In this chapter, the implementation details of the classifiers applied and the reason behind choosing these classifiers to handle sentiment classification are discussed. First, the experimental details of text preprocessing are presented. In the next sections (5.1 and 5.2) experimental implementation of various classifiers used to handle balanced and imbalanced datasets are discussed. Section 5.3 describes various evaluation metrics used in the analysis.

The methods evaluated in this experiment are divided into two groups. The first group includes evaluating various classification methods for balanced datasets. The classification method includes the standard SVM, NB, LR, BPN, PNN, bagging and boosting. The second group refers to evaluating various classification methods for imbalanced datasets. Weka, rapidminer and knime are used as tools to implement the classifier in the experiments.

The preprocessing steps of the datasets used involve tokenization, removing stop words and stemming. The preprocessing is implemented using rapidminer’s text mining plugin. After preprocessing of reviews, the features of the product (called product features) that customers have expressed their sentiments are to be identified for each dataset used. For the benchmark datasets used (BD1 and IBD1), the product features are annotated for each review sentence in the dataset itself. A sample annotated sentence from the dataset (BD1 and IBD1) is as given below,
“Picture quality [+1] ##a few of my co-workers highly recommended the canon for picture quality.”

In the above review sentence, the following observations are made.
“Picture quality [+1]” denotes that picture quality is a product feature.
“+” denotes positive opinion, “1” is the opinion strength and “##”
denotes start of review sentence.

For the customized datasets used (BD2 and IBD2), a sample review
crawled from the Amazon.com for datasets BD2 and IBD2 is as given
below,

![Five Stars](https://example.com/five_stars.png)

**Five Stars.** July 21, 2014
By [Ricky D](https://example.com/ricky_d) (Naples, Florida United States) - [See all my reviews](https://example.com/all_reviews)
Verified Purchase (what's this?)
This review is from: NIKON D3100 DSLR Camera with 18-55mm f/3.5-5.6 AF-S Nikkor Zoom Lens (KIT MODEL) (Camera)

Good camera with quality pictures. Many features some of which never used. Holds battery life OK.

Was this review helpful to you? [Yes] [No]

**Fig 5.1 Sample review from Amazon**

The reviews of the format in Fig 5.1 are crawled using Amazon
reviews downloader and parser. Then product id is given as input to the
downloader. The data format after downloading is shown in Fig 5.2.

```
product/productId: B003ZYF3LO
review/userId: A1RSDE90N6RSZF
review/profileName: Joseph M. Kotow
review/helpfulness: 9/9
review/score: 5.0
review/time: 1042502400
review/text: Good camera with quality pictures. Many features some of
which never used. Holds battery life OK.
```

**Fig 5.2 Data format of downloaded reviews**
Then the reviews are extracted and sentiment class is assigned based on the review score in the data format (sentiment class is positive if review score is $\geq 3$, else class label is assigned negative). For each of the review sentences, the product features are identified by POS tagging as mentioned in section 3.3 (Appendix A). The sample product features identified for dataset IBD1 is shown in Table 5.1.

| Table 5.1   Sample product features identified for the dataset (IBD1) |
|-------------|---------------------------------------------------------------------|
| Unigram features | camera, digital, g, price, battery, flash, quality, setting, lens, lcd, manual, viewfinder, light, mode, zoom, use, software, optical, picture, canon, lag, mp, download, speed. |
| Bigram features | digital camera, lens quality, manual, optical zoom, mega pixel, metering option, movie mode, battery life, image download, compact flash, lag time, auto mode, raw format, exposure control, indoor picture, indoor image, manual function. |
| Trigram features | indoor image quality, light auto correction, hot shoe flash, mb memory card |

After product feature identification, the preprocessed review sentences are transformed into a vector space representation. Three different vector space models are created with various combinations of n-grams. A sample sub process flow for vector creation using rapidminer tool is shown in Fig 5.3. The features used in vector space representation are reduced by applying PCA (Appendix B). After feature reduction, vector space representation is reconstructed again with reduced feature set. This vector space representation is needed as input for the classification methods.
5.1. Experimental Analysis on Balanced Datasets

