Statistical Test:

Statistical tests are used in hypothesis testing. In general, they can be used to:
determine whether an input variable has a statistically significant relationship with an output (target) variable.
estimate the difference between two or more groups.

Decide which Test is right for us

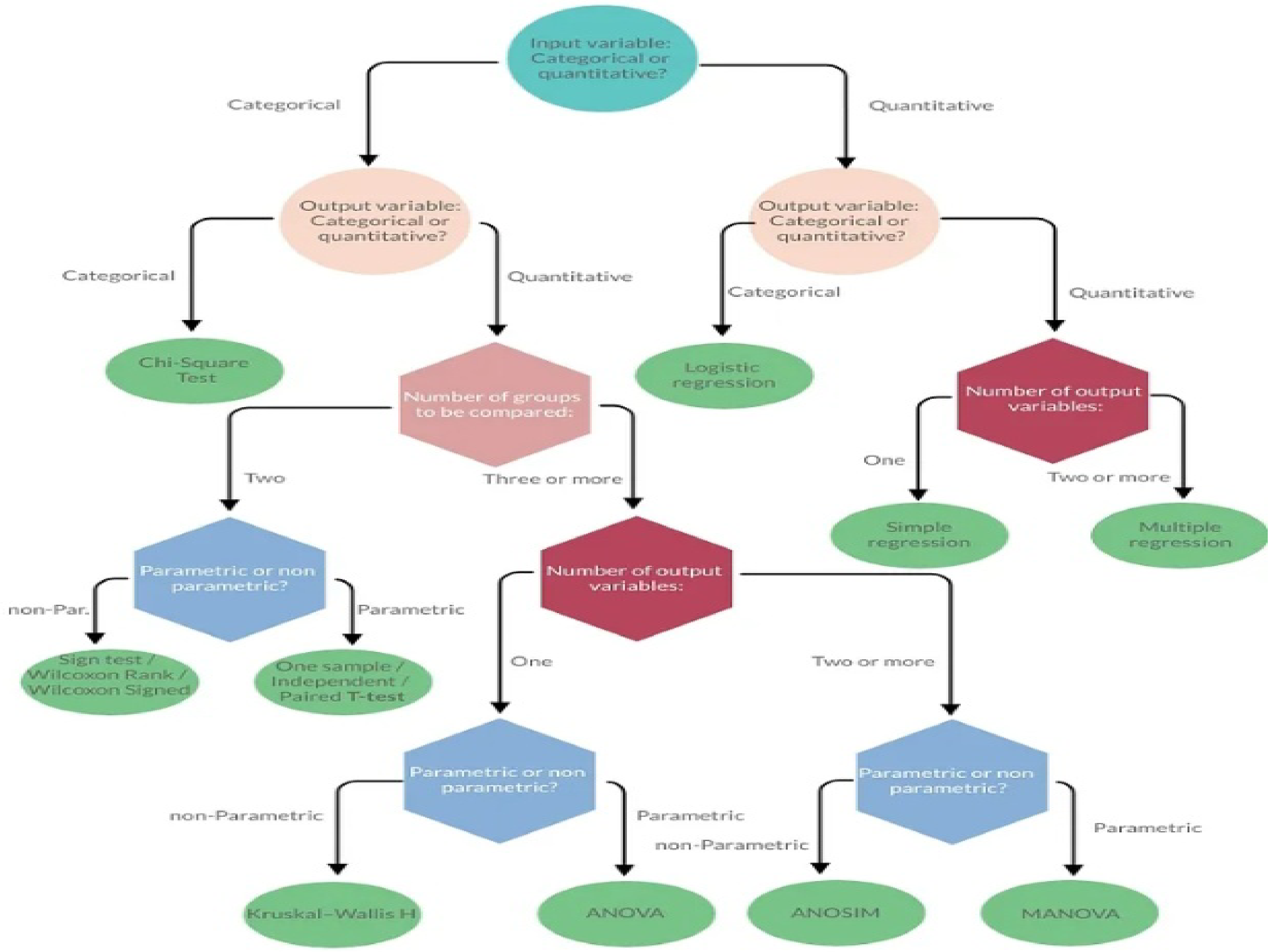
As we have said, there are many statistical tests, and each of them can only be applied in specific scenarios. To determine which statistical test to use, you need to know:

the types of variables that you’re dealing with (categorical, quantitative etc.)

Whether your data meets certain assumptions (Independence of observations Homogeneity of variance,Normality of data)
Statistic tests are divided into two main categories:
1.Parametric
2.Non-parametric

The former are the most “powerful” and are therefore those recommended to be used, but they must respect the abovementioned assumptions. Let’s see them in detail:ided

Independence of observations: the individual observations (each entry of the variables) are independent of each other.
Normality of data: the data follows a normal distribution. This assumption is required only for quantitative data.
Homogeneity of variance: the variance (i.e ., the distribution, or "spread," of scores around the mean) within each group being compared is similar among all groups. If one group has much more variance than the others, this will reduce the "power" of the test in identifying differences.on.



