para intuir qué información parece estar correlacionada, etc.
 - Analítica predictiva: si un modelo no es bueno prediciendo, poco se puede confiar en ***su criterio***.
- * Para nuestro proyecto vamos a ejecutar con mucha agilidad estas etapas.

El objetivo

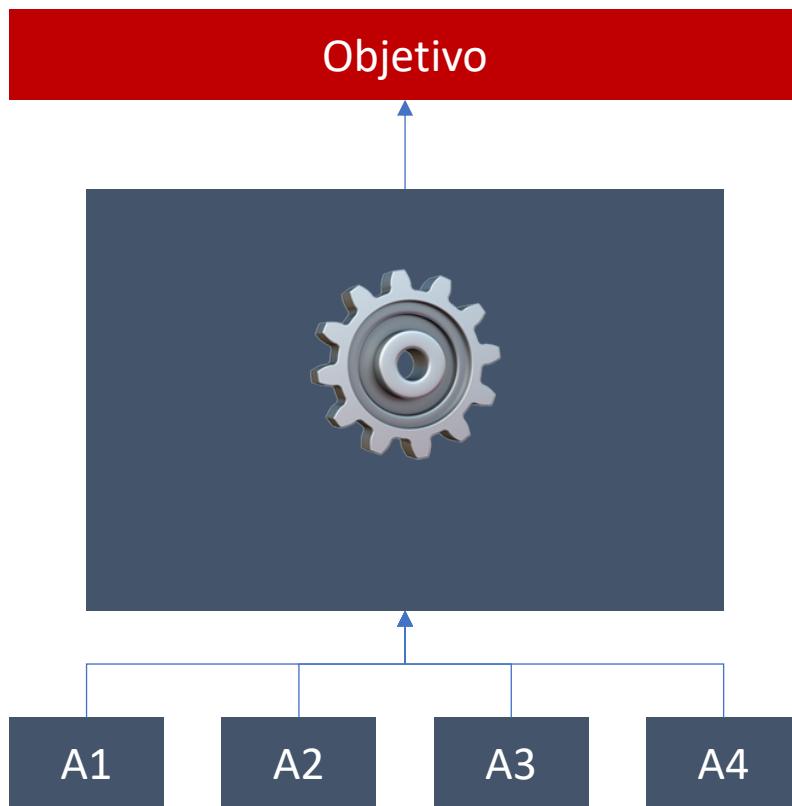


Los mismos datos nos pueden dar conclusiones distintas.

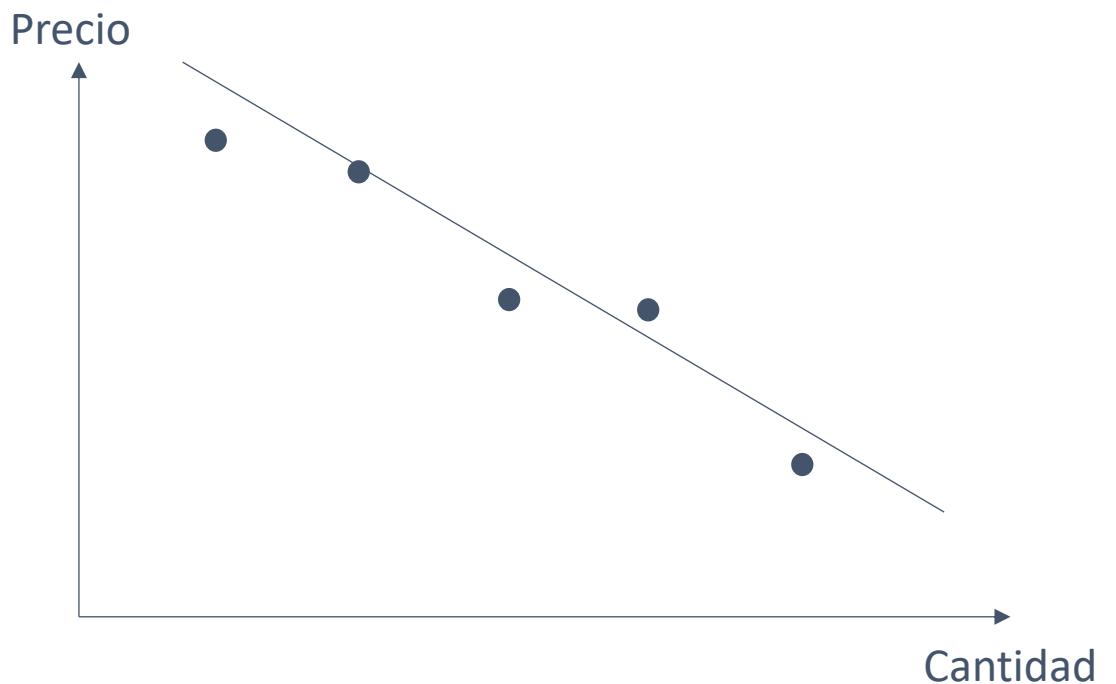
- Caso de Clara: vender a precio 0 para lograr mayor cantidad de ventas.



