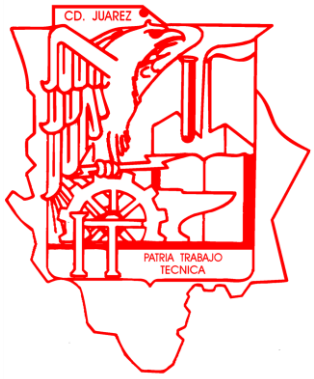
INSTITUTO TECNOLÓGICO DE CIUDAD JUÁREZ

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN



"DETERMINACIÓN DE TOXICIDAD DE LÍQUIDOS IÓNICOS UTILIZANDO DESCRIPTORES MOLECULARES SOAP, EN CONJUNTO CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL"

TESIS

QUE PRESENTA

KAREN LIZETH CARDENAS ALMANZA

COMO REQUISITO PARCIAL

PARA OBTENER EL GRADO DE

MAESTRA EN INGENIERÍA INDUSTRIAL

CD. JUÁREZ, CHIH. **JUNIO 2024**

AGRADECIMIENTOS:

Quiero agradecer grandemente a todas las personas que me apoyaron en este proceso, a mi familia, a mis amigos y a mis maestros. Quiero agradecer también al CONACYT por la beca otorgada para realizar estos estudios de Maestría.

Agradecemos también al Maestro Arnulfo Castro por la ayuda proporcionada para la elaboración del código computacional en lenguaje Python y sus respectivas librerías.

JORGE ADOLFO PINTO SANTOS
JEFE DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN
P R E S E N T E.

Por medio de la presente se hace constar que la tesis denominada "**DETERMINACIÓN DE TOXICIDAD DE LÍQUIDOS IONICOS UTILIZANDO DESCRIPTORES MOLECULARES SOAP EN CONJUNTO CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL**", presentado por el(la) alumno(a) **C. KAREN LIZETH CARDENAS ALMANZA** con número de control **M21110207**, para obtener el grado de Maestro(a) en el programa de Maestría en Ingeniería Industrial, ha sido revisada y aprobada en su forma y contenido por los suscritos, por lo que no existe ningún inconveniente para la impresión de la misma.

Se extiende la presente constancia a petición de él(la) interesado(a) y para los fines legales que a él(ella) convengan, en Ciudad Juárez, Chihuahua, a los veintiocho días del mes de mayo del año dos mil veinticuatro.

A T E N T A M E N T E
"Excelencia en Educación Tecnológica"

C. MARCO TULIO GALLO ESTRADA
DIRECTOR (A)

C. JORGE DE LA RIVA RODRÍGUEZ
CO-DIRECTOR (A)

C. ANA ISELA GARCÍA ACOSTA
REVISOR (A)

C. HUMBERTO GARCIA CASTELLANOS
REVISOR (A)

EDUCACIÓN | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CIUDAD JUÁREZ
06 JUN 2024
COORDINACIÓN DE TITULACIÓN
RECIBIDO

Instituto Tecnológico de Ciudad Juárez
06 JUN 202
Departamento de Servicios Escolares

C.c.p. División de Estudios de Posgrado e Investigación
Alumno(a)

Ciudad Juárez, Chihuahua, 04/junio/2024

C. MARÍA YOLANDA FRAUSTO VILLEGAS
JEFA DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES
P R E S E N T E.

Me permito hacer de su conocimiento que se le autoriza a él(la) **C. KAREN LIZETH CARDENAS ALMANZA** con número de control **M21110207**, la defensa de su tesis para obtener el grado de Maestro(a) en el programa de Maestría en Ingeniería Industrial quien desea titularse por medio de tesis.

El tema a desarrollar será denominado "**DETERMINACIÓN DE TOXICIDAD DE LÍQUIDOS IONICOS UTILIZANDO DESCRIPTORES MOLECULARES SOAP EN CONJUNTO CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL**", quedando conformado su jurado de la siguiente manera:

Presidente:	Marco Tulio Gallo Estrada
Secretario:	Humberto García Castellanos
Vocal:	Ana Isela García Acosta
Vocal Suplente:	Jorge De la Riva Rodríguez

Sin otro particular de momento, me es grato enviarle un cordial saludo.

A T E N T A M E N T E
"Excelencia en Educación Tecnológica"

JORGE ADOLFO PINTO SANTOS
JEFE DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN

C.c.p. Departamento de Servicios Escolares
JAPS/serc

Ciudad Juárez, Chihuahua, 04/junio/2024
Oficio Nº. DEPI/017/2024
Asunto: Autorización de Impresión de Tesis

C. KAREN LIZETH CARDENAS ALMANZA
CANDIDATO(A) AL GRADO DE MAESTRO(A) EN
MAESTRÍA EN INGENIERÍA INDUSTRIAL
P R E S E N T E.

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado "**DETERMINACIÓN DE TOXICIDAD DE LÍQUIDOS IONICOS UTILIZANDO DESCRIPTORES MOLECULARES SOAP EN CONJUNTO CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL**", ha informado a esta División de Estudios de Posgrado e Investigación, que está de acuerdo con el trabajo presentado. Por lo anterior se le autoriza se procesa con la **IMPRESIÓN DEFINITIVA DE SU TRABAJO DE TESIS.**

Esperando que el logro del mismo sea acorde con sus aspiraciones profesionales, reciba un cordial saludo.

A T E N T A M E N T E
"Excelencia en Educación Tecnológica"

JORGE ADOLFO PINTO SANTOS
JEFE DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN

EDUCACIÓN | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CIUDAD JUÁREZ
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

C.c.p. Departamento de Servicios Escolares
División de Estudios Profesionales

JAPS/serc

Instituto Tecnológico de Ciudad Juárez

División de Estudios de Posgrado e Investigación

CARTA CESIÓN DE DERECHOS

En la Ciudad de Juárez, Chihuahua; siendo el día **14** del mes de JUNIO del año **2024**, el (la) que suscribe **C. Karen Lizeth Cardenas Almanza** alumno(a) del Programa de **Maestría en Ingeniería Industrial**, con número de control **M21110207**, adscrito(a) a la **División de Estudios de Posgrado e Investigación**, manifiesto(a) que es el (la) autor(a) intelectual del presente trabajo de Tesis bajo la dirección del (de la, de los) **Maestro Marco Tulio Gallo** y cede los derechos del trabajo titulado **"DETERMINACIÓN DE TOXICIDAD DE LÍQUIDOS IÓNICOS UTILIZANDO DESCRIPTORES MOLECULARES SOAP, EN CONJUNTO CON ALGORITMOS DE INTELIGENCIA ARTIFICIAL"**, al Instituto Tecnológico de Ciudad Juárez para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del (de la) autor(a) y/o director(es) del trabajo. Este puede ser obtenido escribiendo a las siguientes direcciones **karen_cardenas_a@hotmail.com**. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Karen Lizeth Cardenas Almanza

Nombre y firma del alumno(a)

RESUMEN

La toxicidad de solventes ocasiona muchas enfermedades respiratorias, así como defectos en embarazos en diversas industrias en la región como es el caso de la industria automotriz, medica, de cementos, plásticos, etc.

Por esta razón es importante contar con modelos matemáticos aproximados que nos permitan su cálculo de forma rápida sin utilizar animales vivos y a bajo costo.

Por lo que en este trabajo utilizando descriptores moleculares del tipo SOAP desarrollamos un modelo regresivo para calcular la toxicidad de líquidos iónicos, en donde los datos experimentales fueron obtenidos del artículo de 'Machine Learning for Ionic Liquid Toxicity Prediction doi:10.3390/pr9010065' y en conjunto con herramientas de Machine Learning, como el algoritmo de regresión no-lineal "Kernel Ridge Regresion" se parametrizo el modelo matemático.

Los resultados predictivos muestran un nivel razonable de precisión, lo que permitirá reducir el tiempo de experimentación en seres vivos, así como poder determinar la toxicidad de sustancias químicas afines de forma rápida.

CONTENIDO

INTRODUCCIÓN 9

CAPÍTULO 1. PLANTEAMIENTO DEL PROBLEMA 10

1.1 DEFINICIÓN DEL PROBLEMA 12

1.2 OBJETIVO GENERAL 12

1.3 OBJETIVOS ESPECÍFICOS 12

1.4 PREGUNTAS DE INVESTIGACIÓN 12

1.5 HIPÓTESIS 13

1.6 DELIMITACIÓN 13

CAPÍTULO 2. MARCO TEÓRICO 14

2.1. CONCEPTOS RELACIONADOS CON EL TEMA DE INVESTIGACIÓN 14

2.1.1 Toxicidad 14

2.1.2 Solventes Orgánicos 15

2.1.3 Solventes verdes. 16

2.1.4 Líquidos Iónicos 17

2.1.5 Bases de Datos de Toxicidad 19

2.2 INTELIGENCIA ARTIFICIAL 20

2.2.1 Aprendizaje Maquina o Machine Learning (ML) 20

2.3 MODELOS MATEMÁTICOS 24

2.3.1 Definición de Kernel en Machine Learning 24

2.4 SOFTWARE 25

2.4.1 Python 25

2.4.2 Librería de descriptores moleculares para Machine Learning en la ciencia de materiales. "Dscribe" 27

2.4.3 Smooth overlap of atomic positions o Superposición suave de posiciones atómicas (SOAP) 27

2.5 PREDICCIÓN DE TOXICIDAD DE LÍQUIDOS IÓNICOS MEDIANTE TÉCNICAS DE MACHINE LEARNING 29

2.6 XGBOOST: REFUERZO DE GRADIENTE EXTREMO, EXTREME GRADIENT BOOSTING 30

CAPÍTULO 3. METODOLOGÍA 31

CAPÍTULO 4. RESULTADOS 43

4.1 Predicción de toxicidad Mediante XGBoost Y Kernel Ridge Regression 43
Capítulo 5. Conclusiones 50
Bibliografía 51

Índice de Figuras

Figura 1.1 Representación de un descriptor molecular en las técnicas de machine learning (Fullmetalfelix/ML-CSC-tutorial, n.d.)……11

Figura 2.1.4 Aplicaciones detalladas de los líquidos iónicos en diferentes industrias (Kaur y colaboradores,2022)……18

Figura 2.2.1 Diagrama esquemático de los procesos involucrados en el aprendizaje automático (Kim,2017) ……21

Figura 2.2.2 Aprendizaje Maquina aplicado a las ciencias ambientales e ingeniería (Zhong,2021) ……21

Figura 2.2.3. Esquema de una red neuronal profunda con 3 capas. (Wu Y. , 2018) ……24

Figura 2.3.1 Kernel que transforma los datos desde una representación bidimensional a una representación tridimensional. (Zhang, 2018)……25

Figura 2.4.2 Flujo con el cual machine learning se basa para la predicción de propiedades para estructuras atómicas. (Himanen, 2019)……27

Figura 2.4.3 La densidad local alrededor de un átomo central es expandida en funciones de base armónicas radiales y esféricas (Langer, 2022)……28

Figura 3.1. Etapas involucradas en el desarrollo de un modelo de aprendizaje maquina (Zhong, 2021)……32

Figura 3.2 Subrutina para obtener el descriptor molecular SOAP (Dscribe 2020)……33

índice de Tablas

Tabla 3.1 Moléculas y su grado de toxicidad experimental……42

INTRODUCCIÓN

La toxicidad de solventes ocasiona muchas enfermedades respiratorias, así como defectos en embarazos en diversas industrias en la región como es el caso de la industria automotriz, medica, de cementos, plásticos, etc. (Mora-Barrantes, 2022)

Por esta razón es importante contar con modelos matemáticos aproximados que nos permitan el cálculo de la toxicidad de forma rápida sin utilizar animales vivos y a bajo costo.

Como mencionan Jose F. Rodriguez Jr. y colaboradores (2021), hoy en día se están desarrollando nuevas tecnologías innovadoras, como la inteligencia artificial, que asemeja el cerebro humano, y el análisis de un gran número de datos, así como la identificación de patrones o relaciones entre ellos. Algunos de los beneficios del uso de la inteligencia artificial en la ciencia de materiales, es que permite el descubrimiento de nuevas sustancias, así como el análisis de compuestos ya existentes, todo esto en un menor tiempo, costo y esfuerzo.

En ciencias computacionales se emplea el lenguaje de programación para codificar algoritmos que enseñen a la computadora a ejecutar una tarea en particular, sin embargo, con el aprendizaje maquina o “machine learning” se desarrollan algoritmos que le dicen a la computadora como aprender, con un grupo de datos o ejemplos y algunas suposiciones subyacentes y así poder crear modelos matemáticos empíricos de razonable precisión. Generalmente los datos alimentados a la computadora provienen de experimentos de laboratorio. (Rodrigues, 2021)

En este trabajo utilizando descriptores moleculares del tipo SOAP desarrollamos un modelo regresivo no-lineal por computadora para calcular la toxicidad de líquidos iónicos en base a datos experimentales utilizando algoritmos de inteligencia artificial como el Kernel Ridge Regresión, para poder predecir de forma rápida el nivel de toxicidad de nuevas sustancias químicas afines al conjunto de entrenamiento, ya que en estas técnicas no es posible realizar extrapolaciones para sustancias completamente diferentes.

CAPÍTULO 1. PLANTEAMIENTO DEL PROBLEMA

La toxicidad de los compuestos químicos es de suma importancia; en diversas áreas científicas como: la industria farmacéutica para el desarrollo de medicinas y vacunas, en agricultura se emplean fertilizantes y diversas sustancias químicas en los cultivos, así como solventes que se utilizan en la industria química y de manufactura. Estos compuestos pueden migrar a diversos ecosistemas como el aire, suelo, agua y ocasionar enfermedades graves como son problemas: genéticos, cancerígenos, respiratorios, reproductivos a corto, mediano y largo plazo, entre otras consecuencias a la salud. (Mora-Barrantes, 2022)

Existen varios métodos in-vitro e in-vivo para determinar la toxicidad de compuestos químicos. Sin embargo, estas técnicas experimentales resultan muy costosas, requieren mucho tiempo para su investigación y en algunas ocasiones emplean seres vivos. Otra alternativa son las técnicas computaciones o en silicio que han abierto una nueva oportunidad para la predicción de toxicidad de compuestos químicos de forma rápida y con una precisión aceptable. Recientemente se han utilizado técnicas de aprendizaje maquina (Machine Learning) como método de 'k-vecinos más cercanos', máquinas de vector de soporte, arboles de decisión, técnicas de estructura-actividad (QSAR), regresión del tipo Kernel y redes neuronales artificiales para la predicción confiable y de bajo costo de compuestos tóxicos. (Karim, 2019)

Estas técnicas de inteligencia artificial o aprendizaje automático utilizan como entrada descriptores moleculares, es decir codifican la geometría espacial de las moléculas en forma numérica, para el desarrollo de modelos regresivos que nos permitan obtener la toxicidad. En la Figura 1, se observa la representación de un descriptor molecular en código binario que se alimentara a la computadora, el descriptor molecular debe tener ciertas características como ser suave (matemáticas), invariante a la permutación, traslación, rotación, además de describir de forma precisa el número de átomos, es decir el ambiente molecular cercano (que

enlaces y tipos de átomos se encuentran unidos entre sí, etc.). Su representación computacional para un ejemplo de cierto descriptor molecular es a partir de números binarios como son 010000101, como se observa en la figura. (Fullmetalfelix/ML-CSC-tutorial, n.d.)

