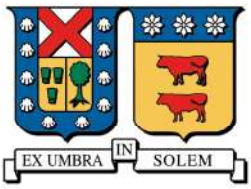

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA

Disertación doctoral

para obtener el grado académico de
Doctor en Ingeniería Informática

Algoritmo multi-fuente de imputación de datos faltantes basado en algoritmo EM y vecinos recomendados.

by

Sergio Campos Chartier

Composición de la comisión:

Guia:	Ph.D. Héctor Allende O.	UTFSM, Chile
Co-guia:	Ph.D. Juan Zamora O.	PUCV, Chile
Correferente interno:	Ph.D. Julio Sotelo P.	UTFSM, Chile
Correferente externo nacional:	Ph.D. Ignacio Araya Z.	PUCV, Chile
Correferente externo internacional:	Ph.D. J. Ariel Carrasco O.	INAOE, México
Presidente de la comisión:	Ph.D. Julio Sotelo P.	UTFSM, Chile

2025

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☐ Memoria o trabajo de título; ☒ Tesis de Postgrado;

Título del trabajo: Algoritmo multi-fuente de imputación de datos faltantes basado en algoritmo EM y vecinos recomendados.

Nombre del candidato(a): Sergio Manuel Campos Chartier

Carrera / Grado: Doctor en Ingeniería Informática

Campus: Casa Central Valparaiso **;** **Departamento**: Informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Héctor Allende Olivares, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 13/08/2025 ; **Firma:** _____

Héctor Allende Olivares

Estudiante o Candidato(a):

Fecha: 13/08/2025 ; **Firma:** _____

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

Declaration of authorship

I hereby declare that I wrote this thesis on the subject

Algoritmo multi-fuente de imputación de datos faltantes basado en algoritmo EM y vecinos recomendados.

by

independently. I did not use any other aids, sources, figures or resources than those stated in the references. I clearly marked all passages that were taken from other sources and cited them correctly.

Furthermore I declare that – to my best knowledge – this work or parts of it have never before been submitted by me or somebody else at this or any other university.

Sergio Campos Chartier

Valparaíso, July 22, 2025 _____

A las personas que me acompañaron, de alguna u otra manera, en este muy largo proceso. Gracias.

Acknowledgements

Este trabajo fue financiado por Proyecto ANID PIA/apoyo AFB 230003, en parte por proyecto DGIIP-UTFSM PI-LIR23-13. Además, por el proyecto FONDECYT Iniciación 11200826 (2020-2023). Título del proyecto: “Incremental and emotion-aware recommendations of news streams under non-stationary user preferences.”

Resumen

En los problemas de Machine Learning, la presencia de datos faltantes es un desafío común, especialmente cuando las variables del estudio provienen de múltiples fuentes de información. Cada fuente puede tener distintos formatos, niveles de precisión y frecuencias de actualización, lo que genera inconsistencias y vacíos en los datos recopilados. Por ejemplo, en un sistema de predicción de demanda que depende de datos meteorológicos, económicos y de redes sociales, algunas fuentes pueden proporcionar información en tiempo real, mientras que otras pueden presentar retrasos en las actualizaciones o contener valores faltantes debido a errores de registro. Estas discrepancias pueden afectar la calidad del modelo, reduciendo su capacidad predictiva y aumentando la incertidumbre en los resultados. Por lo tanto, es fundamental aplicar estrategias como la imputación de datos si se desea realizar una tarea de aprendizaje supervisado, como regresión o clasificación, sin perder datos en el proceso.

El algoritmo de Expectation Maximization (EM) ha sido utilizado con éxito para manejar valores faltantes, pero no está diseñado para escenarios típicos de Machine Learning, donde se crea un modelo de imputación sobre los datos de entrenamiento y luego se aplica a un conjunto de prueba. En este trabajo, proponemos EMreg-KNN, un novedoso algoritmo de imputación supervisado y de múltiples fuentes. Basado en el algoritmo EM y en el concepto de vecinos recomendados, EMreg-KNN construye un modelo de conjunto basado en regresión para la imputación de datos futuros, lo que permite la utilización posterior de cualquier método de Machine Learning basado en vectores para evaluar automáticamente tareas de clasificación.

Para evaluar esta propuesta, se utilizan tres bases de datos diferentes con datos faltantes y cuatro algoritmos de clasificación. El método propuesto competirá con otros métodos de imputación para garantizar que los algoritmos de clasificación ofrezcan los mejores resultados según métricas ROC. EMreg-KNN obtiene mejores resultados en la mayoría de los escenarios. Además, los clasificadores muestran un comportamiento más estable gracias a la imputación de los valores faltantes.

Abstract

In Machine Learning problems, the presence of missing data is a common challenge, particularly when the study variables originate from multiple sources of information. Each source may have different formats, levels of accuracy, and update frequencies, leading to inconsistencies and gaps in the collected data. For instance, in a demand prediction system that relies on meteorological, economic, and social media data, some sources may provide real-time information, while others may experience delays in updates or contain missing values due to recording errors. These discrepancies can affect the quality of the model, reducing its predictive capability and increasing uncertainty in the results. Therefore, it is essential to apply strategies such as data imputation if we want to perform a supervised learning task such as regression or classification without losing data in the process.

The Expectation Maximization (EM) algorithm has been successfully employed to handle missing values, but it is not designed for typical Machine Learning scenarios where an imputation model is created over training data and subsequently applied on a testing set. In this work, we propose EMreg-KNN, a novel supervised and multi-source imputation algorithm. Based on the EM algorithm and the concept of recommended neighbors, EMreg-KNN builds a regression ensemble model for the imputation of future data thus allowing the further utilization of any vector based Machine Learning method to automatically assess classification tasks.

In order to evaluate this proposal, three different databases with missing data and four classification algorithms are employed. The proposed method will compete with other imputation methods to ensure that the classification algorithms provide the best results based on ROC metrics. EMreg-KNN achieves better results in most scenarios. Additionally, the classifiers exhibit more stable behavior thanks to the imputation of missing values.

Contents

Índice de cuadros

Índice de figuras

Notación

$X_{N \times D}$	Una matriz X con N ejemplos y D variables.
x_{ij}	Elemento de la fila i columna j de la matriz X.
$x_{i:}$	El vector fila i-ésimo de la matrix X. Representa un ejemplo de los datos.
$x_{:j}$	El vector columna j-ésimo de la matriz X. Representa una variable o característica de los datos.
$\mathbf{Y}$	Conjunto de clases o etiquetas.
y_l	Valor de la l-ésima clase perteneciente de $\mathbf{Y}$.
Y	Vector de clases o etiquetas.
M	Matrix binaria indicadora de datos faltantes. 1 si hay datos flatantes, 0 si no.
n_{obs}	Número de datos observados.
n_{miss}	Número de datos faltantes.
$\mathbb{R}$	Conjunto de los números reales.
$\hat{x}_{ij}$	Elemento estimado de la fila i columna j.
$\bar{x}_j$	Media de la j-ésima variable.
$\bar{\hat{x}}_j$	Media de los valores estimados de la j-ésima variable.
$\mathbf{x}_{io}$	Sub-vector compuesto de las variables con datos observados del vector fila i-ésimo.
$\mathbf{x}_{im}$	Sub-vector compuesto de las variables con datos faltantes del vector fila i-ésimo.
X_o	Matriz solo con los datos observados.
X_m	Matriz solo con los datos faltantes.

Capítulo 1

Introducción

En la mayoria de los campos científicos y de investigación, la presencia de conjuntos de datos incompletos es un problema recurrente. Incluso los estudios más rigurosamente diseñados y ejecutados no están exentos de datos faltantes (en adelante MV por sus siglas en ingles: Missing Values). La existencia de MV dificulta la comprensión y explicación de los fenómenos u objetos en estudio, ya que la investigación se fundamenta en la recopilación de observaciones que buscan capturar y describir dichas realidades. Los resultados de un estudio dependen de manera crucial del análisis de estas observaciones; en consecuencia, los MV representan un desafío significativo para la validez científica de los hallazgos Mcknight et al. (2007). Además, muchas de las decisiones en los ámbitos científico, empresarial y/o económico están directa o indirectamente vinculadas con la información que las investigaciones proporcionan en momentos críticos. Por ello, es fundamental que la gestión de los MV sea abordada con especial atención y rigor.

En los últimos decadas, el avance en la capacidad de computo de los sistemas computacionales ha impulsado significativamente la adopción y desarrollo de técnicas de Inteligencia Artificial (IA) como herramientas fundamentales para la extracción de información valiosa a partir de grandes volúmenes de datos. Estas técnicas han demostrado ser altamente eficaces en una amplia variedad de tareas, tales como clasificación, predicción, extracción y selección de características, fusión de datos multimodales y reducción de dimensionalidad, entre otras. Estas capacidades resultan esenciales para abordar preguntas clave relacionadas con los fenómenos objeto de estudio, permitiendo así una comprensión más profunda y precisa de las dinámicas y relaciones subyacentes en los datos.

A pesar de los avances en las técnicas de IA y su capacidad para procesar y analizar grandes volúmenes de datos, la presencia de MV sigue representando un desafío importante para la extracción de información significativa. Los métodos de IA, si bien son poderosos, dependen en gran medida de la calidad y la completitud de los datos de entrada, ya que cualquier carencia o inconsistencia puede comprometer la precisión de las predicciones y análisis realizados. En este contexto, abordar el problema de los datos

faltantes se vuelve fundamental, no solo para garantizar la robustez de los modelos desarrollados, sino también para maximizar el aprovechamiento del conocimiento que los datos pueden ofrecer en el estudio de fenómenos complejos.

El problema de MV se puede manejar de dos maneras principalmente. En primer lugar, todos las instancias a las que les falta información se eliminan por completo antes de realizar cualquier otro análisis. Este es un enfoque razonable cuando el porcentaje de las observaciones con MV es bajo. En segundo lugar, los MV pueden estimarse a partir de los datos incompletos. Este enfoque se conoce como *Imputación* Little and Rubin (2002) y se recomienda cuando las técnicas de análisis de datos adoptadas no están diseñadas para funcionar con entradas faltantes y, además, y no menos importante, el porcentaje de datos con MV es tan alto que al utilizar el primer enfoque la base de datos se reduciría a tal punto que produciría un enorme perdida de información.

Esta tesis abordará el problema de imputación de MV en un ambiente de aprendizaje supervisado. Esto implica que los algoritmos utilizados para la imputación de MV son aplicados a los datos de entrenamiento, y en ese proceso se obtiene un modelo de imputación, el cual debiese poder imputar MV en los datos de prueba sin la necesidad de realizar cálculos adicionales. Este enfoque de trabajo no siempre es aplicado de esta manera para algoritmos de imputacion, ya que se considera como parte del preprocesamiento, pero hay miembros en la comunidad Van Der Maaten et al. (2009) Gisbrecht et al. (2012) que han hecho resaltar la importancia de esto utilizando el término Out-of-Sample (OOS) [1]. En el contexto del aprendizaje supervisado y la imputación de valores faltantes, implica que los métodos o modelos desarrollados deben ser capaces de manejar datos nuevos o no observados previamente. En el contexto de algoritmos de imputación de MV, un modelo OOS permite que las estrategias de imputación desarrolladas con los datos de entrenamiento puedan ser aplicadas de manera eficiente y coherente a datos nuevos (de prueba o en producción).

La hipótesis de esta investigación plantea que: es factible desarrollar una versión Out-Of-Sample (OOS) del algoritmo Expectation-Maximization (EM) que posibilite la imputación de valores faltantes (MV) mediante una estrategia basada en la recomendación de vecinos más cercanos. Este enfoque, aplicado en el contexto del machine learning, permitirá la reconstrucción de conjuntos de datos completos, permitien-

[1] La distinción entre datos dentro de la muestra (in-sample) y fuera de la muestra (Out-of-Sample) es fundamental para evitar problemas como el sobreajuste (overfitting), situación en la cual un modelo se ajusta excesivamente a los datos de entrenamiento y pierde capacidad predictiva frente a nuevos datos Hastie et al. (2009). En consecuencia, la evaluación OOS se considera una práctica estándar para estimar el rendimiento real del modelo y garantizar su robustez. En términos metodológicos, se utilizan estrategias como la validación cruzada (cross-validation) o la división entrenamiento-prueba (train-test split) para garantizar que el conjunto de datos OOS sea representativo y adecuadamente separado del conjunto de entrenamiento.

do así el entrenamiento de algoritmos de clasificación. Como consecuencia, se espera obtener mejoras en las métricas de desempeño basadas en curvas Receiver Operating Characteristic (ROC) tales como Accuracy, Area bajo la curva (AUC) y medida F (F-measure) entre otras. Esto en comparación con la versión clásica del algoritmo EM y otros algoritmos estado del arte.

El objetivo principal de este trabajo es diseñar, implementar y evaluar una versión modificada del algoritmo EM que permita crear un modelo de imputación en base a datos de entrenamiento. Este modelo debe permitir imputar datos no vistos anteriormente de tal manera que permita mejorar el rendimiento de algoritmos de clasificación.
Los objetivos especificos son:

1. Preparar base de datos que cumplan con los requisitos necesarios para probar los algoritmos.
2. Diseñar, implementar y probar una versión OOS del algoritmo EM.
3. Diseñar, implementar y probar una mejora de la versión OOS del algoritmo EM con la inclusión de vecinos recomendados.
4. Comparar los resultados obtenidos con algoritmos del estado del arte.

Este trabajo de investigación se organiza de la siguiente manera: el capítulo 2 se definirá formalmente el problema de MV, para luego entregar un estado del arte de los algoritmos de imputación. Al finalizar el capítulo se presenta el dataset ADNI, el cual sirvio como motivación para este trabajo de tesis.

En el capítulo 3 primero se mostrará el marco teórico, el cual describe el algoritmo EM (Expectation Maximization) Dempster et al. (1977) y una posterior versión Schneider (2001) que se usa como base para las algoritmos propuestos en esta tesis. Luego, se presenta el primer algoritmo propuesto, el cual crea una versión OOS del algoritmo EM de Schneider (2001). Por último, se presenta un algoritmo basado en el primer algoritmo propuesto que el que utiliza la idea de crear vecindarios recomendados para realizar la imputación de MV de manera iterativa.

El capítulo 4 explicará la forma en que se hicieron los experimentos y los resultados obtenidos en tres set de datos distintos. Los algoritmos de imputación propuestos se compararán con otros algoritmos del estado del arte enfocándose en el rendimiento obtenido, basado en análisis ROC, en la posterior clasificación.

Finalmente, el capitulo 5 desarrrolla las conclusiones finales junto al trabajo futuro.

Capítulo 2

Estado del arte

2.1. El problema de datos faltantes

En las últimas décadas, la información es generada casi en cualquier lugar: redes de sensores Nakamura et al. (2007), redes sociales, hospitales, empresas, centros astronómicos, etc. Muchos de los escenarios de la vida real son afectados por un inconveniente común: datos faltantes o datos incompletos. La razón por la cual existe este problema puede ser diferente en cada caso, por ejemplo: en las redes de sensores, estos pueden fallar al tratar de capturar algún dato por efectos ambientales (ruido externo o interno) o fallas de hardware (desgaste de material). En las redes sociales, las personas pueden decidir no responder preguntas por motivos de privacidad o ahorro de tiempo. En los hospitales, realizar seguimiento a las enfermedades con el fin de entender el comportamiento de éstas es de suma importancia, pero la adquisición de esta información no siempre está disponible debido a protocolos médicos, falta de recursos (máquinas, tiempo, dinero) o simplemente por la inasistencia del paciente.

El problema de los MV ha sido estudiado extensamente en la literatura desde el punto de vista estadístico, y también, en un menor grado, en la literatura de reconocimiento de patrones. Los casos de la vida real anteriormente explicados tiene la necesidad fundamental de manejar MV, ya que el manejo inapropiado puede causar grandes errores al momento de realizar la tarea solicitada (regresión, clasificación, etc.).

En esta tesis se abordará la tarea de clasificación Bishop (2007), en la cual el objetivo es crear algoritmos que identifiquen a que grupo pertenecen los datos que se le ingresan. Estos deben aprender los patrones que contienen los datos de entrenamiento que se les entrega, y luego, gracias a su capacidad de

generalización, poder clasificar automáticamente nuevos datos.

Un set de datos se representa por una matriz $X_{N\times D}$ compuesta por N ejemplos y D atributos ó características. Un ejemplo i en un problema de clasificación esta representado por $x_{i:} = [x_{i1}, x_{i2}, ..., x_{ij}, ...x_{iD}]$, donde $i = 1...N$ y $j = 1...D$ es la j-ésima variable. Además, cada vector i-ésimo tiene un valor $y_l \in \mathbf{Y} = \{y_1, y_2, ..., y_l, ..., y_L\}$, donde el índice $l = 1, 2, ...L$ indica la l-ésima clase a la que pertenece del conjunto $\mathbf{Y}$.
Cuando se trata con datos faltantes, un set de datos se representa por la ecuación 2.1:

$$D = \{X_{N\times D}, Y_{N\times 1}, M_{N\times D}\} \tag{2.1}$$

donde $x_{i:}$ es el i-ésimo vector de entrada, Y es el vector de clases y $m_{i:} = [m_{i1}, m_{i2}, ...m_{iD}] \in \{0, 1\}^D$ indica cuales variables del vector i son desconocidas (con valor uno) y cuales son observadas (con valor cero).
X, para el caso de datos faltantes, es dividida en: $X = \{X_o, X_m\}$ donde X_o son los datos observables y X_m son los datos faltantes.

El mecanismo de datos faltantes esta caracterizado por la distribución condicional de M dada X,

$$p[M \mid X, \xi] = p[M \mid X_o, X_m, \xi] \tag{2.2}$$

donde ξ denota el vector de los parámetros desconocidos que definen el mecanismo de datos faltantes. Little and Rubin (2002) definen 3 tipos de mecanismos de datos faltantes:

1. Datos perdidos completamente al azar (en inglés Missing completely at random): **(MCAR)** por sus siglas en inglés, ocurre cuando la probabilidad de que una variable tenga valor faltante es independiente de la misma variable y de cualquier otra influencia externa. MCAR puede ser expresada por:

   $$p[M \mid X_o, X_m, \xi] = p[M \mid \xi] \tag{2.3}$$

   lo cual significa que la falta de valor en las variables no depende de los valores de entrada. Los valores observables de las variables contienen toda la información para realizar la inferencia.

2. Datos perdidos al azar (en inglés Missing at Random): **MAR** por sus siglas en inglés, ocurre cuando la probabilidad de que una variable tenga valor faltante es independiente de las variables con valores faltantes pero dependiente de las otras variables con valores observables. MAR puede ser expresada por:

   $$p[M \mid X_o, X_m, \xi] = p[M \mid X_o, \xi] \tag{2.4}$$

3. Datos perdidos no al azar (en inglés Missing Not at Random): **MNAR** por sus siglas en inglés, ocurre cuando la probabilidad de que una variable tenga valor faltante no es al azar, por lo tanto depende de las variables faltantes. En contraste al mecanismo MAR, los valores faltantes no pueden ser estimados solo con las variables con datos observables. En este caso, no hay un método general para manejar los datos faltantes apropiadamente.