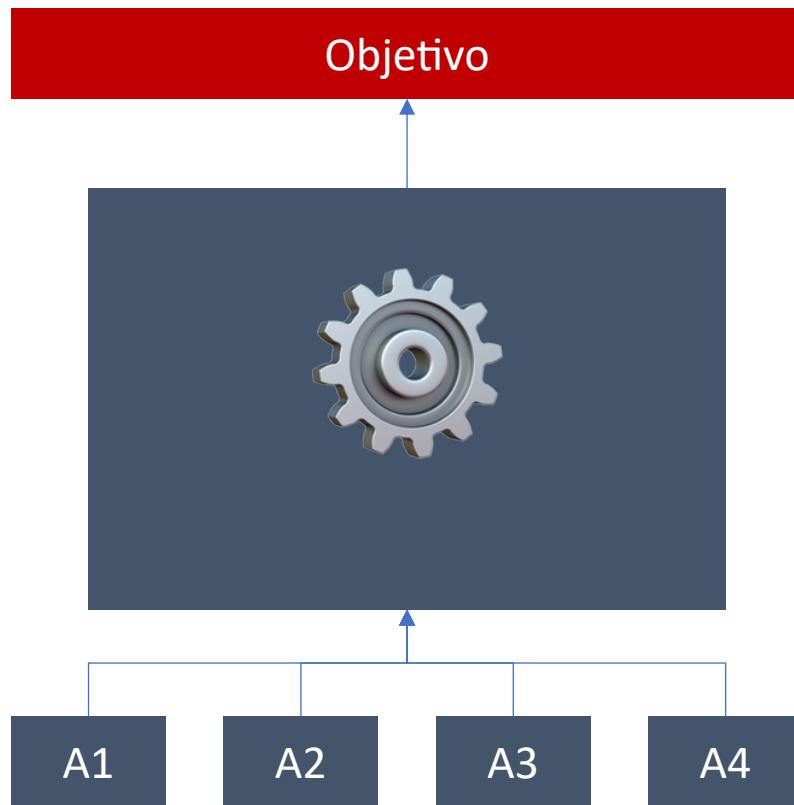
El objetivo



**¿Será que la empresa de Clara
realmente quiere maximizar
ventas?**

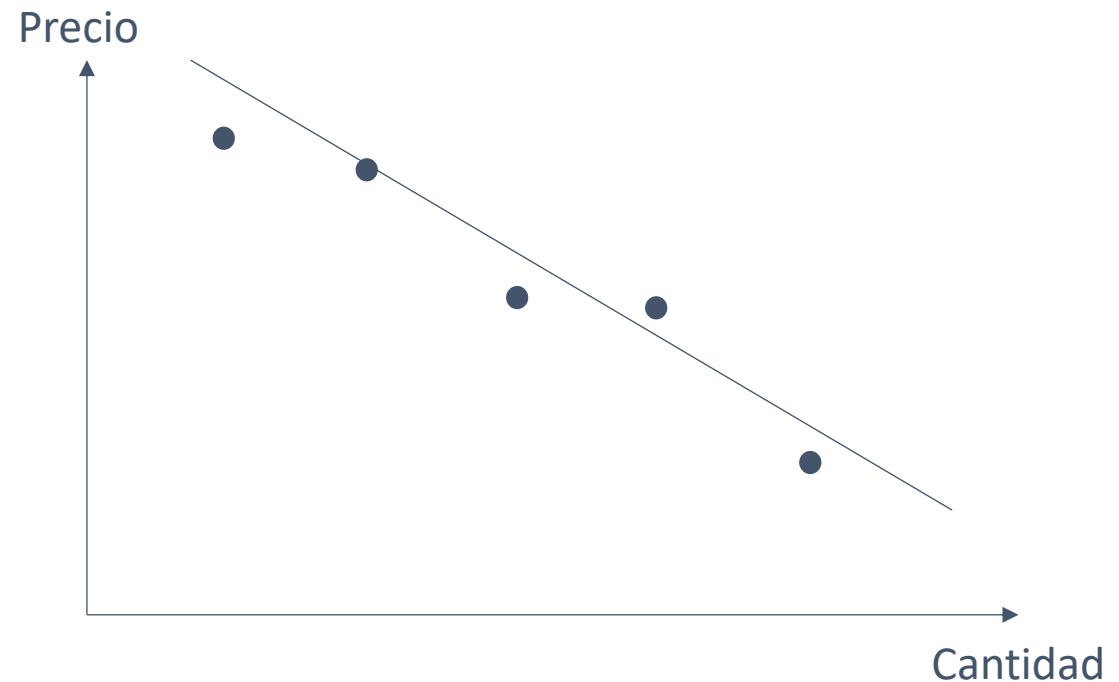


El objetivo

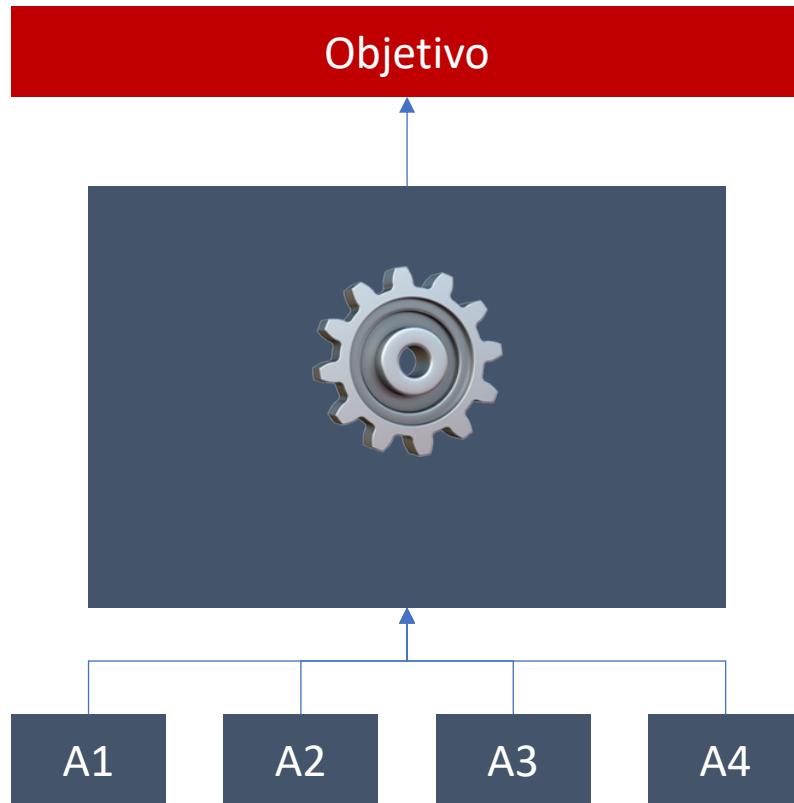


Una alternativa es maximizar beneficios:

$$\pi = \text{precio} \cdot \text{cantidad} - \text{costos}$$



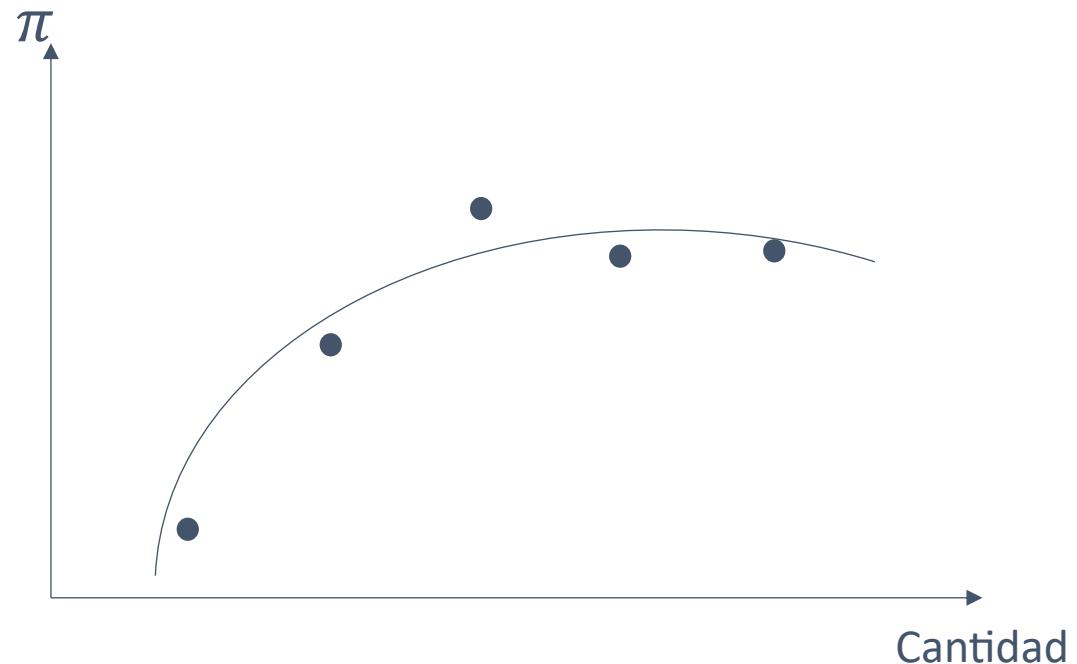
El objetivo



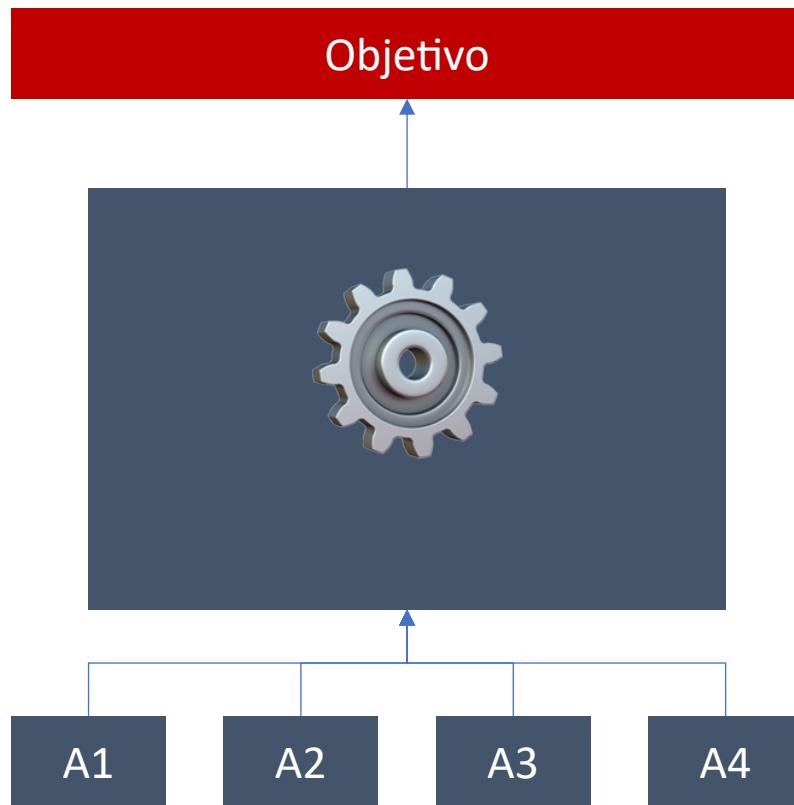
Una alternativa es maximizar beneficios:

$$\pi = \text{precio} \cdot \text{cantidad} - \text{costos}$$

Si clara calculara los beneficios para cada precio preguntado...



