	Import Re	equire	d Pacl	kages	5												
In [1]:	from mlxtend.	<pre>import pandas as pd from mlxtend.preprocessing import TransactionEncoder from mlxtend.frequent_patterns import apriori, association_rules</pre>															
	Load the	Datas	et														
In [2]:	<pre># Load the dataset df = pd.read_csv('bank.csv')</pre>																
	<pre># Display the df.head()</pre>	e first f	ew rows	of the	dataset												
Out[2]:	Account No	D DATI	<b>E</b>	TRANSA	ACTION DE	ETAILS	CHO.N	O. VALUE	DATE WIT	HDRAWAL AMT	DEPOSIT AMT	BALANCE AMT					
000[2].	<b>0</b> 409000611074		7 TRF FR				Na		Jun-17	NaN	1,000,000.00	1,000,000.00					
	<b>1</b> 409000611074	' 05-Jul-1	7 TRF FR	OM Indiafo	rensic SEF	VICES	Na	ıN 05-	Jul-17	NaN	1,000,000.00	2,000,000.00					
	<b>2</b> 409000611074	' 18-Jul-1	7 FDRL/	INTERNAL	. FUND TR	ANSFE	Na	ıN 18-	Jul-17	NaN	500,000.00	2,500,000.00					
	<b>3</b> 409000611074	_			rensic SEF		Na		Aug-17	NaN	3,000,000.00	5,500,000.00					
	<b>4</b> 409000611074	!' 16-Aug-1	7 FDRL/	INTERNAL	. FUND TR.	ANSFE	Na	IN 16-A	Aug-17	NaN	500,000.00	6,000,000.00	•				
	Prepare t	he Da	ta														
In [4]:	<pre>df.columns = print(df.columns)</pre>		ns.str.s	trip()													
		unt No', DRAWAL AM object')							VALUE DA	TE',							
In [5]:	<pre># Combine relevant columns into a single string per transaction df['transaction'] = df.apply(lambda row: f"{row['TRANSACTION DETAILS']}, {row['WITHDRAWAL AMT']}, {row['DEPOSIT AMT']}", axis=1)</pre>																
	<pre># Create a list of transactions transactions = df['transaction'].apply(lambda x: x.split(','))</pre>																
	<pre># Convert to list of lists data = transactions.tolist()</pre>																
	Transforn	n Data	using	g Tran	sactio	on E	ncod	der									
In [6]:	<pre># Initialize te = Transact</pre>			ler													
	<pre># Transform t te_data = te.</pre>		).transf	orm(dat	a)												
	# Convert to df_transforme	ed = pd.D	ataFrame	(te_dat	a, colum	ıns=te	.colum	ns_)									
Out[6]:	df_transforme	ed.head()							ZEN LEF				ZEN LEFIN	ZEN LEFIN	ZEN LEFIN	ZEN LEFIN	Z
	SERVICE	0.01 0.	02 0.03	0.04 0	0.05 0.06	0.07	0.08	0.09		VT PV 46 LTD161223B			PVT LTD170123CF	PVT LTD170125A0	PVT LTD1701302E	PVT LTD170206BA L	.TC
		False Fal							Fa				False	False	False	False	
		False Fal							Fa Fa				False False	False False	False False	False False	
	2 1 4136	raise rai							Fa					False	False	False	
	<b>3</b> False	False Fal	se Faise							ise Fais	t raist	e Faise	False				
		False Fal		False Fa	alse False	False			Fa				False False	False	False	False	
		False Fal		False Fa	alse False	False									False	False	
	4 False 5 rows × 64476 c	False Fal columns	se False		alse False	False									False	False	
	4 False	False Fal columns	se False		alse False	False									False	False	
	4 False 5 rows × 64476 c	False Falcolumns  riori Alcori algori	se False  gorith  ithm	m			False	False	Fa	lse Fals					False	False	
	4 False 5 rows × 64476 c  Apply April	False Falcolumns  riori Algorinsets = a  th column nsets['le	gorith	<b>M</b> f_trans number	formed, of item	min_su	False upport:	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
	4 False 5 rows × 64476 c  Apply Api  # Apply apric frequent_item  # Add a lengt frequent_item	False Falcolumns  riori Algorinsets = a  th column nsets['le	gorith	<b>M</b> f_trans number	formed, of item nt_items	min_su	False upport: the ite	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False 5 rows × 64476 c  Apply Api  # Apply apric frequent_item  # Add a lengt frequent_item frequent_item	False Falcolumns  riori Algorinsets = a  th column nsets['le	gorith	<b>M</b> f_trans number	formed,  of items  items	min_su ns in u sets[':	False upport: the ite	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False 5 rows × 64476 c  Apply Api  # Apply apric frequent_item  # Add a lengt frequent_item frequent_item  support  0 0.090679  1 0.063115	False Falcolumns  riori Algorinsets = a  th column nsets['le	gorith	<b>M</b> f_trans number	formed,  of items  items	