



# Машинное обучение

---

НИЯУ МИФИ, КАФЕДРА ФИНАНСОВОГО МОНИТОРИНГА

КУРС ЛЕКЦИЙ

В.Ю. РАДЫГИН. Д.Ю. КУПРИЯНОВ

ЛЕКЦИЯ 4. ЧАСТЬ 2

# Библиотеки

---

В данной лекции будут рассмотрены примеры с использованием следующих библиотек:

- NumPy – <https://numpy.org/>
- SciPy – <https://scipy.org/>
- Pandas – <https://pandas.pydata.org/>
- Statsmodels – <https://www.statsmodels.org/stable/index.html>

# Часть 1

---

ДИСПЕРСИОННЫЙ АНАЛИЗ

# Дисперсионный анализ

---

Цель дисперсионного анализа – исследование наличия или отсутствия существенного влияния какого-либо качественного или количественного фактора на изменения исследуемого результативного признака.

Для этого фактор, предположительно имеющий или не имеющий существенного влияния, разделяют на классы градации (говоря иначе, группы) и выясняют, одинаково ли влияние фактора путём исследования значимости связи между средними в наборах данных, соответствующих градациям фактора.

# Дисперсионный анализ

---

Фактически, дисперсионный анализ применяется для изучения влияния одной или нескольких независимых переменных на одну зависимую переменную. Влияние изучается с точки зрения изменения среднего значения зависимой переменной. Причём зависимая переменная (результативный признак) должна быть представлена хотя бы в шкале отношений, а независимые переменные (влияющие факторы) должны быть представлены хотя бы в номинальных шкалах.

Например, зависимая переменная – доход. Независимые переменные: пол, уровень возраста, образование, семейный статус. Исследовать, как зависит доход от пола, уровня возраста, образования, семейного статуса.

Если фактор один, то дисперсионный анализ называют однофакторным (one-way). В противном случае его называют многофакторным (N-way). Иногда отдельно говорят о двухфакторном анализе (Two-way).

# Нулевая и альтернативная гипотезы

---

В дисперсионном анализе в качестве нулевой гипотезы ( $H_0$ ) выступает гипотеза, что «Средние величины зависимой переменной для всех значений независимой(ых) переменной(ых) одинаковы».

Альтернативная гипотеза ( $H_1$ ) это «Средние величины зависимой(ы) переменной(ых) для различных значений независимой переменной различны».

# Эффект взаимодействия

---

В отличие от обычного исследования взаимосвязи отдельных независимых переменных и зависимой переменной, дисперсионный анализ позволяет также изучить влияние эффекта взаимодействия независимых факторов на результат. Таким образом, мы можем исследовать не только взаимосвязь между результативным признаком и отдельными факторами, но и взаимосвязь между результативным признаком и комбинациями факторов.

В этом и заключается основное преимущество дисперсионного анализа.

# Идея дисперсионного анализа

---

Чем дальше друг от друга отстоят средние значения в группах, тем меньше вероятность, что верна  $H_0$ . Если  $H_0$  верна, то все группы можно считать, полученными из **одной общей выборки** с конкретными средним  $\mu$  и дисперсией  $\sigma^2$  зависимой переменной.

Тогда получим две независимые точечные оценки  $\sigma^2$  и сравним их!

1. Оценим  $\sigma^2$  на основе дисперсии групповых средних (как будто это выборочные средние).
2. Оценим  $\sigma^2$  на основе дисперсий внутри групп.



# Идея дисперсионного анализа

---

Пусть у нас  $k$  групп. В каждой  $i$ -й группе  $n_i$  наблюдений. Тогда:

1. Вычислим для каждой группы мат. ожидание  $x_{cpi}$ .
2. Вычислим общее мат. ожидание  $x_{cp}$ .
3. Вычислим общую дисперсию между группами  $MSB = \frac{\sum_i n_i (x_{cpi} - x_{cp})^2}{k-1}$
4. Вычислим дисперсии внутри групп:  $S_i^2 = \sum_j (x_{i,j} - x_{cpi})^2$ ,  $D_i = \frac{S_i^2}{n_i-1}$
5. Вычислим общую дисперсию внутри групп  $MSW = \frac{S_1^2 + S_2^2 + \dots + S_k^2}{n-k} = \frac{\sum_i \sum_j (x_{i,j} - x_{cpi})^2}{n-k}$
6.  $F = MSB / MSW$

# Зарубежные обозначения

---

$SSB = MSB * DFB$ , где  $DFB$  – число степеней свободы групп  $= k - 1$

$SSW = MSW * DFW$ , где  $DFW$  – оставшееся число степеней  $= n - k$ .

# Условия применимости

---

1. Нормальность распределения зависимой переменной (выборка наблюдений для каждой комбинации факторов должна быть взята из нормально-распределённой выборки).
2. Гомоскедастичность зависимой переменной (гомогенность дисперсии).
3. Независимость наблюдений.
4. Независимость факторов (для многофакторного дисперсионного анализа).

# Что надо проверить на деле

---

1. Нормальность распределения можно проверить известными ранее тестами Шапиро-Уилка, Андерсона-Дарлинга, Колмогорова-Смирнова или использовать Критерий Жака (Харке)-Бера.
2. Гомоскедастичность зависимой переменной можно проверить при помощи универсального теста (его ещё называют омнибус тест).
3. Независимость наблюдений означает отсутствие автокорреляции для исследуемой зависимой переменной. Проверить отсутствие автокорреляции можно при помощи Критерия Дарбина-Уотсона.
4. Независимость факторов (для многофакторного дисперсионного анализа) — отсутствие мультиколлинеарности между параметрами.
5. Независимость выборок одного уровня\* относительно другого — отсутствие мультиколлинеарности между выборками различных уровней одного параметра (и между возмущениями).

\* Уровни — это значения независимой переменной в номинальной шкале или группы значений в других шкалах.

# Проверка нормальности

---

Проверку нормальности стоит проводить двумя способами. Сначала взгляните на гистограмму. Если из неё сразу видно, что распределение не нормально, то произведите трансформацию данных для приведения к нормальному распределению. Например, лог-трансформацию.

Если число наблюдений достаточно мало, то стоит использовать тесты нормальности, например, Шапиро-Уилка. При невыполнении тестов нужно проводить трансформацию данных ( $p\text{-value} > 0.05$  – «вероятность наличия нормальности при таких данных не мала»).

Если число наблюдений велико, то слабыми (невидимыми визуально на гистограмме) отклонениями от нормальности можно пренебречь. ANOVA не настолько чувствительна к данному требованию при большом числе наблюдений.

Конечно, при большом числе наблюдений всегда можно использовать такие тесты, как тест Колмогорова-Смирнова, Андерсона-Дарлинга или Жака (Харке)-Бера, и в случае их невыполнения выполнить трансформацию ( $p\text{-value} > 0.05$  – «вероятность наличия нормальности при таких данных не мала»).

# Асимметрия

---

Асимметрия может показывать близость распределения к нормальному (асимметрия показывает отклонение в сторону центрального пика). Так как графически нормальное распределение является симметричным:

- если асимметрия  $< -1$  или  $> 1$ , то распределения крайне асимметрично;
- если асимметрия в диапазоне от  $-1$  до  $-0.5$  или в диапазоне от  $0.5$  до  $1$ , то распределение средне асимметрично и в некоторых случаях можно это считать достаточным;
- если асимметрия в диапазоне от  $-0.5$  до  $0.5$ , то распределение приблизительно симметрично.

# Эксцесс

---

Эксцесс показывает высоту и резкость центрального пика по сравнению со стандартной колоколообразной кривой (отклонение центрального пика вниз или вверх от нормального).

Эксцесс нормального распределения должен быть близок к 0 (если рассматривается значение Kurtosis, смещённое на 3, иначе около 3). Метод `ols` печатает несмещённое на 3 значение, поэтому нам стоит смотреть на значения, близкие к 3.

# Проверка гомоскедастичности

---

Проверку гомоскедастичности стоит проводить двумя способами. Сначала взгляните на scatter-диаграмму. Если на ней полоса точек имеет одинаковую ширину по всей координатной плоскости и нет отдельных ярких выбросов, то можно сделать предварительный вывод о гомоскедастичности, иначе надо либо проводить трансформацию, либо удалять выбросы, либо и то и другое.

Если графически все хорошо, то стоит провести статистический тест. Это может быть универсальный тест (его ещё называют омнибус тест) ( $p\text{-value} > 0.05$  – «вероятность наличия гомоскедастичности при таких данных не мала») или Тест Левена ( $p\text{-value} > 0.05$  – «вероятность наличия гомоскедастичности при таких данных не мала»). Нулевая гипотеза для данных тестов говорит, что дисперсия во всех сравниваемых группах одинаковая.



# Проверка отсутствия автокорреляции

---

Для проверки отсутствия автокорреляции можно использовать статистический критерий Дарбина-Уотсона.

В случае отсутствия автокорреляции статистика Дарбина-Уотсона близка к 2. При положительной автокорреляции статистика Дарбина-Уотсона стремится к нулю. При отрицательной автокорреляции статистика Дарбина-Уотсона стремится к 4.

