CoGrammar

Welcome to this session:

Skills Bootcamp - Q&A Session

The session will start shortly...

Questions? Drop them in the chat.
We'll have dedicated moderators answering questions.

Skills Bootcamp Data Science Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. We will be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

Skills Bootcamp Data Science Housekeeping

- For all **non-academic questions**, please submit a query: ***www.hyperiondev.com/support***

- **Report a safeguarding incident: *www.hyperiondev.com/safeguardreporting***

- We would love your feedback on lectures: ***Feedback on Lectures.***

- Find all the lecture **content** in your **Lecture Backpack** on GitHub.

- If you are hearing impaired, kindly use your computer's function through Google chrome to enable captions.

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar HyperionDev

Skills Bootcamp Progression Overview

✅ Criterion 1 - Initial Requirements

Specific achievements within the first two weeks of the program.

To meet this criterion, students need to, by no later than **01 December 2024 (C11)** or **22 December 2024 (C12):**

- **Guided Learning Hours** (GLH): Attend a minimum of 7-8 GLH per week (lectures, workshops, or mentor calls) for a total minimum of **15 GLH**.
- **Task Completion:** Successfully complete the **first 4 of the assigned tasks**.

✅ Criterion 2 - Mid-Course Progress

Progress through the successful completion of tasks within the first half of the program.

To meet this criterion, students should, by no later than **12 January 2025 (C11)** or **02 February 2025 (C12):**

- **Guided Learning Hours** (GLH): Complete at least **60 GLH**.
- **Task Completion :** Successfully complete the **first 13 of the assigned tasks**.

CoGrammar

Skills Bootcamp Progression Overview

✅ Criterion 3 – End-Course Progress

Showcasing students' progress nearing the completion of the course.

To meet this criterion, students should:

- **Guided Learning Hours** (GLH)**:** Complete the **total minimum required GLH,** by the **support end date.**
- **Task Completion : Complete all mandatory tasks**, including any necessary resubmissions, by the end of the bootcamp, **09 March 2025 (C11)** or **30 March 2025 (C12).**

✅ Criterion 4 - Employability

Demonstrating progress to find employment.

To meet this criterion, students should:

- **Record an Interview Invite:** Students are required to record proof of invitation to an interview by **30 March 2025 (C11)** or **04 May 2025 (C12)**.
  - **South Holland Students** are required to proof and interview by **17 March 2025.**
- **Record a Final Job Outcome :** Within 12 weeks post-graduation, students are required to record a job outcome.

CoGrammar

Is there a specific topic from this week that you'd like to review or gain more clarity on?

CoGrammar

Any more questions on HPC?

CoGrammar

Learning Outcomes

- ❖ **Define image processing**.
- ❖ **Describe the key steps** involved in preparing image data for machine learning algorithms, including loading, reshaping, and vectorising images.
- ❖ Apply machine learning algorithms, such as **Random Forest Classifiers**, to image classification tasks using Python libraries like scikit-learn.
- ❖ Evaluate the performance of image classification models using appropriate metrics, such as **accuracy, precision, recall, and F1-score**.
- ❖ Analyse the **impact of hyperparameter tuning** on model performance in image classification tasks.
- ❖ Employ techniques like **grid search to optimise hyperparameters.**

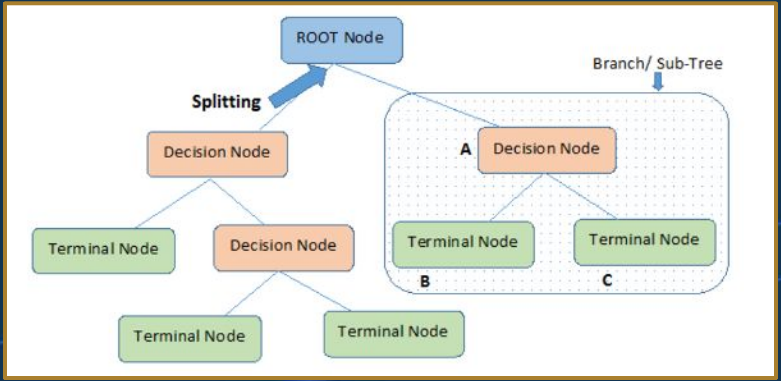
Decision Trees and Random Forests

Recap

CoGrammar

Decision Trees

- Decision trees can be used for image **classification tasks**
- Each internal node represents a feature (pixel intensity), branches represent decisions
- Leaf nodes represent **class labels** (e.g., digits in MNIST)
- **Advantages:** Interpretable, can handle non-linear relationships
- **Disadvantages:** Prone to overfitting, may not generalise well to unseen images



CoGrammar

Random Forests

- Random Forests are an **ensemble of decision trees**, well-suited for image classification
- Each tree is **trained on a random subset of features** (pixels) and samples (images)
- Predictions are made by **aggregating the outputs of individual trees** (majority vote)
- **Advantages:** Reduces overfitting, improves generalisation, provides feature importance

Image Processing

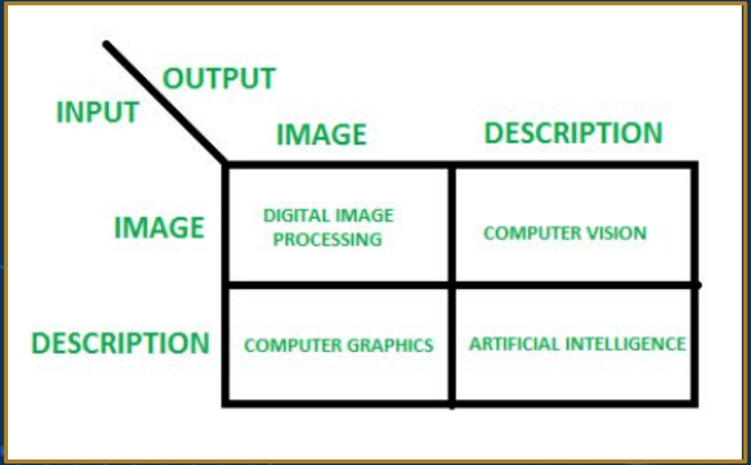
CoGrammar

Image Processing

- Image processing refers to the **manipulation and analysis of digital images** to **extract meaningful information** and enhance visual content.

