Published in Towards Data Science

Wei-Meng Lee

Oct 18, 2021 · 10 min read ★ · Listen

Statistics in Python — Using ANOVA for Feature Selection

Understand how to use ANOVA for comparing between a categorical and numerical variable

Photo by Gabriel Gurrola on Unsplash

In my previous article, I talked about using the chi-square statistics to select features from a dataset for machine learning. The chi-square test is used when both your independent and dependent variables are all *categorical* variables. However, what if your independent variable is *categorical* and your dependent variable is *numerical*? In this case, you have to use another statistic test known as ANOVA — **An**alysis **o**f **Va**riance.

And so in this article, our discussion will revolve around ANOVA and how you use it in machine learning for feature selection. Like all my previous articles, I will use a concrete example to explain the concept.

Before we get started, it is useful to summarize the different methods for feature selection that we have discussed so far :

Variable 1	Variable 2	Method to test for correlation (dependence)
Numerical	Numerical	Pearson correlation
Categorical (Ordinal)	Numerical	Spearman's rank correlation
Categorical (Nominal)	Categorical (Nominal)	Chi-Square
Categorical (Ordinal/Nominal)	Numerical	ANOVA

Image by author

If you need a refresher on **Pearson correlation**, **Spearman's rank correlation**, and **Chi-Square**, I suggest you go and check them out now (see the links below) and come back to this article once you are done. Some of the concepts discussed in this article is similar to that of the chi-square test, and so I recommend you check that out.

Statistics in Pyth...

In my previous...

towardsdatascience.com

Statistics in Pyth...
Understa nd how t...
towardsdata science.com

Statistics in Pyth...
Understa nd the...
towardsdata science.com

What is ANOVA?

ANOVA is used for testing two variables, where:

- one is a *categorical* variable
- another is a *numerical* variable

ANOVA is used when the categorical variable has *at least 3 groups* (i.e three different unique values).

> *If you want to compare just two groups, use the t-test. I will cover t-test in another article.*

ANOVA lets you know if your numerical variable changes according to the level of the categorical variable.

> ***ANOVA uses the f-tests to statistically test the equality of means***. *F-tests are named after its test statistic, F, which was named in honor of* ***Sir Ronald Fisher***.

Here are some examples that makes it easier to understand when you can use ANOVA.

- You have a dataset containing information of a group of people pertaining to their social media usage and the number of hours they sleep:

Social Media Usage	Hours of sleep
Low	8
Medium	7
High	6

Image by author

You want to find out if the amount of social media usage (categorical variable) has a direct impact on the number of hours of sleep (numerical variable).

- You have a dataset containing three different brands of medication and the number of days for the medication to take effect:

Brand	Number of Days to Take Effect
BrandX	1
BrandY	3
BrandZ	2

Image by author

You want to find out if there is a direct relationship between a specific brand and its effectiveness.

> *ANOVA checks whether there is equal variance between groups of categorical feature with respect to the numerical response.*

If there is equal variance between groups, it means this feature has no impact on the response and hence it (the categorical variable) cannot be considered for model training.

Performing AVONA by hand

The best way to understand ANOVA is to use an example. In the following example, I use a fictitious dataset where I recorded the reaction time of a group of people when they are given a specific type of drink.

Sample Dataset

I have a sample dataset named **drinks.csv** containing the following content:

```
team,drink_type,reaction_
time
1,water,14
2,water,25
3,water,23
4,water,27
5,water,28
6,water,21
7,water,26
8,water,30
9,water,31
10,water,34
1,coke,25
2,coke,26
3,coke,27
4,coke,29
5,coke,25
6,coke,23
```

```
7,coke,22
8,coke,27
9,coke,29
10,coke,21
1,coffee,8
2,coffee,20
3,coffee,26
4,coffee,36
5,coffee,39
6,coffee,23
7,coffee,25
8,coffee,28
9,coffee,27
10,coffee,25
```

There are 10 teams in all — each team comprises of 3 persons. Each person in the team is given three different types of drinks — water, coke, and coffee. After consuming the drink, they were asked to perform some activities and their reaction time recorded. The aim of this experiment is to determine if the drinks have any effect on a person's reaction time.

