Documentación *Dashborad* de Beer Company: ¿Qué tan fresco es lo que consumes?

Grupo: Cuenteros de la MIAD

Santiago Pulido, Diego Peñaloza, Miller Patiño

MIAD - Visualización y Storytelling 202401

A continuación, encontrará la documentación para generación del *dashboard* ¿Qué tan fresco es lo que consumes? Que presenta a un público general información relevante acerca de la frescura, y por consiguiente de la calidad, de la cerveza en el mercado colombiano y los factores que la afectan.

La medida de frescura es importante ya que indica cuánto dura un producto en el mercado y por consiguiente sirve por una parte para indicarle al consumidor cuales son los productos más frescos y de mejor calidad en el mercado; mientras que por otra parte sirve de indicador a los productores de cómo se encuentra la saturación del mercado de acuerdo a los productos que produce. Si bien esta segunda parte es muy relevante de cara a las empresas productoras, ya que influencia directamente la decisión de producción, decidimos enfocarnos en comunicar al consumidor promedio de cerveza en Colombia que influye en la calidad de la cerveza que consume. Por ello nuestro objetivo es que el consumidor esté informado de cómo puede obtener el mejor producto según sus preferencias de consumo e informarle sobre el estado del mercado cervecero nacional.

- El dashboard se puede acceder en este vínculo.
- La explicación del uso del reporte se encuentra en el siguiente vinculo.
- La documentación y el proyecto se encuentra en este vínculo.

Datos utilizados

La base de datos proviene del histórico de mediciones de calidad de distintas cervezas producidas por la empresa *Beer Company*[1] en el mercado colombiano, desde abril de 2021 y diciembre de 2023. Esta base contiene 9 variables, de las cuales 5 son categóricas, 2 son fechas y 2 son numéricas. La medición de frescura se realiza a partir de la edad del producto (cantidad de días del producto desde su fecha e producción hasta su fecha de medición) y niveles de frescura estimados de acuerdo a los días y la marca. Dentro de la información inicial, se utilizará el campo 'Referencia' para extraer las variables:

- Marca
- Tipo de envase
- Tamaño o capacidad del envase

[1] Para efectos de este ejercicio se solicitó enmascarar el nombre de la empresa productora que nos facilitó la información.

- Retornabilidad

También se calcularán las siguientes variables:

- Fecha de producción
- Mes de producción
- Mes de medición

Finalmente se crea una variable categórica de nivel de frescura a partir de las reglas:

Tabla 1. Parámetros de construcción de la variable de clasificación de frescura

Marca	Edad de producto	Clasificación
1	< 90	Ideal
2		
3		
1	90 <= Edad_producto <= 120	Debe mejorar
2		
3		
1	> 120	Inaceptable
2		
3		
4	< 60	Ideal
5		
6		
7		
4	60 <= Edad_producto <= 100	Debe mejorar
5		
6		
7		
4	> 100	Inaceptable
5		
6		
7		

Que nos daría un total de 11 variables categóricas, 4 son fechas y 3 son numéricas.
A continuación, se realiza una breve descripción de las variables disponibles.

Tabla 2. Descripción de los campos incluidos en la base final

Columnas	Fuente	Tipo de dato	Descripción
Referencia	Original en la base	Categórica	Referencia de la marca, compuesta por la marca, el envase, el tamaño y el tipo

Establecimiento	Original en la base	Categórica	Lugar donde se realizó la encuesta
Ciudad	Original en la base	Categórica	Ciudad
Canal	Original en la base	Categórica	Grupo de lugar donde se realizó la encuesta
Fecha_vencimiento	Original en la base	Fecha	Fecha de vencimiento de la muestra evaluada
Edad_producto	Original en la base	Numérica	Se obtiene a partir: Fecha_encuesta - (Fecha_vencimiento - Vida_Util)
Lote	Original en la base	Categórica	Lote del producto evaluado
Vida_util	Original en la base	Numérica	Vida útil del producto evaluado
Fecha_encuesta	Original en la base	Fecha	Fecha en la que se realizó la evaluación
Marca[2]	Construcción	Categórica	Marca del producto
Envase	Construcción	Categórica	Tipo de envase (lata / botella)
Tamaño	Construcción	Numérica	Tamaño del envase en ml
Retornabilidad	Construcción	Categórica	Tipo de retornabilidad del envase (retornable / no retornable)
Fecha de producción	Construcción	Fecha	Fecha vencimiento – vida útil
Mes de producción	Construcción	Categórica	Mes de la fecha de producción
Mes de medición	Construcción	Categórica	Mes de la fecha de medición
Nivel_Frescura	Construcción	Categórica	Construcción de acuerdo con las reglas indicadas arriba (Ideal, debe mejorar, inaceptable)

Procesamiento de los datos

[2] Para efectos de este ejercicio, decidimos enmascarar los nombres reales de las marcas reportadas. Esto por respeto a la confidencialidad de la empresa que nos permitió el uso de su información.

