Project Business Statistics: E-news Express

Business Context

The advent of e-news, or electronic news, portals has offered us a great opportunity to quickly get updates on the day-to-day events occurring globally. The information on these portals is retrieved electronically from online databases, processed using a variety of software, and then transmitted to the users. There are multiple advantages of transmitting new electronically, like faster access to the content and the ability to utilize different technologies such as audio, graphics, video, and other interactive elements that are either not being used or aren't common yet in traditional newspapers.

E-news Express, an online news portal, aims to expand its business by acquiring new subscribers. With every visitor to the website taking certain actions based on their interest, the company plans to analyze these actions to understand user interests and determine how to drive better engagement. The executives at E-news Express are of the opinion that there has been a decline in new monthly subscribers compared to the past year because the current webpage is not designed well enough in terms of the outline & recommended content to keep customers engaged long enough to make a decision to subscribe.

[Companies often analyze user responses to two variants of a product to decide which of the two variants is more effective. This experimental technique, known as A/B testing, is used to determine whether a new feature attracts users based on a chosen metric.]

Objective

The design team of the company has researched and created a new landing page that has a new outline & more relevant content shown compared to the old page. In order to test the effectiveness of the new landing page in gathering new subscribers, the Data Science team conducted an experiment by randomly selecting 100 users and dividing them equally into two groups. The existing landing page was served to the first group (control group) and the new landing page to the second group (treatment group). Data regarding the interaction of users in both groups with the two versions of the landing page was collected. Being a data scientist in E-news Express, you have been asked to explore the data and perform a statistical analysis (at a significance level of 5%) to determine the effectiveness of the new landing page in gathering new subscribers for the news portal by answering the following questions:

1. Do the users spend more time on the new landing page than on the existing landing page?
2. Is the conversion rate (the proportion of users who visit the landing page and get converted) for the new page greater than the conversion rate for the old page?
3. Does the converted status depend on the preferred language? [Hint: Create a contingency table using the pandas.crosstab() function]
4. Is the time spent on the new page the same for the different language users?

Data Dictionary

The data contains information regarding the interaction of users in both groups with the two versions of the landing page.

1. user_id - Unique user ID of the person visiting the website
2. group - Whether the user belongs to the first group (control) or the second group (treatment)
3. landing_page - Whether the landing page is new or old
4. time_spent_on_the_page - Time (in minutes) spent by the user on the landing page
5. converted - Whether the user gets converted to a subscriber of the news portal or not
6. language_preferred - Language chosen by the user to view the landing page

Import all the necessary libraries

```
# Installing the libraries with the specified version.
!pip install numpy>=1.25.2 pandas>=1.5.3 matplotlib>=3.7.1 seaborn>=0.13.1 scipy>=1.11.4 -q --user
```

Note: *After running the above cell, kindly restart the notebook kernel and run all cells sequentially from the start again.*

```
# Import Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots

import plotly.io as pio
pio.renderers.default = "notebook_connected"
```

```
# Mount Drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Load the Dataset

```
# Load Dataset
df = pd.read_csv('/content/drive/MyDrive/Business Statistics Course/abtest.csv')
```

```
# Custom Color Set
ElleSet = [
    (102/255, 194/255, 165/255),  # Muted Green
    (214/255, 95/255, 95/255),    # Muted Red
    (141/255, 160/255, 203/255),  # Soft Blue
    (130/255, 198/255, 226/255),  # Muted Blue
    (166/255, 216/255, 84/255),   # Lime Green
    (230/255, 196/255, 148/255),  # Beige
    (179/255, 179/255, 179/255),  # Neutral Gray
    (255/255, 217/255, 47/255)    # Yellow
]

# Function to apply ElleSet globally in Seaborn
def use_ElleSet():
    sns.set_palette(ElleSet)
```

```
# Remove Future Warnings
import warnings

# Suppress all FutureWarnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

Exploratory Data Analysis

- Data Overview
  - Viewing the first and last few rows of the dataset
  - Checking the shape of the dataset
  - Getting the statistical summary for the variables
- Check for missing values
- Check for duplicates

View data head and tail

```
# EDA Head
df.head()
```

	user_id	group	landing_page	time_spent_on_the_page	converted	language_preferred
0	546592	control	old	3.48	no	Spanish
1	546468	treatment	new	7.13	yes	English
2	546462	treatment	new	4.40	no	Spanish
3	546567	control	old	3.02	no	French
4	546459	treatment	new	4.75	yes	Spanish

- The dataset contains the unique user ids of the visitors, which landing page they viewed, time spent on the page, whether or not they converted to subscriber, and the language preference of the visitor.

```
# EDA Tail
df.tail()
```

	user_id	group	landing_page	time_spent_on_the_page	converted	language_preferred
95	546446	treatment	new	5.15	no	Spanish
96	546544	control	old	6.52	yes	English
97	546472	treatment	new	7.07	yes	Spanish
98	546481	treatment	new	6.20	yes	Spanish
99	546483	treatment	new	5.86	yes	English

Shape of the dataset

```
# EDA Shape
df.shape
```

```
(100, 6)
```

- The dataset contains information about a sample of 100 site visitors.

Review of the datatypes of the columns of the dataset

```
# datatypes and general information from the dataset
df.info(memory_usage=False)
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 100 entries, 0 to 99
Data columns (total 6 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   user_id                100 non-null    int64  
 1   group                  100 non-null    object 
 2   landing_page           100 non-null    object 
 3   time_spent_on_the_page  100 non-null    float64
 4   converted              100 non-null    object 
 5   language_preferred     100 non-null    object 
dtypes: float64(1), int64(1), object(4)
```

There are 6 columns.

- The user_id column is a numerical datatype = int64. Consider amending to categorical type to avoid analysis in mathematical operations.
- group, landing page, converted, and language_preferred are catergorical datatypes. *time_spent_on_the_page is a numerical data type = float64. Consider amending to boolean datatype for mathematical operations and statistical analysis.

- There are no missing values in the dataset.

```python
# Check for dupilicates in dataset
duplicate_count = df.duplicated().sum()
print(f"Number of duplicate rows: {duplicate_count}")
```

```
Number of duplicate rows: 0
```

- The dataset contains no duplicates.

Statistical Summary of the dataset

```python
# Dataset Statistical Summary: Categorical
df.describe(include=['object'])
```

	group	landing_page	converted	language_preferred
count	100	100	100	100
unique	2	2	2	3
top	control	old	yes	Spanish
freq	50	50	54	34

- Group: As anticipated, evenly split between the control (old page) and treatment (new page).
- Landing Page: As designed, even spilt between old and new.
- Converted: 'yes' appears most frequently, meaning more users converted than did not.
- Language Preference: Spanish appears to be the most preferred language. Additional analysis required to confirm.

```python
# Dataset Statistical Summary: Numerical
df.describe(include=['float']) #int not included as user_id is not intended to be a numerical value, it is an identifer.
```

	time_spent_on_the_page
count	100.000000
mean	5.377800
std	2.378166
min	0.190000
25%	3.880000
50%	5.415000
75%	7.022500
max	10.710000

Overall:

- Count: There are 100 observations.
- Mean time spent on the page is ~5.38 minutes.
- Standard Deviation is ~2.38 minutes
- Minimum and Maximum Values: ~.19 to ~10.1 minutes, respectively.