min_sums in a sets[': ets len (1) 10)	upport: the ite itemse	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False 5 rows × 64476 co  Apply Api  # Apply aprio frequent_item  # Add a lengt frequent_item frequent_item  support  0 0.090679  1 0.063115  2 0.083467	False Falcolumns  riori Algorinsets = a  th column nsets['le	gorith	<b>M</b> f_trans number	formed,  of items  items  (	min_sins in a sets[': ets len(1) 10) 15)	ralse  upport: the ite itemse  ugth  1 1	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	Apply Aprice frequent_item frequent_item support 0 0.090679 1 0.063115 2 0.083467 3 0.053494	False Falcolumns  riori Algorinsets = a  th column nsets['le	gorith	<b>M</b> f_trans number	formed,  of items  items  (	min_sin_sin_sets[': ets len (1) 10) 15) (2)	ralse  upport: the ite itemse  upport:  the ite itemse  upport:  the ite itemse	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False 5 rows × 64476 c  Apply April  # Apply april frequent_item  # Add a lengt frequent_item frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759	False Falcolumns  riori Algorinsets = a  th column nsets['le	gorith	<b>M</b> f_trans number	formed,  of items  items  (	min_sin_sin_sets[': ets len (1) 10) 15) (2) 00)	ralse  upport: the it itemse	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	Apply Aprice frequent_item frequent_item support 0 0.090679 1 0.063115 2 0.083467 3 0.053494	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith	f_trans number freque	formed,  of items  items  ( ( (000.0	min_sin_sin_sets[': ets len (1) 10) 15) (2) 00)	ralse  upport: the ite itemse  upport:  the ite itemse  upport:  the ite itemse	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False  5 rows × 64476 c  Apply April  # Apply april frequent_item  # Add a lengt frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of items  items  ( ( (000.0	min_sin is in is sets[': ets len (1) 10) 15) (2) 00) 00) FE)	upport: the it itemse	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False  5 rows × 64476 c  Apply Api  # Apply aprid frequent_item  # Add a lengt frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of items  items  ( ( (000.0) ND TRANS ECTRONIC	min_sin is in is sets[': ets len (1) 10) 15) (2) 00) 00) FE)	upport: the ite itemse	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False 5 rows × 64476 of  Apply April  # Apply april frequent_item  # Add a lengt frequent_item frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of items  nt_items  ( ( (000.0 ND TRANS ECTRONIC (nan	min_si ns in n sets[': ets len (1) 10) 15) (2) 00) (2) 00) FE) (F) an) , 1)	raise  upport: the itc itemse  upport:  the itc itemse  upport:  the itc itemse  upport:  upp	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False 5 rows × 64476 of  Apply April # Apply april frequent_item # Add a lengt frequent_item frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of item nt_items  items  ( (000.00) ND TRANS ECTRONIC (nam (000.00)	min_si ns in ns sets[': ets len (1) 10) 15) (2) 00) (2) 00) (FE) (F) an) (1) 10)	raise  upport: the it itemse  upport:  the it itemse  upport:  the it itemse  upport:  upport	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False  5 rows × 64476 of  Apply April frequent_item  # Add a lengt frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055  11 0.063115	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of item nt_items  items  ( (000.00, ND TRANS ECTRONIC (nan (000.00, (nan,	min_si ns in ns sets[': ets len (1) 10) 15) (2) 00) (2) 00) (5) (5) an) (1) 10)	raise  upport: the it itemse  upport:  1 1 1 1 1 2 2 2	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False  5 rows × 64476 of  Apply April  # Apply april frequent_item  # Add a lengt frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055  11 0.063115  12 0.076084	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of items  items  items  (000.00 ND TRANS ECTRONIC (nan (000.00, (nan, (000,	min_sin is in is sets[': ets len (1) 10) 15) (2) 