



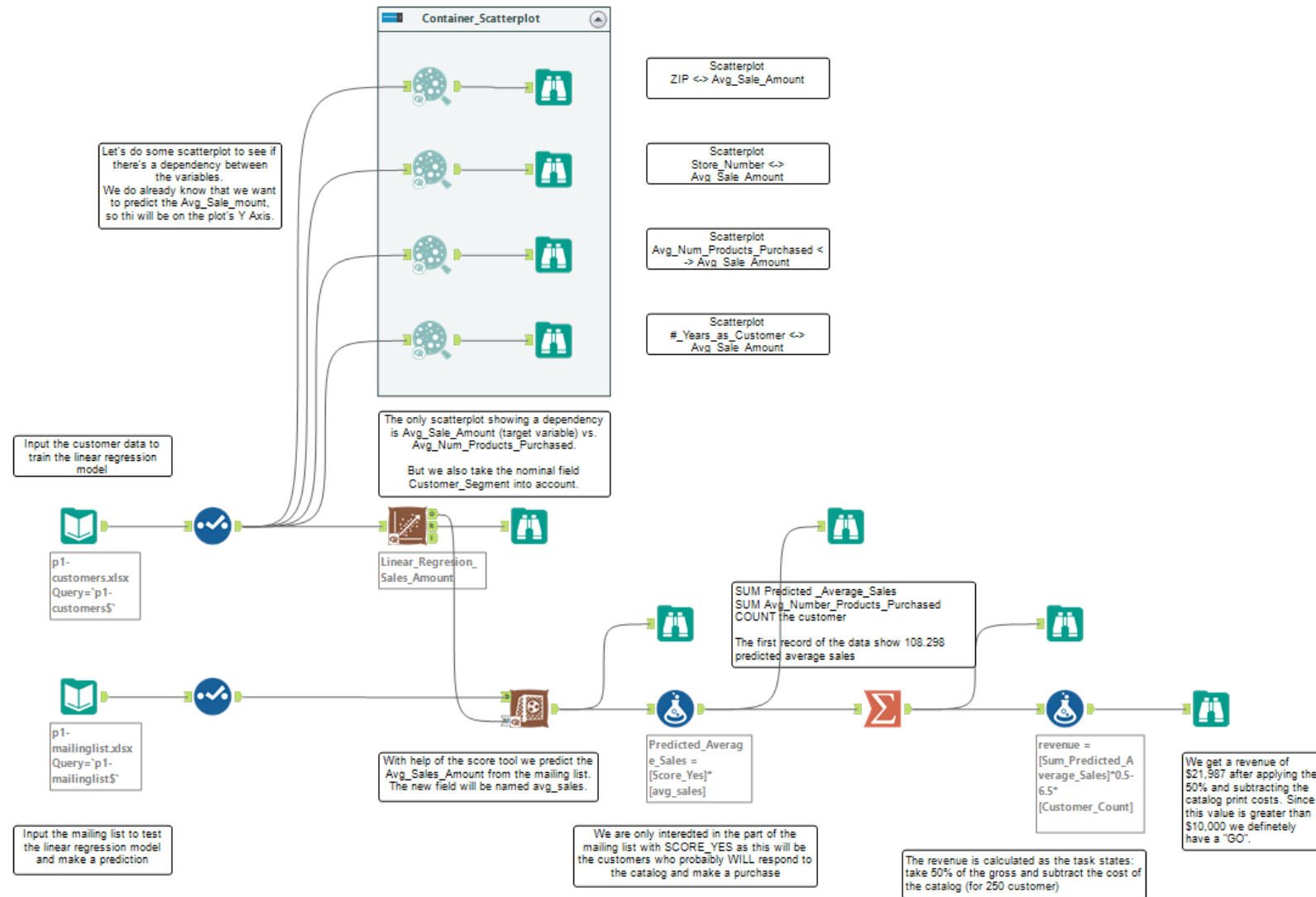
# Workflow-Comparison

## Alteryx, Knime & Python

by @Jens Gaulke

using the example of lesson 5

# The Alteryx Workflow of Lesson 5



# The input tools



a



Input Data (1) - Configuration

Connect a File or Database

C:\Users\jens\Lesson-5\p1-customers.xlsx

Options

Name	Value
1 Record Limit	
2 File Format	Microsoft Excel (*.xlsx)

Select Excel-File.

Preview (first 100 records)

Name	Customer_Segment	Customer_ID
1 Pamela Wright	Store Mailing List	2
2 Danell Valdez	Store Mailing List	7
3 Jessica Rinehart	Store Mailing List	8
4 Nancy Clark	Store Mailing List	9
5 Andrea Brun	Store Mailing List	10
6 Denise Pentico	Store Mailing List	11
7 Erna Arustamyan	Store Mailing List	12
8 Karen Osborne	Store Mailing List	16
9 Shirley Krywonis	Store Mailing List	17
10 Dianne Vangilder	Store Mailing List	19
11 Mary Kiniry	Store Mailing List	26
12 Belinda Deherrera	Store Mailing List	28
13 Keli Scott	Store Mailing List	31
14 Cynthia Wyat	Store Mailing List	36
15 Ann Ballant	Store Mailing List	41



## Excel Reader

Right click on tool. Choose „Configure“.

customers.xl

Configure... Execute F7  
Execute and Open Views Shift+F10  
Cancel F9  
Reset F8  
Edit Node Description... Alt+F2

Dialog - 3:14 - Excel Reader (customers.xlsx)

Settings Transformation Advanced Settings Flow Variables Memory Policy

Input location

Read from Local File System

Mode ☒ File ☐ Files in folder

File C:\Users\jens\Lesson-5\p1-customers.xlsx

Sheet selection

☒ Select first sheet with data (p1-customers)

☐ Select sheet with name p1-customers

☐ Select sheet at index 0 (Sheet indexes start with 0.)

Column header

☒ Table contains column names in row number 1 (Row numbers start with 1. See "File Content" tab to identify row numbers.)