Quartile Insights:

1. 25% of users (Q1) spent less than ~3.88 minutes on the page

- This suggests that a significant portion of users left quickly.
- We may want to investigate whether this group had a lower conversion rate.

2. The median (Q2) is 5.41 minutes

- Half of all users spent less than ~5.41 minutes on the page.
- This is useful for setting a benchmark for engagement.

3. 75% of users (Q3) spent less than ~7.02 minutes

- Only 25% of users spent more than 7 minutes.
- If conversion is higher in this group, it suggests that longer engagement leads to higher conversion.

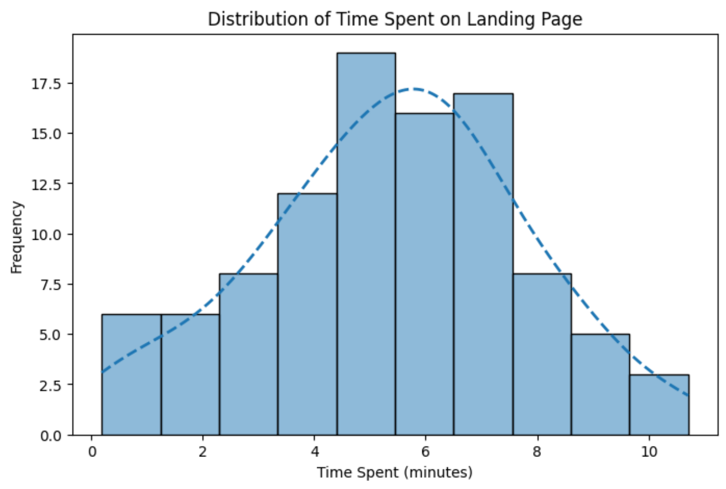
4. Potential Drop-Off Between Q1 and Q2

- The gap between Q1 (3.88 min) and Q2 (5.41 min) suggests that many users either leave early or stay longer.
- A follow-up question: Do people who stay beyond Q2 convert at a higher rate?

Univariate Analysis

Shape of Distribution

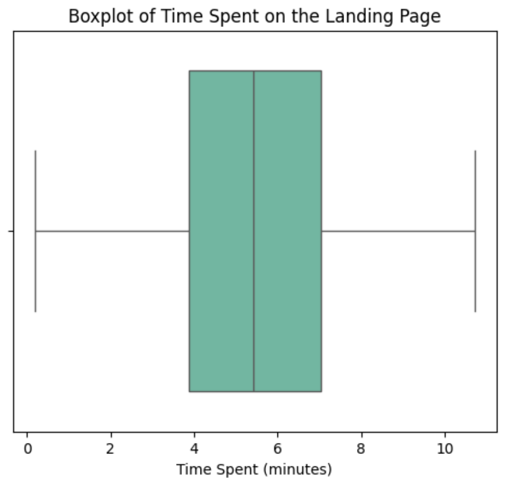
```
# Shape of Distribution
plt.figure(figsize=(8,5))
ax = sns.histplot(df['time_spent_on_the_page'], bins=10, kde=True, line_kws={'linewidth': 2, 'linestyle': '--'})

plt.title("Distribution of Time Spent on Landing Page")
plt.xlabel("Time Spent (minutes)")
plt.ylabel("Frequency")
plt.show()
```



- The distribution is approximately normal, with a peak around 5-7 minutes, indicating most users spend this amount of time on the landing page.
- There is a slight right skew, meaning some users spend longer than the average, but these were fewer in number.
- The presence of a smooth KDE curve suggests a well-defined engagement pattern where users tend to cluster around the central time range.
- Suggest further analysis to explore whether time spent correlates with conversion rates.

Time Spent on Landing Page

```
#Boxplot analysis of time spent on landing page
plt.figure(figsize=(6,5))
sns.boxplot(x=df['time_spent_on_the_page'], color=ElleSet[0])

plt.title("Boxplot of Time Spent on the Landing Page")
plt.xlabel("Time Spent (minutes)")
plt.show()
```



- The median time spent is around 5-6 minutes, indicating that half of the users spent at least this amount of time on the page.
- Interquartile Range (IQR) appears to span from about 4-7 minutes, meaning most users engaged within this range.
- There are no extreme outliers, suggesting that time spent is fairly consistent across users without significant anomalies.
- Next Step: Recommend comparison of time spent between the old and new landing pages to determine if the redesign had an impact on engagement.

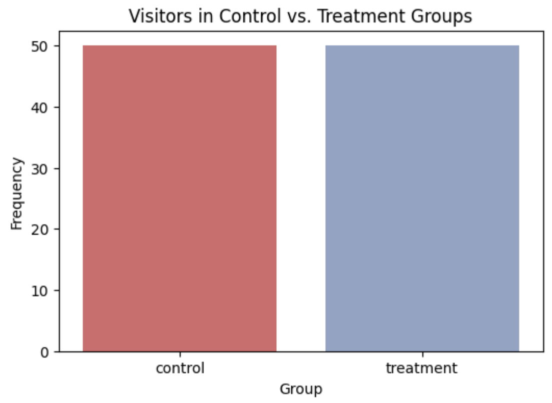
Visitor Group

```
df['group'].value_counts()
```

	count
group	
control	50
treatment	50

dtype: int64

```
#A/B Test Group Distribution
plt.figure(figsize=(6,4))

# Select colors from ElleSet
selected_colors = [ElleSet[1], ElleSet[2]]
sns.countplot(x=df['group'], palette=selected_colors)

# add labels
plt.title("Overview: Visitor Group")
plt.xlabel("Group")
plt.title("Visitors in Control vs. Treatment Groups")
plt.xlabel("Group")
plt.ylabel("Frequency")
plt.show()
```



- The control and treatment groups are evenly distributed, confirming a balanced A/B test setup.
- This ensures that any differences in conversion rate can be attributed to the landing page design rather than uneven group sizes.
- Equal sample sizes reduces bias, ensuring statistical validity when comparing conversion rates.
- Recommend further analysis of conversion rates within each group to determine whether the new landing page had a significant impact.
- If conversion rates differ significantly, we can infer that design changes influenced user behavior rather than random variation.

