PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
Facultad de Ingeniería
IIC2613 Inteligencia Artificial
2022-2

Tarea 3

Aspectos generales

Formato y plazo de entrega

El formato de entrega son dos archivos con extensión .ipynb, uno para cada sección de la tarea. Es importante que a la hora de entregar, tus notebooks se encuentren **con las celdas ejecutadas**. Las respuestas a las preguntas teóricas deben incluirse en celdas de texto adyacentes al código que utilices, de manera tal que el texto y las celdas de código sigan una estructura lógica que evidencie tu trabajo. El lugar de entrega es en el repositorio de la tarea, en la *branch* por defecto, hasta el **9 de noviembre a las 23:59**. Para crear tu repositorio, debes entrar en el enlace del anuncio de la tarea en Canvas. Por último, recuerda que los cupones de atraso son días **no hábiles** extra.

Integridad Académica

Este curso se adhiere al Código de Honor establecido por la universidad, el cual tienes el deber de conocer como estudiante. Todo el trabajo hecho en esta tarea debe ser **totalmente individual**. La idea es que te des el tiempo de aprender estos conceptos fundamentales, tanto para el curso, como para tu formación profesional. Las dudas se deben hacer exclusivamente al cuerpo docente a través de las *issues* en GitHub.

Por otra parte, sabemos que estás utilizando material hecho por otras personas, por lo que es importante reconocerlo de la forma apropiada. Todo lo que obtengas de internet debes citarlo de forma correcta (ya sea en APA, ICONTEC o IEEE). Cualquier falta a la ética y/o a la integridad académica será sancionada con la reprobación del curso y los antecedentes serán entregados a la Dirección de Pregrado.

Comentarios adicionales

Esta tarea está divida en dos partes. En la primera parte, **DCClínica** deberás usar los árboles de decisión para predecir la presencia o no de diabetes en los pacientes del *dataset* entregado. En la segunda parte, **DCCaracteres**, deberás utilizar SVM para clasificar imágenes de caracteres escritos a mano. **En ningún caso debes subir el dataset de alguna de las dos partes a tu repositorio**. Esto terminaría en un descuento significativo en tu nota.

Puedes encontrar los datasets en este link[1].

[1] https://drive.google.com/drive/u/2/folders/1p$_X iaXYjwxfT27t243WVsqUewABltitU$

1. DCClínica (3 pts.)

Te han contratado en la DCClínica como experto en el uso de inteligencia artificial para investigaciones médicas. Tu objetivo es buscar cuáles son las características que más puedan influir en que una persona desarrolle **diabetes**. Para esto se te entrega un *dataset* conteniendo valiosa información sobre pacientes que han visitado la clínica, con lo que te pones manos a la obra y comienzas a procesar los datos para luego entrenar un modelo clasificador que te permita hacer las predicciones.

Dataset

Los datos se encuentran en **data/db_diabetes.csv**. Donde **db_diabetes.csv** es el *dataset* y este posee las siguientes columnas, las cuales todas entregan información sobre el paciente:

- Diabetes_binary: diabetes, no_diabetes
- HighBP: 1 si tiene la presión alta, 0 en caso contrario
- HighChol: 1 si tiene el colesterol alto, 0 en caso contrario
- CholCheck: 1 si se ha hecho un chequeo de colesterol en los últimos 5 años, 0 en caso contrario
- BMI: índice de masa corporal
- Smoker: 1 si ha fumado al menos 100 cigarros en su vida, 0 en caso contrario
- Stroke: 1 si ha tenido un ACV (al menos que el paciente sepa), 0 en caso contrario
- HeartDiseaseorAttack: 1 si ha tenido enfermedad coronaria o infarto agudo de miocardio, 0 en caso contrario
- PhysActivity: 1 si ha hecho alguna actividad física en los últimos 30 días, sin incluir el trabajo, 0 en caso contrario
- Fruits: 1 si consume frutas una o más veces al día, 0 en caso contrario
- Veggies: 1 si consume vegetales una o más veces al día, 0 en caso contrario
- HvyAlcoholConsump: 1 si consume una cantidad alta de alcohol por semana (desde 14 tragos para hombres y desde 7 para mujeres), 0 en caso contrario
- NoDocbcCost: 1 si en los últimos 12 meses necesitó ir al doctor pero no pudo por costo, 0 en caso contrario
- GenHlth: escala del estado general de la salud, desde 5 a 1 (1 es excelente, 5 es malo)
- MentHlth: cantidad de días con problemas de salud mental en los últimos 30 días
- PhysHlth: cantidad de días con problemas de salud física en los últimos 30 días
- DiffWalk: 1 si tiene una seria dificultad para caminar o subir las escaleras, 0 en caso contrario
- Sex: male, female
- Age: escala de edad de 13 niveles (ver AGEG5YR codebook), donde 1 = 18-24, 9 = 60-64, 13 = 80 o más

