Reto: Conociendo el desempeño de los colaboradores del Área de Marketing de Socialize your knowledge

Entregable final del reto.
Documento editable para el participante.

Nombre del participante:

Indicaciones:

- Guarda este archivo de la siguiente manera: **NombreApellido_reto_C4SC4**
- Una vez terminado el reto deberás subirlo a la plataforma.

Instrucciones:

Elabora una aplicación web para mostrar el análisis de desempeño de los colaboradores de Socialize your knowledge.

Para lograrlo, lee cada uno de los pasos y realiza lo solicitado.

1. Analiza la base de datos del documento **employee_data.csv** e identifica únicamente los siguientes puntos:

    - Nombre del empleado (name_employee)
    - Fecha de nacimiento (birth_date)
    - Edad (age)
    - Género (gender)
    - Estado civil (marital_status)
    - Fecha de contratación (hiring_date)
    - Puesto (position)
    - Salario (salary)
    - Puntaje de desempeño (de 1 a 5, donde 5 es la máxima calificación) (performance_score)
    - Fecha de revisión de desempeño más reciente (last_performance_date)
    - Promedio de horas mensuales trabajadas (average_work_hours)
    - Nivel de satisfacción de los empleados (satisfaction_level)
    - Ausencias (absences)

2. Utilizando el lenguaje Python y la plataforma Streamlit, genera un código para construir un *dashboard* que contenga lo siguiente:

    - Código que contenga las instrucciones para el despliegue de un título y una breve descripción de la aplicación web.

- Código que permita desplegar el logotipo de la empresa en la aplicación web.
- Código que permita desplegar un control para seleccionar el género del empleado.
- Código que permita desplegar un control para seleccionar un rango del puntaje de desempeño del empleado.
- Código que permita desplegar un control para seleccionar el estado civil del empleado.
- Código que permita mostrar un gráfico en donde se visualice la distribución de los puntajes de desempeño.
- Código que permita mostrar un gráfico en donde se visualice el promedio de horas trabajadas por el género del empleado.
- Código que permita mostrar un gráfico en donde se visualice la edad de los empleados con respecto al salario de los mismo.
- Código que permita mostrar un gráfico en donde se visualice la relación del promedio de horas trabajadas versus el puntaje de desempeño.
- Código que permita desplegar una conclusión sobre el análisis mostrado en la aplicación web.

3. Coloca el código completo identificando cada una de las secciones (ejemplo: visualización de la edad de los empleados vs. el salario) de toda la aplicación desarrollada.

