Master of Management in Artificial Intelligence

MMAI 869

Machine Learning and AI

Dr. Stephen W. Thomas

Individual Assignment 1

November 17th, 2019

He Li (Jimi)

Question 1:

After each feature of the dataset is standardized, I have tried 3 algorithms and fine tuned each one by comparing silhouette scores. The table below is a summary of what I tried and the parameters I've decided to use for each model.

		Finalized Model		
Algorithm	**Parameters Tuned**	**Parameter Selected**	**Silhouette Score**	**Number of Clusters**
K-means	K (from 2 to 20)	K = 5	0.805	5
DBSCAN	minipts (from 3 to 6) eps (from 0.1 to 1) distance Metric (cityblock, cosine, euclidean, l1, l2, manhattan)	minipts = 9 eps = 0.2 distance Metric = cosine	0.922	4
Hierarchical	distance Metric (euclidean, l1, l2, manhattan, cosine) linkage (ward, complete, average, single) n_clusters (from 2 to 10)	distance Metric = cosine linkage = average n_clusters = 5	0.931	5

Although I noticed Income/Age and Savings/SpendingScore are highly correlated feature pairs, I have decided not to remove any features because:

- Silhouette Scores are very high even when using all features
- We don't want to potentially lose understanding of our customers because
    - Income and age doesn't make sense to always be correlated
    - Savings and spending behavior doesn't make sense to always be correlated

Interestingly, I've noticed that the cluster means I've got from the best K-means and the best Hierarchical model are exactly the same in these aspects:

- the number of unique clusters
- the feature means for each cluster
- the size of each cluster

This finding together with the fact that Hierarchical has the highest Silhouettes Score(0.931) is what convinced me to select Hierarchical Clustering with distance Metric = cosine, linkage = average, and n_clusters = 5

After the model and its parameters are chosen, I have then printed the cluster means from the Hierarchical model I've chosen, and I've labeled/described each customer persona:

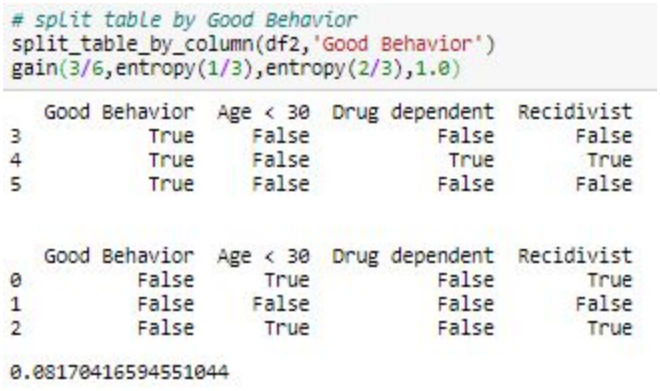
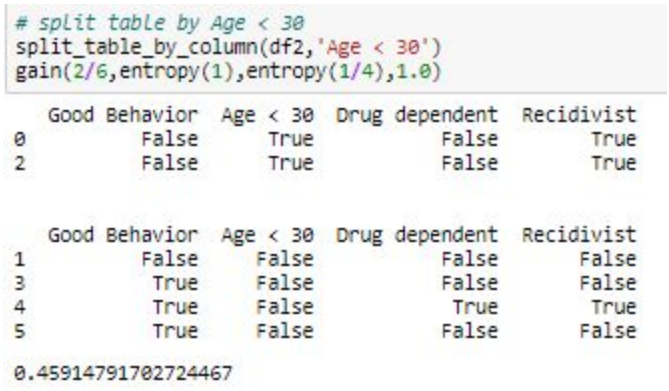
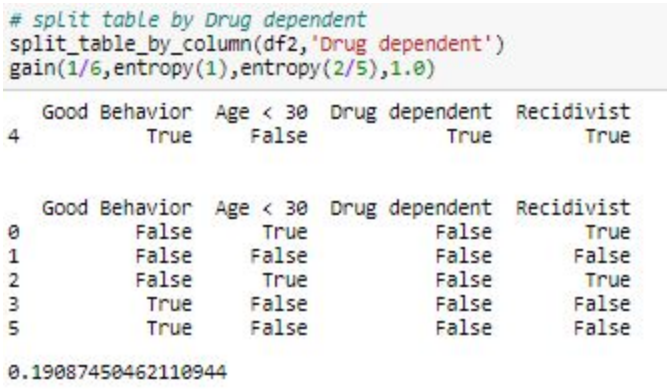
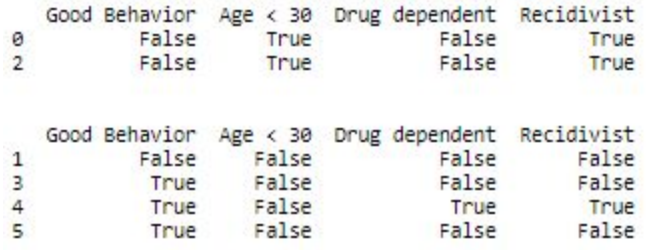
Description	Age	Income	Spending Score	Savings	%
Young Spendthrift High-Income Professionals with Low Savings	24.18	128029.12	0.90	4087.52	10
Young High-Income Professionals with High Savings	32.78	105265.81	0.31	14962.78	25
Mid-aged Spendthrift Professionals	59.96	72448.06	0.77	6889.97	31
High-Income Frugal Seniors with High Savings	86.00	119944.04	0.07	14808.68	5
Retired Seniors with High Savings	87.78	27866.10	0.33	16659.26	29

