Activity_Hypothesis testing with Python

November 14, 2023

1 Activity: Hypothesis testing with Python

1.1 Introduction

As you've been learning, analysis of variance (commonly called ANOVA) is a group of statistical techniques that test the difference of means among three or more groups. It's a powerful tool for determining whether population means are different across groups and for answering a wide range of business questions.

In this activity, you are a data professional working with historical marketing promotion data. You will use the data to run a one-way ANOVA and a post hoc ANOVA test. Then, you will communicate your results to stakeholders. These experiences will help you make more confident recommendations in a professional setting.

In your dataset, each row corresponds to an independent marketing promotion, where your business uses TV, social media, radio, and influencer promotions to increase sales. You have previously provided insights about how different promotion types affect sales; now stakeholders want to know if sales are significantly different among various TV and influencer promotion types.

To address this request, a one-way ANOVA test will enable you to determine if there is a statistically significant difference in sales among groups. This includes: * Using plots and descriptive statistics to select a categorical independent variable * Creating and fitting a linear regression model with the selected categorical independent variable * Checking model assumptions * Performing and interpreting a one-way ANOVA test * Comparing pairs of groups using an ANOVA post hoc test * Interpreting model outputs and communicating the results to nontechnical stakeholders

1.2 Step 1: Imports

Import pandas, pyplot from matplotlib, seaborn, api from statsmodels, ols from statsmodels.formula.api, and pairwise_tukeyhsd from statsmodels.stats.multicomp.

[1]:
```
# Import libraries and packages.

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from statsmodels.formula.api import ols
```

```
from statsmodels.stats.multicomp import pairwise_tukeyhsd
```

`Pandas` was used to load the dataset `marketing_sales_data.csv` as `data`, now display the first five rows. The variables in the dataset have been adjusted to suit the objectives of this lab. As shown in this cell, the dataset has been automatically loaded in for you. You do not need to download the .csv file, or provide more code, in order to access the dataset and proceed with this lab. Please continue with this activity by completing the following instructions.

[2]:
```
# RUN THIS CELL TO IMPORT YOUR DATA.

### YOUR CODE HERE ###
data = pd.read_csv('marketing_sales_data.csv')

# Display the first five rows.

data.head(5)
```

[2]:

	TV	Radio	Social Media	Influencer	Sales
0	Low	1.218354	1.270444	Micro	90.054222
1	Medium	14.949791	0.274451	Macro	222.741668
2	Low	10.377258	0.061984	Mega	102.774790
3	High	26.469274	7.070945	Micro	328.239378
4	High	36.876302	7.618605	Mega	351.807328

The features in the data are: * TV promotion budget (in Low, Medium, and High categories) * Social media promotion budget (in millions of dollars) * Radio promotion budget (in millions of dollars) * Sales (in millions of dollars) * Influencer size (in Mega, Macro, Nano, and Micro categories)

Question: Why is it useful to perform exploratory data analysis before constructing a linear regression model?

