Preliminary Project Documentation
UMA

1 Project: Random Forest with SVM

Task Description

"Combining a Random Forest with SVM in a classification task. Proceed as with the creation of a Random Forest, but a certain percentage of classifiers in the forest are SVMs. One of the classifiers (SVM or ID3 tree) can come from an existing implementation."

Design Decision

In this project, we will use an existing implementation of **SVM** (external library - `scikit-learn`), while the **ID3** decision tree algorithm will be implemented entirely from scratch.

The implementation includes: tree node structure, recursive set splitting function, calculation of Entropy, and Information Gain.

2 Goal and Assumptions

The goal of the project is to create a hybrid ensemble of classifiers (based on the Random Forest principle), where the base units are selected randomly:

- With probability p_{svm}, an SVM classifier (ready-made implementation) is created.
- With probability $1 - p_{svm}$, an ID3 tree (custom implementation) is created.

Training is performed according to the *Bagging* procedure:

1. For each of the T estimators, a bootstrap sample (with replacement) D_i is drawn.
2. The selected model is trained on the drawn sample.
3. The final decision is made via majority voting.

3 Author's Implementation: ID3 Algorithm

The ID3 algorithm builds a decision tree using a greedy method, selecting the attribute that best splits the training set regarding classification at each step. Information Gain is used as the splitting criterion.

3.1 Entropy

The measure of disorder of set S with respect to the decision class C is defined as:

$$H(S) = -\sum_{i=1}^{k} p_i \log_2(p_i)$$

where:

- k - number of decision classes,

- p_i - probability of the occurrence of the i-th class in set S (fraction of examples belonging to class i).

If $p_i = 0$, we assume $0 \cdot \log_2(0) = 0$.

3.2 Information Gain (IG)

Information Gain for splitting set S with respect to attribute A is calculated as the difference between the entropy of the set before the split and the weighted average entropy of the subsets resulting from the split:

$$IG(S, A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} H(S_v)$$

where:

- $Values(A)$ - set of unique values of attribute A,
- S_v - subset of examples from S for which attribute A takes the value v,
- $|S|$ - cardinality of set S.

3.3 Tree Building Procedure

1. If all examples in the node belong to one class, create a leaf with that class.
2. If the set of attributes is empty, create a leaf with the most frequent class (mode).
3. Otherwise:
   (a) For each available attribute, calculate $IG(S, A)$.
   (b) Select the attribute A_{best} with the highest gain.
   (c) Create a decision node testing A_{best}.
   (d) Split set S into subsets S_v according to the values of A_{best}.
   (e) Recursively call the procedure for each S_v (removing A_{best} from the list of available attributes).

3.4 Example Calculations (First Step of ID3 Algorithm)

Consider a training set S with two binary attributes A and B, and a decision class $Y \in \{\text{Yes}, \text{No}\}$:

Sample	A	B	Y
1	0	0	Yes
2	0	1	No
3	0	0	No
4	1	0	Yes
5	1	1	No
6	1	0	Yes
7	0	1	Yes
8	1	1	No
9	0	0	Yes
10	1	0	No

The set contains 5 "Yes" examples and 5 "No" examples, so:

$$p_{\text{Yes}} = \frac{5}{10} = 0.5, \qquad p_{\text{No}} = \frac{5}{10} = 0.5.$$

The entropy of the entire set (base-2 logarithms) is:

$$H(S) = -\sum_{c\in\{\text{Yes},\text{No}\}} p_c \log_2 p_c = -0.5\log_2(0.5) - 0.5\log_2(0.5) = 1.000.$$

Evaluation of attribute A.
Splitting by value A yields two subsets:

$$A=0: \ \{1,2,3,7,9\} \quad (|S_{A=0}|=5), \qquad A=1: \ \{4,5,6,8,10\} \quad (|S_{A=1}|=5).$$

In $S_{A=0}$, we have 3 "Yes" and 2 "No" examples:

$$p_{\text{Yes}|A=0} = \frac{3}{5} = 0.6, \qquad p_{\text{No}|A=0} = \frac{2}{5} = 0.4,$$

hence

$$H(S_{A=0}) = -0.6\log_2(0.6) - 0.4\log_2(0.4) \approx 0.971.$$

For $S_{A=1}$, we symmetrically have 2 "Yes" and 3 "No", so

$$H(S_{A=1}) \approx 0.971.$$

Weighted entropy after split:

$$\frac{5}{10}\cdot 0.971 + \frac{5}{10}\cdot 0.971 = 0.971.$$

Information Gain:

$$IG(S,A) = H(S) - 0.971 \approx 0.029.$$

Evaluation of attribute B.
Splitting by B:

$$B=0: \ \{1,3,4,6,9,10\} \quad (|S_{B=0}|=6), \qquad B=1: \ \{2,5,7,8\} \quad (|S_{B=1}|=4).$$

In $S_{B=0}$, we have 4 "Yes" and 2 "No", thus

$$p_{\text{Yes}|B=0} = \frac{4}{6} \approx 0.6667, \qquad p_{\text{No}|B=0} = \frac{2}{6} \approx 0.3333,$$

$$H(S_{B=0}) \approx -\tfrac{4}{6}\log_2\left(\tfrac{4}{6}\right) - \tfrac{2}{6}\log_2\left(\tfrac{2}{6}\right) \approx 0.918.$$

