Sentiment Analysis for Auto Reviews

Vishal Bachal

Abstract

This project designs, implements, and evaluates classifiers that would be able to recognize sentiment in automotive e-commerce reviews. Here we present an unsupervised clustering approach using K-means and a discriminative classification approach using XGBoost. Our experiments result in models that demonstrate high silhouette scores for K-means and very high accuracies for both F1 scores and XGBoost. The results obtained show that both models are effective in sentiment classification, but each of them exhibits advantages over the others with respect to clustering quality and predictive accuracy. The report further explores all preprocessing, model selection rationale, and the analysis of performance hence, a complete insight into practical applications in sentiment analysis.

1 Task 1: Model Selection

1.1 Summary of 2 selected Models

In this project, we selected two sentiment classification models: unsupervised learning based on K-means clustering and discriminative classification based on XGBoost.

K-means Clustering: K-means is an effective unsupervised learning algorithm that divides the input data into some number of clusters based on similarity in some feature between different samples. It iteratively assigns and reassigns data points to clusters and updates the cluster centroids till convergence. It works effectively to analyze patterns from the data without needing any information on the labeled samples. The performance of K-means clustering can be evaluated with the Silhouette score, which gives a measure of how similar an observation is to its own cluster compared to other clusters(Rousseeuw, 1987). We carried out our implementation on the Scikit-learn library, providing easy-to-use and efficient tools for machine learning(Pedregosa et al., 2011).

XGBoost is a powerful and scalable machine learning algorithm. It is perfectly for solving problems of classification and regression. It works by the implementation of gradient boosting procedures in developing an ensemble of many decisions trees that refine the model in terms of a loss function. XGBoost is highly efficient in its performance and optimized to handle large data sets characterized by imbalanced classes. The hyperparameters make it possible to tune the model, hence achieve better accuracy and F1 scores(Chen and Guestrin, 2016).

1.2 Critical discussion and justification of model selection

K-means Clustering: This is used because of its simplicity and effectiveness in classifying data points into categories, which will depend on feature similarity. It does not use labels and hence falls under unsupervised learning; therefore, it is a good tool for exploratory data analysis. This plays a particularly major advantage in identifying underlying patterns and natural groupings within the dataset when no prior knowledge was used. The use of the Silhouette score as an evaluation metric helps in assessing the quality of the clusters formed(Rousseeuw, 1987).The implementation of K-means in Scikit-learn gives a robust and efficient way to apply this algorithm, with benefits ranging from optimization to ease of use of the library, among others(Pedregosa et al., 2011). However, K-means has limitation in that it requires the number of clusters to be preset and is sensitive to initial centroid placement. Despite these difficulties, K-means is still a very powerful algorithm for unsupervised sentiment analysis. XGBoost: We chose XGBoost because it is an optimized implementation and highly scalable for supervised learning. Known for its robustness, speed, and the ability to handle large and complex data with imbalanced classes, XGBoost has an ensemble learning approach in which multiple weak learners are com-

bined to form a strong learner(Chen and Guestrin, 2016). It would provide a wide range of hyperparameters that can be tuned for the best fit in the model; hence, it is pretty flexible and adaptive to different types of data. A fundamental strength of XGBoost lies in its means to undertake regularization; hence, it inhibits overfitting and increases the ability to generalize. However, XGBoost may be computationally intensive and require a large amount of resources while training on very large datasets. In light of this imbalanced handling having more advantages in terms of flexibility and accuracy, XGBoost could be a great choice for sentiment classification tasks.

