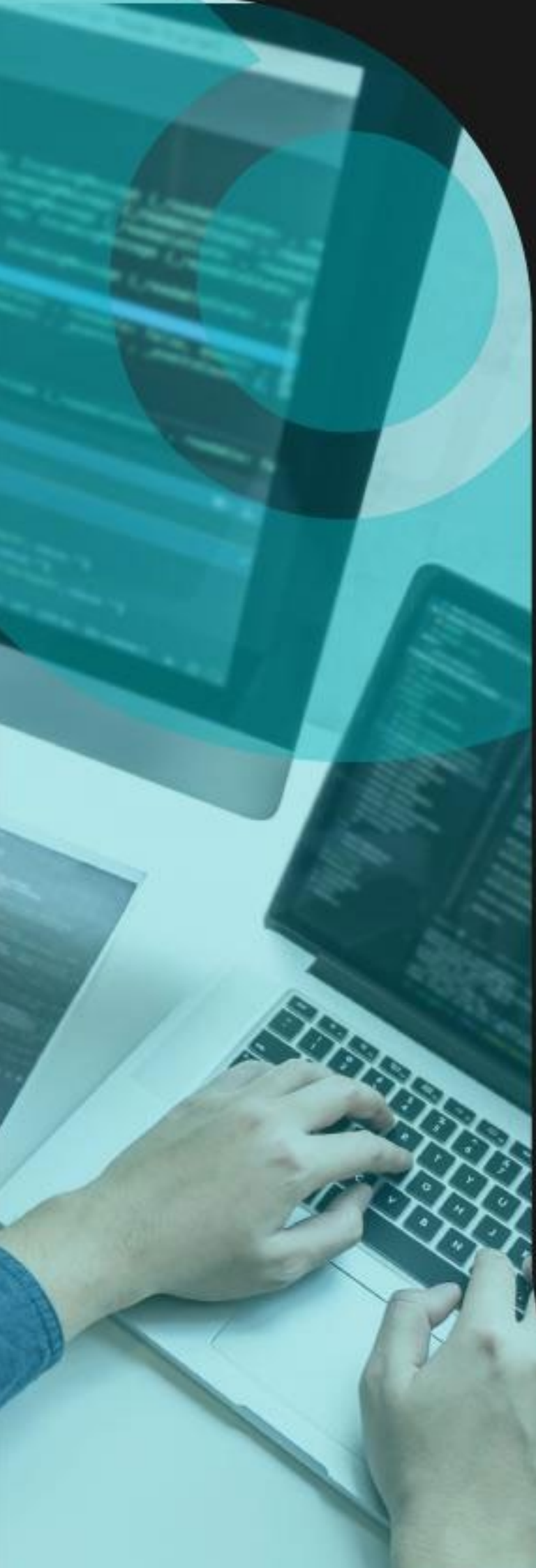


AWAKELAB

BASECAMP

Ciencia de Datos

Módulo: Aprendizaje de Máquina Supervisado

Aprendizaje Esperado

6. Elaborar un modelo predictivo aplicando el algoritmo Random Forest para resolver un problema de clasificación utilizando lenguaje Python.

Random Forest

Un bosque aleatorio es un algoritmo de aprendizaje automático supervisado que se construye a partir de algoritmos de árboles de decisión. Este algoritmo se aplica en varias industrias, como la banca y el comercio electrónico, para predecir el comportamiento y los resultados.

¿Cómo es este algoritmo?

Corresponde a una técnica de Machine Learning que se basa en un conjunto de árboles de decisiones seleccionando aleatoriamente submuestras (con reemplazamiento) para elaborar cada árbol.

El punto clave del Bosque de árboles es utilizar una serie de árboles de decisión (diferentes individuos y diferentes variables), con el fin de mejorar la tasa de clasificación correcta. La diferencia con el bagging es que en el bosque de árboles también se toma una muestra de los features, es decir, no se utilizan todas las variables como en el bagging.