Let's first load the dataset into a Pandas DataFrame:

```
import pandas as pd
df =
pd.read_csv('drinks.csv')
```

Record the *observation size*, which we will make use of later:

```
observation_size = df.shape[0]   # number of observations
```

observation_size

	team	drink_type	reaction_time
0	1	water	14
1	2	water	25
2	3	water	23
3	4	water	27
4	5	water	28
5	6	water	21
6	7	water	26
7	8	water	30
8	9	water	31
9	10	water	34
10	1	coke	25
11	2	coke	26
12	3	coke	27
13	4	coke	29
14	5	coke	25
15	6	coke	23
16	7	coke	22
17	8	coke	27
18	9	coke	29
19	10	coke	21
20	1	coffee	8
21	2	coffee	20
22	3	coffee	26
23	4	coffee	36
24	5	coffee	39
25	6	coffee	23
26	7	coffee	25
27	8	coffee	28
28	9	coffee	27
29	10	coffee	25

Image by author

Visualizing the dataset

It is useful to visualize the distribution of the data using a

Boxplot:

```
_ =
df.boxplot('reaction_time
', by='drink_type')
```

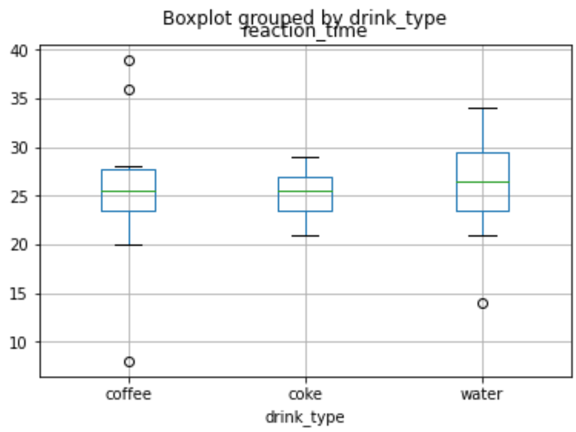


Image by author

You can see that the three types of drinks have about the same median reaction time.

Pivoting the dataframe

To facilitate the calculation for ANOVA, we need to pivot the dataframe:

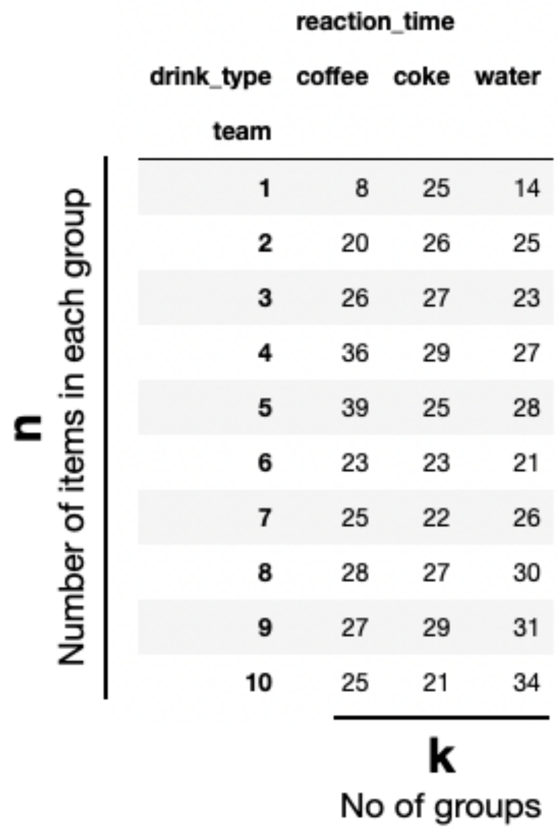
```
df =
df.pivot(columns='drink_t
ype', index='team')
display(df)
```

	reaction_time		
drink_type	coffee	coke	water
team			
1	8	25	14
2	20	26	25
3	26	27	23
4	36	29	27
5	39	25	28
6	23	23	21
7	25	22	26
8	28	27	30
9	27	29	31
10	25	21	34

