Aplicación de Algoritmos de Machine Learning en la Prevención de Lesiones en el Ámbito Deportivo

Hugo D. Planiscig

Facultad de ingeniería

Maestría en tecnología de información

Director de Tesis

Prof. Dr. Guillermo Rodríguez

Buenos Aires

Agosto 2024

Agradecimientos

En primer lugar, a la Universidad de Palermo por ser parte de mi crecimiento profesional.

A mis compañeros de estudio en la Maestría, por compartir sus conocimientos y ayudarme en cada materia.

Gracias a mi director de Tesis, Dr. Guillermo Rodríguez por su dedicación y motivación por el tema de investigación.

A mi mujer que me apoyó y motivó a lo largo de todo este camino, a mis hijos Felicitas y Manuel que me acompañaron en cada clase, y al resto de mi familia y amigos que siempre me dieron palabras de aliento para poder terminar la maestría y la tesis.

Tabla de contenidos

Resumen....14
Introducción....15
1.1 Planteamiento del Problema....15
1.2 Objetivos....21
1.3 Alcance....21
1.4 Justificación....21
1.4.1 Seguro de Salud....24
1.4.2 Accidentes Personales....24
1.4.3 Responsabilidad Civil....25
1.4.6 Seguro para deportistas....25
Marco teórico....26
2.1 Lesiones en el vóley....26
2.2 Influencia de la psicología....33
2.3 Machine Learning....34
2.4 Tipos de Algoritmos/modelos....35
2.5 La diferencia entre aprendizaje supervisado y no supervisado....36
2.6 Cómo elegir entre aprendizaje supervisado y no supervisado....38
2.7 Aprendizaje por refuerzo....39
2.7.1 ¿Cómo funciona el aprendizaje por refuerzo?....39
2.7.2 Casos de Uso del Aprendizaje por Refuerzo....40
2.8 Evaluación y validación del modelo....41

2.8.1 Cross-validation: Evaluador de performance del estimador. 41

2.8.2 Cross-validation: Métricas de validación..... 45

2.8.3 Matriz de confusión: 46

2.8.3.1 ¿Por qué es útil la Matriz de Confusión? 46

2.8.3.2 Ejemplo del mundo real de matriz de confusión 47

2.8.3.3 Ejemplo de código Python para matriz de confusión 49

2.8.4 Errores Tipo I y Tipo II..... 51

2.8.5 ¿Qué es Precision score?..... 53

2.8.5.1 ¿Cómo calcular la precisión en Python? 53

2.8.5.2. Escenarios del mundo real en los que precisión score se puede utilizar como métricas de evaluación 54

2.8.6 ¿Qué es Recall Score?..... 55

2.8.6.1 Cómo calcular la Recall Score en Python..... 56

2.8.6.2 Escenarios del mundo real en los que las Recall scores se pueden utilizar como métricas de evaluación 56

2.8.7 Precision – Recall Score Tradeoff 57

2.8.8 ¿Qué es Accuracy Score?..... 58

2.8.8.1 ¿Cómo calcular Accuracy score en Python?..... 59

2.8.8.2 Precaución con el Accuracy score 59

2.8.9 ¿Qué es F1-Score? 60

2.8.9.1 Como calcular F1-Score en Python 61

2.8.10 FAQs de precision, recall, accuracy y F1-score. 62

2.8.10.1 ¿Cuál es la diferencia entre Recall y Precision score? 62

2.8.10.2 ¿Cuándo es más importante precisión que Recall? 62

2.8.10.3 ¿Cuáles son los buenos valores de precisión y recuperación? 62

2.8.10.4 Conclusiones 63

2.8.10.5 En resumen: 63

Trabajos relacionados 65

3.1 Trabajos relacionados con enfoque fisiológico 65

3.2 Trabajos relacionados con enfoque psicológico 67

Propuesta 71

4.1 Hipótesis 71

4.2 Enfoque propuesto 71

4.2.1 Proceso 73

4.2.2 Definición de problemas y recopilación de datos. 73

4.2.3 Preprocesamiento de datos 73

4.2.4 Ingeniería de características 74

4.2.5 Selección de modelo 75

4.2.6 Entrenamiento modelo 75

4.2.7 Evaluación del modelo 75

4.2.8 Ajuste del modelo 75

4.2.9 Implementación del modelo..... 76

4.3 Explicación de los algoritmos utilizados 76

4.3.1 Algoritmos de clasificación..... 76

4.3.2 Decision Tree 77

4.3.2.1 Conceptos y ejemplos de Árboles de decisión..... 80

4.3.2.2Tipos de árboles de decisión 81

4.3.2.3 Antes de aprender más sobre los árboles de decisión, familiaricémonos con algunas de las terminologías. 81

4.3.2.4 ¿Por qué utilizar árboles de decisión?..... 82

4.3.2.5 Ejemplo de un árbol de decisión 82

4.3.3 Random Forest 84

4.3.3.1 Cómo funciona el algoritmo de bosque aleatorio: Comprender los árboles de decisión 85

4.3.3.2 Clasificación en bosques aleatorios..... 88

4.3.4 XGBoost..... 91

4.3.4.1 ¿Qué es XGBoost? 91

4.3.4.2 ¿Por qué XGBoost?..... 94

4.3.4.3 Beneficios y atributos de XGBoost..... 95

4.4 Justificación de los algoritmos 96

4.4.1 Naive Bayes..... 96

4.4.2 Decision Trees 96

4.4.3 Random Forests 98

4.4.4 Support Vector Machines (SVMs) 99

4.4.5 k-Nearest Neighbors (k-NN) 101

4.4.6 Regularized Regression (e.g., Ridge Regression, Lasso Regression) 102

4.4.7 Gradient Boosting Machines (e.g., XGBoost, LightGBM) 102

4.4.8 Fortalezas y debilidades de los algoritmos 103

4.4.9 Resumen de análisis de los algoritmos 104

4.4.10 Conclusión 108

Experimento 109

5.1 Set de datos 109

5.2 Métricas 109

5.2.1 Resultados de las métricas aplicadas para el algoritmo Deciscion Tree 109

5.2.2 Resultados de las métricas aplicadas para el algoritmo Random Forest 113

5.2.3 Resultados de las métricas aplicadas para el algoritmo XGBoost 116

5.3 Escenarios 120

5.3.1 Escenario 1: Comportamiento del algoritmo Deciscion Tree 120

5.3.2 Escenario 2 Comportamiento del algoritmo Random Forest 121

5.3.3 Escenario 3 Comportamiento del algoritmo XGBoost 123

5.4 Discusión 124

Conclusión 125
6.1 Ventajas 125
6.2 Limitaciones 126
6.3 Próximos pasos 126
Bibliografía 127

Tabla de figuras

Figura 1	The stress-injury model (Williams & Andersen, 1998) …….	8
Figura 2	Lesiones más comunes en el Vóley ……..	24
Figura 3	Porcentaje de lesiones por zonas corporales. *p<0.05. …….	25
Figura 4	Frecuencia de las características de las lesiones ……	25
Figura 5	Manguito rotador. ……..	26
Figura 6	Rodilla de saltador (jumping Knee ……	28
Figura 7	Inteligencia artificial ……	31
Figura 8	Diferencias entre modelo supervisado y no supervisado ……..	36
Figura 9	Modelo de aprendizaje por refuerzo ……..	37
Figura 10	Diagrama de flujo del flujo de cross validation, evaluador de performance del estimador ……	39
Figura 11	Carga de datos Iris …….	39
Figura 12	Conjunto de entrenamiento …….	40
Figura 13	Modelos k -fold …….	42
Figura 14	Cross-validation: Métricas de validación. cross_val_score …….	42
Figura 15	Puntuación media y desviación standard ……..	43
Figura 16	Ejemplo de matriz de confusión ……	45

Figura 17 Ejemplo de código Python para matriz de confusión …… 47

Figura 18 Ejemplo de Matriz de confusión con código Python …… 47

Figura 19 Ejemplo de Hipótesis …… 49

Figura 20 Validación de hipótesis …….. ……49

Figura 21 Ejemplo de cálculo de precisión_score en Python……. 51

Figura 22 Ejemplo de cálculo de recall_score en Python…….. 53

Figura 23 Ejemplo de calculo del Accurancy en Python…… 56

Figura 24 Ejemplo de cálculo de F1_score en Python…… 58

Figura 25 Formula de Precision y Recall ……..……. 61

Figura 26 Formula de F-Measure …….. 61

Figura 27 Arquitectura propuesta ……. 69

Figura 28 Diagrama de flujo de estimadores …… ……..74

Figura 29 Regresion decisión tree …… 75

Figura 30 Estructura de un árbol de decisión……. 79

Figura 31 Ejemplo de árbol de decisión ……80

Figura 32 Estructura de Random Forest …… 83

Figura 33 Ejemplo de Random Forest ……85

Figura 34 Clasificación de Random Forest ……86

Figura 35 Ejemplo de clasificación de Random Forest …….87

Figura 36 XGBoost Modelo ejemplo……. 89

Figura 37 XGBoost Arbol ……..89

Figura 38 Modelo de combinación de árboles …… 90

Figura 39 Tabla comparativa de algoritmos…….. 101

Figura 40 Resultado de Accurency y desvío standard para el algoritmo Deciscion Tree …. 107

Figura 41 Código para la utilización las métricas de Precision, Recall y F1 Score para validar desempeño del modelo ……. 107

Figura 42 Resultados de las métricas de validación del modelo de Precision, Recall y F1 Score para validar desempeño del modelo. …….. 107

Figura 43 Resultado de la matriz de confusión aplicada al modelo ……..108

Figura 44 Matriz de observación …….. 108

Figura 45 Resultado de las métricas de métricas de validación por cada clase ……109

Figura 46 Datos de ingresados para la simulación de la predicción ……. 109

Figura 47 Resultado de la predicción ……109

Figura 48 Resultado de accurency y desvío standard para el algoritmo Random Forest …110

Figura 49 Script utilizado para la validación del modelo con Random Forest ……110

Figura 50 Resultados de la validacion del desempeño del modelo con Random Forest ….111

Figura 51 Matriz de confusión con Random Forest ……111

Figura 52 Matriz de observación para el algoritmo Random Forest…….. 112

Figura 53 Validación por cada clase para el algoritmo de Random Forest ……112

Figura 54 Datos ingresados de prueba para el test del algoritmo Random Forest…. 113

Figura 55 Resultados del test para el algoritmo Random Forest……. 113

Figura 56 Resultado de Accurency y desvío standard para el algoritmo XGBoost…..113

Figura 57 Validación del modelo de clasificación con la matriz de confusión y utilización de las métricas de Precision, Recall y F1 Score para validar desempeño del modelo……..114

Figura 58 Resultados obtenidos de las métricas de validación del modelo XGBoost..114

Figura 59 Resultados de la matriz de confusión para el algoritmo XGBoost… 115

Figura 60 Matriz de observación para XGBoost …….. 116

Figura 61 Métricas de validación por cada clase para el algoritmo XGBoost ……116

Figura 62 Datos ingresados como test para el algoritmo XGBoost ……..116

Figura 63 Resultado de la predicción con el algoritmo XGBoost ……..117

Figura 64 Importe de librerías para el usa del algoritmo decisión tree ……117

Figura 65 Set de datos utilizado …….118

Figura 66 Preparación de los datos para la utilización del algoritmo Deciscion Tree...118

Figura 67 Librerías utilizadas para el algoritmo de Random Forest…….. 119

Figura 68 Set de datos para la utilización del algoritmo Random Forest……. 119

Figura 69 Cross Validation como el evaluador de desempeño del estimador…… 120

Figura 70 Librerías utilizadas para el algoritmo XGBoost ……120

Figura 71 Set de datos levantados para el algoritmo XGBoost …….121

Figura 72 Cross Validation como el evaluador de desempeño del estimador ……..121

Resumen

Hasta el momento, distintas investigaciones han sido enfocadas en el estudio de cómo prevenir las lesiones en el deporte, ayudando a los distintos actores a prevenir las mismas utilizando las herramientas de Machine Learning (ML) e Inteligencia Artificial (AI). El uso de estas herramientas tiene un potencial enorme para la obtención de información preventiva de las lesiones en el campo psicológico y fisiológico por separado. Pero no se han encontrado investigaciones que traten de utilizar estas herramientas con el objetivo de relacionar o vincular variables, las psicológicas con las fisiológicas en conjunto, y esto sería de gran ayuda para la prevención de lesiones teniendo en cuenta que son variables muy influyentes en los deportistas.

El objetivo de este trabajo es desarrollar un modelo de prevención de lesiones utilizando ML y AI aplicado a deportes y vinculando variables psicológicas y fisiológicas al mismo tiempo, identificando las mismas y aplicando los distintos tipos de algoritmos existentes.

Como conclusión, los resultados arrojados muestran que nuestro enfoque es viable para predecir lesiones en el deporte combinando 2 universos que hasta el momento no estaban vinculados o relacionados, estos son los de origen fisiológicos y los psicológicos. A lo largo del desarrollo se puede observar como a partir de un set de datos y la utilización de distintos algoritmos de ML, estos universos tienen muy buena correlación arrojando un resultado final muy preciso respecto al tipo de lesión puede tener un deportista, estas son clasificadas como del tipo grave, moderada o leve.

Introducción

Hasta el momento, distintas investigaciones han sido enfocadas en el estudio de cómo prevenir las lesiones en el deporte, ayudando a los distintos actores a prevenir las mismas utilizando las herramientas de Machine Learning (ML) e Inteligencia Artificial (AI). El uso de estas herramientas tiene un potencial enorme para la obtención de información preventiva de las lesiones en el campo psicológico y fisiológico por separado. Pero no se han encontrado investigaciones que traten de utilizar estas herramientas con el objetivo de relacionar o vincular variables, las psicológicas con las fisiológicas en conjunto, y esto sería de gran ayuda para la prevención de lesiones teniendo en cuenta que son variables muy influyentes en los deportistas.

El objetivo de este trabajo es desarrollar un modelo de prevención de lesiones utilizando ML y AI aplicado a deportes y vinculando variables psicológicas y fisiológicas al mismo tiempo, identificando las mismas y aplicando los distintos tipos de algoritmos existentes.

1.1 Planteamiento del Problema

Las lesiones deportivas constituyen un aspecto de marcada relevancia para la comunidad científica internacional debido a sus altas tasas de incidencia y prevalencia en todos los deportes a nivel global, constituyéndose como un factor de riesgo inherente a la práctica deportiva que en su materialización produce afectaciones sobre el estado de salud y el rendimiento de los deportistas con los consecuentes gastos económicos que implica el tratamiento y la rehabilitación (García et al., 2015; Padegimas et al., 2016;Pujals et al., 2016).

La naturaleza multicausal de este flagelo ha propiciado que con mayor frecuencia sea abordado a partir de un enfoque multidisciplinar donde la Psicología juega un importante rol en la

prevención, el tratamiento, la rehabilitación y la vuelta al terreno del deportista lesionado. La extensa producción científica acumulada desde los primeros estudios en la década del 70 del pasado siglo XX, ha permitido arribar a consensos sobre las relaciones que establecen determinadas **variables psicológicas** con las lesiones (Herring et al., 2017; Schinke et al., 2017; Soligard, et al., 2016).

A partir del modelo teórico propuesto por Andersen y Williams (1988) y su actualización diez años después (Williams y Andersen, 1998), que explica cómo la relación entre el **estrés psicosocial** y los recursos de afrontamiento del deportista configuran la vulnerabilidad a la lesión, múltiples estudios han contribuido a indicar evidencia de la mediación de variables psicológicas en la ocurrencia de las lesiones (Glendon, Clarke y McKenna, 2016; Gómez-Espejo et al., 2017; Johnson e Ivarsson,2017; Olmedilla, Rubio y Ortega, 2015; Singh y Conroy, 2017).

No obstante, de acuerdo con el Modelo Global Psicológico de las Lesiones Deportivas de Olmedilla y García-Mas (2009) el estudio de la relación entre factores psicológicos y lesiones deportivas ha mostrado una gran diversidad de resultados atendiendo a la heterogeneidad de las investigaciones en cuanto a la selección de las muestras, los instrumentos de evaluación utilizados, los deportes objeto de estudio y los niveles de profesionalidad de los deportistas, así como los análisis estadísticos. Estos aspectos han obstaculizado la sistematización teórica conceptual de esta relación tal como afirman Ortín et al. (2010) y Palmi y Solé (2014). En consecuencia, la predicción de lesiones teniendo en cuenta variables psicológicas cuenta con un reducido número de estudios en los últimos años (Berengüí y Puga, 2015; García-Mas et al., 2014; Olmedilla et al., 2018; Rubio et al., 2014), los cuales se han desarrollado fundamentalmente en deportes como el fútbol, baloncesto, balonmano, atletismo, ciclismo, lucha

olímpica y taekwondo hallando resultados igualmente divergentes a los estudios descriptivos y correlacionales.

Si bien la evidencia científica ha mostrado la relevancia de los factores psicológicos como antecedentes y consecuentes de las lesiones, aún no se tienen en cuenta de manera suficiente para su prevención, por lo que la predicción psicológica de las mismas constituye un área del conocimiento de relevancia y actualidad debido a los escasos estudios que han determinado predictores psicológicos de lesiones en un grupo reducido de deportes.

Al respecto, Juan (2021) expresó que la determinación de los predictores de lesiones mediante **modelos matemáticos** constituye una herramienta de alto valor para entrenadores y deportistas para su prevención, no obstante, la mayoría de las modelaciones realizadas no tienen en cuenta todos los factores de riesgo como es el caso de los aspectos psicológicos. Además, la mayoría de las modelaciones se han realizado mediante **modelos de regresión lineal** por lo que resulta necesario emplear métodos informáticos que faciliten la labor de los especialistas del deporte en este sentido. (Jesús Ríos Garit, Yanet Pérez Surita, María del Carmen Chávez Cárdenas, (2021), [Jesús Ríos Garit, Yanet Pérez Surita, María del Carmen Chávez Cárdenas,2021, Predictores psicológicos de lesiones en lanzadores de béisbol de alto Rendimiento, Universidad Central “Marta Abreu” de Las Villas].

Los atletas que participan en el deporte están expuestos a un alto riesgo de lesiones. Investigaciones anteriores han encontrado un gran número de factores de riesgo (tanto fisiológicos como psicológicos) que podrían aumentar el riesgo de lesión. Una limitación en estudios previos es que pocos han considerado la compleja interacción entre factores psicológicos en su diseño de la investigación.

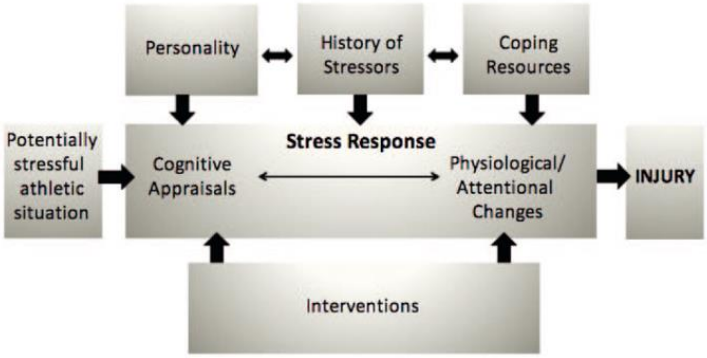
Además de tener para hacer frente a las tensiones físicas de la lesión (p. ej., dolor, incomodidad, los rigores de la rehabilitación), los atletas deben lidiar con el **estrés psicosocial** de una lesión, como amenazas a la autoestima, amenazas a la participación en la carrera atlética, y el aislamiento de los compañeros. Dada la profunda carga física y psicosocial de las lesiones, predicción los esfuerzos destinados a minimizar el riesgo de lesiones son importantes.

En un intento por identificar los principales factores de riesgo de lesión en el área de la psicología del deporte, varios conceptos se han desarrollado modelos. Uno de los más influyentes y bien probado de estos modelos es Williams y Modelo de lesión por estrés de Andersen (**Figura 1**). Williams y Andersen postular que una situación potencialmente estresante generará una respuesta al estrés que varía en intensidad a lo largo de un continuo. Se sugiere que la fuerza de la respuesta al estrés sea influenciada por cuán amenazante percibe el atleta a situación (es decir, la medida en que las demandas de la situación superan la capacidad percibida para cumplir tales demandas).

Tres amplias categorías de variables, **personalidad rasgos** (p. ej., rasgo de ansiedad, perfeccionismo), **historia de factores estresantes** (p. ej., estrés por eventos importantes de la vida, problemas diarios, antecedentes de lesiones anteriores) **y estrategias/recursos de afrontamiento** (p. ej., uso de habilidades psicológicas, apoyo social) también se sugieren para influir en la fuerza de la respuesta al estrés y la subsiguiente probabilidad de lesión.

Figura 1

The stress-injury model (Williams & Andersen, 1998)



El mecanismo porque se cree que la respuesta al estrés aumenta las lesiones ocurre a través de cambios tensionales y somáticos como el aumento de la distracción y el estrechamiento periférico, así como tensión muscular, fatiga y reducción tiempo/coordinación.

Por ejemplo, un jugador profesional de fútbol que está preocupado por las obligaciones contractuales (es decir, un principal estresante de la vida) y que está experimentando una relación difícil (es decir, un factor de estrés crónico) pueden, cuando se colocan en una situación potencialmente estresante (por ejemplo, una final de liga), experimenta aumento de los déficits atencionales y físicos que impiden que él o ella vea o reaccione a un lado. Bajo tales circunstancias, este jugador está en mayor riesgo de lesionarse. (Andreas Ivarsson, Urban Johnson, and Leslie Podlog, (2013)) [Casper DEVANTIER, Sport Science Review, vol. XX, No. 5-6, December 2011]

Las lesiones en los deportes profesionales ocurren regularmente. Algunos pueden ser menores, mientras que otros pueden causar un gran impacto en la carrera de un jugador y su potencial de ingresos. En el fútbol, existe un alto riesgo de que los jugadores se lesionen durante el tiempo de

juego. Existen herramientas (ML) de predicción de lesiones que utilizan la cantidad de minutos jugados en el campo de un jugador de fútbol, la cantidad de apariciones, la distancia recorrida y los datos de rendimiento de la temporada actual y anterior como variables para realizar análisis estadísticos y proporcionar resultados predictivos de lesiones utilizando un modelo de regresión lineal de aprendizaje automático. [Amiel Satvedi,2020, Injury Predictor For Soccer Players Using Machine Learning, Sheridan College (Oakville)]

Como se viene mencionando en los antecedentes, el uso de estas herramientas tiene un potencial enorme para la obtención de información preventiva de las lesiones en el campo psicológico y fisiológico por separado. Hasta el momento no se han encontrado investigaciones que traten de utilizar estas herramientas con el objetivo de relacionar variables psicológicas con las fisiológicas, y esto sería de gran ayuda para la prevención de lesiones teniendo en cuenta que son variables muy influyentes en los deportistas.

1.2 Objetivos

Desarrollar un modelo de prevención lesiones utilizando ML, aplicado a deportes(vóley) y vinculando variables psicológicas y fisiológicas.

- Identificar las variables psicológicas y fisiológicas que inciden en las lesiones de deportistas(vóley).
- Identificar las relaciones entre ambos tipos de variables y acción conjunta.
- Seleccionar algoritmos de ML que puedan correlacionar la ocurrencia de lesiones con las variables identificadas.
- Implementar los Algoritmos aplicados de ML.

1.3 Alcance

El alcance de la investigación es puntualmente en Vóley como deporte y tomando algoritmos de ML que se describen a continuación:

- Decision tree
- Random forest
- XGBoost

El resultado es etiquetar la respuesta en lesiones del tipo: Grave (3), Moderada (2) o Leve (1).

1.4 Justificación

Los atletas que participan en el deporte están expuestos a un alto riesgo de lesiones. Investigaciones anteriores han encontrado un gran número de factores de riesgo (tanto fisiológicos como psicológicos) que podrían aumentar el riesgo de lesión.

Desde la Psicología, cada vez está mejor establecida la relación entre factores psicológicos y las lesiones. Lesionarse tiene consecuencias con clara repercusión en la esfera psíquica y emocional:

- Al sufrir una lesión se producen cambios y reajustes en la actividad diaria, cambiando la asistencia a entrenos y competiciones por intervenciones, reposo o rehabilitación. **Esta interrupción genera un mayor aislamiento ya que se ve interrumpida la participación en la actividad social y de grupo**.
- A causa de la lesión se pone en riesgo la posición y permanencia en el equipo, incluso tener acceso a nuevos contratos. **El riesgo de pérdida puede causar una sensación de amenaza**.
- Los plazos de recuperación no son fijos y generan una gran carga de incertidumbre y sensación de pérdida de control.
- Una lesión de gravedad incluso puede dificultar realizar actividades básicas de la vida diaria como vestirse solo, desplazarse.