00) (5) (5) (7) (7) (1) (1) (1) (1) (15)	raise  upport: the it itemse  upport:  1 1 1 1 1 2 2 2 2	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False  5 rows × 64476 of  Apply April frequent_item  # Add a lengt frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055  11 0.063115	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of item nt_items  items  ( (000.00, ND TRANS ECTRONIC (nan (000.00, (nan,	min_sin is in is sets[':  ets len (1) 10) 15) (2) 00) (5) (5) (7) (7) (8) (9) (9) (1) (10) (10) (15) (10) (10)	raise  upport: the it itemse  upport:  1 1 1 1 1 2 2 2	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False  5 rows × 64476 of  Apply April  # Apply april frequent_item  # Add a lengt frequent_item  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055  11 0.063115  12 0.076084  13 0.079380	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of item nt_items  items  (000.00, ND TRANS ECTRONIC (nan, (000.00, (nan, (000, (15,000.00)	min_sinsins in insins in i	raise  upport: the it itemse  upport:  1 1 1 1 1 2 2 2 2 2 2 2	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False  5 rows × 64476 of  Apply April  # Apply april frequent_iten  # Add a lengt frequent_iten  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055  11 0.063115  12 0.076084  13 0.079380  14 0.083467	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d for the ngth'] =	f_trans number freque	formed,  of item nt_items  items  (000.00, ND TRANS ECTRONIC (nan, (000.00, (nan, (000, (15,000.00) (15,000.00)	min_sinsins in insets[':  ets len (1) 10) 15) (2) 00) (2) (0) (5) (7) (7) (7) (8) (9) (9) (9) (9) (9) (9) (9) (9) (9) (9	raise  upport: the it itemse  upport:  1 1 1 1 1 2 2 2 2 2 2	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False 5 rows × 64476 of  Apply April  # Apply april frequent_iten  # Add a lengt frequent_iten  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055  11 0.063115  12 0.076084  13 0.079380  14 0.083467  15 0.053494  16 0.218527  17 0.218759	False Fall columns  riori Algorinsets = a th column nsets['lensets	gorith  ithm priori(d  for the ngth'] =	m If_trans number freque	formed,  of item nt_items  items  (000.00 (000.00 (nan (000.00 (15,000.00 (15, n (2, n (000, 000.00 (000, n	min_si  sets   in   is    sets   in   in    sets   in    s	raise  upport: the it itemse  upport:  the it itemse  upport:  2 2 2 2 2 2 2 2 2 2 2 2	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False  5 rows × 64476 of  Apply April  # Apply april frequent_iten  # Add a lengt frequent_iten  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055  11 0.063115  12 0.076084  13 0.079380  14 0.083467  15 0.053494  16 0.218527  17 0.218759  18 0.075344	False Fall columns  riori Algorinsets = a th column nsets['lensets	gorith  ithm priori(d for the ngth'] =	m  If_trans  Inumber  If freque  IT TONAL FUN	formed,  of item nt_items  items  (000.00 (000.00 (nan, (000, (15, 000.0 (15, n (2, n (000, 000.0 (000, n) ND TRANS	min_si ns in ns sets[': ets len (1) 10) 15) (2) 00) (5) (5) (7) (1) (1) (1) (1) (1) (1) (1) (1) (1) (1	raise  upport: the it itemse  upport:  the it itemse  upport:  2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	
In [7]:	4 False 5 rows × 64476 of  Apply April  # Apply april frequent_iten  # Add a lengt frequent_iten  support  0 0.090679  1 0.063115  2 0.083467  3 0.053494  4 0.218759  5 0.509058  6 0.076066  7 0.053889  8 1.000000  9 0.090679  10 0.058055  11 0.063115  12 0.076084  13 0.079380  14 0.083467  15 0.053494  16 0.218527  17 0.218759	False Fall columns  riori Ale pri algor msets = a th column msets['le msets	gorith  ithm priori(d  for the ngth'] =	f_trans number freque	formed,  of item nt_items  items  (000.00 ND TRANS ECTRONIC (nan, (000.00, (15,000.0)	min_sins in a sets[': ets len (1) 10) 15) (2) 00) (5) (5) (7) (8) (9) (1) (1) (1) (1) (1) (1) (1) (1) (1) (1	raise  upport: the it itemse  upport:  the it itemse  upport:  2 2 2 2 2 2 2 2 2 2 2 2	False =0.05, us	Fa	lse Fals nes= <b>True</b> )					False	False	

