ETGG 4803 **Lab #7**
80 points **Due Date: 12 April**

Outcomes:

- Implement simple probability-based binary classifiers.
- Implement the distance-based algorithm NN, *k*NN, and IB1.
- Understand pros and cons of each of the algorithms.

Download the Occupancy Detection data set:
https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+
Different forms of preprocessing will be necessary for implementation of different algorithms.

1. A simple statistical classifier SSC
    - **a.** For each classification category in the training set, calculate the mean of each attribute.
    - **b.** Given a test data point ***t***, determine the distance between the attribute value for ***t*** and the means for each category. Assign one vote to the category for which the test value is closest to the mean.
    - **c.** Calculate the vote total and assign ***t*** to the category which receives the most votes. (Bonus: Use training data to determine weights for each attribute and calculate a weighted linear combination of the tests to determine classification rather than doing a simple voting procedure.)
2. The naïve Bayesian classifier BC
    - **a.** For each attribute in the training data, separately determine the range of values for each category. (If there is no overlap in the data values for each category, then the rest of this classifier is much simpler.)
    - **b.** For each attribute, use Bayes's Theorem to calculate $P(C \mid A)$ for each classification category C and each attribute interval A. (In binary classification, we need calculate the probability for only one of the categories.)
    - **c.** Given a test vector ***t***, use Bayes's Theorem to calculate $P(C \mid \boldsymbol{t})$ for each category C. (Assume independence of the attributes.)
    - **d.** Assign ***t*** to the most probable category. (This is called the *maximum a posteriori* (MAP) decision rule.)
3. The nearest-neighbour classifier NN
    - **a.** Treat each data point as an n-dimensional vector, where n is equal to the number of attributes in the data. To classify a test instance ***t***, find the training instance closest to ***t*** using the Euclidean distance and assign ***t*** to the same class as this training instance. If multiple training instances give the same smallest distance, randomly select one.
    - **b.** Modify your implementation to allow the user to specify 1-norm, 2-norm, or the infinity norm.

4. The *k*-nearest-neighbour classifier *k*NN
    a. To classify a test instance ***t***, find the *k* training instances closest to ***t*** using the Euclidean distance. The training instance ***t*** is assigned to the class of the majority of the nearest *k* training instances. If there is no majority, a plurality may be used. If there is a tie, randomly select amongst the classes with most votes.

    b. Modify your implementation to allow the user to specify 1-norm, 2-norm, or the infinity norm.

5. The instance-based algorithm IB1 (Aha, Kibler, and Albert, 1991)
    a. Given two *n*-dimensional data vectors ***v*** and ***w***, define

$$\text{Similarity}(\boldsymbol{v}, \boldsymbol{w}) = -\sqrt{\sum_{i=1}^{n} f(v_i - w_i)}$$

    where $f(v_i, w_i) = (v_i - w_i)^2$. If an attribute value is missing in one vector, assign the missing value such that the difference $|v_i - w_i|$ is as large as possible in the context of the data. If both values are missing, assign $f(v_i, w_i) = 1$.

    b. To classify a test instance ***t***, find the training instance most similar to ***t*** and assign ***t*** to the same class as this training instance. If multiple training instances give the same Similarity, select the training instance of that Similarity found first.

6. Apply each of the algorithms to the Occupancy Detection data set.
    a. (50 points) Make a table showing the percentage of successful classification for the first test set and the percentage of successful classification for the second test set for each algorithm.

    b. (Essay, 30 points) Compare the results of each of the algorithms. Some questions to consider: What are the advantages of each algorithm over the others? What are the disadvantages of each algorithm? Which seems most appropriate for the given data set? Be sure to include an analysis of what happens in the algorithms for Problem #3 and Problem #4 using different norms.