AlbaneCM /
tripadvisor_sentiment_predictions

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

0 stars 0 forks 1 watching 1 Branch 0 Tags Activity

Public repository

main 1 Branch 0 Tags Go to file t Go to file + Add file Code ···

AlbaneCM final models now

| | | |
|---|---|---|
| data | pushed data | last week |
| images | final models | now |
| .gitignore | added gitignore | 20 hours ago |
| README.md | final models | 3 minutes ago |
| environment.yml | added gitignore | 20 hours ago |
| tripadvisor_reviews_detractors.ipy... | final models | now |

README

# Predicting TripAdvisor Review Sentiment

## Table of Content

1. Overview
2. Business Understanding
3. Data Understanding
4. Data Preparation
5. Modeling
6. Evaluation
7. Findings & Recommendations
8. Limits & Next Steps

## 1. Overview

This notebook examines reviews about hotels from guests who stayed there and predicts whether the sentiment of reviews is positive or negative. The organization of this notebook follows the CRoss Industry Standard Process for Data Mining (CRISP-DM) is a process model that serves as the base for a data science process.

## 2. Business Understanding

The increasing significance of online reputation and social media in the hospitality industry has become integral to a hotel's operational and financial success. The report from Cornell's Center for Hospitality Research highlights the rising trend of consumers relying on online reviews during hotel bookings. The study reveals the link between a hotel's online reputation and its pricing, occupancy rates and revenue performance.

Recognizing the challenges faced by hotels in managing online reputation, a **new branch of TripAdvisor** is now dedicated to supporting and advising hotels on how to **improve their online reputation**. The branch launched a new B2B product identifying **unsatisfied guests in real time** with the goal to improve their experience before they leave. Our first client is a worldwide hotel chain: Hyatt. The intended users are corporate directors of guest experience.

Through real-time guest reviews and a sophisticated algorithm, the platform alerts general managers about guests requiring immediate attention. The goal is to enable proactive measures to enhance satisfaction. This innovative tool, backed by an analysis of over 20,000 real guest reviews, also offers tailored recommendations for improvement upon implementation, aligning with the goal of turning detractors into promoters.

**The Task**: Deploying Hyatt's strategy to increase their online reputation

**Objectives**

1. Short-term: improve guest experience
    - thanks to our customized recommendations targeting areas that historically caused guest complaints

2. Long-term: reduce the share of negative reviews
    - by identifying unsatisfied guests through real-time reviews, aiming at improving their experience before they leave

**Target Audience:** Hyatt's corporate director of guest experience

## 3. Data Understanding

- **Data Source**

The data is hosted on Kaggle and is provided by *LARXEL*.

In order to access the file, access the data source through the link provided and click on the `download` button at the top right corner. This will download a zipped folder. The below code unzips it upon loading the data.

The dataset contains about twenty thousand reviews collected from a hotel's TripAdvisor page. Each review is assigned a rating from 1 to 5, 1 being the lowest (negative review), 5 being the highest (positive review).

- **Features**

Prior to preprocessing, the columns are:

- `Review` : the actual review's record
- `Rating` : a number from 1 to 5, given by the guest at the time of the review

- **Target**

The target will be identifying negative reviews. It will be created based on the `Rating` column, and will be named `Sentiment`. Based on a given set of reviews, I will predict if the review's sentiment was negative. To be more precise, it will be identified as `detractors`. This term is defined by the metric measuring online reputation: Net Promoter Score.

More on NPS below.

## 4. Data Preparation

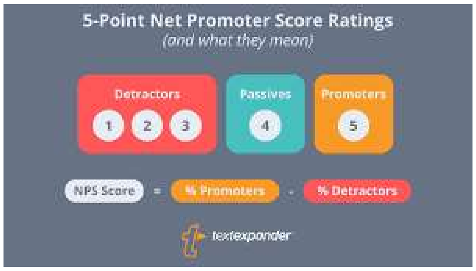
### 4. 1- Data Cleaning

In the first part of data preparation, the typical data cleaning tasks are addressed before splitting the set between train and test data. The steps include:

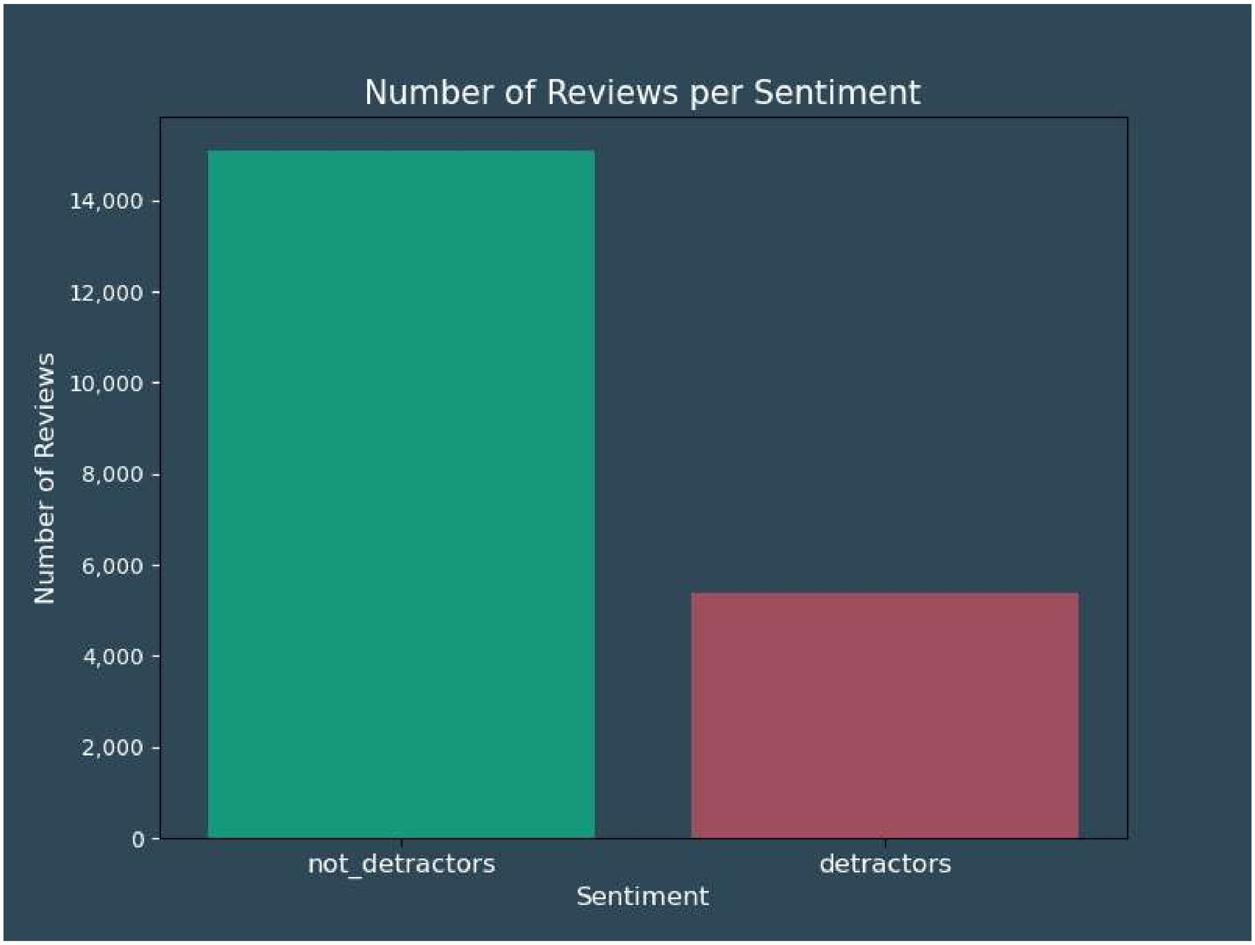
- Missing data
  Missing values were inspected but none were found, no modification was necessary.
- Duplicates
  Duplicates were researched but none were found: no modification was necessary.
- Binary classification
  Ratings range from 1 to 5, 1 being the lowest and 5 the highest.
  A new column `Sentiment` was created to group customers' reviews as follows:
  - Reviews' rating ranging from 1 to 3 included: `detractors`
  - All other reviews: `not_detractors`

The reason for such a classification follows the definition of NPS score.