- Education: nivel de educación basado en la escala EDUCA (ver EDUCA codebook), la escala va desde 1 a 6 (1 peor 6 mejor)
- Income: escala de ingreso (ver INCOME2 codebook), la escala va desde 1 a 8, donde 1 = 10.000 USD a 8 = 75.000 USD (anuales)
- HairColor: color de pelo (blonde, brown, black, red, white, grey)
- Height: altura en metros redondeada al primer decimal
- Weight: peso en kg

Actividad 1: Limpiando los datos (0.4 pts.)

Como es usual al trabajar con datos reales, hay que limpiar los datos antes de comenzar a utilizarlos. En esta parte deberás cargar el *dataset* y limpiarlo como estimes conveniente, para luego guardar la versión nueva. Es importante que expliques tu razonamiento logres justificar los cambios realizados.

Actividad 2: Visualizando y comprendiendo (0.8 pts.)

Ahora llegó el momento de entender los datos. Lo primero que debes hacer es observar bien qué datos tienes y si es que te das cuenta que tienes alguna columna que no aporta realmente, o es redundante (su info. ya esta contenida en otra columna), puedes eliminarla. Posteriormente debes graficar los datos de alguna forma, no hace falta que incluyas todas las columnas. Sin embargo, debes poder generar al menos 3 hipótesis a partir de los gráficos de por qué algunas características son reelevantes o no para predecir si una persona tiene diabetes.

Actividad 3: Pre-procesando los datos (0.4 pts.)

Lo último antes de entrenar al árbol de decisión es terminar de procesar todos los datos. Debes discretizar cualquier dato continuo, y representar con valores númericos las columnas que no lo sean.
Luego debes dividir los datos en set de entrenamiento y set de test, comentando **qué** proporción decidiste usar para el split y **por qué**.
Finalmente debes chequear que estos sets estén balanceados en cuanto a la proporción de *diabetes* y *no_diabetes* y qué podría ocurrir en caso contrario.

IMPORTANTE: Un mal uso de los sets de datos resultará en un descuento considerable ya que no obtendrás puntos por el rendimiento de tus modelos. Esto aplica tanto para esta parte como para DCCaracteres. Recuerda usar el set de validación para mejorar los hiperparámetros de tus modelos.

Actividad 4: Entrenando al árbol (0.4 pts.)

Usando la librería *sklearn* entrena un *DecisionTreeClassifier* que pueda predecir si una persona tendrá o no diabetes en base a los datos procesados de la actividad anterior.
Debes lograr que al árbol tenga un *Accuracy* mínimo de 67 %.
Adicionalmente, investiga y comenta sober las siguientes métricas, para luego calcularlas sobre el árbol:

- *recall_score*
- *precision_score*

¿Por qué son importantes?

Actividad 5: Ensamblaje simple (0.6 pts.)

