DMDW ASSGN-1 REPORT

Srujan R - BT18CSE041

Libraries used – pandas, matplotlib, numpy, scipy, sklearn

- I have taken a dataset on students' performance from Kaggle.
- I read the csv file using the pandas function - read_csv(), which gives a data frame(an in built data structure of pandas) as output.
- The dataset's shape is 1000 x 8, meaning, it has 1000 rows and 8 columns which got to know from the attribute dataframe.shape .
- The columns were 'gender', 'race/ethnicity', 'parental level of education', 'lunch', 'test preparation course', 'math score', 'reading score', 'writing score' which I got from the attribute dataframe.columns .
- The dataset didn't require any cleaning as there were no null values in any column, which could have been filled using dataframe.fillna() function if present.
- I checked the first 5 rows using the dataframe.head() function and the dataframe.tail() function for the last 5 rows. By passing an integer as an argument, we can get the required number of rows to be displayed from the start or the end.
- In this dataset, the columns – "math score", "reading score" and "writing score" are numerical data and the rest are categorical data.
- Gender has – "male" and "female" categories.
- Race/ethnicity has – groups "A" to "E" categories.
- Parental level of education has - "associate's degree", "bachelor's degree", "high school", "master's degree", "some college", "some high school" categories.
- Lunch has – "standard" and "free/reduced" categories.
- Test preparation course has – "none" and "completed" categories.
- I was able to get the info about the categories of these columns using the function dataframe[column name].value_counts() which gives the different

values in that column along with the number of times these values appeared in the dataset.

- I used df.describe() function to get the statistical information of the numerical data, info like mean, standard deviation, percentiles etc.
- I used dataframe.loc[slicing,[labels]] to get a view of the columns “gender” and “math score” and the rows from 0 to 10 only, this function is very useful for isolating a portion of the dataframe.
- I got more statistical info(averages) of the numerical data based on the categories of a column like “parental level of education” using the function dataframe.groupby(by=column name).mean().
- I used the matplotlib.pyplot and numpy libraries to create a bar graph representation.
- The bar graph was about numerical data averages of male vs those of female in the dataset, giving 3 bars adjacent bars for each of the 2 divisions.
- I stored the averages of all the 3 numerical data for both male and female in 3 lists. Then passed these lists as arguments in the plt.bar() function which represents a single bar in the bar graph, so 3 plt.bar() were used.
- I used the np.arange() function to get an np array with values equally spaced from a given start to a given end value. This array was used for the y-axis of bar graph which was Marks.
- I used plt.title() to set the title of bar graph.
- I used plt.xlabel() and plt.ylabel() to set names to the axes of the bar graph.
- I used plt.xticks() and plt.yticks() for setting the intervals between the divisions on the axes of the bar graph.
- I used plt.legend() to display the legend of the bar graph.
- I finally used plt.show() to show the bar group in stdout.
- I used the dataframe[[labels]].to_dict(‘list’) to get the required data stored in a dictionary where the column names are keys and the value is a list containing all the data from that column.
- Using this dictionary I was able to created a nested list of gender + race values present in the dataset – [[gender,race], [gender,race],……..].

- I used the dataframe[column].to_list() to get the values of a particular column into a python list.
- Using this list of math scores as input, I used the cube root function cbrt() and exponential of 10 function exp10() of scipy library to get the cube root and exponential values of all the math scores.
- I made a deep copy of the current dataframe to be used for prediction using ML. Deep copy makes sure that any changes made in the new dataframe doesn't affect the main previous dataframe.
- I used the dataframe.drop() function to drop unnecessary columns from the new database.
- I used the dataframe.replace() function to replace the categories of the categorical data into integer values so that these can be used as input for the prediction model.
- Using the sklearn libraries like train_test_split, I made a prediction model.
- Initially I created x and y dataframes where x would be all categorical data (leaving "lunch") converted into integers and y would be the "math score" column.
- Then I converted these x and y data structures into x_train, y_train and x_test, y_test, with 0.1 of the dataset reserved for testing the model and the rest for training the model, so 900 rows of data for training and 100 rows for testing.
- After the training, I used the accuracy library from the sklearn to calculate the accuracy of prediction among the testing dataset, where I got a score of 0.02 which means that the model got 2 tests correct out of 100 tests.
- The main reasons for the low score would be the small amount of data for training and treating categorical data as input by converting them into integers.

----END----