CSE 4/587 Project Phase 2

Fashion Recommender System

**Machine Learning Algorithms and Statistical Models:**

Our main goal in this project is to use analysis to build a recommendation system that will forecast what consumers would likely purchase in the upcoming set time, taking into account seasonal events and other factors. Our goal is to develop a model that suggests a mix of products based on what users often receive when they buy one item from the website.

To achieve this we have decided to apply KNN with cosine similarity, Logistic Regression, Catboost, Kmeans, and Neural Networks.

KNN enhances H&M's customer engagement by personalizing experiences and segmenting users for targeted outreach, while also optimizing inventory through predictive analytics on transaction data. It involves creating a feature set from merged customer and transaction datasets, scaling numerical data, encoding categorical variables, and determining the best number of neighbors (k) during training. Performance is assessed by accuracy and other classification metrics. Separately, cosine similarity measures textual similarity between product descriptions, aiding in product recommendation systems.

K Means:

In machine learning, the K-means clustering technique groups data points into 'k' clusters according to how similar they are. Until stable clusters form or a predetermined number of iterations is reached, it updates these centers by iteratively allocating points to the closest cluster center. We have used K means in this project for customer segmentation. Our aim was to cluster the customers on the basis of their age and then put them in groups.

In this project, we have segmented our customer base using K techniques. Our goal was to group the clients after classifying them according to age. To make the computation easier, we have simply selected one lakh transactions from the transactions csv for this. Only the article\_id, prod\_name, product\_type\_name, product\_group\_name, department\_name, index\_name, index\_group\_name, section\_name, and garment\_group\_name were taken from our four datasets in order to determine which age group of customers is purchasing what kinds of products, and then we clustered the similar types of customers in one group.

Logistic Regression:

A statistical technique called logistic regression is used to forecast a binary outcome's probability based on one or more predictor factors. It is frequently used for classification problems in machine learning and statistics when the objective is to place input data into one of two possible groups.

The main reason for applying the logistic regression for this problem statement is that we would like to take customer data into consideration and try to predict whether the customer is a potential member or not. So, here we know if the customer could be given to be a potential member and offer deals on the platform using this binary classifier. So, our main aim is to increase the potential loyal customer base who are interested in shopping and increasing the revenue by offering them potential deals. We took features of customers like age and fashion news frequency and we would be predicting their club member status. So, when a new customer joins taking this data into consideration we can offer loyalty membership deals to attract them to the premium club of shopping.

DBScan:

Density-Based Spatial Clustering of Applications with Noise is referred to as DBSCAN. This well-liked clustering technique is applied in data mining and machine learning. Unlike many other clustering algorithms, DBSCAN does not require the user to specify the number of clusters in advance.

DBSCAN's capacity to manage diverse, irregular clusters, deal with outliers, and adjust to different cluster sizes makes it useful for customer segmentation in recommendation systems. Its adaptability in detecting noise guarantees a strong segmentation. But fine-tuning the parameters is essential. Since DBSCAN excels at identifying complex, ill-defined customer patterns without predefining cluster numbers, combining it with other techniques can improve understanding of customer behavior and recommendation accuracy.

Multilayer Perceptron:

An artificial neural network with multiple layers of connected nodes, or neurons, is called a Multilayer Perceptron (MLP). It's a strong and adaptable class of neural networks that are used for many different machine learning applications, particularly in supervised learning for problems involving both regression and classification.

The main intention for using a multilayer perceptron is that it could handle multiple features in building a model to classify the channel. There are two sales channels one offline and online. We build a neural network to predict the sales channel if we give the details of the user and to understand which channel the user is more interested in buying the articles. So, if the user is predicted to be online interested the e-commerce can target a user online with brand deals and offers to improve purchases using reward points. This helps to increase the revenue in certain clothing segments which could be sold through particular targeted sales channels.

Random Forest:

A potent ensemble learning technique for both regression and classification applications is called Random Forest. It is built from several decision trees and produces a class that is the mean prediction of the individual trees in regression or the mode of the classes in classification.