Todos estos factores influyen en el bienestar psicológico de la persona. Síntomas como mayor irritabilidad, hostilidad, pensamientos negativos, síntomas depresivos o ansiedad son frecuentes en deportistas lesionados.

Por otro lado, las lesiones tienen un gran impacto en los deportistas, no solo en lo que refiere a lo fisiológico y psicológico, sino también en lo económico/financiero:

- Rendimiento de los deportistas con los consecuentes gastos económicos que implica el tratamiento y la rehabilitación.
- Gran impacto en la carrera de un deportista y su potencial de ingresos.

En este sentido se pierden millones en la recuperación de los deportistas, por ejemplo, en la Premier League inglesa (EPL), en las temporadas 2012-2013 a 2016-2017 se llevó a cabo una estimación completa de los costos de las lesiones basada en datos de cinco temporadas (Eyal Eliakim, Elia Morgulev , Ronnie Lidor, Yoav Meckel, (2020) y el hallazgo fue el siguiente:

- Los equipos EPL perdieron en promedio £ 45 millones por temporada debido a disminuciones en el rendimiento relacionadas con lesiones. Esta estimación se basa en la evaluación del bajo rendimiento de los equipos debido a jugadores lesionados (£36 millones) y en un cálculo directo de los salarios pagados a los jugadores lesionados (£9 millones).

Tabla 1

Impacto de las lesiones en los logros del equipo en la temporada 2016-2017 de la EPL

Club	No of injuries	Days lost	Team worth (in millions of Pounds Sterling)	Expected rank	Actual rank	Rank difference	Expected points	Actual points	League points lost
Man United	75	1262	481	1	6	-5	93	69	−24
Chelsea	46	877	466	2	1	1	86	93	7
Man City	51	1153	466	3	3	0	78	78	−8
Arsenal	71	1573	422	4	5	-1	76	75	−1
Liverpool	81	1840	348	5	4	1	75	76	1
Tottenham	55	1029	329	6	2	4	69	86	17
West Ham	82	2191	217	7	11	-4	61	45	−16
Everton	41	1618	215	8	7	1	46	61	15
Leicester	44	885	189	9	12	-3	46	44	−2
Southampton	52	1655	179	10	8	2	45	46	1
Stoke	60	1624	155	11	13	-2	45	44	−1
Crystal Palace	67	1857	142	12	14	-2	44	41	−3
Watford	77	1852	117	13	17	-4	44	40	−4
Bournemouth	52	1163	110	14	9	5	41	46	5
West Brom	36	203	97	15	10	5	41	45	4
Swansea	44	991	95	16	15	1	40	41	1
Middlesborough	48	1055	95	17	19	-2	40	28	−12
Sunderland	85	2265	83	18	20	-2	34	24	−10
Hull City	49	2289	70	19	18	1	28	34	6
Burnley	35	822	60	20	16	4	26	40	16

Eyal Eliakim, Elia Morgulev , Ronnie Lidor, Yoav Meckel, (2020), Estimación de los costos de las lesiones: daño financiero de los equipos de la Premier League inglesa por bajo rendimiento debido a las lesiones.

El deportista tiene probabilidades de sufrir un accidente o alguna clase de siniestro. Todas las actividades deportivas tienen diferentes niveles de riesgo: no es lo mismo jugar al jockey sobre hielo que al fútbol, ser un deportista profesional o amateur, o tener 20 años que 40. En cualquier caso, adquirir un **seguro para deportistas** es una buena opción para estar cubierto ante gastos médicos, muerte accidental, robo de equipos o responsabilidad civil, entre otros incidentes. Cada póliza tiene sus beneficios y exclusiones y habrá que ver, según el deporte y los riesgos que se necesitan cubrir, cuál conviene.

1.4.1 Seguro de Salud

Un Seguro de Salud puede ser muy útil para cubrir los costos por lesiones deportivas. Esguinces, sobrecargas de ligamentos, torceduras y otros traumatismos leves pueden requerir medicamentos y recuperaciones más o menos largas. Lesiones más graves, como fracturas o contusiones cerebrales, tal vez necesitan cirugías y rehabilitación para volver a practicar. Por lo tanto, conviene tener un seguro de gastos médicos diseñado especialmente para deportistas, que brinde cobertura de acuerdo al riesgo de ese deporte. Este tipo de seguro para deportistas contemplará todas las maneras en que puede suceder un accidente deportivo y sus consecuencias.

1.4.2 Accidentes Personales

El Seguro de Accidentes Personales tradicional tiene algunas exclusiones para las muertes que ocurren a causa de siniestros durante la realización de deportes de riesgo o prácticas deportivas profesionales. Sin embargo, existen pólizas pensadas especialmente para deportistas, que cubren las consecuencias de accidentes en entrenamientos o competencias. Estos seguros sí contemplan el fallecimiento por accidentes en el ámbito deportivo, invalidez total o parcial y reintegro de gastos médicos para amateurs y profesionales, de forma individual o grupal.

1.4.3 Responsabilidad Civil

Otro seguro para deportistas que puede servir para varios deportes es el de Responsabilidad Civil. Una póliza así cubriría los gastos médicos por daños a terceros (en una práctica de boxeo, por ejemplo). Este seguro se haría cargo de las lesiones físicas a otras personas y también podría incluir el daño de bienes materiales (por ejemplo, si chocás con otro ciclista y rompés su bicicleta). De ser necesario, el Seguro de Responsabilidad Civil además brinda apoyo legal.

1.4.5 Seguro de Vida

El Seguro de Vida está pensado para que, en caso de invalidez total o fallecimiento por accidente o enfermedad, la familia del asegurado reciba una compensación económica. Como en los demás seguros, hay que ampliar la póliza o buscar una especialmente diseñada para deportistas, ya que algunas prácticas deportivas peligrosas (paracaidismo, aladeltismo, buceo) y competencias de velocidad no están incluidas.

1.4.6 Seguro para deportistas

Hay muchos, se tiene que encontrar el que mejor se adapte a la actividad. Como conclusión, tanto por el impacto económico-financiero, como por las consecuencias profesionales, psicológicas y fisiológicas, así como motivacionales, el estudio de las lesiones en deportistas y su multicausalidad adquiere una trascendencia fundamental para el propio deportista como también para su entorno.

Marco teórico

En este capítulo se verán los distintos tipos de lesiones comunes en el vóley, la influencia de la psicología, una introducción a ML y a los algoritmos utilizados.

2.1 Lesiones en el vóley

¿Qué es una lesión deportiva?

Las lesiones deportivas son lesiones que ocurren durante la práctica de una actividad deportiva de tipo profesional o amateur. Algunas de estas lesiones ocurren casualmente, otras se deben a prácticas o posiciones incorrectas durante un entrenamiento o en una competición, o al uso incorrecto del material.

Las lesiones deportivas suelen consistir en traumatismos musculares y traumatismos de las articulaciones y los ligamentos:

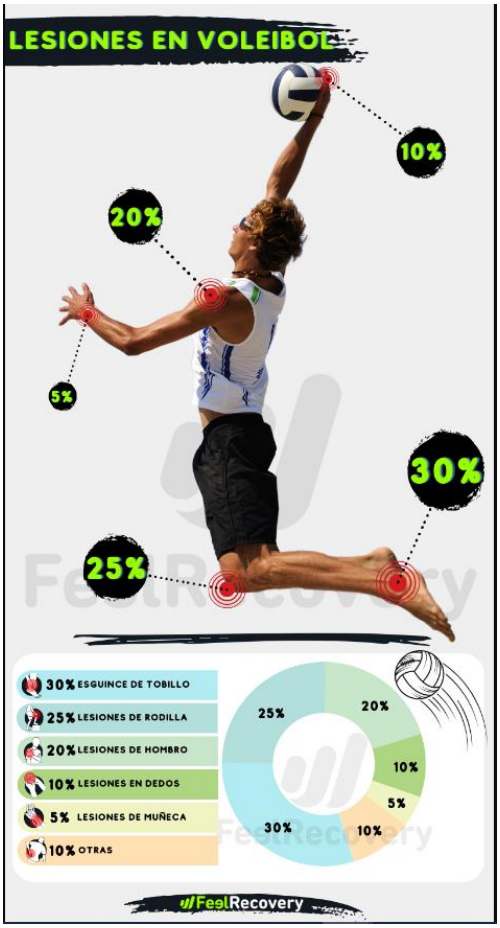
- Contusiones
- Luxaciones
- Lesiones musculares (desgarros, tirones, calambres)
- Esguince de tobillo o de la rodilla
- Fracturas óseas
- Lesiones de los tendones
- Lesiones de la tibia
- Contracturas

¿Qué tipos de lesiones son más comunes en el vóley?

Debido a que el voleibol involucra movimientos repetitivos de arriba, como el remate y el bloqueo, los jugadores son propensos a tener lesiones por uso excesivo del hombro. Además, los jugadores de voleibol son particularmente susceptibles a las lesiones en los dedos tal como se muestra en la Figura 2.

Figura 2

Lesiones más comunes en el Vóley.



En las figuras 3 y 4 se muestran estadisticas que acompañan como y donde se realizan las lesiones:

Figura 3

Porcentaje de lesiones por zonas corporales. *p<0.05.

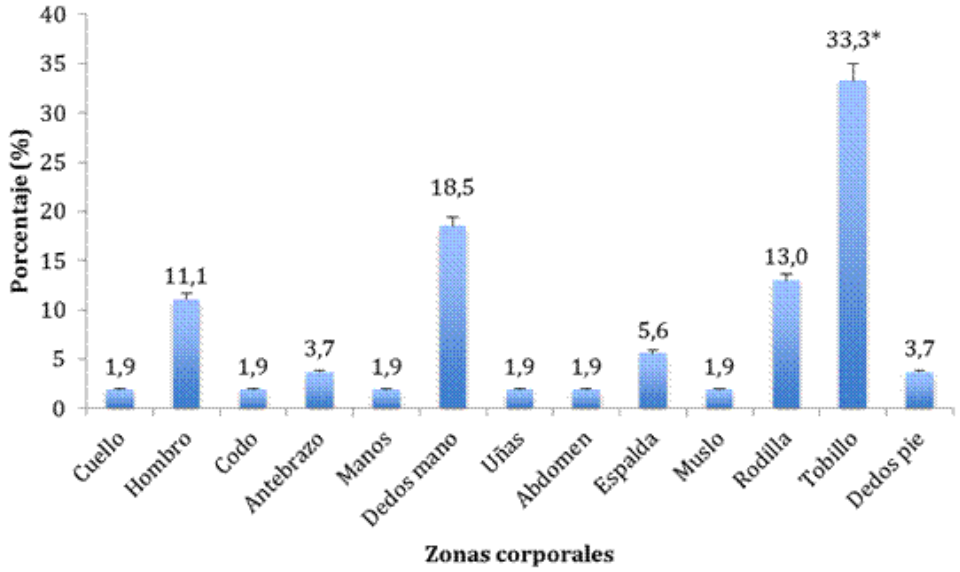


Figura 4

Frecuencia de las características de las lesiones

Tabla 1. Frecuencia de las características de las lesiones

		Frecuencia (*n*)	Incidencia %	X^2
Momento de la lesión	Competición	17	51,5	0,862
	Entrenamiento	16	48,5	
Modo en que la hizo	Impacto	17	51,5	0,862
	Uso Excesivo	16	48,5	
Tipo de lesión	Nueva lesión	18	58,1	0,369
	Reincidente	13	41,9	

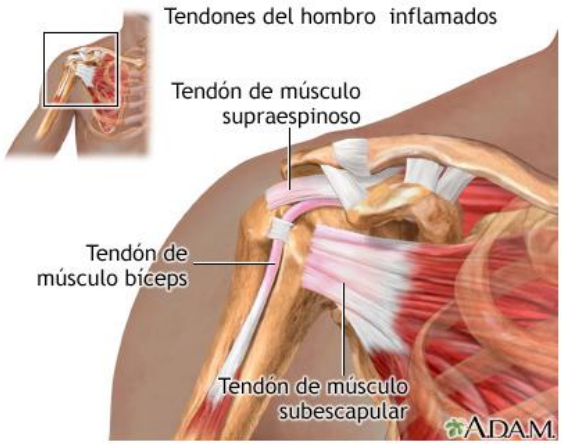
¿Cómo se tratan las lesiones de vóleibol?

La tendinitis del manguito rotador

Los músculos del manguito de los rotadores son importantes en la generación de la energía necesaria para mover el hombro en los diferentes gestos con los brazos elevados en este deporte. Aunque rara vez se presenten desgarros completos en jugadores jóvenes, estos músculos pueden irritarse o fatigarse con el uso excesivo. A menudo, el descanso y la terapia física puede ser suficiente para resolver el dolor. Si el dolor persiste, consulte con su médico traumatólogo. En la figura 5 se muestra un manguito rotador.

Figura 5

Manguito rotador.



Traumatismos de los Dedos

Los dedos son vulnerables a las lesiones durante las actividades de voleibol, como el bloqueo, toque de dedos o en el remate. Las lesiones en los dedos más comunes son fracturas, dislocaciones, tendinitis y desgarros de los ligamentos de los dedos. Si no puede doblar el dedo,

consulte con su entrenador o traumatólogo. El tratamiento puede variar significativamente dependiendo del grado de su lesión.

Los esguinces de tobillo

Las lesiones de tobillo son las lesiones más comunes en los jugadores de voleibol y responsable de la mayoría de bajas. Los esguinces de tobillo deben ser inmovilizados por el más corto tiempo posible para permitir la rehabilitación más rápida. Por lo general, las lesiones pueden ser tratadas sin cirugía, con refuerzos y rehabilitación. Ocasionalmente, sin embargo, los esguinces de tobillo pueden estar asociados con fracturas sutiles o lesiones del cartílago. Si el dolor continuo después de varias semanas debería requerir una mayor evaluación, incluyendo rayos X y / o imágenes de resonancia magnética.

Volver a jugar por lo general se permite una vez que los jugadores no tienen dolor y son capaces de soportar su peso corporal sobre los dedos del pie. La cirugía se reserva para las personas con esguinces de tobillo recurrentes que no han respondido a las medidas conservadoras o con fracturas asociadas.

La tendinitis rotuliana

La tendinitis rotuliana es la inflamación del tendón que conecta la rótula con la tibia. Es común en cualquier deportista sometido a actividades de salto contundentes y repetitivas, como en el remate y el bloqueo.

Los brace para el tendón rotuliano son útiles para la descarga de la tensión en el tendón rotuliano y son a menudo la primera línea de tratamiento. Sesiones de terapia física más fortalecimiento con ejercicios excéntricos de cuádriceps y estiramiento también son útiles. Educación específica

del aterrizaje o del salto. En algunos casos, la tendinitis patelar persiste a pesar de la rehabilitación y se requiere de cirugía. En la figura 6 se muestra parte de la rodilla donde se lesiona un jugador.

Figura 6

Rodilla de saltador (jumping Knee)



Ligamento Cruzado Anterior (LCA)

Al igual que los esguinces de tobillo, la mayoría de las lesiones del LCA en jugadores de voleibol ocurren cuando un jugador cae torpemente después de saltar. Por lo general, las rupturas del Ligamento Cruzado Anterior están asociados con un “chasquido” y la hinchazón inmediata de la rodilla. El examen por un médico y la Resonancia Magnética se utilizan a menudo para confirmar la lesión del LCA. El tiempo de recuperación luego de la cirugía es por lo general entre 8 y 12 meses. Hay técnicas de entrenamiento que pueden disminuir el riesgo de lesión del

LCA en atletas que saltan, especialmente las mujeres. Es importante hablar con un entrenador atlético calificado o con un médico traumatólogo especializado en medicina del deporte antes de tomar estas actividades.

Dolor en la espalda baja

La zona lumbar es una fuente común de dolor crónico entre los jugadores de voleibol. La causa de la mayoría de los dolores de espalda baja está relacionada con el músculo o la tensión de los ligamentos. El dolor generalmente se resuelve con el descanso, terapia física y rehabilitación.

Si el dolor lumbar se acompaña de dolor que se irradia hacia las piernas y entumecimiento o debilidad en el pie o el tobillo, la causa puede ser un disco herniado. En los casos de dolor que se irradia, una resonancia magnética puede ser útil en la evaluación de la presencia de una hernia de disco. En la mayoría de los casos, los jugadores de voleibol pueden volver a jugar una vez que el dolor, entumecimiento y debilidad resuelve.

Los jugadores de voleibol también pueden estar en mayor riesgo de una especie de fractura por estrés en la espalda baja llama espondilólisis. Si el dolor persiste más de un mes y empeora con la flexión hacia atrás, considere consultar a un médico.

¿Cómo se pueden prevenir las lesiones en el vóley?

Muchas de las lesiones del voleibol se pueden prevenir siguiendo las directrices de formación adecuadas y algunos consejos:

Utilizar técnicas de entrenamiento de fuerza adecuados para la espalda baja, hombros y piernas.

Utilice un soporte para el tobillo externo, como una tobillera o cinta adhesiva, para evitar que el tobillo se mueva más de lo permitido, especialmente si usted ha tenido un esguince previo.

Minimizar la cantidad de entrenamiento de saltos sobre superficies duras.

Calentar los músculos con estiramientos y ejercicios aeróbicos ligeros.

Asegúrese de enfriar correctamente después de la práctica.

Si tiene dolor, visite a su médico traumatólogo y siga las instrucciones para el tratamiento.

El atleta debe volver a jugar sólo cuando el alta se concede por un profesional de la salud.

2.2 Influencia de la psicología

En lo que respecta a la parte psicológica para esta tesis se proporcionaron datos enfocados a cuestionarios Wellnes que tratan de reflejar el estado psicológico de los jugadores en un periodo de tiempo, tomando como referencia el cuestionario PSE.

El Examen del Estado Actual (PSE) es un método de entrevista estructurada que se utiliza en evaluaciones psiquiátricas para recopilar sistemáticamente información sobre el estado mental actual de un individuo. Desarrollado por John Wing y sus colegas, es un componente clave del Programa de Evaluación Clínica en Neuropsiquiatría (SCAN), una herramienta de diagnóstico integral para trastornos psiquiátricos.

Durante un PSE, un médico capacitado formula una serie de preguntas estandarizadas que cubren diversos aspectos de la salud mental del individuo, incluido el estado de ánimo, los procesos de pensamiento, la percepción, la cognición, el comportamiento y el funcionamiento social. El objetivo es obtener una evaluación detallada y objetiva del estado psicológico actual de la persona, que luego puede utilizarse para formular un diagnóstico y desarrollar un plan de tratamiento adecuado.

El PSE sigue un formato estructurado para garantizar la coherencia entre las evaluaciones y minimizar el sesgo. Por lo general, implica hacer preguntas específicas sobre síntomas y comportamientos, así como buscar detalles adicionales cuando sea necesario. La información recopilada durante el PSE ayuda a los médicos a identificar trastornos psiquiátricos, determinar su gravedad y monitorear los cambios a lo largo del tiempo.

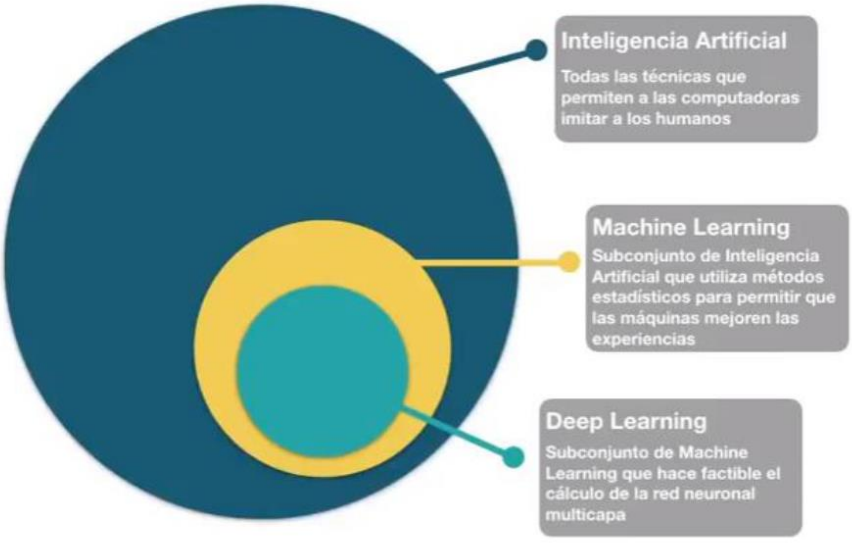
En general, el PSE es una herramienta valiosa en la práctica psiquiátrica, ya que proporciona un marco sistemático para evaluar la salud mental y guiar la toma de decisiones clínicas.

2.3 Machine Learning

En líneas generales el ML es una rama de la IA tal como se muestra en la figura 7.

Figura 7

Inteligencia artificial



Inteligencia Artificial es el concepto más amplio, en donde las máquinas son capaces de llevar a cabo las tareas de una manera considerada “inteligentes”

Machine Learning es una aplicación actual de la Inteligencia Artificial basada en la idea de que deberíamos ser capaces realmente de dar a las máquinas acceso a los datos y dejarles aprender por sí mismos, es decir, algoritmos que hacen a las máquinas más inteligentes. El activador de la inteligencia artificial es el aprendizaje automático

Un sistema con Machine Learning podría “aprenderse” los datos y clasificar con mayor precisión los vehículos, pero uno con Deep Learning puede “entrenarse” sobre los nuevos datos que va recibiendo. Es decir, puede emplear un diferenciador erróneo y equivocarse una vez, pero a la siguiente emplea otro para acercarse cada vez más al resultado correcto.

2.4 Tipos de Algoritmos/modelos

Según su entrenamiento se pueden clasificar en:

- Supervisados
- No supervisados
- Por refuerzo

Según la salida/objetivo:

- Clasificación
- Regresión
- Clustering
- Reducción de dimensionalidad

2.5 La diferencia entre aprendizaje supervisado y no supervisado

La mayor diferencia entre el aprendizaje automático supervisado y no supervisado es el tipo de datos utilizados. El aprendizaje supervisado utiliza datos de entrenamiento etiquetados y el aprendizaje no supervisado no.

Más simplemente, los modelos de aprendizaje supervisado tienen una comprensión básica de cuáles *deberían* ser los valores de salida correctos.

Con el aprendizaje supervisado, un algoritmo utiliza un conjunto de datos de muestra para entrenarse a sí mismo para hacer predicciones, ajustándose iterativamente para minimizar el error. Estos conjuntos de datos están etiquetados según el contexto y proporcionan los valores de salida deseados para permitir que un modelo dé una respuesta "correcta".

Por el contrario, los algoritmos de aprendizaje no supervisados funcionan de forma independiente para aprender la estructura inherente de los datos sin ninguna guía o instrucción específica. Simplemente proporciona datos de entrada sin etiquetar y deja que el algoritmo identifique cualquier patrón que ocurra naturalmente en el conjunto de datos.

Si bien el tipo de datos es la forma más fácil de diferenciar entre estos dos enfoques, cada uno tiene objetivos y aplicaciones diferentes que también los diferencian entre sí.

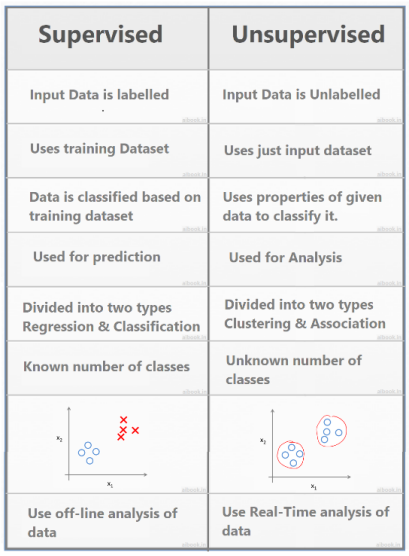
Los modelos de aprendizaje supervisado se centran más en aprender las relaciones entre los datos de entrada y salida. Por ejemplo, se podría utilizar un modelo supervisado para predecir tiempos de vuelo en función de parámetros específicos, como las condiciones climáticas, el tráfico del aeropuerto, las horas pico de vuelo y más.

Por otro lado, el aprendizaje no supervisado es más útil para descubrir nuevos patrones y relaciones en datos sin procesar y sin etiquetar. Los modelos de aprendizaje no supervisados, por ejemplo, podrían usarse para identificar grupos de compradores que compran productos relacionados juntos para brindar sugerencias sobre otros artículos para recomendar a clientes similares.

Como resultado, se implementa el aprendizaje automático supervisado y no supervisado para resolver diferentes tipos de problemas. El aprendizaje automático supervisado es adecuado para tareas de clasificación y regresión, como pronóstico del tiempo, cambios de precios, análisis de opiniones y detección de spam. Mientras que el aprendizaje no supervisado se utiliza más comúnmente para el análisis exploratorio de datos y tareas de agrupación, como la detección de anomalías, la visualización de big data o la segmentación de clientes. En la figura 8 se ilustra un cuadro con las diferencias.