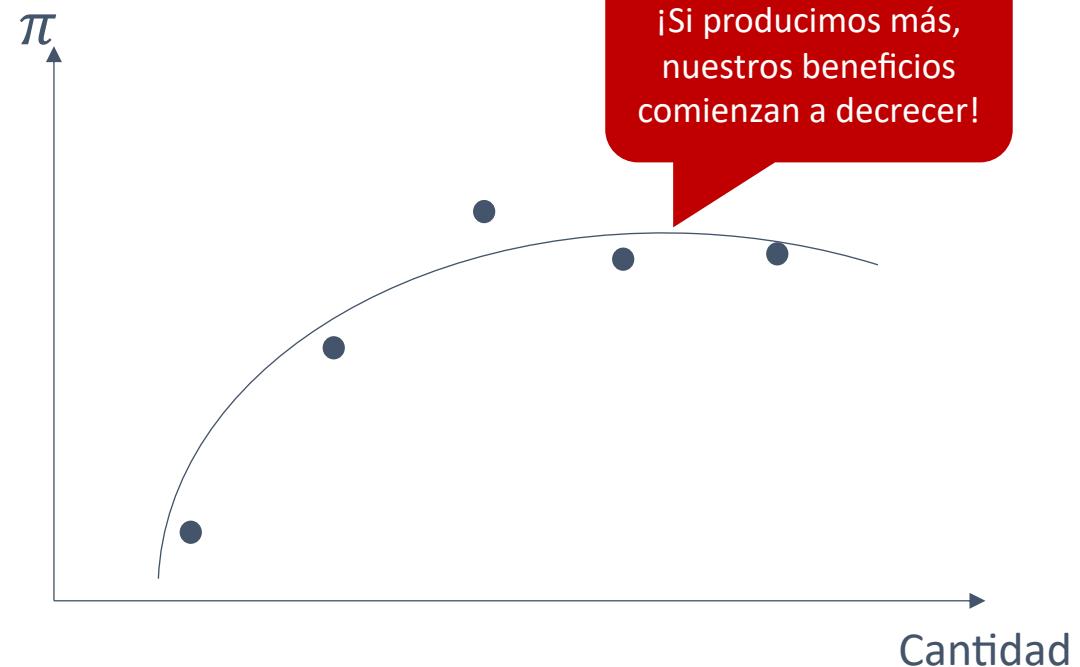
El objetivo



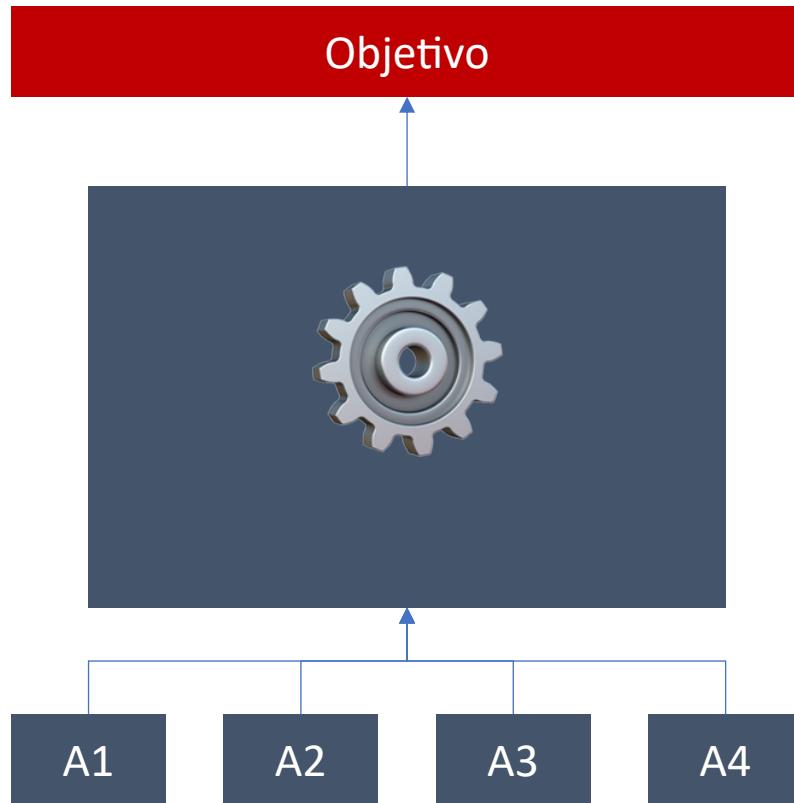
Una alternativa es maximizar beneficios:

$$\pi = \text{precio} \cdot \text{cantidad} - \text{costos}$$

Si clara calculara los beneficios para cada precio preguntado...



El objetivo



Una alternativa es maximizar beneficios:

$$\pi = \text{precio} \cdot \text{cantidad} - \text{costos}$$

Si Clara calculara los beneficios para cada precio preguntado...



Advertencias

- Situaciones con **incertidumbre**, o que requieren **precisión**, en las que hablamos de **eficiencia**, se pueden beneficiar de análisis prescriptivo.
- Estos sistemas pueden **agilizar decisiones** que requieren mucha experticia cuando se las puede entrenar.
- Los modelos de analítica prescriptiva **no son infalibles**.
- Un modelo de analítica prescriptiva **es tan bueno como se haya formulado el sistema**.
 - ¿Qué información vamos a considerar?
 - ¿Cómo estamos midiendo los objetivos?
 - ¿Qué modelos funcionan mejor?

Potenciales retos

- Al principio pueden salir relaciones descriptivas **contraintuitivas**...
 - Para eso vale la pena correr más correlaciones con otras variables relevantes
 - A veces, un modelo que predice muy bien, no es interpretable
 - Ojo con el problema de *overfitting*: considerar cuando evaluemos la máquina
- En estas clases, puede que no nos alcance el tiempo para probar todo lo que quisiéramos.
 - Distribuyan el trabajo en el grupo de forma coordinada
 - Concéntrense en entender las pocas cosas que alcancen a implementar (el principio *deliver something*) en lugar de cubrir todas las posibilidades



Anécdota de principiante...

- Caso de **contratación de profesores** para minimizar probabilidad cursos huérfanos.
- **Acciones** (áreas de docencia): historia pensamiento, historia del análisis, macro, micro, cuantitativos, (mixtos también).
- El **modelo** debía considerar: sabáticos, número de cargas disponibles, tipo de contratación (tiempo), demanda de los cursos...
- El modelo debía relacionar una planta profesoral con el **número de cursos huérfanos** en los próximos 2 años.
- El objetivo era **minimizar el número de cursos huérfanos** en los próximos dos años (asumiendo que sólo contratamos una vez).

Anécdota de principiante...

Anécdota de principiante...

- Desarrollé una formalización cuidadosa, detallada, “preciosa”.
- Noten: aquí no usé Machine Learning ni IA, pura aritmética y lógica de optimización discreta: **calcular el objetivo en cada posible caso, escoger el mejor**.
- El sistema (y mi motivación) explotó en complejidad: se volvió abrumador para mí y difícil de calcular.
- Una posible mejor aproximación: entrenar una red neuronal, que correspondiera una **planta profesoral con algún indicador de riesgo** de cursos huérfanos.
- Como hubiera tenido que levantar esos datos a mano, el proyecto quedó ahí.



