



ugr

Universidad
de Granada

TRABAJO FIN DE GRADO
INGENIERÍA INFORMÁTICA

pyMIL-BNF

Multiple Instance Learning Bag Noise Filters library in
Python

Autor

Juan Carlos Orte Cardona

Director

Julián Luengo Martín



FACULTAD DE ECONOMÍA, EDUCACIÓN Y TECNOLOGÍA DE CEUTA

—
Ceuta, 1 de julio de 2019

pyMIL-BNF

Multiple Instance Learning Bag Noise Filters library in
Python



<https://github.com/jcarlosorte/TFG>

Autor

Juan Carlos Orte Cardona

Director

Julián Luengo Martín

Título del Proyecto: Multiple Instance Learning Bag Noise Filters library in Python

Juan Carlos Orte Cardona

Palabras clave: Multiple Instance Learning, MIL, Machine Learning, Data Science, Data Mining, Noisy Filter, attribute noise, class noise.

Resumen

El auge de las técnicas de visión artificial en nuestros días está potenciando las técnicas de Ciencia de Datos relacionadas con la clasificación de objetos. Uno de los paradigmas más influyentes en la clasificación de objetos es el Multiple Instance Learning (MIL), donde se dispone de diferentes representaciones para un mismo objeto.

Sin embargo, la clasificación de objetos en MIL presenta desafíos hoy día, puesto que la identificación de objetos no es exacta y pueden recogerse representaciones erróneas (ruidosas) o incluso etiquetas incorrectas. En sistemas críticos como conducción autónoma, reconocimientos biométricos y otros, estos errores pueden ser catastróficos.

En este TFG se plantea el desarrollo de una biblioteca de técnicas de limpieza de ruido para MIL escrita en Python. No existen propuestas de esta naturaleza aún en la literatura especializada, por lo que este TFG incluye una vertiente de investigación.

Project Title: Multiple Instance Learning Bag Noise Filters library in Python

Juan Carlos Orte Cardona

Keywords: Multiple Instance Learning, MIL, Machine Learning, Data Science, Data Mining, Noisy Filter, attribute noise, class noise.

Abstract

The rise of artificial vision techniques in our days is enhancing the techniques of Data Science related to the classification of objects. One of the most influential paradigms in the objects classification is the Multiple Instance Learning (MIL), where different representations are available for the same object.

However, the classification of MIL objects presents challenges today, since the identification of objects is not exact and erroneous representations (noisy) or even incorrect labels can be collected. In critical systems such as autonomous driving, biometric surveys and others, these errors can be catastrophic.

In this project (TFG) the development of a library of noise cleaning techniques for MIL written in Python is proposed. There are no proposals of this nature even in the specialized literature, so this project (TFG) includes a research aspect.

Yo, Juan Carlos Orte Cardona, alumno de la titulación de Ingeniería Informática de la Facultad de Economía, Educación y Tecnología de Ceuta, con DNI 75135406-H, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Juan Carlos Orte Cardona

Ceuta a 1 de julio de 2019.

D. **Julián Luengo Martín**, Profesor del Área de Ingeniería Informática del Departamento de de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado *pyMIL-BNF, Multiple Instance Learning Bag Noise Filters library in Python*, ha sido realizado bajo su supervisión por **Juan Carlos Orte Cardona**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Ceuta a 1 de julio de 2019.

El director:

Julián Luengo Martín

Agradecimientos

Al igual que un algoritmo de aprendizaje, las personas somos un cúmulo de experiencias que nos entrenan a medida que crecemos, unas aportan ruido y pueden ser filtradas y otras hacen que mejores en tu evaluación. Si has sido alguna vez una instancia que ayudó al entrenamiento de mi algoritmo, sea cual sea tu etiqueta, GRACIAS!.

Índice general

1. Introducción	1
1.1. Introducción	2
1.2. Definición del Problema	2
1.3. Objetivos	2
1.4. Motivación	3
1.5. Organización del documento	4
2. Planificación	5
2.1. Gestión Temporal del Proyecto	6
2.1.1. Estructura de descomposición del trabajo	7
2.1.2. Comparativa	8
2.2. Gestión de comunicaciones	9
2.3. Gestión de costes	9
2.4. Gestión de costes	10
3. Antecedentes	11
3.1. Ciencia de Datos	12
3.2. Minería de Datos	13
3.2.1. Machine Learning	14
3.2.2. Validación	14
3.2.3. Medidas de evaluación	15
3.3. Aprendizaje Supervisado: Multiple-Instace Learning	17
3.4. Pre-procesamiento de Datos	18
3.4.1. Tratamiento de Ruido	19
3.4.2. Tipos de Ruido	20
3.4.3. Filtrado de Ruido en Clases	20
4. Marco Tecnológico	22
4.1. Herramientas para la gestión del proyecto	23
4.1.1. GitHub	23
4.1.2. GitHub Desktop	23
4.1.3. Hercules	23
4.1.4. PuTTY	24

4.1.5.	FileZilla	24
4.2.	Herramientas para modelado del proyecto	25
4.2.1.	yEd graph editor	25
4.3.	Herramientas para desarrollo del proyecto	25
4.3.1.	Anaconda Navigator	25
4.3.2.	Python	25
4.3.2.1.	Justificación	25
4.3.2.2.	Bibliotecas	27
4.4.	Herramientas Externas para desarrollo del proyecto	29
4.4.1.	Multiple-Instance Learning Python Toolbox	29
4.4.1.1.	Justificación	29
4.4.1.2.	Especificaciones	29
4.4.1.3.	Clasificadores	30
4.4.2.	Dataset	31
4.4.2.1.	Musk	31
4.4.2.2.	Mutagenesis	32
4.4.2.3.	Tiger, Fox y Elephant	33
4.4.2.4.	Corel African	35
4.4.2.5.	Birds	36
4.4.2.6.	Gaussian	37
4.5.	Herramientas para elaboración de la documentación	37
4.5.1.	TeXstudio	37
4.5.2.	Apache OpenOffice Calc	37
5.	pyMIL-BNF: Filtros de Ruido de Clase para MIL	38
5.1.	Análisis del problema: Limpieza de Datos en MIL	39
5.1.1.	Arquitectura de flujo de datos	40
5.1.2.	Análisis de requisitos	41
5.1.3.	Análisis de soluciones	42
5.1.4.	Solución propuesta	42
5.2.	Diseño General de pyMIL-BNF	44
5.2.1.	Grupos funcionales	45
5.2.2.	Requisitos funcionales	45
5.2.3.	Roles	46
5.2.4.	Modelo de Casos de Uso	46
5.2.5.	Dependencia de los Casos de Uso	48
5.2.6.	Flujo general de los filtros	48
5.2.7.	Descripciones	50
5.2.7.1.	MIL-EF	50
5.2.7.2.	MIL-CVCF	50
5.2.7.3.	MIL-IPF	50
5.2.7.4.	Ruido Artificial	50
5.2.8.	Diagrama de flujo	51
5.2.8.1.	MIL-EF	51

5.2.8.2.	MIL-CVCF	52
5.2.8.3.	MIL-IPF	53
5.3.	Implementación	54
6.	Estudio Experimental	55
6.1.	Marco Experimental	56
6.1.1.	Clasificadores MILpy	56
6.1.1.1.	Parámetros	56
6.1.1.2.	Medida de Evaluación	57
6.1.2.	Filtros pyMIL-BNF	57
6.1.2.1.	Parámetros	57
6.1.3.	Dataset	58
6.2.	Resultados Experimentales	58
6.2.1.	Tablas Resumen y análisis	59
6.2.1.1.	Clasificador SimpleMIL-max	59
6.2.1.2.	Clasificador SimpleMIL-min	60
6.2.1.3.	Clasificador SimpleMIL-extreme	61
6.2.1.4.	Clasificador BOW	62
6.2.1.5.	Clasificador CKNN	63
6.2.1.6.	Clasificador maxDD	64
6.2.1.7.	Clasificador EMDD	65
6.2.1.8.	Clasificador MILBoost	66
7.	Conclusiones	67
7.1.	Introducción	68
7.2.	Análisis del Estudio	68
7.3.	Consecución de los objetivos	70
7.4.	Trabajos Futuros	70
7.5.	Lecciones Aprendidas y conclusiones personales	71
Bibliografía		73
Referencias	73
Anexos		77
A. Manual de Usuario		78
A.1.	Librería pyMIL-BNF	80
A.2.	Interfaz gráfica pyMIL-BNF	82
B. Código Fuente de pyMIL-BNF		85
B.1.	Interfaz gráfica pyMIL-BNF	86
B.2.	Módulo MIL-EF	94
B.3.	Módulo MIL-CVCF	104
B.4.	Módulo MIL-IPF	112

C. Resultados	120
C.1. Musk 1	121
C.2. Mutagenesis 2	135
C.3. Tiger	149
C.4. Fox	163
C.5. Elephant	177
 D. Tablas Detalladas	 192
D.1. Ruido 0 %	193
D.2. Ruido 5 %	195
D.3. Ruido 10 %	197
D.4. Ruido 15 %	200
D.5. Ruido 20 %	202
D.6. Ruido 25 %	204
D.7. Ruido 30 %	207

Índice de Figuras

1.1. Paradigma Gargabe in-Garbage out	3
2.1. Planificación Original	6
2.2. Planificación Real	6
2.3. Estructura de descomposición del trabajo	7
2.4. Tiempo de ejecución en Hercules	8
3.1. KDD	12
3.2. Pipe and Filter	13
3.3. Preparación de Datos	13
3.4. Pasos en Machine Learning	14
3.5. k-fold	15
3.6. Curva ROC	16
3.7. AUC	17
3.8. MIL	18
3.9. Pre-procesamiento de Datos	18
3.10. Tipos de Ruido	19
4.1. Código QR GitHub	23
4.2. Colas en Hercules	24
4.3. Lenguajes de programación	26
4.4. Encuesta KDnuggets	26
4.5. Bibliotecas Python	28
4.6. Gráfico de dispersión de Musk 1	32
4.7. Gráfico de dispersión de Musk 2	32
4.8. Gráfico de dispersión de Mutagenesis Easy	33
4.9. Gráfico de dispersión de Mutagenesis Hard	33
4.10. Gráfico de dispersión de Tiger	34
4.11. Gráfico de dispersión de Fox	34
4.12. Gráfico de dispersión de Elephant	35
4.13. Gráfico de dispersión de Corel African	35
4.14. Gráfico de dispersión de Bids BrCr	36
4.15. Gráfico de dispersión de Bids WiWr	36
4.16. Gráfico de dispersión de Gaussian-MI	37

5.1. Bolsa de Datos	39
5.2. Pipe and Filter de pyMIL-BNF	40
5.3. Esquema Filtro MIL	41
5.4. Esquema Clasificador MIL	43
5.5. Filtro MIL	44
5.6. Esquema Grupos Funcionales	45
5.7. Diagrama de Casos de Uso	47
5.8. Diagrama flujo de procesos	49
5.9. Diagrama flujo MIL-EF	51
5.10. Diagrama flujo MIL-CVCF	52
5.11. Diagrama flujo MIL-IPF	53
5.12. Interfaz Gráfica para pyMIL-BNF	54
A.1. Ejemplo consola Python	82
A.2. Inicio Interfaz gráfica pyMIL-BNF	82
A.3. Interfaz parámetros Filtros	83
A.4. Interfaz con predicción	84

Índice de Tablas

2.1. Comparación de Timelines	9
2.2. Marco de costos del proyecto	9
2.3. Gestión de Riesgos de pyMIL-BNF	10
4.1. Clasificadores MIL	30
4.2. Dataset MILpy	31
5.1. Esquemas de Filtrado	39
5.2. Requisitos Funcionales	45
5.3. Roles en Casos de Uso	46
5.4. Dependencia de los Casos de Uso	48
6.1. Parámetros Clasificadores MIL	56
6.2. Parámetros Filtros pyMIL-BNF	57
6.3. Clasificadores usados por cada filtro	58
6.4. Dataset usados en estudio	58
6.5. Tabla Resumen Clasificador SimpleMIL-max	59
6.6. Tabla Resumen Clasificador SimpleMIL-min	60
6.7. Tabla Resumen Clasificador SimpleMIL-extreme	61
6.8. Tabla Resumen Clasificador BOW	62
6.9. Tabla Resumen Clasificador CKNN	63
6.10. Tabla Resumen Clasificador maxDD	64
6.11. Tabla Resumen Clasificador EMDD	65
6.12. Tabla Resumen Clasificador MILBoost	66
7.1. Tabla Resumen Completa	68
7.2. Consecución de Objetivos	70
C.1. Precisión MIL-EF Musk 1, Consenso, Ruido 0 %	121
C.2. Precisión MIL-EF Musk 1, Max Votos, Ruido 0 %	122
C.3. Precisión MIL-EF Musk 1, Consenso, Ruido 5 %	122
C.4. Precisión MIL-EF Musk 1, Max Votos, Ruido 5 %	122
C.5. Precisión MIL-EF Musk 1, Consenso, Ruido 10 %	123
C.6. Precisión MIL-EF Musk 1, Max Votos, Ruido 10 %	123

C.7. Precisión MIL-EF Musk 1, Consenso, Ruido 15 %	123
C.8. Precisión MIL-EF Musk 1, Max Votos, Ruido 15 %	124
C.9. Precisión MIL-EF Musk 1, Consenso, Ruido 20 %	124
C.10. Precisión MIL-EF Musk 1, Max Votos, Ruido 20 %	124
C.11. Precisión MIL-EF Musk 1, Consenso, Ruido 25 %	125
C.12. Precisión MIL-EF Musk 1, Max Votos, Ruido 25 %	125
C.13. Precisión MIL-EF Musk 1, Consenso, Ruido 30 %	125
C.14. Precisión MIL-EF Musk 1, Max Votos, Ruido 30 %	126
C.15. Precisión MIL-CVCF Musk 1, Consenso, Ruido 0 %	126
C.16. Precisión MIL-CVCF Musk 1, Max Votos, Ruido 0 %	126
C.17. Precisión MIL-CVCF Musk 1, Consenso, Ruido 5 %	127
C.18. Precisión MIL-CVCF Musk 1, Max Votos, Ruido 5 %	127
C.19. Precisión MIL-CVCF Musk 1, Consenso, Ruido 10 %	127
C.20. Precisión MIL-CVCF Musk 1, Max Votos, Ruido 10 %	128
C.21. Precisión MIL-CVCF Musk 1, Consenso, Ruido 15 %	128
C.22. Precisión MIL-CVCF Musk 1, Max Votos, Ruido 15 %	128
C.23. Precisión MIL-CVCF Musk 1, Consenso, Ruido 20 %	129
C.24. Precisión MIL-CVCF Musk 1, Max Votos, Ruido 20 %	129
C.25. Precisión MIL-CVCF Musk 1, Consenso, Ruido 25 %	129
C.26. Precisión MIL-CVCF Musk 1, Max Votos, Ruido 25 %	130
C.27. Precisión MIL-CVCF Musk 1, Consenso, Ruido 30 %	130
C.28. Precisión MIL-CVCF Musk 1, Max Votos, Ruido 30 %	130
C.29. Precisión MIL-IPF Musk 1, Consenso, Ruido 0 %	131
C.30. Precisión MIL-IPF Musk 1, Max Votos, Ruido 0 %	131
C.31. Precisión MIL-IPF Musk 1, Consenso, Ruido 5 %	131
C.32. Precisión MIL-IPF Musk 1, Max Votos, Ruido 5 %	132
C.33. Precisión MIL-IPF Musk 1, Consenso, Ruido 10 %	132
C.34. Precisión MIL-IPF Musk 1, Max Votos, Ruido 10 %	132
C.35. Precisión MIL-IPF Musk 1, Consenso, Ruido 15 %	133
C.36. Precisión MIL-IPF Musk 1, Max Votos, Ruido 15 %	133
C.37. Precisión MIL-IPF Musk 1, Consenso, Ruido 20 %	133
C.38. Precisión MIL-IPF Musk 1, Max Votos, Ruido 20 %	134
C.39. Precisión MIL-IPF Musk 1, Consenso, Ruido 25 %	134
C.40. Precisión MIL-IPF Musk 1, Max Votos, Ruido 25 %	134
C.41. Precisión MIL-IPF Musk 1, Consenso, Ruido 30 %	135
C.42. Precisión MIL-IPF Musk 1, Max Votos, Ruido 30 %	135
C.43. Precisión MIL-EF Muta 2, Consenso, Ruido 0 %	135
C.44. Precisión MIL-EF Muta 2, Max Votos, Ruido 0 %	136
C.45. Precisión MIL-EF Muta 2, Consenso, Ruido 5 %	136
C.46. Precisión MIL-EF Muta 2, Max Votos, Ruido 5 %	136
C.47. Precisión MIL-EF Muta 2, Consenso, Ruido 10 %	137
C.48. Precisión MIL-EF Muta 2, Max Votos, Ruido 10 %	137
C.49. Precisión MIL-EF Muta 2, Consenso, Ruido 15 %	137
C.50. Precisión MIL-EF Muta 2, Max Votos, Ruido 15 %	138

C.51.Precisión MIL-EF Muta 2, Consenso, Ruido 20 %	138
C.52.Precisión MIL-EF Muta 2, Max Votos, Ruido 20 %	138
C.53.Precisión MIL-EF Muta 2, Consenso, Ruido 25 %	139
C.54.Precisión MIL-EF Muta 2, Max Votos, Ruido 25 %	139
C.55.Precisión MIL-EF Muta 2, Consenso, Ruido 30 %	139
C.56.Precisión MIL-EF Muta 2, Max Votos, Ruido 30 %	140
C.57.Precisión MIL-CVCF Muta 2, Consenso, Ruido 0 %	140
C.58.Precisión MIL-CVCF Muta 2, Max Votos, Ruido 0 %	140
C.59.Precisión MIL-CVCF Muta 2, Consenso, Ruido 5 %	141
C.60.Precisión MIL-CVCF Muta 2, Max Votos, Ruido 5 %	141
C.61.Precisión MIL-CVCF Muta 2, Consenso, Ruido 10 %	141
C.62.Precisión MIL-CVCF Muta 2, Max Votos, Ruido 10 %	141
C.63.Precisión MIL-CVCF Muta 2, Consenso, Ruido 15 %	142
C.64.Precisión MIL-CVCF Muta 2, Max Votos, Ruido 15 %	142
C.65.Precisión MIL-CVCF Muta 2, Consenso, Ruido 20 %	142
C.66.Precisión MIL-CVCF Muta 2, Max Votos, Ruido 20 %	143
C.67.Precisión MIL-CVCF Muta 2, Consenso, Ruido 25 %	143
C.68.Precisión MIL-CVCF Muta 2, Max Votos, Ruido 25 %	143
C.69.Precisión MIL-CVCF Muta 2, Consenso, Ruido 30 %	144
C.70.Precisión MIL-CVCF Muta 2, Max Votos, Ruido 30 %	144
C.71.Precisión MIL-IPF Muta 2, Consenso, Ruido 0 %	144
C.72.Precisión MIL-IPF Muta 2, Max Votos, Ruido 0 %	145
C.73.Precisión MIL-IPF Muta 2, Consenso, Ruido 5 %	145
C.74.Precisión MIL-IPF Muta 2, Max Votos, Ruido 5 %	145
C.75.Precisión MIL-IPF Muta 2, Consenso, Ruido 10 %	146
C.76.Precisión MIL-IPF Muta 2, Max Votos, Ruido 10 %	146
C.77.Precisión MIL-IPF Muta 2, Consenso, Ruido 15 %	146
C.78.Precisión MIL-IPF Muta 2, Max Votos, Ruido 15 %	147
C.79.Precisión MIL-IPF Muta 2, Consenso, Ruido 20 %	147
C.80.Precisión MIL-IPF Muta 2, Max Votos, Ruido 20 %	147
C.81.Precisión MIL-IPF Muta 2, Consenso, Ruido 25 %	148
C.82.Precisión MIL-IPF Muta 2, Max Votos, Ruido 25 %	148
C.83.Precisión MIL-IPF Muta 2, Consenso, Ruido 30 %	148
C.84.Precisión MIL-IPF Muta 2, Max Votos, Ruido 30 %	149
C.85.Precisión MIL-EF Tiger, Consenso, Ruido 0 %	149
C.86.Precisión MIL-EF Tiger, Max Votos, Ruido 0 %	149
C.87.Precisión MIL-EF Tiger, Consenso, Ruido 5 %	150
C.88.Precisión MIL-EF Tiger, Max Votos, Ruido 5 %	150
C.89.Precisión MIL-EF Tiger, Consenso, Ruido 10 %	150
C.90.Precisión MIL-EF Tiger, Max Votos, Ruido 10 %	151
C.91.Precisión MIL-EF Tiger, Consenso, Ruido 15 %	151
C.92.Precisión MIL-EF Tiger, Max Votos, Ruido 15 %	151
C.93.Precisión MIL-EF Tiger, Consenso, Ruido 20 %	152
C.94.Precisión MIL-EF Tiger, Max Votos, Ruido 20 %	152

C.95.Precisión MIL-EF Tiger, Consenso, Ruido 25 %	152
C.96.Precisión MIL-EF Tiger, Max Votos, Ruido 25 %	153
C.97.Precisión MIL-EF Tiger, Consenso, Ruido 30 %	153
C.98.Precisión MIL-EF Tiger, Max Votos, Ruido 30 %	153
C.99.Precisión MIL-CVCF Tiger, Consenso, Ruido 0 %	154
C.100.Precisión MIL-CVCF Tiger, Max Votos, Ruido 0 %	154
C.101.Precisión MIL-CVCF Tiger, Consenso, Ruido 5 %	154
C.102.Precisión MIL-CVCF Tiger, Max Votos, Ruido 5 %	155
C.103.Precisión MIL-CVCF Tiger, Consenso, Ruido 10 %	155
C.104.Precisión MIL-CVCF Tiger, Max Votos, Ruido 10 %	155
C.105.Precisión MIL-CVCF Tiger, Consenso, Ruido 15 %	156
C.106.Precisión MIL-CVCF Tiger, Max Votos, Ruido 15 %	156
C.107.Precisión MIL-CVCF Tiger, Consenso, Ruido 20 %	156
C.108.Precisión MIL-CVCF Tiger, Max Votos, Ruido 20 %	157
C.109.Precisión MIL-CVCF Tiger, Consenso, Ruido 25 %	157
C.110.Precisión MIL-CVCF Tiger, Max Votos, Ruido 25 %	157
C.111.Precisión MIL-CVCF Tiger, Consenso, Ruido 30 %	158
C.112.Precisión MIL-CVCF Tiger, Max Votos, Ruido 30 %	158
C.113.Precisión MIL-IPF Tiger, Consenso, Ruido 0 %	158
C.114.Precisión MIL-IPF Tiger, Max Votos, Ruido 0 %	159
C.115.Precisión MIL-IPF Tiger, Consenso, Ruido 5 %	159
C.116.Precisión MIL-IPF Tiger, Max Votos, Ruido 5 %	159
C.117.Precisión MIL-IPF Tiger, Consenso, Ruido 10 %	160
C.118.Precisión MIL-IPF Tiger, Max Votos, Ruido 10 %	160
C.119.Precisión MIL-IPF Tiger, Consenso, Ruido 15 %	160
C.120.Precisión MIL-IPF Tiger, Max Votos, Ruido 15 %	161
C.121.Precisión MIL-IPF Tiger, Consenso, Ruido 20 %	161
C.122.Precisión MIL-IPF Tiger, Max Votos, Ruido 20 %	161
C.123.Precisión MIL-IPF Tiger, Consenso, Ruido 25 %	162
C.124.Precisión MIL-IPF Tiger, Max Votos, Ruido 25 %	162
C.125.Precisión MIL-IPF Tiger, Consenso, Ruido 30 %	162
C.126.Precisión MIL-IPF Tiger, Max Votos, Ruido 30 %	163
C.127.Precisión MIL-EF Fox, Consenso, Ruido 0 %	163
C.128.Precisión MIL-EF Fox, Max Votos, Ruido 0 %	163
C.129.Precisión MIL-EF Fox, Consenso, Ruido 5 %	164
C.130.Precisión MIL-EF Fox, Max Votos, Ruido 5 %	164
C.131.Precisión MIL-EF Fox, Consenso, Ruido 10 %	164
C.132.Precisión MIL-EF Fox, Max Votos, Ruido 10 %	165
C.133.Precisión MIL-EF Fox, Consenso, Ruido 15 %	165
C.134.Precisión MIL-EF Fox, Max Votos, Ruido 15 %	165
C.135.Precisión MIL-EF Fox, Consenso, Ruido 20 %	166
C.136.Precisión MIL-EF Fox, Max Votos, Ruido 20 %	166
C.137.Precisión MIL-EF Fox, Consenso, Ruido 25 %	166
C.138.Precisión MIL-EF Fox, Max Votos, Ruido 25 %	167

C.13	Precisión MIL-EF Fox, Consenso, Ruido 30 %	167
C.14	Precisión MIL-EF Fox, Max Votos, Ruido 30 %	167
C.141	Precisión MIL-CVCF Fox, Consenso, Ruido 0 %	168
C.142	Precisión MIL-CVCF Fox, Max Votos, Ruido 0 %	168
C.143	Precisión MIL-CVCF Fox, Consenso, Ruido 5 %	168
C.144	Precisión MIL-CVCF Fox, Max Votos, Ruido 5 %	169
C.145	Precisión MIL-CVCF Fox, Consenso, Ruido 10 %	169
C.146	Precisión MIL-CVCF Fox, Max Votos, Ruido 10 %	169
C.147	Precisión MIL-CVCF Fox, Consenso, Ruido 15 %	170
C.148	Precisión MIL-CVCF Fox, Max Votos, Ruido 15 %	170
C.149	Precisión MIL-CVCF Fox, Consenso, Ruido 20 %	170
C.15	Precisión MIL-CVCF Fox, Max Votos, Ruido 20 %	171
C.151	Precisión MIL-CVCF Fox, Consenso, Ruido 25 %	171
C.152	Precisión MIL-CVCF Fox, Max Votos, Ruido 25 %	171
C.153	Precisión MIL-CVCF Fox, Consenso, Ruido 30 %	172
C.154	Precisión MIL-CVCF Fox, Max Votos, Ruido 30 %	172
C.155	Precisión MIL-IPF Fox, Consenso, Ruido 0 %	172
C.156	Precisión MIL-IPF Fox, Max Votos, Ruido 0 %	173
C.157	Precisión MIL-IPF Fox, Consenso, Ruido 5 %	173
C.158	Precisión MIL-IPF Fox, Max Votos, Ruido 5 %	173
C.159	Precisión MIL-IPF Fox, Consenso, Ruido 10 %	174
C.16	Precisión MIL-IPF Fox, Max Votos, Ruido 10 %	174
C.161	Precisión MIL-IPF Fox, Consenso, Ruido 15 %	174
C.162	Precisión MIL-IPF Fox, Max Votos, Ruido 15 %	175
C.163	Precisión MIL-IPF Fox, Consenso, Ruido 20 %	175
C.164	Precisión MIL-IPF Fox, Max Votos, Ruido 20 %	175
C.165	Precisión MIL-IPF Fox, Consenso, Ruido 25 %	176
C.166	Precisión MIL-IPF Fox, Max Votos, Ruido 25 %	176
C.167	Precisión MIL-IPF Fox, Consenso, Ruido 30 %	176
C.168	Precisión MIL-IPF Fox, Max Votos, Ruido 30 %	177
C.169	Precisión MIL-EF Elephant, Consenso, Ruido 0 %	177
C.17	Precisión MIL-EF Max Votos, Consenso, Ruido 0 %	177
C.171	Precisión MIL-EF Elephant, Consenso, Ruido 5 %	178
C.172	Precisión MIL-EF Max Votos, Consenso, Ruido 5 %	178
C.173	Precisión MIL-EF Elephant, Consenso, Ruido 10 %	178
C.174	Precisión MIL-EF Max Votos, Consenso, Ruido 10 %	179
C.175	Precisión MIL-EF Elephant, Consenso, Ruido 15 %	179
C.176	Precisión MIL-EF Max Votos, Consenso, Ruido 15 %	179
C.177	Precisión MIL-EF Elephant, Consenso, Ruido 20 %	180
C.178	Precisión MIL-EF Max Votos, Consenso, Ruido 20 %	180
C.179	Precisión MIL-EF Elephant, Consenso, Ruido 25 %	180
C.18	Precisión MIL-EF Max Votos, Consenso, Ruido 25 %	181
C.181	Precisión MIL-EF Elephant, Consenso, Ruido 30 %	181
C.182	Precisión MIL-EF Max Votos, Consenso, Ruido 30 %	181

C.183	Precisión MIL-CVCF Elephant, Consenso, Ruido 0 %	182
C.184	Precisión MIL-CVCF Elephant, Max Votos, Ruido 0 %	182
C.185	Precisión MIL-CVCF Elephant, Consenso, Ruido 5 %	182
C.186	Precisión MIL-CVCF Elephant, Max Votos, Ruido 5 %	183
C.187	Precisión MIL-CVCF Elephant, Consenso, Ruido 10 %	183
C.188	Precisión MIL-CVCF Elephant, Max Votos, Ruido 10 %	183
C.189	Precisión MIL-CVCF Elephant, Consenso, Ruido 15 %	184
C.190	Precisión MIL-CVCF Elephant, Max Votos, Ruido 15 %	184
C.191	Precisión MIL-CVCF Elephant, Consenso, Ruido 20 %	184
C.192	Precisión MIL-CVCF Elephant, Max Votos, Ruido 20 %	185
C.193	Precisión MIL-CVCF Elephant, Consenso, Ruido 25 %	185
C.194	Precisión MIL-CVCF Elephant, Max Votos, Ruido 25 %	185
C.195	Precisión MIL-CVCF Elephant, Consenso, Ruido 30 %	186
C.196	Precisión MIL-CVCF Elephant, Max Votos, Ruido 30 %	186
C.197	Precisión MIL-IPF Elephant, Consenso, Ruido 0 %	186
C.198	Precisión MIL-IPF Elephant, Max Votos, Ruido 0 %	187
C.199	Precisión MIL-IPF Elephant, Consenso, Ruido 5 %	187
C.200	Precisión MIL-IPF Elephant, Max Votos, Ruido 5 %	187
C.201	Precisión MIL-IPF Elephant, Consenso, Ruido 10 %	188
C.202	Precisión MIL-IPF Elephant, Max Votos, Ruido 10 %	188
C.203	Precisión MIL-IPF Elephant, Consenso, Ruido 15 %	188
C.204	Precisión MIL-IPF Elephant, Max Votos, Ruido 15 %	189
C.205	Precisión MIL-IPF Elephant, Consenso, Ruido 20 %	189
C.206	Precisión MIL-IPF Elephant, Max Votos, Ruido 20 %	189
C.207	Precisión MIL-IPF Elephant, Consenso, Ruido 25 %	190
C.208	Precisión MIL-IPF Elephant, Max Votos, Ruido 25 %	190
C.209	Precisión MIL-IPF Elephant, Consenso, Ruido 30 %	190
C.210	Precisión MIL-IPF Elephant, Max Votos, Ruido 30 %	191
D.1.	Detalles Ruido 0 MIL max	193
D.2.	Detalles Ruido 0 MIL MIN	193
D.3.	Detalles Ruido 0 MIL EXTREME	193
D.4.	Detalles Ruido 0 BOW	194
D.5.	Detalles Ruido 0 CKNN	194
D.6.	Detalles Ruido 0 MAXDD	194
D.7.	Detalles Ruido 0 EMDD	194
D.8.	Detalles Ruido 0 BOOST	195
D.9.	Detalles Ruido 5 MIL MAX	195
D.10.	Detalles Ruido 5 MIL MIN	195
D.11.	Detalles Ruido 5 MIL EXTREME	196
D.12.	Detalles Ruido 5 BOW	196
D.13.	Detalles Ruido 5 CKNN	196
D.14.	Detalles Ruido 5 MAXDD	196
D.15.	Detalles Ruido 5 EMDD	197

D.16.Detalles Ruido 5 BOOST	197
D.17.Detalles Ruido 10 MIL MAX	197
D.18.Detalles Ruido 10 MIL MIN	198
D.19.Detalles Ruido 10 MIL EXTREME	198
D.20.Detalles Ruido 10 BOW	198
D.21.Detalles Ruido 10 CKNN	198
D.22.Detalles Ruido 10 MAXDD	199
D.23.Detalles Ruido 10 EMDD	199
D.24.Detalles Ruido 10 BOOST	199
D.25.Detalles Ruido 15 MIL MAX	200
D.26.Detalles Ruido 15 MIL MIN	200
D.27.Detalles Ruido 15 MIL EXTREME	200
D.28.Detalles Ruido 15 BOW	201
D.29.Detalles Ruido 15 CKNN	201
D.30.Detalles Ruido 15 MAXDD	201
D.31.Detalles Ruido 15 EMDD	201
D.32.Detalles Ruido 15 BOOST	202
D.33.Detalles Ruido 20 MIL MAX	202
D.34.Detalles Ruido 20 MIL MIN	202
D.35.Detalles Ruido 20 MIL EXTREME	203
D.36.Detalles Ruido 20 BOW	203
D.37.Detalles Ruido 20 CKNN	203
D.38.Detalles Ruido 20 MAXDD	203
D.39.Detalles Ruido 20 EMDD	204
D.40.Detalles Ruido 20 BOOST	204
D.41.Detalles Ruido 25 MIL MAX	204
D.42.Detalles Ruido 25 MIL MIN	205
D.43.Detalles Ruido 25 MIL EXTREME	205
D.44.Detalles Ruido 25 BOW	205
D.45.Detalles Ruido 25 CKNN	205
D.46.Detalles Ruido 25 MAXDD	206
D.47.Detalles Ruido 25 EMDD	206
D.48.Detalles Ruido 25 BOOST	206
D.49.Detalles Ruido 30 MIL MAX	207
D.50.Detalles Ruido 30 MIL MIN	207
D.51.Detalles Ruido 30 MIL EXTREME	207
D.52.Detalles Ruido 30 BOW	208
D.53.Detalles Ruido 30 CKNN	208
D.54.Detalles Ruido 30 MAXDD	208
D.55.Detalles Ruido 30 EMDD	208
D.56.Detalles Ruido 30 BOOST	209

Capítulo 1

Introducción

1.1. Introducción

Las técnicas en Ciencia de Datos están evolucionando en los últimos años a un ritmo vertiginoso gracias a auge de la Inteligencia Artificial para la clasificación de objetos. Una variante para ello es Multiple-Instance Learning (MIL) que forma parte del grupo de aprendizaje supervisado para predecir datos a través de conjuntos de entrenamiento los cuales tienen diferentes representaciones de un mismo objeto. Uno de los problemas más significativos al que se enfrenta esta arquitectura es el ruido presente en estas bolsas de datos, en la detección de objetos puede comprometer el sistema a un estado crítico, supongamos como ejemplo que no se detecta una célula cancerígena porque una bolsa de datos estaba mal etiquetada. Es por ello que técnicas de detección y limpieza de ruido en las etiquetas de las bolsas es uno de los elementos necesarios para continuar mejorando la arquitectura MIL.

1.2. Definición del Problema

Con el auge de las técnicas de Ciencia de Datos basadas en la clasificación de objetos y la continua mejora de los algoritmos para ello, uno de los paradigmas que más están influenciando en este campo es Multiple-Instance Learning ya que su esquema en bolsas de datos permite tener diferentes instanciaciones de objetos en un mismo Dataset, estas grandes cantidades de información, al igual que en las clasificaciones simples, son susceptibles a ruido en los valores de los atributos o en las etiquetas de las clases, siendo estos unos puntos críticos a tratar en el pre-procesado de los datos para la mejora en las predicciones de los clasificadores. Estas técnicas de filtrado están extendidas para la clasificación simple pero cuando entramos en el modelo de Multiple-Instance Learning observamos que no existe ninguna técnica para ello desarrollada en toda la literatura especializada.

1.3. Objetivos

Poder desarrollar el TFG desde cero sin que existan propuestas del mismo campo será uno de los puntos fuertes de este trabajo de investigación y desarrollo que, junto al auge de Python en los ámbitos de Machine Learning y Data mining, pretende desarrollar una biblioteca de técnicas de filtrado de ruido en software libre.

El presente proyecto plantea el desarrollo de tres diferentes tipos de filtros de ruido para Multiple-Instance Learning (MIL) escrito en Python, para ello nos inspiraremos en los esquemas que se usan para los actuales paradigmas de clasificación simple como son Ensemble Filter (EF) (Brodley y Friedl,

1999), Iterative Partitioning Filter (IPF) (Khoshgoftaar y Rebours, 2007) y Cross-Validated Committees Filter (CVCF) (Verbaeten y Assche, 2003), los cuales pretenden, mediante limpieza y etiquetado de las bolsas de datos, mejorar la precisión de los clasificadores. Estas herramientas se desarrollarán para el entorno de Python sin que exista soporte en su librería oficial de Ciencia de Datos, scikit-learn, para Multiple-Instance Learning, por eso no contaremos con Clasificadores con versiones estables para MIL, necesitando de un toolbox externo llamado MILpy (Arrieta y Mera, 2017).

1.4. Motivación

El desarrollo de este proyecto se originó en principio por la revolución digital que hay actualmente sobre Big Data, y tras hacer una pequeña investigación sobre el tema encontramos la arquitectura de Multiple-Instance Learning y la poca cobertura de librerías estables y oficiales que existe entorno a él. Además nos llamó la atención que teniendo esta poca cobertura tuviera aplicaciones muy importantes en bioquímica o en identificación de imágenes entre otras.

Gracias al conocimiento que adquirimos sobre pre-procesado de datos en el grado sabemos que esta área es muy importante para mejorar las evaluaciones de los clasificadores ya que esto evita tener resultados de poca calidad (Garbage) al igual que tener un buen modelo para entrenamiento, si alguna de las 2 áreas, datos o modelo es deficiente, el resultado también lo será, esto es el paradigma garbage in-garbage out, sólo datos de calidad pueden producir resultados de calidad, Figura 1.1 , así que proseguimos esta línea de investigación para preparar el proyecto y no encontramos nada en la literatura científica especializada sobre técnicas de filtrado de ruido en modelos Multiple-Instance Learning.

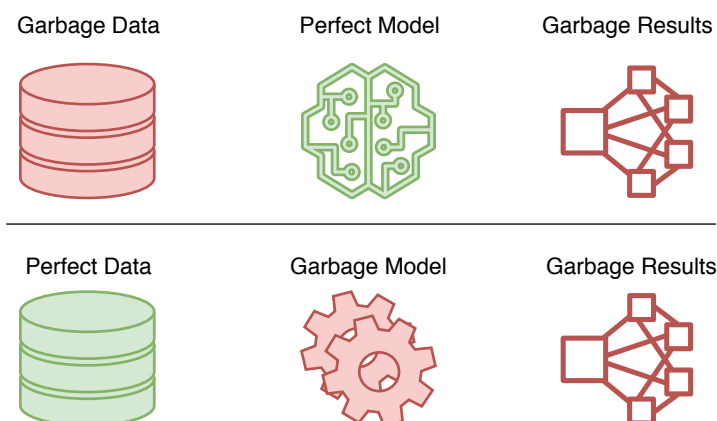


Figura 1.1: Paradigma Gargabe in-Garbage ou

Esto hace que la relevancia de este TFG empiece porque no existen propuestas de este tipo en las líneas de investigación actuales ni futuras y plantea una oportunidad única de expandir este proyecto aun más.

Cabe destacar que el objetivo de este TFG es llevar a cabo un trabajo académico, con lo que no se pretende entrar en ningún mercado.

1.5. Organización del documento

El resto del presente documento se divide en los siguientes capítulos:

- Capítulo 2: Mostraremos los Timeline Original y Real del proyecto y el por qué de los cambios entre estos.
- Capítulo 3: Abordaremos toda la terminología referente al proyecto en materia de Ciencia de Datos para tener conocimiento suficiente para tratar con el trabajo desarrollado.
- Capítulo 4: Se presentan las herramientas usadas en el desarrollo del proyecto y su justificación.
- Capítulo 5: Se desarrollará el análisis y el diseño de los Filtros de ruido.
- Capítulo 6: Se presentan los parámetros de configuración del estudio para englobar el marco completo y las tablas resumen analizadas.
- Capítulo 7: Conclusiones finales sobre la información del capítulo anterior y las posibles vertientes futuras, así como una reflexión final.
- Anexo A: Manual de Usuario de pyMIL-BNC para librería e Interfaz gráfica.
- Anexo B: Código fuente en Python de cada esquema de filtro de pyMIL-BNC así como de la interfaz gráfica.
- Anexo C: Contiene todas las tablas de resultados obtenidos de las diferentes ejecuciones de los algoritmos.
- Anexo D: Contiene las tablas detalladas de los resultados.

Capítulo 2

Planificación

2.1. Gestión Temporal del Proyecto

La pretensión y el planteamiento de trabajo original se muestra en el diagrama de la Figura 2.1 que daban como resultado la entrega del proyecto en el mes de Mayo de 2019.

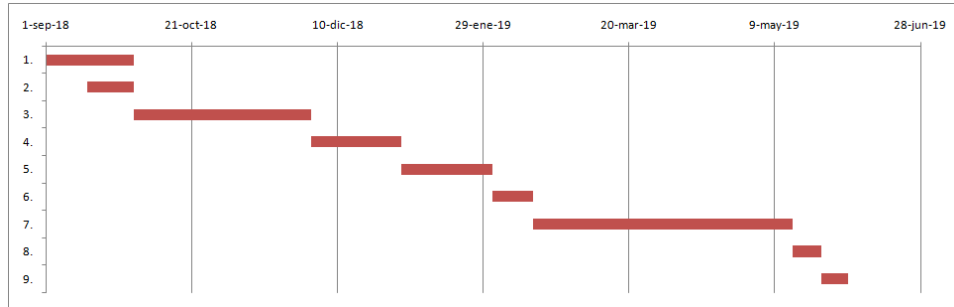


Figura 2.1: Diagrama de Gantt de la Planificación Original

Las tareas desarrolladas en el timeline se enumeran a continuación:

1. Aprendizaje Data Science.
2. Aprendizaje MIL.
3. Aprendizaje Python.
4. Búsqueda herramientas MIL para Python.
5. Diseño Filtros MIL.
6. Pruebas Filtros.
7. Ejecución clúster Hercules.
8. Recogida de Datos.
9. Análisis de Datos.

En el siguiente diagrama, Figura 2.2, vemos el timeline real que ha conllevado retrasos en la entrega y cambio en los periodos de las tareas.

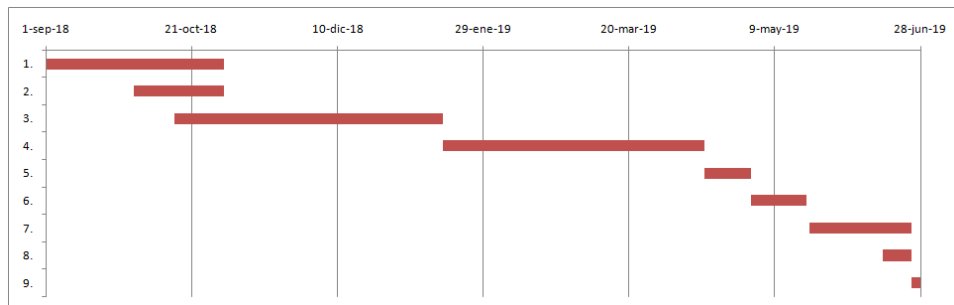


Figura 2.2: Diagrama de Gantt de la Planificación Real

2.1.1. Estructura de descomposición del trabajo

Para explicar los contratiempos surgidos durante el desarrollo del TFG que han llevado al cambio del timeline presentado en el punto anterior, nos apoyaremos en el EDT (Estructura de descomposición del trabajo) del proyecto que vemos en la Figura 2.3.

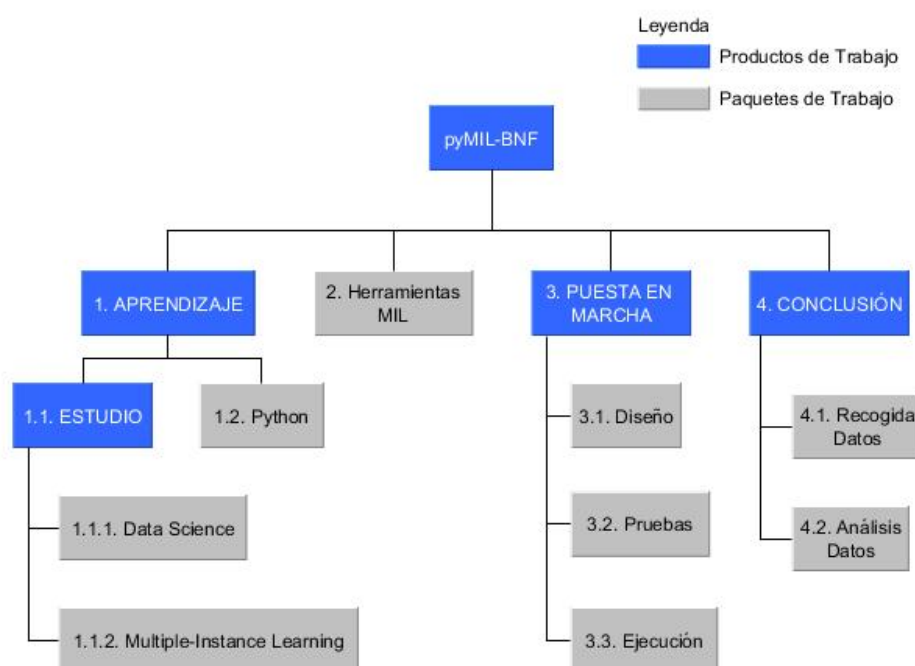


Figura 2.3: Estructura de descomposición del trabajo

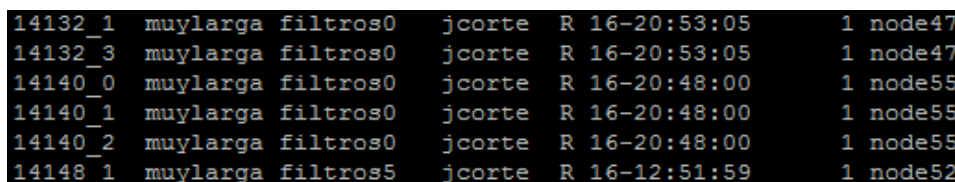
La EDT del Proyecto es una descomposición jerárquica del este y nos muestra los trabajos (Leyenda Figura 2.3: productos de trabajo) que hay que realizar para finalizar el proyecto, a su vez estos trabajos los divide en componentes pequeños (Leyenda Figura 2.3: paquetes de trabajo). De forma que el primer nivel lo componen las tareas principales y son secuenciales para el desarrollo del proyecto y a partir de estos en el diagrama, si las tareas están al mismo nivel no tienen dependencia, y si son descendentes, dependerán de que el nodo padre se haya completado para iniciarse.

2.1.2. Comparativa

Por lo expuesto veremos que las tareas serán más críticas a medida que afecten a más paquetes de trabajo, en nuestro caso, casi todas las fases tienen tareas descendentes, por lo que podemos deducir que cuanto antes ocurra una adversidad, más crítica será la tarea.

A continuación enumeramos cada uno de estos puntos críticos y los paquetes de trabajo (PT) afectados:

- En los PT 1.1.1. y 1.1.2., encontramos que la mayoría de la información relevante en estas áreas estaban en Inglés, un idioma que no se domina como el propio y por eso la lectura y entendimiento sobre este área presentó el retraso.
- Para PT 1.2., Python aunque es un lenguaje sencillo de aprender y así fue, no lo es tanto para sus bibliotecas científicas, para lo que al igual que pasó anteriormente la información de sus librerías está en inglés y lo que provocó de nuevo un retraso en el proyecto.
- PT 2., fue la tarea que más retraso aplicó en la planificación, al no existir nada oficial y con versiones estables, se tuvo que afrontar esta fase a base de prueba y error con los proyectos que autores propios habían desarrollado hasta dar con algo que funcionara y encajara en nuestro propio desarrollo.
- PT 3.3. El último problema que vino derivado de estos retrasos fue que el tiempo que se debía de invertir en los estudios de los casos de uso de los filtros se vio drásticamente reducido, y por ende se tuvieron que recortar los dataset que se iban a utilizar, ya que la ejecución de alguna de las tareas llevaría más de 2 semanas como se ve en la Figura 2.4 y hay límite de ejecuciones simultáneas en Hercules.



14132_1	muylarga filtros0	jcorte	R 16-20:53:05	1 node47
14132_3	muylarga filtros0	jcorte	R 16-20:53:05	1 node47
14140_0	muylarga filtros0	jcorte	R 16-20:48:00	1 node55
14140_1	muylarga filtros0	jcorte	R 16-20:48:00	1 node55
14140_2	muylarga filtros0	jcorte	R 16-20:48:00	1 node55
14148_1	muylarga filtros5	jcorte	R 16-12:51:59	1 node52

Figura 2.4: Ejemplo de tiempo de ejecución de alguna de las tareas mandadas a Hercules

También cabe destacar que no todo han sido retrasos, el diseño y la implementación de los filtros en Python fue más rápida de lo esperado.

En la Tabla 2.1 vemos una comparación de los tiempos y fechas para tener una visión más clara de los tiempos invertidos en las tareas y sus diferencias temporales.

Tabla 2.1: Comparación de los tiempos de los Diagramas de Grantt

PT	ORIGINAL			REAL		
	Fecha Inicio	Fecha Fin	Dias	Fecha Inicio	Fecha Fin	Dias
1.1.1	1-sep-18	1-oct-18	30	1-sep-18	1-nov-18	61
1.1.2	15-sep-18	1-oct-18	16	1-oct-18	1-nov-18	31
1.2	1-oct-18	1-dic-18	61	15-oct-18	15-ene-19	92
2	1-dic-18	1-ene-19	31	15-ene-19	15-abr-19	90
3.1	1-ene-19	1-feb-19	31	15-abr-19	1-may-19	16
3.2	1-feb-19	15-feb-19	14	1-may-19	20-may-19	19
3.3	15-feb-19	15-may-19	89	21-may-19	25-jun-19	35
4.1	15-may-19	25-may-19	10	15-jun-19	25-jun-19	10
4.2	25-may-19	3-jun-19	9	25-jun-19	28-jun-19	3

La tareas en rojo han aumentado el tiempo de ejecución de la tarea, en verde han disminuido y sin color no ha variado su tiempo, por consiguiente las fechas de inicio y finalización también varían.

2.2. Gestión de comunicaciones

Las comunicaciones entre el Tutor y el alumno del TGF para informar del estado del proyecto y resolución de errores, dudas o aclaraciones que han surgido durante el desarrollo de este, se han realizado mediante reuniones bajo petición por ambas partes, correo electrónico y vía telefónica. La comunicación ha sido constante y fluida durante todo el proyecto.

2.3. Gestión de costes

Los costes derivados del desarrollo del proyecto se representan en la siguiente Tabla 2.2.

Tabla 2.2: Marco de costos del proyecto

Concepto	Taea asociada	Precio	Canidad	Subtotal
<i>Recursos Humanos</i>	PT 2,3 y 4	20€/hora	1464 horas	29280€
<i>Software</i>	PT 2, 3 y 4	0	-	0
<i>Hardware Hercules</i>	PT 3.3	2,78€/hora	840 horas	2335,2€
<i>Puesto de trabajo</i>	PT 2, 3 y 4	0,8€/hora	1464 horas	1171,2€
TOTAL				32.786,4€

2.4. Gestión de costes

La gestión de riesgos se ha basado en utilizando las propuestas de (S., Cano, y Arán, 1997) para el desarrollo de un proyecto informático. Los posibles riesgos, ocurrencias e impacto en días se muestran en la Tabla 2.3.