Figura 8

Diferencias entre modelo supervisado y no supervisado



2.6 Cómo elegir entre aprendizaje supervisado y no supervisado

Ahora que comprende las diferencias entre el aprendizaje supervisado y no supervisado, ¿qué enfoque es el adecuado para usted?

Elegir el enfoque correcto depende de sus objetivos y requisitos generales, los casos de uso que desea resolver y el enfoque general de su equipo para analizar, procesar y administrar datos.

Generalmente, deberá considerar lo siguiente al decidir qué opción funciona mejor para su organización.

- **¿Sus datos están etiquetados o no?** El aprendizaje supervisado requiere conjuntos de datos etiquetados. Deberá evaluar si su organización tiene el tiempo, los recursos y la experiencia para validar y etiquetar los datos.
- **¿Cuáles son tus metas?** Es importante considerar el tipo de problema que intenta resolver y si está intentando crear un modelo de predicción o buscando descubrir nuevos conocimientos o patrones ocultos en los datos.
- **¿Qué tipos de algoritmos necesitas?** Al decidir qué enfoque es mejor para su organización, también es importante evaluar si existen algoritmos que puedan soportar el volumen de datos y coincidir con las dimensiones requeridas, como la cantidad de características y atributos.

La elección entre aprendizaje supervisado o no supervisado se reduce al problema específico que desea resolver, los datos que tiene disponibles y si tiene las herramientas y la experiencia para crear y administrar sus modelos.

2.7 Aprendizaje por refuerzo

Consiste en aprender a decidir ante una situación determinada, qué acción es la más adecuada para lograr un objetivo. Es un enfoque computacional para la automatización de la comprensión y el aprendizaje orientado a objetivos y toma de decisiones. Se centra en el aprendizaje por parte del individuo desde su medio ambiente, no requiere de supervisión o modelos completos del ambiente. En el aprendizaje por refuerzo existe un agente que actúa en un entorno cuyo objetivo es maximizar las recompensas por sus acciones.

2.7.1 ¿Cómo funciona el aprendizaje por refuerzo?

El proceso de aprendizaje de los algoritmos de aprendizaje por refuerzo (RL) es similar al aprendizaje por refuerzo animal y humano en el campo de la psicología del comportamiento. Por ejemplo, un niño puede descubrir que recibe elogios de sus padres cuando ayuda a un hermano o a limpiar, pero recibe reacciones negativas cuando tira juguetes o grita. Pronto, el niño aprende qué combinación de actividades trae como resultado la recompensa final.

Un algoritmo RL imita un proceso de aprendizaje similar. Prueba diferentes actividades para aprender los valores negativos y positivos asociados para lograr el resultado final de la recompensa.

2.7.2 Conceptos clave

En el aprendizaje por refuerzo, hay algunos conceptos clave con los que debe familiarizarse:

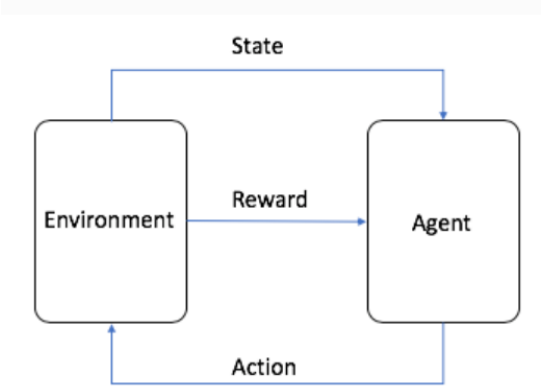
- El agente es el algoritmo ML (o el sistema autónomo)
- El entorno es el espacio de problemas adaptativo con atributos como variables, valores límite, reglas y acciones válidas
- La acción es un paso que el agente de RL realiza para navegar por el entorno

- El estado es el medio ambiente en un momento dado
- La recompensa es el valor positivo, negativo o cero (en otras palabras, la recompensa o el castigo) por llevar a cabo una acción
- La recompensa acumulada es la suma de todas las recompensas o el valor final

A continuación, se comparte el modelo en la figura 9.

Figura 9

Modelo de aprendizaje por refuerzo



2.7.2 Casos de Uso del Aprendizaje por Refuerzo

El aprendizaje por refuerzo puede ser usado en robots, por ejemplo en brazos mecánicos en donde en vez de enseñar instrucción por instrucción a moverse, podemos dejar que haga intentos “a ciegas” e ir recompensando cuando lo hace bien.

También puede usarse en ambientes que interactúan con el mundo real, como en otro tipo de maquinaria industrial y para el mantenimiento predictivo, pero también en el ambiente financiero, por ejemplo, para decidir cómo conformar una cartera de inversión sin intervención humana.

Otro caso de uso que está ganando terreno es el de usar RL para crear “webs personalizadas” para cada internauta. Y si lo piensas… tiene algo de sentido tomar el concepto de “premiar” al algoritmo si acierta con las sugerencias que hace al usuario si hace clic ó penalizar al modelo si sus recomendaciones no le son de utilidad.

También se utiliza el Reinforcement Learning para entrenar sistemas de navegación de coches, drones ó aviones.

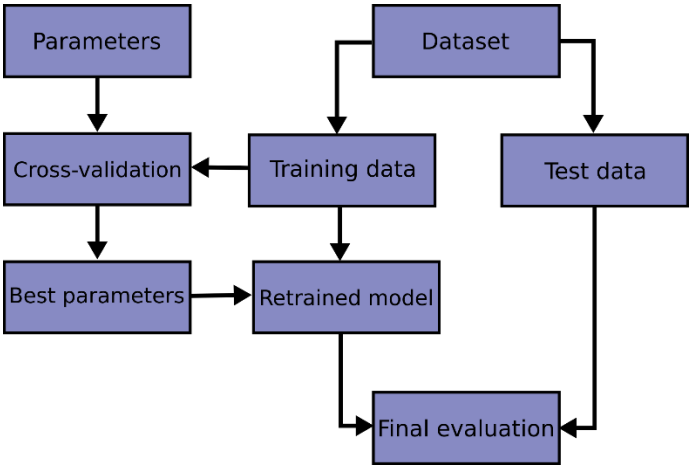
2.8 Evaluación y validación del modelo

2.8.1 Cross-validation: Evaluador de performance del estimador.

Aprender los parámetros de una función de predicción y probarla con los mismos datos es un error metodológico: un modelo que simplemente repetiría las etiquetas de las muestras que acaba de ver tendría una puntuación perfecta pero no predeciría nada útil todavía. datos no vistos. Esta situación se llama sobreajuste. Para evitarlo, es una práctica común al realizar un experimento de aprendizaje automático (supervisado) mantener parte de los datos disponibles como un conjunto de prueba. Tenga en cuenta que la palabra "experimento" no pretende denotar un uso académico únicamente, porque incluso en entornos comerciales el aprendizaje automático suele comenzar de forma experimental. A continuación, se muestra en la figura 10 un diagrama de flujo del flujo de trabajo típico de validación cruzada en el entrenamiento de modelos. Los mejores parámetros pueden determinarse mediante técnicas de búsqueda en cuadrícula .*X_test, y_test*

Figura 10

Diagrama de flujo del flujo de cross validation, evaluador de performance del estimador



A modo de ejemplo se plantea el siguiente escenario: En scikit-learn, se puede calcular rápidamente una división aleatoria en conjuntos de entrenamiento y prueba con la train_test_splitfunción auxiliar. Carguemos el conjunto de datos del iris para que se ajuste a una máquina de vectores de soporte lineal como se vé en la figura 11:

Figura 11

Carga de datos Iris.

```
>>> import numpy as np
>>> from sklearn.model_selection import train_test_split
>>> from sklearn import datasets
>>> from sklearn import svm

>>> X, y = datasets.load_iris(return_X_y=True)
>>> X.shape, y.shape
((150, 4), (150,))
```

Ahora podemos muestrear rápidamente un conjunto de entrenamiento mientras reservamos el 40% de los datos para probar (evaluar) nuestro clasificador tal como se vé en la figura 12:

Figura 12

Conjunto de entrenamiento

```
>>> X_train, X_test, y_train, y_test = train_test_split(
...     X, y, test_size=0.4, random_state=0)

>>> X_train.shape, y_train.shape
((90, 4), (90,))
>>> X_test.shape, y_test.shape
((60, 4), (60,))

>>> clf = svm.SVC(kernel='linear', C=1).fit(X_train, y_train)
>>> clf.score(X_test, y_test)
0.96...
```

Al evaluar diferentes configuraciones (“hiperparámetros”) para estimadores, como la Configuración que se debe configurar manualmente para una SVM, todavía existe el riesgo de sobreajuste en el conjunto de prueba porque los parámetros se pueden modificar hasta que el estimador funcione de manera óptima. De esta manera, el conocimiento sobre el conjunto de pruebas puede "filtrarse" al modelo y las métricas de evaluación ya no informan sobre el desempeño de la generalización. Para resolver este problema, se puede presentar otra parte del conjunto de datos como el llamado “conjunto de validación”: el entrenamiento continúa en el conjunto de entrenamiento, después de lo cual se realiza la evaluación en el conjunto de validación, y cuando el experimento parece tener éxito, la evaluación final se puede realizar en el conjunto de prueba.

Sin embargo, al dividir los datos disponibles en tres conjuntos, reducimos drásticamente la cantidad de muestras que se pueden usar para aprender el modelo, y los resultados pueden depender de una elección aleatoria particular para el par de conjuntos (entrenamiento, validación).

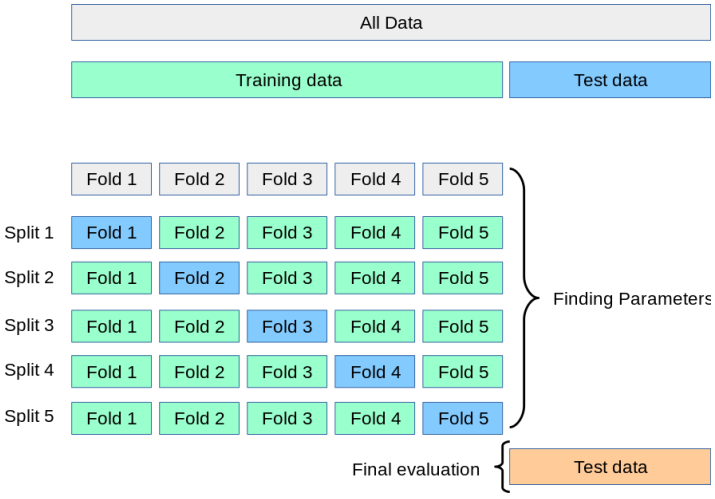
Una solución a este problema es un procedimiento llamado validación cruzada (CV para abreviar). Aún se debe reservar un conjunto de pruebas para la evaluación final, pero el conjunto de validación ya no es necesario al realizar el CV. En el enfoque básico, llamado k -fold CV, el conjunto de entrenamiento se divide en k conjuntos más pequeños (otros enfoques se describen a continuación, pero generalmente siguen los mismos principios). Se sigue el siguiente procedimiento para cada uno de los k “pliegues”:

- Un modelo se entrena usando K-1 de los pliegues como datos de entrenamiento;
- El modelo resultante se valida con la parte restante de los datos (es decir, se utiliza como conjunto de pruebas para calcular una medida de rendimiento como la precisión).

La medida de rendimiento informada por la validación cruzada de k veces es entonces el promedio de los valores calculados en el bucle. Este enfoque puede ser costoso desde el punto de vista computacional, pero no desperdicia demasiados datos (como es el caso cuando se fija un conjunto de validación arbitrario), lo cual es una gran ventaja en problemas como la inferencia inversa donde el número de muestras es muy pequeño tal como se vé en la Figura 13:

Figura 13

Modelos k -fold



2.8.2 Cross-validation: Métricas de validación.

La forma más sencilla de utilizar cross-validation es llamar a la ***cross_val_score***, función auxiliar en el estimador y el conjunto de datos.

El siguiente ejemplo demuestra cómo estimar la precisión de una máquina de vectores de soporte de núcleo lineal en el conjunto de datos del iris dividiendo los datos, ajustando un modelo y calculando la puntuación 5 veces consecutivas (con divisiones diferentes cada vez):

Figura 14

Cross-validation: Métricas de validación. ***cross_val_score***

```
>>> from sklearn.model_selection import cross_val_score
>>> clf = svm.SVC(kernel='linear', C=1, random_state=42)
>>> scores = cross_val_score(clf, X, y, cv=5)
>>> scores
array([0.96..., 1. , 0.96..., 0.96..., 1. ])
```

Por tanto, la puntuación media y la desviación standard se reflejan en la figura 15:

Figura 15

Puntuación media y desviación standard

```
>>> print("%0.2f accuracy with a standard deviation of %0.2f" % (scores.mean(), scores.std()))
0.98 accuracy with a standard deviation of 0.02
```

2.8.3 Matriz de confusión:

Una matriz de confusión es una tabla utilizada para describir el desempeño de un modelo de clasificación en un conjunto de datos de prueba cuyos valores verdaderos se conocen. Es más útil cuando necesita saber más sobre la precisión del modelo que solo la precisión general.

En el caso más simple, para un problema de clasificación binaria, la matriz de confusión consta de cuatro combinaciones diferentes de valores predichos y reales:

- Verdaderos Positivos (TP): Los casos en los que el modelo predijo correctamente la clase positiva.
- Verdaderos Negativos (TN): Los casos en los que el modelo predijo correctamente la clase negativa.
- Falsos positivos (FP): Los casos en los que el modelo predijo incorrectamente la clase positiva (también conocido como error de tipo I).
- Falsos negativos (FN): Los casos en los que el modelo predijo incorrectamente la clase negativa (también conocido como error de tipo II).

2.8.3.1 ¿Por qué es útil la Matriz de Confusión?

Las siguientes son algunas de las razones por las que la matriz de confusión es muy útil y los científicos deben dominar bien este concepto:

- **Medidas de desempeño:** más allá de la simple precisión (que puede ser engañosa, especialmente en conjuntos de datos desequilibrados), la matriz de confusión permite el cálculo de varias métricas de desempeño como precisión, recuperación, puntuación F1, especificidad, etc. Cada una de estas métricas brinda diferentes conocimientos sobre las fortalezas y debilidades del modelo.
- **Información sobre los errores:** proporciona información sobre los tipos de errores que comete el modelo. Por ejemplo, en el diagnóstico médico, un falso negativo puede ser más peligroso que un falso positivo.
- **Manejo de conjuntos de datos desequilibrados**: en los casos en que la distribución de clases no está equilibrada, la matriz de confusión ayuda a evaluar el rendimiento del modelo para cada clase por separado.
- **Ajuste del modelo:** comprender los tipos de errores y sus cantidades puede guiarle a la hora de ajustar su modelo. Por ejemplo, si hay demasiados falsos positivos, es posible que desee aumentar el umbral para predecir la clase positiva.

2.8.3.2 Ejemplo del mundo real de matriz de confusión

Tomemos un ejemplo del mundo real de un modelo de clasificación de incumplimiento de préstamos. En este escenario, el modelo tiene como objetivo predecir si un prestatario no pagará un préstamo (es decir, no lo pagará) en función de varios factores como puntaje crediticio, ingresos, monto del préstamo, etc.

Se recopilaron datos históricos de los prestatarios, incluido si incumplieron y otras características relevantes. Los datos se utilizaron para entrenar un modelo de clasificación (por ejemplo, un modelo de regresión logística) para predecir el incumplimiento. El modelo se evaluó en un

conjunto de datos de prueba separado que consta de 1000 prestatarios para ver qué tan bien predice los incumplimientos de los préstamos.

Digamos que el siguiente fue el resultado:

Verdaderos positivos (TP): 150

Verdaderos negativos (TN): 700

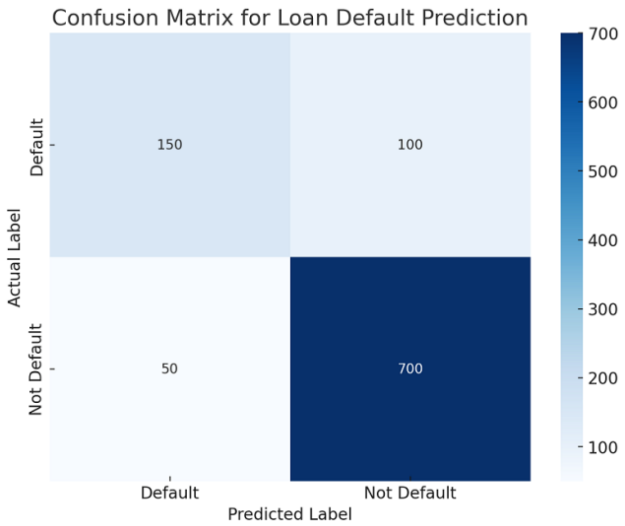
Falsos positivos (FP): 50

Falsos negativos (FN): 100

La matriz de confusión para este ejemplo se vería como refleja la figura 16:

Figura 16

Ejemplo de matriz de confusión



A continuación, se explica cómo se puede interpretar la matriz de confusión anterior:

TP (150): El modelo identificó correctamente a 150 prestatarios que incumplieron.

TN (700): El modelo identificó correctamente a 700 prestatarios que no incumplieron.

FP (50): El modelo identificó incorrectamente a 50 prestatarios con probabilidades de incumplir, pero no lo hicieron.

FN (100): El modelo omitió a 100 prestatarios que incumplieron, prediciendo que no lo harían.

2.8.3.3 Ejemplo de código Python para matriz de confusión

En esta sección, verá el ejemplo de código de matriz de confusión de Sklearn Python . El siguiente modelo se entrena utilizando el algoritmo clasificador de vectores de soporte (SVC). El paquete Sklearn.svm se utiliza para importar SVC. Los siguientes son algunos de los aspectos ilustrados en el ejemplo de código que se proporciona a continuación:

- El conjunto de datos Sklearn IRIS se utiliza para entrenar el modelo.
- El método train_test_split se utiliza para crear la división de datos de entrenamiento y prueba.
- La instancia de StandardScaler se utiliza para la estandarización de datos.
- El algoritmo SVC se utiliza para ajustar el modelo.
- Sklearn.metrics confusion_matrix se utiliza para calcular la matriz de confusión.

Así es como se verá el código tal como lo refleja la figura 17

Figura 17

Ejemplo de código Python para matriz de confusión

```python
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
import seaborn as sns

# Load Iris dataset
iris = load_iris()
X = iris.data
y = iris.target

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)

dieciséis # Train an SVC model
svc = SVC(gamma='auto')
svc.fit(X_train, y_train)

# Make predictions
y_pred = svc.predict(X_test)

# Construct the confusion matrix
conf_matrix_iris = confusion_matrix(y_test, y_pred)

# Plotting the confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_iris, annot=True, fmt="d", cmap="Blues",
            xticklabels=iris.target_names,
yticklabels=iris.target_names)

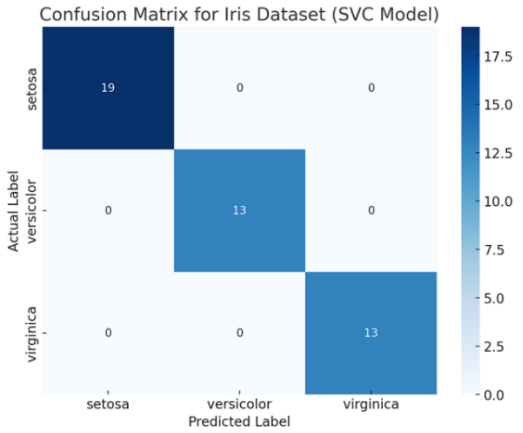
# Adding labels and title
plt.ylabel('Actual Label')
plt.xlabel('Predicted Label')
plt.title('Confusion Matrix for Iris Dataset (SVC Model)')

plt.show()
```

Así es como se vería la matriz de confusión como se refleja en la figura 18:

Figura 18

Ejemplo de Matriz de confusión con código Python



La imagen que se muestra arriba es la matriz de confusión para el conjunto de datos Iris, utilizando un clasificador de vectores de soporte (SVC). La matriz se construye a partir de las predicciones realizadas en el conjunto de prueba.

En esta matriz:

- Las filas representan las clases reales del conjunto de datos de Iris (setosa, versicolor, virginica).
- Las columnas representan las clases predichas por el modelo SVC.

Cada celda muestra el número de instancias de la clase real (fila) que se predijeron como una determinada clase (columna). Por ejemplo, una celda en la intersección de "versicolor" real y "virginica" predicha mostraría la cantidad de veces que el modelo predijo incorrectamente un iris versicolor como virginica.

Esta matriz de confusión proporciona una visión clara de qué tan bien se está desempeñando el modelo para cada clase, indicando cualquier sesgo o tendencia en sus predicciones.

2.8.4 Errores Tipo I y Tipo II

El problema radica en que al medir la precisión del algoritmo de esa forma no distinguimos entre los errores de tipo falso positivo y falso negativo, como si ambos tuvieran la misma importancia. Y esto no es así. Lo vamos a ver más claro con un ejemplo

- **Falso positivo o “Error tipo I”:** El paciente no tiene cáncer, pero el algoritmo ha diagnosticado que sí lo padece. Se realizarán pruebas diagnósticas adicionales que acabarán descartando el diagnóstico. Tendrá un costo económico y un impacto emocional sobre el paciente, pero no se traducirá en riesgo vital

- **Falso negativo o "Error tipo II"**: El paciente sí tiene cáncer, pero el algoritmo predice que no. Este error del algoritmo se traduce en una falta de detección temprana de la enfermedad. El paciente no recibirá tratamiento a tiempo y esto, indudablemente reduce sus posibilidades de superar la enfermedad

Figura 19

Ejemplo de Hipótesis

		True condition	
	Total population	Condition positive	Condition negative
Predicted condition	Predicted condition positive	**True positive**, Power	**False positive**, Type I error
	Predicted condition negative	**False negative**, Type II error	**True negative**

True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$
False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$

Figura 20

Validación de hipótesis

Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$	F_1 score = $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$		

2.8.5 ¿Qué es Precision score?

El aprendizaje automático mide la proporción de etiquetas predichas positivamente que en realidad son correctas. La precisión también se conoce como valor predictivo positivo. La precisión score se utiliza junto con la recall score para compensar los falsos positivos y los falsos negativos. Precision score se ve afectada por la distribución de clases. Si hay más muestras en la clase minoritaria, la precisión será menor. Si queremos minimizar los falsos positivos, elegiríamos un modelo con alta precisión. Por el contrario, si queremos minimizar los falsos negativos, elegiríamos un modelo con alto recuerdo. La precisión se utiliza principalmente cuando necesitamos predecir la clase positiva y hay un mayor costo asociado con los falsos positivos que con los falsos negativos, como en el diagnóstico médico o el filtrado de spam. Por ejemplo, si un modelo tiene una precisión del 99% pero solo tiene un 50% de precisión, eso significa que la mitad de las veces que predice que un correo electrónico es spam, en realidad no es spam. La siguiente es la fórmula para calcular precisión score:

Precision score = Verdaderos positivos/ (falsos positivos + Verdaderos positivos)

A partir de la fórmula anterior, se puede observar que el valor de falso positivo afectaría la puntuación de precisión. Por lo tanto, al crear modelos predictivos con alta precisión, puede optar por crear modelos con menos falsos positivos.

2.8.5.1 ¿Cómo calcular la precisión en Python?

Se puede obtener la misma puntuación utilizando el método precision_score de sklearn.metrics dado que y_test representa las etiquetas reales del conjunto de datos de prueba y y_pred representa su predicción tal como muestra la figura 21

Figura 21

Ejemplo de cálculo de precisión_score en Python

```
print('Precision: %.3f' % precision_score(y_test, y_pred))
```

2.8.5.2. Escenarios del mundo real en los que precisión score se puede utilizar como métricas de evaluación

Ejemplo n.º 1: Lo ideal es que los oncólogos quieran modelos que puedan identificar todas las lesiones cancerosas sin ningún resultado falso positivo o mínimos y, por lo tanto, se podría utilizar una puntuación de precisión en tales casos. Tenga en cuenta que un mayor número de falsos positivos provocará mucho estrés para los pacientes en general, aunque esto puede no resultar fatal desde el punto de vista de la salud. Otras pruebas podrán negar la predicción falsa positiva.

Ejemplo #2: En problemas de detección de fraude con tarjetas de crédito, los modelos de clasificación se evalúan utilizando la puntuación de precisión para determinar cuántas muestras positivas fueron clasificadas correctamente por el modelo de clasificación. No le gustaría tener una gran cantidad de falsos positivos o podría terminar bloqueando muchas tarjetas de crédito y, por lo tanto, muchas frustraciones con los usuarios finales.

