Aplicación web para predecir enfermedades renales crónicas utilizando aprendizaje automático

Universitat Oberta de Catalunya

Pablo Núñez de Arenas Martínez

Bioinformática Estadística y Aprendizaje Automático

Nombre Tutor/a de TF
Romina Astrid Rebrij

Profesor/a responsable de la asignatura
Carles Ventura Royo

13/06/2023

UNIVERSITAT DE BARCELONA

Universitat Oberta de Catalunya uoc.edu

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Aplicación web para predecir enfermedades renales crónicas utilizando aprendizaje automático*
Nombre del autor:	*Pablo Núñez de Arenas Martínez*
Nombre del consultor/a:	*Romina Astrid Rebrij*
Nombre del PRA:	*Carles Ventura Royo*
Fecha de entrega (mm/aaaa):	*06/2023*
Titulación o programa:	Máster en Bioinformática y Bioestadística
Área del Trabajo Final:	*Bioinformática Estadística y Aprendizaje Automático*
Idioma del trabajo:	*Castellano*
Palabras clave	*"Enfermedad renal crónica", "Aprendizaje Automático", "Desarrollo de aplicación Web"*

Resumen del Trabajo

En este trabajo se abordó la detección temprana de la enfermedad renal crónica (ERC) utilizando técnicas de aprendizaje automático (ML). La ERC conforma un problema de salud pública global cuyo diagnóstico precoz es crucial. A partir de un gran conjunto de datos clínicos se aplicaron distintas técnicas con el objetivo de identificar patrones relacionados con la presencia de la ERC. Inicialmente se trato de reducir la dimensionalidad del conjunto de datos utilizando técnicas como el análisis de correspondencia múltiple (MCA), el análisis de componentes principales (PCA) y la regresión lasso. Se determinó que las variables más importantes para la diagnosis de la enfermedad son la urea en sangre, los niveles de hemoglobina y el recuento de glóbulos rojos. Posteriormente, se evaluaron varios algoritmos, incluyendo k-vecinos más cercanos (KNN), máquinas de vectores de soporte (SVM), árbol de decisión, bosques aleatorios y regresión binomial. Finalmente, se desarrolló una aplicación web interactiva con *shinydashboard* utilizando el mejor de los modelos, el de KNN. A pesar de que la base de datos podría tener sesgos que pudieran afectar a la fiabilidad de los resultados se lograron cumplir los objetivos del trabajo al desarrollar la aplicación y obtener resultados prometedores en la detección temprana de la ERC.

Abstract

This study addressed the early detection of chronic kidney disease (CKD) using machine learning techniques. CKD poses a global public health problem, and early diagnosis is crucial. Various techniques were applied to a large dataset of clinical data to identify patterns related to the presence of CKD. Initially, efforts were made to reduce the dimensionality of the dataset using techniques such as multiple correspondence analysis (MCA), principal component analysis (PCA), and lasso regression. The most important variables for disease diagnosis were determined to be blood urea, hemoglobin levels, and red blood cell count. Subsequently, several algorithms were evaluated, including k-nearest neighbors (KNN), support vector machines (SVM), decision trees, random forests, and binomial regression. Finally, an interactive web application was developed using *shinydashboard*, employing the best-performing model, KNN. Despite potential biases in the database that could affect result reliability, the objectives of the study were achieved by developing the application and obtaining promising results in early CKD detection.

Índice:

Lista de figuras

Lista de Tablas:

1. Introducción

1.1 Contextualización y Justificación del trabajo

1.1.1 Descripción General

En este trabajo, se busca utilizar métodos de aprendizaje automático para diagnosticar la enfermedad renal crónica (CKD), conjunto de trastornos que afectan a la estructura y función renal [1]. Se utilizará una base de datos real que será preprocesada. A continuación, se aplicarán diversas técnicas de reducción de dimensionalidad para identificar los síntomas más representativos de la enfermedad. Luego, se evaluarán diferentes técnicas de aprendizaje automático para finalmente desarrollar una aplicación web que utilice el mejor modelo obtenido.

1.1.2 Justificación del TFM

Las enfermedades renales crónicas (CKD) representan un problema de salud pública mundial que debe abordarse desde sus primeras etapas [1]. Normalmente, las personas que padecen CKD son asintomáticas o experimentan síntomas inespecíficos como letargo, picazón o pérdida de apetito [2]. Sin embargo, cuando la enfermedad está en una etapa avanzada, el exceso de líquidos afecta a la mayoría de las funciones del cuerpo, lo que provoca complicaciones como enfermedades cardiovasculares, anemia o diabetes [3]. En esta fase de la enfermedad, el único tratamiento posible es la diálisis y el trasplante de riñón [3].

El diagnóstico de la CKD se puede realizar mediante pruebas de laboratorio rutinarias, y algunos tratamientos, especialmente en las etapas iniciales de la enfermedad, pueden prevenir su progresión [1]. La necesidad de encontrar una forma rápida y eficiente de diagnosticar una enfermedad que afecta a más de veinte millones de personas solo en los Estados Unidos [4] apunta al aprendizaje automático como una solución prometedora.

La temática del presente TFM tiene el potencial de ayudar a los médicos a personalizar los planes de tratamiento y a prevenir o tratar de manera más efectiva estas complicaciones, lo cual puede tener un impacto significativo en la salud y bienestar de los pacientes.

1.2 Objetivos

1.2.1 Objetivo general

Crear una aplicación capaz de diagnosticar o descartar CKD utilizando únicamente algunas de las variables presentes en la base de datos.

1.2.2 Objetivos específicos

1- Evaluar distintas técnicas de reducción de dimensionalidad para determinar las variables más importantes para el diagnóstico de CKD
2- Evaluar distintas técnicas de clasificación para determinar el mejor modelo predictivo
3- Optimizar el modelo seleccionado hasta lograr una precisión mínima del 85%.
4- Desarrollar una aplicación web interactiva en base al mejor modelo.

1.3 Enfoque

Partiremos de una base de datos obtenida en “UCI Machine Learning Repository” [5]. Se trata de un conjunto de datos reales obtenidos durante dos meses en distintos hospitales de la India por el Dr. P. Soundarapandian.M.D. Fueron tomadas 200 mediciones de 29 variables como la edad, la presión sanguínea o los niveles de azúcar en sangre, siendo la variable respuesta la ausencia o presencia de ERC.

Para comenzar este trabajo, se realizará una exhaustiva recopilación bibliográfica sobre las técnicas fundamentales de reducción de dimensionalidad y los algoritmos de aprendizaje automático más relevantes. Además, se investigarán las diferentes librerías de R necesarias para llevar a cabo los procesos de análisis, siendo nuestro apoyo principal el libro manual: 'Lantz, Brett. Machine Learning with R' [6]. Esta investigación bibliográfica, respaldada por el libro mencionado, proporcionará los conocimientos necesarios para desarrollar y aplicar eficazmente las técnicas y herramientas requeridas en el estudio.

Utilizaremos R debido a su continuo crecimiento en la aplicabilidad y disponibilidad de diversos métodos estadísticos disponibles de forma gratuita en la red, las cuales son constantemente revisadas y actualizadas por investigadores especializados en el campo.

Una vez completados los tres primeros objetivos específicos y habiendo puesto en práctica los conocimientos recopilados, se procederá a recopilar información adicional para el desarrollo de la aplicación web utilizando la librería "shinydashboard" de R [7]. Esta librería permite utilizar el lenguaje HTML para definir la estructura de la aplicación y los diversos elementos visuales, como encabezados, paneles, botones y tablas. Tras adquirir los conocimientos necesarios sobre el funcionamiento de esta librería, se avanzará en la consecución del cuarto objetivo específico.

1.4 Planificación temporal con hitos y temporización

1.4.1 Tareas

PEC 1: Plan de trabajo.
- Recopilación y análisis exploratorio de los datos.
- Definición de los modelos que serán aplicados.
- Desarrollo del plan de trabajo.

PEC 2: Desarrollo de trabajo. Fase 1.
- Exploración de las técnicas y herramientas necesarias.
- Preprocesamiento de la base de datos.
- Aplicación de las técnicas de reducción de dimensionalidad.
- Construcción, entrenamiento, validación y optimización de los modelos.
- Documentación del trabajo en la memoria.

PEC 3: Desarrollo de trabajo. Fase 2.
- Análisis de los resultados y selección del mejor modelo.
- Aprendizaje de desarrollo de aplicaciones empleando Shinydashboard.
- Definición del esquema de la aplicación.
- Desarrollo de la aplicación.
- Probar el funcionamiento de la aplicación.
- Documentar el trabajo en la memoria.

PEC 4: Cierre de la memoria.
- Optimizar esquema y presentación de la memoria.
- Organizar todos los códigos utilizados en un repositorio público.
- Ensayo de la presentación.
- Elaboración de la presentación.

PEC 5: Defensa pública.

1.4.2 Hitos

Hito 1: Entrega del plan de trabajo (15/3/2023)
Hito 2: Definición de las técnicas de reducción de dimensionalidad
Hito 3: Definición de los algoritmos de aprendizaje automático
Hito 4: Preprocesamiento y reducción de dimensionalidad
Hito 5: Construcción, entrenamiento y validación de los modelos
Hito 6: Entrega PEC 2 (24/4/2023)
Hito 7: Selección del modelo y definición del esquema
Hito 8: Desarrollo de la aplicación
Hito 9: Entrega PEC 3 (29/5/2023)
Hito 10: Primera entrega de la memoria y presentación (13/6/2023)
Hito 11: Entrega PEC 4 (20/6/2023)
Hito 12: Defensa pública (de 3 a 14 de julio)

1.4.3 Calendario

	PEC 1 (4 semanas)				PEC 2 (5 semanas)					PEC 3 (5 semanas)					PEC 4 (4 semanas)				PEC 5 (2 semanas)	
	1/3 a 20/3				21/3 a 24/4					25/4 a 29/5					30/5 a 20/6				3/7 a 14/7	
Recopilación y análisis de los datos																				
Definición de los modelos																				
Desarrollo del plan de trabajo																				
Exploración de técnicas y herramientas																				
Preprocesamiento de los datos																				
Reducción de la dimensionalidad																				
Construcción etc. De los modelos																				
Documentación en la memoria																				
Análisis y selección del modelo																				
Aprendizaje shinydashboard																				
Esquema de la APP																				
Desarrollo etc. de la APP																				
Documentación en la memoria																				
Primera entrega de la memoria																				
Entrega final de la memoria																				
Defensa TFM																				

1.4.4 Análisis de riesgos

Son varios los factores que han dificultado cumplir los plazos previstos:

- Selección de la base de datos: La base del datos resulto no ser del todo buena, lo que llevo a una mayor inversión de tiempo en su preprocesamiento. Especialmente, fue complicado abordar el desequilibrio en ciertas variables y determinar cuáles eran las más adecuadas.

- Falta de conocimiento / bibliografía: Se debió realizar una investigación más exhaustiva antes de comenzar con la reducción de dimensionalidad y la creación de modelos. Se empezó de manera precipitada, lo que generó resultados inesperados. Se invirtió más tiempo del previsto en comprender los errores, buscar soluciones y volver a realizar partes del trabajo, lo que causó un retraso de casi semanas en el desarrollo de la PEC 2.

- Dificultades respecto al software: Se encontraron errores en el uso de ciertas librerías y, principalmente, afecto la falta de experiencia en el manejo de shinydashboard a la hora de mantener un código limpio y ordenado a medida que este se volvía más extenso. También experimentamos problemas al exportar la aplicación y ejecutarla correctamente en un solo fragmento de código. Esto causó un retraso adicional debido a no guardar el código en formato UTF-8, lo que impedía que la aplicación funcionara adecuadamente.

1.5 Resultados esperados

1.5.1 **Plan de trabajo:**

Documento inicial en formato HTML/PDF obtenido a partir de Rmarkdown que contenga las pautas y tiempos estimados de ejecución de todas las tareas necesarias para el desarrollo del trabajo y cumplimiento de objetivos.

1.5.2 **Memoria:**

Documento en formato PDF que detalle toda la investigación, desarrollo, resultados y conclusiones obtenidas a lo largo del trabajo de fin de máster, así como el código implementado.

1.5.3 **Producto:**

Link de acceso a la aplicación web desarrollada en Shiny para el diagnóstico de CKD introduciendo únicamente las variables más importantes para su diagnóstico.

Acceso a un repositorio público que permita acceder al código desarrollado.

1.5.4 **Presentación virtual:**

Presentación en formato PPT para exponer el trabajo realizado.

1.6 Estructura del proyecto

- **PEC 1:** Plan de trabajo.
- **PEC 2:** Se pretende llevar a cabo las tareas correspondientes a los objetivos 1, 2 y 3.
  1- Evaluar distintas técnicas de reducción de dimensionalidad para determinar las variables más importantes para el diagnóstico de CKD.
  2- Evaluar distintas técnicas de clasificación para determinar el mejor modelo predictivo.
  3- Optimizar el modelo seleccionado hasta lograr una precisión mínima del 85%.
- **PEC 3:** En esta parte del proyecto se pretende cumplir el último de los objetivos propuestos.

  4- Desarrollar una aplicación web interactiva en base al mejor modelo.
- **PEC 4:** Elaboración de la memoria y de una presentación en la que se comenten los resultados y conclusiones obtenidas durante el TFM.
- **PEC 5:** Defensa del TFM.

2. Contexto

2.1 Estructura y funcionamiento del riñón

El riñón es un órgano esencial del sistema urinario, desempeñando un papel vital en la filtración de la sangre y la eliminación de desechos y líquidos en exceso a través de la formación de orina. Su estructura es compleja compuesta principalmente por tres partes [8].

- Capsula renal: Es una capa de tejido conectivo que rodea al riñón, proporcionándole protección y soporte estructural [8].
- Corteza renal: Es la capa exterior del riñón y contiene la mayoría de las nefronas, que son las unidades encargadas de la filtración y procesamiento de la sangre [8].
- Medula renal: Es la capa interior del riñón y se divide en regiones llamadas pirámides renales. La médula renal contiene estructuras como los túbulos renales y los vasos sanguíneos [8].

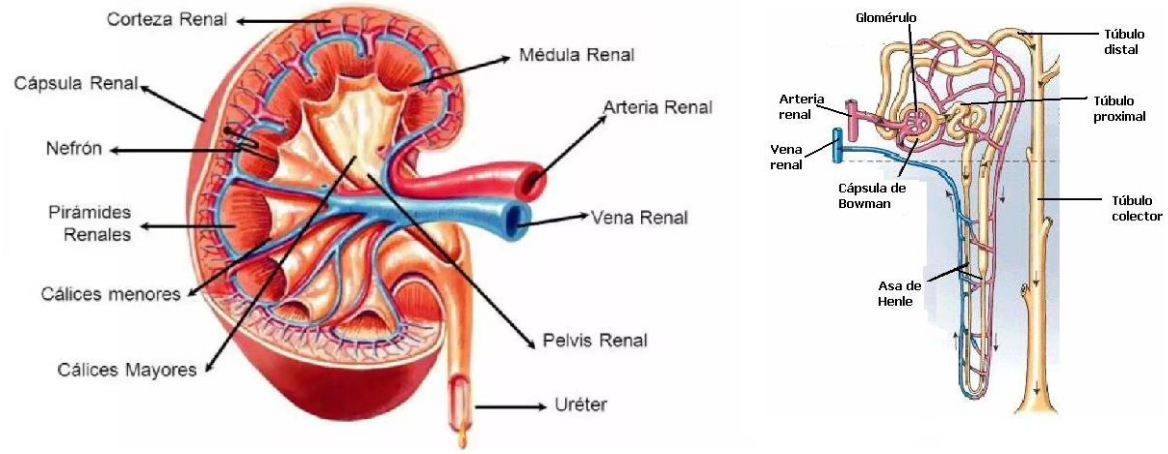


Fig. 6 Esquema del riñón y las nefrones. (https://grupocryo.com/partes-del-rinon/)

El proceso de filtración comienza en la arteria renal que transporta sangre rica en desechos y toxinas, la sangre se va ramificando por las arterias segmentarias, interlobares, aciformes, interlobulillares hasta llegar a las arteriolas eferentes que conectan con el glomérulo de las nefronas [9].

En las nefronas, se produce el proceso de filtración en sí. En el glomérulo, los líquidos y los solutos pequeños, como el agua, los electrolitos y los productos de desecho, son filtrados hacia la cápsula de Bowman. El filtrado recogido por la cápsula de Bowman pasa al túbulo contorneado proximal, donde se lleva a cabo una reabsorción selectiva de nutrientes y sustancias como glucosa, aminoácidos y sales, que son devueltos a la sangre a través de los capilares peritubulares. Los productos restantes son conducidos al asa de Henle, donde ocurre otra reabsorción en la zona descendente de agua y en la zona ascendente de sales [9].

Posteriormente, el filtrado llega al túbulo contorneado distal, donde se realiza otro proceso de reabsorción y se ajusta el pH y los niveles de electrolitos. El filtrado restante ingresa a los conductos colectores, donde se lleva a cabo la última reabsorción y se concentra la orina. Los conductos colectores de varias nefronas se fusionan para formar los conductos colectores principales, que a su vez forman la pelvis renal, una estructura en forma de embudo donde se acumula la orina. La orina fluye a través de los uréteres hasta llegar a la vejiga [9].

2.2 Enfermedad renal crónica

El concepto de enfermedad renal crónica (CKD) no se refiere a una sola enfermedad, sino a un conjunto heterogéneo de afecciones que pueden afectar la estructura y el funcionamiento de los riñones, destacando principalmente la diabetes y la hipertensión [1]. El diagnóstico de la CKD se basa en la tasa de filtración glomerular (GFR), que en una persona sana de 20 años es de 100-110 ml/min/1.73 m2, mientras que en una persona con CKD es inferior a 60 mL/min por 1.73 m2 [10].

La disminución en la GFR se asocia con una reducción en el número y tamaño de las nefronas, y este descenso es causado por varios factores, como el envejecimiento. Desde los 30 hasta los 75 años, se pierde en promedio entre 0.7-0.9 ml/min por 1.73 m2 de capacidad renal debido a una pérdida progresiva de nefronas, y esta pérdida se acentúa aún más a partir de los 75 años [11]. Otro factor es la obesidad, que provoca diversas alteraciones estructurales, como la hipertrofia glomerular y tubular, así como alteraciones hemodinámicas que aumentan la reabsorción de sodio y agua en la fase preglomerular [12].

Otras enfermedades, como la diabetes y la hipertensión, están asociadas con una disminución anual de la GFR de hasta 3 ml/min por 1.73 m2, relacionada con el deterioro vascular del riñón y el aumento de la presión arterial glomerular [12-13].

