STUDENT M.S ADMISSION PREDICTION USING PYTHON AND FLASK

kowshick srinivasan

INDEX:

PROBLEM STATEMENT

PROBLEM CLASSFICATION

DATASET

DATA CLEANING

EDA

NORMALIZATION AND RESAMPLING

TRAIN-TEST SPLIT

MODEL CREATION

MODEL EVALUATION

FLASK APP

STUDENT M.S ADMISSION PREDICTION USING PYTHON AND FLASK

PROBLEM STATEMENT: To build a web app using flask to predict the admission prediction of a student in a particular university and also recommend up to more 5 universities.

PROBLEM CLASSFICATION: The above problem can be classified as supervised learning. It is a binary classification task which as classifies into either of 2 categories namely, accept/reject.

DATASET: The given data set consists of 11 dependent variable and 1 independent variable. The task is to classify into accept/reject based on the 11 dependent variables.

DATA CLEANING: Data cleaning is the process of preparing raw data for analysis by removing bad data, organizing the raw data, and filling in the null values. Ultimately, cleaning data prepares the data for the process of data mining when the most valuable information can be pulled from the data set.

The ability to understand and correct the quality of your data is imperative in getting to accurate final analysis. The data needs to be prepared in order to discover crucial patterns. Data mining is considered exploratory; data cleaning in data mining gives the user the ability to discover inaccurate or incomplete data–prior to the business analysis and insights. In most cases, data cleaning in data mining can be a laborious process and typically requires IT resources to help in the initial step of evaluating your data. Because data cleaning prior to data mining is so time-consuming, it creates a dilemma for data analysts: you don't have enough staff or time to clean the data. But without proper data quality, your final analysis will suffer in accuracy or you could potentially arrive at the wrong conclusion.

Methods of data cleaning:

- **Prevention of Unnecessary observation**

One of the important achievements of data cleansing is to assure that the information portfolio is clean from unnecessary observations, the unnecessary dataset is of two types: alternative observations and irrelevances observation.

- **Fix Data Structure**

Structural errors may arise during data exchange due to oversight of human omission or the inability of the person who is not well trained.

Here, we rectify wrong words and summarize group headings that are taking too much time. This is vital because a long group at the top may not be wholly seen on the chart.

- **Filter out Outliers**

Outliers are information spots that depart importantly from supervision in an information sort.

It is much designing, in the insight the same type of examinations.

- **Handle Missing Data**

You may terminate with absent values in data because of the omission of attention during information gathering or lack of confidentiality towards anyone.

There are two types of managing unavailable data, one is displaying the examinations from the information notes and the second is filling in new information.

