Applied-Statistics Project

Binary Classification

Mellissa HAFIS
Idrissa DICKO
Alois VINCENT

A report presented for Applied Statistics course AI track

université
PARIS-SACLAY

FACULTÉ
DES SCIENCES
D'ORSAY

31/10/2024

Binary classification

Mellissa HAFIS
Idrissa DICKO
Alois VINCENT

Abstract

In this project, we focused on the evaluation of binary classification predictors in the context of both balanced and unbalanced datasets. Our primary objectives included building a comprehensive dataset that facilitates the assessment of predictor quality, learning effective prediction techniques applicable to different dataset distributions, and employing various quality estimators and metrics for a robust evaluation. Additionally, we explored strategies for model optimization and selection, aiming to identify the most effective model for specific classification problems. Thus, through this investigation, we enhanced our understanding of the intricacies involved in binary classification. And found that the best predictor for our dataset was the logistic regression.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The ability to accurately classify data points into distinct categories is a fundamental task in data science and statistics, particularly in the realm of binary classification. This report aims to explore the evaluating binary predictors by focusing on how different dataset characteristics—specifically balance and imbalance—impact the performance of classification models.

Building a suitable dataset is crucial for understanding the behavior of various predictors. In this project, we construct an unbalanced dataset with two clusters in high dimension, enabling us to analyze how well different models can perform under these conditions.

Moreover, we look at predictive methodologies, highlighting the need for careful evaluation of model performance. We utilize specific metrics, including F1 Score, Specificity, and Negative Predictive Value, to measure the effectiveness of each predictor. These metrics provide insight into the strengths and weaknesses of the models, allowing us to make informed decisions based on empirical evidence.

Lastly, this project includes a focus on model optimization, exploring techniques to enhance model performance and selecting the most appropriate model for given classification challenges. By integrating these elements, we aim to develop a comprehensive understanding of binary classification and improve our skills in applying statistical methodologies to real-world problems.

Chapter 2

Methodology

For this project we will create an artificial dataset with two clusters in high dimensions. We will assign a label 0 and 1 to each of these clusters and the goal is to evaluate the performance of different predictors to classify which entry belongs to each cluster. During the training and evaluation process, we will start by evaluating the quality of the predictions and secondly fine tune the parameters to optimize the performance of each predictor. In the end, we will rank all three predictors based on their performance.

We plan to study the following predictors:

1. **Random assignment**
   - Strategy: Randomly assign label 1 based on a predefined probability (p).
   - Parameter to Tune: The probability (p) of choosing label 1.
2. **PCA-based predictor**
   - Strategy: Use principal component analysis (PCA) to assign label 1 based on the first principal component (PC1).
   - Parameter to Tune: The threshold (a) so that label 1 is selected when PC1 < (a).
3. **Logistic Regression**
   - Strategy: Implement logistic regression to estimate the probability that a given entry belongs to label 1.
   - Tuning parameter: The threshold for determining the label based on the estimated probability.

In order to effectively evaluate the performance of binary classifiers, we employed a collaborative approach involving dividing the work among three team members, each focusing on a pair of predictors. This structure allowed for a comprehensive examination of multiple methodologies and facilitated direct comparison of results.

2.1 Data generation

In this project, we generated a synthetic dataset specifically designed for binary classification tasks. The data generation process involved creating two random clusters for the test classification, with the following characteristics:

1. **Dataset Structure:** The dataset is represented as a matrix X containing 10,000 samples (rows) and 20 features (columns). Each feature represents a different dimension of the data, contributing to the overall characteristics of the samples.

2. **Labels:** The target variable y is structured as a vector with n entries, where each entry corresponds to a label indicating the class of the sample. The labels can take on one of two values: 0 (representing the false class) or 1 (representing the true class).

3. **Class Distribution:** We defined two distinct classes within the dataset:
   - **False Class:** Labeled as 0.
   - **True Class:** Labeled as 1.

4. **Imbalance in Classes:** The dataset is unbalanced, as the probability of a sample belonging to the true class (label 1) is set to 0.7, while the probability of belonging to the false class (label 0) is 0.3. This imbalance reflects common scenarios in real-world data where one class may significantly outnumber the other.

5. **Cluster Configuration:** Each class consists of two clusters, which represent the inherent grouping of samples within each class. The default value for the number of clusters was utilized to capture the data's structure effectively.

6. **Class Separation:** The separation between the two classes was established at a default value of 1. This parameter helps define the distance between the clusters associated with the true and false classes, influencing the classification boundaries.

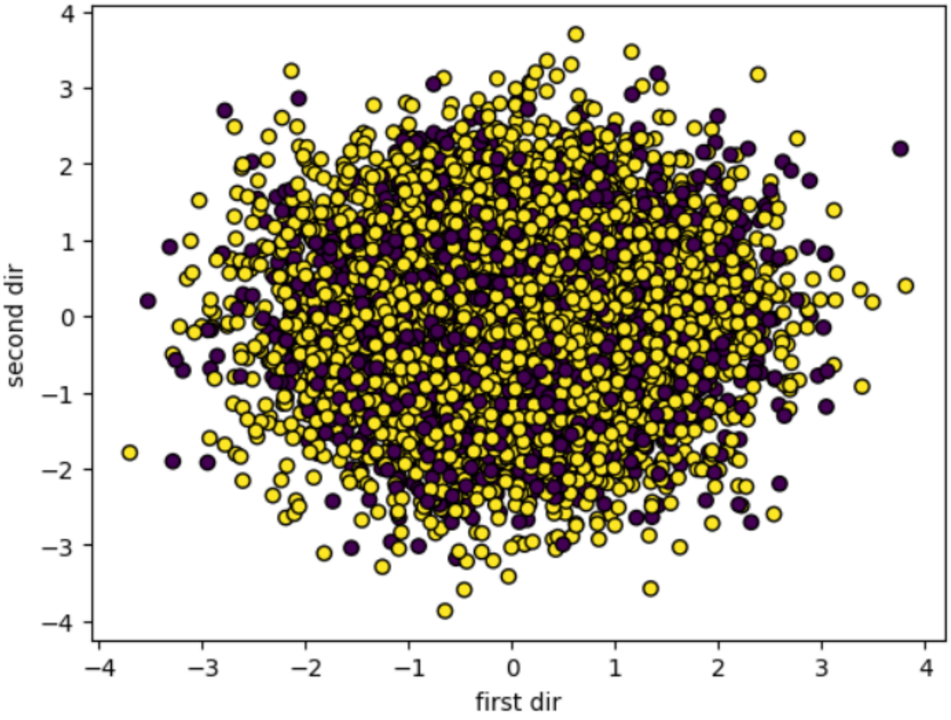


Figure 2.1: Generated data set

By generating this synthetic dataset, we aimed to create a controlled environment for evaluating the performance of various binary classification models while examining the effects of class imbalance and clustering on predictive accuracy.