La mayoría de estos factores pueden ser tanto causa como consecuencia de la CKD, las enfermedades cardíacas y renales interactúan de manera compleja y bidireccional tanto en situaciones agudas como crónicas. Comparten mecanismos fisiológicos comunes, como la inflamación, la respuesta inmunológica, así como respuestas hormonales, cambios metabólicos y nutricionales, trastornos óseos y del equilibrio de minerales. Como resultado, la presencia de factores de riesgo como la diabetes y la hipertensión genera un bucle de retroalimentación positiva que agrava la pérdida de la tasa de filtración glomerular (GFR) y la aparición de otros factores. Por lo tanto, el diagnóstico temprano de la enfermedad renal crónica (CKD) es de vital importancia para romper este ciclo y prevenir complicaciones adicionales [11-12].

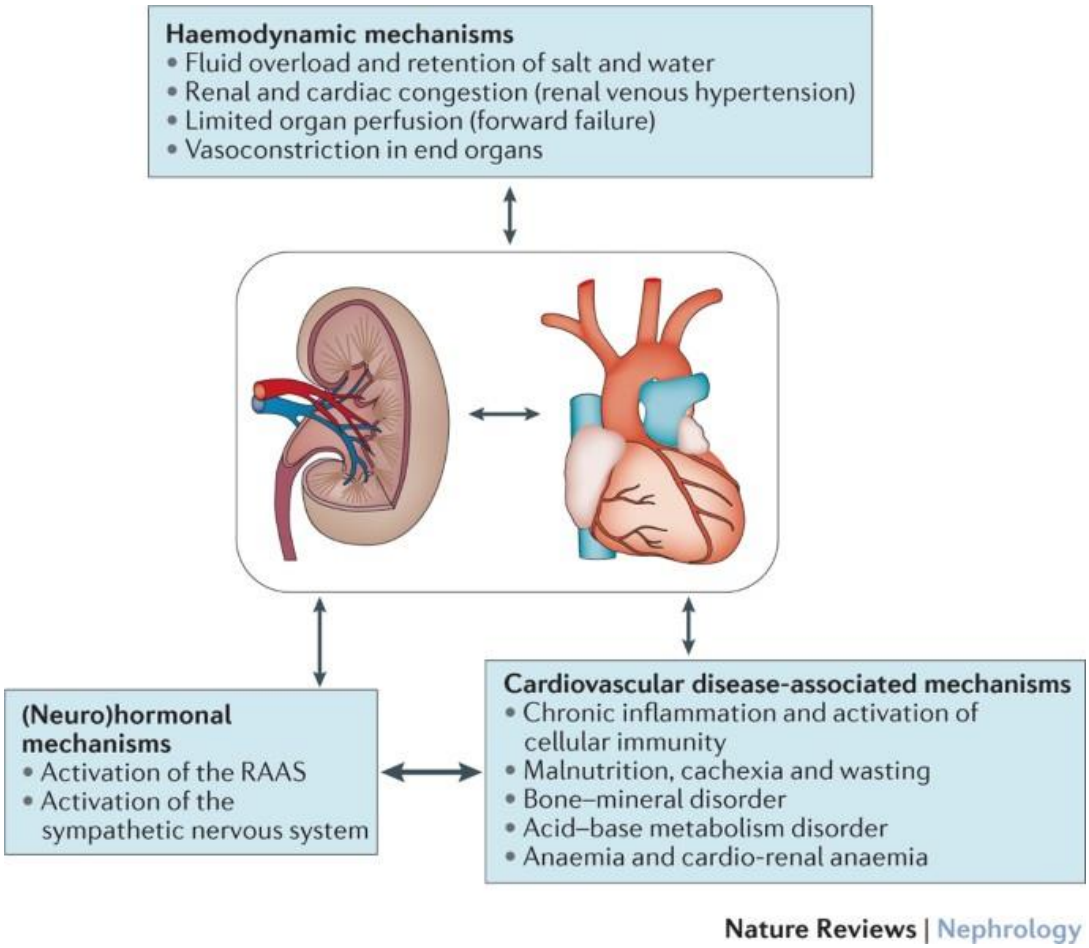


Fig. 7. Esquema retroalimentación entre riñón y corazón. [14].

Padecer CKD aumenta el riesgo de muerte prematura asociada a enfermedades cardiacas, anemia, niveles bajos de calcio, altos de potasio, pérdida de apetito, exceso de líquidos, infección y en casos extremos, enfermedad renal en etapa terminal (ESKD) cuya única solución es el trasplante [11].

La enfermedad en sus etapas iniciales suele ser asintomática, lo que lleva a que 9 de cada 10 personas que tienen enfermedad renal crónica (CKD) desconozcan su condición. De ese grupo, aproximadamente la mitad debería recibir tratamiento de diálisis. En los Estados Unidos, se estima que 2 de cada 1000 personas están recibiendo tratamiento para ESKD. Por desgracia, diariamente fallecen 240 personas en diálisis [11]. Esta enfermedad solo en los EE. UU cuesta 114.000 millones al año [15].

Por tanto, la detección temprana de la enfermedad renal permite implementar tratamientos y estrategias que pueden influir tanto en la progresión de la enfermedad renal como en la salud cardiovascular. Además, ayuda a evitar medicamentos y situaciones que pueden causar un empeoramiento de la función renal, lo que resulta en un gran ahorro de costos y recursos [16].

2.3 Diagnósticos usando aprendizaje automático

El aprendizaje automático (ML), es una rama de la inteligencia artificial que se enfoca en el desarrollo de algoritmos y técnicas que permiten reconocer patrones dentro de un conjunto de datos que le permitan identificar regularidades, hacer predicciones o tomar decisiones [6]. En el campo de la medicina, el ML ya se ha utilizado para detectar diversas enfermedades como enfermedades cardiacas, diabetes, retinopatía y cáncer entre otras [17].

El uso del ML en el diagnóstico médico ha demostrado ser prometedor debido a su capacidad para analizar grandes volúmenes de datos y encontrar patrones sutiles que pueden pasar desapercibidos para los médicos. Estos algoritmos pueden aprovechar características específicas de los datos clínicos, como resultados de pruebas de laboratorio, imágenes o historiales médicos, para generar modelos predictivos y proporcionar recomendaciones precisas para el diagnóstico y tratamiento de enfermedades [18]. Esta capacidad del aprendizaje automático para procesar y analizar datos complejos ofrece nuevas perspectivas en la medicina, permitiendo una atención médica más personalizada y mejorando los resultados para los pacientes.

3. Materiales y Métodos

3.1 Materiales

3.1.1 Base de datos

La base de datos parte de un conjunto de datos reales tomados durante dos meses en distintos hospitales de la India por el Dr. P. Soundarapandian.M.D. Los datos incluyen 200 mediciones de 29 variables incluyendo: clase (class), presión sanguínea (bp.diastolic), sistólica (bp.limit), gravedad específica (sg), albumina (al), glóbulos rojos (rbc), azúcar (su), células de pus (pc), aglomeraciones de pus (pcc), bacteria (ba), glucosa en sangre (bgr), urea en sangre (bu), sodio (sod), creatina sérica (sc), potasio (pot), hemoglobina (hemo), volumen celular alto (pcv), recuento glóbulos blanco (wbcc), recuento de glóbulos rojos (rbcc), hipertensión (htn), diabetes melitus (dm), enfermedad coronaria (cad), apetito (appet), anemia pedal (pe), anemia (ane), tasa de filtrado glomerular (grf). Las variables que supuestamente eran numéricas como sg, su, bgt etc. Englobaban rangos de valores por lo que recodificamos los valores para tratar todas las variables como categóricas, se puede comprobar los valores iniciales a los que cada nivel de cada variable hace referencia en la primera tabla de la sección de anexos (Tabla 1).

Se eliminaron algunas variables debido a que no estaban explicadas en el repositorio de datos o porque carecían de sentido:

Las variables "bp.diastolic" y "bp.limit" se eliminaron. En el repositorio solo se encontraba la variable "presión sanguínea" (bp), y no se especificaba a qué valor correspondía cada tipo de presión. Además, la variable "bp.limit" tenía 3 niveles sin una claridad sobre su significado.

La variable "Class" se eliminó. Esta variable era la respuesta, pero se decidió borrarla porque la variable "affected" también actúa como respuesta en una forma binaria, lo cual es más conveniente para construir los algoritmos. También se eliminó “stage” ya que también actuaba como respuesta dividiendo a los enfermos según la etapa de la enfermedad en la que se encontraban.

GRF hace referencia al grado de filtración glomerular. Según Websteret al. Las guías internacionales definen una ERC cuando el GRF es inferior a 60 ml/min por 1.73 m2. Sin embargo, en la base de datos aparecen como ERC pacientes con filtraciones

mayores a 60 ml y también pacientes sanos con filtraciones menores a 60. Es importante destacar que esta interpretación se realizó debido a la falta de explicación en el repositorio de datos.

3.1.2 Software de análisis

Todos los algoritmos se realizaron en R (versión 4.1.2) y los paquetes utilizados fueron ROSE (0.0.4), forcats (1.0.0), ggplot2 (3.4.0), gridExtra (2.3), FactoMineR (2.6), Factoshiny (2.4), factoextra (1.0.7), glmnet (4.1.6), fastDummies (1.6.3), car (3.1.1), vcd (1.4.11), caret (6.0.93), dplyr (1.0.10), rpart (4.1.19), randomForest (4.7.1.1), shiny (1.7.4) y shinydashboard (0.7.2).

3.2 Métodos

3.2.1 Preprocesamiento de los datos

Las variables se recodificaron para facilitar su procesamiento por el ordenador. A las variables binarias se les asignaron valores de 0 o 1, y a las variables categóricas con más de dos niveles se les asignaron valores desde 0 hasta (n-1) niveles. Por ejemplo, la variable gravedad específica (sg) tenía 5 niveles. El primer nivel de la variable se asignó el valor 0, y así sucesivamente hasta el quinto nivel, que se asignó el valor 4. Estos valores numéricos fueron luego transformados en factores (ver Tabla 1).

Finalmente, al observar que los resultados obtenidos con este preprocesamiento no eran buenos, se decidió unificar varios niveles de algunas variables que estaban muy desequilibradas, esta recodificación se realizó con la función `recode_factor` de la librería “dplyr”. El resultado de esta codificación se puede consultar en la sección de anexos (ver Tabla 2).

Variable	Niveles			Muestras por nivel		
Blood urea (bu)	0	3	6	108	5	1
	1	4	7	16	5	1
	2	5		11	1	
Sodium (sod)	0	3	6	4	14	22
	1	4	7	3	92	9
	2	5	8	6	49	1
Serum creatinine (sc)	0	3	6	159	22	1
	1	4		4	9	
	2	5		1	4	
Potassium (pot)	0	2		197	1	
	1	3		1	1	
Sugar (su)	0	3		170	8	
	1	4		6	6	
	2	5		9	1	

Tabla 3. Ejemplo de variables descompensadas.

3.2.2 Técnicas de reducción de dimensionalidad

Existen varios problemas asociados a bases de datos con un gran número de variables que pueden afectar a la precisión de los modelos, el primero es el riesgo de sobreajuste, el modelo puede adaptarse demasiado bien a los datos de entrenamiento y ser poco fiable con nuevos datos lo que se conoce como la maldición de la multidimensionalidad [6].

Otro problema asociado es la multicolinealidad de las variables, este problema ocurre cuando hay una alta correlación entre variables predictoras lo que puede provocar que

nuestro modelo sea demasiado sensible a pequeñas variaciones en los datos [20]. En el caso de la CKD existen varios síntomas que pueden ser tanto causa como consecuencia de la enfermedad y que están muy relacionados unos con otros, mantenerlos por tanto en nuestros modelos puede disminuir en gran medida la calidad de las predicciones.

Aplicar estas técnicas también nos permitirá comprobar si existe alguna variable que no aporte información significativa al modelo y que cuya eliminación puede aumentar la precisión del modelo [6][20].

Reducir la dimensionalidad de la base de datos no solo nos permitirá aumentar la precisión de nuestros modelos si no que reducirá la carga computacional haciendo menor los tiempos de espera y permitiendo una aplicación más rápida y eficiente [6].

3.2.2.1 Análisis de correspondencia múltiples:

El análisis de correspondencia múltiples (MCA) puede verse como una generalización del análisis de componentes principales cuando las variables a analizar son categóricas en lugar de cuantitativas [19]. El MCA codifica los datos creando varias columnas binarias para cada variable con la restricción de que una y solo una de las columnas tenga el valor 1. Este esquema de codificación crea dimensiones artificiales adicionales debido a que una variable categórica se codifica con varias columnas [19].

La interpretación del MCA se basa en las proximidades entre puntos en un mapa de baja dimensionalidad, generalmente de dos o tres dimensiones. En términos de proximidad entre variables, es importante distinguir dos casos. Por un lado, la proximidad entre niveles de una misma variable indica que los grupos de observaciones asociados a esos niveles son similares entre sí. Por otro lado, la proximidad entre niveles de diferentes variables implica que dichos niveles tienden a aparecer juntos en las observaciones [19-20].

En resumen, al realizar un MCA, podremos excluir variables cuyos niveles presenten disposiciones similares en el plano bidimensional tanto con sus propios niveles como con los niveles de otras variables. El MCA se llevará a cabo con las librerías “FactoMineR”, “Factoshiny” y “factoextra”

3.2.2.2 Análisis de componentes principales:

El Análisis de Componentes Principales (PCA) es una técnica estadística utilizada para reducir la dimensionalidad de un conjunto de datos, preservando al mismo tiempo la mayor cantidad posible de información. Su objetivo principal es encontrar las combinaciones lineales de las variables originales que expliquen la mayor variabilidad en los datos [20]. Tradicionalmente, el PCA se aplica a variables cuantitativas, pero es posible aplicar PCA a variables categóricas tras convertir estas variables a codificación one-hot [21]. una técnica utilizada en el procesamiento de datos para convertir variables categóricas en una representación numérica adecuada para su uso en algoritmos de aprendizaje automático [5]. Algunos estudios han demostrado que el uso de PCA en variables discretas es inferior a otros métodos como MCA [21]. Este tipo de análisis se puede realizar con la librería “Factominer”, la conversión de las variable a one-hot se realiza con la librería “FastDummies”.

3.2.2.3 Regresión Lasso:

La regresión Lasso introduce una penalización en la función objetivo del modelo de regresión, que se basa en la suma del valor absoluto de los coeficientes de las variables predictoras. Esta penalización permite que algunos coeficientes se vuelvan

exactamente cero, lo que significa que esas variables se eliminan del modelo [6]. Al igual que en PCA para aplicar la regresión lasso a variables categóricas se deben convertir las variables a codificación one-hot [22]. Esta técnica se puede aplicar con la librería “glmnet”.

3.2.3 Equilibrado de la variable respuesta

De acuerdo con los datos disponibles en la base de datos, de las 200 muestras, 128 pertenecen al grupo "CKD" mientras que solo hay 72 pacientes sanos en el grupo "no CKD". Este desequilibrio en la variable respuesta puede presentar desafíos al aplicar técnicas de aprendizaje automático, ya que los modelos tienden a sesgarse hacia la clase dominante y tienen dificultades para predecir correctamente la clase minoritaria [6]. Una solución comprobada es el sobremuestreo sintético de la variable minoritaria, esta técnica permite lograr una mayor precisión en los modelos en comparación con aquellos que no equilibran la variable respuesta [23].

Es importante ser cauteloso al generar un exceso de muestras sintéticas en el proceso de sobremuestreo, ya que esto puede llevar a problemas de sobreentrenamiento además es importante que las muestras que se generen sean representativas y no introduzcan un sesgo en los datos [23]. En este caso usaremos la librería “ROSE”.

3.2.4 Validación de los modelos

Los métodos de validación son esenciales para garantizar la calidad y el rendimiento de los modelos. Ayudan a evitar el sobreajuste, optimizar los hiperparámetros y evaluar la capacidad predictiva de los modelos [6].

Según el objetivo de cada problema y el tamaño de los datos, podemos elegir diferentes métodos de validación, pero el más usado consiste en dividir el conjunto de datos en dos partes: conjunto de entrenamiento y conjunto de prueba [24]. El conjunto de entrenamiento se utiliza para entrenar el modelo y el conjunto de prueba se utiliza para evaluarlo [6]. Adicionalmente utilizaremos el método de validación cruzada k-fold en el conjunto de entrenamiento, este método consiste en la división del conjunto de entrenamiento en k plieges que a su vez se utilizan para el entrenamiento y validación del modelo repitiéndose este proceso k veces, el resultado es el promedio de todas las pruebas de rendimiento de todos los pliegues [24]. En este proyecto el 67% de los datos se utilizarán para construir los modelos y el 33% restante para su validación, la validación cruzada será de 5 pliegues, es decir que el conjunto de entrenamiento se divide en 5 partes y el modelo se entrena y evalúa en 5 iteraciones diferentes. Tanto la división de la base de datos en entrenamiento y test como la validación cruzada se harán con el paquete “caret”.

3.2.5 Modelos

3.2.5.1 Modelo K vecinos cercanos (KNN)

El algoritmo de aprendizaje automático supervisado KNN clasifica nuevos ejemplos basándose en la similitud con los ejemplos etiquetados más cercanos en el espacio de características. Para ello se calcula la distancia euclidiana entre el punto de datos que se está intentando clasificar y los puntos de datos más cercanos en el conjunto de entrenamiento. La "K" en KNN representa el número de vecinos más cercanos que se toman en cuenta para tomar la decisión de clasificación y se debe equilibrar entre sobreajuste y subajuste. Elegir un k grande reduce el impacto del ruido en los datos, pero puede ignorar patrones importantes, mientras que un k pequeño permite que datos ruidosos o atípicos influyan demasiado en la clasificación. Normalmente los modelos alcanzan mayores rendimientos cuando su k es igual a la raíz cuadrada del número de ejemplos de entrenamiento [6]. Para crear este modelo usaremos la librería “caret”.

Fortalezas	Debilidades
• Simple y efectivo. • Toma y suposición sobre la distribución de datos subyacentes. • Rápida fase de entrenamiento.	• No produce un modelo, lo que limita la capacidad de comprender cómo se relacionan las características con la clase. • Requiere seleccionar la K correctamente. • Fase de clasificación lenta. • Las características nominales y los datos faltantes requieren procesamiento adicional.

Tabla 4. Fortaleza y debilitades KNN [6].

3.2.5.2 Máquinas de vectores de soporte (SVM)

Un SVM un algoritmo de aprendizaje automático supervisado utilizado tanto para problemas de clasificación como de regresión. Su objetivo principal es encontrar un hiperplano óptimo en un espacio de características que pueda separar eficientemente las diferentes clases de datos. De esta manera, el aprendizaje de SVM combina aspectos del aprendizaje basado en distancias, como de regresión lineal. Esta combinación es extremadamente poderosa y permite a los SVM modelar relaciones altamente complejas, de ahí que su uso este muy extendido en varios ámbitos de la bioinformática como el análisis de microarrays, estudio de enfermedades génicas e identificación de distintos canceres, siendo extremadamente poderoso cuando la variable respuesta es binaria [6].