Despite its robustness and versatility, Random Forest is not the top option for recommendation systems. Its scalability and ability to handle a variety of data types might be useful in some situations, but other approaches, such as collaborative filtering, are better suited due to its incapacity to handle complex tuning and implicit feedback.

Naive Bayes:

The 'naive' assumption of feature independence underlies the Naive Bayes family of probabilistic classifiers, which are based on the Bayes theorem and are straightforward yet effective. It is extensively utilized in natural language processing and machine learning for classification tasks.

Naive Bayes is not an ideal choice for a recommendation system but we have used it to compare with the results of our other classifiers.

Cosine Similarity:

The cosine similarity matrix indicates the pairwise similarity between the first five articles based on their textual description. Here's how to interpret and use this information in the context of articles for the H&M shop:

### Interpretation of Cosine Similarity Matrix:

* A cosine similarity score of 1 means that the two article descriptions are identical (or very similar if not exact due to floating-point precision).
* A score of 0 would mean that the descriptions share no words in common (after accounting for stop words, term frequency, etc.).
* Scores between 0 and 1 indicate the degree of similarity, with higher scores representing more similarity. In your case, the scores of approximately 0.3474 suggest a moderate similarity between the articles.

### Usefulness for Articles:

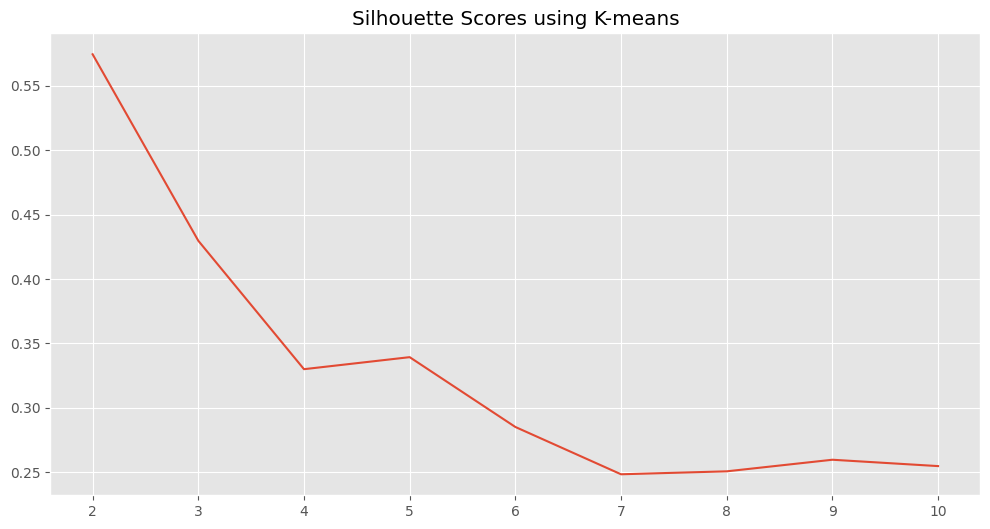
* Recommendation Systems: If a customer likes a particular item, you can recommend items with similar descriptions.
* Inventory Management: Group similar articles together to help with stocking and distribution.
* Customer Insights: Understand which articles are similar and may appeal to similar customer segments

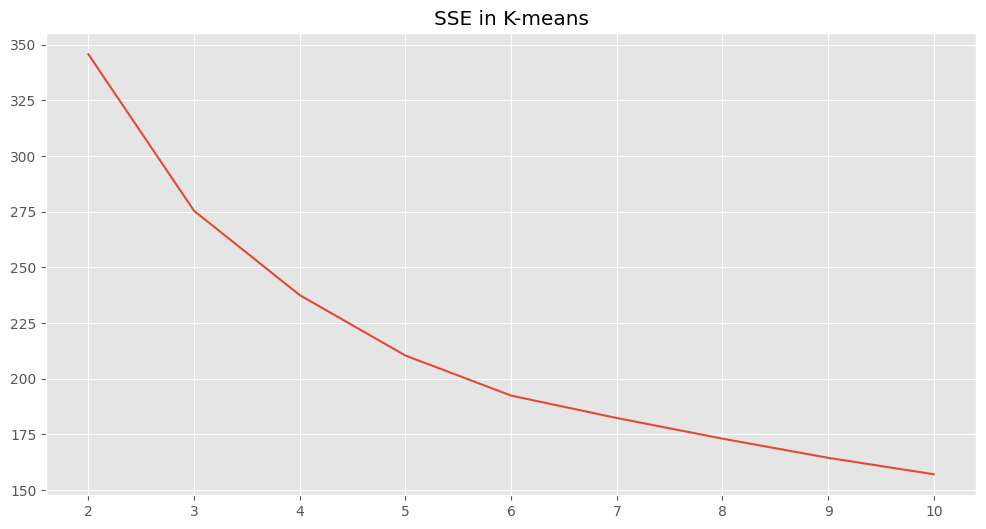
**Implementation and Hyperparameter Tuning:**

