ML Major Project Summary

Dataset:

Name: Placement_Data_Full_Class.csv
Source:https://www.kaggle.com/benroshan/factors-affecting-campus-placement
Description: The above data set consists of placement data pertaining to students enrolled in Jain University, Banglore. It includes secondary and higher secondary grades (in percentages) and the stream that students chose for their higher secondary education. The data set also includes degree specialization, degree type, work experience and salary offers to the students who got placed.

Classification Algorithms Chosen:

1. Logistic Regression(LR)
2. KNN(K Nearest Neighbours)
3. Random Forest

Note: We also tried implementing our own version of a Logistic Regression algorithm, coded entirely from scratch.

Questions Chosen:

1. To get placed with the highest salary package, which Degree should one opt for?

2. People with which degree and specialization are more likely to be placed?

EDA and Working of Algorithms:

- First, we converted all the data set columns to a list to get an overview of the data we had on our hands.
- We first checked for null values. All the null values in the dataset were in the salary column and after a quick glance at the data set we found that the null values were only for those students who didn't get placed. So, we filled all the non-placed students' salaries as 0 and after rechecking for null values and not finding any, we moved on.
- Lastly we checked for any columns/features that could be dropped, and decided that the school board (for both secondary and higher secondary) and serial number were not relevant and thus dropped them.

Data Visualisation:

- We plotted all relevant columns in various types of graphs and plots to get a nice visual representation of our data set.

Feature Engineering:

- After careful analysis of our data, keeping in mind all the questions we wanted to answer using that data, we came to the conclusion that most of our data set had categorical data which could easily be converted to numeric data and it would help us train the algorithm better.
- We converted the following features to numeric/booleans (at times we increased column counts by splitting one column's

data into multiple columns since columns containing more than 2 values couldn't be used as booleans):

- Gender data converted to bool, with 1 representing Male and 0 representing Female
- HigherSecondary Stream (hsc_s) was split into 3 columns, each containing a bool that confirmed which stream the student opted for. (i.e only one of the 3 columns contained a 1 and the other two were set to zero, naturally)
- Degree Chosen (degree_t) was also split into 3 columns containing numeric booleans, in a similar fashion to the above entry, with the 3 columns representing Sci&Tech, Comm&Mgt and Others.
- Specialisation was transformed from categorical to boolean again by setting all Marketing and HR (Mkt&HR) entries as 1 and Marketing and Finance (Mkt&Fin) entries as 0.
- Status and Work Experience (WorkEx) again were converted from categorical to numeric(boolean) by assuming Placed as 1 and Not Placed as 0; and assuming work experience 'yes' as 1 and 'no' as 0.

- After taking another look at our data following these conversions, we dropped two more columns - hsc_s and degree_t; since we had already extracted relevant data from them and split them.

Assigning Variables and Setting Train/Test Ratio:

- Keeping in mind the questions we chose to answer with this dataset, we assigned a set of variables as Independent

variables and put them in a list and put the dependent variable in another list.

- Dependent Variable - Status (of placement)
- Independent Variables - Everything else that remained.
- After this, we set the Train/Test to a 0.75/0.25 split and proceeded with our chosen models.

ML Models:

Logistic Regression:

- For this model, we simply imported the concerned module from the sklearn package and passed our data set to the concerned function from said module.
- The team then generated a confusion matrix and a classification report using more functions from the sklearn package (metrics module). An accuracy percentage was also calculated using similar methods.

KNN(K Nearest Neighbors):

- For this model, we imported the KNN and GridSearchCV models from SKlearn library. Then we created a dictionary called 'params' containing a set of hyperparameters for the KNN model. The KNN model was then fitted with the training data set and then the GridSearchCV model was fitted with our trained KNN model and hyperparameters were set. GridSearchCV model chose the best hyperparameters from the set and used them to carry out KNN classification.

- The team again generated a confusion matrix and a classification report using the same functions as before, along with an accuracy percentage.

Random Forest:

- For this model, we again imported the concerned module from the sklearn package and passed our data set to the relevant function from said module.
- The team again generated a confusion matrix and a classification report using the same functions as before, along with an accuracy percentage.

Logistic Regression (From Scratch):

- For this model, we first saved the data as a csv file (to use a numerical-categorical data for the regression model(lr)).
- We then loaded the new file into the placeholder object "dataset". This object was then put through a data-type check and if strings were found, they were converted to floats.
- The data was then sorted and normalised, to make sure input values lay between 0-1 (to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values.) Finally, all available data was split into testing and training segments.
- Coefficients for each column were predicted using the coefficients_sgd function. Stochastic gradient descent requires two parameters:
    - Learning Rate: Used to limit the amount each coefficient is corrected each time it is updated.(0.1)

    - Epochs: The number of times to run through the training data while updating the coefficients.(100)

- Coefficients were then updated based on the error % of the model (Which was calculated as the difference between the expected output value and the prediction made with the candidate coefficients)
- There is one coefficient to weight each input attribute, and these were updated in a consistent way, using the formula:

  *b1(t+1) = b1(t) + learning_rate * (y(t) - yhat(t)) * yhat(t) * (1 - yhat(t)) * x1(t)*

  where, y(t) is the output value, yhat(t) is the predicted value, x1(t) is input value, b1 is the coefficient being updated.
- Values were then predicted using the sigmoid function once the coefficients for each candidate columns were determined

  Sigmoid fun: probability P= 1/1+e^-(b0+ b1*x1+ b2*x2...)

  where b0,b1,b2... are the coefficients, x1,x2,x3... are the respective values for each columns

  *b0+b1*x1+b2*x2.... is the weighted sum, used in the sigmoid function to predict the probability of dependent variable.*

  The predicted values were then rounded off, for binary classification.
- The predicted and actual values were compared using acurracy_matrix function. The accuracy for each fold was finally stored in the list "scores", and the average of all the scores was taken to be the accuracy of our logistic regression model. (n-fold(5) Cross Validation)

Comparing Accuracies:

Logistic Regression: 83.33%

KNN(K nearest neighbors): 78%

Random Forest: 74%

Logistic Regression (From Scratch): 99%

Conclusions:

- We can clearly see that Logical Regression wins in terms of accuracy, followed closely by KNN.
- To answer our **first question**, we isolated the required data (placement status and degree chosen) and got value counts for both features. We then used these value counts to make an lmplot which we then used to finally make a violinplot, which gave us our answer. We can conclude from the plot that people who opt for Commerce and Management get placed with the highest salary packages.
- To answer our **second question**, we again isolated data features relevant to our question (degree, specialisation and placement status) and got value counts for all three of them. We used these value counts to calculate probabilities (of being placed) for each possible combination of unique values from all 3 columns. Then we simply passed these calculated probabilities through a max function, which gave us our answer - people with a degree in Sci&Tech and specialisation in Mkt&Fin have the highest probability of being placed.