MGS 616: PREDICTIVE ANALYTICS PROJECT-4

School of Management, University at Buffalo, The State University of New York

By Sreyashi Samanta

Title: *Comparative Analysis of Predictive Models for US Accidents Severity Classification*

ABSTRACT/INTRODUCTION: This report presents the findings of a study conducted on the US Accidents dataset, intending to develop predictive models to classify accident severity into four categories: Fatal, Injury, Property Damage Only, and Unknown Severity. Eight machine learning algorithms were employed and evaluated based on classification metrics.

METHODOLOGY: The dataset is pre-processed by selecting relevant features and handling missing values. The response variable "Severity" is encoded into four classes and is categorical in nature. In case of each Algorithm, the training dataset, which consists of 70% of the total dataset, is used to train the predictive models. The remaining 30% of the dataset is kept as the testing dataset to evaluate the performance of the models on unseen data. Through analysis and experimentation with different feature combinations, we observed that including "Wind_Speed", "Visibility", "Temperature", "Pressure", "Weather_Condition", "Traffic_Signal", "Bump", "Crossing" elements in the predictive models resulted in improved accuracies.

As the response variable of the dataset is a multiclass categorical variable and the dataset consists of 4714 rows and 18 columns, the following models were considered:

1. Multinomial Logistic Regression: It fits a logistic regression model to the training data by estimating the probabilities of each severity class and assigns the class with the highest probability as the predicted class.
2. Random Forest: It constructs an ensemble of decision trees. Each tree is trained on a bootstrap sample of the training data and a random subset of features.
3. Support Vector Machines (SVM): It constructs a hyperplane that maximally separates the instances of different severity classes. It finds the optimal decision boundary by maximizing the margin between the classes.

4. Gradient Boosting Machine (GBM): It builds an ensemble of weak learners in a sequential manner. Each subsequent tree is trained to correct the mistakes made by the previous trees.
5. K-Nearest Neighbors (KNN): It classifies instances based on their proximity to the nearest neighbors in the feature space. It calculates the distance between instances and assigns the class label based on the majority vote of the k nearest neighbors.
6. Extreme Gradient Boosting (XGBoost): Is an advanced gradient boosting algorithm known for its speed and performance.
7. Naïve Bayes: It calculates the probabilities of each class based on the feature values and assigns the class with the highest probability as the predicted class.
8. Decision Tree: It splits the data based on the features to create nodes and leaf nodes, which represent the predicted class labels.

RESULTS:

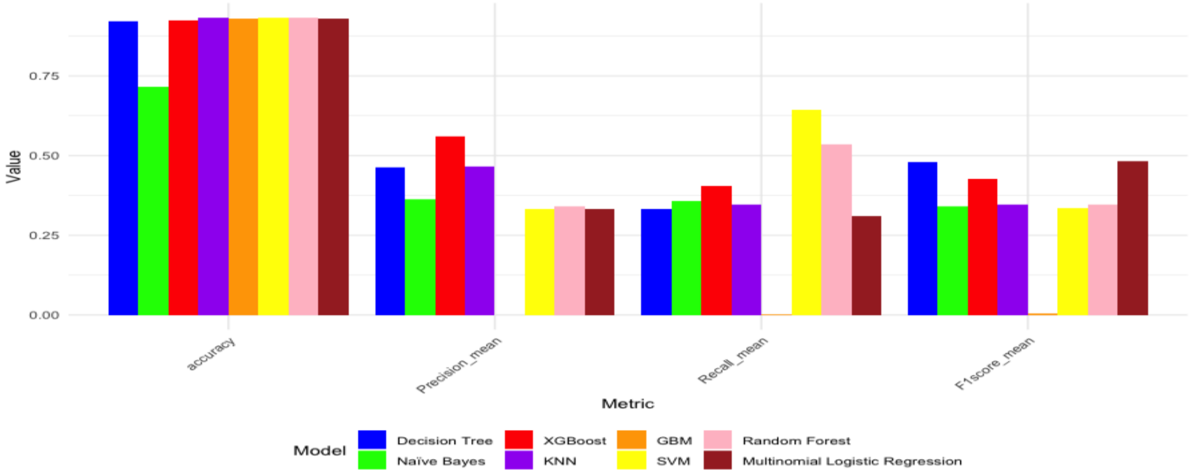
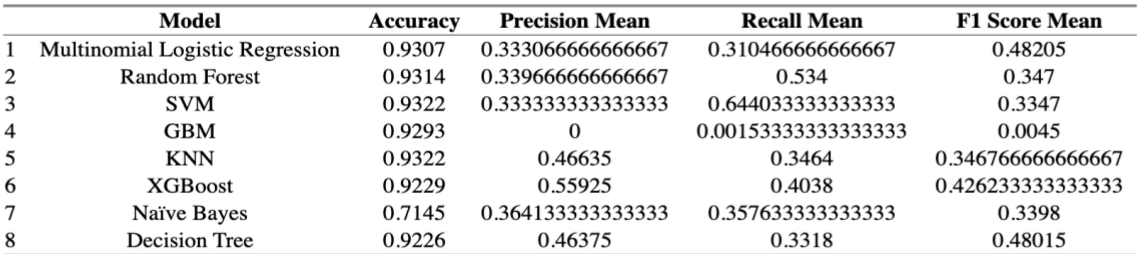