# Тест отсутствия мультиколлинеарности

---

Проверку отсутствия мультиколлинеарности стоит проводить двумя способами. Сначала стоит построить графики наблюдений для разных уровней фактора (если проверяется отсутствие мультиколлинеарности для выборок разных уровней, иначе сравниваем выборки различных факторов). Графики могут быть получены каким-либо способом аппроксимации. Если их поведение носит одинаковый характер, то можно говорить о наличии мультиколлинеарности.

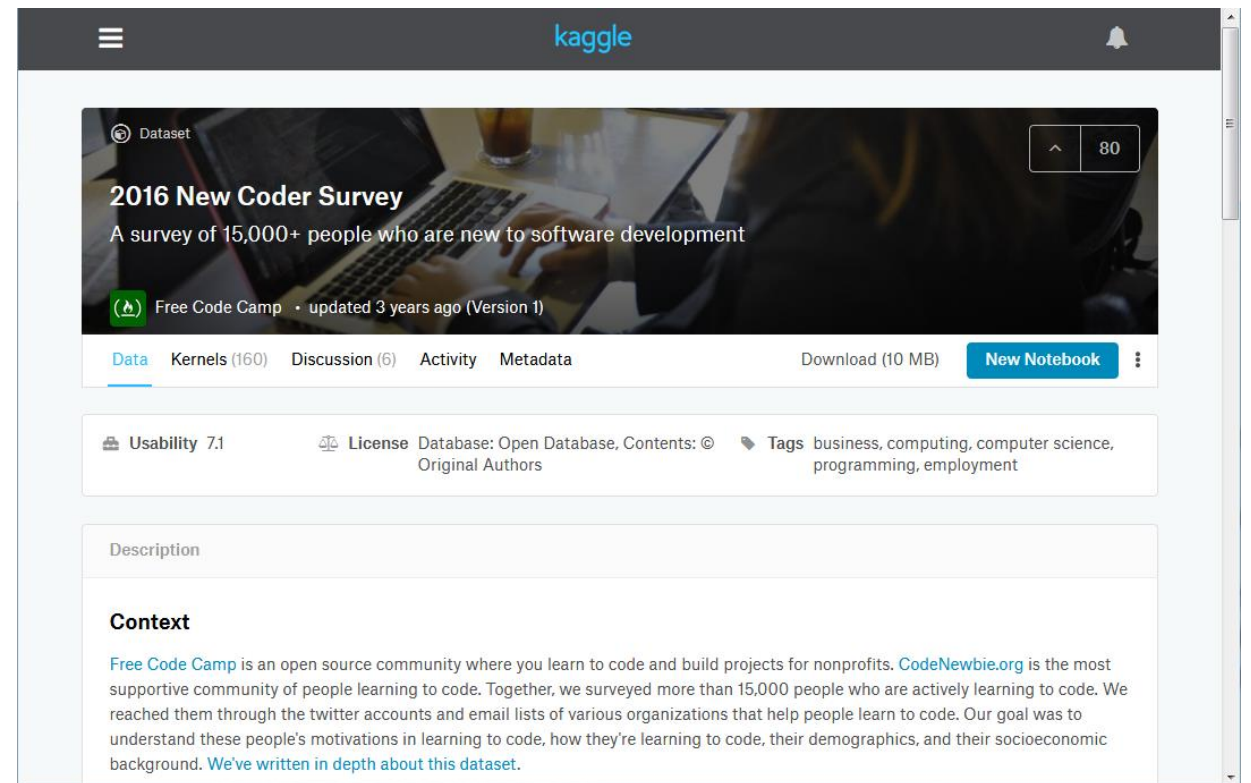
Если графически все хорошо, то стоит провести статистический тест. Для этого можно использовать условное число (Condition Number). Если условное число больше 20, то можно говорить о наличии мультиколлинеарности.

Многие учёные считают, что наличием мультиколлинеарности между выборками различных уровней можно пренебречь.

Часто мультиколлинеарность является следствием «переобучения» модели.

# Набор для примеров

Для изучения данной тематики используем набор данных с сайта [kaggle.com](https://www.kaggle.com), посвящённый опросам начинающих кодеров [1].



# Загрузка и подготовка набора

```
example4runner.py - E:\Works\Victor\Students\STDB\Lecture5\example4runner.py (3.7.2)
File Edit Format Run Options Window Help

import numpy as np
import pandas as pd

import statsmodels.api as sm
from statsmodels.formula.api import ols
from scipy import stats
import matplotlib.pyplot as plt

pd.set_option('display.max_columns', 2000)
pd.set_option('display.width', 2000)

table = pd.read_csv("../Lab3/2016-FCC-New-Coders-Survey-Data.csv", sep = ",",
                    dtype = {'CodeEventOther': str, 'JobRoleInterestOther': str})

pd.to_numeric(table['Income'], errors = 'coerce')

table = table.loc[:, ['Income', 'Gender', 'MaritalStatus', 'JobWherePref', 'SchoolDegree']]
table2 = table.dropna().sample(100, random_state = 1234567)
print(table2)
```

Ln: 20 Col: 58

# Готовые данные

```
Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: E:\Works\Victor\Students\STDB\Lec5\example4runner.py =====
Income Gender MaritalStatus JobWherePref SchoolDegree
2528 15000.0 female single, never married from home some college cre
dit, no degree
1475 100000.0 female married or domestic partnership no preference professional degree (MBA,
MD, JD, etc.)
13121 70000.0 male married or domestic partnership no preference some college cre
dit, no degree
1360 20000.0 male married or domestic partnership in an office with other developers professional degree (MBA,
MD, JD, etc.)
3239 42000.0 male married or domestic partnership in an office with other developers so
me high school
15468 34800.0 male single, never married in an office with other developers bac
helor's degree
2447 16000.0 male married or domestic partnership from home so
...
Ln: 71 Col: 4
```

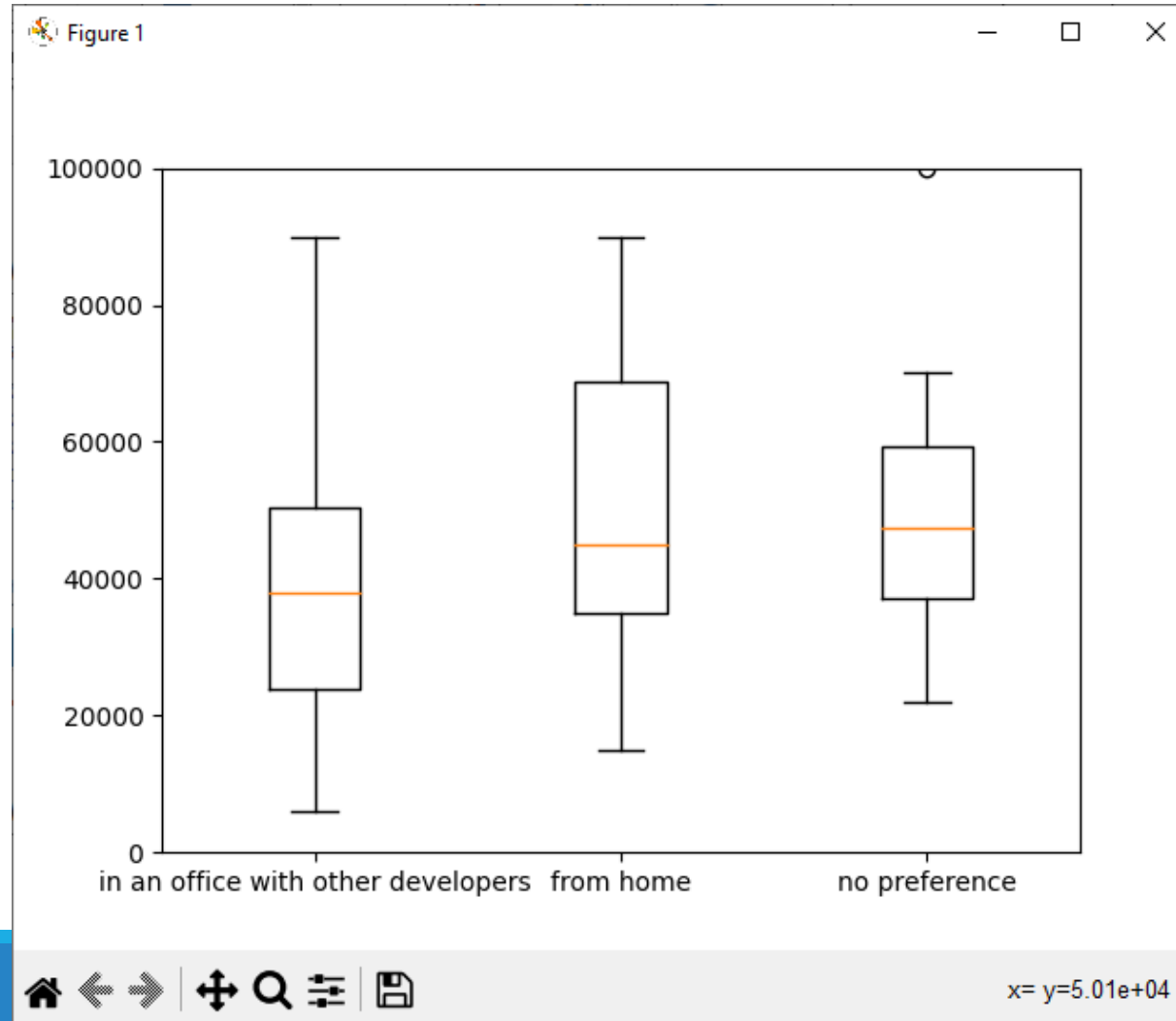
# Анализ данных по уровням (группам)

