

✓ Project Business Statistics: E-news Express

Marks: 60

Define Problem Statement and Objectives

✓ Problem

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.

Objective

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

```
1 !jupyter nbconvert --to html Project.ipynb
```



```
[NbConvertApp] WARNING | pattern 'Project.ipynb' matched no files
This application is used to convert notebook files (*.ipynb)
to various other formats.
```

```
WARNING: THE COMMANDLINE INTERFACE MAY CHANGE IN FUTURE RELEASES.
```

```
Options
```

```
=====
```

```
The options below are convenience aliases to configurable class-options,
as listed in the "Equivalent to" description-line of the aliases.
```

```
To see all configurable class-options for some <cmd>, use:
```

```
<cmd> --help-all
```

```
--debug
```

```
set log level to logging.DEBUG (maximize logging output)
```

```
Equivalent to: [--Application.log_level=10]
```

```
--show-config
```

```
Show the application's configuration (human-readable format)
```

```
Equivalent to: [--Application.show_config=True]
```

```

--show-config-json
  Show the application's configuration (json format)
  Equivalent to: [--Application.show_config_json=True]
--generate-config
  generate default config file
  Equivalent to: [--JupyterApp.generate_config=True]
-y
  Answer yes to any questions instead of prompting.
  Equivalent to: [--JupyterApp.answer_yes=True]
--execute
  Execute the notebook prior to export.
  Equivalent to: [--ExecutePreprocessor.enabled=True]
--allow-errors
  Continue notebook execution even if one of the cells throws an error and include
  Equivalent to: [--ExecutePreprocessor.allow_errors=True]
--stdin
  read a single notebook file from stdin. Write the resulting notebook with default
  Equivalent to: [--NbConvertApp.from_stdin=True]
--stdout
  Write notebook output to stdout instead of files.
  Equivalent to: [--NbConvertApp.writer_class=StdoutWriter]
--inplace
  Run nbconvert in place, overwriting the existing notebook (only
  relevant when converting to notebook format)
  Equivalent to: [--NbConvertApp.use_output_suffix=False --NbConvertApp.export_for
--clear-output
  Clear output of current file and save in place,
  overwriting the existing notebook.
  Equivalent to: [--NbConvertApp.use_output_suffix=False --NbConvertApp.export_for
--no-prompt
  Exclude input and output prompts from converted document.
  Equivalent to: [--TemplateExporter.exclude_input_prompt=True --TemplateExporter.c
--no-input
  Exclude input cells and output prompts from converted document.
  This mode is ideal for generating code-free reports.
  Equivalent to: [--TemplateExporter.exclude_output_prompt=True --TemplateExporter
--allow-chromium-download

```

> Import all the necessary libraries

[] ↴ 1 cell hidden

> Reading the Data into a DataFrame


[] ↴ 2 cells hidden



Explore the dataset and extract insights using 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

```
1 df.head()
```




	user_id	group	landing_page	time_spent_on_the_page	converted	language_preferre
0	546592	control	old	3.48	no	Spanis
1	546468	treatment	new	7.13	yes	Englis
2	546462	treatment	new	4.40	no	Spanis
3	546567	control	old	3.02	no	Frenc
4	546459	treatment	new	4.75	yes	Spanis

```
1 df.tail()
```



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

```
1 df.shape
```



```
(100, 6)
```

Observations:

There are 100 rows and 6 columns in this data set.

```
1 df.info()
```

```

↳ <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)
  memory usage: 4.8+ KB

```

Observations:

- User_id is an integer
- time_spent_on_the_page is a float
- Group, landing_page, converted, and language_preferred are object types

```
1 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

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

```

↳ control    50
   treatment  50
  Name: group, dtype: int64

```

```
1 df['landing_page'].value_counts()
```

```

↳ old    50
   new   50
  Name: landing_page, dtype: int64

```

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

```
↵ yes    54  
   no    46  
   Name: converted, dtype: int64
```

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

```
↵ Spanish  34  
   French   34  
   English  32  
   Name: language_preferred, dtype: int64
```

Observations:

- All the variables have a 100 entries each, meaning that there is no missing data.
- The group variable has two unique values, control and treatment, which both have 50 entries
- The landing_page variable has two unique values, old and new, with 50 entries each
- The converted variable has two unique values, yes and no, yes with 54 and no with 46 entries
- The language_preferred variable has three unique values: Spanish, French, English. Spanish with 34, French with 34, and English with 32 entries.

```
1 df.describe()
```

```
↵
```

	user_id	time_spent_on_the_page
count	100.000000	100.000000
mean	546517.000000	5.377800
std	52.295779	2.378166
min	546443.000000	0.190000
25%	546467.750000	3.880000
50%	546492.500000	5.415000
75%	546567.250000	7.022500
max	546592.000000	10.710000

Observations:

- The average time spent on the page is about 5.38 minutes with a standard deviation of 2.38 minutes

- The minimum time spent on the page was 0.19 minutes and the maximum time spent on the page was 10.71 minutes
- 50% of the entries spent about 5.42 minutes on the page

```
1 df.isnull().sum()
```

```
⇒ user_id          0
   group           0
   landing_page    0
   time_spent_on_the_page  0
   converted       0
   language_preferred  0
   dtype: int64
```

Observations:

There are no null values in this dataset.

