Machine-Learning for Handling Imbalanced Data

Dr Niamh Mimnagh, PR Statistics

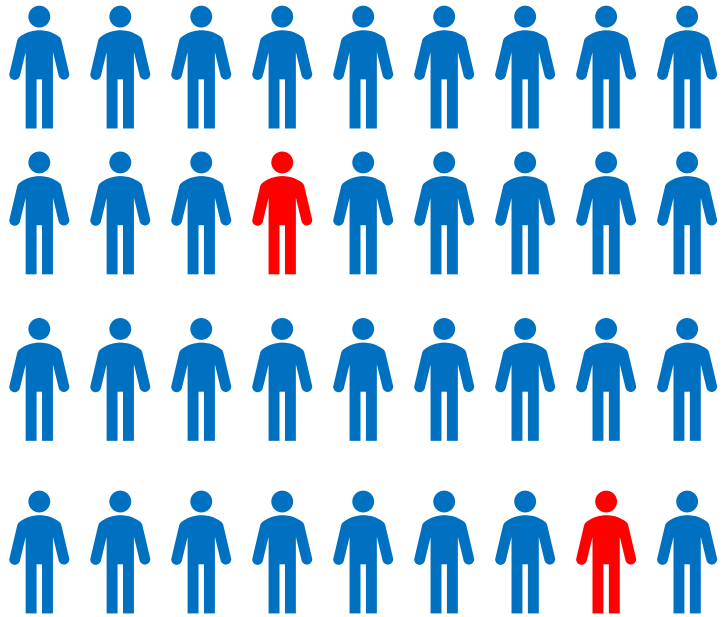
https://github.com/niamhmimnagh/imbalanced-data-course

69th Rbras

5th August 2025

What is Imbalanced Data?

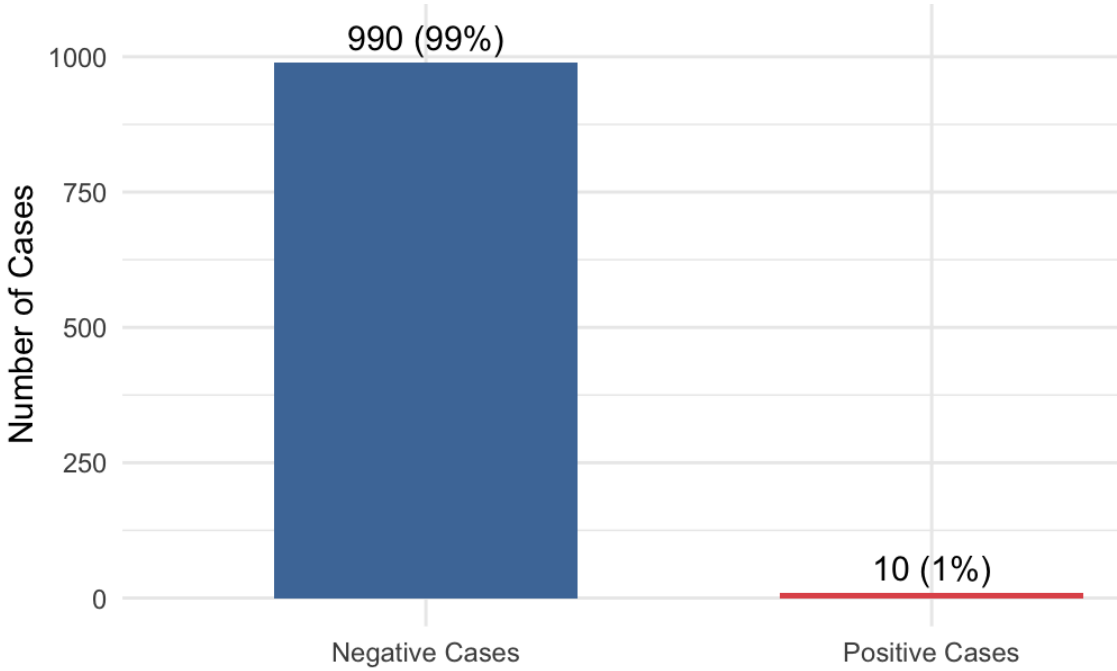
- A dataset is imbalanced when one class heavily outnumbers the others.
- In many critical applications, rare events are the most important:
  - Medical diagnosis: rare disease detection.
  - Finance: identifying fraudulent transactions.
  - Manufacturing: spotting defective products.
  - Ecology: endangered species classification.
- Standard machine learning models may fail to perform well on rare classes because they focus on overall accuracy.
- This course helps you understand how to build better models for such imbalanced problems.



https://github.com/niamhmimnagh/imbalanced-data-course

Impact of Imbalance on Modelling

- Models learn to minimise overall error. This means they are ‘rewarded’ for ignoring the minority class.
- Minority class predictions are often treated as noise.
- Consider a dataset with 990 negative cases and 10 positive cases. A model that always predicts ‘negative’ gets 99% accuracy. But it never detects any positives, which are often the most important!



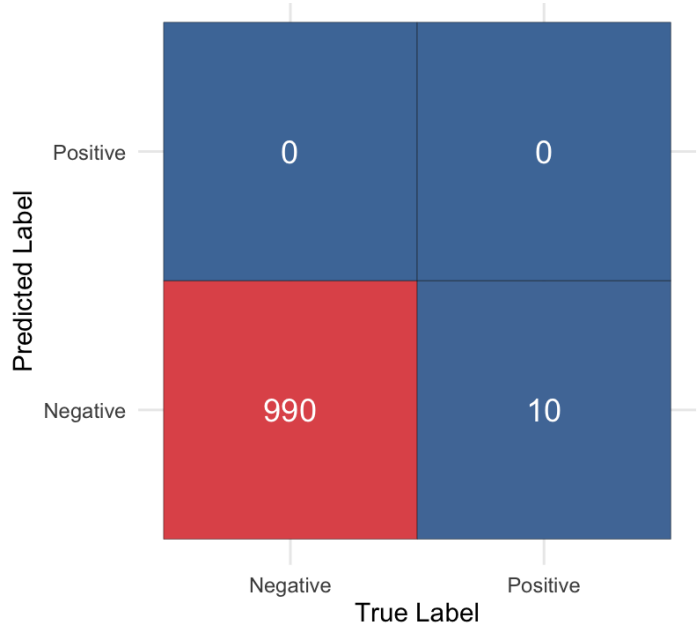
https://github.com/niamhmimnagh/imbalanced-data-course

Accuracy Can Be Misleading

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

- Accuracy measures overall correctness, but can be misleading with imbalanced data, as it is dominated by the majority class.
- A model predicting only the majority class can have >95% accuracy.

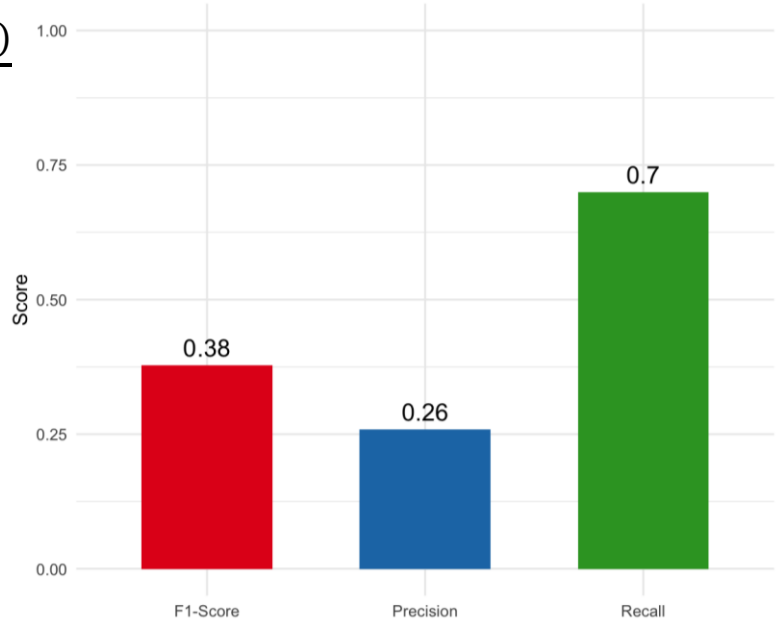
	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)



https://github.com/niamhmimnagh/imbalanced-data-course

Precision and Recall

Precision: $\frac{TP}{TP+FP.}$ Recall: $\frac{TP}{TP+FN}$ F1 Score: $\frac{2(Precision)(Recall)}{Precision+Recall}$



- These metrics focus on performance for the minority class, so they're more reliable in determining model performance when we have imbalanced data.
- If we have a model with high precision, this means we have fewer false positives. If the model has higher recall, we have fewer false negatives.
- Use Recall when missing positives is to be avoided (e.g., medical diagnosis). Use Precision when false alarms are expensive (e.g., fraud alerts).

https://github.com/niamhmimnagh/imbalanced-data-course

Undersampling

Undersampling handles imbalanced data by reducing the size of the majority class so it's closer to the size of the minority class.

How it works:

- Identify the majority class (e.g., 990 negatives) and the minority class (e.g., 10 positives).
- Randomly remove a subset of majority class samples so that the two classes become more balanced.
- Train the model on this smaller, more balanced dataset so it pays more attention to the minority class.

Advantages	Limitations
Fast and simple to implement	Potential information loss when discarding majority-class samples
Reduces training time	Can lead to underfitting
	May not work well if the dataset is already small

Oversampling

Oversampling is a technique for handling imbalanced datasets by increasing the size of the minority class so it's closer to the size of the majority class.

How it works:

- Identify the majority class (e.g., 990 negatives) and the minority class (e.g., 10 positives).
- Duplicate minority-class samples until the dataset is more balanced.
- Train the model on this larger, more balanced dataset, where the minority class has a stronger presence.

Advantages	**Limitations**
Keeps all original data	Risk of overfitting – duplicating minority examples can make the model memorise them
Helps the model learn the patterns in the minority class	Increases training time

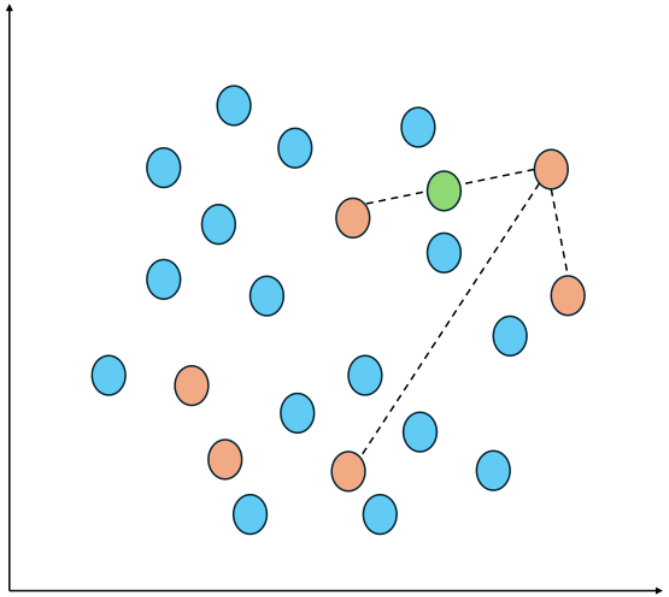
Synthetic Minority Oversampling Technique

- SMOTE creates synthetic minority-class samples instead of simply duplicating existing ones.