---

Для первичного анализа данных построим диаграммы размаха (ящики с усами, усиковые диаграммы, коробчатые диаграммы) для каждого уровня предпочтений в месте работы.

```
table2_1 = table2[table2['JobWherePref'] == 'in an office with other developers']
table2_2 = table2[table2['JobWherePref'] == 'from home']
table2_3 = table2[table2['JobWherePref'] == 'no preference']
plt.ylim(0, 100000)
plt.boxplot((table2_1['Income'], table2_2['Income'], table2_3['Income']),
            labels = ['in an office with other developers',
                      'from home', 'no preference'])
plt.show()
```

# Анализ данных по уровням (группам)



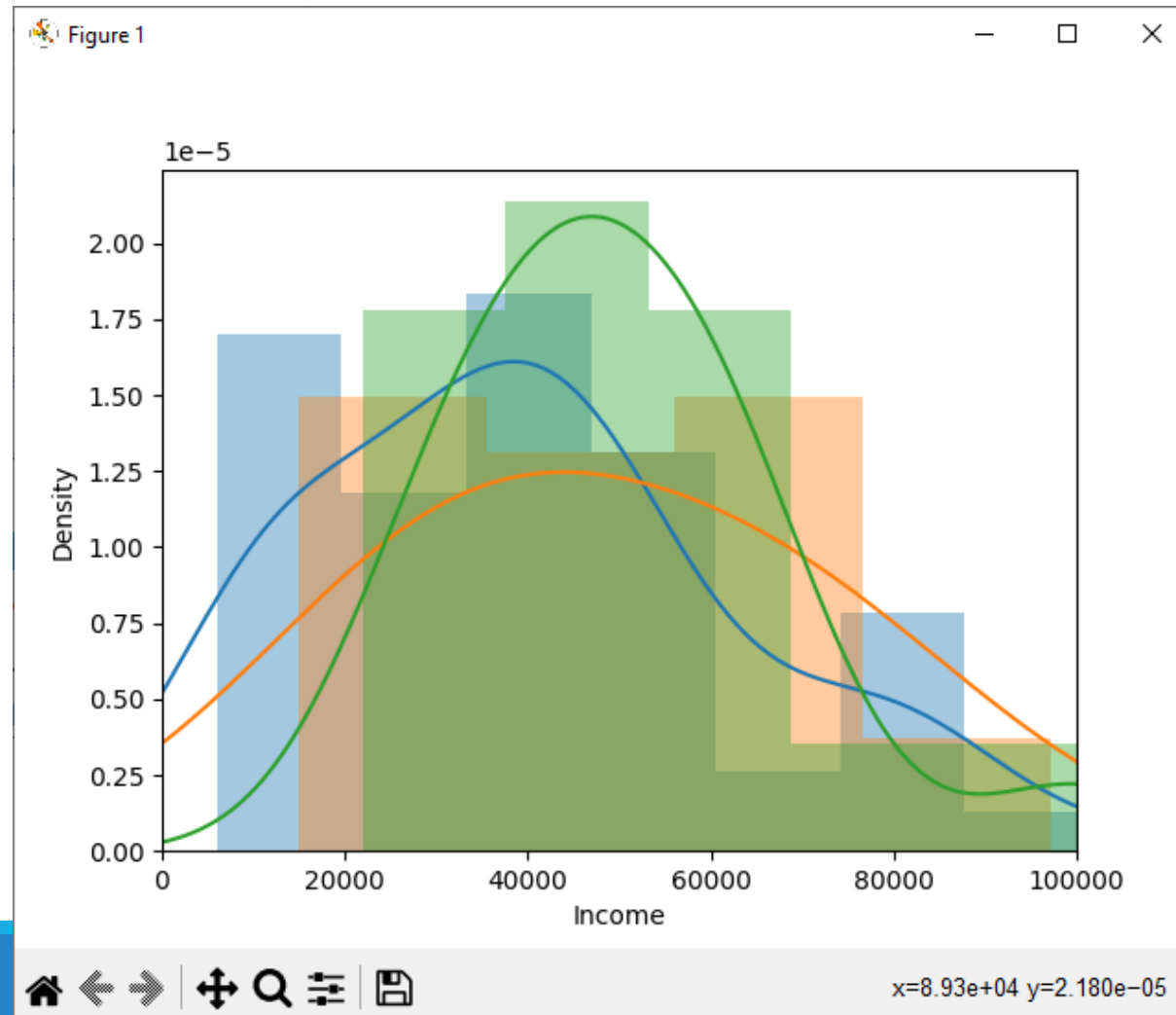
# Графическая проверка отсутствия мультиколлинеарности между группами

---

```
import seaborn as sns
sns.distplot(table2_1['Income'], label = 'office')
sns.distplot(table2_2['Income'], label = 'home')
sns.distplot(table2_3['Income'], label = 'no preference')
plt.xlim(0, 100000)
plt.show()
```



# Графическая проверка отсутствия мультиколлинеарности между группами



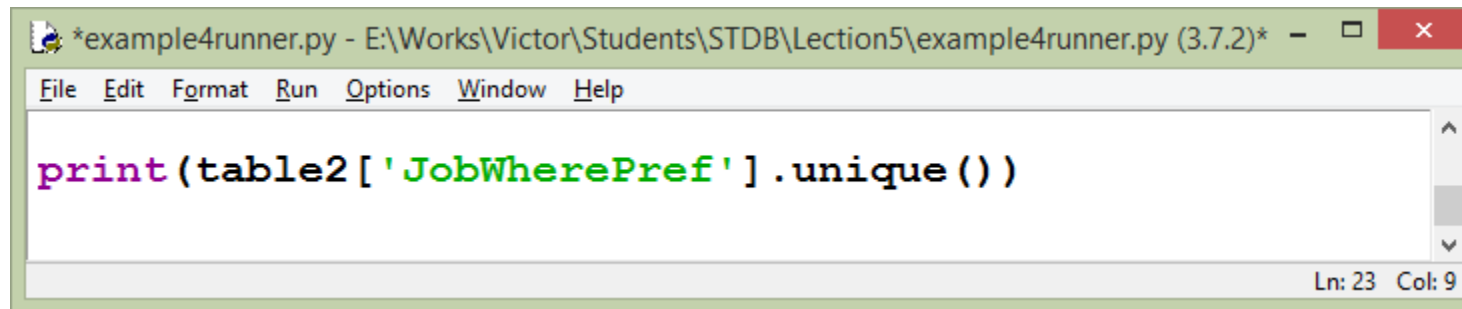
# Однофакторный анализ

---

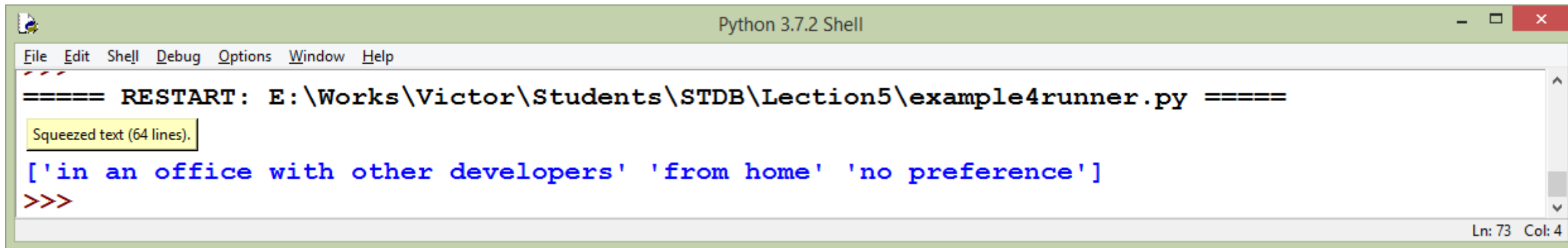
Проведём исследование зависимости дохода (Income) от предпочитаемого места работы (JobWherePref). Для начала используем библиотеку `scipy`.

Метод `f_oneway` модуля `stats` требует, чтобы ему передали отдельно серии для каждого из возможных значений фактора. Придётся их сформировать. Но для начала узнаем, какие значения фактора у нас есть.

