|  |  |
| --- | --- |
| Name Of The Student | Vaishnavi G |
| Internship Project Topic | 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 | SRM Institute of Science and Technology |

|  |  |  |
| --- | --- | --- |
| Date | Day # | Hours Spent |
| 22/10/2022 | 11 | 7 hours |
| Activities done during the day:  Learned about data pre-processing steps for Machine Learning & Data analytics.  **Data pre processing with Python - Commands**  Different data set types with python   1. Missing values 2. Outliers 3. Overfitting 4. Data with no numerical values 5. Different date formats   **Missing values**   * Missing values are a common problem while dealing with data! The values can be missed because of various reasons such as human errors, mechanical errors, etc. * There are three techniques to solve the missing values’ problem in order to find out the most accurate features, and they are:  1. Dropping 2. Numerical imputation 3. Categorical imputation  * Dropping is the most common method to take care of the missed values. Those rows in the data set or the entire columns with missed values are dropped in order to avoid errors to occur in data analysis.  |  | | --- | | #Dropping columns in the data higher than 60% threshold  data = data[data.columns[data.isnull().mean() < threshold]]  #Dropping rows in the data higher than 60% threshold  data = data.loc[data.isnull().mean(axis=1) < threshold] |  * Numerical imputation - The word imputation implies replacing the missing values with such a value that makes sense. And, numerical imputation is done in the data with numbers.  |  | | --- | | #For filling all the missed values as 0  data = data.fillna(0)  #For replacing missed values with median of columns  data = data.fillna(data.median()) |  * Categorical imputation - This technique of imputation is nothing but replacing the missed values in the data with the one which occurs the maximum number of times in the column. But, in case there is no such value that occurs frequently or dominates the other values, then it is best to fill the same as “NAN”.  |  | | --- | | #Categorical imputation  data['column\_name'].fillna(data['column\_name'].value\_counts().idxmax(), inplace=True) |   **Outliers -**  An outlier differs significantly from other values and is too distanced from the mean of the values. Such values that are considered outliers are usually due to some systematic errors or flaws.   |  | | --- | | #For identifying the outliers with the standard deviation method  outliers = [x for x in data if x < lower or x > upper]  print('Identified outliers: %d' % len(outliers))  #Remove outliers  outliers\_removed = [x for x in data if x >= lower and x <= upper]  print('Non-outlier observations: %d' % len(outliers\_removed)) |   In the codes above, “lower” and “upper” signify the upper and lower limit in the data set.  **Overfitting -**  Overfitting occurs when the model fits the data too well or simply put when the model is too complex. Overfitting model learns the detail and noise in the training data to such an extent that it negatively impacts the performance of the model on new data/test data.   |  | | --- | | data['bin'] = pd.cut(data['value'], bins=[100,250,400,500], labels=["Lowest", "Mid", "High"]) | | | |