Ensemble, Voting and Bagging classifiers

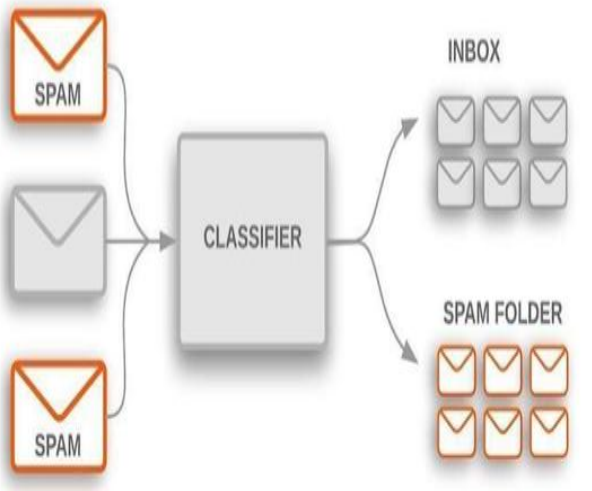
- Mohsen Shahabi
- Amir Mousavi
- Vida Zarei
- Professor: Dr. Yasser Alginahi

Agenda

1. The Main Concepts of Machine Learning
2. Ensemble Learning
3. Pros and Cons of Ensemble Learning
4. Voting Classification
5. Hard Voting and Soft Voting
6. Voting Example with Python
7. Bagging Classification
8. Bagging Example with Python
9. Conclusion

The Main Concepts

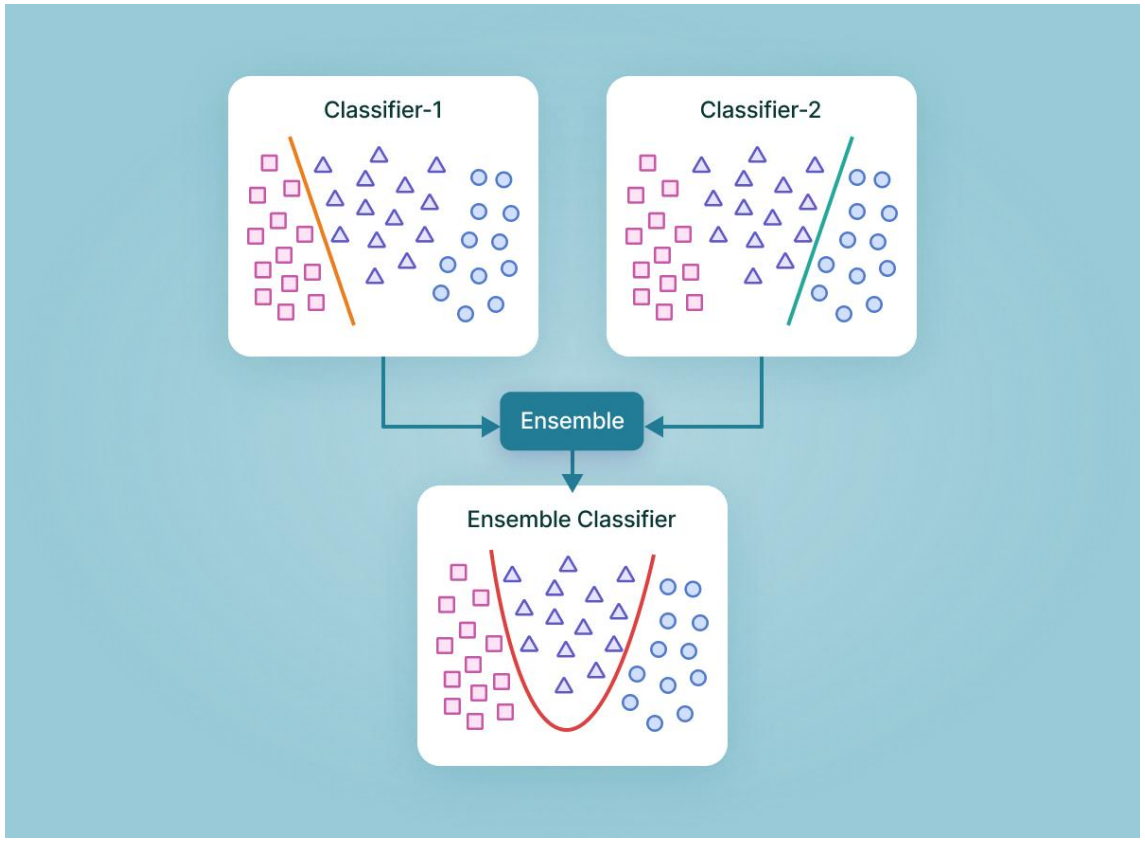
- Machine learning: is a branch of artificial intelligence that teaches computers to learn from data and make predictions or decisions without explicit programming.
- Supervised Learning: is a type of ML where a computer learns by example, using labeled data with known outcomes to make predictions or classifications.
- Classification: Classification is a fundamental technique in supervised machine learning where the goal is to categorize data points into predefined classes or categories.
- Categorizing Data: Classification is used to assign a class or label to each data point based on its features or attributes.