# Значения фактора



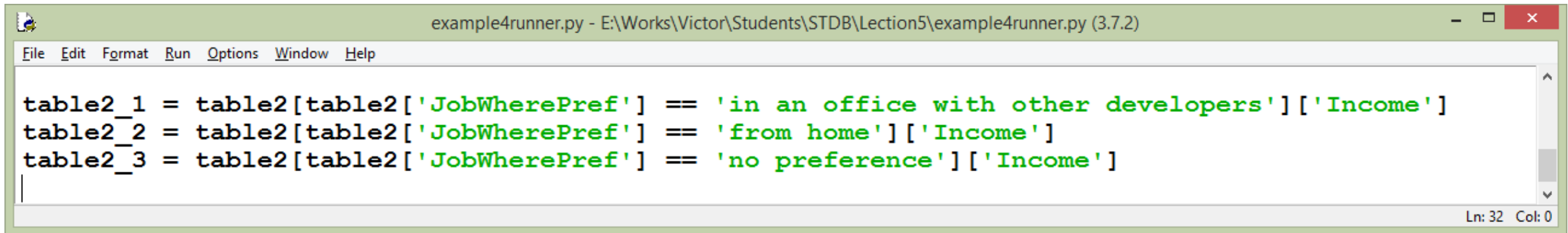
```
print(table2['JobWherePref'].unique())
```



```
==== RESTART: E:\Works\Victor\Students\STDB\Lecton5\example4runner.py ====  
Squeezed text (64 lines).  
['in an office with other developers' 'from home' 'no preference']  
>>>
```

# Выделим группы

---



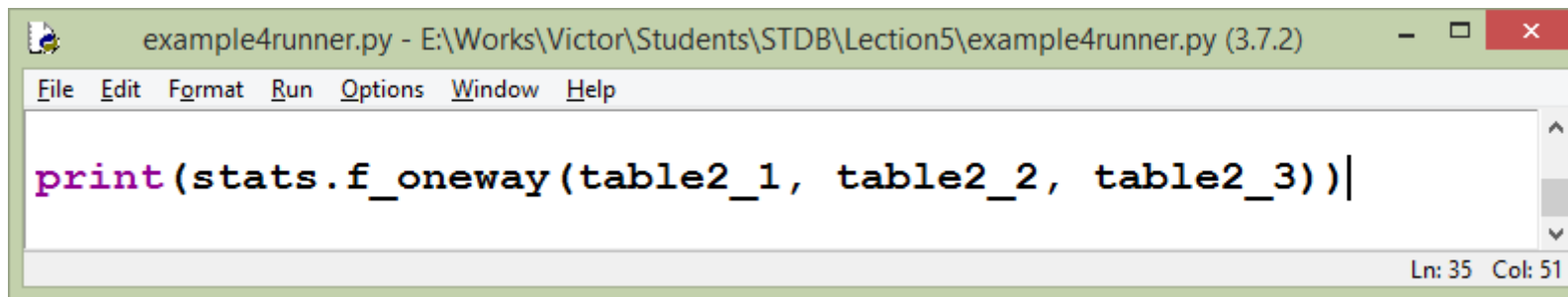
The screenshot shows a Python IDE window titled "example4runner.py - E:\Works\Victor\Students\STDB\Lection5\example4runner.py (3.7.2)". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code in the editor is as follows:

```
table2_1 = table2[table2['JobWherePref'] == 'in an office with other developers']['Income']  
table2_2 = table2[table2['JobWherePref'] == 'from home']['Income']  
table2_3 = table2[table2['JobWherePref'] == 'no preference']['Income']
```

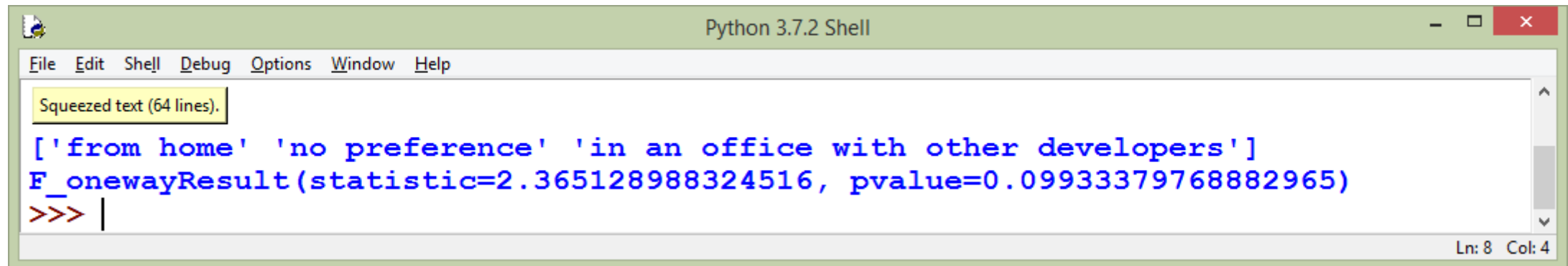
The status bar at the bottom right indicates "Ln: 32 Col: 0".

# Применим ANOVA

---



```
print(stats.f_oneway(table2_1, table2_2, table2_3))
```



```
Squeezed text (64 lines).  
['from home' 'no preference' 'in an office with other developers']  
F_onewayResult(statistic=2.365128988324516, pvalue=0.09933379768882965)  
>>> |
```

# Интерпретация результата

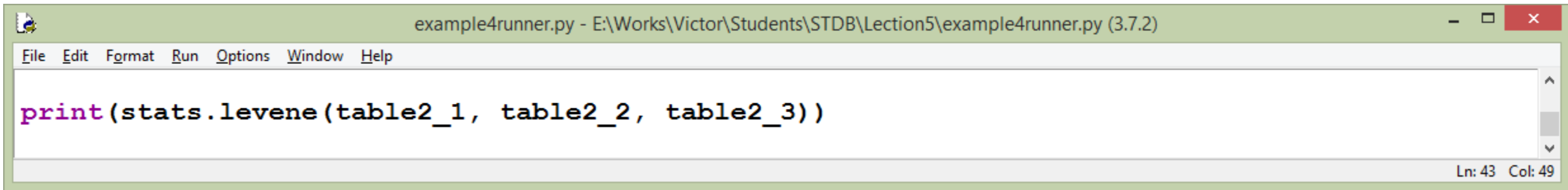
---

Если взять в качестве достаточного 5% уровень значимости (0,05), то можно сказать, что полученное  $p\text{-value} = 0,0993 > 0,05$ . Из это следует, что вероятность гипотезы  $H_0$  при таких данных не мала и можно говорить о независимости среднего значения дохода от уровня предпочтений в месте работы.

Можно ли доверять этому результату? НЕТ! Мы не проверили необходимые требования дисперсионного анализа.

# Проверим гомоскедастичность

---

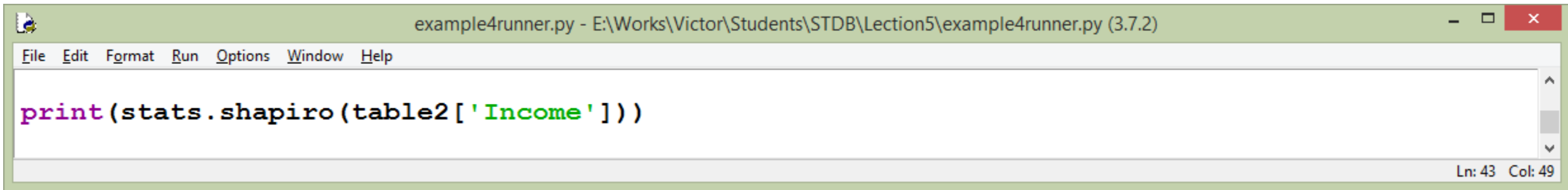


The image shows a screenshot of a Python IDE window. The title bar reads "example4runner.py - E:\Works\Victor\Students\STDB\Lecture5\example4runner.py (3.7.2)". The menu bar includes "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the code `print(stats.levene(table2_1, table2_2, table2_3))`. The status bar at the bottom right indicates "Ln: 43 Col: 49".

```
print(stats.levene(table2_1, table2_2, table2_3))
```

# Проверим нормальность

---



The image shows a screenshot of a Python IDE window. The title bar reads "example4runner.py - E:\Works\Victor\Students\STDB\Lecture5\example4runner.py (3.7.2)". The menu bar includes "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the code `print(stats.shapiro(table2['Income']))`. The status bar at the bottom right indicates "Ln: 43 Col: 49".

```
example4runner.py - E:\Works\Victor\Students\STDB\Lecture5\example4runner.py (3.7.2)
```

