**Regresión Lineal Múltiple**

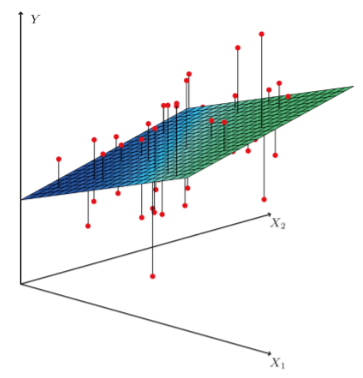
Vamos a usar **más de una variable predictora** para mejorar la performance del modelo.

Un camino sería hacer regresiones lineales simples por separado, cada una usando un medio como predictor. Pero, esto tiene desventajas: Se consiguen ecuaciones de regresión por separado, sin saber cómo hacer una única predicción a partir de todas las variables predictoras trabajando en conjunto. Cada una de las regresiones simples no tiene en cuenta a las demás variables predictoras al hacer la estimación. Si las variables predictoras están correlacionadas entre sí, esto puede llevar a estimaciones erróneas de los efectos individuales de cada una.

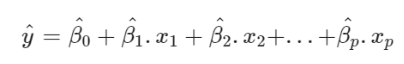
Un camino más conveniente es **extender** el **modelo de regresión simple** para que **incluya múltiples predictores**. Si tenemos p predictores distintos, el modelo de regresión lineal múltiple toma la siguiente forma:



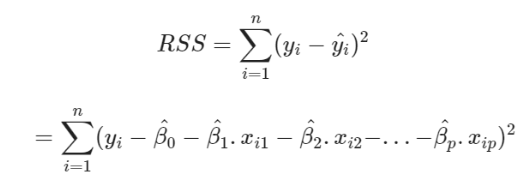
En el caso de que tuviéramos 2 variables predictoras y una respuesta buscada, en vez de una línea, lo que vamos a conseguir es un plano en la regresión mínimo cuadrática. Un plano tal que minimice la suma de cuadrados de las distancias entre cada observación y el plano:



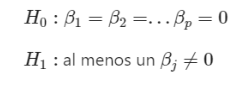
Si tenemos los estimadores de los coeficientes de pendiente, podemos pronosticar la variable de respuesta para una observación con valores dados de los predictores como:



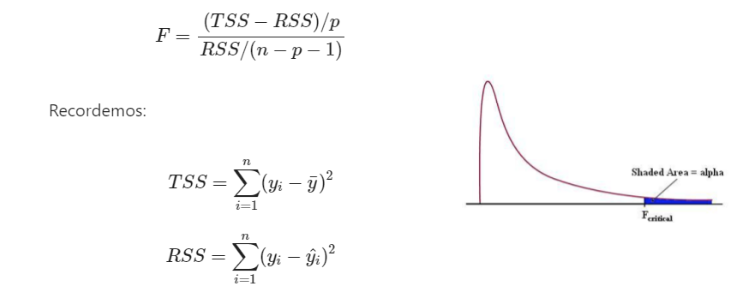
Vamos a elegir los valores para los estimadores de forma tal que minimicen la suma de residuos al cuadrado:



**Test de Hipótesis:** Hay alguna relación entre la variable objetivo y todas las variables explicativas?:



En este caso se usa el estadístico F:



Xj representa el predictor j; βj cuantifica la asociación entre la variable predictora j y la respuesta y. βj se interpreta como el **efecto promedio en y de un incremento unitario en Xj**, manteniendo el **resto de los predictores constantes**.