**NPS Score**



The Net Promoter Score ([NPS](#)) is a metric used to measure customer satisfaction and loyalty with a hotel, (or more generally a product, service, or company). It is based on the question: "How likely is it that you would recommend our *hotel*? The responses can be measured on a scale from 0 to 10 or in our case: 1 to 5.



- Train-Test split
  The dataset is being divided into two separate subsets: a training set, and a testing (or validation) set. The validation set will allow to assess the performance of the model. The dataset is split before any further transformation is done to prevent data leakage.

Two parameters were assigned:

- `random_state=42` for reproducibility
- `stratify=y` to address class imbalance issues

- Encoding target In order to make predictions useable in calculations, I encoded the target:
  - 0 replaced `not_detractors`
  - 1 replaced `detractors`

## 4. 2- Data Preprocessing & Exploratory Analysis

In order to preprocess the reviews, the following transformations were performed:

- **Standardizing case**
  This step was verified as it is important to ensure text is uniform and consistent. Nevertheless, no extra step was taken as text was already saved as lower case.

- **Character Encoding**
  This was done to ensure consistent representation of text by transforming non-text characters into a normalized format, which will facilitate proper text processing, analysis, and model training.

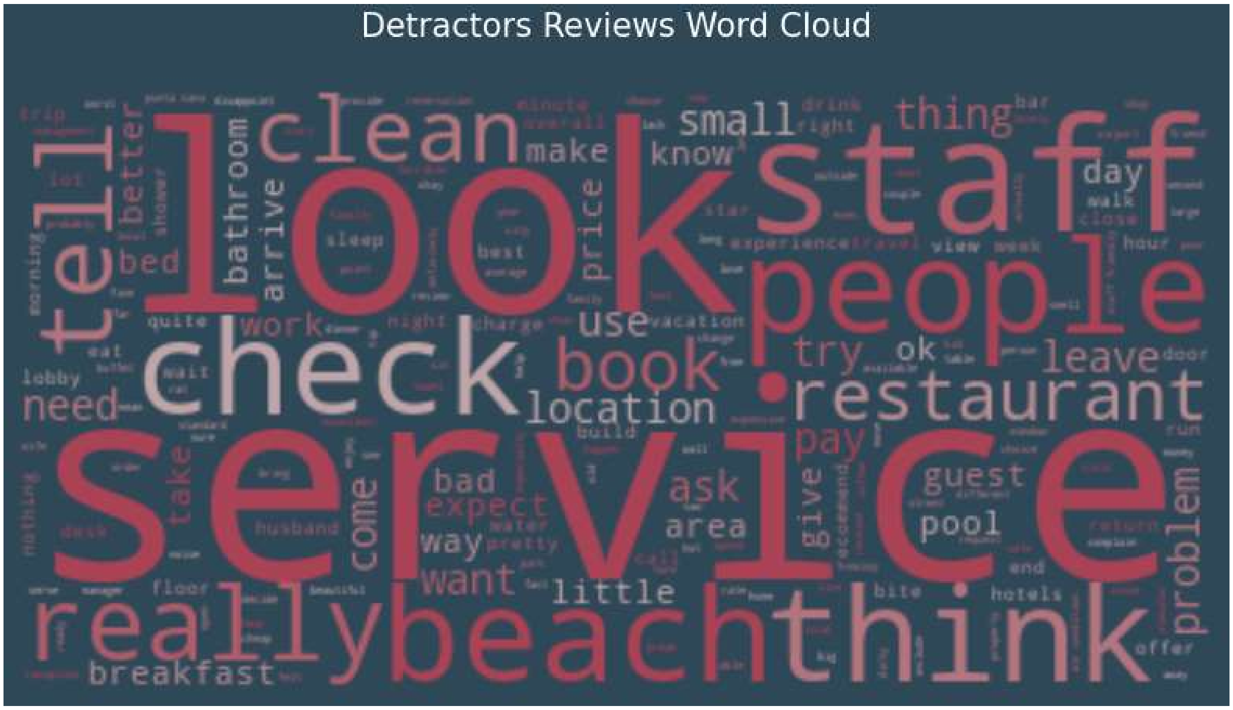
- **Tokenizing**
  Tokens of words were created. This was done with the `RegexpTokenizer` package from nltk.tokenize. This step will facilitate the conversion of word into a suitable form for analysis and modeling.

