W5. Dealing with Imbalanced Data

Guang Cheng

University of California, Los Angeles

guangcheng@ucla.edu

Week 5

Example: Credit Card Fraud Detection

- Let's look at an example first

Example: Credit Card Fraud Detection

- Let's look at an example first
- Dataset Characteristics:

Example: Credit Card Fraud Detection

- Let's look at an example first
- Dataset Characteristics:
  - Total Transactions: 1,000,000

Example: Credit Card Fraud Detection

- Let's look at an example first
- Dataset Characteristics:
  - Total Transactions: 1,000,000
  - Legitimate Transactions: 995,000 (99.5%)

Example: Credit Card Fraud Detection

- Let's look at an example first
- Dataset Characteristics:
    - Total Transactions: 1,000,000
    - Legitimate Transactions: 995,000 (99.5%)
    - Fraudulent Transactions: 5,000 (0.5%)

Example: Credit Card Fraud Detection

- Let's look at an example first
- Dataset Characteristics:
  - Total Transactions: 1,000,000
  - Legitimate Transactions: 995,000 (99.5%)
  - Fraudulent Transactions: 5,000 (0.5%)
- The goal is to develop a machine learning model that accurately identifies fraudulent transactions.

Example: Credit Card Fraud Detection

- **Challenges**:

Example: Credit Card Fraud Detection

- **Challenges**:
  - **Model Bias**: Most classification algorithms will favor the majority class, leading to poor identification of fraudulent transactions.

Example: Credit Card Fraud Detection

- **Challenges**:
  - **Model Bias**: Most classification algorithms will favor the majority class, leading to poor identification of fraudulent transactions.
  - **Evaluation Metrics**: Traditional accuracy is not an appropriate performance metric, as a model that predicts all transactions as legitimate would achieve a 99.5% accuracy rate, ignoring the critical fraudulent transactions.

Example: Credit Card Fraud Detection

- **Challenges**:
  - **Model Bias**: Most classification algorithms will favor the majority class, leading to poor identification of fraudulent transactions.
  - **Evaluation Metrics**: Traditional accuracy is not an appropriate performance metric, as a model that predicts all transactions as legitimate would achieve a 99.5% accuracy rate, ignoring the critical fraudulent transactions.
  - **Over-fitting Minority Class**: Attempts to focus on the minority class can lead the model to over-fit the fraudulent transactions, reducing its generalization ability.

Approaches to Handle Imbalanced Data Set Problem

- **Choose proper evaluation metric**

Approaches to Handle Imbalanced Data Set Problem

- **Choose proper evaluation metric**
- **Resampling methods**

Approaches to Handle Imbalanced Data Set Problem

- **Choose proper evaluation metric**
- **Resampling methods**
- **Synthetic Minority Oversampling Technique (SMOTE)**

Choose proper evaluation metric

- The accuracy of a classifier is the total number of correct predictions by the classifier divided by the total number of predictions. This may be good enough for a well-balanced class, but not ideal for the imbalanced class problem.

Choose proper evaluation metric

- The accuracy of a classifier is the total number of correct predictions by the classifier divided by the total number of predictions. This may be good enough for a well-balanced class, but not ideal for the imbalanced class problem.
- The other metrics such as precision, i.e.,

$$\frac{TP}{TP+FP}$$

is the measure of how accurate the classifier's prediction of the positive class, while recall, i.e.,

$$\frac{TP}{TP+FN}$$

is the measure of the classifier's ability to identify the positive class.

Choose proper evaluation metric

- Rather, for an imbalanced class dataset, F_1 score is a more appropriate metric. It is the harmonic mean of precision and recall:

$$F_1 = 2 * \frac{\text{precison} * \text{recall}}{\text{precison} + \text{recall}}$$

Choose proper evaluation metric

- Rather, for an imbalanced class dataset, F_1 score is a more appropriate metric. It is the harmonic mean of precision and recall:

$$F_1 = 2 * \frac{\text{precison} * \text{recall}}{\text{precison} + \text{recall}}$$

- If the classifier predicts the minority (negative) class wrongly, i.e., false-positive increases, the precision metric will be low and so as F_1 score.

Choose proper evaluation metric

- Rather, for an imbalanced class dataset, F_1 score is a more appropriate metric. It is the harmonic mean of precision and recall:

$$F_1 = 2 * \frac{\text{precison} * \text{recall}}{\text{precison} + \text{recall}}$$

- If the classifier predicts the minority (negative) class wrongly, i.e., false-positive increases, the precision metric will be low and so as F_1 score.
- On the other hand, if the classifier identifies the minority class poorly, i.e. more of majority class wrongfully predicted as the minority class, then false negatives will increase, so recall and F_1 score will low.

Choose proper evaluation metric

- Rather, for an imbalanced class dataset, F_1 score is a more appropriate metric. It is the harmonic mean of precision and recall:

$$F_1 = 2 * \frac{\text{precison} * \text{recall}}{\text{precison} + \text{recall}}$$

- If the classifier predicts the minority (negative) class wrongly, i.e., false-positive increases, the precision metric will be low and so as F_1 score.
- On the other hand, if the classifier identifies the minority class poorly, i.e. more of majority class wrongfully predicted as the minority class, then false negatives will increase, so recall and F_1 score will low.
- In summary, F_1 score only increases if both the number and quality of prediction improves.