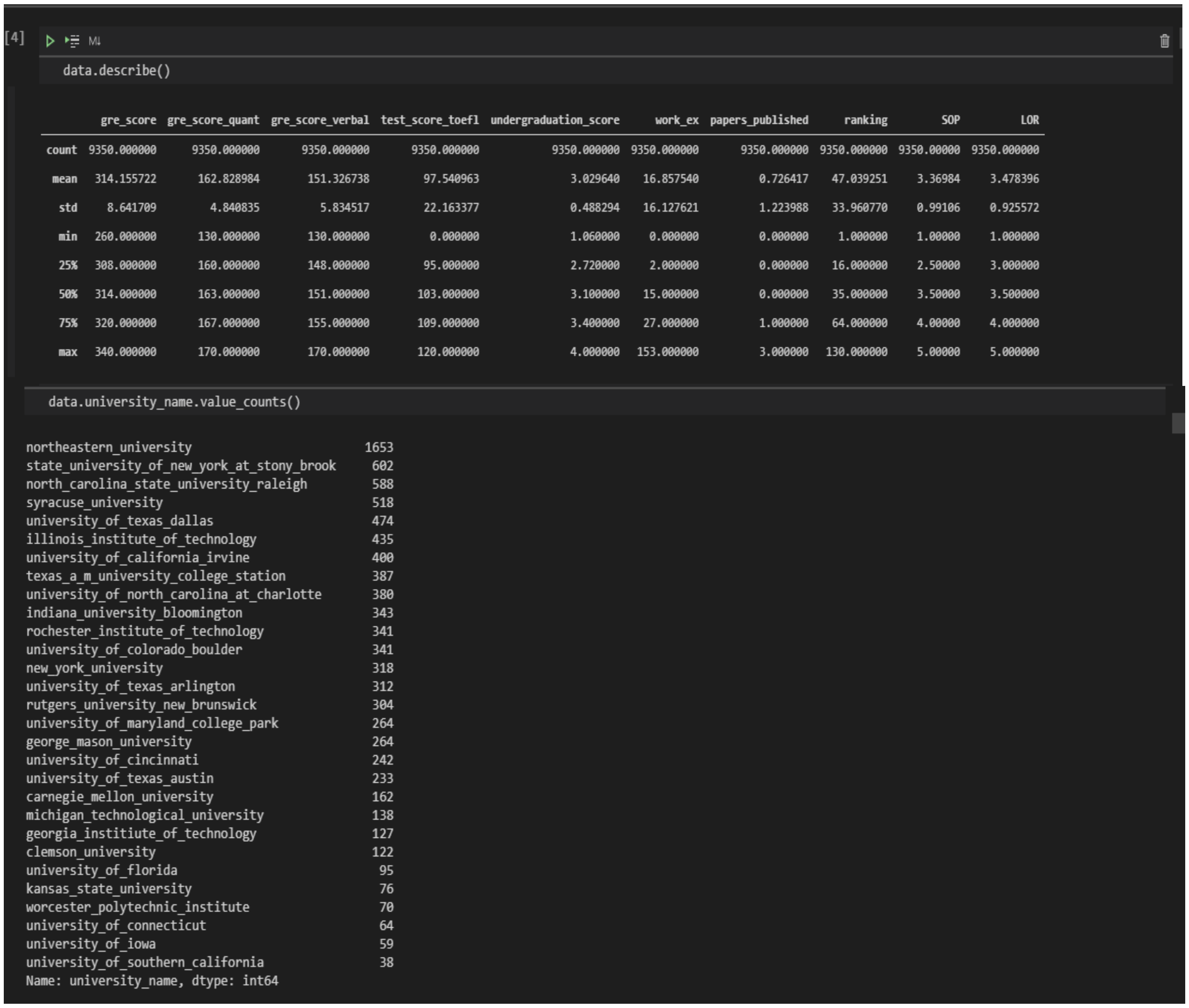
- **Drop Missing Values**

Dropping unavailable information assists in making a good decision.

There was no missing or inappropriate data in the dataset so that the dataset can be used as such and there is no need for data cleaning.

EXPLORATORY DATA ANALYSIS(EDA): eda refers to a critical process of investigation the initial data so to discover patterns, and spot anomalies, to test hypothesis and test assumptions with the help of summary statistics and graphical representations. In this project eda is done to discover the patterns in the data and to check for anomalies (if any). Methods used for EDA,

- **Describe data-** count, mean, median, standard deviation, minimum value, maximum value , 25% value, 50% value, 75% value.
- **Correlation plot analysis.**
- **Heatmap analysis**
- **Box plot analysis**
- **Feature extraction**
- **Kernel density estimate**



```
data.describe()
```

	gre_score	gre_score_quant	gre_score_verbal	test_score_toefl	undergraduation_score	work_ex	papers_published	ranking	SOP	LOR
count	9350.000000	9350.000000	9350.000000	9350.000000	9350.000000	9350.000000	9350.000000	9350.000000	9350.00000	9350.000000
mean	314.155722	162.828984	151.326738	97.540963	3.029640	16.857540	0.726417	47.039251	3.36984	3.478396
std	8.641709	4.840835	5.834517	22.163377	0.488294	16.127621	1.223988	33.960770	0.99106	0.925572
min	260.000000	130.000000	130.000000	0.000000	1.060000	0.000000	0.000000	1.000000	1.00000	1.000000
25%	308.000000	160.000000	148.000000	95.000000	2.720000	2.000000	0.000000	16.000000	2.50000	3.000000
50%	314.000000	163.000000	151.000000	103.000000	3.100000	15.000000	0.000000	35.000000	3.50000	3.500000
75%	320.000000	167.000000	155.000000	109.000000	3.400000	27.000000	1.000000	64.000000	4.00000	4.000000
max	340.000000	170.000000	170.000000	120.000000	4.000000	153.000000	3.000000	130.000000	5.00000	5.000000

```
data.university_name.value_counts()
```

```
northeastern_university                     1653
state_university_of_new_york_at_stony_brook  602
north_carolina_state_university_raleigh      588
syracuse_university                          518
university_of_texas_dallas                   474
illinois_institute_of_technology             435
university_of_california_irvine              400
texas_a_m_university_college_station         387
university_of_north_carolina_at_charlotte    380
indiana_university_bloomington               343
rochester_institute_of_technology            341
university_of_colorado_boulder               341
new_york_university                          318
university_of_texas_arlington                312
rutgers_university_new_brunswick             304
university_of_maryland_college_park          264
george_mason_university                      264
university_of_cincinnati                     242
university_of_texas_austin                   233
carnegie_mellon_university                   162
michigan_technological_university            138
georgia_instituite_of_technology             127
clemson_university                           122
university_of_florida                         95
kansas_state_university                       76
worcester_polytechnic_institute               70
university_of_connecticut                     64
university_of_iowa                            59
university_of_southern_california             38
Name: university_name, dtype: int64
```

