

CAJAMAR UNIVERSITYHACK

DATATHON

---

## Descripción Procedimiento: Fase Nacional

---

### E-No-Logos

*Autores:*

Gabriel Abilleira Rodríguez

Paula Álvarez Ferrero

Javier Godos de la Puente

---

# Índice

<b>1. Descripción del trabajo</b>	<b>1</b>
<b>2. Análisis exploratorio</b>	<b>1</b>
2.1. Investigación de los ciclos . . . . .	1
2.2. Investigación sobre el crecimiento y desarrollo de la vid . . . . .	1
2.3. Variables analíticas introducidas . . . . .	1
<b>3. Procesos de manipulación de variables</b>	<b>2</b>
3.1. Imputación de los datos de superficie . . . . .	2
3.2. Agrupación de las variables meteorológicas . . . . .	2
<b>4. Elección del modelo</b>	<b>4</b>
4.1. Proceso de selección . . . . .	4
4.2. Aspectos comunes . . . . .	4
4.3. Descripción de los modelos candidatos . . . . .	4
4.3.1. <i>Gradient Boosting Regressor</i> . . . . .	5
4.3.2. <i>Light Gradient Boosting Model</i> . . . . .	5
4.3.3. <i>Extreme Gradient Boosting Boosting Model</i> . . . . .	5
4.3.4. <i>Histogram-based Gradient Boosting Regressor</i> . . . . .	5
4.4. Selección del mejor modelo . . . . .	6
4.4.1. Explicabilidad y transparencia del modelo . . . . .	6
4.4.2. Justicia . . . . .	7
4.4.3. Sostenibilidad ecológica . . . . .	7
<b>Bibliografía</b>	<b>7</b>

---

## 1. Descripción del trabajo

Incluimos en este documento la información del procedimiento seguido para la realización de esta entrega final.

Podemos dividir el trabajo realizado en las siguientes fases:

- **Análisis exploratorio:** En esta fase realizamos un primer estudio de los *datasets*, analizamos las variables y vimos cómo podíamos enlazarlos entre sí. Además, llevamos a cabo una investigación preliminar sobre el tema, leyendo artículos y estudios realizados sobre el mismo, para poder obtener una primera idea de las variables relevantes.
- **Manipulación de variables:** El objetivo de esta segunda fase fue la elaboración del *data-frame* final, que sería luego el *input* de los distintos modelos.
- **Elección del modelo:** En esta fase probamos con distintos modelos que evaluamos con un conjunto de los datos reservado para este fin. Realizamos una selección y un ajuste de los hiperparámetros y obtuvimos la predicción final empleando el modelo cuya métrica de error era menor. Además, se llevó a cabo un análisis del mismo considerando distintos aspectos que nos ayudaron a tomar la decisión.

En las siguientes secciones detallamos cada una de estas fases con más profundidad.

## 2. Análisis exploratorio

Esta fase supuso la primera toma de contacto con los datos disponibles y el diseño de un primer plan para el abordaje del problema.

### 2.1. Investigación de los ciclos

Antes de hacer nada, llevamos a cabo una investigación previa sobre cómo manejar/utilizar la información obtenida en los *datasets* dados. Puesto que la mayor parte de la información proporcionada en el set de entrenamiento está relacionada con el tiempo atmosférico, se hizo hincapié en ello.

Primero, se hizo una investigación sobre los ciclos de la vid, a través de la que pudimos distinguir 6 fases en ellos [2], que son el **reposo vegetativo (R)**, el **lloro (L)**, la **brotación (B)**, la **floración (F1, F2)**, la **maduración (M)** y el **crecimiento otoñal (C)**.

### 2.2. Investigación sobre el crecimiento y desarrollo de la vid

En el proceso de nuestra investigación encontramos un artículo académico relacionado con la producción de uva en un viñedo situado en Galicia [1]. Gracias a él, encontramos que la información meteorológica obtenida en el periodo de floración era muy relevante para explicar la producción de la vid. Aunque el estudio se enfocaba en un análisis aerobiológico del aire, se encuentra que las incidencias climatológicas tenían más relevancia en la producción de la uva que la observación de más o menos polen en el aire. Aunque no haya una relación causal, hay evidencias de la correlación de algunos fenómenos meteorológicos con la cantidad de polen en el aire.

