THE BRIDGE

Ensamblado de Modelos

Bagging: Random Forest

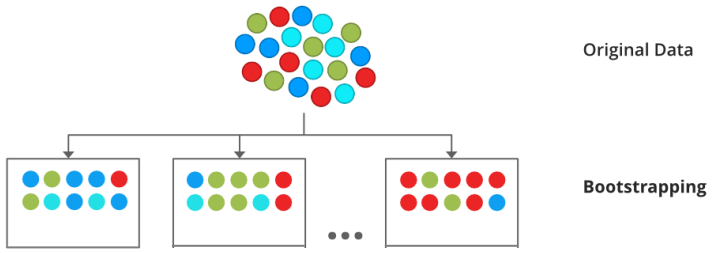
Bagging: Boostrap Aggregating

Con esta técnica se entrenan un conjunto de modelos del mismo tipo, mediante muestras **con reemplazamiento**.
(Esta es la parte del boostrap)

Bagging: Boostrap Aggregating

Con esta técnica se entrenan un conjunto de modelos del mismo tipo, mediante muestras **con reemplazamiento**. (Esta es la parte del boostrap)

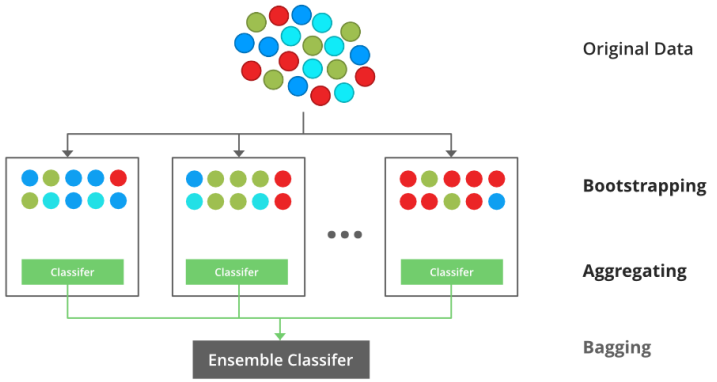
Es decir partimos aleatoriamente el dataset de train en tantos subdatasets como modelos formen el ensamblado y esos subdatasets pueden repetir elementos. (eso es el bootstrap)



Bagging: Boostrap Aggregating

Con esta técnica se entrenan un conjunto de modelos del mismo tipo, mediante muestras **con reemplazamiento**. (Esta es la parte del boostrap)

Luego cada modelo se entrena con su subdataset. Cuando hay que predecir o estimar una salida (cuando llamamos al predict), cada modelo hace su predicción y esas predicciones se agregan (esa es la parte del aggregating) y se usa un mecanismo de voto (como los que vimos en la sesión anterior).

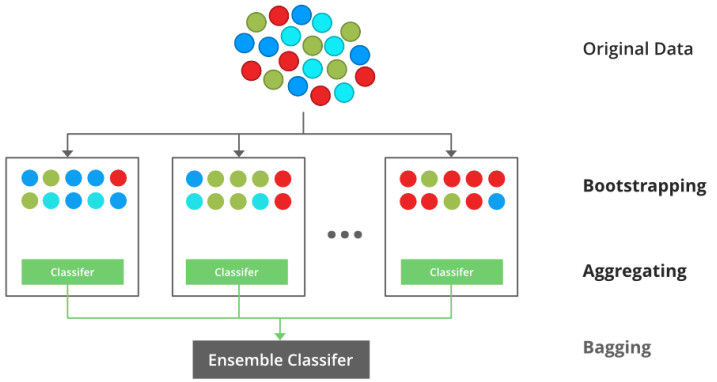


Bagging: Boostrap Aggregating

Con esta técnica se entrenan un conjunto de modelos del mismo tipo, mediante muestras **con reemplazamiento**. (Esta es la parte del boostrap)

Luego cada modelo se entrena con su subdataset. Cuando hay que predecir o estimar una salida (cuando llamamos al predict), cada modelo hace su predicción y esas predicciones se agregan (esa es la parte del aggregating) y se usa un mecanismo de voto (como los que vimos en la sesión anterior).