Ejemplo #3: Una mayor cantidad de falsos positivos en un filtro de spam significaría que uno o más correos electrónicos importantes podrían etiquetarse como spam y moverse a carpetas de spam. Esto podría obstaculizar de muchas maneras diferentes, incluido el impacto en su trabajo diario.

2.8.6 ¿Qué es Recall Score?

Recall Score en el aprendizaje automático representa la capacidad del modelo para predecir correctamente los aspectos positivos a partir de los positivos reales. Esto es diferente a la precisión, que mide cuántas predicciones hechas por los modelos son realmente positivas de todas las predicciones positivas realizadas. Por ejemplo: si su modelo intenta identificar reseñas positivas, la puntuación de recuerdo sería el porcentaje de esas reseñas positivas que predijo correctamente como positivas. Recall score también se conoce **como sensibilidad o tasa de verdaderos positivos.**

Cuanto mayor sea Recall score, mejor será el modelo para identificar ejemplos tanto positivos como negativos. Por el contrario, una puntuación de Recall score baja indica que el modelo no es bueno para identificar ejemplos positivos.

El Recall score se utiliza a menudo junto con otras métricas de rendimiento, como la puntuación de precisión y la exactitud, para obtener una imagen completa del rendimiento del modelo. La siguiente es la fórmula para calcular Recall Score.

Recall Score = verdaderos positivos / (falsos negativos + verdaderos positivos)

A partir de la fórmula anterior, se puede observar que el valor de falso positivo afectaría la puntuación de precisión. Por lo tanto, es deseable crear modelos con menos falsos negativos si una puntuación de recuperación alta es importante para los requisitos comerciales.

La puntuación de recuperación de la matriz de confusión anterior será la siguiente:

Puntuación de recuerdo = 61 / (2 + 61) = 61/63 = 0,968

2.8.6.1 Cómo calcular la Recall Score en Python

La Recall score se puede obtener utilizando el método de recuperación_score de sklearn.metrics tal como se muestra en la figura 22

Figura 22

Ejemplo de cálculo de recall_score en Python

```
1 | print('Recall: %.3f' % recall_score(y_test, y_pred))
```

La Recall score se puede utilizar en escenarios en los que las etiquetas no están divididas equitativamente entre las clases. Por ejemplo, si hay una proporción de desequilibrio de clases de 20:80 (datos desequilibrados), entonces la Recall score será más útil que la precisión porque puede proporcionar información sobre qué tan bien el modelo de aprendizaje automático identificó eventos más raros.

2.8.6.2 Escenarios del mundo real en los que las Recall scores se pueden utilizar como métricas de evaluación

La Recall Score en el aprendizaje automático es una métrica importante a considerar al medir la efectividad de sus modelos. Se puede utilizar en una variedad de escenarios del mundo real y es importante apuntar siempre a mejorar juntos las Recall score y precisión. Los siguientes son ejemplos de algunos escenarios del mundo real donde las puntuaciones de recuperación se pueden utilizar como métricas de evaluación:

Ejemplo #1: En el diagnóstico médico, la puntuación de recuerdo debe ser extremadamente alta, de lo contrario, un mayor número de falsos negativos resultaría fatal para la vida de los pacientes. La puntuación de recuerdo más baja significaría un mayor falso negativo, lo que

esencialmente significaría que algunos pacientes que son positivos se denominan falsamente negativos. Eso significaría que los pacientes tendrían la seguridad de que no padecen la enfermedad y, por lo tanto, no tomarán ninguna medida adicional. Esto podría provocar que la enfermedad se agrave y resulte fatal para la vida.

Ejemplo #2: En la fabricación, es importante identificar los defectos en los productos lo antes posible para evitar producir productos defectuosos y desperdiciar recursos. Los modelos de clasificación se pueden entrenar para identificar defectos y anomalías. Una puntuación de recuperación alta es importante en este escenario porque indica que el modelo es capaz de identificar correctamente todos los casos de defectos, incluidos aquellos que pueden ser raros o difíciles de detectar. Un modelo con una puntuación de retirada baja puede pasar por alto algunos casos de defectos, lo que genera productos defectuosos y posibles problemas de seguridad para los consumidores.

Ejemplo #3: En un sistema de detección de fraude con tarjetas de crédito, usted querrá tener una puntuación de recuperación más alta para predecir transacciones fraudulentas. Una puntuación de recuperación más baja significaría un mayor falso negativo, lo que significaría un mayor fraude y, por tanto, una pérdida para el negocio en términos de usuarios molestos.

2.8.7 Precision – Recall Score Tradeoff

La compensación entre la Recall score y la precisión es un problema común que surge al evaluar el desempeño de un modelo de clasificación. Estas métricas a menudo entran en conflicto entre sí.

En general, aumentar la puntuación de precisión de un modelo disminuirá su puntuación de recuperación y viceversa. Esto se debe a que la precisión y la recuperación están inversamente relacionadas: mejorar una normalmente resultará en una disminución de la otra. Por ejemplo, un modelo con una puntuación de alta precisión hará pocas predicciones falsas positivas, pero también puede pasar por alto algunos casos verdaderos positivos. Por otro lado, un modelo con un alto recuerdo identificará correctamente la mayoría de los casos verdaderos positivos, pero también puede hacer más predicciones falsas positivas.

Para evaluar el modelo, es importante considerar tanto la recuperación como la puntuación de precisión, en lugar de solo una de estas métricas. El equilibrio adecuado entre la puntuación de recuperación de precisión dependerá de los objetivos y requisitos específicos del modelo, así como de las características del conjunto de datos. En algunos casos, puede ser más importante tener una alta precisión (por ejemplo, en el diagnóstico médico), mientras que en otros, un alto recuerdo puede ser más importante (por ejemplo, en la detección de fraude).

Para equilibrar la puntuación de recuperación de precisión en el aprendizaje automático, los profesionales suelen utilizar la puntuación F1 , que es una combinación de las dos métricas. Se analiza en la sección posterior. Sin embargo, incluso la puntuación F1 no es una solución perfecta, ya que puede resultar difícil determinar el equilibrio óptimo entre la puntuación de recuperación de precisión para una aplicación determinada.

2.8.8 ¿Qué es Accuracy Score?

Accuracy score son las métricas que se definen como la proporción de verdaderos positivos y verdaderos negativos con respecto a todas las observaciones positivas y negativas. En otras palabras, la precisión nos dice con qué frecuencia podemos esperar que nuestro modelo de

aprendizaje automático prediga correctamente un resultado del número total de veces que hizo predicciones. Por ejemplo: supongamos que estaba probando su modelo con un conjunto de datos de 100 registros y que su modelo de aprendizaje automático predijo correctamente las 90 instancias. La métrica de precisión, en este caso, sería: (90/100) = 90%. La tasa de precisión es excelente, pero no nos dice nada sobre los errores que cometen nuestros modelos de aprendizaje automático en datos nuevos que no hemos visto antes. La siguiente es la fórmula de la puntuación de precisión.

Accuracy score = (TP + TN)/ (TP + FN + TN + FP)

2.8.8.1 ¿Cómo calcular Accuracy score en Python?

Se puede obtener la misma puntuación utilizando el método precision_score de sklearn.metrics tal como se muestra en la figura 23.

Figura 23

Ejemplo de cálculo del Accurancy en Python

```
1 | print('Accuracy: %.3f' % accuracy_score(y_test, y_pred))
```

2.8.8.2 Precaución con el Accuracy score

Los siguientes son algunos de los problemas con las métricas /puntuaciones de precisión:

- Las mismas métricas de precisión para dos modelos diferentes pueden indicar un rendimiento diferente del modelo hacia diferentes clases.
- En caso de un conjunto de datos desequilibrado, las métricas de precisión no son las métricas más efectivas que se pueden utilizar.

Se debe tener cuidado al confiar en las métricas de precisión del modelo para evaluar el desempeño del modelo.

Las métricas de precisión tampoco son confiables para los modelos entrenados en conjuntos de datos desequilibrados o sesgados. Tomemos un escenario de conjunto de datos con un desequilibrio del 95 % (el 95 % de los datos son de clase negativa). La precisión del clasificador será muy alta ya que realizará correctamente la predicción correcta y emitirá resultados negativos la mayor parte del tiempo. Es probable que un clasificador mejor que realmente aborde el problema del desequilibrio de clases tenga una puntuación de métricas de precisión peor. En tal escenario de conjunto de datos desequilibrado, otra métrica AUC (el área bajo la curva ROC) es más sólida que la puntuación de la métrica de precisión.

Las métricas de precisión no tienen en cuenta la importancia relativa de los diferentes tipos de errores, como los falsos positivos y los falsos negativos. Por ejemplo, si se utiliza un modelo para predecir si un paciente tiene una determinada enfermedad, un falso positivo (predecir que un paciente tiene la enfermedad cuando en realidad no la tiene) puede ser menos grave que un falso negativo (predecir que un paciente sí la padece). no tienen la enfermedad cuando en realidad la tienen). En este caso, utilizar la precisión como única métrica de evaluación puede no proporcionar una imagen clara del rendimiento del modelo.

2.8.9 ¿Qué es F1-Score?

F1 Score es la media armónica de la precisión y la puntuación de recuperación. Da el mismo peso tanto a la Precisión como a la Recuperación para medir su rendimiento en términos de precisión, lo que la convierte en una alternativa a las métricas de Precisión (no requiere que conozcamos el número total de observaciones). A menudo se utiliza como un valor único que

proporciona información de alto nivel sobre la calidad de salida del modelo. Esta es una medida útil del modelo en los escenarios en los que se intenta optimizar la precisión o la puntuación de recuperación y, como resultado, el rendimiento del modelo se ve afectado.

A continuación, se representan los aspectos relacionados con los problemas relacionados con la optimización de la precisión o la puntuación de recuperación:

- La optimización del recuerdo ayuda a minimizar la posibilidad de no detectar un cáncer maligno. Sin embargo, esto tiene el coste de predecir cáncer maligno en pacientes, aunque los pacientes estén sanos (un número elevado de FP).
- Optimizar para obtener precisión ayuda con la corrección si el paciente tiene cáncer maligno. Sin embargo, esto tiene el costo de pasar por alto el cáncer maligno con mayor frecuencia (un número elevado de FN).

La siguiente es la fórmula de F1 Score :

F1 Score = 2* Precision Score * Recall Score/ (Precision Score + Recall Score)

2.8.9.1 Como calcular F1-Score en Python

Se puede obtener la misma puntuación utilizando el método f1_score de sklearn.metrics tal como se muestra en la figura 24.

Figura 24

Ejemplo de cálculo de F1_score en Python

```
1 | print('F1 Score: %.3f' % f1_score(y_test, y_pred))
```

2.8.10 FAQs de precision, recall, accuracy y F1-score.

2.8.10.1 ¿Cuál es la diferencia entre Recall y Precision score?

Recall mide la proporción de casos positivos reales identificados correctamente por un modelo, centrándose en minimizar los falsos negativos. Precision, por otro lado, mide la proporción de identificaciones positivas que realmente fueron correctas, con el objetivo de minimizar los falsos positivos. En resumen, Recall consiste en capturar tantos aspectos positivos como sea posible, mientras que precisión consiste en ser exacto en los aspectos positivos identificados.

2.8.10.2 ¿Cuándo es más importante precisión que Recall?

Precisión es más importante que la recuperación en escenarios donde el costo de los falsos positivos es alto. Por ejemplo, en la detección de spam, es fundamental evitar clasificar erróneamente los correos electrónicos legítimos como spam. En las pruebas médicas, un falso positivo podría dar lugar a procedimientos invasivos e innecesarios. En estos casos, es preferible tener un modelo muy preciso que identifique con confianza los verdaderos positivos, incluso si pasa por alto algunos casos positivos, en lugar de clasificar erróneamente los negativos como positivos.

2.8.10.3 ¿Cuáles son los buenos valores de precisión y recuperación?

Una buena precisión y valores de Recall dependen de la aplicación específica y del equilibrio entre el costo de los falsos positivos y los falsos negativos. En algunos contextos, como los diagnósticos médicos o la detección de fraudes, la alta precisión es crucial para evitar falsas alarmas. En otros, como los sistemas de monitoreo de seguridad, una alta Recall es más importante para garantizar que no se pase por alto ninguna amenaza. Idealmente, ambos deberían ser lo más altos posible, pero un valor superior a 0,5 a menudo se considera aceptable y los

valores más cercanos a 1 son excelentes. El umbral aceptable varía mucho según las necesidades y tolerancias específicas de la tarea en cuestión.

2.8.10.4 Conclusiones

Aquí está el resumen de lo que aprendió en relación con Precision, Recall, Accurency y F1 Score.

- Se utiliza Precision para medir el rendimiento del modelo a la hora de medir el recuento de verdaderos positivos de la manera correcta entre todas las predicciones positivas realizadas.
- Recall se utiliza para medir el rendimiento del modelo en términos de medir el recuento de verdaderos positivos de manera correcta entre todos los valores positivos reales.
- Precision-Recall es una medida útil del éxito de la predicción cuando las clases están muy desequilibradas.
- Precisio se utiliza para medir el rendimiento del modelo en términos de medir la relación entre la suma de verdaderos positivos y verdaderos negativos de todas las predicciones realizadas.
- F1-Score se utiliza en el escenario en el que elegir la puntuación de precisión o de Recall puede resultar en un compromiso en términos de que el modelo dé altos falsos positivos y falsos negativos, respectivamente. Ver figura 25.

2.8.10.5 En resumen:

- **Precision** es la probabilidad de que un documento recuperado (seleccionado al azar) sea relevante. Ver figura 25.

- **Recall** es la probabilidad de que un documento relevante (seleccionado al azar) se recupere en una búsqueda. Ver figura 26.

Figura 25

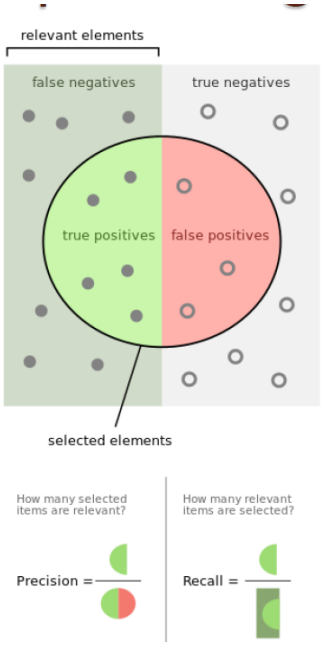
Formula de Precision y Recall

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

Figura 26

Formula de F-Measure



F-Measure combines precision and recall by the harmonic mean of precision and recall

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Trabajos relacionados

En este capítulo se verán los trabajos relacionados con la tesis, sobre las distintas maneras que se abordan la predicción de las lesiones en los deportes con un enfoque fisiológico y psicológico por separado.

En la actualidad se encuentran diversos trabajos publicados donde se pueden encontrar la aplicación de AI para la predicción de las lesiones en diferentes deportes.

A alto nivel se detectan diferentes técnicas aplicadas de IA como son: redes neuronales, decision tree classifier, support vector machine, y procesos de Markov con buenas métricas de performance. Futbol, basquetbol, handball, y voleibol son los deportes donde más se aplica AI. [Claudino et al. Sports Medicine – Open,2019, Current Approaches to the Use of Artificial Intelligence for Injury Risk Assessment and Performance Prediction in Team Sports: a Systematic Review].

3.1 Trabajos relacionados con enfoque fisiológico

Las lesiones en los deportes profesionales ocurren de forma regular. Algunos pueden ser menores, mientras que otros pueden causar un gran impacto en la carrera y el potencial de ingresos de un jugador. En el fútbol, existe un alto riesgo de que los jugadores sufran lesiones durante el tiempo de juego. Este trabajo de investigación busca ayudar a los jugadores de fútbol a reducir el riesgo de lesionarse prediciendo la probabilidad de sufrir lesiones mientras juegan en un futuro cercano y luego brindando recomendaciones de intervención. La herramienta de predicción de lesiones utilizará la cantidad de minutos jugados en el campo de un jugador de fútbol, la cantidad de apariciones, la distancia recorrida y los datos de rendimiento de la temporada actual y anterior como variables para realizar análisis estadísticos y proporcionar resultados predictivos de lesiones utilizando un modelo de regresión lineal de aprendizaje

automático.[Amiel atvedi,2020, Injury Predictor For Soccer Players Using Machine Learning, Sheridan College (Oakville)]

Las investigaciones realizadas hasta el momento ponen de manifiesto que las ecuaciones de regresión logística pueden ser un método válido y fiable para la predicción y prevención de lesiones, si bien se considera incompleto. Los modelos matemáticos desarrollados hasta la fecha sólo tienen en cuenta variables antropométricas y biomecánicas y no incluyen factores psicológicos, situacionales o ambientales que, como se ha demostrado, también influyen en la aparición de lesiones en el deportista. Por este motivo, se observa la necesidad de desarrollar herramientas y estrategias específicas para la medición de las características psicológicas y de factores externos que afectan a la vulnerabilidad del deportista con el objeto de poder realizar una mejora de las ecuaciones de regresión logística ya existentes.

Asimismo, se proponen modelizaciones matemáticas que se adapten a cada modalidad deportiva y a miembros superiores e inferiores, además de la realización del diseño y la implementación de herramientas informáticas que realicen estos cálculos. El entrenador o especialista del deporte sólo tendría que introducir los datos concretos del atleta para saber si existe riesgo de lesión, por lo que dichas aplicaciones servirían para facilitar su trabajo, y su existencia produciría un aumento del uso de las ecuaciones de regresión logística, con el fin de anticipar y prevenir las lesiones de los deportistas.

Por lo tanto, se concluye que sería necesario seguir investigando para consolidar la modelización matemática como instrumento para el pronóstico de lesiones en el futuro. [María del Carmen Juan Llamas,2021, Modelización matemática para la predicción y prevención de lesiones deportivas, Universidad Complutense de Madrid].

Las lesiones tienen un gran impacto en el fútbol profesional, debido a su gran influencia en el rendimiento del equipo y los considerables costos de rehabilitación para los jugadores. Los estudios existentes en la literatura proporcionan sólo una información preliminar de comprensión de qué factores afectan principalmente el riesgo de lesiones, mientras que todavía falta conocer una evaluación el potencial de los modelos estadísticos para pronosticar las lesiones. En este artículo, proponemos un enfoque multidimensional para la predicción de lesiones en fútbol profesional que se basa en mediciones GPS y aprendizaje automático. Mediante el uso de tecnología de seguimiento GPS, recopilamos datos que describen el entrenamiento. Carga de trabajo de los jugadores de un club de fútbol profesional durante una temporada. Mostramos que nuestros predictores de lesiones sean precisos e interpretables al proporcionar un conjunto de estudios de caso de interés para los practicantes de fútbol. Nuestro enfoque abre una perspectiva novedosa sobre la prevención de lesiones, que proporciona un conjunto de medidas sencillas y reglas prácticas para evaluar e interpretar las complejas relaciones entre el riesgo de lesión y rendimiento formativo en el fútbol profesional. [Alessio Rossi, Luca Pappalardo , Paolo Cintia , F. Marcello Iaia ,Javier Fernandez , Daniel Medina,2017, Effective injury prediction in professional soccer with GPS data and machine learning].

3.2 Trabajos relacionados con enfoque psicológico

Las lesiones deportivas constituyen un aspecto de marcada relevancia para la comunidad científica internacional debido a sus altas tasas de incidencia y prevalencia en todos los deportes a nivel global, constituyéndose como un factor de riesgo inherente a la práctica deportiva que en su materialización produce afectaciones sobre el estado de salud y el rendimiento de los

deportistas con los consecuentes gastos económicos que implica el tratamiento y la rehabilitación (García et al., 2015; Padegimas et al., 2016; Pujals et al., 2016).

La naturaleza multicausal de este flagelo ha propiciado que con mayor frecuencia sea abordado a partir de un enfoque multidisciplinar donde la Psicología juega un importante rol en la prevención, el tratamiento, la rehabilitación y la vuelta al terreno del deportista lesionado. La extensa producción científica acumulada desde los primeros estudios en la década del 70 del pasado siglo XX, ha permitido arribar a consensos sobre las relaciones que establecen determinadas variables psicológicas con las lesiones (Herring et al., 2017; Schinke et al., 2017; Soligard, et al., 2016).

A partir del modelo teórico propuesto por Andersen y Williams (1988) y su actualización diez años después (Williams y Andersen, 1998), que explica cómo la relación entre el estrés psicosocial y los recursos de afrontamiento del deportista configuran la vulnerabilidad a la lesión, múltiples estudios han contribuido a indicar evidencia de la mediación de variables psicológicas en la ocurrencia de las lesiones (Glendon, Clarke y McKenna, 2016; Gómez-Espejo et al., 2017; Johnson e Ivarsson, 2017; Olmedilla, Rubio y Ortega, 2015; Singh y Conroy, 2017).

No obstante, de acuerdo con el Modelo Global Psicológico de las Lesiones Deportivas de Olmedilla y García-Mas (2009) el estudio de la relación entre factores psicológicos y lesiones deportivas ha mostrado una gran diversidad de resultados atendiendo a la heterogeneidad de las investigaciones en cuanto a la selección de las muestras, los instrumentos de evaluación utilizados, los deportes objeto de estudio y los niveles de profesionalidad de los deportistas, así como los análisis estadísticos. Estos aspectos han obstaculizado la sistematización teórica conceptual de esta relación tal como afirman Ortín et al. (2010) y Palmi y Solé (2014).

En consecuencia, la predicción de lesiones teniendo en cuenta variables psicológicas cuenta con un reducido número de estudios en los últimos años (Berengüí y Puga, 2015; García-Mas et al., 2014; Olmedilla et al., 2018; Rubio et al., 2014), los cuales se han desarrollado fundamentalmente en deportes como el fútbol, baloncesto, balonmano, atletismo, ciclismo, lucha olímpica y taekwondo hallando resultados igualmente divergentes a los estudios descriptivos y correlacionales.

Si bien la evidencia científica ha mostrado la relevancia de los factores psicológicos como antecedentes y consecuentes de las lesiones, aún no se tienen en cuenta de manera suficiente para su prevención, por lo que la predicción psicológica de las mismas constituye un área del conocimiento de relevancia y actualidad debido a los escasos estudios que han determinado predictores psicológicos de lesiones en un grupo reducido de deportes.

Al respecto, Juan (2021) expresó que la determinación de los predictores de lesiones mediante modelos matemáticos constituye una herramienta de alto valor para entrenadores y deportistas para su prevención, no obstante, la mayoría de las modelaciones realizadas no tienen en cuenta todos los factores de riesgo como es el caso de los aspectos psicológicos. Además, la mayoría de las modelaciones se han realizado mediante modelos de regresión lineal por lo que resulta necesario emplear métodos informáticos que faciliten la labor de los especialistas del deporte en este sentido.

Como menciona [Jesús Ríos Garit, Yanet Pérez Surita, María del Carmen Chávez Cárdenas,2021, Predictores psicológicos de lesiones en lanzadores de béisbol de alto Rendimiento, Universidad Central “Marta Abreu” de Las Villas].

Se logró determinar que cuando un lanzador de lanzadores de beisbol presenta una autoconfianza baja y experimenta un alto nivel de ansiedad en la competencia, el control de afrontamiento negativo y visuoimaginativo es bajo, lo cual condiciona una alta probabilidad de la lesión. Estas relaciones entre variables psicológicas asociadas al rendimiento deportivo configuran el mayor grado de vulnerabilidad a la lesión en los lanzadores estudiados.

Este hallazgo permite orientar la preparación psicológica de los lanzadores analizados para potenciar el desarrollo de la autoconfianza y del autocontrol emocional ante situaciones potencialmente estresantes, contribuyendo a la prevención de la lesión.

Si bien los hallazgos obtenidos aportan al conocimiento sobre la configuración del riesgo psicológico a las lesiones en lanzadores de béisbol de alto rendimiento, se considera necesario la realización de nuevos estudios donde se tengan en cuenta las relaciones entre múltiples factores de riesgo de diversa naturaleza para desarrollar programas integrales de prevención de lesiones.

Mediante la administración del 16PF-5, IPED y SCAT, se obtiene que los deportistas con bajas puntuaciones en Estabilidad Emocional y Control de Afrontamiento Negativo, y altas en Ansiedad, presentan un mayor promedio de lesiones. Los sujetos sin lesiones se diferencian de los lesionados por mayores niveles de Estabilidad Emocional, Privacidad y Apertura al cambio, Control de Afrontamiento, y menores de Ansiedad. Existen correlaciones estadísticamente significativas entre el número de lesiones y las puntuaciones de Estabilidad Emocional, Tensión, Aprensión, Ansiedad, y ansiedad rasgo competitiva [María Dolores Hidalgo,2013, Características psicológicas asociadas a la incidencia de lesiones en deportistas de modalidades individuales, Grados de Educación y Psicología, Universidad Católica San Antonio de Murcia].

Propuesta

En este capítulo se desarrollará la hipótesis, el enfoque propuesto, así como también la explicación y justificación de los algoritmos utilizados.

4.1 Hipótesis

Se plantea la hipótesis en la cual, a partir de un set de datos como input, se obtiene información tanto fisiológica como psicológica de jugadores de vóley, a este set de datos se aplica distintos algoritmos de ML enfocados en modelos supervisados, con el objetivo de poder predecir qué tipo de lesión puede llegar a tener un jugador, esta lesión puede tener 3 valores: Grave, moderada o leve.

4.2 Enfoque propuesto

El enfoque propuesto parte tomando como datos de entradas un archivo Excel CSV y los algoritmos aplicados junto con la precisión y validación de los mismos, así como también el análisis de las variables de respuesta.

En el set de datos se distinguen las siguientes columnas:

- Jugador
- Diagnostico
- Tipo
- Región
- Preguntas desde la 1hasta la 12.

Como paso siguiente se levanta el archivo en Python utilizando Jupyter como software.

Los datos provienen de la selección juvenil de voleibol de la Argentina, categorías SUB 19, SUB 21 y SUB 23.

En un archivo Excel se procedió a trabajar con los datos de manera de prepararlos para su proceso en Python. En un primer paso se los limpio de espacios en blanco, se corrigieron los tipos y se estructuraron para ser bien interpretados.

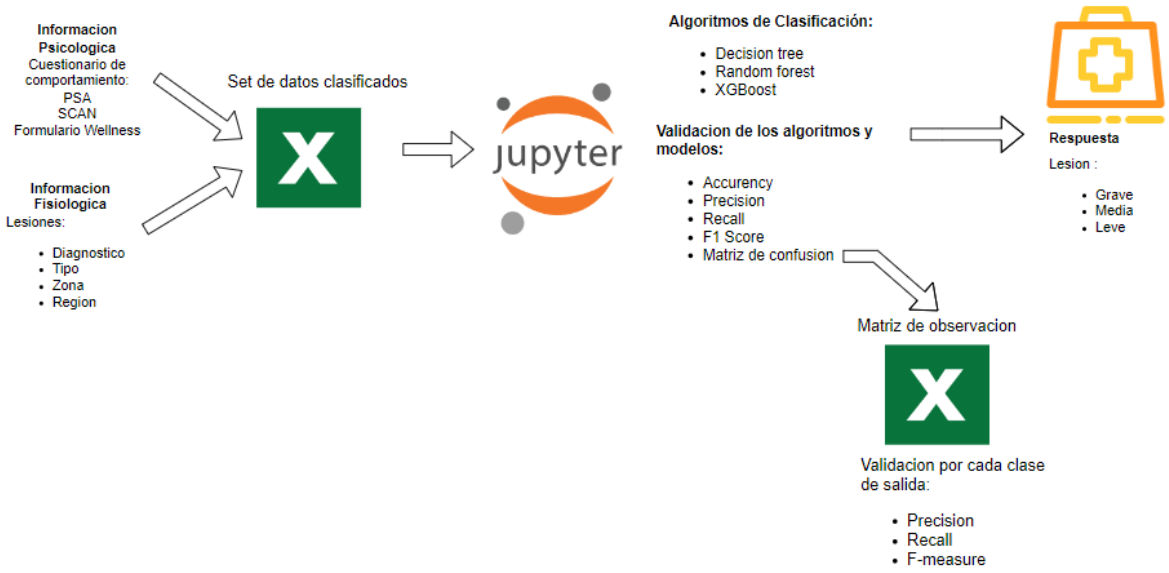
Finalmente se obtiene un archivo CSV el cual es trabajado en Python para su posterior análisis.

En el mismo archivo conviven los datos fisiológicos y psicológicos de los jugadores.

En la figura 27 se detalla la arquitectura propuesta para la solución:

Figura 27

Arquitectura propuesta



4.2.1 Proceso

A continuación, se describe el proceso utilizado para plantear la solución:

- Definición de problemas y recopilación de datos.
- Preprocesamiento de datos
- Ingeniería de características
- Selección de modelo
- Entrenamiento modelo
- Evaluación del modelo
- Ajuste del modelo
- Implementación del modelo

4.2.2 Definición de problemas y recopilación de datos.

En el mundo del aprendizaje automático, el primer paso es definir el problema que se quiere resolver. Por ejemplo, digamos que desea crear un modelo para predecir la probabilidad de que un cliente compre en su tienda en línea. Para hacer esto, necesitará recopilar datos sobre sus clientes y sus compras anteriores. Puede recopilar esta información encuestando a sus clientes, rastreando su historial de compras o ambas cosas.

4.2.3 Preprocesamiento de datos

El preprocesamiento de datos es un paso crítico en el proceso de aprendizaje automático supervisado. Es como darles a nuestros datos un día de spa para que estén actualizados y listos para modelar. Estos son los pasos involucrados en el preprocesamiento de datos:

- Manejo de valores faltantes: este es un paso importante para evitar cualquier sesgo en nuestro modelo. Existen varias técnicas para manejar los valores faltantes, como reemplazar los valores faltantes con la media, la mediana o la moda de la columna.
- Normalizar los datos: este paso es importante para garantizar que todas las variables estén en una escala similar y evitar que alguna variable domine el modelo. Existen varias técnicas para normalizar los datos, como escalar los datos entre 0 y 1 o normalizarlos para que tengan una media de 0 y una desviación estándar de 1.
- Dividir los datos en conjuntos de entrenamiento y prueba: este paso es importante para evitar el sobreajuste y evaluar el rendimiento de nuestro modelo. Dividimos nuestros datos en un conjunto de entrenamiento y un conjunto de prueba. El modelo se entrena en el conjunto de entrenamiento y se evalúa en el conjunto de prueba.

Al completar estos tres pasos, nuestros datos ahora están limpios y preparados para el modelado.

4.2.4 Ingeniería de características

La ingeniería de funciones es un paso crucial en el proceso de aprendizaje automático supervisado.

- Extracción de información de datos de texto: este paso es importante cuando se trata de datos de texto, como reseñas de clientes o tweets. Podemos extraer información relevante como el sentimiento del texto o la presencia de palabras clave específicas
- Transformar datos numéricos en datos categóricos: este paso es importante cuando se trata de datos numéricos, como la edad del cliente o el salario. Podemos transformar datos numéricos en datos categóricos, como grupos de edad o rangos salariales.

Al completar estos pasos, nuestros datos sin procesar se transformaron en funciones útiles que ayudarán a nuestro modelo a realizar predicciones precisas.

4.2.5 Selección de modelo

Según el problema y los datos, ahora puede elegir un modelo de aprendizaje automático que se adapte bien.

Por ejemplo, si el problema es un problema de regresión lineal y los datos se comportan bien, puede elegir un modelo de regresión lineal. Si el problema es de clasificación y los datos no son lineales, puede elegir un modelo de árbol de decisión.

4.2.6 Entrenamiento modelo

Usamos los datos de entrenamiento para entrenar nuestro modelo, ¡pero debemos tener cuidado de no sobre ajustarlo. El sobreajuste se produce cuando nuestro modelo está tan centrado en los datos de entrenamiento que no funciona bien con datos nuevos. Es como estudiar para un examen, pero sólo memorizar las respuestas a las preguntas de práctica.

4.2.7 Evaluación del modelo

hora es el momento de ver si nuestro modelo tiene lo que se necesita. Evaluamos su desempeño utilizando varias métricas, como exactitud, precisión y recuperación. Estas métricas nos ayudan a determinar si nuestro modelo está listo para las grandes ligas.

4.2.8 Ajuste del modelo

El tuning de modelo es como darle a nuestro modelo un estilista personal. Probamos diferentes combinaciones de parámetros y elegimos la que mejores resultados nos da. Por ejemplo, consideremos un escenario simple en el que queremos ajustar los parámetros de un modelo de árbol de decisión. Podríamos usar GridSearchCV de la biblioteca scikit-learn para realizar una

búsqueda en cuadrícula sobre las diferentes combinaciones de parámetros y encontrar la mejor combinación.

4.2.9 Implementación del modelo

Piense en la implementación del modelo como enviar su modelo de aprendizaje automático al mundo por primera vez. Es un gran paso, pero con la preparación adecuada, su modelo estará listo para abordar problemas del mundo real y hacer predicciones precisas.

4.3 Explicación de los algoritmos utilizados

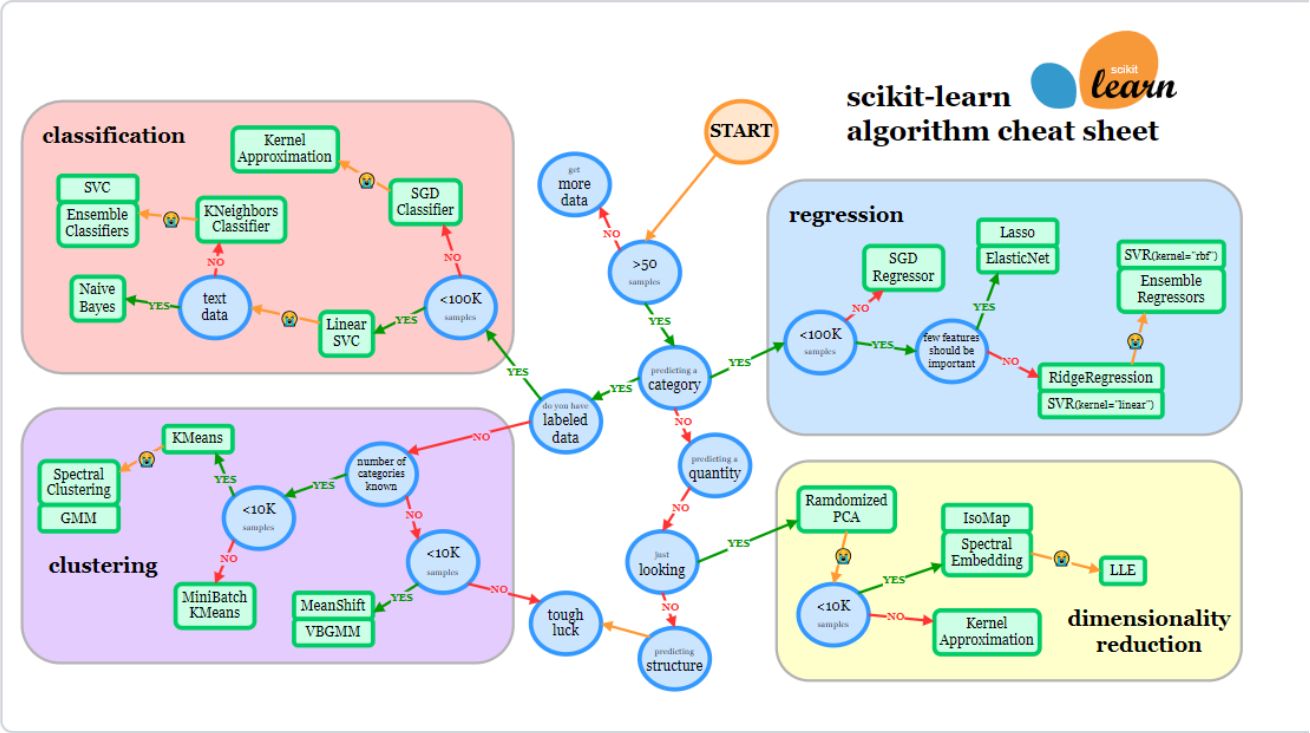
A continuación, se explica los distintos tipos de algoritmos utilizados.

4.3.1 Algoritmos de clasificación

Se planteo la solución de categorizar la salida de respuesta con lo cual los algoritmos optados fueron del tipo de CLASIFICACION, donde la variable de respuesta tendrá la salida de lesiones Graves, Moderadas y Leves. El modelo según el entrenamiento es el SUPERVISADO. En la Figura 28 se detalla el modelo de los algoritmos que se pueden utilizar

Figura 28

Diagrama de flujo de estimadores



Algoritmos utilizados

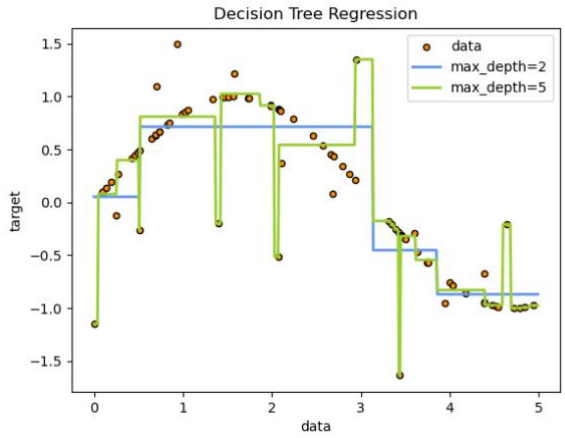
- Deciscion Tree:
- Random Forest:
- XGBoost:

4.3.2 Decision Tree

Son un método de aprendizaje supervisado no paramétrico que se utiliza para clasificación y regresión. El objetivo es crear un modelo que prediga el valor de una variable objetivo aprendiendo reglas de decisión simples inferidas de las características de los datos. Un árbol puede verse como una aproximación constante por partes tal como se vé en la figura 29.

Figura 29

Regresion decisión tree



https://scikit-learn.org/stable/modules/tree.html

Algunas ventajas de los árboles de decisión son:

- Sencillo de entender e interpretar. Se pueden visualizar árboles.
- Requiere poca preparación de datos. Otras técnicas a menudo requieren la normalización de datos, es necesario crear variables ficticias y eliminar valores en blanco. Algunas combinaciones de árboles y algoritmos admiten valores faltantes.
- El costo de usar el árbol (es decir, predecir datos) es logarítmico en el número de puntos de datos utilizados para entrenar el árbol.
- Capaz de manejar datos tanto numéricos como categóricos. Sin embargo, la implementación de scikit-learn no admite variables categóricas por ahora. Otras técnicas suelen estar especializadas en analizar conjuntos de datos que tienen un solo tipo de variable. Consulte algoritmos para obtener más información.

- Capaz de manejar problemas de múltiples salidas.
- Utiliza un modelo de caja blanca. Si una situación dada es observable en un modelo, la explicación de la condición se explica fácilmente mediante la lógica booleana. Por el contrario, en un modelo de caja negra (por ejemplo, en una red neuronal artificial), los resultados pueden ser más difíciles de interpretar.
- Es posible validar un modelo mediante pruebas estadísticas. Eso permite dar cuenta de la confiabilidad del modelo.
- Funciona bien incluso si sus supuestos son de alguna manera violados por el modelo real a partir del cual se generaron los datos.

Las desventajas de los árboles de decisión incluyen:

- Los estudiantes de árboles de decisión pueden crear árboles demasiado complejos que no generalizan bien los datos. A esto se le llama sobreajuste. Para evitar este problema son necesarios mecanismos como la poda, establecer el número mínimo de muestras requeridas en un nodo de hoja o establecer la profundidad máxima del árbol.
- Los árboles de decisión pueden ser inestables porque pequeñas variaciones en los datos pueden generar un árbol completamente diferente. Este problema se mitiga mediante el uso de árboles de decisión dentro de un conjunto.
- Las predicciones de los árboles de decisión no son suaves ni continuas, sino aproximaciones constantes por partes, como se ve en la figura anterior. Por lo tanto, no son buenos para la extrapolación.

- Se sabe que el problema de aprender un árbol de decisión óptimo es NP-completo en varios aspectos de optimización e incluso para conceptos simples. En consecuencia, los algoritmos prácticos de aprendizaje de árboles de decisiones se basan en algoritmos heurísticos como el algoritmo codicioso donde se toman decisiones localmente óptimas en cada nodo. Dichos algoritmos no pueden garantizar que se devuelva el árbol de decisión globalmente óptimo. Esto se puede mitigar entrenando varios árboles en un alumno conjunto, donde las características y muestras se muestrean aleatoriamente con reemplazo.
- Hay conceptos que son difíciles de aprender porque los árboles de decisión no los expresan fácilmente, como por ejemplo los problemas XOR, de paridad o de multiplexor.
- Los estudiantes de árboles de decisión crean árboles sesgados si algunas clases dominan. Por lo tanto, se recomienda equilibrar el conjunto de datos antes de ajustarlo al árbol de decisión.

4.3.2.1 Conceptos y ejemplos de Árboles de decisión

Un árbol de decisiones es un algoritmo de aprendizaje supervisado no paramétrico. Tiene una estructura jerárquica en forma de árbol, que consta de un nodo raíz, ramas, nodos internos y nodos hoja.

Los árboles de decisión son la base de muchos algoritmos clásicos de aprendizaje automático, como los bosques aleatorios, el bagging y los árboles de decisión potenciados. Su idea era representar los datos como un árbol en el que cada nodo interno denota una prueba sobre un

atributo (básicamente, una condición), cada rama representa un resultado de la prueba y cada nodo de hoja (nodo terminal) contiene una etiqueta de clase.

4.3.2.2Tipos de árboles de decisión

- CART (árboles de clasificación y regresión) → utiliza el índice de Gini (clasificación) como métrica.
- ID3 (Dicotómico iterativo 3) → utiliza la función de entropía y la ganancia de información como métricas.

4.3.2.3 Antes de aprender más sobre los árboles de decisión, familiaricémonos con algunas de las terminologías.

Nodos raíz: es el nodo presente al comienzo de un árbol de decisión, a partir de este nodo la población comienza a dividirse de acuerdo a varias características.

Nodos de decisión: los nodos que obtenemos después de dividir los nodos raíz se denominan Nodos de decisión.

Nodos hoja: los nodos en los que no es posible realizar más divisiones se denominan nodos hoja o nodos terminales.

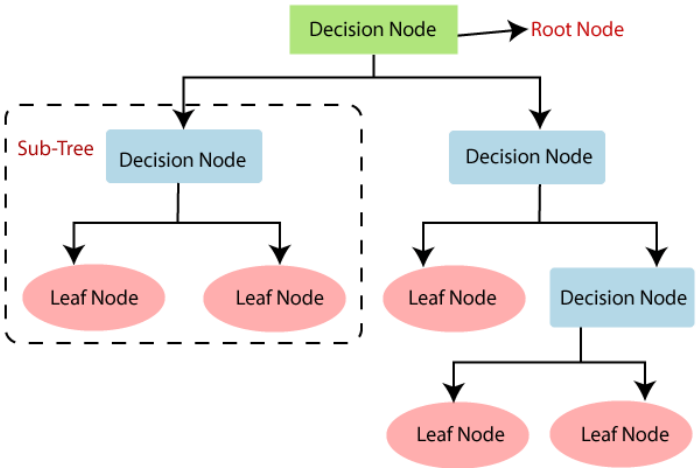
Rama / Subárbol: así como una pequeña porción de un gráfico se llama subgráfico, de manera similar una subsección de este árbol de decisión se llama subárbol.

La poda no es más que cortar algunos nodos para evitar el sobreajuste.

En la figura 30 se observa la estructura del árbol de decisión.

Figura 30

Estructura de un árbol de decisión



4.3.2.4 ¿Por qué utilizar árboles de decisión?

Existen varios algoritmos en el aprendizaje automático. A continuación, se indican las dos razones para utilizar el árbol de decisiones:

- Los árboles de decisión generalmente imitan la capacidad de pensamiento humano al tomar una decisión, por lo que son fáciles de entender.
- La lógica detrás del árbol de decisiones se puede entender fácilmente porque muestra una estructura similar a un árbol.

4.3.2.5 Ejemplo de un árbol de decisión

Comprendamos los árboles de decisión con la ayuda de un ejemplo.

El árbol de decisión construye modelos de clasificación o regresión en forma de una estructura de árbol. Divide un conjunto de datos en subconjuntos cada vez más pequeños mientras que al

mismo tiempo se desarrolla de forma incremental un árbol de decisión asociado. El resultado final es un árbol con nodos de decisión y nodos de hoja. Un nodo de decisión (por ejemplo, Outlook) tiene dos o más ramas (por ejemplo, Sunny, Overcast y Rainy). El nodo de hoja (por ejemplo, Play) representa una clasificación o decisión. El nodo de decisión superior en un árbol que corresponde al mejor predictor se denomina nodo raíz. Los árboles de decisión pueden manejar datos tanto categóricos como numéricos. En la figura 31 se observa el ejemplo mencionado.

Figura 31

Ejemplo de árbol de decisión



Los árboles de decisión están al revés, lo que significa que la raíz está en la parte superior y luego esta raíz se divide en varios nodos. Los árboles de decisión no son más que un montón de declaraciones if-else en términos sencillos. Comprueba si la condición es verdadera y, si lo es, pasa al siguiente nodo adjunto a esa decisión.

4.3.3 Random Forest

En este algoritmo, cada árbol del conjunto se construye a partir de una muestra extraída con reemplazo (es decir, una muestra de arranque) del conjunto de entrenamiento.

Además, al dividir cada nodo durante la construcción de un árbol, la mejor división se encuentra mediante una búsqueda exhaustiva de los valores de las características de todas las características de entrada o de un subconjunto aleatorio de tamaño max_features. (Consulte las pautas de ajuste de parámetros para obtener más detalles).

El propósito de estas dos fuentes de aleatoriedad es disminuir la varianza del estimador forestal. De hecho, los árboles de decisión individuales suelen presentar una gran variación y tienden a sobre ajustarse. La aleatoriedad inyectada en los bosques produce árboles de decisión con errores de predicción algo desacoplados. Al tomar un promedio de esas predicciones, algunos errores pueden anularse. Los bosques aleatorios logran una variación reducida al combinar diversos árboles, a veces a costa de un ligero aumento en el sesgo. En la práctica, la reducción de la varianza suele ser significativa, por lo que se obtiene un mejor modelo en general.

A diferencia de la publicación original [B2001], la implementación de scikit-learn combina clasificadores promediando su predicción probabilística, en lugar de permitir que cada clasificador vote por una sola clase.

Una alternativa competitiva a los bosques aleatorios son los modelos de aumento de gradiente basado en histogramas (HGBT):

- Construcción de árboles: los bosques aleatorios generalmente dependen de árboles profundos (que se ajustan individualmente) que utilizan muchos recursos computacionales, ya que requieren varias divisiones y evaluaciones de divisiones

candidatas. Los modelos de impulso construyen árboles poco profundos (que no se ajustan individualmente) que son más rápidos de ajustar y predecir.

- Impulso secuencial: En HGBT, los árboles de decisión se construyen de forma secuencial, donde cada árbol es entrenado para corregir los errores cometidos por los anteriores. Esto les permite mejorar iterativamente el rendimiento del modelo utilizando relativamente pocos árboles. Por el contrario, los bosques aleatorios utilizan una votación mayoritaria para predecir el resultado, lo que puede requerir una mayor cantidad de árboles para lograr el mismo nivel de precisión.
- Agrupación eficiente: HGBT utiliza un algoritmo de agrupación eficiente que puede manejar grandes conjuntos de datos con una gran cantidad de funciones. El algoritmo de agrupación puede preprocesar los datos para acelerar la construcción posterior del árbol (consulte Por qué es más rápido). Por el contrario, la implementación scikit-learn de bosques aleatorios no utiliza agrupación y se basa en una división exacta, lo que puede resultar costoso desde el punto de vista computacional.

En general, el costo computacional de HGBT versus RF depende de las características específicas del conjunto de datos y la tarea de modelado. Es una buena idea probar ambos modelos y comparar su rendimiento y eficiencia computacional en su problema específico para determinar qué modelo se adapta mejor.

4.3.3.1 Cómo funciona el algoritmo de bosque aleatorio: Comprender los árboles de decisión

Los árboles de decisión son los componentes básicos de un algoritmo de bosque aleatorio. Un árbol de decisión es una técnica de apoyo a la toma de decisiones que forma una estructura

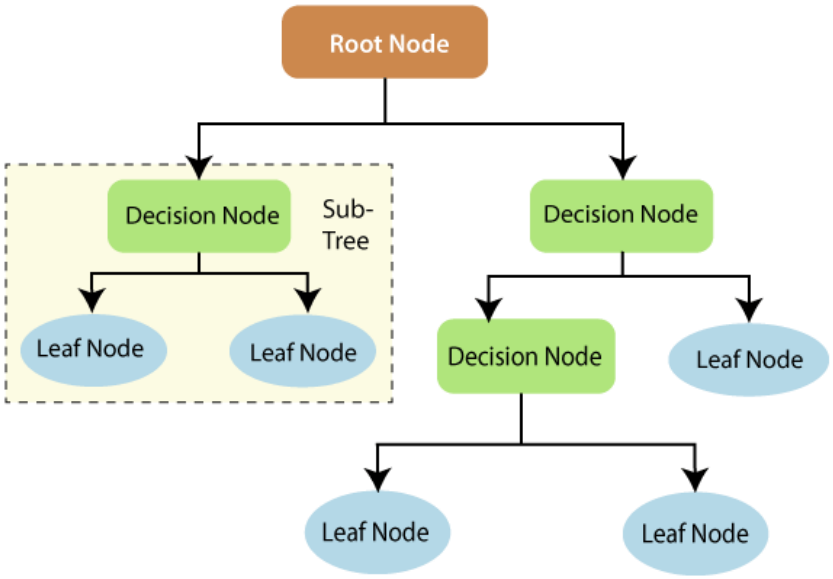
similar a un árbol. Una descripción general de los árboles de decisión nos ayudará a comprender cómo funcionan los algoritmos de bosque aleatorio.