### 2.3. Variables analíticas introducidas

En la exploración posterior a la fase intermedia un número de variables se han introducido para intentar predecir números más consistentes de producción que den menor error. En este caso se

---

han introducido las variables `SUP_MOD` y `SUP_VAR` que marcan la relación de la superficie de la finca con el modo de cultivo y con la variedad de uva respectivamente. También se introducen `LAG_PROD` y `LAG2_PROD` que indican la producción del año anterior y del anterior a éste respectivamente. Finalmente, también se han introducido una variable de eficiencia que se considera como la relación entre la producción y la superficie. Como los valores de la superficie pueden variar en un rango muy grande, se toma el logaritmo de la variable para evitar manejar cifras extremadamente altas. Por último, se introduce una variable de diferencia `LAG_DIFF`, que es simplemente el resultado de la diferencia de los dos años anteriores.

### 3. Procesos de manipulación de variables

En esta tercera fase, con la información obtenida en la anterior, obtuvimos el *dataframe* final, a través del proceso descrito a continuación:

El primer problema que nos encontramos fue que faltaban datos para las variables `ALTITUD` y `SUPERFICIE`. Para la primera variable, lo primero que hicimos en este punto fue transformar los valores a datos numéricos. En el caso de que la variable tuviera la forma ‘`ALTITUD1-ALTITUD2`’, colocamos el valor promedio. Una vez hecho esto, los 54 valores faltantes fueron imputados utilizando la mediana de la altitud para la estación meteorológica a la que pertenecía la finca en cada caso.

Seguidamente, para resolver el problema en la variable `SUPERFICIE`, que consideramos de máxima importancia, ya que es una de las que más afectan al valor de la producción, y para la que solamente había datos en 3083 filas, tras valorar diferentes métodos para realizar la imputación, optamos por crear nuestra propia estrategia, de acuerdo con el conocimiento general que teníamos, e hicimos lo siguiente:

#### 3.1. Imputación de los datos de superficie

En primer lugar, elaboramos una tabla con la información de superficie que teníamos en los datos, dejando sólo las variables relevantes y eliminando los datos duplicados. Así, dicha tabla tenía las combinaciones posibles de las columnas `ID_FINCA`, `VARIEDAD`, `MODO` y `TIPO` y la columna con el valor de la superficie. Tras varias pruebas, llegamos a la conclusión de que en nuestros datos tenían la misma superficie aquellas filas (datos de una finca por campaña) para las que el valor de las variables mencionadas anteriormente fueran idénticas.

Una vez dispusimos de esta información, recorrimos todas las filas del *dataframe*. Si en una fila el valor de la superficie estaba vacío (`NaN`), introducíamos en él el valor de la tabla que correspondiera a la misma finca, misma variedad de uva, mismo modo y tipo de cultivo si existía una entrada en la tabla con esta misma información. Si no, se introduciría el valor de superficie de la entrada con la misma finca, misma variedad de uva y mismo modo. Si no, sólo de la entrada con la misma finca y variedad y, en último recurso, considerando sólo la finca. En el primer caso, sólo era posible encontrar una entrada en la tabla con los datos especificados, pero a partir del segundo, podía haber más de una coincidencia. Cuando más de una fila en la tabla correspondía a los datos requeridos, se calculaba la media y se introducía el valor como superficie. Tras realizar los pasos anteriores, aún quedaban 949 valores vacíos de superficie. Decidimos mantenerlos así y emplear modelos que pudiesen trabajar con datos faltantes (como se verá en la siguiente sección).

El siguiente paso fue abordar los datos meteorológicos:

#### 3.2. Agrupación de las variables meteorológicas

Disponíamos de dos *datasets* con información meteorológica, almacenados en los ficheros `DATOS_METEO.txt` y `DATOS_ET0.txt`. El primero tenía información de variables meteorológicas cada hora y el segundo, agrupaciones de datos para cada día. Ambos tenían en común la variable `ID_ESTACION`.

