# Week 10: Machine Learning

For Distance Metrics, Chapter 10, and the python notebook "lab10_NN.ipynb" covers Distance Measures, Nearest Neighbor Algorithm for classification, and K-Means for clustering.

For Various Machine Learning Algorithms, chapter 11 covers various ML algorithms. The accompanying python notebook "lab11_ML_2.ipynb" is available in surrey learn. However, the book focuses on how to compare these algorithms, rather than how to use them. COMM055 next semester will cover these topics, but this lab serves as introduction and material you might like to refer back to in the future. Comparing the performance of the various methods subjectively, the author produced the following table:

| **Method** | **Power of Expression** | **Ease of Interpretation** | **Ease of Use** | **Training Speed** | **Prediction Speed** |
|---|---|---|---|---|---|
| Linear Regression | 5 | 9 | 9 | 9 | 9 |
| Nearest Neighbor | 5 | 9 | 8 | 10 | 2 |
| Naïve Bayes | 4 | 8 | 7 | 9 | 8 |
| Decision Trees | 8 | 8 | 7 | 7 | 9 |
| Support Vector Machines | 8 | 6 | 6 | 7 | 7 |
| Boosting | 9 | 6 | 6 | 6 | 6 |
| Graphical Models | 9 | 8 | 3 | 4 | 4 |
| Deep Learning | 10 | 3 | 4 | 3 | 7 |

*Figure 1: Subjective rankings of machine learning approaches along five dimensions, on a 1 to 10 scale with higher being better.*

From Lab 9, it was recommended that the scikit-learn python library offer various tutorials on various topics for machine learning:
https://scikit-learn.org/stable/

To support chapter 11 contents, it is advised to do the following tutorials, and research the associated concepts:

1. Naïve Bayes and concepts such as discounting techniques,
https://scikit-learn.org/stable/modules/naive_bayes.html

2. Decision Tree Classifiers: It is interesting to read section 11.2.1 on how to construct the decision tree, various partitioning methods, pruning methods, how they handle non-linearity, and Ensembles. Then go through the various algorithms in this tutorial:

https://scikit-learn.org/stable/modules/tree.html

3. Ensemble Learning, and concepts such as finding weights of classifiers and using boosting:
https://scikit-learn.org/stable/modules/ensemble.html

4. SVM, and concept of deciding the largest margin, how non-linearity is handled, how multiclass classifier can be built, and the use of kernels to avoid computing a distance matrix for all points :

https://scikit-learn.org/stable/modules/svm.html

5. Unsupervised Modelling such as: latent Dirichlet allocation (LDA), K-Means

https://medium.com/mlreview/topic-modeling-with-scikit-learn-e80d33668730
https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.LatentDirichletAllocation.html

https://scikit-learn.org/stable/modules/clustering.html#k-means
https://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering

6. Deep Learning and concepts such that feature engineering is not required anymore, suitability to huge datasets, use of overfitting, less precise ways to encode knowledge to solve overfitting problems, depth and backpropagation, and handling non-linearity by various activation functions:

https://scikit-learn.org/stable/modules/neural_networks_supervised.html

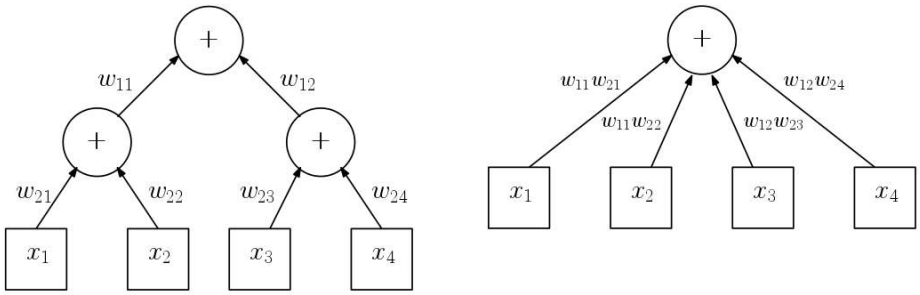


*Figure 2: Addition networks do not benefit from depth. The two layer network (left) computes exactly the function as the equivalent one layer network (right). Depth of the network should correspond to the conceptual hierarchy associated with the objects being modelled. For example: modelling images, will require layers of abstraction to represent pixels, neighbourhood patches, edges, textures, regions, simple objects, compound objects, and scenes each on separate layers..*

However, TensorFlow, PyTorch, and Keras are now more powerful in deep learning models, with plenty of online tutorials and references. The following is an easy introduction to these Models:
https://www.kdnuggets.com/2016/02/scikit-flow-easy-deep-learning-tensorflow-scikit-learn.html

For Reinforcement Learning, the following articles discusses the various python libraries:
https://medium.com/data-from-the-trenches/choosing-a-deep-reinforcement-learning-library-890fb0307092