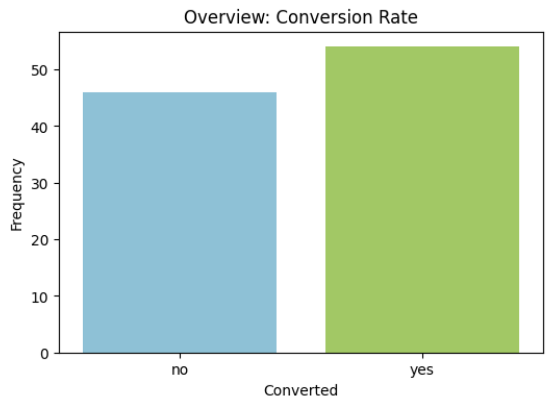
Conversion Rate

```
df['converted'].value_counts()
```

	count
converted	
yes	54
no	46

dtype: int64

```
# Conversion Rate
plt.figure(figsize=(6,4))

# Select colors from ElleSet
selected_colors = [ElleSet[3], ElleSet[4]]
sns.countplot(x=df['converted'], palette=selected_colors)

# add labels
plt.title("Overview: Conversion Rate")
plt.xlabel("Converted")
plt.ylabel("Frequency")

plt.show()
```



- There were more visitors who converted (yes) overall than did not (no), indicating a strong overall conversion rate.

This suggests that the landing pages are generally effective at driving subscriptions.

- Further analysis is needed to see which page performed better.

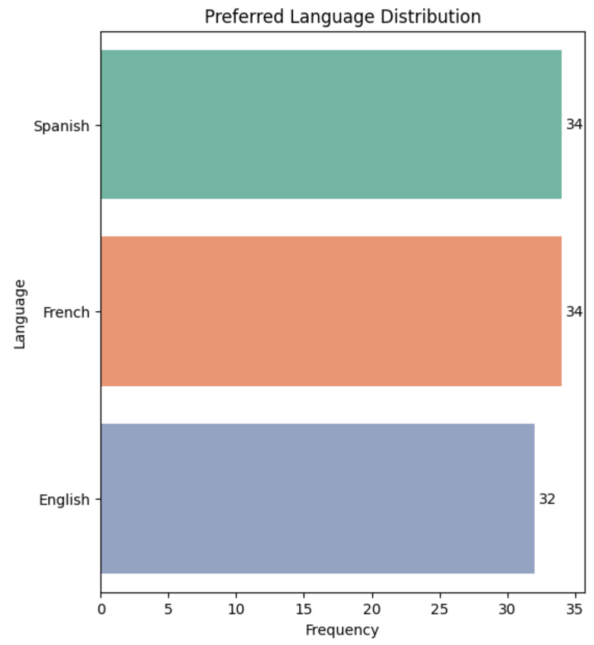
Language Preference

```
df['language_preferred'].value_counts()
```

language_preferred	count
Spanish	34
French	34
English	32

dtype: int64

```
# Language Preference Distribution
plt.figure(figsize=(6,7))  # Adjust figure size for better readability
ax = sns.countplot(y=df['language_preferred'], order=df['language_preferred'].value_counts().index, palette="Set2")

plt.title("Preferred Language Distribution")
plt.ylabel("Language")
plt.xlabel("Frequency")

# Add count labels on bars
for p in ax.patches:
    ax.annotate(f'{int(p.get_width())}',
                (p.get_width() + 0.3, p.get_y() + p.get_height()/2),
                ha='left', va='center', fontsize=10)

plt.show()
```



- Even language distribution: Spanish and French have the highest number of visitors, 34 each. English is slightly lower with 32. This suggests a balanced multilingual audience.
- Implications for Engagement & Conversion: Since no single language overwhelmingly dominates, localized content across all three languages may be equally important for user engagement.
- Next Step: Investigate whether conversion rates differ by language preference- this could reveal if a certain language group is more likely to subscribe.

Bivariate Analysis

Time Spent by Landing Page Version

```
# Interactive Swarm Plot Time Spent by Landing Page Version
fig = px.strip(df,
               x="landing_page",
               y="time_spent_on_the_page",
               color="landing_page",
               color_discrete_sequence=['rgb(166, 216, 84)', 'rgb(255, 153, 102)'],
               title="Interactive Swarm Plot: Time Spent by Landing Page Version",
               hover_data={'time_spent_on_the_page': ':.2f'}
              )

fig.update_layout(
    title_x=0.5,
    xaxis_title="Landing Page Version",
    yaxis_title="Time Spent (minutes)",
    showlegend=False
)

fig.show()
```

- The old page shows a highly dispersed pattern, with users scattered across all time ranges.
- Demonstrates inconsistent engagement—some users leave almost immediately, while others stay much longer.
- The new page shows tighter clustering, with most users engaging for 5-7 minutes.
- Fewer users drop off extremely early.
- Distinct outliers exist on the new page, but no clear secondary cluster.
- If multiple peaks appeared, it might suggest different user behaviors (e.g., skimmers vs. deep readers), but here we mostly see a single main engagement group with some outliers.

Business Insights:

- New landing page reduces erratic engagement patterns, keeping users more consistently engaged.
- No evidence of multiple distinct behavior groups—most users follow a similar engagement pattern.
- Extreme outliers still exist—further refinement in Call to Action (CTA) placement or content could help improve engagement.

Time Spent by Conversion Status

```
# Interactive Boxplot Time Spent by Conversion Status

# Select colors from ElleSet as strings
selected_colors = ['rgb(166, 216, 84)', 'rgb(255, 102, 102)']

fig = px.box(df,
             x="converted",
             y="time_spent_on_the_page",
             color="converted",
             color_discrete_sequence=selected_colors,
             title="Interactive Boxplot: Time Spent by Conversion Status",
             boxmode="group",
             points="all",
             hover_data={'time_spent_on_the_page':':.2f',
                         'converted':False}
             )

# Update x-axis labels for clarity
```

```python
fig.update_layout(
    xaxis_title="Conversion Status",
    yaxis_title="Time Spent (minutes)",
    xaxis=dict(tickmode="array", tickvals=[0, 1], ticktext=["Not Subscribed", "Subscribed"]),
    title_x=0.5,
    showlegend=False
)

fig.show()
```

- Subscribers engage longer (median 6-7 min) vs. non-subscribers (4 min), reinforcing that longer engagement boosts conversion.
- Early drop-off issue: Many non-subscribers leave within 2 minutes, suggesting a potential content or CTA gap.
- Greater variance among subscribers: Some convert quickly, while others take longer, indicating different decision-making patterns.
- Outliers exist in both groups, but high-end outliers are more common among subscribers, suggesting deep engagement before conversion.