2.2 Dimensional Reduction and Visualization

To further investigate the distribution of the two classes within our dataset, we employed Principal Component Analysis (PCA) as a dimensionality reduction technique. PCA enables us to project high-dimensional data into a lower-dimensional space, facilitating easier visualization and analysis of the data structure.

We applied PCA to our training data, resulting in a two-dimensional representation that captures the most significant variance in the dataset. This allowed us to verify the presence of two distinct classes of data points visually and there is two clusters per class of data points.

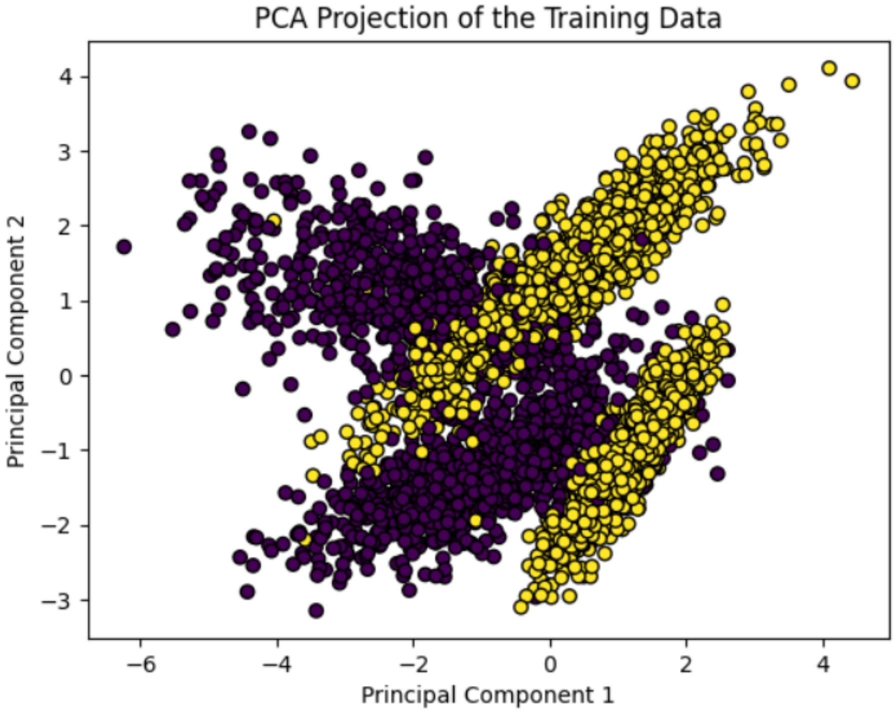


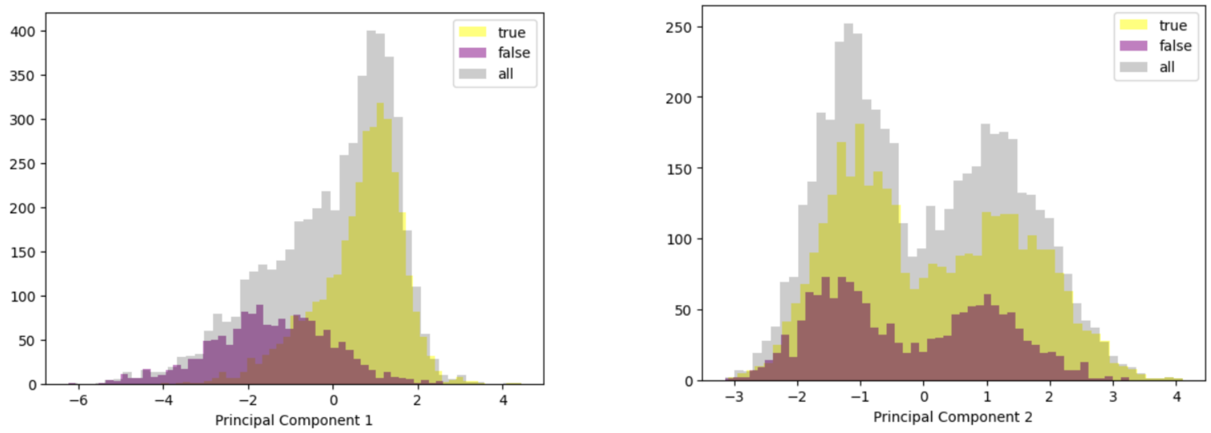
Figure 2.2: Visualization of the two distinct classes of data points

PCA Visualization We generated histograms for the first two principal components to visualize the distribution of samples belonging to the true class (label 1) and the false class (label 0).

In the histograms:

- The yellow bars represent the distribution of true class samples (label 1).
- The purple bars represent the distribution of false class samples (label 0).
- The gray bars provide a comprehensive view of the entire dataset.

We notice the size of the true class is larger than the size of the false class. Moreover each of them follows a normal distribution.



(a) Distribution of samples in Principal Component 1

(b) Distribution of samples in Principal Component 2

Figure 2.3: PCA Visualization

The plots allow us to visually assess how well the two classes are separated in the PCA-reduced space, providing insights into the classification potential of the selected models.

Chapter 3

Definition of the different predictors

1. **No skills** The first predictor is a random guess. We will assign the label 1 to each entry with a probability (p).

2. **Naive: PCA based predictor** The second predictor is based on the first principal component. We will assign the label 1 to each entry with a value of the first principal component smaller than a given threshold. ***Why to choose the first component ?*** *Choosing the first principal component (PC1) in PCA is common because: By focusing on the first principal component, we give more importance to variance in the data for classification tasks. It's a practical approach that balances complexity and interpretability while aiming to achieve good predictive performance.*

3. **Logistic regression predictor** We can learn to classify this data using logistic regression. I include a discussion about its training, but we do not need this for this lab.