Image by author

The columns represent the three different types of drinks and the rows represents the 10 teams. We will also use this chance to record the *number of items in each group*, as well as the *number of groups*, which we will make use of later:

```
n = df.shape[0]   # 10;
number of items in each
group
k = df.shape[1]   # 3;
number of groups
```



	reaction_time		
drink_type	coffee	coke	water
team			
1	8	25	14
2	20	26	25
3	26	27	23
4	36	29	27
5	39	25	28
6	23	23	21
7	25	22	26
8	28	27	30
9	27	29	31
10	25	21	34

Image by author

Defining the Hypotheses

You now define your *null hypothesis* and *alternate hypothesis*, just like the chi-square test. They are:

- **H0** (Null hypothesis) — that there is no difference among group means.
- **H1** (Alternate hypothesis) — that at least one group differs significantly from the overall mean of the dependent variable.

Step 1 — Calculating the means for all groups

We are now ready to begin our calculations for ANOVA. First, let's find the mean for each group:

```
df.loc['Group Means'] =
df.mean()
df
```

	reaction_time		
drink_type	coffee	coke	water
team			
1	8.0	25.0	14.0
2	20.0	26.0	25.0
3	26.0	27.0	23.0
4	36.0	29.0	27.0
5	39.0	25.0	28.0
6	23.0	23.0	21.0
7	25.0	22.0	26.0
8	28.0	27.0	30.0
9	27.0	29.0	31.0
10	25.0	21.0	34.0
Group Means	25.7	25.4	25.9

Image by author

From here, you can now calculate the **overall mean**:

```
overall_mean =
df.iloc[-1].mean()
overall_mean    #
25.666666666666668
```

7	25.0	22.0	26.0
8	28.0	27.0	30.0
9	27.0	29.0	31.0
10	25.0	21.0	34.0
Group Means	$\boxed{25.7}$	$\boxed{25.4}$	$\boxed{25.9}$

$$\frac{25.7 + 25.4 + 25.9}{3} = 25.667$$

Image by author

Step 2 — Calculate the Sum of Squares

Now that we have calculated the *overall mean*, we can proceed to calculate the following:

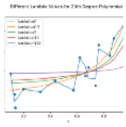
- Sum of squares of all observation — **SS_total**
- Sum of squares within — **SS_within**
- Sum of squares between — **SS_between**

Sum of squares of all observation — SS_total

How to Set X and y in Pandas

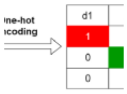
Bran... in Toward...

Complete Guide to Regressional Analysis Using...



Ruks... in Towar...

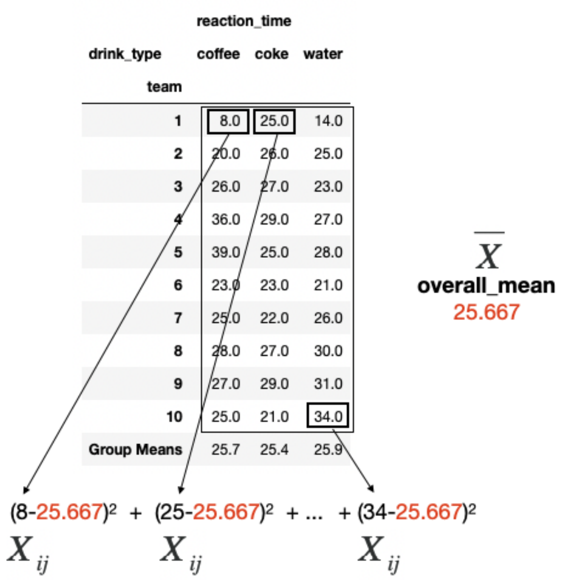
Encoding Categorical Variables: One-h...



The *sum of squares of all observation* is calculated by deducting each observation from the *overall mean*, and then summing all the squares of the differences:

$$SS_{total} = \Sigma(X_{ij} - \overline{X})^2$$

Image by author



drink_type	reaction_time coffee	coke	water
team			
1	8.0	25.0	14.0
2	20.0	26.0	25.0
3	26.0	27.0	23.0
4	36.0	29.0	27.0
5	39.0	25.0	28.0
6	23.0	23.0	21.0
7	25.0	22.0	26.0
8	28.0	27.0	30.0
9	27.0	29.0	31.0
10	25.0	21.0	34.0
Group Means	25.7	25.4	25.9