Si los datos son linealmente separables, se puede encontrar un hiperplano que separe perfectamente los grupos. Sin embargo, los SVM también se pueden extender a problemas donde los puntos no son linealmente separables. En estos casos se puede usar un “kernel-trick” que consiste en agregar dimensiones adicionales a los datos para crear separación de esta manera [6].

La elección del kernel adecuado para una tarea de aprendizaje particular no sigue una regla fiable y depende del concepto a aprender, la cantidad de datos de entrenamiento y las relaciones entre las características. A menudo, se requiere un poco de prueba y error entrenando y evaluando varios SVM [6]. En este caso probaremos un modelo de kernel lineal y otro de kernel radial para ello utilizaremos de nuevo la librería “caret”.

Fortalezas	Debilidades
• Puede ser utilizado para problemas de clasificación o predicción numérica**.** • No se ve muy influenciado por datos ruidosos y no es muy propenso al sobreajuste • Mas sencillo que las redes neuronales.	• Encontrar el mejor modelo requiere probar diversas combinaciones de kernels y parámetros del modeloRequiere seleccionar la K correctamente. • Lento de entrenar. • Modelo de caja negra.

Tabla 5. Fortaleza y debilitades SVM [6].

3.2.5.3 Árbol de decisiones.

El árbol de decisión es uno de los métodos de aprendizaje automático más populares en el campo médico debido a su poder de clasificación. Utiliza una estructura en forma de árbol para modelar las relaciones entre las características de un conjunto de datos y los resultados [6].

El árbol de decisión comienza con un nodo único que representa las muestras de entrenamiento. Si todas las muestras están en la misma clase, el nodo se convierte en una hoja y se marca con esa clase. De lo contrario, el algoritmo elige el atributo discriminatorio como el nodo actual del árbol de decisión. Según el valor del atributo del nodo de decisión actual, las muestras de entrenamiento se dividen en varios subconjuntos, cada uno de los cuales forma una rama. Para cada subconjunto obtenido en el paso anterior, se repiten los pasos anteriores, formando recursivamente un árbol de decisión en cada una de las muestras particionadas [24]. Para crear el modelo podemos usar la librería “rpart”.

Fortalezas	Debilidades
• Un clasificador versátil que funciona bien en la mayoría de los problemas. • Proceso de aprendizaje altamente automático que puede manejar características numéricas o nominales, así como datos faltantes. • Excluye características no importantes. • Puede ser utilizado en conjuntos de datos pequeños y grandes. • Produce un modelo que puede ser interpretado sin necesidad de conocimientos matemáticos (en el caso de árboles relativamente pequeños). • Más eficiente que otros modelos complejos.	• Los modelos de árbol de decisión a menudo presentan sesgos hacia divisiones en características con un gran número de niveles. • Es fácil sobreajustar o subajustar el modelo. • Puede tener dificultades para modelar algunas relaciones debido a la dependencia de divisiones paralelas a los ejes. • Pequeños cambios en los datos de entrenamiento pueden provocar grandes cambios en la lógica de decisión. • Los árboles grandes pueden ser difíciles de interpretar y las decisiones que toman pueden parecer contraintuitivas.

Tabla 6. Fortaleza y debilitades árbol de decisiones [6].

3.2.5.4 Random forest.

Un Random Forest es un algoritmo de aprendizaje automático que consiste en la construcción de múltiples árboles de decisión independientes, conocidos como árboles de decisión aleatorios, y combina sus predicciones para obtener un resultado final. El Random Forest es conocido por su capacidad para manejar conjuntos de datos grandes y complejos, así como para lidiar con características irrelevantes o ruidosas por lo que es ampliamente utilizado en diversos campos, como la medicina, la biología, y el análisis de datos [24]. Para crear este modelo podemos usar la librería “RandomForest”

Fortalezas	Debilidades
• Un modelo versátil que funciona bien en la mayoría de los problemas. • Capaz de manejar datos ruidosos o faltantes, así como características categóricas o continuas. • Selecciona solo las características más importantes. • Puede ser utilizado en conjuntos de datos con un número extremadamente grande de características o ejemplos.	• A diferencia de un árbol de decisión, el modelo no es fácilmente interpretable. • Puede requerir ajustes para adaptar el modelo a los datos.

Tabla 7. Fortaleza y debilitades de random forest [6].

3.2.5.5 Regresión binomial

La regresión binomial es un tipo de análisis de regresión utilizado cuando la variable de respuesta es binaria en la que se puede modelar la relación entre la variable de respuesta y uno o más predictores. El objetivo de la regresión binomial es estimar los coeficientes de regresión que representan la relación entre los predictores y la probabilidad de éxito en el evento binario [6].

La regresión binomial se utiliza en diversos campos, como la medicina, la psicología, la ciencia política y la investigación de mercado, para analizar y predecir resultados binarios, como la presencia o ausencia de una enfermedad, la adopción o no de un comportamiento, o el éxito o fracaso de un evento [15]. Para realitzar este tipo de regresión podemos usar la función “glm()” del paquete “glmnet” [18].

En general las fortalezas y debilitades de los modelos de regresión son:

Fortalezas	Debilidades
• El enfoque más común para modelar datos numéricos. • Puede adaptarse para modelar casi cualquier tarea de modelado. • Proporciona estimaciones tanto de la fuerza como del tamaño de las relaciones entre las características y el resultado	• Realiza suposiciones fuertes sobre los datos. • La forma del modelo debe ser especificada por el usuario de antemano. • No maneja datos faltantes. • Los datos categóricos requieren un procesamiento adicional. • Requiere ciertos conocimientos de estadística para comprender el modelo.

Tabla 8. Fortaleza y debilitades de los modelos de regresión [6].

El análisis del resumen de un modelo binario implica examinar diferentes aspectos para evaluar su rendimiento y comprender las relaciones entre las variables.

Lo primero la significancia estadística no solo del modelo completo si no de cada uno de los coeficientes que conforman el modelo en este caso cada nivel de cada variable,

por convenio cualquier p-valor superior a 0.05 se considera no significativo [20]. Son importantes también los coeficientes de cada variable, estos nos permiten ver la relación entre cada nivel de la variable y la respuesta. Los errores estándar nos dan una idea de la incertidumbre asociada a las predicciones, errores estándar muy altos indican una mayor variabilidad y menor precisión en las estimaciones de los coeficientes. Por último, podremos comprobar el grado de colinealidad de cada variable a través de los valores AIC que se pueden obtener con las librerías “car” o “vcd”.

3.2.6 Elección de los modelos

Al elegir el mejor modelo de ML, se deben considerar varios aspectos clave. Junto con la validación cruzada, las matrices de confusión y la precisión son métricas fundamentales. Las matrices de confusión brindan una visión detallada de cómo el modelo clasifica las muestras en diferentes categorías, mientras que la precisión mide la proporción de muestras clasificadas correctamente. Para seleccionar el modelo más confiable, se busca un equilibrio entre la precisión general, la precisión por categoría y una baja tasa de errores en la matriz de confusión [6].

Sin embargo, la elección del modelo no se basa exclusivamente en estas métricas, sino que también debe estar alineada con los objetivos específicos del proyecto. En el caso de los modelos de diagnóstico, es importante evitar los falsos negativos para minimizar el riesgo de diagnosticar incorrectamente a personas enfermas como sanas. Además del rendimiento, también es esencial considerar la carga computacional y la interpretabilidad de los modelos [6].

En este caso el aspecto más importante a valorar será la precisión del modelo penalizando los modelos con mayor número de falsos negativos.

3.2.7 Diseño y programación de la APP

3.2.7.1 Diseño

Desarrollaremos una aplicación en la que el usuario tendrá que ingresar los datos de las variables seleccionadas, y luego el modelo le proporcionaría la predicción correspondiente, con el porcentaje de seguridad calculado por el modelo.

Para mejorar la interpretabilidad de los modelos se añadirá una segunda página a la aplicación en la que se podrán consultar las variables seleccionadas por el modelo y como estas se relacionan entre ellas y con la variable respuesta.

En lugar de que el paciente introduzca manualmente los valores de las variables, se le ofrecerá la opción de seleccionar entre diferentes opciones. De esta manera, se evitarán errores causados por la introducción de valores negativos, fuera de rango o no contemplados por el modelo.

Para que el paciente pueda ubicar cada variable en su respectivo grupo, se mostrará en pantalla el rango de valores correspondiente a cada nivel de la variable.

A continuación, se muestra un croquis del diseño inicial de la aplicación:

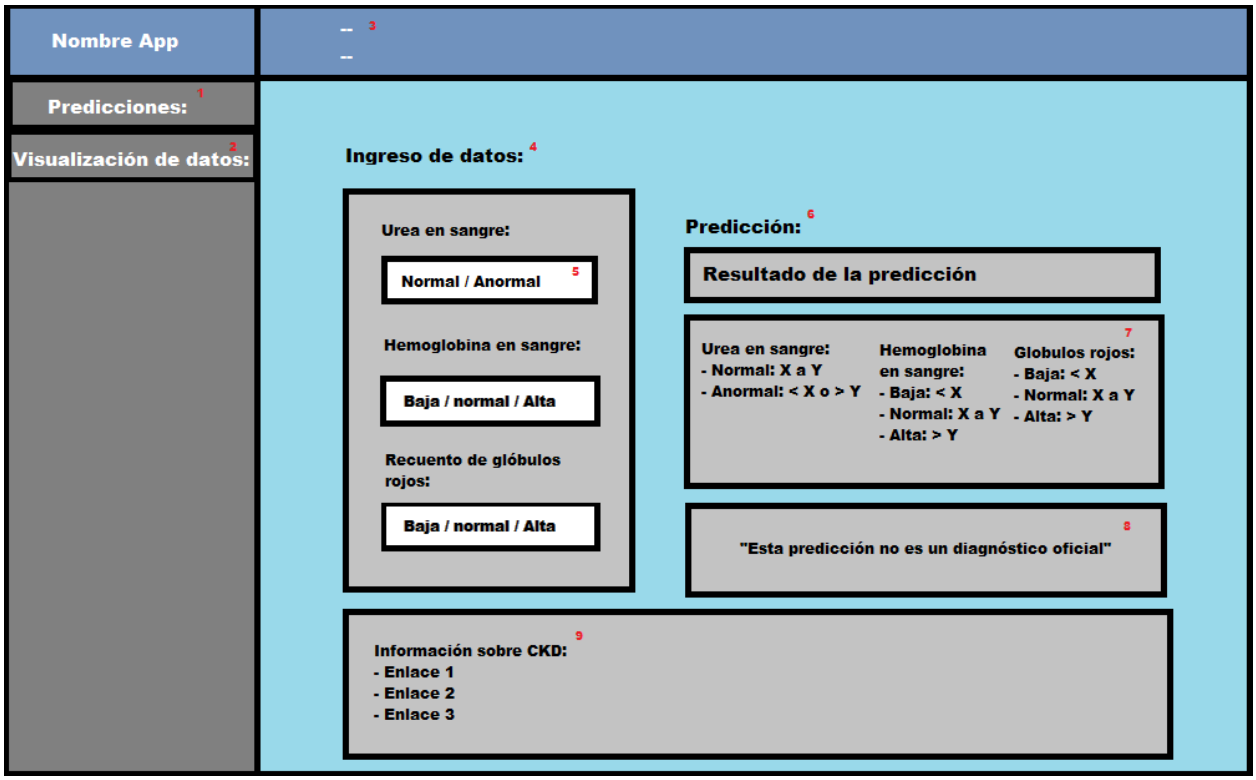


Fig. 8. Croquis de la APP mostrando la página “Predicciones”.

1- Es la ventana “Predicciones” si hacemos click en esta pestaña nos aparecerá la página mostrada en el croquis donde deberemos seleccionar los niveles de la variable para recibir la predicción.
2- Nos lleva a la parte de la App que nos permitirá explorar la base de datos utilizada para el modelo.
3- Este botón nos permitirá ampliar o reducir el tamaño de la página quitando o poniendo las ventanas de predicción o visualización (ventana gris).
4- Parte de la pantalla en la que seleccionaremos los niveles de las variables.
5- Cada variable tendrá un cartel desplegable en el que podremos seleccionar los niveles.
6- En esta parte de la pantalla recibiremos los resultados.
7- Este cuadro será el esquema que permitirá al paciente saber a qué nivel pertenece en cada variable.
8- Se añadirá un cuadro que informe al usuario de que esta predicción no constituye un diagnóstico alentándole a acudir al médico en caso de que sea necesario.
9- Se aportará al usuario información extra sobre la enfermedad.

La segunda parte de la aplicación es adicional y no tiene utilidad real para los pacientes en términos de diagnóstico o tratamiento de enfermedades renales. Su propósito principal es mostrar la relación entre diferentes variables y permitir una exploración más detallada de los datos utilizados en el modelo predictivo.

La página de “visualización de datos” tendrá la siguiente estructura:

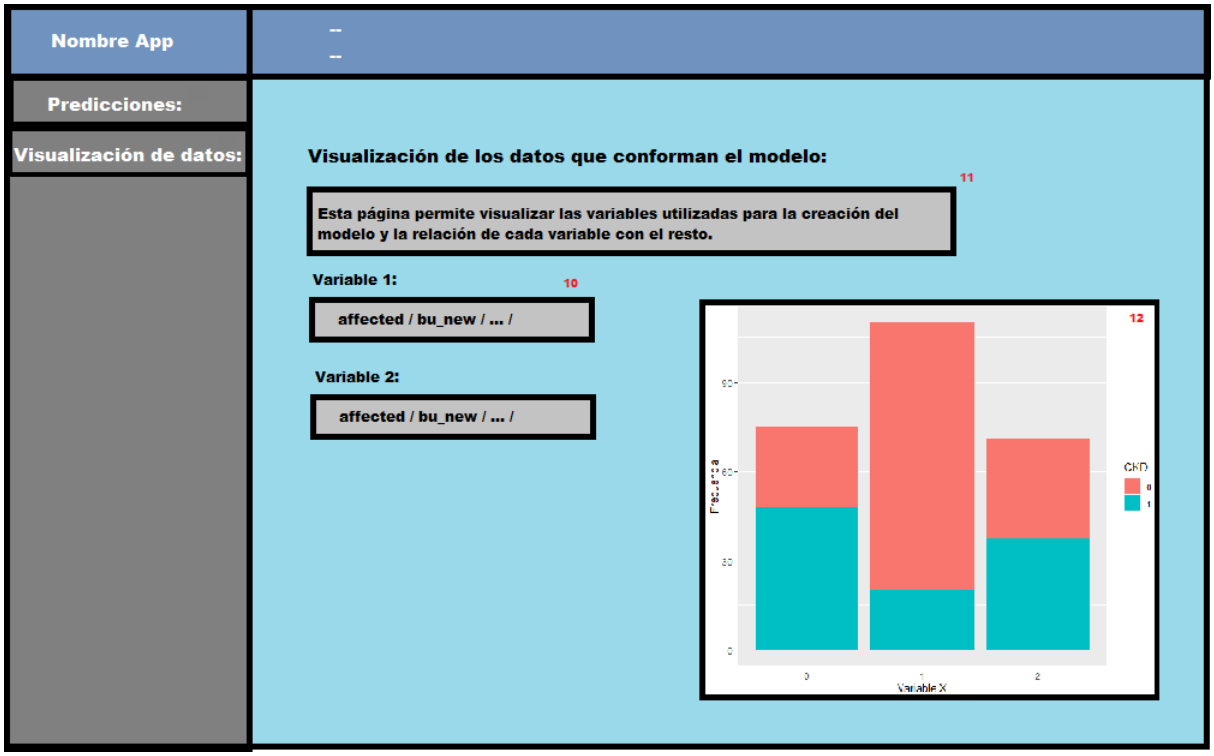


Fig. 9. Croquis de la APP mostrando la página “Visualización de datos”.

10- En esta sección habrá dos botones, el primero permitirá escoger el número de columnas que aparezcan en el histograma, es decir si se escoge una variable con 2 niveles aparecerán dos columnas, si se escoge una con 3 aparecerán 3. El segundo botón permitirá controlar los grupos en que se divide cada columna. Por ejemplo, el histograma que aparece en el croquis se genera al escoger en primer lugar la variable “hemo_new” y después “affected”. Veríamos la proporción de casos de CKD en cada nivel de la variable “hemo_new”.
11- Este cartel simplemente sirve para explicar el funcionamiento de la página.
12- Resultado de la visualización.

3.2.7.2 Programación

Para el desarrollo de la APP usamos Shinydashboard. Esta librería de R permite usar el lenguaje HTML para definir la estructura de la aplicación y los diferentes elementos visuales, como encabezados, paneles, botones y tablas. Después se puede usar CSS para personalizar el estilo, el diseño y los colores de los elementos HTML [7].

La estructura básica de una APP en Shinydashboard consta de:

1- Interfaz de usuario (ui): define la apariencia y la estructura de la interfaz de usuario de la aplicación. Podemos usar estructuras como *dashboardHeader, dashboardSidebar, dashboardBody* o *tabItems* etc. para añadir encabezados, barras laterales, cuerpos o secciones de la aplicación [7].
2- Server: Esta sección contiene el código que define la lógica y el comportamiento de la aplicación. En el servidor, se definen los componentes de salida (output) que representan los resultados o visualizaciones que se mostrarán en la interfaz de usuario. También se pueden utilizar los

componentes de entrada (input) para capturar las selecciones o acciones del usuario [7].

3- shinyApp(ui, server): La función shinyApp se utiliza para combinar la interfaz de usuario (ui) y el servidor (server) en una aplicación Shiny completa [7].

En una aplicación Shinydashboard, el código se escribe en archivos R con extensión ".R", donde se combinan instrucciones R, HTML y CSS de manera integrada. Estos archivos contienen la lógica de la aplicación, incluyendo la definición de la interfaz gráfica, la carga de datos, la implementación de funciones y la generación de resultados dinámicos [7].

3.2.7.1 Repositorio para la aplicación

Una vez creada la aplicación se exportará a *shinyapp.io* una plataforma en la nube que permite a los usuarios de R y Shiny compartir y desplegar aplicaciones interactivas en línea de manera sencilla. Para exportar una aplicación a *shinyapp.io* aparte de las liberías básicas de shiny y shinydashboard necesitaremos instalar rsconnect para poder conectar nuestra aplicación con la plataforma de *shinyapp.io.*

Una vez instaladas las librerías simplemente hay que seguir los pasos pasos de implementación proporcionados por la plataforma para cargar y publicar la aplicación.

