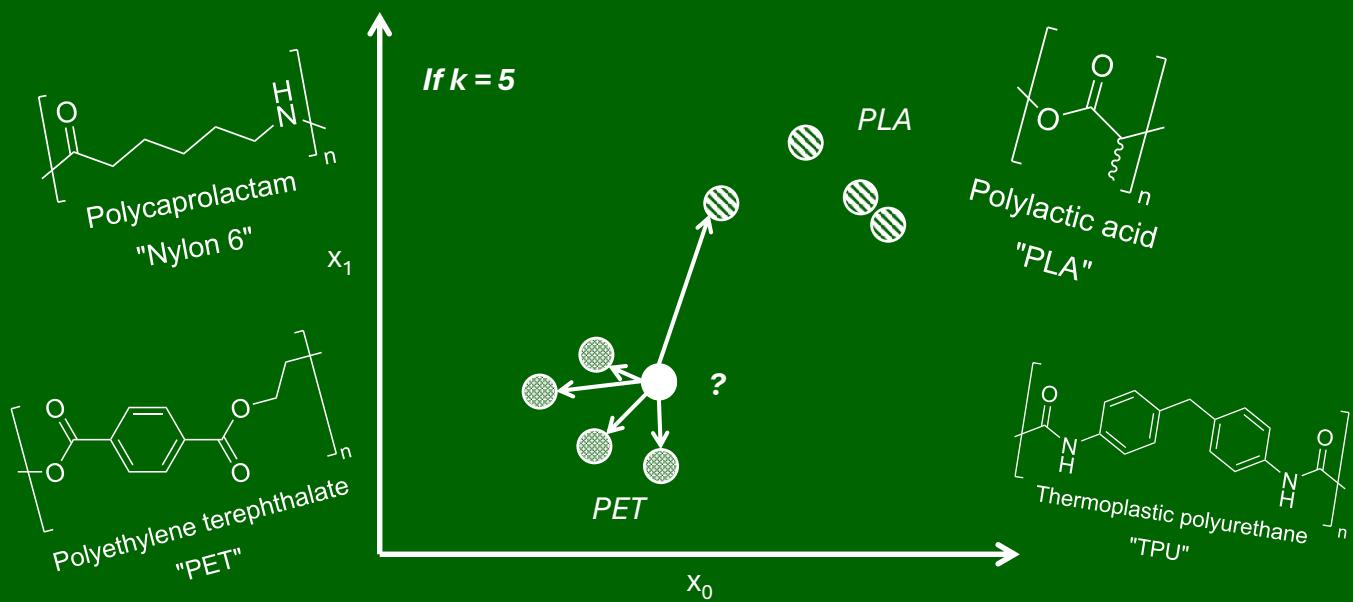


ID1:

Informing Chemical Recycling Strategies using AI-Aided Polymer Identification

By the end of this lab, you should be able to:

1. Develop and implement machine learning models.
2. Evaluate data quality and machine learning model performance.
3. Utilise generative AI as a tool-building resource.
4. Design chemical recycling processes.
5. Integrate computational tools with practical chemistry.
6. Critically analyse and reflect on laboratory work.



Contents

1.	Brief Introduction.....	4
1.1.	GenAI Acknowledgement.....	4
2.	Contextual Background.....	5
2.1	Polluting Polymers.....	5
2.2	Challenges in Polymer Waste Management	6
2.2.1	Polymer Identification	6
2.3	Polymer Recycling	6
2.4	Global Disparities in Waste Management.....	8
2.5	Technological and Policy Gaps	8
2.6	Potential Solutions to Polymer Waste Management	8
3.	Aim and Objectives	10
3.1	Practical Objectives.....	10
4.	Learning Outcomes	11
5.	Experimental Section: Getting Started.....	12
6.	Section A: Identifying a Suitable Chemical Recycling Route	12
6.1	Polymer Allocation and Research	12
7.	Section B: Building an AI Model for Identifying the Target Polymer.....	16
7.1	Activity 1: Building a Polymer Reference Database	16
7.1.1	Task 1.01: Switching on PlasTell	16
7.1.2	Task 1.02: Connecting PlasTell to the Matoha App	16
7.1.3	Task 1.03: Creating and Managing Spectral Collections	17
7.1.4	Task 1.04: Measuring & Saving Samples.....	17

IMPERIAL

7.1.5	Task 1.06: Exporting Data.....	18
7.2	Activity 2: Exploring your Data	19
7.2.1	Task 2.01: Counting Spectra & Polymers from your .CSV File	20
7.2.2	Task 2.02: Filtering and Saving Specific Data	21
7.2.3	Task 2.03: Transforming Spectrophotometer Data	22
7.2.4	Task 2.04: Combining Datasets	24
7.2.5	Task 2.05: Counting Columns by Polymer	27
7.2.6	Task 2.06: Looking for Duplicates	28
7.2.7	Task 2.07: Checking for Invalid Values in Transformed Data	31
7.3	Activity 3: Visualising Your Data	32
7.3.1	Task 3.01: Visualisation of Central Tendency & Dispersion	33
7.3.2	Task 3.02: Using Heatmaps to Explore Spectra Variation	35
7.4	Activity 4: Classifying Your Data using a Machine Learning Model.....	37
7.4.1	Task 4.01: Spectral Data Classification Using k-Nearest Neighbors (kNN)	38
7.4.2	Task 4.02: Building a Weighted-kNN Model	42
7.4.3	Task 4.03: Extension.....	44
7.5	Activity 5: Identifying Your Unknown Polymers	44
8.	Section C: Carrying out Chemical Polymer Recycling	46
9.	Ancillary: Methods of Assessment	48

1. Brief Introduction

This lab provides you with a hands-on learning experience that integrates data science principles with the critical environmental challenge of plastic waste management. Over two weeks, you will engage in an interdisciplinary project focused on the identification and subsequent chemical recycling of a common polymer.

The first phase will require you to work in collaboration with your partner to design a sustainable depolymerisation method for your assigned polymer. You will each compile a full risk assessment and experimental protocol which will be submitted prior to starting the lab (see **Section A**).

The second phase of the lab focuses on identifying and classifying various polymers using a Near-Infrared (NIR) spectrophotometer (see **Section B**). You will work in pairs to build a reference database of NIR spectra for known polymers and then apply machine learning algorithms, such as k-Nearest Neighbors (kNN), using Python to train models for accurately identifying unknown polymers. This phase introduces key concepts in data evaluation, machine learning, and AI, highlighting the importance of data quality and computational tools in solving real-world problems.

In the third and final phase, you will shift from data science to synthetic chemistry. You will implement your selected chemical recycling process for the polymer you have identified in week one (see **Section C**). This phase reinforces practical polymer chemistry skills and encourages critical thinking about the sustainability of chemical processes.

By the end of the lab, you will gain a comprehensive understanding of polymer identification and recycling, integrating skills in both data science and chemical synthesis.

1.1. GenAI Acknowledgement

This lab is designed to integrate GenAI as a key tool in the development and implementation of ML code. In accordance with departmental policy, this project

acknowledges the essential use of GenAI software (ChatGPT 4.0, Claude 3.5 Sonnet, and Gemini 1.5 Flash) in the design and creation of the lab protocol.

2. Contextual Background

2.1 Polluting Polymers

Polymer pollution is [a significant global issue](#), with far-reaching environmental, societal, and economic consequences. Polymers, both organic (such as [plastics](#)) and inorganic (such as [polysiloxanes](#)), are produced in massive quantities, [exceeding 500 million metric tons of plastics](#) alone in 2024. The broad utility of polymers in various industries, from packaging and construction to electronics and medical devices, contributes to their extensive use and subsequent waste generation. However, only a small fraction of this waste, [around 9% for plastics](#), is recycled, with the majority ending up in landfills, oceans, and other ecosystems.

The [persistence of polymers](#) in the environment is particularly concerning. Organic polymers like plastics can take centuries to degrade, leading to the accumulation of waste in natural habitats, while inorganic polymers may also persist or transform into substances that are hazardous. Marine ecosystems, in particular, are heavily affected, with a [predicted 20 – 53 million tonnes](#) of plastics entering aquatic ecosystems annually by 2030. This pollution leads to the death of aquatic species, the disruption of habitats, and the contamination of the food chain with microplastics.

From a societal standpoint, the [implications of polymer pollution may extend to human health](#), with micro-sized polymers found in drinking water, food, and even the air. These particles raise significant health concerns, as their long-term effects on human health are still not fully understood. Moreover, economically, the costs associated with polymer pollution are immense. The World Wildlife Fund (WWF) estimated that the cost of from annual plastic production over its lifetime costs the [global economy in excess of a trillion dollars](#), and this figure likely underestimates the total cost when considering all types of polymers.

