Benemérita Universidad Autónoma de Puebla

Facultad de Ingeniería Química

Colegio de Ingeniería Química

Modelo basado en redes neuronales artificiales para estimar la producción de ácido cítrico

Tesis presentada para obtener el grado de: Licenciatura en Ingeniería Química

Abigail Yañez Martínez

Director de tesis: Dra. Ma. Dolores Guevara Espinosa

Co-director de tesis: Dr. Azgad Casiano Ramos

Heroica Puebla de Zaragoza, Pue., diciembre de 2024

BUAP

Oficio No. FIQ/AC/249/2024
Asunto: Registro de Tema de Tesis.

C. ABIGAIL YAÑEZ MARTÍNEZ
PASANTE DE LA LICENCIATURA EN
INGENIERÍA QUÍMICA
P R E S E N T E:

Por medio del presente me permito informarle, de la aprobación del Registro de Tema de Tesis de la **Licenciatura en Ingeniería Química** cuyo título es el siguiente:

"Modelo basado en redes neuronales artificiales para estimar la producción de ácido cítrico"

Con el siguiente contenido:

INTRODUCCIÓN

CAPÍTULO 1	ANTECEDENTES
CAPÍTULO 2	METODOLOGÍA
CAPÍTULO 3	RESULTADOS Y DISCUSIÓN

CONCLUSIONES
BIBLIOGRAFÍA

Directora de Tesis: Dra. María Dolores Guevara Espinosa.

Lo cual me permito comunicarle para su conocimiento y fines consiguientes aclarando que la vigencia de este tema será **UNICAMENTE POR UN AÑO**.

A t e n t a m e n t e
"Pensar Bien, Para Vivir Mejor"
H. Puebla de Z., a 4 de Diciembre de 2024

Dra. Valeria Jordana González Coronel
Secretaria Académica

C.c.p. Directora de Tesis: Dra. María Dolores Guevara Espinosa.
C.c.p. Archivo.

Facultad
de Ingeniería
Química

Av. San Claudio s/n, Col. San
Manuel, Ciudad Universitaria,
Puebla, Pue. C.P. 72590
01 (222) 229 55 00
Exts. 7250 y 7251

CTAI01

Benemérita Universidad Autónoma de Puebla
FACULTAD DE INGENIERÍA QUÍMICA

CIUDAD UNIVERSITARIA

ASUNTO:
AUTORIZACIÓN
IMPRESIÓN DE TESIS

Mtro. Omar Gerardo Aguirre Ibarra
Director de la Administración Escolar
De la BUAP.
Presente

Por este conducto me permito presentar a Ud. al C. pasante de la carrera de Ingeniería Química

Abigail Yañez Martínez

Quién presenta como tema de tesis:

Modelo basado en redes neuronales artificiales para estimar la producción de ácido cítrico

La cual ha sido debidamente revisada y se autoriza para su impresión correspondiente.

Sin otro particular y para los fines que se estimen conducentes reitero mi distinción.

ATENTAMENTE
"Pensar Bien, para Vivir Mejor"
H. Puebla de Z., a 3 de diciembre de 2024

Director de Tesis
Dra. Ma. Dolores Guevara Espinosa

Agradecimientos

A mis padres, Abel y Grisel, por todo su trabajo y esfuerzo día a día para darme todo lo necesario y más para hacerme llegar hasta aquí, por sus consejos, por su guía y todo su apoyo incondicional.

A mis hermanos, Irvin e Ivan por su compañía en cada paso de mi vida, por impulsarme siempre a soñar en grande, por apoyarme siempre en todo lo que he necesitado.

A mi novio, Hugo, por ser el pilar de este trabajo, por pasar días enteros trabajando, por todo su apoyo, motivación, compañía y amor, este trabajo no habría sido posible de no ser por él.

Índice

Resumen

La creciente generación de residuos de procesos agroindustriales ha impulsado el interés por su aprovechamiento, con la fermentación mediante *Aspergillus niger* emergiendo como una alternativa destacada para la producción de ácido cítrico. Sin embargo, uno de los principales desafíos en este campo es la descripción precisa de la cinética de crecimiento en estos sistemas. Este estudio se centró en entrenar una red neuronal que pudiera modelar las curvas de crecimiento y estimar la concentración final de ácido cítrico. Para ello, se utilizaron datos experimentales obtenidos a partir de simulaciones con software especializado, cuyos resultados se compararon con modelos matemáticos como el modelo de Monod y el modelo logístico. Se desarrolló una red neuronal para estimar la concentración final de ácido cítrico basándose en las concentraciones iniciales de biomasa y sustrato. El entrenamiento resultó en un coeficiente de determinación de 0.990 para la estimación de la concentración final.

Introducción

Este trabajo de investigación se centra en el desarrollo y validación de un modelo matemático en el lenguaje de programación Python, que simule el proceso de fermentación para la producción de ácido cítrico utilizando la cepa de *Aspergillus niger*, junto con una simulación en Super Pro Designer que valide el funcionamiento del modelo matemático y entrenar modelos predictivos en Python para la estimación de la producción de ácido cítrico. El ácido cítrico, un compuesto orgánico de gran importancia industrial se encuentra con amplia aplicación en la industria alimentaria, farmacéutica y de detergentes. La producción tradicional de ácido cítrico se basa en procesos fermentativos, donde microorganismos como *Aspergillus niger* convierten fuentes de carbono (sustratos como glucosa, fructosa, sacarosa, etc.) en ácido cítrico.
A través de la simulación computacional y matemática, se buscó comprender los mecanismos que subyacen al crecimiento del hongo y la producción de ácido cítrico, así como evaluar el impacto de diferentes variables operativas, como la concentración de sustrato, la temperatura y el tiempo de reacción en el rendimiento del proceso. El modelo matemático desarrollado permite predecir el comportamiento del sistema fermentativo bajo diversas condiciones, facilitando la optimización de los parámetros de operación y el diseño de estrategias de control más eficientes.
Los resultados obtenidos en este estudio contribuyen al avance del conocimiento en el área de la biotecnología industrial, proporcionando una herramienta valiosa para el diseño y escalado de procesos de producción de ácido cítrico.

Justificación

La creciente demanda global de ácido cítrico, impulsada por su amplia aplicación en las industrias alimentaria, farmacéutica y de detergentes, ha renovado el interés por desarrollar procesos de producción más eficientes y sostenibles. A pesar de los avances alcanzados en la fermentación para producir ácido cítrico, todavía existen oportunidades para optimizar los procesos y reducir los costos asociados a su producción. La simulación computacional se ha consolidado como una herramienta clave para el análisis y diseño de procesos biotecnológicos. Sin embargo, la implementación de modelos matemáticos en la fermentación de ácido cítrico aún enfrenta desafíos debido a la complejidad del proceso y la escasez de datos experimentales disponibles. Este estudio tiene como objetivo avanzar en el conocimiento mediante el desarrollo de un modelo matemático que describa con precisión el comportamiento de un fermentador en la producción de ácido cítrico. Los resultados obtenidos permitirán identificar las variables cruciales que afectan el proceso fermentativo y optimizar las condiciones operativas para maximizar la producción de ácido cítrico. Además, este modelo podrá utilizarse como una herramienta de diseño para el escalado de procesos y la integración de nuevas tecnologías de producción.

Antecedentes

En este apartado se mencionarán brevemente trabajos en los que las redes neuronales hayan sido aplicadas para estimar concentraciones o algún parámetro de algún determinado compuesto.

(López de la Maza, Zumalacárregui de Cárdenas, Pérez, & Llanes, 2018), realizó la investigación: **Obtención de un modelo neuronal para la estimación de la concentración de etanol en la destilería Héctor Molina** en la Universidad Tecnológica de La Habana “José Antonio Echeverría” (CUJAE), Cuba, llegando a la conclusión de que con una red neuronal multicapa con una estructura de 4-6-1 fue capaz de estimar satisfactoriamente la concentración de etanol final de su etapa de fermentación, teniendo este entrenamiento un coeficiente de determinación de 0.9377

(Urtubia & Román Russell, 2010), realizó la investigación titulada: **Utilización de Redes Neuronales Artificiales para predecir problemas de fermentaciones vínicas** en la Universidad de Valparaíso, llegando a la conclusión de una red neuronal puede predecir de manera correcta problemas en el proceso de fermentación de vinos antes de que el proceso llegue a las 72 horas.

(Hernández Regalado, López de la Maza, Pérez Ones, & Zumalacárregui de Cárdenas, 2019), realizó el trabajo: **Modelación neuronal de la etapa de fermentación en una ronera cubana**, en este trabajo se modeló el proceso de fermentación por medio de dos

redes neuronales, una con un diseño más “simple” y la segunda con una estructura multicapas.

(Moya Almeida, Diezma Iglesias, & Correa Hernando, 2023) desarrolló el trabajo llamado: **Desarrollo de un sistema de predicción y control de solventes durante la fermentación de cerveza con S. Cerevisiae mediante redes neuronales y lógica difusa** en la Universidad Politécnica de Madrid, concluyendo que se tuvo un entrenamiento de las redes neuronales, notando que no se encuentran diferencias significativas entre los datos predichos y los experimentales, obtuvieron coeficientes de determinación de 0.941 y 0.997.

Planteamiento del problema

Los diversos procesos agroindustriales generan una gran cantidad de desechos, los cuales son dejados a la intemperie para pudrirse, desechados a la basura, o incluso son quemados a cielo abierto, generando así un aumento en la contaminación ambiental. Este tipo de residuos pueden ser ocupados como materia prima para otros procesos de gran importancia, como lo es la producción de ácido cítrico por medio de fermentación. Este proceso fermentativo puede tener como sustrato cáscaras y pulpa de diversas frutas, suero de leche proveniente de la producción de productos lácteos, diversos tipos de melazas y jarabes.

Objetivo general:

- Desarrollar un modelo matemático en lenguaje de programación Python que sea capaz de modelar los datos del Software Super Pro Designer, usando estos para poder hacer estimaciones por medio de modelos predictivos.

Objetivos particulares:

- Desarrollar en Python el modelo matemático para describir el comportamiento del fermentador.
- Manipular fluidamente el Software Super Pro Designer.
- Modelar el fermentador para ácido cítrico en Super Pro Designer.
- Comparar los datos obtenidos en Python y Super Pro Designer.
- Optimizar los modelos para obtener resultados más cercanos entre estos.
- Realizar diversas corridas experimentales para generar una base de datos amplia de entrenamiento para los modelos.
- Programar y entrenar una red neuronal que estime la producción de ácido cítrico
- Programar y entrenar diversos modelos predictivos.
- Poner a prueba los modelos.
- Optimizar los modelos.

- Poner en prueba la red neuronal vs los modelos para comprobar su eficiencia.

Marco teórico

Fermentación

La fermentación es un proceso biológico por el cual, microorganismos descomponen materia orgánica de diversas fuentes, generalmente aquellas que tienen grandes cantidades de carbono y azúcares, como lo son la glucosa, melazas, sueros provenientes de los lácteos, etc., para dar lugar a la producción de nuevos compuestos que pueden tener una gran utilidad como materia prima primaria.

Si bien, no es una técnica nueva, no hay mucha investigación y mejoría con sus procesos.

Fermentaciones industriales

En la industria, muchos productos son llevados a cabo gracias a la fermentación. Estos se producen en unos recipientes llamados biorreactores o fermentadores, diseñados especialmente para que los microorganismos tengan las condiciones óptimas para su crecimiento y dar lugar a grandes cantidades del producto deseado.

Tipos de fermentación industrial

Las Fermentaciones Industriales se pueden clasificar en tres grandes tipos, el factor decisivo para la utilización de algún tipo u otro está directamente relacionada con el tiempo de producción de cada industria, el proceso a realizar y la manera de operación de cada industria, a continuación, se mencionan los 3 tipos:

Fermentación discontinua o en tanda: En este tipo de fermentación, solo se realiza una alimentación en todo el proceso y se obtiene una salida.

Fermentación semicontinua o “fed-batch” (con alimentación a intervalos): La diferencia con el tipo de fermentación anterior es que aquí se realizan más de una alimentación durante todo el proceso, pero solo una salida.

Fermentación continua: Aquí se cuenta con un balance de masa, que todo lo que se alimenta será igual a lo que sale, se tienen diversas alimentaciones a lo largo del proceso, pero manteniendo un flujo constante todas.

Existen diferentes **tipos de fermentación**, con características particulares dependiendo del producto a generar, se mencionan las principales:

- Fermentación alcohólica
- Fermentación láctica
- Fermentación acética
- Fermentación maloláctica

- Fermentación propiónica
- Fermentación butírica

Usos de la fermentación

Entre las diversas áreas en las que se puede utilizar la fermentación se pueden mencionar los siguientes:

- Industria alimentaria y de bebidas
- Biotecnología y farmacéutica
- Energía y medio ambiente
- Productos químicos industriales

Cinética de la fermentación

En los sistemas de cultivo de alimentación discontinua, la tasa de crecimiento de las células bacterianas se puede definir mediante la siguiente expresión: (Najafpour G. D., 2015)

$$\frac{dX}{dt} = \mu X \qquad \boldsymbol{Ec.1}$$

En dónde:

$$\frac{dX}{dt} = tasa\ de\ crecimiento\ bacteriano, masa/volumen\ unitario\ *\ tiempo$$
$$\mu = tasa\ de\ crecimiento\ específico, tiempo^{-1}$$
$$X = concentración\ de\ microorganismos, masa/\ volumen\ unitario$$

Como en cualquier reacción química, se tiene la presencia de un reactivo que frene o potencie la reacción, mejor conocido como reactivo limitante, este, por definición es el compuesto que cuando se agota, la reacción se frena, lo mismo si se pone un poco concentración, la reacción puede terminar muy rápido, no logrando la cantidad de producto deseado.

En el caso de la reacción por fermentación, el conocido reactivo limitante vendrá siendo el sustrato, ya que, este actúa como alimento para la biomasa, entonces si se tiene muy poco sustrato, la biomasa no podrá crecer tanto y por ende, no dará una gran concentración de producto.