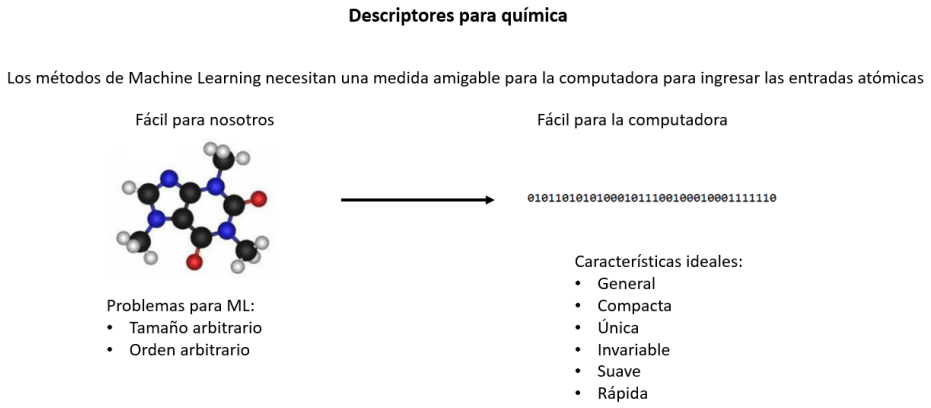


Figura 1.1 Representación de un descriptor molecular en las técnicas de machine learning (Fullmetalfelix/ML-CSC-tutorial, n.d.)

Una analogía simple para los descriptores moleculares, es el código de barras que se encuentra en los productos de los supermercados. “El **código de barras** es una representación por medio de un conjunto de líneas paralelas de distinto grosor y espaciado con información detallada sobre las características del producto, es decir, las barras y espacios del código representan pequeñas cadenas de caracteres”. Al escanear el código de barras, es posible identificar las características de forma rápida y precisa de un producto o bien de consumo. (Código de Barras, la enciclopedia libre, n.d.)

1.1 DEFINICIÓN DEL PROBLEMA

El uso de las técnicas de Inteligencia Artificial (Machine Learning) en la industria en México es limitado, por lo cual existe un gran potencial de aplicación en diversos procesos industriales, para minimizar los efectos de las sustancias peligrosas, toxicas y carcinógenas, así como, mejorar las condiciones ambientales de los trabajadores.

1.2 OBJETIVO GENERAL

Implementar técnicas simples de machine learning, mediante descriptores moleculares, para determinar el grado de toxicidad de compuestos químicos como solventes, para incrementar la sustentabilidad en los procesos industriales, y eliminar sustancias peligrosas, sustancias toxicas y carcinógenas, remplazándolas por solventes verdes.

1.3 OBJETIVOS ESPECÍFICOS

Utilizar descriptores moleculares como ‘Superposición suave de posiciones atómicas’ conocido por sus siglas en inglés como SOAP para predecir la toxicidad de compuestos químicos en base a datos experimentales.
Implementar técnicas de Machine learning como Regresión del tipo Kernel y XGBoost para crear un modelo matemático que nos permita identificar el grado de toxicidad de sustancias o solventes basados en líquidos iónicos.

1.4 PREGUNTAS DE INVESTIGACIÓN

¿Es posible que nuevos descriptores moleculares puedan ser utilizados para desarrollar modelos matemáticos que nos permitan determinar de forma precisa la toxicidad de un compuesto químico a base de líquidos iónicos?

¿Es posible implementar técnicas de machine learning simples, para identificar de forma robusta y confiable la toxicidad de compuestos químicos basados en líquidos iónicos utilizando determinados descriptores moleculares?

1.5 HIPÓTESIS

Se determinará la toxicidad de los compuestos químicos basados en líquidos iónicos de forma computacional mediante los descriptores moleculares SOAP.
Se determinará de forma rápida y precisa la toxicidad de compuestos químicos a a base de líquidos iónicos, utilizando técnicas de machine Learning como Regresión del Tipo Kernel o XGBoost, mediante los descriptores moleculares definidos anteriormente, con un algoritmo computacional utilizando librerías computacionales en lenguaje Python.

1.6 DELIMITACIÓN

Los compuestos tóxicos que se estudiaran con el descriptor molecular SOAP son obtenidos de base de datos publicadas en la literatura.
Se utilizarán los métodos de inteligencia artificial de Regresión del Tipo Kernel o técnicas XGBoost mediante cálculos computacionales en computadoras de escritorio, utilizando librerías científicas de código-abierto en python.

CAPÍTULO 2. MARCO TEÓRICO

2.1. CONCEPTOS RELACIONADOS CON EL TEMA DE INVESTIGACIÓN

2.1.1 Toxicidad

La palabra “toxicidad” describe el grado en el cual una sustancia es letal o capaz de producir una lesión en los seres vivos. La toxicidad depende de varios factores como dosis, tiempo de exposición, así como su ruta oral, respiratoria, cutánea, además de la forma y estructura de la sustancia química etc., (Módulo I - Introducción a la toxicología | Notas para la instrucción | ATSDR en español, 2022)

La toxicidad está presente en sustancias con las que convivimos diariamente y que pueden existir en nuestros espacios laborales, induciendo distintos tipos de enfermedades, cancerígenas, respiratorias, así como enfermedades neurodegenerativas. Ejemplos de enfermedades del sistema nervioso y del cerebro son: el Alzheimer, Parkinson y enfermedad de Huntington. Por ejemplo, la exposición a metales tales como el plomo pueden ocasionar enfermedades neurodegenerativas, esta exposición ocurre en industrias de esmaltado y refinería, manufactura de baterías, plásticos y pinturas.

Una vez que el plomo es absorbido puede acumularse en los órganos del cuerpo, como los huesos, cerebro, hígado e inclusive puede llegar al neonato en mujeres en estado de embarazo. El sistema nervioso es muy sensible a la exposición del plomo, y una exposición frecuente puede afectar el sistema renal neuromuscular y hepático. (García-Montalvo et al., 2019).

El cadmio, es otro metal muy empleado en la industria, y puede afectar al ser humano ocasionando osteoporosis, daño dermatológico y afecciones en el sistema nervioso. Un nivel alto de este metal en la sangre, provoca efectos neurotóxicos, alterando la concentración de neurotransmisores, afectando el cerebro, y promoviendo la demencia. (García-Montalvo et al., 2019)

El uso y la exposición de sustancias toxicas a la cual un trabajador está expuesto, en el ambiente industrial es de suma importancia en la actualidad. Se han establecido varios estándares internacionales para salvaguardar la salud de los trabajadores, sin embargo, aún existen casos en donde las personas están expuestas a sustancias toxicas durante largos periodos de tiempo. (Guidelines on the prevention of toxic exposures, 2004)

Además, existe en la mayoría de las especies el riesgo de sufrir alteraciones reproductivas o de desarrollo como la infertilidad, aborto, malformaciones físicas o deficiencias funcionales a nivel neurológico, endocrino o inmune. En el caso de defectos congénitos se estima que el 47% ocurre por causas desconocidas, 25% genéticas, otros 25% mediante una combinación de factores genéticos y ambientales y el 3% restante se debe a los ambientes físico, químicos o biológicos. Existe evidencia en la literatura de toxicidad por medio de agentes químicos e infecciosos en el ambiente afectando el desarrollo en diversos seres vivos. (Cavieres, 2004)

2.1.2 Solventes Orgánicos

El termino solventes orgánicos se refiere a compuestos volátiles o mezclas que tienen cierta estabilidad química y se encuentran en fase liquida. Los solventes orgánicos más comunes son hidrocarburos alifáticos, hidrocarburos cíclicos, hidrocarburos aromáticos, hidrocarburos halogenados, cetonas, aminas, ésteres, alcoholes, aldehídos y éteres. Muchos solventes se encuentran en forma de mezclas de compuestos químicos. ("Organic solvent neurotoxicity," 2020)
En 1984, 49 millones de toneladas de solventes industriales fueron producidos en Estados unidos. Además, basados en datos históricos de los años setenta, alrededor de 9.8 millones de trabajadores fueron expuestos a solventes orgánicos en los Estados Unidos. ("Organic solvent neurotoxicity," 2020)

Varios solventes orgánicos han sido categorizados por la NIOSH (National Institute for Occupational Safety and Health) como cancerígenos y con riesgos para la salud reproductiva y son solventes que se encuentran presentes en los espacios de trabajo. Por ejemplo, solventes como el benceno producen efectos neurotóxicos, y las enfermedades neurotóxicas son diez de las enfermedades que ocurren con mayor frecuencia entre los trabajadores. Estos solventes están presentes en las industrias de la: pintura, farmacéutica, textil, pegamentos, limpiadores, plásticos y agricultura.

La exposición de solventes orgánicos puede ocasionar efectos nocivos en la salud como: narcosis, anestesia, depresión del sistema nervioso central, fallas respiratorias, perdida de conciencia y en casos muy graves hasta la muerte; El instituto nacional de salud y seguridad ocupacional NIOSH en los Estados Unidos recomienda que se adopten controles de ingeniería para reducir la concentración y exposición de estos solventes a límites permitidos por la OSHA (administración de la salud y seguridad ocupacional), así como el uso de equipo y ropa de protección personal y programas de difusión entre los trabajadores. ("Organic solvent neurotoxicity," 2020)

2.1.3 Solventes verdes.

El impacto de las emisiones de los solventes en los ecosistemas y en los seres vivos requiere de implementaciones tecnológicas para minimizarlas o eliminarlas por completo. También es importante el concepto de un ‘solvente verde’, como aquella sustancia que es amigable con el medio ambiente y de baja ecotoxicidad. (Winterton, 2021)

La búsqueda de solventes no tóxicos para la industria requiere de innovaciones que permitan diseñar nuevos solventes, ya sea sintetizando nuevas sustancias o modificándolas al adicionar ciertos átomos o grupos funcionales moleculares (Winterton, 2021)

2.1.4 Líquidos Iónicos

Los líquidos iónicos son compuestos en fase liquida compuestos por aniones y cationes, los cuales poseen propiedades fisicoquímicas favorables, que permiten su uso en varias aplicaciones como electrolitos para baterías, en celdas de combustible, como agentes de extracción en procesos de separación químicos, y a la vez son más amigables con el medio ambiente que varios solventes orgánicos volátiles, y algunos líquidos iónicos poseen baja ecotoxicidad. (MacFarlane, 2014)

Kaur y colaboradores (2022), describen a los líquidos iónicos como sales orgánicas compuestas por un catión orgánico asimétrico y un anión orgánico o inorgánico que se encuentran en estado líquido a temperatura ambiente, a diferencia del cloruro de sodio que se encuentra en estado sólido a 25°C. Los cationes orgánicos como el amonio, sulfonio, imidazolio, triazolio, piridinio, fosfonio entre otros pueden ser combinados con diferentes aniones como tetrafloruro de borato, bromo, cloro, Bis(trifluorometilsulfonil)imida, etc., para diseñar líquidos iónicos de acuerdo a ciertas propiedades. Algunas propiedades interesantes de los líquidos iónicos incluyen no volatilidad, baja presión de vapor, no inflamabilidad, amplias ventanas electroquímicas, estabilidad ante altas temperaturas y buena conductividad eléctrica. Propiedades fisicoquímicas que los denominan como "solventes verdes", sin embargo, se ha encontrado que algunos de estos líquidos iónicos pueden ocasionar efectos tóxicos en los ecosistemas. (Kaur, 2022) .

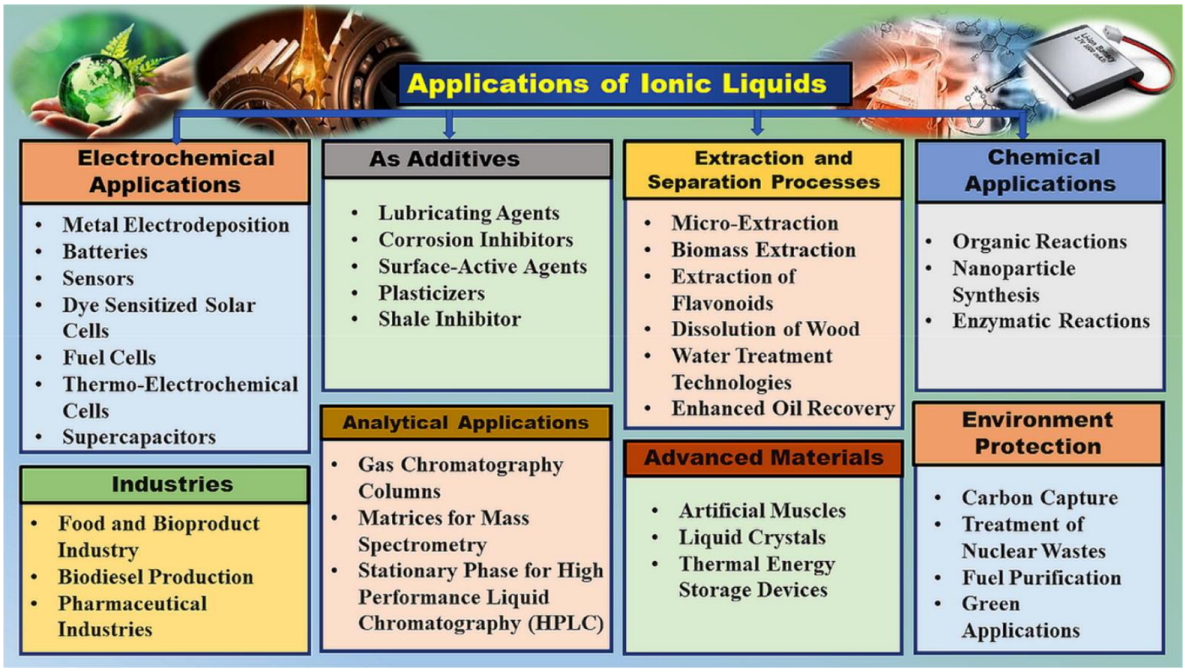


Figura 2.1.4 Aplicaciones detalladas de los líquidos iónicos en diferentes industrias (Kaur y colaboradores, 2022).

