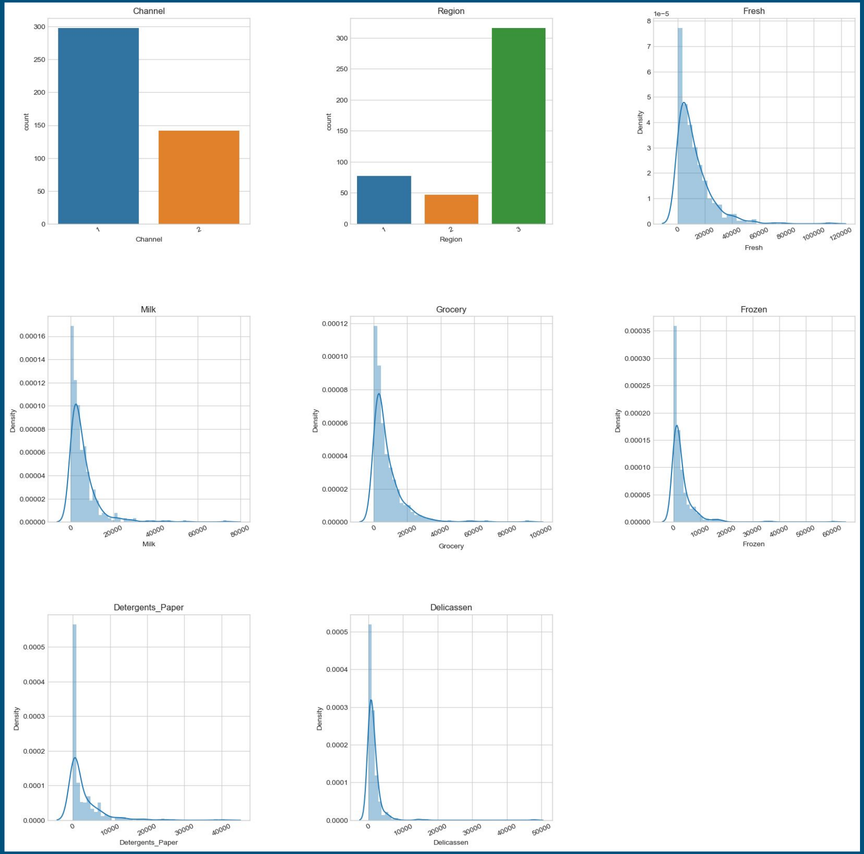
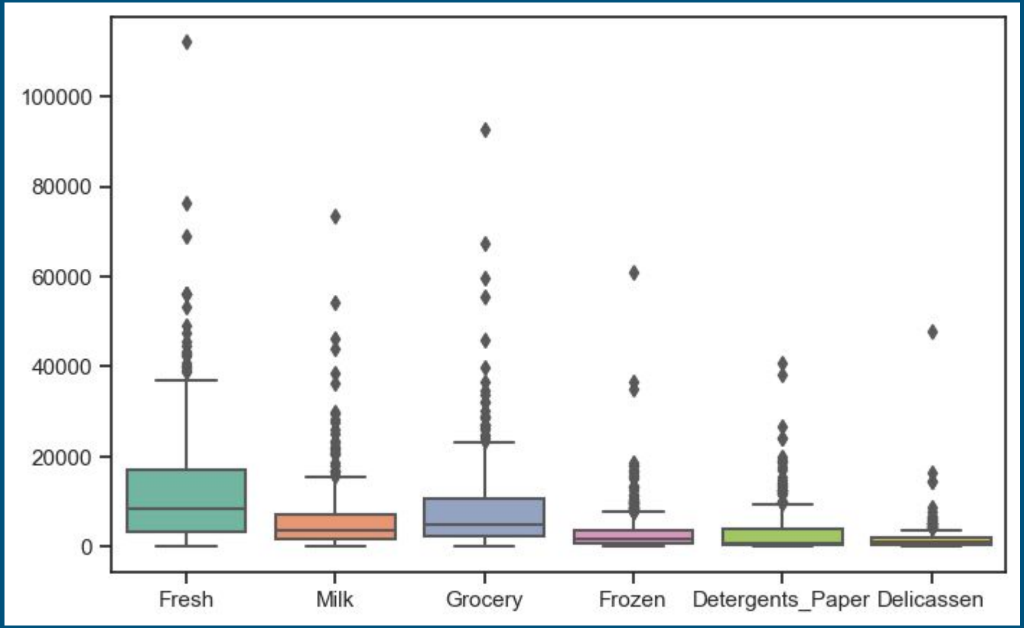
Lab ML For Data Science: Part I

Getting Insights into an Unsupervised Dataset

1.

