Universidad de Córdoba

Escuela Politécnica Superior de Córdoba
Grado en Ingeniería informática
Especialidad en Computación
Curso 2024 - 2025

Análisis y modelado de series temporales: aplicación a la predicción de características de las olas

Anteproyecto
Tipología de investigación aplicada.

Autor:
Francisco Javier Pérez Castillo

Directores:
Javier Sanchez-Monedero
David Guijo Rubio

Córdoba, Enero del 2025

Índice del anteproyecto

1. Introducción

La energía undimotriz es aquella obtenida a través del movimiento de las olas en cuerpos de agua. El éxito de la generación de energía a través de este método reside en la recopilación de diferentes observaciones meteorológicas, donde encontramos la altura de la ola y el periodo de energía de la misma. Estos datos se obtienen a través de boyas en aguas cercanas a la costa y mar adentro, equipadas con sensores que permiten medir estas observaciones. Con el conocimiento de esto, se pueden abordar problemas tales como la cantidad de energía que se puede generar, previsión de posibles daños en infraestructura causadas por oleaje intenso, seguridad en operaciones marítimas o navegación, etc. El problema radica en que la precisión de los datos obtenidos de las boyas, junto con la naturaleza del oleaje, dificultan una estimación precisa de sus características [1]. Además, eventos climáticos extremos y fallos en la adquisición de datos pueden generar problemas a la hora de usar métodos tradicionales.

Es por esto que es fundamental el desarrollo de modelos predictivos capaces de analizar patrones en series temporales y mejorar la precisión de las predicciones. Estas técnicas permiten estimar valores futuros a partir de datos históricos, proporcionando herramientas más robustas para la gestión de la energía undimotriz y la seguridad marítima [2]. Este trabajo se centra en la aplicación de modelos de aprendizaje automático para predecir con mayor precisión las características del oleaje.

1.1. *Machine Learning*

En la actualidad, la inteligencia artificial es el tema de moda. El desarrollo de proyectos informáticos relacionados con este tema ha crecido exponencialmente en los últimos años, pero sus fundamentos siguen recayendo mayormente en el aprendizaje automático. El aprendizaje automático, conocido como *machine learning*, es una rama de la inteligencia artificial que permite a los sistemas aprender a partir de datos, sin necesidad de ser programados paso a paso. Para lograrlo, se analizan datos que pueden estar etiquetados o no, buscando patrones útiles para tareas como predecir, clasificar o agrupar información.

En este área, existen varios tipos de aprendizaje automático [3]. El aprendizaje supervisado utiliza datos con etiquetas para identificar patrones y hacer predicciones sobre una variable dependiente, que puede ser continua (velocidad del viento) o discreta (detección de una enfermedad), mientras que el no supervisado trabaja con datos sin etiquetar para encontrar relaciones entre estos. También existe el aprendizaje semisupervisado, que combina ambos enfoques, y el aprendizaje por refuerzo, donde el sistema aprende interactuando con un entorno, recibiendo recompensas o penalizaciones según su desempeño.

1.2. Aprendizaje supervisado

Como se explicó previamente, el aprendizaje supervisado consta de extraer información de patrones que han sido etiquetados previamente por un experto. Dentro de esta área, podemos considerar los problemas dentro de dos grupos. El primero es la clasificación. Se aplica cuando queremos que los datos de entrada se asignen a una clase concreta. Dependiendo de estas clases, los problemas podrán ser de clasificación nominal o de clasificación ordinal. La nominal, reside en que las clases no tienen un orden propio, mientras que la ordinal sí poseen un orden. El segundo es la regresión. En este tipo, la salida no posee clases concretas, sino que toma un valor cualquiera dentro de un rango.

2. Objetivos

El objetivo principal de este proyecto es la predicción de características de las olas de cuerpos de agua a partir de series temporales mediante técnicas de aprendizaje automático aplicando además marcos de trabajo (*frameworks*) que semi-automatizan las fases de un proyecto de *machine learning*.

