Artificial Intelligence and Machine Learning

Error Based Learning

Big Idea

- A parameterized prediction model is initialized with a set of random parameters and an error function is used to judge how well this initial model performs when making predictions for instances in a training dataset.
- Based on the value of the error function, the parameters are iteratively adjusted to create a more and more accurate model.

Simple Linear Regression

- When there appears to have a linear relationship between a descriptive feature and target feature
- The equation of a line can be written as:

$$y = mx + b$$

- Find the best <m, b> to minimize the error.
- Measuring Error:

$$Error = \sum_{i=1}^{n} (y_i - (m \times x_i + b))^2$$

Multivariable Linear Regression

- Model equation:

$$y = w[0] + \sum_{i=1}^{M} w[i] \times x[i]$$

- Error measurement equation:

$$Error = \sum_{i=1}^{N} (y_i - \left(w[0] + \sum_{i=1}^{M} w[i] \times x[i] \right))^2$$

Minimize the Error

- To find the best weight W that minimizes error, the equations are:

$$\frac{\partial}{\partial w[i]} Error = 0, for\ i = 0\ to\ k$$

- Several ways to find the solution
  - Solve the equations
  - Guided search approach known as gradient descent algorithm

Gradient Descent Algorithm

Require: set of training instances D
Require: a learning rate α that controls how quickly the algorithm converges
Require: a function, **errorDelta**, that determines the direction in which to adjust a given weight, **w**[j], so as to move down the slope of an error surface determined by the dataset, D
Require: a convergence criterion that indicates that the algorithm has completed

```
1: w = random starting point in the weight space
2: repeat
3:     for each w[j] in w do
4:         w[j] = w[j] + α × errorDelta(D, w[j])
5:     end for
6: until convergence occurs
```

Error Delta

- Each weight is considered independent and for each one a small adjustment is made by adding a small delta value to the current weight, w[j].
- This adjustment should ensure that the change in the weight leads to a move downwards on the error surface.
- Equation:

$$errorDelta(D, w[j]) = \sum_{i=1}^{N}((y_i - \sum_{k=0}^{M}(w[k] \times x_i[k])) \times x_i[j])$$

Learning Rate

- The learning rate, α, determines the size of the adjustment made to each weight at each step in the process.
- Unfortunately, choosing learning rates is not a well defined science.
- Most practitioners use rules of thumb and trial and error to determine the learning rate.
- A typical range for learning rates is [0.00001, 10].
- Based on empirical evidence, choosing random initial weights uniformly from the range [-0.2, 0.2] tends to work well.

Interpreting Multivariable Linear Regression Models

- The weights used by linear regression models indicate the effect of each descriptive feature on the predictions returned by the model.
- Both the sign and the magnitude of the weight provide information on how the descriptive feature affects the predictions of the model.
- It is tempting to infer the relative importance of the different descriptive features in the model from the magnitude of the weights.
- However, direct comparison of the weights tells us little about their relative importance.
- A better way to determine the importance of each descriptive feature in the model is to perform a statistical significance test.

T-test

- T-test is a statistical significance test that can be used to analyze the importance of a descriptive feature x[j] in a linear regression model.
- The standard error for the overall model is calculated as

$$se = \sqrt{\frac{\sum_{i=1}^{N}(y_i - W \cdot x_i)^2}{n-2}}$$

- A standard error calculation is then done for a descriptive feature as follows:

$$se(x[j]) = \frac{se}{\sqrt{\sum_{i=1}^{N}(x_i[j] - \overline{x[j]})^2}}$$

- The t-statistic for this test is calculated as:

$$t = \frac{w[j]}{se(x[j])}$$

T-test (II)

- Using a standard t-statistic look-up table, determine the p-value associated with this test (this is a two tailed t-test with degrees of freedom set to the number of instances in the training set minus 2).
- If the p-value is less than the required significance level, typically 0.05, then the descriptive feature has a significant impact on the model; otherwise it does not.

Learning Rate Decay

- Learning rate decay allows the learning rate to start at a large value and then decay over time according to a predefined schedule.
- A good approach is to use the following decay schedule:

$$\alpha_\tau = \alpha_0 \frac{c}{c + \tau}$$

Handling Categorical Descriptive Features

- The basic structure of the multivariable linear regression model allows for only continuous descriptive features.
- The most common approach of handling categorical features uses a transformation that converts a single categorical descriptive feature into number of continuous descriptive feature values that can encode the levels of the categorical feature.

Handling Categorical Target Features

- Find a linear separator as the decision boundary.
- All the data points above the decision boundary will result in a negative value when plugged into the decision boundary equation, while all data points below the decision boundary will result in a positive value.
- Model equation:

$$y = \begin{cases} 1 \; if \sum_{i=0}^{M} w[i] \times x[i] \geq 0 \\ 0 \quad otherwise \end{cases}$$

- The surface defined by this rule is known as a decision surface.