```
1 df.duplicated().sum()
```

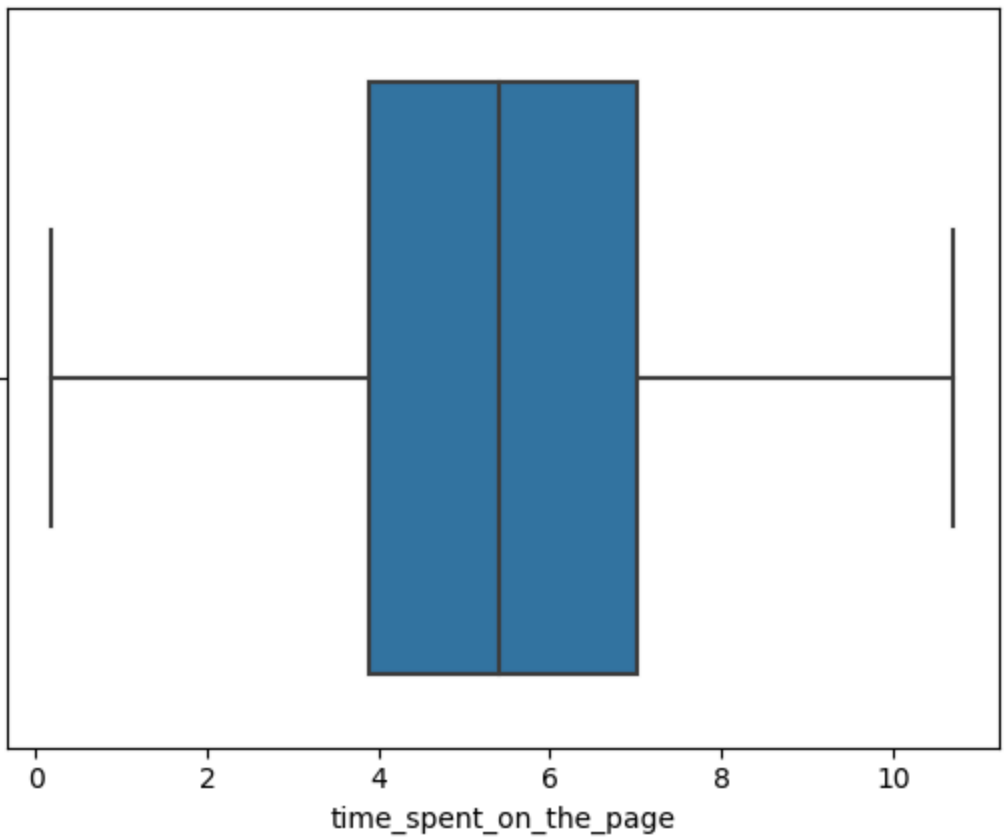
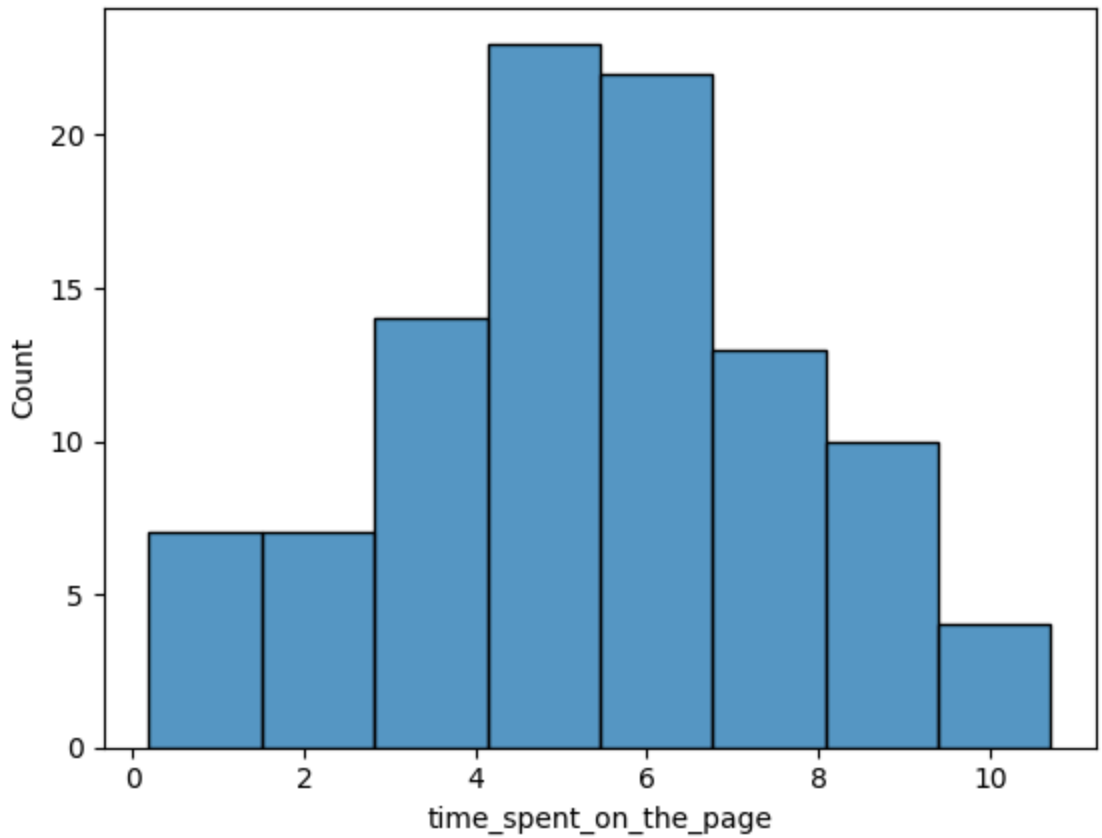
```
⇒ 0
```

Observations:

There are no duplicated values in this dataset.

✓ Univariate Analysis

```
1 sns.histplot(data=df,x='time_spent_on_the_page')
2 plt.show()
3 sns.boxplot(data=df,x='time_spent_on_the_page')
4 plt.show()
```



Observation:

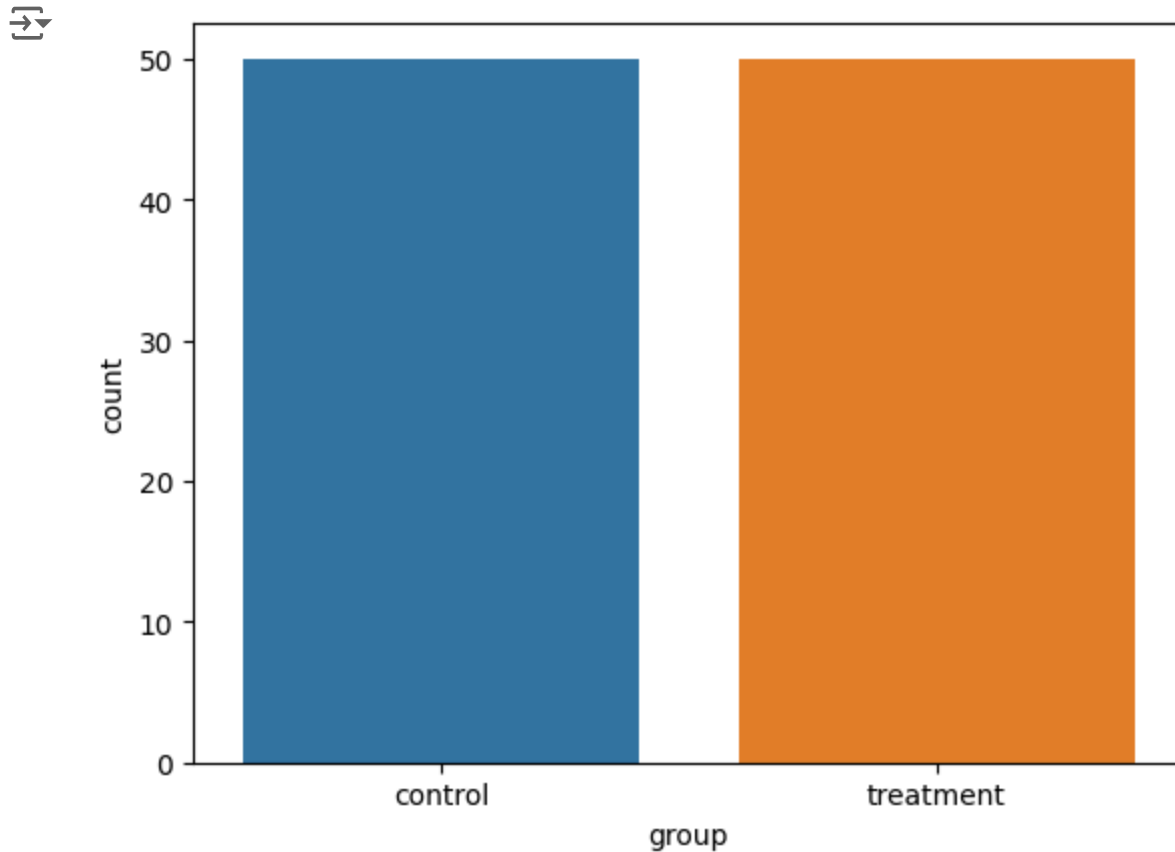
The time spent on the page appears to be normally distributed and has no outliers.

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

