**PROJECT PHASE 4**

**PRODUCT CLASSIFICATION**

**CIS 508**

**FINAL PROJECT REPORT**

**Team 12-Cohort B**

Akshay Anandrao Jagadale

Juntao Zhang

Po-Chuan Chen

Aastha Rajvanshi

**CONTENTS**

**1.** **Introduction……………………………………………………………………..2**

1.1 Problem Definition

1.2 Data Description

**2.** **Initial Approach (Work Plan).............................................................................2**

2.1 Initial Understanding

2.2 Approaches in mind

**3.** **Data Pre-processing……………………………………………………………...4**

**4. Approaches tried in AZURE ML……………………………………………….4**

**5.** **Final Approach…………………………………………………………………..7**

**6.**  **Conclusion and Lessons Learned……………………………………………….8**

1. **INTRODUCTION**

The otto group product classification problem is a classification problem faced by an e-commerce company that has a huge amount of products sold everyday with a lot of different categories. In order to analyse the performance of these products, these categories play a significant role. Hence, it is important for the company to accurately classify each product in its accurate category which is not always possible due to the diverse nature of its global infrastructure.

**1.1 Problem Definition:** To classify a product in a particular class accurately is the major concern for Otto Group. Problem arises when the same type of products are classified in different categories. Therefore, the difficulty is how to classify the products more accurately by constructing a brand-new model in order to enhance the accuracy of their classification rate.

**1.2 Data Description:** The training data consists 61878 observations, and the test data consists 144368 observations with unique IDs of individual products. Each observation has 93 columns which means there are 93 features to describe each product, and each feature for each observation (the interception of column and row) has a numeric value which can be used to present its own feature.

The combination of these numeric values of the 93 features, assigns each product in a particular class. There are a total of 9 classes in the training data.

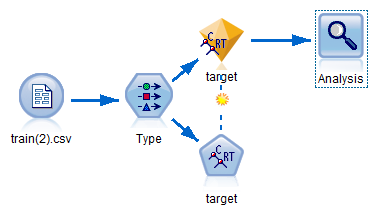
**2. INITIAL APPROACH**

**2.1 Initial approach:**

By using the training data set with different algorithms to build the model for classification. Then using the model to predict the classification for testing data, and acquire predicted classification.

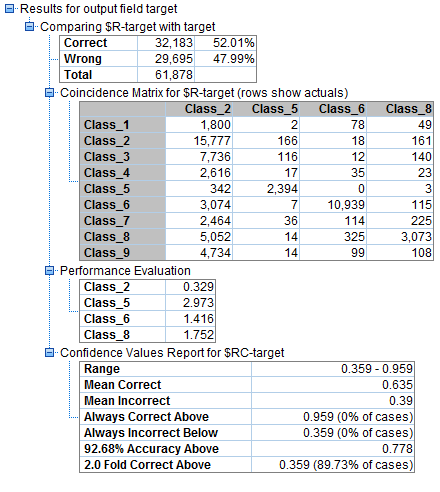
**2.1 Approaches in mind:**

To start with, we used various decision trees and got the following results in SPSS-



1. **C&R Tree:**

Setting id column to be record id and target column to be target in type node. Then, after ran the C&R Tree model with tree depth =4, we got a model and analysis.

Then, we know:

Highest confidence level: 95.9%

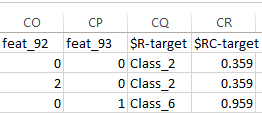
Lowest confidence level: 35.9%

Accuracy rate: 52.01%

Error rate: 47.99%

Top 3 predictors: feat\_11, feat\_14, feat\_34

Utilize the model to run with training data, and gain $R-target as predicted class and $RC-target.



Similarly, we have tried the following decision trees and tried to look for the best model.

**2. QUEST:**

Changing the role of id to none and setting the role of target column as target. Applied the Quest decision tree and ran the model which gave us the following analysis:

Highest confidence level: 86.1%

Lowest confidence level: 34.4%

Accuracy rate: 51.7%

Error rate: 48.3%

Top 3 predictors: feat\_11, feat\_34, feat\_60

**3. CHAID:**

Again, role of id is none and target column is target. We get the following result from analysis..

