Assignment 3

In [2]:
```
import os
import warnings
warnings.filterwarnings("ignore")
grocery_path = r'Grocery_Items_10.csv'

import pandas as pd
grocery_data= pd.read_csv(grocery_path)
groceries_list= [line.dropna().tolist() for idx, line in grocery_data.iterrows()]

from mlxtend.preprocessing import TransactionEncoder
te = TransactionEncoder()
te_ary = te.fit(groceries_list).transform(groceries_list)
final_groceries_df = pd.DataFrame(te_ary, columns=te.columns_)

final_groceries_df.head(3)
```

Out[2]:

| | Instant food products | UHT-milk | abrasive cleaner | artif. sweetener | baby cosmetics | bags | baking powder | bathroom cleaner | beef | berries | ... | turkey | vinegar | waffles | whipped/sour cream | whisky | white bread | white wine | whole milk | yogurt | zwieback |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **1** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **2** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |

3 rows × 167 columns

## min_sup = 0.01 and min_conf 0.1

In [3]:
```
from mlxtend.frequent_patterns import apriori,association_rules
def gen_assoc_rules(df,i,j):
    frequent_items = apriori(df, min_support=i, use_colnames=True)
    ar=association_rules(frequent_items, metric="confidence", min_threshold=j)
    return ar
gen_assoc_rules(final_groceries_df,0.01,0.1)
```

Out[3]:

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
|---|---|---|---|---|---|---|---|---|---|
| **0** | (whole milk) | (other vegetables) | 0.157500 | 0.11975 | 0.016000 | 0.101587 | 0.848328 | -0.002861 | 0.979784 |
| **1** | (other vegetables) | (whole milk) | 0.119750 | 0.15750 | 0.016000 | 0.133612 | 0.848328 | -0.002861 | 0.972428 |
| **2** | (rolls/buns) | (whole milk) | 0.111625 | 0.15750 | 0.014375 | 0.128779 | 0.817647 | -0.003206 | 0.967034 |
| **3** | (soda) | (whole milk) | 0.098625 | 0.15750 | 0.012125 | 0.122940 | 0.780574 | -0.003408 | 0.960596 |
| **4** | (yogurt) | (whole milk) | 0.088750 | 0.15750 | 0.012125 | 0.136620 | 0.867427 | -0.001853 | 0.975816 |

## (msv): 0.001, 0.005, 0.01 and (mct): 0.05, 0.075, 0.1. For each pair (msv, mct), find the number of association rules

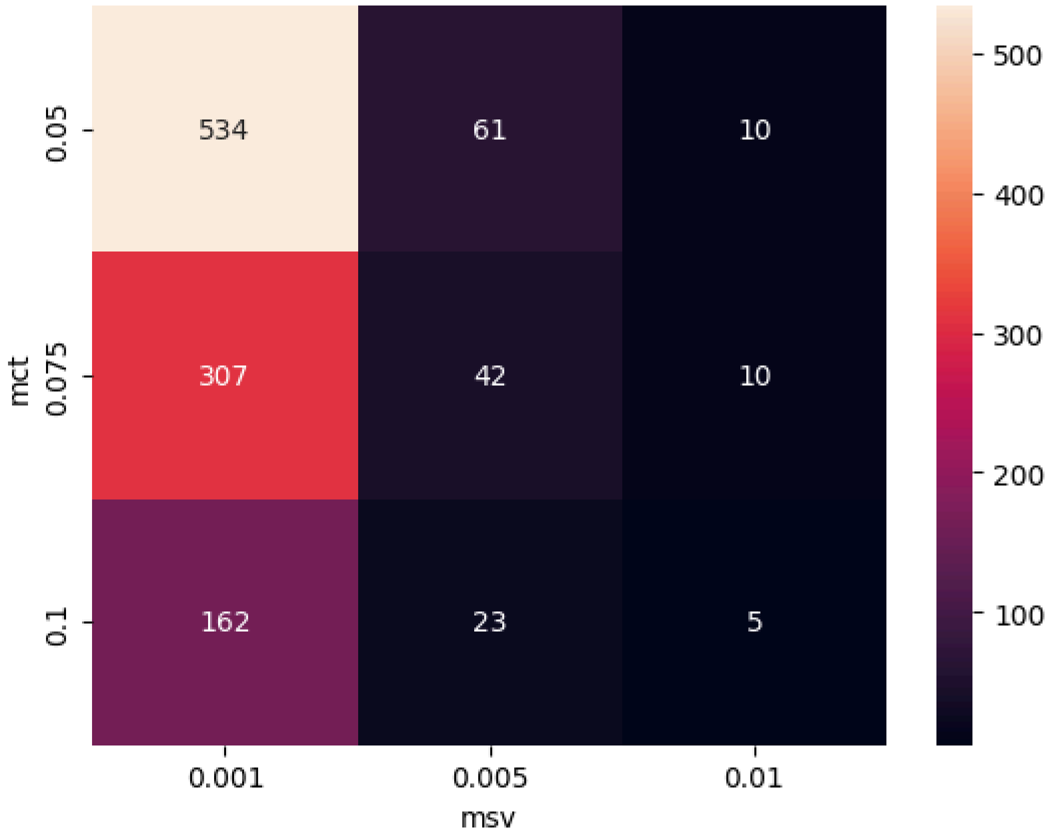
In [7]:

```
import seaborn as sns
msv =[0.001,0.005,0.01]
mct =[0.05,0.075,0.1]

pair_df = pd.DataFrame(columns=['msv', 'mct', 'count'])
for i in msv:
    for j in mct:
        pair_df = pair_df.append({'msv': i, 'mct': j, 'count': len(gen_assoc_rules(final_groceries_df,i,j))}, ignore_index=True)
heatmap = pair_df.pivot("mct", "msv", "count")
sns.heatmap(heatmap,annot=True,fmt=".0f")
```

Out[7]:

```
<AxesSubplot:xlabel='msv', ylabel='mct'>
```




### Split the dataset into 50:50 (i.e., 2 equal subsets) and extract association rules for each data subset for minimum support = 0.005 and minimum confident threshold = 0.075.