Ejemplo **en Python**, con el ejemplo de Advertising



**import** statsmodels **as** sm

feature\_cols = [“TV”, “Radio”, “Newspaper”]

X = advertising[feature\_cols]

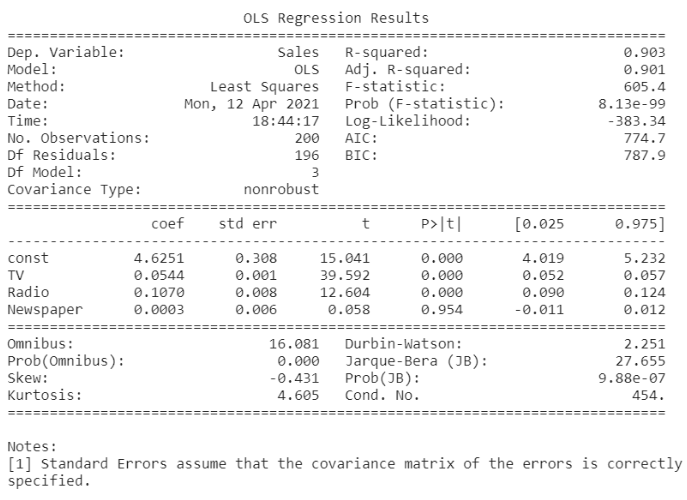
y = advertising.Sales

# Tenemos que agregar explícitamente una constante

X = sm.add\_constant(X)

model = sm.OLS(y, X).fit()

print(model.summary())



En **coef** tenemos los estimadores de coeficientes obtenidos por cuadrados mínimos para la regresión múltiple; que relacionan el número de unidades vendidas con el presupuesto de publicidad en TV, Radio y periódico.

Ahora vamos a hacer una regresión lineal simple entre Newspaper y Sales para contrastar los resultados:

X\_t = np.array(advertising.Newspaper, ndmin = 2)

X = np.transpose(X\_t)

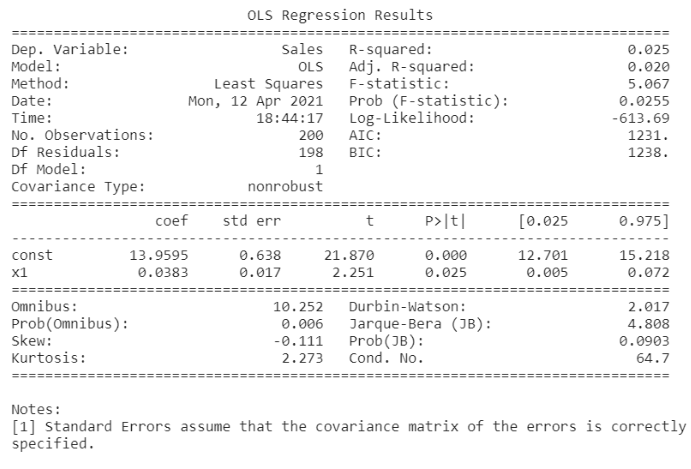
y = advertising.Sales

# Tenemos que agregar explícitamente una constante:

X = sm.add\_constant(X)

model = sm.OLS(y, X).fit()

print(model.summary())



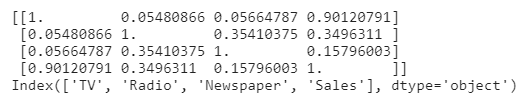
Con un nivel de significancia de 0.05, el estimador del coeficiente para Newspaper no es significativo en la regresión lineal múltiple, pero sí lo es en la regresión lineal simple.

Vamos a estudiar la matriz de correlaciones entre las 3 variables predictoras y la variable respuesta:

correlations = np.corrcoef(np.transpose(advertising))

print(correlations)

print(advertising.columns)



La correlación entre Radio y Newspaper es 0.35; en mercados donde gastamos más en radio, Sales tiende a valores más altos; y se tiende a gastar más en publicidad en Newspaper en mercados donde se gasta más en publicidad en Radio.

Si suponemos que la regresión múltiple es correcta, y la publicidad en Newspaper no tiene impacto directo en Sales, mientras que la publicidad en Radio sí lo tiene; al analizar la regresión lineal simple, vemos que a mayores valores de inversión en Newspaper obtenemos mayores valores de Ventas; incluso aunque la publicidad en Newspaper no afecta las ventas en Sales, según lo que nos dice la regresión múltiple. Esto puede deberse a que Radio y Newspaper están correlacionadas. Puede que la variable Newspaper se esté tomando el crédito de los efectos que la variable Radio tiene en Sales.

