DoC SoC presents

Intermediate Python

Reminders:

- Code can be found on github.com/jackel119/python102
- Slides on docsoc.co.uk/education
- Today we'll be looking at more numpy, pandas, matplotlib, and a little bit of machine learning/AI!

(Recap) NumPy:

- Has a very powerful N-dimensional array object
  - Fast
  - Easy to generate
  - Can enforce types
  - Has TONS of useful methods/operations
- Linear Algebra (and Matrix operations) support
- Other useful functions as well

Enter Pandas:

- Series object, similar to 1-D Numpy Array (actually built on top of it)
- DataFrame object, which represents a table
  - Has column names (which are accessible)
  - Row accessible
  - Again, LOTS of features
- Lots of other useful datatypes (dates, times, etc)
- Combined with Numpy, has anything and everything you will ever need for data processing

What about visualising data?

What about visualising data?
We have **matplotlib**

Numpy, Pandas, MatPlotLib

- Has endless amount of features and functionality
    - If it’s something you want to do, they’ve got it
    - Would take FOREVER to cover everything in class
- Almost every data/maths related library in Python is compatible or built on top of these

Numpy, Pandas, MatPlotLib

- Has endless amount of features and functionality
    - If it’s something you want to do, they’ve got it
    - Would take FOREVER to cover everything in class
- Almost every data/maths related library in Python is compatible or built on top of these
- *Note: this is meant to serve as an introduction to these libraries, not an end-all-be-all. There are an endless amount of tutorials and documentation on the internet, and you should all be at a point where you can make use of them if you so wish.*

A more sophisticated demo

A more sophisticated demo

with a bit of machine learning!

Drug Use Dataset

- We have some (a lot) of data of 1885 people, and for each of them:
  - Age, gender, ethnicity, country, personality traits
  - Their consumption of legal substances e.g. chocolate, nicotine, alcohol, etc...
  - Their consumption of illegal drugs, as well as an overall 'severity' score, etc

Drug Use Dataset

- We have some (a lot) of data of 1885 people, and for each of them:
  - Age, gender, ethnicity, country, personality traits
  - Their consumption of legal substances e.g. chocolate, nicotine, alcohol, etc…
  - Their consumption of illegal drugs, as well as an overall ‘severity’ score, etc
- Let’s explore the data!

Exploratory Data Analysis

- What can we learn from the data?
    - Correlation between how much someone likes chocolate vs how much they drink? Or nicotine (smoking) and coffee?
    - Age and drug use?
    - Certain countries do more drugs?
- Many of you have done R in a scientific concept before - the concept is the same here!

Onto the machine learning bit!

Machine Learning Demo

- This is meant to show you how/what Python can do in the domain of machine learning.
- You will NOT be an expert after this, and may not understand every single thing.
- However, you should be able to follow along most of it, and appreciate the power of Python in machine learning.
- If you want to learn more/do some yourself, there's plenty of great tutorials from the internet!

Machine Learning Intro

- Supervised Learning:
  - Train a model to be able to predict/identify things, i.e. there are ‘right or wrong’ answers - called labeled data.
- Unsupervised Learning:
  - Given some data, have a model tell us about the structure, arrangement of the data, etc..
- Reinforcement Learning:
  - Train a model to make decisions, play games, etc.

Machine Learning Intro

- Supervised Learning:
  - Train a model to be able to predict/identify things, i.e. there are 'right or wrong' answers - called labeled data.
- Unsupervised Learning:
  - Given some data, have a model tell us about the structure, arrangement of the data, etc..
- Reinforcement Learning:
  - Train a model to make decisions, play games, etc.

Supervised Learning

- We might be interested in:
    - Given all the data in our dataset apart from druguse (age, gender, personality, chocolate…), can we predict if someone is a drug user?
        - Or how severe their drug usage is?
    - What about predicting personality traits from other features (drug use, age, country, alcohol, nicotine…)?