Un árbol de decisiones consta de tres componentes: nodos de decisión, nodos de hoja y un nodo raíz. Un algoritmo de árbol de decisiones divide un conjunto de datos de entrenamiento en ramas, que a su vez se segregan en otras ramas. Esta secuencia continúa hasta que se alcanza un nodo de hoja. El nodo de hoja no se puede segregar más.

Los nodos del árbol de decisiones representan atributos que se utilizan para predecir el resultado. Los nodos de decisión proporcionan un vínculo a las hojas. El siguiente diagrama, la figura 32 muestra los tres tipos de nodos de un árbol de decisiones.

Figura 32

Estructura de Random Forest



La teoría de la información puede brindar más información sobre cómo funcionan los árboles de decisión. La entropía y la ganancia de información son los componentes básicos de los árboles de

decisión. Una descripción general de estos conceptos fundamentales mejorará nuestra comprensión de cómo se construyen los árboles de decisión.

La entropía es una métrica para calcular la incertidumbre. La ganancia de información es una medida de cómo se reduce la incertidumbre en la variable objetivo, dado un conjunto de variables independientes.

El concepto de ganancia de información implica el uso de variables independientes (características) para obtener información sobre una variable objetivo (clase). La entropía de la variable objetivo (Y) y la entropía condicional de Y (dada X) se utilizan para estimar la ganancia de información. En este caso, la entropía condicional se resta de la entropía de Y.

La ganancia de información se utiliza en el entrenamiento de árboles de decisión. Ayuda a reducir la incertidumbre en estos árboles. Una ganancia de información alta significa que se ha eliminado un alto grado de incertidumbre (entropía de información). La entropía y la ganancia de información son importantes en la división de ramas, que es una actividad importante en la construcción de árboles de decisión.

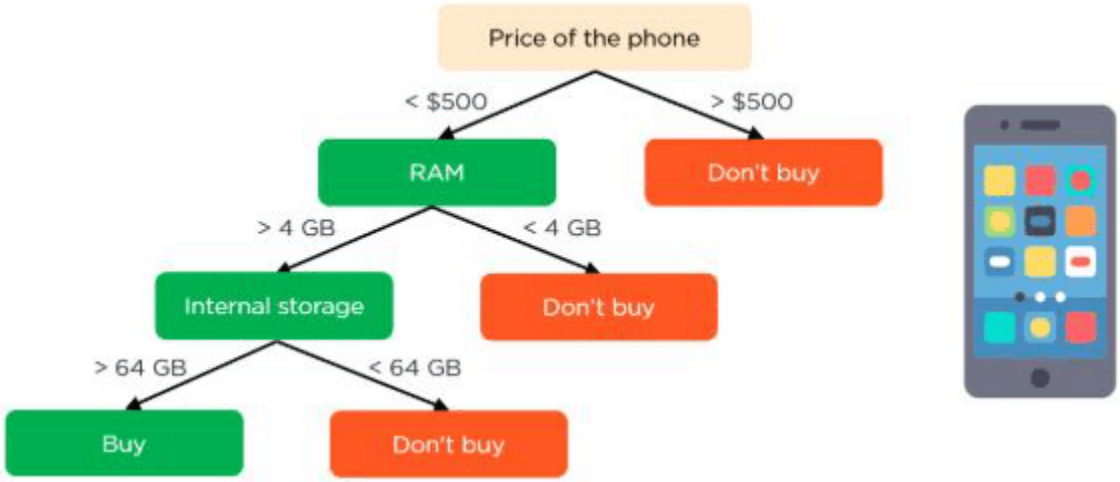
Tomemos un ejemplo sencillo de cómo funciona un árbol de decisiones como se observa en la figura 33. Supongamos que queremos predecir si un cliente comprará o no un teléfono móvil. Las características del teléfono forman la base de su decisión. Este análisis se puede presentar en un diagrama de árbol de decisiones.

El nodo raíz y los nodos de decisión representan las características del teléfono mencionadas anteriormente. El nodo hoja representa el resultado final, ya sea comprar o no comprar . Las características principales que determinan la elección incluyen el precio, el almacenamiento

interno y la memoria de acceso aleatorio (RAM). El árbol de decisiones se verá de la siguiente manera.

Figura 33

Ejemplo de Random Forest



4.3.3.2 Clasificación en bosques aleatorios

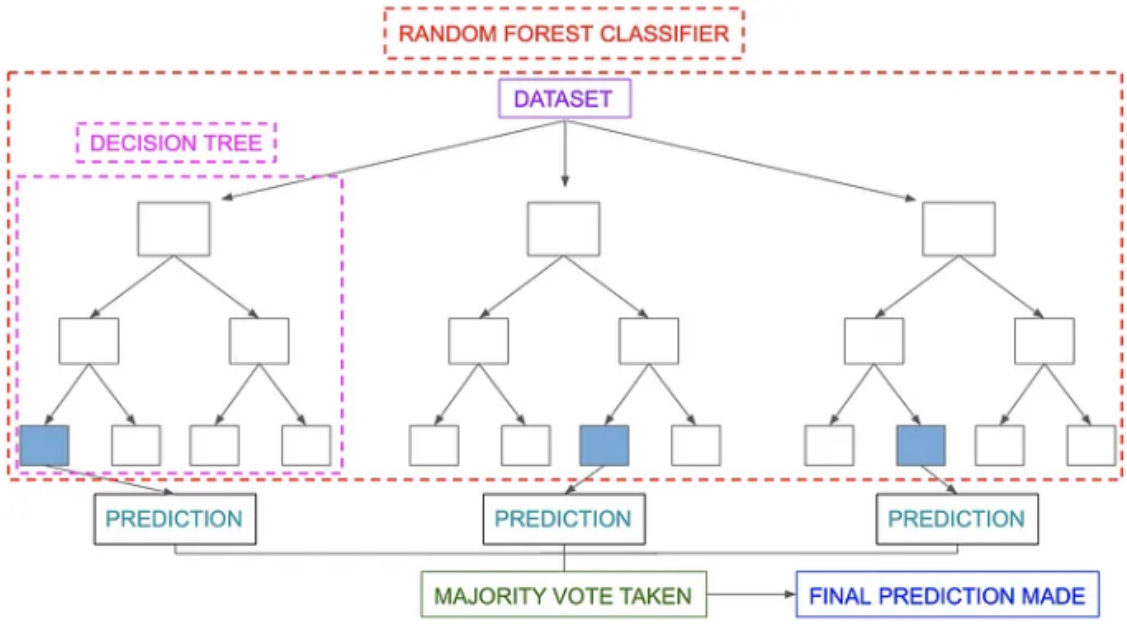
La clasificación en bosques aleatorios emplea una metodología de conjunto para obtener el resultado. Los datos de entrenamiento se utilizan para entrenar varios árboles de decisión. Este conjunto de datos consta de observaciones y características que se seleccionarán aleatoriamente durante la división de nodos.

Un sistema de selva tropical depende de varios árboles de decisión. Cada árbol de decisión consta de nodos de decisión, nodos de hoja y un nodo raíz. El nodo de hoja de cada árbol es el resultado final producido por ese árbol de decisión específico. La selección del resultado final sigue el sistema de votación por mayoría. En este caso, el resultado elegido por la mayoría de los

árboles de decisión se convierte en el resultado final del sistema de selva tropical. El diagrama de la figura 34 se muestra un clasificador aleatorio simple de selva.

Figura 34

Clasificación de Random Forest



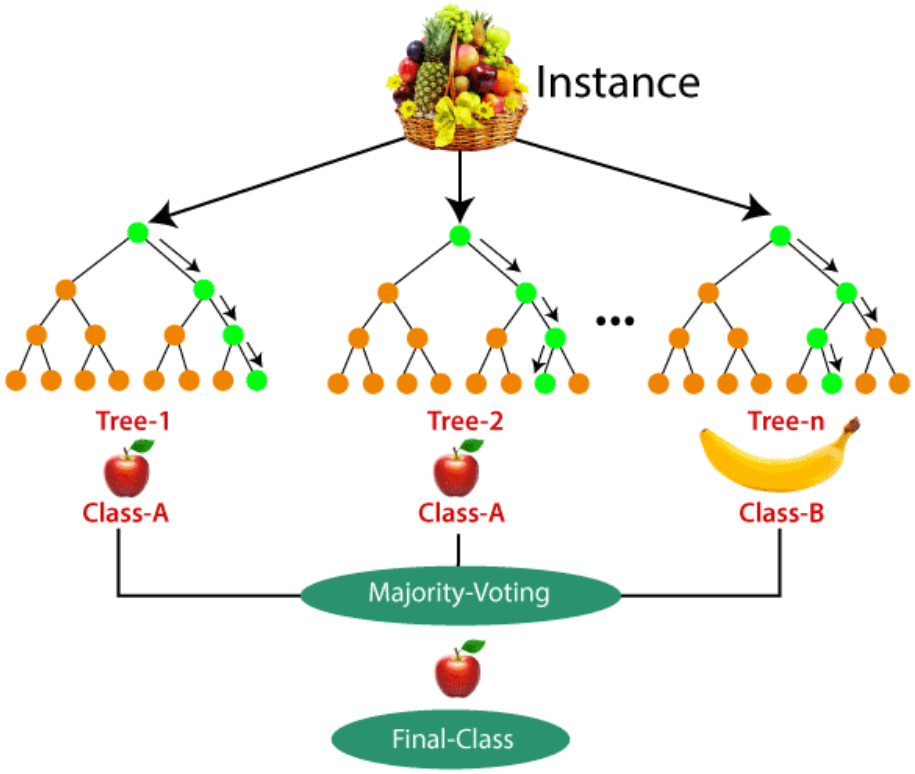
Tomemos como ejemplo un conjunto de datos de entrenamiento que consta de varias frutas, como plátanos, manzanas, piñas y mangos. El clasificador de bosque aleatorio divide este conjunto de datos en subconjuntos. Estos subconjuntos se asignan a cada árbol de decisión en el sistema de bosque aleatorio. Cada árbol de decisión produce su salida específica. Por ejemplo, la predicción para los árboles 1 y 2 es manzana.

Otro árbol de decisión (n) ha predicho que el resultado será un plátano. El clasificador de bosque aleatorio recopila la mayoría de votos para proporcionar la predicción final. La mayoría de los

árboles de decisión han elegido manzana como su predicción. Esto hace que el clasificador elija manzana como la predicción final tal como se muestra en la figura 35.

Figura 35

Ejemplo de clasificación de Random Forest



4.3.4 XGBoost

XGBoost es una biblioteca de software de código abierto que implementa algoritmos de aprendizaje automático de aumento de gradiente distribuido optimizados bajo el marco de aumento de gradiente.

4.3.4.1 ¿Qué es XGBoost?

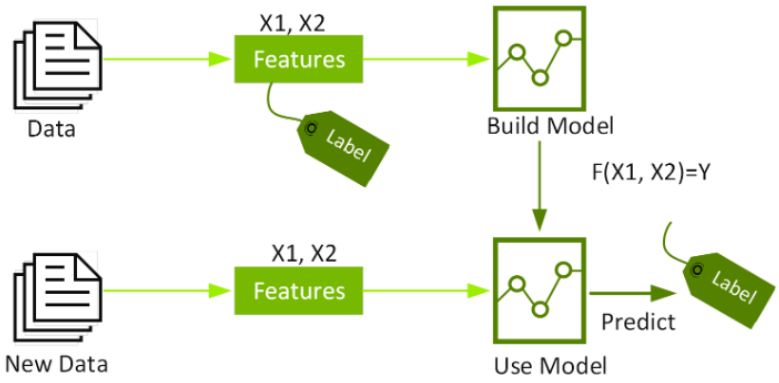
XGBoost, que significa Extreme Gradient Boosting, es una biblioteca de aprendizaje automático de árbol de decisión impulsado por gradiente (GBDT), escalable y distribuida. Proporciona impulso de árboles paralelos y es la biblioteca líder de aprendizaje automático para problemas de regresión, clasificación y ranking.

Para comprender XGBoost, es vital comprender primero los conceptos y algoritmos de aprendizaje automático en los que se basa XGBoost: aprendizaje automático supervisado, árboles de decisión, aprendizaje conjunto y aumento de gradiente.

El aprendizaje automático supervisado utiliza algoritmos para entrenar un modelo para encontrar patrones en un conjunto de datos con etiquetas y características y luego usa el modelo entrenado para predecir las etiquetas en las características de un nuevo conjunto de datos tal como se muestra en la figura 36.

Figura 36

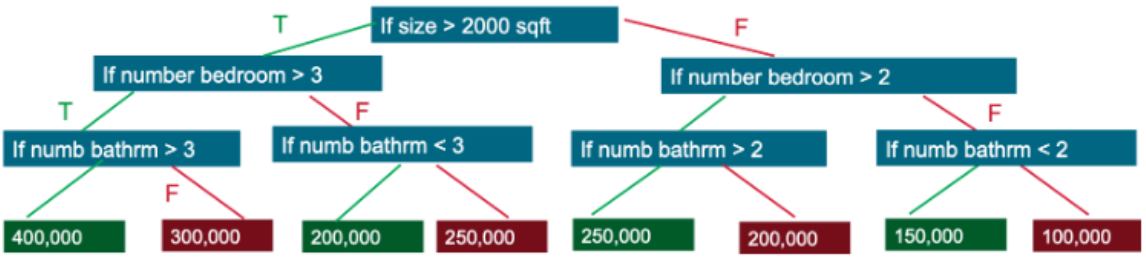
XGBoost Modelo ejemplo



Los árboles de decisión crean un modelo que predice la etiqueta evaluando un árbol de preguntas de características verdaderas/falsas si-entonces-si no y estimando el número mínimo de preguntas necesarias para evaluar la probabilidad de tomar una decisión correcta. Los árboles de decisión se pueden utilizar para clasificación para predecir una categoría o regresión para predecir un valor numérico continuo. En el ejemplo sencillo siguiente, se utiliza un árbol de decisión para estimar el precio de una casa (la etiqueta) en función del tamaño y la cantidad de dormitorios (las características) tal como muestra la figura 37.

Figura 37

XGBoost Arbol

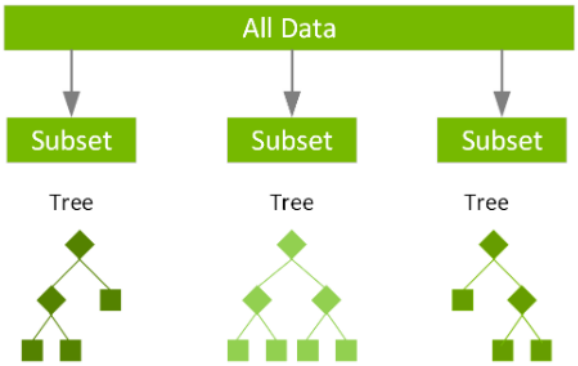


Un árbol de decisión de impulso de gradiente (GBDT) es un algoritmo de aprendizaje conjunto de árboles de decisión similar al bosque aleatorio, para clasificación y regresión. Los algoritmos de aprendizaje conjunto combinan múltiples algoritmos de aprendizaje automático para obtener un mejor modelo.

Tanto el bosque aleatorio como el GBDT construyen un modelo que consta de múltiples árboles de decisión. La diferencia está en cómo se construyen y combinan los árboles tal como muestra la figura 38.

Figura 38

Modelo de combinación de arboles



El bosque aleatorio utiliza una técnica llamada embolsado para construir árboles de decisión completos en paralelo a partir de muestras de arranque aleatorias del conjunto de datos. La predicción final es un promedio de todas las predicciones del árbol de decisión.

El término "impulso de gradiente" proviene de la idea de "impulsar" o mejorar un único modelo débil combinándolo con otros modelos débiles para generar un modelo colectivamente fuerte. El aumento de gradiente es una extensión del aumento en el que el proceso de generación aditiva de modelos débiles se formaliza como un algoritmo de descenso de gradiente sobre una función objetivo. El aumento de gradiente establece resultados específicos para el siguiente modelo en un

esfuerzo por minimizar los errores. Los resultados previstos para cada caso se basan en el gradiente del error (de ahí el nombre de aumento de gradiente) con respecto a la predicción.

Los GBDT entrenan de forma iterativa un conjunto de árboles de decisión poco profundos, y cada iteración utiliza los residuos de error del modelo anterior para ajustarse al siguiente modelo. La predicción final es una suma ponderada de todas las predicciones del árbol. El "embolsado" aleatorio del bosque minimiza la variación y el sobreajuste, mientras que el "impulso" de GBDT minimiza el sesgo y el desajuste.

XGBoost es una implementación escalable y altamente precisa de aumento de gradiente que supera los límites de la potencia informática para algoritmos de árbol impulsados, y se creó en gran medida para potenciar el rendimiento del modelo de aprendizaje automático y la velocidad computacional. Con XGBoost, los árboles se construyen en paralelo, en lugar de secuencialmente como GBDT. Sigue una estrategia por niveles, escaneando los valores de gradiente y utilizando estas sumas parciales para evaluar la calidad de las divisiones en cada división posible en el conjunto de entrenamiento.

4.3.4.2 ¿Por qué XGBoost?

XGBoost ganó un gran favor en los últimos años como resultado de ayudar a individuos y equipos a ganar prácticamente todas las competencias de datos estructurados de Kaggle. En estas competencias, empresas e investigadores publican datos, después de lo cual los estadísticos y mineros de datos compiten para producir los mejores modelos para predecir y describir los datos.

Inicialmente se crearon implementaciones de XGBoost en Python y R. Debido a su popularidad, hoy XGBoost tiene implementaciones de paquetes para Java, Scala, Julia, Perl y otros lenguajes.

Estas implementaciones han abierto la biblioteca XGBoost a aún más desarrolladores y han mejorado su atractivo en toda la comunidad de Kaggle.

XGBoost se ha integrado con una amplia variedad de otras herramientas y paquetes como scikit-learn para entusiastas de Python y caret para usuarios de R. Además, XGBoost está integrado con marcos de procesamiento distribuido como Apache Spark y Dask.

En 2019, XGBoost fue nombrado entre los ganadores del codiciado premio Tecnología del Año de InfoWorld.

4.3.4.3 Beneficios y atributos de XGBoost

La lista de beneficios y atributos de XGBoost es extensa e incluye lo siguiente:

- Una lista grande y creciente de científicos de datos a nivel mundial que contribuyen activamente al desarrollo de código abierto de XGBoost.
- Uso en una amplia gama de aplicaciones, incluida la resolución de problemas de regresión, clasificación, ranking y desafíos de predicción definidos por el usuario.
- Una biblioteca que es altamente portátil y actualmente se ejecuta en plataformas OS X, Windows y Linux.
- Integración en la nube que admite AWS, Azure, clústeres Yarn y otros ecosistemas
- Uso de producción activa en múltiples organizaciones en varias áreas del mercado vertical.
- Una biblioteca construida desde cero para ser eficiente, flexible y portátil

4.4 Justificación de los algoritmos

Este capítulo explica el porqué de los algoritmos utilizados, así como también la aplicación de los mismos en el ámbito de la prevención de lesiones en distintos deportes.

Al trabajar con conjuntos de datos pequeños en el aprendizaje automático, es fundamental elegir algoritmos que sean robustos y menos propensos al sobreajuste. Por otro lado, a raíz de estudios realizados referidos a aplicaciones del aprendizaje automático en la predicción de lesiones deportivas, han demostrado la utilización de varios algoritmos, entre ellos los que se describen a continuación:

4.4.1 Naive Bayes

Este algoritmo es simple y funciona bien con conjuntos de datos pequeños. Supone que las características son condicionalmente independientes dada la etiqueta de clase, lo que a veces puede ser una limitación, pero a menudo funciona sorprendentemente bien.

4.4.2 Decision Trees

Los árboles de decisión son otra buena opción para conjuntos de datos pequeños. Son fáciles de interpretar y pueden manejar relaciones no lineales entre características y variables objetivo.

Las evoluciones modernas del algoritmo clásico del árbol de decisiones se han aplicado ampliamente en los últimos años. En 2018, Connaboy et al. utilizaron árboles de decisión construidos con detección automática de interacción de chi-cuadrado (CHAID) para analizar los factores que contribuyen a las lesiones de las extremidades inferiores en el personal militar. Utilizando su modelo, los autores identificaron varios factores que conducen a un mayor riesgo de lesiones durante un período de 365 días [Using machine learning to predict lower-extremity injury in US special forces. Connaboy C, Eagle SR, Johnson CD, Flanagan SD, Mi QI, Nindl

BC. Med Sci Sports Exerc. 2019;51:1073–1079]. Utilizando un árbol de clasificación y regresión (CART), Mendonça et al. investigaron las asociaciones entre varios factores de riesgo y la tendinopatía rotuliana en jugadores de voleibol y baloncesto [Association of hip and foot factors with patellar tendinopathy (jumper's knee) in athletes. Mendonça LD, Ocarino JM, Bittencourt NF, Macedo LG, Fonseca ST. J Orthop Sports Phys Ther. 2018;48:676–684]. Un artículo de 2021 de Kolodziej et al. aplicó un árbol de decisión CART para predecir las lesiones del fútbol juvenil, logrando una sensibilidad de 0,73 y una especificidad de 0,91 [Identification of neuromuscular performance parameters as risk factors of non-contact injuries in male elite youth soccer players: a preliminary study on 62 players with 25 non-contact injuries. Kolodziej M, Nolte K, Schmidt M, Alt T, Jaitner T. Front Sports Act Living. 2021;3:615330]. Otro artículo de 2021 de Ruiz-Pérez et al. intentó reproducir un modelo de 2020 de Rommers et al., que utilizó datos de campo recopilados a través de GPS. Si bien compararon favorablemente los árboles de decisión C4.5 con varios enfoques de modelado, incluidos KNN, SVM y ADTree, no utilizaron el mismo algoritmo que Rommers et al. y no lograron un rendimiento comparable (área bajo la curva ROC, o AUC, 0,767 frente a 0,850) [A field-based approach to determine soft tissue injury risk in elite futsal using novel machine learning techniques. Ruiz-Pérez I, López-Valenciano A, Hernández-Sánchez S, Puerta-Callejón JM, De Ste Croix M, Sainz de Baranda P, Ayala F. Front Psychol. 2021;12:610210.; A machine learning approach to assess injury risk in elite youth football players. Rommers N, Rössler R, Verhagen E, et al. Med Sci Sports Exerc. 2020;52:1745–1751]. Contrariamente a estos resultados relativamente prometedores, Rossi et al. descubrieron que los árboles de decisión, aunque superaban a los algoritmos de comparación, no podían lograr una precisión mayor al 50% al pronosticar lesiones de fútbol [Effective injury forecasting in soccer with GPS training data and machine learning. Rossi A, Pappalardo L,

Cintia P, Iaia FM, Fernàndez J, Medina D. PLoS One. 2018;13:0]. Los árboles de decisión, sin duda, tienen un lugar en la predicción de lesiones deportivas, aunque su rendimiento varía con los datos y la estructura del modelo. Además, pueden carecer de generalización y sobreajustarse durante el entrenamiento, lo que limita su precisión [Random forests. Breiman L. Mach Learn. 2001;45:5–32.].

4.4.3 Random Forests

Este método de conjunto crea múltiples árboles de decisión y combina sus predicciones, lo que puede mejorar la precisión y la solidez, especialmente con conjuntos de datos más pequeños.

