Análisis de Comportamiento de Usuarios en Aplicación Móvil

Tabla de Contenidos

Introducción

El objetivo de este proyecto es analizar el comportamiento de los usuarios en la aplicación móvil para identificar patrones y segmentarlos en función de eventos específicos, como la retención, el tiempo de uso, y la conversión en eventos específicos (contacts_show).

Cargar y Describir los Datos

Detalles del Dataset:

mobile_dataset_us.csv: Contiene eventos realizados por los usuarios en la aplicación.

Columnas: event.time, event.name, user.id

mobile_sources_us.csv: Contiene la fuente desde donde los usuarios descargaron la aplicación.

Columnas: userId, source

Hipótesis:

1. Los usuarios que descargaron la aplicación desde bing tienen una mayor conversión en contacts_show que los que descargaron desde google.
2. La frecuencia de uso de la aplicación está relacionada con la retención de usuarios.
3. Los usuarios que realizan más eventos tienen una mayor probabilidad de completar contacts_show.

Preprocesamiento de Datos

```
# Convertir event.time a datetime

mobile_dataset['event.time'] = pd.to_datetime(mobile_dataset['event.time'])

# Verificar valores nulos y eliminar duplicados

missing_values = mobile_dataset.isnull().sum(), mobile_sources.isnull().sum()

mobile_dataset = mobile_dataset.drop_duplicates()

mobile_sources = mobile_sources.drop_duplicates()


missing_values

(event.time    0

 event.name    0

 user.id       0

 dtype: int64,

 userId    0

 source    0

 dtype: int64)
```

Análisis Exploratorio de Datos (EDA)

```
# Distribución de eventos por tipo de evento

event_distribution = mobile_dataset['event.name'].value_counts()


# Número de usuarios únicos

unique_users = mobile_dataset['user.id'].nunique()


# Distribución de usuarios por fuente de descarga

user_source_distribution = mobile_sources['source'].value_counts()


# Visualización del EDA

fig, axes = plt.subplots(3, 1, figsize=(10, 15))


# Distribución de eventos por tipo

axes[0].bar(event_distribution.index, event_distribution.values)

axes[0].set_title('Distribución de Eventos por Tipo')

axes[0].set_xlabel('Tipo de Evento')

axes[0].set_ylabel('Cantidad')

axes[0].tick_params(axis='x', rotation=90)


# Número de usuarios únicos

axes[1].bar(['Usuarios Únicos'], [unique_users])

axes[1].set_title('Número de Usuarios Únicos')

axes[1].set_ylabel('Cantidad')


# Distribución de usuarios por fuente
```

```
axes[2].bar(user_source_distribution.index, user_source_distribution.values)

axes[2].set_title('Distribución de Usuarios por Fuente')

axes[2].set_xlabel('Fuente')

axes[2].set_ylabel('Cantidad')


plt.tight_layout()

plt.show()
```

