MULTINOMIAL CLASSIFIER FOR E-COMMERCE DATA

AKRITI SHARMA
akritish@usc.edu
USC ID: 7522731597

ISHANI SHETH
isheth@usc.edu
USC ID: 4727532860

PROJECT DEFINITION

In this project, multinomial classification takes place for complete dataset into nine different target classes, for a large data set provided by the Otto group [1].

Data analytics is a key factor in decision making whether it involves better customers and business partnership or product performance analysis. For a group like Otto, being world's biggest e-commerce companies, it's important for them to have a consistent analysis of the performance of their products.

The problem that we are going to solve with this project is, to accurately divide the product ID's into 9 different classes on the basis of 93 features defining each one of them.

More accurate the classification is, the group gets a better insight of their product range and accordingly they can determine the performance of each class accordingly.

BACKGROUND

Classification is a form of data analysis that can be used to extract models describing important data classes or to predict future data trends. However, according to No free lunch theorem – "There is no single classifier that will outperform others in any given domain" [2]. The aim of this project is to classify list of products into multiple classes based on data provided by the Otto group, one of the largest e-commerce group of companies. The classification techniques examined in this project are an extension of original binary classification techniques, called Multi-class classification. So far in this domain of classification; implementation of ensemble schemes has taken place

DATASET

Our project is an ongoing competition on Kaggle [1] and we are provided with below mentioned two datasets:

- Training Set (.csv format)
- Test Set (.csv format)

Training Set (5,878,505 observations):

Attributes	Product ID'S	Feat_1	……..to…….	Feat_93	Target/label class
Number of instances	**61,878**	**61,878**	**…….**	**61,878**	**61,878 (for 9 classes combined)**

Explanation of the data fields:

Training Dataset consists of the following attributes:

- Product ID: Indicates the product ids, which is unique for each mentioned product in the dataset.
- Feat_1.... Feat_93: These 93 columns indicate various features which are numeric values and have been used to classify the products into nine different classes.
- Target: This column indicates the target class or "Label" to which a given ID belongs to. In all the complete dataset is divided into nine different classes.
- Domain of the data : E-commerce
- Number of missing data : none

Test Set (13,570,686 observations):

Attributes	Product ID'S	Feat_1	…….to....….	Feat_93	Target/label class
No. of instances	**144,368**	**144,368**	**…….....**	**144,368**	**(To be determined using algorithms)**

Explanation of the data fields:

Training Dataset consists of the following attributes:

- Product ID: Indicates the product ids, which is unique for each mentioned product in the dataset.
- Feat_1.... Feat_93: These 93 columns indicate various features which are numeric values and have been used to classify the products into nine different classes.
- Number of missing data : none

METHOD

The Software, tools and packages that have been used for the multi-classifier designing are:

- R language
- R Studio

R Language:

R has become the most popular language for data science and an essential tool for Finance and analytics-driven companies such as Google, Facebook, and LinkedIn. R includes virtually every data manipulation, statistical model, and chart that the modern data scientist could ever need.

R Studio:

RStudio is a free and open source integrated development environment (IDE) for R, a programming language for statistical computing and graphics. RStudio is available in two editions: RStudio Desktop, where the program is run locally as a regular desktop application; and RStudio Server, which allows accessing RStudio using a web browser while it is running on a remote Linux server. **For Our project we have used RStudio Desktop version.**

ALGORITHMS

The algorithms that have been used for this project are:

- Random forest (randomForest Package of R)
- Neural Network (nnet Package of R)

Random Forest: A random forest is a Meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting.

R-Language Program for RANDOM FOREST:

```
rm(list=ls())
train <- read.csv("E:/INF-550 Data Informatics/Otto/train.csv")  #Reading Training Dataset
test <- read.csv("E:/INF-550 Data Informatics/Otto/test.csv")  #Reading Testing Dataset
# Creating a CSV file to save probability of each product ID for corresponding classes
sample_sub <- read.csv("E:/INF-550 Data Informatics/Otto/sampleSubmission.csv")
train2 <- train[,-1]
library(randomForest) #Using Random forest library of R to perform the classification
set.seed(6000) # seed – The seed values are set to get the same random output variables for ntree generation which gives us same accuracy for the algorithm for the given number of ntrees.
fit <- randomForest(as.factor(target) ~ ., data=train2, importance=TRUE, ntree=500)
#ntree - No. of trees grown. #importance- tells whether the importance of the predictors should be assessed or not.
pred <- predict(fit,test,type="prob") #predicting target value for the test data
submit <- data.frame(id = test$id, pred)
write.csv(submit, file = "E:/INF-550 Data Informatics/Otto/submit-4.csv", row.names = FALSE)
```

