Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑARÍA INFORMÁTICA
MENCIÓN EN COMPUTACIÓN

Análise, deseño e implementación de modelos de aprendizaxe máquina: predición usando datos sensóricos de buques arrastreiros

Estudante: Alejandro Fernández Luces
Dirección: José Rouco Maseda

A Coruña, setembro de 2020.

Ao nordés da ría

Agradecementos

Quixera agradecer primeiramente a José Rouco Maseda por aceptar este proxecto, se non fora pola súa tutela este traballo sería imposible.

Tamén me gustaría mostrar o meu agradecemento a todos os informantes que me ensinaron con paixón todos os detalles do seu traballo, decisivos para levar a cabo este estudo.

Por último quero agradecer profundamente a miña familia por estar aí, a Clara por aturarme nos momentos máis duros, e aos meus compañeiros de facultade, cos que compartín tantos momentos inesquecibles.

Resumo

En Galicia, a pesca representa un sector de alta importancia a nivel socioeconómico. En concreto, o eido da pesca de arrastre é unha das artes pesqueiras con máis peso. Con todo, os patróns dos buques arrastreiros enfróntanse a certos riscos debido ao descoñecemento do fondo mariño. A isto súmase un certo atraso tecnolóxico do propio sector.

Partindo da infraestrutura xa existente neste tipo de barcos, nesta proposta preténdese deseñar unha posible solución a un dos problemas que máis perdas económicas xera: o enfangamento do aparello. O obxectivo principal do proxecto é obter unha predición precoz do estado do aparello para evitar este problema. Isto conseguirase mediante a aplicación de técnicas de aprendizaxe máquina a datos sensóricos de buques arrastreiros. Aplícanse algoritmos tales como a regresión lineal, os bosques aleatorios, e as máquinas de soporte vectorial para obter modelos de predición.

Búscase ademais despregar os modelos obtidos nunha aplicación web que poida ser instalada nas computadoras das embarcacións. Os resultados obtidos nos modelos foron adecuados para este problema, pero ao ser este un estudo novidoso neste tema, non se puideron comparar os resultados con outros traballos semellantes.

Abstract

In Galicia, fishing represents a socioeconomic high impact sector. Particularly, trawl fishing is one of the most important fishing techniques. Nonetheless, captains from trawling ships face some risks as a result of the lack of knowledge of the marine bottom. Also, the sector suffers some technological backwardness.

Starting from the current infrastructure in this kind of ships, in this proposal a possible solution is designed for one of the problems that generates more economical losses: the mud catch in the trawling rig. Tha main objective in this project is to obtain an early prediction of the state of the trawling rig to avoid this problem. This will be achieved through the application of machine learning techniques to sensory data from trawling vessels. Algorithms like linear regression, random forests and support vector machines are used to obtain the prediction models.

We also seek to deploy these models in a web application that can be installed in the vessels computers. The obtained results were good for our data, but because this research is unique in this field, it couldn't be compared with similar works.

Palabras chave:

- Barcos de arrastre
- Pesca
- Sensores
- Aprendizaxe máquina
- Regresión Lineal
- Máquinas de soporte vectorial
- Bosques aleatorios

Keywords:

- Trawling ships
- Fishing
- Sensors
- Machine Learning
- Linear regression
- Support vector machines
- Random forests

Índice Xeral

Índice de Figuras

Índice de Táboas

Capítulo 1

Introdución

Neste capítulo expoñerase o problema que este proxecto pretende resolver. Explicarase tamén a motivación da que xorde, así como os obxectivos que se desexan alcanzar. Finalmente describirase a estrutura desta memoria.

1.1 O proxecto

1.1.1 Descrición do problema

O proxecto centrarase nunha problemática que aflixe a buques pesqueiros de arrastre que traballan tanto pola costa de Galicia coma noutros caladoiros. A arte de pesca do arrastre consiste en lanzar un aparello pola popa do barco, que consta dunha rede que, grazas a uns pesos, vaise arrastrando polo fondo para así capturar os bancos de peixes que alí se atopan. Debido a que o aparello se move polo fondo, pode atoparse con diversos problemas, como enfangar, quedar enganchado nun casco dun barco afundido, romper contra pedras, etc. Correspóndelle ao patrón do barco, coa súa experiencia e coa información que lle poidan aportar as cartas mariñas, intuír sobre que tipo de terreo está traballando, mais nunca sabe a ciencia certa as características do lugar por onde está pasando.

Isto implica que hai un grao de incerteza que supón unha serie de riscos para o barco. A máis grave é a perda de todo o aparello co que se está a pescar, que pode chegar a custar decenas de miles de euros, supoñendo un impacto económico moi grande para o barco. Tamén pode darse o caso de que se produza a perda de só o segmento final da rede, que é o máis común, xa que hai mecanismos para desprender esta parte no caso de que algo vaia mal. Deste xeito, evítase perder tamén toda a sensórica do aparello. Por último, o inconveniente máis frecuente é que o aparello recolla fango do fondo. Cando isto sucede, inicialmente o patrón non recoñece se o que está a coller son capturas ou non, polo que segue adiante. A confusión repercute negativamente no rendemento do barco xa que provoca un consumo de combustible moi alto, fai que se perda tempo de pesca e aumenta significativamente o risco

de chegar a perder o aparello.

Para esta investigación partirase do emprego e comparación de varias técnicas de aprendizaxe máquina que analizarán os datos sensóricos que proveñen do aparello para intentar evitar riscos durante a pesca. Estas técnicas permiten predicir de forma temperá cales serán as medidas da abertura do aparello e da lonxitude do cable. O fin último é que o patrón poida interpretar estes sinais preditos e estimar se o aparello vai enfangar proximamente ou non. Para conseguir isto experimentarase con datos sensóricos, proporcionados por unha empresa de electrónica naval, procedentes de diferentes barcos en distintas localizacións. A característica común da información utilizada é que provén de buques de arrastre sinxelo.

Unha vez rematada a experimentación, valorarase cales son as características dos datos máis adecuadas para ter en conta e que modelos de aprendizaxe máquina son os máis axeitados para realizar a predición. Seleccionarase en último lugar o modelo que proporcione mellores resultados e despregarase nunha aplicación web.

1.1.2 Motivación

A principal razón de ser deste proxecto xorde da necesidade de actualización da flota pesqueira. Son moi poucos os traballos dispoñibles sobre aprendizaxe máquina aplicada a este sector, mais aínda son menos aqueles que analizan sinais procedentes dun aparello en funcionamento.

Despois de conversar con expertos e profesionais da pesca arrastreira é fácil comprender mellor as particularidades e necesidades desta flota. Estas entrevistas informais evidencian que a infraestrutura dispoñible nun barco destas características fai posible incorporar de maneira sinxela transformacións informáticas que poidan axudar a mellorar e facilitar o traballo neste sector. No caso particular do problema que se aborda nesta investigación, abondaría cunha modificación sinxela da computadora do barco para implantar a mellora aquí proposta e minimizar o risco de perda das redes, xa que a computadora aporta informacións útiles ao patrón e é quen de dar conta do estado do aparello sen necesidade de engadir ningún sistema físico adicional.

Este traballo tamén responde á vontade de que a aprendizaxe máquina non quede só na investigación teórica ou nas facultades de informática e se recoñezan os seus usos e aplicacións para mellorar o rendemento naqueles ámbitos onde sexa posible. É especialmente interesante manexar estes conceptos no campo dos distintos sectores industriais, na liña da transformación paulatina que se está a producir neste ámbito e que se coñece como industria 4.0 [2]. Deste xeito. preténdese incluír nos procesos de produción a intelixencia artificial, utilizar *big data* para tirar rendemento da grande cantidade de datos dos que xa se dispoñen e fomentar a interconexión entre máquinas de fábricas para automatizar a coordinación entre elas. Polo tanto, tanto a carencia de traballos sobre aprendizaxe máquina e pesca coma a necesidade real

de dar solución a problemas dos traballadores deste sector que mellorarían coa aplicación de tecnoloxías informáticas xustifican a realización deste proxecto de investigación.

1.1.3 Obxectivos

Como xa se avanzou anteriormente, o obxectivo fundamental deste traballo é a obtención dun modelo de aprendizaxe máquina que mellore lixeiramente as condicións de traballo no sector da pesca de arrastre. O proxecto estará dividido en dúas partes ben diferenciadas entre elas, sendo a primeira todo aquilo relacionado coa investigación previa necesaria, e a segunda dedicada á implementación dun servizo para darlle usabilidade aos resultados obtidos da investigación realizada na primeira parte.

Na primeira parte construirase un modelo de aprendizaxe máquina que permita, dada unha serie temporal de datos, predicir os próximos valores desta serie. Para isto necesitarase facer un preprocesado dos datos en bruto dos que se dispoñen. A continuación, empregaranse diversos modelos de aprendizaxe supervisada. Mediante o uso de hiperparametrización, faranse adestramentos de diversas configuracións dos modelos de máquinas de soporte vectorial, *random forest*, e regresión lineal. Grazas a isto poderanse comparar todas as configuracións de todos os modelos en función da precisión con respecto aos datos de test.

A continuación, na segunda parte, implementarase un servizo web no que se despregará este modelo. Seguindo unha arquitectura REST, farase un *back-end* que permita obter predicións e subir conxuntos de datos novos, ademais de soportar a subida de datos en vivo. Tamén se implementará a parte do cliente, na que poderemos ver mediante unha gráfica tanto a información cargada coma a predición.

O obxectivo final será dispoñer dun sistema de predición de series temporais dos datos sensóricos do aparello de arrastre, cunha interface que permita ao usuario interactuar co sistema de forma sinxela.

1.1.4 Estrutura da memoria

A configuración desta memoria está estruturada de acordo coas peculiaridades do tipo de traballo que se realizou, e ademais conta cunha introdución para contextualizar o campo no que se fai a investigación e unha reflexión final.

- **Capítulo 1: Introdución.** Descrición do proxecto, xunto coa motivación da que xorde e os obxectivos que se queren acadar. Tamén inclúe a explicación da estrutura da memoria.
- **Capítulo 2: Contextualización.** Define o dominio no que se desenvolve o traballo, ademais dos conceptos teóricos máis importantes dos que se fai uso ao longo do proxecto e as tecnoloxías concretas empregadas.

- **Capítulo 3: Planificación e metodoloxía.** Neste capítulo expoñerase a planificación do desenvolvemento do proxecto, a metodoloxía empregada, os materiais necesarios e unha estimación de custos.
- **Capítulo 4: Análise de datos.** Explícase nesta sección todo o proceso da análise de datos levada a cabo, dende as entrevistas cos expertos ata a obtención dos modelos de aprendizaxe máquina.
- **Capítulo 5: Desenvolvemento web.** O apartado está constituído por unha descrición do proceso de desenvolvemento dunha aplicación web na que despregar o modelo obtido.
- **Capítulo 6: Conclusións e traballo futuro.** Este apartado inclúe o resumo dos resultados obtidos e que supoñen con respecto ao estado da arte actual. Por último, reflexiónase sobre as posibles melloras e camiños que se poden seguir a raíz deste proxecto.

Capítulo 2

Contextualización

PARA ter un mellor entendemento do interese deste traballo, debemos comprender o contexto no que se desenvolve. Neste capítulo realizarase unha descrición detallada do dominio do proxecto, así como do estado da arte, dos conceptos teóricos empregados, e das tecnoloxías que se usarán.

2.1 A pesca

Debido a que o proxecto se desenvolve nun dominio que non é habitual tratar dentro da informática, é conveniente contextualizar a importancia histórica da pesca en Galicia, así como explicar a grandes trazos a arte de pesca na que se centra esta proposta.

2.1.1 A pesca en Galicia

Historicamente, o sector da pesca tivo un alto impacto socioeconómico en Galicia. Varias vilas das nosas costas xorden precisamente da explotación pesqueira e de toda a industria que esta xera. Foron significativas na historia económica de Galicia as fábricas de conservas, as de salgaduras e todas as actividades derivadas da venda do peixe, etc. Nas últimas décadas, dende a entrada na Unión Europea [3], produciuse un progresivo declive do sector pesqueiro e outras ocupacións foron gañando máis peso.

Cómpre mencionar tamén que o sector da pesca sofre de certo abandono tecnolóxico, tal e como reportaron nas entrevistas profesionais do sector en referencia tanto a buques que traballan en caladoiros estatais como en barcos que traballan no estranxeiro. Está xeneralizado o emprego de tecnoloxías básicas coma a navegación por GPS e diversos sensores que axudan a facilitar a pesca, mais non hai dispoñibles ferramentas informáticas avanzadas que permitan automatizar certas tarefas. Por este motivo, gran parte do traballo e do éxito das campañas dependen da experiencia, coñecemento e intuición do propio patrón. Isto resulta rechamante, posto que, a pesar de que en Galicia hai forte presencia do sector da enxeñaría informática, e de

que as vendas nas lonxas no ano 2019 tiveron un valor de 464.888.248€ [4], non parece haber neste ámbito un gran interese nunha modernización nos termos da industria 4.0 referidos anteriormente [2].

Hai que valorar que dentro do sector, existen diversas formas de captura, como podemos ver na figura 2.1. A forma na que se captura o peixe denomínase arte de pesca, e esta inflúe crucialmente no tipo de especies que se busca obter. Por exemplo, para a captura de mariscos, a arte de pesca máis habitual é o emprego de nasas, armazóns de madeira ou metal cun pequeno furado no que entran os animais. Outra arte de pesca con moita importancia é a pesca do cerco, na que un ou varios barcos cercan un banco de peixes empregando unha rede, pechándoa despois por debaixo e capturando así unha gran cantidade de exemplares. Mais o obxecto

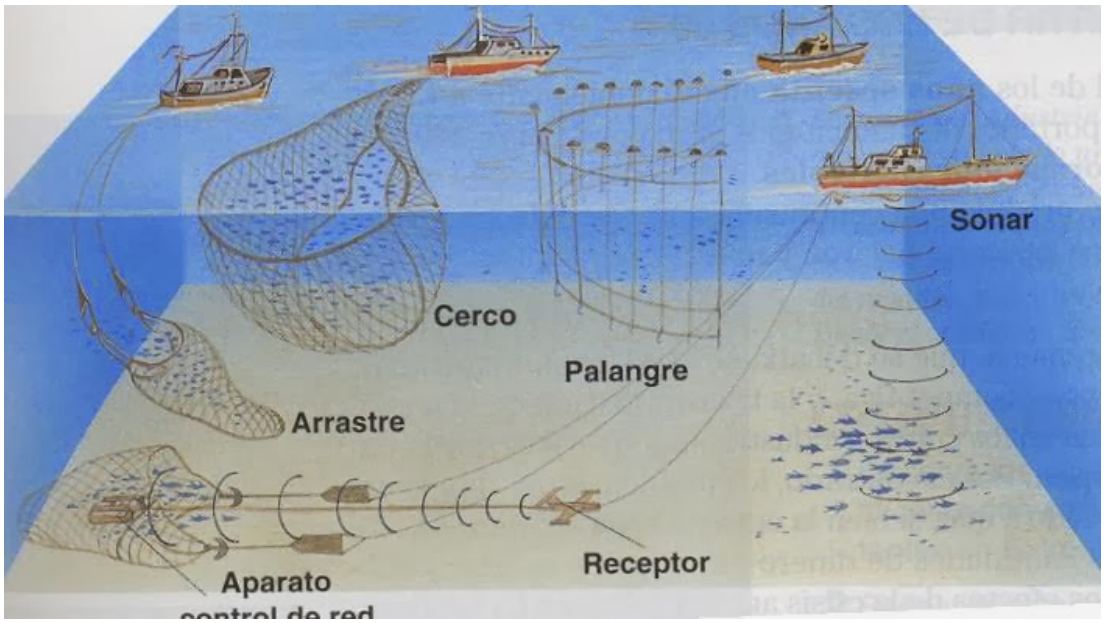

Figura 2.1: Exemplos de artes de pesca industriais.

de interese deste proxecto será a pesca de arrastre, que é a máis común para os peixes que viven no fondo mariño: a raia, a meiga e outras especies de alto valor comercial son algunhas das capturas habituais. Grazas a que o aparello de pesca desta arte contén sensores, existe a posibilidade de procesar a información que proporcionan e de crear aplicacións informáticas baseadas nestes datos.

2.1.2 A pesca de arrastre

A arte de pesca do arrastre sinxelo é unha técnica que consiste no lanzamento dun aparello pola popa do barco, o cal se vai arrastrando polo fondo mariño co impulso dado do barco.

Este aparello é preparado en terra antes de saír ao mar. O habitual é que o deseño o faga

o patrón do barco e os mariñeiros se encarguen de construilo.

No barco, hai unhas máquinas a bordo, chamadas comunmente *maquinillas*, motores eléctricos que teñen a función de recoller e largar o cable do aparello. Estes aparellos conteñen uns tensiómetros que miden a carga que leva cada un dos dous cables. Con todo, normalmente estes tensiómetros son analóxicos, polo que non pasan pola informática do barco, ao contrario ca os sensores.

Na figura 2.2 podemos ver as partes das que se compón un aparello de arrastre. O barco tira polos cables de arrastre, e á súa vez tiran das portas. Estas teñen dúas funcións, a de exercer de peso para que o aparello vaia arrastrando polo fondo, e a de dirixir a apertura da rede. As portas dispoñen dun sensor que detecta a distancia que hai entre elas. Esta distancia indícanos a carga que leva o aparello, aínda que non é quen de identificar de que é a carga, deixando así unha marxe de incerteza e de risco. Canto menor sexa a abertura entre as portas, máis carga estará soportando o aparello. Ademais, grazas aos sensores tamén é posible saber a inclinación con respecto a todos os eixos das portas. Unha inclinación indebida indica un contratempo, que pode ser un problema ao armar o aparello, un cable máis longo ca o outro, un problema na malleta, etc.

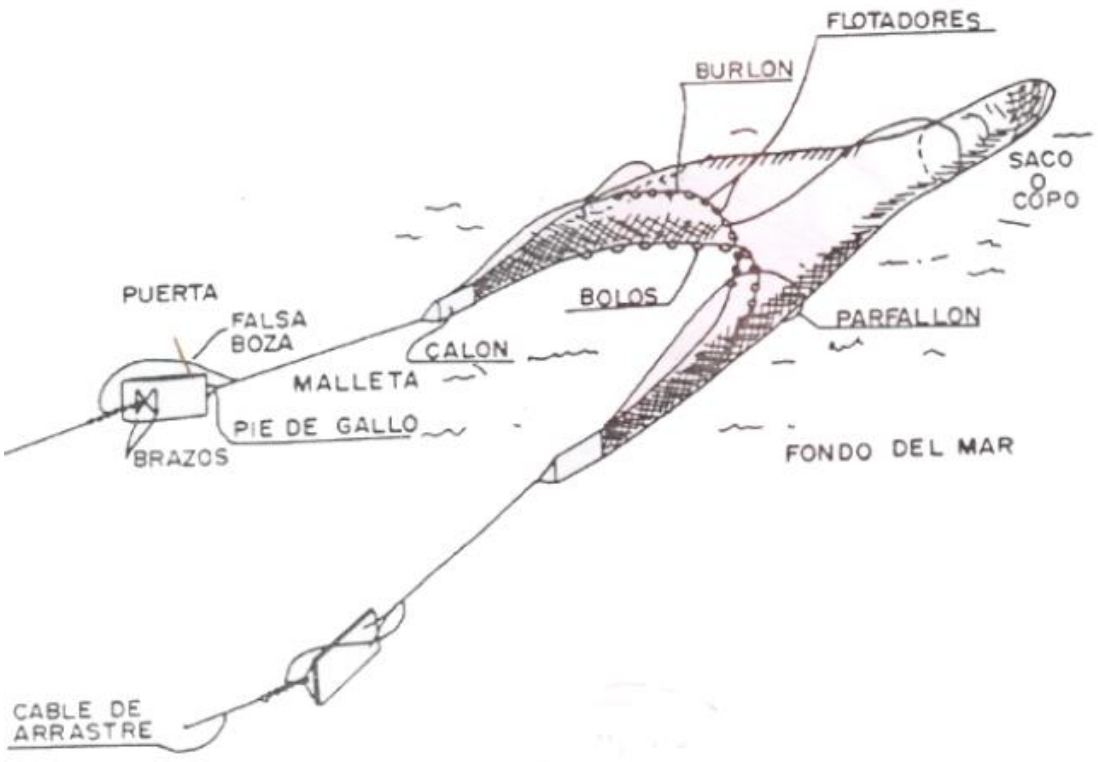

Figura 2.2: Partes do aparello de arrastre.