Issues

- The hard decision boundary given in the previous slide is discontinuous, so is not differentiable and we can't calculate the gradient of the error surface.
- Furthermore, the model always makes completely confident predictions of 0 or 1, whereas a little more subtlety is desirable.
- Solution: use a more sophisticated threshold function that is continuous, and therefore differentiable, and that allows for the subtlety desired --- the logistic function

Logistic Function

- Equation:
  Logistic(z) = $\frac{1}{1+e^{-z}}$
  where z is a numeric value and e is the Euler's number, 2.718281828…
- Before training a logistic regression model, the binary target feature is mapped to 0 or 1.
- Logistic regression model equation:

$$y = Logistic(\sum_{i=0}^{M} w[i] \times x[i])$$

Training Logistic Model

- To repurpose the gradient descent algorithm for training logistic regression models, the only change that needs to be made is in the weight update rule.
- The new error delta rule is:

$$errorDelta(D, w[j]) = \sum_{i=1}^{N} \left(\left(y_i - M_w(X_i) \right) \times M_w(X_i) \times \left(1 - M_w(X_i) \right) \times X_i[j] \right)$$

where $M_w(X_i) = \sum_{k=0}^{M} (w[k] \times X_i[k])$

- For logistic regression models, it is recommended that descriptive feature values always be normalized to the range [-1, 1].

Modelling Non-linear Relationships

- In order to handle non-linear relationships, we transform the data rather than the model using a set of **basis functions**.
- The advantage of this is that, except for introducing the mechanism of basis functions, we do not need to make any other changes to the approach we have presented so far.
- For example, for each attribute A, we introduce a set of basis functions as:

$$\varphi_0(A) = 1$$
$$\varphi_1(A) = A$$
$$\varphi_2(A) = A^2$$

Basis Functions

- Typically, the number of basis functions in ø is larger than the number of descriptive features, so the application of the basis functions moves the data into a higher dimensional space.
- The expectation is that a linear separating hyperplane will exist in this higher dimensional space even though it does not in the original feature space.

Multinomial Logistic Regression

- For r target feature levels, we can build r separate logistic regression models, where each model is a one-versus-all logistic regression model.
- Then the outputs from the r models are combined.
- The r one-versus-all models can be trained in parallel.

Support Vector Machines

- Goal of training a support vector machine (SVM): find the decision boundary, or separating hyperplane, that leads to the maximum margin.
- The instances in a training dataset that fall along the margin extents, and so define the margins, are known as the support vectors and define the decision boundary.
- Training a support vector machine is framed as a constrained quadratic optimization problem.
- This type of problem is defined in terms of
  - A set of constraints
  - An optimization criterion

SVM

- The constraints required by the training process is:

$$y_i \times (w_0 + \boldsymbol{w} \cdot X_i) \geq 1$$

- The optimization criterion used is defined in terms of the perpendicular distance from any instance to the decision boundary and is given by

$$dist(X_i) = \frac{abs(w_0 + \boldsymbol{w} \cdot X_i)}{\|\boldsymbol{w}\|}$$

where $\|\boldsymbol{w}\| = \sqrt{\sum_{i=1}^{M} \boldsymbol{w}_i{}^2}$, and is known as the Euclidean norm of **w**.

- For instances along the margin extents, $abs(w_0 + \boldsymbol{w} \cdot X_i) = 1$.
- So the distance from any instance along the margin extents to the decision boundary is $\frac{1}{\|\boldsymbol{w}\|}$, and because the margin is symmetrical to either side of the decision boundary, the size of the margin is $\frac{2}{\|\boldsymbol{w}\|}$.

Training SVM

- The goal when training a support vector machine is:
  - Minimize $\|\boldsymbol{w}\|$
  - Subject to the constraint

$$y_i \times (w_0 + \boldsymbol{w} \cdot X_i) \geq 1 \; for \; all \; i$$

- Basis functions can be used with support vector machines to handle data that is not linearly separable.
- The constraint becomes:

$$y_i \times (w_0 + \boldsymbol{w} \cdot \varphi(X_i)) \geq 1 \; for \; all \; i$$

Kernel Trick

- The dot product with basis functions is a computationally very expensive operation.
- To avoid it, kernel trick is usually used.
- Applying a much less costly kernel function to the original descriptive feature values.

$$\boldsymbol{w} \cdot \varphi(X_i) \approx kernel(\boldsymbol{w}, X_i)$$

- Some popular options for kernel functions:
  - Linear kernel -- $kernel(\boldsymbol{w}, X_i) = \boldsymbol{w} \cdot X_i + c$
  - Polynomial kernel -- $kernel(\boldsymbol{w}, X_i) = (\boldsymbol{w} \cdot X_i + 1)^p$
  - Gaussian radial basis kernel --

$$kernel(\boldsymbol{w}, X_i) = \exp(-\gamma \|\boldsymbol{w} - X_i\|^2)$$