The main objective of this experiment analysis is to compare the performance of the various hybrid combinations of

i. PCA with individual classifiers

ii. PCA with ensembles

iii. PCA with neural network based individual classifier

for sentiment mining using balanced datasets. Three experimental analysis were conducted with two different balanced datasets. The first analysis aimed at identifying the optimal learning classifier among the three individual (statistical / probabilistic based) classifiers. The second experimental analysis attempted to compare the effectiveness of two popular ensemble based classifiers. The third experimental analysis is done to empirically evaluate the performance of the neural network based classifiers on two balanced datasets. In the following subsections, the motivation behind choosing each of the classifiers and the implementation details of the classifiers are discussed.
5.1.1. Individual classifiers (statistical/ probabilistic)

To compare the performance of the proposed hybrid classifiers, the following popular statistical and probabilistic based individual classifiers are implemented to show the effectiveness of the proposed hybrid approaches.

5.1.1.1. Support vector machine (SVM)

SVM is powerful classifier arising from statistical learning theory that has proven to be efficient for various classification tasks in text categorization. The motivation behind choosing SVM as classifier is SVM is robust in high dimensional spaces and any feature is relevant. SVM’s are robust when there is a sparse set of samples. In addition, SVM possesses the best performance for the text sentiment classification problem (Gamon.M. (2004), Pang.B. et al., (2002), Tan.H. et al., (2008)). Therefore, SVM is adopted to construct the classifier in the present work.

SVM model is employed using weka tool. The kernel type chosen is a polynomial kernel with default values for most of the parameters. Some parameters chosen for SVM model is shown in the Table 5.2.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel type</td>
<td>Type of kernel</td>
<td>polynomial</td>
</tr>
<tr>
<td>Cache Size</td>
<td>The size of the cache (kernel)</td>
<td>1</td>
</tr>
<tr>
<td>Exponent</td>
<td>The exponent value. (kernel)</td>
<td>1.0</td>
</tr>
<tr>
<td>C</td>
<td>Complexity parameter</td>
<td>10.0</td>
</tr>
</tbody>
</table>
The knowledge flow model of the hybrid combination (PCA + SVM) is shown in Fig 5.4.

Fig 5.4 Knowledge flow of SVM

5.1.1.2. Logistic regression (LR)

Logistic regression is a standard technique based on maximum likelihood estimation. The main motivation behind choosing logistic regression modeling is for the following reasons:

- logit modeling is well known, conceptually simple and frequently used in marketing, especially at the level of the individual consumer.
- The ease of interpretation of logit is an important advantage over other methods
- logit modeling has been shown to provide good and robust results in general comparison studies of text classification (Peduzzi.P. et al., (1996), Neslin.S.A. et al., (2006)).

Logistic regression model is also employed using weka tool. The model is used with default values for classification parameters. The knowledge flow representation of the hybrid model (PCA + LR) is shown in Fig 5.5.
5.1.1.3. Naive bayes (NB)

Among the supervised learning methods, NB and SVM are always in the comparison list. NB is a generative classifier and is considered a simple but effective classification algorithm. Despite its simplicity, the naive bayes classifier is a popular machine learning technique for text classification, and it performs well in many domains. Naive bayes model is also employed using weka tool. The model is used with default values for classification parameters. The knowledge flow representation of the hybrid model (PCA +NB) is shown in Fig 5.6.
5.1.2. Ensemble based classifiers

In recent years, there has been a growing interest in using ensemble learning techniques, to enhance the classification accuracy. Previous theoretical and empirical research has shown that an ensemble is often more accurate than any of the single classifiers in the ensemble. However, compared with other research domains, related work about ensemble methods contributing to sentiment classification are still limited and more extensive experimental work is needed in this area (Li.W. et al., (2012), Tsutsumi.K. et al., (2007)).