Input variable: Categorical or quantitative?
Categorical
Quantitative
Output variable: Categorical or quantitative?
Categorical
Quantitative
Output variable: Categorical or quantitative?
Categorical
Quantitative
Chi-Square Test
Number of groups to be compared:
Logistic regression
Number of output variables:
Two
Three or more
One
Two or more
Simple regression
Multiple regression
Parametric or non parametric?
non-Par.
Parametric
Number of output variables:
Sign test / Wilcoxon Rank / Wilcoxon Signed
One sample / Independent / Paired T-test
One
Two or more
Parametric or non parametric?
non-Parametric
Parametric
Parametric or non parametric?
non-Parametric
Parametric
Kruskal-Wallis H
ANOVA
ANOSIM
MANOVA

Flowchart to guide the choose of the correct Statistical Test. [Image by author]

Python libraries for statistical tests
The most famous and supported python libraries that collect the main statistical tests are:
Statsmodel: a Python module that provides classes and functions for the estimation of many different statistical models, as well as for conducting statistical tests, and statistical data exploration.
Pingouin: an open-source statistical package written in Python 3 and based mostly on Pandas and NumPy.
Scipy: a Python-based ecosystem of open-source software for mathematics, science, and engineering.

Coding:

```
plt.figure(figsize=(7,4))
ax = sns.countplot(x='no_employees',
data=df)
ax.set_xticklabels(ax.get_xticklabels(),
                   rotation=45,
                   horizontalalignment='right')
# Then we also display the values for
each bar above it;
for p in ax.patches:
   ax.annotate(p.get_height(),
(p.get_x()+0.25, p.get_height()+0.01),
ha='center')
```

```
plt.figure(figsize=(5,5))
ax = sns.countplot(x='family_history', data=df)
ax.set_xticklabels(ax.get_xticklabels(),
                   horizontalalignment='right')
for p in ax.patches:
   ax.annotate(p.get_height(), (p.get_x()+0.25,
p.get_height()+0.01), ha='center')

df["Timestamp"].head()eight

df_year =
pd.to_datetime(df["Timestamp.1"]).dt.year
df_year.head()
```

```
df["Year"] = df_year
plt.figure(figsize=(5,5))
ax = sns.countplot(x='Year', data=df)
ax.set_xticklabels(ax.get_xticklabels(),
                   horizontalalignment='right')
for p in ax.patches:
   ax.annotate(p.get_height(), (p.get_x()+0.25,
p.get_height()+0.01), ha='center')


df.describe()

print(f'Family History Unique Entries:
{df["family_history"].unique()}')eight
```

```
df["family_history_num"] = df["family_history"].map({"No": 0, "Yes":
1})

df[["family_history", "family_history_num"]].head()

print(f'Self Employed Unique Entries:
{df["self_employed"].unique()}'
      f'\nTreatment Unique Entries: {df["treatment"].unique()}'
      f'\nRemote Work Unique Entries:
{df["remote_work"].unique()}'
      f'\nBenefits Unique Entries: {df["benefits"].unique()}'
      f'\nWellness Program Unique Entries:
{df["wellness_program"].unique()}'
      f'\nSeek Help Unique Entries: {df["seek_help"].unique()}'
      f'\nAnonymity Unique Entries: {df["anonymity"].unique()}'
      f'\nMental Health Consequence Unique Entries:
{df["mental_health_consequence"].unique()}'
      f'\nPhy. Health Consequence Unique Entries:
{df["phys_health_consequence"].unique()}')s
```

```
df["self_employed_num"] = df["self_employed"].map({"No": 0, "Yes":
1})
df["treatment_num"] = df["treatment"].map({"No": 0, "Yes": 1})
df["remote_work_num"] = df["remote_work"].map({"No": 0, "Yes":
1})
df["benefits_num"] = df["benefits"].map({"No": 0, "Yes": 1, "Don't
know": 2})
df["wellness_programs_num"] = df["wellness_program"].map({"No":
0, "Yes": 1, "Don't know": 2})
df["seek_help_num"] = df["seek_help"].map({"No": 0, "Yes": 1, "Don't
know": 2})
df["anonymity_num"] = df["anonymity"].map({"No": 0, "Yes": 1
, "Don't know": 2})
df["mental_health_consequence_num"] =
df["mental_health_consequence"].map({"No": 0, "Yes": 1, "Maybe":
2})
df["phys_health_consequence_num"] =
df["phys_health_consequence"].map({"No": 0, "Yes": 1, "Maybe": 2})

plt.figure(figsize=(12, 10))
sns.heatmap(data.corr(), annot=True)alth_consequence
```