Row ID

☒ Generate row IDs ☐ Table contains row IDs in column A

Sheet area

☒ Read entire data of the sheet ☐ Read only data in columns from A to and rows from 1 to (See "File Content" tab to identify columns and rows.)

Preview File Content

The suggested column types are based on the first 50 rows only. See "Advanced Settings" tab.

Row ID	Name	Customer...	Customer...	Address	City	State	ZIP	Avg.Sa...	Store...	Respon...	Avg.Nu...	#
Row0	Pamela Wright	Store Mailing List	2	376 S Jasmine St	Denver	CO	80224	227.90	100	No	1	6
Row1	Danell Valdez	Store Mailing List	7	12066 E Lake Cir	Greenwood Village	CO	80111	55.00	105	Yes	1	6
Row2	Jessica Rinehart	Store Mailing List	8	7225 S Gaylord St	Centennial	CO	80122	212.57	101	No	1	5
Row3	Nancy Clark	Store Mailing List	9	4497 Cornish Way	Denver	CO	80239	195.31	105	Yes	1	6
Row4	Andrea Brun	Store Mailing List	10	2316 E 5th Ave	Denver	CO	80206	110.55	100	Yes	1	2
Row5	Denise Pentico	Store Mailing List	11	3883 Quinlan St	Denver	CO	80212	149.01	106	No	1	8
Row6	Erna Arustamyan	Store Mailing List	12	1865 Vinton St	Lakewood	CO	80214	49.37	108	No	1	7
Row7	Karen Osborne	Store Mailing List	16	5400 Sheridan Ave	Arvada	CO	80002	153.97	103	No	3	1
Row8	Shirley Krywonis	Store Mailing List	17	195 Jade St	Broomfield	CO	80020	171.15	107	No	2	2
Row9	Dianne Vangilder	Store Mailing List	19	22872 E Clifton Pl	Aurora	CO	80016	103.24	102	No	1	6
Row10	Mary Kiniry	Store Mailing List	26	16087 E Lehigh	Aurora	CO	80013	245.16	104	No	4	3
Row11	Belinda Deherrera	Store Mailing List	28	14303 E Napa Pl	Aurora	CO	80014	85.02	102	No	2	1
Row12	Keli Scott	Store Mailing List	31	1653 Jola St	Aurora	CO	80010	14.53	100	Yes	1	1
Row13	Cynthia Wyat	Store Mailing List	36	200 Rampart	Denver	CO	80230	180.29	105	No	2	6
Row14	Ann Ballant	Store Mailing List	41	8192 W 81st Dr	Arvada	CO	80005	209.3	103	Yes	1	5
Row15	J Ritz	Store Mailing List	55	3445 S Downing	Englewood	CO	80113	273.15	101	No	4	6
Row16	Jesselyn Brad	Store Mailing List	57	10 S Sherman St	Denver	CO	80209	197.43	100	No	1	2
Row17	Amy Haller	Store Mailing List	60	9214 E Floyd Ave	Denver	CO	80231	210.63	105	No	3	1
Row18	Joan Delisa	Credit Card Only	3287	1657 S King St	Denver	CO	80219	818.72	101	No	5	5
Row19	Helen Cordner	Credit Card Only	3299	2102 S Lansing Ct	Aurora	CO	80014	564.93	105	No	6	6
Row20	Angela Finley	Credit Card Only	3303	1068 S Jasper St	Aurora	CO	80017	605.07	105	No	6	6
Row21	Christine Sullivan	Credit Card Only	3304	7901 W 52nd Ave	Arvada	CO	80002	656.79	107	No	7	7
Row22	Elissa Engledow	Credit Card Only	3315	9360 E Center Ave	Denver	CO	80247	167.59	104	No	1	1



In [21]:

```
# a reference to the pandas library
import pandas as pd

# To visualize the data we need matplotlib
# and seaborn for nice pairplots
import matplotlib.pyplot as plt
import seaborn as sns

# statsmodel will do the linear regression for us
import statsmodels.api as sm

# second library we can use for linear regression
from sklearn import linear_model
from sklearn.linear_model import LinearRegression

# the excel file must be in the same directory as this notebook
# be sure to use the right excel data file.
# This one is the customer excel file for building the model
catalog_customers_file= 'p1-customers.xlsx'

# via pandas, the contents are read into a variable or data frame named catalog_customers_data
# pandas is able not to only read excel, but does a great job on csv, too.
catalog_customers_data = pd.read_excel(catalog_customers_file)

# now I know I can show the data by typing the variable name
catalog_customers_data
```

Using pandas for file read.

Reading the file.

Out[21]:

	Name	Customer_Segment	Customer_ID	Address	City	State	ZIP	Avg_Sale_Amount	Store_Number	Responded_to_Last_Catalog	Avg
0	Pamela Wright	Store Mailing List	2	376 S Jasmine St	Denver	CO	80224	227.90	100	No	1
1	Danell Valdez	Store Mailing List	7	12066 E Lake Cir	Greenwood Village	CO	80111	55.00	105	Yes	1
2	Jessica Rinehart	Store Mailing List	8	7225 S Gaylord St	Centennial	CO	80122	212.57	101	No	1
3	Nancy Clark	Store Mailing List	9	4497 Cornish Way	Denver	CO	80239	195.31	105	Yes	1
4	Andrea Brun	Store Mailing List	10	2316 E 5th Ave	Denver	CO	80206	110.55	100	Yes	1
...	...	...	...	...	...	...	...	...	...	...	...
2370	Joan Delisa	Credit Card Only	3287	1657 S King St	Denver	CO	80219	818.72	101	No	5
2371	Helen Cordner	Credit Card Only	3299	2102 S Lansing Ct	Aurora	CO	80014	564.93	105	No	6
2372	Angela Finley	Credit Card Only	3303	1068 S Jasper St	Aurora	CO	80017	605.07	105	No	6
2373	Christine Sullivan	Credit Card Only	3304	7901 W 52nd Ave	Arvada	CO	80002	656.79	107	No	7
2374	Elissa Engledow	Credit Card Only	3315	9360 E Center Ave	Denver	CO	80247	167.59	104	No	1

