


# A Primer to Ensemble Learning – Bagging and Boosting

 [analyticsindiamag.com/primer-ensemble-learning-bagging-boosting](https://analyticsindiamag.com/primer-ensemble-learning-bagging-boosting)

Rohit Garg

February 19, 2018



## Read Next



## How Google's Arts & Culture App Uses AI, Facial Recognition To Find Your Museum Doppelgänger



POST GRADUATE PROGRAMME IN  
ADVANCED BUSINESS ANALYTICS (PGP-ABA)

KNOW  
MORE

Ensemble is a machine learning concept in which multiple models are trained using the same learning algorithm. Bagging is a way to decrease the variance in the prediction by generating additional data for training from dataset using combinations with repetitions to produce multi-sets of the original data. Boosting is an iterative technique which adjusts the weight of an observation based on the last classification. If an observation was classified incorrectly, it tries to increase the weight of this observation. Boosting in general builds strong predictive models.

## Ensemble Methods

### Definition and Objective of Analysis

**Definition:** Ensemble methods combine several decision trees classifiers to produce better predictive performance than a single decision tree classifier. The main principle behind the ensemble model is that a group of weak learners come together to form a strong learner, thus increasing the accuracy of the model.



**Objective of Analysis:** Minimization of risk and maximization of profit for the bank. To minimize loss from the bank's perspective, the bank needs a decision rule regarding loan approval. An applicant's demographic and socio-economic profiles are considered by loan managers before a decision is taken regarding his/her loan application. **The dataset contains data on 13 variables and the classification whether an applicant is considered a credit worthy (1) or a non-credit worthy (0) for 1000 loan applicants.** A predictive model developed on this data should provide the bank manager guidance for making a decision whether to approve a loan to a prospective applicant based on his/her profile.

**Cost-Profit Consideration:** The statistical decisions must be translated into profit consideration for the bank. Let us assume:

Leaders look at data

IIM Indore's Integrated Program  
in Business Analytics (IPBA)



- **Correct decision:** A correct decision of the bank manager would result in 35% profit at the end of 5 years. A correct decision here means that the bank manager predicts an application to be good or credit-worthy and it actually turns out to be credit worthy.
- **Wrong decision:** When the bank manager predicts the application to be good but it turns out to be bad credit, then the loss is 100%.
- If the bank manager predicts an application to be non-creditworthy, then loan facility is not extended to that applicant and bank does not incur any loss. The cost matrix, therefore, is as follows:

		Predicted	
		Credit Worthy	Non-Credit Worthy
Actual	Credit Worthy	+0.35	0.00
	Non-Credit Worthy	-1.00	0.00

## Dataset Source and Contents

The following is a description of the dataset:

- **of Classes:** 2 ('Credit Worthy' and 'Non-Credit Worthy')
- **of attributes (Columns):** 13
- **of instances (Rows):** 1,000

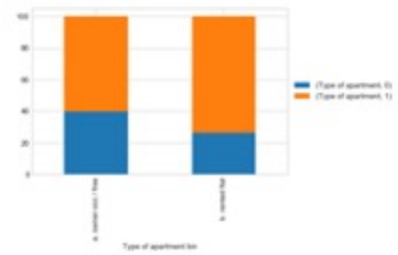
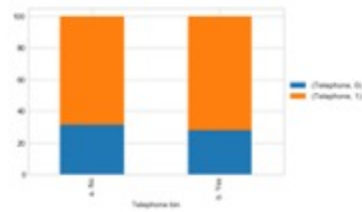
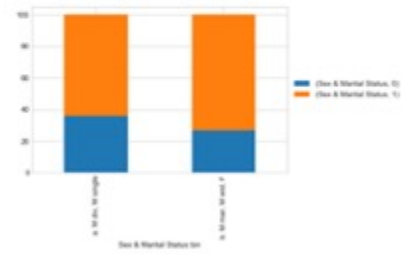
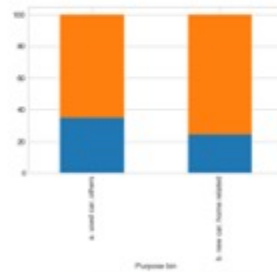
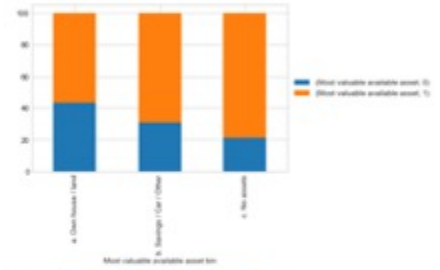
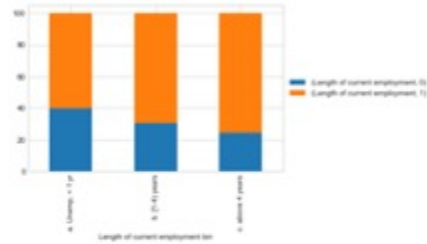
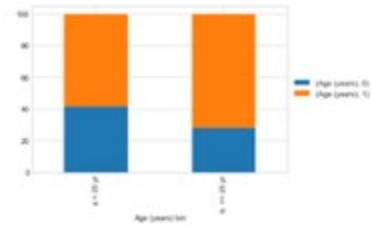
This data was extracted from the census bureau database found at:

