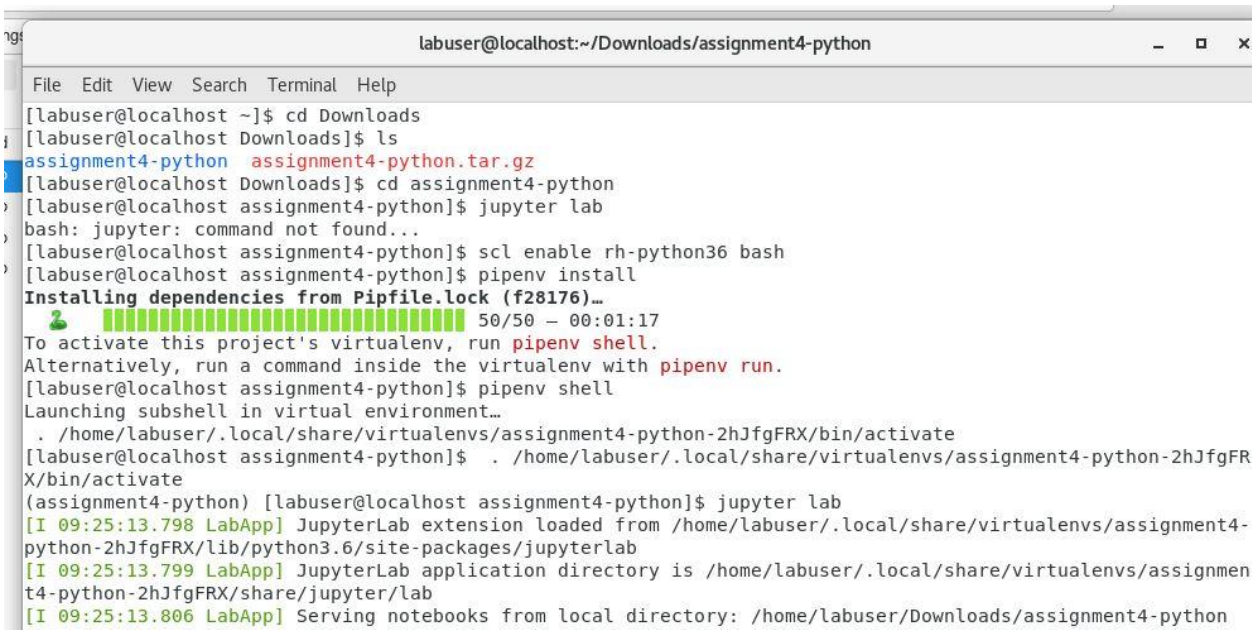
User's Manual

Initially, activate virtual environment as shown below and displayed in Figure 1.

- Open bash and direct to main folder.
- Switch to python 3 with following command:
  - scl enable rh-python36 bash
- Create virtual environment:
  - pipenv install
- Start Jupyter Notebook by:
  - pipenv shell
  - jupyter lab



labuser@localhost:~/Downloads/assignment4-python
File Edit View Search Terminal Help

