INFORME FINAL

Raúl Andrés Pinilla Melo

Introducción

Aprender y analizar un tipo de aprendizaje en el que se proporcionan al algoritmo datos etiquetados, es decir, conjuntos de datos que contienen ejemplos de entradas y salidas deseadas. El objetivo es entrenar un modelo que pueda predecir las salidas correspondientes para nuevas entra

Algoritmos de aprendizaje supervisado

Su importancia radica en la capacidad de entrenar modelos precisos y predictivos para realizar tareas complejas y tomar decisiones basadas en datos.

Algunos ejemplos →

- Regresión lineal
- Regresión logística
- Máquinas de vectores de soporte (SVM)
- Árboles de decisión Bosques aleatorios Redes neuronales artificiales

Metodología

Realizar tareas de predicción y clasificación precisas en una amplia

Estos modelos son fundamentales en áreas como la detección de fraudes, el análisis de sentimientos, la recomendación de productos, la detección de spam, la visión por computadora, entreotros campos, lo que los convierte en herramientas esenciales para extraer conocimientos y tomar decisiones
informadas

Algoritmos

Los algoritmos son esenciales en la programación y la ciencia de la computación, ya que proporcionan un plan claro y sistemático.
Para entender un poco mas debes tener en cuenta los siguientes datos;

1

SECUENCIA DE PASOS

2

ENTRADAS Y SALIDAS

3

DETERMINISMO

Algoritmos

Los algoritmos son esenciales en la programación y la ciencia de la computación, ya que proporcionan un plan claro y sistemático.
Para entender un poco mas debes tener en cuenta los siguientes datos;

4

FINITUD

5

EFECTIVIDAD

¿QUÉ ES UN MÓDELO DE ENTRENAMIENTO?

El proceso de entrenamiento implica proporcionar al modelo
un conjunto de ejemplos (datos de entrada) junto con las respuestas correctas correspondientes (etiquetas o salidas esperadas). El modelo aprende patrones y relaciones a partir de estos ejemplos para
hacer predicciones sobre nuevos datos.

¿QUÉ ES UN MÓDELO DE PRUEBA?

Este
conjunto de prueba consta de datos no utilizados durante la fase de entrenamiento, y se emplea para medir la capacidad del modelo para generalizar y hacer predicciones precisas en nuevos datos.

Sobreajuste

Es un fenómeno en el aprendizaje automático en el cual un modelo se ajusta demasiado a los datos de entrenamiento, capturando patrones que no son representativos de la verdadera relación subyacente entre las variables.

Subajuste

Ocurre cuando un modelo es demasiado simple para capturar la complejidad de los datos de entrenamiento y, como resultado, no logra aprender la verdadera
relación subyacente entre las variables.

Validación cruzada

El propósito principal de la validación cruzada es obtener una estimación más precisa del rendimiento del modelo en datos no vistos. El procedimiento básico de validación cruzada implica dividir el conjunto de datos en varios subconjuntos y realizar múltiples evaluaciones del modelo, utilizando diferentes combinaciones de subconjuntos como conjuntos de entrenamiento y prueba.

Las métricas de rendimiento son medidas cuantitativas utilizadas para evaluar el desempeño de un modelo predictivo o clasificador en aprendizaje automático y estadísticas. Estas métricas proporcionan información sobre la calidad de las predicciones realizadas por el modelo en comparación con los valores reales. La elección de las métricas de rendimiento depende del tipo de tarea (clasificación, regresión, etc.) y de los objetivos específicos del problema.

Clasificación Binaria

$$\text{Exactitud} = \frac{\text{Número de predicciones correctas}}{\text{Número total de predicciones}}$$

Exactitud (Accuracy): Mide la proporción de predicciones correctas sobre el total de predicciones.

$$\text{Precisión} = \frac{\text{Verdaderos Positivos}}{\text{Verdaderos Positivos} + \text{Falsos Positivos}}$$

Precisión (Precision): Mide la proporción de verdaderos positivos sobre la suma de verdaderos

$$\text{Recall} = \frac{\text{Verdaderos Positivos}}{\text{Verdaderos Positivos} + \text{Falsos Negativos}}$$

Recall (Recuperación o Sensibilidad): Mide la proporción de verdaderos positivos sobre la suma

$$F1 = 2 \times \frac{\text{Precisión} \times \text{Recall}}{\text{Precisión} + \text{Recall}}$$

F1-Score: Es la media harmónica de precisión y recuperación.