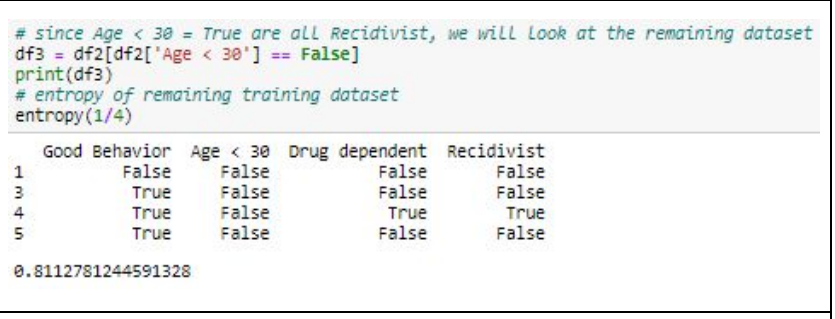
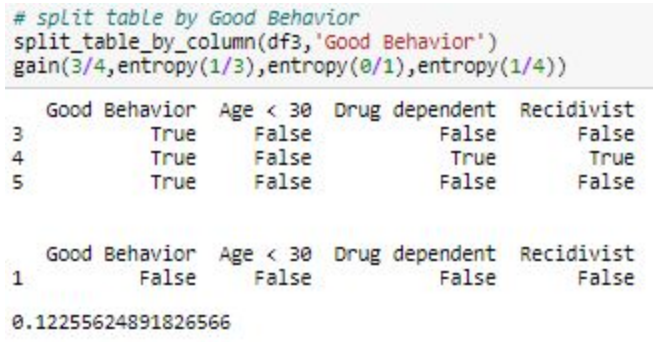
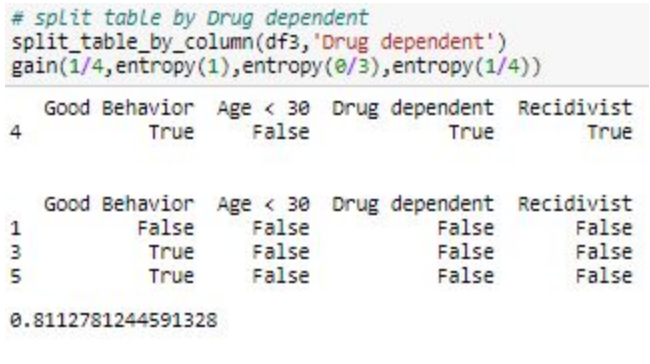
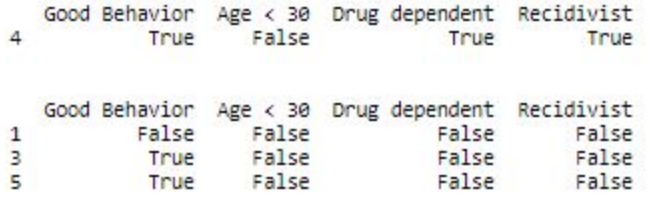
With a high Silhouettes Score(0.931), and 5 distinct, fairly evenly distributed, and meaningful customer personas. I believe the results are very good and will be very impactful when Marketing team is trying to deploy tailored advertising campaigns for each of the customer personas!