Figure 1: Gives a consolidated view of each model for the considered performance metrics in a bar chart.



	Model	Accuracy	Precision Mean	Recall Mean	F1 Score Mean
1	Multinomial Logistic Regression	0.9307	0.333066666666667	0.310466666666667	0.48205
2	Random Forest	0.9314	0.339666666666667	0.534	0.347
3	SVM	0.9322	0.333333333333333	0.644033333333333	0.3347
4	GBM	0.9293	0	0.00153333333333333	0.0045
5	KNN	0.9322	0.46635	0.3464	0.346766666666667
6	XGBoost	0.9229	0.55925	0.4038	0.426233333333333
7	Naïve Bayes	0.7145	0.364133333333333	0.357633333333333	0.3398
8	Decision Tree	0.9226	0.46375	0.3318	0.48015

Figure 2: Gives a consolidated view of each model for the considered performance metrics in a tabular form.

Performance Evaluation: While accuracy represents overall model performance, it is essential to calculate precision, recall, and F1-score to assess the models comprehensively. Considering the multi-class nature of the "Severity" prediction and the need to evaluate the models, classification metrics such as accuracy, precision, recall, and F1-score, along with the confusion matrix, would be appropriate for assessing the predictive models' performance on the dataset.

1. Accuracy: All the models achieved reasonable accuracies ranging from 0.7145 to 0.9322, indicating their ability to predict the severity. The models with the highest accuracies are Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Random Forest, all achieving accuracies around 0.93.
2. Precision: Ability of a model to correctly identify positive cases. XGBoost and Support Vector Machines (SVM) show high precision for some classes, while K-Nearest Neighbors (KNN) demonstrates consistent precision across all classes.
3. Recall: XGBoost, Random Forest, and SVM exhibit high recall values for different classes, indicating their ability to capture true positive instances.
4. F1-score: Multinomial Regression, Decision Tree, and XGBoost show competitive F1 scores, indicating a good balance between precision and recall for the predicted classes.

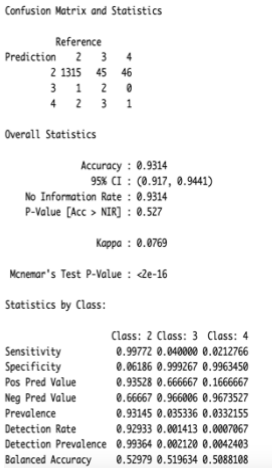
Based on the evaluation metrics, XGBoost, SVM, KNN, Random Forest models, and Decision Tree perform better than the other models for predicting the Severity of US accidents. Although the Naïve Bayes model has a relatively lower accuracy, it has the highest precision and recall score for predicting Severity level 2 which indicates that Naïve Bayes is particularly good at predicting Severity level 2. Although KNN has a higher accuracy, the consistent performance across precision, recall, and F1 score makes XGBoost better for evaluating the reliability of accident severity.