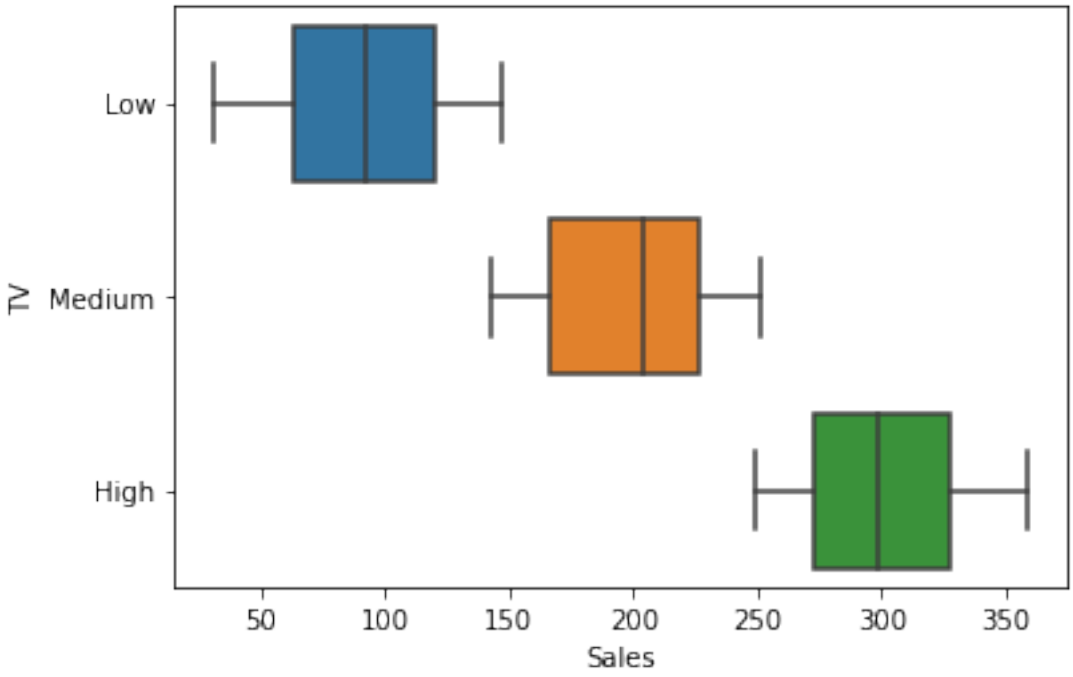
EDA is important so we can uncover information within the dataset to make accurate models.

1.3 Step 2: Data exploration

First, use a boxplot to determine how `Sales` vary based on the `TV` promotion budget category.

[3]:
```
# Create a boxplot with TV and Sales.
sns.boxplot(x = "Sales", y = "TV", data = data)
```

[3]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb873238d10>
```



Hint 1

There is a function in the `seaborn` library that creates a boxplot showing the distribution of a variable across multiple groups.

Hint 2

Use the `boxplot()` function from `seaborn`.

Hint 3

Use `TV` as the `x` argument, `Sales` as the `y` argument, and `data` as the `data` argument.

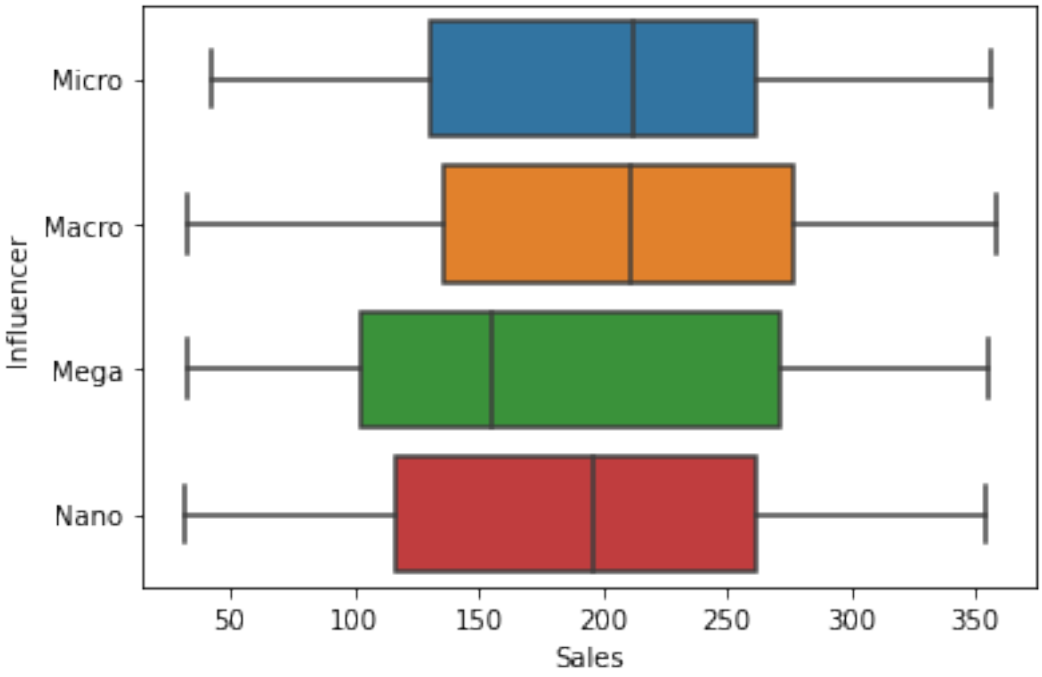
Question: Is there variation in `Sales` based off the `TV` promotion budget?

Yes, the variation between the budgets low and high is 200 million dollars.

Now, use a boxplot to determine how `Sales` vary based on the `Influencer` size category.

[4]:
```
# Create a boxplot with Influencer and Sales.