   Let us describe the data in terms of a probability function

   $$p_{A,B}(\boldsymbol{x}) = \frac{1}{1 + e^{-(A+\boldsymbol{B}\cdot\boldsymbol{x})}},$$

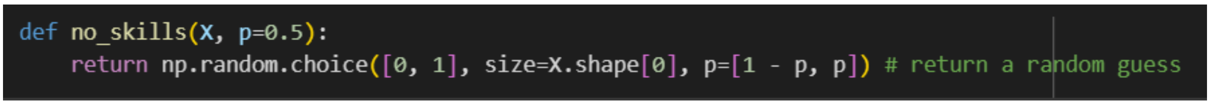
   using labeled data $(\boldsymbol{x}_k, y_k)$. We want to train the model such that the probability $p_k = p_{A,B}(\boldsymbol{x}_k) = 1$ for $y_k = 1$ and 0 for $y_k = 0$. For this purpose, we can define a log loss

   $$L = -\sum_k y_k \log p_k - (1 - y_k) \log p_k$$

   (it is clear that L is minimized $p_k = 1$ when $y_k = 1$ and $p_k = 0$ when $y_k = 0$). One can then minimize it by calculating the gradient

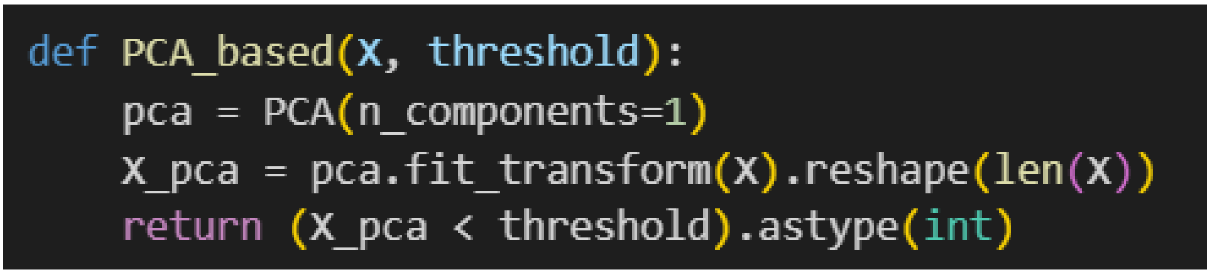
   $$\frac{\partial L}{\partial A} = \sum_k (p_k - y_k) = 0$$

   $$\frac{\partial L}{\partial \boldsymbol{B}} = \sum_k \boldsymbol{x}_k (p_k - y_k) = 0$$



```python
def no_skills(X, p=0.5):
    return np.random.choice([0, 1], size=X.shape[0], p=[1 - p, p]) # return a random guess
```

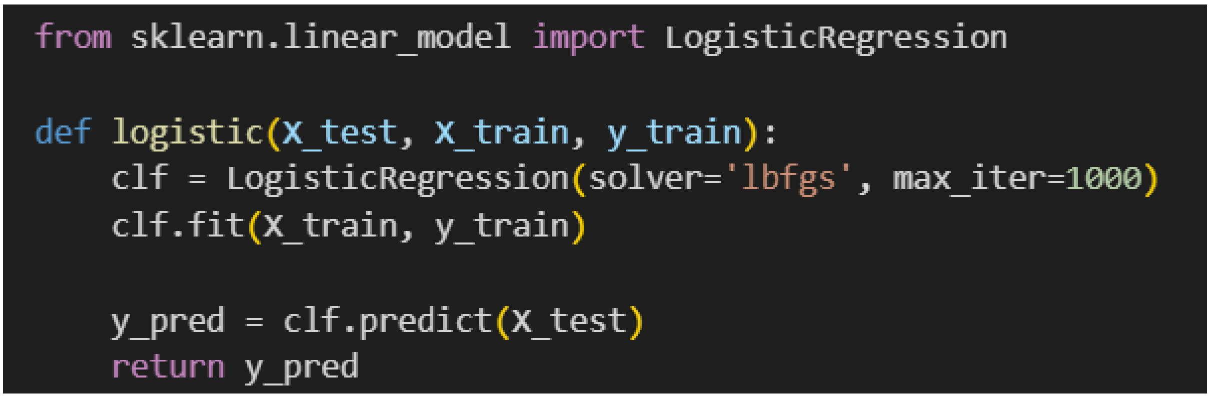
Figure 3.1: No-skills predictor model



```python
def PCA_based(X, threshold):
    pca = PCA(n_components=1)
    X_pca = pca.fit_transform(X).reshape(len(X))
    return (X_pca < threshold).astype(int)
```

Figure 3.2: PCA-bases predictor model

Which is solved numerically. As a result of this process, we obtain a way to assign to each data point a probability of being in the category $y = 1$.



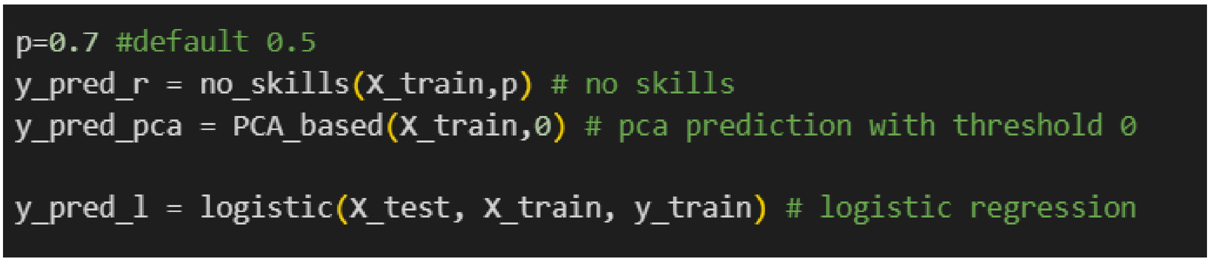
```python
from sklearn.linear_model import LogisticRegression

def logistic(X_test, X_train, y_train):
    clf = LogisticRegression(solver='lbfgs', max_iter=1000)
    clf.fit(X_train, y_train)

    y_pred = clf.predict(X_test)
    return y_pred
```

Figure 3.3: Logistic regression model

Chapter 4

Predictions and model evaluation

4.1 Making predictions:



```
p=0.7 #default 0.5
y_pred_r = no_skills(X_train,p) # no skills
y_pred_pca = PCA_based(X_train,0) # pca prediction with threshold 0

y_pred_l = logistic(X_test, X_train, y_train) # logistic regression
```

Figure 4.1: Deriving the predictions by using each model defined in the chapter above

- **No-Skills Model** This model does not involve training, as it simply makes random predictions based on a predefined probability distribution for the two classes.