In $S_{B=1}$, we have 1 "Yes" and 3 "No", so

$$p_{\text{Yes}|B=1} = \frac{1}{4} = 0.25, \qquad p_{\text{No}|B=1} = 0.75,$$

$$H(S_{B=1}) = -0.25\log_2(0.25) - 0.75\log_2(0.75) \approx 0.811.$$

Weighted entropy after split:

$$\frac{6}{10}\cdot 0.918 + \frac{4}{10}\cdot 0.811 \approx 0.551 + 0.324 = 0.875.$$

Information Gain:

$$IG(S,B) = H(S) - 0.875 \approx 0.125.$$

Selection of attribute for splitting: since

$$IG(S,B) \approx 0.125 > IG(S,A) \approx 0.029,$$

according to the ID3 criterion, we select attribute B for the first split.

Further algorithm steps (recursive splits in subsets, stopping criteria, leaf creation) are performed analogously using the same procedure of calculating entropy and information gain.

4 SVM Algorithm

Support Vector Machine (SVM) is a linear classifier whose goal is to find a hyperplane that maximizes the margin between classes. The model has the form:

$$f(x) = \mathrm{sign}(w^T x + b).$$

In the project, we use a library implementation (`scikit-learn`) with a linear kernel. Training involves minimizing the function:

$$\frac{1}{2}\|w\|^2 + C \sum_i \xi_i,$$

where parameter C controls the trade-off between the margin and the number of errors.

For multi-class sets, the *One-vs-Rest* strategy is used. Each SVM classifier is trained on a bootstrap sample and participates in the majority voting within the forest.

5 Forest Algorithm (Ensemble)

1. Input: training set D, number of classifiers T, SVM share p_{svm}.
2. For $i = 1$ to T:
    - Draw a random number $r \in [0, 1]$.
    - Draw a bootstrap sample D_i.
    - If $r < p_{svm}$:
        - Create an SVM instance (library).
        - Train SVM on D_i.
    - Otherwise:
        - Create a new instance of the custom ID3 tree.
        - Train the tree on D_i (recursive split).
    - Add the trained model to list M.
3. Prediction for new x: each model from M returns a class, the final result is the class indicated by the majority.

6 Experimental Plan

6.1 Datasets

The experiments will use three datasets from the UCI Machine Learning Repository. They were selected to test the algorithm's performance on both naturally discrete and continuous data.

1. **Mushroom Data Set** (Discrete set)
    - **Number of examples:** 8124
    - **Number of features:** 22
    - **Decision classes:**
        - Edible: 4208
        - Poisonous: 3916
2. **Wisconsin Breast Cancer (Diagnostic)** (Continuous set)
    - **Number of examples:** 569
    - **Number of features:** 30

- **Decision classes:**
    - Benign: 357
    - Malignant: 212

3. **Wine Quality – Red** (Continuous set)
    - **Number of examples:** 1599
    - **Number of features:** 11
    - **Decision classes** (after binarization: quality ≥ 6 is the positive class):
        - Good (score ≥ 6): 855
        - Poor (score < 6): 744

Handling Continuous Attributes
The *Mushroom* set consists exclusively of discrete attributes, which are natively supported by the implemented ID3 algorithm. In the case of the *Breast Cancer* and *Wine Quality* sets, which contain continuous attributes, discretization will be applied before running the ID3 algorithm. Continuous attribute values will be divided into k bins of equal width or equal frequency, and then encoded as discrete values (e.g., 0, 1, ..., $k-1$).

6.2 Research Methodology

- Each experiment will be repeated 25 times.
- Results will be aggregated: mean, standard deviation, best and worst result.
- 5-fold cross-validation will be applied.

6.3 Quality Metrics

- Accuracy.
- Confusion Matrix.

6.4 Verification and Comparison with Reference Methods

In accordance with project requirements, the correctness of the custom ID3 implementation and the operation of the entire hybrid forest algorithm will be verified by comparing them with existing implementations in Python (`scikit-learn` library):

- **ID3 Verification:** Results obtained by the custom tree will be compared with the results of the `DecisionTreeClassifier` class (with the `criterion='entropy'` parameter). Any discrepancies, e.g., resulting from the applied method of discretizing continuous attributes, will be analyzed and discussed.
- **Comparison with Classic Approach:** The results obtained for the hybrid algorithm (for $p_{svm} > 0$) will be compared with the results of the classic Random Forest (`RandomForestClassifier`). This will allow assessing whether the modification of the standard algorithm (adding SVM) resulted in improved performance.

6.5 Research Scenarios

As part of the research into the properties of the developed algorithm, the impact of the following parameters will be examined:

1. **Impact of SVM share (p_{svm}):** Examining values from the range $\{0, 20, 50, 80, 100\}\%$. This comparison will determine how hybridization affects classification quality relative to a pure tree forest ($p_{svm} = 0$) and pure SVM bagging ($p_{svm} = 100$).

2. **Impact of the number of classifiers (T):** Examining values $T \in \{10, 20, 50, 100\}$ to assess the stabilization of the result as the number of estimators increases.

3. **Impact of SVM regularization parameter (C):** Checking the sensitivity of the hybrid model to the selection of parameter C for the component classifiers.