# Reporte: Modelo Random Forest para Clasificación de Diabetes

## Justificación del Algoritmo

El algoritmo Random Forest fue seleccionado por su capacidad para manejar conjuntos de datos con múltiples características, incluso si algunas no son altamente predictivas. Es robusto frente al sobreajuste y proporciona una métrica de importancia de características, lo cual es útil para entender qué variables son más relevantes en la predicción de diabetes.

## Descripción del Diseño del Modelo

El modelo fue diseñado siguiendo los pasos descritos a continuación:

1. Carga y preprocesamiento de datos: Se cargaron los datos del archivo 'diabetes.csv' y se inspeccionaron para identificar valores nulos y estadísticas básicas. No fue necesario realizar un tratamiento especial, ya que los datos estaban completos.

2. División del conjunto de datos: Los datos fueron divididos en conjuntos de entrenamiento (80%) y prueba (20%) de forma estratificada para garantizar una representación equitativa de ambas clases.

3. Entrenamiento del modelo base: Se entrenó un modelo Random Forest inicial con 100 estimadores y otros hiperparámetros predeterminados para establecer una línea base de precisión.

División del conjunto de datos

```
[4]: # Separar características y variable objetivo
     X = diabetes_dataset.drop(columns=['Outcome'])
     y = diabetes_dataset['Outcome']

     # Dividir en conjuntos de entrenamiento y prueba
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
```

Entrenamiento del modelo Random Forest

```
[5]: # Entrenar un modelo Random Forest
     clf = RandomForestClassifier(random_state=42, n_estimators=100)
     clf.fit(X_train, y_train)
```

```
[5]: RandomForestClassifier
     RandomForestClassifier(random_state=42)
```

## Explicación de la Optimización

Para mejorar la precisión del modelo, se optimizaron los hiperparámetros utilizando Random Search y Grid Search. Random Search permitió explorar un rango amplio de combinaciones de hiperparámetros, mientras que Grid Search afinó los resultados alrededor de los mejores valores obtenidos. Los hiperparámetros ajustados incluyeron el número de estimadores, la profundidad máxima del árbol, el número mínimo de muestras por nodo y el criterio de calidad de división ('gini' o 'entropy').

El mejor conjunto de hiperparámetros encontrado fue:
- Número de estimadores: 64
- Profundidad máxima: 15
- Mínimo de muestras para dividir un nodo: 8
- Mínimo de muestras en una hoja: 1
- Criterio de calidad: 'gini'

Con esta configuración, el modelo alcanzó una precisión del 78.34% en el conjunto de prueba.

```
# Ejecutar la búsqueda
random_search.fit(X_train, y_train)

# Mostrar los mejores parámetros
print("Mejores hiperparámetros:", random_search.best_params_)
print("Mejor precisión:", random_search.best_score_)
```

```
Fitting 3 folds for each of 50 candidates, totalling 150 fits
Mejores hiperparámetros: {'criterion': 'gini', 'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 8, 'n_estimators': 64}
Mejor precisión: 0.7834210106806951
```

```
# Ejecutar Grid Search
grid_search.fit(X_train, y_train)

# Mostrar los mejores parámetros y precisión
print("Mejores hiperparámetros:", grid_search.best_params_)
print("Mejor precisión:", grid_search.best_score_)
```

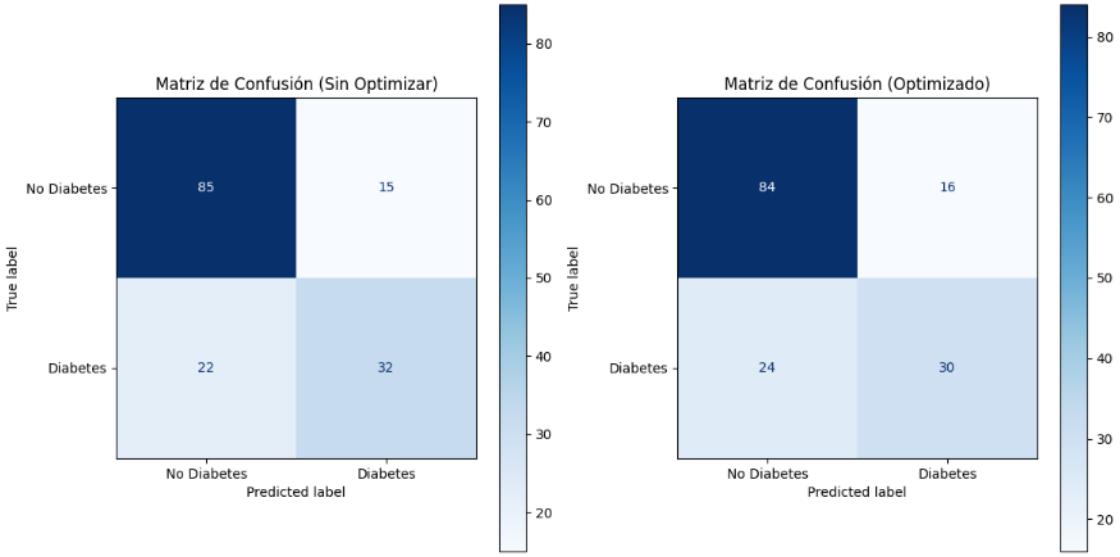
```
Fitting 3 folds for each of 162 candidates, totalling 486 fits
Mejores hiperparámetros: {'criterion': 'gini', 'max_depth': 15, 'min_samples_leaf': 1, 'min_samples_split': 8, 'n_estimators': 64}
Mejor precisión: 0.7834210106806951
```