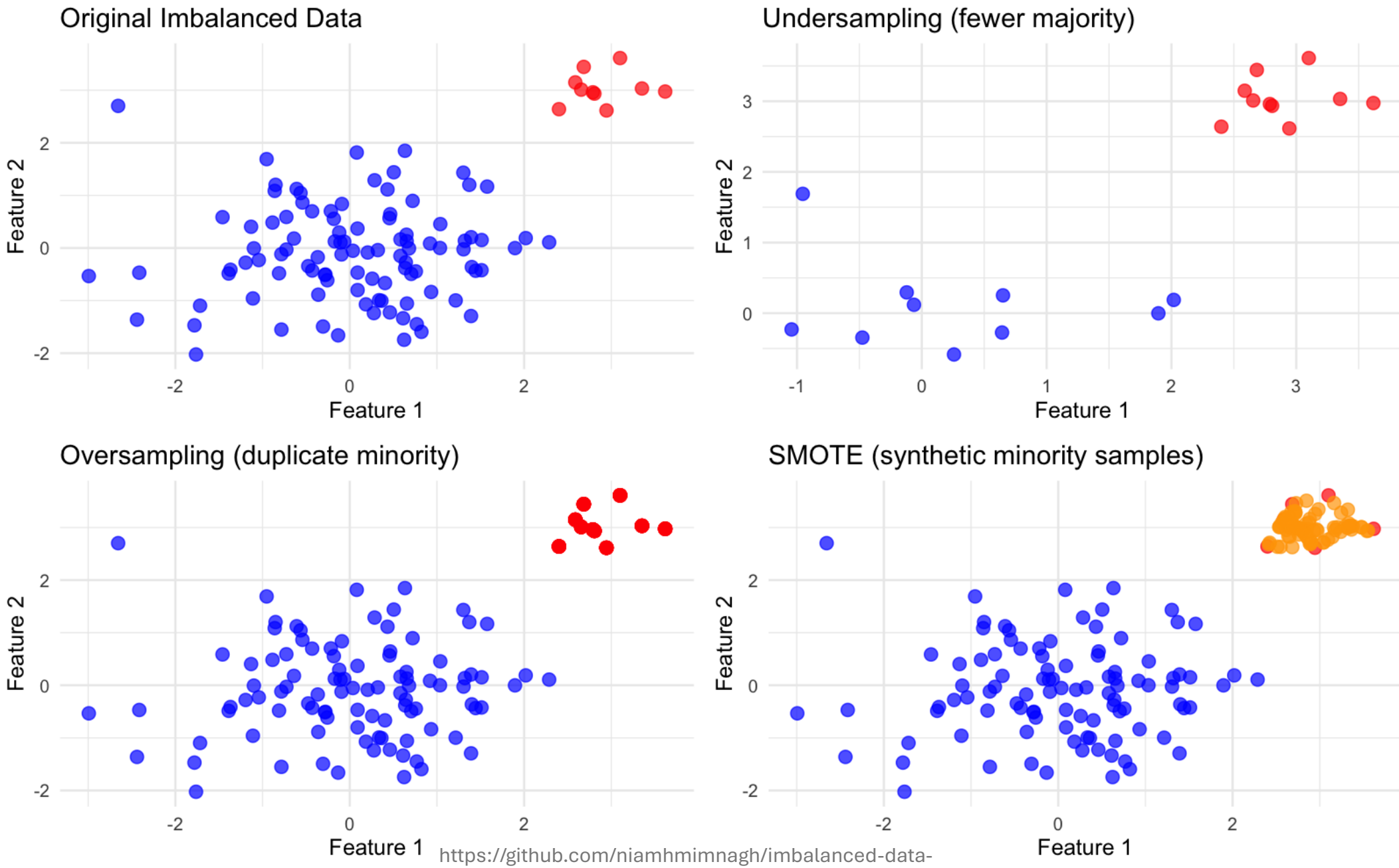
How it works:

- For each minority-class example, find its k-nearest neighbours in the minority class.
- Randomly pick one neighbour and interpolate a new synthetic sample between the two points.
- Repeat this process until the minority class reaches the desired si

Advantages	Limitations
Reduces overfitting compared to simple oversampling	Can create overlapping classes
Creates more diverse minority samples	May introduce unrealistic samples



https://github.com/niamhmimnagh/imbalanced-data-course



https://github.com/niamhmimnagh/imbalanced-data-course

Choosing a Strategy

Undersampling

- Use when you have plenty of majority-class data and can afford to discard some
- Good for very large datasets where training time matters
- Avoid if dataset is already small, as there is a risk of losing important information

Oversampling

- Use when you want to retain all majority data but need to balance the classes
- Works well for small to medium datasets
- Risk of overfitting since you're just duplicating samples

SMOTE

- Use when you want more diverse minority samples
- Helps models generalise better than plain oversampling
- Be cautious if minority & majority classes overlap heavily. SMOTE can create noisy samples

https://github.com/niamhmimnagh/imbalanced-data-course

German Credit Scores Example

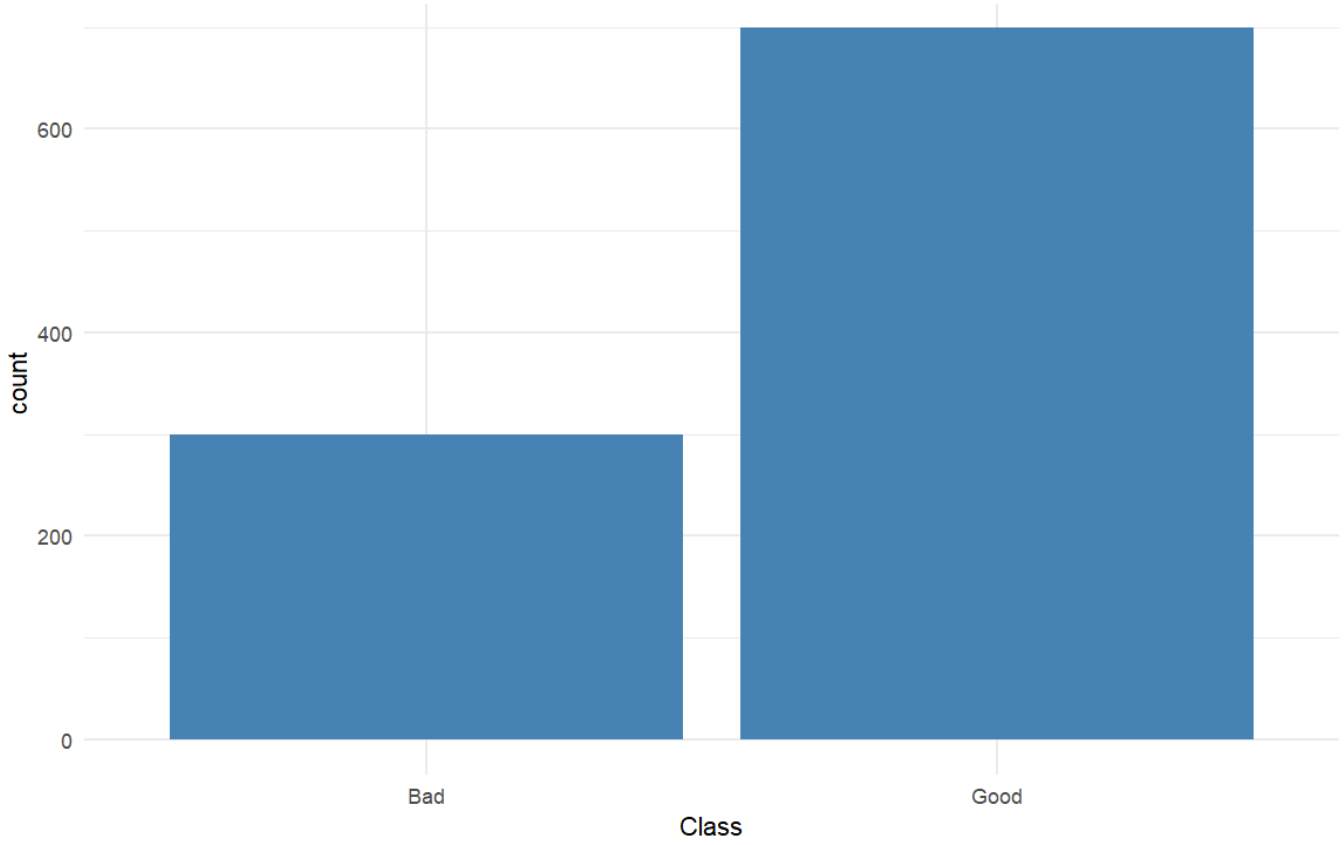
```
# Data structure
str(GermanCredit)
```

```
'data.frame':   1000 obs. of  62 variables:
 $ Duration                              : int  6 48 12 42 24 36 24 36 12 30 ...
 $ Amount                                : int  1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...
 $ InstallmentRatePercentage             : int  4 2 2 2 3 2 3 2 2 4 ...
 $ ResidenceDuration                     : int  4 2 3 4 4 4 4 2 4 2 ...
 $ Age                                   : int  67 22 49 45 53 35 53 35 61 28 ...
 $ NumberExistingCredits                 : int  2 1 1 1 2 1 1 1 1 2 ...
 $ NumberPeopleMaintenance               : int  1 1 2 2 2 2 1 1 1 1 ...
 $ Telephone                             : num  0 1 1 1 1 0 1 0 1 1 ...
 $ ForeignWorker                         : num  1 1 1 1 1 1 1 1 1 1 ...
 $ Class                                 : Factor w/ 2 levels "Bad","Good": 2 1 2 2 1 2 2 2 2 1 ...
 $ CheckingAccountStatus.lt.0            : num  1 0 0 1 1 0 0 0 0 0 ...
 $ CheckingAccountStatus.0.to.200        : num  0 1 0 0 0 0 0 1 0 1 ...
 $ CheckingAccountStatus.gt.200          : num  0 0 0 0 0 0 0 0 0 0 ...
 $ CheckingAccountStatus.none            : num  0 0 1 0 0 1 1 0 1 0 ...
 $ CreditHistory.NoCredit.AllPaid        : num  0 0 0 0 0 0 0 0 0 0 ...
 $ CreditHistory.ThisBank.AllPaid        : num  0 0 0 0 0 0 0 0 0 0 ...
 $ CreditHistory.PaidDuly                : num  0 1 0 1 0 1 1 1 1 0 ...
 $ CreditHistory.Delay                   : num  0 0 0 0 1 0 0 0 0 0 ...
 $ CreditHistory.Critical                : num  1 0 1 0 0 0 0 0 0 1 ...
 $ Purpose.NewCar                        : num  0 0 0 0 1 0 0 0 0 1 ...
 $ Purpose.UsedCar                       : num  0 0 0 0 0 0 0 1 0 0 ...
 $ Purpose.Furniture.Equipment           : num  0 0 0 1 0 0 1 0 0 0 ...
```