2 Task 2: Design and implementation of classifiers

2.1 Dataset Details:

This project is using an e-commerce platform to deal with automobile reviews. The classification model built from this dataset will be using the training, validation, and test sets as its subsets. More specifically, 3,681 examples are for training, 454 for validation, and 409 for testing in classification. There are two classes in total: either a positive or a negative sentiment. The training set contains 82.31% positive and 17.69% negative reviews. It seems there is an imbalance in the classes. There are 82.38% positive and 17.62% negative reviews in the validation set. Imbalance between classes is common among datasets related to real life and is corrected to avoid biased predictions by the model. These datasets are described in the table below:

Dataset	Total	% Positive	% Negative
Train	3681	82.31%	17.69%
Valid	454	82.38%	17.62%
Test	409	N/A	N/A

Table 1: Dataset Details

Text data has to go through preprocessing before it is fed into machine learning models. These important two steps in preprocessing are cleaning and vectorization. This can be achieved through various ways, including lemmatization, which reduces words to their basic form. This normalization of the input will reduce dimensionality. TF-IDF vectorization follows in an attempt to convert the text into numerical features. An approach involves considering the word frequency weighted by the inverse document frequency and then multiplying the importance of those words by their commonality between documents(Scott, 2019). The approach processes data with respect to effective learning and classification, giving the best solution to the challenge of high dimensionality by focusing on only informative features of the data, thus improving model accuracy in sentiment prediction.

2.2 Model Implementation

Overview of Preprocessing

In terms of preprocessing, the text data is lemmatized to reduce words to their base forms, standardizing the input and reducing dimensionality. This step is crucial for handling variations in word forms. The lemmatized text is then transformed into numerical features using the TF-IDF vectorizer, which converts the text data into numerical values reflecting the importance of words in the entire dataset. The diagrams below show the complete process of preprocessing, feature extraction, and model evaluation for both pipelines.

K-means Clustering

The K-means clustering pipeline uses several important hyperparameters. The lemmatizer reduces words to base forms. It is a TF-IDF vectorizer that extracts features from text with up to 3000 features, includes both unigrams and bigrams and removes English stopwords. The 'to array' step converts the TF-IDF vectors into appropriate array format. UMAP is responsible for reducing dimensionality of feature vectors to 2 components, where the random state is set based on student ID for consistency. Finally, K-means clustering groups data into 2 clusters using random state depended on students' ID and 10 initializations(n_init=10) to ensure stable clustering outcomes.

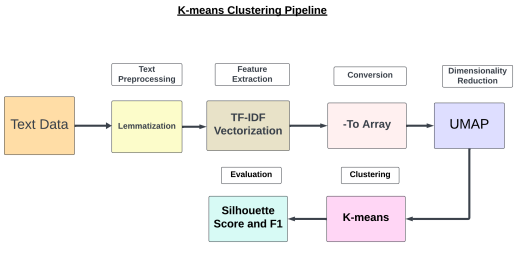

Figure 1: K-means Clustering Pipeline

Lemmatization is a text preparation step that reduces words to their basic forms, standardizing the input, before the raw text data from automotive reviews is sent into the K-means clustering pipeline. After that, the preprocessed text is vectorized using

TF-IDF, which turns it into numerical features by calculating term frequency and inverse document frequency to measure term importance. UMAP is utilized to reduce these feature vectors to two dimensions and convert them into an appropriate array structure for clustering(McInnes et al., 2018). The low-dimensional data is subjected to K-means clustering, which is optimized by counting the number of clusters using the Silhouette score, which is effectively constructed using the Scikit-learn library(Pedregosa et al., 2011).Since K-means clusters based on feature similarity without taking real sentiment labels into account, the F1 score for the K-means model was unsatisfactory demonstrating the limitations of unsupervised learning in tasks that need established sentiment labels.

XGBoost

For binary classification tasks such as sentiment analysis, the objective should be set to 'binary:logistic' according to the XGBoost hyperparameters. 'Logloss' is the evaluation metric that is employed; it is a measure of the accuracy of probabilistic predictions and increases when predicted probabilities diverge from the actual labels. In order to avoid overfitting, regularization is conducted using a lambda parameter set to 0.2, encouraging simpler models that more closely approximate unseen data.