Para obtener esto, se definen los siguientes objetivos, los cuales se buscan cumplir al final del proyecto:

1. Estudio de métodos de predicción para problemas de series temporales en el caso de caracterización de olas en el contexto de energía undimotriz.
2. Diseño e implementación de un proyecto de *machine learning* básico para el caso de energía undimotriz sin el uso de herramientas de automatización, generando un código y resultados iniciales que sirvan como referencia.
3. Aprendizaje de *frameworks* de automatización y estructura de proyectos de *machine learning* junto con el aprendizaje de bibliotecas científicas relacionadas con procesamiento y modelado de datos.
4. Implementación del flujo estándar de un proyecto de experimentación de *machine learning* utilizando los entornos y bibliotecas mencionados.
5. Evaluación de los resultados experimentales, identificando tanto las ventajas como las limitaciones de estos entornos de desarrollo en el contexto del proyecto.

3. Antecedentes

3.1. Clasificación ordinal

La clasificación ordinal es una variante de la clasificación supervisada, donde las etiquetas de las clases poseen un orden. Esta estructura ordinal es crucial, ya que influye tanto en el comportamiento del modelo como en sus predicciones. A diferencia de la clasificación nominal, donde las categorías no tienen relación entre sí, en la clasificación ordinal es fundamental preservar la relación entre clases. Este tipo de problemas aparecen en múltiples situaciones, como por ejemplo en encuestas de satisfacción (“malo”, “regular”, “bueno”, “excelente”), en evaluaciones de riesgos (“riesgo bajo”, “riesgo medio”, “riesgo alto”) o en la intensidad de fenómenos naturales (“leve”, “moderado”, “severo”).

Para poder abordar este problema, y siguiendo el artículo *Ordinal Regression Methods: Survey and Experimental Study* [4], se presenta la siguiente taxonomía:

- ***Naïve Approaches***: Consiste en la aplicación de métodos que no están especializados en la clasificación ordinal, perdiendo por tanto la estructura de orden de los datos. El problema se trataría como si fuera de regresión (asignando valores numéricos a las clases) o como clasificación nominal.
- ***Ordinal Binary Decompositions***: Este enfoque consiste en descomponer el problema en múltiples problemas de clasificación binaria. Se puede realizar de dos formas: con un modelo de salida múltiple único (donde cada salida pertenece o no a una clase mayor de un umbral) o con múltiples modelos independientes (se entrenan varios modelos binarios en paralelo).
- ***Threshold Models***: Este tipo consiste en proyectar los datos de entrada en un espacio latente, típicamente de una dimensión, en el que los valores contínuos se asignan a etiquetas ordinales usando umbrales. Existen varias formas, como el uso de máquinas de vectores soporte, clasificación binaria aumentada, aprendizaje discriminante, modelos de enlace acumulativo, entre otros.

3.2. *Frameworks*

En el ámbito del desarrollo software, un marco de trabajo (*framework*) es un conjunto de herramientas, bibliotecas, reglas y prácticas destinadas para facilitar el desarrollo de los programas. Estos permiten establecer una estructura común para que así los desarrolladores tengan resueltos problemas generales [5]. A continuación, se expondrán *frameworks* de interés que serán usados durante el proyecto.

3.2.1. *Kedro*

Los marcos de trabajo establecidos en las áreas de ciencia de datos y *machine learning*, se centran tanto en cada una de las fases como en el proyecto en general. En nuestro contexto, nos encontramos en este último caso, donde el *framework Kedro* [6] se aplica para estructurar los proyectos con un enfoque en los flujos de trabajo (*pipelines*), buscando crearlos de forma reproducible y mantenible. Su principio es que un proyecto organizado y modular facilita tanto el desarrollo como la escalabilidad. Por ello, este marco permite mejorar la trazabilidad de los experimentos, asegurar la reproducibilidad de los resultados y optimizar la gestión de datos.

