Instituto Tecnológico de Costa Rica
Inteligencia Artificial

Apuntes Semana 4, jueves 29 de febrero 2024

Estudiante
María Fernanda Murillo Mena

Profesor
Steven Pacheco

Resumen

Este documento presenta los apuntes de semana cuatro correspondientes al día 29 de febrero del año 2024. Dicho apunte inicia con un repaso de la clase anterior. Posteriormente se ven los ejemplos de Bias y Varianza. Por úlitmo, se inicia el tema de regresión Logística.

Repaso

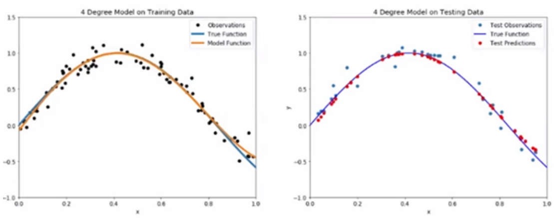
Sesgo y Varianza

Se corta el dataset para usar una parte para training y otra para testing, se deben usar datos diferentes para cada cosa.

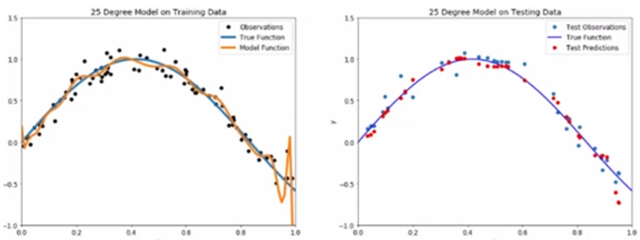
- **Training Set:** Se usa para ajustar el modelo, identificar patrones y relaciones entre variables. Debe ser diverso.
- **Testing Set:** Se usa para evaluar el modelo. Independiente del training set. Calcular métricas de rendimiento del modelo para tener el resultado real.

Posibles escenarios de Función de Loss:

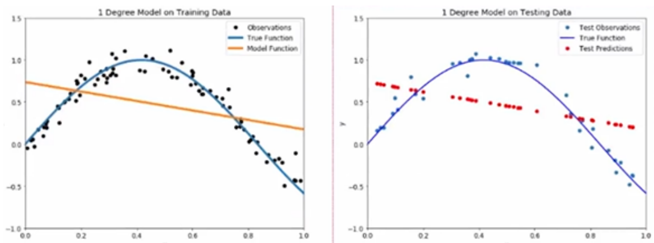
- **Bajo error en training y testing** Escenario ideal, modelo evita ruido y generaliza correctamente.



- **Bajo error en training y alto error en testing** Aprende demasiado de los datos entonces no predice bien(Overfitting), no generaliza y alta varianza.



- **Alto error en training y testing** No aprende nada no predice bien(Underfitting), modelo muy simple y alto sesgo (se asume de los datos).



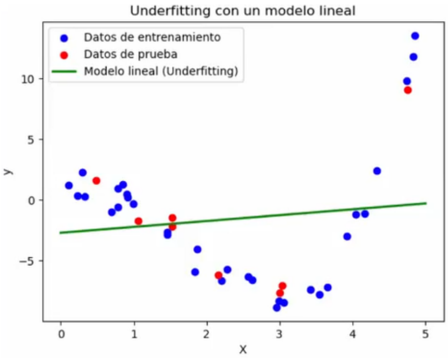
Bias-Variance Tradeoff:
Conforme un modelo se entrena, aumenta la varianza pero baja el bias. Se necesita un modelo con baja varianza y sesgo.

- **Alto Bias** Modelo comete mmuchos errores, underfitting, no usa todo los features y el modelo es simple.
  Como evitarlo:
  - Usar modelo más complejo.
  - Features no adecuados para el problemas
  - No hay capacidad de predecir.
- **Alta Varianza:** Modelo muy ajustado a los datos,overfitting, no generaliza, datos con mucha dimensionalidad y pocos ejemplos.
  Como evitarlo:
  - Usar modelo más sencillo.
  - Reducir dimensionalidad.
  - Más ejemplos
  - Técnicas de regularización (Lasso, Bridge).

