Análisis de datos

Edad:

* Promedio: 43.48 años.
* Desviación estándar: 14.91, lo que indica una dispersión moderada en los datos de edad.
* Mínimo y máximo: Las edades oscilan entre 18 y 69 años, cubriendo un buen rango de la vida adulta.

Ingresos:

* Promedio: 109,356, lo que indica un nivel de ingresos relativamente alto.
* Desviación estándar: 52,062, lo que muestra una dispersión significativa.
* Mínimo y máximo: Los ingresos van desde 20,005 hasta casi 200,000, lo que sugiere una amplia variedad de ingresos entre los individuos.

Gastos:

* Promedio: 52,401.
* Desviación estándar: 27,549, lo que muestra una variabilidad notable en los gastos.
* Mínimo y máximo: Los gastos oscilan entre 5,014 y 99,994, indicando una gran variabilidad en el consumo de los individuos.

Balance de Deuda:

* Promedio: 25,309.
* Desviación estándar: 14,147, lo que indica que hay una dispersión moderada en los niveles de deuda.
* Mínimo y máximo: Los valores van desde 1,001 hasta 49,995.

Puntaje de Crédito:

* Promedio: 575, lo cual está dentro de un rango típico para puntajes de crédito.
* Desviación estándar: 158, lo que sugiere una amplia dispersión en los puntajes de crédito.
* Mínimo y máximo: El puntaje de crédito varía de 300 a 849, que es el rango estándar en muchos países.

Número de Productos:

* Promedio: 5 productos.
* Desviación estándar: 2.57 productos.
* Mínimo y máximo: Los usuarios tienen entre 1 y 9 productos, lo cual parece razonable.

Historial de Incidentes:

* Promedio: 1.99 incidentes.
* Desviación estándar: 1.41.
* Mínimo y máximo: Varía entre 0 y 4 incidentes, lo que indica que algunos individuos no tienen historial de incidentes, mientras que otros pueden tener hasta 4.

Estado Financiero (Variable objetivo):

* Promedio: 0.503, lo que indica que aproximadamente el 50% de los individuos tienen una buena condición financiera (supongo que 1 representa una buena condición financiera y 0 una mala).
* Distribución equilibrada: La desviación estándar es 0.5, lo que sugiere una distribución casi equitativa entre las dos clases.

Random Forest

La configuración elegida sugiere que el modelo se ajusta sin restricciones en la profundidad y tiene flexibilidad para dividirse con facilidad. Esto puede generar modelos altamente complejos, lo que es efectivo cuando los datos tienen una estructura complicada, pero también puede correr el riesgo de sobreajuste si los datos tienen ruido o si hay muchos outliers.

Árbol de Decisión

Este modelo se ajusta bastante bien, ya que la profundidad máxima está limitada, lo que ayuda a controlar el sobreajuste. Sin embargo, dado que el valor de min\_samples\_split es 2, el árbol seguirá dividiéndose siempre que haya al menos 2 muestras, lo que permite que el árbol crezca bastante. En este caso, la limitación de la profundidad del árbol es una buena medida de regularización.

SVM (Support Vector Machine)

El modelo con C=1 y el kernel RBF es un ajuste estándar para muchas tareas de clasificación. El kernel RBF es útil cuando los datos no son linealmente separables, lo que probablemente es el caso aquí. El valor de C está equilibrado, lo que sugiere que el modelo está buscando evitar el sobreajuste al mismo tiempo que minimiza los errores.

K-Vecinos (KNN)

Este ajuste sugiere que el modelo está priorizando la cercanía de los puntos en la predicción. Tener n\_neighbors=7 significa que el modelo es relativamente flexible y considera un número moderado de vecinos, mientras que el uso de pesos basados en la distancia ayuda a enfatizar los puntos más cercanos, lo que puede mejorar la precisión en ciertos tipos de datos.

Conclusiones Generales

* Random Forest: El modelo es muy flexible y captura bien relaciones complejas, ya que no tiene restricciones en la profundidad de los árboles. Es probable que se comporte bien en situaciones donde los datos tienen muchas interacciones complejas entre las variables.
* Árbol de Decisión: El límite de profundidad a 15 niveles es adecuado para controlar el sobreajuste, lo que permite que el modelo sea lo suficientemente flexible pero no excesivamente complejo.
* SVM: El kernel RBF es una excelente elección cuando los datos no son linealmente separables. El valor de C=1 sugiere que el modelo tiene un equilibrio razonable entre la maximización del margen y la penalización de errores.
* K-Vecinos (KNN): El uso de 7 vecinos y la ponderación basada en la distancia indican que el modelo está haciendo predicciones en función de la cercanía relativa de los puntos más cercanos. Esto puede ser útil cuando las clases están distribuidas en patrones localizados.