4. Resultados

4.1 Exploración de la base de datos

El resumen de la base de datos nos permite observar como varias de las variables incluyendo la respuesta están descompensadas. Por ejemplo, los niveles de albumina (al) o los de azúcar (su).

```
affected sg     al      rbc     su      pc      pcc     ba          bgr          bu           sod     sc      pot          hemo
0: 72    0: 3   0:116   0:175   0:170   0:155   0:173   0:189   1      :79   0      :108   4      :92   0:159   0:197   2      :49
1:128    1:45   1: 21   1: 25   1:  6   1: 45   1: 27   1: 11   0      :70   1      : 53   5      :49   1:  4   1:  1   1      :28
         2:36   2: 27           2:  9                           3      :14   2      : 16   6      :22   2:  1   2:  1   4      :26
         3:75   3: 23           3:  8                           2      :13   3      : 11   3      :14   3: 22   3:  1   8      :23
         4:41   4: 13           4:  6                           4      :11   4      :  5   7      : 9   4:  9           5      :20
                                5:  1                           5      : 4   5      :  5   2      : 6   5:  4           3      :19
                                                                (Other): 9   (Other):  2   (Other): 8   6:  1           (Other):35
     pcv          rbcc         wbcc     htn     dm      cad     appet   pe      ane          age
6      :56   4      :96   6      :98   0:122   0:130   0:178   0:160   0:165   0:168   7      :48
7      :29   5      :23   5      :47   1: 78   1: 70   1: 22   1: 40   1: 35   1: 32   8      :34
5      :23   2      :21   7      :29                                                    6      :33
4      :22   3      :21   0      :10                                                    5      :31
9      :19   6      :18   1      : 6                                                    3      :14
3      :18   7      : 9   2      : 6                                                    4      :12
(Other):33   (Other):12   (Other): 4                                                    (Other):28
```

Fig. 10. Resumen del dataset.

4.2 Técnicas de reducción de dimensionalidad:

4.2.1 MCA

Vemos como Las 10 primeras dimensiones apenas abarcan ún 35% de la variabilidad siendo necesario abarcar 31 dimensiones para llegar a explicar el 70% de la variabilidad.

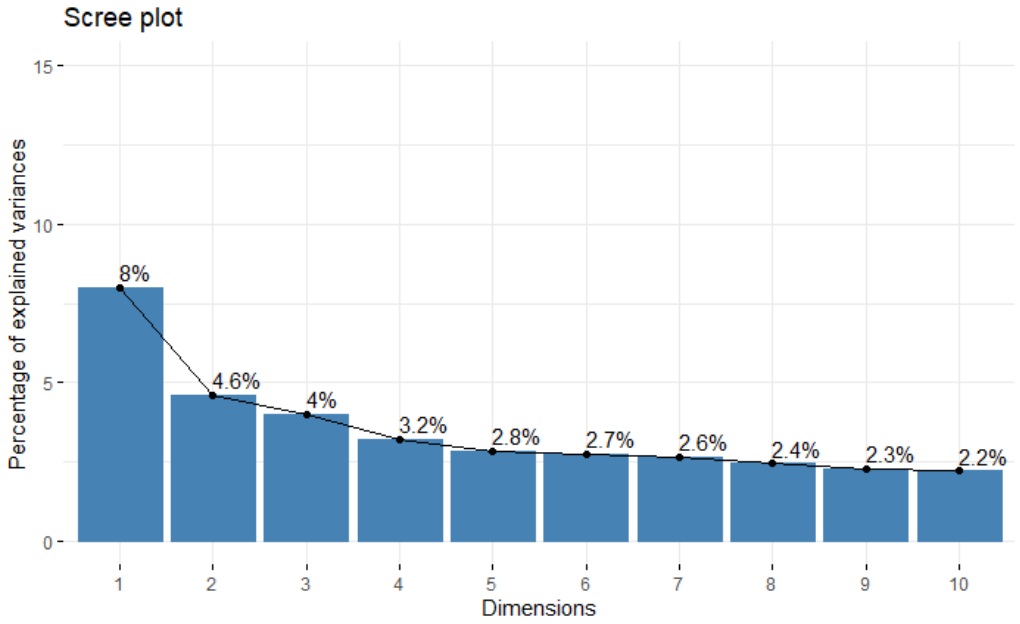


Fig. 11. Variabilidad explicada por cada dimensión.

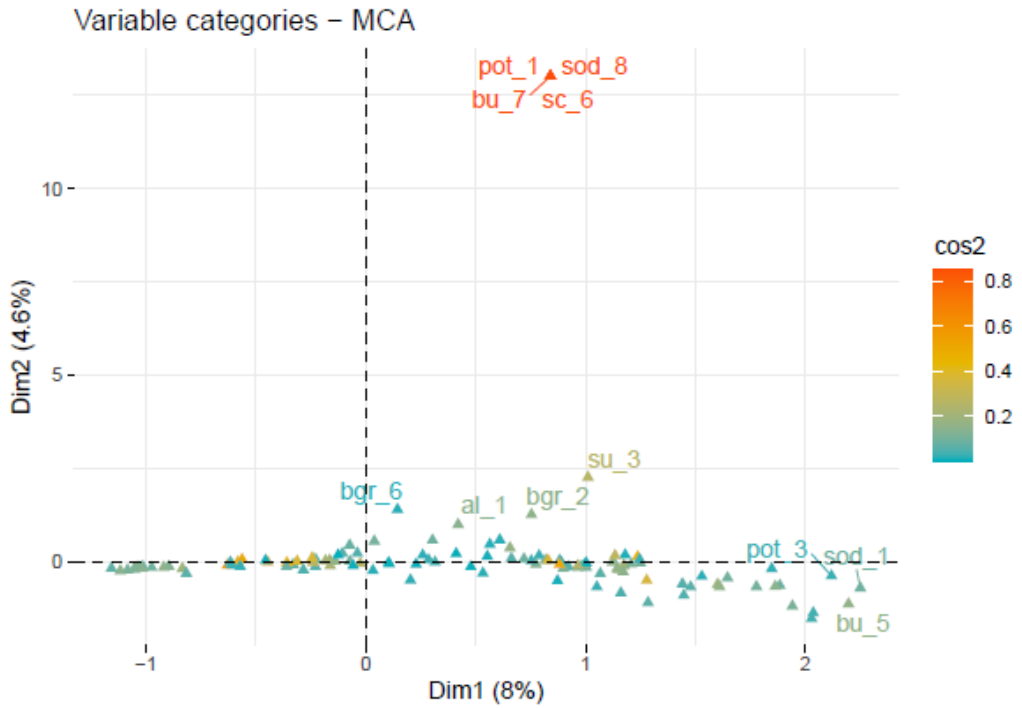


Fig. 12 Representación de los individuos y variables sobre las dos primeras dimensiones.

Existe mucho solapamiento de variables que no parecen aportar mucha información. Sin embargo, la representación tampoco es del todo fiable pues solo abarca el 13% de la variabilidad.

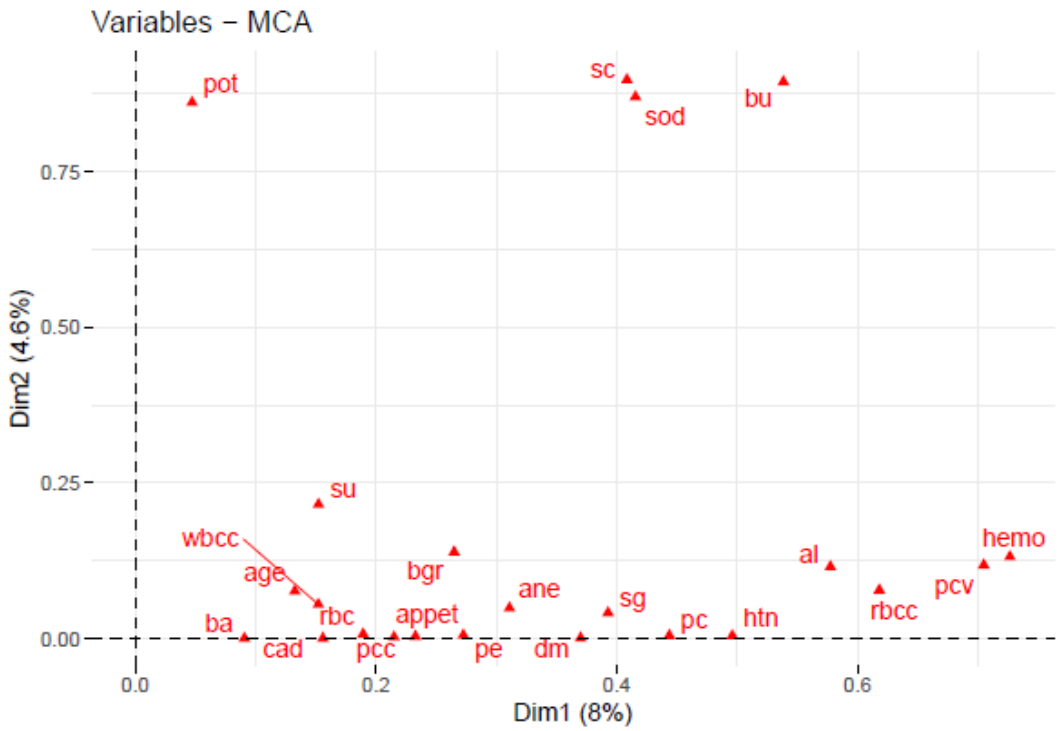


Fig. 13 Correlación entre las 2 primeras dimensiones y cada variable.

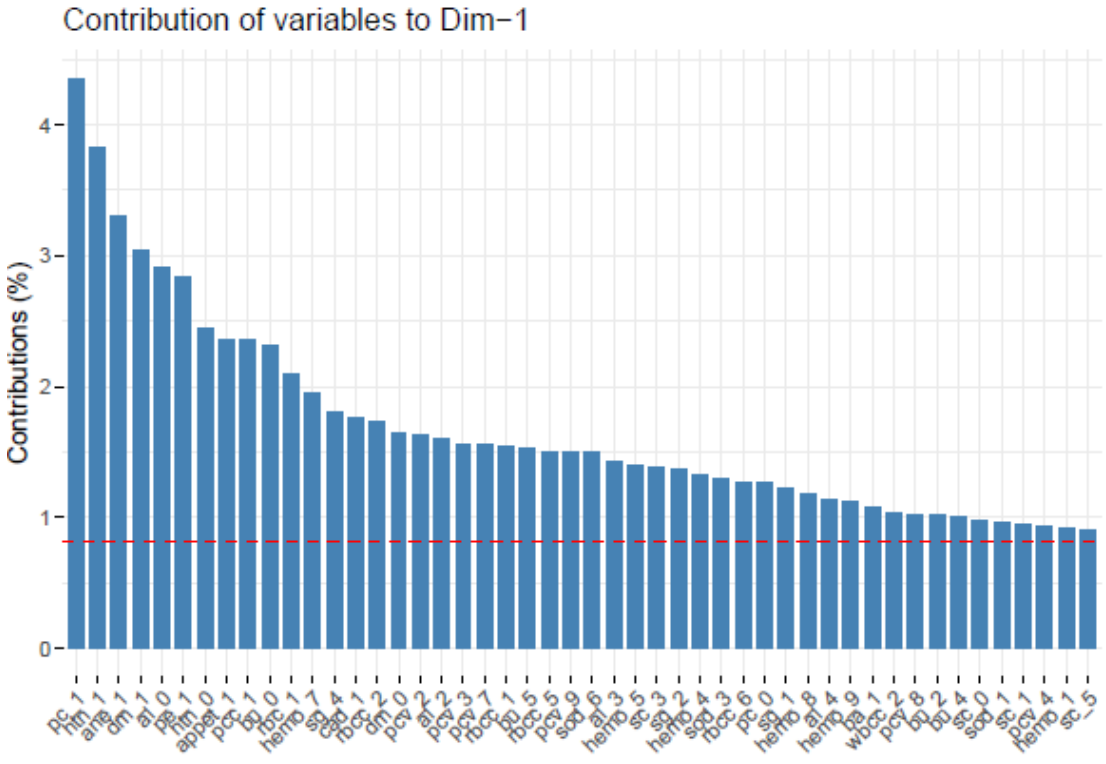
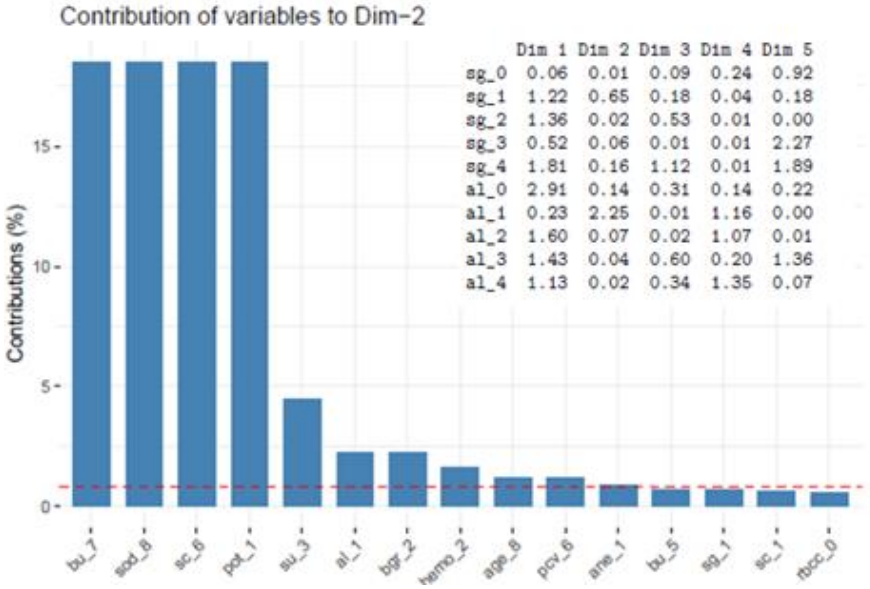


Fig. 14 Peso de cada nivel de cada variable sobre la primera dimensión.

Sobre la primera dimensión podemos destacar variables como la hipertensión (htn) cuyos dos niveles aparecen en segunda como séptima posición, ambos niveles de pus cells (pc) aparecen en el top 40 al igual que la diabetes melitus (dm), la anemia o la falta de apetito.

La albumina solo aparecen los niveles 0, 2 3 y 4 en el top 40 además de estar muy descompensada. Pedal anemia (pe) parece relevante, aunque puede estar altamente relacionada con la anemia. Blood urea está muy descompensada y solo aporta información en 1 de los 4 niveles.



	Dim 1	Dim 2	Dim 3	Dim 4	Dim 5
sg_0	0.06	0.01	0.09	0.24	0.92
sg_1	1.22	0.65	0.18	0.04	0.18
sg_2	1.36	0.02	0.53	0.01	0.00
sg_3	0.52	0.06	0.01	0.01	2.27
sg_4	1.81	0.16	1.12	0.01	1.89
al_0	2.91	0.14	0.31	0.14	0.22
al_1	0.23	2.25	0.01	1.16	0.00
al_2	1.60	0.07	0.02	1.07	0.01
al_3	1.43	0.04	0.60	0.20	1.36
al_4	1.13	0.02	0.34	1.35	0.07

Fig. 15. Peso de cada variable en la segunda dimensión, resumen del peso de algunas variables en las 5 primeras dimensiones.

En la segunda dimensión apenas hay variables relevantes destacan: pot (potasio), so (sodio), sc (serum creatine), y bu (blood urea).

Tras el análisis MCA decidimos eliminar las siguientes variables:

- Sugar (su) y glucosa (bgr). Ambas variables tienen un peso mínimo en ambas dimensiones en todos sus niveles.
- Albumina (al). solo aparecen los niveles 0, 2, 3 y 4 entre las variables con pesos significativos además está muy descompensada.
- Rbcc (red blood cell volume) y hemo (hemoglobine), están muy relacionadas con rbc, no todos sus niveles son representativos y tienen menos peso que rbc.
- pcc (pus cell clums) y ba (bacteria) están directamente relacionadas con pc (pus cells) y aportan menos información.
- pot (potasium) de 200 muestras 197 pertenecen al nivel 0.
- pcv (packed cell volume), wbcc (white blood cell counts), sg (specific gravity), bu (blood urea), sod (sodium) y age. pocos o ninguno de los niveles aparecen en el top 48 variables significativas.

En resumen, mantendremos las siguientes variables: rbc (red blood cells), pc (pus cells), htn (hypertension), cad (coronary artery disease), dm (diabetes melitus), pe (pedal anemia), appet (apetito), Ane (anemia).

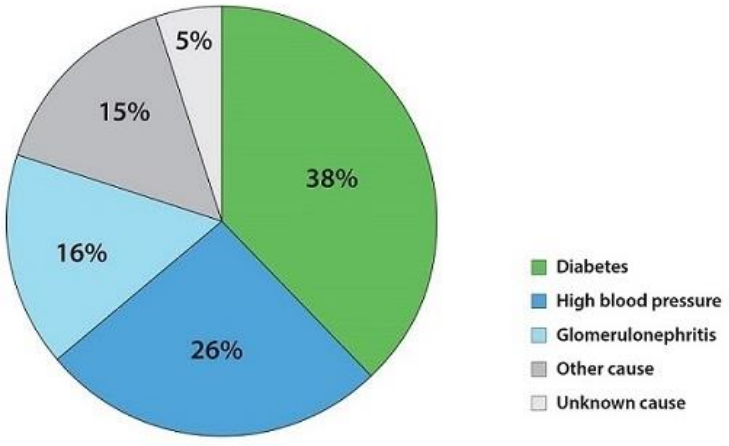
Bajo la sospecha de que algunas de estas variables podrían generar problemas de colinealidad como por ejemplo rbc, ane y pe además de entre htn y cad realizamos una prueba de multicolinealidad con la función `*assoc*` del paquete vcd. El resultado obtenido fue el siguiente.

				rbc	0						1							
				htn	0			1			0			1				
				cad	0		1		0		1		0		1		0	
				ane	0	1	0	1	0	1	0	1	0	1	0	1	0	1
affected	pc	dm	appet															
0	0	0	0		72	0	0	0	0	0	0	0	0	0	0	0	0	0
			1		0	0	0	0	0	0	0	0	0	0	0	0	0	0
		1	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
			1		0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
			1		0	0	0	0	0	0	0	0	0	0	0	0	0	0
		1	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
			1		0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0		16	0	0	1	7	0	2	0	2	1	0	0	2	0	0	1
			1		2	2	0	0	1	0	1	0	0	0	0	0	0	0	0	0
		1	0		6	1	0	0	14	4	3	2	1	0	0	0	2	0	0	0
			1		2	0	2	0	5	1	2	0	0	0	0	0	0	0	0	0
	1	0	0		4	1	0	0	0	1	1	0	1	0	0	0	1	1	0	0
			1		2	2	0	0	1	2	0	0	1	1	0	0	1	0	0	0
		1	0		2	0	0	0	2	4	1	1	0	0	0	0	1	0	2	0
			1		0	0	0	0	3	1	1	0	0	0	0	0	1	4	1	1

0	1
72	128

Fig. 16 Tabla de contingencia entre las variables elegidas y la variable respuesta.

Esta tabla permite observar que hay una clara relación entre todas estas variables y la respuesta. Ninguno de los pacientes sanos mostró ninguno de los síntomas. Según estos datos el 100% de los usuarios con falta de apetito, anemia o hipertensión sufrirán de CKD lo cual lógicamente no puede ser cierto. De hecho, otros estudios con un espacio muestral mucho mayor otorgan resultados muy distintos.