sns.boxplot(x = "Sales", y = "Influencer", data = data)
```

[4]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb86f72d250>
```



Question: Is there variation in `Sales` based off the `Influencer` size?

Yes, the data shows that Mega influencers have a lower quartile range of about 150 million where as other sized influencers are 200 million and up.

1.3.1 Remove missing data

You may recall from prior labs that this dataset contains rows with missing values. To correct this, drop these rows. Then, confirm the data contains no missing values.

[18]:
```
# Drop rows that contain missing data and update the DataFrame.

data = data.dropna(axis = 0)

# Confirm the data contains no missing values.

data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 569 entries, 0 to 571
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   TV            569 non-null    object
 1   Radio         569 non-null    float64
```

```
 2   Social Media  569 non-null    float64
 3   Influencer    569 non-null    object
 4   Sales         569 non-null    float64
dtypes: float64(3), object(2)
memory usage: 26.7+ KB
```

Hint 1

There is a `pandas` function that removes missing values.

Hint 2

The `dropna()` function removes missing values from an object (e.g., DataFrame).

Hint 3

Verify the data is updated properly after the rows containing missing data are dropped.

1.4 Step 3: Model building

Fit a linear regression model that predicts `Sales` using one of the independent categorical variables in `data`. Refer to your previous code for defining and fitting a linear regression model.

[25]:
```
# Define the OLS formula.

ols_formula = "Sales ~ TV"

# Create an OLS model.

OLS = ols(formula = ols_formula, data = data)

# Fit the model.

model = OLS.fit()

# Save the results summary.

summary = model.summary()

# Display the model results.

summary
```

[25]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Sales   R-squared:                       0.874
Model:                            OLS   Adj. R-squared:                  0.874
Method:                 Least Squares   F-statistic:                     1971.
```

```
Date:                Tue, 14 Nov 2023   Prob (F-statistic):          8.81e-256
Time:                        20:32:28   Log-Likelihood:                -2778.9
No. Observations:                 569   AIC:                             5564.
Df Residuals:                     566   BIC:                             5577.
Df Model:                           2
Covariance Type:            nonrobust
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
Intercept      300.5296      2.417    124.360      0.000     295.783     305.276
TV[T.Low]     -208.8133      3.329    -62.720      0.000    -215.353    -202.274
TV[T.Medium]  -101.5061      3.325    -30.526      0.000    -108.038     -94.975
==============================================================================
Omnibus:                      450.714   Durbin-Watson:                   2.002
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               35.763
Skew:                          -0.044   Prob(JB):                     1.71e-08
Kurtosis:                       1.775   Cond. No.                         3.86
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

Hint 1

Refer to code you've written to fit linear regression models.

Hint 2

Use the `ols()` function from `statsmodels.formula.api`, which creates a model from a formula and DataFrame, to create an OLS model.

Hint 3

Use `C()` around the variable name in the ols formula to indicate a variable is categorical.

Be sure the variable string names exactly match the column names in `data`.