# The select tools



a



p1-  
customers.xlsx  
Query = `p1-  
customers`

Select (3) - Configuration

TIP: To reorder multiple rows: select, n

Field	Type	Size
<input checked="" type="checkbox"/> Name	V_String	255
<input checked="" type="checkbox"/> Customer_Segment	V_String	255
<input checked="" type="checkbox"/> Customer_ID	Double	8
<input checked="" type="checkbox"/> Address	V_String	255
<input checked="" type="checkbox"/> City	V_String	255
<input checked="" type="checkbox"/> State	V_String	255
<input checked="" type="checkbox"/> ZIP	Double	8
<input checked="" type="checkbox"/> Avg_Sale_Amount	Double	8
<input checked="" type="checkbox"/> Store_Number	Double	8
<input checked="" type="checkbox"/> Responded_to_Last_Catalog	V_String	255
<input checked="" type="checkbox"/> Avg_Num_Products_Purchased	Double	8
<input checked="" type="checkbox"/> #_Years_as_Customer	Double	8
<input checked="" type="checkbox"/> Unknown	Unknown	0

(De-)Select your  
columns.



## Excel Reader



Right click on tool.  
Choose „Configure“.

customers.xl

Select tool is  
integrated in  
Configure dialog.

See Tab  
„Transformation“

Dialog - 3:14 - Excel Reader (customers.xlsx)

File

Settings (Transformation) Advanced Settings Flow Variables Memory Policy

Reset actions

Column	New name
<input checked="" type="checkbox"/> Name	
<input checked="" type="checkbox"/> Customer_Segment	
<input checked="" type="checkbox"/> Customer_ID	
<input checked="" type="checkbox"/> Address	
<input checked="" type="checkbox"/> City	
<input checked="" type="checkbox"/> State	
<input checked="" type="checkbox"/> ZIP	
<input checked="" type="checkbox"/> Avg_Sale_Amount	
<input checked="" type="checkbox"/> Store_Number	
<input checked="" type="checkbox"/> Responded_to_Last_Catalog	
<input checked="" type="checkbox"/> Avg_Num_Products_Purchased	
<input checked="" type="checkbox"/> #_Years_as_Customer	
<input checked="" type="checkbox"/> <any unknown new column>	

Preview File Content

Preview with current settings

The suggested column types are based on the first 50 rows only. See 'Advanced Settings' tab.

Row ID	S Name	S Customer...	I Custom...	S Address	S City	S State	I ZIP	D Avg_Sa...	I Store...	S Respon...	I Avg_N...	I #
Row0	Pamela Wright	Store Mailing List	2	376 S. Jeanne St.	Denver	CO	80224	227.9	100	No	1	6
Row1	Daniel Valdez	Store Mailing List	7	12966 E Lake Cir	Greenwood	CO	80111	95	105	Yes	1	6
Row2	Jessica Rine...	Store Mailing List	8	7225 S Gaylord...	Centennial	CO	80122	212.57	101	No	1	3
Row3	Nancy Clark	Store Mailing List	9	4497 Cornish ...	Denver	CO	80239	195.31	105	Yes	1	6
Row4	Andrew Brun	Store Mailing List	10	2318 E 5th Ave	Denver	CO	80206	110.55	100	Yes	1	2
Row5	Danise Pentico	Store Mailing List	11	3883 Quitman St	Denver	CO	80212	149.01	106	No	1	8
Row6	Erna Arusta...	Store Mailing List	12	1985 Yukon St	Lakewood	CO	80214	49.37	108	No	1	7
Row7	Karen Osborne	Store Mailing List	16	5403 Sheridan ...	Arvada	CO	80002	153.97	103	No	3	1
Row8	Shirley Kryn...	Store Mailing List	17	193 Jade St	Broomfield	CO	80020	173.15	107	No	2	2
Row9	Dianne Yang	Store Mailing List	19	22873 E Clifton Pl	Aurora	CO	80016	105.24	102	No	1	6
Row10	Mary Kinyr	Store Mailing List	26	16087 E Lehigh...	Aurora	CO	80013	245.16	104	No	4	3
Row11	Reinhold Del...	Store Mailing List	28	14303 S Napa Pl	Aurora	CO	80014	85.02	102	No	2	1
Row12	Kell Scott	Store Mailing List	31	1653 Iola St	Aurora	CO	80010	14.53	100	Yes	1	1
Row13	Cynthia Whit	Store Mailing List	36	200 Rampart ...	Denver	CO	80230	190.29	105	No	2	6
Row14	Ann Ballett	Store Mailing List	40	8192 W 81st Dr	Arvada	CO	80005	209.3	103	Yes	1	5

OK Apply Cancel ?



```
In [73]: # We drop the columns without statistical significance from our dataset
cleaned_customers_data = catalog_customers_data.drop(['Customer_ID', 'ZIP', 'Store_Number', '#_Years_as_Customer'], axis=1)
# We also can drop the nominal data without significance as described in analysis section
cleaned_customers_data = cleaned_customers_data.drop(['Name', 'Address', 'City', 'State', 'Responded_to_Last_Catalog'], axis=1)

# We still have nominal data in the dataset, namely Customer_Segment
# As this column is relevant for us, we have to create dummy variables
# and we have to drop "Credit card only" as stated in the task description
cleaned_customers_data = pd.get_dummies(cleaned_customers_data, columns=['Customer_Segment'], drop_first=True)

# set the target variable
Y = cleaned_customers_data['Avg_Sale_Amount']
# set the predictor variables
X = cleaned_customers_data.drop(['Avg_Sale_Amount'], axis=1)

# Let's to the evaluation with statsmodels
# we have to add a constant to the calculation or
# we do not have a y-intercept
X = sm.add_constant(X)

# build the model
model = sm.OLS(Y, X).fit()
model_prediction = model.predict(X)
model_details = model.summary()

# print the details, so we can compare to ateryx
model_details

X
```

2) ... and assign the  
remaining columns  
to a new dataframe

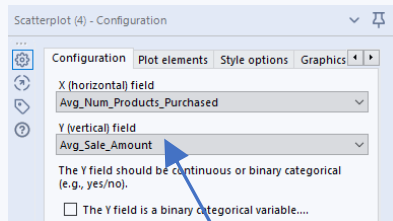
1) Drop columns  
from pandas  
dataframe...

