A List of Dataset and Feature Descriptors (Properties) - extracted from Expose2 Ontology

By: Noureddin Sadawi

31 Jul 2014

1) **dataset_property**

   A) **qualitative_dataset_property**

      I) **data_cleanliness**

         1) contains_missing_values
         2) missing_value_free

      II) **supervision**

         1) semi-supervised
         2) supervised
         3) unsupervised

   B) **quantitative_dataset_property**

      I) **concept-based_dataset_property**

         1) concept_variation: the (non-) uniformity of the class-label distribution throughout the feature space (measured through the distance between two examples of a different class)
         2) example_cohesiveness: the density of the example distribution in the training set

      II) **information_theoretic_dataset_property**

         1) average_mutual_information
         2) coefficient_of_variation_of_target: The coefficient of variation of the target is defined as the ratio of the standard deviation to the mean of the target attribute and can be used instead of entropy on numerical targets. It is a normalization of the standard deviation of the target useful for numerical targets X_{target}. A related measure, *sparsity of the target*, is $VarCoef_{target}$ discretized into 3 values.

$$VarCoef_{target} = \frac{\sigma_X}{\mu_X} \quad (1)$$

3) equivalent_number_of_attributes: The equivalent number of attributes is a quick estimate of the number of attributes required, on average, to describe the class.

$$EN - atrr = \frac{H(C)}{\overline{MI(C,X)}} \tag{2}$$

4) median_of_uncertainty_coefficients
5) noise_to_signal_ratio: The noise to signal ratio is an estimate of the amount of non-useful information in the attributes regarding the class. $\overline{H(X)}$ is the average information (useful or not) of the attributes.

$$NS - ratio = \frac{\overline{H(X)} - \overline{MI(C,X)}}{\overline{MI(C,X)}} \tag{3}$$

6) target_feature_entropy

III) **instance-based_property:** Compares entire instances with each other. A dataset may contain two observations with similar or equal attribute values, but with different labels which might cause a classifier to get confused. Analogously, there might be two or more observations which are identical, which (unfairly?) gives them more weight in some algorithms. Here, properties derived from case-based learning are used originally intended to assess the quality of a given case-base.

1) instance_consistency: A single example is consistent within the dataset if and only if there does not exist any other example that is identical, but has a different target value
2) instance_incoherence: Incoherence is a measure for how dissimilar the examples are in their attribute space. An example is called incoherent within a dataset if it does not overlap with any other example in a predefined number of attributes.
3) instance_minimality: An example is subsumed by another example if its attributes form a true subset of another example with the same label. It is useful mostly for relational representations. An example that is not subsumed by another example is minimal.
4) instance_similarity: The overall similarity of examples in a dataset is defined as the normalized weighted sum of four different local similarity measures
5) instance_uniqueness: An example is unique if and only if there does not exist another identical example.

IV) **landmarker:** A different way of probing the structure of the concepts hidden in the data is running some algorithms on it with very different biases, and see how well they perform: the better they do, the closer their bias fits the data. This is what is done naturally when we manually seek an appropriate algorithm: we first select a wide range of very different algorithms, do some preliminary evaluations, and then we remove all algorithms that dont seem to perform well, leaving us with a small set of candidates to evaluate in detail.

1) 1-nearest_neighbor_landmarker: Define a distance on the instance space, e.g. the euclidean distance, and label new observation with the observation of the closest training example. In classification problems, the goal of this landmark learner is to determine how close instances belonging to the same class are.
2) decision_stump_landmarker: Using a decision tree learner, C5.0 to be precise, a single decision node is constructed (representing a single split of the data) which is then to be used for classifying test observations. The goal of this landmark learner is to establish closeness to linear separability.
3) elite_1-nearest_neighbor_landmarker: A 1-Nearest Neighbor is used again, but only on a subset of attributes, i.e. the most informative attributes according to the information gain ratio. It intents to establish whether a task involves parity-like relationships between its attributes, which means that no single attribute is considerably more informative than another.
4) linear_discriminant_landmarker: A single linear target function is computed splitting the instance space in two. Like decision stumps, it also establish closeness to linear separability, but not axis-parallel as is the case in the former.
5) naive_Bayes_landmarker: A simple learning algorithm using Bayes theorem to calculate the possibility that an observation belongs to a certain class. Since it assumes that the attributes are conditionally independent from each other, this landmarker is used to measure the extent to which the attributes actually are independent given the class.
6) random_tree_landmarker: Also using decision trees, an attribute is chosen randomly at each node until the entire tree is built. The goal of this landmark learner is to inform about irrelevant attributes.
7) worst_node_landmarker: By using the decision trees information

gain ratio again, the least informative attribute is used to make the single split. Together with the first landmark learner, this landmarker is supposed to inform about linear separability.