Tabla 2.3: Gestión de Riesgos de pyMIL-BNF

Riesgo	Probabilidad	Impacto en días
A. Elaboración de planificación		
A2. Planificación optimista <<Mejor caso>>	20 %	15
A3. La planificación no incluye tareas necesarias	10 %	5
A11. Un retraso en una tarea produce un retraso global en el proyecto	50 %	1-20
B. Organización y gestión		
B2. El proyecto languidece demasiado el inicio difuso	5 %	1
C. Ambiente de desarrollo		
C.5. Las herramientas de desarrollo no funcionan como se esperaba; el personal de desarrollo necesita tiempo para resolverlo o adaptarse.	25 %	10
C.7. La curva de aprendizaje para la nueva herramienta de desarrollo es más larga de lo esperado.	10 %	2
D. Usuarios finales (No se aplica)		
E. Cliente (No se aplica)		
F. Personal contratado (No se aplica)		
G. Requisitos		
G.2. Los requisitos no se han definido correctamente y su redefinición aumenta el ámbito del proyecto.	10 %	2
G.3. Se añaden requisitos extra	10 %	5
H. Producto		
H.3. Utilizar lo último en informática alarga la planificación de forma impredecible.	20 %	2
I. Fuerzas mayores (No se aplica)		
J. Persona		
J.5. La falta de motivación y de moral reduce la productividad	10 %	2
J.6. La falta de la especialización necesaria aumenta los defectos y la necesidad de repetir el trabajo.	10 %	3
J.7. El personal necesita un tiempo extra para acostumbrarse a trabajar con herramientas o entornos nuevos.	10 %	2
K. Diseño e implementación		
K.4. La utilización de metodologías desconocidas deriva en un periodo extra de formación y tener que volver atrás para corregir los errores iniciales cometidos en la metodología.	40 %	10
K.7. Las bibliotecas de código o clases tienen poca calidad, y generan una comprobación extra, corrección de errores y la repetición de algunos trabajos.	50 %	10
K.9. Los componentes desarrollados por separado no se pueden integrar de forma sencilla, teniendo que volver a diseñar y repetir algunos trabajos.	25 %	5
L. Proceso (No se aplica)		

Capítulo 3

Antecedentes

3.1. Ciencia de Datos

Para entender el ámbito del proyecto y situarnos en contexto antes de explicar el MIL es necesario saber todos los conceptos que involucra.

La Ciencia de Datos es una disciplina que abarca los campos de la Estadística y Machine Learning entre otros, todos ellos para extraer información útil de los datos también llamado extracción del conocimiento.

El proceso de extracción del conocimiento en bases de datos o KDD (Brachman y Anand, 1996) es un modelo para identificar patrones útiles en grandes cantidades de información, normalmente es confundido con la Minería de Datos, siendo este uno de los pasos pertenecientes al proceso KDD. El proceso completo de KDD se descompone en 6 etapas:

1. Comprensión del dominio.
2. Creación de un Dataset objetivo.
3. Preprocesado y transformación de Datos.
4. Minería de Datos.
5. Evaluación del conocimiento.
6. Uso del conocimiento descubierto.

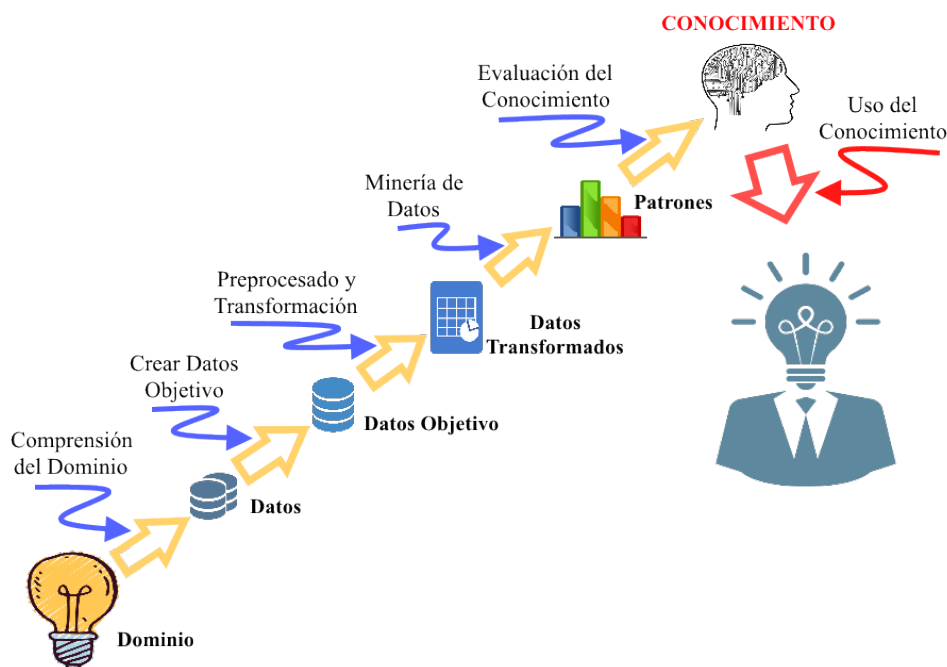


Figura 3.1: Proceso Knowledge Discovery in Databases (KDD)

La ciencia de datos presenta una arquitectura de trabajo denominada 'Pipe and Filter', que se basa en un patrón que divide la tarea en pasos de procesamiento secuenciales Figura 3.2.



Figura 3.2: Pipe and Filter

3.2. Minería de Datos

Esta fase de Minería de Datos es la más conocida del proceso KDD aunque no sea la que más tiempo o esfuerzo se lleva en el proceso como vemos en la Figura 3.3.

Este proceso explora grandes cantidades de datos para obtener patrones o modelos útiles y sus relaciones y posteriormente analizar los resultados. El uso de algoritmos, supervisados o no supervisados dependiendo del objetivo del análisis, aquí es fundamental para recorrer los datos y obtener nueva información, lo que se conoce también como fase de analítica. Algunas de las técnicas usadas son: Redes Neuronales, Regresión, Agrupamiento...

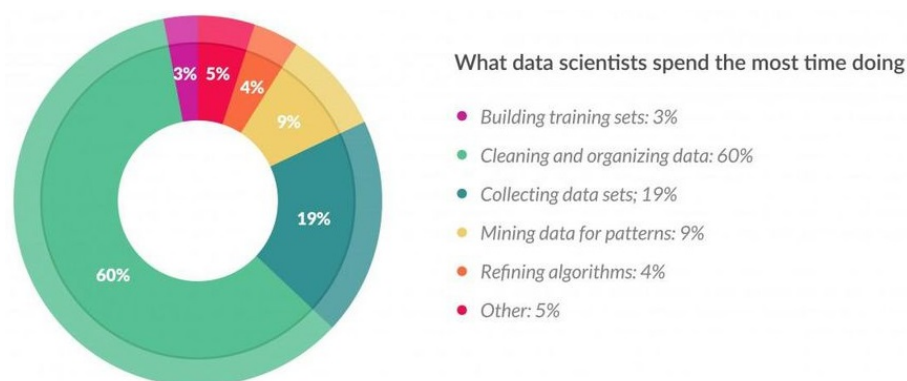


Figura 3.3: Organización del tiempo en Data Science. **Fuente:** (Forbes, 2016)

A continuación describiremos el marco teórico de la Minería de Datos usado en el desarrollo del presente TFG, como son Machine Learning, vali-

dación y evaluación.

3.2.1. Machine Learning

Este paradigma de la Inteligencia Artificial crea sistemas que generan patrones a través de los datos de entrenamiento. Estos patrones que realmente son algoritmos auto-generados la máquina los usa para analizar datos y ser capaz de predecir comportamientos, además son capaces de seguir evolucionando a medida que analizan esos nuevos datos. Realmente esa evolución viene dada por la optimización de la función objetivo del algoritmo generado de forma que se minimice la pérdida (Función de pérdida).

“La función de pérdida es como la regla para ganar un juego de mesa, optimizarla es descubrir cómo jugar para obtener la mejor puntuación posible. Kozyrkov

Los algoritmos utilizados en Machine Learning normalmente se dividen en 2 grandes categorías, aunque hay otras intermedias, denominados «Supervisados» y «No Supervisados», en función de si se conoce o no la clase a la que pertenecen los datos que se usan para construir los modelos.

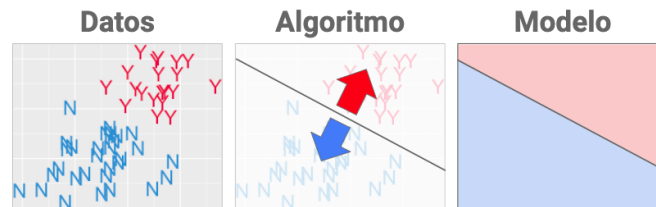


Figura 3.4: Pasos para obtener modelo de Datos. **Fuente:** (Kozyrkov, 2019)

Dependiendo del tipo de problemas que deseemos clasificar las mediciones varían, por ejemplo en problemas de regresión es usado el error cuadrático medio y en problemas de clasificación, como los que tenemos en este proyecto, usualmente la exactitud y la precisión.

3.2.2. Validación

La validación es una técnica para crear modelos mediante una partición de entrenamiento y evaluar con un conjunto de test externo, esto hace que podamos optimizar los parámetros a un modelo más eficiente. La evaluación de la clasificación de los resultados sigue unos protocolos de validación, el más usado es el esquema "k-fold cross validation", que consiste en dividir los datos en K particiones iguales, el clasificador se entrenará con las k-1 partes

restantes para cada partición i , la evaluación final resultará de la media de los K resultados obtenidos.

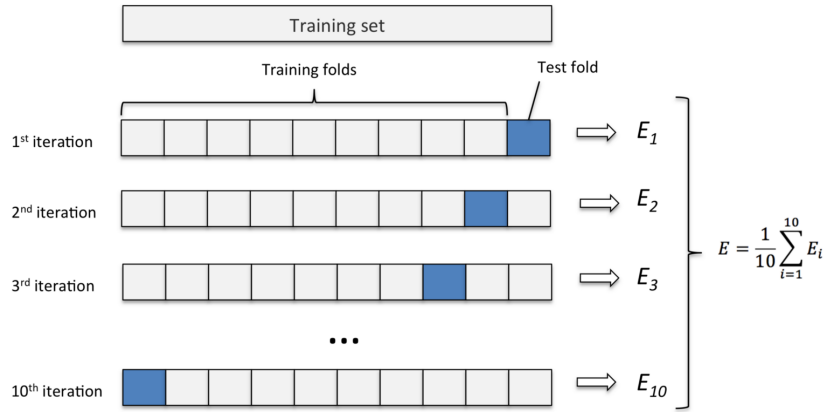


Figura 3.5: Esquema K-Fold Cross Validation **Fuente:** (Raschka, 2016)

Todos los modelos de validación usan una parte de los datos para entrenar el clasificador y posteriormente son testados con datos reales para comprobar su eficacia. Esquemas de validación para MIL:

1. Holdout.
2. Repeated Holdout.
3. k-fold cross validation, Figura 3.5.
4. Repeated cross validation.
5. Bootstrapping.

3.2.3. Medidas de evaluación

Para poder evaluar como de bueno es el clasificador entrenado es necesario tener alguna medida para comparar su efectividad con otros o su mejora respecto a diferentes configuraciones, para ello disponemos de diferentes métricas:

1. Exactitud: Nos devuelve que fracción de está bien clasificada del conjunto de Test.

$$Acc = \frac{N \text{ predicciones correctas}}{N \text{ total de predicciones}}$$

2. Tasa de error: Nos devuelve la fracción de elementos mal clasificados del conjunto de Test.

$$Err = 1 - Acc$$

3. Media geométrica: Nos da el promedio de la exactitud.
4. Coeficiente kappa de Cohen: Es una medida estadística algo más compleja que las anteriores, puesto que incorpora un componente para ajustar el efecto del azar.

$$K = \frac{Pr(a) - Pr(e)}{(1 - Pr(e))}$$

Siendo $Pr(a)$ el acuerdo observado relativo entre los observadores, y $Pr(e)$ es la probabilidad hipotética de acuerdo por azar, utilizando los datos observados para calcular las probabilidades de que cada observador clasifique aleatoriamente cada categoría.

5. Curva ROC (curva de característica operativa del recepto): Muestra el rendimiento de un modelo de clasificación en todos los umbrales de clasificación representados por Tasa de verdaderos positivos y Tasa de falsos positivos Figura 3.6.

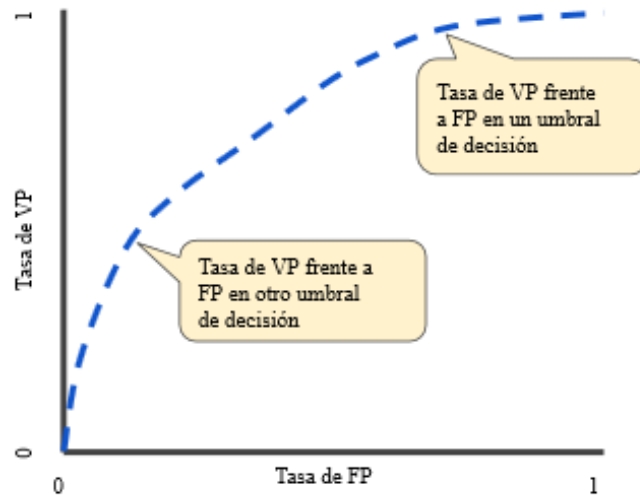


Figura 3.6: Curva de Característica Operativa del Recepto.

Fuente: (Mishra, 2018)

6. AUC (Área bajo la curva ROC): mide toda el área bidimensional por debajo de la curva ROC completa Figura 3.7.

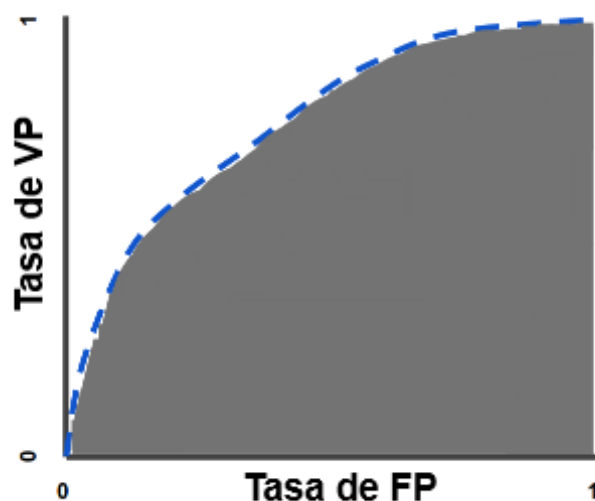


Figura 3.7: AUC: Área bajo la curva ROC. **Fuente:** (Mishra, 2018)

3.3. Aprendizaje Supervisado: Multiple-Instace Learning

Como se ha comentado anteriormente en el aprendizaje supervisado hay conocimiento previo sobre la clase a la que pertenecen las instancias, estando estas pre-etiquetadas, y aprendiendo a partir de este conjunto, denominado conjunto de entrenamiento, un método o modelo para predecir las nuevas instancias que se le presenten.

Los clasificadores más comunes que pertenecen a este tipo son: Redes Neuronales (NN), Máquinas de Vectores de Soporte (SVMs), K vecinos más cercanos (K-NN), Árboles de decisión, etc.

Hasta ahora imaginamos las tablas de datos como una serie de instancias etiquetadas cada una con su clase, pero en este proyecto se va a tratar con un modelo de datos más avanzado, Multiple-Instace Learning, introducido por Dietterich (Dietterich, Lathrop, y Lozano-Pérez, 1997), el cual es un tipo de aprendizaje supervisado, en esta estructura encontramos que el objeto etiquetado ahora es una bolsa de datos, cada una de estas "bolsas" es un conjunto de instancias, cada una perteneciente a una característica, como las tradicionales estructuras, relacionadas entre ellas para crear el conjunto de cualidades del objeto etiquetado.

Sigue un esquema de clasificación binaria, una bolsa es positiva si al menos una de sus instancias lo es, y negativa si todas son clasificadas como negativas, Figura 3.8.

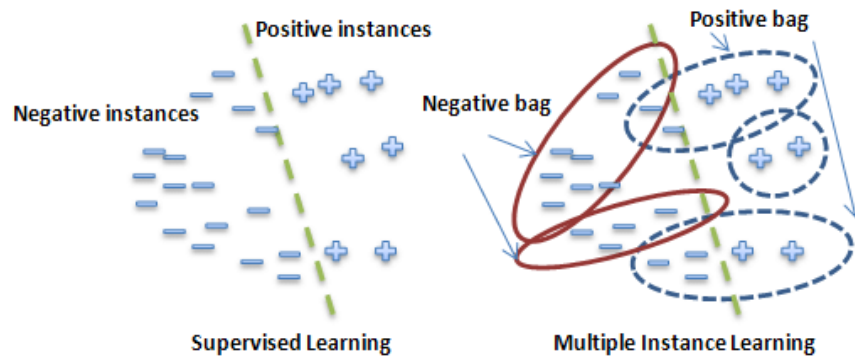


Figura 3.8: Esquema de Clasificación MIL.

Fuente: (Kumar y cols., 2011)

3.4. Pre-procesamiento de Datos

El Pre-procesado de los datos previo a su uso es una parte esencial en cualquier ámbito de la Ciencia de Datos, tal como se representa en la Figura 3.3 es la que más tiempo y esfuerzo requiere, normalmente nos referimos a esta parte como un conjunto de técnicas para preparar y reducir los datos para el uso adecuado en los algoritmos.

La fase preparación incluye técnicas de integración, limpieza, normalización, identificación de ruido y transformación de los datos y la fase reducción de la selección de características, selección de instancias y discretización para obtener un conjunto reducido de datos útiles. (Garcia, Luengo, y Herrera, 2014)

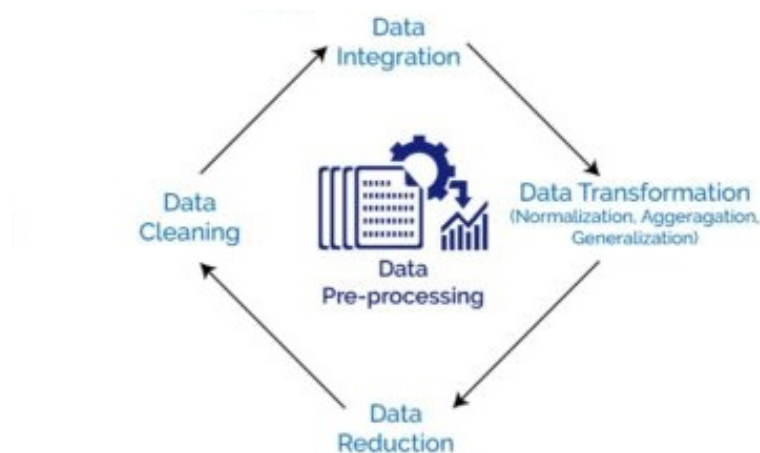


Figura 3.9: Esquema de Pre-procesamiento de Datos. **Fuente:** (Media, 2017)

3.4.1. Tratamiento de Ruido

El tratamiento de ruido tiene conjuntas las tareas de identificación y limpieza, lo más común es encontrar más técnicas de limpieza de datos, encontrando inconsistencias y sustituyendo el valor por uno adecuado a su entorno o incluso eliminándolo, esto se debe a que en tiempos de computación es menos costoso que identificar la inconsistencia, saber a qué se debe y tratarla acorde a su procedencia.

La variedad de técnicas es muy amplia cuando hablamos de ruido en los atributos de las instancias, pero la dificultad se vuelve mayor cuanto el etiquetado de la clase a la que pertenecen es la que presenta el ruido. Estos son los 2 tipos de ruido, Figura 3.10, que nos podemos encontrar en los conjuntos de datos: Ruido de atributos y Ruido de clase (Zhu y Wu, 2004).

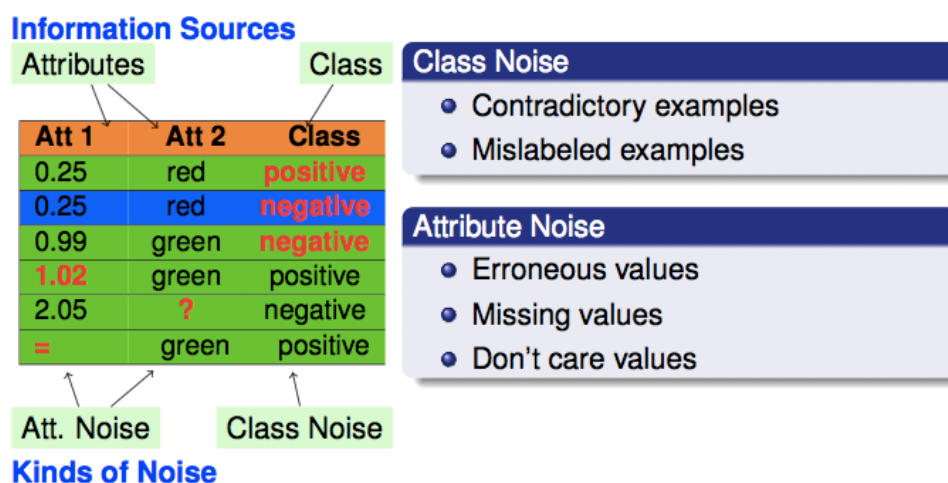


Figura 3.10: Tipos de Ruido. **Fuente:** (Morales y cols., 2017)

Cabe destacar la importancia del tratamiento de ruido, ya que éste ensucia los datos y cuando tratamos con modelos de predicción, como es el caso de este proyecto, esos atributos o clases con ruido interfieren en la creación de un buen modelo, haciendo que la precisión dependa del nivel del ruido en los datos directamente. Además es importante saber que da igual la herramienta que se use para construir el modelo, si los datos están contaminados, este nunca será lo suficientemente preciso (Ferreira, 2007).

3.4.2. Tipos de Ruido

Como hemos visto en la Figura 3.10 existen 2 tipos de ruido dependiendo a que datos son afectados:

- Ruido en los valores de los atributos: Este ruido se presenta en la caracterización de las instancias de un Dataset y dependiendo de la forma que tenga el valor del atributo encontramos:
 1. Que el valor es nulo o no existe.
 2. Que el atributo aunque presenta un valor válido, está fuera de la escala correcta.
 3. Que el atributo tiene un valor no válido.
- Ruido en las etiquetas de la clase: Este ruido representa que el valor de la etiqueta de la instancia no es correcto, en ningún caso nulo. Así pues se presenta de estas formas:
 1. Que la etiqueta de la clase es incorrecta, es decir sabemos de está mal clasificada.
 2. Que tenemos las mismas instancias con diferente etiquetado de la clase a la que pertenecen, esto hace que su clasificación sea contradictoria.

Poder filtrar estos tipos de ruido en los dataset y tratarlos de forma válida harán que nuestro clasificador mejore en su evaluación, siempre y cuando su detección sea posible.

3.4.3. Filtrado de Ruido en Clases

Puesto que el ruido de clase es el más disruptivo de las 2 tipologías presentadas, nos centraremos en esquemas de filtrado que podamos aplicar en él en este TFG.

Las técnicas más usuales de filtrado de ruido en clases usan un modelo de votación y otro de validación cruzada. A continuación trataremos las 3 técnicas en las que posteriormente nos inspiraremos para crear nuestros filtros:

1. Ensemble Filter (EF) (Brodley y Friedl, 1999):
El dataset se divide en K particiones (Cross-Validation), para cada modelo de validación, K, un conjunto de 3 Clasificadores evalúa el conjunto de entrenamiento, finalmente para eliminar las etiquetas ruidosas se aplica el esquema de consenso o mayoría de votos a la combinación de las K validaciones.
2. Cross-Validated Committees Filter (CVCF) (Verbaeten y Assche, 2003):
El dataset se divide en K particiones (Cross-Validation), para cada modelo de validación, K, un Clasificador evalúa el conjunto de entrenamiento, finalmente para eliminar las etiquetas ruidosas se aplica el

esquema de consenso o mayoría de votos a la combinación de las K validaciones.

3. Iterative-Partitioning Filter (IPF) (Khoshgoftaar y Rebours, 2007):

Al filtro se le definen N iteraciones, para cada N , el dataset se divide en K particiones (Cross-Validation), para cada modelo de validación, K , un Clasificador evalúa el conjunto de entrenamiento, finalmente para eliminar las etiquetas ruidosas se aplica el esquema de consenso o mayoría de votos a la combinación de las K validaciones de modo que para la siguiente iteración el dataset ya ha podido ser filtrado.

Estas técnicas de filtrado de ruido son las que se desarrollan a lo largo del proyecto para adaptarlas a Multiple-Instance Learning, aunque han sido creadas para estructuras de datos tradicionales, comprobaremos su eficacia para el uso de bolsas de datos.

Capítulo 4

Marco Tecnológico

4.1. Herramientas para la gestión del proyecto

4.1.1. GitHub

GitHub (Chris Wanstrath y Chacon, 2019) es una suite para alojar y administrar proyectos en la nube utilizando el sistema de control de versiones Git. Nuestra cuenta está creada con email de la Universidad de Granada y se puede acceder a través de <https://github.com/jcarlosorte/TFG>.



Figura 4.1: Código QR para acceso al proyecto alojado en GitHub

4.1.2. GitHub Desktop

Es un programa Open Source para manipular los repositorios alojado en GitHub a través de una interfaz gráfica para Windows sin tener que usar el tradicional Git Bash (GitHub, 2019).

4.1.3. Hercules

Debido a la necesidad de procesamiento para obtener datos de los clasificadores, el uso de los portátiles o sobremesa personales se relegaron solo a la parte de programación y pequeñas pruebas de configuración de los algoritmos. Para este gran calculo fue necesario solicitar el uso del cluster de la Universidad de Granada "Hercules" para procesamiento de colas `hercules.ugr.es`.

Las tareas fueron mandadas a través de scripts configurados mediante SLURM, que son ficheros de texto plano con una serie de indicaciones que permitirán al sistema de colas conocer algunos datos sobre la ejecución.

```
1 #!/bin/bash
2
3 #SBATCH -J filtros30_9ArrayTFG
4 #SBATCH -p muylarga
```

```

5 #SBATCH -o /home/jcorte/TFG/tablas/Ruido30/slurm_output_%A_dir%.dat
6 #SBATCH -e /home/jcorte/TFG/tablas/Ruido30/slurm_err_%A_dir%.dat
7 #SBATCH --array=0-5
8
9 DIR[0]=/home/jcorte/TFG/tablas/Ruido30/consenso/CVCF/
10 DIR[1]=/home/jcorte/TFG/tablas/Ruido30/consenso/EF/
11 DIR[2]=/home/jcorte/TFG/tablas/Ruido30/consenso/IPF/
12 DIR[3]=/home/jcorte/TFG/tablas/Ruido30/maxVotos/CVCF/
13 DIR[4]=/home/jcorte/TFG/tablas/Ruido30/maxVotos/EF/
14 DIR[5]=/home/jcorte/TFG/tablas/Ruido30/maxVotos/IPF/
15
16 cd ${DIR[${SLURM_ARRAY_TASK_ID}]}; python3 menu_9.py

```

En la Figura 4.2 podemos ver un ejemplo de gestión de las tareas que se mandaron, como se puede observar hay tareas que llegan a tardar 10-15 días en ejecutarse y otras algunas horas, los 4 factores que influyen en el tiempo de ejecución son:

1. Número de instancias del dataset.
2. Complejidad del Clasificador.
3. Complejidad del Filtro.
4. Valor de K en el esquema de validación cruzada.

```

Last login: Sun Jun 23 14:08:46 2019 from 83.61.143.105
jcorte@hercules:~$ squeue -u jcorte

```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	ODELIST (REASON)
14132_1	muylarga	filtros0	jcorte	R	13-22:20:34	1	node47
14132_3	muylarga	filtros0	jcorte	R	13-22:20:34	1	node47
14140_0	muylarga	filtros0	jcorte	R	13-22:15:29	1	node55
14140_1	muylarga	filtros0	jcorte	R	13-22:15:29	1	node55
14140_2	muylarga	filtros0	jcorte	R	13-22:15:29	1	node55
14148_1	muylarga	filtros5	jcorte	R	13-14:19:28	1	node52
14186_1	muylarga	filtros1	jcorte	R	6-06:16:58	1	node45
14188_1	muylarga	filtros1	jcorte	R	6-04:56:33	1	node44
14211_5	muylarga	filtros2	jcorte	R	5-02:44:09	1	node58
14221_5	muylarga	filtros2	jcorte	R	5-02:43:07	1	node48
14265_5	muylarga	filtros3	jcorte	R	4-10:16:42	1	node54
14381_0	muylarga	filtros3	jcorte	R	3:00:32	1	node31
14381_1	muylarga	filtros3	jcorte	R	3:00:32	1	node55

Figura 4.2: Ejemplo de gestión de colas en Hercules

4.1.4. PuTTY

Es un cliente SSH de Licencia MIT (Licencia de software libre permisiva), que ejecutaremos bajo Windows y que necesitaremos para conectarnos a Hercules y usar el bash de Unix para administrar y ejecutar tareas en Python en remoto (Tatham, 2019).

4.1.5. FileZilla

Este software nos permite la descarga y envío de archivos a nuestro almacenamiento de la cuenta de Hercules a través de los protocolos de FTP (Kosse, 2019) y tiene Licencia GPLv2.

4.2. Herramientas para modelado del proyecto

4.2.1. yEd graph editor

Es una aplicación Freeware de escritorio que se usa para generar diagramas de flujo, diagramas UML o mapas conceptuales (of Tuebingen, 2001).

4.3. Herramientas para desarrollo del proyecto

4.3.1. Anaconda Navigator

Es una suite con licencia BSD (software libre permisiva) para los lenguajes Python y R, utilizada en ciencia de datos, y aprendizaje automático (machine learning). Las versiones de los paquetes se administran mediante el sistema de gestión de paquetes conda. Nosotros usaremos Spyder el cual es un entorno de desarrollo integrado (IDE) multiplataforma para Python (Anaconda, 2012).

4.3.2. Python

En los últimos años Python ha escalado puestos como lenguaje de programación para Data Science gracias a sus bibliotecas dedicadas a esta área como Pandas, Numpy, Matplotlib, SciPy, scikit-learn, etc. Además, de otras más avanzadas, como Tensorflow, Keras y Pytorch para Deep Learning. Pero no es solo esto, si no también su sencillez y ser un lenguaje de programación interpretado, no necesita compilador, lo que ha hecho que prolifere su uso en todos los entornos, haciendo que las personas que no han estudiado específicamente programación también pueda desarrollar sus propias aplicaciones (Python, 2019). Además Python opera bajo su propia licencia "Python Software Foundation License", que es un tipo de licencia de software libre permisiva.

4.3.2.1. Justificación

Como vemos en la Figura 4.3 Python está copando el mercado según la encuesta realizada por la plataforma Kaggle a 24.000 profesionales de datos. Además en esta misma encuesta, 3 de cada 4 profesionales de datos recomendaron que los aspirantes a científicos de datos aprendan primero sobre Python.

Según la encuesta anual de www.kdnuggets.com (KDnuggets, 2018) sobre herramientas Data Science y Machine Learning, vemos el avance de Pyt-

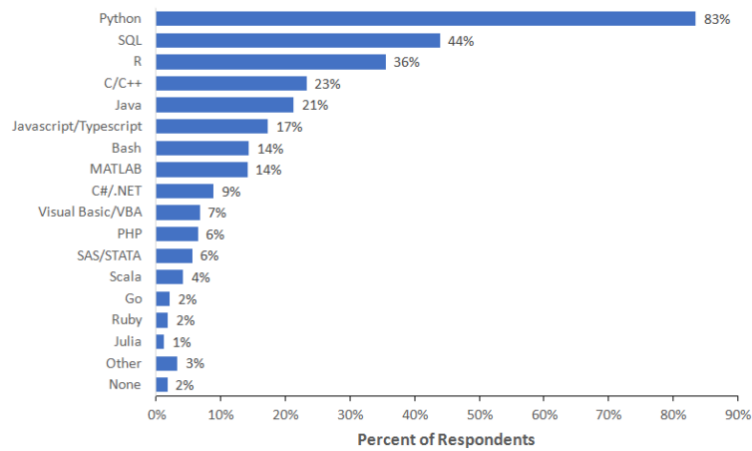


Figura 4.3: Lenguajes de programación más usados en 2018

hon en los últimos 3 años, Figura 4.4 y como desbanca a R como plataforma de programación para esta área.

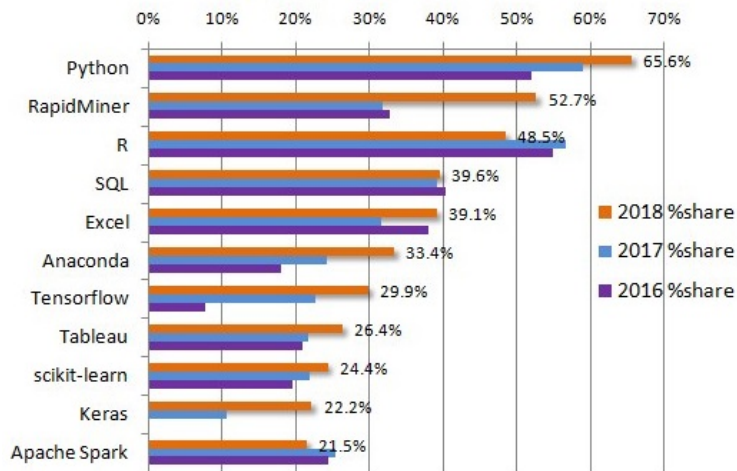


Figura 4.4: Encuesta sobre herramientas más usadas en Data Science, 2016-2018

4.3.2.2. Bibliotecas

Python fue lanzado en la década de los 90 y creado por Guido van Rossum (Python, 2019), desde entonces ha sabido sacar provecho de su dinamismo, dado que soporta programación orientada a objetos, programación funcional y programación imperativa con tipado dinámico y multiplataforma. Gracias a esto el desarrollo de bibliotecas para Ciencia de Datos es muy amplia y con gran soporte y actualización, las principales que usaremos en el proyecto para crear los algoritmos de filtros de ruido son:

1. NumPy. Provee un los arreglos n-dimensionales. <https://www.numpy.org>
2. SciPy. Posee gran variedad de módulos de alto nivel, como transformada discreta de Fourier, álgebra lineal, y matrices de optimización. <https://www.scipy.org/>
3. Pandas. Provee estructuras, como los son los data frames, y funciones para el tratamiento de los datos y los nulos. <https://pandas.pydata.org/>
4. Scikit-learn. Biblioteca especializada en todo el tema del Machine Learning y posee algoritmos para problemas de clasificación, regresión, clustering, procesamiento de datos, entre muchos otros. <https://scikit-learn.org>
5. PyQt. Es una biblioteca gráfica Qt para el lenguaje de programación Python. <https://riverbankcomputing.com/software/pyqt/intro>

Aunque estas son solo algunas como hemos comentado existen muchas más, cada una dedicada al tratamiento de datos, análisis y visualización de forma específica Figura 4.5.













Github data Python 2018										
Library Name	Type	Commits	Contributors	Releases	Watch	Star	Fork	Commits/ Contributors	Commits/ Releases	Star/ Contributors
 matplotlib	Visualization	25 747	725	70	498	7 292	398	36	368	10
 Bokeh	Visualization	16 983	294	58	363	7 615	2 000	58	293	26
 plotly	Visualization	2 906	48	8	198	3 444	850	61	363	72
Seaborn	Visualization	2 044	83	13	205	4 856	752	25	157	59
pydot	Visualization	169	12	12	17	193	80	14	14	16
 leann	Machine learning	22 753	1 084	86	2 114	28 098	14 005	21	265	26
XGBoost LightGBM CatBoost	Machine learning	3277	280	9	868	11 991	5 425	12	364	43
	Machine learning	1083	79	14	363	5 488	1 467	14	77	69
	Machine learning	1509	61	20	157	2 780	369	25	75	46
elis	Machine learning	922	6	22	39	672	89	154	42	112
 SciPy	Data wrangling	19 150	608	99	301	4 447	2 318	31	193	7
 NumPy	Data wrangling	17 911	641	136	390	7 215	2 766	28	132	11
 pandas	Data wrangling	17 144	1 165	93	858	14 294	5 788	15	184	12
 Statsmodels	Statistics	10 067	153	21	234	2 868	1 240	66	479	19
 TensorFlow	Deep learning	33 339	1 469	58	7 968	99 664	62 952	23	575	68
PYTORCH	Deep learning	11 306	635	16	816	15 512	3 483	18	707	24
 Keras	Deep learning	4 539	671	41	1 673	29 444	10 964	7	1111	44
dist-keras elephas spark-deep-learning	Distributed deep learning	1125	5	7	41	431	106	225	161	86
		170	13	5	97	913	189	13	34	70
Natural Language Toolkit	NLP	67	11	3	116	920	206	6	22	84
		13 041	236	24	467	6 405	1 804	55	543	27
spaCy	NLP	8 623	215	56	425	9 258	1 446	40	154	43
gensim	NLP	3 603	273	52	415	6 995	2 689	13	69	26
 Scrapy	Data scraping	6 625	281	81	1 723	27 277	6 469	24	82	97
<div> <div>Last reviewed: 13.02.2018</div> <div>Created by  ActiveWhizards</div> </div>										

Figura 4.5: Principales Bibliotecas para Python en Data Science

4.4. Herramientas Externas para desarrollo del proyecto

4.4.1. Multiple-Instance Learning Python Toolbox

Esta herramienta, llamada MILpy (Arrieta y Mera, 2017), contiene algoritmos para entrenar y evaluar clasificadores de Multiple-Instance Learning. Creada por Jose Arrieta (jmarrieta@unal.edu.co) y Carlos Mera (camerab@unal.edu.co) basándose en Matlab Toolbox for Multiple-Instance Learning, los autores permiten en su licencia el uso, copia y distribución para fines educativos y de investigación.

4.4.1.1. Justificación

Debido a la naturaleza del proyecto, encontrar herramientas para Multiple-Instance Learning se convirtió en una tarea que llevó más tiempo del que se creía, ya que no hay implementaciones en las bibliotecas oficiales para Python y los que se han atrevido a crear este tipo de software presentan bastantes dificultades en su uso o hay poca información.

Aunque esta herramienta no presenta todos los clasificadores o herramientas necesarias para poder decir que está completa, por lo menos sienta una base con la que poder trabajar implementando algunos de los clasificadores y medidas que existen para MIL.

4.4.1.2. Especificaciones

MILpy, desarrollado en 2016-2017 en Python V2 presenta una herramienta que convierte Dataset con extensión .mat provenientes de MatLab en estructuras de datos con las bibliotecas de Python para su uso con los algoritmos de clasificación de MIL desarrollados como funciones de entrenamiento y predicción para cada uno de estos junto con sus parámetros de ajuste. Nuestro proyecto ha sido desarrollado con las últimas versiones de las bibliotecas y con Python V3, con lo que se han tenido que re-formular los algoritmos puesto que en estos años, como se ha comentado anteriormente, Python está en auge en la Ciencia de Datos y evoluciona constantemente ofreciendo versiones mejoradas de sus funcionalidades.

No se tienen las especificaciones completas del software puesto que son proyectos personales sin soporte, así que ha sido a base de prueba y error comprender el funcionamiento completo de la herramienta, viendo que algunos fallos venían precisamente de la incompatibilidad de las versiones usadas para el proyecto y de MILpy.

4.4.1.3. Clasificadores

Aunque tiene implementados más clasificadores de los que se van a usar en el proyecto, no se usarán porque su funcionalidad no está completa y la herramienta dejó de actualizarse hace 2 años. Los que se han probado y funcionan para el proyecto se ven en la Tabla 4.1.

Tabla 4.1: Clasificadores implementados en MILpy.

Acrónimo	Nombre completo	Opciones	Variante	Referencia
MILBoost	Multiple Instance Learning Boost	Si	No	Viola, Platt, y Zhang
SimpleMiL	Simple Multiple Instance Learning	Si	Si	Pedregosa y cols.
MaxDD	Max Diverse Density	Si	No	Maron y Lozano-Pérez
EMDD	Expectation Maximization Maximum Diverse Density	Si	No	Zhang y A. Goldman
CKNN	Citation-KNN	Si	No	Goodwin y Garfield
BOW	Multiple Instance Learning Bag Of Words	Si	No	Carbonneau, Cheplygina, Granger, y Gagnon

Según los autores todas las implementaciones están basadas en MATLAB Udelft MIL toolbox (Tax y Cheplygina, 2016).

1. Multiple Instance Learning Boost (MILBoost): Usa funciones de costo provenientes de MIL combinadas con esquemas de AnyBoost. Se adapta el criterio de selección de características de MILBoost para optimizar el rendimiento de la cascada Viola-Jones (Viola y cols., 2006).
2. Simple Multiple Instance Learning (SimpleMiL): Este modelo clasifica y combina los resultados usando de base el clasificador C-Support Vector Classification (SVC) de scikit-learn (Pedregosa y cols., 2011).
3. Max Diverse Density (MaxDD): Es un modelo de intersección de las bolsas positivas menos la unión de las bolsas negativas. Para maximizar esta medida, podemos encontrar el punto de intersección y también el conjunto de pesos de características que conducen a la mejor intersección (Maron y Lozano-Pérez, 1998).
4. Expectation Maximization Maximum Diverse Density (EMDD): Este método busca estimaciones máximas de parámetros del clasificador, donde el modelo depende de variables latentes no observadas (Zhang y A. Goldman, 2002).
5. Citation-KNN (CKNN): Adapta K-Nearest Neighbor a the Multiple Instance Learning (Wang y Zucker, 2000) inspirándose en el concepto

citation” de (Goodwin y Garfield, 1980) basado en métodos de indexación de librerías.

6. Multiple Instance Learning Bag Of Words (MIL-BOW): Este método crea un diccionario de palabras representativas realizando k-medias para la agrupación en todas las instancias de entrenamiento. A continuación, las instancias serán representadas por la palabra más similar contenida en el diccionario (Carbonneau y cols., 2016).

4.4.2. Dataset

Aunque existen algunos repositorios con más Dataset que los que tienen en esta herramienta, nos centraremos en los que nos proporciona MILpy incluidos en la Tabla 4.2, ya que han sido comprobados por los propios autores con sus algoritmos, para no entrar en otro tipo de fallos que se escaparían al área de este proyecto.

Tabla 4.2: Características Generales de los Dataset de MILpy.

Dataset	Bolsas	+Bags	-Bags	Instancias	Atributos
Musk 1	92	47	45	476	166
Musk 2	102	39	63	6598	166
Mutagenesis easy	188	125	63	10486	7
Mutagenesis hard	42	13	29	2132	7
Tiger	200	100	100	1220	230
Fox	200	100	100	1320	230
Elephant	200	100	100	1391	230
Corel African	2000	100	1900	7947	9
Birds Brown Creeper	548	197	351	10232	38
Birds Winter Wren	548	109	439	10232	38
Gaussian-MI	20	10	10	133	2

A continuación se muestra una descripción sobre los dataset utilizados.

4.4.2.1. Musk

Este dataset describe si un conjunto de moléculas tiene un olor a almizcle suave o fuerte. El objetivo es aprender a predecir si las nuevas moléculas tendrán olor a almizcle o no.

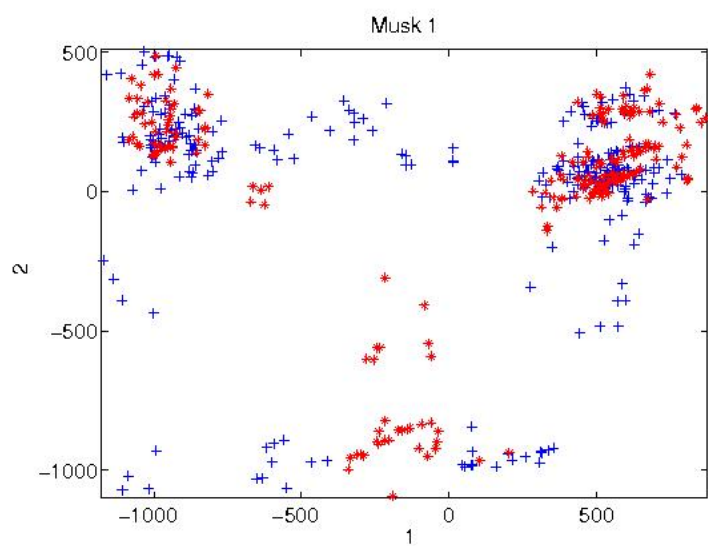


Figura 4.6: Gráfico de dispersión de Musk 1

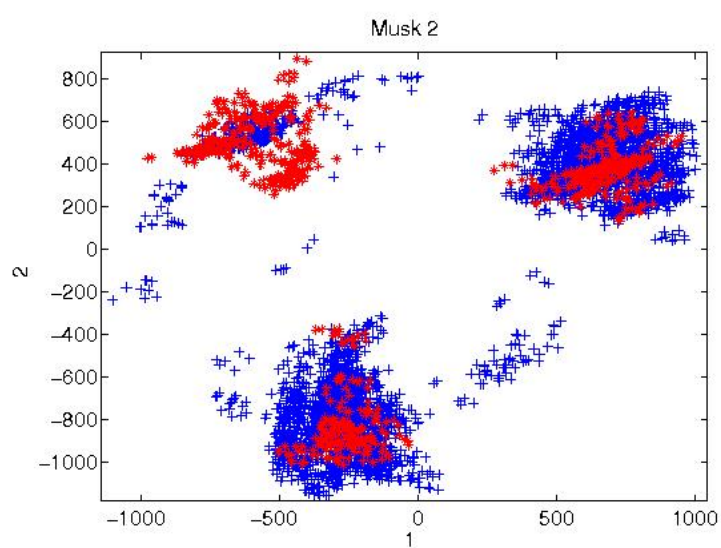


Figura 4.7: Gráfico de dispersión de Musk 2

4.4.2.2. Mutagenesis

Este dataset describe un conjunto de moléculas que pueden tener o no la mutación "Salmonella typhimurium" (Srinivasan, Muggleton, y King, 1995).

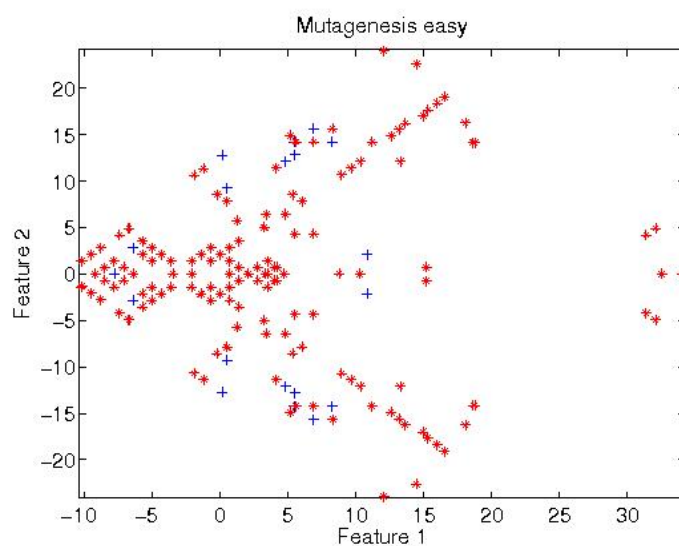


Figura 4.8: Gráfico de dispersión de Mutagenesis Easy

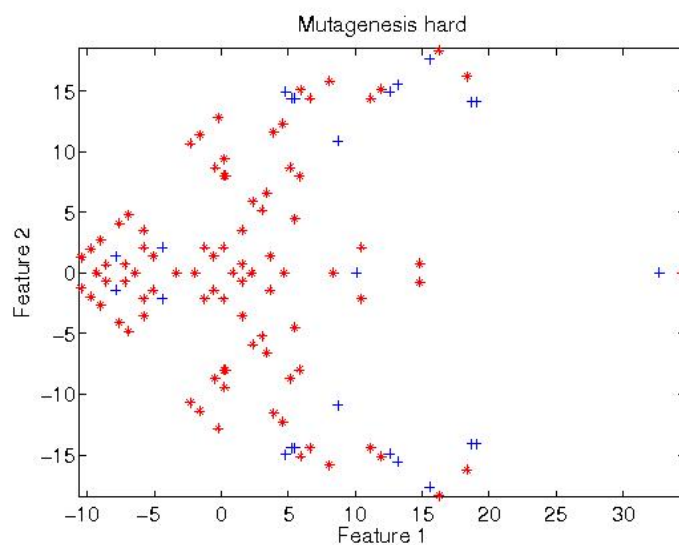


Figura 4.9: Gráfico de dispersión de Mutagenesis Hard

4.4.2.3. Tiger, Fox y Elephant

Cada uno de estos dataset está cualificado para identificar un animal (Tigre, Zorro o Elefante) en imágenes (Andrews, Hofmann, y Tsochantaridis, 2002)

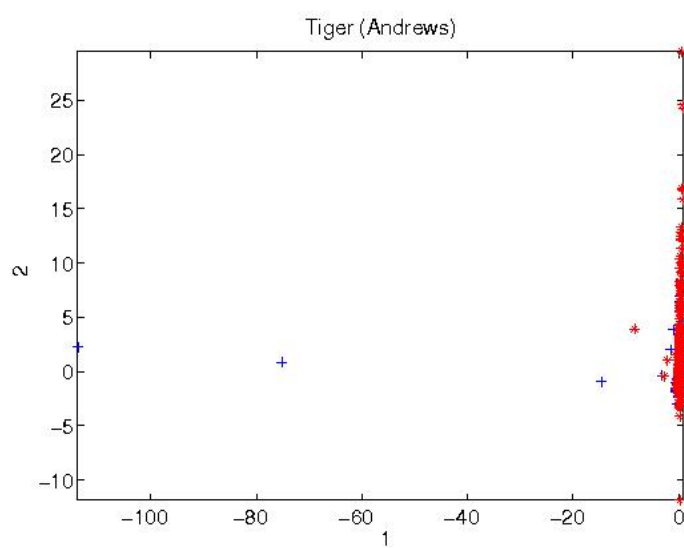


Figura 4.10: Gráfico de dispersión de Tiger

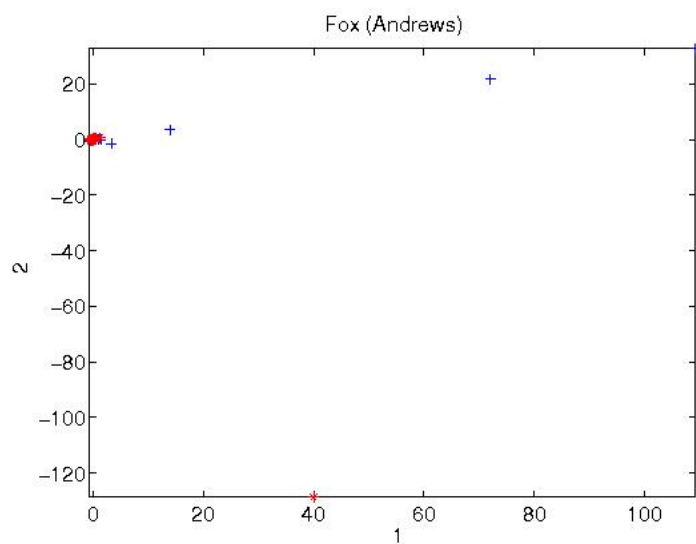


Figura 4.11: Gráfico de dispersión de Fox

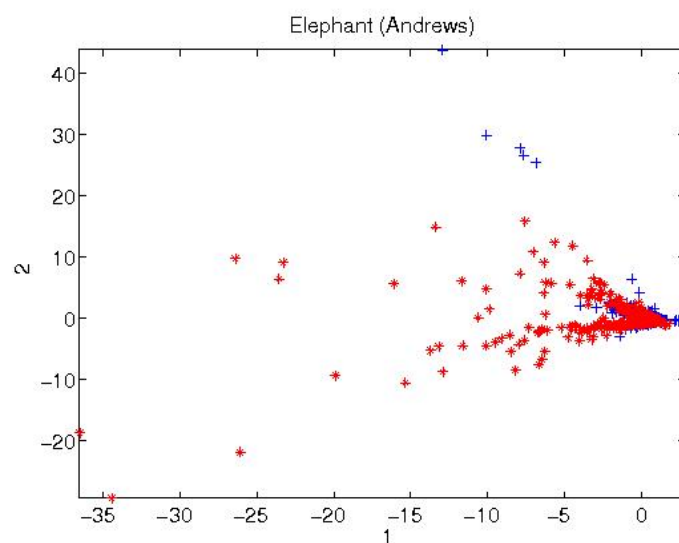


Figura 4.12: Gráfico de dispersión de Elephant

4.4.2.4. Corel African

Este dataset identifica en imágenes si aparece un escenario africano (Chen, Bi, y Wang, 2006).