Loading the Data, Preprocessing, Initial Data Analysis





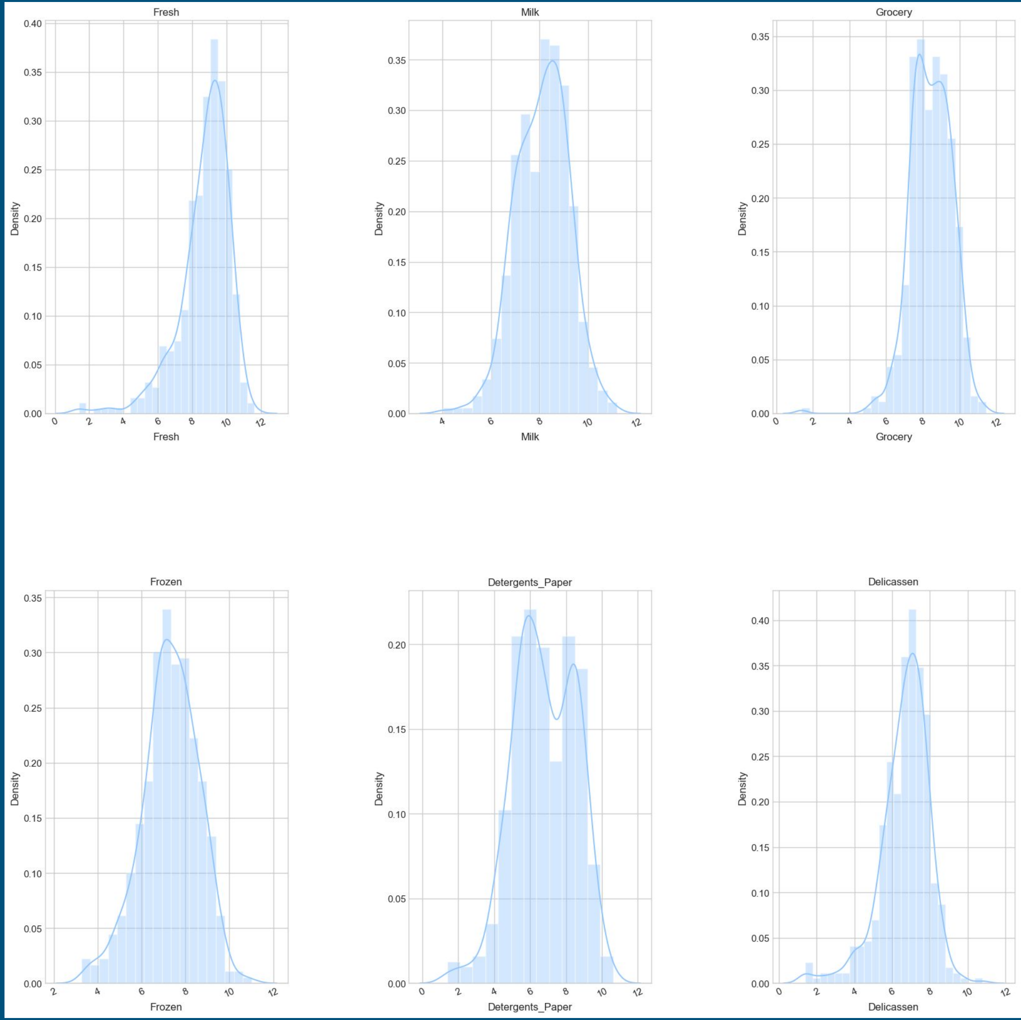
A general distribution shows that there is an imbalance between the number of instances in channel and in region.

For each category, there are many outliers towards the higher end indicating that data is highly skewed to the right.

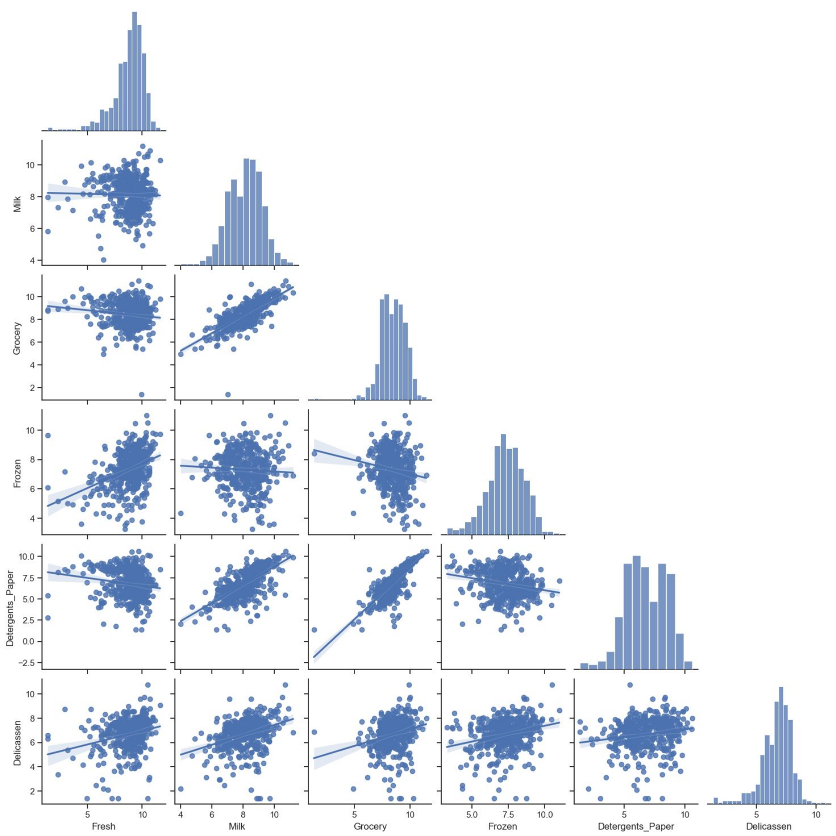
We drop the metadata Channel and Region and focus on numerical data to analyse anomalies.

