Modelado de Conjunto

El modelado de conjunto es un proceso en el que se crean múltiples modelos diversos para predecir un resultado, ya sea utilizando muchos algoritmos de modelado diferentes o utilizando conjuntos de datos de entrenamiento distintos. El modelo de conjunto luego agrega la predicción de cada modelo base y resulta en una predicción final para los datos no vistos.

La motivación para utilizar modelos de conjunto es reducir el error de generalización de la predicción. Siempre y cuando los modelos base sean diversos e independientes, el error de predicción del modelo disminuye cuando se utiliza el enfoque de conjunto.

El enfoque busca la sabiduría de las multitudes para hacer una predicción. Aunque el modelo de conjunto tiene múltiples modelos base dentro del modelo, actúa y funciona como un solo modelo. La mayoría de las soluciones prácticas de minería de datos utilizan técnicas de modelado de conjunto. El capítulo 4, Clasificación, cubre los enfoques de diferentes técnicas de modelado de conjunto y su implementación en detalle.

Al final de la etapa de modelado del proceso de minería de datos, hemos (1) analizado la pregunta comercial, (2) obtenido los datos relevantes para responder la pregunta, (3) elegido una técnica de minería de datos para responder la pregunta, (4) seleccionado un algoritmo de minería de datos y preparado los datos para adaptarse al algoritmo, (5) dividido los datos en conjuntos de entrenamiento y prueba, (6) construido un modelo generalizado a partir del conjunto de datos de entrenamiento y (7) validado el modelo con respecto al conjunto de datos de prueba. Este modelo ahora se puede utilizar para predecir la variable objetivo basándose en una variable de entrada de datos no vistos. Esto responde a la pregunta comercial sobre la predicción. Ahora, el modelo debe implementarse, por ejemplo, integrando el modelo en el proceso de aprobación de préstamos en producción de una empresa.

APLICACIÓN

La implementación o aplicación es la etapa en la que el modelo se vuelve listo para la producción o "en vivo". En las aplicaciones empresariales, los resultados de la minería de datos, ya sea el modelo para tareas predictivas o el marco de aprendizaje para reglas de asociación o agrupación, deben ser asimilados en el proceso comercial, generalmente en aplicaciones de software. La etapa de implementación del modelo conlleva algunas consideraciones clave: evaluar la preparación del modelo, la integración técnica, el tiempo de respuesta, el mantenimiento del modelo y la asimilación.

Preparación para la Producción

La parte de preparación para la producción de la implementación determina las cualidades críticas necesarias para el objetivo de despliegue. Consideremos dos casos de uso distintos: determinar si un consumidor califica para una cuenta de préstamo con una institución líder comercial y determinar la agrupación de clientes para una empresa.

El proceso de aprobación de crédito al consumidor es una iniciativa en tiempo real. Ya sea a través de un sitio web dirigido al consumidor o mediante una aplicación especializada para agentes de primera línea, las decisiones de crédito y los términos deben proporcionarse en tiempo real tan pronto como los clientes potenciales proporcionen información relevante. Se

considera una ventaja competitiva proporcionar una decisión rápida mientras se ofrecen resultados precisos en interés del cliente y la empresa. El modelo de toma de decisiones

En resumen, las principales ventajas de las Máquinas de Vectores de Soporte (SVM) son:

- Flexibilidad en la aplicación: las SVM se han aplicado en actividades que van desde el procesamiento de imágenes hasta la detección de fraudes y la minería de texto.

- Robustez: pequeños cambios en los datos no requieren una remodelación costosa.

- Resistencia al sobreajuste: generalmente, el límite entre las clases dentro de los conjuntos de datos puede describirse adecuadamente mediante solo unos pocos vectores de soporte.

Estas ventajas deben equilibrarse con los costos computacionales algo elevados de las SVM.

4.7 **APRENDICES DE CONJUNTO**

En la minería de datos supervisada, el objetivo es construir un modelo que explique la relación entre las entradas y las salidas. Este modelo se considera una hipótesis que puede mapear nuevos datos de entrada a una salida predicha. Dado un conjunto de entrenamiento, varias hipótesis pueden explicar la relación con diferentes niveles de precisión. Aunque es difícil encontrar la hipótesis exacta en un espacio de hipótesis infinito, el objetivo es encontrar la hipótesis que mejor explique la relación con el menor error.

Los métodos o aprendices de conjunto abordan este problema al emplear una variedad de modelos de predicción individuales y combinarlos para formar una hipótesis o modelo agregado. Esta técnica busca generar una hipótesis mejor al combinar múltiples hipótesis. Dado que una única hipótesis puede ser óptima localmente o ajustarse demasiado a un conjunto de entrenamiento específico, la combinación de múltiples modelos puede mejorar la precisión al lograr una solución de meta-hipótesis. Se ha demostrado que, en ciertas condiciones, este poder predictivo combinado es superior al poder predictivo de modelos individuales.

Debido a que diferentes métodos a menudo capturan distintas características del espacio de soluciones en cualquier modelo, los conjuntos de modelos han surgido como la técnica más importante para abordar muchos problemas prácticos de clasificación.

4.7.1 Sabiduría de la Multitud

Los modelos de conjunto tienen un conjunto de modelos base que aceptan las mismas entradas y predicen el resultado individualmente. Luego, las salidas de todos estos modelos base se combinan, generalmente mediante votación, para formar una salida de conjunto. Este enfoque es similar a la toma de decisiones por parte de un comité o una junta. El método de mejorar la precisión reuniendo las predicciones de varios modelos también se llama metaaprendizaje. Observamos esta metodología de toma de decisiones similar en tribunales superiores de justicia, juntas corporativas y varios comités en cuerpos legislativos. Si bien los miembros individuales tienen sesgos y opiniones, la idea aquí es que la toma de decisiones colectiva es mejor que la evaluación de un individuo. Los métodos de conjunto se utilizan para mejorar la tasa de error y superar el sesgo de modelado de modelos individuales. Pueden producir un aprendiz fuerte al combinar muchos aprendices débiles. La Figura 4.59 proporciona el marco de trabajo de los modelos de conjunto.

