University of Canberra
Faculty of Science and Technology

11482 Pattern Recognition and Machine Learning
11512 Pattern Recognition and Machine Learning PG

Tutorial Weeks 1-2
Understanding AI Concepts, Anaconda installation, Python constructs

Objectives

- To understand concepts about AI, pattern recognition and machine learning
- Making sure you have downloaded and installed Anaconda.
- Writing your first Python program.
- Become familiar with the two different ways of interacting with Python.
- Looking at Python data types and operators.

Discussion

- What is AI? Is AI taking over the world? Do we need to be fearful?
- Assume you are leading an AI project? The system you develop will need to be ethical. What do you understand by AI ethics? How will you ensure your system complies with ethics requirements?
- There has been a major resurgence of AI since 2011 and it is being embedded in most walks of life. Discuss the reasons for AI 'winters' and the new directions that make AI as the major leap for the future.
- What is generative AI (eg ChatGPT)? What is it doing? Why does it work? Is it intelligent? why does it hallucinate and say wrong things, give false conclusions? https://chat.openai.com/. Use ChatGPT to write a python program!
- An AI Bard is also a generative AI system developed by Google. Explore it. Get ChatGPT and Bard (https://bard.google.com/) to explain the concept of PRML and prompt them with difficult questions. Which one performs better?
- Pattern recognition (PR) is a major driver for the new AI. How does PR define AI? Identify 4-6 application problems of AI that realise PR.
- Differentiate between PR and machine learning? Discuss the PR/ML cycle of modelling & problem solving.
- Differentiate between regression, classification and clustering.

Lab

Accessing Anaconda:

Note all lab machines will have anaconda installed and you can access it through your virtual access on lab machines or remotely. The student access is through https://frame.nutanix.com/university-of-canberra/

This same link is also on the intranet for both tutorial staff and students.

You could install anaconda on your own laptop as well. A set of instructions is attached under Lab-Setup and a script under Week 1 Module or you could follow the following instructions:

- **Visit https://www.anaconda.com/products/individual#download-section and download the latest version.**
- **Instruction to install: https://docs.anaconda.com/anaconda/install/**
- **Software Installation Demonstration, https://www.kdnuggets.com/2020/02/install-python-anaconda-windows.html**
- **Python IDE Spyder, Jupiter Notebook**
- **Packages Installation https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-pkgs.html**

Simple training of models

Try these in your python environment using the given code.

Exercise 1: Fitting n-degree polynomial to data

Here's an example of fitting an n-degree polynomial to a dataset using the scikit-learn library in Python:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression

# Dataset
X = np.array([1, 2, 3, 4, 5])
y = np.array([2, 4, 6, 8, 10])

# Reshape the feature array
X = X.reshape(-1, 1)

# Define the degree of the polynomial
degree = 3

# Create polynomial features
poly_features = PolynomialFeatures(degree=degree)
X_poly = poly_features.fit_transform(X)

# Create and train the polynomial regression model
model = LinearRegression()
model.fit(X_poly, y)

# Generate data for plotting
X_plot = np.linspace(0, 6, 100).reshape(-1, 1)
X_plot_poly = poly_features.transform(X_plot)
```

```
y_plot = model.predict(X_plot_poly)

# Plot the original data and the fitted polynomial curve
plt.scatter(X, y, label='Original Data')
plt.plot(X_plot, y_plot, color='red', label='Polynomial Fit')
plt.xlabel('X')
plt.ylabel('y')
plt.title('Polynomial Fit of Degree {}'.format(degree))
plt.legend()
plt.show()
```

In this example, we have a simple dataset with five points where the X values are [1, 2, 3, 4, 5] and the corresponding y values are [2, 4, 6, 8, 10].

We first reshape the X values to a column vector, then define the degree of the polynomial we want to fit (in this case, degree=3).

Next, we use the `PolynomialFeatures` class from scikit-learn to create polynomial features up to the specified degree. We transform the original X values to the polynomial features using `fit_transform`.

After that, we create an instance of the `LinearRegression` model and fit it to the transformed polynomial features `X_poly` and the target variable `y`.

To visualize the fitted polynomial curve, we generate a range of X values using `linspace`, transform it to polynomial features using `poly_features.transform`, and then predict the corresponding y values using `model.predict`.

Finally, we plot the original data points and the fitted polynomial curve using matplotlib.

You can change the degree of the polynomial (`degree`) to fit polynomials of different degrees and observe the changes in the curve.

Exercise 2: Fitting n-degree polynomial to data

Here's an example of Python code that builds a linear regression model using the scikit-learn library and trains it on the given dataset:

```
import numpy as np
from sklearn.linear_model import LinearRegression

# Dataset
X = np.array([[1200, 2, 1, 1995],
              [1500, 3, 2, 2002],
              [1800, 3, 2, 1985],
              [1350, 2, 1, 1998],
              [2000, 4, 3, 2010]])

y = np.array([250, 320, 280, 300, 450])
```

```
# Create and train the model
model = LinearRegression()
model.fit(X, y)

# Predict house prices
new_data = np.array([[1650, 3, 2, 2005],
                     [1400, 2, 1, 2000]])

predicted_prices = model.predict(new_data)
print("Predicted prices:", predicted_prices)
```

In this code, we first import the necessary libraries: NumPy for numerical operations and scikit-learn's `LinearRegression` class for building the linear regression model.

Next, we define the dataset `X` and the corresponding target variable `y` using NumPy arrays.

We then create an instance of the `LinearRegression` model and call the `fit` method, passing in the features (`X`) and target variable (`y`). This step trains the linear regression model on the given dataset.

After training the model, we can use it to predict the prices of new houses. We create a new array `new_data` containing the features of the new houses we want to predict. We pass this array to the `predict` method of the model, which returns an array of predicted prices.

Finally, we print the predicted prices of the new houses.