Cómpre explicar tamén o papel dos bolos, uns pesos que manteñen a boca do aparello

contra o chan, e o do burlón, no que hai uns flotadores que abren a boca do aparello. No segmento final atópase o saco, que é onde remata o peixe, e que dispón dun ou varios sensores de captura. O funcionamento deste mecanismo consiste nun interruptor accionado por unha corda que queda tirante cando a rede se expande co peso das especies pescadas. É importante destacar que cando no saco entra fango, o sensor de capturas non se acciona, xa que o fango vaise coando polos ocos da rede. Aínda así, o fango fai que o aparello simule levar unha carga moi alta, que pode ser confundida cunha gran cantidade de capturas.

Normalmente, o patrón pode distinguir entre unha captura de peixe e o fango grazas por un lado ao sensor de capturas, e por outro, á súa experiencia. Pode intuír que o que está collendo é fango cando a abertura das portas se reduce máis rápido do que debería, ao mesmo tempo que o barco se vai detendo. Tamén pode mirar os tensiómetros da *maquinilla*, que se dispararán tanto se se enfangou coma se se enganchou o aparello no fondo mariño.

É o feito de ser esta unha arte de pesca cuxas embarcacións están altamente informatizadas o que xustifica a investigación neste ámbito concreto, xa que noutros tipos de pesca non se conta con apenas tecnoloxía informática ou esta non se precisa para realizar o traballo no mar.

2.2 Estado da arte

O propósito desta sección é analizar cales son as tecnoloxías punteiras das que se dispoñen actualmente para a pesca de arrastre, ademais do estado da arte con respecto á investigación dispoñible actualmente neste sector.

2.2.1 Programas empregados nos barcos de arrastre

Preséntanse aquí os programas informáticos de máis relevancia que se empregan actualmente nos barcos pesqueiros. Hai que ter en conta que cada empresa pesqueira ten os seus métodos e os seus tratos comerciais, polo que o uso dunha aplicación ou outra varía entre buques e compañías. É importante destacar, ademais, a heteroxeneidade dos posibles sensores que leva un barco, o que fai tamén variar as aplicacións que emprega para procesar os datos procedentes destes.

Turbowin

Turbowin é o programa por excelencia da frota pesqueira de arrastre. É unha aplicación de escritorio na que se amosan as cartas náuticas, e na cal podemos gardar os lances que se fan, así como cargar externamente os lances realizados por outras persoas para coñecer as características destes. Cómpre destacar que ademais permite realizar anotacións para cada lance. Isto permite ao usuario ter un coñecemento a priori sobre as zonas nas que pode traballar e nas que non.

Este programa conta coa gran vantaxe de ter unha interface sinxela, moi necesaria, posto que en xeral, non hai un bo manexo de aparellos informáticos mentres se está nunha ruta ou nun lance. Tamén mostra toda a información referente a posición do barco, a súa velocidade e a súa dirección grazas á súa conexión co GPS. Na figura 2.3 obsérvase un exemplo dunha captura deste programa, e pódese ver que tipo de información achega ao usuario.

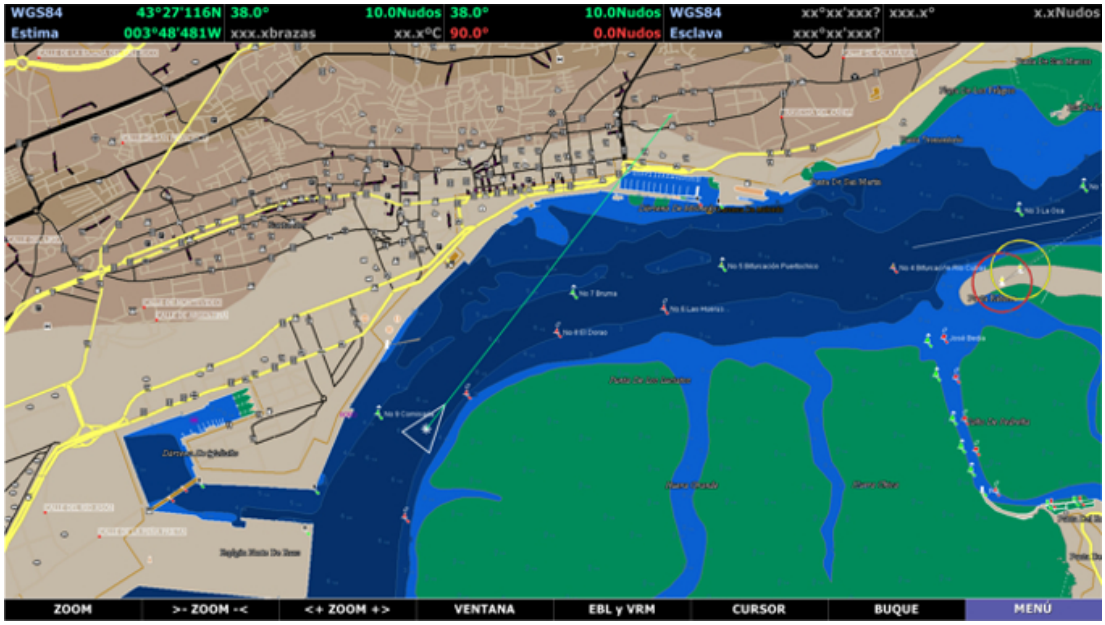

Figura 2.3: Captura de pantalla do programa TurboWin.

Unha alternativa a este programa sería o MaxSea, tamén moi empregado polos barcos pesqueiros, e cunhas funcións semellantes.

Programas de procesamento da información dos sensores

Existe diversidade de empresas de sensórica naval, cada unha das cales vende a sensórica do barco xunto co software informático que procesa a información dos propios sensores. Por mor do carácter privativo destas empresas, cada conxunto de sensores dunha empresa require dun programa específico de procesamento de datos sensóricos.

Un exemplo deste tipo de programas é TrawlMaster. Este programa proporciona dunha forma tanto visual coma numérica os valores dos sensores. Amosa en 3D como está a situación tanto das portas coma da propia rede, aínda que o que máis lle interesa ao patrón neste caso e a información visual das portas.

Na figura 2.4 pódese comprobar como o patrón visualiza tanto en 3D como a través dunha gráfica temporal o estado das portas e o estado do aparello. As medidas de babor e estribor correspóndese cos dous cables que van dende o barco ata o aparello.

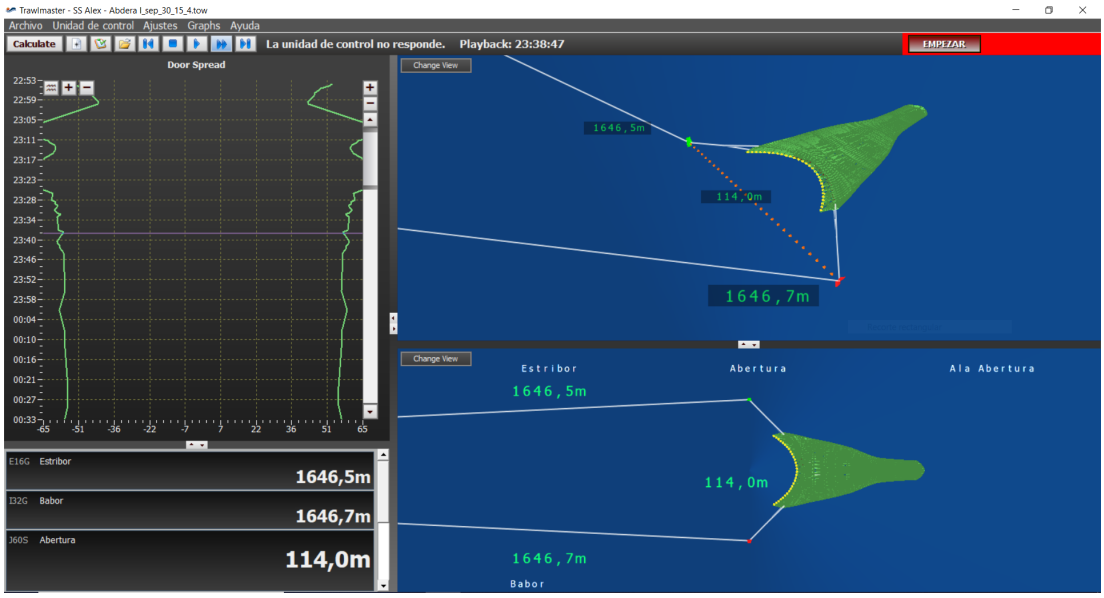

Figura 2.4: Captura de pantalla do programa TrawlMaster.

Trawl Vision Simulator

Este programa, creado pola empresa Acruxsoft, emprégase para simular como actuaría un aparello cunha configuración determinada. Resulta moi útil á hora de armar o aparello. No canto de facer proba e erro, e perder tempo de pesca debido a problemas coas medidas do aparello, simúlase neste programa a configuración desexada e ponse a proba. Cando se obtén a configuración óptima, procédese a armar o aparello.

Na figura 2.5 vemos un exemplo de como se vería o programa, así como unhas das moitas variables que se poden cambiar para configurar o aparello.

2.2.2 Investigacións

En canto á aplicación de aprendizaxe máquina para procesamento de sensores en barcos, desafortunadamente, e segundo os nosos coñecementos, non hai nada escrito aínda. Mais si podemos atopar estudos que empregan datos procedentes da actividade pesqueira para procesalos con aprendizaxe máquina. Por exemplo, desenvolveuse un modelo de aprendizaxe máquina con datos procedentes de S-AIS, un sistema de identificación e localización automático que empregan a gran maioría dos barcos de forma obrigatoria. Neste traballo conseguiuse distinguir mediante o sinal GPS se un barco de arrastre estaba a pescar ou non cunha precisión do 83%[5]. En relación á optimización da pesca, é posible atopar, tamén, estudos sobre o comportamento dos bancos de peixes, co fin de ter un mellor entendemento á hora de desenvolver un método de pesca máis eficiente [6]. Neses mesmos estudos, obsérvanse ademais

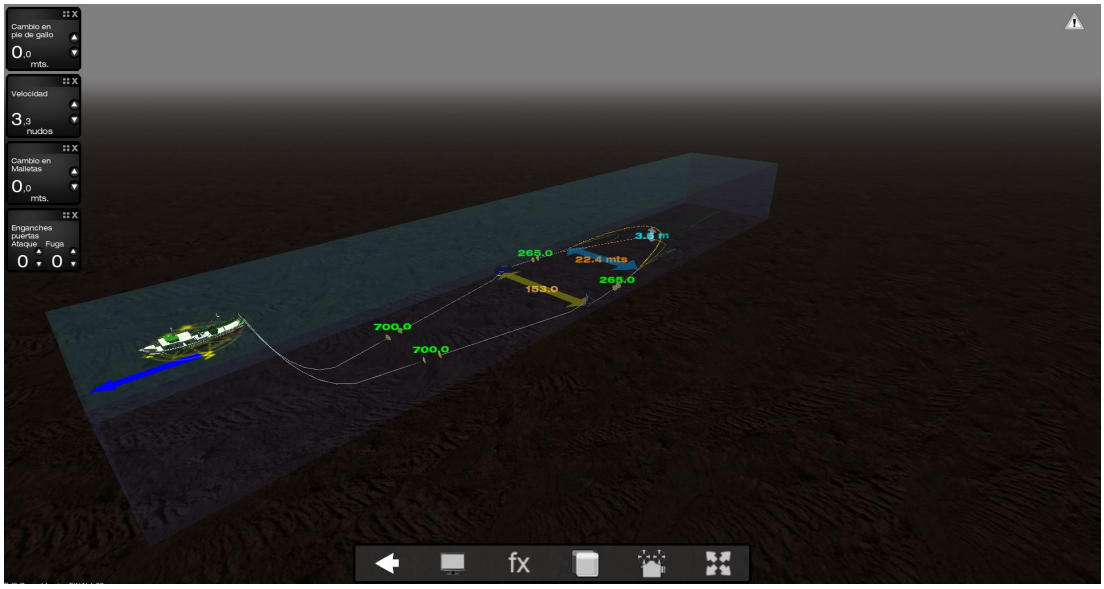

Figura 2.5: Captura de pantalla do programa Trawl Vision Simulator.

investigacións sobre como os peixes aprenden respostas á pesca de arrastre, o cal provoca a diminución do número de capturas. Así mesmo, comprobouse que engadir luces artificiais no aparello de arrastre non implica ningunha mellora sobre a cantidade de capturas obtidas [7].

Sobre a aplicación de aprendizaxe máquina a sensores non relacionados coa pesca, si podemos atopar unha gran cantidade de información. Como en moitas aplicacións do aprendizaxe máquina, séguese un proceso típico de recollida de datos, procesado, aprendizaxe e análise de resultados [8]. Algúns dos modelos de aprendizaxe máquina empregados para datos sensóricos son a regresión lineal sinxela [8], os modelos ARIMA [9] e random forest [10]. Aínda que os datos procedentes dos sensores poden ser tratados como series temporais, convén ter en conta que modelos como o ARIMA están pensados para series con estacionalidade [11], o cal non é aplicable aos datos sensóricos dos barcos.

Unha vez precisado o estado da arte na cuestión que estamos a tratar, pódese determinar que se trata dun campo pouco explorado. Na área da sensórica, polo contrario, si temos unha gran cantidade de estudos sobre a aprendizaxe máquina, polo que se intentará aplicar os conceptos da sensórica xeral á sensórica do sector pesqueiro.

2.2.3 Discusión

Ata a data de hoxe, a investigación centrouse no impacto da pesca do arrastre sobre o medio ambiente en vez de en melloras sobre os sistemas actuais da pesca. En concreto, prestouse pouca atención á mellora dos sistemas sensóricos nas embarcacións do arrastre. Por outra banda, afortunadamente dispoñemos dunha considerable cantidade de literatura científi-

fica publicada sobre o proceso de información que provén de sensores en xeral. Contamos con moitos estudos sobre a aprendizaxe máquina aplicada a datos sensóricos, cuxos resultados se poden extrapolar ata certo punto para o noso caso, sobre todo á hora de escoller que algoritmos poderían funcionar ben. Estes estudos tamén proporciona unha plataforma sobre como é habitualmente o procesado de datos: é necesario limpalos de datos erróneos, normalizalos a unha frecuencia fixa, e adaptalos ao sistema que os vaia a procesa. Ademais estes traballos indican que transformacións poden facerse sobre este tipo de datos para engadir características, coma a transformada de fourier, o seno, o coseno, etc.
Grazas á base que nos proporcionan as investigacións previas, podemos combinar o coñecemento da literatura científica sobre a sensórica e a aprendizaxe máquina cos datos de sensores procedentes das embarcacións de arrastre. O obxectivo de acadar un modelo de aprendizaxe máquina que nos permita obter predicións dos datos procedentes dos sensores dos barcos intenta encher un baleiro existente actualmente na literatura científica no eido da pesca. Ademais, proporciona ao usuario unha información complementaria sobre o estado do aparello da que actualmente non se dispón neste tipo de barcos. A predición de datos sensóricos nesta área supón que os seus usuarios teñan un maior coñecemento sobre o que está a suceder no aparello de arrastre. Grazas a esta información, pódese dar un potencial aumento da eficiencia na pesca, o que provoca unha mellora no impacto medioambiental que supón esta arte de pesca. Esta mellora débese a dous factores: o primeiro, ao obter máis capturas durante un lance, redúcense os danos provocados no chan mariño, xa que se mellora a proporción de capturas por lance, reducindo así a cantidade de lances, e segundo, o aforro de combustible que supón coñecer de forma temperá se un aparello está enfangando ou non. É importante resaltar que toda a infraestrutura necesaria que requiría o despregamento dun modelo deste estilo xa se atopa na maioría das embarcacións, dende os sensores ata a computadora do barco. En xeral, este proxecto constitúe unha oportunidade para fomentar o avance das investigacións sobre a informatización do eido da pesca do arrastre.

Por outra banda, en relación aos programas informáticos que se manexan actualmente, estes ofrecen aos seus usuarios tanto información necesaria en relación á navegación en xeral, como certa axuda sobre onde se pode pescar e asistencia durante o lance. Mais ningún deles ofrece solucións intelixentes que proporcionen un procesado de datos máis alá de amosalos. Con todo, vemos unha clara carencia dunha interpretación dos datos no software, acentuada pola falta de literatura científica que apoie unha implementación desta interpretación. Este proxecto parte da vantaxe que supón a infraestrutura dispoñible nos buques arrastreiros e deseña unha ferramenta facilmente integrable en calquera aplicación onde se procesen os valores dos sensores, que xa contan coa información que necesitan os modelos de aprendizaxe máquina.

Todo isto indica que hai unha lagoa na investigación neste ámbito, que o traballo que

se pretende facer intenta encher mediante a obtención dun modelo de aprendizaxe máquina válido para os requirimentos do sector náutico de arrastre. Para contribuír a este fin realizarase, en primeiro lugar, unha investigación de aprendizaxe máquina sobre os sensores dos arrastreiros, partindo do procesado de datos extraídos de diversas embarcacións desta tipoloxía. O procesado garante que a información quede libre de datos erróneos e permite que estes sexan procesables pola linguaxe de programación que se vaia empregar, neste caso *python*. Posteriormente, estes datos transformaranse para que poidan ser entrada dos modelos de aprendizaxe máquina que se escollan. Será a comparación do erro de predición de cada modelo respecto dos datos de test o que permita seleccionar que modelo se despregará para ser utilizado. Por último, para facer unha correcta visualización do modelo resultante desta investigación, implementarase unha sinxela aplicación web que permita simular un uso real. Preténdese, así, facer un proxecto novidoso, dun tema que, a día de hoxe, aínda non recibiu atención desde a literatura científica.

2.3 Marco teórico

A continuación explicaranse os conceptos teóricos máis importante que se empregarán ao longo do proxecto, co fin de facilitar a comprensión deste.

2.3.1 Sensores

Un sensor é un aparello cuxo propósito consiste en detectar eventos do mundo real e transmitir esa información do ambiente a un sistema electrónico, sendo este normalmente unha computadora. O conxunto de sistemas que procesan os datos dende o mundo real ata que eses datos son lexibles por unha computadora, no que se inclúen os sensores, denomínase sistema de adquisición de datos [12]. Este sistema leva a cabo unha transformación do sinal analóxico de entrada, de forma que o sinal pode amplificarse en caso de que sexa necesario, faise un filtrado para eliminar frecuencias non desexadas, e finalmente realízase un paso do sinal eléctrico a dixital, para que poida ser lido por unha computadora. Na figura 2.6 pódese ver o proceso que se leva a cabo de forma visual. Tamén cómpre ter en conta a frecuencia coa que o sensor recibe datos, xa que os sinais continuos debe pasar a ser discretos para poderen ser procesados.