Importance

- Enables computers to **interpret and understand visual data**.
- Facilitates the development of intelligent systems capable of **automating tasks related to visual perception**.

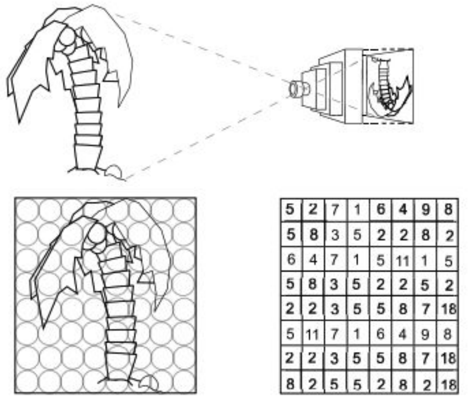


INPUT \ OUTPUT	IMAGE	DESCRIPTION
IMAGE	DIGITAL IMAGE PROCESSING	COMPUTER VISION
DESCRIPTION	COMPUTER GRAPHICS	ARTIFICIAL INTELLIGENCE

Source: GeeksForGeeks

Key Concepts

- **Digital Image:** A digital image is represented as a two-dimensional array of pixels, where each pixel contains intensity or colour values.



5	2	7	1	6	4	9	8
5	8	3	5	2	2	8	2
6	4	7	1	5	11	1	5
5	8	3	5	2	2	5	2
2	2	3	5	5	8	7	18
5	11	7	1	6	4	9	8
2	2	3	5	5	8	7	18
8	2	5	5	2	8	2	18

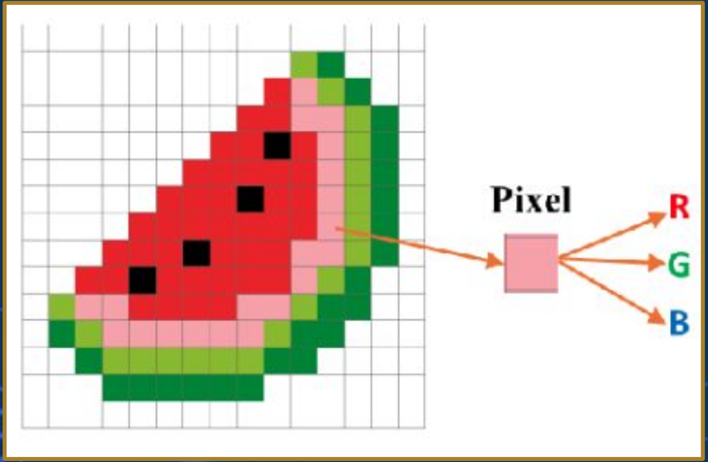
Figure 1.1: Capturing a 2D Continuous Image of a Scene

Source: [Clemson University]()

CoGrammar

Key Concepts

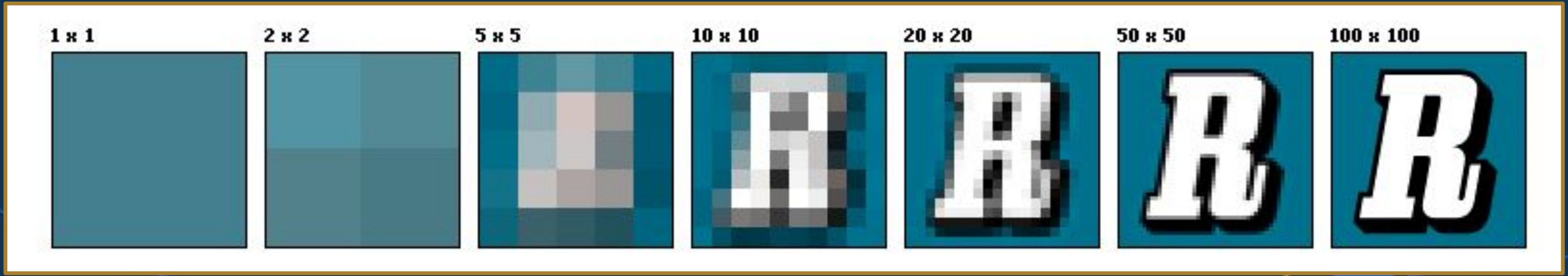
- **Pixel:** A pixel is the smallest unit of a digital image and is represented by **one or more numerical values**. In grayscale images, each pixel has a single intensity value, while in colour images, pixels typically have three values corresponding to the **red, green, and blue** colour channels.



Source: [ResearchGate]()

Key Concepts

- **Image Resolution:** Image resolution refers to the number of pixels in an image, usually expressed in terms of **width and height** (e.g., 1024x768). Higher resolution images contain more pixels and provide **more detailed and clearer representations** of the visual content.