Since data was highly skewed towards right, we apply log transform
X <- log(x + 1)

We add offset 1 in log so that 0 values get mapped to a specific value.



There is a high correlation between Detergents_Paper and Grocery. Also, Milk and Grocery are considerably correlated.



2.

Detecting Anomalies

Anomaly Scores from Distance Matrix (Hardmin)

```
dist_matrix = calculate_dist_matrix(df_log)
dist_matrix
```

	0	1	2	3	4	5	6	7	8	9	...	430	431	432	433	434
0	0.000000	2.224584	3.085157	4.366366	3.331161	1.307943	1.645392	2.330232	1.597930	2.345042	...	3.195456	4.266183	2.642341	3.901349	1.575182
1	2.224584	0.000000	1.541680	3.263750	1.988811	1.362131	2.174861	0.780828	2.092733	1.169105	...	3.126408	2.383102	3.582755	3.361482	1.780825
2	3.085157	1.541680	0.000000	3.431039	1.651504	2.294735	3.340546	1.337177	3.137447	1.921839	...	3.550237	2.656078	4.519018	3.651499	2.882593
3	4.366366	3.263750	3.431039	0.000000	2.406136	3.262152	3.557725	2.905404	3.407516	4.238947	...	3.455373	2.036131	3.668271	2.804716	3.128861
4	3.331161	1.988811	1.651504	2.406136	0.000000	2.411643	3.233237	1.699094	3.257427	2.746759	...	3.630900	2.048335	4.048988	3.706398	2.573983
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
435	5.069015	3.848109	4.030637	3.060203	2.953376	4.127355	4.921473	3.951802	4.778894	4.711158	...	4.008258	2.771055	5.387995	4.462180	4.273217
436	5.564116	5.161151	5.187257	2.677452	4.048232	4.616688	5.134254	4.925099	4.909501	6.240254	...	4.710604	4.049138	4.677306	3.630164	4.717930
437	2.390978	2.502427	3.148681	5.387722	3.702175	2.904145	2.933326	2.685797	3.298699	1.598576	...	4.633749	4.682201	4.327725	5.569987	2.807914
438	3.789719	3.735002	3.950812	2.296451	3.330583	2.943401	3.540550	3.484226	2.993919	4.704888	...	2.547515	3.588645	3.204004	1.992430	3.215658
439	4.616926	5.708060	6.821026	6.012160	6.799926	4.748969	4.066376	5.757638	3.757454	6.120177	...	4.797128	6.614062	3.424337	4.664636	4.576984

440 rows × 440 columns

1. Calculate distance of one point from all the other points.

2. Get Min. distance of each point to any other point (i.e. taking the minimum value for each row, denoting a data point)

3. Check the data points with the Maximum of minimum distance.

```
338    4.945913
75     4.647444
154    4.201955
142    3.774788
95     3.746081
```

Top Anomalous Points

Anomaly Scores from Distance Matrix (Softmin)

$$\operatorname{soft\,min}_{k \neq j} \{z_{jk}\} = -\frac{1}{\gamma} \log \left(\frac{1}{N-1} \sum_{k \neq j} \exp(-\gamma z_{jk}) \right).$$

1. Apply Softmin function on distance scores instead of directly taking the minimum.

(Initially, we assume γ to be 0.5)

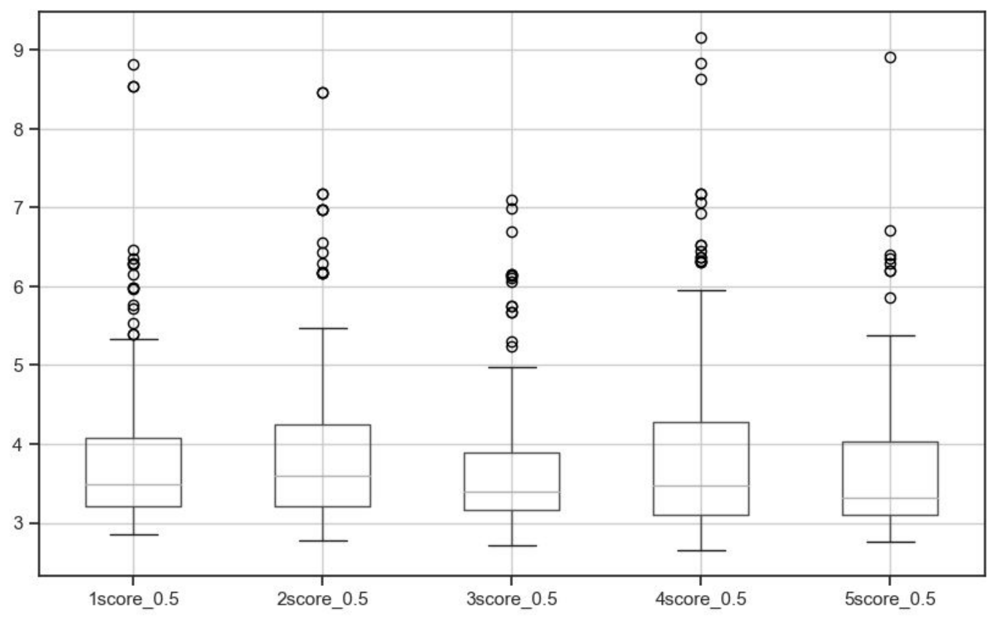
2. Check the data points with Maximum minimum distance.