- **PCA-Based Model** The PCA model was not directly trained in the conventional sense but was evaluated by varying a threshold over the first principal component values in the test set.

- **Logistic Regression** The Logistic Regression model was trained using the training dataset, with cross-entropy loss optimized over 1,000 iterations. The model was then used to generate class probabilities for the test set, and different threshold values were applied to convert these probabilities into binary predictions.

Remark: *Given that our dataset has an imbalance where about 70% of the cases are positive (i.e., $P(1)\ \ 0.7$), setting $p = 0.7$ for the no-skills predictor will provide a more relevant baseline that reflects our dataset's characteristics. This approach ensures that the no-skills predictor has the same proportion of positive and negative predictions as our actual data, making comparisons with other models more meaningful. This approach gives a realistic reference point that reflects the dataset's inherent imbalance and allows for fairer comparisons with other models, such as the PCA-based and logistic regression models.*

4.2 Confusion matrix for evaluating the models:

4.2.1 An introduction to confusion matrix

The confusion matrix is a table that is often used to describe the performance of a classification model on a set of data for which the true values are known. The confusion matrix is a 2x2 table with counts of the following events:

- True Negative (TN): The number of correct predictions that an entry is in the negative class.
- False Positive (FP): The number of incorrect predictions that an entry is in the positive class.
- False Negative (FN): The number of incorrect predictions that an entry is in the negative class.
- True Positive (TP): The number of correct predictions that an entry is in the positive class.

We can visualize the confusion matrix as follows:

/	Pred -	Pred +
Real 0	TN	FP
Real 1	FN	TP

Table 4.1: Confusion matrix

It is more convenient to divide by the total number of entries to have an idea of the proportions of each case.

/	Pred -	Pred +
Real 0	TNR	FPR
Real 1	FNR	TPR

Table 4.2: Confusion matrix with rates (probabilities)

4.2.2 Interpretation of results:

Before we start using the confusion matrix, we will make the connection with probability theory. So, let's define a set of two pairs mutually incompatible events:

- True cluster being part of cluster [0,1]
- Predicted label: getting [,+]

At the end of this section we will compute the conditional probabilities $P(0|+)$, $P(1|+)$, $P(0|-)$, $P(1|-)$, and the joint probabilities $P(0 \cap +)$, $P(1 \cap +)$, $P(0 \cap -)$, $P(1 \cap -)$ and use theme to evaluate the quality of each predictor.

After defining each model and calculating the values of y_pred with each predictor we have the following results The confusion matrix:

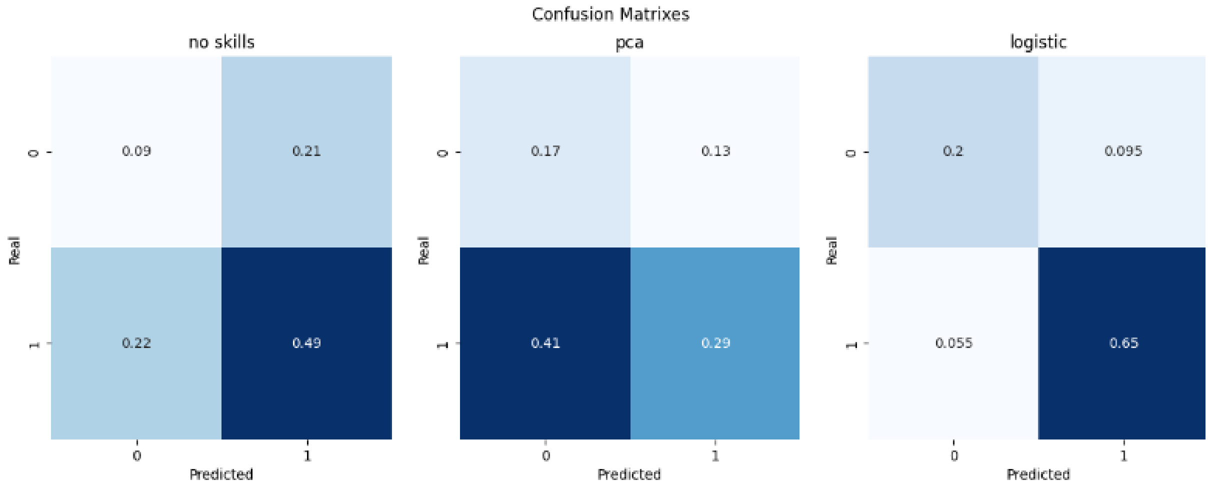


Figure 4.2: Confusion matrix of the three predictor, No-skills, PCA-based, Logistic regression.

1- The best and the worst predictor

To say which is the best and the worest predictor we will make a comparison among the three predictors based on both the values of (TPR and TNP) which represent the ratio of correct predictions and the values of (FPR and FNR) which represent the ration the incorrect predictions.

Before starting this analysis there are some important definitions:

- $TPR = P(+|1) = P(+ \cap 1)/P(1)$ and by Bayes' Theorem $TPR = P(+).P(1|+)/P(1)$
- $TNR = P(-|0) = P(- \cap 0)/P(0)$ and by Bayes' Theorem $TNR = P(-).P(0|-)/P(0)$
- $FPR = P(+|0) = P(+ \cap 0)/P(0)$ and by Bayes' Theorem $FPR = P(+).P(0|+)/P(0)$
- $FNR = P(-|1) = P(- \cap 1)/P(1)$ and by Bayes' Theorem $FNR = P(-).P(1|-)/P(1)$
- $P(1) = np.sum[y\_test == 1]/len(y\_test)$
- $P(0) = np.sum[y\_test == 0]/len(y\_test)$
- $P(+) = np.sum[y\_pred == 1]/len(y\_pred)$
- $P(-) = np.sum[y\_pred == 0]/len(y\_pred)$
- C is the event ”Have a correct prediction”
- I is the event ”Have an incorrect prediction”
- $P(C) = TPR + TNR$
- $P(I) = FPR + FNR$

Here is a analysis of the results shown in the confusion matrix:

1. **No skills predictor:**

- 50% of TP and 9.3% of TN, this means 59,3% of the predictions are correct. Which means:
    - $P(C) = TPR+TNR = 0.5+0.093 = 0.593$ ie the probability of classifying a point x in the True class is 0.593
- And we have 20% of FT and 20% of FN, this means 40% of the predictions are incorrect. Then:
    - $P(I) = FPR + FNR = 0.2 + 0.2 = 0.4$ ie the probability of classifying a point x in the False class is 0.4