2.2 Challenges in Polymer Waste Management

Amongst most pressing challenges in managing polymer waste is the [complexity of identifying and recycling various polymer types](#). Polymers have distinct structures and properties that dictate their behaviour in the environment and their recyclability. [Common organic polymers](#) include polyethylene (PE), polypropylene (PP), and polyvinyl chloride (PVC), while inorganic polymers include materials such as polysiloxanes (commonly known as silicones). Each polymer type requires specific identification and recycling processes, and incorrect identification can severely compromise the recycling process.

2.2.1 Polymer Identification

[Accurately identifying polymers](#) is the first critical step in effective recycling. Misidentification can lead to contamination of the recycling stream, reducing the quality of the recycled material and increasing waste. This in turn detracts from the economic viability of such methods. Traditional methods, such as manual sorting and basic chemical tests, are often inadequate, particularly for mixed or complex polymer wastes. Advanced techniques like [Near-Infrared \(NIR\) spectroscopy](#), when combined with machine learning algorithms, offer improved accuracy in polymer identification. However, these techniques have required sophisticated equipment and expertise, limiting their accessibility and widespread use, especially in lower-income regions.

2.3 Polymer Recycling

Once polymers have been accurately identified and sorted — a critical step to ensure the success of subsequent processes — recycling can begin. Recycling polymers, which is the conversion of waste polymers into a reusable material, can be achieved via mechanical or chemical recycling, both of which present several challenges.

Mechanical Recycling: This is the [most widely used method](#), involving the physical processing of polymers through steps such as sorting, cleaning, shredding, melting, and

IMPERIAL

remoulding. The primary changes during mechanical recycling are physical rather than chemical. The primary benefit of mechanical recycling is its simplicity and [cost-effectiveness](#), as it directly processes polymers into new products without extensive chemical manipulation. However, the process can lead to several issues:

- **Degradation:** The high temperatures and shear forces involved can cause the breakdown of polymer chains, reducing the molecular weight and forming shorter chains.
- **Physical Property Changes:** Although the chemical structure remains largely intact, the physical properties of the polymer, such as tensile strength, flexibility, and impact resistance, can degrade over multiple recycling cycles.

Chemical Recycling: This method involves the use of chemicals to break down polymers into constituent components which can be used as raw materials for the manufacture of other products. [Chemical recycling](#) can be achieved via processes such as pyrolysis, hydrolysis, and/or glycolysis. Chemical recycling offers several advantages and faces its own set of challenges:

- **Depolymerisation:** Chemical recycling breaks down the polymer into its monomers or other smaller molecules.
- **Re-polymerisation:** If monomers are cleanly obtained, they can be repolymerised to form new polymers, potentially matching the quality of the original material. This process differs from mechanical recycling, where the polymeric structure is not purposefully broken down and thus often results in lower-quality recycled material.
- **Product Variety:** Depending on the process, chemical recycling can produce various by-products, such as gases, oils, or other chemicals, which may be used in other industrial applications. However, chemical recycling is [currently more expensive and energy-intensive](#) compared to mechanical recycling.

2.4 Global Disparities in Waste Management

The challenges of polymer waste management are compounded by significant global disparities, which directly relate to several UN Sustainable Development Goals (SDGs). High-income countries generally have more advanced recycling infrastructures, yet even in these regions, overall recycling rates fall short of optimal levels. For instance, the European Union reports a [mean recycling rate of ~40 % for plastics](#), with lower rates for more complex or less common polymers. This inefficiency contributes to the broader goal of the [United Nations Social Development Goal \(SDG\) 12 \(Responsible Consumption and Production\)](#), which aims to substantially reduce waste generation through prevention, reduction, recycling, and reuse. In contrast, low- and middle-income countries [often lack the necessary infrastructure for even basic waste management](#), leading to higher rates of mismanaged polymer waste.

2.5 Technological and Policy Gaps

Efforts to improve polymer waste management are often hindered by technological limitations and insufficient policy frameworks. Current recycling technologies are not keeping pace with the [increasing complexity and variety of polymer waste](#). Additionally, existing policy frameworks frequently lack the necessary incentives or regulatory measures to drive higher recycling rates. [Extended Producer Responsibility \(EPR\)](#) schemes, which hold manufacturers accountable for the end-of-life management of their products, are a promising approach but are not yet widely adopted globally.

2.6 Potential Solutions to Polymer Waste Management

Despite these challenges, several innovative approaches are being developed and implemented to improve polymer waste management:

- **Policy Interventions:** International efforts are increasingly focusing on reducing polymer waste. For example, the [European Union's Circular Economy Action Plan](#) aims to make all plastic packaging recyclable by 2030 and is pushing for similar

advancements in the recycling of other polymers. Countries like Japan and Germany have achieved high recycling rates through stringent regulations and advanced waste management systems.

- **Public Awareness and Education:** Public campaigns and educational initiatives play a crucial role in promoting better recycling practices and reducing polymer waste. While the effectiveness of these programs varies, they are an essential part of a comprehensive strategy to tackle polymer pollution.
- **Advanced Sorting Technologies:** [Integrating AI and machine learning with optical and spectroscopic methods](#) can significantly enhance the accuracy and efficiency of polymer identification. These technologies could transform the sorting process, making it more precise and scalable (and potentially economical), and are particularly promising for complex waste streams containing both organic and inorganic polymers.

3. Aim and Objectives

In response to the global challenges of sustainable polymer recycling in the 21st century, this practical session aims to offer you the opportunity to investigate chemical recycling strategies by employing AI models for the accurate identification and classification of polymers.

3.1 Practical Objectives

1. Polymer Identification and Classification:

- Utilise NIR spectroscopy to build a reference database for various known polymers.
- Explore and evaluate data to ensure the quality and reliability of the spectral data.
- Employ machine learning (ML) algorithms, particularly k-Nearest Neighbours (kNN) using Python, to train models for the accurate identification of unknown polymers.

2. Chemical Recycling of Polymers:

- Design and implement a chemical recycling procedure for the identified polymers, focusing on chemical rather than mechanical recycling methods.
- Conduct characterisation of the products of the recycling process to evaluate the efficiency and viability of the recycling route.

3. Critical Analysis and Reporting:

- Analyse the strengths and limitations of the machine learning models and sustainability of the recycling processes employed.
- Present your findings in a structured presentation, demonstrating your understanding of the AI model, the chemical recycling process, and their implications.

4. Learning Outcomes

By the end of this lab, you should be able to:

- 1. Develop and Implement ML Models:** Construct, train, and evaluate ML models, specifically kNN, using Python to classify polymers based on spectral data.
- 2. Evaluate Data Quality and Model Performance:** Critically assess the quality of spectral data and optimise machine learning models, understanding the impact of data quality on model accuracy and reliability.
- 3. Utilise Generative AI as a Tool-Building Resource:** Leverage generative AI to assist in writing and optimising Python code, enhancing your ability to build and refine machine learning models. You will also develop an understanding of how generative AI can serve as a tool to accelerate problem-solving and encourage creation in data science and chemistry.
- 4. Design Chemical Recycling Processes:** Propose and execute a chemical recycling strategy for polymers, differentiating between chemical and mechanical recycling methods and assessing the efficiency and sustainability of the chemical process.
- 5. Integrate Computational Tools with Practical Chemistry:** Synthesise knowledge of AI and machine learning with hands-on laboratory techniques to solve complex problems in polymer chemistry.
- 6. Critically Analyse and Reflect on Laboratory Work:** Reflect on the strengths and limitations of both the AI models and the sustainability of the chemical recycling methods used, articulating these insights in both oral and written formats.

5. Experimental Section: Getting Started

Familiarise yourself with the lab guidelines. Watch the [guest lecture](#) by Dr Martin Holicky of Matoha Instrumentation, which covers the application of NIR and AI algorithms in polymer identification. This foundational knowledge will be essential for your work in the lab.

6. Section A: Identifying a Suitable Chemical Recycling Route

6.1 Polymer Allocation and Research

Two weeks before attending the lab, you and your partner will be assigned a specific common polymer (**Figure 1**). Your first task is to **design a sustainable chemical recycling** strategy for it, which you will implement during week two.