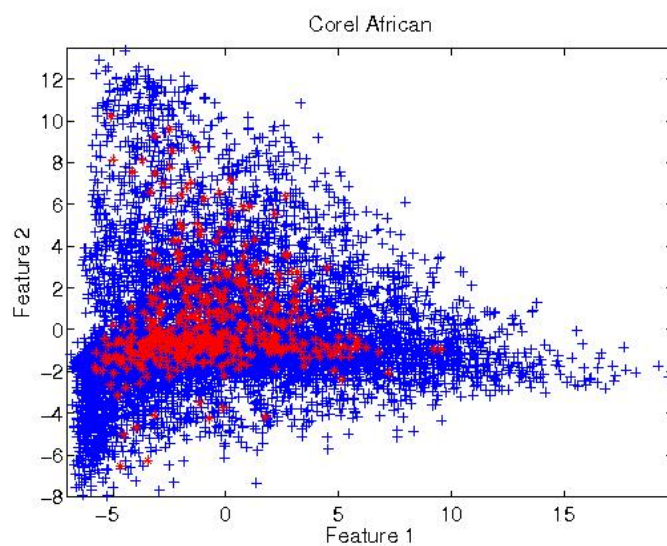


Figura 4.13: Gráfico de dispersión de Corel African

4.4.2.5. Birds

Estos dataset se entrenan para predecir si un tipo de ave está presente en un archivo de sonido. Una bolsa es el espectrograma de una grabación y una instancia es un segmento de ese espectrograma (Briggs y cols., 2012).

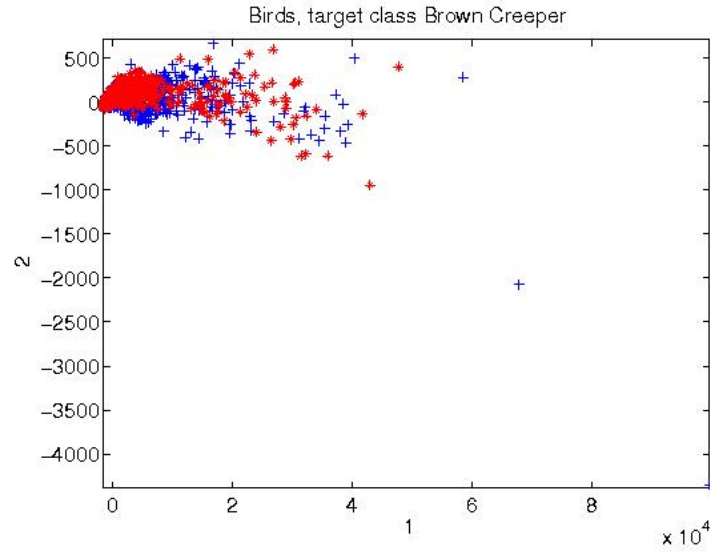


Figura 4.14: Gráfico de dispersión de Birds, clase Brown Creeper

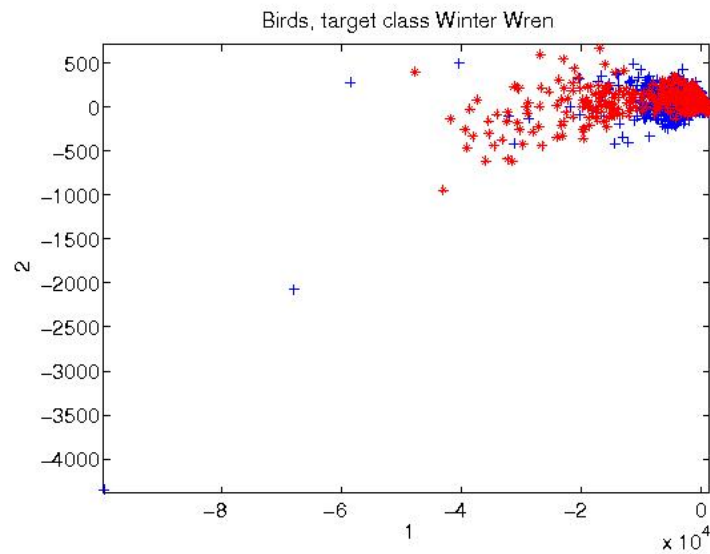


Figura 4.15: Gráfico de dispersión de Birds, clase Winter Wren

4.4.2.6. Gaussian

Es un dataset artificial, para las bolsas positivas, las instancias se extraen de la distribución gaussiana centrado alrededor de (7,1), y un conjunto aleatorio de instancias se extrae alrededor de (0,0).

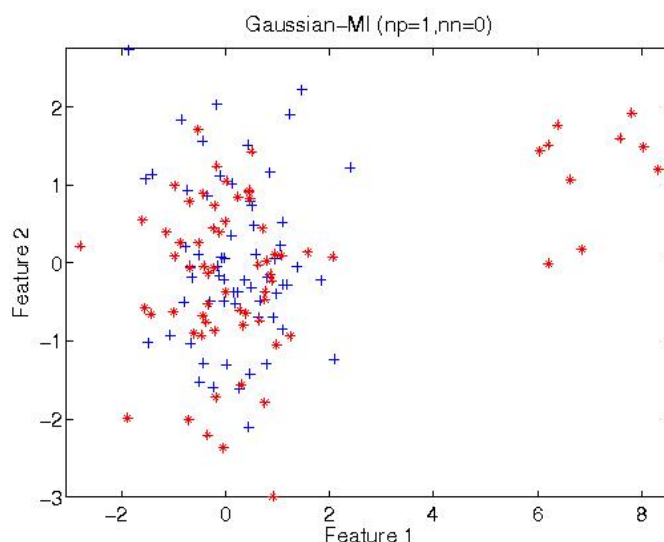


Figura 4.16: Gráfico de dispersión de Gaussian-MI

4.5. Herramientas para elaboración de la documentación

4.5.1. TeXstudio

TeXstudio es un IDE de LaTeX con licencia Pública General de GNU que proporciona un soporte moderno de escritura, como la corrección ortográfica interactiva, plegado de código y resaltado de sintaxis (van der Zander, 2009).

4.5.2. Apache OpenOffice Calc

Es una hoja de cálculo libre y de código abierto que forma parte de la suite ofimática Apache OpenOffice (Foundation, 2016) con licencia Pública General Reducida de GNU (LGPL). Es una aplicación utilizada en tareas financieras y contables, con fórmulas, gráficos y un lenguaje de programación. En nuestro caso la usaremos para la planificación de tareas y gestión de las tablas de información del marco de estudio.

Capítulo 5

pyMIL-BNF: Filtros de Ruido de Clase para MIL

5.1. Análisis del problema: Limpieza de Datos en MIL

En el campo de la ciencia de datos existen herramientas para limpieza de datos con ruido, como vemos en la Tabla 5.1.

Tabla 5.1: Esquemas de Filtrado de ruido para Simple-Instance

Filtro	Referencia
Ensemble Filter	Brodley y Friedl
Cross-Validated Committees Filter	Verbaeten y Assche
Iterative-Partitioning Filter	Khoshgoftaar y Rebours
Automatic Noise Remover	Zeng y Martinez
Classification Filter	Gamberger, Lavrac, y Groselj
Pairwise Attribute Noise Detection Algorithm Filter	Hulse, Khoshgoftaar, y Huang
Saturation Filter	Gamberger, Lavrac, y Dzroski

El problema es que la estructura de la información para Multiple-Instance Learning difiere con la utilizada para esas técnicas de limpieza de datos por lo que el primer problema es adaptar estos esquemas a la nueva taxonomía de bolsas de datos.

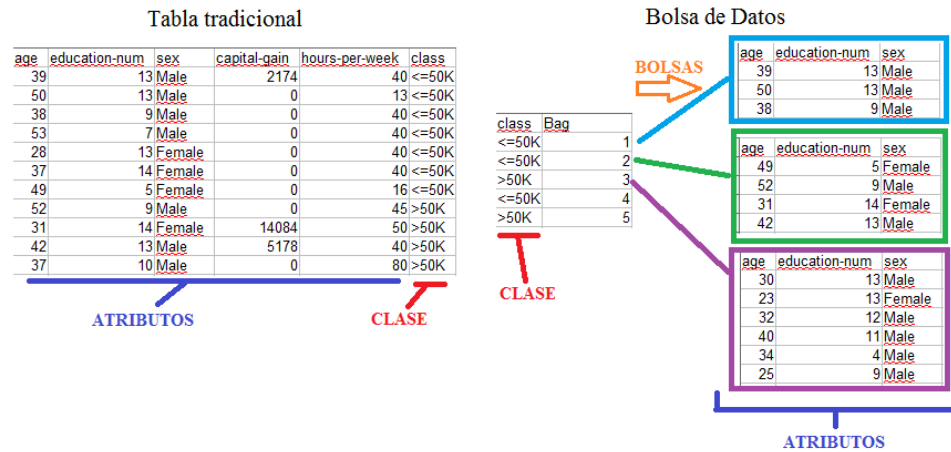


Figura 5.1: Esquema Bolsa de Datos

Además el desarrollo de filtros para MIL necesitará de clasificadores específicos que funcionen con este tipo de datos y que nos proporcionará la herramienta mencionada anteriormente como MILpy.

5.1.1. Arquitectura de flujo de datos

El patrón de arquitectura que usamos en el diseño de pyMIL-BNF se llama 'Pipe and Filter', nos viene impuesto por el modelo de Ciencia de datos que construye sistemas en los que se procesa o transforma un flujo de datos de entrada, esto sería un componente llamado 'Filter', y nos aporta una salida, este paso es llamado 'Pipe', que puede conectarse a otro u otros componentes 'Filter' o finalizar. En la Figura 5.2 vemos un ejemplo de los posibles flujos de pyMIL-BNF.

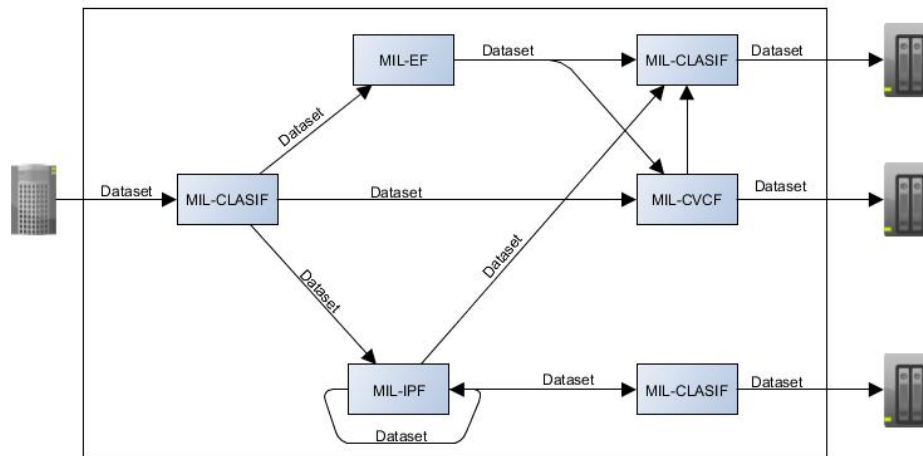


Figura 5.2: Pipe and Filter de pyMIL-BNF

En nuestro caso, los mensajes están conformados por los propios datasets, que normalmente se almacenan en fichero. Estos datasets contienen la información necesaria para realizar los diferentes pasos del proceso, ya sea limpieza o tareas de aprendizaje, extracción de conocimiento o visualización.

Así pues, el resto de análisis se realizarán teniendo en cuenta esta elección de arquitectura que asegura la compatibilidad de la biblioteca desarrollada en el presente TFG con otros paquetes de aprendizaje MIL.

5.1.2. Análisis de requisitos

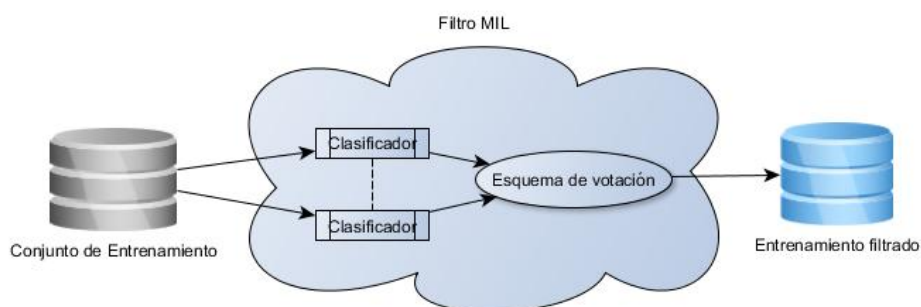


Figura 5.3: Esquema básico de Filtro MIL

Los componentes básicos explicados de forma general del conjunto de herramientas presentan las siguientes necesidades:

- **Conjuntos de datos:** Representan la colección de datos que se va a tratar y su arquitectura en forma de bolsa de datos está compuesta por:
 - Diferentes conjuntos de instancias y cada uno asociado a un identificador de bolsa.
 - Conjunto de etiquetas de clase asociadas a un Identificador de bolsa.
- **Filtros MIL:** Identificarán las etiquetas de las bolsas de datos que presenten ruido y se eliminarán, de forma que la entrada a nuestro algoritmo de filtrado estará compuesto por:
 - Conjunto de entrenamiento del dataset.
 - Esquema de votación.

Y la salida:

- Conjunto de entrenamiento limpio.
 - **Clasificador del filtro:** La característica fundamental del funcionamiento de los diferentes tipos de filtros, es que internamente usan diferentes clasificadores para identificar los clases ruidosas, cada uno aportando su propio resultado, en nuestro caso la entrada de un clasificador del filtro está compuesta por:
 - Conjunto de entrenamiento.
 - Parámetros de configuración del clasificador.
- Y nos devuelve:
- Tabla de bolsas etiquetadas como ruidosas.
- **Esquemas de votación:** Se usan para llegar a una única solución de entre todas las que aporta cada clasificador del filtro, así pues la

entrada está compuesta de:

- Una tabla de bolsas etiquetadas como ruidosas por cada Clasificador del filtro.

Y como salida nos proporciona:

- Conjunto de las bolsas etiquetadas como válidas.

5.1.3. Análisis de soluciones

El primer problema planteado sobre la necesidad de clasificadores para MIL se ha solucionado con el toolbox de MILpy que nos proporciona cierta cantidad de clasificadores específicos, las otras herramientas encontradas para Python eran bastante pobres o no cumplían con las necesidades del proyecto, en todo caso no se ha tenido un gran abanico de posibilidades.

El segundo problema era qué esquemas de limpieza de datos usar, así que tras revisar de entre todos los de la Tabla 5.1, se optó por adaptar los más usuales que son Ensemble Filter (EF), Cross-Validated Committees Filter (CVCF) e Iterative-Partitioning Filter (IPF).

En resumen:

- Biblioteca de Clasificadores:
 - MILpy.
- Esquemas base para filtros:
 - Ensemble Filter (EF).
 - Cross-Validated Committees Filter (CVCF).
 - Iterative-Partitioning Filter (IPF).

5.1.4. Solución propuesta

Debido a que el desarrollo del proyecto se basa en MILpy, una herramienta externa y sin mantenimiento desde su desarrollo en 2016, ha hecho falta ir paso a paso comprobando cada elemento que se iba a usar. Quitando las fases de aprendizaje que se señalaron en el Capítulo 2 sobre la planificación vamos a desarrollar las etapas sobre MILpy y los filtros MIL y las soluciones por las que se optaron para el avance del proyecto.

- Obtención de versión estable de MILpy:
 1. Elección: No hubo mucho problema en elegir esta herramienta porque las que se encontraban desarrolladas para Python eran solo algunos algoritmos de clasificación sueltos para problemas concretos, ciertamente esta era la única que nos proveía de una variedad más amplia de Clasificadores para MIL.
 2. Prueba: Su funcionamiento venía determinado por unos dataset provenientes de MatLab, así que el formato de lectura de los

clasificadores era específico para ese tipo de archivos.

3. Errores:

- Sin información de las versiones usadas de las librerías.
- Incompatibilidad con las nuevas versiones de bibliotecas científicas de Python.
- No tiene bloques de detección de errores y en ocasiones al almacenar gran cantidad de datos en memoria los clasificadores fallaban en la lectura de los dataset.
- En ocasiones se producían fallos en las predicciones porque las matrices de datos estaban corruptas.
- Algunos clasificadores no funcionaban completamente.
- Variables globales configuradas para el uso en entornos concretos del autor.

4. Soluciones:

- Se actualizó todo a Python v3, reescribiendo el código para llamar a las nuevas versiones de las bibliotecas y adaptando las funciones al nuevo formato, así como las variables de entorno. El esquema de un clasificador MILpy se ve en la Figura 5.4.
- Se añadieron bloques try-catch para detectar y corregir los errores en la ejecución.
- Los clasificadores que no funcionaban correctamente no se usarán.

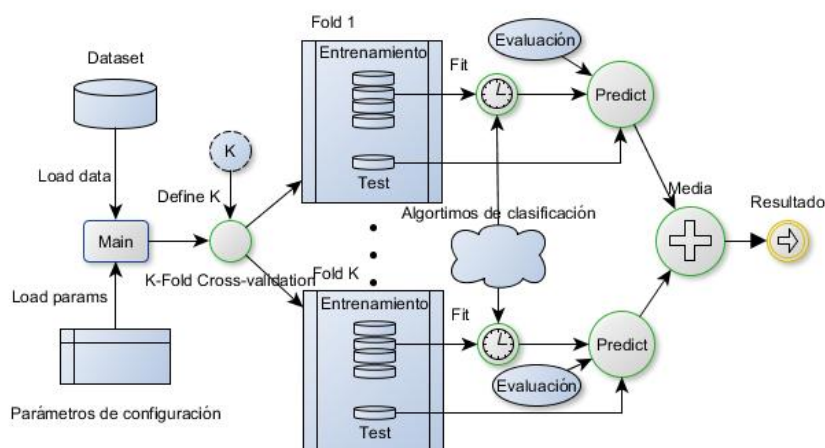


Figura 5.4: Esquema Avanzado Clasificador MIL

■ Desarrollo de filtros:

1. Estudio: Se analizaron los diferentes esquemas de limpieza de ruido (Tabla 5.1) y se escogieron 3 para implementar.
2. Pruebas: Se hicieron pruebas de código en Python para adaptar

la limpieza de ruido en datos MIL.

3. Implementaciones: Se escribió el código desde cero en Python v3.
4. Errores: Se detectó algún error de incompatibilidad de bibliotecas que fue solventado durante el desarrollo de los filtros.
5. Solución: Ya que una parte esencial del filtro es el clasificador usado y se solventaron los problemas en las etapas previas, se obtuvo rápidamente una versión final de los filtros. El esquema del resultado final se ve en la Figura 5.5.

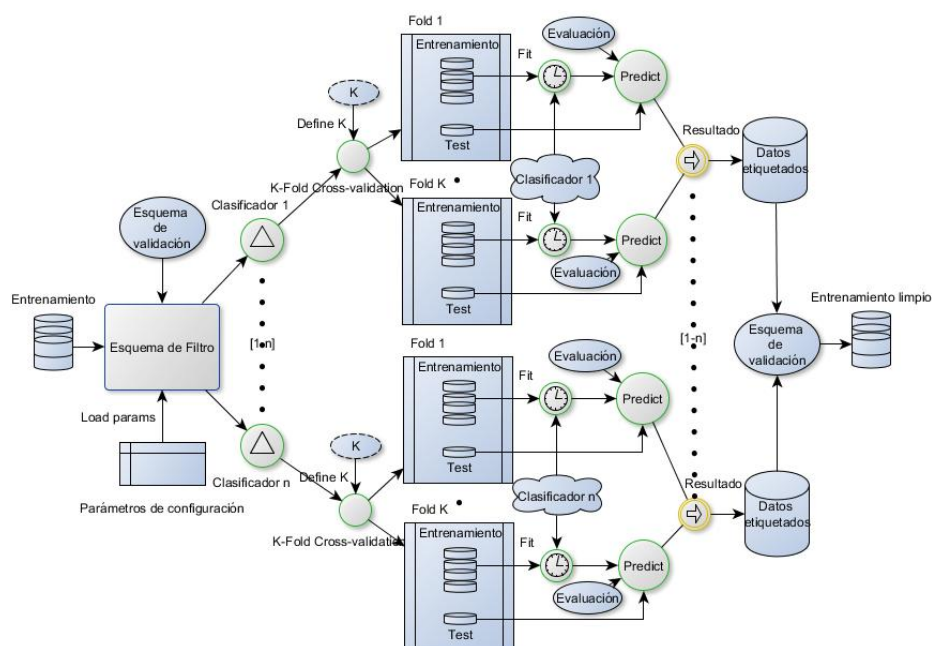


Figura 5.5: Esquema Avanzado Filtro MIL

5.2. Diseño General de pyMIL-BNF

En este apartado vamos a presentar como se ha desarrollado la biblioteca de técnicas, dando lugar a 3 esquemas de filtros:

1. Ensemble MIL-Filter: Técnica basada en Ensemble Filter (MIL-EF).
2. Cross-Validated Committees MIL-Filter: Técnica basada en Cross-Validated Committees Filter (MIL-CVCF).
3. Iterative-Partitioning MIL-Filter: Técnica basada en Iterative-Partitioning Filter (MIL-IPF).

5.2.1. Grupos funcionales

Vamos a listar lo que a grandes rasgos tendrán en común los diseños de los filtros MIL:

- FIL-GF.01 Administrar conjunto de datos.
- FIL-GF.02 Administrar validación cruzada.
- FIL-GF.03 Administrar inserción de ruido artificial.
- FIL-GF.04 Administrar Clasificador MIL.
- FIL-GF.05 Administrar esquema de votación.

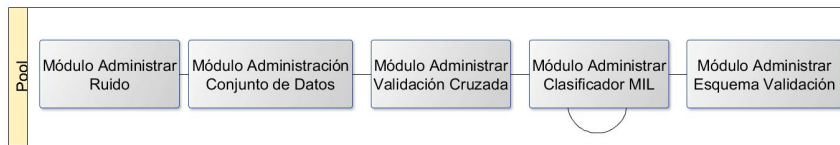


Figura 5.6: Esquema Grupos Funcionales

5.2.2. Requisitos funcionales

A partir de los grupos anteriormente descritos, identificaremos los requisitos asociados a cada uno. Véase Tabla 5.2.

Tabla 5.2: Identificación de requisitos funcionales para cada grupo funcional

Grupo Funcional	Requisito Funcional
FIL-GF.01	FIL-RF.01.01. Lectura conjunto de datos. FIL-RF.01.02. Salida conjunto de datos.
FIL-GF.02	FIL-RF.02.01. Seleccionar K. FIL-RF.02.02. Generar K particiones de datos.
FIL-GF.03	FIL-RF.03.01. Seleccionar porcentaje de ruido. FIL-RF.03.02. Lectura de conjunto de entrenamiento. FIL-RF.03.03. Modificación de conjunto de datos. FIL-RF.03.04. Salida de conjunto de datos modificado.
FIL-GF.04	FIL-RF.04.01. Selección de Clasificador MIL FIL-RF.04.02. Administrar validación cruzada. FIL-RF.04.03. Lectura conjunto de entrenamiento. FIL-RF.04.04. Lectura conjunto de test. FIL-RF.04.05. Entrenar clasificador. FIL-RF.04.06. Evaluar clasificador. FIL-RF.04.07. Generar conjunto de datos etiquetados.
FIL-GF.05	FIL-RF.05.01. Lectura de conjunto de datos etiquetados. FIL-RF.05.02. Seleccionar esquema de votación. FIL-RF.05.03. Modificar conjunto de datos. FIL-RF.05.04. Generar conjunto de datos filtrado.

5.2.3. Roles

El uso de la biblioteca solo presenta un rol o actor, Tabla 5.3.

Tabla 5.3: Roles en Casos de Uso

Rol	Descripción
Sistema	Un sistema/app será el encargado de ejecutar pyMIL-BNF

5.2.4. Modelo de Casos de Uso

Se utiliza una representación por colores para identificar los casos de uso asociados a cada grupo funcional, esta relación la vemos en la Tabla 5.2.4 y el diagrama en la Figura 5.7.

Grupo funcional	Color
Administrar conjunto de datos	Azul
Administrar validación cruzada	Verde
Administrar inserción de ruido artificial	Amarillo
Administrar Clasificador MIL	Naranja
Administrar esquema de votación	Rojo

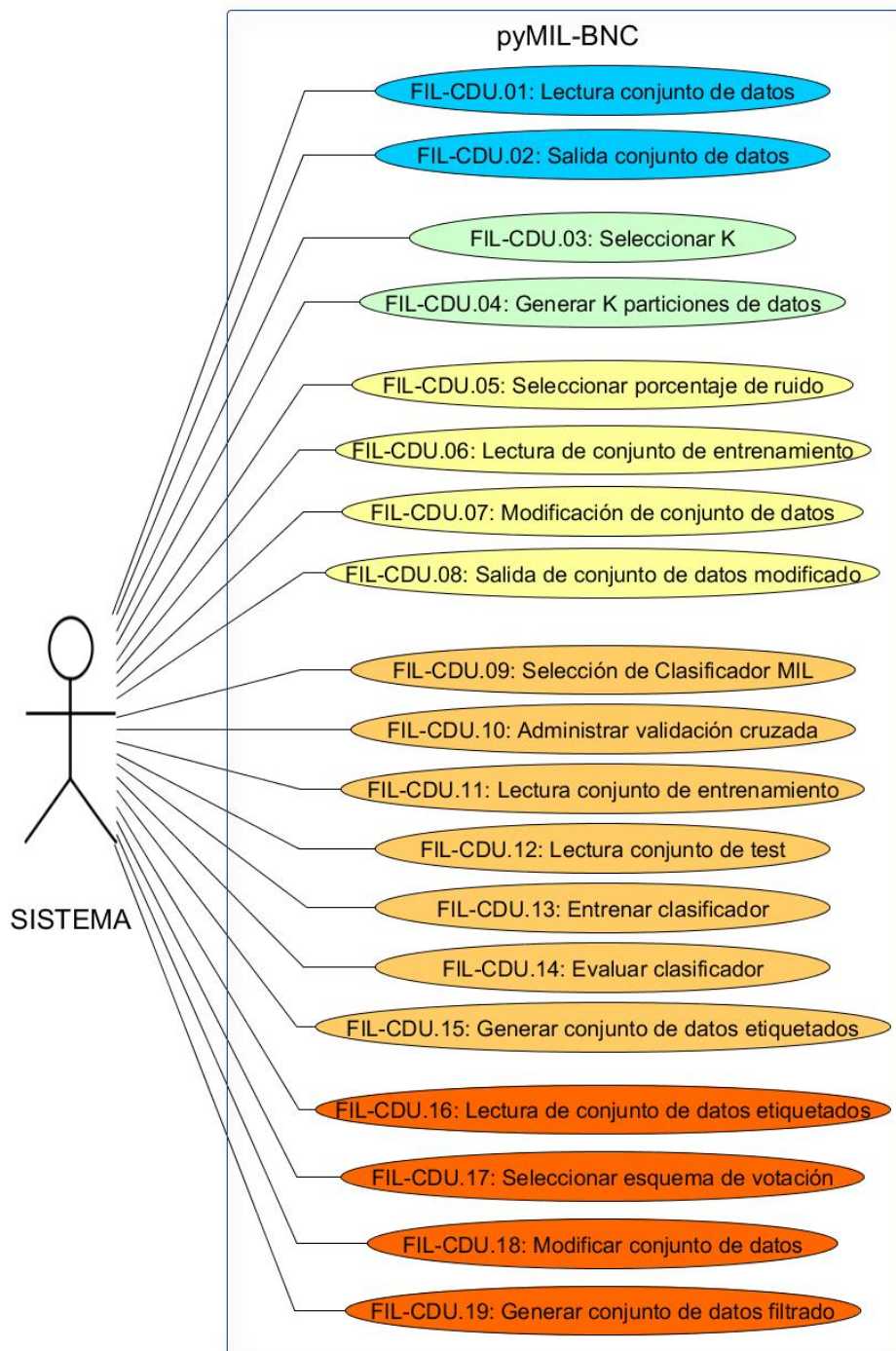


Figura 5.7: Diagrama de Casos de Uso Sistema-pyMIL-BNC

5.2.5. Dependencia de los Casos de Uso

Extraemos los casos de uso de los requisitos funcionales anteriormente descritos y mostraremos el nivel de dependencia que tienen entre ellos, para saber en que nivel ejecutar as diferentes tareas. Lo mostramos en la Tabla 5.4.

Tabla 5.4: Dependencia de los Casos de Uso

Caso de Uso	Nombre	RF	Prioridad
FIL-CDU.01	Lectura conjunto de datos.	FIL-RF.01.01.	1
FIL-CDU.02	Salida conjunto de datos.	FIL-RF.01.02.	1
FIL-CDU.03	Seleccionar K.	FIL-RF.02.01.	2
FIL-CDU.04	Generar K particiones de datos.	FIL-RF.02.02.	2
FIL-CDU.05	Seleccionar porcentaje de ruido.	FIL-RF.03.01.	3
FIL-CDU.06	Lectura de conjunto de entrenamiento.	FIL-RF.03.02.	3
FIL-CDU.07	Modificación de conjunto de datos.	FIL-RF.03.03.	3
FIL-CDU.08	Salida de conjunto de datos modificado.	FIL-RF.03.04.	3
FIL-CDU.09	Selección de Clasificador MIL.	FIL-RF.04.01.	4
FIL-CDU.10	Administrar validación cruzada.	FIL-RF.04.02.	4
FIL-CDU.11	Lectura conjunto de entrenamiento.	FIL-RF.04.03.	4
FIL-CDU.12	Lectura conjunto de test.	FIL-RF.04.04.	4
FIL-CDU.13	Entrenar clasificador.	FIL-RF.04.05.	4
FIL-CDU.14	Evaluar clasificador.	FIL-RF.04.06.	4
FIL-CDU.15	Generar conjunto de datos etiquetados.	FIL-RF.04.07.	4
FIL-CDU.16	Lectura de conjunto de datos etiquetados.	FIL-RF.05.01.	5
FIL-CDU.17	Seleccionar esquema de votación.	FIL-RF.05.02.	5
FIL-CDU.18	Modificar conjunto de datos.	FIL-RF.05.03.	5
FIL-CDU.19	Generar conjunto de datos filtrado.	FIL-RF.05.04.	5

La prioridad en los casos de uso viene dada por la dependencia entre las diferentes funcionalidades y la necesidad de hacerlo todo de forma secuencial.

5.2.6. Flujo general de los filtros

En la Figura 5.8 se presenta un diagrama con la relación de entre los casos de uso y el flujo de los procesos que comparten los diferentes esquemas de los filtros, más adelante se presentarán los diagramas individuales para ver las particularidades de cada uno de los modelos que se desarrollarán.

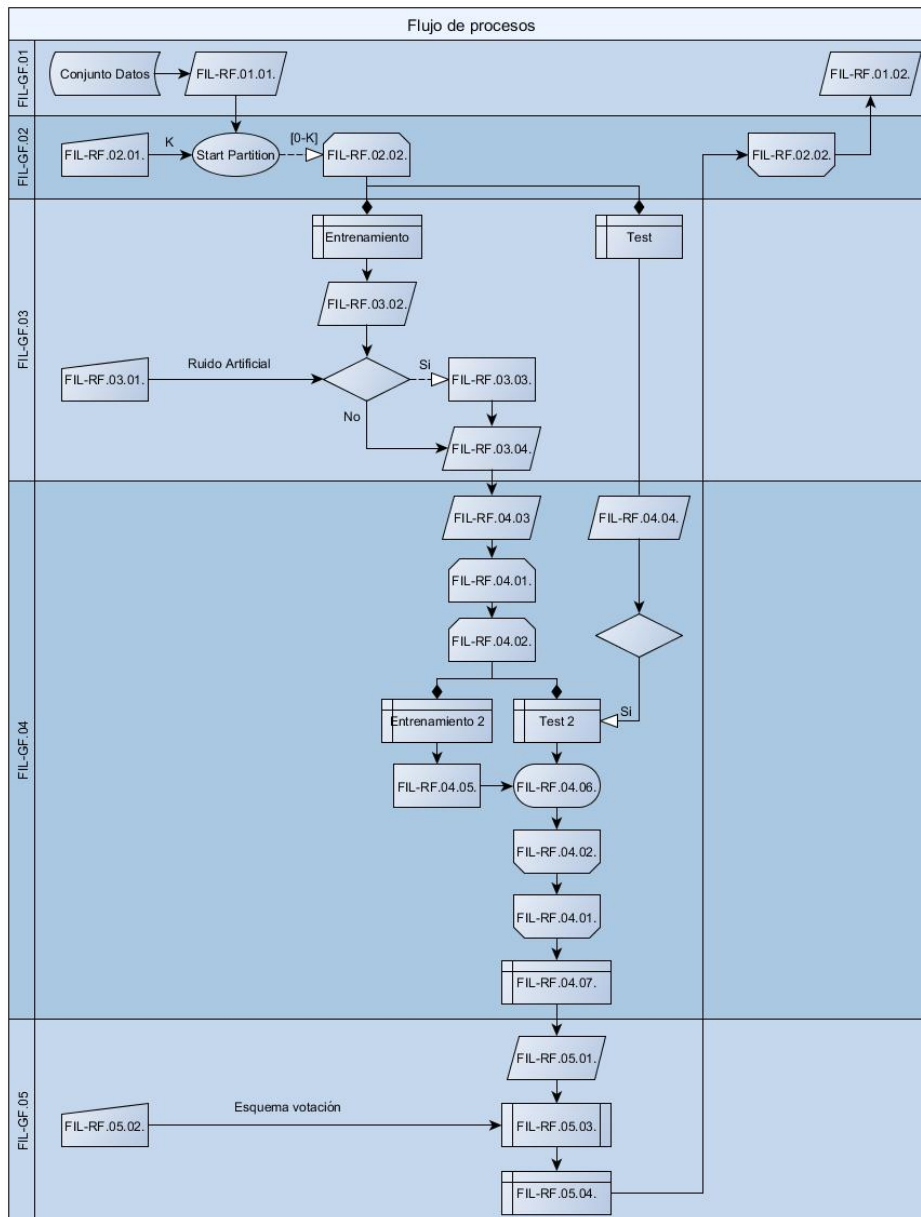


Figura 5.8: Diagrama de flujo de procesos general

5.2.7. Descripciones

5.2.7.1. MIL-EF

El dataset se divide en K particiones (Cross-Validation), para cada modelo de validación, K , un conjunto de 3 Clasificadores de los implementados en MILpy evalúa el conjunto de entrenamiento, finalmente para eliminar las etiquetas ruidosas se aplica el esquema de consenso o mayoría de votos a la combinación de las K validaciones, Figura 5.9.

5.2.7.2. MIL-CVCF

El dataset se divide en K particiones (Cross-Validation), para cada modelo de validación, K , un Clasificador de los implementados en MILpy evalúa el conjunto de entrenamiento, finalmente para eliminar las etiquetas ruidosas se aplica el esquema de consenso o mayoría de votos a la combinación de las K validaciones, Figura 5.10.

5.2.7.3. MIL-IPF

Al filtro se le definen un máximo de N -iteraciones, N se incrementará mientras N -error (variable definida al inicio) sea mayor al porcentaje de bolsas etiquetadas como ruido, si es menor se restablece N a 0, para cada N -iteración, el dataset se divide en K particiones (Cross-Validation), para cada modelo de validación, K , un Clasificador de los implementados en MILpy evalúa el conjunto de entrenamiento, finalmente para eliminar las etiquetas ruidosas se aplica el esquema de consenso o mayoría de votos a la combinación de las K validaciones de modo que para la siguiente iteración el dataset ya ha podido ser filtrado, Figura 5.11.

5.2.7.4. Ruido Artificial

Además se implementa la funcionalidad de insertar ruido artificial en los conjuntos de entrenamiento para comprobar la evaluación con diferentes porcentajes de ruido en las clases, se define desde 0 % (No modifica el dataset) hasta 30 % con saltos de 5 %, calculando según el tamaño del dataset a cuantas etiquetas de bolsas de datos equivale ese porcentaje y modificándolas aleatoriamente sin repetir la bolsa en todo caso a una clase diferente a la que tiene originalmente.

5.2.8.1. MIL-EF

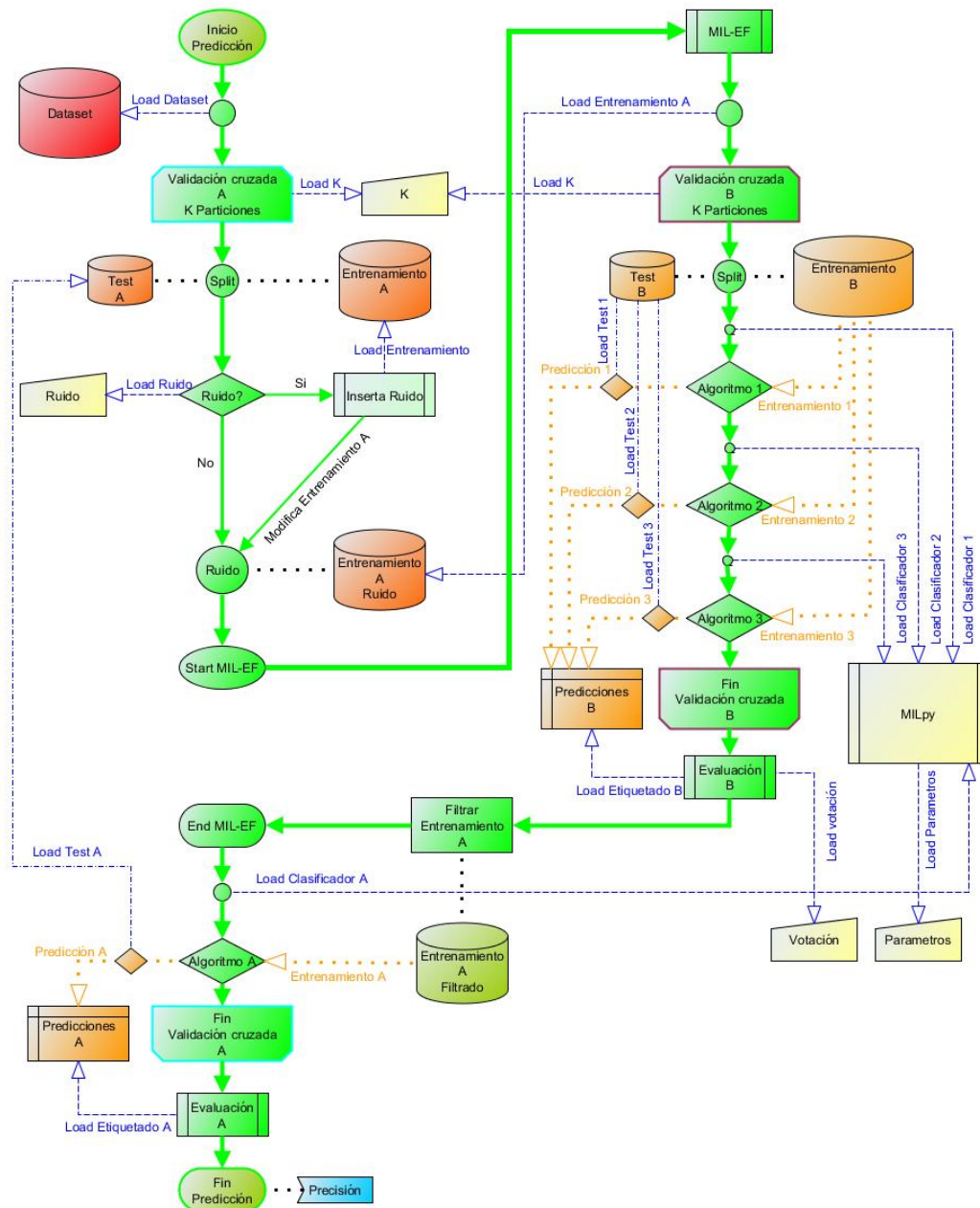


Figura 5.9: Diagrama flujo MIL-EF

5.2.8.3. MIL-IPF

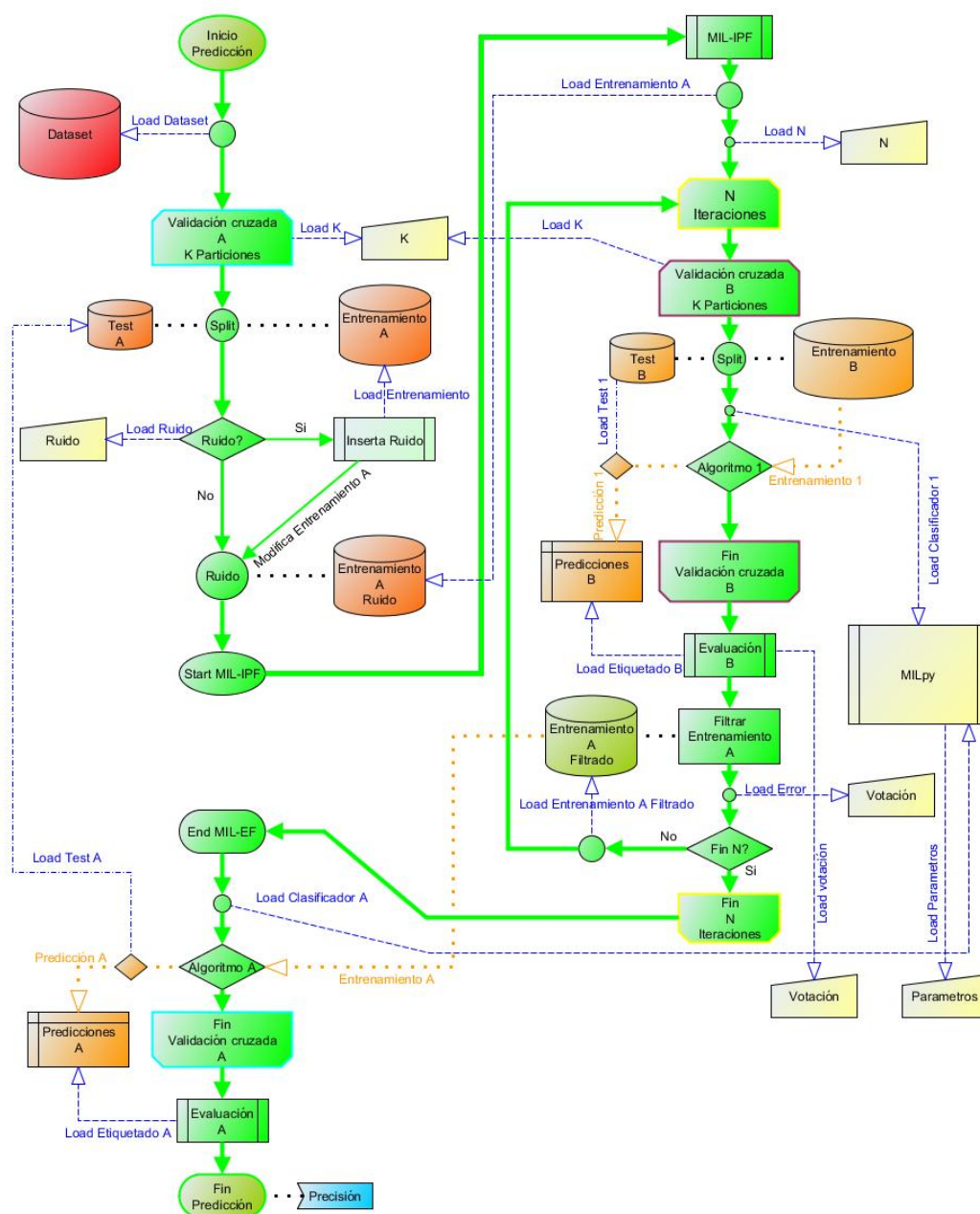


Figura 5.11: Diagrama flujo MIL-IPF

5.3. Implementación

Para darle un uso más confortable para posibles presentaciones o pruebas locales a la librería pyMIL-BNF se le ha preparado una interfaz gráfica con PyQt v5, multi-hilo para soportar los tiempos de ejecución y carga de datos, en la Figura 5.12 podemos ver esta interfaz. En el Anexo A se encuentra el manual de usuario para los filtros y para la aplicación y en el Anexo B la implementación de código en Python.

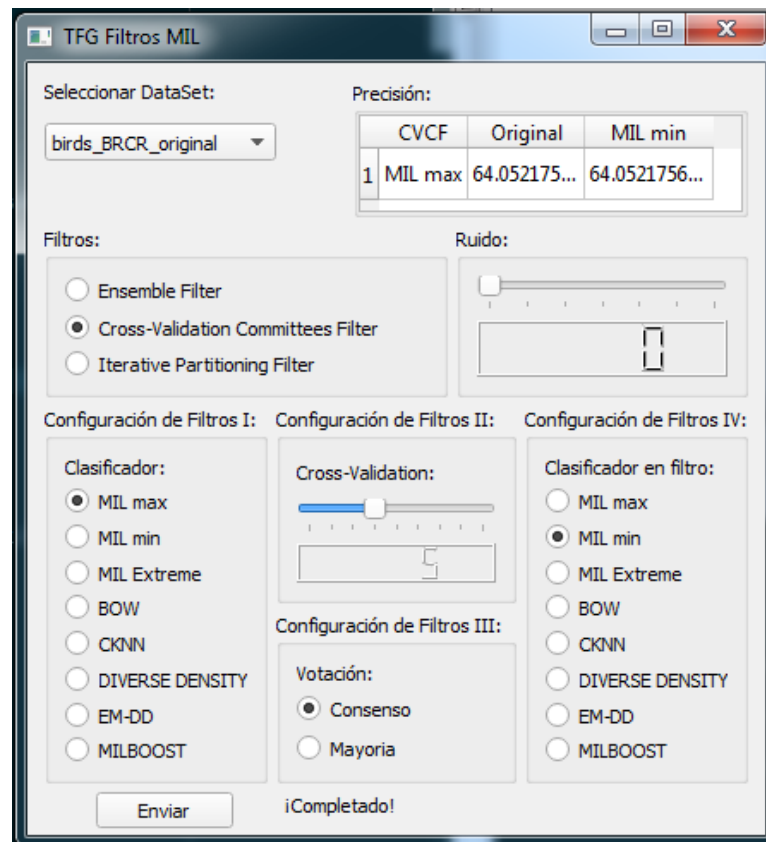


Figura 5.12: Ejemplo de la Interfaz Gráfica para pyMIL-BNF

Capítulo 6

Estudio Experimental

6.1. Marco Experimental

6.1.1. Clasificadores MILpy

La forma de evaluar el impacto en los dataset de los filtros de pyMIL-BNF es a través de las métricas que nos proporcionan los Clasificadores MILpy, en nuestro caso la Exactitud (Accuracy) vista en Capitulo 3.2.3, y comparando los modelos de predicción que construyen estos.

Estos son los clasificadores para Multiple-Instance Learning proporcionados por MILpy que usaremos para obtener la evaluación después del filtrado del dataset:

- Multiple Instance Learning Boost.
- Simple Multiple Instance Learning. Versiones:
 - SimpleMIL min: Usa los índices de los valores mínimos de la bolsa.
 - SimpleMIL max: Usa los índices de los valores máximos de la bolsa.
 - SimpleMIL extreme: Usa los índices de los valores máximos y mínimos de la bolsa.
- Max Diverse Density.
- Expectation Maximization Maximum Diverse Density.
- Citation-KNN.
- Multiple Instance Learning Bag Of Words.

6.1.1.1. Parámetros

Los parámetros de configuración de los clasificadores de MILpy que vemos en la Tabla 6.1, se ha establecido según la más óptima que proporciona los autores en sus casos de estudio (Arrieta y Mera, 2017).

Tabla 6.1: Parámetros de configuración para Clasificadores MIL

Clasificador	type				
<i>SimpleMiL-Max</i>	max				
<i>SimpleMiL-Min</i>	min				
<i>SimpleMiL-Extreme</i>	extreme				
<i>BOW</i>	k	covar_type	n_iter		
	90	diag	20		
	references	citers			
<i>CKNN</i>	3	5			
<i>maxDD</i>	spoints	epochs	tol	frac	alf
	10	[4,4]	1	[1e-5,1e-5,1e-7,1e-7]	-
	<i>EMDD</i>	10	[3,3]	1	[1e-4,1e-4,1e-4,1e-4]
<i>MILBoost</i>	errtol	T			
	1e-15	100			

6.1.1.2. Medida de Evaluación

Los clasificadores de MILpy muestran la evaluación con las medidas de:

- Exactitud.
- Curva ROC.

Las tablas del estudio muestran el porcentaje de Exactitud.

6.1.2. Filtros pyMIL-BNF

Los esquemas de filtros utilizados son:

1. Ensemble MIL-Filter: Técnica basada en Ensemble Filter (MIL-EF).
2. Cross-Validated Committees MIL-Filter: Técnica basada en Cross-Validated Committees Filter (MIL-CVCF).
3. Iterative-Partitioning MIL-Filter: Técnica basada en Iterative-Partitioning Filter (MIL-IPF).

6.1.2.1. Parámetros

En esta primera Tabla 6.2 vemos los parámetros de configuración para los filtros nombrados anteriormente usados en la ejecución del estudio.

Tabla 6.2: Parámetros de configuración para Filtros pyMIL-BNF

Filtro	K-Folds	Ruido	Votación
MIL-EF	5	[0,5,10,15,20,25,30]	[consenso, maxVotos]
MIL-CVCF	5	[0,5,10,15,20,25,30]	[consenso, maxVotos]
MIL-IPF	5	[0,5,10,15,20,25,30]	[consenso, maxVotos]

La siguiente Tabla 6.3 muestra qué Clasificadores ha usado cada filtro para los que ha generado conjuntos filtrados de los dataset.

Tabla 6.3: Clasificadores usados por cada filtro de pyMIL-BNF

Clasificador	MIL-EF		MIL-CVCF	MIL-IPF
	Grupo 1	Grupo 2		
SimpleMiL-Max	No	Si	Si	Si
SimpleMiL-Min	No	No	Si	Si
SimpleMiL-Extreme	No	Si	Si	Si
BOW	Si	No	Si	Si
CKNN	Si	No	Si	Si
maxDD	No	Si	Si	Si
EMDD	Si	No	Si	Si
MILBoost	No	No	Si	Si

6.1.3. Dataset

Los dataset no requieren de configuración previa, la siguiente Tabla 6.4 muestra cuales de los mostrados en la sección '4.4.2 Dataset' de este documento se han usado y las características de cada uno de ellos.

Tabla 6.4: Dataset usados en estudio por pyMIL-BNF

Dataset	Bolsas	+Bags	-Bags	Instancias	Atributos
Musk 1	92	47	45	476	166
Mutagenesis hard	42	13	29	2132	7
Tiger	200	100	100	1220	230
Fox	200	100	100	1320	230
Elephant	200	100	100	1391	230

6.2. Resultados Experimentales

Debido a la gran extensión que presentan todos los resultados, las tablas completas se encuentran en el Anexo C y las tablas detalladas en el Anexo D donde desgranamos la información a varios niveles para analizarla combinando los clasificadores con el nivel de ruido y los esquemas de votación señalando los mejores y peores modelos.

En la siguiente sección se presentan los análisis de las tablas de forma simplificada en la que encontramos las 'Tablas Resumen', que son una visión general del marco de estudio, donde comparamos los esquemas de votación y el ruido imputado, para cada filtro de pyMIL-BNF sobre cada uno de los clasificadores.

6.2.1. Tablas Resumen y análisis

A continuación de cada tabla expondremos un análisis individualizado de cada clasificador en el que trataremos la degradación de la medida de Exactitud de forma general, mejor filtro pyMIL-BNF por esquema de votación, el mejor filtro por ruido y en caso necesario alguna anotación suplementaria.