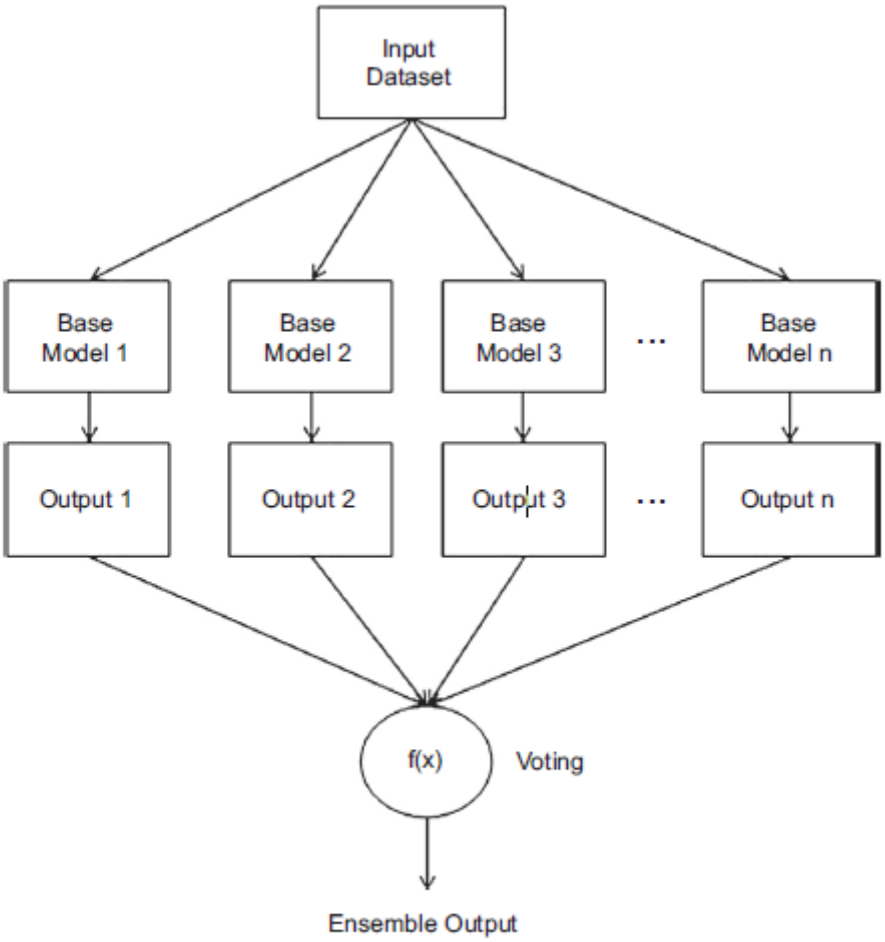

FIGURE 4.59
Ensemble model.

El paso final de agregar la predicción generalmente se realiza mediante votación. La clase predicha con más votos de los aprendices base es la salida del modelo de conjunto combinado. Los modelos base predicen el resultado con grados variables de precisión. Por lo tanto, también podemos ponderar el voto por la tasa de precisión de los modelos individuales, lo que hace que los modelos base con mayor precisión tengan una representación mayor en la agregación final que los modelos con una tasa de precisión menor (Dietterich, 2007).

4.7.2 Cómo Funciona

Tomemos un ejemplo de una sala de juntas corporativa hipotética con tres miembros del consejo. Supongamos que individualmente cada miembro del consejo toma decisiones incorrectas aproximadamente el 20% del tiempo. El consejo debe tomar una decisión de sí/no para una propuesta importante de proyecto. Si todos los miembros del consejo toman decisiones consistentemente unánimes cada vez, entonces la tasa de error del consejo en su conjunto es del 20%. Pero, si las decisiones de cada miembro del consejo son independientes y sus resultados no están correlacionados, el consejo comete un error solo cuando más de dos miembros del consejo cometen un error al mismo tiempo. El consejo comete un error solo cuando la mayoría de sus miembros cometen un error. Podemos calcular la tasa de error del consejo utilizando la distribución binomial.

En la distribución binomial, la probabilidad de k éxitos en n ensayos independientes, cada uno con una tasa de éxito de p, se da mediante una función de masa de probabilidad:

$$P(k) = \left(\frac{n}{k}\right) p^k (1-p)^{n-k}$$

$$P(\text{Board wrong}) = P(k \geq 2) = P(2 \text{ members wrong}) + P(3 \text{ members wrong})$$

$$\begin{aligned} P(\text{Board wrong}) &= \left(\frac{n}{3}\right) p^k (1-p)^{n-k} + \left(\frac{n}{2}\right) p^k (1-p)^{n-k} \\ &= \left(\frac{3}{3}\right) 0.2^3 (1-0.2)^0 + \left(\frac{3}{2}\right) 0.2^2 (1-0.2)^1 \\ &= 0.008 + 0.096 \\ &= 0.104 \\ &= 10.4\% \end{aligned}$$

En este ejemplo, ¡la tasa de error del consejo (10.4%) es menor que la tasa de error de los individuos (20%)! Por lo tanto, vemos el impacto de la toma de decisiones colectiva. Una fórmula genérica para calcular la tasa de error para el conjunto se da por:

$$P(\text{ensemble wrong}) = P(k \geq \text{round}(n/2)) = \sum_{k=0}^{n} \left(\frac{n}{k}\right) p^k (1-p)^{n-k}$$

Sin embargo, hay algunos criterios importantes que se deben tener en cuenta:

- Cada miembro del conjunto debe ser independiente.

- La tasa de error individual del modelo debe ser inferior al 50% para clasificadores binarios.