Classification representation
[source: https://tinyurl.com/363ft6tn]

What is Ensemble Learning?

Ensemble Learning is a machine learning technique where multiple individual models are combined to make more accurate predictions or decisions than any single model could achieve on its own. It's like asking a group of experts for advice to make better choices.



Representation of Ensemble
[source: https://tinyurl.com/yzhw6umn]

Classification and Ensemble methods

The Relationship between Classification and Ensemble methods like voting and bagging classifiers is about categorizing data but ensemble methods improve classification accuracy.

Ensemble Learning - Pros

1. **Improved Accuracy:** Ensemble learning combines multiple models to make more accurate predictions by taking advantage of their collective wisdom.

2. **Reduces Overfitting:** Ensemble methods help prevent a single model from memorizing the training data and making errors on new data by averaging or combining predictions.

3. **Handles Imbalanced Data:** Ensemble techniques can give fair consideration to all classes in imbalanced data, ensuring better predictions, especially for underrepresented classes.

Ensemble Learning- Cons

1. Complexity: Ensemble methods can introduce added complexity to the model, making it harder to understand and maintain. It's like having many cooks in the kitchen, which can lead to confusion.

2. Risk of Bias: Depending on the data and models used, ensemble learning may amplify biases present in the individual models. It's like if a group of people all have the same incorrect belief, their combined decision could still be biased.

3. Data Requirements: Some ensemble methods require a larger amount of data to be effective, which may not always be available. It's like needing a bigger sample size for more reliable survey results; sometimes, you just don't have enough data.

7

Voting Classification

- Voting in machine learning is like asking a group of experts for their opinion, and then going with what most of them say to make a decision.
- It's typically used for classification tasks, and it combines different types of classifiers.

Bagging (Bootstrap Aggregation)

Bagging (Bootstrap Aggregation) is a method in machine learning that creates multiple copies of a model, trains each on different subsets of the data, and combines their predictions for better accuracy and reliability.

The difference between Voting and Bagging

The main difference between voting classification and bagging (Bootstrap Aggregation) lies in how they combine the predictions of multiple classifiers: The key difference is that voting classification combines predictions from different types of classifiers using a voting mechanism, while bagging uses multiple instances of the same classifier on different data subsets and aggregates their predictions to reduce variance and improve accuracy.

