Week 3- Bias Mitigation and Dimensionality Reduction

Dr. Smitha K G
Senior Lecturer,
College of Computing and Data Science,
Nanyang technological University

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

SMOTE for Bias Mitigation

Synthetic Minority Over-sampling Technique (SMOTE)

- In many real-world datasets: Fraud detection, Disease diagnosis, Loan default, Rare event prediction, the minority class is the class we care about most, yet it is under-represented.
- When data is imbalanced:
  - Models optimize overall accuracy
  - Majority class dominates loss functions
  - Minority class errors are treated as “cheap”
  - Model learns a structural bias toward the majority
- This is “Data Bias” and not “ Algorithm bias”

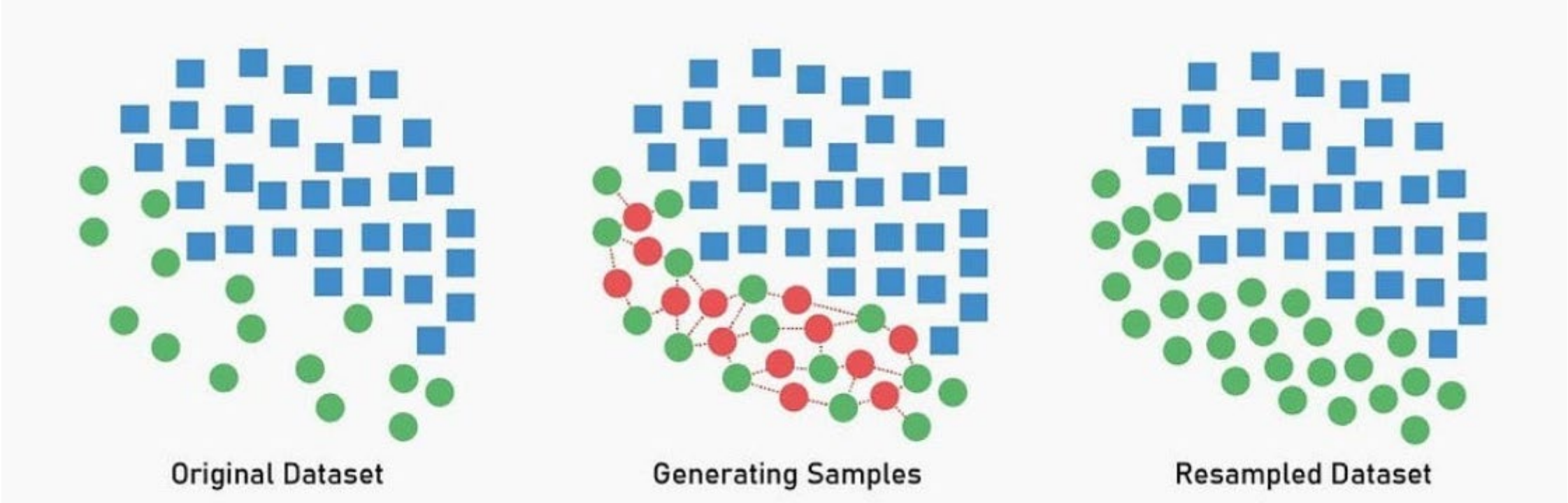
Key Challenges of Imbalanced Data:

- **Biased Models:** Algorithms prioritize the majority class, leading to poor performance on the minority class.
- **Misleading Metrics:** Accuracy can be deceptive. A model predicting everything as the majority class might still show 95% accuracy on a 95:5 imbalanced dataset.
- **Difficulty in Learning Patterns:** The minority class has fewer examples, making it harder for the model to learn its underlying patterns.
- Instead of accuracy, it's crucial to use metrics like **Precision, Recall, F1-score, and ROC-AUC** when evaluating models on imbalanced datasets. For a deeper dive into these metrics, check out our guide on Evaluating Classification Models.

How does SMOTE work?

- It selects a minority class instance (let's call it '*x_i*').
- It finds its "*k*" nearest neighbors within the same minority class.
- It then randomly picks one of these neighbors (let's say '*x_ j*').
- A new synthetic instance is created by taking the difference between '*x_i*' and '*x_ j*', multiplying it by a random number between 0 and 1, and adding it to '*x_i*'. This effectively creates a new instance along the line segment connecting *'x_i*' and '*x_ j*'.
- This process creates new, but similar, minority class samples, enriching the dataset without simply copying existing ones. This helps the model learn more robust decision boundaries.