En general para clasificación sale la clase más votada y para regresión la media de las estimaciones de todos los modelos.

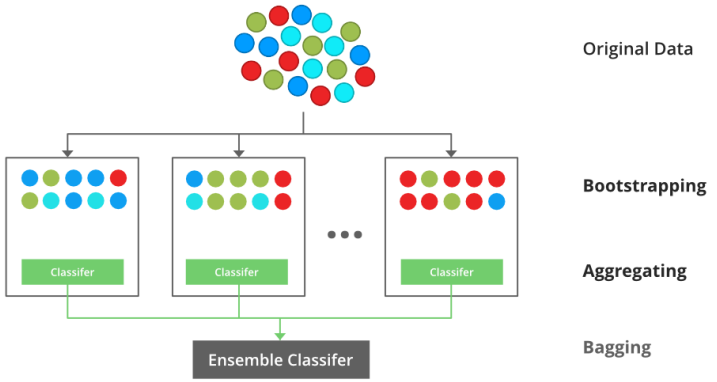


Bagging: Boostrap Aggregating

Con esta técnica se entrenan un conjunto de modelos del mismo tipo, mediante muestras **con reemplazamiento**. (Esta es la parte del boostrap)

Luego cada modelo se entrena con su subdataset. Cuando hay que predecir o estimar una salida (cuando llamamos al predict), cada modelo hace su predicción y esas predicciones se agregan (esa es la parte del aggregating) y se usa un mecanismo de voto (como los que vimos en la sesión anterior).

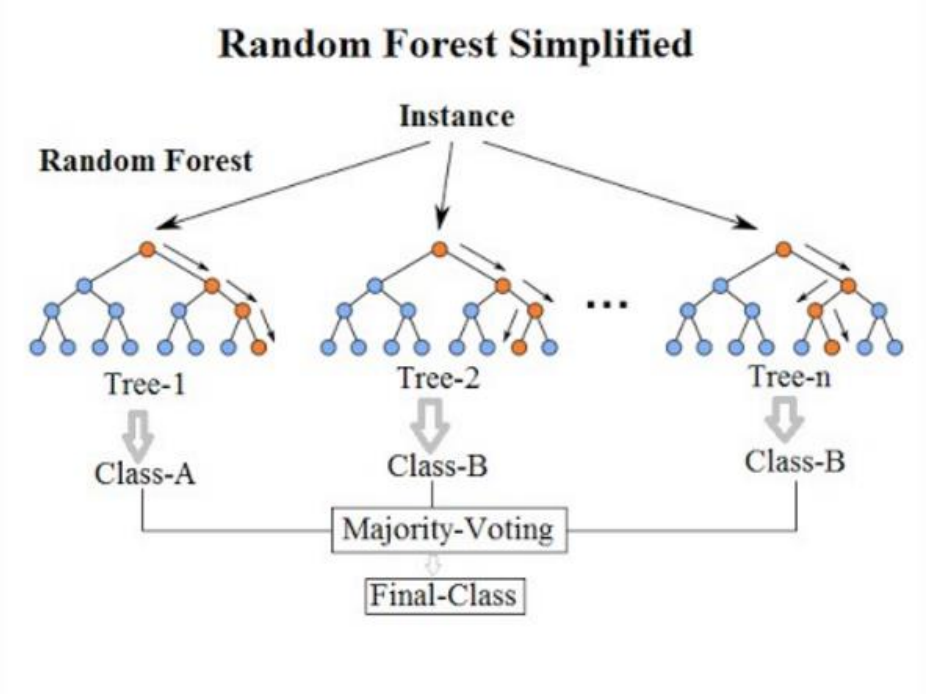
En general para clasificación sale la clase más votada y para regresión la media de las estimaciones de todos los modelos.