File Edit Format Run Options Window Help

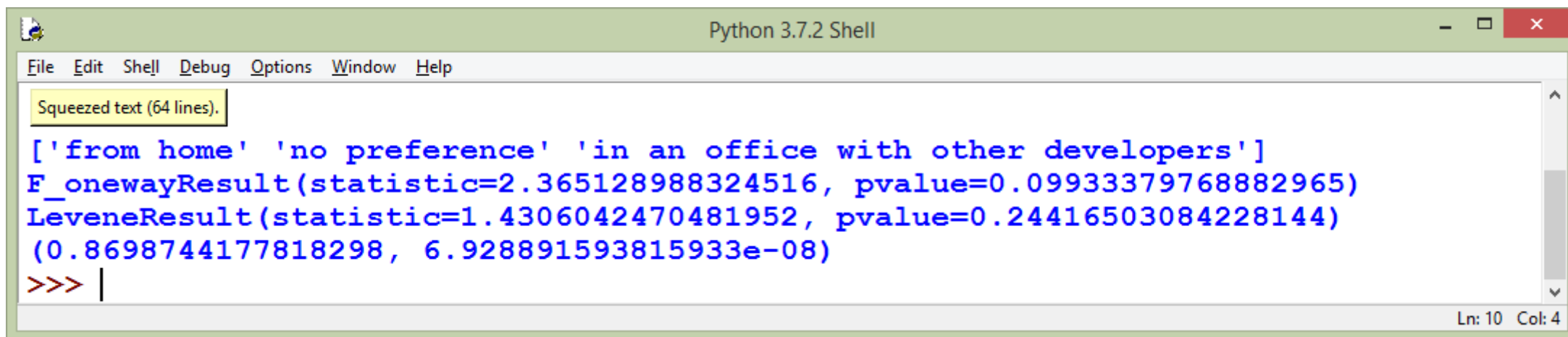
```
print(stats.shapiro(table2['Income']))
```

Ln: 43 Col: 49



# Не всё хорошо!

---

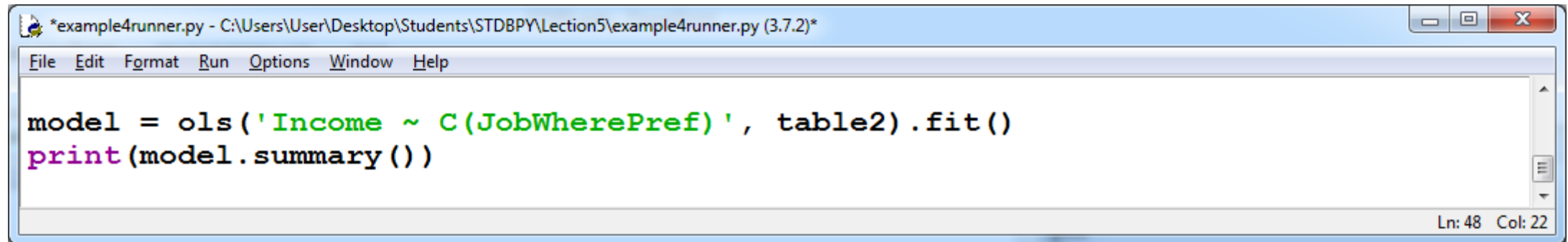


A screenshot of a Python 3.7.2 Shell window. The window has a title bar with the text "Python 3.7.2 Shell" and standard window controls. Below the title bar is a menu bar with options: File, Edit, Shell, Debug, Options, Window, and Help. The main text area shows a list of strings: `['from home' 'no preference' 'in an office with other developers']`. Below this, there are two lines of statistical results: `F_onewayResult(statistic=2.365128988324516, pvalue=0.09933379768882965)` and `LeveneResult(statistic=1.4306042470481952, pvalue=0.24416503084228144)`. The prompt `>>>` is visible at the bottom left of the text area. A status bar at the bottom right shows "Ln: 10 Col: 4".

```
Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Squeezed text (64 lines).
['from home' 'no preference' 'in an office with other developers']
F_onewayResult(statistic=2.365128988324516, pvalue=0.09933379768882965)
LeveneResult(statistic=1.4306042470481952, pvalue=0.24416503084228144)
(0.8698744177818298, 6.928891593815933e-08)
>>> |
Ln: 10 Col: 4
```

# Statsmodels ANOVA

---



A screenshot of a Python IDE window titled `*example4runner.py - C:\Users\User\Desktop\Students\STDBPY\Lection5\example4runner.py (3.7.2)*`. The window has a menu bar with `File`, `Edit`, `Format`, `Run`, `Options`, `Window`, and `Help`. The main text area contains the following Python code:

```
model = ols('Income ~ C(JobWherePref)', table2).fit()
print(model.summary())
```

The status bar at the bottom right of the window indicates `Ln: 48 Col: 22`.

# Немного о формуле в OLS

---

Общий вид формул в OLS:

Зависимая переменная ~ комбинация независимых переменных

Комбинация может включать сами переменные и знаки +, \*, :. Если переменная содержит числа, а их надо обрабатывать, как номинальные значения, то переменная берется внутрь функции C().

Например:

`Income ~ Gender`

`Income ~ C(Gender)`

Дают одинаковый однофакторный анализ.

`Income ~ Gender + MaritalStatus`

Даёт двухфакторный анализ без синергетического исследования

`Income ~ Gender + MaritalStatus + Gender : MaritalStatus`

Даёт двухфакторный анализ, включая синергетического исследования

`Income ~ Gender * MaritalStatus`

Даёт двухфакторный анализ, включая синергетического исследования

# Результат

---

## OLS Regression Results

```
=====
Dep. Variable:          Income    R-squared:                0.046
Model:                  OLS       Adj. R-squared:           0.027
Method:                 Least Squares    F-statistic:              2.365
Date:                   Wed, 13 Nov 2019    Prob (F-statistic):       0.0993
Time:                   12:19:46           Log-Likelihood:          -1161.6
No. Observations:      100              AIC:                     2329.
Df Residuals:          97                BIC:                     2337.
Df Model:               2
Covariance Type:       nonrobust
```

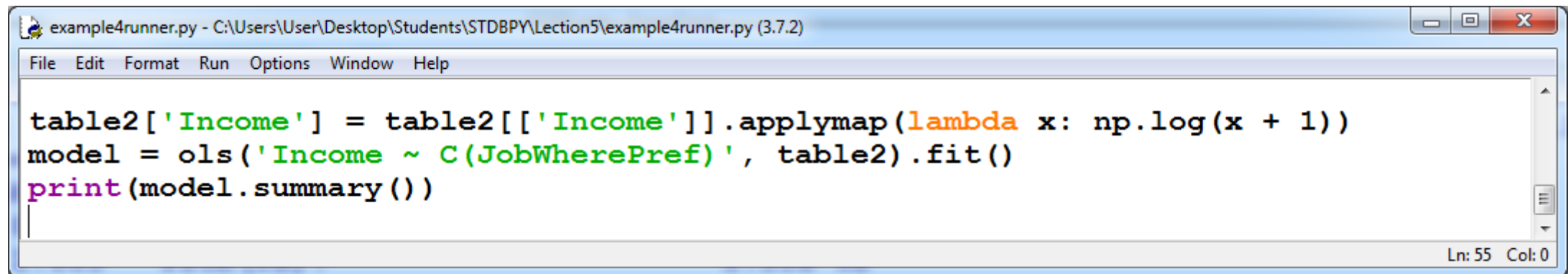
# Проверка требований

	coef	std err	t	P> t	[0.025	0.975]
Intercept	5.396e+04	5342.493	10.100	0.000	4.34e+04	6.46e+04
C(JobWherePref) [T.in an office with other developers]	-1.334e+04	6464.827	-2.063	0.042	-2.62e+04	-507.658
C(JobWherePref) [T.no preference]	-4231.3034	8352.838	-0.507	0.614	-2.08e+04	1.23e+04
Omnibus:	58.722	Durbin-Watson:	1.883			
Prob(Omnibus) :	0.000	Jarque-Bera (JB) :	284.876			
Skew:	1.885	Prob(JB) :	1.38e-62			
Kurtosis:	10.360	Cond. No.	4.50			

Также не всё хорошо!

# Лог-трансформация

---

A screenshot of a Python IDE window titled 'example4runner.py - C:\Users\User\Desktop\Students\STDBPY\Lection5\example4runner.py (3.7.2)'. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The main text area contains three lines of Python code: 

```
table2['Income'] = table2[['Income']].applymap(lambda x: np.log(x + 1))  
model = ols('Income ~ C(JobWherePref)', table2).fit()  
print(model.summary())
```

 The code is color-coded: 'Income' is green, 'lambda' is orange, 'np.log' is blue, 'x' is black, 'x + 1' is blue, 'C' is green, and 'JobWherePref' is green. The status bar at the bottom right shows 'Ln: 55 Col: 0'.