How does SMOTE work? Visualization



How does SMOTE work? Details…

- SMOTE mitigates representation bias by creating synthetic minority samples instead of duplicating existing ones.
- The core idea is instead of random oversampling (duplicate points → overfitting), SMOTE Interpolates new samples between minority neighbors

Bias Type	Before SMOTE	After SMOTE
Representation Bias	Minority under-represented	Balanced exposure
Decision Boundary Bias	Skewed toward majority	Fairer boundary
Error Asymmetry	High FN on minority	Improved recall
Algorithmic Blindness	Minority ignored	Minority learned

Confusion matrix before SMOTE

	Predicted SAFE (Blue)	Predicted RISKY (Red)
Actual SAFE (Blue)	**1413**	**4**
Actual RISKY (Red)	**66**	**17**

Total=1413+4+66+17=1500

Accuracy=1413+17/1500=95.3

Confusion matrix after SMOTE

	Predicted SAFE (blue)	Predicted RISKY (red)
Actual SAFE (blue)	**1205**	**212**
Actual RISKY(red)	**14**	**69**

Total=1205+212+14+69=1500

Accuracy=1205+69/1500=85

Conscious bias correction is done by bank as missing a risky person (FN) → very costly compared to flagging a safe person (FP) → less costly

Classification data before SMOTE

	precision	recall	f1-score	support
0	0.96	1	0.98	1417
1	0.81	0.2	0.33	83

Total=1413+4+66+17=1500
Accuracy=1413+17/1500=95.3

Classification data after SMOTE

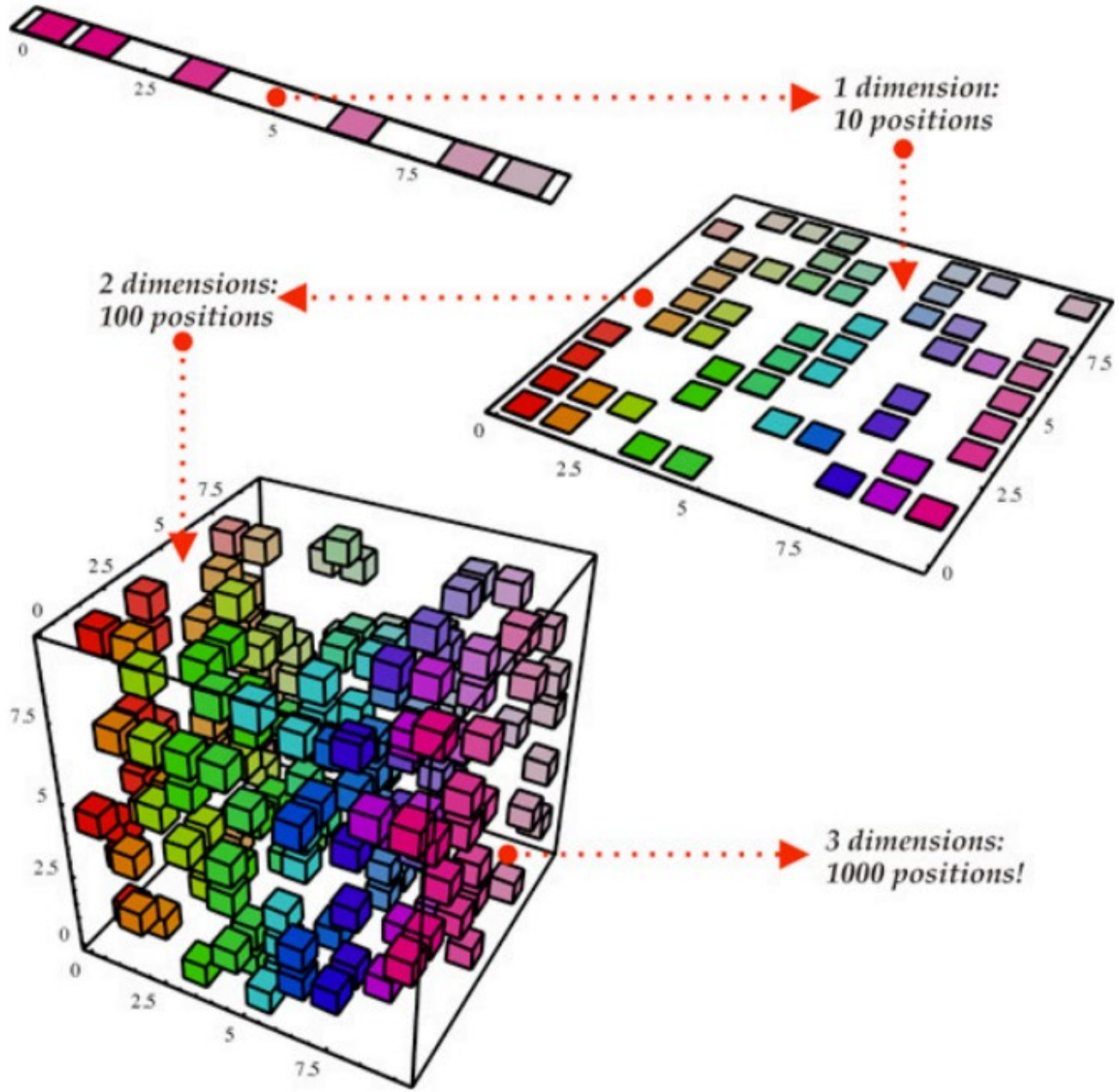
	precision	recall	f1-score	support
0	0.99	0.85	0.91	1417
1	0.25	0.83	0.38	83

Total=1205+212+14+69=1500
Accuracy=1205+69/1500=85

SMOTE improves fairness and safety by increasing minority recall at the cost of accuracy and precision, and this trade-off is both expected and necessary in imbalanced problems.