5.1.2.1. Hybrid ensemble method-1 (HEM1)

The hybrid ensemble method-1 is built using the combination of PCA and bagged SVM ensemble. A Bagging classifier is a collection of several classifiers whose individual decisions are combined in such a way to classify the test examples. It is known that the combined model often shows much better performance than the individual classifiers used. SVM has been known to show a good generalization performance and is easy to learn exact parameters for the global optimum. Due to these advantages, their combination may not be considered as a method for increasing the classification performance. However, when implementing SVM practically, approximated algorithms have been used in order to reduce the computation complexity of time and space. Thus, a single SVM may not learn exact parameters for the global optimum. Sometimes, the support vectors obtained from the learning are not sufficient to classify all unknown test examples completely. So, it is not a guarantee that a single SVM always provides the global optimal classification performance overall test examples.

The bagging model is employed using weka tool. SVM is used as base classifier and the number of iterations used is 5. The other parameters
for base learner use the default values available in the tool. The knowledge flow representation of the hybrid model (PCA + bagged ensemble) is shown in Fig 5.7.

![Fig 5.7 Knowledge flow of HEM1](image)

**5.1.2.2. Hybrid ensemble method-2 (HEM2)**

The HEM2 is built using the combination of PCA and bayesian boosting ensemble methods. Most research on ensemble is currently concentrated on the integration of decision trees or neural networks. The ensembles of the simple bayesian classifiers have traditionally not been in a focus of research. One reason for this is that the simple bayes relies on an assumption that the attributes used for deriving a prediction are independent of each other, given the predicted value. Another reason is that the simple bayes is an extremely stable learning algorithm, while many ensemble techniques are variance reduction techniques (as bagging) not being able to benefit from the integration of the simple bayesian classifiers.

However, it has been recently shown that the simple bayes can be optimal even when the independence assumption is violated by a wide margin (Whitehead.M. et al., (2010)). Second, the simple bayes can be effectively used in ensemble techniques, which also performs bias
reduction, as boosting. The proposed bayesian boosting model is employed using rapidminer tool. Naive Bayes classifier is used as inner classifier and the number of iterations to combine the classifier is 5. Other parameters are used with default values. The Fig 5.8 (a) shows the main process work flow implementation of hybrid bayesian boosting. The Fig 5.8 (b) shows the sub process work flow implementation of validation component of hybrid bayesian boosting.

**Fig 5.8 (a) Work flow of HEM2 main process**

**Fig 5.8 (b) Work flow of HEM2 sub process**

### 5.1.3. Neural network based classifiers

Existing works on the effectiveness of neural network based models have been mainly conducted on text based topic classification (M. Ghiassi et al., (2012)). There is a lack of a comparative study on the effectiveness of neural network based models in text sentiment classification. Given the importance of text sentiment classification in the real world applications, it is believed that a comparative study of neural network based models in sentiment mining will greatly benefit application developers as well as researchers in related areas.
Specifically, the effectiveness of the neural network based models in sentiment classification is investigated as the interest of this study for three reasons.

- First, neural networks based models has been very successfully applied to text classification and many other supervised learning tasks (Ur-Rahman.N et al.,(2012), Ghiassi.M. et al., (2012)).

- The deep architectures of neural networks with layers (hidden) represents intelligent behavior more efficiently than "shallow architectures" like SVMs.

- The major features of neural networks such as adaptive learning, parallelism, fault tolerance, and generalization provide superior performance.

5.1.3.1. Hybrid neural network method-1 (HNM1)

The HNM1 is built using the combination of PCA and BPN methods. The BPN has been selected as the basic learner based on its strength of fault tolerance. Among the feed forward networks, BPN is the best known networks and it remains one of the most useful ones. This iterative gradient algorithm is designed to minimize the mean square error between the actual output of a multilayer feed forward perceptron and the desired output. In order to obtain optimal neural network architecture, different architectures are tested. The architectures are varied by changing the number of hidden layer neurons, learning rate, momentum rate and epochs. Table 5.3 summarizes the details of suitable architecture for the models I, II and III of BD1 and BD2. The neural network architecture is designed using weka tool (Fig 5.9).
Table 5.3 BPN parameters

