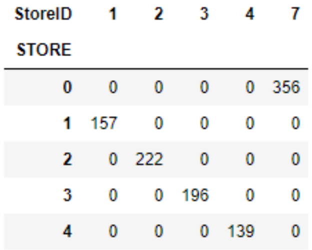
CLASSIFICATION MODEL

Build a model to predict whether a grocery store customer will Purchase Citrus Hill (CH) or Minute Maid (MM) orange juice.

1. Download the file *OJ.csv*. The target feature is *Purchase*. The rest of the features are self-explanatory, hopefully.
2. Preprocess the data however you see fit. Describe what you did and why.
3. Split the data into training and testing sets. Describe what you did and why.
4. Choose an appropriate metric to analyze a model's performance. Justify.
5. Build five different models, using five different classifier algorithms. (Any five will do.)
    a. Tune each model. What were the best parameter values for each model?
6. Describe and compare the performance of each fine-tuned model.
7. Select the best model. Justify.
8. Is this model good enough to deploy today? Justify.

Solution:

1. Initial analysis and dimensionality reduction:
    a. Dataset has 1070 instances with 19 attributes including target and row ID.
    b. There are no null values.
    c. The following attributes are categorical: SpecialCH, SpecialMM, Store, StoreID
2. Preprocessing – dimensionality reduction and feature selection
    a. Attribute number 1 is dropped as it just the row ids.
    b. The attributes 'STORE' and 'StoreID' are the same. If the store is 7 then the storeID has a value of 0. Figure 1 below, shows the crosstab between Store and StoreID. Similarly, the attribute 'Store7' identifies if the purchase was made at store 7 by indicating yes or no. Therefore, I dropped features 'STORE' and 'Store7.'



StoreID	1	2	3	4	7
STORE					
0	0	0	0	0	356
1	157	0	0	0	0
2	0	222	0	0	0
3	0	0	196	0	0
4	0	0	0	139	0

Figure 1: Crosstabe between Store and StoreID attributes

    c. I converted target variable (Purchase) to Boolean value by giving CH a value of 0 and MM a value of 1, this way during feature selection I can check if any variables are highly correlated.
    d. I converted 'StoreID' and 'WeekofPurchase' to string variables as they are categorical variables and want to avoid any aggregations in the model.

e. Performed Min Max scaling on the following attributes: ' PriceCH', 'PriceMM', 'DiscCH', 'DiscMM', 'LoyalCH', 'SalePriceMM', 'SalePriceCH', 'PriceDiff', 'PctDiscMM', 'PctDiscCH', 'ListPriceDiff.'

f. As part of feature selection, I performed multicollinearity test for all non-categorical variables. Figure 2 shows the results matrix.

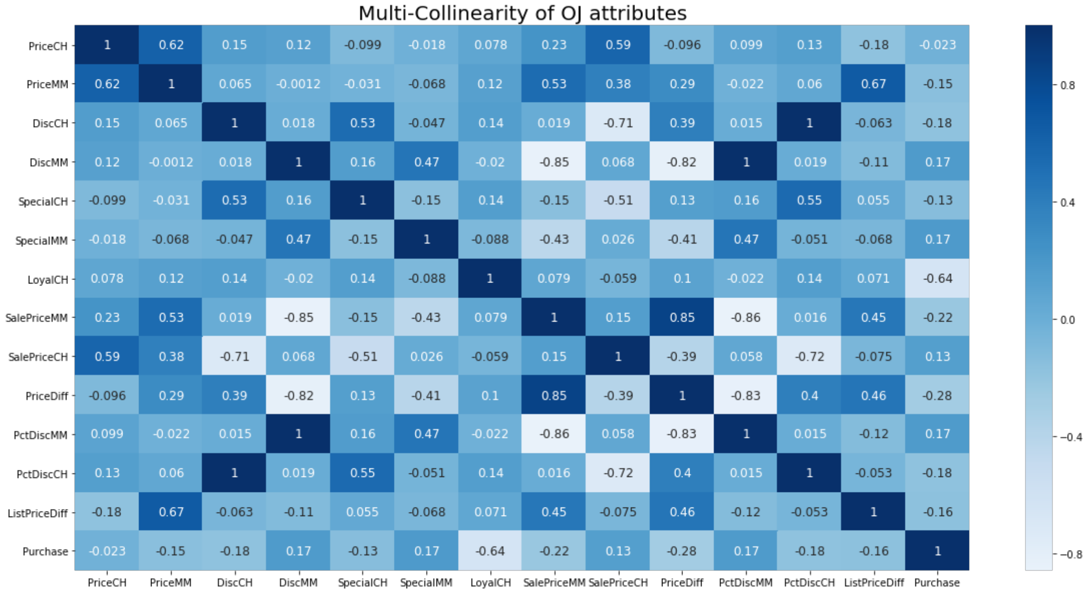


Figure 2: Multi-collinearity test between non-categorical attributes

g. To study high correlation, I gave a threshold of 0.7, that is and variables with great than 0.7 correlation will be dropped. As seen the matrix above, there are some highly correlated features. The table below shows these

Variable	Correlation
DiscCH	PctDiscCH, SalePriceCH
DiscMM	PctDiscMM, PriceDiff, SalePriceMM

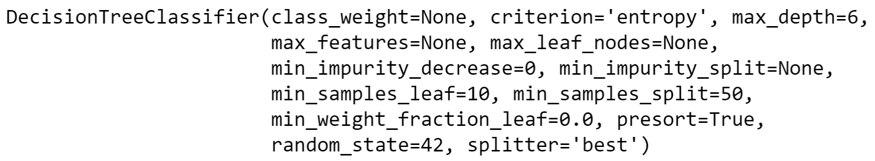
I dropped PctDiscCH, PctDiscMM, SalePriceCH and SalePriceMM.

h. I performed One Hot Encoding on ‘StoreID’ and ‘WeekofPurchase’ for better prediction capabilities. This created a total of 71 columns including target variable. There were no record under few stores, so no attributes were returned for them, such as StoreID 5 & 6.