## Filter the Frequent Itemsets # Filter for itemsets of length >= 2 and support >= 0.05 filtered\_itemsets = frequent\_itemsets[(frequent\_itemsets['length'] >= 2) & (frequent\_itemsets['support'] >= 0.05)] filtered\_itemsets itemsets length Out[8]: support **9** 0.090679 (nan, 1) **10** 0.058055 (000.00, 10)**11** 0.063115 (nan, 10) **12** 0.076084 (000, 15)**13** 0.079380 (15,000.00) **14** 0.083467 (15, nan) **15** 0.053494 (2, nan) **16** 0.218527 (000, 000.00)

27 0.075344 (nan, 000.00, FDRL/INTERNAL FUND TRANSFE)

(nan, FDRL/NATIONAL ELECTRONIC F)

(nan, 000.00, 10)

(000, 15, 000.00)

(15, nan, 000.00)

(000, nan, 000.00)

(15,000, nan,000.00)

(15,000, nan)

3

3

3

3

3

2

2

2

2

2 2

2

2

2 2

3

3

3

1

1

1

2

1

1

1.000000

1.057484

6.123421

1.000000

1.086270

8.833929

1.411555

1.000000

1.086270

0.911537 4.171289 0.057844

0.347797 4.381435 0.058719

0.076084 1.000000 0.000000

0.149460 1.964414 0.037353

lift

0.998938 1.962328

inf

0.000000

0.000000

0.909621

0.476656

0.000000

1.000000

0.829502

0.987870

0.000000

1.000000

1

1

1

1

1

2

3 3

3

3

(000, nan)

(15, nan, 000.00)

(000, nan, 000.00)

(15,000, nan,000.00)

**21** 0.053889

**22** 0.058055

**23** 0.076084

**24** 0.076084

**25** 0.079380

**26** 0.218527

**28** 0.076084

**17** 0.218759

**18** 0.075344

**25** 0.079380

**26** 0.218527

**28** 0.076084

72

73

74

75

Out[10]:

**50** 

Conclusion

practical applications in the real world, particularly in the financial sector.

**Association Rule Mining** 

**19** 0.509058 (nan, 000.00) **20** 0.076066 (nan, FDRL/INTERNAL FUND TRANSFE) **21** 0.053889 (nan, FDRL/NATIONAL ELECTRONIC F) 2 **22** 0.058055 (nan, 000.00, 10) 3 **23** 0.076084 (000, 15, 000.00) 3 **24** 0.076084 3 (15, 000, nan)

27 0.075344 (nan, 000.00, FDRL/INTERNAL FUND TRANSFE)

(15) (000, nan, 000.00)

(nan)

(000.00)

76 rows × 12 columns

(15,000.00, nan)

(000, 15, 000.00)

(000, 15, nan)

(000.00, FDRL/INTERNAL FUND TRANSFE)

Generate and Display Association Rules In [9]: # Generate association rules rules = association\_rules(frequent\_itemsets, metric="lift", min\_threshold=1) # Add length columns for antecedents and consequents rules['antecedents length'] = rules['antecedents'].apply(lambda x: len(x)) rules['consequents length'] = rules['consequents'].apply(lambda x: len(x)) # Display rules rules lift leverage conviction zhangs\_metric antecedents length consequents length Out[9]: antecedents consequents antecedent support consequent support support confidence 0 (nan) (1)1.000000 0.090679 0.090679 0.090679 1.000000 0.000000 1 (1)(nan) 0.090679 1.000000 0.090679 1.000000 1.000000 0.000000 2 (000.00)(10)0.509058  $0.063115 \quad 0.058055$ 0.114043 1.806918 0.025926 3 (000.00)0.063115 0.509058 0.058055 0.919825 1.806918 0.025926 (10)4 1.000000 0.063115 0.063115 (nan) (10)0.063115 1.000000 0.000000 **71** (nan, 000.00) (000, 15)0.509058 0.076084 0.076084 0.149460 1.964414 0.037353