Supervised Learning

- We might be interested in:
  - Given all the data in our dataset apart from druguse (age, gender, personality, chocolate...), can we predict if someone is a drug user?
    - Or how severe their drug usage is?
  - What about predicting personality traits from other features (drug use, age, country, alcohol, nicotine...)?

1.) Prepare Data

2.) Create, train, and use the model

Data Preparation

- (Machine Learning/Statistical) Models rely on Maths, and being able to perform calculation on numbers.
- Can you see a problem with our data right now?

Data Preparation

- What does “Male” or “Female” mean?
- Or “UK”, “US”?
- The string “18-24” doesn’t mean anything either!

Data Preparation

- What does “Male” or “Female” mean?
- Or “UK”, “US”?
- The string “18-24” doesn’t mean anything either!
- We have to encode this data numerically!

Data Encoding Approach #1

- Change strings to numerical values e.g.
  - Male = 1, Female = 0 (or vice versa)
  - “18-24” -> 21 (mean), same with other ages, or we might just use 1, 2, 3, 4….

Data Encoding Approach #1

- Change strings to numerical values e.g.
  - Male = 1, Female = 0 (or vice versa)
  - “18-24” -> 21 (mean), same with other ages, or we might just use 1, 2, 3, 4....
  - UK = 1, US = 2, Canada = 3, Other = 4

Data Encoding Approach #1

- Change strings to numerical values e.g.
  - Male = 1, Female = 0 (or vice versa)
  - “18-24” -> 21 (mean), same with other ages, or we might just use 1, 2, 3, 4….
  - UK = 1, US = 2, Canada = 3, Other = 4
    - What’s wrong with this?

Data Encoding Approach #1

UK = 1, US = 2, Canada = 3, Other = 4

- This implies US > UK, Canada > US, that there is an ordering of some sort
- This could mislead our model

Data Encoding Approach #2

"One Hot Encode"

Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

Data Encoding Approach #2

“One Hot Encode”

Traits are now independent, and there is no implied order/hierarchy.

Data Preparation

- We can think of any supervised learning model as trying to estimate a function f(x) = y.
    - x is our predictor(s), usually a vector
    - y is the value(s) we want to predict

Data Preparation

- Think of a student trying to learn a course purely by doing exam papers
    - The first attempt is “blind” - then the student checks his/her answers with the real answers, and that is how they learn. Called “training”.

Data Preparation

- Think of a student trying to learn a course purely by doing exam papers
    - We now want to evaluate how well the student has learned. Obviously, if we use the same exam paper, the student already knows the answers to this. Since we are testing how well a **student has learned the course**, we would give him/her an unseen paper. This is ”**test data**”.

Data Preparation

- Think of a student trying to learn a course purely by doing exam papers
    - In other words, how well does the model we train generalize to data it hasn't seen before?

Data Preparation

- Therefore, we need 4 sets of data:
    - train_x
    - train_y
    - test_x
    - test_y

Data Preparation

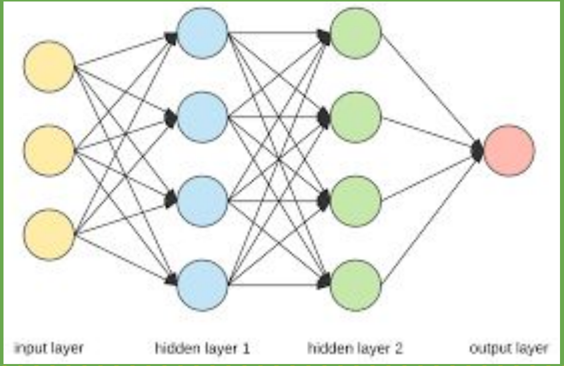
- Therefore, we need 4 sets of data:
    - train_x Training
    - train_y Training
    - test_x We make test predictions on this -> pred_y
    - test_y We compare our pred_y to this to evaluate