K Means: Although it is not as involved as with other machine learning algorithms, K-means hyperparameter tuning mostly involves modifying the value of 'k', or the number of clusters, and on occasion the initialization strategy or convergence criteria.

The hyperparameter that matters the most in K-means is the number of clusters (k). It's common practice to use techniques like the Elbow Method, Silhouette Score, or Gap Statistics to determine the ideal number of clusters. These techniques aid in achieving the best possible trade-off between clustering performance and model complexity.

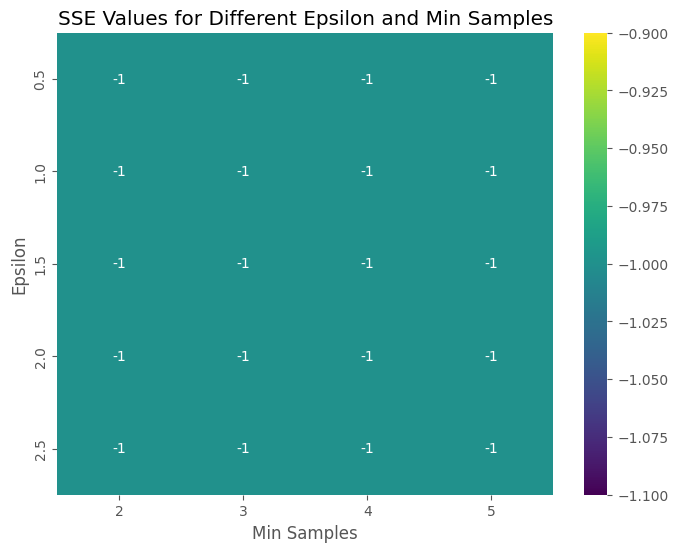
We used Silhouette Score and SSE to figure out the efficient number of clusters. After running the algorithm for k = 2 to k = 10, we were able to determine from the graph that selecting 4 clusters is a good number.

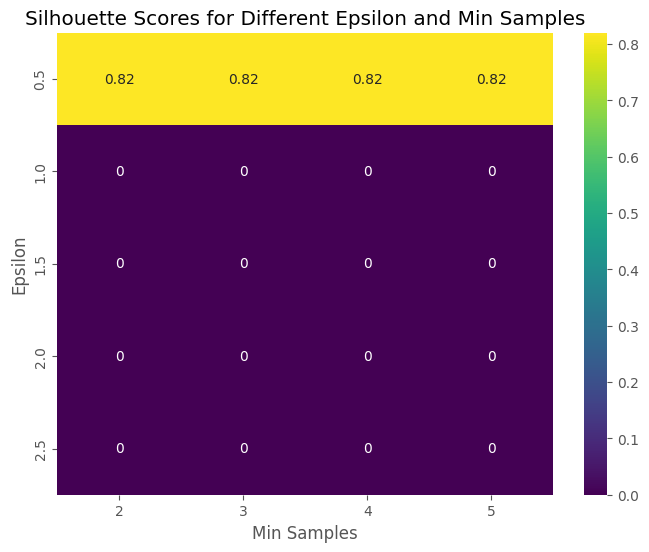




DBSCAN:

In DBSCAN the hyperparameter tuning is done by changing the epsilon and the min samples value. The search radius for nearby points is determined by Eps (Epsilon); small eps may identify many as noise, while large eps may combine clusters. Raising min\_samples may result in larger clusters and less noise, but setting it too high could cause points to be incorrectly classified as outliers. Min\_samples defines the minimum points required in the eps range to form a core. Again from the figures we were able to make out the efficient eps = 0.5 and min samples = 5.