CONCLUSION: The ability to predict the severity of accidents enables stakeholders to allocate resources efficiently, prioritize emergency response, and develop targeted strategies for accident prevention. In this study, we evaluated multiple machine learning models using the US Accidents dataset and found that the XGBoost model consistently outperformed other models.

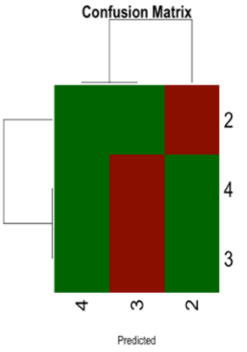
Future Directions: To enhance the severity classification models, further investigation can be conducted to optimize hyperparameters, feature selection, and handling imbalanced classes. Lastly, ensemble techniques like stacking and boosting can be utilized to combine the predictions of multiple severity classification models.

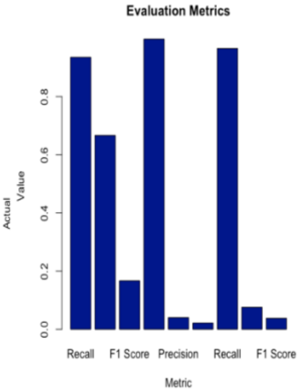
Appendix:

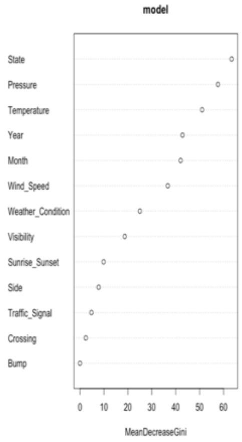
1. Random Forest



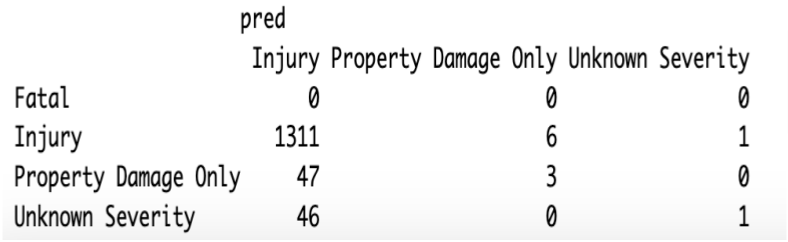
```
Confusion Matrix and Statistics

          Reference
Prediction    2    3    4
         2 1315   45   46
         3    1    2    0
         4    2    3    1

Overall Statistics

               Accuracy : 0.9314
                 95% CI : (0.917, 0.9441)
    No Information Rate : 0.9314
    P-Value [Acc > NIR] : 0.527

                  Kappa : 0.0769

 Mcnemar's Test P-Value : <2e-16

Statistics by Class:

                     Class: 2 Class: 3  Class: 4
Sensitivity           0.99772 0.040000 0.0212766
Specificity           0.06186 0.999267 0.9963450
Pos Pred Value        0.93528 0.666667 0.1666667
Neg Pred Value        0.66667 0.966006 0.9673527
Prevalence            0.93145 0.035336 0.0332155
Detection Rate        0.92933 0.001413 0.0007067
Detection Prevalence  0.99364 0.002120 0.0042403
Balanced Accuracy     0.52979 0.519634 0.5088108
```