Un par de aprendizajes

- Construir un sistema prescriptivo puede parecer abrumador en algunos casos pero puede ser porque lo estamos formalizando de forma muy compleja.
- Es bueno evaluar en qué situaciones es más útil cada herramienta:
 - Optimización discreta programada a mano puede ser útil para optimizar algunos procesos
 - Machine Learning e IA pueden ser buenos modelos para lidiar con incertidumbre, etc.
 - Programación lineal (que vamos a ver a continuación) es una técnica eficiente de computar en muchos casos.



Optimización

Un vistazo general a la programación
lineal

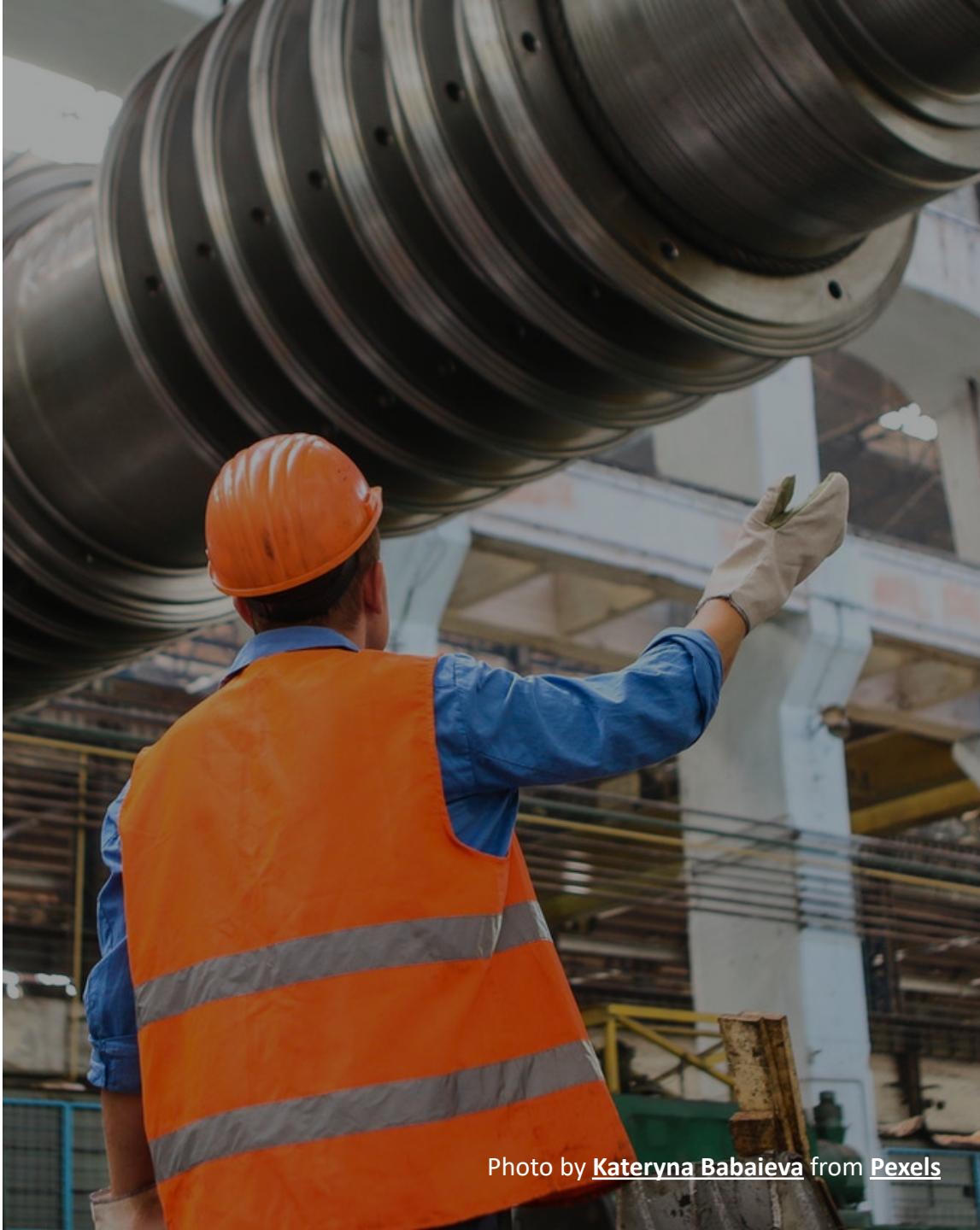


Photo by [Kateryna Babaieva](#) from [Pexels](#)

Notas generales sobre optimización:

- Optimizar significa **maximizar o minimizar** una variable objetivo
- Para esto tenemos **variables de decisión**
- Sobre esas variables de decisión definimos **restricciones**
- **Note:** en la vida real, una situación puede tener una solución, o varias...
 - Imagine que el caso de Clara hubiera sido este, ¿Cuál es el precio?



Disyuntivas: no se puede tenerlo todo

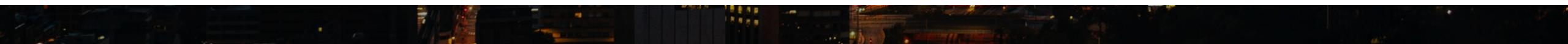
Uno de los puntos clave de la optimización es mediar en problemas donde las variables de decisión le pegan de formas diversas al objetivo.

Por ejemplo:

Hay un presupuesto fijo para organizar un coctail, se quiere lograr la máxima satisfacción posible con la comida en las encuestas... Pero:

- Más comida puede dejar más satisfechos/as a los/as asistentes
- Comida cocinada por alguien más experto puede ser más satisfactoria

Mientras más experto el chef, alcanza para menor cantidad de comida



Hasta ahora hemos mencionado:

Optimización discreta explícita:

1. Para cada opción de decisión, se calcula el valor de la variable objetivo
2. Se van anotando esos valores
3. Al final se revisa cuál es el mayor/menor
4. Se toma la decisión correspondiente a ese valor objetivo

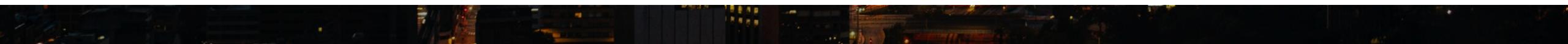


Hemos mencionado el uso de ML e IA

Se puede entrenar una máquina de IA para que identifique cómo una decisión afecta el estado de la variable objetivo.

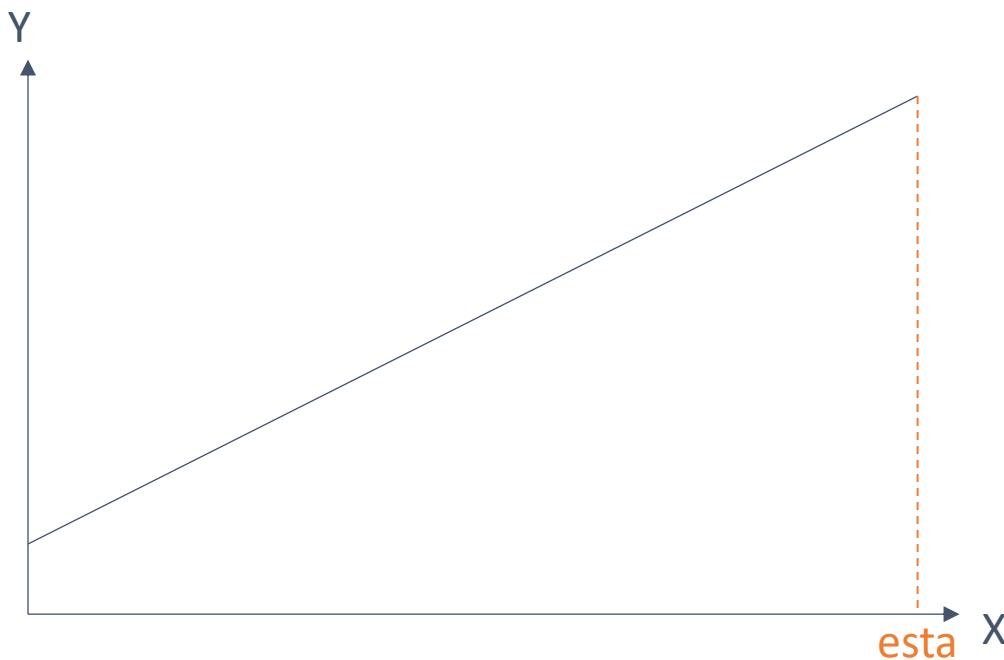
Esto nos permite construir un modelo que probabilísticamente relaciona acciones con objetivos