Cuando los datos son MCAR o MAR, el mecanismo de datos faltantes puede ser ignorado y así, simplificar los métodos usados para el análisis de datos faltantes. Por esta razón, la mayoría de los estudios consideran que los mecanismos de datos faltantes son MCAR y MAR. Además, los mecanismos se pueden dividir de acuerdo al objeto a ser estudiado:

- Informativo: Cuando el valor faltante provee información acerca de la etiqueta del ejemplo.
- No Informativo: Cuando la distribución de los valores faltantes es la misma para todas las variables, por lo tanto no entrega información adicional el hecho de que existan más o menos datos faltantes.

Un ejemplo del caso informativo ocurre cuando se trabaja con base de datos de clientes. Se espera que clientes pasivos tengan muchos más datos faltantes que clientes activos.

Cuando se trabaja los datos faltantes en el área de aprendizaje automático, existen 3 escenarios:

1. Los datos de entrenamiento están completos y los datos de prueba incompletos Saar-Tsechansky and Provost (2007).
2. Los datos de entrenamiento están incompletos y los datos de prueba completos Farhangfar et al. (2008).
3. Tanto los datos de entrenamiento como los de prueba son incompletos Luengo et al. (2012) García-Laencina et al. (2010).

El primer escenario es poco común, ya que el clasificador es entrenado con datos completos suponiendo que los datos de prueba también lo están. Bajo esta situación, existen 2 soluciones: (i) se excluyen los ejemplos con datos faltantes o (ii) se realiza imputación a los datos faltantes para así tener solo datos completos. Aquí la imputación se podría realizar basado en los datos de entrenamiento.
El segundo escenario es más común en la literatura, pero en la vida real sería bastante improbable ya que si se tienen datos incompletos para entrenar, es de esperar que los datos de prueba también esten en esta situación. Un trabajo que analiza esta situación es Farhangfar et al. (2008), estudiando varios métodos de imputación con varios tipos de clasificador.
El tercer escenario ha sido estudiado hace poca más de un par de décadas. Bajo esta situación, el clasificador es entrenado de manera similar a como será probado y, por lo tanto, el proceso de imputación

debe ser realizado tanto en el entrenamiento del clasificador como en la prueba.

Aquí se estudiará en detalle la imputación del dato faltante para la posterior tarea de clasificación. En la literatura no existe una clara conclusión sobre que método de imputación de datos permite obtener mejores metricas de clasificación, por lo tanto, la elección de este es de suma importancia. Además, se tiene evidencia empírica que el proceso de imputación afecta al rendimiento del clasificador, pero no esta claro que siempre mejore. Trabajos como Acuna and Rodriguez (2004) muestra que los métodos de imputación no aportan una mejora significativa al posterior proceso de clasificación, en cambio Farhangfar et al. (2008) y Campos et al. (2015) muestra evidencia que el proceso de imputación si puede ofrecer mejoras significativas, pero esto dependiendo del clasificador y datos a utilizar.

2.2. Estado del arte de imputación de datos faltantes

La imputación del dato faltante es el enfoque más estudiado en la literatura y una clara evidencia es la gran variedad de algoritmos que existen para dicha tarea. Trabajos como Grzymala-Busse and Hu (2001) Liu and Lei (2006) Luengo et al. (2012) muestran el rendimiento de una gran variedad de algoritmos en base de datos tanto sintéticas como reales, y como conclusión general se obtiene que no existe el algoritmo que sea mejor en todos los casos.

La clasificación de los algoritmos de imputación no esta plenamente definida, pero a continuación se detallará los algoritmos más importantes de imputación según la metodología utilizada:

- **Basados en análisis estadístico:** estos métodos son los más básicos dentro de los métodos de imputación. El método más básico es el *método de la media* McCombe et al. (2021) el cual realiza la imputación del dato faltante calculando la media de todos los valores observables perteneciente a la variable en cuestión. Considerando que hay valores faltantes en la j-ésima variable:

$$\widetilde{x}_j = \frac{1}{n_{obs}} \sum_{i=1}^{N} (1 - m_{ij}) x_{ij} \tag{2.5}$$

  donde n_{obs} es el número de valores observables en x_j. Otra opción de este método es el que esta condicionado a la clase que pertenece el ejemplo Little and Rubin (2019). En este caso, la imputación se realiza con la media, pero solo considerando los ejemplos que pertenecen a la clase del ejemplo a imputar. Además de utilizar la media también se puede utilizar la mediana, esto para que los valores extremos tengan un menor efecto en la imputación del dato, o la moda en caso de

atributos categóricos.

Otro método basado en análisis estadístico es el *método por regresión* Little and Rubin (2019). En este caso, se realiza un análisis mediante regresión para imputar el dato faltante. Utilizando las variables observables se imputa la variable faltante, generalmente con regresión lineal. Suponiendo que $f(\cdot)$ es un modelo de regresión:

$$\widetilde{x_i} = f(x_{obs}) \approx x_i \tag{2.6}$$

donde x_{obs} es el vector de entrada compuesto por solo variables observables. El modelo de regresión dependerá de la naturaleza de los datos. Comúnmente es usado un modelo lineal, pero cuando las variables no cumplen con tener una relación de este tipo es posible usar un modelo de regresión no lineal. Una ventaja por sobre el método de la media es que este preserva la varianza y covarianza de las variables con datos faltantes. Una desventaja de este enfoque es que todos los valores imputados siguen una curva de regresión, y no puede representar de forma adecuada, alguna variación en los datos.

Hot deck y *Cold deck* Little and Rubin (2002) son dos métodos simples de imputación. El primero reemplaza los datos faltantes por los datos del ejemplo más similar al ejemplo a imputar considerando todas las variables observables en ambos ejemplos, es decir, aplica una función de distancia con las variables observables. *Cold deck* sigue la misma idea que *Hot deck*, pero se busca el ejemplo más diferente (o distante) al ejemplo a imputar.

Imputación Multiple es un método que se dice que es superior a los 2 métodos anteriores. Este método fue introducido por Rubin (1978) y tiene 3 principales pasos:

1. Los datos faltantes (cada dato) son imputados K veces para así generar K bases de datos completas. El método de imputación generalmente incorpora una cuota de variación aleatoria.
2. Las K bases de datos completas son analizadas usando algún procedimiento estándar (por ejemplo ANOVA).
3. Los resultados de los K análisis son combinados para realizar la imputación (inferencia o estimación) final.

El método propuesto por Rubin para combinar esta pensado para los parámetros del modelo a utilizar (pero no exclusivamente), como por ejemplo un coeficiente de regresión.
Si el parámetro de interés es σ (y $\widehat{\sigma}$ el estimador de σ) entonces el estimador final es simplemente

el promedio de los K estimadores (esto pensando en un modelo de regresión lineal):

$$\overline{\sigma} = \frac{1}{K}\sum_{i=1}^{K}\widehat{\sigma}_i \tag{2.7}$$

La incertidumbre en $\overline{\sigma}$ tiene 2 partes: la varianza promedio dentro de las imputaciones:

$$\overline{W} = \frac{1}{K}\sum_{i=1}^{K}\widehat{W}_i \tag{2.8}$$

donde $\widehat{W}_i = var(\widehat{\sigma}_i)$ y la varianza promedio entre imputaciones:

$$B = \frac{1}{K-1}\sum_{i=1}^{K}(\widehat{\sigma}_i - \overline{\sigma})^2 \tag{2.9}$$

La varianza total es una suma modificada de ambas varianzas:

$$T = \overline{W} + (\frac{K+1}{K})B \tag{2.10}$$

Otra técnica estadística utilizada para imputar datos faltantes es *Bootstrap*. Bootstrap para imputación Efron (1994) esta altamente ligada a la técnica *Imputación Multiple*, debido a las reiteradas imputaciones realizadas a los datos originales. El procedimiento se puede resumir en los siguientes pasos:

1. Generar B muestras Bootstrap X^b, $b = 1...B$ con reemplazo de la muestra original (incompleta) X.
2. Imputar los datos faltantes en X^b aplicando algún método de imputación Imp a X^b, es decir: $\widehat{X}^b = Imp(X^b)$.
3. Calcular el parámetro θ^b de $\widehat{X}^b$.

Bootstrap no imputa el dato, pero su estrategia de remuestreo permite obtener un gran número de imputaciones que ayudan a lograr un resultado final mejor, en sentido que al dar mayor variabilidad al proceso de imputación, este es más robusto y menos sesgado. Comúnmente es utilizada con otras técnicas, como por ejemplo modelos basados en árboles He (2006) Ciaccio (2011). La estrategia utilizada para fusionar el resultado de las B imputaciones es otro parámetro del método. Cabe mencionar que además de obtener la imputación del dato, es posible obtener estimaciones de los parámetros de las distribuciones subyacentes (lo cual es el objetivo original de Bootstrap). Una desventaja es el gran costo computacional que conlleva Bootstrap, ya que se recomienda un número grande para B para obtener un buen resultado.

Por último, otro algoritmo relacionado a imputación multiple es MICE Raghunathan et al. (2001). Se puede encontrar una explicación sencilla en Azur et al. (2011). En resumen, MICE realiza una imputación inicial por algun método sencillo y rápido, como la media. Luego, por cada dato imputado se utilizan modelos de regresión que actualizan las imputaciones anteriores. De esta manera, los valores se actualizan tantas veces como iteraciones uno decide realizar de este proceso.

- **Basados en factorización de matrices:** completación de matrices (matrix completion) es la tarea de llenar los MV de una matriz observada parcialmente mediante procedimientos de factorización de matrices, lo cual equivale a realizar imputación de datos. En Troyanskaya et al. (2001) se realiza imputación para datos en microarreglos mediante *Descomposición en Valores Singulares (Singular Value Decomposition, SVD)*. Basado en la descomposicion que ofrece SVD: $X = U\Sigma V^t$ la idea es resolver el problema de la ecuación 2.11:

$$\min_{U_k,\Sigma_k,V_k} \|X - U_k\Sigma_k V_k\|^*, \tag{2.11}$$

donde $\|\cdot\|^*$ es la norma matricial cuadrada que ignora los elementos donde X tiene MV, Σ_k es la matriz diagonal que contiene los $k \leq D$ valores singulares mayores y U_k y V_k son matrices ortogonales que contienen los vectores singulares izquierdos y derechos de rango-k.
Este método ayudado de los vectores y valores propios realiza la imputación del dato, siguiendo un modelo regresivo lineal, de forma iterativa. Ya que SVD puede utilizarse solo en matrices completas, inicialmente se procede a imputar los datos mediante el método de la media. El método iterativo utilizado para llegar finalmente al valor final imputado es el algoritmo EM.

Singular Value Thresholding (SVT) Cai et al. (2010) es un algoritmo que minimiza la norma nuclear (también conocida como la traza de la norma) de una matriz, sujeta a cierto tipo de restricciones. En este caso, el algoritmo SVT resuelve el problema de la ecuación 2.12:

$$\min_{X \in C} \lambda\|X\|_* + \frac{1}{2}\|X\|_F^2, \tag{2.12}$$

donde $C = \{X \in \mathbb{R}^{N\times D} s.t. |X(i,j) - M(i,j)| \leq \varepsilon \;\; \forall (i,j) \in \Omega\}$ y Ω es el conjunto de datos. La primera norma de la ecuación 2.12 es la norma nuclear y la segunda es la norma de Frobenius. La matriz M es la matriz original con datos faltantes imputados con ceros.
El algoritmo SVT tiene los siguientes parámetros:

- λ: la penalización sobre los valores singulares.
- δ: parámetro que afecta la convergencia y comunmente inicializado en $\delta = 1{,}2\frac{(N\times D)}{\eta}$ donde η es el número de datos faltantes.

- ε: el umbral de convergencia. Comunmente inicializado en 10^{-4}.
- $iter_{max}$: el máximo número de iteraciones.

Entonces, el método iterativo para resolver el problema es:

1. Inicializar los parámetros: λ, δ, ε, $iter_{max}$.
2. Inicializar $k = \lfloor \frac{\lambda}{\delta \|M\|_F^2} \rfloor$
3. Inicializar $i = 0$ e $Y^i = k \times \delta \times M$.
4. Calcular la SVD de Y^i (U^i, Σ^i, V^i).
5. Calcular $r_i = \max\{j : \sigma_j^i > \lambda\}$.
6. Calcular $X^i = \sum_{j=1}^{r_i} (\sigma_j^i - \lambda) u_j^i v_j^i$.
7. Si $\frac{\|X^i - M\|_F}{\|M\|_F} < \varepsilon$ para terminar el método debido a criterio de convergencia.
8. Fijar $i = i + 1$ y calcular

$$Y^i = \begin{cases} 0, & \forall (i,j) \notin \Omega_\eta \\ Y_{ij}^{i-1} + \delta (M_{ij} - X_{ij}^i), & \forall (i,j) \in \Omega_\eta \end{cases}$$

donde Ω_η es el conjunto de datos faltantes en M. Repetir desde el paso 4 hasta $i > iter_{max}$.

En general, estos algoritmos no crean un modelo que permita la imputación de datos futuros. Esto genera una desventaja al trabajar en un entorno de aprendizaje supervisado.

- **Basados en máquinas de aprendizaje:** estos métodos se basan de la información disponible en la base de datos para crear un modelo para realizar la imputación. Estos métodos son mas sofisticados que los métodos basado en análisis estadístico y generalmente crean un modelo predictivo para estimar el valor faltante.

Uno de los métodos más usado y simple es el método de imputación $K - nn$ o de los K vecinos más cercanos. Este método imputa datos faltantes ayudándose de los K ejemplos (vecinos) más similares, basado en una métrica de distancia, que compartan los mismos patrones de datos faltantes.

Una vez son encontrados los K ejemplos más cercanos, es posible usar la moda para datos discretos o la media para datos continuos para realizar la imputación. Los principales inconvenientes de este método es la extrema dependencia del parámetro K y la métrica de distancia utilizada. En Batista and Monard (2002) se propone la *Heterogeneous Euclidean Overlap Metric* (HEOM) la

cual considera la distancia entre 2 ejemplo como:

$$Dist(x_a, x_b) = \sqrt{\sum_{j=1}^{D} Dist(x_{aj}, x_{bj})^2} \tag{2.13}$$

donde $Dist(x_{aj}, x_{bj})$ es:

$$Dist(x_{aj}, x_{bj}) = \begin{cases} D_{cat}(x_{aj}, x_{bj}), & x_j \text{ es una variable categórica} \\ D_{num}(x_{aj}, x_{bj}), & x_j \text{ es una variable numérica} \end{cases}$$

La función D_{cat} asigna un valor 0 si los valores de las variables categóricas son los mismos y 1 si son distintos. La función D_{num} es:

$$D_{num}(x_{aj}, x_{bj}) = \frac{|x_{aj} - x_{bj}|}{max(x_j) - min(x_j)} \tag{2.14}$$

Otro método utilizado es la red neuronal *SOM (Self-Organizing Map)* Kohonen et al. (2001). Este método que inicialmente fue utilizado para reducir la dimensión de datos altamente dimensionales, conservando la topología (estructura) de estos, puede ser utilizado para realizar la imputación de datos faltantes. Fessant and Midenet (2002) presentan esta extensión del uso de la SOM. Primero, por cada ejemplo de entrada con valores faltantes, la neurona ganadora es elegida midiendo la distancia con solo variables observables. Luego, cada valor faltante es imputado en base a los pesos de activación del grupo de nodos pertenecientes a su vecindad. La elección de la neurona ganadora es elegida ignorando las variables donde existen datos faltantes. Este método solo trabaja con datos faltantes en los datos de prueba. Para trabajar con los datos de entrenamiento también, Piela (2002) implementa una red SOM con estructura de árbol.

Otro método usado es basado en *Perceptron Multicapa (Multi-layer perceptron, MLP)*. Este método se basa en la buena capacidad que tiene para realizar regresión para imputar el dato faltante. El modelo de regresión usa las variables observables para realizar la imputación, y construye tantos modelos como patrones de MV existen en la base de datos. Claramente esto aumenta rápidamente su costo computacional segun aumenta el número de patrones de MV. Aquí la variable dependiente es la variable faltante y las variables independientes son las variables observables. Luego de realizar la imputación, es posible entrenar una red nuevamente para realizar la clasificación. Estos métodos han sido estudiados en Sharpe and Solly (1995) Brouwer and Pedrycz (2003).

$MissForest$ Stekhoven and Bühlmann (2012) es un algoritmo basado en Random Forest (RF) que permite realizar imputación de datos sin importar si son numéricos o categóricos. Primero

realiza una imputación inicial mediante la media si el dato es numérico o la moda si es categórico. La variable a imputar se transforma en la variable dependiente y las otras en las variables independientes. Entonces, utilizando random forest, se imputa el dato con los distintos árboles creados hasta que el valor del dato imputado converge. Esto se realiza con cada valor a imputar. Los autores Stekhoven and Bühlmann (2012) muestran que el método es mucho mejor que la imputación mediante KNN.

Máquinas de Soporte Vectorial (Support Vector Machines, SVM) puede ser usada para realizar imputación mediante regresión. El enfoque con el cual la SVM realiza regresión (siendo que fue pensada para clasificación) se llama SVR (Support Vector Regression) Drucker et al. (1996). El enfoque trabaja de forma similar a MLP, es decir, la regresión es realizada para imputar el dato faltante considerando las variables observables. Cada imputación es realizada solo para un patrón de MV, por lo tanto y al igual que en MLP, la SVR construye tantos modelos regresivos (no lineales) como patrones de datos faltantes existan en la base de datos.

Imputación del dato mediante *Clustering* también ha sido utilizado en la literatura. Clustering es la tarea de agrupar objetos considerados similares bajo una determinada métrica de similaridad. En Patil et al. (2010) realizan imputación utilizando el conocido algoritmo *k-means*, el cual genera k clusters que sirven para realizar una búsqueda de vecinos más cercanos acotada. La imputación es realizada localmente con los datos que solo pertenecen al cluster, lo cual disminuye la complejidad que tiene el método $k-nn$ que realiza la búsqueda de los k-vecinos en los datos completos. Al igual que en Patil et al. (2010), en Zhang et al. (2008) utilizan $k-means$ para realizar la imputación. La diferencia radica en el modelo utilizado para imputar:

$$Y_i = m(x_{i1}, x_{i2}, ..., x_{iD}) + e_i \tag{2.15}$$

donde $i = 1...N$, e_i es un error aleatorio con media cero y varianza σ^2, $m(x_{i1}, x_{i2}, ..., x_{iD})$ es una función (no lineal) desconocida que es calculada mediante una función de kernel gaussiana. Este calculo es realizado solo con los datos que pertenecen al mismo cluster que el dato a imputar.
Otro ejemplo que se ayuda de clustering para imputar datos se puede encontrar en Nikfalazar et al. (2019). Aquí, el dataset es dividido en 2 subconjuntos: D_I que contiene datos con MV's y D_C que no contiene MV's. Mediante el algoritmo de árboles de decisión, se imputa de a una variable a la vez, dejando la variable a imputar como la variable objetivo o target. Si la variable es númerica, se utiliza un árbol de regresión, y si la variable es categórica, se utiliza un árbol de clasificación. Entonces, se utiliza fuzzy K-means para mejorar la imputación realizada previamente. La tarea

de clustering se aplica con los datos de cada nodo terminal (nodo hoja) del árbol. Entonces, utilizando el coeficiente de Silhouette, se calcula el número de clusters, y con los centroides de estos se realiza la mejora de la imputación de manera iterativa hasta que los cambios no superen un cierto umbral predefinido.

