Suggesteria Programming Assignment-1 Report

Google Developers Group Project (GDG)

By-> Keshav Bansal

Colab File Link -> Suggesteria-Programming-Assignment-1.ipynb

Online Shoppers Purchasing Intention Dataset

Dataset Overview:

Source ->
https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset

The dataset captures the purchasing behavior of online shoppers, with 10 numerical features and 8 categorical features. The target variable **Revenue** indicates whether a session resulted in a purchase (True) or not (False).

Numerical features explanation:

1. **Administrative**: Count of administrative pages visited.
2. **Administrative Duration**: Total time spent on administrative pages.
3. **Informational**: Count of informational pages visited.
4. **Informational Duration**: Total time spent on informational pages.
5. **Product Related**: Count of product-related pages visited.
6. **Product Related Duration**: Total time spent on product-related pages.
7. **Bounce Rate**: Percentage of sessions that exited the site after viewing one page.
8. **Exit Rate**: Percentage of sessions where a page was the last one viewed.
9. **Page Value**: Average value of a page, reflecting its contribution to eventual transactions.
10. **Special Day**: Proximity of the session to a special day (e.g.Mother's Day).

Categorical features explanation:

1. **Operating System**: OS used during the session.
2. **Browser**: Browser used during the session.
3. **Region**: Geographical region of the session.
4. **Traffic Type**: Type of traffic source.
5. **Visitor Type**: Whether the visitor is new or returning.
6. **Weekend**: Boolean indicating if the session occurred on a weekend.
7. **Month**: Month when the session took place.
8. **Revenue** (Target Variable): Boolean indicating if a purchase was made.

Key Observations:

- **Class Imbalance**: Only 15.5% of sessions result in purchases, posing challenges for classification.
- Range of features: Numerical features vary significantly in scale (e.g., BounceRates vs. ProductRelated Duration). **Used Standard Scaler to address this issue .**

Addressing Class Imbalance:

I could have resolved the issue of Class Imbalance using oversampling the minority class (By duplicating entries) or undersampling the majority class to balance the dataset. Rather I have used the SMOTE (Synthetic Minority Over-sampling Technique) to generate new samples for the minority class .

Implemented Algorithms

1) K-Nearest Neighbors (KNN)

- A distance-based algorithm that predicts a class based on the majority class of its k nearest neighbors.
- Sensitive to feature scaling and requires normalization.

2) **Logistic Regression**

- A linear model for binary classification that estimates the probability of a class using a logistic function.
- A good algorithm for binary classification like in this case (True/False).

3) **Random Forest**

- An ensemble method that builds multiple decision trees and aggregates their outputs for prediction.
- Single decision tree is prone to overfitting , Random Forest addresses this by averaging out predictions from multiple decision trees .

4) **Gradient Boosting**

- An ensemble technique that builds trees sequentially, minimizing prediction errors at each step.
- **Effective for imbalanced datasets.**

5) **AdaBoost**

- A boosting algorithm that combines weak classifiers (e.g., decision stumps) to improve overall performance.

6) **Support Vector Machine (SVM)**

- Constructs a hyperplane in a high-dimensional space to separate classes.
- Kernel trick allows handling non-linear relationships.

7) **XGBoost (Extreme Gradient Boosting):**

- It is an ensemble learning technique based on gradient boosting. It builds sequential models where each new model corrects the errors of the previous one, optimizing for speed and performance using advanced regularization (L1 & L2).

8) **Naive Bayes**

- It is a simple and fast probabilistic model based on Bayes' Theorem. It assumes conditional independence among features, making it computationally efficient

Comparison of Classification Accuracies

(with Class Imbalance)

Algorithm	**Accuracy**	**ROC-AUC**
KNN	87%	0.79
Logistic Regression	88%	0.87
Random Forest	90%	0.92
Gradient Boosting	90%	0.93
AdaBoost	89%	0.91
SVM	88%	0.85
Naive Bayes	78%	0.80
XGBoost	89%	0.92

Comparison of Classification Accuracies

(without Class Imbalance)

Algorithm	**Accuracy**	**ROC-AUC**
KNN	89%	0.95
Logistic Regression	84%	0.90
Random Forest	94%	0.99
Gradient Boosting	92%	0.98
AdaBoost	90%	0.96
SVM	87%	0.94
Naive Bayes	71%	0.83
XGBoost	94%	0.99

Insights on Algorithm Performance

1. **Best Performing Algorithm (on Class Imbalanced data):**

    - **Gradient Boost** achieved the highest ROC-AUC (0.93) and strong accuracy (90%), demonstrating its ability to handle the imbalanced dataset effectively.

2. **Best Performing Algorithm (on Class Balanced data):**

    - **XGBoost** and **Random Forest** Classifiers achieved the highest ROC-AUC (0.99) and strong accuracy of (94%) .
    - **SMOTE** was crucial for addressing the class imbalance.
    - Ensemble methods like XGBoost and Random Forest handled the imbalance better than KNN or Logistic Regression.

Conclusion

- Ensemble methods particularly **Random Forest** and **XGBoost**(Extreme Gradient Boosting) provided the best performance for predicting purchasing intentions.
- Addressing class imbalance was essential for reliable model evaluation.