**1. Random Forest**

* **Accuracy**: 50%
* **Precision**:
  + Clase 0: 0.50
  + Clase 1: 0.50
* **Recall**:
  + Clase 0: 0.51
  + Clase 1: 0.49
* **F1-Score**:
  + Clase 0: 0.51
  + Clase 1: 0.50

**2. Árbol de Decisión**

* **Accuracy**: 50%
* **Precision**:
  + Clase 0: 0.50
  + Clase 1: 0.50
* **Recall**:
  + Clase 0: 0.66
  + Clase 1: 0.34
* **F1-Score**:
  + Clase 0: 0.57
  + Clase 1: 0.41

**3. SVM (Support Vector Machine)**

* **Accuracy**: 50%
* **Precision**:
  + Clase 0: 0.50
  + Clase 1: 0.50
* **Recall**:
  + Clase 0: 0.48
  + Clase 1: 0.51
* **F1-Score**:
  + Clase 0: 0.49
  + Clase 1: 0.51

**4. K-Vecinos (KNN)**

* **Accuracy**: 49%
* **Precision**:
  + Clase 0: 0.49
  + Clase 1: 0.49
* **Recall**:
  + Clase 0: 0.48
  + Clase 1: 0.50
* **F1-Score**:
  + Clase 0: 0.49
  + Clase 1: 0.50
* Análisis Comparativo
* Al observar los resultados de los modelos, todos tienen una precisión cercana al 50%. Esto nos dice que ninguno está logrando un rendimiento notable en la clasificación de los datos. Este bajo rendimiento sugiere que los datos probablemente no son fáciles de separar o que pueden contener ruido que está afectando la capacidad de los modelos para clasificar correctamente.
* En cuanto al F1-score, el modelo de Random Forest destaca por ser el más equilibrado entre las dos clases, obteniendo un F1-score de 0.51 para la clase 0 y 0.50 para la clase 1. Esto indica que el modelo trata de manera uniforme ambas clases, lo cual es positivo si necesitamos que ambas categorías se manejen de manera justa. En contraste, el Árbol de Decisión muestra una diferencia notable entre las clases. Para la clase 0, tiene un F1-score de 0.57, que es bastante bueno, pero en la clase 1 su rendimiento cae a 0.41. Esto sugiere que este modelo favorece la clase 0, clasificando correctamente muchos más ejemplos de esa clase, pero fallando a menudo en la clase 1. Los modelos de SVM y KNN muestran resultados similares entre sí, con F1-scores ligeramente inferiores a los de Random Forest, pero también con un buen equilibrio entre las dos clases.
* Cuando miramos el recall, el Árbol de Decisión tiene el mejor rendimiento para la clase 0, con un 66%. Esto significa que es capaz de identificar correctamente la mayoría de los ejemplos de la clase 0. Sin embargo, su recall para la clase 1 es de solo 34%, lo que significa que muchos ejemplos de esta clase están siendo clasificados incorrectamente. En cambio, Random Forest tiene un recall más equilibrado, con 51% para la clase 0 y 49% para la clase 1, lo que lo convierte en un modelo más consistente si es importante manejar bien ambas clases.
* Al analizar las matrices de confusión, podemos ver que todos los modelos están teniendo dificultades para predecir correctamente ambas clases de manera equilibrada. El Árbol de Decisión tiende a favorecer más la clase 0, mientras que los modelos de Random Forest y SVM parecen ser más balanceados en cuanto a los errores que cometen entre las dos clases.
* En conclusión, el modelo de Random Forest parece ser el más equilibrado de todos en cuanto a las métricas clave como F1-score y recall. Si lo que necesitas es un modelo que maneje de manera justa ambas clases y que no se incline demasiado hacia una en particular, Random Forest es la opción más adecuada. Aunque el Árbol de Decisión tiene un mejor rendimiento para la clase 0, su pobre desempeño en la clase 1 lo hace menos confiable en situaciones donde ambas clases son importantes. Por otro lado, los modelos de SVM y KNN son opciones viables, aunque su rendimiento es un poco inferior al de Random Forest.