Algunas aplicaciones de los líquidos iónicos se muestran en la figura 2.1.4 en celdas de combustible, supercapacitores, celdas solares sensibilizadas y celdas termo electromecánicas, producción de biocombustible, transferencia y almacenamiento de calor, conversión de energía electroquímica, conversión y captura de calor, almacenamiento de energía electroquímica, entre otras. Un ejemplo del uso de los líquidos iónicos es en baterías de litio presentando un mayor rendimiento energético, baja volatilidad y flamabilidad, constituyendo una fuente de energía estable y segura. Sin embargo, un factor que ha detenido su comercialización se debe a los altos costos de síntesis de algunos líquidos ionicos. (MacFarlane, 2014)

Los líquidos iónicos son sales fundidas con diversas propiedades favorables, generalmente poseen una mayor conductividad térmica y eléctrica, así como propiedades electromecánicas en comparación con otras sustancias orgánicas tradicionales. Algunos Líquidos iónicos son capaces de disolver celulosas y extraer varios compuestos químicos lo que les da un gran potencial en la producción de biocombustibles, además de ser utilizados en catálisis y como sustancias capaces

de secuestrar gases responsables del efecto de invernadero como dióxido de carbono (Zhou, 2023)

Estudios en la literatura han señalado que los líquidos iónicos son eficientes en catálisis y síntesis química, electroquímica, conversión de biomasa, biocombustibles, biotecnología y muchos otros campos. Algunos ejemplos de los líquidos iónicos en el campo de la industria farmacéutica y medicinas son: agentes para la liberación de medicamentos, biosensores, etc; Existen 3 generaciones de líquidos iónicos de acuerdo a su estructura química y sus propiedades. La primera generación combina cationes del tipo alquilimidazolio y alquilpiridinio con aniones de haluros metálicos. La segunda generación involucra cationes del tipo alquilimidazolio, alquilpiridinio, amonio y fosfonio, en combinación con aniones del tipo haluros, tetrafluoroborato y hexafluorofosfato. La tercera generación de IL emplea iones a base de constituyentes naturales y biodegradables, como colina y aminoácidos (Egorova, 2017).

2.1.5 Bases de Datos de Toxicidad

Tox21 “Toxicology in the 21st century’ en español toxicología en el siglo 21, es una base de datos de acceso abierto que contiene información sobre la toxicidad de más de 8500 compuestos químicos, además, es una base de datos de toxicidad muy amplia que involucra varias agencias de gobierno en los Estados Unidos como: environmental protection agency (EPA), National center for advancing transnational sciences (NCATS), National toxicologic program (NTP) y Food and Drug Administration (FDA). Uno de los objetivos de estas agencias era el poder transitar de modelos mamíferos a modelos in vitro y no mamíferos en la investigación toxicológica (Roper, 2020).

El programa Nacional de Toxicología en los Estados Unidos (NTP) orienta el programa tox21 mediante avances científicos encaminados a mejorar la salud pública. Cada una de las diversas agencias de gobierno aporta con su conocimientos y desarrollos, por ejemplo, la NTP se enfoca en la toxicología

experimental, la NCGC (National Center of Advanced Translational Science) en tecnologías de alto rendimiento y la EPA en toxicología computacional. En el 2010 la FDA (Food and Drug administration) se unió a la base de datos tox 21 incorporando varias sustancias que regula, además de coadyuvar con investigaciones basadas en modelos de animales para enfermedades humanas y análisis de rutas de toxicidad en el cuerpo humano. (Roper, 2020)

2.2 INTELIGENCIA ARTIFICIAL

2.2.1 Aprendizaje Maquina o Machine Learning (ML)

El aprendizaje automático (Machine Learning) es una técnica computacional que utiliza datos para crear modelos generalmente regresivos, infiriendo las relaciones o patrones existentes entre los datos. En donde el termino de datos se puede referir la información en forma tabular, audio, imágenes, etc., El "modelo" es el resultado que por medio de la técnica de Machine Learning permite predecir o generar información a partir de valores nuevos para las variables de entrada (Predictorio). (Kim, 2017)

El nombre de Machine Learning se debe a que analiza los datos de entrenamiento y desarrolla el modelo con la mínima intervención humana. Los datos que se alimentan para la creación del modelo matemático se denominan datos de "entrenamiento". La figura 2.2.1 ilustra los procesos involucrados en el aprendizaje-maquina o el Machine Learning, como resultado la salida del modelo constituye una predicción basada en nuevos datos de alimentación denominada (inferencia). Por ejemplo, por medio de la creación de modelos de ML podemos conocer el grado de toxicidad de una nueva sustancia y el nivel de concentración considerado como peligroso en el ser humana (log (EC50)).

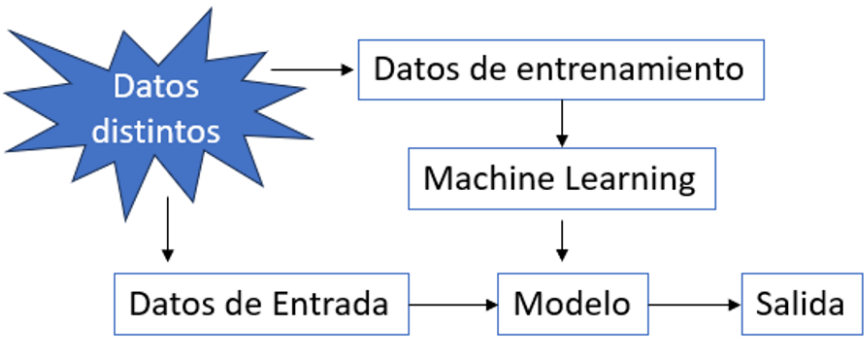


Figura 2.2.1 Diagrama esquemático de los procesos involucrados en el aprendizaje automático (Phil, 2017)

El resultado del modelo depende de forma crucial de la calidad de los datos del conjunto de entrenamiento. Existen diferentes algoritmos para la creación del modelo matemático como son: Regresión del tipo Kernel, Redes Neuronales Artificiales, Arboles de búsqueda, etc., (Kim, 2017)

Como observamos en la figura 2.2.2 en machine learning se encuentran con varias entradas de datos como son: los datos de forma tabular, texto, grafica o imágenes. Estos son alimentados a las redes neuronales, las cuales ayudan a identificar las variables importantes y a descubrir otras que impactan al modelo. El modelo nos arroja información como predicción o datos estadísticos relevantes y nos ayuda a determinar anomalías en estos.

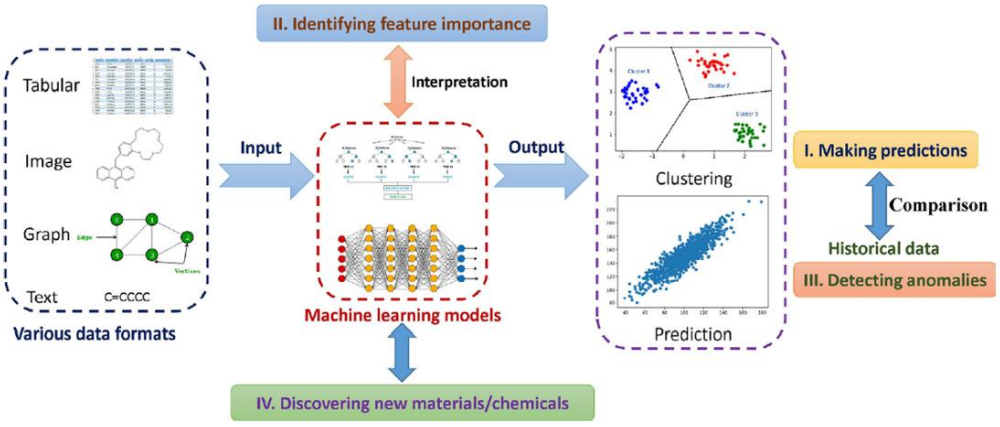


Figura 2.2.2 Aprendizaje Maquina aplicado a las ciencias ambientales e ingeniería (Zhong,2021)

Una aplicación de Machine Learning, es para la predicción de solidos suspendidos en el aire en diferentes regiones espaciales, la predicción del suministro de agua, modelado bioquímico de plantas de tratamiento de aguas residuales, diseño de nuevos materiales para secuestrar el dióxido de carbono, el diseño de nuevos fármacos, así como plásticos biodegradables, predicciones climáticas, predicciones de terremotos y tsunamis, etc. (Zhong, 2021)

Para desarrollar el modelo de ML de predicción de la concentración de PM2.5 solidos suspendidos de forma diaria o anual, las variables de entrada (features o características) son temperatura del aire, temperatura de roció, humedad relativa, densidad de flujo de radiación de onda corta y larga, velocidad del viento, densidad de forestación, acceso a vías libres en carreteras, etc., (Zhong, 2021)

La predicción del grado de toxicidad de sustancias basada en algoritmos de inteligencia artificial se pueden obtener mediante el uso de redes neuronales y se requiere de suficiente información como conjunto de entrenamiento, y así poder obtener una respuesta más precisa que nos permita determinar los efectos de las sustancias químicas relacionados con la carcinogenicidad, o efectos en algún sistema del cuerpo humano, tales como efectos en el sistema reproductivo o nervioso. (Karim, 2019)

En su artículo Wu y colaboradores (Wu Y. , 2018) indican que la inteligencia artificial, actualmente se aplicar en áreas médicas y farmacéuticas, ayudando a reducir los tiempos de investigación y la experimentación en seres vivos gracias a la predicción tan exacta que arroja sobre la toxicidad de sustancias utilizadas en los medicamentos y en las vacunas.

Estos autores señalan que la mayoría de los medicamentos atraviesan por muchos experimentos, largos y tediosos para evaluar su grado de toxicidad, y más de la mitad de los medicamentos evaluados en laboratorios resultan ser dañinos de cierta forma en las personas. También, al experimentar con animales, los resultados obtenidos, no son del todo precisos, debido a la diferencia entre el organismo animal con el organismo humano.

Actualmente los modelos de inteligencia artificial basados en redes neuronales artificiales y en arboles de decisiones, han sido utilizados en aplicaciones como reconocimiento de imágenes, identificación de patrones, resultando en modelos matemáticos predictivos de bajo costo, y con gran exactitud aplicados en la industria farmaceutica (Wu Y. , 2018)

Tang y colaboradores (Tang, 2019) mencionan que el aprendizaje profundo, utiliza las DNN (Deep Neural networks) en español redes neuronales que contienen varias capas ocultas, para el análisis de datos y la creación de modelos matemáticos regresivos no-lineales. El Deep learning, ha resultado ser un modelo de inteligencia artificial eficaz para la predicción de toxicidad de gran número de sustancias. Como se observa en la figura 2.2.4, está constituido por una o más capas de entrada, varias capas ocultas y una o más capa de salida. Para el procesamiento de la información, cada neurona recibe la información de la capa anterior (valores de las variables independientes o x´s, y por medio de funciones matemáticas no-lineales, trasmite la información hacia las capas de salida en donde se obtiene predicción final o el valor de y, en este caso el grado de toxicidad o el log(EC50) (Tang, 2019).

En la vida cotidiana las personas se encuentran en contacto con una gran cantidad de sustancias químicas, naturales o sintéticas, por lo cual es de suma importancia determinar el grado de toxicidad en los diferentes sistemas que comprende el cuerpo humano. Los modelos más utilizados para la predicción de la toxicidad de las sustancias son los QSAR (Relación cuantitativa estructura-actividad), de tal manera que las sustancias con estructuras químicas similares podrían tener el mismo grado de toxicidad. (Tang, 2019).

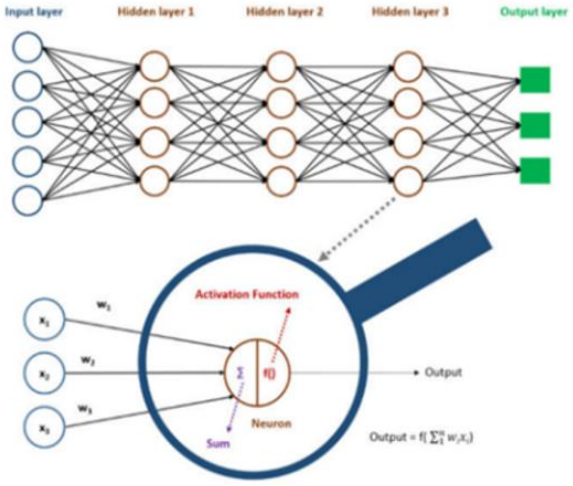


Figura 2.2.3. Esquema de una red neuronal profunda con 3 capas. (Tang, 2019)

2.3 MODELOS MATEMÁTICOS

2.3.1 Definición de Kernel en Machine Learning

Un aspecto clave en ML es la utilización de Kernels para la determinación de los diferentes modelos matemáticos. En esencia un Kernel (k) es una función matemática que me permite proyectar datos (x, y) de una dimensión a otra de mayor orden. (Zhang, 2018, Vovk, V. , 2013)

$$\begin{aligned} k(\mathbf{x}, \mathbf{y}) &= (\mathbf{x}^T \mathbf{y})^2 \\ &= (x_1 y_1 + x_2 y_2 + x_3 y_3)^2 \\ &= \sum_{i,j=1}^{3} x_i x_j y_i y_j \end{aligned}$$

En la figura 2.3.1 se observa que si se aplica una función matemática (kernel) es posible mapear los datos del espacio bidimensional al espacio tridimensional y así poder determinar una superficie que divida los datos en diferentes clases.

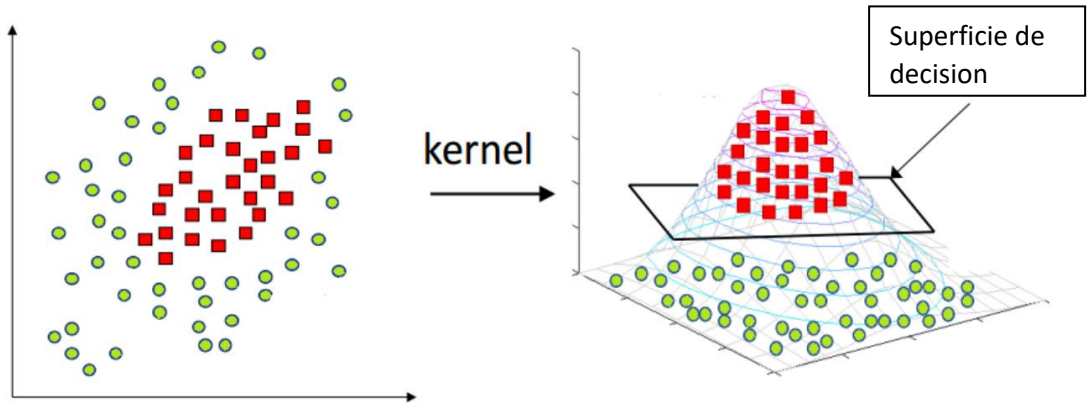


Figura 2.3.1 Kernel que transforma los datos desde una representación bidimensional a una representación tridimensional. (Zhang, 2018)

2.4 SOFTWARE.

2.4.1 Python

La inteligencia artificial o machine learning, se basa en el manejo de grandes cantidades de datos, por lo cual para crear modelos matemáticos regresivos, así como visualizaciones y análisis estadísticos. Actualmente el lenguaje de programación basado en objetos mayormente utilizado en inteligencia artificial es Python, ya que dispone de grandes cantidades de subrutinas y códigos de acceso libre, tales como la librería Pandas, que se utiliza para la lectura, manipulación y análisis de base de datos, así como las librerías numéricas para el manejo de matrices como numpy, matplotlib, tensorflow, scikit-learn, pytorch, etc., (Challenger-Pérez, 2014)

La librería Scikit-learn en Python incluye varios algoritmos de aprendizaje máquina para resolver problemas de aprendizaje supervisados y no supervisados a gran escala. Su amplia utilización, se basa en la creciente necesidad de análisis de datos de personas no especializadas en softwares computacionales para su uso en las diferentes industrias, en la medicina, física o biología (Deitel, 2021)

Numpy, (Python numérico) es una librería científica de algebra lineal, que involucra, vectores, tensores o matrices, así como operaciones aritméticas básicas y complejas.