- Pueden ser lentos de entrenar
- Pero son rápidos de usar

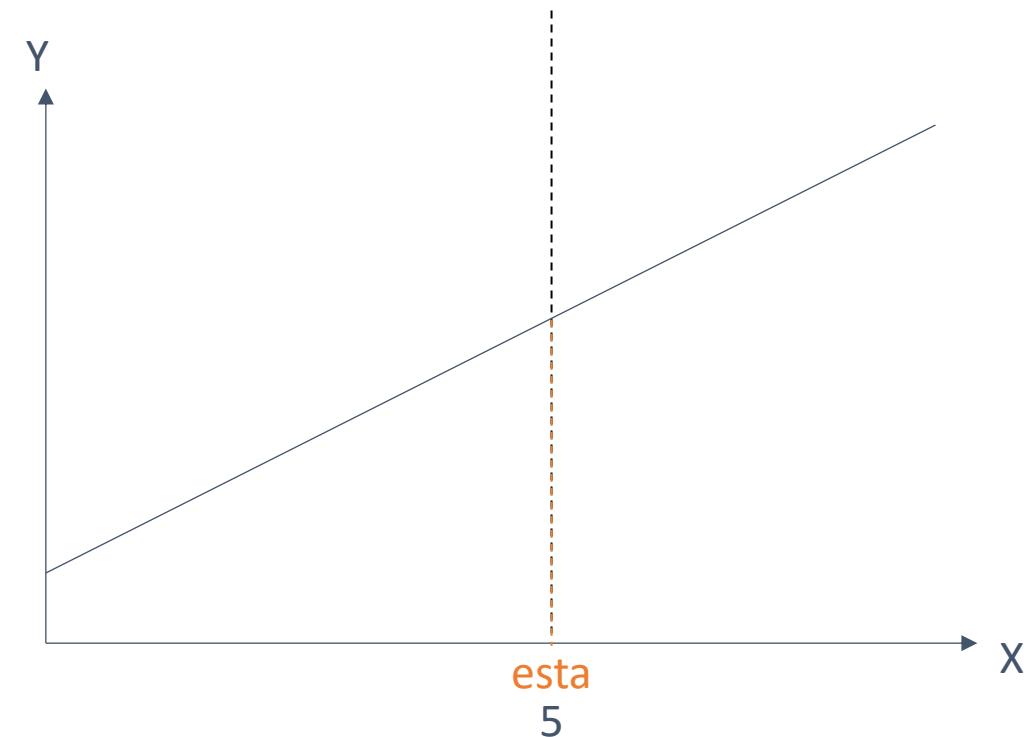


¿Cómo se ve optimizar? (caso lineal)

$$\max_X Y = f(X)$$

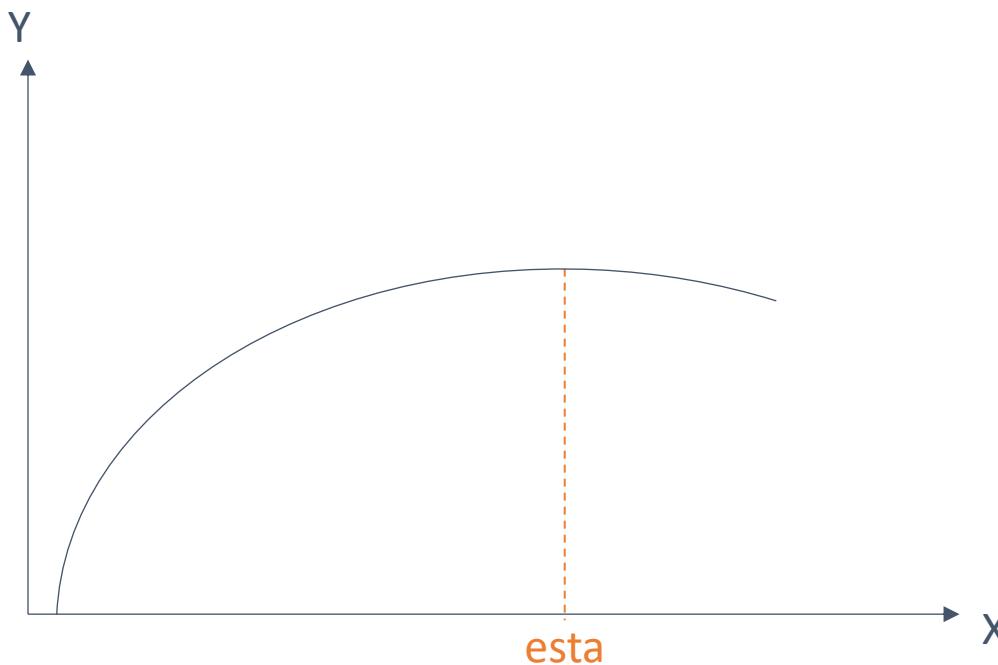


$$\max_X Y = f(X) \text{ s.a. } X \leq 5$$

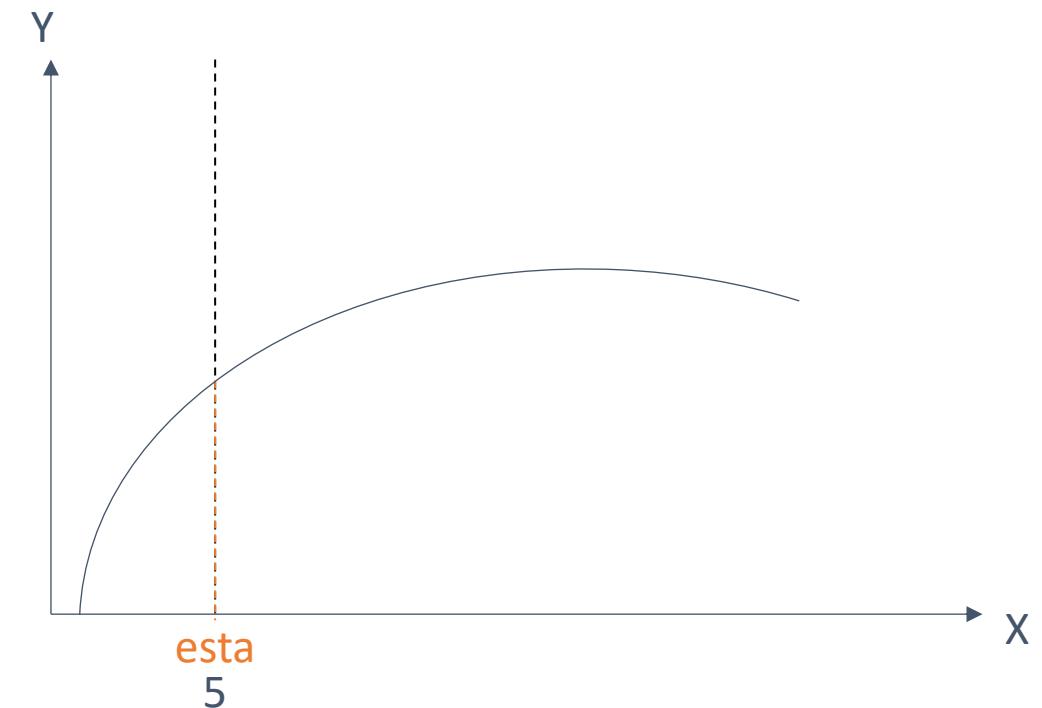


¿Cómo se ve optimizar? (caso no lineal)