Image by author

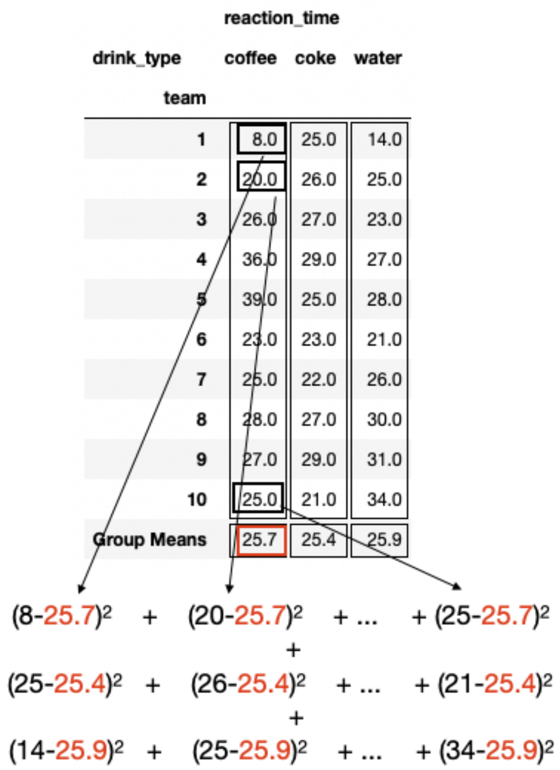
Programmatically, **SS_total** is computed as:

```
SS_total =
(((df.iloc[:-1] -
overall_mean)**2).sum()).
sum()
SS_total   #
```

```
1002.6666666666667
```

Sum of squares within — SS_within

The *sum of squares within* is the sum of squared deviations of scores around their group's mean:



	reaction_time		
drink_type	coffee	coke	water
team			
1	8.0	25.0	14.0
2	20.0	26.0	25.0
3	26.0	27.0	23.0
4	36.0	29.0	27.0
5	39.0	25.0	28.0
6	23.0	23.0	21.0
7	25.0	22.0	26.0
8	28.0	27.0	30.0
9	27.0	29.0	31.0
10	25.0	21.0	34.0
Group Means	25.7	25.4	25.9

$$(8-25.7)^2 + (20-25.7)^2 + \ldots + (25-25.7)^2$$
$$+$$
$$(25-25.4)^2 + (26-25.4)^2 + \ldots + (21-25.4)^2$$
$$+$$
$$(14-25.9)^2 + (25-25.9)^2 + \ldots + (34-25.9)^2$$

Image by author

Programmatically, **SS_within** is computed as:

```
SS_within =
(((df.iloc[:-1] -
df.iloc[-1])**2).sum()).s
um()
```

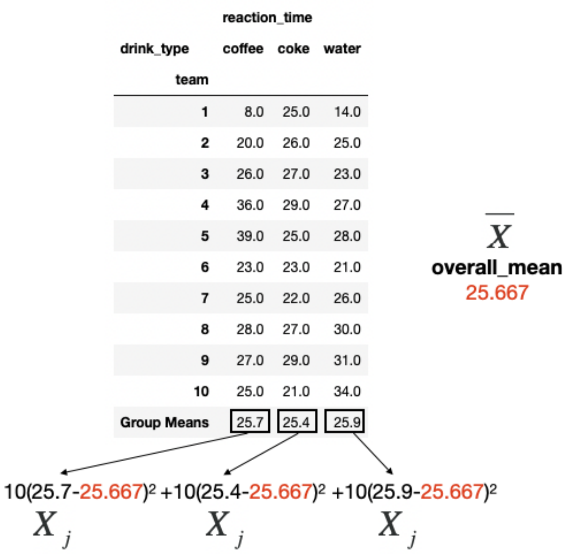
```
SS_within    # 1001.4
```

Sum of Squares between — SS_between

Next we calculate the sum of squares of the group means from the overall mean:

$$SS_{between} = n\Sigma(X_j - \overline{X})^2$$

Image by author



	reaction_time		
drink_type	coffee	coke	water
team			
1	8.0	25.0	14.0
2	20.0	26.0	25.0
3	26.0	27.0	23.0
4	36.0	29.0	27.0
5	39.0	25.0	28.0
6	23.0	23.0	21.0
7	25.0	22.0	26.0
8	28.0	27.0	30.0
9	27.0	29.0	31.0
10	25.0	21.0	34.0
Group Means	25.7	25.4	25.9