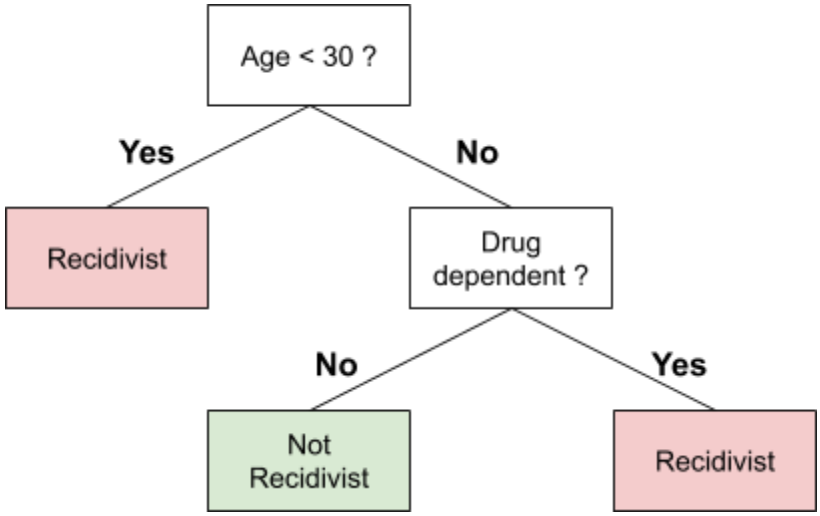
Question 2:

I have constructed an ID3 decision tree “by hand” on Python while only using pandas and numpy library, please refer to the Jupyter Notebook file for more details:

Step	Description	Result
1	Made a helper function that calculates entropy	`# the function below calculates the entropy with input x as the % of a "True" Target` `def entropy(x):` `  if x == 1 or x == 0:` `  return 0.0` `  else:` `  return -x*np.log2(x)-(1-x)*np.log2(1-x)`
2	Made a helper function that calculates gain	`# the function below returns gain based on inputs` `def gain(p_true,e_true,e_false,original_gain):` `  p_false = 1 - p_true` `  return original_gain - (p_true * e_true + p_false * e_false)`
3	Made a helper function that splits and displays any table by any one particular column name	`# the function below prints the splited table based on column_name input` `def split_table_by_column(df,column_name):` `  print(df[df[column_name] == True])` `  print('\n')` `  print(df[df[column_name] == False])`
4	Calculate the entropy of the entire dataset (**1.0**)	`# training data imported as a Pandas Dataframe` `data2 = {'Good Behavior': [False, False, False, True, True, True],` `  'Age < 30': [True, False, True, False, False, False],` `  'Drug dependent': [False, False, False, False, True, False],` `  'Recidivist': [True, False, True, False, True, False],` `  }` `df2 = pd.DataFrame(data=data2)` `print(df2)` `# entropy of entire training dataset` `entropy(3/6)` `  Good Behavior  Age < 30  Drug dependent  Recidivist` `0  False  True  False  True` `1  False  False  False  False` `2  False  True  False  True` `3  True  False  False  False` `4  True  False  True  True` `5  True  False  False  False` `1.0`

5	Splitting entire dataset by “Good Behavior” and calculate its gain (**0.082**)	
6	Splitting entire dataset by “Age < 30” and calculate its gain (**0.459**)	
7	Splitting entire dataset by “Drug dependent” and calculate its gain (**0.191**)	
8	Since splitting the remaining dataset by “Age < 30” gives the highest gain(**0.459**), <u>we first split by “Age < 30”</u>. Also,since Age < 30 = True are all Recidivists, we will look at the remaining	

	dataset where Age < 30 = False	
9	Calculate the entropy of the remaining dataset (where Age < 30 = False) (**0.811**)	
10	Splitting the remaining dataset by “Good Behavior” and calculate its gain (**0.123**)	
11	Splitting the remaining dataset by “Drug dependent” and calculate its gain (**0.811**)	
12	Since splitting the remaining dataset by “Drug dependent” gives the highest gain(**0.811**), <u>we secondly split by “Drug dependent”</u>. Also,since Drug dependent = True are all Recidivists and Drug dependent = False are all non-Recidivist. This	