Un enfoque dentro de las técnicas de máquinas de aprendizaje es el *aprendizaje multi-tarea* (Multi-task Learning, MTL). MTL es un enfoque que trata de resolver un problema principal junto a otros problemas relacionados al mismo tiempo, compartiendo la misma representación. En García-Laencina et al. (2007) se usa una red neuronal Feed-Forward que tiene como objetivo principal la clasificación de los datos y como objetivo secundario la imputación de los datos faltantes. La estructura de la red tiene 3 capas, la primera correspondiente a la capa de entrada, la cual tiene tantos nodos como variables más un nodo que entrega la información relacionada a la etiqueta del ejemplo, una capa escondida (segunda capa), y una última capa correspondiente a las salidas. En este caso, el número de salidas es igual al número de variables con datos faltantes más la salida correspondiente a la etiqueta del dato a clasificar. La imputación se realiza utilizando los demás datos disponibles además de la etiqueta de los ejemplos en la fase de entrenamiento. En la fase de prueba, a diferencia de algunos métodos (SVD, SVT, EM), se puede probar con ejemplos con datos faltantes. Al ingresar un ejemplo con datos faltantes se realiza una imputación por cada clase (etiqueta) existente, midiendo la consistencia de la clase analizada. La consistencia es una métrica que incluye la diferencia entre la etiqueta que entrega la red y la etiqueta analizada. La salida final es la etiqueta que entrego la mayor consistencia.

Así como existe el Multi-task Learning, también existe el *aprendizaje multi-vista* (Multi-view Learning, MVL). MVL es un enfoque que realiza aprendizaje cuando los datos están distribuidos en distintas vistas o fuentes. En Zhang et al. (2018) se utiliza un enfoque de MVL para realizar imputación de datos con distintas fuentes. En este trabajo los autores crean un subespacio de características común para poder llevar los datos ahí y permitir que los datos de las mismas clases queden lo más juntos posibles y los de las clases distintas lo más separados posibles. Mediante una formulación como problema de optimización, encuentran las transformaciones lineales que permiten esto. Luego, se realiza la imputación de datos utilizando la idea de factorización de matrices. Teniendo la matriz con MV X_M se propone que es posible obtener $X_M = L_M + S_M$, donde L_M es una matriz low-rank que representa la diferencia entre clases y S_M, una matriz sparse que representa la diferencia dentro de las clases. Al igual que en el primer paso, L_M y S_M son calculadas mediante una formulación como problema de optimización.

En los últimos años, enfoques basados en Aprendizaje Profundo (Deep Learning, DL) han sido utilizados para realizar imputación de datos. Un ejemplo es usar una red Feed Forward profunda. En Cheng et al. (2020) utilizan una red con 3 grupos de capas ocultas con 15 capas cada una. Esto lo hacen para imputar una variable categórica importante del dataset, y así, posteriormente unir esta información con el dataset completo.

En la mayoria de los trabajos relacionados con imputación de datos y aprendizaje profundo, los modelos basados en Auto Encoders (AE) son los más populares. En Macias et al. (2019) utilizan un modelo clásico de AE para imputar datos que posteriormente son procesados por una red Long Short-Term Memory (LSTM). Un trabajo interesante es el de Abiri et al. (2019). Aquí, realizan imputación de datos heterogéneos con redes AE. Para poder imputar datos binarios, continuos y categóricos modifican la función de pérdida de la red, siendo distinta para cada tipo de dato. Además, en este cálculo, solo consideran las variables observables. Los autores demuestran que AE se comporta mejor que técnicas clásicas como: KNN, MICE y Mean.

Existen distintas variantes entre los modelos basados en AE. Entre ellos estan los: Denoising AE (DAE) y los Variational AE (VAE). En Gondara and Wang (2018), utilizan un DAE para realizar la imputación. Primero realizan una imputación inicial utilizando métodos simples como el promedio o la moda. Luego, aplican DAE varias veces considerando inicializaciones distintas de los pesos de la red. De esta manera, buscan seguir una estrategia de imputación multiple como otros métodos clásicos. Los autores toman una decisión poco común con respecto a la arquitectura de la red. La capas del Encoder aumentan su tamaño en vez de disminuir como es lo tradicional. Luego, las capas del Decoder disminuyen su tamaño en la misma proporción en que aumentaron en el Encoder. De esta manera, la capa de entrada y salida terminan teniendo el mismo tamaño.

En Pereira et al. (2020) los autores realizan un barrido de trabajos entre los años 2014 y 2020 de imputación de datos con modelos basados en AE. Los autores resumen aspectos como: función de perdida, arquitectura de la red, sintonización de parámetros utilizadas, etc. de los trabajos revisados. Los mejores resultados son obtenidos por DAE y VAE, comparandolos con métodos clásicos como: KNN, MICE, SVD, Mean, etc.

Para clasificación, existen diversas métricas de rendimiento, pero las más ampliamente utilizada en la literatura es la basada en análisis ROC (clasificación binaria). Análisis ROC Fawcett (2006) (Receiver Operating Charasteristics) entrega métricas para medir el rendimiento de un clasificador basado en la matriz de confusión. Esta matriz entrega los Falsos Positivos (FP), Falsos Negativos (FN), Verdaderos Positivos (VP) y Verdaderos Negativos (VN). Las métricas comúnmente usadas a partir del análisis ROC son:

1. Precision = $\frac{VP}{VP+FP}$

2. Recall = $\frac{VP}{P}$
3. Accuracy = $\frac{VP+VN}{P+N}$
4. Specificity = $\frac{VN}{FP+VN}$
5. F-measure = $\frac{2}{\frac{1}{Precision}+\frac{1}{Recall}}$

donde P son todos los ejemplos de una clase (clase positiva) y N son todos los ejemplos de la otra clase (clase negativa).

2.3. ADNI: un ejemplo de datos faltantes

ADNI (Alzheimers Disease Neuroimaging Initiative)[1] es un estudio que comenzó el año 2004 con el fin de entender de mejor manera la enfermedad de Alzheimer. El proyecto inicialmente se componía de 800 sujetos, de los cuales se extraían distintas características (bio-químicas, genéticas, imágenes, etc.) y se etiquetaban entre 3 grupos: Alzheimers Disease (AD), Mild Cognitive Impairment (MCI) y Healthy Control (HC). MCI se puede dividir en 2: sMCI (stable MCI) y pMCI (progressive MCI). sMCI son los pacientes que su estado no evoluciona a AD, es decir la enfermedad esta estable, y pMCI son los pacientes que su estado evoluciona a AD, es decir, su estado progresa al más avanzado de la enfermedad. Los principales desafíos en este proyecto son la predicción y la clasificación del estado del sujeto, refiriéndose a estado a una de las 4 etiquetas recién mencionadas.

El proyecto ADNI hasta el día de hoy Weiner et al. (2012) Weiner (2015), esta compuesto por 4 grandes etapas: ADNI, ADNI-GO, ADNI-2 y ADNI-3 Weber et al. (2021). En cada etapa se agregan nuevas fuentes de información y más ejemplos tanto de nuevos sujetos como de los mismos que han participado anteriormente. Con esto se permite realizar estudios longitudinales que pretenden estudiar la evolución del patrón de la enfermedad. La tabla 2.1 muestra detalles de las 3 primeras etapas de ADNI.

ADNI-3 empezo el 2016 y esta actualmente en proceso de desarrollo, teniendo como objetivo incluir cerca de 2000 pacientes nuevos, entre pacientes sanos y con un cierto grado de la enfermedad.
Los datos de ADNI poseen varias particularidades: datos heterogéneos (datos de distintas fuentes), datos faltantes, datos de gran dimensión y datos históricos. Estas características han originado grandes desafíos al momento de analizar los datos y una abundante literatura demuestra que las técnicas computacionales tradicionales de análisis de datos no siempre resultan adecuadas. La figura 2.1 muestra un escenario

[1] http://www.adni-info.org

Características de estudio	ADNI	ADNI-GO	ADNI-2
Duración/Comienzo	5 años / 2004	2 años / 2009	5 años/ 2011
	200 HC	ADNI más:	ADNI-GO más:
Cohorte	200 MCI / 400 AD	200 EMCI	150 HC / 100 EMCI
			150 MCI / 150 AD
Técnicas de estudio			
MRI	X	X	X
fMRI		X	X
FDG-PET	X	X	X
AV45		X	X
BioSamples	X	X	X

Cuadro 2.1: Comparación entre ADNI, ADNI-GO y ADNI-2. Weiner et al. (2012)

ejemplificando la situación de ADNI-1.

Es así como Hinrichs et al. (2009) Gray et al. (2013) tratan el problema de fusión de datos. La pregunta acá es como utilizar la información proveniente de las distintas fuentes para poder realizar una mejor clasificación. En los trabajos de Zhang y Shen Zhang and Shen (2011) Zhang et al. (2011) utilizan 3 fuentes de datos distintas, pero acá realizan un ranking de características lo cual permite obtener información acerca de cuales son más importantes con respecto a su poder discriminatorio entre sujetos sanos y enfermos. Este problema es bastante importante, ya que el número de variables pueden llegar a los miles (debido a imágenes de RM[2] y TEP[3]), lo que hace atractivo aplicar técnicas de reducción de dimensionalidad.

De los 800 sujetos presentes (en ADNI 1) 657 poseen datos faltantes en una o más fuentes de datos. Desde aproximadamente el año 2012 aparecen los primeros trabajos donde tratan de resolver este problema no descartando simplemente los ejemplos con esta cualidad. Yuan Yuan et al. (2012) trata el problema de fusión de datos y datos faltantes en ADNI. Ellos trabajan con 4 fuentes de datos y 780 sujetos, lo cual permite realizar un mejor estudio al disponer de más ejemplos de cada clase. Se realiza imputación del dato faltante mediante 4 métodos clásicos y uno propuesto, y la clasificación es realizada mediante SVM. Más trabajos de fusión multimodal con MV pueden ser encontrados en Liu et al. (2017) Thung et al. (2014) Zhou et al. (2019). EL problema de MV en ADNI sigue siendo de interes para la

[2]Resonancia Magnética, en adelante MR por su sigla en inglés.

[3]Tomografía por Emisión de Positrones, en adelante PET por su sigla en inglés.

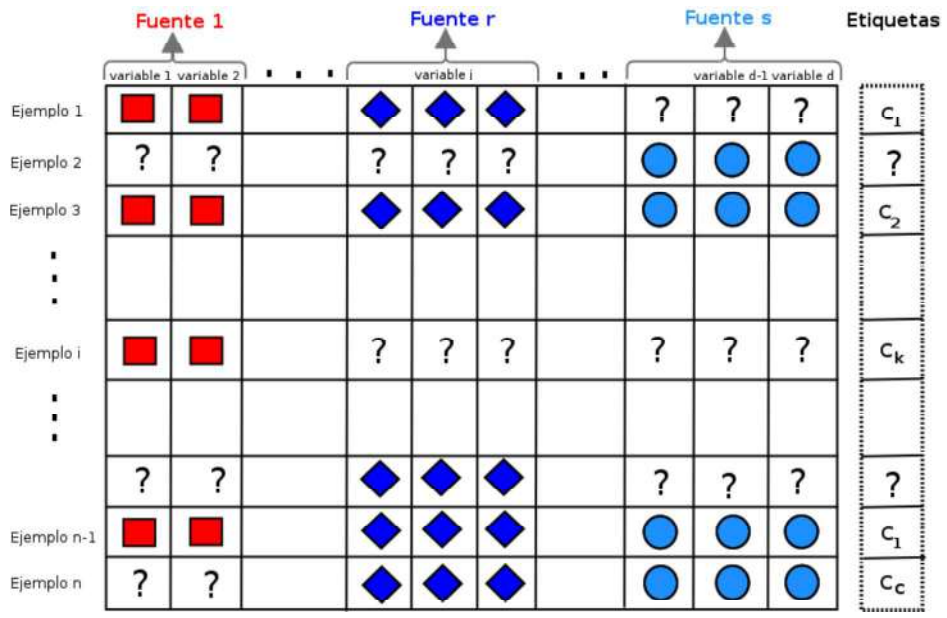

Figura 2.1: ADNI-1 tiene datos faltantes por bloques. Cada bloque es una fuente de información (s fuentes en total). Los simbolos de interrogación (?) representan datos faltantes.

comunidad. Ejemplos recientes de esto se pueden encontrar en Aghili et al. (2022) Aracri et al. (2023) Stempfle et al. (2023).

La comunidad que trabaja con ADNI cada vez ha logrado mejores resultados. Los procedimientos de preprocesamiento generalmente son distintos, resultando set de datos con diferentes número de variables y de ejemplos. Adicionalmente, la comunidad crea algoritmos cada vez más ad-hoc a las características de ADNI y rara vez muestran resultados con otros data sets.

Esta tesis propone dos mejoras del algoritmo EM, un algoritmo ampliamente usado para tratar los MV, el cual es usado en muchas comunidades científicas, incluyendo la que trabaja con ADNI. Las mejoras fueron motivadas considerando algunas características de este dataset, principalmente la cualidad de ser multi-fuente. Sin embargo, este trabajo no quiere solamente limitar sus resultados a ADNI, sino que se desea demostrar que las mejoras propuestas del algoritmo EM pueden replicarse a otros set de datos que cumplan la característica de ser multi-fuente.

Capítulo 3

Marco teórico y propuesta

3.1. Marco Teórico.

Dentro del estado del arte del problema de MV, existe un algortimo que realiza supuestos acerca de las distribuciones conjuntas de las variables del modelo. Este algoritmo se llama: Expectation Maximization (EM) propuesto por Dempster Dempster et al. (1977). El algoritmo EM es un método general e iterativo para la estimación de parámetros distribucionales mediante la maximización de la verosimilitud. Este método es usado generalmente cuando algunas de las observaciones de las variables aleatorias no son observadas. El algoritmo consiste en 2 etapas o pasos que se van ejecutando iterativamente: el paso E (Expectation) y el paso M (Maximization). El paso E se encarga de obtener el valor esperado de la verosimilitud dado los valores observados y los valores faltantes. Luego, en la segunda etapa, el paso M maximiza la verosimilitud esperada con respecto a los parámetros. Formalmente, los pasos se pueden expresar como:

$$Step\ E : Q(\theta_t) = E[l(\theta|X, z)]$$
$$Step\ M : \theta_{t+1} = arg\ \max_{\theta} Q(\theta_t)$$

donde θ_t es el vector de parámetros en la iteración t, X es la matriz que contiene todos los valores observados, $l(\cdot)$ es la función de log-verosimilitud y z el vector que representa los valores faltantes.

Un aporte muy importante para esta tesis es el trabajo de Tapio Schneider Schneider (2001). Schneider propone un algoritmo basado en EM que agrega una tercera etapa en cada iteración. Esta tercera etapa realiza una imputación de los MV mediante un modelo de regresión lineal.

El enfoque es un poco distinto, y como método paramétrico, hace los siguientes supuestos: *los datos faltantes son "Missing at random"(MAR)* y *los datos provienen de una distribución Gaussiana*.
Entonces, el algoritmo comienza calculando la media y matriz de covarianza (la cual es obtenida a partir de los valores observados). Con esto se realiza imputación de los valores faltantes a través de pasos alternados (pasos E y M) y re-estima la media y matriz de covarianza de los valores completos. El algoritmo termina cuando los valores imputados, la media y la matriz de covarianza no cambian de una iteración a otra (o el cambio es menor que un valor umbral e).

La imputación del dato faltante en el i-ésimo vector viene dada por un modelo de regresión lineal descrito en la ecuación 3.1 que relaciona variables con datos faltantes y variables con datos observados:

$$\mathbf{x}_{im} = \mu_m + (\mathbf{x}_{io} - \mu_o)\beta + e \tag{3.1}$$

$\mathbf{x}_{io} \in \mathbb{R}^{1\times p_o}$ es un sub-vector con p_o variables con datos observados, $\mathbf{x}_{im} \in \mathbb{R}^{1\times p_m}$ es el sub-vector con p_m variables con datos faltantes (missing), $\mu_o \in \mathbb{R}^{1\times p_o}$ es el vector de medias de las variables con datos observados y $\mu_m \in \mathbb{R}^{1\times p_m}$ es el vector de medias de las variables con datos faltantes (missing). La matriz $\beta \in \mathbb{R}^{p_a\times p_m}$ es una matriz de coeficientes de regresión y el residual $e \in \mathbb{R}^{1\times p_m}$ es un vector aleatorio con media cero y matriz de covarianza $COV \in \mathbb{R}^{p_m\times p_m}$ desconocida.

La matriz de coeficientes de regresión β es estimada mediante al ecuación 3.2:

$$\widehat{\beta} = \widehat{\Sigma}_{oo}^{-1}\widehat{\Sigma}_{om} \tag{3.2}$$

y la matriz de covarianza COV mediante la ecuación 3.3

$$\widehat{COV} = \widehat{\Sigma}_{mm} - \widehat{\Sigma}_{mo}\widehat{\Sigma}_{oo}^{-1}\widehat{\Sigma}_{om} \tag{3.3}$$

donde $\widehat{\Sigma}_{mm}$ es la matriz de covarianza estimada de variables con datos faltantes, $\widehat{\Sigma}_{oo}$ es la matriz de covarianza estimada de variables con datos observados y $\widehat{\Sigma}_{om} = \widehat{\Sigma}_{mo}^{T}$ es la matriz de covarianza estimada de variables con datos observados y faltantes.

Luego que los datos faltantes son imputados, se itera para actualizar la media μ de cada característica (atributo) mediante la ecuación 3.4 :

$$\widehat{\mu}^{t+1} = \frac{1}{N}\sum_{i=1}^{N} x_{i:} \tag{3.4}$$

donde el nuevo estimador de la media es la media muestral, Σ mediante la ecuación 3.5:

$$\widehat{\Sigma}^{t+1} = \frac{1}{\widetilde{n}} \sum_{i=1}^{n} \{\widehat{S}_i^t - (\widehat{\mu}^{t+1})^T \widehat{\mu}^{t+1}\} \tag{3.5}$$

donde $\widehat{S}^t \equiv E[X_i^T X_i | x_o; \widehat{\mu}^t, \widehat{\Sigma}^t]$ es la esperanza condicional y $\widetilde{n}$ es la constante de normalización que indica los grados de libertad de la matriz de covarianza muestral de la base de datos completa. Si por ejemplo, solo un vector μ es estimado, el número de grados de libertad es $\widetilde{n} = n - 1$.