```
↳ control    50  
   treatment  50  
   Name: group, dtype: int64
```

```
1 sns.countplot(data=df,x='group')
```

```
2 plt.show()
```



Observation:

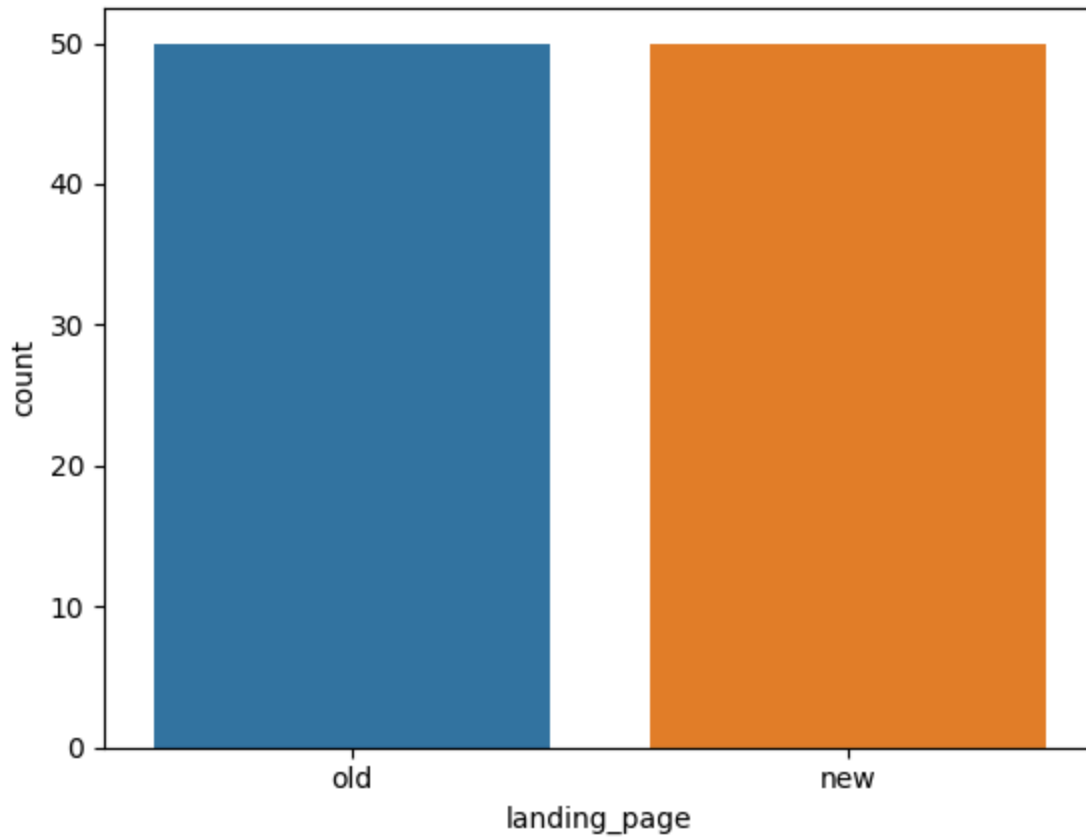
The sample is equally split amongst the control and treatment groups.

```
1 df['landing_page'].value_counts()
```

```
↳ old    50  
   new    50  
   Name: landing_page, dtype: int64
```

```
1 sns.countplot(data=df,x='landing_page')
```

```
2 plt.show()
```


**Observation:**

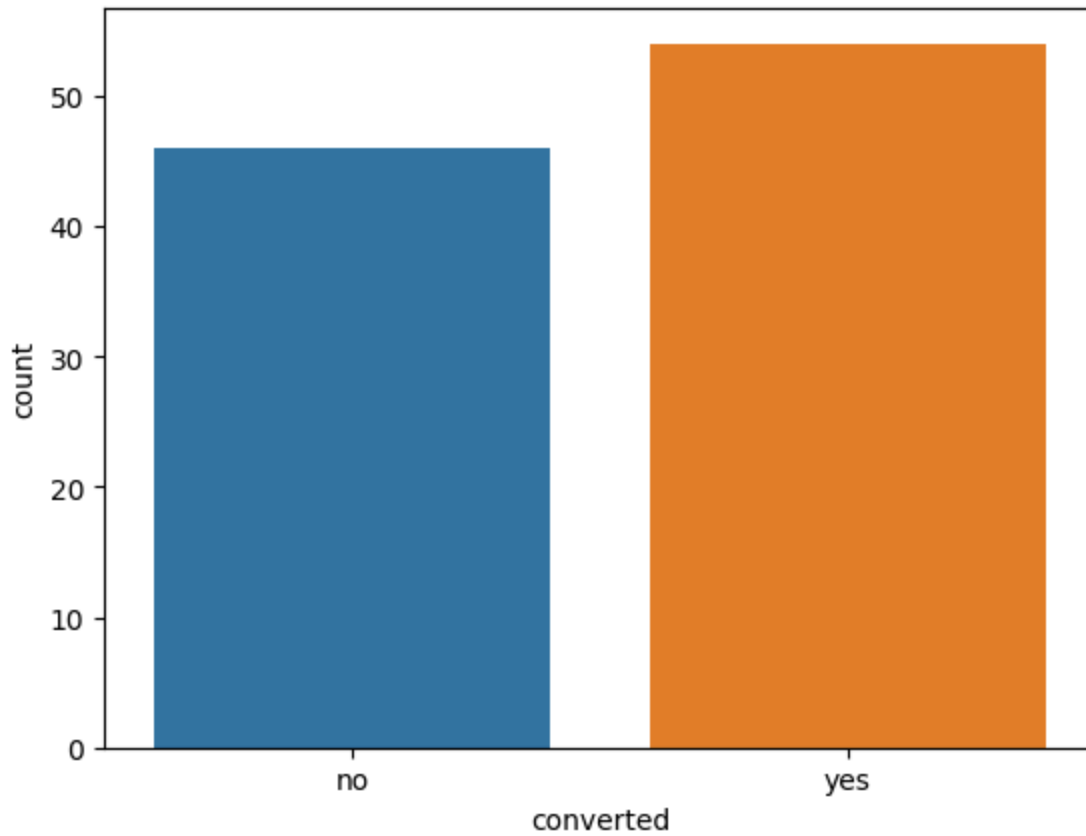
The sample is equally split amongst the old and new landing pages.

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



```
yes    54  
no     46  
Name: converted, dtype: int64
```

```
1 sns.countplot(data=df, x='converted')  
2 plt.show()
```



Observations:

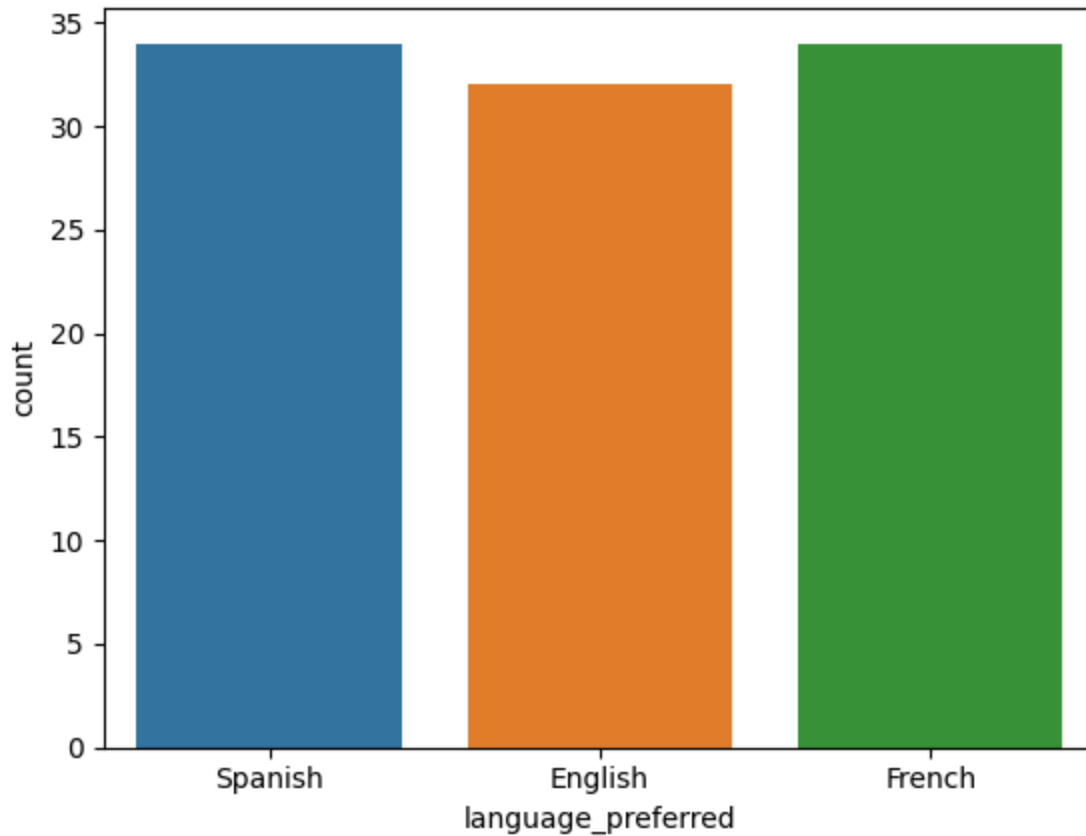
There are more people who converted to the new landing page than people who did not convert.

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