Follow-Up:

Assess whether landing page version impacts engagement time per conversion status.

```python
# Interactive KDE Plot: Time Spent by Conversion Status

from scipy.stats import gaussian_kde

# Select colors from ElleSet for consistency
selected_colors = ['rgb(141, 160, 203)', 'rgb(255, 102, 102)']

# Prepare KDE data
x_values = np.linspace(df["time_spent_on_the_page"].min(), df["time_spent_on_the_page"].max(), 200)

# Compute KDE for each group
kde_subscribed = gaussian_kde(df[df['converted'] == 'yes']['time_spent_on_the_page'])(x_values)
kde_not_subscribed = gaussian_kde(df[df['converted'] == 'no']['time_spent_on_the_page'])(x_values)

# Create DataFrame for Plotly
kde_df = pd.DataFrame({
    "Time Spent (minutes)": np.tile(x_values, 2),
    "Density": np.concatenate([kde_subscribed, kde_not_subscribed]),
    "Conversion Status": np.repeat(["Subscribed", "Not Subscribed"], len(x_values))
})

# Create interactive KDE plot
fig = px.line(kde_df,
              x="Time Spent (minutes)",
              y="Density",
              color="Conversion Status",
              color_discrete_sequence=selected_colors,
              title="Interactive KDE Plot: Time Spent by Conversion")

# Update hover format to show only 2 decimal places
fig.update_traces(hovertemplate="Time Spent: %{x:.2f} min<br>Density: %{y:.2f}")

fig.update_layout(
    title_x=0.5,
    xaxis_title="Time Spent (minutes)",
    yaxis_title="Density",
    showlegend=True
)

fig.show()
```

- Users who convert stay engaged significantly longer than those who don’t.
- The converted user density peaks around 6-7 minutes, while non-subscribers drop off closer to 4 minutes.
- There’s a sharp decline in density for non-subscribers after 4 minutes.
- This suggests that if users haven’t engaged past this threshold, they are much less likely to convert.
- Subscribers show a wider range of engagement times, while non-subscribers are more concentrated in early exits.
- Some subscribers take longer than 7 minutes to decide, meaning prolonged engagement can still lead to conversions.

Business Insights:

- The 4-minute mark is a critical drop-off point—users who leave before this are unlikely to subscribe.
- Keeping users engaged past 4 minutes should be a priority—content restructuring or interactive elements may help.
- Next Step: Analyze this by landing page version to see if the new design improves retention.

Time Spent by Preferred Language

```
# Interactive Boxplot Time Spent by Preferred Language
# Select colors from ElleSet as strings
selected_colors = ['rgb(141, 160, 203)', 'rgb(130, 198, 226)', 'rgb(166, 216, 84)']

fig = px.box(df,
             y="language_preferred",
             x="time_spent_on_the_page",
             color="language_preferred",
             color_discrete_sequence=selected_colors,
             title="Interactive Boxplot: Time Spent by Preferred Language",
             boxmode="group",
             points="all",
             hover_data={'time_spent_on_the_page':':.2f',
                         'language_preferred':False,
                         'converted':True},
             )

fig.update_layout(xaxis_title="Time Spent (minutes)",
                  yaxis_title="Preferred Language",
                  showlegend=False,
                  xaxis_range=[0, df['time_spent_on_the_page'].max() + 1]
                  )

fig.show()
```

- French-speaking users show the widest spread of time spent, ranging from close to 0 up to 10+ minutes, indicating high variability in engagement.
- Some users leave quickly, while others stay significantly longer.
- Spanish-speaking users have the most concentrated engagement, with a tighter IQR around 5-7 minutes.
- This suggests more consistent engagement behavior within this language group.
- English-speaking users fall between the other two groups, with moderate spread and a slightly lower median than Spanish users.
- Their time spent varies more than Spanish speakers but is less erratic than French speakers.

Business Insights:

- Spanish-speaking users engage most consistently—content in Spanish may already be well-optimized.
- French users display extreme variability—further analysis on content alignment or cultural differences could help.
- Consider A/B testing CTA placements and content modifications for English and French speakers to improve retention.

```
# Interactive Density Heatmap Time Spent by Language Preference

fig = px.density_heatmap(df,
                         x="time_spent_on_the_page",
                         y="language_preferred",
                         color_continuous_scale=[(0, "white"), (1, "rgb(102, 194, 165)")],
                         title="Interactive Density Heatmap: Time Spent by Preferred Language",
                         hover_data={'time_spent_on_the_page':':.2f',
                                     'language_preferred':False,
                                     'converted':True}
                         )

fig.update_layout(
    title="Interactive Density Heatmap: Time Spent by Preferred Language",
    title_x=0.5,
    xaxis_title="Time Spent (minutes)",
    yaxis_title="Preferred Language",
    coloraxis_colorbar_title="Density"
)

fig.show()
```

- Spanish speakers have the highest density at 6-8 minutes – reinforcing stronger engagement in this group.
- English and French users show more spread-out engagement – but English has higher density earlier (around 4 minutes).
- Density fades past 10 minutes, suggesting most users make a decision before that point (either converting or leaving).

Conversion Rate by Landing Page

```python
# Interactive Stacked Bar Chart
# Prepare data for conversion status by landing page
conversion_summary = df.groupby(['landing_page', 'converted']).size().reset_index(name='count')

# Normalize the count to proportions within each landing page group
conversion_summary['percentage'] = conversion_summary.groupby('landing_page')['count'].transform(lambda x: x / x.sum())

# Create an interactive stacked bar chart with correct proportions
fig = px.bar(conversion_summary,
             x="landing_page",
             y="percentage",
             color="converted",
             barmode="relative",
             text_auto='.0%',
             title="Interactive Conversion Rate Comparison: Old vs. New Landing Page",
             color_discrete_sequence=['rgb(141, 160, 203)', 'rgb(130, 198, 226)']
            )

# Format layout
fig.update_layout(
    title_x=0.5,
    xaxis_title="Landing Page Version",
    yaxis_title="Proportion of Visitors",
    yaxis=dict(tickformat=".0%"),
    showlegend=True
)

fig.show()
```

- The new landing page has a higher conversion rate, with 66% of visitors subscribing, compared to 42% on the old page.
- This suggests the new design is more effective at driving conversions.
- The old landing page has a higher proportion of non-converters (58%), indicating more user drop-offs.

- This reinforces concerns that the previous page was not engaging enough to retain users.
- Despite improvements, 34% of visitors still don’t convert on the new page.
- While the new design performs better, further refinements could push conversion rates even higher.

Business Insights:

- The new landing page is a clear improvement, but there’s room for optimization.
- Understanding why 34% of users still don’t convert could reveal additional refinements needed.

Next Step: Conduct statistical testing to confirm if this difference is statistically significant.

Statistical Analysis: Business Questions

1. Do the users spend more time on the new landing page than the existing landing page?

Perform Visual Analysis

```python
# Boxplot Landing Page vs. Time Spent
# Select colors from ElleSet in RGB format
selected_colors = ['rgb(141, 160, 203)', 'rgb(130, 198, 226)']

fig = px.box(df,
             x="landing_page",
             y="time_spent_on_the_page",
             color="landing_page",
             color_discrete_sequence=selected_colors,
             title="Interactive Boxplot: Time Spent by Landing Page Version",
             boxmode="group",
             points="all",
             hover_data={'time_spent_on_the_page':':.2f'}
             )

# Update Layout
fig.update_layout(
    title_x=0.5,
    xaxis_title="Landing Page Version",
    yaxis_title="Time Spent (minutes)",
    showlegend=False
)

fig.show()
```

- Users appear to spend more time on the new landing page, with a higher median (6 min) vs. the old page (4.5 min).
- The new page has a tighter IQR, indicating more consistent engagement, while the old page shows greater variability.
- Some users on the old page leave quickly, while others stay much longer.
- Outliers on the new page: A few users disengage early (less than 2 min), while others stay unusually long (as long as 10 min).