Question: Which categorical variable did you choose for the model? Why?

I chose TV because the variation between "Low" to "High" is much greater than the other categories within the dataset.

1.4.1 Check model assumptions

Now, check the four linear regression assumptions are upheld for your model.

[ ]:

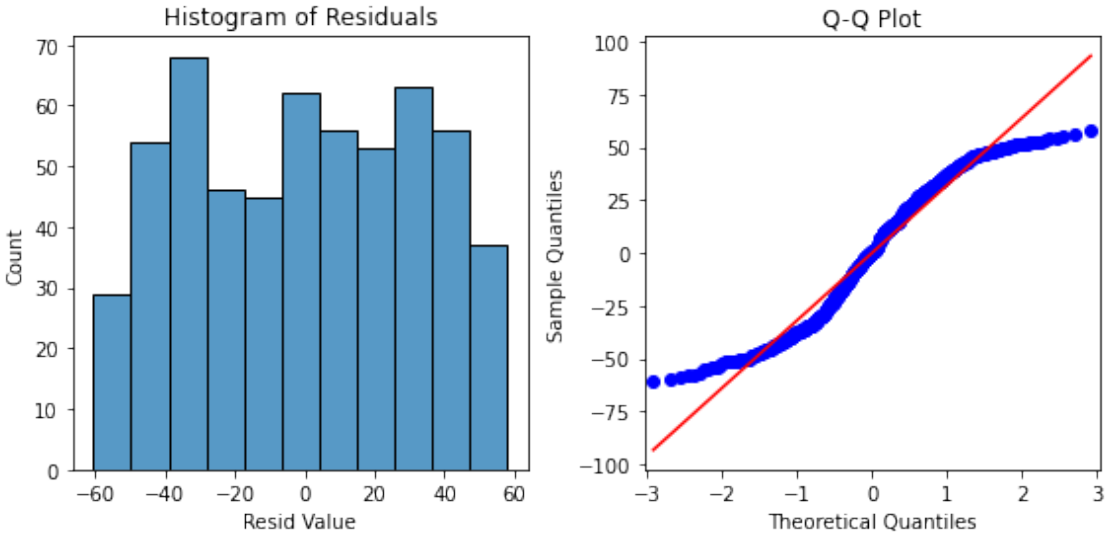
Question: Is the linearity assumption met?

There is no independent variables so linearity is not required

The independent observation assumption states that each observation in the dataset is independent. As each marketing promotion (row) is independent from one another, the independence assumption is not violated.

Next, verify that the normality assumption is upheld for the model.

[21]:
```
# Calculate the residuals.

residuals = model.resid

# Create a histogram with the residuals.

fig, axes = plt.subplots(1,2, figsize = (8,4))

sns.histplot(residuals, ax=axes[0])
axes[0].set_xlabel("Resid Value")
axes[0].set_title("Histogram of Residuals")

# Create a QQ plot of the residuals.

sm.qqplot(residuals, line = 's', ax = axes[1])
axes[1].set_title("Q-Q Plot")

plt.tight_layout()
```



Hint 1

Access the residuals from the fit model object.

Hint 2

Use `model.resid` to get the residuals from a fit model called `model`.

Hint 3

For the histogram, pass the residuals as the first argument in the `seaborn histplot()` function.