Data Preparation

- Therefore, we need 4 sets of data:
    - train_x Training
    - train_y Training
    - test_x We make test predictions on this -> pred_y
    - test_y We compare our pred_y to this to evaluate
    - Train:test split usually around 80:20 or 90:10

Neural Networks

- What you hear about on the news - Deep Learning, AlphaGo, etc…
- “Cool”
- Usually requires lots of computational power

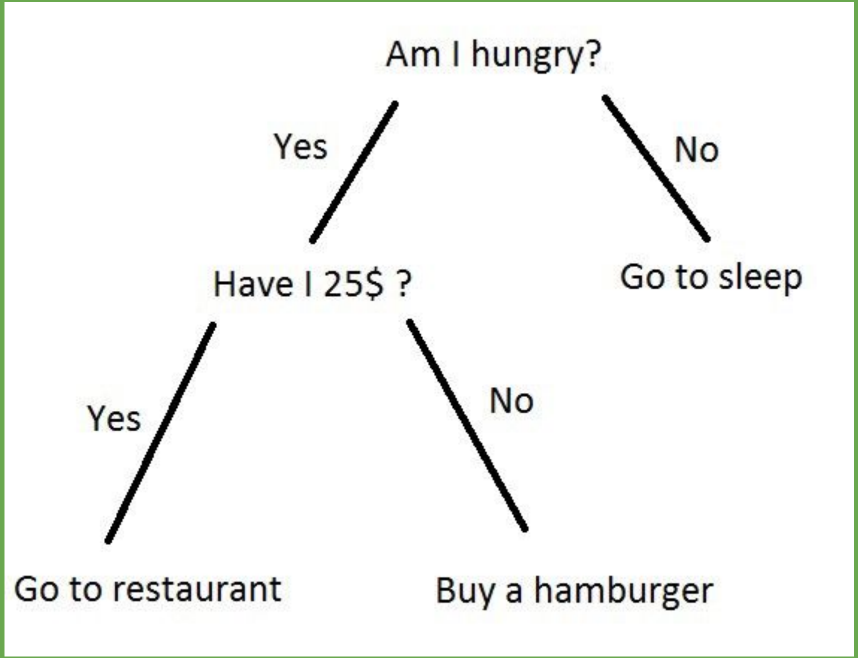


Classical Models

- Lots of different types:
    - Linear Regression, Logistic Regression, Decision Trees, Random Forests, Matrix Factorization, K-Means Clustering
- “Old” Machine Learning
- Not as computationally expensive, and still usually VERY good results!