In [8]:
```
subset1 = final_groceries_df.iloc[:len(final_groceries_df)//2]
subset2 = final_groceries_df.iloc[len(final_groceries_df)//2:]
```

In [9]:
```
gen_assoc_rules(subset1,0.005,0.075)
```

Out[9]:

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
|---|---|---|---|---|---|---|---|---|---|
| **0** | (beef) | (whole milk) | 0.02900 | 0.15925 | 0.00500 | 0.172414 | 1.082661 | 0.000382 | 1.015906 |
| **1** | (bottled beer) | (whole milk) | 0.03900 | 0.15925 | 0.00825 | 0.211538 | 1.328342 | 0.002039 | 1.066317 |
| **2** | (bottled water) | (whole milk) | 0.05950 | 0.15925 | 0.00600 | 0.100840 | 0.633220 | -0.003475 | 0.935040 |

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
|---|---|---|---|---|---|---|---|---|---|
| **3** | (brown bread) | (whole milk) | 0.04025 | 0.15925 | 0.00500 | 0.124224 | 0.780054 | -0.001410 | 0.960005 |
| **4** | (citrus fruit) | (other vegetables) | 0.05625 | 0.12025 | 0.00525 | 0.093333 | 0.776161 | -0.001514 | 0.970313 |
| **5** | (citrus fruit) | (whole milk) | 0.05625 | 0.15925 | 0.00800 | 0.142222 | 0.893075 | -0.000958 | 0.980149 |
| **6** | (citrus fruit) | (yogurt) | 0.05625 | 0.08400 | 0.00525 | 0.093333 | 1.111111 | 0.000525 | 1.010294 |
| **7** | (domestic eggs) | (whole milk) | 0.03725 | 0.15925 | 0.00550 | 0.147651 | 0.927165 | -0.000432 | 0.986392 |
| **8** | (newspapers) | (whole milk) | 0.03625 | 0.15925 | 0.00500 | 0.137931 | 0.866129 | -0.000773 | 0.975270 |
| **9** | (rolls/buns) | (other vegetables) | 0.10725 | 0.12025 | 0.01025 | 0.095571 | 0.794770 | -0.002647 | 0.972713 |
| **10** | (other vegetables) | (rolls/buns) | 0.12025 | 0.10725 | 0.01025 | 0.085239 | 0.794770 | -0.002647 | 0.975938 |
| **11** | (sausage) | (other vegetables) | 0.05725 | 0.12025 | 0.00700 | 0.122271 | 1.016805 | 0.000116 | 1.002302 |
| **12** | (soda) | (other vegetables) | 0.09825 | 0.12025 | 0.01175 | 0.119593 | 0.994535 | -0.000065 | 0.999254 |
| **13** | (other vegetables) | (soda) | 0.12025 | 0.09825 | 0.01175 | 0.097713 | 0.994535 | -0.000065 | 0.999405 |
| **14** | (tropical fruit) | (other vegetables) | 0.07075 | 0.12025 | 0.00675 | 0.095406 | 0.793400 | -0.001758 | 0.972536 |
| **15** | (whole milk) | (other vegetables) | 0.15925 | 0.12025 | 0.01550 | 0.097331 | 0.809407 | -0.003650 | 0.974610 |
| **16** | (other vegetables) | (whole milk) | 0.12025 | 0.15925 | 0.01550 | 0.128898 | 0.809407 | -0.003650 | 0.965157 |
| **17** | (yogurt) | (other vegetables) | 0.08400 | 0.12025 | 0.00825 | 0.098214 | 0.816751 | -0.001851 | 0.975564 |
| **18** | (pastry) | (whole milk) | 0.04950 | 0.15925 | 0.00650 | 0.131313 | 0.824572 | -0.001383 | 0.967840 |
| **19** | (pip fruit) | (rolls/buns) | 0.04575 | 0.10725 | 0.00650 | 0.142077 | 1.324723 | 0.001593 | 1.040594 |
| **20** | (pip fruit) | (whole milk) | 0.04575 | 0.15925 | 0.00575 | 0.125683 | 0.789219 | -0.001536 | 0.961608 |
| **21** | (pork) | (whole milk) | 0.03700 | 0.15925 | 0.00650 | 0.175676 | 1.103144 | 0.000608 | 1.019926 |
| **22** | (root vegetables) | (rolls/buns) | 0.07225 | 0.10725 | 0.00600 | 0.083045 | 0.774312 | -0.001749 | 0.973603 |
| **23** | (soda) | (rolls/buns) | 0.09825 | 0.10725 | 0.00825 | 0.083969 | 0.782932 | -0.002287 | 0.974585 |
| **24** | (rolls/buns) | (soda) | 0.10725 | 0.09825 | 0.00825 | 0.076923 | 0.782932 | -0.002287 | 0.976896 |
| **25** | (whole milk) | (rolls/buns) | 0.15925 | 0.10725 | 0.01575 | 0.098901 | 0.922155 | -0.001330 | 0.990735 |
| **26** | (rolls/buns) | (whole milk) | 0.10725 | 0.15925 | 0.01575 | 0.146853 | 0.922155 | -0.001330 | 0.985469 |
| **27** | (yogurt) | (rolls/buns) | 0.08400 | 0.10725 | 0.00875 | 0.104167 | 0.971251 | -0.000259 | 0.996558 |

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
|---|---|---|---|---|---|---|---|---|---|
| **28** | (rolls/buns) | (yogurt) | 0.10725 | 0.08400 | 0.00875 | 0.081585 | 0.971251 | -0.000259 | 0.997371 |
| **29** | (root vegetables) | (whole milk) | 0.07225 | 0.15925 | 0.00875 | 0.121107 | 0.760485 | -0.002756 | 0.956601 |
| **30** | (root vegetables) | (yogurt) | 0.07225 | 0.08400 | 0.00625 | 0.086505 | 1.029824 | 0.000181 | 1.002742 |
| **31** | (sausage) | (soda) | 0.05725 | 0.09825 | 0.00500 | 0.087336 | 0.888919 | -0.000625 | 0.988042 |
| **32** | (sausage) | (whole milk) | 0.05725 | 0.15925 | 0.00750 | 0.131004 | 0.822633 | -0.001617 | 0.967496 |
| **33** | (sausage) | (yogurt) | 0.05725 | 0.08400 | 0.00500 | 0.087336 | 1.039717 | 0.000191 | 1.003656 |
| **34** | (shopping bags) | (whole milk) | 0.05000 | 0.15925 | 0.00725 | 0.145000 | 0.910518 | -0.000713 | 0.983333 |
| **35** | (tropical fruit) | (soda) | 0.07075 | 0.09825 | 0.00650 | 0.091873 | 0.935092 | -0.000451 | 0.992978 |
| **36** | (soda) | (whole milk) | 0.09825 | 0.15925 | 0.01075 | 0.109415 | 0.687063 | -0.004896 | 0.944042 |
| **37** | (tropical fruit) | (whole milk) | 0.07075 | 0.15925 | 0.00825 | 0.116608 | 0.732231 | -0.003017 | 0.951729 |
| **38** | (yogurt) | (whole milk) | 0.08400 | 0.15925 | 0.01100 | 0.130952 | 0.822307 | -0.002377 | 0.967438 |