- **Stopwords**
  To focus on the data's theme, English stopwords were removed. Manual additions were made in this text's context, such as adding industry specific word (i.e. "hotel", "stay")

- **Lemmatize**
  The `WordNetLemmatizer` package from nltk.stem.wordnet was used to reduce words to their base form, allowing a more accurate analysis

- **Frequency Distribution**
  The `FreqDist` package was used to review in a dictionary-like output, the words and their frequencies

- **WordCloud**
  The words' frequencies were represented visually thanks to the `WordCloud` package



- **Bigrams**
  Bigrams were drawn to have a better understanding of the themes thanks to the `collocations` package and its BigramAssocMeasures

- **Mutual Information Scores**
  Bigrams that occur more than 5 times were examined through `mutual information scores`

The preprocessing tasks were summarized as a function which was called both on the train and test data.

## 5. Modeling

I now have an initial idea for recommendations on focus areas to improve guest satistfaction.
My objective is now to:

- Develop a tool to identify unsatisfied customer reviews in real time

Incorrectly identifying a guest as satisfied when they weren't would lead in a detractor posting a review instead of catching them while they are in-house, and having a chance of improving their satisfaction.

As a consequence, `recall` is the main evaluation metric for this model.

As the dataset is a text, it requires a transformation before it can be used for modeling. Like other types of dataset would one-hot encoded, here, the reviews were vectorized, using the common method in natural language processing: `TfidfVectorizer`.

It converts a collection of text documents to a matrix of tf-idf features.

- Term-Frequency
  Measures how often a term (word) appears in a document
- Inverse Document Frequency (IDF)
  Measures the importance of a term in the entire collection of documents.

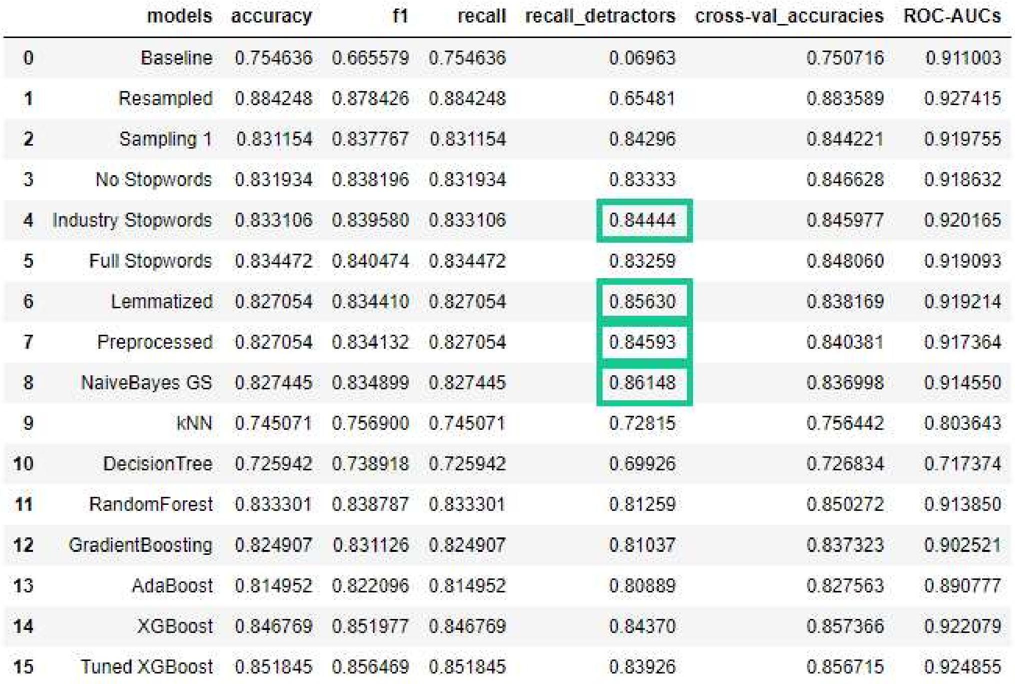
7 main classification models were explored:

1. Multinomial Naive Bayes
2. K-Nearest Neighbor
3. Decision Tree
4. Random Forest
5. Gradient Boosting
6. AdaBoost
7. XGBoost

Undersampling, stopwords, lemmatize, and hyperparameter tuning were parameters used to tune models. In addition, X_train was split with `stratify` in order to keep the distribution ratios for ech sentiment.