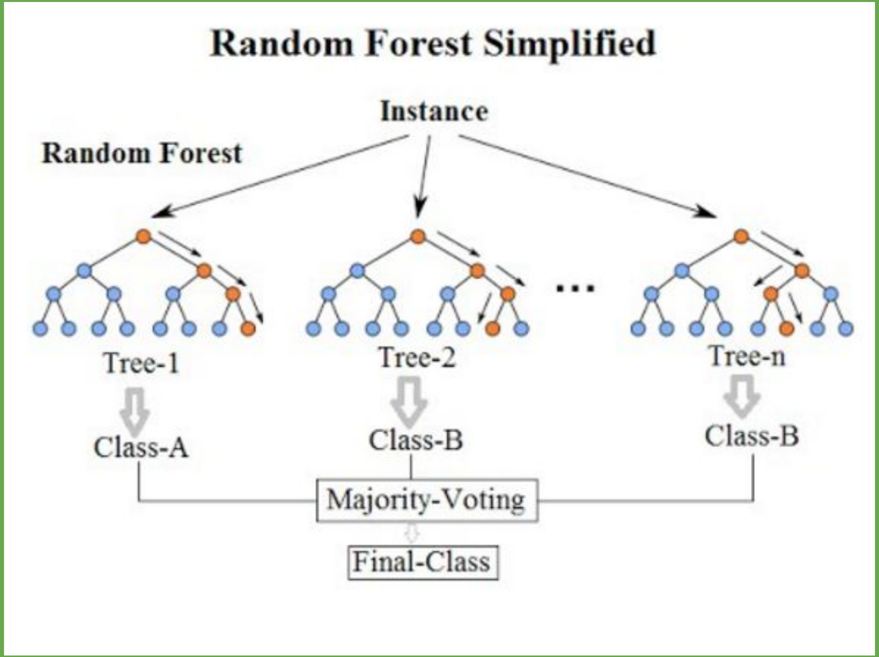
Decision Trees

- Algorithms to generate decision trees based on “information entropy” (what can we find out with a true/false question?).
- In real cases, the questions are **“is feature N > value?”**



Random Forests

Basically....a lot of trees that vote on what y (the prediction) should be!



Quick note about types used in models

- Different Machine Learning libraries will support different types, have different functions, arguments (**the interface!**).
- Most, if not all, support input/output as Numpy arrays!
- We will first look at Sci-kit-learn.
- `pip install sklearn`

Model #1

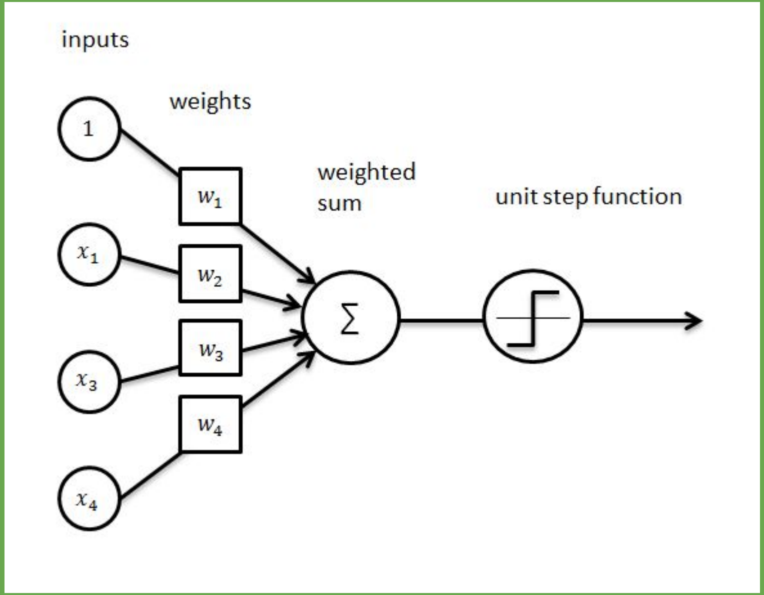
```python
from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor()
rf.fit(train_x, train_y)

pred_y = rf.predict(test_y)

# Now compare pred_y and actual test_y
```