Existe una vairante en la que los subdatasets son exclusivos, no comparten instancias… Se le denomina **Pasting**

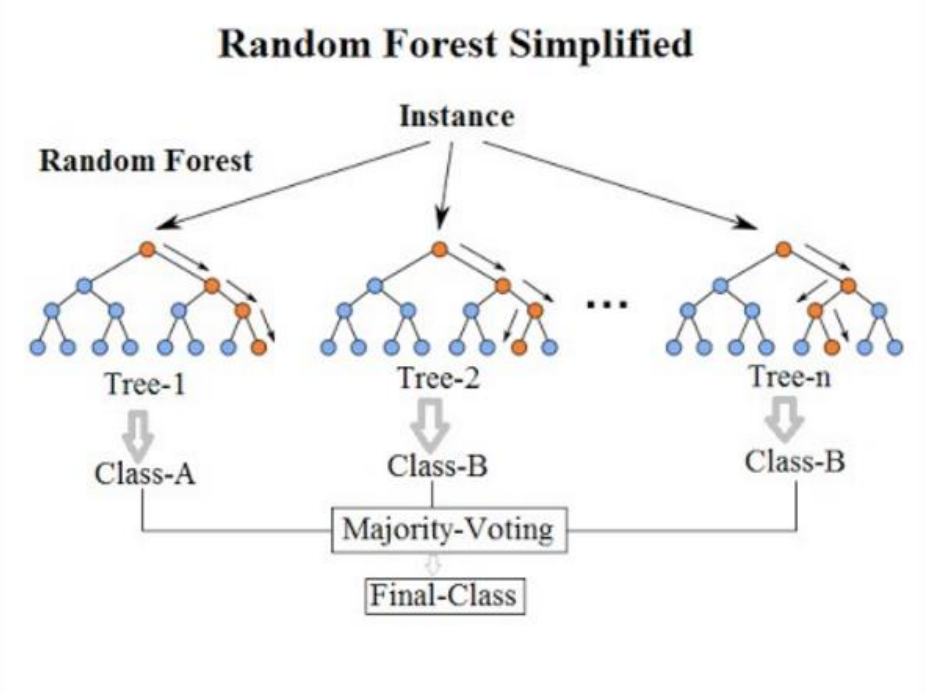
Bagging: Random Forest

- El algoritmo de bagging que más se utiliza es el random forest.



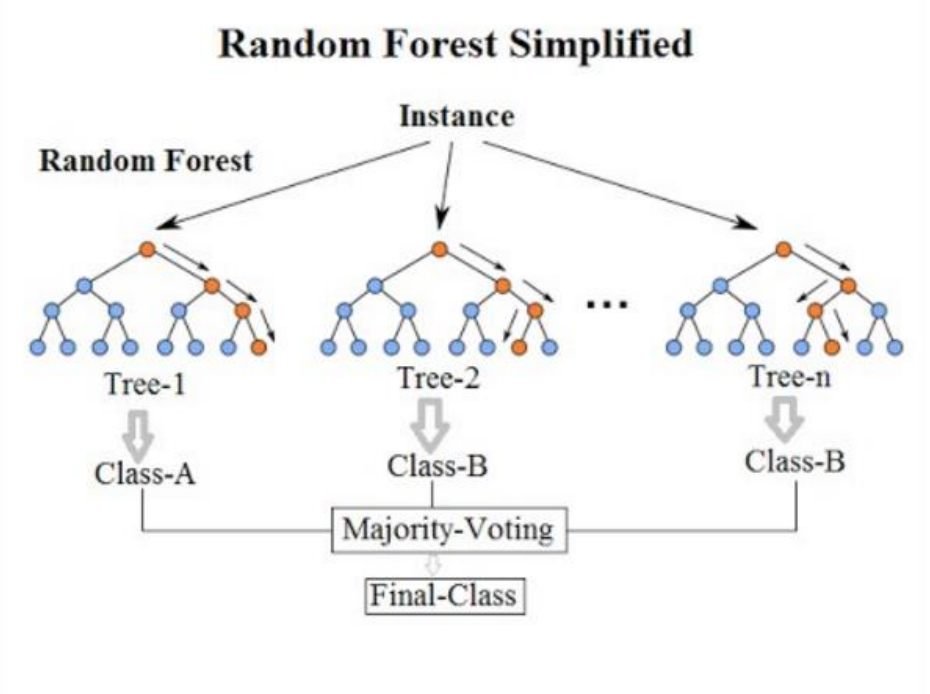
Bagging: Random Forest

- El algoritmo de bagging que más se utiliza es el random forest.
- Se denomina así porque emplea como modelos internos los árboles de decisión.