Por eso, para modelar el crecimiento de la biomasa, se hace en función del sustrato: (Eddy, 1995)

$$\mu = \mu_{max} \frac{S}{K_S + S} \qquad \boldsymbol{Ec.2}$$

Donde:

$\mu = tasa\ de\ crecimiento\ específico, tiempo^{-1}$
$\mu_{max} = máxima\ tasa\ de\ crecimiento\ específico, tiempo^{-1}$
$S = concentración\ del\ substrato\ que\ limita\ el\ crecimiento, masa/unidad\ de\ volumen$
$K_S = constante\ de\ velocidad, masa/unidad\ de\ volumen$

Como parte complementaria a la cinética de reacción, se hizo uso del modelo de crecimiento logístico, o también conocido como el modelo de Verhulst, el cuál es una modificación en la manera en la que se representan las curvas. Generalmente, se representa un crecimiento exponencial del crecimiento de la biomasa, lo que hace el modelo logístico es tener una tendencia exponencial, pero, llegando al punto máximo de crecimiento, modela una ralentización en el crecimiento, observando casi una línea recta, que físicamente, en el fermentador esto representa que se ha consumido la mayor parte de sustrato, y lo que resta no es lo suficiente para hacer que la biomasa siga creciendo aceleradamente, considerando esto un estado estacionario, pues las variaciones en las concentraciones de las sustancias de interés son mínimas.

$$\mu = \mu_{max}\left[1 - \frac{X}{X_m}\right] \qquad \boldsymbol{Ec.3}$$

Uniendo el modelo de Monod y el modelo logístico, se obtiene una nueva ecuación que modela el crecimiento de la biomasa de una manera más cercana a la realidad.

$$\frac{1}{X}\frac{dX}{dt} = \mu_{max}\left(\frac{S}{K_S + S}\right)\left(1 - \frac{X}{X_m}\right) \qquad \boldsymbol{Ec.4}$$

El rendimiento está definido como la relación de la biomasa con la masa del sustrato: (Najafpour G. D., 2015)

$$\frac{dS}{dt} = -\frac{1}{\gamma_{x/s}} * \mu X \qquad \boldsymbol{Ec.5}$$

En dónde:

$\frac{dS}{dt} = tasa\ de\ producción\ de\ biomasa$
$\mu = tasa\ de\ crecimiento\ específico, tiempo^{-1}$
$X = concentración\ de\ microorganismos, \frac{masa}{volumen\ unitario}$
$\gamma_{x/s} = rendimiento\ de\ la\ biomasa$

Del mismo modo, la tasa de formación del producto se define como: (Najafpour G. D., 2015)

$$\frac{dP}{dt} = \gamma_{p/s} * \mu X \qquad \boldsymbol{Ec.6}$$

En dónde:

$$\frac{dP}{dt} = tasa\ de\ producción\ de\ producto$$
$$\mu = tasa\ de\ crecimiento\ específico, tiempo^{-1}$$
$$X = concentración\ de\ microorganismos, \frac{masa}{volumen\ unitario}$$
$$\gamma_{x/s} rendimiento\ del\ producto\ con\ respecto\ a\ la\ biomasa$$

Etapas de crecimiento

Las principales etapas de crecimiento en un proceso de fermentación son:
Tiempo de inoculación: En este tiempo, los microorganismos se colocan en el fermentador e inician su crecimiento.
Crecimiento exponencial: Ya que los microorganismos están ambientados en las condiciones óptimas para su crecimiento, este se llevará a cabo de una manera muy rápida, van consumiendo el sustrato que tengan como alimento, dando simultáneamente el producto deseado, graficando esto dará la forma de una curva exponencial.

Estado estacionario: Una vez que los microorganismos llegan a su punto máximo, llegan a un punto de equilibrio, en el que ya no crecen más, pero no disminuye su concentración, sino, se queda constante, formando una línea recta.

Conforme va aumentando la concentración de biomasa o microorganismos en el fermentador, la concentración del sustrato irá bajando, así mismo, la concentración del producto irá aumentando en la misma proporción que la biomasa. El objetivo principal de una buena fermentación es lograr el agotamiento del sustrato para producir la mayor cantidad de producto posible.

Ácido cítrico

El ácido cítrico se encuentra presente de manera natural en una basta cantidad de cáscaras de diversas frutas, sin embargo, si se extrae directamente de estas cáscaras se obtiene muy poca cantidad y el procedimiento es muy costoso, sin mencionar que se requiere de una gran cantidad de cáscaras para poder obtener unos cuantos ml de ácido cítrico, por lo que se han buscado diversos métodos para poder obtener más producto con una menor cantidad de producto.

Usos principales del ácido cítrico

Los usos del **ácido cítrico** son muy diversos. Suele utilizarse en la **industria cosmética, alimentaria y farmacéutica**. En la industria cosmética es usado debido a su gran propiedad antioxidante, situándolo en una gran cantidad de cremas antimanchas y con propiedades despigmentantes, en la industria farmacéutica se usa unos varios suplementos de vitamina C, también puede ayudar a prevenir la formación de cálculos renales, finalmente en la industria alimentaria tiene un gran uso como conservante, mejorando también el sabor y textura de diversos alimentos.

Aspergillus Niger

Aspergillus niger es la especie más común de aspergillus.

Este hongo se encuentra naturalmente en algunas frutas, causando un moho negro, acelerando la descomposición natural de estos.

Pero para el proceso de fermentación es ampliamente utilizado, ya que de los hongos, levaduras y bacterias que pueden dar lugar al proceso de fermentación, la cepa de Aspergillus Niger es el que presenta una mayor eficiencia en el consumo del sustrato para dar lugar al producto deseado.

Machine Learning

El Machine learning puede ser definido como aquel proceso por el cual un modelo matemático puede ser entrenado para aprender a realizar diversas actividades y poder eficientar procesos que si un humano lo realizara le llevaría el doble de tiempo.

(Müller, 2017) menciona que: “Utilizando el aprendizaje automático, con simplemente presentar un programa con una gran colección de imágenes de caras es suficiente para que un algoritmo determine qué características se necesitan para identificar una cara”.

Esto solo nos habla un poco del gran alcance que la inteligencia artificial tiene en estos días, ya que se encuentra presente en un sinfín de actividades que nos hacen la vida más práctica.

Evaluación de los resultados del aprendizaje

Algo fundamental del Machine learning es la constante evaluación de los modelos creados, ya que claro, estos pueden estar propensos a cometer errores, y es trabajo del programador poder evaluar de manera correcta estos errores para poder así eficientar el modelo y sea más exacto y confiable.

La métrica de evaluación más común es el coeficiente de determinación, indica que tan exacto es el modelo, sus valores van de 0 a 1, cuanto más cercano sea su valor a 1, se dice que es más exacto en las predicciones realizadas.

Pero se considera un valor aceptable para un sistema complejo a partir de 0.80.

Modelos predictivos

K-Nearest Neighbors

Este modelo predictivo es un modelo bastante sencillo pero eficiente para conjunto de datos con alguna relación entre ellos o cierta tendencia, pues que su manera de funcionamiento es que, del dato que se desea predecir, busca algún punto cercano entre los datos de entrenamiento que se le ha sido brindado, calcula su distancia Euclidiana de todos los puntos de los datos de entrenamiento y busca el más cercano, dando así una estimación en el valor.

Se le indica el número de "vecinos" (datos cercanos al punto que se quiere estimar) que se quiere que considere para realizar sus cálculos.

Regresión lineal

¿Qué es?

Probablemente, este modelo predictivo es el más sencillo con respecto a su matemática de entrenamiento y a su facilidad de uso.

Su fase, como su nombre lo indica, es partir de la ecuación de la recta:

$$y = mx + b \quad \boldsymbol{Ec.7}$$

(Bagnato, 2022) define estas variables como; "Donde "y" es el resultado, "x" es la variable, "m" la pendiente (o coeficiente) de la recta y b la constante o también conocida como el "punto de corte con el eje Y" en la gráfica (cuando X=0)".

¿Cómo funciona?

Su funcionamiento es bastante simple, pues este busca una tendencia líneal en los datos de entrenamiento y traza una línea recta imaginaria, calcula el error cuadrático medio (MSE), y se optimiza hasta que encuentra el error más pequeño.

Red neuronal

¿Qué es una red neuronal?

Las redes neuronales pertenecen a un grupo de Machine learning llamado Deep learning o aprendizaje profundo, que su entrenamiento se enfoca en el aprendizaje de actividades complejas, tratando de replicar el proceso de aprendizaje de un cerebro humano.

¿Cómo funcionan las redes neuronales?

Como se mencionó arriba, se busca replicar el funcionamiento de una neurona humana, tratando de imitar el proceso de sinapsis, que es el proceso por el cual un humano aprende, ya que es la conexión y comunicación de información de una neurona a otra, generando así, el aprendizaje.

Entonces, una red neuronal artificial lo que busca es generar esta conexión entre neuronas.

Arquitectura de una red neuronal simple

La estructura básica de una red neuronal se conforma de 3 capas, la primera denominada capa de entrada es por donde se va a recibir la primera información, haciendo una analogía a la neurona humana, este sería el estimulo o la información que se logra captar y percibir del mundo exterior, seguida de esta capa viene la llamada capa oculta,o también llamados preceptrones, la cual recibe e interpreta la información recibida de la capa de entrada, para finalmente pasarla a la capa final, la capa de salida, siendo esta por donde sale la respuesta de la interpretación de la neurona de la capa oculta.

Básicamente es similar a como funciona la interpretación de lo que vemos, los ojos vendrían siendo la capa de entrada, pasando la información que pueden ver y percibir, este estimulo es pasado a las neuronas que procesan lo que ven y emiten una conclusión, pasándola a la capa de salida, que sería cuando ya entendemos lo que estamos viendo.

Arquitectura de una red neuronal profunda

Así como hay estructuras de redes neuronales muy simples, están las que tienen una composición más elaborada, como lo son las redes neuronales profundas.

En estas se pueden tener múltiples capas de entradas unidas a un sinfín de perceptrones en cada capa, generando una red mucho más compleja, ya que, como se muestra en la Figura 1, cada capa está unida a cada perceptrón, así, la información presente a en cada capa y precpetrón es transmitida a toda la red.
Aquí se tiene los denominados pesos, que son justamente las conexiones entre cada red, estos pesos tienen un valor numérico positivo, que dependiendo el valor que se le asigne a este, dependerá la importancia de la información presente en esa conexión, pudiendo así generar un sinfín de entrenamientos, puesto que se puede jugar con los pesos hasta lograr el aprendizaje deseado, así como se le puede dar “más importancia” a alguna conexión, también se puede anular la comunicación entre dos preceptrones, dándoles un peso de 0.

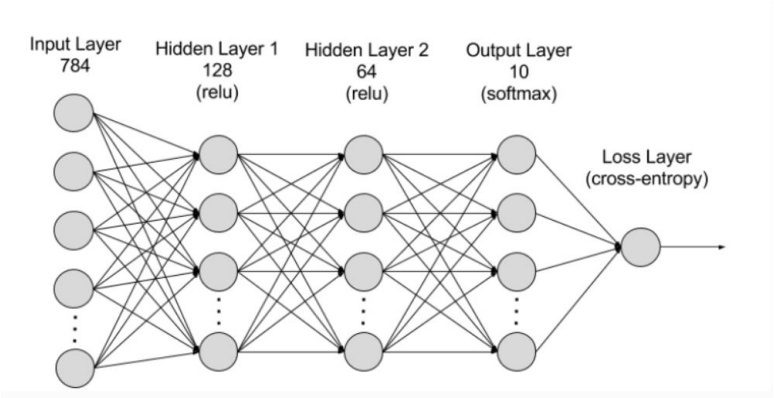


Figura 1. Estructura de una red neuronal. Fuente: (AWS, 2024)

Metodología

SuperPro Designer

Como primer objetivo de este trabajo, se buscó en diferente literatura información que pudiera ser de utilidad para modelar la producción de ácido cítrico, encontrando así que las mejores condiciones operativas provienen de tres fuentes, las cuales son: *Biochemical Engineering and Biotechnology* , *Production of citric acid by Aspergillus niger using cane molasses in a stirred fermentor y Citric Acid Production Process Modeling and Evaluation with SuperPro Designer*.
Una vez que se tenía definida la información, se procedió a simular el proceso en el software SuperPro Designer. Para poder manipular este software fue necesario ver varios videos tutoriales, ya que se dominaban las acciones básicas se inició la simulación.

El procedimiento consistió en ingresar primero los compuestos necesarios para la simulación, los cuales son: el sustrato, que al igual que en la literatura consultada, fue glucosa, la biomasa, que en este caso, el software ya trae pre cargada una sustancia denominada “biomasa”, con la siguiente composición $\boldsymbol{CH_{1.8}O_{0.5}N_{0.2}}$, investigando más a fondo se descubrió que esta fórmula química no corresponde a un compuesto específico, sino a una **fórmula empírica promedio** utilizada comúnmente para representar la **biomasa microbiana**. En este caso particular, se emplea para describir la composición elemental promedio de un microorganismo, en este caso, un hongo.

Permite representar de manera sencilla la composición elemental promedio de un organismo complejo como un hongo, es utilizada en cálculos estequiométricos para determinar las cantidades de nutrientes necesarios para el crecimiento microbiano y los productos generados en los procesos metabólicos y también permite comparar la composición elemental de diferentes microorganismos.

Como la biomasa recomendada por la diferente literatura disponible para tratar la fermentación es usar el hongo Aspergillus Niger, se consultó si esta biomasa que viene

precargada en el software podría servir para representar al hongo, ya que de este no hay información con respecto a su estructura y composición debido a su complejidad de análisis, hallando que la fórmula $CH_{1.8}O_{0.5}$ $N_{0.2}$ es una representación simplificada, pero útil, para describir la composición elemental promedio de la biomasa microbiana, incluyendo hongos como el Aspergillus niger, ya que, la mayoría de los microorganismos, incluidos los hongos, tienen una composición elemental relativamente similar. Esta fórmula proporciona una estimación razonable de las proporciones de carbono, hidrógeno, oxígeno y nitrógeno presentes en el Aspergillus niger.

Siguiendo con los componentes a trabajar en el software, los siguientes en la lista son el agua, aire, los nutrientes necesarios para el crecimiento, como lo son el cobre y el sulfato de amonio, CO_2, y, finalmente, el mismo ácido cítrico.

Ya que se tienen registrados los componentes, se pasó al entorno de simulación, en donde se selecciona un fermentador tipo batch, se colocan las entradas y salidas necesarias en este, las cuales son tres entradas y una salida, en la primera entrada se colocó la biomasa, el agua y la glucosa a la concentración recomendada en literatura, que para la biomasa es 0.1 g/l, 20 g/l para el sulfato de amonio y para la glucosa es 150 g/l, en la segunda entrada se coloca una corriente de agua con cobre, 120 g/l para el cobre y la tercera entrada es una corriente de aire para poder llevar a cabo el proceso de fermentación, dicha corriente cuenta con una ayuda por parte del software, la cual es que dependiendo del proceso de fermentación a llevar a cabo y a la cantidad de materia alimentada, se autoajusta el flujo másico de aire requerido, por último, en la salida, se le indica al software que a través de ella saldrá todo lo que se logre producir en el fermentador, incluyendo los compuestos que no hayan reaccionado completamente.

Definidas las entradas y salidas se pasa a armar la secuencia de operaciones presentes en el fermentador, el software proporciona muchas opciones como cristalización, reacciones de diversos tipos, enfriamiento, calentamiento, etc., en este caso, las operaciones necesarias fueron, las dos cargas de las corrientes de entrada, calentamiento, la reacción de fermentación por modelo cinético y la descarga por la corriente de salida.

Se configuraron todas las operaciones de manera que, las corrientes de entrada entren con los valores inicialmente definidos, que se calienten los compuestos hasta una temperatura óptima para el crecimiento de la biomasa (30-35°C), en el modelo cinético de la reacción se ingresó la reacción propiamente, la cual es que la glucosa, con el cobre y el sulfato de amonio, por medio del calor suministrado dará origen a la biomasa, el ácido cítrico, agua y CO_2 , también se escogió el modelo para el sustrato, que como se explicó

anteriormente, se trabajó con la ecuación de Monod, dándole al software el valor de $ks = 250\frac{g}{l}, \mu_{max} = 0.25, \alpha = 10$, finalmente todo esto es sacado por la corriente de salida.

