T-RESEARCHER



*Código Fuente*

**T-RESEARCHER**

**Autores:**

1

2

3

**ÍNDICE**

**04**

**Código Fuente**

**05**

**Archivo: app.r**

**1. CÓDIGO FUENTE:**

La aplicación “T-Researcher” desarrollada con Python Streamlit tiene como objetivo facilitar la realización de pruebas estadísticas t de Student. La aplicación permite a los usuarios comparar medias de dos grupos independientes, una muestra con un valor teórico, o muestras emparejadas de manera sencilla y visual. Además, la aplicación genera reportes detallados en texto y PDF, incluyendo gráficos que ayudan a la interpretación de resultados.

En la presente aplicación se desarrollado con el lenguaje de programación Python el cual es el más usado para análisis de datos, así mismo, se usaron las tecnologías de Streamlit, lo que permite una interfaz web interactiva y dinámica, pandas y Numpy, para la carga y manipulación de datos, estos paquetes permiten el manejo eficiente de datos tabulares y operaciones matemáticas para las pruebas estadísticas, SciPy es fundamental para la realización de las pruebas t de Student el cual proporciona las funciones estadísticas necesarias para comparar medias y calcular valores p, Seaborn y Matplotlib, se utiliza para la visualización de datos, así como también se utilizó EPDF para la generación de informes en PDF.

La app contiene la carga y Exploración de datos, posteriormente la selección de variables de interés para realizar las pruebas t dependiendo del tipo de prueba seleccionada, posteriormente la prueba t de student, luego se muestra algunos gráficos importantes para comprender mejor y tener un mejor entendimiento de los datos, al final se tiene la generación de informes pdf.

ARCHIVO: APP.PY

import streamlit as st

import pandas as pd

import numpy as np

from scipy import stats

import seaborn as sns

import matplotlib.pyplot as plt

from io import BytesIO

from fpdf import FPDF

import tempfile

# Clase personalizada para agregar encabezado y pie de página

class PDF(FPDF):

    def header(self):

        # Establecer fuente para el encabezado

        self.set\_font("Arial", "B", 10)

        # Encabezado (se repite en todas las páginas)

        self.cell(0, 10, "Informe de Prueba t de Student de la app hecha por estudiantes de la Universidad Nacional del Altiplano de", ln=True, align="C")

        self.ln(5)  # Espacio extra después del encabezado

    def footer(self):

        # Establecer posición del pie de página a 1.5 cm del final

        self.set\_y(-15)

        # Establecer fuente para el pie de página

        self.set\_font("Arial", "I", 8)

        # Número de página

        self.cell(0, 10, f"Página {self.page\_no()}", 0, 0, "C")

# Título y descripción de la aplicación

st.title("Aplicación de Prueba t de Student")

st.write("""

Esta aplicación permite realizar pruebas t de Student para comparar las medias de dos grupos o para comparar la media de un grupo con un valor teórico.

Puedes cargar tus datos, seleccionar las variables y realizar la prueba de forma interactiva.

""")

# Carga de datos

uploaded\_file = st.file\_uploader("Cargar archivo CSV o Excel", type=["csv", "xlsx"])

if uploaded\_file:

    if uploaded\_file.name.endswith('.csv'):

        df = pd.read\_csv(uploaded\_file)

    else:

        df = pd.read\_excel(uploaded\_file)

    st.write("Datos cargados:")

    st.dataframe(df.head())

    # Selección de variables

    st.write("### Selección de Variables")

    columns = df.select\_dtypes(include=np.number).columns.tolist()

    test\_type = st.radio("Tipo de Prueba t", ["Dos muestras independientes", "Una muestra", "Muestras emparejadas"])

    if test\_type == "Dos muestras independientes":

        variable1 = st.selectbox("Selecciona la primera variable numérica", columns)

        variable2 = st.selectbox("Selecciona la segunda variable numérica", [col for col in columns if col != variable1])

        equal\_var = st.checkbox("¿Asumir varianzas iguales?", value=True)

        t\_stat, p\_value = stats.ttest\_ind(df[variable1], df[variable2], equal\_var=equal\_var)

        st.write(f"Resultados de la Prueba t de Dos Muestras Independientes:")

        st.write(f"t-statistic: {t\_stat:.4f}")

        st.write(f"p-value: {p\_value:.4f}")

        st.write(f"Grados de libertad: {len(df[variable1]) + len(df[variable2]) - 2}")

        # Gráfico de comparación

        st.write("### Visualización Gráfica")

        fig1, ax1 = plt.subplots()

        sns.boxplot(data=df[[variable1, variable2]], ax=ax1)

        st.pyplot(fig1)

        # Histograma de las variables

        st.write("### Histogramas")

        fig2, (ax2, ax3) = plt.subplots(1, 2, figsize=(10, 4))

        sns.histplot(df[variable1], kde=True, ax=ax2)

        ax2.set\_title(f'Histograma de {variable1}')

        sns.histplot(df[variable2], kde=True, ax=ax3)

        ax3.set\_title(f'Histograma de {variable2}')

        st.pyplot(fig2)

        # Gráfico de dispersión

        st.write("### Gráfico de Dispersión")

        fig3, ax4 = plt.subplots()

        sns.scatterplot(x=df[variable1], y=df[variable2], ax=ax4)

        ax4.set\_title(f'Dispersión entre {variable1} y {variable2}')

        st.pyplot(fig3)

    elif test\_type == "Una muestra":

        variable = st.selectbox("Selecciona la variable numérica", columns)

        test\_value = st.number\_input("Ingresa el valor teórico para comparar", value=0.0)

        t\_stat, p\_value = stats.ttest\_1samp(df[variable], test\_value)

        st.write(f"Resultados de la Prueba t de Una Muestra:")

        st.write(f"t-statistic: {t\_stat:.4f}")

        st.write(f"p-value: {p\_value:.4f}")

        st.write(f"Grados de libertad: {len(df[variable]) - 1}")