Source: Wikipedia

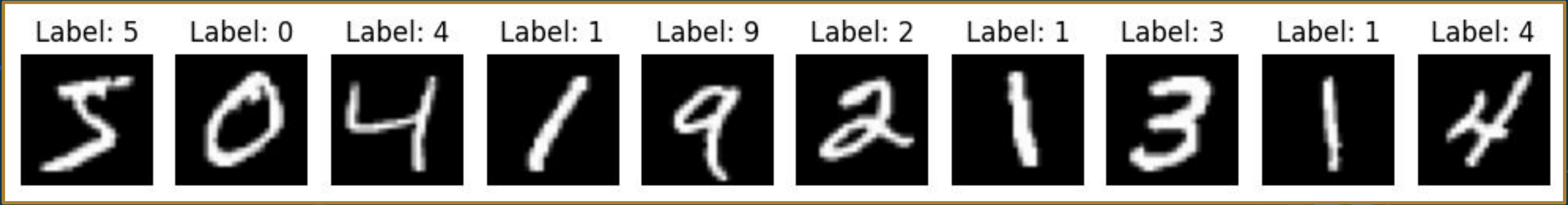
CoGrammar

Detecting Handwritten Digits

CoGrammar

MNIST

- The MNIST dataset is a **widely-used benchmark dataset** in the field of machine learning, consisting of a large collection of **grayscale images of handwritten digits (0-9)**.
- The dataset contains 70,000 images, split into **60,000 training images and 10,000 testing images**. Each image has a size of **28x28 pixels**.

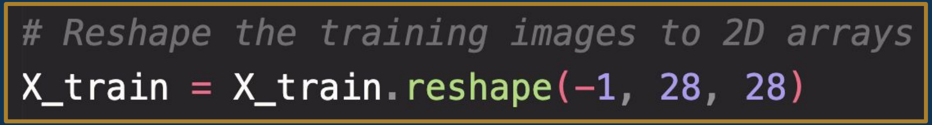


CoGrammar

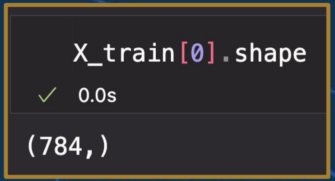
Reshaping Images

- The loaded image data is often flattened into a **1D array** for storage and processing purposes. However, for applying image processing techniques and visualisation, **it is necessary to reshape the data back into a 2D array representing the original image dimensions.**

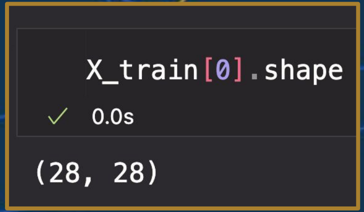
- The -1 parameter in the reshape() function automatically calculates the number of images based on the total number of pixels.



```python
# Reshape the training images to 2D arrays
X_train = X_train.reshape(-1, 28, 28)
```



```
X_train[0].shape
✓ 0.0s

(784,)
```



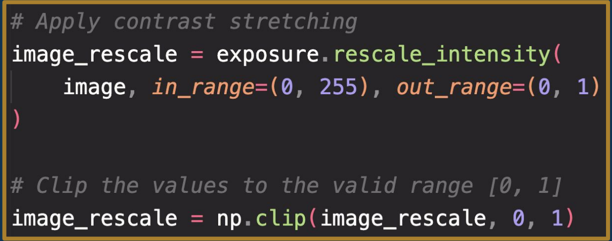
```
X_train[0].shape
✓ 0.0s

(28, 28)
```

Preprocessing

- Preprocessing techniques are applied to enhance the **quality of images**, **normalise pixel values**, and **remove any artifacts or noise** that may affect the performance of machine learning algorithms.

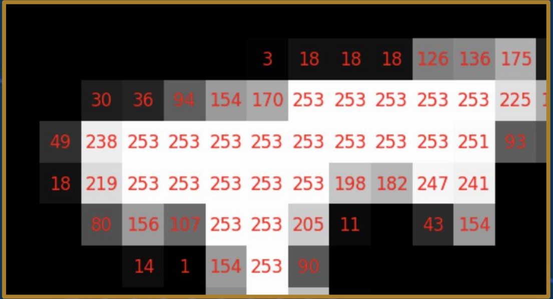
Preprocessing

❖ Contrast stretching improves the contrast of an image by **spreading out the pixel intensities**. It can be achieved using the exposure module from the scikit-image library.

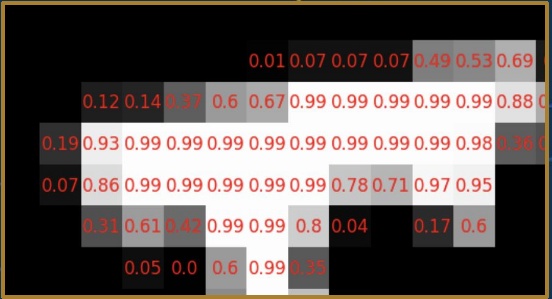


```python
# Apply contrast stretching
image_rescale = exposure.rescale_intensity(
    image, in_range=(0, 255), out_range=(0, 1)
)

# Clip the values to the valid range [0, 1]
image_rescale = np.clip(image_rescale, 0, 1)
```

This code calculates the 2nd and 98th percentile of pixel intensities and rescales the image intensity values to the range [0, 1] **(normalisation)**.