- Is the observed difference significant enough to conclude that the new page improves conversion rates? To detemine this, we will test the difference using a statistical test.

Step 1: Define the null and alternate hypotheses

The null and alternative hpyotheses can be formulated as:

> H_0 : (Null Hypothesis) The mean time spent on the new landing page is equal to the mean time spent on the old landing page.

> H_a : (Alternative Hypothesis) The mean time spent on the new landing page is greater than the mean time spent on the old landing page.

Let μ_1 and μ_2 be the mean time spent on the new landing page, and the mean time spent on the old landing page, respectively.

> $H_0 : \mu_1 = \mu_2$
> $H_a : \mu_1 > \mu_2$

Step 2: Select Appropriate test

Selection: The Two Independent Sample T-Test

Review Assumptions:

1. **Continuous Data:** The variable time spent on page is measured in minutes, which is a continuous numerical variable.
2. **Normally Distributed Populations:** If the sample size is large (n ≥ 30 per group), the Central Limit Theorem suggests that the sampling distribution of the mean will be approximately normal, even if the original data is not perfectly normal. Group size is greater than 30 per group.
3. **Independent Populations:** The two groups (old landing page users and new landing page users) were randomly assigned and do not overlap, ensuring independence.
4. **Unequal Population Standard Deviations:**We will need to conduct a Levene's test to look at homogeneity of variances.
5. **Random Sampling from the Population:** Users were randomly assigned to either the control (old page) or treatment (new page) group, supporting the validity of the test.

Confirm unequal population standard deviations

```
# Split data into two groups: old landing page vs. new landing page
old_page_times = df[df['landing_page'] == 'old']['time_spent_on_the_page']
new_page_times = df[df['landing_page'] == 'new']['time_spent_on_the_page']

# Perform Levene's test for equality of variances
stat, p_value = stats.levene(old_page_times, new_page_times)

# Display results
print(f"Levene's test statistic: {stat:.4f}")
print(f"P-value: {p_value:.4f}")

# Interpretation
alpha = 0.05
if p_value < alpha:
    print("Reject the null hypothesis: The variances are significantly different (unequal variances).")
else:
    print("Fail to reject the null hypothesis: The variances are not significantly different (equal variances).")
```

```
Levene's test statistic: 7.1137
P-value: 0.0090
Reject the null hypothesis: The variances are significantly different (unequal variances).
```

The Levene's test confirms our assumption that the population standard deviations are unequal. We can move forward with our test.

Step 3: Decide the significance level

As given in the problem statement, we select $\alpha = 0.05$.

Step 4: Collect and prepare data

```
# Create separate variables for time spent on each landing page version
time_spent_new = df[df['landing_page'] == 'new']['time_spent_on_the_page']
time_spent_old = df[df['landing_page'] == 'old']['time_spent_on_the_page']
```

```
# Calculate and print sample standard deviations
print('The sample standard deviation of the time spent on the new page is:', round(time_spent_new.std(), 2))
print('The sample standard deviation of the time spent on the old page is:', round(time_spent_old.std(), 2))

# Print confirmation message
print("Data has been collected and is ready for analysis.")
```

```
The sample standard deviation of the time spent on the new page is: 1.82
The sample standard deviation of the time spent on the old page is: 2.58
Data has been collected and is ready for analysis.
```

Step 5: Calculate the p-value

```
# Complete the code to import the required function
```

```python
from scipy.stats import ttest_ind

# Write the code to calculate the p-value
test_stat, p_value = ttest_ind(time_spent_new, time_spent_old, equal_var=False, alternative='greater')

# Print the p-value
print('The p-value is', p_value)
```

```
The p-value is 0.0001392381225166549
```

Step 6: Compare the p-value with α

```python
## Step 6: Compare p-value with significance level
alpha = 0.05  # Significance level

if p_value < alpha:
    print(f'As the p-value {p_value:.4f} is less than the level of significance ({alpha}), we reject the null hypothesis.')
    print("Conclusion: Users spend significantly more time on the new landing page than on the old landing page.")
else:
    print(f'As the p-value {p_value:.4f} is greater than the level of significance ({alpha}), we fail to reject the null hypothesis.')
    print("Conclusion: There is no significant difference in time spent between the old and new landing pages.")
```

```
As the p-value 0.0001 is less than the level of significance (0.05), we reject the null hypothesis.
Conclusion: Users spend significantly more time on the new landing page than on the old landing page.
```

Step 7: Draw inference

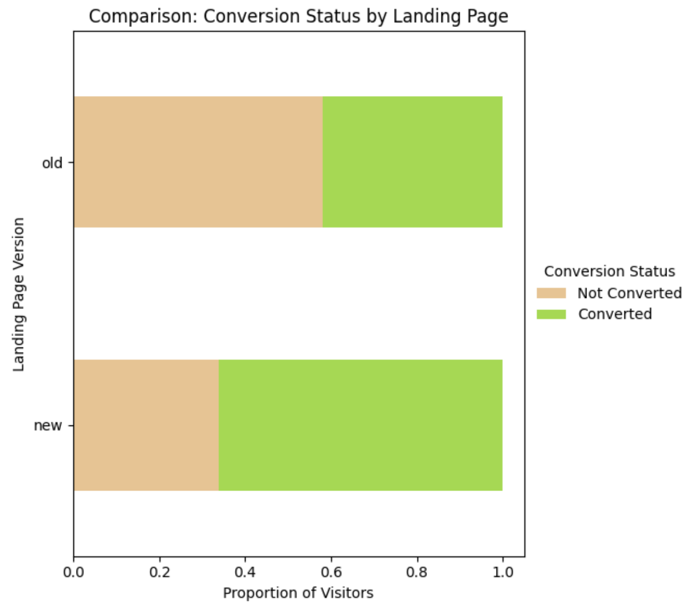
At a 5% significance level, we reject the null hypothesis. Hence, we have enough statistical evidence to conclude that the mean time spent on the new landing page is greater than the mean time spent on the old landing page.

A similar approach can be followed to answer the other questions.

2. Is the conversion rate (the proportion of users who visit the landing page and get converted) for the new page greater than the conversion rate for the old page?