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)
  - False Positives (FP): 10 (Incorrectly identified as having the disease)

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)
  - False Positives (FP): 10 (Incorrectly identified as having the disease)
  - False Negatives (FN): 10 (Incorrectly identified as not having the disease)

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)
  - False Positives (FP): 10 (Incorrectly identified as having the disease)
  - False Negatives (FN): 10 (Incorrectly identified as not having the disease)
  - True Negatives (TN): 960 (Correctly identified as not having the disease)

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)
  - False Positives (FP): 10 (Incorrectly identified as having the disease)
  - False Negatives (FN): 10 (Incorrectly identified as not having the disease)
  - True Negatives (TN): 960 (Correctly identified as not having the disease)
- Scenario 2: Model B

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)
  - False Positives (FP): 10 (Incorrectly identified as having the disease)
  - False Negatives (FN): 10 (Incorrectly identified as not having the disease)
  - True Negatives (TN): 960 (Correctly identified as not having the disease)
- Scenario 2: Model B
  - True Positives (TP): 15

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)
  - False Positives (FP): 10 (Incorrectly identified as having the disease)
  - False Negatives (FN): 10 (Incorrectly identified as not having the disease)
  - True Negatives (TN): 960 (Correctly identified as not having the disease)
- Scenario 2: Model B
  - True Positives (TP): 15
  - False Positives (FP): 5

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)
  - False Positives (FP): 10 (Incorrectly identified as having the disease)
  - False Negatives (FN): 10 (Incorrectly identified as not having the disease)
  - True Negatives (TN): 960 (Correctly identified as not having the disease)
- Scenario 2: Model B
  - True Positives (TP): 15
  - False Positives (FP): 5
  - False Negatives (FN): 15

An example to illustrate the choice of F1

Let's consider a concrete example involving a medical diagnosis system designed to identify a rare disease.

- Dataset Description:
  - Total Patients: 1000
  - Patients with Disease (Positive Class): 30
  - Patients without Disease (Negative Class): 970
- Scenario 1: Model A
  - True Positives (TP): 20 (Correctly identified as having the disease)
  - False Positives (FP): 10 (Incorrectly identified as having the disease)
  - False Negatives (FN): 10 (Incorrectly identified as not having the disease)
  - True Negatives (TN): 960 (Correctly identified as not having the disease)
- Scenario 2: Model B
  - True Positives (TP): 15
  - False Positives (FP): 5
  - False Negatives (FN): 15
  - False Negatives (FN): 965

An example to illustrate the choice of F1

- Model A Metrics
  - Precision (Model A) $=$?
  - Recall (Model A) $=$?
  - F1 Score (Model A) $=$?
- Model B Metrics
  - Precision (Model B) $=$?
  - Recall (Model B) $=$?
  - F1 Score (Model B) $=$?

An example to illustrate the choice of F1

- Model A Metrics

An example to illustrate the choice of F1

- Model A Metrics
  - Precision (Model A) $= 20/(20+10) = 0.67$

An example to illustrate the choice of F1

- Model A Metrics
  - Precision (Model A) $= 20/(20+10) = 0.67$
  - Recall (Model A) $= 20/(20+10) = 0.67$

An example to illustrate the choice of F1

- Model A Metrics
  - Precision (Model A) $= 20/(20+10) = 0.67$
  - Recall (Model A) $= 20/(20+10) = 0.67$
  - F1 Score (Model A) $= 2 * (0.67 * 0.67)/(0.67 + 0.67) = 0.67$

An example to illustrate the choice of F1

- Model A Metrics
  - Precision (Model A) $= 20/(20 + 10) = 0.67$
  - Recall (Model A) $= 20/(20 + 10) = 0.67$
  - F1 Score (Model A) $= 2 * (0.67 * 0.67)/(0.67 + 0.67) = 0.67$
- Model B Metrics

An example to illustrate the choice of F1

- Model A Metrics
  - Precision (Model A) $= 20/(20 + 10) = 0.67$
  - Recall (Model A) $= 20/(20 + 10) = 0.67$
  - F1 Score (Model A) $= 2 * (0.67 * 0.67)/(0.67 + 0.67) = 0.67$
- Model B Metrics
  - Precision (Model B) $= 15/(15 + 5) = 0.75$

An example to illustrate the choice of F1

- Model A Metrics
  - Precision (Model A) $= 20/(20 + 10) = 0.67$
  - Recall (Model A) $= 20/(20 + 10) = 0.67$
  - F1 Score (Model A) $= 2 * (0.67 * 0.67)/(0.67 + 0.67) = 0.67$
- Model B Metrics
  - Precision (Model B) $= 15/(15 + 5) = 0.75$
  - Recall (Model B) $= 15/(15 + 15) = 0.50$

An example to illustrate the choice of F1

- Model A Metrics
  - Precision (Model A) $= 20/(20 + 10) = 0.67$
  - Recall (Model A) $= 20/(20 + 10) = 0.67$
  - F1 Score (Model A) $= 2 * (0.67 * 0.67)/(0.67 + 0.67) = 0.67$
- Model B Metrics
  - Precision (Model B) $= 15/(15 + 5) = 0.75$
  - Recall (Model B) $= 15/(15 + 15) = 0.50$
  - F1 Score (Model B) $= 2 * (0.75 * 0.50)/(0.75 + 0.50) \approx 0.60$

An example to illustrate the choice of F1 – Analysis

- Precision focuses on the proportion of positive identifications that were actually correct. Model B has a higher precision than Model A, suggesting it is better at ensuring that when it predicts the disease, it is more likely correct.

An example to illustrate the choice of F1 – Analysis

- Precision focuses on the proportion of positive identifications that were actually correct. Model B has a higher precision than Model A, suggesting it is better at ensuring that when it predicts the disease, it is more likely correct.
- Recall emphasizes the proportion of actual positives that were identified correctly. Model A has a higher recall than Model B, indicating it is better at capturing as many actual cases of the disease as possible.

An example to illustrate the choice of F1 – Analysis

- Precision focuses on the proportion of positive identifications that were actually correct. Model B has a higher precision than Model A, suggesting it is better at ensuring that when it predicts the disease, it is more likely correct.
- Recall emphasizes the proportion of actual positives that were identified correctly. Model A has a higher recall than Model B, indicating it is better at capturing as many actual cases of the disease as possible.
- F1 Score provides a balance between precision and recall. It is particularly useful in imbalanced datasets because it considers both false positives and false negatives. In this scenario, despite Model B having a higher precision, its recall is significantly lower, resulting in a lower F1 score. Model A, despite having lower precision, has a balanced performance between precision and recall, leading to a higher F1 score.