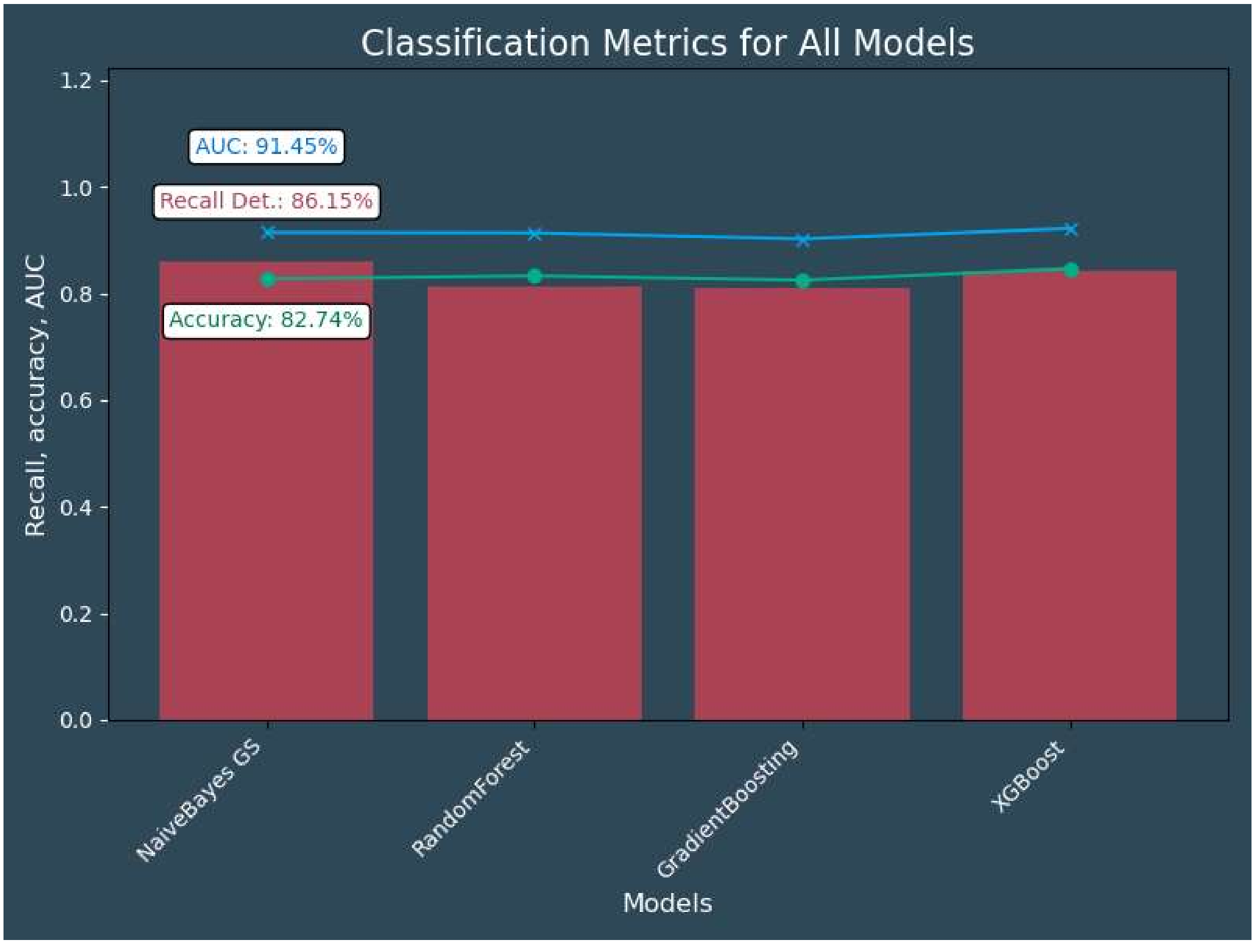
All models went through 4 steps: 1) Fitting and training on train data 2) Evaluation Metrics 3) Classification Report 4) Confusion Matrix 5) ROC-AUCs