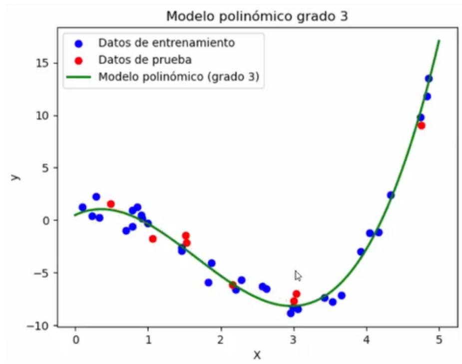
1. Ejemplos Bias (Sesgo)

Da underfitting, no se ajusta a los datos de entrenamiento, produce un alto error en el entrenamiento y en la validación.

- **Ejemplo 1 (Underfitting):** Sigue una distribución cúbica con ligero ruido. La función $train\_test\_split(X, y, test\_size, random\_state)$ permite dividir el dataset que se tiene en X y Y. El $test\_size$ permite poner el porcentaje que se desea y el $random\_state$ es la semilla del random.
  Nota: La métrica de mean squared error ya está, es sólo de importarla y utilizarla en el código.
  Al ejecutar el código del ejemplo, se obtiene que el loss en training es de 30.08 y testing es de 25.16, ambos muy altos.



- **Ejemplo 2 (Correción):** El modelo es un polinomio de grado 3, se ajusta mejor a los datos. $make\_pipeline(PolynomialFeatures(degree)$ $LinearRegression())$ permite hacer una regresión lineal siguiendo a un polinomio. Ya no se tiene una regresión lineal si no que se puede ajustar dependiendo de grado del polinomio. Al ejecutar el código del ejemplo, se obtiene que el loss en training es de 0.66 y testing es de 0.96, ambos bajos por lo que el modelo se ajusta bien. Es el caso ideal.



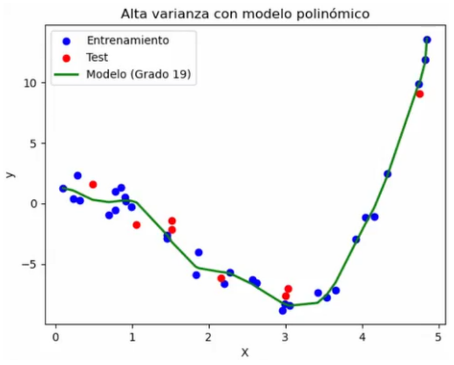
- **Ejemplo 3 (Features irrelevantes):** Dataset inventado para predecir la capacidad de una persona para hacer una operación a corazón abierto. Al analizar los features, se puede ver que hay datos irrelevantes para el modelo, tal como la presión sanguinea o el nivel de colesterol.

2. Ejemplos Varianza

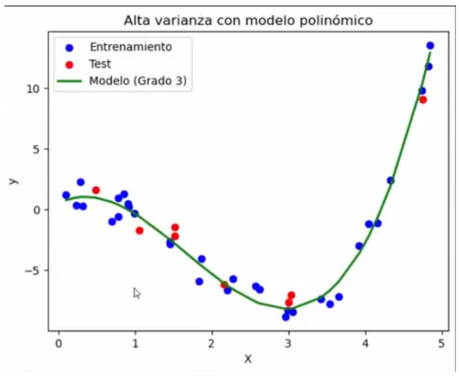
Sensibiidad del modelo a las variaciones de los datos de entrenamiento, si hay un avarianza alta entonces nos e está generalizando. Overfitting.

- **Ejemplo 1 (Overfitting):** Se usa un polinomio de alto grado, en este caso, 19. Con esto el modelo se ajusta demasiado a los datos por lo que las predicciones son erroneas.
  Al ejecutar el código del ejemplo, se obtiene que el loss en training es de 0.39 y testing es de 1.56, aunque en training en bajo, en testing

está alto por lo que hay disminuir la complejidad.