338	9.151482
154	8.881429
75	8.698126
95	7.730110
66	7.595647

Top Anomalous Points

Anomaly Scores from Distance Matrix (Softmin)

Is the Softmin score Reproducible?



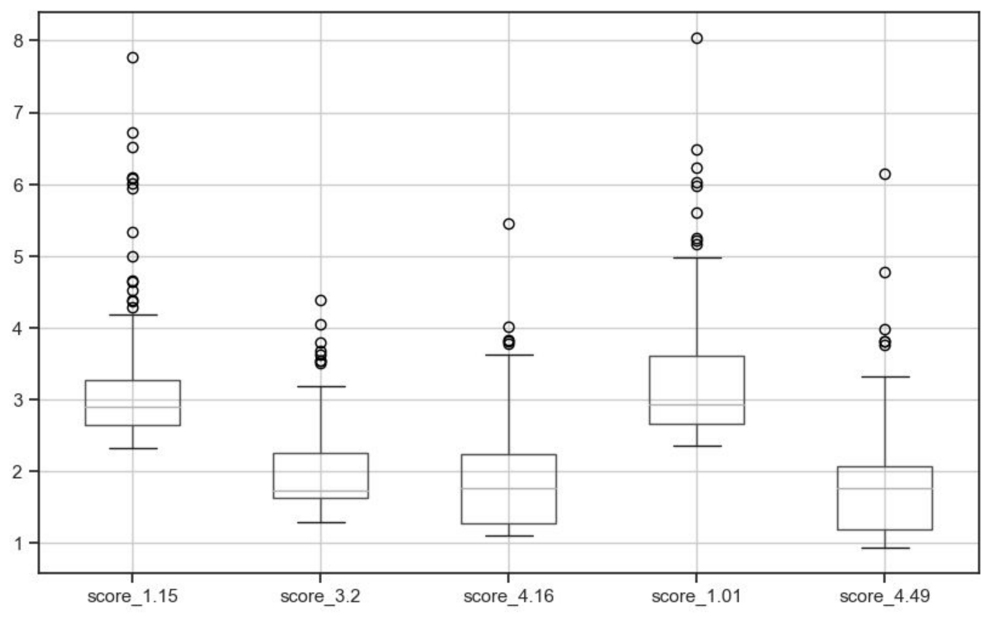
1. Apply Bootstrapping to the dataset (with a 50% sampling)

2. The Softmin scores usually fall in the range of 3 to 4 (for γ = 0.5).

3. There are some outliers for each case - deemed to be anomalous.

Anomaly Scores from Distance Matrix (Softmin)

Choosing γ :

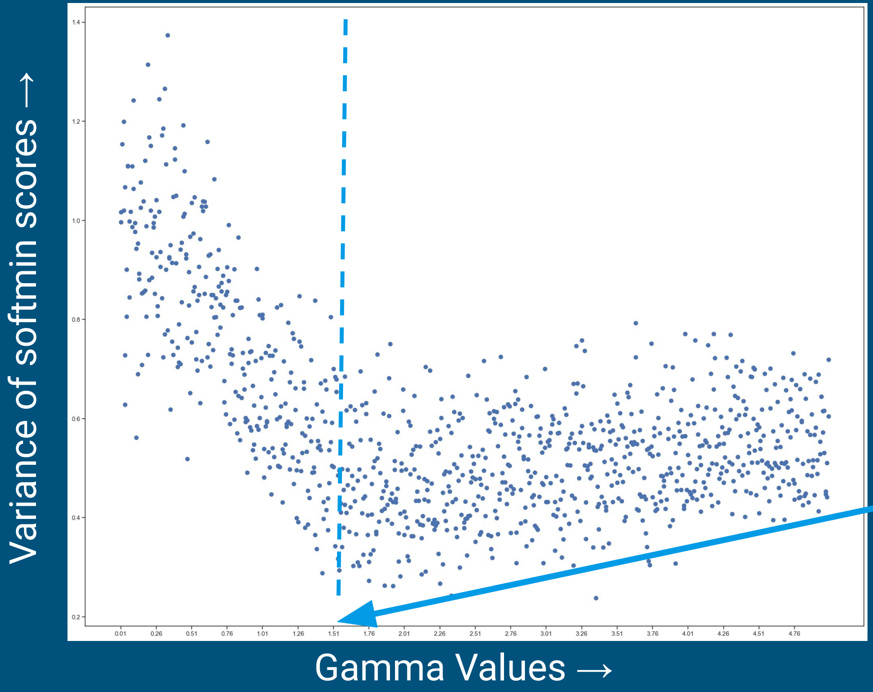


1. Use different γ values on bootstrapped data.

2. The range of Softmin scores change with different γ

Anomaly Scores from Distance Matrix (Softmin)

What is the best value of γ ?