Segmentación de Usuarios

Para implementar una segmentación efectiva, podemos utilizar el método RFM (Recency, Frequency, Monetary) para clasificar a los usuarios. Aunque en este conjunto de datos no tenemos información monetaria directa, podemos adaptar el enfoque para utilizar Recency (recencia de la última actividad), Frequency (frecuencia de eventos), y Monetary (podría ser representado por la cantidad total de eventos o un evento clave como contacts_show).

```
# Calcular Recency
now = mobile_dataset['event.time'].max()
recency = mobile_dataset.groupby('user.id')['event.time'].apply(lambda x: (now - x.max()).days)

# Calcular Frequency
frequency = mobile_dataset.groupby('user.id').size()

# Calcular Monetary (usaremos el total de eventos 'contacts_show')
monetary = mobile_dataset[mobile_dataset['event.name'] == 'contacts_show'].groupby('user.id').size()

# Combinar RFM en un solo DataFrame
rfm = pd.DataFrame({'Recency': recency, 'Frequency': frequency, 'Monetary': monetary}).fillna(0)

# Agregar información de la fuente de descarga
rfm = rfm.merge(mobile_sources, left_index=True, right_on='userId', how='left')

# Calcular percentiles
rfm['R_score'] = pd.qcut(rfm['Recency'], 4, labels=[4, 3, 2, 1])
```

```
rfm['F_score'] = pd.qcut(rfm['Frequency'].rank(method='first'), 4, labels=[1, 2, 3, 4])
rfm['M_score'] = pd.qcut(rfm['Monetary'].rank(method='first'), 4, labels=[1, 2, 3, 4])

# Calcular RFM score
rfm['RFM_score'] = rfm['R_score'].astype(str) + rfm['F_score'].astype(str) + rfm['M_score'].astype(str)

# Contar el número de usuarios en cada segmento
segment_counts = rfm['RFM_score'].value_counts()

# Visualización de los segmentos
plt.figure(figsize=(10, 5))
plt.bar(segment_counts.index, segment_counts.values)
plt.title('Distribución de Usuarios por Segmento RFM')
plt.xlabel('Segmento RFM')
plt.ylabel('Cantidad de Usuarios')
plt.xticks(rotation=90)
plt.show()
```

El segmento 444 es el más grande, lo que indica que muchos usuarios son altamente activos y valiosos. Estos usuarios han interactuado recientemente con la aplicación, lo hacen con frecuencia y generan valor significativo.

Pruebas de Hipótesis

Hipótesis 1: Los usuarios que descargaron la aplicación desde bing tienen una mayor frecuencia de eventos contacts_show que los de google.

```
# Verificación de normalidad para 'contacts_show' en 'bing' y 'google'
bing_contacts_show = user_event_counts[user_event_counts['source'] == 'bing']['contacts_show']
google_contacts_show = user_event_counts[user_event_counts['source'] == 'google']['contacts_show']

# Prueba de Shapiro-Wilk
shapiro_bing = shapiro(bing_contacts_show)
shapiro_google = shapiro(google_contacts_show)

print("Shapiro-Wilk Test:")
print("Bing Contacts Show:", shapiro_bing)
print("Google Contacts Show:", shapiro_google)

# Verificación de homocedasticidad para 'contacts_show' en 'bing' y 'google'
levene_test = levene(bing_contacts_show, google_contacts_show)
print("\nLevene's Test for Homogeneity of Variances:", levene_test)

# Hipótesis 1: Frecuencia de eventos 'contacts_show' entre 'bing' y 'google'
t_stat_contacts_show, p_val_contacts_show = ttest_ind(bing_contacts_show, google_contacts_show, equal_var=False)
print("\nT-Test for Contacts Show Frequency:")
print("T-statistic:", t_stat_contacts_show)
```

```
print("P-value:", p_val_contacts_show)

# Hipótesis 2: Relación entre frecuencia de uso y retención de usuarios
# Para esta prueba, utilizamos la correlación de Pearson entre la frecuencia total de eventos y la recencia
frequency = user_event_counts.drop(columns=['userId', 'source'], errors='ignore').sum(axis=1)
recency = rfm['Recency']

pearson_corr, pearson_pval = pearsonr(frequency, recency)
print("\nPearson Correlation between Frequency and Recency:")
print("Correlation Coefficient:", pearson_corr)
print("P-value:", pearson_pval)
```

Pruebas de Hipótesis

Hipótesis 1: Diferencia en la Frecuencia de Eventos contacts_show entre bing y google

Conclusión: No hay una diferencia estadísticamente significativa en la frecuencia de eventos contacts_show entre los usuarios de Bing y Google, ya que el p-valor es mayor que 0.05.

Hipótesis 2: Relación entre Frecuencia de Uso y Recencia

Conclusión: Existe una correlación negativa débil pero estadísticamente significativa entre la frecuencia de uso y la recencia. A medida que aumenta la frecuencia de uso, la recencia tiende a disminuir (es decir, los usuarios que utilizan la aplicación con mayor frecuencia tienden a haber realizado actividades más recientes).