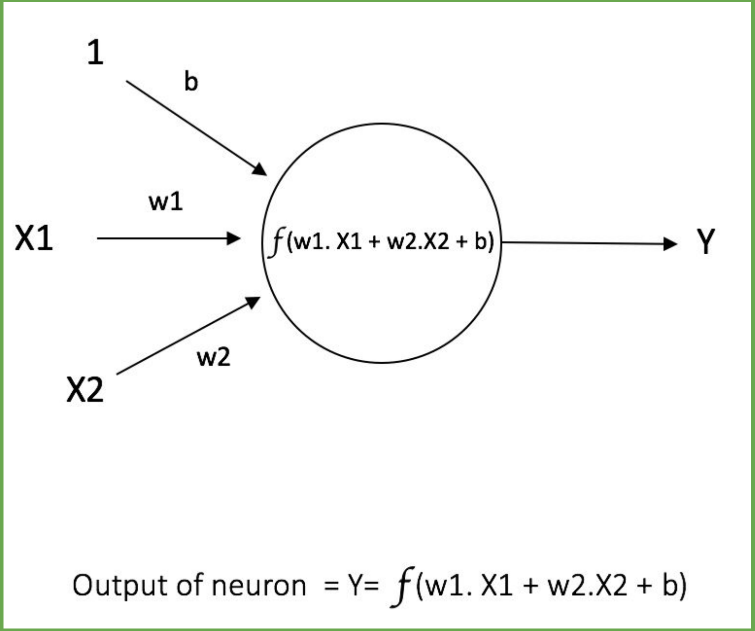
Looking at and evaluating results

- Can just look and compare each prediction and actual result individually
- Metrics like Root Mean Squared Error, Mean Absolute Error, Percentage Error, etc....
    - Mostly basic statistics

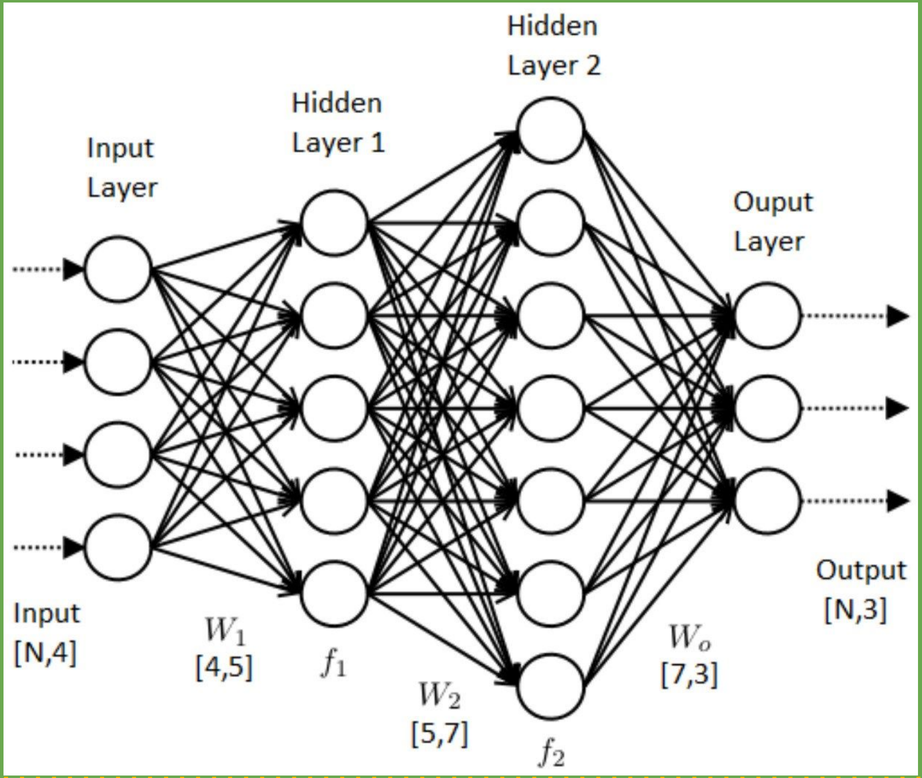
Neural Networks Explained (Very quickly!)

- Consists of Layers of “Neurons”, which take in numerical vectors





Neural Networks Explained (Very quickly!)



Neural Networks Explained (Very quickly!)

- If you want to know more:
  - https://blog.goodaudience.com/artificial-neural-networks-explained-436fcf36e75
  - http://neuralnetworksanddeeplearning.com/chap1.html
  - Lots of good resources online!
  - “Optimizers”, “Loss function”, “Activation Function”, etc....

Model #2

```python
from keras.models import Sequential
from keras.layers.core import Dense, Activation

model = Sequential()
model.add(Dense(32, input_shape=(28,), activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(train_x, train_y, nb_epoch=100, batch_size=1)

pred_y = model.predict(test_x).reshape(len(test_x))
```

Thanks for coming!

- Next week:
  - HTTP Requests, web servers, scripting...
  - Possibly more!
- jackel119/python102 on Github for code and csv data
- docsoc.co.uk/education for slides