El algoritmo propuesto por Schneider (2001) agrega una diferencia a lo recien explicado. Esta diferencia se ve reflejada en que la matriz $\widehat{\Sigma}_{oo}^{-1}$ estimada iteración por iteración (ecuación 3.2) se reemplaza por la ecuación 3.6:

$$\widehat{\Sigma}_{oo}^{-1} \longleftarrow (\widehat{\Sigma}_{oo} + h^2 \widehat{D})^{-1} \tag{3.6}$$

donde $\widehat{D} = Diag(\widehat{\Sigma}_{oo})$ es la matriz diagonal consistente de los elementos de la diagonal de la matriz de covarianza y h es un parámetro de regularización. Esto busca solucionar el problema que aparece cuando el número de variables con datos observables (p_o) es mayor que el número de grados de libertad $\widetilde{n}$ de la matriz de covarianza. Esto produce que la matriz $\widehat{\Sigma}$ sea singular y la estimación de la máxima verosimilitud de la matriz de coeficientes de regresión β no este definida. Dado esto, llamaremos al algoritmo de Schneider **EMreg**.

3.2. Propuesta 1: EMreg Out-of-Sample.

El algoritmo EMreg de Tapio Schneider ha sido usado ampliamente en la literatura para imputar datos faltantes. Desde el 2001 hasta la fecha, ha sido referenciado más de 800 veces y ejemplos de estos son Thung et al. (2014) Rahman and Islam (2016).

En el contexto de Aprendizaje Supervisado, es común encontrar set de datos con MV. Este fenómeno debe tratarse para, posteriormente, realizar alguna tarea como predicción de alguna variable continua (regresión) o alguna variable categórica (clasificación). Bajo este escenario, el algoritmo EMreg trabaja separadamente sobre datos de entrenamiento (training set) y prueba (testing set), lo que no es correcto del todo, ya que todo algoritmo que trabaje con un training set, debe crear un modelo, el cual posteriormente será aplicado al testing set. Esta metodología siempre se cumple con cualquier algoritmo supervisado y no deberia ser excluyente con algoritmos de pre-procesamiento como técnicas de reducción de dimensionalidad y de imputación de MV. La figura 3.1 muestra lo anteriormente mencionado:

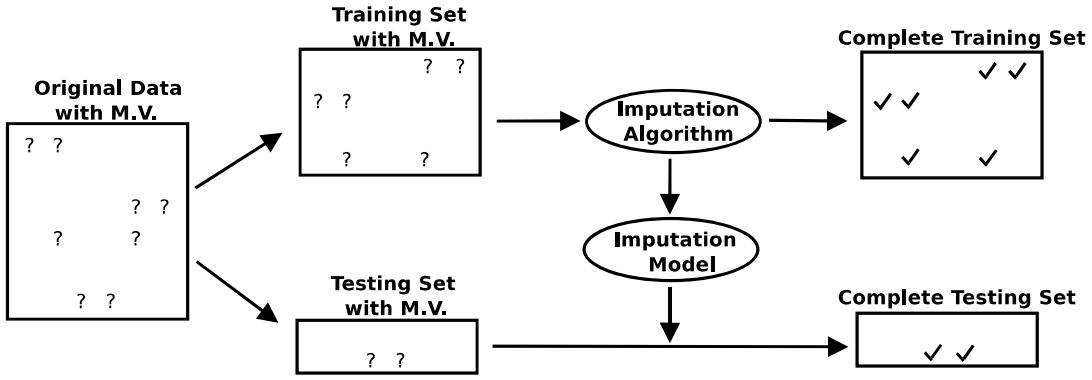

Figura 3.1: Metodología de aplicación de algoritmos en aprendizaje supervisado. Tanto los datos de entrenamiento como de prueba pueden tener MV.

En Van Der Maaten et al. (2009) se destaca para las técnicas de reducción de dimensionalidad, la necesidad de que estos algoritmos deben tener la facultad de poder proyectar datos futuros a un espacio de menor dimensión. Esto incluye la situación en donde los datos futuros vienen en bloques o individualmente. Los algoritmos que tienen esta facultad es porque poseen una extensión *Out-of-Sample (oos)*. Dicho de otra forma, estos algoritmos en la fase de entrenameinto crean un modelo que permite realizar la tarea por la cual fueron diseñados, sea clasificación, regresión, reducción de dimensionalidad o imputación, de manera eficiente y directa a datos nuevos o no observados previamente (fase de prueba).

Aquí se presenta una nueva versión del algoritmo EM, basado en la propuesta de Schneider. Esta versión resuelve el problema que aparece cuando los datos de prueba (testing set) llegan de un dato a la vez (situación real), ya que la propuesta original no puede construir el modelo de imputación, al ser este un modelo de regresión.

A esta nueva versión la llamaremos: **EMreg Out-of-Sample** (**EMreg-oos**) Campos et al. (2018). EMreg-oos puede ser aplicado sin problemas en un escenario donde tanto los datos de entrenamiento (training set) como el testing set tienen MV. Una vez utilizado en el training set, el algortimo crea un modelo general que consta de tantos modelos de regresión como patrones de MV existen en el training set. Estos modelos de regresión estan basados en la ecuación 3.7:

$$x_{im} = \mu_m + (x_{io} - \mu_o)\beta \tag{3.7}$$

lo que además de utilizar la matriz β, utilizan el vector de medias μ_m y μ_o.

El algoritmo trabaja en un escenario multi-source, es decir, se tienen distintas fuentes de información donde cada una tiene un número determinado de características. Esto implica que el fenómeno de MV es a nivel de fuentes (bloques de variables), como en el caso de ADNI. La figura 3.2 ejemplifica esta situación con 3 fuentes de información S_s (S_1, S_2 y S_3) y 3 patrones de MV (solo S_1, solo S_3 y $S_1 - S_3$):

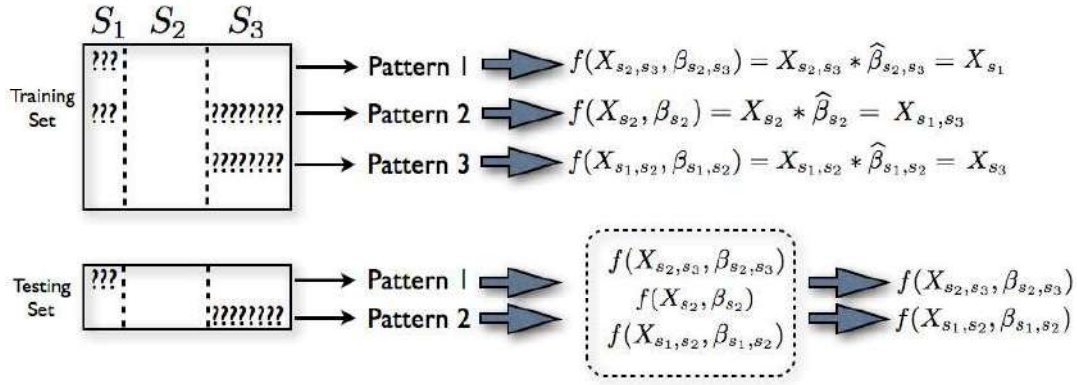

Figura 3.2: Cada patrón de MV tiene un modelo de regresión obtenido en el entrenamiento. Estos modelos se usan para imputar los MV de los datos de prueba segun el patrón respectivo.

donde X_{S_s} y $\widehat{\beta}_{S_s}$ son los datos de la fuente y los coeficientes regresores estimados, respectivamente, de la fuente S_s. De esta manera, el modelo de regresión $f(X_{S_s}, \widehat{\beta}_{S_s})$ construido en base a los datos de las fuentes observadas, imputa los datos de testing con un patrón de MV determinado. A modo de ejemplo, el modelo de imputación asociado al patrón 1 de MV en los datos de entrenamiento $f(X_{S_2,S_3}, \beta_{S_2,S_3}) = X_{S_1}$, permite imputar los MV de la fuente 1.

Aunque es poco frecuente, es posible que un patrón nuevo de MV aparezca en los datos de prueba. Si esto sucediera, el modelo de imputación creado de los datos de entrenamiento no podría realizar su tarea. Para resolver esto, el algoritmo propuesto crea un modelo de imputación acorde al patrón de MV nuevo usando los datos de entrenamiento. Utilizando estos datos, se crea el patrón de MV de manera sintética para posteriormente entrenar el algoritmo y generar el nuevo modelo de imputación. Para generar el patrón de MV sintético, se toma el 20 % de ejemplos de los datos de entrenamiento de manera uniforme, considerando que este 20 % incluya las distintas clases. De esta manera, el modelo de imputación se actualiza, agregando un nuevo modelo de regresión para futuras imputaciones. Detalles pueden ser revisados en Campos et al. (2018).

3.3. Propuesta 2: EMreg-oos con vecinos recomendados.

Esta versión actualizada del algortimo EMreg-oos incorpora varios aspectos adicionales que buscan mejorar la clasificación posterior al proceso de imputación. *EMreg-oos con vecinos recomendados (de ahora en adelante **EMreg-KNN**)* se basa en el mismo supuesto que la versión EMreg-oos, es decir, que los datos provienen de distintas fuentes, por lo tanto, se observan bloques de MV. Al trabajar con distintas fuentes de datos, el algoritmo creará vecindarios para cada dato ayudado de recomendaciones que las

otras fuentes haran.

La estructura general se puede dividir en 5 bloques los cuales se muestran en la figura 3.3:

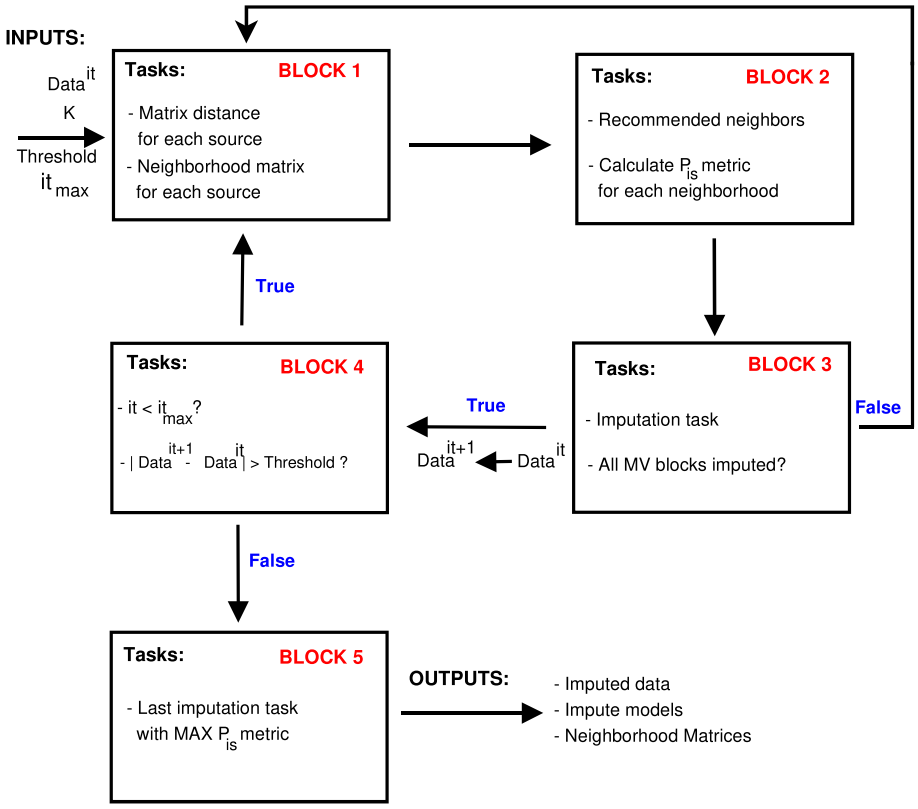

Figura 3.3: Estructura del algoritmo EMreg-KNN.

- **Block 1:** El algoritmo recibe como entrada la matriz de datos con MV X, los 2 criterios de parada: it_{max} y $threshold$, los cuales reflejan el número máximo de iteraciones del algoritmo y la diferencia mínima que debe tener la matriz de datos imputada de una iteración a la siguiente ($|X^{it+1} - X^{it}|$) respectivamente, y el hiperparámetro K que representa el número de vecinos.

  Por cada fuente s se calcula una matriz de distancias $MD_s \in \mathbb{R}^{N \times N}$ y una matriz de vecinos $MVe_s \in \mathbb{N}^{N \times K}$. MD_s estará incompleta si la fuente s tiene MV. MVe_s, indica para cada dato, que datos son vecinos.

- **Block 2:** una vez obtenidas MD_s y MVe_s, es posible obtener la lista de vecinos recomendados. Por cada bloque de MV perteneciente a la fuente s que se desea imputar, las otras fuentes recomiendan sus propios vecinos para crear el vecindario del bloque a imputar. La figura 3.4 explica el procedimiento para un caso de 3 fuentes y 4 ejemplos: el primer bloque de MV a imputar será el ejemplo 4 perteneciente a la fuente 1. Se calculan las matrices de distancia MD_s, las cuales

estarán incompletas mientras existan bloques de MV (solo en la primera iteración). Se calculan los vecindarios segun el valor de k, que en este caso $k = 2$. Cada fuente $s \neq 1$ recomienda los vecinos de su propia fuente para crear un vecindario al bloque que estamos imputando. En este caso, la fuente 2 recomienda los ejemplos 1 y 2, y la fuente 3 recomienda los vecinos 2 y 3. Con esto, el vecindario recomendado para el bloque a imputar se compone de los vecinos 2 y 3. El vecino 1 recomendado por la fuente 2 no se considera, porque ese ejemplo es MV en la fuente 1.

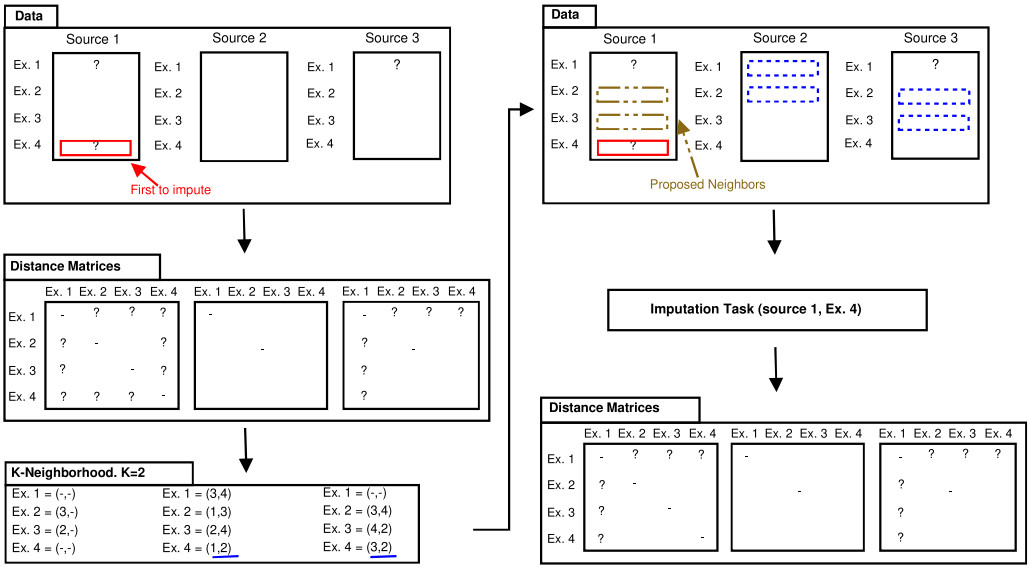

Figura 3.4: Procedimiento de calculo de vecinos recomendados. Ejemplo para 3 fuentes y 4 ejemplos.

Cada vecindario pertenece a un ejemplo, el cual tiene asociada una clase y_l. Entonces, se calcula una métrica $\pi(i, s)$ asociada al ejemplo i de la fuente s que mide la proporción de la clase del ejemplo en su vecindario. Al ser una proporción, se tiene que $0 \leq \pi(i, s) \leq 1$, donde 0 es cuando la clase y_l no esta presente entre los ejemplos que componen el vecindario y 1 cuando todos los ejemplos del vecindario tiene la misma clase y_l.

- **Block 3:** una vez teniendo los vecindarios construidos, estos se usarán para realizar la imputación. Por cada bloque de MV, se construirá una matriz de datos que incluya todas las fuentes del dato a imputar y su respectivo vecindario. Esto busca construir un modelo de imputación con solo los datos del vecindario.

  El modelo de imputación esta basado en la ecuación 3.7, donde es necesario almacenar los parámetros del modelo μ y β, para poder construir posteriormente el modelo Out-of-Sample. Este procedimiento se realiza con cada bloque de MV del dataset. Se tendrán tantos modelos de imputación como bloques de MV existan.

  El output de este bloque es el dataset imputado completamente, pasando el dataset de la versión it

a la $it + 1$.

- **Block 4:** luego que son imputados todos los bloques de MV, se verifican las condiciones de termino: $it < it_{max}$ y si $|data^{it+1} - data^{it}| > threshold$. En caso de que alguna de estas condiciones sea falsa, el algoritmo pasa al block 5. Si ambas condiciones son verdaderas, entonces el algoritmo vuelve al block 1 donde el input es $data^{it+1}$.

  Desde la iteración $it = 2$ no existen bloques de MV, por lo tanto, las matrices MD_s estan completas. No obstante, MVe puede aun estar incompleta debido al valor de k, que fue muy grande para encontrar ese número de vecinos para un ejemplo en particular. Desde la segunda iteración en adelante, se podrán encontrar todos los vecinos solicitados segun el valor de k. Es importante mencionar que iteración por iteración los vecindarios van cambiando segun las multiples imputaciones realizadas. Esto, hasta que se llega al número de iteraciones máxima it_{max} o hasta que el dataset converge a valores estables no superando el umbral $threshold$.

- **Block 5:** cuando el algoritmo pasa al block 5, se realiza una última imputación de todos los bloques de MV. Para realizar esto, se consideran los vecindarios con el $\pi(i, s)$ mayor. Con esto, se obtiene la versión final imputada de la data de entrenamiento.

  La matriz de vecindarios MVe es un resultado que podria ser valioso segun la tarea a realizar posteriormente. A modo de ejemplo, técnicas de Manifold Learning Van Der Maaten et al. (2009) como Isomap Tenenbaum et al. (2000), Local Linear Embedding (LLE) Roweis and Saul (2000) Saul et al. (2003) e incluso UMAP McInnes et al. (2018) utilizan los vecindarios de los ejemplos para poder reducir la dimensionalidad. Técnicas de Clustering como KNN o DBSCAN Ester et al. (1996) tambien usan información de vecinos. En general, algoritmos de Aprendizaje no Supervisado comúnmente pueden utilizar esta información.