0.083467

0.218759

1.000000

0.509058

0.218527 0.076084

0.079380 0.076084

0.076084 0.076084

0.076084 0.076084

Filter and Display the Rules # Filter rules based on antecedents length and lift filtered\_rules = rules[(rules['antecedents length'] > 1) & (rules['consequents length'] == 1) & (rules['lift'] > 1)] # Sort and display the rules by confidence and lift filtered\_rules\_sorted = filtered\_rules.sort\_values(by=["confidence", "lift"], ascending=False) # Display relevant columns filtered\_rules\_sorted[['antecedents', 'consequents', 'support', 'confidence', 'lift']] consequents support confidence antecedents 32 (000, 15)(000.00) 0.076084 1.000000 1.964414 62 (000.00) 0.076084 1.000000 1.964414 (000, 15, nan)

(000, nan)

(000.00) 0.218527 (nan, FDRL/INTERNAL FUND TRANSFE) (000.00) 0.075344 0.990497 1.945746 34 (15,000.00) (000) 0.076084 0.958478 4.381435 64 (15,000.00, nan) (000) 0.076084 0.958478 4.381435 44 (15, nan) (000.00) 0.079380 0.951026 1.868209 (000.00) 0.058055 0.919825 1.806918 27 (nan, 10) 39 (15, nan) (000) 0.076084 0.911537 4.166858 **52** (nan, 000.00) (000)0.218527 0.429277 1.962328 33 (000, 000.00)(15) 0.076084 0.348167 4.171289 (000, nan, 000.00) (15) 0.076084 0.348167 4.171289 65 40 (000, nan) (15) 0.076084 0.347797 4.166858 46 (nan, 000.00) (15) 0.079380 0.155935 1.868209 56 (nan, 000.00) (FDRL/INTERNAL FUND TRANSFE) 0.075344 0.148006 1.945746 (nan, 000.00) (10) 0.058055 0.114043 1.806918 26 Learning Reflection on Association Rule Mining (ARM) Implementation Implementing Association Rule Mining (ARM) in Python using the Apriori algorithm provided a valuable learning experience, especially in the context of analyzing a bank transaction dataset. Here are the key reflections from this process: Understanding the dataset techniques.

• Initial Exploration: The initial steps involved loading and exploring the dataset. Understanding the structure and content of the dataset was crucial. The dataset contained columns like 'Account

No', 'DATE', 'TRANSACTION DETAILS', 'WITHDRAWAL AMT', 'DEPOSIT AMT', and 'BALANCE AMT'. This step highlighted the importance of data familiarity before applying any mining Data Preparation: Preparing the data by combining relevant columns into a single string per transaction was a critical step. This transformation made it possible to analyze the data in a meaningful way for ARM. The use of a lambda function to concatenate transaction details with withdrawal and deposit amounts demonstrated the need for creative data manipulation to fit the

algorithm's requirements.

Transforming the data TransactionEncoder: The use of TransactionEncoder from the mlxtend library was an enlightening experience. It transformed the list of transactions into a format suitable for the Apriori algorithm,

specifically a one-hot encoded DataFrame. This transformation was essential for identifying the presence of items in each transaction efficiently.

Applying the Apriori Algorithm Apriori Algorithm: Applying the Apriori algorithm to the transformed data was a core part of the process. This algorithm helped identify frequent itemsets based on a minimum support threshold. It was interesting to see how changing the support threshold could affect the number and type of itemsets generated.

• Frequent Itemsets: The algorithm's ability to discover frequent itemsets provided insight into the common combinations of transaction details and amounts in the dataset. Adding a length column to indicate the number of items in each itemset helped in further analysis and filtering.

**Generating Association Rules** 

• Association Rules: Generating association rules from the frequent itemsets was a crucial step. It involved setting a minimum lift threshold to ensure that the rules identified were significant. The

rules provided actionable insights into how certain transaction details were related to each other. • Filtering and Sorting Rules: Filtering rules based on antecedent and consequent lengths, and sorting by confidence and lift, highlighted the most interesting and useful patterns. This step

reinforced the importance of not just generating rules but also evaluating and interpreting them effectively.

The implementation of Association Rule Mining using the Apriori algorithm provided a comprehensive learning experience. It demonstrated the importance of data preparation, the intricacies of applying machine learning algorithms, and the significance of interpreting results to derive actionable insights. This project not only deepened my understanding of ARM but also showcased its