En un **escenario ideal**, los **predictores no** están **correlacionados.** Las correlaciones entre los predictores causan problemas, ya que la **varianza de los estimadores de los coeficientes** tiende a **aumentar**. Esto hace que los estimadores de los coeficientes sean **menos precisos.**

El **modelo lineal** implica **dos supuestos básicos** en la **relación entre cada predictor y el target**:

1. Es **aditiva**: el efecto del cambio de **Xj** sobre **Y** es aditivo a efectos de los demás predictores.
2. **Lineal:** El efecto del cambio de una unidad de **Xj** sobre **Y** es **constante**.

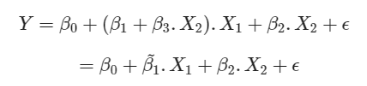
Hay modelos que relajan estos supuestos.

Hasta ahora veníamos suponiendo que el gasto en TV no dependía del gasto en publicidad en otros medios. Pero qué pasa si esto no es así? Si el gasto en TV incrementara también el efecto del gasto en Radio ()en marketing, *efecto sinergia*, en Estadística, ***interacción***, entonces tendría más sentido repartir el gasto entre ambos medios en vez de asignárselo sólo a uno.

El efecto interacción puede modelarse agregando un predictor en el modelo que sea el **producto de los predictores**: X1 \* X2:

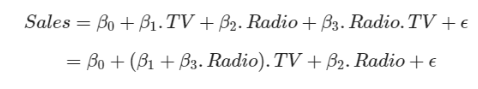


**Relajando** el **supuesto de aditividad**:



El efecto de X1 sobre Y ya no es constante, porque el cambio en una unidad de X1 sobre Y está afectado también por X2. De la misma manera, el efecto de X2 sobre Y tampoco eso constante, porque el cambio de una unidad de X2 sobre Y está afectado también por X1.

Volviendo **en Python** al ejemplo de Advertising, vamos a aplicar interacciones entre predictores:



Podemos interpretar β3 como el incremento de efectividad de la pauta en TV por cada unidad de incremento de la pauta en Radio (o viceversa). Lo que vamos a hacer es calcular la regresión lineal múltiple para Sales con las variables predictoras TV y Radio y ver cómo cambian los resultados en función de si modelamos con o sin interacción entre variables predictoras.

Sin Interacción:

feature\_cols = [“TV”, “Radio”]

X = advertising[feature\_cols]

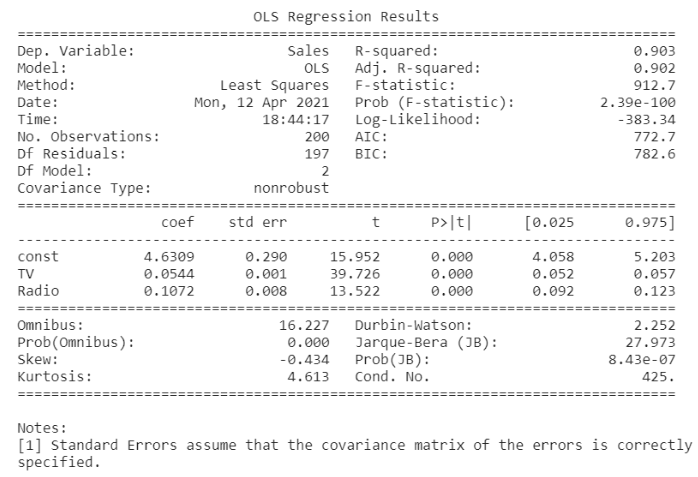
y = advertising.Sales

#Tenemos que agregar explíticamente una constante:

X = sm.add\_constant(X)

Model\_no\_interacción = sm. OLS(y, X).fit()

print(model\_no\_interaccion.summary())



Con Interacción:

Advertising[“TV\_x\_RADIO”]

feature\_cols = [“TV”, “Radio”, “TV\_x\_RADIO”]