A continuación, se muestran los coeficientes estequiométricos de la reacción de fermentación ingresada al software, obtenida del trabajo de (Carmichael & Petrides, 2020):

2 (Ammonium sulfate) + 180 (Glucose) + 12 (Copper) + 56 (Oxygen) → 16 (Biomass) + 30 (Carbon Dioxide) + 154 (Citric Acid) + 50 (Water)

Ya que, en la literatura consultada, recomiendan un tiempo de reacción de 5 a 6 días, se configuró el sistema para que dure un proceso total de 120 h.

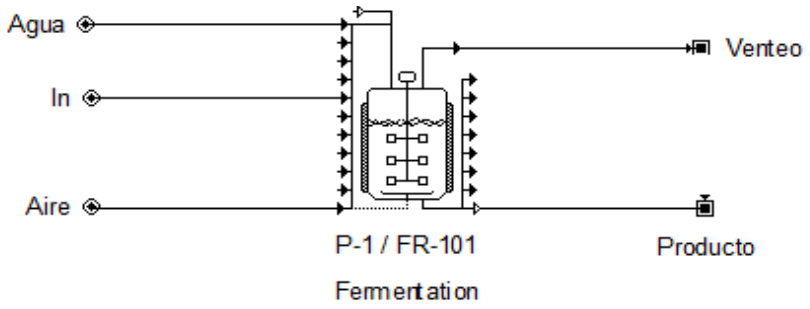


Figura 2. Modelado del fermentador en SuperPro Designer. Fuente: SuperPro Designer

Modelo en Python

Ya que se tuvo lista la parte experimental por medio del software, se pasó a programar el modelo matemático en Python. Para esto, solo fue necesario programar las ecuaciones mostradas en el marco teórico, darle las condiciones iniciales para las ecuaciones diferenciales, que fueron: 150 g/l para la concentración de sustrato, 0.1028 g/l de biomasa y 0 g/l para el producto, poner los rendimientos de la biomasa con respecto al sustrato, del producto con respecto al sustrato (estos fueron calculados de la siguiente manera: $\gamma_{x/s} = \frac{\Delta x}{\Delta s}, \gamma_{p/s} = \frac{\Delta p}{\Delta s}$), dando como resultado 0.08832 y 0.85012 respectivamente y llamar una librería para resolver ecuaciones diferenciales, llamada Odeint, la cual utiliza un sistema complejo para poder resolver todo tipo de ecuaciones diferenciales. Ya que estuvieron listas las ecuaciones, se escribió la sintaxis para graficar el valor de la glucosa, la biomasa y el ácido cítrico con respecto al tiempo y poder compararla con la gráfica obtenida en el software.

Entrenamiento en Orange

Para el entrenamiento de los modelos predictivos, se utilizó el software llamado Orange. Su uso es mucho más sencillo e intuitivo que los demás softwares de programación, ya que su sintaxis se basa en unos símbolos denominados “widgets”, tomando el ícono que se necesita y pegándolo en el área de trabajo, estos widgets se pueden unir los unos con los otros, formando así diferentes combinaciones de funciones, la gran ventaja de Orange es que nos permite cargar las bases de datos directamente desde algún documento en Excel, alguna liga, etc.

Es importante recalcar que todos los modelos y funciones que tiene Orange son con una sintaxis hecha en Python que ya está cargada en su base.

El procedimiento que seguir aquí para hacer el modelo predictivo es, primero insertar un widget para cargar los datos, una vez que el widget ya está en el área de trabajo, se le da clic al ícono y se selecciona desde donde se cargará el archivo, se selecciona el archivo y ya está cargado, lo siguiente que es muy útil es poner el widget de tabla y unirlo con el widget de los datos, esto permite visualizar los datos en una tabla. Después, se coloca el widget llamado “select data”, esta herramienta permite hacer la separación de cuáles serían las variables predictoras, el objetivo a predecir y también da la opción de omitir o ignorar algunas variables que no se quiere incluir en el modelo predictivo, y se une con el ícono de la tabla, seguido de esto, se coloca el widget del modelo predictivo a usar, ya sea KNN, regresión lineal, alguna red elástica o una red neuronal y dependiendo del modelo predictivo que se haya seleccionado es la información que se le deberá brindar a Orange para que pueda programarlo, antes de unir este widget con algún otro, es necesario insertar un widget llamado “test and score”, en este widget se unirán los datos seleccionados y el modelo que se haya seleccionado, así, aquí podremos observar varios parámetros que nos permiten evaluar la eficiencia del modelo, como es el que hemos ocupado R^2, o el error por mínimos cuadrados, entre otras, y así de sencillo es programar un modelo predictivo en el software Orange. A diferencia de Python, aquí podemos ocupar imágenes como datos para modelos predictivos.

Siguiendo el procedimiento anteriormente mencionado se programó modelos predictivos con KNN y regresión lineal para la predicción de ácido cítrico con los siguientes datos de entrenamiento:

Tabla 1. Datos de entrenamiento para todos los modelos.

Run	Biomass (g/L)	Citric Acid (g/L)	Glucose (g/L)
1	0.1028	127.859	150.4354
2	0.1081	82.7267	98.2388
3	0.1045	113.09	133.05

4	0.259	170.269	200.7077
5	0.1118	130.6086	153.6791
6	0.2622	160.2212	188.7692
7	0.0966	224.7226	265.6177
8	0.1064	96.872	114.1906
9	0.1158	143.7229	169.1999
10	0.1277	82.9374	98.237
11	0.1109	180.7611	213.1873
12	0.1231	115.9081	136.3332
13	0.1123	168.6223	198.7511
14	0.1245	106.6028	125.4159
15	0.109	192.594	227.2769

Posterior a eso, se programaron dos redes neuronales, una que es capaz de modelar las curvas del proceso de fermentación para la producción de ácido cítrico, y, la segunda que estima la concentración final de ácido cítrico tomando solo la concentración de biomasa y sustrato alimentada al fermentador.

Para la primera red se administró una corrida experimental que contenía la concentración de glucosa y biomasa inicial, ácido cítrico final, variándola en intervalos de 6 horas, hasta alcanzar el fin del proceso en SuperPro Designer a las 120 horas.

Se usó el intervalo de tiempo del proceso de fermentación, las concentraciones de biomasa y sustrato como variables predictoras de la red, el ácido cítrico toma el papel del target u objetivo a estimar, con una partición de los datos de 70% para datos de entrenamiento y 30% para datos de prueba.

Tabla 2. Datos de entrenamiento para la primera red.

Time (h)	Biomass (g/l)	Citric Acid (g/l)	Glucose (g/l)
0	0.1028	0	150.4354
6	0.1695	0.642	149.6797
12	0.2789	1.6953	148.441
18	0.4574	3.4134	146.418
24	0.746	6.1911	143.1527
30	1.2056	10.6149	137.9465
36	1.9197	17.489	129.8469
42	2.9868	27.76	117.7811
48	4.4814	42.1462	100.8494

54	6.3779	60.4006	79.3726
60	8.4594	80.436	55.8193
66	10.3443	98.5798	34.4775
72	11.7232	111.8519	18.8604
78	12.5568	119.8758	9.4269
84	12.9955	124.0985	4.4602
90	13.2084	126.148	2.0499
96	13.3074	127.1005	0.929
102	13.3525	127.5352	0.4183
108	13.3729	127.7313	0.1878
114	13.3821	127.8194	0.0842
120	13.3862	127.859	0.0377

La red fue establecida con una capa de entrada con tres perceptrones, seguida de una capa oculta con diez perceptrones y una capa final con un perceptrón de salida. La función de activación que se ocupó fue Rectified Linear Unit (ReLu) y el optimizador L-BFGS-B para la actualización de los pesos y sesgos con una tasa de aprendizaje de 0.01, con un total de mil épocas.

Para la segunda red se entrenó con 15 corridas experimentales (Tabla 1), las cuales contenían la concentración de glucosa y biomasa inicial, así como el ácido cítrico obtenido al final de la corrida experimental en SuperPro Designer. Se usó las concentraciones de biomasa y sustrato como variables entrenadoras de la red y la concentración del ácido cítrico como el target a estimar. La partición de los datos alimentados fueron 70% para datos de entrenamiento y 30% para datos de prueba.

La red se estableció con una capa de entrada con dos perceptrones, seguida de cinco capas ocultas con diez perceptrones cada una y una capa final con un perceptrón de salida. La función de activación que se ocupó fue Rectified Linear Unit (ReLu) y el optimizador Adam para la actualización de los pesos y sesgos con una tasa de aprendizaje de 0.1, con un total de mil épocas.

Modelo predictivo en Python

El siguiente paso que se trabajó fue un modelo predictivo en Python, que, dándole las condiciones de biomasa y sustrato estime la cantidad de ácido cítrico a obtener, utilizando los datos de entrenamiento de la Tabla 1.

Para la construcción del modelo predictivo se inició por cargar los datos a Python, usando una librería llamada sklearn.datasets que nos permite visualizar los datos en una tabla,

pudiendo así identificar cuáles serán nuestras variables predictoras (feature) y cuál es el objetivo (target) que se desea estimar, que en este caso es la producción de ácido cítrico.

Una vez cargados los datos, se le indica a Python cuales son los que se quiere que utilice como variables predictoras cantidad de biomasa y glucosa que utilizará para predecir la cantidad de ácido cítrico a producir bajo esas condiciones.

De la librería sklearn.model_selection importando su acción train_test_split se realiza la partición aleatoria de nuestro conjunto total de datos en dos grupos, los datos de entrenamiento y los datos de prueba, en un porcentaje de 70 y 30% respectivamente

Antes de realizar la división, la función train_test_split baraja el conjunto de datos utilizando un generador de números pseudoaleatorios

Los datos son mezclados para asegurar que los datos de prueba contengan datos de todas las variables predictoras. Para asegurar que se obtenga la misma salida si se ejecuta la misma función varias veces, proporcionamos el generador de números pseudoaleatorios con una semilla fija utilizando el parámetro *random_state*.

Una vez particionados los datos se nombran las variables *X_train, X_test, y_train, y_test*; siendo las “x” los datos brindados a Python y las “y” los datos predichos por el modelo.

El primer modelo de predicción que se usó es el denominado *“k-Nearest Neighbors”*, el cual ha sido abordado en el marco teórico.

Como primera prueba, se entrenó el modelo KNN con un valor igual a 1 y se pasa a la evaluación del modelo.

Se siguió el mismo procedimiento para entrenar ahora un modelo predictivo basado en la regresión lineal.

Una vez que se entrenó estos dos modelos predictivos, se pasó a entrenar la red neuronal que ya está establecida en Orange. Se generó un código en el cual se conocerá la concentración final de ácido cítrico con la red neuronal entrenada con la base de datos experimentales. En este código, se ingresan las concentraciones iniciales de biomasa y glucosa. Para su entrenamiento fueron necesarias librerías tales como como *Tensorflow*, *Numpy* y *Keras*, esta red fuera armada con base en lo entrenado en Orange. Se registró el Mean Squared Error de las diversas iteraciones realizadas para poder observar en qué momento se alcanza un valor mínimo, ya que aquí será el punto óptimo del entrenamiento de la red.

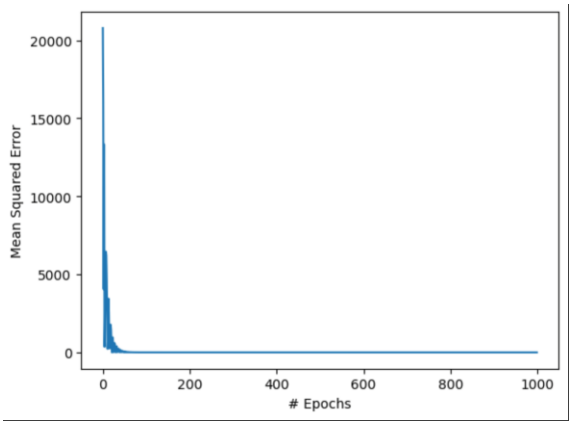


Figura 3. Mean Squared Error vs Epochs.

Resultados y conclusión

SuperPro Designer

Al ejecutar la simulación, se obtuvieron los resultados esperados, las curvas obtenidas son bastante similares a las mostradas en los artículos anteriormente mencionados.

Tabla 3. Datos experimentales obtenidos en SuperPro Designer.

Time (h)	Biomass (g/l)	Citric Acid (g/l)	Glucose (g/l)
0	0.1028	0	150.4354
6	0.1695	0.642	149.6797
12	0.2789	1.6953	148.4410
18	0.4574	3.4134	146.4180
24	0.746	6.1911	143.1527
30	1.2056	10.6149	137.9465
36	1.9197	17.4890	129.8469
42	2.9868	27.7600	117.7811
48	4.4814	42.1462	100.8494
54	6.3779	60.4006	79.3726
60	8.4594	80.4360	55.8193
66	10.3443	98.5798	34.4775
72	11.7232	111.8519	18.8604
78	12.5568	119.8758	9.4269
84	12.9955	124.0985	4.4602
90	13.2084	126.1480	2.0499
96	13.3074	127.1005	0.9290
102	13.3525	127.5352	0.4183
108	13.3729	127.7313	0.1878

114	13.3821	127.8194	0.0842
120	13.3862	127.8590	0.0377

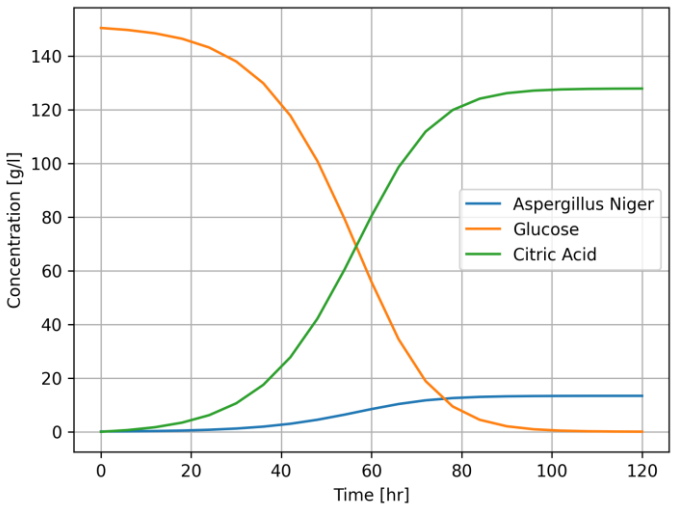


Figura 4. Modelo experimental para la producción de ácido cítrico. Fuente: Python

Las curvas generadas a partir de los datos experimentales muestran que la glucosa se consume rápidamente durante las primeras 60 horas, lo que impulsa la producción de ácido cítrico. La concentración de glucosa disminuye de 150 g/l a cero, mientras que la concentración de ácido cítrico aumenta de manera acelerada y se estabiliza en aproximadamente 120 g/l, después de 80 horas. El crecimiento del hongo es más lento, estabilizándose entre 15 y 20 g/l cuando se agota la glucosa, indicando que el sustrato limita tanto el crecimiento del microorganismo como la producción de ácido cítrico.