3) Expand dataframe  
by simply adding new  
columns from  
another dataset

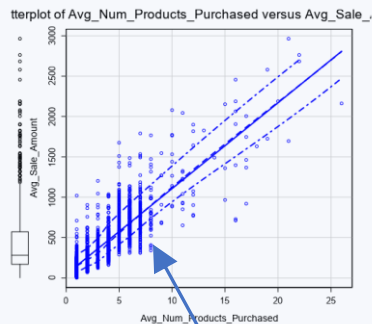
# The scatterplot tools



a



Set X & Y axis



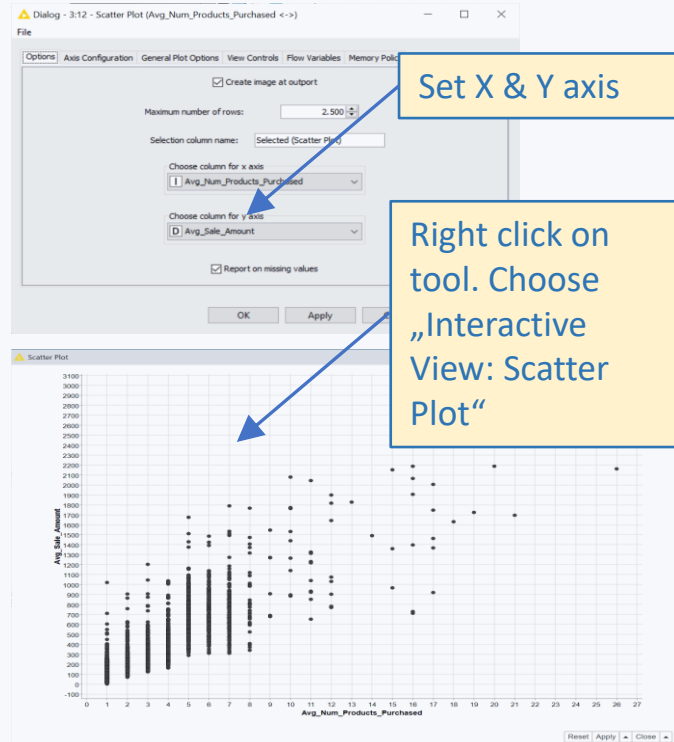
Use browse tool



## Scatter Plot

Avg\_Num\_Products\_Purchased <->  
Avg\_Sale\_Amount

Right click on tool.  
Choose „Configure“.



Set X & Y axis

Right click on  
tool. Choose  
„Interactive  
View: Scatter  
Plot“



In [7]: 

```
# with a seaborn pairplot, we can check for linear relationship
# in each and every combination of the data columns.
# as we can see, the only relationship seems to be between
# Avg_Sales_Amount and Avg_Num_Products_Purchased

sns.pairplot(catalog_customers_data)
plt.show()
```

Seaborn's pairplot  
plots each and every  
combination of  
variables at once.



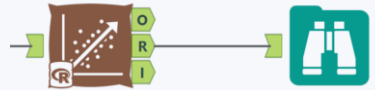
Here is the  
desired plot



# The linear regression tools



a



Model name: Linear\_Regression\_6

Select the target variable: Avg\_Sale\_Amount

Select the predictor variables:

Selected: 2 Fields: 11

Show: All Selected

Exclude:

- ☒ Name
- ☒ Customer\_Segment
- ☒ Customer\_ID
- ☒ Address
- ☒ City
- ☒ State
- ☒ ZIP
- ☒ Store\_Number
- ☒ Responded\_to\_Last\_Catalog
- ☒ Avg\_Num\_Products\_Purchased

Set predictors and target

Report

1 of 1 Fields: 11

Basic Summary

Call: lm(formula = Avg\_Sale\_Amount ~ Customer\_Segment + Avg\_Num\_Products\_Purchased, data = the\_data)

Residuals:

	Min	1Q	Median	3Q	Max
	-663.6	-67.3	-1.9	70.7	971.7

Coefficients:

	Estimate	Std. Error	t Pr(> t )
(Intercept)	303.46	10.576	28.69 < 2.2e-***
Customer_SegmentLoyalty Club Only	-149.36	8.973	-16.65 < 2.2e-***
Customer_SegmentLoyalty Club and Credit Card	281.84	11.910	23.66 < 2.2e-***
Customer_SegmentStore Mailing List	-245.42	9.768	-25.13 < 2.2e-***
Avg_Num_Products_Purchased	66.98	1.515	44.21 < 2.2e-***

Significance codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 137.46 on 2370 degrees of freedom

Multiple R-squared: 0.8369, Adjusted R-Squared: 0.8366

F-statistic: 3040 on 4 and 2370 degrees of freedom (DF), p-value < 2.2e-16

Type II ANOVA Analysis

Response: Avg\_Sale\_Amount

	Sum of Squares	DF	F-value	Pr(>F)
Customer_Segment	14936.0	3	16.65	< 2.2e-***
Avg_Num_Products_Purchased	6698.0	1	44.21	< 2.2e-***
Residuals	13746.0	2370		

Use browse tool



## Linear Regression Learner



Right click on tool. Choose „Configure“.

