Data Mining and Machine Learning

Comp 3027J

Dr Catherine Mooney
Assistant Professor

catherine.mooney@ucd.ie

Lectures and Text

- **Core Text:**
  Fundamentals of Machine Learning for Predictive Data Analytics
  By John D. Kelleher, Brian Mac Namee and Aoife D'Arcy
- Last week we covered Chapter 4, sections 4.2 and 4.3 (Information-based Learning – Decision Trees and The ID3 Algorithm).
- This week we will cover Chapter 4, sections 4.4.2, 4.4.4 and 4.4.5 (Handling Continuous Descriptive Features, Tree Pruning and Model Ensembles).
- Please read these sections of the book.

1. **Information-based Learning – Decision Trees**
2. **Handling Continuous Descriptive Features**
3. **Noisy Data, Overfitting and Tree Pruning**
4. **Model Ensembles**
5. **Advantages/Disadvantages**
6. **Preview of Lab 6**

Information-based Learning – Decision Trees

- A decision tree consists of:
  1. a **root node** (or starting node),
  2. **interior nodes**
  3. and **leaf nodes** (or terminating nodes).
- Each of the non-leaf nodes (root and interior) in the tree specifies a test to be carried out on one of the query's descriptive features.
- Each of the leaf nodes specifies a predicted classification for the query.

- It is often possible to build many different decision trees that are consistent with the data.
- We should chose decision trees that use less tests (shallower trees).

How do we create shallow trees?

- Descriptive features that split the dataset into pure sets with respect to the target feature provide information about the target feature.
- So we can make shallow trees by testing the informative features early on in the tree.
- To do this we need a computational metric of the purity of a set: entropy

Claude Shannon’s Entropy Model

- Claude Shannon’s entropy model defines a computational measure of the impurity of the elements of a set.
- An easy way to understand the entropy of a set is to think in terms of the uncertainty associated with guessing the result if you were to make a random selection from the set.

- Entropy is related to the probability of a outcome.
  - High probability $\rightarrow$ Low entropy
  - Low probability $\rightarrow$ High entropy
- If we take the log of a probability and multiply it by -1 we get this mapping!

- Shannon's model of entropy is a weighted sum of the logs of the probabilities of each of the possible outcomes when we make a random selection from a set.

$$H(t) = -\sum_{i=1}^{l} (P(t = i) \times log_2(P(t = i)))$$

Information Gain

- The information gain of a descriptive feature can be understood as a measure of the reduction in the overall entropy of a prediction task by testing on that feature.

Computing information gain is a three-step process:

1. Compute the entropy of the original dataset with respect to the target feature.
    - This gives us an measure of how much information is required in order to organize the dataset into pure sets.
2. For each descriptive feature, create the sets that result by partitioning the instances in the dataset using their feature values, and then sum the entropy scores of each of these sets.
    - This gives a measure of the information that remains required to organize the instances into pure sets after we have split them using the descriptive feature.
3. Subtract the remaining entropy value (computed in step 2) from the original entropy value (computed in step 1) to give the information gain.

- We need to define three equations to formally specify information gain (one for each step).

The first equation calculates the entropy for a dataset with respect to a target feature:

$$H(t, \mathcal{D}) = - \sum_{l \in levels(t)} (P(t = l) \times log_2(P(t = l)))$$

where *levels(t)* is the set of levels in the domain of the target feature *t*, and *P(t = l)* is the probability of a randomly selected instance having the target feature level *l*.

The second equation defines how we compute the entropy remaining after we partition the dataset using a particular descriptive feature *d*.

$$rem(d, \mathcal{D}) = \sum_{l \in levels(d)} \underbrace{\frac{|\mathcal{D}_{d=l}|}{|\mathcal{D}|}}_{\text{weighting}} \times \underbrace{H(t, \mathcal{D}_{d=l})}_{\substack{\text{entropy of} \\ \text{partition } \mathcal{D}_{d=l}}}$$

The weighting is determined by the size of each partition – so a large partition should contribute more to the overall remaining entropy than a smaller partition.

The third equation – Using Equation 1 and Equation 2, we can now calculate the information gain made from splitting the dataset $\mathcal{D}$ using the feature d

$$IG(d, \mathcal{D}) = H(t, \mathcal{D}) - rem(d, \mathcal{D})$$

Handling Continuous Descriptive Features

- The easiest way to handle continuous valued descriptive features is to turn them into **boolean features** by defining a threshold and using this threshold to partition the instances.
- How do we set the threshold?

1. The instances in the dataset are sorted according to the continuous feature values.
2. The adjacent instances in the ordering that have different classifications are then selected as possible threshold points.
3. The optimal threshold is found by computing the information gain for each of these classification transition boundaries and selecting the boundary with the highest information gain as the threshold.