Scipy, (Python científico) contiene subrutinas de análisis numérico, funciones especiales, así como funciones estadísticas básicas. Scipy puede interfacearse con otros paquetes numéricos en Fortran o en C, como LAPACK para realizar operaciones matemáticas de forma rápida y en el menor tiempo de computo. (Pedregosa, 2011)

Python es una excelente opción como lenguaje de programación para principiantes, ya que es un lenguaje de programación orientado a objetos que no requiere de compilación. Fue desarrollado en 1991 por Guido van Rossum del Instituto Nacional de Investigación de Matemáticas y Ciencias de la Computación en Amsterdam. Algunas de las razones por las que Python se está convirtiendo en uno de los lenguajes de programación más utilizados en el ámbito de la inteligencia artificial son las siguientes:

• Es de código abierto, gratuito y ampliamente disponible en la comunidad científica con una infinidad de códigos de aplicación en diversas áreas.

• Es más amigable y de mayor facilidad de comprensión, en comparación a los lenguajes como C, C++, Fortran y Java.

• Mejora la productividad de los usuarios, debido a la existencia de un gran numero de bibliotecas de código abierto o subrutinas elaboradas por terceros, disponibles en internet, lo cual permite escribir códigos más rápido y realizar tareas complejas con mínima energía.

2.4.2 Librería de descriptores moleculares para Machine Learning en la ciencia de materiales. “Dscribe”

Dscribe es un software computacional para el cálculo de descriptores moleculares dentro de los algoritmos de inteligencia artificial como se observa en la Figura 2.4.2.1.

El paquete incluye varios descriptores moleculares como la matriz de Coulomb, representación de muchos cuerpos tensorial (MBTR), funciones de simetría centradas en átomos (ACSF) y superposición suave de posiciones atómicas “Smooth overlap of atomic positions (SOAP)”. Esta librería tiene una interfaz de Python con rutinas computacionales escitas en C y en C++ (Himanen, 2020).

En la figura 2.4.2 se observa como la estructura tridimensional de una molécula por medio de un descriptor molecular se traduce o mapea a un vector numérico, para alimentarse al modelo de aprendizaje, basado en redes neuronales y finalmente obtener la propiedad fisicoquímica o de interés de una molécula al crear el modelo matemático de regresión no-lineal.

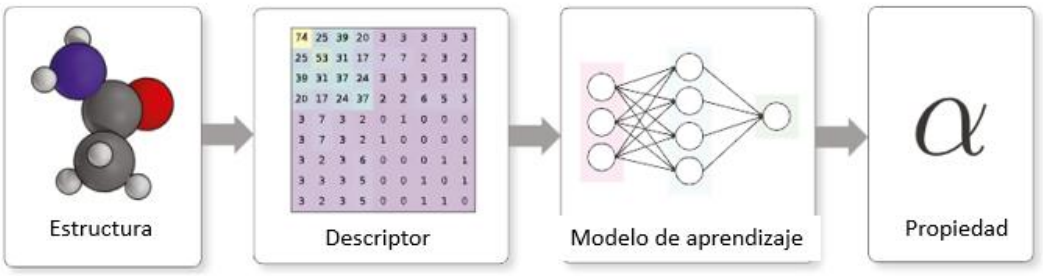


Figura 2.4.2 Flujo computacional de machine learning para la construcción de un modelo mattematico regresivo para la predicción de propiedades fisicoquimicas (Himanen, 2020).

2.4.3 Smooth overlap of atomic positions o Superposición suave de posiciones atómicas (SOAP)

El descriptor molecular SOAP (Bartók y col, 2013, Bartók y col. 2017, Willat y col. 2019, Langer y col. 2022) como se observa en la figura 2.4.3 se basa en una

representación local de la densidad atómica en un cierto radio r, por medio de la expansión en funciones de base gaussianas ortogonales radiales y esféricas mediante la siguiente ecuación:

$$\rho(\mathbf{r}) = \sum_{n,l,m} c_{nlm}\, g_n(\mathbf{r})\, Y_{lm}(\mathbf{r})$$

En donde c_{nlm} son los coeficientes de expansión, g_n y Y_{lm} son las funciones de base radiales y esféricas armónicas. A partir de los coeficientes es posible obtener espectros de potencia, que describen las interacciones entre dos y tres cuerpos en un determinado radio.

$$p_{nn'\ell} = \sum_{m} c_{n\ell m} c^{*}_{n'\ell m}$$

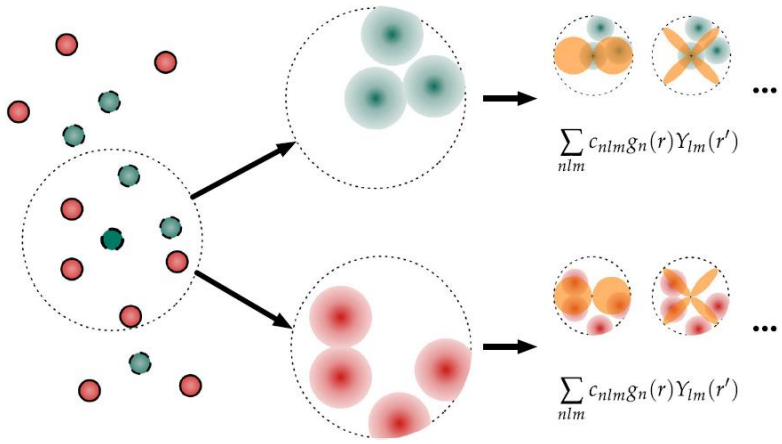


Figura 2.4.3 La densidad local alrededor de un átomo central es expandida en funciones de base armónicas radiales y esféricas (Langer, 2022)

Estos descriptores moleculares son invariantes con respecto a la traslación, permutación y rotación, y se utilizan para codificar la geometría espacial de las moléculas en vectores numéricos. Estos vectores numéricos se promedian en base al número de átomos con la finalidad de que los vectores tengan la misma dimensión en diferentes moléculas compuestas de números de átomos desiguales. (Langer, 2022)

2.5 PREDICCIÓN DE TOXICIDAD DE LÍQUIDOS IÓNICOS MEDIANTE TÉCNICAS DE MACHINE LEARNING

Los líquidos iónicos (LI) presentan varias características fisicoquímicas favorables, como bajo punto de fusión, baja volatilidad y alta estabilidad térmica. Por estas razones, los LI constituyen una alternativa como solventes verdes para remplazar los solventes orgánicos tradicionales. A pesar de sus amplias propiedades y grandes perspectivas de aplicación, muchos LI presentan posibles efectos negativos sobre los ecosistemas. La toxicidad es una variable de suma importancia en la selección de los líquidos iónicos para su utilización en proceso químicos sustentables. (Wang, 2020)

En este contexto Wang y colaboradores utilizaron las técnicas basadas en aprendizaje maquina (ML) para predecir la toxicidad de IL basándose en un conjunto de datos experimentales (logaritmo de la concentración media efectiva, log(EC50)) utilizando líneas de células de ratones del tipo IPC-81, por medio de un procedimiento sistemático que incluye la utilizacion de descriptores moleculares estructurales, optimización de hiperparámetros, entrenamiento y evaluación de los modelos matemáticos obtenidos. En este estudio la red neuronal de alimentación hacia adelante (FNN) y algoritmos de máquina de vectores de soporte (SVM) se adoptaron para predecir la toxicidad de las IL directamente de sus estructuras moleculares, obteniéndose un coeficiente de ajuste R^2 superior a 0.90

Wu y colaboradores utilizaron descriptores basados en la existencia de ciertos grupos funcionales en el catión y anión, tamaño de la cadena alquílica, numero de átomos de oxígeno, numero de átomos sustituyentes en el catión, para determinar la toxicidad de 304 líquidos iónicos en la línea celular de leucemia de rata IPC-81, obteniendo un coeficiente de determinación de alrededor de 0.90 (Wu T. , 2020)

2.6 XGBoost: Refuerzo de Gradiente Extremo, Extreme Gradient Boosting

El algoritmo “XGBoost” es un algoritmo de aprendizaje máquina que se basa en la combinación lineal de un ensamble de modelos multiplicados por sus respectivos pesos, en donde cada modelo está conformado por árboles de decisión. Los modelos se añaden uno por uno de forma secuencial con la finalidad de mejorar los errores de predicción en cada iteración, una especie de refuerzo o boosting, en donde se busca que la nueva función maximice el producto interno con el gradiente negativo de la función de pérdida o error evaluada en la iteración previa (Brownlee, 2021; Chen & Guestrin, 2016); El algoritmo XGBoost se encuentra disponible como código abierto en internet para su utilización en la creación de modelos matemáticos regresivos.

CAPÍTULO 3. METODOLOGÍA

La figura 3.1 muestra las tres etapas involucradas en el aprendizaje automático (ML); En la primera denominada preprocesamiento o preparación de modelaje es importante contar con datos significativos y representativos en lugar de datos inmensos y superfluos. En esta primera etapa se deben de escalar los valores de las variables, para que los valores se encuentren en el mismo orden de magnitud (normalización), también es importante eliminar los valores extremos (Outliers) para evitar ruido en el modelo matemático, generalmente los datos se dividen en tres bloques que son: entrenamiento, prueba y validación donde se utilizan en el proceso el 80%, 10%, 10% respectivamente del total de los datos. Los datos de validación se utilizan para la optimización de los hiper parámetros del modelo matemático, que puede ser las variables del kernel, el número de capas de la red neuronal, las diferentes funciones matemáticas usadas, etc. La segunda etapa, conocida como desarrollo de modelo, involucra el entrenamiento basado en métricas de salida como el error medio al cuadrado (RMSE) y se basa en una serie de iteraciones hasta determinar los parámetros del modelo matemático. Subsecuentemente se procede a la optimización de los hiper parámetros con el conjunto de validación y posteriormente al empleo del conjunto de prueba. (Zhong, 2021)

La etapa después de modelado final es el despliegue del modelo, basado en datos frescos o nuevos para las diferentes predicciones o clasificaciones científicas, o determinación de características significativas entre las variables del modelo. (Zhong, 2021)

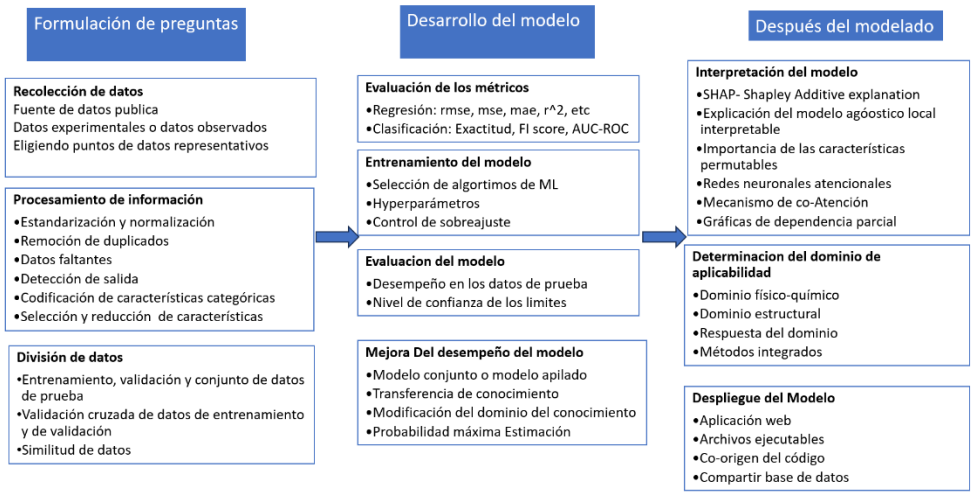


Figura 3.1. Etapas involucradas en el desarrollo de un modelo de aprendizaje maquina (Zhong, 2021)

Las moléculas utilizadas para la predicción de la toxicidad con respecto a la línea celular en ratas IPC-81 (log EC50), se basan en solventes a base de líquidos iónicos y se obtuvieron de las siguientes fuentes bibliográficas (Sosnowska y col. 2017, Wang y col., 2020, Wu y col. 2020)

La estructura de los 355 líquidos iónicos se encuentran en formato SMILES “Simplified Molecular Input Line Entry System” (Weininger, 1988) y se convirtieron a formato XYZ para ser alimentados a la subrutina SOAP en Dscribe utilizando el software open babel (O'Boyle, N.M. y col. 2011)

La siguiente figura 3.2 muestra la subrutina computacional utilizada en el cálculo de los descriptores soap obtenida del paquete computacional DSCRIBE.

```python
from ase.build import molecule
from dscribe.descriptors import SOAP

# Define the atomic system
system = molecule("H2O")

# Setup the descriptor
soap = SOAP(
    species=["H", "O"],
    rbf="gto",
    rcut=3,
    nmax=10,
    lmax=8,
    sparse=True
)

# Query the number of features
n_features = soap.get_number_of_features()

# Get output as a dense or sparse array
output = soap.create(system)

# Get output for multiple systems
systems = [molecule("H2"), molecule("O2")]
outputs = soap.create(systems, n_jobs=2)
```

Figura 3.2 Subrutina para obtener el descriptor molecular SOAP (Smooth Overlap of Atomic Positions — DScribe 0.3.x Documentation, n.d.)

Una vez teniendo la información de una manera que la computadora lo puede leer, es necesario normalizar los valores de x, y, en donde x corresponde a el vector numérico de SOAP para cada sustancia Y, donde Y es el valor logarítmico de la concentración media de toxicidad en la línea celular IPC-81, después se procede a la utilización de las subrutinas en Python (scikit-learn) para dividir el conjunto de datos en subconjuntos de entrenamiento y prueba (80-20) utilizando los algoritmos Kernel Ridge Regression, y Xgboost.