Si la tasa de error del clasificador base es superior al 50%, su capacidad de predicción es peor que el azar y, por lo tanto, no es un buen modelo para empezar. Lograr el primer criterio de independencia entre los clasificadores base es difícil. Sin embargo, hay algunas técnicas disponibles para hacer que los modelos base sean lo más diversos posible. En la analogía de la junta, tener una junta con miembros diversos e independientes tiene sentido estadístico. Por supuesto, todos deben tomar decisiones correctas más de la mitad del tiempo.

Cumplir con las condiciones para el modelado de conjuntos

Solo podremos aprovechar el poder de toma de decisiones combinada si los modelos base son buenos desde el principio. Si bien los meta aprendices pueden formar un aprendiz sólido a partir de varios aprendices débiles, esos aprendices débiles deben ser mejores que adivinar al azar. Dado que todos los modelos se desarrollan en función del mismo conjunto de entrenamiento, la condición de diversidad e independencia del modelo es difícil de lograr. Aunque no se puede lograr la completa independencia de los modelos base, podemos tomar medidas para promover la independencia cambiando los conjuntos de entrenamiento para cada modelo base, variando los atributos de entrada, construyendo diferentes clases de

técnicas y algoritmos de modelado y cambiando los parámetros de modelado para construir los modelos base. Para lograr la diversidad en los modelos base, podemos alterar las condiciones en las que se construye el modelo base.

Las condiciones más comúnmente utilizadas son:

- Algoritmos de modelo diferentes: Se puede utilizar el mismo conjunto de entrenamiento para construir diferentes clasificadores, como árboles de decisión utilizando múltiples algoritmos, Bayes ingenuo, vecinos más cercanos, redes neuronales artificiales, etc. Las características inherentes de estos modelos serán diferentes, lo que produce diferentes tasas de error y un conjunto diverso de modelos base.

- Parámetros dentro de los modelos: Cambiar los parámetros como la profundidad del árbol, la relación de ganancia y la división máxima para el modelo de árbol de decisión puede producir múltiples árboles de decisión. Se puede utilizar el mismo conjunto de entrenamiento para construir todos los modelos base.

- Cambio del conjunto de registros de entrenamiento: Dado que los datos de entrenamiento son el principal contribuyente al error en un modelo, cambiar el conjunto de entrenamiento para construir el modelo base es un método efectivo para construir múltiples modelos base independientes. Se puede dividir un conjunto de entrenamiento en varios conjuntos y cada conjunto se puede utilizar para construir un modelo base. Sin embargo, esta técnica requiere un conjunto de entrenamiento lo suficientemente grande y rara vez se utiliza. En su lugar, podemos muestrear datos de entrenamiento con reemplazo de un conjunto de datos y repetir el mismo proceso para otros modelos base.

- Cambio del conjunto de atributos: Similar a cambiar los datos de entrenamiento donde se utiliza una muestra de registros para la construcción de cada modelo base, podemos muestrear los atributos para cada modelo base. Esta técnica funciona si los datos de entrenamiento tienen un gran número de atributos.

En las próximas secciones, revisaremos enfoques específicos para construir modelos de conjunto basados en las técnicas mencionadas para promover la independencia entre los modelos base. Hay algunas limitaciones en el uso de modelos de conjunto. Si se utilizan algoritmos diferentes para los modelos base, imponen restricciones diferentes al tipo de datos de entrada que se pueden usar. Por lo tanto, podría crear un superset de restricciones a las entradas para un modelo de conjunto.

4.7.3 Cómo Implementar

En las herramientas de minería de datos, los operadores de modelado de conjunto se pueden encontrar en agrupamientos de aprendizaje meta o aprendizaje de conjunto. En RapidMiner, dado que el modelado de conjunto se utiliza en el contexto de la predicción, todos los operadores se encuentran en Modelado > Clasificación y Regresión > Modelado Meta. El proceso de construcción de modelos de conjunto es muy similar al de construir cualquier modelo de clasificación, como árboles de decisión o redes neuronales. Consulte los algoritmos de clasificación anteriores para conocer los pasos para desarrollar procesos y modelos de clasificación individuales en RapidMiner. Hay algunas opciones para elegir al implementar el modelado meta en RapidMiner. En las próximas páginas, revisaremos la implementación del modelado de conjunto con votación simple y algunas otras técnicas para hacer que los

modelos base sean independientes mediante la alteración de ejemplos para el conjunto de entrenamiento.

Conjunto por Votación

La implementación de un clasificador de conjunto comienza con la construcción de un proceso de clasificación base simple. Para este ejemplo, podemos construir un proceso de árbol de decisión con el conjunto de datos Iris, como se muestra en la Sección 4.1 Árboles de Decisión. El proceso estándar de árbol de decisión implica la recuperación de datos y un modelo de árbol de decisión, seguido de la aplicación del modelo a un conjunto de datos de prueba no visto obtenido del conjunto de datos Iris y el uso de un operador de evaluación de rendimiento. Para convertirlo en un modelo de conjunto, el operador Decision Tree debe ser reemplazado por el operador Vote de la carpeta de aprendizaje meta. Todos los demás operadores permanecerán iguales. El proceso de conjunto se verá similar al proceso mostrado en la Figura 4.60.

