COMP10200: ML Unplugged Debrief

Task 1 Solution (Sets of Symbols)

The rule used to generate the labels for the training data was the following:

Rule 1: The "Correct" Theory

```
IF number of colors == 1 OR two shapes, both ovals THEN IN
OTHERWISE OUT
```

Applying this rule to the testing set, the correct labels are:

1. OUT
2. IN
3. OUT
4. IN
5. IN
6. IN
7. OUT
8. IN
9. OUT
10. OUT
11. OUT
12. IN
13. IN
14. OUT
15. IN

Questions

1. How accurate was your theory on the training data?

2. How accurate was your theory on the testing data?

3. Could the training data be improved to help steer a learner towards the correct theory?

Task 1 Debrief (Sets of Symbols)

If you did not score 100% correct, don't be disappointed. Machine learning algorithms usually don't get to anywhere near that level of accuracy. This was meant to be a hard task that highlights a number of important issues in Machine Learning.

The Importance of the Training Data

In many cases the available training data is consistent with multiple theories. More training data is usually better, but it's also important to make sure (if you can) that the training set contains a wide and balanced variety of examples. And when two available theories seem just as good as one another, you need some way to choose between them.

The training data in this task is also consistent with Rule 2 below.

Rule 2: Alternate Match for Training Data

```
IF one shape THEN IN
ELSE IF two shapes, both ovals THEN IN
ELSE IF purple present THEN OUT
ELSE IN
```

This rule scores 100% on the training set, but only 80% on the testing set (it gets 5, 6, and 7 wrong). A better variety of training data (i.e. examples like number 5 in the testing set) might have made this theory seem less attractive.

The Importance of Generalizing

Rule 2 is an example of a theory that slightly **overfits** the training data. It gets too specific and misses a crucial **generalization** about the number of colors that would have allowed it to get 100% on both the training and the testing set.

Rule 3 below shows an even worse case of **overfitting**.

Rule 3: More Overfitting

```
IF one purple shape OR one red shape OR two oval or diamond
shapes OR three shapes that are all green or all red THEN IN
ELSE OUT
```

Rule 3 also scores 100% on the training data, but it only scores 60% on the testing data, which is better than a coin flip, but not by much.

Finally, consider an extreme example of overfitting, in which we just make a list of all the IN examples and categorize something as IN if it matches one of them and OUT if it doesn't. In other words, do not attempt to generalize at all about the IN label. This approach leads to the following rule, which also scores 100% on the training data:

Rule 4: Extreme Overfitting

```
IF matches example 2, 3, 4, 6, 7, 9, 12, 14 or 15 THEN IN
ELSE OUT
```

Nothing in the testing set is an exact match for anything in the training set, so this rule categorizes everything as OUT and gets less than 50% correct (i.e. worse than a coin flip).

In real-world machine learning tasks, it is rare to encounter a problem like this one, in which there is a "correct" answer that will always achieve 100% accuracy on both training and testing data. Because of this, it is often desirable to go with a more general theory that **underfits** the training data but has a greater power to generalize. Consider the following rule:

Rule 5: Underfitting the Training Data

```
IF number of colors == 1 THEN IN
OTHERWISE OUT
```

This rule only scores 87% correct on the training data, which is worse than rules 2 through 4, but it scores 93% correct on the testing data, which is much better than the other rules. This ability to generalize to unseen examples makes it a much better rule than the others, despite the poorer performance on the training data.

Quick Question

Take a look at rules 1 through 5. Do you see any relationship between the rules and overfitting?

The Importance of Representation

The data provided to a machine learning algorithm consists of a list of **features** or attributes with different **values** given for each example.

It is often not at all obvious what would make for a good set of features. When learning to identify a person's face, should each pixel of an image be a feature, or should we do some kind of image processing to identify edges or other features of the image first? When learning to identify spam emails, how do we encode words and sentence structures?

Representation 1: Bitmaps

One possible way to represent the Sets of Symbols data would be to provide a list of pixel color values for, say, a 150 by 60 pixel image for each example, as shown on the right.

This image has 9000 pixels.

This would amount to 9000 different feature values for each image, which can overwhelm some algorithms, and would make the problem very difficult to solve. For example, what could it mean for the task at hand that the pixel at location (20, 40) has RGB values of 240, 83, and 74 respectively while the pixel at location (34, 25) has RGB values of 248, 201, and 198?

This is not to say that a learning algorithm couldn't find a good theory with this data, but it would be much easier if it knew higher level facts about the image

Representation 2: Logical Features

Another way to represent the data would be to use more meaningful features that encode prior knowledge of what's important, such as the shape, color, and shading of each symbol. This leads to 9 features for each example, which is much easier for the computer to deal with. If you still wanted to use bitmaps, you could add image processing algorithms to extract the 9 features from each image first, then pass these features to the learning algorithm.

Here's an example of what the some of the images in the training set would look like to the computer using this representational scheme:

ID	Shape1	Color1	Shade1	Shape2	Color2	Shade2	Shape3	Color3	Shade3
1	Diamond	Green	Filled	Diamond	Purple	Filled	Null	Null	Null
6	Oval	Red	Empty	Null	Null	Null	Null	Null	Null
8	Diamond	Purple	Filled	Oval	Green	Half	Curve	Red	Empty

This would make the learning problem much more tractable, but it would still be difficult for the computer to form the generalizations in the target rules.

Representation 3: Higher Order Features

To do a good job on this problem, higher order features such as "number of ovals", "number of diamonds", "number of colors", and so on would be very helpful. These features could be added to the set shown above, and could be used by the algorithm to quickly find a good theory.

Most of the time, we don't know in advance what's going to turn out to be important (that's why we're using Machine Learning in the first place!) So it may be a good idea, at least at first, to give the learning algorithm as many features as you can think of, and then analyze the model it comes up with to determine which features turned out to be most important.