X = advertising[feature\_cols]

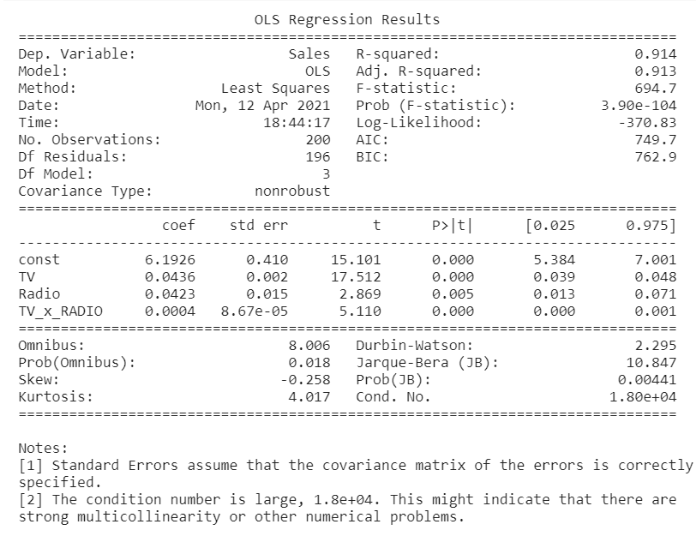
y = advertising.Sales

#Tenemos que agregar explíticamente una constante:

X = sm.add\_constant(X)

Model\_no\_interacción = sm. OLS(y, X).fit()

print(model\_no\_interaccion.summary())



El p-value para el término de interacción es muy bajo. Esto nos indica que tenemos evidencia para rechazar que β3 es igual a cero en la población. Obtuvimos un R2 de 91,4%. Es mejor que el R2 de 90,3% que obtuvimos del modelo sin interacción. El modelo con interacción no está dando, entonces, un mejor resultado.

También puede pasar que los términos principales no sean significativos, mientras que los términos de interacción sí lo sean. El **principio jerárquico** indica que aunque esto suceda, si incluimos interacciones en un modelo, también debemos incluir los efectos principales.

Es posible generar nuevas variables, elevando las existentes a alguna potencia:



En estos casos, el modelo sigue siendo líneal, porque es lineal en sus parámetros.

Ahora vamos a hacer la comparación entre un modelo lineal simple de TV con Sales vs un modelo de regresión lineal múltiple con TV elevado a alguna potencia.

Regresión Lineal Simple:

X\_t = np.array(advertising.TV, ndmin = 2)

X = np.transpose(X\_t)

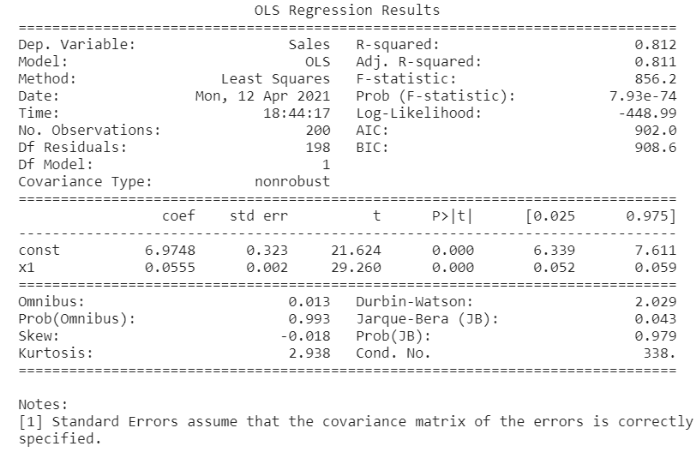
y = advertising.Sales

# Tenemos que agregar explícitamente una constante:

X = sm.add\_constant(X)

model = sm.OLS(y, X).fit()

print(model.summary())



Regresión Lineal Múltiple:

Advertising[“TV\_2”] = advertising.TV \* advertising.TV

Advertising[“TV\_3”] = advertising.TV \* advertising.TV\_2

Advertising[“TV\_4”] = advertising.TV \* advertising.TV\_3

Advertising[“TV\_5”] = advertising.TV \* advertising.TV\_4

feature\_cols = [“TV”, “TV\_2”, “TV\_3”, “TV\_4”, “TV\_5”]

X = advertising[feature\_cols]

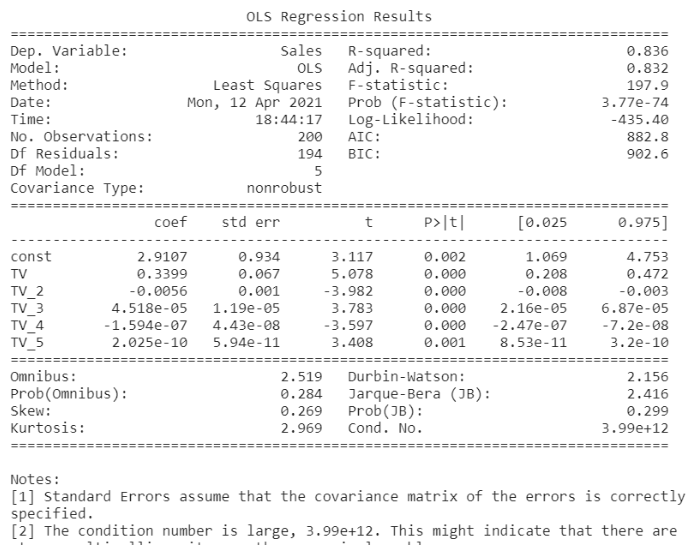
y = advertising.Sales

# Tenemos que agregar explícitamente una constante

X = sm.add\_constant(X)

model = sm.OLS(y, X).fit()

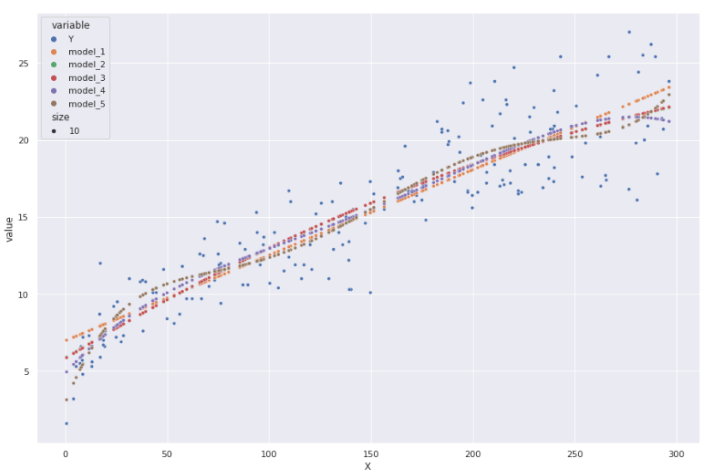
print(model.summary())



Vemos que el modelo con potencias de TV da un mejor R2 que el modelo de regresión lineal simple. Sin embargo, tenemos un warning: “The condition number is large. This might indicate that there are strong multicollinearity or other numerical problems” que indica que el modelo puede tener problemas de multicolinearidad. Los p-values obtenidos para cada coeficiente nos indican que tenemos evidencia para rechazar que los β sean cero. Ahora vamos a graficar los modelos:

sns.set(rc={”figure.figsize:(15,10)})

sns.scatterplot(data = data\_to\_plot, x= “X”, y=”value”, hue = “variable”, size= “size”)



**Evaluación de Modelos**

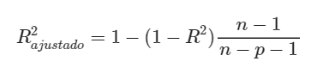
**R2** nos indica **qué tan bien se ajustan los datos** a una curva o línea.

**R2** **ajustado** nos indica **qué tan bien se ajustan los términos** a una curva o línea, pero este número **se corrige en función** de la **cantidad de variables predictoras**.

