Proyecto entrega final

Materia:

Introducción a la inteligencia artificial

Victor Gabriel Navarro Serna 1037577906

Profesor:

Raúl Ramos Pollan

UNIVERSIDAD DE ANTIOQUIA

FACULTAD DE INGENIERIA

MEDELLIN 2022

Introducción

Como sabemos realizar predicciones de problemas a nivel molecular son bastantes complicadas, lo que representa un gran desafío para la ciencia de datos. Nuestro objetivo es predecir las interacciones entre átomos, es decir, predecir la interacción magnética entre dos átomos en una molécula, para mejor compresión lo que se desea hallar es la constante de acoplamiento escalar, que no es más que un número que determina la fuerza de la interacción respecto a la energía cinética. Está constante depende de los electrones y de los enlaces químicos que forman la estructura tridimensional de una molécula. Así que el obtener un método fiable y rápido para predecir estas interacciones permitirá a los científicos comprender cómo la estructura química 3D de una molécula afecta a sus propiedades y comportamiento, con lo que finalmente se pueden realizar tareas celulares o ayudar a mejorar el desarrollo de fármacos.

Dataset

Vamos a utilizar los siguientes dataset:

- train.csv: El conjunto de entrenamiento, donde la primera columna (molecule_name) es el nombre de la molécula donde se origina la constante de acoplamiento (el archivo XYZ correspondiente se encuentra en./structures/. xyz), la segunda (atom_index_0) y la tercera columna (atom_index_1) son los índices de los átomos del par de átomos que crean el acoplamiento y la cuarta columna (scalar_coupling_constant) es la constante de acoplamiento escalar que queremos poder predecir.
- test.csv: Es el conjunto de prueba; la misma información que el tren, sin la variable de destino.
- sample_submission.csv: Es un archivo de envío de muestra en el formato correcto.
- structures.csv: Este archivo contiene la misma información que los archivos de estructura xyz individuales, pero en un solo archivo.
- scalar_couplin_contributions.cvs: En este archivo tenemos las constantes de acoplamiento magnético.
- potential_energy.csv: En este archivo tenemos la energía potencial de un cuerpo dentro de un campo de fuerza.
- mulliquen_charges.csv: Este archivo contiene la afinidad electrónica, asociados a los potenciales de ionización.
- magnetic_shielding_tensor.cvs: Corresponde a un archivo que contiene las interacciones entre la fuerza eléctrica/magnética y el impulso mecánico.
- dipole_moment.cvs: Es un archivo que contiene la magnitud vectorial del dipolo magnético, el cual determina la intensidad de una fuente de campo magnético.

Iniciamos la exploración de los datasets

```
pot_energy=pd.read_csv('../input/potential_energy.csv')
mulliken_charges=pd.read_csv('../input/mulliken_charges.csv')
train_df=pd.read_csv('../input/train.csv')
test_df=pd.read_csv('../input/test.csv')
```

```
magnetic_shield_tensor=pd.read_csv('../input/magnetic_shielding_tensors.csv)
dipole_moment=pd.read_csv('../input/dipole_moments.csv')
structures=pd.read_csv('../input/structures.csv')
scalar_coupling_cont=pd.read_csv('../input/scalar_coupling_contributions.csv')
```

```
print('Shape of potential energy dataset:',pot_energy.shape)
print('Shape of mulliken_charges dataset:',mulliken_charges.shape)
print('Shape of train dataset:',train_df.shape)
print('Shape of dipole moments dataset:',dipole_moment.shape)
print('Shape of structures dataset:',structures.shape)
print('Shape of test dataset:',test_df.shape)
print('Shape of magnetic shielding tensors dataset:',magnetic_shield_tensor.shape)
print('Shape of scalar coupling contributions dataset:',scalar_coupling_cont.shape)
```

```
print('Data Types:\n',pot_energy.dtypes)
print('Descriptive statistics:\n',np.round(pot_energy.describe(),3))
pot_energy.head(6)
print('Data Types:\n',mulliken_charges.dtypes)
print('Descriptive statistics:\n',np.round(mulliken_charges.describe(),3))
mulliken_charges.head(6)
print('Data Types:\n',train_df.dtypes)
print('Descriptive statistics:\n',np.round(train_df.describe(),3))
train_df.head(6)
print('Data Types:\n',scalar_coupling_cont.dtypes)
print('Descriptive statistics:\n',np.round(scalar_coupling_cont.describe(),3))
scalar_coupling_cont.head(6)
print('Data Types:\n',test_df.dtypes)
print('Descriptive statistics:\n',np.round(test_df.describe(),3))
test_df.head(6)
print('Data Types:\n',magnetic_shield_tensor.dtypes)
print('Descriptive statistics:\n',np.round(magnetic_shield_tensor.describe(),3))
magnetic_shield_tensor.head(6)
print('Data Types:\n',structures.dtypes)
print('Descriptive statistics:\n',np.round(structures.describe(),3))
structures.head(6)
```