Highest confidence level: 99.6%

Lowest confidence level: 19.0%

Accuracy rate: 54.73%

Error rate: 45.27%

Top 3 predictors: feat\_11, feat\_60, feat\_25

**4. C5.0**

Using the type node as same as used for Quest and CHAID, the result from analysis of C5.0 is :

Highest confidence level: 99.8%

Lowest confidence level: 16.7%

Accuracy rate: 86.76%

Error rate: 13.24%

Top 3 predictors: feat\_34, feat\_11, feat\_60

We tried changing different parameters for all the above decision trees but these were the accuracy of the best models we achieved.

Looking at the models we reach the following conclusions:

1. C5.0 decision tree is giving the best accuracy out of the models used so far.
2. The top 3 predictors of the models used are very similar.
3. Features 11, 34 and 60 seem to be important in predicting the class of the product.
4. Most of the trees are not giving a good accuracy rate.

For the next steps, we will try some other models like random forest, ensemble models or neural network models to find better alternatives.

**3. Data Pre-processing:**

The otto group classification problem classifies different products into 9 different classes on the basis of the count of 93 features. After getting an accuracy of 86.76% in Phase 2 using C5.0 tree in SPSS modeler, we explore various classification models in Microsoft Azure ML. But before modeling, we processed the data in the following ways-

1. The target column in the train dataset had 5 missing values. Since, there is no point of keeping a product that has no class defined, we deleted these five rows by using the Clean Missing Data Node.
2. Not all the attributes are necessary and we need to exclude them before we train the model. Id column is one such column that we have excluded before training the model using Project Column node.
3. We need to tell Azure ML that target is the column we are predicting and it is a categorical variable. Hence, we make target a label and change it to categorical and also change all the feature columns to non categorical using Metadata Editor.

**4. Approaches tried in AZURE ML:**

We tried many different multi class models under classification in machine learning section. After data preprocessing, we tried 2 ways to run our models. By splitting data and by partitioning our data. And the results we obtained are as follows:

**Using Split Data Node -** The split data node split the data into 2 samples, 75% and 25% for training and validation respectively.

1. *Multiple Decision Forest (Bagging)-*

|  |  |
| --- | --- |
| Resampling method - Bagging  Create Trainer Mode - Single parameter  Number of decision trees - 8  Maximum depth - 32  Number of random splits - 128  Minimum number of sample -1 | Result obtained :  Screen Shot 2015-11-16 at 8.11.17 PM.png |

1. *Multiple Decision Forest (Replicate)-*

|  |  |
| --- | --- |
| Resampling method - Replicate  Create Trainer Mode - Single parameter  Number of decision trees - 8  Maximum depth - 32  Number of random splits - 128  Minimum number of sample -1 | Result obtained :  Screen Shot 2015-11-16 at 8.14.19 PM.png |

3. *Multiclass Neural Network -*

|  |  |
| --- | --- |
| Hidden Layer Specification - Fully connected case  Create Trainer Mode - Single parameter  Number of hidden nodes - 100  learning rate - 0.1  Number of learning iterations - 100  Initial Learning weight - 0.1 | Result obtained:  Screen Shot 2015-11-16 at 8.25.58 PM.png |

4. *Multiclass Decision Jungle -*

|  |  |
| --- | --- |
| Resampling method - Bagging  Create Trainer Mode - Single parameter  Number of decision - 8  Maximum depth - 32  Maximum width - 128  Number of optimization steps -2048 | Result obtained:  Screen Shot 2015-11-16 at 8.33.31 PM.png |

5. Multiclass Logistic regression -

|  |  |
| --- | --- |
| Create Trainer Mode - Single parameter  Optimization Tolerance - 1E-07  L1 regularization weight- 1  L2 regularization weight - 1  Memory size - 20 | Result obtained: |