6.2.1.1. Clasificador SimpleMIL-max

Tabla 6.5: Tabla Resumen Clasificador SimpleMIL-max

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
<i>SIN ESQUEMA DE VOTACIÓN</i>							
SMIL max	68,93	65,43	65,05	61,47	59,97	57,99	58,84
<i>POR CONSENSO</i>							
MIL-EF	75,03	73,05	70,12	69,57	70,88	67,82	60,79
MIL-CVCF	73,22	72,04	72,22	70,55	67,42	64,22	64,93
MIL-IPF	71,77	70,13	70,37	68,64	67,42	64,14	66,33
<i>POR MAX VOTOS</i>							
MIL-EF	73,00	70,99	69,49	69,06	66,31	66,38	63,83
MIL-CVCF	65,31	65,07	63,50	63,25	62,07	59,86	56,49
MIL-IPF	38,64	25,72	26,21	12,58	12,32	10,46	24,14

Análisis para SimpleMIL-max

- La degradación en la métrica utilizada de Exactitud se ve reducida a medida que en los conjuntos de entrenamiento se imputa mayor porcentaje de ruido en las clases en todo caso con alguna excepción.
- El mejor Filtro MIL para esquema de votación por Consenso es MIL-EF, que aunque no sea el mayor en todos los casos es el más estable en las predicciones.
- El mejor Filtro MIL para esquema de votación por Max Votos en MIL-EF por unanimidad.
- El mejor Filtro MIL para:
 - Ruido 0 % es MIL-EF con esquema de votación por consenso.
 - Ruido 5 % es MIL-EF con esquema de votación por consenso.
 - Ruido 10 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 15 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 20 % es MIL-EF con esquema de votación por consenso.
 - Ruido 25 % es MIL-EF con esquema de votación por consenso.
 - Ruido 30 % es MIL-IPF con esquema de votación por consenso.

- De forma general se observa que el mejor filtro para SimpleMIL-max es MIL-EF con esquema de votación por consenso, que mejora la predicción frente a no usar ninguno en todos los casos.

6.2.1.2. Clasificador SimpleMIL-min

Tabla 6.6: Tabla Resumen Clasificador SimpleMIL-min

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
<i>SIN ESQUEMA DE VOTACIÓN</i>							
SMIL min	67,83	64,84	63,16	59,00	56,75	54,41	54,30
<i>POR CONSENSO</i>							
MIL-EF	73,89	71,99	68,61	67,56	68,81	65,36	58,96
MIL-CVCF	71,49	70,05	66,90	66,33	63,53	60,85	57,18
MIL-IPF	70,61	71,59	67,66	67,06	63,26	59,55	58,35
<i>POR MAX VOTOS</i>							
MIL-EF	72,69	70,21	68,96	67,96	64,13	65,25	62,02
MIL-CVCF	64,56	62,10	60,95	61,86	56,12	57,00	55,92
MIL-IPF	37,70	26,76	26,41	11,26	12,67	10,28	23,54

Análisis para SimpleMIL-min

- La degradación en la métrica utilizada de Exactitud se ve reducida a medida que en los conjuntos de entrenamiento se imputa mayor porcentaje de ruido en las clases en todo caso con alguna excepción.
- El mejor Filtro MIL para esquema de votación por Consenso es MIL-EF por unanimidad.
- El mejor Filtro MIL para esquema de votación por Max Votos en MIL-EF por unanimidad.
- El mejor Filtro MIL para:
 - Ruido 0 % es MIL-EF con esquema de votación por consenso.
 - Ruido 5 % es MIL-EF con esquema de votación por consenso.
 - Ruido 10 % es MIL-EF con esquema de votación por consenso.
 - Ruido 15 % es MIL-EF con esquema de votación por Max Votos.
 - Ruido 20 % es MIL-EF con esquema de votación por consenso.
 - Ruido 25 % es MIL-EF con esquema de votación por consenso.
 - Ruido 30 % es MIL-EF con esquema de votación por Max Votos.
- De forma general se observa que el mejor filtro para SimpleMIL-min es MIL-EF con esquema de votación por consenso, que mejora la predicción frente a no usar ninguno en todos los casos.

6.2.1.3. Clasificador SimpleMIL-extreme

Tabla 6.7: Tabla Resumen Clasificador SimpleMIL-extreme

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
<i>SIN ESQUEMA DE VOTACIÓN</i>							
SMIL ext	70,69	66,99	66,19	61,99	60,96	58,65	57,52
<i>POR CONSENSO</i>							
MIL-EF	75,10	72,76	70,19	69,05	69,74	66,50	59,90
MIL-CVCF	75,63	74,12	73,32	72,51	72,88	69,03	62,51
MIL-IPF	73,73	72,48	73,37	69,73	69,40	66,85	62,23
<i>POR MAX VOTOS</i>							
MIL-EF	73,33	71,40	69,90	69,10	65,23	66,15	62,81
MIL-CVCF	67,51	66,34	63,63	64,16	62,29	57,30	58,57
MIL-IPF	39,95	27,91	26,90	12,66	12,65	10,58	23,79

Análisis para SimpleMIL-extreme

- La degradación en la métrica utilizada de Exactitud se ve reducida a medida que en los conjuntos de entrenamiento se imputa mayor porcentaje de ruido en las clases en todo caso.
- El mejor Filtro MIL para esquema de votación por Consenso es MIL-CVCF por unanimidad, la excepción tiene una diferencia imperceptible.
- El mejor Filtro MIL para esquema de votación por Max Votos en MIL-EF por unanimidad.
- El mejor Filtro MIL para:
 - Ruido 0 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 5 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 10 % es MIL-IPF con esquema de votación por consenso.
 - Ruido 15 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 20 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 25 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 30 % es MIL-EF con esquema de votación por Max Votos.
- De forma general se observa que el mejor filtro para SimpleMIL-extreme es MIL-CVCF con esquema de votación por consenso, que mejora la predicción frente a no usar ninguno en todos los casos.

6.2.1.4. Clasificador BOW

Tabla 6.8: Tabla Resumen Clasificador BOW

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
<i>SIN ESQUEMA DE VOTACIÓN</i>							
BOW	65,45	62,78	61,76	58,94	56,11	55,15	56,47
<i>POR CONSENSO</i>							
MIL-EF	73,98	71,69	69,47	68,41	68,42	65,72	59,88
MIL-CVCF	69,51	68,12	67,99	65,96	62,95	63,26	62,67
MIL-IPF	68,84	67,69	67,66	67,58	64,26	63,07	61,69
<i>POR MAX VOTOS</i>							
MIL-EF	72,19	70,31	68,96	68,09	64,98	65,69	62,73
MIL-CVCF	61,11	59,40	58,61	59,39	57,80	58,73	56,29
MIL-IPF	35,24	23,89	24,05	11,93	10,86	10,38	22,46

Análisis para BOW

- La degradación en la métrica utilizada de Exactitud se ve reducida a medida que en los conjuntos de entrenamiento se imputa mayor porcentaje de ruido en las clases en todo caso con alguna excepción.
- El mejor Filtro MIL para esquema de votación por Consenso es MIL-EF casi por unanimidad, fallando en el porcentaje más alto de ruido.
- El mejor Filtro MIL para esquema de votación por Max Votos en MIL-EF por unanimidad.
- El mejor Filtro MIL para:
 - Ruido 0 % es MIL-EF con esquema de votación por consenso.
 - Ruido 5 % es MIL-EF con esquema de votación por consenso.
 - Ruido 10 % es MIL-EF con esquema de votación por consenso.
 - Ruido 15 % es MIL-EF con esquema de votación por consenso.
 - Ruido 20 % es MIL-EF con esquema de votación por consenso.
 - Ruido 25 % es MIL-EF con esquema de votación por consenso.
 - Ruido 30 % es MIL-EF con esquema de votación por Max Votos.
- De forma unificada se observa que el mejor filtro para BOW es MIL-EF con esquema de votación por consenso, que mejora la predicción frente a no usar ninguno en todos los casos.

6.2.1.5. Clasificador CKNN

Tabla 6.9: Tabla Resumen Clasificador CKNN

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
<i>SIN ESQUEMA DE VOTACIÓN</i>							
CKNN	69,97	67,04	64,69	60,28	58,53	56,76	56,30
<i>POR CONSENSO</i>							
MIL-EF	74,47	72,57	70,28	68,76	68,93	66,31	60,78
MIL-CVCF	75,26	74,68	68,86	69,86	66,07	66,98	59,47
MIL-IPF	75,05	74,61	73,00	66,83	67,93	65,81	62,90
<i>POR MAX VOTOS</i>							
MIL-EF	73,07	71,19	69,95	69,14	66,22	66,60	63,57
MIL-CVCF	76,04	73,49	72,47	70,52	67,89	66,67	65,71
MIL-IPF	41,36	29,46	28,36	13,30	13,69	10,34	25,27

Análisis para CKNN

- La degradación en la métrica utilizada de Exactitud se ve reducida a medida que en los conjuntos de entrenamiento se imputa mayor porcentaje de ruido en las clases en todo caso con alguna excepción.
- El mejor Filtro MIL para esquema de votación por Consenso por mayoría de casos en que mejora es MIL-CVCF, en los casos en los que no es el mejor la diferencia es muy poca.
- El mejor Filtro MIL para esquema de votación por Max Votos en MIL-CVCF por unanimidad.
- El mejor Filtro MIL para:
 - Ruido 0 % es MIL-CVCF con esquema de votación por Max Votos.
 - Ruido 5 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 10 % es MIL-IPF con esquema de votación por consenso.
 - Ruido 15 % es MIL-CVCF con esquema de votación por Max Votos.
 - Ruido 20 % es MIL-EF con esquema de votación por consenso.
 - Ruido 25 % es MIL-CVCF con esquema de votación por consenso.
 - Ruido 30 % es MIL-CVCF con esquema de votación por Max Votos.
- De forma general se observa que el mejor filtro para CKNN es MIL-EF con esquema de votación por consenso, y que en los casos donde otros mejoran también es MIL-EF con esquema por Max Votos, además en todos los casos mejora la predicción frente a no usar ninguno.

6.2.1.6. Clasificador maxDD

Tabla 6.10: Tabla Resumen Clasificador maxDD

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
<i>SIN ESQUEMA DE VOTACIÓN</i>							
maxDD	62,37	60,15	59,22	55,66	54,88	53,66	55,05
<i>POR CONSENSO</i>							
MIL-EF	73,04	71,27	69,31	67,93	68,05	65,22	60,28
MIL-CVCF	63,43	63,22	62,39	59,97	61,54	56,85	58,03
MIL-IPF	62,92	61,46	61,87	61,63	61,26	58,03	58,95
<i>POR MAX VOTOS</i>							
MIL-EF	71,99	70,19	68,79	68,41	65,64	65,87	62,96
MIL-CVCF	59,99	59,27	58,52	57,95	56,88	55,11	55,12
MIL-IPF	36,93	25,54	24,24	11,60	11,57	10,73	22,27

Análisis para maxDD

- La degradación en la métrica utilizada de Exactitud se ve reducida a medida que en los conjuntos de entrenamiento se imputa mayor porcentaje de ruido en las clases en todo caso con alguna excepción.
- El mejor Filtro MIL para esquema de votación por Consenso es MIL-EF por unanimidad.
- El mejor Filtro MIL para esquema de votación por Max Votos en MIL-EF por unanimidad.
- El mejor Filtro MIL para:
 - Ruido 0 % es MIL-EF con esquema de votación por consenso.
 - Ruido 5 % es MIL-EF con esquema de votación por consenso.
 - Ruido 10 % es MIL-EF con esquema de votación por consenso.
 - Ruido 15 % es MIL-EF con esquema de votación por Max Votos.
 - Ruido 20 % es MIL-EF con esquema de votación por consenso.
 - Ruido 25 % es MIL-EF con esquema de votación por Max Votos.
 - Ruido 30 % es MIL-EF con esquema de votación por Max Votos.
- De forma unificada se observa que el mejor filtro para maxDD es MIL-EF y bueno con ambos esquemas de votación ya que las diferencias en sus % son muy reducidas, además se mejora la predicción frente a no usar ninguno en todos los casos.

6.2.1.7. Clasificador EMDD

Tabla 6.11: Tabla Resumen Clasificador EMDD

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
<i>SIN ESQUEMA DE VOTACIÓN</i>							
EMDD	67,77	64,40	63,48	58,75	58,63	55,69	57,21
<i>POR CONSENSO</i>							
MIL-EF	73,14	71,44	69,46	68,30	67,99	65,26	60,46
MIL-CVCF	70,16	70,29	69,11	67,23	65,54	65,14	60,53
MIL-IPF	70,07	68,78	67,80	65,87	66,42	63,15	61,58
<i>POR MAX VOTOS</i>							
MIL-EF	72,33	70,50	69,16	68,93	66,33	66,36	63,33
MIL-CVCF	65,60	63,41	61,45	63,07	61,08	59,66	57,72
MIL-IPF	39,81	26,53	26,91	12,05	13,00	11,00	23,91

Análisis para EMDD

- La degradación en la métrica utilizada de Exactitud se ve reducida a medida que en los conjuntos de entrenamiento se imputa mayor porcentaje de ruido en las clases en todo caso con alguna excepción.
- El mejor Filtro MIL para esquema de votación por Consenso es MIL-EF casi por unanimidad, fallando en el porcentaje más alto de ruido.
- El mejor Filtro MIL para esquema de votación por Max Votos en MIL-EF por unanimidad.
- El mejor Filtro MIL para:
 - Ruido 0 % es MIL-EF con esquema de votación por consenso.
 - Ruido 5 % es MIL-EF con esquema de votación por consenso.
 - Ruido 10 % es MIL-EF con esquema de votación por consenso.
 - Ruido 15 % es MIL-EF con esquema de votación por Max Votos.
 - Ruido 20 % es MIL-EF con esquema de votación por consenso.
 - Ruido 25 % es MIL-EF con esquema de votación por Max Votos.
 - Ruido 30 % es MIL-EF con esquema de votación por Max Votos.
- De forma unificada se observa que el mejor filtro para EMDD es MIL-EF y bueno con ambos esquemas de votación ya que las diferencias en sus % son muy reducidas, además se mejora la predicción frente a no usar ninguno en todos los casos.

6.2.1.8. Clasificador MILBoost

Tabla 6.12: Tabla Resumen Clasificador MILBoost

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
SIN ESQUEMA DE VOTACIÓN							
MILBoost	51,28	49,08	48,81	46,59	46,18	45,74	46,91
POR CONSENSO							
MIL-EF	70,09	68,44	66,73	65,97	65,55	63,26	59,04
MIL-CVCF	47,27	47,27	47,27	47,27	47,27	47,27	47,27
MIL-IPF	47,27	47,27	47,27	47,27	47,27	47,27	47,27
POR MAX VOTOS							
MIL-EF	69,41	67,78	66,61	66,66	64,11	64,37	61,66
MIL-CVCF	38,10	37,85	39,13	39,02	38,87	39,02	39,89
MIL-IPF	28,05	18,64	18,50	9,18	8,28	9,62	17,42

Análisis para MILBoost

- La degradación en la métrica utilizada de Exactitud se ve reducida a medida que en los conjuntos de entrenamiento se imputa mayor porcentaje de ruido en las clases en todo caso con alguna excepción.
- El mejor Filtro MIL para esquema de votación por Consenso es MIL-EF por unanimidad.
- El mejor Filtro MIL para esquema de votación por Max Votos en MIL-EF por unanimidad.
- El mejor Filtro MIL para:
 - Ruido 0 % es MIL-EF con esquema de votación por consenso.
 - Ruido 5 % es MIL-EF con esquema de votación por consenso.
 - Ruido 10 % es MIL-EF con esquema de votación por consenso.
 - Ruido 15 % es MIL-EF con esquema de votación por Max Votos.
 - Ruido 20 % es MIL-EF con esquema de votación por consenso.
 - Ruido 25 % es MIL-EF con esquema de votación por Max Votos.
 - Ruido 30 % es MIL-EF con esquema de votación por Max Votos.
- De forma unificada se observa que el mejor filtro para MILBoost es MIL-EF y bueno con ambos esquemas de votación ya que las diferencias en sus % son muy reducidas, además se mejora la predicción frente a no usar ninguno en todos los casos.

Capítulo 7

Conclusiones

7.1. Introducción

En este capítulo analizaremos la información del estudio individualizado por clasificadores MIL del capítulo anterior comparando información con los datos de las tablas detalladas del Anexo D y determinaremos la consecución del desarrollo del proyecto planteado en el Capítulo 2, además incluiremos algunas propuestas de trabajos futuros y como cierre, se proporciona una reflexión tras la conclusión del TFG.

7.2. Análisis del Estudio

En esta sección analizaremos los datos desde un punto de vista globalizado, ya que el capítulo anterior detallamos el análisis más en profundidad de forma individualizada por cada clasificador MIL utilizado.

En la Tabla 7.1 mostramos los mejores resultados obtenidos en la métrica de Exactitud utilizada hasta ahora para evaluar los filtros de pyMIL-BNF.

Tabla 7.1: Tabla Resumen Completa

Ruido->	0 %	5 %	10 %	15 %	20 %	25 %	30 %
<i>SIN ESQUEMA DE VOTACIÓN</i>							
NONE	70,69 MILExt	67,04 CKNN	66,19 MILExt	61,99 MILExt	60,96 MILExt	58,65 MILExt	58,84 MILmax
<i>POR CONSENSO</i>							
MIL-EF	75,10 MILExt	73,05 MILmax	70,28 CKNN	69,57 MILmax	70,88 MILmax	67,82 MILmax	60,79 MILmax
MIL-CVCF	75,63 MILExt	74,68 CKNN	73,32 MILExt	72,51 MILExt	72,88 MILExt	69,03 MILExt	64,93 MILmax
MIL-IPF	75,05 CKNN	74,61 CKNN	73,37 MILExt	69,73 MILExt	69,40 MILExt	66,85 MILExt	66,33 MILmax
<i>POR MAX VOTOS</i>							
MIL-EF	73,33 MILExt	71,40 MILExt	69,95 CKNN	69,14 CKNN	66,33 EM-DD	66,60 CKNN	63,83 MILmax
MIL-CVCF	76,04 CKNN	73,49 CKNN	72,47 CKNN	70,52 CKNN	67,89 CKNN	66,67 CKNN	65,71 CKNN
MIL-IPF	41,36 CKNN	29,46 CKNN	28,36 CKNN	13,30 CKNN	13,69 CKNN	11,00 EM-DD	25,27 CKNN

Cuando analizamos los clasificadores en el capítulo anterior se repetía de forma generalizada que la mejor opción era el filtro MIL-EF con esquema de votación por Consenso, pero al ver los datos generalizados los mejores resultados los ha obtenido el filtro MIL-CVCF en todo caso.

De esta tabla además sacamos que los clasificadores MIL con mejores resultados suelen ser las variantes de SimpleMIL y C-KNN.

Otra apreciación de las más importantes es que demostramos que siem-

pre hay mejora en los resultados de la predicción cuando se usan filtros de pyMIL-BNF oscilando entre 6 % y 12 % la exactitud.

También observamos el bajo % obtenido por la combinación de MIL-IPF y esquema de votación por Max Votos, analizando la información más detallada esto se debe a que este esquema de votación llegaba a eliminar el conjunto entero de entrenamiento, haciendo que los resultados fueran nulos, cosa que no le pasaba con los demás filtros MIL.

A parte de este conocimiento extraído del análisis de las tablas, también nos planteamos 5 cuestiones previas a disponer de la información y que solo tenían respuesta una vez analizado en detenimiento toda esta gran cantidad de datos:

1. *¿Hay 1 filtro mejor para todos los niveles de ruido o bien cambia el mejor filtro según aumenta/disminuye el ruido?*

Como hemos comentado antes el que ha obtenido los mejores resultados ha sido filtro MIL-CVCF en general para todos.

2. *¿Hay 1 filtro mejor para todos los clasificadores o bien hay filtros que van mejor según qué clasificador? ¿Podemos pensar por qué?*

Al analizar los clasificadores MIL individualmente y hemos comentado previamente, de forma general el filtro MIL-EF con esquema de votación por Consenso, podemos discernir que se debe a que presenta mayor estabilidad en los resultados de sus % de exactitud que los demás.

3. *¿Se pueden comparar los resultados de los dataset filtrados con los originales sin ruido? ¿Atenúan los filtros bastante la pérdida de información: el % de ruido vs. el % de precisión que se pierde?*

Aquí tenemos 2 vertientes que vemos según el esquema de votación:

- Por consenso se pierde ruido real siempre, es decir que mejoramos de forma limpia sin perder precisión ni información útil puesto que elimina muy pocas bolsas mal etiquetadas y las que elimina es ruido de verdad.
- Por Max Votos se pierde mucha información, elimina demasiadas bolsas que son buenas y aunque el resultado final no es malo, habría que evaluar si merece la pena perder esa información.

4. *¿Hay algún nivel de ruido que se pueda considerar el límite para el cual el uso de filtros arregla poco?*

En nuestro límite de 30 % del caso de estudio aun sigue siendo viable aplicarlos, puesto que llega a una mejora del 7 %.

5. *¿Hay conjuntos de datos que se benefician más del ruido que otros? ¿Tiene algo que ver con sus características (n^o bolsas/instancias o n^o de clases o n^o de atributos)? ¿Hay 1 patrón aquí?*

El dataset que suele obtener mejores resultados es Mutagénesis hard y que por el contrario es el que menos bolsas tiene, esto puede ser causa

de que con un solo elemento que se filtre, su porcentaje de exactitud aumentará más que para los demás. Un patrón generalizado para los dataset, es que sus mejores resultados vienen de los filtros MIL-EF y MIL-CVCF con esquema Consenso y los peores resultados de el esquema de votación Max Votos y sobre todo con el filtro MIL-IPF.

7.3. Consecución de los objetivos

El objetivo principal del TFG era el desarrollo de una librería para filtro de ruido de clase en Multiple-Instance Learning. Para ello se planificaron una serie de tareas que mostramos en el Capítulo 2 y que analizamos su cumplimiento en la Tabla 7.2.

Tabla 7.2: Consecución de Objetivos

ID	Objetivo	Consecución
O.1	Aprendizaje Data Science	Si
O.2	Aprendizaje MIL	Si
O.3	Aprendizaje Python	Si
O.4	Búsqueda herramientas MIL para Python	Si
O.5	Diseño Filtros MIL	Si
O.6	Pruebas Filtros	Si
O.7	Ejecución clúster Hercules	Se redujo la muestra
O.8	Recogida de Datos	Si
O.9	Análisis de Datos	Si

7.4. Trabajos Futuros

Este trabajo aunque novedoso en su campo dista aún de ser una versión final, durante el desarrollo de este TFG has surgido ideas para posibles mejoras y ampliación de la biblioteca. A continuación listaremos las más sobresalientes:

- Referente a implementación:
 - Ampliar Filtrado basándonos en otros esquemas como Classification Filter (Gamberger, Lavrac, y Groselj, 1999b), Automatic Noise Remover (Zeng y Martinez, 2003b) o Pairwise Attribute Noise Detection Algorithm Filter (Hulse y cols., 2007).
 - Adicionalmente crear una propia biblioteca de Clasificadores MIL y ampliar la actual ya que es bastante escasa la variedad actualmente.
 - Implementar un esquema de filtrado que utilice métricas de ruido

como INFFC (iterative class noise filter based on the fusion of classifiers) (Sáez, Galar, Luengo, y Herrera, 2016).

- Referente a robustez de la librería:
 - Reparación de etiqueta: actualmente la técnica de filtrado elimina la bolsa de datos cuando es identificada como ruidosa para el calculo de la predicción, pero sería interesante intentar encontrar el valor real de la etiqueta antes de eliminarla para que entrara como un dato bueno de entrenamiento.
 - Filtros avanzados I: Afinar el filtrado por medio de los atributos de las bolsas de datos, no solo por el etiquetado de las clases.
 - Filtros avanzados II: Combinar un esquema de filtro de clases con esquemas de filtro de instancias para eliminar datos ruidosos de las bolsas de datos también, para usarlo en combinación con el ejemplo anterior.

7.5. Lecciones Aprendidas y conclusiones personales

Este TFG presenta un gran reto personal y académico, escogido ex profeso con ilusión, primero para demostrar que las competencias específicas adquiridas de cada área de la carrera tienen cabida en una corriente única, como son matemáticas, estadística, programación orientada a objetos y programación distribuida, algorítmica, IA, Inteligencia de Negocio, bases de datos, sistemas de información, etc. intentando siempre en el desarrollo aplicar alguno de estos conocimientos aprendidos durante el grado.

En segundo lugar este proyecto puede tener continuación en El Máster Universitario Oficial en Ciencia de Datos e Ingeniería de Computadores <http://masteres.ugr.es/datcom/> por su vertiente investigadora en el área de análisis de datos y sus herramientas.

Y por último, que estos conocimientos adquiridos sirvan como herramientas para adquirir otros nuevos, con esto quiero decir que todo lo que se ha usado en este TFG se ha tenido que aprender durante su desarrollo, desde la suite de Anaconda para escribir en Python v3, hasta \LaTeX para desarrollar esta misma documentación, pasando por GitHub para soporte de versiones de la aplicación, el uso de clústeres de supercomputación y una gran cantidad de conocimientos en Data Science, todo ha sido posible a la base de estudio que se tenía previamente.

Referencias

- Anaconda, I. (2012). *Anaconda navigator*. (<https://anaconda.org/>)
- Andrews, S., Hofmann, T., y Tsochantaridis, I. (2002). Multiple instance learning with generalized support vector machines. En *Eighteenth national conference on artificial intelligence* (pp. 943–944). Menlo Park, CA, USA: American Association for Artificial Intelligence. (<http://dl.acm.org/citation.cfm?id=777092.777234>)
- Arrieta, J., y Mera, C. (2017). *Milpy: Multiple-instance learning python toolbox*. (<https://github.com/jmarrieta/MILpy>)
- Brachman, R., y Anand, T. (1996). *The process of knowledge discovery in databases: A human-centered approach, em: Usama m. fayyad, gregory piatetsky-shapiro, padhraic smyth, and ramasamy uthurusamy, advances in knowledge discovery and data mining, aaai*. MIT Press.
- Briggs, F., Lakshminarayanan, B., Neal, L., Fern, X., Raich, R., Frey, S., . . . Betts, M. (2012, 06). Acoustic classification of multiple simultaneous bird species: A multi-instance multi-label approach. *The Journal of the Acoustical Society of America*, 131, 4640-50. doi: 10.1121/1.4707424
- Brodley, C., y Friedl, M. (1999). Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11, 131–167.
- Carbonneau, M., Cheplygina, V., Granger, E., y Gagnon, G. (2016). Multiple instance learning: A survey of problem characteristics and applications. *CoRR*, abs/1612.03365. (<http://arxiv.org/abs/1612.03365>)
- Chen, Y., Bi, J., y Wang, J. Z. (2006, diciembre). Miles: Multiple-instance learning via embedded instance selection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(12), 1931–1947. (<http://dx.doi.org/10.1109/TPAMI.2006.248>) doi: 10.1109/TPAMI.2006.248
- Chris Wanstrath, T. P.-W., P. J. Hyett, y Chacon, S. (2019). *Github*. (<https://github.com/>)
- Dietterich, T. G., Lathrop, R. H., y Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1), 31 - 71. (<http://www.sciencedirect.com/science/article/pii/S0004370296000343>) doi: [https://doi.org/10.1016/S0004-3702\(96\)00034-3](https://doi.org/10.1016/S0004-3702(96)00034-3)

- Ferreyra, M. R. (2007). *¿qué es el ruido?* (<http://powerhousedm.blogspot.com/2007/10/qu-es-el-ruido.html>)
- Forbes. (2016). *Cleaning big data: Most time-consuming, least enjoyable data science task, survey says.* (<https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/>)
- Foundation, A. S. (2016). *Apache openoffice.* (<http://www.openoffice.org/>)
- Gamberger, D., Lavrac, N., y Dzroski, S. (2000). Noise detection and elimination in data preprocessing: Experiments in medical domains. *Applied Artificial Intelligence*, 14(2), 205–223.
- Gamberger, D., Lavrac, N., y Groselj, C. (1999a). Experiments with noise filtering in a medical domain. En *16th international conference on machine learning(ICML99)* (pp. 143–151).
- Gamberger, D., Lavrac, N., y Groselj, C. (1999b). Experiments with noise filtering in a medical domain. En *16th international conference on machine learning(ICML99)* (pp. 143–151).
- Garcia, S., Luengo, J., y Herrera, F. (2014). *Data preprocessing in data mining.* Springer Publishing Company, Incorporated.
- GitHub, I. (2019). *Github desktop.* (<https://desktop.github.com/>)
- Goodwin, J., y Garfield, E. (1980, 10). Citation indexing-its theory and application in science, technology, and humanities. *Technology and Culture*, 21, 714. doi: 10.2307/3104125
- Hulse, J., Khoshgoftaar, T., y Huang, H. (2007). The pairwise attribute noise detection algorithm. *Knowledge and Information Systems*, 11(2), 171–190.
- KDnuggets. (2018). *Python eats away at r: Top software for analytics, data science, machine learning in 2018: Trends and analysis.* (<https://www.kdnuggets.com/2018/05/poll-tools-analytics-data-science-machine-learning-results.html>)
- Khoshgoftaar, T., y Rebours, P. (2007). Improving software quality prediction by noise filtering techniques. *Journal of Computer Science and Technology*, 22, 387–396.
- Kosse, T. (2019). *Filezilla.* (<https://filezilla-project.org/>)
- Kozyrkov, C. (2019). *Machine learning:secretos de magia revelados.* (<https://medium.com/datos-y-ciencia/machine-learning-secretos-de-magia-revelados-182fe6ae28c3>)
- Kumar, J., Pillai, J., y Doermann, D. (2011, 10). Document image classification and labeling using multiple instance learning. En (p. 1059 - 1063). doi: 10.1109/ICDAR.2011.214
- Maron, O., y Lozano-Pérez, T. (1998). A framework for multiple-instance learning. En *Advances in neural information processing systems* (pp. 570–576).

- Media, E. (2017). *What do you mean by data preprocessing and why it is needed?* (<https://www.electronicmedia.info/2017/12/20/what-is-data-preprocessing/>)
- Mishra, M. (2018). *Understanding roc and auc curve.* (<https://medium.com/datadriveninvestor/understanding-roc-auc-curve-7b706fb710cb>)
- Morales, P., Luengo, J., Garcia, L. P., Lorena, A. C., C.P.L.F. de Carvalho, A., y Herrera, F. (2017). *Noisy data in data mining.* (<https://sci2s.ugr.es/noisydata>)
- of Tuebingen, U. (2001). *yed graph editor.* (<https://www.yworks.com/>)
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Python. (2019). *Python: History and license.* (<https://docs.python.org/3/license.html>)
- Raschka, S. (2016). *Python model tuning methods using cross validation and grid search.* (<http://karlroosaen.com/ml/learning-log/2016-06-20/>)
- S., M., Cano, I., y Arán, A. (1997). *Desarrollo y gestión de proyectos informáticos.*
- Sáez, J. A., Galar, M., Luengo, J., y Herrera, F. (2016). Inffc: An iterative class noise filter based on the fusion of classifiers with noise sensitivity control. *Information Fusion*, 27, 19 - 32. Descargado de <http://www.sciencedirect.com/science/article/pii/S156625351500038X> doi: <https://doi.org/10.1016/j.inffus.2015.04.002>
- Srinivasan, A., Muggleton, S., y King, R. (1995, 01). Comparing the use of background knowledge by inductive logic programming systems. *Comparing the use of background knowledge by inductive logic programming systems*, 99–230.
- Tatham, S. (2019). *Putty.* (<https://www.putty.org/>)
- Tax, D. M., y Cheplygina, V. (2016, Jun). *MIL, a Matlab toolbox for multiple instance learning.* (<http://prlab.tudelft.nl/david-tax/mil.html>, version 1.2.1)
- van der Zander, B. (2009). *Texstudio.* (<http://texstudio.sourceforge.net/>)
- Verbaeten, S., y Assche, A. (2003). Ensemble methods for noise elimination in classification problems. En *4th international workshop on multiple classifier systems(MCS 2003)* (Vol. 2709, pp. 317–325). Springer.
- Viola, P., Platt, J., y Zhang, C. (2006, 01). Multiple instance boosting for object detection. *Proc. Adv. Neural Inf. Process. Syst.*, 18, 1417–1426.
- Wang, J., y Zucker, J.-d. (2000, 01). Solving the multiple-instance problem: A lazy learning approach. En (p. 1119–1126).

- Zeng, X., y Martinez, T. (2003a). A noise filtering method using neural networks. En *Ieee international workshop on soft computing techniques in instrumentation and measurement and related applications(SCIMA2003)* (pp. 26–31).
- Zeng, X., y Martinez, T. (2003b). A noise filtering method using neural networks. En *Ieee international workshop on soft computing techniques in instrumentation and measurement and related applications(SCIMA2003)* (pp. 26–31).
- Zhang, Q., y A. Goldman, S. (2002, 04). Em-dd: An improved multiple-instance learning technique. En (Vol. 14).
- Zhu, X., y Wu, X. (2004, noviembre). Class noise vs. attribute noise: A quantitative study of their impacts. *Artif. Intell. Rev.*, 22(3), 177–210. (<http://dx.doi.org/10.1007/s10462-004-0751-8>) doi: 10.1007/s10462-004-0751-8

Anexos

Anexos A

Manual de Usuario

En las siguientes secciones desarrollamos un pequeño manual de uso de la librería y de la Interfaz gráfica de pyMIL-BNF. En cualquier caso lo primero es ir al sitio de descarga del proyecto que se encuentra en GitHub <https://github.com/jcarlosorte/TFG> y clonar o descargar el proyecto en nuestro entorno.

pyMIL-BNF: Multiple Instance Learning Bag Noise Filters library in Python

TFG Multiple-Instance Learning Python Project by Juan Carlos Orte (jcarlosorte@ugr.es)

Overview

The TFG project uses the toolbox MILpy, the toolbox contains algorithms to train and evaluate multiple instance learning classifiers.

Stage of Development:

-> Development still on process

Disclaimer

This Python project implementation is inspired by MILpy toolbox.

Data from <http://www.miprobems.org/datasets/>

Cannot guarantee any support for this software.

License

Copyright 2018-2019 by UGR - Universidad de Granada (Ceuta) Permission to use, copy, or modify these programs and their documentation for educational and research purposes only and without fee is hereby granted, provided that this copyright notice appears on all copies and supporting documentation. For any other uses of this software, in original or modified form, including but not limited to distribution in whole or in part, specific prior permission must be obtained from Universidad de Granada. These programs shall not be used, rewritten, or adapted as the basis of a commercial software or hardware product without first obtaining appropriate licenses from the Universidad de Granada. Universidad de Granada makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

A.1. Librería pyMIL-BNF

Vamos a analizar un menú simple que implementa la librería pyMIL-BNF en el código y los variables necesarias para iniciarla, así como las dependencias de otras bibliotecas o cabeceras necesarias.

```

1  import sys,os
2
3  os.chdir('MILpy')
4  sys.path.append(os.path.realpath('..'))
5
6  from filters import EF
7  from filters import CVCF
8  from filters import IPF
9
10 folds = 5
11 votacion = 'consenso'
12 DataSet = ['tiger_scaled']
13 ruido = [0]
14
15 print('***** MIL-Ensemble Filter *****')
16 EF.EF(DataSet,votacion,folds,ruido)
17 print('***** MIL-CV Committees Filter *****')
18 CVCF.CVCF(DataSet,votacion,folds,ruido)
19 print('***** MIL-Iterative Partitioning Filter *****')
20 IPF.IPF(DataSet,votacion,folds,ruido)

```

1. Línea 1: Importamos las librerías sys y os que utilizaremos para renombrar las variables del entorno de trabajo.
2. Líneas 2 y 3: Re-nombramos a estas variables con esas carpetas de trabajo, que será necesario para poder usar MILpy.
3. Líneas 6, 7 y 8: Importamos con 'from' nuestra biblioteca 'filters' cada uno de los Filtros que queramos usar con 'import'.
4. Líneas 10, 11, 12 y 13: Aquí declararemos las variables necesarias para que los filtros trabajen con la configuración que queramos:
 - folds: Variable numérica que sirve para definir la K en la validación cruzada.
 - votacion: Variable para elegir el esquema de votación que tendrán los filtros. Opciones: 'consenso' y 'maxVotos'.
 - DataSet: Esta variable es un array que usamos para definir los dataset que queremos usar en la ejecución, permite introducir todos los dataset que queramos en él siempre y cuando se encuentren en la carpeta \MILpy\data y nombrado con el mismo nombre que tenga la propia carpeta de dataset. Ej: [tiger_scaled,corel.african original]
 - ruido: Esta variable es un array y sirve para definir el nivel o

niveles de ruido artificial con el que se imputarán las clases de entrenamiento siendo 0% el valor que no hace modificaciones.

Ej: [0,5,10,15,20,25,30]

5. Líneas 16, 18 y 20: Llamadas a las funciones de los filtros pasando las variables definidas anteriormente.

Con esto tendríamos totalmente operativa la librería, esta ejecución sacará por consola los resultados de la combinación de todos clasificadores. Podemos definir que clasificadores usar para el filtro y para obtener la predicción indiferentemente de la siguiente manera:

- Definir Clasificador MIL para predicción: dentro de cada archivo .py de los filtros (EF.py, CVCF.py e IPF.py) en la carpeta \filters\buscar la función 'clasif()' y dentro encontraremos las siguientes líneas:

```
1 aux.append(SMILaMax)
2 aux.append(SMILaMin)
3 aux.append(SMILaExt)
4 aux.append(BOW_clas)
5 aux.append(CKNN_cla)
6 aux.append(maxDD_cl)
7 aux.append(EMDD_cla)
8 aux.append(MILB_cla)
```

Basta con comentar con # delante de la línea de cada clasificador que no se quiera usar.

- Definir Clasificadores MIL para Filtrar Ruido: al igual que en el anterior hay que buscar una función en los archivos nombrados, esta vez la función se llama 'cla_filter_cvcf()' para CVCF.py y IPF.py y 'cla_filter()' para EF.py donde dentro encontramos para todos las mismas líneas:

```
1 aux.append(SMILaMax)
2 aux.append(SMILaMin)
3 aux.append(SMILaExt)
4 aux.append(BOW_clas)
5 aux.append(CKNN_cla)
6 aux.append(maxDD_cl)
7 aux.append(EMDD_cla)
8 aux.append(MILB_cla)
```

Al igual que antes basta con comentar con # delante de la línea de cada clasificador que no se quiera usar para filtrar. Reacuerdo que el esquema de MIL-EF usa un mínimo de 3 clasificadores.

Estos son todas las configuraciones básicas y opcionales que se tienen que manejar, se ha tentado simplificar lo máximo posible para que su uso no sea muy complicado. Si todo va bien veremos algo como lo que se muestra en la Figura A.1

```
***** MIL-Iterative Partitioning Filter *****  
  
DATASET: tiger_scaled  
  
=>RUIDO : 0  
=>Elementos eliminados por MIL max: 0  
=>Elementos eliminados por MIL max: 0  
=>Elementos eliminados por MIL max: 0  
=>Elementos eliminados por MIL max: 0  
=>Elementos eliminados por MIL max: 0  
-->Clasificador :MIL max  
-->Original  
Precision: 73.5%  
Roc Score: 73.5  
-->Filtrado por MIL max  
Precision: 73.5%  
Roc Score: 73.5  
  
In [3]:
```

Figura A.1: Ejecución en consola de pyMIL-BNF

A.2. Interfaz gráfica pyMIL-BNF

Para usar la interfaz gráfica basta con ejecutar el archivo 'filtroApp.py'. Así iniciaremos la aplicación como se ve en la Figura A.2.

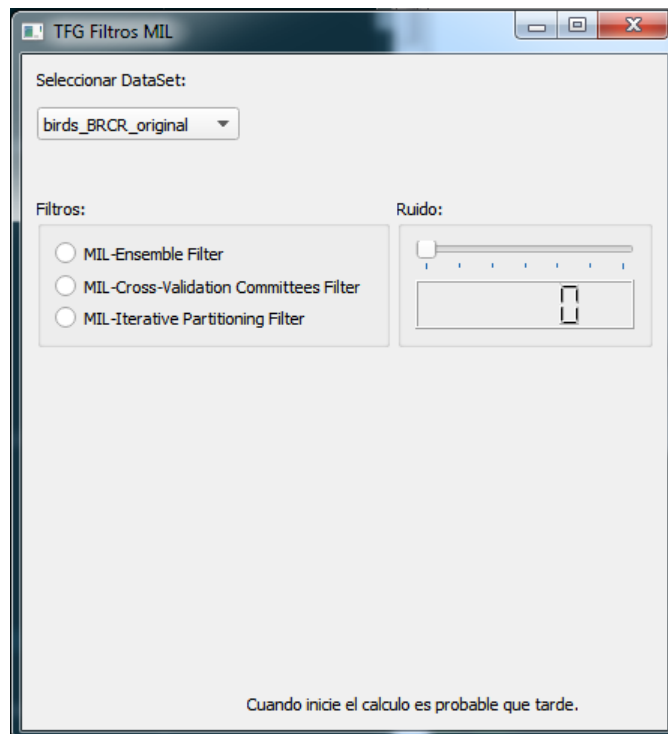


Figura A.2: Inicio Interfaz gráfica pyMIL-BN

Los 3 parámetros iniciales son:

- Seleccionar Dataset: Listado que comprueba la carpeta donde se alojan los dataset (\MILpy\data\) y nos da un listado de todos los que podamos usar.
- Filtros: Seleccionamos el filtro que se quiere usar para el dataset de los 3 implementados para el proyecto.
- Ruido: Establecemos nivel de ruido artificial, permite los valores: 0, 5, 10, 15, 20, 25 y 30. (Con 0 % no se modifica el conjunto de entrenamiento)

Al pinchar sobre uno de los esquemas de filtro, se nos abren las características de configuración propias de cada uno como vemos en la Figura A.3.

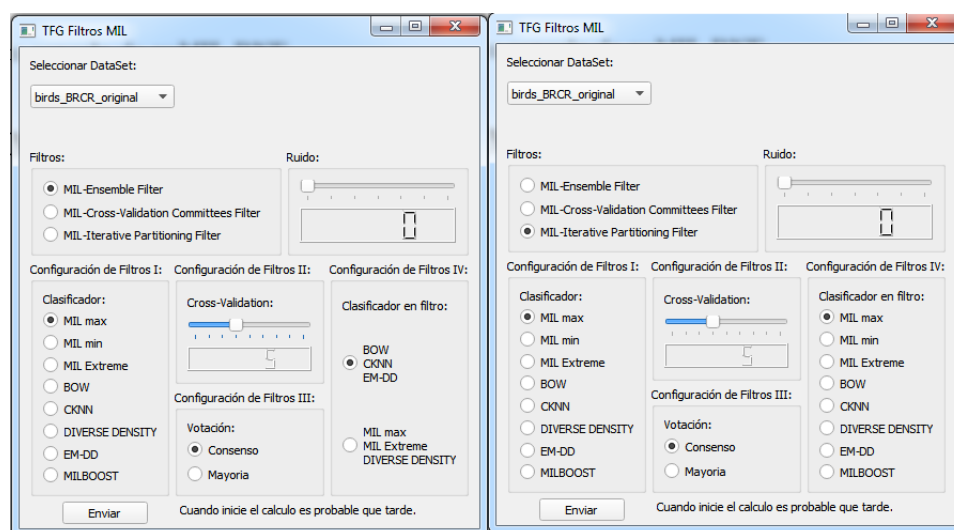
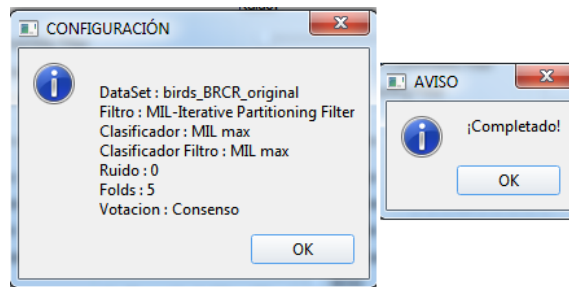


Figura A.3: Ejemplo de interfaz para parámetros de los Filtros

Se nos abren las siguientes opciones:

- Configurador de Filtros I: Elegir el Clasificador MIL para predicción.
- Configurador de Filtros II: Seleccionar el valor de la validación cruzada, se ha establecido un rango de [2-10].
- Configurador de Filtros III: Señalar el esquema de votación.
- Configurador de Filtros IV: Elegir el Clasificador MIL para usar en el filtro (o grupo de ellos en caso de MIL-EF).

Cuando tengamos la configuración deseada y pulsemos en el botón enviar nos aparecerá una ventana de confirmación de datos Figura A.2 Izda y empezará el programa a ejecutar los filtros y cuando finalice una nueva ventana de alerta aparecerá, Figura A.2 Dcha.



Ahora se cargará la información de la predicción en la parte superior derecha de la ventana como vemos en la Figura .. y esto será todo. Podremos volver a usarla hasta que salgamos de ella.