- Once a threshold has been set the new boolean feature can compete with the other categorical features for selection as the splitting feature at that node.
- This process can be repeated at each node as the tree grows.

Dataset for predicting the vegetation in an area with a continuous ELEVATION feature (measured in feet).

ID	STREAM	SLOPE	ELEVATION	VEGETATION
1	false	steep	3 900	chapparal
2	true	moderate	300	riparian
3	true	steep	1 500	riparian
4	false	steep	1 200	chapparal
5	false	flat	4 450	conifer
6	true	steep	5 000	conifer
7	true	steep	3 000	chapparal

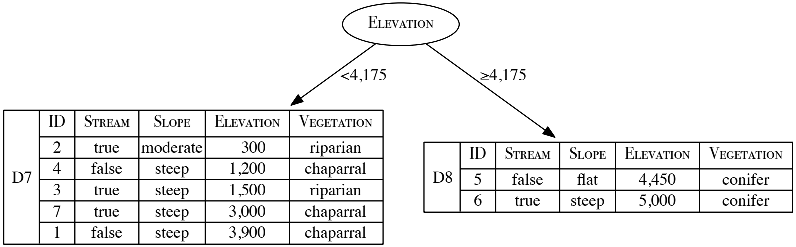
Dataset for predicting the vegetation in an area sorted by the continuous ELEVATION feature.

ID	STREAM	SLOPE	ELEVATION	VEGETATION
2	true	moderate	300	riparian
4	false	steep	1 200	chapparal
3	true	steep	1 500	riparian
7	true	steep	3 000	chapparal
1	false	steep	3 900	chapparal
5	false	flat	4 450	conifer
6	true	steep	5 000	conifer

The adjacent instances in the ordering that have different classifications are then selected as possible threshold points.

Split by Threshold	Part.	Instances	Partition Entropy	Rem.	Info. Gain
$\geq$750	$\mathcal{D}_1$	$\mathbf{d}_2$	0.0	1.2507	0.3060
	$\mathcal{D}_2$	$\mathbf{d}_4, \mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.4591		
$\geq$1350	$\mathcal{D}_3$	$\mathbf{d}_2, \mathbf{d}_4$	1.0	1.3728	0.1839
	$\mathcal{D}_4$	$\mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.5219		
$\geq$2250	$\mathcal{D}_5$	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_3$	0.9183	0.9650	0.5917
	$\mathcal{D}_6$	$\mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.0		
$\geq$4175	$\mathcal{D}_7$	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1$	0.9710	0.6935	0.8631
	$\mathcal{D}_8$	$\mathbf{d}_5, \mathbf{d}_6$	0.0		

The optimal threshold is found by computing the information gain for each of the classification transition boundaries ($\geq$750, $\geq$1,350, $\geq$2,250 and $\geq$4,175) and selecting the boundary with the highest information gain as the threshold.



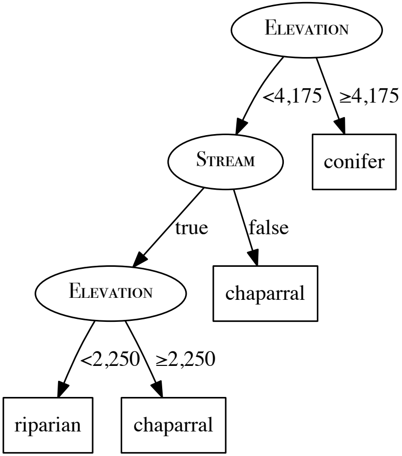
D7

ID	STREAM	SLOPE	ELEVATION	VEGETATION
2	true	moderate	300	riparian
4	false	steep	1,200	chaparral
3	true	steep	1,500	riparian
7	true	steep	3,000	chaparral
1	false	steep	3,900	chaparral

D8

ID	STREAM	SLOPE	ELEVATION	VEGETATION
5	false	flat	4,450	conifer
6	true	steep	5,000	conifer

The vegetation classification decision tree after the dataset has been split using ELEVATION $\geq 4\,175$



The decision tree that would be generated for the vegetation classification dataset using information gain.

Noisy Data, Overfitting and Tree Pruning

Overfitting

- In the case of a decision tree, overfitting involves splitting the data on an irrelevant feature.
- The likelihood of overfitting occurring increases as a tree gets deeper because the resulting classifications are based on smaller and smaller subsets of the dataset.

Pre-pruning

- **Pre-pruning**: stop the recursive partitioning early.
- Pre-pruning is also known as **forward pruning**.
- Common Pre-pruning Approaches:
  1. **early stopping**
  2. χ^2 **pruning**

Post-pruning Approach

- **Post-pruning**: allow the algorithm to grow the tree as much as it likes and then prune the tree of the branches that cause overfitting.
- Using the validation set evaluate the prediction accuracy achieved by both the fully grown tree and the pruned copy of the tree.
- If the pruned copy of the tree performs no worse than the fully grown tree the node is a candidate for pruning.