Os erros que poden producirse no funcionamento dun sensor poden clasificarse como sistemáticos ou aleatorios. O primeiro deles, o erro sistemático é aquel que pode ser rectificado cun axuste no sensor. Pola contra, o caso dos erros aleatorios, coñecidos xeralmente como ruído, dáse cando o sinal desexado de entrada se mestura con sinais indesexados, como por exemplo un ruído de fondo. Este último tipo de erro é o que máis se debe tomar en consideración, posto que non se solventa cun axuste no sensor, se non que se debe investigar a fonte

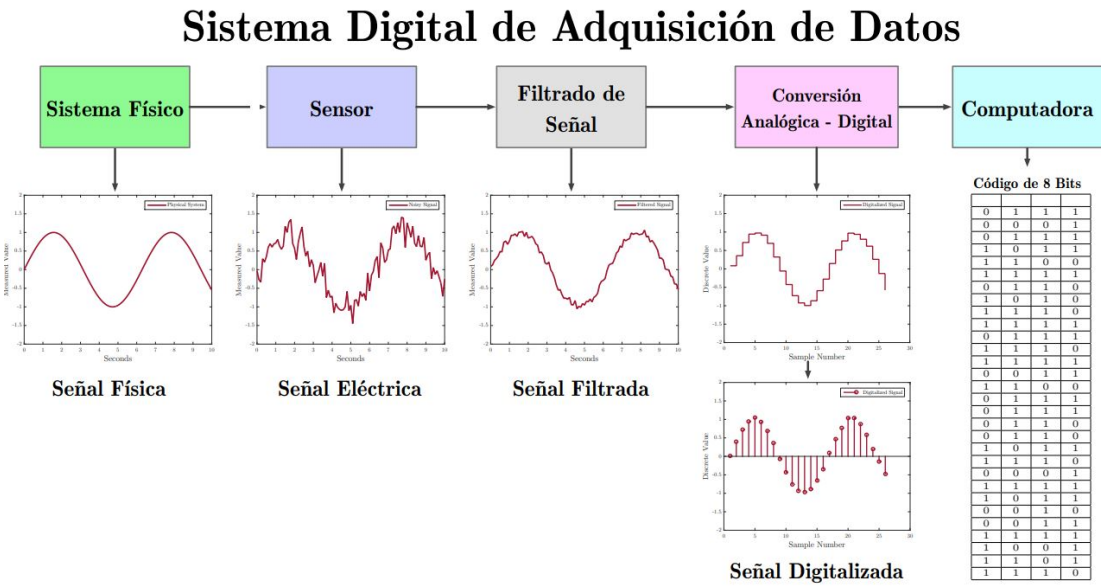

Figura 2.6: Proceso de transformación dun sinal analóxico a un sinal dixital.

e características da fonte de ruido.

En relación aos erros sistemáticos, os sensores deben ser calibrados para funcionar de forma correcta. Así, débense aplicar unha serie de datos de entrada coñecidos, e observar a continuación a resposta do sensor, axustándoo en consecuencia. Un exemplo de desaxuste dun sensor serían as entradas interferentes [13], que son unha superposición lineal sobre a variable de entrada. Tamén se require de calibración no caso das entradas modificadoras [13]. Ambos desaxustes deben ser solventados para o adecuado funcionamento dos sensores. Para traballar con sensores, e en relación ao calibrado destes, débense coñecer dous conceptos fundamentais- O primeiro deles, *accuracy* ou exactitude é a capacidade dun instrumento de proporcionar resultados próximos ao verdadeiro valor. O segundo concepto, *precision* ou fidelidade, fai referencia á capacidade dun instrumento de ofrecer os mesmos resultados cando se realizan medicións da mesma cantidade e baixo as mesmas circunstancias. Na figura 2.7 explícanse estes conceptos graficamente.

2.3.2 Series temporais

Chámase serie temporal [14] a un conxunto de datos secuencial, medido de forma sucesiva. Matematicamente defínese como un conxunto de vectores $x(t), t = 0, 1, 2...$ onde t representa o tempo. A variable $x(t)$ trátase como unha variable aleatoria. As medidas que se toman durante un evento nunha serie temporal son ordenados en orde cronolóxica. Unha serie temporal que só conten unha variable chámase univariable. Pola contra se ten máis dunha variable, denomínase multivariable. As series temporais poden ser continuas, onde as

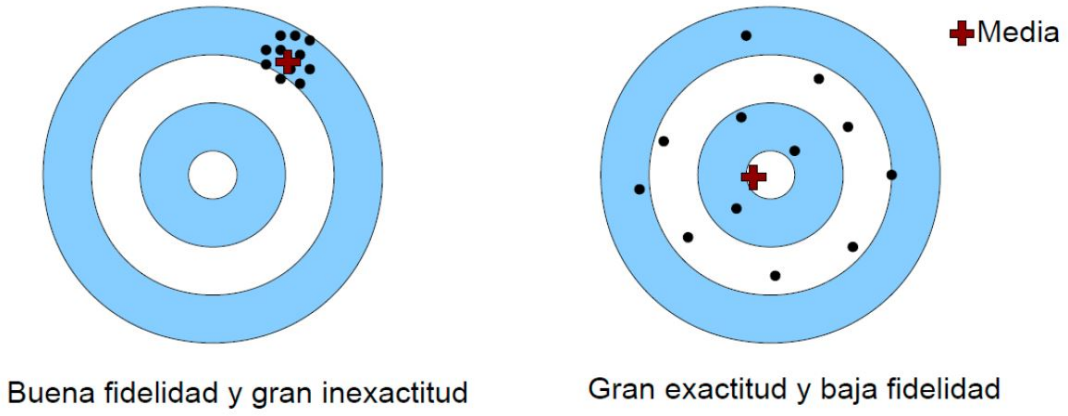

Figura 2.7: Exemplo de exactitude e fidelidade.

observacións se fan en cada instante de tempo, ou discretas, onde as observacións se realizan en puntos discretos do tempo.

Existe unha vinculación entre estas series e os sensores, que se manifesta seguindo unha serie de ecuacións. Mediante estas, establécese unha relación entre a serie temporal de entrada e as respostas que emite o sensor. Para este proxecto, a tipoloxía de respostas relevantes son os sistemas de orde cero e os sistemas de primeira orde. As series temporais relacionanse cos sensores seguindo unha serie de ecuacións, que vinculan a serie temporal de entrada coa resposta que emite o sensor. A tipoloxía de respostas relevantes para este proxecto son os sistemas de orde cero e os sistemas de primeira orde. Un sistema de orde cero é aquel no que a súa relación entre a entrada e a saída e a seguinte:

$$y(t) = k \cdot x(t) \Longrightarrow \frac{Y(s)}{X(s)} = k \tag{2.1}$$

, onde y(t) é o valor da lectura, x(t) é o valor de entrada do sistema, Y(s) é a saída do sistema no instante s, X(s) é a entrada do sistema no instante s, e k é unha constante lineal que corresponde coa sensibilidade do sistema. Os sistemas de orde cero ofrecen a resposta máis desexable nun sistema, xa que hai unha relación directa entre a entrada e saída. Isto implica que non hai retardo dende que se detecta ata que se transmite o sinal, e que o comportamento do sensor é independente da velocidade á que varíe a entrada. Pola contra, nos sistemas de primeira orde a relación entre entrada e saída é a seguinte:

$$a_1 \frac{dy(t)}{dt} + a_0 y(t) = x(t) \Longrightarrow \frac{Y(s)}{X(s)} = \frac{k}{\tau s + 1} \tag{2.2}$$

, onde a_k son coeficientes constantes coñecidos, $k = 1/a_0$ é a sensibilidade estática e

$\tau = a_1/a_0$ é a constante de tempo. Neste caso, o retardo e o erro dinámico dependen da forma do sinal de entrada.

2.3.3 Datos de sensórica de arrastre

Os datos cos que se conta son a peza clave que fai posible este traballo de investigación. Por seren estes fundamentais é imprescindible comprender tanto á súa orixe como as súas características e atributos. Para a adquisición de datos, realizáronse entrevistas informais a tres expertos do campo da pesca do arrastre. Nelas, ademais do coñecemento xeral sobre o campo, dous destes expertos proporcionaron datos informáticos dos seus traballos. Un deles proporciona datos referentes ás localizacións xeográficas onde se realizan os lances, ademais da velocidade do barco neses instantes e outros tipos de información procedente do GPS. Desafortunadamente, estes datos non aportan utilidade para o proxecto concreto que se quere desenvolver, pero si nos axudan a comprender mellor como e a forma de traballar neste sector. O outro dos informantes proporciona unha gran cantidade de series temporais procedentes de distintos sensores situados no aparello en barcos de arrastre. A información procede de barcos que traballan en localizacións diferentes, o que permite evitar nesgos sobre a localización. Estes datos tamén proceden de distintas compañías de pesca, pero todos os datos fornecidos proveñen de sensores fabricados pola mesma empresa.

Os datos que finalmente foron utilizados, aqueles que son series temporais, escolléronse por seren válidos para acadar os obxectivos deste proxecto. Estes están orixinalmente en arquivos propios do software de lectura de sensores, un por cada lance. Podense transformar a texto plano empregando ese mesmo software. En cada un destes arquivos de texto, hai unha táboa de datos por cada sensor que ten o barco. Estes sensores poden medir o seguinte: ángulo das portas a babor e estribor, distancia entre as portas, profundidade, se hai capturas ou non e cabeceo das portas. Na figura 2.8 podemos ver o formato dunha destas táboas. Debido a que non todos os barcos dispoñen de todos os sensores, só se escollen os datos de abertura e profundidade, para obter a maior xenericidade posible. En cada táboa, temos os seguintes atributos: código do sensor, hora, profundidade, datos1, datos2, latitude, lonxitude, sonda, comentario. A meirande parte das táboas só enchen os atributos de código de sensor, hora, profundidade e os datos que correspondan a ese sensor. Na figura 2.9 amósanos unha gráfica de como se ven os datos sen procesar, resaltando os problemas previamente mencionados.

É importante resaltar que estes datos contan con tres problemas que nos afectaran á hora do procesado. O primeiro deles é que hai datos nulos en case todos os lances. Sobre estes datos hai que tomar a decisión de como interpretalos, sendo algunha destas opcións repetir o dato anterior, asumir que e un cero, etc. O segundo dos problemas que presenta este conxunto de datos é que os datos non se reciben en un tempo determinado, se non que se van recibindo segundo se envían. Tamén debemos tomar unha decisión sobre como normalizar os datos

```
File Created: 13-nov-2019 18:57:49
Tow Date: 02-mar-2015 4:51:45


Transponder: E16G
Placement: Estribor

Code,Tiempo,Escalas(m),Datos1(°),Datos2(m),Latitude,Longitude,Sonda(m),Comment
E16G,04:51:48,98,2,14,1,---,---,---,---,
E16G,04:51:51,---,9,1,---,---,---,---,
E16G,04:52:18,211,2,---,---,---,---,---,
E16G,04:52:44,---,17,4,---,---,---,---,
E16G,04:53:08,410,9,29,6,---,---,---,---,
E16G,04:53:30,---,35,1,---,---,---,---,
E16G,04:53:56,619,0,---,---,---,---,---,
E16G,04:54:22,---,25,3,---,---,---,---,
E16G,04:54:45,805,4,---,---,---,---,---,
E16G,04:55:06,---,---,---,---,---,---,
```

Figura 2.8: Cabeceira dunha táboa de datos do ángulo dunha das portas.

para normalizar os instantes de tempo. Por último, as veces os patróns non indica o final do lance no programa de recollida de datos, polo que pode haber varios lances nun mesmo ficheiro de texto. As decisións tomadas con respecto a estes problemas trataranse máis en profundidade no capítulo 4.

2.4 Formulación global

O principal obxectivo é obter un sistema que nos proporcione unha predición sobre a serie temporal da abertura das portas. No seu funcionamento, este sistema obterá os datos sensóricos da abertura das portas durante o lance, cos cales predí cales serán os seguintes valores da serie de datos, tanto a curto como medio prazo. Grazas a esta predición, o patrón obtén unha información a priori sobre o estado do aparello da que actualmente non se dispón. A toma de decisións que fai un patrón durante o seu traballo vese mellorada debido á predición que proporciona este sistema. Os datos amósanselle ao patrón en forma de gráfica temporal a través dunha aplicación web, de xeito que poida ver as saídas do sistema facilmente. Na figura 2.10 podemos ver un exemplo esquemático do sistema.

2.5 Traballo proposto

Unha vez exposto o contexto da investigación, pódese concretar en máis detalle que se pretende acadar con este proxecto.

Para acadar este sistema, primeiramente débese obter un modelo de aprendizaxe máquina

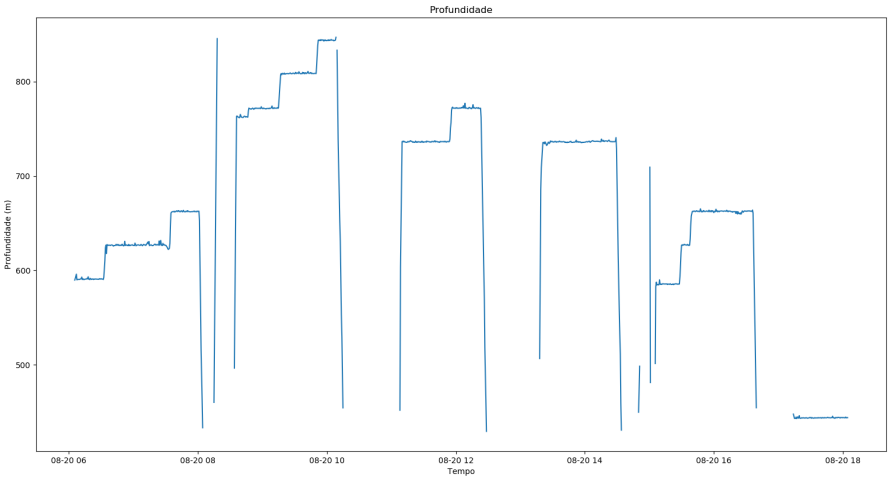

Figura 2.9: Datos da profundidade dun arquivo de texto antes de procesalo. Pódese ver que son en realidade varios lances, e que contén datos inválidos.

que permita facer unha predición sobre as series temporais procedente dos sensores dos barcos. Seguidamente débese facer unha análise exhaustiva dos datos iniciais, de forma que se coñezan todas as posibilidades que poidan dar e as súas limitacións. Tras isto, débense procesar estes datos, de forma que queden libres de datos erróneos e se extraian deles as características necesarias para o posterior adestramento dos modelos de aprendizaxe. Despois disto, farase unha escolla de modelos de regresión de aprendizaxe máquina para sensores baseada na literatura científica dispoñible. Mediante un proceso de hiperparametrización, seleccionaranse tres modelos: regresión lineal, bosques aleatorios e máquinas de soporte vectorial, e de cada un moitas das súas posibles configuracións. Cada unha destas configuracións adéstrase e compróbase a súa eficacia con respecto ao resto de configuracións. Con isto pódese obter o modelo que mellor funcione para predicir series temporais destas características, e así obter unha predición desta serie a diversos puntos temporais.

Deseguido, situarase o foco en acadar un servizo web REST para facer operacións CRUD cos datos de series temporais xa procesados, e unha operación de predición co mellor modelo obtido na investigación sobre os datos subidos. Para isto débese obter primeiro un deseño de como será esta API mediante un documento OpenAPI. Con este documento podemos xerar un esqueleto do servidor, polo que só resta implementar a lóxica do servidor. É necesario despregar neste servidor tantos modelos de predición como puntos de predición no futuro haxa. Por último, deséxase ter unha vista que dea usabilidade a todo o anterior. Para isto desenvolverase un cliente que permita ao usuario as operacións CRUD sobre o servizo REST, e ademais amose nunha gráfica os datos dos que se dispoñen e a predición obtida sobre eses datos. Esta

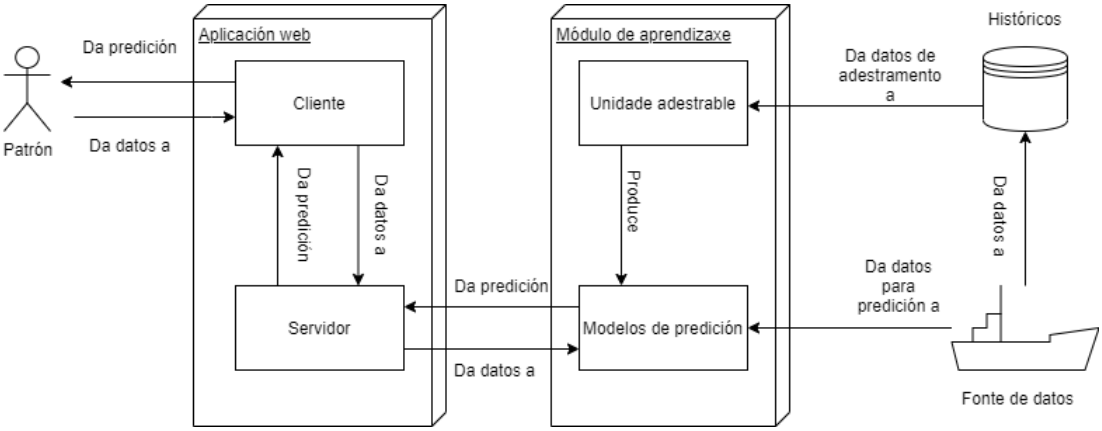

Figura 2.10: Esquema xeral da arquitectura do sistema xunto coas entradas e saídas deste.

gráfica será en vivo, de forma que se actualice periodicamente amosando os datos conforme se actualizan.

Capítulo 3

Planificación e metodoloxía

Neste capítulo da memoria exponse todo o relacionado coa planificación deste traballo. Ademais, detállase que recursos foron necesarios para o seu desenvolvemento e que custos supuxeron.

3.1 Metodoloxías

Unha metodoloxía é un conxunto de métodos que se seguen durante o desenvolvemento dun estudo ou traballo. É relevante definir unha metodoloxía antes de comezar o estudo para que se minimicen os erros no traballo que se está a realizar, levando o proxecto a ter unha maior calidade.

Por mor das características desta investigación, requírense dúas metodoloxías diferentes e complementarias para o seu desenvolvemento. Como o proxecto ten dúas partes moi diferenciadas, decídese que a mellor opción é desenvolver a primeira parte relacionada coa investigación en métodos de predición de sensórica cunha metodoloxía diferente á que se emprega no desenvolvemento web.

Para a primeira parte, considérase que a mellor opción é empregar unha metodoloxía incremental e iterativa. Esta metodoloxía permite realizar un seguimento continuo do proxecto, facilitando coñecer o progreso do traballo en cada momento. Inclúense reunións periódicas e metas definidas por quincenas. En cada unha destas reunións valórase o estado do proxecto, analizando os avances feitos así como posibles melloras, podendo incluso, a raíz destas reunións, cambiar a dirección do proxecto. Nas primeiras iteracións do traballo é habitual necesitar grandes cambios, polo que debe haber certa flexibilidade en canto a mudar de obxectivos. Ademais da flexibilidade que permite os cambios de rumbo necesarios nos estadios temperás do desenvolvemento, tamén posibilita o seguimento dos progresos en fases máis avanzadas. Por este motivo, esta metodoloxía adoita ser a máis habitual no desenvolvemento de investigacións en aprendizaxe máquina, debido a que se adapta ao seu carácter experimen-

tal e exploratorio.

Por outra banda, para o desenvolvemento web, úsase unha metodoloxía, tamén incremental e iterativa, pero coa diferenza de que se emprega un desenvolvemento guiado por probas [15]. Isto quere dicir que, antes de facer o código dunha funcionalidade, realízase o test que debe superar, o que axuda a obter un código que se axusta ao cumprimento dos requisitos funcionais do proxecto. Tamén se emprega nesta parte un taboleiro *kanban* para optimizar a xestión do traballo. Esta metodoloxía tamén contará con reunións quincenais, nas cales se analizan os progresos acadados e se propoñen os seguintes pasos a realizar. Esta metodoloxía adáptase mellor ao desenvolvemento web, que conta cun carácter menos cambiante que o propio dunha investigación en aprendizaxe máquina. Este carácter permite ademais realizar unha definición inicial de características concretas necesarias para o proxecto, polo que teremos unha planificación máis ríxida nesta parte.

Considérase que esta aproximación híbrida é a mellor opción para afrontar o problema que nos atinxe, xa que nos proporciona o marco de traballo axeitado para cada parte do proxecto e contribúe á adquisición dos obxectivos formulados.

3.2 Tarefas

Neste apartado explícase detidamente a planificación e posta en marcha do proxecto proposto, que se divide principalmente nunha fase inicial de investigación, unha fase na que se plasma o resultado da investigación no desenvolvemento dunha aplicación web e finalmente unha última etapa na que se redacta o proceso en forma de memoria.

3.2.1 Fase de investigación

Durante esta fase descríbense todos aqueles pasos relacionados coa obtención do modelo de aprendizaxe máquina que permita facer predicións sobre os datos sensóricos dos barcos. Non só se inclúe aquí o proceso de desenvolvemento en si mesmo, senón que tamén se realiza un estudo do dominio e unha recolección dos datos necesarios para solventar o problema. Podemos dividir esta etapa nas seguintes fases:

1. **Estudo do dominio**: durante esta subfase estúdase o dominio do traballo en profundidade. Primeiramente, faise unha revisión da literatura científica da que se dispón na actualidade sobre este tema, extraendo dela a información que pode ser útil para o desenvolvemento. Deseguido, realízanse entrevistas informais con varios expertos do sector, que axudan a comprender mellor o ámbito do arrastre na práctica. Ademais, nas conversas con estes profesionais revélase que programas informáticos se manexan no sector, de tal maneira que axudan a facer unha revisión previa do estado da arte sobre

as aplicacións do sector pesqueiro. Tamén se estuda neste punto o propio problema do que se parte, ideando e investigando posibles solucións.

2. **Recolección de datos**: despois dunha primeira achega ao dominio, é necesario conseguir os datos precisos para a realización do traballo. Para conseguir este obxectivo, investíganse as posibles fontes de datos dispoñibles e valórase con quen contactar para obtelos.

3. **Limpeza de datos e preprocesado**: é neste punto cando, unha vez obtidos os datos, estes son procesados de xeito que se garanta que non conteñen erros e están preparados para seren introducidos en modelos de aprendizaxe máquina.

4. **Extracción de características**: nesta etapa estúdase cales son as características que se poden extraer dos datos para utilizar e conseguir as mellores predicións.

5. **Escolla de algoritmos e construción de modelos**: seleccionadas as características, procédese a escoller os algoritmos de aprendizaxe máquina que se van empregar en base á literatura científica revisada e ó coñecemento adquirido sobre o campo de traballo. Unha vez seleccionados estes algoritmos, adéstranse cos datos previamente recollidos.

6. **Avaliación dos modelos obtidos**: por último, na etapa final desta fase, compáranse todos os modelos obtidos coa finalidade de seleccionar aquel co mellor rendemento para dar solución a este problema.

3.2.2 Fase de desenvolvemento web

Unha vez concluídas as etapas da fase de investigación, procédese á fase de desenvolvemento web, que engloba as tarefas necesarias para o deseño e implementación da aplicación que dará usabilidade ao modelo de aprendizaxe máquina escollido. Nesta etapa do traballo seguiranse os seguintes pasos:

1. **Deseño do software**: neste punto realízase un deseño xeral, a grandes trazos, de como será o cliente e máis o servidor. Escóllense nesta altura os padróns que se van empregar durante o desenvolvemento desta segunda fase da investigación.

2. **Implementación do servidor**: despois do deseño inicial, implementaranse tanto os tests coma a lóxica que serve de base para a nosa aplicación.

3. **Implementación do cliente**: finalmente, impleméntase o cliente web que permite usar o servizo previamente implementado, así como os tests necesarios.

4. **Probas**: Pásanse os tests realizados previamente, asegurándose que o código implementado cumpre os requisitos funcionais.

3.2.3 Realización da memoria

Tras facer as tarefas necesarias para acadar o obxectivo principal do proxecto, deixarase constancia por escrito de todo o proceso de investigación, do traballo vinculado coa aprendizaxe máquina e da realización da aplicación web. En último lugar, farase unha avaliación do proceso en conxunto que analice tanto os resultados obtidos coma aqueles puntos febles que poderían mellorar de cara a un traballo futuro.

3.3 Planificación

Para cada unha das tarefas enumeradas no apartado anterior, faise unha estimación da carga de traballo que supoñen, empregando o tempo que pode tardar en facela unha persoa como medida. Esta estimación represéntase a través dun diagrama de Gantt, que podemos ver na figura 3.1.

Nesta planificación tivéronse en conta certos riscos. O máis importante deles é a fase inicial de estudo do dominio, xa que este determina o rumbo que leva a totalidade do traballo. Por este motivo, estímase un tempo moi amplo para esta tarefa. Outro dos riscos reside na recollida de datos, xa que pode que a información obtida non sexa válida, obrigando a procurar outras fontes diferentes. Afortunadamente, se se dera este caso, a busca doutra fonte de datos non requiriría dunha gran cantidade de tempo. Aínda así, esta posibilidade tense en conta na planificación temporal.

3.3.1 Seguimento

As tarefas referentes á fase de investigación axustáronse debidamente á planificación inicial. Porén, no mes de xuño debido a un imprevisto de carácter persoal, a realización do proxecto quedou en pausa. Por mor disto, o resto do proxecto retrasouse aproximadamente un mes, obrigando a reaxustar o resto de tarefas ao tempo restante. Isto resultou en que certas tarefas se tiveran que realizar en un tempo moito máis reducido do planificado inicialmente, repercutindo particularmente na realización do cliente web.

3.4 Recursos

Para realizar este proxecto foron necesarios diferentes tipos de recursos, que cómpre detallar tanto para describir as características deste tipo de traballo como para estimar os custos da investigación.

3.4.1 Recursos materiais

En canto aos recursos materiais necesarios para realizar este proxecto, foron precisos tanto recursos físicos coma recursos software. Os recursos físicos foron os seguintes:

- Portátil Asus R510VX
  - Procesador: Intel® Core™ i5-6300HQ Quad-Core (6M Cache, 2.3GHz ata 3.2GHz)
  - Memoria RAM: 8GB
  - Sistema operativo: Ubuntu 18.04 LTS
  - Tarxeta gráfica: NVIDIA® GeForce® GTX950M
  - Almacenamento: SSD 240GB
- Servidor Amazon Web Services Elastic Compute Cloud
  - Memoria RAM: 8GB
  - Sistema operativo: Ubuntu 18.04 LTS
- Conxunto de datos de series temporais procedentes de sensores de barcos de arrastre.

No ordenador Asus fíxose a práctica totalidade do traballo, excepto os adestramentos dos modelos de aprendizaxe máquina, que se fixeron nunha instancia creada en Amazon Web Services. Esta instancia é unha máquina virtual especializada en problemas de aprendizaxe máquina, que fai que o adestramento sexa moito máis rápido ca nunha computadora persoal. Seguidamente, os datos obtidos nas misións de campo non supuxeron ningún custo para este proxecto, xa que os colaboradores dispoñían deles nos seus bancos de datos. O resumo dos custos exponse na táboa 3.1.

Recurso	Custo
Portátil ASUS	600€
Amazon Web Services	0*
Datos	0

Táboa 3.1: Táboa cos custos dos recursos materiais. *O custo de amazon web services é gratuíto grazas ao crédito de estudantes proporcionado.

En canto aos recursos software, foron necesarios os seguintes elementos:

- PyCharm Community 2018
- scikit-learn 0.23.1

- Python 3.7.2
- Visual Studio Code 1.43.7

PyCharm foi requirido para a implementación do código necesario na investigación e do servidor web. En Visual Studio implementouse o cliente web. Todo o software utilizando foi de uso gratuíto.

3.4.2 Recursos humanos

En canto aos recursos humanos, asúmense os roles de xefe de proxecto, programador e investigador. O primeiro será asumido polo director da investigación, cuxo traballo se manifesta principalmente nas reunións periódicas de supervisión e avaliación en todo o curso do proxecto. Os outros dous roles, programador e investigador, serán asumidos neste caso polo alumno. Os custos relacionados cos recursos humanos son detallados na táboa 3.2

Rol	Salario	Horas Home	Custo total
Director	19.24€/h	25h	481€
Investigador	11.85€/h	200h	2.370€
Programador	14.36€/h	200h	2.873€

Táboa 3.2: Custos dos recursos humanos necesarios para o proxecto. A información sobre os salarios de cada posto provén de [1]

3.4.3 Resumo dos custos

En total, os custos requeridos para a realización deste proxecto serían, aproximadamente, os presentados na táboa 3.3

Tipo recurso	Custo
Materiais	600€
Humanos	5.724€
Total	6.324€

Táboa 3.3: Táboa resumo dos custos do proxecto.

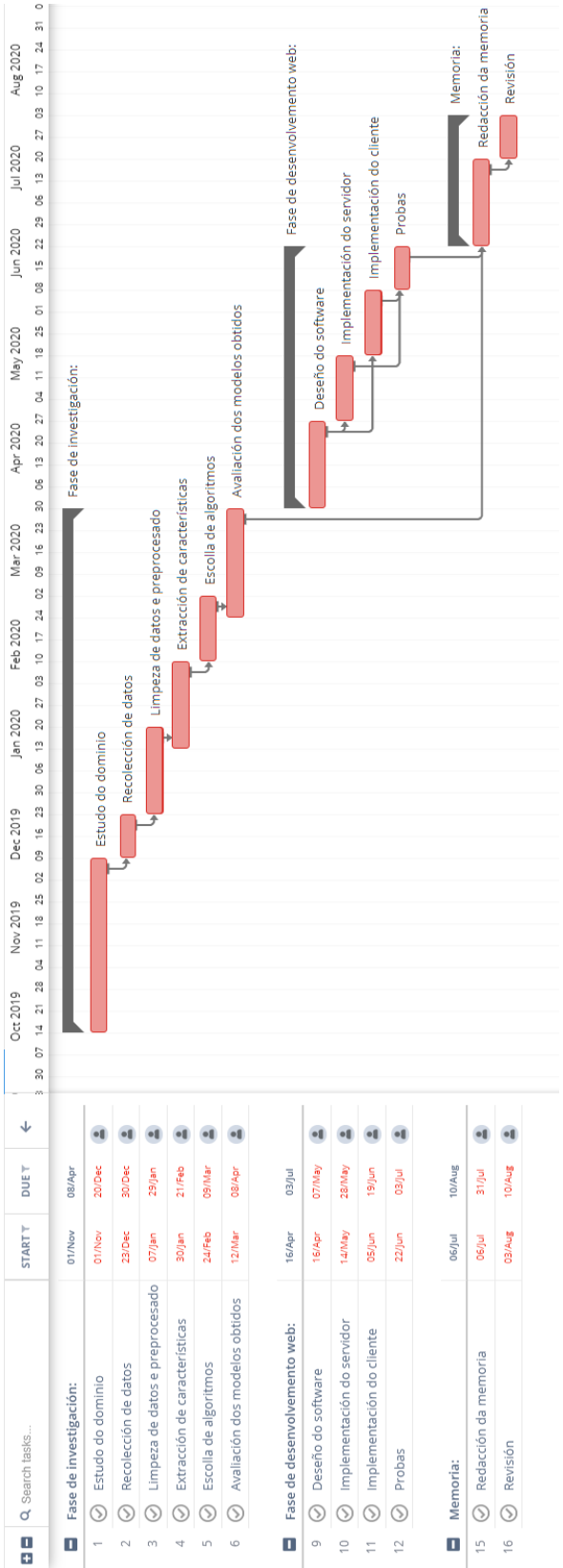

Figura 3.1: Diagrama de Gantt coa liña base do proxecto

Capítulo 4

Análise de datos

Nesta sección da memoria detállase o tema central de interese deste traballo: o proceso levado a cabo dende a adquisición do coñecemento sobre o dominio ata a proposta dunha solución ao problema inicial. Estúdanse primeiramente os fundamentos tecnolóxicos necesarios no traballo para pasar posteriormente a expoñer, fase por fase e detalladamente, o traballo levado a cabo para acadar os obxectivos propostos.

4.1 Resumo

Primeiramente expóñense os conceptos teóricos referentes a aprendizaxe máquina necesarios para poder entender o núcleo deste traballo. Seguidamente, explícanse as razóns polas que se escolle *python* para a investigación, así como o motivo da escolla da libraría *scikit-learn*. No seguinte apartado proponse o sistema de aprendizaxe máquina que se quere desenvolver, xunto cun esquema xeral deste. Seguidamente explícase detidamente como, mediante entrevistas informais a expertos do sector, foron adquiridos tanto o coñecemento necesario coma os datos que se empregan no traballo. Tras isto, explícase o procesado destes datos, detallando cada paso da limpeza dos valores inválidos así coma as transformacións precisas. Na seguinte sección explícase o motivo de por que se seleccionaron os modelos de bosques aleatorios, regresión lineal e máquinas de soporte vectorial. Na sección de adestramento, amósase o proceso levado a cabo para realizar o adestramento e a comparación de resultados respecto dos erros de test entre os distintos modelos. Por último, faise unha análise dos resultados e expóñense as conclusións da investigación.

4.2 Fundamentos tecnolóxicos

Este apartado está dedicado a explicar certos conceptos teóricos necesarios para a realización desta investigación, tamén se explican as tecnoloxías que se requiren para levar a cabo

o propio proxecto.

4.2.1 Aprendizaxe máquina

Unha posible definición de aprendizaxe máquina [16] é que esta consiste nunha serie de métodos que detectan automaticamente padróns nun conxunto de datos, coa fin de usar estes padróns para predicir datos futuros ou realizar outros tipos de decisións en situacións incerteza. A aprendizaxe máquina subdivídese en dous grupos, en función do obxectivo para o que se utilice.

Aprendizaxe supervisada

Na aprendizaxe supervisada [16], o obxectivo é asociar unhas entradas X a unhas saídas y, dado un conxunto de pares de entradas-saídas $D = (x_i, y_i)_{i=1}^n$. Este conxunto denomínase conxunto de adestramento, onde N é o número de mostras. As entradas x_i son un vector no que cada compoñente se corresponde cunha característica. Xeralmente, as características represéntanse con números, aínda que poderían ser obxectos complexos, como imaxes, frases, etc. Por outra banda, a saída, ou variable resposta y_i, pode ser calquera tipo de información. Con todo, habitualmente as saídas son, ou ben unha categoría dentro dun conxunto finito, correspondéndose deste xeito o problema cun de clasificación, ou ben un valor real, no que o problema se correspondería cun de regresión.

Para a clasificación, as saídas corresponden a un conxunto $y \in \{1, ..., C\}$, sendo C o número de clases. Se $C = 2$, o problema será de clasificación binaria. Se $C > 2$ o problema será de clasificación multiclase, e se as etiquetas non son mutuamente exclusivas, denominarase clasificación multietiqueta. As clasificacións multiclase son tratadas como varios problemas de clasificación binaria, sendo as aproximacións usadas as dun contra todos, onde se clasifica se pertence a unha clase determinada ou no, ou a de un contra un, onde se comparan as entradas por pares.

O obxectivo deste subtipo de métodos é facer predicións en entradas nunca antes vistas, xa que predicir sobre o conxunto de adestramento non sería de interese debido a que non aportaría información nova.

Para a regresión, o problema é esencialmente o mesmo, excepto que a variable resposta pasa a ser continua no canto de ser discreta. A regresión é especialmente interesante para acadar os obxectivos formulados nesta investigación.

Aprendizaxe non supervisada

Neste tipo de aprendizaxe, só se traballa cos datos de entrada, sen dispoñer das saídas desexadas. O obxectivo é descubrir estruturas nos datos, motivo polo cal tamén se coñece

como descubrimento de coñecemento. Ao contrario que na aprendizaxe supervisada, non se sabe cal é a saída desexada, se non que se fai unha estimación da densidade [16]. Isto quere dicir que se estima sobre as saídas cales delas poden estar relacionadas e cales non en función da súa proximidade. O exemplo por antonomasia de aprendizaxe non supervisada é o problema da agrupación de datos.

Dimensionalidade

Cando se traballa con conxuntos de adestramento, pode darse o caso de que estes teñan unha alta dimensionalidade, é dicir, que os vectores teñan un alto número de características. Adoita ser útil, se este é o caso, reducir a dimensionalidade dos datos, de forma que obteñamos a parte esencial destes, descartando aquelas partes redundantes ou superfluas. Esta redución da dimensionalidade tamén pode permitir a visualización de datos de alta dimensionalidade. Un dos métodos máis comúns para realizar esta operación e a técnica de análise de compoñentes principais.

Validación cruzada

Unha vez obtidos os modelos, débense comparar entre eles para poder escoller aquel que mellor se adecúe á resolución do problema. Para realizar esta selección, cómpre contar cun conxunto de datos que non foran utilizados durante o adestramento do sistema, e que se coñecen como conxunto de validación. A partir destes, que se presentan en forma de entrada-saída, pásase polo modelo só a entrada e este predí a saída. A comparación entre a saída predita e a real serve para obter o nivel de eficacia de cada modelo.

Unha forma de avaliación que posibilita comprobar a validez dun modelo para evitar nesgos no adestramento é a validación cruzada. Esta consiste en particionar o conxunto de datos en K partes. Despois, seguindo a filosofía *leave one out*, adéstrase o modelo con todas as partes excepto unha delas, e así iterativamente realízase este procedemento con cada subconxunto [17]. Na figura 4.1, explícase o proceso. Este proceso de validación cruzada permite sacar maior partido aos datos dispoñibles para acadar unha estimación do desempeño dos sistemas candidatos. Neste sentido, a distribución dos erros de validación para cada unha das iteracións permite facer unha estimación robusta do erro esperado.

Teorema ”No free lunch”

Este teorema explica que non hai un modelo universal que funcione para todos os problemas [18]. Isto débese a que un conxunto de asuncións válidas para un dominio poden non funcionar noutro distinto. A falta dun modelo único, o uso da técnica da validación cruzada permite escoller empiricamente o mellor modelo posible.

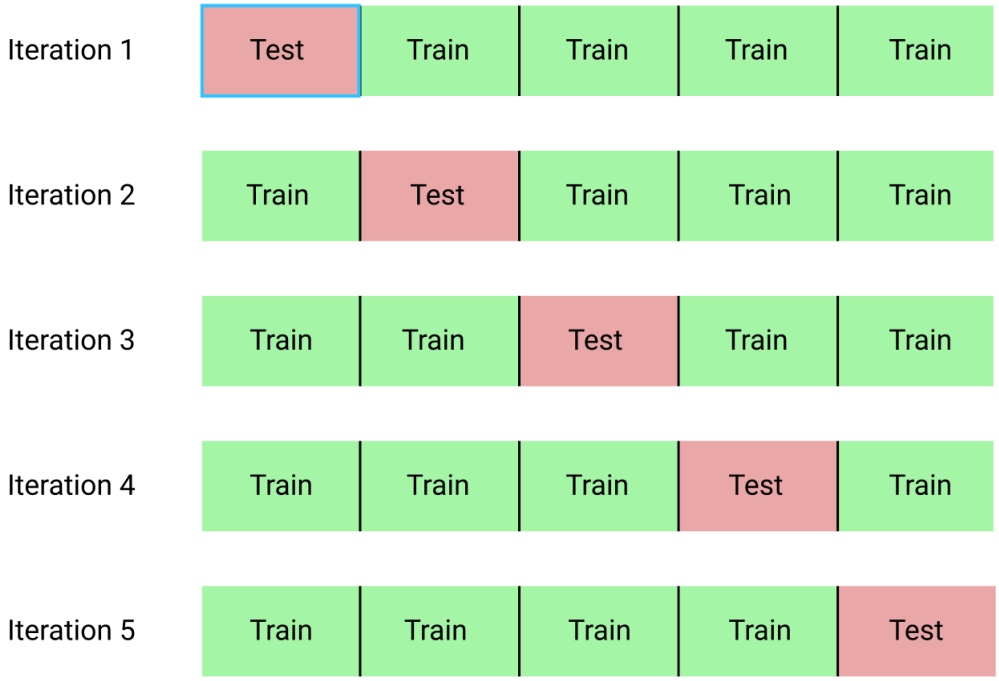

Figura 4.1: Exemplo de validación cruzada con 5 partes.

Xanelado

En relación coa regresión, outra noción relevante é o xanelado. Nos sets de datos de series temporais, resulta complicado normalizar o conxunto, xa que no modelo a entrada sempre debe ser un vector de X características e y saídas, e os sets de datos reais adoitan ser de lonxitudes variables. Unha das posibles solucións é xanelar a serie temporal, de forma que se escollen fraccións de N características e M saídas do conxunto orixinal. Estas fraccións denomínanse xanelas de tamaño fixo e desprázanse sobre o conxunto total tal e como está representado na figura 4.2. No problema que se trata cada xanela será un elemento do conxunto de adestramento.