Dialog - 3/2 - Linear Regression Learner (Linear Regression)

Target: Avg\_Sale\_Amount

Exclude:

- ☒ Name
- ☒ Customer\_ID
- ☒ Address
- ☒ City
- ☒ State
- ☒ ZIP
- ☒ Store\_Number
- ☒ Responded\_to\_Last\_Catalog
- ☒ #\_Years\_as\_Customer

Include:

- ☒ Customer\_Segment
- ☒ Avg\_Num\_Products\_Purchased

Regression Properties

Missing Values in Input Data

Scatter Plot View

OK Apply Cancel

Set predictors and target

Right click on tool. Choose „View: Linear regression result view“

Linear Regression Result View - 3/2 - Linear Regression Learner

Statistics on Linear Regression

Variable	Coeff.	Std. Err.	t-value	P> t
Customer_Segment=Loyalty Club Only	-149.3557	8.9728	-16.6455	0.0
Customer_Segment=Loyalty Club and Credit Card	281.8388	11.9099	23.6643	0.0
Customer_Segment=Store Mailing List	-245.4177	9.7678	-25.1252	0.0
Avg_Num_Products_Purchased	66.9762	1.515	44.2075	0.0
Intercept	303.4635	10.5757	28.6944	0.0

Multiple R-Squared: 0.8369

Adjusted R-Squared: 0.8366

b

```
In [7]: # We drop the columns without statistical significance from our dataset
cleaned_customers_data = catalog_customers_data.drop(['Customer_ID', 'ZIP', 'Store_Number', '#_Years_as_Customer'], axis=1)
# We also can drop the nominal data without significance as described in analysis section
cleaned_customers_data = cleaned_customers_data.drop(['Name', 'Address', 'City', 'State', 'Responded_to_Last_Catalog'], axis=1)

# we still have nominal data in the dataset, namely Customer_Segment
# As this column is relevant for us, we have to create dummy variables
# and we have to drop "Credit card only" as stated in the task description
cleaned_customers_data = pd.get_dummies(cleaned_customers_data, columns=['Customer_Segment'], drop_first=True)

# set the target variable
Y = cleaned_customers_data['Avg_Sale_Amount']
# set the predictor variables
X = cleaned_customers_data.drop(['Avg_Sale_Amount'], axis=1)

# Let's do the evaluation with statsmodels
# we have to add a constant to the calculation or
# we do not have a Y-intercept
X = sm.add_constant(X)

# build the model
model = sm.OLS(Y, X).fit()
model_prediction = model.predict(X)
model_details = model.summary()

# print the details, so we can compare to alteryx
print(model_details)
```

Out[7]: OLS Regression Results

Dep. Variable:	Avg_Sale_Amount	R-squared:	0.837
Model:	OLS	Adj. R-squared:	0.837
Method:	Least Squares	F-statistic:	3040.
Date:	Mon, 08 Mar 2021	Prob (F-statistic):	0.00
Time:	20:38:21	Log-Likelihood:	-15061.
No. Observations:	2375	AIC:	3.013e+04
Df Residuals:	2370	BIC:	3.016e+04
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
const	303.4635	10.576	28.694	0.000	282.725	324.202
Avg_Num_Products_Purchased	66.9762	1.515	44.208	0.000	64.005	69.947
Customer_Segment_Loyalty Club Only	-149.3557	8.973	-16.645	0.000	-166.951	-131.760
Customer_Segment_Loyalty Club and Credit Card	281.8388	11.910	23.664	0.000	258.484	305.194
Customer_Segment_Store Mailing List	-245.4177	9.768	-25.125	0.000	-264.572	-226.263

Omnibus: 359.638 Durbin-Watson: 2.045

Prob(Omnibus): 0.000 Jarque-Bera (JB): 4770.580

Skew: 0.232 Prob(JB): 0.00

Kurtosis: 9.928 Cond. No. 25.0

Alteryx and Knime can use categorical data in regression. Python has to create dummy variables.

Use statsmodels ordinary least squares (OLS) regression model

# The score tools



a



Model Type

- Local Model
- Alteryx Promote Model

Configure Local Model Options

The new field name (continuous target) or prefix (categorical target)

Expected Profit

☐ The target field has an oversampled value

Non-regularized linear regression only options

☐ The target field has been natural log transformed

☐ Include a prediction confidence interval

XDF input specific options

☐ Append scores to the input XDF file

The number of records to score at a time

250000

Set new field name.

Remember to de-select unwanted columns in the input data for the score tool.



Regression Predictor

Node 5

Right click on tool. Choose „Configure“.

Dialog - 3:5 - Regression Predictor

File

Settings Flow Variables Memory Policy

Prediction column

☒ Custom prediction column name: Prediction (Avg\_Sale\_A

OK Apply

Set new field name.

Remember to de-select unwanted columns in the input data for the score tool.