https://github.com/niamhmimnagh/imbalanced-data-course

German Credit Scores Example

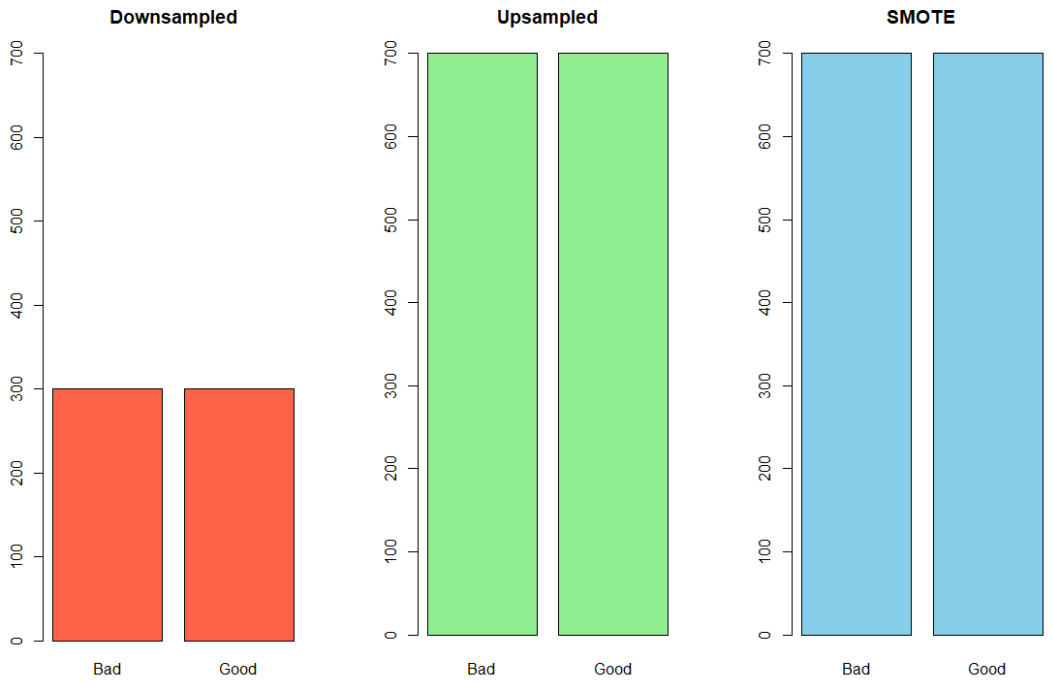
```
# Bar plot
library(ggplot2)
ggplot(GermanCredit, aes(x = Class))
+  geom_bar(fill = "steelblue") +
theme_minimal()
```



https://github.com/niamhmimnagh/imbalanced-data-course

German Credit Scores Example

```
X <- GermanCredit %>% dplyr::select(-Class)
y <- GermanCredit$Class

# Downsampling
library(caret)
down_train <- downSample(x = X, y = y)

# Upsampling
up_train <- upSample(x = X, y = y)

# SMOTE
library(nestedcv)
X_mat <- model.matrix(Class ~ . - 1, data =
GermanCredit)smote_train <- smote(y = y, x =
X_mat)
```



https://github.com/niamhmimnagh/imbalanced-data-course

German Credit Scores Example

```
# Random Forest

library(randomForest)

rf_model <- randomForest(Class ~ ., data = train_data, ntree = 1000)

rf_probs <- predict(rf_model, test_data %>% dplyr::select(-Class), type =
"prob")[, "Bad"]

caret::confusionMatrix(rf_probs, test_data$Class)


rf_model <- randomForest(Class ~ ., data = train_data_imbalanced, ntree =
1000)

rf_probs <- predict(rf_model, test_data %>% dplyr::select(-Class), type =
"prob")[, "Bad"]

caret::confusionMatrix(rf_probs, test_data$Class)
```

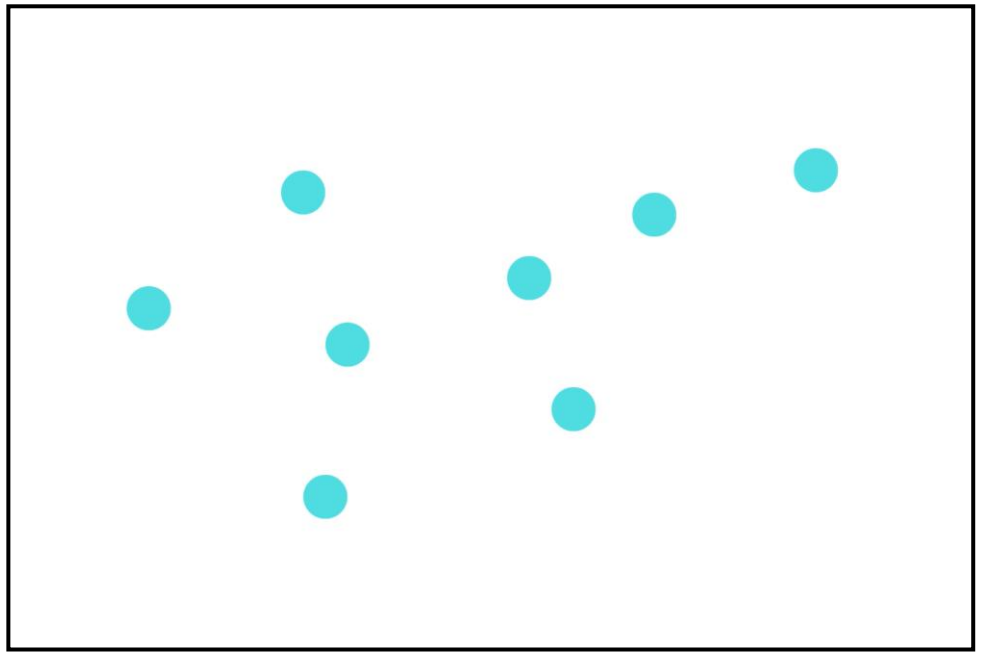
https://github.com/niamhmimnagh/imbalanced-data-course

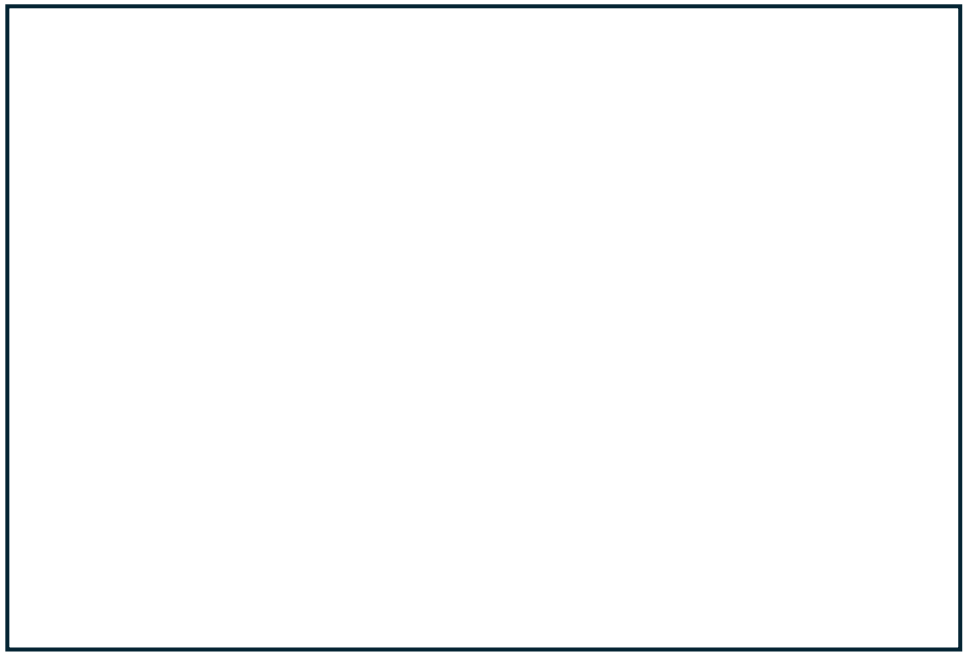
German Credit Scores Example

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Good
      Bad   46   17
      Good  39  193

               Accuracy : 0.8102
                 95% CI : (0.7607, 0.8533)
    No Information Rate : 0.7119
    P-Value [Acc > NIR] : 7.111e-05

                  Kappa : 0.4986

 Mcnemar's Test P-Value : 0.005012

            Sensitivity : 0.5412
            Specificity : 0.9190
         Pos Pred Value : 0.7302
         Neg Pred Value : 0.8319
             Prevalence : 0.2881
         Detection Rate : 0.1559
   Detection Prevalence : 0.2136
      Balanced Accuracy : 0.7301

       'Positive' Class : Bad


Precision: 0.73
Recall: 0.541
F1-score: 0.622
```

```
Confusion Matrix and Statistics

          Reference
Prediction Bad Good
      Bad   68    8
      Good  17  202

               Accuracy : 0.9153
                 95% CI : (0.8774, 0.9444)
    No Information Rate : 0.7119
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.7867

 Mcnemar's Test P-Value : 0.1096

            Sensitivity : 0.8000
            Specificity : 0.9619
         Pos Pred Value : 0.8947
         Neg Pred Value : 0.9224
             Prevalence : 0.2881
         Detection Rate : 0.2305
   Detection Prevalence : 0.2576
      Balanced Accuracy : 0.8810