Voting vs. Bagging

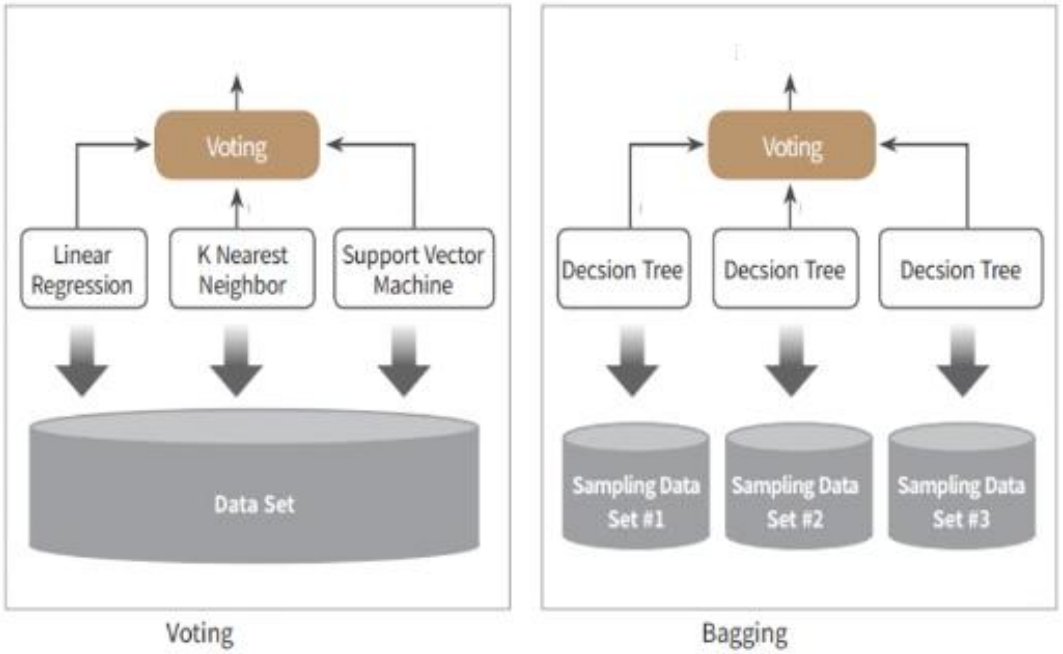
Voting

- Generally, combine different classifiers
- Classifiers use identical dataset

Bagging

- Generally, use identical classifiers
- Classifiers use different datasets

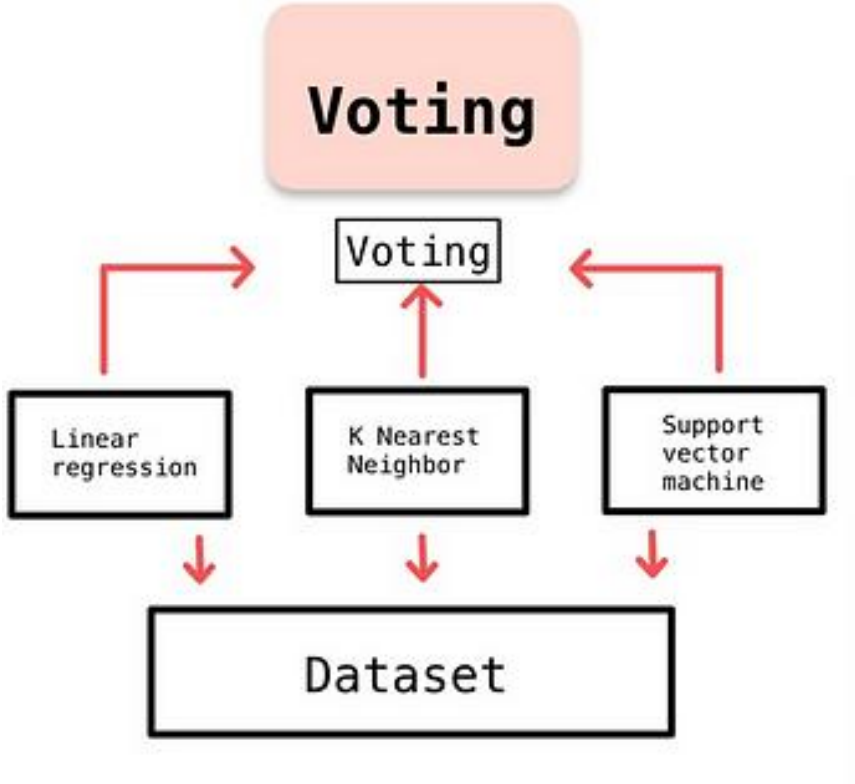
Voting vs. Bagging



[source: https://tinyurl.com/45hke9vt]