					3	18	18	18	126	136	175
	30	36	94	154	170	253	253	253	253	253	225
49	238	253	253	253	253	253	253	253	253	251	93
18	219	253	253	253	253	253	198	182	247	241	
	80	156	107	253	253	205	11		43	154	
		14	1	154	253	90					



					0.01	0.07	0.07	0.07	0.49	0.53	0.69
	0.12	0.14	0.37	0.6	0.67	0.99	0.99	0.99	0.99	0.99	0.88
0.19	0.93	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.98	0.36
0.07	0.86	0.99	0.99	0.99	0.99	0.99	0.78	0.71	0.97	0.95	
	0.31	0.61	0.42	0.99	0.99	0.8	0.04		0.17	0.6	
		0.05	0.0	0.6	0.99	0.35					

CoGrammar

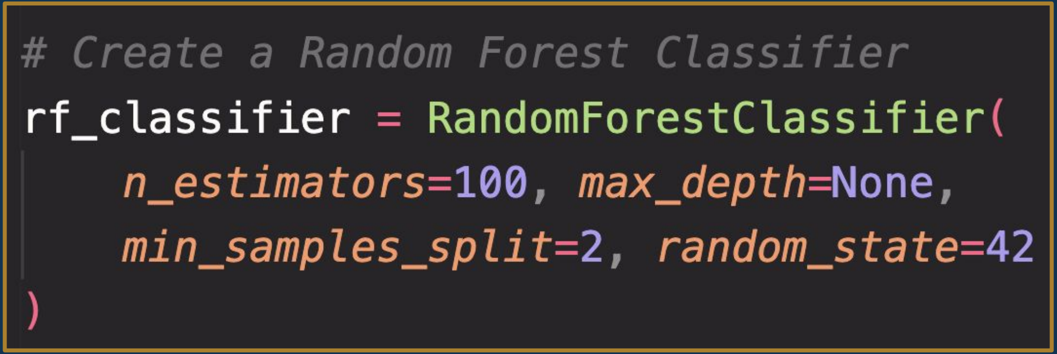
Random Forest Classifier

- The Random Forest Classifier is an ensemble learning algorithm that combines multiple decision trees to make robust and accurate predictions. It is particularly well-suited for image classification tasks.

- The Random Forest Classifier has **several hyperparameters that can be tuned to optimise its performance, such as the number of trees (n_estimators), the maximum depth of each tree (max_depth), and the minimum number of samples required to split an internal node (min_samples_split)** → More on this at the end.

Training

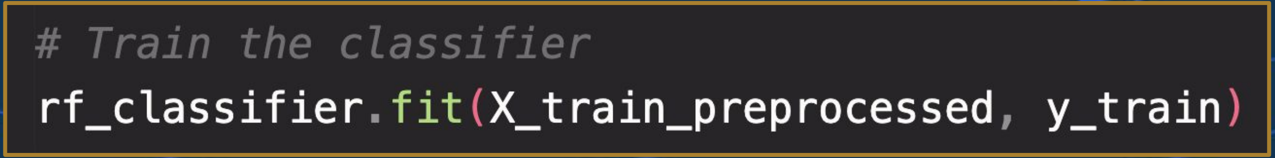
- To train the Random Forest Classifier, we first create an instance of the RandomForestClassifier class from the scikit-learn library, specifying the desired hyperparameters.



```
# Create a Random Forest Classifier
rf_classifier = RandomForestClassifier(
    n_estimators=100, max_depth=None,
    min_samples_split=2, random_state=42
)
```

Training

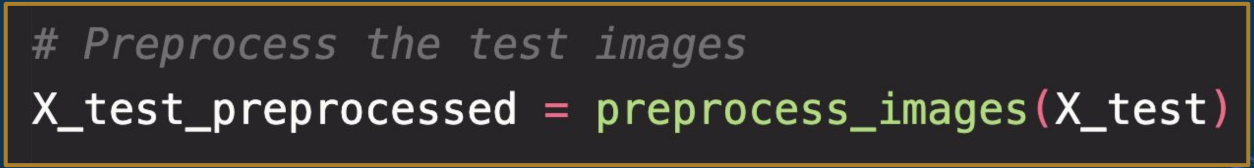
- Next, **we fit the classifier to the preprocessed training data using the fit() method**. This involves providing the feature matrix (X_train_preprocessed) and the corresponding labels (y_train) to the classifier.
- During the training process, the Random Forest Classifier learns the underlying patterns and **relationships between the input features (pixel values) and the corresponding labels (digit classes)**.



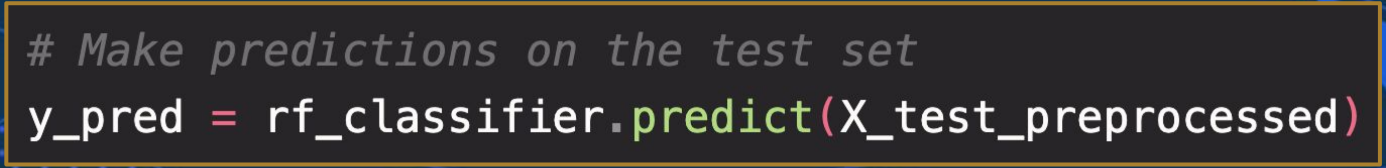
```
# Train the classifier
rf_classifier.fit(X_train_preprocessed, y_train)
```

Making Predictions

- Once the Random Forest Classifier is trained, we can use it to make predictions on **new, unseen images**.
- To make predictions, we first **preprocess the test images in the same way as the training images**, applying the necessary reshaping and preprocessing steps.