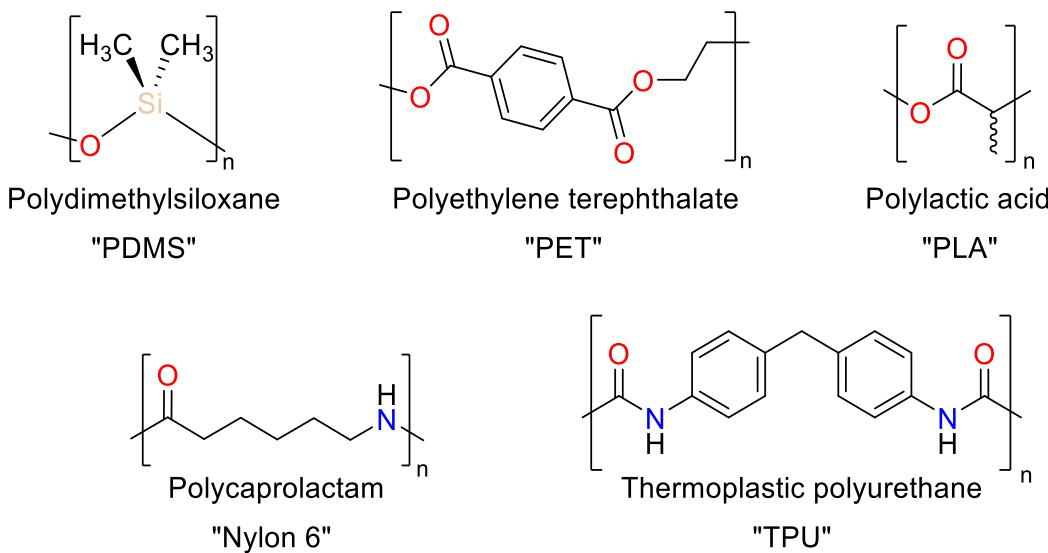


Figure 1: Five "common" polymers used in this lab.

With your partner, begin by conducting a thorough **literature review to identify suitable chemical recycling methods for your assigned polymer**. Cross-reference the chemicals and equipment required for your proposed procedure with the lab's inventory

IMPERIAL

lists (ID1 – Chemicals/Equipment List) to confirm their availability. This is crucial to ensure that you can execute your planned experiments.

Part of the assessment criteria for this stage is to assess how sustainable your chosen route is with respect to the “[12 Principles of Green Chemistry](#)”? (click on the [links](#) or refer to the Synthesis 1.3 notes for a refresher on what these are). A useful tool that may help with this is “[Dodecagreen](#)” – a free, online platform which will help you quantify the sustainability of a given method and generate plots to visually represent your evaluation (**Figure 2**).

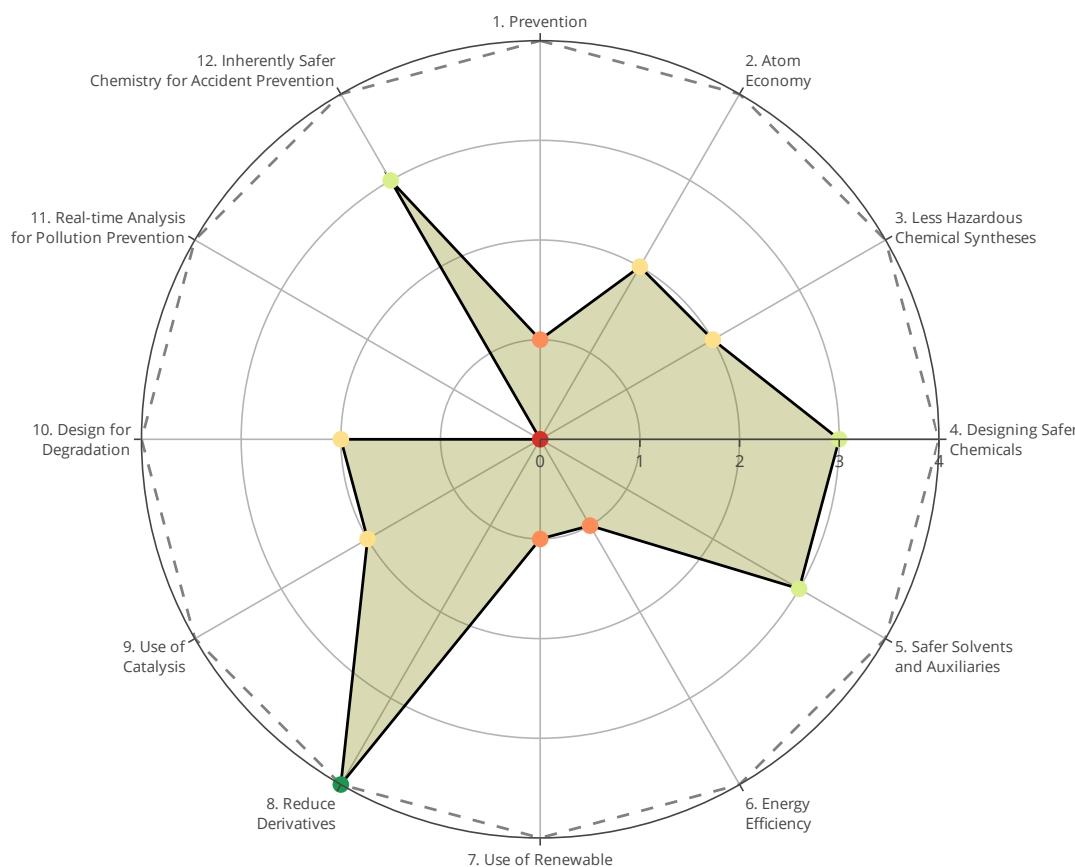


Figure 2: An example sustainability plot produced using “[Dodecagreen](#)”.

Once you have selected a route, you and your partner should each **draft a risk assessment and experimental procedure**, ensuring to cite the relevant scientific papers. These must be completed and **submitted on LabArchives before the start of your lab session**. *In order to generate the possibility of data comparisons, you and your partner should use the same method with a slight variation in conditions.*

Have you considered?

- How well the different methods align with the sustainable chemistry attributes?
- Is special apparatus required for various operations of the reaction?
- Reaction conditions, e.g., should the solvent be especially dry; should the reaction be carried out under nitrogen due to the sensitivity of reactants/intermediates/products to oxidation?
- Does the reaction require monitoring (quite often an essential part of any well-planned experiment)?
- Do any of the reagents need carefully quenching prior to work-up?
- The work-up steps required, such as the choice of solvent for extraction, whether any acid/base washes are required, and does care need to be taken if any of the key intermediates or products are volatile?
- How will the product be purified, and how will purity be evaluated?
- What physical data needs to be checked on the product?
- Is the process safe? What precautions are necessary?

Hints & Tips

- Refer back to your Y1 I-Engage “Finding Chemical Information” session.
- If you wish to use [Reaxys](#), you can sign in with your Imperial email and password.
- If you wish to use [Scifindern](#), you will need to either access it on campus or setup the [VPN](#), and register – you can find some instructions on how to do so here:
- If you encounter problems connecting to the VPN, try Imperial’s [Remote Desktop](#).
- Check out [Scifindern tutorials and videos](#) if you are new to the platform.

IMPERIAL

Reaction Risk Assessment (RRA) Guidelines (for further guidance, see the separate document in the ID1 lab folder on Blackboard):

- The Reaction Risk Assessments should include ALL chemicals to be used for the reaction, the isolation by work-up, and the purification.
- If you are following a literature procedure, a link to this (e.g. DOI) should be included.
- You need to look at the individual MSDSs and assess the hazards of each chemical carefully (for a guide on how to read and interpret MSDSs, please see separate pdf), and also consider any incompatibilities (the [Bretherick's Index](#) is a great resource for this).
- It is equally important to assess how you will carry out the reaction, for example, what is the order of reagent addition (for example, we add acid to water and not water to acid), and how will you quench any reactive reagents (for example, bromine, thionyl chloride, etc, all need quenching in a controlled manner) – you should therefore include both a scheme and a detailed procedure of how you will carry out the whole process in your Reaction Risk Assessments.
- You should consider how you will transport your samples around the lab, for example, from your fumehood to the rotary evaporator – if you are using something that is particularly hazardous by inhalation, add notes in your Reaction Risk Assessments about moving them in sealed vessels.
- You should include details of how you will dispose of everything safely, taking into account incompatibilities and the overall hazards.

The lab coordinators, Rebecca and Benji, can advise you on anything you are not sure about, so if in doubt, please reach out!

During week one, a lab coordinator will give you feedback on your risk assessment and experimental procedure. If edits to your documents are required, it is your responsibility to make them before the start of week two. **No handling of chemicals can take place before your pre-lab work has been approved.**

7. Section B: Building an AI Model for Identifying the Target Polymer

7.1 Activity 1: Building a Polymer Reference Database

To begin, you will identify and catalogue a set of known polymers. Each polymer sample is labelled with a QR code and a name, representing its known composition. These labelled spectra will serve as the “known” data, which you will use to train your AI model.

For each sample, acquire five spectra by varying sample orientation and position on the spectrophotometer. At the end of your analysis, you should have a reference database with at least 150 spectra.

7.1.1 Task 1.01: Switching on PlasTell

Start by powering on the PlasTell spectrophotometer. Plug the machine into a power source using the supplied USB adapter. It is crucial to use only the provided USB adapter, as other adapters may not deliver the necessary power quality. The machine will automatically start with no need for a power button. When it shows "CALIBRATING, PLEASE WAIT," avoid placing any items or samples on the sensor. Calibration takes about 10 seconds, after which the display will show "READY."