Voting Classification

1. Choosing Base Algorithm
2. Training Multiple Base Classifiers
3. Final Prediction through Voting



Voting demonstration
[source: https://tinyurl.com/4k7fnmzk]

Hard Voting

- In hard voting, each individual classifier in the ensemble makes a prediction.
- The final prediction is the majority vote, i.e., the class that the majority of individual classifiers predict.
- It works well when the individual classifiers are diverse and have some independence.

Hard Voting

Example:

If three base classifiers vote [0, 1, 1], the voting classifier selects 1 as the final prediction because it's the most common choice.

Soft Voting

- In soft voting, each individual classifier in the ensemble predicts class probabilities (for classification problems).
- The final prediction is based on the weighted average of the predicted probabilities.
- It's particularly useful when individual classifiers can output class probabilities, such as in the case of logistic regression or support vector machines with probability outputs.

Soft Voting

Example:

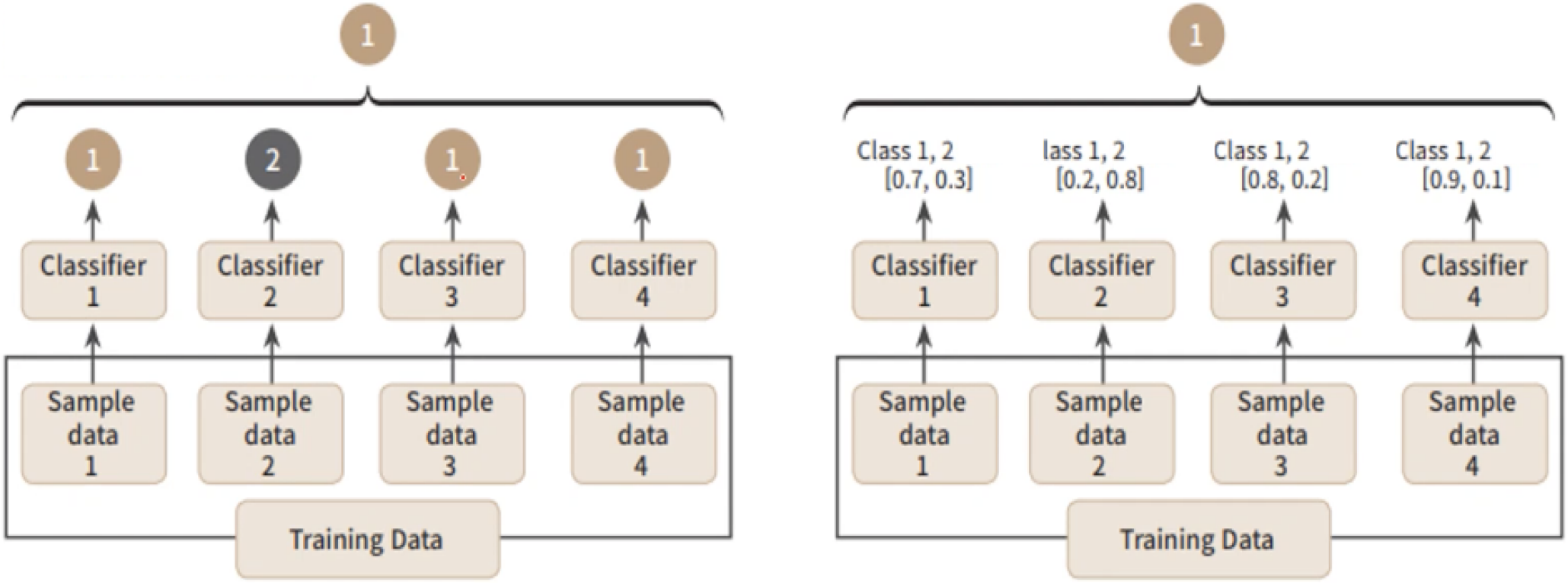
If three base classifiers predict probabilities like [0.2, 0.8], [0.3, 0.7], and [0.4, 0.6], the voting classifier outputs 1 because it has the highest probability of 0.7.