       'Positive' Class : Bad


Precision: 0.895
Recall: 0.8
F1-score: 0.845
```

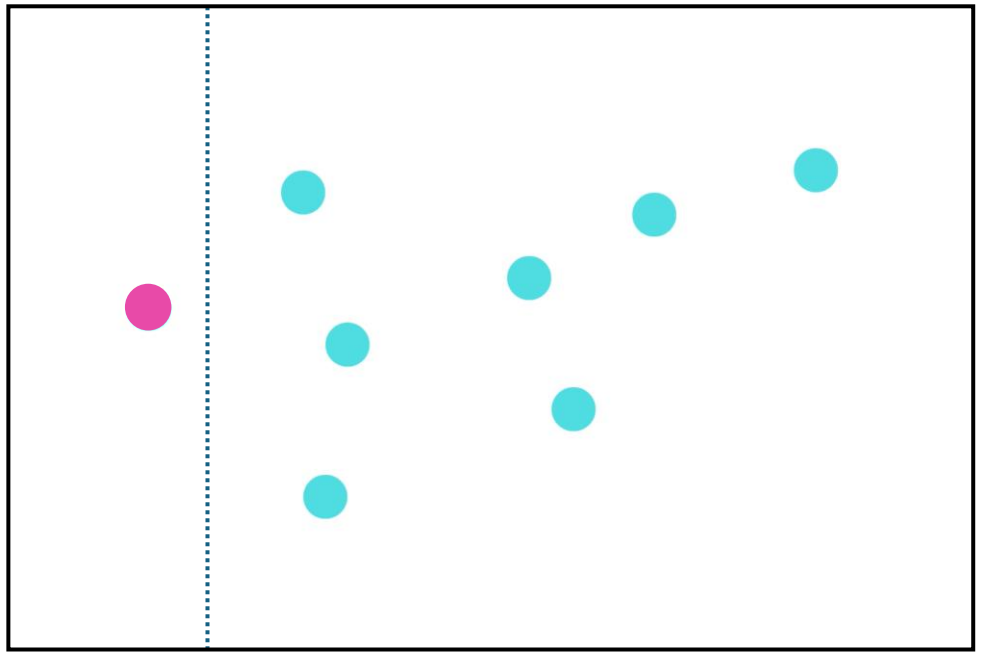
Isolation Forests

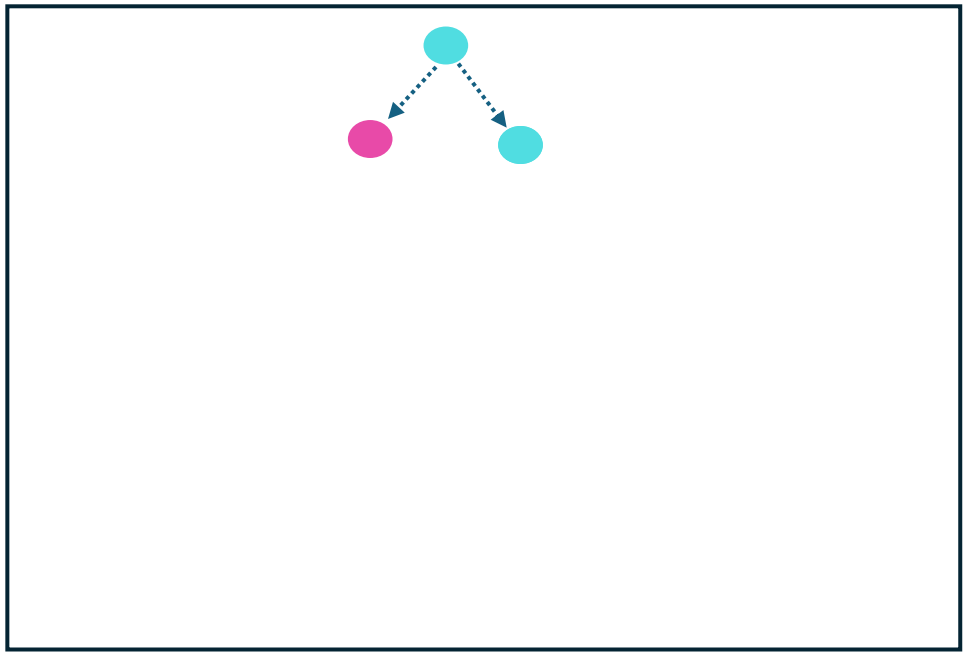




https://github.com/niamhmimnagh/imbalanced-data-course

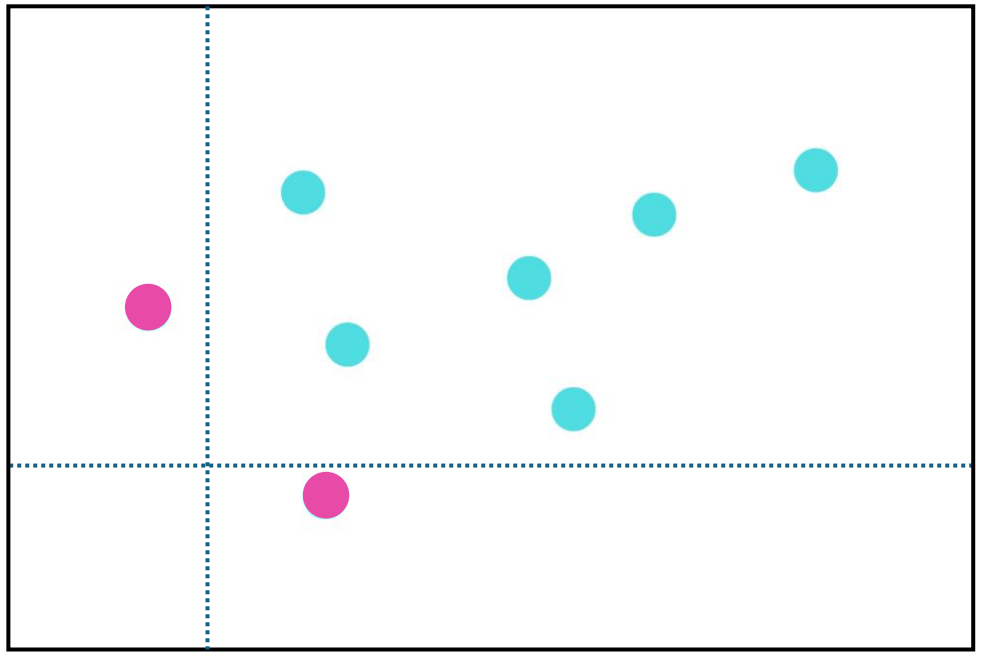
Isolation Forests

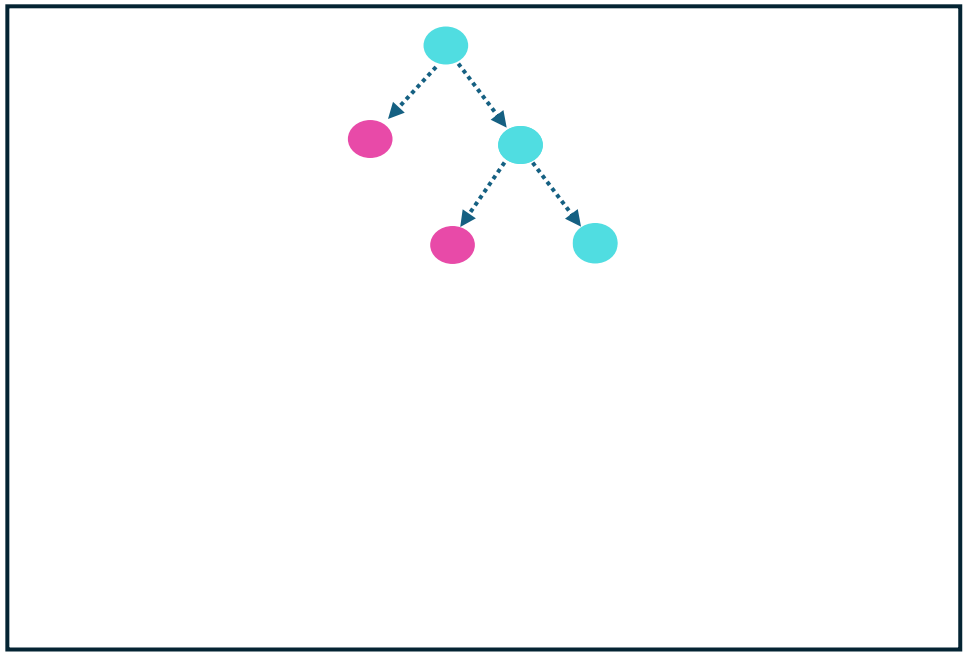




https://github.com/niamhmimnagh/imbalanced-data-course

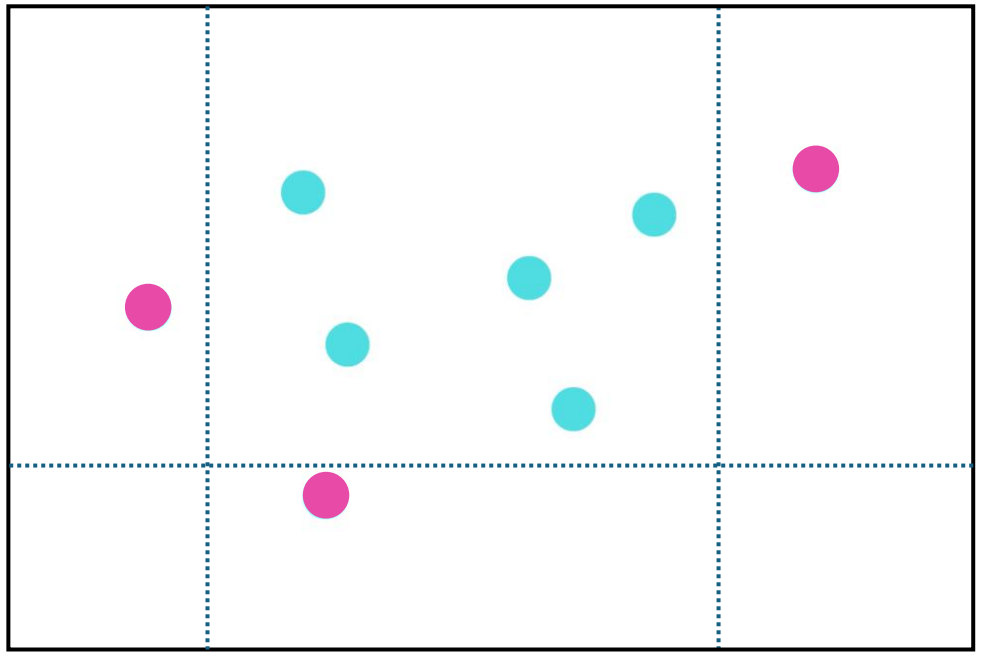
Isolation Forests

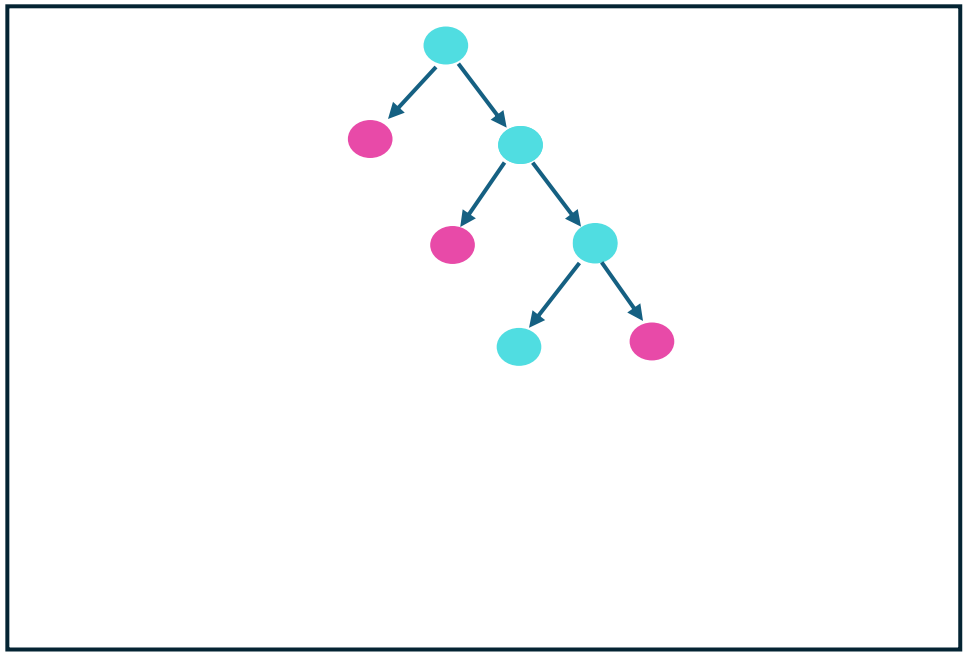




https://github.com/niamhmimnagh/imbalanced-data-course

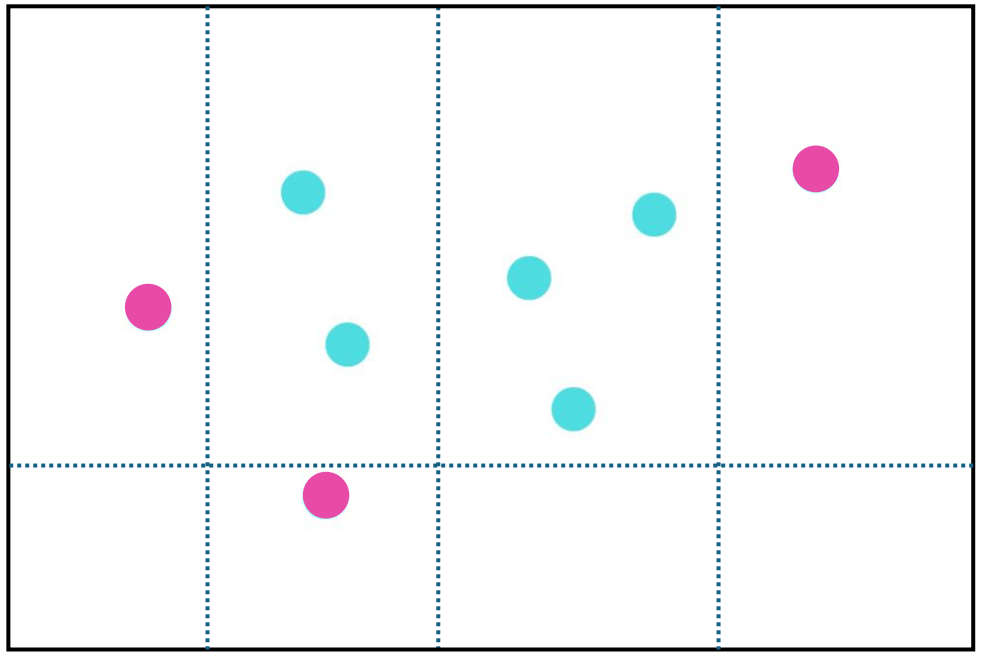
Isolation Forests

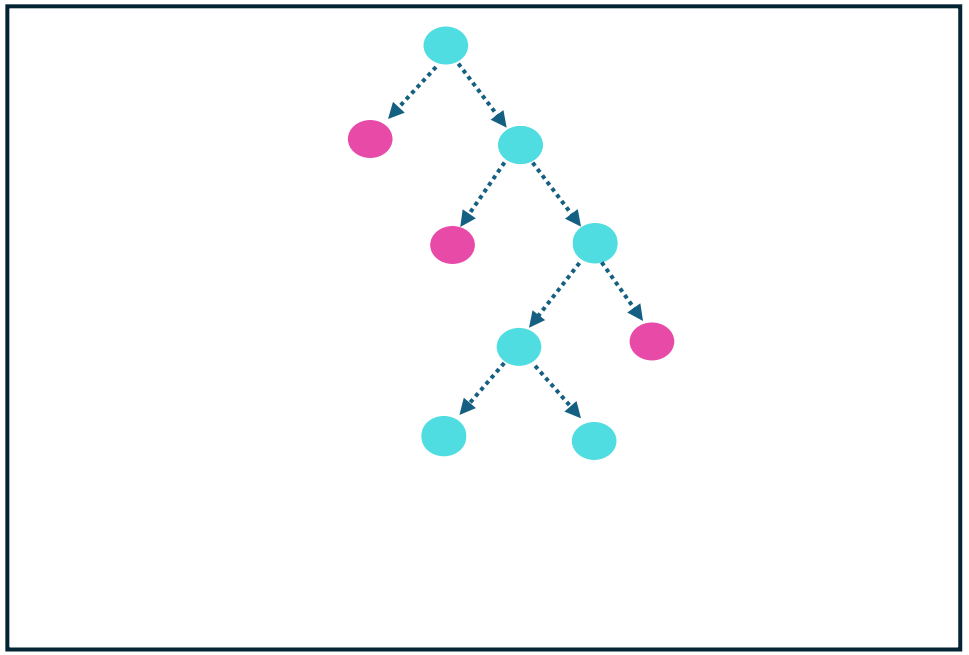




https://github.com/niamhmimnagh/imbalanced-data-course

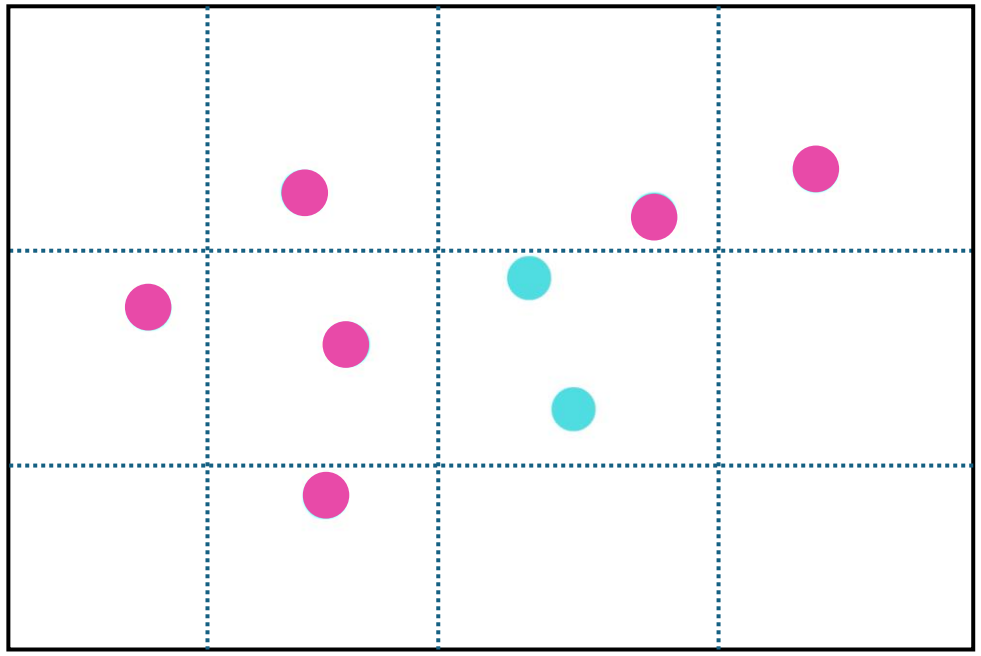
Isolation Forests

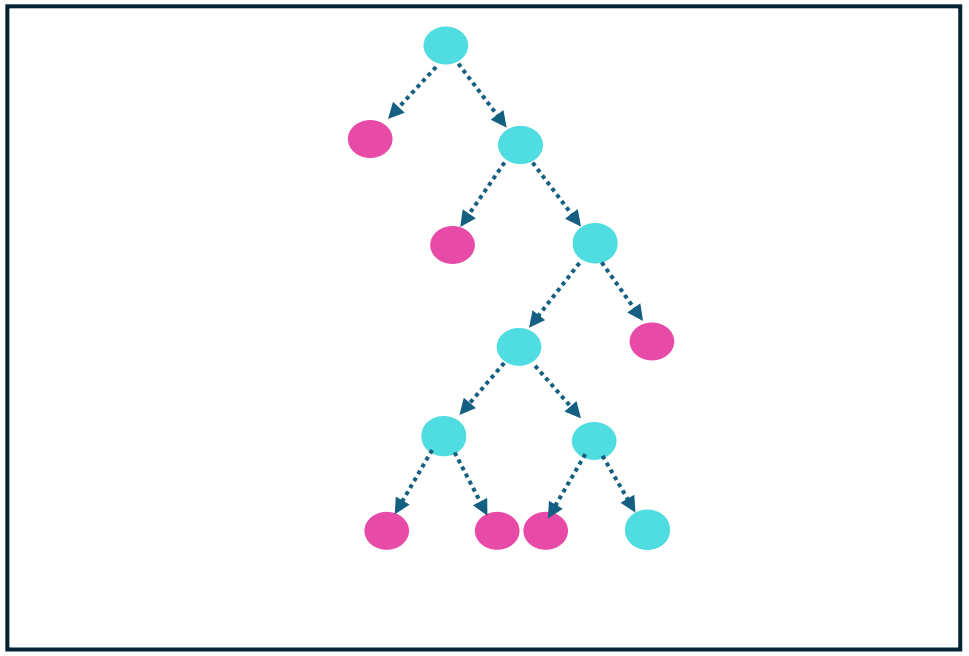




https://github.com/niamhmimnagh/imbalanced-data-course

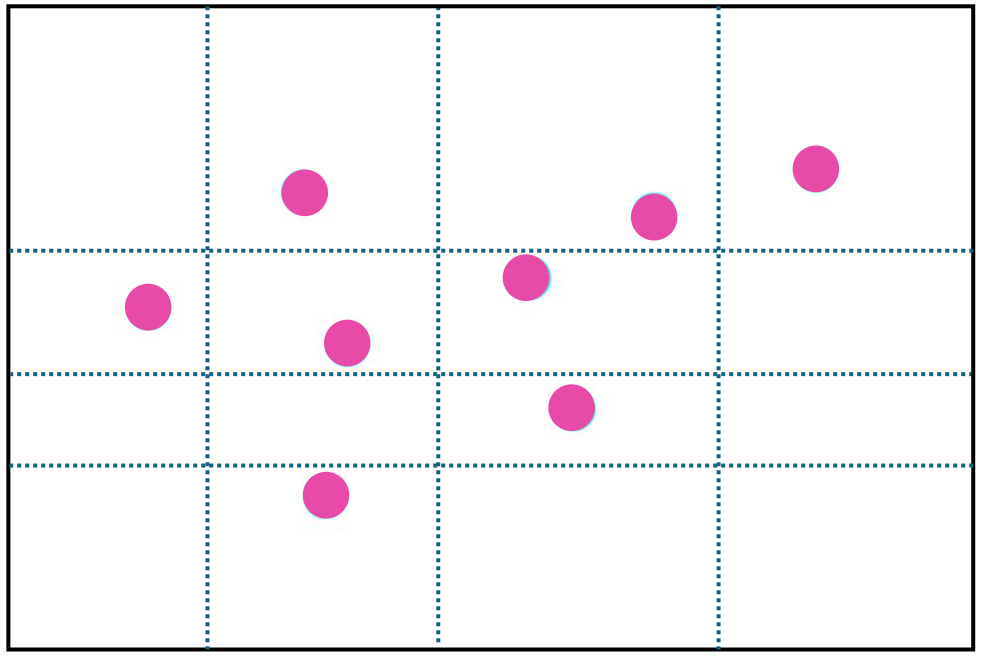
Isolation Forests

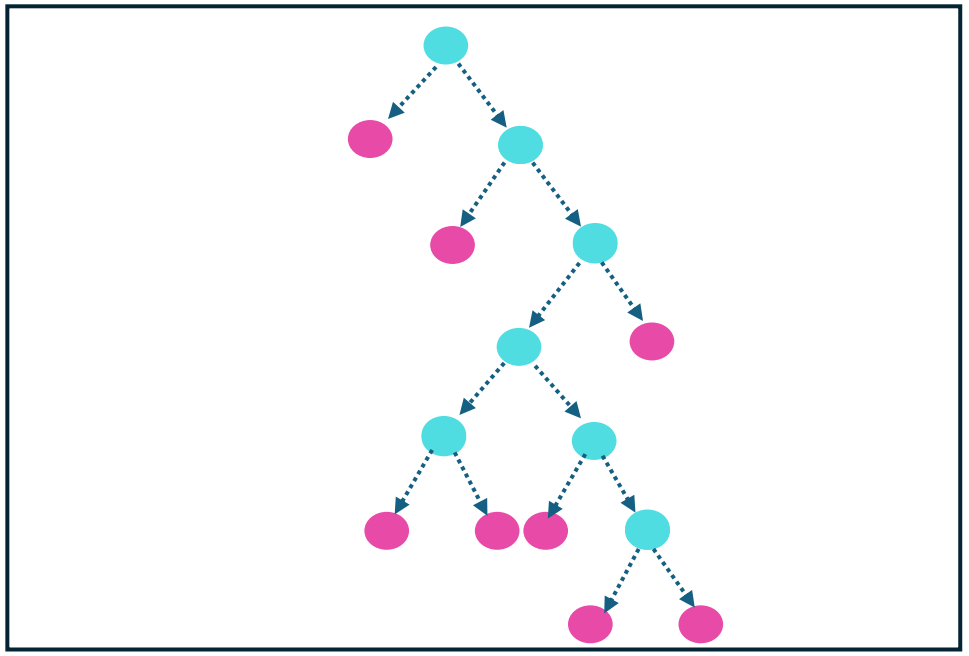




https://github.com/niamhmimnagh/imbalanced-data-course

Isolation Forests





https://github.com/niamhmimnagh/imbalanced-data-course

Isolation Forests

- Given a dataset X, an isolation tree uses path length $h(x)$ (the number of edges between the root and terminal nodes) to identify anomalies.

- The anomaly score for observation *x* in a sample of size *n* is:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}$$

- Where $c(n)$ is a regularising factor based on how many splits it usually takes to isolate a random point in a dataset of size n.
- If the anomaly score is very close to 1, this indicates an anomaly.
- If the anomaly score is much smaller than 0.5 this indicates a normal observation.

https://github.com/niamhmimnagh/imbalanced-data-course

Isolation Forests

- Advantages
  - Efficient on large datasets
  - No assumptions about data distribution
  - Insensitive to outlier contamination in training

- Limitations
  - Assumes anomalies are few and different
  - Sensitive to subsampling parameters
  - Performance may degrade on small datasets

Isolation Forest