Advantages of pruning:

- Smaller trees are easier to interpret
- Increased generalization accuracy when there is noise in the training data (**noise dampening**).

Model Ensembles

- Create a set of models and then make predictions by aggregating the outputs of these models.
- A prediction model that is composed of a set of models is called a **model ensemble**.
- In order for this approach to work the models that are in the ensemble must be different from each other.

- There are two standard approaches to creating ensembles:
  1. **boosting**
  2. **bagging**.

- **Boosting** works by iteratively creating models and adding them to the ensemble.
- The iteration stops when a predefined number of models have been added.
- Each new model added to the ensemble is biased to pay more attention to instances that were miss-classified by previous models.
- This is done by incrementally adapting the dataset used to train the models.
- To do this we use a **weighted dataset**

Weighted Dataset

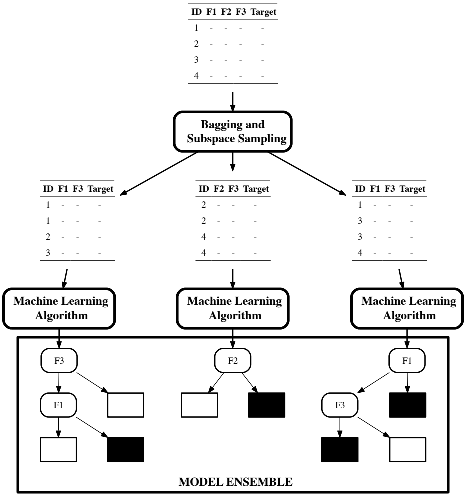
- Each instance has an associated weight $\mathbf{w}_i \geq 0$,
- Initially set to $\frac{1}{n}$ where n is the number of instances in the dataset.
- After each model is added to the ensemble it is tested on the training data and the weights of the instances the model gets correct are decreased and the weights of the instances the model gets incorrect are increased.
- These weights are used as a distribution over which the dataset is sampled to created a replicated training set, where the replication of an instance is proportional to its weight.

- Once the set of models have been created the ensemble makes predictions using a weighted aggregate of the predictions made by the individual models.
- The weights used in this aggregation are the confidence factors associated with each model.

- When we use **bagging** (or **bootstrap aggregating**) each model in the ensemble is trained on a random sample of the dataset known as **bootstrap samples**.
- Each random sample is the same size as the dataset and **sampling with replacement** is used.
- Consequently, every bootstrap sample will be missing some of the instances from the dataset so each bootstrap sample will be different and this means that models trained on different bootstrap samples will also be different

- Frequently, when bagging is used with decision trees, the sampling process is extended so that each bootstrap sample uses a randomly selected subset of the descriptive features.
- This is known as **subspace sampling**.
- Subspace sampling further encourages the diversity of the trees within the ensemble and has the advantage of reducing the training time for each tree.

- The combination of bagging, subspace sampling, and decision trees is known as a **random forest** model.
- Once the individual models have been induced, the ensemble makes predictions by returning the majority vote or the median depending on the type of prediction required.



A **Random Forest** Model.

- Which approach should we use?
- Bagging is simpler to implement and parallelize than boosting and, so, may be better with respect to ease of use and training time.

Advantages/Disadvantages

Decision Trees: Advantages

- Interpretable.
- Handle both categorical and continuous descriptive features.
- Has the ability to model the interactions between descriptive features (diminished if **pre-pruning** is employed)
- Relatively, robust to the **curse of dimensionality**.
- Relatively, robust to noise in the dataset if **pruning** is used.

Decision Tress: Potential Disadvantages

- Trees become large when dealing with continuous features.
- Decision trees are very expressive and sensitive to the dataset, as a result they can overfit the data if there are a lot of features (curse of dimensionality)
- Eager learner (concept drift).

Recommended Reading

- **Core Text:**
  Fundamentals of Machine Learning for Predictive Data Analytics
  By John D. Kelleher, Brian Mac Namee and Aoife D'Arcy
- This week we covered Chapter 4, sections 4.4.2, 4.4.4 and 4.4.5 (Handling Continuous Descriptive Features, Tree Pruning and Model Ensembles).
- I would suggest that you would read over these sections again.
- Email me if you have any questions and I will cover them at the beginning of class next week.
- Next week we will cover Chapter 7, sections 7.2 and 7.3 (Error-based Learning – Linear Regression and Gradient Descent).

Preview of Lab 6

Preview of Lab 6

- In Lab 6 we will be using R for Information-based Learning (Random Forest)
- We will also use a number of different Performance Measures to evaluate your Random Forest
- You should download the following packages in advance:
  - install.packages("randomForest")
  - install.packages("class")
  - install.packages("e1071")
  - install.packages("caret")