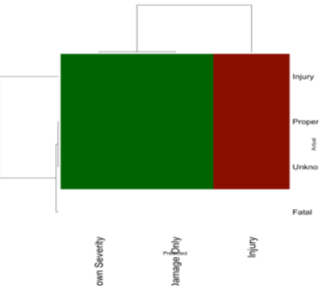




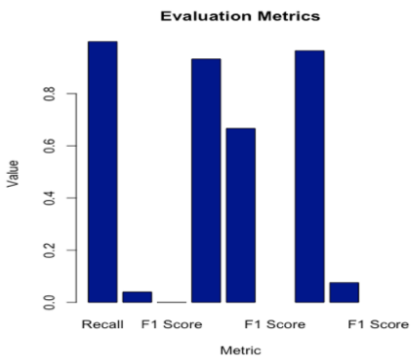
2. GBM

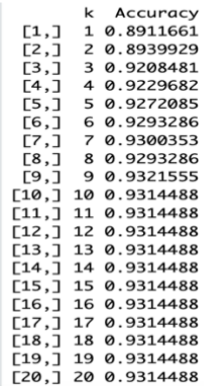


```
                   pred
                    Injury Property Damage Only Unknown Severity
Fatal                    0                    0                0
Injury                1311                    6                1
Property Damage Only    47                    3                0
Unknown Severity        46                    0                1
```

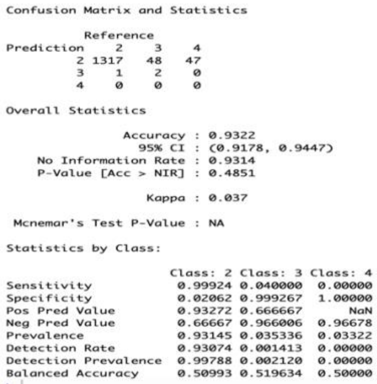


3. KNN



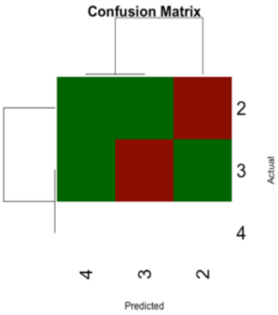


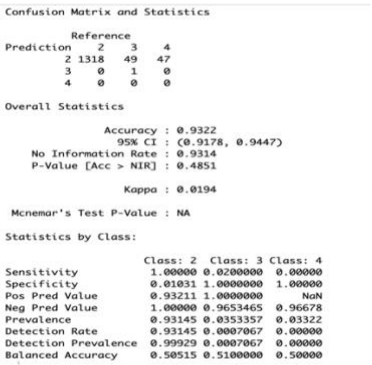
```
      k  Accuracy
 [1,]  1 0.8911661
 [2,]  2 0.8939929
 [3,]  3 0.9208481
 [4,]  4 0.9229682
 [5,]  5 0.9272085
 [6,]  6 0.9293286
 [7,]  7 0.9300353
 [8,]  8 0.9293286
 [9,]  9 0.9321555
[10,] 10 0.9314488
[11,] 11 0.9314488
[12,] 12 0.9314488
[13,] 13 0.9314488
[14,] 14 0.9314488
[15,] 15 0.9314488
[16,] 16 0.9314488
[17,] 17 0.9314488
[18,] 18 0.9314488
[19,] 19 0.9314488
[20,] 20 0.9314488
```



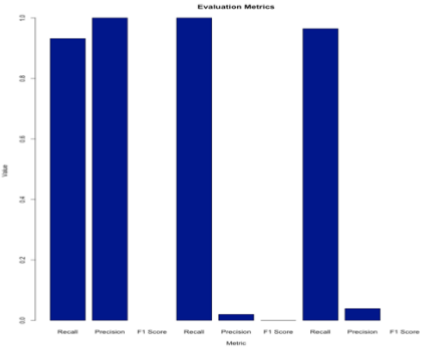
```
Confusion Matrix and Statistics

          Reference
Prediction    2    3    4
         2 1317   48   47
         3    1    2    0
         4    0    0    0

Overall Statistics

               Accuracy : 0.9322
                 95% CI : (0.9178, 0.9447)
    No Information Rate : 0.9314
    P-Value [Acc > NIR] : 0.4851

                  Kappa : 0.037

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 2 Class: 3 Class: 4
Sensitivity           0.99924 0.040000  0.00000
Specificity           0.02062 0.999267  1.00000
Pos Pred Value        0.93272 0.666667      NaN
Neg Pred Value        0.66667 0.966006  0.96678
Prevalence            0.93145 0.035336  0.03322
Detection Rate        0.93074 0.001413  0.00000
Detection Prevalence  0.99788 0.002120  0.00000
Balanced Accuracy     0.50993 0.519634  0.50000
```