Regresión lineal

A regresión lineal sinxela é un procedemento estatístico empregado para calcular o valor dunha variable dependente partindo do valor dunha variable independente. Este algoritmo mide a asociación entre estas dúas variables [19]. Unha extensión deste procedemento é a regresión lineal múltiple, que se diferencia da anterior pola característica de que existe máis dunha variable independente. Dende o punto de vista matemático, o modelo de regresión

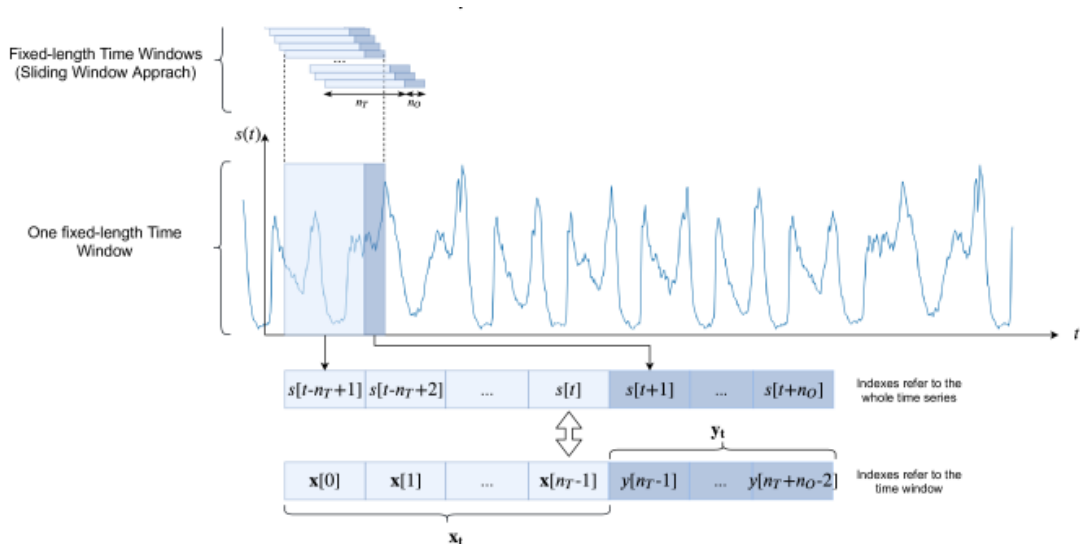

Figura 4.2: Xanelado con xanela fixa.

lineal sinxela [20] consiste en dúas ecuacións, sendo a primeira delas a función media:

$$E(Y|X = x) = \beta_0 + \beta_1 x, \tag{4.1}$$

, onde a intersección ou termo constante β_0 corresponde co valor de $E(Y|X = x) = x$ cando x é 0, e a pendente β_1 é a velocidade de cambio de $E(Y|X = x) = \sigma^2$ por cada unidade de cambio en X. A segunda das ecuacións é a función:

$$Var(Y|X = x) = \sigma^2, \tag{4.2}$$

, onde σ^2 é a varianza.

Os valores dos parámetros β_0 e β_1 adoitan ser descoñecidos e deben ser estimados usando valores. Estes parámetros son coñecidos como estimadores.

A función criterio empregada para obter os estimadores está baseada nos erros residuais, que son a distancia vertical entre a liña predita e o valor real de y. Os estimadores dos mínimos cadrados son os valores de β_0 e β_1 que minimizan a seguinte función:

$$RSS(\beta_0, \beta_1) = \Sigma_{n=1}^{n}[y_i - (\beta_0 + \beta_1 x_i)]^2 \tag{4.3}$$

As estimacións dos mínimos cadrados poden acharse coas seguintes expresións:

$$\hat{\beta_1} = \frac{SXY}{SXX} = r_{xy}\frac{SD_y}{SD_x} = r_{xy}\left(\frac{SYY}{SXX}\right)^{1/2} \hat{\beta_0} = \bar{y} - \hat{\beta_1}\bar{x}, \tag{4.4}$$

onde r_xy é o coeficiente de correlación entre x e y, SXX o sumatorio de x_i^2, SXY o sumatorio

de $x_i y_i$ e SD é a varianza da mostra. A varianza σ^2 é basicamente a media cadrática do erro residual. É esperable que $\hat{\sigma}^2$ se obteña coa media dos erros residuais ao cadrado. A estimación de σ^2 é dada pola división da suma dos erros residuais cadráticos $RSS = \Sigma \hat{e}_i^2$ entre os seus graos de liberdade, onde os graos de liberdade residuais son o número de casos menos o número de parámetros na función da media. Para a regresión sinxela, o grao de liberdade é n - 2, polo que a estimación sería a seguinte:

$$\hat{\sigma}^2 = \frac{RSS}{n-2}, \tag{4.5}$$

onde RSS denomínase erro residual medio cadrático e é de utilidade para comparar a efectividade de distintos modelos.

Bosque aleatorio

Unha árbore de decisión [21] é un tipo de clasificador que está formado por nodos. Existen tres categorías de nodos: o nodo raíz, os internos e os terminais. O primeiro destes caracterízase por ter ningún nodo entrante, só pode haber un nodo raíz en cada árbore. O segundo dos tipos, o nodo interno ou de test, defínese por ter un nodo entrante, e varios de saída. Por último, os nodos terminais ou follas, son aqueles que só teñen un nodo entrante e non teñen nodos saída. Na árbore de decisión, cada nodo interno subdivide o espazo, en dous ou máis subespazos de acordo co resultado dunha función discreta á que se lle aplican valores dos atributos de entrada. O caso máis sinxelo de árbores de decisión considera un só atributo, e o espazo da instancia é particionado en función deste. Cada folla é asignada a unha clase que representa o mellor obxectivo para eses atributos. Cada instancia é clasificada dende a raíz da árbore cara abaixo.

O modelo de bosques aleatorios, ou *random forest* [22] é unha especialización do concepto das árbores de decisión. A esencia deste modelo consiste en construír varias árbores en subespazos escollidos aleatoriamente dentro do espazo de características. As árbores de diferentes subespazos xeneralizan a súa clasificación de forma complementaria entre elas.

O algoritmo dos bosques aleatorios é o seguinte:

1. Sexa N o número de casos de proba, M é o número de variables no clasificador.
2. Sexa m o número de variables de entrada a ser empregado para determinar a decisión nun nodo dado. m debe ser moito menor que M.
3. Escoller un conxunto de adestramento para esa árbore, e usar o resto de casos de proba para estimar o erro.
4. Para cada nodo da árbore, escoller aleatoriamente m variables nas cales basear a decisión. Calcular a mellor partición do conxunto de adestramento a través das m variables.

Entre as súas vantaxes podemos destacar as seguintes [23]:

- É dos mellores modelos en precisión que existen.
- É eficiente con grandes cantidades de datos.
- Pode soportar miles de variables de entrada sen borrar ningunha.
- Da unha estimación de que importancia ten unha variable na clasificación.
- As árbores xeradas poden gardarse para seren empregadas con outros datos.

Todas estás características fan do algoritmo de bosques aleatorios unha ferramenta proveitosa para ter en conta nunha investigación coas características do proxecto ao que fai referencia esta memoria.

Máquinas de soporte vectorial

As máquinas de soporte vectorial [24] son un modelo de aprendizaxe supervisado, cuxo principal obxectivo é atopar un hiperplano óptimo que divida un conxunto de datos en dúas clases. A forma máis sinxela de máquina de soporte vectorial é a lineal, onde o hiperplano se atopa no mesmo espazo que os datos de entrada[25]. Neste caso, o espazo é un subconxunto de todos os hiperplanos da seguinte forma:

$$f(x) = w \cdot x + b, \tag{4.6}$$

onde w é a inclinación do plano e b é o intercepto. Existen infinitos hiperplanos que dividen a dúas clases. O modelo SVM debe obter aquel hiperplano que maximice a distancia entre el mesmo e os puntos máis próximos a el. Na figura 4.3 pódense ver exemplos de hiperplanos lineais.

As máquinas de soporte vectorial son unha técnica dispersa. Isto quere dicir que unha vez adestrado o modelo, estas máquinas só necesitan dun subconxunto do conxunto de adestramento, chamado vectores de soporte, para facer unha predición do futuro. Estes vectores de soporte definen as marxes dos hiperplanos, atopados despois da optimización.

Por outra banda, cómpre explicar que a complexidade dunha tarefa de clasificación non depende da dimensionalidade do espazo de entradas, senón do número de vectores de soporte. O número de vectores de soporte que se teñen en conta depende á súa vez da complexidade dos datos. Tanto a dimensionalidade como a separabilidade entre clases definen a complexidade destes [26]. Ademais, para engadir flexibilidade ao modelo, inclúese un parámetro C que permite dar certa flexibilidade cos erros de clasificación. Esta técnica denomínase *soft-margin*.

Tamén é relevante o feito de que as máquinas de soporte vectorial empreguen o método do kernel para mapear os datos a un espazo de alta dimensionalidade antes de resolver

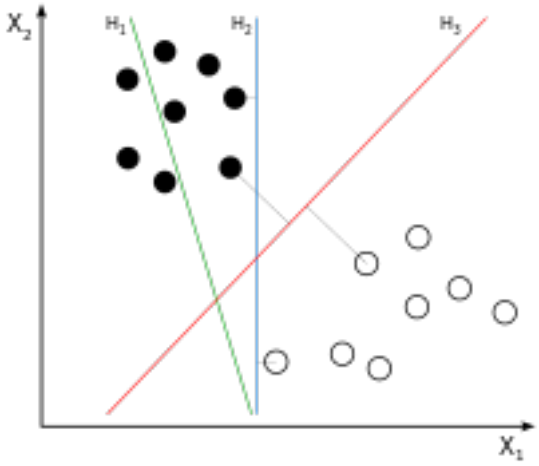

Figura 4.3: Comparación entre un plano que non separa as clases, un que si o fai e por último o plano que separa as clases de xeito óptimo.

o problema. Isto débese a que habitualmente os datos da vida real non son linearmente separables, polo que ao mapear os datos a unha maior dimensión empregando unha función de kernel predefinida pódese conseguir que as clases si sexan linearmente separables [26]. Algúns exemplos de funcións de kernel son os seguintes:

- Polinomial:

$$k(x_i, x_j) = (x_i \cdot x_j + 1)^d, \tag{4.7}$$

  onde d e o grao do polinomial.

- Función de base radial gaussiana:

$$k(x_i, x_j) = exp(-\gamma ||x_i - x_j||^2), \tag{4.8}$$

  onde $\gamma > 0$.

- Sigmoide:

$$k(x, y) = tanh(\alpha x^T y + c), \tag{4.9}$$

  onde C é o intercepto e α a pendente.

Ao seren as máquinas de soporte vectorial un algoritmo que non só busca a minimización do erro, senón que tamén maximiza a distancia entre o hiperplano e as clases, obtense unha mellor xeneralización, que resulta esencial cando o modelo se enfronta a datos nunca antes vistos [26].

4.2.2 Python para uso científico

Para o desenvolvemento da investigación escolleuse a linguaxe de programación *python*, debido aos seguintes motivos:

- É unha linguaxe sinxela de entender e de desenvolvemento rápido.
- As estruturas de datos son sinxelas de manexar, facilitando moito o manexo dos datos. Ademais conta coa libraría de pandas, que grazas as súas estruturas de datos en forma de táboas facilitan as operacións sobre os datos.
- Contén un gran número de librarías, e unha gran comunidade soportándoas, o que fai que as librarías estean moi ben documentadas e libres de erros.
- A libraría *Scikit-learn* na que se basean as implementacións deste traballo está dispoñible para *python*.

Todos estes motivos son máis que suficientes para a escolla desta linguaxe. Outras alternativas son a linguaxe de programación R, coa desvantaxe que está pensada para usar de xeito interactivo, o que a fai máis complexo de usar, mentres que python emprégase para desenvolver código. Tamén se pode empregar *matlab*, pero o carácter privativo deste dificulta o emprego por parte de estudantes, sobre todo á hora de acceder a librarías de pago.

4.2.3 Scikit-learn

A libraría *Scikit-learn* [27] aprovéitase do rico entorno que prové *python* para dar implementacións dos máis coñecidos algoritmos de aprendizaxe máquina. Responde á crecente necesidade de persoas non especializadas na industria do software de facer análise de datos. Estas son as principais vantaxes con respecto a outras librarías de aprendizaxe máquina porque:

- Está distribuída baixo a licenza BSD.
- Incorpora código compilado para mellorar a súa eficiencia.
- Só depende de *numpy* e *scipy* para facilitar a súa distribución.
- Enfócase na programación imperativa.

Neste proxecto é relevante destacar os principios de deseño de *Scikit-learn* [28], por mor da súa importancia para o desenvolvemento desta investigación. Estes principios son os seguintes:

- Prover ferramentas de aprendizaxe máquina que presenten unha interface consistente en todos os seus obxectos.

- Os parámetros tanto dos construtores coma dos valores determinados polos algoritmos de aprendizaxe son gardados e expostos a través de atributos públicos.
- Os algoritmos de aprendizaxe son os únicos obxectos representados como clases customizadas. Os datasets serán *arrays* de *numpy* ou matrices dispersas de *scipy*. Os hiperparámetros son representados con *strings* ou números na medida do posible.
- Moitas das tarefas de aprendizaxe máquina son secuencias ou combinacións de transformacións de datos. Algúns algoritmos son tamén vistos como meta algoritmos parametrizados en algoritmos. Cando sexa posible, estes algoritmos deben ser implementados e compostos de bloques de construción xa existentes.
- Se unha operación require dun parámetro definido polo usuario, un valor por defecto apropiado será definido pola libraría.

Estes principios fan desta libraría unha gran opción para o desenvolvemento de traballos de aprendizaxe máquina, principalmente pola súa sinxeleza de uso. Ademais a súa interface unificada facilita a implementación e comparación de diferentes modelos de aprendizaxe.

4.2.4 Amazon Elastic Comput Cloud

Amazon Web Services [29], ou AWS polas súas siglas, é unha colección de servizos de computación na nube. Un destes servizos é Amazon Elastic Compute Cloud, que permite aos usuarios dispoñer dun clúster virtual de ordenadores, dispoñibles todo o tempo. Este clúster emula a maioría dos atributos dun ordenador. O sistema de pagamentos está baseado nun modelo no que, no canto de pagar o servizo por meses ou anos, págase por tempo de uso. AWS proporciona un crédito para estudantes, grazas ao cal se poden usar os servizos de Amazon Elastic Compute Cloud de forma gratuíta para usos académicos.

Neste proxecto, AWS facilita o adestramento dos modelos de aprendizaxe, xa que externaliza fóra da computadora de uso habitual a alta carga de traballo que supón para un ordenador doméstico. Isto permite continuar co traballo mentres se está a realizar un adestramento. Ademais, ao seren máquinas especializadas neste tipo de tarefas, conséguense rematar os adestramentos rapidamente.

4.3 Sistema proposto

Para acadar o obxectivo de obter predicións sobre as series temporais procentes de barcos de arrastre, proponse un subsistema de adestramento que obteña os mellores modelos para varios puntos de predición. O subsistema de adestramento terá como entradas as series temporais históricas dispoñibles dos sensores de abertura e profundidade xanelados, xunto co

valor do punto de predición que se desexa obter no modelo en concreto. Con este subsistema obtemos os modelos que menos erro cadrático teñen con respecto aos datos reais. Unha vez obtidos os mellores modelos en cada punto de predición, pódense despregar para poñelos en funcionamento en casos reais. Nestes modelos xa adestrados, a entrada consiste nas series temporais da abertura e da profundidade, mais sen o punto de predición, xa que este vén dado polo propio modelo. No esquema visto na figura 4.4 podemos ver como é o sistema proposto a grandes trazos.

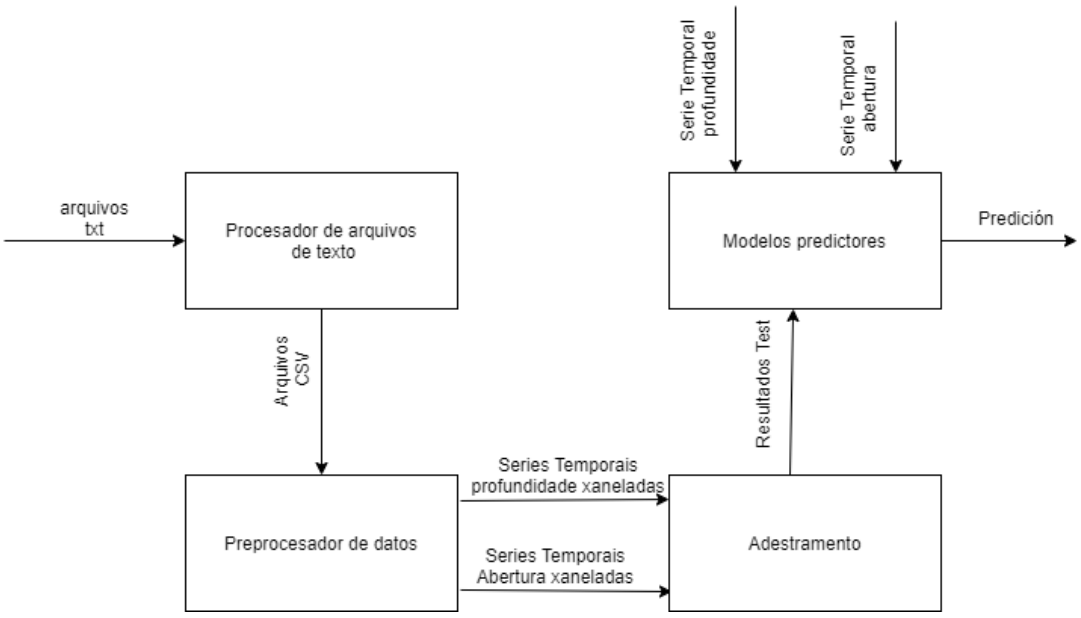

Figura 4.4: Esquema do subsistema do módulo de aprendizaxe.

4.4 Adquisición do coñecemento

O coñecemento do dominio máis relevante foi adquirido mediante entrevistas informais realizadas a tres expertos de distintas categorías dentro do sector: un patrón de costa en Galicia, cuxo rol consiste en facer as gardas do barco, un capitán de pesca en bancos pesqueiros africanos, e por último, un profesional do sector da venda de sensores para barcos de arrastre.

O primeiro dos entrevistados proporciona unha visión xeral do estado actual da pesca do arrastre na práctica. En xeral, fai comentarios sobre o proceso de lance, descrito no capítulo 2 no apartado sobre a pesca do arrastre. Ademais,tamén fala dos problemas cos que se encontra no seu traballo entre os que destaca o descoñecemento do estado do aparello durante o seu funcionamento. A única información da que dispón neste intre é a que dan os sensores, e a cantidade de potencia que o barco está a requirir para traballar. Para este informante, o óptimo

sería ter unha cámara no fondo mariño varios metros por diante do aparello para coñecer por onde vai a pasar este, mais esta solución resulta moi custosa e difícil de implementar. A aproximación deste traballo camiña na dirección de ter unha información a priori sobre o estado do aparello.

O segundo dos informantes aporta máis coñecemento sobre como funcionan os sistemas informáticos do barco e como funciona a información no tipo de buque no que traballa. Comenta os programas informáticos que se empregan habitualmente nos barcos, tales coma o *TurboWin* e o *MaxSea*, explicados no capítulo 2. Tamén aporta datos sobre que é o que sucede cando o aparello comeza a coller fango, e como el o soluciona cando se dá o caso. Os indicios que lle indican un aparello enfangado son os seguintes: a velocidade do barco vaise reducindo, sendo o habitual pasar dunha velocidade de 3 nós a 1 nun período de 5 minutos aproximadamente; a abertura horizontal das portas vaise reducindo gradualmente e a abertura vertical aumenta. A diferenza entre enfangar e coller peixe radica en que cando hai capturas a reducción da velocidade non é tan acentuada, e que nalgún momento o sensor de capturas actívase se o que está a entrar na rede son capturas.

O procedemento para solucionar un enfangamento segundo este capitán é o seguinte:

1. Parar a marcha do barco.
2. Levantar o aparello do fondo, pero non ata o barco.
3. Avanzar un pouco o barco coa esperanza de que o aparello se limpe coa auga do mar.
4. Se o anterior non funciona, débese levantar de todo o aparello e limpar a cuberta.

Outra das informacións que achega este informante ten que ver con que tipo de datos contén o barco. Comenta que na computadora do barco gárdase a información sobre a localización e duración dun lance. O resto da información dun lance é habitualmente escrita nun caderno. Particularmente, no seu caderno escribe a seguinte información: localización, metros de fondo, metros de cable (cuxo cálculo resulta de multiplicar os metros de fondo por 3), altura da rede con respecto ao fondo, temperatura da auga, abertura horizontal de portas, data e hora. Outro problema que atopa no día a día é a non informatización desta información, que se podería incluír facilmente a un sistema de información. Finalmente, este informante proporciona tanto a información gardada dixital coma a das súas anotacións persoais mediante fotografías. Desafortunadamente, a información dixital non resulta ser útil para o problema do fango. O mesmo sucede coa información do caderno, que ademais é necesario dixitalizar antes de usar.