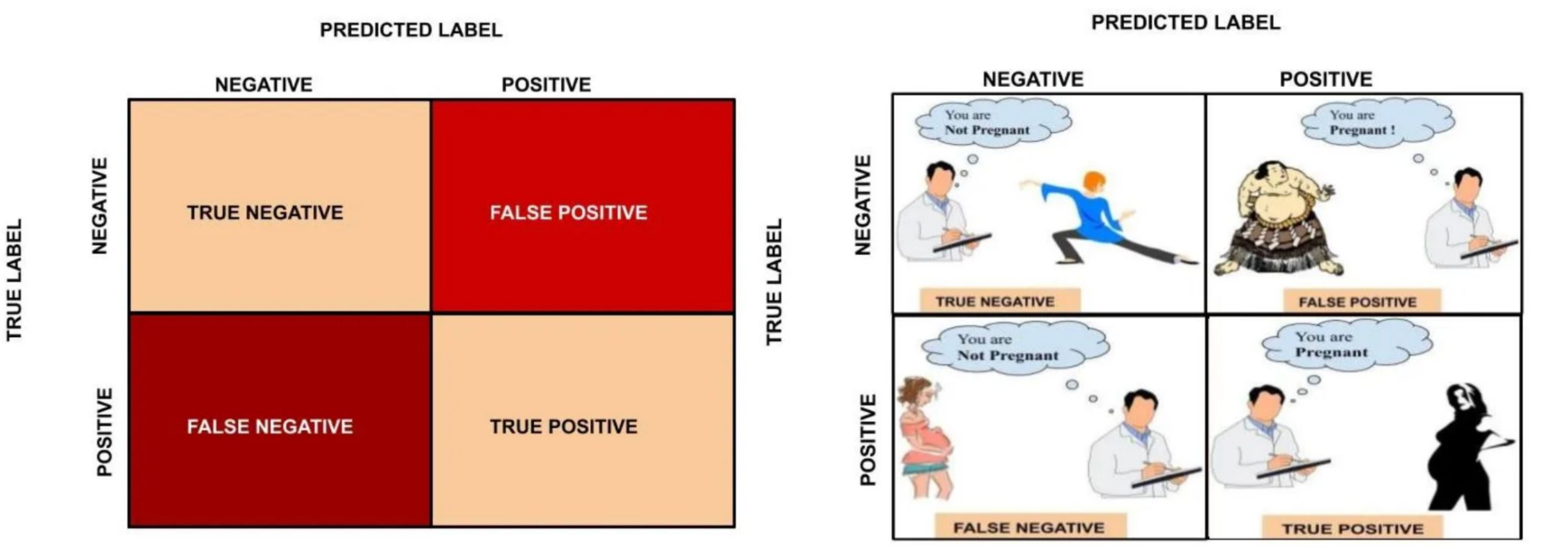
```
# Preprocess the test images
X_test_preprocessed = preprocess_images(X_test)
```



```
# Make predictions on the test set
y_pred = rf_classifier.predict(X_test_preprocessed)
```

Evaluation

Some Context



		PREDICTED LABEL	
		NEGATIVE	POSITIVE
TRUE LABEL	NEGATIVE	TRUE NEGATIVE	FALSE POSITIVE
	POSITIVE	FALSE NEGATIVE	TRUE POSITIVE

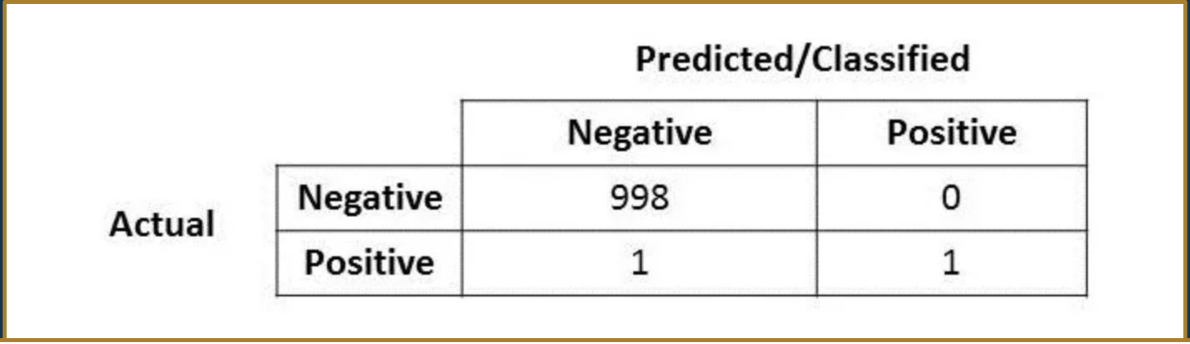
Figure 5: Confusion Matrix.

Source: Medium

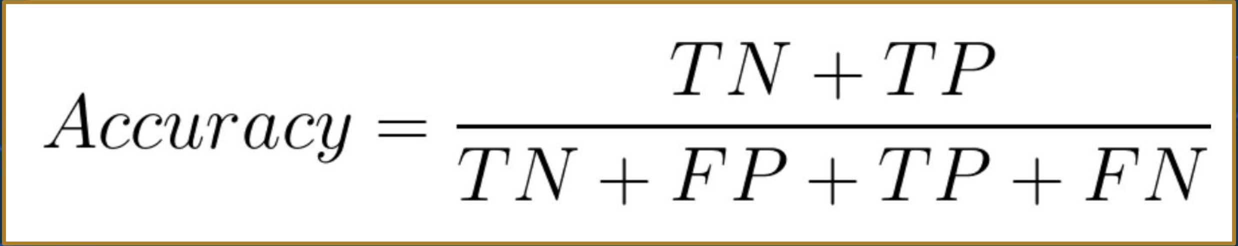
Evaluation Metrics

- **Accuracy:** Accuracy measures the overall correctness of the model's predictions by calculating the **proportion of correctly classified samples out of the total number of samples**.