A continuación, se presentan las moléculas y su grado de toxicidad experimental utilizados en este trabajo de investigación (Sosnowska y col. 2017, Wang y col., 2020, Wu y col. 2020, Zhao y col 2014):

Id	SMILES	**Toxicidad Experimental logEC50**
1	[N+](C)(C)(CC)COCC.[Cl-]	3.59
2	[N+](C)(C)(CC)COCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.8

3	O1c4c(O[B-]12Oc3c(O2)cccc3)cccc4.CC[N+](CC)(CC)CC	1.17
4	[N+](C)(C)(Cc1ccccc1)CCCCCCCCCC.[Cl-]	0.64
5	[N+](C)(C)(CC)CC(=O)OCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.53
6	[N+](C)(C)(C)CCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.61
7	[N+](C)(C)(CC)CCOCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.27
8	[N+](C)(C)(CC)CCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.7
9	[N+](C)(C)(CC)CCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.3
10	[N+](C)(C)(CC)CCCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.54
11	[N+](CCCC)(CCCC)(CCCC)CCCC.[Br-]	2.25
12	[N+](C)(C)(CC)CC#N.[Cl-]	4.6
13	[N+](C)(C)(CC)CCOC.[Cl-]	4.6
14	[N+](C)(C)(CC)CCO.[I-]	4.6
15	[N+](C)(C)(CC)CCOCC.[Cl-]	4.6
16	[N+](C)(C)(CC)CCCOC.[Cl-]	4.9
17	[N+](C)(C)(CC)CC#N.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.86
18	[N+](C)(C)(CC)CCCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.83
19	[NH2+](CCOC)CCOC.[O-]S(=O)(=O)N	3.48
20	[N+](C)(C)(C)CCO.[O-]C(=O)CO	3.89
21	[NH3+]CCO.[O-]C=O	4.15
22	[Cl-].O(CC)CC[NH+](C)C	4.6
23	[N+](C)(C)(CC)CCCC.[Cl-]	4.6
24	[N+](C)(C)(C)CCO.[O-]C(=O)C(C)O	4.6
25	[N+](C)(C)(Cc1ccccc1)CCCCCCCCCCCCCC.[Cl-]	0.16
26	[N+](C)(C)(Cc1ccccc1)CCCCCCCCCCCC.[Cl-]	0.28
27	[N+](C)(C)(Cc1ccccc1)CCCCCCCCCCCCCCCC.[Cl-]	0.33
28	[NH+](C)(C)CCO.[O-]C(=O)C	3.39
29	[NH+](C)(C)CCO.[O-]C(=O)CO	3.4
30	[N+](CC)(CC)(CC)CC.[Cl-]	3.78
31	[N+](C)(C)(C)C.[P-](F)(F)(F)(C(F)(F)C(F)(F)F)(C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)F	2.03
32	[N+](C)(C)(CC)CCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.53
33	[N+](CCCC)(CCCC)(CCCC)CCCC.[B-]12(Oc3ccccc3O1)Oc3ccccc3O2	1.61
34	[N+](C)(C)(CC)CC(=O)OCC.[Br-]	3.85
35	[N+]1(C)=CN(C=C1)CC.[B-](F)(F)(F)F	3.44
36	[N+]1(C)=CN(C=C1)CCCCCCCC.[Cl-]	2.012837
37	[N+]1(C)=CN(C=C1)CCCCCCCC.[P-](F)(F)(F)(F)(F)F	1.96
38	[N+]1(C)=CN(C=C1)CCCCCCCC.[B-](F)(F)(F)F	1.591065
39	[N+]1(C)=CN(C=C1)CCCCCCCCC.[Cl-]	1.399674
40	[N+]1(C)=CN(C=C1)CCCCCCCCC.[P-](F)(F)(F)(F)(F)F	1.854913
41	[N+]1(C)=CN(C=C1)CCCCCCCCC.[B-](F)(F)(F)F	1.65
42	[N+]1(C)=CN(C=C1)CCC.[B-](F)(F)(F)F	3.454845
43	[N+]1(C)=CN(C=C1)CCCC.[B-](F)(F)(F)F	3.230449
44	[N+]1(C)=CN(C=C1)CCCC.[Br-]	3.426511
45	[N+]1(CC)=CN(C=C1)CCCC.[B-](F)(F)(F)F	3.257679

46	[Cl-].c1c[n+](cn1C)CCCCC	3.16
47	F[P-](F)(F)(F)(F)F.c1c[n+](cn1C)CCCCC	3.07
48	[Cl-].OCCC[n+]1ccn(c1)C	3.09
49	[N+]1(C)=CN(C=C1)CCCCCC.[Cl-]	2.85
50	[N+]1(C)=CN(C=C1)CCCCCC.[B-](F)(F)(F)F	2.980003
51	[N+]1(CC)=CN(C=C1)CCCCCC.[Br-]	2.01
52	[N+]1(CC)=CN(C=C1)CCCCCC.[B-](F)(F)(F)F	2.26
53	[N+]1(C)=CN(C=C1)CCCCCCC.[Cl-]	2.53
54	[N+]1(C)=CN(C=C1)CCCCCCC.[P-](F)(F)(F)(F)(F)F	2.3
55	[N+]1(C)=CN(C=C1)CCCCCCC.[B-](F)(F)(F)F	2.58
56	[N+]1(C)=CN(C=C1)CCCCCCCCCC.[Cl-]	1.34
57	[N+]1(C)=CN(C=C1)CCCCCCCCCC.[P-](F)(F)(F)(F)(F)F	1.498311
58	[N+]1(C)=CN(C=C1)CCCCCCCCCC.[B-](F)(F)(F)F	0.769377
59	[N+]1(CC)=CN(C=C1)CCCCCCCCCC.[Br-]	0.53
60	[N+]1(C)=CN(C=C1)CCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	2.682145
61	FC(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F.OCC[n+]1ccn(c1)C	3.763428
62	[N+]1(C)=CN(C=C1)Cc1ccccc1.[B-](F)(F)(F)F	2.974051
63	CCOS([O-])(=O)=O.CCN1C=C[N+](C)=C1	3.928396
64	[N+]1(C)=CN(C=C1)CCCC.[Cl-]	3.553883
65	[N+]1(C)=CN(C=C1)CCCCCC.[P-](F)(F)(F)(F)(F)F	2.909021
66	[N+]1(C)=CN(C=C1)CCCC.[P-](F)(F)(F)(F)(F)F	3.09691
67	FC(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F.O(C)C(Cn1cc[n+](c1)C)C	3.342423
68	[Cl-].O(C)C(Cn1cc[n+](c1)C)C	4.49
69	[N+]1(CC)=CN(C=C1)CCCC.[O-]C(=O)C(F)(F)F	3.313867
70	[N+]1(CC)=CN(C=C1)CCCC.[O-]S(=O)(=O)C(F)(F)F	3.426511
71	[N+]1(C)=CN(C=C1)CCCC.[O-]S(=O)(=O)C(F)(F)F	3.021189
72	CCCCN1C=C[N+](=C1)C.C(#N)[N-]C#N	3.15
73	[N+]1(C)=CN(C=C1)CCCC.[O-]S(=O)(=O)OCCOCCOC	3.158362
74	[N+]1(C)=CN(C=C1)CCCC.[O-]S(=O)(=O)OC	3.212188
75	[N+]1(C)=CN(C=C1)CCCC.[O-]S(=O)(=O)OCCCCCCCC	3.225309
76	[N+]1(C)=CN(C=C1)CCCC.[O-]S(=O)(=O)O	3.287802
77	CCCCN1C=C[N+](=C1)C.CC1=CC=C(C=C1)S(=O)(=O)[O-]	3.29
78	[N+]1(C)=CN(C=C1)CCCC.[S-]C#N	3.416641
79	[N+]1(C)=CN(C=C1)CCCC.[O-]S(=O)(=O)C	3.511883
80	[N+]1(C)=CN(C=C1)CCCC.[I-]	3.481443
81	[N+]1(C)=CN(C=C1)CC.[B-]12(OC(=O)C(=O)O1)OC(=O)C(=O)O2	2.93
82	[N+]1(C)=CN(C=C1)CC.[B-](C#N)(C#N)(C#N)C#N	3.5
83	[N+]1(C)=CN(C=C1)CC.[P-](F)(F)(F)(F)(F)F	3.91803
84	[N+]1(C)=CN(C=C1)CC.[O-]C(=O)C(F)(F)F	4.004321
85	CCN1C=C[N+](=C1)C.CC1=CC=C(C=C1)S(=O)(=O)[O-]	3.81
86	[N+]1(C)=CN(C=C1)CC.[O-]S(=O)(=O)O	3.99
87	[N+]1(C)=CN(C=C1)CC.[O-]S(=O)(=O)C(F)(F)F	4.08636
88	[N+]1(C)=CN(C=C1)CCCCCCCCCCCCCCCC.[Cl-]	-0.24033

89	[O-]C2=NS(=O)(=O)c1ccccc12.c1c[n+](cn1CCCCCC)C	2.29
90	[N+]1(C)=CN(C=C1)CCc1ccccc1.[P-](F)(F)(F)(F)(F)F	2.927883
91	Cc1ccc(cc1)Cn2cc[n+](c2)C.[Cl-]	2.64
92	F[B-](F)(F)F.c1cc(ccc1C)Cn2cc[n+](c2)C	2.67
93	F[B-](F)(F)F.c1c[n+](c(n1C)C)CCCCCC	1.9
94	[N+]1(C)=CN(C=C1)CCCCCCCCCCCCCCCCCC.[Cl-]	0.012837
95	FC(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F.c1c[n+](cn1CCCCCCCC)C	1.64
96	[N+]1(C)=CN(C=C1)CCCCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	2.243038
97	[N+]1(C)=CN(C=C1)CC.[P-](F)(F)(F)(C(F)(F)C(F)(F)F)(C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)F	2
98	[N+]1(C)=CN(C=C1)CCCC.[P-](F)(F)(F)(C(F)(F)C(F)(F)F)(C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)F	1.374748
99	[N+]1(C)=CN(C=C1)CCCCCC.[P-](F)(F)(F)(C(F)(F)C(F)(F)F)(C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)F	1.428135
100	[N+]1(C)=CN(C=C1)CCCCCC.[P-](F)(F)(F)(C(F)(F)C(F)(F)C(F)(F)F)(C(F)(F)C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)C(F)(F)F	1.4
101	[N+]1(C)=CN(C=C1)CC.[O-]P(=O)(C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)F	2.832509
102	CCCCN1C=C[N+](=C1)C.[C-]#[O+].[C-]#[O+].[C-]#[O+].[C-]#[O+].[Co]	2.447158
103	[N+]1(C)=CN(C=C1)CCO.[I-]	4.6
104	[N+]1(C)=CN(C=C1)CCCO.[Cl-]	4.6
105	[N+]1(C)=CN(C=C1)COCC.[Cl-]	3.603144
106	[N+]1(C)=CN(C=C1)CCOCC.[Br-]	4.139879
107	[N+]1(C)=CN(C=C1)CCOC.[Cl-]	4.6
108	[N+]1(C)=CN(C=C1)CCCOC.[Cl-]	4.79
109	[N+]1(C)=CN(C=C1)CC#N.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.900367
110	[N+]1(C)=CN(C=C1)CCCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.659916
111	[N+]1(C)=CN(C=C1)COCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.201397
112	[N+]1(C)=CN(C=C1)CCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.25042
113	[N+]1(C)=CN(C=C1)CCOCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.181844
114	[N+]1(C)=CN(C=C1)CCCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.342423
115	[N+]1(C)=CN(C=C1)CCC.[Cl-]	4.6
116	[N+]1(C)=CN(C=C1)CCCC(=O)O.[Cl-]	3.78
117	[N+]1(C)=CN(C=C1)CCCCCCCC(=O)O.[Br-]	3.78
118	[N+]1(C)=CN(C=C1)CCCCO.[Cl-]	3.78
119	[N+]1(CC)=CN(C=C1)CCC.[Br-]	3.6
120	[B-](F)(F)(F)F.C[NH+]1C=CN=C1	3.09
121	[N+]1(C)=CN(C=C1)CCC.[P-](F)(F)(F)(F)(F)F	3.3
122	[N+]1(C)=CN(C=C1)CCCC.[N-](C(F)(F)F)C(F)(F)F	2.187521
123	[N+]1(C)=CN(C=C1)CCO.[B-](F)(F)(F)F	3.482874
124	[N+]1(CCCCCCCCCC)=C(C)N(C=C1)CCCCCCCCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	-0.12843
125	[N+]1(CC)=C(CCCCCCCC=CCCCCCCCC)N(CC1)CCO.[O-]S(=O)(=O)OCC	0.068186
126	[N+]1(C)=CN(C=C1)CC.[O-]C(=O)C	4.225309

127	[N+]1(C)=CN(C=C1)CCC(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)C(F)(F)F.[P-](F)(F)(F)(F)(F)F	1.93902
128	[N+]1(C)=CN(C=C1)CCc1ccccc1.[Cl-]	3.3
129	[N+]1(C)=CN(C=C1)CCC(=O)C.[Br-]	3.78
130	[N+]1(C)=CN(C=C1)CCCCCCCCO.[Br-]	2.357935
131	[N+]1(C)=CN(C=C1)Cc1ccc(cc1)C.[P-](F)(F)(F)(F)(F)F	2.663701
132	[N+]1(C)=CN(C=C1)CCCC.[O-]S(=O)(=O)c1ccc(C)cc1	3.290035
133	[N+]1(C)=CN(C=C1)Cc1ccccc1.[P-](F)(F)(F)(F)(F)F	3.3
134	[N+]1(C)=CN(C=C1)Cc1ccccc1.[Cl-]	3.3
135	[N+]1(C)=CN(C=C1)CCCCCCCC.[O-]S(=O)(=O)OCCCCCCCC	1.525045
136	[N+]1(C)=CN(C=C1)CC.[O-]S(=O)(=O)OCCOCCOC	3.845718
137	[N+]1(C)=CN(C=C1)CCCCCC.[C-](S(=O)(=O)C(F)(F)F)(S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.44
138	[N+]1(C)=CN(C=C1)CCCCCC.[N-]1C(=O)c2ccccc2S1(=O)=O	2.307496
139	[N+]1(C)(CCOCC1)CCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.18
140	[N+]1(C)(CCOCC1)COCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.36
141	[N+]1(C)(CCOCC1)CCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.43
142	CCOC[N+]1(CCOCC1)C.[Cl-]	3.52
143	N#C[N+]1(C)CCOCC1.FC(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F	3.53
144	[N+]1(C)(CCOCC1)CCCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.53
145	[N+]1(C)(CCOCC1)CCOCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.69
146	[N+]1(C)(CCOCC1)CCCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.77
147	[N+]1(C)(CCOCC1)CC.[O-]S(=O)(=O)c1ccc(C)cc1	3.81
148	[N+]1(C)(CCOCC1)CCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.81
149	[Br-].O1CC[N+](CCCC)(C)CC1	4.6
150	[N+]1(C)(CCOCC1)CCO.[I-]	4.6
151	[N+]1(C)(CCOCC1)CCOC.[Cl-]	4.6
152	[Br-].O(CC[N+]1(C)CCOCC1)CC	4.6
153	[Cl-].N#CC[N+]1(C)CCOCC1	4.6
154	[N+]1(C)(CCOCC1)CCCO.[Cl-]	4.6
155	[N+]1(C)(CCOCC1)CCCOC.[Cl-]	4.6
156	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]N(=O)=O	4
157	[N+]1(C)(CCOCC1)Cc1ccccc1.[B-](F)(F)(F)F	2.61
158	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]S(=O)(=O)O	3.64
159	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C=O	4.1
160	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)C	4.1
161	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)COC	3.9
162	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)COCCOC	3.89
163	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)CCCCC	3.63
164	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)CCCCCCC	3.22
165	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]S(=O)(=O)OCCCCCCCCCCCCC	2.4
166	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)C(CC)CC	4.13
167	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)C(C)O	4.3