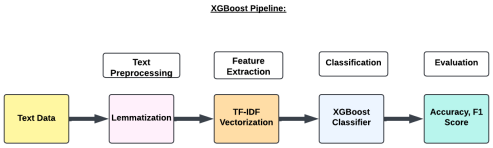

Figure 2: XGBoost Pipeline

The XGBoost pipeline receives its raw text data from automotive reviews. Lemmatization is a text preparation technique that reduces words to their most basic forms in order to standardize and sanitize the input. After that, this text is vectorized using TF-IDF to measure word importance using numerical features. The XGBoost classifier generates an ensemble of decision trees to maximize performance using these TF-IDF vectors as inputs. Measures such as accuracy and F1 score are used to assess the classifier. While recall and precision are combined to provide the F1 score, accuracy indicates the proportion of properly identified cases. This pipeline is appropriate for sentiment classification problems since it makes use of XGBoost's advantages in managing huge datasets and unbalanced classes.

Model Outcomes:

K-means clustering The following table summarizes the performance metrics for the K-means clustering model over the training and validation datasets. The Silhouette scores represent that there are well-formed clusters, though the model's performance in terms of precision, recall, and F1 score is low. This test shows, in practice, the limitation of using an unsupervised learning algorithm for sentiment classification tasks that require precise labeling.

Metric	Training Set	Validation Set
Precision	0.9754	0.9677
Recall	0.0915	0.0803
Specificity	0.9893	0.9875
F1 Score	0.1677	0.1481

Table 2: Performance Metrics for the K-means Model.

The high specificity values indicate that the model fits perfectly in identifying negative reviews, while low precision and recall, especially the F1 scores, show how K-means clustering is much less effective in accurately classifying positive sentiments. This is mainly because it is an unsupervised technique in nature, where predefined labels are not employed during training.

XGBoost

The XGBoost classifier's performance metrics on the training and validation datasets are presented in the table below. On the training set, the model shows good recall, accuracy, Specificity, and F1 scores, showing its effectiveness in correctly classifying positive and negative sentiments. Even though the validation set performance was somewhat slightly lower, it still demonstrated excellent accuracy , recall, and F1 showing the model's robustness and capacity for generalization. In this

Metric	Training Set	Validation Set
Precision	0.9597	0.8955
Recall	0.9990	0.9635
Specificity	0.8049	0.4750
F1 Score	0.9789	0.9284

Table 3: Performance Metrics for the XGBoost Model.

case, the XGBoost classifier is very successful and reliable for sentiment classification, as seen by the

high recall and accuracy values, particularly the F1 scores.

3 Task 3: Analysis and Discussion

This section analyzes and compares the performance of two models K-means clustering and XGBoost classifier for determining the sentiment of automotive reviews. The evaluation takes into consideration a number of factors, including clustering effectiveness, classification precision, and the ability to control class imbalance.

3.1 Justification of Model's performance

The dataset in this unsupervised learning process was therefore divided into two clusters: positive and negative sentiments using the K-means clustering algorithm. Performance of the developed K-means clustering was measured with the Silhouette score, which evaluates how similar a data point is to its own cluster compared to other clusters. The computed Silhouette scores were 0.8627 for the training dataset and 0.8719 for the validation dataset, meaning that good clusters were formed by this approach. However, K-means clustering does not use predefined labels to train and therefore falls into the task of discriminating between positive and negative reviews, as supervised learning methods do. We calculated the F1 score for the model based on K-means clustering, but the result was not very satisfactory since we obtained an F1 score value of 0.1677 on the training dataset and 0.1481 on the validation dataset. The fact that K-means clustering did not have a particularly good classification ability with the reviews, which was evident from the low F1 scores, points to the limitation of applying the unsupervised technique to such tasks, which are usually supervisory classification methods.

XGBoost, being a supervised learning algorithm, performed quite well for sentiment classification. The accuracy of the model trained from preprocessed text data that feeds to the XGBoost classifier is 96%, with an F1 score of 0.979 for the training set, while for the validation one, the score was 88% and gave an F1 score of 0.928. This underlines the capacity of XGBoost to provide correct identification of sentiments. In comparison to K-means clustering, the supervised nature of XGBoost is much more appropriate for precise sentiment analysis, considering also the fact that labeled data allows learning and generalization.