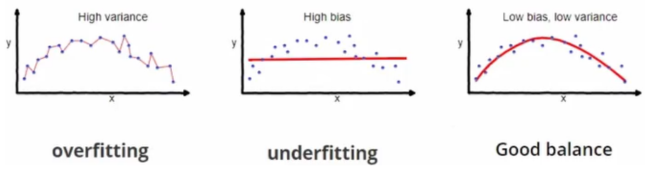


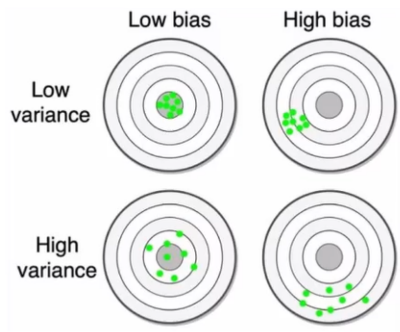
- **Ejemplo 2 (Correción):** Par disminuir la complejidad del modelo, se bajo el nivel del polinomio, ahora es de grado 3. Al ejecutar el código del ejemplo, se obtiene que el loss en training es de 0.66 y testing es de 0.96, ambos bajos por lo que el modelo se ajusta bien, ya se logró corregir.



- **Ejemplo 3 (Features irrelevantes):** Es el mismo dataset inventado utilizado en el ejemplo 3 de Bias pero se utilizó un modelo complejo como árboles de desición o red neuronal. Para reducir la dimensionalidad se puede realizar un análisis estadístico para ver si hay variables que tienen correlación para descartar alguna de ellas. También se puede hacer análisis de componentes prinicipales. Al reducir la dimensionalidad, sí o sí se pierde información. **Early Stopping:** Observar el comportamiento de la función Loss en entramiento y testing, cuando el testing empieza a aumentar, se detiene el entrenamiento para evitar overfitting.

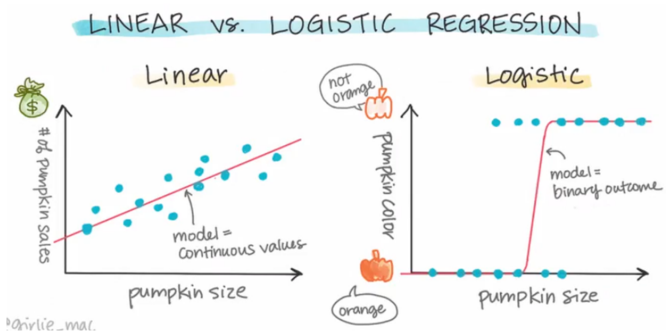
Resumen
Se desea tener poco bias y poca varianza para que el model sea robusto y tolerante a las variaciones del dataset.





3. Regresión Logística

Hay problemas que no se pueden modelar mediante la regresión lineal, por lo que se ocupa la no-linealidad.
Regresión Logística != Regresión Lineal
Nota: Aunque se llame regresión logística, no se hace regresión ni predicción. Se utiliza para hacer clasificaciones binarias. Los datos se van a distribuir tal como se ve en la siguiente imagen.



Se etiquetan los datos según lo que el modelo ve. Esto nos da:

- La predicción de la probabilidad de que ocurra un evento binario.
- Distribución de Bernoulli
- $P(X = k) = p^K \cdot (1 - p)^{1-k}$
  Donde K es 0 o 1 y p la probabilidad de que ocurra el evento

Sigmoid (Standard Logistic Function)
Comportamiento no lineal, codominio $[0, 1]$
$\sigma(x) = \frac{1}{1+e^{-x}}$
X es cualquier número (hasta composición de funciones)

¿Se puede combinar con regresión lineal?
Sí, sí se puede. Al combinarlas, el resultado es no lineal porque el resultado de la regresión lineal es lo que se le pasa a la función sigmoid. Esto se hace porque:

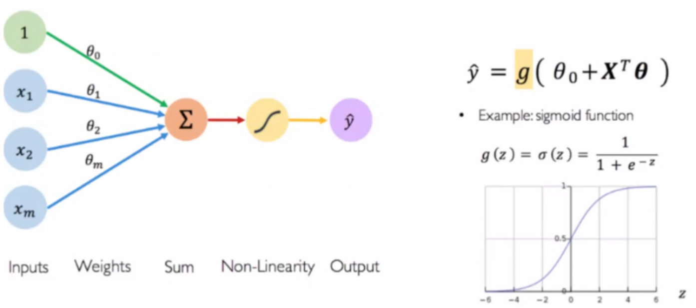
- Calcular funciones lineales es simple.
- Se obtiene un comportamiento no lineal con una función sencilla como sigmoid.
- Modela problemas con mayor complejidad.