Los modelos de bosque aleatorio se han aplicado a la predicción de lesiones con un éxito mixto. En un estudio de lesiones dentales relacionadas con el deporte en niños, los algoritmos de bosque aleatorio tuvieron una precisión de predicción ligeramente mayor en comparación con los métodos de regresión tradicionales [Random forest algorithm to identify factors associated with sports-related dental injuries in 6 to 13-year-old athlete children in Hamadan, Iran-2018 - a cross-sectional study. Farhadian M, Torkaman S, Mojarad F. BMC Sports Sci Med Rehabil. 2020;12:69]. Un artículo de 2020 buscó abordar la inconsistencia en el rendimiento predictivo identificando los factores de riesgo clave antes del entrenamiento del modelo. Pudieron lograr un AUC de 0,79 [Machine learning to predict lower extremity musculoskeletal injury risk in student athletes. Henriquez M, Sumner J, Faherty M, Sell T, Bent B. Front Sports Act Living. 2020;2:576655.]. En un artículo de 2022, se construyó un modelo de bosque aleatorio y logró un rendimiento similar con un AUC de 0,72 [Detecting injury risk factors with algorithmic models in elite women's pathway cricket. Goggins L, Warren A, Osguthorpe D, et al. Int J Sports Med. 2022;43:344–349]. En una investigación de nadadores paralímpicos que clasificaron a los participantes con y sin lesión cerebral para determinar la elegibilidad, los bosques aleatorios

clasificaron con éxito al 96% de los 51 participantes [Classifying motor coordination impairment in Para swimmers with brain injury. Hogarth L, Payton C, Nicholson V, et al. J Sci Med Sport. 2019;22:526–531]. Contrariamente a estos estudios, un artículo de 2021 encontró que los bosques aleatorios predijeron lesiones de tobillo en atletas jóvenes con un rendimiento similar a una regresión logística (ROC 0,63 frente a 0,65, respectivamente) [New machine learning approach for detection of injury risk factors in young team sport athletes. Jauhiainen S, Kauppi JP, Leppänen M, et al. Int J Sports Med. 2021;42:175–182]. Con una aplicación adecuada y una selección de características imparcial, los modelos de bosque aleatorio pueden ajustarse para superar a los métodos de clasificación existentes, aunque son sensibles a las variaciones en los conjuntos de datos.

4.4.4 Support Vector Machines (SVMs)

Las SVM pueden funcionar bien con conjuntos de datos de tamaño pequeño a mediano. Encuentran el hiperplano óptimo que maximiza el margen entre clases, lo que ayuda a generalizar bien.

Para aplicaciones específicas de deportes, los SVM se han entrenado utilizando métricas modificables como carga de entrenamiento, técnicas de rendimiento, evaluaciones psicológicas y neuromusculares, y métricas no modificables como mediciones antropométricas, historial de lesiones previas y marcadores genéticos para predecir con precisión lesiones futuras [Machine learning methods in sport injury prediction and prevention: a systematic review. Van Eetvelde H, Mendonça LD, Ley C, Seil R, Tischer T. J Exp Orthop. 2021;8:27 ; Genomic prediction of tendinopathy risk in elite team sports. Rodas G, Osaba L, Arteta D, Pruna R, Fernández D, Lucia A. Int J Sports Physiol Perform. 2019;15:489–495.]. La identificación de factores de riesgo de lesiones como estos permite a los entrenadores y al personal médico modificar las cargas de

entrenamiento, los regímenes y las técnicas para prevenir potencialmente lesiones futuras [Current approaches to the use of artificial intelligence for injury risk assessment and performance prediction in team sports: a systematic review. Claudino JG, Capanema DO, de Souza TV, Serrão JC, Machado Pereira AC, Nassis GP. Sports Med Open. 2019;5:28.]. Por ejemplo, un artículo de 2018 de Ruddy et al. utilizó varios algoritmos de ML, incluido SVM, para evaluar los factores de riesgo identificados en las lesiones por distensión de los isquiotibiales [Predictive modeling of hamstring strain injuries in elite Australian footballers. Ruddy JD, Shield AJ, Maniar N, et al. Med Sci Sports Exerc. 2018;50:906–914]. En otro artículo de 2018 de Carey et al., que también explora la predicción de lesiones de los isquiotibiales y los factores de riesgo, SVM se benefició sustancialmente del preprocesamiento de datos, aunque en última instancia fue superado por la regresión logística simple [Predictive modelling of training loads and injury in Australian football. Carey DL, Ong K, Whiteley R, Crossley KM, Crow J, Morris ME. Int J Comput Sci Sport. 2018;17:49–66]. Utilizando datos no fisiológicos, un artículo de 2017 que predecía lesiones en el juego en la Major League Soccer descubrió que los SVM eran los más precisos de varios algoritmos probados, incluidos la regresión logística, el perceptrón multicapa y el bosque aleatorio [Using weather and playing surface to predict the occurrence of injury in major league soccer games: a case study. Landset S, Bergeron MF, Khoshgoftaar TM. 2017 IEEE Int Conf Inf Reuse Integr. 2017:366–371]. Sin embargo, en la literatura reciente, incluidos dos artículos de 2021 que comparan la eficacia de los algoritmos de ML, los SVM han demostrado ser menos efectivos que otros algoritmos [Analysis and design of dual-feature fusion neural network for sports injury estimation model. Meng L, Qiao E. Neural Comput Applic. 2023;35:14627–14639; Prediction simulation of sports injury based on embedded system and neural network. Shen H. Microprocess Microsyst. 2021;82:103900]. A

pesar de esto, los SVM aún pueden ser valiosos dada su idoneidad para predecir conjuntos de datos de alta dimensionalidad, especialmente cuando se combinan con otras técnicas, como en un artículo de 2022 de Wang et al. que predice lesiones por triple salto [Evidence-based sports medicine to prevent knee joint injury in triple jump. Wang S, Lyu B. Rev Bras Med Esporte. 2022;28:195–198.].

4.4.5 k-Nearest Neighbors (k-NN)

Este algoritmo es simple y se basa en instancias, lo que significa que no crea un modelo de manera explícita, sino que almacena instancias de datos de entrenamiento. Puede funcionar bien con conjuntos de datos pequeños, especialmente cuando el límite de decisión no es muy complejo.

En medicina deportiva, se pueden conectar sensores especiales como acelerómetros, giroscopios, sensores infrarrojos y magnetómetros a los atletas para recopilar datos. Al utilizar datos recopilados de diferentes partes del cuerpo de los atletas, KNN puede analizar comportamientos de los atletas en eventos deportivos únicos. Con este modelo de reconocimiento, se pueden determinar patrones que predisponen a lesiones, lo que permite una posible prevención de lesiones [Sports injury rehabilitation intervention algorithm based on visual analysis technology. Chen X, Yuan G. Mob Inf Syst. 2021;2021:9993677]. Además de su uso general como algoritmos de comparación, un artículo de 2018 aplicó KNN como parte de un modelo más grande, que incluye tanto K-means como una máquina de vectores de soporte (SVM), para la predicción de lesiones [Athlete-customized injury prediction using training load statistical records and machine learning. Naglah A, Khalifa F, Mahmoud A, et al. 2018 IEEE Int Symp Signal Process Inf Technol. 2018:459–464].

4.4.6 Regularized Regression (e.g., Ridge Regression, Lasso Regression)

Estas técnicas pueden ser eficaces para reducir el sobreajuste cuando la cantidad de características es grande en comparación con la cantidad de muestras, lo que es común en conjuntos de datos pequeños.

4.4.7 Gradient Boosting Machines (e.g., XGBoost, LightGBM)

Los métodos de potenciación de gradiente pueden ser potentes, incluso con conjuntos de datos pequeños, al agregar secuencialmente aprendices débiles (generalmente árboles de decisión) para mejorar el rendimiento predictivo.

Al trabajar con conjuntos de datos pequeños, también es importante considerar técnicas de aumento de datos, selección de características y regularización para mejorar el rendimiento del modelo y evitar el sobreajuste. Además, la validación cruzada es crucial para evaluar la solidez del modelo elegido debido a los datos limitados disponibles.

Más allá del aumento de gradiente estándar, un estudio de 2022 utilizó XGBoost (aumento de gradiente extremo) para predecir lesiones posteriores a conmociones cerebrales en 74 jugadores de fútbol universitario con una precisión del 91,9 % [A predictive paradigm for identifying elevated musculoskeletal injury risks after sport-related concussion. Mansouri M, Roland J, Rahmati M, Sartipi M, Wilkerson G. Sports Orthop Traumatol. 2022;38:66–74]. Rommers et al. en un artículo de 2020 también utilizaron XGBoost, esta vez prediciendo lesiones en 734 jugadores de fútbol juvenil con una precisión y un recuerdo del 84 % y el 83 %, respectivamente. Los autores también pudieron clasificar las lesiones como por uso excesivo o agudas con una precisión y un recuerdo del 82 % [A machine learning approach to assess injury risk in elite youth football players. Rommers N, Rössler R, Verhagen E, et al. Med Sci Sports Exerc.

2020;52:1745–1751.]. Además, una revisión retrospectiva reciente utilizó un modelo XGBoost para explorar la relación entre la biomecánica y las lesiones autoinformadas de los atletas [A retrospective study of foot biomechanics and injury history in varsity football athletes at the U.S. Naval Academy. Windsor J, Jeffries J, Sorensen J, Bach K, Benedek E, Bicher J, Pasquina P. Mil Med. 2022;187:684–689.]. En particular, solo se encontró un artículo reciente que utilizaba AdaBoost, un estudio de 2022 que predecía lesiones en practicantes de CrossFit. Se descubrió que AdaBoost tenía un mejor rendimiento en general que los algoritmos de comparación con un AUC del 77,93 % [Prediction of injuries in CrossFit training: a machine learning perspective. Moustakidis S, Siouras A, Vassis K, Misiris I, Papageorgiou E, Tsaopoulos D. Algorithms. 2022;15:77].

4.4.8 Fortalezas y debilidades de los algoritmos

A continuación, se muestra en la figura 39 una tabla comparativa resumiendo las principales **fortalezas y debilidades** de los algoritmos descriptos.

Figura 39

Tabla comparativa de algoritmos

Summary of findings

Strengths and weaknesses of each algorithm have been presented, along with the number of papers included in this survey. Note that some studies have been counted in more than one category.

Algorithm	Number of studies	Strengths	Weaknesses
K-nearest neighbor	2	Simple to implement, unsupervised.	Sample size and data set size limitations, may be less accurate than other techniques, struggles with high-dimensionality data.
K-means	2	Simple to implement, unsupervised, useful for blind feature extraction and data exploration.	Better suited to initial data exploration than final classification when compared to other algorithms.
Support vector machines	9	Commonly integrated into ensemble models, increasing accuracy, able to handle high-dimensionality data.	Mixed success reported in the literature.
Decision tree	7	Reasonable accuracy combined with transparent decision making.	Struggles with high-dimensionality data.
Random forest	5	More accurate than decision trees while retaining high transparency.	Struggles with high-dimensionality data.
Gradient boosting and AdaBoost	9	Significantly improved accuracy when compared to random forest or decision trees, able to better handle high-dimensionality data.	Less transparent than random forest or decision trees and more complicated to implement.
Convolutional neural networks	6	Increased accuracy, able to handle high-dimensionality data.	Lacks transparency ("black-box"), difficult to implement and computationally expensive, requires a large data set, ideally suited to pose estimation, which has not been applied extensively in the literature.
Long short-term memory-based neural networks	1	Accurate and able to handle high-dimensionality data, excellent for time series data.	Lacks transparency ("black-box"), difficult to implement and computationally expensive, requires a large data set, may not be suited to all data sets.
Deep Gaussian covariance networks	1	Leverages neural networks to train parameters for Gaussian covariance functions.	Lacks strong real world validation in the literature, suffers from the same general drawbacks as other neural networks.
Radial basis function	2	May provide improved accuracy, relatively simple architecture.	Lacks strong real world validation in the literature, suffers from the same general drawbacks as other neural networks
Fuzzy and grey neural networks	2	Potential solution to handling high degrees of uncertainty and variability inherent to sports data.	Lacks strong real world validation in the literature, suffers from the same general drawbacks as other neural networks

4.4.9 Resumen de análisis de los algoritmos

El modelo de vecino más cercano K tiene algunas limitaciones prácticas en cuanto a los tamaños de muestra que puede analizar de manera eficiente. Sin embargo, su simplicidad y versatilidad son claras. La integración de sensores especiales que permiten una recopilación de datos más precisa ha mejorado los modelos de reconocimiento de lesiones de KNN y ha aumentado su capacidad para identificar los factores que contribuyen a las lesiones. La identificación mejorada de las características predictivas de lesiones con la resolución de un atleta individual permite a los entrenadores y al personal médico modificar los métodos de entrenamiento para evitar el riesgo de lesión identificado. Sin embargo, KNN ha sido relegado al papel de algoritmo de comparación en muchos de los artículos analizados en este artículo. Sin embargo, esto no debería disuadir a los futuros investigadores de considerar su uso.

Otro algoritmo simple, K-means, se presta bien para la extracción de características. Según el trabajo reciente en la literatura, K-means se puede utilizar para clasificar datos biocinéticos. Alternativamente, K-means se puede utilizar de manera efectiva para predecir futuros jugadores de alto rendimiento. Sin embargo, se puede encontrar una aplicación más interesante en el preprocesamiento de datos. La agrupación de K-means se puede aplicar a conjuntos de datos en las primeras etapas de la fase de exploración, en lugar de como un algoritmo predictivo final. En cualquier caso, K-means debe considerarse cuando sea posible.

Las máquinas de vectores de soporte se pueden utilizar tanto para predecir la aparición de una lesión como para dilucidar los factores de riesgo que contribuyen a la lesión. Sin embargo, en la literatura reciente, los modelos basados en SVM han tenido un éxito desigual. Aun así, las SVM deben tenerse en cuenta al predecir eventos de lesiones deportivas, especialmente cuando se trabaja con datos de alta dimensionalidad. Cabe destacar que los modelos SVM con mejor rendimiento se construyen como modelos de conjunto, que combinan las ventajas de varios algoritmos.

Los algoritmos de Decision tree también pueden ser adecuados para la toma de decisiones médicas, ya que proporcionan una precisión de clasificación razonable combinada con una representación sencilla del conocimiento recopilado. Más importante aún, proporcionan un proceso de toma de decisiones notablemente transparente, lo que permite una exploración profunda de las características. Y, debido a esta transparencia, el proceso de toma de decisiones puede ser fácilmente validado por un experto, lo que mejora enormemente su utilidad en situaciones que contienen alta incertidumbre.

Los algoritmos de Random Forest aumentan la precisión predictiva en comparación con los árboles de decisión a expensas de una transparencia reducida. Además, pueden tener dificultades cuando los datos contienen alta dimensionalidad, aunque la condensación puede proporcionar una reducción adecuada. Incluso con las limitaciones mencionadas, tanto los árboles de decisión como los modelos de bosque aleatorio han tenido un rendimiento razonablemente bueno en situaciones específicas y se debe considerar su aplicación.

Los algoritmos de Gradient boosting y AdaBoos representan mejoras significativas en las capacidades predictivas con respecto a la regresión clásica, así como a los árboles de decisión en los que se basan. Son más fáciles de implementar y más transparentes que las redes neuronales, a la vez que poseen capacidad para grandes conjuntos de características. Además, son particularmente útiles cuando se aplican en el contexto de la predicción de lesiones, donde la clasificación puede limitarse a una elección binaria. En los casos en los que la transparencia es menos crítica que la precisión predictiva, el aumento de gradiente proporciona un equilibrio entre complejidad y rendimiento.

Si bien gradient boosting ofrece varias ventajas sobre los modelos más simples, las redes neuronales tienden a ser los algoritmos de ML más precisos y poderosos disponibles actualmente. Este rendimiento se produce a costa de una mayor complejidad, tiempo de entrenamiento, requisitos de datos y recursos computacionales. A pesar de estos inconvenientes, los artículos clasifican las CNN, las redes neuronales recurrentes (RNN) y otras arquitecturas de NN de manera favorable frente a los algoritmos de comparación. Sin embargo, existe una falta de validación sólida en el mundo real en gran medida debido a la falta de grandes conjuntos de datos fácilmente disponibles. Los investigadores también están utilizando sensores montados en los jugadores para recopilar datos de series temporales sin procesar. Si bien este es un enfoque

válido para la recopilación de datos, no aprovecha el poderoso potencial de reconocimiento de imágenes y estimación de pose de las CNN y limita el entusiasmo de los jugadores por la recopilación de datos en escenarios del mundo real. Existe una ruta clara para explorar enfoques más novedosos para la recopilación y estructuración de datos, así como para desarrollar estudios sólidos utilizando datos del mundo real. Cualquier arquitectura de modelo dada o combinación de arquitecturas podría aplicarse a cualquier conjunto de datos correctamente ajustado. Este conocimiento por sí solo tiene poco valor práctico; Sin embargo, demuestra la necesidad de disponer de conjuntos más grandes de datos del mundo real para clasificar mejor la utilidad de los algoritmos entre situaciones. Incluso con las limitaciones mencionadas, si los datos y los recursos computacionales están disponibles, las redes neuronales deberían considerarse seriamente.

Para ilustrar una última observación, vale la pena examinar una revisión sistemática reciente de Bullock et al. La revisión en cuestión presentó 30 estudios que aplican el aprendizaje automático a la predicción de lesiones deportivas. En sus criterios de selección, se destacó la inclusión de la regresión logística y de Poisson, ambos enfoques válidos pero anticuados para el análisis predictivo, así como la exclusión de metodologías novedosas para el modelado. De hecho, 22 de los 30 artículos incluyeron regresión logística y 2 de los 8 restantes utilizaron regresión de Poisson [Just how confident can we be in predicting sports injuries? A systematic review of the methodological conduct and performance of existing musculoskeletal injury prediction models in sport. Bullock GS, Mylott J, Hughes T, Nicholson KF, Riley RD, Collins GS. Sports Med. 2022; 52:2469–2482]. Creemos que esto ilustra sucintamente un importante cuello de botella en la aplicación del aprendizaje automático a la medicina deportiva. Una cantidad significativa de estudios de calidad no están haciendo un uso completo de los algoritmos de aprendizaje

automático modernos y poderosos. En cambio, se basan en técnicas de regresión bien estudiadas, pero potencialmente inadecuadas, además de caer presa de algunas otras trampas discutidas anteriormente. La investigación reciente que intenta ir más allá de estos modelos relativamente simples a menudo no produce resultados confiables y generalizables. Además, estos artículos suelen tener un valor limitado para quienes buscan aplicaciones prácticas del aprendizaje automático. A pesar de estos inconvenientes, creemos que no es razonable descartar la utilidad o la aplicabilidad en el mundo real del aprendizaje automático basándose en metodologías decididamente obsoletas.

4.4.10 Conclusión

De todos los algoritmos mencionados se seleccionaron para realizar el experimento los algoritmos de Decision tree, Random Forest y XBoost ya que se observa comportamientos adecuados y precisos teniendo en cuenta su utilización en la prevención de lesiones deportivas con un enfoque más Fisiológico.

Respecto al campo psicológico no se encuentran estudios donde se apliquen algoritmos de ML, más bien se utilizan Redes Bayesianas Naive Bayes junto con la recaudación de información psicológica como por ejemplo a través del El Examen del Estado Actual (PSE) , que es un método de entrevista estructurada que se utiliza en evaluaciones psiquiátricas para recopilar sistemáticamente información sobre el estado mental actual de un individuo. Desarrollado por John Wing y sus colegas, es un componente clave del Programa de Evaluación Clínica en Neuropsiquiatría (SCAN), una herramienta de diagnóstico integral para trastornos psiquiátricos.

Experimento

En este capítulo se verá el experimento que se realizó, tomando como datos de entradas el archivo Excel CSV mencionado y los algoritmos aplicados junto con la precisión y validación de los mismos asi como también las el análisis de las variables de respuesta.

5.1 Set de datos

Se consolido un solo archivo donde se estandarizo y homogeneizó la información. Armado de la estructura de datos, tipo, campos. Luego se guardó el archivo con extensión CSV para poder gestionarlo en Jupyter.

Se solicitaron set de datos de jugadores correspondientes a las selecciones menores de voley Argentino, esta información contiene información respecto a formularios PSE y wellnes , evaluaciones funcionales : DEEP SQUAT, MOVILIDAD HOMBROS , ELEVACION ACTIVA PIERNA, ESTABILIDAD TRONCO PUSH UPS , TEST THOMAS PSOAS. Con respecto a las lesiones se obtuvo información respecto a: Diagnostico, Tipo, Zona, Región.

5.2 Métricas

A continuación, se describen los resultados de las métricas utilizadas para los modelos de los algoritmos.

5.2.1 Resultados de las métricas aplicadas para el algoritmo Deciscion Tree

Los datos de precisión del algoritmo son los siguientes: Accurency del 99% y un desvió standard de 0,01% tal como se ve en la figura 40.

Figura 40

Resultado de accurency y desvío standard para el algoritmo Deciscion Tree

```
Accuracy: 0.99 (+/- 0.01)
```

Por otro lado, se valida el modelo de clasificación con la matriz de confusión y se utilizan las métricas de Precision, Recall y F1 Score para validar desempeño del modelo.

El código es el siguiente como se muestra en la figura 41:

Figura 41

Código para la utilización las métricas de Precision, Recall y F1 Score para validar desempeño del modelo.

```
print('Precision: %.3f' % precision_score(y_test, y_pred,average ='micro' )) #calculo la precision de la prediccion
print('Recall: %.3f' % recall_score(y_test, y_pred,average ='micro')) #calculo la precision de la prediccion
print('F1 Score: %.3f' % f1_score(y_test, y_pred,average ='micro')) #calculo la precision de la prediccion

# Matriz de Confusion
#actual = y_pred=clf.predict(y_test) # realizamos la prediccion con los datos de test
#predicted = y_pred=clf.predict(X_test) # realizamos la prediccion con los datos de test
confusion_matrix = metrics.confusion_matrix(y_test, y_pred)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [0, 1, 2])
cm_display.plot()
plt.show()

#Predict  # Datos de entrada para probar
clf.predict([[120,85,6,26,1,2,3,4,5,6,7,8,9,10,10,10]])
```

Los resultados obtenidos de las métricas de validación del modelo son los lo que muestra la Figura 42:

Figura 42

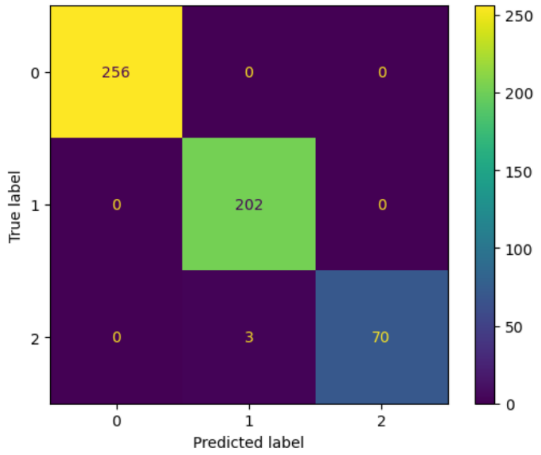
Resultados de las métricas de validación del modelo de Precision, Recall y F1 Score para validar desempeño del modelo.

```
Precision: 0.994
Recall: 0.994
F1 Score: 0.994
```

La matriz de confusión arrojó los siguientes datos que muestra el grafico de figura 43:

Figura 43

Resultado de la matriz de confusión aplicada al modelo



Finalmente se valida con las mismas métricas los 3 tipos de respuesta del modelo. Para ello primero se debe obtener la matriz de observación como se observa en la figura 44

Figura 44

Matriz de observación

Matriz de observacion				
Clases:	Medidas			
	TP	TN	FP	FN
0	256	275	0	0
1	202	326	3	0
2	70	461	0	3

Ahora se procede a obtener las métricas de validación por cada clase como se observa en la figura 45:

Figura 45

Resultado de las métricas de métricas de validación por cada clase

Precision

0	1
1	0.9853659
2	1

$$\text{Precision} = \frac{tp}{tp + fp}$$

Recall

0	1
1	1
2	0.9589041

$$\text{Recall} = \frac{tp}{tp + fn}$$

F-measure

0	1.00
1	0.99
2	0.98

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Finalmente se ingresa un set de datos con valores para simular una predicción.

Los datos ingresados son los siguientes que se muestran en la figura 46.

Figura 46

Datos de ingresados para la simulación de la predicción.

```
#Predict  # Datos de entrada para probar
clf.predict([[120,85,6,26,1,2,3,4,5,6,7,8,9,10,10,10]])
```

El resultado de la predicción es la que se muestra en la figura 47.

Figura 47

Resultado de la predicción

```
array([2], dtype=int64)
```

Nota: el resultado es 2, lesión Media.

5.2.2 Resultados de las métricas aplicadas para el algoritmo Random Forest

El resultado obtenido de la precisión del algoritmo se muestra en la figura 48:

Accurency del 92% y un desvió standard de 0,13%.

Figura 48

Resultado de accurency y desvío standard para el algoritmo Random Forest

```
Accuracy: 0.92 (+/- 0.13)
```

Por otro lado, se valida el modelo de clasificación con la matriz de confusión y se utilizan las métricas de Precision, Recall y F1 Score para validar desempeño del modelo.