```
# Isolation Forest
library(isotree)
iso_model <- isolation.forest(X_num, nthreads = 1)
iso_scores <- predict(iso_model, X_num)
iso_outliers <- iso_scores > quantile(iso_scores, 0.7)
y_true <- GermanCredit$Class
y_true_bin <- as.factor(ifelse(y_true == "Bad", 1, 0))
iso_pred_bin <- as.factor(as.numeric(iso_outliers))
caret::confusionMatrix(iso_pred_bin, y_true_bin, positive = "1")
```

```
          Reference
Prediction   0   1
         0 513 187
         1 187 113
```

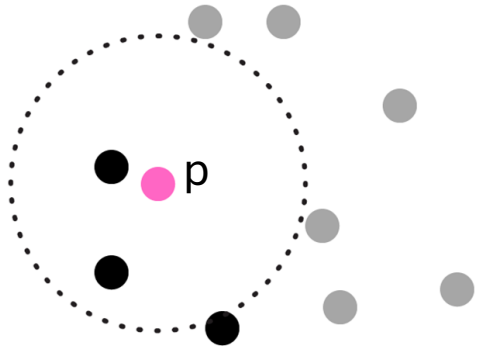
https://github.com/niamhmimnagh/imbalanced-data-course

Local Outlier Factor

We calculate the density of points around an observation and compare it to the densities around nearby points.

$$N_k(p) = \text{k-neighbourhood of p}$$

$$\text{k} - \text{distance}(p) = \text{distance from } p \text{ to its k}^{\text{th}} \text{ nearest neighbour}$$



https://github.com/niamhmimnagh/imbalanced-data-course

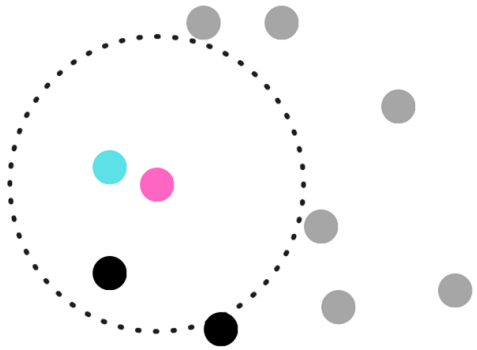
Local Outlier Factor

We calculate the density of points around an observation and compare it to the densities around nearby points.

$$N_k(p) = \text{k-neighbourhood of p}$$

$$\text{k} - \text{distance}(p) = \text{distance from } p \text{ to its k}^{\text{th}} \text{ nearest neighbour}$$

$$\text{Reachability Distance}(p, o) = \max(\text{k} - \text{distance}(o), d(p, o))$$



https://github.com/niamhmimnagh/imbalanced-data-course

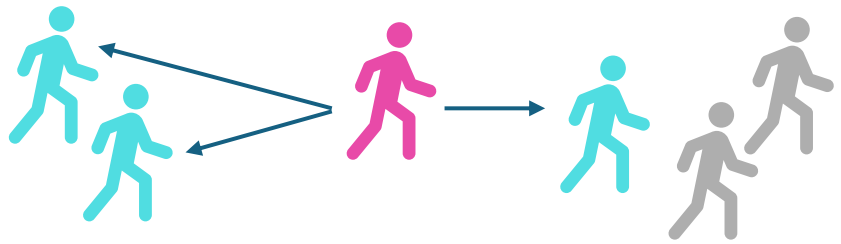
Local Outlier Factor

We calculate the density of points around an observation and compare it to the densities around nearby points.

$$N_k(p) = \text{k-neighbourhood of p}$$

$$\text{k} - \text{distance}(p) = \text{distance from } p \text{ to its k}^{\text{th}} \text{ nearest neighbour}$$

$$\text{Reachability Distance}(p, o) = \max(\text{k} - \text{distance}(o), d(p, o))$$



https://github.com/niamhmimnagh/imbalanced-data-course

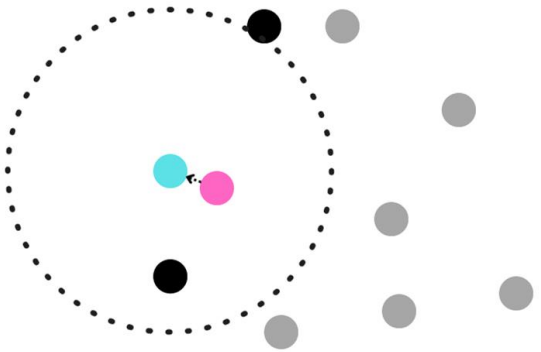
Local Outlier Factor

We calculate the density of points around an observation and compare it to the densities around nearby points.

$$N_k(p) = \text{k-neighbourhood of p}$$

$$\text{k} - \text{distance}(p) = \text{distance from } p \text{ to its k}^{\text{th}} \text{ nearest neighbour}$$

$$\text{Reachability Distance}(p, o) = \max(\text{k} - \text{distance}(o), d(p, o))$$



https://github.com/niamhmimnagh/imbalanced-data-course

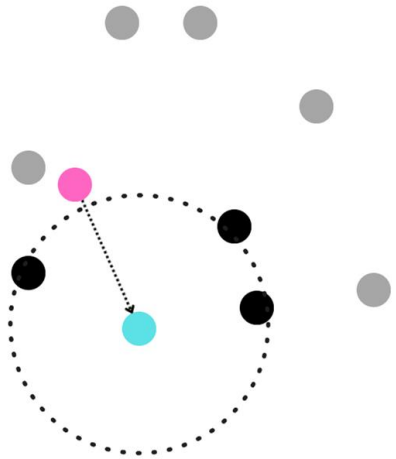
Local Outlier Factor

We calculate the density of points around an observation and compare it to the densities around nearby points.

$$N_k(p) = \text{k-neighbourhood of p}$$

$$\text{k} - \text{distance}(p) = \text{distance from } p \text{ to its k}^{\text{th}} \text{ nearest neighbour}$$

$$\text{Reachability Distance}(p, o) = \max(\text{k} - \text{distance}(o), d(p, o))$$



https://github.com/niamhmimnagh/imbalanced-data-course

Local Outlier Factor

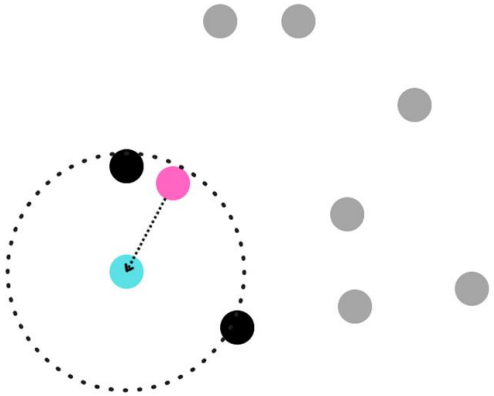
We calculate the density of points around an observation and compare it to the densities around nearby points.

$$N_k(p) = \text{k-neighbourhood of p}$$

$$\text{k} - \text{distance}(p) = \text{distance from } p \text{ to its k}^{\text{th}} \text{ nearest neighbour}$$

$$\text{Reachability Distance}(p, o) = \max(\text{k} - \text{distance}(o), d(p, o))$$

$$\text{Local Reachability Density}(p) = \left(\frac{\sum_{o \in N_k(p)} \text{Reachability Distance}(p, o)}{|N_k(p)|} \right)^{-1}$$



https://github.com/niamhmimnagh/imbalanced-data-course

Local Outlier Factor

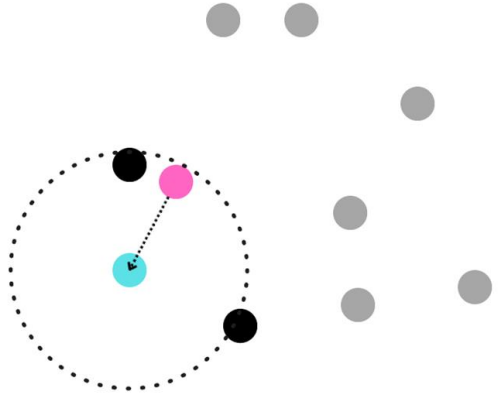
We calculate the density of points around an observation and compare it to the densities around nearby points.

$$N_k(p) = \text{k-neighbourhood of p}$$

$$\text{k} - \text{distance}(p) = \text{distance from } p \text{ to its k}^{\text{th}} \text{ nearest neighbour}$$

$$\text{Reachability Distance}(p, o) = \max(\text{k} - \text{distance}(o), d(p, o))$$

$$\text{Local Reachability Density}(p) = \left(\frac{\sum_{o \in N_k(p)} \text{Reachability Distance}(p, o)}{|N_k(p)|} \right)^{-1}$$



$$\text{Local Outlier Factor}(p) = \frac{\sum_{o \in N_k(p)} \frac{\text{Local Reachability Density}(o)}{\text{Local Reachability Density}(p)}}{|N_k(p)|}$$

https://github.com/niamhmimnagh/imbalanced-data-course

Local Outlier Factor