---

Nuestro objetivo era unir estos dos *datasets*, por lo que el primer paso fue agrupar los datos del primero por días. Las variables incluidas y sus formas de agrupación son las siguientes:

- **precip1Hour**: En este caso guardamos la información del valor máximo, mínimo y de la media de todos los datos.
- **relativeHumidity**: Nos quedamos con la mediana de los datos de todas las horas.
- **snow1Hour**: Al igual que en el caso de las precipitaciones, tomamos el valor máximo, mínimo y la media.

Una vez llegamos a este punto, pudimos obtener la tabla conjunto de los *datasets* **DATOS\_METEO** y **DATOS\_ETO** con toda la información meteorológica diaria disponible, realizando la unión con el identificador de la estación. Esto permitió añadir los valores máximo, mínimo y promedio para las variables **FeelsLikeLocalDaytime**, **FeelsLikeLocalOvernight** y **WindSpeedLocalDaytime**

La siguiente agrupación la hicimos por fases del ciclo de la vid. Añadimos una columna más al *dataset* obtenido, etiquetando cada día con la fase del ciclo de la vid a la que pertenecía. Las distintas fases son<sup>1</sup>:

- **Reposo vegetativo (R)**: (16 Nov - 4 Mar)
- **Lloro (L)**: (5 Mar - 23 Mar)
- **Brotación (B)**: (24 Mar - 20 May)
- **Floración (F1)**: (21 May - 30 May)
- **Floración (F2)**: (1 Jul - 10 Ago)
- **Maduración (M)**: (10 Ago - 31 Sep)
- **Crecimiento otoñal (C)**: (31 Sep - 15 Nov)

Además, en cada fila disponíamos de la información del año en el que los datos fueron tomados, por lo que añadimos una columna extra que indicaba el año para el cuál esa información iba a ser utilizada para obtener la predicción, ya que las predicciones de la producción se harían utilizando la información meteorológica desde Julio del año anterior hasta el último día de Junio de ese mismo año. Los datos meteorológicos previos a Julio del 2015 fueron eliminados porque no se utilizaban en ninguna predicción.

Una vez llegados a este punto, se agruparon las filas por fase del ciclo de la vid a la que correspondían. Esta agrupación se hizo utilizando la media para todas las variables.

El último paso para obtener el **dataframe** era conseguir que cada fila correspondiera a la información de un año de predicción y a una estación meteorológica, el resto de columnas tenían la información meteorológica con todas las variables consideradas para las 7 fases del ciclo de la vid.

Inicialmente optamos por incluir la información meteorológica para todas las fases del ciclo de la vid, pero tras leer algunos artículos y hacer pruebas, vimos que prácticamente todas las variables importantes estaban en las fases de reposo, lloro, brotación, floración y maduración, así que sólo mantuvimos esas columnas.

Para terminar, unimos el *dataset* obtenido con el *dataset* inicial, utilizando como clave el identificador de la estación meteorológica y el año de campaña para el que se quiere obtener la predicción y añadimos varias variables adicionales, a saber: La interacción de la superficie con el modo de cultivo y con la variedad de la uva, los retardos de primer y segundo orden de la producción, el logaritmo de la eficiencia de la finca, calculada como el retardo de la producción entre la superficie disponible y las primeras diferencias para el retardo de primer orden de la producción (estas

---

<sup>1</sup>Probamos con otra agrupación de los datos distinta, comprendida por períodos de 10 días para cada mes, pero al final la descartamos por tener una menor capacidad predictiva y ser menos eficiente.

---

dos últimas a través de una *Pipeline*, como será explicado más adelante. Finalmente, eliminamos aquellas variables que fuesen constantes y las que no añadían información adicional al modelo.

El resultado de esta parte fue la obtención del *dataset* final, que disponía de 7337 observaciones, de las que 1075 son para predecir, y 96 variables.

## 4. Elección del modelo

### 4.1. Proceso de selección

El problema planteado en este reto es un problema de regresión en el que pretendemos obtener, de la manera más acertada posible, la información de la producción de las distintas fincas en la campaña del 2022.