A continuación, se indica cual es el proceso de obtención y transformación de datos para alimentar la visualización:

1. Fuente de Información: La fuente de información se obtiene desde el departamento de calidad de la empresa *Beer Company*. Cada registro de esta base constituye una medida de frescura de un producto en específico. La información se diligencia por miembros del equipo de calidad.
2. La información se obtiene semanalmente de parte del grupo de calidad en un archivo llamado calidad.
3. Se debe modificar el nombre a 'Datos.xlsx', asegurarse que la información se encuentra en la hoja 'Export' y guardar la información en la ruta del reporte.
4. Esta información actualizada se debe reemplazar en el repositorio de Github sobre el cual se construyó la visualización web. El acceso al Github se puede realizar a través del siguiente vinculo: https://github.com/diegoalejop/MIAD-visualizacion.
5. El proceso del proyecto toma el notebook de Python "dashboars.py" con el cual se depura la información y se construye la estructura necesaria para el reporte.
6. Como resultado de este proceso se genera el *dataframe* sobre el cual se construyó la visualización del reporte en la plataforma Streamlit.
7. Una vez actualizada la base de datos y generado el reporte se deben verificar las visualizaciones y estar atento a cambios abruptos en los datos que ameriten una revisión profunda del proceso.

Descripción de los métodos y clases principales del Proyecto

Carga de Datos: El archivo Excel se carga en un DataFrame pandas desde una ruta específica, seleccionando la hoja 'Export'.

```
datos = pd.read_excel(r"D:\Maestria_Andes_2023\Ciclo_3\Visualización y storytelling\6.Semana\MIAD-visualizacion\Datos.xlsx",
                      sheet_name='Export')
```

Transformación de Datos:

Fecha de Producción: Se calcula la fecha de producción de los productos restando la Vida útil (expresada en días como timedelta) de la Fecha_vencimiento. Esto da como resultado una nueva columna Fecha_produccion.

```
# Obtener columna Fecha_produccion: Fecha_vencimiento  - Vida_util
datos['Vida_util_timedelta'] = pd.to_timedelta(datos['Vida_util'], unit='D')
datos['Fecha_produccion'] = datos['Fecha_vencimiento'] - datos['Vida_util_timedelta']
```

Descomposición de la Columna Referencia: La columna Referencia se divide en múltiples columnas (Marca, Empaque, Volumen, Und_Volumen, Retornable, Texto) utilizando el

espacio como delimitador. Esta operación permite analizar por separado cada componente de la referencia del producto.

```
# Split por espacio de la columna Referencia para obtener Marca, Empaque, Volumen, Und_volumen, Retornable y Texto
datos[['Marca', 'Empaque', 'Volumen', 'Und_Volumen', 'Retornable','Texto']] = datos['Referencia'].str.split(expand=True)
```

Normalización de la Columna Retornable: Los valores de la columna Retornable se normalizan cambiando cualquier instancia de 'Retornable' por 'SI', lo que indica que el empaque es retornable.

```
# Split por espacio de la columna Referencia para obtener Marca, Empaque, Volumen, Und_volumen, Retornable y Texto
datos[['Marca', 'Empaque', 'Volumen', 'Und_Volumen', 'Retornable','Texto']] = datos['Referencia'].str.split(expand=True)
```

Extracción de Año y Mes de Producción y Encuesta: Se crean nuevas columnas para separar el año y el mes tanto de la Fecha_produccion como de la Fecha_encuesta. Estas columnas permiten análisis temporales más detallados.

```
# Creación de columnas año_produccion y mes_produccion, a partir de Fecha_produccion
datos['año_produccion'] = datos['Fecha_produccion'].dt.year
datos['mes_produccion'] = datos['Fecha_produccion'].dt.month
```

Eliminación de Columnas: Se eliminan varias columnas que se consideran innecesarias para la visualización o análisis posterior.

```
# Eliminación de columnas innecesarias para la visualización
datos = datos.drop(columns=['Referencia', 'Establecimiento', 'Lote', 'Vida_util_timedelta', 'Texto', 'Und_Volumen', 'Fecha_vencimiento'])
```

Cálculo de la Frescura:

Se define una función calculo_frescura para determinar la frescura de los productos basándose en su Edad_producto y Marca. Las condiciones varían según la marca: para Marca_1, Marca_2, y Marca_3, el producto es considerado ideal si su edad es menor a 90 días, debe mejorar si está entre 90 y 120 días, e inaceptable si supera los 120 días. Para Marca_4, Marca_5, Marca_6, y Marca_7, los límites son menores a 60 días para ideal, entre 60 y 100 días para debe mejorar, e inaceptable si supera los 100 días.