<table>
<thead>
<tr>
<th>Model</th>
<th>Neurons in three layers</th>
<th>Learning rate</th>
<th>Momentum</th>
<th>Gain</th>
<th>Epochs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balanced dataset-1 (BD1)</td>
<td>4,7,2</td>
<td>0.1</td>
<td>0.4</td>
<td>1</td>
<td>358</td>
</tr>
<tr>
<td>II</td>
<td>6,19,2</td>
<td>0.1</td>
<td>0.4</td>
<td>1</td>
<td>379</td>
</tr>
<tr>
<td>III</td>
<td>6,19,2</td>
<td>0.1</td>
<td>0.4</td>
<td>1</td>
<td>388</td>
</tr>
<tr>
<td>Balanced dataset-2 (BD2)</td>
<td>9,12,2</td>
<td>0.1</td>
<td>0.4</td>
<td>1</td>
<td>453</td>
</tr>
<tr>
<td>II</td>
<td>14,19,2</td>
<td>0.1</td>
<td>0.4</td>
<td>1</td>
<td>469</td>
</tr>
<tr>
<td>III</td>
<td>16,23,2</td>
<td>0.1</td>
<td>0.4</td>
<td>1</td>
<td>498</td>
</tr>
</tbody>
</table>

Fig 5.9 Knowledge flow of HNM1

5.3.1.2. Hybrid neural network method-2 (HNM2)

The HNM2 is built using the combination of PCA and PNN methods. Though few studies exist in sentiment classification using neural networks, the literature does not contribute much work in sentiment classification using the probabilistic neural network in sentiment mining of product reviews to our knowledge. But many researchers have proved that PNN model is more effective than other models for data classification in various other domains

**Fig 5.10 (a) Work flow of HNM2 main process**

**Fig 5.10 (b) Work flow of HNM2 sub process**

PNN is implemented using the knime tool. The size of the training dataset is the number of (500 for BD1 and 1200 for BD2) neurons in the hidden layer. The smoothing factor value is 1 for the models I, II and III. The cross validation meta node in Fig 5.10 (a) encapsulates an inner workflow shown in Fig 5.10 (b) which is executed several times. The results of each iteration are collected and presented in aggregate through the output ports of the cross validation meta node. The first
output port reports the predicted class values for each row while the second output port reports the error rates for each iteration.

5.2. Experimental Analysis on Imbalanced Datasets

The main objective of this experiment analysis is to compare the performance of the various hybrid approaches (PCA + classification method) to deal with the class imbalance problem. The classification approaches used for sentiment prediction using imbalanced datasets are as follows,

1. Single SVM without resampling
2. Data level imbalance handling
   a. Single SVM with random under sampling
   b. Single SVM with SMOTE
3. Algorithm level imbalance handling
   a. Bagged SVM without resampling
   b. Modified bagging (M-bagging).

Three experimental analysis are conducted with two different imbalanced datasets. As outlined in the previous chapter there are two approaches to dealing with the class imbalance problem, these are data methods and algorithmic methods. Data methods utilized in this study includes RUS and SMOTE. All (SVM and RUS) methods except SMOTE are implemented directly in rapidminer, while SMOTE is implemented in weka.

5.2.1. Single SVM without resampling

SVM has a superior generalization capability. Geometrically, the SVM modeling algorithm works by constructing a separating hyperplane
with the maximal margin. Compared with other standard classifiers, SVM is more accurate on moderately imbalanced data (Li.S. et al., (2012)). The reason is that only support vectors (SVs) are used for classification and many majority samples far from the decision boundary can be removed without affecting classification. SVM is specifically chosen to attack the problem of imbalanced data, because SVM is based on strong theoretical foundations. Its unique learning mechanism makes it an interesting candidate for dealing with imbalanced data sets, since SVM only takes into account those instances that are close to the boundary, i.e. the support vectors, for building its model. The knowledge flow model of the hybrid combination (PCA +SVM) is same as shown in Fig 5.3 for IBD1 and IBD2.