Por último, o profesional da empresa de sensores facilita toda a información en referencia ás características dos seus sensores e do seu programa. Explica como funcionan algúns dos sensores máis importantes: o sensor de captura, que consiste nun cable que se expande cando

a rede tamén o fai; e os sensores de abertura das portas, que funcionan medindo a latencia que tarda en emitir un sinal dunha porta a outra. Ambos sensores transmiten a través dun sinal a súa información a un sensor mestre na popa do barco. Un problema que comenta este profesional é que cando o mar está revolto, xa sexa polo tempo ou pola propia hélice do barco, o sinal córtase e non chega ao sensor mestre.

Este informante proporciona os datos sensóricos de múltiples barcos. Algúns destes datos correspóndense con erros nos sensores, polo que deben ser filtrados apropiadamente. Mais estes datos si poden ser útiles debido a que, ao seren sinais temporais, permiten facer regresión sobre eles. A información da predición pode resultar útil para o patrón que estea a manexar o barco. Cun etiquetado que indique se o lance enfangou ou non, estes datos serían de moita máis utilidade, xa que poderiamos facer un clasificador que indique en tempo real se o lance enfanga ou non. Desafortunadamente, só cos datos proporcionados non se pode determinar se hai enfangamento ou non.

4.5 Procesado dos datos obtidos

Unha vez obtidos o coñecemento necesario e o conxunto de datos, pódese proceder a observar detidamente os datos, de xeito que se poida ver que problemas poden presentar.

Nunha primeira inspección, podemos ver que os datos procedentes de determinados barcos non conteñen os suficientes datos por lance ou directamente non conteñen datos. Isto ten que ver con que os datos son recollidos mentres o barco está parado para comprobar se os sensores están a funcionar correctamente. Estes datos deben ser descartados por careceren de utilidade. Os arquivos en orixe veñen no formato propio do programa da empresa de sensores e o programa permite exportar os datos a un ficheiro de texto para poder manexalos. En cada ficheiro de lance hai varias táboas e cada unha destas corresponde con cada sensor dos que dispón o barco.

O primeiro proceso que se aplica a estes ficheiros é o de separar cada ficheiro nas táboas que corresponden. Cada unha destas táboas gárdase en formato CSV, de xeito que a libraría *pandas* poida lelos e transformalos a *dataframes*. Estes *dataframes* son obxectos da libraría *pandas*, en concreto son táboas de datos de dúas dimensións con filas e columnas etiquetadas, semellante a unha táboa dunha base de datos. Esta libraría permite manipular os datos dun xeito sinxelo.

Antes de continuar co procesado, decídese que só se poden aproveitar para esta investigación as táboas de abertura das portas, debido a que é o único sensor común a todos os barcos dos que hai información.

O seguinte paso no procesado dos datos é substituír os valores nulos. Debido a que o

formato orixinal é texto plano, os valores nulos están expresados como "—". Cómpre transformalos a un formato lexible para a implementación, polo que se substitúen estes valores polo valor "not a number" que proporciona a libraría de *numpy*. Seguidamente solvéntase o problema de que os decimais nos números reais das táboas están separados por comas, o que supón un problema á hora de separar as columnas no formato CSV.

Máis adiante, transfórmase a data das táboas de formato texto a un obxecto da propia libraría *pandas*, para poder empregar esta data como índice de cada fila. Unha vez feito isto, pódese proceder a normalizar o conxunto de datos, de xeito que cada instante temporal sexa regular. Se non se normaliza, os instantes de tempo entre medidas abranguen un rango de entre segundos e ata un minuto. Neste caso normalizarase o conxunto a 30 segundos, empregando a media de todos os datos que se atopen nese rango. Esta frecuencia solventa parte dos datos inválidos, de xeito que non se teñen en conta na media de todos os datos que haxa neses 30 segundos. Pódese dar o caso de que haxa lecturas entre as que pasen máis de 30 segundos, cuestión que se resolve mediante un interpolado lineal, que permite reencher os valores nulos cunha aproximación do valor que debería ter tendo en conta os valores anteriores e seguintes. Se hai máis de 20 valores nulos continuos, cóntase como que ou ben hai demasiados datos inválidos, ou que se corresponde coa diferenza entre dous lances, polo que non se realiza a interpolación.

Seguidamente, para eliminar os valores extremos inválidos, seleccionamos de cada arquivo os datos que abranguen entre o 15% e o 99% do valor da profundidade. Traballamos coa profundidade debido a que en función desta podemos saber se o aparello está recollido ou non, de xeito que cando o valor se aproxima a 0, é que se está a recoller o aparello. Na figura 4.5 vemos como é o cambio entre antes e despois de facer este filtrado. Nesa mesma figura podemos ver que unha vez feito este proceso resulta fácil dividir os lances dun mesmo ficheiro, posto que xa están separados por datos eliminados. O único problema restante en canto ao procesado é que poden quedar algúns datos residuais que non puideron ser filtrados. Afortunadamente estes conxuntos adoitan ser duns poucos datos, polo que unha vez separadas as series, escóllense aquelas que conteñen máis de 50 datos para quedarnos cos lances válidos.

Unha vez chegado este punto, os datos xa están preparados para as entradas nos modelos de aprendizaxe automático. Durante a exploración do problema, tenteouse a posibilidade de etiquetar os datos conforme se un lance enfangou ou non. Debido a isto, fíxose unha extracción de características dos sinais de apertura e profundidade, onde se seleccionaron características coma o histograma, media, desviación e a transformada de ondícula. Estas características seríannos útiles no caso dun problema de clasificación, pero ao ser o noso un problema de regresión non lle podemos dar utilidade, xa que as nosas características deben ser sinais completos e non características destes sinais.

A continuación, seleccionamos das características dispoñibles a profundidade e a abertu-

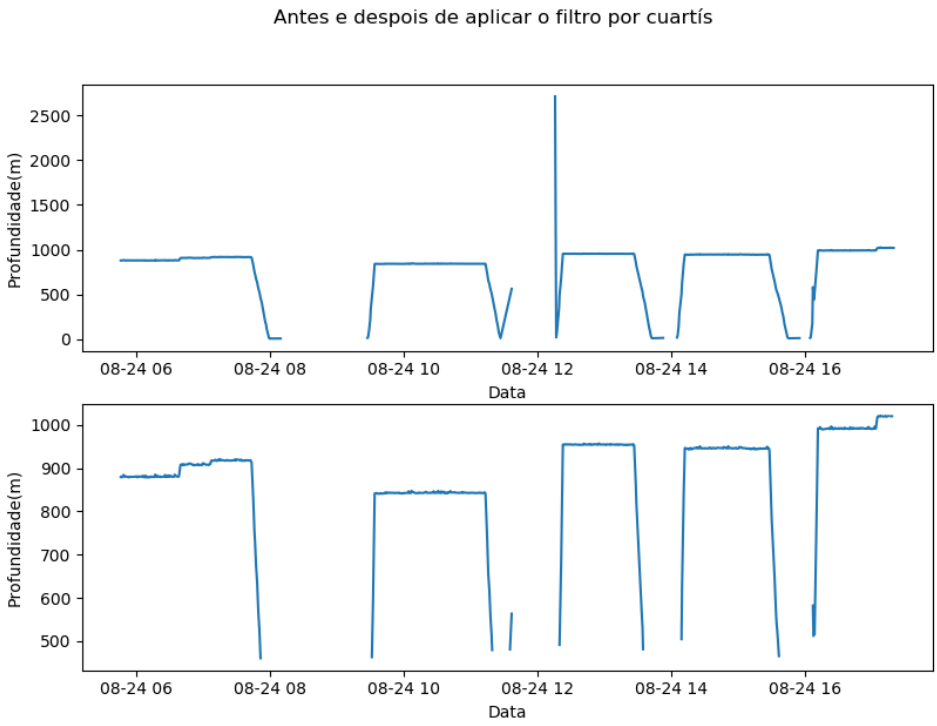

Figura 4.5: Comparación entre a serie temporal da profundidade antes e despois de filtrar os datos.

ra horizontal de portas, por seren as únicas características comúns en todos os barcos. Por último, realízase un xanelado sobre cada lance. Escóllese que cada xanela terá un tamaño de 80 datos, ademais do dato de saída.

Unha vez feito todo este proceso, o conxunto de datos está listo para ser a entrada dos modelos de aprendizaxe máquina de *sklearn*.

4.6 Escolla de modelos e adestramento

Neste aparato detallase o proceso de escolla dos modelos de aprendizaxe máquina que se levou a cabo, así como o proceso que se levou a cabo para o adestramento.

4.6.1 Escolla de modelos

Unha vez preparados os datos, débense seleccionar os algoritmos de aprendizaxe máquina que máis se apropien para este tipo de problema. Como xa se comentou anteriormente, non contamos con outros estudos sobre este tipo de sensores, polo que a escolla débese basear en estudos de sensores en xeral. Selecciónanse tres algoritmos, o de regresión lineal, *random*

forests e *support vector machines*, baseándonos en [8], [10] e [30]. Hai que ter en conta que estes estudos soamente nos serven de orientación, posto que non se poden extrapolar os resultados dun experimento a outro. Serven de guía á hora de ter unhas pistas sobre que modelos poden funcionar.

4.6.2 Adestramento

Puntos de predición

Para o adestramento, búscase obter oito modelos. Cada un destes modelos fai unha predición a un momento distinto do futuro, seguindo unha sucesión de fibonacci. Tendo en conta que cada punto de predición son 30 segundos, faise predición sobre os puntos 1, 2, 3, 5, 8, 13, 21 e 34. Deste xeito non só temos a predición inmediata a 30 segundos, se non que obtemos unha estimación a longo prazo sen engadir moita complexidade ao modelo. Para acadar isto constrúense varios conxuntos de datos de xanelado, variando a saída en función do punto de predición que se este a buscar.

Hiperparametrización

Excepto a regresión lineal, os modelos de aprendizaxe máquina teñen uns parámetros que deben ser configurados para obter o mellor modelo posible para este problema. De novo atopamos que non hai unha configuración óptima para todos os problemas, polo que se deben explorar o máximo posible de configuracións para acadar aquela que mellor aproxime o problema. Para acadar isto, empregarase a técnica de *grid search*, que consiste en crear unha estrutura de datos que conteña cada un dos parámetros do algoritmo cunha lista de valores asociada a cada un destes parámetros. Seguidamente, empregando esta estrutura, adéstranse todas as configuracións posibles, de xeito que exploramos todas as posibilidades de cada modelo. Os parámetros do algoritmo de random forest son: número de estimadores, que corresponde co número de árbores; máximo de características, que é o número de características a considerar cando se busca pola partición óptima dunha árbore; mínimo de mostrar para unha partición, que é o número de mostras mínimo necesario para dividir un nodo interno; mínimo de mostras para ser nodo folla; e o máximo da profundidade da árbores. Os valores escollidos para estes parámetros pódense ver na táboa 4.1.

Para o algoritmo de *support vector machines*, os parámetros son os seguintes: kernel, que especifica a función do kernel que se vai a empregar; C, que é o grao de regularización co que se flexibiliza o modelo; épsilon, que é a marxe de tolerancia na que non se penalizan os erros; e gamma, que corresponde co coeficiente da función do kernel. Os rangos de valores escollidos para estes parámetros preséntanse na táboa 4.2.

Táboa 4.1: Parámetros do algoritmo random forest

Nome do parámetro	Valores
Número de estimadores	200 a 2000
Máximo de características	raíz cadrada do número de características
Mínimo de mostras para partición	2, 5, 10
Mínimo de mostras para ser folla	1, 2, 4
Máximo de profundidade	10 a 110

Táboa 4.2: Parámetros do algoritmo SVM

Nome do parámetro	Valores
Kernel	rbf, lineal e polinómico
C	0.1, 1, 100, 1000
Epsilon	0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10
Gamma	0.0001, 0.001, 0.005, 0.1, 1, 3, 5

Batería de adestramentos

Unha vez determinados os datos de adestramentos e os modelos que se adestrarán, así como as súas configuracións, pódese comezar cos adestramentos. Para isto farase unha batería de adestramentos automatizada, que recolla os resultados destes adestramentos. Esta batería consiste en adestrar, por cada punto de predición, todos os modelos, cada un con todas as combinacións de parámetros mencionadas no apartado anterior. Ao mesmo tempo, con cada combinación fanse 5 adestramentos, un por cada partición da validación cruzada. Da validación cruzada obtense a mellor configuración de cada un dos algoritmos. Esta batería lánzase nunha máquina de AWS mediante unha conexión SSH á mesma. A continuación, recuperaranse os datos mediante o emprego da libraría *pickle*, que permite gardar un obxecto de python a un ficheiro externo, para despois cargalo en outro programa.

Resultados

Tras completar a batería de adestramentos, obtense un dataframe cos resultados de cada algoritmo. No cálculo do erro empregamos a medida da raíz cadrada do erro cadrático medio. A continuación, na táboa 4.4, pódense ver os valores dos erros para cada punto de predición e modelo, e na figura 4.6 temos a mesma información pero de forma visual. Compróbase que na maioría dos casos o modelo de regresión lineal é o que mellor funciona, excepto cando a predición é a 4 minutos e a 6 minutos e medio, onde é superado polo algoritmo SVM. A configuración do SVM neses puntos é a presentada na táboa 4.3.

Como era de esperar, a medida que se afasta o punto de predición, o erro dos modelos aumenta en xeral. Tamén pode influír sobre os resultados o tipo de datos de entrada, que non

C	Epsilon	gamma	kernel
10	0.0001	0.005	rbf

Táboa 4.3: Mellor configuración do SVM para un gap de 4 minutos e 6 minutos 30 segundos.

teñen moitos exemplos de cambios bruscos na abertura. Unha mellora de variedade de datos pode mellorar significativamente a eficacia destes modelos.

Gap	Regresión lineal	Random Forest	SVM
1(30s)	**0.007567**	0.011816	0.009021
2(1min)	**0.009388**	0.013027	0.011223
3(1min 30s)	**0.011212**	0.015157	0.013384
5(2min 30s)	**0.014510**	0.021360	0.015004
8(4min)	0.017435	0.026252	**0.016286**
13(6min 30s)	0.019429	0.026742	**0.016570**
21(10min 30s)	**0.024188**	0.027339	0.035083
34(17min)	**0.022114**	0.029105	0.025883

Táboa 4.4: Táboa cos erros cadráticos de cada algoritmo en cada punto de predición.

Como exemplos dos resultados obtidos, podemos ver un caso na figura 4.7 no que se amosa a predición contra os datos reais, coa área de erro de predición do modelo sombreada. Podemos ver que a predición se axusta á liña real, e os datos reais sempre están contidos dentro da área do erro. Tamén vemos na figura 4.8 unha comparación de datos reais contra todos os modelos de predición. Pódese ver que todos levan á tendencia do sinal orixinal.

4.7 Conclusións

No traballo realizado neste apartado, fíxose un amplo estudo do dominio, en gran parte facilitado polos colaboradores, que non só explicaron como realizan o seu labor con todo o detalle necesario, senón que tamén proporcionan datos históricos precisos para a realización do estudo. Desafortunadamente, estes datos están nun formato non válido para seren procesados polos algoritmos de aprendizaxe máquina, ademais de que algúns deles teñen unha gran cantidade de valores inválidos. Por este motivo foi necesario facer un cribado dos arquivos iniciais, e logo unha serie de transformacións aos datos restantes. Uns exemplos destas transformacións son a interpolación para que non haxa datos nulos no conxunto ou a normalización a 30 segundos para que o tempo sempre sexa no mesmo punto. Seguidamente, foi necesario facer un xanelado para que os datos puidesen ser empregados como entrada aos algoritmos de aprendizaxe. A continuación, escolléronse os algoritmos de bosques aleatorios, regresión lineal e máquinas de soporte vectorial, para adestralos. Durante este adestramento empregouse unha hiperparametrización de forma que se probaron varias configuracións por

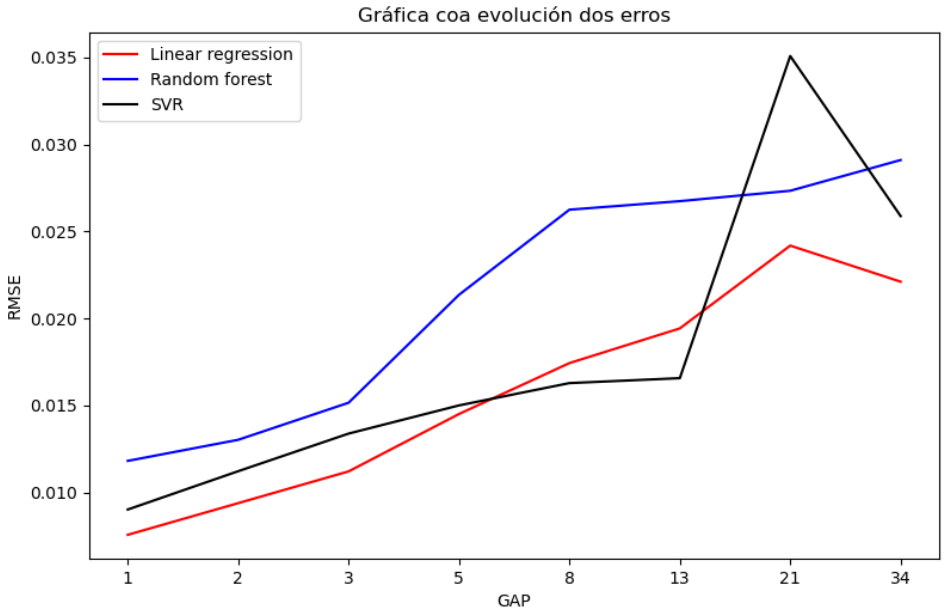

Figura 4.6: Comparación visual dos erros obtidos.

cada algoritmo. Os resultados deste adestramento manifestan que o algoritmo que mellor se adapta ao noso caso é a regresión lineal, excepto para os casos de predición a 4 minutos e a 6 minutos e medio, nos que as máquinas de soporte vectorial teñen unha maior eficacia.

Con estes resultados, podemos concluír que os modelos de regresión lineal e de máquinas de soporte vectorial funcionan para datos destas características. Acadouse un erro reducido en todos os casos, o que significa que a diferenza entre os valores reais e a predición é moi baixa en xeral. Con todo, a falta de traballos semellantes dispoñibles impídenos facer comparacións significativas. Sería especialmente interesante poñer a proba o resultado desta investigación nun entorno real de traballo, no que a opinión do usuario final avaliaría a utilidade da ferramenta.

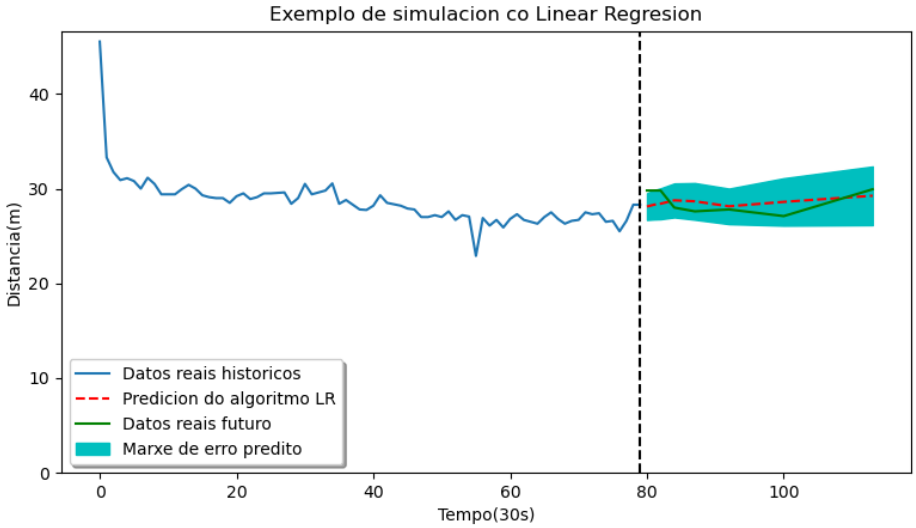

Figura 4.7: Exemplo do resultado dunha predición cos modelos óptimos xunto co marxe de erro.