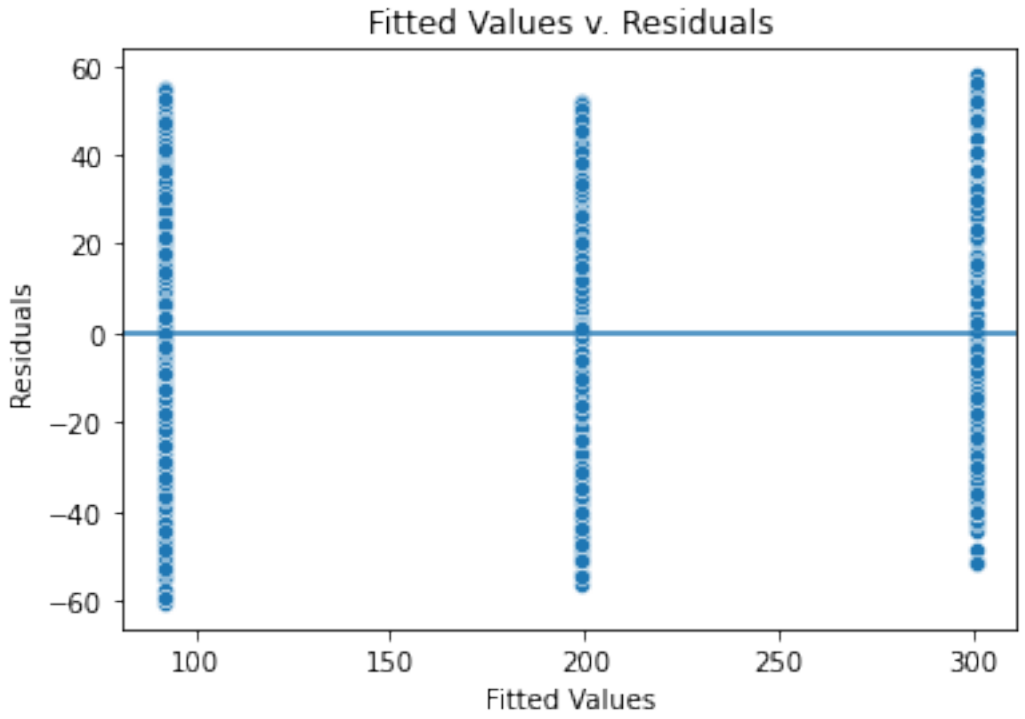
For the QQ-plot, pass the residuals as the first argument in the `statsmodels qqplot()` function.

Question: Is the normality assumption met?

No it is not met, neither the histogram nor the Q-Q plot follow normality

Now, verify the constant variance (homoscedasticity) assumption is met for this model.

[26]:
```
# Create a scatter plot with the fitted values from the model and the residuals.

fig = sns.scatterplot(x = model.fittedvalues, y = model.resid)
fig.set_xlabel("Fitted Values")
fig.set_ylabel("Residuals")
fig.set_title("Fitted Values v. Residuals")

# Add a line at y = 0 to visualize the variance of residuals above and below 0.

fig.axhline(0)

plt.show()
```



Hint 1

Access the fitted values from the model object fit earlier.

Hint 2

Use `model.fittedvalues` to get the fitted values from the fit model called `model`.

Hint 3

Call the `scatterplot()` function from the `seaborn` library and pass in the fitted values and residuals.

Add a line to a figure using the `axline()` function.

Question: Is the constant variance (homoscedasticity) assumption met?

The fitted values are nearly evenly distributed so this validates the constant variance.

1.5 Step 4: Results and evaluation

First, display the OLS regression results.

[27]:
```
# Display the model results summary.

summary
```

[27]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Sales   R-squared:                       0.874
Model:                            OLS   Adj. R-squared:                  0.874
Method:                 Least Squares   F-statistic:                     1971.
Date:                Tue, 14 Nov 2023   Prob (F-statistic):          8.81e-256
Time:                        20:32:28   Log-Likelihood:                -2778.9
No. Observations:                 569   AIC:                             5564.
Df Residuals:                     566   BIC:                             5577.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      300.5296      2.417    124.360      0.000     295.783     305.276
TV[T.Low]     -208.8133      3.329    -62.720      0.000    -215.353    -202.274
TV[T.Medium]  -101.5061      3.325    -30.526      0.000    -108.038     -94.975
==============================================================================
Omnibus:                      450.714   Durbin-Watson:                   2.002
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               35.763
```

```
Skew:                          -0.044   Prob(JB):                     1.71e-08
Kurtosis:                       1.775   Cond. No.                         3.86
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

Question: What is your interpretation of the model's R-squared?