Comparison: In summary, though, K-means clustering does give some insights into the structure of the data and forms clusters with satisfying performance reflected in the Silhouette score, but it totally fails in sentiment classification due to its unsupervised nature. The F1 scores of K-means reveal some inadequacy in good classification of sentiments. Clearly, this limits the application of an unsupervised technique to tasks that demand supervised classification. On one hand, XGBoost as a supervised learning algorithm highly performs sentiment classification with superior accuracy and F1 scores. XGBoost is the preferable model for this job due to the following: it directly predicts sentiment with high accuracy and effectively handles class imbalances. Therefore, XGBoost is definitely the better model since it does handle the class imbalances well and it can give very accurate sentiment predictions.

3.2 Examples

This section shows the model's accuracy and performance through these below examples, where positive sentiments are labeled as 1 and negative sentiments as 0.

Text No.	GT	K-means	XGBoost
1	0	1	1
2	1	0	1
3	1	1	1
4	1	1	1
5	0	0	0

Table 4: Comparing K-means and XGBoost Models with Diverse Examples.

The following examples demonstrate some of the sensitive capabilities and limits. They show a comparison between XGBoost and K-means clustering for review sentiment classification: A common challenge in sentiment analysis is that some negative short phrases can be misclassified due to brevity and lack of explicit negativity. For instance, in the sentence "Not good,"(1) the sentiments expressed were misclassified by both K-means and XGBoost, they shows it as positive, while the ground truth is negative.This error shows how well both models can identify straightforward but negatively contextualized sentences.

On the other hand, the statement "Hubby is happy with this, has it filled already. Second Pit Posse cabinet purchased and very pleased."(2), was actually positive in sentiment, which XGBoost cor-

rectly classified, but K-means classified as negative. This case shows how XGBoost is better in handling positive sentiment. This is because XGBoost can make good use of the labeled data; hence, it works much better in sentiment understanding, especially where the sentiment is directly positive. The difference in performance brings out the superiority of models like XGBoost in the accurate capturing of context and sentiment expressed in the text.

Additionally,the models have marked "Worked and works great"(3) and "It works well, good product"(4) as positive feedback, managing easily positive statements. However, in the more nuanced and longer text, "You live and you learn... I purchased this product because I would need it but I didn't need it immediately so it sat in the tool drawer until I did. That was my mistake. When I finally did go to use it, it leaked all over... Unfortunately neither of them will hold on to the tire valve... Sadly, they have gone into the trash can because that's the most appropriate place for them."(5) both models correctly identified it as bad. The examples above show that K-means might fail in sentiment classification because no labelling is done for the data, while XGBoost does great in sentiment analysis by using data labels and being aware of the contextual information this makes it very well-suited to high-fidelity sentiment analysis.

4 Summary

This project involves the design, implementation, and evaluation of sentiment classification models on automotive reviews. We compared the performance of K-means clustering and XGBoost to bring out the strengths and weaknesses in unsupervised and supervised learning approaches. Emphasis is given in the project to data preprocessing, proper model selection, and handling imbalanced datasets to attain sentiment classification with high accuracy. In general, XGBoost outperformed other models because of the ability to use labeled data to capture context and nuance in the wording.

4.1 Lessons Learned

Key lessons of this project include the development, implementation, and evaluation of sentiment classification models. The point is to select the right model for what needs to be achieved. The supervised learning models, based on the comparison, show that they significantly outperform the unsupervised model K-means clustering in the sentiment classification task. It is apparent that supervised models would capture the labeled data and nuanced contextual information more effectively. The preprocessing techniques of data, such as lemmatization and TF-IDF vectorization, were really important for turning raw textual data into meaningful features that increased the performance of the models. Therefore, it enforces clean, well-structured data requirements in developing machine learning applications. In retrospection, i would feel that something could be done better in certain areas. For instance, the inclusion of an experiment with other supervised models like SVM or deep learning approaches, e.g., LSTM networks, should add more insight and possibly lead to better performances. Further optimization of the models can be done with a much deeper hyperparameter tuning process. Cross-validation techniques may also be applied. These reflections only serve to confirm the importance of model selection, extensive data preprocessing, and continuous experimentations in the face of achieving results in sentiment analysis.

References

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

William Scott. 2019. Tf-idf from scratch in python on real world dataset. *Towards Data Science*, 15.