### 5.2.2. Data level imbalance handling

These methods modify the distribution of rare and frequent patterns in order to favour the detection of the rare ones. This operation is called resampling and aims at increasing the rate of rare samples by creating a new dataset from the original one. This rebalance can be obtained by both removing samples belonging to the frequent class and adding samples to the rare one. The two techniques used are under sampling and oversampling.

#### 5.2.2.1. Single SVM with random under sampling

Random under sampling (RUS) was implemented by simply selecting a random sample of the majority class which matches the number of minority class examples. Random under sampling of the majority class was implemented using the methodology shown below in rapidminer tool as shown in Fig 5.11 (a) and 5.11 (b).
5.2.2.2. Single SVM with SMOTE

SMOTE algorithm (Chawla.N. et al., (2002)) over samples the minority class by generating artificially interpolated data. The over fitting problem is avoided in SMOTE and causes the decision boundaries for the minority class to be spread further into the majority class space. It has been reported that SMOTE has achieved favourable results in many class imbalance studies (Chawla.N et al., (2002)). SMOTE is implemented using weka tool. The value of parameters (neighbors, percentage) are set as (20,100) for IBD1 and (50,700) for IBD2. The knowledge flow representation is shown in Fig 5.12.
5.2.3. Algorithm level imbalance handling

Ensemble based technique is used to deal with imbalanced data sets to address the class imbalance problem. Modified bagging is a variation of bagging ensemble approach. Bagged ensemble is also employed for the imbalanced datasets, so as to compare with the results obtained for the modified bagging approach. Also, the bagged ensemble is identified as the best classifier for the two balanced datasets used. Bagging is implemented using weka tool. The knowledge flow representation is same as shown in Fig 5.5.

5.2.3.1. Modified bagging (M-bagging)

In the modified bagging approach, the dataset (D) is sampled into sub samples which are used to train different base learners. To address the information loss and over fitting problems arising from using either of the two sampling approaches alone, the under sampling and over sampling methods are integrated together. Given an imbalance ratio, first over sample the minority instances with the SMOTE algorithm to some extent, and then under sample the majority class so that both sides have similar amounts of instances. To under sample the majority class, the bootstrap sampling approach is used with all available
majority instances, provided that the size of the new majority class is the same as that of the minority class after running SMOTE. The benefit of this approach is that it inherits the strength of both sampling methods and eliminates the over fitting and information loss problems. The proposed model with an ensemble of SVM classifiers is graphically illustrated in Fig. 5.13.
Rebalancing is still necessary in this context because when learning from imbalanced data, it is likely that a bootstrap sample used to train a SVM in the ensemble is composed of few or even none of the minority instances. Hence, each component learner of the ensemble would suffer from severe skewness, and the improvement of using an ensemble would be limited. To overcome this drawback, a new method is proposed. The pseudo code of the proposed method, called M-bagging, is given in Fig 5.14.

The negative class (minority class) is oversampled with the SMOTE method to smooth the decision boundary (Chawla.N.V et al., (2002)). When the SMOTE method is applied to each negative instance (minority class), it finds the K-nearest neighbors, draws a line between the instance and each of its K-nearest neighbors and a point on each line is randomly selected to use as a new minority class instance. In this way, K X n new minority class instances are added to the training data, where n is the number of negative reviews in the original training data. Then the majority class is under sampled ‘M’ times to generate ‘M’ bootstrap samples so that each bootstrap sample has the same or similar size as the over sampled negative instances. Each bootstrap sample is combined with the over sampled negative reviews to form a training set to train an SVM. Therefore, M-base learners can be obtained from M different training sets. Finally, the M-base learners form an ensemble to make a prediction on a test instance through majority voting. The value of M is set to be 10, and an arbitrary class is selected when there is a tie in the vote. The proposed modified bagging approach is implemented using weka java API.
5.3. Evaluation Metrics

Machine learning has recently benefited from attention to the performance measures used in classification. Evaluation of learning algorithms concentrates on two goals: comparison of algorithms and the applicability of algorithms on a specific domain. Most of the evaluation metrics are based on the ‘confusion matrix’ for a binary classification task.

