

Machine Learning Systems Design

Data Lifecycle

Lecture 7: Data Preparation



CE 40959 Spring 2023

Ali Zarezade

[SharifMLSD.github.io](https://github.com/SharifMLSD)

Agenda

1. Questions About the Data
2. Data Quality
3. Data Sampling
4. Data Labeling
- 5. Data Partitioning**
- 6. Data Leakage**
- 7. Data Imbalance**
- 8. Data Augmentation**

5. Data Partitioning

Data partitioning



Data partitioning

- **Training set** — Which you run your learning algorithm on.
- **Dev (development) / validation set** — Which you use to tune parameters, select features, and make other decisions regarding the learning algorithm. Sometimes also called the hold-out cross validation set.
- **Test set** — which you use to evaluate the performance of the algorithm, but not to make any decisions regarding what learning algorithm or parameters to use.

Good partitioning conditions

- **Validation and test sets follow the same distribution**
- Split was applied to raw data.
- Data was randomized before the split
- Leakage during the split was avoided
- Be careful when working with **time series** or time dependent data

Ideal split ratio

What is the ideal ratio, 70%/15%/15% or 80%/10%/10% or ...99%/0.5%/0.5%?

Ideal split ratio

- It **depends** on the **size of the dataset**.
- The validation and test data are only used to calculate statistics reflecting the performance of the model. They just need to be large enough to provide **reliable statistics**.

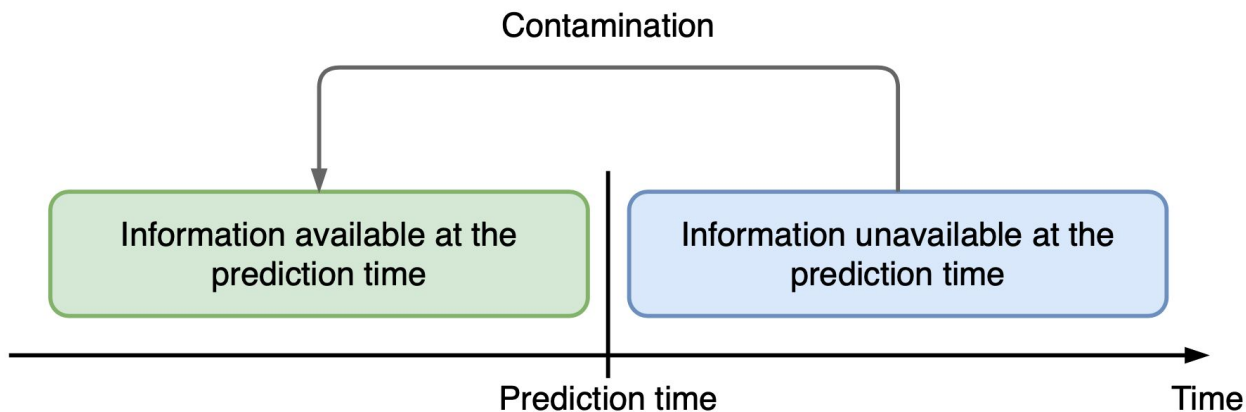
When to change dev/test sets

- The actual distribution you need to do well on is different from the dev/test sets.
- You have overfit to the dev set
- If you change test set, how to make sure old results are comparable with new results

6. Data Leakage

Data leakage

- Some form of the label “leaks” into the features
- This same information is not available during inference



Data leakage: example 1


- Problem: detect lung cancer from CT scans
- Data: collected from hospital A
- Performs well on test data from hospital A
- Performs poorly on test data from hospital B

Patient ID	Date	Doctor note	Medical record	Scanner type	CT scan
------------	------	-------------	----------------	--------------	---------

Data leakage: example 1

- Problem: detect lung cancer from CT scans
- Data: collected from hospital A
- Performs well on test data from hospital A
- Performs poorly on test data from hospital B

Patient ID	Date	Doctor note	Medical record	Scanner type	CT scan
------------	------	-------------	----------------	--------------	---------



At hospital A, when doctors suspect that a patient has lung cancer, they send that patient to a higher-quality scanner

Data leakage: example 2

- Problem: predicting how many views an article will get
- Data: historical data on the site
- Where might data leakage come from?

Article ID	Date	Title	Article	Author	Language	Translations
------------	------	-------	---------	--------	----------	--------------

Data leakage: example 2

- Problem: predicting how many views an article will get
- Data: historical data on the site

Not leakage because author popularity also available during inference



Article ID	Date	Title	Article	Author	Language	Translations
------------	------	-------	---------	--------	----------	--------------

The site only translate articles that are already gaining attention

Data leakage: Kaggle edition



What will be done about data leaks?

Louka Ewington-Pitsos · Last comment 2Y ago by Gunes Evitan

▲ 23

7 comments ...



Time for Kaggle to clarify

bluetrain · Last comment 2Y ago by Vishal

▲ 56

8 comments ...



There is something new about Leaks ?

Haythem Tellili · Last comment 2Y ago by Oskin Nikita

▲ 4

1 comment ...



New leakage web finded

Zhang Yunfei · Last comment 2Y ago by Zhang Yunfei

▲ 19

19 comments ...



The leak explained!

Posted in [liverpool-ion-switching](#) 2 years ago

▲ 60

The cat5 data (category with 10 open channels) is very similar to the addition of two signals of cat4 data, as several teams noticed. But also, we can find that the data from 4000001 to 4100000 (cat4) was used to create the data from 5700001 to 5800000 : $\text{openchannels}(5700001:5800000) = \text{openchannels}(4000001 \text{ to } 4100000) + \text{openchannels of other cat4 data (that I didn't really look for)}$. So we can just subtract data $\text{openchannels}(4000001 \text{ to } 4100000)$ from the private LB data. Then we have cat4 data, much easier to predict. By using this method, we just reached a score of 0.9543. I guess by digging further, it can lead to the score of 0.985!