168	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)/C=C/C	4.15
169	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)/C=CC(=O)O	3.35
170	[N+]1(C)(CCOCC1)Cc1ccccc1.[O-]C(=O)c1c(O)cccc1	2.9
171	[N+]1(C)(CCOCC1)Cc1ccccc1.[N-]1C(=O)c2ccccc2S1(=O)=O	3.5
172	[P+](CCCCCC)(CCCCCC)(CCCCCC)CCCCCCCCCCCCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	0.24
173	[P+](CCCCCC)(CCCCCC)(CCCCCC)CCCCCCCCCCCCCC.[B-](F)(F)(F)F	0.48
174	[P+](CCCC)(CCCC)(CCCC)CCCC.[B-]12(Oc3ccccc3O1)Oc3ccccc3O2	1.32
175	[P+](CCCC)(CCCC)(CCCC)CCCC.[Br-]	1.66
176	[O-]C=O.OCCC[NH3+]	4.490661
177	O=C([O-])CCC.OCCC[NH3+]	3.260071
178	O=C([O-])C.OCC[NH2+]CCO	4.032216
179	O=C([O-])CC.OCC[NH2+]CCO	3.767156
180	O=C([O-])CCC.OCC[NH2+]CCO	3.245513
181	O=C([O-])C(C)C.OCC[NH2+]CCO	3.474216
182	O=C([O-])CCCC.OCC[NH2+]CCO	3.724276
183	[O-]C(=O)CCC.OCC[NH+](CCO)CCO	3.394452
184	[O-]C(=O)CCCC.OCC[NH+](CCO)CCO	3.5302
185	[N+]1(C)(CCCCC1)CCCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.27
186	[N+]1(C)(CCCCC1)CCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.28
187	[N+]1(C)(CCCCC1)CCOCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.34
188	[N+]1(C)(CCCCC1)CCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.41
189	[N+]1(C)(CCCCC1)CCCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.63
190	[N+]1(C)(CCCCC1)CCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.65
191	[N+]1(C)(CCCCC1)CC#N.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	4
192	CCCC[N+]1(CCCCC1)C.[Br-]	4.03
193	CCOCC[N+]1(CCCCC1)C.[Br-]	4.31
194	C[N+]1(CCCCC1)CCCOC.[Cl-]	4.4
195	C[N+]1(CCCCC1)CCO.[I-]	4.58
196	C[N+]1(CCCCC1)CC#N.[Cl-]	4.58
197	C[N+]1(CCCCC1)CCOC.[Br-]	4.6
198	C[N+]1(CCCCC1)CCCO.[Cl-]	4.6
199	CCOC[N+]1(CCCCC1)C.[Cl-]	4.23
200	[N+]1(C)(CCCCC1)COCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.41
201	[N+]1(C)(CCCCC1)CCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.16
202	[n+]1(ccccc1)CCCS(=O)(=O)O.[O-]S(=O)(=O)C(F)(F)F	3.9
203	[n+]1(cc(C)ccc1)CCCC.[N-](C#N)C#N	3.46
204	[n+]1(ccccc1)CCOCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.26
205	[n+]1(ccccc1)CCOCC.[Br-]	4.24
206	[n+]1(ccccc1)CCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.79
207	[n+]1(ccccc1)CCO.[I-]	4.16
208	[n+]1(ccccc1)CCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.19
209	[n+]1(ccccc1)CCCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.55

210	[n+]1(ccccc1)CCCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.38
211	[n+]1(ccccc1)COCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.12
212	[n+]1(ccccc1)COCC.[Cl-]	3.32
213	[n+]1(c(C)cccc1)CCCC.[B-](F)(F)(F)F	3.25
214	[n+]1(cc(C)c(C)cc1)CCCC.[B-](F)(F)(F)F	3.02
215	[n+]1(cc(C)cc(C)c1)CCCC.[B-](F)(F)(F)F	3.26
216	[n+]1(cc(C)ccc1)CCCC.[B-](F)(F)(F)F	3.3
217	[n+]1(ccc(C)cc1)CCCC.[B-](F)(F)(F)F	2.98
218	[n+]1(ccccc1)CCCC.[B-](F)(F)(F)F	3.6
219	[n+]1(cc(O)ccc1)COCCC.[N-]1C(=O)C=C(C)OS1(=O)=O	3.5
220	[Cl-].NCCN1C=CCC=C1	3.79
221	[n+]1(ccc(C)cc1)CCCCCC.[B-](F)(F)(F)F	2.17
222	[n+]1(ccc(C)cc1)CCCCCCCC.[B-](F)(F)(F)F	1.49
223	[n+]1(ccc(C)cc1)CCCCCCCC.[Cl-]	1.63
224	[n+]1(ccccc1)CCCCCCCC.[Cl-]	1.27
225	[n+]1(ccc(N(C)C)cc1)CCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	1.82
226	[n+]1(ccc(N(C)C)cc1)CCCC.[Cl-]	1.94
227	[n+]1(ccc(N(C)C)cc1)CC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	2.79
228	[n+]1(ccc(N(C)C)cc1)CC.[Br-]	2.9
229	[n+]1(ccc(N(C)C)cc1)CCCCCC.[Cl-]	0.93
230	[n+]1(ccc(N(C)C)cc1)CCCCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	0.93
231	[n+]1(ccccc1)CCCO.[Cl-]	4.6
232	[n+]1(ccccc1)CCOC.[Cl-]	4.6
233	[n+]1(ccccc1)CCCOC.[Cl-]	4.6
234	[n+]1(cc(O)ccc1)COCCCCCC.[N-]1C(=O)c2ccccc2S1(=O)=O	3.3
235	[n+]1(ccccc1)CCCC.[Br-]	3.9
236	[n+]1(cc(C)c(C)cc1)CCCC.[Cl-]	3.05
237	[n+]1(cc(C)cc(C)c1)CCCC.[Cl-]	3.42
238	[n+]1(cc(C)ccc1)CCCCCC.[Cl-]	2.4
239	[n+]1(cc(C)ccc1)CCCC.[Cl-]	3.86
240	[n+]1(ccc(C)cc1)CCCCCC.[Cl-]	2.67
241	[n+]1(ccc(C)cc1)CCCC.[Cl-]	3.32
242	[n+]1(ccc(C)cc1)CCCC.[B-](C#N)(C#N)(C#N)C#N	2.88
243	[n+]1(ccccc1)CCCCC.[Br-]	3.15
244	[n+]1(ccccc1)CCCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	2.85
245	[n+]1(ccccc1)CCCC.[O-]S(=O)(=O)OC	3.92
246	[n+]1(ccccc1)CCCC.[P-](F)(F)(F)(F)(F)F	3.85
247	[n+]1(ccccc1)CCCC.[O-]S(=O)(=O)C(F)(F)F	3.66
248	[n+]1(ccccc1)CCCCCC.[Cl-]	2.8
249	[n+]1(ccccc1)CCCCCC.[P-](F)(F)(F)(F)(F)F	2.51
250	[n+]1(ccccc1)CCCCCC.[O-]S(=O)(=O)C(F)(F)F	2.54
251	[n+]1(ccccc1)CCCCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	2.87
252	[n+]1(ccccc1)CC.[Cl-]	4.22

253	[n+]1(ccccc1)CCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.2
254	[n+]1(cc(O)ccc1)COCCC.[Cl-]	2.9
255	[n+]1(cc(O)ccc1)COCCCCCCC.[Cl-]	3
256	[n+]1(cc(O)ccc1)COCCCCCCCCCCC.[Cl-]	1.6
257	NCCN1C=CCC=C1.FC(F)(F)S(=O)(=O)NS(=O)(=O)C(F)(F)F	3.5
258	[n+]1(cc(O)ccc1)COCCCC.[N-]1C(=O)C=C(C)OS1(=O)=O	3.7
259	[n+]1(cc(O)ccc1)COCCCCCC.[N-]1C(=O)C=C(C)OS1(=O)=O	3.4
260	[n+]1(cc(O)ccc1)COCCCCCCC.[N-]1C(=O)C=C(C)OS1(=O)=O	3
261	[n+]1(cc(O)ccc1)COCCCCCCCCCCC.[N-]1C(=O)C=C(C)OS1(=O)=O	1.6
262	[n+]1(cc(O)ccc1)COCCC.[N-]1C(=O)c2ccccc2S1(=O)=O	3.6
263	[n+]1(cc(O)ccc1)COCCCC.[N-]1C(=O)c2ccccc2S1(=O)=O	3.6
264	[n+]1(ccc(C)cc1)CCCC.[P-](F)(F)(F)(F)(F)F	3.3
265	[n+]1(cc(O)ccc1)COCCCCCCC.[N-]1C(=O)c2ccccc2S1(=O)=O	3
266	[n+]1(cc(O)ccc1)COCCCCCCCCCCC.[N-]1C(=O)c2ccccc2S1(=O)=O	1.6
267	[n+]1(cc(C)ccc1)CCCCCCCC.[Cl-]	1.47
268	[n+]1(ccc(N(C)C)cc1)C.[I-]	2.89
269	[N+]1(C)(CCCC1)CCCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.01
270	[N+]1(C)(CCCC1)CCOCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.2
271	[N+]1(C)(CCCC1)CCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.72
272	[N+]1(C)(CCCC1)CCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.3
273	[N+]1(C)(CCCC1)CC#N.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.8
274	[N+]1(C)(CCCC1)CCCO.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.6
275	[N+]1(C)(CCCC1)CCCOC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.4
276	[N+]1(C)(CCCC1)COCC.[Cl-]	3.05
277	[N+]1(C)(CCCC1)COCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.26
278	[N+]1(CCCCCC)(CCCC1)CCCCCC.[B-](F)(F)(F)F	1.23
279	[N+]1(C)(CCCC1)CCCC.[B-](F)(F)(F)F	2.9
280	[N+]1(C)(CCCC1)CCCC.[Br-]	3.77
281	[N+]1(C)(CCCC1)CCCC.[N-](C#N)C#N	4.23
282	[N+]1(C)(CCCC1)CCCCCC.[Cl-]	2.91
283	[N+]1(C)(CCCC1)CCCCCCCC.[B-](F)(F)(F)F	1.82
284	[N+]1(C)(CCCC1)CCCCCCCC.[Cl-]	2.59
285	[N+]1(C)(CCCC1)CCO.[I-]	4.6
286	[N+]1(C)(CCCC1)CCCC.[Cl-]	4.6
287	[N+]1(C)(CCCC1)CCCO.[Cl-]	4.6
288	[N+]1(C)(CCCC1)CCOC.[Cl-]	4.6
289	[N+]1(C)(CCCC1)CCOCC.[Br-]	4.6
290	[N+]1(C)(CCCC1)CCCOC.[Cl-]	4.6
291	[N+]1(C)(CCCC1)CCCC.[P-](F)(F)(F)(C(F)(F)C(F)(F)F)(C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)F	2.41
292	[N+]1(C)(CCCC1)CC#N.[Cl-]	4.23
293	[N+]1(C)(CCCC1)CC(=O)OCC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.592177
294	[N+]1(C)(CCCC1)CC(=O)OCC.[Br-]	3.844477

295	C[N+]1(CCCC1)CCOC.C(F)(F)(F)C(F)(F)[P-](F)(F)(F)(C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)F	2.16
296	C[N+]1(CCCC1)CCOC(=O)C.C(F)(F)(F)C(F)(F)[P-](F)(F)(F)(C(F)(F)C(F)(F)F)C(F)(F)C(F)(F)F	2.91
297	F[B-](F)(F)F.c12ccccc1ccc[n+]2CCCCCCCC	0.17
298	F[B-](F)(F)F.c12ccccc1ccc[n+]2CCCC	2.16
299	[Br-].c12ccccc1ccc[n+]2CCCC	2.32
300	[n+]1(cccc2ccccc12)CCCCCC.[B-](F)(F)(F)F	1.07
301	[Br-].c12ccccc1ccc[n+]2CCCCCCCC	-0.03
302	[Br-].[S+](CC)(CC)CC	3.4
303	[S+](C)(CC)CC.[N-](S(=O)(=O)C(F)(F)F)S(=O)(=O)C(F)(F)F	3.48
304	CCCCCCCCN1C=C[N+](=C1)C.[Cl-]	2
305	CCCCN1C=C[N+](=C1)C.[Cl-]	3.55
306	CCCCCCCCCCCCCCCCN1C=C[N+](=C1)C.[Cl-]	-0.24
307	CCCCCCC[NH+]1CN(C=C1)C.[Cl-]	2.53
308	CCCCN1C=C[N+](=C1)C.[Br-]	3.43
309	CCCCN1C=C[N+](=C1)C.[I-]	3.48
310	[B-](F)(F)(F)F.CCN1C=C[N+](=C1)C	3.44
311	[B-](F)(F)(F)F.CCCCCCCCN1C=C[N+](=C1)C	1.59
312	[B-](F)(F)(F)F.CCCN1C=C[N+](=C1)C	3.45
313	CC1=CC=C(C=C1)CN2C[NH+](C=C2)C.[Cl-]	2.64
314	[B-](F)(F)(F)F.CCCCCCN1C=C[N+](=C1C)C	1.9
315	[B-](F)(F)(F)F.C[N+]1=CN(C=C1)CC2=CC=CC=C2	2.97
316	CCN1C=C[N+](=C1)C.CC(=O)[O-]	4.23
317	CCCCN1C=C[N+](=C1)C.COCCOCCOS(=O)(=O)[O-]	3.15
318	CCN1C=C[N+](=C1)C.C(F)(F)(F)S(=O)(=O)[O-]	4.09
319	CCCCN1C=C[N+](=C1)C.C(F)(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F	2.68
320	CCCCCCCCN1C=C[N+](=C1)C.C(F)(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F	1.64
321	CCN1C=C[N+](=C1)C.C(=O)(C(F)(F)F)[O-]	4
322	CCN1C=C[N+](=C1)C.CS(=O)(=O)[O-]	3.97
323	[B-](F)(F)(F)F.CCCC[N+]1=CC=CC=C1	3.16
324	CCCC[N+]1=CC=C(C=C1)C.[Cl-]	3.32
325	CCCCCCCC[N+]1(CCCC1)C.[Cl-]	2.59
326	CCCC[N+]1=CC=CC=C1.[Br-]	3.9
327	[B-](F)(F)(F)F.CCCC[N+]1=CC=CC(=C1)C	3.3
328	CCCC[N+]1=CC=CC=C1.C(F)(F)(F)S(=O)(=O)[O-]	3.66
329	CCCCCC[N+]1=CC=CC=C1.C(F)(F)(F)S(=O)(=O)[O-]	2.54
330	CCCC[N+]1=CC=CC=C1.[Cl-]	3.77
331	CCCCCC[N+]1=CC=CC(=C1)C.[Cl-]	2.4
332	CCCCCC[N+]1=CC=C(C=C1)C.[Cl-]	2.67
333	[B-](F)(F)(F)F.CCCCCC[N+]1=CC=C(C=C1)C	2.17
334	CCOC[N+]1=CC=CC=C1.[Cl-]	3.32
335	[B-](F)(F)(F)F.CCCC[N+]1=CC=CC=C1C	3.25