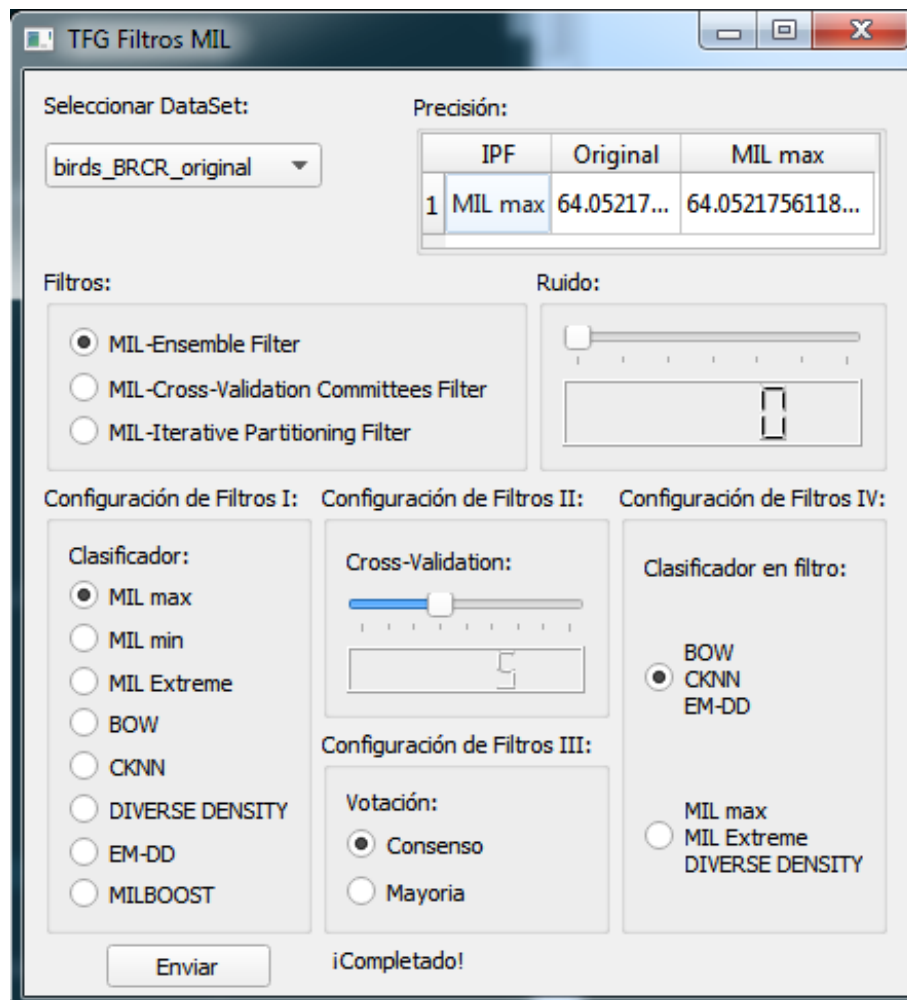


Figura A.4: Interfaz con predicción del clasificador MIL

Anexos B

Código Fuente de pyMIL-BNF

B.1. Interfaz gráfica pyMIL-BNF

```

1  import sys,os,warnings
2  from os import walk
3  from PyQt5.QtCore import Qt, QThread
4  from PyQt5.QtWidgets import QWidget, QApplication, QVBoxLayout,
    QLabel, \
5  QPushButton, QComboBox, QStyleFactory, QTableWidgetItem, \
6  QTableWidgetItem, QGroupBox, QRadioButton, \
7  QSlider, QLCDNumber, QMessageBox
8  warnings.filterwarnings('ignore')
9  os.chdir('MILpy')
10 sys.path.append(os.path.realpath('..'))
11 import pandas as pd
12 from filtersApp import EF
13 from filtersApp import CVCF
14 from filtersApp import IPF
15
16 class CargaAlgoritmo(QThread):
17     def __init__(self,ck_clasif, DataSet,votacion,folds,ruido,
    clasif,clasif_F):
18         super().__init__()
19         self._ck_clasif = ck_clasif
20         self._DataSet = DataSet
21         self._votacion = votacion
22         self._folds = folds
23         self._ruido = ruido
24         self._clasif = clasif
25         self._clasif_F = clasif_F
26     def run(self):
27         if self._ck_clasif == 1:
28             print("MIL-Ensemble Filter")
29             EF.EF(self._DataSet,self._votacion,self._folds,self
    ._ruido,self._clasif,self._clasif_F)
30         elif self._ck_clasif == 2:
31             print("MIL-Cross-Validation Committees Filter")
32             CVCF.CVCf(self._DataSet,self._votacion,self._folds,
    self._ruido,self._clasif,self._clasif_F)
33         elif self._ck_clasif == 3:
34             print("MIL-Iterative Partitioning Filter")
35             IPF.IPF(self._DataSet,self._votacion,self._folds,
    self._ruido,self._clasif,self._clasif_F)
36
37
38 class FilterTFG(QWidget):
39     def __init__(self, parent=None):
40         super(FilterTFG, self).__init__(parent)
41         self.initUI()
42         QApplication.setStyle(QStyleFactory.create('Fusion'))
43     def initUI(self):
44         self.addDataSet()
45         self.addRadioBotonFilter()

```

```

46         self.addSliderRuido()
47         self.addTable()
48         self.addEnviar()
49         self.resize(430, 450)
50         self.setWindowTitle('TFG Filtros MIL')
51
52     def addTable(self):
53         self.gbx_t = QGroupBox('Precisión:', self)
54         self.gbx_t.setGeometry(190, 10, 230, 80)
55         self.layout_t = QVBoxLayout(self)
56         self.layout_t.setContentsMargins(0, 0, 0, 0)
57         # Create table
58         self.tableWidget = QTableWidgetItem()
59         self.layout_t.addWidget(self.tableWidget)
60         self.gbx_t.setLayout(self.layout_t)
61         self.gbx_t.hide()
62
63     def addDataSet(self):
64         items = self.ls(os.getcwd()+str("/data/"))
65         self.Ldt = QComboBox(self)
66         self.Ldt.addItem(items)
67         self.Ldt.move(10, 35)
68         lbl = QLabel('Seleccionar DataSet:', self)
69         lbl.move(10, 10)
70
71     def addEnviar(self):
72         self.label = QLabel("Cuando inicie el calculo es
73 probable que tarde.", self)
74         self.label.setGeometry(150, 420, 220, 25)
75         self.btn1 = QPushButton("Enviar", self)
76         self.btn1.move(40, 425)
77         self.btn1.hide()
78         self.btn1.pressed.connect(self.buttonClicked)
79     def buttonClicked(self):
80         self.file_data = '../filtersApp/tabla.csv'
81         folds = 0
82         votacion = ''
83         DataSet = []
84         ruido = []
85         clasif = ""
86         clasif_F = ""
87         self.label.setText("Esto puede tardar.Calculando...")
88         self.btn1.setEnabled(False)
89         btn_txt = "\nDataSet : " + self.Ldt.currentText()
90         DataSet.append(self.Ldt.currentText())
91         ck_clasif = 0
92         #Valor Filter
93         for i, radio in enumerate(self.RbFilter):
94             if radio.isChecked():
95                 btn_txt = btn_txt + "\nFiltro : " + radio.text
96
97         ()
98
99         if radio.text() == "MIL-Ensemble Filter":
100             ck_clasif = 1

```

```

97         for j, radio2 in enumerate(self.RbClasif_ef
98     ):
99         if radio2.isChecked():
100             btn_txt = btn_txt + "\nClasificador
101             : " + radio2.text()
102             clasif = radio2.text()
103         for p, radio4 in enumerate(self.
104     RbClasif_F_ef):
105         if radio4.isChecked():
106             btn_txt = btn_txt + "\nClasificador
107             Filtro : " + radio4.text()
108             clasif_F = str(p)
109         elif radio.text() == "MIL-Cross-Validation
110     Committees Filter":
111             ck_clasif = 2
112             for j, radio2 in enumerate(self.
113     RbClasif_cvcf):
114             if radio2.isChecked():
115                 btn_txt = btn_txt + "\nClasificador
116                 : " + radio2.text()
117                 clasif = radio2.text()
118             for p, radio4 in enumerate(self.
119     RbClasif_F_cvcf):
120             if radio4.isChecked():
121                 btn_txt = btn_txt + "\nClasificador
122                 Filtro : " + radio4.text()
123                 clasif_F = radio4.text()
124             elif radio.text() == "MIL-Iterative
125     Partitioning Filter":
126                 ck_clasif = 3
127                 for j, radio2 in enumerate(self.
128     RbClasif_ipf):
129                 if radio2.isChecked():
130                     btn_txt = btn_txt + "\nClasificador
131                     : " + radio2.text()
132                     clasif = radio2.text()
133             for p, radio4 in enumerate(self.
134     RbClasif_F_ipf):
135                 if radio4.isChecked():
136                     btn_txt = btn_txt + "\nClasificador
137                     Filtro : " + radio4.text()
138                     clasif_F = radio4.text()
139             btn_txt = btn_txt + "\nRuido : " + str(int(self.num.
140     value()))
141             ruido.append(int(self.num.value()))
142             btn_txt = btn_txt + "\nFolds : " + str(int(self.num2.
143     value()))
144             folds = int(self.num2.value())
145             for u, radio3 in enumerate(self.RbVotacion):
146                 if radio3.isChecked():
147                     btn_txt = btn_txt + "\nVotacion : " + radio3.
148     text()
149                 if radio3.text() == "Consenso":
150                     votacion = 'consenso'

```

```

134         elif radio3.text() == "Mayoria":
135             votacion = 'maxVotos'
136
137         QMessageBox.information(self, 'CONFIGURACIÓN', btn_txt)
138
139         self.cargaAlgoritmo = CargaAlgoritmo(ck_clasif, DataSet,
140         votacion, folds, ruido, clasif, clasif_F)
141         self.cargaAlgoritmo.finished.connect(self.cargaCompleta
142         )
143         self.cargaAlgoritmo.start()
144
145     def cargaCompleta(self):
146         QMessageBox.information(self, 'AVISO', "Completado")
147         self.label.setText("Completado")
148         self.loadCsv(self.file_data)
149         self.btn1.setEnabled(True)
150         del self.cargaAlgoritmo
151
152     def loadCsv(self, fileName):
153         df = pd.read_csv(filepath_or_buffer=fileName, sep=';',
154         index_col=0)
155         # df.loc[:, ~df.columns.str.contains('~Unnamed')]
156         df1 = df.columns
157         self.tableWidget.setRowCount(0)
158         self.tableWidget.setColumnCount(3)
159         self.tableWidget.setHorizontalHeaderLabels([df1[0], df1
160         [1], df1[2]])
161         for i, row in enumerate(df.values):
162             rowPosition = self.tableWidget.rowCount()
163             self.tableWidget.insertRow(rowPosition)
164             for h, li in enumerate(row):
165                 self.tableWidget.setItem(i, h, QTableWidgetItem
166                 (str(li)))
167             self.tableWidget.resizeColumnsToContents()
168             self.gbx_t.show()
169
170     def addRadioBotonFilter(self):
171         gbx = QGroupBox('Filtros:', self)
172         gbx.setGeometry(10, 95, 235, 100)
173         # crear tres QRadioButton
174         self.RbFilter = []
175         self.radio1 = QRadioButton("MIL-Ensemble Filter")
176         self.radio2 = QRadioButton("MIL-Cross-Validation
177         Committees Filter")
178         self.radio3 = QRadioButton("MIL-Iterative Partitioning
179         Filter")
180         self.radio1.toggled.connect(lambda: self.btnstate(self.
181         radio1))
182         self.radio2.toggled.connect(lambda: self.btnstate(self.
183         radio2))
184         self.radio3.toggled.connect(lambda: self.btnstate(self.
185         radio3))
186         self.RbFilter.append(self.radio1)
187         self.RbFilter.append(self.radio2)

```

```

178         self.RbFilter.append(self.radio3)
179         # agregar los widgets al layout vertical
180         self.vbox = QVBoxLayout(self)
181         self.vbox.addWidget(self.radio1)
182         self.vbox.addWidget(self.radio2)
183         self.vbox.addWidget(self.radio3)
184
185         gbx.setLayout(self.vbox)
186         #EF-----
187         self.gbx2 = QGroupBox('Configuración de Filtros I:',
self)
188         self.gbx2.setGeometry(10, 200, 130, 220)
189         self.vbox2 = QVBoxLayout(self)
190         self.vbox2.addWidget(QLabel('Clasificador:', self))
191         clasif_ef = EF.clasif()
192         check1 = True
193         self.RbClasif_ef = []
194         for s,cl in enumerate(clasif_ef):
195             if check1:
196                 butt = QRadioButton(str(cl[2]))
197                 butt.setChecked(True)
198                 self.vbox2.addWidget(butt)
199                 self.RbClasif_ef.append(butt)
200                 check1 = False
201             else:
202                 butt = QRadioButton(str(cl[2]))
203                 self.vbox2.addWidget(butt)
204                 self.RbClasif_ef.append(butt)
205         self.gbx2.setLayout(self.vbox2)
206         self.gbx2.hide()
207         #CVCF-----
208         self.gbx3 = QGroupBox('Configuración de Filtros I:',
self)
209         self.gbx3.setGeometry(10, 200, 130, 220)
210         self.vbox3 = QVBoxLayout(self)
211         self.vbox3.addWidget(QLabel('Clasificador:', self))
212         clasif_cvcf = CVCF.clasif()
213         check2 = True
214         self.RbClasif_cvcf = []
215         for s,cl in enumerate(clasif_cvcf):
216             if check2:
217                 butt = QRadioButton(str(cl[2]))
218                 butt.setChecked(True)
219                 self.vbox3.addWidget(butt)
220                 self.RbClasif_cvcf.append(butt)
221                 check2 = False
222             else:
223                 butt = QRadioButton(str(cl[2]))
224                 self.vbox3.addWidget(butt)
225                 self.RbClasif_cvcf.append(butt)
226         self.gbx3.setLayout(self.vbox3)
227         self.gbx3.hide()
228         #IPF-----

```



```

229         self.gbx4 = QGroupBox('Configuración de Filtros I:',
self)
230         self.gbx4.setGeometry(10, 200, 130, 220)
231         self.vbox4 = QVBoxLayout(self)
232         self.vbox4.addWidget(QLabel('Clasificador:', self))
233         clasif_ipf = IPF.clasif()
234         check3 = True
235         self.RbClasif_ipf = []
236         for s,cl in enumerate(clasif_ipf):
237             if check3:
238                 butt = QRadioButton(str(cl[2]))
239                 butt.setChecked(True)
240                 self.vbox4.addWidget(butt)
241                 self.RbClasif_ipf.append(butt)
242                 check3 = False
243             else:
244                 butt = QRadioButton(str(cl[2]))
245                 self.vbox4.addWidget(butt)
246                 self.RbClasif_ipf.append(butt)
247         self.gbx4.setLayout(self.vbox4)
248         self.gbx4.hide()
249
250         self.gbx6 = QGroupBox('Configuración de Filtros II:',
self)
251         self.gbx6.setGeometry(145, 200, 140, 115)
252         self.vbox6 = QVBoxLayout(self)
253         self.vbox6.addWidget(QLabel('Cross-Validation:', self))
254         sld2 = QSlider(Qt.Horizontal, self)
255         sld2.setMinimum(2)
256         sld2.setMaximum(10)
257         sld2.setValue(5)
258         sld2.setTickPosition(QSlider.TicksBelow)
259         sld2.setTickInterval(1)
260         self.num2 = QLCDNumber(self)
261         self.num2.display(sld2.value())
262         sld2.valueChanged.connect(self.num2.display)
263         self.vbox6.addWidget(sld2)
264         self.vbox6.addWidget(self.num2)
265         self.gbx6.setLayout(self.vbox6)
266         self.gbx6.hide()
267         #Votos-----
268         self.gbx5 = QGroupBox('Configuración de Filtros III:',
self)
269         self.gbx5.setGeometry(145, 320, 140, 100)
270         self.vbox5 = QVBoxLayout(self)
271         self.vbox5.addWidget(QLabel('Votación:', self))
272         self.RbVotacion = []
273         butt2 = QRadioButton("Consenso")
274         butt2.setChecked(True)
275         self.vbox5.addWidget(butt2)
276         butt2_b = QRadioButton("Mayoria")
277         self.vbox5.addWidget(butt2_b)
278         self.RbVotacion.append(butt2)
279         self.RbVotacion.append(butt2_b)

```

```

280         self.gbx5.setLayout(self.vbox5)
281         self.gbx5.hide()
282         #EF-----
283         self.gbx7 = QGroupBox('Configuración de Filtros IV:',
self)
284         self.gbx7.setGeometry(290, 200, 130, 220)
285         self.vbox7 = QVBoxLayout(self)
286         self.vbox7.addWidget(QLabel('Clasificador en filtro:',
self))
287         cla_F_EF = EF.cla_filter()
288         nombre1 = "\n"
289         self.RbClasif_F_ef = []
290         for s,cl in enumerate(cla_F_EF):
291             nombre1 = nombre1 + str(cl[2]) + "\n"
292         cla_F2_EF = EF.cla_filter2()
293         nombre2 = "\n"
294         for s,cl in enumerate(cla_F2_EF):
295             nombre2 = nombre2 + str(cl[2]) + "\n"
296         butt3 = QRadioButton(nombre1)
297         butt3.setChecked(True)
298         self.vbox7.addWidget(butt3)
299         butt3_b = QRadioButton(nombre2)
300         self.vbox7.addWidget(butt3_b)
301         self.RbClasif_F_ef.append(butt3)
302         self.RbClasif_F_ef.append(butt3_b)
303         self.gbx7.setLayout(self.vbox7)
304         self.gbx7.hide()
305         #CVCF-----
306         self.gbx8 = QGroupBox('Configuración de Filtros IV:',
self)
307         self.gbx8.setGeometry(290, 200, 130, 220)
308         self.vbox8 = QVBoxLayout(self)
309         self.vbox8.addWidget(QLabel('Clasificador en filtro:',
self))
310         cla_F_CVCF = CVCF.cla_filter_cvcf()
311         check4 = True
312         self.RbClasif_F_cvcf = []
313         for s,cl in enumerate(cla_F_CVCF):
314             if check4:
315                 butt = QRadioButton(str(cl[2]))
316                 butt.setChecked(True)
317                 self.vbox8.addWidget(butt)
318                 self.RbClasif_F_cvcf.append(butt)
319                 check4 = False
320             else:
321                 butt = QRadioButton(str(cl[2]))
322                 self.vbox8.addWidget(butt)
323                 self.RbClasif_F_cvcf.append(butt)
324         self.gbx8.setLayout(self.vbox8)
325         self.gbx8.hide()
326         #IPF-----
327         self.gbx9 = QGroupBox('Configuración de Filtros IV:',
self)
328         self.gbx9.setGeometry(290, 200, 130, 220)

```

```

329         self.vbox9 = QVBoxLayout(self)
330         self.vbox9.addWidget(QLabel('Clasificador en filtro:',
self))
331         cla_F_IPF = IPF.cla_filter_ipf()
332         check5 = True
333         self.RbClasif_F_ipf = []
334         for s,cl in enumerate(cla_F_IPF):
335             if check5:
336                 butt = QRadioButton(str(cl[2]))
337                 butt.setChecked(True)
338                 self.vbox9.addWidget(butt)
339                 self.RbClasif_F_ipf.append(butt)
340                 check5 = False
341             else:
342                 butt = QRadioButton(str(cl[2]))
343                 self.vbox9.addWidget(butt)
344                 self.RbClasif_F_ipf.append(butt)
345         self.gbx9.setLayout(self.vbox9)
346         self.gbx9.hide()
347     def btnstate(self,b):
348
349         if b.text() == "MIL-Ensemble Filter":
350             if b.isChecked() == True:
351                 self.gbx2.show()
352                 self.gbx5.show()
353                 self.gbx6.show()
354                 self.gbx7.show()
355                 self.btn1.show()
356             # print (b.text()+" is selected")
357             else:
358                 # print (b.text()+" is deselected")
359                 self.gbx2.hide()
360                 self.gbx7.hide()
361         if b.text() == "MIL-Cross-Validation Committees Filter"
:
362             if b.isChecked() == True:
363                 self.gbx3.show()
364                 self.gbx5.show()
365                 self.gbx6.show()
366                 self.gbx8.show()
367                 self.btn1.show()
368             # print (b.text()+" is selected")
369             else:
370                 # print (b.text()+" is deselected")
371                 self.gbx3.hide()
372                 self.gbx8.hide()
373         if b.text() == "MIL-Iterative Partitioning Filter":
374             if b.isChecked() == True:
375                 self.gbx4.show()
376                 self.gbx5.show()
377                 self.gbx6.show()
378                 self.gbx9.show()
379                 self.btn1.show()
380             # print (b.text()+" is selected")

```

```

381         else:
382     #             print (b.text()+" is deselected")
383                 self.gbx4.hide()
384                 self.gbx9.hide()
385     def addSliderRuido(self):
386         self.gbx10 = QGroupBox('Ruido:', self)
387         self.gbx10.setGeometry(250, 95, 170, 100)
388         self.vbox10 = QVBoxLayout(self)
389         sld = QSlider(Qt.Horizontal, self)
390         sld.setMinimum(0)
391         sld.setMaximum(6)
392         sld.setValue(0)
393         sld.setSingleStep(1)
394         sld.setPageStep(1)
395         sld.setTickPosition(QSlider.TicksBelow)
396         sld.setTickInterval(1)
397         self.num = QLCDNumber(self)
398         self.num.display(sld.value())
399         sld.valueChanged.connect(self.valChange)
400         self.vbox10.addWidget(sld)
401         self.vbox10.addWidget(self.num)
402         self.gbx10.setLayout(self.vbox10)
403
404     def valChange(self, value):
405         value = value * 5
406         self.num.display(value)
407
408     def ls(self, ruta):
409         dir, subdirs, archivos = next(walk(ruta))
410
411         try:
412             subdirs.remove("__pycache__")
413         except:
414             print()
415
416         return subdirs
417
418 if __name__ == '__main__':
419     app = QApplication(sys.argv)
420     ejm = FilterTFG()
421     ejm.show()
422     app.aboutToQuit.connect(app.deleteLater)
423     sys.exit(app.exec_())

```

B.2. Módulo MIL-EF

```

1  import warnings, copy
2  from MILpy.data.load_data import load_data
3  import random as rand
4  import numpy as np
5  import pandas as pd

```

```

6 from sklearn.utils import shuffle
7 from sklearn.model_selection import StratifiedKFold
8 warnings.filterwarnings('ignore')
9 from sklearn.metrics import roc_auc_score, accuracy_score
10 #from funciones import fun_aux
11 #Import Algorithms
12 from MILpy.Algorithms.simpleMIL import simpleMIL
13 from MILpy.Algorithms.MILBoost import MILBoost
14 from MILpy.Algorithms.maxDD import maxDD
15 from MILpy.Algorithms.CKNN import CKNN
16 from MILpy.Algorithms.EMDD import EMDD
17 from MILpy.Algorithms.BOW import BOW
18
19 def EF(b,votacion,folds,ruido,clasif_0,clasif_F):
20     for DataSet in b:
21         bags,labels,X = load_data(DataSet)
22         bags,labels = shuffle(bags, labels, random_state=rand.
23                               randint(0, len(labels)-1))
24         skf = StratifiedKFold(n_splits=folds)
25     # dataAcc = np.zeros((len(b),len(ruido),folds,2))
26     print('\n\tDATASET: '+str(DataSet)+'\n')
27     for ny,k in enumerate(ruido):
28         print('\t\t=>RUIDO : '+str(k))
29         file_data = '../filtersApp/tabla.csv'
30         data = {}
31         data['EF'] = []
32         data['Original'] = []
33         data['Filtro_1'] = []
34         # data['Filtro_2'] = []
35         fold = 1
36         results_Fil = []
37         # results_Fil2 = []
38         results_Ori = []
39         Clasificadores_fake = clasif()
40         Clasificadores = []
41         for s,cl in enumerate(Clasificadores_fake):
42             if str(cl[2]) == clasif_0:
43                 Clasificadores.append(cl)
44             for train_index, test_index in skf.split(bags,
45             labels.reshape(len(labels))):
46                 print('\t\t=>FOLD : '+str(fold))
47                 X_train = [bags[i] for i in train_index]
48                 Y_train = labels[train_index]
49                 X_test = [bags[i] for i in test_index]
50                 Y_test = labels[test_index]
51                 LabelToChange = Porcentaje(len(train_index),k)
52                 aleatorios = rand.sample(range(0,len(
53                 train_index)),int(LabelToChange))
54                 for al in aleatorios:
55                     if Y_train[al] == 0:
56                         Y_train[al] = Y_train[al]+1
57                     else:
58                         Y_train[al] = Y_train[al]-1
59                 num = int(clasif_F) + 1

```

```

57         X_train_NoNy, Y_train_NoNy = mil_cv_filter_ef(
X_train, Y_train, folds, votacion, num)
58 #         num = 2
59 #         X_train_NoNy2, Y_train_NoNy2 = mil_cv_filter_ef
(X_train, Y_train, folds, votacion, num)
60 #         print('\t\t\t=>Original')
61         results_Ori.append(filtrado_final(X_train,
Y_train, X_test, Y_test, Clasificadores))
62 #         print('\t\t\t=>Filtrado')
63         results_Fil.append(filtrado_final(X_train_NoNy,
Y_train_NoNy, X_test, Y_test, Clasificadores))
64 #         results_Fil2.append(filtrado_final(
X_train_NoNy2, Y_train_NoNy2, X_test, Y_test))
65         fold = fold + 1
66         Clasificadores_filtro = ""
67         if clasif_F == "0":
68             Clasificadores_filtro = cla_filter()
69         elif clasif_F == "1":
70             Clasificadores_filtro = cla_filter2()
71         results_accuracie_F = []
72         results_auc_F = []
73 #         results_accuracie_F2 = []
74 #         results_auc_F2 = []
75         results_accuracie_0 = []
76         results_auc_0 = []
77         for h in range(0, len(Clasificadores)):
78             print('\t\t\t\t-->Clasificador :'+str(
Clasificadores[h][2]))
79             data['EF'].append(str(Clasificadores[h][2]))
80             for g in range(0, folds):
81                 results_accuracie_F.append(results_Fil[g][h
] [0])
82                 results_auc_F.append(results_Fil[g][h][1])
83 #                 results_accuracie_F2.append(results_Fil2[g
] [h][0])
84 #                 results_auc_F2.append(results_Fil2[g][h
] [1])
85                 results_accuracie_0.append(results_Ori[g][h
] [0])
86                 results_auc_0.append(results_Ori[g][h][1])
87             print('\t\t\t\t-->Original')
88             print('\t\t\t\t Precision: ' + str(np.mean(
results_accuracie_0))+ '%')
89             data['Original'].append(np.mean(
results_accuracie_0))
90             print('\t\t\t\t\t Roc Score: ' + str(np.mean(
results_auc_0)))
91             print('\t\t\t\t-->Filtrado por ')
92             for h, cl_f0 in enumerate(Clasificadores_filtro)
:
93                 print('\t\t\t\t\t * '+str(cl_f0[2]))
94 #                 print('\t\t\t\t-->Filtrado')
95                 print('\t\t\t\t Precision: ' + str(np.mean(
results_accuracie_F))+ '%')

```

```

96         data['Filtro_1'].append(np.mean(
97             results_accuracie_F))
98         print('\t\t\t\t\t Roc Score: ' + str(np.mean(
99             results_auc_F)))
100     #         for h,cl_f0 in enumerate(
101         #             Clasificadores_filtro2):
102         #             print('\t\t\t\t\t * '+str(cl_f0[2]))
103         #             print('\t\t\t\t\t-->Filtrado')
104         #             print('\t\t\t\t\t Precision: ' + str(np.mean(
105                 results_accuracie_F2))+ '%')
106         #             data['Filtro_2'].append(np.mean(
107                 results_accuracie_F2))
108         #             print('\t\t\t\t\t Roc Score: ' + str(np.mean(
109                 results_auc_F2)))
110     df = pd.DataFrame(data)
111     df.to_csv(file_data, sep=';')
112
113 def roc_auc_score_FIXED(y_true, y_pred):
114     if len(np.unique(y_true)) == 1 or len(np.unique(y_true)) ==
115         0: # bug in roc_auc_score
116         return accuracy_score(y_true, np.rint(y_pred))
117     return roc_auc_score(y_true, y_pred)
118
119 def Porcentaje(X,Y):
120     return X*Y/100
121
122 def clasif():
123     aux = []
124     resul1 = [[],[],[],[],[],[],[]]
125     resul2 = [[],[],[],[],[],[],[]]
126     resul3 = [[],[],[],[],[],[],[]]
127     resul4 = [[],[],[],[],[],[],[]]
128     resul5 = [[],[],[],[],[],[],[]]
129     resul6 = [[],[],[],[],[],[],[]]
130     resul7 = [[],[],[],[],[],[],[]]
131     resul8 = [[],[],[],[],[],[],[]]
132     roc_m_1 = [[],[],[],[],[],[],[]]
133     roc_m_2 = [[],[],[],[],[],[],[]]
134     roc_m_3 = [[],[],[],[],[],[],[]]
135     roc_m_4 = [[],[],[],[],[],[],[]]
136     roc_m_5 = [[],[],[],[],[],[],[]]
137     roc_m_6 = [[],[],[],[],[],[],[]]
138     roc_m_7 = [[],[],[],[],[],[],[]]
139     roc_m_8 = [[],[],[],[],[],[],[]]
140     SMILaMax = [simpleMIL(),{'type': 'max'},'MIL max',resul1,
141                 roc_m_1]
142     SMILaMin = [simpleMIL(),{'type': 'min'},'MIL min',resul2,
143                 roc_m_2]
144     SMILaExt = [simpleMIL(),{'type': 'extreme'},'MIL Extreme',
145                 resul3,roc_m_3]
146     BOW_clas = [BOW(),{'k':90,'covar_type':'diag','n_iter':20},
147                 'BOW',resul4,roc_m_4]

```

```

138     CKNN_cla = [CKNN(),{'references': 3, 'citters': 5},'CKNN',
139                 resul5,roc_m_5]
139     maxDD_cl = [maxDD(),{'},'DIVERSE DENSITY',resul6,roc_m_6]
140     EMDD_cla = [EMDD(),{'},'EM-DD',resul7,roc_m_7]
141     MILB_cla = [MILBoost(),{'},'MILBOOST',resul8,roc_m_8]
142     aux.append(SMILaMax)
143     aux.append(SMILaMin)
144     aux.append(SMILaExt)
145     aux.append(BOW_clas)
146     aux.append(CKNN_cla)
147     aux.append(maxDD_cl)
148     aux.append(EMDD_cla)
149     aux.append(MILB_cla)
150     return aux
151
152 def mil_cv_filter_ef(bags_f,labels_f,folds,votacion,num):
153     # print('\t\t\tFiltrando...')
154     if num == 1:
155         Clasificadores = cla_filter()
156     else:
157         Clasificadores = cla_filter2()
158     bags_f,labels_f = shuffle(bags_f, labels_f, random_state=
159                               rand.randint(0, 100))
159     if len(labels_f) < folds:
160         folds = len(labels_f)
161     skf = StratifiedKFold(n_splits=folds)
162     isCorrectLabel = np.ones((len(Clasificadores), len(labels_f
163 ))), dtype=bool)
163     for train_index, test_index in skf.split(bags_f, labels_f.
164       reshape(len(labels_f))):
164         X_train = [bags_f[i] for i in train_index]
165         Y_train = labels_f[train_index]
166         X_test = [bags_f[i] for i in test_index]
167         Y_test = labels_f[test_index]
168         for s,cl in enumerate(Clasificadores):
169
170             try:
171                 if len(Clasificadores[s][1]) > 0:
172                     Clasificadores[s][0].fit(X_train, Y_train,
173 **Clasificadores[s][1])
174                 else:
175                     Clasificadores[s][0].fit(bags_f, labels_f)
176                     predictions = Clasificadores[s][0].predict(
177 X_test)
178                     if (isinstance(predictions, tuple)):
179                         predictions = predictions[0]
180             except:
181                 print('Fallo, segundo intento')
182                 try:
183                     if len(Clasificadores[s][1]) > 0:
184                         Clasificadores[s][0].fit(X_train,
185 Y_train, **Clasificadores[s][1])
186                     else:

```



```

184         Clasificadores[s][0].fit(bags_f,
labels_f)
185         predictions = Clasificadores[s][0].predict(
X_test)
186         if (isinstance(predictions, tuple)):
187             predictions = predictions[0]
188             print('OK')
189         except:
190             print('Posible fallo en bolsa...')
191             try:
192                 if len(Clasificadores[s][1]) > 0:
193                     Clasificadores[s][0].fit(X_train,
Y_train, **Clasificadores[s][1])
194             else:
195                 Clasificadores[s][0].fit(X_train,
Y_train)
196         predictions = Clasificadores[s][0].
predict(X_test)
197         if (isinstance(predictions, tuple)):
198             predictions = predictions[0]
199             print('OK')
200         except:
201             try:
202                 print('Cambiano clasificador..')
203                 Cla_error = simpleMIL()
204                 par_error = {'type': 'max'}
205                 if len(par_error) > 0:
206                     Cla_error.fit(X_train, Y_train,
**par_error)
207             else:
208                 Cla_error.fit(X_train, Y_train)
209                 predictions = Cla_error.predict(
X_train)
210             if (isinstance(predictions, tuple)):
211                 predictions = predictions[0]
212                 print('OK')
213             except:
214                 print('Fallo')
215         for l,p in enumerate(test_index):
216             try:
217                 isCorrectLabel[s][p] = (Y_test.T[0][l] ==
np.sign(predictions[l]))
218             except IndexError:
219                 print("Fallo en ultimo indice!")
220         if votacion == 'maxVotos':
221             noisyBags = []
222             for n in range(0,len(labels_f)):
223                 aux = 0
224                 for m in range(0,len(Clasificadores)):
225                     if not isCorrectLabel[m][n]:
226                         aux = aux+1
227                 if aux > len(Clasificadores)/2:
228                     noisyBags.append(n)

```

```

229     if votacion == 'consenso':
230         noisyBags = []
231         for n in range(0, len(labels_f)):
232             aux = True
233             for m in range(0, len(Clasificadores)):
234                 if aux:
235                     if isCorrectLabel[m][n]:
236                         aux = False
237             if aux:
238                 noisyBags.append(n)
239     nonNoisyBags = []
240     cont = 0
241     if len(noisyBags) == 0:
242         for z in range(0, len(bags_f)):
243             nonNoisyBags.append(z)
244     else:
245         for z in range(0, len(bags_f)):
246             if cont < len(noisyBags) and noisyBags[cont] == z:
247                 cont = cont + 1
248             else:
249                 nonNoisyBags.append(z)
250     print('\t\t\t=>Elementos eliminados con Filter '+str(num +
251         1)+' ': '+str(len(noisyBags)))
252     X_train_NoNy = [bags_f[i] for i in nonNoisyBags]
253     Y_train_NoNy = labels_f[nonNoisyBags]
254     return X_train_NoNy, Y_train_NoNy
255
256 def filtrado_final(X_train, Y_train, X_test, Y_test, cl_fil):
257     Clasificadores = cl_fil
258     results = np.zeros((len(Clasificadores), 2))
259     aux_lab = True
260     if len(np.unique(Y_train)) == 1:
261         if Y_train[0] == 0:
262             for b in range(0, len(Y_test)):
263                 if aux_lab:
264                     if Y_test[b] == 1:
265                         aux_Y_train = copy.copy(Y_test[b])
266                         aux_X_train = copy.copy(X_test[b])
267                         Y_test[b] = copy.copy(Y_train[0])
268                         X_test[b] = copy.copy(X_train[0])
269                         Y_train[0] = copy.copy(aux_Y_train)
270                         X_train[0] = copy.copy(aux_X_train)
271                         aux_lab = False
272         else:
273             for b in range(0, len(Y_test)):
274                 if aux_lab:
275                     if Y_test[b] == 0:
276                         aux_Y_train = copy.copy(Y_test[b])
277                         aux_X_train = copy.copy(X_test[b])
278                         Y_test[b] = copy.copy(Y_train[0])
279                         X_test[b] = copy.copy(X_train[0])
280                         Y_train[0] = copy.copy(aux_Y_train)
281                         X_train[0] = copy.copy(aux_X_train)
282                         aux_lab = False

```

```

282     for s,cl in enumerate(Clasificadores):
283 #         print('\t\t\t\t-->Clasificador :'+str(cl[2]))
284         try:
285             if len(Clasificadores[s][1]) > 0:
286                 Clasificadores[s][0].fit(X_train, Y_train, **
Clasificadores[s][1])
287             else:
288                 Clasificadores[s][0].fit(X_train, Y_train)
289                 predictions = Clasificadores[s][0].predict(X_test)
290                 if (isinstance(predictions, tuple)):
291                     predictions = predictions[0]
292                 accuracie = (100 * np.average(Y_test.T == np.sign(
predictions)))
293                 auc_score = (100 * roc_auc_score_FIXED(Y_test,
predictions))
294             except:
295                 print('Fallo, segundo intento')
296
297             try:
298                 if len(Clasificadores[s][1]) > 0:
299                     Clasificadores[s][0].fit(X_train, Y_train,
**Clasificadores[s][1])
300                 else:
301                     Clasificadores[s][0].fit(X_train, Y_train)
302                     predictions = Clasificadores[s][0].predict(
X_test)
303                     if (isinstance(predictions, tuple)):
304                         predictions = predictions[0]
305                     accuracie = (100 * np.average(Y_test.T == np.
sign(predictions)))
306                     auc_score = (100 * roc_auc_score_FIXED(Y_test,
predictions))
307                     print('OK')
308                 except:
309                     print('Fallo en calculo')
310                 results[s][0] = accuracie
311                 results[s][1] = auc_score
312 #         print('\t\t\t\t\t Precisión: '+ str(accuracie)+'%\n\t\t\t\t\t Roc Score: '+ str(auc_score))
313     return results
314
315 def cla_filter():
316     aux = []
317     resul1 = [[],[],[],[],[],[],[]]
318     resul2 = [[],[],[],[],[],[],[]]
319     resul3 = [[],[],[],[],[],[],[]]
320     resul4 = [[],[],[],[],[],[],[]]
321     resul5 = [[],[],[],[],[],[],[]]
322     resul6 = [[],[],[],[],[],[],[]]
323     resul7 = [[],[],[],[],[],[],[]]
324     resul8 = [[],[],[],[],[],[],[]]
325     roc_m_1 = [[],[],[],[],[],[],[]]
326     roc_m_2 = [[],[],[],[],[],[],[]]
327     roc_m_3 = [[],[],[],[],[],[],[]]

```

```

328     roc_m_4 = [[], [], [], [], [], [], []]
329     roc_m_5 = [[], [], [], [], [], [], []]
330     roc_m_6 = [[], [], [], [], [], [], []]
331     roc_m_7 = [[], [], [], [], [], [], []]
332     roc_m_8 = [[], [], [], [], [], [], []]
333     SMILaMax = [simpleMIL(), {'type': 'max'}, 'MIL max', resul1,
roc_m_1]
334     SMILaMin = [simpleMIL(), {'type': 'min'}, 'MIL min', resul2,
roc_m_2]
335     SMILaExt = [simpleMIL(), {'type': 'extreme'}, 'MIL Extreme',
resul3, roc_m_3]
336     BOW_clas = [BOW(), {'k': 90, 'covar_type': 'diag', 'n_iter': 20},
'BOW', resul4, roc_m_4]
337     CKNN_cla = [CKNN(), {'references': 3, 'citers': 5}, 'CKNN',
resul5, roc_m_5]
338     maxDD_cl = [maxDD(), {}, 'DIVERSE DENSITY', resul6, roc_m_6]
339     EMDD_cla = [EMDD(), {}, 'EM-DD', resul7, roc_m_7]
340     MILB_cla = [MILBoost(), {}, 'MILBOOST', resul8, roc_m_8]
341     #     aux.append(SMILaMax)
342     #     aux.append(SMILaMin)
343     #     aux.append(SMILaExt)
344     aux.append(BOW_clas)
345     aux.append(CKNN_cla)
346     #     aux.append(maxDD_cl)
347     aux.append(EMDD_cla)
348     #     aux.append(MILB_cla)
349     return aux
350
351 def cla_filter2():
352     aux = []
353     resul1 = [[], [], [], [], [], [], []]
354     resul2 = [[], [], [], [], [], [], []]
355     resul3 = [[], [], [], [], [], [], []]
356     resul4 = [[], [], [], [], [], [], []]
357     resul5 = [[], [], [], [], [], [], []]
358     resul6 = [[], [], [], [], [], [], []]
359     resul7 = [[], [], [], [], [], [], []]
360     resul8 = [[], [], [], [], [], [], []]
361     roc_m_1 = [[], [], [], [], [], [], []]
362     roc_m_2 = [[], [], [], [], [], [], []]
363     roc_m_3 = [[], [], [], [], [], [], []]
364     roc_m_4 = [[], [], [], [], [], [], []]
365     roc_m_5 = [[], [], [], [], [], [], []]
366     roc_m_6 = [[], [], [], [], [], [], []]
367     roc_m_7 = [[], [], [], [], [], [], []]
368     roc_m_8 = [[], [], [], [], [], [], []]
369     SMILaMax = [simpleMIL(), {'type': 'max'}, 'MIL max', resul1,
roc_m_1]
370     SMILaMin = [simpleMIL(), {'type': 'min'}, 'MIL min', resul2,
roc_m_2]
371     SMILaExt = [simpleMIL(), {'type': 'extreme'}, 'MIL Extreme',
resul3, roc_m_3]
372     BOW_clas = [BOW(), {'k': 90, 'covar_type': 'diag', 'n_iter': 20},
'BOW', resul4, roc_m_4]

```

```
373     CKNN_cla = [CKNN(),{'references': 3, 'citters': 5},'CKNN',
374               resul5,roc_m_5]
374     maxDD_cl = [maxDD(),{'','DIVERSE DENSITY',resul6,roc_m_6]
375     EMDD_cla = [EMDD(),{'','EM-DD',resul7,roc_m_7]
376     MILB_cla = [MILBoost(),{'','MILBOOST',resul8,roc_m_8]
377     aux.append(SMILaMax)
378     #     aux.append(SMILaMin)
379     aux.append(SMILaExt)
380     #     aux.append(BOW_clas)
381     #     aux.append(CKNN_cla)
382     aux.append(maxDD_cl)
383     #     aux.append(EMDD_cla)
384     #     aux.append(MILB_cla)
385     return aux
```

B.3. Módulo MIL-CVCF

```

1  import warnings,copy
2  from MILpy.data.load_data import load_data
3  import random as rand
4  import numpy as np
5  import pandas as pd
6  from sklearn.utils import shuffle
7  from sklearn.model_selection import StratifiedKFold
8  warnings.filterwarnings('ignore')
9  from sklearn.metrics import roc_auc_score, accuracy_score
10 #from funciones import fun_aux
11 #Import Algorithms
12 from MILpy.Algorithms.simpleMIL import simpleMIL
13 from MILpy.Algorithms.MILBoost import MILBoost
14 from MILpy.Algorithms.maxDD import maxDD
15 from MILpy.Algorithms.CKNN import CKNN
16 from MILpy.Algorithms.EMDD import EMDD
17 from MILpy.Algorithms.BOW import BOW
18
19 def CVcF(b,votacion,folds,ruido,clasif_0,clasif_F):
20     for DataSet in b:
21         bags,labels,X = load_data(DataSet)
22         bags,labels = shuffle(bags, labels, random_state=rand.
23                               randint(0, len(labels)-1))
24         skf = StratifiedKFold(n_splits=folds)
25         # dataAcc = np.zeros((len(b),len(ruido),folds,2))
26         print('\n\tDATASET: '+str(DataSet)+'\n')
27
28         for ny,k in enumerate(ruido):
29             print('\t\t=>RUIDO : '+str(k))
30             file_data = '../filtersApp/tabla.csv'
31             data = {}
32             data['CVCF'] = []
33             data['Original'] = []
34
35             Clasificadores_fake = clasif()
36             Clasificadores = []
37             for s,cl in enumerate(Clasificadores_fake):
38                 if str(cl[2]) == clasif_0:
39                     Clasificadores.append(cl)
40
41             Clasificadores_fake2 = cla_filter_cvcf()
42             Clasificadores_filtro = []
43             for s,cl in enumerate(Clasificadores_fake2):
44                 if str(cl[2]) == clasif_F:
45                     Clasificadores_filtro.append(cl)
46             # Clasificadores_filtro = cla_filter_cvcf()
47             for h,cl_f0 in enumerate(Clasificadores_filtro):
48                 data[str(cl_f0[2])] = []
49
50             for s,cl in enumerate(Clasificadores):
51                 fold = 1

```

```

51 #         results_Fil = np.zeros((len(
52     Clasificadores_filtro), folds))
53     results_Fil = [[] for x in range(len(
54     Clasificadores_filtro))]
55     results_Ori = []
56     clasificador_ = Clasificadores[s]
57     for train_index, test_index in skf.split(bags,
58     labels.reshape(len(labels))):
59         #         print('\t\t\t=>FOLD : '+str(fold))
60         X_train = [bags[i] for i in train_index]
61         Y_train = labels[train_index]
62         X_test = [bags[i] for i in test_index]
63         Y_test = labels[test_index]
64         LabelToChange = Porcentaje(len(train_index)
65         ,k)
66         aleatorios = rand.sample(range(0, len(
67         train_index)), int(LabelToChange))
68         for al in aleatorios:
69             if Y_train[al] == 0:
70                 Y_train[al] = Y_train[al]+1
71             else:
72                 Y_train[al] = Y_train[al]-1
73         for j, cl_f in enumerate(
74         Clasificadores_filtro):
75             clasificador_f = Clasificadores_filtro[
76             j]
77             #         print('\t\t\t=>Filtrado con '+str(cl_f
78             [2]))
79             X_train_NoNy, Y_train_NoNy =
80             mil_cv_filter_cvcf(X_train, Y_train, folds, votacion,
81             clasificador_f)
82             results_Fil[j].append(filtrado_final(
83             X_train_NoNy, Y_train_NoNy, X_test, Y_test, clasificador_))
84             #         print(len(X_train_NoNy))
85             #         print(len(X_train))
86             #         print('\t\t\t=>Original ')
87             results_Ori.append(filtrado_final(X_train,
88             Y_train, X_test, Y_test, clasificador_))
89             fold = fold + 1
90             results_accuracie_0 = []
91             results_auc_0 = []
92             print('\t\t\t\t\t-->Clasificador :'+str(
93             clasificador_[2]))
94             data['CVCF'].append(str(clasificador_[2]))
95             for g in range(0, folds):
96                 results_accuracie_0.append(results_Ori[g
97                 ][0])
98                 results_auc_0.append(results_Ori[g][1])
99                 print('\t\t\t\t\t-->Original')
100                 print('\t\t\t\t\t Precision: ' + str(np.mean(
101                 results_accuracie_0, dtype=np.float64))+ '%')
102                 data['Original'].append(np.mean(
103                 results_accuracie_0))

```

```

89         print('\t\t\t\t\t Roc Score: ' + str(np.mean(
results_auc_0, dtype=np.float64)))
90
91         for h,cl_f0 in enumerate(Clasificadores_filtro)
:
92             results_accuracie_F = []
93             results_auc_F = []
94             print('\t\t\t\t\t-->Filtrado por ' +str(
cl_f0[2]))
95             for f in range(0,folds):
96                 results_accuracie_F.append(results_Fil[
h][f][0])
97                 results_auc_F.append(results_Fil[h][f
][1])
98                 print('\t\t\t\t\t Precision: ' + str(np.mean
(results_accuracie_F, dtype=np.float64))+ '%')
99                 data[str(cl_f0[2])].append(np.mean(
results_accuracie_F))
100                 print('\t\t\t\t\t Roc Score: ' + str(np.mean
(results_auc_F, dtype=np.float64)))
101                 df = pd.DataFrame(data)
102                 df.to_csv(file_data, sep=';')
103
104 def mil_cv_filter_cvcf(bags_f,labels_f,folds,votacion,
clasificador_):
105     # print('\t\t\tFiltrando...')
106     bags_f,labels_f = shuffle(bags_f, labels_f, random_state=
rand.randint(0,len(labels_f)-1))
107     if len(labels_f) < folds:
108         folds = len(labels_f)
109     skf = StratifiedKFold(n_splits=folds)
110     isCorrectLabel = np.ones((folds, len(labels_f)), dtype=bool
)
111     fold = 0
112     for train_index, test_index in skf.split(bags_f, labels_f.
reshape(len(labels_f))):
113         X_train = [bags_f[i] for i in train_index]
114         Y_train = labels_f[train_index]
115
116         try:
117             if len(clasificador_[1]) > 0:
118                 clasificador_[0].fit(X_train, Y_train, **
clasificador_[1])
119             else:
120                 clasificador_[0].fit(bags_f, labels_f)
121                 predictions = clasificador_[0].predict(X_train)
122                 if (isinstance(predictions, tuple)):
123                     predictions = predictions[0]
124             except:
125                 print('Fallo, segundo intento')
126                 try:
127                     if len(clasificador_[1]) > 0:
128                         clasificador_[0].fit(X_train, Y_train, **
clasificador_[1])

```



```

129         else:
130             clasificador_[0].fit(bags_f, labels_f)
131             predictions = clasificador_[0].predict(X_train)
132             if (isinstance(predictions, tuple)):
133                 predictions = predictions[0]
134             print('OK')
135         except:
136             print('Posible fallo en bolsa...')
137         try:
138             if len(clasificador_[1]) > 0:
139                 clasificador_[0].fit(X_train, Y_train,
**clasificador_[1])
140             else:
141                 clasificador_[0].fit(X_train, Y_train)
142                 predictions = clasificador_[0].predict(
X_train)
143                 if (isinstance(predictions, tuple)):
144                     predictions = predictions[0]
145                 print('OK')
146             except:
147                 try:
148                     print('Cambiano clasificador..')
149                     Cla_error = simpleMIL()
150                     par_error = {'type': 'max'}
151                     if len(par_error) > 0:
152                         Cla_error.fit(X_train, Y_train, **
par_error)
153                     else:
154                         Cla_error.fit(X_train, Y_train)
155                         predictions = Cla_error.predict(X_train
)
156                     if (isinstance(predictions, tuple)):
157                         predictions = predictions[0]
158                     print('OK')
159                 except:
160                     print('Fallo')
161             for l,p in enumerate(train_index):
162                 try:
163                     isCorrectLabel[fold][p] = (Y_train.T[0][l] ==
np.sign(predictions[l]))
164                 except IndexError:
165                     print("Fallo en ultimo indice!")
166
167             fold = fold + 1
168         if votacion == 'maxVotos':
169             noisyBags = []
170             for n in range(0, len(labels_f)):
171                 aux = 0
172                 for m in range(0, folds):
173                     if not isCorrectLabel[m][n]:
174                         aux = aux+1
175                 if aux > folds/2:
176                     noisyBags.append(n)
177         if votacion == 'consenso':

```

[illegible]

```

230             aux_lab = False
231     try:
232         if len(clasificador_[1]) > 0:
233             clasificador_[0].fit(X_train, Y_train, **
clasificador_[1])
234         else:
235             clasificador_[0].fit(X_train, Y_train)
236             predictions = clasificador_[0].predict(X_test)
237             if (isinstance(predictions, tuple)):
238                 predictions = predictions[0]
239             accuracie = (100 * np.average(Y_test.T == np.sign(
predictions)))
240             auc_score = (100 * roc_auc_score_FIXED(Y_test,
predictions))
241     except:
242         print('Fallo, segundo intento')
243
244     try:
245         if len(clasificador_[1]) > 0:
246             clasificador_[0].fit(X_train, Y_train, **
clasificador_[1])
247         else:
248             clasificador_[0].fit(X_train, Y_train)
249             predictions = clasificador_[0].predict(X_test)
250             if (isinstance(predictions, tuple)):
251                 predictions = predictions[0]
252             accuracie = (100 * np.average(Y_test.T == np.sign(
predictions)))
253             auc_score = (100 * roc_auc_score_FIXED(Y_test,
predictions))
254             print('OK')
255     except:
256         print('Fallo en calculo')
257     results[0] = accuracie
258     results[1] = auc_score
259     # print('\t\t\t\t\t Precisión: ' + str(accuracie) + '%\n\t\t\t\t\t
\t\t\t\t\t Roc Score: ' + str(auc_score))
260     return results
261
262 def roc_auc_score_FIXED(y_true, y_pred):
263     if len(np.unique(y_true)) == 1 or len(np.unique(y_true)) ==
0: # bug in roc_auc_score
264         return accuracy_score(y_true, np.rint(y_pred))
265     return roc_auc_score(y_true, y_pred)
266
267 def Porcentaje(X,Y):
268     return X*Y/100
269
270 def clasif():
271     aux = []
272     resul1 = [[], [], [], [], [], [], []]
273     resul2 = [[], [], [], [], [], [], []]
274     resul3 = [[], [], [], [], [], [], []]
275     resul4 = [[], [], [], [], [], [], []]

```

```

276     resul5 = [[], [], [], [], [], [], []]
277     resul6 = [[], [], [], [], [], [], []]
278     resul7 = [[], [], [], [], [], [], []]
279     resul8 = [[], [], [], [], [], [], []]
280     roc_m_1 = [[], [], [], [], [], [], []]
281     roc_m_2 = [[], [], [], [], [], [], []]
282     roc_m_3 = [[], [], [], [], [], [], []]
283     roc_m_4 = [[], [], [], [], [], [], []]
284     roc_m_5 = [[], [], [], [], [], [], []]
285     roc_m_6 = [[], [], [], [], [], [], []]
286     roc_m_7 = [[], [], [], [], [], [], []]
287     roc_m_8 = [[], [], [], [], [], [], []]
288     SMILaMax = [simpleMIL(), {'type': 'max'}, 'MIL max', resul1,
roc_m_1]
289     SMILaMin = [simpleMIL(), {'type': 'min'}, 'MIL min', resul2,
roc_m_2]
290     SMILaExt = [simpleMIL(), {'type': 'extreme'}, 'MIL Extreme',
resul3, roc_m_3]
291     BOW_clas = [BOW(), {'k': 90, 'covar_type': 'diag', 'n_iter': 20},
'BOW', resul4, roc_m_4]
292     CKNN_cla = [CKNN(), {'references': 3, 'citters': 5}, 'CKNN',
resul5, roc_m_5]
293     maxDD_cl = [maxDD(), {}, 'DIVERSE DENSITY', resul6, roc_m_6]
294     EMDD_cla = [EMDD(), {}, 'EM-DD', resul7, roc_m_7]
295     MILB_cla = [MILBoost(), {}, 'MILBOOST', resul8, roc_m_8]
296     aux.append(SMILaMax)
297     aux.append(SMILaMin)
298     aux.append(SMILaExt)
299     aux.append(BOW_clas)
300     aux.append(CKNN_cla)
301     aux.append(maxDD_cl)
302     aux.append(EMDD_cla)
303     aux.append(MILB_cla)
304     return aux
305
306 def cla_filter_cvcf():
307     aux = []
308     resul1 = [[], [], [], [], [], [], []]
309     resul2 = [[], [], [], [], [], [], []]
310     resul3 = [[], [], [], [], [], [], []]
311     resul4 = [[], [], [], [], [], [], []]
312     resul5 = [[], [], [], [], [], [], []]
313     resul6 = [[], [], [], [], [], [], []]
314     resul7 = [[], [], [], [], [], [], []]
315     resul8 = [[], [], [], [], [], [], []]
316     roc_m_1 = [[], [], [], [], [], [], []]
317     roc_m_2 = [[], [], [], [], [], [], []]
318     roc_m_3 = [[], [], [], [], [], [], []]
319     roc_m_4 = [[], [], [], [], [], [], []]
320     roc_m_5 = [[], [], [], [], [], [], []]
321     roc_m_6 = [[], [], [], [], [], [], []]
322     roc_m_7 = [[], [], [], [], [], [], []]
323     roc_m_8 = [[], [], [], [], [], [], []]

```

```
324     SMILaMax = [simpleMIL(),{'type': 'max'},'MIL max',resul1,
325     roc_m_1]
326     SMILaMin = [simpleMIL(),{'type': 'min'},'MIL min',resul2,
327     roc_m_2]
328     SMILaExt = [simpleMIL(),{'type': 'extreme'},'MIL Extreme',
329     resul3,roc_m_3]
330     BOW_clas = [BOW(),{'k':90,'covar_type':'diag','n_iter':20},
331     'BOW',resul4,roc_m_4]
332     CKNN_cla = [CKNN(),{'references': 3, 'citters': 5},'CKNN',
333     resul5,roc_m_5]
334     maxDD_cl = [maxDD(),{},{},'DIVERSE DENSITY',resul6,roc_m_6]
335     EMDD_cla = [EMDD(),{},{},'EM-DD',resul7,roc_m_7]
336     MILB_cla = [MILBoost(),{},{},'MILBOOST',resul8,roc_m_8]
337     aux.append(SMILaMax)
338     aux.append(SMILaMin)
339     aux.append(SMILaExt)
340     aux.append(BOW_clas)
341     aux.append(CKNN_cla)
342     aux.append(maxDD_cl)
343     aux.append(EMDD_cla)
344     aux.append(MILB_cla)
345     return aux
```

B.4. Módulo MIL-IPF

```

1  import warnings,copy
2  from MILpy.data.load_data import load_data
3  import random as rand
4  import numpy as np
5  import pandas as pd
6  from sklearn.utils import shuffle
7  from sklearn.model_selection import StratifiedKFold
8  warnings.filterwarnings('ignore')
9  from sklearn.metrics import roc_auc_score, accuracy_score
10 #from funciones import fun_aux
11 #Import Algorithms
12 from MILpy.Algorithms.simpleMIL import simpleMIL
13 from MILpy.Algorithms.MILBoost import MILBoost
14 from MILpy.Algorithms.maxDD import maxDD
15 from MILpy.Algorithms.CKNN import CKNN
16 from MILpy.Algorithms.EMDD import EMDD
17 from MILpy.Algorithms.MILES import MILES
18 from MILpy.Algorithms.BOW import BOW
19
20 def IPF(b,votacion,folds,ruido,clasif_0,clasif_F):
21     for DataSet in b:
22         bags,labels,X = load_data(DataSet)
23         bags,labels = shuffle(bags, labels, random_state=rand.
24                               randint(0, len(labels)-1))
25         skf = StratifiedKFold(n_splits=folds)
26         # dataAcc = np.zeros((len(b),len(ruido),folds,2))
27         print('\n\tDATASET: '+str(DataSet)+'\n')
28
29         for ny,k in enumerate(ruido):
30             print('\t\t=>RUIDO : '+str(k))
31             file_data = '../filtersApp/tabla.csv'
32             data = {}
33             data['IPF'] = []
34             data['Original'] = []
35
36             Clasificadores_fake = clasif()
37             Clasificadores = []
38             for s,cl in enumerate(Clasificadores_fake):
39                 if str(cl[2]) == clasif_0:
40                     Clasificadores.append(cl)
41
42             Clasificadores_fake2 = cla_filter_ipf()
43             Clasificadores_filtro = []
44             for s,cl in enumerate(Clasificadores_fake2):
45                 if str(cl[2]) == clasif_F:
46                     Clasificadores_filtro.append(cl)
47             for h,cl_f0 in enumerate(Clasificadores_filtro):
48                 data[str(cl_f0[2])] = []
49
50             for s,cl in enumerate(Clasificadores):
51                 fold = 1