# Теперь условия выполнены!

---

```
=====
Omnibus:                4.153    Durbin-Watson:                2.102
Prob(Omnibus) :         0.125    Jarque-Bera (JB) :          3.865
Skew:                  -0.481    Prob(JB) :                  0.145
Kurtosis:              3.022    Cond. No.                   4.50
=====
```

# Результат

---

```
=====
Dep. Variable:                Income    R-squared:                0.067
Model:                        OLS       Adj. R-squared:           0.047
Method:                       Least Squares    F-statistic:              3.456
Date:                         Wed, 13 Nov 2019    Prob (F-statistic):       0.0355
Time:                         12:27:17          Log-Likelihood:           -94.504
No. Observations:             100             AIC:                     195.0
Df Residuals:                  97             BIC:                     202.8
Df Model:                      2
Covariance Type:              nonrobust
```



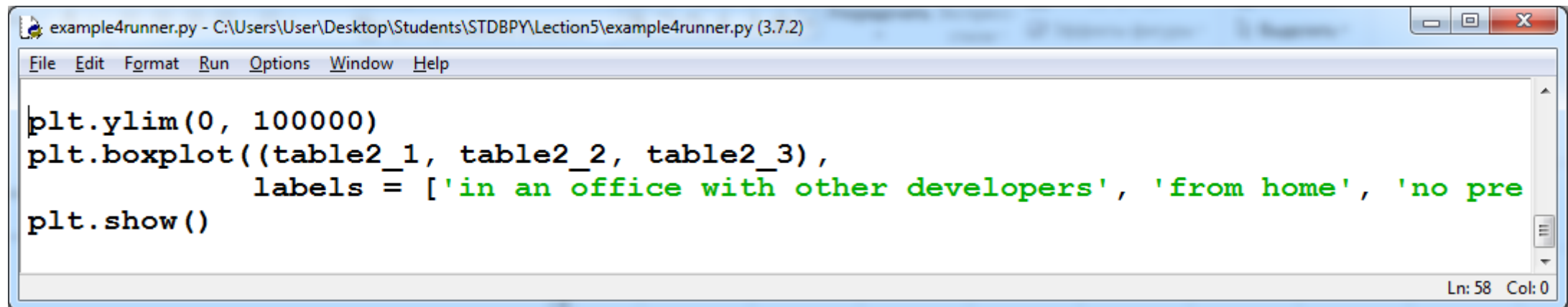
# Интерпретация результата

---

Если взять в качестве достаточного 5% уровень значимости (0,05), то можно сказать, что полученное  $p\text{-value} = 0,0355 < 0,05$ . Из это следует, что вероятность гипотезы  $H_0$  при таких данных мала и можно говорить о зависимости лог-трансформированного среднего значения дохода от уровня предпочтений в месте работы.

# Графическая проверка

---

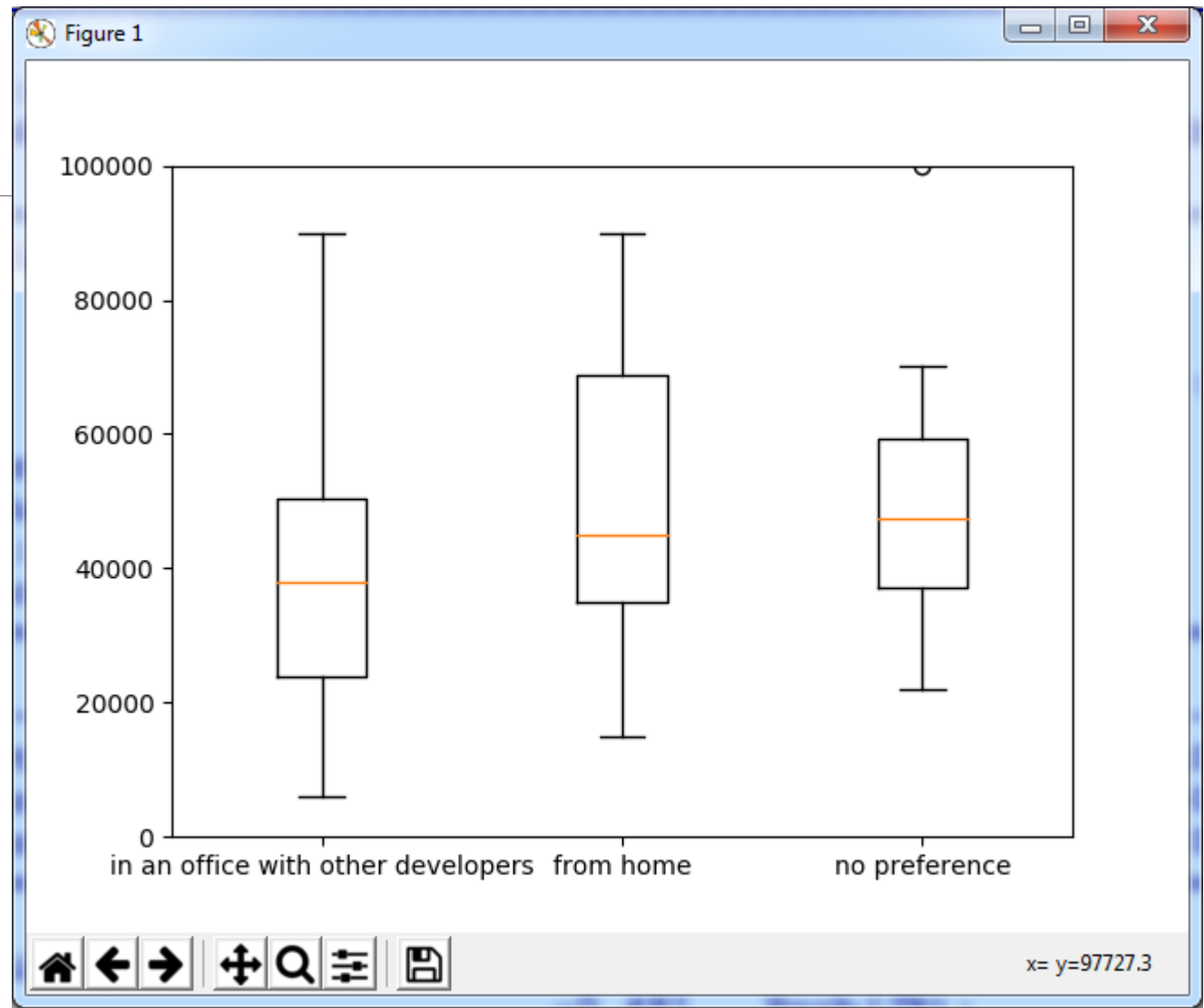
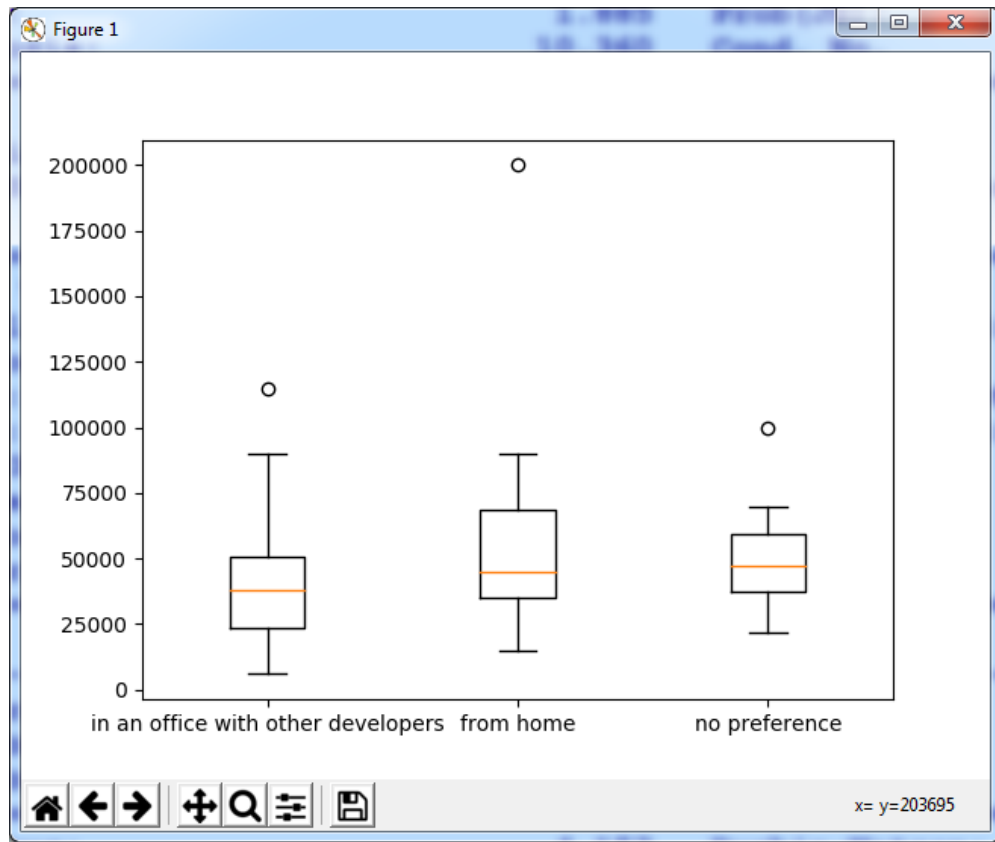


The image shows a screenshot of a Python IDE window titled "example4runner.py - C:\Users\User\Desktop\Students\STDBPY\Lection5\example4runner.py (3.7.2)". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the following Python code:

```
plt.ylim(0, 100000)
plt.boxplot((table2_1, table2_2, table2_3),
            labels = ['in an office with other developers', 'from home', 'no pre
plt.show()
```

The status bar at the bottom right indicates "Ln: 58 Col: 0".

# Результат



# Многофакторный анализ

---

Проведём исследование зависимости дохода (Income) от нескольких (двух) параметров – пола (Gender) и предпочтительного места работы (JobWherePref).