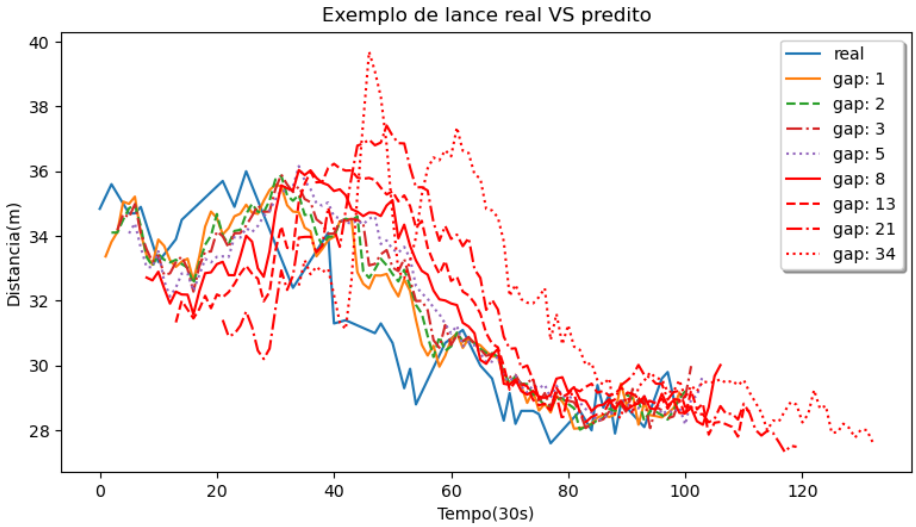

Figura 4.8: Exemplo de comparación de efectividade dos gaps sobre uns datos concretos.

Capítulo 5

Desenvolvemento web

Durante este capítulo desenvolvese todo o relacionado co desenvolvemento web deste proxecto. Descríbense os conceptos empregados, o deseño empregado e como se desenvolve.

5.1 Resumo

Ao comezo deste apartado descríbense as principais tecnoloxías empregadas no desenvolvemento web, tales como a arquitectura REST, *swagger* e *angular*. Seguidamente, explícanse os principais razoamentos do deseño desta aplicación, sendo obxecto principal deste deseño o padrón Modelo Vista Controlador. A continuación exponse como se desenvolve o servidor, empregando OpenAPI e *swagger*, así coma os tests que se levan a cabo. Tras isto, preséntase como se levou a cabo a implementación do cliente e a súa compoñente principal, que é a gráfica coa predición. Por último, comentaranse os resultados e conclusións que se acadan neste capítulo.

5.2 Fundamentos tecnolóxicos

Nesta sección explícanse aqueles conceptos máis importantes no desenvolvemento web deste proxecto. Coméntanse tanto os conceptos teóricos necesarios como as principais tecnoloxías empregadas.

5.2.1 Arquitectura REST

A transferencia de estado representacional, ou REST polas súas siglas en inglés [31], é unha arquitectura de rede para sistemas de hipermedia distribuídos. O estilo REST defínese a partir dun estilo nulo, é dicir, unha arquitectura na que non hai ningún tipo de restrición. Seguidamente, engádense as seguintes restricións:

- **Cliente-servidor**: a primeira restrición que se fai é a de separar cliente de servidor. Ao separar os asuntos da interface de usuario e os de gardar datos mellorase a portabilidade da interface de usuario entre plataformas. Tamén mellora a escalabilidade, simplificando as compoñentes do servidor. Ademais permite a ambos compoñentes evolucionar de forma independente.

- **Sen estados**: no segundo tipo de restrición determínase que a comunicación entre o cliente e servidor debe ser sen estados, de forma que cada petición do cliente ao servidor debe conter toda a información necesaria para entender a petición, así como que o cliente non pode aproveitarse de calquera información contextual que poida haber no servidor. O estado da sesión está contido completamente no cliente.

- **Caché**: esta restrición require que os datos dunha resposta deben estar marcados explicitamente ou implicitamente como cacheables ou non cacheables: se unha resposta é cacheable, o cliente ten dereito a reutilizar esa resposta. A vantaxe desta restrición é que potencialmente reduce ou elimina algunhas interaccións, mellorando así a eficiencia, a escalabilidade e a rapidez da resposta a ollos de usuario, xa que se reduce a latencia media dunha serie de interaccións.

- **Interface uniforme**: é a característica central de REST. Para esta restrición aplícase o principio da xeneralidade do software á compoñente da interface. O conxunto do sistema vese simplificado e a visibilidade das interaccións melloradas. As implementacións quedan desacopladas dos servizos que provén, o cal anima a evolución independente. Para manter esta interface uniforme, REST define catro restricións sobre a interface: manipulación dos recursos a través de representacións, identificación dos recursos, mensaxes autodescritivos e hipermedia como o motor do estado da aplicación.

- **Sistema por capas**: segundo esta restrición, cada compoñente do sistema non pode ver máis ala da capa coa que está interactuando. Restrinxindo o coñecemento a unha soa capa conseguimos poñer límites a complexidade do sistema e promovese a encapsulación. Isto provoca que un servizo desactualizado non inflúa sobre outro novo. Tamén facilita a escalabilidade dos sistemas.

- **Code-on-demand** REST permite á funcionalidade do cliente ser estendida a través dunha descarga e execución de código en forma de scripts. Permitir que novas características sexan descargadas despois do despregamento mellora a estensibilidade do sistema, pero ao mesmo tempo reduce a visibilidade, polo que esta restrición é opcional.

Un concepto clave para comprender a arquitectura REST é o de recurso. Un recurso en REST é calquera información que pode ser nomeada, mais non a instancia concreta desta información, senón un mapeado conceptual dun conxunto de entidades.

5.2.2 Python para desenvolvemento web

Python [32] é unha linguaxe de programación interpretada, orientada a obxectos e de alto nivel cunha semántica dinámica. As estruturas de datos de alto nivel que inclúe, combinadas co tipado dinámico e coa vinculación de variables dinámica, fai que sexa unha linguaxe apropiada para o desenvolvemento rápido de aplicacións, así como para creación de scripts e para conectar compoñentes de outras aplicacións xa existentes entre elas. A sintaxe sinxela e fácil de aprender de python salienta na lexibilidade, característica que reduce o custo de mantemento dos programas. *Python* soporta módulos e paquetes, que animan á modularidade e reutilización de código.

Estas características fan de *python* unha boa opción para o desenvolvemento do servidor web deste proxecto. A facilidade de emprego da linguaxe, xunto coas de librarías de *sklearn* e *flask*, axudan a un desenvolvemento cómodo e rápido. Mais o principal motivo do emprego de python débese ao módulo de predición do servidor. Este debe estar escrito en *python*, posto que os modelos obtidos pertencen a libraría *sklearn*, e deben ser empregados con esta mesma. Se o servidor fose implementado noutra linguaxe, o módulo de predición debería ser un script aparte en python, o cal só engade complexidade ao modelo de xeito innecesario.

5.2.3 Swagger

Para entender *swagger*, é conveniente explicar primeiro que é OpenAPI. A especificación *OpenAPI* [33] é unha interface para API's REST estándar e independente da linguaxe de programación. Permite tanto a humanos como a máquinas entender os servizos dunha API sen necesidade de acceso ao código fonte, á documentación ou á inspección do tráfico. Se esta especificación está ben implementada, un consumidor pode entender e interactuar cun servizo remoto cunha mínima cantidade de implementación de lóxica. Os documentos de OpenAPI son obxectos JSON, que poden ser representados en formato JSON ou YAML.

Unha vez obtido este documento, software de *swagger* [34] proporciona unha suite de ferramentas que empregan este documento. Neste proxecto empregase *swagger-codegen*, que permite xerar unha base de servidor nunha das linguaxes que *swagger* teña dispoñible. Unha vez xerado este esqueleto, só é necesario implementar a lóxica dos servizos e a persistencia de datos. Neste caso decidiuse implementar o servidor en *python* empregando a libraría *flask*.

Swagger proporciona unha forma cómoda de desenvolver un servidor de xeito rápido. Posto que a implementación dunha web non e obxectivo central deste proxecto, *swagger* abstráenos de implementar o relacionado coas comunicacións co cliente.

5.2.4 Angular

Angular [35] é un *framework* para a creación de páxinas web dunha soa páxina. Emprega HTML, CSS e *typescript*, que é unha linguaxe de programación moi semellante a *javascript* pero incluíndo tipado. *Angular* converte os seus padróns en código optimizado para as máquinas virtuais de *javascript*. A vista do proxecto será sinxela, só requirirá dunha gráfica que mostre o sinal do sensor e a predición, e un botón para actualizar os datos sobre os que se está a traballar. Debido a isto, o único requirimento necesario para a escolla dun *framework* é que soporte *canvasJS*, unha libraría para gráficas de datos en vistas web. Tanto *angular* coma *react*, que é un framework cunhas características moi semellantes, e cunha popularidade similar, cumpren esta característica. Por ser *angular* lixeiramente máis popular, escolleuse como opción para este traballo, mais *react* sería outra opción perfectamente válida para este proxecto.

5.3 Deseño e implementación

A aplicación web deseñouse seguindo o padrón Modelo Vista Controlador [36]. Segundo este patrón, chámaselle modelo á parte da aplicación que resolve o problema, isto é, a lóxica que goberna a aplicación, así como a persistencia de datos. A vista correspóndese coa parte da aplicación coa que o usuario interactúa. Por último o controlador actúa como comunicador entre ambos. Isto sérvenos de xeito que a vista e a implementación queden desacopladas, xa que é habitual que o modelo non cambie, mais a vista si é máis habitual que sufra modificacións. Na figura 5.1 podemos ver un esquema das relacións entre estes elementos.

Ademais, na aplicación web en si acóplanse os modelos de predición obtidos durante a investigación. Na figura 5.2 observamos os principais módulos que compoñen esta web, así como o módulo de predición e un script que simula a entrada de datos en tempo real. Este esquema tamén inclúe as entradas e saídas de cada módulo.

No noso problema, o modelo corresponde coa lóxica e a persistencia dos datos, a vista será unha páxina web, e o controlador será xestionado nunha clase en python no back-end.

5.3.1 Servidor

A implementación do servidor comeza co deseño dunha especificación de OpenAPI, de xeito que teñamos antes de comezar que operacións debe implementar o servidor. Estas operacións serán subir un conxunto de datos, actualizalo, borralo, lelo e obter unha predición con estes datos. Na figura 5.3 vemos os endpoints que terá a nosa aplicación.

Seguidamente, unha vez feita a especificación, xeramos un esqueleto de servidor de python-flask empregando swagger-codegen e este documento. Esta funcionalidade xunto co docu-

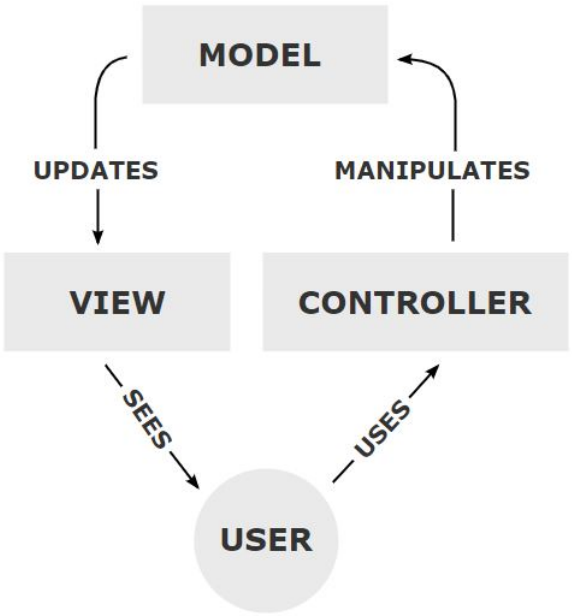

Figura 5.1: Esquema das relacións entre os elementos do patrón MVC.

mento permítenos xerar automaticamente a conexión co cliente, así coma o controlador. Unha vez feito isto, podemos implementar a lóxica do servidor.

A linguaxe escollida para a implementación do servidor é *python*, sendo o principal motivo para a escolla de python que só nesta linguaxe podemos despregar os modelos de predición obtidos na fase de investigación. Se outra linguaxe fora escollida, a predición debía facerse nun script en python, o cal só engade complexidade ao proxecto.

Unha vez feito isto, debemos implementar a capa de lóxica de negocio do servidor. Para comezar, e seguindo a metodoloxía test-driven development, primeiro desenvólvense os test que deben superar as implementacións. Os test de crear e actualizar consisten en que os datos do ficheiro subido e os que se atopan no servidor teñan o mesmo hash, así asegurámonos de que a subida de datos sexa correcta. Para o borrado, comprobamos que o ficheiro onde se gardan os datos está baleiro. Para a predición, obtense unha predición directamente co modelo, e a continuación obtense unha predición cos mesmos datos no servidor, comparando ambos resultados e asegurando que son iguais. A característica de ler os datos que residen no servidor foi implementada nunha revisión posterior, xa que nun principio non se cría necesaria. Debido a isto e a falta de tempo, non se puido realizar un test sobre esta característica.

Unha vez implementados os tests, desenvólvese a funcionalidade. Cabe destacar que se emprega o padrón data access object para desacoplar a lóxica de negocio da persistencia de datos. A persistencia de datos realizase nun ficheiro CSV, debido a que nesta primeira versión só hai soporte para un lance.

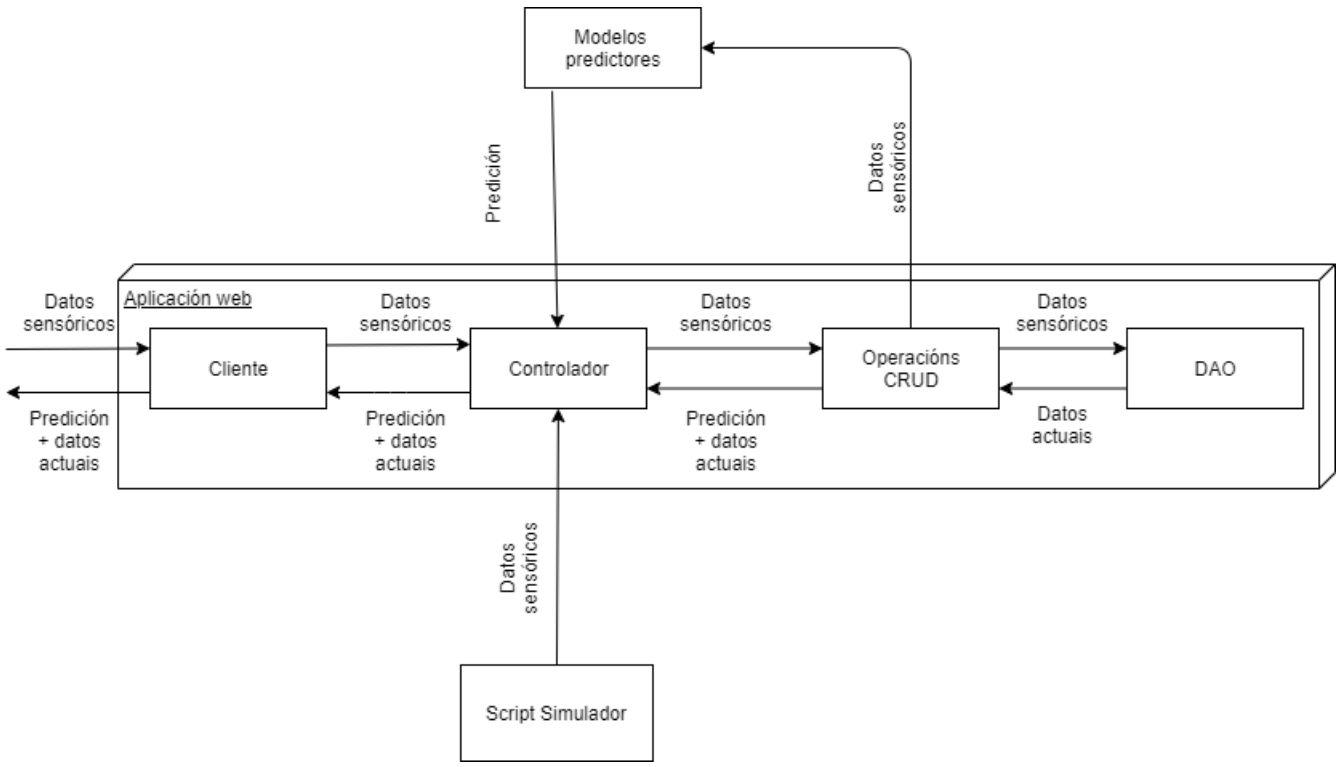

Figura 5.2: Esquema de entradas e saídas da aplicación web.

Para a predición realízase un padrón estratexia, de xeito que se permitan engadir novos modelos de predición de xeito sinxelo. Na figura 5.4 podemos ver unha representación UML do servidor.

5.3.2 Cliente

Para o cliente só requiriremos implementar unha vista que dispoña dunha funcionalidade de subir datos e unha vista destes xunto coa predición. A actualización dos datos deixase para un sistema automatizado de actualización, que simula un caso real de funcionamento. A gráfica actualízase cada 5 segundos mediante peticións GET ao servidor, de xeito que sempre contén os últimos datos dispoñibles. Por mor da falta de tempo, non se puido aplicar test driven development nesta parte do proxecto, mais debido ao carácter sinxelo e a pouca funcionalidade que se implementa, non sofre un grande impacto na calidade da vista.

A implementación en *angular* caracterízase pola alta modularidade. A aplicación divídese en compoñentes, directivas, que non se empregan neste caso, e servizos. Cada compoñente caracterízase por tres ficheiros: un HTML, un CSS e un arquivo TypeScript. Esta aplicación consta de tres compoñentes

- **Banner**: Esta compoñente é puramente visual, unha presentación da páxina.

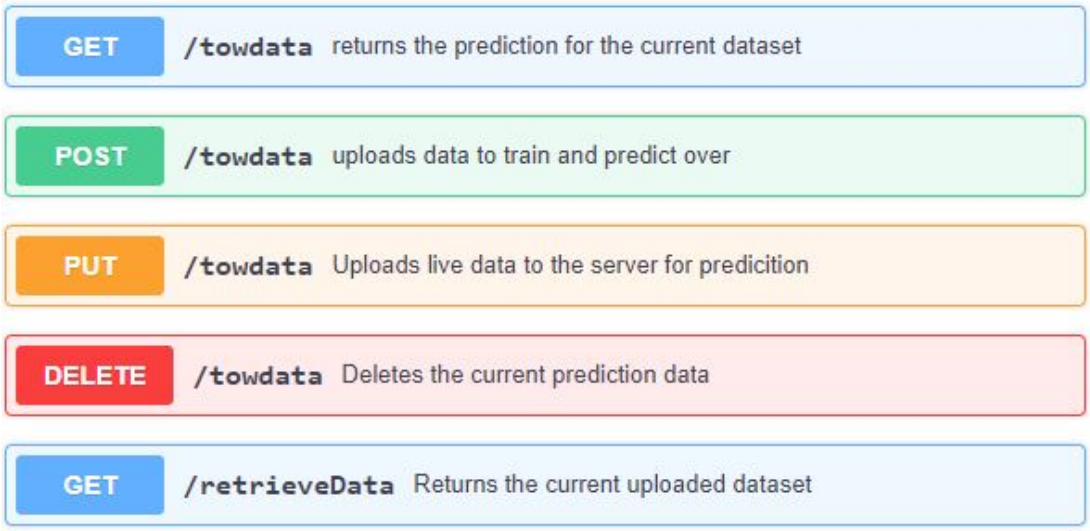

Figura 5.3: Endpoints do servidor

- **Botón de engadir**: Ao premer neste botón salta unha ventá na que poderemos escoller o ficheiro que se desexe subir. A continuación, envíase este ficheiro a directiva POST que se explica máis adiante.

- **Gráfica**: Representación visual tanto dos datos subidos coma da predición destes. Actualízase cada 5 segundos mediante dúas peticións GET ao servidor: unha para a predición e outra para os datos. Emprégase a libraría de *canvasJS*, que nos permite de forma sinxela representar gráficas.