2. **PCA based predictor:**
    - 29% of TP and 17% of TN, this means 46% of the results are correct. Which means:
        - $P(C) = TPR+TNR = 0.29+0.17 = 0.46$ ie the probability of classifying a point x in the True class is 0.46
    - And we have 41% of FT and 13% of FN, this means 54% of the predictions are incorrect. Then:
        - $P(I) = FPR+FNR = 0.41+0.13 = 0.54$ ie the probability of classifying a point x in the False class is 0.54

3. **Logistic regression predictor:**
    - 65% of TP and 20% of TN, this means 85% of the results are correct. Which means:
        - $P(C) = TPR + TNR = 0.65 + 0.2 = 0.85$ ie the probability of classifying a point x in the True class is 0.85
    - And we have 5.5% of FT and 9.5% of FN, this means 15% of the predictions are incorrect. Then:
        - $P(I) = FPR + FNR = 0.055 + 0.095 = 0.15$ ie the probability of classifying a point x in the False class is 0.15

To summarize we plotted the result in the figure 4.3. Then we conclude that:

- Based on the $P(C)$ the best predictor is the logistic regression predictor
- Based on the $P(I)$ the worst predictor is the PCA based predictor

Moreover, from the previous computations we can now compute the values of the conditional probabilities $P(0|+)$, $P(1|+)$, $P(0|-)$, $P(1|-)$, and the joint probabilities $P(0 \cap +)$, $P(1 \cap +)$, $P(0 \cap -)$, $P(1 \cap -)$:

- $P(1) = np.sum[y\_test == 1]/len(y\_test) = 0,70340,7 = p$
- $P(0) = np.sum[y\_test == 0]/len(y\_test) = 0,29660,3 = 1 - p$
- $P(1|+) = TPR.P(1)/P(+)$
- $P(0|+) = FPR.P(0)/P(+)$

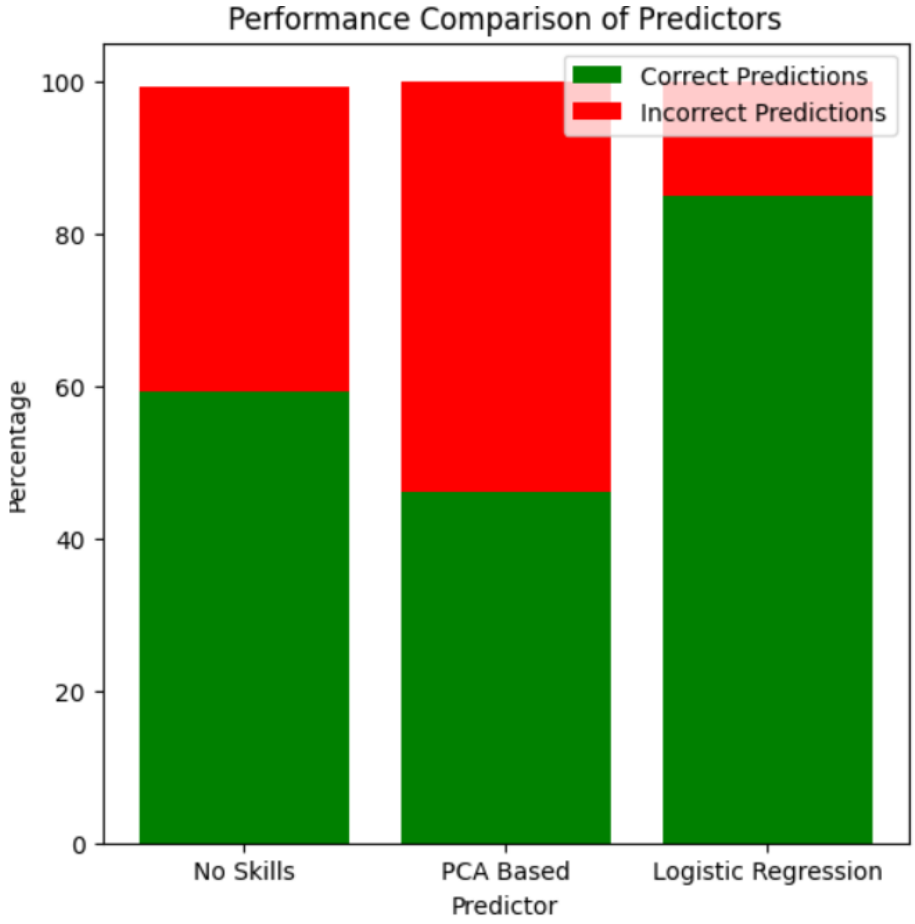


Figure 4.3: Histogram showing the correct and incorrect prediction rate of each predictor

- $P(0|-) = TNR.P(0)/P(-)$
- $P(1|-) = FNR.P(1)/P(-)$
- $P(1 \cap +) = P(1) * TPR$
- $P(0 \cap +) = P(0) * FPR$
- $P(0 \cap -) = P(0) * TNR$
- $P(1 \cap -) = P(1) * FNR$

We did all the computations, For more details on the calculations of numerical values...etc, please consult the notebook attached to this report, and and we visualize them in the histogram of the figure 4.4. Finally, knowing that the **conditional probabilities** $P(0|+)$**,** $P(1|+)$**,** $P(0|-)$**,** $P(1|-)$, show **the quality of the predictions** for each model by evaluating the likelihood that predictions match the true classes. They help us understand, in probabilistic terms, how well the model discriminates between classes in real situations. Higher values of $P(1|+)$ and $P(0|-)$ **indicate that the model effectively uses information from the data to predict accurately**, whereas higher values of $P(0|+)$ and $P(1|-)$ indicate issues like false positives and false negatives. And the joint probabilities $P(0 \cap +)$, $P(1 \cap +)$, $P(0 \cap -)$, $P(1 \cap -)$ show how well the model classifies each class and identifying where it makes the most mistakes.