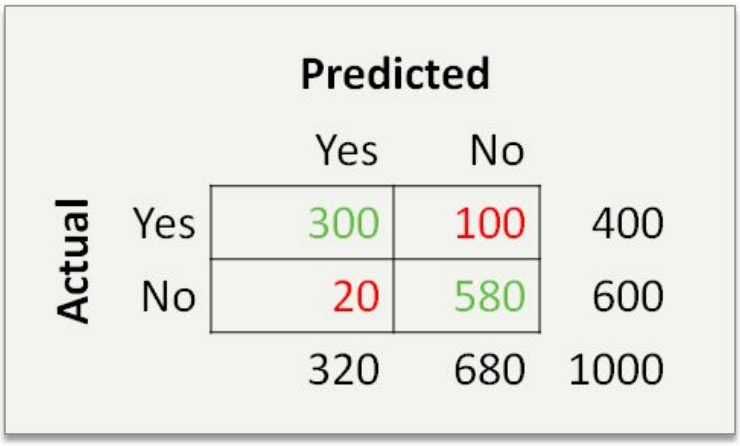
	concludes the algorithm.	
13	Here is the final decision tree	

Prediction of prisoner (Good Behavior = false, Age < 30 = false, Drug dependent = true) is Recidivist when following the decision tree above.

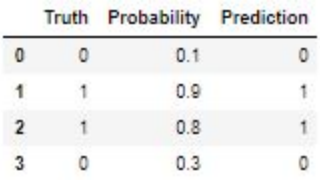
I have also confirmed my decision tree prediction by running the same prediction on a trained classifier with sklean's DecisionTreeClassifier and the result matches with what I've found above (Recidivist):

Question 3:

The Classifier Performance Measures taught in class includes: Accuracy, Error, Specificity, Precision, Recall / Sensitivity, F1, ROC Curve / AUC, Log Loss. I will introduce 3 additional measures below:

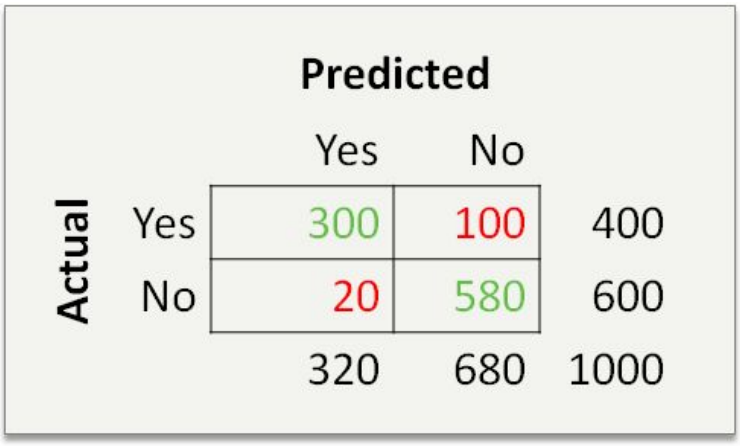
Classification Measure	**Jaccard similarity coefficient score (JSCS)**
What it is?	Jaccard similarity coefficient measures the size of intersection divided by the size of the union of the sample sets. Best value is 1 and worst value is 0.
How it is calculated (2-label sets)?	JSCS = (TP / (TP + FN + FP) + TN / (TN + FN + FP)) / 2
How it is different from Accuracy?	Accuracy = (TP +TN) / (TP + TN + FP + FN)
How it is different from Accuracy (Example)	 Accuracy = 880 / 1000 = 0.88 JSCS = (300 / 420 + 580 / 700) / 2 = 0.77
Which scenarios it is best used?	Whenever we want the sample sets to be as similar as possible, while the impact of False Negative and False Positive are equally important, we should use JSCS
Reference(s)	- https://stats.stackexchange.com/questions/255465/accuracy-vs-jaccard-for-multiclass-problem - https://scikit-learn.org/stable/modules/generated/sklearn.metrics.jaccard_score.html#sklearn.metrics.jaccard_score