An example to illustrate the choice of F1 – Conclusion

- The F1 score is a more appropriate metric than precision and recall individually in scenarios like this because it captures the balance between the two. This is crucial in imbalanced datasets where a model might have a high precision by predicting the majority class correctly but fails to capture the minority class effectively, or vice versa.

An example to illustrate the choice of F1 – Conclusion

- The F1 score is a more appropriate metric than precision and recall individually in scenarios like this because it captures the balance between the two. This is crucial in imbalanced datasets where a model might have a high precision by predicting the majority class correctly but fails to capture the minority class effectively, or vice versa.
- The F1 score helps to identify models that maintain a balance between identifying positive cases correctly and minimizing false positives, which is especially important in critical applications like medical diagnostics.

Resampling methods

- **Resampling methods** are a fundamental approach to addressing the challenge of imbalanced data in machine learning and statistics. Resampling methods aim to correct this imbalance by altering the dataset to have a more balanced class distribution.

Resampling methods

- **Resampling methods** are a fundamental approach to addressing the challenge of imbalanced data in machine learning and statistics. Resampling methods aim to correct this imbalance by altering the dataset to have a more balanced class distribution.
- Data centric AI vs Algorithm centric AI

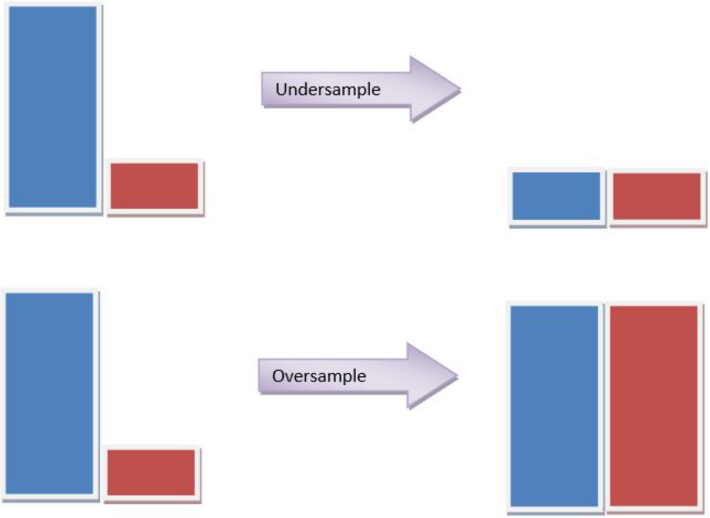
Types of Resampling methods

- **Oversampling the Minority Class**: It refers to the process of increasing the number of instances in the minority class(es) within a dataset to address the issue of class imbalance.

Types of Resampling methods

- **Oversampling the Minority Class**: It refers to the process of increasing the number of instances in the minority class(es) within a dataset to address the issue of class imbalance.
- **Undersampling the Majority Class**: It involves reducing the number of instances in the majority class(es) to achieve a more balanced class distribution in situations where one class significantly outnumbers the other(s).

Sktech map of oversampling and undersamping



Exercise: Manual Resampling Methods

- **Given Dataset**:

Exercise: Manual Resampling Methods

- **Given Dataset**:
  - Imagine you have a small dataset representing a binary classification problem, with two classes: Positive (P) and Negative (N). The dataset is imbalanced, consisting of 8 Negative instances and 2 Positive instances.

Exercise: Manual Resampling Methods

- **Given Dataset**:
  - Imagine you have a small dataset representing a binary classification problem, with two classes: Positive (P) and Negative (N). The dataset is imbalanced, consisting of 8 Negative instances and 2 Positive instances.
  - Negative instances: N1, N2, N3, N4, N5, N6, N7, N8

Exercise: Manual Resampling Methods

- **Given Dataset**:
  - Imagine you have a small dataset representing a binary classification problem, with two classes: Positive (P) and Negative (N). The dataset is imbalanced, consisting of 8 Negative instances and 2 Positive instances.
  - Negative instances: N1, N2, N3, N4, N5, N6, N7, N8
  - Positive instances: P1, P2

Exercise: Manual Resampling Methods

- **Given Dataset**:
  - Imagine you have a small dataset representing a binary classification problem, with two classes: Positive (P) and Negative (N). The dataset is imbalanced, consisting of 8 Negative instances and 2 Positive instances.
  - Negative instances: N1, N2, N3, N4, N5, N6, N7, N8
  - Positive instances: P1, P2
- **Task**: Your task is to manually apply undersampling and oversampling to make the dataset balanced.

Exercise: Manual Resampling Methods

- **Undersampling**

Exercise: Manual Resampling Methods

- **Undersampling**
  - Goal: Reduce the number of instances in the majority class (Negative) to match the minority class (Positive).

Exercise: Manual Resampling Methods

- **Undersampling**
  - Goal: Reduce the number of instances in the majority class (Negative) to match the minority class (Positive).
  - Instructions: Choose 2 instances randomly from the Negative class to keep. Your new dataset should have 2 Positive instances and 2 Negative instances.

Exercise: Manual Resampling Methods

- **Example Solution:**

Exercise: Manual Resampling Methods

- **Example Solution:**
  - Selected Negative instances: N3, N7

Exercise: Manual Resampling Methods

- **Example Solution:**
  - Selected Negative instances: N3, N7
  - New Dataset: N3, N7, P1, P2

- **Oversampling**

Exercise: Manual Resampling Methods

- **Oversampling**
  - Goal: Increase the number of instances in the minority class (Positive) to match the majority class (Negative).

Exercise: Manual Resampling Methods

- **Oversampling**
  - Goal: Increase the number of instances in the minority class (Positive) to match the majority class (Negative).
  - Instructions: Duplicate instances from the Positive class until you have as many Positive instances as Negative. Your new dataset should have 8 Positive instances and 8 Negative instances.