$$\max_X Y = f(X)$$



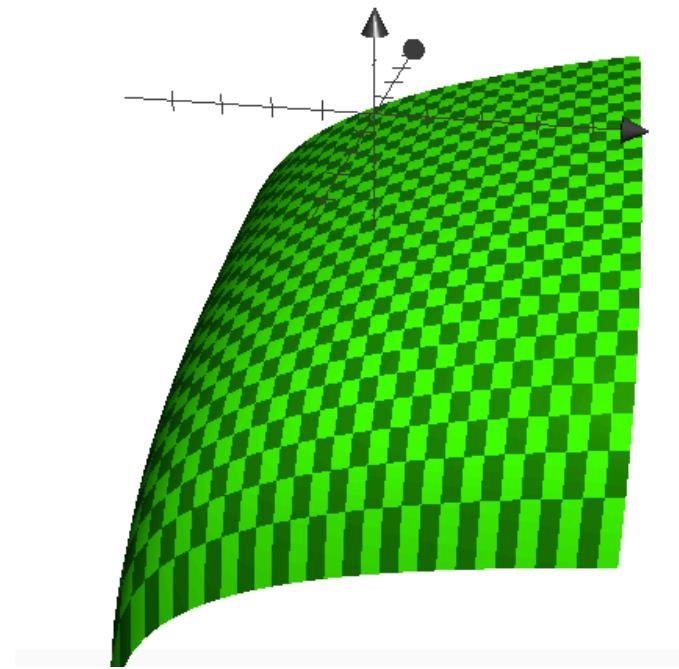
$$\max_X Y = f(X) \text{ s. a. } X \leq 5$$



¿Cómo se ve optimizar? (con 2 variables de decisión)

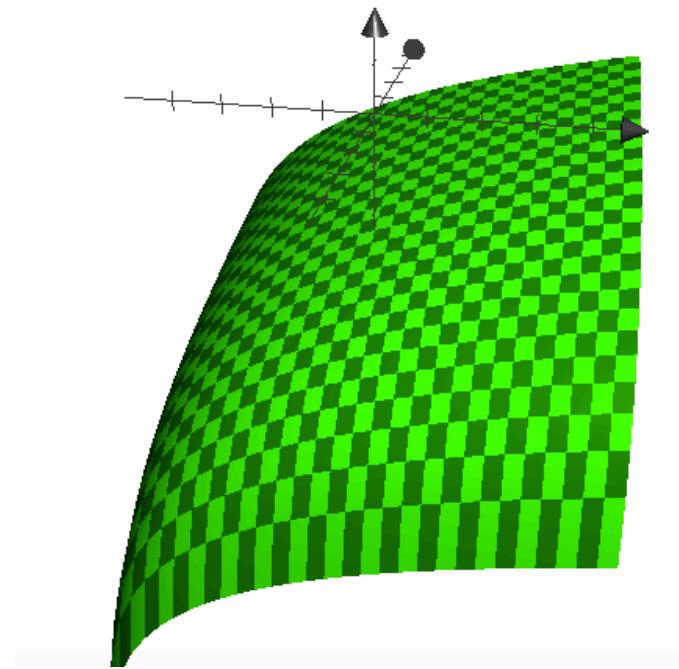
$$\max_{X,Y} Z = f(X, Y)$$

$$\max_{X,Y} Z = f(X, Y) \text{ s. a. } X + Y = 15$$

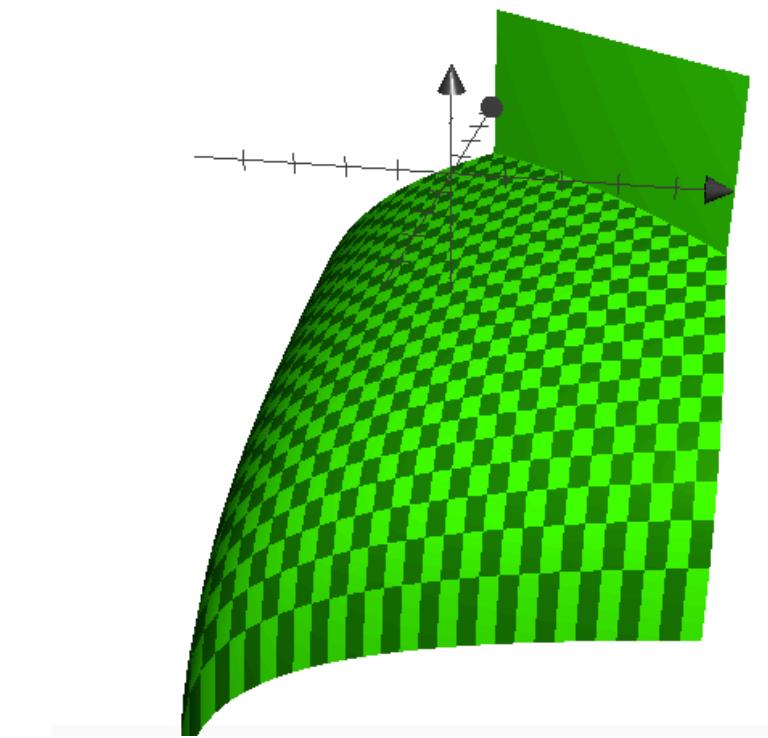


¿Cómo se ve optimizar? (con 2 variables de decisión)

$$\max_{X,Y} Z = f(X, Y)$$



$$\max_{X,Y} Z = f(X, Y) \text{ s. a. } X + Y = 15$$



Programación lineal y no lineal

- Se define el modelo que relaciona las variables de decisión con el objetivo
- Se escoge la combinación de variables de decisión que optimiza el objetivo

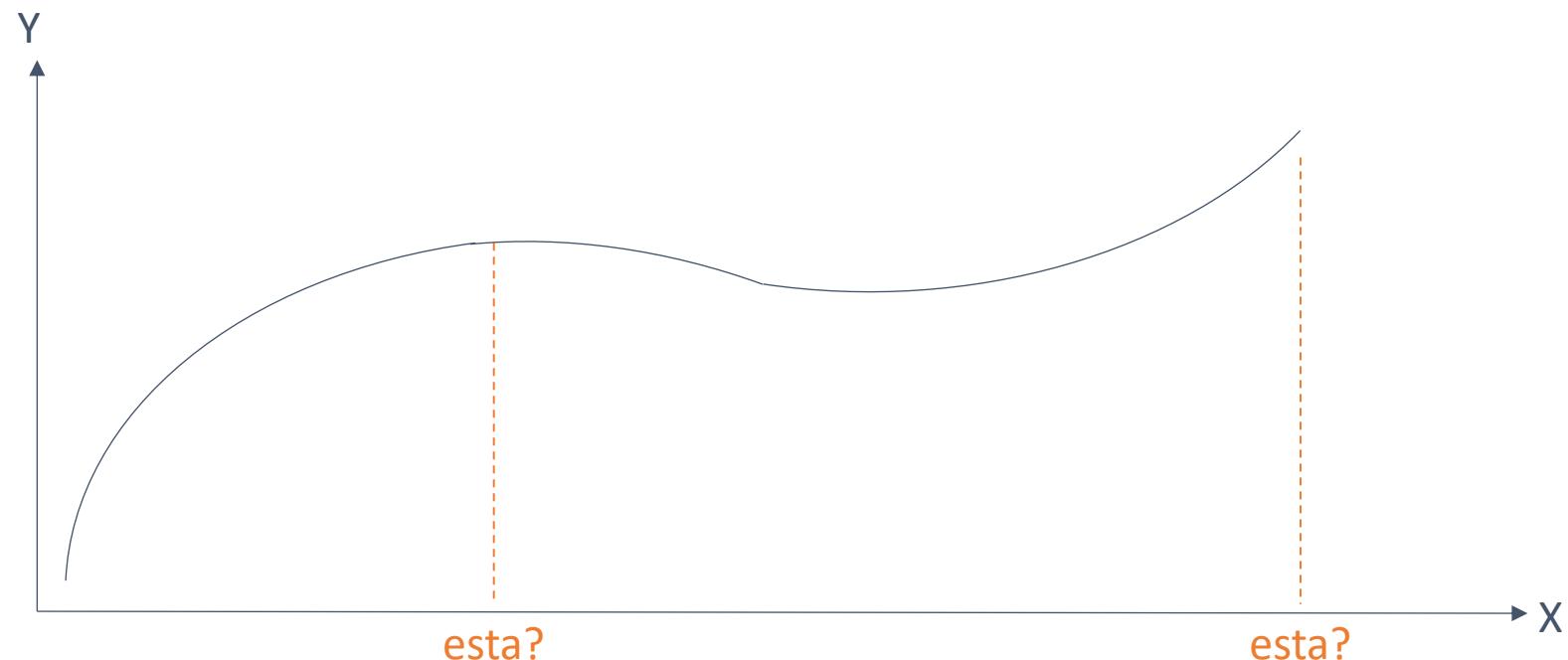
Note:

Puede haber múltiples soluciones.