7.1.2 Task 1.02: Connecting PlasTell to the Matoha App

Labelling of your data can be conducted directly on the Matoha app. To do so, open the app on your designated iPad and press "Connect to your instrument." This will open a page with detailed instructions. Check that the PlasTell is switched on, and the tablet provided has Bluetooth and location sharing enabled. The app may request permissions for Bluetooth and Location — both must be granted for the connection to work. Note that Location permission is necessary because phone manufacturers require it for Bluetooth scanning. Press "Scan for instruments." ***If your tablet or the Matoha app are not working, please reach out to your GTA for assistance.***

IMPERIAL

After scanning, a list of available devices within Bluetooth range will be displayed. If multiple devices are detected, choose your instrument from the dropdown list by matching the serial number (found on the bottom of your device), and press connect. If prompted, enter the pairing code displayed on your spectrometer screen (usually 123456) into your device. The instrument should now be fully connected.

7.1.3 Task 1.03: Creating and Managing Spectral Collections

Collections are folders used to organise your spectra (measurements). Follow these steps to create and manage collections:

- 1. Open the Matoha app and select "My Collections" from the homepage or side menu (the three lines in the top-right corner).**
- 2. Create a New Collection:**
 - Press the green "+" icon in the top-right corner.
 - Name your collection and select your machine type (PlasTell).
 - Click "Add." Your collection will now be listed under "My Collections."

7.1.4 Task 1.04: Measuring & Saving Samples

Ensuring your tablet is connected to the switched-on device, and having set up your collection, you may now start measuring your spectra by logging onto “Measurement interface”. Before measuring, you may register the item details by:

- Using the tablet’s camera to scan the QR code sticker

To measure, place the item on the sensor, and within approximately one second, the machine will display the results. A correct measurement will be indicated by the word “recorded” and a green hexagon, while a faulty or dubious measurement will show as “unknown” with a red hexagon. If you encounter any issues, consult your GTA for assistance.

Hints & Tips on Collecting Good Spectra

To ensure the collection of accurate spectra, adhere to the following best practices:

- 1. Do Not Move the Sample:** Keep the sample stationary during measurement. The machine detects movement, which can lead to errors (e.g., the "KEEP STEADY" error). Place the item on the sensor quickly, then remain still.
- 2. Place the Item Flat:** Ensure the item is as flat and stable as possible on the sensor.
- 3. Press Down Soft Items:** For soft samples, particularly fabrics, press them onto the sensor to create a flat, continuous layer.
- 4. Orient Small Samples:** If a small sample doesn't completely cover the detector, place it on the right side of the sensor.
- 5. Transparent Samples:** Do not cover transparent plastic samples with your hand, as it will block the external lamp, preventing accurate measurement.
- 6. Avoid Measuring Printed Areas:** Printed areas or labels may be made of different materials than the rest of the item, leading to inaccurate identification.
- 7. Fold Thin Samples:** For thin plastic films and fabrics, folding them can improve measurement accuracy.
- 8. Ensure Samples are Dry:** Any moisture in the sample can distort the results, so make sure it is completely dry.
- 9. Thick Transparent Plastics:** For very thick transparent polymers, measure across a thinner section for more reliable results.

7.1.5 Task 1.06: Exporting Data

Go to "My Collections" and select the relevant collection. A list of your saved measurements will appear. Press the green "Export" button. Click "Generate link" to create a unique link for your dataset. Keep this link secure, as it grants access to the collection. Press the "Copy link" icon to save the link to your clipboard. You can paste it into any web browser to access the data. The data is exported in CSV format (comma-separated values), with columns such as index, ID, timestamp, serial number, materials,

and comments. ***At this stage, reach out to your GTA who will assist you with exporting the data with the spectral column included.***

7.2 Activity 2: Exploring your Data

In this task, you will begin exploring your data using Python. Each sub-task will clearly outline its objective, and key steps will be suggested to guide your code design. These steps help break down complex processes but are not the only way to achieve the objectives. *While you may choose to use alternative routes to those suggested in this script to achieve the same objectives, you are advised not to deviate drastically from these guidelines. Doing so may make it more difficult to achieve the objectives or to allow technical staff to more easily assist with any debugging or issues that may arise.*

You May Use Generative AI to Help Design Your Code

The use of generative AI (e.g. Copilot, ChatGPT, Claude, Gemini) tools for writing and troubleshooting Python code ***is encouraged in this laboratory practical***, as it aligns with the learning goals of this lab. However, you must critically evaluate any AI-generated code to ensure it is appropriate for your project. ***You will be assessed on your understanding of mechanisations of the code and its application to your work.*** The emphasis is on demonstrating that you comprehend the functionality and logic of the code rather than simply using an AI tool to complete tasks. If you have any doubts, please reach out to Benji or Rebecca for clarification.

Collaboration & Plagiarism

While collaboration with your lab partner is encouraged, sharing or copying code from other students is strictly prohibited. This will be considered plagiarism and addressed according to the college's policy. You are responsible for producing your own code with GenAI assistance and ensuring its originality.

7.2.1 Task 2.01: Counting Spectra & Polymers from your .CSV File

The objective of this exercise is to load the NIR spectral data you have collected from the .CSV file, determine the total number of spectra, and calculate how many times each unique polymer appears in the dataset.

Key Steps:

1. Loading the CSV File

Begin by importing the necessary Python libraries (e.g., Pandas) for working with structured data. Load the CSV file into an object for data manipulation, and store the file path in a variable for easy updates.

2. Counting the Total Spectra

Next, define a function to count how many spectra are in your dataset. You should create this function to output the result clearly, such as by printing it to the console.

3. Counting Unique Polymers

Write another function that will examine the column containing the polymer labels and dynamically count how many times each unique polymer appears in the data. The goal here is to identify the frequency of each polymer based on the spectra available.

4. Displaying the Results

Once your functions are ready, call them to display the total number of spectra and the count for each polymer. Ensure the output is easy to interpret, as you will use it to analyse polymer distribution within the dataset.

Representative display of results for Task 2.01

```
Total number of spectra: 40.  
Number of spectra for each unique polymer:  
labelMaterialsString  
PMMA    14  
PET     14  
PP      12
```

Hints & Tips for Task 2.01

- Consider how to check if the file was loaded correctly before proceeding with analysis.
- There's a built-in method in the data-handling library that can be used to count unique occurrences in a particular column.

7.2.2 Task 2.02: Filtering and Saving Specific Data

The objective of this task is to filter specific columns from a CSV file, create a new dataset with selected data, and save it as a new CSV file, “[2.02_filtered-data.csv](#)”.

Key Steps:

1. Loading the CSV File

Begin by using a Python library for handling structured data (e.g. Pandas). You'll first need to load the CSV file into a structured format, using a file path that can be easily modified in the code. Store the path as a variable so you can adjust it later if necessary.

2. Selecting Specific Columns

Once the data is loaded, identify the columns of interest — in this case, the columns related to the polymer labels and the spectra. Extract only these columns from the larger dataset and create a new subset of data that contains just this information.

3. Saving the Filtered Data

After creating the filtered dataset, save it as a new CSV file named “[2.02_filtered-data.csv](#)”. Exclude any extra information, such as the row index, when saving. Specify the file path for the new CSV and ensure it saves correctly.

4. Displaying a Confirmation Message

After saving the new CSV file, ensure your program outputs a message to confirm the successful completion of the task, including the name or location of the saved file.

Representative display of first 5 rows for Task 2.02.

	labelMaterialsString	spectrum
0	PMMA	[59412, 59023, 58551, 58072, 57675, 57438, 574...
1	PMMA	[60498, 59479, 58194, 56780, 55406, 54239, 534...
2	PMMA	[36707, 38337, 39372, 39880, 39955, 39707, 392...
3	PMMA	[63681, 63724, 63583, 63328, 63041, 62808, 626...
4	PMMA	[64207, 64258, 64129, 63887, 63618, 63403, 633...

Hints & Tips for Task 2.02

- Look for a method in the data-handling library that allows you to select only certain columns from a dataset. (Hint: You can reference column names directly using brackets.)
- When saving the new CSV file, ensure the function you use has an option to exclude row indices. (Hint: There's an argument that can be set to avoid saving the row index—something like `index=False`.)

7.2.3 Task 2.03: Transforming Spectrophotometer Data

The objective of this task is to transform spectrophotometer data by converting a string representation of spectra into usable data, calculating wavelengths, and saving the result as a new CSV, “[2.03_transformed-data.csv](#)”.

Key Steps:

1. Importing Necessary Libraries

Import the libraries required for this task. These will help you manage data, convert data types, perform mathematical operations, and handle CSV files.

2. Function Definition

Define a function that will take two arguments: the file path to your input CSV file and the file path where you want to save the transformed data. This function will process the data and perform the necessary transformations.

3. Converting the 'spectrum' Column

Load the CSV file into a structured format that allows for easy manipulation. The 'spectrum' column contains string representations of lists. You'll need to convert these strings into actual Python lists. A specific method from the ast library can help convert these safely.