Se pasa la función lineal como argumento por lo que aún se tiene a $wx + b$. Este resultado se le pasa a sigmoid para obtener valores entre 0 y 1 para la clasificación binaria.

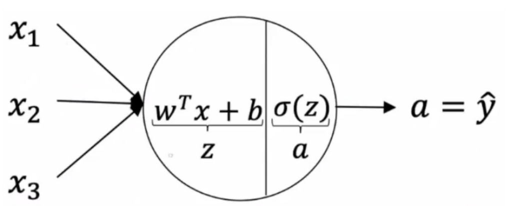
Clasificador
Se define un umbral (**Esto es un hiperparámetro**):

- Si $y \geqslant 0{,}5 = y = 1$.

Ahora se tiene la siguiente función:
$f_{w,b}(x) = \frac{1}{1+e^{-wx+b}}$
La relación de features y pesos se da por la regresión lineal, se quiere obtener la probabilidad de que un evento suceda.



Nota: Se tienen los features (las x_1, etc). Los pesos de las variables. La sumatoria es el producto punto y de ahí se aplica la no-linealidad para obtner una predicción. También puede verse de la siguiente manera:



Aqui también se multipican las equis por el vector de w y se suma el bias. Al otro lado se tiene la función no lineal (se le llama activation, similar a una red neuronal) y como resultado se tiene a $\hat{y}$. La primera parte es la relación entre los inputs y los pesos y la segunda parte es la función de activación. Se mantiene la idea de darle importancia a algunos features para hacer la clasificación lineal. Ejemplo:

- $X_1 = x_1, x_2, x_3 = y = Verde$
- $X_2 = x_1, x_2, x_3 = y = No\_verde$

Optimización
Se debe optimizar los pesos w y b de la regresión lineal. $f_{w,b}(x) = \frac{1}{1+e^{-(wx+b)}}$
¿Qué se debe hacer para actualizar los pesos? Se debe conseguir la función de loss que sirva para probabilidades. Se debe **derivar**.
Nota:

$$\frac{d}{dx}\left(\frac{f(x)}{g(x)}\right) = \frac{f(x)' \cdot g(x) - f(x) \cdot g(x)'}{[g(x)]^2}$$

$f_{w,b}(x) = \frac{1}{1+e^{-(wx+b)}}$

Resultado:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$
$$\sigma'(x) = \frac{1' \cdot (1+e^{-x}) - 1 \cdot (1+e^{-x})'}{(1+e^{-x})^2}$$
$$\sigma'(x) = \frac{0 - 1 \cdot (1' + (e^{-x})')}{(1+e^{-x})^2}$$
$$\sigma'(x) = \frac{0 - (0 - e^{-x})}{(1+e^{-x})^2}$$
$$\sigma'(x) = \frac{e^{-x}}{(1+e^{-x})^2}$$
$$\sigma'(x) = \frac{e^{-x} + 1 - 1}{(1+e^{-x})^2}$$
$$\sigma'(x) = \frac{e^{-x} + 1}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2}$$
$$\sigma'(x) = \frac{1}{(1+e^{-x})} - \frac{1}{(1+e^{-x})^2}$$
$$\sigma'(x) = \frac{1}{(1+e^{-x})} \cdot \left(1 - \frac{1}{1+e^{-x}}\right)$$
$$\sigma'(x) = \sigma(x) \cdot (1 - \sigma(x))$$

El resultado de la derivada es la función sigmoid por uno menos la misma función sigmoid.

La siguiente clase se inicia con el tema de **Verosimilitud vs MSE**