```
Spanish    34  
French     34  
English    32  
Name: language_preferred, dtype: int64
```

```
1 sns.countplot(data=df, x='language_preferred')  
2 plt.show()
```



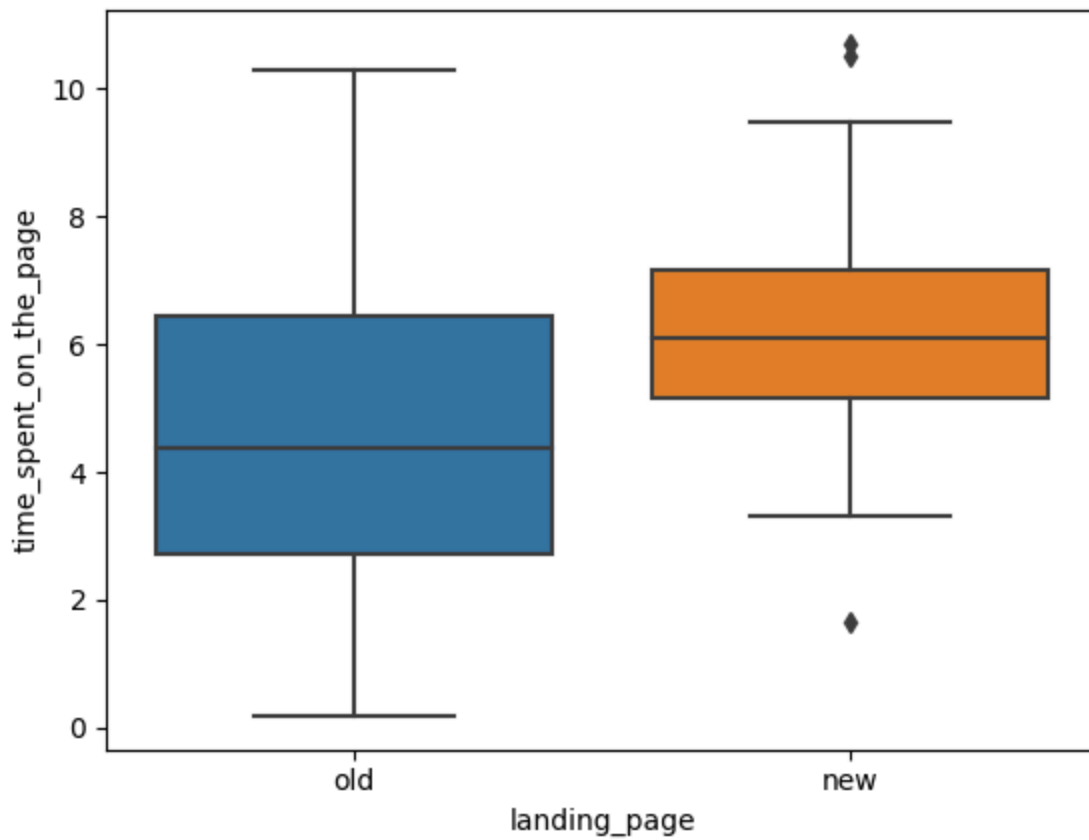
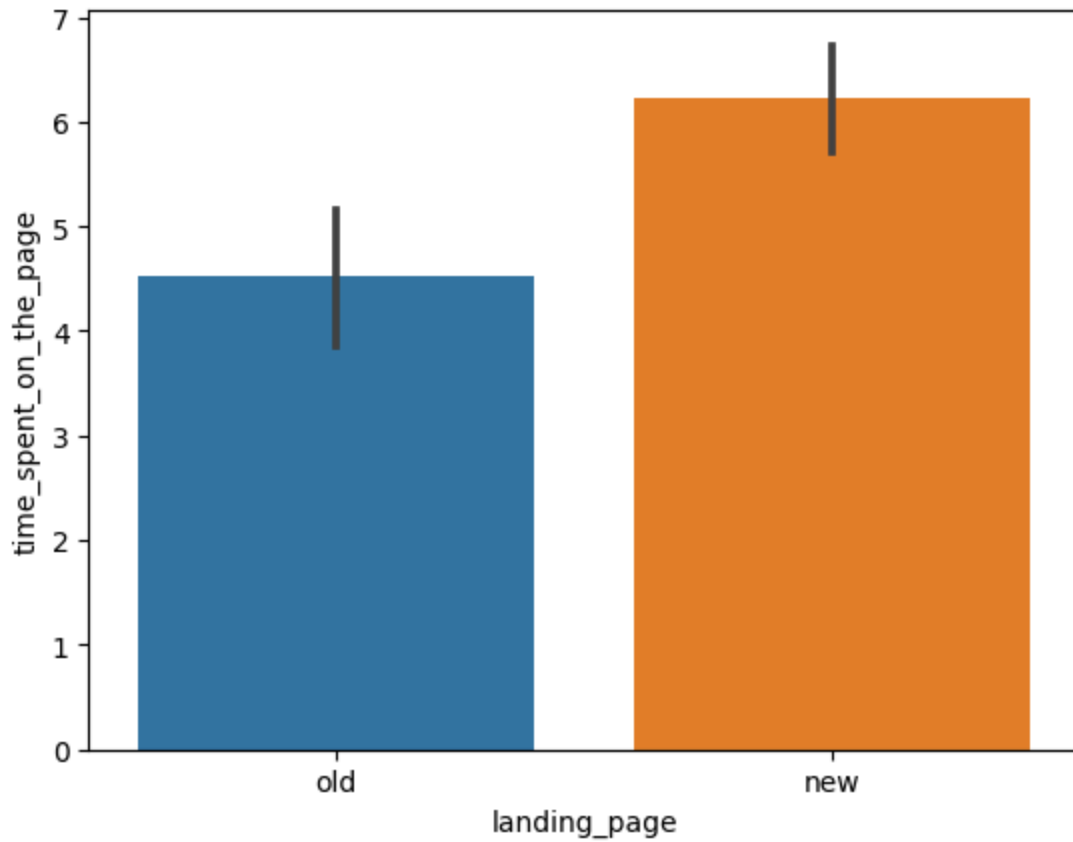
Observations:

- Spanish and French have the most amount of entries with 34 each and English had the least amount of entries with 32.

Bivariate Analysis

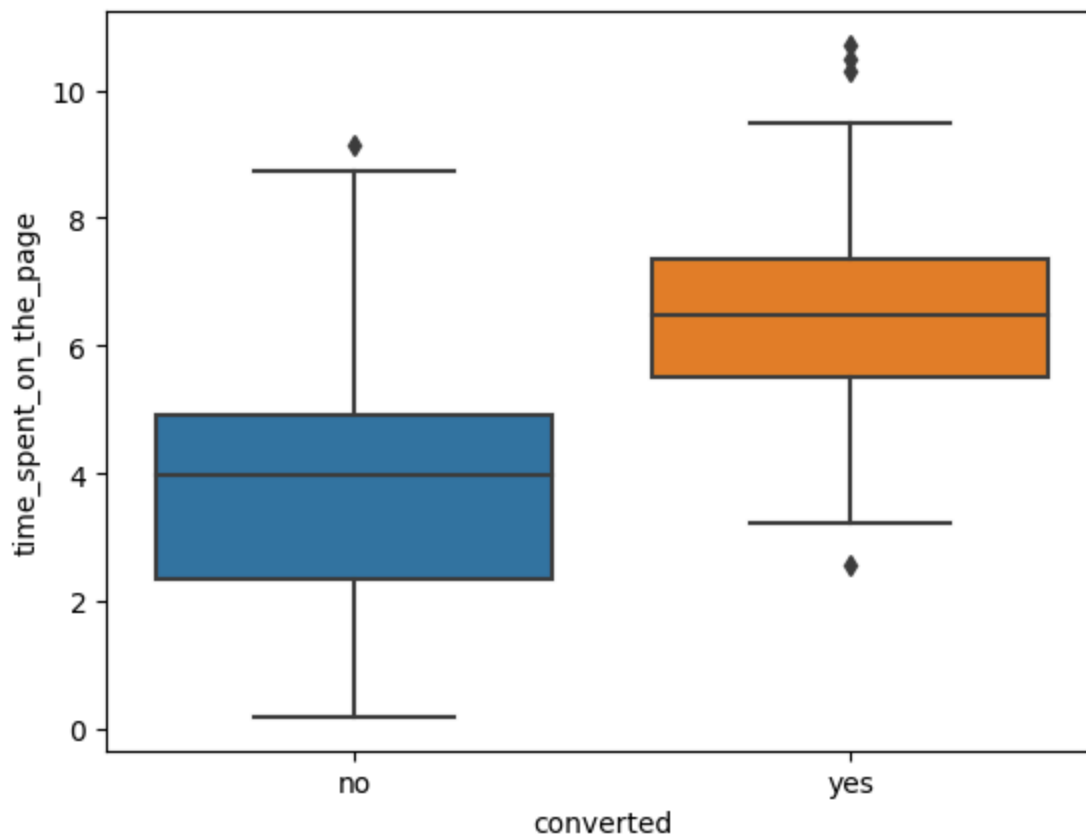
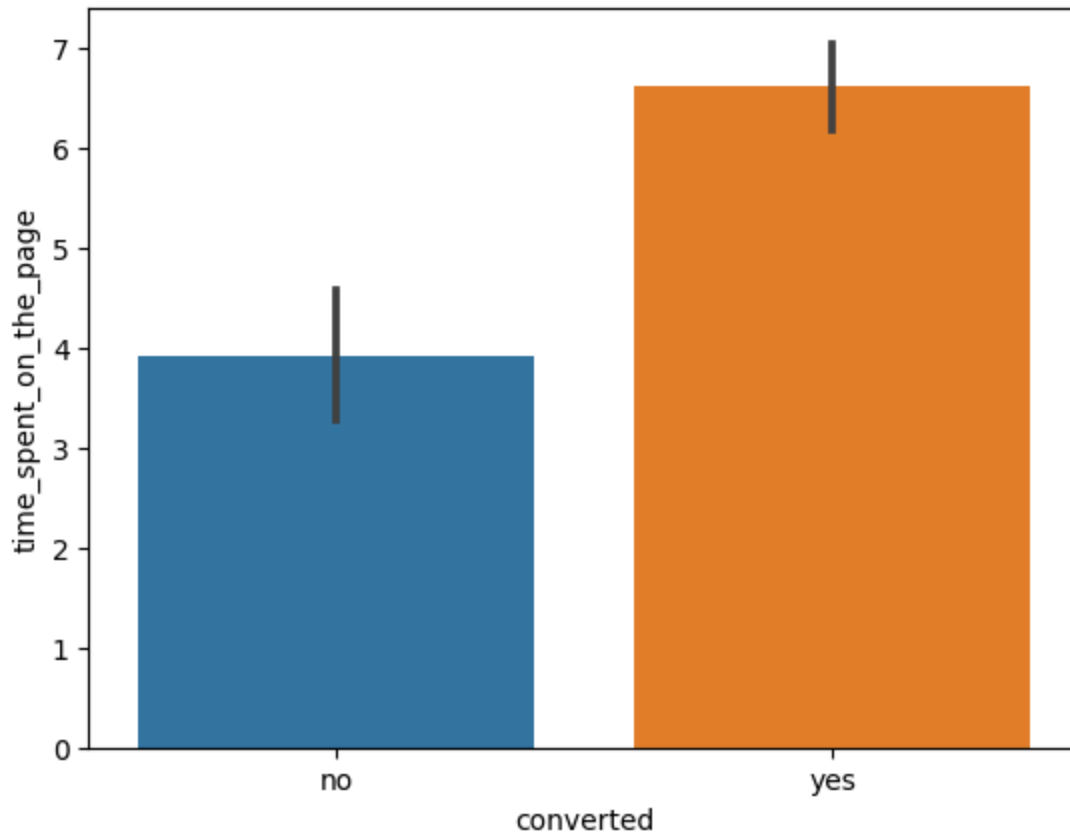
✓ Landing page vs. Time spent on the page