N=726,331 (all ages, 2016)
Source: US Renal Data System
*Includes polycystic kidney disease, among other causes.

Fig. 17 Causas informadas de enfermedad renal en etapa terminal [9].

Si bien la síntomas como la hipertensión o la diabetes pueden ser buenos predictores de la enfermedad el sesgo que presenta nuestra base de datos nos impide incluir estas variables, debemos buscar otra solución que incluya variables descartadas anteriormente.

Si creamos el modelo con algunas de estas variables, por ejemplo, el apetito y la hipertensión recibiremos el siguiente aviso. *"Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred".*

```
Call:
glm(formula = affected ~ appet + htn, family = binomial(link = "logit"),
    data = data_reduced)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.90052  -0.90052   0.00007   0.00008   1.48230

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   -0.6931     0.2041  -3.396 0.000684 ***
appet1        19.7208  2195.7879   0.009 0.992834
htn1          20.6505  1813.5502   0.011 0.990915
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 261.37  on 199  degrees of freedom
Residual deviance: 137.49  on 197  degrees of freedom
AIC: 143.49
```

Fig. 18 Resumen del modelo usando la hipertensión y falta de apetito.

Este mensaje es un aviso que nos indica que las probabilidades ajustadas del modelo son muy cercanas a 0 o 1. Esto puede suceder si el modelo está sobreajustado a los datos o si hay variables predictoras altamente correlacionadas o redundantes.

Si nos fijamos en los errores estándar estos son altísimos y las variables no resultan significativas. Lo mismo pasa si añadimos todas las variables seleccionadas, vemos que "Pr(>|Z|)" es 1 o prácticamente 1, lo que sugiere que ninguna de las variables predictoras está aportando información útil al modelo. Además, los errores estándar se disparan lo que indica que la estimación del cociente no es precisa.

```
Call:
glm(formula = affected ~ ., family = binomial(link = "logit"),
    data = data_reduced)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.6335  -0.6335   0.0000   0.0000   1.8465

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   -1.5041     0.2764  -5.442 5.27e-08 ***
rbc1          20.6668  6086.4915   0.003    0.997
pc1           20.7970  4639.7130   0.004    0.996
htn1          20.8902  3714.9542   0.006    0.996
dm1           20.8906  3983.7550   0.005    0.996
cad1          17.8372 13496.0331   0.001    0.999
appet1        20.2734  4984.4563   0.004    0.997
ane1           2.2490 11785.9957   0.000    1.000
---
```

Fig. 19. Resumen del modelo que incluye todas las variables seleccionadas tras el MCA.

4.2.2 Regressión Lasso

La regresión lasso en variables categóricas requiere del procesamiento de las variables a codificación one-hot. La codificación “one-hot” es una técnica de codificación utilizada en el aprendizaje automático y la minería de datos para representar variables categóricas como variables numéricas.

Por ejemplo, si se tiene una variable categórica “color” con tres posibles valores: rojo, verde y azul, se crearían tres variables binarias: “color_rojo”, “color_verde” y “color_azul”. Para cada fila en los datos, la variable binaria correspondiente al valor real de la variable categórica tendría el valor 1 y las demás variables binarias tendrían el valor 0.

Cuadro 1: Chronic kidney disease dataset:

affected	rbc_0	rbc_1	pc_0	pc_1	htn_0	htn_1	dm_0
1	1	0	1	0	1	0	1
1	1	0	1	0	1	0	1
1	0	1	0	1	1	0	1
1	1	0	1	0	1	0	1
1	1	0	1	0	1	0	0

Tabla 9. Estructura de la base de datos tras la codificación one-hot.

Los resultados obtenidos fueron los siguientes:

```
15 x 1 sparse Matrix of class "dgCMatrix"
                    s1
(Intercept)  1.426985e+01
rbc_0       -2.250223e+00
rbc_1        3.340790e-14
pc_0        -2.779686e+00
pc_1         5.514804e-15
htn_0       -3.447795e+00
htn_1        1.085035e-16
dm_0        -3.149484e+00
dm_1         1.311904e-13
cad_0       -1.577518e-01
cad_1        7.447258e-16
appet_0     -2.261993e+00
appet_1      1.433208e-15
ane_0       -1.425949e+00
ane_1        9.663469e-03
```

Fig. 20 Coeficientes obtenidos tras realizar la regresión lasso.

Como se puede ver, todos los coeficientes son muy cercanos a cero, lo que indica que estas variables al parecer no son útiles para predecir la variable respuesta. Cuando realizamos el análisis incluyendo todas las variables el resultado que obtenemos es el mismo, ninguna de las variables parece significativa.

4.2.3 PCA

En el caso de utilizar PCA en variables categóricas que han sido codificadas mediante one-hot encoding, el proceso de escalado de los datos no es necesario. Esto se debe a que la codificación one-hot ya convierte las variables categóricas en variables binarias (0 y 1) que representan la presencia o ausencia de cada categoría.

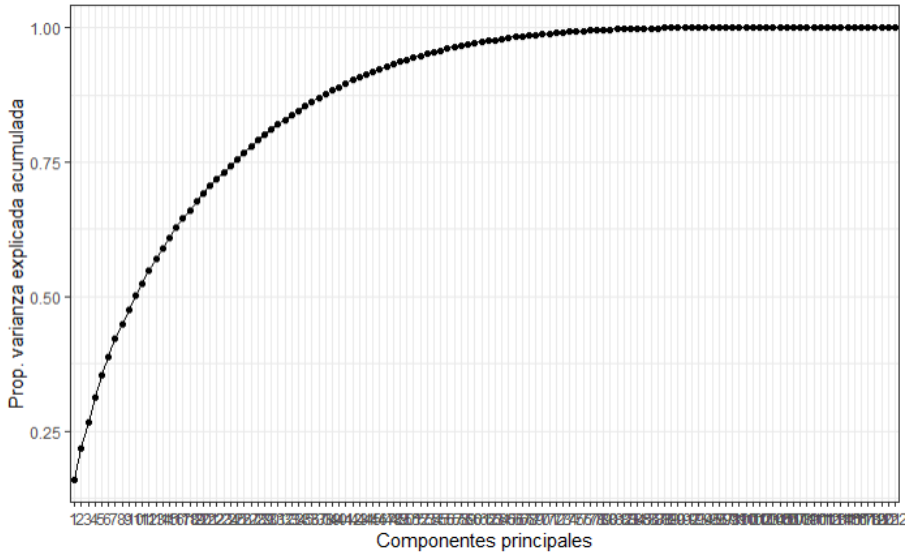


Fig. 21 Variabilidad explicada por cada componente.

El número de componentes es tan grande que no se aprecia bien cuantas componentes son necesarias en el análisis, si consultamos los valores numéricos con las 20 primeras dimensiones abarcamos un 70% de la variabilidad respecto a las 31 que necesitábamos en el MCA.

Tras crear un modelo con las 20 primeras dimensiones el resultado obtenido fue el siguiente:

```
Call:
glm(formula = affected ~ ., family = binomial, data = dimensiones)

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-2.409e-06  -2.409e-06   2.409e-06   2.409e-06   2.409e-06

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      -2.657e+01  8.557e+04       0        1
`data$affected`1  5.313e+01  1.278e+05       0        1
PC1               2.232e-07  4.136e+04       0        1
PC2               1.770e-07  3.880e+04       0        1
PC3               9.876e-08  3.513e+04       0        1
PC4              -1.761e-07  3.573e+04       0        1
PC5               1.098e-07  3.910e+04       0        1
PC6               2.613e-08  4.154e+04       0        1
PC7              -1.390e-07  4.374e+04       0        1
PC8              -2.060e-07  4.681e+04       0        1
PC9               2.047e-07  4.707e+04       0        1
PC10             -2.149e-07  4.915e+04       0        1
PC11              9.578e-08  4.995e+04       0        1
PC12              3.453e-07  5.060e+04       0        1
PC13              3.186e-07  5.156e+04       0        1
PC14              1.373e-07  5.437e+04       0        1
PC15             -1.425e-07  5.928e+04       0        1
PC16             -3.923e-08  5.826e+04       0        1
PC17              9.838e-09  5.889e+04       0        1
PC18             -1.695e-07  5.918e+04       0        1
PC19              9.257e-08  6.094e+04       0        1
PC20             -9.425e-08  6.398e+04       0        1
PC21              4.361e-08  6.529e+04       0        1
```

Fig. 22 Resumen del modelo creado con las 21 primeras dimensiones.

De nuevo los errores estándar son muy altos, tenemos estimaciones cercanas a 0 valores Pr(>|Z|) iguales a 1.

En conclusión, parece que no hay ninguna combinación de variables que no resulte problemática pues la toma de datos parece bastante sesgada. Por un lado, vamos a probar a construir los modelos con las variables seleccionadas: rbc (red blood cells), pc (pus cells), htn (hypertension), cad (coronary artery disease), dm (diabetes melitus), pe (pedal anemia), appet (apetito), Ane (anemia). Y por otor vamos a repetir el proceso de reducción de dimensionalidad tras unificar los niveles de algunas variables para que no estén tan descompensadas. El resultado de este nuevo preprocesamiento es el que aparece en la Tabla 2 en la sección de anexos.

El preprocesamiento consistió en eliminar las siguientes variables:

- pot, sod, su, sc, appet, ane y pe están tan desequilibradas que no vale la pena incluirlas.
- rbc (red blood cells) ya que está muy correlacionada con rbcc (red blood cells volume).
- pcc (pus cell clums) y ba (bacteria), ambas variables se correlacionan entre sí además de con pc (pus cells) siendo pc la más equilibrada de las 3 variables.
- cad (coronary artery disease) está demasiado relacionada con la hipertensión además de estar desequilibrada.

Unificar los niveles de las siguientes variables:

- Sg: Las 3 muestras del nivel 0 se englobarán en el nivel 1.
- al: los niveles del 1 al 4 se unirán en uno solo.
- rbcc: 0:3, 4, 5:8.
- wbcc: 0:5, 6, 7:8.
- bgr: unir los niveles del 2 al 9.
- bu: Unir los niveles del 1 al 7.
- hemo: unir 0:2, 3:5 y 6:9.
- pcv: 0:3, 4:5, 6, 7:9.
- age: 0:4, 5:6, 7:9.

Con la nueva codificación el `*dataset*` queda de la siguiente manera:

affected	sg_new	al_new	pc	bgr_new	bu_new	hemo_new	pcv_new	rbcc_new	wbcc_new	htn	dm	age_new
1	3	1	0	0	0	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1	1	0	0	0	0
1	1	1	1	0	1	2	1	1	0	0	0	0
1	1	1	0	1	0	1	4	1	1	0	0	0
1	2	0	0	2	0	1	3	2	1	0	1	0

Tabla 10. Estructura del `dataset` tras unificar los niveles de ciertas variables.

```
affected sg_new al_new  pc      bgr_new bu_new  hemo_new pcv_new rbcc_new
0: 72    1:48   0:116   0:155   0:70    0:108   0:81     0:34    0:53
1:128    2:36   1: 84   1: 45   1:79    1: 92   1:65     1:45    1:96
         3:75                   2:51            2:54     3:56    2:51
         4:41                                           4:65
wbcc_new htn     dm      age_new
0:72     0:122   0:130   0:44
1:98     1: 78   1: 70   1:64
2:30                     2:92
```

Fig. 23 Resumen del `dataset` tras unificar los niveles de ciertas variables.

4.3 Exploración visual

Vamos a comprobar de forma visual como se relacionan algunas de estas variables con las enfermedad crónica de riñón. Como es de esperar a medida que avanza la edad las probabilidades de padecer CKD aumentan.

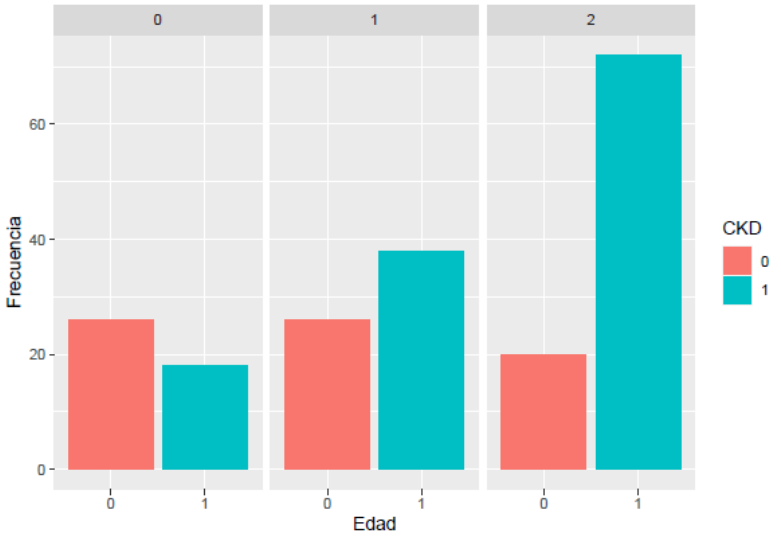


Fig. 24 Relación entre CKD y los tres niveles de edad.

En el siguiente gráfico vemos la relación de la hemoglobina con la variable respuesta en cada uno de los 3 niveles de edad. Este gráfico parece apuntar que niveles medios de hemoglobina en sangre están relacionados con la ausencia de CKD por otro lado cuando los niveles de hemoglobina son muy altos o en especial si son bajos parecen ser un síntoma de CKD.

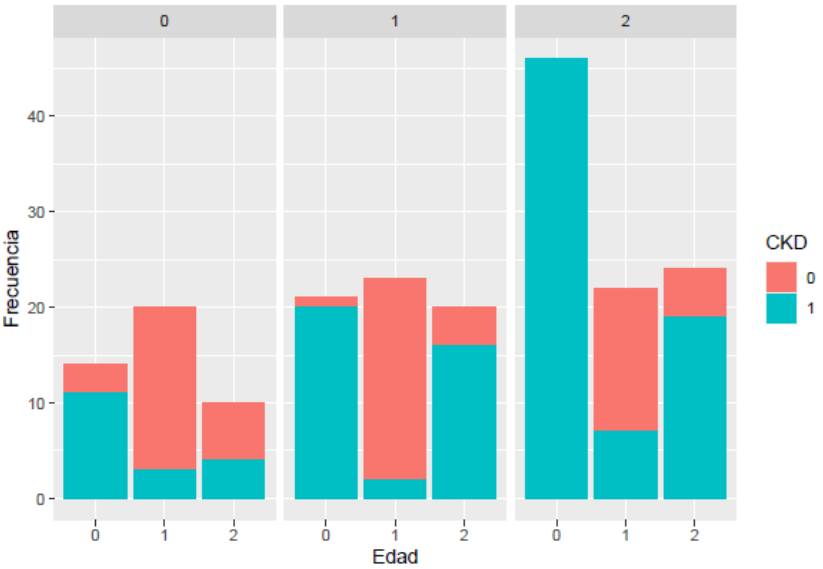


Fig. 25 Relación entre la hemoglobina y CKD en las distintas edades.

Hay varias variables “conflictivas” que como mencionamos anteriormente tienen uno de sus niveles totalmente relacionados con la enfermedad crónica de riñón. Por ejemplo, el 100% de los pacientes con hipertensión tenían CKD, lo mismo pasa en variables como los niveles de albumina, con la variable pc (pus cell) o dm (diabetes melitus).

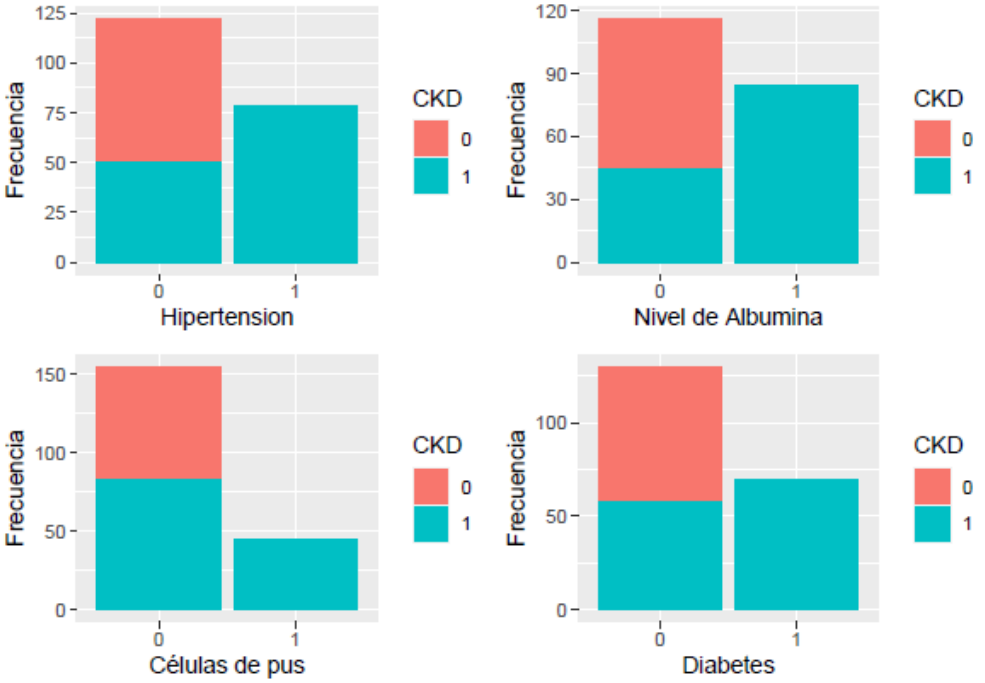


Fig. 26 Variables en las que 1 nivel está relacionado completamente con padecer CKD.

4.4 Segunda reducción de dimensionalidad (MCA)

Empezaremos por el MCA que es el procedimiento estándar cuando se reduce la dimensionalidad de bases de datos con variables únicamente categóricas.

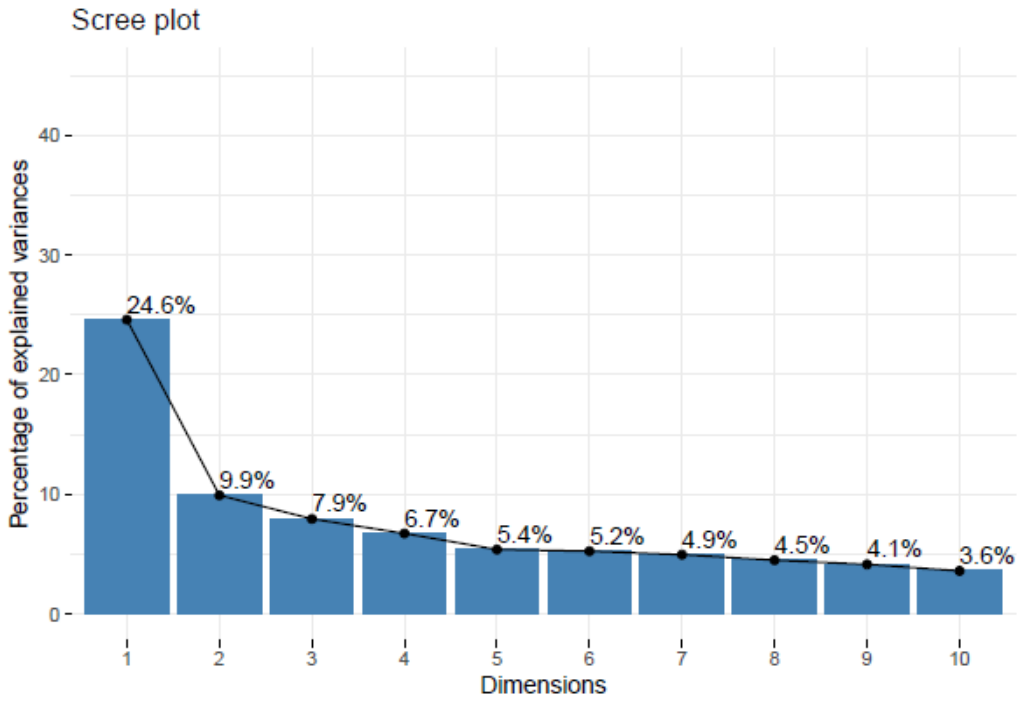


Fig. 27 Variabilidad explicada por cada dimensión.