Perform Visual Analysis

```python
# Create a stacked bar chart of conversion status by landing page

# Define colors from ElleSet for consistency
selected_colors = [ElleSet[5], ElleSet[4]]

# Create a stacked horizontal bar chart
ax = pd.crosstab(df['landing_page'], df['converted'], normalize='index').plot(
    kind="barh", figsize=(8,6), stacked=True, color=selected_colors
)

# Move legend completely outside the plot
plt.legend(["Not Converted", "Converted"], title="Conversion Status", loc="center left", bbox_to_anchor=(1, 0.5), frameon=False)

# Add labels and title
plt.xlabel("Proportion of Visitors")
plt.ylabel("Landing Page Version")
plt.title("Comparison: Conversion Status by Landing Page")

# Adjust layout to ensure the legend does not overlap
plt.tight_layout(rect=[0, 0, 0.85, 1])

plt.show()
```



- Higher conversion on the new page: A greater proportion of visitors convert on the new page compared to the old one.
- Improved effectiveness: The new design appears to better guide users toward subscribing.
- Clear shift in engagement: The increase in conversions suggests the updated content or layout is more compelling.
- Next Step: Verify statistical significance to confirm whether this difference is meaningful.

Step 1: Define the null and alternate hypotheses

The null and alternative hpyotheses can be formulated as:

> H_0 : (Null Hypothesis) The conversion rate for the new landing page is **equal to or lower than** the conversion rate for the old landing page.

> H_a : (Alternative Hypothesis) The conversion rate for the new landing page is **greater** than the conversion rate for the old landing page.

Let μ_1 and μ_2 be the proportion of users who converted on the **new landing page** and the **old landing page**, respectively.

> $H_0 : \mu_1 \le \mu_2$
> $H_a : \mu_1 > \mu_2$

Step 2: Select Appropriate test

Selection: 2-Sample z-Test

Review Assumptions:

1. **Categorical Data:** The conversion outcome ("yes" or "no") is a categorical variable, making a proportion-based test appropriate.
2. **Large Sample Size (Normality Assumption):** The sample size in each group is ≥ 30, and the Central Limit Theorem (CLT) ensures that the sampling distribution of the proportions is approximately normal.
3. **Independent Populations:** The two groups (old landing page users vs. new landing page users) were randomly assigned and do not overlap, ensuring independence.
4. **Random Sampling from the Population:** Users were randomly assigned to either the control (old page) or treatment (new page) group, supporting the validity of the test.
5. **Proportion-Based Comparison:** Since we are comparing the conversion rates (proportions) between two independent groups, a 2-sample Z-test for proportions is the correct statistical test.

Step 3: Decide the significance level

As given in the problem statement, we select $\alpha=0.05$.

Step 4: Collect and prepare data

```
# Calculate the number of converted users in the treatment group (new page)
new_converted = df[df['landing_page'] == 'new']['converted'].value_counts()['yes']

# Calculate the number of converted users in the control group (old page)
old_converted = df[df['landing_page'] == 'old']['converted'].value_counts()['yes']

# Total number of users in the control group (old page)
n_control = df['landing_page'].value_counts()['old']
```

```
# Total number of users in the treatment group (new page)
n_treatment = df['landing_page'].value_counts()['new']

# Print summary
print('The numbers of users served the new and old pages are {0} and {1}, respectively'.format(n_treatment, n_control))

# Print confirmation message
print("Data has been collected and is ready for analysis.")
```

```
The numbers of users served the new and old pages are 50 and 50, respectively
Data has been collected and is ready for analysis.
```

Step 5: Calculate the p-value

```
# Import the required function
from statsmodels.stats.proportion import proportions_ztest

# Calculate the p-value
test_stat, p_value = proportions_ztest([new_converted, old_converted],
                                       [n_treatment, n_control],
                                       alternative='larger')  # One-tailed test

# Print the p-value
print('The p-value is', p_value)
```

```
The p-value is 0.008026308204056278
```

Step 6: Compare the p-value with α

```
# Set the significance level
alpha = 0.05

# Print the conclusion based on the p-value
if p_value < alpha:
    print(f'As the p-value {p_value:.4f} is less than the level of significance ({alpha}), we reject the null hypothesis.')
    print('Conclusion: The new landing page has a significantly higher conversion rate than the old page.')
else:
    print(f'As the p-value {p_value:.4f} is greater than the level of significance ({alpha}), we fail to reject the null hypothesis.')
    print('Conclusion: There is no significant evidence that the new landing page has a higher conversion rate than the old page.')
```

```
As the p-value 0.0080 is less than the level of significance (0.05), we reject the null hypothesis.
Conclusion: The new landing page has a significantly higher conversion rate than the old page.
```

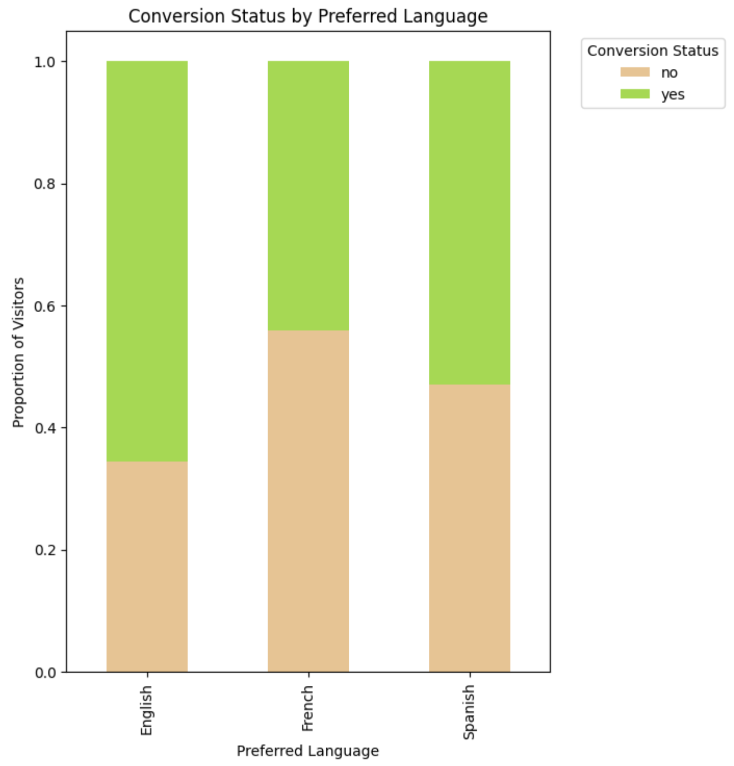
Step 7: Draw Inference

At a 5% significance level, we reject the null hypothesis. Hence, we have enough statistical evidence to conclude that the conversion rate for the new landing page is greater than that of the old landing page.