- **Example Solution:**

Exercise: Manual Resampling Methods

- **Example Solution:**
  - Duplicated Positive instances: P1, P1, P1, P1, P2, P2, P2, P2

Exercise: Manual Resampling Methods

- **Example Solution:**
  - Duplicated Positive instances: P1, P1, P1, P1, P2, P2, P2, P2
  - New Dataset: N1, N2, N3, N4, N5, N6, N7, N8, P1, P1, P1, P1, P2, P2, P2, P2

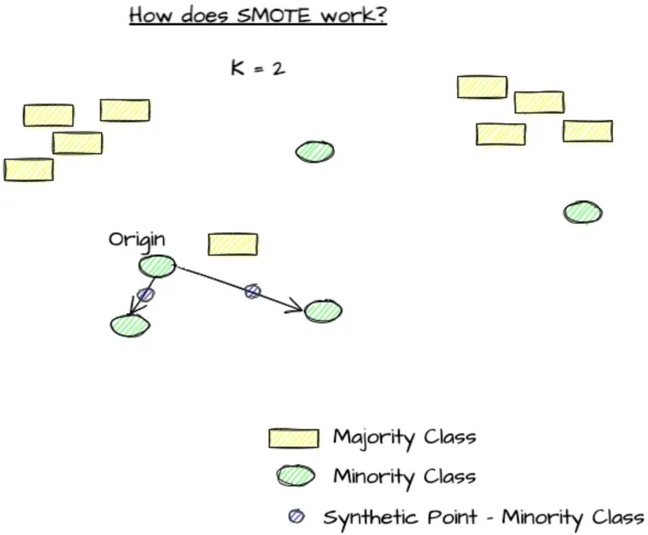
Synthetic Minority Over-sampling Technique (SMOTE)

- It is an advanced oversampling method that generates synthetic examples rather than duplicating existing ones. This approach helps in addressing the overfitting problem, which is common when simply duplicating minority class samples.

Synthetic Minority Over-sampling Technique (SMOTE)

- It is an advanced oversampling method that generates synthetic examples rather than duplicating existing ones. This approach helps in addressing the overfitting problem, which is common when simply duplicating minority class samples.
- A brief introduction to generative AI: GAN and Diffusion Models for unstructured data such as image; marginal based method for structured data such as tabules.

How does a basic SMOTE work ?



How does a basic SMOTE work ?

- Select a sample, let's call it O (for Origin), from the minority class randomly.

How does a basic SMOTE work ?

- Select a sample, let's call it O (for Origin), from the minority class randomly.
- Find the K-Nearest Neighbours of O that belong to the same class.

How does a basic SMOTE work ?

- Select a sample, let's call it O (for Origin), from the minority class randomly.
- Find the K-Nearest Neighbours of O that belong to the same class.
- Connect O to each of these neighbours using a straight line.

How does a basic SMOTE work ?

- Select a sample, let's call it O (for Origin), from the minority class randomly.
- Find the K-Nearest Neighbours of O that belong to the same class.
- Connect O to each of these neighbours using a straight line.
- Select a scaling factor 'z' in the range [0,1] randomly.

How does a basic SMOTE work ?

- Select a sample, let's call it O (for Origin), from the minority class randomly.
- Find the K-Nearest Neighbours of O that belong to the same class.
- Connect O to each of these neighbours using a straight line.
- Select a scaling factor 'z' in the range [0,1] randomly.
- For each new connection, place a new point on the line (z*100)% away from O. These will be our synthetic samples.

How does a basic SMOTE work ?

- Select a sample, let's call it O (for Origin), from the minority class randomly.
- Find the K-Nearest Neighbours of O that belong to the same class.
- Connect O to each of these neighbours using a straight line.
- Select a scaling factor 'z' in the range [0,1] randomly.
- For each new connection, place a new point on the line (z*100)% away from O. These will be our synthetic samples.
- Repeat this process until you get the desired number of synthetic samples.

What is the weakness of the traditional SMOTE ?

- If the point(s) selected in either steps 1 or 2 are located in a region dominated by majority class samples, the synthetic points might be generated inside the majority class region (which may make classification difficult!).

What is the weakness of the traditional SMOTE ?

- If the point(s) selected in either steps 1 or 2 are located in a region dominated by majority class samples, the synthetic points might be generated inside the majority class region (which may make classification difficult!).
- We need an improved version

Borderline SMOTE

- Borderline SMOTE works similarly to traditional SMOTE but with a few caveats. In order to overcome the shortcoming of SMOTE, it identifies two sets of points — Noise and Border.

Borderline SMOTE

- Borderline SMOTE works similarly to traditional SMOTE but with a few caveats. In order to overcome the shortcoming of SMOTE, it identifies two sets of points — Noise and Border.
  - A point is called "Noise" if all its nearest neighbours belong to a different class (i.e. the majority).

Borderline SMOTE

- Borderline SMOTE works similarly to traditional SMOTE but with a few caveats. In order to overcome the shortcoming of SMOTE, it identifies two sets of points — Noise and Border.
  - A point is called "Noise" if all its nearest neighbours belong to a different class (i.e. the majority).
  - On the other hand, "Border" points are those that have a mix of majority and minority class points as their nearest neighbours.

Borderline SMOTE

- Choose O as one of the Border points. Afterwards, when finding the nearest neighbours, the criteria of selecting only points belonging to the same class is relaxed to include points belonging to any class.

Borderline SMOTE

- Choose O as one of the Border points. Afterwards, when finding the nearest neighbours, the criteria of selecting only points belonging to the same class is relaxed to include points belonging to any class.
- This relaxation helps select points that are at risk of mis-classification and new points closer to the boundary. Everything else is the same.