Source: Medium



		Predicted/Classified	
		Negative	Positive
Actual	Negative	998	0
	Positive	1	1

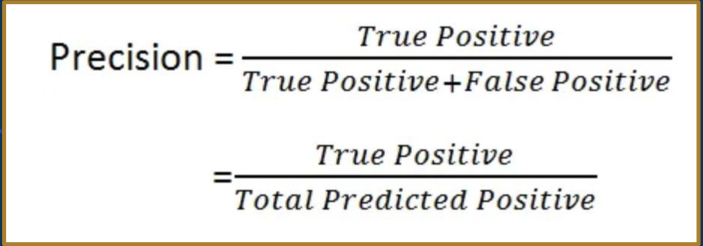


$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN}$$

CoGrammar

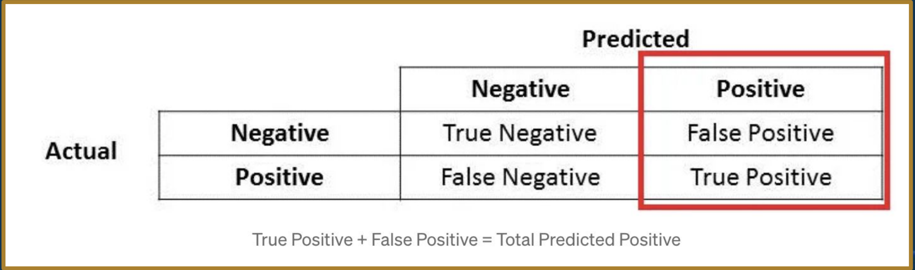
Evaluation Metrics

- **Precision:** Precision focuses on the quality of the model's positive predictions. It is calculated as the ratio of true positive predictions to the total number of positive predictions (true positives + false positives). **High precision indicates that when the model predicts a specific class, it is more likely to be correct.**



$$\text{Precision} = \frac{\textit{True Positive}}{\textit{True Positive} + \textit{False Positive}}$$

$$= \frac{\textit{True Positive}}{\textit{Total Predicted Positive}}$$



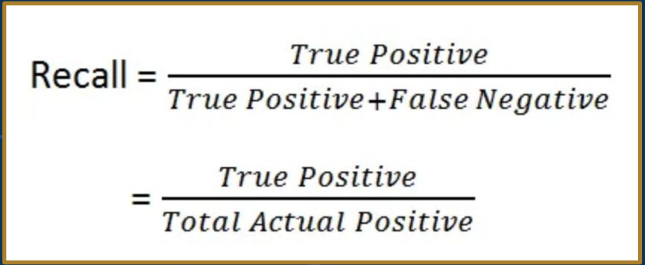
		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

True Positive + False Positive = Total Predicted Positive

Source: Medium

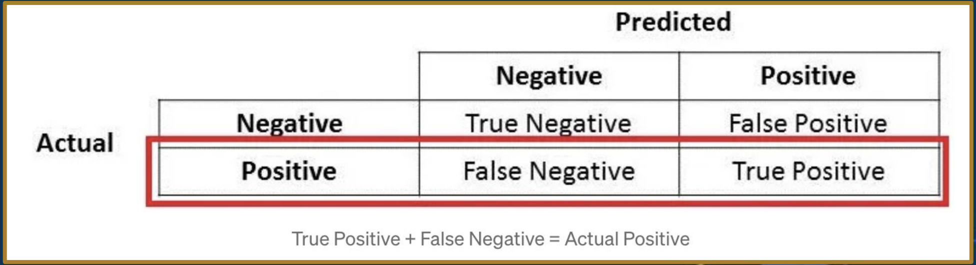
Evaluation Metrics

- **Recall:** Recall, also known as sensitivity or true positive rate, measures the model's ability to correctly identify positive instances. It is calculated as the ratio of true positive predictions to the total number of actual positive instances (true positives + false negatives). **High recall indicates that the model is able to capture a large proportion of the positive instances.**



$$\text{Recall} = \frac{\mathit{True\ Positive}}{\mathit{True\ Positive} + \mathit{False\ Negative}}$$

$$= \frac{\mathit{True\ Positive}}{\mathit{Total\ Actual\ Positive}}$$



		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

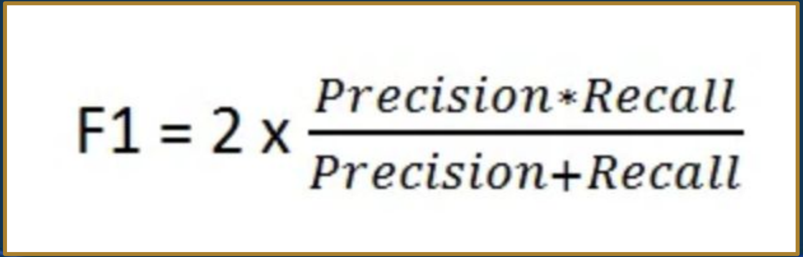
True Positive + False Negative = Actual Positive