Puede no haber solución (si dos restricciones son contradictorias)

Programación lineal y no lineal

Note: Cuando programamos no linealmente puede haber máximos locales pero probablemente esto ya lo consideran los paquetes de R



Programación lineal y no lineal

Programación lineal siempre va a ser muucho más rápida y eficiente computacionalmente que programación no lineal.

Si un problema se puede plantear o aproximar de forma lineal, es mejor hacerlo (así las relaciones no sean estrictamente lineales).





Caso tomado de https://flovv.github.io/From_descriptive_to_prescriptive/

Ejemplo de código

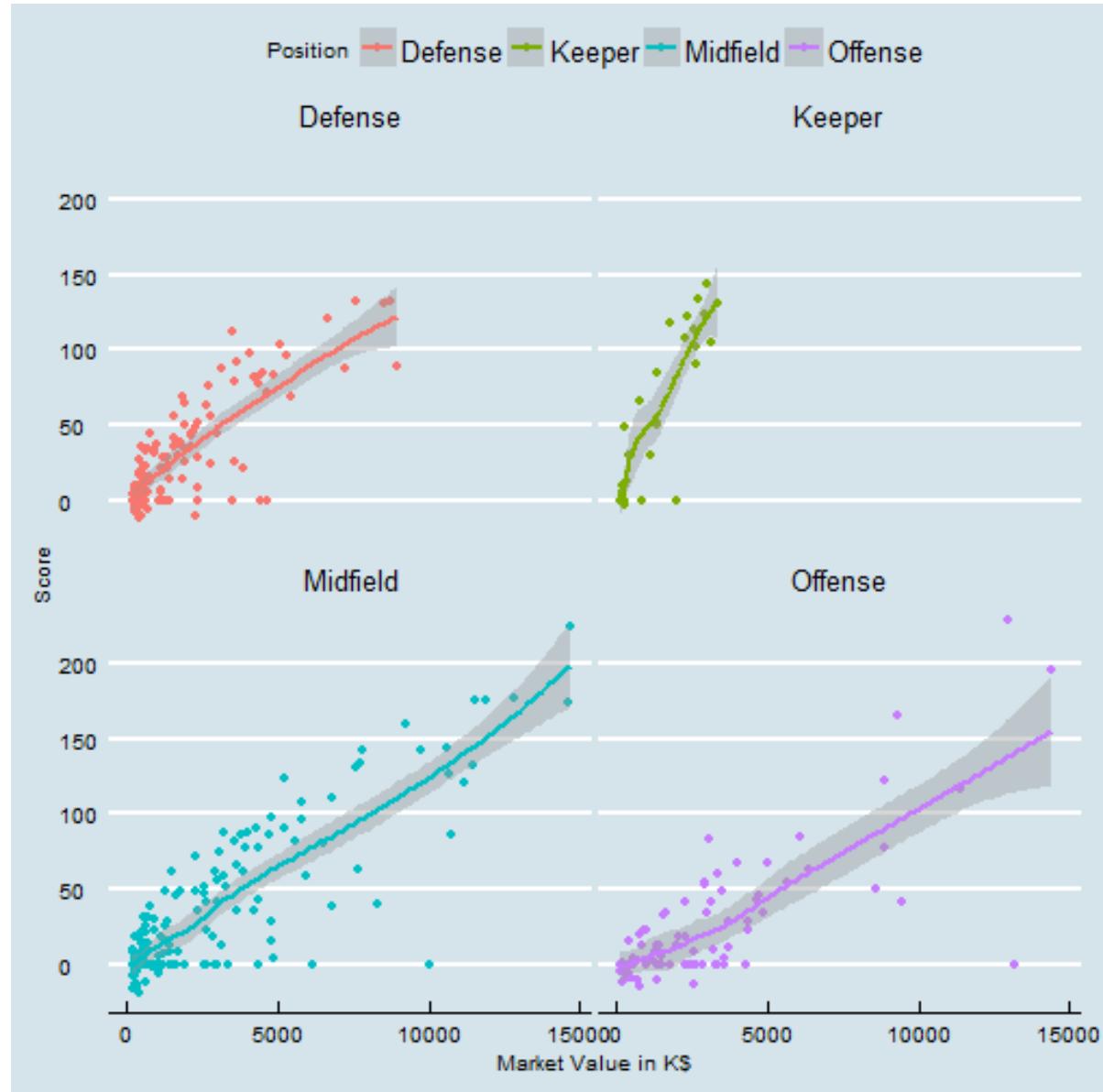
Ejemplo de analítica prescriptiva en R



Situación

Queremos configurar un equipo de jugadores de fútbol, manteniendo la restricción presupuestal del club.

Se muestran 488 posibles compras.



Un posible problema:

- **Objetivo:** maximizar el puntaje del equipo sujeto a:
 - Mantenerse en el presupuesto
 - El equipo debe tener exactamente un portero y un número dinámico de defensas, mediocampistas y delanteros:
 - Máximo: 1 portero, 5 defensas, 5 mediocampistas and 3 goleadores
 - El equipo total conste de 13 jugadores exactamente (por si se lesionan 2?)

Un posible problema:

- ¿Qué forma tiene la solución? (las variables de decisión):
De 1 a 488 variables binarias (1 para sí o 0 para no)

Jugador	Valor de mercado	Score/Puntaje	Posición
Alfredo	7000	103	Mediocampista
Julián	600	20	Defensa
Camilo	9000	160	Delantero

Insertando información: planteamiento

```
library(lpSolve)
f.obj <- simple$Score ### objetivo!
## Restricciones
#presupuesto 20 mill, 13 jugadores, 1 portero, 5 defensas, ... 3 goleadores
f.rhs <- c(20000000, 13, 1, 5, 5, 3)
# el presupuesto correspondiente debe ser menor que los definidos arriba,
# exactamente un portero
f.dir <- c("<=", "<=", "=", "<=", "<=", "<=")
```

Insertando información: restricciones

```
f.con <- t(simple$MarketValue) ### Restricción: máximo valor <= presupuesto  
player <- rep(1, nrow(simple)) ## Restricción: máximo número de jugadores!  
f.con <- rbind(f.con, player)
```

Insertando información: restricciones

```
f.con <- t(simple$MarketValue) ### Restricción: máximo valor <= presupuesto  
player <- rep(1, nrow(simple)) ## Restricción: máximo número de jugadores!  
f.con <- rbind(f.con, player)  
  
## restricción: por posición puede haber solo un cierto número de jugadores  
## (ej. Sólo un portero)  
## Definimos matriz - una variable categórica que indica la posición del jugador  
A <- as.data.frame(model.matrix(MarketValue ~ Position -1, simple) )  
f.con <- rbind(f.con, t(as.matrix(A)))
```

Solución del problema (lpSolve)

```
### resolver el problema
solved <- lp("max", f.obj, f.con, f.dir, f.rhs, all.bin=TRUE) ## simplemente
variables de decisión binarias!
#####
##### salida!
simple$buy <- solved$solution
```