Por último, a aplicación ten o servizo POST, o cal se encarga de engadir o ficheiro nun JSON para poder envialo no corpo dunha petición POST. Tamén se encarga de xestionar a resposta que se recibe do servidor. Na figura 5.5 podemos ver unha captura de como o usuario ve esta vista.

5.4 Resultados

Ao longo desta fase deseñouse unha aplicación web empregando o padrón Modelo Vista Controlador, que permite abstraer as funcións da parte de vista e da parte de servizo. Seguidamente créase o servidor empregando OpenAPI, que permite xerar automaticamente o esqueleto do servidor, para despois implementar a lóxica deste. Ten particular interese a implementación da funcionalidade da predición, que segue un padrón estratexia co fin de engadir facilmente novos modelos de predición. A continuación créase a vista web empregando o framework de *angular*. Impleméntase unha gráfica que permite ver tanto os datos actuais

do servidor coma a predición destes. Ademais permite ao usuario subir os seus propios datos de lance.

Con esta aplicación web acadamos ter unha funcionalidade para os modelos de aprendizaxe automático obtidos na investigación deste traballo. Por un lado, obtemos un servidor sinxelo e eficaz, que non só nos da as funcionalidades requiridas para os datos de lances e a predición, se non que grazas o deseño levado a cabo permite unha fácil ampliación das súas características, como engadir novos modelos ou conectalo a unha base de datos. Así mesmo, grazas ó emprego de OpenAPI e swagger-codegen, engadir novos endpoints a aplicación resulta moi sinxelo, xa que ao engadir á especificación os datos necesarios, o código requirido para crear eses endpoints xérase automaticamente.

Por outra banda obtense unha vista moi fácil de manexar e de ler por parte do usuario. Grazas a esta vista podemos darlle usabilidade aos modelos desenvolvidos. Sen isto, quedaríase nun modelo experimental só funcional a ollos de expertos en aprendizaxe máquina.

Ademais de todo isto, aínda que este desenvolvemento estea feito en web, non quere dicir que deba estar aloxado nun servidor remoto. Está pensado para ser empregado só na computadora do barco, máis facéndoo como aplicación web gañamos a vantaxe de que se se requírese aloxala en remoto, poderíase facer facilmente. Se o desenvolvemento fora completamente nunha aplicación de escritorio, a web deberíase facer de cero.

Ao mesmo tempo, esta aplicación web conta con certas limitacións. Por exemplo, a entrada de datos só soporta o formato CSV. Este formato, aínda que resulta moi útil nas cuestións de aprendizaxe máquina, non é o máis empregado fóra deste ámbito. Outra das limitacións é que, debido ao carácter privativo das aplicacións de sensores, tense descoñecemento de como chegan os datos dos sensores á computadora das embarcacións, que impide coñecer cal sería a mellor forma de introducir as actualizacións de datos. A posible instalación deste sistema nun barco debe ser tendo en conta as particularidades de cada embarcación. Por último, o ideal na gráfica sería que no eixo das abscisas mostrara o punto temporal no que se atopa cada dato, pero por cuestión de tempo non se puido implementar.

Con todo resulta unha aplicación útil para visualizar como funciona o modelo con datos simulados, que nos axuda a ter un exemplo do que un potencial usuario pode ver mentres o usa.

Como conclusión podemos dicir que aínda que é unha aplicación sinxela, cumpre a función para a que foi creada. Ademais, grazas ao deseño feito, resultaría sinxelo engadir novas funcionalidades ao servidor, como engadir máis modelos, cambiar como se fai a persistencia de datos, etc. A vista da aplicación da o mínimo necesario para a visualización dos datos, o cal axuda ao usuario a centrar a súa atención no importante, que é a predición. En xeral considérase que a aplicación é apropiada para a creación e visualización dos modelos de predición obtidos durante a investigación.

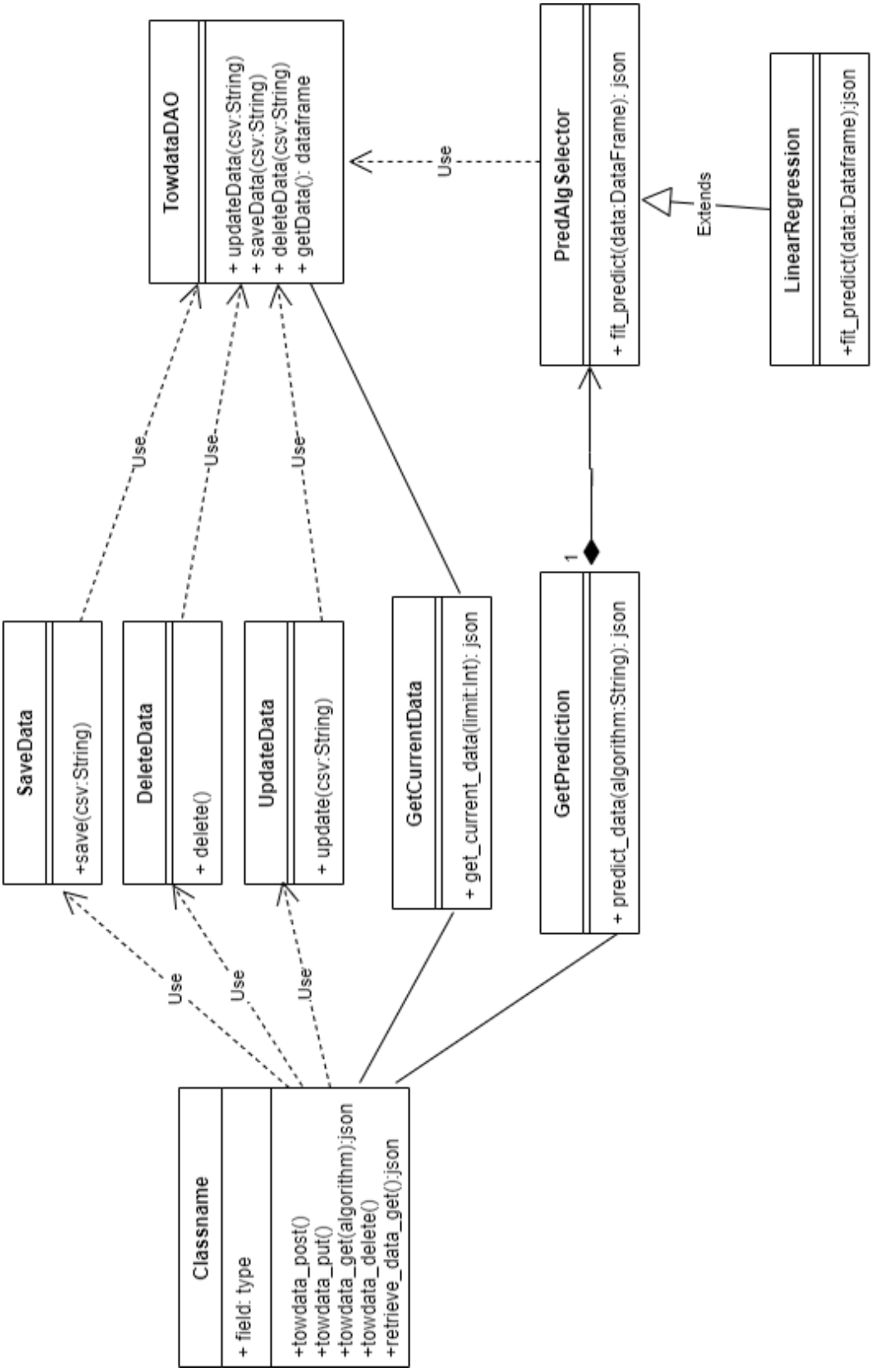

Figura 5.4: Diagrama de clases do servidor

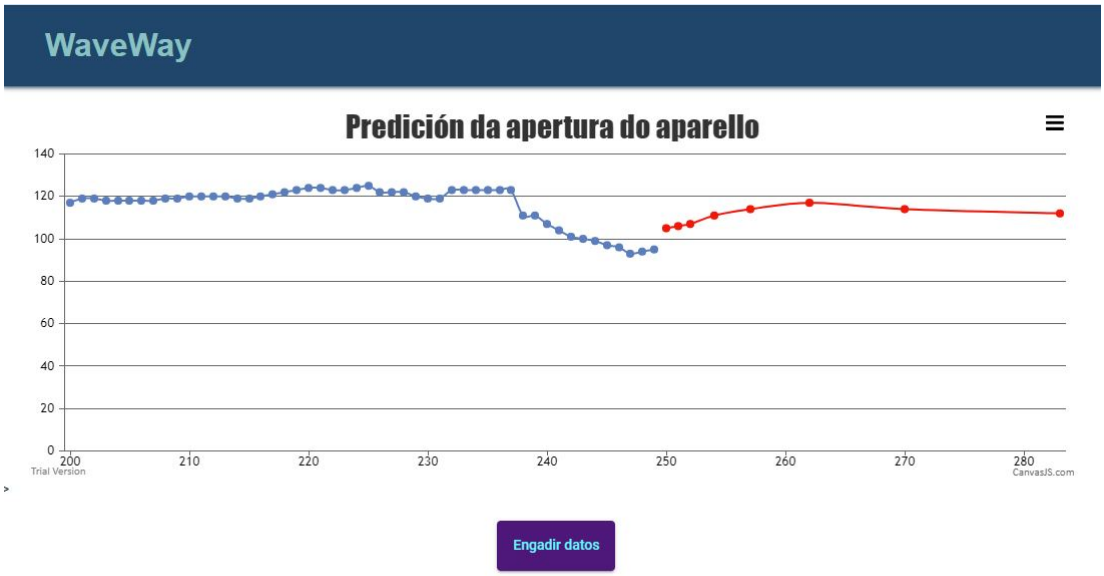

Figura 5.5: Captura do cliente deseñado. En azul, os datos reais, en vermello os preditos. Cada punto son 30 segundos nos datos reais. Na predición, sigue a serie de fibonacci comezando en 1, sendo cada punto o seu valor da sucesión de fibonacci multiplicada por 30 segundos

Capítulo 6

Conclusións e traballo futuro

Como final para este estudo, preséntanse as conclusións que se poden extraer del, as cuestións que aporta ao estado da arte, as carencias que ten e os seus puntos fortes. Tamén se inclúen posibles liñas de traballo futuras, xa que ao ser un traballo novo no seu eido, abre moitas posibilidades para investigar sobre este tema.

6.0.1 Conclusións

Para concluír, neste estudo analizouse o papel que pode ter a aprendizaxe máquina no sector da pesca do arrastre, que carece de avances tecnolóxicos en canto á análise de datos. Acadouse o obxectivo do proxecto de crear un modelo de aprendizaxe máquina que obteña unha predición sobre datos sensóricos, mais, por desgraza, non existen ata o momento traballos de referencia cos que comparar resultados, como adoita ser común noutras investigacións de aprendizaxe automática. Isto, sumado ao feito de que non se puido probar nun entorno de traballo real o resultado, complica determinar o grao de eficacia dos modelos obtidos, sendo os únicos marcadores de eficacia os erros obtidos nos datos de test con respecto ao datos reais, que son en xeral baixos.

Tamén se conseguiu o obxectivo secundario de implementar unha aplicación web na que despregar os modelos de aprendizaxe previamente obtidos. Isto dálle usabilidade á investigación sobre a aprendizaxe máquina, que doutro xeito só se quedaría nun modelo.

Como puntos fortes do traballo, podemos destacar que é a primeira aproximación á resolución dun problema en base aos sensores de barcos de arrastre. Supón un avance de cara a proporcionar unha capa de análise aos datos obtidos por estes sensores, dando unha información ao patrón a priori que doutra forma non podería conseguir. Ademais, aínda que non dispoñamos de moitos marcadores para confirmalo, o erro obtido no modelo é moi baixo, o que significa que a predición se axusta moi ben aos datos reais.

Por outra banda, o traballo ten certas limitacións que deben ser tomadas en consideración. A principal limitación do traballo débese aos datos de adestramento usados. Estes datos son

de lances reais, mais non coñecemos que situacións se deron neses lances, é dicir, non teñen un etiquetado ou unha información que nos diga se un lance tivo problemas ou non, ou se enfangou ou non. Isto supón que non se coñeza a variabilidade dos datos, no sentido de que non sabemos cantos dos lances que temos son completamente normais e cantos non. Outro dos obstáculos ten que ver con que a implantación da aplicación web nun entorno real supón un reto, por mor de que se descoñece como é a entrada de datos dos sensores ao ordenador, ou incluso se estes datos poden ser interpretados por un programa que non sexa propio da empresa de sensores.

Ao ser un traballo proposto polo alumno, a motivación intrínseca axuda a involucrar máis ao investigador. Como este traballo está intimamente relacionado cun sector próximo a nivel persoal, axuda a que o entorno queira involucrarse e contribuír a este proxecto.

Grazas a este traballo mellorouse a capacidade de traballo autónomo, xa que se trata dun proxecto individual. Coa necesidade de entrevistar a expertos, trabállanse aptitudes de relación interpersoal, que non adoitan incluírse na formación que proporciona o grao. Aínda que o traballo sexa autónomo, o alumno sempre ten a guía do director do proxecto, xa que o alumno realiza con este estudo a súa primeira aproximación ao mundo da investigación, área na que o director do proxecto é experto. No traballo afóndase no eido da aprendizaxe máquina, cuxos conceptos se aprenderon ao longo da mención en computación. Permitiu coñecer máis en profundidade que procesos se deben levar a cabo para acadar os mellores modelos de aprendizaxe máquina posibles. Tamén se aplicaron padróns de deseño que se aprenderon durante o grao, imprescindibles para obter un código de calidade. Por outra banda tamén se aplicaron tecnoloxías que si se ensinan no grao, mais non na mención de computación, como a implementación dunha vista en *angular*. Por último, adquiríronse competencias de escrita científica en lingua galega por medio do emprego de recursos de terminoloxía específica tales como *TERGAL* [37].

6.0.2 Traballos futuros

En canto as posibles liñas de traballo futuras, estudáronse varias delas, mais non foi viable realizalas por mor da limitación temporal que impoñen as características dun traballo de fin de grao. Algunhas destas posibilidades son as seguintes:

- Estudar máis modelos de aprendizaxe máquina que quizais funcionen mellor, destacando que se podería facer un estudo de aprendizaxe profundo empregando redes neuronais.

- Obter uns datos etiquetados e de mellor calidade, que nos axuden a que os algoritmos aprendan padróns de datos máis diversos.

- Se se obtiveran os datos mencionados no punto anterior, outra liña é resolver problemas de clasificación de lances enfangados, non enfangados e con capturas, que axudaría nunha posible automatización da pesca.
- Mellorar a aplicación web en varios aspectos, como dar soporte para históricos de lances, crear usuarios para que varios poidan empregar a web ao mesmo tempo e permitir a escolla por parte do usuario de varios modelos.

Relación de Acrónimos

S-AIS *Satellite-based Automatic Information System*

GPS *Global Positioning System*

CRUD *Create, Read, Update, Delete*

AWS *Amazon Web Services*

CSV *Comma Separated Values*

JSON *JavaScript Object Notation*

Glosario

Tensiómetro Sensor de medición que mide a forza que esta exercendo o cable sobre o barco.

Carta náutica Representación a escala de augas navegables e rexións terrestres adxuntas.

Lance Período durante o cal o aparello de pesca está no fondo do mar.

Cable No eido mariño, un cable correspóndese cunha corda grosa.

Babor Lado esquerdo dunha embarcación

Estribor Lado dereito dunha embarcación

Bibliografía

[1] "Salarios para empleos en españa." [Online]. Available: https://es.indeed.com/salaries/

[2] J. L. del Val Román, "Industria 4.0: la transformación digital de la industria," in *Proceedings of the Conferencia de Directores y Decanos de Ingeniería Informática, Informes CODDII, Valencia, Spain*, 2016, p. 10.

[3] A. C. Penela, C. S. Villasante, and Y. Z. Tarrío, "Análisis estructural del sector pesquero gallego: Retos y perspectivas de futuro en la unión europea," in *XIX Reunión Asepelt*, España, Badajoz, 2005.

[4] I. G. de Estatística, 2020. [En liña]. Dispoñible en: https://www.ige.eu/

[5] E. N. de Souza, K. Boerder, S. Matwin, and B. Worm, "Improving fishing pattern detection from satellite ais using data mining and machine learning," *PloS one*, vol. 11, no. 7, p. e0158248, 2016.

[6] S. J. Walsh and W. M. Hickey, "Behavioural reactions of demersal fish to bottom trawls at various light conditions," in *ICES Mar. Sci. Symp*, vol. 196, 1993, pp. 68–76.

[7] J. Gordon, O. Bergstad, and P. Pascoe, "The influence of artificial light on the capture of deep-water demersal fish by bottom trawling," *Marine Biological Association of the United Kingdom. Journal of the Marine Biological Association of the United Kingdom*, vol. 82, no. 2, p. 339, 2002.

[8] A. Moraru, M. Pesko, M. Porcius, C. Fortuna, and D. Mladenic, "Using machine learning on sensor data," *Journal of computing and information technology*, vol. 18, no. 4, pp. 341–347, 2010.

[9] A. Kanawaday and A. Sane, "Machine learning for predictive maintenance of industrial machines using iot sensor data," in *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*. IEEE, 2017, pp. 87–90.

[10] K. Tan, W. Ma, F. Wu, and Q. Du, "Random forest–based estimation of heavy metal concentration in agricultural soils with hyperspectral sensor data," *Environmental monitoring and assessment*, vol. 191, no. 7, p. 446, 2019.

[11] R. De Arce and R. Mahía, "Modelos arima," *Programa CITUS: Técnicas de Variables Financieras*, 2003.

[12] J. J. Lamas Seco and J. A. García Naya, "Dispositivos hardware e interfaces - adquisición de datos," Transparencias, 2019.

[13] ——, "Dispositivos hardware e interfaces - computación física," Transparencias, 2019.

[14] R. Adhikari and R. Agrawal, *An Introductory Study on Time series Modeling and Forecasting*, 01 2013.

[15] "Test driven development." [En liña]. Dispoñible en: http://www.chuidiang.org/java/herramientas/test-automaticos/tdd-test-driven-development.php

[16] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[17] A. R. Webb, *Statistical Pattern Recognition*. John Wiley Sons, 2002.

[18] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.

[19] K. Kumari and S. Yadav, "Linear regression analysis study," *Journal of the Practice of Cardiovascular Sciences*, vol. 4, p. 33, 01 2018.

[20] S. Weisberg, *Applied linear regression*. John Wiley & Sons, 2005, vol. 528.

[21] L. Rokach and O. Maimon, *Decision Trees*, 01 2005, vol. 6, pp. 165–192.

[22] T. K. Ho, "Random decision forests," in *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*, ser. ICDAR '95. USA: IEEE Computer Society, 1995, p. 278.

[23] L. Breiman and A. Cutler, "Random forests." [En liña]. Dispoñible en: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm

[24] V. Vapnik, "The support vector method of function estimation," in *Nonlinear Modeling*. Springer, 1998, pp. 55–85.

[25] T. Evgeniou and M. Pontil, "Support vector machines: Theory and applications," vol. 2049, 01 2001, pp. 249–257.

[26] M. Awad and R. Khanna, *Support Vector Machines for Classification*, 01 2015, pp. 39–66.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[28] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[29] "What is aws?" [En liña]. Dispoñible en: https://aws.amazon.com/es/what-is-aws/

[30] C.-Y. Yang, C.-Y. Lin, S. Galsanbadam, and H. Samani, "Multivariable support vector regression with multi-sensor network data fusion," 10 2018, pp. 4029–4034.

[31] R. T. Fielding and R. N. Taylor, *Architectural styles and the design of network-based software architectures*. University of California, Irvine Irvine, 2000, vol. 7.

[32] "What is python?" [En liña]. Dispoñible en: https://www.python.org/doc/essays/blurb/

[33] "Openapi especification." [En liña]. Dispoñible en: https://swagger.io/specification/

[34] "What is swagger?" [En liña]. Dispoñible en: https://swagger.io/docs/specification/2-0/what-is-swagger/

[35] "Angular." [En liña]. Dispoñible en: https://angular.io/

[36] J. Deacon, "Model-view-controller (mvc) architecture," 2009.

[37] TERMIGAL, "Tergal: Banco de termos galegos recomendados," 2016.