Modelo en Python

Afortunadamente, el modelo matemático fue capaz de modelar adecuadamente el comportamiento obtenido en el software, no alcanzando los valores exactos, pero si una gran cercanía a ellos y la misma tendencia de las curvas, justo como se mostrará a continuación.

Las curvas obtenidas por el modelo de ecuaciones matemáticas tienen una tendencia similar a las curvas de datos experimentales obtenidos por (Najafpour G. D., 2015) para la producción de ácido cítrico por fermentación con Saccharomycopsis lipolyticain en un periodo de 6 días. De esta manera, los datos experimentales mostrados en el presente trabajo pueden ser validados a partir del modelo matemático y de la comparación directa con el software de simulación.

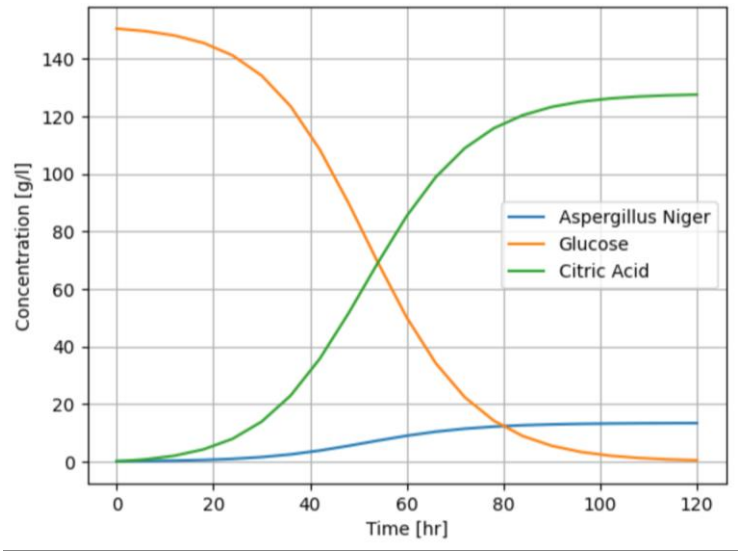


Figura 5. Modelo cinético para la producción de ácido cítrico.

Tabla 4. Datos del modelo matemático a 120 h.

Time (h)	Aspergillus Niger (g/l)	Citric Acid (g/l)	Glucose (g/l)
0	0.1028	0	150.4354
6	0.1799	0.7427	149.5616
12	0.3136	2.0291	148.0485
18	0.5421	4. 2288	145.4610
24	0.9246	7.9105	141.1302
30	1.5424	13.8575	134.1347
36	2.4856	22.9360	123.4556
42	3.8091	35.6753	108.4703
48	5.4651	51.6150	89.7203
54	7.2692	68.9806	69.2931
60	8.9689	85.3405	50.0489
66	10.3723	98.8487	34.1592
72	11.4157	108.8920	22.3452
78	12.1354	115.8194	14.1964
84	12.6076	120.3650	8.8493
90	12.9078	123.2545	5.4505
96	13.0949	125.0552	3.3323
102	13.2101	126.1640	2.0280
108	13.2805	126.8417	1.2308
114	13.3233	127.2541	0.7457
120	13.3493	127.5044	0.4513

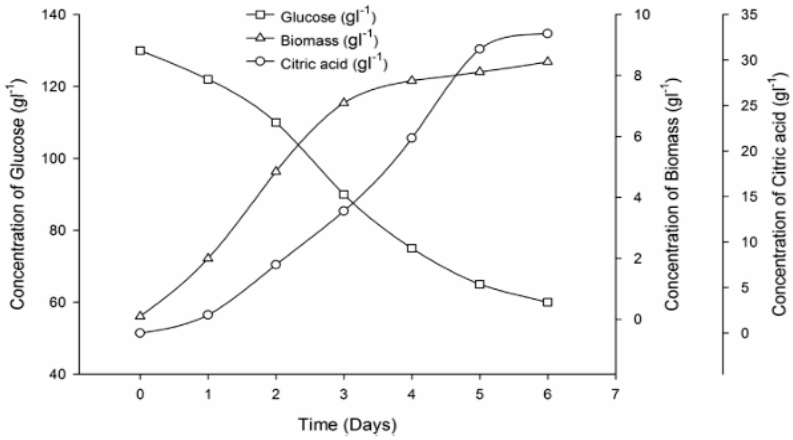


Figura 6. Substrate, biomass, and citric acid concentration profiles using Saccharomycopsis lipolyticain batch culture. (Najafpour G. D., 2015).

Modelo predictivo en Python en Orange y Python

Para la evaluación de los modelos predictivos de KNN y regresión lineal se hizo por dos maneras:, la primera es haciendo uso de los datos que particionamos anteriormente, los etiquetados como *X_test*, estos datos no fueron usados propiamente para entrenar al modelo, sino que sirven como un parámetro para medir la eficiencia del modelo usado, ya que de estos datos se tiene cual es la respuesta correcta, para evaluar el modelo por este método, solo es necesario generar una nueva variable que le llamamos *y_pred*, en donde le daremos los datos de una especie que pertenezca al conjunto de los *X_test*, le pedimos a Python que corra esa celda y arroje el valor de ácido cítrico. En la primera prueba que se realizó, se programó que se hiciera una tabla comparativa entre el valor verdadero y el valor predicho, obteniendo valores muy cercanos, se puede concluir que esta primera prueba tuvo un resultado satisfactorio.

La segunda manera de evaluar la eficiencia del modelo entrenado es por medio de su factor de correlación R^2, para esto solo es necesario insertar el siguiente comando: *print("Test set score: {:.2f}".format(knn.score(X_test, y_test)))*, en donde se le está pidiendo a Python que imprima el valor de correlación de los conjuntos de prueba, dándonos un valor de 0.998 para la regresión lineal y 0.94 para KNN y , es sabido que entre más cercano sea el valor a 1, nos habla de un buen desempeño del modelo.

Concluyendo de este código, que, por las dos evaluaciones realizadas, obtuvimos resultados bastante satisfactorios para poder predecir la cantidad de ácido cítrico producido basándonos en sus características de operación, como lo son la glucosa alimentada.

En Orange, hay un widget que ya da la evaluación de los modelos de manera automática:

Tabla 5. Métricas de desempeño para la primera red neuronal.

Model	MSE	RMSE	MAE	R2
Neural Network	0.006	0.075	0.023	1
Linear Regression	0.055	0.234	0.223	1
kNN	62.762	7.922	5.167	0.978

Para la estimación de la concentración final de ácido cítrico se puede observar (Tabla 6) que los modelos con mayor exactitud son la ANN y la regresión lineal, basándonos en su coeficiente de determinación que son bastante cercanos, sin embargo, se observa que la regresión lineal tiene menor grado de error en sus métricas, haciéndola la más precisa para la estimación de concentración final de ácido cítrico, puede ser debido a la naturaleza de los datos de entrenamiento que tienen una tendencia lineal, ya que la ANN y la regresión lineal son cercanas en su coeficiente de determinación se puede concluir que se logró un buen entrenamiento de un modelo complejo como lo es la red neuronal.

El modelo que presentó mayor deficiencia fue el KNN, pese a que sus métricas no son malas, no presenta gran exactitud en la estimación de los valores.

Tabla 6. Métricas de desempeño para la segunda red neuronal.

Model	MSE	RMSE	MAE	R2
Neural Network	16.789	4.097	3.079	0.990
Linear Regression	0.207	0.455	0.323	1
kNN	113.213	10.640	7.136	0.932

Tabla 7. Concentración final de Ácido Cítrico con diferentes modelos.

Experimental Citric Acid (g/l)	Neural Network	Linear Regression	kNN
82.726	92.720	83.953	82.937
82.937	86.341	83.883	82.726
96.872	95.496	97.204	106.603
106.603	102.629	106.576	113.090
113.090	114.516	113.193	115.908
115.908	113.709	115.907	113.090
127.859	130.198	127.655	130.609
130.609	131.010	130.377	127.859
143.723	142.958	143.440	130.609
160.221	166.580	159.921	168.622
168.622	167.325	168.333	170.269
170.269	163.774	169.979	168.622
180.761	180.395	180.480	170.269
192.594	191.743	192.351	180.761
224.723	229.668	224.635	192.594

El software Orange puede realizar predicciones de manera aleatoria con los datos de prueba dados, obteniendo una comparativa entre los diferentes modelos entrenados y el dato experimental registrado. (Tabla 7). La comparación de los tres modelos de predicción muestra que la regresión lineal es más precisa, ya que sus predicciones son casi idénticas a los valores experimentales de concentración de ácido cítrico en todo el rango de datos, la red neuronal también ofrece predicciones cercanas, comienza a desviarse ligeramente en concentraciones más altas.

Tabla 8. Comparativa de la red neuronal con el modelo matemático para la predicción de concentración final de Ácido Cítrico.

Initial conc. Biomass (g/l)	Initial conc. Glucose (g/l)	Conc. Citric acid Math. Model (g/l)	Conc. Citric acid ANN (g/l)
0.1234	90	74.9068	76.3965
0.1234	170	144.2522	144.1869
0.1234	230	195.1123	195.0297
0.3241	150	127.3849	127.0674
0.3241	70	58.3019	59.2771
0.3241	135	114.6017	114.3567
0.2567	190	161.3734	161.0203
0.1389	190	161.2836	161.1212
0.9854	190	161.4654	160.3964
0.4528	250	212.1540	211.6952
0.8643	250	212.2090	211.3428
0.6197	250	212.1878	211.5523

Se comparó la cercanía entre el modelo matemático y la red neuronal programada en Python a diferentes concentraciones iniciales tanto de biomasa como de glucosa (Tabla 6), dando un resultado bastante parecido entre ellos, de tal manera que no es posible diferenciar las líneas entre un modelo y otro. (Figura 7).

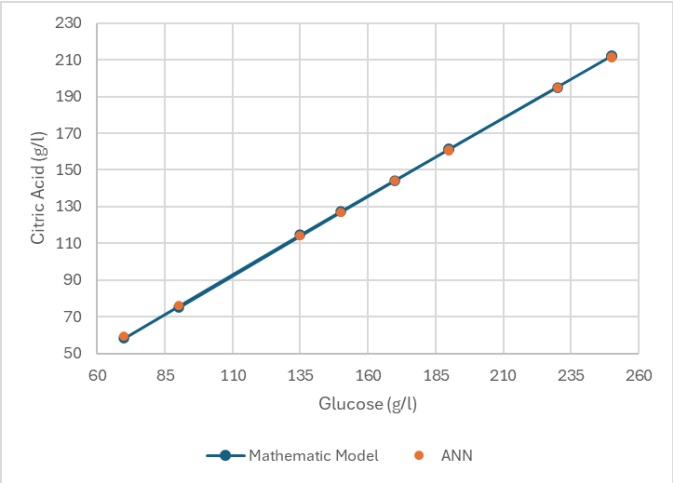


Figura 7. Modelo matemático vs ANN.

Los resultados expuestos, demuestran la viabilidad de utilizar las Redes Neuronales Artificiales bajo el enfoque predictivo de diversos procesos de producción por fermentación y el modelado de su comportamiento a lo largo de su proceso.

Conclusiones

En este proyecto de investigación se ha tenido la oportunidad de explorar un área de la ingeniería que tiene mucho potencial en la industria y aun no hay tantos datos disponibles en el proceso de fermentación, pese que está teniendo cada vez más relevancia para la producción de diversos compuestos.

Se comprueba la viabilidad de utilizar redes neuronales para la estimación de la concentración en un proceso de fermentación, dando pie a que se migre este uso a más procesos de fermentación que presenten dificultad para el acceso a la información de su cinética de reacción o demás parámetros.

Ha sido un reto el realizar esta investigación, se ha logrado adquirir nuevas herramientas y aprendizajes como lo es el uso de los diversos softwares y lenguajes de programación implementados en este proyecto.

Se espera que este proyecto sea útil para sentar las bases para seguir implementado los modelos de predicción en sistemas que se rigen por fermentación, así como brindar datos importantes para cualquier modelado base de fermentación en cualquier modelo o software, ya que la obtención de la información base es muy escasa y de difícil acceso, limitando así los posibles trabajos a desarrollar.

Se concluye satisfactoriamente este trabajo con el orgullo de que se han obtenido muy buenos resultados en todos los modelos trabajados.

Bibliografía

AWS. (2024). *Amazon*. Obtenido de https://aws.amazon.com/es/what-is/neural-network/#:~:text=Una%20red%20neuronal%20es%20un,lo%20hace%20el%20cerebro%20humano.

Bagnato, J. I. (2022). *Aprende Machine Learning.* México: Porrúa.

Carmichael, D., & Petrides, D. (2020). Production of Citric Acid via Fermentation - Process Modeling and Techno Economic Assessment (TEA) using SuperPro Designer. *Intelligen, Inc*.

Eddy, M. (1995). *Ingeniería de Aguas residuales. Tratamiento, vertido y reutilización. .* Madrid: McGraw-Hill.

Hernández Regalado, R., López de la Maza, L., Pérez Ones, O., & Zumalacárregui de Cárdenas, L. (2019). Modelación neuronal de la etapa de fermentación en una ronera cubana. *VII Simposio Internacional de Química 2019*.

López de la Maza, L. E., Zumalacárregui de Cárdenas, L., Pérez, O., & Llanes, O. (2018). Obtención de un modelo neuronal para la estimación de la concentración de etanol en la destilería Héctor Molina. *Tecnología Química*.

Mezcal, L. L. (2022). *La Luna Mezcal*. Obtenido de https://lalunamezcal.com/blog/fermentacion/

Moya Almeida, V., Diezma Iglesias, B., & Correa Hernando, E. C. (2023). Desarrollo de un sistema de predicción y control de solventes durante la fermentación de cerveza con S. Cerevisiae mediante redes neuronales y lógica difusa. *Universidad Poltécnica de Madrid*.

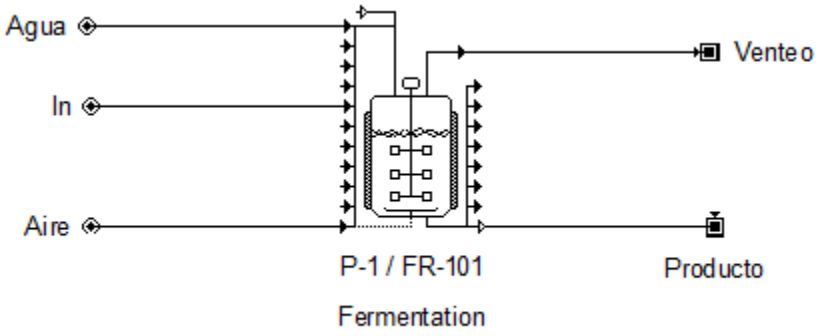
Najafpour, G. D. (2015). *Biochemical Engineering and Biotechnology.* Amsterdam: Elsevier.