Clasificación Multiclase

Se pueden utilizar extensiones de las métricas de clasificación binaria, como la macro-precisión, la macro-recuperación y la micro-precisión.

Regresión

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Error Cuadrático Medio (MSE): Mide la media de los cuadrados de las diferencias entre las predicciones y los valores reales.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Error Absoluto Medio (MAE): Mide la media de las diferencias absolutas entre las predicciones y los valores reales.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Coeficiente de Determinación : Indica la proporción de la varianza en la variable dependiente que es predecible a partir de las variables independientes.

Regresión

Objetivo Continuo:

A diferencia de la clasificación, donde el objetivo es predecir una etiqueta categórica, la regresión se utiliza cuando el objetivo es una variable continua. Por ejemplo, predecir el precio de una casa, la temperatura, o la cantidad de ventas

Modelo Matemático:

En este caso se busca encontrar la mejor función matemática que describa la relación entre las variables de entrada y la variable de salida. Este modelo puede ser lineal o no lineal, dependiendo de la naturaleza de los datos.

Parámetros Ajustables:

Estos suelen tener parámetros ajustables que se modifican durante el entrenamiento del modelo para minimizar la diferencia entre las predicciones del modelo y los valores reales observados en los datos de entrenamiento.

Evaluación del Rendimiento:

La calidad de un modelo de regresión se evalúa típicamente mediante métricas como el error cuadrático medio (MSE), el coeficiente de determinación (R-squared), o el error absoluto medio (MAE). Estas métricas cuantifican qué tan bien el modelo se ajusta a los datos observados.

Clasificación

OBJETIVO CATEGÓRICO:

La clasificación se utiliza cuando el resultado deseado es una etiqueta o categoría.

APRENDIZAJE DE PATRONES:

Durante el entrenamiento, el modelo analiza ejemplos de datos etiquetados para aprender patrones que puedan generalizarse a nuevas instancias

MODELO DE DECISIÓN:

Los modelos de clasificación buscan aprender una función que pueda tomar datos de entrada y asignarlos a categorías predefinidas.

EVALUACIÓN DEL RENDIMIENTO:

La calidad de un modelo de clasificación se evalúa mediante métricas como precisión, recall, F1-score, matriz de confusión y área bajo la curva ROC, entre otras. Estas métricas indican cómo de bien el modelo puede distinguir entre las clases.