Borderline SMOTE

- Choose O as one of the Border points. Afterwards, when finding the nearest neighbours, the criteria of selecting only points belonging to the same class is relaxed to include points belonging to any class.
- This relaxation helps select points that are at risk of mis-classification and new points closer to the boundary. Everything else is the same.
- But this solution is not a silver bullet. Restricting sampling from just border points and relaxing the neighbourhood selection criteria need not work in every scenario.

K-Means SMOTE

- This is a fairly recent method and aims to reduce the noisy synthetic points that other oversampling methods generate. The way it works is fairly straightforward:

K-Means SMOTE

- This is a fairly recent method and aims to reduce the noisy synthetic points that other oversampling methods generate. The way it works is fairly straightforward:
- Do K-Means Clustering on the data. Select clusters that have a high proportion ($>$50% or user-defined) of minority class samples.

K-Means SMOTE

- This is a fairly recent method and aims to reduce the noisy synthetic points that other oversampling methods generate. The way it works is fairly straightforward:
- Do K-Means Clustering on the data. Select clusters that have a high proportion ($>50\%$ or user-defined) of minority class samples.
- Apply conventional SMOTE to these selected clusters. Each cluster will be assigned new synthetic points. The number of these generated points will depend on the sparsity of the minority class in the cluster; the more the sparsity, the more new points.

K-means Clustering

- One type of unsupervised learning algorithm, i.e., no label y.

K-means Clustering

- One type of unsupervised learning algorithm, i.e., no label y.
- Clustering is to group similar data points into k clusters by minimizing the variance within each cluster and maximizing the variance between clusters. It has the following 3 steps.

K-means Clustering

- One type of unsupervised learning algorithm, i.e., no label y.
- Clustering is to group similar data points into k clusters by minimizing the variance within each cluster and maximizing the variance between clusters. It has the following 3 steps.
- Initialization: Starts with a random selection of k centroids, then iteratively adjusts them to minimize the total sum of distances within clusters.

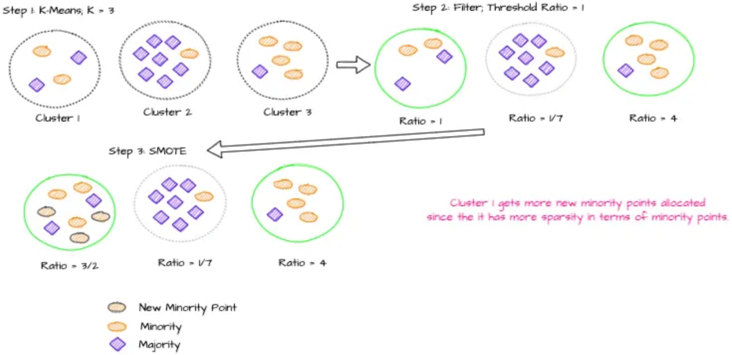
K-means Clustering

- One type of unsupervised learning algorithm, i.e., no label y.
- Clustering is to group similar data points into k clusters by minimizing the variance within each cluster and maximizing the variance between clusters. It has the following 3 steps.
- Initialization: Starts with a random selection of k centroids, then iteratively adjusts them to minimize the total sum of distances within clusters.
- Optimization: Iterates by re-assigning data points to the nearest centroid and recalculating centroids until centroids stabilize or a set number of iterations is reached.

K-means Clustering

- One type of unsupervised learning algorithm, i.e., no label y.
- Clustering is to group similar data points into k clusters by minimizing the variance within each cluster and maximizing the variance between clusters. It has the following 3 steps.
- Initialization: Starts with a random selection of k centroids, then iteratively adjusts them to minimize the total sum of distances within clusters.
- Optimization: Iterates by re-assigning data points to the nearest centroid and recalculating centroids until centroids stabilize or a set number of iterations is reached.
- Evaluation: Uses metrics like the silhouette score or the elbow method to determine the optimal number of clusters.

K-Means SMOTE



K-Means SMOTE

K-Means SMOTE

- Assign new minority samples to cluster 1 not cluster 3 due to a trade-off: the distribution of minority in a cluster should not be too dense (becomes majority, cluster 3) nor too sparse (not representative to the minority distribution, cluster 2).

K-Means SMOTE

- Assign new minority samples to cluster 1 not cluster 3 due to a trade-off: the distribution of minority in a cluster should not be too dense (becomes majority, cluster 3) nor too sparse (not representative to the minority distribution, cluster 2).
- In essence, this method helps create clusters of the minority class (that are not greatly influenced by other classes). This can ultimately benefit the ML model. However, it inherits the weaknesses of the K-Means algorithm — such as finding the right K, among others.

An example of SMOTE

- Suppose we had an imbalanced dataset. In other words, there are a lot more rows of a given class than the other.

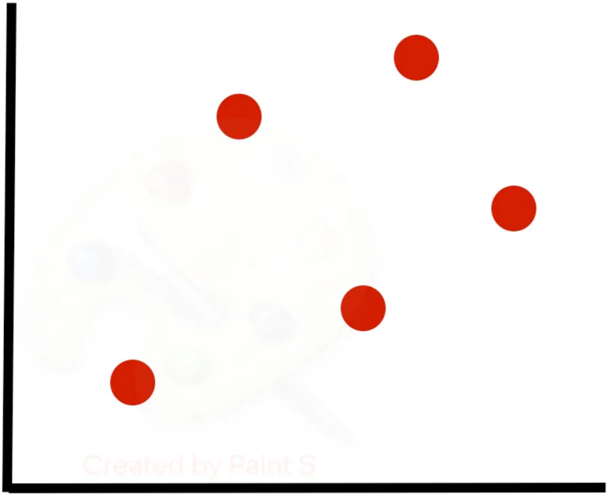
An example of SMOTE

- Suppose we had an imbalanced dataset. In other words, there are a lot more rows of a given class than the other.
- We proceed to plot a subset of the rows belonging to the minority class.