Image by author

Programmatically, **SS_between** is computed as:

```
SS_between = (n *
(df.iloc[-1] -
overall_mean)**2).sum()
SS_between    #
```

```
1.2666666666666667
```

You can verify that:

SS_total = SS_between + SS_within

Creating the ANOVA Table

With all the values computed, you can now complete the ANOVA table. Recall you have the following variables:

Variable	Description
observation_size	Number of observations
n	Number of items in each group (teams)
k	Number of groups (e.g. water, coke, coffee)

Image by author

You can compute the various *degrees of freedoms* as follows:

```
df_total = observation_size - 1
# 29
df_within = observation_size - k
# 27
df_between = k - 1
# 2
```

From the above, compute the various *mean squared* values:

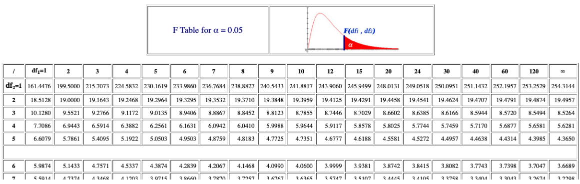
```
mean_sq_between =
SS_between / (k - 1)
# 0.6333333333333335
mean_sq_within = \
    SS_within /
(observation_size - k)
# 37.08888888888889
```

Finally, you can calculate the **F-value**, which is the ratio of two variances:

```
F = mean_sq_between /
mean_sq_within       #
0.017076093469143204
```

> *Recall earlier that I mentioned ANOVA uses the f-tests to statistically test the equality of means.*

Once the F-value is obtained, you now have to refer to the *f-distribution table* (see http://www.socr.ucla.edu/Applets.dir/F_Table.html for one example) to obtain the **f-critical value**. The f-distribution table is organized based on the **α** value (usually 0.05). So you need to first locate the table based on **α**=0.05:



F Table for α = 0.05

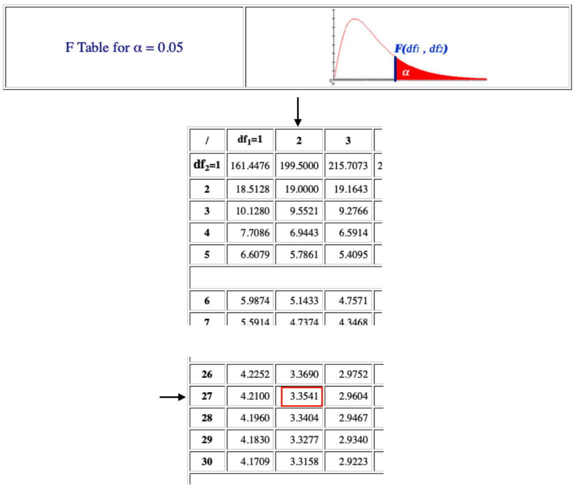
/	df_1=1	2	3	4	5	6	7	8	9	10	12	15	20	24	30	40	60	120	∞
df_2=1	161.4476	199.5000	215.7073	224.5832	230.1619	233.9860	236.7684	238.8827	240.5433	241.8817	243.9060	245.9499	248.0131	249.0518	250.0951	251.1432	252.1957	253.2529	254.3144
2	18.5128	19.0000	19.1643	19.2468	19.2964	19.3295	19.3532	19.3710	19.3848	19.3959	19.4125	19.4291	19.4458	19.4541	19.4624	19.4707	19.4791	19.4874	19.4957
3	10.1280	9.5521	9.2766	9.1172	9.0135	8.9406	8.8867	8.8452	8.8123	8.7855	8.7446	8.7029	8.6602	8.6385	8.6166	8.5944	8.5720	8.5494	8.5264
4	7.7086	6.9443	6.5914	6.3882	6.2561	6.1631	6.0942	6.0410	5.9988	5.9644	5.9117	5.8578	5.8025	5.7744	5.7459	5.7170	5.6877	5.6581	5.6281
5	6.6079	5.7861	5.4095	5.1922	5.0503	4.9503	4.8759	4.8183	4.7725	4.7351	4.6777	4.6188	4.5581	4.5272	4.4957	4.4638	4.4314	4.3985	4.3650
6	5.9874	5.1433	4.7571	4.5337	4.3874	4.2839	4.2067	4.1468	4.0990	4.0600	3.9999	3.9381	3.8742	3.8415	3.8082	3.7743	3.7398	3.7047	3.6689
7	5.5914	4.7374	4.3468	4.1203	3.9715	3.8660	3.7870	3.7257	3.6767	3.6365	3.5747	3.5107	3.4445	3.4105	3.3758	3.3404	3.3043	3.2674	3.2298