Logistic Regression:

There is not much parameter tuning involved in this classification algorithm but the performance and scalability of the model depend on how well we selected the features and target variables. We selected two features from the customer data frame as we found that by intuition on EDA, they are influencing the target variable. So, before training the data with the classifier we cleaned the values in those columns and converted them into class labels [0,1,2] after checking the unique values. After further processing, we train the model using test data of 20 percent of the total dataset. Once the model is trained using the scikit learn library we get predicted values on which we apply accuracy, precision, and recall for a better understanding of how the models performed.

Random Forest: Again not there is not much parameter tuning involved in this classification algorithm. But we used a method called grid search which entails creating a hyperparameter grid and looking through every possible combination.

Naive Bayes: The hyperparameter tuning of Naive Bayes involve the same method. In this instance, the GaussianNB model's grid search is conducted over a range of values. As necessary, make changes to the parameter grid and other settings. The model is retrained and assessed using the test set once the optimal parameter has been determined.

Multi-Layer Perceptron:

Before training the model we merge the customer data and transaction data to get a merged dataframe. In that, we select a few features which we can use for training the MLP model. Our features include FN, Active, club member status, fashion news frequency, age, and price. We are classifying which sales channel the customer would be interested in buying clothes. We used the following parameters with one hot encoding applied to some to make them compatible for training. The hidden layers are in 64 and 32 configurations with RELU activation functions Based on the predicted values we calculate accuracy, precision, recall, and the confusion matrix.

Cosine Similarity

The cosine similarity matrix indicates the pairwise similarity between the first five articles based on their textual description. Here's how to interpret and use this information in the context of articles for the H&M shop:

### Cosine Similarity:

### Interpretation of Cosine Similarity Matrix:

* A cosine similarity score of 1 means that the two article descriptions are identical (or very similar if not exact due to floating-point precision).
* A score of 0 would mean that the descriptions share no words in common (after accounting for stop words, term frequency, etc.).
* Scores between 0 and 1 indicate the degree of similarity, with higher scores representing more similarity. In your case, the scores of approximately 0.3474 suggest a moderate similarity between the articles.

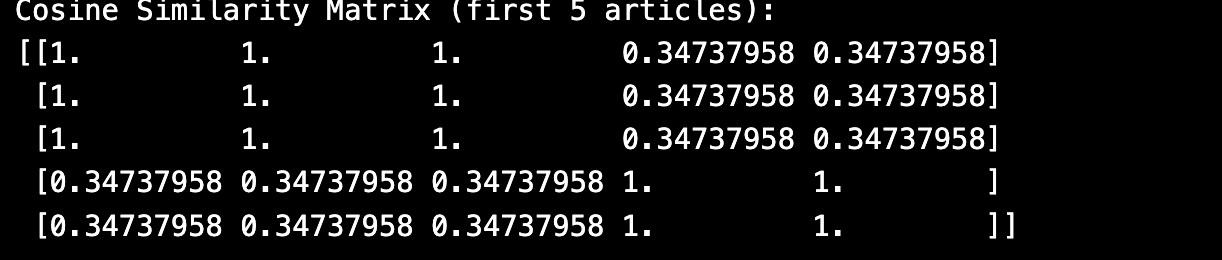
### Training the Model:

* Technically, the cosine similarity calculation is not a "training" process but an "inference" or "evaluation" process. we are not training a model but rather calculating the similarity based on TF-IDF vectors.
* we first create the TF-IDF matrix, which transforms the text data into a numerical form that captures the importance of terms (words) in the context of the document corpus (article descriptions).