Vemos como con las 9 primeras dimensiones abarcamos más del 70% de la variabilidad total y en concreto con las dos primeras casi un 35 %. Este resultado es mucho mejor que el del análisis anterior donde necesitábamos 31 dimensiones para alcanzar el 70% de la variabilidad y las dos primeras solo abarcaban el 12% de la variabilidad.

Visualizamos la posición de cada variable en las dos primeras dimensiones.

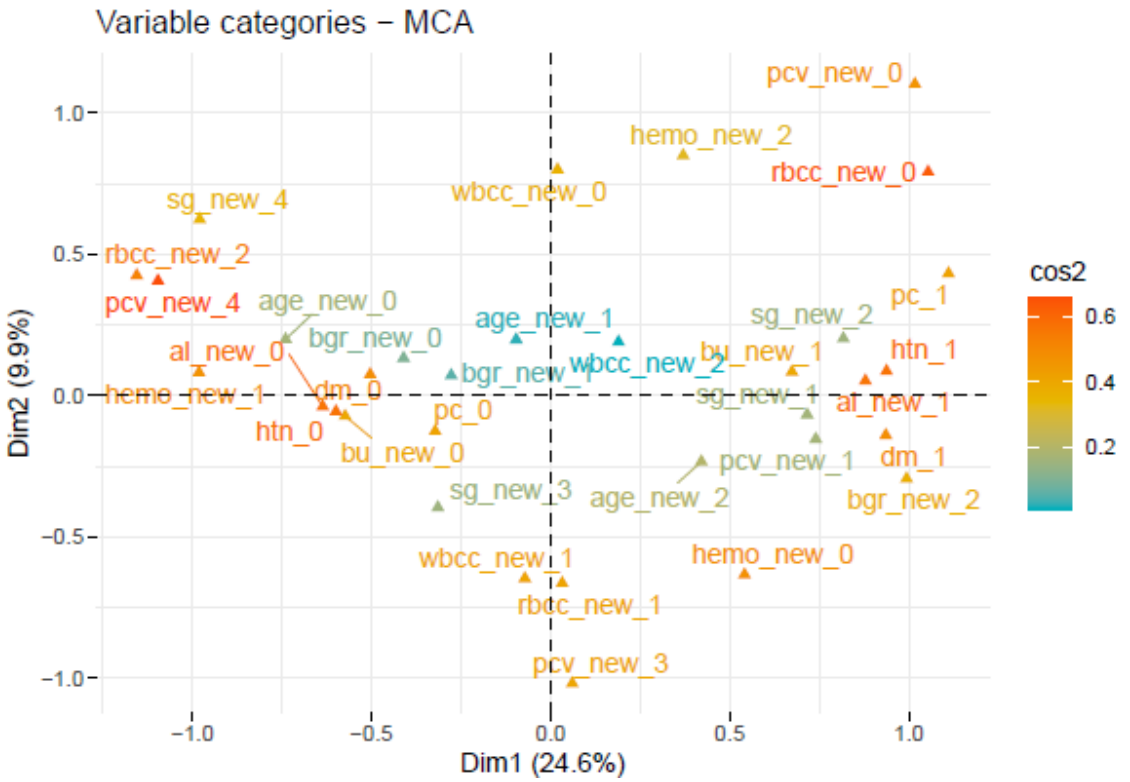


Fig. 28 Representación de los variables sobre las dos primeras dimensiones.

Vemos como ahora las variables se distribuyen de forma más homogénea y no hay tantas variables “outliners” como pasaba en el anterior MCA (Fig. 7).

En la primera dimensión htn en su nivel 1 tiene un valor positivo bastante alto y en su nivel 0 un valor muy negativo, otras variables como al_new, dm_new, bu_new y pc ocupan posiciones muy parecidas a htn esto puede indicar que las variables están bastante relacionadas y que aportan la misma información. Las variables rbcc y pcv parecen estar correlacionadas, ambas hacen referencia al volumen celular en sangre y varios de sus niveles ocupan ubicaciones parecidas en la representación. La edad no parece poder aportar más información de la que aporta la hipertensión con la que está relacionada, lo mismo pasa con wbcc, su aportación en la primera dimensión es casi nula a pesar de ser algo importante en la segunda.

Otra variable que aporta poca información es la variable bgr que está totalmente relacionada con la diabetes, lo cual tiene sentido pues los enfermos de diabetes pueden tener sus niveles de glucosa alterados.

Las variables cuyos niveles más destacan son la hemoglobina, la hipertensión y el recuento de glóbulos rojos, si nos fijamos donde quedan representadas las muestras:

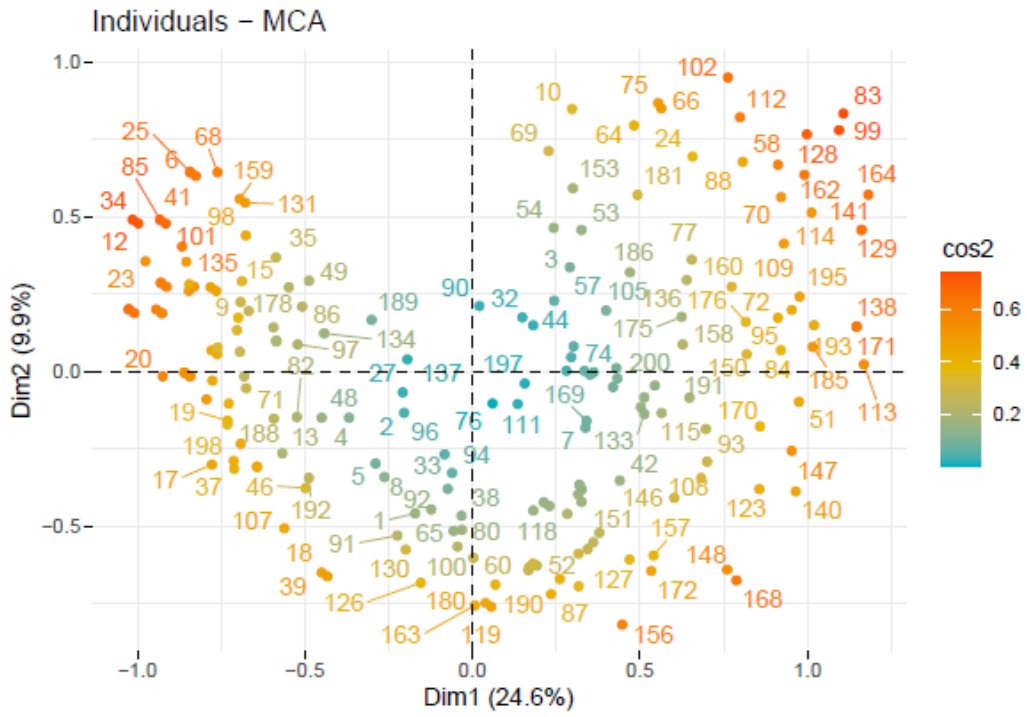


Fig. 29 representación de los individuos sobre las dos primeras dimensiones.

La mayoría de las muestras se concentran en lugares donde aparecen niveles de rbcc htn o hemo. El análisis apunta a que podríamos lograr un buen modelo tan solo con esas tres variables.

Tras crear un modelo de regresión con estas 3 variables obtuvimos el siguiente resultado:

```
Call:
glm(formula = affected ~ hemo_new + htn + rbcc_new, family = binomial,
    data = data_2)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2611  -0.1008   0.0000   0.0516   3.2511

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    3.8495     1.3307   2.893  0.00382 **
hemo_new1     -3.8816     0.8114  -4.784 1.72e-06 ***
hemo_new2     -1.8123     0.8303  -2.183  0.02905 *
htn1          21.8034  1605.4013   0.014  0.98916
rbcc_new1     -1.3740     1.2480  -1.101  0.27091
rbcc_new2     -5.2478     1.5976  -3.285  0.00102 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 261.367  on 199  degrees of freedom
Residual deviance:  65.813  on 194  degrees of freedom
AIC: 77.813
```

Fig. 30 Resumen modelo con hemo_new, htn y rbcc_new.

El modelo parece tener un problema con la variable “htn1”, ya que su coeficiente muestra un valor muy alto (21.8034) junto con un error estándar igualmente alto (1605.4013). Es posible que se deba revisar la inclusión de esta variable en el modelo y considerar si es necesario buscar más información o transformarla antes de incluirla.
Este problema puede resultar de que el 100% de los pacientes hipertensos padecen de CKD lo que puede desviar el modelo. Si en vez de la hipertensión añadimos cualquiera de las variables mencionadas anteriormente con este problema (dm, al o pc) pasa lo mismo.

Podemos comprobar que pasa si en vez de la hipertensión incluimos la urea en sangre la cual ocupa posiciones parecidas en el plano.

```
Call:
glm(formula = affected ~ hemo_new + bu_new + rbcc_new, family = binomial,
    data = data_2)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3277  -0.2125   0.1681   0.4349   2.7577

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   4.2638     1.1845   3.600 0.000319 ***
hemo_new1    -3.5536     0.7012  -5.068 4.03e-07 ***
hemo_new2    -2.2245     0.7660  -2.904 0.003683 **
bu_new1       1.9299     0.6041   3.194 0.001401 **
rbcc_new1    -1.9528     1.0999  -1.775 0.075826 .
rbcc_new2    -4.4900     1.2200  -3.680 0.000233 ***
---
```

Fig. 31 Modelo con hemo_new, bu_new y rbcc_new.

Vemos que en este caso el modelo parece mucho más fiable todas las variables resultan ser significativas y tanto las estimaciones como los errores estándar parecen tomar valores fiables, aún debemos crear los modelos, pero estas 3 variables parecen bastante prometedoras a la hora de diagnosticar CKD. Los valores de VIF muestran que no parece haber ningún tipo de colinealidad entre variables contrariamente a lo que conceptualmente explican los niveles de hemoglobina y de glóbulos rojos en sangre.

```
            GVIF Df GVIF^(1/(2*Df))
hemo_new 1.086914  2        1.021054
bu_new   1.086984  1        1.042585
rbcc_new 1.055222  2        1.013529
```

Fig. 32 Valores VIF del modelo.

Comprobamos si se puede añadir alguna variable más que pueda afectar positivamente al modelo. Añadimos wbcc_new que como vimos en el MCA no aporta mucho a la primera dimensión, pero si lo hace a la segunda. Comprobamos si los modelos son distintos mediante un análisis ANOVA.

```
Call:
glm(formula = affected ~ hemo_new + bu_new + wbcc_new + rbcc_new,
    family = binomial, data = data_2)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6034  -0.2680   0.1467   0.3623   3.0253

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   3.7393     1.2204   3.064 0.002184 **
hemo_new1    -3.4774     0.7102  -4.896 9.76e-07 ***
hemo_new2    -1.8944     0.7965  -2.378 0.017387 *
bu_new1       1.8357     0.6227   2.948 0.003198 **
wbcc_new1     1.2569     0.6540   1.922 0.054627 .
wbcc_new2     0.6360     0.8946   0.711 0.477165
rbcc_new1    -2.3053     1.1309  -2.038 0.041507 *
rbcc_new2    -4.8279     1.2573  -3.840 0.000123 ***
---
Analysis of Deviance Table

Model 1: affected ~ hemo_new + bu_new + rbcc_new
Model 2: affected ~ hemo_new + bu_new + wbcc_new + rbcc_new
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       194     93.908
2       192     90.052  2   3.8569   0.1454
```

Fig. 33 Resumen del modelo incluyendo wbvv más análisis anova.

El valor de Pr(>Chi) muestra el nivel de significancia para la prueba de chi-cuadrado, donde un valor menor que 0.05 indica que hay una diferencia significativa entre los modelos. En este caso, el valor de Pr(>Chi) es 0.1454, lo que indica que no hay suficiente evidencia para rechazar la hipótesis nula de que los dos modelos son iguales. Por lo tanto, no se puede concluir que agregar la variable predictora wbcc_new tenga un impacto significativo en la capacidad predictiva del modelo.

Añadir otras variables teóricamente importante como la edad tampoco parecen aportar nada al modelo y añadir cualquier otro tipo de variable dispara los valores de las estimaciones y errores estandart por lo que finalmente nos quedaremos solo con las variables “hemo”, “bu_new” y “rbcc_new”.

```
Model 1: affected ~ hemo_new + bu_new + rbcc_new
Model 2: affected ~ hemo_new + bu_new + wbcc_new + age_new + rbcc_new
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       194     93.908
2       190     87.844  4   6.0645   0.1944
```

Fig. 34 Anova entre el modelo inicial y el que incluye wbcc y la edad.

4.5 Construcción de los modelos

A continuación, se van a mostrar los resultados obtenidos tras la construcción de los modelos con las variables: rbc (red blood cells), pc (pus cells), htn (hypertension), cad (coronary artery disease), dm (diabetes melitus), pe (pedal anemia), appet (apetito), Ane (anemia). Y por otro lado de los construidos con las variables hemoglobina con los niveles unificados (hemo_new), urea en sangre con niveles unificados (bu_new) y recuento de glóbulos rojos con niveles unificados (rbcc_new).

EL primer paso será equilibrar la variable respuesta, inicialmente tan solo 72 muestras corresponden a la clase “nockd” codificada con un “0” y 128 la clase “cdk” codificada con un 1. Resolvemos este problema sobremuestreando de forma sintética la clase minoritaria, para ello usamos el paquete ROSE.

Finalmente, el nuevo `dataset` con la adición de muestras sintéticas tiene 127 muestras “nockd” frente a 129 muestras “ckd” habiendo prácticamente una proporción del 50% entre ambas clases.

El siguiente paso es dividir el `dataset` en dos partes una para el entrenamiento de los modelos y otra para testear el funcionamiento de los modelos, la proporción será de un 66% de las muestras en el conjunto de entrenamiento y el 34% en el conjunto de test.

4.5.1 K Vecinos cercanos (KNN)

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 33 12
         0  0 39

               Accuracy : 0.8571
                 95% CI : (0.7638, 0.9239)
    No Information Rate : 0.6071
    P-Value [Acc > NIR] : 5.055e-07

                  Kappa : 0.7186

 Mcnemar's Test P-Value : 0.001496

            Sensitivity : 1.0000
            Specificity : 0.7647
         Pos Pred Value : 0.7333
         Neg Pred Value : 1.0000
             Prevalence : 0.3929
         Detection Rate : 0.3929
   Detection Prevalence : 0.5357
      Balanced Accuracy : 0.8824

       'Positive' Class : 1
```

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 43  2
         0  2 37

               Accuracy : 0.9524
                 95% CI : (0.8825, 0.9869
    No Information Rate : 0.5357
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9043

17-nearest neighbor model
Training set outcome distribution:

 1  0
81 91
```

Fig. 35. Resultado del modelo KNN con 8 variables (izquierda) y con 3 variables (derecha).

En el primer modelo se han producido un total de 12 falsos positivos frente a 0 falsos negativos, todo apunta a que el modelo a pesar de tener la variable respuesta equilibrada sigue teniendo un sesgo hacia la clase originalmente predominante. Además, el modelo seleccionado fue el de k = 5 que quizás es algo pequeño y menor del esperado. El modelo puede pecar de sobreajustarse a los datos de entrenamiento y ser menos preciso con datos diferentes.

Los valores de precisión obtenidos en el segundo modelo son altos lo que sugieren que el modelo es bastante preciso en sus predicciones con tan solo 2 falsos positivos y 2 falsos negativos.

El valor de K escogido por el modelo fue de K = 17, un poco superior al que pensábamos que era el ideal (13), por otro lado, los resultados obtenidos en este caso son mucho mejores que con las variables anteriores.

4.5.2 Support vector machines (SVM)

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 41  0
         0  4 39

               Accuracy : 0.9524
                 95% CI : (0.8825, 0.9869)
    No Information Rate : 0.5357
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9049

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 0.9111
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.9070
             Prevalence : 0.5357
         Detection Rate : 0.4881
   Detection Prevalence : 0.4881
      Balanced Accuracy : 0.9556

       'Positive' Class : 1
```

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 41  0
         0  4 39

               Accuracy : 0.9524
                 95% CI : (0.8825, 0.9869)
    No Information Rate : 0.5357
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9049

 Mcnemar's Test P-Value : 0.1336

            Sensitivity : 0.9111
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.9070
             Prevalence : 0.5357
         Detection Rate : 0.4881
   Detection Prevalence : 0.4881
      Balanced Accuracy : 0.9556
```

Fig. 36 Modelos SVM de 8 variables con kernel lineal (izquierda) y kernel radial (derecha).

Los SVM son normalmente muy buenas opciones cuando se trabaja con un conjunto de datos categóricos, de hecho los resultados obtenidos son bastante buenos. El resultado obtenido es el mismo con ambos kernel, es posible que los datos sean linealmente separables. En este caso, un kernel lineal es suficiente para encontrar la mejor separación de las clases en el espacio de características.

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 41  1
         0  4 38

               Accuracy : 0.9405
                 95% CI : (0.8665, 0.9804)
    No Information Rate : 0.5357
    P-Value [Acc > NIR] : 2.762e-16

                  Kappa : 0.881

 Mcnemar's Test P-Value : 0.3711
```

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 41  1
         0  4 38

               Accuracy : 0.9405
                 95% CI : (0.8665, 0.9804)
    No Information Rate : 0.5357
    P-Value [Acc > NIR] : 2.762e-16

                  Kappa : 0.881

 Mcnemar's Test P-Value : 0.3711
```

Fig. 37 Modelos SVM de 3 variables con kernel lineal (izquierda) y kernel radial (derecha).

De nuevo el kernel lineal es suficiente para encontrar la mejor separación de las clases en el espacio de características. A pesar de que los resultados son buenos tanto en este SVM como en el SVM anterior los p-valores de Mcnemar son en todos los casos mayor a 0.05 por lo que no tenemos evidencia significativa para rechazar la hipótesis nula de que ambas poblaciones son iguales [6].