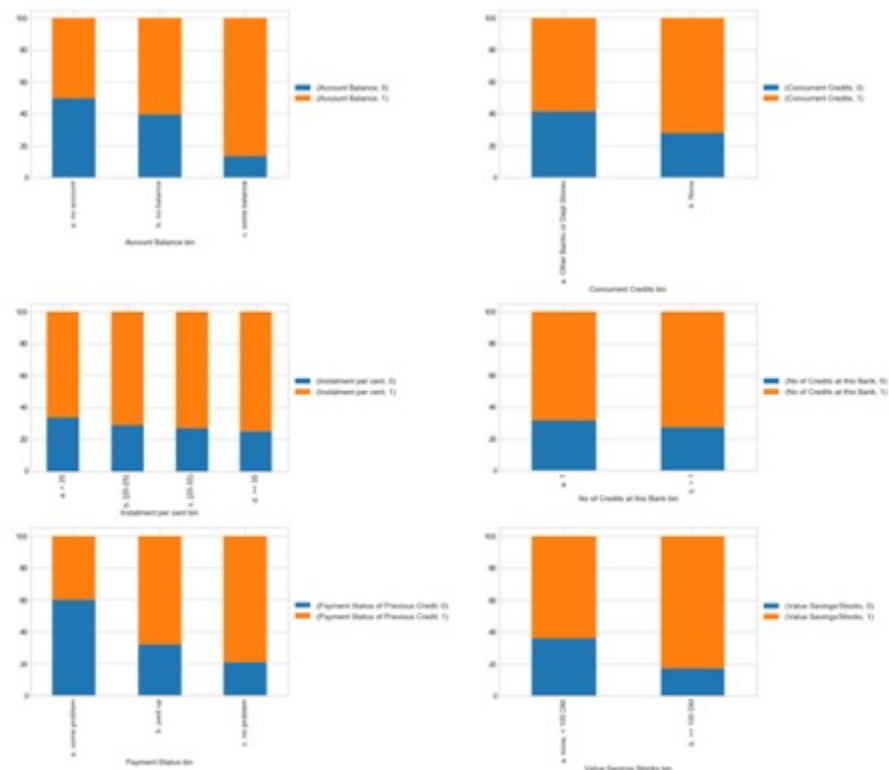
<http://archive.ics.uci.edu/ml/datasets>

## Exploratory Data Analysis

There are 13 explanatory variables:

1. Age
2. Length of Current Employment
3. Most Valuable Asset
4. Purpose
5. Sex and Marital Status
6. Telephone
7. Type of Apartment
8. Account Balance
9. Concurrent Credits
10. Instalment Percent
11. Number of Credits at this Bank
12. Payment Status
13. Value of Savings / Stocks





## Bagging and Boosting

### Bagging

Partitioning of data	Random
Goal to achieve	Minimum variance
Methods used	Random subspace
Functions to combine single model	Weighted average
Example	Random Forest

#### Definition:

Bagging is used when the goal is to reduce the variance of a decision tree classifier. **Here the objective is to create several subsets of data from training sample chosen randomly with replacement. Each collection of subset data is used to train their decision trees.** As a result, we get an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single decision tree classifier.

#### Bagging Steps:

- Suppose there are N observations and M features in training data set. A sample from training data set is taken randomly with replacement.
- A subset of M features are selected randomly and whichever feature gives the best split is used to split the node iteratively.
- The tree is grown to the largest.
- Above steps are repeated n times and prediction is given based on the aggregation of predictions from n number of trees.