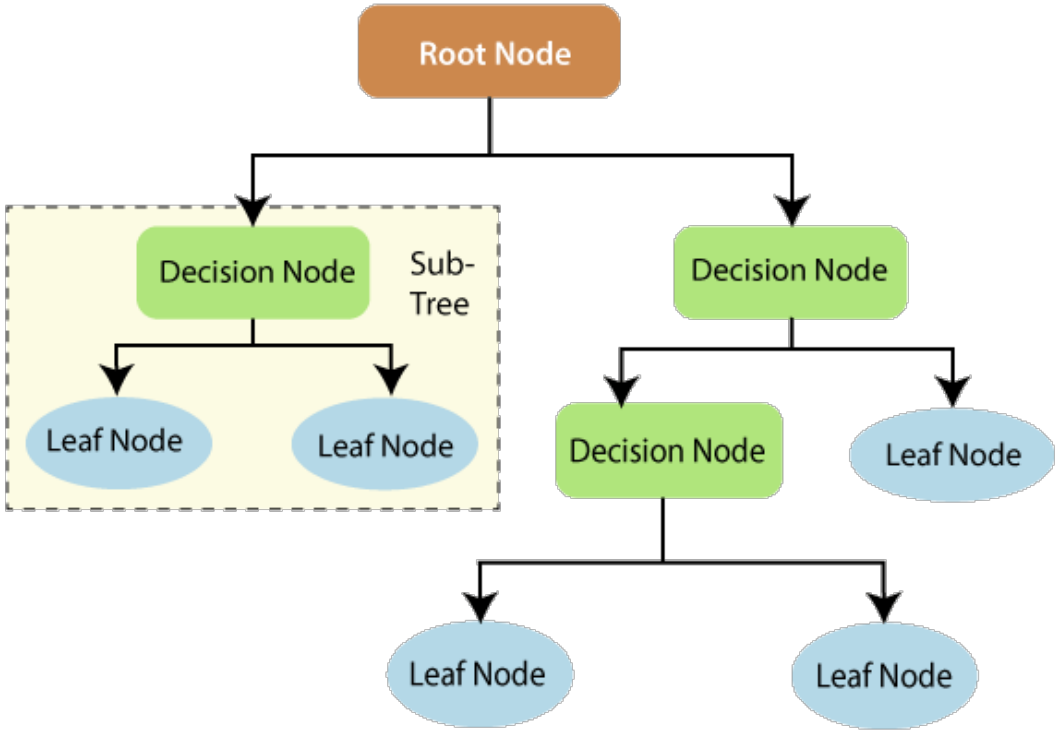
Características de un algoritmo de bosque aleatorio

- Es más preciso que el algoritmo del árbol de decisión.
- Proporciona una forma eficaz de manejar los datos que faltan.
- Puede producir una predicción razonable sin ajuste de hiperparámetros.
- Resuelve el problema del sobreajuste en los árboles de decisión.

- En cada árbol forestal aleatorio, se selecciona aleatoriamente un subconjunto de características en el punto de división del nodo.

Comprender los árboles de decisión

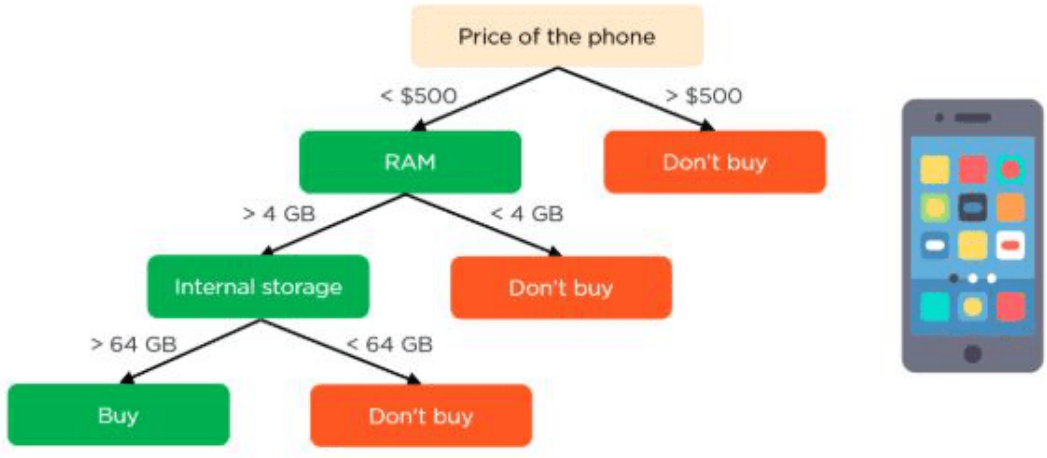
Los árboles de decisión son los componentes básicos de un algoritmo de bosque aleatorio. Un árbol de decisión es una técnica de apoyo a la decisión que forma una estructura similar a un árbol. Una descripción general de los árboles de decisión nos ayudará a comprender cómo funcionan los algoritmos de bosque aleatorio.
Un árbol de decisión consta de tres componentes: nodos de decisión, nodos hoja y un nodo raíz. Un algoritmo de árbol de decisiones divide un conjunto de datos de entrenamiento en ramas, que luego se segregan en otras ramas. Esta secuencia continúa hasta que se alcanza un nodo hoja. El nodo hoja no se puede segregar más.
Los nodos del árbol de decisión representan atributos que se utilizan para predecir el resultado. Los nodos de decisión proporcionan un enlace a las hojas. El siguiente diagrama muestra los tres tipos de nodos en un árbol de decisión.