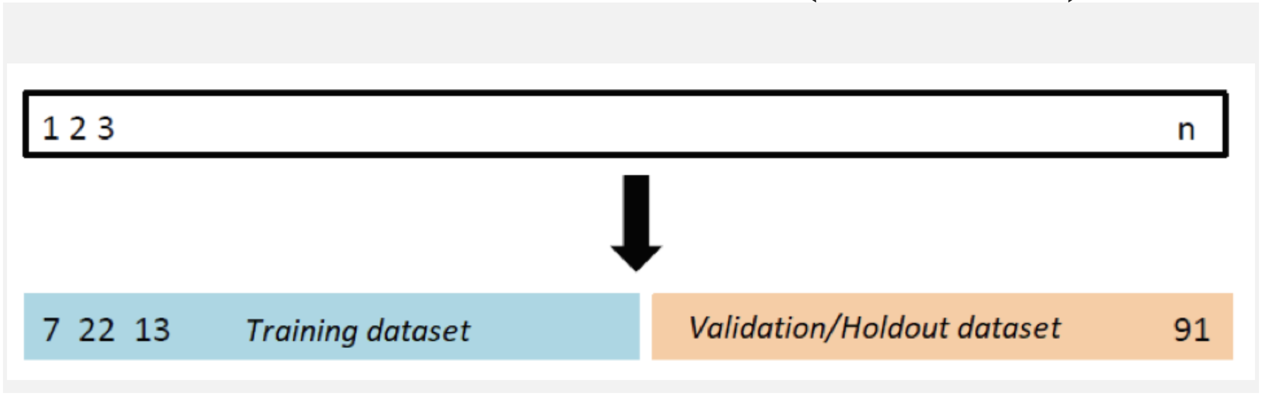
NORMALISATION AND RESAMPLING:

Resampling methods are very useful and beneficial in statistics and machine learning to fit more accurate models, model selection and parameter tuning.They draw samples from train data and fit model to check the variability of model and get additional information.We cannot best sure of the result of the model by just unique fit without testing on different sample or samples. It can be computationally expensive because of fitting model more than one, but recent improvements tackle this issue easily without too much effort.

Cross-Validation: The main point in the Machine Learning modelling is a training of model. Basically, we build a model on the train data, the data that we have in hand. And we are unsure about the result of our model on unseen data if we have not test data additionally to check the model accuracy. Only train accuracy is not good estimate of the test accuracy, because after one point (optimal point) train score tends to decrease and it leads to overfitting. It means that while the flexibility of our model increase , model try to predict each observation in the train set exactly , and in this scenario we get almost 100 percent of train accuracy , but is it good estimate ? No, due to the bias-variance trade-off after the optimal point of flexibility test error increases, and that is what we do not want. Our main aim is getting the optimal model which provide lower test error. To reach our goal, we should split our data into train and test set and always check the accuracy on our own test set.(or sets). This method is cross-validation and there are some types of cross validation as : Validation set Approach, Leave One Out Cross Validation(LOOCV), and k-fold cross-validation.

The Validation Set Approach:

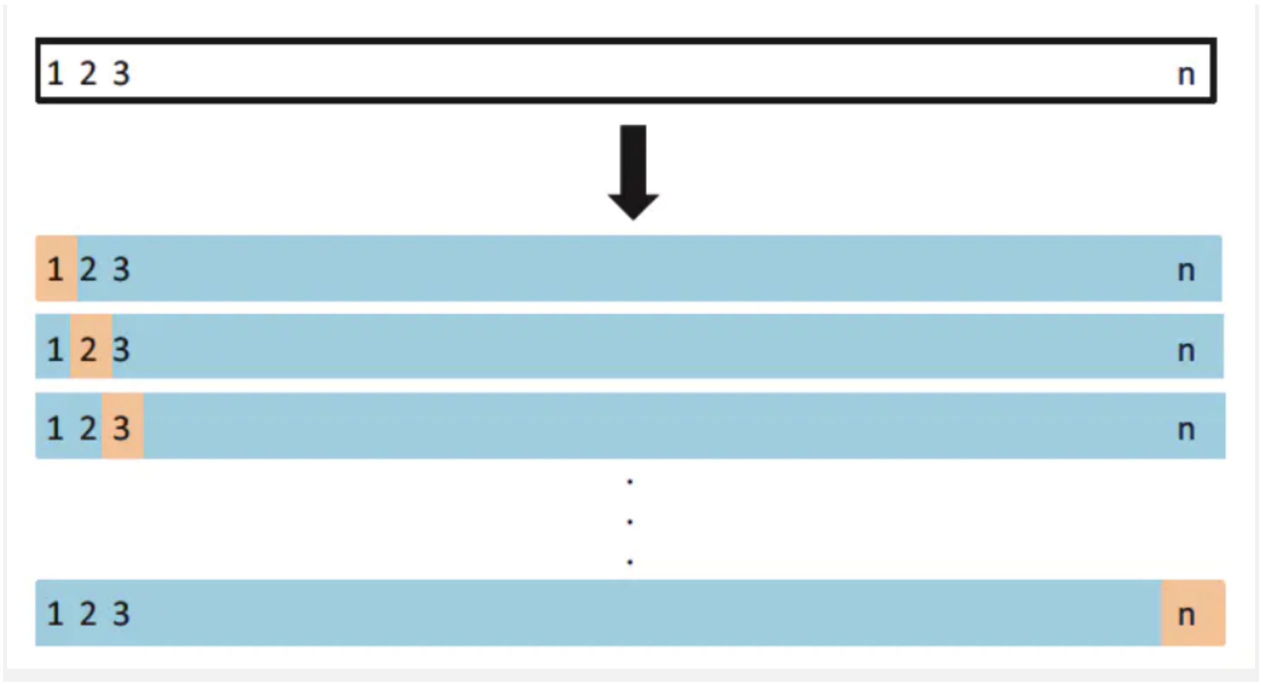
Validation Set approach is very simple method and frequently used method when there is sufficiently enough number of observations to get reasonable results. It is basically dividing the data that we have to two places as train set and validation set (or holdout set), and building model on train set, then checking the model accuracy on validation set. And resulting accuracy from validation set is the estimate about the real test data(unseen data).



But there are some drawbacks about this method. Firstly, we split the data set randomly to train and validation parts and fit our model on those train observations. So, our model trains on only those observations and maybe the observations in validation set are easy to predict, and this led to high variability in the accuracy, especially in small data sets. If we choose ten times different subsets as train and validation set and check the accuracy, we can see that it is not always same.

And also, we train our model on some part of observations only, and it leads to if we have not enough amount of data, it can be overestimating the real accuracy. For example, KNN can be best classifier for our problem but as we train on the small set, our validation set accuracy will be lower and we will think that it is not appropriate model, but in fact it is.

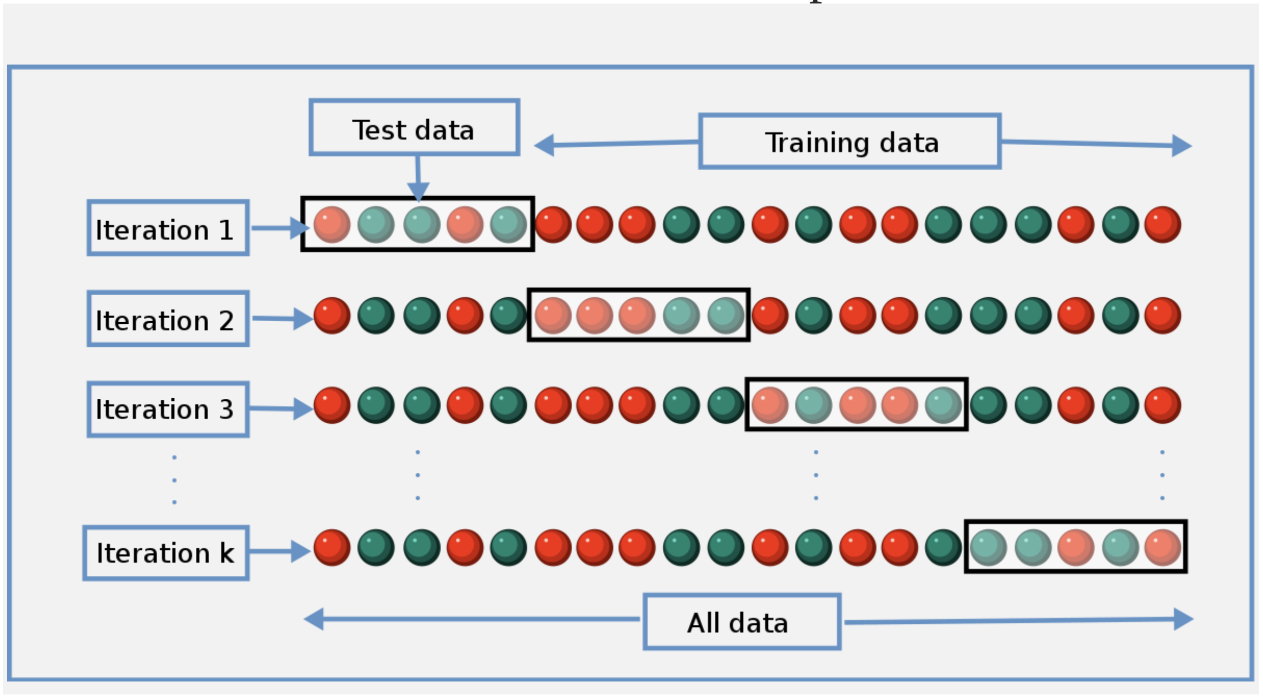
LOOCV: Leave One Out Cross Validation method is addressed to the drawbacks of the validation set approach and it is also simple method as validation set approach. Main point here is to take each observation as a validation set one time. It means that if we have "n" number of observations, we will fit model "n" times. And in every try we will keep one observation as a test sample, and will train the model on "n-1" observations. At the end we will have "n" accuracy scores, and we can take the mean of these scores as the overall score of the model.



As in this method each observation becomes test sample one time, it is highly unbiased method for estimating the real accuracy. But as we said bias is not the only case, and there should be also low variance. In LOOCV variance is high, because we test our model every time on single observation and it is not enough to get more accurate estimations. Also, it is computationally expensive, as we fit model “n” times. For example, if we have 1000 observations, it means that we will fit 1000 models.

k-Fold Cross-Validation:

Most used cross-validation technique is k-Fold method. Here the procedure is actually same with LOOCV but we do not fit model “n” times. “K” is the number of folds, for example 5-Fold cross-validation. Let’s say that we have 100 observations and we will use 5-Fold cross validation. It means that we will fit model only five times. In each iteration we will keep 20 observation as a validation set and 80 observation as train. As in the picture below:



After 5 iterations, we will have 5 score and we can take the mean of these scores as overall estimation of accuracy. This method is optimal method, bias is not high as validation set approach and variance is not high as LOOCV. And also, is not so computationally expensive. In practice frequently used ones are 5-Fold and 10-Fold.

Basically, resampling is used to remove the imbalance between the data thus avoiding overfitting of the data during the model evaluation.

Normalization is the techniques in data analysis used to scale the data into a particular window say 0 to 1 where all the values in the data set is fitted in the range of 0 and 1.

Types of normalization:

- Standardization
- Min-Max normalization

Train-Test Split

The train-test split is a technique for evaluating the performance of a machine learning algorithm. It can be used for classification or regression problems and can be used for any supervised learning algorithm. The procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model and is referred to as the training dataset. The second subset is not used to train the model; instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values. This second dataset is referred to as the test dataset.

- **Train Dataset**: Used to fit the machine learning model.
- **Test Dataset**: Used to evaluate the fit machine learning model.

The objective is to estimate the performance of the machine learning model on new data: data not used to train the model. This is how we expect to use the model in practice. Namely, to fit it on available data with known inputs and outputs, then make predictions on new examples in the future where we do not have the expected output or target values. The train-test procedure is appropriate when there is a sufficiently large dataset available.

MODEL CREATION: A machine learning model can be a mathematical representation of a real-world process. To generate a machine learning model, you will need to provide training data to a machine learning algorithm to learn from.