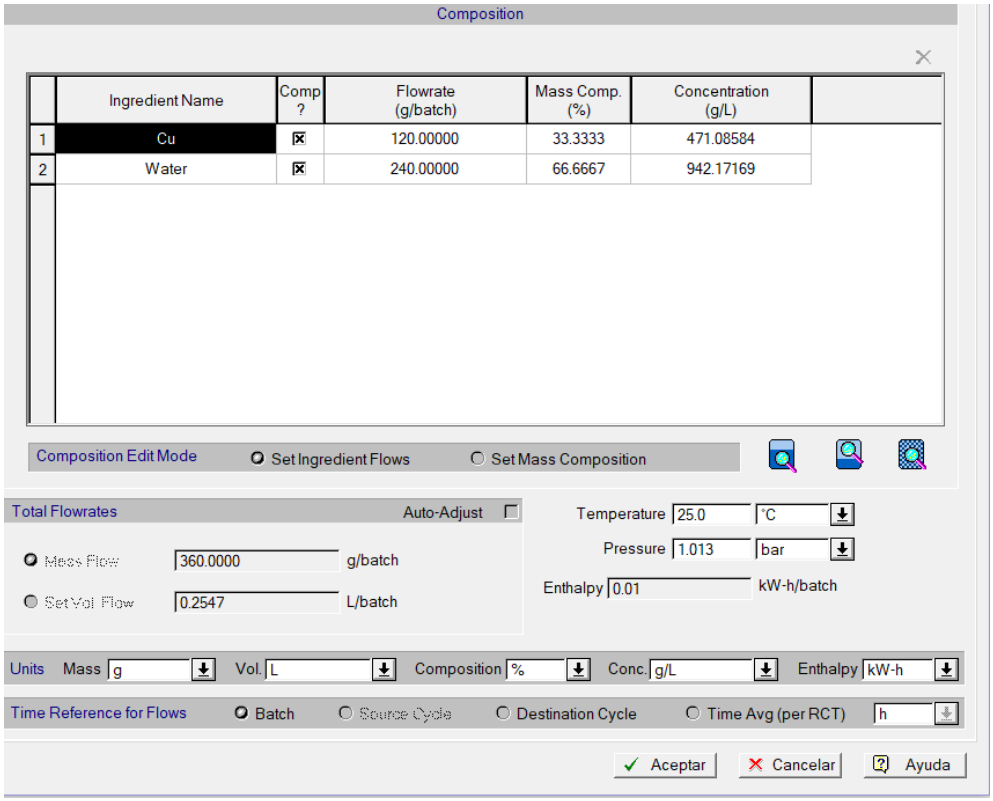
Najafpour, G. D. (2015). *Biochemical Engineering and Biotechnology.* Amsterdam: Elsevier.

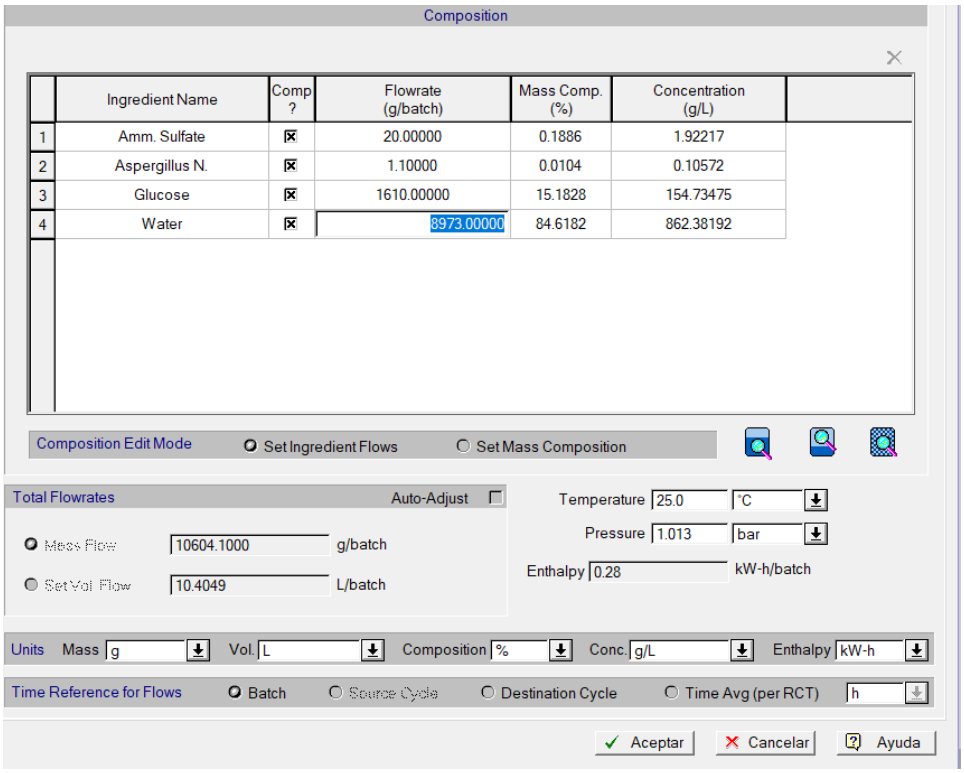
Urtubia, A., & Román Russell, C. A. (2010). Utilización de Redes Neuronales Artificiales para predecir problemas de fermentaciones vínicas. *Dircción de Bibliotecas y Recursos para el Aprendizaje*.