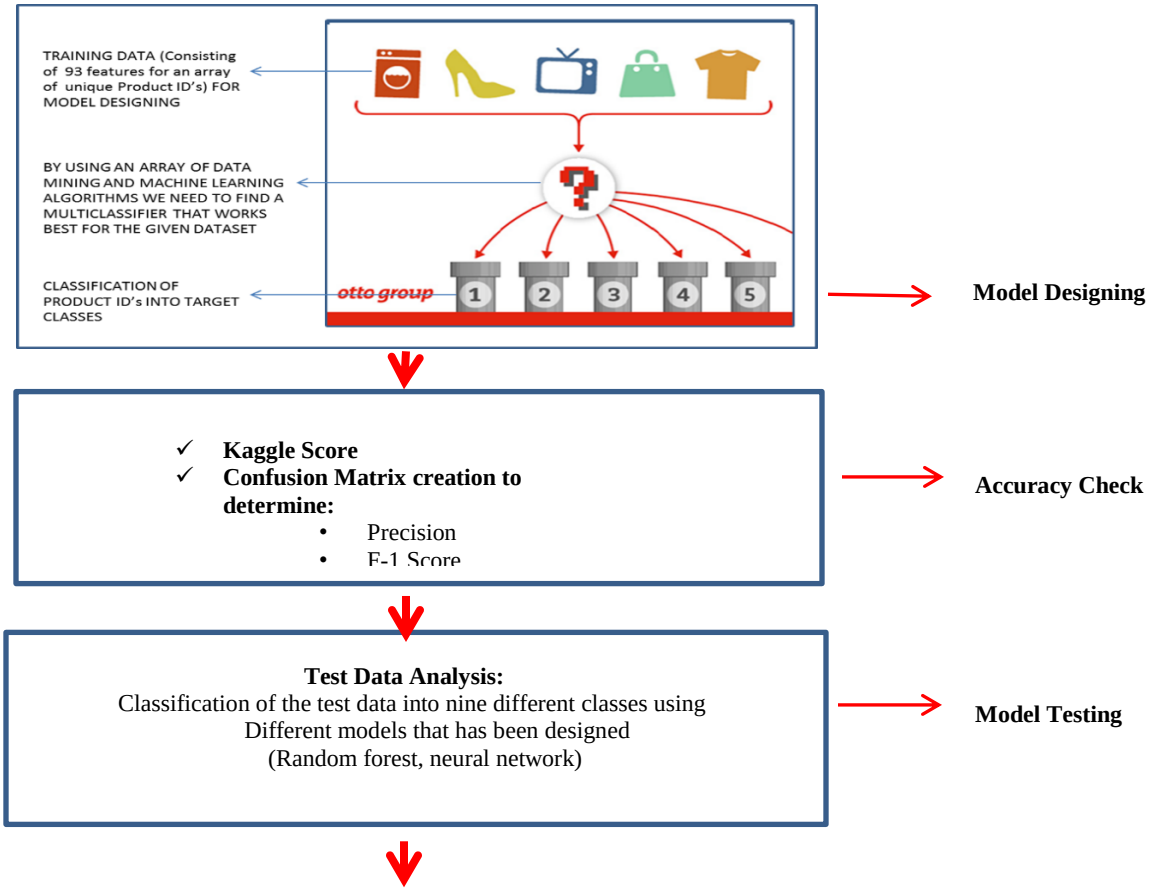
Neural Network: A neural network (NN) model is very similar to a non-linear regression model, with the exception that the former can handle an incredibly large amount of model parameters. For this reason, neural network models are said to have the ability to approximate any continuous function. Neural network is very good at learning non-linear function and also multiple outputs can be learnt at the same time.