Esta función se aplica al DataFrame para crear la columna Frescura, que clasifica cada producto según las condiciones definidas.

Resultado Final: El DataFrame datos transformado se almacena en la variable df, listo para ser utilizado en análisis o visualizaciones subsiguientes. Este DataFrame contiene información relevante y limpia sobre los productos, incluyendo su frescura, fecha de producción, y categorías derivadas de la columna Referencia, entre otros.

```python
# Cálculo de la columna frescura teniendo en cuenta sus respectivas condiciones
def calculo_frescura(row):
    # Condiciones Marca_1, Marca_2, Marca_3
    if row['Marca'] in ['Marca_1', 'Marca_2', 'Marca_3']:
        if row['Edad_producto'] < 90:
            return 'Ideal'
        elif 90 < row['Edad_producto'] <= 120:
            return 'Debe Mejorar'
        else:
            return 'Inaceptable'
    # Condiciones Marca_4, Marca_5, Marca_6, Marca_7
    elif row['Marca'] in ['Marca_4', 'Marca_5', 'Marca_6', 'Marca_7']:
        if row['Edad_producto'] < 60:
            return 'Ideal'
        elif 60 < row['Edad_producto'] <= 100:
            return 'Debe Mejorar'
        else:
            return 'Inaceptable'

# Aplicar la función para crear la nueva columna
datos['Frescura'] = datos.apply(calculo_frescura, axis=1)
```

El código demuestra un flujo de trabajo práctico para la preparación de datos, donde se limpian, transforman y enriquecen los datos originales para facilitar análisis y visualizaciones más eficientes.

Archivo de construcción del Dashboard (dashboard.py)

Este código es para una aplicación web utilizando Streamlit, diseñada para analizar y visualizar la frescura de bebdas alcohólicas a partir de un conjunto de datos cargado desde un archivo Excel. La aplicación incorpora elementos visuales interactivos y permite al usuario filtrar los datos según ciertos criterios. Aquí se detallan las funcionalidades clave:

Inyección de CSS: Se personaliza la apariencia de la aplicación inyectando CSS para cambiar el color de fondo y mejorar el estilo de los elementos de texto dentro de los contenedores de información.

```
# Inyectar CSS para cambiar el color de fondo utilizando st.markdown
st.markdown("""
    <style>
    @import url('https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;700&display=swap');

    .stApp {
        background-color: #F5F7F8;
    }

    /* Aumentar la especificidad para los elementos de texto dentro de st.info */
    .stAlert>div>div {
        font-family: 'Open Sans', sans-serif !important;
        font-size: 15px !important; /* Asegúrate de que el tamaño de la fuente se aplique */
        color: #000000 !important; /* Asegura que el color se aplique */
    }
    </style>
    """, unsafe_allow_html=True)
```

Filtros en la Barra Lateral: La aplicación ofrece filtros en la barra lateral para que los usuarios puedan seleccionar años de producción, ciudades, marcas y canales específicos, y así ajustar los datos mostrados en la aplicación.

```
#########
# Sidebar
st.sidebar.title('Filtros del reporte')
st.sidebar.header("Filtra el reporte de acuerdo con tu selección:")

ciudad_unico = df['Ciudad'].unique()
año_unico = df['año_produccion'].unique()
marca_unico = df['Marca'].unique()
canal_unico = df['Canal'].unique()

## Ordena las opciones si es necesario
ciudad_unico.sort()
año_unico.sort()
marca_unico.sort()
canal_unico.sort()

## objeto sidebar
año_select = st.sidebar.selectbox('Selecciona años de producción:', año_unico)
ciudad_select = st.sidebar.multiselect('Selecciona una o más ciudades:', ciudad_unico)
marca_select = st.sidebar.multiselect('Selecciona una marca:', marca_unico)
canal_select = st.sidebar.multiselect('Selecciona uno o más canales:', canal_unico)

if not canal_select and not ciudad_select and not marca_select:
    # Si no se seleccionó ningún canal, ciudad o marca, filtrar solo por año
    filtered_df = df[df['año_produccion'] == año_select]
else:
    # Aplicar filtros basados en selecciones
    filtered_df = df[df['año_produccion'] == año_select]
    if marca_select:  # Si se seleccionaron marcas
        filtered_df = filtered_df[filtered_df['Marca'].isin(marca_select)]
    if canal_select:  # Si se seleccionaron canales
        filtered_df = filtered_df[filtered_df['Canal'].isin(canal_select)]
    if ciudad_select:  # Si se seleccionaron ciudades
        filtered_df = filtered_df[filtered_df['Ciudad'].isin(ciudad_select)]

#Créditos
```

Visualización de Datos:

Se utilizan bibliotecas de visualización como Plotly para crear gráficos interactivos, incluyendo gráficos de líneas para tendencias mensuales de frescura, gráficos de embudo para distribución de frescura por empaque y ciudad, y mapas con Folium para representar la frescura en diferentes ciudades.