El proceso llevado a cabo para la elección de este modelo tuvo en cuenta todo el estudio y la información del proceso de la fase local. Dado que el *dataset* empleado en este caso supuso una reducción muy considerable en el número de variables, no llevamos a cabo en esta ocasión ninguna prueba de reducción de la dimensionalidad, aunque sí se llevó a cabo un análisis estadístico en el que se estudió la correlación entre las variables y no se tuvieron en cuenta aquellas que no aportaban información.

Nuestro objetivo en esta sección era probar diferentes modelos y seleccionar el más adecuado siguiendo los criterios detallados a continuación. Todos los modelos tienen algunos aspectos en común antes de su entrenamiento.

### 4.2. Aspectos comunes

Las variables que no aportan información relevante y es evidente desde un inicio son eliminadas. Estas son: `CAMPAÑA`, `ID_FINCA`, `ID_ZONA` e `ID_ESTACION`.

Definimos 10 *folds* para la validación cruzada con 3 repeticiones (*Repeated K-fold*) que emplearemos para la *inner evaluation* de los distintos métodos. Además, separamos un 30 % de los datos para *test* con el objetivo de obtener también una *outer evaluation* del mejor modelo.

Para obtener una primera estimación de los errores y hacer un filtro de los mejores modelos, se empleó `LazyRegressor`, que es una función de la librería `LazyPredict` que ofrece una primera estimación del error de un gran conjunto de modelos, considerando en estos los parámetros por defecto.

Todos los modelos se definen dentro de una *pipeline* en la que se incluye o no una fase de imputación (sólo para los modelos que no trabajan con valores vacíos o que proporcionaban peores resultados con ellos, emplea como estimador `KNeighborsRegressor`), una fase en la que se crean las columnas de `diff` (diferencia entre la producción hace dos años con la del año pasado tras una transformación con logaritmo) y `eficiencia` (relación entre la producción y la superficie).

Con esta primera información, pudimos desarrollar los distintos modelos candidatos y realizar su evaluación.

### 4.3. Descripción de los modelos candidatos

A continuación, veremos los modelos probados y las conclusiones obtenidas. Para todos ellos, realizamos una búsqueda de los mejores hiper parámetros a través de la librería `Optuna`. Inicialmente llevamos a cabo 100 intentos para cada uno de los estudios correspondientes a cada modelo, y después, una vez escogidos los mejores, probamos con 500 intentos para mejorar la métrica obtenida en la *inner evaluation*.

---

Fueron muchos los modelos probados y muchas las variaciones en los campos de parámetros a optimizar, a continuación incluiremos algunos detalles de los modelos con los que obtuvimos mejores resultados.

#### 4.3.1. *Gradient Boosting Regressor*

Este modelo encabezaba la lista obtenida en el `LazyRegressor`, por ese motivo fue el primer modelo considerado.

El primer aspecto a tener en cuenta es que este modelo no acepta valores vacíos, lo que supone que la *pipeline* debe incluir una fase de imputación en primer lugar.

En la búsqueda de la mejora del modelo, se trataron de optimizar tres hiper parámetros: `learning_rate`, `n_estimators` y `max_depth`.

Los resultados obtenidos en la *inner evaluation* fueron considerablemente peores que el resto de modelos, pero el principal motivo del descarte de esta opción fue el elevado coste computacional y por tanto energético que suponía, tanto que en varias ocasiones nos produjo fallos en la memoria de nuestros equipos.

#### 4.3.2. *Light Gradient Boosting Model*

Este modelo fue la segunda mejor alternativa en el análisis realizado por el `LazyRegressor`. Se trata de un modelo capaz de trabajar con valores vacíos, pero dado que los resultados obtenidos eran peores, decidimos añadir la fase de imputación para estos, con las mismas características que en el caso anterior.

En este caso, los hiper parámetros que se trataron de optimizar fueron: `n_estimators`, `num_leaves`, `max_depth` y `learning_rate`. Los resultados obtenidos para este modelo estaban muy próximos a los del mejor modelo obtenido, pero eran ligeramente peores. Además, la fase de imputación requería de bastante tiempo, lo que elevaba el coste computacional, lo que para un error similar, empeoraba la eficiencia energética de nuestro modelo.