4. Calculating the Wavelength Range

Once the 'spectrum' data has been converted into lists, calculate a range of wavelengths from 1550 nm to 1950 nm (both values inclusive). The number of points (128) in this range will depend on the length of the first spectrum list in the dataset.

5. Preparing the Data for Output

After calculating the wavelengths, prepare the data for saving. Create a header row that includes the calculated wavelength values and the unique polymer labels from the dataset. You will need to combine these to align the spectra and labels correctly.

6. Transposing the Data

In order to save the data properly, transpose it so that the wavelength values and spectra align as required. This will ensure the correct format for saving.

7. Writing the Transformed Data

Finally, save the transformed data to a new CSV file, ensuring that your header row is included. It is recommended you name your output file "[2.03_transformed-data.csv](#)". Use a method that writes to CSV files and makes sure the output is correctly formatted.

8. Confirmation Message

After successfully saving the file, print a message confirming where the new CSV file has been stored. This provides feedback to ensure the task has been completed correctly.

Representative display of for Task 2.03 for Two PMMA Spectra

	Wavelength (nm)	PMMA	PMMA.1
0	1550.000000	59412	60498
1	1553.149606	59023	59479
2	1556.299213	58551	58194
3	1559.448819	58072	56780
4	1562.598425	57675	55406

Hints & Tips for Task 2.03

- There is a method in Python that can safely evaluate string representations of Python data types. (Hint: It comes from the ast library and can convert strings into lists.)
- When preparing the wavelengths, consider the total length of the 'spectrum' list and create a range from 1550 nm to 1950 nm using a mathematical library. (Hint: You might use a function to evenly space numbers in that range.)
- For the final output, ensure your data is saved in a transposed format so that each row corresponds to a wavelength and its corresponding spectra. (Hint: Use the transpose() function to reformat the data.)
- At this point, your columns will likely be numbered per polymer. E.g. PET.01. It is recommended that for single digit numbers, you input a zero at the front, such that the nomenclature is e.g. “PET.01” rather than “PET.1”.

7.2.4 Task 2.04: Combining Datasets

The objective of this task is to combine two datasets, that collected by yourself (“[2.03_transformed-data.csv](#)”) and that provided by the lab (“[data_source2.csv](#)”), into a single dataset, and saving it as “[2.04_combined-data.csv](#)”.

Have you considered?

Data consistency is crucial to ensuring uniformity when working with multiple datasets or instruments. Inconsistent data can lead to significant errors in analysis, particularly in machine learning applications where small discrepancies can skew results. To maintain consistency, consider the following:

1. **File Format:** Ensure all files are saved in the same .csv format, with consistent delimiters and headers.
2. **Data Format:** Verify that each file's data follows the expected format, such as having the wavelength in the first column and intensity in the second. This consistency is crucial for accurate data processing and analysis.
3. **Units:** Confirm that all spectra use the same units for both wavelength and intensity (e.g., nanometres for wavelength and arbitrary units for intensity). Inconsistent units can lead to incorrect interpretations and analysis errors.
4. **Data Types:** Check that the data types in each column are consistent across all files. This ensures that numerical data is properly recognised and processed by analytical tools.
5. **Metadata:** If the datasets include metadata — such as polymer type or measurement conditions — ensure that this information is complete and consistent across all files. Metadata plays a critical role in understanding the context of your data and ensuring accurate classification and analysis.
6. **Resolution:** Verify that the resolution, or the step size between data points, is consistent across all datasets. In cases where resolution varies, data processing techniques such as interpolation may be required to standardise the data.

Key Steps:

1. Importing the Necessary Library

Start by importing an appropriate library for data manipulation in Python.

2. Function Definition

Define a function that takes three arguments: the paths for the first and second CSV files, as well as the output path for the combined dataset.

3. Loading the Datasets

Load the first dataset (“`2.03_transformed-data.csv`”), ensuring that all columns are included. For the second dataset (“`data_source2.csv`”), load it while skipping the first column.

4. Combining the DataFrames

Use an appropriate method to merge the two datasets side-by-side, ensuring they are aligned by their rows.

5. Saving the Combined Dataset

Save the resulting DataFrame to a new CSV file, ensuring that the index is not included in the saved output. It is recommended you name your output file “`2.04_combined-data.csv`”.

6. Confirmation Message

Print a message to confirm the location where the combined dataset has been saved.

Hints & Tips for Task 2.04

- When loading the second dataset, “`data_source2.csv`”, check for parameters that allow you to skip specific columns. (Hint: Many functions provide options for selecting which columns to load.)
- For combining the datasets, look for a function that enables you to concatenate DataFrames horizontally. (Hint: This typically involves specifying an axis in the function call.)

- Ensure that when saving your combined dataset, you use an option to exclude the index from the output file. (Hint: There's a common argument in CSV writing functions that handles this.)
- After saving, include a message that verifies the output path and confirms successful file creation. (Hint: Clear messaging helps ensure you know where to find your results.)

7.2.5 Task 2.05: Counting Columns by Polymer

The objective of this task is to count the occurrences of column headers that share a common polymer, following the combination of the two datasets. This analysis differs from previous efforts by focusing on the **frequency of polymer types within column headers**. By analysing the distribution of polymer types across the merged columns, we can assess the composition of our reference database and identify potential biases. Note that column headers may now include a suffix (e.g., "PET.01") alongside the polymer name.

Key Steps:

1. Import Libraries

Import the necessary libraries for data manipulation and counting.

2. Define the Function

Create a function that takes a file path as an argument.

3. Load the CSV File

Load the "[2.04_combined-data.csv](#)" into a DataFrame for easy access to its structure.

4. Retrieve Column Headers

Extract all column headers except the wavelengths column.

5. Initialise Prefix Storage

Prepare a structure to hold the prefixes from the column headers. (Note: there is an assumption here that the polymer name is held within a prefix e.g. "PET.1". If this is not

the case, you will need to adapt the code to match the naming conventions used in your dataset).

6. Extract and Count Prefixes

Loop through the headers to find and count the prefixes, using a counting method.

7. Report Results

Print the total number of relevant columns and the counts for each unique prefix.

Have you considered?

- Does the proportion of polymers in your dataset impact your machine learning model?
- Do you need to take any actions to adjust for any discrepancies in the proportion of polymers in your reference database?

Hints & Tips for Task 2.05

- Familiarise yourself with how to access DataFrame attributes for headers. (Hint: Check the documentation for properties that return column names.)
- Consider how you can manipulate strings to extract parts of the column names effectively. (Hint: Look for string methods that can help with splitting.)
- Use a counting method that can efficiently tally occurrences from a list. (Hint: Think about built-in Python features designed for counting.)

7.2.6 Task 2.06: Looking for Duplicates

The objective of this task is to identify duplicate spectra in a CSV file by computing and comparing hashes of the spectrum data. Save the updated data as “[2.06_duplicate-checked-data.csv](#)”.

Have you considered?

Having duplicate values in a machine learning dataset can negatively impact model performance. Duplicates effectively give some data points more weight than others, which biases the model toward those duplicated examples. The model may become skewed toward learning features that are only relevant to these overrepresented samples, leading to poor generalisation. As such, it is important to inspect your datasets to avoid the presence of duplicate datasets. This is especially pertinent when multiple datasets have been collated together.

Hash functions are useful for detecting data duplicates because they take an input, like a spectrum, and convert it into a fixed-length string of characters called a hash. This hash acts like a unique fingerprint for the input data: even small changes in the original input will result in a completely different hash (most of the time). By comparing these hashes instead of the actual data, we can quickly identify duplicates — if two inputs produce the same hash, we know they are identical. This method is efficient because comparing short hash values is much faster than comparing potentially long and complex datasets.

Key Steps:

1. Import Libraries

Begin by importing the necessary libraries for data manipulation and hashing.

2. Define Hash Function

Create a function that takes a spectrum column as input, converts it to a string, encodes it, and generates a hash.

3. Define Function for Finding Duplicates

Create another function that accepts the file path of the CSV. This function will handle loading the data and identifying duplicate spectra.

4. Load the CSV File

Read the “[2.04_combined-data.csv](#)” into a DataFrame to access the spectra data.

IMPERIAL

5. Initialise Storage for Hashes

Set up a structure to store computed hashes and their corresponding column names.

6. Compute Hashes

Iterate through the relevant columns (excluding the first one), compute the hash for each spectrum, and store the column names.

7. Identify Duplicates

Find and collect columns that share the same hash, indicating duplicate spectra.

Return all instances of duplicate spectra.

8. Decide how to handle the duplicate columns

If you discover duplicate data, how are you going to handle that data? Save any altered data as a CSV file. It is recommended you name your output file “[2.06_duplicate-checked-data.csv](#)”.