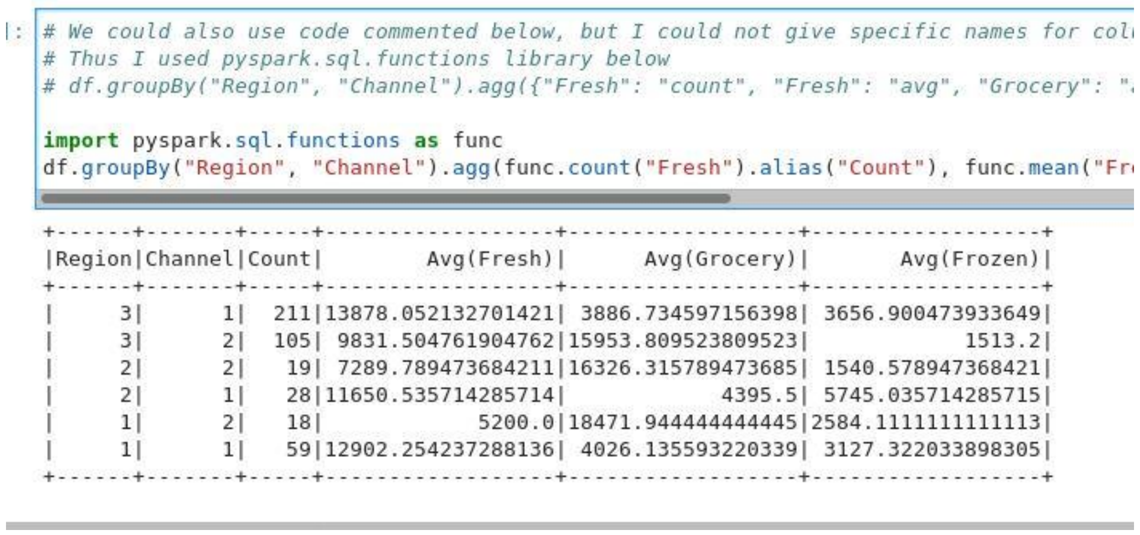
```
[labuser@localhost ~]$ cd Downloads
[labuser@localhost Downloads]$ ls
assignment4-python  assignment4-python.tar.gz
[labuser@localhost Downloads]$ cd assignment4-python
[labuser@localhost assignment4-python]$ jupyter lab
bash: jupyter: command not found...
[labuser@localhost assignment4-python]$ scl enable rh-python36 bash
[labuser@localhost assignment4-python]$ pipenv install
Installing dependencies from Pipfile.lock (f28176)…
      ████████████████████████████████ 50/50 — 00:01:17
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
[labuser@localhost assignment4-python]$ pipenv shell
Launching subshell in virtual environment…
 . /home/labuser/.local/share/virtualenvs/assignment4-python-2hJfgFRX/bin/activate
[labuser@localhost assignment4-python]$  . /home/labuser/.local/share/virtualenvs/assignment4-python-2hJfgFR
X/bin/activate
(assignment4-python) [labuser@localhost assignment4-python]$ jupyter lab
[I 09:25:13.798 LabApp] JupyterLab extension loaded from /home/labuser/.local/share/virtualenvs/assignment4-
python-2hJfgFRX/lib/python3.6/site-packages/jupyterlab
[I 09:25:13.799 LabApp] JupyterLab application directory is /home/labuser/.local/share/virtualenvs/assignmen
t4-python-2hJfgFRX/share/jupyter/lab
[I 09:25:13.806 LabApp] Serving notebooks from local directory: /home/labuser/Downloads/assignment4-python
```

Figure 1.

Task 1.

- First group data by region and channel (after completing importing necessary libraries and data).
- In the grouped elements; aggregate count, average Fresh products, average Grocery products and average Frozen products.
- Order by region to see table in order.

The resulting table is as follows:



```
# We could also use code commented below, but I could not give specific names for col
# Thus I used pyspark.sql.functions library below
# df.groupBy("Region", "Channel").agg({"Fresh": "count", "Fresh": "avg", "Grocery": "

import pyspark.sql.functions as func
df.groupBy("Region", "Channel").agg(func.count("Fresh").alias("Count"), func.mean("Fr
```

```
+------+-------+-----+------------------+------------------+------------------+
|Region|Channel|Count|        Avg(Fresh)|      Avg(Grocery)|       Avg(Frozen)|
+------+-------+-----+------------------+------------------+------------------+
|     3|      1|  211|13878.052132701421| 3886.734597156398| 3656.900473933649|
|     3|      2|  105| 9831.504761904762|15953.809523809523|            1513.2|
|     2|      2|   19| 7289.789473684211|16326.315789473685| 1540.578947368421|
|     2|      1|   28|11650.535714285714|            4395.5| 5745.035714285715|
|     1|      2|   18|            5200.0|18471.944444444445|2584.1111111111113|
|     1|      1|   59|12902.254237288136| 4026.135593220339| 3127.322033898305|
+------+-------+-----+------------------+------------------+------------------+
```