R-Language Program for NEURAL NETWORK:

```
rm(list=ls())
train <- read.csv("E:/INF-550 Data Informatics/Otto/train.csv")#Reading Training Dataset
test <- read.csv("E:/INF-550 Data Informatics/Otto/test.csv")#Reading Training Dataset
# Creating a CSV file to save probability of each product ID for corresponding classes
sample_sub <- read.csv("E:/INF-550 Data Informatics/Otto/sampleSubmission.csv")
train2 <- train[,-1]
library(nnet) #Using Neural Network library of R to perform the classification
fit <- nnet(as.factor(target) ~ ., data=train2, size=9, maxit=10000, decay=.001)
#maxit(Maximum iteration) – maximum number of iterations the defined nnet library will work for
#decay - parameter for weight decay
pred <- predict(fit, test, type="raw") #predicting target value for test data for the model that had been designed using the training data using Random forest library
submit <- data.frame(id = test$id, pred)
write.csv(submit, file = "E:/INF-550 Data Informatics/Otto/neural1.csv", row.names = FALSE)
```

EXPERIMENT

The complete Experiment performed for the data analysis for the given dataset can be visualized as:



Graphical Representation of:
- Insights of different classification algorithms

→ **Visualization of the performance of tested Algorithms**

Model Designing:

For the project we have done designing of random forest and neural network as shown in above algorithm section.

Accuracy Check:

1. Kaggle Score

Submissions are evaluated using the multi-class logarithmic loss. The formula is then,

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij}),$$

1. N- Number of products in the test set
2. M - Number of class labels
3. Log - Natural logarithm
4. y_{ij} - 1 if observation i is in class j and 0 otherwise
5. p_{ij} - Predicted probability that observation i belongs to class j

For Kaggle score – Lower the value better the algorithm. The best Score of Kaggle for our team is:

- Random Forest Score 0.55690
- Neural Network Score 0.61399

2. Confusion Matrix Creation

- "Confusion Matrix for Random Forest"

		Predicted classes								
		Class_1	Class_2	Class_3	Class_4	Class_5	Class_6	Class_7	Class_8	Class_9
Actual class	**Class_1**	**60**	155	1	3	0	81	7	266	204
	Class_2	0	**5711**	78	55	57	50	8	221	56
	Class_3	4	328	**2567**	24	26	23	11	171	27
	Class_4	1	600	26	**309**	5	88	2	59	6
	Class_5	1	55	7	1	**978**	0	0	16	1
	Class_6	4	232	16	32	1	**4999**	18	259	67
	Class_7	3	524	49	18	8	123	**106**	288	30
	Class_8	11	575	33	5	3	205	14	**2481**	158
	Class_9	**20**	**256**	**15**	**7**	**3**	**103**	**11**	**239**	**1386**

- "Confusion matrix for Neural Networks"

		Predicted Classes								
		Class-1	Class-2	Class-3	Class-4	Class-5	Class-6	Class-7	Class-8	Class-9
Actual Class	**Class-1**	**1**	135	4	2	8	78	12	312	225
	Class-2	0	**5728**	30	43	77	61	17	264	116
	Class-3	0	2771	**36**	27	40	33	34	188	52
	Class-4	0	743	2	**167**	7	106	11	48	12
	Class-5	0	37	0	0	**994**	0	4	21	3
	Class-6	11	222	15	24	3	**4980**	33	273	67
	Class-7	1	525	20	12	9	127	**89**	333	33
	Class-8	3	613	27	8	8	302	30	**2267**	227
	Class-9	1	243	3	9	4	101	19	344	**1316**

OBSERVATION

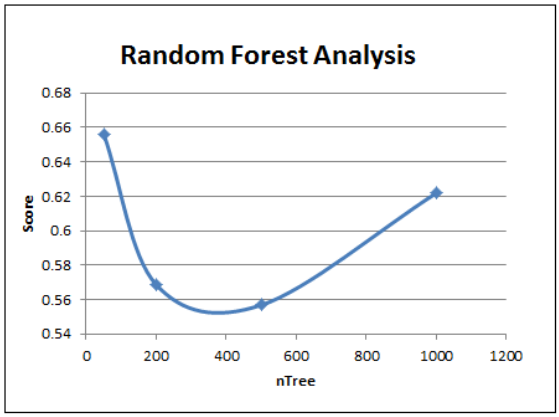
Precision and F1 score analysis:

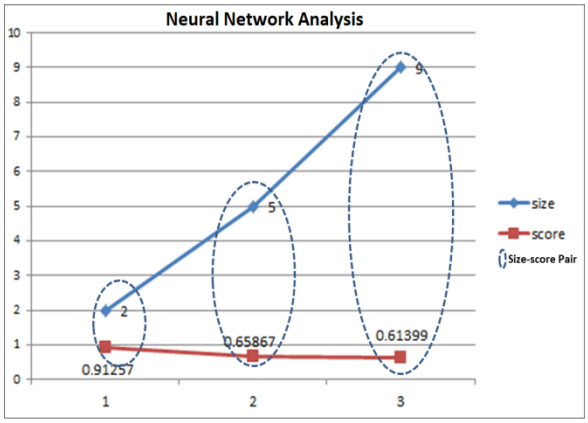
Random Forest:

- Precision – 75%
- F1-score – 75.44%

Neural Network:

- Precision - 63%
- F1-score - 62.02%





Observation of Random Forest Analysis Graph:

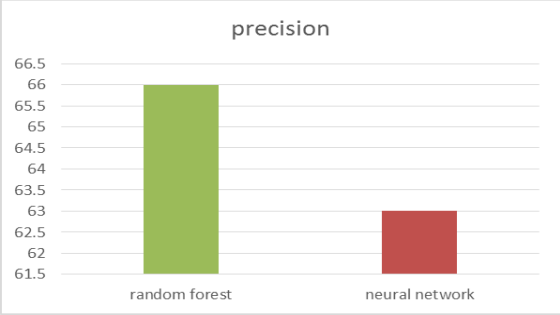
Number of tree (nTree) – Do partitioning of data into specified number of base learner tress. Final classification for Product ID is done by voting (majority) between all base learners individual tree results output. **Increment** in the **value of nTree**, **better the kaggle score** is. **(nTree – 500, Score - 0.5569 - Best Kaggle Score).** But this is not an indication of the fact that nTree should be high always. If value is more than a threshold score it leads to over fitting of model as the nTrees generated become biased towards the training data assigned. **(nTree – 1000, Score – 0.62 – Over fitting condition)**

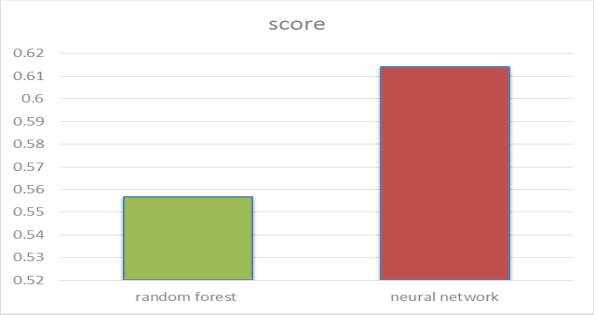
Observation of Neural Network Analysis Graph:

The **accuracy** - **Increases** (i.e. good Score) with an **increment in "size"**. "Size" – Number of units in hidden layer of the network. With an increment in size, there will be more accuracy obtained as each layer validated the data coming from former one. Graph gives a clear indication - For a high size (i.e. -9) a more accurate model can be obtained **(kaggle score improvement).** A **very high value should be avoided** as it leads to **over fitting**

CONCLUSION

The graph below shows the precision and score values of random forest technique vs neural network technique. Random forest technique has higher precision compared to neural network and the kaggle score of random forest is also more near to zero compared to neural network. These observations and analysis show that random forest is the better classification technique compared to neural network for classifying the Otto group data.





REFERENCES

1. https://www.kaggle.com/c/otto-group-product-classification-challenge/data - Dataset for project
2. http://en.wikipedia.org/wiki/No_free_lunch_theorem - No Free Lunch Theorem
3. https://www.jair.org/media/105/live-105-1426-jair.pdf - T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error correcting output codes. Journal of Artificial Intelligence Research, pages 263–286, 1995.