Predicted data - 3:5 - Regression Predictor

File Edit Table Navigation View

Table 'Default' - Rows: 250 | Sigs: Columns: 13 | Properties: Flow Variables

Row ID	S Name	S Customer	S Custom...	S Address	S City	S State	S ZIP	S Store...	S Avg_N...	S #_Year...	S Score_M...	S Predict...
Row0	A Gamerts	Loyalty Club Only	2213	3326 S Union Way	Centennial	CO	80015	105	0.3	0.685	0.305	355.036
Row1	Abby Pearson	Loyalty Club an...	2785	4344 W Riverside Pl	Denver	CO	80238	101	0.6	0.527	0.473	987.159
Row2	Adrian Malheur	Loyalty Club Only	2511	5219 S Delaware St	Englewood	CO	80110	101	0.9	0.421	0.579	622.941
Row3	Alexandra B.	Loyalty Club Only	2231	2301 Lawrence St	Denver	CO	80205	103	0.6	0.685	0.305	288.06
Row4	Alisa Orenti	Loyalty Club Only	2530	1549 S Wambol	Centennial	CO	80015	104	0.6	0.612	0.388	422.013
Row5	Amanda O...	Credit Card Only	2946	10955 E Waverne	Denver	CO	80242	104	0.9	0.514	0.486	722.584
Row6	Amanda H...	Loyalty Club an...	1212	3889 Alderbrook	Highlands R...	CO						
Row7	Ange Laffer	Credit Card Only	369	1952 S Buckley	Aurora	CO						
Row8	Arb Tran	Credit Card Only	1683	7328 E Maple Ave	Denver	CO						
Row9	Arnel Crumme	Loyalty Club Only	1940	7345 S Catalina	Aurora	CO						
Row10	April Johnson	Store Selling List	174	6893 S Shafter	Aurora	CO						
Row11	Aut Pedeges	Loyalty Club Only	256	7537 E Warren Cr	Denver	CO						
Row12	B Corina	Loyalty Club Only	1602	2721 S Ingle	Aurora	CO						
Row13	Barbara K...	Credit Card Only	2072	9421 W 95th	Denver	CO						
Row14	Barbara Vogel	Credit Card Only	3004	9457 W Hamilton Dr	Lakewood	CO						
Row15	Becca Bae	Loyalty Club Only	1553	12900 Lake Song	Broomfield	CO						
Row16	Beverly Buc...	Loyalty Club Only	1405	2552 S Clatsie Way	Aurora	CO						
Row17	Beverly Hoyt	Loyalty Club Only	987	3821 Lowell Blvd	Denver	CO						
Row18	Beverly Mancu	Loyalty Club an...	1193	4144 E Hensdale Cr	Centennial	CO						
Row19	Bria Cunnin...	Credit Card Only	259	142 Kenton St	Aurora	CO						
Row20	Brittany Joh...	Loyalty Club Only	2282	9281 W 98th Way	Broomfield	CO						

Right click on tool. Choose „Predicted data“.



```
In [85]: # Now we have a working model, we can use the data from
# p1-mailinglist.xlsx to test the model and to make a prediction
catalog_test_file = "p1-mailinglist.xlsx"
catalog_test_data = pd.read_excel(catalog_test_file)

# see above
cleaned_test_data = pd.get_dummies(catalog_test_data, columns=['Customer_Segment'], drop_first=True)
cleaned_test_data = cleaned_test_data.drop(['Customer_ID', 'ZIP', 'Store_Number', '#_Years_as_Customer'], axis=1)
cleaned_test_data = cleaned_test_data.drop(['Name', 'Address', 'City', 'State'], axis=1)

# even if we know we need Score_Yes and Score_No, these columns are not present in the customer dataset
# so a correlation between the model data and the test data will not work
# alteryx can do this automatically, we have to get our hands dirty
cleaned_test_data1 = cleaned_test_data.drop(['Score_Yes', 'Score_No'], axis=1)

# set the predictor variables
X_test = cleaned_test_data1

# let's do the evaluation with statsmodels
# we have to add a constant to the calculation or
# we do not have a Y-intercept
X_test = sm.add_constant(X_test)

# and make a prediction with the test data
model_prediction_test = model.predict(X_test)

# we have to make a prediction according to the value of Score_Yes
# so let's read the score values again and add them to the model
score = pd.read_excel('p1-mailinglist.xlsx', usecols=['Score_Yes'])

# add the model to the data frame
score['Predicted_Sales'] = model_prediction_test

# show the results
score
```

Out[85]:

	Score_Yes	Predicted_Sales
0	0.305036	355.036364
1	0.472725	987.159466
2	0.578882	622.941184
3	0.305138	288.060159
4	0.387706	422.012569
...	...	...

Do the same to mailinglist what you did to customer data.

Then use the trained model with the mailinglist data

And build a new table with score\_yes and predicted\_sales

# The formula tools



a



Output Column	Data Preview
revenue	21987.4356865455
[Sum_Predicted_Average_Sales]*0.5-6.5* [Customer_Count]	

Type in your calculation

Create new columns



## Math Formula



Right click on tool.  
Choose „Configure“.

Dialog - 3:9 - Math Formula (Pred\_Sales :=)

File

Math Expression Flow Variables Memory Policy

Column List

- ROWINDEX
- ROWCOUNT
- [Customer\_ID (Count\*)]
- [Avg\_Num\_Products\_Purchased (Sum)]
- [Predicted\_Avg\_sales (Sum)]

Category

All

Function

- ROWCOUNT
- ROWINDEX
- pi
- COL\_MIN(col\_name)
- COL\_MAX(col\_name)
- COL\_MEAN(col\_name)
- COL\_MEDIAN(col\_name)
- COL\_SUM(col\_name)
- COL\_STDDEV(col\_name)
- COL\_VAR(col\_name)
- ln(x)

Description

Expression

1 \$Predicted\_Avg\_sales (Sum)\$\*0.5-6.5\*\$Customer\_ID (Count\*)\$

Flow Variable List

Append Column: Pred\_Sales

Replace Column: [Predicted\_Avg\_sales (Sum)]

Convert to Int

OK Apply Cancel ?

Create new columns

Type in your calculation



```
In [108]: # now we do the evaluation

print("Catalog marketing campaign results")
print("-----\n")

total_revenue = sum(score['Predicted_Sales']*score['Score_Yes'])
print("Total expected revenue from the marketing campaign: ", '${:,.2f}'.format(total_revenue))

adjusted_revenue = total_revenue * 0.5
print("Applying gross margin of 50%: ", '${:,.2f}'.format(adjusted_revenue))

number_customers = score.shape[0]
print_costs = 6.5
catalog_costs = print_costs * number_customers
print("Print costs are $6.5 x", number_customers, "customer(s):", '${:,.2f}'.format(catalog_costs))

print("")
expected_profit = adjusted_revenue - catalog_costs
print("The expected profit from the marketing campaign is:", '${:,.2f}'.format(expected_profit))

if (expected_profit > 10000):
    print("Recommendation: GO! We should do the campaign.")
else:
    print("Recommendation: NO GO! We should not do the campaign.")
```

Catalog marketing campaign results

Total expected revenue from the marketing campaign: \$47,224.87  
Applying gross margin of 50%: \$23,612.44  
Print costs are \$6.5 x 250 customer(s): \$1,625.00

The expected profit from the marketing campaign is: \$21,987.44  
Recommendation: GO! We should do the campaign.