Source: Medium

CoGrammar

Evaluation Metrics

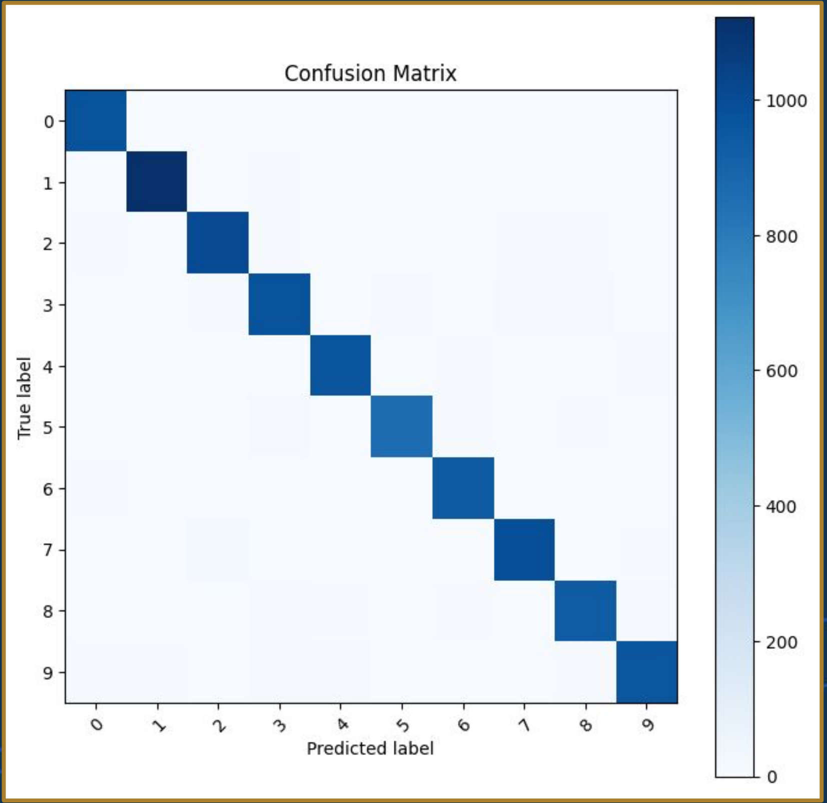
- **F1-score:** The F1-score is the harmonic mean of precision and recall, providing a balanced measure that considers both the quality and completeness of the model's predictions. **It is especially useful when dealing with imbalanced datasets or when equal importance is given to precision and recall.**



$$F1 = 2 \text{ x } \frac{Precision * Recall}{Precision + Recall}$$

Confusion Matrix

- A confusion matrix is a powerful tool for visualising the **performance of a classification model** by providing a tabular summary of the model's predictions compared to the true labels.



CoGrammar

Hyperparameter Tuning

CoGrammar

Importance

- Hyperparameters are the **adjustable settings** of a machine learning algorithm that are **not learned from the data but are set by the user before training**. These hyperparameters can significantly impact the performance and behaviour of the model.

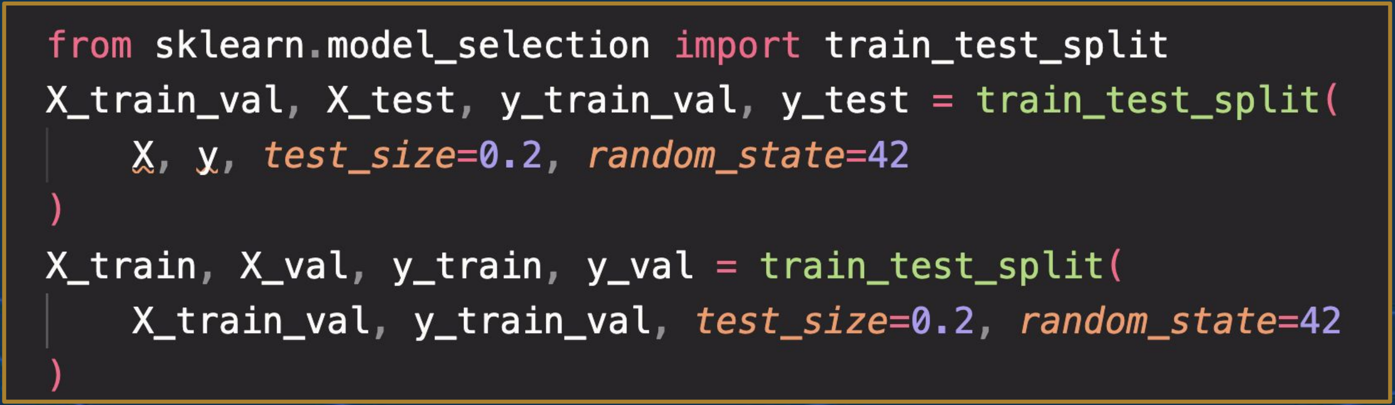
- Proper tuning of hyperparameters is crucial for optimising the model's performance and achieving the best possible results. It involves **finding the right combination of hyperparameter values that strike a balance between model complexity and generalisation ability**.

Data Splits

- To properly tune hyperparameters and evaluate the model's performance, it is essential to split the data into three sets: **training, validation, and test**.
- The training set is used to **train the model**, the validation set is used to **tune the hyperparameters and assess the model's performance during the tuning process**, and the test set is used to **evaluate the final model's performance on unseen data.**

Data Splits

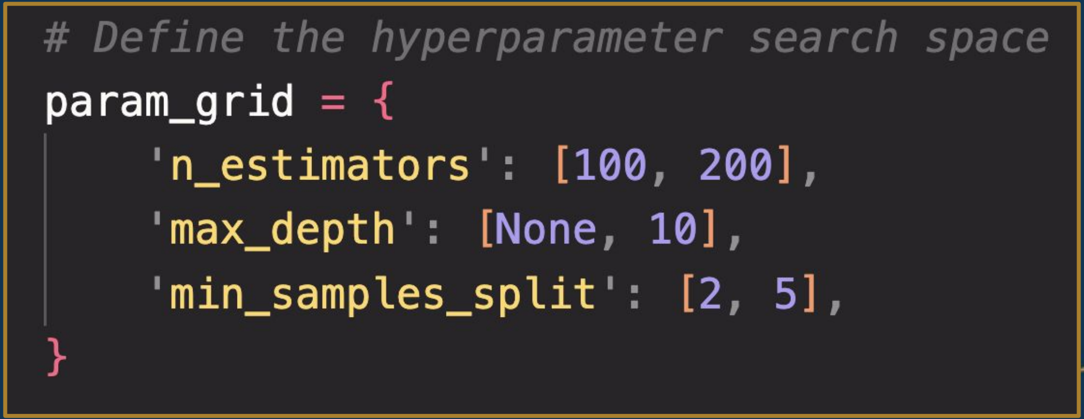
- In this example, we first split the data into training/validation (X_train_val, y_train_val) and test (X_test, y_test) sets using a 80-20 split. Then, we further split the training/validation set into training (X_train, y_train) and validation (X_val, y_val) sets using another 80-20 split.



```python
from sklearn.model_selection import train_test_split
X_train_val, X_test, y_train_val, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
X_train, X_val, y_train, y_val = train_test_split(
    X_train_val, y_train_val, test_size=0.2, random_state=42
)
```