Hard Voting vs. Soft Voting

- Classification by hard-voting is like Winner Take All system. Out of multiple outputs produced by the classifiers, the majority output is chosen to be the final result of the model.

- In contrast, soft-voting is a voting process which every classifiers' outputs are taken into account. Soft-voting sums the predicted probabilities for class labels and returns the final classification with the largest sum probability.

Hard Voting vs. Soft Voting



[source: https://tinyurl.com/45hke9vt]

Voting Example with Python

```
# Load dataset
cancer = load_breast_cancer()
# Create dataframe and display first 3 rows
data_df = pd.DataFrame(cancer.data, columns=cancer.feature_names)
data_df.head(3)
print(data_df.head(3))
# Create 2 Classifiers for later using in the voting ensemble
lr_clf = LogisticRegression(max_iter=200)
knn_clf = KNeighborsClassifier(n_neighbors=8)
# Create a Voting Classifier that combines the predictions of LR and KNN
vo_clf = VotingClassifier(estimators=[('LR', lr_clf), ('KNN', knn_clf)], voting='soft')
# Split dataset into training and test sets
X_train, X_test, y_train, y_test = train_test_split(cancer.data, cancer.target, test_size=0.2, random_state=156)
# Fit the Voting Classifier on the training data and make prediction on the test data
vo_clf.fit(X_train, y_train)
pred = vo_clf.predict(X_test)
print('Voting Classifier Accuracy : {0:4f} \n'.format(accuracy_score(y_test, pred)))
# Iterate over a list of individual classifiers (LR and KNN), fit on training data, make pred on test data
classifiers = [lr_clf, knn_clf]
for classifier in classifiers:
    classifier.fit(X_train, y_train)
    pred = classifier.predict(X_test)
    class_name = classifier.__class__.__name__
    print('{0} Classifier Accuracy {1} \n'.format(class_name, accuracy_score(y_test, pred)))
```

[source: https://tinyurl.com/45hke9vt]

Voting Example with Python

Program output showing first 3 rows of the dataset:

	mean radius	mean texture	...	worst symmetry	worst fractal dimension
0	17.99	10.38	...	0.4601	0.11890
1	20.57	17.77	...	0.2750	0.08902
2	19.69	21.25	...	0.3613	0.08758

Program outputs:

```
Voting Classifier Accuracy : 0.956140
LogisticRegression Classifier Accuracy 0.9473684210526315
KNeighborsClassifier Classifier Accuracy 0.9385964912280702
```

Voting Classification- Pros

1. **Improving Model Performance:** Voting classification combines the predictions of multiple models to enhance overall accuracy. It's like taking a poll from different experts to make a more informed decision.

2. **Reducing Bias and Individual Weaknesses:** By considering the opinions of multiple models, voting can help mitigate biases and weaknesses present in any single model. It's similar to getting different viewpoints to make a fairer judgment.

3. **Error Reduction:** Voting can help minimize errors by averaging or selecting the most common prediction. It's akin to double-checking with others to reduce mistakes.

4. **Reduce Overfitting:** Voting can reduce overfitting by combining models with different strengths, making it less likely that any single model will fit the training data too closely. It's like balancing different approaches to prevent extreme behavior.

Voting Classification- Cons

1. **Sensitivity to Outliers:** Voting methods can be sensitive to outliers (unusual data points), potentially skewing the results. It's like a few loud voices in a vote having a disproportionate influence.

2. **Impact on Majority Vote and Probabilities:** Voting may not provide probabilities or account for the significance of predictions within each model, potentially missing valuable information. It's like treating every opinion equally, even when some are more reliable.

3. **Incorrect Classification:** Voting can still make incorrect classifications if most models agree on a wrong prediction. It's akin to following the crowd, even if the crowd is mistaken.