3. Now that I have my final features, the next step I took to split the data into training and testing sets. I decided to split the data into 70-30 parts because there is only ~1000 rows of data and this gives enough samples (30% of the data) under the test set; anything more might reduce the learning opportunity for the model as it will reduce the training set samples, and anything less might comprise on test set prediction quality.

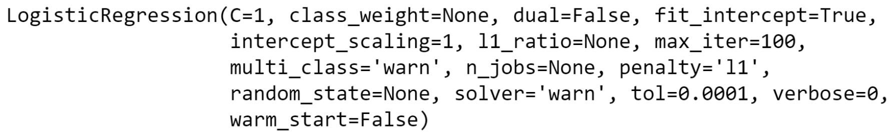
4. To analyze the model's performance, I choose accuracy as my metric because it tells me the true positives of which juice a customer will buy. Since the dataset is not highly imbalanced – 471 instances for MM and 653 for CH – accuracy results will not be misleading.

5. Below are the 5 models I build and their best parameter values after performing grid search tuning.
   a. Decision Tree



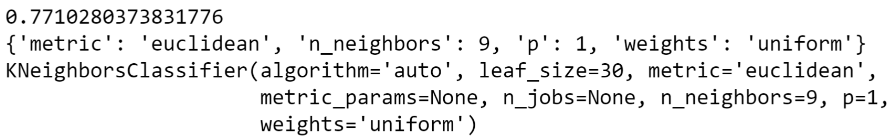
```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=6,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0, min_impurity_split=None,
                       min_samples_leaf=10, min_samples_split=50,
                       min_weight_fraction_leaf=0.0, presort=True,
                       random_state=42, splitter='best')
```

   b. Logistic Regression



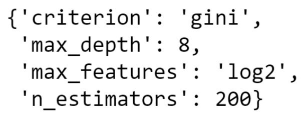
```
LogisticRegression(C=1, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l1',
                   random_state=None, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

   c. K-Nearest Neighbors



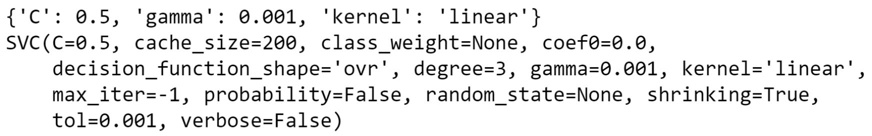
```
0.7710280373831776
{'metric': 'euclidean', 'n_neighbors': 9, 'p': 1, 'weights': 'uniform'}
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
                     metric_params=None, n_jobs=None, n_neighbors=9, p=1,
                     weights='uniform')
```

   d. Random Forest



```
{'criterion': 'gini',
 'max_depth': 8,
 'max_features': 'log2',
 'n_estimators': 200}
```

   e. SVM



```
{'C': 0.5, 'gamma': 0.001, 'kernel': 'linear'}
SVC(C=0.5, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.001, kernel='linear',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

6. Summary of the models:
   a. Decision Tree – As seen in confusion matrix below, decision tree did a good job predicting both MM and CH purchases, however it did have several instances where it was not able to identify the customer's purchase, especially for MM.

	Predicted: CH	**Predicted: MM**
True: CH	175	18
True: MM	52	76

   b. Logistic Regression – did a good job predicting the purchases correctly despite the few wrongly predicted ones. It has highly correctly predicted values.

	Predicted: CH	**Predicted: MM**
True: CH	170	23
True: MM	43	85

c. KNN – did not perform well. It was unable to recognize most of the purchases correctly under both the products.

	Predicted: CH	**Predicted: MM**
True: CH	165	28
True: MM	46	82

d. Random Forest – Random Forest performance was good in terms of correctly predicting the purchases, but it did predict quite a few CH as MM and MM as CH.

	Predicted: CH	**Predicted: MM**
True: CH	166	27
True: MM	42	86

e. SVM – This model did a good job in identifying MM and CH purchases correctly. However, it did predict quite a few CH as MM and vice versa. But overall, it was better at predicting correctly.

	Predicted: CH	**Predicted: MM**
True: CH	165	28
True: MM	39	89

Below are the results for all 5 models

Model **Performance Metric**	**Decision Tree**	**Logistic Regression**	**KNN**	**Random Forest**	**SVM**
Accuracy	78.19%	**79.44%**	76.95%	78.50%	79.13%
Precision	80.85%	78.70%	74.55%	76.11%	76.07%
Recall	59.38%	66.41%	64.06%	67.19%	69.53%
F1 Score	68.47%	72.03%	68.91%	71.37%	72.65%
Log Loss	7.5319	7.1015	7.9623	7.4243	7.2091
AUC	75.02%	77.24%	74.78%	76.60%	77.51%

Compared to the other models, Logistic Regression performed the best since accuracy is our main metric, although it was not too far off from SVM.

7. I do not think the model is good enough to deploy it today because 79% accuracy is not good enough especially when the model also has a quite a few wrongly predicted purchases. 79% accuracy is not enough to make business decisions on what product the customer will buy give additional promotional offers or cut down on a product. To improve this model, we need more data containing more transactions of both purchases. This was we can further train the model and evaluate its performance.