3. Are conversion and preferred language independent or related?

Perform Visual Analysis

```
# Visualize the relationship between conversion status and preferred language

import matplotlib.pyplot as plt
# Create a stacked bar chart to show conversion status across language preferences
pd.crosstab(df['language_preferred'], df['converted'], normalize='index').plot(
    kind="bar", figsize=(6,8), stacked=True, color=[ElleSet[5], ElleSet[4]]
)

# Add labels and legend
plt.xlabel("Preferred Language")
plt.ylabel("Proportion of Visitors")
plt.title("Conversion Status by Preferred Language")
plt.legend(["Not Converted", "Converted"], title="Conversion Status")

# Move legend outside the plot
plt.legend(title="Conversion Status", bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```



- Conversion patterns appear similar across languages: No significant visual differences suggest that language preference may not be a key driver of conversion.
- French-speaking users show a slightly lower conversion rate: A higher proportion of non-converted users among French speakers might indicate lower engagement or content relevance issues.
- English and Spanish speakers exhibit similar trends: Both groups display comparable proportions of converted and non-converted users, hinting that language may not strongly influence conversion.
- Statistical testing needed: While the visualization suggests independence, a formal test is required to determine if language preference and conversion status are statistically related.

Step 1: Define the null and alternate hypotheses

> H_0 : (Null Hypothesis) Conversion status is independent of the preferred language.

> H_a : (Alternative Hypothesis) Conversion status is dependent on the preferred language.

Let p_1, p_2, and p_3 represent the proportion of users who converted for each language (English, French, and Spanish), respectively.

$H_0 : p_1 = p_2 = p_3$ (The conversion rates across all preferred languages are the same) H_a : At least one p_i differs (At least one language group has a different conversion rate)

Step 2: Select Appropriate test

Selection: Chi Square Test for Independence

Review Assumptions:

1. **Categorical Variables:** Both preferred language and conversion status are categorical (Yes/No for conversion, and language groups).
2. **Expected Frequency ≥ 5:** Each cell in the contingency table should have an expected frequency of at least 5 to ensure validity of the Chi-Square approximation. *We will verify this by computing expected frequencies before proceeding.*
3. **Random Sampling:** The data collection method should ensure that individuals are randomly sampled and independent observations exist. Since users were randomly assigned to different landing pages and conversion data was recorded independently, this assumption seems reasonable.

Contingency Table: Verify Expected Frequencies

```python
 # Create a contingency table for conversion status vs. preferred language
contingency_table = pd.crosstab(df['language_preferred'], df['converted'])

# Check if all expected frequencies are greater than 5
if (contingency_table.values >= 5).all():
    print("Expected frequencies are all greater than 5. We can proceed with the Chi-Square test.")
else:
    print("Some expected frequencies are less than 5. The Chi-Square test assumptions may not hold.")

contingency_table
```

```
Expected frequencies are all greater than 5. We can proceed with the Chi-Square test.
```

converted	no	yes
language_preferred		
English	11	21
French	19	15
Spanish	16	18

Step 3: Decide the significance level

As given in the problem statement, we select α=0.05 .

Step 4: Collect and prepare data

```
# Contingency table for conversion status vs. preferred language
contingency_table = pd.crosstab(df['language_preferred'], df['converted'])

contingency_table
```

converted	no	yes
language_preferred		
English	11	21
French	19	15
Spanish	16	18

Step 5: Calculate the p-value

```
# Import the required function
from scipy.stats import chi2_contingency

# Perform the Chi-Square test
chi2, p_value, dof, exp_freq = chi2_contingency(contingency_table)

# Print the p-value
print('The p-value is', p_value)
```

```
The p-value is 0.2129888748754345
```

Step 6: Compare p-value with α

```
# Set the significance level
alpha = 0.05

# Print the conclusion based on the p-value
if p_value < alpha:
    print(f'As the p-value {p_value} is less than the level of significance ({alpha}), we reject the null hypothesis.')
    print('Conclusion: Conversion status and preferred language are not independent; there is a relationship between them.')
else:
    print(f'As the p-value {p_value} is greater than the level of significance ({alpha}), we fail to reject the null hypothesis.')
    print('Conclusion: We do not have enough statistical evidence to conclude that conversion status and preferred language are related.')
```

```
As the p-value 0.2129888748754345 is greater than the level of significance (0.05), we fail to reject the null hypothesis.
Conclusion: We do not have enough statistical evidence to conclude that conversion status and preferred language are related.
```

Checking Work

```
# Recalculate Chi-Square test to verify
from scipy.stats import chi2_contingency

chi2, p_value, dof, exp_freq = chi2_contingency(contingency_table)

# Print results again to verify
print(f'Chi-Square Statistic: {chi2}')
print(f'Degrees of Freedom: {dof}')
print(f'P-value: {p_value}')
```

```
Chi-Square Statistic: 3.0930306905370832
Degrees of Freedom: 2
P-value: 0.2129888748754345
```

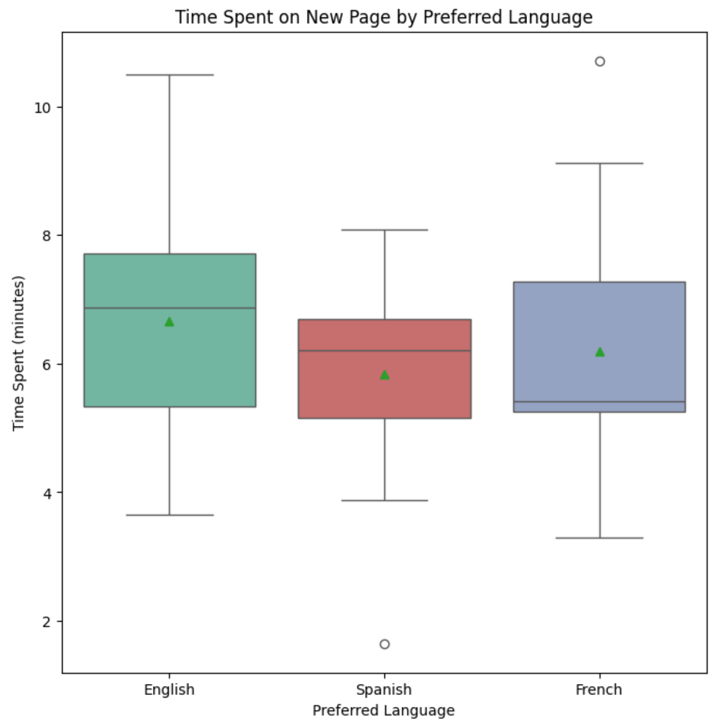
Step 7: Draw Inference

At a 5% significance level, we fail to reject the null hypothesis. Hence, we do not have enough statistical evidence to conclude that conversion status and preferred language are related.

4. Is the time spent on the new page the same for the different language users?

Perform Visual Analysis

```python
# Create a new DataFrame for users who got served the new page
df_new = df[df['landing_page'] == 'new']

# Select colors from ElleSet for consistency
selected_colors = [ElleSet[0], ElleSet[1], ElleSet[2]]

# Complete the code to visually plot the time spent on the new page for different language users
plt.figure(figsize=(8,8))
sns.boxplot(x=df_new['language_preferred'],
            y=df_new['time_spent_on_the_page'],
            showmeans=True,
            data=df_new,
            palette=selected_colors)

plt.xlabel("Preferred Language")
plt.ylabel("Time Spent (minutes)")
plt.title("Time Spent on New Page by Preferred Language")
plt.show()
```



- Median time spent is similar across languages: The central tendency appears close for all three language groups, suggesting no major differences in engagement.
- English users show a slightly higher average: The mean marker indicates English users may spend marginally more time on the new page compared to French and Spanish speakers.
- French users exhibit greater variability: A wider interquartile range (IQR) and a high outlier suggest some French users engage significantly longer, while others leave sooner.
- Outliers exist in all groups: Extreme cases are present across languages, though Spanish users have at least one notably low outlier.
- Statistical testing required: While time spent appears somewhat consistent, an ANOVA test will confirm whether differences are statistically significant.