Classification Measure	**Brier Score Loss**
What it is?	the Brier score(BS) measures the mean squared difference between: the predicted probability assigned to

	the possible outcome, and the actual outcome. Best value is 0 and worst value is 1.
How it is calculated (2-label sets)?	$$BS = \frac{1}{N}\sum_{t=1}^{N}(f_t - o_t)^2$$ - N = the number of items you're calculating a Brier score for - ft = the forecast probability - ot is the outcome (1 if it happened, 0 if it didn't) - Σ is the summation symbol. It just means to "add up" all of the values
How it is different from Accuracy?	Accuracy = (TP +TN) / (TP + TN + FP + FN)
How it is different from Accuracy (Example)	Given a sample dataset:  Accuracy = 4/4 = 1 BS = ((0.1-0)^2 + (0.9-1)^2 + (0.8-1)^2 + (0.3-0)^2)/4 = 0.0375
Which scenarios it is best used?	SImilar to Log Loss, whenever our algorithm forecasts the probabilities of a set of specific observations. BS can be used to verify the accuracy of such forecasts. However, it doesn't penalize predictions that are confident but incorrect as much as Log Loss.
Reference(s)	- https://scikit-learn.org/stable/modules/generated/sklearn.metrics.brier_score_loss.html#sklearn.metrics.brier_score_loss - https://en.wikipedia.org/wiki/Brier_score

Classification Measure	**Balanced Accuracy Score**
What it is?	Balanced Accuracy Score(BAS) is the average recall obtained on each unique class. Best value is 1 and worst value is 0.
How it is calculated (2-label sets)?	BAS = (TP / (TP + FN) + TN / (TN + FP)) / 2

	Truth	Probability	Prediction
0	0	0.1	0
1	1	0.9	1
2	1	0.8	1
3	0	0.3	0

How it is different from Accuracy?	Accuracy = (TP +TN) / (TP + TN + FP + FN)
How it is different from Accuracy (Example)	 Accuracy = 880 / 1000 = 0.88 BAS = (300 / 400 + 580 / 600) / 2 = 0.858
Which scenarios it is best used?	Especially useful for imbalanced datasets, BAS will calculate recall for each class independently and take the average of them. Preventing any possible accuracy skew due to an imbalance of observations per class.
Reference(s)	https://scikit-learn.org/stable/modules/generated/sklearn.metrics.balanced_accuracy_score.html#sklearn.metrics.balanced_accuracy_score

Question 4:

Project	**The fraud department at a bank wanted to predict which transactions were fraudulent. The training dataset had 100K credit card transactions, of which 97K are legit and 3K are fraud.**
Best Measure(s) and why	AUC, F1-Score, Balanced Accuracy - Focuses on minimizing false negatives (undetected fraudulent transactions) because of the potential damages in costs - Focuses on minimizing false positives (good transactions mislabeled as fraudulent) to enhance customer shopping experience
Horrible Measure and why	Accuracy - Because the data is heavily imbalanced, by always guessing non-fraudulent, I would be right 97% of the time. Therefore counting the number of correct predictions is not a good idea here.
Assumption(s)	Dataset: fraudulent transaction are labeled as the 'True' or 'Yes' class Business Priorities: - the bank doesn't want to falsely label a good transaction as fraudulent too often as it will decrease customer experience and it also doesn't want to miss an actual fraudulent transaction - Bank wanted to automatically stop a transaction if it thinks it is fraudulent in real-time

Project	**A hospital wanted to predict whether a MRI scan contained cancer.**
Best Measure(s) and why	Recall/Sensitivity - Focuses mainly on reducing false negatives (someone who has cancer but wasn't caught by the test)
Horrible Measure and why	Accuracy - Because the data is heavily imbalanced, by always guessing non-cancer, I would be right most of the time. Therefore counting the number of correct predictions is not a good idea here.
Assumption(s)	Dataset: - cancer detections are labeled as the 'True' or 'Yes' class - Most MRI scans actually doesn't contain cancer Business Priorities: Hospital would rather prefer falsely identify someone who actually doesn't have cancer than to miss someone who has cancer because this is only a screening test

Project	**An IT team wanted to filter spam from email inboxes.**

Best Measure(s) and why	AUC, F1-Score, Balanced Accuracy - Focuses on minimizing false negatives (undetected spam mails) and minimizing false positives (non-spam emails labeled as spam)
Horrible Measure and why	Accuracy - Because the data is heavily imbalanced, by always guessing non-spam, I would be right most of the time. Therefore counting the number of correct predictions is not a good idea here.
Assumption(s)	Dataset: - spam emails are labeled as the 'True' or 'Yes' class - Most emails are not spam Business Priorities: IT teams wants to correctly detect all spam emails because that is the definition of a good spam filter. In addition, the team also wants to avoid mislabeling non-spam emails as spam emails because the email owner will likely never review mails in the spam folder