```
1 sns.barplot(data = df, x = 'landing_page', y = 'time_spent_on_the_page')
2 plt.show()
3 sns.boxplot(data=df,x='landing_page',y='time_spent_on_the_page')
4 plt.show()
```



✓ Conversion status vs Time spent on the page

```
1 sns.barplot(data = df, x = 'converted', y = 'time_spent_on_the_page')  
2 plt.show()  
3 sns.boxplot(data = df, x = 'converted', y = 'time_spent_on_the_page')  
4 plt.show()
```

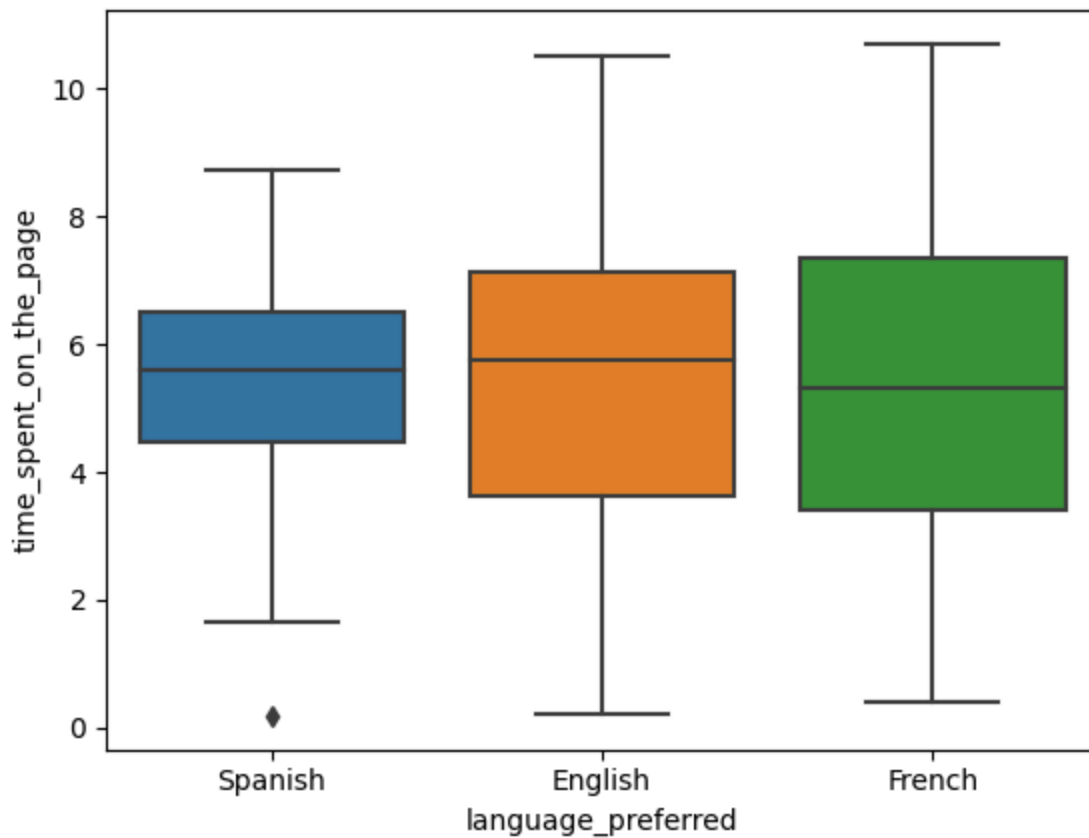
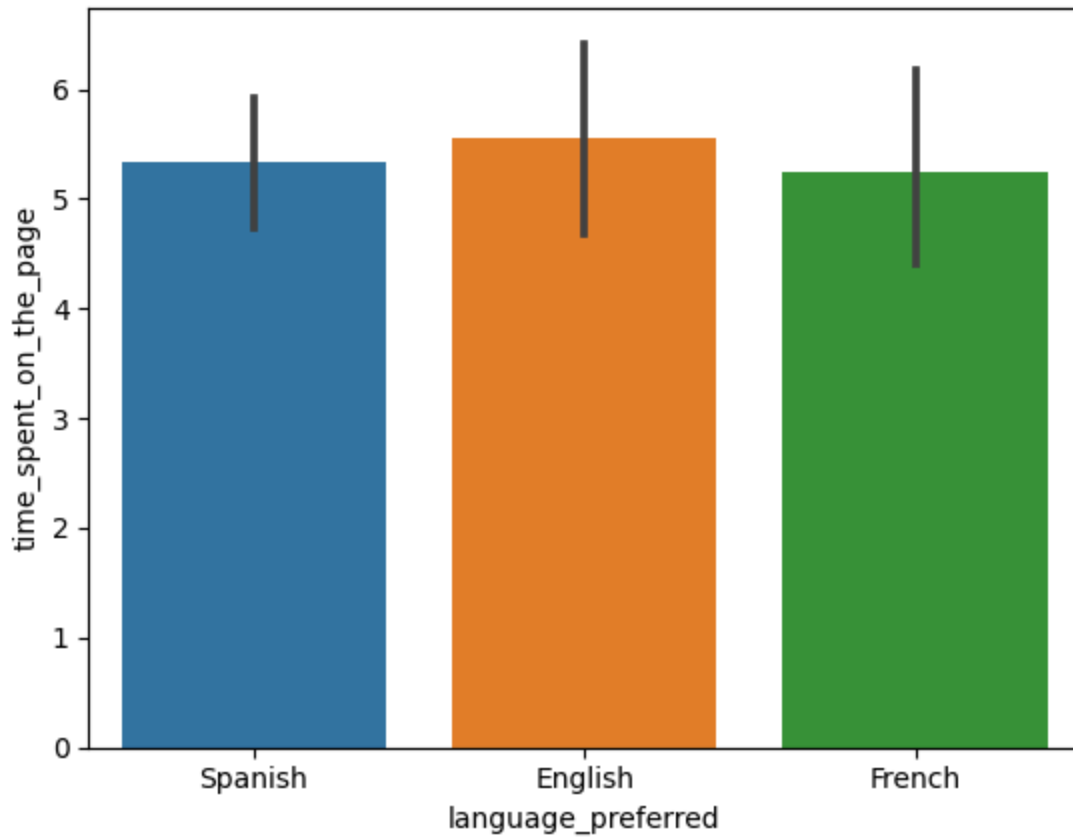


Observations:

The graph shows that those who converted to a subscriber spent more time on the page.

✓ Language preferred vs Time spent on the page

```
1 sns.barplot(data = df, x = 'language_preferred', y = 'time_spent_on_the_page')
2 plt.show()
3 sns.boxplot(data = df, x = 'language_preferred', y = 'time_spent_on_the_page')
4 plt.show()
```

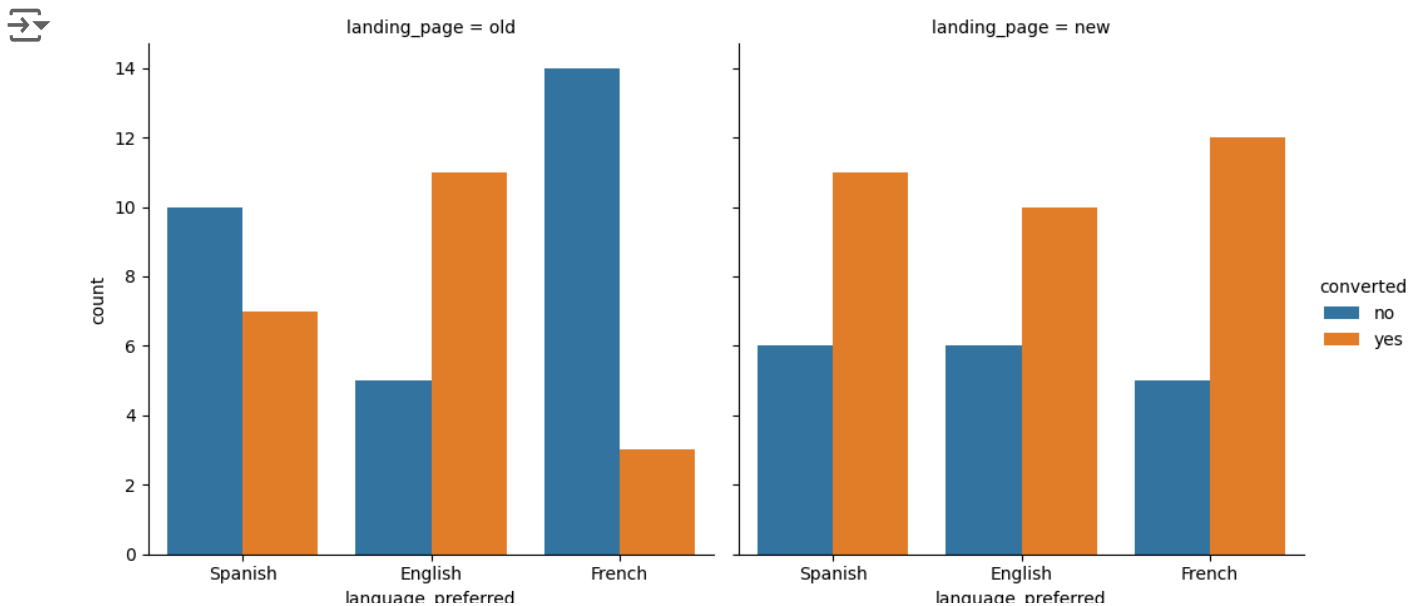


Observations:

- The average time spent on the page shows similar for all the preferred languages. Users who preferred Spanish have the smallest spread in time spent on the page.

Relationship between preferred language, conversion status, and landing page