V) **model-based_property:** model-based characterization, builds a model that is typically very fast to induce, and characterizes the data based on properties of that model without doing a full-fletched evaluation of a wide range of learning algorithms.
   1) **decision_tree-based_property:**
      a) distribution_of_branch_lengths
      b) distribution_of_feature_occurrence_in_nodes_
      c) distribution_of_number_of_nodes_per_level
      d) number_of_leafs
      e) number_of_nodes
      f) tree_depth
      g) tree_width

VI) **simple_dataset_property:**
   1) dimensionality_of_the_data: ratio of number_of_attributes and number_of_instances
   2) number_of_binary_features
   3) number_of_features
   4) number_of_instances
   5) number_of_instances_with_missing_values
   6) number_of_missing_values
   7) number_of_nominal_features
   8) number_of_numerical_features
   9) number_of_target_classes
   10) percentage_of_features_with_outliers
   11) percentage_of_missing_values
   12) percentage_of_nominal_features
   13) percentage_of_numerical_features
   14) presence_of_outliers_in_target

VII) **statistical_dataset_property**
   1) average_correlation_to_target: Measures the correlation between a numerical attribute X and a numerical target Y_{target}.
   2) average_feature_kurtosis
   3) average_feature_skewness
   4) average_p-value_for_target: measures the correlation between a nominal attribute X and a numerical target Y_{target}.

5) Box's_M_statistic: Box's M-statistic measures the equality of the covariance matrices S_i of the different classes. If they are equal, then linear discriminants could be used, otherwise, quadratic discriminant functions should be used instead. As such, M predicts whether a linear discriminant algorithm should be used or not. In the following, $S_i = \frac{S_{c_i}}{n_i - 1}$ is the i class covariance matrix with S_{c_i} the i class scatter matrix and n_i the number of examples pertaining to class i, and $S = \frac{1}{n-cl} \sum_i S_{c_i}$ the pooled covariance matrix. It is zero when all individual covariance matrices are equal to the pooled covariance matrix.

$$M = \gamma \sum_i (n_i - 1) log \frac{|S|}{|S_i|} \quad (4)$$

$$\gamma = 1 - \frac{2num^2 + 3num - 1}{6(num + 1)(cl - 1)} (\sum_i \frac{1}{n_i - 1} - \frac{1}{n - cl}) \quad (5)$$

6) first_canonical_correlation: the first canonical correlation coefficient measures the association between all numerical attributes and a nominal (class) attribute. In principal component analysis (PCA), datasets are transformed into a new dataset with fewer dimensions (attributes). The first dimension, called the first principal component is a new axis in the direction of maximum variance. The variance of this principal axis is given by the largest eigenvalue λ_1. It thus measures how well the classes can be separated by the numerical attributes.

$$\rho_{max} = \sqrt{\frac{\lambda_1}{1 + \lambda_1}} \quad (6)$$

7) frac1: the fraction of the total variance retained in the 1-dimensional space defined by the first principal component can be computed as the ratio between the largest eigenvalue λ_1 of the covariance matrix S and the sum of all its eigenvalues:

$$frac1 = \frac{\lambda_1}{\sum_i \lambda_i} \quad (7)$$

8) SD-ratio: SD-ratio, the standard deviation ratio, is a reexpression of Box's M-statistic M which is one if M is zero and strictly greater than one if the covariances differ.

$$SD\text{-}ratio = exp(\frac{M}{num \sum_i (n_i - 1)}) \quad (8)$$

9) stationarity_of_target: Indicates whether the standard deviation of the target feature is larger that its mean

10) target_feature_kurtosis
11) target_feature_skewness

VIII) **sub-sampling_landmarker:** Runs the learning algorithms on a small sample of the data, which for most algorithms will result in much faster training times. Indeed, learning algorithms typically learn the most from the first few examples. More examples will further fine-tune the model, but the performance gains will be small in comparison with the first few examples. When plotting the performance of learning algorithms on increasingly larger samples of a dataset, the resulting curve is called a learning curve. It will typically shoot up after a small percentage of the data and will, depending on the learning algorithm, start to level off shortly after that. The assumption of subsampling landmarking is that the performance of one point in the beginning of the learning curve will help us to predict the performance on the entire dataset.

