Retail product image classification

Ashish Pujari
apujari@uchicago.edu

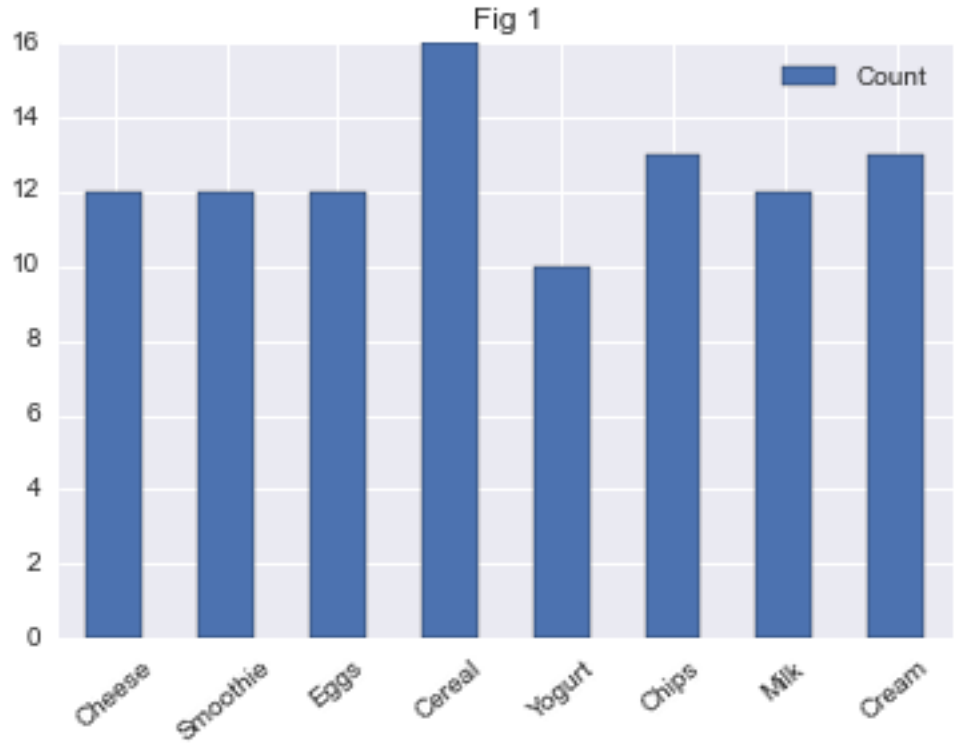
December 10, 2015

1 Introduction

In the Consumer Packaged Goods (CPG) industry, product images are becoming popular in web based applications and planogramming tools. Use of images and alternate views of products enhance user experience and are sometimes more intuitive than textual product descriptions. However it becomes cumbersome and expensive to manage, track and catalog millions of product images manually. The objective of this project is to test and compare various machine learning techniques to classify and label product images into product categories.