An example of SMOTE

- Suppose we had an imbalanced dataset. In other words, there are a lot more rows of a given class than the other.
- We proceed to plot a subset of the rows belonging to the minority class.
- We're only considering 2 of the many features (one corresponds to X-axis; another to Y-axis).

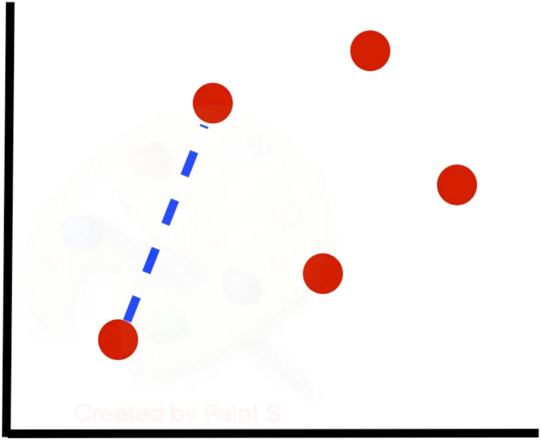
An example of SMOTE



An example of SMOTE

- We consider the first row (or a random row), and compute its k nearest neighbors. We then select a random nearest neighbor out of the k nearest neighbors.

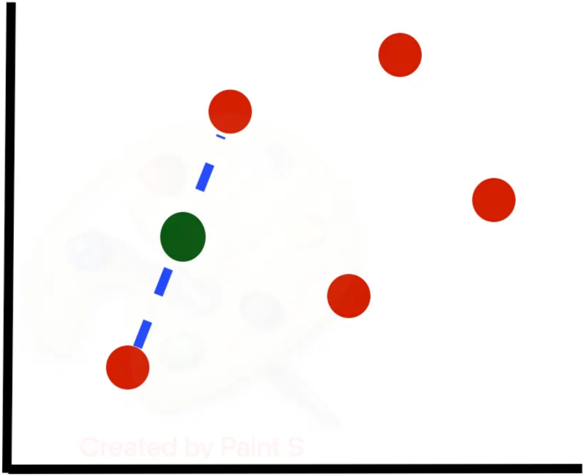
An example of SMOTE



An example of SMOTE

- We compute the difference between the two points and multiply it by a random number between 0 and 1. This gives us a synthetic example along the line between the two points.

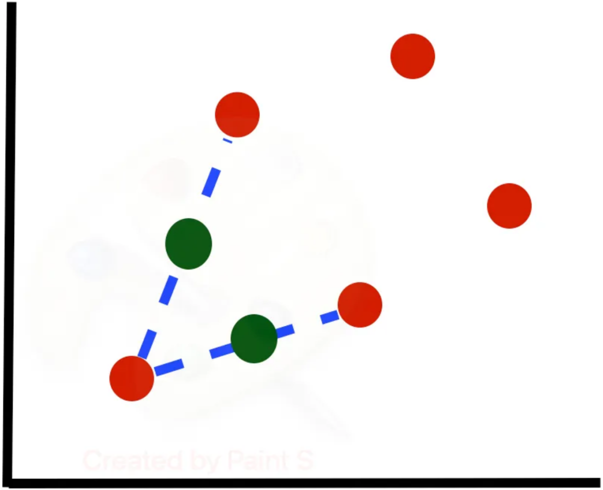
An example of SMOTE



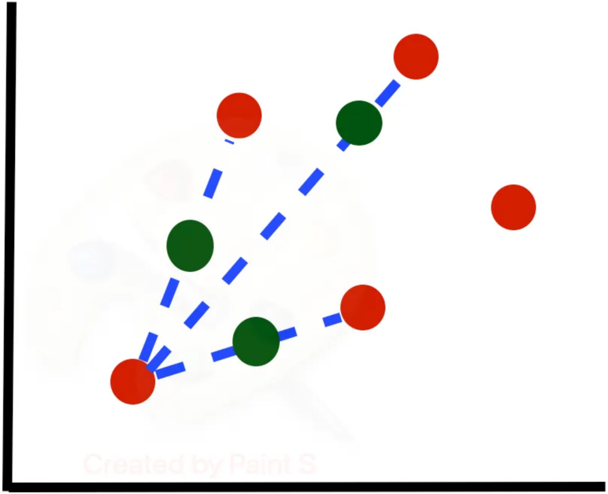
An example of SMOTE

- We repeat the process L times. Here, L represents the over-sampling rate. For example, the original minority class has 100 samples, while we aim to oversample to achieve 300 samples. Then, $L = 3$.
- In this case, only $300/100 = 3$ neighbors from the $k = 5$ nearest neighbors are chosen and one sample is generated in the direction of each.

An example of SMOTE



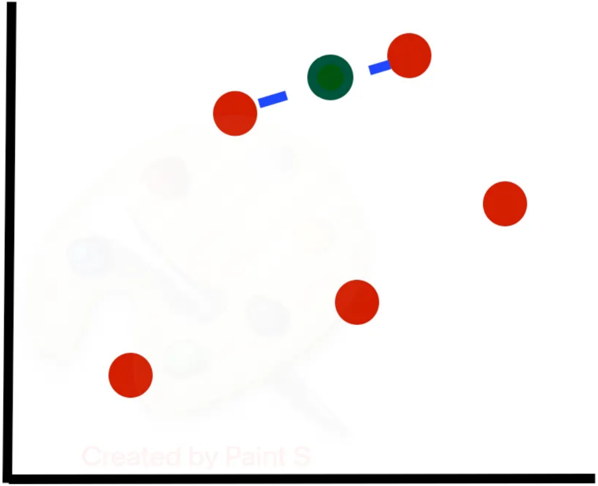
An example of SMOTE



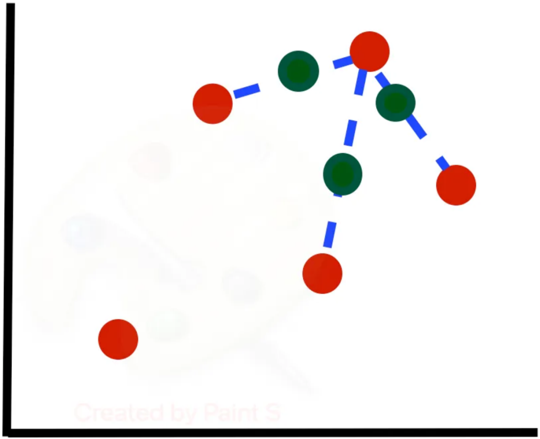
An example of SMOTE

- We then move on to the next row, compute its k nearest neighbors and select $300/100 = 3$ of its nearest neighbors at random to use in generating new synthetic examples.