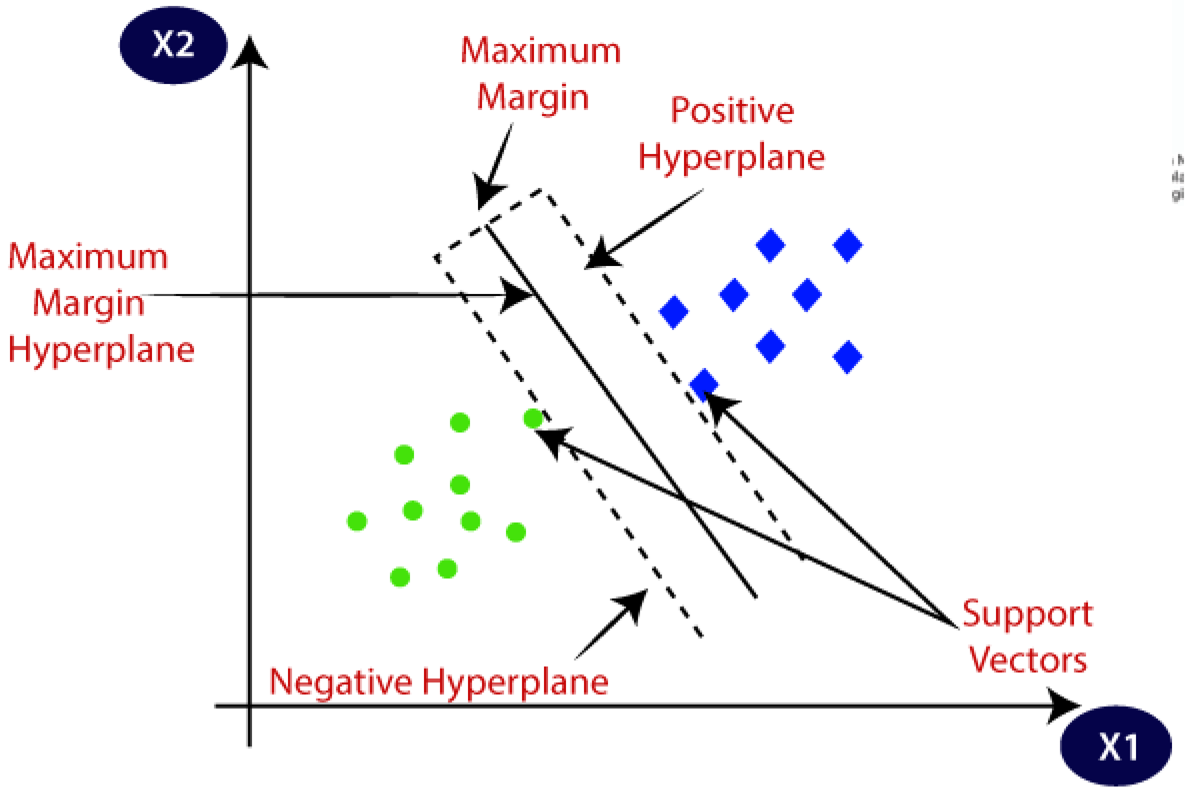
Models used in this, project:

- **Support vector machine**
- **Decision tree classification**
- **Random Forest**
- **Gaussian Classification**
- **XGBoost**
- **MLP Classification**
- **Bagging**

Support vector machine:

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
- SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

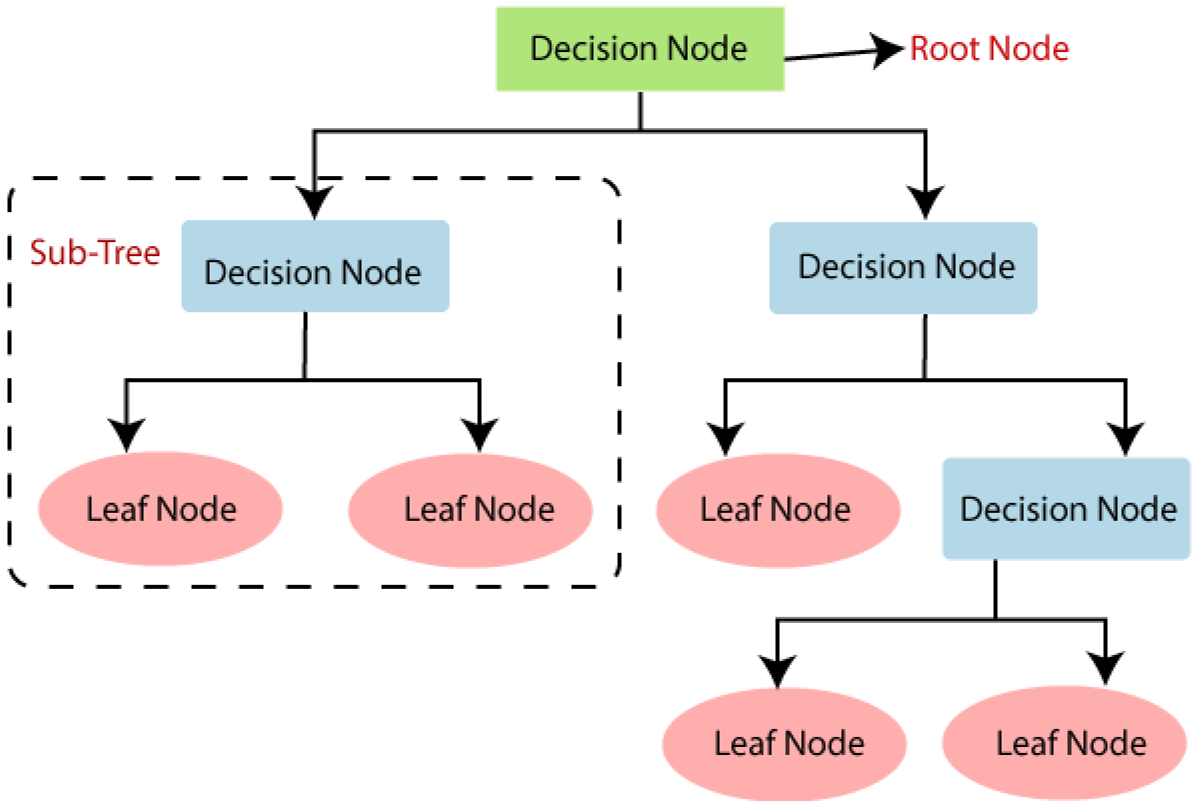


Decision Tree classification:

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

- In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:

Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.