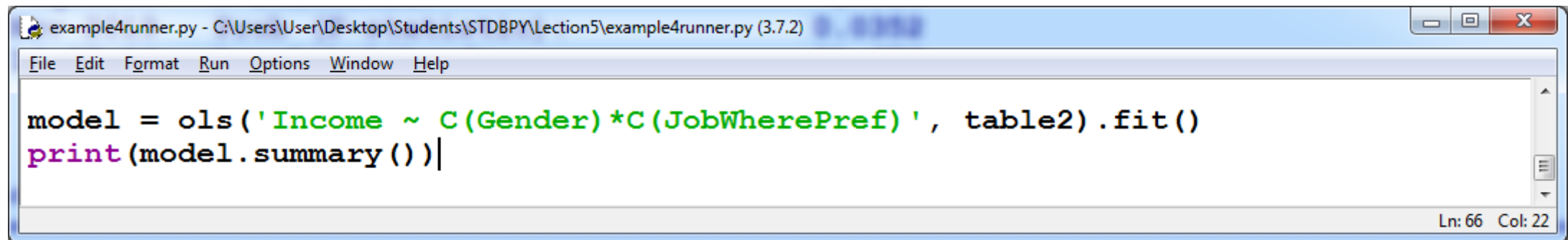
С помощью многофакторного дисперсионного анализа можно проверить оказывает ли взаимодействие параметров эффект на зависимую переменную. Для этого в модели используется символ \*.

Или можно оценить независимый эффект от факторов. Для этого в модели указывается +.

Сначала оценим, есть ли эффект от взаимодействия факторов.

# Запускаем

---



The image shows a screenshot of a Python IDE window titled "example4runner.py - C:\Users\User\Desktop\Students\STDBPY\Lection5\example4runner.py (3.7.2)". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the following Python code:

```
model = ols('Income ~ C(Gender)*C(JobWherePref)', table2).fit()  
print(model.summary())|
```

The status bar at the bottom right of the window indicates "Ln: 66 Col: 22".

# Проверка выполнения условий

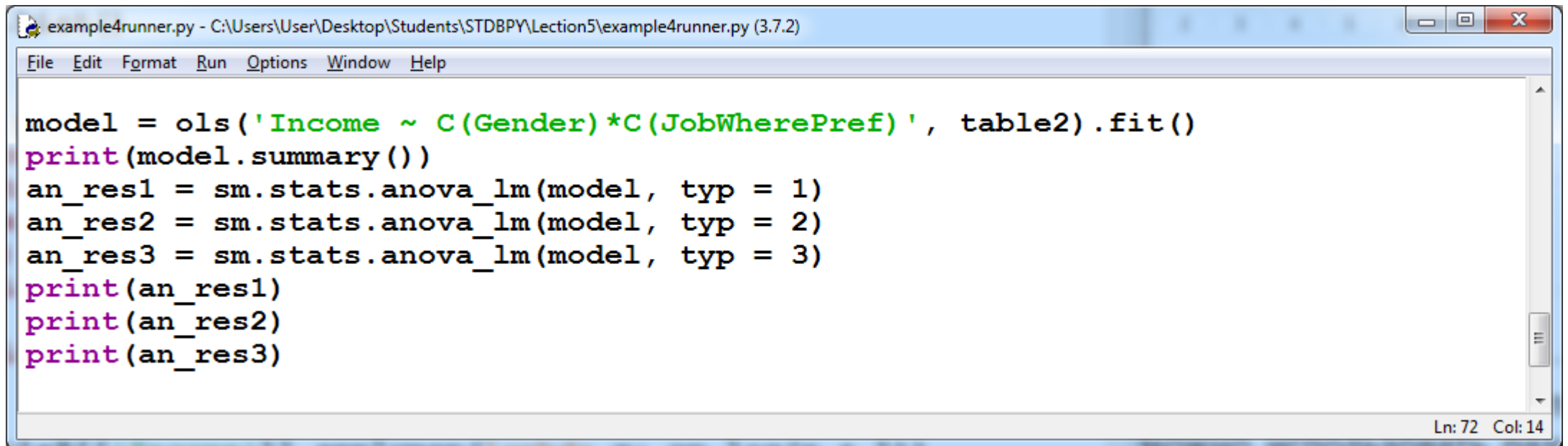
---

```
=====
Omnibus:                3.258    Durbin-Watson:            2.147
Prob(Omnibus) :         0.196    Jarque-Bera (JB) :       3.022
Skew:                  -0.426    Prob(JB) :               0.221
Kurtosis:              2.969    Cond. No.                19.4
=====
```

Все критерии выполнены.

# Таблица ANOVA

Для получения оценки значимости влияния взаимодействия факторов построим таблицу ANOVA. Стоит отметить, что при её построении можно использовать разные типы суммы квадратов. По умолчанию R и Python используют тип I, SAS – тип III. Мы рассмотрим все три варианта.

A screenshot of a text editor window titled 'example4runner.py - C:\Users\User\Desktop\Students\STDBPY\Lec5\example4runner.py (3.7.2)'. The window contains a Python script for fitting an OLS model and calculating ANOVA. The code is as follows:

```
model = ols('Income ~ C(Gender)*C(JobWherePref)', table2).fit()
print(model.summary())
an_res1 = sm.stats.anova_lm(model, typ = 1)
an_res2 = sm.stats.anova_lm(model, typ = 2)
an_res3 = sm.stats.anova_lm(model, typ = 3)
print(an_res1)
print(an_res2)
print(an_res3)
```

The status bar at the bottom right indicates 'Ln: 72 Col: 14'.

# Результат

```

- -
              df      sum_sq    mean_sq      F      PR(>F)
C (Gender)      1.0      0.421598    0.421598    1.081911    0.300938
C (JobWherePref) 2.0      2.556482    1.278241    3.280240    0.041969
C (Gender) : C (JobWherePref) 2.0      1.913108    0.956554    2.454721    0.091377
Residual      94.0     36.629841    0.389679      NaN      NaN

              sum_sq      df      F      PR(>F)
C (Gender)      0.216379      1.0    0.555274    0.458030
C (JobWherePref) 2.556482      2.0    3.280240    0.041969
C (Gender) : C (JobWherePref) 1.913108      2.0    2.454721    0.091377
Residual     36.629841     94.0      NaN      NaN

              sum_sq      df      F      PR(>F)
Intercept     536.740286      1.0   1377.390274    6.054065e-58
C (Gender)      0.810123      1.0      2.078948    1.526666e-01
C (JobWherePref) 0.805611      2.0      1.033685    3.597009e-01
C (Gender) : C (JobWherePref) 1.913108      2.0      2.454721    9.137652e-02
Residual     36.629841     94.0      NaN      NaN
OLS Regression Results

```



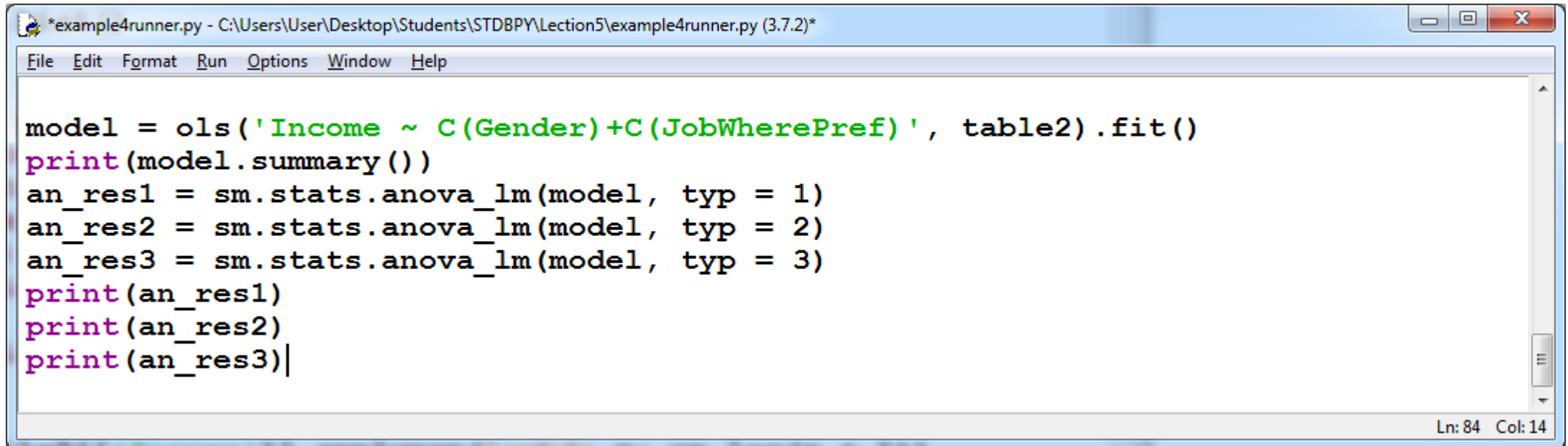
# Интерпретация результата

---

Если взять в качестве достаточного 5% уровень значимости (0,05), то для всех трёх типов суммы квадратов можно сказать, что полученное для взаимодействия параметров  $p\text{-value} = 0,0914 > 0,05$ . Из это следует, что вероятность гипотезы  $H_0$  при таких данных не мала и можно говорить об отсутствии эффекта взаимодействия факторов.