Project	**A sports analytics department wants to predict which team will win the match.**
Best Measure(s) and why	Accuracy - Since we simply just wanted to track the number of total correct predictions over all predictions, and the dataset is balanced, accuracy is a good candidate to use here Log Loss - Similar to accuracy but takes into account how right or wrong the predictions are and penalizes heavily on falsely confident predictions
Horrible Measure and why	Recall/Sensitivity - Because false positives are not considered in recall, using this measure might introduce many unwanted false positives
Assumption(s)	Dataset: a balanced dataset consists of match histories between any team and any other team Business Priorities: The department is aiming to increase correct predictions on which team will win the match, false positives and false negatives are equally bad in this case.

Project	**A city government wanted to build a system to monitor Twitter to see if any local residents were tweeting about emergencies that needed quick response from the police department. They don't trust Twitter that much; they only want to send police in true emergencies.**

Best Measure(s) and why	AUC, F1-Score, Balanced Accuracy - Focuses on minimizing false negatives (spamming police with non-emergencies which is wasting police's time where they should be focusing on actual emergencies) and minimizing false positives (real emergencies that wasn't detected)
Horrible Measure and why	Accuracy - Because the data is heavily imbalanced, by always guessing non-emergency, I would be right most of the time. Therefore counting the number of correct predictions is not a good idea here.
Assumption(s)	Dataset: - True emergencies are labeled as the 'True' or 'Yes' class - Most twitter messages are not emergencies Business Priorities: The city only wanted to send police true emergencies, in other words, we need to minimize false positives (spam) and false negatives (undetected true emergencies)

Project	**A matchmaking app is launching a new algorithm, aiming to recommend customers their one perfect match right after customers agree to share all of their social media data**
Best Measure(s) and why	Specificity - Focuses on minimizing false positives (bad matches labeled as good matches)
Horrible Measure and why	Recall/Sensitivity - Focuses on minimizing false negatives (perfect match in disguise) and totally ignoring false positives (bad matches labeled as good matches). This will likely spam Tinder users with bad matches and hurt their bottom lines.
Assumption(s)	Dataset: - Perfect matches are labeled as the 'True' or 'Yes' class - Most matches are not ideal Business Priorities: Tinder is willing to miss perfect matches(false negatives) as long as the perfect matches identified are all actually perfect matches

Project	**A telecommunications company, is trying to predict which customers will churn and provide them with free service for a month in advance (Bonus)**
Best	Precision

Measure(s) and why	- Since we only care about increasing the precision of about-to-churn detections, precision is the best measure to use in this case
Horrible Measure and why	Accuracy - Because the data is heavily imbalanced, by always guessing non-about-to-churn, I would be right most of the time. Therefore counting the number of correct predictions is not a good idea here.
Assumption(s)	Dataset: - About-to-churn customers are labeled as the 'True' or 'Yes' class - Most customers are not about to churn Business Priorities: - the company does not want to give out free services to those who will not churn - Even if the company misses a about-to-churn customer, the company can still email him/her about this free service for a month

Question 5:

Rule	**A rule that has high support and high confidence**
Example Rule	Customers who buy eggs also buy milk
Subjectively Interesting?	No
Explanation	A large percentage of transactions contains eggs and milk and milk frequently appears in transactions that contains eggs. This rule is not interesting to uncle Steve because this is a highly occurring combination that also makes sense intuitively (a lot of baked goods requires these 2 ingredients).

Rule	**A rule that has reasonably high support but low confidence**
Example Rule	Customers who buy eggs also buy bread
Subjectively Interesting?	Yes
Explanation	A reasonably large percentage of transactions contains eggs and bread but bread do not frequently appear in transactions that contains eggs. This rule is potentially interesting to uncle Steve because one would have intuitively guessed a high confidence relationship between eggs and bread. By possibly looking at where these 2 items are located in the store. We might be able to increase the confidence of these two items, ultimating increasing our revenue.