In [10]:

```
gen_assoc_rules(subset2,0.005,0.075)
```

Out[10]:

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
|---|---|---|---|---|---|---|---|---|---|
| **0** | (bottled beer) | (whole milk) | 0.04525 | 0.15575 | 0.00800 | 0.176796 | 1.135124 | 0.000952 | 1.025565 |
| **1** | (bottled water) | (other vegetables) | 0.05725 | 0.11925 | 0.00525 | 0.091703 | 0.768998 | -0.001577 | 0.969672 |
| **2** | (bottled water) | (rolls/buns) | 0.05725 | 0.11600 | 0.00650 | 0.113537 | 0.978768 | -0.000141 | 0.997222 |
| **3** | (bottled water) | (soda) | 0.05725 | 0.09900 | 0.00600 | 0.104803 | 1.058621 | 0.000332 | 1.006483 |
| **4** | (bottled water) | (whole milk) | 0.05725 | 0.15575 | 0.00900 | 0.157205 | 1.009343 | 0.000083 | 1.001727 |
| **5** | (butter) | (whole milk) | 0.03675 | 0.15575 | 0.00575 | 0.156463 | 1.004575 | 0.000026 | 1.000845 |
| **6** | (canned beer) | (whole milk) | 0.04800 | 0.15575 | 0.00800 | 0.166667 | 1.070091 | 0.000524 | 1.013100 |
| **7** | (citrus fruit) | (other vegetables) | 0.05225 | 0.11925 | 0.00525 | 0.100478 | 0.842587 | -0.000981 | 0.979132 |
| **8** | (citrus fruit) | (whole milk) | 0.05225 | 0.15575 | 0.00725 | 0.138756 | 0.890889 | -0.000888 | 0.980268 |
| **9** | (citrus fruit) | (yogurt) | 0.05225 | 0.09350 | 0.00525 | 0.100478 | 1.074636 | 0.000365 | 1.007758 |
| **10** | (domestic eggs) | (whole milk) | 0.03675 | 0.15575 | 0.00575 | 0.156463 | 1.004575 | 0.000026 | 1.000845 |

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
|---|---|---|---|---|---|---|---|---|---|
| **11** | (frankfurter) | (other vegetables) | 0.03525 | 0.11925 | 0.00525 | 0.148936 | 1.248941 | 0.001046 | 1.034881 |
| **12** | (frankfurter) | (whole milk) | 0.03525 | 0.15575 | 0.00550 | 0.156028 | 1.001787 | 0.000010 | 1.000330 |
| **13** | (newspapers) | (whole milk) | 0.03800 | 0.15575 | 0.00625 | 0.164474 | 1.056011 | 0.000332 | 1.010441 |
| **14** | (rolls/buns) | (other vegetables) | 0.11600 | 0.11925 | 0.01125 | 0.096983 | 0.813273 | -0.002583 | 0.975341 |
| **15** | (other vegetables) | (rolls/buns) | 0.11925 | 0.11600 | 0.01125 | 0.094340 | 0.813273 | -0.002583 | 0.976083 |
| **16** | (sausage) | (other vegetables) | 0.06100 | 0.11925 | 0.00550 | 0.090164 | 0.756092 | -0.001774 | 0.968032 |
| **17** | (shopping bags) | (other vegetables) | 0.04650 | 0.11925 | 0.00500 | 0.107527 | 0.901693 | -0.000545 | 0.986864 |
| **18** | (tropical fruit) | (other vegetables) | 0.06825 | 0.11925 | 0.00700 | 0.102564 | 0.860076 | -0.001139 | 0.981407 |
| **19** | (whole milk) | (other vegetables) | 0.15575 | 0.11925 | 0.01650 | 0.105939 | 0.888377 | -0.002073 | 0.985112 |
| **20** | (other vegetables) | (whole milk) | 0.11925 | 0.15575 | 0.01650 | 0.138365 | 0.888377 | -0.002073 | 0.979823 |
| **21** | (yogurt) | (other vegetables) | 0.09350 | 0.11925 | 0.01025 | 0.109626 | 0.919293 | -0.000900 | 0.989191 |
| **22** | (other vegetables) | (yogurt) | 0.11925 | 0.09350 | 0.01025 | 0.085954 | 0.919293 | -0.000900 | 0.991744 |
| **23** | (pastry) | (whole milk) | 0.05300 | 0.15575 | 0.00650 | 0.122642 | 0.787425 | -0.001755 | 0.962263 |
| **24** | (pip fruit) | (whole milk) | 0.05125 | 0.15575 | 0.00725 | 0.141463 | 0.908272 | -0.000732 | 0.983359 |
| **25** | (pip fruit) | (yogurt) | 0.05125 | 0.09350 | 0.00500 | 0.097561 | 1.043433 | 0.000208 | 1.004500 |
| **26** | (root vegetables) | (rolls/buns) | 0.06800 | 0.11600 | 0.00600 | 0.088235 | 0.760649 | -0.001888 | 0.969548 |
| **27** | (shopping bags) | (rolls/buns) | 0.04650 | 0.11600 | 0.00525 | 0.112903 | 0.973304 | -0.000144 | 0.996509 |
| **28** | (soda) | (rolls/buns) | 0.09900 | 0.11600 | 0.00850 | 0.085859 | 0.740160 | -0.002984 | 0.967028 |
| **29** | (tropical fruit) | (rolls/buns) | 0.06825 | 0.11600 | 0.00600 | 0.087912 | 0.757863 | -0.001917 | 0.969205 |
| **30** | (whole milk) | (rolls/buns) | 0.15575 | 0.11600 | 0.01300 | 0.083467 | 0.719544 | -0.005067 | 0.964504 |
| **31** | (rolls/buns) | (whole milk) | 0.11600 | 0.15575 | 0.01300 | 0.112069 | 0.719544 | -0.005067 | 0.950806 |
| **32** | (yogurt) | (rolls/buns) | 0.09350 | 0.11600 | 0.00725 | 0.077540 | 0.668449 | -0.003596 | 0.958307 |
| **33** | (root vegetables) | (soda) | 0.06800 | 0.09900 | 0.00650 | 0.095588 | 0.965538 | -0.000232 | 0.996228 |
| **34** | (root vegetables) | (whole milk) | 0.06800 | 0.15575 | 0.00800 | 0.117647 | 0.755358 | -0.002591 | 0.956817 |
| **35** | (sausage) | (soda) | 0.06100 | 0.09900 | 0.00600 | 0.098361 | 0.993542 | -0.000039 | 0.999291 |