```
# Local Outlier Factor
library(DescTools)
lof_scores <- LOF(X_num, k = 150)
lof_outliers <- lof_scores >= quantile(lof_scores, 0.7)
lof_pred_bin <- as.factor(as.numeric(lof_outliers))
caret::confusionMatrix(lof_pred_bin, y_true_bin, positive = "1")
```

```
          Reference
Prediction   0   1
         0 515 185
         1 185 115
```

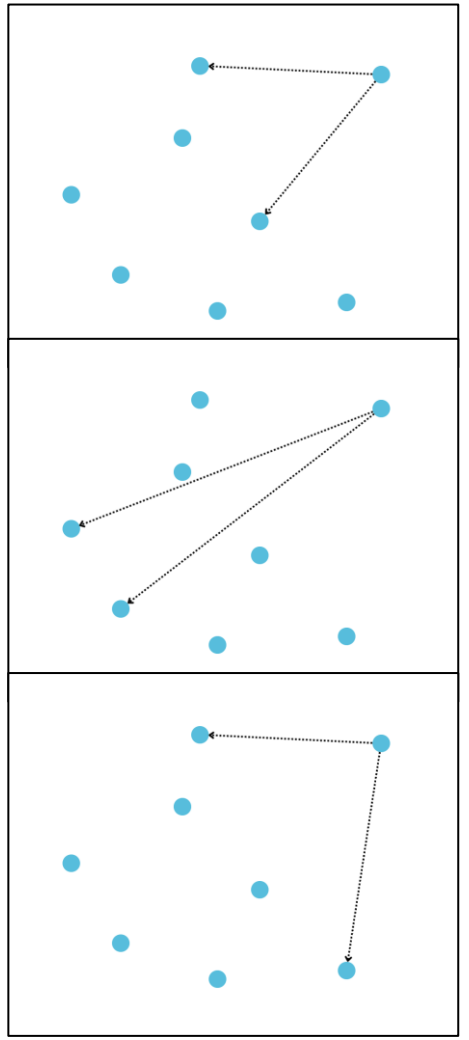
https://github.com/niamhmimnagh/imbalanced-data-course

Angle-Based Outlier Detection

- For each point, we calculate the angle that it creates with every other pair of points.

$$\text{Angle} - \text{based outlier factor}(A) = Var\left(\frac{\langle AB, AC\rangle}{||AB||^2 . ||AC||^2}\right)$$

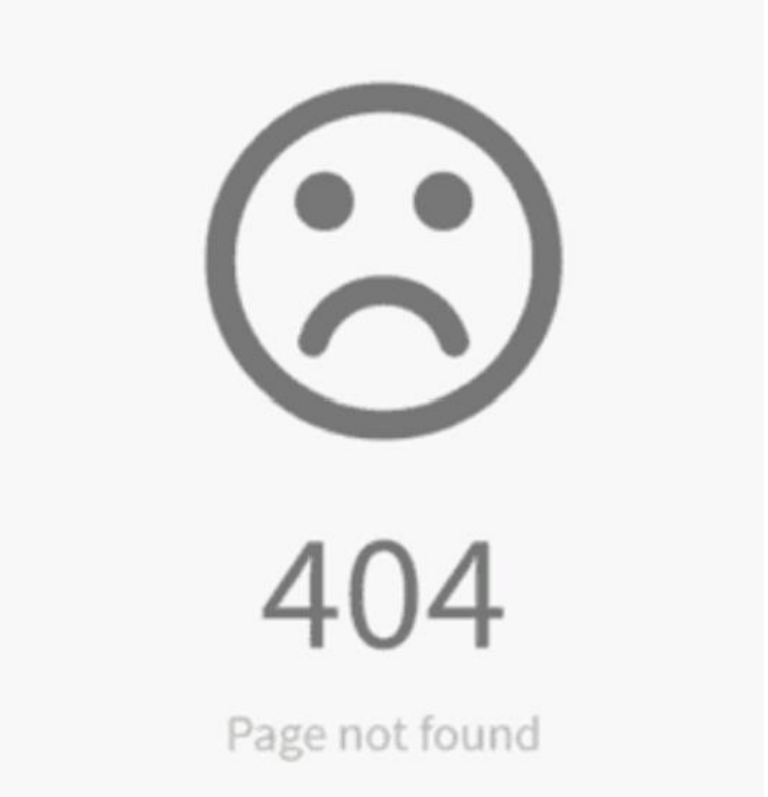
- A small angle-based outlier factor is indicative of an anomalous point.



https://github.com/niamhmimnagh/imbalanced-data-course

Angle-Based Outlier Detection