The R-squared is 0.847 which is an 87.4% of variance in sales. Which would make the model effective.

Question: What is your intepretation of the coefficient estimates? Are the coefficients statistically significant?

The p-value for all coeffecients is 0.000, so all are statiscally significant. The 95% confidence interval is great for my stakeholders when using the model for predictions.

Question: Do you think your model could be improved? Why or why not? How?

With 95% as our confidence level it would be difficult to make it anymore improved than it already is. But there are always smaller variables that could be added in depending on the questions stakeholders ask.

1.5.1 Perform a one-way ANOVA test

With the model fit, run a one-way ANOVA test to determine whether there is a statistically significant difference in `Sales` among groups.

[28]:
```
# Create an one-way ANOVA table for the fit model.

sm.stats.anova_lm(model, typ=2)
```

[28]:

	sum_sq	df	F	PR(>F)
TV	4.052692e+06	2.0	1971.455737	8.805550e-256
Residual	5.817589e+05	566.0	NaN	NaN

Hint 1

Review what you've learned about how to perform a one-way ANOVA test.

Hint 2

There is a function in `statsmodels.api` (i.e. `sm`) that peforms an ANOVA test for a fit linear model.

Hint 3

Use the `anova_lm()` function from `sm.stats`. Specify the type of ANOVA test (for example, one-way or two-way), using the `typ` parameter.

Question: What are the null and alternative hypotheses for the ANOVA test?

The null hypothesis is: no difference in sales based on tv promo budget The alternative: There is a difference in sales based on tv promo budget

Question: What is your conclusion from the one-way ANOVA test?

The test shows there is significant difference in sales within tv groups. The p-value is less than 0.05.

Question: What did the ANOVA test tell you?

The test is telling me I can reject the null hypothesis and that there is a statiscally significant difference in sales

1.5.2 Perform an ANOVA post hoc test

If you have significant results from the one-way ANOVA test, you can apply ANOVA post hoc tests such as the Tukey's HSD post hoc test.

Run the Tukey's HSD post hoc test to compare if there is a significant difference between each pair of categories for TV.

[29]:
```
# Perform the Tukey's HSD post hoc test.

tukey_oneway = pairwise_tukeyhsd(endog = data["Sales"], groups = data["TV"])

tukey_oneway.summary()
```

[29]:
```
<class 'statsmodels.iolib.table.SimpleTable'>
```

Hint 1

Review what you've learned about how to perform a Tukey's HSD post hoc test.

Hint 2

Use the `pairwise_tukeyhsd()` function from `statsmodels.stats.multicomp`.

Hint 3

The `endog` argument in `pairwise_tukeyhsd` indicates which variable is being compared across groups (i.e., `Sales`). The `groups` argument in `pairwise_tukeyhsd` tells the function which variable holds the group you're interested in reviewing.

Question: What is your interpretation of the Tukey HSD test?

This test shows that I can reject the null hypothesis

Question: What did the post hoc tell you?**

All of the groups are true to reject, the summary provides more information as to why I can reject the null.

1.6 Considerations

What are some key takeaways that you learned during this lab?

ANOVA tests give a lot more information regarding why either reject or fail to reject a hypothesis

What summary would you provide to stakeholders? Consider the statistical significance of key relationships and differences in distribution.

With linear regression performed and giving a confidence level of 95% we can use this model to make predictions about future sale regarding TV promo budgets. Following up with those test I performed ANOVA tests as well giving further information as to why we should go this route.

Reference Saragih, H.S. *Dummy Marketing and Sales Data*

Congratulations! You've completed this lab. However, you may not notice a green check mark next to this item on Coursera's platform. Please continue your progress regardless of the check mark. Just click on the "save" icon at the top of this notebook to ensure your work has been logged.