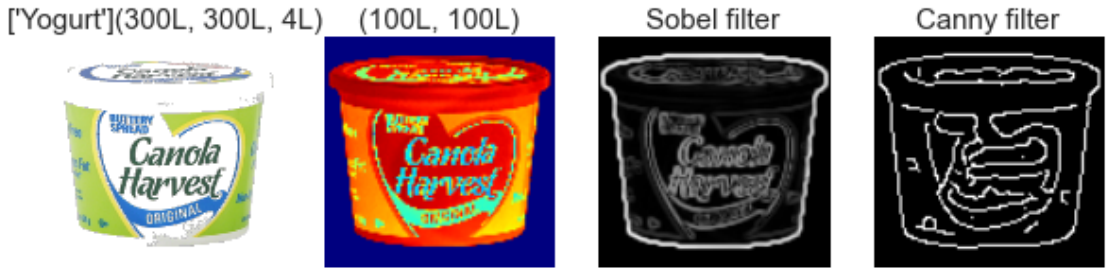
2 Dataset

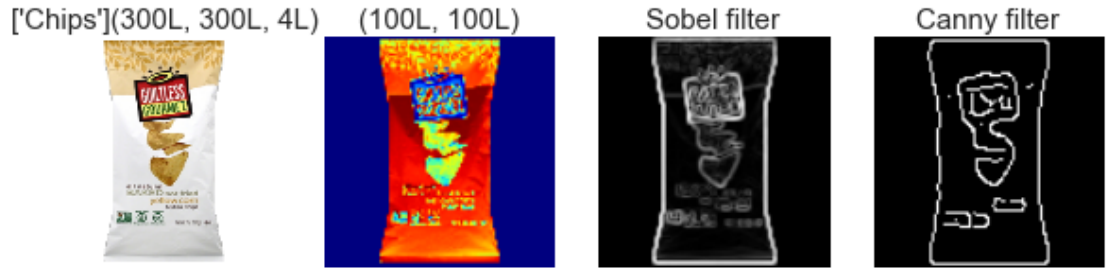
100 front-facing product images in the PNG format and corresponding text labels stored in a CSV file are read into memory. Fig 1 below shows the distribution of the 8 retail categories in the image data set.

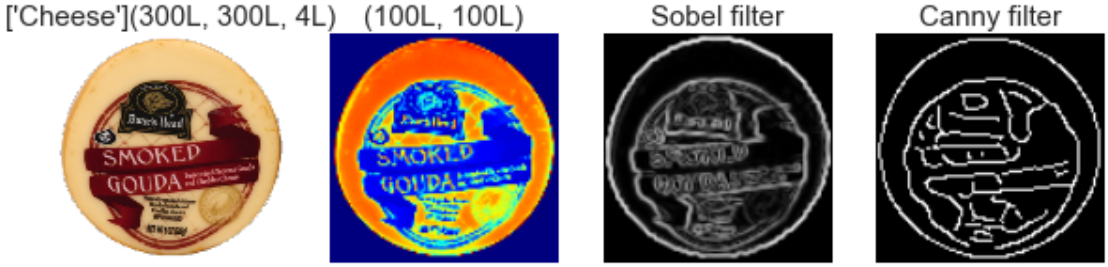


3 Image Processing

Since each of the product categories have distinct object shapes, edge detection algorithms are expected to perform well in extracting features for this classification problem. Sobel and Canny edge detection techniques are applied on a low resolution sample of the raw image files. Some of the processed images are visually compared below to verify the results of edge detection. The Sobel algorithm will be selected to process images because, unlike Canny, it places more emphasis on the edges and captures the essential features for shape recognition.