**Advantages:**

- Reduces over-fitting of the model.
- Handles higher dimensionality data very well.
- Maintains accuracy for missing data.

**Disadvantages:**

Since final prediction is based on the mean predictions from subset trees, it won't give precise values for the classification and regression model.

**Python Syntax:**

- `rfm = RandomForestClassifier(n_estimators=80, oob_score=True, n_jobs=-1, random_state=101, max_features = 0.50, min_samples_leaf = 5)`
- `fit(x_train, y_train)`
- `predicted = rfm.predict_proba(x_test)`

## Boosting

<b>Partitioning of data</b>	<b>Higher vote to misclassified samples</b>
<b>Goal to achieve</b>	<b>Increase accuracy</b>
<b>Methods used</b>	<b>Gradient descent</b>
<b>Functions to combine single model</b>	<b>Weighted majority vote</b>
<b>Example</b>	<b>Ada Boost</b>

**Definition:**

Boosting is used to create a collection of predictors. **In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analysing data for errors. Consecutive trees (random sample) are fit and at every step, the goal is to improve the accuracy from the prior tree.** When an input is misclassified by a hypothesis, its weight is increased so that next hypothesis is more likely to classify it correctly. This process converts weak learners into better performing model.

## Boosting Steps:

- Draw a random subset of training samples  $d_1$  without replacement from the training set  $D$  to train a weak learner  $C_1$
- Draw second random training subset  $d_2$  without replacement from the training set and add 50 percent of the samples that were previously falsely classified/misclassified to train a weak learner  $C_2$
- Find the training samples  $d_3$  in the training set  $D$  on which  $C_1$  and  $C_2$  disagree to train a third weak learner  $C_3$
- Combine all the weak learners via majority voting.

## Advantages:

- Supports different loss function (we have used 'binary:logistic' for this example).
- Works well with interactions.

## Disadvantages:

- Prone to over-fitting.
- Requires careful tuning of different hyper-parameters.

## Python Syntax:

- `from xgboost import XGBClassifier`
- `xgb = XGBClassifier(objective='binary:logistic', n_estimators=70, seed=101)`
- `fit(x_train, y_train)`
- `predicted = xgb.predict_proba(x_test)`

## Conclusion

---

## Comparison Matrix

---

- **Accuracy: (True Positive + True Negative) / Total Population**
  - **Accuracy is a ratio of correctly predicted observation to the total observations. Accuracy is the most intuitive performance measure.**
  - True Positive: The number of correct predictions that the occurrence is positive
  - True Negative: The number of correct predictions that the occurrence is negative
- **F1-Score: (2 x Precision x Recall) / (Precision + Recall)**
  - **F1-Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. F1-Score is usually more useful than accuracy, especially if you have an uneven class distribution.**
  - Precision: When a positive value is predicted, how often is the prediction correct?
  - Recall: When the actual value is positive, how often is the prediction correct?

Classification Algorithms	Accuracy	F1-Score
Decision Tree	76.67 %	0.74
Bagging	77.67 %	0.77
Boosting	78.67 %	0.78

Code location: <https://github.com/f2005636/Ensemble>

## Cost-Profit Consideration

Although the improvement in **Accuracy** and **F1-Score** for Bagging and Boosting is not significant when compared to the Decision Tree, **significant improvement is seen for overall profit.**

NCW (0): Non-Credit  
Worthy

CW (1): Credit Worthy

Decision Tree			Bagging			Boosting		
	NCW	CW		NCW	CW		NCW	CW
NCW	34	56	NCW	48	42	NCW	50	40
CW	14	196	CW	25	185	CW	24	186

Classification Algorithms	Calculation	Profit
Decision Tree	$0.35 * 196 - 1.00 * 56$	12.60
Bagging	$0.35 * 185 - 1.00 * 42$	22.75
Boosting	$0.35 * 186 - 1.00 * 40$	25.10

*Enjoyed this story? Join our Telegram group. And be part of an engaging community.*

## Provide your comments below

comments

Rohit Garg

Rohit Garg has close to 7 years of work experience in field of data analytics and machine learning. He has worked extensively in the areas of predictive modeling, time series analysis and segmentation techniques. Rohit holds BE from BITS Pilani and PGDM from IIM Raipur.