In conclusion:

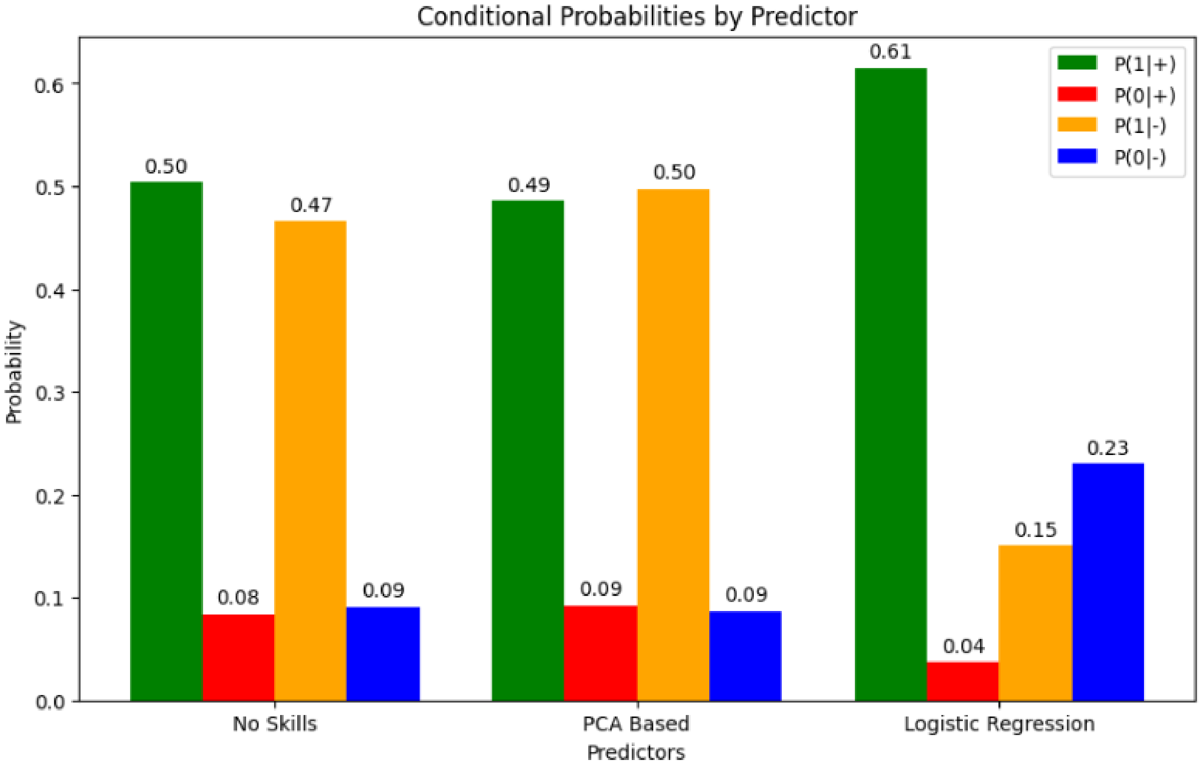


Figure 4.4: Histogram showing the conditional probabilities of each predictor

- Logistic Regression has the highest rate of correct classifications both in terms of true positives $P(1|+)$ and true negatives $P(0|-)$, while also achieving the lowest rates of false positives and false negatives.
- PCA-Based performs worst, with lower accuracy in identifying true positives and a higher rate of false negatives, showing it doesn't use the information effectively compared to Logistic Regression.
- No Skills performs slightly better than PCA but worse than Logistic Regression, as expected for a baseline model.

Which confirms the first conclusion made on the percentage of the correct and incorrect predictions, logistic regression is the best and PCA based is the worst.

2- The best and worst use of information:

Considering the no skills predictor is the baseline model, and by the conclusions and the analysis we made in the previous question:

- Logistic regression demonstrates **the highest quality** of prediction by effectively leveraging the information in the dataset.
- But the PCA-based predictor shows limitations and **performs below the baseline** set by the random classifier.

4.2.3 Random confusion matrix:

The random confusion matrix is a special case where **the events are assumed to be independent**. It is a baseline metric that can be used to evaluate the extent to which the predictors make use of the information in the dataset.

Knowing that the independence of events implies the following equalities:

- $P(0 \cap +) = P(0)P(+)$
- $P(1 \cap +) = P(1)P(+)$
- $P(0 \cap -) = P(0)P(-)$
- $P(1 \cap -) = P(1)P(-)$

Therefore, the corresponding confusion matrix is: By following the same approach we calculated the random confusion matrix for each predictor and the results are shown in the figure below:

/	Pred -	Pred +
Real 0	P(0)P(-)	P(0)P(+)
Real 1	P(1)P(-)	P(1)P(+)

Table 4.3: Random Confusion matrix

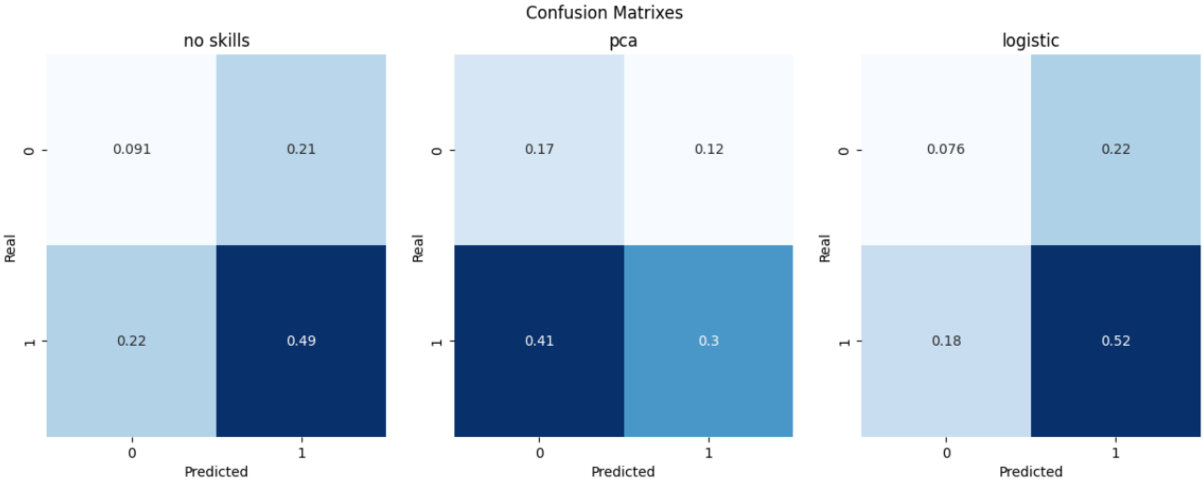


Figure 4.5: Random confusion matrices of the predictors

By comparing the random confusion matrix with the confusion matrix of the predictors, we conclude that the logistic regression makes the best use of the information in the dataset and the PCA based predictor makes the worst use for the following reasons are:

- The logistic regression uses both the relationship between the features Xi and the outcome y to make predictions where it maximizes the separation between classes by learning (using X_train and y_test correspondence) patterns that indicate if an observation is more likely to belong to one class or another.