The 7 models' results and their evaluation metrics are summarized below. The highest records are highlighted.



| | models | accuracy | f1 | recall | recall_detractors | cross-val_accuracies | ROC-AUCs |
|---|---|---|---|---|---|---|---|
| 0 | Baseline | 0.754636 | 0.665579 | 0.754636 | 0.06963 | 0.750716 | 0.911003 |
| 1 | Resampled | 0.884248 | 0.878426 | 0.884248 | 0.65481 | 0.883589 | 0.927415 |
| 2 | Sampling 1 | 0.831154 | 0.837767 | 0.831154 | 0.84296 | 0.844221 | 0.919755 |
| 3 | No Stopwords | 0.831934 | 0.838196 | 0.831934 | 0.83333 | 0.846628 | 0.918632 |
| 4 | Industry Stopwords | 0.833106 | 0.839580 | 0.833106 | 0.84444 | 0.845977 | 0.920165 |
| 5 | Full Stopwords | 0.834472 | 0.840474 | 0.834472 | 0.83259 | 0.848060 | 0.919093 |
| 6 | Lemmatized | 0.827054 | 0.834410 | 0.827054 | 0.85630 | 0.838169 | 0.919214 |
| 7 | Preprocessed | 0.827054 | 0.834132 | 0.827054 | 0.84593 | 0.840381 | 0.917364 |
| 8 | NaiveBayes GS | 0.827445 | 0.834899 | 0.827445 | 0.86148 | 0.836998 | 0.914550 |
| 9 | kNN | 0.745071 | 0.756900 | 0.745071 | 0.72815 | 0.756442 | 0.803643 |
| 10 | DecisionTree | 0.725942 | 0.738918 | 0.725942 | 0.69926 | 0.726834 | 0.717374 |
| 11 | RandomForest | 0.833301 | 0.838787 | 0.833301 | 0.81259 | 0.850272 | 0.913850 |
| 12 | GradientBoosting | 0.824907 | 0.831126 | 0.824907 | 0.81037 | 0.837323 | 0.902521 |
| 13 | AdaBoost | 0.814952 | 0.822096 | 0.814952 | 0.80889 | 0.827563 | 0.890777 |
| 14 | XGBoost | 0.846769 | 0.851977 | 0.846769 | 0.84370 | 0.857366 | 0.922079 |
| 15 | Tuned XGBoost | 0.851845 | 0.856469 | 0.851845 | 0.83926 | 0.856715 | 0.924855 |

The most relevant ones can be summarized through the below bar chart.



## 6. Evaluation

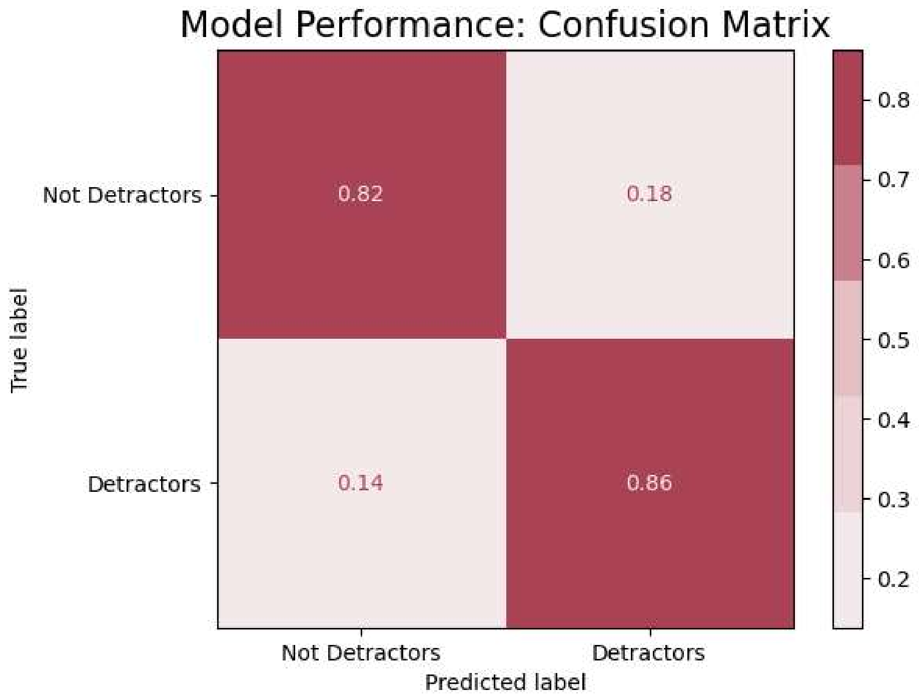
### 6. a) Final Model and Classification Metrics