        # Histograma de la variable

        st.write("### Histograma")

        fig4, ax5 = plt.subplots()

        sns.histplot(df[variable], kde=True, ax=ax5)

        ax5.set\_title(f'Histograma de {variable}')

        st.pyplot(fig4)

    elif test\_type == "Muestras emparejadas":

        variable1 = st.selectbox("Selecciona la primera variable numérica", columns)

        variable2 = st.selectbox("Selecciona la segunda variable numérica", [col for col in columns if col != variable1])

        t\_stat, p\_value = stats.ttest\_rel(df[variable1], df[variable2])

        st.write(f"Resultados de la Prueba t de Muestras Emparejadas:")

        st.write(f"t-statistic: {t\_stat:.4f}")

        st.write(f"p-value: {p\_value:.4f}")

        st.write(f"Grados de libertad: {len(df[variable1]) - 1}")

        # Gráfico de comparación

        st.write("### Visualización Gráfica")

        fig1, ax1 = plt.subplots()

        sns.boxplot(data=df[[variable1, variable2]], ax=ax1)

        st.pyplot(fig1)

        # Histograma de las variables

        st.write("### Histogramas")

        fig2, (ax2, ax3) = plt.subplots(1, 2, figsize=(10, 4))

        sns.histplot(df[variable1], kde=True, ax=ax2)

        ax2.set\_title(f'Histograma de {variable1}')

        sns.histplot(df[variable2], kde=True, ax=ax3)

        ax3.set\_title(f'Histograma de {variable2}')

        st.pyplot(fig2)

    # Interpretación de resultados

    st.write("### Interpretación de Resultados")

    alpha = st.number\_input("Nivel de significancia (alpha)", value=0.05)

    interpretacion = ("Rechazamos la hipótesis nula. Existe una diferencia significativa."

                      if p\_value < alpha else

                      "No podemos rechazar la hipótesis nula. No existe evidencia suficiente para afirmar que hay una diferencia significativa.")

    st.write(interpretacion)

    # Generar reporte

    reporte = f"""

    Resultados de la Prueba t de Student\n

    Tipo de prueba: {test\_type}\n

    t-statistic: {t\_stat:.4f}\n

    p-value: {p\_value:.4f}\n

    Nivel de significancia (alpha): {alpha}\n

    Interpretación: {interpretacion}

    """

    # Descargar reporte en texto

    st.download\_button(

        label="Descargar Reporte (Texto)",

        data=reporte,

        file\_name='reporte\_prueba\_t.txt',

        mime='text/plain'

    )

    # Opción para generar PDF

    if st.button("Generar Informe en PDF"):

        pdf = PDF()  # Usar la clase personalizada con encabezado y pie de página

        pdf.add\_page()

        # Título (solo en la primera página)

        pdf.set\_font("Arial", "B", 16)

        pdf.cell(0, 10, "Resultados de la Prueba t de Student", ln=True, align="C")

        pdf.ln(10)

        # Resultados

        pdf.set\_font("Arial", size=12)

        pdf.multi\_cell(0, 10, reporte)

        # Gráfico de comparación

        pdf.ln(10)

        pdf.cell(200, 10, txt="Gráficos:", ln=True, align='C')

        # Guardar los gráficos en archivos temporales y luego insertarlos en el PDF

        if test\_type == "Dos muestras independientes":

            with tempfile.NamedTemporaryFile(delete=False, suffix=".png") as tmpfile1:

                fig1.savefig(tmpfile1.name, format="png")

                pdf.image(tmpfile1.name, x=10, y=None, w=180)

            with tempfile.NamedTemporaryFile(delete=False, suffix=".png") as tmpfile2:

                fig2.savefig(tmpfile2.name, format="png")

                pdf.add\_page()

                pdf.image(tmpfile2.name, x=10, y=None, w=180)

            with tempfile.NamedTemporaryFile(delete=False, suffix=".png") as tmpfile3:

                fig3.savefig(tmpfile3.name, format="png")

                pdf.add\_page()

                pdf.image(tmpfile3.name, x=10, y=None, w=180)

        elif test\_type == "Una muestra":

            with tempfile.NamedTemporaryFile(delete=False, suffix=".png") as tmpfile4:

                fig4.savefig(tmpfile4.name, format="png")

                pdf.image(tmpfile4.name, x=10, y=None, w=180)

        elif test\_type == "Muestras emparejadas":

            with tempfile.NamedTemporaryFile(delete=False, suffix=".png") as tmpfile1:

                fig1.savefig(tmpfile1.name, format="png")

                pdf.image(tmpfile1.name, x=10, y=None, w=180)

            with tempfile.NamedTemporaryFile(delete=False, suffix=".png") as tmpfile2:

                fig2.savefig(tmpfile2.name, format="png")

                pdf.add\_page()

                pdf.image(tmpfile2.name, x=10, y=None, w=180)

        # Guardar el archivo PDF en un buffer

        pdf\_buffer = BytesIO()

        pdf\_output = pdf.output(dest='S').encode('latin1')

        pdf\_buffer.write(pdf\_output)

        pdf\_buffer.seek(0)

        # Descargar el archivo PDF

        st.download\_button(

            label="Descargar Informe en PDF",

            data=pdf\_buffer,

            file\_name="informe\_prueba\_t.pdf",

            mime="application/pdf"

        )

else:

    st.write("Por favor, carga un archivo de datos para comenzar.")