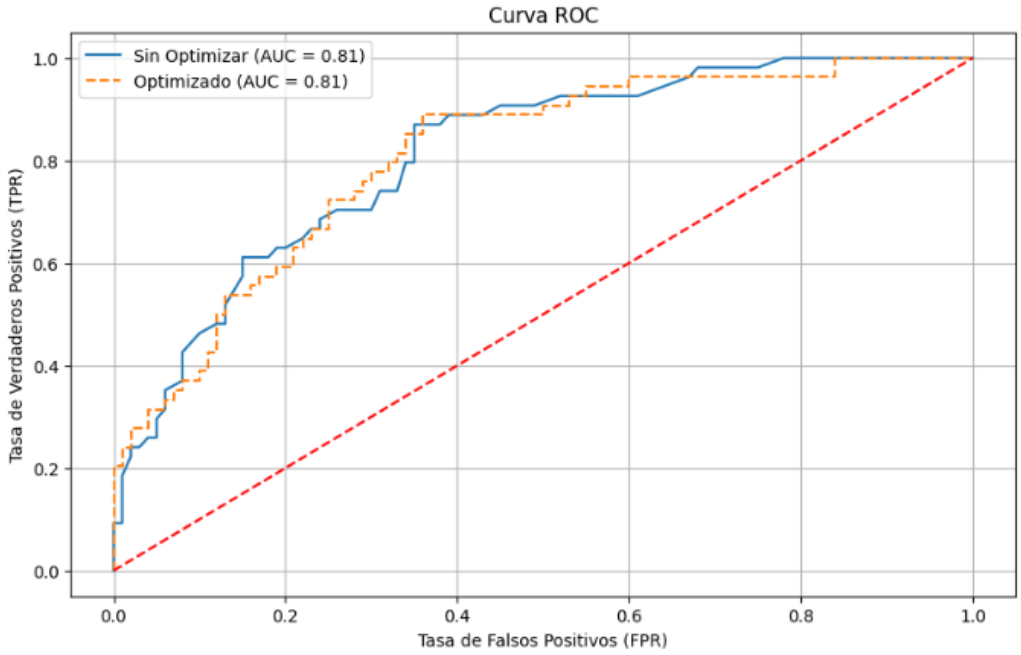
## Gráfica Personalizada e Interpretación de Resultados

En esta sección se deben incluir las gráficas generadas en el entorno Jupyter Notebook:

1. Matriz de Confusión: Permite visualizar los aciertos y errores del modelo en ambas clases (diabetes y no diabetes). Se deben incluir las matrices para el modelo base y el optimizado.

2. Curva ROC: Muestra la capacidad de discriminación del modelo. Se incluye la curva para el modelo base y el optimizado, indicando el área bajo la curva (AUC) para cada uno.


Matriz de Confusión (Sin Optimizar)
No Diabetes
Diabetes
True label
85
15
22
32
No Diabetes
Diabetes
Predicted label
80
70
60
50
40
30
20
Matriz de Confusión (Optimizado)
No Diabetes
Diabetes
True label
84
16
24
30
No Diabetes
Diabetes
Predicted label
80
70
60
50
40
30
20


Curva ROC
Sin Optimizar (AUC = 0.81)
Optimizado (AUC = 0.81)
Tasa de Verdaderos Positivos (TPR)
Tasa de Falsos Positivos (FPR)
0.0
0.2
0.4
0.6
0.8
1.0