Si bien existen otras herramientas con propósitos similares [7], como *ZenML* y *Metaflow*, *Kedro* se distingue por su énfasis en la ingeniería de software, ofreciendo una estructura de proyecto estandarizada y una amplia gama de plugins para mejorar la gestión y visualización de *pipelines*. En contraste, *ZenML* prioriza la flexibilidad y la integración con múltiples plataformas, mientras que *Metaflow* se enfoca en simplificar la ejecución de flujos de trabajo, especialmente en entornos basados en *AWS*. En este contexto, *Kedro* representa la opción más adecuada para nuestro proyecto, ya que su enfoque modular y su robustez en la gestión de *pipelines* permiten garantizar la escalabilidad y mantenibilidad a largo plazo, integrando eficientemente varios modelos y la realización de sus pruebas.

3.2.2. *ORCA-Python*

El *framework ORCA* [8] (Ordinal Regression and Classification Algorithms) es un conjunto de métodos implementado en los lenguajes *Matlab* y *Octave*, para la clasificación y regresión de problemas de clasificación ordinal. Fue desarrollado por el grupo de investigación *AYRNA* [9], perteneciente al Departamento de Informática y Análisis Numérico de la Universidad de Córdoba.

Con el objetivo de ampliar su accesibilidad y facilitar su integración con herramientas modernas de análisis de datos, Iván Bonaque Muñoz desarrolló *ORCA-Python* en su Trabajo de Fin de Grado [10]. Posteriormente, varios alumnos ampliaron este *framework*. Este es una adaptación de *ORCA* al lenguaje *Python*, permitiendo el uso de los métodos originales usando dicho lenguaje. *ORCA-Python* mantiene las funcionalidades del *framework* original, pero se beneficia de las ventajas de *Python*, como su compatibilidad con bibliotecas como *NumPy* [11] y *Scikit-learn* [12]. Gracias a esto, los usuarios pueden usar fácilmente *ORCA* en flujos de trabajo de ciencia de datos, realizar experimentos en clasificación ordinal y desarrollar programas sin la necesidad de utilizar *Matlab* u *Octave*.

3.2.3. *MORD*

La librería *MORD* [13] (Multiclass Ordinal Regression) es una implementación en Python de métodos de regresión ordinal, diseñada para abordar problemas donde el orden de las categorías importa. Su compatibilidad con *Numpy*, *Scikit-learn* y otras librerías facilita su integración en entornos de *machine learning*, permitiendo su uso en combinación con herramientas de preprocesamiento y evaluación de modelos. Esta librería organiza sus modelos según en qué se basan: en umbrales, en regresión o en clasificación. Gracias a estas características, se podrá usar sus modelos para nuestro presente problema de las olas.

4. Partes del trabajo de fin de grado

Dada la naturaleza del proyecto, con foco en la implementación y obtención de resultados a través de principios científicos, se categoriza en el tipo de Investigación aplicada. Por ende, el documento futuro constará de las siguientes partes:

1. **Introducción**: se introducirá a la temática del trabajo, junto con la problemática de la predicción de características de las olas, indicando sus motivaciones. Adicionalmente, se hablará sobre *frameworks* usados en proyectos de ciencia de datos, como *Kedro*.
2. **Estado de la técnica**: aquí se expondrá los conocimientos necesarios y teoría relacionada con el tema a trabajar. Se mostrará desarrollos previos de este área junto con la parte que se busca abarcar. En lo referente a *Kedro,* se expondrá su uso y estructura, mostrando las características de su implementación.
3. **Formulación del problema y objetivos**: se describirá el problema a tratar junto con su fragmentación en objetivos, para lograr el cumplimiento de este.
4. **Metodología del trabajo**: en este apartado, se encontrará los métodos, diseño y forma de estudio para solventar el problema del proyecto. Se comentarán las tecnologías y herramientas usadas, destacando la automatización, junto con los procedimientos para el tratamiento de las series temporales y el modelado estadístico de estos datos.
5. **Desarrollo y experimentación**: se redactará cómo se ha estructurado el proyecto a nivel de código en esta parte, junto con las diferentes pruebas realizadas para validar el modelo. Se indicará el uso de los *frameworks* usados para ello y su funcionamiento.
6. **Resultados y discusión**: los resultados obtenidos de predicción de características de olas, junto con otros resultados de interés que estén relacionados con el problema, se expondrán en este punto.
7. **Conclusiones y recomendaciones**: finalmente, se resumirá las ideas más relevantes sacadas del proyecto, con las implicaciones que conllevan, para poder ser aplicadas al problema real.
8. **Bibliografía**: listado de todas las fuentes usadas durante todo el trabajo.