Código

```
# *****************************************************************************
#Importing Libraries
# *****************************************************************************
import streamlit as st
import os
import pandas as pd
import numpy as np
import altair as alt
from PIL import Image
from bokeh.plotting import figure, show
import plotly.graph_objects as go
import plotly.express as px
import matplotlib.pyplot as plt
import base64

# *****************************************************************************
#Code
# *****************************************************************************

st.set_page_config(page_title="Sales Dashboard", page_icon=":cactus:", layout="wide")


# Escribe un título y algo de texto para la app:
'''
# Reto: Analysis for *Socialize your knowledge*
```

```
## Marketing Department Performance
'''

image = Image.open(os.path.join(os.getcwd(),'Images', 'TLG.png'))
path = os.path.join(os.getcwd(), 'Data', 'Employee_data.csv')
df = pd.read_csv(path)
df.gender = df.gender.str.strip()
df['date'] = pd.to_datetime(df.last_performance_date)
df['year'] = df.date.dt.year
df['month'] = df.date.dt.month
df['month_name'] = df.date.dt.month_name()


with open(os.path.join(os.getcwd(),'Images', 'TLG.png'), "rb") as f:
    data = base64.b64encode(f.read()).decode("utf-8")

    st.sidebar.markdown(
        f"""
        <div style="display:table;margin-top:-20%;margin-left:5%;">
            <img src="{data}" width="250" height="120">
        </div>
        """,
        unsafe_allow_html=True,
    )


#'''********************** Creating Control sidebars **********************'''

st.sidebar.header("Control Filters")
st.sidebar.subheader("Company Collaborators")

Gender = st.sidebar.multiselect(
    "Gender:",
    options=df["gender"].unique(),
    default=df["gender"].unique()
)

Performance = st.sidebar.multiselect(
    "Performance Score:",
    options=df["performance_score"].unique(),
    default=df["performance_score"].sort_values().unique(),
)

Marital = st.sidebar.multiselect(
    "Marital Status:",
    options=df["marital_status"].unique(),
    default=df["marital_status"].sort_values().unique(),
)
```

```
Year = st.sidebar.multiselect(
    "Year:",
    options=df["year"].unique(),
    default=df["year"].sort_values().unique(),
)

# '''************* Using the sidebars control to filter the data ***************'''

cols = []
vals = []

# Filtering dataframe
if len(Gender) != 0:
    cols.append('gender')
    vals.append(Gender)

if len(Performance) != 0:
    cols.append('performance_score')
    vals.append(Performance)

if len(Marital) != 0:
    cols.append('marital_status')
    vals.append(Marital)

if len(Year) != 0:
    cols.append('year')
    vals.append(Year)


df_filt = df.copy()
for col, val in zip(cols, vals):
    df_filt = df_filt[df_filt[col].isin(val)]

# '''************************* Filtering the Data ******************************'''

st.markdown("""---""")
st.title(":sparkles: Summary Highlights")

# TOP KPI's
total_employees = int(df_filt.id_employee.unique().shape[0])
print(df_filt.gender.unique())
if 'M' in df_filt.gender.tolist():
    total_employeesM = round(100 *
df_filt[df_filt.gender=='M'].id_employee.unique().shape[0]/total_employees, 1)
    male_number = ":male-office-worker:" * int(round(1, 0))
if 'F' in df_filt.gender.tolist():
    total_employeesF = (round(100 *
df_filt[df_filt.gender=='F'].id_employee.unique().shape[0]/total_employees, 1))
```

```
    female_number = ":female-office-worker:" * int(round(1, 0))
avg_performance = round(df_filt["performance_score"].mean(), 1)
star_rating = ":star:" * int(round(avg_performance, 0))
avg_satisfaction = round(df_filt["satisfaction_level"].mean(), 1)
check_rating = ":white_check_mark:" * int(round(avg_satisfaction, 0))

left_column, middle_column, right_column = st.columns(3)
with left_column:
    st.subheader("Number of Employees:")
    if ('M' in df_filt.gender.tolist()) & ('F' in df_filt.gender.tolist()):
        st.subheader(f"{total_employees}( {total_employeesM}% {male_number} -
{total_employeesF}% {female_number})")
    elif 'M' in df_filt.gender.tolist():
        st.subheader(f"{total_employees}({total_employeesM}% {male_number})")
    else:
        st.subheader(f"{total_employees}({total_employeesF}% {female_number})")

with middle_column:
    st.subheader("Employees Performance Score:")
    st.subheader(f"{avg_performance} {star_rating}")
with right_column:
    st.subheader("Customers Satisfaction:")
    st.subheader(f"{avg_satisfaction} {check_rating}")

st.markdown("""---""")

# '''********************** Creating Plots as Required ************************'''

st.title(":bar_chart: Analysis Dashboard")

#Histogram - performance score
fig_performance_score =
px.histogram(df_filt.rename(columns={'performance_score':'Score'}).sort_values("Score"),
                            x="Score", color='Score',
                            color_discrete_sequence=['#00ff04', '#fbff00', '#00f7ff', '#ff7000'],
                            title="Performance Score for Marketing Employees - Histogram")


#barchart - Workinghours Vs. Gender
df_hrs = df_filt.groupby(['month', 'month_name',
'gender'])[['average_work_hours']].mean().reset_index().sort_values('month')
df_hrs.rename(columns={'month_name':'Month', 'average_work_hours':'Worked Hours',
'gender':'Gender'}, inplace=True)
fig_working_hrs = px.bar(df_hrs[df_hrs.Gender.isin(['M', 'F'])], x="Month", y="Worked Hours",
barmode='group',
                    color='Gender', color_discrete_sequence=["royalblue", "darkorange"],
                    title="Yearly Average Working Hours per Gender")
```

```
#Scatterplot - AgeVsSalary
cmap = {1: '#00ff04',
        2: '#fbff00',
        3: '#00f7ff',
        4: '#ff7000'}

fig_agevssalary =  alt.Chart(df_filt.rename(columns={'age':'Age', 'salary':'Salary',
'performance_score':'Score'})).mark_circle(size=60).encode(
    x=alt.X('Age', scale=alt.Scale(domain=[df_filt.age.min()-2, df_filt.age.max()+2])),
    y=alt.Y('Salary', scale=alt.Scale(domain=[df_filt.salary.min()-5000, df_filt.salary.max()+5000])),
    color=alt.Color('Score', scale=alt.Scale(domain=list(cmap.keys()), range=list(cmap.values()))),
type='nominal'),
    tooltip=['position', 'Score', 'Age', 'Salary']).properties(title="Relationship Between Employees'
Ages and Their Salaries").interactive()

#Boxplor - WorkerdHrsVsScore
cmap1 = {1: '#029900',
        2: '#9da512',
        3: '#026ac6',
        4: '#af0e0e'}

fig_hrsvsscore_bp = alt.Chart(df_filt.rename(columns={'average_work_hours':'Worked Hours',
'salary':'Salary', 'performance_score':'Score'})).mark_boxplot(size=50, ticks=True,
color='lightblue', box={'stroke': 'yellow'}, median=alt.MarkConfig(stroke='cyan')).encode(
    x=alt.X("Score:O", scale=alt.Scale(padding=2)),
    y=alt.Y("Worked Hours", scale=alt.Scale(domain=[df_filt.average_work_hours.min(),
df_filt.average_work_hours.max()])),
    color=alt.Color('Score', scale=alt.Scale(domain=list(cmap.keys()),
range=list(cmap1.values())), type='nominal'),
    tooltip=['position', 'Score', 'Worked Hours', 'Salary']).properties(title="Understanding the Link
Between Worked Hours and Performance Scores").interactive()

#Displaying Plots
left_column, right_column = st.columns(2)
left_column.plotly_chart(fig_working_hrs, use_container_width=True)
right_column.plotly_chart(fig_performance_score, use_container_width=True)

left_column2, right_column2 = st.columns(2)
left_column2.altair_chart(fig_agevssalary, use_container_width=True)
right_column2.altair_chart(fig_hrsvsscore_bp, use_container_width=True)

# ********************************************************************************
# END
# ********************************************************************************
```