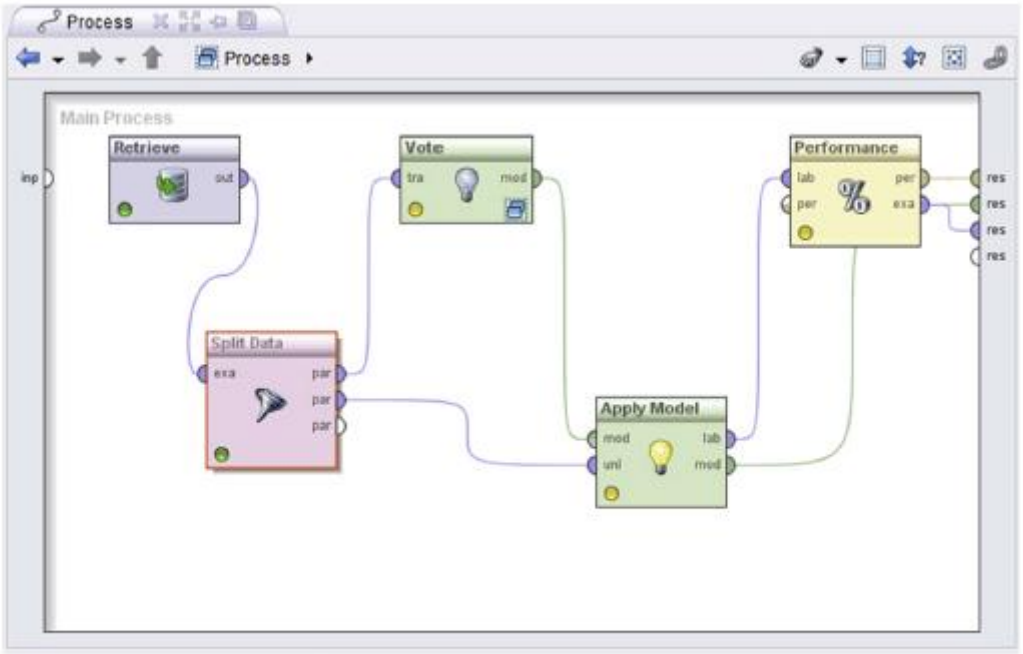

FIGURE 4.60
Data mining process using ensemble model.

El operador Vote es un aprendiz de conjunto que alberga múltiples modelos base dentro del subproceso interno (Mierswa et al., 2006). La salida del modelo del proceso de voto se comporta como cualquier otro modelo de clasificación y se puede aplicar en cualquier escenario donde se pueda usar un árbol de decisión. En la fase de aplicación del modelo, las clases predichas se suman entre todos los clasificadores base y la clase con el mayor número de votos es la clase predicha para el modelo de conjunto. Al hacer doble clic en el operador Vote de modelado meta anidado, podemos agregar múltiples modelos base de clasificación dentro del operador Vote. Todos estos modelos aceptan el mismo conjunto de entrenamiento y proporcionan un modelo base individual como salida. En este ejemplo, hemos agregado tres modelos: árbol de decisión, k-NN y Bayes ingenuo. La Figura 4.61 muestra el subproceso interno del operador Vote de modelado meta. El acto de sumar todas las predicciones de estos aprendices base y proporcionar la predicción mayoritaria es el trabajo del meta modelo, el operador de modelado Vote. Esta es la etapa de agregación en el modelado de conjunto y en

RapidMiner se llama un modelo de apilamiento. Un modelo de apilamiento está incorporado en el operador Vote y no es visible en la pantalla. El proceso de conjunto con el modelo meta Vote se puede guardar y ejecutar. Una vez que se ejecuta el proceso, el panel de salida del vector de rendimiento no es diferente de un vector de rendimiento normal. Dado que este proceso tiene un meta modelo, el panel del modelo en la ventana de resultados muestra nueva información, como se muestra en la Figura 4.62. La ventana del modelo muestra todos los modelos base individuales y un modelo de apilamiento, que es la combinación de todos los modelos base. El modelo meta Vote es fácil de usar en cualquier lugar donde se podría haber utilizado un modelo base individual de forma independiente. La limitación del modelo es que todos los aprendices base utilizan el mismo conjunto de datos de entrenamiento y diferentes modelos base imponen restricciones sobre qué tipos de datos pueden aceptar.

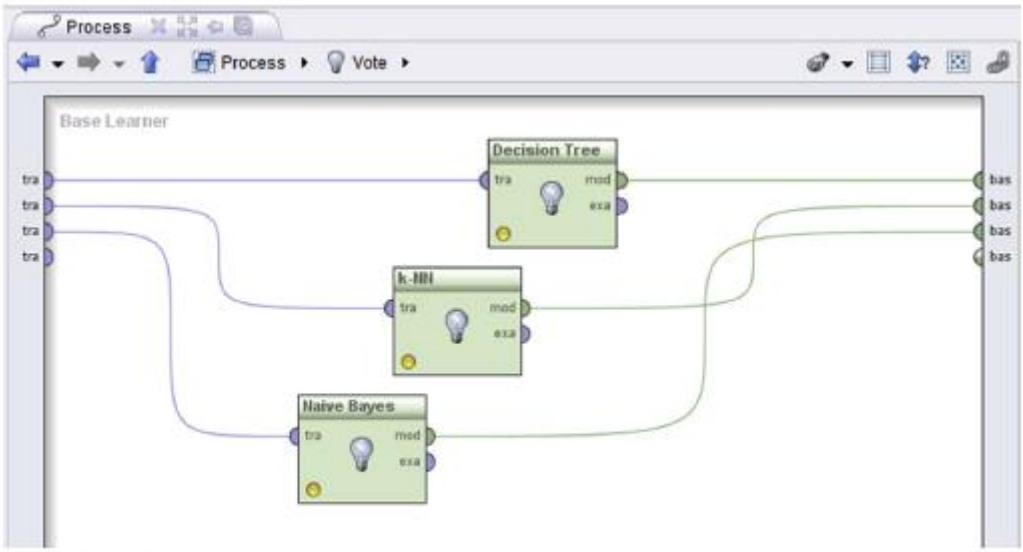

FIGURE 4.61
Subprocess inside the Vote operator.

Bootstrap Aggregating o Bagging

Bagging es una técnica donde se desarrollan modelos base al cambiar el conjunto de entrenamiento para cada modelo base. En un conjunto de entrenamiento T de n registros, se desarrollan m conjuntos de entrenamiento, cada uno con n registros, mediante muestreo con reemplazo. Cada conjunto de entrenamiento T1, T2, T3, ..., Tm tendrá el mismo número de registros n que el conjunto de entrenamiento original T. Debido a que se muestrean con reemplazo, pueden contener registros duplicados. Esto se llama bootstrap. Luego, cada conjunto de entrenamiento muestreado se utiliza para la preparación de un modelo base. A través del bootstrap, obtenemos un conjunto de m modelos base y la predicción de cada modelo se agrega para formar un modelo de conjunto. Esta combinación de bootstrap y agregación se llama bagging.