Finalmente, se debe calcular el modelo de imputación que permitirá realizar las imputaciones en el dataset de prueba, es decir, el modelo Out-of-sample.

Para imputar cada ejemplo del dataset de prueba, primero se debe identificar el tipo de patrón de MV. Por cada patrón de MV V, existen M modelos de imputación asociados a cada uno de los ejemplos con ese patrón de MV durante el entrenamiento. Esos distintos modelos se usan para crear un Ensamblado que permite imputar el dato de test. La figura 3.5 muestra lo recién explicado para un patrón de MV V, donde cada uno de los modelos depende de su vecindario, un centroide y los coeficientes de regresión.

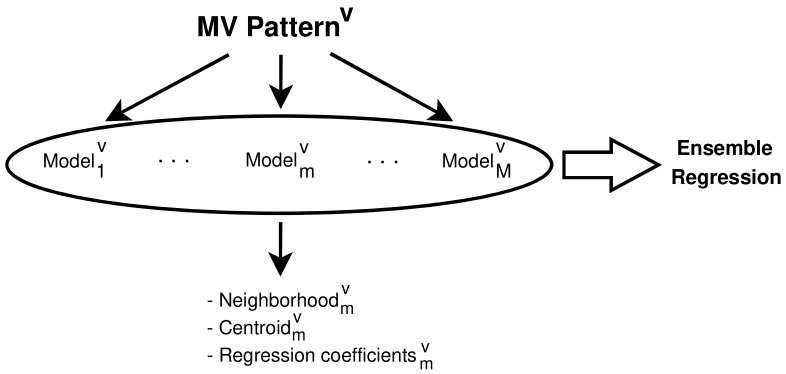

Figura 3.5: La versión Out-of-sample se realiza mediante un ensamblado de módelos de regresiones que fueron construidos durante la fase de entrenamiento.

El centroide de cada modelo es un vector representativo del vecindario construido en base a la media de cada variable, es decir, es un vector de medias. Este vector, el cual llamaremos r_m^V, será el vector representante del modelo m con el patrón de MV V y se utilizará para calcular la distancia entre este vector y el vector de prueba x_i^{test} que se debe imputar. El cálculo de la distancia es realizada segun la ecuacion 3.8:

$$d_{mt}^V = D_*(r_m^V, x_t^{test}) \tag{3.8}$$

donde $D_*(\cdot)$ es una función de distancia aplicada solo a las variables con datos observados del vector de test y el centroide del modelo m con patrón de MV V. En este caso usaremos la distancia Euclidiana, pero podria ser otro tipo de métrica de distancia. Entonces, la imputación final del dato de test considerando que el patrón de MV V tiene M modelos de imputación es dado por la ecuación 3.9:

$$\begin{aligned} x_i &= \sum_{m=1}^{M} W_{mi}^V \cdot f_m^V(X, \beta) \\ W_{mi}^V &= \frac{(1 - H_{mi}^V)}{M - 1} \\ H_{mi}^V &= \frac{d_{mi}^V}{\sum_{m=1}^{M} d_{mi}^V} \end{aligned} \tag{3.9}$$

donde W_{mi}^V es un factor de ponderación asociado al modelo de imputación m con respecto al ejemplo de test i, $f_m^V(\cdot)$ es el modelo de regresión m mostrado en la figura 3.2, que se basa en la ecuación 3.1. Ambos con patrón de MV V.

H_{mi}^V es un factor de ponderación, que al igual que W_{mi}^V, cumple con la condición: $\sum_{m=1}^{M} H_{mi}^V = 1$. La diferencia entre ambos es que H_{mi}^V pondera más cuando la distancia d_{mi}^V es mayor, y W_{mi}^V

pondera más cuando la distancia d_{mi}^{V} es menor. De esta manera, cada modelo de regresión será ponderado con un factor mayor entre más cercano sea el dato de test i al centroide r_{m}^{V}.
El denominador de W_{mi}^{V} es debido a que como $\sum_{m=1}^{M} H_{mi}^{V} = 1$, $\sum_{m=1}^{M}(1-H_{mi}^{V}) = M-\sum_{m=1}^{M} H_{mi}^{V} = M-1$.

La etapa de entrenamiento del método **EMreg-KNN** es descrito en el algoritmo 1:

La investigación reproducible es fundamental para garantizar la transparencia, confiabilidad y validez de los resultados científicos. Al permitir que otros investigadores puedan replicar los experimentos, analizar los mismos datos y verificar los métodos utilizados se han dejado disponible códigos y datos utilizados en esta tesis en el siguiente sitio web Zenodo.org

Algorithm 1: Algoritmo EMreg-Knn: etapa de entrenamiento

Input: Datos con MV $X^{it=0}$, número de vecinos más cercanos k, número de iteraciones máxima It_{max} y umbral de aceptación $Threshold$.

Output: Data imputada X, conjunto de modelos de imputación $\mathbb{F}$ y matrices de vecinos mas cercanos MVe_s.

$it \leftarrow 0$;
$flag \leftarrow True$;
$\mathbb{F} \leftarrow \{f^1, ..., f^v, ..., f^V\}$; /* Conjunto de modelos de imputación */
$\mathbb{P}_{max} \leftarrow \{()^{is}\}$; /* Conjunto que guarda la métrica mayor y su vecindario por cada bloque de MV */
while $it \leq It_{max} \wedge flag$ **do**
    $X^{it+1} \leftarrow X^{it}$;
    foreach *bloque de MV en la fila i fuente s* **do**
        $MD_s \leftarrow$ Calcular matrices de distancias por fuente;
        $MVe_s \leftarrow$ Calcular matrices de vecinos por fuente;
        $R_{neighbors} \leftarrow$ Calcular vecinos recomendados;
        $P_{is} \leftarrow$ Calcular métrica a $R_{neighbors}$;
        $W \leftarrow x^{it}_{i:} \cup R_{neighbors}$;
        $W \leftarrow f^v_m(W)$; /* Tarea de imputación */
        $X^{it+1}_i \leftarrow w_{1:}$; /* Reemplaza vector imputado en X^{it+1} */
        Agregar f^v_m en $\mathbb{F}$;
        if $\pi(i,s) \geq \mathbb{P}_{max}(\pi(i,s))$ **then**
            Agregar $(\pi(i,s), W)$ en $\mathbb{P}_{max}$;
        end
    end
    if $|X^{it+1} - X^{it}| \leq Threshold$ **then**
        $flag \leftarrow False$;
    end
    $it \leftarrow it + 1$;
end
foreach *bloque de MV en la fila i fuente s* **do**
    $W \leftarrow \mathbb{P}^{is}_{max}$;
    $W \leftarrow f^v_m(W)$; /* Tarea de imputación final */
    $X_i \leftarrow w_{1:}$;
end
$MVe_s \leftarrow$ Calcular matrices de vecinos por fuente;
return $X, \mathbb{F}, MVe_s$

Capítulo 4

Estudio experimental y resultados

Por cada set de datos, se realizó un proceso de estandarización a los datos de entrenamiento y prueba. Considerando que toda la estandarización debe cumplir con la estrategia Out-of-Sample, los datos de prueba se estandarizaron segun la ecuación 4.1:

$$X_{std,j}^{test} = \frac{X_{raw,j}^{test} - \mu_j^{train}}{\sigma_j^{train}} \tag{4.1}$$

donde μ_j^{train} y σ_j^{train} representa la media y desviación estandar de la variable j-ésima calculada de los datos de entrenamiento y $X_{raw,j}^{test}$ es el dato de prueba de la variable j-ésima en la escala original .

Teniendo los datos estandarizados, se procede a realizar la imputación de estos, utilizando ambas propuestas: *EMreg-oos* y *EMreg-KNN*, a modo de compararlas con el algoritmo original *EMreg*.
Además, los resultados se compararán con el rendimiento de los siguientes algoritmos:

- MissForest Stekhoven and Bühlmann (2012): utilizando una implementación en python (missingpy [1]) que permite crear modelos de imputación para aplicarlos al testing set. Se utilizan los parámetros por defecto con un número de árboles igual a 100 y cantidad máxima de iteraciones 10.
- MICE Raghunathan et al. (2001): utilizando una implementación en python que permite crear modelos de imputación para aplicarlos al testing set. Se utilizan los parámetros por defecto.
- SVD Troyanskaya et al. (2001): utilizando una implementación en python del paquete *fancyimpute* que permite imputar matrices con datos faltantes. Este método no crea un modelo de imputación que permita imputar un testing set, por lo que se debe aplicar el algoritmo de manera separada

[1] https://github.com/epsilon-machine/missingpy

tanto en el testing como en el training set.

En cada experimento, se dividió el dataset en 75 % para entrenamiento y 25 % para prueba. Los clasificadores usados fueron: K-Nearest Neighbors (K-NN), ν-Support Vector Machine (ν-SVM), Random Forest (RF) y Artificial Neural Networks (ANN). Las implementaciones usadas fueron utilizadas de la libreria scikit-learn [2] para los primeros 3 clasificadores y TensorFlow [3] para ANN.

La sintonización de hiperparámetros para los algoritmos de clasificación fue realizada utilizando 5-fold Cross Validation, división estratificada de los datos y búsqueda en grilla (Grid Search). Los hiperparámetros sintonizados y sus valores se detallan a continuación:

- K-NN: el número de vecinos K y la función de ponderación $weights$. K en un rango entre 10 y $\lfloor num\_subject/num\_classes \rfloor$ lo cual busca relacionar la cantidad de datos con el número de clases. Para $weights$ se utilizo $uniform$ y $distance$ lo cual refleja como se pondera las distancias.
- ν-SVM: ν en un rango $(0, 1]$, para $kernel$ se utilizó kernel $polinomial$ y rbf y $degree$ = $[1, 2, 3, 4, 5]$ el cual es el grado del polinomio para kernel polinomial. Este último se omite cuando se utiliza otro tipo de kernel.
- RF: el número de árboles $n\_estimators$ = $[10, 50, 100, 150, 200]$ y el número de variables a considerar para realizar la mejor división
  $max\_features$ = $[\lfloor n\_var/8 \rfloor, \lfloor n\_var/4 \rfloor, \lfloor n\_var/2 \rfloor, \lfloor n\_var \cdot (3/4) \rfloor]$, donde n_var es el número de variables de los datos.
- ANN: el valor de drop-out es $dropout\_rate$ = $[0.0, 0.1, 0.3, 0.5, 0.7, 0.9]$. La arquitectura de la red, la cual se dejo fija después de probar varias configuraciones, esta compuesta de 2 capas ocultas, ambas con cantidad de neuronas igual a $n\_var \cdot 1.2$ con función de activacion $Relu$. La capa de salida es de una solo neurona con función de activación $sigmoid$. La red utiliza como función de pérdida $binary\ cross\ entropy$, algoritmo de optimización $Adam$ y número de epocas 10.

Para los casos multi-clase, se utilizó la estrategia $One\ vs\ Rest$ y las métricas de rendimiento son basadas en análisis ROC. Para detalles de las métricas puede revisar el capítulo 2, sección 2.2.

[2] scikit-learn.org/stable
[3] https://www.tensorflow.org

4.1. ADNI

Con el proposito de ilustrar de mejor manera el rendimiento de nuestra propuesta, se considerá 3 fuentes de información: Fluido cerebro espinal (CSF), Imagen de resonancia magnética (MRI) y Tomografia por emisión de positrones (PET). Los datos son preprocesadas segun Gray et al. (2013), sacando 43 de los 819 sujetos por no pasar el control de calidad. La fuente de informacion CSF contiene 3 variables que miden el nivel de algunas proteinas y aminoácidos que estan involucrados con la enfermedad. La fuente de información MRI tiene 83 variables que representan el volumen de regiones de interes del cerebro. Por último, la fuente de información PET, tiene 83 variables que representan el metabolismo cerebral asociado a la glucosa de las mismas regiones de interes que se estudian con MRI. De esta manera, cada sujeto contiene 169 variables. El cuadro 4.1 muestra detalles de como se distribuyen los datos de los grupos (Healthy Control HC, stable Mild Cognitive Impairment sMCI, progressive Mild Cognitive Impairment pMCI y Alzheimer Disease AD) a los que pertenecen los pacientes segun el avance de la enfermedad:

	ADNI	CSF	MRI	PET
AD	185	85	0	114
HC	210	107	0	141
pMCI	164	80	0	102
sMCI	217	114	0	132
Total	776	386	0	489

Cuadro 4.1: La primera columna muestra el numero de datos por cada clase. Las otras columnas muestra la cantidad de datos con MV por fuente de información. Sujetos con clase MCI pueden dividirse en 2 grupos: los que se mantienen en una condición estable de la enfermedad (sMCI: stable MCI), y los que no (pMCI: progressive MCI)

Se considerarán 4 problemas de clasificación, los cuales se dividen segun las clases involucradas. Se agrega información sobre el número de instancias por clase antes y luego de descartar los ejemplos con MV. El detalle se muestra en el cuadro 4.2:

A modo de complementar los resultados, para cada escenario se incluye las métricas obtenidas al trabajar solo con datos observados. Esto implica eliminar los datos que tengan MV, reduciendo la muestra de 395 a 72 para AD vs HC, de 591 a 110 para MCI vs HC, de 381 a 75 para pMCI vs sMCI y de 776 a 151 para AD vs HC vs MCI. Este caso se ve reflejado en las tablas de resultados como $None$ en la columna $Imputation$.

Problema	# Instancias (# Instancias por clase)	# Instancias después que los MV son descartados (# Instancias por clase)
(a) AD vs HC	395 (185 AD - 210 HC)	72 (37 AD - 35 HC)
(b) MCI vs HC	591 (381 MCI - 210 HC)	110 (75 MCI - 35 HC)
(c) pMCI vs sMCI	381 (164 pMCI - 217 sMCI)	75 (34 pMCI - 41 sMCI)
(d) AD vs HC vs MCI	776 (185 AD - 210 HC - 381 MCI)	147 (37 AD - 35 HC - 75 MCI)

Cuadro 4.2: Cantidad de instancias por clase en cada uno de los cuatro problemas de clasificación. A modo de ejemplo, el problema AD vs HC tiene 395 instancias, pero solo 72 si las instancias con MV son descartados. Entre paréntesis se muestra el grado de desbalance.

Los cuadros 4.3, 4.4, 4.5 y 4.6 muestran los resultados de los escenarios anteriomente descritos, donde los valores en negrita refleja los mejores resultados por clasificador, y con asterisco (*) cuando la diferencia entre el algoritmo original *EMreg* y el algoritmo con mejor resultado es estadísticamente significativa. Esto se realiza ya que nos interesa si mejoramos la versión original *EMreg*. Para esto último, se aplicó un test de diferencia de medias t-Student con $\alpha = 0.05$.

Classifier	Imputation	Acc. (%)	AUC (%)	Sens. (%)	Spec. (%)	F (%)
	none	80.6(8.6)	92.1(6.8)	91.1(9.2)	72.2(13.9)	81.3(9.1)
	MICE	73.7(5.8)	83.5(5.7)	86.8(6.1)	58.4(10.9)	78.1(4.9)
	SVDimpute	84.1(3.6)	**92.4(3.0)**	**95.0(2.6)**	71.4(7.6)	86.6(3.2)
	MissForest	83.4(4.1)	91.9(3.1)	93.8(3.2)	72.2(8.0)	86.3(3.6)
K-NN	EMreg	83.3(4.3)	91.3(3.8)	94.5(3.5)	70.1(9.1)	86.0(3.7)
	EMreg-oos	84.4(3.8)	91.4(3.8)	94.3(3.1)	72.8(7.0)	86.7(3.3)
	EMreg-KNN	**85.7(3.3)***	92.1(2.9)	92.8(3.3)	**77.5(6.6)***	**87.5(3.0)***
	none	85.4(9.2)	93.7(5.8)	84.1(14.3)	87.0(11.8)	83.8(11.7)
	MICE	78.7(3.4)	89.4(2.7)	78.6(8.9)	79.5(12.0)	79.9(3.3)
	SVDimpute	88.1(3.4)	94.1(2.4)	90.8(4.2)	85.2(5.7)	89.2(3.2)
	MissForest	87.9(3.0)	93.9(2.4)	89.9(4.3)	85.8(6.2)	88.9(2.9)
SVM	EMreg	88.2(3.3)	94.0(2.4)	91.5(4.4)	84.5(6.3)	89.3(3.2)
	EMreg-oos	88.6(3.0)	94.1(2.3)	**91.9(3.1)**	85.0(5.6)	89.7(2.9)
	EMreg-KNN	**89.1(2.6)**	**94.5(2.2)**	90.5(3.2)	**87.7(5.6)***	**90.0(2.4)**
	none	82.7(8.9)	91.9(6.5)	84.6(13.4)	82.2(12.7)	81.8(10.0)
	MICE	86.2(3.3)	92.9(2.7)	87.3(4.3)	85.0(5.9)	87.3(3.1)
	SVDimpute	86.3(3.6)	93.3(2.5)	87.9(5.2)	84.7(5.6)	87.4(3.3)
	MissForest	83.3(4.5)	92.0(2.7)	81.5(7.9)	85.8(8.6)	84.0(4.4)
RF	EMreg	84.6(3.8)	92.9(2.6)	87.2(7.5)	81.5(10.2)	85.9(3.8)
	EMreg-oos	**86.8(3.2)***	**93.5(2.4)**	**88.0(4.5)**	85.7(6.4)	**87.8(2.9)***
	EMreg-KNN	86.5(3.8)	93.2(2.4)	86.4(5.0)	**86.9(5.5)***	87.4(3.5)
	none	84.9(8.6)	**94.0(6.4)**	84.9(11.7)	85.4(12.1)	83.7(9.5)
	MICE	79.1(3.8)	78.8(4.4)	81.6(5.0)	76.3(7.4)	80.8(3.9)
	SVDimpute	87.5(3.5)	92.7(2.7)	**89.9(3.9)***	85.0(6.6)	88.6(3.4)
	MissForest	86.7(3.9)	92.8(2.8)	87.0(6.5)	86.8(6.3)	87.6(4.0)
ANN	EMreg	85.7(4.1)	91.7(3.1)	87.1(7.9)	84.2(6.6)	86.7(4.3)
	EMreg-oos	87.6(3.3)	92.6(2.5)	89.5(4.7)	85.4(5.9)	88.6(3.3)
	EMreg-KNN	**88.1(2.9)***	93.7(2.2)*	89.3(4.0)	**86.9(5.3)***	**89.1(2.7)***

Cuadro 4.3: Resultados del escenario AD vs HC, basado en 50 ejecuciones distintas de training y testing set. Se muestra Accuracy (acc.), Area Under the Curve (AUC), Sensitivity (sens.), Specificity (spec.) y F-measure (F). Resultados son expresados en media (desviación estándar).