Se muestra información resumida sobre la frescura de los productos, incluyendo edad promedio, mínima y máxima, y desviación estándar.

Se proporcionan explicaciones sobre conceptos importantes relacionados con el producto, como la edad del producto, la calidad, la retornabilidad y los canales de venta, mediante botones que, al hacer clic, despliegan textos explicativos.

```python
with st.expander("📖 Ver explicación"):
    st.markdown(explicacion_general)

datos_linea_g = filtered_df.groupby('mes_encuesta')['Edad_producto'].agg(['mean', 'std']).reset_index()
fig_line_g = go.Figure()
# Línea de la media
fig_line_g.add_trace(go.Scatter(
    x=datos_linea_g['mes_encuesta'].astype(str),
    y=datos_linea_g['mean'],
    mode='lines+markers',
    name='Media de Frescura',
    showlegend=False,
    line = dict(color = 'rgb(0,97,57,1)')
))

# Banda de error para la desviación estándar
fig_line_g.add_trace(go.Scatter(
    x=pd.concat([datos_linea_g['mes_encuesta'], datos_linea_g['mes_encuesta'][::-1]]).astype(str),
    y=pd.concat([datos_linea_g['mean'] + datos_linea_g['std']/20, (datos_linea_g['mean'] - datos_linea_g['std']/20)[::-1]]),
    fill='toself',
    fillcolor='rgba(0,100,80,0.2)',
    line=dict(color='rgba(0,97,57,0)'),
    hoverinfo="skip",
    showlegend=False
))

# Actualizar layout para quitar la leyenda (no es necesario ya que se aplicó showlegend=False en cada traza)
fig_line_g.update_layout(
    title='Tendencia Mensual de la Frescura',
    title_font=dict(family="Roboto", size=20, color="black"),
    xaxis_title='Mes',
    yaxis_title='Frescura'
)

st.plotly_chart(fig_line_g)
```

Análisis y Segmentación por Categorías Específicas: Se analiza la frescura de los productos en función de diversas categorías, como tipo de empaque, ciudad, retornabilidad y canal de venta, permitiendo un análisis más detallado y personalizado.

```python
with st.expander("📖 Ver explicación"):
    st.markdown(explicacion_retornabilidad)

fig_line_r = go.Figure()
datos_linea_r = filtered_df.groupby(['mes_encuesta', 'Retornable'])['Edad_producto'].agg(['mean', 'std']).reset_index()

for retornable in datos_linea_r['Retornable'].unique():
    df_ret = datos_linea_r[datos_linea_r['Retornable'] == retornable]
    # Definir colores de fillcolor para la banda de error basados en el empaque
    fillcolor_r = {
        'SI': 'rgba(0,97,55, 0.4)',
        'No': 'rgba(195,192,0, 0.4)'
    }.get(retornable, 'rgba(0, 0, 0, 0.4)')

    # Línea de la media para cada empaque
    fig_line_r.add_trace(go.Scatter(
        x=df_ret['mes_encuesta'].astype(str),
        y=df_ret['mean'],
        mode='lines+markers',
        name=f'{retornable}',
        line=dict(color=fillcolor_r)
    ))

    # Agregar banda de error para la desviación estándar
    fig_line_r.add_trace(go.Scatter(
        x=pd.concat([df_ret['mes_encuesta'], df_ret['mes_encuesta'][::-1]]).astype(str),
        y=pd.concat([df_ret['mean'] + df_ret['std']/10, (df_ret['mean'] - df_ret['std']/10  )[::-1]]),
        fill='toself',
        fillcolor=fillcolor_r,
        line=dict(color='rgba(255,255,255,0)'),
        showlegend=False,
        hoverinfo="skip"
    ))
```

Interactividad y Navegación: La aplicación hace uso de pestañas para organizar los análisis por categorías, botones para revelar explicaciones adicionales, y seleccionadores y multiseleccionadores en la barra lateral para filtros personalizados, lo que contribuye a una experiencia de usuario interactiva y atractiva.

En resumen, esta aplicación Streamlit proporciona una herramienta interactiva y visual para analizar la frescura de los productos, permitiendo a los usuarios explorar los datos a través de diferentes filtros y visualizaciones, con el objetivo de obtener insights útiles sobre la calidad y aceptación de los productos en el mercado.