Para esta actividad deberás construir un **Random Forest** personalizado que sirva para clasificar de la misma forma que en la actividad anterior. El ensamblaje debe tener 3 arboles y funcionar de la siguiente forma:

- Crea una clase que encapsule al ensamblaje.
- Debe tener un método *fit* el cual reciba un *dataset* X y otro arreglo y que contenga las etiquetas (diabetes, no_diabetes) para X.
- Lo que debe hacer *fit* es entrenar al primer árbol con todos los datos, luego entrenar al segundo solo con los datos en los que el primero clasifica mal, y finalmente entrenar al tercer árbol con los datos que el segundo clasifica mal.
- Luego debe tener un método *predict* que reciba un *dataset* X y similar al de *fit* y debe retornar un arreglo y, como el de *fit*, el cual contiene las predicciones asociadas a X.
- Usando el método *predict_proba* (se recomienda investigar brevemente) debes calcular la probabilidad que asigna cada uno de los árboles a las dos categorías (diabetes, no_diabetes), y usarlas para el siguiente cálculo: $P_1 * w_1 + P_2 * w_2 + P_3 * w_3 = P_f$, con $w_1 + w_2 + w_3 = 1$. Donde P_i es la probabilidad asignada por el árbol i y w_i son **hiperparámetros** que actúan como pesos.
- Finalmente debes tomar P_f como la probabilidad final asignada por el ensamble y usarla para asignar una categoría. Por ejemplo si P_f es 0.7, significando 70 % de probabilidad de que la categoría **diabetes**, se debe clasificar como **diabetes** (usar *round* puede ser buena idea).

Debes entrenar y evaluar al ensamblaje con los mismos datos que en la actividad anterior. Evalúa usando: *accuracy_score*, *recall_score* y *precision_score*. Compara los resultados con los de la actividad anterior y comenta.
Debes lograr un *accuracy_score* mínimo de 67 %.

Actividad 6: Random forest real (0.4 pts.)

Usando *RandomForestClassifier* de **sklearn**, entrénalo y evalúalo con los mismos datos que las dos actividades anteriores. Igual que antes, muestra los resultados en *accuracy_score*, *recall_score* y *precision_score*. Debes lograr un *accuracy_score* mínimo de 67 %.

Compara los 3 clasificadores y reflexiona sobre cual funciona mejor y por qué.

2. DCCaracteres (3 pts.)

Luego de permanecer tanto tiempo trabajando en la salud, te das cuenta de que un problema serio es que nadie puede entender la letra de las/os doctores. Inspirado por tu reciente éxito en la predicción de la diabétes, decides usar tus habilidades en *SVM* para crear un sistema capaz de interpretar la jeroglífica letra de estos profesionales.

Dataset

Los datos se encuentran en **data/caracteres.csv**. Donde **caracteres.csv** es el *dataset*, y este posee las siguientes columnas:

- *label*: Corresponde a la categoría de la imagen.
- Columnas del 1 al 784: Representan el valor de los pixeles de la imagen, que van desde 0 a 255.

Actividad 1: Análisis preliminar de los datos

- Verifica cuales son las categorías que existe, y el balance entra la cantidad de instancias de cada una.
- Indica las ventajas de que balancear el número de instancias entre categorías para el problema de clasificar caracteres escritos a mano. Considere en su respuesta el eventual desempeño en producción (cuando prediga datos reales) que tendría un modelo entrenado con datos no balanceados.
- Lee el siguiente artículo, que aborda el problema de conjuntos de datos imbalanceados. Explica brevemente las dos estrategias mencionadas para lidiar el desbalance en los conjuntos de datos.

Actividad 2: Preprocesamiento

- Separa el conjunto de datos entre variables dependientes e independientes.
- Realiza un escalamiento de los datos. Justifica la importancia de escalar los datos antes de entrenar modelos como *SVM*.
- Separa el conjunto de datos en subconjuntos de entrenamiento y testeo, justificando las proporciones elegidas para tal división. Verifica las proporciones de los datos entre categorías para cada subconjunto.

Actividad 3: Entrenamiento de modelo lineal

- Entrena un modelo lineal que identifique caracteres. Para eso, utiliza el módulo SVC de la librería *sklean.svm*. Visualiza los resultados de tu modelo mediane una matriz de confusión, e identifica las categorías más confundidas entre sí.
- Calcula el accuracy para el modelo lineal. Obtén además los puntajes de precission, recall y f1-score para cada uno de las categorías del modelo entrenado. Comente sobre el rendimiento del modelo en base a estos resultados y los de la matriz de confusión, y formula una hipótesis que pueda explicar los resultados obtenidos. En tu explicación, se espera que incorpores las métricas solicitadas y demuestres un manejo de su significado.