In a binary decision problem, a classifier labels examples as either positive or negative. The decision made by the classifier can be...
represented as a confusion matrix as shown in Table 5.4. The confusion matrix has four categories

<table>
<thead>
<tr>
<th></th>
<th>Predicted Positive</th>
<th>Predicted Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actual Positive</strong></td>
<td>( t_p )</td>
<td>( f_n )</td>
</tr>
<tr>
<td><strong>Actual Negative</strong></td>
<td>( f_p )</td>
<td>( t_n )</td>
</tr>
</tbody>
</table>

Where

- \( t_p \) is the number of 'positive' reviews classified as 'positive',
- \( t_n \) is the number of 'negative' reviews classified as 'negative',
- \( f_p \) is the number of 'negative' reviews classified as 'positive',
- \( f_n \) is the number of 'positive' reviews classified as 'negative'.

For classification tasks, the terms true positives, true negatives, false positives and false negatives compare the results of the classifier under test with expected values. The terms positive and negative refer to the classifier’s prediction, and the terms true and false refer to whether that prediction corresponds to the expectation.

Misclassification rate is defined as the ratio of number of wrongly classified reviews to the total number of reviews classified by the classification model (Eq. 5.3). The wrong classifications fall into two categories. If negative reviews are classified as positive (C1), it is named as type I error (Eq. 5.1). If positive are classified as negative (C2), it is named as type II error (Eq. 5.2).

\[
\text{Type I error} = \frac{C_1}{\text{Total no of positive reviews}} \quad (5.1)
\]
Type II error $= \frac{C_2}{(\text{Total no of negative reviews})}$ \hspace{1cm} (5.2)

Overall misclassification rate $= \frac{C_1 + C_2}{(\text{Total no of reviews})}$ \hspace{1cm} (5.3)

Precision and recall are two metrics in information retrieval literature and likely the most common metric used in most machine learning applications. Precision and recall values are not symmetrical, i.e. the definition of 'positive' and 'negative' labels makes a difference. If the labels are swapped, the result may change drastically. Thus the precision and recall of the classifier on each single class label is measured. Also the main focus is to find how class distribution may affect the classification accuracy, so the positive and negative performance measures are studied separately with datasets.

A review which is correctly recognized as positive is a very important source of information for sentiment analysis. Therefore, it is very important to build a classifier with high precision for the positive class (Eq. 5.4).

$$\text{Positive precision} = \frac{\text{No. of reviews correctly classified as positive}}{\text{Total no. of reviews classified as positive}}$$

i.e. Positive precision $= \frac{t_p}{t_p + f_p}$ \hspace{1cm} (5.4)

A review which is correctly recognized as negative is also an important source of information for sentiment analysis. Therefore, it is important to build a classifier with high precision for the negative class. It is also mentioned as negative predicate value (Eq. 5.5).

$$\text{Negative precision} = \frac{\text{No. of reviews correctly classified as negative}}{\text{Total no. of reviews classified as negative}}$$

i.e. Negative precision $= \frac{t_n}{t_n + f_n}$ \hspace{1cm} (5.5)
Ideally, all positive and negative reviews should be recognized correctly. The positive recall (also called the true positive rate or Sensitivity) measures the proportion of actual positive reviews which are correctly identified as such (Eq. 5.6).

\[
\text{Positive recall} = \frac{\text{No. of reviews correctly classified as positive}}{\text{Total no. of positive reviews}}
\]

\[
\text{i.e. Positive recall} = \frac{t_p}{t_p + f_n}
\] (5.6)

The negative recall (also called true negative rate or specificity) measures the proportion of actual negative reviews which are correctly identified as such (Eq. 5.7).

\[
\text{Negative recall} = \frac{\text{No. of reviews correctly classified as negative}}{\text{Total no. of negative reviews}}
\]

\[
\text{i.e. Negative recall} = \frac{t_n}{t_n + f_p}
\] (5.7)

The combination of precision and recall values are called f-score, and in most common form, it is the harmonic mean of both, where a f-score reaches its best value at 1 and worst score at 0 (Eq. 5.8 and 5.9).