Process

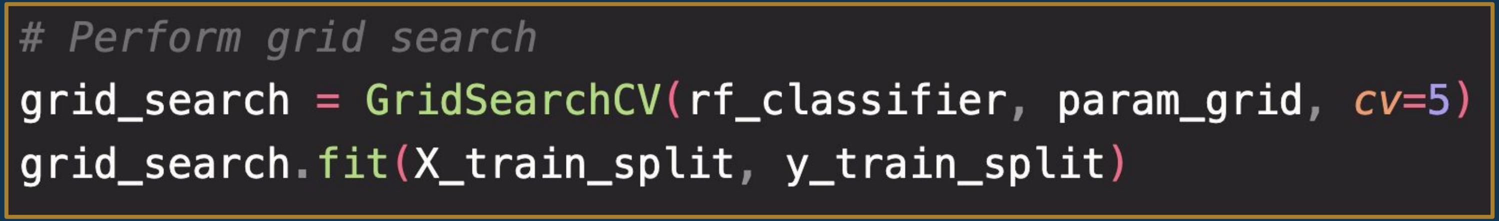
- Define the hyperparameter search space



```
# Define the hyperparameter search space
param_grid = {
    'n_estimators': [100, 200],
    'max_depth': [None, 10],
    'min_samples_split': [2, 5],
}
```

Process

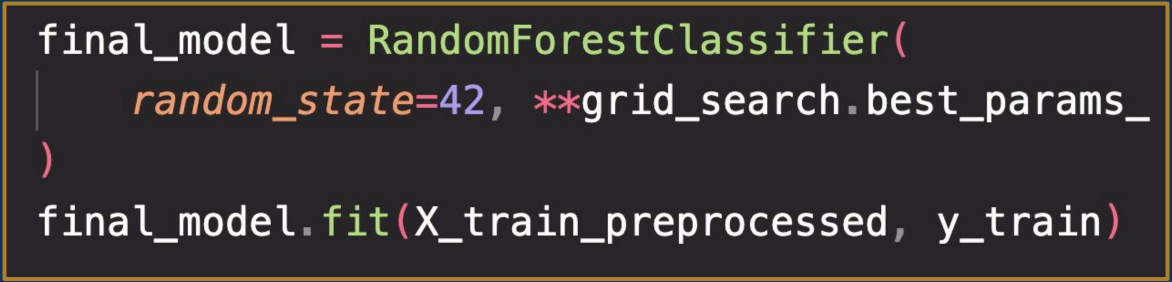
- Create an instance of the model and perform a grid search



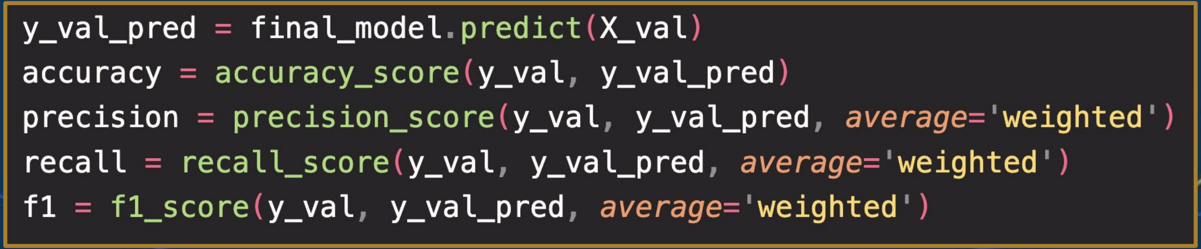
```
# Perform grid search
grid_search = GridSearchCV(rf_classifier, param_grid, cv=5)
grid_search.fit(X_train_split, y_train_split)
```

Process

- ❖ Evaluate the model's performance on the validation set



```python
final_model = RandomForestClassifier(
    random_state=42, **grid_search.best_params_
)
final_model.fit(X_train_preprocessed, y_train)
```



```python
y_val_pred = final_model.predict(X_val)
accuracy = accuracy_score(y_val, y_val_pred)
precision = precision_score(y_val, y_val_pred, average='weighted')
recall = recall_score(y_val, y_val_pred, average='weighted')
f1 = f1_score(y_val, y_val_pred, average='weighted')
```

Overfitting

- If the model achieves **high accuracy on the training set but performs poorly on the validation set**, it indicates overfitting.
- Example:
  - **Training Accuracy: 0.98**
  - **Validation Accuracy: 0.75**
- In this case, the model is **too complex** and has learned noise or specific patterns from the training data that do not generalise well to unseen data.
- To mitigate overfitting, you can try reducing the model complexity

Good Fit

- If the model achieves **high accuracy on both the training and validation sets**, it indicates a good fit.
- Example:
  - **Training Accuracy: 0.95**
  - **Validation Accuracy: 0.93**
- In this case, the model has found a **good balance between capturing the relevant patterns in the data and generalising well to unseen data**.
- You can proceed with evaluating the model on the test set to assess its final performance.

Questions and Answers

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar