**INTELLIGENT ADMISSION : THE FUTURE OF UNIVERSITY DECISION MAKING WITH MACHINE LEARNING**

Define problem/Problem Understanding:

Specify The Business Problem

The process typically involve several steps,including submitting an application ,taking entrance exams, and participating in interviews or other evaluation.

The aim of this project is to help students in short listing universities with their proflies . Machine learing algorithms are then used to train a model on this data, which can be used to predict the changes of future , students can make more informed decision about which universitis to apply to , and universities can make more effcient use to their resources by focusing on the most promising application.

Business Requirements

The business requirements for a machine learing model to predict chanhes of student admission in university. A project aims to certain factors the business value of this project is that it well help student make more informed decisions about which university to apply to , and help university they are most likely to be admitted to the university.

Literauture survey

One study by (Hsu and Chen, 2019) used decision tree, random forest, and logistic regression algorithms to predict the chance of university admission based on students' GPA, test scores, and personal information. The study found that the random forest algorithm performed the best with an accuracy of 85.5%.Another study by (Al-Shammari et al., 2018) used the k-nearest neighbor (KNN) algorithm to predict the chance of university admission based on students' GPA, test scores, and family income. The study found that the KNN algorithm performed well with an accuracy of 81.2%.A study by (Najafabadi et al., 2015) used a neural network to predict the chance of university admission based on students' GPA, test scores, and personal information. The study found that the neural network performed well with an accuracy of 94.3%..Overall

Social or Business impact

using machine learning models to predict university admission, the service can help universities more efficiently process and evaluate applications, potentially increasing the number of successful admissions. An increase in the number of successful admissions can lead to an increase in revenue for universities, as well as for the company providing the prediction service.

Data Collection & Preperation

Collect The Dataset

In this project we have used ***.csv*** data. This data is downloaded from kaggle.com.

***“Link*** : <https://www.kaggle.com/rishal005/admission-predict>”

This .csv data have some student data.

Import the Libraries

import numpy as np

import pandas as pd

import matplotlib.pyplt as plt

import seaborn as sns

Read The Dataset

pandas madule can read **.csv , .json , .csv** , etc.. files.

student\_data = pd.read\_csv(‘Admission\_data.csv’)

Data preparation

As we have undershoot how the data, is let’s pre-process the collected data.The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

* Handling missing values
* Handling categorical data
* Handling Imbalance Data

Handling missing Value

**student\_data.shape() :** To fine the shape on our dataframe

**student\_data.info() :** To fine the data type on our dataframe

**student\_data.isnull.any() :**  To find the any null value from dataframe

Then, **change rename the colume**

student\_data.rename(columns = { ‘Chance of Admit : ’ Chance of Admit’ })

### **Exploratory Data Analysis**

### Descriptive Statistical

Descriptive analysis is to study the basic features of data with the statistical process. we can find mean, std, min, max and percentile values of continuous features.

**student.describe()**

Visual Analysis

Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data.

**Univariate Analysis**

Have displayed two different graphs such as distplot and countplot.

* The Seaborn package provides a wonderful function distplot. With the help of distplot, we can find the distribution of the feature. To make multiple graphs in a single plot, we use subplot.

**Sns.sidtplot(student[‘TOEFL Score’])**

**Bivariate Analysis**

**pair plot :** Plot pairwise relationships in a dataset

Pair plot usually gives pair wise relationships of the columns in the dataset

**Scatter plot**

Matplot has a built-in function to create scatterplots called scatter(). A scatter plot is a type of plot that shows the data as a collection of points

**Scaling the Data**

Scaling is one the important process, we have to perform on the dataset, because of data measures in different ranges can leads to mislead in prediction

Models such as KNN, Logistic regression need scaled data, as they follow distance based method and Gradient Descent concept.

Splitting data into **x** and **y**

x=student.iloc[:,0:7].values

y=student.iloc[:,7:].values

**Model Building**

**Training The Model In Multiple Algorithums**

Now our data is cleaned and it’s time to build the model. We can train our data on different algorithms. For this project we are applying four  classification algorithms. The best model is saved based on its performance.

**Logistic Regression Model**

A LogisticRegression algorithm is initialised and training data is passed to the model with the .fit() function. Test data is predicted with .predict() function and saved in a new variable.

**ANN Model**

Building and training an Artificial Neural Network (ANN) using the Keras library with TensorFlow as the backend. The ANN is initialised as an instance of the Sequential class, which is a linear stack of layers. Then, the input layer and two hidden layers are added to the model using the Dense class, where the number of units and activation function are specified. The output layer is also added using the Dense class with a sigmoid activation function. The model is then compiled with the Adam optimizer, binary cross-entropy loss function, and accuracy metric. Finally, the model is fit to the training data with a batch size of 100, 20% validation split, and 100 epochs.

**Import tensorflow as tf**

**from tensorflow import keras**

**from tensorflow.keras.layers import Dense,Activation,Dropout**

**from tensorflow.keras.optimizers import adam**

**model=keras.Sequential()**

**model.add(Dense(7,activation = ‘relu’,input\_dim=7))**

**model.add(Dense(7,activation = ‘relu’))**

**model.add(Dense(1,activation = ‘linear’))**

**model.summary()**

**Testing and Model**

In ANN we first have to save the model to the test the inputs

**Performance Testing & Hyperparameter Tuning**

**Testing Model With Multiple Evaluation Metrics**

Multiple evaluation metrics means evaluating the model's performance on a test set using different performance measures. This can provide a more comprehensive understanding of the model's strengths and weaknesses.

**Compare The Model**

**Logistics Regression Model**

**from sklearm.meterics import accuracy\_score,roc\_auc\_score,confusion\_matrix**

**print(“Accuracy score : %f“ %(accuracy\_score(y\_test,Y\_pred) \* 100))**

**print(“Recall score : %f ” % (recall\_score(y\_test , y \_pred) \* 100))**

**print(“ROC score : &f ” %(roc\_auc\_score(y\_test , y\_pred) \* 100))**

**print(confusion\_matrix(y\_test , y\_pred))**

**output**

**accuracy score : 90.00000**

**Recall score : 99.0673035**

**ROC score : 53.703712**

**[ [ 1 11]**

**[ 1 107] ]**

**Model Deployment**

**Save The Best Model**

Saving the best model after comparing its perfomance using different evaluation metrics meance select the model with the highesst performance and saving its weights

**# save the model in HDF5 format**

**model.save(‘model.h5’)**

**Integrate With Web Framework**

Building a web application that is integrated to the model we built.

* Building HTML pages
* Building server side script
* Run the web application

**Building HTML Pages**

* home.html
* predict.html

and save them in the templates folder.

**Build Python Code**

Import the libraries

Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (\_\_name\_\_) as argument.

Render HTML page:

Here we will be using a declared constructor to route to the HTML page which we have created earlier.

In the above example, ‘/’ URL is bound with the home.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrives the value from UI:

Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlie

Main fumcion :

**Run The Web Application**

**Open anaconda prompt from the start menu**

* Navigate to the folder where your python script is.
* Now type “python app.py” command
* Navigate to the localhost where you can view your web page.
* Click on the predict button from the top left corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

Now,Go the web browser and write the localhost url (http://127.0.0.1:5000) .

Input 1- Now, the user will give inputs to get the predicted result after clicking onto the predict button.

**Project Demonstration & Documentation**

I am using some modules such us pandas,numpy,matplotlib,ect..

pandas : Using to analysis data .

Numpy : numpy module using for fast and best perfomence.

Matplotlib : using to understanding the data’s on visue mode.

Then , tensorflow and etc moudle are claculate accuracy values, flack module create web page .