4. SVM

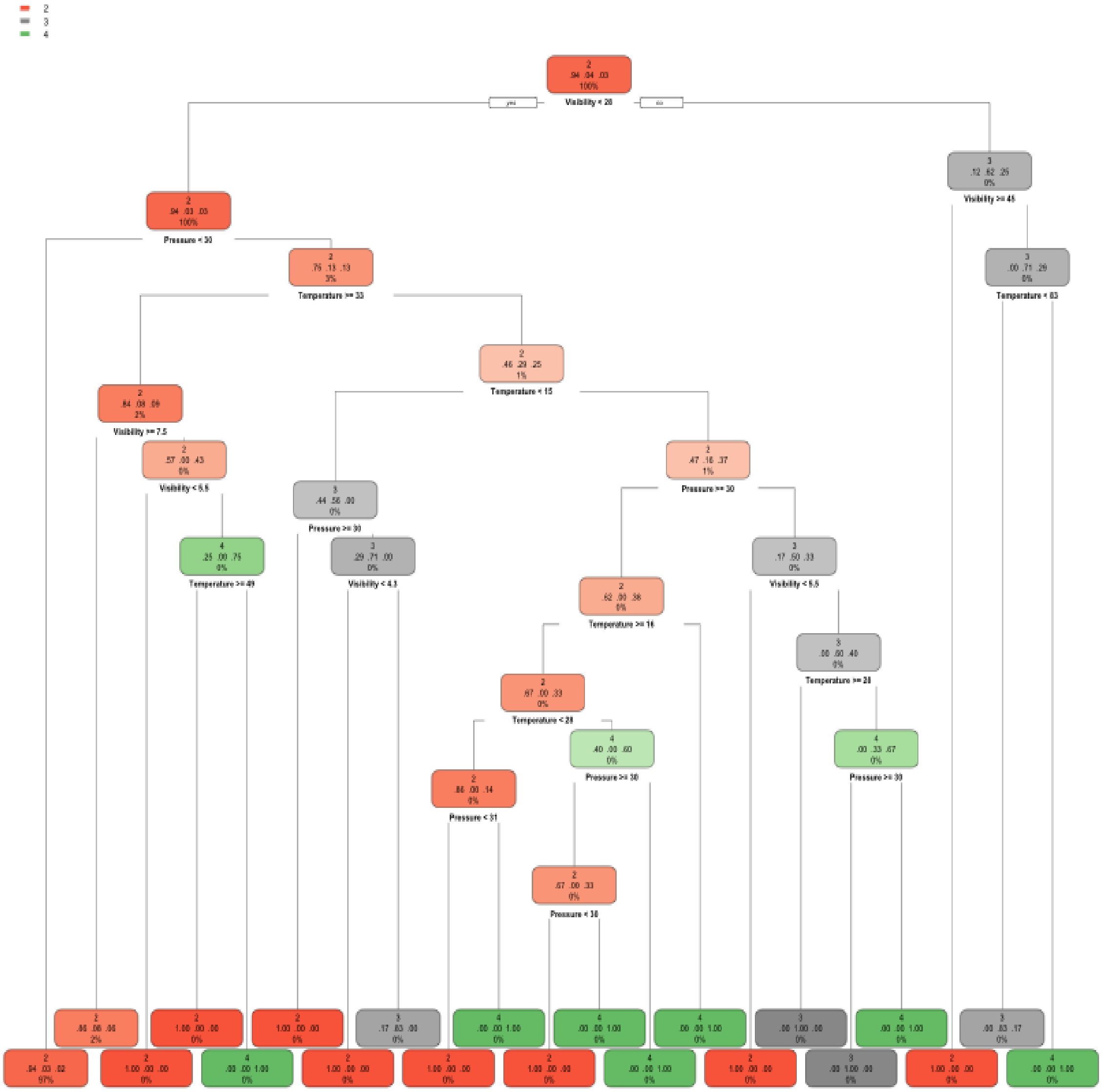




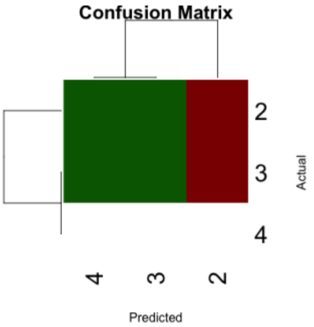
```
Confusion Matrix and Statistics

          Reference
Prediction    2    3    4
         2 1318   49   47
         3    0    1    0
         4    0    0    0

Overall Statistics

               Accuracy : 0.9322
                 95% CI : (0.9178, 0.9447)
    No Information Rate : 0.9314
    P-Value [Acc > NIR] : 0.4851

                  Kappa : 0.0194

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 2  Class: 3 Class: 4
Sensitivity           1.00000 0.0200000  0.00000
Specificity           0.01031 1.0000000  1.00000
Pos Pred Value        0.93211 1.0000000      NaN
Neg Pred Value        1.00000 0.9653465  0.96678
Prevalence            0.93145 0.0353357  0.03322
Detection Rate        0.93145 0.0007067  0.00000
Detection Prevalence  0.99929 0.0007067  0.00000
Balanced Accuracy     0.50515 0.5100000  0.50000
```