Zidmie

Topic Author

3rd place

[ASHRAE - Great Energy Predictor III](#)

[University of Liverpool - Ion Switching](#)

Common causes for data leakage

- Splitting time-correlated data randomly instead of by time
- Scaling before splitting
- Filling in missing data with statistics from the test split
- Poor handling of data duplication before splitting
- Group leakage
- Leakage from data generation process

Common causes for data leakage

- Target is a function of a feature

Country	Population	Region	...	GDP per capita	GDP
France	67M	Europe	...	38,800	2.6T
Germany	83M	Europe	...	44,578	3.7T
...
China	1386M	Asia	...	8,802	12.2T

Common causes for data leakage

- Feature hides the target

Customer ID	Group	Yearly Spendings	Yearly Pageviews	...	Gender
1	M18-25	1350	11,987	...	M
2	F25-35	2365	8,543	...	F
...
18879	F65+	3653	6,775	...	F

Common causes for data leakage

- Feature from the future

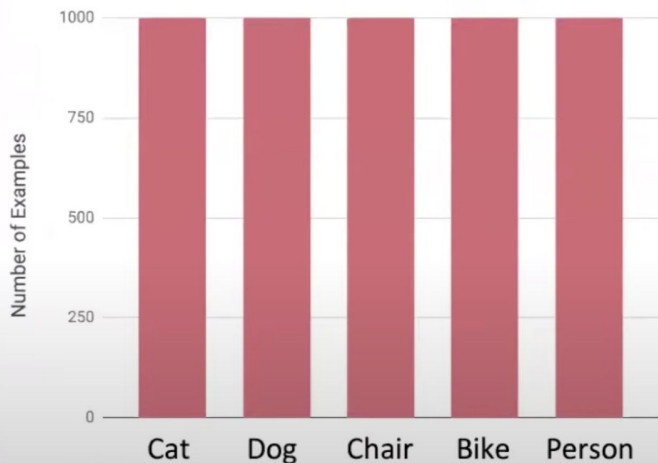
Borrower ID	Demographic Group	Education	...	Late Payment Reminders	Will Pay Loan
1	M35-50	High school	...	0	Y
2	F25-35	Master's	...	1	N
...
65723	M25-35	Master's	...	3	N

7. Data Imbalance

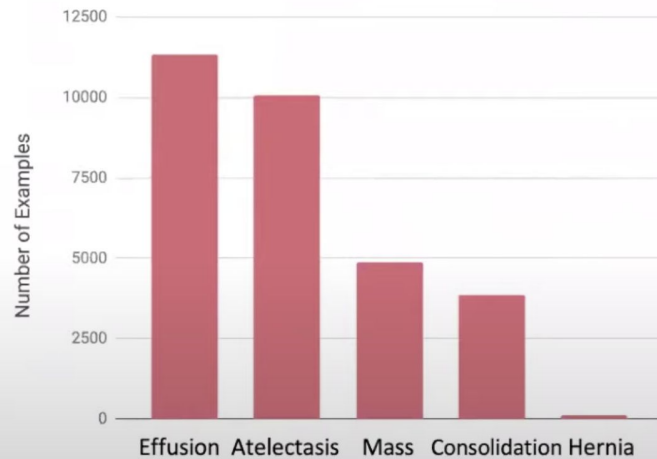
Class imbalance

Small data and rare occurrences

ML works well when
the data distribution
is this:



Not so well when it
is this:



Why is class imbalance hard?

- Not enough signal to learn about rare classes

Why is class imbalance hard?

- Not enough signal to learn about rare classes
- Statistically, predicting majority label has higher chance of being right
 - If a majority class accounts 99% of data, always predicting it gives 99% accuracy

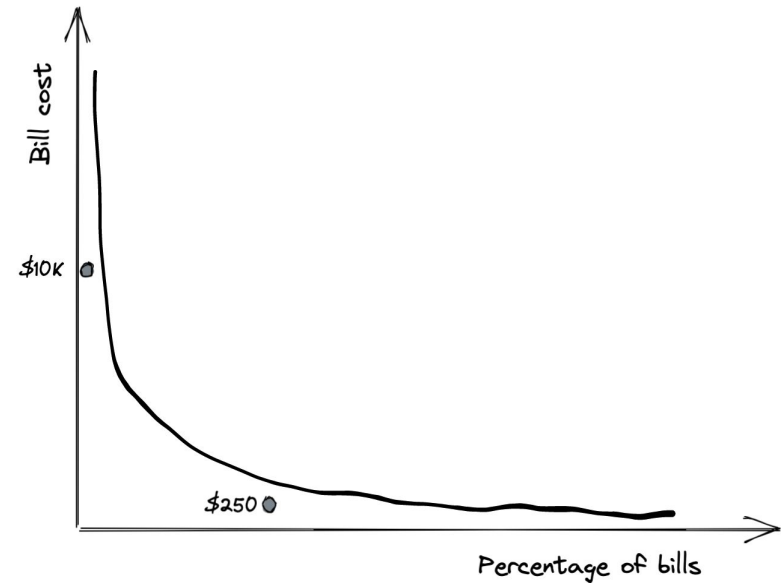


Why is class imbalance hard?

- Not enough signal to learn about rare classes
- Statistically, predicting majority label has higher chance of being right
- Asymmetric cost of errors: different cost of wrong predictions

Asymmetric cost of errors: regression

- 95th percentile: \$10K
- Median: \$250



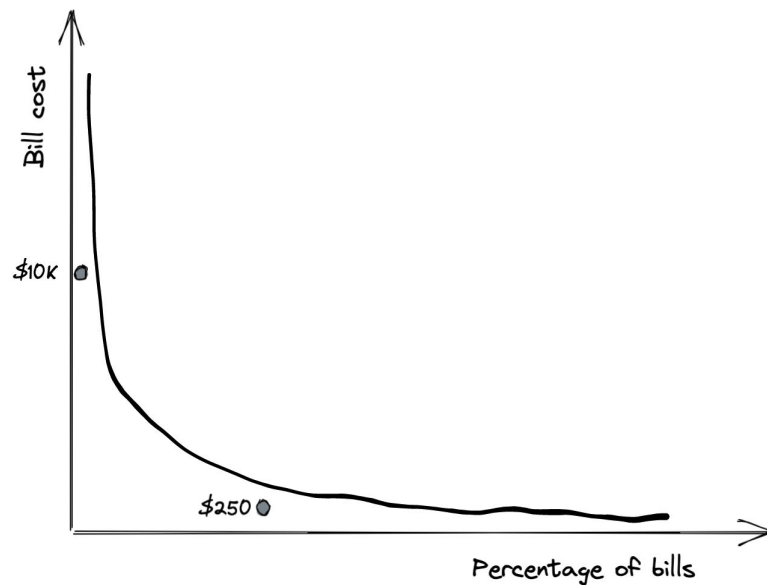
Asymmetric cost of errors: regression

100% error difference

Not OK

- \$10K bill: off by \$10K
- \$250 bill: off by \$250

OK



Class imbalance is the norm

- Fraud detection
- Spam detection
- Disease screening
- Churn prediction
- Resume screening
 - E.g. 2% of resumes pass screening
- Object detection
 - Most bounding boxes don't contain any object

People are more interested in unusual/potentially catastrophic events



Sources of class imbalance

- Domain specific
 - Costly, slow, or infeasible to collect data of certain classes
- Sampling biases
 - Narrow geographical areas (self-driving cars)
 - Selection biases
- Labeling errors

How to deal with class imbalance

1. Choose the right metrics
2. Data-level methods
3. Algorithm-level methods

Choose the right metrics

Model A vs. Model B confusion matrices

Which model would you choose?