Hints & Tips for Task 2.06

- When creating the hash function, ensure you know how to convert data types to strings and handle encoding. (Hint: Investigate string methods and encoding functions).
- Consider how to manage data storage for hashes efficiently. (Hint: A dictionary can help relate hashes to their corresponding column names).
- Use pandas functions to simplify reading and processing CSV data. (Hint: Familiarise yourself with methods for reading CSV files into DataFrames.)
- Think about how to check for duplicates within your dictionary after computing the hashes. (Hint: Consider using conditional checks to identify entries with multiple column names).

7.2.7 Task 2.07: Checking for Invalid Values in Transformed Data

The objective of this task is to identify and report invalid values from “[2.06_duplicate-checked-data.csv](#)”. This includes checking for missing values and non-numeric entries.

Save the corrected data in as “[2.07_checked-data.csv](#)”.

Key Steps:

1. Importing the Necessary Library

Start by importing the relevant Python library that you will use for data manipulation.

2. Function Definition

Define a function that accepts a file path as an argument. This function will be responsible for loading the CSV file and performing checks on the data.

3. Loading the CSV File

Use the chosen library to read the CSV file “[2.06_duplicate-checked-data.csv](#)” into a structured format, enabling you to easily access and manipulate the data within.

4. Iterating Over Columns

In your function, set up a loop to iterate through each column of the DataFrame. You may want to skip the first column, which contains the wavelength values and focus on the relevant data columns.

5. Checking for Missing Values

For each column, check for any missing values (NaN). If any are found, print a message indicating the column name and the row number, adjusting for the presence of a header.

6. Checking for Non-Numeric Values

One possible approach here, is to attempt to convert the value to a float. If an entry cannot be converted (due to being non-numeric), catch the exception and print a message identifying the column name and row number.

7. Completion Message

After completing the checks across all relevant columns, print a final message confirming that the checks have been completed.

Have you considered?

If your dataset has errors, how are you going to handle that data? Common approaches include data imputation (filling in missing values using statistical methods), discarding incomplete data, or consulting additional resources to retrieve the missing information. The choice of strategy should be guided by the potential impact of missing data on the overall analysis and the specific requirements of your machine learning model. Save any altered data as a CSV file. It is recommended you name your output file “[2.07_checked-data.csv](#)”.

Hints & Tips for Task 2.07

- ***Tip: You may want to create a temporary version of your data which contains the errors you are looking for. In so doing, this acts as a control for your code.***
- Find a method to load your CSV file that automatically handles headers and data types. (Hint: This returns a DataFrame, which is very useful for data manipulation.)
- Use a conditional check to identify NaN values in each column. (Hint: The library has built-in functionality for detecting missing values.)
- When attempting to convert values to floats, consider using a try-except block to handle any conversion errors gracefully. (Hint: This allows you to catch and respond to errors without crashing your program.)

7.3 Activity 3: Visualising Your Data

To understand the overall quality of your data, it is important to examine its central tendency (such as the mean or median), dispersion (such as variance or standard deviation) and the existence of outlier data. These metrics provide insights into what a

“typical” spectrum for your labelled polymer looks like and how much variation exists among the collected spectra.

7.3.1 Task 3.01: Visualisation of Central Tendency & Dispersion

The objective of this task is to extract, process, and visualise spectral data. You will calculate the average spectra and the dispersion (e.g. standard deviation) for different polymer types and then visualise this information using error bars. Save the averaged data as “[3.01_averaged-data.csv](#)”.

Key Steps:

1. Import the Required Libraries

Begin by importing the necessary Python libraries (e.g., pandas, numpy, matplotlib). These will help you with data handling, calculations, and visualisations.

2. Extracting Prefixes from Spectral Data

Write a function that extracts the portion of each column name that will allow you to group spectra based on polymer names.

3. Reading and Processing the Spectrum Data

Define a function that reads “[2.07_checked-data.csv](#)”. You’ll need to extract the first column as wavelengths and the remaining columns as spectra. Ensure the function can handle errors if the file format isn’t as expected.

4. Processing Spectra for Central Tendency and Dispersion

Group the spectra by polymer type and compute the average intensity and standard deviation for each group. Decide which average (mean, median, or mode) to use to describe your data, and justify your choice.

5. Visualisation

Plot the average spectra using the calculated standard deviation as error bars. In line with convention, don’t forget to reverse the x-axis (wavelengths) so that larger numbers are on the left. An example of the expected output is shown in **Figure 3**.

6. Saving the Results

Save your calculated average and standard deviation spectra to a new CSV file. Make sure the file is properly formatted with clear column names for each polymer type's mean and standard deviation. It is recommended you name your output file “[3.01_averaged-data.csv](#)”.

7. Output Confirmation

Print a message confirming that the results have been saved and provide the file path for the saved CSV.

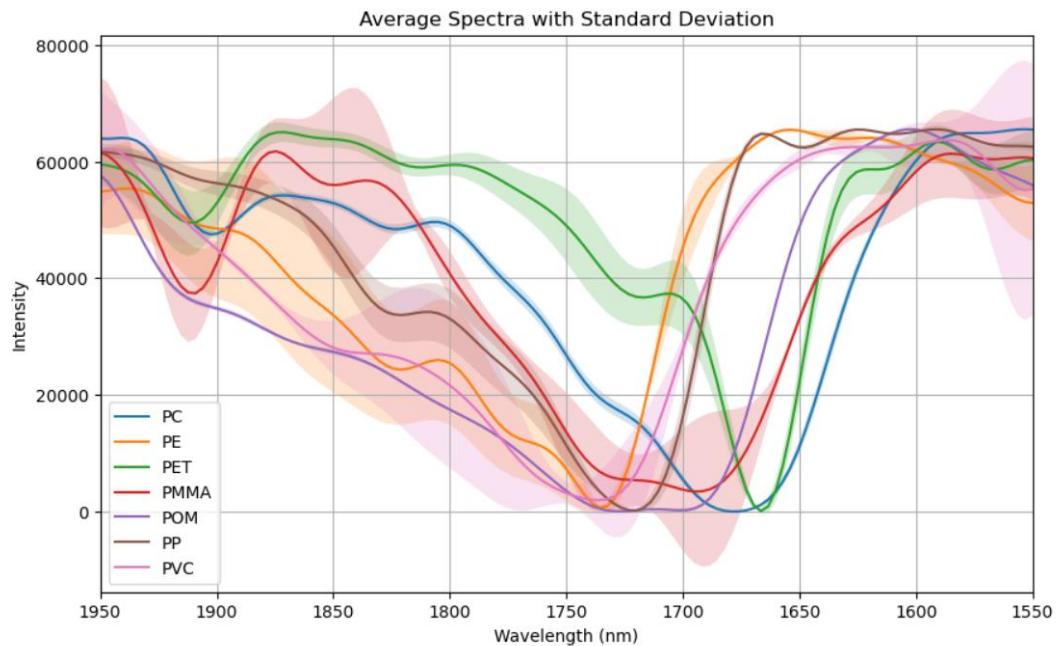


Figure 3: Representation of Central Tendency and Dispersion for Selection of Polymers.

Hints & Tips for Task 3.01

- Consider how you will handle zero values when reading the CSV file.
- There are built-in functions that can help you calculate averages and standard deviations for grouped data.
- For visualising error bars, think about which plotting functions in matplotlib will best suit your needs.

7.3.2 Task 3.02: Using Heatmaps to Explore Spectra Variation

The objective of this task is to visualise the variation in spectra data using heatmaps. Save the processed data as “[3.02_outlier-checked-data.csv](#)”.

2D heat maps (**Figure 4**) are valuable tools for visualising variation in spectral data because they represent complex information in an easily interpretable format. In a heatmap, different colours indicate the intensity or value of data points, allowing you to quickly see patterns and trends across multiple variables, such as wavelength and intensity. This makes it easy to spot outliers — data points that stand out from the rest, often indicating anomalies or interesting features in the spectral data. By providing a visual overview, heat maps quickly help researchers identify areas that may require further investigation, making it easier to analyse large datasets effectively.

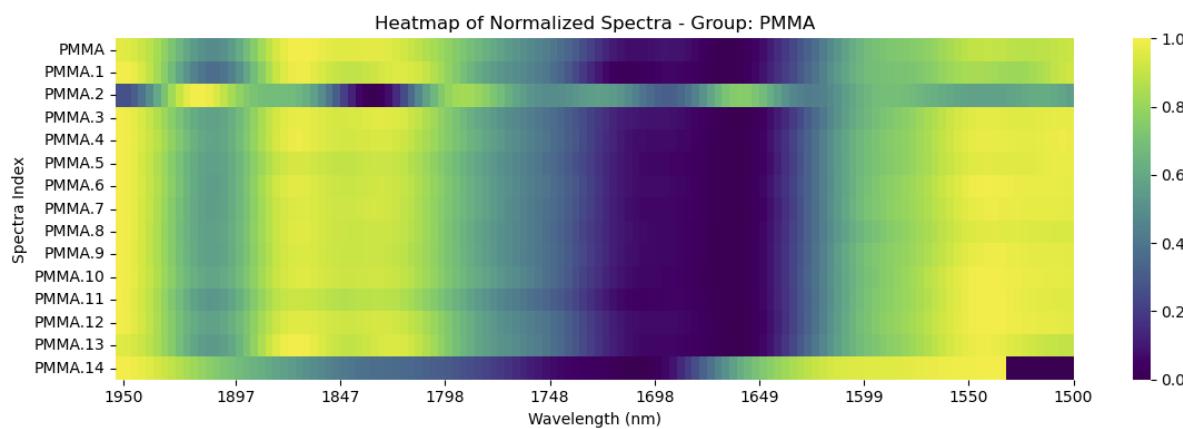


Figure 4: 2D Heat Map for PMMA.