La mayor de las dificultades en el presente trabajo fue orfanizar los datos antes de empezar con los modelos ya que eran demasiados datos y organizar fue lo que mas me costo, adicional a la hora de hacer el documento en rmakdown asi compilara en el código a veces arrojaba error por lo que para cumplir tuve que recurrir a cargar el código en imágenes.

```
datos_concardenados <- merge(DynamicCancerDriverKM::BRCA_normal,
                             DynamicCancerDriverKM::BRCA_PT,all = TRUE)
```

```
datos_concardenados <- datos_concardenados[, -c(1:3, 5:7)]
```

```r
knn_train
plot(knn_train)

# Realiza predicciones en los datos de prueba utilizando el modelo entrenado.
knn_predictor <- predict(knn_train, newdata = test.data)
knn_predictor

confusionMatrix(knn_predictor, test.data$sample_type)
```

```r
#CAMBIO LOS TITULOS EN FUNCION DE HGNC
titulos_resultantes <- AMCBGeneUtils::changeGeneId(Resultantes$PPI_genes,from="HGNC.symbol")
rownames(Resultantes) <- titulos_resultantes$HGNC.symbol

#Accedo a los titulos de cada variable genica

Cambio_titulos <- colnames(Variables_expresadas)
# Convertir los nombres de las variables en una matriz para unir con las resultantes
Cambio_titulos_matriz <- as.matrix(Cambio_titulos)

interseccion <- intersect(Resultantes$PPI_genes, Cambio_titulos_matriz)
interseccion <- as.matrix(interseccion)
```

```r
datos_concardenados <- merge(DynamicCancerDriverKM::BRCA_normal,
                             DynamicCancerDriverKM::BRCA_PT,all = TRUE)

#Resumo para entender un poco mejor los datos pero aun no es muy concluyente
summary(datos_concardenados)
#intento sacar la media pero tengo variables categoricas
median(datos_concardenados)

# Es por esto que elimino las columas que no me interesan y dejo solo genes y sample
datos_concardenados <- datos_concardenados[, -c(1:3, 5:7)]
```

```r
install.packages("caret")
library(caret)

# duplico primero mis datos
interseccion_datos <- Matriz_interseccion
# Ahora convierto sample_type en un factor
interseccion_datos$sample_type <- as.factor(interseccion_datos$sample_type)

set.seed(1)# Establezco una semilla para garantizar la reproducibilidad de los resultados
# cuando se generan números aleatorios. Cada que ejecute el mismo código, obtengo los
# mismos resultados.


#mediante muestreo aleatorio tomo un 70% para entrenamiento
sample.index <- sample(1:nrow(interseccion_datos),nrow(interseccion_datos)*0.7
                       ,replace = F)
```

```r
#mediante muestreo aleatorio tomo un 70% para entrenamiento
sample.index <- sample(1:nrow(interseccion_datos),nrow(interseccion_datos)*0.7
                       ,replace = F)

#creo una variable para guardar los titulos de las columnas menos el de sample_type
predictor <- colnames(interseccion_datos)[-1]

#creo el conjunto de datos con el 70% que tome del muestreo aleatorio
train.data <- interseccion_datos[sample.index,c(predictor,"sample_type"),drop=F]
#creo un conjuntos de datos con el 30% restante
test.data <- interseccion_datos[-sample.index,c(predictor,"sample_type"),drop=F]
```

```r
Grupo_in<- PPI %>% group_by(`Input-node Gene Symbol`) %>%
  summarise(NN = n()) %>%
  rename(`PPI_genes`=`Input-node Gene Symbol`)

Grupo_out <- PPI %>% group_by(`Output-node Gene Symbol`) %>%
  summarise(NN = n()) %>%
  rename(`PPI_genes`=`Output-node Gene Symbol`)

### Agrupo en un resultante mediante la columna PPI_genes
Resultantes <- bind_rows(Grupo_in, Grupo_out) %>%
  group_by(`PPI_genes`) %>%
  summarise(NN = sum(NN)) %>%
  arrange(desc(NN)) %>% #organizo en orden desc
  top_n(100, NN) #Saco el top 100 de las variables con mayor presencia
```

```r
medias_variables <- colMeans(Variables_expresadas[, 2:ncol(Variables_expresadas)]==1)
medias_variables

# Cargue PPI que contiene todos los genes
PPI <- DynamicCancerDriverKM::PPI
# renombre las variables para que concuerden respecto a HGNC.symbol para poder concardenarlas
titulos_ppi <- AMCBGeneUtils::changeGeneId(PPI$`Input-node Gene Symbol`,from="HGNC.symbol")

# nuevamente para acceder por columnas con colnames
colnames(Variables_expresadas)[c(2:ncol(Variables_expresadas))] <- titulos_ppi$HGNC.symbol
```

```r
ctrl <- trainControl(method="cv", p = 0.7)

# Entreno el modelo utilizando los datos de entrenamiento. Se indica que se realizará
# una validación cruzada y se aplicará procesamiento previo como centrado y escalado.
# Se busca la longitud óptima del vecino (k) en el rango de 1 a 25.

knn_train <- train( sample_type ~ .
                  , data = train.data
                  , method = "knn", trControl = ctrl
                  , preProcess = c("center","scale")
                  , tuneLength = 25)
```