Model A	Actual CANCER	Actual NORMAL
Predicted CANCER	10	10
Predicted NORMAL	90	890

Model B	Actual CANCER	Actual NORMAL
Predicted CANCER	90	90
Predicted NORMAL	10	810

Choose the right metrics

Model A vs. Model B confusion matrices

Model B has a better chance of telling if you have cancer

Model A	Actual CANCER	Actual NORMAL
Predicted CANCER	10	10
Predicted NORMAL	90	890

Model B	Actual CANCER	Actual NORMAL
Predicted CANCER	90	90
Predicted NORMAL	10	810

Both have the same accuracy: 90%

Symmetric metrics vs. asymmetric metrics

Symmetric metrics	Asymmetric metrics
Treat all classes the same	Measures a model's performance w.r.t to a class
Accuracy	F1, recall, precision, ROC

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{FP} + \text{TN} + \text{FN})}$$

$$F_1\text{-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}$$

- TP: True positives
- TN: True negatives
- FP: False positives
- FN: False negatives

Class imbalance: asymmetric metrics

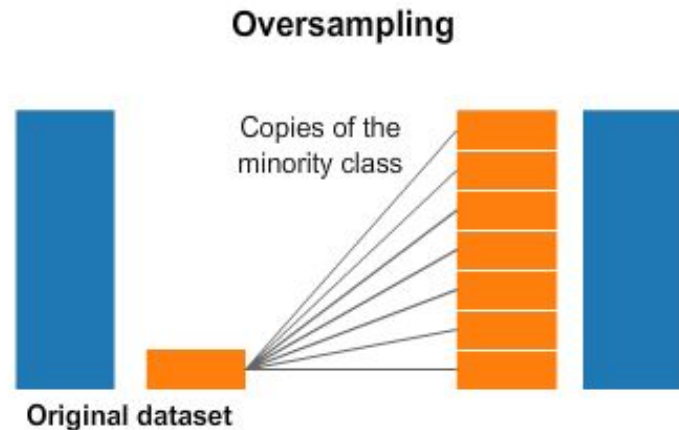
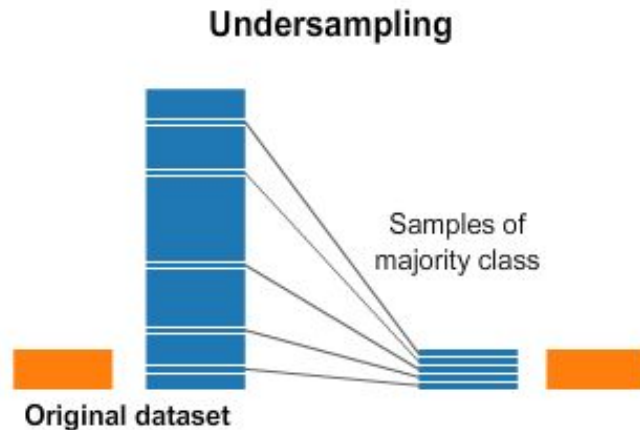
- Your model's performance w.r.t to a class

	CANCER (1)	NORMAL (0)	Accuracy	Precision	Recall	F1
Model A	10/100	890/900	0.9	0.5	0.1	0.17
Model B	90/100	810/900	0.9	0.5	0.9	0.64

! F1 score for CANCER as 1
is different from F1 score for
NORMAL as 1 !

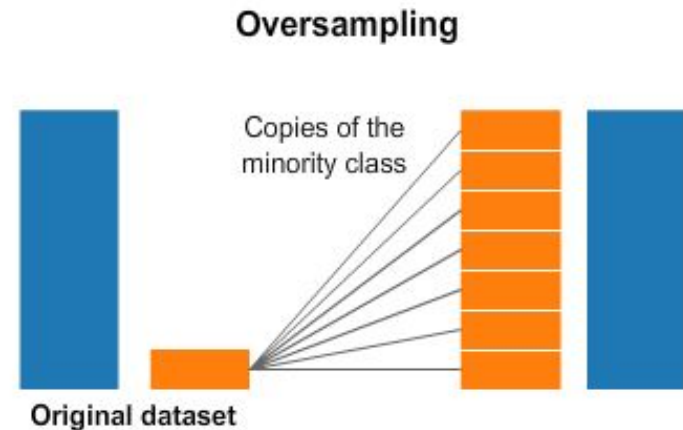
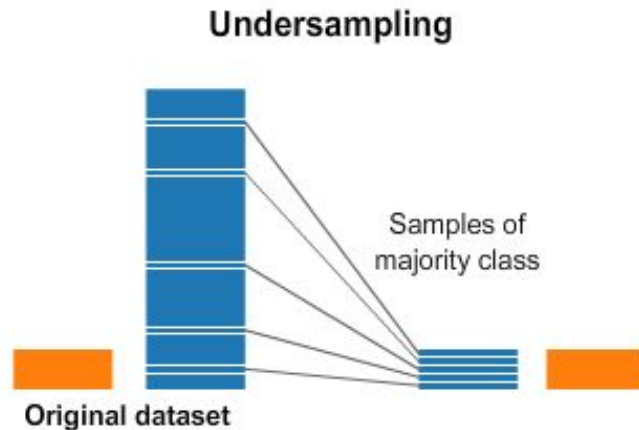
2. Data-level methods: Resampling

Undersampling	Oversampling
Remove samples from the majority class	Add more examples to the minority class