Key Steps:

1. Import Required Libraries

Start by importing the necessary Python libraries (e.g., pandas, numpy, matplotlib, and seaborn). These will help with data processing and visualisation.

2. Reading and Processing Spectral Data

Define a function that reads the spectral data from “[2.07_checked-data.csv](#)” into a DataFrame. The first column represents wavelengths, and the remaining columns represent the spectra.

3. Extracting Prefixes from Spectral Data

Write a function that extracts the polymer names from the column names to help group the spectra by polymer types. You may be able to reuse code from previous tasks.

4. Organising and Normalising the Spectra

Once the data is loaded, group the spectra by polymer. Normalise the intensity values for each spectrum using min-max normalisation. Be careful to avoid division by zero when performing this step. For each prefix, create a combined DataFrame that contains both the normalised intensity values and their corresponding wavelengths.

5. Creating the Heatmap

Construct a pivot table with spectrum indices as rows and wavelengths as columns. Use seaborn to generate a heatmap that visualises the normalised spectra. Ensure you add meaningful titles and axis labels for clarity. Reverse the x-axis so the wavelengths are displayed in line with convention (i.e., larger numbers on the left).

6. Handling Errors and Displaying Results

Include exception handling for errors that may arise during the heatmap creation process and print relevant error messages when necessary.

Hints & Tips for Task 3.02

- Look into seaborn’s heatmap function for proper configuration of axis labels and color scaling.
- Min-max normalisation is key to ensure all spectra are on the same scale — make sure to handle cases where the maximum and minimum values are the same to avoid division by zero errors.
- Think about how you will group spectra by polymer prefix efficiently.

Have you considered?

Outliers may arise for various reasons, such as measurement error, true variation in the data, having a heavily tailed distribution, or the presence of multiple underlying distributions. For example, an outlier might represent a rare but valid observation or could indicate a flaw in the experimental process. However, they can also distort statistical analyses and lead to incorrect conclusions if not properly managed.

If you discover outlier data, how are you going to handle that data? Identifying and deciding how to handle outliers is a critical step in ensuring the integrity of your analysis. For instance, you may choose to disregard data which is deemed as an outlier, or use model-based methods like [Grubbs' test](#) and [Dixon's Q-test](#) can be used to identify outliers, especially when data is assumed to follow a normal distribution. Save any altered data as a CSV file. It is recommended you name your output file “[3.02_outlier-checked-data.csv](#)”.

7.4 Activity 4: Classifying Your Data using a Machine Learning Model

When a computer performs classification, it typically employs statistical methods to develop the algorithm. This involves analysing individual observations to extract quantifiable properties, often referred to as explanatory variables or features. These features can take various forms: categorical (e.g., the type of chemical bond — ionic, covalent, or metallic), ordinal (e.g., the acidity of a solution classified as "strong", "moderate", or "weak"), integer-valued (e.g., the number of carbon atoms in a molecular structure), or real-valued (e.g., the concentration of a solute in a solution, measured in moles per litre).

Some classifiers function by comparing new observations to previously observed data using a similarity or distance function. An algorithm designed for classification is known as a classifier, and this term can also refer to the mathematical function that maps input

data to a specific category. In machine learning, observations are often called instances, features are grouped into a feature vector, and the potential categories to be predicted are called classes. Imagine you want to classify different types of molecules based on their chemical properties, like distinguishing between acids and bases. Each instance is a specific molecule, such as hydrochloric acid (HCl) or sodium hydroxide (NaOH). The feature vector might include attributes such as molecular weight, pH level, dissociation constant (K_a or K_b), and solubility. The classes are "acid" and "base," which are the categories the algorithm aims to predict. In this case, the algorithm uses the chemical properties (features) of each molecule to classify it as either an acid or a base (class), and this mirrors how a classifier works in machine learning.

Classification is a specific instance of the broader field of pattern recognition, which involves assigning an output value to a given input. Other related tasks include regression (assigning a real-valued output to each input), sequence labelling (assigning a class to each member of a sequence, such as part-of-speech tagging in sentences), and parsing (assigning a syntactic structure to a sentence).

No single classification method is universally applicable to all datasets, which is why a wide range of classification algorithms has been developed to address the diverse challenges presented by different types of data.

7.4.1 Task 4.01: Spectral Data Classification Using k-Nearest Neighbors (kNN)

The objective of this task is to classify spectral data using the kNN algorithm. You will split the dataset, train a kNN model, and evaluate the model's performance using metrics such as a confusion matrix and a classification report.

The kNN algorithm is a straightforward yet powerful method used for classification and regression tasks in data analysis. It works by identifying the 'k' closest data points (neighbors) to a given point based on their features. For classification, the algorithm assigns the point to the most common class among its neighbours, while for regression,

it calculates the average of the neighbours' values. This approach is intuitive because it relies on the idea that similar data points tend to be near each other in feature space.

Key Steps:

1. Import Required Libraries

Start by importing the necessary libraries such as scikit-learn for classification, pandas for data handling, numpy for array manipulation, and seaborn/matplotlib for visualisations.

2. Reading and Processing the Spectral Data

Write a function to read “[3.02_outlier-checked-data.csv](#)”. Extract the wavelength (first column) and the spectra (remaining columns). The labels for the spectra are derived from the prefix (before the dot) in the column names. Ensure the wavelength column is **not** treated as a feature.

3. Splitting the Dataset

Divide the dataset into features (spectra) and labels (polymer names). Perform a split (e.g. 50:50) of the data into training and test sets, ensuring that the labels are distributed appropriately between the two sets.

4. Standardising Features

Use [scikit-learn's StandardScaler](#) to standardise the features. Since kNN is a distance-based algorithm, feature scaling is essential for accurate classification.

5. Training the kNN Classifier

Train a kNN model using the training set. Set the number of neighbours. Ensure the model is trained using the standardised feature set.

6. Predicting and Evaluating the Model

Predict the labels of the test set using the trained kNN model. Print how many times each polymer type occurs in the test set for comparison purposes.

7. Visualising the Results

Use seaborn to create a heatmap of the confusion matrix, which will visualise the model's performance in terms of correctly classified spectra vs. misclassifications. Generate a classification report that includes precision, recall, and F1-score for each polymer type.

8. Plotting the Confusion Matrix

Display the confusion matrix using a heatmap, ensuring that the axes are clearly labelled, and the results are easy to interpret (**Figure 5**).

Hints & Tips for Task 4.01

- Exclude the wavelength column when selecting features for classification.
- Standardising the data is crucial for distance-based models like kNN — review how StandardScaler works to ensure proper scaling.
- Scikit-learn has built-in functions for splitting datasets and calculating evaluation metrics like confusion matrices and classification reports.
- Look into seaborn's heatmap function to effectively display the confusion matrix.

```
Occurrences of each polymer type in the test set:
PMMA    7
PET     7
PP      6
PE      6
POM     4
PC      4
PVC     4
Name: count, dtype: int64

Classification Report:
precision    recall   f1-score   support
PC       1.00    1.00    1.00      4
PE       0.60    0.50    0.55      6
PET      1.00    0.86    0.92      7
PMMA    0.86    0.86    0.86      7
POM     0.57    1.00    0.73      4
PP       1.00    1.00    1.00      6
PVC      1.00    0.75    0.86      4

accuracy                           0.84      38
macro avg   0.86    0.85    0.84      38
weighted avg 0.87    0.84    0.84      38
```

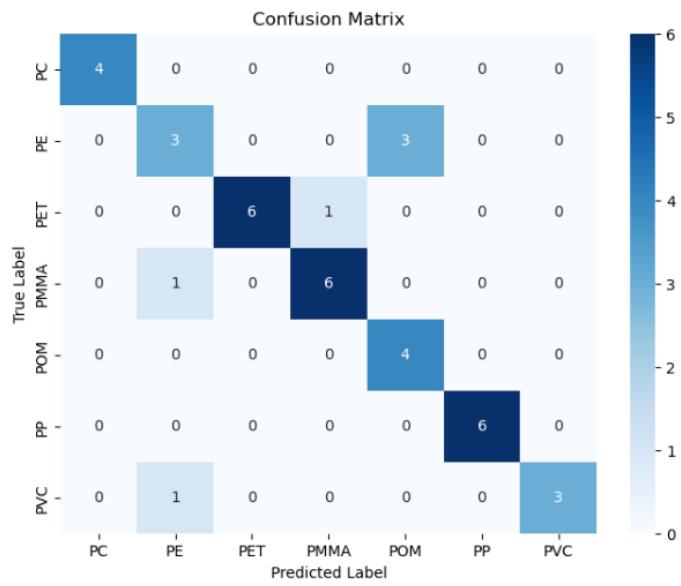


Figure 5: Representative output of classification report and associated confusion matrix.

