

Optimización

Universidad Nacional del Altiplano - FINESI

Evaluación: Unidad II

Apellidos y Nombres: Nestor Ademir Ruelas Yana
Código: 230868

Pregunta 1

¿Cuáles son las principales diferencias entre la optimización de funciones convexas y no convexas, y cómo influyen en la elección de métodos?

Respuesta:

Las principales diferencias entre la optimización de funciones convexas y no convexas son:

Funciones Convexas:

- Cualquier mínimo local es también un mínimo global
- Garantizan convergencia a la solución óptima
- Los algoritmos de gradiente convergen de manera eficiente
- No presentan múltiples mínimos locales

Funciones No Convexas:

- Pueden tener múltiples mínimos locales
- No garantizan convergencia al mínimo global
- Requieren estrategias especiales para evitar mínimos locales
- Mayor complejidad computacional

Influencia en la elección de métodos:

- Para funciones convexas: métodos de gradiente, Newton, quasi-Newton
- Para funciones no convexas: algoritmos genéticos, simulated annealing, métodos estocásticos, múltiples inicializaciones

Pregunta 2

¿Cuándo resulta más efectivo usar variantes del descenso de gradiente y cuáles son los factores clave en su selección?

Respuesta:

Las variantes del descenso de gradiente son más efectivas en diferentes situaciones:

Gradient Descent Clásico:

- Datasets pequeños y funciones suaves
- Cuando se requiere convergencia determinística

Stochastic Gradient Descent (SGD):

- Datasets muy grandes
- Cuando se necesita convergencia rápida inicial

Mini-batch Gradient Descent:

- Balance entre eficiencia y estabilidad
- Aprovecha paralelización

Adam, RMSprop, AdaGrad:

- Funciones con diferentes escalas en las variables
- Redes neuronales profundas
- Espacios de parámetros de alta dimensionalidad

Factores clave en la selección:

- Tamaño del dataset
- Dimensionalidad del problema
- Recursos computacionales disponibles
- Requerimientos de precisión
- Características del paisaje de la función objetivo

Pregunta 3

¿Cómo se aplican los criterios de optimalidad en un método de optimización?

Respuesta:

Los criterios de optimalidad se aplican de la siguiente manera:

Condiciones de Primer Orden (Necesarias):

- Para un mínimo local: $\nabla f(x^*) = 0$
- Se verifica que el gradiente sea cero o suficientemente pequeño

Condiciones de Segundo Orden (Suficientes):

- La matriz Hessiana debe ser definida positiva: $\nabla^2 f(x^*) > 0$
- Garantiza que el punto crítico es un mínimo local

Aplicación práctica:

- Criterio de parada: $\|\nabla f(x)\| < \epsilon$
- Verificación de convergencia
- Validación de la solución encontrada
- Determinación de la naturaleza del punto crítico

En algoritmos iterativos:

- Se evalúa en cada iteración
- Define cuándo terminar el algoritmo
- Proporciona garantías teóricas de la solución

Pregunta 4

¿De qué manera facilitan las herramientas de optimización automática (p.ej., Optuna) la búsqueda de configuraciones óptimas en aprendizaje automático?

Respuesta:

Optuna y herramientas similares facilitan la optimización de hiperparámetros mediante:

Algoritmos Inteligentes:

- Tree-structured Parzen Estimator (TPE)
- Bayesian optimization
- Búsqueda más eficiente que grid search o random search

Características principales:

- Pruning automático de trials poco prometedores
- Paralelización nativa
- Manejo de espacios de búsqueda complejos
- Integración con frameworks populares (scikit-learn, PyTorch, TensorFlow)

Ventajas:

- Reducción significativa del tiempo de búsqueda
- Mejor exploración del espacio de hiperparámetros

- Visualización automática de resultados
- Persistencia de estudios y resultados
- Distribución de trials en múltiples procesos/máquinas

Impacto en ML:

- Automatización del proceso de tuning
- Mejores resultados con menos esfuerzo manual
- Reproducibilidad de experimentos
- Escalabilidad a problemas complejos

Pregunta 5

¿Cómo influye la selección de estrategias de regularización en el desempeño y la generalización de un modelo?

Respuesta:

La regularización influye significativamente en el desempeño y generalización:

Tipos de regularización y sus efectos:

L1 (Lasso):

- Promueve sparsidad en los parámetros
- Selección automática de características
- Útil cuando hay muchas características irrelevantes

L2 (Ridge):

- Reduce la magnitud de todos los parámetros
- Previene el overfitting suavizando el modelo
- Mantiene todas las características pero con menor peso

Elastic Net:

- Combina L1 y L2
- Balance entre selección de características y suavizado

Dropout (redes neuronales):

- Previene co-adaptación de neuronas
- Mejora la robustez del modelo

Impacto en el desempeño:

- Reduce overfitting
- Mejora la capacidad de generalización
- Puede reducir ligeramente el rendimiento en entrenamiento
- Aumenta la estabilidad del modelo
- La intensidad de regularización debe ser calibrada cuidadosamente

Pregunta 6

¿Cómo se pueden integrar rutinas de validación y corrección de código en Optuna para mejorar la calidad y eficacia del proceso de optimización de hiperparámetros?