En promedio, cada conjunto de entrenamiento base Ti contiene aproximadamente el 63% de registros de entrenamiento únicos en comparación con el conjunto de entrenamiento original T. El muestreo con reemplazo de n registros contiene 1 - (1 - 1/n)^n registros únicos. Cuando n es lo suficientemente grande, obtenemos 1 - 1/e = 63.2% de registros únicos en promedio. El

resto de los datos contiene duplicados de datos ya muestreados. El proceso de bagging mejora la estabilidad de modelos inestables. Modelos inestables como árboles de decisión y redes neuronales son altamente dependientes incluso de pequeños cambios en los datos de entrenamiento. Debido a que un conjunto de bagging combina múltiples hipótesis del mismo conjunto de datos, la nueva hipótesis agregada ayuda a neutralizar estas variaciones en los datos de entrenamiento.

Implementación

El operador Bagging está disponible en la carpeta de aprendizaje meta: Modelado > Clasificación y Regresión > Modelado Meta > Bagging. Al igual que el operador meta Vote, Bagging es un operador anidado con un subproceso interno. A diferencia del proceso de voto, bagging tiene solo un modelo en el subproceso interno. Se generan varios modelos base al cambiar el conjunto de datos de entrenamiento. El operador Bagging tiene dos parámetros:

- Ratio de muestra: indica la fracción de registros utilizados para el entrenamiento.

- Iteraciones (m): número de modelos base que deben generarse.

La Figura 4.63 muestra el proceso RapidMiner para el operador Bagging. La Figura 4.64 muestra el subproceso interno para el operador Bagging con una especificación de modelo. Internamente, se generan varios modelos base según las iteraciones (m) configuradas en el parámetro Bagging. En este ejemplo, estamos utilizando un modelo de árbol de decisión para el subproceso interno.

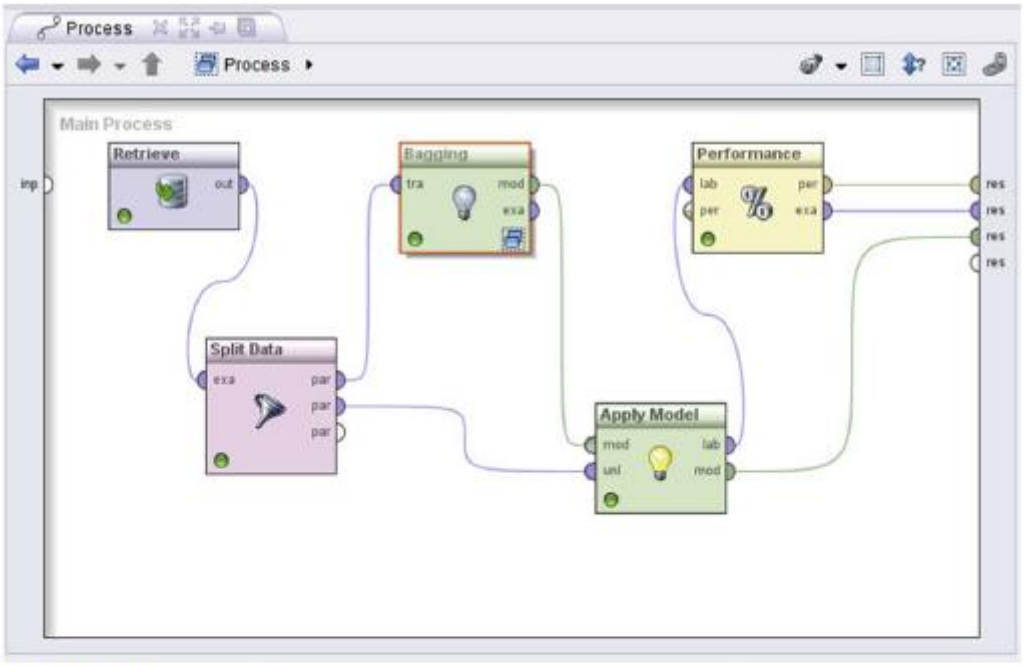

FIGURE 4.63
Ensemble process using bagging.

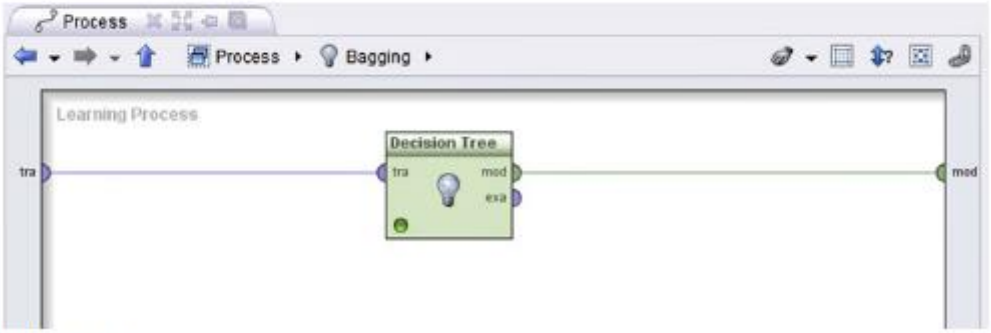

FIGURE 4.64
Bagging subprocess.

El proceso RapidMiner para bagging se puede guardar y ejecutar. Al igual que el modelo meta Vote, el modelo meta Bagging actúa como un solo modelo con varios modelos base en su interior. La ventana de resultados muestra el conjunto de ejemplos practicado, el vector de rendimiento y la descripción del modelo de bagging. En la ventana de resultados, podemos examinar todos los m (en este caso, 10) modelos que se desarrollan según las m iteraciones del conjunto de entrenamiento. Los resultados del modelo base se agregan mediante votación simple. Bagging es particularmente útil cuando hay anomalías en el conjunto de datos de entrenamiento que afectan significativamente al modelo individual. Bagging proporciona un marco útil donde se utiliza el mismo algoritmo de minería de datos para todos los aprendices base. Sin embargo, cada modelo base difiere porque los datos de entrenamiento utilizados por los aprendices base son diferentes. Dado que cada modelo base explora un espacio de soluciones diferente, el rendimiento del modelo de conjunto sería mejor que el de los modelos base. La Figura 4.65 muestra la salida del modelo del modelo meta Bagging con árboles de decisión constituyentes.