```python
# Complete the code to calculate the mean time spent on the new page for different language users
df_new.groupby(['language_preferred'])['time_spent_on_the_page'].mean()
```

```
time_spent_on_the_page
```

language_preferred	
English	6.663750
French	6.196471
Spanish	5.835294

dtype: float64

Step 1: Define the null and alternate hypotheses

H_0 : (Null Hypothesis) The mean time spent on the new landing page is the same across all language groups.

H_a : (Alternative Hypothesis) At least one language group's mean time spent is significantly different from the others.

Let μ_1, μ_2, and μ_3 represent the mean time spent on the new landing page for users who prefer English, French, and Spanish, respectively.

> $H_0 : \mu_1 = \mu_2 = \mu_3$

> H_a : At least one of these means is not the same.

Step 2: Select Appropriate test

Selction: One-Way ANOVA Test

Review Assumptions:

1. **Normally Distributed Populations:** The time spent on the new page for each language group should follow a normal distribution. Given that we have a sample size greater than 30 per group, the Central Limit Theorem suggests the sampling distribution of the mean will be approximately normal.
2. **Independent Samples:** The users were randomly assigned to landing pages, and each user's time spent is an independent observation, ensuring that the samples are independent.
3. **Equal Population Variances (Homogeneity of Variance):** ANOVA assumes that the variance in time spent across language groups is approximately equal. We will test this assumption using Levene's test before proceeding with ANOVA.

Levene's Test: Homegeneity of Variance

```python
from scipy.stats import levene

# Extract time spent for each language group
time_english = df_new[df_new['language_preferred'] == 'English']['time_spent_on_the_page']
time_french = df_new[df_new['language_preferred'] == 'French']['time_spent_on_the_page']
time_spanish = df_new[df_new['language_preferred'] == 'Spanish']['time_spent_on_the_page']

# Perform Levene's test
stat, p_value = levene(time_english, time_french, time_spanish)

# Print results
print(f"Levene's test statistic: {stat:.4f}")
print(f"P-value: {p_value:.4f}")

# Decision based on significance level
if p_value < 0.05:
    print("Reject the null hypothesis: The variances are significantly different (unequal variances).")
else:
    print("Fail to reject the null hypothesis: The variances are not significantly different (equal variances).")

if p_value >= 0.05:
    print("Levene’s test confirms that the assumption of equal variances holds. We can proceed with ANOVA.")
```

```
Levene's test statistic: 0.7736
P-value: 0.4671
Fail to reject the null hypothesis: The variances are not significantly different (equal variances).
Levene’s test confirms that the assumption of equal variances holds. We can proceed with ANOVA.
```

Step 3: Decide the signficance level

As given in the problem statement, we select α=0.05 .

Step 4: Collect and prepare the data

```python
# Create a subsetted data frame of the time spent on the new page by English language users
time_spent_English = df_new[df_new['language_preferred'] == "English"]['time_spent_on_the_page']

# Create subsetted data frames of the time spent on the new page by French and Spanish language users
time_spent_French = df_new[df_new['language_preferred'] == "French"]['time_spent_on_the_page']
time_spent_Spanish = df_new[df_new['language_preferred'] == "Spanish"]['time_spent_on_the_page']
```

Step 5: Calculate the p-value

```
# Import the required function
from scipy.stats import f_oneway

# Calculate the p-value using ANOVA
test_stat, p_value = f_oneway(time_spent_English, time_spent_French, time_spent_Spanish)

# Print the p-value
print('The p-value is', p_value)
```

```
The p-value is 0.43204138694325955
```

Step 6: Compare p-value with α

```
# Set the significance level
alpha = 0.05

# Print the conclusion based on the p-value
if p_value < alpha:
    print(f'As the p-value {p_value} is less than the level of significance ({alpha}), we reject the null hypothesis.')
    print('Conclusion: The time spent on the new page is significantly different across at least one language group.')
else:
    print(f'As the p-value {p_value} is greater than the level of significance ({alpha}), we fail to reject the null hypothesis.')
    print('Conclusion: We do not have enough statistical evidence to conclude that time spent on the new page differs by language preference.')
```

```
As the p-value 0.43204138694325955 is greater than the level of significance (0.05), we fail to reject the null hypothesis.
Conclusion: We do not have enough statistical evidence to conclude that time spent on the new page differs by language preference.
```

Step 7: Draw Inference

At a 5% significance level, we fail to reject the null hypothesis. Hence, we do not have enough statistical evidence to conclude that the mean time spent on the new landing page is different across language groups.

Conclusion and Business Recommendations

Summary

E-News Express conducted an A/B test to evaluate whether a new landing page design would drive higher engagement and conversion rates. A statistical analysis was performed on 100 users randomly assigned to either the old or new landing page. The goal was to assess whether the new page increases time spent, improves conversion rates, and if conversion behavior varies by language preference.

Key Findings

1. Users spend significantly more time on the new landing page than on the old page.
2. The new page converts users at a significantly higher rate than the old page.
3. Conversion status and preferred language are independent—language preference does not influence conversion rates.
4. Time spent on the new page does not significantly differ by language group.

Business Recommendations

1. Roll out the new landing page: It improves both engagement and conversions.
2. Optimize Call to Action (CTA) placement and timing: Identify when users are most likely to subscribe and position CTAs accordingly.
3. Refine engagement tracking: Look beyond time spent and analyze interactions (clicks, scroll depth, video views) to measure true engagement.
4. Personalize based on user behavior: Explore whether new vs. returning visitors engage differently and adjust content accordingly.
5. Consider minor localization tweaks: While language doesn't impact conversion, small A/B tests on translated CTAs could enhance user experience.

Final Take-Away

The new landing page successfully increases both engagement and conversions, proving it is more effective than the old version. However, further refinements—such as optimizing CTA timing, segmenting users, and enhancing engagement tracking—could maximize long-term impact.

This data-driven approach ensures that E-News Express continues to refine its digital strategy for maximum subscriber growth.

```
# Upload ipynb file
from google.colab import files
import subprocess

uploaded = files.upload()  # User selects a file
file_name = list(uploaded.keys())[0]  # Get uploaded filename

# Install nbconvert if not installed
subprocess.run(["pip", "install", "nbconvert"])

# Convert notebook to HTML
subprocess.run(["jupyter", "nbconvert", "--to", "html", "--template", "lab", file_name])

# Download the HTML file
html_filename = file_name.replace(".ipynb", ".html")
files.download(html_filename)
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.