Actividad 4: Entrenamiento de modelo no lineal

- Investiga sobre el kernel RBF. Explica su funcionamiento en términos generales y las ventajas o desventajes que puede tener éste modelo en comparación con un kernel lineal.
- Entrena un modelo no lineal que identifique caracteres. Visualiza los resultados de tu modelo mediane una matriz de confusión, e identifica las categorías más confundidas entre sí.
- Calcula el accuracy para el modelo no lineal. Obtén además los puntajes de precission, recall y f1-score para cada uno de las categorías del modelo entrenado. Comente sobre el rendimiento del modelo en base a estos resultados y los de la matriz de confusión, y formula una hipótesis que pueda explicar los resultados obtenidos. En tu explicación, se espera que incorpores las métricas solicitadas y demuestres un manejo de su significado.

Actividad 5: Optimización del modelo

- Investiga los siguientes hiperparámetros que pueden ajustarse para el entrenamiento de SVM:
  - C
  - γ

  Detalla en tu respuesta qué hace cada uno de estos, en qué circunstancias pueden especificarse su valor, y cuales son los posibles efectos no deseados de utilizar valores extremos para ámbos hiperparámetros.
- Investiga sobre el método de *cross validation*. Explica brevemente en qué consiste, y cuál es su utilidad en la validación de un modelo entrenado.
- Investiga y explica brevemente en qué consiste la optimización de hiperparámetros en base a GridSearch, e indica el rol de *cross validation* en este método.
- Ejecuta una optimización en grid search en base al conjunto de hiperparámetros que consideres conveniente. Evalúa el modelo obtenido en el set de testeo, y grafica la matriz de confusión.
- Reporta el accuracy obtenido, y el precision, recall y f1-score para cada categoría. Compara los resultados con los obtenidos para los otros modelos.
- En base a los resultados obtenidos con el modelo optimizado, comenta brevemente sobre las principales virtudes y limitaciones del SVM en el contexto de visión por computador. En tu respuesta, procura aludir a las categorías con mejor y con peor desempeño bajo las métricas estudiadas. Por último, comenta sobre métodos adicionales a los vistos en esta tarea que podrían mejorar el rendimiento de SVM en el contexto del reconocimiento de caracteres.

Comentarios

Como podrás notar, buena parte de esta tarea involucra respuestas de desarrollo escrito, donde debes transparentar tu razonamiento y explicar las decisiones que tomas. Por este motivo, para que una respuesta se considere correcta, debes tener cuidado de fundamentar apropiadamente lo que digas, aludiendo a referencias confiables y a la documentación de las librerías que utilices. Por ejemplo, si te pedimos explicar un hiperparámetro en particular, o calcular una métrica de desempeño, se espera que demuestres un dominio general de lo que hace dicho hiperparámetro, o de la información que entrega la métrica solicitada. Para esto, es esperable que busques recursos (libros, artículos, documentación oficial, etc.) para fundamentar tus respuestas, donde debes indicar claramente la fuente de tal recurso.

Al mismo tiempo, es posible que al intentar ejecutar operaciones costosas (como entrenar un modelo sobre un conjunto de datos), la ejecución sea más lenta de lo que esperas. Esto es un escenario cotidiano al trabajar con modelos de aprendizaje de máquina, de modo que se espera que seas capaz de lidiar con tales situaciones, y que transparentes y fundamentes las decisiones que tomes en el proceso.

3. BONUS: DCChar-mpionship

Una vez hayas terminado con DCCaracteres tendrás la posibilidad de probar tus habilidades en Machine Learning participando del DCChar-mpionship. Este es un campeonato que consiste en clasificar la mayor cantidad de caracteres posible. Se otorgarán 2 décimas sólo por paticipar y el ganador se llevará 3 décimas.

Las bases del campeonato y los detalles de inscripción serán explicados más adelante. Por ahora lo importante es que te motives a encontrar el mejor modelo posible.