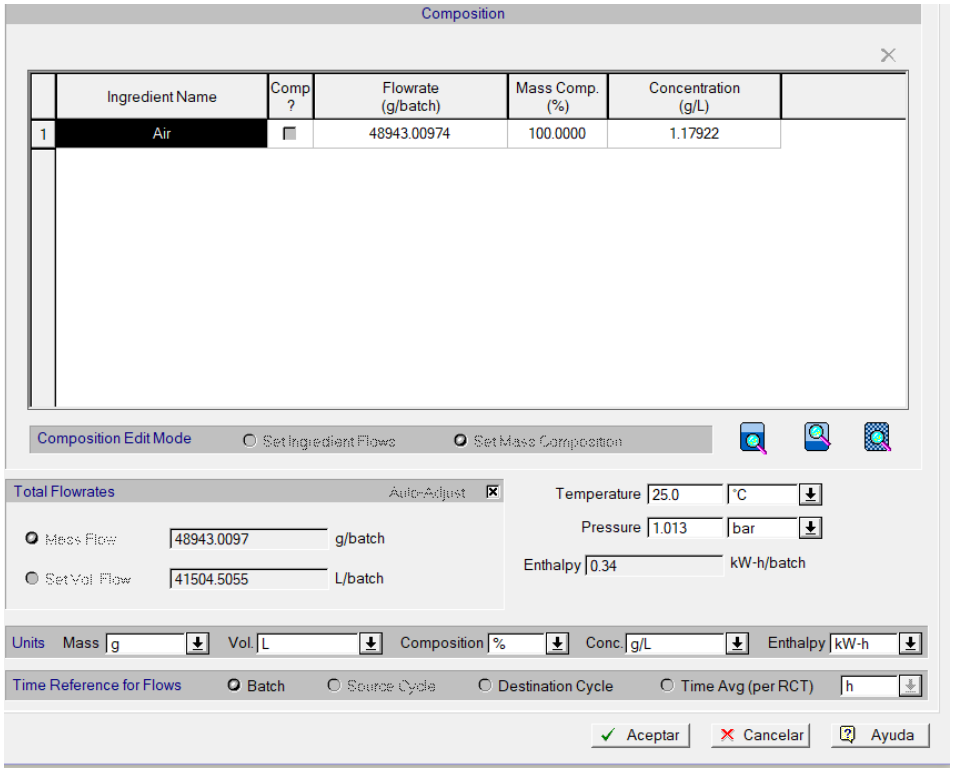
Anexo

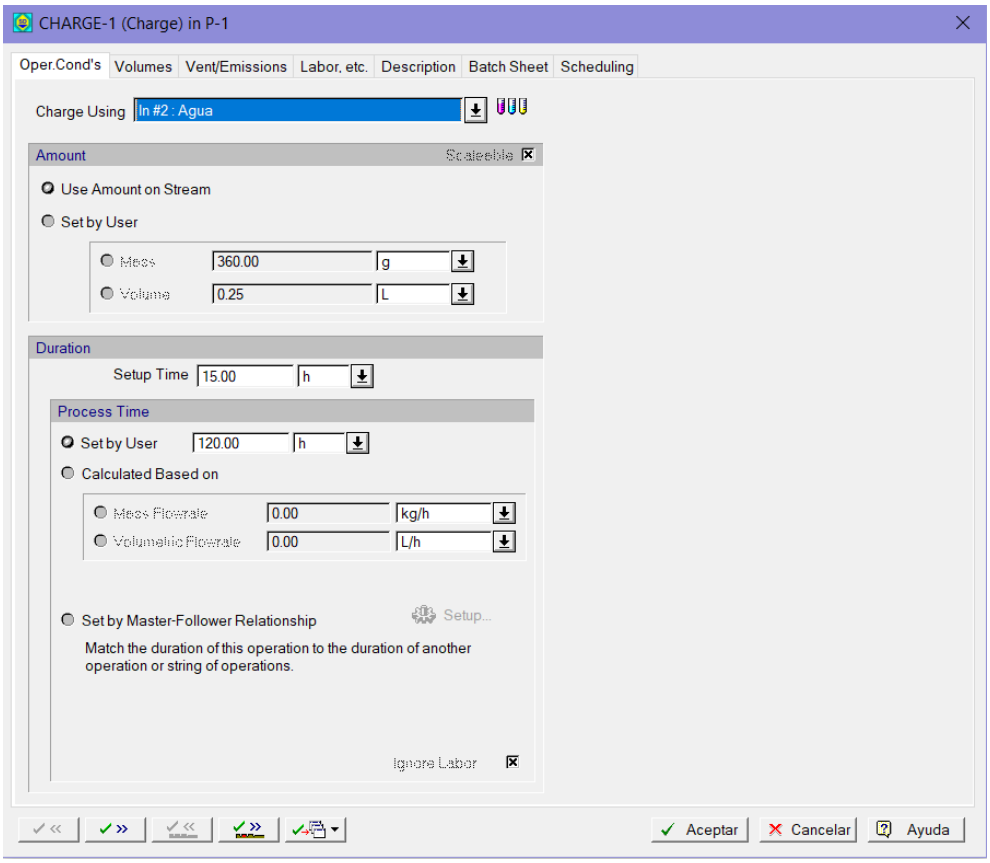
Simulación en SuperPro Designer

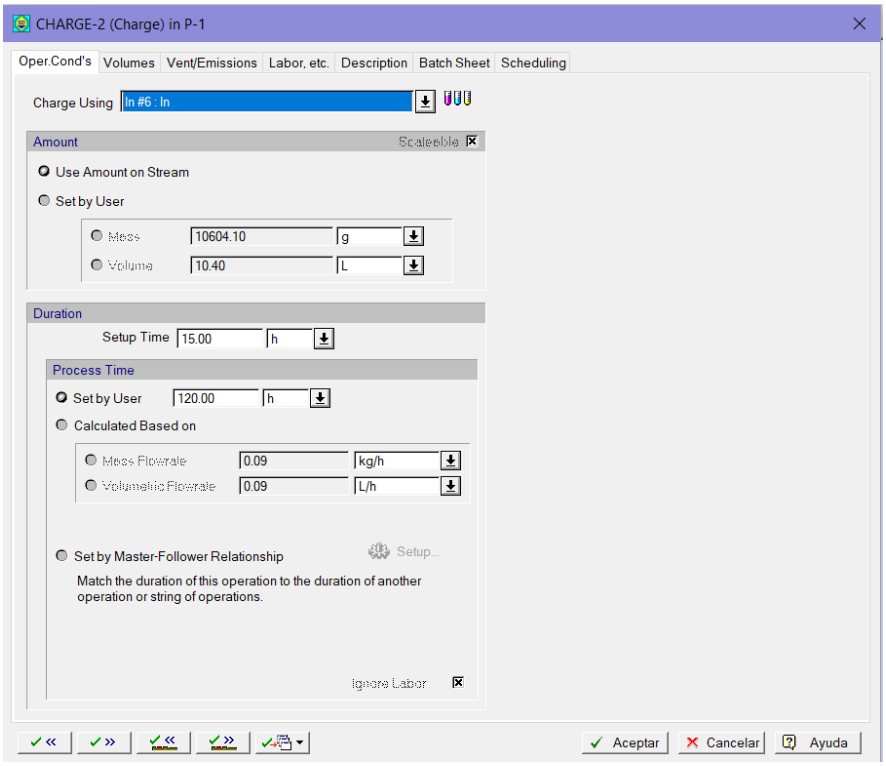


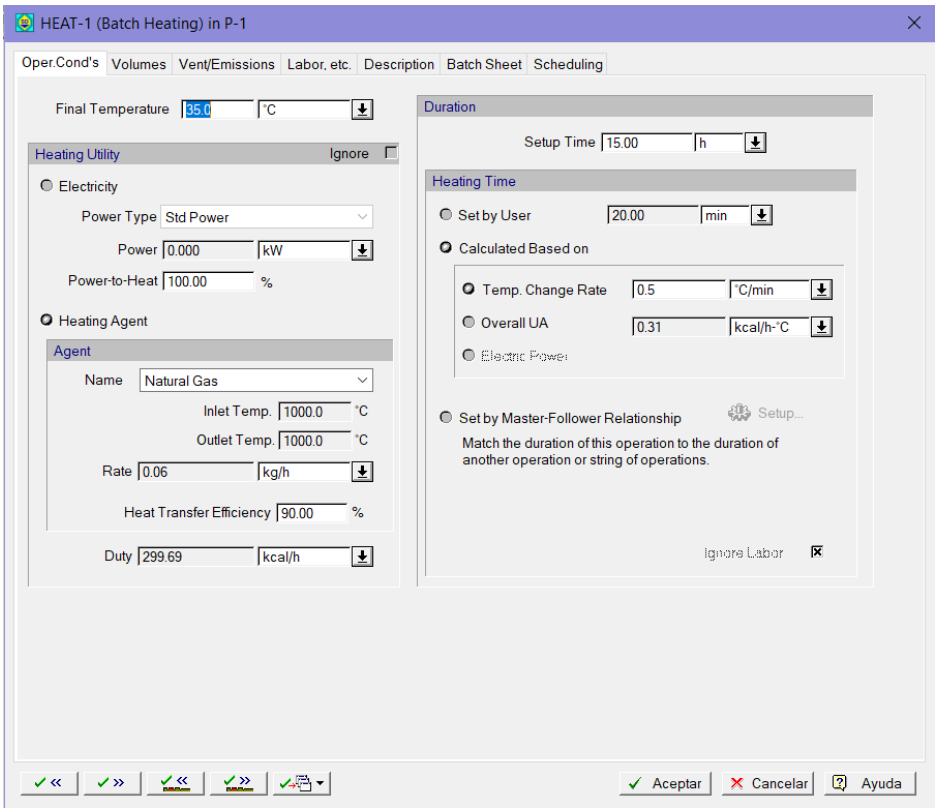


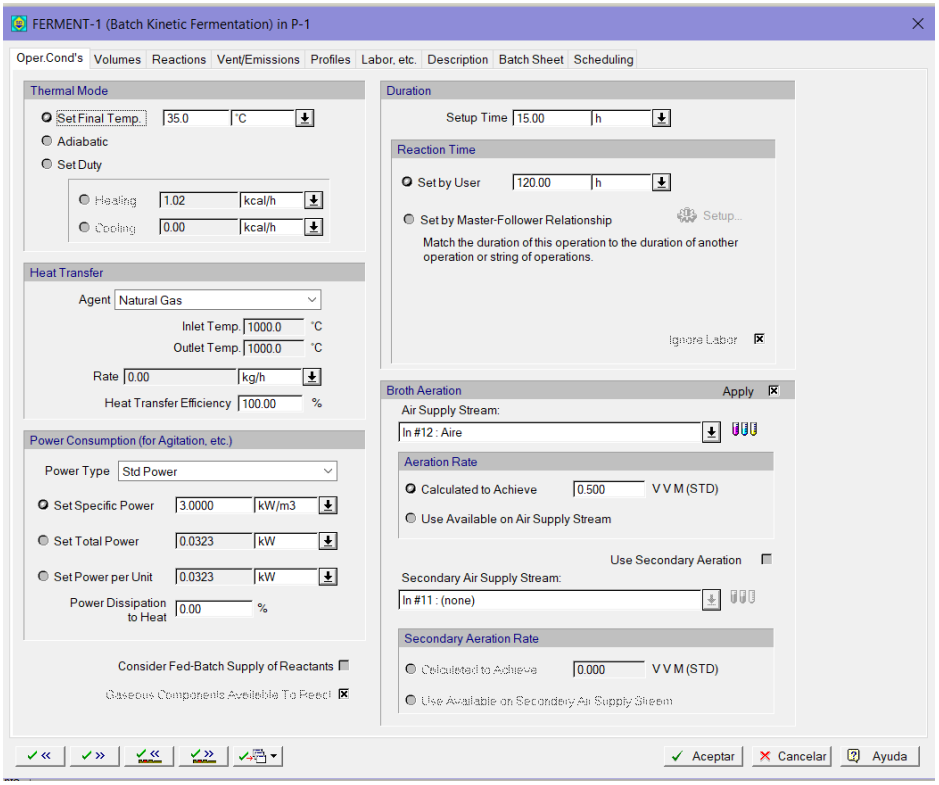

Composition
Ingredient Name
Comp ?
Flowrate (g/batch)
Mass Comp. (%)
Concentration (g/L)
1 Amm. Sulfate 20.00000 0.1886 1.92217
2 Aspergillus N. 1.10000 0.0104 0.10572
3 Glucose 1610.00000 15.1828 154.73475
4 Water 8973.00000 84.6182 862.38192
Composition Edit Mode
Set Ingredient Flows
Set Mass Composition
Total Flowrates
Auto-Adjust
Mass Flow 10604.1000 g/batch
Set Vol Flow 10.4049 L/batch
Temperature 25.0 °C
Pressure 1.013 bar
Enthalpy 0.28 kW-h/batch
Units Mass g Vol. L Composition % Conc. g/L Enthalpy kW-h
Time Reference for Flows
Batch
Source Cycle
Destination Cycle
Time Avg (per RCT) h
Aceptar
Cancelar
Ayuda

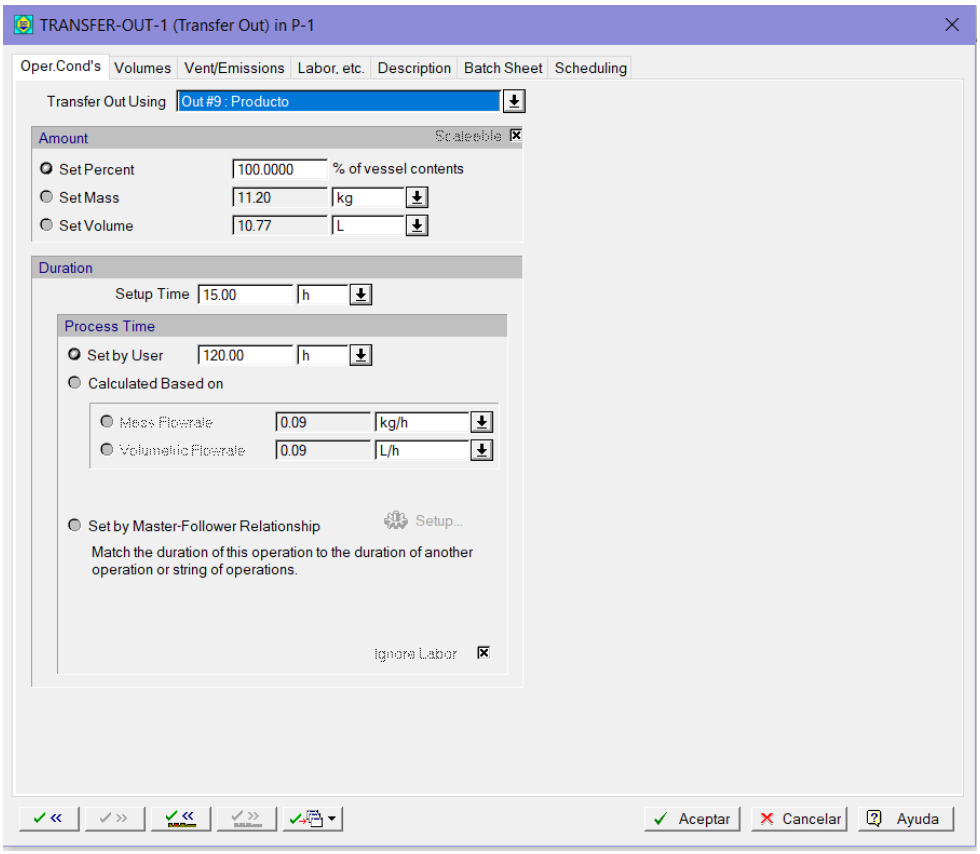

Composition
Ingredient Name
Comp ?
Flowrate (g/batch)
Mass Comp. (%)
Concentration (g/L)
1 Air 48943.00974 100.0000 1.17922
Composition Edit Mode
Set Ingredient Flows
Set Mass Composition
Total Flowrates
Auto-Adjust
Mass Flow 48943.0097 g/batch
Set Vol Flow 41504.5055 L/batch
Temperature 25.0 °C
Pressure 1.013 bar
Enthalpy 0.34 kW-h/batch
Units Mass g Vol. L Composition % Conc. g/L Enthalpy kW-h
Time Reference for Flows
Batch
Source Cycle
Destination Cycle
Time Avg (per RCT) h
Aceptar
Cancelar
Ayuda

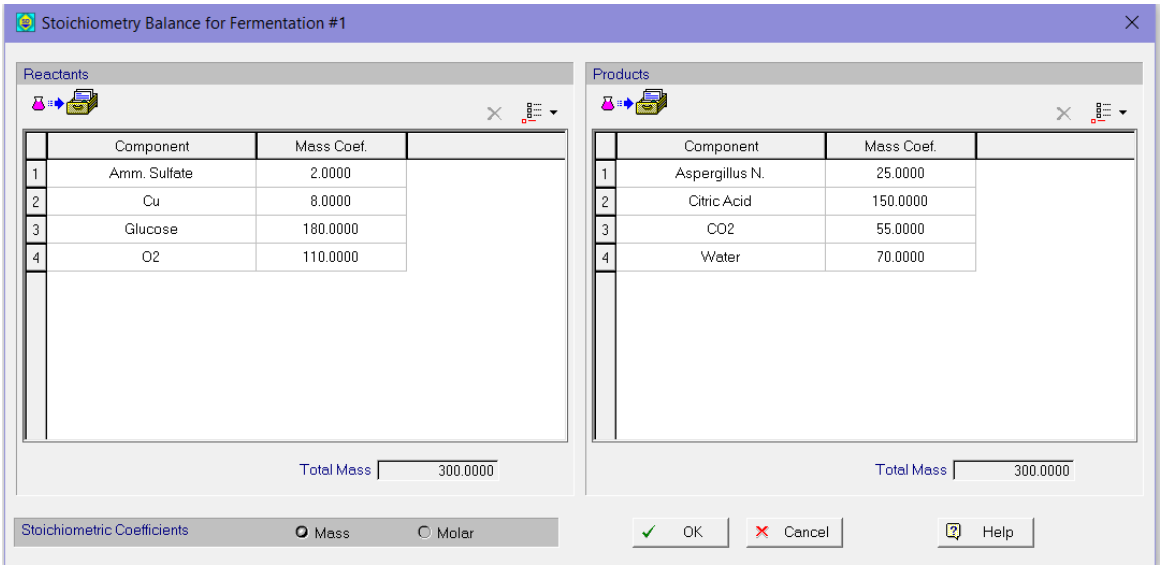

CHARGE-1 (Charge) in P-1
Oper.Cond's
Volumes
Vent/Emissions
Labor, etc.
Description
Batch Sheet
Scheduling
Charge Using
In #2 : Agua
Amount
Scaleable
Use Amount on Stream
Set by User
Mass
360.00
g
Volume
0.25
L
Duration
Setup Time
15.00
h
Process Time
Set by User
120.00
h
Calculated Based on
Mass Flowrate
0.00
kg/h
Volumetric Flowrate
0.00
L/h
Set by Master-Follower Relationship
Setup...
Match the duration of this operation to the duration of another operation or string of operations.
Ignore Labor
Aceptar
Cancelar
Ayuda

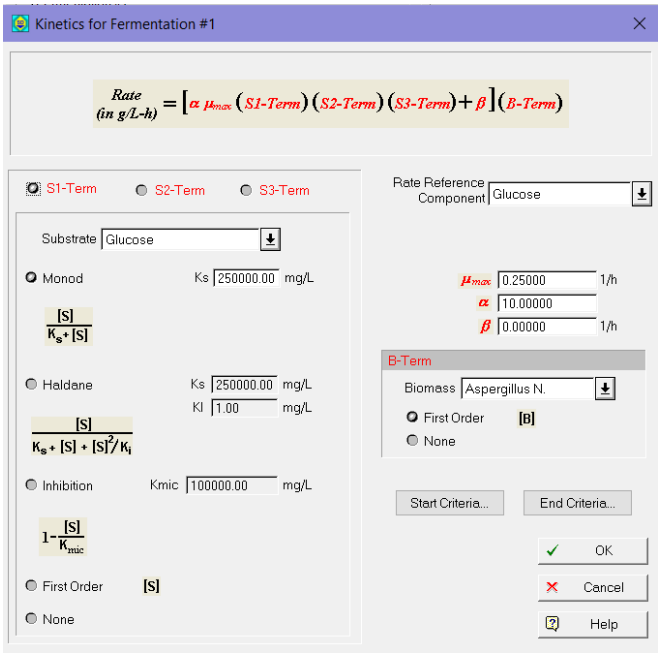

CHARGE-2 (Charge) in P-1
Oper.Cond's
Volumes
Vent/Emissions
Labor, etc.
Description
Batch Sheet
Scheduling
Charge Using
In #6 : In
Amount
Scaleable
Use Amount on Stream
Set by User
Mass
10604.10
g
Volume
10.40
L
Duration
Setup Time
15.00
h
Process Time
Set by User
120.00
h
Calculated Based on
Mass Flowrate
0.09
kg/h
Volumetric Flowrate
0.09
L/h
Set by Master-Follower Relationship
Setup...
Match the duration of this operation to the duration of another operation or string of operations.
Ignore Labor
Aceptar
Cancelar
Ayuda


HEAT-1 (Batch Heating) in P-1
Oper.Cond's
Volumes
Vent/Emissions
Labor, etc.
Description
Batch Sheet
Scheduling
Final Temperature
35.0
°C
Heating Utility
Ignore
Electricity
Power Type
Std Power
Power
0.000
kW
Power-to-Heat
100.00
%
Heating Agent
Agent
Name
Natural Gas
Inlet Temp.
1000.0
°C
Outlet Temp.
1000.0
°C
Rate
0.06
kg/h
Heat Transfer Efficiency
90.00
%
Duty
299.69
kcal/h
Duration
Setup Time
15.00
h
Heating Time
Set by User
20.00
min
Calculated Based on
Temp. Change Rate
0.5
°C/min
Overall UA
0.31
kcal/h-°C
Electric Power
Set by Master-Follower Relationship
Setup...
Match the duration of this operation to the duration of another operation or string of operations.
Ignore Labor
Aceptar
Cancelar
Ayuda


FERMENT-1 (Batch Kinetic Fermentation) in P-1
Oper.Cond's
Volumes
Reactions
Vent/Emissions
Profiles
Labor, etc.
Description
Batch Sheet
Scheduling
Thermal Mode
Set Final Temp.
35.0
°C
Adiabatic
Set Duty
Heating
1.02
kcal/h
Cooling
0.00
kcal/h
Heat Transfer
Agent
Natural Gas
Inlet Temp.
1000.0
°C
Outlet Temp.
1000.0
°C
Rate
0.00
kg/h
Heat Transfer Efficiency
100.00
%
Power Consumption (for Agitation, etc.)
Power Type
Std Power
Set Specific Power
3.0000
kW/m3
Set Total Power
0.0323
kW
Set Power per Unit
0.0323
kW
Power Dissipation to Heat
0.00
%
Consider Fed-Batch Supply of Reactants
Gaseous Components Available To React
Duration
Setup Time
15.00
h
Reaction Time
Set by User
120.00
h
Set by Master-Follower Relationship
Setup...
Match the duration of this operation to the duration of another operation or string of operations.
Ignore Labor
Broth Aeration
Apply
Air Supply Stream:
In #12 : Aire
Aeration Rate
Calculated to Achieve
0.500
V V M (STD)
Use Available on Air Supply Stream
Use Secondary Aeration
Secondary Air Supply Stream:
In #11 : (none)
Secondary Aeration Rate
Calculated to Achieve
0.000
V V M (STD)
Use Available on Secondary Air Supply Stream
Aceptar
Cancelar
Ayuda


TRANSFER-OUT-1 (Transfer Out) in P-1
Oper.Cond's
Volumes
Vent/Emissions
Labor, etc.
Description
Batch Sheet
Scheduling
Transfer Out Using
Out #9 : Producto
Amount
Scaleable
Set Percent
100.0000
% of vessel contents
Set Mass
11.20
kg
Set Volume
10.77
L
Duration
Setup Time
15.00
h
Process Time
Set by User
120.00
h
Calculated Based on
Mass Flowrate
0.09
kg/h
Volumetric Flowrate
0.09
L/h
Set by Master-Follower Relationship
Setup...
Match the duration of this operation to the duration of another operation or string of operations.
Ignore Labor
Aceptar
Cancelar
Ayuda