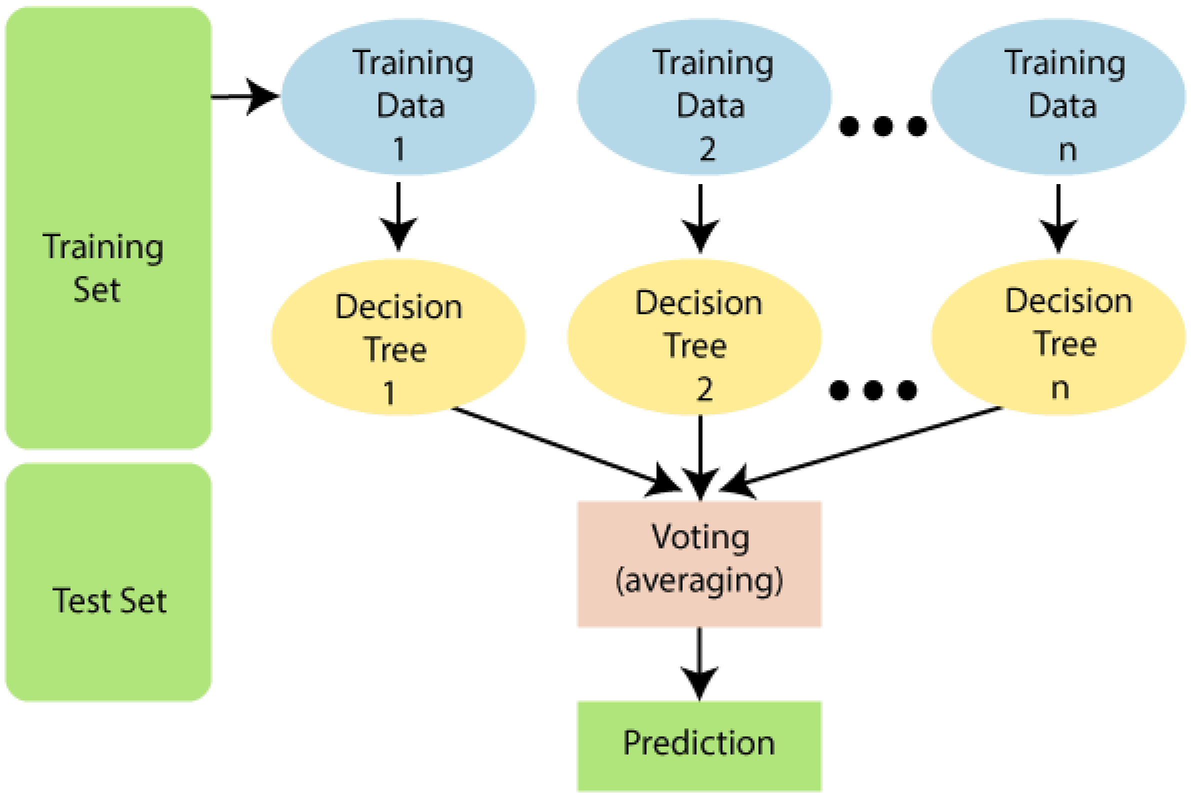
Random forest Classification:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning,** which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

As the name suggests, ***"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."*** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:



Gaussian Classification:
The **Gaussian Processes Classifier** is a classification machine learning algorithm.
Gaussian Processes are a generalization of the Gaussian probability distribution and can be used as the basis for sophisticated non-parametric machine learning algorithms for classification and regression.

They are a type of kernel model, like SVMs, and unlike SVMs, they are capable of predicting highly calibrated class membership probabilities, although the choice and configuration of the kernel used at the heart of the method can be challenging.

XGBoost:

Basically, XGBoost is an algorithm. Also, it has recently been dominating applied machine learning. XGBoost is an implementation of **gradient boosted** decision trees. Although, it was designed for speed and performance. Basically, it is a type of software library. That you can download and install on your machine. Then have to access it from a variety of interfaces.