```

```

51 #         results_Fil = np.zeros((len(
52     Clasificadores_filtro), folds))
53     results_Fil = [[] for x in range(len(
54     Clasificadores_filtro))]
55     results_Ori = []
56     clasificador_ = Clasificadores[s]
57     for train_index, test_index in skf.split(bags,
58     labels.reshape(len(labels))):
59         #         print('\t\t\t=>FOLD : '+str(fold))
60         X_train = [bags[i] for i in train_index]
61         Y_train = labels[train_index]
62         X_test = [bags[i] for i in test_index]
63         Y_test = labels[test_index]
64         LabelToChange = Porcentaje(len(train_index)
65         ,k)
66         aleatorios = rand.sample(range(0, len(
67         train_index)), int(LabelToChange))
68         for al in aleatorios:
69             if Y_train[al] == 0:
70                 Y_train[al] = Y_train[al]+1
71             else:
72                 Y_train[al] = Y_train[al]-1
73         for j, cl_f in enumerate(
74         Clasificadores_filtro):
75             clasificador_f = Clasificadores_filtro[
76             j]
77             #         print('\t\t\t=>Filtrado con '+str(cl_f
78             [2]))
79             X_train_NoNy, Y_train_NoNy =
80             mil_cv_filter_ipf(X_train, Y_train, folds, votacion,
81             clasificador_f)
82             results_Fil[j].append(filtrado_final(
83             X_train_NoNy, Y_train_NoNy, X_test, Y_test, clasificador_))
84             #         print(len(X_train_NoNy))
85             #         print(len(X_train))
86             #         print('\t\t\t=>Original ')
87             results_Ori.append(filtrado_final(X_train,
88             Y_train, X_test, Y_test, clasificador_))
89             fold = fold + 1
90             results_accuracie_0 = []
91             results_auc_0 = []
92             print('\t\t\t\t\t-->Clasificador :'+str(
93             clasificador_[2]))
94             data['IPF'].append(str(clasificador_[2]))
95             for g in range(0, folds):
96                 results_accuracie_0.append(results_Ori[g
97                 ][0])
98                 results_auc_0.append(results_Ori[g][1])
99             print('\t\t\t\t\t-->Original')
100             print('\t\t\t\t\t Precision: ' + str(np.mean(
101             results_accuracie_0, dtype=np.float64))+ '%')
102             data['Original'].append(np.mean(
103             results_accuracie_0))

```

```

89         print('\t\t\t\t\t Roc Score: ' + str(np.mean(
results_auc_0, dtype=np.float64)))
90
91         for h,cl_f0 in enumerate(Clasificadores_filtro)
:
92             results_accuracie_F = []
93             results_auc_F = []
94             print('\t\t\t\t\t-->Filtrado por ' +str(
cl_f0[2]))
95             for f in range(0,folds):
96                 results_accuracie_F.append(results_Fil[
h][f][0])
97                 results_auc_F.append(results_Fil[h][f
][1])
98                 print('\t\t\t\t\t Precision: ' + str(np.mean
(results_accuracie_F, dtype=np.float64))+ '%')
99                 data[str(cl_f0[2])].append(np.mean(
results_accuracie_F))
100                 print('\t\t\t\t\t Roc Score: ' + str(np.mean
(results_auc_F, dtype=np.float64)))
101                 df = pd.DataFrame(data)
102                 df.to_csv(file_data, sep=';')
103
104 def mil_cv_filter_ipf(bags_f,labels_f,folds,votacion,
clasificador_):
105     # print('\t\t\t\tFiltrando...')
106     error = 0.01
107     toStop = 3
108     stop = True
109     countToStop = 0
110     vuelta = 0
111     if len(labels_f) < folds:
112         folds = len(labels_f)
113     skf = StratifiedKFold(n_splits=folds)
114
115     while stop:
116         bags_f,labels_f = shuffle(bags_f, labels_f,
random_state=rand.randint(0, len(labels_f)-1))
117         isCorrectLabel = np.ones((folds, len(labels_f)), dtype=
bool)
118         fold = 0
119
120         for train_index, test_index in skf.split(bags_f,
labels_f.reshape(len(labels_f))):
121             X_train = [bags_f[i] for i in train_index]
122             Y_train = labels_f[train_index]
123             # print('\t\t\t\t=>FOLD : ' +str(fold))
124             try:
125                 if len(clasificador_[1]) > 0:
126                     clasificador_[0].fit(X_train, Y_train, **
clasificador_[1])
127                 else:
128                     clasificador_[0].fit(bags_f, labels_f)
129                     predictions = clasificador_[0].predict(X_train)

```



```

130         if (isinstance(predictions, tuple)):
131             predictions = predictions[0]
132     except:
133         print('Fallo, segundo intento')
134     try:
135         if len(clasificador_[1]) > 0:
136             clasificador_[0].fit(X_train, Y_train,
**clasificador_[1])
137         else:
138             clasificador_[0].fit(bags_f, labels_f)
139             predictions = clasificador_[0].predict(
X_train)
140         if (isinstance(predictions, tuple)):
141             predictions = predictions[0]
142         print('OK')
143     except:
144         print('Posible fallo en bolsa...')
145     try:
146         print('Cambiano clasificador..')
147         Cla_error = simpleMIL()
148         par_error = {'type': 'max'}
149         if len(par_error) > 0:
150             Cla_error.fit(X_train, Y_train, **
par_error)
151         else:
152             Cla_error.fit(X_train, Y_train)
153             predictions = Cla_error.predict(X_train
)
154         if (isinstance(predictions, tuple)):
155             predictions = predictions[0]
156         print('OK')
157     except:
158         print('Fallo')
159     for l,p in enumerate(train_index):
160         try:
161             isCorrectLabel[fold][p] = (Y_train.T[0][l]
== np.sign(predictions[l]))
162         except IndexError:
163             print("Fallo en ultimo indice!")
164             fold = fold + 1
165     if votacion == 'maxVotos':
166         noisyBags = []
167         for n in range(0,len(labels_f)):
168             aux = 0
169             for m in range(0,folds):
170                 if not isCorrectLabel[m][n]:
171                     aux = aux+1
172             if aux > folds/2:
173                 noisyBags.append(n)
174     if votacion == 'consenso':
175         noisyBags = []
176         for n in range(0,len(labels_f)):
177             aux = True
178             for m in range(0,folds):

```

```

179             if aux:
180                 if isCorrectLabel[m][n]:
181                     aux = False
182             if aux:
183                 noisyBags.append(n)
184         nonNoisyBags = []
185         cont = 0
186         if len(noisyBags) == 0:
187             for z in range(0, len(bags_f)):
188                 nonNoisyBags.append(z)
189         else:
190             for z in range(0, len(bags_f)):
191                 if cont < len(noisyBags) and noisyBags[cont] ==
z:
192                     cont = cont + 1
193             else:
194                 nonNoisyBags.append(z)
195         if len(noisyBags) < (len(bags_f)*error):
196             countToStop = countToStop + 1
197         else:
198             countToStop = 0
199         if countToStop == toStop:
200             stop = False
201         else:
202             bags_f = [bags_f[d] for d in nonNoisyBags]
203             labels_f = labels_f[nonNoisyBags]
204
205         vuelta = vuelta + 1
206         print('\t\t\t=>Elementos eliminados por ' + clasificador_[2] +
': ' + str(len(noisyBags)))
207         X_train_NoNy = bags_f
208         Y_train_NoNy = labels_f
209         return X_train_NoNy, Y_train_NoNy
210
211 def filtrado_final(X_train, Y_train, X_test, Y_test, clasificador_)
:
212     results = np.zeros((2))
213     accuracie = 0
214     auc_score = 0
215     aux_lab = True
216     if len(np.unique(Y_train)) == 1:
217         if Y_train[0] == 0:
218             for b in range(0, len(Y_test)):
219                 if aux_lab:
220                     if Y_test[b] == 1:
221                         aux_Y_train = copy.copy(Y_test[b])
222                         aux_X_train = copy.copy(X_test[b])
223                         Y_test[b] = copy.copy(Y_train[0])
224                         X_test[b] = copy.copy(X_train[0])
225                         Y_train[0] = copy.copy(aux_Y_train)
226                         X_train[0] = copy.copy(aux_X_train)
227                         aux_lab = False
228                 else:
229                     for b in range(0, len(Y_test)):

```

```

230         if aux_lab:
231             if Y_test[b] == 0:
232                 aux_Y_train = copy.copy(Y_test[b])
233                 aux_X_train = copy.copy(X_test[b])
234                 Y_test[b] = copy.copy(Y_train[0])
235                 X_test[b] = copy.copy(X_train[0])
236                 Y_train[0] = copy.copy(aux_Y_train)
237                 X_train[0] = copy.copy(aux_X_train)
238                 aux_lab = False
239     try:
240         if len(clasificador_[1]) > 0:
241             clasificador_[0].fit(X_train, Y_train, **
clasificador_[1])
242         else:
243             clasificador_[0].fit(X_train, Y_train)
244             predictions = clasificador_[0].predict(X_test)
245             if (isinstance(predictions, tuple)):
246                 predictions = predictions[0]
247             accuracie = (100 * np.average(Y_test.T == np.sign(
predictions)))
248             auc_score = (100 * roc_auc_score_FIXED(Y_test,
predictions))
249     except:
250         print('Fallo, segundo intento')
251
252     try:
253         if len(clasificador_[1]) > 0:
254             clasificador_[0].fit(X_train, Y_train, **
clasificador_[1])
255         else:
256             clasificador_[0].fit(X_train, Y_train)
257             predictions = clasificador_[0].predict(X_test)
258             if (isinstance(predictions, tuple)):
259                 predictions = predictions[0]
260             accuracie = (100 * np.average(Y_test.T == np.sign(
predictions)))
261             auc_score = (100 * roc_auc_score_FIXED(Y_test,
predictions))
262             print('OK')
263     except:
264         print('Fallo en calculo')
265     results[0] = accuracie
266     results[1] = auc_score
267     # print('\t\t\t\t\t Precisión: ' + str(accuracie) + '%\n\t\t\t\t\t
\t\t Roc Score: ' + str(auc_score))
268     return results
269
270 def roc_auc_score_FIXED(y_true, y_pred):
271     if len(np.unique(y_true)) == 1 or len(np.unique(y_true)) ==
0: # bug in roc_auc_score
272         return accuracy_score(y_true, np.rint(y_pred))
273     return roc_auc_score(y_true, y_pred)
274
275 def Porcentaje(X,Y):

```

```

276         return X*Y/100
277
278     def clasif():
279         aux = []
280         resul1 = [[],[],[],[],[],[],[]]
281         resul2 = [[],[],[],[],[],[],[]]
282         resul3 = [[],[],[],[],[],[],[]]
283         resul4 = [[],[],[],[],[],[],[]]
284         resul5 = [[],[],[],[],[],[],[]]
285         resul6 = [[],[],[],[],[],[],[]]
286         resul7 = [[],[],[],[],[],[],[]]
287         resul8 = [[],[],[],[],[],[],[]]
288         resul9 = [[],[],[],[],[],[],[]]
289         roc_m_1 = [[],[],[],[],[],[],[]]
290         roc_m_2 = [[],[],[],[],[],[],[]]
291         roc_m_3 = [[],[],[],[],[],[],[]]
292         roc_m_4 = [[],[],[],[],[],[],[]]
293         roc_m_5 = [[],[],[],[],[],[],[]]
294         roc_m_6 = [[],[],[],[],[],[],[]]
295         roc_m_7 = [[],[],[],[],[],[],[]]
296         roc_m_8 = [[],[],[],[],[],[],[]]
297         roc_m_9 = [[],[],[],[],[],[],[]]
298         SMILaMax = [simpleMIL(),{'type': 'max'},'MIL max',resul1,
299                     roc_m_1]
300         SMILaMin = [simpleMIL(),{'type': 'min'},'MIL min',resul2,
301                     roc_m_2]
302         SMILaExt = [simpleMIL(),{'type': 'extreme'},'MIL Extreme',
303                     resul3,roc_m_3]
304         BOW_clas = [BOW(),{'k':90,'covar_type':'diag','n_iter':20},
305                     'BOW',resul4,roc_m_4]
306         CKNN_cla = [CKNN(),{'references': 3, 'citters': 5},'CKNN',
307                     resul5,roc_m_5]
308         maxDD_cl = [maxDD(),{},'DIVERSE DENSITY',resul6,roc_m_6]
309         EMDD_cla = [EMDD(),{},'EM-DD',resul7,roc_m_7]
310         MILB_cla = [MILBoost(),{},'MILBOOST',resul8,roc_m_8]
311         MILES_cl = [MILES(),{},'MILES',resul9,roc_m_9]
312         aux.append(SMILaMax)
313         aux.append(SMILaMin)
314         aux.append(SMILaExt)
315         aux.append(BOW_clas)
316         aux.append(CKNN_cla)
317         aux.append(maxDD_cl)
318         aux.append(EMDD_cla)
319         aux.append(MILB_cla)
320         # aux.append(MILES_cl)
321         return aux
322
323     def cla_filter_ipf():
324         aux = []
325         resul1 = [[],[],[],[],[],[],[]]
326         resul2 = [[],[],[],[],[],[],[]]
327         resul3 = [[],[],[],[],[],[],[]]
328         resul4 = [[],[],[],[],[],[],[]]
329         resul5 = [[],[],[],[],[],[],[]]

```

```

325     resul6 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
326     resul7 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
327     resul8 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
328     resul9 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
329     roc_m_1 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
330     roc_m_2 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
331     roc_m_3 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
332     roc_m_4 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
333     roc_m_5 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
334     roc_m_6 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
335     roc_m_7 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
336     roc_m_8 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
337     roc_m_9 = [[] , [[] , [[] , [[] , [[] , [[] , [[]
338     SMILaMax = [simpleMIL() , {'type': 'max'} , 'MIL max' , resul1 ,
roc_m_1]
339     SMILaMin = [simpleMIL() , {'type': 'min'} , 'MIL min' , resul2 ,
roc_m_2]
340     SMILaExt = [simpleMIL() , {'type': 'extreme'} , 'MIL Extreme' ,
resul3 , roc_m_3]
341     BOW_clas = [BOW() , {'k':90 , 'covar_type': 'diag' , 'n_iter':20} ,
'BOW' , resul4 , roc_m_4]
342     CKNN_cla = [CKNN() , {'references': 3 , 'citters': 5} , 'CKNN' ,
resul5 , roc_m_5]
343     maxDD_cl = [maxDD() , {} , 'DIVERSE DENSITY' , resul6 , roc_m_6]
344     EMDD_cla = [EMDD() , {} , 'EM-DD' , resul7 , roc_m_7]
345     MILB_cla = [MILBoost() , {} , 'MILBOOST' , resul8 , roc_m_8]
346     MILES_cl = [MILES() , {} , 'MILES' , resul9 , roc_m_9]
347     aux.append(SMILaMax)
348     aux.append(SMILaMin)
349     aux.append(SMILaExt)
350     aux.append(BOW_clas)
351     aux.append(CKNN_cla)
352     aux.append(maxDD_cl)
353     aux.append(EMDD_cla)
354     aux.append(MILB_cla)
355     #     aux.append(MILES_cl)
356     return aux

```

Anexos C

Resultados

Los resultados que se muestran a continuación se dividen en primer lugar por el Dataset utilizado, y dentro de esta sección se analizarán los resultados para cada uno de los filtros desarrollados en pyMIL-BNF subdivididos a su vez por el esquema de votación que tienen configurado, y por último cada tabla se muestra por el porcentaje de ruido artificial imputado en las etiquetas de las clases.

Como aclaración para el esquema MIL-EF, la nomenclatura 'Filtro_1' y 'Filtro_2' pertenecen a un grupo de clasificadores ya que este modelo agrupa 3 clasificadores diferentes en su iteración, a continuación diremos cuales:

- Filtro 1:
 - Clasificador 1: BOW.
 - Clasificador 2: CKNN.
 - Clasificador 3: EM-DD.
- Filtro 2:
 - Clasificador 1: SimpleMIL max.
 - Clasificador 2: SimpleMIL extreme.
 - Clasificador 3: maxDD.

C.1. Musk 1

MIL-EF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.1: Precisión MIL-EF Musk 1, Consenso, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	71.75438596491229	73.91812865497076	71.75438596491229
MIL min	77.63157894736841	78.71345029239765	77.07602339181287
MIL Ext	80.33138401559454	79.96101364522416	79.96101364522416
BOW	76.82748538011695	76.02339181286548	77.66081871345028
CKNN	78.61988304093566	78.4093567251462	79.49707602339181
maxDD	77.46588693957115	76.92007797270955	79.82456140350877
EM-DD	78.33751044277359	78.18713450292398	80.51796157059314
MILBoost	74.91305016928285	74.78147122191444	76.82094490612496

- Esquema votación por Max Votos.

Tabla C.2: Precisión MIL-EF Musk 1, Max Votos, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	81.6374269005848	72.92397660818713	78.42105263157896
MIL min	83.18713450292398	77.2514619883041	80.00000000000001
MIL Ext	84.50292397660819	79.78557504873295	80.54580896686161
BOW	80.17543859649123	76.09649122807016	74.79532163742691
CKNN	81.74269005847952	78.05847953216374	77.23976608187135
maxDD	80.9551656920078	77.35867446393762	77.05653021442497
EM-DD	81.81286549707602	78.1203007518797	78.01169590643273
MILBoost	77.95398584179748	74.72299168975069	74.62796244998461

- Ruido 5 %.
 - Esquema votación por consenso.

Tabla C.3: Precisión MIL-EF Musk 1, Consenso, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	71.9298245614035	76.3157894736842	74.03508771929826
MIL min	76.2280701754386	77.86549707602339	78.39181286549709
MIL Ext	79.12280701754385	80.60428849902534	79.08382066276802
BOW	72.9093567251462	75.96491228070174	74.80994152046785
CKNN	74.65497076023392	77.53216374269006	77.05263157894737
maxDD	74.16179337231968	77.67056530214425	76.91033138401559
EM-DD	74.77861319966583	78.07852965747703	77.87802840434422
MILBoost	71.79901508156357	74.68644198214835	74.51100338565712

- Esquema votación por Max Votos.

Tabla C.4: Precisión MIL-EF Musk 1, Max Votos, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	71.92982456140352	75.14619883040936	69.64912280701753
MIL min	78.42105263157896	76.2280701754386	76.75438596491229
MIL Ext	81.98830409356725	78.38206627680312	79.49317738791423
BOW	77.57309941520467	74.60526315789474	74.04970760233918
CKNN	79.23976608187134	76.42105263157895	76.19883040935672
maxDD	78.17738791423001	75.45808966861597	75.32163742690058
EM-DD	79.12280701754386	76.64160401002506	76.36591478696742
MILBoost	75.60018467220684	73.42913204062788	73.18790397045244

- Ruido 10 %.
 - Esquema votación por consenso.

Tabla C.5: Precisión MIL-EF Musk 1, Consenso, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	61.988304093567265	65.3216374269006	64.21052631578947
MIL min	69.12280701754386	70.26315789473685	70.23391812865498
MIL Ext	72.98245614035089	71.16959064327486	72.61208576998051
BOW	69.15204678362574	68.02631578947368	69.67836257309942
CKNN	70.76023391812865	69.41520467836257	71.82456140350877
maxDD	70.23391812865498	68.73294346978557	71.80311890838206
EM-DD	71.55388471177945	69.34001670843776	73.0409356725146
MILBoost	68.97737765466297	67.04024315173899	70.27854724530626

- Esquema votación por Max Votos.

Tabla C.6: Precisión MIL-EF Musk 1, Max Votos, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	74.97076023391813	68.42105263157895	72.7485380116959
MIL min	78.30409356725146	73.39181286549709	76.05263157894737
MIL Ext	80.81871345029239	74.60038986354776	77.83625730994153
BOW	76.37426900584798	68.74269005847952	74.13742690058479
CKNN	77.82456140350877	70.64327485380117	75.81286549707602
maxDD	77.36842105263158	69.775828460039	74.4346978557505
EM-DD	77.31829573934837	71.13617376775272	74.5112781954887
MILBoost	74.02123730378578	68.61188057863959	71.56509695290859

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.7: Precisión MIL-EF Musk 1, Consenso, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	67.48538011695905	66.37426900584796	64.21052631578948
MIL min	69.06432748538012	68.56725146198832	67.39766081871345
MIL Ext	71.40350877192982	71.11111111111111	69.92202729044834
BOW	69.2982456140351	69.61988304093568	67.14912280701755
CKNN	70.44444444444444	71.12280701754385	67.62573099415205
maxDD	71.41325536062378	70.89668615984404	69.03508771929823
EM-DD	73.01587301587301	72.54803675856306	70.49289891395154
MILBoost	70.25661742074485	69.84726069559864	68.04901508156355

- Esquema votación por Max Votos.

Tabla C.8: Precisión MIL-EF Musk 1, Max Votos, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	74.03508771929823	71.87134502923975	71.87134502923978
MIL min	74.44444444444443	73.30409356725144	75.49707602339181
MIL Ext	74.97076023391813	74.58089668615983	75.71150097465888
BOW	72.28070175438597	67.86549707602339	70.90643274853801
CKNN	72.80701754385964	69.50877192982458	72.57309941520467
maxDD	72.84600389863546	69.17153996101365	73.18713450292397
EM-DD	74.70342522974101	70.48454469507101	74.53634085213032
MILBoost	71.73322560787935	68.04170514004309	71.58702677746999

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.9: Precisión MIL-EF Musk 1, Consenso, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	72.80701754385964	77.13450292397661	72.86549707602339
MIL min	73.88888888888889	75.52631578947368	73.94736842105263
MIL Ext	74.61988304093568	76.43274853801171	73.93762183235869
BOW	69.31286549707603	70.89181286549707	69.2982456140351
CKNN	70.69005847953217	72.80701754385964	70.66666666666666
maxDD	71.2280701754386	73.53801169590642	70.69200779727095
EM-DD	71.77109440267336	73.90142021720969	71.15288220551378
MILBoost	69.16743613419513	71.03147122191443	68.62650046168051

- Esquema votación por Max Votos.

Tabla C.10: Precisión MIL-EF Musk 1, Max Votos, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	67.48538011695906	64.21052631578948	66.54970760233918
MIL min	69.70760233918129	65.90643274853802	68.09941520467837
MIL Ext	71.87134502923978	66.10136452241716	69.35672514619883
BOW	68.61111111111111	63.21637426900585	65.62865497076025
CKNN	70.54970760233918	65.40350877192982	67.73099415204678
maxDD	69.1325536062378	64.28849902534112	66.80311890838206
EM-DD	70.62656641604009	65.99832915622389	68.90559732664995
MILBoost	68.16597414589104	64.11626654355186	66.66012619267467

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.11: Precisión MIL-EF Musk 1, Consenso, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	68.88888888888889	74.15204678362572	70.93567251461988
MIL min	69.73684210526315	72.39766081871346	68.07017543859648
MIL Ext	71.50097465886938	73.29434697855751	70.74074074074073
BOW	68.59649122807016	70.2046783625731	68.5233918128655
CKNN	69.23976608187132	70.31578947368422	69.39181286549707
maxDD	68.74269005847951	68.0214424951267	67.5925925925926
EM-DD	69.49874686716792	68.23725981620719	68.36257309941521
MILBoost	67.17913204062788	66.07533087103724	66.18497999384427

- Esquema votación por Max Votos.

Tabla C.12: Precisión MIL-EF Musk 1, Max Votos, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	71.46198830409358	65.02923976608187	66.14035087719297
MIL min	69.38596491228068	65.64327485380116	66.25730994152046
MIL Ext	71.89083820662768	67.66081871345028	68.05068226120858
BOW	68.59649122807016	64.83918128654972	64.86842105263158
CKNN	69.21637426900584	66.60818713450293	66.21052631578948
maxDD	67.8167641325536	66.17933723196882	64.41520467836257
EM-DD	69.30659983291561	66.49122807017544	66.22389306599833
MILBoost	67.01100338565712	64.54755309325947	64.31363496460449

- Ruido 30 %.
- Esquema votación por consenso.

Tabla C.13: Precisión MIL-EF Musk 1, Consenso, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	58.71345029239766	58.83040935672515	57.77777777777777
MIL min	59.7953216374269	58.27485380116959	61.549707602339176
MIL Ext	61.6374269005848	59.55165692007798	63.216374269005854
BOW	60.3654970760234	57.73391812865498	61.30116959064328
CKNN	59.78947368421053	57.26315789473684	62.280701754385966
maxDD	61.9980506822612	57.524366471734886	62.96296296296296
EM-DD	64.18546365914786	58.63826232247285	64.55304928989139
MILBoost	62.53000923361035	57.67620806401969	62.85164666051092

- Esquema votación por Max Votos.

Tabla C.14: Precisión MIL-EF Musk 1, Max Votos, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	67.2514619883041	58.53801169590643	60.877192982456144
MIL min	66.22807017543859	56.95906432748537	63.07017543859648
MIL Ext	67.6803118908382	57.836257309941516	64.50292397660819
BOW	63.27485380116959	57.74853801169591	61.69590643274855
CKNN	62.99415204678362	59.02923976608187	62.8421052631579
maxDD	62.82651072124756	58.23586744639376	61.9785575048733
EM-DD	63.467000835421885	58.44611528822056	62.89891395154553
MILBoost	61.9013542628501	57.646968297937825	61.527735029581756

MIL-CVCF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.15: Precisión MIL-CVCF Musk 1, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	Mext	BOW	CKNN	m-DD	EMDD	MILB
Mmax	74,795	74,795	74,795	74,795	74,795	74,795	74,795	74,795	74,795
Mmin	84,678	84,678	84,678	84,678	84,678	84,678	84,678	84,678	84,678
MExt	86,842	86,842	86,842	86,842	86,842	86,842	86,842	86,842	86,842
BOW	65,263	61,988	61,871	67,368	72,749	69,708	66,199	61,988	60,877
CKNN	86,842	86,842	86,842	86,842	86,842	86,842	86,842	86,842	86,842
mDD	71,696	75,088	78,304	74,912	74,094	72,924	77,193	81,579	71,754
EMDD	88,129	83,860	87,018	82,573	88,129	85,906	84,971	82,749	85,965
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.16: Precisión MIL-CVCF Musk 1, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	Mext	BOW	CKNN	m-DD	EMDD	MILB
Mmax	73,743	73,801	69,532	72,690	77,018	74,912	67,193	74,971	53,216
Mmin	83,743	83,743	82,632	82,632	82,632	82,632	83,743	83,743	53,216
MExt	85,731	83,626	76,140	83,567	83,567	85,673	74,035	79,474	54,327
BOW	63,099	59,766	56,374	67,310	64,152	66,257	60,819	55,439	55,439
CKNN	91,228	86,901	88,012	91,228	89,006	89,006	85,848	90,175	67,544
mDD	72,749	73,977	78,304	73,918	74,912	72,749	74,152	75,088	54,269
EMDD	83,626	84,795	75,965	83,684	83,626	80,351	85,848	84,795	52,164
MILB	53,859	50,942	50,942	50,942	50,942	50,942	50,942	50,942	11,413

- Ruido 5 %.
 - Esquema votación por consenso.

Tabla C.17: Precisión MIL-CVCF Musk 1, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	78,421	78,421	78,421	78,421	78,421	78,421	78,421	78,421	78,421
Mmin	85,848	85,848	85,848	85,848	85,848	85,848	85,848	85,848	85,848
MExt	87,135	87,135	87,135	87,135	87,135	87,135	87,135	87,135	87,135
BOW	68,421	66,082	62,982	67,427	74,152	70,468	58,713	67,310	59,474
CKNN	85,906	85,906	85,906	85,906	85,906	85,906	85,906	85,906	85,906
mDD	71,462	72,573	78,070	77,135	73,918	74,912	65,029	78,246	76,257
EMDD	82,573	87,018	84,854	85,848	86,959	83,801	82,632	83,743	84,854
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.18: Precisión MIL-CVCF Musk 1, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	70,643	70,702	64,211	69,649	71,813	73,918	69,766	70,643	52,105
Mmin	80,702	73,099	78,480	74,211	75,205	79,591	79,474	81,579	55,439
MExt	83,918	77,368	79,532	80,702	83,918	88,187	75,088	78,480	55,439
BOW	54,386	66,082	53,158	64,327	64,152	52,456	45,497	58,713	53,216
CKNN	84,854	81,579	84,795	83,801	82,573	84,795	76,082	86,901	59,766
mDD	73,099	70,643	77,076	77,135	64,620	71,988	69,708	76,023	54,327
EMDD	82,632	79,415	77,193	77,193	75,146	85,906	72,749	77,251	53,216
MILB	53,859	50,942	50,942	50,942	50,942	50,942	50,942	50,942	21,975

- Ruido 10 %.
- Esquema votación por consenso.

Tabla C.19: Precisión MIL-CVCF Musk 1, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	78,304	78,304	78,304	78,304	78,304	78,304	78,304	78,304	78,304
Mmin	74,737	74,737	74,737	74,737	74,737	74,737	74,737	74,737	74,737
MExt	84,795	84,795	84,795	84,795	84,795	84,795	84,795	84,795	84,795
BOW	58,713	63,041	69,591	65,322	60,760	61,813	53,041	64,152	55,439
CKNN	80,409	80,409	80,409	80,409	80,409	80,409	80,409	80,409	80,409
mDD	71,754	71,696	67,427	69,474	73,918	70,585	72,924	71,696	76,082
EMDD	82,690	84,854	86,959	84,854	85,906	86,959	82,749	84,854	80,643
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.20: Precisión MIL-CVCF Musk 1, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	73,918	71,813	67,544	72,865	73,977	74,035	69,649	68,480	54,327
Mmin	81,520	79,357	82,632	81,520	78,363	82,632	76,082	83,567	54,327
MExt	82,632	76,140	79,240	82,632	83,684	83,743	78,187	77,018	53,158
BOW	64,269	58,772	52,222	66,257	57,602	66,433	54,327	56,374	54,269
CKNN	83,626	82,573	80,234	82,515	82,573	81,520	74,737	84,678	57,661
mDD	73,801	62,865	68,304	63,918	63,099	67,135	63,977	75,906	52,164
EMDD	84,620	78,129	78,187	80,292	81,404	80,234	77,018	75,965	53,216
MILB	53,859	50,942	50,942	50,942	50,942	50,942	50,942	50,942	22,400

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.21: Precisión MIL-CVCF Musk 1, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	72,982	72,982	72,982	72,982	72,982	72,982	72,982	72,982	72,982
Mmin	78,363	78,363	78,363	78,363	78,363	78,363	78,363	78,363	78,363
MExt	83,743	83,743	83,743	83,743	83,743	83,743	83,743	83,743	83,743
BOW	67,427	63,275	68,596	65,380	60,819	65,263	62,047	63,099	59,708
CKNN	81,520	81,520	81,520	81,520	81,520	81,520	81,520	81,520	81,520
mDD	61,053	66,491	68,480	58,947	61,111	64,327	66,433	60,117	60,994
EMDD	81,579	76,257	77,193	78,480	72,924	78,421	80,468	77,368	81,637
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.22: Precisión MIL-CVCF Musk 1, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	76,023	76,023	69,532	73,860	72,865	76,082	70,585	71,637	53,216
Mmin	83,509	78,070	81,287	82,398	83,509	79,181	75,965	78,012	54,327
MExt	82,573	79,181	75,088	83,567	80,351	82,573	74,795	78,129	55,380
BOW	65,205	65,263	60,819	62,982	54,444	58,655	53,216	56,433	52,164
CKNN	81,404	82,398	82,515	82,456	73,743	78,246	86,959	87,895	62,924
mDD	68,596	72,749	65,263	71,930	62,865	70,760	71,696	69,474	52,105
EMDD	75,906	74,795	78,070	75,848	78,070	79,181	74,854	75,848	52,105
MILB	52,871	50,942	50,942	50,942	50,942	50,942	50,942	50,942	0,000

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.23: Precisión MIL-CVCF Musk 1, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	68,304	68,304	68,304	68,304	68,304	68,304	68,304	68,304	68,304
Mmin	74,854	74,854	74,854	74,854	74,854	74,854	74,854	74,854	74,854
MExt	89,123	89,123	89,123	89,123	89,123	89,123	89,123	89,123	89,123
BOW	58,655	60,760	51,930	58,713	62,807	57,427	60,760	59,591	61,813
CKNN	78,070	78,070	78,070	78,070	78,070	78,070	78,070	78,070	78,070
mDD	69,591	75,965	69,357	71,637	67,368	74,912	67,310	70,702	72,749
EMDD	79,415	78,363	79,357	77,193	83,743	84,795	79,357	75,029	80,526
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.24: Precisión MIL-CVCF Musk 1, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	75,146	70,760	64,269	68,596	68,713	69,825	63,275	65,380	55,380
Mmin	65,263	64,327	66,316	66,374	64,152	68,538	70,936	67,485	53,216
MExt	83,743	81,579	73,918	80,468	70,585	79,415	69,649	77,310	54,269
BOW	52,398	61,053	55,497	61,170	55,731	54,444	53,275	51,053	53,216
CKNN	72,807	80,585	69,474	81,579	68,421	69,532	71,637	84,854	53,275
mDD	62,281	58,772	68,538	65,263	63,216	59,883	64,327	75,029	53,158
EMDD	77,193	72,865	69,649	75,029	65,322	78,421	64,152	70,585	54,269
MILB	53,905	50,942	50,942	50,942	50,942	50,942	40,942	51,929	21,975

- Ruido 25 %.
- Esquema votación por consenso.

Tabla C.25: Precisión MIL-CVCF Musk 1, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	58,655	58,655	58,655	58,655	58,655	58,655	58,655	58,655	58,655
Mmin	69,474	69,474	69,474	69,474	69,474	69,474	69,474	69,474	69,474
MExt	74,854	74,854	74,854	74,854	74,854	74,854	74,854	74,854	74,854
BOW	53,099	54,444	47,953	49,006	44,327	50,994	51,228	48,889	51,930
CKNN	73,860	73,860	73,860	73,860	73,860	73,860	73,860	73,860	73,860
mDD	68,129	68,246	59,532	61,813	65,205	59,708	61,813	59,649	69,415
EMDD	70,468	77,018	74,737	72,515	73,626	74,795	74,737	71,462	71,520
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.26: Precisión MIL-CVCF Musk 1, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	68,596	71,813	65,205	70,643	61,871	64,152	65,146	70,585	54,269
Mmin	73,860	61,930	71,579	71,637	68,538	77,251	66,257	74,854	54,327
MExt	74,035	72,865	64,094	71,754	71,871	71,871	56,608	68,363	54,327
BOW	58,655	60,994	53,099	56,491	53,216	59,825	53,158	55,497	54,327
CKNN	69,474	68,129	73,801	70,526	63,918	71,520	61,930	76,082	60,877
mDD	69,415	65,146	58,538	60,760	62,982	61,520	58,713	68,363	53,216
EMDD	73,918	70,468	69,532	73,977	76,082	76,023	72,807	72,749	53,275
MILB	53,859	50,942	50,942	50,942	50,942	50,942	50,942	50,942	22,400

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.27: Precisión MIL-CVCF Musk 1, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	61,871	61,871	61,871	61,871	61,871	61,871	61,871	61,871	61,871
Mmin	66,316	66,316	66,316	66,316	66,316	66,316	66,316	66,316	66,316
MExt	67,368	67,368	67,368	67,368	67,368	67,368	67,368	67,368	67,368
BOW	61,053	69,708	66,433	58,655	57,544	63,099	62,982	65,439	56,491
CKNN	67,485	67,485	67,485	67,485	67,485	67,485	67,485	67,485	67,485
mDD	59,766	56,608	58,713	57,544	51,228	52,398	58,713	51,287	48,947
EMDD	70,468	71,696	69,357	71,696	69,474	69,474	68,304	66,140	71,754
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.28: Precisión MIL-CVCF Musk 1, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	64,211	63,099	50,877	63,041	64,152	59,766	57,602	69,357	53,275
Mmin	70,585	65,088	65,205	65,146	68,363	72,690	58,889	68,421	53,216
MExt	63,275	57,895	54,444	60,000	64,327	64,211	63,860	66,082	55,439
BOW	58,713	52,164	52,164	55,380	56,374	53,392	52,222	52,164	57,602
CKNN	68,655	65,497	68,830	74,094	63,450	68,713	67,836	81,520	64,094
mDD	65,263	55,322	50,000	53,158	59,825	60,936	51,053	57,485	55,439
EMDD	71,579	69,415	66,374	73,918	63,158	69,415	72,982	72,632	53,216
MILB	52,871	50,942	50,942	50,942	50,942	50,942	50,942	50,942	22,400

MIL-IPF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.29: Precisión MIL-IPF Musk 1, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	76,082	76,082	76,082	76,082	76,082	76,082	76,082	76,082	76,082
Mmin	82,749	82,749	82,749	82,749	82,749	82,749	82,749	82,749	82,749
MExt	84,854	84,854	84,854	84,854	84,854	84,854	84,854	84,854	84,854
BOW	59,883	65,322	64,386	61,988	66,199	61,871	62,982	68,713	64,269
CKNN	85,965	85,965	85,965	85,965	85,965	85,965	85,965	85,965	85,965
mDD	76,316	70,760	71,871	76,257	65,263	73,918	77,427	75,088	74,035
EMDD	82,573	83,743	83,684	83,743	84,737	82,573	83,743	85,789	85,789
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.30: Precisión MIL-IPF Musk 1, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	76,257	70,702	72,982	75,146	75,088	76,257	62,047	69,591	58,655
Mmin	89,123	80,526	85,848	84,795	78,246	85,848	56,491	81,462	60,877
MExt	85,965	83,743	82,573	83,743	77,076	84,854	65,380	78,187	58,596
BOW	63,158	67,251	55,322	59,883	56,433	63,158	53,216	57,602	58,655
CKNN	90,117	82,573	89,006	89,006	80,292	89,006	76,023	87,895	64,035
mDD	75,146	77,251	72,807	73,041	70,702	71,871	59,766	71,813	57,602
EMDD	89,064	81,462	82,515	88,012	83,684	84,795	61,871	77,193	58,713
MILB	57,764	50,942	50,942	50,942	50,942	50,942	30,942	52,917	33,813

- Ruido 5 %.
- Esquema votación por consenso.

Tabla C.31: Precisión MIL-IPF Musk 1, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	72,865	72,865	72,865	72,865	72,865	72,865	72,865	72,865	72,865
Mmin	84,620	84,620	84,620	84,620	84,620	84,620	84,620	84,620	84,620
MExt	83,626	83,626	83,626	83,626	83,626	83,626	83,626	83,626	83,626
BOW	63,099	66,374	66,433	64,211	66,199	64,094	54,561	64,211	59,649
CKNN	90,234	90,234	90,234	90,234	90,234	90,234	90,234	90,234	90,234
mDD	71,813	69,474	60,819	68,421	69,591	65,146	63,099	68,480	61,053
EMDD	83,509	84,620	83,450	81,228	84,561	85,673	81,228	84,620	86,784
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.32: Precisión MIL-IPF Musk 1, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	76,023	69,474	66,316	70,526	74,912	73,860	53,158	61,871	60,819
Mmin	85,965	82,515	73,977	86,959	72,865	84,795	68,538	77,135	59,766
MExt	86,901	81,462	73,743	83,626	79,240	83,626	56,491	73,977	59,708
BOW	66,374	56,316	56,550	58,713	60,760	70,409	45,380	55,380	56,433
CKNN	85,731	82,398	85,789	83,509	75,848	83,567	67,251	81,404	67,310
mDD	76,082	68,596	68,187	80,409	65,205	74,035	59,649	67,310	57,602
EMDD	82,573	71,754	74,912	80,351	76,199	80,351	56,433	62,924	56,608
MILB	56,730	50,942	50,942	50,942	50,942	50,942	29,483	51,884	23,388

- Ruido 10 %.
 - Esquema votación por consenso.

Tabla C.33: Precisión MIL-IPF Musk 1, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71,579	71,579	71,579	71,579	71,579	71,579	71,579	71,579	71,579
Mmin	81,520	81,520	81,520	81,520	81,520	81,520	81,520	81,520	81,520
MExt	85,789	85,789	85,789	85,789	85,789	85,789	85,789	85,789	85,789
BOW	60,994	64,094	59,766	64,094	60,936	60,819	62,105	61,053	63,041
CKNN	83,450	83,450	83,450	83,450	83,450	83,450	83,450	83,450	83,450
mDD	67,544	76,257	77,251	67,544	71,813	70,702	67,544	68,655	78,421
EMDD	79,415	81,462	80,468	81,462	79,298	77,193	84,795	82,573	81,462
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.34: Precisión MIL-IPF Musk 1, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	75,848	70,526	69,357	70,468	66,082	73,684	59,825	64,035	60,819
Mmin	86,842	82,456	82,398	81,345	82,573	86,959	54,386	75,965	55,497
MExt	82,456	74,912	73,860	79,181	72,807	83,509	56,491	70,409	61,930
BOW	67,310	54,386	54,327	63,216	57,602	59,825	55,439	58,713	60,877
CKNN	86,959	80,351	82,456	85,731	71,930	82,573	66,199	80,292	60,819
mDD	69,591	73,860	65,263	65,322	53,275	63,099	57,602	74,971	50,058
EMDD	83,684	74,912	76,140	82,573	73,977	80,585	56,491	67,310	58,772
MILB	57,672	50,942	50,942	50,942	49,954	49,954	30,425	52,825	33,342

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.35: Precisión MIL-IPF Musk 1, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71,930	71,930	71,930	71,930	71,930	71,930	71,930	71,930	71,930
Mmin	81,520	81,520	81,520	81,520	81,520	81,520	81,520	81,520	81,520
MExt	73,860	73,860	73,860	73,860	73,860	73,860	73,860	73,860	73,860
BOW	61,871	58,772	59,766	59,649	63,041	63,041	68,421	60,877	56,374
CKNN	76,082	76,082	76,082	76,082	76,082	76,082	76,082	76,082	76,082
mDD	68,713	65,439	67,544	69,532	67,602	64,386	67,602	74,971	74,035
EMDD	78,246	74,971	74,971	78,246	76,023	79,415	77,135	79,357	77,193
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.36: Precisión MIL-IPF Musk 1, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 20 %.
- Esquema votación por consenso.

Tabla C.37: Precisión MIL-IPF Musk 1, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	62,047	62,047	62,047	62,047	62,047	62,047	62,047	62,047	62,047
Mmin	76,023	76,023	76,023	76,023	76,023	76,023	76,023	76,023	76,023
MExt	77,193	77,193	77,193	77,193	77,193	77,193	77,193	77,193	77,193
BOW	61,871	59,766	64,094	59,766	55,380	58,830	57,544	58,830	60,643
CKNN	76,140	76,140	76,140	76,140	76,140	76,140	76,140	76,140	76,140
mDD	60,994	66,608	65,146	65,205	69,766	67,544	65,439	63,041	67,544
EMDD	78,304	79,532	81,637	82,632	78,304	81,637	79,474	81,579	82,749
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.38: Precisión MIL-IPF Musk 1, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71,754	66,374	58,772	61,930	59,825	67,602	56,550	63,977	57,778
Mmin	77,310	65,263	59,825	68,480	64,211	74,152	55,439	62,865	56,667
MExt	72,632	64,971	66,316	69,298	56,608	70,468	56,491	72,924	48,947
BOW	58,596	54,211	56,491	53,041	48,830	54,211	58,713	54,444	54,327
CKNN	70,468	69,298	72,690	72,515	65,146	71,637	64,971	69,415	61,813
mDD	57,544	55,439	62,105	58,304	48,889	54,327	62,105	69,357	52,222
EMDD	75,906	70,526	67,310	71,520	61,637	73,801	58,480	63,801	53,041
MILB	53,905	50,942	50,942	50,942	50,000	50,000	39,529	39,012	0,000

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.39: Precisión MIL-IPF Musk 1, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71,813	71,813	71,813	71,813	71,813	71,813	71,813	71,813	71,813
Mmin	70,643	70,643	70,643	70,643	70,643	70,643	70,643	70,643	70,643
MExt	74,971	74,971	74,971	74,971	74,971	74,971	74,971	74,971	74,971
BOW	58,889	56,550	68,596	60,877	66,491	60,994	64,211	65,439	57,836
CKNN	72,924	72,924	72,924	72,924	72,924	72,924	72,924	72,924	72,924
mDD	64,094	60,994	69,766	64,386	66,491	60,994	66,433	65,380	64,327
EMDD	73,801	72,807	76,082	78,129	72,865	74,912	76,023	75,906	73,918
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.40: Precisión MIL-IPF Musk 1, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.41: Precisión MIL-IPF Musk 1, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71,871	71,871	71,871	71,871	71,871	71,871	71,871	71,871	71,871
Mmin	67,368	67,368	67,368	67,368	67,368	67,368	67,368	67,368	67,368
MExt	73,743	73,743	73,743	73,743	73,743	73,743	73,743	73,743	73,743
BOW	59,883	57,544	62,047	58,830	60,877	60,877	61,170	59,883	56,550
CKNN	65,322	65,322	65,322	65,322	65,322	65,322	65,322	65,322	65,322
mDD	66,082	63,918	50,000	64,971	59,357	60,760	60,760	64,854	53,216
EMDD	74,912	71,520	77,076	74,854	74,854	76,023	75,906	73,860	72,690
MILB	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942	50,942

- Esquema votación por Max Votos.

Tabla C.42: Precisión MIL-IPF Musk 1, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71,696	61,813	62,982	62,982	58,713	64,035	60,819	65,146	61,930
Mmin	73,860	64,094	67,427	74,795	60,936	66,316	57,602	67,310	57,544
MExt	77,135	65,146	59,591	67,193	62,924	70,526	55,439	61,754	56,667
BOW	66,491	58,596	52,164	56,491	55,673	64,327	56,550	59,766	60,936
CKNN	74,912	69,532	64,035	67,193	58,713	73,977	69,474	71,696	66,199
mDD	57,719	57,778	53,333	54,444	54,386	56,433	58,772	61,930	52,164
EMDD	67,135	67,193	62,105	61,053	59,825	68,480	56,491	69,532	52,105
MILB	56,730	50,942	50,942	50,942	39,529	50,942	20,988	42,446	11,975

C.2. Mutagenesis 2

MIL-EF

- Ruido 0 %.
- Esquema votación por consenso.

Tabla C.43: Precisión MIL-EF Muta 2, Consenso, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	81.30952380952382
MIL min	69.28571428571428	81.30952380952381	81.30952380952381
MIL Ext	69.28571428571426	81.30952380952382	81.30952380952382
BOW	70.95238095238093	81.30952380952382	81.30952380952382
CKNN	71.45238095238095	81.75396825396825	82.32539682539682
maxDD	71.09126984126985	81.67989417989419	82.15608465608466
EM-DD	71.15079365079366	81.62698412698413	82.03514739229026
MILBoost	66.68331916099773	78.82212144116906	75.06950349584278

- Esquema votación por Max Votos.

Tabla C.44: Precisión MIL-EF Muta 2, Max Votos, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	81.30952380952382
MIL min	69.28571428571428	81.30952380952381	81.30952380952381
MIL Ext	69.28571428571426	81.30952380952382	81.30952380952382
BOW	71.66666666666667	81.30952380952382	81.30952380952382
CKNN	70.36507936507937	81.75396825396825	82.76984126984127
maxDD	70.18518518518519	81.67989417989419	82.52645502645504
EM-DD	70.87301587301586	81.62698412698413	83.07823129251702
MILBoost	66.44026360544218	79.11816578483244	75.9822019085412

- Ruido 5 %.
 - Esquema votación por consenso.

Tabla C.45: Precisión MIL-EF Muta 2, Consenso, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	81.30952380952382
MIL min	69.28571428571428	81.30952380952381	81.30952380952381
MIL Ext	69.28571428571426	81.30952380952382	81.30952380952382
BOW	71.57738095238093	81.30952380952382	81.30952380952382
CKNN	73.07936507936508	81.30952380952381	81.75396825396825
maxDD	72.44708994708996	81.30952380952382	81.67989417989419
EM-DD	71.99546485260771	81.30952380952381	81.94444444444444
MILBoost	67.42240646258503	76.90371866339129	74.9901384164777

- Esquema votación por Max Votos.

Tabla C.46: Precisión MIL-EF Muta 2, Max Votos, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	81.30952380952382
MIL min	69.28571428571428	81.30952380952381	81.30952380952381
MIL Ext	69.28571428571426	81.30952380952382	81.30952380952382
BOW	71.66666666666667	81.30952380952382	81.30952380952382
CKNN	71.74603174603175	81.30952380952381	81.75396825396825
maxDD	71.33597883597885	81.30952380952382	81.67989417989419
EM-DD	71.04308390022675	81.30952380952381	81.94444444444443
MILBoost	66.5890731292517	78.54434366339129	74.9901384164777

- Ruido 10 %.
 - Esquema votación por consenso.

Tabla C.47: Precisión MIL-EF Muta 2, Consenso, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	81.30952380952382
MIL min	69.28571428571428	81.30952380952381	81.30952380952381
MIL Ext	69.28571428571426	81.30952380952382	81.30952380952382
BOW	72.22222222222221	81.30952380952382	81.30952380952382
CKNN	71.15079365079364	81.43650793650794	80.69841269841271
maxDD	70.8399470899471	81.41534391534393	80.80026455026456
EM-DD	70.30045351473923	81.40022675736962	80.87301587301587
MILBoost	65.9392715419501	77.27912808641975	74.0526384164777

- Esquema votación por Max Votos.

Tabla C.48: Precisión MIL-EF Muta 2, Max Votos, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	86.38888888888889
MIL min	69.28571428571428	81.30952380952381	86.38888888888889
MIL Ext	69.28571428571426	81.30952380952382	86.38888888888889
BOW	71.64682539682539	81.30952380952382	86.38888888888889
CKNN	72.95238095238093	81.30952380952381	86.83333333333334
maxDD	72.34126984126985	81.30952380952382	86.75925925925927
EM-DD	72.26190476190476	81.30952380952381	87.06349206349206
MILBoost	67.65554138321995	77.3097757621567	80.7265487213404

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.49: Precisión MIL-EF Muta 2, Consenso, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	81.30952380952382
MIL min	69.28571428571428	81.30952380952381	81.30952380952381
MIL Ext	69.28571428571426	81.30952380952382	81.30952380952382
BOW	70.625	81.30952380952382	81.30952380952382
CKNN	70.35714285714285	80.86507936507937	80.36507936507937
maxDD	70.17857142857143	80.93915343915344	80.52248677248677
EM-DD	70.45918367346938	80.99206349206351	80.63492063492065
MILBoost	66.078160430839	77.03199798437893	76.3134408856135

- Esquema votación por Max Votos.

Tabla C.50: Precisión MIL-EF Muta 2, Max Votos, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	83.53174603174604
MIL min	68.17460317460316	81.30952380952381	83.53174603174604
MIL Ext	68.54497354497353	81.30952380952382	83.53174603174604
BOW	69.77182539682539	81.30952380952382	83.53174603174604
CKNN	72.07936507936508	81.30952380952381	84.86507936507937
maxDD	71.61375661375662	81.30952380952382	84.64285714285714
EM-DD	71.0034013605442	81.30952380952381	84.8015873015873
MILBoost	66.55435090702947	81.30952380952381	77.27408903376165

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.51: Precisión MIL-EF Muta 2, Consenso, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	81.30952380952382	83.53174603174604
MIL min	70.7142857142857	81.30952380952381	83.53174603174604
MIL Ext	70.23809523809521	81.30952380952382	83.53174603174604
BOW	70.55555555555556	81.30952380952382	83.53174603174604
CKNN	69.15873015873015	81.30952380952381	83.53174603174602
maxDD	69.17989417989419	81.30952380952382	83.53174603174602
EM-DD	69.19501133786848	81.30952380952381	83.53174603174602
MILBoost	64.9720096371882	78.43433090828924	76.16297792265054

- Esquema votación por Max Votos.