Source: http://www.socr.ucla.edu/Applets.dir/F_Table.html

Next, observe that the columns of the f-distribution table is based on **df1** while the rows are based on **df2**. You can get your **df1** and **df2** from the previous variables that we have created:

```
df1 = df_between    # 2
df2 = df_within     # 27
```

Using the values of **df1** and **df2**, you can now locate the **f-critical value** by locating the **df1** column and **df2** row:



/	df_1=1	2	3
df_2=1	161.4476	199.5000	215.7073
2	18.5128	19.0000	19.1643
3	10.1280	9.5521	9.2766
4	7.7086	6.9443	6.5914
5	6.6079	5.7861	5.4095
6	5.9874	5.1433	4.7571
7	5.5914	4.7374	4.3468
26	4.2252	3.3690	2.9752
27	4.2100	3.3541	2.9604
28	4.1960	3.3404	2.9467
29	4.1830	3.3277	2.9340
30	4.1709	3.3158	2.9223

Table from http://www.socr.ucla.edu/Applets.dir/F_Table.html; annotations by author

From the above figure, you can see that the **f-critical value** is **3.3541**. Using this value, you can now decide if you will accept or reject the null hypothesis using the **F-distribution curve**:

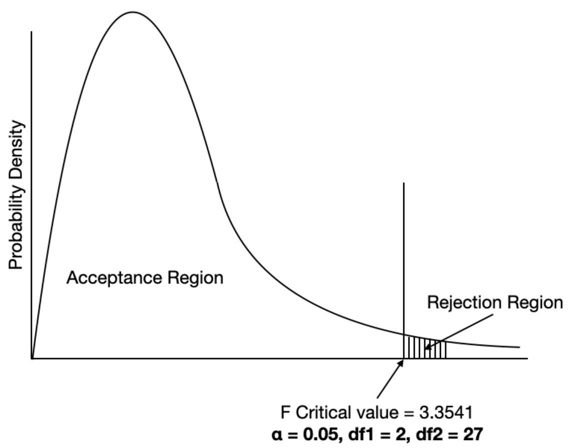


Image by author

Since the **f-value** (0.0171, which is what we can calculated) is less than the f-critical value in the f-distribution table, we accept the null hypothesis — **this means there is no variance in different groups — all the means are the same**.

> *For machine learning, this feature —* drink_type*, should* **not** *be included for training as it seems the different types of drinks have no effect on the reaction time.*
>
> *You should only include a feature for training only if you reject the null hypothesis as this means that the values in the drink types have an effect on the reaction time.*

Using the Stats module to calculate f-score

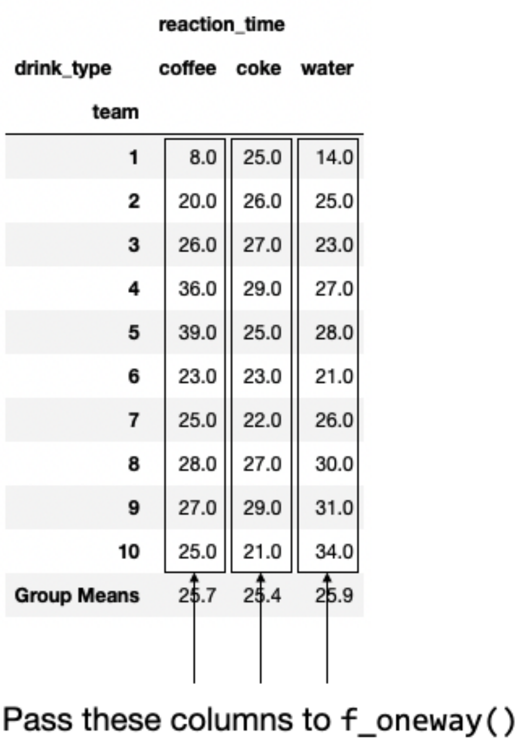
In the previous section, we manually calculated the f-value for our dataset. Actually, there is an easier way — use the **stats** module's **f_oneway()** function to calculate the f-value and p-value:

```
import scipy.stats as
stats

fvalue, pvalue =
stats.f_oneway(
    df.iloc[:-1,0],
    df.iloc[:-1,1],
    df.iloc[:-1,2])


print(fvalue, pvalue)
# 0.0170760934691432
0.9830794846682348
```