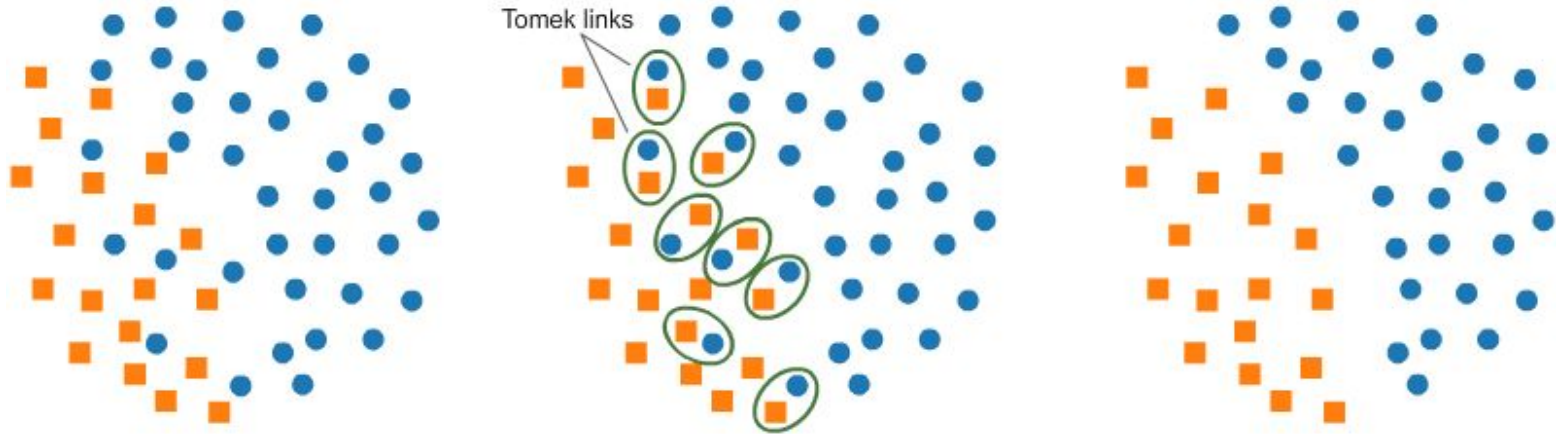
2. Data-level methods: Resampling

Undersampling	Oversampling
Remove samples from the majority class	Add more examples to the minority class
Can cause loss of information	Can cause overfitting



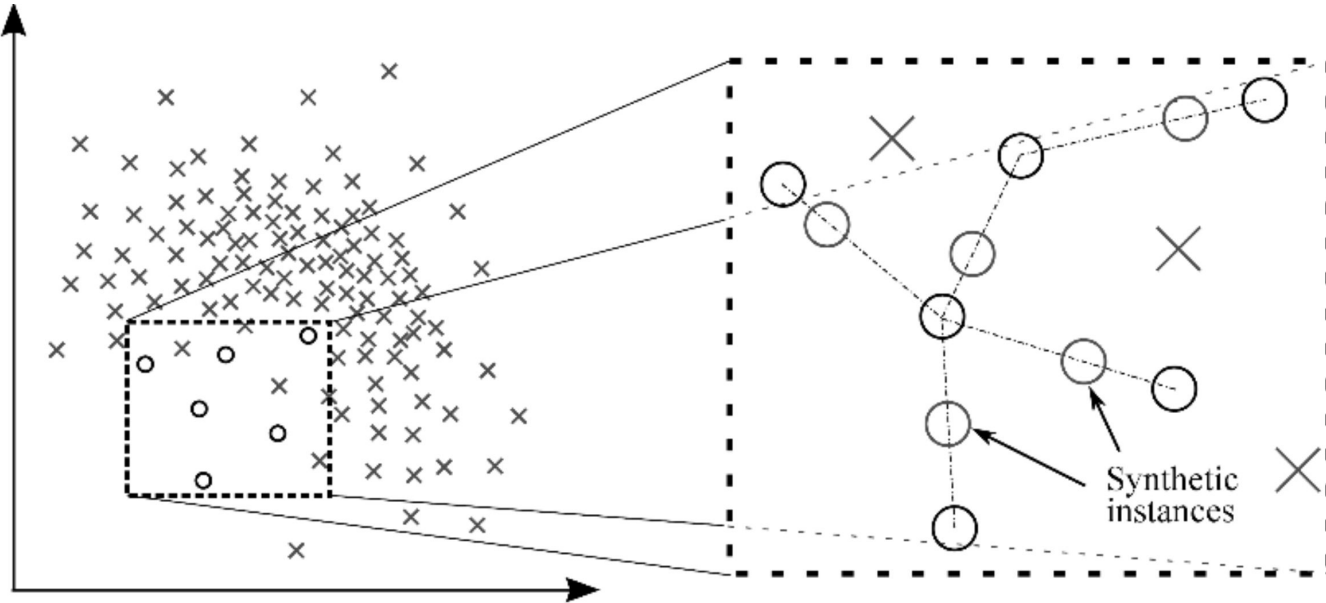
Undersampling: Tomek Links

- Find pairs of close samples of opposite classes
- Remove the sample of majority class in each pair
 - Pros: Make decision boundary more clear
 - Cons: Make model less robust



Oversampling: SMOTE

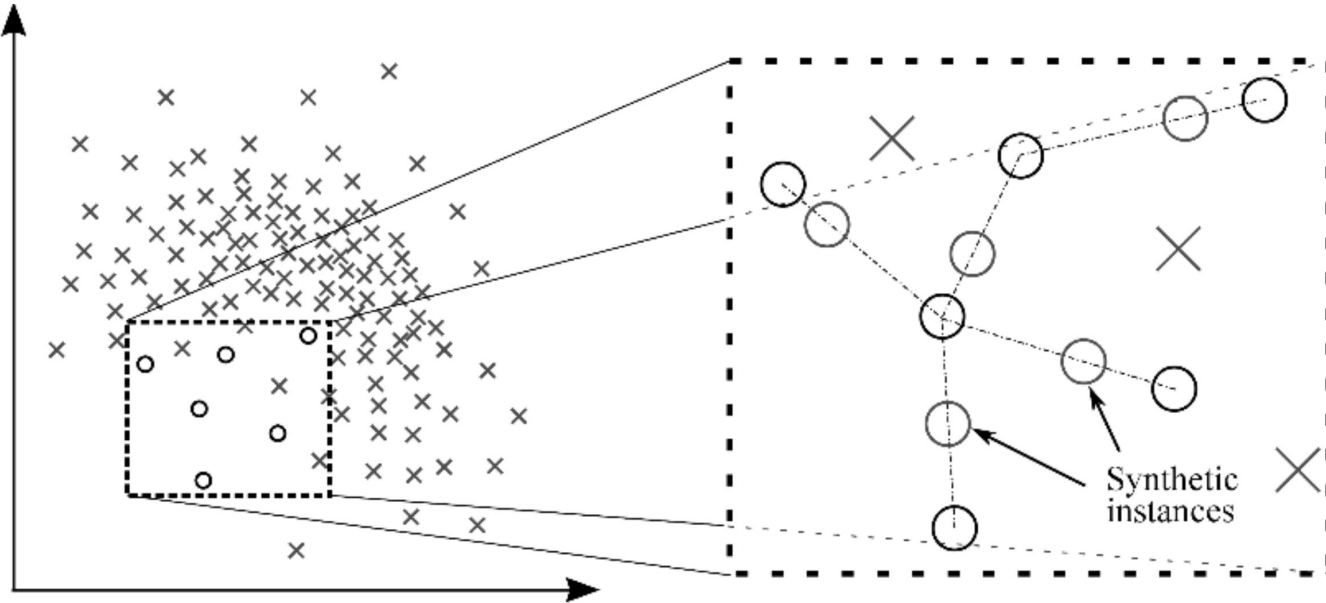
- Synthesize samples of minority class as convex (\sim linear) combinations of existing points and their nearest neighbors of same class.