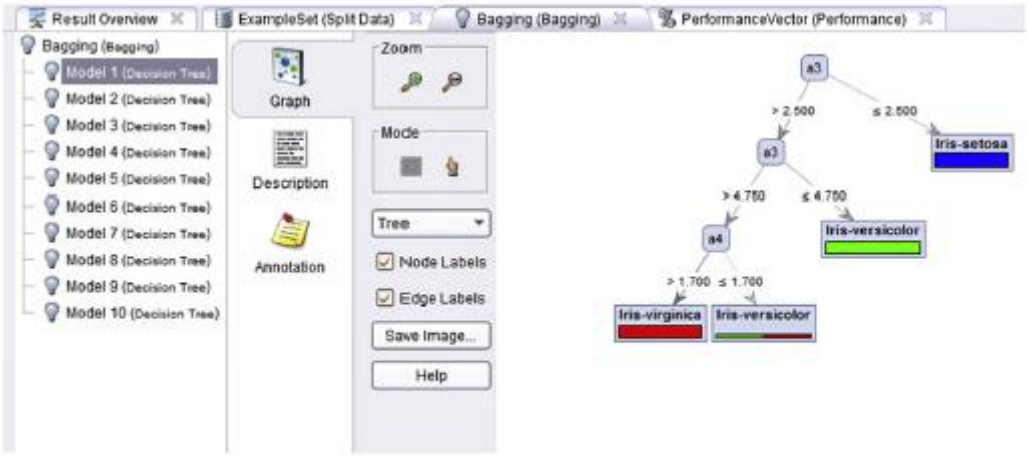

FIGURE 4.65
Output of bagging models.

Boosting

Boosting ofrece otro enfoque para construir un modelo de conjunto al manipular datos de entrenamiento de manera similar al bagging. Al igual que el bagging, proporciona una solución para combinar muchos aprendices débiles en un aprendiz fuerte, minimizando sesgo o varianza debido a registros de entrenamiento. A diferencia del bagging, boosting entrena los modelos base de manera secuencial uno por uno y asigna pesos a todos los registros de entrenamiento. El proceso de boosting se concentra en los registros de entrenamiento que son difíciles de clasificar y los sobreexpone en el conjunto de entrenamiento para la siguiente iteración.

El modelo de boosting se construye mediante un proceso iterativo y secuencial, donde se construye un modelo base y se prueba con todos los datos de entrenamiento, y según el resultado, se desarrolla el siguiente modelo base. Para empezar, todos los registros de entrenamiento tienen el mismo peso. El peso del registro se utiliza para la distribución de muestreo para la selección con reemplazo. Se selecciona una muestra de entrenamiento basada en los pesos y se utiliza para la construcción del modelo. Luego, el modelo se utiliza para probar con todo el conjunto de entrenamiento. Los registros clasificados incorrectamente se les asigna un peso más alto y los registros clasificados correctamente se les asigna un peso bajo, por lo que los registros difíciles de clasificar tienen una mayor propensión a ser seleccionados para la siguiente ronda. La muestra de entrenamiento para la próxima ronda probablemente estará llena de registros clasificados incorrectamente de la ronda anterior. Por lo tanto, el próximo modelo se enfocará en el espacio de datos difícil de clasificar.

Boosting asigna el peso para cada registro de entrenamiento y tiene que cambiar adaptativamente el peso según la dificultad de la clasificación. Esto da como resultado un conjunto de aprendices base especializados en clasificar registros fáciles y difíciles de clasificar. Al aplicar el modelo, todos los aprendices base se combinan mediante una simple agregación de votos.

AdaBoost

AdaBoost es una de las implementaciones más populares del enfoque de conjunto de boosting. Es adaptativo porque asigna pesos para los modelos base (α) basados en la precisión del modelo y cambia los pesos de los registros de entrenamiento (w) basándose en la precisión de la predicción. Aquí está el marco del modelo de conjunto AdaBoost con m clasificadores base y n registros de entrenamiento ((x1, y1), (x2, y2), ..., (xn, yn)). A continuación se presentan los pasos involucrados en AdaBoost:

1. A cada registro de entrenamiento se le asigna un peso uniforme \(w_i = 1/n\).

2. Se muestrean los registros de entrenamiento y se construye el primer clasificador base $b_k(x)$

3. La tasa de error para el clasificador base se puede calcular mediante la ecuación 4.23:

$$e_k = \sum_{k=1}^{n} w_i * I\left(b_k\left(x_i\right) \neq y_i\right)$$

donde \(I(x) = 1\) cuando la predicción es correcta y 0 cuando la predicción es incorrecta.

4. El peso del clasificador se puede calcular como $\alpha_k = \ln\left(1 - e_k\right)/e_k$. Si el modelo tiene una tasa de error baja, entonces el peso del clasificador es alto y viceversa.

5. A continuación, se actualizan los pesos de todos los registros de entrenamiento mediante

$w_{k+1}(i + 1) = w_k(i) * e^{(\alpha_k F(bk(xi) \neq yi))}$

donde $F(x) = -1$ si la predicción es correcta y \(F(x) = 1\) si la predicción es incorrecta.

Por lo tanto, el modelo AdaBoost actualiza los pesos en función de la predicción y la tasa de error del clasificador base. Si la tasa de error es superior al 50%, el peso del registro no se actualiza y se revierte en la siguiente ronda.

