OTTO GROUP PRODUCT CLASSIFICATION CHALLENGE

Applied Artificial Intelligence System Project-5

Username: **Cyclops**

Domain: Machine Learning

INDEX

Abstract:

Applied various machine learning models for the classification for Otto group classification challenge and compared their running time and log loss score.

Software requirements

To run the source code, you must have the below software installed in your machine.

Software	Download link
Anaconda (RECOMMENDED)	https://www.continuum.io/
OR	
Python 3.5	https://www.python.org/downloads/
sklearn	http://scikit-learn.org/stable/install.html
matplotlib	http://matplotlib.org/downloads.html
numpy	http://www.scipy.org/scipylib/download.html
OR	
jupyter notebook	http://jupyter.readthedocs.io/en/latest/install.html

Instructions:

After downloading the entire source code and the data folders, store it in any location. The python scripts are customized to automatically adjust and find the data files subject to both the 'script' and the 'data' folder are under the same parent folder.
Open a command terminal and type the following command:

```
cd<absolute path of the script directory>
python keras.py
```

.pynb File
I have included .pynb file also so you can run python notebook also using jupyter notebook.

Python file:

I tried multiple models but I have included only python file with keras model as it gave the best log loss score.

Output file

I have included one output files (otto.csv) as it gave the best result but I have included result summary with all models in the result page.

Models used

Weka:
AdaBoostM1
IB1 Classifier

Python:
K Nearest Neighbour
Logistic Regression
Random forest
Keras Model

Weka

First tried classification models available on Weka and got the following results based on training the data.

Following is the run information

=== Run information ===

AdaBoostM1

=== Evaluation on training set ===

Time taken to test model on training data: 0.41 seconds

=== Summary ===

```
Correctly Classified Instances       30257              48.8978 %
Incorrectly Classified Instances      31621              51.1022 %
Kappa statistic                    0.3234
Mean absolute error                  0.1472
Root mean squared error                0.2713
Relative absolute error              79.7143 %
Root relative squared error            89.2833 %
Total Number of Instances            61878
```

=== Detailed Accuracy By Class ===

```
           TP Rate  FP Rate  Precision  Recall   F-Measure  MCC     ROC Area  PRC Area  Class
           0.000    0.000    0.000      0.000    0.000      0.000   0.753     0.061     Class_1
           1.000    0.336    0.512      1.000    0.677      0.583   0.832     0.512     Class_2
           0.000    0.000    0.000      0.000    0.000      0.000   0.782     0.254     Class_3
```

```
                 0.000   0.000   0.000    0.000   0.000    0.000   0.757    0.085    Class_4
                 0.000   0.000   0.000    0.000   0.000    0.000   0.757    0.087    Class_5
                 1.000   0.341   0.465    1.000   0.635    0.554   0.830    0.465    Class_6
                 0.000   0.000   0.000    0.000   0.000    0.000   0.767    0.093    Class_7
                 0.000   0.000   0.000    0.000   0.000    0.000   0.795    0.278    Class_8
                 0.000   0.000   0.000    0.000   0.000    0.000   0.777    0.163    Class_9
Weighted Avg.    0.489   0.165   0.240    0.489   0.321    0.278   0.803    0.337

=== Confusion Matrix ===

    a    b     c    d    e    f     g    h    i   <-- classified as
    0  1929    0    0    0    0     0    0    0 |    a = Class_1
    0 16122    0    0    0    0     0    0    0 |    b = Class_2
    0  8004    0    0    0    0     0    0    0 |    c = Class_3
    0  2691    0    0    0    0     0    0    0 |    d = Class_4
    0  2739    0    0    0    0     0    0    0 |    e = Class_5
    0    0    0    0    0 14135    0    0    0 |    f = Class_6
    0    0    0    0    0  2839    0    0    0 |    g = Class_7
    0    0    0    0    0  8464    0    0    0 |    h = Class_8
    0    0    0    0    0  4955    0    0    0 |    i = Class_9
```