Bagging: Random Forest

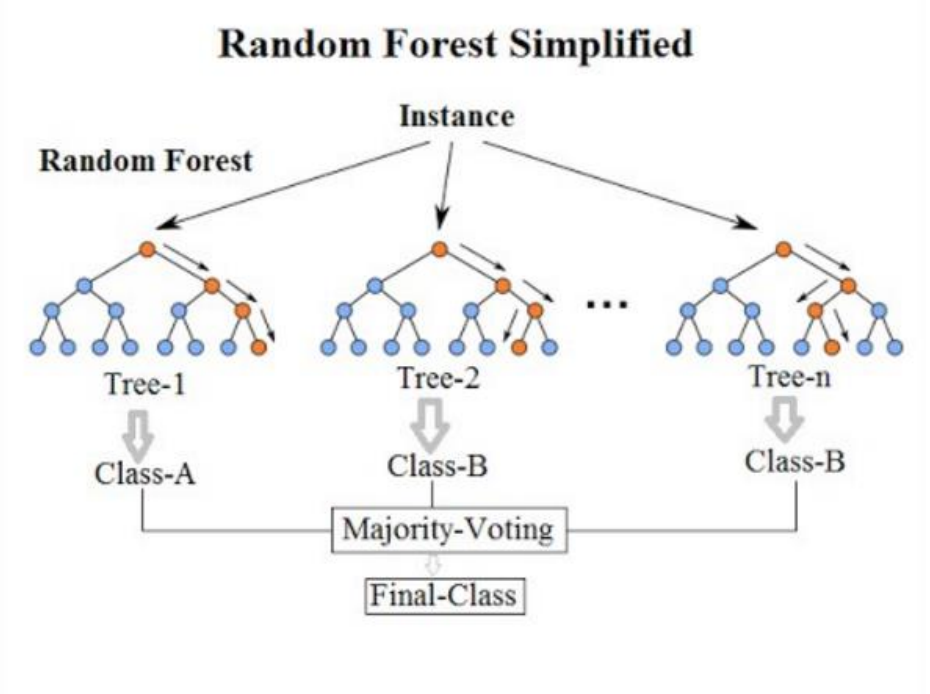
- El algoritmo de bagging que más se utiliza es el random forest.
- Se denomina así porque emplea como modelos internos los árboles de decisión.
- Dependiendo de si se emplea para clasificación o regresión usaremos árboles de decisión para clasificación o regresión



Bagging: Random Forest

El funcionamiento básico ya lo hemos comentado:

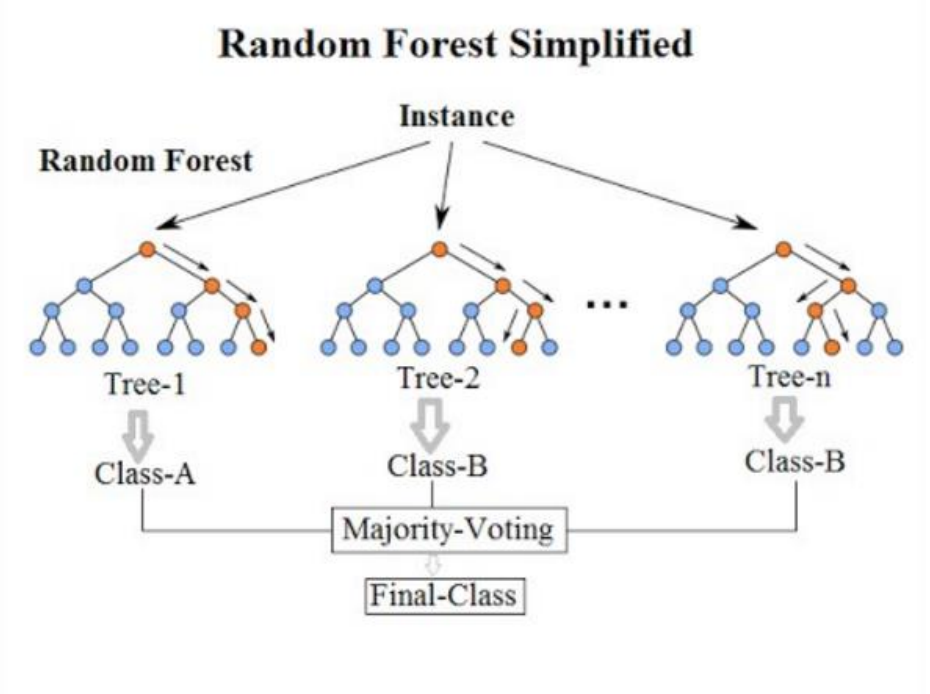
1. Se parte el dataset en tantos subsets como número de árboles