Stoichiometry Balance for Fermentation #1
Reactants
Component
Mass Coef.
1 Amm. Sulfate 2.0000
2 Cu 8.0000
3 Glucose 180.0000
4 O2 110.0000
Total Mass
300.0000
Products
Component
Mass Coef.
1 Aspergillus N. 25.0000
2 Citric Acid 150.0000
3 CO2 55.0000
4 Water 70.0000
Total Mass
300.0000
Stoichiometric Coefficients
Mass
Molar
OK
Cancel
Help



Código predictive en Python (Red neuronal)

```
import tensorflow as tf

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

import keras

from keras import layers

Data=pd.read_csv("/content/Primer Entrenamiento 1.csv")

Data

Biomasa = Data['Biomass (g/L)'].astype(float)

Sustrato = Data['Glucose (g/L)'].astype(float)

Producto = Data['Citric Acid (g/L)'].astype(float)

modelo = tf.keras.Sequential(

    [

        layers.Dense(10, activation="relu", input_shape=[2],name="oculta1"),

        layers.Dense(10, activation="relu",name="oculta2"),

        layers.Dense(10, activation="relu",name="oculta3"),
```

```
        layers.Dense(10, activation="relu",name="oculta4"),
        layers.Dense(10, activation="relu",name="oculta5"),
        layers.Dense(1,name="salida")
    ]
)
modelo.compile(
    optimizer=tf.keras.optimizers.Adam(0.1),
    loss='mean_squared_error'
)
print("Comenzando entrenamiento...")
datos_in = np.column_stack((Biomasa,Sustrato))
historial = modelo.fit(datos_in, Producto, epochs=1000, verbose=False)
print("Modelo entrenado!")
plt.xlabel("# Epochs")
plt.ylabel("Mean Squared Error")
plt.plot(historial.history["loss"])
plt.grid()
plt.savefig('grafico_error.png', dpi=300, bbox_inches='tight')
from sklearn.metrics import r2_score
# Obtener las predicciones del modelo para los datos de entrenamiento
y_pred = modelo.predict(datos_in)
# Calcular el R2
r2 = r2_score(Producto, y_pred)
print("El R2 del modelo es:", r2)
resultado = modelo.predict(np.array([[0.6197, 250]]))
print("La concentración final de ácido cítrico es " + str(resultado))
```

Código de modelo matemático

```
import numpy as np
import matplotlib, matplotlib.pyplot as plt
import pandas as pd
from scipy.integrate import odeint
from google.colab import drive
drive.mount('/content/drive')
images_dir = '/content/drive/MyDrive/biorreactor/'
#Parametros experimentales
mumax = 0.25 #1/hour
xmax = 30    #g/liter
Ks = 250     #g/liter
Yxs = 0.08832  #g/g
Yps = 0.85012  #g/g
def mu(X,S):
  return mumax * (1 - X/xmax) * (S/(Ks+S))
def z1(X,S):
  return X * mu(X,S)
def z2(X,S):
  return -1 * (1/Yxs) * z1(X,S)
def z3(X,S):
  return -1 * (Yps) * z2(X,S)
#Ecuaciones Diferenciales
def xdot(x,t):
```

```
  X,S,P=x
  dX=z1(X,S)
  dS=z2(X,S)
  dP=z3(X,S)
  return [dX,dS,dP]
 # Condiciones Iniciales
CI = [0.1028, 150, 0]
t = np.linspace(0,120,21)
sol = odeint(xdot,CI,t)
X,S,P = sol.transpose()
plt.plot(t,P,'--',color='#008000')  # verde
plt.plot(t,S, '-.', color='#F97306')  # naranja
plt.plot(t,X,'--',color='#00008B')  # azul
plt.xlabel('Time [hr]')
plt.ylabel('Concentration [g/l]')
plt.legend(['Citric Acid','Glucose','Aspergillus Niger'])
#plt.title('Kinetic Model Citric Acid Production')
plt.grid()
#fig_name = 'grafico_python.pdf'
#plt.savefig(f"{images_dir}{fig_name}", bbox_inches='tight')
#plt.savefig('grafico_Python.png', dpi=300, bbox_inches='tight')
tabla = pd.DataFrame({'Time [hr]': t, 'Conc Biomass': X, 'Conc Citric Acid': P, 'Conc Glucose': S})
#print(tabla)
Data=pd.read_csv("/content/drive/MyDrive/Glucosa 150.csv")
#Data
```

```
plt.plot(Data['Time (h)'],Data['Citric Acid (g/L)'],'-',color='#008000') #verde
plt.plot(t,P,'--',color='#008080')  # verde agua
plt.plot(Data['Time (h)'],Data['Glucose (g/L)'],':', color='#F97306') #naranja
plt.plot(t,S, '-.', color='#A0522D')  # cafe
plt.plot(Data['Time                                                    (h)'],Data['Aspergillus
(g/L)'],'.',color='#4B0082')#'#FF00FF')#'#00FFFF')#'#00008B') # azul
plt.plot(t,X,'x',color='#800080')#'#069AF3')#'#4B0082')  # morado
plt.xlabel('Time [hr]')
plt.ylabel('Concentration [g/l]')
plt.legend(['Citric Acid SPD','Citric Acid ODE',
        'Glucose SPD','Glucose ODE',
         'Aspergillus Niger SPD','Aspergillus Niger ODE'])
#plt.title('Experimental Model Citric Acid Production SPD')
plt.grid()

#fig_name = 'grafico_SPD.pdf'
#plt.savefig(f"{images_dir}{fig_name}", bbox_inches='tight')
plt.savefig('grafico_SPD.png', dpi=300, bbox_inches='tight')
plt.plot(t,P)
plt.plot(Data['Time (h)'],Data['Citric Acid (g/L)'],'-.')
plt.xlabel('Time [hr]')
plt.ylabel('Concentration [g/l]')
plt.legend(['Python',
        'SuperPro Designer'])
#plt.title('Citric Acid Concentration Python vs SPD')
plt.grid()
```

```
plt.savefig('grafico_Python_SPD.png', dpi=300, bbox_inches='tight')
error = np.array ((Data['Citric Acid (g/L)'])-P)#/P)
tabla_comparativa = pd.DataFrame({
    'Tiempo [hr]': t,
    'Concentración Producto [g/litro]': P,
    'Citric Acid': Data['Citric Acid (g/L)'],
    'Error': error
})
print(tabla_comparativa)
plt.plot(t,error)
plt.xlabel('Tiempo [hr]')
plt.ylabel('Error [g/L]')
plt.legend(['In Silico ODE error'])
plt.grid()
plt.plot(Data['Time'],Data['SPD'],'o',color='#FF00FF')
plt.plot(Data['Time'],Data['kNN'],'^',color='#FFA500')##F97306)
plt.plot(Data['Time'],Data['LR'],':',color='c')#'#000080')
plt.plot(Data['Time'],Data['ANN'],'x',color='k')#'#808080')
plt.xlabel('Time [hr]')
plt.ylabel('Concentration [g/l]')
plt.legend(['In Silico Data','kNN', 'Linear Regression', 'Neural Network'])
#fig_name = 'annoutcome2.pdf'
plt.grid()
#plt.savefig(f"{images_dir}{fig_name}", bbox_inches='tight') # Recorta justamente
#plt.savefig('grafico_comparativo_curvas.png', dpi=300, bbox_inches='tight')
```

Código modelos predictivos

```
#importamos librerias

import pandas as pd

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import r2_score

#importamos los datos

Datos=pd.read_csv("/content/Ácido citrico.csv")

Datos

#particionamos el conjunto de datos

Predictores=Datos.drop(["Producto (g/l)"], axis=1)#(["Biomasa(g/l)", "Sustrato(g/l)"],axis=1)

Target=Datos["Producto (g/l)"]

X_train, X_test, y_train, y_test =train_test_split(Predictores, Target, train_size=0.7, test_size=0.3,random_state=0)

Regresión lineal

#entrenamos el modelo

#Aqui se hacen las modificaciones de los modelos a probar y ver cual es el mejor

modelo_rl=LinearRegression() #no tiene hiperparametros

modelo_rl.fit(X_train, y_train)

r2_train=modelo_rl.score(X_train, y_train)

r2_test=modelo_rl.score(X_test, y_test)

print("El valor de R2 para el conjunto de entrenamiento es: {}".format(r2_train))

print("El valor de R2 para el conjunto de prueba es: {}".format(r2_test))

#diagrama de paridad valores reales vs los arrojados por el modelo

#Realizamos pronósticos
```

```
y_predict=modelo_rl. predict(X_test)
Tabla=pd. DataFrame()
Tabla["Valor de ácido cítrico experimental"]=y_test
Tabla["ácido de predicción de predicción"]=y_predict
Tabla
```

Regresión Ridge

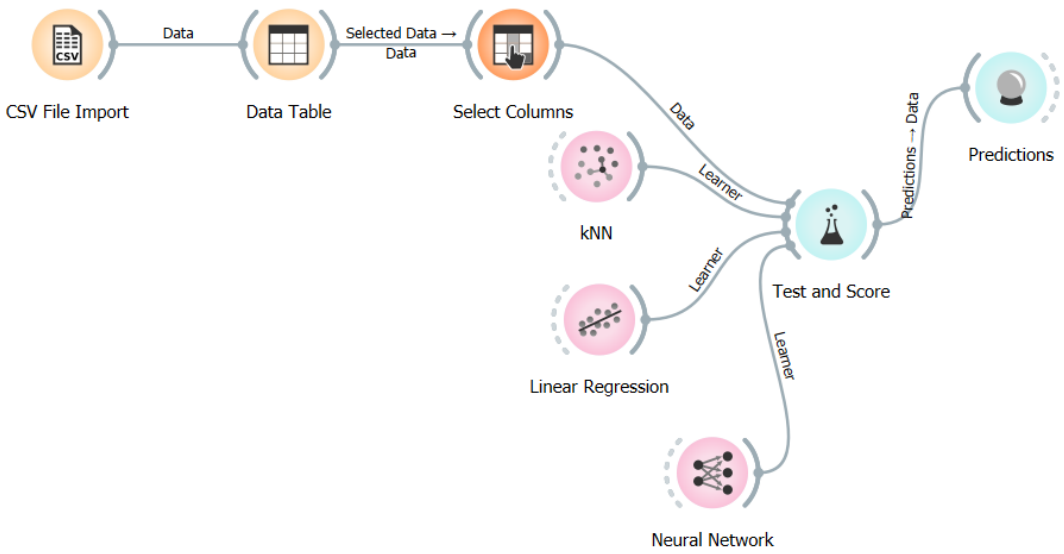
```
from sklearn. linear_model import Ridge, Lasso
#Entrenamos y evaluamos el modelo
modelo_ridge=Ridge(alpha=0. 5)
modelo_ridge. fit(X_train, y_train)
r2_train=modelo_ridge. score(X_train, y_train)
r2_test=modelo_ridge. score(X_test, y_test)
print("El valor de R2 para el conjunto de entrenamiento es: {}". format(r2_train))
print("El valor de R2 para el conjunto de prueba es: {}". format(r2_test))
#Entrenamos y evaluamos el modelo
modelo_lasso=Lasso(alpha=0. 5)
modelo_lasso. fit(X_train, y_train)
r2_train=modelo_lasso. score(X_train, y_train)
r2_test=modelo_lasso. score(X_test, y_test)
print("El valor de R2 para el conjunto de entrenamiento es: {}". format(r2_train))
print("El valor de R2 para el conjunto de prueba es: {}". format(r2_test))
```

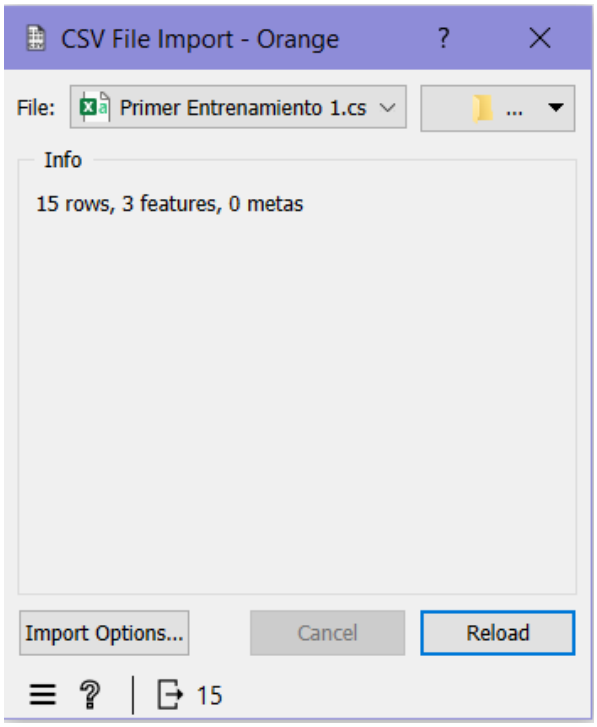
K-vecinos más cercanos

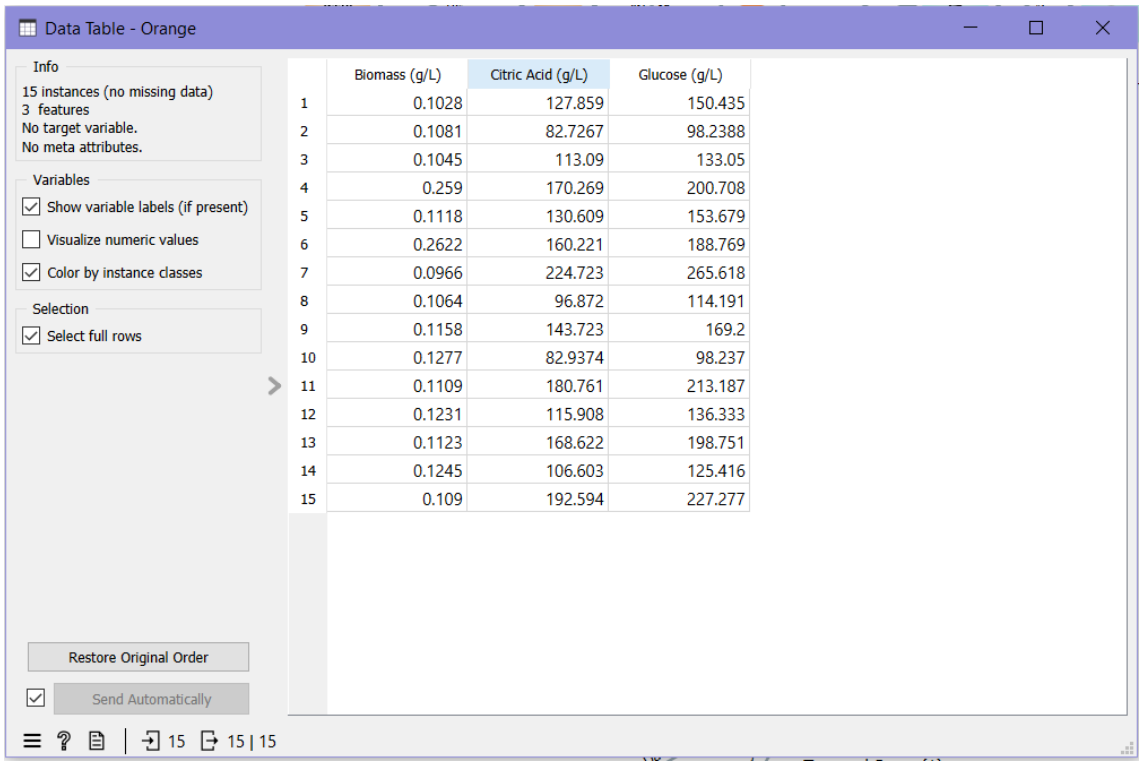
```
from sklearn. neighbors import KNeighborsRegressor
knn = KNeighborsRegressor(n_neighbors=2)
knn. fit(X_train, y_train)
```

```
import numpy as np
X_new = np. array([[20, 92,50]])
print("X_new. shape: {}". format(X_new. shape))
prediction = knn. predict(X_new)
print("Predicción: {}". format(prediction))
print("Nombre objetivo predicho: {}". format)
#Datos["Producto (g/l)"][prediction]))
closest_index = (Datos['Producto (g/l)'] - prediction). abs(). idxmin()
print("Nombre objetivo predicho: {}". format(Datos["Producto (g/l)"][closest_index]))
y_pred = knn. predict(X_test)
print("Predicciones del conjunto de prueba:
{}". format(y_pred))
print("Puntuación del conjunto de prueba: {:. 2f}". format(knn. score(X_test, y_test)))
#diagrama de paridad valores reales vs los generados por el modelo
#Realizamos pronósticos
y_pred = knn. predict(X_test)
Tabla=pd. DataFrame()
Tabla["Valor de ácido cítrico experimental"]=y_test
Tabla["ácido de predicción de predicción"]=y_pred
Tabla
```