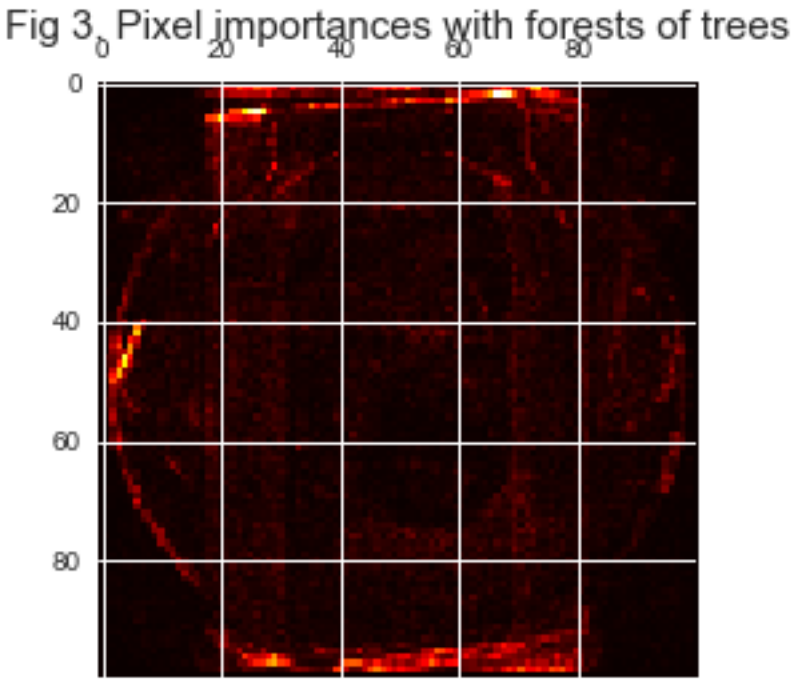






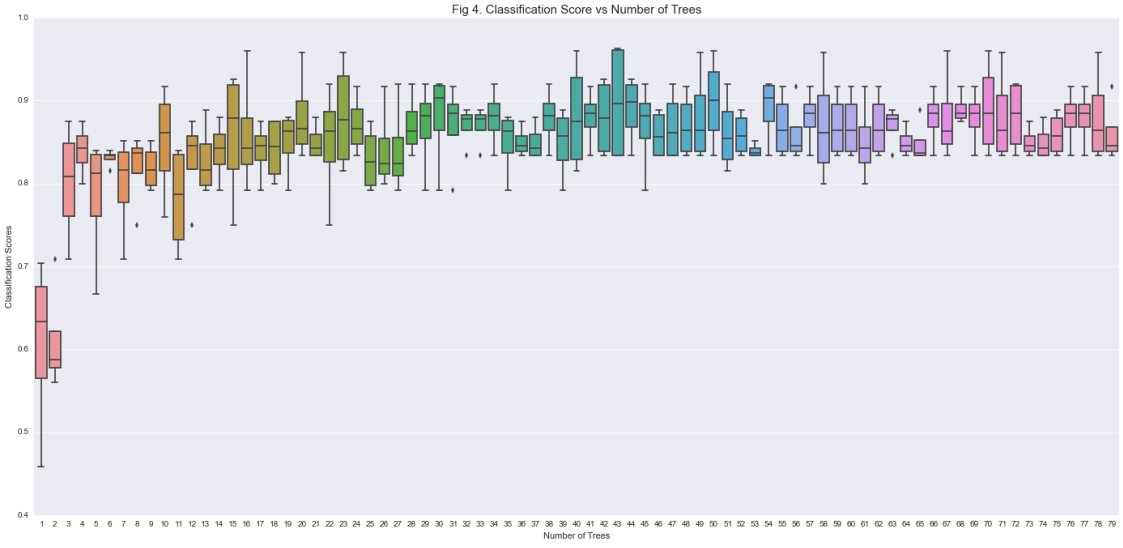
What does the classifer see ?

Pixel importances - Since we are dealing with raw images, it is useful to see how a classifier extracts the important features. In Fig 3, the 2D image data was fit to an ExtraTreesClassifier to determine pixels importances for classification. The brighter pixels are given more weightage by the classifier. The number of estimators are increased to get a better picture.



3.1 Determine Estimator Size

We will be using tree ensembles later. It would be computationally efficient to determine an optimum size of the *n_estimators* parameter. Below we see that the classification score doesn't improve beyond n_estimators = 60.



