Adaboost algorithm: Binary classification

Learning framework

Two key implementations

Evaluation

Adaboost specification

Input data

SET OF INDEPENDENT VARIABLES AND A BINARY TARGET

Output data

SET OF PREDICTIONS

Data format

SCALED NUMERICAL VARIABLES

Expected behaviour

COMBINE WEAK CLASSIFIERS (DECISION TREES WITH 1 DEPTH) TO GET ACCURATE BINARY PREDICTIONS

Adaboost algorithm: Binary classification

- **Hypothesis:** Combining weak classifiers $c_j(X)$ can improved the overall accuracy of the prediction $C_M(X)$.

$$C_M(X) = sign(\sum_{j=1}^{M} \alpha_j * c_j(X))$$

Significance of each weak classifier

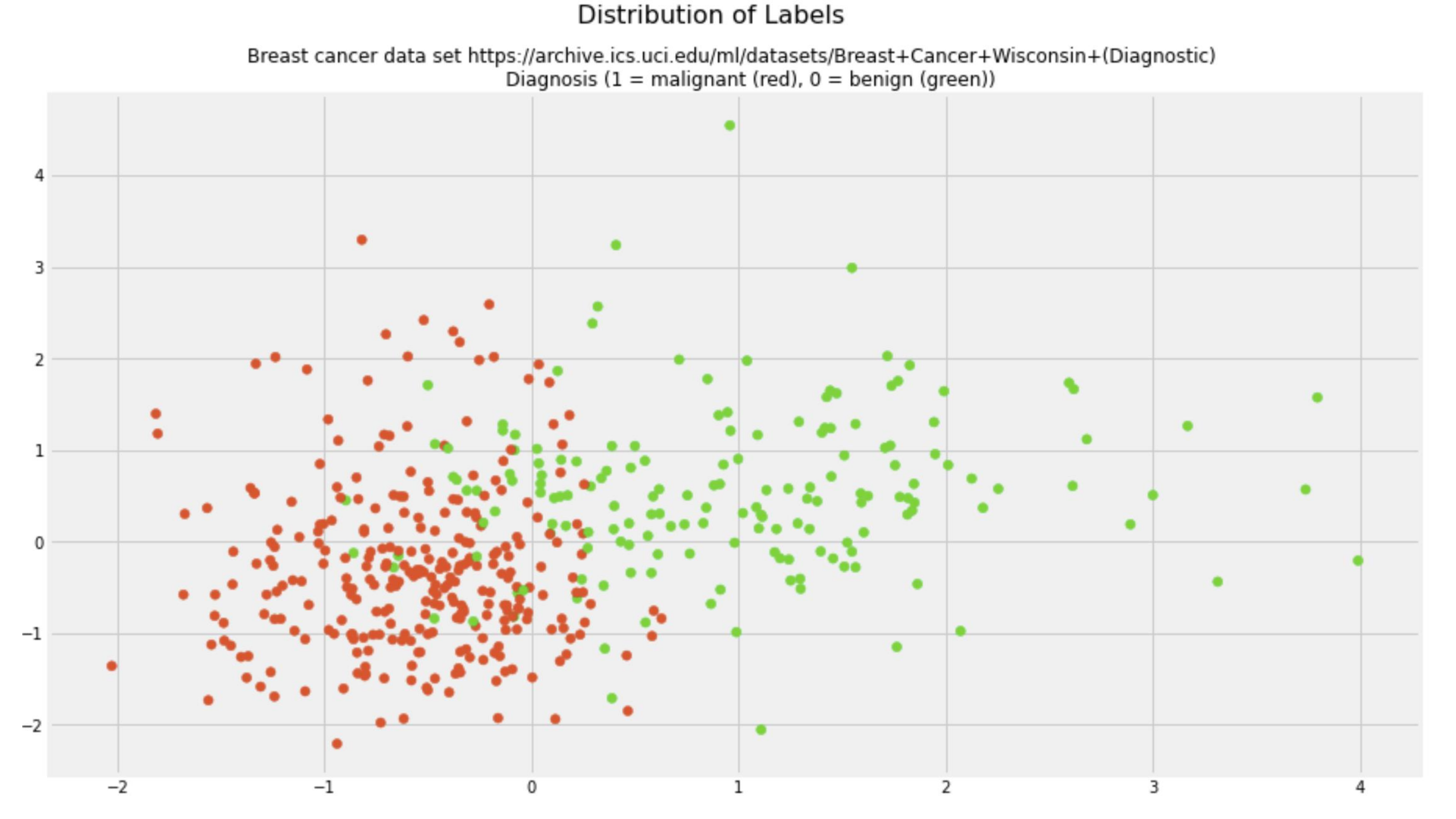
$$\alpha_j = \frac{1}{2}\log\frac{1 - Error_j}{Error_j}$$

Adaboost algorithm: Binary classification

- **Loss function:** Uses an exponential with the weight of each boosting round.

$$w_i^{(j+1)} = \frac{w_i^j}{Z_j} * \begin{cases} e^{-\alpha_j} & if\ Prediction\ of\ base\ classifier(x_i) = y_i \\ e^{\alpha_j} & if\ Prediction\ of\ base\ classifier(x_i)\ != y_i \end{cases}$$

$$Error_j = \frac{1}{N}\sum_{i=1}^{N} w_i * Misclassification_i$$


Distribution of Labels
Breast cancer data set https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)
Diagnosis (1 = malignant (red), 0 = benign (green))
4
3
2
1
0
−1
−2
−2
−1
0
1
2
3
4

Adaboost algorithm: Binary classification

Two key implementations

Loss function:

Modifications can improve the algorithm performance.