### Accuracy and Evaluation:

* Cosine similarity is not about "accuracy" in the same sense as a predictive model. Instead, it's a measure of how text documents are related to each other based on their content.
* To evaluate if the cosine similarity scores are useful, we could:
  + Perform user testing to see if items deemed similar by the metric align with human judgment.
  + Use A/B testing to see if recommendations based on cosine similarity perform better compared to a different method.
  + Measure the impact on sales or customer engagement when using cosine similarity for recommendations.

We determine a cosine similarity threshold for article grouping, validate recommendations with user feedback, regularly update the TF-IDF matrix, and integrate it with behavioral data for dynamic, personalized recommendation systems.



Logistic Regression:

As per the result we got good accuracy in prediction. This model seems to be effectively performance for classification with 93 percent accuracy which is good performance. We also saw good precision and recall values.

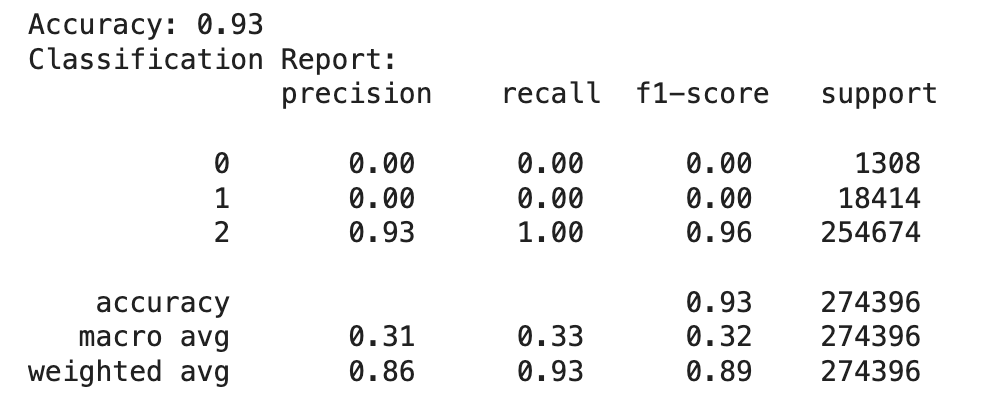
Confusion Matrix:

[[ 0 0 1308]

[ 0 0 18414]

[ 0 0 254674]]

Accuracy: 93%



Multi-Layer Perceptron:

Confusion matrix:

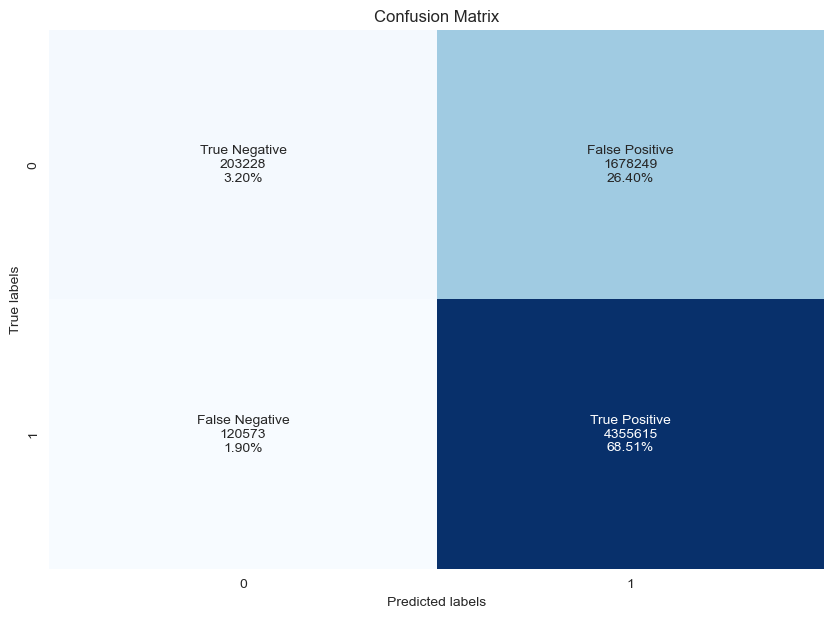
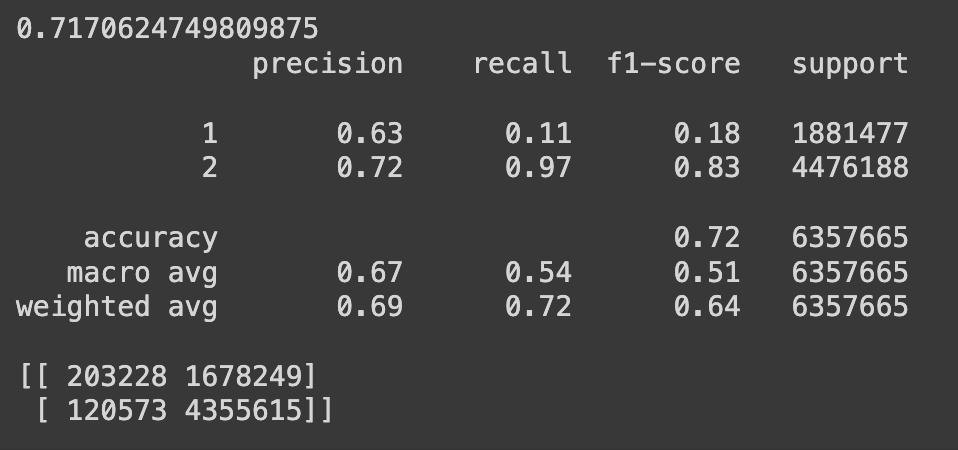
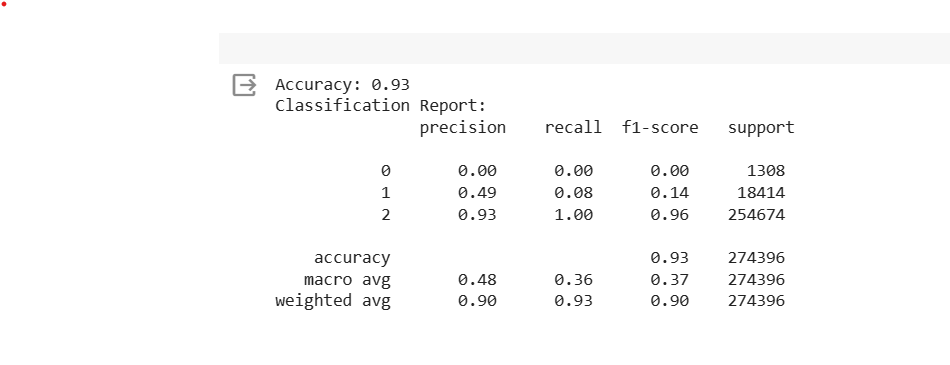


Figure.

Accuracy: 71.70 %

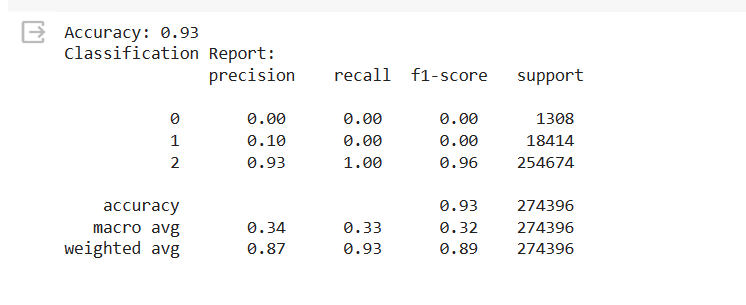


Random Forest:



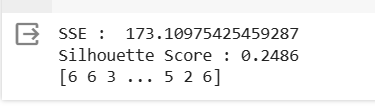
Naive Bayes:

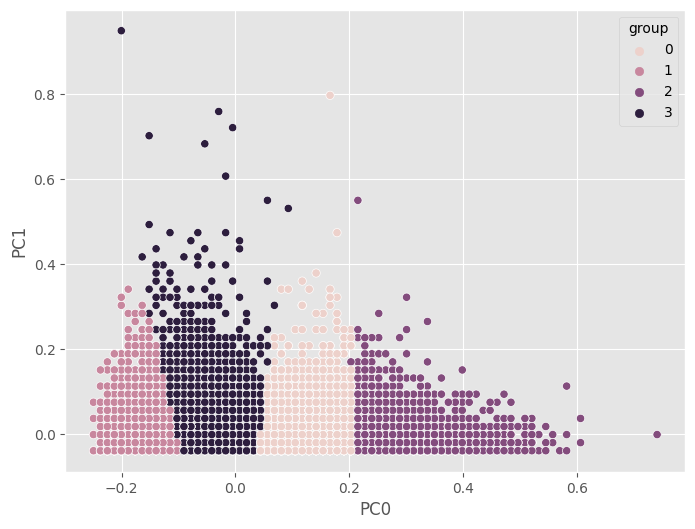
Random Forest, Naive Bayes and Logistic have the same accuracy, precision and recall.



K Means:

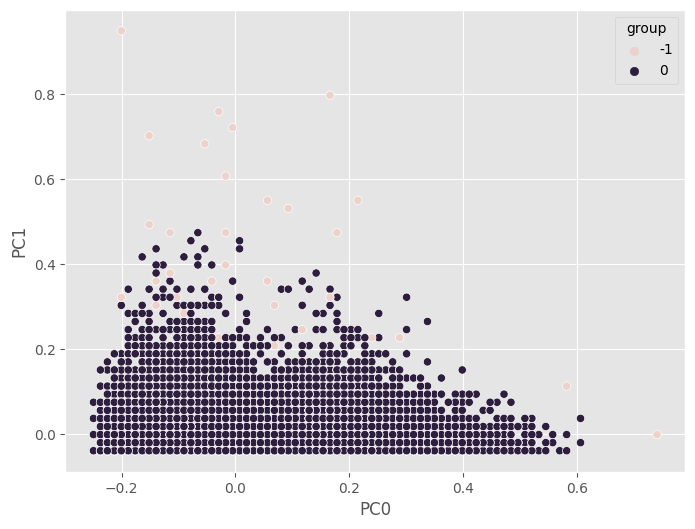
The sum of squared errors (SSE) measures the sum of the squared distances between each data point and the centroid of the cluster it belongs to. Compact clustering and moderate separation are indicated by a silhouette score of 0.24. The fact that it is positive means that the objects within the clusters are relatively close to one another when compared to objects in other clusters, and that the clusters are spaced reasonably widely apart from one another. From the graph we can see that the clusters are clearly divided without any overlapping.

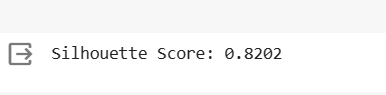




DBSCAN:

A silhouette score of 0.84 is considered quite high and generally suggests very well-defined and appropriately separated clusters.





Summary

The model that seems to be the best in terms of accuracy is one with a reported performance of 93% accuracy. The confusion matrix for this model indicates that all predictions fell into a single class, which suggests a highly imbalanced dataset or that the model may be biased towards the majority class​​​​. The silhouette score associated with the clustering aspect of the analysis, which was used to evaluate the separation between clusters, is 0.24. This positive silhouette score suggests that, to a moderate extent, the objects within clusters are closer to each other than to objects in other clusters, indicating a reasonable level of cluster separation and cohesion​​.

However, it's important to note that while the accuracy is high, the confusion matrix suggests that the model may not be performing well in terms of precision and recall across different classes, given that all predictions are for a single class. This is a classic indication that the model might be exhibiting a high true negative rate but failing to correctly identify positive cases, which could be problematic for a recommendation system that requires a balance between various classes for effective performance.

For a more comprehensive evaluation, other metrics such as precision, recall, and the F1-score are necessary. Cosine similarity is mentioned but not used as a performance metric; instead, it's used to assess the similarity between article descriptions, which is a separate task from the classification goals associated with the KNN model.

References:

<https://www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations/data>

<https://youtu.be/l0KNPQn5EmE>

<https://www.youtube.com/watch?v=l0KNPQn5EmE&ab_channel=ChaiTimeDataScience>

<https://www2.hm.com/en_us/index.html>