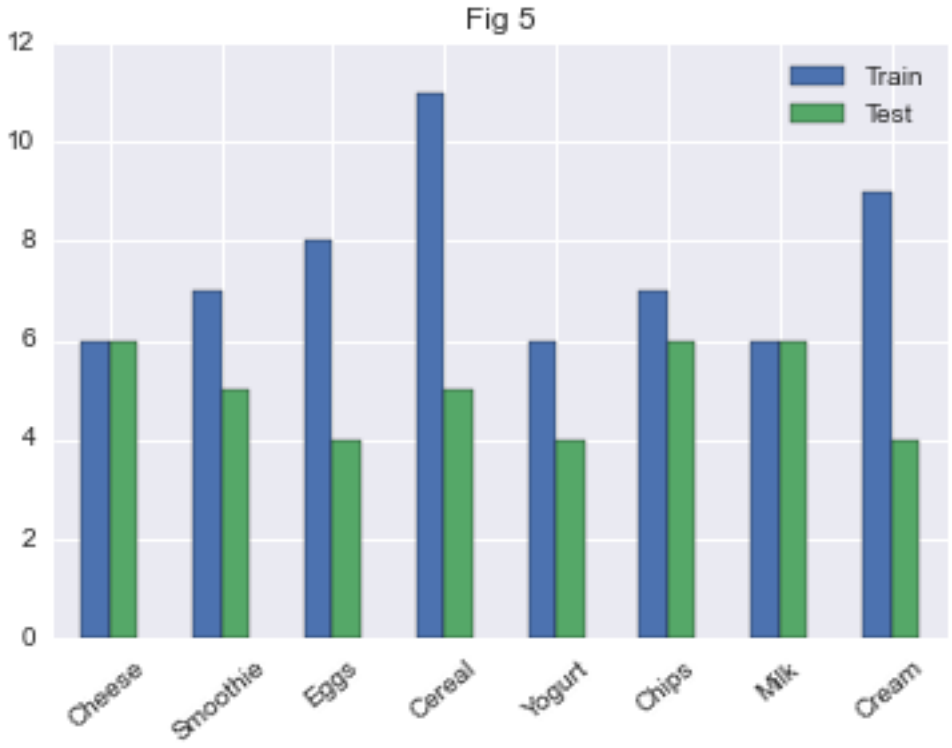
4 Model Generation

An ensemble of classification models would be fit to a training images dataset and tested against a test dataset. Where applicable, exhaustive grid search using GridSearchCV is used to scan a range of parameters to obtain best results using N-fold cross-validation.

4.0.1 `Split Data`

The data is split into training 60% and 40% test sets. Fig 5 shows distribution of the train and test data.

```
Total dataset X: (100L, 10000L) y: 100
Training set: (60L, 10000L) Testing set: (40L, 10000L)
```



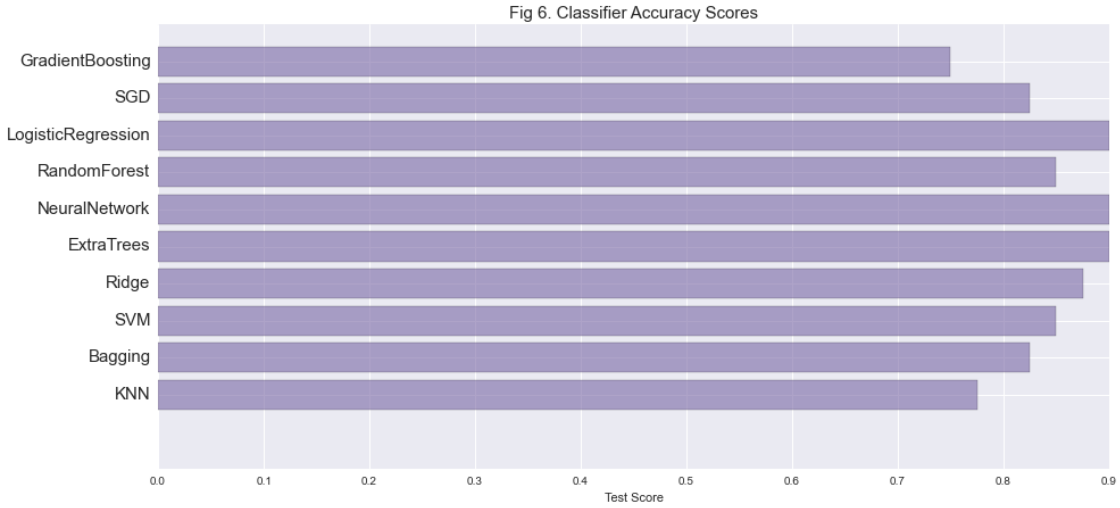
4.0.2 `Define Classifiers`

Although certain models such as Neural Networks and Trees are known to outperform other methods for image classification, we will compare various methods in the list below during the modeling process. The best estimators from Grid Search can be seen in the Appendix. In addition to train-test split, we will apply a 3-fold cross-validation for the grid search. Also, given the analysis above, the estimator size will be set to 100. Since computation is being run on a Windows platform it will run single-threaded given known limitations of iPython.

