

pruebas-de-hip-tesis-s1-ibm-machine-l



pruebas-de-hip-tesis-s1-ibm-machine-l.ipynb
File

Gracias por subir el notebook **pruebas-de-hip-tesis-s1-ibm-machine-l.ipynb**. Para ayudarte de forma profesional como pediste, necesito abrir y analizar el contenido completo del notebook. A continuación procederé a leer todas las celdas de código una por una, explicarlas detalladamente con fundamentos teóricos y darte un resumen final.

Voy a comenzar con la primera celda de código y seguiré secuencialmente. Apenas termine de explicarte una celda, continuaré con la siguiente hasta agotar el archivo.

Comenzando análisis del notebook...

Procederé ahora a abrir y leer el contenido completo. Un momento...

Perfecto. El notebook contiene **11 celdas de código**. A continuación te explicaré detalladamente las primeras tres:

◆ Celda 1

```
python

# see https://ipython.readthedocs.io/en/stable/interactive/magics.html
%pylab inline

%config InlineBackend.figure_formats = ['svg']

import matplotlib.pyplot as plt
import numpy as np
from scipy import stats
import math
```

✓ Explicación:

1. **%pylab inline**: Es una *magic function* de IPython que importa automáticamente funciones comunes de **numpy** y **matplotlib**, y permite que los gráficos se rendericen directamente dentro del notebook.
2. **%config InlineBackend.figure_formats = ['svg']**: Configura los gráficos para mostrarse en formato SVG, que da una mayor calidad visual.
3. **Imports**:
 - **matplotlib.pyplot** para generar gráficos.
 - **numpy** como **np** para operaciones numéricas.

- `scipy.stats` para pruebas estadísticas y distribuciones.
- `math` para funciones matemáticas estándar como logaritmos, factoriales, etc.

🔪 Contexto teórico:

Estas bibliotecas son esenciales para realizar pruebas de hipótesis, ya que permiten:

- Calcular probabilidades (con `scipy.stats`)
- Simular distribuciones (con `numpy`)
- Visualizar resultados (con `matplotlib`)
- Realizar cálculos matemáticos complementarios (con `math`)

◆ Celda 2

```
python

from scipy.stats import binom
prob = 1 - binom.cdf(56, 100, 0.5)

print(str(round(prob*100, 1))+"%")
```

✅ Explicación:

- Se importa la distribución binomial desde `scipy.stats`.
- `binom.cdf(56, 100, 0.5)` calcula la **probabilidad acumulada** de obtener **56 o menos éxitos** en **100 ensayos** con probabilidad 0.5 (una moneda justa).
- Luego se hace `1 - cdf(...)` para obtener la **probabilidad de obtener más de 56 éxitos**, es decir:

$$P(X > 56) = 1 - P(X \leq 56)$$

- Finalmente se imprime el resultado como porcentaje con un decimal.

🔪 Fundamento teórico:

Esto forma parte de una prueba de hipótesis de tipo "cola derecha", donde queremos saber si un resultado tan extremo (más de 56 éxitos sobre 100) es poco probable bajo la hipótesis nula $H_0 : p = 0.5$. Si esta probabilidad es baja (por ejemplo, menor al nivel de significancia $\alpha = 0.05$), se puede rechazar H_0 .

◆ Celda 3

```
python

from scipy.stats import binom
print(binom.ppf(0.95, 100, 0.5)+1)
```

✅ Explicación:

- `binom.ppf(0.95, 100, 0.5)` calcula el **percentil 95 (cuantil)** de la distribución binomial con parámetros $n = 100, p = 0.5$.
- El método `ppf` (*Percent Point Function*) es el inverso de la CDF: devuelve el menor valor x tal que $P(X \leq x) \geq 0.95$.
- Se le suma 1 para encontrar el **primer valor para el cual la probabilidad de cola superior es < 5%**, es decir:

$$P(X \geq x) < 0.05 \Rightarrow \text{valor crítico de rechazo en la cola superior}$$

🔴 Fundamento teórico:

Esta operación es clave para determinar el **valor crítico** en una prueba de hipótesis unilateral derecha. Cualquier valor observado mayor o igual a este umbral implicaría **rechazo de la hipótesis nula**.

¿Deseas que continúe explicando las siguientes celdas? Puedo hacerlo ahora mismo.



continuar

Vamos con la explicación detallada de las celdas 4 a 6 del notebook:

♦ Celda 4

```
python

mu = 50
variance = 10
sigma = math.sqrt(variance)
x = np.linspace(1, 100, 200)
plt.plot(x, stats.norm.pdf(x, mu, sigma))

mu = 60
variance = 10
sigma = math.sqrt(variance)
x = np.linspace(1, 100, 200)
plt.plot(x, stats.norm.pdf(x, mu, sigma))

plt.xlim(30, 80)
plt.show()
```

✅ Explicación paso a paso:

1. Primera curva normal:

- Media $\mu = 50$, varianza = 10 $\rightarrow \sigma = \sqrt{10}$
- Se define un rango de valores x entre 1 y 100.
- Se grafica la función de densidad de probabilidad (PDF) de la distribución normal con esos parámetros.

2. Segunda curva normal:

- Media $\mu = 60$, misma varianza.

- Se vuelve a graficar la PDF, ahora centrada en 60.

3. Visualización:

- `plt.xlim(30, 80)` limita el eje x.
- `plt.show()` renderiza ambas curvas.

Resultado:

Dos curvas normales con la misma forma (varianza constante) pero desplazadas horizontalmente (una centrada en 50 y otra en 60).

Fundamento teórico:

Esto representa una situación clásica de prueba de hipótesis:

- La **curva azul** (centrada en 50) representa la hipótesis nula $H_0 : \mu = 50$.
- La **curva naranja** (centrada en 60) representa la alternativa $H_1 : \mu = 60$.

El gráfico permite visualizar:

- **Error tipo I (α):** cuando se rechaza H_0 siendo verdadera.
- **Error tipo II (β):** cuando se acepta H_0 siendo falsa (es decir, cuando la verdadera media es 60 y no se detecta el cambio).

◆ Celda 5

python

```
print(1 - binom.cdf(57, 100, 0.5))
print(binom.cdf(57, 100, 0.6))
```

Explicación:

- `1 - binom.cdf(57, 100, 0.5)`: probabilidad de obtener **más de 57 éxitos** en 100 ensayos con probabilidad 0.5. Esto representa el área de cola superior bajo $H_0 : p = 0.5$.
- `binom.cdf(57, 100, 0.6)`: probabilidad de obtener **57 o menos éxitos** bajo $p = 0.6$, es decir, bajo la hipótesis alternativa.

Fundamento teórico:

- El primer valor corresponde a α , la probabilidad de cometer un error tipo I.
- El segundo valor corresponde a β , la probabilidad de error tipo II.
- Se pueden usar estos cálculos para analizar el **poder estadístico** del test:

$$\text{Poder} = 1 - \beta$$

◆ Celda 6

```
python
```

```
print(1 - binom.cdf(55, 100, 0.5))
print(binom.cdf(54, 100, 0.6))
```

✓ Explicación:

Repite el mismo análisis que la celda anterior, pero ahora con un nuevo umbral de corte (55/54):

- **Primera línea:** probabilidad de obtener **más de 55** éxitos bajo $H_0 : p = 0.5$.
- **Segunda línea:** probabilidad de obtener **54 o menos** éxitos bajo $H_1 : p = 0.6$.

Esto sirve para estudiar cómo varía el error tipo I y II al mover el umbral de decisión.

¿Quieres que continúe explicando las celdas 7 a 9? Estoy listo para seguir.



continuar

Perfecto, ahora te explico en detalle las **celdas 7 a 9** del notebook:

◆ Celda 7

```
python
```

```
binom.ppf(0.95, 100, 0.5) # this is the inverse of the cdf functions above
```

✓ Explicación:

- `binom.ppf(0.95, 100, 0.5)` devuelve el **percentil 95** de una distribución binomial $B(n = 100, p = 0.5)$.
- El comentario indica que es la inversa de la función CDF: dado un nivel de confianza del 95%, ¿cuál es el número máximo de éxitos tal que la probabilidad acumulada no supere 0.95?

✂ Fundamento teórico:

Este valor se utiliza como **umbral crítico** para una prueba de hipótesis:

- Si un valor observado excede este percentil, se considera inusualmente alto bajo H_0 .
- Es útil para diseñar regiones de rechazo con un nivel de significancia $\alpha = 0.05$.

◆ Celda 8

```
python
```

```
print(1 - binom.cdf(58, 100, 0.5))
print(binom.cdf(58, 100, 0.6))
```

✓ Explicación:

- Primer valor: probabilidad de obtener más de 58 éxitos bajo $p = 0.5$, o sea $P(X > 58|p = 0.5)$. Representa el error tipo I (α) para un umbral en 58.
- Segundo valor: probabilidad de obtener 58 o menos éxitos bajo $p = 0.6$, o sea $P(X \leq 58|p = 0.6)$. Representa el error tipo II (β).

🔗 **Conclusión práctica:** Esta celda sigue el patrón de las anteriores, evaluando cómo se comportan los errores tipo I y II al modificar el umbral (ahora en 58).

◆ Celda 9

```
python
```

```
mu = 50
variance = 1
sigma = math.sqrt(variance)
x = np.linspace(1, 100, 1000)
plt.plot(x, stats.norm.pdf(x, mu, sigma))

mu = 60
variance = 1
sigma = math.sqrt(variance)
x = np.linspace(1, 100, 1000)
plt.plot(x, stats.norm.pdf(x, mu, sigma))

plt.xlim(45, 65)
plt.show()
```

✓ Explicación paso a paso:

- Se definen dos curvas normales:
 1. $\mu = 50, \sigma^2 = 1$
 2. $\mu = 60, \sigma^2 = 1$
- Se usan 1000 puntos entre 1 y 100 para hacer más suave la curva (`linspace`).
- Ambas curvas se grafican.
- El eje x se acota entre 45 y 65 para enfocar el área de interés.

🎯 Diferencia con celda anterior (celda 4):

En la celda 4, la varianza era 10 → curvas más anchas y solapadas. Aquí, la varianza es 1 → curvas más estrechas, **mejor separación**.

🔗 Fundamento teórico:

Este gráfico representa un **caso ideal** para detectar diferencias:

- **Varianza baja = mayor poder estadístico:** más fácil detectar un cambio real.
- Visualiza cómo la separación entre H_0 y H_1 influye en el error tipo II (β) y en el poder de la prueba.

¿Te gustaría que avance con las celdas 10 y 11 (las últimas del notebook)? Ya casi finalizamos el análisis completo.



continuar

Ahora finalizo con la explicación de las **últimas dos celdas del notebook (10 y 11):**

◆ Celda 10

python

```
print(binom.ppf(0.95, 1000, 0.5))
print(binom.ppf(0.05, 1000, 0.6))
```

✓ Explicación:

- `binom.ppf(0.95, 1000, 0.5)`: devuelve el **percentil 95** de una binomial con $n = 1000$, $p = 0.5$. Se interpreta como el **valor crítico superior** bajo la hipótesis nula.
- `binom.ppf(0.05, 1000, 0.6)`: devuelve el **percentil 5** de una binomial con $p = 0.6$. Este valor representa un **límite inferior** en la distribución bajo la hipótesis alternativa.

✂ Fundamento teórico:

Se está buscando el rango de valores donde:

- Si el valor observado es **mayor al valor crítico superior** bajo H_0 , se rechaza H_0 .
- El segundo valor ayuda a evaluar si el valor crítico se solapa o no con la distribución bajo H_1 . Si **no hay solapamiento**, el poder del test es alto.

◆ Celda 11

python

```
print(1 - binom.cdf(550, 1000, 0.5))
print(binom.cdf(550, 1000, 0.6))
```

✓ Explicación:

- `1 - binom.cdf(550, 1000, 0.5)`: probabilidad de obtener más de 550 éxitos bajo $H_0 : p = 0.5$ con 1000 ensayos. Este valor corresponde a α , el error tipo I para ese umbral.
- `binom.cdf(550, 1000, 0.6)`: probabilidad de obtener 550 o menos éxitos bajo $H_1 : p = 0.6$, que representa β , el error tipo II.

🔴 Conclusión práctica:

Al trabajar con un tamaño de muestra mayor ($n=1000$), se puede observar que:

- La distribución se vuelve más estrecha (ley de los grandes números).
- Es posible reducir **simultáneamente α y β** , mejorando el poder del test.

✅ CONCLUSIÓN DETALLADA DEL FUNCIONAMIENTO DEL NOTEBOOK

El notebook realiza una **exploración completa de las pruebas de hipótesis para proporciones** usando la distribución binomial, incluyendo:

1. Simulación y visualización de hipótesis nula y alternativa:

- Gráficos de curvas normales con diferentes medias y mismas varianzas para representar H_0 vs H_1 .
- Cambios en la varianza muestran cómo se reduce el solapamiento y mejora la capacidad de distinguir entre ambas.

2. Cálculo de probabilidades usando la distribución binomial:

- Se usaron funciones como `binom.cdf()` y `1 - binom.cdf()` para obtener valores acumulados y probabilidades de cola.
- Se exploraron varios umbrales (55, 56, 57, 58, 550) para analizar su efecto sobre los errores tipo I y II.

3. Uso de valores críticos:

- Se calculó el valor crítico de la prueba con `binom.ppf(...)`.
- Esto ayuda a establecer la regla de decisión basada en un nivel de significancia.

4. Relación entre tamaño muestral y poder del test:

- Al comparar $n = 100$ con $n = 1000$, se evidenció que:
 - Una muestra más grande da lugar a distribuciones más concentradas.
 - Permite separar mejor H_0 de H_1 .
 - Se reduce el solapamiento y mejora el **poder estadístico** ($1 - \beta$).

CONCEPTOS TEÓRICOS ADICIONALES

♦ Prueba de hipótesis

- Se establece una hipótesis nula H_0 y una alternativa H_1 .
- Se calcula la probabilidad de observar un valor tan extremo como el obtenido si H_0 fuera verdadera (valor-p).
- Si $p < \alpha$, se **rechaza** H_0 .

♦ Distribución binomial

- Modela el número de éxitos en n ensayos independientes con probabilidad p .
- Para n grandes, se puede aproximar con una distribución normal (teorema central del límite).

♦ Errores estadísticos

- **Error tipo I (α):** Rechazar H_0 siendo verdadera.
- **Error tipo II (β):** No rechazar H_0 siendo falsa.
- **Poder estadístico:** Probabilidad de rechazar correctamente H_0 cuando H_1 es verdadera.

$$\text{Poder} = 1 - \beta$$

♦ Valores críticos y función PPF

- **ppf** devuelve el valor de corte que delimita el área bajo la curva hasta un cierto percentil.
- Útil para definir regiones de rechazo (valores extremos).

🔔 **AVISO:** Ya no hay más celdas de código para explicar. Si querés, puedo ayudarte a complementar este análisis con un ejercicio práctico, resumen visual o aplicación en datos reales.

¿Te gustaría hacer eso?



Podrías hacer un resumen de la teoría de Machine Learning utilizada en el proyecto

¡Por supuesto! El proyecto que analizamos se centra en **pruebas de hipótesis estadísticas**, no en algoritmos de *Machine Learning (ML)* tradicionales como regresión, clasificación, clustering, etc. Sin embargo, esta teoría **es fundamental para entender cómo funcionan y se evalúan los modelos de ML**, especialmente en cuanto a su validación y evaluación.