Tabla C.52: Precisión MIL-EF Muta 2, Max Votos, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	64.28571428571428	73.80952380952382	81.03174603174604
MIL min	63.45238095238095	76.30952380952381	81.03174603174604
MIL Ext	62.24867724867725	75.4761904761905	81.03174603174604
BOW	66.38888888888889	76.30952380952382	81.65674603174604
CKNN	69.5	76.30952380952381	82.54761904761905
maxDD	69.46428571428572	76.7261904761905	82.29497354497354
EM-DD	69.7562358276644	77.02380952380953	82.43197278911565
MILBoost	65.46308106575964	73.38809366339129	76.66961923658353

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.53: Precisión MIL-EF Muta 2, Consenso, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	78.80952380952382	78.80952380952382
MIL min	68.17460317460316	78.80952380952381	78.80952380952381
MIL Ext	68.54497354497353	78.80952380952382	78.80952380952382
BOW	67.61904761904762	78.80952380952382	78.80952380952382
CKNN	69.35714285714285	78.80952380952381	78.30952380952381
maxDD	69.3452380952381	78.80952380952382	78.39285714285715
EM-DD	69.33673469387755	78.80952380952381	78.45238095238096
MILBoost	65.09601757369614	75.24663800705467	73.69957010582011

- Esquema votación por Max Votos.

Tabla C.54: Precisión MIL-EF Muta 2, Max Votos, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	62.61904761904761	70.1984126984127	84.44444444444444
MIL min	65.95238095238095	74.64285714285714	84.44444444444444
MIL Ext	64.84126984126983	73.16137566137566	84.44444444444444
BOW	64.84126984126983	74.0873015873016	84.44444444444446
CKNN	64.15873015873015	74.64285714285715	84.44444444444446
maxDD	65.0132275132275	75.3835978835979	84.44444444444444
EM-DD	65.62358276643991	75.9126984126984	84.44444444444443
MILBoost	61.84700963718821	72.80360292265054	80.32627865961199

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.55: Precisión MIL-EF Muta 2, Consenso, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	55.3968253968254	59.84126984126984	70.95238095238096
MIL min	62.34126984126984	61.78571428571429	68.45238095238095
MIL Ext	60.02645502645502	61.13756613756615	69.28571428571428
BOW	63.70039682539683	64.28571428571429	70.32738095238095
CKNN	63.92857142857142	65.73015873015873	71.45238095238095
maxDD	64.82142857142857	65.3968253968254	70.53571428571428
EM-DD	64.74489795918367	65.15873015873018	69.88095238095238
MILBoost	61.078160430839	63.161100088183424	66.15862780297304

- Esquema votación por Max Votos.

Tabla C.56: Precisión MIL-EF Muta 2, Max Votos, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	69.28571428571429	79.0873015873016	81.30952380952382
MIL min	69.28571428571428	79.08730158730158	81.30952380952381
MIL Ext	69.28571428571426	79.0873015873016	81.30952380952382
BOW	70.625	79.0873015873016	81.30952380952382
CKNN	71.42857142857143	79.0873015873016	83.21428571428572
maxDD	71.0714285714286	79.0873015873016	82.89682539682539
EM-DD	70.81632653061223	79.08730158730158	83.30498866213152
MILBoost	66.390660430839	74.68149644116906	79.33358528596624

MIL-CVCF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.57: Precisión MIL-CVCF Muta 2, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	80,397	78,175	78,175	75,952	80,397	75,952	80,397	80,397	78,175
CKNN	72,143	72,143	72,143	72,143	72,143	72,143	72,143	74,643	72,143
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	69,286	69,286	71,508	69,286	71,508	67,063	66,786	67,063	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.58: Precisión MIL-CVCF Muta 2, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	100,000	81,310	93,333	60,000	100,000	100,000	0,000	20,000	0,000
Mmin	97,778	81,310	93,333	60,000	100,000	100,000	0,000	40,000	2,222
MExt	97,778	81,310	93,333	60,000	100,000	100,000	0,000	20,000	2,222
BOW	92,421	81,310	93,333	60,000	100,000	100,000	0,000	0,000	7,579
CKNN	86,389	85,754	93,333	100,000	100,000	92,778	100,000	100,000	19,881
mDD	95,278	81,310	93,333	60,000	100,000	100,000	0,000	20,000	4,722
EMDD	94,921	81,310	93,333	60,000	100,000	100,000	0,000	20,000	5,079
MILB	15,481	10,054	9,308	4,444	9,802	12,024	0,000	0,000	7,901

- Ruido 5 %.
 - Esquema votación por consenso.

Tabla C.59: Precisión MIL-CVCF Muta 2, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	76,230	78,452	78,452	76,230	78,452	76,230	76,230	74,008	76,230
CKNN	74,008	74,008	74,008	73,373	74,008	74,008	76,230	74,008	76,230
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	69,286	69,286	69,286	69,286	69,286	71,508	71,508	71,508	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.60: Precisión MIL-CVCF Muta 2, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	100,000	81,310	93,333	60,000	100,000	100,000	0,000	60,000	0,000
Mmin	93,056	81,310	93,333	60,000	100,000	100,000	0,000	0,000	6,944
MExt	94,643	81,310	93,333	60,000	100,000	100,000	0,000	40,000	5,357
BOW	100,000	81,310	93,333	60,000	100,000	100,000	0,000	0,000	5,079
CKNN	95,278	83,532	93,333	100,000	100,000	97,778	100,000	100,000	22,302
mDD	97,778	81,310	93,333	60,000	100,000	100,000	0,000	20,000	2,222
EMDD	97,778	81,310	93,333	60,000	100,000	100,000	0,000	0,000	2,222
MILB	12,024	14,952	10,758	2,222	12,024	12,024	0,000	2,222	0,000

- Ruido 10 %.
- Esquema votación por consenso.

Tabla C.61: Precisión MIL-CVCF Muta 2, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	79,087	79,087	79,087	74,008	76,230	79,087	76,230	76,230	79,087
CKNN	61,706	63,929	63,929	61,706	63,929	63,929	63,929	63,929	61,706
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	75,952	69,008	69,286	69,286	69,286	69,286	69,286	71,508	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.62: Precisión MIL-CVCF Muta 2, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	97,778	81,310	91,111	57,778	77,778	97,778	20,000	40,000	2,222
Mmin	100,000	81,310	93,333	60,000	80,000	100,000	0,000	0,000	0,000
MExt	97,143	81,310	93,333	60,000	80,000	100,000	0,000	0,000	2,857
BOW	90,198	81,310	93,333	60,000	80,000	100,000	0,000	20,000	9,802
CKNN	88,611	81,310	95,556	100,000	100,000	88,611	100,000	100,000	14,524
mDD	97,778	81,310	90,833	80,000	80,000	100,000	0,000	40,000	2,222
EMDD	94,643	81,310	93,333	60,000	60,000	100,000	0,000	0,000	0,000
MILB	15,940	10,577	9,308	4,444	7,579	12,024	0,000	2,500	9,273

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.63: Precisión MIL-CVCF Muta 2, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	74,008	71,151	76,230	76,230	76,230	73,373	71,786	76,230	71,786
CKNN	64,881	64,881	62,659	62,659	62,659	64,881	64,881	62,659	64,881
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	71,786	74,008	69,286	69,286	69,286	71,786	69,286	71,786	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.64: Precisión MIL-CVCF Muta 2, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	92,421	78,452	90,476	80,000	100,000	100,000	0,000	40,000	7,579
Mmin	94,921	81,310	88,611	77,778	97,778	97,778	20,000	0,000	5,079
MExt	90,198	81,310	93,333	60,000	80,000	100,000	0,000	20,000	9,802
BOW	90,278	78,810	90,833	77,500	97,500	97,500	20,000	20,000	6,944
CKNN	92,421	76,310	88,611	97,778	92,421	90,198	100,000	95,556	19,683
mDD	92,698	78,810	88,611	77,778	77,778	97,778	20,000	0,000	7,302
EMDD	90,278	81,310	93,333	60,000	100,000	91,111	0,000	60,000	4,722
MILB	19,231	16,827	17,008	10,694	10,694	15,774	0,000	4,375	7,901

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.65: Precisión MIL-CVCF Muta 2, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	64,286	64,286	64,286	64,286	64,286	64,286	64,286	64,286	64,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	68,651	68,651	68,651	71,508	71,508	68,651	68,651	70,873	65,794
CKNN	59,048	61,270	63,492	63,492	59,048	59,048	61,270	59,048	63,492
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	69,286	64,286	64,286	69,286	66,786	69,286	72,143	69,286	71,508
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.66: Precisión MIL-CVCF Muta 2, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	93,333	81,310	87,976	100,000	100,000	100,000	0,000	40,000	6,667
Mmin	94,643	81,310	93,333	60,000	60,000	100,000	0,000	0,000	5,357
MExt	87,778	76,587	86,111	68,056	70,556	90,556	32,778	25,000	9,722
BOW	95,278	78,810	90,833	77,500	97,500	97,500	15,000	17,500	12,579
CKNN	81,944	81,310	93,333	95,556	95,556	86,111	97,778	95,556	6,667
mDD	90,833	79,087	88,254	72,698	72,698	92,698	37,778	20,000	9,167
EMDD	97,778	83,810	88,333	77,500	77,500	97,500	20,000	20,000	2,222
MILB	15,481	25,015	14,005	4,375	15,133	17,356	3,951	3,951	3,951

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.67: Precisión MIL-CVCF Muta 2, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	66,429	66,429	66,429	66,429	66,429	66,429	66,429	66,429	66,429
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	76,865	76,865	79,087	81,944	81,944	74,643	76,865	74,643	71,786
CKNN	69,286	71,508	73,730	71,508	71,508	71,508	71,508	71,508	73,730
mDD	59,286	59,286	64,286	59,286	61,508	59,286	59,286	64,286	59,286
EMDD	69,286	71,508	69,286	73,730	69,286	69,286	74,365	69,286	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.68: Precisión MIL-CVCF Muta 2, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	92,698	74,643	86,667	51,111	35,556	95,556	35,556	35,556	18,413
Mmin	92,698	78,452	90,476	77,143	77,143	97,143	20,000	0,000	7,302
MExt	95,278	81,310	93,333	60,000	40,000	100,000	0,000	0,000	4,722
BOW	70,873	74,286	78,175	80,675	72,103	85,754	55,476	63,056	11,667
CKNN	80,754	78,452	85,119	97,143	89,286	89,563	100,000	100,000	12,579
mDD	100,000	81,310	91,111	57,778	57,778	97,778	20,000	20,000	0,000
EMDD	85,198	66,310	80,833	75,000	85,000	92,778	5,000	35,000	17,024
MILB	17,521	10,054	14,215	9,630	14,987	17,209	0,000	8,536	4,898

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.69: Precisión MIL-CVCF Muta 2, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	61,071	61,071	61,071	61,071	61,071	61,071	61,071	61,071	61,071
MExt	58,175	58,175	58,175	58,175	58,175	58,175	58,175	58,175	58,175
BOW	76,587	76,587	74,365	76,587	76,587	76,587	74,365	72,143	72,143
CKNN	56,349	56,349	56,349	56,349	56,349	56,349	56,349	56,349	56,349
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	66,786	61,071	61,071	56,071	61,071	64,286	58,929	66,786	58,571
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.70: Precisión MIL-CVCF Muta 2, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	97,778	78,452	88,254	77,778	57,778	97,778	20,000	40,000	2,222
Mmin	85,754	56,270	68,294	72,738	72,738	74,960	62,540	27,619	35,198
MExt	97,778	79,087	86,032	70,476	70,476	90,476	57,778	40,000	2,222
BOW	85,556	71,587	73,889	60,556	82,778	92,778	20,556	30,278	9,722
CKNN	81,349	85,754	93,333	100,000	97,143	86,429	100,000	100,000	12,381
mDD	95,278	79,087	86,032	73,333	73,333	93,333	37,778	40,000	4,722
EMDD	93,333	81,310	91,111	57,778	31,111	97,778	20,000	20,000	6,667
MILB	24,728	13,657	26,144	17,608	18,737	22,687	17,008	14,786	12,799

MIL-IPF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.71: Precisión MIL-IPF Muta 2, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	78,730	76,508	78,730	78,730	78,730	78,730	78,730	76,230	76,508
CKNN	75,317	75,317	75,317	73,095	75,317	75,317	75,317	73,095	75,317
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	71,786	67,063	69,563	69,286	69,286	69,286	69,286	69,286	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.72: Precisión MIL-IPF Muta 2, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 5 %.
- Esquema votación por consenso.

Tabla C.73: Precisión MIL-IPF Muta 2, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	76,230	74,008	74,008	76,230	74,008	74,008	71,786	74,008	74,008
CKNN	72,817	72,817	72,817	72,817	72,817	72,817	72,817	72,817	72,817
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	69,286	74,008	69,286	69,286	69,286	69,286	75,952	69,286	71,508
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.74: Precisión MIL-IPF Muta 2, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 10 %.
- Esquema votación por consenso.

Tabla C.75: Precisión MIL-IPF Muta 2, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	81,587	76,508	76,508	74,008	78,730	76,587	78,810	81,587	78,730
CKNN	75,595	75,595	75,595	75,595	73,373	75,595	75,595	75,595	73,373
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.76: Precisión MIL-IPF Muta 2, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 15 %.
- Esquema votación por consenso.

Tabla C.77: Precisión MIL-IPF Muta 2, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	76,865	81,310	83,532	81,310	80,675	76,230	79,087	78,452	81,310
CKNN	64,563	64,563	64,563	64,563	64,563	64,563	64,563	64,563	64,563
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	69,286	69,286	67,063	69,286	67,063	73,730	69,286	62,619	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.78: Precisión MIL-IPF Muta 2, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.79: Precisión MIL-IPF Muta 2, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	71,151	64,206	71,786	74,008	73,373	74,008	70,873	66,429	73,373
CKNN	77,500	77,500	77,500	77,500	77,500	77,500	77,500	77,500	77,500
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	74,365	69,286	72,143	69,286	69,286	72,143	69,286	72,143	69,286
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.80: Precisión MIL-IPF Muta 2, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.81: Precisión MIL-IPF Muta 2, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	60,397	60,397	60,397	60,397	60,397	60,397	60,397	60,397	60,397
Mmin	62,619	62,619	62,619	62,619	62,619	62,619	62,619	62,619	62,619
MExt	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
BOW	71,508	71,508	71,508	71,508	76,587	71,508	74,365	73,730	72,143
CKNN	64,206	64,206	66,429	66,429	66,429	66,429	66,429	66,429	66,429
mDD	64,286	64,286	64,286	64,286	64,286	64,286	59,286	64,286	64,286
EMDD	62,619	62,619	62,619	72,063	62,619	62,619	62,619	62,619	62,619
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.82: Precisión MIL-IPF Muta 2, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 30 %.
- Esquema votación por consenso.

Tabla C.83: Precisión MIL-IPF Muta 2, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
Mmin	56,905	56,905	56,905	56,905	56,905	56,905	56,905	56,905	56,905
MExt	56,905	56,905	56,905	56,905	56,905	56,905	56,905	56,905	56,905
BOW	62,421	72,143	74,365	72,143	72,143	64,643	69,643	72,143	72,143
CKNN	57,183	57,183	57,183	57,183	57,183	57,183	57,183	57,183	57,183
mDD	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286	69,286
EMDD	64,841	64,841	62,341	66,786	62,341	64,563	69,286	64,563	67,063
MILB	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411	35,411

- Esquema votación por Max Votos.

Tabla C.84: Precisión MIL-IPF Muta 2, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

C.3. Tiger

MIL-EF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.85: Precisión MIL-EF Tiger, Consenso, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	75.0	75.0	78.0
MIL min	76.0	76.0	77.75
MIL Ext	76.83333333333333	76.33333333333333	77.5
BOW	75.25	75.375	76.75
CKNN	74.3	74.8	75.9
maxDD	72.83333333333333	71.5	71.58333333333333
EM-DD	71.21428571428571	70.57142857142857	70.35714285714286
MILBoost	68.5625	68.0	67.8125

- Esquema votación por Max Votos.

Tabla C.86: Precisión MIL-EF Tiger, Max Votos, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	73.0	71.0	72.0
MIL min	74.5	73.75	75.25
MIL Ext	74.5	73.33333333333333	73.83333333333333
BOW	72.625	72.75	72.25
CKNN	72.1	73.1	72.2
maxDD	69.33333333333333	70.0	68.75
EM-DD	68.57142857142857	69.5	68.28571428571429
MILBoost	66.25	67.0625	66.0

- Ruido 5 %.

- Esquema votación por consenso.

Tabla C.87: Precisión MIL-EF Tiger, Consenso, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	75.0	74.0	74.5
MIL min	76.5	73.25	75.75
MIL Ext	75.83333333333333	73.66666666666667	74.83333333333333
BOW	74125	73125	74.0
CKNN	73.6	73.9	74.1
maxDD	70.91666666666667	69.83333333333333	70.08333333333333
EM-DD	69.85714285714286	68.78571428571429	68.64285714285714
MILBoost	67375	66.4375	66.3125

- Esquema votación por Max Votos.

Tabla C.88: Precisión MIL-EF Tiger, Max Votos, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	70.5	67.0	68.5
MIL min	72.5	69.75	69.5
MIL Ext	72.0	69.33333333333333	69.0
BOW	70.875	68.75	68.0
CKNN	71.0	69.7	69.3
maxDD	67.75	66.91666666666667	66.08333333333333
EM-DD	66.35714285714286	66.42857142857143	65.64285714285714
MILBoost	64.3125	64375	63.6875

- Ruido 10 %.
- Esquema votación por consenso.

Tabla C.89: Precisión MIL-EF Tiger, Consenso, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	72.5	67.0	72.5
MIL min	71.0	61.75	69.25
MIL Ext	72.33333333333333	63.5	70.66666666666667
BOW	71.25	64.125	69.875
CKNN	70.3	65.2	69.9
maxDD	67.41666666666667	62.91666666666664	66.83333333333333
EM-DD	66.35714285714286	63.0	65.71428571428571
MILBoost	64.3125	61.375	63.75

- Esquema votación por Max Votos.

Tabla C.90: Precisión MIL-EF Tiger, Max Votos, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	73.5	68.0	69.0
MIL min	74.25	67.25	70.5
MIL Ext	74.16666666666667	67.5	70.33333333333333
BOW	72.375	67.625	69.375
CKNN	71.5	68.3	69.8
maxDD	68.33333333333333	65.16666666666667	66.66666666666667
EM-DD	67.07142857142857	65.21428571428571	65.92857142857143
MILBoost	64.9375	63.3125	63.9375

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.91: Precisión MIL-EF Tiger, Consenso, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	71.5	71.0	73.0
MIL min	72.25	67.75	70.75
MIL Ext	72.33333333333333	69.66666666666667	72.33333333333333
BOW	70.875	69.875	70.25
CKNN	70.1	69.9	70.1
maxDD	66.33333333333333	66.33333333333333	66.83333333333333
EM-DD	65.14285714285714	65.5	65.85714285714286
MILBoost	63.25	63.5625	63.875

- Esquema votación por Max Votos.

Tabla C.92: Precisión MIL-EF Tiger, Max Votos, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	71.5	62.0	69.0
MIL min	70.5	61.75	68.0
MIL Ext	70.16666666666667	62.83333333333333	68.83333333333333
BOW	70.25	63.375	68.875
CKNN	69.3	64.9	70.1
maxDD	66.5	62.25	66.66666666666667
EM-DD	65.14285714285714	62.71428571428571	66.14285714285714
MILBoost	63.25	61.125	64.125

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.93: Precisión MIL-EF Tiger, Consenso, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	67.0	67.5	72.0
MIL min	68.75	67.75	69.75
MIL Ext	69.16666666666667	67.83333333333333	70.16666666666667
BOW	68.625	67.25	69.0
CKNN	67.4	67.3	68.3
maxDD	64.91666666666667	64.16666666666667	65.08333333333333
EM-DD	63.642857142857146	63.57142857142857	63.642857142857146
MILBoost	61.9375	61.875	61.9375

- Esquema votación por Max Votos.

Tabla C.94: Precisión MIL-EF Tiger, Max Votos, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	67.5	64.5	65.5
MIL min	64.5	60.0	59.75
MIL Ext	65.83333333333333	62.0	61.5
BOW	65.875	62.0	62.0
CKNN	65.8	63.7	63.6
maxDD	63.33333333333336	61.83333333333336	61.416666666666664
EM-DD	62.142857142857146	61.714285714285715	61.5
MILBoost	60.625	60.25	60.0625

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.95: Precisión MIL-EF Tiger, Consenso, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	72.5	61.5	72.0
MIL min	67.0	58.5	67.0
MIL Ext	68.33333333333333	60.166666666666664	67.83333333333333
BOW	67.625	60.125	67.0
CKNN	66.8	60.9	66.9
maxDD	65.25	59.83333333333336	63.916666666666664
EM-DD	63.857142857142854	59.214285714285715	62.92857142857143
MILBoost	62.125	58.0625	61.3125

- Esquema votación por Max Votos.

Tabla C.96: Precisión MIL-EF Tiger, Max Votos, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	67.0	65.5	66.5
MIL min	65.75	65.0	63.5
MIL Ext	66.66666666666667	65.33333333333333	64.83333333333333
BOW	66.5	66.5	64.375
CKNN	66.4	67.2	65.4
maxDD	64.25	64.25	62.75
EM-DD	62.857142857142854	64.0	62.0
MILBoost	61.25	62.25	60.5

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.97: Precisión MIL-EF Tiger, Consenso, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	59.5	55.0	62.0
MIL min	60.5	54.5	59.25
MIL Ext	60.166666666666664	54.333333333333336	60.333333333333336
BOW	60.0	54.25	60.25
CKNN	59.7	55.0	60.1
maxDD	58.333333333333336	54.166666666666664	58.25
EM-DD	57.07142857142857	53.785714285714285	58.0
MILBoost	56.1875	53.3125	57.0

- Esquema votación por Max Votos.

Tabla C.98: Precisión MIL-EF Tiger, Max Votos, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	64.0	60.5	63.0
MIL min	60.0	58.75	60.25
MIL Ext	60.333333333333336	59.0	60.666666666666664
BOW	59.5	59.125	61.0
CKNN	60.2	59.7	62.1
maxDD	59.75	58.416666666666664	60.666666666666664
EM-DD	59.214285714285715	58.642857142857146	60.92857142857143
MILBoost	58.0625	57.5625	59.5625

MIL-CVCF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.99: Precisión MIL-CVCF Tiger, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5
Mmin	78	78	78	78	78	78	78	78	78
MExt	75	75	75	75	75	75	75	75	75
BOW	73,5	71	69,5	72,5	67,5	69,5	70	71,5	67
CKNN	75,5	75,5	75,5	75,5	75,5	75,5	75,5	75,5	75,5
mDD	59,5	50,5	54	47,5	56,5	51	57	56,5	53
EMDD	62	61,5	64,5	64,5	60,5	60	61,5	64,5	64
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.100: Precisión MIL-CVCF Tiger, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	72	75	71	72,5	72	72,5	59,5	70,5	51,5
Mmin	77	75,5	74	75	74,5	77,5	63	59,5	51
MExt	78,5	76	71,5	75	72,5	76	70	70	52
BOW	68,5	71,5	68,5	70	68,5	71,5	66	64,5	52
CKNN	72,5	77	72,5	76	76	73	69	74,5	54
mDD	55	54,5	52	54,5	52	54,5	50,5	49	51
EMDD	65	64,5	69,5	64,5	64,5	67,5	61	58,5	52
MILB	51,9	50	50	50	50	50	50	50	41,9

- Ruido 5 %.
- Esquema votación por consenso.

Tabla C.101: Precisión MIL-CVCF Tiger, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	73	73	73	73	73	73	73	73	73
Mmin	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5
MExt	68	67,5	68	68	68	68	68	67,5	68
BOW	73	72	70	70	68,5	70,5	72	72,5	67
CKNN	74	74	74	74	74	74	74	74	74
mDD	51,5	51	54	52	56	55,5	58	55	51,5
EMDD	62	62,5	61	62,5	66,5	66,5	60	62	61,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.102: Precisión MIL-CVCF Tiger, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71,5	67,5	69	70	70	68	62	66,5	51
Mmin	76,5	75	70,5	75	71,5	72,5	54,5	57	50,5
MExt	72,5	71,5	70,5	72	74	72,5	50,5	65	51,5
BOW	71	69	67	70,5	68	68,5	58,5	63,5	51,5
CKNN	72,5	75,5	73,5	76,5	74,5	70,5	61,5	73	55
mDD	48,5	59	50,5	53	51,5	58,5	52	48	50,5
EMDD	59	64,5	67	65,5	62	60	56	57,5	51,5
MILB	50,95	50	50	50	50	50	50	50	10,475

- Ruido 10 %.
 - Esquema votación por consenso.

Tabla C.103: Precisión MIL-CVCF Tiger, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	73	73	73	73	73	73	73	73	73
Mmin	72,5	72,5	72,5	72,5	72,5	72,5	72,5	72,5	72,5
MExt	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5
BOW	65,5	69	73,5	67,5	69	66,5	67,5	68	67,5
CKNN	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5
mDD	51	50	56	54	55	50,5	50	56	50
EMDD	57,5	59,5	60	57,5	59,5	62	60	60	59
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.104: Precisión MIL-CVCF Tiger, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71	69,5	64,5	68,5	68,5	66,5	53	66	51
Mmin	73	72,5	62,5	73	70	69	53,5	55	52
MExt	71,5	72	64	72	68,5	70,5	50,5	65	52
BOW	64	67	62,5	69	61	74,5	55,5	62,5	51,5
CKNN	66,5	73	75,5	73	73,5	69,5	61,5	70,5	57
mDD	48,5	52,5	54	51,5	53	52,5	49	52,5	52
EMDD	59,5	63	63,5	64,5	59,5	60,5	54	60	50,5
MILB	52,375	50	50	50	50	50	50	50	31,425

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.105: Precisión MIL-CVCF Tiger, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	70	70	70	70	70	70	70	70	70
Mmin	68	68	68	68	68	68	68	68	68
MExt	70,5	70,5	70,5	70,5	70,5	70,5	70,5	70,5	70,5
BOW	70	69,5	68	69	67	72,5	66	66	69,5
CKNN	65,5	65,5	65,5	65,5	65,5	65,5	65,5	65,5	65,5
mDD	58	52,5	53,5	51	47	54,5	52,5	49	48
EMDD	55,5	56	60	59	60,5	60,5	57	59	61
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.106: Precisión MIL-CVCF Tiger, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	73	71	58,5	70,5	71,5	71	56	60	50
Mmin	76,5	74	64	73,5	72,5	73,5	53,5	51,5	51,5
MExt	75,5	71,5	69,5	71	71	71,5	51,5	58,5	51
BOW	67,5	69	63	72,5	70,5	65	52,5	59,5	51,5
CKNN	69	73	67,5	71	72	69,5	57	62,5	51,5
mDD	53,5	58,5	51	49,5	54,5	54,5	48	49,5	49
EMDD	60	63	58,5	59	61	61	50,5	56	52
MILB	51,9	50	50	50	50	50	50,475	50,475	20,95

- Ruido 20 %.
- Esquema votación por consenso.

Tabla C.107: Precisión MIL-CVCF Tiger, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	67,5	67,5	67,5	67,5	67,5	67,5	67,5	67,5	67,5
Mmin	63	63	63	63	63	63	63	63	63
MExt	73	73	73	73	73	73	73	73	73
BOW	63	66,5	64	67,5	66,5	69	64,5	64	62,5
CKNN	66	66	66	66	66	66	66	66	66
mDD	57	57	53	52	57	48	51	51	54,5
EMDD	53	55,5	55	50	51	53	56	57,5	55,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.108: Precisión MIL-CVCF Tiger, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	68	67	55	64	63	63	55	58	51
Mmin	64	64	58	63,5	62	65	55,5	52	51,5
MExt	70,5	69	64,5	72	70	71	56,5	53,5	51
BOW	63	65	55	64	67	62	56	57	52
CKNN	62,5	69	61,5	69,5	69	64	56	61,5	53
mDD	61	49	50	48,5	51	48,5	52,5	49,5	50,5
EMDD	59,5	62,5	61	67,5	64,5	62	57,5	59	50
MILB	51,425	50	50	50	50	50	50	50	31,425

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.109: Precisión MIL-CVCF Tiger, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	66	66	66	66	66	66	66	66	66
Mmin	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5
MExt	66	66	66	66	66	66	66	66	66
BOW	67,5	65,5	72	68,5	67	65,5	67	66	64
CKNN	67	67	67	67	67	67	67	67	67
mDD	51,5	52,5	55,5	51	52,5	51,5	53	50,5	56
EMDD	55,5	54,5	56,5	53,5	56,5	55	51,5	55	53,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.110: Precisión MIL-CVCF Tiger, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69	68,5	58	69,5	70	67	53	54	51
Mmin	55,5	55,5	51,5	55	60	58	50,5	51	51
MExt	67	63,5	50	64	64	66	51	51,5	50,5
BOW	65,5	64,5	51	62	66	65,5	51	55	50,5
CKNN	64,5	67,5	63,5	67,5	70,5	65,5	56	59	50,5
mDD	52	51	50	48,5	48,5	53	55,5	51	52
EMDD	55	59,5	58,5	61,5	60	55	53	51,5	40
MILB	51,9	50	50	50	50	50	50,475	50,475	31,9

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.111: Precisión MIL-CVCF Tiger, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69,5	69,5	69,5	69,5	69,5	69,5	69,5	69,5	69,5
Mmin	56	56	56	56	56	56	56	56	56
MExt	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5
BOW	63,5	61,5	63,5	61,5	62	61,5	60	60,5	58
CKNN	60	60	60	60	60	60	60	60	60
mDD	56	55	56	55,5	53,5	50,5	52	50,5	52,5
EMDD	47,5	52	54,5	47	45	37	53	45	48
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.112: Precisión MIL-CVCF Tiger, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69	63	50	59	65,5	64	49,5	51	52,5
Mmin	54,5	50,5	50,5	50,5	52	51	52,5	56	49,5
MExt	63	63	56	62,5	62,5	64,5	52,5	52	50,5
BOW	64,5	66	56	62	64,5	62	56	53,5	51,5
CKNN	62,5	65	55	66	68,5	61	51,5	59,5	56
mDD	51,5	50	50	50	54,5	51	50	51	51
EMDD	52,5	56,5	55	60,5	59,5	57	51,5	57	49,5
MILB	50,95	50	50	50	50	50	50	50	20,95

MIL-IPF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.113: Precisión MIL-IPF Tiger, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	73	73	73	73	73	73	73	73	73
Mmin	76	76	76	76	76	76	76	76	76
MExt	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5
BOW	68,5	70	70,5	72,5	68,5	69,5	67,5	64	72,5
CKNN	70	70	70	70	70	70	70	70	70
mDD	55,5	52	56	52,5	56	54,5	56,5	50,5	55
EMDD	64	63,5	61	59	59	63	61,5	61	63,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.114: Precisión MIL-IPF Tiger, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 5 %.
- Esquema votación por consenso.

Tabla C.115: Precisión MIL-IPF Tiger, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	68	68	68	68	68	68	68	68	68
Mmin	77	77	77	77	77	77	77	77	77
MExt	70	70	70	70	70	70	70	70	70
BOW	69	72,5	71	64,5	70,5	71,5	70	66,5	71,5
CKNN	71	71	71	71	71	71	71	71	71
mDD	56,5	54	53,5	51	60	53	61,5	59	53
EMDD	60	61,5	57	58	63,5	47	59	59,5	55,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.116: Precisión MIL-IPF Tiger, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 10 %.
- Esquema votación por consenso.

Tabla C.117: Precisión MIL-IPF Tiger, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	71	71	71	71	71	71	71	71	71
Mmin	75	75	75	75	75	75	75	75	75
MExt	73,5	73,5	73,5	73,5	73,5	73,5	73,5	73,5	73,5
BOW	67	69,5	72	68,5	72,5	70	66,5	66,5	65,5
CKNN	70	70	70	70	70	70	70	70	70
mDD	52,5	56,5	51	51	52,5	48	54,5	53	48,5
EMDD	63,5	58	65,5	60,5	59,5	61,5	61	60	58,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.118: Precisión MIL-IPF Tiger, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 15 %.
- Esquema votación por consenso.

Tabla C.119: Precisión MIL-IPF Tiger, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	65	65	65	65	65	65	65	65	65
Mmin	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5
MExt	73	73	73	73	73	73	73	73	73
BOW	69	69	65,5	63	65,5	72	71,5	68,5	70,5
CKNN	64,5	64,5	64,5	64,5	64,5	64,5	64,5	64,5	64,5
mDD	51,5	54	50	55	56,5	54	53	51	53
EMDD	55	56,5	54	58	56,5	55,5	56,5	56,5	50,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.120: Precisión MIL-IPF Tiger, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.121: Precisión MIL-IPF Tiger, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	70	69,5	70	69,5	70	69,5	69,5	70	70
Mmin	61	61	61	61	61	61	61	61	61
MExt	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5
BOW	64,5	68	65,5	64,5	62,5	66,5	65	69,5	67
CKNN	61	61	61	61	61	61	61	61	61
mDD	47,5	55,5	49	57	55	57	51,5	55,5	52,5
EMDD	56	55,5	58,5	56	55	56	50,5	55	55,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.122: Precisión MIL-IPF Tiger, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.123: Precisión MIL-IPF Tiger, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	66,5	66,5	66,5	66,5	66,5	66,5	66,5	66,5	66,5
Mmin	60,5	60,5	60,5	60,5	60,5	60,5	60,5	60,5	60,5
MExt	60,5	60,5	60,5	60,5	60,5	60,5	60,5	60,5	60,5
BOW	62	58	64,5	66,5	56	62,5	64,5	65	62
CKNN	68	68	68	68	68	68	68	68	68
mDD	53	52	50	53	53	49	55	56,5	54,5
EMDD	56,5	58,5	58,5	61	56	55,5	53	57,5	51
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.124: Precisión MIL-IPF Tiger, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 30 %.
- Esquema votación por consenso.

Tabla C.125: Precisión MIL-IPF Tiger, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	64	64	64	64	64	64	64	64	64
Mmin	66,5	66,5	66,5	66,5	66,5	66,5	66,5	66,5	66,5
MExt	66	66	66	66	66	66	66	66	66
BOW	59,5	60	64,5	60	61,5	62,5	59,5	60	59
CKNN	61	61	61	61	61	61	61	61	61
mDD	55,5	58,5	52	54,5	53,5	54,5	53	52,5	57
EMDD	49,5	51,5	50	51,5	49,5	49,5	49,5	49,5	53
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.126: Precisión MIL-IPF Tiger, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

C.4. Fox

MIL-EF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.127: Precisión MIL-EF Fox, Consenso, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	67.5	64.0	66.0
MIL min	61.75	60.0	59.75
MIL Ext	62.5	61.0	60.5
BOW	62.25	59.5	60.0
CKNN	62.4	59.6	60.5
maxDD	60.75	58.166666666666664	59.833333333333336
EM-DD	60.785714285714285	58.714285714285715	59.714285714285715
MILBoost	59.4375	57.625	58.5

- Esquema votación por Max Votos.

Tabla C.128: Precisión MIL-EF Fox, Max Votos, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	64.5	62.0	62.0
MIL min	60.5	57.75	57.25
MIL Ext	61.166666666666664	58.666666666666664	58.333333333333336
BOW	60.125	58.25	58.375
CKNN	60.3	58.8	59.3
maxDD	59.833333333333336	59.0	58.416666666666664
EM-DD	59.785714285714285	59.07142857142857	59.0
MILBoost	58.5625	57.9375	57.875

- Ruido 5 %.

- Esquema votación por consenso.

Tabla C.129: Precisión MIL-EF Fox, Consenso, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	65.5	59.0	61.0
MIL min	59.0	55.0	56.25
MIL Ext	59.83333333333336	54.83333333333336	56.5
BOW	59.375	55.125	56.0
CKNN	60.2	56.0	57.3
maxDD	58.83333333333336	54.916666666666664	55.916666666666664
EM-DD	59.142857142857146	55.285714285714285	56.92857142857143
MILBoost	58.0	54.625	56.0625

- Esquema votación por Max Votos.

Tabla C.130: Precisión MIL-EF Fox, Max Votos, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	63.0	55.5	57.5
MIL min	57.75	53.75	54.25
MIL Ext	58.166666666666664	54.0	56.33333333333336
BOW	56.5	54.5	55.875
CKNN	57.6	54.4	56.2
maxDD	56.83333333333336	54.83333333333336	55.916666666666664
EM-DD	57.357142857142854	55.42857142857143	56.42857142857143
MILBoost	56.4375	54.75	55.625

- Ruido 10 %.
 - Esquema votación por consenso.

Tabla C.131: Precisión MIL-EF Fox, Consenso, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	62.0	58.5	61.5
MIL min	56.5	55.25	57.25
MIL Ext	58.5	55.666666666666664	59.5
BOW	56.75	55.625	59.5
CKNN	58.0	56.9	60.4
maxDD	57.416666666666664	56.08333333333336	58.916666666666664
EM-DD	57.214285714285715	56.357142857142854	58.857142857142854
MILBoost	56.3125	55.5625	57.75

- Esquema votación por Max Votos.

Tabla C.132: Precisión MIL-EF Fox, Max Votos, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	58.0	52.5	55.0
MIL min	55.5	53.0	54.0
MIL Ext	56.166666666666664	53.166666666666664	53.833333333333336
BOW	56.25	53.125	53.75
CKNN	57.4	54.1	54.5
maxDD	56.666666666666664	53.916666666666664	53.833333333333336
EM-DD	56.642857142857146	53.92857142857143	54.57142857142857
MILBoost	55.8125	53.4375	54.0

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.133: Precisión MIL-EF Fox, Consenso, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	57.5	56.0	58.0
MIL min	54.25	53.0	54.5
MIL Ext	55.0	54.0	55.0
BOW	55.5	53.875	54.75
CKNN	56.5	54.6	56.1
maxDD	54.75	54.583333333333336	55.5
EM-DD	55.0	55.07142857142857	55.5
MILBoost	54.375	54.4375	54.8125

- Esquema votación por Max Votos.

Tabla C.134: Precisión MIL-EF Fox, Max Votos, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	60.0	55.0	57.0
MIL min	56.0	53.0	54.75
MIL Ext	56.333333333333336	54.666666666666664	55.5
BOW	55.875	54.25	54.25
CKNN	56.2	54.4	54.6
maxDD	55.5	54.916666666666664	54.25
EM-DD	55.785714285714285	55.357142857142854	54.857142857142854
MILBoost	55.0625	54.6875	54.25

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.135: Precisión MIL-EF Fox, Consenso, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	64.0	54.0	60.5
MIL min	57.25	53.5	56.0
MIL Ext	59.166666666666664	54.5	58.0
BOW	57.75	53.375	57.75
CKNN	58.0	54.5	57.9
maxDD	56.916666666666664	54.0	56.75
EM-DD	56.714285714285715	54.357142857142854	56.357142857142854
MILBoost	55.875	53.8125	55.5625

- Esquema votación por Max Votos.

Tabla C.136: Precisión MIL-EF Fox, Max Votos, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	64.5	56.5	56.0
MIL min	58.5	53.25	53.5
MIL Ext	59.0	54.333333333333336	54.333333333333336
BOW	58.5	53.5	55.0
CKNN	59.5	54.0	55.8
maxDD	59.333333333333336	53.916666666666664	56.083333333333336
EM-DD	60.0	54.57142857142857	56.857142857142854
MILBoost	58.75	54.0	56.0

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.137: Precisión MIL-EF Fox, Consenso, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	58.0	52.0	54.0
MIL min	54.0	51.25	52.75
MIL Ext	55.333333333333336	51.333333333333336	53.333333333333336
BOW	55.25	50.75	53.125
CKNN	56.1	52.3	54.0
maxDD	55.333333333333336	51.75	53.166666666666664
EM-DD	55.57142857142857	52.07142857142857	53.642857142857146
MILBoost	54.875	51.8125	53.1875

- Esquema votación por Max Votos.

Tabla C.138: Precisión MIL-EF Fox, Max Votos, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	60.5	56.0	60.0
MIL min	57.0	54.25	55.5
MIL Ext	56.666666666666664	55.0	56.666666666666664
BOW	56.125	55.625	56.625
CKNN	56.0	55.8	57.2
maxDD	54.5	55.416666666666664	57.25
EM-DD	55.142857142857146	55.714285714285715	57.857142857142854
MILBoost	54.5	55.0	56.875

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.139: Precisión MIL-EF Fox, Consenso, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	53.0	53.5	54.5
MIL min	51.75	52.25	53.75
MIL Ext	52.5	52.5	53.333333333333336
BOW	51.75	51.875	52.625
CKNN	52.7	51.7	54.3
maxDD	52.583333333333336	51.416666666666664	54.25
EM-DD	53.357142857142854	51.57142857142857	54.357142857142854
MILBoost	52.9375	51.375	53.8125

- Esquema votación por Max Votos.

Tabla C.140: Precisión MIL-EF Fox, Max Votos, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	54.0	50.5	54.5
MIL min	51.5	50.25	52.25
MIL Ext	51.666666666666664	50.5	52.666666666666664
BOW	51.25	51.25	52.75
CKNN	53.1	51.4	53.6
maxDD	52.416666666666664	51.083333333333336	53.25
EM-DD	52.857142857142854	51.07142857142857	53.214285714285715
MILBoost	52.5	50.9375	52.8125

MIL-CVCF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.141: Precisión MIL-CVCF Fox, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	67	67	67	67	67	67	67	67	67
Mmin	57	57	57	57	57	57	57	57	57
MExt	64	64	64	64	64	64	64	64	64
BOW	60,5	62	58,5	65	61,5	56	57,5	57	65,5
CKNN	62	62	62	62	62	62	62	62	62
mDD	52,5	50,5	54	47	51,5	54	50	51,5	54,5
EMDD	59	61	58	61,5	59,5	61,5	61	59,5	60
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.142: Precisión MIL-CVCF Fox, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	64,5	61	50	58	63,5	65,5	57,5	59	52,5
Mmin	56	51	51	51,5	54,5	53,5	56,5	51,5	51,5
MExt	62	60	50	59	60,5	60,5	58,5	60	51,5
BOW	59	57,5	48,5	61	56	57,5	57	58,5	52
CKNN	64,5	61	57,5	59	66	64	61,5	63	53,5
mDD	51	57,5	51,5	57	54	51	52,5	56	52
EMDD	62	61	53,5	57	60	60,5	56	58	51,5
MILB	50	50	50	50	50	50	50	50	0

- Ruido 5 %.
- Esquema votación por consenso.

Tabla C.143: Precisión MIL-CVCF Fox, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	66	66	66	66	66	66	66	66	66
Mmin	53	53	53	53	53	53	53	53	53
MExt	65	65	65	65	65	65	65	65	65
BOW	54,5	54	54	58,5	55	64	61,5	56,5	59,5
CKNN	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5
mDD	55	54	48	53,5	56,5	56,5	54	55	56,5
EMDD	59,5	57	59	57,5	59,5	54,5	58	57,5	60,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.144: Precisión MIL-CVCF Fox, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	62,5	61,5	50	63,5	66	62,5	60,5	60,5	51
Mmin	52	51,5	49,5	51	53,5	50	53,5	52	51
MExt	60,5	61,5	51,5	58	58	61	51,5	57	50,5
BOW	58,5	58	51	54	55,5	54	49,5	51,5	51
CKNN	61,5	60,5	51,5	59,5	60,5	61	58,5	62,5	55
mDD	51	54,5	53,5	55,5	55,5	55	56	55,5	51
EMDD	62,5	58,5	53,5	58,5	59	58	56	58	50,5
MILB	50,95	50	50	50	50	50	50	50	10,475

- Ruido 10 %.
 - Esquema votación por consenso.

Tabla C.145: Precisión MIL-CVCF Fox, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	62,5	62,5	62,5	62,5	62,5	62,5	62,5	62,5	62,5
Mmin	54	54	54	54	54	54	54	54	54
MExt	59,5	59,5	59,5	59,5	59,5	59,5	59,5	59,5	59,5
BOW	59	63	59,5	55	53	62	59	57,5	53,5
CKNN	59,5	59,5	59,5	59,5	59,5	59,5	59,5	59,5	59,5
mDD	50	54	54	53,5	54	52	53,5	56,5	59,5
EMDD	56,5	57	58,5	58,5	61	61,5	63	60	60
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.146: Precisión MIL-CVCF Fox, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	65,5	61,5	50	59	60,5	62	54,5	53	51
Mmin	53,5	50,5	50	51	52,5	53	50,5	51,5	52
MExt	53	54	50	52,5	53,5	55	51	58	51
BOW	56,5	56	50	53,5	55	52,5	49	54	51,5
CKNN	61	57,5	54,5	58,5	60	60	63,5	57	52,5
mDD	53	55	52,5	51,5	53,5	53,5	50,5	55,5	51
EMDD	60	59,5	53	57	59	55,5	52	57,5	52
MILB	50,475	50	49,525	49,525	49,525	49,525	49,525	49,525	10

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.147: Precisión MIL-CVCF Fox, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5
Mmin	53	53	53	53	53	53	53	53	53
MExt	62	62	62	62	62	62	62	62	62
BOW	56	50,5	51	43,5	56	54	53,5	52	52,5
CKNN	65,5	65,5	65,5	65,5	65,5	65,5	65,5	65,5	65,5
mDD	51	51,5	51	53	47,5	50	53,5	49,5	53,5
EMDD	58	59	56,5	59,5	56,5	61,5	58	58	57,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.148: Precisión MIL-CVCF Fox, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	61,5	58	50,5	56,5	59	58	52	52	50,5
Mmin	51,5	53	50	51	51	53	51	51	49,5
MExt	60,5	56	49,5	55,5	60,5	56,5	56,5	57,5	51
BOW	52,5	51	49	58	49,5	48,5	50	55	52,5
CKNN	57,5	57	52,5	58	59	56	50	53,5	51
mDD	53,5	51,5	50,5	54,5	56	51,5	56	57	52
EMDD	56	59,5	53,5	60	58,5	61	53,5	59	50,5
MILB	50,475	50	39,525	49,525	49,525	49,525	49,05	49,05	20,95

- Ruido 20 %.
- Esquema votación por consenso.

Tabla C.149: Precisión MIL-CVCF Fox, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	63	63	63	63	63	63	63	63	63
Mmin	51,5	51,5	51,5	51,5	51,5	51,5	51,5	51,5	51,5
MExt	58,5	58,5	58,5	58,5	58,5	58,5	58,5	58,5	58,5
BOW	50,5	48	52	45	54	54,5	53	58,5	53
CKNN	57	57	57	57	57	57	57	57	57
mDD	50,5	50	47,5	52	52	53,5	51,5	54,5	52,5
EMDD	56,5	57,5	59	54	58	57	58	56	57
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.150: Precisión MIL-CVCF Fox, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	60,5	61	52,5	55,5	58	57,5	52	54,5	51,5
Mmin	51	50,5	50,5	51	52	53,5	49,5	52	51
MExt	60	57,5	50,5	58	54,5	54	51,5	55	52
BOW	53	58	51,5	50	50,5	53	52,5	51,5	50,5
CKNN	61	56	53	58	58	59,5	53,5	55,5	52,5
mDD	53,5	55	50,5	54,5	48,5	48	52	54,5	52
EMDD	57,5	58	51,5	55	58	58	52,5	58	51,5
MILB	50	50	39,525	49,05	49,05	49,05	49,05	49,05	20,95

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.151: Precisión MIL-CVCF Fox, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	60	60	60	60	60	60	60	60	60
Mmin	50,5	50,5	50,5	50,5	50,5	50,5	50,5	50,5	50,5
MExt	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5
BOW	55,5	57	61	58,5	53,5	55,5	57,5	54,5	56,5
CKNN	57	57	57	57	57	57	57	57	57
mDD	49,5	48,5	52	49	46,5	47,5	47	51	54
EMDD	56	56,5	58,5	58	56	57,5	62	59,5	59,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.152: Precisión MIL-CVCF Fox, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	58,5	57,5	50	55	53,5	54,5	55,5	53,5	51
Mmin	52,5	54	51	55,5	52,5	50,5	51,5	50	50,5
MExt	56,5	53,5	50,5	50,5	49	52	51,5	51	51,5
BOW	58	52,5	50	54,5	58,5	54,5	53,5	51,5	52
CKNN	60,5	58	50,5	58	58,5	60,5	55,5	58,5	53,5
mDD	56	54	50,5	51	54,5	51	54,5	55,5	51,5
EMDD	57	54,5	52	55,5	54,5	57	51,5	55,5	51
MILB	50	50	40	50	50	50	49,05	49,05	20,475

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.153: Precisión MIL-CVCF Fox, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	55	55	55	55	55	55	55	55	55
Mmin	50,5	50,5	50,5	50,5	50,5	50,5	50,5	50,5	50,5
MExt	56,5	56,5	56,5	56,5	56,5	56,5	56,5	56,5	56,5
BOW	56	51,5	57	54,5	51	56	55	56	54
CKNN	51,5	51,5	51,5	51,5	51,5	51,5	51,5	51,5	51,5
mDD	51	50,5	56	51	51,5	54,5	48	53,5	58,5
EMDD	55,5	56	55,5	56	51,5	56	51,5	54	53,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.154: Precisión MIL-CVCF Fox, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	54,5	54	50	55	51,5	55	54	53	52
Mmin	52,5	50,5	50,5	50,5	50,5	50,5	52	50,5	52
MExt	58	51	51,5	51	58	57,5	53,5	58	49,5
BOW	55,5	52	50,5	53	54,5	54	52,5	52,5	50,5
CKNN	55	56,5	49	58	56,5	53	55,5	58	50,5
mDD	55,5	53	51	53,5	53,5	52,5	55	55	48,5
EMDD	57	55,5	51,5	53	56	53	55,5	53,5	52
MILB	48,575	50	49,525	49,525	49,525	49,525	48,575	48,575	0

MIL-IPF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.155: Precisión MIL-IPF Fox, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	62	62	62	62	62	62	62	62	62
Mmin	56,5	56,5	56,5	56,5	56,5	56,5	56,5	56,5	56,5
MExt	59,5	59,5	59,5	59,5	59,5	59,5	59,5	59,5	59,5
BOW	56,5	57	58,5	62,5	60	59,5	55	56	59,5
CKNN	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5
mDD	53	52	51,5	49,5	54	50,5	51	55	53
EMDD	62	60,5	60,5	61	58,5	61,5	60,5	58,5	58
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.156: Precisión MIL-IPF Fox, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	65,5	56,5	50	55	64,5	66,5	55,5	50	51,5
Mmin	59	51,5	50,5	50,5	52	52,5	52,5	51	53
MExt	64	55	50	54,5	57,5	64,5	53	50,5	51,5
BOW	58	52,5	50	56	56,5	56	51,5	50	51,5
CKNN	61	58,5	51,5	58,5	60,5	59,5	53,5	52,5	53,5
mDD	52,5	57,5	50,5	54,5	55	53,5	53,5	52	52,5
EMDD	60	58,5	51	56,5	61,5	56	54,5	52,5	52
MILB	50,95	50	40	50	49,525	49,525	48,575	48,575	50,95

- Ruido 5 %.
- Esquema votación por consenso.

Tabla C.157: Precisión MIL-IPF Fox, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	61	61	61	61	61	61	61	61	61
Mmin	53,5	53,5	53,5	53,5	53,5	54	53,5	53,5	53,5
MExt	59	59	59	59	59	59	59	59	59
BOW	53	59,5	56	55,5	61	56	54,5	59	55
CKNN	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5
mDD	50,5	51,5	51,5	50,5	49	54	55,5	51,5	50
EMDD	60	60,5	59	58	60,5	59	58	57,5	60,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.158: Precisión MIL-IPF Fox, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 10 %.
- Esquema votación por consenso.

Tabla C.159: Precisión MIL-IPF Fox, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	64	64	64	64	64	64	64	64	64
Mmin	51	51	51	51	51	51	51	51	51
MExt	60,5	60,5	60,5	60,5	61	60,5	60,5	60,5	60,5
BOW	56,5	59	56	61	62	56,5	58	59,5	54,5
CKNN	65,5	65,5	65,5	65,5	65,5	65,5	65,5	65,5	65,5
mDD	54,5	49,5	53	53,5	51,5	50,5	52	52	49
EMDD	57	56,5	56	56	57,5	58,5	57	57,5	58,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.160: Precisión MIL-IPF Fox, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 15 %.
- Esquema votación por consenso.