Luego se procedió a realizar el estudio de los data sets, en donde se hizo un mapa de la estructura atómica, luego se inicia el proceso de predicción de los modelos. En un segundo colab se hace el estudio con greedy forest. Ambos funcionan pero por falta de tiempo no se pudo comparar los resultados de predicción con los reales y someter el greedy forest.

Modelos utilizados

1. Modelo de regresión lineal

Como sabemos el modelo de regresión lineal es una técnica de modelado estadístico que se emplea para describir una variable de respuesta continua como una función de una o varias variables

predictoras, lo que nos ayuda a comprender y predecir el comportamiento de sistemas complejos, en nuestro caso analizar datos experimentales.

2. Modelo de regresión de Lasso

El modelo de regresión de Lasso (least absolute shrinkage and selection operator) es un método de análisis de regresión que realiza selección de variables y regularización para mejorar la exactitud e interpretabilidad, originalmente fue formulado para el método de mínimos cuadrados.

3. Greedy forest

El RGF es una poderosa técnica desarrollada por Rie Johnson y Tong Zhang en el documento 'Aprendizaje de funciones no lineales usando bosque codicioso regularizado'. Los árboles de decisión son quizás una de las técnicas más venerables utilizadas en el aprendizaje automático, en particular los algoritmos ID3 y C4.5 de Ross Quinlan. Los árboles de decisión son fáciles de implementar y explicar, pero tienden a sobre ajustarse. Se pueden usar tanto para la clasificación, como para la regresión. Los árboles de decisión son lo que ahora se conoce como alumnos débiles. Se demostró que uno es capaz de crear un alumno fuerte a partir de una colección, o 'conjunto', de alumnos débiles en un artículo famoso de Robert Schapire 'La fuerza de la capacidad de aprendizaje débil'. Fue a partir de esta idea que surgió el bosque de decisión, en el que, como su nombre indica, se crea una colección de árboles de decisión uno por uno. Esto se conoce con el nombre de impulsar. El proceso de impulso es lo que se conoce como ser codicioso; cada paso individual es óptimo (por ejemplo, cada árbol agregado al bosque) en ese momento, pero esto no conduce necesariamente a una solución óptima general.

Retos y consideraciones

- Retos:

Los principales retos de la programación es entender el que hacer y después el hacer, en nuestro caso nos encontramos con retos como: familiarizarse con el lenguaje, aprender a programar y finalmente entender por qué y para que de nuestro programa. Nos encontramos con el reto de que los datos a utilizar son muy grandes (pesados) por lo que en muchos casos el tiempo de ejecución es largo, lo cual se traduce en un alto costo en recurso de computación. Finalmente, el mayor reto fue la parte de generación de modelos y diseños de algoritmos, ya que son la parte fundamental del programa a modelar.

- Consideraciones:

Los problemas de física son difíciles de programar, más que nada en entenderlos, debido a su grado de complejidad, esto se ve reflejado en alto beneficio y grandes aplicaciones tanto en la industria como en la medicina. Por lo tanto, los problemas de la física son de un gran respeto y estima.

Conclusiones

1. La programación es una herramienta bastante potente permitiendo la solución de problemas cotidianos, los cuales nos dan una respuesta ante eventos a futuros.
2. La programación es fundamental para acrecentar el avance tecnológico dentro de las industrias que para llevar a cabo sus funciones necesitan de sitios y aplicaciones creadas a partir de códigos.
3. Vivimos en una sociedad en la de conseguir un trabajo es en una odisea, por lo que la programación se ha convertido en una fuente de empleo, ya que la programación facilita la resolución de problemas y la automatización de tareas a partir de una computadora.
4. Los problemas de programación son tareas arduas, largas y de un gran desgaste, lo que se traduce en un alto salario para los programadores.