Bagging Classifier

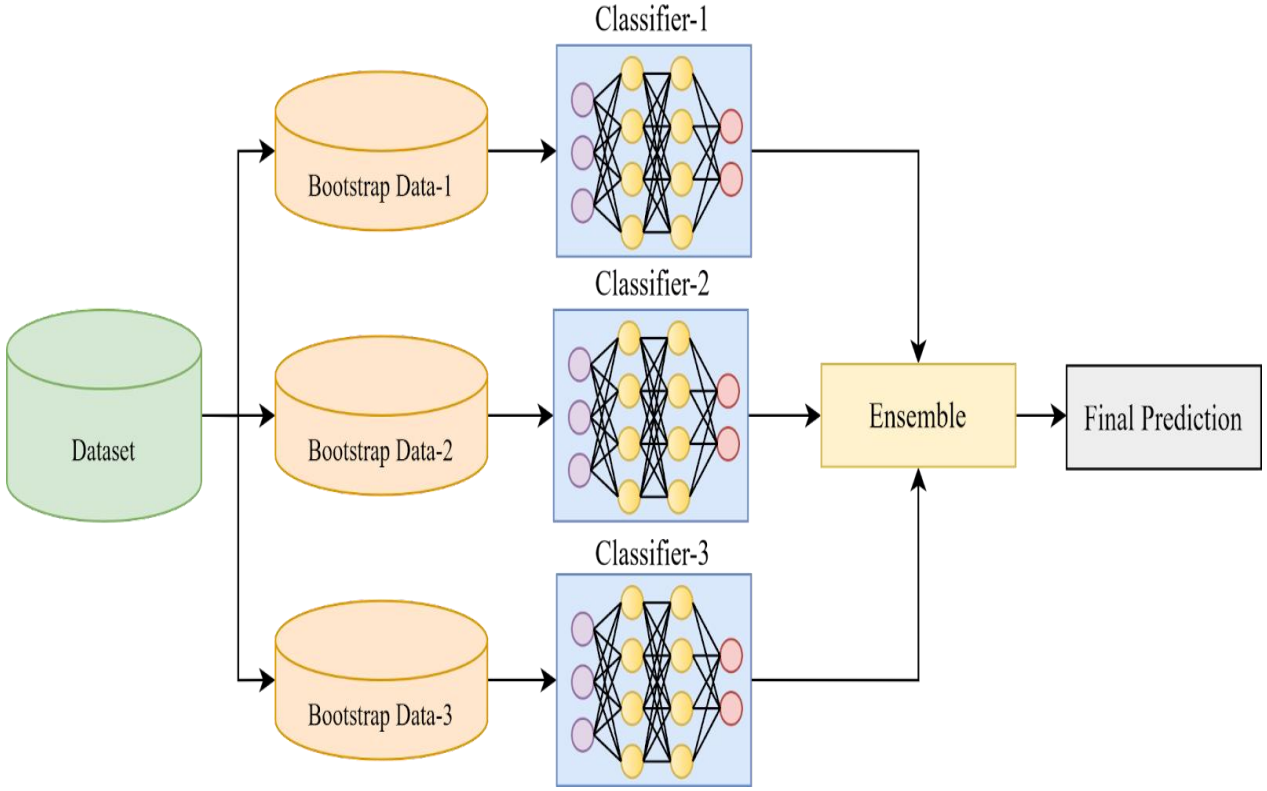
1. Bootstrap Sampling

2. Diverse Base Classifiers

3. Individual Predictions

4. Aggregation of Predictions

5. Common Base Classifiers: Random Forest, Decision Tree



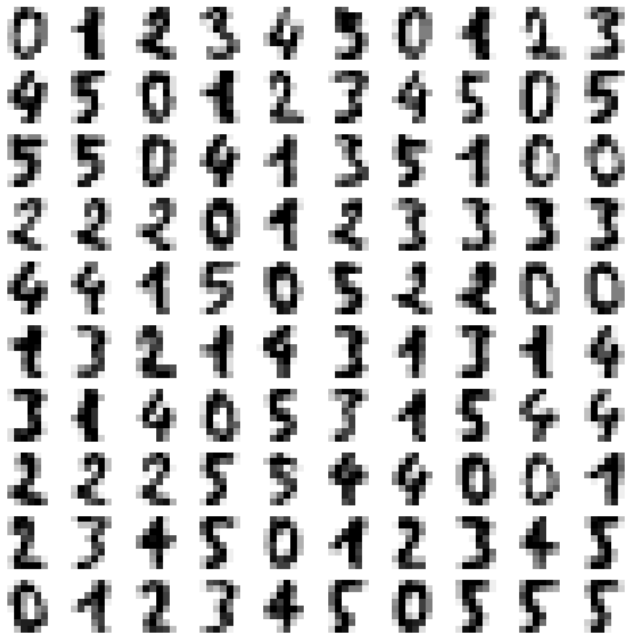
Representation of Bagging
[source: https://tinyurl.com/yzhw6umn]

Bagging Example With Python

```
# Load the dataset
digit = load_digits()
X, y = digit.data, digit.target
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Create the base classifier
base_classifier = DecisionTreeClassifier()
# Number of base models (iterations)
n_estimators = 10
# Create the Bagging classifier
bagging_classifier = BaggingClassifier(base_estimator=base_classifier, n_estimators=n_estir
# Train the Bagging classifier
bagging_classifier.fit(X_train, y_train)
# Make predictions on the test set
y_pred = bagging_classifier.predict(X_test)
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

[source: https://tinyurl.com/vczsux66]

Bagging Example With Python



A selection from the 64-dimensional digits dataset
[source: https://tinyurl.com/5yt6weue]

Accuracy: 0.9361111111111111

Figure: Output of the program

Bagging Classifier- Pros

1. **Reducing Variance:** Bagging reduces the variance in model predictions by combining multiple models trained on different subsets of the data. It's like getting multiple opinions to make a more stable decision.

2. **Random Sampling for Variability:** Bagging uses random sampling to create diverse subsets of the data for training each model. This diversity helps improve the overall model's performance and robustness. It's like looking at different random samples of a problem to gain a more comprehensive understanding.

Bagging Classifier- Cons