El código es el que se vé en la figura 49:

Figura 49

Script utilizado para la validación del modelo con Random Forest

```
print('Precision: %.3f' % precision_score(y_test, y_pred,average ='micro' )) #calculo la precision de la prediccion
print('Recall: %.3f' % recall_score(y_test, y_pred,average ='micro')) #calculo la precision de la prediccion
print('F1 Score: %.3f' % f1_score(y_test, y_pred,average ='micro')) #calculo la precision de la prediccion

# Matriz de Confusion
#actual = y_pred=clf.predict(y_test) # realizamos la prediccion con los datos de test
#predicted = y_pred=clf.predict(X_test) # realizamos la prediccion con los datos de test
confusion_matrix = metrics.confusion_matrix(y_test, y_pred)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [0, 1, 2])
cm_display.plot()
plt.show()

#Predict  # Datos de entrada para probar
clf.predict([[120,85,6,26,1,2,3,4,5,6,7,8,9,10,10,10]])
```

Los resultados obtenidos de las métricas de validación del modelo son los que se observan en la figura 50.

Figura 50

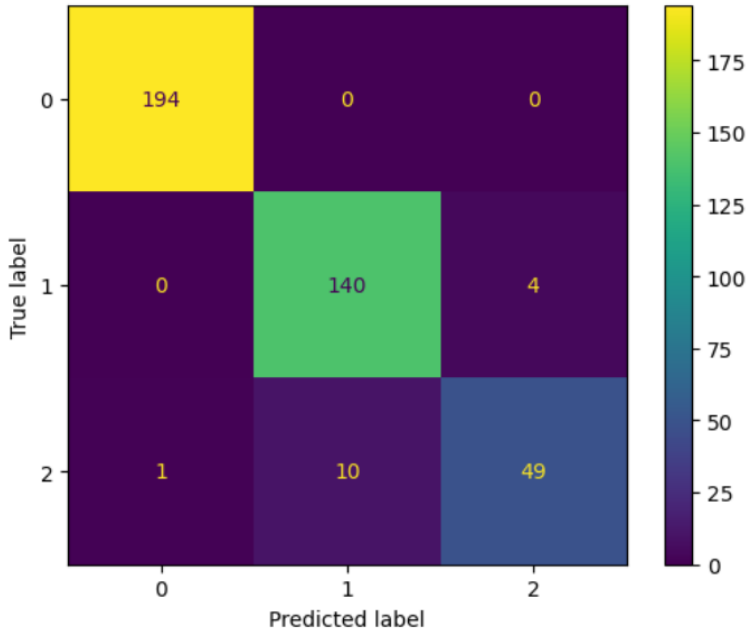
Resultados de la validacion del desempeño del modelo con Random Forest

```
Precision: 0.955
Recall: 0.955
F1 Score: 0.955
```

La matriz de confusión arrojó los datos que se observan en la figura 51:

Figura 51

Matriz de confusión con Random Forest



Finalmente se valida con las mismas métricas los 3 tipos de respuesta del modelo. Para ello primero se debe obtener la matriz de observación como se observa en la figura 52

Figura 52

Matriz de observación para el algoritmo Random Forest

Matriz de observacion				
Clases:	Medidas			
	TP	TN	FP	FN
0	194	204	1	0
1	149	248	10	4
2	49	334	4	11

Ahora se procede a obtener las métricas de validación por cada clase tal como se observa en la figura 53:

Figura 53

Validación por cada clase para el algoritmo de Random Forest

Precision

0	0.99487179
1	0.93710692
2	0.9245283

$$\text{Precision} = \frac{tp}{tp + fp}$$

Recall

0	1
1	0.97385621
2	0.81666667

$$\text{Recall} = \frac{tp}{tp + fn}$$

F-measure

0	1.00
1	0.96
2	0.87

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Los datos ingresados son los los que se muestran en la figura 54:

Figura 54

Datos ingresados de prueba para el test del algoritmo Random Forest

```
#Predict  # Datos de entrada para probar
clf.predict([[120,85,6,26,1,2,3,4,5,6,7,8,9,10,10,10]])
```

El resultado de la predicción es la que se observa en la figura 55 :

Figura 55

Resultados del test para el algoritmo Random Forest

```
array([2], dtype=int64)
```

Nota: el resultado es 2, lesión Moderada.

5.2.3 Resultados de las métricas aplicadas para el algoritmo XGBoost

El resultado obtenido de la precisión del algoritmo es el que se observa en la figura 56:

Accurency del 93% y un desvió standard de 0,14%.

Figura 56

Resultado de Accurency y desvío standard para el algoritmo XGBoost

```
Accuracy: 0.93 (+/- 0.14)
```

Por otro lado, se valida el modelo de clasificación con la matriz de confusión y se utilizan las métricas de Precision, Recall y F1 Score para validar desempeño del modelo.

El código es el que se observa en la figura 57:

Figura 57

Validación del modelo de clasificación con la matriz de confusión y utilización de las métricas de Precision, Recall y F1 Score para validar desempeño del modelo.

```
print('Precision: %.3f' % precision_score(y_test, y_pred,average ='micro' )) #calculo la precision de la prediccion
print('Recall: %.3f' % recall_score(y_test, y_pred,average ='micro')) #calculo la precision de la prediccion
print('F1 Score: %.3f' % f1_score(y_test, y_pred,average ='micro')) #calculo la precision de la prediccion

# Matriz de Confusion
#actual = y_pred=clf.predict(y_test) # realizamos la prediccion con los datos de test
#predicted = y_pred=clf.predict(X_test) # realizamos la prediccion con los datos de test
confusion_matrix = metrics.confusion_matrix(y_test, y_pred)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [0, 1, 2])
cm_display.plot()
plt.show()

#Predict  # Datos de entrada para probar
clf.predict([[120,85,6,26,1,2,3,4,5,6,7,8,9,10,10,10]])
```

Los resultados obtenidos de las métricas de validación del modelo son los que se observan en la figura 58:

Figura 58

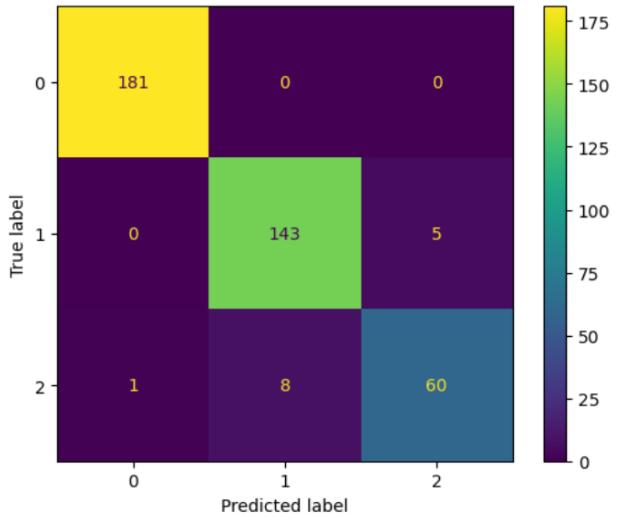
Resultados obtenidos de las métricas de validación del modelo XGBoost

```
Precision: 0.992
Recall: 0.992
F1 Score: 0.992
```

La matriz de confusión arrojó los datos que se observan en la figura 59:

Figura 59

Resultados de la matriz de confusión para el algoritmo XGBoost



Finalmente se valida con las mismas métricas los 3 tipos de respuesta del modelo. Para ello primero se debe obtener la matriz de observación como se refleja en la figura 60

Figura 60

Matriz de observación para XGBoost

Matriz de observacion				
Clases:	Medidas			
	TP	TN	FP	FN
0	181	216	1	0
1	143	242	8	5
2	60	324	5	9

Ahora se procede a obtener las métricas de validación por cada clase como se observa en la figura 61:

Figura 61

Métricas de validación por cada clase para el algoritmo XGBoost

Precision

0	0.99450549
1	0.94701987
2	0.92307692

$$\text{Precision} = \frac{tp}{tp + fp}$$

Recall

0	1
1	0.96621622
2	0.86956522

$$\text{Recall} = \frac{tp}{tp + fn}$$

F-measure

0	1.00
1	0.96
2	0.90

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Los datos ingresados son los que se observa en la figura 62:

Figura 62

Datos ingresados como test para el algoritmo XGBoost

```
#Predict  # Datos de entrada para probar
clf.predict([[120,85,6,26,1,2,3,4,5,6,7,8,9,10,10,10]])
```

El resultado de la predicción es la que se observa en la figura 63

Figura 63

Resultado de la predicción con el algoritmo XGBoost

```
array([3], dtype=int64)
```

Nota: el resultado es el número 3, lesión grave.

5.3 Escenarios

A continuación, se describen los distintos escenarios planteados para la implementación de la solución.

5.3.1 Escenario 1: Comportamiento del algoritmo Deciscion Tree

Para este algoritmo se importan las siguientes librerías tal como se muestra en la figura 64:

Figura 64

Importe de librerías para el usa del algoritmo decisión tree

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn import metrics
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import precision_score,recall_score,f1_score,precision_recall_fscore_support
```

se levanta el set de datos como se muestra en la figura 65.

Figura 65

Set de datos utilizado

```
# Levanto el Set de Datos
s = pd.read_csv("consultorio categorizado_CSV.csv")
```

Se preparan los datos para su clasificación, otorgando un entrenamiento de 60-40, se realiza el testeo y entrenamiento de los datos. Se utiliza **Cross Validation** como el evaluador de desempeño del estimador

Su código es el siguiente como vé en la figura 66.

Figura 66

Preparación de los datos para la utilización del algoritmo Deciscion Tree

```
# Preparo el set de datos
Data=s[['Jugador', 'Diagnostico','Tipo','Region','P1','P2','P3','P4',
     'P5','P6','P7','P8','P9','P10','P11','P12']]  # Features/variables independientes DATA
Target=s['Grado']  # Labels /variable dependiente o etiqueta de clase  TARGET

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(Data,Target,test_size=0.4, random_state=0) # 60% training and 40% test

#Create a Gaussian Classifier
clf = tree.DecisionTreeClassifier()

#Train the model using the training sets y_pred=clf.predict(X_test)
clf = clf.fit(X_train, y_train)

#print(clf.score(X_test, y_test))
clf.predict(X_test)
clf = tree.DecisionTreeClassifier()
scores = cross_val_score(clf, Data,Target, cv=5)
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))

#Split
y_pred = clf.fit(X_train,y_train).predict(X_test)
```

5.3.2 Escenario 2 Comportamiento del algoritmo Random Forest

Para este algoritmo se importan las siguientes librerías como se muestra en la figura 67:

Figura 67

Librerías utilizadas para el algoritmo de Random Forest

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import precision_score,recall_score,f1_score,precision_recall_fscore_support
```

se levanta el set de datos tal como se ve en la figura 68.

Figura 68

Set de datos para la utilización del algoritmo Random Forest.

```
# Levanto el Set de Datos
s = pd.read_csv("consultorio categorizado_CSV.csv")
```

Se preparan los datos para su clasificación, otorgando un entrenamiento de 60-40, se realiza el testeo y entrenamiento de los datos. Se utiliza **Cross Validation** como el evaluador de desempeño del estimador,

Su código es el siguiente como se muestra en la figura 69.

Figura 69

Cross Validation *como el evaluador de desempeño del estimador*

```
X=s[['Jugador', 'Diagnostico','Tipo','Region','P1','P2','P3','P4',
     'P5','P6','P7','P8','P9','P10','P11','P12']]  # Features/variables independientes DATA
y=s['Grado']  # Labels /variable dependiente o etiqueta de clase  TARGET

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% training and 30% test

############################## Acá empieza Machine learning ###################################

#Create a Gaussian Classifier
clf=RandomForestClassifier(n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf.fit(X_train,y_train)  #Pasamos los parametros para que se pueda entrenar

y_pred=clf.predict(X_test) # realizamos la prediccion con los datos de test
print("Accuracy:",metrics.accuracy_score(y_test, y_pred)) #calculamos el accuracy
```

5.3.3 Escenario 3 Comportamiento del algoritmo XGBoost

Para este algoritmo se importan las librerías que se observan en la figura 70:

Figura 70

Librerías utilizadas para el algoritmo XGBoost

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.ensemble import HistGradientBoostingClassifier
from sklearn.datasets import make_hastie_10_2
from sklearn import metrics
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import precision_score,recall_score,f1_score,precision_recall_fscore_support
```

se levanta el set de datos como se observa en la figura 71.

Figura 71

Set de datos levantados para el algoritmo XGBoost

```
# Levanto el Set de Datos
s = pd.read_csv("consultorio categorizado_CSV.csv")
```

Se preparan los datos para su clasificación, otorgando un entrenamiento de 60-40, se realiza el testeo y entrenamiento de los datos. Se utiliza **Cross Validation** como el evaluador de desempeño del estimador

El código es el que se observa en la figura 72:

Figura 72

Cross Validation como el evaluador de desempeño del estimador

```
# Preparo el set de datos
X=s[['Jugador', 'Diagnostico','Tipo','Region','P1','P2','P3','P4',
     'P5','P6','P7','P8','P9','P10','P11','P12']]  # Features/variables independientes DATA
y=s['Grado']  # Labels /variable dependiente o etiqueta de clase  TARGET

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% training and 30% test

############################## Acá empieza Machine learning ###################################
clf = HistGradientBoostingClassifier(max_iter=100).fit(X_train, y_train)
clf.score(X_test, y_test)

y_pred=clf.predict(X_test) # realizamos la prediccion con los datos de test
print("Accuracy:",metrics.accuracy_score(y_test, y_pred)) #calculamos el accuracy
```

5.4 Discusión

Tal como se observa en los distintos escenarios, las métricas que acompañan la validación de los modelos y la implementación de los algoritmos con la información fisiológica y psicológica son muy precisas. Esto quiere decir que existe una muy buena correlación entre dichas variables y además arrojan un resultado final muy preciso.

En este contexto se observa que este trabajo puede aplicarse a otros deportes y que puede servir de imput para inclusive ir más allá del tipo de lesión que puede tener un deportista ya que en este caso se contempla un tipo de lesión a muy alto nivel, permitiendo explorar la aplicación del mismo para lesiones de más bajo nivel o más específicas.

Conclusión

Esta tesis describe cómo se puede predecir lesiones en el deporte combinando 2 universos que hasta el momento no estaban vinculados o relacionados, estos son los de origen fisiológicos y los psicológicos. A lo largo del desarrollo se puede observar como a partir de un set de datos y la utilización de distintos algoritmos de ML, estos universos tienen muy buena correlación arrojando un resultado final muy preciso respecto a que tipo de lesión puede tener un deportista, estas son clasificadas como del tipo grave, moderada o leve.

6.1 Ventajas

La ventaja que tiene poder prevenir las lesiones en los deportistas tiene que ver con bajar el riesgo que conlleva estar lesionado en cualquier deporte, estos riesgos tienen que ver con consecuencias de repercusión en la esfera psíquica, emocional y financiera. Algunas consecuencias en las que se bajaría el riesgo son:

- Cambios y reajustes en la actividad diaria, cambiando la asistencia a entrenos y competiciones por intervenciones, reposo o rehabilitación. **Esta interrupción genera un mayor aislamiento ya que se ve interrumpida la participación en la actividad social y de grupo**.
- A causa de la lesión se pone en riesgo la posición y permanencia en el equipo, incluso tener acceso a nuevos contratos. **El riesgo de pérdida puede causar una sensación de amenaza**.
- Los plazos de recuperación no son fijos y generan una gran carga de incertidumbre y sensación de pérdida de control.
- Una lesión de gravedad incluso puede dificultar realizar actividades básicas de la vida diaria como vestirse solo, desplazarse.

Todos estos factores influyen en el bienestar psicológico de la persona. Síntomas como mayor irritabilidad, hostilidad, pensamientos negativos, síntomas depresivos o ansiedad son frecuentes en deportistas lesionados.

Por otro lado, las lesiones tienen un gran impacto en los deportistas, no solo en lo que refiere a lo fisiológico y psicológico, sino también en lo **económico/financiero**:

- Rendimiento de los deportistas con los consecuentes gastos económicos que implica el tratamiento y la rehabilitación.
- Gran impacto en la carrera de un deportista y su potencial de ingresos.

6.2 Limitaciones

La limitación que presenta el trabajo es la cantidad o volumen de información con la que se trabaja, a raíz de ello, se utilizaron algoritmos que responden muy bien con volúmenes pequeños. Esto claramente limita la utilización de otros algoritmos que son muy buenos para volúmenes más grandes de información.

6.3 Próximos pasos

Como próximos pasos un punto a trabajar es el de obtener mayor volumen de información como fuente de datos para poder trabajar con otros algoritmos más potentes y de esta manera también poder distinguir otro tipo de lesiones más específicas enfocadas como por ejemplo en hombros, rodillas y otras extremidades. Por otro lado, este trabajo se puede tomar para que se aplique en distintos deportes, no solo en el vóley.

Bibliografía

- A field-based approach to determine soft tissue injury risk in elite futsal using novel machine learning techniques. Ruiz-Pérez I, López-Valenciano A, Hernández-Sánchez S, Puerta-Callejón JM, De Ste Croix M, Sainz de Baranda P, Ayala F. Front Psychol. 2021;12:610210.

- A machine learning approach to assess injury risk in elite youth football players. Rommers N, Rössler R, Verhagen E, et al. Med Sci Sports Exerc. 2020;52:1745–1751

- A machine learning approach to assess injury risk in elite youth football players. Rommers N, Rössler R, Verhagen E, et al. Med Sci Sports Exerc. 2020;52:1745–1751. [PubMed] [Google Scholar]

-A predictive paradigm for identifying elevated musculoskeletal injury risks after sport-related concussion. Mansouri M, Roland J, Rahmati M, Sartipi M, Wilkerson G. Sports Orthop Traumatol. 2022;38:66–74. [Google Scholar]

- A retrospective study of foot biomechanics and injury history in varsity football athletes at the U.S. Naval Academy. Windsor J, Jeffries J, Sorensen J, Bach K, Benedek E, Bicher J, Pasquina P. Mil Med. 2022;187:684–689. [PubMed] [Google Scholar]

- Alessio Rossi, Luca Pappalardo , Paolo Cintia , F. Marcello Iaia ,Javier Fernandez , Daniel Medina,2017, Effective injury prediction in professional soccer with GPS data and machine learning

- Amiel Satvedi,2020, Injury Predictor For Soccer Players Using Machine Learning, Sheridan College (Oakville)

- Analysis and design of dual-feature fusion neural network for sports injury estimation model. Meng L, Qiao E. Neural Comput Applic. 2023;35:14627–14639. [Google Scholar]

Association of hip and foot factors with patellar tendinopathy (jumper's knee) in athletes. Mendonça LD, Ocarino JM, Bittencourt NF, Macedo LG, Fonseca ST. J Orthop Sports Phys Ther. 2018;48:676–684

- Athlete-customized injury prediction using training load statistical records and machine learning. Naglah A, Khalifa F, Mahmoud A, et al. 2018 IEEE Int Symp Signal Process Inf Technol. 2018:459–464. [Google Scholar]

- Berengüí y Puga, 2015; García-Mas et al., 2014; Olmedilla et al., 2018; Rubio et al., 2014

- Casper DEVANTIER, Sport Science Review, vol. XX, No. 5-6, December 2011,DOI:10.2478/v10237-011-0062-3

- Classifying motor coordination impairment in Para swimmers with brain injury. Hogarth L, Payton C, Nicholson V, et al. J Sci Med Sport. 2019;22:526–531. [PubMed] [Google Scholar]

- "Claudino et al. Sports Medicine – Open,2019, Current Approaches to the Use of Artificial Intelligence for Injury Risk Assessment and

Performance Prediction in Team Sports: a Systematic Review, https://doi.org/10.1186/s40798-019-0202-3"

- Current approaches to the use of artificial intelligence for injury risk assessment and performance prediction in team sports: a systematic review. Claudino JG, Capanema DO, de Souza TV, Serrão JC, Machado Pereira AC, Nassis GP. Sports Med Open. 2019;5:28

- Detecting injury risk factors with algorithmic models in elite women's pathway cricket. Goggins L, Warren A, Osguthorpe D, et al. Int J Sports Med. 2022;43:344–349. [PubMed] [Google Scholar]

- Effective injury forecasting in soccer with GPS training data and machine learning. Rossi A, Pappalardo L, Cintia P, Iaia FM, Fernàndez J, Medina D. PLoS One. 2018;13:0. [PMC free article] [PubMed] [Google Scholar]

- Evidence-based sports medicine to prevent knee joint injury in triple jump. Wang S, Lyu B. Rev Bras Med Esporte. 2022;28:195–198. [Google Scholar]

- García et al., 2015; Padegimas et al., 2016;Pujals et al., 2016

- Genomic prediction of tendinopathy risk in elite team sports. Rodas G, Osaba L, Arteta D, Pruna R, Fernández D, Lucia A. Int J Sports Physiol Perform. 2019;15:489–495. [PubMed] [Google Scholar]

- Glendon, Clarke y McKenna, 2016; Gómez-Espejo et al., 2017; Johnson e Ivarsson,2017; Olmedilla, Rubio y Ortega, 2015; Singh y Conroy, 2017

- Herring et al., 2017; Schinke et al., 2017; Soligard, et al., 2016

- Identification of neuromuscular performance parameters as risk factors of non-contact injuries in male elite youth soccer players: a preliminary study on 62 players with 25 non-contact injuries. Kolodziej M, Nolte K, Schmidt M, Alt T, Jaitner T. Front Sports Act Living. 2021;3:615330

- Jesús Ríos Garit, Yanet Pérez Surita, María del Carmen Chávez Cárdenas,2021, Predictores psicológicos de lesiones en lanzadores de béisbol de alto Rendimiento, Universidad Central "Marta Abreu" de Las Villas

- Just how confident can we be in predicting sports injuries? A systematic review of the methodological conduct and performance of existing musculoskeletal injury prediction models in sport. Bullock GS, Mylott J, Hughes T, Nicholson KF, Riley RD, Collins GS. Sports Med. 2022;52:2469–2482. [PubMed] [Google Scholar]

- Machine learning methods in sport injury prediction and prevention: a systematic review. Van Eetvelde H, Mendonça LD, Ley C, Seil R, Tischer T. J Exp Orthop. 2021;8:27. [PMC free article] [PubMed] [Google Scholar]

- Machine learning to predict lower extremity musculoskeletal injury risk in student athletes. Henriquez M, Sumner J, Faherty M, Sell T, Bent B. Front Sports Act Living. 2020;2:576655. [PMC free article] [PubMed] [Google Scholar]

- María del Carmen Juan Llamas,2021, Modelización matemática para la predicción y prevención de lesiones deportivas, Universidad Complutense de Madrid

- María Dolores Hidalgo,2013,Características psicológicas asociadas a la incidencia de lesiones en deportistas de modalidades individuales,Grados de Educación y Psicología, Universidad Católica San Antonio de Murcia

- New machine learning approach for detection of injury risk factors in young team sport athletes. Jauhiainen S, Kauppi JP, Leppänen M, et al. Int J Sports Med. 2021;42:175–182. [PubMed] [Google Scholar]

- Prediction of injuries in CrossFit training: a machine learning perspective. Moustakidis S, Siouras A, Vassis K, Misiris I, Papageorgiou E, Tsaopoulos D. Algorithms. 2022;15:77. [Google Scholar]

- Prediction simulation of sports injury based on embedded system and neural network. Shen H. Microprocess Microsyst. 2021;82:103900. [Google Scholar]

- Predictive modeling of hamstring strain injuries in elite Australian footballers. Ruddy JD, Shield AJ, Maniar N, et al. Med Sci Sports Exerc. 2018;50:906–914. [PubMed] [Google Scholar]

- Predictive modelling of training loads and injury in Australian football. Carey DL, Ong K, Whiteley R, Crossley KM, Crow J, Morris ME. Int J Comput Sci Sport. 2018;17:49–66. [Google Scholar]

- Random forest algorithm to identify factors associated with sports-related dental injuries in 6 to 13-year-old athlete children in Hamadan, Iran-2018 - a cross-sectional study. Farhadian M, Torkaman S, Mojarad F. BMC Sports Sci Med Rehabil. 2020;12:69. [PMC free article] [PubMed] [Google Scholar]

- Random forests. Breiman L. Mach Learn. 2001;45:5–32. [Google Scholar]

- Sports injury rehabilitation intervention algorithm based on visual analysis technology. Chen X, Yuan G. Mob Inf Syst. 2021;2021:9993677. [Google Scholar]

- Using machine learning to predict lower-extremity injury in US special forces. Connaboy C, Eagle SR, Johnson CD, Flanagan SD, Mi QI, Nindl BC. Med Sci Sports Exerc. 2019;51:1073–1079

- Using weather and playing surface to predict the occurrence of injury in major league soccer games: a case study. Landset S, Bergeron MF, Khoshgoftaar TM. 2017 IEEE Int Conf Inf Reuse Integr. 2017:366–371. [Google Scholar]