1. Experiment with different γ in the range from 0 to 5.

2. Notice, after increasing γ to a certain limit, drop in Variance is stagnant.

3. We choose the cut-off value as our γ

We decide γ to be 1.5!

Analyzing Anomaly Points:

We apply softmin with γ = 1.5 and look at most anomalous points:

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
338	3	333	7021	15601	15	550
75	20398	1137	3	4407	3	975
154	622	55	137	75	7	8
95	3	2920	6252	440	223	709
142	37036	7152	8253	2995	20	3
128	140	8847	3823	142	1062	3
187	2438	8002	9819	6269	3459	3
66	9	1534	7417	175	3468	27
109	1406	16729	28986	673	836	3
183	36847	43950	20170	36534	239	47943

How does these data points look like compared to the rest?

Comparing Anomaly Points:

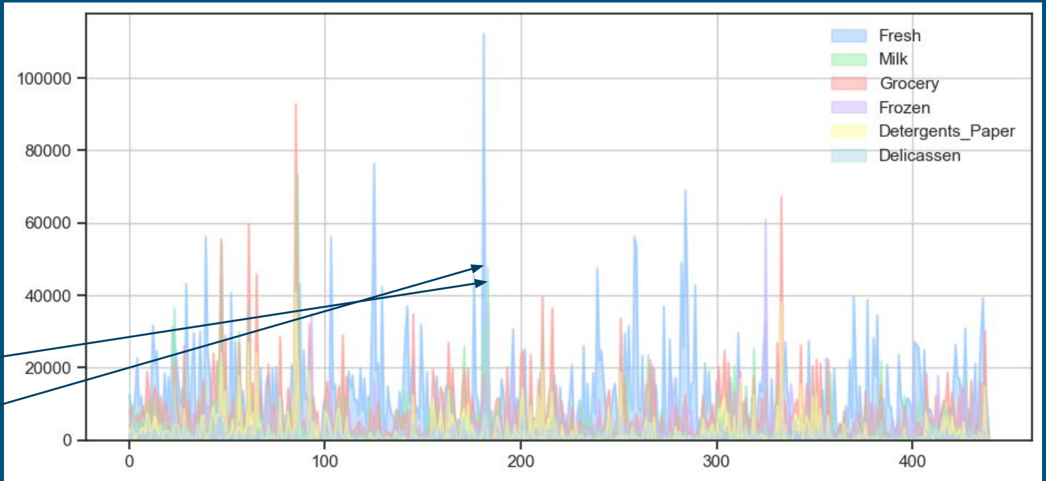
	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
338	3	333	7021	15601	15	550
75	20398	1137	3	4407	3	975
154	622	55	137	75	7	8
95	3	2920	6252	440	223	709
142	37036	7152	8253	2995	20	3
128	140	8847	3823	142	1062	3
187	2438	8002	9819	6269	3459	3
66	9	1534	7417	175	3468	27
109	1406	16729	28986	673	836	3
183	36847	43950	20170	36534	239	47943

	Mean	Std.	25%	50%	75%
Fresh	12000.30	12647.33	3127.75	8504.0	16933.75
Milk	5796.27	7380.38	1533.00	3627.0	7190.25
Grocery	7951.28	9503.16	2153.00	4755.5	10655.75
Frozen	3071.93	4854.67	742.25	1526.0	3554.25
Detergents_Paper	2881.49	4767.85	256.75	816.5	3922.00
Delicassen	1524.87	2820.11	408.25	965.5	1820.25

Customer 338 seems to be a valid outlier based on their purchase of Fresh Foods, Milk etc.

Comparing Anomaly Points:

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
338	3	333	7021	15601	15	550
75	20398	1137	3	4407	3	975
154	622	55	137	75	7	8
95	3	2920	6252	440	223	709
142	37036	7152	8253	2995	20	3
128	140	8847	3823	142	1062	3
187	2438	8002	9819	6269	3459	3
66	9	1534	7417	175	3468	27
109	1406	16729	28986	673	836	3
183	36847	43950	20170	36534	239	47943



	Mean	Std.	25%	50%	75%
Fresh	12000.30	12647.33	3127.75	8504.0	16933.75
Milk	5796.27	7380.38	1533.00	3627.0	7190.25
Grocery	7951.28	9503.16	2153.00	4755.5	10655.75
Frozen	3071.93	4854.67	742.25	1526.0	3554.25
Detergents_Paper	2881.49	4767.85	256.75	816.5	3922.00
Delicassen	1524.87	2820.11	408.25	965.5	1820.25

Comparing with the distribution of expenses for each customer, we can also validate our anomalies, especially when they overspend on a category.