336	CCC[N+]1=CC=CC=C1.C(F)(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F	3.2
337	CCCCC[N+]1=CC=CC=C1.[Br-]	3.15
338	CCCC[N+]1=CC=CC=C1C.[Cl-]	3.75
339	CCCCCCCCCCCCCC[N+](C)(C)CC1=CC=CC=C1.[Cl-]	0.16
340	CCC[N+]1(CCCCC1)C.C(F)(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F	3.16
341	C[N+]1(CCCCC1)CCOC.C(F)(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F	3.25
342	CCCC[N+]1(CCCCC1)C.C(F)(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F	3.41
343	CCCCCC[N+]1=CC=CC=C1.[Cl-]	2.8
344	C1=CC=[N+](C=C1)CC#N.[Cl-]	3.79
345	CCCCCCCCCCN1C=C[N+](=C1)C.[Cl-]	1.34
346	CCCCCCN1C=C[N+](=C1)C.[Cl-]	2.82
347	CCN1C=C[N+](=C1)C.[Cl-]	3.86
348	[B-](F)(F)(F)F.CCCC[N+]1=CN(C=C1)CC	3.26
349	CCCCN1C=C[N+](=C1)C.C(F)(F)(F)S(=O)(=O)[O-]	3
350	CCCCCCN1C=C[N+](=C1)C.C(F)(F)(F)S(=O)(=O)[N-]S(=O)(=O)C(F)(F)F	2.24
351	CCN1C=C[N+](=C1)C.COS(=O)(=O)[O-]	4.2
352	CCCCCCCC[N+]1=CC=C(C=C1)C.[Cl-]	2.97
353	CC[N+]1=CC=CC=C1.[Cl-]	4.22
354	CCCCCCCCCC[N+](C)(C)CC1=CC=CC=C1.[Cl-]	0.65
355	[B-](F)(F)(F)F.CCCCN1C=C[N+](=C1)C	3.11

Tabla 3.1 Moléculas y su grado de toxicidad experimental.

CAPÍTULO 4. RESULTADOS

4.1 PREDICCIÓN DE TOXICIDAD MEDIANTE XGBOOST Y KERNEL RIDGE REGRESSION

En esta predicción utilizando los descriptores SOAP con regresion de Kernel los datos son los siguientes:

Conjunto de prueba es de 0.8212
Conjunto de entrenamiento es de 0.9417

Donde podemos observar que la predicción de los datos tiene un error mínimo de acuerdo con los errores cuadráticos medios antes mencionados, donde los valores nos dan una alta precisión de la toxicidad de las sustancias que se utilizaron.

A continuación, se muestra el código computacional y las gráficas de regresión obtenidas.

In [3]:
```
        #!python --version
import sklearn

import dscribe
import os
import glob
import gc
import time, copy
import ase

import ase.visualize
import deepchem as dc
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import tensorflow as tf

from ase import atoms
from ase import io
from ase.io import read
from ase.build import molecule
from dscribe.descriptors import ACSF
from dscribe.descriptors import ACSF, LMBTR, SOAP
from numpy.linalg import eig
from numpy.linalg import svd
from numpy.random import randint
from sklearn.decomposition import PCA
from tqdm import tqdm_notebook, tqdm

#from sklearn.linear_model import Ridge
from sklearn.kernel_ridge             import KernelRidge
from sklearn.model_selection          import GridSearchCV
from sklearn.ensemble                 import RandomForestRegressor
from sklearn.gaussian_process         import GaussianProcessRegressor
from sklearn.gaussian_process.kernels import WhiteKernel, ExpSineSquared
from sklearn import preprocessing
```

In [4]:
```
# estableciendo el directorio en donde se toman los archivos fuente en formato xyz
#os.chdir(r"/home/arnulfo/00PracticaCodigosEntVir/ACSFDescriptorPredictingMullikenChargers/all2210")
import os
os.chdir(r"/home/aurora/Rika/toxicity/arcXYZ")

# creando una lista con los nombres ordenados mediante la librería glob
j=0
filenames = [i for i in sorted(glob.glob("pd*.xyz"))]

# Determinando la cantidad de archivos
for i in filenames:
    j+=1
    #print(i, end='  \n')
print('total =', j)
```

```
total = 355
```

In [23]:
```
# funcion que obtiene sumatoria de gausiannas para generar la variable XX
def calRegMol(filename):
    soap = SOAP(
    #  species tomados del càlculo de las 2210 , y estàn ordenadas de acuerdo al peso especìfico de la tabla
    #  periòdica de elementos
    species= ['H','B','C','N','O','F','P','S','Cl','Co', 'Br', 'I'],

    rcut=12.0,
    nmax=6,
    lmax=6,
    average="inner",
    sparse=False
    )

    mm = io.read(filename)
    pD = soap.create(mm)
    arPD = np.array(pD)

    return arPD
```

In [26]:
```
#Generando lista de gausianas por moléculas
cal_X=[]
#print(filenames)
#calRegMol('pd008.xyz')
for ar in filenames:
#    print(ar)
    calRegMol(ar)
    cal_X.append(calRegMol(ar))

df= pd.DataFrame(cal_X)
```

In [27]:
```
df
```

Out[27]:

	0	1	2	3	4	5	6	7	8	9	...	18386	18387	18388	18389	1839
0	0.030078	0.514672	1.266474	2.186144	2.986683	3.449775	8.806717	21.671049	37.407819	51.106096	...	0.0	0.0	0.0	0.0	0.
1	0.011973	0.209611	0.481845	0.821846	1.372256	2.035100	3.669537	8.435374	14.387562	24.023266	...	0.0	0.0	0.0	0.0	0.
2	0.010372	0.232184	0.457004	0.838768	1.500297	2.581028	5.197743	10.230643	18.776927	33.586139	...	0.0	0.0	0.0	0.0	0.
3	0.027302	0.407571	1.071292	2.139695	2.413375	4.218538	6.084280	15.992417	31.941709	36.027243	...	0.0	0.0	0.0	0.0	0.
4	0.012709	0.201862	0.478707	0.854889	1.223719	1.992620	3.206274	7.603537	13.578625	19.436945	...	0.0	0.0	0.0	0.0	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
350	0.021004	0.365194	0.771842	1.043800	1.661941	2.203435	6.349554	13.419882	18.148366	28.895865	...	0.0	0.0	0.0	0.0	0.
351	0.031938	0.462357	1.220768	2.089560	2.269158	3.619939	6.693349	17.672557	30.249691	32.849657	...	0.0	0.0	0.0	0.0	0.
352	0.018938	0.362774	0.761787	1.120159	1.531295	1.570797	6.949380	14.592970	21.458022	29.333828	...	0.0	0.0	0.0	0.0	0.
353	0.026451	0.394795	1.008681	2.046118	2.477822	4.531766	5.892546	15.055171	30.539534	36.982964	...	0.0	0.0	0.0	0.0	0.
354	0.024616	0.343248	0.821007	1.298584	1.891720	2.492105	4.786335	11.448338	18.107794	26.378639	...	0.0	0.0	0.0	0.0	0.

355 rows × 18396 columns

In [29]:
```
import os

os.chdir(r"/home/aurora/Rika/toxicity/")
!pwd
```

```
/home/aurora/Rika/toxicity
```

In [30]:
```
df.to_csv('soap_dataframe.csv', index= False)
```

Extrayendo el valor a calcular

In [31]:
```
# leyendo datos del archivo de similes y temperaturas de fusión
cal_y= pd.read_csv('datoToxicity20.csv', delimiter=',')

# eliminando las columnas que no son necesarias
cal_y= cal_y.drop(['SMILES'], axis=1)
cal_y= cal_y.drop(['Id'], axis=1)
#cal_y= cal_y.drop(['Nombre '], axis=1)
#print(cal_y)
```

Normalizando los Dataframe

In [32]:
```
# normalizando la variable a calcular  "cal_y"
min_max_scaler = preprocessing.MinMaxScaler()
X_dfNor = pd.DataFrame(min_max_scaler.fit_transform(df))
#print(X_dfNor)
```

In [33]:
```
#normalizando las temperaturas
y_d = cal_y
min_max = preprocessing.MinMaxScaler()
y_dfNor= min_max.fit_transform(y_d)

yNorm = pd.DataFrame(y_dfNor)
# nor
```

Asignando los valores de y

In [34]:
```
# Asignando la columna de temperatura de fusioń al dataframe
# la finalidad es que cuando se separen 70 30 vaya cada dato con su temp en un solo dataframe
# todos los datos están normalizados
dfNor = X_dfNor.assign(Toxicity= yNorm)
dfNor
```

Out[34]:

	0	1	2	3	4	5	6	7	8	9	...	18387	18388	18389	18390	18391
0	0.495035	0.766893	0.671561	0.749147	0.740448	0.384244	0.804928	0.861928	0.912957	0.766938	...	1.0	0.0	1.0	0.0	1.0
1	0.166550	0.248528	0.206898	0.229765	0.293047	0.197217	0.274356	0.278460	0.295894	0.316099	...	1.0	0.0	1.0	0.0	1.0
2	0.137488	0.286884	0.192187	0.236207	0.328531	0.269392	0.432190	0.357601	0.413552	0.475289	...	1.0	0.0	1.0	0.0	1.0
3	0.444673	0.584905	0.555973	0.731464	0.581569	0.485878	0.523752	0.611597	0.766437	0.515925	...	1.0	0.0	1.0	0.0	1.0
4	0.179895	0.235361	0.205039	0.242344	0.251884	0.191601	0.226510	0.241790	0.274211	0.239752	...	1.0	0.0	1.0	0.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
350	0.330401	0.512897	0.378636	0.314262	0.373327	0.219472	0.551150	0.498192	0.396703	0.397212	...	1.0	0.0	1.0	0.0	1.0
351	0.528790	0.677998	0.644494	0.712377	0.541603	0.406740	0.586657	0.685663	0.721082	0.463029	...	1.0	0.0	1.0	0.0	1.0
352	0.292909	0.508785	0.372682	0.343331	0.337121	0.135835	0.613101	0.549905	0.485419	0.404502	...	1.0	0.0	1.0	0.0	1.0
353	0.429227	0.563195	0.518894	0.695839	0.599429	0.527288	0.503950	0.570281	0.728851	0.531835	...	1.0	0.0	1.0	0.0	1.0
354	0.395930	0.475606	0.407752	0.411257	0.437005	0.257636	0.389700	0.411281	0.395616	0.355308	...	1.0	0.0	1.0	0.0	1.0

355 rows × 18397 columns

Preparando los datos de entrenamiento para comparación del modelo

Dividiendo los datos entrenamiento y prueba

In [35]:
```
from sklearn.model_selection import train_test_split

train, test = train_test_split(dfNor, test_size=0.2)
```

In [36]:
```
# extrayendo la variable toxicidad del dataframe de entrenamiento
# para dejar solo la variable X que alimentará al modelo
y_train = train['Toxicity']
X_train = train.drop(['Toxicity'], axis=1)
#train_X
y_test = test['Toxicity']
X_test = test.drop(['Toxicity'], axis =1)
#test_X
```

In [ ]:
```
```

Generando el Modelo XGBoost con parámetros

In [37]:
```
# create an xgboost regression model
from xgboost import XGBRegressor
mod_ACSF_XGBoost = XGBRegressor(n_estimators=500,
                     max_depth=7,
                     eta=0.1,
                     subsample=0.7,
                     colsample_bytree=0.8
                    )
```

In [38]:
```
mod_ACSF_XGBoost.fit(X_train, y_train)
```

Out[38]:

```
XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
             colsample_bylevel=1, colsample_bynode=1, colsample_bytree=0.8,
             early_stopping_rounds=None, enable_categorical=False, eta=0.1,
             eval_metric=None, feature_types=None, gamma=0, gpu_id=-1,
             grow_policy='depthwise', importance_type=None,
             interaction_constraints='', learning_rate=0.100000001, max_bin=256,
             max_cat_threshold=64, max_cat_to_onehot=4, max_delta_step=0,
             max_depth=7, max_leaves=0, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=500, n_jobs=0,
             num_parallel_tree=1, predictor='auto', ...)
```

Evaluando métricas del modelo de regresión

In [39]:
```
y_pred       = mod_ACSF_XGBoost.predict(X_test)
y_pred_train = mod_ACSF_XGBoost.predict(X_train)
#y_pred_train
```

In [57]:
```
# Calculando el mean cuadratic error
from sklearn.metrics import r2_score

print("Error r2 y_test" , r2_score(y_test, y_pred))
print("Error r2 y_train" , r2_score(y_train, y_pred_train))
```

```
Error r2 y_test 0.784326500086026
Error r2 y_train 0.9999982563638234
```

In [41]:
```
# Calculando el Medio absolute error
from sklearn.metrics import mean_absolute_error

print("Mean Absolute Error y_test" , mean_absolute_error( y_test , y_pred))
print("Mean Absolute Error y_train" , mean_absolute_error( y_train, y_pred_train))
```

```
Mean Absolute Error y_test 0.08021886759961652
Mean Absolute Error y_train 0.00021009134587546564
```

In [42]:
```
# Calculando el mean cuadratic error
from sklearn.metrics import mean_squared_error

print("Mean squared Error y_test" ,mean_squared_error( y_test , y_pred))
print("Mean squared Error y_train" ,mean_squared_error( y_train, y_pred_train))
```

```
Mean squared Error y_test 0.011004012103681909
Mean squared Error y_train 7.098520688752963e-08
```

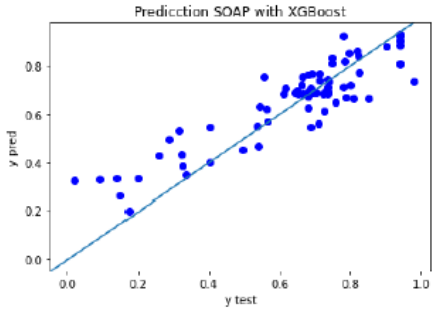
Graficando los resultados

In [43]:
```
print("train X", X_train.shape)
print("train y", y_train.shape)
print("test X", X_test.shape)
print("test y", y_test.shape)
print("pred y", y_pred.shape)
```

```
train X (284, 18396)
train y (284,)
test X (71, 18396)
test y (71,)
pred y (71,)
```