Modelo AdaBoost en RapidMiner

El operador AdaBoost está disponible en la carpeta de metaaprendizaje: Modelado > Clasificación y Regresión > Meta modelado > AdaBoost. El operador funciona de manera similar al Bagging y tiene un subprocesso interno. El número de iteraciones o modelos base es un parámetro configurable para el operador AdaBoost. La Figura 4.66 muestra el proceso de minería de datos AdaBoost. Este ejemplo utiliza el conjunto de datos Iris con el operador Split Data para generar conjuntos de datos de entrenamiento y prueba. La salida del modelo AdaBoost se aplica al conjunto de prueba y el rendimiento se evalúa mediante el operador Performance.

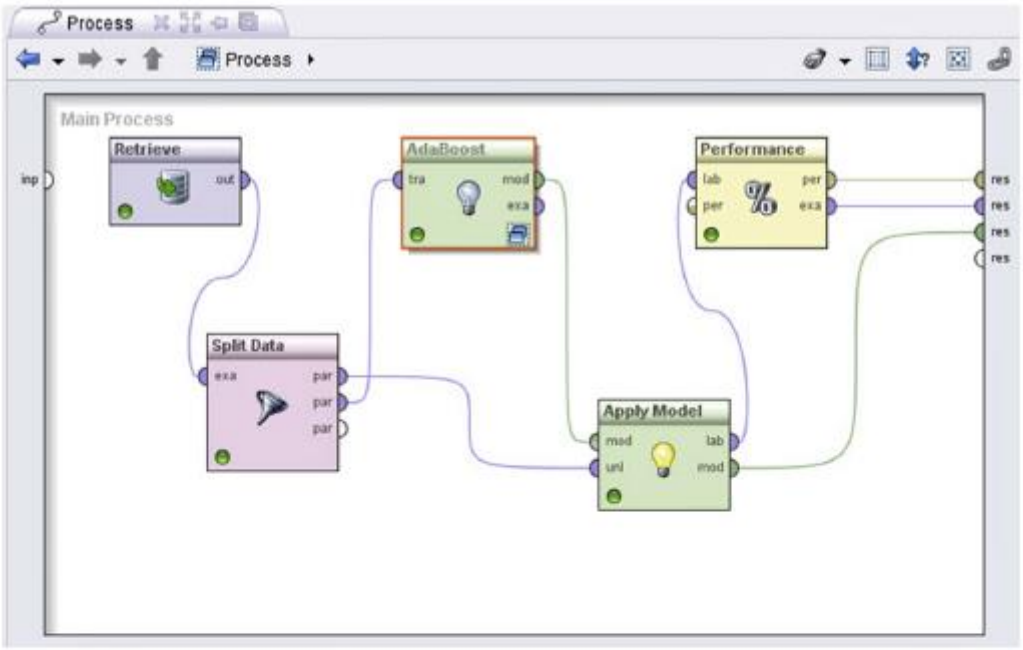

FIGURE 4.66
Data mining process using AdaBoost.

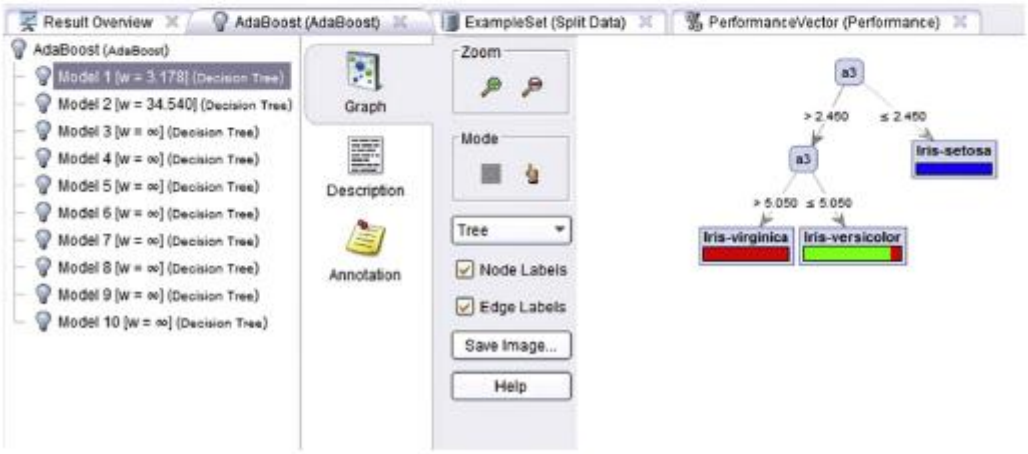

FIGURE 4.67
Output of AdaBoost model.

El número de iteraciones utilizado en AdaBoost es tres, como se especifica en el parámetro. En el proceso interno, se puede especificar el tipo de modelo. En este ejemplo, se utiliza el modelo de árbol de decisiones. El proceso completado en RapidMiner se guarda y ejecuta. La ventana de resultados muestra el modelo de conjunto de salida, los modelos base y los registros predichos. La ventana del modelo muestra los árboles de decisiones para los clasificadores base. La Figura 4.67 muestra la salida de resultados para el modelo AdaBoost.

Random Forest

Recuerde que en la técnica de bagging, para cada iteración, se considera una muestra de registros de entrenamiento para construir el modelo. La técnica de bosques aleatorios utiliza

un concepto similar al utilizado en el bagging. Al decidir sobre la división de cada nodo en un árbol de decisiones, el bosque aleatorio solo considera un subconjunto aleatorio de todos los atributos en el conjunto de entrenamiento. Para reducir el error de generalización, el algoritmo se aleatoriza en dos niveles, selección de registros de entrenamiento y selección de atributos, en el funcionamiento interno de cada clasificador base. El concepto de bosques aleatorios fue propuesto por primera vez por Leo Breiman y Adele Cutler (Breiman, 2001).

En general, el modelo funciona utilizando los siguientes pasos. Si hay n registros de entrenamiento con m atributos, y sea k el número de árboles en el bosque; entonces, para cada árbol:

1. Se selecciona una muestra aleatoria de n con reemplazo. Este paso es similar al bagging.

2. Se selecciona un número D, donde D << m. D determina la cantidad de atributos a considerar para la división del nodo.

3. Se inicia un árbol de decisiones. Para cada nodo, en lugar de considerar todos los m atributos para la mejor división, se considera un número aleatorio D de atributos. Este paso se repite para cada nodo.