- The PCA based prediction is based on reducing the number of features by focusing on variance, not class separation. And in our case, we selected the first component with the highest variance (instead of using the maximum separation).

In order to justify these results we will calculate the probability of having a correct prediction and the probability of having a false prediction. Thus, let's take a look of the percentages of correct $P(C)$ and incorrect $P(I)$ predictions of each predictor where:

- $P(1) = np.sum[y\_test == 1]/len(y\_test)$
- $P(0) = np.sum[y\_test == 0]/len(y\_test)$
- $P(+) = np.sum[y\_pred == 1]/len(y\_pred)$
- $P(-) = np.sum[y\_pred == 0]/len(y\_pred)$
- $FPR = P(0 \cap +) = P(0)P(+)$
- $TPR = P(1 \cap +) = P(1)P(+)$
- $TNR = P(0 \cap -) = P(0)P(-)$
- $FNR = P(1 \cap -) = P(1)P(-)$
- C is the event "Have a correct prediction"
- I is the event "Have an incorrect prediction"
- $P(C) = TPR + TNR$
- $P(I) = FPR + FNR$

So from the Random Confusion matrix we have:

/	% Correct	% Incorrect
No skills	57,8	42
PCA based	47	53
Logistic regression	59,6	40

Table 4.4: Percentage of correct and incorrect predictions made by each predictor.

So the logistic regression still the best predictor, based on the percentage of correct predictions, and the PCA is the worst.

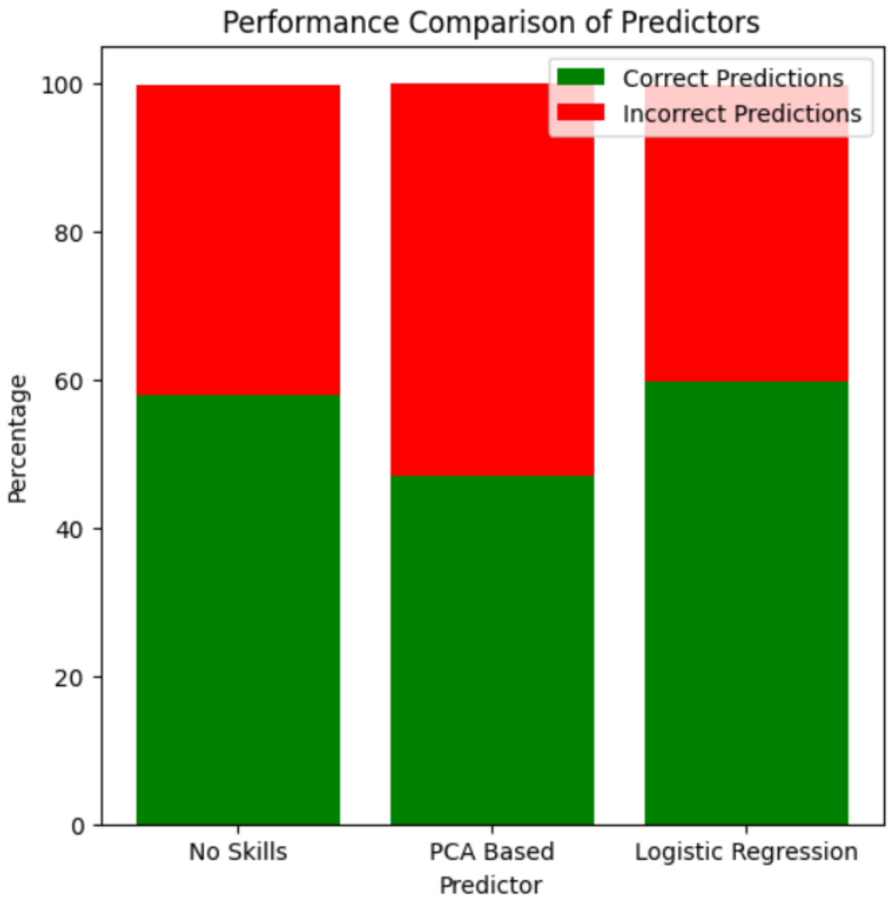


Figure 4.6: Performance comparison of the predictors using a random confusion matrix

Chapter 5

Quality estimators

To assess the models, several evaluation metrics were calculated, including Sensitivity (True Positive Rate), Specificity (True Negative Rate), Precision, Negative Predictive Value (NPV), Accuracy, and F1 Score. These metrics provide a comprehensive understanding of each model's ability to classify positive and negative samples accurately. Given the class imbalance in the dataset, we focused on metrics like F1 Score and Specificity to ensure the model's robustness in identifying the minority class.

5.1 closed-form expression for each of the quality metrics

We define a set of quality metrics to evaluate the performance of the predictors. These metrics are:

- Sensitivity (Recall): The proportion of actual positive cases that were correctly predicted.
- Specificity: The proportion of actual negative cases that were correctly predicted.
- Precision: The proportion of predicted positive cases that were correctly predicted.
- Negative Predictive Value (NPV): The proportion of predicted negative cases that were correctly predicted.
- Accuracy: The proportion of correct predictions.
- F1 Score: The harmonic mean of precision and sensitivity.

NOTE: The reason we use the harmonic mean as opposed to the regular mean, is that the harmonic mean punishes values that are further apart.

- $Sen = NbOfTruePositive/NbOfCases(1/.) = TP/(TP + FN)$
- $Spe = NbOfTrueNegative/NbOfCases(0/.) = TN/(TN + FP)$
- $Pre = NbOfTruePositive/NbOfCases(./+) = TP/(TP + FP)$
- $NPV = NbTrueNegative/NbOfCases(0/.) = TN/(TN + FP)$

- $Acc = (NbCases(0/-) + NbCases(1/+))/AllCases = (TN + TP)/(TN + TP + FN + FP)$

- $F1 = 2 * (Pre * Sen)/(Pre + Sen) = 2TP/(2TP + FN + FP)$

5.2 Computing the metrics and model evaluation

Before starting the comparison, we need to select the good metrics to use with our problem. Knowing that:

- Recall is used to focus on limiting the FP (1/-)
- Spec is used if the FP (0/+) are costly
- Pre to focus on limiting FP (0/+)
- NPV to be used if we want to focus on the accuracy of the negatives ie to ensure that the negatives are truly negatives
- Accuracy good for balanced datasets
- F1 score to focus on limiting both FN and FP and it's good for unbalanced datasets.
- Imbalanced Datasets: Focus on F1 Score, Precision, Recall, Specificity, and NPV. AUC-ROC is also useful here as it shows how well the model differentiates between classes.
- Balanced Datasets: Accuracy can be more reliable here, along with Precision, Recall, and F1 Score.