1) partial_learning_curve
2) single_subsample

IX) **task-specific_dataset_property:**

1) clustering_dataset_property
2) **time_series_analysis_dataset_property:**
   a) coefficient_of_variation
   b) mean_of_first_5_autocorrelations
   c) statistical_significance_of_autocorrelations

2) **feature_property:**

A) **qualitative_feature_property:**

I) **feature_datatype:**

1) nominal_datatype
2) **numerical_datatype:**
   a) boolean_datatype
   b) integer_datatype
   c) **real_datatype:**
      i) real_from_0_to_1_datatype

B) **quantitative_feature_property:**

I) feature_entropy: the entropy of a nominal attribute X is a measure of the uncertainty (or randomness) associated with it. It measures the average information content one is missing when one does not know the exact value of X. If entropy is zero (if all values are the

same), the attribute contains no information. The class entropy H(C) is the amount of information required to specify the class of an instance, a measure for how informative the attributes need to be. A low H(C) means that the distribution of examples among classes is very skewed (containing some very infrequent classes) which some algorithms cannot handle well.

$$H(X)_{norm} = \frac{H(X)}{log_2 n} \tag{9}$$

Although a definition exists for numerical distributions (using an integral instead of a summation), it is of no use for empirical data, and the entropy of numerical attributes (or targets) is calculated by discretizing the values in equal-length intervals.

II) feature_kurtosis: β, the kurtosis or fatness of the distribution's tail, or the 4th standardized moment

$$\beta = \frac{E(X - \mu_X)^4}{\sigma_X^4} \tag{10}$$

III) **feature_redundancy_property:** Determines the degree of redundancy in a dataset: if two or more attributes are dependent, they don't add much information and only increase the dimensionality of the dataset. This is measured by estimating the strength of the relationship between attributes.

1) feature_concentration_coefficient: τ_{XY}, the *concentration coefficient* measuring the association between two nominal attributes, or the proportional reduction in the probability of an incorrect guess predicting Y, with J distinct values, using X, with I distinct values

$$\tau_{XY} = \frac{\sum_i \sum_j \frac{\pi_{ij}^2}{\pi_{i+}} - \sum_j \pi_{+j}^2}{1 - \sum_j \pi_{+j}^2} \tag{11}$$

2) feature_correlation_coefficient: ρ_{XY}, the correlation coefficient measuring the association between two numerical attributes

$$\rho = \frac{\sigma_{XY}}{\sqrt{\sigma_X^2 \sigma_Y^2}} \tag{12}$$

3) multiple_correlation_coefficient: R_i, the multiple correlation coefficient measuring the maximal correlation coefficient between a numerical attribute X_i and some linear combination of all other numerical attributes $Z_i\alpha$, with $Z_i = (X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_{num})$ and α a non-zero vector.

$$R_i = argmax_{\alpha \neq 0} \frac{\sigma_{X_i Z_i \alpha}}{\sqrt{\sigma_{X_i}^2 \sigma_{Z_i \alpha}^2}} \tag{13}$$

4) mutual_information: $MI(Y,X)$, the *mutual information* between nominal attributes X and Y describes the reduction in uncertainty of Y due to the knowledge of X, and leans on the conditional entropy $H(Y|X)$. It is also the underlying measure of the information gain metric used in decision tree learners.

$$MI(Y,X) = H(Y) - H(Y|X) \quad (14)$$

$$H(Y|X) = \sum_i p(X=x_i)H(Y|X=x_i) \quad (15)$$

$$= -\sum_i \pi_{i+} \sum_j \pi_{j|i} log_2(\pi_{j|i}) \quad (16)$$

5) p-value_of_F_distribution: $p_{val_{XY}}$, the p-value of the F-distribution for a nominal attribute X with I values and a numeric attribute Y. The Analysis of Variance (ANOVA) examines how a numerical variable affects a nominal one by examining whether the means of the I groups defined on Y by X are different. The ratio of the *between group variance* and the *within group variance* $MS(B)/MS(W)$ follows the F-distribution and the p-value of that distribution gives the probability of observing that ratio under the assumption that the group means are equal. A p-value close to one means we can accept that assumption, an indication that X heavily affects Y

6) uncertainty_coefficient: $UC(X,Y)$, the uncertainty coefficient is the mutual information between an attribute X and target attribute Y divided by the entropy of Y. It measures the proportional reduction in the *variance* of Y when X is known. It is strongly related to the *information gain ratio* used in decision trees, which is defined as $UC(Y,X)$, or the proportional reduction in in the variance of X when target Y is known.

$$UC(X,Y) = \frac{MI(Y,X)}{H(X)} \quad (17)$$

IV) feature_skewness: γ, the skewness or lack of symmetry in the distribution, or the 3rd standardized moment

$$\gamma = \frac{E(X-\mu_X)^3}{\sigma_X^3} \quad (18)$$

V) **nominal_feature_properties:**

1) number_of_feature_values

VI) normalized_feature_entropy: $H(X)_{norm}$, the normalized entropy of a nominal attribute X rescales entropy to the [0..1] interval

$$H(X)_{norm} = \frac{H(X)}{log_2 n} \quad (19)$$

VII) **numerical_feature_properties:**

1) maximum_feature_value
2) minimum_feature_value