Oversampling: SMOTE

Both SMOTE and Tomek links only work on low-dimensional data!

- Synthesize samples of minority class as convex (\sim linear) combinations of existing points and their nearest neighbors of same class.



3. Algorithm-level methods

- Naive loss: all samples contribute equally to the loss
- Idea: training samples we care about should contribute more to the loss

$$L(X; \theta) = \sum_x L(x; \theta)$$

3. Algorithm-level methods

- Cost-sensitive learning
- Class-balanced loss
- Focal loss

Cost-sensitive learning

- C_{ij} : the cost if class i is classified as class j

	Actual NEGATIVE	Actual POSITIVE
Predicted NEGATIVE	$C(0, 0) = C_{00}$	$C(1, 0) = C_{10}$
Predicted POSITIVE	$C(0, 1) = C_{01}$	$C(1, 1) = C_{11}$

- The loss caused by instance x of class i will become the weighted average of all possible classifications of instance x .

$$L(x ; \theta) = \sum_j C_{ij} P(j | x; \theta)$$

Class-balance loss

- Give more weight to rare classes

Non-weighted loss

$$L(X; \theta) = \sum_i L(x_i; \theta)$$

Weighted loss

$$L(X; \theta) = \sum_i W_{y_i} L(x_i; \theta)$$

$$W_c = \frac{N}{\text{number of samples of class } C}$$

```
model.fit(features, labels, epochs=10, batch_size=32, class_weight={"fraud": 0.9, "normal": 0.1})
```

Focal loss

- Give more weight to difficult samples:
 - downweights well-classified samples

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases}$$

$$\text{CE}(p_t) = -\log(p_t)$$

$$\text{FL}(p_t) = -(1 - p_t)^\gamma \log(p_t)$$

8. Data Augmentation

Data augmentation: goals

- Improve model's performance overall or on certain classes
- Generalize better
- Enforce certain behaviors

Data augmentation

1. Simple label-preserving transformation
2. Perturbation
3. Data synthesis

Label-preserving: Computer Vision

Random cropping, flipping,
erasing, etc.



Image from [An Efficient Multi-Scale Focusing Attention Network for Person Re-Identification](#) (Huang et al., 2021)

Label-preserving: NLP

Original sentences	I'm so happy to see you.
Generated sentences	I'm so glad to see you. I'm so happy to see y'all . I'm very happy to see you.

Perturbation: neural networks can be sensitive to noise

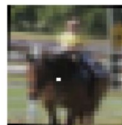
- 67.97% Kaggle CIFAR-10 test images
- 16.04% ImageNet test images

can be misclassified by changing just one pixel
([Su et al., 2017](#))

AllConv



SHIP
CAR(99.7%)



HORSE
DOG(70.7%)



CAR
AIRPLANE(82.4%)



DEER
AIRPLANE(49.8%)



HORSE
DOG(88.0%)

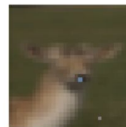
NiN



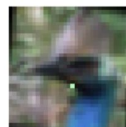
HORSE
FROG(99.9%)



DOG
CAT(75.5%)



DEER
DOG(86.4%)

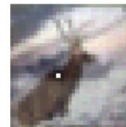


BIRD
FROG(88.8%)

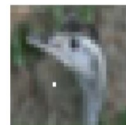


SHIP
AIRPLANE(62.7%)

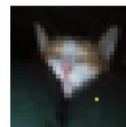
VGG



DEER
AIRPLANE(85.3%)



BIRD
FROG(86.5%)



CAT
BIRD(66.2%)



SHIP
AIRPLANE(88.2%)

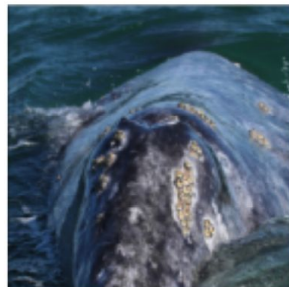


CAT
DOG(78.2%)

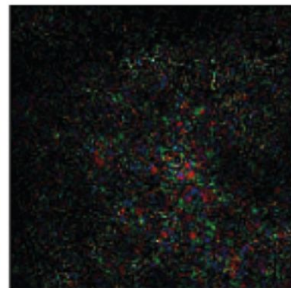
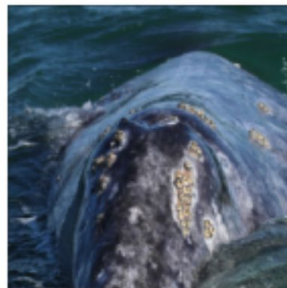
Perturbation: Computer Vision

- Random noise
- Search strategy
 - [DeepFool](#) (Moosavi-Dezfooli et al., 2016): find the minimal noise injection needed to cause a misclassification with high confidence.

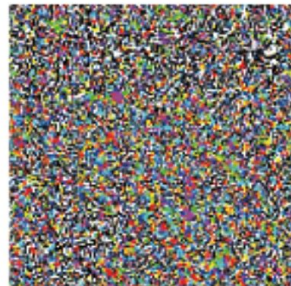
Whale



Turtle
noise by
DeepFool



Turtle
noise by fast
gradient sign



Perturbation: NLP

- Random replacement
 - e.g. BERT ($10\% * 15\% = 1.5\%$)
- 80% of the time: Replace the word with the [MASK] token, e.g., my dog is hairy → my dog is [MASK]
- 10% of the time: Replace the word with a random word, e.g., my dog is hairy → my dog is apple
- 10% of the time: Keep the word unchanged, e.g., my dog is hairy → my dog is hairy. The purpose of this is to bias the representation towards the actual observed word.