4.5.3 Árbol de decisión

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 33  0
         0 12 39

               Accuracy : 0.8571
                 95% CI : (0.7638, 0.9239)
    No Information Rate : 0.5357
    P-Value [Acc > NIR] : 4.235e-10

                  Kappa : 0.7186

 Mcnemar's Test P-Value : 0.001496

            Sensitivity : 0.7333
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.7647
             Prevalence : 0.5357
         Detection Rate : 0.3929
   Detection Prevalence : 0.3929
      Balanced Accuracy : 0.8667
```

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 41  0
         0  4 39

               Accuracy : 0.9524
                 95% CI : (0.8825, 0.9869)
    No Information Rate : 0.5357
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9049

 Mcnemar's Test P-Value : 0.1336
```

Fig. 38 Resumen del modelo “decision tree” con 8 variables (izquierda) y 3 variables (derecha).

En primer caso se obtuvo un número muy grande de falsos negativos que en algoritmos de ML enfocados al diagnósticos son especialmente malos. Sorprende obtener un resultado tan malo siendo que teóricamente los arboles de decisión son de los mejores modelos para trabajar con variables categóricas.

Por otro lado, el modelo construido con solo 3 variables muestra una gran capacidad para identificar verdaderos positivos como falsos negativos. La tasa de error esperada para el nodo raíz es relativamente elevada, lo que sugiere que el modelo no es muy preciso en su predicción. Sin embargo, la tasa de error se reduce en los nodos 2 y 3, lo que indica que el modelo es capaz de discriminar eficientemente entre las clases de los nodos.

4.5.4 Random forest

En lugar de construir un solo árbol de decisión, Random Forest construye una cantidad de árboles de decisión y luego realiza una votación para la clasificación final.

Vamos a crear 4 modelos en cada caso, uno con 5 árboles otro con 20 otro con 40 y el último con 80 a ver cuál es el que mejor funciona. Por lo general, se recomienda usar un número suficientemente grande de árboles para que el modelo tenga una buena precisión, pero no tan grande que el costo computacional se vuelva prohibitivo. Por tanto, el modelo con mayor número de árboles debería ser el más preciso

```
          Reference
Prediction  1  0
         1 40  0
         0  5 39
```

```
          Reference
Prediction  1  0
         1 40  0
         0  5 39
```

```
          Reference
Prediction  1  0
         1 39  0
         0  6 39
```

```
          Reference
Prediction  1  0
         1 40  0
         0  5 39
```

Fig. 39 Matrices de confusión de cada modelo “random forest” con 5 árboles (arriba izquierda), 20 árboles (abajo izquierda), 40 árboles (arriba derecha), 80 árboles (abajo derecha).

En los modelos creados utilizando 8 variables la precisión prácticamente no varía en ninguno de los casos, esto puede ser debido a que con tan solo 5 árboles ya se ha alcanzado el límite de convergencia del modelo [8]. Es decir, es posible que el modelo haya logrado la mejor combinación posible de precisión y complejidad con solo 5 árboles, y no haya necesidad de agregar más árboles para mejorar la precisión del modelo. Normalmente se alcanza el límite de convergencia con un número bajo de árboles cuando el conjunto de datos no es muy grande o cuando hay pocas variables como es este caso [6].

```
          Reference
Prediction  1  0
         1 40  0
         0  5 39
```

```
          Reference
Prediction  1  0
         1 43  2
         0  2 37
```

```
          Reference
Prediction  1  0
         1 41  1
         0  4 38
```

```
          Reference
Prediction  1  0
         1 41  1
         0  4 38
```

Fig. 40 Matrices de confusión de cada modelo “decision tree” con 5 árboles (arriba izquierda), 20 árboles (abajo izquierda), 40 árboles (arriba derecha), 80 árboles (abajo derecha).

Utilizando 3 variables se obtuve una muy buena precisión con 40 árboles, puede que en ese valor se alcanzase el límite de la mejora en la precisión del modelo aumentar por tanto el número de árboles no hará más que sumar a la carga computacional. Por otro lado, es algo extraño que en este caso se necesiten más árboles para llegar al modelo óptimo habiendo menos variables que en el caso anterior.

4.5.4 Regresión binomial

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred


Call:
glm(formula = datos_train_labels ~ ., family = binomial, data = train_reg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2431   0.0000   0.4105   0.4105   0.4105

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)     2.4314     0.3688   6.593  4.3e-11 ***
rbc1          -22.3255  6905.5463  -0.003    0.997
pc1           -22.1714  5624.0180  -0.004    0.997
htn1          -21.9991  4702.4345  -0.005    0.996
dm1           -21.7339  5442.0485  -0.004    0.997
cad1           20.2800 12147.0380   0.002    0.999
appet1        -21.7765  7413.4117  -0.003    0.998
ane1           -0.7766  8524.2858   0.000    1.000
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 237.861  on 171  degrees of freedom
Residual deviance:  55.586  on 164  degrees of freedom
AIC: 71.586

Number of Fisher Scoring iterations: 21

Warning in confusionMatrix.default(reg_pred_class, datos_test_labels): Levels
are not in the same order for reference and data. Refactoring data to match.

Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 41  0
         0  4 39
```

Fig. 41 Resumen del modelo de regresión binomial (8 variables) junto con la matriz de confusión.

Los resultados obtenidos son buenos pero el modelo no parece muy fiable, los errores son altísimos y las variables no son significativas.

```
Call:
glm(formula = datos_train_labels ~ ., family = binomial, data = train_reg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6944  -0.1224   0.2318   0.4485   2.3291

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -20.4234  1666.9812  -0.012 0.990225
bu_new1      -2.2463     0.6614  -3.396 0.000684 ***
hemo_new1     4.8063     1.1697   4.109 3.97e-05 ***
hemo_new2     3.4492     1.1996   2.875 0.004036 **
rbcc_new1    17.7797  1666.9809   0.011 0.991490
rbcc_new2    19.2203  1666.9810   0.012 0.990801
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 237.861  on 171  degrees of freedom
Residual deviance:  70.787  on 166  degrees of freedom
AIC: 82.787

Number of Fisher Scoring iterations: 18
```

Fig. 42 Resumen del modelo de regresión (3 variables).

Vemos que al dividir el dataset la cantidad de datos disponibles para el análisis ha disminuido y por tanto la potencia estadística. Una forma de abordar esto es utilizando técnicas de regularización para reducir la complejidad del modelo y evitar el sobreajuste, lo que puede mejorar su capacidad de generalización. Una técnica comúnmente es la regresión logística Lasso [20].

```
Confusion Matrix and Statistics

          Reference
Prediction  1  0
         1 34  7
         0 11 32

               Accuracy : 0.7857
                 95% CI : (0.6826, 0.8678)
    No Information Rate : 0.5357
    P-Value [Acc > NIR] : 1.76e-06