Tomemos un ejemplo simple de cómo funciona un árbol de decisión. Supongamos que queremos predecir si un cliente comprará un

teléfono móvil o no. Las características del teléfono forman la base de su decisión. Este análisis se puede presentar en un diagrama de árbol de decisión.

El nodo raíz y los nodos de decisión de la decisión representan las funciones del teléfono mencionadas anteriormente. El nodo hoja representa el resultado final, ya sea *comprando* o *no comprando* . Las características principales que determinan la elección incluyen el precio, el almacenamiento interno y la memoria de acceso aleatorio (RAM). El árbol de decisión aparecerá de la siguiente manera.



Diferencias

La principal diferencia entre el algoritmo de **árbol de decisión** y el **algoritmo de bosque aleatorio** es que el establecimiento de nodos raíz y la segregación de nodos se realiza aleatoriamente en este último. El bosque aleatorio emplea el método de embolsado para generar la predicción requerida.
El embolsado implica el uso de diferentes muestras de datos (datos de entrenamiento) en lugar de una sola muestra. Un conjunto de datos de entrenamiento comprende observaciones y características que se utilizan para hacer predicciones. Los árboles de decisión producen diferentes resultados, según los datos de entrenamiento que se alimentan al algoritmo de bosque aleatorio. Estos resultados se clasificarán y el más alto se seleccionará como resultado final.

Dónde se puede aplicar hoy

Algunas de las aplicaciones del bosque aleatorio pueden incluir:

- *Bancario*

El bosque aleatorio se utiliza en la banca para predecir la solvencia de un solicitante de préstamo. Esto ayuda a la institución crediticia a tomar una buena decisión sobre si otorgar el préstamo al cliente o no. Los bancos también utilizan el algoritmo de bosque aleatorio para detectar a los estafadores.

- *Cuidado de la salud*

Los profesionales de la salud utilizan sistemas de bosques aleatorios para diagnosticar a los pacientes. Los pacientes se diagnostican evaluando su historial médico previo. Se revisan los registros médicos anteriores para establecer la dosis correcta para los pacientes.

- *Bolsa de Valores*

Los analistas financieros lo utilizan para identificar mercados potenciales para acciones. También les permite identificar el comportamiento de las acciones.

- *Comercio electrónico*

A través de los algoritmos de la selva tropical, los proveedores de comercio electrónico pueden predecir la preferencia de los clientes en función del comportamiento de consumo anterior.

Cuándo evitar el uso de bosques aleatorios

Los algoritmos de bosque aleatorio no son ideales en las siguientes situaciones:

- *Extrapolación*

La regresión de bosque aleatorio no es ideal en la extrapolación de datos. A diferencia de la regresión lineal, que usa observaciones existentes para estimar valores más allá del rango de observación. Esto explica por qué la mayoría de las aplicaciones del bosque aleatorio se relacionan con la clasificación.

- *Datos escasos*

El bosque aleatorio no produce buenos resultados cuando los datos son muy escasos. En este caso, el subconjunto de características y la muestra de arranque producirán un espacio invariante. Esto conducirá a divisiones improductivas, lo que afectará el resultado.

Tradeoff de bias versus variance (sesgo versus varianza)

El error debido al **Bias** de un modelo es simplemente la diferencia entre el valor esperado del estimador (es decir, la predicción media del modelo) y el valor real. Cuando se dice que un modelo tiene un bias muy alto quiere decir que el modelo es muy simple y no se ha ajustado a los datos de entrenamiento (suele ser underfitting), por lo que produce un error alto en todas las muestras: entrenamiento, validación y test.