Data synthesis: NLP

- Template-based
 - Very common in conversational AI
- Language model-based

Template	Find me a [CUISINE] restaurant within [NUMBER] miles of [LOCATION].
Generated queries	<ul style="list-style-type: none">● Find me a Vietnamese restaurant within 2 miles of my office.● Find me a Thai restaurant within 5 miles of my home.● Find me a Mexican restaurant within 3 miles of Google headquarters.

Data Synthesis: Computer Vision

- Mixup
 - Create convex combination of samples of different classes
 - Labels: cat [3], dog [4]
 - Mixup: 30% dog, 70% cat [$0.3 * 3 + 0.7 * 4 = 3.7$]



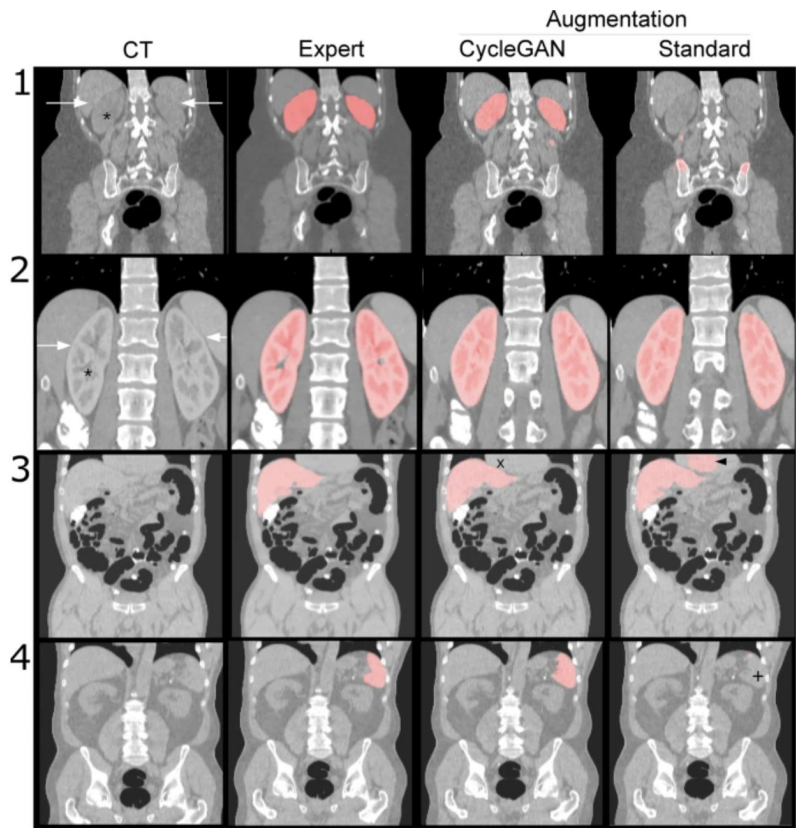
Data Synthesis: Computer Vision

- Mixup
 - Incentivize models to learn linear relationships
 - Improves generalization on speech and tabular data
 - Can be used to stabilize the training of GANs