4.0.3 `Generate Models`

The results of model execution are printed below.

```
Fitting KNN
Fitting Bagging
Fitting SVM
Fitting Ridge
Fitting ExtraTrees
Fitting NeuralNetwork
Fitting RandomForest
Fitting LogisticRegression
Fitting SGD
Fitting GradientBoosting
```



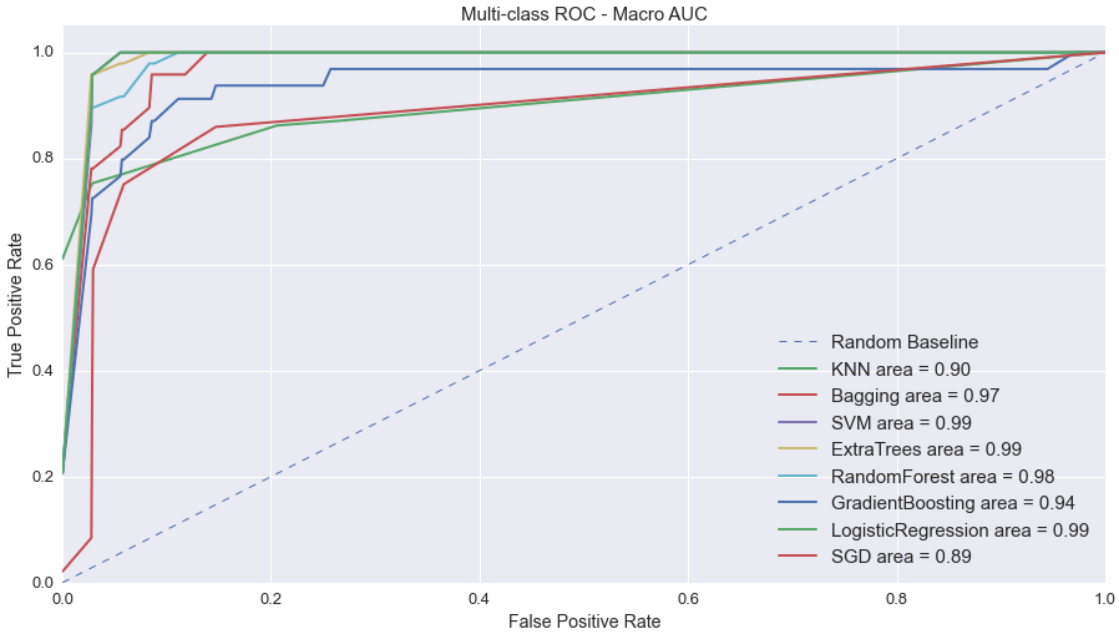
Out[14]:

	Classifier	Test Score	Train Score	Train Time
4	ExtraTrees	0.900	1.00	0.124
5	NeuralNetwork	0.900	1.00	592.029
7	LogisticRegression	0.900	1.00	10.156
3	Ridge	0.875	1.00	0.218
2	SVM	0.850	1.00	144.706
6	RandomForest	0.850	1.00	0.187
1	Bagging	0.825	1.00	9.095
8	SGD	0.825	1.00	1.342
0	KNN	0.775	0.85	0.172
9	GradientBoosting	0.750	1.00	132.367

4.1 Confusion Matrices

4.2 Receiver Operating Characteristics

Plot multi-class ROCs using macro-AUC averages for all classes.



4.3 Holdout Testing

To test model performance we will run two of the best cross-validated classifier against images it has never seen before. The Neural Network correctly predicts all images and also recognizes different shapes of milk cans, yogurt, cheese and eggs. The Logistic Regression correctly labels all but one where it labels a "Yogurt" as "Chips".

4.3.1 `Neural Network Classifier`

4.3.2 Logistic Regression Classifier

5 Conclusion

The tests were run with and without image filters. The use of a Sobel filter on the images helped improved the test accuracy scores. Other image processing techniques such use of pixel distribution histograms instead of raw pixels could also be applied.

Although most classifiers are able to distinguish beween well differentiated shapes such as "Eggs" and "Smoothie", classifing similar-shaped images of "Chips", "Yogurt" and "Cereal" is challenging. This can be seen from the confusion matrices in the Appendix. Some improvement was seen by selecting better training samples and applying cross-validation, but further work needs to be done.

Neural Network and Logistic Regression consistently prediced the test set with over 90% accuracy score. In the hold out test the Neural Network predicted all samples with 100% accuracy. Although Neural Networks take a longer time to train, they may be better suited for generalizing more complex and noisy images and a larger number of classes. As can be seen in the best estimator selection in the Appendix, the Convolution Recitifier performs best among the neural network configurations. It is also interesting to note that tree based classfiers - Extra Trees and Random Forests provide satisfactory results with minimal training time.