4. Como en cualquier conjunto, cuanto mayor sea la diversidad de los árboles base, menor será el error del conjunto.

Una vez que se construyen todos los árboles en el bosque, para cada nuevo registro, todos los árboles predicen una clase y votan por la clase con pesos iguales. La clase más predicha por los árboles base es la predicción del bosque (Gashler et al., 2008).

Implementación

El operador Random Forest está disponible en Modelado > Clasificación y Regresión > Inducción de Árboles > Random Forest. Funciona de manera similar a otros modelos de conjunto donde el usuario debe especificar el número de árboles base a construir. Dado que el modelo base interno es siempre un árbol de decisiones, no hay una especificación explícita del subprocesso interno. Los modelos de conjunto Bagging o Boosting requieren una especificación explícita del subprocesso interno. Todos los parámetros específicos del árbol, como el tamaño de la hoja, la profundidad y el criterio de división, se pueden especificar en el operador Random Forest. El parámetro clave que especifica el número de árboles base es Número de Árboles. La Figura 4.68 muestra el proceso RapidMiner con el conjunto de datos Iris, el operador de modelado Random Forest y el operador Apply Model. Para este ejemplo, se especifica el número de árboles base como 10. El proceso se ve y funciona de manera similar a un clasificador de árbol de decisiones simple.

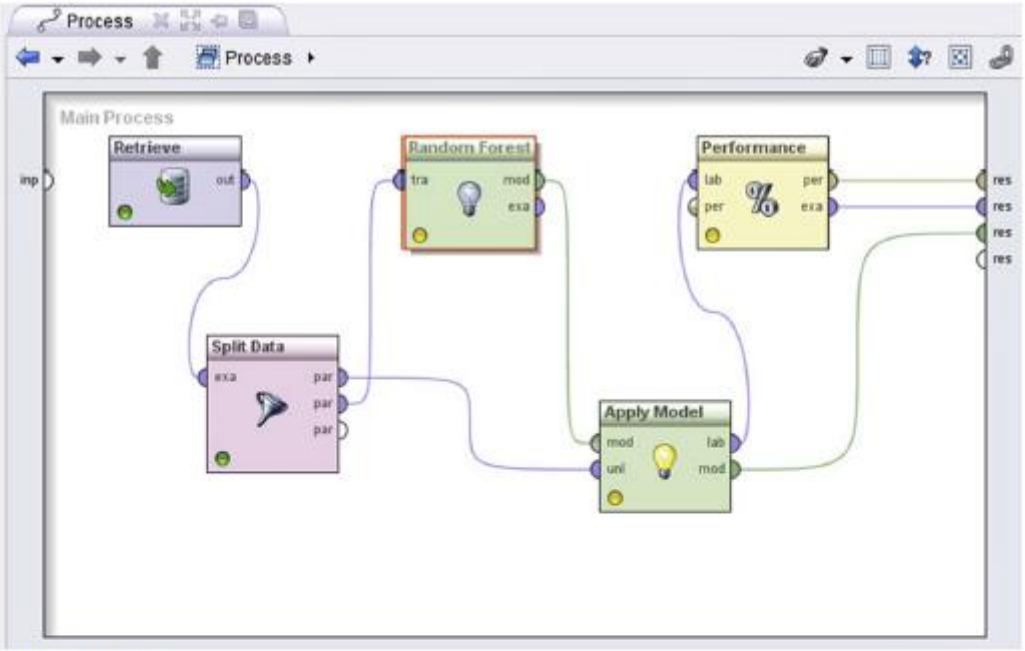

FIGURE 4.68
Data mining process using the Random Forest operator.

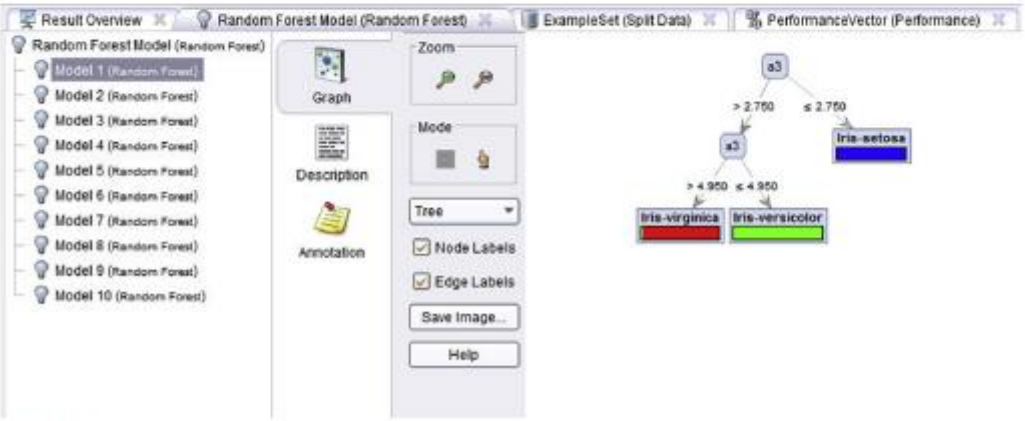

FIGURE 4.69
Output of Random Forest models.

Una vez que se ejecuta el proceso, la ventana de resultados muestra el modelo, la salida predicha y el vector de rendimiento. Similar a la salida de otros modelos de meta, el modelo Random Forest muestra los árboles para todos los clasificadores base. La Figura 4.69 muestra la salida del modelo para el operador Random Forest. Observe que los nodos son diferentes en cada árbol. Dado que la selección de atributos para cada nodo es aleatoria, cada árbol base es diferente. Por lo tanto, los modelos de Random Forest se esfuerzan por reducir el error de generalización del modelo de árbol de decisiones. Los modelos de Random Forest son muy útiles como modelos de conjunto de referencia para fines comparativos.