| | antecedents | consequents | antecedent support | consequent support | support | confidence | lift | leverage | conviction |
|---|---|---|---|---|---|---|---|---|---|
| **36** | (whole milk) | (sausage) | 0.15575 | 0.06100 | 0.01175 | 0.075441 | 1.236744 | 0.002249 | 1.015620 |
| **37** | (sausage) | (whole milk) | 0.06100 | 0.15575 | 0.01175 | 0.192623 | 1.236744 | 0.002249 | 1.045670 |
| **38** | (yogurt) | (sausage) | 0.09350 | 0.06100 | 0.00725 | 0.077540 | 1.271149 | 0.001547 | 1.017930 |
| **39** | (sausage) | (yogurt) | 0.06100 | 0.09350 | 0.00725 | 0.118852 | 1.271149 | 0.001547 | 1.028772 |
| **40** | (shopping bags) | (whole milk) | 0.04650 | 0.15575 | 0.00750 | 0.161290 | 1.035572 | 0.000258 | 1.006606 |
| **41** | (tropical fruit) | (soda) | 0.06825 | 0.09900 | 0.00600 | 0.087912 | 0.888001 | -0.000757 | 0.987843 |
| **42** | (soda) | (whole milk) | 0.09900 | 0.15575 | 0.01350 | 0.136364 | 0.875529 | -0.001919 | 0.977553 |
| **43** | (whole milk) | (soda) | 0.15575 | 0.09900 | 0.01350 | 0.086677 | 0.875529 | -0.001919 | 0.986508 |
| **44** | (tropical fruit) | (whole milk) | 0.06825 | 0.15575 | 0.00825 | 0.120879 | 0.776110 | -0.002380 | 0.960334 |
| **45** | (tropical fruit) | (yogurt) | 0.06825 | 0.09350 | 0.00525 | 0.076923 | 0.822707 | -0.001131 | 0.982042 |
| **46** | (yogurt) | (whole milk) | 0.09350 | 0.15575 | 0.01325 | 0.141711 | 0.909863 | -0.001313 | 0.983643 |
| **47** | (whole milk) | (yogurt) | 0.15575 | 0.09350 | 0.01325 | 0.085072 | 0.909863 | -0.001313 | 0.990789 |

In [11]:

```python
pd.merge(gen_assoc_rules(subset1,0.005,0.075), gen_assoc_rules(subset2,0.005,0.075),on=['antecedents', 'consequents'])
```

Out[11]:

| | antecedents | consequents | antecedent support_x | consequent support_x | support_x | confidence_x | lift_x | leverage_x | conviction_x | antecedent support_y | consequent support_y | support_y | confidence_y | lift_y | leverage_y | cc |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | (bottled beer) | (whole milk) | 0.03900 | 0.15925 | 0.00825 | 0.211538 | 1.328342 | 0.002039 | 1.066317 | 0.04525 | 0.15575 | 0.00800 | 0.176796 | 1.135124 | 0.000952 | |
| **1** | (bottled water) | (whole milk) | 0.05950 | 0.15925 | 0.00600 | 0.100840 | 0.633220 | -0.003475 | 0.935040 | 0.05725 | 0.15575 | 0.00900 | 0.157205 | 1.009343 | 0.000083 | |
| **2** | (citrus fruit) | (other vegetables) | 0.05625 | 0.12025 | 0.00525 | 0.093333 | 0.776161 | -0.001514 | 0.970313 | 0.05225 | 0.11925 | 0.00525 | 0.100478 | 0.842587 | -0.000981 | |
| **3** | (citrus fruit) | (whole milk) | 0.05625 | 0.15925 | 0.00800 | 0.142222 | 0.893075 | -0.000958 | 0.980149 | 0.05225 | 0.15575 | 0.00725 | 0.138756 | 0.890889 | -0.000888 | |
| **4** | (citrus fruit) | (yogurt) | 0.05625 | 0.08400 | 0.00525 | 0.093333 | 1.111111 | 0.000525 | 1.010294 | 0.05225 | 0.09350 | 0.00525 | 0.100478 | 1.074636 | 0.000365 | |
| **5** | (domestic eggs) | (whole milk) | 0.03725 | 0.15925 | 0.00550 | 0.147651 | 0.927165 | -0.000432 | 0.986392 | 0.03675 | 0.15575 | 0.00575 | 0.156463 | 1.004575 | 0.000026 | |
| **6** | (newspapers) | (whole milk) | 0.03625 | 0.15925 | 0.00500 | 0.137931 | 0.866129 | -0.000773 | 0.975270 | 0.03800 | 0.15575 | 0.00625 | 0.164474 | 1.056011 | 0.000332 | |