To solve this problem at industrial scale however, a broader set of product categories with noise needs to be tested on larger machines with multiple cores or GPUs. Also additional techniques besides shape detection would need to be applied to correctly classify product categories that have similar shapes.

6 Appendix

6.1 Classification Report

```
-------------------------------------------------------
KNN
```

```
-------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       1.00      1.00      1.00         5
     Cheese       0.57      0.67      0.62         6
      Chips       1.00      0.83      0.91         6
      Cream       1.00      1.00      1.00         4
       Eggs       0.80      1.00      0.89         4
       Milk       0.80      0.67      0.73         6
   Smoothie       0.56      1.00      0.71         5
     Yogurt       0.00      0.00      0.00         4

avg / total       0.73      0.78      0.74        40

-------------------------------------------------------------
Bagging
-------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       0.56      1.00      0.71         5
     Cheese       1.00      0.83      0.91         6
      Chips       1.00      0.83      0.91         6
      Cream       0.80      1.00      0.89         4
       Eggs       0.80      1.00      0.89         4
       Milk       1.00      0.83      0.91         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       0.00      0.00      0.00         4

avg / total       0.80      0.82      0.80        40

-------------------------------------------------------------
SVM
-------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       1.00      1.00      1.00         5
     Cheese       1.00      0.50      0.67         6
      Chips       0.75      1.00      0.86         6
      Cream       0.80      1.00      0.89         4
       Eggs       0.80      1.00      0.89         4
       Milk       0.86      1.00      0.92         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       0.50      0.25      0.33         4

avg / total       0.85      0.85      0.83        40

-------------------------------------------------------------
Ridge
-------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       1.00      0.80      0.89         5
     Cheese       1.00      0.83      0.91         6
      Chips       0.75      1.00      0.86         6
```

```
      Cream       0.80      1.00      0.89         4
       Eggs       0.80      1.00      0.89         4
       Milk       0.86      1.00      0.92         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       1.00      0.25      0.40         4

avg / total       0.90      0.88      0.86        40

------------------------------------------------------------
RandomForest
------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       0.71      1.00      0.83         5
     Cheese       0.83      0.83      0.83         6
      Chips       1.00      1.00      1.00         6
      Cream       0.80      1.00      0.89         4
       Eggs       0.80      1.00      0.89         4
       Milk       1.00      0.67      0.80         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       0.50      0.25      0.33         4

avg / total       0.85      0.85      0.84        40

------------------------------------------------------------
GradientBoosting
------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       0.67      0.80      0.73         5
     Cheese       0.80      0.67      0.73         6
      Chips       1.00      0.83      0.91         6
      Cream       0.60      0.75      0.67         4
       Eggs       0.80      1.00      0.89         4
       Milk       0.60      0.50      0.55         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       0.50      0.50      0.50         4

avg / total       0.76      0.75      0.75        40

------------------------------------------------------------
ExtraTrees
------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       0.83      1.00      0.91         5
     Cheese       1.00      0.83      0.91         6
      Chips       0.75      1.00      0.86         6
      Cream       0.80      1.00      0.89         4
       Eggs       1.00      1.00      1.00         4
       Milk       1.00      1.00      1.00         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       1.00      0.25      0.40         4
```

```
avg / total       0.92      0.90      0.88        40

-------------------------------------------------------------
NeuralNetwork
-------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       0.83      1.00      0.91         5
     Cheese       1.00      1.00      1.00         6
      Chips       0.86      1.00      0.92         6
      Cream       1.00      1.00      1.00         4
       Eggs       1.00      1.00      1.00         4
       Milk       1.00      0.67      0.80         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       0.50      0.50      0.50         4

avg / total       0.91      0.90      0.90        40

-------------------------------------------------------------
LogisticRegression
-------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       0.83      1.00      0.91         5
     Cheese       1.00      1.00      1.00         6
      Chips       0.75      1.00      0.86         6
      Cream       1.00      1.00      1.00         4
       Eggs       1.00      1.00      1.00         4
       Milk       1.00      0.83      0.91         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       0.50      0.25      0.33         4

avg / total       0.89      0.90      0.89        40

-------------------------------------------------------------
SGD
-------------------------------------------------------------
             precision    recall  f1-score   support

     Cereal       0.83      1.00      0.91         5
     Cheese       1.00      0.67      0.80         6
      Chips       0.75      1.00      0.86         6
      Cream       0.80      1.00      0.89         4
       Eggs       1.00      1.00      1.00         4
       Milk       1.00      0.67      0.80         6
   Smoothie       1.00      1.00      1.00         5
     Yogurt       0.25      0.25      0.25         4

avg / total       0.85      0.82      0.82        40
```