5. Fases de desarrollo y cronograma

5.1. Estudio y análisis del problema

En esta fase se profundizará en el conocimiento teórico necesario para abordar el problema de predicción de características de olas basadas en series temporales. Se revisarán modelos de clasificación ordinal y técnicas de recuperación de datos faltantes para aplicarlas en caso de ser necesario, asegurando una base sólida para el desarrollo del proyecto.

En un estudio más práctico, se conocerá en profundidad el lenguaje *Python*, junto con la herramienta *Kedro* asociada a este lenguaje, buscando con ello cómo estructurar el proyecto usando dicha herramienta. Tras ello, se estudiará la documentación relacionada con las librerías con las que implementaremos los modelos de datos.

5.2. Preprocesamiento de datos

Se realizará un preprocesamiento que incluirá la limpieza de datos, gestión de valores faltantes así como otras técnicas que sean necesarias. Este proceso garantizará la calidad y adecuación de los datos para su uso en modelos predictivos, permitiendo una implementación más precisa y robusta.

5.3. Implementación y experimentación con modelos

El lenguaje *Python* será el lenguaje con el que se implementará la solución al problema. *Python* cuenta con un amplio abanico de librerías que resuelven directamente problemas de codificación presentados en este ámbito. Al ser un lenguaje interpretado, permite ejecutar y probar diferentes partes del código de manera incremental, evitando largos tiempos de compilación. Se utilizarán los modelos de bibliotecas mencionadas, así como el uso de otras en diversas partes del proyecto.

En lo referente a la estructura interna del proyecto, se realiza a través de la herramienta *Kedro*. Este *framework*, nos permitirá estructurar el proyecto de forma modular y reutilizable. Esto junto con la posibilidad de crear *pipelines* de datos, supondrá la capacidad de automatizar diversas partes del proyecto, evitar re-ejecutar partes innecesarias, controlar dependencias y tener una mejor trazabilidad del programa.

5.4. Evaluación y análisis de resultados

En el ámbito de las pruebas, se comprobará el correcto funcionamiento de los modelos, así como la evaluación de los resultados que se obtengan. El objetivo es que los modelos predigan información correcta y realista de las olas.

La incorporación de *Kedro* en este ámbito permite la realización de pruebas de varias partes del código a través de las *pipelines*, permitiendo un rápido desarrollo de las pruebas. También se utilizará para probar diferentes configuraciones, tanto como para los modelos como para rutas de datos. Todo esto con el objetivo de conseguir una mejor predicción en los modelos.

5.5. Cronograma

La planificación temporal tendrá como objetivo establecer un uso eficiente del tiempo para este proyecto, buscando realizar todos los elementos mencionados en las fases de desarrollo. Se presenta la siguiente tabla para la estimación horaria.

Fase de desarrollo	**Mes 1**	**Mes 2**	**Mes 3**	**Mes 4**	**Horas estimadas**
Estudio y análisis del problema	30	20	0	0	50
Preprocesamiento de datos	20	30	10	0	60
Implementación y experimentación con modelos	0	30	40	30	100
Evaluación y análisis de resultados	0	0	20	10	30
Documentación	10	10	10	30	60
Total	60	90	80	70	**300**

6. Recursos

6.1. Recursos humanos

Se exponen los recursos humanos del proyecto:

Autor: Francisco Javier Pérez Castillo
Alumno del Grado de Ingeniería informática.

- **e-mail**: i72pecaf@uco.es
- **Titulación**: Grado en Ingeniería Informática
- **Mención**: Computación

Director: D. Javier Sanchez-Monedero
Investigador Doctor en la Universidad de Córdoba. Miembro investigador del grupo AYRNA.