Have you considered?

- How is the distance calculated between neighbours?
- What is the difference between precision, recall and F-1 score? Which metric are you prioritising and why?
- What are the benefits and limitations of using kNN?
- What happens if we try to analyse a polymer which wasn't in the dataset?

7.4.2 Task 4.02: Building a Weighted-kNN Model

The objective of this task is to classify spectral data using the kNN algorithm. You will split the dataset, train a kNN model, and evaluate the model's performance using metrics such as a confusion matrix and a classification report.

Weighted k-Nearest Neighbors (kNN) is a variation of the kNN algorithm where each neighbour's influence on the prediction is weighted by its distance from the query point. Unlike regular kNN, which gives all neighbours equal weight, weighted kNN makes closer neighbours more influential in the prediction.

Key Steps:

1. Import Required Libraries

Begin by importing the essential libraries such as pandas for data handling, numpy for array manipulation, and machine learning libraries like scikit-learn for classification. You'll also want to include visualization libraries such as seaborn or matplotlib.

2. Reading and Processing Spectral Data

Write a function that reads your CSV file, “[3.02_outlier-checked-data.csv](#)” (ensure it includes wavelengths and spectra data). Separate the wavelength column from the spectra and extract labels (e.g. polymer types) from the column names. Make sure the wavelength column is excluded from the features used for classification.

3. Splitting the Dataset

Split the data into training and test sets, making sure that the labels (such as polymer types) are properly divided. A 50:50 split is a good starting point, but feel free to explore other splits depending on your data size.

4. Standardising the Data

Use StandardScaler to standardise feature data before applying the kNN model. This ensures all features are on the same scale, which is crucial for distance-based models like kNN.

5. Implementing a Weighted kNN Classifier

Implement a weighted kNN classifier where the closer neighbors have more influence on the prediction than those farther away. Use distance-based weighting and ensure the model accounts for prediction certainty.

6. Predicting and Evaluating

After training your kNN model on the training data, predict the labels for the test set. Additionally, incorporate a certainty threshold that determines when the model is confident enough to make a prediction or label it as “Uncertain.”

7. Counting Polymer Types

Count the occurrences of each polymer type in the test set and in your predictions. If there are “Uncertain” labels, ensure you also include these in your count to see how often the threshold is met.

8. Confusion Matrix and Visualization

Create a confusion matrix that compares the true labels with the predicted ones (including "Uncertain" predictions). Use a heatmap to visualise this matrix and evaluate the performance of your model.

9. Classification Report

Generate a classification report that shows precision, recall, and F1-score for each class. This report should also account for the “Uncertain” category, if applicable.

10. Visualising Certainty

Finally, plot a histogram of prediction certainties and mark the threshold with a vertical line. This helps illustrate how confident the model is in its predictions and where the threshold lies in separating certain and uncertain predictions.

7.4.3 Task 4.03: Extension

Now that you have successfully deployed your weighted kNN model, it's time to extend your work by exploring additional avenues. You are encouraged to investigate at least one of the following options:

- Perform a grid search optimisation to identify the optimal kNN parameters.
- Does the polymer colour play a role?
- Explore alternative machine learning classification models, such as:
 - Support Vector Machines (SVM)
 - Random Forest
 - Neural Networks
 - Naïve Bayes

As you explore these extensions, reflect on your choices and consider how they could improve your ability to accurately identify the spectra of unknown samples.

7.5 Activity 5: Identifying Your Unknown Polymers

You have now built your model. It's now time to pick your optimised model and feed into it the spectra of the 10 unknown materials. If you have not yet done so, collect the spectra for these spectra using the PlasTell device. Carry out the appropriate checks using the code from the previous steps. Present your final predictions in an appropriate format.

What is most important at this stage, are not the final predictions but your ability to critically assess the performance of your model.

Hints & Tips for Task 5

- You may directly use your preferred Machine Learning model. Make sure that this code has been trained before deploying onto the unknown samples.
- Use classification metrics like accuracy, precision, recall, and F1-score to evaluate your model and ensure it is operating as previously predicted on the original training data (NOT the unknown data). The confusion matrix provides valuable insight into true positives, false positives, true negatives, and false negatives.
- Plotting the distributions of predicted probabilities can help assess the model's confidence in its predictions.

Have you considered?

- Are there any visual cues from the unknown samples that allow you to speculate on the identity of the polymer?
- Does your model perform as expected?
- What may be causing the model to not work as expected?
- What chemical or physical factors can influence the efficacy of the machine learning model?

8. Section C: Carrying out Chemical Polymer Recycling

In week two of this lab, you will be conducting some synthetic chemistry in Lab360. You will initially be **carrying out the proposed depolymerisation** route of your assigned polymer and characterising the resultant sample.

You will also then **design and perform an extension project** (this does not have to be the same as your lab partner) – *possible examples include:*

- Repolymerisation: isolate pure monomer and use it in a repolymerisation – does the second-generation polymer have similar macro properties to the original material?
- Creation of a new polymer: isolate pure monomer and co-polymerise it with another compound to form a new material – what are its intrinsic properties/how do they compare to the original polymer?
- Depolymerisation optimisation: what modifications can be made to the initial conditions to improve the efficacy of the reaction?
- Improving depolymerisation selectivity: can you vary the reaction conditions so that the depolymerisation results in an increased yield of a desired species?
- Developing a greener recycling route: for example, by using no solvent or a safer/more recyclable alternative such as ionic liquids or deep eutectic solvents?
- Functionalising polymer waste: isolate pure monomer and transform it into a high value commodity of industrial relevance.

Have you considered?

- Could you use the “[Dodecagreen](#)” tool to investigate whether or not your extension procedure is more sustainable than the original?

IMPERIAL

Make sure to think about the **type of characterisation** you will need to carry out to collect and analyse the results you are interested in (e.g. extent of depolymerisation, change in polymer molecular weight, reaction control, monomer purity, polymer macro properties etc.). **Information about the available analytical techniques, and deadlines for samples submissions, can be found in the lab inventory list (ID1 – Equipment List).**

A suggested timeline for week two is shown below...

Monday (11:00-17:00)	Tuesday (09:30-17:00)	Wednesday (no lab)	Thursday (09:30-16:00)	Friday (09:30-15:30)
Depolymerisation and characterisation				
Plan and prepare risk assessment for extension				Extension and characterisation

The **deadline for the submitting the extension risk assessment is 12 NOON on the Wednesday of week two.**

9. Ancillary: Methods of Assessment

Assessment Form (% Contribution)	What is being assessed?	Timing
Written depolymerisation reaction risk assessment and protocol (5 %)	<ul style="list-style-type: none"> Relation of proposed protocol to green chemistry principles Correct identification of risks and hazards Use of a sensible method with clear, well structured (and timed) steps 	<p>Draft submitted prior to start of lab</p> <p>Corrections must be made before start of week two</p>
Good Practice in Jupyter Notebook Submission (5%; GTA assessed)	<ul style="list-style-type: none"> Code reproducibility, runs start to finish Well-structured notebook Explain objectives and steps in Markdown Write clear and readable code Write concise code and avoid unnecessary repetition Clear outputs and visuals 	<p>10 AM Friday of week one (refer to ID1 Good Practice to Jupyter Notebooks doc).</p>
AI demonstration viva, ~ 30 mins (35 %)	<ul style="list-style-type: none"> Efficacy of data handling Understanding of the mechanisations of the code utilised to build the design models Execution of machine learning models for identification of unknown samples Awareness of the strengths and limitations of the applied ML model 	<p>Friday of week one</p>
Final presentation, ~ 30 mins (55 %)	<ul style="list-style-type: none"> Understanding the benefits and limitations of the machine learning models deployed Understanding of the benefits and limitations of the use of GenAI for designing machine learning models Rationale for chosen depolymerisation method, including consideration of environmental impact Critical discussion of synthesis results and analysis, highlighting relevance to sustainable practices and future applications Reflection on the entire experiment Clarity, organisation, and visual appeal of the presentation slides Effectiveness in clearly and confidently explaining complex scientific concepts, engaging the audience and responding to questions 	<p>~10 days after end of week two</p> <p>Slides should be emailed to both lab coordinators at least 48 hours prior to the assessment</p>