escuela
británica de
artes creativas
y tecnología

Buenas prácticas

Clase 01

Buenas prácticas

escuela británica de artes creativas y tecnología

A continuación te compartimos algunas buenas prácticas sobre intervalos de confianza y supuestos en modelos de regresión lineal múltiple incluyen:

1. Entender y aplicar correctamente los intervalos de confianza: Es crucial para un científico de datos comprender cómo calcular y interpretar los intervalos de confianza, ya que proporcionan una medida de la precisión de las estimaciones del modelo y la incertidumbre asociada con las predicciones.

2. Validar los supuestos del modelo de regresión: Antes de confiar en las predicciones de un modelo de regresión lineal múltiple, es esencial verificar que se cumplen los supuestos estadísticos, como la normalidad de los residuales, la ausencia de autocorrelación, la homocedasticidad y la inexistencia de multicolinealidad.

3. Uso de herramientas de programación para análisis estadístico: La habilidad para implementar estos conceptos utilizando herramientas de programación como Python, junto con librerías como Pandas y NumPy, es fundamental. Esto permite automatizar los cálculos y realizar análisis más complejos de manera eficiente.

Clase 01

Buenas prácticas

escuela británica de artes creativas y tecnología

4. Documentación y comunicación efectiva: Al entregar un Jupyter Notebook con el código desarrollado, es importante incluir comentarios claros y pertinentes que expliquen cada paso del análisis, facilitando la comprensión y la reproducibilidad del trabajo.

Ejemplos de utilización en el mercado laboral incluyen:

- Análisis de riesgo financiero: Los intervalos de confianza se utilizan para estimar la incertidumbre en las predicciones de modelos financieros, lo que es crucial para la toma de decisiones en inversiones y la gestión de riesgos.

- Optimización de campañas de marketing: Al pronosticar el retorno de la inversión en publicidad, los intervalos de confianza ayudan a los especialistas en marketing a entender el rango de resultados posibles y a planificar estrategias más efectivas.

Clase 01

Buenas prácticas

escuela británica de artes creativas y tecnología

- Planificación de la producción: En la industria manufacturera, los modelos de regresión pueden predecir la demanda de productos, y los intervalos de confianza permiten a los gerentes de producción prepararse para variaciones en la demanda.

En cuanto a la programación, un ejemplo de código en Python que ilustra cómo calcular un intervalo de confianza para un pronóstico en un modelo de regresión lineal múltiple podría ser el siguiente:

```python
import numpy as np
import pandas as pd
from scipy import stats

# Supongamos que ya tenemos el modelo ajustado y queremos calcular el intervalo de confianza para un nuevo punto X_new
X_new = np.array([1, 50, 150])  # Incluye el término de intercepción (1) y los valores de las variables independientes
beta_hat = np.array([5, 0.4, 0.1])  # Los coeficientes estimados del modelo
```

Clase 01

Buenas prácticas

escuela británica de artes creativas y tecnología

```
sigma = 2  # Desviación estándar de los errores del modelo
n = 100  # Número de observaciones
p = 3  # Número de parámetros (incluyendo la intercepción)

# Calculamos el pronóstico puntual
y_hat = np.dot(X_new, beta_hat)

# Calculamos el error estándar del pronóstico
X = np.random.rand(n, p)  # Esta sería la matriz de diseño del modelo
X_inv = np.linalg.inv(np.dot(X.T, X))
se_forecast = np.sqrt(sigma2 * np.dot(np.dot(X_new.T, X_inv), X_new))

# Obtenemos el valor crítico de la distribución t de Student
alpha = 0.1  # Nivel de significancia para un intervalo de confianza del 90%
df = n - p  # Grados de libertad
t_value = stats.t.ppf(1 - alpha/2, df)

# Calculamos los límites del intervalo de confianza
margin_error = t_value * se_forecast
CI_lower = y_hat - margin_error
CI_upper = y_hat + margin_error
```

Clase 01

Buenas prácticas

escuela británica de artes creativas y tecnología

```
print(f"El intervalo de confianza al 90% para el pronóstico es: ({CI_lower}, {CI_upper})")
```

Este código realiza los siguientes pasos:

- Importa las librerías necesarias.
- Define un nuevo punto para el cual se quiere hacer el pronóstico.
- Calcula el pronóstico puntual usando el producto punto entre los coeficientes estimados y el nuevo punto.
- Calcula el error estándar del pronóstico.
- Obtiene el valor crítico de la distribución t de Student.
- Calcula los límites del intervalo de confianza.
- Imprime el intervalo de confianza.

Es importante destacar que este código es un ejemplo simplificado y en una aplicación real se utilizarían los valores reales de los coeficientes y la desviación estándar de los errores obtenidos del modelo ajustado. Además, se verificarían los supuestos del modelo antes de realizar las predicciones.

Clase 02

Buenas prácticas

escuela británica de artes creativas y tecnología

4. Aplicación en el Mercado Laboral: Estas prácticas son aplicadas en el análisis de riesgos financieros, en la evaluación de la efectividad de campañas de marketing, y en la optimización de procesos en la cadena de suministro, entre otros.

5. Código Ejecutable en Python:

```python
import statsmodels.api as sm
from statsmodels.stats.diagnostic import het_white
from statsmodels.stats.stattools import jarque_bera
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Supongamos que 'X' son las variables independientes y 'y' la variable dependiente
# Añadir una constante a las variables independientes
X = sm.add_constant(X)

# Ajustar el modelo de regresión
modelo = sm.OLS(y, X).fit()
```

Clase 02

Buenas prácticas

escuela británica de artes creativas y tecnología

```
# Residuales del modelo
residuales = modelo.resid

# Prueba de Jarque-Bera para normalidad de los residuales
jb_test = jarque_bera(residuales)
print(f"Jarque-Bera test statistic: {jb_test[0]}, p-value: {jb_test[1]}")

# Prueba de White para homocedasticidad
white_test = het_white(residuales, modelo.model.exog)
print(f"White test statistic: {white_test[0]}, p-value: {white_test[1]}")

# Cálculo de VIF para cada variable independiente
vif = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
print(f"VIF values: {vif}")

# Nota: Se asume que 'X' y 'y' ya están definidos y que 'X' incluye todas las variables independientes del modelo.
```

Clase 02

Buenas prácticas

escuela británica de artes creativas y tecnología

Explicación del código:

- Se importan las librerías y funciones necesarias de `statsmodels`.
- Se añade una constante a las variables independientes para incluir el intercepto en el modelo.
- Se ajusta el modelo de regresión lineal utilizando mínimos cuadrados ordinarios (OLS).
- Se calculan los residuales del modelo.
- Se realiza la prueba de Jarque-Bera para evaluar la normalidad de los residuales.
- Se lleva a cabo la prueba de White para detectar la homocedasticidad de los residuales.
- Se calcula el VIF para cada variable independiente para identificar posibles problemas de multicolinealidad.

Cada línea de código contribuye a la verificación de los supuestos estadísticos y es una práctica recomendada en la comunidad de Científicos de Datos para asegurar la fiabilidad de los modelos de regresión.

Clase 03

Buenas prácticas

escuela británica de artes creativas y tecnología

1. Validación de Supuestos Estadísticos: Es crucial validar los supuestos estadísticos de los modelos de regresión, como la normalidad de los residuos, la ausencia de autocorrelación y la homocedasticidad. Esto asegura la confiabilidad de los resultados del modelo.

2. Uso de Paquetes Estadísticos: Utilizar paquetes como `scipy` y `statsmodels` para realizar pruebas estadísticas y validar supuestos. Estos paquetes ofrecen funciones y métodos que facilitan la implementación de pruebas complejas.

3. Interpretación de Resultados: Comprender e interpretar correctamente los resultados de las pruebas estadísticas, como el estadístico de Jarque-Bera y el estadístico Durbin-Watson, para tomar decisiones informadas sobre el modelo.

4. Transformación de Datos: Convertir los datos en formatos adecuados, como dataframes, para facilitar su manipulación y análisis en Python.

Clase 03

Buenas prácticas

escuela británica de artes creativas y tecnología

5. Construcción de Modelos Auxiliares: En algunos casos, como para verificar la homocedasticidad, puede ser necesario construir modelos auxiliares para realizar pruebas estadísticas.

6. Conocimiento de Pruebas Alternativas: Estar al tanto de pruebas alternativas disponibles, como la prueba de White, y saber cómo implementarlas en Python.

7. Aplicación Práctica: Aplicar los conocimientos teóricos en ejercicios prácticos, como la construcción de un modelo de regresión lineal múltiple y la validación de sus supuestos, para reforzar el aprendizaje.

Ejemplos de utilización en el mercado laboral incluyen:

- En el sector financiero, para validar modelos de riesgo crediticio y asegurar que las predicciones sean confiables.
- En marketing, para analizar la efectividad de campañas publicitarias mediante modelos de regresión y verificar que los supuestos se cumplan para tomar decisiones basadas en datos.

Clase 03

Buenas prácticas

escuela británica de artes creativas y tecnología

- En la investigación de operaciones, para modelar y optimizar procesos logísticos, asegurándose de que los modelos estadísticos utilizados son sólidos y confiables.

Código Ejemplificando la Validación de Supuestos de Regresión:

```python
import numpy as np
import pandas as pd
import scipy.stats as stats
import statsmodels.api as sm

# Supongamos que tenemos un conjunto de datos con variables X e Y
X = np.random.rand(100)
Y = 2 * X + np.random.normal(0, 0.1, 100)

# Convertimos X e Y en dataframes para su manipulación
X_df = pd.DataFrame(X, columns=['Variable_X'])
Y_df = pd.DataFrame(Y, columns=['Variable_Y'])
```

Clase 03

Buenas prácticas

escuela británica de artes creativas y tecnología

```
# Ajustamos un modelo de regresión lineal
X_df = sm.add_constant(X_df)  # Añadimos una constante al modelo
model = sm.OLS(Y_df, X_df).fit()

# Extraemos los residuos del modelo
residuos = model.resid

# Calculamos el sesgo y la curtosis de los residuos
sesgo = stats.skew(residuos)
curtosis = stats.kurtosis(residuos)

# Realizamos la prueba de Jarque-Bera para normalidad de los residuos
jb_test = stats.jarque_bera(residuos)

# Calculamos el estadístico Durbin-Watson para autocorrelación
dw_statistic = sm.stats.stattools.durbin_watson(residuos)
```

Clase 03

Buenas prácticas

escuela británica de artes creativas y tecnología

```
# Imprimimos los resultados
print(f"Sesgo: {sesgo}")
print(f"Curtosis: {curtosis}")
print(f"Estadístico de Jarque-Bera: {jb_test}")
print(f"Estadístico de Durbin-Watson: {dw_statistic}")

# Interpretación de los resultados:
# Un sesgo cercano a 0 y una curtosis cercana a 3 sugieren normalidad en los residuos.
# Un estadístico de Jarque-Bera bajo y un p-valor alto indican que no se rechaza la normalidad.
# Un estadístico de Durbin-Watson cercano a 2 sugiere ausencia de autocorrelación.
```

Cada línea del código anterior realiza una tarea específica para validar los supuestos de regresión, desde la preparación de los datos hasta la ejecución de pruebas estadísticas y la interpretación de sus resultados.

¡Mucho éxito en tus estudios!