Data augmentation: GAN

Example: kidney segmentation with data augmentation by CycleGAN



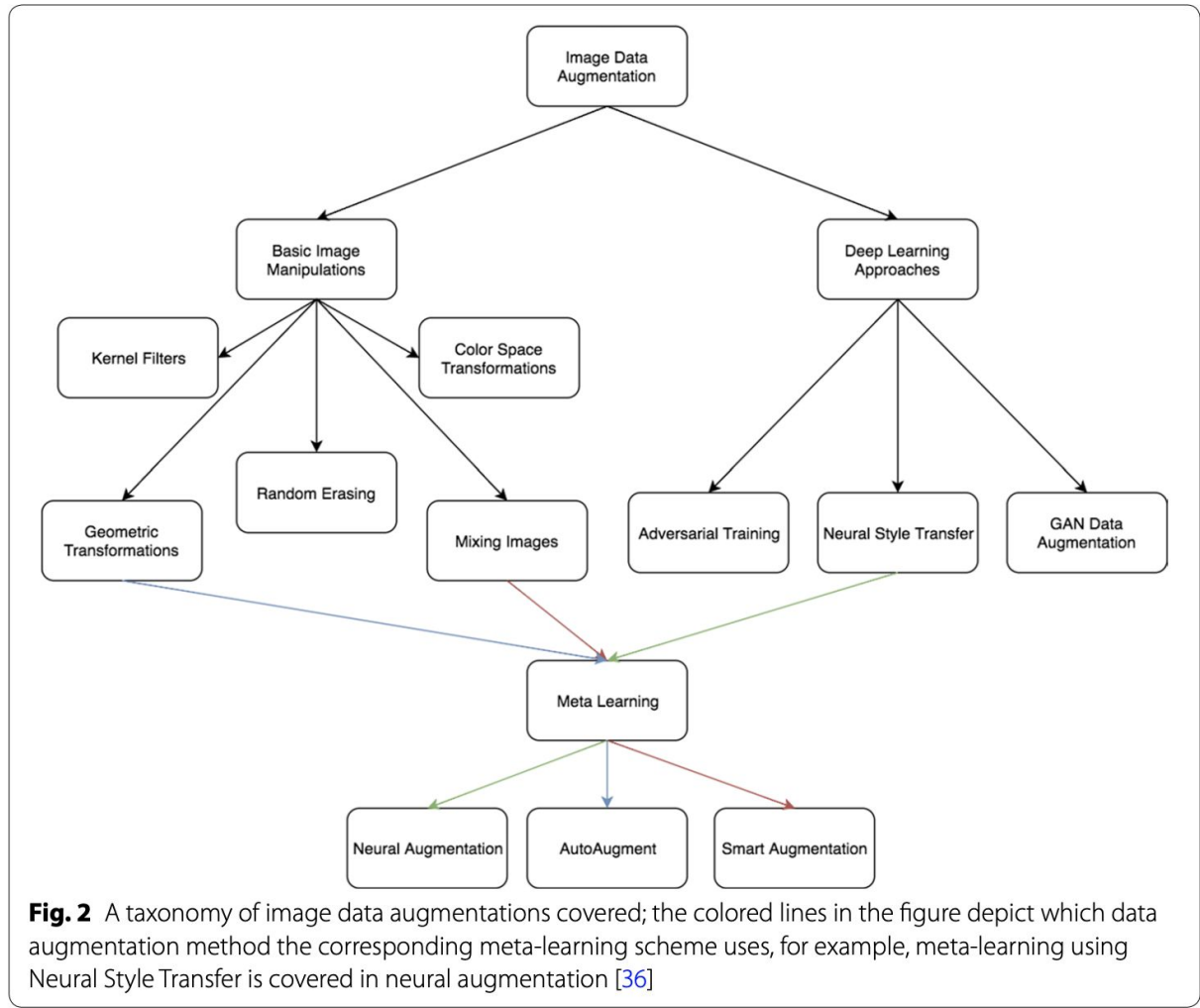


Fig. 2 A taxonomy of image data augmentations covered; the colored lines in the figure depict which data augmentation method the corresponding meta-learning scheme uses, for example, meta-learning using Neural Style Transfer is covered in neural augmentation [36]

Machine Learning Systems Design

Data Lifecycle

Next Lecture: Feature Engineering



CE 40959 Spring 2023

Ali Zarezade

[SharifMLSD.github.io](https://github.com/SharifMLSD)

TEMP slides

Data leakage (ctd.)


Data leakage

- Some form of the label “leaks” into the features
- This same information is not available during inference

Data leakage: example 1

- Problem: detect lung cancer from CT scans
- Data: collected from hospital A
- Performs well on test data from hospital A
- Performs poorly on test data from hospital B

Patient ID	Date	Doctor note	Medical record	Scanner type	CT scan
------------	------	-------------	----------------	--------------	---------



At hospital A, when doctors suspect that a patient has lung cancer, they send that patient to a higher-quality scanner

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time

Partition: shuffle then split

	Week 1	Week 2	Week 3	Week 4	Week 5
Test split	X11	X21	X31	X41	X51
Valid split	X12	X22	X32	X42	X52
Train split	X13	X23	X33	X43	X53
	X14	X24	X34	X44	X54

Aim for similar distributions of labels across splits
e.g. each split has 90% NEGATIVE, 10% POSITIVE

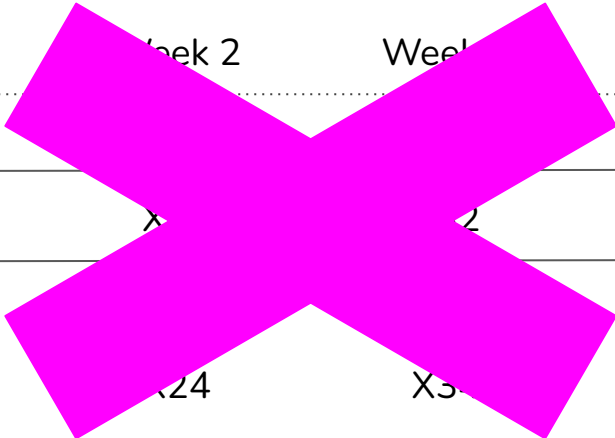
Partition: shuffle then split

	Week 1	Week 2	Week 3	Week 4	Week 5
Test split	X11	X21	X31	X41	X51
Valid split	X12	X22	X32	X42	X52
Train split	X13	X23	X33	X43	X53
	X14	X24	X34	X44	X54



Not representative of real-world usage!

Partition: shuffle then split



	Week 1	Week 2	Week 3	Week 4	Week 5
Test split	X11	X21	X31	X41	X51
Valid split	X12	X22	X32	X42	X52
Train split	X13	X23	X33	X43	X53
	X14	X24	X34	X44	X54

A source of data leakage. Examples:

- stock price prediction
- song recommendation

A better partition

Train split					
Week 1	Week 2	Week 3	Week 4	Week 5	
X11	X21	X31	X41	X51	Valid split
X12	X22	X32	X42	X52	
X13	X23	X33	X43	X53	Test split
X14	X24	X34	X44	X54	
...	

Solution: split data by time

Train split					
Week 1	Week 2	Week 3	Week 4	Week 5	
X11	X21	X31	X41	X51	Valid split
X12	X22	X32	X42	X52	
X13	X23	X33	X43	X53	Test split
X14	X24	X34	X44	X54	
...	

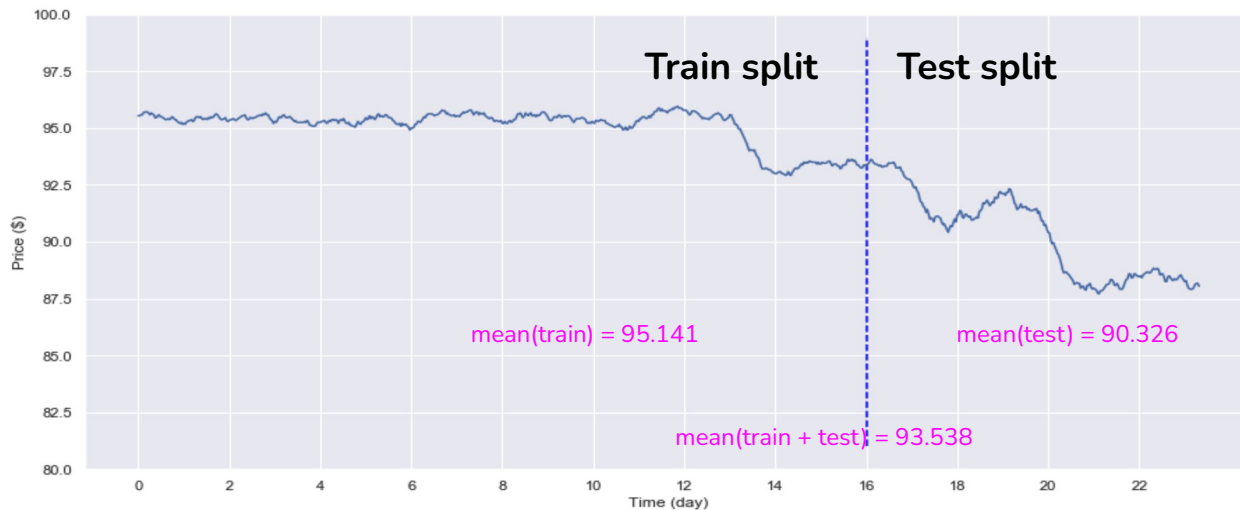
Also forces you to think about
the cold-start problem