3.

Explaining Anomalies

Explainable AI

It becomes important to explain the anomalies along with identifying them. In this task, we use the method of Layer-wise Relevance Propagation to explain high anomaly score.

$$R_k^{(j)} = \frac{\exp(-\gamma z_{jk})}{\sum_{k \neq j} \exp(-\gamma z_{jk})} \cdot y_j$$

For each instance j, the anomaly score is determined by softmin, and it is influenced by every other datapoint. Hence, as a first step, we identify how much each datapoint contributes to anomaly score of each instance.

	0	1	2	3	4	5	6	7	8	9	...	430	
0	0.000000e+00	2.671634e-03	2.818816e-06	1.701283e-12	2.641206e-07	3.437024e-01	7.708246e-02	1.298071e-03	9.710369e-02	1.170013e-03	...	9.972019e-07	6.225
1	2.330389e-04	0.000000e+00	1.103909e-02	4.488127e-08	1.034124e-03	2.413125e-02	3.235429e-04	1.563458e-01	5.473370e-04	5.021935e-02	...	1.674156e-07	7.791
2	3.072270e-06	1.379350e-01	0.000000e+00	1.045236e-07	8.151190e-02	1.809839e-03	2.620634e-07	3.335802e-01	1.885782e-06	1.913940e-02	...	2.999974e-08	1.236
3	3.754520e-13	1.135511e-07	2.116409e-08	0.000000e+00	1.670611e-04	1.153419e-07	5.608237e-09	3.128444e-06	2.693978e-08	1.944679e-12	...	1.646026e-08	1.966
4	1.078237e-07	4.839867e-03	3.053097e-02	3.090364e-04	0.000000e+00	2.969804e-04	2.827938e-07	2.403619e-02	2.234534e-07	2.221279e-05	...	4.712875e-09	3.375
...	...	...	...	...	...	...	...	...	...	...	...	...	
435	1.068199e-15	1.320921e-08	1.527715e-09	4.642375e-05	1.216846e-04	4.677782e-10	9.747337e-15	3.926401e-09	7.760250e-14	2.035311e-13	...	2.000966e-09	5.822
436	1.045218e-19	6.833979e-17	4.557073e-17	3.291693e-04	3.247086e-10	2.007871e-13	1.035302e-16	2.430392e-15	3.059199e-15	6.603872e-25	...	5.396174e-14	3.211
437	1.301597e-03	5.743844e-04	2.399415e-06	8.489178e-19	8.126344e-09	2.209650e-05	1.711411e-05	1.378613e-04	5.623617e-07	1.492442e-01	...	7.097483e-14	3.606
438	5.346820e-10	9.915559e-10	8.237261e-11	4.452303e-04	7.207491e-08	2.755808e-06	8.278268e-09	1.498615e-08	1.757323e-06	4.610923e-15	...	7.183499e-05	4.949
439	2.373244e-10	1.087025e-17	8.958749e-27	5.181062e-20	1.378695e-26	3.712847e-11	3.087601e-07	4.633684e-18	1.159115e-05	7.256126e-21	...	1.863010e-11	5.802

440 rows × 440 columns

For each instance, we calculate the relevance contribution of each datapoint as a 440x440 matrix, where each row represents each instance and contribution of each datapoint to softmin is given in columns.

Explainable AI

$$R_i^{(j)} = \sum_{k \neq j} \frac{[\boldsymbol{x}_k - \boldsymbol{x}_j]_i^2}{\|\boldsymbol{x}_k - \boldsymbol{x}_j\|^2} \cdot R_k^{(j)}$$

The relevance contribution of each datapoint can then further be backtracked to each features by observing that the (squared) Euclidean distance entering the anomaly score can be decomposed in terms of individual components:

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen	Total_Relevance
0	0.904516	0.583854	0.250292	1.437025	0.468388	0.468318	4.112392
1	0.644680	0.661569	0.185157	0.631423	0.350877	0.279917	2.753624
2	0.589532	0.614826	0.175790	0.448208	0.342300	1.991170	4.161826
3	0.464723	0.687186	0.493128	0.437054	0.460333	0.717498	3.259922
4	0.472894	0.313277	0.237213	1.141064	0.595644	0.843142	3.603235
...	...	...	...	...	...	...	...
435	0.202129	1.107492	1.623817	0.656590	1.101971	0.926649	5.618648
436	0.600827	0.809939	1.203615	0.727913	1.038167	0.448858	4.829319
437	0.324045	0.274384	0.377892	1.894288	0.638277	0.852863	4.361749
438	0.708035	0.479280	0.290250	0.788470	0.299125	0.809403	3.374563
439	1.141659	1.065617	1.690130	2.347667	0.499341	2.379591	9.124005

440 rows × 7 columns

For each instance, with this split of relevance contribution, we can clearly see what feature contributes the most to the anomaly score, making it more explainable.