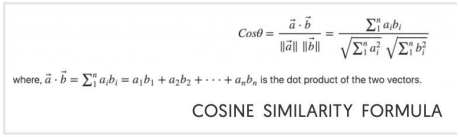
Rule	**A rule that has low support and low confidence**
Example Rule	Customers who buy eggs also buy shampoo
Subjectively Interesting?	No
Explanation	A small percentage of transactions contains eggs and shampoo and shampoo do not frequently appear in transactions that contains eggs. This rule is not interesting to uncle Steve because from our intuition, this combination shouldn’t regularly be bought together commonly.

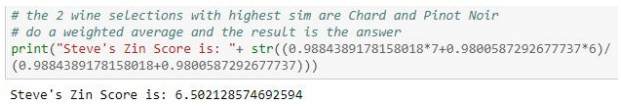
Rule	**A rule that has low support and high confidence**
Example Rule	Customers who buy baby diapers also buy beer
Subjectively Interesting?	Yes
Explanation	A small percentage of transactions contains baby diapers and beer but beer frequently appear in transactions that contains baby diapers. This rule is very interesting to uncle Steve because it is counter intuitive, uncommon, yet highly correlated. Relocating these items closer to each other in the store can potentially increase sales of both items.

Rule	**A rule that has high support, high confidence, but low lift (Bonus)**
Example Rule	Customers who buy coffee also buy tea
Subjectively Interesting?	No
Explanation	Although a large percentage of transactions contains coffee and tea and tea frequently appear in transactions that contains coffee. This rule is not very interesting to uncle Steve because tea is appearing in most of the transactions anyway, meaning tea actually appears with coffee less often than random chance. Another reason why this rule is not interesting is because it matches with the intuition that most people have a preference for coffee or tea, rarely both.

Question 6:

I have applied user-based collaborative filtering with cosine similarity to predict Steve's Zin score on Python. Please refer to the Jupyter Notebook file for more details:

Step	Description	Result
1	wrote a helper function to calculate the cosine similarity between Steve and anyone else (the column with missing score is ignored)  	```# function to calculate cosine sim with Steve def cosine_sim_person(person): steve = [7,6,4,3,4] AB = 0. sumA = 0. sumB = 0. for n in range(0,len(steve)): AB += person[n]*steve[n] sumA += person[n]**2 sumB += steve[n]**2 return ((AB)/((sumA**0.5)*(sumB**0.5)))```
2	I then calculated the cosine similarity between each person to Steve	```# printing the cosine sim for the other 4 people print("Yuri's Cos Sim: "+ str(cosine_sim_person([6,7,4,5,4]))) print("Gary's Cos Sim: "+ str(cosine_sim_person([3,3,1,1,5]))) print("Qurat's Cos Sim: "+ str(cosine_sim_person([2,1,3,7,4]))) print("Brigid's Cos Sim: "+ str(cosine_sim_person([6,7,2,3,3])))``` ```Yuri's Cos Sim: 0.9793588471021603 Gary's Cos Sim: 0.8765009801785553 Qurat's Cos Sim: 0.6915924962229032 Brigid's Cos Sim: 0.9731981665692433```
3	the 2 person with highest cosine similarity are Brigid and Yuri, so I used their scores for Zin and their similarities to calculate a weighted prediction for Steve and got **6.00 (or 6)**	```# the 2 person with highest sim are Brigid and Yuri # do a weighted average and the result is the answer print("Steve's Zin Score is: "+ str((0.9793588471021603*7+0.9731981665692433*5)/(0.9793588471021603+0.9731981665692433)))``` ```Steve's Zin Score is: 6.003155185989337```
4	Out of personal interest, I have also tried content-based recommendation with cosine similarity by writing a helper function to calculate the cosine similarity between Zin and every other wine (the row with missing score is ignored)	```# let's try content-based also with cosine sim def cosine_sim_content(person): zin = [7,3,2,5] AB = 0. sumA = 0. sumB = 0. for n in range(0,len(zin)): AB += person[n]*zin[n] sumA += person[n]**2 sumB += zin[n]**2 return ((AB)/((sumA**0.5)*(sumB**0.5))) # printing the cosine distances for the other 4 people print("Pinot Noir's Cos Sim: "+ str(cosine_sim_content([6,3,2,6]))) print("Chard's Cos Sim: "+ str(cosine_sim_content([7,3,1,7]))) print("Merlot's Cos Sim: "+ str(cosine_sim_content([4,1,3,2]))) print("Cab's Cos Sim: "+ str(cosine_sim_content([5,1,7,3]))) print("Pinot Gris's Cos Sim: "+ str(cosine_sim_content([4,5,4,3])))``` ```Pinot Noir's Cos Sim: 0.9884389178158018 Chard's Cos Sim: 0.9800587292677737 Merlot's Cos Sim: 0.9199783439063688 Cab's Cos Sim: 0.7837463537503494 Pinot Gris's Cos Sim: 0.8709883407113853```

