Machine Learning – Supervised Learning Notes

Definition and what problems it's used for

- Supervised learning is the subcategory of machine learning where labelled data sets are used to train models for various uses
- These models have training sets, where inputs are fed into the model with correct outputs, and the model is trained on this by adjusting weights as it receives more data, where the adjustments are made to minimise some predetermined loss function
- Broadly speaking supervised ML splits into two categories – first classification problems, where models will attempt to predict an outcome, or the target, from the input data, sometimes called the feature. Some examples of this include looking at features such as credit history, loans etc. and the output being whether the person has defaulted in the past.
- Secondly, regression problems – this is using a model to predict relationships between features and continuous data, such as house prices from location, number of bedrooms etc.

Binary Classification

- Binary classification is a supervised ML technique used to classify observations into one of two categories (say 0, 1) – straightforward examples include classifying email as spam or not, transactions as fraudulent or not, patient diagnoses as healthy or sick and images as dog or not.
- Evaluation of binary classification comes into 4 different categories:

	Model says 1	Model says 0
Data is actually 1	True Positive (TP)	False Negative (FN)
Data is actually 0	False Positive (FP)	True Negative (TN)

- This is the confusion matrix – the accuracy of the model is given by $\frac{TP+TN}{TP+FP+FN+TN}$
- You can also get the precision = $\frac{TP}{TP+FP}$, sensitivity = $\frac{TP}{TP+FN}$ and specificity = $\frac{TN}{FP+TN}$
- There are various types of models that are used in binary classification –
  - One simple one is logistic regression on the data – if the output is greater than 0.5, it is 1, otherwise it is 0.
  - K-nearest Neighbours – define a distance metric between points; for each new example, look at the nearest k points, whatever the majority classification is between them, that is the classification of this point – not great with large datasets/many dimensions
  - Naïve Bayes – apply the 'naïve' assumption that every pair of data is independent of each other given a specific class. Then the $P(x_1, \dots, x_n|y) = \prod_{i=1}^{n} P(x_i|y)$, so from Bayes' theorem $P(y|x_1, \dots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)$, where y is the class. Choose the y that maximises this term.
  - Support Vector Machines (SVM) – fairly in depth, essentially tried to create a series of hyperplanes that separate the data, with the aim of maximising the distance between the hyperplanes and the nearest data point either side of the plane (margin)
  - Some form of neural network; standard architectures discussed below
- One common issue is class imbalance, which can often bias the model towards the majority class – resample whilst under sampling the majority and oversampling the minority. Also, neural networks will need regularization etc.

Standard Network Architectures

- Neural network architecture refers to how the neurons and layers are designed to work/interface with each other. Key components include:
  - Input data – the data that is fed into any one neuron
  - Weights and bias – the various weightings for the different input data, and the bias is the additional shifting to the data

  - Activation function – after receiving the weighted input, the value goes through the activation function which produces the output – usually non-linear to produce non-linear output
  - Input/output layer – the initial/final data we have/get
  - Hidden layers – intermediate layers where all the computations and non-linearity comes from - sometimes layers are ordered to go from higher level features to least

- Standard networks include the following:
  - MLP – multi layer perceptron – simplest type of feedforward NN, each layer goes through weighting, activation function and repeat till you get the output – exactly how the neuron works above – trained with backpropagation
  - RNN – recurrent neural networks – the hidden layers have an element of recurrence, i.e. they rely not only on the data provided by the most recent layer but also previous layers – trained with backpropagation through time (BPTT) , have the same weighting for every layer
  - CNN – convolutional neural networks – specifically designed for structured grid-like data, e.g. images or sequential data. Made up of convolutional layers, which apply filters/kernels to the data to classify features (for example searching for eyes/ears etc.) – then there are pooling layers, which reduce the dimension of the problem by retaining the most relevant features – then repeat with different filters etc. until you have a final fully connected layer, which is a classification problem as before.
  - UNets – designed for image segmentation originally, designed as a CNN on a 'contracting' path, also called the encoder, followed by an 'expanding' path, which reconstructs the resolution lost in pooling in the encoder, also called the decoder. The contracting and expanding shape is why it's called 'U'Nets. The transition between the encoder and decoder is the bottleneck. Upsampling is done by up-convolutions.

Diagrams provided on separate document for more clarity.