# Новая таблица ANOVA

Для анализа независимого эффекта от факторов перезапустим модель, заменив \* на +.

A screenshot of a Python IDE window titled '\*example4runner.py - C:\Users\User\Desktop\Students\STDBPY\Lec5\example4runner.py (3.7.2)\*'. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The code editor contains the following Python code:

```
model = ols('Income ~ C(Gender)+C(JobWherePref)', table2).fit()
print(model.summary())
an_res1 = sm.stats.anova_lm(model, typ = 1)
an_res2 = sm.stats.anova_lm(model, typ = 2)
an_res3 = sm.stats.anova_lm(model, typ = 3)
print(an_res1)
print(an_res2)
print(an_res3)|
```

The status bar at the bottom right indicates 'Ln: 84 Col: 14'.

# Проверка выполнения условий

---

```
=====
Omnibus:                3.956    Durbin-Watson:                2.101
Prob(Omnibus) :         0.138    Jarque-Bera (JB) :          3.691
Skew:                  -0.471    Prob(JB) :                 0.158
Kurtosis:              3.000    Cond. No.                   6.04
=====
```

Все критерии вновь выполнены.

# Результат

---

```
              df      sum_sq  mean_sq      F      PR(>F)
C (Gender)      1.0    0.421598  0.421598  1.050087  0.308062
C (JobWherePref) 2.0    2.556482  1.278241  3.183751  0.045839
Residual      96.0   38.542949  0.401489      NaN      NaN
              sum_sq      df      F      PR(>F)
C (Gender)      0.216379      1.0  0.538940  0.464662
C (JobWherePref) 2.556482      2.0  3.183751  0.045839
Residual      38.542949     96.0      NaN      NaN
              sum_sq      df      F      PR(>F)
Intercept     1403.549480      1.0  3495.859934  2.565210e-77
C (Gender)      0.216379      1.0    0.538940  4.646616e-01
C (JobWherePref) 2.556482      2.0    3.183751  4.583933e-02
Residual      38.542949     96.0      NaN      NaN
>>> |
```

# Интерпретация результата

---

Если взять в качестве достаточного 5% уровень значимости (0,05), то для всех трёх типов суммы квадратов можно сказать, что p-value для пола  $> 0,05$ , а p-value для предпочтений в работе  $< 0,05$ .

Упрощая формулировки, можно сказать, что от пола эффекта нет, а вот от предпочтений в работе есть!

# Критерий Тьюки

---

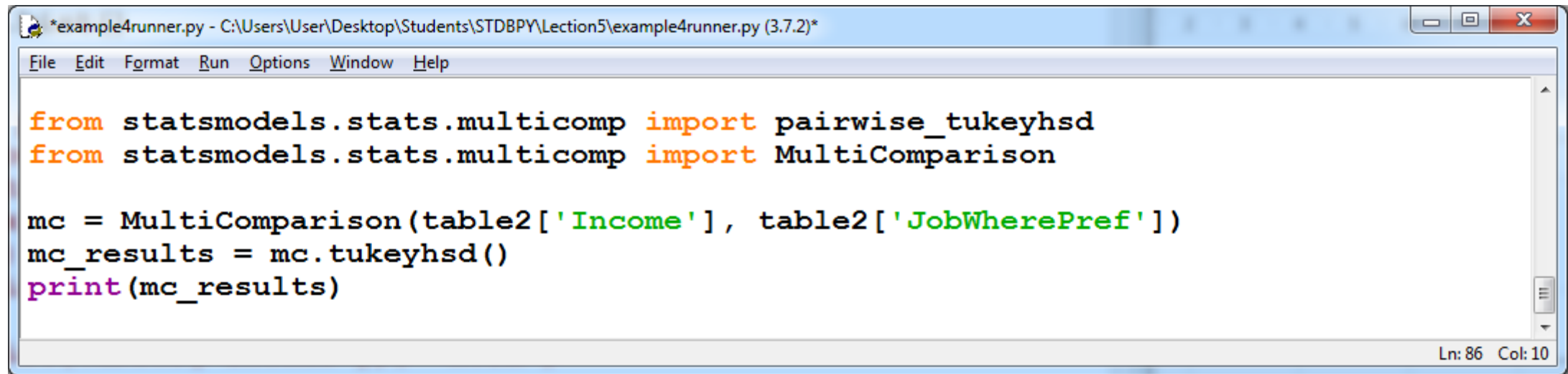
Если при однофакторном дисперсионном анализе было выявлено, что гипотеза  $H_0$  не выполняется, то становится интересно узнать, какие группы приводят к такому результату. Сам однофакторный анализ нужного ответа не даёт. Решить данную проблему можно с помощью Критерия Тьюки (Tukey's honestly significant difference test).

Критерий Тьюки проверяет гипотезу о равенстве средних значений попарно для каждой двух групп уровней фактора.

Вернёмся, для примера, к однофакторному анализу.

# Запуск критерия Тьюки

---



The image shows a screenshot of a Python IDE window titled '\*example4runner.py - C:\Users\User\Desktop\Students\STDBPY\Lec5\example4runner.py (3.7.2)\*'. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The code editor contains the following Python code:

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

mc = MultiComparison(table2['Income'], table2['JobWherePref'])
mc_results = mc.tukeyhsd()
print(mc_results)
```

The status bar at the bottom right indicates 'Ln: 86 Col: 10'.

# Результат

---

Multiple Comparison of Means - Tukey HSD, FWER=0.05

group1	group2	meandiff	p-adj	lower	upper	reject
from home	in an office with other developers	-0.3217	0.0864	-0.6788	0.0354	False
from home	no preference	0.0305	0.9	-0.4309	0.4919	False
in an office with other developers	no preference	0.3522	0.1045	-0.0555	0.7599	False



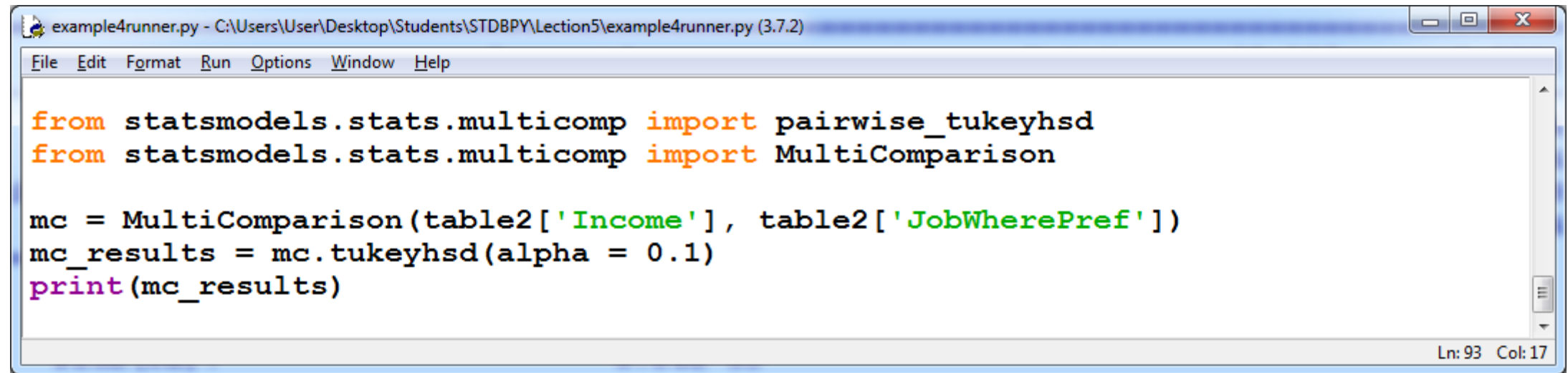
# Интерпретация результата

---

Поле reject говорит нам, надо ли отвергать гипотезу  $H_0$ . Как ни странно, мы не смогли выявить, где находится различие. Но если повысить требуемый уровень значимости, то всё станет ясно.

# ИТОГ

---



The image shows a screenshot of a Python IDE window titled "example4runner.py - C:\Users\User\Desktop\Students\STDBPY\Lec5\example4runner.py (3.7.2)". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the following Python code:

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

mc = MultiComparison(table2['Income'], table2['JobWherePref'])
mc_results = mc.tukeyhsd(alpha = 0.1)
print(mc_results)
```

The status bar at the bottom right of the window indicates "Ln: 93 Col: 17".

# Итог

---

group1	group2	meandiff	p-adj	lower	upper	reject
from home	in an office with other developers	-0.3217	0.0864	-0.6333	-0.0101	True
from home	no preference	0.0305	0.9	-0.3721	0.4331	False
in an office with other developers	no preference	0.3522	0.1045	-0.0035	0.7079	False

# Интернет ресурсы и литература

---

1. <https://www.kaggle.com/freecodecamp/2016-new-coder-survey-/version/1>
2. <https://pythonfordatascience.org/anova-python/>
3. <https://pythonfordatascience.org/anova-2-way-n-way/>
4. <http://www.statisticalassociates.com/assumptions.pdf>