5	The 2 wines with the highest cosine similarity are Chard and Pinot Noir, so I used their scores for Steve and their similarities to calculate a weighted prediction for Zin and got **6.50 (or 6)**	

Question 7:

EDA

1. Target variable 'Purchase' is binary and the ratio between the 2 classes are 653:417 which is quite balanced (random guesses will be correct 61% of the time)
2. 4 features had >60% zeros(DiscCH, DiscMM, PctDiscCH, PctDiscMM), but since they indicate discount status, this is expected
3. There are 24 duplicated rows(2.2% of the entire dataset), but since this is transaction data, this is expected
4. There are no missing cells which is great

Data Preprocessing

1. 7 features were removed for the following reasons:

Features Removed	Reason
DiscCH, DiscMM,SalePriceCH,SalePriceMM, PriceDiff	Highly correlated with PctDiscCH and PctDiscMM
STORE, Store7	Completely redundant with feature 'StoreID'

2. Since StoreID feature is categorical, I have one hot encoded it and replaced it with 5 features (StoreID_1,StoreID_2,StoreID_3,StoreID_4,StoreID_7)
3. The 14 finalized features are then standardized (only if they are non-binary)
4. Target variable has been encoded to '1' (for CH) and '0'(for MM)

Data Split Methodology

1. Randomly Split the entire dataset such that 80% of the dataset will be used for k-fold cross validation, and 20% of the dataset will be used to measure the tuned models' final performances, this is to ensure fairness when comparing between models, since none of the tuned model will be aware of this 20% of test dataset
2. Decided to use 20-fold cross validation to tune each model because it is a very useful technique for assessing the effectiveness of models, particularly in cases where there is a need to mitigate overfitting

Performance Measure Selection

Since this classification problem is binary and I want to find the best balance between False Positives and False Negatives for each class. I have chosen AUC score as the performance measure to fine tune each model.

Fine Tuning 6 Classifier Algorithms

For the classifier algorithms I've fine tuned, the table below contains details for each, please refer to the Jupyter Notebook file for more details:

Algorithm	Hyperparameters Tuned and their best candidate	AUC for CV	AUC for test

		set	set
Decision Tree	'class_weight': 'balanced', 'criterion': 'gini', 'max_depth': None, 'max_leaf_nodes': 8, 'min_samples_leaf': 1, 'min_samples_split': 2	0.89	0.83
Random Forest	'class_weight': 'balanced', 'criterion': 'gini', 'max_depth': None, 'max_leaf_nodes': 40, 'min_samples_leaf': 3, 'min_samples_split': 2, 'n_estimators': 100	0.95	0.86
Logistic Regression	'C': 0.5, 'class_weight': None, 'max_iter': 50, 'penalty': 'l2', 'solver': 'lbfgs', 'tol': 1.0	0.90	0.88
KNN	'leaf_size': 10, 'metric': 'minkowski', 'n_neighbors': 20, 'p': 5, 'weights': 'uniform'	0.89	0.84
Naive Bayes	'var_smoothing': 0	0.85	0.80
SVM	'C': 0.1, 'decision_function_shape': 'ovo', 'degree': 1, 'kernel': 'linear', 'tol': 0.001	0.91	0.88

Model Selection

Since Logistic Regression, and SVM has the highest and identical test AUC score (.88). Between the 2 models I have chosen Logistic Regression because it has:

- Better interpretability
- Faster training speed

Deployment Plan

Assuming the business implementation of this model will bring in additional revenue, even with a balanced accuracy of 80%.

Since a balanced accuracy score of 80%(from fine tuned logistic regression model, using the best threshold which is 0.5) is much better than a balanced accuracy of 50% (if we were to always predict grocery store customers will purchase Citrus Hill). I believe this model is good enough to be deployed today. However, we have to:

- continuously monitor performances of new data and make sure performances increase
- periodically re-train the model
- Make sure we are always only predicting Citrus Hill or Minute Maid, and not another brand of orange juice because this classifier will only predict Citrus Hill or Minute Maid