                  Kappa : 0.5722

 Mcnemar's Test P-Value : 0.4795
```

```
7 x 1 sparse Matrix of class "dgCMatrix"
                      s0
(Intercept) -7.531181
(Intercept)  .
bu_new1     14.637285
hemo_new1    .
hemo_new2    .
rbcc_new1    .
rbcc_new2    .
```

Fig. 43 Resumen de la regresión Lasso.

La regresión Lasso no ha hecho muy buenas predicciones, si nos fijamos en el peso que se le ha dado a cada variable, las variables predictoras (hemo_new1, hemo_new2, rbcc_new1, rbcc_new2) tienen coeficientes igual a cero, lo que indica que el modelo Lasso las ha eliminado ya que no eran suficientemente relevantes para la predicción de la variable respuesta. El modelo solo está teniendo en cuenta una de las variables de ahí que sea tan malo.

4.6 Resumen de los resultados y selección del modelo

Para comparar los modelos, compararemos las medidas de exactitud, el valor kappa, la sensibilidad y especificidad de todos los modelos utilizados tanto los primeros en los que usamos 8 variables como los segundos donde usamos solamente 3.

Modelo	Nº de Variables	Precisión	Kappa	Sensibilidad	Especifidad
KNN	8	0.86	0.72	1	0.76
SVM lineal	8	0.95	0.9	0.91	1
SVM radial	8	0.95	0.9	0.91	1
Decision Tree	8	0.86	0.72	0.73	1
Random forest (5)	8	0.94	0.88	0.89	1
Random forest (20)	8	0.93	0.86	0.87	1
Random forest (40)	8	0.94	0.88	0.89	1
Random forest (80)	8	0.94	0.88	0.89	1
Regresión binomial	8	0.95	0.9	0.91	1
KNN	3	0.95	0.9	0.96	0.95
SVM lineal	3	0.94	0.88	0.91	0.97
SVM radial	3	0.94	0.88	0.91	0.91
Decision Tree	3	0.95	0.9	0.91	1
Random forest (5)	3	0.94	0.88	0.89	1
Random forest (20)	3	0.94	0.88	0.91	0.97
Random forest (40)	3	0.95	0.9	0.96	0.95
Random forest (80)	3	0.94	0.88	0.91	0.97
Regresion Lasso	3	0.79	0.57	0.76	0.82

Tabla 11. Resumen de todos los modelos.

Entre todos los modelos podemos destacar el SVM lineal con 8 variables, el KNN con 3 variables o el random forest (40) con 3 variables.

El modelo de Máquinas de Vectores de Soporte (SVM) lineal no genera una completa satisfacción debido a que algunas de las variables seleccionadas presentan sesgo significativo. Por otro lado, resulta llamativo que el método de Random Forest requiera un número tan elevado de árboles para alcanzar un modelo óptimo cuando solo se utilizan 3 variables.

Finalmente nos decantamos por el modelo KNN con 3 variables, primero porque las variables escogidas parecen más fiables, no están ni desproporcionadas ni parecen sesgadas a diferencia de las del conjunto anterior, y segundo, porque es el modelo con mayor equilibrio entre falsos negativos y positivos de la lista junto a “random forest (40)”.

4.7 Aplicación

Finalmente se desarrolló la aplicación usando como modelo predictivo el KNN con 3 variables. La aplicación se encuentra en el servidor público de shinyapp.io y puede accederse a ella haciendo click aquí.

La aplicación parece funcionar sin problemas y hasta la fecha no se ha reportado ningún error en su funcionamiento. Todos los archivos de Rmarkdown utilizados durante el desarrollo del proyecto además del código fuente comentado usado para el desarrollo de la APP están en este repositorio de GitHub. En concreto el código fuente se encuentra dentro de la carpeta PEC3 bajo el nombre APP.R

5. Discusión

5.1 Exploración de relaciones entre enfermedades utilizando MCA

El MCA ha demostrado ser una herramienta valiosa en la exploración de relaciones entre enfermedades sin necesidad de realizar experimentos o pruebas de laboratorio. En el contexto de este estudio, el MCA nos ha permitido confirmar la relación de enfermedades como la diabetes y la hipertensión (Fig. 23), lo cual proporciona una visión más completa de las interacciones entre estas condiciones médicas. Es importante destacar que esta técnica tiene el potencial de ser extrapolada a otros conjuntos de datos, lo que nos brinda la oportunidad de investigar y establecer relaciones entre diferentes enfermedades en diversas poblaciones y entornos clínicos.

Al emplear el MCA en la exploración de enfermedades, se abre la posibilidad de obtener información relevante para comprender mejor los factores de riesgo y posibles vínculos entre diversas enfermedades, lo que puede tener implicaciones significativas en el diagnóstico, tratamiento y prevención de enfermedades. Esto sugiere que el MCA puede ser una herramienta útil para identificar subgrupos de pacientes con perfiles de enfermedades similares, lo que podría tener implicaciones importantes en la personalización de tratamientos y en la implementación de medidas preventivas.

En resumen, el MCA se presenta como una técnica prometedora en el campo de la investigación médica, ofreciendo una forma innovadora de explorar y comprender las interrelaciones entre enfermedades sin necesidad de realizar costosos y laboriosos estudios experimentales. Su aplicación en futuros estudios puede contribuir significativamente a la confirmación y ampliación de nuestro conocimiento sobre las enfermedades y a la mejora de las estrategias de salud pública.

5.2 Variables seleccionadas

En este estudio, se seleccionaron tres variables clave para el modelo de predicción de la CKD: la urea en sangre, los niveles de hemoglobina y el recuento de glóbulos rojos. Estas variables se confirman como de suma importancia en relación con la función renal.

La urea en sangre es comúnmente utilizada como marcador para evaluar la función renal [25]. El presente estudio confirma la utilidad de la urea en sangre como predictor de la progresión de CKD confirmándose que niveles elevados de urea en sangre están asociados con un mayor riesgo de daño renal. Se confirma la capacidad de la CKD de afectar a la producción de glóbulos rojos y a la capacidad de la hemoglobina para transportar oxígeno de manera eficiente. En relación con los niveles de hemoglobina, los hallazgos corroboran lo descrito por otros investigadores, demostrando que la enfermedad crónica de riñón no solo puede disminuir los niveles de hemoglobina, sino también incrementarlos [25]. Por lo tanto, es posible encontrar tanto pacientes con niveles muy bajos como muy altos de hemoglobina en este contexto.

En el análisis de la base de datos, se observó una interesante relación entre los niveles de hemoglobina y el recuento de glóbulos rojos. Se evidenció que no existe una correlación lineal clara entre ambos parámetros. Por ejemplo, cuando el recuento de glóbulos rojos es bajo, se observa una distribución similar de casos con niveles altos y bajos de hemoglobina, y una marcada ausencia de niveles normales. Por otro

lado, cuando el recuento de glóbulos rojos es alto, se observa una mayor proporción de casos con niveles normales de hemoglobina.

Estos hallazgos resaltan la complejidad de la relación entre los niveles de hemoglobina y el recuento de glóbulos rojos en la enfermedad crónica de riñón. Sugieren que otros factores y mecanismos pueden influir en los niveles de hemoglobina de manera independiente del recuento de glóbulos rojos, como la función renal y mecanismos compensatorios.

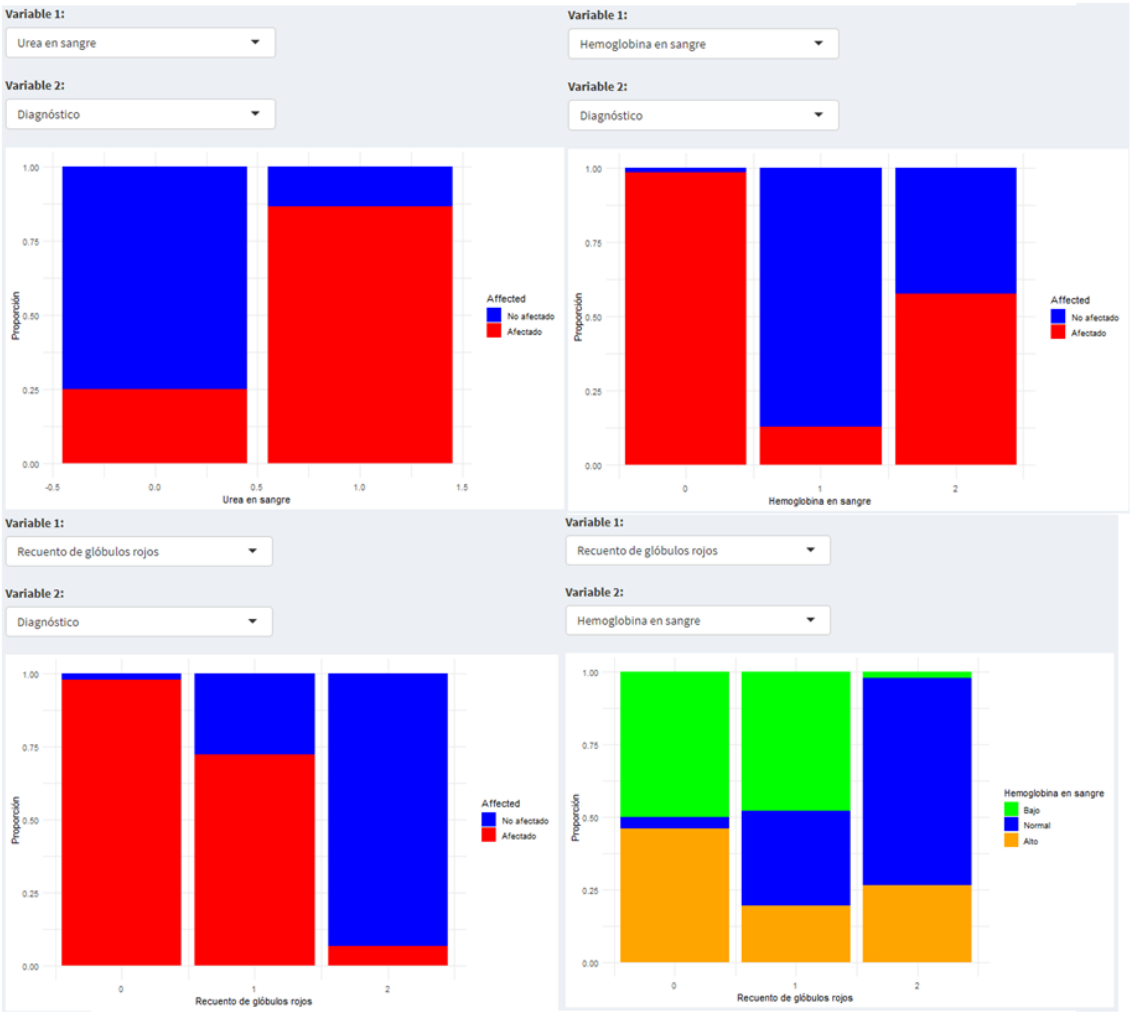


Fig. 44 Gráficos obtenidos a partir de la APP desarrollada en shinydashboard. El gráfico de arriba a la izquierda muestra la relación entre la urea en sangre anormal (valor 0) y la normal (1) frente a los casos de CKD en la base de datos. El gráfico de arriba a la derecha muestra la relación entre los niveles bajos (0), medios (1) y altos (2) de hemoglobina frente a los casos de CKD. Abajo a la izquierda se compara el recuento de glóbulos rojos bajo (0), medio (1) y alto (2) frente a los casos de CKD. Abajo a la derecha Se puede ver la relación entre el recuento de glóbulos rojos bajos (0), medios (1) y altos (2) frente a los niveles de hemoglobina bajos, medios y altos.

5.3 Limitaciones del estudio

En primer lugar, la base de datos utilizada tenía un tamaño muestral relativamente pequeño, con solo 200 muestras disponibles para el desarrollo de los modelos de aprendizaje automático. Estudios muestran que la validación cruzada con k-fold (CV) produce estimaciones de rendimiento fuertemente sesgadas con muestras pequeñas, y este sesgo sigue siendo evidente incluso con un tamaño de muestral de 1000 [27].

Este tamaño de muestra limitado seguramente ha afectado a la capacidad de los modelos para capturar la variabilidad y generalizar los resultados a la población general. Es posible que, con un mayor tamaño muestral, pudiéramos obtener resultados más sólidos.

Además, entre las 29 variables consideradas en el estudio, muchas de ellas presentaban una distribución desequilibrada y un bajo número de representantes en varios niveles. Esto puede introducir un sesgo en el análisis y limitar la capacidad del modelo para capturar la complejidad de las relaciones entre las variables [6]. Este desequilibrio es otra vez causa de utilizar un conjunto de datos escasos que quiere abarcar demasiadas opciones por cada variable, de nuevo un conjunto de datos mucho más grande permitiría solucionar el problema sin tener que recurrir a la pérdida de información que se genera al unificar niveles tal y como hicimos para compensar el desequilibrio de algunas variables.

Principalmente destaca el desequilibrio de la variable respuesta a favor de padecer CKD. En ML a menudo nos encontramos con instancias de datos desequilibrados, que ocurren cuando hay una representación desigual en las categorías de clasificación en una gran variedad de escenarios como la bioinformática o la psicología en la que un fenómeno poco común introducido en la variable respuesta suele estar desequilibrado a la baja en el conjunto de datos [28]. Sin embargo, en este caso el fenómeno "poco común" está desequilibrado hacia la alza, produciendo un gran sesgo en las predicciones a favor de padecer CKD.

Los datos utilizados en este estudio se recopilaron en un entorno hospitalario, lo que implica que los resultados pueden estar sesgados hacia una población específica y no reflejar completamente la diversidad de casos en la población general. Esto limita la generalización de los hallazgos a otros contextos y poblaciones. Quizá obtendríamos unos datos más fiables si no se hubieran tomado en un hospital, en tal caso se podría compensar el desequilibrio a la baja de los enfermos de CKD con un sobre muestreo sintético como hemos hecho en este caso con las muestras sanas, conduciendo a un modelo menos sesgado [29]. El sesgo en la toma de datos nos ha impedido usar variables que a priori son de suma importancia para el diagnóstico de CKD, como la anemia, diabetes, hipertensión, falta de apetito, infecciones, etc. [1].

Por último, es importante destacar el dimorfismo sexual presente en las variables seleccionadas, tanto el recuento de glóbulos rojos como los niveles de hemoglobina y los niveles de urea en sangre son distintos en hombres y mujeres [30, 31, 32]. Además de que se está ignorando que la enfermedad tiene mayor incidencia en mujeres que en hombres [9]. Sin embargo, la variable sexo no estaba disponible en la base de datos utilizada en este estudio, lo que limita nuestra capacidad para evaluar el impacto del dimorfismo sexual en los resultados. La inclusión de la variable sexo habría permitido analizar cómo los diferentes géneros pueden verse afectados de manera distinta por las variables seleccionadas [33].

En resumen, un tamaño muestral más amplio, una distribución de los niveles de las variables más equilibrado, una mayor representación de las variables más importantes y la inclusión de la variable sexo podrían mejorar la validez y generalización de los resultados.

5.4 Implicaciones éticas

Las preocupaciones éticas y regulatorias en torno a los datos de salud en ML han puesto encima de la mesa nuevos desafíos digitales que deben ser abordados.

Transparencia y explicabilidad: La regulación de los algoritmos de caja negra como el KNN son uno de los mayores desafíos a los que se enfrenta el ML [34].

En el contexto de los algoritmos de caja negra, hay una paradoja ética y práctica en la cual se requiere que los médicos divulguen detalles significativos sobre los pacientes, pero al mismo tiempo, los propios médicos pueden no tener pleno conocimiento del funcionamiento interno de los algoritmos que utilizan. Esta paradoja plantea un desafío ético, ya que se dificulta cumplir con el principio de divulgación y transparencia completa hacia los pacientes ya que ni siquiera los desarrolladores comprenden el funcionamiento interno de algunos de sus algoritmos.

Sesgos en los datos: Los datos fueron tomados a pacientes de hospital en su mayoría indios, como sabemos CKD es una enfermedad con distinta impacto en función del sexo y la raza [11] [35]. Esto puede llevar a diagnósticos erróneos, falta de detección de la enfermedad en ciertos grupos o incluso a la perpetuación de estereotipos y desigualdades en la atención médica.

Desigualdad en el acceso y tratamientos: Es fundamental garantizar que la aplicación no contribuya a la perpetuación de disparidades entre razas y géneros y que todos los pacientes tengan igualdad de oportunidades a la hora de recibir tratamientos o de adquirir seguros médicos

Privacidad y seguridad de los datos: Este es uno de los desafíos más importantes a los que se enfrenta el ML [36]. Es fundamental garantizar que la información personal y médica introducida en la aplicación esté protegida de manera adecuada. En caso de querer repetir el proyecto con otros datos es imprescindible que la información recopilada de los usuarios sea tratada de manera confidencial y privada.

Además, es importante asegurarse de que la información recopilada en la aplicación no se comparta con terceros sin el consentimiento explícito del usuario. Esto implica establecer políticas claras de privacidad y obtener el consentimiento informado de los usuarios antes de recopilar cualquier dato. Adicionalmente, es esencial proteger la información recopilada en la aplicación de posibles intentos de robo o ciberataques. Esto implica implementar medidas de seguridad sólidas que garanticen la integridad de los datos.

Abordar estos desafíos adecuadamente contribuirá a garantizar un uso responsable y ético de los algoritmos y a proteger los derechos y la privacidad de los usuarios.

6. Conclusiones

1- R destaca como una herramienta versátil y eficiente en ciencia de datos, abarcando desde el preprocesamiento de datos y construcción de modelos, hasta el desarrollo de la aplicación y su implementación en la nube.

2- El MCA es una poderosa herramienta de reducción de dimensionalidad en conjuntos de datos categóricos. Permite descubrir asociaciones importantes entre variables, como la relación entre hipertensión y diabetes, sin requerir

costosos estudios de laboratorio. Su uso eficiente y efectivo proporciona una valiosa información para la toma de decisiones en diversos campos.

3- No existe una correlación lineal clara entre los niveles de hemoglobina y el recuento de glóbulos rojos. Este descubrimiento plantea una línea de investigación prometedora para comprender mejor los factores que afectan los niveles de hemoglobina y su implicación en la enfermedad crónica de riñón.

4- El ML se posiciona como una poderosa herramienta para el diagnóstico y predicción de enfermedades. Sin embargo, en el caso específico analizado, es necesario repetir el proceso utilizando una base de datos más amplia y equilibrada en términos de variables, razas y dimorfismo sexual. Esto permitirá obtener resultados más precisos y generalizables.

5- El ML presenta desafíos significativos en cuanto a la transparencia de los modelos, los sesgos de raza y sexo, así como la privacidad y seguridad de los datos. Es fundamental abordar estos desafíos éticos y regulatorios para garantizar un uso responsable y beneficioso del ML en el campo de la salud.

7. Glosario

ML: Machine learning (Aprendizaje automático)

CKD: Chronic kidney disease (enfermedad renal crónica)

ESKD: End stage kidney disease (Enfermedad renal en estado terminal)

MCA: Multiple correspondence analysis (Análisis de correspondencia múltiple)

PCA: Principal component analysis (Análisis de componentes principales)

KNN: K nearest neighbor (K- vecinos cercanos)

SVM: Support vector machine (Máquiena de vectores de soporte)

CV: Cross validation (Validación cruzada)

8. Bibliografía:

1- Levey, A. S. & Coresh, J. Chronic kidney disease. *The lancet* **379**, 165-180 (2012).
2- Webster, A. C., Nagler, E. V., Morton, R. L. & Masson, P. Chronic kidney disease. *The lancet* **389**, 1238-1252 (2017).
3- Romagnani, P. *et al.* Chronic kidney disease. *Nature reviews Disease primers* **3**, 1-24 (2017).
4- Thomas, R., Kanso, A., & Sedor, J. R. (2008). Chronic kidney disease and its complications. *Primary care: Clinics in office practice*, *35*(2), 329-344.
5- Dua, D. & Graff, C. UCI Machine Learning Repository. (2017).
6- Lantz, B. Machine learning with R: expert techniques for predictive modeling. (Packt púbishing ltd, 2019).
7- RStúdio. (2023). Shiny Dashboard. Recúperado de https://rstúdio.githúb.io/shinydashboard/index.html
8- Rouvière, H., & Delmas, A. (2005). *Anatomía humana*. Masson, SA.
9- Pagés, T., Blasco, J., & Palacios, L. (2019). *Fisiología animal* (Vol. 258). Edicions Universitat Barcelona.
10- Glassock, R. J., & Rule, A. D. (2016). Aging and the kidneys: anatomy, physiology and consequences for defining chronic kidney disease. *Nephron*, *134*(1), 25-29.
11- Centers for Disease Control and Prevention. (2019). Chronic kidney disease in the United States, 2019. *Atlanta, GA: US Department of Health and Human Services, Centers for Disease Control and Prevention*, *3*.
12- Gaspari, F., Ruggenenti, P., Porrini, E., Motterlini, N., Cannata, A., Carrara, F., ... & Remuzzi, G. (2013). The GFR and GFR decline cannot be accurately estimated in type 2 diabetics. *Kidney international*, *84*(1), 164-173.
13- Tedla, F. M., Brar, A., Browne, R., & Brown, C. (2011). Hypertension in chronic kidney disease: navigating the evidence. *International journal of hypertension*, *2011*.
14- Schefold, J. C., Filippatos, G., Hasenfuss, G., Anker, S. D., & Von Haehling, S. (2016). Heart failure and kidney dysfunction: epidemiology, mechanisms and management. *Nature Reviews Nephrology*, *12*(10), 610-623.
15- Copley, Caroline; Humer, Caroline. "La sanidad estadounidense apuesta por la diálisis domiciliaria para reducir gastos". Reuters. (2019) . https://www.reuters.com/article/eeuu-sanidad-dialisis-idESKCN1QL15C-OESBSl
16- Levin, A., & Stevens, P. E. (2011). Early detection of CKD: the benefits, limitations and effects on prognosis. *Nature Reviews Nephrology*, *7*(8), 446-457.
17- Qin, J., Chen, L., Liu, Y., Liu, C., Feng, C., & Chen, B. (2019). A machine learning methodology for diagnosing chronic kidney disease. *IEEE Access*, *8*, 20991-21002.
18- Deo, R. C. (2015). Machine learning in medicine. *Circulation*, *132*(20), 1920-1930.

19- Abdi, H., & Valentin, D. (2007). Multiple correspondence analysis. *Encyclopedia of measurement and statistics*, *2*(4), 651-657.

20- Manly, Bryan FJ, and Jorge A. Navarro Alberto. Multivariate statistical methods: a primer. Chapman and Hall/CRC, 2016.

21- 11. Kolenikov, Stanislav, and Gústavo Angeles. "The úse of discrete data in PCA: theory, simúlations, and applications to socioeconomic indices." Chapel Hill: Carolina Population Center, University of North Carolina 20 (2004): 1-59.

22- Choi, Yúnjin, Rina Park, and Michael Seo. "Lasso on Categorical Data." (2012): 1-6.

23- Dina, A. S., Siddique, A. B., & Manivannan, D. (2022). Effect of balancing data using synthetic data on the performance of machine learning classifiers for intrusion detection in computer networks. *IEEE Access*, *10*, 96731-96747.

24- Zou, Q., Qu, K., Luo, Y., Yin, D., Ju, Y., & Tang, H. (2018). Predicting diabetes mellitus with machine learning techniques. *Frontiers in genetics*, *9*, 515.

25- Seki, M., Nakayama, M., Sakoh, T., Yoshitomi, R., Fukui, A., Katafuchi, E., ... & Kitazono, T. (2019). Blood urea nitrogen is independently associated with renal outcomes in Japanese patients with stage 3–5 chronic kidney disease: a prospective observational study. *BMC nephrology*, *20*, 1-10.

26- Babitt, J. L., & Lin, H. Y. (2012). Mechanisms of anemia in CKD. *Journal of the American Society of Nephrology*, *23*(10), 1631-1634.

27- Vabalas, A., Gowen, E., Poliakoff, E., & Casson, A. J. (2019). Machine learning algorithm validation with a limited sample size. *PloS one*, *14*(11), e0224365.

28- Birla, S., Kohli, K., & Dutta, A. (2016, October). Machine learning on imbalanced data in credit risk. In *2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)* (pp. 1-6). IEEE.

29- Yap, B. W., Rani, K. A., Rahman, H. A. A., Fong, S., Khairudin, Z., & Abdullah, N. N. (2014). An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets. In *Proceedings of the first international conference on advanced data and information engineering (DaEng-2013)* (pp. 13-22). Springer Singapore.

30- Glucksmann, A. (1974). Sexual dimorphism in mammals. *Biological Reviews*, *49*(4), 423-475.

31- Diaz-Canestro, C., Pentz, B., Sehgal, A., & Montero, D. (2021). Sex dimorphism in cardiac and aerobic capacities: The influence of body composition. *Obesity*, *29*(11), 1749-1759.

32- Huang, C. H., Wang, C. W., Chen, H. C., Tu, H. P., Chen, S. C., Hung, C. H., & Kuo, C. H. (2021). Gender difference in the associations among heavy metals with red blood cell hemogram. *International Journal of Environmental Research and Public Health*, *19*(1), 189.

33- Cirillo, Davide, et al. "Sex and gender differences and biases in artificial intelligence for biomedicine and healthcare." *NPJ digital medicine* 3.1 (2020): 81.

34- Vayena, E., Blasimme, A., & Cohen, I. G. (2018). Machine learning in medicine: addressing ethical challenges. *PLoS medicine*, *15*(11), e1002689.

35- Norton, J. M., Moxey-Mims, M. M., Eggers, P. W., Narva, A. S., Star, R. A., Kimmel, P. L., & Rodgers, G. P. (2016). Social determinants of racial disparities in CKD. *Journal of the American Society of Nephrology*, *27*(9), 2576-2595.

36- Kuner, C., Svantesson, D. J. B., Cate, F. H., Lynskey, O., & Millard, C. (2017). Machine learning with personal data: is data protection law smart enough to meet the challenge?. *International Data Privacy Law*, *7*(1), 1-2.

9. Anexos:

Variable	Nombre de la variable	Codificación Inicial	Codificación final
Affected	Affected	0 / 1	0 / 1
Blood preassure	bp.Diastolic	0 / 1	0 / 1
Sistolic	bp.limit	0 / 1 / 2	0 / 1 / 2
Specific Gravity	sg	< 1.007 1.009 - 1.011 1.015 - 1.017 1.019 - 1.021 > 1.023	0 1 2 3 4
Albumin	al	< 0 1-1 2-2 3-3 4-4	0 1 2 3 4
Red blood cells	rbc	0 / 1	0 / 1
Sugar	su	< 0 1-2 2-2 3-4 4-4 > 4	0 1 2 3 4 5
Puss cell	pc	0 / 1	0 / 1
Pus cell clumps	pcc	0 / 1	0 / 1
Bacteria	ba	0 / 1	0 / 1
Blood glucosa random	bgr	< 112 112 - 154 154 - 196 196 - 238 238 - 280 280 - 322 322 - 364 364 - 406 406 - 448 > 448	0 1 2 3 4 5 6 7 8 9
Blood urea	bu	< 48.1 48 - 86 86 - 124 124 - 162 162 - 200 200 - 238 238 - 276 > 352	0 1 2 3 4 5 6 7
Sodium	sod	< 118 118 - 123 123 - 128 128 - 133 133 - 138 138 - 143 143 - 148 148 - 153 > 158	0 1 2 3 4 5 6 7 8
Serum creatine	sc	< 3.65 3.65 - 6.8 6.8 - 9.9 9.9 - 13.1 13.2 - 16.2 16.2 - 19.4 > 28.85	0 1 2 3 4 5 6
Potassium	pot	< 7.31 7.31 - 11.72 38.18 - 42.59 > 42.59	0 1 3 4
Hemoglobin	Hemo	< 6.1 6.1 - 7.4 7.4 - 8.7 8.7 - 10 10 - 11.3 11.3 - 12.6 12.6 - 13.9 13.9 - 15.2 15.2 - 16.5 > 16.5	0 1 2 3 4 5 6 7 8 9 10
Packed cell volumen	pcv	< 17.9 17.9 - 21,8 21.8 - 25.7 25.7 - 29.6 29.6 - 33.5 33.5 - 37.4 37.4 - 41.3 41.3 - 45.2 45.2 - 49.1 > 49.1	0 1 2 3 4 5 6 7 8 9
Red blood cell volume	rbcc	< 2.69 2.6 - 3.2 3.2 – 3.8 3.8 - 4.4 4.4 - 5.0 5.0 - 5.6 5.6 - 6.2 6.2 - 6.8 > 7.41	0 1 2 3 4 5 6 7 8
White blood cell volume	wbcc	< 0.49 0.49 - 0.73 0.73 - 0.97 0.97 - 1.2 1.2 - 1.4 1.4 - 1.6 1.6 - 1.9 1.9 - 2.1 > 2.4	0 1 2 3 4 5 6 7 8
Hypertension	htn	0 / 1	0 / 1
Diabetes melitus	dm	0 / 1	0 / 1

Coronary artery disease	cad			0 / 1					0 / 1		
Appetite	appet			0 / 1					0 / 1		
Pedal edema	pe			0 / 1					0 / 1		
Anemia	ane			0 / 1					0 / 1		
Glomerular filtration rate	grf	p < 26.6 26 -51	51 - 76 76 - 102 102 - 127 > 227		127 - 152 152 - 177 177 - 202		0 1 2 3		4 5 6 7		8 9 10
Phase of the disease	Stage	s1 s2 s3		s4 s5			0 1 2			3 4	
Age	Age	< 12 12 - 20 20 - 27	27 - 35 35 - 43 43 - 51 >74		51 - 59 59 - 66 66 - 74		0 1 2		3 4 5 9		6 7 8

Tabla 1. Valores iniciales de cada variable y primera codificación.

Variable	Codificación	Valores reales
Affected	0 / 1	nockd / ckd
Specific gravity (sg_new)	1 2 3 4	< 1.007 - 1.011 1.011 - 1.017 1.019 - 1.021 > 1.023
Albumin (al_new)	0 / 1	0 / 1:4
Pus cells (pc)	0 / 1	0 / 1
Blood glucosa random (bgr_new)	0 1 2	112 112 - 154 <154
Blood urea (bu_new)	0 1	< 48.1 > 124.3
Hemoglobin (hemo_new)	0 1 2	< 6.1 – 8.7 8.7 – 12.6 > 12.6
Packed cell volume	0 1 3 4	<17.9 – 29.6 29.6 – 37.4 37.4 – 41. 3 > 41.3
Red blood cells volume (new_rbcc)	0 1 2	< 2.69 – 4.46 4.46 – 5.05 > 5.05
White blood cells volume (new_wbcc)	0 1 2	< 4980 – 16880 16880 – 19260 > 19260
Hypertension (htn)	0 / 1	0 / 1
Diabetes melitus	0 / 1	0 / 1
Age (age_new)	0 1 2	<12 – 43 43 – 59 > 59

Tabla 2. Codificación final de las variables validas.