Respuesta:

La integración de rutinas de validación y corrección en Optuna se puede realizar mediante:

Validación de parámetros:

- Callbacks personalizados para validar rangos de hiperparámetros
- Constraints para asegurar combinaciones válidas
- Verificación de tipos de datos y formatos

Manejo de errores:

- Try-catch blocks para capturar excepciones
- Logging detallado de errores y warnings
- Retry mechanisms para trials fallidos

Validación cruzada integrada:

- K-fold cross-validation dentro de cada trial
- Validación en conjunto de datos separado
- Métricas de validación como objetivo de optimización

Callbacks y pruning inteligente:

- Early stopping basado en métricas de validación
- Pruning de trials con bajo rendimiento
- Callbacks para logging y monitoreo

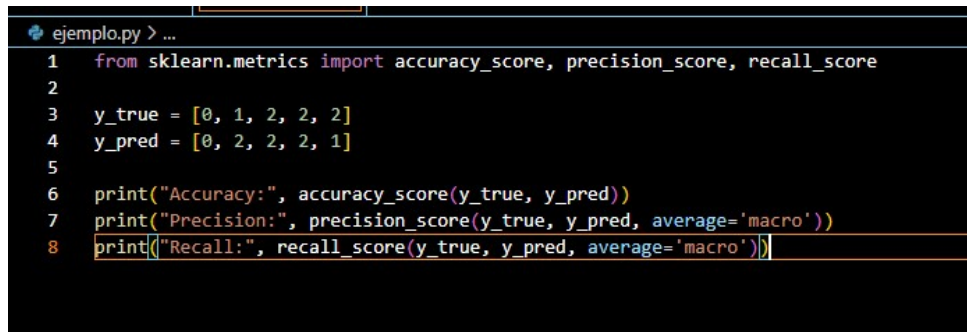
Mejores prácticas:

- Semillas fijas para reproducibilidad
- Validación de convergencia del modelo
- Checkpointing para recuperación de errores
- Monitoreo de recursos computacionales
- Testing automático de configuraciones antes del entrenamiento completo

Pregunta 7

¿Cómo podemos utilizar las métricas provistas por scikit-learn para evaluar un modelo de clasificación, y qué información adicional brindan además de la exactitud? Completa y Responde:

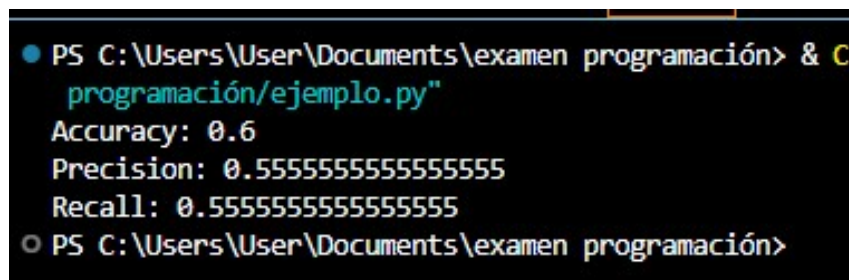
Código:



```
ejemplo.py > ...
1 from sklearn.metrics import accuracy_score, precision_score, recall_score
2
3 y_true = [0, 1, 2, 2, 2]
4 y_pred = [0, 2, 2, 2, 1]
5
6 print("Accuracy:", accuracy_score(y_true, y_pred))
7 print("Precision:", precision_score(y_true, y_pred, average='macro'))
8 print("Recall:", recall_score(y_true, y_pred, average='macro'))
```

Figura 1: Código de la Pregunta 7

Resultado:



```
PS C:\Users\User\Documents\examen programación> & C:\Users\User\Documents\examen programación\ejemplo.py"
Accuracy: 0.6
Precision: 0.5555555555555555
Recall: 0.5555555555555555
PS C:\Users\User\Documents\examen programación>
```

Figura 2: Resultado del Código

Respuesta:

Las métricas de scikit-learn proporcionan información valiosa más allá de la exactitud:

Accuracy (Exactitud):

- Proporción de predicciones correctas
- Para el ejemplo: $3/5 = 0.6$ (60 %)
- Limitación: no considera el desbalance de clases

Precision (Precisión):

- Proporción de predicciones positivas que son correctas
- $TP / (TP + FP)$
- Importante cuando el costo de falsos positivos es alto
- `Average='macro'`: promedio no ponderado entre clases

Recall (Sensibilidad):

- Proporción de casos positivos correctamente identificados
- $TP / (TP + FN)$
- Crítico cuando es importante no perder casos positivos

Información adicional que brindan:

- **Desempeño por clase:** Identifican clases problemáticas
- **Balance precision-recall:** Trade-off entre ambas métricas
- **Detección de sesgos:** Revelan preferencias del modelo
- **F1-score:** Media armónica de precision y recall
- **Matriz de confusión:** Distribución detallada de errores
- **Curvas ROC/AUC:** Rendimiento a diferentes umbrales

Otras métricas útiles:

- Classification report: resumen completo por clase
- Cohen's kappa: acuerdo corregido por casualidad
- Balanced accuracy: exactitud ajustada por desbalance
- Micro/macro/weighted averages: diferentes formas de promediar