Solución del problema (IpSolve)

```
sum(out[out$buy == 1,]$MarketValue) ## de cuánto quedó el presupuesto  
## [1] 19800000
```

```
sum(out[out$buy == 1,]$Score) ## cuál es el puntaje/Score  
## [1] 784
```

```
sum(out[out$buy == 1,]$buy) ## número de jugadores comprados  
## [1] 13
```

```
paste(out[out$buy == 1,]$Name, collapse=", ")  
## [1] "Badstuber, Baier, Balitsch, Ede, Gomez, Hasebe, Klavan, Krmas, Piszczeck,  
R. Schäfer, Soto, Svensson, Werner"
```

Herramientas

Para programar analítica prescriptiva en
R



Photo by [cottonbro](#) from [Pexels](#)

Algunas herramientas

Recapitulemos algunas herramientas que usamos en analítica prescriptiva:

- Herramientas para la descripción y visualización (ggplot, y paquetes similares)
- Herramientas para el modelado
 - Paquetes de aprendizaje de máquinas
 - Paquetes de regresión lineal
- Herramientas para la optimización
 - Paquetes para la programación lineal (lpSolve)
 - Otros paquetes de optimización (NLOpt)



Sistemas de desarrollo en equipo

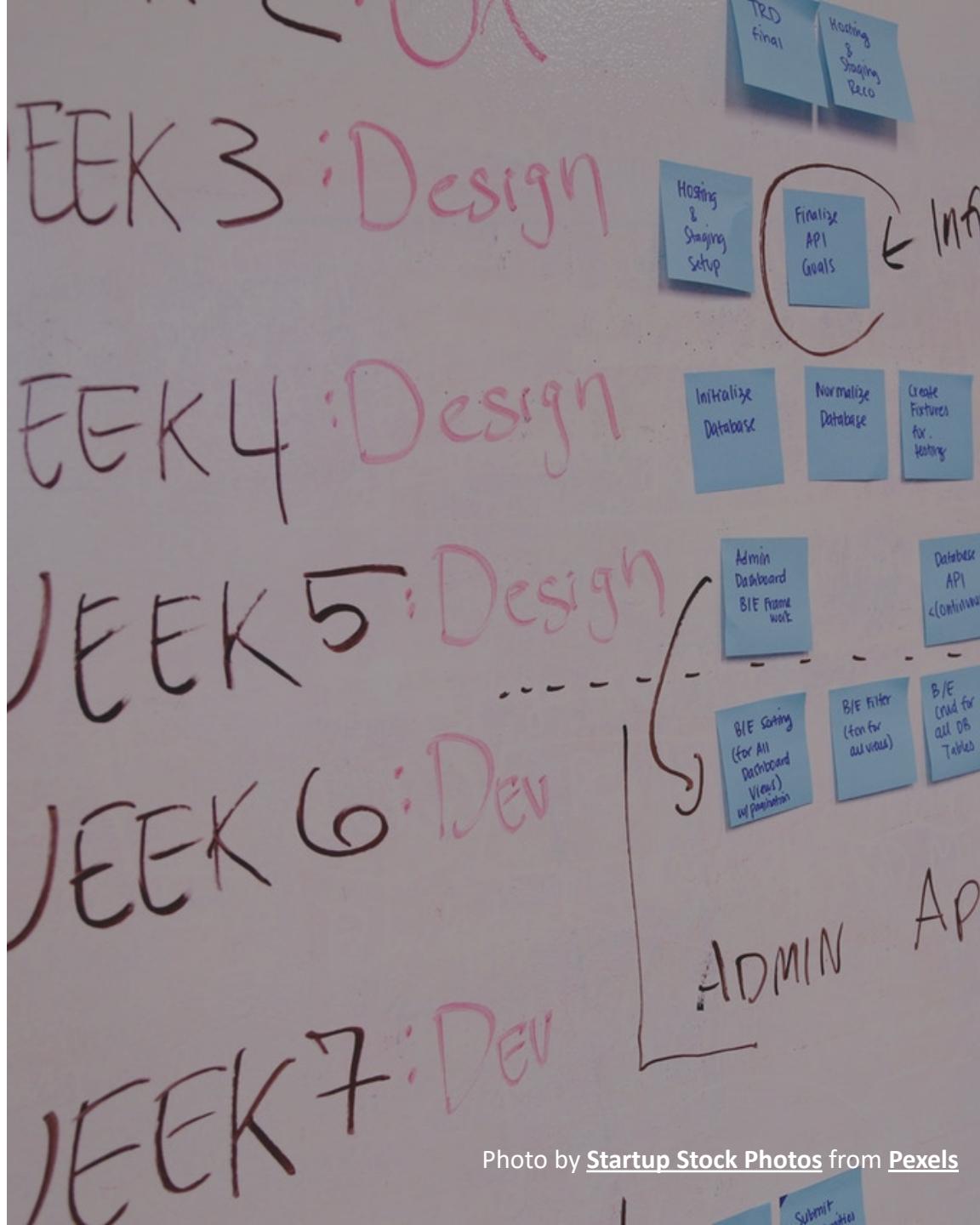
Al final, programar es un proyecto



Photo by [Canva Studio](#) from [Pexels](#)

Agile

Los del manifiesto filosófico



El manifiesto de Agile

En un proyecto es más importante:

- **Quiénes trabajan y cómo interactúan**, que los procesos y herramientas
- **Software que funcione**, que documentación detallada
- **Colaboración con los usuarios**, que la negociación de contratos
- **Responer a los cambios**, que seguir un plan
(aunque lo demás también es)



Los 12 principios de Agile

- Satisfacer al usuario con entregas continuas
- Acoger cambios en los requerimientos, así sean tardíos
- Entregar frecuentemente software que funciona en periodos más cortos de tiempo
- El equipo de desarrollo debe trabajar de la mano con los *stakeholders* (usuarios, directivos, etc.) a diario.
- Construir proyectos alrededor de individuos motivados
- Adquirir información conversando cara a cara



Los 12 principios de Agile

- Software que funciona es la principal medida de progreso
- Agile promueve el desarrollo sostenible en sus procesos
- Atención a la excelencia técnica y al buen diseño
- Simplicidad: maximizando el trabajo no hecho
- Las mejores arquitecturas y diseños surgen de equipos que se auto-organizan (en lugar de dirigidos por un tercero).
- El proyecto debe incluir reflexiones regulares acerca de cómo lograr mayor efectividad ¿Qué podemos estar haciendo mejor?



Scrum

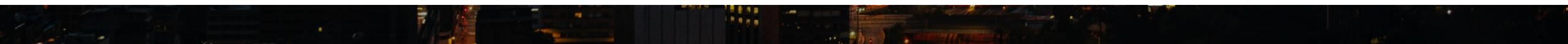
Un marco en la filosofía Agile



Photo by [Canva Studio](#) from [Pexels](#)

El flujo de trabajo: sprints

- En lugar de planear un proyecto a 6 meses, se plantean objetivos concretos y cortos que se desarrollan en períodos menores: ***sprints***
- ***Los sprints:***
 - Duran de 1 a 4 semanas
 - Incluyen reuniones de seguimiento todos los días
 - Al final de cada ciclo incluyen una reflexión retrospectiva
 - Requieren de retroalimentación rápida
 - Involucran colaboración continua con usuarios y *stakeholders*



Los roles del trabajo en Scrum

- Hay tres roles en un equipo Scrum.
- ***Team lead:*** brinda apoyo al equipo y pavimenta el camino hacia los objetivos. Ayuda al equipo a sobreponerse ante dificultades de recursos, contratiempos, etc.
- ***Team members:*** trabajan en el desarrollo de software. Pueden ser ingenieros, diseñadores, analistas, etc.
- ***Product owner:*** es responsable de la visión del proyecto y las prioridades.



Unos minutos para planear

Últimos minutos para dialogar



Photo by [Canva Studio](#) from [Pexels](#)

Pregunta:

Unos minutos para
conversar ideas

En la próxima clase también vamos a tratar de replicar la optimización lineal que vimos hoy.

- ¿Qué problema de decisión están pensando en abordar?
- ¿Con qué datos esperan contar?
- ¿Qué dudas y retos identifican en su problema de decisión?
- ¿Qué aprendizajes de las metodologías de desarrollo en equipo creen que pueden ser útiles en este proyecto?