The **f_oneway()** function takes the groups as input and returns the ANOVA F and p-value:



	reaction_time		
drink_type	coffee	coke	water
team			
1	8.0	25.0	14.0
2	20.0	26.0	25.0
3	26.0	27.0	23.0
4	36.0	29.0	27.0
5	39.0	25.0	28.0
6	23.0	23.0	21.0
7	25.0	22.0	26.0
8	28.0	27.0	30.0
9	27.0	29.0	31.0
10	25.0	21.0	34.0
Group Means	25.7	25.4	25.9

Image by author

In the above, the **f-value** is **0.017076093469143 2** (identical to the one we calculated manually) and the **p-value** is **0.9830794846682348**.

Observe that the **f_oneway()** function takes in a variable number of arguments:

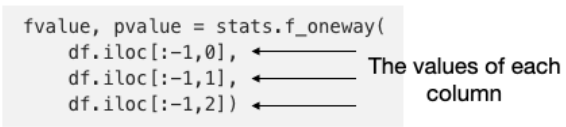


Image by author

If you have many groups, it would be quite tedious to pass in the values of all the groups one by one. So, there is an easier way:

```
fvalue, pvalue =
stats.f_oneway(

*df.iloc[:-1,0:3].T.values
)
```

I will leave the above as an exercise for you to understand how it works.

Using the statsmodels module to calculate f-score

Another way to calculate the f-value is to use the **statsmodel** module. You first build the model using the **ols()** function, and then call the **fit()** function on the instance of the model. Finally, you call the **anova_lm()** function on the fitted model and specify the type of ANOVA test to perform on it:

> *There are 3 types of ANOVA tests to perform, but their discussion is beyond the scope of this article.*

```
import pandas as pd
import statsmodels.api as
sm
from
statsmodels.formula.api
import ols

df =
pd.read_csv('drinks.csv')
```

```
model =
ols('reaction_time ~
drink_type',
data=df).fit()
sm.stats.anova_lm(model,
typ=2)
```

The above code snippet produces the following result, which is the same as the f-value that we calculated earlier (**0.017076**):

	sum_sq	df	F	PR(>F)
drink_type	1.266667	2.0	0.017076	0.983079
Residual	1001.400000	27.0	NaN	NaN

Image by author

The **anova_lm()** function also returns the p-value (**0.983079**). You can make use of the following rules to determine if the categorical variable has any influence on the numerical variable:

- if $p < 0.05$, this means that the categorical variable has significant influence on the numerical variable
- if $p > 0.05$, this means that the categorical variable has

no significant influence on the numerical variable

Since the p-value is now 0.983079 (>0.05), this means that the **drink_type** has no significant influence on the **reaction_time**.

Summary

In this article, I have explained how ANOVA helps to determine if a categorical variable has influence on a numerical variable. So far the ANOVA test that we have discussed is known as the **one-way ANOVA** test. There are a few variations of ANOVA:

- **One-way ANOVA**— used to check how a numerical variable responds to the levels of *one* independent categorical variables
- **Two-way ANOVA** —used to check how a numerical variable responds to the levels of *two* independent categorical variables

- **Multi-way ANOVA** — used to check how a numerical variable responds to the levels of *multiple* independent categorical variables

Using a **two-way ANOVA** or **multi-way ANOVA**, you can investigate the combined impact of two (or more) independent categorical variables on one dependent numerical variable.

I hope you find this article useful. Stay tuned for the next article!