En el escenario AD vs HC se puede observar que *EMreg-KNN* permite obtener mejores resultados en 3 de los 4 clasificadores considerando Accuracy y F-measure. Solo con RF *EMreg-oos* es mejor, pero de manera poco significativa (0.3 %). No obstante, la mayor Specificity se alcanza con EMreg-KNN, lo que denota una mayor capacidad para distinguir a los sujetos sanos. Las 2 propuestas mejoran al algoritmo original *EMreg* en todas las métricas. Al observar la desviación estándar, podemos decir que en casi todos los casos, nuestra propuesta tiene el valor más bajo. Esto significa que nuestro algoritmo es más estable.

Classifier	Imputation	Acc. (%)	AUC (%)	Sens. (%)	Spec. (%)	F (%)
	none	70.9(7.8)	72.5(10.2)	32.8(16.7)	**89.4(8.4)***	39.8(16.7)
	MICE	67.7(3.8)	71.2(8.0)	41.3(17.9)	83.0(7.7)	45.3(16.0)
	SVDimpute	70.6(3.5)	**76.5(3.3)**	58.3(9.7)	78.0(6.0)	58.4(5.6)
	MissForest	70.2(3.1)	75.4(3.5)	55.4(9.5)	78.9(5.4)	56.9(5.7)
K-NN	EMreg	69.5(4.6)	75.8(4.0)	53.7(13.1)	79.0(7.2)	55.1(9.6)
	EMreg-oos	70.0(4.0)	75.8(3.8)	56.9(9.9)	77.9(6.0)	57.4(6.4)
	EMreg-KNN	**70.9(3.9)**	75.9(3.4)	**58.3(9.5)***	78.4(5.5)	**58.8(8.1)***
	none	69.8(11.4)	68.8(18.4)	47.8(22.4)	81.0(15.6)	48.0(17.9)
	MICE	70.6(3.9)	72.7(11.8)	51.3(13.3)	**81.6(5.9)**	54.7(9.9)
	SVDimpute	70.8(9.3)	74.6(14.5)	56.9(10.6)	78.8(13.1)	58.5(8.8)
	MissForest	70.3(4.9)	74.5(13.9)	55.2(17.7)	79.3(12.0)	55.9(10.2)
SVM	EMreg	70.5(3.8)	76.4(8.2)	53.3(13.8)	80.5(8.7)	55.8(7.6)
	EMreg-oos	71.6(4.1)	75.9(11.8)	57.3(13.4)	80.0(7.9)	58.6(7.7)
	EMreg-KNN	**72.1(3.4)***	**77.6(8.3)**	**59.2(13.7)***	79.5(6.9)	**59.6(8.1)***
	none	72.1(7.2)	72.4(10.8)	40.6(17.7)	**87.3(6.9)**	46.1(17.2)
	MICE	72.3(3.2)	77.3(3.1)	51.7(7.5)	84.1(3.8)	57.1(5.4)
	SVDimpute	72.9(3.1)	**78.4(3.1)**	51.0(7.6)	85.4(4.4)	57.3(5.6)
	MissForest	70.8(4.1)	76.2(3.7)	43.1(13.8)	86.7(5.6)	50.3(10.5)
RF	EMreg	71.2(4.3)	76.8(3.7)	44.1(15.5)	86.6(6.5)	50.7(13.2)
	EMreg-oos	72.1(3.4)	77.7(3.4)	52.8(7.0)	83.2(4.4)	57.5(5.4)
	EMreg-KNN	**73.4(3.0)***	78.1(3.1)	**54.7(6.3)***	84.2(3.9)	**59.6(4.2)***
	none	70.9(7.2)	74.0(10.2)	43.0(23.4)	**84.8(8.0)***	44.7(20.5)
	MICE	70.2(4.2)	73.3(7.4)	54.1(16.8)	79.5(8.6)	54.7(15.1)
	SVDimpute	72.6(3.5)	78.5(3.6)	60.2(9.1)	79.8(4.4)	61.0(6.8)
	MissForest	71.8(3.5)	77.9(3.7)	55.1(11.2)	81.3(4.9)	57.9(7.3)
ANN	EMreg	71.5(4.4)	77.7(3.7)	55.9(11.8)	80.5(5.7)	58.0(8.4)
	EMreg-oos	72.6(3.6)	78.8(3.6)	60.1(9.6)	80.0(4.9)	60.9(6.2)
	EMreg-KNN	**73.3(3.3)***	**79.4(3.2)***	**61.8(9.5)***	80.1(4.5)	**62.2(5.6)***

Cuadro 4.4: Resultados del escenario MCI vs HC, basado en 50 ejecuciones distintas de training y testing set. Se muestra Accuracy (acc.), Area Under the Curve (AUC), Sensitivity (sens.), Specificity (spec.) y F-measure (F). Resultados son expresados en media (desviación estándar).

En el escenario MCI vs HC se puede observar que *EMreg-KNN* obtiene los mejores resultados con todos los clasificadores en Accuracy, Sensitivity y F-measure. La diferencia es estadísticamente significativa para 3 de los 4 clasificadores. A pesar del valor relativamente bajo de la Sensitivity, es importante notar que la propuesta muestra la mayor capacidad para identificar a los sujetos enfermos. Considerando la métrica AUC, superamos en 2 de los 4 clasificadores, pero las diferencias entre el mejor resultado y nuestra propuesta son muy pequeñas en los otros 2 clasificadores. También debemos considerar que este escenario tiene un grado de desbalance, por lo que debemos examinar más de cerca la F-measure. En esta métrica, como se mencionó antes, tenemos el mejor resultado en todos los casos. Considerando la desviación estándar, nuestra propuesta es mejor notoriamente solo en RF y ANN, pero comparando nuestra propuesta con el EMreg, se observa una disminución prácticamente en todos los casos.

Classifier	Imputation	Acc. (%)	AUC (%)	Sens. (%)	Spec. (%)	F (%)
	none	54.6(8.8)	59.7(9.8)	**53.2(18.2)***	60.9(19.5)	54.9(12.5)
	MICE	62.4(4.8)	66.1(5.8)	42.0(12.3)	**79.0(9.2)**	48.0(8.8)
	SVDimpute	63.9(4.9)	69.9(5.2)	45.9(12.0)	78.2(8.4)	51.4(8.1)
	MissForest	63.5(5.5)	69.3(5.0)	47.4(11.4)	76.1(8.2)	52.1(8.0)
K-NN	EMreg	63.9(4.5)	70.0(4.7)	45.5(11.3)	78.2(8.0)	51.2(8.0)
	EMreg-oos	63.5(4.6)	70.1(4.4)	46.5(11.4)	77.0(8.6)	51.5(7.5)
	EMreg-KNN	**65.3(4.8)**	**71.1(4.9)**	50.8(9.2)	76.8(7.8)	**55.4(5.8)***
	none	52.0(9.5)	51.4(12.0)	**59.2(24.7)***	46.7(26.3)	55.1(14.8)
	MICE	62.3(4.5)	67.2(7.8)	50.4(12.9)	71.9(10.3)	52.7(7.5)
	SVDimpute	63.4(3.8)	69.5(4.4)	51.9(9.9)	**72.4(7.3)**	54.4(6.2)
	MissForest	62.3(5.7)	67.1(9.8)	54.4(14.3)	68.9(13.1)	54.4(8.1)
SVM	EMreg	62.9(4.4)	65.2(10.9)	51.7(10.2)	71.7(8.1)	54.0(6.4)
	EMreg-oos	63.5(4.1)	67.7(7.6)	53.7(8.6)	71.2(6.6)	55.5(5.5)
	EMreg-KNN	**64.1(5.0)**	**70.6(4.4)***	56.6(12.0)	70.3(10.8)	**57.0(6.4)***
	none	57.1(8.8)	61.9(10.1)	**64.3(15.4)***	51.3(16.4)	**61.5(9.3)***
	MICE	62.4(4.8)	67.9(5.2)	49.6(8.2)	72.5(7.0)	52.8(5.8)
	SVDimpute	63.0(4.5)	69.4(4.5)	52.0(9.2)	72.0(7.0)	54.3(5.3)
	MissForest	62.1(4.8)	67.6(4.5)	52.9(13.5)	69.9(9.8)	53.6(8.0)
RF	EMreg	62.3(4.7)	68.3(4.8)	48.2(12.6)	**73.2(9.6)**	51.5(8.1)
	EMreg-oos	63.0(4.4)	70.1(4.7)	54.2(8.4)	70.1(5.8)	55.5(5.6)
	EMreg-KNN	**64.9(4.6)***	**71.2(4.6)***	54.9(7.9)	72.9(6.4)	57.1(5.3)
	none	58.2(10.1)	66.5(10.6)	**68.1(18.9)***	50.1(22.7)	**63.2(10.3)***
	MICE	62.8(5.2)	68.1(4.7)	54.3(12.4)	69.9(9.0)	54.8(10.0)
	SVDimpute	64.6(3.9)	70.0(4.8)	56.1(9.5)	71.5(6.9)	57.2(6.0)
	MissForest	63.3(3.9)	68.4(4.3)	55.1(11.3)	69.9(8.5)	55.7(6.3)
ANN	EMreg	64.2(4.2)	68.7(3.8)	52.9(11.5)	**72.8(8.2)**	55.2(8.1)
	EMreg-oos	64.1(4.3)	69.5(4.1)	54.6(11.1)	71.7(7.9)	55.8(9.3)
	EMreg-KNN	**65.2(4.1)**	**70.7(3.8)***	57.3(8.5)	71.6(7.5)	58.3(5.2)

Cuadro 4.5: Resultados del escenario pMCI vs sMCI, basado en 50 ejecuciones distintas de training y testing set. Se muestra Accuracy (acc.), Area Under the Curve (AUC), Sensitivity (sens.), Specificity (spec.) y F-measure (F). Resultados son expresados en media (desviación estándar).

En el escenario pMCI vs sMCI se puede observar que *EMreg-KNN* logra los mejores resultados en las métricas de Accuracy y AUC con todos los clasificadores. Considerando el AUC, es estadísticamente significativo en SVM, RF y ANN. La Sensitivity es mayor para el caso *none* en todos los clasificadores. Este es un resultado extraño y el único en todos nuestros experimentos. Algo similar ocurrió con la F-measure. Claramente, el impacto de eliminar ejemplos con MV favorece el cálculo de estas métricas. A pesar de esto, como se destacó anteriormente, Accuracy es mejor en nuestra propuesta, y en un escenario con tan poco desbalance, esto tiene más importancia. En cuanto a la desviación estándar, la único claro es que el caso *none* tiene el valor más alto en todas las instancias. Esto disminuye el valor de los buenos resultados que este caso logró en algunas métricas. No hay un claro ganador entre los métodos de imputación en términos de su estabilidad.

Classifier	Imputation	Acc. (%)	AUC (%)	Sens. (%)	F (%)
	none	50.7(9.7)	53.2(6.5)	50.7(9.7)	42.5(11.0)
	MICE	53.2(3.4)	58.1(2.3)	53.2(3.4)	47.5(4.3)
	SVDimpute	54.0(3.2)	59.1(2.3)	54.0(3.2)	48.6(4.1)
	MissForest	53.3(3.8)	58.5(2.7)	53.3(3.8)	48.1(4.8)
K-NN	EMreg	53.0(3.4)	57.8(2.6)	53.0(3.4)	47.2(4.6)
	EMreg-oos	53.9(3.5)	59.2(2.5)	53.9(3.5)	48.8(4.4)
	EMreg-KNN	**54.5(3.3)***	**59.7(2.6)***	**54.5(3.3)***	**49.9(4.6)***
	none	53.1(10.5)	59.0(8.4)	53.1(10.5)	50.7(11.8)
	MICE	54.0(3.7)	61.0(3.2)	54.0(3.7)	52.9(4.5)
	SVDimpute	54.7(3.8)	61.5(3.2)	54.7(3.8)	53.7(4.3)
	MissForest	54.5(4.0)	60.9(3.2)	54.5(4.0)	53.1(4.3)
SVM	EMreg	54.8(4.0)	60.8(3.1)	54.8(4.0)	52.9(4.4)
	EMreg-oos	55.5(4.0)	62.3(3.1)	55.5(4.0)	54.6(4.2)
	EMreg-KNN	**56.5(3.3)***	**63.3(2.8)***	**56.5(3.3)***	**55.8(3.5)***
	none	55.5(8.5)	60.6(7.1)	55.5(8.5)	53.9(8.7)
	MICE	57.2(3.5)	63.1(2.9)	57.2(3.5)	56.0(3.6)
	SVDimpute	57.4(3.7)	62.9(3.1)	57.4(3.7)	56.0(3.9)
	MissForest	56.5(3.6)	61.7(2.9)	56.5(3.6)	54.6(4.2)
RF	EMreg	55.8(3.7)	60.2(3.1)	55.8(3.7)	52.5(4.6)
	EMreg-oos	57.3(3.5)	63.4(2.8)	57.3(3.5)	56.5(3.5)
	EMreg-KNN	**58.0(3.8)***	**64.4(3.1)***	**58.0(3.8)***	**57.5(3.9)***
	none	52.5(9.1)	58.4(7.0)	52.5(9.1)	49.1(10.9)
	MICE	55.0(3.0)	62.2(2.8)	55.0(3.0)	54.2(3.5)
	SVDimpute	56.2(3.1)	63.0(3.3)	56.2(3.1)	55.4(3.8)
	MissForest	56.0(3.4)	62.7(3.6)	56.0(3.4)	55.0(4.4)
ANN	EMreg	55.1(3.4)	61.4(3.2)	55.1(3.4)	53.7(4.7)
	EMreg-oos	55.5(2.9)	62.6(2.5)	55.5(2.9)	55.0(3.0)
	EMreg-KNN	**56.3(3.1)**	**63.7(2.4)***	**56.3(3.1)***	**55.8(3.1)***

Cuadro 4.6: Resultados del escenario AD vs HC vs MCI, basado en 50 ejecuciones distintas de training y testing set. Se muestra Accuracy (acc.), Area Under the Curve (AUC), Sensitivity (sens.), Specificity (spec.) y F-measure (F). Resultados son expresados en media (desviación estándar).

En el escenario AD vs HC vs MCI se puede observar que *EMreg-KNN* permite obtener mejores resultados con todos los clasificadores en todas las métricas.
Con respecto a la desviación estándar, se observa que las propuestas tiene una mejora sostenida con respecto a *EMreg*.

En general, se observa que los métodos de imputación mejoran los resultados de clasificación al considerar todos los datos disponibles, apesar de poder incluir algun sesgo en los datos. Los resultados con datos sin imputar (*none*) demuestran una baja sostenibles, y cosiderable en muchos casos, en todas las métricas.
Con respecto a la desviación estándar, se muestra una baja sostenible, y en algunos casos considerabale, gracias a los métodos de imputación. Ambas propuestas obtienen, en general, mayor estabilidad (baja desviación estándar) que el algoritmo original *EMreg*.
La propuesta *EMreg-KNN* demuestra ser significativamente mejor en muchos casos (casos con *).

4.2. Handwritten digits

Handwritten [4] es un data set clásico de dígitos manuscritos. Son imágenes binarias de los dígitos del 0 al 9, donde cada dígito tiene 200 imágenes, y por lo tanto, el data set completo esta compuesto por 2000 imágenes. Las variables se dividen en 6 fuentes, cada una con el siguiente número de variables :

- 76 coeficientes de Fourier.
- 216 correlaciones de perfil (profile correlations).
- 64 coeficiente de Karhunen-Love.
- 240 promedio de pixeles de ventanas de 2×3.
- 47 momentos de Zernike.
- 6 características morfológicas.

Para estos experimentos, se tomaron tres fuentes: Coeficientes de Fourier, coeficientes de Karhunen-Love y momentos de Zernike, obteniendo un total de 187 variables.
Este set de datos no tienen, de manera natural, datos faltantes. Por este motivo, se procedio a crear patrones de MV en diferentes porcentajes: 20 %, 40 % y 60 %.
Aleatoriamente, se elijen los ejemplos que serán afectados por MV. Una vez que es elejido un ejemplo, aleatoriamente se elije el patrón de MV que se le aplicará. La aleatoriedad esta basada en una distribución

[4] https://archive.ics.uci.edu/ml/datasets/Multiple+Features

uniforme.

Los cuadros 4.7 y 4.8 muestran los resultados de los escenarios anteriomente descritos. Al igual que con ADNI, el caso *none* representa la clasificación con los datos sin MV, esperando que sea la cota superior en rendimiento. Los resultados en negrita representan el mejor resultado para cada clasificador y el resultado con asterisco (*) si la diferencia entre EMreg y el valor es estadísticamente significativa. Los resultados son obtenidos sobre 30 corridas, expresando la media y desviación estándar.

Cuadro 4.7: Resultados de Handwritten digits basado en 30 ejecuciones distintas de training y testing set. Se muestra Accuracy (acc.), Area Under the Curve (AUC), Sensitivity (sens.) y F-measure (F). Resultados son expresados en media (desviación estándar).