Example:

| Day | Outlook | Temp | Humidity | Beach? |
|---|---|---|---|---|
| 1 | Sunny | High | High | Yes |
| 2 | Sunny | High | Normal | Yes |
| 3 | Sunny | Low | Normal | No |
| 4 | Sunny | Mild | High | Yes |
| 5 | Rain | Mild | Normal | No |
| 6 | Rain | High | High | No |
| 7 | Rain | Low | Normal | No |
| 8 | Cloudy | High | High | No |
| 9 | Cloudy | High | Normal | Yes |
| 10 | Cloudy | Mild | Normal | No |

| P(X\|Class) | Probability in Class | |
|---|---|---|
| **Outlook** | **Beach** | **No Beach** |
| Sunny | 3/4 | 1/6 |
| Rain | 0/4 | 3/6 |
| Cloudy | 1/4 | 2/6 |

| **Temp** | **Beach** | **No Beach** |
|---|---|---|
| High | 3/4 | 2/6 |
| Mild | 1/4 | 2/6 |
| Low | 0/4 | 2/6 |

| **Humidity** | **Beach** | **No Beach** |
|---|---|---|
| High | 2/4 | 2/6 |
| Normal | 2/4 | 4/6 |

| **P(Beach Day)** | **4/10** | **6/10** |
|---|---|---|

*Figure 3: Probabilities to support a naive Bayes calculation on whether today is a good day to go to the beach: tabulated events (left) with marginal probability distributions (right).*

For example: P (Beach|(Sunny,Mild,High))
= (P (Sunny|Beach) × P (Mild|Beach) × P (High|Beach) × P (Beach) = (3/4) × (1/4) × (2/4) × (4/10) = 0.0375

Decision Tree sample output, pruning branches or values of close to equal probability on both decisions:

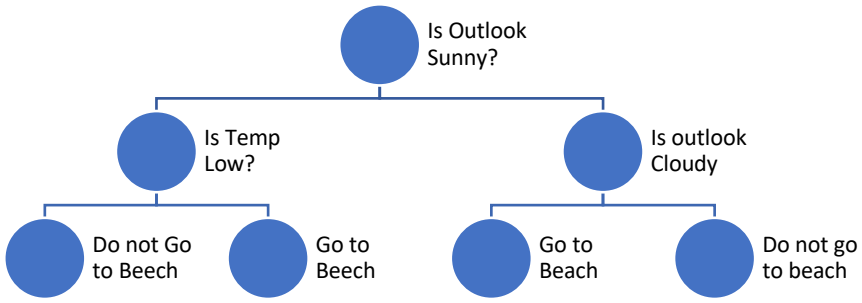


*Figure 4: Decision Tree Output*

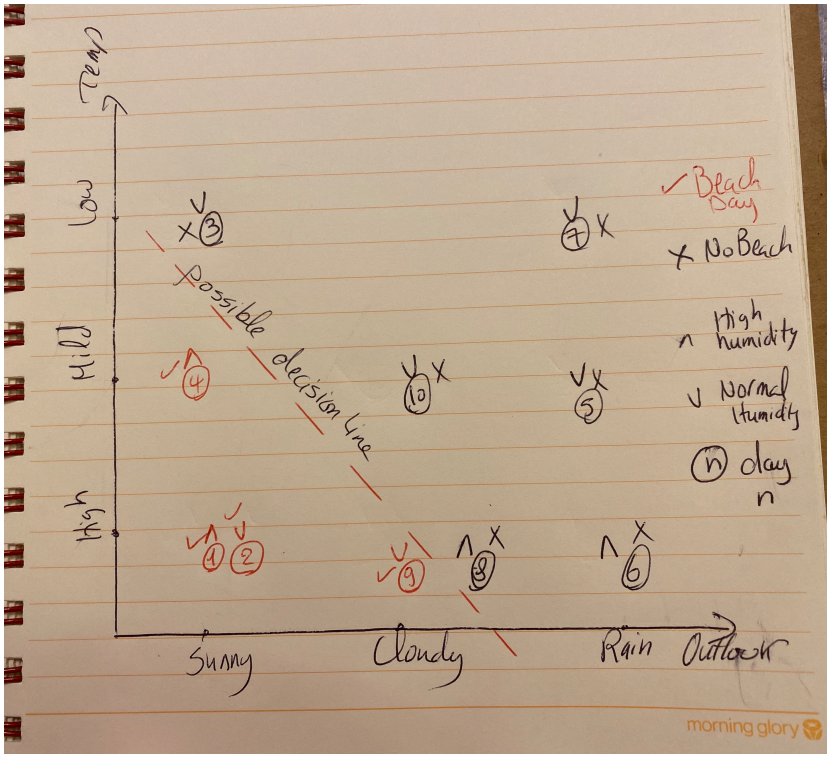


*Figure 5: Logistic Regression or SVM or generally decision line methods possible output*

As an exercise, attempt the Kaggle Challenge to Predict which product a consumer is most likely to buy, and discuss on Surrey Learn:

https://www.kaggle.com/c/coupon-purchase-prediction