Los modelos lineales suelen sufrir errores por Bias (Regresión Lineal. Qué es, para qué se utiliza y ejemplo práctico).

Sin embargo, hay algoritmos que tienen un bias pequeño como son los árboles de decisión (por ejemplo el Random Forest) , el KNN (Algoritmo KNN, conviértete en experto: cómo funciona y ejemplos en Python) y los SVM.

La **varianza** de un estimador es cuánto varía la predicción según los datos que utilicemos para el entrenamiento.

Como bien sabemos, la mayoría de algoritmos de Machine Learning aprenden según van entrando datos de entrenamiento. Así que es normal que todos los modelos tengan cierta varianza. Aunque si creamos un modelo robusto, debería aprender las relaciones entre las variables y el target.

Entonces, lo que se saca de esto es lo siguiente:

- Un modelo con varianza baja indica que cambiar los datos de entrenamiento produce cambios pequeños en la estimación.

- Al contrario, un modelo con varianza alta quiere decir que pequeños cambios en el dataset conlleva a grandes cambios en el output (suele ser overfitting).

Los algoritmos que suelen tener un error de bias alto suelen tener una varianza baja. A su vez, los que tienen bias bajo suelen tener varianza alta.

Ventajas	**Desventajas**
Puede realizar tanto tareas de regresión como de clasificación.	Cuando se usa un bosque aleatorio, se requieren más recursos para el cálculo.
Un bosque aleatorio produce buenas predicciones que pueden entenderse fácilmente.	Consume más tiempo en comparación con un algoritmo de árbol de decisión.
Puede manejar grandes conjuntos de datos de manera eficiente.	

El algoritmo de bosque aleatorio proporciona un mayor nivel de precisión en la predicción de resultados sobre el algoritmo de árbol de decisión.	

El algoritmo de la selva tropical es un algoritmo de aprendizaje automático que es fácil de usar y flexible. Utiliza el aprendizaje conjunto, que permite a las organizaciones resolver problemas de regresión y clasificación.

Este es un algoritmo ideal para desarrolladores porque resuelve el problema del sobreajuste de conjuntos de datos. Es una herramienta muy ingeniosa para hacer predicciones precisas necesarias en la toma de decisiones estratégicas en las organizaciones.

Ejemplo en Python

```
#librerias base

import numpy as np

import pandas as pd


from sklearn.metrics import mean_squared_error,
median_absolute_error,r2_score #metricas de
evaluacion
```

```
from sklearn.model_selection import train_test_split,
cross_val_score


#librerias

import warnings

warnings.filterwarnings("ignore")


df = pd.read_csv("ames_housing.csv").drop(['Unnamed:
0'],axis=1)


a=[]

for i,j in enumerate(df):

    if df[j].dtype=='object':

        a.append(j)

df2 = pd.get_dummies(df, columns=a,
prefix=a,drop_first=True)


df.MS_SubClass.value_counts()


df2.MS_SubClass_One_Story_1946_and_Newer_All_Styles

X_train, X_test, y_train, y_test =
train_test_split(df2.drop('Sale_Price',axis=1),


df2['Sale_Price'],test_size= .33 ,random_state= 65 )
```

```
from sklearn.ensemble import RandomForestRegressor

RanForReg = RandomForestRegressor(random_state=0)

RanForReg.fit(X_train,y_train)

print("MSE:", mean_squared_error(y_test,
RanForReg.predict(X_test)))

print("MAE:",
median_absolute_error(y_test,RanForReg.predict(X_test
)))

print("R2:", r2_score(y_test,
RanForReg.predict(X_test)))
```

Referencias

[1] Entendiendo Random Forest

https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

[2] Random Forest

https://www.bigdata-insider.de/was-ist-random-forest-a-913937/

[3] Ejemplos en py

https://programmerclick.com/article/84431135962/

Material Complementario

[1] Qué es el Random Forest

https://www.youtube.com/watch?v=VH7eLWsLCks

[2] Ejemplo de Predicción

https://www.youtube.com/watch?v=J4Wdy0Wc_xQ&feature=youtu.be