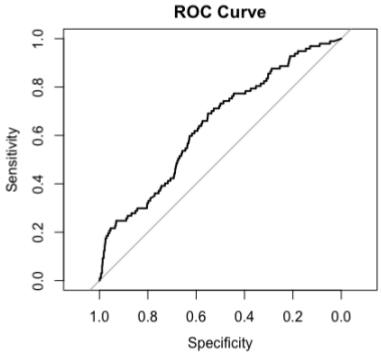


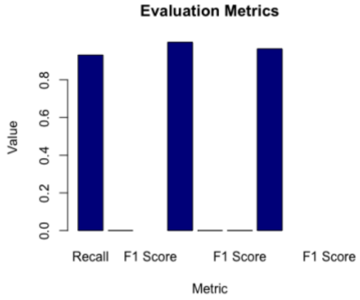
5. Decision Tree



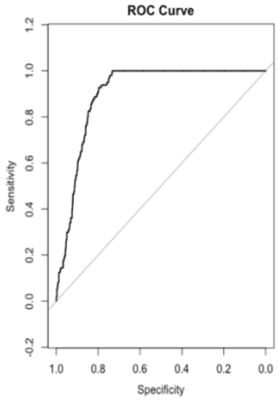
6. Multinomial Logistic Regression

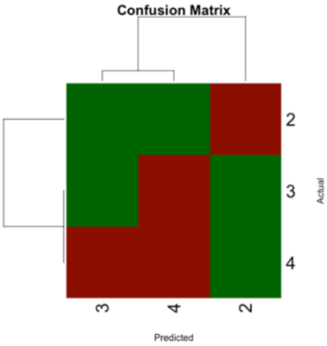


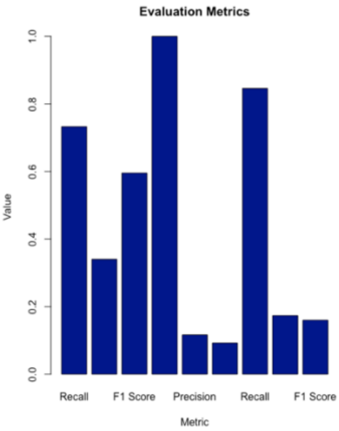




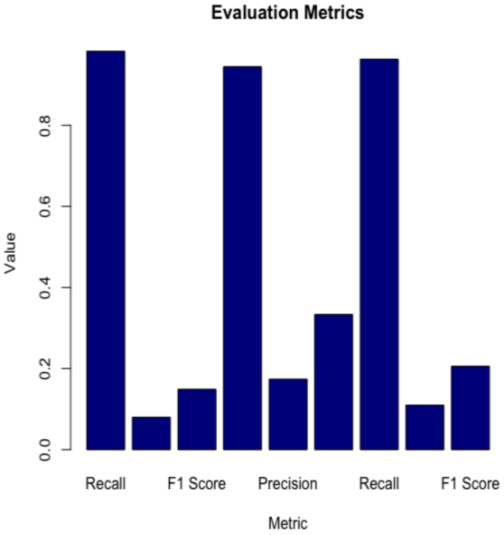
7. Naïve Bayes

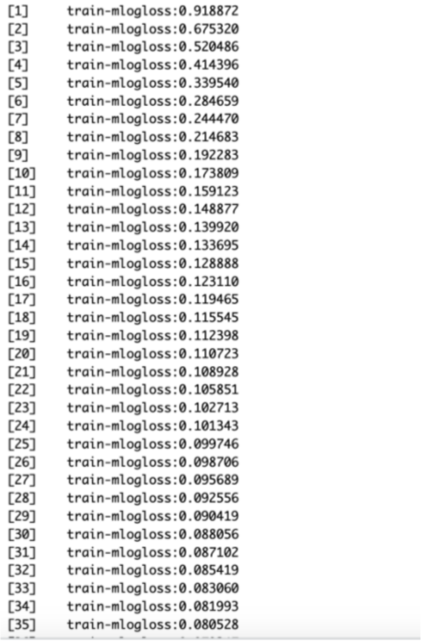






8. XGBoost





```
[1]     train-mlogloss:0.918872
[2]     train-mlogloss:0.675320
[3]     train-mlogloss:0.520486
[4]     train-mlogloss:0.414396
[5]     train-mlogloss:0.339540
[6]     train-mlogloss:0.284659
[7]     train-mlogloss:0.244470
[8]     train-mlogloss:0.214683
[9]     train-mlogloss:0.192283
[10]    train-mlogloss:0.173809
[11]    train-mlogloss:0.159123
[12]    train-mlogloss:0.148877
[13]    train-mlogloss:0.139920
[14]    train-mlogloss:0.133695
[15]    train-mlogloss:0.128888
[16]    train-mlogloss:0.123110
[17]    train-mlogloss:0.119465
[18]    train-mlogloss:0.115545
[19]    train-mlogloss:0.112398
[20]    train-mlogloss:0.110723
[21]    train-mlogloss:0.108928
[22]    train-mlogloss:0.105851
[23]    train-mlogloss:0.102713
[24]    train-mlogloss:0.101343
[25]    train-mlogloss:0.099746
[26]    train-mlogloss:0.098706
[27]    train-mlogloss:0.095689
[28]    train-mlogloss:0.092556
[29]    train-mlogloss:0.090419