#### 4.3.3. *Extreme Gradient Boosting Boosting Model*

A pesar de que el `LazyRegressor` no colocaba este modelo en una posición alta dentro de las opciones, decidimos probarlo porque acepta valores vacíos, y estos existen también en el conjunto de datos de predicción, lo que pensamos que podría mejorar el error obtenido.

Como producía bastantes valores negativos en la variable de salida (producción), algo que no tiene interpretación coherente, optamos por trabajar con la variable a predecir con la aplicación de un logaritmo que se deshacía convenientemente al final, de esta forma evitamos este problema.

Los parámetros a optimizar empleando `Optuna` fueron: `n_estimators`, `max_depth`, `learning_rate` y `colsample_bytree`.

Tanto el entrenamiento como la obtención de la predicción eran considerablemente más rápidas que los modelos anteriores, lo que nos permitió incrementar el número de intentos en la optimización de hiper parámetros para poder mejorar el modelo.

#### 4.3.4. *Histogram-based Gradient Boosting Regressor*

El motivo por el que se decidió probar este modelo fue, como en el caso anterior, su capacidad para trabajar con valores vacíos. Los hiper parámetros evaluados en este caso fueron: `max_leaf_nodes`, `max_depth`, `learning_rate` y `max_iter`.

En este caso, aunque podía aparecer alguna predicción con valor negativo, esto sólo ocurría para fincas con superficies diminutas, por lo que decidimos no aplicar el logaritmo a la variable de salida.

Este modelo fue el que mejores resultados nos produjo en la *inner evaluation*, y los resultados de la *outer* eran aceptablemente buenos también. Además, cabe destacar que esta alternativa constituía el modelo más rápido tanto entrenando como proporcionando predicciones. De esta forma, reducía el coste computacional y por tanto el coste energético. Por este motivo, este fue el modelo seleccionado.

## 4.4. Selección del mejor modelo

### 4.4.1. Explicabilidad y transparencia del modelo

Para elegir el mejor modelo se ha tenido como criterio sobre todo el menor error posible en las pruebas de evaluación interna dados los datos de entrenamiento basándonos en el RMSE. Si bien el modelo elegido de *Histogram Gradient Boosting* puede ser algo complejo para explicar las variables más relevantes para el modelo a la hora de predecir, la librería **dalex** de Python nos dio una visión de cuáles son las variables que mejor ilustran el valor final.

Así, las variables que mayor aumentan el RMSE si permutadas son las siguientes.

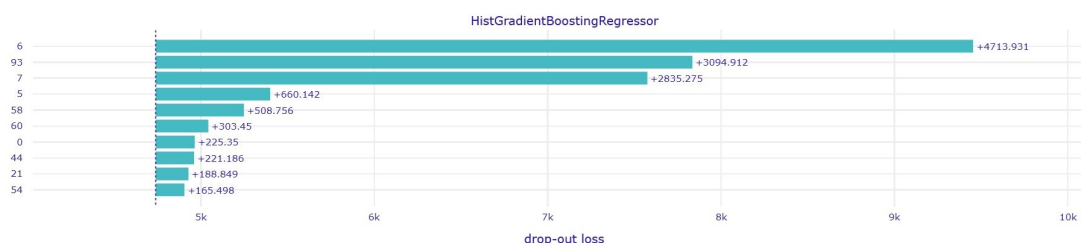


Figura 1: Variabilidad en el error dada la permutación de las variables.

Las variables que más afectan al error son las dadas por las columnas 6, 93 y 7 principalmente, que corresponden con las variables **LAG\_PROD**, **SUP\_MOD** y **LAG2\_PROD** respectivamente. A estas les siguen las columnas correspondientes a las variables '**SUPERFICIE**', '**L\_FeelsLikeLocalDaytimeMax**', '**L\_FeelsLikeLocalOvernightAvg**', '**ALTITUD**', '**F2\_medianRelativehumidity**', '**C\_FeelsLikeLocalDaytimeAvg**' y '**L\_medianRelativehumidity**'.