1. **Increased Complexity:** Bagging involves training multiple models, which can make the overall system more complex and harder to manage. It's like having many pieces to a puzzle, making it more challenging to assemble.

2. **Reduced Interpretability:** The combined predictions from bagging can be harder to interpret because they come from multiple models. It's like listening to a chorus of voices; it can be challenging to identify which voice is saying what.

3. **Computation and Memory Intensive:** Bagging requires substantial computational power and memory as it trains and maintains multiple models simultaneously. It's similar to needing a lot of processing power and memory to handle many tasks at once.

Conclusion

- Ensemble methods combine multiple models for predictions.
- Voting and bagging are techniques used in ensemble methods.
- Voting classifiers use base models to vote on class predictions.
- Bagging classifiers create random data samples for base classifiers and random sampling reduces model variance and mitigates overfitting.
- Ensemble models reduce bias, benefiting unbalanced data.

References

[1] Velog.io. [Online]. Available: https://velog.io/@jiselectric/Ensemble-Learning-Voting-and-Bagging-at6219ae . [Accessed: 02-Oct-2023].

[2] V7labs.com. [Online]. Available: https://www.v7labs.com/blog/ensemble-learning . [Accessed: 05-Oct-2023].

[3] Medium.com. [Online]. Available: https://medium.com/@chyun55555/ensemble-learning-voting-and-bagging-with-python-40de683b8ff0 . [Accessed: 08-Oct-2023].

[4] scikit. [Online]. Available: https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html#sphx-glr-auto-examples-manifold-plot-lle-digits-py . [Accessed: 25-Oct-2023].

[5] GeeksforGeeks. [Online]. https://www.geeksforgeeks.org/ml-bagging-classifier/ . [Accessed: 25-Oct.-2023].