6.2 Selected Classifiers

```
-------------------------------------------------------------
KNN
```

```
------------------------------------------------------------
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_neighbors=4, p=2, weights='uniform')
------------------------------------------------------------
Bagging
------------------------------------------------------------
BaggingClassifier(base_estimator=None, bootstrap=True,
         bootstrap_features=False, max_features=1.0, max_samples=1.0,
         n_estimators=100, n_jobs=1, oob_score=False, random_state=None,
         verbose=0)
------------------------------------------------------------
SVM
------------------------------------------------------------
SVC(C=0.01, cache_size=200, class_weight=None, coef0=0.0, degree=3, gamma=0.0,
  kernel='linear', max_iter=-1, probability=True, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
------------------------------------------------------------
Ridge
------------------------------------------------------------
RidgeClassifier(alpha=100.0, class_weight=None, copy_X=True,
        fit_intercept=True, max_iter=None, normalize=False, solver='auto',
        tol=0.001)
------------------------------------------------------------
ExtraTrees
------------------------------------------------------------
ExtraTreesClassifier(bootstrap=False, class_weight=None, criterion='gini',
           max_depth=None, max_features='auto', max_leaf_nodes=None,
           min_samples_leaf=1, min_samples_split=2,
           min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=1,
           oob_score=False, random_state=None, verbose=0, warm_start=False)
------------------------------------------------------------
NeuralNetwork
------------------------------------------------------------
Classifier(Rect1=<sknn.nn.Convolution `Rectifier`: channels=8, name='Rect1', frozen=False, kernel_shape=
      SM1=<sknn.nn.Layer `Softmax`: units=8L, name='SM1', frozen=False>,
      batch_size=1, debug=False, dropout_rate=None, f_stable=0.001,
      layers=[<sknn.nn.Convolution `Rectifier`: channels=8, name='Rect1', frozen=False, kernel_shape=(3,
      learning_momentum=0.6, learning_rate=0.005, learning_rule=u'sgd',
      loss_type=None, mutator=None, n_iter=50, n_stable=10,
      random_state=None, regularize=None, valid_set=None, valid_size=0.0,
      verbose=None, warning=None, weight_decay=None, weights=None)
------------------------------------------------------------
RandomForest
------------------------------------------------------------
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
------------------------------------------------------------
LogisticRegression
------------------------------------------------------------
LogisticRegression(C=100000.0, class_weight=None, dual=False,
```

```
          fit_intercept=True, intercept_scaling=1, max_iter=100,
          multi_class='multinomial', penalty='l2', random_state=None,
          solver='lbfgs', tol=0.0001, verbose=0)
------------------------------------------------------------
SGD
------------------------------------------------------------
SGDClassifier(alpha=0.0001, average=False, class_weight=None, epsilon=0.1,
       eta0=0.0, fit_intercept=True, l1_ratio=0.15,
       learning_rate='optimal', loss='log', n_iter=5, n_jobs=1,
       penalty='none', power_t=0.5, random_state=None, shuffle=True,
       verbose=0, warm_start=False)
------------------------------------------------------------
GradientBoosting
------------------------------------------------------------
GradientBoostingClassifier(init=None, learning_rate=0.005, loss='deviance',
              max_depth=3, max_features=None, max_leaf_nodes=None,
              min_samples_leaf=1, min_samples_split=2,
              min_weight_fraction_leaf=0.0, n_estimators=100,
              random_state=None, subsample=1.0, verbose=0,
              warm_start=False)
```