Explainable AI

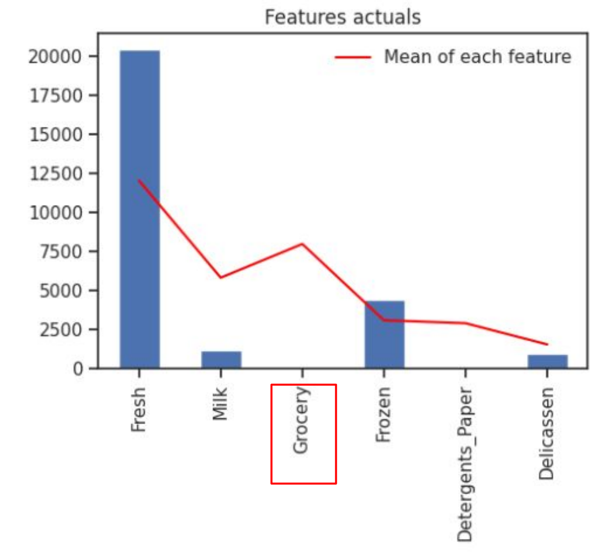


Fig: Actual Spending across categories

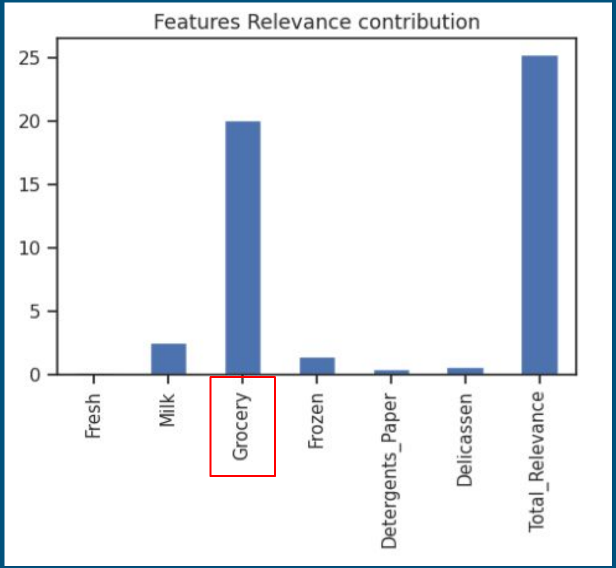


Fig: Relevance Contribution across categories

This is an example where we can see, the highest contributor to anomaly score is Grocery; where the spending on grocery is nil and the average spending is much higher.

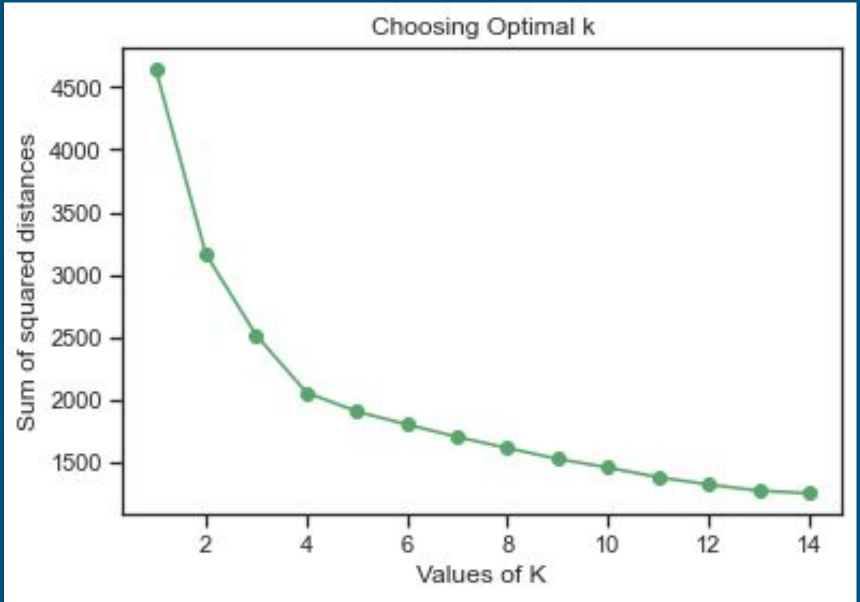
Only with anomaly score, it might not be intuitive to say if it is overspending or underspending which makes this an outlier.

	Mean	Std.	25%	50%	75%
Fresh	12000.30	12647.33	3127.75	8504.0	16933.75
Milk	5796.27	7380.38	1533.00	3627.0	7190.25
Grocery	7951.28	9503.16	2153.00	4755.5	10655.75
Frozen	3071.93	4854.67	742.25	1526.0	3554.25
Detergents_Paper	2881.49	4767.85	256.75	816.5	3922.00
Delicassen	1524.87	2820.11	408.25	965.5	1820.25

4.