A continuación te resumo la **teoría de Machine Learning implícita o relacionada** con este tipo de proyecto:

🧠 RESUMEN TEÓRICO DE MACHINE LEARNING RELACIONADO CON EL PROYECTO

📌 1. Pruebas de hipótesis en el contexto de ML

Las pruebas de hipótesis se usan en ML para:

- **Comparar modelos:** ¿Es significativamente mejor un modelo nuevo?
- **Validar mejoras:** ¿Una nueva feature mejora significativamente el rendimiento?
- **Evaluar métricas:** ¿La precisión de un clasificador es realmente mayor al azar?

Ejemplo aplicado:

Supongamos que tu clasificador acierta el 58% de las veces. ¿Es eso mejor que un clasificador aleatorio al 50%?

→ Prueba de hipótesis para evaluar si la diferencia es significativa o producto del azar.

📌 2. Distribución Binomial y su rol en ML

En tareas de clasificación binaria:

- Cada predicción correcta/incorrecta puede verse como un experimento de Bernoulli.
- La suma total de aciertos sigue una **distribución binomial**.

En validación de modelos, es común estimar:

- Si el número de aciertos supera lo esperado bajo una hipótesis nula (por ejemplo, aciertos por azar).

Esto permite aplicar:

- Cálculo de valores críticos.
- Estimación de p-valores.
- Toma de decisiones estadísticas.

📌 3. Errores tipo I y II → Evaluación de modelos

En pruebas de hipótesis y en ML, se usan términos análogos:

Estadística	ML / Clasificación
Error Tipo I (falso positivo)	Predecir clase positiva siendo negativa
Error Tipo II (falso negativo)	Predecir clase negativa siendo positiva
Poder estadístico	Sensibilidad del modelo
Región de rechazo	Umbral de decisión (threshold)

Esto se conecta directamente con las métricas de evaluación:

- **Precision:** ¿Qué proporción de las predicciones positivas fueron correctas?
- **Recall (sensibilidad):** ¿Qué proporción de los positivos reales fueron detectados?
- **F1-score:** Media armónica entre precision y recall.

4. Tamaño de muestra y varianza

En este proyecto se vio cómo aumentar el tamaño muestral:

- Reduce la varianza.
- Mejora la capacidad de detectar diferencias (aumenta el poder estadístico).

En ML:

- **Más datos** → **mejor generalización**, menor overfitting.
- **Menos datos** → **alta varianza**, modelos inestables.

5. Poder estadístico y validación cruzada

El concepto de **poder estadístico** ($1 - \beta$) es análogo a:

- La **capacidad de detectar diferencias reales** entre modelos.
- Similar a cómo se usa la **validación cruzada (cross-validation)** para reducir el error estimado.

Ejemplo:

Entrenar 10 veces un modelo y comparar su precisión media con otro.

¿La diferencia observada es significativa?

→ Aplicamos prueba t o prueba de hipótesis para determinarlo.

6. Curvas normales y distribución de errores

La suposición de **distribuciones normales de errores** está detrás de muchos algoritmos ML:

- Regresión lineal clásica: errores $\sim \mathcal{N}(0, \sigma^2)$
- Intervalos de confianza en parámetros: basados en normalidad
- Técnicas bayesianas también usan normales como priors/posteriors.

CONCLUSIÓN: ¿Qué nos aporta esto al ML?

El estudio realizado:

- **No entrena modelos de ML**, pero brinda herramientas estadísticas **imprescindibles para evaluar resultados**.
- Permite:
 - Validar mejoras reales.
 - Definir tamaños muestrales adecuados.
 - Comprender errores y riesgos al interpretar salidas de modelos.
- Por tanto, **complementa el ciclo de vida del ML** en su etapa de validación y justificación.

Si querés, puedo mostrarte cómo aplicar esto a un ejemplo de modelo real (regresión, clasificación, etc.) usando pruebas de hipótesis para validar resultados.

¿Te gustaría ver eso?