An example of SMOTE



An example of SMOTE



SMOTE in Python

- Let's walk through an example of using SMOTE in Python.
- We begin by importing the required libraries.

SMOTE in Python

```
from random import randrange, uniform
from sklearn.neighbors import NearestNeighbors
import numpy as np
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, recall_score
```

SMOTE in Python

- In this example, we will make use of the Credit Card Fraud Detection dataset on Kaggle to train a model to determine whether a given transaction is fraudulent. We read the CSV file and store its contents in a Pandas DataFrame as follows:

SMOTE in Python

```
df = pd.read_csv("creditcard.csv")
```

- Unfortunately, due to confidentiality issues, they cannot provide the original features. Features $V_1, V_2, \cdots, V_{28}$ are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'.

What is PCA?

- Principal Component Analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.
- Will give details in later lecture.

SMOTE in Python

```
df.head(5)
```

...	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	149.62	0
...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	2.69	0
...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	378.66	0
...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	123.50	0
...	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	69.99	0

SMOTE in Python

```
df['Class'].value_counts()
Out:
0    284315
1       492
Name: Class, dtype: int64
```

- As we can see, there are significantly more negative samples (class 0) than positive samples (class 1).

SMOTE in Python

- For simplicity, we remove the time dimension.

```
df = df.drop(['Time'], axis=1)
```

- We split the dataset into features and labels.

```
X = df.drop(['Class'], axis=1)
y = df['Class']
```

- In order to evaluate the performance of our model, we split the data into training and test sets.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

SMOTE in Python

- Next, we initialize an instance of the RandomForestClassifier class.

```
rf = RandomForestClassifier(random_state=42)
```

- We fit our model to the training set.

```
rf.fit(X_train, y_train)
```

- Finally, we use our model to predict whether a transaction is fraudulent given what it has learnt.

```
y_pred = rf.predict(X_test)
```

SMOTE in Python

- Without implementing SMOTE, below is the confusion matrix that evaluates the model's performance. As we can see, our model classified 23 samples as non-fraudulent when, in fact, they were.

```
confusion_matrix(y_test, y_pred)
Out:
array([[56862,     2],
       [   23,    75]])
```

SMOTE using library

- The Python implementation of SMOTE actually comes in its own library (outside Scikit-Learn) which can be installed as follows:

```
pip install imbalanced-learn
```

- We can then import the SMOTE class.

```
from imblearn.over_sampling import SMOTE
```

- To avoid confusion, we read the csv file again.

```
df = pd.read_csv("creditcard.csv")
df = df.drop(['Time'], axis=1)
X = df.drop(['Class'], axis=1)
y = df['Class']
```

- We instantiate an instance of the SMOTE class. It's worth noting that, by default, it will ensure that there are an equal number of positive samples as negative samples.

```
sm = SMOTE(random_state=42, k_neighbors=5)
```

- We apply the SMOTE algorithm to the dataset as follows:

```
X_res, y_res = sm.fit_resample(X, y)
```

SMOTE using library

- Again, we split the dataset, train the model and predict whether the samples in the testing dataset should be considered fraudulent or not.

```
X_train, X_test, y_train, y_test = train_test_split(X_res, y_res, test_size=0.2, random_state
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)
```

- If we look at the confusion matrix, we can see that there are an equal number of positive samples as negative samples and the model didn't have any false negatives. The recall is 1.

```
confusion_matrix(y_test, y_pred)
Out:
array([[56737,    13],
       [    0, 56976]])
recall_score(y_test, y_pred)
Out: 1.0
```

Empirical comparison of different approaches to deal with imbalanced data

- Empirically compare undersampling, oversampling, and SMOTE methods for dealing with imbalanced data

Empirical comparison of different approaches to deal with imbalanced data

- Empirically compare undersampling, oversampling, and SMOTE methods for dealing with imbalanced data
- Wongvorachan, T., He, S., and Bulut, O. A comparison of undersampling, oversampling, and smote methods for dealing with imbalanced classification in educational data mining. Information, 14(1):54, 2023.

Empirical comparison of different approaches to deal with imbalanced data

Procedure to follow

- **Step 1: Prepare Your Dataset**

Empirical comparison of different approaches to deal with imbalanced data

Procedure to follow

- **Step 1: Prepare Your Dataset**
  - Start with a dataset that has an imbalanced class distribution. For instance, in educational data mining, this could involve predicting student dropouts where the number of dropouts (minority class) is much smaller than the number of students who continue (majority class).

Empirical comparison of different approaches to deal with imbalanced data

Procedure to follow

- **Step 1: Prepare Your Dataset**
  - Start with a dataset that has an imbalanced class distribution. For instance, in educational data mining, this could involve predicting student dropouts where the number of dropouts (minority class) is much smaller than the number of students who continue (majority class).
- **Step 2: Apply Resampling Methods**

Empirical comparison of different approaches to deal with imbalanced data

Procedure to follow

- **Step 1: Prepare Your Dataset**
  - Start with a dataset that has an imbalanced class distribution. For instance, in educational data mining, this could involve predicting student dropouts where the number of dropouts (minority class) is much smaller than the number of students who continue (majority class).
- **Step 2: Apply Resampling Methods**
  - Undersampling: Randomly remove samples from the majority class to match the number of samples in the minority class.