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
 - a. Use the whole dataset (including valid/test) to generate global statistics/info

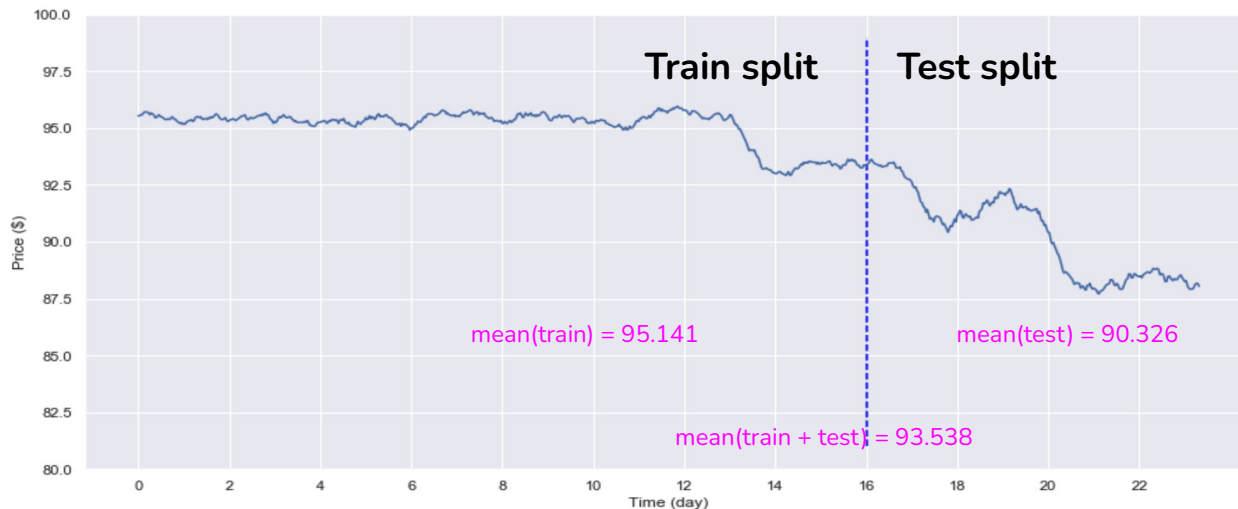
2. Data processing before splitting

- Use the whole dataset (including valid/test) to generate global statistics/info
 - mean, variance, min, max, n-gram count, vocabulary, etc.
- Statistics are then used to process test data
 - scale, fill in missing values, etc.



2. Data processing before splitting

- Use the whole dataset (including valid/test) to generate global statistics/info
- **Solution:**
 - Split your data before scaling/filling in missing values
 - Split even before any EDA to ensure you're blind to the test set

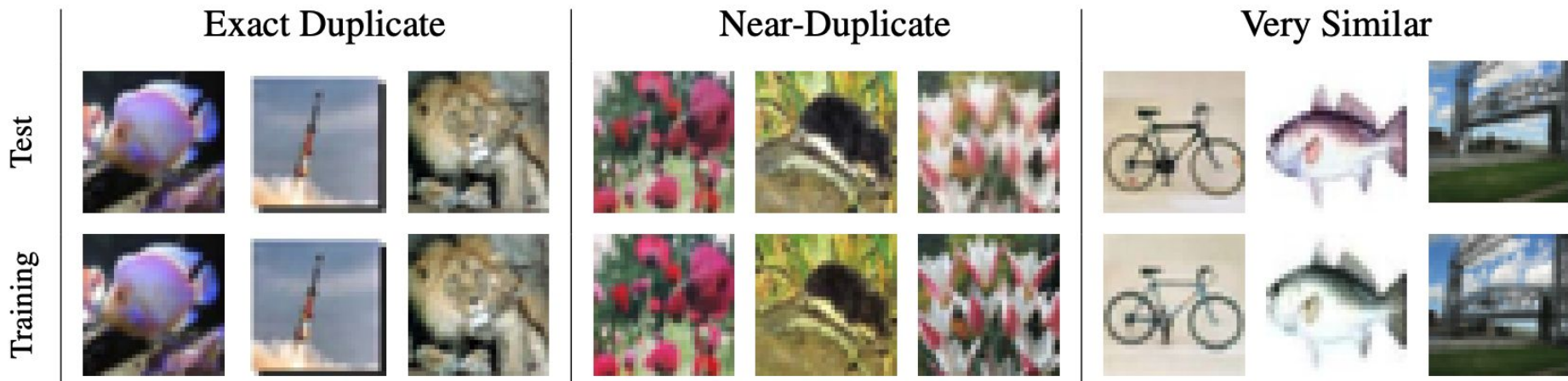


Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
3. Poor handling of data duplication before splitting
 - a. Test set includes data from the train set

3. Poor handling of data duplication before splitting

- Datasets come with duplicates & near-duplicates
 - 3.3% CIFAR-10 and 10% CIFAR-100 test images have dups in training set
 - Removing dups increases errors 17.05% -> 19.38% on CIFAR-100 [PyramidNet-272-200]



3. Poor handling of data duplication before splitting

- Datasets come with duplicates & near-duplicates
- Oversampling can cause duplications

3. Poor handling of data duplication before splitting

- Test set includes data from the train set
- Solution:
 - Deduplicate data before splitting
 - Oversample after splitting

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
3. Poor handling of data duplication before splitting
4. **Group leakage**
 - a. A group of examples have strongly correlated labels but are divided into different splits

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
3. Poor handling of data duplication before splitting
4. Group leakage
 - a. A group of examples have strongly correlated labels but are divided into different splits
 - b. Example: CT scans of the same patient a week apart
 - c. **Solution: Understand your data and keep track of its metadata**

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
3. Poor handling of data duplication before splitting
4. Group leakage
5. Leakage from data generation & collection process
 - a. Example: doctors send high-risk patients to a better scanner
 - b. Solution: Data normalization + subject matter expertise

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
3. Poor handling of data duplication before splitting
4. Group leakage
5. Leakage from data generation & collection process

How to detect leakage?

How to detect leakage?

1. Measure correlation of a feature with labels
 - a. A feature alone might not cause leakage, but 2 features together might

How to detect leakage?

1. Measure correlation of a feature with labels
2. Feature ablation study
 - a. If removing a feature causes the model performance to decrease significantly, figure out why.

How to detect leakage?

1. Measure correlation of a feature with labels
2. Feature ablation study
3. Monitor model performance as more features are added
 - a. Sudden increase: either a very good feature or leakage!