```
1 sns.catplot(data = df, x = 'language_preferred', hue = 'converted', col = 'landing_page',
```

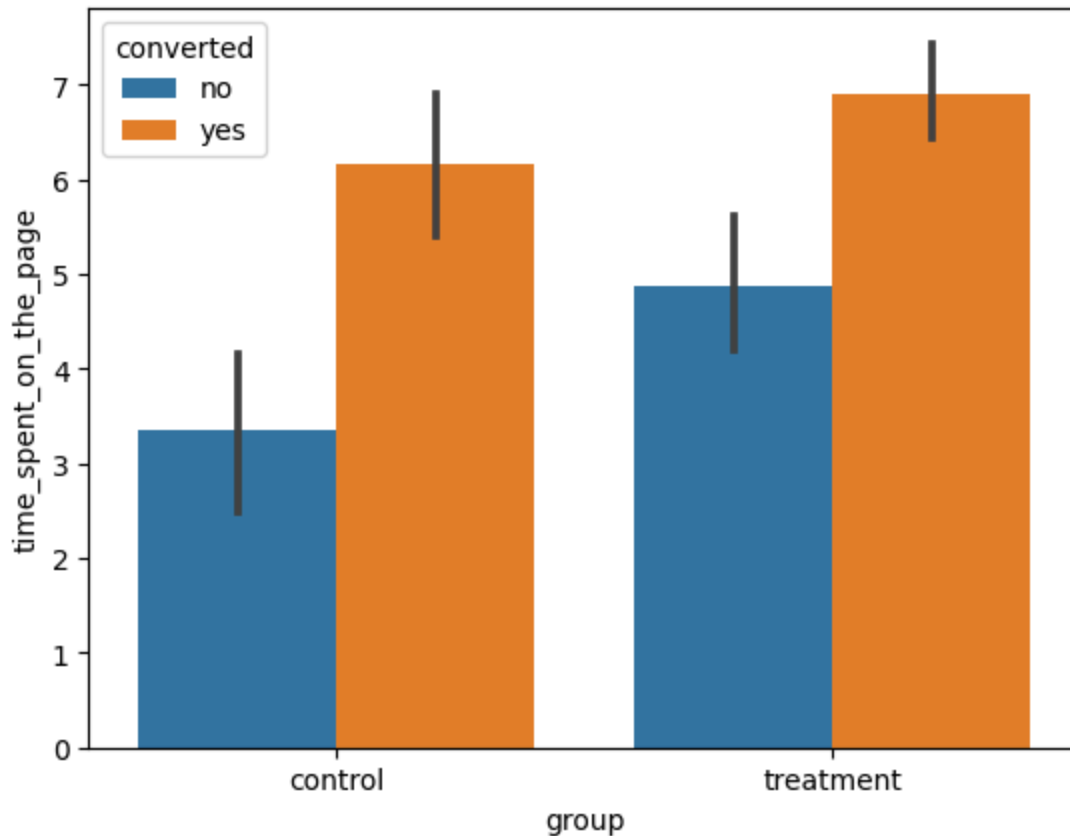


Observations:

Users that preferred Spanish and French chose not to convert to a subscriber when viewing the old landing page. Users of all language preferences converted to subscribers when viewing the new landing page.

Relationship between preferred language, time spent on the page, and conversion status

```
1 sns.barplot(data = df, x = 'group', y = 'time_spent_on_the_page', hue = 'converted');
```

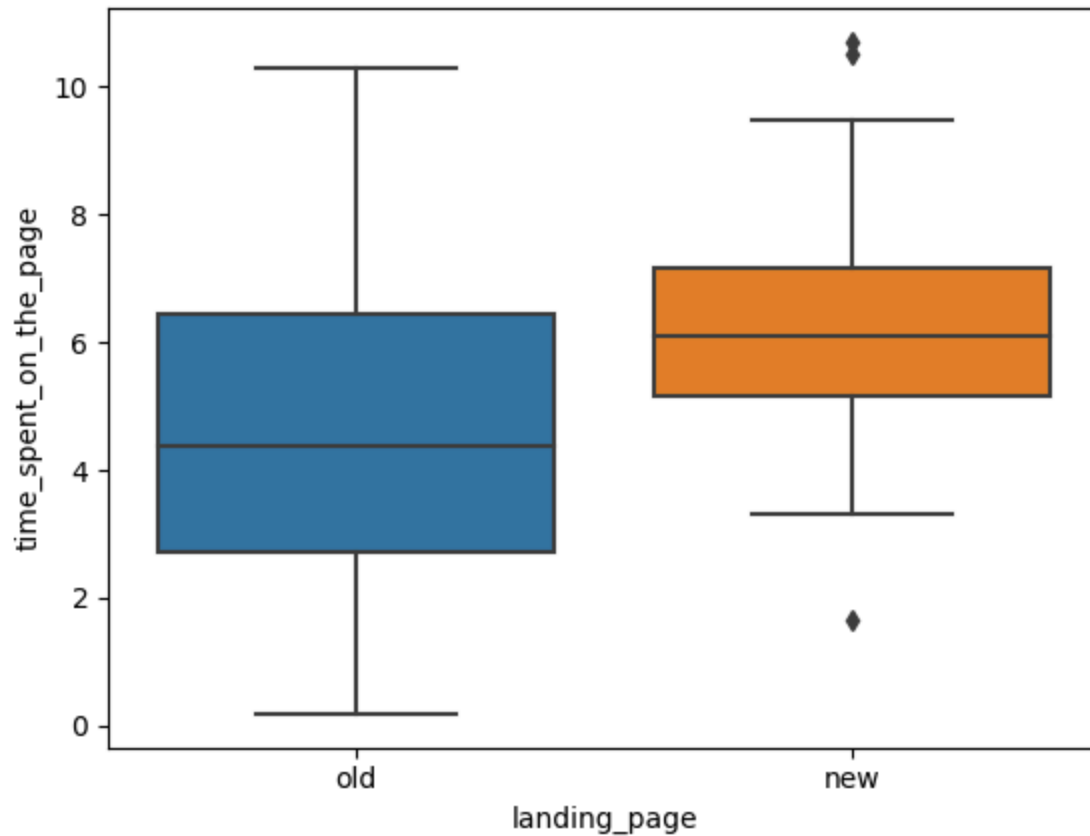



Observations:

More people converted to subscribers in the treatment group compared to the control group. Also users in the treatment group spent more time on the page in comparison to people in the control group.

- ✓ 1. Do the users spend more time on the new landing page than the existing landing page?
- ✓ Perform Visual Analysis

```
1 sns.boxplot(x = 'landing_page', y = 'time_spent_on_the_page', data = df)
2 plt.show()
```



Observations:

The average time spent on the new page seems greater than the old page.

> Step 1: Define the null and alternate hypotheses

↳ 1 cell hidden

> Step 2: Select Appropriate test

↳ 1 cell hidden

✓ Step 3: Decide the significance level

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

> Step 4: Collect and prepare data

[] ↳ 4 cells hidden

> Step 5: Calculate the p-value

[] ↴ 1 cell hidden

> Step 6: Compare the p-value with α

[] ↴ 1 cell hidden

✓ Step 7: Draw inference

Since the p-value is much less than the level of significance of 5%, the null hypothesis is rejected. This means that there is significant evidence that the mean time spent by the users on the new page is greater than the mean time spent by the users on the old page.

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

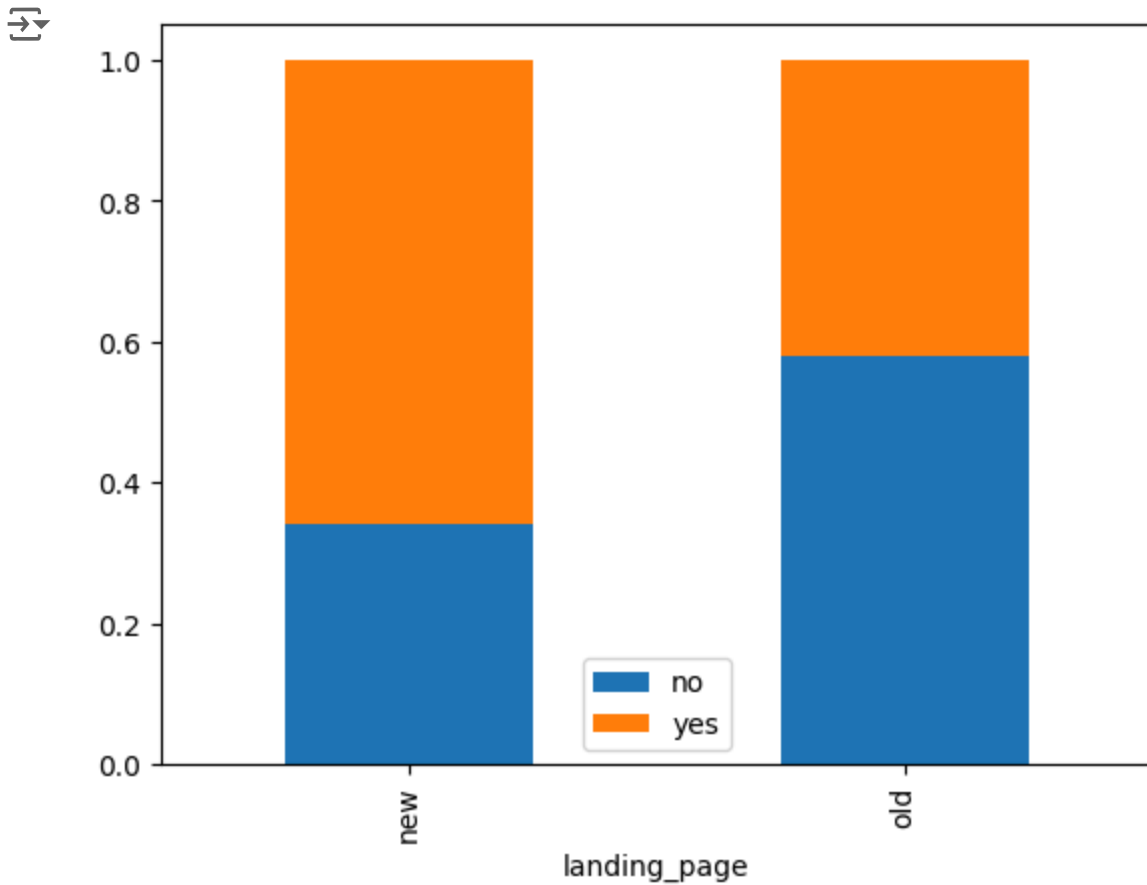
2. Is the conversion rate (the proportion of users who visit the landing page and get converted) for the new page greater than the conversion rate for the old page?