Classifier	Imputation	MV rate (%)	Acc. (%)	AUC (%)	Sens. (%)	F (%)
K-NN	*none*	0	97.2(0.7)	98.5(0.4)	97.2(0.7)	97.2(0.7)
	MICE	20	93.6(1.1)	96.4(0.6)	93.6(1.1)	93.5(1.0)
	SVDimpute	20	92.7(1.3)	95.9(0.7)	92.7(1.3)	92.7(1.2)
	MissForest	20	92.2(1.3)	95.7(0.7)	92.2(1.3)	92.2(1.2)
	EMreg	20	93.5(1.1)	96.4(0.6)	93.5(1.1)	93.5(1.1)
	EMreg-oos	20	94.0(1.0)	96.6(0.6)	94.0(1.0)	93.9(1.0)
	EMreg-KNN	20	**94.0(1.0)**	**96.7(0.6)**	**94.0(1.0)**	**94.0(1.0)**
	MICE	40	91.1(1.2)	95.0(0.7)	91.1(1.2)	91.0(1.2)
	SVDimpute	40	89.5(1.5)	94.2(0.9)	89.5(1.5)	89.4(1.5)
	MissForest	40	88.5(1.5)	93.6(0.8)	88.5(1.5)	88.6(1.5)
	EMreg	40	90.8(1.2)	94.9(0.7)	90.8(1.2)	90.8(1.2)
	EMreg-oos	40	91.4(1.3)	95.2(0.7)	91.4(1.3)	91.3(1.3)
	EMreg-KNN	40	**91.6(1.2)***	**95.3(0.7)***	**91.6(1.2)**	**91.5(1.2)***
	MICE	60	89.2(1.7)	94.0(1.0)	89.2(1.7)	89.2(1.7)
	SVDimpute	60	87.3(1.7)	92.9(0.9)	87.3(1.7)	87.2(1.7)
	MissForest	60	86.2(1.8)	92.3(1.0)	86.2(1.8)	86.2(1.8)
	EMreg	60	88.3(1.3)	93.5(0.7)	88.3(1.3)	88.3(1.3)
	EMreg-oos	60	89.5(1.3)	94.2(0.7)	89.5(1.3)	89.4(1.4)
	EMreg-KNN	60	**90.3(1.5)***	**94.6(0.9)***	**90.3(1.5)**	**90.2(1.6)***
	none	0	98.3(0.5)	99.1(0.3)	98.3(0.5)	98.3(0.5)
	MICE	20	96.5(0.8)	98.0(0.4)	96.5(0.8)	96.5(0.8)
	SVDimpute	20	96.2(0.8)	97.9(0.5)	96.2(0.8)	96.2(0.8)

	MissForest	20	95.6(1.0)	97.6(1.0)	95.6(1.0)	95.6(1.0)
	EMreg	20	96.4(0.9)	98.0(0.5)	96.4(0.9)	96.4(0.9)
	EMreg-oos	20	96.6(0.8)	98.1(0.4)	96.6(0.8)	96.6(0.8)
	EMreg-KNN	20	**96.7(0.8)**	**98.2(0.4)**	**96.7(0.8)**	**96.7(0.8)**
	MICE	40	94.4(1.1)	96.9(0.6)	94.4(1.1)	94.3(1.1)
	SVDimpute	40	94.2(1.2)	96.8(0.7)	94.2(1.2)	94.2(1.2)
	MissForest	40	92.7(1.4)	96.0(0.8)	92.7(1.4)	92.8(1.4)
SVM	EMreg	40	94.0(1.3)	96.7(0.7)	94.0(1.3)	94.0(1.3)
	EMreg-oos	40	94.5(1.2)	97.0(0.7)	94.5(1.2)	94.5(1.2)
	EMreg-KNN	40	**94.8(1.0)***	**97.1(0.6)***	**94.8(1.0)**	**94.8(1.0)***
	MICE	60	92.6(1.1)	95.9(0.6)	92.6(1.1)	92.6(1.1)
	SVDimpute	60	92.3(1.2)	95.7(0.6)	92.3(1.2)	92.2(1.2)
	MissForest	60	90.6(1.5)	94.8(0.8)	90.6(1.5)	90.6(1.4)
	EMreg	60	92.3(1.0)	95.7(0.6)	92.3(1.0)	92.2(1.0)
	EMreg-oos	60	**93.0(0.8)***	**96.1(0.4)***	**93.0(0.8)**	**93.0(0.8)***
	EMreg-KNN	60	92.9(0.8)	96.1(0.5)	92.9(0.9)	92.9(0.9)

Para el caso del clasificador *K-NN* se observa una mejora estadísticamente significativa usando la propuesta *EMreg-KNN*, tanto con 40 % como 60 % de MV. Con 20 % de MV se observa una mejora con ambas propuestas, pero no son significativas.

Para el caso del clasificador *SVM*, la mejora significativa tambien es con 40 % y 60 % de MV. Con 20 % de MV existen mejoras pero no significativas.

Cuadro 4.8: Resultados de Handwritten digits basado en 30 ejecuciones distintas de training y testing set. Se muestra Accuracy (acc.), Area Under the Curve (AUC), Sensitivity (sens.) y F-measure (F). Resultados son expresados en media (desviación estándar).

Classifier	Imputation	MV rate (%)	Acc. (%)	AUC (%)	Sens. (%)	F (%)
	none	0	97.7(0.7)	98.7(0.4)	97.7(0.7)	97.7(0.7)
	MICE	20	95.5(1.1)	97.5(0.6)	95.5(1.1)	95.5(1.1)
	SVDimpute	20	95.4(1.1)	97.4(0.6)	95.4(1.1)	95.4(1.1)
	MissForest	20	94.4(1.0)	96.9(0.6)	94.4(1.0)	94.4(1.0)
	EMreg	20	95.4(0.9)	97.4(0.5)	95.4(0.9)	95.4(0.9)
	EMreg-oos	20	95.8(1.0)	97.7(0.6)	95.8(1.0)	95.8(1.0)

	EMreg-KNN	20	**96.0(0.9)***	**97.8(0.5)***	**96.0(0.9)**	**96.0(0.9)***
	MICE	40	93.2(1.4)	96.3(0.8)	93.2(1.4)	93.2(1.4)
	SVDimpute	40	92.9(1.3)	96.1(0.7)	92.9(1.3)	92.9(1.3)
	MissForest	40	91.0(1.2)	95.0(0.7)	91.0(1.2)	91.0(1.2)
RF	EMreg	40	93.0(1.2)	96.1(0.7)	93.0(1.2)	92.9(1.2)
	EMreg-oos	40	93.5(1.2)	96.4(0.6)	93.5(1.2)	93.5(1.2)
	EMreg-KNN	40	**93.7(1.0)***	**96.5(0.6)***	**93.7(1.0)**	**93.7(1.0)***
	MICE	60	91.2(1.2)	95.1(0.6)	91.2(1.2)	91.2(1.7)
	SVDimpute	60	91.4(1.3)	95.2(0.7)	91.4(1.3)	91.4(1.3)
	MissForest	60	88.2(1.6)	93.4(0.9)	88.2(1.6)	88.2(1.6)
	EMreg	60	90.9(1.4)	94.9(0.8)	90.9(1.4)	90.9(1.4)
	EMreg-oos	60	91.9(1.1)*	95.5(0.6)*	91.9(1.1)	91.9(1.1)*
	EMreg-KNN	60	**91.9(1.0)***	**95.5(0.6)***	**91.9(1.0)**	**91.9(1.0)***
	none	0	97.7(0.6)	98.7(0.3)	97.7(0.6)	97.7(0.6)
	MICE	20	95.6(0.9)	97.6(0.5)	95.6(0.9)	95.6(1.0)
	SVDimpute	20	95.6(1.0)	97.6(0.6)	95.6(1.0)	95.6(1.0)
	MissForest	20	94.8(1.0)	97.1(0.5)	94.8(1.0)	94.8(1.0)
	EMreg	20	95.6(0.9)	97.5(0.5)	95.6(0.9)	95.6(0.9)
	EMreg-oos	20	96.0(0.9)	97.8(0.5)	96.0(0.9)	96.0(0.9)*
	EMreg-KNN	20	**96.0(0.8)***	**97.8(0.5)***	**96.0(0.8)**	**96.0(0.9)***
	MICE	40	94.7(0.8)	97.0(0.5)	94.7(0.8)	94.7(0.8)
	SVDimpute	40	93.4(1.4)	96.3(0.8)	93.4(1.4)	93.4(1.4)
	MissForest	40	91.8(1.3)	95.4(0.7)	91.8(1.3)	91.8(1.3)
ANN	EMreg	40	93.5(1.3)	96.4(0.7)	93.5(1.3)	93.4(1.3)
	EMreg-oos	40	93.9(1.2)	96.6(0.7)	93.9(1.2)	93.4(1.2)
	EMreg-KNN	40	**94.1(1.1)**	**96.7(0.6)**	**94.1(1.1)**	**94.1(1.1)***
	MICE	60	92.0(1.0)	95.6(0.6)	92.0(1.0)	92.0(1.0)
	SVDimpute	60	91.6(1.1)	95.3(0.6)	91.6(1.1)	91.6(1.1)
	MissForest	60	89.3(1.6)	94.1(0.9)	89.3(1.6)	89.3(1.6)
	EMreg	60	91.1(1.1)	95.1(0.6)	91.1(1.1)	91.0(1.2)
	EMreg-oos	60	**92.6(1.0)***	**95.9(0.6)***	**92.6(1.0)**	**92.6(1.0)***
	EMreg-KNN	60	92.3(1.0)*	95.7(0.6)*	92.3(1.0)	92.2(1.0)*

Para el caso del clasificador *RF*, la propuesta *EMreg-KNN* obtienen mejoras significativas en todos los escenarios con MV. La propuesta *EMreg-oos* alcanza resultados similares.

Para el caso del clasificador *ANN*, las propuestas obtienen mejores resultados de manera significativa solo con 60 % de MV. Luego, si se analiza los resultados con 20 % de MV, solo *EMreg-KNN* obtiene mejoras significativas. Aunque se observa mejoras para el escenario de 40 % de MV, los resultados no son estadisticamente significativos.

Al observar la desviación estándar, vemos que en general las propuestas mejoran este aspecto, traduciendoce que los algoritmos son más estables. La excepción es para el caso del clasificador *K-NN*, donde las propuestas igualan o incluso aumentan la desviación estándar en algunos casos.

Podemos decir que a medida aumenta la presencia de MV, las métricas empeoran para cada caso, lo cual es esperable ya que el problema de clasificación aumenta su dificultad. Pero se debe resaltar que, las diferencias en los resultados entre el algoritmo original *EMreg* y las propuestas (*EMreg-oos* y *EMreg-KNN*) se hacen más notorias. Este buen resultado nos dice que entre más dificil sea el problema, mejor es el rendimiento relativo con respecto al algoritmo original.

4.3. Caltech

Caltech es una base de datos de imágenes destinada para la investigación de Visión por Computador y Máquinas de aprendizaje. Originalmente, esta consta de 9146 imágenes de 101 objetos distintos (pianos, motos, caras, etc). Esta versión es conocida como Caltech-101.
Aquí se redujo el dataset a solo 4 objetos: Caras (435), Leopardos (200), Motos (798) y Autos (123), obteniendo 1556 imágenes en total. Las variables estan organizadas por tipo de fuentes, las cuales son:

- 46 Coeficientes de Gabor.
- 40 Wavelets moments.
- 254 CENsus TRansform hISTogram (CENTRIST features)

Este set de datos no tienen, de manera natural, datos faltantes. Por este motivo, se procedio a crear patrones de MV en diferentes porcentajes: 20 %, 40 % y 60 %.
Aleatoriamente, se elijen los ejemplos que serán afectados por MV. Una vez que es elejido un ejemplo, aleatoriamente se elije el patrón de MV que se le aplicará. La aleatoriedad esta basada en una dis-

tribución uniforme. En resumen, se realizó el mismo procedimiento que con los datos Handwritten digits.

Las tablas 4.9 y 4.10 muestran los resultados de los escenarios anteriomente descritos. El caso *none* representa la clasificación con los datos sin MV, esperando que sea la cota superior en rendimiento.

Cuadro 4.9: Resultados de Caltech basado en 30 ejecuciones distintas de training y testing set. Se muestra Accuracy (acc.), Area Under the Curve (AUC), Sensitivity (sens.) y F-measure (F). Resultados son expresados en media (desviación estandar).

Classifier	Imputation	MV rate (%)	Acc. (%)	AUC (%)	Sens. (%)	F (%)
K-NN	*none*	0	97.7(0.9)	98.5(0.6)	97.7(0.9)	97.7(0.9)
	MICE	20	**96.1(1.4)***	97.2(1.0)	**96.1(1.4)***	**96.1(1.4)***
	SVDimpute	20	95.2(1.1)	96.7(0.8)	95.2(1.1)	95.2(1.1)
	MissForest	20	96.0(1.2)	97.2(0.9)	96.0(1.2)	96.0(1.2)
	EMreg	20	95.6(1.2)	97.1(0.8)	95.6(1.2)	95.6(1.2)
	EMreg-oos	20	95.7(1.2)	**97.2(0.8)**	95.7(1.2)	95.7(1.2)
	EMreg-KNN	20	95.6(1.3)	97.1(0.9)	95.6(1.3)	95.7(1.3)
	MICE	40	95.3(1.2)	96.5(1.0)	95.3(1.2)	95.3(1.2)
	SVDimpute	40	94.3(1.3)	95.9(1.1)	94.3(1.3)	94.3(1.4)
	MissForest	40	94.5(1.6)	96.0(1.1)	94.5(1.6)	94.5(1.6)
	EMreg	40	95.0(1.5)	96.5(1.2)	95.0(1.5)	95.0(1.5)
	EMreg-oos	40	95.2(1.4)	96.7(1.0)	95.2(1.4)	95.2(1.4)
	EMreg-KNN	40	**95.4(1.3)**	**96.8(0.9)**	**95.4(1.3)**	**95.4(1.3)**
	MICE	60	94.2(1.5)	95.5(1.2)	94.2(1.5)	94.2(1.5)
	SVDimpute	60	94.1(1.6)	95.8(1.2)	94.1(1.6)	94.1(1.6)
	MissForest	60	93.9(1.5)	95.5(1.1)	93.9(1.5)	93.9(1.5)
	EMreg	60	94.6(1.5)	96.3(1.1)	94.6(1.5)	94.6(1.5)
	EMreg-oos	60	95.2(1.4)	96.7(1.0)	95.2(1.4)	95.2(1.4)
	EMreg-KNN	60	**95.2(1.2)***	**96.7(0.8)**	**95.2(1.2)***	**95.2(1.2)***
	none	0	99.4(0.3)	99.6(0.2)	99.4(0.3)	99.4(0.3)
	MICE	20	98.5(0.7)	98.9(0.6)	98.5(0.7)	98.5(0.7)
	SVDimpute	20	98.4(0.9)	98.8(0.6)	98.4(0.9)	98.4(0.9)
	MissForest	20	97.9(0.9)	98.3(0.7)	97.9(0.9)	97.9(0.9)
	EMreg	20	98.8(0.7)	99.1(0.5)	98.8(0.7)	98.8(0.7)

	EMreg-oos	20	**98.9(0.6)**	**99.3(0.4)**	**98.9(0.6)**	**98.9(0.6)**
	EMreg-KNN	20	98.9(0.7)	99.2(0.5)	98.9(0.7)	98.9(0.7)
	MICE	40	98.2(0.8)	98.6(0.7)	98.2(0.8)	98.2(0.8)
	SVDimpute	40	98.0(0.8)	98.5(0.6)	98.0(0.8)	98.0(0.8)
	MissForest	40	97.1(1.1)	97.8(0.8)	97.1(1.1)	97.1(1.1)
SVM	EMreg	40	98.2(0.8)	98.7(0.6)	98.2(0.8)	98.2(0.8)
	EMreg-oos	40	98.6(0.7)	**99.0(0.5)**	98.6(0.7)	98.6(0.7)
	EMreg-KNN	40	**98.6(0.6)**	**99.0(0.5)**	**98.6(0.6)**	**98.6(0.6)**
	MICE	60	97.6(0.9)	98.2(0.7)	97.6(0.9)	97.6(0.9)
	SVDimpute	60	97.3(0.9)	98.0(0.7)	97.3(0.9)	97.3(0.9)
	MissForest	60	96.3(2.0)	97.2(1.4)	96.3(2.0)	96.3(1.9)
	EMreg	60	97.7(0.8)	98.4(0.6)	97.7(0.8)	97.7(0.8)
	EMreg-oos	60	**98.1(0.7)**	**98.6(0.6)**	**98.1(0.7)**	**98.1(0.7)**
	EMreg-KNN	60	98.0(0.8)	98.5(0.6)	98.0(0.8)	98.0(0.8)

Para el caso del clasificador *K-NN* se observa que el algoritmo *MICE* es el que entrega mejores resultados para un 20 % de MV. En cambio, para los casos de 40 % y 60 % de MV el mejor algoritmo es la propuesta *EMreg-KNN*.

Para el caso del clasificador *SVM*, ambas propuestas son las que entregan mejores resultados, destacando esta vez el algoritmo *EMreg-oos*. En algunos casos, el resultado al ser el mismo, se utilizó la desviación estándar para desempatar. No existen diferencias estadísticamente significativas, por lo que todos los algoritmos son muy competitivos en este caso.

Cuadro 4.10: Resultados de Caltech basado en 30 ejecuciones distintas de training y testing set. Se muestra Accuracy (acc.), Area Under the Curve (AUC), Sensitivity (sens.) y F-measure (F). Resultados son expresados en media (desviación estandar).

Classifier	Imputation	MV rate (%)	Acc. (%)	AUC (%)	Sens. (%)	F (%)
	none	0	99.1(0.4)	99.4(0.3)	99.1(0.4)	99.1(0.4)
	MICE	20	97.8(0.8)	98.3(0.6)	97.8(0.8)	97.8(0.8)
	SVDimpute	20	98.0(0.9)	98.5(0.7)	98.0(0.9)	98.0(0.9)
	MissForest	20	97.0(1.0)	97.5(0.9)	97.0(1.0)	97.0(1.0)
	EMreg	20	98.2(0.7)	98.7(0.6)	98.2(0.7)	98.2(0.7)
	EMreg-oos	20	**98.4(0.7)**	**98.8(0.5)**	**98.4(0.7)**	**98.4(0.7)**

	EMreg-KNN	20	98.3(0.8)	98.8(0.6)	98.3(0.8)	98.3(0.8)
	MICE	40	96.9(1.1)	97.6(0.9)	96.9(1.1)	96.9(1.1)
	SVDimpute	40	96.9(0.9)	97.7(0.8)	96.9(0.9)	96.9(0.9)
	MissForest	40	95.2(1.7)	96.0(1.4)	95.2(1.7)	95.2(1.7)
RF	EMreg	40	97.3(0.9)	98.0(0.7)	97.3(0.9)	97.3(0.9)
	EMreg-oos	40	**97.9(0.8)**	**98.5(0.6)**	**97.9(0.8)**	**97.9(0.8)**
	EMreg-KNN	40	97.7(0.7)	98.3(0.6)	97.7(0.7)	97.7(0.7)
	MICE	60	96.5(1.0)	97.2(0.8)	96.5(1.0)	96.4(1.0)
	SVDimpute	60	96.3(1.1)	97.1(0.9)	96.3(1.1)	96.3(1.1)
	MissForest	60	93.8(2.4)	94.9(1.8)	93.8(2.4)	93.8(2.3)
	EMreg	60	96.9(0.8)	97.6(0.6)	96.9(0.8)	96.9(0.8)
	EMreg-oos	60	**97.4(0.8)**	**98.1(0.6)**	**97.4(0.8)**	**97.4(0.8)**
	EMreg-KNN	60	97.2(0.9)	98.0(0.7)	97.2(0.9)	97.2(0.9)
	none	0	99.7(0.3)	99.8(0.2)	99.7(0.3)	99.7(0.3)
	MICE	20	98.5(0.7)	98.9(0.6)	98.5(0.7)	98.5(0.7)
	SVDimpute	20	98.1(1.2)	98.6(1.0)	98.1(1.2)	98.1(1.3)
	MissForest	20	97.8(0.9)	98.4(0.6)	97.8(0.9)	97.8(0.8)
	EMreg	20	98.5(0.8)	98.9(0.6)	98.5(0.8)	98.5(0.8)
	EMreg-oos	20	**98.7(0.7)**	**99.0(0.6)**	**98.7(0.7)**	**98.7(0.7)**
	EMreg-KNN	20	98.4(0.7)	98.8(0.6)	98.4(0.7)	98.4(0.7)
	MICE	40	98.1(0.9)	98.6(0.7)	98.1(0.9)	98.1(0.9)
	SVDimpute	40	97.6(0.9)	98.2(0.7)	97.6(0.9)	97.6(0.9)
	MissForest	40	96.0(1.7)	97.1(1.1)	96.0(1.7)	96.0(1.7)
ANN	EMreg	40	97.8(1.2)	98.3(1.1)	97.8(1.2)	97.8(1.3)
	EMreg-oos	40	98.2(1.0)	**98.7(0.7)**	98.2(1.0)	98.2(1.0)
	EMreg-KNN	40	**98.2(0.8)**	98.6(0.6)	**98.2(0.8)**	**98.2(0.8)**
	MICE	60	97.5(1.0)	98.1(0.8)	97.5(1.0)	97.5(0.9)
	SVDimpute	60	97.1(1.2)	97.9(0.9)	97.1(1.2)	97.1(1.2)
	MissForest	60	95.2(2.1)	96.6(1.4)	95.2(2.1)	95.3(2.0)
	EMreg	60	97.1(0.9)	97.8(0.7)	97.1(0.9)	97.1(0.9)
	EMreg-oos	60	**97.8(0.9)***	**98.3(0.6)**	**97.8(0.9)***	**97.8(0.9)***
	EMreg-KNN	60	97.5(0.9)	98.1(0.7)	97.5(0.9)	97.5(0.9)

Para el caso del clasificador *RF* ambas propuestas son superiores a los demas algoritmos y sobre esto, *EMreg-oos* es el claro vencedor en todas las métricas y todos los casos de MV. Eso si, en ningun caso la diferencia es estadísticamente significativa.