In [46]:
```
# plot impresión

import matplotlib.pyplot as plt
fig, ax = plt.subplots()
plt.scatter(y_test, y_pred, color= "blue")
#plt.scatter(X_test, y_test, color= "black")              en el ejemplo de abajo funciona aunque no son del mismo t
#plt.plot(y_test, y_pred, color= "blue", linewidth=3)     si se activa crea una line en cada uno de los puntos
plt.xticks()
plt.yticks()
plt.xlabel("y test")
plt.ylabel("y pred")
ax.axline((0,0), slope=1 )
plt.title("Predicction SOAP with XGBoost")
plt.show()
```



In [ ]:

Guardando el modelo matematico

In [47]:
```
from sklearn import model_selection, datasets
from sklearn.tree import DecisionTreeClassifier
import joblib
import pickle
```

In [48]:
```
mod = mod_ACSF_XGBoost
```

In [49]:
```
fil_Y_train = pd.DataFrame(y_train)
fil_Y_train.to_csv("y_train.csv", index=False)

fil_X_train = pd.DataFrame(X_train)
fil_X_train.to_csv("X_train.csv", index=False)

fil_Y_test = pd.DataFrame(y_test)
fil_Y_test.to_csv("y_test.csv", index=False)

fil_X_test = pd.DataFrame(X_test)
fil_X_test.to_csv("X_test.csv", index=False)

fil_X_test = pd.DataFrame(y_pred)
fil_X_test.to_csv("y_pred.csv", index=False)

fil_X_test = pd.DataFrame(y_pred_train)
fil_X_test.to_csv("y_pred_train.csv", index=False)

filename= "respaldo.joblib"
joblib.dump(mod, filename)
```

Out[49]:
```
['respaldo.joblib']
```

Calculo con KRR

In [50]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVR
from sklearn.kernel_ridge import KernelRidge
```

In [53]:
```
kr = GridSearchCV(KernelRidge(), param_grid={"alpha": [10, 1e0, 0.1, 5e-2, 1e-2, 5e-3, 1e-3], "kernel": ['rbf'],
                            "gamma": np.logspace(-4, 5, 10)}, cv=5)
```

In [54]:
```
# entrenamiento del metodo KRR

kr.fit(X_train,y_train)

print(f"Best KRR with params: {kr.best_params_} and R2 score: {kr.best_score_:.3f}")
#print("KRR complexity and bandwidth selected and model fitted in %.3f s" % kr_fit)
print("training", kr.score(X_train,y_train))
```

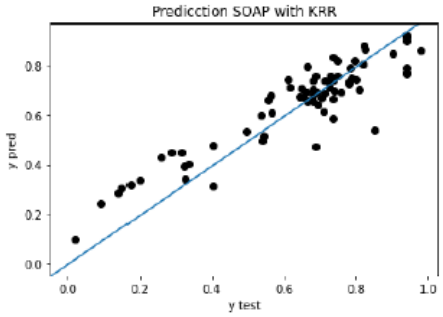
```
Best KRR with params: {'alpha': 0.005, 'gamma': 0.0001, 'kernel': 'rbf'} and R2 score: 0.710
training 0.9417406693349948
```

In [60]:
```
# Calculo del Coeficiente R2 para el conjunto de prueba
y_pred = kr.predict(X_test)
r2=kr.score(X_test, y_test)
print("test", r2)
```

```
test 0.8212602385511006
```

In [61]:
```
# Grafica de resultados

import matplotlib.pyplot as plt
fig, ax = plt.subplots()
plt.scatter(y_test, y_pred, color= "black")
#plt.scatter(X_test, y_test, color= "black")              en el ejemplo de abajo funciona aunque no son del mismo t
#plt.plot(y_test, y_pred, color= "blue", linewidth=3)     si se activa crea una line en cada uno de los puntos
plt.xticks()
plt.yticks()
plt.xlabel("y test")
plt.ylabel("y pred")
ax.axline((0,0), slope=1 )
plt.title("Predicction SOAP with KRR")
plt.show()
```



In [ ]:

CAPÍTULO 5. CONCLUSIONES

A través de los modelos de Machine Learning podemos predecir las propiedades fisicoquímicas de las sustancias partir de su estructura molecular tridimensional, por ejemplo, en este trabajo, se predijo el nivel de toxicidad que tienen las sustancias seleccionadas basadas en líquidos iónicos y con que precisión nos estamos acercando a los valores experimentales. Esto nos permite el ahorro de recursos tanto tiempo, capital humano, recursos monetarios, entre otros, siendo más eficiente y amigable con el medio ambiente, así como eliminando la necesidad de emplear técnicas experimentales in vivo en animales.

Actualmente los líquidos iónicos son considerado como solventes verdes, con un gran número de aplicaciones industriales, que pueden ser sintetizadas al combinar cationes orgánicos con diferentes aniones dando lugar a un número infinito de posibles combinaciones, las cuales deben evaluarse de forma in silico de forma eficiente antes de proceder a su uso industrial, ya que se ha encontrado que pueden tener cierto grado de toxicidad en los distintos ecosistemas, por lo cual las técnicas de aprendizaje maquina a la par con descriptores moleculares ofrecen una alternativa confiable para obtener el grado de toxicidad en las diferentes líneas celulares.

Bibliografía

Bartók, A. P., De, S., Poelking, C., Bernstein, N., Kermode, J. R., Csänyi, G., & Ceriotti, M. (2017). Machine learning unifies the modeling of materials and molecules. Science Advances, 3(12). https://doi.org/10.1126/sciadv.1701816

Bartók, A. P., Kondor, R., & Csänyi, G. (2013). On representing chemical environments. Physical Review. B, Condensed Matter And Materials Physics, 87(18). https://doi.org/10.1103/physrevb.87.184115

Cavieres, F., & Fernanda, M. (2004). Exposición a pesticidas y toxicidad reproductiva y del desarrollo en humanos: Análisis de la evidencia epidemiológica y experimental. Revista MéDica de Chile, 132(7). https://doi.org/10.4067/s0034-98872004000700014

Challenger-Pérez, I., Díaz-Ricardo, Y., & Becerra-García, R. A. (2014). El lenguaje de programación Python. Ciencias Holguín, XX(2), 1-13. https://www.redalyc.org/articulo.oa?id=181531232001

Brownlee, J. (2021). Xgboost for regression. MachineLearningMastery.com. https://machinelearningmastery.com/xgboost-for-regression/

Chen, T., & Guestrin, C. (2016). XGBoost. Asociation For Cumputing Machinery. KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Pages 785-794. https://doi.org/10.1145/2939672.2939785

Código de Barras, la enciclopédia libre. (n.d.). Wikipedia, la enciclopédia libre. https://es.wikipedia.org/wiki/C%C3%B3digo_de_barras#:~:text=El%20c%C3%B3digo%20de%20barras%20es,representan%20peque%C3%B1as%20cadenas%20de%20caracteres.

Deitel, P. (2021). Intro to Python for Computer Science and Data Science: Learning to Program with AI, Big Data and The Cloud, Global Edition. Pearson Higher Ed.

Egorova, K. S., Gordeev, E. G., & Ananikov, V. P. (2017). Biological Activity of Ionic Liquids and Their Application in Pharmaceutics and Medicine. Chemical Reviews, 117(10), 7132-7189. https://doi.org/10.1021/acs.chemrev.6b00562

Fullmetalfelix/ML-CSC-tutorial. (n.d.). GitHub. https://github.com/fullmetalfelix/ML-CSC-tutorial/blob/master/

García-Montalvo, I. A., Martínez-Gutiérrez, C., Sánchez-Medina, M. A., Pérez-Santiago, A. D., & De Jesús Ramírez Altamirano, M. (2019). Neurodegenerative conditions: toxicity by accumulation of metals. DOAJ

(DOAJ: Directory Of Open Access Journals). https://doi.org/10.19230/jonnpr.2944

Guidelines on the prevention of toxic exposures. (2004, April 13). World Health Organization (WHO). https://www.who.int/publications/i/item/9241546115

Himanen, L., Jäger, M. O. J., Morooka, E. V., Canova, F. F., Ranawat, Y. S., Gao, D. Z., Rinke, P., & Foster, A. S. (2020). DScribe: Library of descriptors for machine learning in materials science. Computer Physics Communications, 247, 106949. https://doi.org/10.1016/j.cpc.2019.106949

Karim, A., Mishra, A., Newton, M. A. H., & Sattar, A. (2019). Efficient Toxicity Prediction via Simple Features Using Shallow Neural Networks and Decision Trees. ACS Omega, 4(1), 1874-1888. https://doi.org/10.1021/acsomega.8b03173

Kaur, G., Kumar, H., & Singla, M. (2022). Diverse applications of ionic liquids: A comprehensive review. Journal Of Molecular Liquids, 351, 118556. https://doi.org/10.1016/j.molliq.2022.118556

Kim, P. (2017). MATLAB Deep learning. En Apress eBooks. https://doi.org/10.1007/978-1-4842-2845-6

Langer, M., Goeßmann, A., & Rupp, M. (2022). Representations of molecules and materials for interpolation of quantum-mechanical simulations via machine learning. Npj Computational Materials, 8(1). https://doi.org/10.1038/s41524-022-00721-x

MacFarlane, D. R., Tachikawa, N., Forsyth, M., Pringle, J. M., Howlett, P. C., Elliott, G. D., Davis, J. H., Watanabe, M., Simon, P., & Angell, C. A. (2014). Energy applications of ionic liquids. Energy & Environmental Science, 7(1), 232-250. https://doi.org/10.1039/c3ee42099j

Módulo I - Introducción a la toxicología | Notas para la instrucción | ATSDR en Español. (2022, August 8). Agency for Toxic Substances and Disease Registry. https://www.atsdr.cdc.gov/es/training/toxicology_curriculum/modules/1/es_lecturenotes.html

Mora-Barrantes, J. C., Ramos, L. M., Ulate-Salas, M., Núñez-Agüero, V., Acuña-Salazar, E., & Cordero-Carvajal, M. (2021). Clasificación del riesgo químico de solventes orgánicos mediante la aplicación del método “CHEM21 selection guide of classical- and less classical-solvents”. Un caso de estudio en cursos de docencia universitaria de química orgánica y bioquímica. Tecnología En Marcha. 35(1), Pág. 28–43. https://doi.org/10.18845/tm.v35i1.5370

O'Boyle, N. M., Banck, M., James, C. A., Morley, C., Vandermeersch, T., & Hutchison, G. R. (2011). Open Babel: An open chemical toolbox. Journal of Cheminformatics, 3(1). https://doi.org/10.1186/1758-2946-3-33

Organic solvent neurotoxicity. (2020, September 29). Centers for Disease Control and Prevention | CDC. https://www.cdc.gov/niosh/docs/87-104/default.html

Pedregosa, F., Varuakuaux, G., Gramfort, A., Michel, V., & Thirion, B. (2007). SciKit-Learn: Machine Learning in Python. Journal Of Machine Learning Research, 12, 2825-2830.

Rodrigues, J. F., Florea, L., De Oliveira, M. C. F., Diamond, D., & Oliveira, O. N. (2021). Big data and machine learning for materials science. Discov Mater 1, 12 (https://doi.org/10.1007/s43939-021-00012-0

Roper, C., & Tanguay, R. L. (2020). Tox21 and adverse outcome pathways. En Elsevier eBooks (pp. 559-568). https://doi.org/10.1016/b978-0-12-813602-7.00040-5

Smooth Overlap of Atomic Positions — DScribe 0.3.x documentation. (s. f.). https://singroup.github.io/dscribe/0.3.x/tutorials/soap.html#:~:text=Smooth%20Overlap%20of%20Atomic%20Positions%20(SOAP)%20is%20a%20descriptor%20that,harmonics%20and%20radial%20basis%20functions

Sosnowska, A., Grzonkowska, M., & Puzyn, T. (2017). Global versus local QSAR models for predicting ionic liquids toxicity against IPC-81 leukemia rat cell line: The predictive ability. Journal of Molecular Liquids, 231, 333-340. https://doi.org/10.1016/j.molliq.2017.02.025

Tang, W., Chen, J., Wang, Z., Xie, H., & Hong, H. (2018). Deep learning for predicting toxicity of chemicals: a mini review. Journal Of Environmental Science And Health. Part C, Environmental Carcinogenesis & Ecotoxicology Reviews./Journal Of Environmental Science And Health. Part C, Environmental Carcinogenesis & Ecotoxicology Reviews, 36(4), 252-271. https://doi.org/10.1080/10590501.2018.1537563

Vovk, V. (2013). Kernel ridge regression. En Springer eBooks (pp. 105-116). https://doi.org/10.1007/978-3-642-41136-6_11

Wang, Z., Song, Z., & Zhou, T. (2020). Machine Learning for Ionic Liquid Toxicity Prediction. Processes, 9(1), 65. https://doi.org/10.3390/pr9010065

Weininger, D. (1988). SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. Journal of Chemical Information and Computer Sciences, 28(1), 31–36. https://doi.org/10.1021/ci00057a005

Willatt, M. J., Musil, F., & Ceriotti, M. (2019). Atom-density representations for machine learning. Journal Of Chemical Physics150(15). https://doi.org/10.1063/1.5090481

Winterton, N. (2021). The green solvent: a critical perspective. Clean Technologies And Environmental Policy, 23(9), 2499-2522. https://doi.org/10.1007/s10098-021-02188-8

Wu, T., Li, W., Chen, M., Zhou, Y., & Zhang, Q. (2020). Estimation of Ionic Liquids Toxicity against Leukemia Rat Cell Line IPC-81 based on the Empirical-like Models using Intuitive and Explainable Fingerprint Descriptors. Molecular Informatics, 39(10). https://doi.org/10.1002/minf.202000102

Wu, Y., & Wang, G. (2018). Machine Learning Based Toxicity Prediction: From Chemical Structural Description to Transcriptome Analysis. International Journal of Molecular Sciences, 19(8), 2358. https://doi.org/10.3390/ijms19082358

Zhang, G. (2018, 11 noviembre). What is the kernel trick? Why is it important? - Grace Zhang - Medium. Medium. https://medium.com/@zxr.nju/what-is-the-kernel-trick-why-is-it-important-98a98db0961d

Zhao, Y.; Zhao, J.; Huang, Y.; Zhou, Q.; Zhang, X.; Zhang, S. (2014) Toxicity of ionic liquids: Database and prediction via quantitative structure-activity relationship method. J. Hazard. Mater. 278, 320–329

Zhong, S., Zhang, K., Bagheri, M., Burken, J. G., Gu, A. Z., Li, B., Ma, X., Marrone, B. L., Ren, Z. J., Schrier, J., Shi, W., Tan, H., Wang, T., Wang, X., Wong, B. M., Xiao, X., Yu, X., Zhu, J., & Zhang, H. (2021). Machine Learning: New Ideas and Tools in Environmental Science and Engineering. Environmental Science & Technology. https://doi.org/10.1021/acs.est.1c01339

Zhou, T., Gui, C., Sun, L., Hu, Y., Lyu, H., Wang, Z., Song, Z., & Yu, G. (2023). Energy Applications of Ionic Liquids: Recent Developments and Future Prospects. Chemical Reviews, 123(21), 12170-12253. https://doi.org/10.1021/acs.chemrev.3c00391