```
1 pd.crosstab(df['landing_page'],df['converted'])
```



	converted	
landing_page	no	yes
new	17	33
old	29	21

```
1 pd.crosstab(df['landing_page'],df['converted'],normalize='index').plot(kind="bar",stacked)
2 plt.legend()
3 plt.show()
```



Observations:

The users on the new landing page are more likely to convert to subscribers than users on the old landing page.

> Step 1: Define null and alternative hypothesis

↳ 1 cell hidden

> Step 2: Select Appropriate Test

↳ 1 cell hidden

> Step 3: Decide the significance level

↳ 1 cell hidden

> Step 4: Collect and prepare data

[] ↳ 2 cells hidden

> Step 5: Calculate the p-value

[] ↴ 1 cell hidden

> Step 6: Compare the p-value α

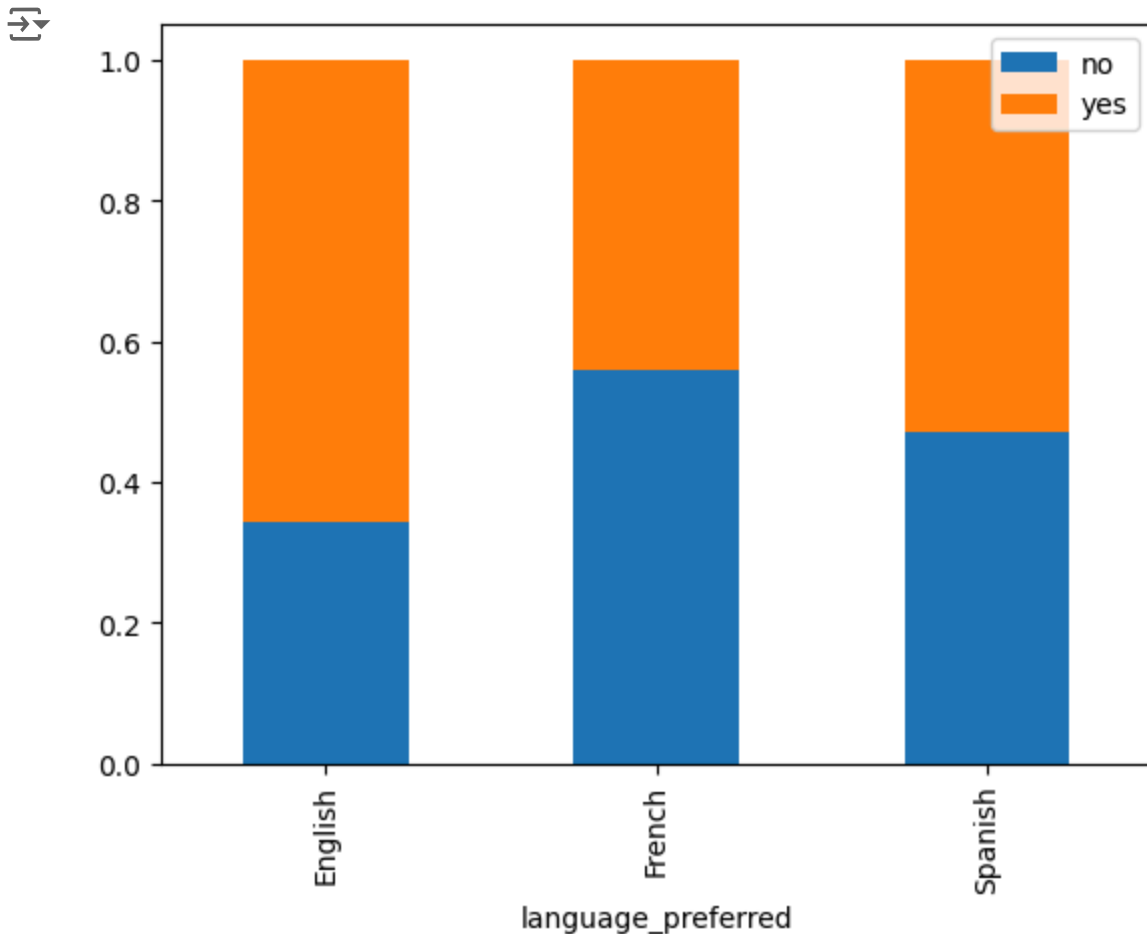
[] ↴ 1 cell hidden

> Step 7: Draw inference

↴ 1 cell hidden

✓ 3. Is the conversion and preferred language are independent or related?

```
1 pd.crosstab(df['language_preferred'],df['converted'],normalize='index').plot(kind="bar",
2 plt.legend()
3 plt.show()
```



➤ Step 1: Define the null and alternate hypotheses

↳ 1 cell hidden

✓ Step 2: Select Appropriate test

This is a problem of the test of independence, concerning two categorical variables - converted status and preferred language. Based on this information, a **chi-square test for independence** would be the best and most effective test.

✓ Step 3: Decide the significance level

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

✓ Step 4: Collect and prepare data

```

1 contingency_table = pd.crosstab(df['language_preferred'], df['converted'])
2
3 contingency_table

```



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

Chi-Squared test for independence assumptions:

- Categorical variables - Yes
- Expected value of the number of sample observations in each level of the variable is at least 5
- Yes, the number of observations in each level is greater than 5.
- Random sampling from the population - Yes, we are informed that the collected sample is a simple random sample.

✓ Step 5: Calculate the p-value

```

1 from scipy.stats import chi2_contingency
2
3 chi2, p_value, dof, exp_freq = chi2_contingency(contingency_table)
4
5 print('The p-value is', p_value)

```



The p-value is 0.2129888748754345

✓ Step 6: Compare the p-value with α

```

1 if p_value < 0.05:
2     print(f'As the p-value {p_value} is less than the level of significance, we reject the null hypothesis')
3 else:
4     print(f'As the p-value {p_value} is greater than the level of significance, we fail to reject the null hypothesis')

```



As the p-value 0.2129888748754345 is greater than the level of significance, we fail to reject the null hypothesis



↳ 1 cell hidden
> Step 7: Draw inference

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

```
1 df_new = df[df['landing_page'] == 'new']
```

```
1 sns.boxplot(x = 'language_preferred', y = 'time_spent_on_the_page', showmeans = True, dat  
2 plt.show()
```