| | antecedents | consequents | antecedent support_x | consequent support_x | support_x | confidence_x | lift_x | leverage_x | conviction_x | antecedent support_y | consequent support_y | support_y | confidence_y | lift_y | leverage_y | cc |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **7** | (rolls/buns) | (other vegetables) | 0.10725 | 0.12025 | 0.01025 | 0.095571 | 0.794770 | -0.002647 | 0.972713 | 0.11600 | 0.11925 | 0.01125 | 0.096983 | 0.813273 | -0.002583 | |
| **8** | (other vegetables) | (rolls/buns) | 0.12025 | 0.10725 | 0.01025 | 0.085239 | 0.794770 | -0.002647 | 0.975938 | 0.11925 | 0.11600 | 0.01125 | 0.094340 | 0.813273 | -0.002583 | |
| **9** | (sausage) | (other vegetables) | 0.05725 | 0.12025 | 0.00700 | 0.122271 | 1.016805 | 0.000116 | 1.002302 | 0.06100 | 0.11925 | 0.00550 | 0.090164 | 0.756092 | -0.001774 | |
| **10** | (tropical fruit) | (other vegetables) | 0.07075 | 0.12025 | 0.00675 | 0.095406 | 0.793400 | -0.001758 | 0.972536 | 0.06825 | 0.11925 | 0.00700 | 0.102564 | 0.860076 | -0.001139 | |
| **11** | (whole milk) | (other vegetables) | 0.15925 | 0.12025 | 0.01550 | 0.097331 | 0.809407 | -0.003650 | 0.974610 | 0.15575 | 0.11925 | 0.01650 | 0.105939 | 0.888377 | -0.002073 | |
| **12** | (other vegetables) | (whole milk) | 0.12025 | 0.15925 | 0.01550 | 0.128898 | 0.809407 | -0.003650 | 0.965157 | 0.11925 | 0.15575 | 0.01650 | 0.138365 | 0.888377 | -0.002073 | |
| **13** | (yogurt) | (other vegetables) | 0.08400 | 0.12025 | 0.00825 | 0.098214 | 0.816751 | -0.001851 | 0.975564 | 0.09350 | 0.11925 | 0.01025 | 0.109626 | 0.919293 | -0.000900 | |
| **14** | (pastry) | (whole milk) | 0.04950 | 0.15925 | 0.00650 | 0.131313 | 0.824572 | -0.001383 | 0.967840 | 0.05300 | 0.15575 | 0.00650 | 0.122642 | 0.787425 | -0.001755 | |
| **15** | (pip fruit) | (whole milk) | 0.04575 | 0.15925 | 0.00575 | 0.125683 | 0.789219 | -0.001536 | 0.961608 | 0.05125 | 0.15575 | 0.00725 | 0.141463 | 0.908272 | -0.000732 | |
| **16** | (root vegetables) | (rolls/buns) | 0.07225 | 0.10725 | 0.00600 | 0.083045 | 0.774312 | -0.001749 | 0.973603 | 0.06800 | 0.11600 | 0.00600 | 0.088235 | 0.760649 | -0.001888 | |
| **17** | (soda) | (rolls/buns) | 0.09825 | 0.10725 | 0.00825 | 0.083969 | 0.782932 | -0.002287 | 0.974585 | 0.09900 | 0.11600 | 0.00850 | 0.085859 | 0.740160 | -0.002984 | |
| **18** | (whole milk) | (rolls/buns) | 0.15925 | 0.10725 | 0.01575 | 0.098901 | 0.922155 | -0.001330 | 0.990735 | 0.15575 | 0.11600 | 0.01300 | 0.083467 | 0.719544 | -0.005067 | |
| **19** | (rolls/buns) | (whole milk) | 0.10725 | 0.15925 | 0.01575 | 0.146853 | 0.922155 | -0.001330 | 0.985469 | 0.11600 | 0.15575 | 0.01300 | 0.112069 | 0.719544 | -0.005067 | |
| **20** | (yogurt) | (rolls/buns) | 0.08400 | 0.10725 | 0.00875 | 0.104167 | 0.971251 | -0.000259 | 0.996558 | 0.09350 | 0.11600 | 0.00725 | 0.077540 | 0.668449 | -0.003596 | |
| **21** | (root vegetables) | (whole milk) | 0.07225 | 0.15925 | 0.00875 | 0.121107 | 0.760485 | -0.002756 | 0.956601 | 0.06800 | 0.15575 | 0.00800 | 0.117647 | 0.755358 | -0.002591 | |
| **22** | (sausage) | (soda) | 0.05725 | 0.09825 | 0.00500 | 0.087336 | 0.888919 | -0.000625 | 0.988042 | 0.06100 | 0.09900 | 0.00600 | 0.098361 | 0.993542 | -0.000039 | |
| **23** | (sausage) | (whole milk) | 0.05725 | 0.15925 | 0.00750 | 0.131004 | 0.822633 | -0.001617 | 0.967496 | 0.06100 | 0.15575 | 0.01175 | 0.192623 | 1.236744 | 0.002249 | |
| **24** | (sausage) | (yogurt) | 0.05725 | 0.08400 | 0.00500 | 0.087336 | 1.039717 | 0.000191 | 1.003656 | 0.06100 | 0.09350 | 0.00725 | 0.118852 | 1.271149 | 0.001547 | |
| **25** | (shopping bags) | (whole milk) | 0.05000 | 0.15925 | 0.00725 | 0.145000 | 0.910518 | -0.000713 | 0.983333 | 0.04650 | 0.15575 | 0.00750 | 0.161290 | 1.035572 | 0.000258 | |

| | antecedents | consequents | antecedent support_x | consequent support_x | support_x | confidence_x | lift_x | leverage_x | conviction_x | antecedent support_y | consequent support_y | support_y | confidence_y | lift_y | leverage_y | cc |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **26** | (tropical fruit) | (soda) | 0.07075 | 0.09825 | 0.00650 | 0.091873 | 0.935092 | -0.000451 | 0.992978 | 0.06825 | 0.09900 | 0.00600 | 0.087912 | 0.888001 | -0.000757 | |
| **27** | (soda) | (whole milk) | 0.09825 | 0.15925 | 0.01075 | 0.109415 | 0.687063 | -0.004896 | 0.944042 | 0.09900 | 0.15575 | 0.01350 | 0.136364 | 0.875529 | -0.001919 | |
| **28** | (tropical fruit) | (whole milk) | 0.07075 | 0.15925 | 0.00825 | 0.116608 | 0.732231 | -0.003017 | 0.951729 | 0.06825 | 0.15575 | 0.00825 | 0.120879 | 0.776110 | -0.002380 | |
| **29** | (yogurt) | (whole milk) | 0.08400 | 0.15925 | 0.01100 | 0.130952 | 0.822307 | -0.002377 | 0.967438 | 0.09350 | 0.15575 | 0.01325 | 0.141711 | 0.909863 | -0.001313 | |

## Image processing, getting images and labels

In [20]:

```
from glob import glob
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.image import ImageDataGenerator

dog_images=r'output folder'

hist_images = []
labels = []
for index,breed in enumerate(os.listdir(dog_images)):
    image_folder=os.path.join(dog_images,breed)
    images=glob(os.path.join(image_folder, '*.jpg'))
    hist_images.extend(images)
    labels.extend([breed] * len(images))

dog_df = pd.DataFrame({'image_path': hist_images, 'breed': labels})

training_data, validation_data = train_test_split(dog_df, test_size=0.2, random_state=42)

train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2)
val_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2)

train = train_datagen.flow_from_dataframe(training_data,x_col='image_path',y_col='breed',target_size=(128, 128),batch_size=16,class_mode='categorical')
validation = val_datagen.flow_from_dataframe(validation_data,x_col='image_path',y_col='breed',target_size=(128, 128),batch_size=16,class_mode='categorical')
```

```
Found 556 validated image filenames belonging to 4 classes.
Found 140 validated image filenames belonging to 4 classes.
```

## classification model as per the given parameters

In [27]:

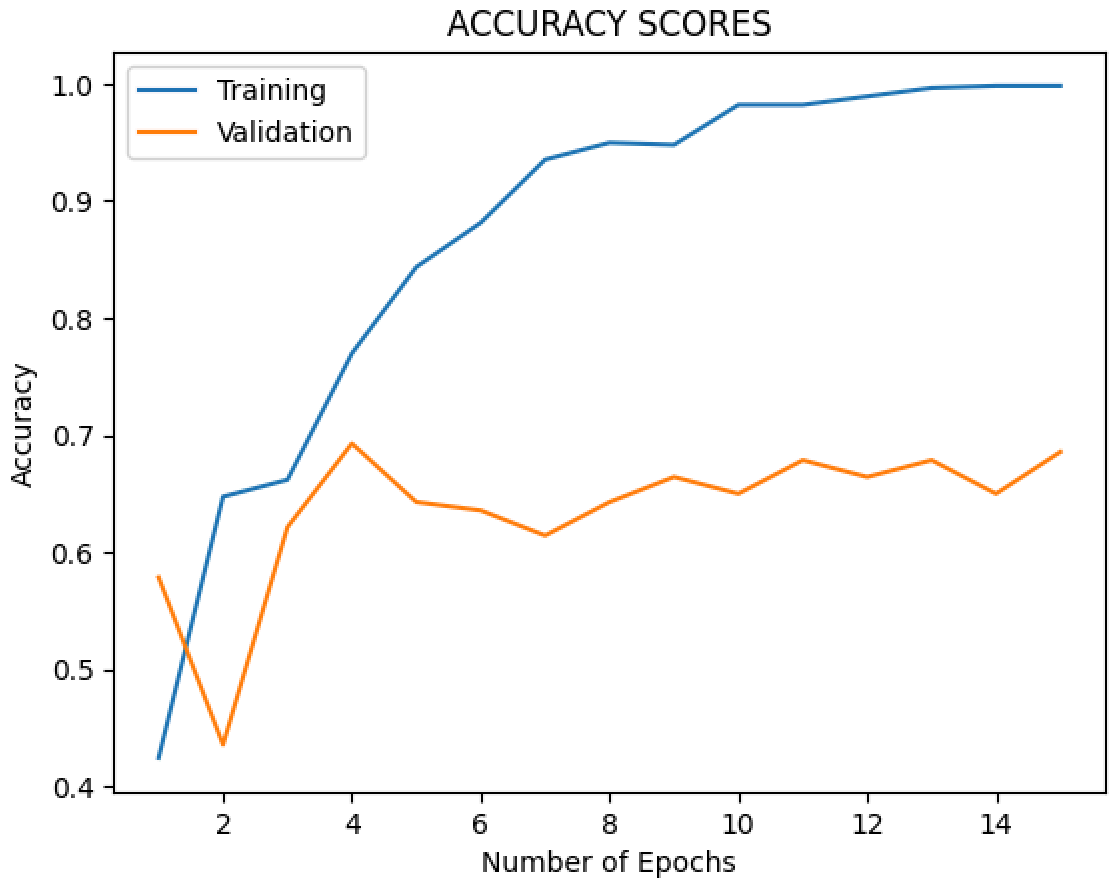
```
from tensorflow.keras.layers import Dense,Conv2D,Flatten,MaxPooling2D
from tensorflow.keras.models import Sequential
import matplotlib.pyplot as plt

warnings.filterwarnings("ignore", category=DeprecationWarning)


model=Sequential([
    Conv2D(8,(3,3),activation='relu',input_shape = (128,128,3)),
    MaxPooling2D(pool_size=(2,2)) ,
    Flatten(),
    Dense(16,activation='relu'),
    Dense(4,activation = 'softmax')
    ])
model.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(train,steps_per_epoch=len(train),epochs=15,validation_data=validation,validation_steps=len(validation)    )
training = history.history['accuracy']
validate = history.history['val_accuracy']
epochs = range(1, len(training) + 1)
plt.plot(epochs, training , label='Training')
plt.plot(epochs, validate, label='Validation')
plt.title("ACCURACY SCORES")
plt.xlabel('Number of Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

```
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
Train for 35 steps, validate for 9 steps
Epoch 1/15
35/35 [==============================] - 4s 118ms/step - loss: 1.5857 - accuracy: 0.4245 - val_loss: 1.0482 - val_accuracy: 0.5786
Epoch 2/15
35/35 [==============================] - 4s 105ms/step - loss: 0.8763 - accuracy: 0.6475 - val_loss: 1.2784 - val_accuracy: 0.4357
Epoch 3/15
35/35 [==============================] - 4s 105ms/step - loss: 0.7887 - accuracy: 0.6619 - val_loss: 1.0064 - val_accuracy: 0.6214
Epoch 4/15
```

```
35/35 [==============================] - 4s 106ms/step - loss: 0.6465 - accuracy: 0.7698 - val_loss: 0.8478 - val_accuracy: 0.6929
Epoch 5/15
35/35 [==============================] - 4s 106ms/step - loss: 0.4626 - accuracy: 0.8435 - val_loss: 0.8863 - val_accuracy: 0.6429
Epoch 6/15
35/35 [==============================] - 4s 120ms/step - loss: 0.4175 - accuracy: 0.8813 - val_loss: 0.9007 - val_accuracy: 0.6357
Epoch 7/15
35/35 [==============================] - 4s 119ms/step - loss: 0.2904 - accuracy: 0.9353 - val_loss: 0.8835 - val_accuracy: 0.6143
Epoch 8/15
35/35 [==============================] - 4s 121ms/step - loss: 0.2479 - accuracy: 0.9496 - val_loss: 0.9372 - val_accuracy: 0.6429
Epoch 9/15
35/35 [==============================] - 4s 111ms/step - loss: 0.2088 - accuracy: 0.9478 - val_loss: 0.8572 - val_accuracy: 0.6643
Epoch 10/15
35/35 [==============================] - 4s 105ms/step - loss: 0.1525 - accuracy: 0.9820 - val_loss: 0.8845 - val_accuracy: 0.6500
Epoch 11/15
35/35 [==============================] - 5s 130ms/step - loss: 0.1269 - accuracy: 0.9820 - val_loss: 0.8881 - val_accuracy: 0.6786
Epoch 12/15
35/35 [==============================] - 4s 122ms/step - loss: 0.1043 - accuracy: 0.9892 - val_loss: 1.0284 - val_accuracy: 0.6643
Epoch 13/15
35/35 [==============================] - 4s 107ms/step - loss: 0.0772 - accuracy: 0.9964 - val_loss: 0.9741 - val_accuracy: 0.6786
Epoch 14/15
35/35 [==============================] - 4s 114ms/step - loss: 0.0557 - accuracy: 0.9982 - val_loss: 0.9564 - val_accuracy: 0.6500
Epoch 15/15
35/35 [==============================] - 4s 106ms/step - loss: 0.0451 - accuracy: 0.9982 - val_loss: 0.9903 - val_accuracy: 0.6857
```




## 916461327

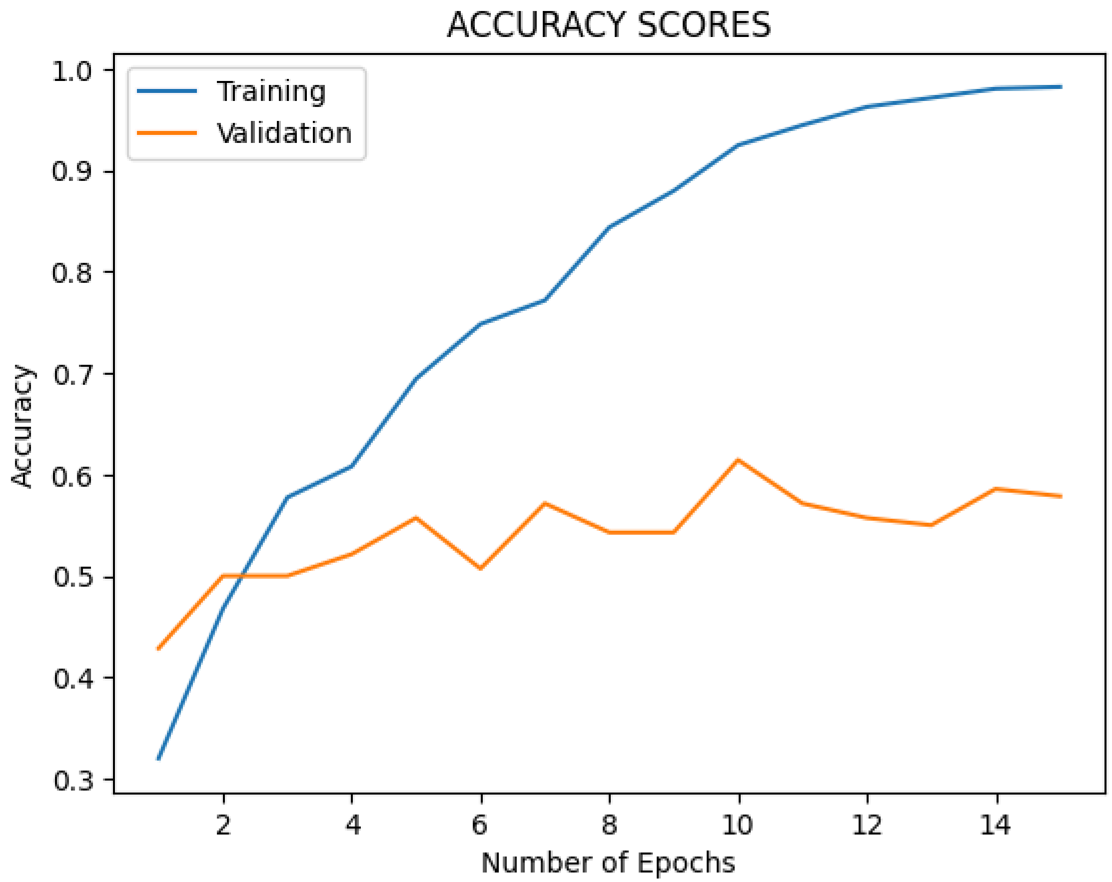
In [31]:

```
model=Sequential([
    Conv2D(8,(3,3),activation='relu',input_shape = (128,128,3)),
    MaxPooling2D(pool_size=(2,2)) ,
    Flatten(),
    Dense(8,activation='relu'),
    Dense(4,activation = 'softmax')
    ])
model.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(train,steps_per_epoch=len(train),epochs=15,validation_data=validation,validation_steps=len(validation)    )
training = history.history['accuracy']
validate = history.history['val_accuracy']
epochs = range(1, len(training) + 1)
plt.plot(epochs, training , label='Training')
```

```
plt.plot(epochs, validate, label='Validation')
plt.title("ACCURACY SCORES")
plt.xlabel('Number of Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

```
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
Train for 35 steps, validate for 9 steps
Epoch 1/15
35/35 [==============================] - 4s 115ms/step - loss: 1.3761 - accuracy: 0.3201 - val_loss: 1.2769 - val_accuracy: 0.4286
Epoch 2/15
35/35 [==============================] - 4s 112ms/step - loss: 1.1586 - accuracy: 0.4676 - val_loss: 1.0770 - val_accuracy: 0.5000
Epoch 3/15
35/35 [==============================] - 4s 118ms/step - loss: 0.9557 - accuracy: 0.5773 - val_loss: 1.1336 - val_accuracy: 0.5000
Epoch 4/15
35/35 [==============================] - 4s 118ms/step - loss: 0.8792 - accuracy: 0.6079 - val_loss: 0.9820 - val_accuracy: 0.5214
Epoch 5/15
35/35 [==============================] - 4s 113ms/step - loss: 0.6651 - accuracy: 0.6942 - val_loss: 0.9597 - val_accuracy: 0.5571
Epoch 6/15
35/35 [==============================] - 4s 113ms/step - loss: 0.5727 - accuracy: 0.7482 - val_loss: 1.0857 - val_accuracy: 0.5071
Epoch 7/15
35/35 [==============================] - 4s 118ms/step - loss: 0.4737 - accuracy: 0.7716 - val_loss: 0.9823 - val_accuracy: 0.5714
Epoch 8/15
35/35 [==============================] - 4s 119ms/step - loss: 0.4129 - accuracy: 0.8435 - val_loss: 1.2853 - val_accuracy: 0.5429
Epoch 9/15
35/35 [==============================] - 4s 112ms/step - loss: 0.3999 - accuracy: 0.8795 - val_loss: 1.1484 - val_accuracy: 0.5429
Epoch 10/15
35/35 [==============================] - 4s 117ms/step - loss: 0.3042 - accuracy: 0.9245 - val_loss: 1.0337 - val_accuracy: 0.6143
Epoch 11/15
35/35 [==============================] - 4s 115ms/step - loss: 0.2475 - accuracy: 0.9442 - val_loss: 1.2213 - val_accuracy: 0.5714
Epoch 12/15
35/35 [==============================] - 4s 113ms/step - loss: 0.1959 - accuracy: 0.9622 - val_loss: 1.4754 - val_accuracy: 0.5571
Epoch 13/15
35/35 [==============================] - 4s 116ms/step - loss: 0.1575 - accuracy: 0.9712 - val_loss: 1.4844 - val_accuracy: 0.5500
Epoch 14/15
35/35 [==============================] - 4s 116ms/step - loss: 0.1325 - accuracy: 0.9802 - val_loss: 1.3893 - val_accuracy: 0.5857
```

```
Epoch 15/15
35/35 [==============================] - 4s 115ms/step - loss: 0.0991 - accuracy: 0.9820 - val_loss: 1.6109 - val_accuracy: 0.5786
```