No olvides colocar una página de presentación y el código completo.

Apéndice

Capturas del Dashboard Creado:

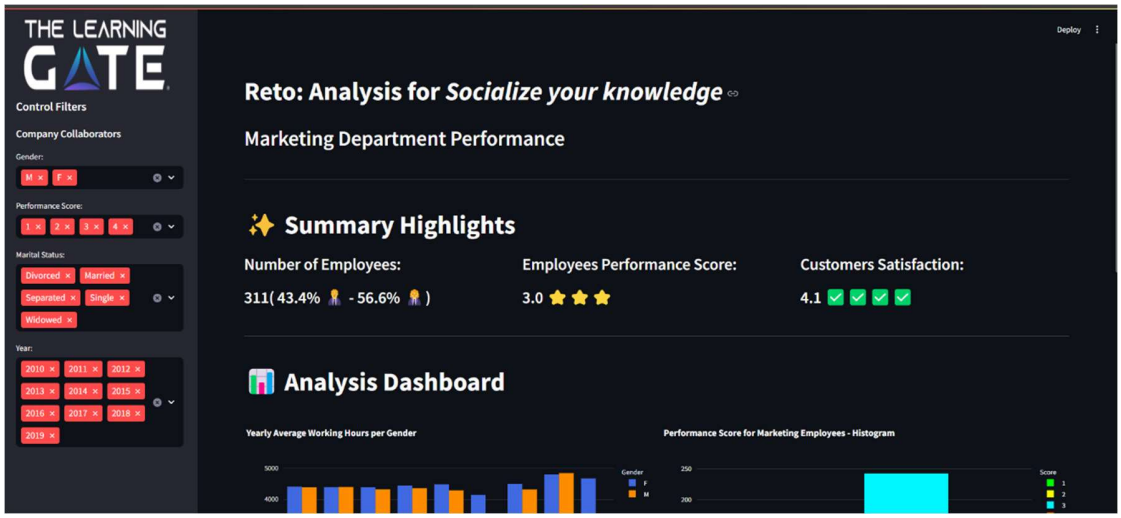

Ilustración 1.- Página de Inicio del Dashboard

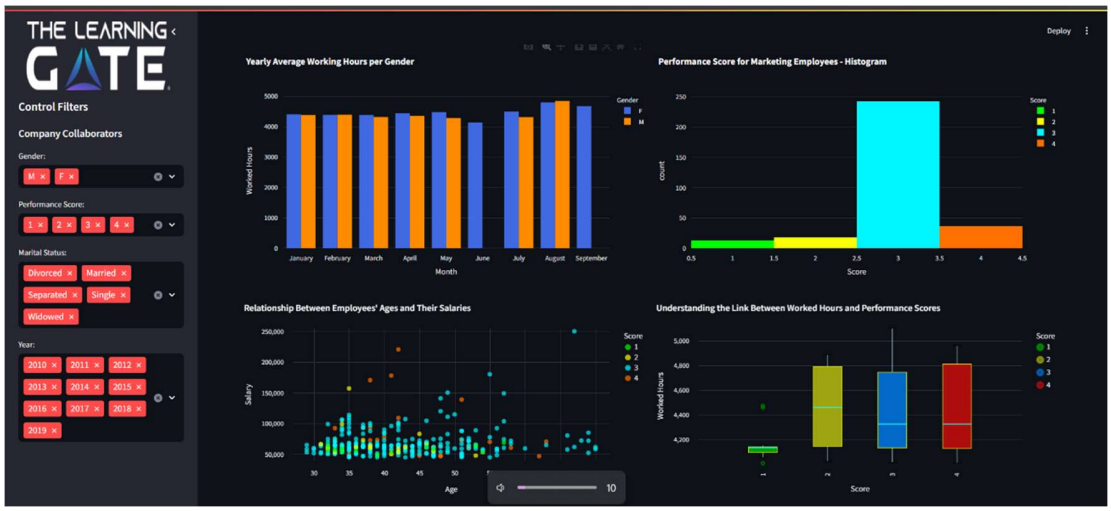

Ilustración 2.- Graficas Requeridas