|  |  |
| --- | --- |
| Name Of The Student | Himanshu |
| Internship Project Topic | TCS iON RIO-210: Build a Classification Model for Drug Trials Dataset |
| Name of the Organization | TCS iON |
| Name of the Industry Mentor | Himdweep Walia |
| Name of the Institute | Amity University |

|  |  |  |
| --- | --- | --- |
| Date | Day # | Hours Spent |
| 03-06-2024 | Day-41 | 6 Hours |
| Activities done during the day:  **Project Hands-on – Encoding the data**  **Link of the google drive google Colab file :-**  <https://colab.research.google.com/drive/1VQRq0l6oc9Uj4cOOqiuhkfS1JmpKr3fU?usp=sharing>  Encoding data involves converting information from one format into another, typically to facilitate storage or transmission. There are various encoding methods depending on the type of data and the purpose. Here are some common ones:  **Text Encoding**  **Numeric Encoding**  **Date & Time Encoding**  The below mentioned code preprocesses the data for machine learning. It removes the 'NAME' column, encodes categorical variables ('GENDER', 'RACE', 'SIDE EFFECTS') into numerical values, and converts 'DRUG NAME' into a list of strings. It then hashes the drug names into 10 dimensions using HashingVectorizer, converts the hashed features into a DataFrame, and concatenates them with the original data, excluding the 'DRUG NAME' column. This process transforms categorical and text data into a format suitable for machine learning algorithms, facilitating analysis and prediction tasks.  Also the second code segment splits the preprocessed data into features (X) and the target variable (y). Features (X) exclude the 'SIDE EFFECTS' column, while the target variable (y) consists solely of 'SIDE EFFECTS'. Then, it further divides the dataset into training and testing sets using train\_test\_split function from scikit-learn. The data is split with a test size of 20% (test\_size=0.2) and a random seed of 42 (random\_state=42) for reproducibility. This splitting prepares the data for training and evaluating machine learning models, ensuring distinct datasets for model training and evaluation.  from sklearn.preprocessing import LabelEncoder  from sklearn.feature\_extraction.text import HashingVectorizer  # Drop the 'NAME' column  data = data.drop('NAME', axis=1)  # Encode Categorical Variables  label\_encoder = LabelEncoder()  data['GENDER'] = label\_encoder.fit\_transform(data['GENDER'])  data['RACE'] = label\_encoder.fit\_transform(data['RACE'])  data['SIDE EFFECTS'] = label\_encoder.fit\_transform(data['SIDE EFFECTS'])  # Convert the 'DRUG NAME' column to a list of strings  drug\_names = data['DRUG NAME'].tolist()  # Initialize the HashingVectorizer with the desired number of dimensions  hashing\_vectorizer = HashingVectorizer(n\_features=10, alternate\_sign=False)  # Transform the drug names using the HashingVectorizer  hashed\_features = hashing\_vectorizer.transform(drug\_names)  # Convert the hashed features to a dense NumPy array  hashed\_features\_array = hashed\_features.toarray()  # Create a DataFrame with the hashed features  hashed\_features\_df = pd.DataFrame(hashed\_features\_array, columns=['DRUG\_NAME\_HASH\_{}'.format(i+1) for i in range(10)])  # Concatenate the hashed features with the original data  data\_encoded = pd.concat([data.drop('DRUG NAME', axis=1), hashed\_features\_df], axis=1)  # Split the data into features (X) and target (y)  X = data\_encoded.drop('SIDE EFFECTS', axis=1)  y = data\_encoded['SIDE EFFECTS']  # Split the data into training and testing sets  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42) | | |
|  | | |