[30]    train-mlogloss:0.088056
[31]    train-mlogloss:0.087102
[32]    train-mlogloss:0.085419
[33]    train-mlogloss:0.083060
[34]    train-mlogloss:0.081993
[35]    train-mlogloss:0.080528
```

REFERENCES

1. https://cran.r-project.org/web/packages/PRROC/vignettes/PRROC.pdf
2. https://www.ibm.com/topics/overfitting
3. https://builtin.com/data-science/step-step-explanation-principal-component-analysis#
4. https://link.springer.com/article/10.1007/s42452-020-3125-1
5. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8915487/
6. https://appsilon.com/r-xgboost/
7. https://towardsdatascience.com/understanding-random-forest-58381e0602d2
8. https://stats.oarc.ucla.edu/stata/dae/multinomiallogistic-regression/
9. https://towardsdatascience.com/multinomial-logistic-regression-in-r-428d9bb7dc70
10. https://www.listendata.com/2017/01/support-vector-machine-in-r-tutorial.html
11. https://neptune.ai/blog/performance-metrics-in-machine-learning-complete-guide
12. https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-regression-evaluation-metrics-1ca3e282a2ce
13. https://cran.r-project.org/web/packages/gbm/gbm.pdf
14. https://www.rdocumentation.org/packages/e1071/versions/1.7-13/topics/naiveBayes
15. https://medium.com/dataman-in-ai/a-wide-choice-for-modeling-multi-class-classifications-d97073ff4ec8