**Using partition and sample node:**

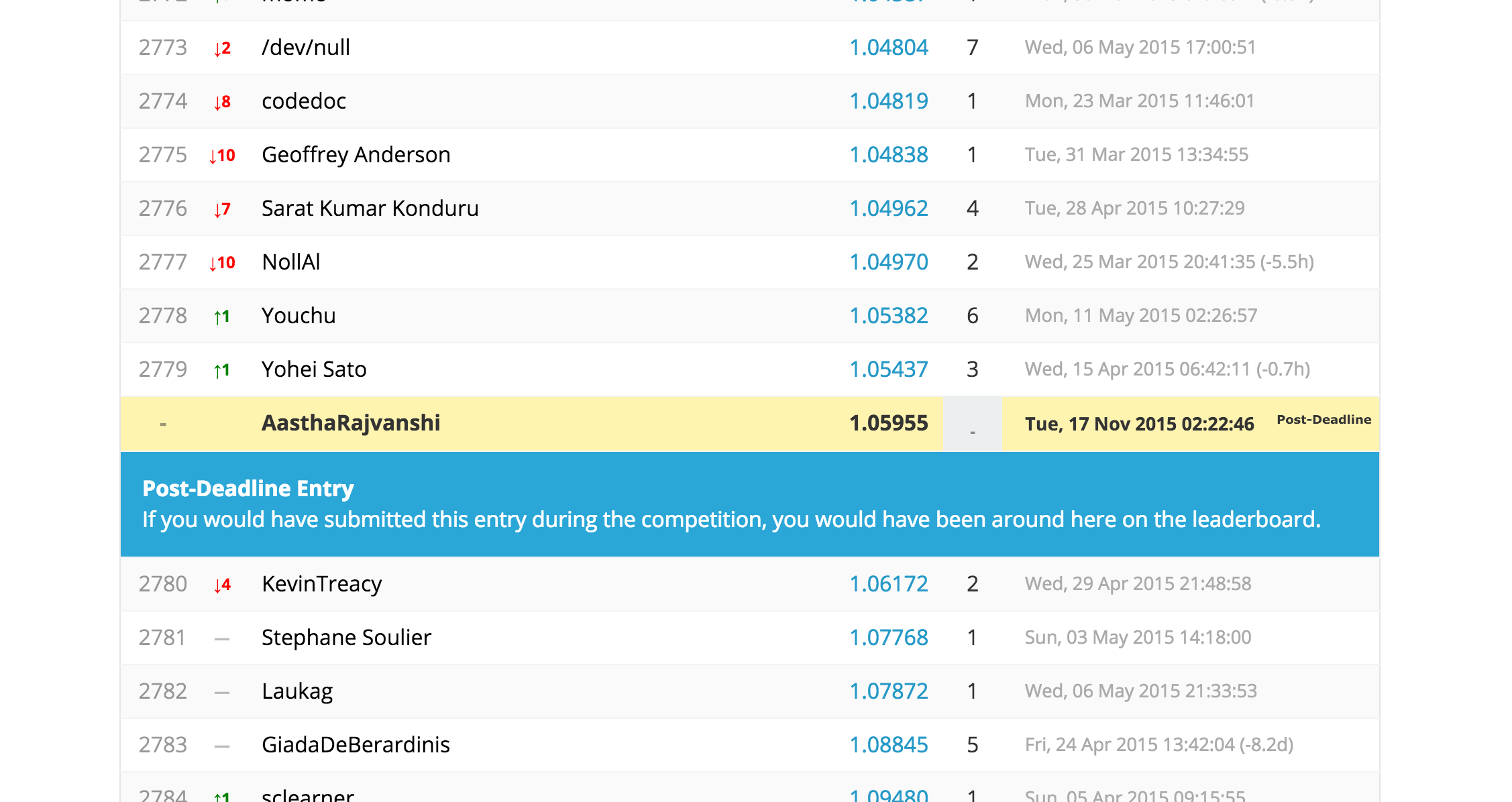
1. *Multiclass decision forest-*

|  |  |
| --- | --- |
| Resampling method - Bagging  Create Trainer Mode - Single parameter  Number of decision trees - 15  Maximum depth - 32  Number of random splits - 128  Minimum number of sample -1 | Result obtained-  Screen Shot 2015-11-16 at 8.51.36 PM.png |

*2. Multiclass Neural Network-*

|  |  |
| --- | --- |
| Hidden Layer Specification - Fully connected case  Create Trainer Mode - Single parameter  Number of hidden nodes - 100  learning rate - 0.1  Number of learning iterations - 100  Initial Learning weight - 0.1  Min-max Normalizer | Result obtained-  Screen Shot 2015-11-16 at 8.53.32 PM.png |

Since we got the best accuracy for Multiclass Decision Forest using partition and sample, we submitted it on kaggle and got a score of 1.05955.