Al **agregar variables que no aportan valor** a la predicción, **R2 ajustado disminuirá**; al **agregar** **variables que aportan valor** a la predicción, **R2 ajustado aumentará**.

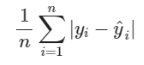
**R2 ajustado siempre <= R2**.

Al trabajar con modelos de regresión lineal múltiple, conviene evaluar la calidad del ajuste con R2 ajustado:

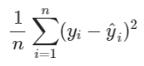


**Métricas de Evaluación para Problemas de Regresión:**

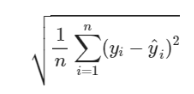
**MAE:** **Error Absoluto Medio.** Es la Media del Valor Absoluto de los Errores.



**MSE: Error Cuadrático Medio.** Es la Media de los Errores al Cuadrado.



**RMSE: Raiz del Error Cuadrático Medio**. Raíz Cuadrada de la Media de los Errores al Cuadrado.



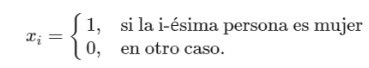
Estas 3 mediciones son **funciones de pérdida**. Queremos **minimizarlas**.

**Variables Dummy:**

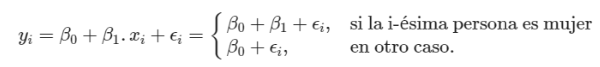
Podemos encontrarnos con problemas de regresión lineal que tienen **predictores cualitativos** (variables categóricas nominales u ordinales); tales como Género, Estado Civil, tipo de propiedad, etc.

Ejemplo: Queremos estimar la diferencia entre el saldo de la tarjeta de crédito entre hombres y mujeres (sin tener en cuenta el resto de las variables). Podemos hacer una regresión lineal simple con un predictor cualitativo.

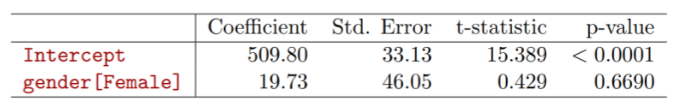
Vamos a definir una variable dummy x*i* tal que:



Entonces el modelo de regresión va a tomar la siguiente forma:



Si tenemos el siguiente resultado al modelo:



Podemos interpretar β0 como el saldo promedio de los hombres; y β0 + β1 como el saldo promedio de las mujeres. Entonces β1 nos indica la diferencia media entre el saldo de ambos grupos. Hombres (509.8); Mujeres (509.8 + 19.73 = 523.53). Pero, el p-value de β1 es muy alto. Por lo tanto, **pareciera no haber evidencia de una diferencia significativa** entre el saldo promedio de crédito de ambos géneros.

La decisión de codificación es arbitraria. Podríamos haber codificado al revés y hubiéramos obtenido un β0 de 523.53 y un β1 de -19.73.

**Normalización:**

Conviene **normalizar** cuando:

* Tenemos cantidades en **diferentes unidades o escalas.**
* Los **Algoritmos de Machine Learning** que usemos **lo requieren**.
* Es posible **aumentar la velocidad de convergencia**  usando el **Método del Gradiente.**

**Si los algoritmos** de Machine Learning que estemos usando **se basan en** el **cálculo de medidas de distancia**, **todas las variables predictoras intervienen**, representando cada una de las coordenadas de la observación. Por default se implementa la **distancia euclídea**; y esto implica que **todos los features (variables predictoras) deben ser numéricos**. La forma de lograr no favorecer ningún feature en particular es **estandarizar y deshacernos de las unidades**.

Cómo normalizar:

1. Mediante **estandarización: **

Tiene más sentido en los **casos donde importa** que los **features tengan las mismas unidades** **y la misma varianza**.Ejemplo: Análisis de Componentes principales.

1. Mediante **min-max**:  siendo min y max el valor mínimo y máximo de la variable predictora, respectivamente.

Tiene más sentido en los **casos donde importa** que los **features tengan las mismas unidades**, pero **no necesariamente la misma varianza**.

1. **L1** y **L2** (a ver en Regularización)

El Hands On de esta práctica es muy interesante.