```
# ABOD
library(abodOutlier)
abod_scores <- abod(as.matrix(X_num))
abod_outliers <- abod_scores < quantile(abod_scores,
0.05)
```

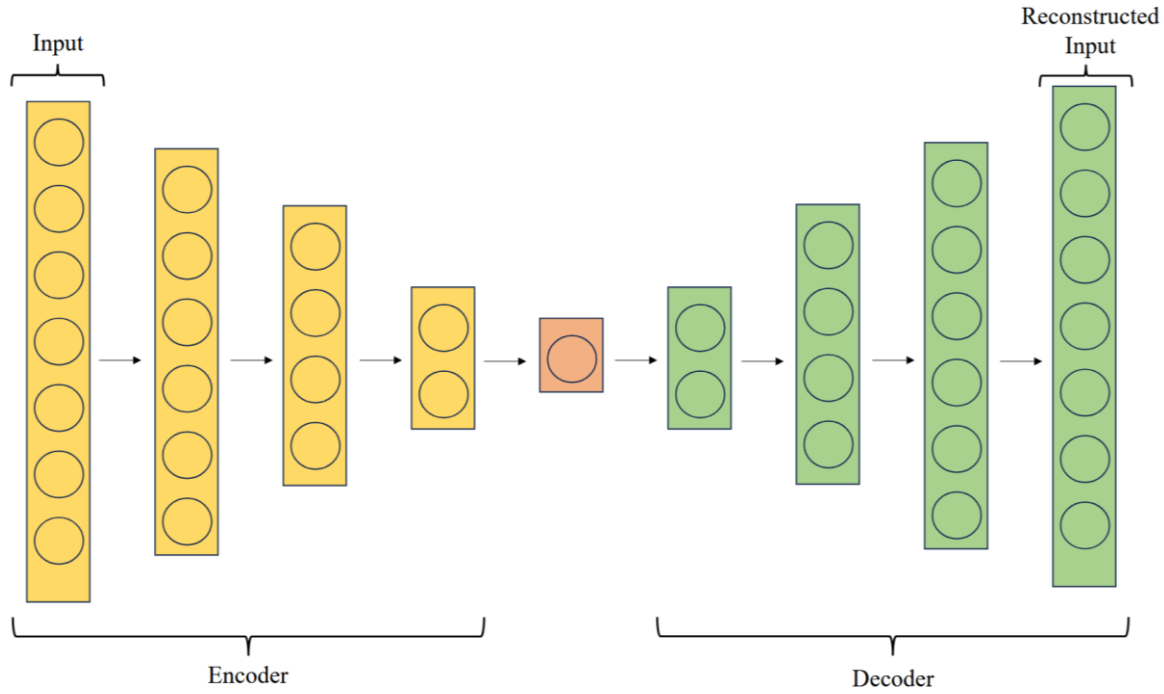


https://github.com/niamhmimnagh/imbalanced-data-course

Autoencoders

- Autoencoders are a type of neural network used to learn efficient, compressed representations of data

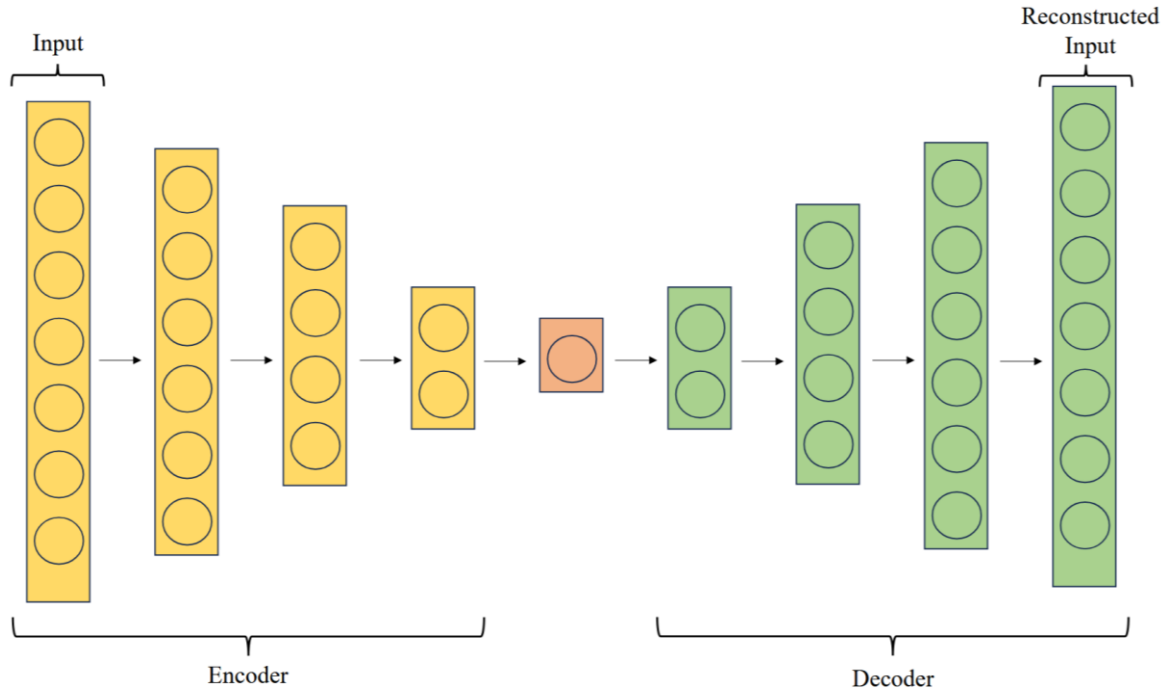
- They consist of two parts:
- The encoder: compresses the data into a latent representation.
- The decoder: reconstructs the input from the latent representation



https://github.com/niamhmimnagh/imbalanced-data-course

Autoencoders

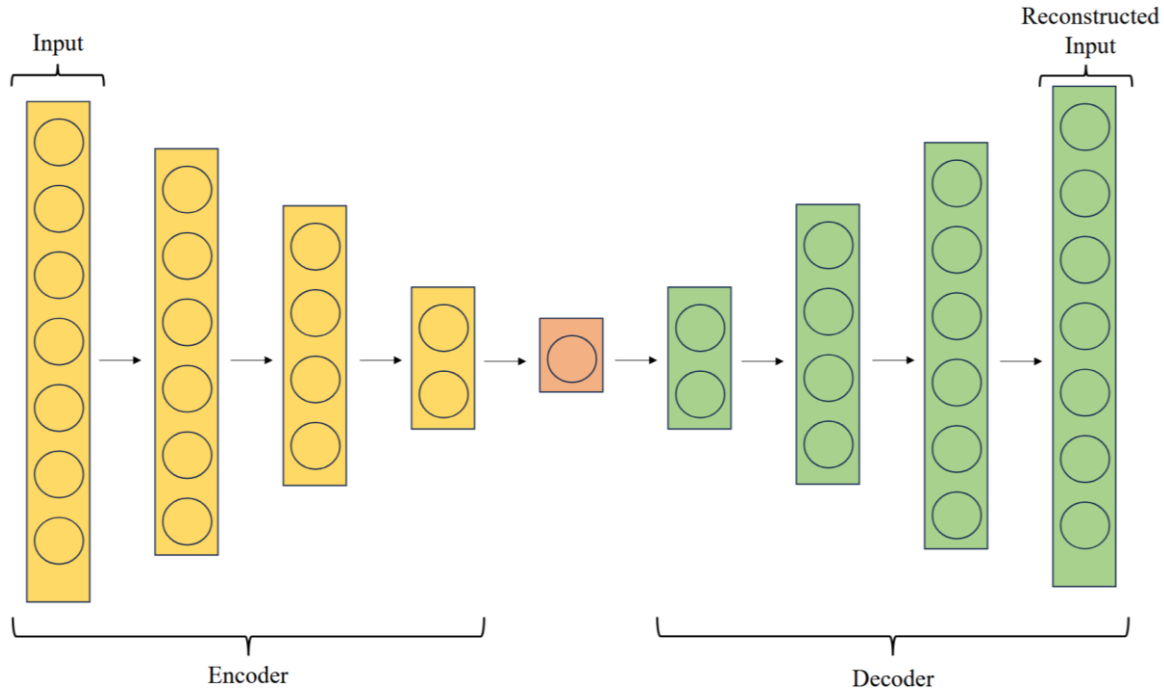
- Input Layer: Accepts the original data vector x.
- Encoder: A sequence of layers that progressively reduce the dimensionality of the input to a compact latent vector z.
- Latent Space (Bottleneck): The compressed representation capturing the most salient features.
- Decoder: A symmetric structure to the encoder that reconstructs the input from z.
- Output Layer: Produces $\hat{x}$, aiming for $\hat{x} \approx x$.



https://github.com/niamhmimnagh/imbalanced-data-course

Autoencoders

- Given input vector x, the encoder maps x to a lower-dimensional latent vector $z = f(x)$
- The decoder maps z back to a reconstruction $\hat{x} = g(z)$
- Anomaly Score: $||x - \hat{x}||$



https://github.com/niamhmimnagh/imbalanced-data-course

Autoencoders

- Each layer applies two key steps:

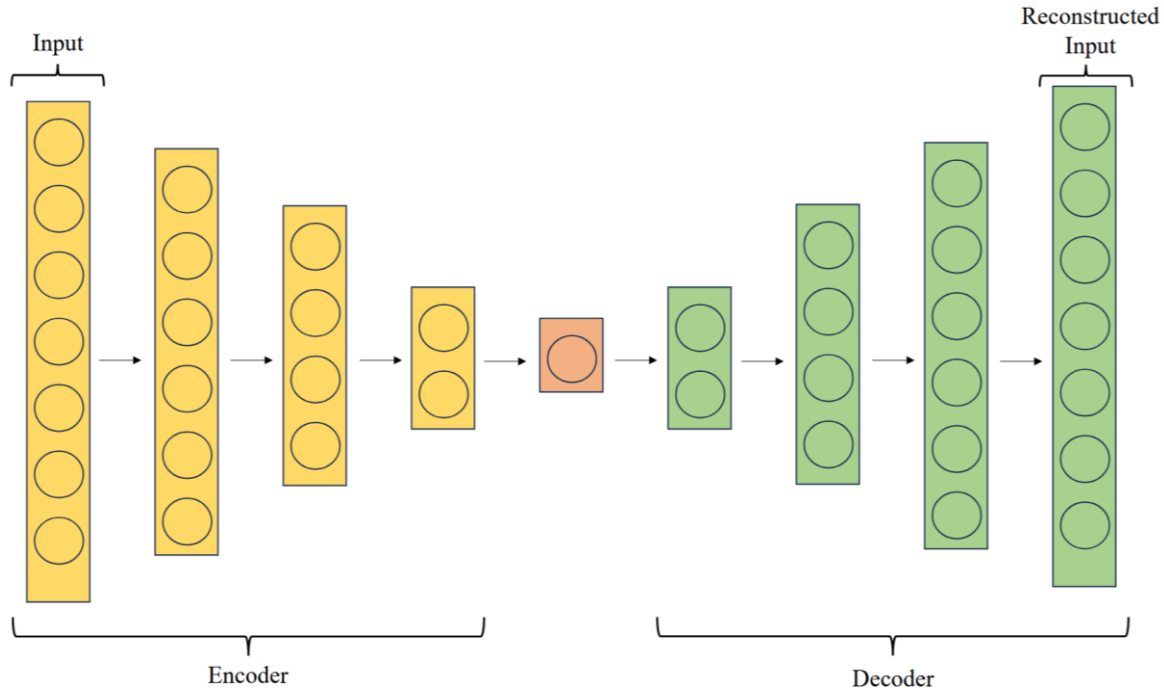
1. Linear transformation:

$$z = w^{\mathrm{T}}x + b$$

2. Non-linear activation function:

$$a = f(z),$$

- These steps allow the network to learn complex, non-linear mappings



https://github.com/niamhmimnagh/imbalanced-data-course

Autoencoders

- Activation functions introduce non-linearity to the network, enabling it to learn complex patterns.

- Common types:
  - ReLU (Rectified Linear Unit): max(0, x), fast and effective
  - Sigmoid: Maps input to (0, 1), useful when output needs to be bounded
  - Tanh: Maps input to (-1, 1), useful when centering data

Autoencoders

- The autoencoder learns to minimise the reconstruction error between the input and output.

- The anomaly score is based on the reconstruction error.
  A well-trained autoencoder will reconstruct normal samples well (low score), but anomalies will have poor reconstructions (high score).

- Outliers typically yield larger reconstruction errors because they differ significantly from the patterns the autoencoder has learned during training.

Autoencoders

- Advantages:
  - Can learn complex, non-linear representations of normal behaviour.
  - Flexible and powerful for high-dimensional data.

- Limitations:
  - Requires a sufficiently large dataset.
  - Sensitive to choice of architecture.
  - May learn to reconstruct anomalies if trained improperly or with contaminated data.

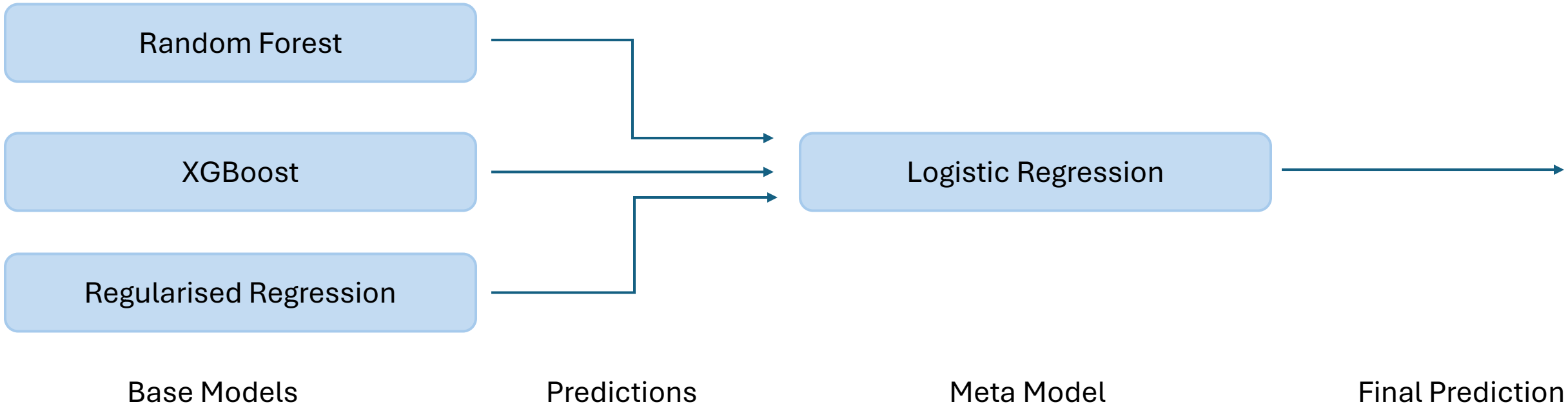
Ensemble Methods

- Ensemble methods combine predictions from multiple models to improve performance.
- They aim to reduce variance, bias, or model instability.
- Core idea: a group of weaker learners can form a strong learner.
- Common types:
  - Bagging (e.g., Random Forest) - reduces variance.
  - Boosting (e.g., XGBoost) - reduces bias.
  - Stacking - learns how to best combine different models.
- Often outperform individual models, especially on complex tasks.

Model Stacking

1. Train multiple base models on your training data.
2. Use their predictions to create a new dataset.
3. Train a meta-model (e.g., logistic regression) on this new dataset.
4. The meta-model learns to combine base model outputs for better performance.

Stacking Example



https://github.com/niamhmimnagh/imbalanced-data-course

Model Stacking

```
# Random Forest
rf_model <- randomForest(Class ~ ., data = train_data, ntree = 100)
rf_probs <- predict(rf_model, test_data %>% dplyr::select(-Class), type = "prob")[, "Bad"]

#  Elastic Net
cv_glmnet <- cv.glmnet(x_train, y_train, family = "binomial", alpha = 0.5)
enet_probs <- predict(cv_glmnet, newx = x_test, s = "lambda.min", type = "response")[, 1]

#  XGBoost
xgb_model <- xgboost(data = dtrain, nrounds = 50, objective = "binary:logistic", verbose =
0)
xgb_probs <- predict(xgb_model, dtest)

# Stacking Logistic Regression
stack_input <- data.frame(rf = rf_probs, enet = enet_probs, xgb = xgb_probs)
stack_y <- ifelse(test_data$Class == "Bad", 1, 0)
stack_model <- glm(stack_y ~ ., data = stack_input, family = "binomial")
stack_probs <- predict(stack_model, newdata = stack_input, type = "response")
```

Model Stacking

Model	Precision	Recall	F1
Random Forest	0.698	0.518	0.595
Elastic Net	0.541	0.776	0.638
XGBoost	0.638	0.6	0.618
Stacked Model	0.721	0.646	0.641

Final Thoughts

- Anomalies are rare by nature - and that rarity can distort traditional ML workflows.
- Choosing the right evaluation metrics (e.g., Precision, Recall, F1, PR-AUC) is critical - accuracy is not enough when the majority class dominates.
- Isolation-based methods like Isolation Forests and LOF excel at capturing *local irregularities* without labels.
- Domain knowledge can be as important as model choice - defining what *counts* as an anomaly is often subjective.
- Ensemble approaches can outperform single models by capturing different patterns of abnormality missed by any one algorithm.
- There is no perfect model for anomaly detection - but combining imperfect models strategically, with the right metrics and domain context, gets you close.

Thank You

https://github.com/niamhmimnagh/imbalanced-data-course

niamhmimnagh@gmail.com