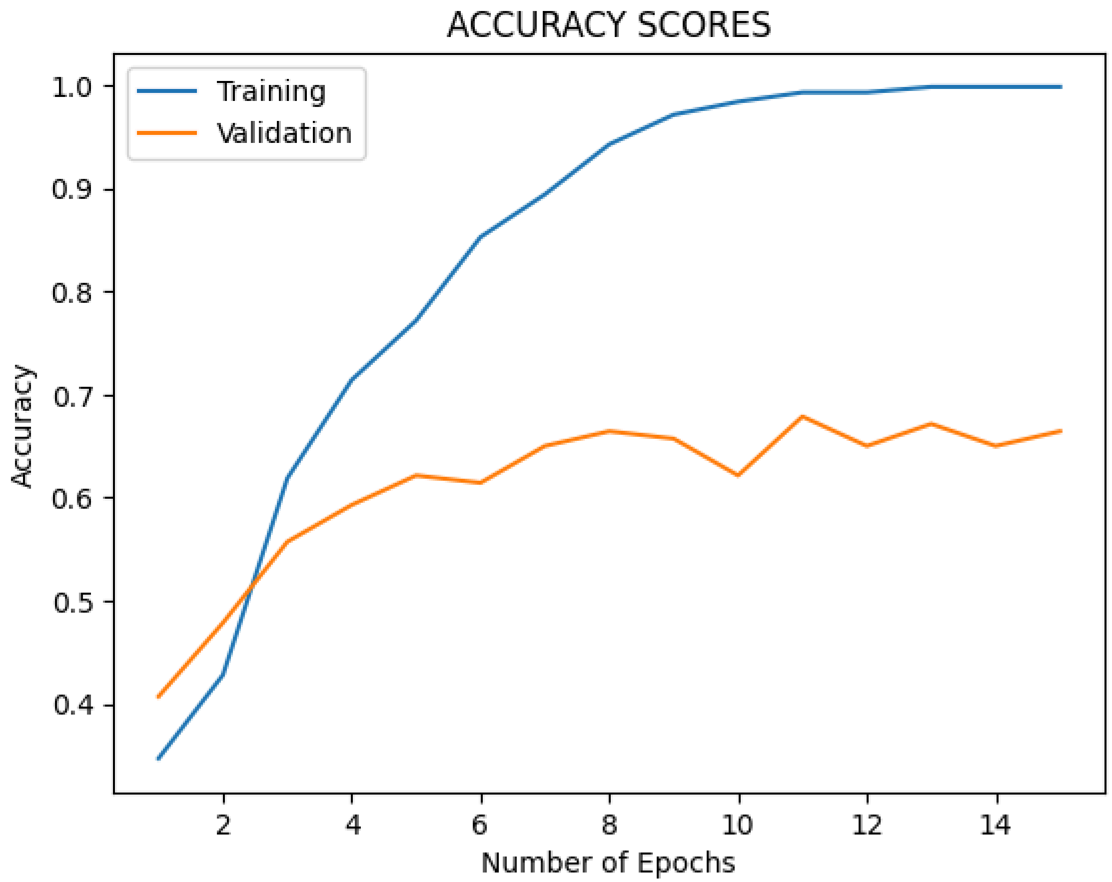

In [30]:

```
model=Sequential([
    Conv2D(8,(3,3),activation='relu',input_shape = (128,128,3)),
    MaxPooling2D(pool_size=(2,2)) ,
    Flatten(),
    Dense(32,activation='relu'),
    Dense(4,activation = 'softmax')
    ])
model.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(train,steps_per_epoch=len(train),epochs=15,validation_data=validation,validation_steps=len(validation)    )
training = history.history['accuracy']
validate = history.history['val_accuracy']
epochs = range(1, len(training) + 1)
plt.plot(epochs, training , label='Training')
```

```
plt.plot(epochs, validate, label='Validation')
plt.title("ACCURACY SCORES")
plt.xlabel('Number of Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

```
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
Train for 35 steps, validate for 9 steps
Epoch 1/15
35/35 [==============================] - 4s 114ms/step - loss: 1.7120 - accuracy: 0.3471 - val_loss: 1.2061 - val_accuracy: 0.4071
Epoch 2/15
35/35 [==============================] - 4s 121ms/step - loss: 1.1592 - accuracy: 0.4281 - val_loss: 1.1586 - val_accuracy: 0.4786
Epoch 3/15
35/35 [==============================] - 4s 116ms/step - loss: 0.9636 - accuracy: 0.6187 - val_loss: 1.2392 - val_accuracy: 0.5571
Epoch 4/15
35/35 [==============================] - 4s 120ms/step - loss: 0.7744 - accuracy: 0.7140 - val_loss: 0.9795 - val_accuracy: 0.5929
Epoch 5/15
35/35 [==============================] - 4s 116ms/step - loss: 0.6030 - accuracy: 0.7716 - val_loss: 0.9346 - val_accuracy: 0.6214
Epoch 6/15
35/35 [==============================] - 4s 111ms/step - loss: 0.4527 - accuracy: 0.8525 - val_loss: 0.8945 - val_accuracy: 0.6143
Epoch 7/15
35/35 [==============================] - 4s 120ms/step - loss: 0.3734 - accuracy: 0.8939 - val_loss: 0.9114 - val_accuracy: 0.6500
Epoch 8/15
35/35 [==============================] - 4s 111ms/step - loss: 0.2620 - accuracy: 0.9424 - val_loss: 0.8783 - val_accuracy: 0.6643
Epoch 9/15
35/35 [==============================] - 4s 127ms/step - loss: 0.1883 - accuracy: 0.9712 - val_loss: 0.9116 - val_accuracy: 0.6571
Epoch 10/15
35/35 [==============================] - 4s 119ms/step - loss: 0.1300 - accuracy: 0.9838 - val_loss: 1.0538 - val_accuracy: 0.6214
Epoch 11/15
35/35 [==============================] - 4s 123ms/step - loss: 0.1009 - accuracy: 0.9928 - val_loss: 0.9435 - val_accuracy: 0.6786
Epoch 12/15
35/35 [==============================] - 4s 120ms/step - loss: 0.0739 - accuracy: 0.9928 - val_loss: 0.9438 - val_accuracy: 0.6500
Epoch 13/15
35/35 [==============================] - 4s 125ms/step - loss: 0.0483 - accuracy: 0.9982 - val_loss: 0.9634 - val_accuracy: 0.6714
Epoch 14/15
35/35 [==============================] - 4s 127ms/step - loss: 0.0384 - accuracy: 0.9982 - val_loss: 0.9903 - val_accuracy: 0.6500
```

```
Epoch 15/15
35/35 [==============================] - 4s 113ms/step - loss: 0.0312 - accuracy: 0.9982 - val_loss: 1.0859 - val_accuracy: 0.6643
```




3 models are overfitting Model 1 generally outperforms Model 2,3 in terms of both training and validation accuracy. model 3 is better than model 2.

In [ ]:

```
References :        https://seaborn.pydata.org/generated/seaborn.heatmap.html
            http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/
```