IB1 Classifier

```
Time taken to build model: 0.08 seconds
=== Evaluation on test split ===

Time taken to test model on test split: 454.95 seconds
=== Summary ===

Correctly Classified Instances       15675              74.5045 %
Incorrectly Classified Instances      5364              25.4955 %
Kappa statistic                          0.6927
Mean absolute error                      0.0567
Root mean squared error                  0.238
Relative absolute error                 30.6978 %
Root relative squared error             78.3397 %
Total Number of Instances            21039

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.582    0.015    0.554      0.582    0.568      0.554    0.784     0.336     Class_1
                 0.717    0.107    0.705      0.717    0.711      0.606    0.805     0.580     Class_2
                 0.482    0.082    0.463      0.482    0.472      0.393    0.699     0.290     Class_3
                 0.394    0.026    0.403     0.394    0.398     0.372    0.684     0.185     Class_4
                 0.961    0.003    0.927     0.961    0.944     0.941    0.979     0.892     Class_5
                 0.918    0.023    0.920     0.918    0.919     0.896    0.947     0.864     Class_6
                 0.560    0.014    0.660     0.560    0.606     0.590    0.773     0.390     Class_7
                 0.863    0.020    0.870     0.863    0.866     0.845    0.921     0.769     Class_8
                 0.814    0.014    0.843     0.814    0.828     0.813    0.900     0.701     Class_9
Weighted Avg.    0.745    0.050    0.746     0.745    0.745     0.696    0.847     0.623
```

```
=== Confusion Matrix ===

    a    b    c    d    e    f    g    h    i   <-- classified as
  382   21   10    1    4   44   24   86   84 |    a = Class_1
    5 3971 1170  254   18   13   53   32   24 |    b = Class_2
    3 1114 1305  189    8   14   52   21    4 |    c = Class_3
    3  292  205  353    9   23    9    1    1 |    d = Class_4
    0   16   12    2  862    0    1    2    2 |    e = Class_5
   63   41   13   33    5 4374   71  100   65 |    f = Class_6
   26  105   80   30   12   94  548   69   15 |    g = Class_7
   83   37   14    5    6  126   57 2475   66 |    h = Class_8
  124   39    7    9    6   64   15   58 1405 |    i = Class_9
```

So with Weka Models the best correct classification on training data achieved was **74.5045 %** so to achieve better accuracy then I ran python script for multiple models.

Python Execution:

The file included is run on keras model but I have tried on multiple models and summarised their results below

K Nearest Neighbours

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 5, metric= 'minkowski', p=2)
knn.fit(X_trn, Y_trn)
```

Random forest

```
knn = RandomForestClassifier(n_jobs=10, random_state=36)
```

Logistic Regression

```
knn = LogisticRegression(penalty='l2', solver='lbfgs', n_jobs=-1, C=0.01, multi_class='multinomial')
```

Keras Model

```
def baseline_model():
            # create model
   model = Sequential()
   model.add(Dense(8, input_dim=93, activation='relu'))
   model.add(Dense(9, activation='softmax'))
            # Compile model
   model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
   return model
knn = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
knn.fit(X_trn, Y_trn)
```

Results

Following are results obtained with different models

Model	Parameters	Running Time	Space Required	Logloss Score
K Nearest Neighbour	5 Neighbors , metric= 'minkowski', p=2;	736.7s	6.08 MB	4.99941
K Nearest Neighbour	10 Neighbors , metric= 'minkowski', p=2;	107.6s	6.08 MB	1.46450
K Nearest Neighbour	256 Neighbors , metric= 'minkowski', p=2;	115.9s	10 MB	0.72287
K Nearest Neighbour	1024 Neighbors , metric= 'minkowski', p=2;	22.8	13.55	0.84514
Random Forest	n_jobs=10, random_state=36	210.4s	6.08 MB	1.50585
Logistic regression	penalty='l2', solver='lbfgs', n_jobs=-1, C=0.01, multi_class='multinomial	264.3s	27.41 MB	0.64041
Keras	build_fn=baseline_model, epochs=200, batch_size=5, verbose=0	86.4s	27.52 MB	0.60681