Empirical comparison of different approaches to deal with imbalanced data

Procedure to follow

- **Step 1: Prepare Your Dataset**
  - Start with a dataset that has an imbalanced class distribution. For instance, in educational data mining, this could involve predicting student dropouts where the number of dropouts (minority class) is much smaller than the number of students who continue (majority class).
- **Step 2: Apply Resampling Methods**
  - Undersampling: Randomly remove samples from the majority class to match the number of samples in the minority class.
  - Oversampling: Randomly duplicate samples in the minority class to match the number of samples in the majority class.

Empirical comparison of different approaches to deal with imbalanced data

Procedure to follow

- **Step 1: Prepare Your Dataset**
  - Start with a dataset that has an imbalanced class distribution. For instance, in educational data mining, this could involve predicting student dropouts where the number of dropouts (minority class) is much smaller than the number of students who continue (majority class).
- **Step 2: Apply Resampling Methods**
  - Undersampling: Randomly remove samples from the majority class to match the number of samples in the minority class.
  - Oversampling: Randomly duplicate samples in the minority class to match the number of samples in the majority class.
  - SMOTE: Generate synthetic samples for the minority class by interpolating between existing minority samples.

Empirical comparison of different approaches to deal with imbalanced data

- **Step 3: Train a Classifier**

Empirical comparison of different approaches to deal with imbalanced data

- **Step 3: Train a Classifier**
  - Use a common classifier for all three datasets. Logistic regression, decision trees, or random forests are popular choices because of their interpretability and ease of use.

Empirical comparison of different approaches to deal with imbalanced data

- **Step 3: Train a Classifier**
  - Use a common classifier for all three datasets. Logistic regression, decision trees, or random forests are popular choices because of their interpretability and ease of use.
- **Step 4: Evaluate Model Performance**

Empirical comparison of different approaches to deal with imbalanced data

- **Step 3: Train a Classifier**
  - Use a common classifier for all three datasets. Logistic regression, decision trees, or random forests are popular choices because of their interpretability and ease of use.
- **Step 4: Evaluate Model Performance**
  - Precision measures the accuracy of positive predictions.

Empirical comparison of different approaches to deal with imbalanced data

- **Step 3: Train a Classifier**
  - Use a common classifier for all three datasets. Logistic regression, decision trees, or random forests are popular choices because of their interpretability and ease of use.
- **Step 4: Evaluate Model Performance**
  - Precision measures the accuracy of positive predictions.
  - Recall measures the ability of the classifier to find all the positive samples.

Empirical comparison of different approaches to deal with imbalanced data

- **Step 3: Train a Classifier**
  - Use a common classifier for all three datasets. Logistic regression, decision trees, or random forests are popular choices because of their interpretability and ease of use.
- **Step 4: Evaluate Model Performance**
  - Precision measures the accuracy of positive predictions.
  - Recall measures the ability of the classifier to find all the positive samples.
  - F1-score provides a balance between precision and recall.

Empirical comparison of different approaches to deal with imbalanced data

- **Step 3: Train a Classifier**
  - Use a common classifier for all three datasets. Logistic regression, decision trees, or random forests are popular choices because of their interpretability and ease of use.
- **Step 4: Evaluate Model Performance**
  - Precision measures the accuracy of positive predictions.
  - Recall measures the ability of the classifier to find all the positive samples.
  - F1-score provides a balance between precision and recall.
  - AUC represents the likelihood of your model distinguishing between classes.

The results in Educational Data Mining [Wongvoracha, et al 2023]

Table 4. Classification results for each resampling condition.

	Performance Metrics	Baseline Mean (SD)	ROS Mean (SD)	RUS Mean (SD)	SMOTE-NC + RUS Mean (SD)
Moderately Imbalanced	Accuracy	0.736 (0.009)	0.877 (0.006)	0.705 (0.014)	0.779 (0.011)
	Precision	0.773 (0.006)	0.926 (0.006)	0.724 (0.014)	0.796 (0.013)
	Recall	0.888 (0.007)	0.819 (0.010)	0.662 (0.024)	0.749 (0.018)
	ROC-AUC	0.763 (0.013)	0.968 (0.003)	0.763 (0.015)	0.868 (0.010)
	F1	0.827 (0.005)	0.870 (0.006)	0.692 (0.017)	0.772 (0.013)
Extremely Imbalanced	Accuracy	0.887 (0.004)	0.984 (0.002)	0.731 (0.021)	0.905 (0.006)
	Precision	0.665 (0.058)	0.971 (0.003)	0.736 (0.024)	0.911 (0.008)
	Recall	0.193 (0.028)	0.999 (0.001)	0.721 (0.031)	0.898 (0.008)
	ROC-AUC	0.801 (0.016)	0.999 (0.0003)	0.800 (0.020)	0.967 (0.002)
	F1	0.299 (0.038)	0.985 (0.002)	0.728 (0.022)	0.904 (0.006)

The Implications in Educational Data Mining

- Undersampling can effectively balance the classes but might lead to a loss of valuable information, potentially reducing model performance on unseen data.

The Implications in Educational Data Mining

- Undersampling can effectively balance the classes but might lead to a loss of valuable information, potentially reducing model performance on unseen data.
- Oversampling can improve the model's ability to detect the minority class but might increase the risk of overfitting because of the duplication of minority class samples.

The Implications in Educational Data Mining

- Undersampling can effectively balance the classes but might lead to a loss of valuable information, potentially reducing model performance on unseen data.
- Oversampling can improve the model's ability to detect the minority class but might increase the risk of overfitting because of the duplication of minority class samples.
- SMOTE tends to offer a good balance by generating synthetic samples, potentially leading to improved model generalization without losing information or overly increasing the risk of overfitting.