Dimensionality Reduction

- Dimensionality Reduction is the process of **reducing the number of input features while preserving as much useful information as possible.**
- It transforms high-dimensional data (many features) into a lower-dimensional space, making it: Easier to visualize, Better for model performance, Less prone to overfitting, Faster to train, Less noisy (removes redundant information), handles multicollinearity (High correlation between variables causes instability — reduction fixes this.)
- Feature Selection: **Select a subset of original features, by removing** unnecessary features and keeping existing features, e.g.: Filter methods (correlation thresholding, chi-square), Embedded methods (Lasso, Tree importance)
- Feature extraction: Transforms the original features into fewer, more meaningful variables. E.g: PCA, SVD, t-SNE, Autoencode


1 dimension:
10 positions
2 dimensions:
100 positions
3 dimensions:
1000 positions!

Technique	Category	Goal & Theoretical Basis	Use Case & Strengths	Limitations & Cons
Principal Component Analysis (PCA)	Linear, Unsupervised	Finds the directions (**Principal Components**) that capture the **maximum variance** in the data.	**Strengths:** Excellent for **data compression** and **noise reduction**. Simple, fast, and highly effective for linear data. **Used for:** Feature extraction and data visualization (2D/3D).	Assumes data has a **linear structure**. Components are **less interpretable** (combinations of original features). Sensitive to feature scaling.
Singular Value Decomposition (SVD)	Linear, Unsupervised	A matrix factorization technique that decomposes the data matrix into three constituent matrices. The reduced dimensions are derived from the largest singular values.	**Strengths:** The mathematical foundation for PCA (PCA is often computed using SVD). Highly efficient and used in applications like **Recommender Systems** and image compression. **Used for:** Latent factor modeling.	Similar limitations to PCA: focuses on **linear relationships**. Output components lack clear interpretability.
t-distributed Stochastic Neighbor Embedding (t-SNE)	Non-linear, Unsupervised	Focuses on preserving **local data structure** (distances between nearby points) by minimizing the difference between high-dimensional and low-dimensional probability distributions.	**Strengths:** Produces **highly insightful visualizations** by effectively separating distinct clusters, even for very complex, **non-linear data**. **Used for: Exploratory Data Analysis (EDA)**.	**Computationally expensive** and slow for large datasets. **Not for production use**; results can be sensitive to hyperparameters.
Autoencoder	Non-linear, Unsupervised (Neural Network)	Uses a neural network architecture (**Encoder-Decoder**) to learn a compressed, low-dimensional representation (**Bottleneck Layer**) of the input data, then reconstructs the original input from this representation.	**Strengths:** Learns **non-linear, highly complex feature relationships** automatically. Can be used for **denoising** and generative modeling. **Used for:** Feature learning and reconstruction tasks.	**Requires a lot of data** to train the network. **Computationally expensive** and requires careful tuning of the network architecture. Output is **difficult to interpret**.

Principal Component Analysis

- PCA is a powerful dimensionality reduction technique that can significantly improve the performance of machine learning models.
- By reducing the number of input features, PCA can accelerate model training and enhance data visualization capabilities.
- If your model is struggling with slow fitting times, consider applying PCA to reduce the dimensionality of your data.

Aspect	Advantages of PCA	Disadvantages of PCA
Dimensionality Reduction	Reduces a large number of features into a smaller set of components, making data easier to handle and store.	Important information may be lost if too few components are retained.
Computational Efficiency	Fewer dimensions often lead to faster training and prediction for many models, especially distance-based ones.	PCA itself adds computational cost, which may outweigh benefits for small or low-dimensional datasets.
Overfitting Control	By removing noise and redundant features, PCA can reduce overfitting and improve generalization.	If discriminative but low-variance features are removed, model performance can degrade.
Data Compression	Provides an efficient way to compress data while retaining most of the variance.	Compression is lossy; original data cannot be perfectly reconstructed unless all components are kept.
Applicability	Excellent for exploratory analysis, preprocessing, and compression.	Not suitable when individual features carry direct semantic or regulatory meaning (e.g., medical tests, financial indicators).

Dimensionality reduction is powerful — but **not harmless**.

Consider we had features(“income”, “age”, “blood pressure”)

- Before reduction: each feature has a clear meaning
- After reduction: features become mixtures (“0.3×income + 0.5×age − 0.2×education…”)
- We can no longer say:“This decision was made because income was high.”
- This is dangerous when: explanations are required and decisions affect people (health, finance, law)
- PCA with very few components does **not fail** — it does exactly what it’s asked to do: keep only the strongest global patterns and discard detail.

Why we use dimensionality reduction?

- Makes models faster
- Reduces overfitting
- Removes redundancy
- Helps learning focus on the “big picture”

Need to be careful

- Loses human interpretability
- May remove important signals
- Not universal — context matters

Dimensionality reduction simplifies data so models can learn faster and more reliably, but it must be used carefully because simplification always comes with information loss.