Debajo se puede ver un ejemplo de cómo las variables afectan a la predicción de los kilos de uva:



Figura 2: *Break down* de la predicción para un *sample* del conjunto de entrenamiento.

En este sentido, lo que afecta más a la producción son el modo de producción y la superficie del terreno, mientras que la información de los años pasados da al modelo cierta estabilidad, basando las cifras de producción en resultados anteriores.

En este sentido, se puede interpretar que la superficie y el modo de producción son lo más relevante



---

en el cultivo y que el resto de factores ambientales. La variable 92, también aparecida en la figura 2, es la variable `SUP_VAR`, que es la que relaciona la superficie con la variedad de la uva plantada. Las columnas correspondientes a las cifras 8 a 91 son variables basados en estadísticas atmosféricas de las estaciones del ciclo de la uva mencionadas en la sección 2. Estas variables afectan de manera mayor o menor a la cifra final de la predicción, llegando así a la conclusión final.

#### 4.4.2. Justicia

En el proceso de selección final se han buscado patrones para identificar aquellos sesgos que pudieran favorecer determinadas predicciones a ciertos modos de producción o ciertos tipos de uva. Con el paquete `daLEX` hemos podido comprobar también la regresión sea justa y equitativa de esta forma. No se ha encontrado ningún sesgo en los modelos relacionado con el tipo de cultivo y color de la uva. Sin embargo, sí que se han encontrado cierta relación de sesgo en términos de independencia con el modo de producción, pero sabiendo que el modo de producción está bastante ligado a la superficie según la variable `SUP_MOD`, analizada previamente como una de las más relevantes, se sigue asumiendo que el modelo es justo a la hora de producir. Simplemente, más superficie equivale a más producción y dependiendo del modo de producción se le puede asignar más o menos terreno a la viña, generando más uva al haber más plantaciones. Los modelos de regresión son insesgados si los tres criterios de independencia se cumplen (esto es así si la *Direct Density Ratio Estimation* está con valores entre 0.8 y 1.25, cumpliendo con el criterio de justicia)

Variable protegida_valor	Independencia	Separación	Suficiencia
Modo_1	33.40	1.17	1.00
Modo_2	1.16	1.04	1.00
Tipo_0	1.00	1.00	1.00
Tipo_1	1.04	1.03	1.01
Color_0	1.02	1.00	1.00
Color_1	1.00	1.00	1.00

#### 4.4.3. Sostenibilidad ecológica

Los procesos de predicción modernos son conocidos por la gran cantidad de energía que requieren. Esto hace del aprendizaje máquina un proceso que puede ser no precisamente respetuoso con el medio ambiente. En este sentido, el modelo que hemos utilizado trata de buscar un balance entre precisión (RMSE bajo), que dé resultados fiables y que no sea demasiado caro computacionalmente (*Histogram-based Gradient Boosting Regressor* ha resultado ser con diferencia el modelo más rápido, por lo que se han podido llevar a cabo, muchas más pruebas para su mejor y mejorar así los resultados). Mientras que los procesos hechos con `Optuna` pueden ser largos e intensos, una vez encontrados los hiperparámetros adecuados, el entrenamiento del proceso es rápido. Así mismo, se ha reducido el tiempo de entrenamiento como de búsqueda de hiperparámetros escogiendo las variables climáticas relevantes a mano, basándonos en fuentes fiables [1].

## Bibliografía

- [1] Estefanía González-Fernández et al. «Prediction of Grapevine Yield Based on Reproductive Variables and the Influence of Meteorological Conditions». en. En: *Agronomy* 10.5 (mayo de 2020), pág. 714. ISSN: 2073-4395. DOI: 10.3390/agronomy10050714. URL: <https://www.mdpi.com/2073-4395/10/5/714> (visitado 15-03-2023) (vid. págs. 1, 7).
- [2] vinetur.com. *Las 7 etapas de un viñedo: el ciclo vegetativo de la vid.* es. URL: <https://www.vinetur.com/2019111458471/las-7-etapas-de-un-vinedo-el-ciclo-vegetativo-de-la-vid.html> (visitado 15-03-2023) (vid. pág. 1).