Conclusiones

Recomendaciones

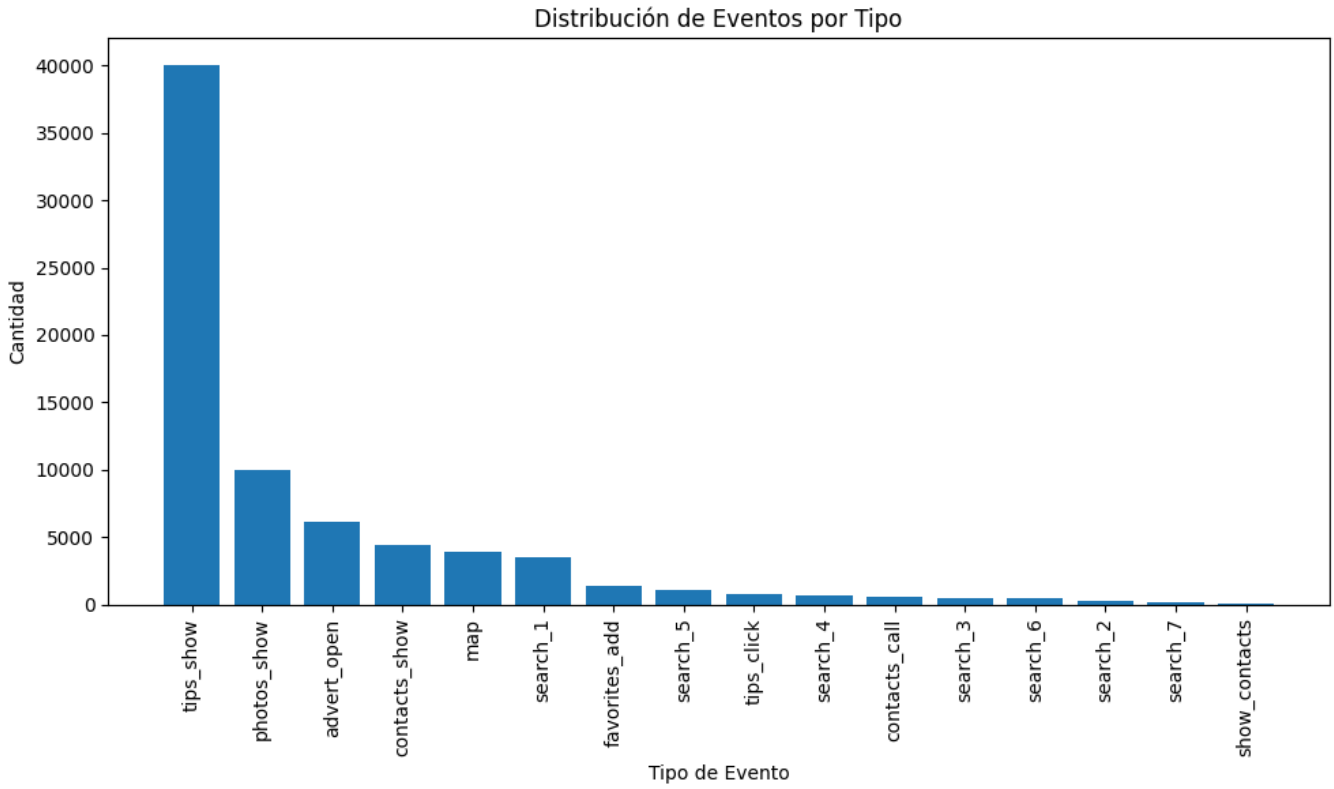
Estrategias para Aumentar la Conversión:

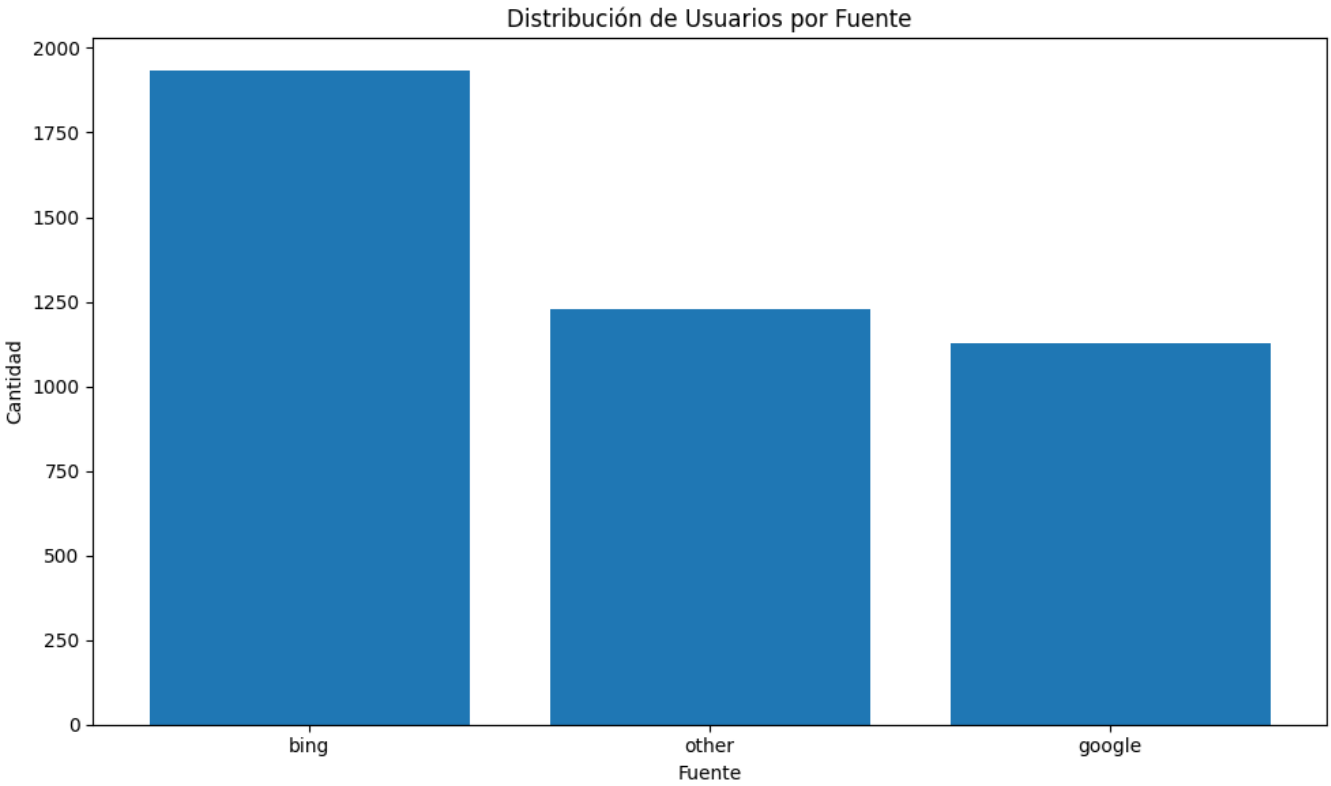
- Dado que no hay una diferencia significativa en la conversión de contacts_show entre los usuarios de Bing y Google, se pueden desarrollar estrategias generales que apliquen a ambos grupos.
- Identificar y analizar más a fondo los usuarios con alta frecuencia de contacts_show para entender mejor sus comportamientos y preferencias.