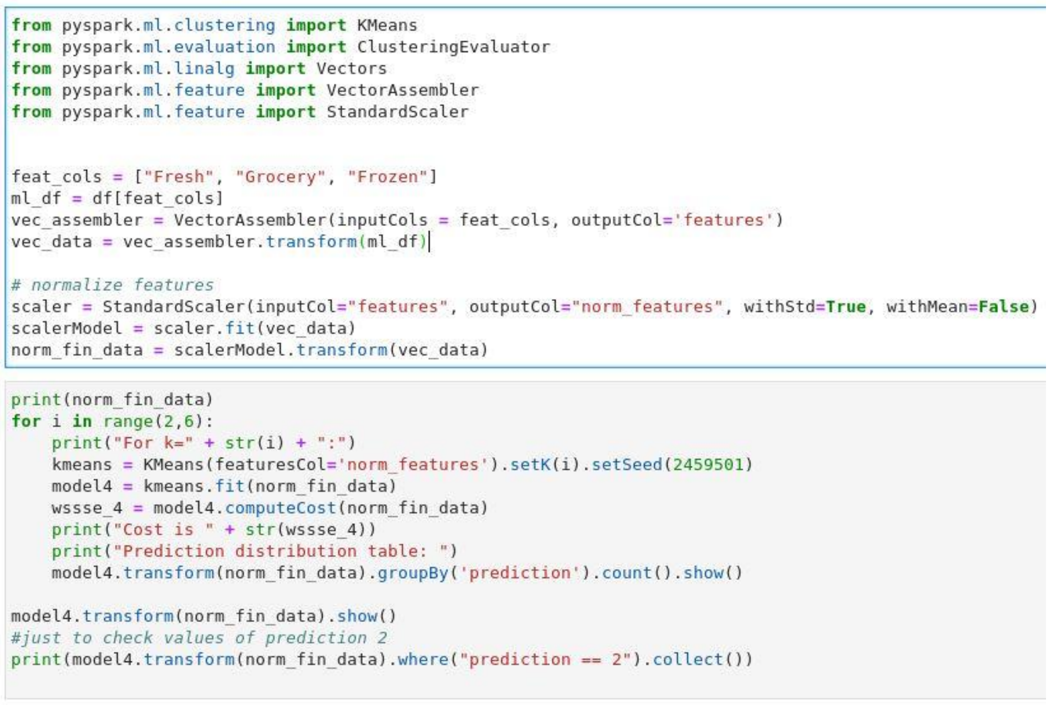
Figure 2.

Even though, rates can change, second channel has higher average annual spending on Grocery products and on every region as we see in Figure 2. Fresh and Frozen products have higher average for the channel 1. So, if we assume that Fresh and Frozen products are generally consumed at hotels, restaurants and cafes, channel 1 is probably the Horeca channel. And if we assume that grocery products are bought at retail places, channel 2 is probably the retail channel.

Also there are differences in regions, too. If we consider that all regions have different geographical, economical features, it is expected to see some differences on average spending of products.

Task 2.

- Import necessary libraries.
- Extract features that will be clustered.
- For kMeans, generate features column as input column.
- Scale data and set results into another column (“norm_features”).
- Run KMeans algorithm in a for loop to see results of different k values.



```
from pyspark.ml.clustering import KMeans
from pyspark.ml.evaluation import ClusteringEvaluator
from pyspark.ml.linalg import Vectors
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.feature import StandardScaler


feat_cols = ["Fresh", "Grocery", "Frozen"]
ml_df = df[feat_cols]
vec_assembler = VectorAssembler(inputCols = feat_cols, outputCol='features')
vec_data = vec_assembler.transform(ml_df)

# normalize features
scaler = StandardScaler(inputCol="features", outputCol="norm_features", withStd=True, withMean=False)
scalerModel = scaler.fit(vec_data)
norm_fin_data = scalerModel.transform(vec_data)
```

```
print(norm_fin_data)
for i in range(2,6):
    print("For k=" + str(i) + ":")
    kmeans = KMeans(featuresCol='norm_features').setK(i).setSeed(2459501)
    model4 = kmeans.fit(norm_fin_data)
    wssse_4 = model4.computeCost(norm_fin_data)
    print("Cost is " + str(wssse_4))
    print("Prediction distribution table: ")
    model4.transform(norm_fin_data).groupBy('prediction').count().show()

model4.transform(norm_fin_data).show()
#just to check values of prediction 2
print(model4.transform(norm_fin_data).where("prediction == 2").collect())
```

