INT354
Machine Learning

Normalization and Feature Scaling

Feature Scaling

- **Used for standardization of independent variables of data features.**
- **Dataset contains features varying in magnitude, units and range. For example:**
  - **Gold_weight measured in gms.**
  - **Iron_weight measured in Kg.**
- **Euclidian distance is not the best method to scale the features.**

Techniques of Feature Scaling

- **Standardisation**
- **Normalization**

Standardisation

$$x' = \frac{x - mean(x)}{\sigma}$$

- **This redistributes the features with their mean =0 and standard deviation =1.**

Normalisation

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

Exercise

Consider the following dataset:

X
0.0
1.0
2.0
3.0
4.0
5.0

Perform standardisation and normalisation on dataset.

Solution

Consider the following dataset:

X	Normalized	Standardized
0.0	0.0	-1.336306
1.0	0.2	-0.801784
2.0	0.4	-0.267261
3.0	0.6	0.267261
4.0	0.8	0.801784
5.0	1.0	1.336306

Over-fitting

- **Model performs much better on a training dataset than on the test dataset.**
- **Model fits the parameter too closely to a particular observation in the training dataset.**
- **Not generalize the real data.**

Reduce Generalization Errors

- **Collect more training data.**
- **Introduce a penalty for complexity via regularization.**
- **Choose a simpler model with fewer parameters.**
- **Reduce the dimensionality of the data.**

Sparse Solution With L1 Regularization

$$\boldsymbol{L1}: \left|\left|\boldsymbol{w}\right|\right|_{1} = \sum_{j=1}^{m} |\boldsymbol{wj}|$$

- **L1 regularization yields sparse feature vectors.**
- **Sparsity is useful if dataset is high dimensional with many irrelevant features.**
- **L1 penalty is the sum of the absolute weight coefficients.**

Sparse Solution With L2 Regularization

$$L2: ||w||_1^2 = \sum_{j=1}^{m} |wj|^2$$

- **L2 penalty is the sum of the square of weights.**

Sequential Feature Selection Algorithms

- **Family of greedy search algorithms.**
- **Reduce an initial d-dimensional feature space into k-dimensional feature sub-space where k<d.**
- **Automatically select a subset of features that are most relevant to the problem.**

Sequential Forward Selection (SFS) Algo.

SFS is the simplest greedy search algorithm.

- **Starting from the empty set, sequentially add the features x+ that maximizes $J(Y_k+x^+)$ when combined with the features Y_k that have already been selected.**
  1. **Start with the empty set $Y_0=\{\emptyset\}$**
  2. **Select the next best feature $x^+=\text{argmax}J(Y_k+x)$**
  3. **Update $Y_k+1=Y_k+x^+$; k=k+1**
  4. **Go to 2**

Sequential Forward Selection (SFS) Algo.

- **SFS performs best when the optimal subset is small.**
- **The search space is drawn like an ellipse to emphasize the fact that there are fewer states towards the full or empty sets.**

Example

- **Run SFS to completion for the following objective function:**

$$J(X)=-2x_1x_2+3x_1+5x_2-2x_1x_2x_3+7x_3+4x_4+-2x_1x_2x_3x_4$$

Where x_k are indicator variables, which indicate whether the k^{th} feature has been selected (x_k=1) or not (x_k=0)

	J(x1)=3	**J(x2)=5**	**J(x3)=7**	**J(x4)=4**
x3 is maximum:	**J(x3x1)=10**	**J(x3x2)=12**		**J(x3x4)=11**
x3x2 is maximum:	**j(x3x2x1)=11**		**j(x3x2x4)=16**	
x3x2x4 is maximum:	**j(x3x2x4x1)=13**			

Sequential Backward Selection (SBS) Algo.

Aims to reduce the dimensionality of the initial feature subspace.

- **Initialize the algorithm with k=d where d is the dimensionality of the full feature space X_d.**
- **Determine the feature x^- that maximizes the criterion x^- =argmaxJ(X_k-x) where x∈X_k.**
- **Remove the feature x^- from the feature set: X_k-1=X_k-x^-, k=k-1.**
- **Terminate if k equals the number of desired features, if not, go to step 2.**

Sequential Backward Selection (SBS)

- SBS works best when the optimal feature subset is large, since SBS spends most of its time visiting large subsets.
- The main limitation of SBS is its inability to re-evaluate the usefulness of a feature after it has been discarded.

Bidirectional Search (BDS)

BDS is a parallel implementation of SFS and SBS.

- **SFS is performed from the empty set.**
- **SBS is performed from the full set.**
- **To guarantee that SFS and SBS converge to the same solution.**
  - **Features already selected by SFS are not removed by SBS.**
  - **Features already removed by SBS are not selected by SFS.**

Bidirectional Search (BDS)

1. **Start SFS with YF={∅}**
2. **Start SBS with YB=X**
3. **Select the best feature**

$$x^+ = \underset{\substack{x \notin Y_{F_k} \\ x \in F_{B_k}}}{\arg\max}\, J\left(Y_{F_k} + x\right)$$
$$Y_{F_{k+1}} = Y_{F_k} + x^+$$

4. **Remove the worst feature**

$$x^- = \underset{\substack{x \in Y_{B_k} \\ x \notin Y_{F_{k+1}}}}{\arg\max}\, J\left(Y_{B_k} - x\right)$$
$$Y_{B_{k+1}} = Y_{B_k} - x^-;\ k = k + 1$$

5. **Go to step 2**

Selecting Features Using Random Forests

There are two different methods for feature selection are:

- **Mean decrease impurity**
- **Mean decrease accuracy**

Mean Decrease Impurity

- **Impurity: measure based on which optimal condition is chosen.**
- **During training, it is computed how each feature decreases the weighted impurity in a tree.**
- **For a forest, the impurity decrease from each feature can be averaged and the features are ranked according to this measure.**

Mean Decrease Impurity

- **Feature selection based on impurity reduction is biased towards preferring variables with more categories.**
- **When the dataset has two or more correlated features, any of these correlated features can be used as the predictor.**

Mean Decrease Accuracy

- **Measure the impact of each feature on accuracy of the model.**
- **Permute the values of each feature and measure how much the permutation decreases the accuracy of the model.**
- **Unimportant variables permutation have little or no effect on model accuracy.**
- **Important variables permutation significantly decrease the accuracy.**


COMING UP