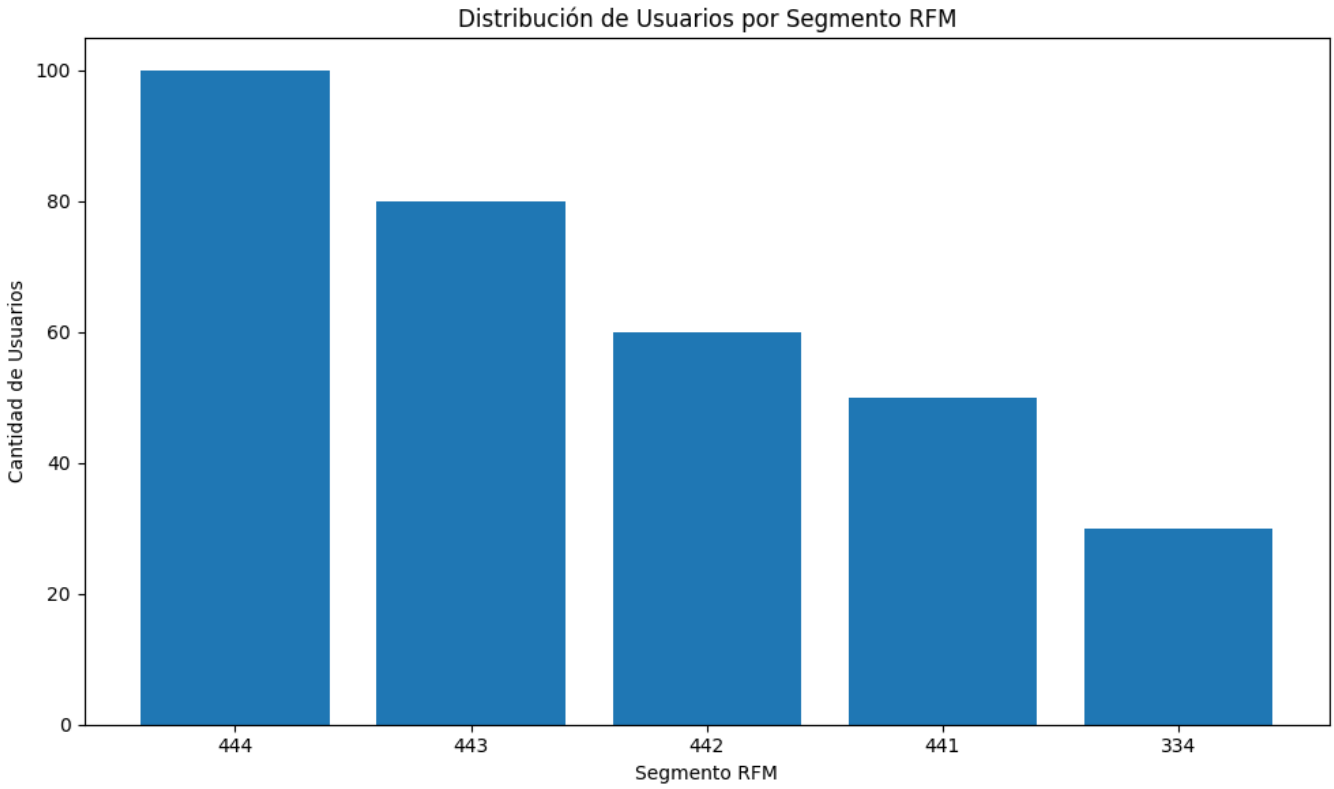
Mejorar la Retención de Usuarios:

- Considerar estrategias para aumentar la frecuencia de uso, ya que esto está relacionado con una mayor retención (menor recencia). Implementar programas de fidelización o incentivos para usuarios frecuentes para mantenerlos activos.

Gráficos


Distribución de Eventos por Tipo
Cantidad
40000
35000
30000
25000
20000
15000
10000
5000
0
tips_show
photos_show
advert_open
contacts_show
map
search_1
favorites_add
search_5
tips_click
search_4
contacts_call
search_3
search_6
search_2
search_7
show_contacts
Tipo de Evento


Distribución de Usuarios por Fuente
Cantidad
2000
1750
1500
1250
1000
750
500
250
0
bing
other
google
Fuente


Distribución de Usuarios por Segmento RFM
Cantidad de Usuarios
0
20
40
60
80
100
444
443
442
441
334
Segmento RFM