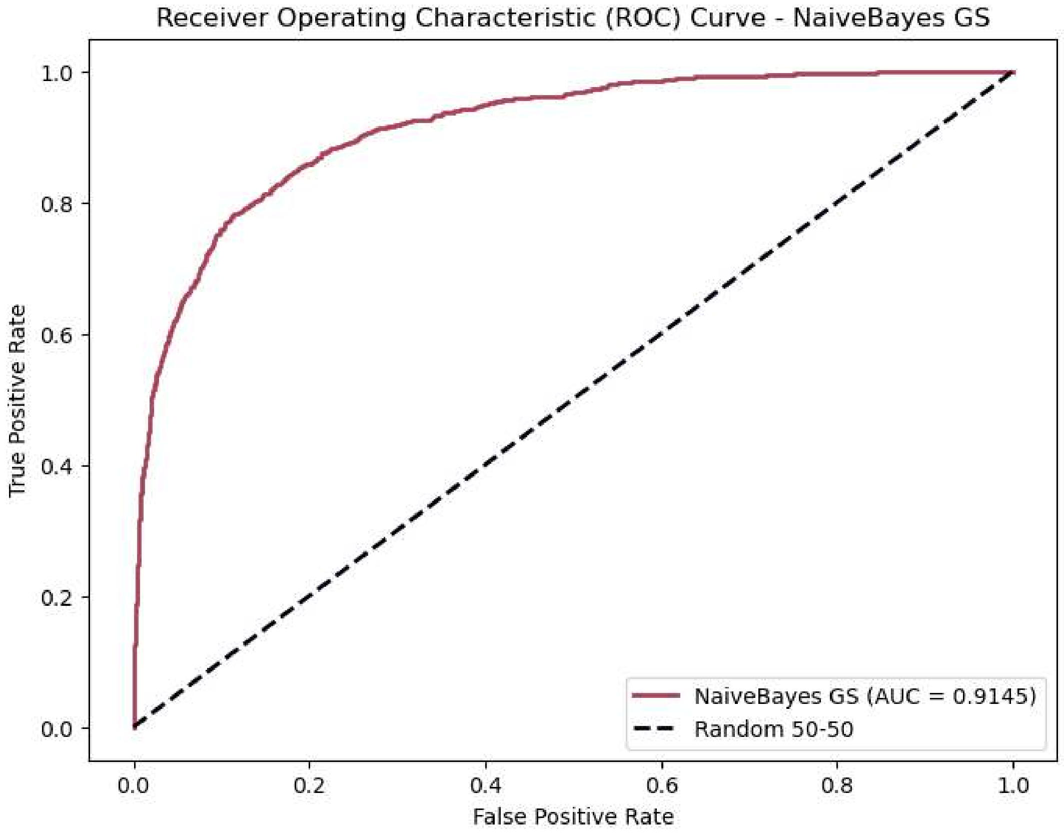
The model that predicts the most accurately the detractors reviews is the **Naive Bayes Tuned with Grid Search**.

Evaluation Metrics

- Evaluation Metrics on Train Data
    - Accuracy: 0.8274
    - Recall Detractors: 0.8615
    - Train data: Mean Cross-Validated Accuracy: 0.8370
- Test data: Mean Cross-Validated Accuracy: 0.8454

### 6. b) Model Performance





The classification report, confusion matrix and ROC-AUC summarize the evaluation of the model's performance on predicting detractors from hotel TripAdvisor reviews.
The model performs better on predicting *detractors* sentiment. More precisely, it has a true positive rate higher for this class than for the *not_detractors*.