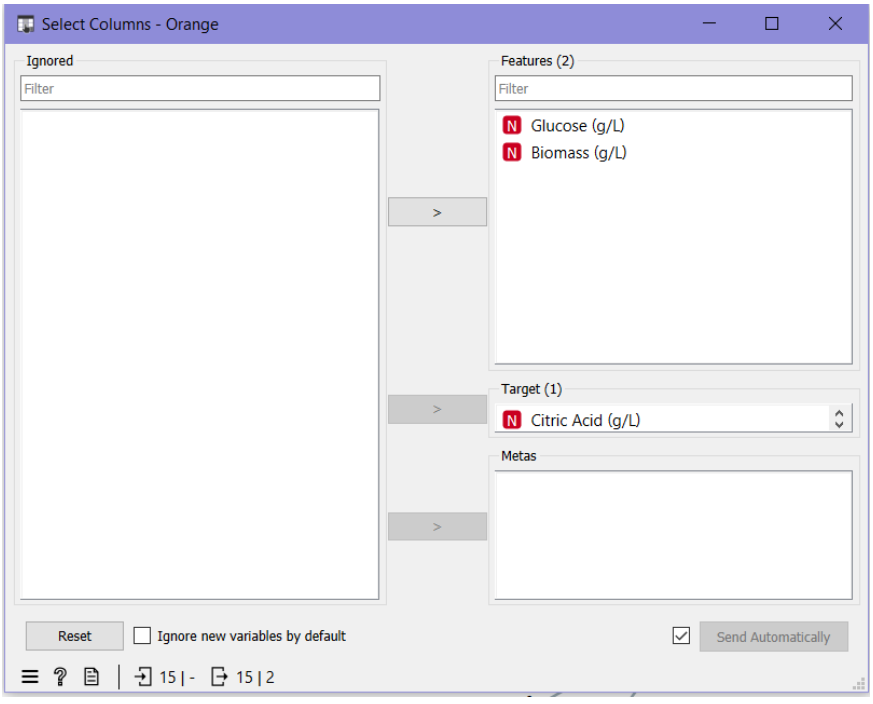
Orange

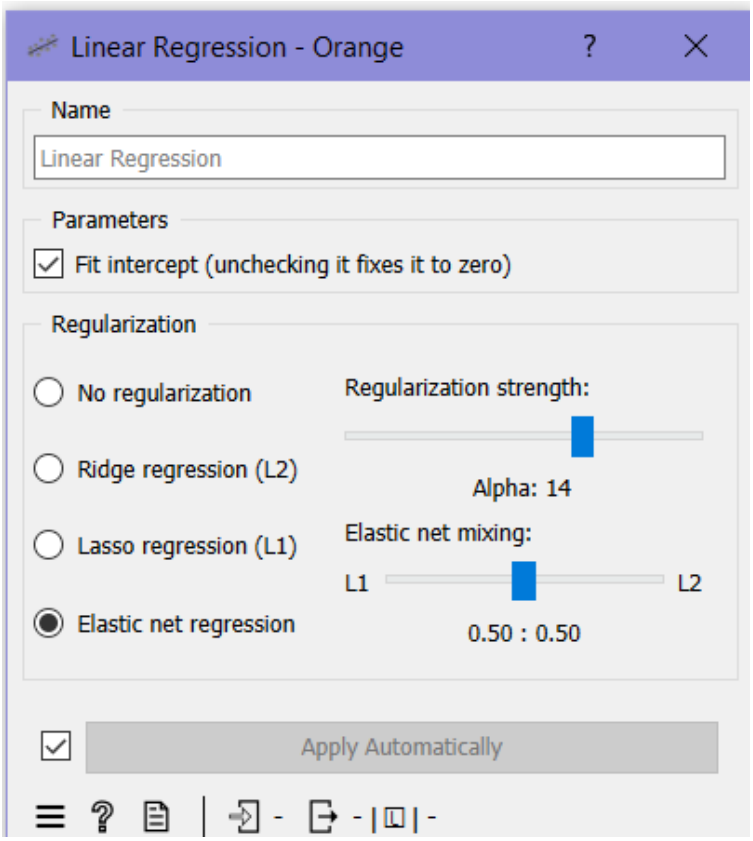

Data
Selected Data → Data
CSV File Import
Data Table
Select Columns
Data
Learner
kNN
Learner
Linear Regression
Learner
Neural Network
Test and Score
Predictions → Data
Predictions

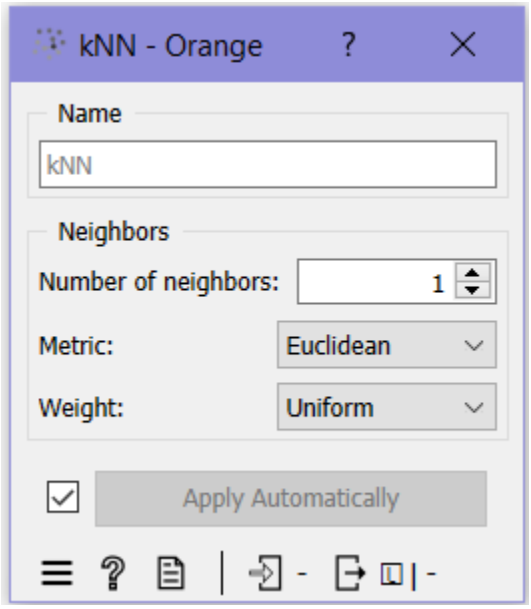

CSV File Import - Orange
File:
Primer Entrenamiento 1.cs
...
Info
15 rows, 3 features, 0 metas
Import Options...
Cancel
Reload
15

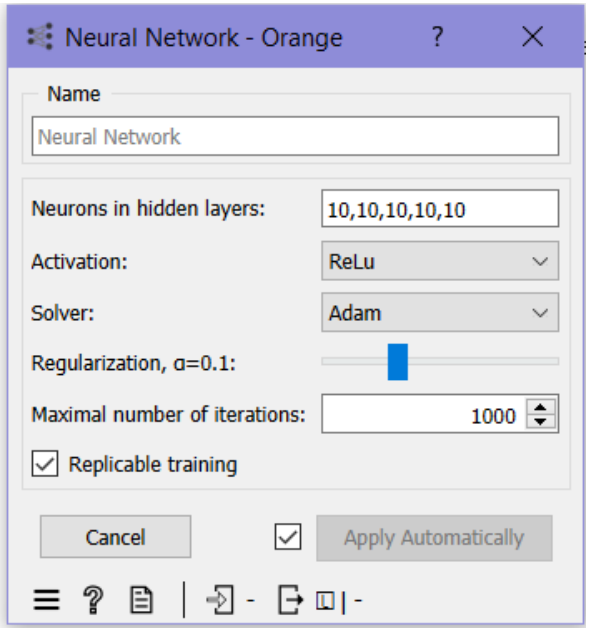


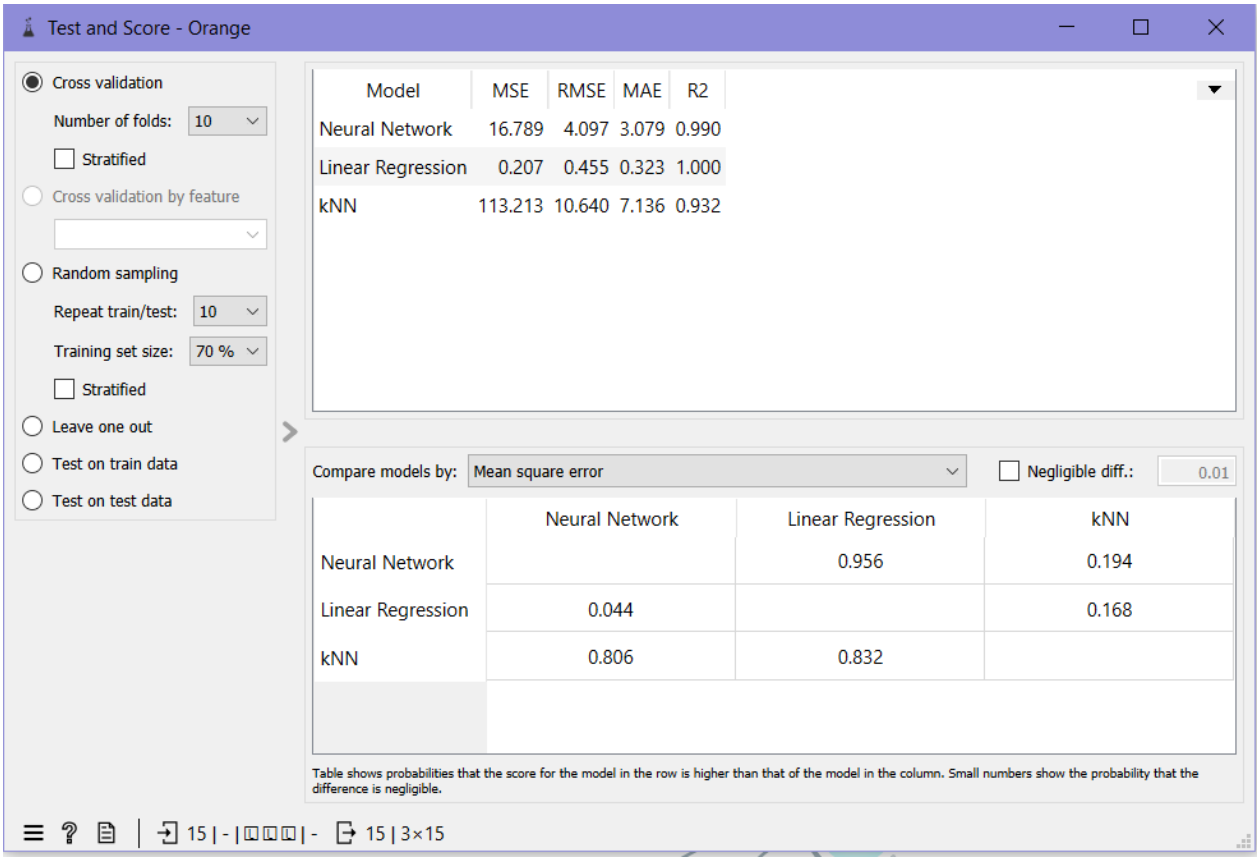
	Biomass (g/L)	Citric Acid (g/L)	Glucose (g/L)
1	0.1028	127.859	150.435
2	0.1081	82.7267	98.2388
3	0.1045	113.09	133.05
4	0.259	170.269	200.708
5	0.1118	130.609	153.679
6	0.2622	160.221	188.769
7	0.0966	224.723	265.618
8	0.1064	96.872	114.191
9	0.1158	143.723	169.2
10	0.1277	82.9374	98.237
11	0.1109	180.761	213.187
12	0.1231	115.908	136.333
13	0.1123	168.622	198.751
14	0.1245	106.603	125.416
15	0.109	192.594	227.277




Linear Regression - Orange
Name
Linear Regression
Parameters
Fit intercept (unchecking it fixes it to zero)
Regularization
No regularization
Ridge regression (L2)
Lasso regression (L1)
Elastic net regression
Regularization strength:
Alpha: 14
Elastic net mixing:
L1
L2
0.50 : 0.50
Apply Automatically


kNN - Orange
Name
kNN
Neighbors
Number of neighbors:
1
Metric:
Euclidean
Weight:
Uniform
Apply Automatically


Neural Network - Orange
Name
Neural Network
Neurons in hidden layers:
10,10,10,10,10
Activation:
ReLu
Solver:
Adam
Regularization, α=0.1:
Maximal number of iterations:
1000
Replicable training
Cancel
Apply Automatically


Test and Score - Orange
Cross validation
Number of folds: 10
Stratified
Cross validation by feature
Random sampling
Repeat train/test: 10
Training set size: 70 %
Stratified
Leave one out
Test on train data
Test on test data
Model MSE RMSE MAE R2
Neural Network 16.789 4.097 3.079 0.990
Linear Regression 0.207 0.455 0.323 1.000
kNN 113.213 10.640 7.136 0.932
Compare models by: Mean square error
Negligible diff.: 0.01
Neural Network Linear Regression kNN
Neural Network 0.956 0.194
Linear Regression 0.044 0.168
kNN 0.806 0.832
Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.
15 | - | 15 | 3×15

Citric Acid (g/L)	Neural Network	Linear Regression	kNN	Fold	Glucose (g/L)	Biomass (g/L)
82.9374	86.3417	83.8838	82.7267	1	98.237	0.1277
115.908	113.709	115.907	113.09	1	136.333	0.1231
127.859	130.198	127.655	130.609	2	150.435	0.1028
106.603	102.629	106.576	113.09	2	125.416	0.1245
160.221	166.58	159.921	168.622	3	188.769	0.2622
143.723	142.958	143.44	130.609	3	169.2	0.1158
82.7267	92.7202	83.9532	82.9374	4	98.2388	0.1081
113.09	114.516	113.193	115.908	4	133.05	0.1045
130.609	131.01	130.377	127.859	5	153.679	0.1118
192.594	191.743	192.351	180.761	5	227.277	0.109
96.872	95.4965	97.2048	106.603	6	114.191	0.1064
180.761	180.395	180.48	170.269	7	213.187	0.1109
168.622	167.325	168.333	170.269	8	198.751	0.1123
170.269	163.774	169.979	168.622	9	200.708	0.259
224.723	229.668	224.635	192.594	10	265.618	0.0966