Tabla C.161: Precisión MIL-IPF Fox, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5
Mmin	51,5	51,5	51,5	51,5	51,5	51,5	51,5	51,5	51,5
MExt	54	54	54	54	54	54	54	54	54
BOW	63	59,5	60,5	55,5	58	64	60	57,5	59
CKNN	56	56	56	56	56	56	56	56	56
mDD	55	51,5	54,5	54,5	54	52	48,5	51,5	52
EMDD	56,5	55	57,5	53,5	57,5	54,5	56	53,5	54,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.162: Precisión MIL-IPF Fox, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.163: Precisión MIL-IPF Fox, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	58	58	58	58	58	58	58	58	58
Mmin	51	51	51	51	51	51	51	51	51
MExt	59	59	59	59	59	59	59	59	59
BOW	54	55	53,5	50	54	52	55,5	56	51,5
CKNN	59	59	59	59	59	59	59	59	59
mDD	54,5	56,5	57,5	51,5	51	55,5	53	51	56,5
EMDD	58	53,5	53,5	52	56	55	60	55	55,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.164: Precisión MIL-IPF Fox, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.165: Precisión MIL-IPF Fox, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	53	53	53	53	53	53	53	53	53
Mmin	50,5	50,5	50,5	50,5	50,5	50,5	50,5	50,5	50,5
MExt	60,5	60,5	60,5	60,5	60,5	60,5	60,5	60,5	60,5
BOW	48,5	58	55,5	55	53,5	55,5	56,5	52	52
CKNN	55,5	55,5	55,5	55,5	55,5	55,5	55,5	55,5	55,5
mDD	48,5	50,5	48,5	45,5	48	51,5	45,5	51	47,5
EMDD	56	55	54	53	54	56	58,5	54,5	54,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.166: Precisión MIL-IPF Fox, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	65	52	51,5	51,5	53,5	56	56	48	50
Mmin	54	50	51	51,5	53,5	52	54	48,5	50,5
MExt	58	51	50	51,5	52,5	52,5	57,5	55	53
BOW	61	51	51	50,5	54,5	53	55,5	48,5	51
CKNN	53	49,5	50,5	48	54,5	54	51,5	51	54,5
mDD	56,5	53,5	50,5	52,5	52,5	54	55	55,5	55,5
EMDD	59,5	55	51,5	53	54,5	61	54,5	54,5	56
MILB	53,325	50	40	50	50,475	50,475	50,95	50,95	41,9

- Ruido 30 %.
- Esquema votación por consenso.

Tabla C.167: Precisión MIL-IPF Fox, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	56	56	56	56	56	56	56	56	56
Mmin	49,5	49,5	49,5	49,5	49,5	49,5	49,5	49,5	49,5
MExt	52,5	52,5	52,5	52,5	52,5	52,5	52,5	52,5	52,5
BOW	51	49	53	51,5	54,5	54,5	54	55,5	49,5
CKNN	62,5	62,5	62,5	62,5	62,5	62,5	62,5	62,5	62,5
mDD	50	50,5	49,5	52,5	46	47,5	49,5	51,5	49
EMDD	55	52	54	52	54	52,5	54	50,5	51,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.168: Precisión MIL-IPF Fox, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

C.5. Elephant

MIL-EF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.169: Precisión MIL-EF Elephant, Consenso, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	79.0	78.5	80.5
MIL min	74.5	73.25	73.75
MIL Ext	77.66666666666667	76.0	77.16666666666667
BOW	76.625	75.875	76.0
CKNN	76.7	75.8	76.1
maxDD	74.66666666666667	73.83333333333333	74.91666666666667
EM-DD	74.92857142857143	74.35714285714286	75.28571428571429
MILBoost	71.8125	71.3125	72.125

- Esquema votación por Max Votos.

Tabla C.170: Precisión MIL-EF Max Votos, Consenso, Ruido 0 %

	Original	Filtro_1	Filtro_2
MIL max	79.5	72.5	76.5
MIL min	75.75	74.75	68.25
MIL Ext	77.33333333333333	74.83333333333333	71.33333333333333
BOW	76.625	75.5	71.25
CKNN	76.5	75.3	72.2
maxDD	75.58333333333333	74.5	70.58333333333333
EM-DD	75.85714285714286	75.14285714285714	71.5
MILBoost	72.625	72.0	68.8125

- Ruido 5 %.

- Esquema votación por consenso.

Tabla C.171: Precisión MIL-EF Elephant, Consenso, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	79.0	73.0	76.0
MIL min	74.0	71.0	69.75
MIL Ext	75.33333333333333	72.33333333333333	73.16666666666667
BOW	75.0	73.125	72.125
CKNN	75.2	73.7	73.1
maxDD	73.66666666666667	72.5	71.83333333333333
EM-DD	73.92857142857143	72.64285714285714	72.92857142857143
MILBoost	70.9375	69.8125	70.0625

- Esquema votación por Max Votos.

Tabla C.172: Precisión MIL-EF Max Votos, Consenso, Ruido 5 %

	Original	Filtro_1	Filtro_2
MIL max	80.0	74.5	79.5
MIL min	75.5	69.75	69.5
MIL Ext	77.66666666666667	72.33333333333333	72.5
BOW	77.5	73.125	71.625
CKNN	77.6	73.5	73.1
maxDD	76.16666666666667	72.66666666666667	71.75
EM-DD	76.21428571428571	73.14285714285714	71.71428571428571
MILBoost	72.9375	70.25	69.0

- Ruido 10 %.
 - Esquema votación por consenso.

Tabla C.173: Precisión MIL-EF Elephant, Consenso, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	78.5	73.5	76.0
MIL min	72.5	70.5	69.0
MIL Ext	75.0	73.33333333333333	72.83333333333333
BOW	73.75	73.0	72.25
CKNN	73.8	73.9	73.1
maxDD	72.75	73.16666666666667	72.41666666666667
EM-DD	72.78571428571429	73.5	72.5
MILBoost	69.9375	70.5625	69.6875

- Esquema votación por Max Votos.

Tabla C.174: Precisión MIL-EF Max Votos, Consenso, Ruido 10 %

	Original	Filtro_1	Filtro_2
MIL max	79.0	70.0	71.5
MIL min	71.25	61.5	66.25
MIL Ext	74.16666666666667	65.0	69.0
BOW	74.25	65.875	69.25
CKNN	75.3	67.6	70.6
maxDD	73.16666666666667	66.25	69.83333333333333
EM-DD	73.0	67.35714285714286	70.57142857142857
MILBoost	70.125	65.1875	68.0

- Ruido 15 %.
 - Esquema votación por consenso.

Tabla C.175: Precisión MIL-EF Elephant, Consenso, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	74.5	71.5	73.0
MIL min	66.5	67.0	64.0
MIL Ext	69.33333333333333	68.33333333333333	67.5
BOW	69.875	68.0	68.0
CKNN	68.8	68.2	68.7
maxDD	67.41666666666667	66.58333333333333	68.08333333333333
EM-DD	68.14285714285714	67.64285714285714	68.71428571428571
MILBoost	65.875	65.4375	66.375

- Esquema votación por Max Votos.

Tabla C.176: Precisión MIL-EF Max Votos, Consenso, Ruido 15 %

	Original	Filtro_1	Filtro_2
MIL max	75.0	69.5	69.5
MIL min	68.5	62.0	66.5
MIL Ext	70.0	65.0	69.0
BOW	71.25	66.5	70.0
CKNN	72.4	67.6	71.5
maxDD	71.16666666666667	66.91666666666667	70.75
EM-DD	71.64285714285714	68.0	71.07142857142857
MILBoost	68.9375	65.75	68.4375

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.177: Precisión MIL-EF Elephant, Consenso, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	70.5	68.5	71.5
MIL min	65.25	62.5	64.25
MIL Ext	68.5	65.16666666666667	66.5
BOW	68.875	65.625	66.125
CKNN	69.1	66.2	66.8
maxDD	67.75	65.58333333333333	65.83333333333333
EM-DD	67.85714285714286	66.07142857142857	66.0
MILBoost	65.625	64.0625	64.0

- Esquema votación por Max Votos.

Tabla C.178: Precisión MIL-EF Max Votos, Consenso, Ruido 20 %

	Original	Filtro_1	Filtro_2
MIL max	72.5	64.5	70.5
MIL min	67.25	60.75	62.75
MIL Ext	70.16666666666667	63.16666666666664	65.0
BOW	70.625	64.125	66.375
CKNN	70.6	65.3	67.8
maxDD	69.33333333333333	66.0	67.08333333333333
EM-DD	69.5	66.85714285714286	67.42857142857143
MILBoost	67.0625	64.75	65.25

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.179: Precisión MIL-EF Elephant, Consenso, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	72.0	66.5	69.5
MIL min	65.5	62.5	63.5
MIL Ext	68.16666666666667	64.66666666666667	66.0
BOW	67.5	65.0	64.875
CKNN	67.4	66.2	66.0
maxDD	66.41666666666667	65.41666666666667	65.33333333333333
EM-DD	66.35714285714286	65.28571428571429	65.57142857142857
MILBoost	64.3125	63.375	63.625

- Esquema votación por Max Votos.

Tabla C.180: Precisión MIL-EF Max Votos, Consenso, Ruido 25 %

	Original	Filtro_1	Filtro_2
MIL max	69.0	61.5	68.5
MIL min	61.75	58.0	65.25
MIL Ext	64.83333333333333	59.33333333333336	67.0
BOW	63.875	59.625	65.875
CKNN	64.7	61.0	67.5
maxDD	64.41666666666667	60.91666666666664	67.66666666666667
EM-DD	65.0	62.642857142857146	68.28571428571429
MILBoost	63.125	61.0625	66.0

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.181: Precisión MIL-EF Elephant, Consenso, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	61.5	65.5	70.0
MIL min	57.75	58.75	61.0
MIL Ext	61.166666666666664	61.33333333333336	64.0
BOW	62.25	61.625	64.5
CKNN	64.1	63.1	66.9
maxDD	62.83333333333336	62.33333333333336	66.0
EM-DD	63.92857142857143	62.57142857142857	66.07142857142857
MILBoost	62.1875	61.0	64.0625

- Esquema votación por Max Votos.

Tabla C.182: Precisión MIL-EF Max Votos, Consenso, Ruido 30 %

	Original	Filtro_1	Filtro_2
MIL max	71.5	62.0	68.0
MIL min	61.5	56.0	62.25
MIL Ext	64.16666666666667	58.16666666666664	64.33333333333333
BOW	66.125	58.875	64.5
CKNN	65.9	59.6	65.1
maxDD	65.16666666666667	59.16666666666664	64.83333333333333
EM-DD	65.21428571428571	60.642857142857146	65.07142857142857
MILBoost	63.3125	59.3125	63.1875

MIL-CVCF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.183: Precisión MIL-CVCF Elephant, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	80,5	80,5	80,5	80,5	80,5	80,5	80,5	80,5	80,5
Mmin	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5
MExt	83	83	83	83	83	83	83	83	83
BOW	71,5	73,5	72,5	76	73,5	75	74,5	70,5	73
CKNN	79,5	79,5	79,5	79,5	79,5	79,5	79,5	79,5	79,5
mDD	62,5	65	69	70	70	64	65,5	65,5	69
EMDD	76,5	77,5	60	75	75,5	75	75	76	76,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.184: Precisión MIL-CVCF Elephant, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	80	79	65,5	80	78	79,5	72,5	77	51,5
Mmin	70	68	57,5	65	63	72,5	62	60,5	51
MExt	82,5	81,5	67,5	83	82	81,5	75,5	78	51
BOW	70	72,5	61	69	70,5	72	73	67,5	50,5
CKNN	80	79,5	74,5	81,5	77	80	75,5	81,5	55
mDD	63	68,5	62	68,5	64,5	70	66,5	61,5	52
EMDD	75,5	76	68,5	74	76,5	76,5	74	75,5	52,5
MILB	50,95	50	50	50	50	50	50	50	10,475

- Ruido 5 %.
- Esquema votación por consenso.

Tabla C.185: Precisión MIL-CVCF Elephant, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	73,5	73,5	73,5	73,5	73,5	73,5	73,5	73,5	73,5
Mmin	68	67,5	67,5	67,5	68	67,5	68	67,5	67,5
MExt	81	81,5	81,5	81,5	81	81	81,5	81	81,5
BOW	68	71,5	71	69,5	69,5	68	67,5	69,5	72
CKNN	77,5	77,5	77,5	77,5	77,5	77,5	77,5	77,5	77,5
mDD	67,5	60	66,5	64	59	66,5	66,5	66,5	62,5
EMDD	74	74	76	74	75,5	74	76,5	76	79
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.186: Precisión MIL-CVCF Elephant, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	78	77	61,5	77,5	75,5	79,5	71,5	72	51,5
Mmin	73	69,5	59,5	69,5	66	71,5	60	59	52
MExt	79	79	62,5	80	78,5	78,5	73	76,5	50,5
BOW	75,5	71,5	60	71,5	72	75	65,5	70	52
CKNN	75,5	76,5	70,5	77	76	74	68	74,5	57
mDD	69	64	58,5	63	64	62	65,5	64,5	51,5
EMDD	76,5	74,5	64	77,5	74	77,5	74,5	72,5	51
MILB	51,9	50	50	50	50	50	50	50	10,475

- Ruido 10 %.
- Esquema votación por consenso.

Tabla C.187: Precisión MIL-CVCF Elephant, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	78	78	78	78	78	78	78	78	78
Mmin	64	64	64	64	64	64	64	64	64
MExt	78,5	78,5	78,5	78,5	78,5	78,5	78,5	78,5	78,5
BOW	74,5	77	76	75	72	71	76,5	76	73
CKNN	72,5	72,5	72,5	72,5	72,5	72,5	72,5	72,5	72,5
mDD	67	66,5	64,5	66	66,5	65	61	60,5	59
EMDD	72	70	74	71	70	71,5	72	72	73
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.188: Precisión MIL-CVCF Elephant, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	79	78	57	76,5	77	76,5	72	72	51,5
Mmin	72	69	56,5	67,5	66	64,5	54	57,5	51,5
MExt	82,5	79,5	56,5	79,5	80,5	82	71,5	73	52
BOW	74	69,5	51,5	71,5	67,5	75,5	60,5	62,5	50
CKNN	74	74	70	77	76,5	74	74,5	74	55,5
mDD	59	66,5	58,5	71,5	70	68	61	62	51,5
EMDD	74	72,5	57	72,5	71,5	74,5	70	68,5	51,5
MILB	51,9	50	50	50	50	50	50	50	41,9

- Ruido 15 %.
- Esquema votación por consenso.

Tabla C.189: Precisión MIL-CVCF Elephant, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	77	77	77	77	77	77	77	77	77
Mmin	63	63	63	63	63	63	63	63	63
MExt	77	77	77	77	77	77	77	77	77
BOW	71,5	72	71	72	68	74,5	72,5	73,5	73
CKNN	73	73	73	73	73	73	73	73	73
mDD	62	62,5	67,5	61	65	65,5	66,5	65,5	66,5
EMDD	71	68,5	70	71	73,5	71	69	70	70
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.190: Precisión MIL-CVCF Elephant, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	78	70	53	70	67	78	69	66	51,5
Mmin	63,5	66	52	63	61	61,5	57	57,5	52
MExt	81	80,5	56	79	76	78,5	68,5	64,5	51,5
BOW	68	67	58,5	70	65,5	69,5	59,5	66	49,5
CKNN	71	75	68	73	72	71	73,5	73	56,5
mDD	61	67,5	56	70	62	62	59	62	51
EMDD	66	69	60	72	70,5	71	65	67	52,5
MILB	51,9	50	50	50	50	50	50	50	41,9

- Ruido 20 %.
- Esquema votación por consenso.

Tabla C.191: Precisión MIL-CVCF Elephant, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	74	74	74	74	74	74	74	74	74
Mmin	59	59	59	59	59	59	59	59	59
MExt	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5	74,5
BOW	69,5	71,5	62,5	70	67	66,5	69	70,5	70,5
CKNN	68	68	68	68	68	68	68	68	68
mDD	53,5	60,5	65	63	66	60	59,5	60	66,5
EMDD	72	69,5	67	67,5	67,5	69	71,5	69	65,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.192: Precisión MIL-CVCF Elephant, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	76	74	50	72	71	76	63,5	64,5	51
Mmin	57	57,5	53	53,5	60	55,5	51	60	51,5
MExt	72	71,5	50	71,5	69,5	71	56,5	63,5	51
BOW	64	68	52,5	68,5	65	65,5	57,5	56,5	50,5
CKNN	69	73,5	58	76	70	70	69,5	64	54
mDD	61,5	64,5	55,5	63,5	63	62	61,5	59	51
EMDD	65,5	69,5	52	66,5	68	73	58	61	51,5
MILB	50,475	50	50	50	50	50	50	50	10,475

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.193: Precisión MIL-CVCF Elephant, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	70	70	70	70	70	70	70	70	70
Mmin	53,5	53,5	53,5	53,5	53,5	53,5	53,5	53,5	53,5
MExt	71,5	71,5	71,5	71,5	71,5	71,5	71,5	71,5	71,5
BOW	65,5	64	66	65	63	67,5	68	65,5	65,5
CKNN	65	65	65	65	65	65	65	65	65
mDD	63	58	58,5	62	54,5	56,5	59	59	56,5
EMDD	67	68	65,5	67	67,5	71,5	68,5	70,5	67
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.194: Precisión MIL-CVCF Elephant, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	73	71,5	50	70,5	69,5	70,5	66,5	65,5	52
Mmin	57,5	56,5	51	53,5	53	64,5	56	52,5	51
MExt	72,5	67	51	67	74	73,5	61	67,5	50
BOW	68	72,5	51,5	61,5	71	67	57	58	50,5
CKNN	66	65	56,5	66,5	69	69	64,5	67,5	57
mDD	52	59	52	59,5	53,5	61,5	58	64,5	49,5
EMDD	66,5	67,5	52	71,5	66	69	54	61,5	52,5
MILB	51,425	50	40	50	50	50	50	50	20,95

- Ruido 30 %.
 - Esquema votación por consenso.

Tabla C.195: Precisión MIL-CVCF Elephant, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69	69	69	69	69	69	69	69	69
Mmin	52	52	52	52	52	52	52	52	52
MExt	67	67	67	67	67	67	67	67	67
BOW	61	58	60,5	59	61	61	64,5	59	60,5
CKNN	62	62	62	62	62	62	62	62	62
mDD	60	60,5	62,5	58	61,5	59,5	60	58,5	62
EMDD	74	73,5	70,5	69	72	69,5	68	67	70,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.196: Precisión MIL-CVCF Elephant, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	61	56	50	54	56	60,5	54,5	55,5	50,5
Mmin	56,5	57,5	50	57,5	54,5	53,5	52,5	53,5	51
MExt	63	62,5	50,5	62	64	63,5	58	55	51
BOW	69,5	67,5	50,5	65	69,5	68	59,5	55,5	51,5
CKNN	59	62,5	54	61,5	62	60	59,5	67,5	53
mDD	59,5	60,5	50	56,5	57	50	57	61,5	52
EMDD	65	60	51,5	61	64	68	59,5	70	51,5
MILB	50,475	50	49,525	49,525	49,525	49,525	49,525	49,525	10

MIL-IPF

- Ruido 0 %.
 - Esquema votación por consenso.

Tabla C.197: Precisión MIL-IPF Elephant, Consenso, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	78,5	78,5	78,5	78,5	78,5	78,5	78,5	78,5	78,5
Mmin	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5
MExt	80,5	80,5	80,5	80,5	80,5	80,5	80,5	80,5	80,5
BOW	73,5	75,5	73	74	73	78	74,5	70,5	73,5
CKNN	81	81	81	81	81	81	81	81	81
mDD	63,5	67,5	64,5	66,5	65	70	61,5	66,5	67
EMDD	76,5	71,5	76,5	76	77	79	73	78	75
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.198: Precisión MIL-IPF Elephant, Max Votos, Ruido 0 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	80,5	80,5	50	81	60,5	80	62	69	52,5
Mmin	69,5	67	50,5	66,5	54	72,5	60	57	53
MExt	84	83,5	50	84	52,5	83,5	67	72,5	54,5
BOW	78	77	50	72	52,5	80	62	66	54,5
CKNN	77	78,5	59	77	67	75	59,5	73	59,5
mDD	60,5	66,5	50	73,5	57	64	62,5	67,5	52,5
EMDD	78	75,5	51,5	76,5	57,5	75,5	67	74,5	53,5
MILB	51,425	50	50	50	50,95	50,95	39,525	49,525	21,425

- Ruido 5 %.
- Esquema votación por consenso.

Tabla C.199: Precisión MIL-IPF Elephant, Consenso, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	79,5	79,5	79,5	79,5	79,5	79,5	79,5	79,5	79,5
Mmin	73,5	73,5	73,5	73,5	73,5	73,5	73,5	73,5	73,5
MExt	80,5	80,5	80,5	80,5	80,5	80,5	80,5	80,5	80,5
BOW	77	72,5	75	73	74	73	74	77	77
CKNN	77,5	77,5	77,5	77,5	77,5	77,5	77,5	77,5	77,5
mDD	57,5	70	64,5	70,5	62	65	64	60	63,5
EMDD	74	72,5	75	72	68,5	72	72,5	73,5	71
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.200: Precisión MIL-IPF Elephant, Max Votos, Ruido 5 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	78	73	50	68	58	79	51	66	53
Mmin	67	61,5	51	62,5	57,5	66	56	56,5	53
MExt	81	79	51	76	64	78,5	53,5	72,5	50
BOW	72	73	50	72	55	72	59,5	61	53
CKNN	76	80	57,5	78	69,5	75	59,5	74,5	57,5
mDD	68,5	66,5	51	68,5	54,5	65,5	55	66,5	53
EMDD	77	75,5	51,5	73,5	55,5	73,5	55	64	53
MILB	51,9	50	50	50	51,9	51,9	40,475	50	41,9

- Ruido 10 %.
- Esquema votación por consenso.

Tabla C.201: Precisión MIL-IPF Elephant, Consenso, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	76	76	76	76	76	76	76	76	76
Mmin	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5	61,5
MExt	78	77,5	78	77,5	78	77,5	77,5	78	77,5
BOW	68	68,5	69	73,5	72,5	72	67,5	71	77,5
CKNN	71	71	71	71	71	71	71	71	71
mDD	66,5	62,5	64,5	71,5	59	63	62	66,5	67,5
EMDD	71	69	71	72,5	72	71	70	70,5	71
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.202: Precisión MIL-IPF Elephant, Max Votos, Ruido 10 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	76,5	74,5	50	70,5	66,5	79,5	58,5	61,5	52,5
Mmin	68	60,5	51	59	52,5	69	55,5	54,5	53
MExt	80	74	50	74	56	78	59,5	57	54,5
BOW	71,5	72	51	71	59,5	78,5	56,5	57,5	51,5
CKNN	73	73,5	57,5	73	59,5	71,5	60	73	56
mDD	69	64,5	51	64	55,5	66	56	58	51
EMDD	78,5	73	52	74	54	76,5	60	62,5	53,5
MILB	51,425	50	39,525	49,525	40,475	50,95	39,525	49,05	51,425

- Ruido 15 %.
- Esquema votación por consenso.

Tabla C.203: Precisión MIL-IPF Elephant, Consenso, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	75,5	75,5	75,5	75,5	75,5	75,5	75,5	75,5	75,5
Mmin	58,5	58,5	58,5	58,5	58,5	58,5	58,5	58,5	58,5
MExt	78,5	78,5	78,5	78,5	78,5	78,5	78,5	78,5	78,5
BOW	69,5	70,5	69,5	67	71	65,5	72,5	65	71
CKNN	73	73	73	73	73	73	73	73	73
mDD	65	61,5	65,5	64,5	63	69,5	65,5	67	58,5
EMDD	73,5	73,5	71,5	73,5	74,5	72,5	71,5	74	73
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.204: Precisión MIL-IPF Elephant, Max Votos, Ruido 15 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	77	76	50,5	76,5	53	78	54,5	58,5	56
Mmin	64	60,5	51	58,5	53,5	64,5	56	53,5	53
MExt	81,5	74,5	51	72,5	54	80	55	67,5	52
BOW	70	71,5	50,5	69,5	52	71	55	53,5	54
CKNN	73,5	76	57,5	75	63	72,5	63,5	67,5	57
mDD	62	61,5	52	60	62,5	57,5	56,5	63	51
EMDD	74	71	52	68,5	53,5	68,5	54,5	59,5	54,5
MILB	53,325	50	29,525	50	41,425	51,9	50,95	50,95	42,375

- Ruido 20 %.
 - Esquema votación por consenso.

Tabla C.205: Precisión MIL-IPF Elephant, Consenso, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	78	78	78	78	78	78	78	78	78
Mmin	59	59	59	59	59	59	59	59	59
MExt	73	73	73	73	73	73	73	73	73
BOW	70	73	72,5	71	74	74	70	68,5	68,5
CKNN	66	66	66	66	66	66	66	66	66
mDD	67	60	59	65	61,5	62,5	64,5	61	67
EMDD	67,5	72	68	70,5	71,5	72,5	72	67	70,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.206: Precisión MIL-IPF Elephant, Max Votos, Ruido 20 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 25 %.
 - Esquema votación por consenso.

Tabla C.207: Precisión MIL-IPF Elephant, Consenso, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	69	69	69	69	69	69	69	69	69
Mmin	53,5	53,5	53,5	53,5	53,5	53,5	53,5	53,5	53,5
MExt	69	69	69	69	69	69	69	69	69
BOW	66,5	69,5	66	59,5	61,5	63	56,5	62,5	63,5
CKNN	66,5	66,5	66,5	66,5	66,5	66,5	66,5	66,5	66,5
mDD	61,5	57	63	64,5	63,5	55	65	58,5	55,5
EMDD	64	67	64	66	65,5	65	67,5	63,5	66
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.208: Precisión MIL-IPF Elephant, Max Votos, Ruido 25 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	0	0	0	0	0	0	0	0	0
Mmin	0	0	0	0	0	0	0	0	0
MExt	0	0	0	0	0	0	0	0	0
BOW	0	0	0	0	0	0	0	0	0
CKNN	0	0	0	0	0	0	0	0	0
mDD	0	0	0	0	0	0	0	0	0
EMDD	0	0	0	0	0	0	0	0	0
MILB	0	0	0	0	0	0	0	0	0

- Ruido 30 %.
- Esquema votación por consenso.

Tabla C.209: Precisión MIL-IPF Elephant, Consenso, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	70,5	70,5	70,5	70,5	70,5	70,5	70,5	70,5	70,5
Mmin	51,5	51,5	51,5	51,5	51,5	51,5	51,5	51,5	51,5
MExt	62	62	62	62	62	62	62	62	62
BOW	66,5	64,5	64	64	65	63,5	63,5	63,5	64
CKNN	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5	68,5
mDD	59	57	66	58	60	64,5	62,5	62,5	64
EMDD	65	63,5	63,5	63,5	64,5	68	66,5	65	65,5
MILB	50	50	50	50	50	50	50	50	50

- Esquema votación por Max Votos.

Tabla C.210: Precisión MIL-IPF Elephant, Max Votos, Ruido 30 %

	Orig	Mmax	Mmin	MExt	BOW	CKNN	mDD	EMDD	MILB
Mmax	76,5	69,5	51	56,5	58	71	56,5	55	49,5
Mmin	57	50,5	51,5	51,5	54,5	55,5	55,5	53	53,5
MExt	71	61	51	54	53,5	63,5	57	57,5	55
BOW	67,5	57,5	51	55,5	53	61,5	56,5	49	50
CKNN	62	61	52,5	59	62,5	67	55,5	57,5	55
mDD	59	55	50	56,5	52,5	60	56	56,5	55
EMDD	71,5	59,5	51,5	60	61,5	64	54	58	51
MILB	53,325	50	30	50	51,425	51,425	51,425	51,425	42,375

Anexos D

Tablas Detalladas

A continuación se muestra la información combinando los clasificadores con el nivel de ruido y los esquemas de votación, de forma que señalaremos el mejor y peor resultado en cada caso.

D.1. Ruido 0 %

Clasificador SimpleMIL-max

Tabla D.1: Detalles Ruido 0 MIL max

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	72,84	74,80	76,08	75,67	70,42	70,06
Muta 2	81,31	69,29	69,29	81,31	56,83	0,00
Tiger	76,50	74,50	73,00	71,50	68,06	0,00
Fox	65,00	67,00	62,00	62,00	58,38	56,19
Elephant	79,50	80,50	78,50	74,50	72,88	66,94

Clasificador SimpleMIL-min

Tabla D.2: Detalles Ruido 0 MIL MIN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	77,89	84,68	82,75	78,63	79,37	76,76
Muta 2	81,31	69,29	69,29	81,31	59,61	0,00
Tiger	76,88	78,00	76,00	74,50	68,75	0,00
Fox	59,88	57,00	56,50	57,50	52,63	51,69
Elephant	73,50	68,50	68,50	71,50	62,44	60,06

Clasificador SimpleMIL-extreme

Tabla D.3: Detalles Ruido 0 MIL EXTREME

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	79,96	86,84	84,85	80,17	77,55	76,77
Muta 2	81,31	69,29	69,29	81,31	57,11	0,00
Tiger	76,92	75,00	74,50	73,58	70,38	0,00
Fox	60,75	64,00	59,50	58,50	57,50	54,56
Elephant	76,58	83,00	80,50	73,08	75,00	68,44

Clasificador BOW

Tabla D.4: Detalles Ruido 0 BOW						
Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	76,84	65,34	64,47	75,45	60,69	58,94
Muta 2	81,31	78,45	77,86	81,31	55,28	0,00
Tiger	76,06	69,81	69,38	72,50	66,56	0,00
Fox	59,75	60,38	58,50	58,31	56,00	53,00
Elephant	75,94	73,56	74,00	73,38	67,00	64,25

Clasificador CKNN

Tabla D.5: Detalles Ruido 0 CKNN						
Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	78,95	86,84	85,96	77,65	85,96	82,23
Muta 2	82,04	72,46	74,76	82,26	86,47	0,00
Tiger	75,35	75,50	70,00	72,65	71,50	0,00
Fox	60,05	62,00	63,50	59,05	60,69	56,00
Elephant	75,95	79,50	81,00	73,75	75,56	68,56

Clasificador maxDD

Tabla D.6: Detalles Ruido 0 MAXDD						
Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	78,37	75,73	73,08	77,21	72,17	69,36
Muta 2	81,92	69,29	69,29	82,10	57,42	0,00
Tiger	71,54	53,25	54,13	69,38	52,25	0,00
Fox	59,00	51,63	52,06	58,71	53,94	53,63
Elephant	74,38	67,25	66,06	72,54	64,19	61,69

Clasificador EMDD

Tabla D.7: Detalles Ruido 0 EMDD						
Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	79,35	85,15	84,23	78,07	78,90	77,28
Muta 2	81,83	68,97	69,04	82,35	57,47	0,00
Tiger	70,46	62,63	61,44	68,89	62,75	0,00
Fox	59,21	60,25	59,88	59,04	57,19	55,31
Elephant	74,82	73,81	75,75	73,32	71,69	66,44

Tabla D.8: Detalles Ruido 0 BOOST

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	75,80	50,94	50,94	74,68	46,00	46,55
Muta 2	76,95	35,41	35,41	77,55	6,69	0,00
Tiger	67,91	50,00	50,00	66,53	48,99	0,00
Fox	58,06	50,00	50,00	57,91	43,75	48,39
Elephant	71,72	50,00	50,00	70,41	45,06	45,30

Clasificador MILBoost

D.2. Ruido 5 %

Clasificador SimpleMIL-max

Tabla D.9: Detalles Ruido 5 MIL MAX

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	75,18	78,42	72,87	72,40	67,85	66,37
Muta 2	81,31	69,29	69,29	81,31	61,83	0,00
Tiger	74,25	73,00	68,00	67,75	65,50	0,00
Fox	60,00	66,00	61,00	56,50	59,44	0,00
Elephant	74,50	73,50	79,50	77,00	70,75	62,25

Clasificador SimpleMIL-min

Tabla D.10: Detalles Ruido 5 MIL MIN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	78,13	85,85	84,62	76,49	74,63	75,82
Muta 2	81,31	69,29	69,29	81,31	55,20	0,00
Tiger	74,50	74,50	77,00	69,63	65,81	0,00
Fox	55,63	53,00	53,56	54,00	51,50	0,00
Elephant	70,38	67,63	73,50	69,63	63,38	58,00

Clasificador SimpleMIL-extreme

Tabla D.11: Detalles Ruido 5 MIL EXTREME

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	79,84	87,13	83,63	78,94	77,34	73,98
Muta 2	81,31	69,29	69,29	81,31	60,00	0,00
Tiger	74,25	67,88	70,00	69,17	65,94	0,00
Fox	55,67	65,00	59,00	55,17	56,13	0,00
Elephant	72,75	81,31	80,50	72,42	72,31	65,56

Clasificador BOW

Tabla D.12: Detalles Ruido 5 BOW

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	75,39	65,83	63,22	74,33	57,20	57,49
Muta 2	81,31	76,79	74,01	81,31	54,97	0,00
Tiger	73,56	70,31	69,75	68,38	64,56	0,00
Fox	55,56	57,88	57,06	55,19	53,06	0,00
Elephant	72,63	69,81	74,44	72,38	67,19	61,94

Clasificador CKNN

Tabla D.13: Detalles Ruido 5 CKNN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	77,29	85,91	90,23	76,31	80,04	78,38
Muta 2	81,53	74,48	72,82	81,53	87,12	0,00
Tiger	74,00	74,00	71,00	69,50	70,00	0,00
Fox	56,65	61,50	61,50	55,30	58,63	0,00
Elephant	73,40	77,50	77,50	73,30	71,69	68,94

Clasificador maxDD

Tabla D.14: Detalles Ruido 5 MAXDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	77,29	74,52	65,76	75,39	70,19	67,62
Muta 2	81,49	69,29	69,29	81,49	57,11	0,00
Tiger	69,96	54,13	55,63	66,50	52,88	0,00
Fox	55,42	54,25	51,69	55,38	54,56	0,00
Elephant	72,17	63,94	64,94	72,21	61,63	60,06

Clasificador EMDD

Tabla D.15: Detalles Ruido 5 EMDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	77,98	84,96	84,02	76,50	74,76	69,94
Muta 2	81,63	70,12	70,99	81,63	54,61	0,00
Tiger	68,71	62,81	57,63	66,04	60,50	0,00
Fox	56,11	57,94	59,13	55,93	56,50	0,00
Elephant	72,79	75,63	72,13	72,43	70,69	62,69

Clasificador MILBoost

Tabla D.16: Detalles Ruido 5 BOOST

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	74,60	50,94	50,94	73,31	47,32	44,93
Muta 2	75,95	35,41	35,41	76,77	6,78	0,00
Tiger	66,38	50,00	50,00	64,03	45,06	0,00
Fox	55,34	50,00	50,00	55,19	45,06	0,00
Elephant	69,94	50,00	50,00	69,63	45,06	48,27

D.3. Ruido 10 %

Clasificador SimpleMIL-max

Tabla D.17: Detalles Ruido 10 MIL MAX

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	64,77	78,30	71,58	70,58	69,09	66,85
Muta 2	81,31	69,29	69,29	83,85	58,50	0,00
Tiger	69,75	73,00	71,00	68,50	63,44	0,00
Fox	60,00	62,50	64,00	53,75	56,44	0,00
Elephant	74,75	78,00	76,00	70,75	70,06	64,19

Clasificador SimpleMIL-min

Tabla D.18: Detalles Ruido 10 MIL MIN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	70,25	74,74	81,52	74,72	77,31	75,20
Muta 2	81,31	69,29	69,29	83,85	51,83	0,00
Tiger	65,50	72,50	75,00	68,88	63,44	0,00
Fox	56,25	54,00	51,00	53,50	51,38	0,00
Elephant	69,75	64,00	61,50	63,88	60,81	56,88

Clasificador SimpleMIL-extreme

Tabla D.19: Detalles Ruido 10 MIL EXTREME

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	71,89	84,80	85,79	76,22	76,73	71,64
Muta 2	81,31	69,29	69,29	83,85	52,19	0,00
Tiger	67,08	74,50	73,50	68,92	64,31	0,00
Fox	57,58	59,50	60,56	53,50	53,13	0,00
Elephant	73,08	78,50	77,69	67,00	71,81	62,88

Clasificador BOW

Tabla D.20: Detalles Ruido 10 BOW

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	68,85	61,64	61,99	71,44	58,28	58,05
Muta 2	81,31	77,38	77,68	83,85	55,56	0,00
Tiger	67,00	68,56	68,88	68,50	62,94	0,00
Fox	57,56	57,81	58,31	53,44	52,69	0,00
Elephant	72,63	74,56	71,44	67,56	63,56	62,19

Clasificador CKNN

Tabla D.21: Detalles Ruido 10 CKNN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	70,62	80,41	83,45	73,23	78,31	76,29
Muta 2	81,07	63,37	75,04	84,07	85,00	0,00
Tiger	67,55	68,50	70,00	69,05	69,19	0,00
Fox	58,65	59,50	65,50	54,30	57,94	0,00
Elephant	73,50	72,50	71,00	69,10	71,94	65,50

Clasificador maxDD

Tabla D.22: Detalles Ruido 10 MAXDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	70,27	71,73	72,27	72,11	64,67	62,93
Muta 2	81,11	69,29	69,29	84,03	59,30	0,00
Tiger	64,88	52,69	51,88	65,92	52,13	0,00
Fox	57,50	54,63	51,38	53,88	52,88	0,00
Elephant	72,79	63,63	64,56	68,04	63,63	58,25

Clasificador EMDD

Tabla D.23: Detalles Ruido 10 EMDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	71,19	84,72	81,09	72,82	75,56	71,35
Muta 2	81,14	69,53	69,29	84,19	49,33	0,00
Tiger	64,36	59,69	60,56	65,57	59,44	0,00
Fox	57,61	59,94	57,19	54,25	55,69	0,00
Elephant	73,00	71,69	70,88	68,96	67,25	63,19

Clasificador MILBoost

Tabla D.24: Detalles Ruido 10 BOOST

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	68,66	50,94	50,94	70,09	47,37	46,17
Muta 2	75,67	35,41	35,41	79,02	6,96	0,00
Tiger	62,56	50,00	50,00	63,63	47,68	0,00
Fox	56,66	50,00	50,00	53,72	44,64	0,00
Elephant	70,13	50,00	50,00	66,59	48,99	46,31

D.4. Ruido 15 %

Clasificador SimpleMIL-max

Tabla D.25: Detalles Ruido 15 MIL MAX

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	65,29	72,98	71,93	71,87	70,48	0,00
Muta 2	81,31	69,29	69,29	82,42	62,06	0,00
Tiger	72,00	70,00	65,00	65,50	63,56	0,00
Fox	57,00	63,50	61,50	56,00	54,56	0,00
Elephant	72,25	77,00	75,50	69,50	65,56	62,88

Clasificador SimpleMIL-min

Tabla D.26: Detalles Ruido 15 MIL MIN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	67,98	78,36	81,52	74,40	76,59	0,00
Muta 2	81,31	69,29	69,29	82,42	58,54	0,00
Tiger	69,25	68,00	74,50	64,88	64,25	0,00
Fox	53,75	53,00	51,50	53,88	51,19	0,00
Elephant	65,50	63,00	58,50	64,25	58,75	56,31

Clasificador SimpleMIL-extreme

Tabla D.27: Detalles Ruido 15 MIL EXTREME

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	70,52	83,74	73,86	75,15	76,13	0,00
Muta 2	81,31	69,29	69,29	82,42	55,56	0,00
Tiger	71,00	70,50	73,00	65,83	64,44	0,00
Fox	54,50	62,00	54,00	55,08	55,38	0,00
Elephant	67,92	77,00	78,50	67,00	69,31	63,31

Clasificador BOW

Tabla D.28: Detalles Ruido 15 BOW

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	68,38	63,52	61,24	69,39	58,00	0,00
Muta 2	81,31	74,13	80,24	82,42	61,14	0,00
Tiger	70,06	68,44	68,19	66,13	62,94	0,00
Fox	54,31	51,63	59,25	54,25	51,69	0,00
Elephant	68,00	72,06	69,00	68,25	63,19	59,63

Clasificador CKNN

Tabla D.29: Detalles Ruido 15 CKNN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	69,37	81,52	76,08	71,04	79,64	0,00
Muta 2	80,62	63,77	64,56	83,09	82,57	0,00
Tiger	70,00	65,50	64,50	67,50	65,50	0,00
Fox	55,35	65,50	56,00	54,50	54,63	0,00
Elephant	68,45	73,00	73,00	69,55	70,25	66,50

Clasificador maxDD

Tabla D.30: Detalles Ruido 15 MAXDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	69,97	63,36	68,89	71,18	67,11	0,00
Muta 2	80,73	69,29	69,29	82,98	56,01	0,00
Tiger	66,58	51,00	53,31	64,46	51,81	0,00
Fox	55,04	51,19	52,31	54,58	53,63	0,00
Elephant	67,33	65,00	64,38	68,83	61,19	58,00

Clasificador EMDD

Tabla D.31: Detalles Ruido 15 EMDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	71,52	77,84	77,16	72,51	73,60	0,00
Muta 2	80,81	70,50	68,45	83,06	61,31	0,00
Tiger	65,68	59,13	55,50	64,43	57,63	0,00
Fox	55,29	58,31	55,25	55,11	56,94	0,00
Elephant	68,18	70,38	73,00	69,54	65,88	60,25

Clasificador MILBoost

Tabla D.32: Detalles Ruido 15 BOOST

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	68,95	50,94	50,94	69,81	44,57	0,00
Muta 2	76,67	35,41	35,41	79,29	10,41	0,00
Tiger	63,72	50,00	50,00	62,63	46,49	0,00
Fox	54,63	50,00	50,00	54,47	44,64	0,00
Elephant	65,91	50,00	50,00	67,09	48,99	45,89

D.5. Ruido 20 %

Clasificador SimpleMIL-max

Tabla D.33: Detalles Ruido 20 MIL MAX

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	75,00	68,30	68,30	65,38	65,77	61,60
Muta 2	82,42	64,29	64,29	77,42	64,49	0,00
Tiger	69,75	67,50	67,50	65,00	59,50	0,00
Fox	57,25	63,00	63,00	56,25	55,31	0,00
Elephant	70,00	74,00	74,00	67,50	65,25	0,00

Clasificador SimpleMIL-min

Tabla D.34: Detalles Ruido 20 MIL MIN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	74,74	74,85	74,85	67,00	65,17	63,36
Muta 2	82,42	69,29	69,29	78,67	50,00	0,00
Tiger	68,75	63,00	63,00	59,88	58,94	0,00
Fox	54,75	51,50	51,50	53,38	51,25	0,00
Elephant	63,38	59,00	59,00	61,75	55,25	0,00

Clasificador SimpleMIL-extreme

Tabla D.35: Detalles Ruido 20 MIL EXTREME

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	75,19	89,12	89,12	67,73	73,40	63,25
Muta 2	82,42	69,29	69,29	78,25	57,42	0,00
Tiger	69,00	73,00	73,00	61,75	63,44	0,00
Fox	56,25	58,50	58,50	54,33	54,13	0,00
Elephant	65,83	74,50	74,50	64,08	63,06	0,00

Clasificador BOW

Tabla D.36: Detalles Ruido 20 BOW

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	70,10	59,23	59,23	64,42	55,68	54,28
Muta 2	82,42	69,29	69,29	78,98	60,90	0,00
Tiger	68,13	65,56	65,56	62,00	59,75	0,00
Fox	55,56	52,25	52,25	54,25	52,19	0,00
Elephant	65,88	68,44	68,44	65,25	60,50	0,00

Clasificador CKNN

Tabla D.37: Detalles Ruido 20 CKNN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	71,74	78,07	78,07	66,57	72,42	68,44
Muta 2	82,42	61,27	61,27	79,43	81,48	0,00
Tiger	67,80	66,00	66,00	63,65	62,94	0,00
Fox	56,20	57,00	57,00	54,90	55,75	0,00
Elephant	66,50	68,00	68,00	66,55	66,88	0,00

Clasificador maxDD

Tabla D.38: Detalles Ruido 20 MAXDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	72,12	71,25	71,25	65,55	63,52	57,84
Muta 2	82,42	69,29	69,29	79,51	59,05	0,00
Tiger	64,63	52,94	52,94	61,63	49,94	0,00
Fox	55,38	51,69	51,69	55,00	51,88	0,00
Elephant	65,71	62,56	62,56	66,54	60,00	0,00

Clasificador EMDD

Tabla D.39: Detalles Ruido 20 EMDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	72,53	79,80	79,80	67,45	68,79	65,01
Muta 2	82,42	68,36	68,36	79,73	58,36	0,00
Tiger	63,61	54,19	54,19	61,61	60,50	0,00
Fox	55,36	57,06	57,06	55,71	55,31	0,00
Elephant	66,04	68,31	68,31	67,14	62,44	0,00

Clasificador MILBoost

Tabla D.40: Detalles Ruido 20 BOOST

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	69,83	50,94	50,94	65,39	46,19	41,42
Muta 2	77,30	35,41	35,41	75,03	10,97	0,00
Tiger	61,91	50,00	50,00	60,16	47,68	0,00
Fox	54,69	50,00	50,00	55,00	44,47	0,00
Elephant	64,03	50,00	50,00	65,00	45,06	0,00

D.6. Ruido 25 %

Clasificador SimpleMIL-max

Tabla D.41: Detalles Ruido 25 MIL MAX

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	72,54	58,65	71,81	65,58	65,46	0,00
Muta 2	78,81	66,43	60,40	77,32	54,13	0,00
Tiger	66,75	66,00	66,50	66,00	61,38	0,00
Fox	53,00	60,00	53,00	58,00	53,81	52,31
Elephant	68,00	70,00	69,00	65,00	64,50	0,00

Clasificador SimpleMIL-min

Tabla D.42: Detalles Ruido 25 MIL MIN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	70,23	69,47	70,64	65,95	68,30	0,00
Muta 2	78,81	69,29	62,62	79,54	55,96	0,00
Tiger	62,75	61,50	60,50	64,25	54,06	0,00
Fox	52,00	50,50	50,50	54,88	51,94	51,38
Elephant	63,00	53,50	53,50	61,63	54,75	0,00

Clasificador SimpleMIL-extreme

Tabla D.43: Detalles Ruido 25 MIL EXTREME

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	72,02	74,85	74,97	67,86	66,47	0,00
Muta 2	78,81	69,29	69,29	78,80	47,42	0,00
Tiger	64,00	66,00	60,50	65,08	57,56	0,00
Fox	52,33	63,50	60,50	55,83	51,19	52,88
Elephant	65,33	71,50	69,00	63,17	63,88	0,00

Clasificador BOW

Tabla D.44: Detalles Ruido 25 BOW

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	69,36	49,85	62,62	64,85	55,83	0,00
Muta 2	78,81	77,22	72,86	79,27	65,15	0,00
Tiger	63,56	66,94	62,38	65,44	58,19	0,00
Fox	51,94	56,75	54,75	56,13	53,38	51,88
Elephant	64,94	65,56	62,75	62,75	61,13	0,00

Clasificador CKNN

Tabla D.45: Detalles Ruido 25 CKNN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	69,85	73,86	72,92	66,41	68,35	0,00
Muta 2	78,56	72,06	66,15	79,54	81,52	0,00
Tiger	63,90	67,00	68,00	66,30	62,50	0,00
Fox	53,15	57,00	55,50	56,50	56,63	51,69
Elephant	66,10	65,00	66,50	64,25	64,38	0,00

Clasificador maxDD

Tabla D.46: Detalles Ruido 25 MAXDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	67,81	63,17	64,85	65,30	61,15	0,00
Muta 2	78,60	60,81	63,66	79,91	53,22	0,00
Tiger	61,88	52,81	52,88	63,50	51,19	0,00
Fox	52,46	49,44	48,50	56,33	52,81	53,63
Elephant	65,38	58,00	60,25	64,29	57,19	0,00

Clasificador EMDD

Tabla D.47: Detalles Ruido 25 EMDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	68,30	73,80	75,08	66,36	70,61	0,00
Muta 2	78,63	70,75	63,80	80,18	57,12	0,00
Tiger	61,07	54,50	56,38	63,00	54,88	0,00
Fox	52,86	58,44	54,94	56,79	53,94	55,00
Elephant	65,43	68,19	65,56	65,46	61,75	0,00

Clasificador MILBoost

Tabla D.48: Detalles Ruido 25 BOOST

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	66,13	50,94	50,94	64,43	47,37	0,00
Muta 2	74,47	35,41	35,41	76,56	9,94	0,00
Tiger	59,69	50,00	50,00	61,38	47,86	0,00
Fox	52,50	50,00	50,00	55,94	44,82	48,09
Elephant	63,50	50,00	50,00	63,53	45,12	0,00

D.7. Ruido 30 %

Clasificador SimpleMIL-max

Tabla D.49: Detalles Ruido 30 MIL MAX

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	58,30	61,87	71,87	59,71	60,15	62,30
Muta 2	65,40	69,29	69,29	80,20	57,78	0,00
Tiger	58,50	69,50	64,00	61,75	56,81	0,00
Fox	54,00	55,00	56,00	52,50	53,06	0,00
Elephant	67,75	69,00	70,50	65,00	54,63	58,38

Clasificador SimpleMIL-min

Tabla D.50: Detalles Ruido 30 MIL MIN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	59,91	66,32	67,37	60,01	64,63	64,50
Muta 2	65,12	61,07	56,90	80,20	58,79	0,00
Tiger	56,88	56,00	66,50	59,50	51,56	0,00
Fox	53,00	50,50	49,50	51,25	50,88	0,00
Elephant	59,88	52,00	51,50	59,13	53,75	53,19

Clasificador SimpleMIL-extreme

Tabla D.51: Detalles Ruido 30 MIL EXTREME

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	61,38	67,37	73,74	61,17	60,78	62,40
Muta 2	65,21	58,17	56,90	80,20	62,07	0,00
Tiger	57,33	63,50	66,00	59,83	57,94	0,00
Fox	52,92	56,50	52,50	51,58	53,75	0,00
Elephant	62,67	67,00	62,00	61,25	58,31	56,56

Clasificador BOW

Tabla D.52: Detalles Ruido 30 BOW

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	59,52	62,54	59,72	59,72	53,93	58,06
Muta 2	67,31	74,92	71,17	80,20	55,27	0,00
Tiger	57,25	61,06	60,88	60,06	58,94	0,00
Fox	52,25	54,38	52,69	52,00	52,44	0,00
Elephant	63,06	60,44	64,00	61,69	60,88	54,25

Clasificador CKNN

Tabla D.53: Detalles Ruido 30 CKNN

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	59,77	67,49	65,32	60,94	69,25	67,60
Muta 2	68,59	56,35	57,18	81,15	84,38	0,00
Tiger	57,55	60,00	61,00	60,90	60,31	0,00
Fox	53,00	51,50	62,50	52,50	54,63	0,00
Elephant	65,00	62,00	68,50	62,35	60,00	58,75

Clasificador maxDD

Tabla D.54: Detalles Ruido 30 MAXDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	60,24	54,43	59,73	60,11	55,40	56,15
Muta 2	67,97	69,29	69,29	80,99	60,95	0,00
Tiger	56,21	53,19	54,44	59,54	50,94	0,00
Fox	52,83	52,94	49,50	52,17	52,75	0,00
Elephant	64,17	60,31	61,81	62,00	55,56	55,19

Clasificador EMDD

Tabla D.55: Detalles Ruido 30 EMDD

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	61,60	69,74	74,60	60,67	67,64	62,10
Muta 2	67,52	60,98	65,22	81,20	50,72	0,00
Tiger	55,89	47,69	50,50	59,79	55,81	0,00
Fox	52,96	54,25	52,56	52,14	53,75	0,00
Elephant	64,32	70,00	65,00	62,86	60,69	57,44

Clasificador MILBoost

Tabla D.56: Detalles Ruido 30 BOOST

Dataset	CONSENSO			MAX VOTOS		
	MIL-EF	MIL-CVCF	MIL-IPF	MIL-EF	MIL-CVCF	MIL-IPF
Musk 1	60,26	50,94	50,94	59,59	47,37	39,84
Muta 2	64,66	35,41	35,41	77,01	17,93	0,00
Tiger	55,16	50,00	50,00	58,56	46,37	0,00
Fox	52,59	50,00	50,00	51,88	43,16	0,00
Elephant	62,53	50,00	50,00	61,25	44,64	47,26