- **e-mail**: jsanchezm@uco.es
- **Departamento**: Informática y Análisis Numérico
- Escuela Politécnica Superior de Córdoba
- Universidad de Córdoba

Director: D. David Guijo Rubio
Profesor en la Universidad de Córdoba. Miembro investigador del grupo AYRNA.

- **e-mail**: dguijo@uco.es
- **Departamento**: Informática y Análisis Numérico
- Escuela Politécnica Superior de Córdoba
- Universidad de Córdoba

6.2. Recursos software

Los recursos software que se usarán son:

- **Windows 10**: sistema operativo para el desarrollo, pruebas y generación de documentación.
- **Python**: lenguaje de programación, junto a librerías y *frameworks* pertenecientes a él.
- **Git**: software de control de versiones.
- **Visual Studio Code**: entorno de desarrollo integrado.
- **Herramientas ofimáticas**: Google Docs, Word y otras herramientas de generación de gráficos.

6.3. Recursos hardware

Como recursos hardware, se utilizará el ordenador del proyectista, cuyas características son las siguientes:

- Intel ® Core (TM) i7-7700 CPU @ 3.60 GHz (8 CPUs)
- Memoria RAM 16GB DDR4
- Controlador gráfico GeForce GTX 1060 3GB GDDR5
- Disco duro Toshiba HDWD110 1TB + SSD Kingston SA40 1TB

Bibliografía

[1] **J. Thomson, A. Fisher, y C.J. Rusch**, «The Next Wave: Buoy Arrays for Deterministic Wave Prediction in Real-time», *Nature Communications*, 2023.
[En línea]. Disponible en: https://www.nature.com/articles/s43247-023-00819-0.

[2] **D. Guijo-Rubio, A. M. Gómez-Orellana, P. A. Gutiérrez, y C. Hervás-Martínez**, «Short- and long-term energy flux prediction using Multi-Task Evolutionary Artificial Neural Networks», *Ocean Engineering, vol. 216*, 2020, p. 108089.

[3] **C. M. Bishop**, «Pattern Recognition and Machine Learning», *Springer*, 2006, pp. 2-10.

[4] **P. A. Gutiérrez, M. Pérez-Ortiz, J. Sánchez-Monedero, F. Fernández-Navarro, y C. Hervás-Martínez**, «Ordinal Regression Methods: Survey and Experimental Study», *IEEE Transactions on Knowledge and Data Engineering*, 2016.
[En línea]. Disponible en: https://ieeexplore.ieee.org/document/7161338.

[5] «IEEE Standards Association, ISO/IEC/IEEE 42010:2011 - Systems and software engineering — Architecture description», 2011.
[En línea]. Disponible en: https://ieeexplore.ieee.org/document/6129467.

[6] *Sitio oficial de Kedro*.
[En línea]. Disponible en: https://kedro.org/.

[7] «Kedro vs ZenML vs Metaflow: How to Choose the Best MLOps Framework for Your Needs», *Neptune.ai*, 2023.
[En línea]. Disponible en: https://neptune.ai/blog/kedro-vs-zenml-vs-metaflow.

[8] **J. Sánchez-Monedero, P. A. Gutiérrez, y M. Pérez-Ortiz**, «ORCA: A Matlab/Octave Toolbox for Ordinal Regression», *Journal of Machine Learning Research*, vol. 20, 2019, pp. 1-4.

[9] *Departamento de Informática y Análisis Numérico de la Universidad de Córdoba, Aprendizaje y redes neuronales artificiales*.
[En línea]. Disponible en: https://www.uco.es/grupos/ayrna/index.php/es.

[10] **Ivan Bonaque Muñoz, P. A. Gutiérrez Peña, y J. Sánchez-Monedero**, «Trabajo de Fin de Grado. Framework en Python para problemas de clasificación ordinal», 2019.

[11] *Sitio oficial de NumPy*.
[En línea]. Disponible en: https://numpy.org/.

[12] *Sitio oficial de Scikit-Learn*.
[En línea]. Disponible en: https://scikit-learn.org/stable/.

[13] *Sitio oficial de Mord*.
[En línea]. Disponible en: https://pythonhosted.org/mord/.