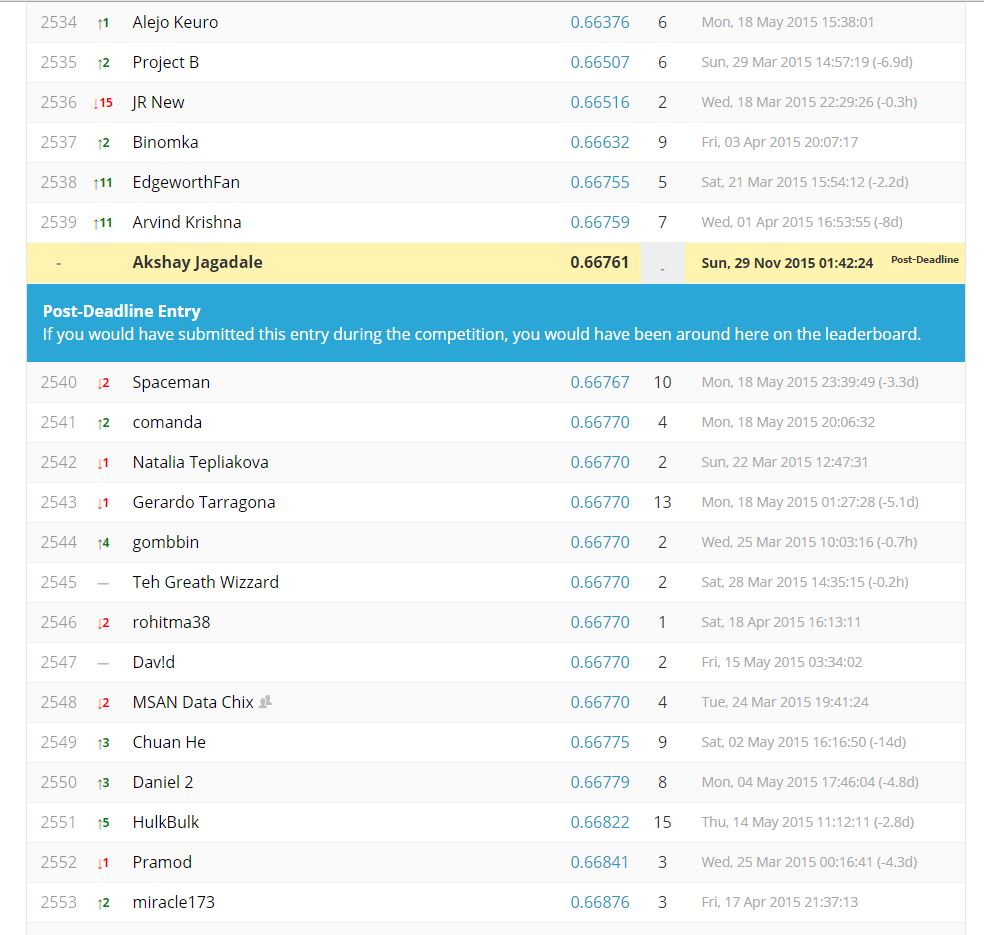
**5. Final Approach :**

After trying a lot of different models, we observed that Multiple decision forest is able to give the best prediction so far. So we tried changing the parameters and ensembled 50 decision trees to get a decision forest and achieved the following result-

**Using partition and sample node:**

1. *Multiclass decision forest-*

|  |  |
| --- | --- |
| Resampling method - Bagging  Create Trainer Mode - Single parameter  Number of decision trees - 50  Maximum depth - 32  Number of random splits - 128  Minimum number of sample -1 | Result obtained-  C:\Users\Akshay Jagadale\AppData\Local\Microsoft\Windows\INetCache\Content.Word\multidecision forest with 50 trees accuracy.jpg |



**6. Conclusion and lessons learned:**

Working on this project, we tried a lot of new algorithms and softwares. We started off with SPSS modeler. Since the data was huge, SPSS took a lot more time in classification than Microsoft Azure ML. So we moved on to Microsoft Azure ML. Data preprocessing was much simpler here. We removed some missing data and made our attributes non categorical. Running decision forests, neural network, logistic regression, decision jungle by splitting data 75-25 still gave a low accuracy. Then we learned that by partitioning our data into folds may lead to better results. Through continuous trial and error, we came to the conclusion that decision forest would be our best shot at this problem and so we focussed on decision forests to get better results. Trying different ensembles, we got out best result by ensembling 50 decision trees.

This project through its various stages, helped us understand how to treat data, what to expect and how to improve. We understood how these algorithms work differently in different kinds of problems.