We will use as metrics:

- F1 score
- Spec
- NPV

The results found, we summarized them in this histogram and we deduce that: The best model is the logistic regression, the worst in the PCA based. However, in order to chose the model that has the best quality and use the metrics above in a more specific way we will plot the AUC-ROC curve.

Remark: why the AUC-ROC curve? *ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. The ROC curve is plotted with TPR against the FPR where TPR is on the y-axis and FPR is on the x-axis.* From the plot, the quality of Logistic regression is better than the quality of the No skills predictor. The PCA-based behaves almost like the No skills predictor but it's still worse than the no skills.

Besides, all the metrics of the logistic regression are higher than the metrics of the no skills predictor All the metrics of the PCA based are lower than the metrics of the no skills predictor So, the logistic regression makes the best use of the information and the PCA-based makes the worst one.

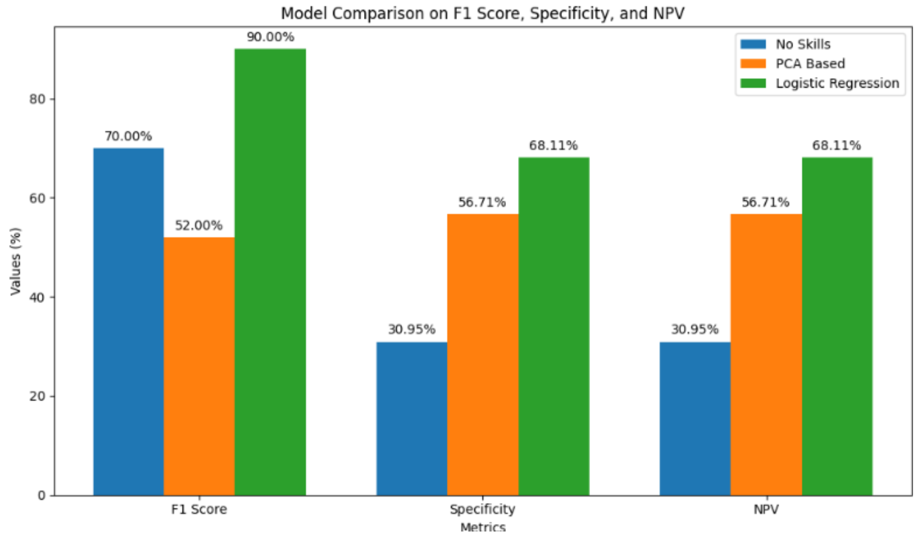


Figure 5.1: Model comparison on F1 Score, Specificity and NPV

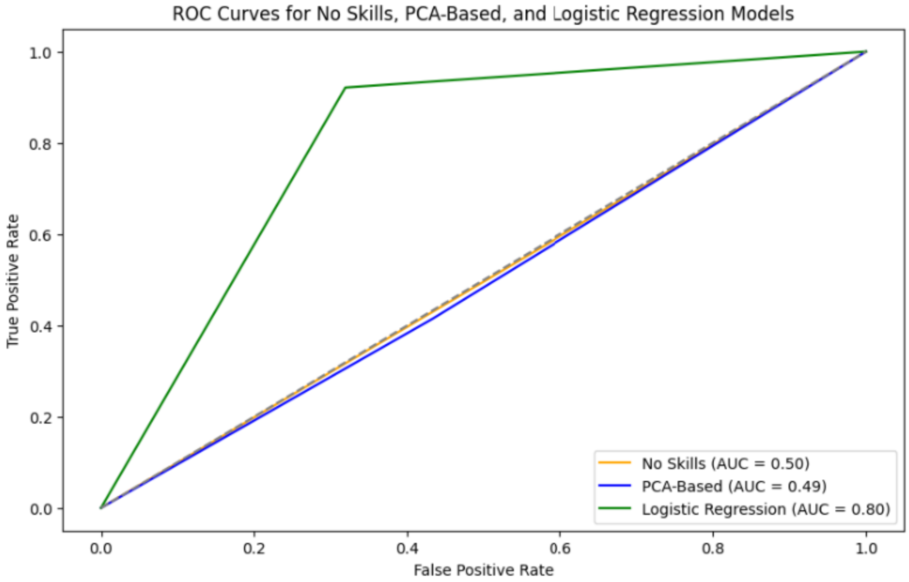


Figure 5.2: ROC Curves of the No skills, PCA and Logistic regression models.

Takeaway:

By training and evaluating these models, we observed the impact of feature engineering (through PCA) and supervised learning (through Logistic Regression) on classification performance. Logistic Regression outperformed the other models due to its ability to learn from labeled data, making it the most reliable model for this binary classification task. **Based on the evaluation metrics, the Logistic Regression model demonstrated superior performance, achieving the highest sensitivity, specificity, and F1 score. The PCA-based model showed moderate performance, with limited ability to separate the classes effectively. The no-skills model performed near the random baseline, as expected.**

By training and evaluating these models, we observed the impact of feature engineering (through PCA) and supervised learning (through Logistic Regression) on classification performance. Logistic Regression outperformed

the other models due to its ability to learn from labeled data, making it the most reliable model for this binary classification task.

Chapter 6

Quality as a function of the parameters

Chapter 7

Conclusion

Summarize the main findings of your report and suggest areas for future research or work.

Bibliography

[1] V7 Labs. ”Confusion Matrix Guide: Understand The Basics of Confusion Matrices in Machine Learning.” [Online]. Available: https://www.v7labs.com/blog/confusion-matrix-guide. Accessed: [Date you accessed the link].

[2] Towards Data Science. ”Guide to Confusion Matrices Classification Performance Metrics.” [Online]. Available: https://towardsdatascience.com/guide-to-confusion-matrices-classification-performance-metrics-a0ebfc08408e. Accessed: [Date you accessed the link].

[3] Towards Data Science. ”Understanding AUC-ROC Curve.” [Online]. Available: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5. Accessed: [Date you accessed the link].