Simple python calculations.



# The summarize / aggregation tools



a



Summarize (25) - Configuration

Fields:

Field	Type
Name	V_WString
Customer_Segment	V_WString
Customer_ID	Double
Address	V_WString
City	V_WString
State	V_WString
ZIP	Double
Store_Number	Double
Avg_Num_Products_Purchased	Double
Score_No	Double
Score_Yes	Double
avg_sales	Double
Predicted_Average_Sales	Double

Actions:

Field	Action	Output Field Name
Predicted_Average_Sales	Sum	Sum_Predicted_Average_Sales
Avg_Num_Products_Purchased	Sum	Sum_Avg_Num_Products_Purchased
Customer_ID	Count	Customer_Count

Set aggregation functions for columns



GroupBy



Right click on tool.  
Choose „Configure“.

Dialog - 3:8 - GroupBy (Customer ID - Count)

Settings Description Flow Variables Memory Policy

Groups Manual Aggregation Pattern Based Aggregation Type Based Aggregation

Aggregation settings

Available columns

Select

To change multiple columns use right mouse click for context menu.

Column	Aggregation (click to change)	Missing	Parameter
Customer_ID	Count	<input type="checkbox"/>	
Avg_Num_Products_Purchased	Sum	<input type="checkbox"/>	
Predicted_Avg_sales	Sum	<input type="checkbox"/>	

add >> add all >> << remove << remove all

Advanced settings

Column naming: Column name (aggregation method) ☐ Enable

Maximum unique values per group 10,000 Value delimiter

Choose „Manual Aggregation“.

Set desired aggregation function.



```
In [108]: # now we do the evaluation

print("Catalog marketing campaign results")
print("-----\n")

total_revenue = sum(score['Predicted_Sales']*score['Score_Yes'])
print("Total expected revenue from the marketing campaign: ", '${:,.2f}'.format(total_revenue))

adjusted_revenue = total_revenue * 0.5
print("Applying gross margin of 50%: ", '${:,.2f}'.format(adjusted_revenue))

number_customers = score.shape[0]
print_costs = 6.5
catalog_costs = print_costs * number_customers
print("Print costs are $6.5 x", number_customers, "customer(s):", '${:,.2f}'.format(catalog_costs))

print("")
expected_profit = adjusted_revenue - catalog_costs
print("The expected profit from the marketing campaign is:", '${:,.2f}'.format(expected_profit))

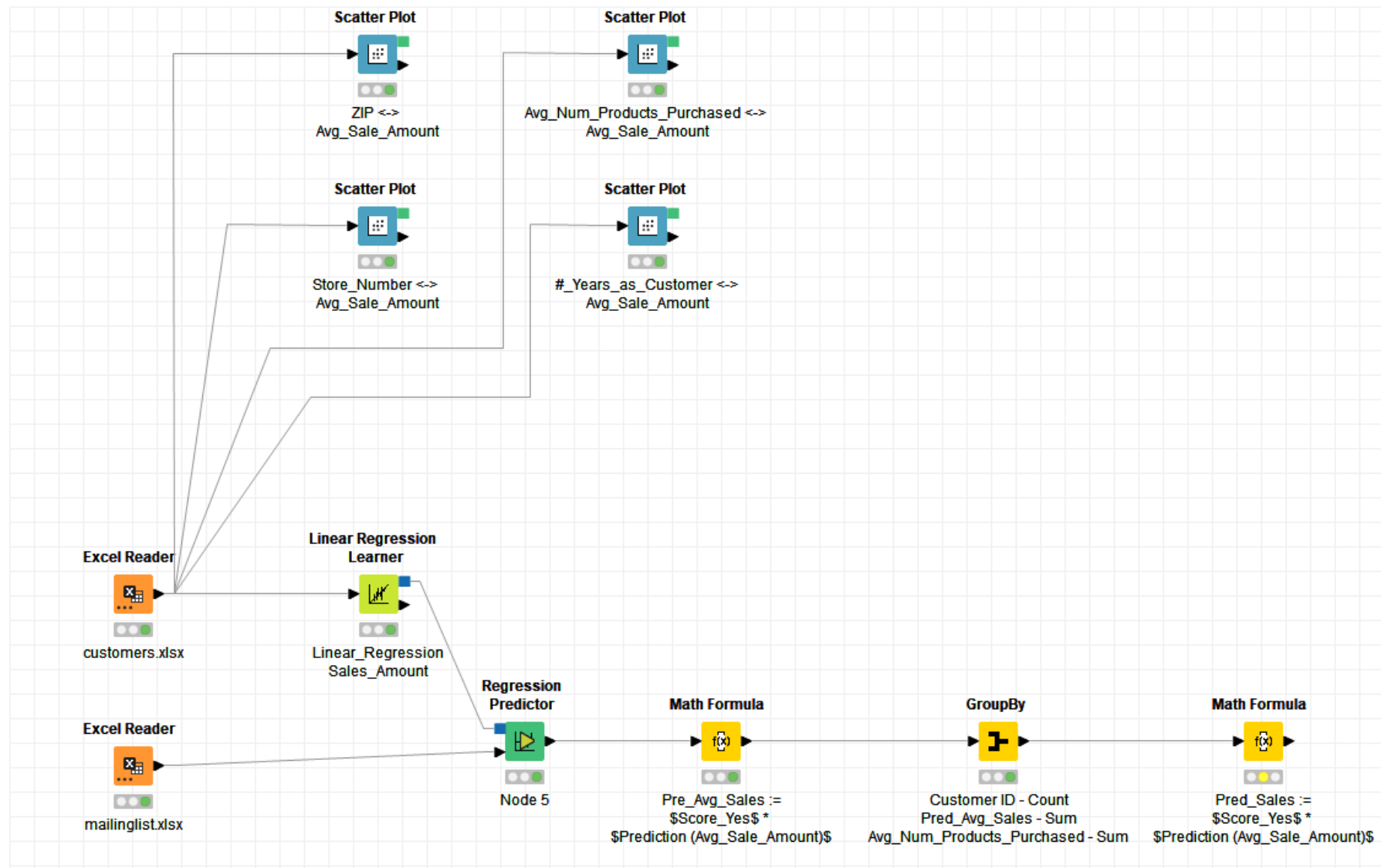
if (expected_profit > 10000):
    print("Recommendation: GO! We should do the campaign.")
else:
    print("Recommendation: NO GO! We should not do the campaign.")

Catalog marketing campaign results
-----
Total expected revenue from the marketing campaign: $47,224.87
Applying gross margin of 50%: $23,612.44
Print costs are $6.5 x 250 customer(s): $1,625.00

The expected profit from the marketing campaign is: $21,987.44
Recommendation: GO! We should do the campaign.
```