This score was the focus for the evaluation of the models, as the cost of false negative is higher than the cost of false positive. If a review from a detractor was incorrectly identified as "not detractor", the hotel would miss an opportunity to transform a guest's negative experience.

Looking at the details by metric:

**Accuracy**
The mode correctly identified close to 83% of all reviews.

**Recall**
86.15% of actual detractors were correctly predicted.

**F1 Score**
The balance between precision and recall. It is lower (72.5%) for the detractors class, which indicates precision is lower for this class.

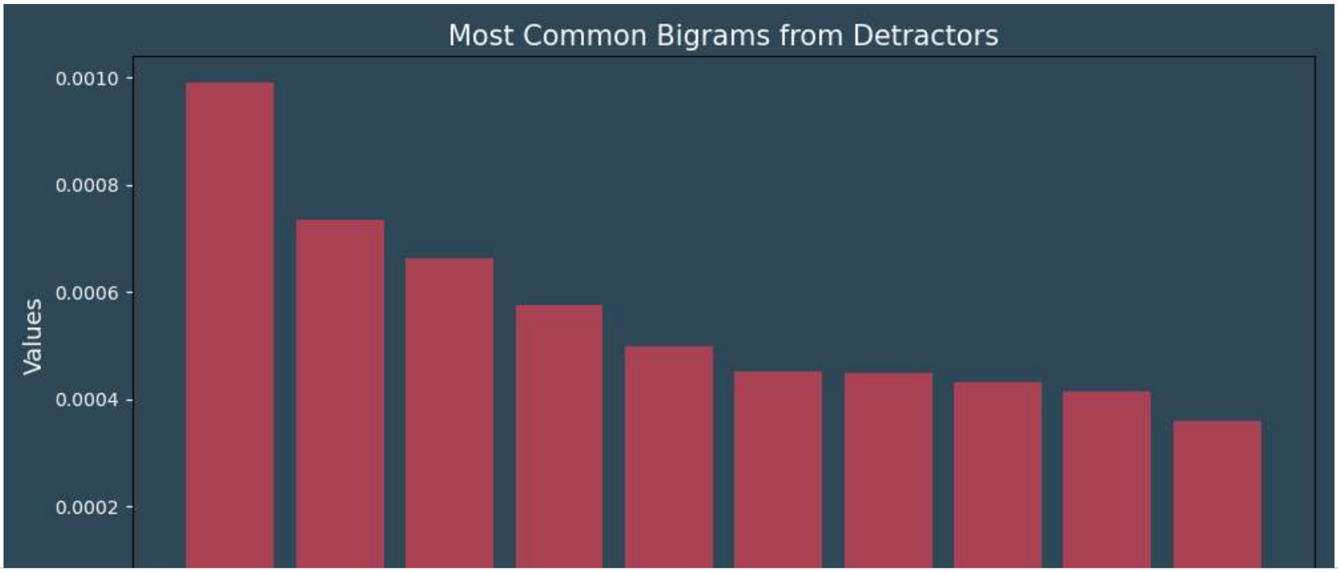
**AUC**
The Area Under the Receiver Operating Characteristic Curve (AUC) value of 0.9145 indicates a high effectiveness in distinguishing between the two classes: `not_detractors` and `detractors`. The ROC curve, which plots the true positive rate against the false positive rate, covers a significant area under the curve.

## 7. Findings & Recommendations

### 7. Recommendations

The advantage of our new service is the deployment can be immediate. Based on historical guest complaints, below are the areas we recommend to focus on to immediately start improving guest satisfactions.

1. Focus on resort hotels
   → These hotels have the most unsatisfied guests
2. Develop a maintenance program with engineering teams
   → Appearance & dysfunctionality cause frustrations
3. Train staff to enhance friendliness
   → Also ensure more languages are spoken
4. Respond to reviews
   → Detractors advise to read reviews



## Releases

No releases published
Create a new release

## Packages

No packages published
Publish your first package

## Languages

● **Jupyter Notebook** 100.0%