Para el caso del clasificador *ANN* se repite la tendencia del caso anterior. *EMreg-oos* obtiene los mejores resultados tanto para el caso de 20 % como 60 % de MV. Solo para el caso de 40 % de MV *EMreg-KNN* obtiene un mejor resultado, pero solo en el sentido de la desviación estándar. Solo algunos casos son estadísticamente significativos con respecto a *EMreg*.

Podemos decir que para el caso del set de datos Caltech el mejor algoritmo fue *EMreg-oos*. Al igual que los set de datos anteriores, el aumento del porcentaje de MV aumenta la dificultad del problema y eso se ve en la disminución en las métricas de clasificación. En este caso, los algoritmos son más competitivos entre sí y, dada la desviación estándar, en la mayoría de los casos tiene una estabilidad parecida.

Capítulo 5

Conclusiones y trabajo futuro

En esta tesis se propone dos mejoras del algoritmo de T. Schneider: *EMreg-oos* y *EMreg-KNN*. El primero ofrece la posibilidad de crear un modelo de imputación que permite imputar datos futuros. Una versión *Out of sample* del algoritmo original que aprende modelos de regresión que luego pueden ser utilizados directamente con datos con MV, es decir, en un ambiente de aprendizaje supervisado.
El segundo algoritmo implementa la idea de *vecinos recomendados*, la cual busca crear modelos de imputación con datos pertenecientes a la misma clase del dato a imputar. Además, el modelo de imputación creado en la fase de entrenamiento es construido en base a varios ensamblados de regresiones, dandole mayor importancia a las regresiones que fueron construidas con datos más parecidos al dato de prueba a imputar.
Se destaca la creación de vecindarios durante el entrenamiento, lo cual permite extender este algoritmo para su uso en tareas de agrupamiento (clustering) o de reducción de dimensionalidad (técnicas de Manifold learning).

Los experimentos con el set de datos ADNI mostraron que las técnicas de imputación permiten el uso de información adicional que de otro modo sería descartada. Los resultados claramente demostraron que construir modelos de clasificación con datos imputados es mejor, aunque esto conlleve la inserción de sesgo.
Los resultados con los tres set de datos demostraron que los algoritmos propuestos fueron superiores en practicamente todos los escenarios, incluso en algunas ocaciones, esta diferencia aumentaba cuando los datos tenian un mayor porcentaje de MV. Se probaron cuatro algoritmos de clasificación de naturaleza distinta precisamente para demostrar que los resultados fuesen independientes de estos.

Al comparar los algoritmos propuestos con el original (*EMreg*), se puede decir que *EMreg-oos* no tiene

limitaciones, pero si *EMreg-KNN*. Estas limitaciones se pueden resumir en: 1) La fase de entrenamiento es computacionalmente más costosa y 2) El hiperparámetro K, que define el número de vecinos en los vecindarios, debe ajustarse. Esto implica un mayor esfuerzo para lograr los mejores resultados. A pesar de esto, *EMreg-KNN*, en general, es el que obtiene los mejores resultados en los set de datos.

Considerando que *EMreg-KNN* es el mejor algoritmo, el trabajo futuro se centra en: 1) Estudiar y calcular la complejidad computacional y las propiedades de convergencia del algoritmo; 2) Incorporar mecanismos en el algoritmo para manejar datos heterogéneos, como utilizar funciones de distancia distintas de la distancia euclidiana para los cálculos de vecindarios. Todo esto pensando en poder usar otras fuentes de datos de ADNI y otros conjuntos de datos con características similares que tengan, por ejemplo, datos categóricos; 3) Profundizar los experimentos con set de datos sin MV, de tal manera de calcular métricas de imputación y contrastarlas con métricas de clasificación. Esto para determinar si una buena imputación implica una buena clasificación; 4) Estudiar la forma de calcular distancias de manera efectiva para el caso en que los vectores sean de gran dimensión, debido a que el concepto de distancia se degrada acorde al aumento de la dimensionalidad.

Esta tesis generó las siguientes publicaciones:

- Campos, S., Pizarro, L., Valle, C., Gray, K. R., Rueckert, D., and Allende, H. (2015). Evaluating imputation techniques for missing data in ADNI: A patient classification study. In Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, pages 3–10. Springer International Publishing. Campos et al. (2015)
- Campos, S., Veloz, A., and Allende, H. (2018). An out of sample version of the EM algorithm for imputing missing values in classification. volume 11401 of Lecture Notes in Computer Science, pages 194–202. Springer. Campos et al. (2018)
- Campos, S., Zamora, J., and Allende, H. (2024). Block-wise imputation EM algorithm in multi-source scenario: Adni case. Pattern Anal. Appl., 27(2).Campos et al. (2024)

Bibliografía

Abiri, N., Linse, B., Edén, P., and Ohlsson, M. (2019). Establishing strong imputation performance of a denoising autoencoder in a wide range of missing data problems. *Neurocomputing*, 365:137–146.

Acuna, E. and Rodriguez, C. (2004). The treatment of missing values and its effect in the classifier accuracy. *Classification, Clustering and Data Mining Applications, Springer, Berlin, Heidelberg*, pages 639–648.

Aghili, M., Tabarestani, S., and Adjouadi, M. (2022). Addressing the missing data challenge in multimodal datasets for the diagnosis of Alzheimer's disease. *Journal of Neuroscience Methods*, Volume 375.

Aracri, F., Giovanna Bianco, M., Quattrone, A., and Sarica, A. (2023). Imputation of missing clinical, cognitive and neuroimaging data of dementia using missforest, a random forest based algorithm. In *2023 IEEE 36th International Symposium on Computer-Based Medical Systems (CBMS)*, pages 684–688.

Azur, M., Stuart, E., Frangakis, C., and Leaf, P. (2011). Multiple imputation by chained equations: What is it and how does it work? *International Journal of Methods in Psychiatric Research*, 20(1):40–49.

Batista, G. E. A. P. A. and Monard, M. C. (2002). A study of k-nearest neighbour as an imputation method. In Abraham, A., del Solar, J. R., and Köppen, M., editors, *HIS*, volume 87 of *Frontiers in Artificial Intelligence and Applications*, pages 251–260. IOS Press.

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1st ed. edition.

Brouwer, R. K. and Pedrycz, W. (2003). Training a feed-forward network with incomplete data due to missing input variables. *Applied Soft Computing*, 3(1):23 – 36.

Cai, J., Candès, E. J., and Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982.

Campos, S., Pizarro, L., Valle, C., Gray, K. R., Rueckert, D., and Allende, H. (2015). Evaluating imputation techniques for missing data in adni: A patient classification study. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 3–10. Springer International Publishing.

Campos, S., Veloz, A., and Allende, H. (2018). An out of sample version of the EM algorithm for imputing missing values in classification. volume 11401 of *Lecture Notes in Computer Science*, pages 194–202. Springer.

Campos, S., Zamora, J., and Allende, H. (2024). Block-wise imputation EM algorithm in multi-source scenario: ADNI case. *Pattern Anal. Appl.*, 27(2).

Cheng, C.-Y., Tseng, W.-L., Chang, C.-F., Chang, C.-H., and Gau, S. S.-F. (2020). A deep learning approach for missing data imputation of rating scales assessing attention-deficit hyperactivity disorder. *Frontiers in Psychiatry*, 11:673.

Ciaccio, A. D. (2011). Bootstrap and nonparametric predictors to impute missing data. *Classification and Multivariate Analysis for Complex Data Structures Studies in Classification, Data Analysis, and Knowledge Organization*, pages 203–210.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.

Drucker, H., Burges, C. J. C., Kaufman, L., Smola, A. J., and Vapnik, V. (1996). Support vector regression machines. In Mozer, M., Jordan, M. I., and Petsche, T., editors, *NIPS*, pages 155–161. MIT Press.

Efron, B. (1994). Missing Data, Imputation, and the Bootstrap. *Journal of the American Statistical Association*, 89(426):463–475.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231.

Farhangfar, A., Kurgan, L., and Dy, J. (2008). Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition*, 41(12):3692 – 3705.

Fawcett, T. (2006). An introduction to {ROC} analysis. *Pattern Recognition Letters*, 27(8):861 – 874.

Fessant, F. and Midenet, S. (2002). Self-organising map for data imputation and correction in surveys. *Neural Computing and Applications*, 10(4):300–310.

García-Laencina, P. J., Sancho-Gómez, J.-L., and Figueiras-Vidal, A. R. (2010). Pattern classification with missing data: a review. *Neural Computing and Applications*, 19(2):263–282.

García-Laencina, P. J., Serrano, J., Figueiras-Vidal, A. R., and Sancho-Gómez, J.-L. (2007). Multi-task neural networks for dealing with missing inputs. In *Proceedings of the 2nd international work-conference on The Interplay Between Natural and Artificial Computation, Part I: Bio-inspired Modeling of Cognitive Tasks*, IWINAC '07, pages 282–291, Berlin, Heidelberg. Springer-Verlag.

Gisbrecht, A., Lueks, W., Mokbel, B., and Hammer, B. (2012). Out-of-sample kernel extensions for nonparametric dimensionality reduction. In *ESANN 2012*, pages 531–536.

Gondara, L. and Wang, K. (2018). MIDA: Multiple imputation using denoising autoencoders. In *Advances in Knowledge Discovery and Data Mining*, pages 260–272. Springer International Publishing.

Gray, K., Aljabar, P., Heckemann, R. A., Hammers, A., Rueckert, D., and ADNI, T. A. D. N. I. (2013). Random forest-based similarity measures for multi-modal classification of Alzheimer's disease. *NeuroImage*, 65:167–175.

Grzymala-Busse, J. W. and Hu, M. (2001). A comparison of several approaches to missing attribute values in data mining. In *Revised Papers from the Second International Conference on Rough Sets and Current Trends in Computing*, RSCTC '00, pages 378–385, London, UK, UK. Springer-Verlag.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

He, Y. (2006). *Missing Data Imputation for Tree-based Models*. Phd. Thesis. University of California, Los Angeles.

Hinrichs, C., Singh, V., Xu, G., and Johnson, S. (2009). Mkl for robust multi-modality ad classification. In *Proceedings of the 12th International Conference on Medical Image Computing and Computer-Assisted Intervention: Part II*, MICCAI '09, pages 786–794, Berlin, Heidelberg. Springer-Verlag.

Kohonen, T., Schroeder, M. R., and Huang, T. S., editors (2001). *Self-Organizing Maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition.

Little, R. J. A. and Rubin, D. B. (2002). *Statistical Analysis with Missing Data, Second Edition*. Wiley-Interscience, 2 edition.

Little, R. J. A. and Rubin, D. B. (2019). *Statistical Analysis with Missing Data, Third Edition*. Wiley-Interscience, 3 edition.

Liu, M., Zhang, J., and Yap, P.-T. (2017). View-aligned hypergraph learning for Alzheimer's disease diagnosis with incomplete multi-modality data. *Medical Image Analysis*, Volume 36.

Liu, P. and Lei, L. (2006). Missing data treatment methods and NBI model. In *ISDA (1)*, pages 633–638. IEEE Computer Society.

Luengo, J., García, S., and Herrera, F. (2012). On the choice of the best imputation methods for missing values considering three groups of classification methods. *Knowl. Inf. Syst.*, 32(1):77–108.

Macias, E., Boquet, G., Serrano, J., Vicario, J., Ibeas, J., and Morel, A. (2019). Novel imputing method and deep learning techniques for early prediction of sepsis in intensive care units. In *2019 Computing in Cardiology (CinC)*, pages 1–4.

McCombe, N., Liu, S., Ding, X., Prasad, G., Bucholc, M., Finn, D. P., Todd, S., McClean, P. L., and Wong-Lin, K. (2021). Practical strategies for extreme missing data imputation in dementia diagnosis. *IEEE journal of biomedical and health informatics*, 26(2):818–827.

McInnes, L., Healy, J., and Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Mcknight, P., Mcknight, K., Sidani, S., and Figueredo, A. (2007). *Missing Data: A Gentle Introduction*. Guilford Press, 1st edition.

Nakamura, E. F., Loureiro, A. A. F., and Frery, A. C. (2007). Information fusion for wireless sensor networks: Methods, models, and classifications. *ACM Comput. Surv.*, 39(3).

Nikfalazar, S., Yeh, C.-H., Bedingfield, S. E., and Khorshidi, H. A. (2019). Missing data imputation using decision trees and fuzzy clustering with iterative learning. *Knowledge and Information Systems*, 62:2419–2437.

Patil, B. M., Joshi, R. C., and Toshniwal, D. (2010). Missing value imputation based on k-mean clustering with weighted distance. In *IC3 (1)*, volume 94 of *Communications in Computer and Information Science*, pages 600–609. Springer.

Pereira, R. C., Santos, M., Rodrigues, P., and Henriques Abreu, P. (2020). Reviewing autoencoders for missing data imputation: Technical trends, applications and outcomes. *Journal of Artificial Intelligence Research*, 69:1255–1285.

Piela, P. (2002). Introduction to self-organizing maps modelling for imputation. In *Techniques and Technology. Research in Official Statistics*, pages 5–19.

Raghunathan, T. E., Lepkowski, J. M., Van Hoewyk, J., Solenberger, P., et al. (2001). A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey methodology*, 27(1):85–96.

Rahman, M. G. and Islam, M. Z. (2016). Missing value imputation using a fuzzy clustering-based EM approach. *Knowl. Inf. Syst.*, 46(2):389–422.

Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding.

Science, 290(5500):2323–2326.

Rubin, D. B. (1978). Multiple imputations in sample surveys - a phenomenological bayesian approach to nonresponse. In *Proceedings of the Survey Research Methods Section*, pages 20–28.

Saar-Tsechansky, M. and Provost, F. J. (2007). Handling missing values when applying classification models. *Journal of Machine Learning Research*, 8:1623–1657.

Saul, L. K., Roweis, S. T., and Singer, Y. (2003). Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155.

Schneider, T. (2001). Analysis of incomplete climate data: Estimation of mean values and covariance matrices and imputation of missing values. *Journal of Climate*, 14:853–871.

Sharpe, P. K. and Solly, R. J. (1995). Dealing with missing values in neural network-based diagnostic systems. *Neural Computing and Applications*, 3(2):73–77.

Stekhoven, D. J. and Bühlmann, P. (2012). MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118.

Stempfle, L., Panahi, A., and Johansson, F. D. (2023). Sharing pattern submodels for prediction with missing values. *Proceedings of the AAAI Conference on Artificial Intelligence*, Volume 37:9882–9890.

Tenenbaum, J. B., de Silva, V., and Langford, J. C. (2000). A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319–2323.

Thung, K.-H., Wee, C.-Y., Yap, P.-T., and Shen, D. (2014). Neurodegenerative disease diagnosis using incomplete multi-modality data via matrix shrinkage and completion. *NeuroImage*, 91:386–400.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. (2001). Missing value estimation methods for DNA microarrays. *Bioinformatics (Oxford, England)*, 17(6):520–525.

Van Der Maaten, L., Postma, E., and Van den Herik, J. (2009). Dimensionality reduction: a comparative review. *J Mach Learn Res*, 10:66–71.

Weber, C., Carrillo, M., Jagust, W., Jack, C., Shaw, L., Trojanowski, J., Saykin, A., Beckett, L., Sur, C., Rao, N., Mendez, P., Black, S., Li, K., Iwatsubo, T., Chang, C., Sosa, A., Rowe, C., Morris, J., and Weiner, M. (2021). The worldwide Alzheimer's disease neuroimaging initiative: ADNI-3 updates and global perspectives. *Alzheimer's & Dementia: Translational Research Clinical Interventions*, 7.

Weiner, M., Veitch, D., Aisen, P., Beckett, L., Cairns, N., Green, R., Harvey, D., Jack, C., Jagust, W., Liu, E., Morris, J., Petersen, R., Saykin, A., Schmidt, M., Shaw, L., Siuciak, J., Soares, H., Toga, A., and Trojanowski, J. (2012). The Alzheimer's disease neuroimaging initiative: a review of papers published

since its inception. *Alzheimers Dement*, 8(1 Suppl):S1–68.

Weiner, M. W. e. a. (2015). Impact of the Alzheimer's disease neuroimaging initiative, 2004 to 2014. *Alzheimer's & Dementia: The Journal of the Alzheimer's Association*, 11(7):865–884.

Yuan, L., Wang, Y., Thompson, P. M., Narayan, V. A., and Ye, J. (2012). Multi-source feature learning for joint analysis of incomplete multiple heterogeneous neuroimaging data. *NeuroImage*, 61(3):622–632.

Zhang, D. and Shen, D. (2011). Multicost: Multi-stage cost-sensitive classification of Alzheimer's disease. In Suzuki, K., Wang, F., Shen, D., and Yan, P., editors, *MLMI*, volume 7009 of *Lecture Notes in Computer Science*, pages 344–351. Springer.

Zhang, D., Wang, Y., Zhou, L., Yuan, H., and Shen, D. (2011). Multimodal classification of Alzheimer's disease and mild cognitive impairment. *NeuroImage*, 55(3):856–867.

Zhang, L., Zhao, Y., Zhu, Z., Shen, D., and Ji, S. (2018). Multi-view missing data completion. *IEEE Transactions on Knowledge and Data Engineering*, 30(7):1296–1309.

Zhang, S., Zhang, J., Zhu, X., Qin, Y., and Zhang, C. (2008). *Missing Value Imputation Based on Data Clustering*, pages 128–138. Springer-Verlag, Berlin, Heidelberg.

Zhou, T., Liu, M., Thung, K.-H., and Shen, D. (2019). Latent representation learning for Alzheimer's disease diagnosis with incomplete multi-modality neuroimaging and genetic data. *IEEE Transactions on Medical Imaging*, Volume 38:2411–2422.