Cluster Analysis

K-Means Clustering on Remaining Data:




```
Cluster 1 has 66 members
Cluster 2 has 129 members
Cluster 3 has 105 members
Cluster 4 has 130 members
```

```
df_log['cluster'] = kmeans.labels_
df_log
```

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen	cluster
0	9.446992	9.175438	8.930891	5.370638	7.891705	7.199678	2
1	8.861917	9.191259	9.166284	7.474772	8.099858	7.482682	2
2	8.756840	9.083529	8.947026	7.785721	8.165364	8.967632	2
3	9.492960	7.087574	8.348064	8.764834	6.230481	7.489412	3
4	10.026413	8.596189	8.881697	8.272826	7.483244	8.553718	2
...	...	...	...	...	...	...	...
435	10.299037	9.396986	9.682092	9.483112	5.209486	7.698483	3
436	10.577172	7.266827	6.639876	8.414274	4.543295	7.760893	3
437	9.584108	9.647885	10.317053	6.082219	9.605216	7.532624	2
438	9.239025	7.591862	7.711101	6.946014	5.129899	7.661998	1
439	7.933080	7.437795	7.828436	4.189655	6.169611	3.970292	1

430 rows × 7 columns

We decide 4 clusters to be optimal. We introduce a new column, "cluster", denoting the membership of a point.

Analyzing Results of K-Means Clustering

These are our 4 cluster centres (how a “typical” customer looks like) in our scaled dataset

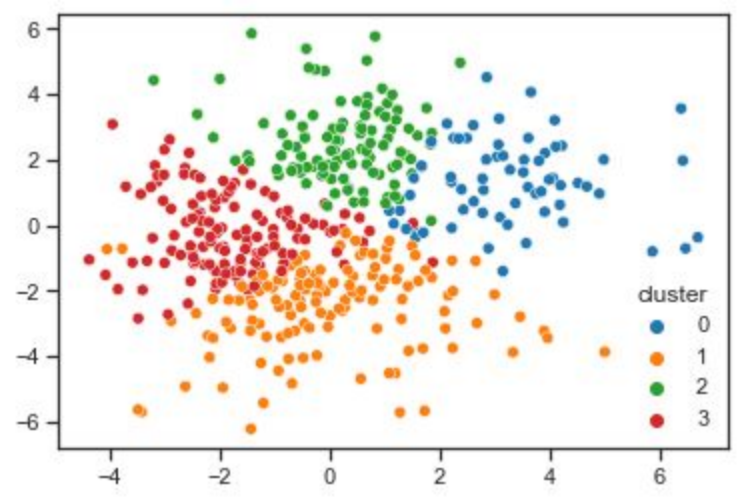
```
array([[7.071, 8.782, 9.372, 5.859, 8.503, 6.262],
       [8.628, 7.224, 7.516, 6.913, 5.315, 6.024],
       [9.143, 9.095, 9.457, 7.351, 8.515, 7.391],
       [9.54 , 7.904, 8.108, 8.389, 6.133, 7.133]])
```

vs.

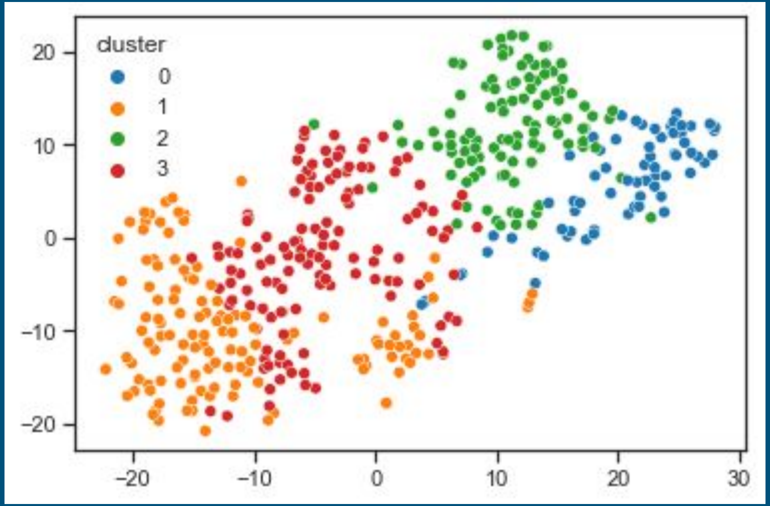
	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
count	430.000000	430.000000	430.000000	430.000000	430.000000	430.000000
mean	8.790636	8.125476	8.453723	7.304672	6.833045	6.729462
std	1.338203	1.054783	1.050306	1.259815	1.667777	1.156351
min	2.944439	4.727388	5.389072	3.258097	1.386294	2.079442
25%	8.109368	7.360222	7.673222	6.646058	5.593719	6.044349
50%	9.057013	8.196435	8.464846	7.331043	6.711132	6.894670
75%	9.735449	8.868730	9.275776	8.166423	8.295236	7.510567
max	11.627610	11.205027	11.437997	11.016496	10.617123	9.712569

Analyzing Results of K-Means Clustering

We also use different embedding methods and check if K-Means cluster membership gives a good result.



MDS



T-SNE