Simple python calculations.

# The Knime Workflow of Lesson 5



# The Python Code of Lesson 5 - I



```
# a reference to the pandas library
import pandas as pd

# To visualize the data we need matplotlib
# and seaborn for nice pairplots
import matplotlib.pyplot as plt
import seaborn as sns

# statsmodel will do the linear regression for us
import statsmodels.api as sm

# second library we can use for linear regression
from sklearn import linear_model
from sklearn.linear_model import LinearRegression

# the excel file must be in the same directory as this notebook
# be sure to use the right excel data file.
# This one is the customer excel file for building the model
catalog_customers_file= 'p1-customers.xlsx'

# via pandas, the contents are read into a variable or data frame named catalog_customers_data
# pandas is able not only to read excel, but does a great job on csv, too.
catalog_customers_data = pd.read_excel(catalog_customers_file)

# now I know I can show the data by typing the variable name
catalog_customers_data
```

# The Python Code of Lesson 5 - II



```
# with a seaborn pairplot, we can check for linear relationship
# in each and every combination of the data columns.
# as we can see, the only relationship seems to be between
# Avg_Sales_Amount and Avg_Num_Products_Purchased

sns.pairplot(catalog_customers_data)
plt.show()

# We drop the columns without statistical significance from our dataset
cleaned_customers_data = catalog_customers_data.drop(['Customer_ID', 'ZIP', 'Store_Number', '#_Years_as_Customer'], axis=1)
#we also can drop the nominal data without significance as described in analysis section
cleaned_customers_data = cleaned_customers_data.drop(['Name', 'Address', 'City', 'State', 'Responded_to_Last_Catalog'], axis=1)
)

# we still have nominal data in the dataset, namely Customer_Segment
# As this column is relevant for us, we have to create dummy variables
# and we have to drop "Credit card only" as stated in the task description
cleaned_customers_data = pd.get_dummies(cleaned_customers_data, columns=['Customer_Segment'], drop_first=True)

# set the target variable
Y = cleaned_customers_data['Avg_Sale_Amount']
# set the predictor variables
X = cleaned_customers_data.drop(['Avg_Sale_Amount'], axis=1)
```

# The Python Code of Lesson 5 - III



```
# let's to the evaluation with statsmodels
# we have to add a constant to the calculation or
# we do not have a Y-intercept
X = sm.add_constant(X)

# build the model
model = sm.OLS(Y,X).fit()
model_prediction = model.predict(X)
model_details = model.summary()

# print the details, so we can compare to alteryx
model_details

# Now we have a working model, we can use the data from
# p1-mailinglist.xlsx to test the model and to make a prediction
catalog_test_file = "p1-mailinglist.xlsx"
catalog_test_data = pd.read_excel(catalog_test_file)

# see above
cleaned_test_data = pd.get_dummies(catalog_test_data, columns=['Customer_Segment'], drop_first=True)
cleaned_test_data = cleaned_test_data.drop(['Customer_ID', 'ZIP', 'Store_Number', '#_Years_as_Customer'], axis=1)
cleaned_test_data = cleaned_test_data.drop(['Name', 'Address', 'City', 'State'], axis=1)

# even if we know we need Score_Yes and Score_No. these columns are not present in the customer dataset
# so a correlation between the model data and the test data will not work
# alteryx can do this automatically, we have to get our hands dirty
cleaned_test_data1 = cleaned_test_data.drop(['Score_Yes', 'Score_No'], axis=1)
```

# The Python Code of Lesson 5 - IV



```
# set the predictor variables
X_test = cleaned_test_data1

# let's to the evaluation with statsmodels
# we have to add a constant to the calculation or
# we do not have a Y-intercept
X_test = sm.add_constant(X_test)

# and make a prediction with the test data
model_prediction_test = model.predict(X_test)

# we have to make a prediction according to the value of Score_Yes
# so let's read the score values again and add them to the model
score = pd.read_excel('p1-mailinglist.xlsx', usecols=['Score_Yes'])

# add the model to the data frame
score['Predicted_Sales'] = model_prediction_test

# show the results
score
```



# The Python Code of Lesson 5 - V



```
# now we do the evaluation

print("Catalog marketing campaign results")
print("-----\n")

total_revenue = sum(score['Predicted_Sales']*score['Score_Yes'])
print("Total expected revenue from the marketing campaign: ", '${:,.2f}'.format(total_revenue))

adjusted_revenue = total_revenue * 0.5
print("Applying gross margin of 50%: ", '${:,.2f}'.format(adjusted_revenue))

number_customers = score.shape[0]
print_costs = 6.5
catalog_costs = print_costs * number_customers
print("Print costs are $6.5 x", number_customers, "customer(s):", '${:,.2f}'.format(catalog_costs))

print("")
expected_profit = adjusted_revenue - catalog_costs
print("The expected profit from the marketing campaign is:", '${:,.2f}'.format(expected_profit))

if (expected_profit > 10000):
    print("Recommendation: GO! We should do the campaign.")
else:
    print("Recommendation: NO GO! We should not do the campaign.")
```