\[
\text{Positive f – score} = \frac{2 \times \text{Positive precision} \times \text{Positive recall}}{\text{Positive Precision} + \text{Positive recall}}
\] (5.8)

\[
\text{Negative f – score} = \frac{2 \times \text{Positive precision} \times \text{Negative recall}}{\text{Negative Precision} + \text{Negative recall}}
\] (5.9)

The above formulation is for the case where precision and recall have equal weights. Precision corresponds to ratio of correctness and recall measures ratio of completeness. In some cases high correctness may be more important, and in some cases high completeness may be more important. However, most cases, aims at improving both values. Low correctness means that a high percentage of the classes being classified
are wrongly assigned. So there is a waste of developer’s effort in correcting them. Hence, correctness is expected to be high always. Completeness is a measure of the percentage of reviews that would have been found if the prediction model is used in the stated manner.

Accuracy is a weighted arithmetic mean of precision and inverse precision as well as a weighted arithmetic mean of recall and inverse recall. Inverse precision and recall are simply the precision and recall of the inverse problem where positive and negative labels are exchanged. Recall and inverse recall are equivalently true positive rate and false positive rate. The application of recall, precision and f-measure are argued to be flawed as they ignore the true negative cell of the contingency table. This problem is solved by using accuracy (Eq. 5.10).

\[
\text{Accuracy} = \frac{(t_p + t_n)}{(t_p + f_p + f_n + t_n)} \tag{5.10}
\]

Traditional metrics like precision and recall discussed has been the most commonly used empirical measures. However, in the framework of imbalanced data sets, these metrics are no longer a proper measure, since it does not distinguish between the numbers of correctly classified examples of different classes. Hence, it may lead to erroneous conclusions, i.e., a classifier that achieves an accuracy of 90% in a data set with an imbalance value of 9, is not accurate if it classifies all examples as negatives.

For this reason, when working in imbalanced domains, there are more appropriate metrics to be considered instead of accuracy. Specifically, four metrics are obtained from Table 5.4 to measure the classification performance of both, positive and negative, classes independently.
1) True positive rate ($TP_{rate}$) is the percentage of positive instances correctly classified.

2) True negative rate ($TN_{rate}$) is the percentage of negative instances correctly classified.

3) False positive rate ($FP_{rate}$) is the percentage of negative instances misclassified.

4) False negative rate ($FN_{rate}$) is the percentage of positive instances misclassified.

Clearly, since classification intends to achieve good quality results for both classes, none of these measures alone is adequate by itself. One way to combine these measures and produce an evaluation criterion is to use the receiver operating characteristic (ROC) graphic. This graphic allows the visualization of the trade-off between the benefits ($TP_{rate}$) and costs ($FP_{rate}$) thus, it evidences that any classifier cannot increase the number of true positives without the increment of the false positives. Thus, ROC curves allow for a visual comparison of classifiers.

ROC space is plotting on a two-dimensional chart, the $TP_{rate}$ ($Y$-axis) against the $FP_{rate}$ ($X$-axis). Points in (0, 0) and (1, 1) are trivial classifiers where the predicted class is always the negative and positive, respectively. On the contrary, (0, 1) point represents the perfect classification. The area under the ROC curve (AUC) provides a single measure of a classifier’s performance for the evaluation that which model is better on average. The larger the area below the ROC curve, the higher classification potential of the classifier. The AUC measure is computed just by obtaining the area of the graphic using trapezoidal approximation method.
In the context of class imbalance problem, however, the higher rate of correct detection on the minority class is particularly required. Hence, “accuracy” is obviously not suitable any more. Recently, some researchers in this area have realized this problem, and proposed another metric. With the help of confusion matrix, the performance measure is expressed as follows (Eq. 5.11),

\[
g\text{– mean} = \sqrt[2]{\frac{t_p}{t_p+n_f} \times \frac{t_n}{t_n+f_p}} \tag{5.11}
\]

\(g\text{-mean}\) is based on the recalls of both classes. The benefit of selecting this metric is that it can measure how balanced the combination scheme is. If a classifier is highly biased toward one class (such as the majority class), the \(g\text{-mean}\) value is still low.