Predictions:

Combining previous predictions deliver a better outcome.

1. Change loss function

$$w_i^{(j+1)} = \frac{w_i^j}{Z_j} * \begin{cases} e^{-\alpha_j} & if\ Prediction\ of\ base\ classifier(x_i) = y_i \\ e^{\alpha_j} & if\ Prediction\ of\ base\ classifier(x_i)\ != y_i \end{cases}$$

		Model Results	
		True	False
Ground Truth	True	26	19
	False	0	69

F1-SCORE: 87.90%

$$w_i^{(j+1)} = \frac{w_i^j}{Z_j} * \begin{cases} 1/(1+e^{-\alpha_j}) & if\ Prediction\ of\ base\ classifier(x_i) = y_i \\ 1/(1+e^{\alpha_j}) & if\ Prediction\ of\ base\ classifier(x_i)\ != y_i \end{cases}$$

		Model Results	
		True	False
Ground Truth	True	28	17
	False	0	69

F1-SCORE: 89.03%

2. Prediction

$$\boxed{C_M(X)} = sign(\sum_{j=1}^{M} \boxed{\alpha_j} * \boxed{c_j(X)})$$

```
# Each decision stump weak learner is multiplied by the corresponding Alpha value
for dsl_alpha, dsl_model in zip(self.alphas,self.models):
    prediction = dsl_alpha * dsl_model.predict(X)
    predictions.append(prediction)

# It is returned +1 for each value greater than 0, -1 for each value lesser than 0 and 0 if the value is zero
predictions = np.sign(np.sum(np.array(predictions),axis=0))
```

Evaluation

TRAINING

TESTING

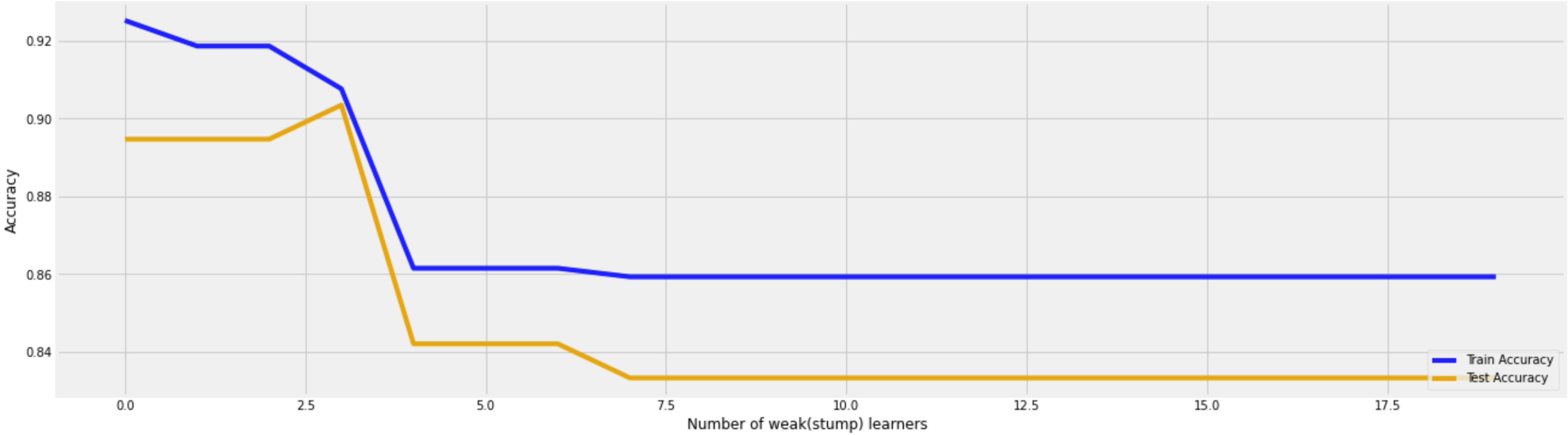
Trade-off between BIAS and VARIANCE

Confusion Matrix

$$w_i^{(j+1)} = \frac{w_i^j}{Z_j} * \begin{cases} e^{-\alpha_j} & if\ Prediction\ of\ base\ classifier(x_i) = y_i \\ e^{\alpha_j} & if\ Prediction\ of\ base\ classifier(x_i)\ != y_i \end{cases}$$

```
# Update the weights which are going to be used for the training of the next decision stump
weight_loop *= np.exp(alpha * np.where(misclassified == 1, 1, -1))
weight_loop /= np.sum(weight_loop)
weights_loop.append(weight_loop)
```

```
Max number of base models executed: 20. Accuracy training data set: 85.93%. Accuracy testing data set: 83.33%
```



$$w_i^{(j+1)} = \frac{w_i^j}{Z_j} * \begin{cases} 1/(1+e^{-\alpha_j}) & if\ Prediction\ of\ base\ classifier(x_i) = \ y_i \\ 1/(1+e^{\alpha_j}) & if\ Prediction\ of\ base\ classifier(x_i)\ ! = \ y_i \end{cases}$$

```
            # Update the weights which are going to be used for the training of the next decision stump
            weight_loop *= 1/(1+ np.exp(alpha * np.where(misclassified == 1, 1, -1)))
            weight_loop /= np.sum(weight_loop)
            weights_loop.append(weight_loop)
```

```
Max number of base models executed: 20. Accuracy training data set: 87.91%. Accuracy testing data set: 85.09%
```

Thank you