Figure 3.

K Means Cluster Result Table

K	Cost	Number of instances
2	1053	[98, 348]
3	776	[362, 68, 10]
4	578	[307, 50, 3, 80]
5	486	[274, 104, 3, 49, 10]
6	442	[209, 4, 6, 78, 42, 101]

Based on these results, which K value would you choose and why?

The cost will keep decreasing as cluster number increases. Thus, we should find an optimal k. We could follow elbow method to decide optimum k value. I checked results with k = 6 and cost was around 442. Cost difference at next k values were around 200 – 300 at the beginning. I think I choose k=5 for optimal k value considering that cost difference of next k value is around only 44. It could be acceptable as the elbow curve point which also could be visible in the Figure 4 shown below.

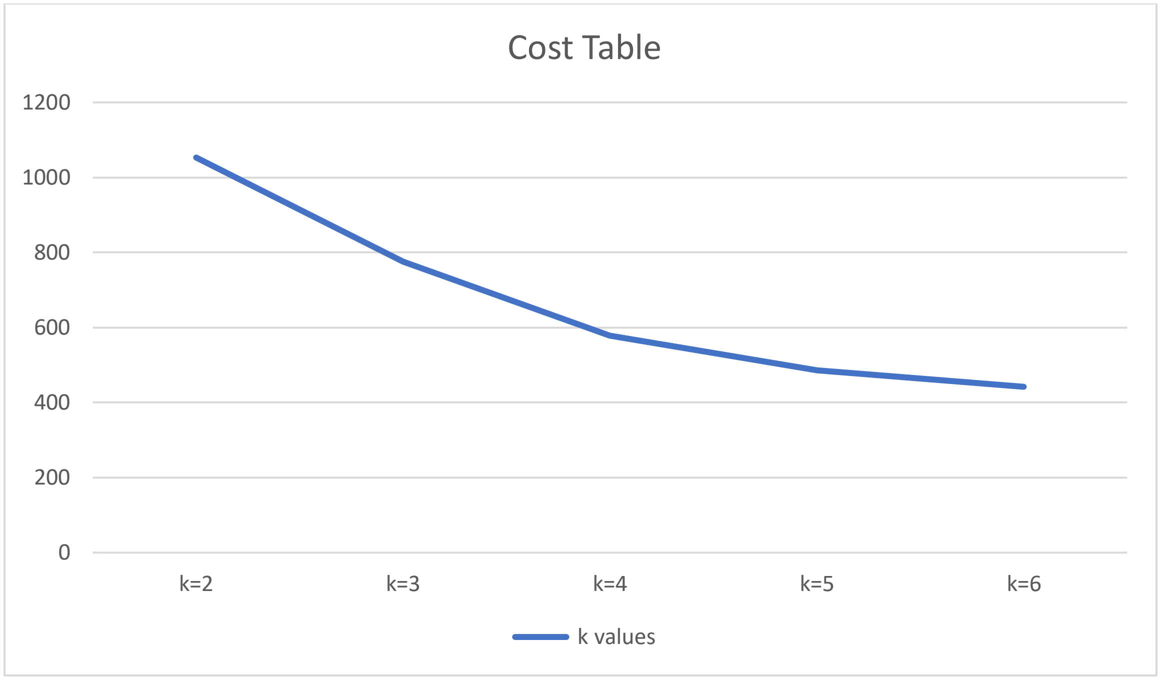


Figure 4.