Bagging: Random Forest

El funcionamiento básico ya lo hemos comentado:

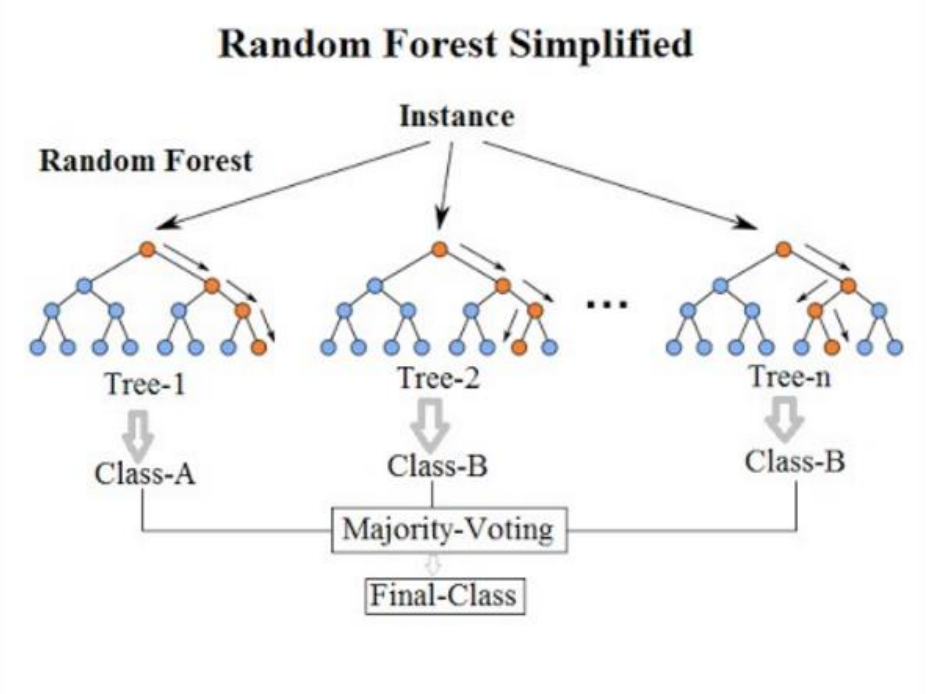
1. Se parte el dataset en tantos subsets como número de árboles
2. Aquí lo diferente: además cada subset puede tener todas o un número aleatorio de features de las totales



Bagging: Random Forest

El funcionamiento básico ya lo hemos comentado:

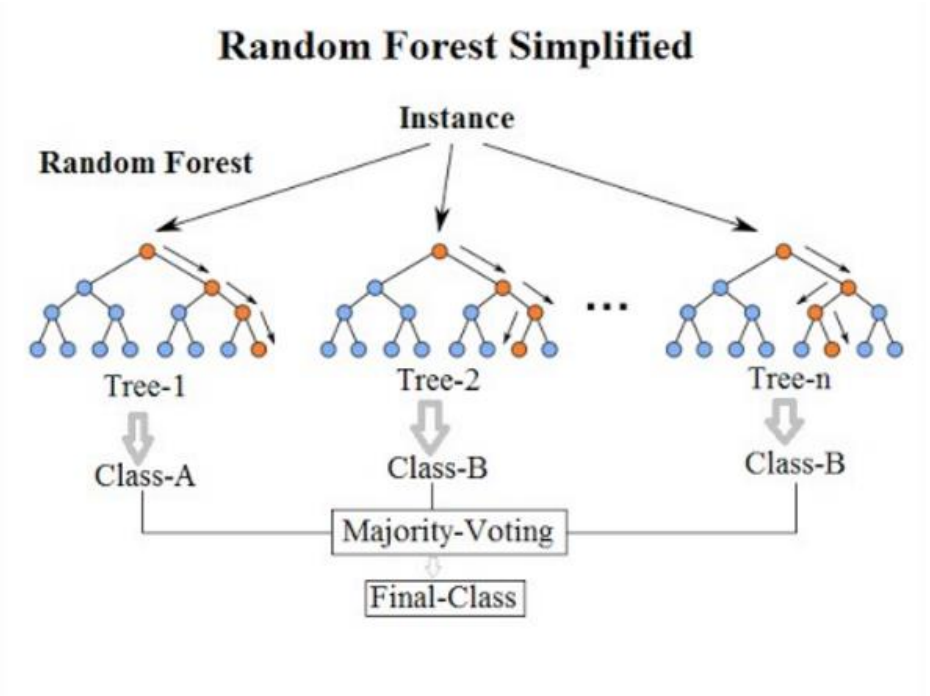
1. Se parte el dataset en tantos subsets como número de árboles
2. Aquí lo diferente: además cada subset puede tener todas o un número aleatorio de features de las totales
3. Se entrenan los árboles



Bagging: Random Forest

El funcionamiento básico ya lo hemos comentado:

1. Se parte el dataset en tantos subsets como número de árboles
2. Aquí lo diferente: además cada subset puede tener todas o un número aleatorio de features de las totales
3. Se entrenan los árboles
4. Se predice y, por tanto, evalúa según un mecanismo de voto de la mayoría para clasificación y media de resultados para regresión



Por supuesto, tú no vas a tener que implementar este mecanismo ya que como en los modelos que hemos visto hasta ahora todo se hace por debajo de su correspondiente clase en sklearn

Random Forest: Feature Importance

Al estar construido sobre árboles de decisión es fácil obtener la "importancia" de cada feature en el modelo final. Es decir cuánto aportan para la estimación final (sea clasificación o regresión)

Random Forest: Feature Importance

Al estar construido sobre árboles de decisión es fácil obtener la "importancia" de cada feature en el modelo final. Es decir cuánto aportan para la estimación final (sea clasificación o regresión)

Para cada feature se obtiene la aportación que ha hecho al gini o entropía de cada árbol en el que ha participado. Luego se obtiene la media ponderada de su aportación.

Random Forest: Feature Importance

Al estar construido sobre árboles de decisión es fácil obtener la "importancia" de cada feature en el modelo final. Es decir cuánto aportan para la estimación final (sea clasificación o regresión)

Para cada feature se obtiene la aportación que ha hecho al gini o entropía de cada árbol en el que ha participado. Luego se obtiene la media ponderada de su aportación.

Por suerte, y como practicaremos, sklearn ya realiza esta operación por nosotros, y lo normaliza a 1, de tal manera que las features más importantes estarán cercanas a 1.

```
>>> from sklearn.datasets import load_iris
>>> iris = load_iris()
>>> rnd_clf = RandomForestClassifier(n_estimators=500, n_jobs=-1)
>>> rnd_clf.fit(iris["data"], iris["target"])
>>> for name, score in zip(iris["feature_names"], rnd_clf.feature_importances_):
...     print(name, score)
...
sepal length (cm) 0.112492250999
sepal width (cm) 0.0231192882825
petal length (cm) 0.441030464364
petal width (cm) 0.423357996355
```

Random Forest: Hiperparámetros

- n_estimators
- max_Depth
- min_samples_leaf
- max_samples_split
- max_features
- class_Weight


THE
BRIDGE