MLP Classifier:

Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function $f(\cdot):Rm \rightarrow Ro$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output. Given a set of features $X=x1,x2,...,xm$ and a target y, it can learn a non-linear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers. Figure 1 shows a one hidden layer MLP with scalar output.

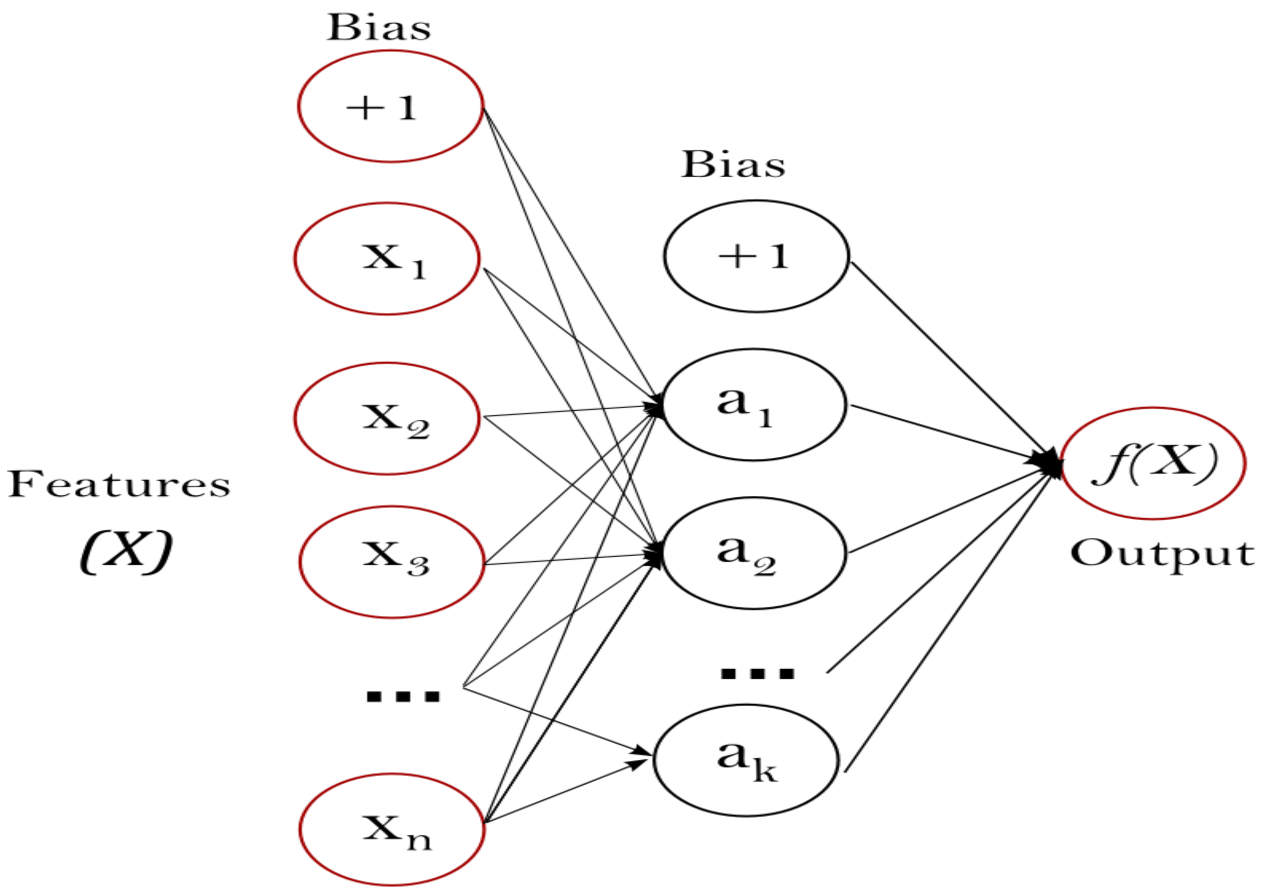


Figure 1: One hidden layer MLP.

The leftmost layer, known as the input layer, consists of a set of neurons {xi|x1, x2, ..., x_m} representing the input features. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation w1x1+w2x2+...+w_m x_m

, followed by a non-linear activation function $g(\cdot):R \rightarrow R$ - like the hyperbolic tan function. The output layer receives the values from the last hidden layer and transforms them into output values.

The module contains the public attributes coefs_ and intercepts_. coefs_ is a list of weight matrices, where weight matrix at index i represents the weights between layer i and layer i+1. intercepts_ is a list of bias vectors, where the vector at index i represents the bias values added to layer i+1.

The advantages of Multi-layer Perceptron are:

- Capability to learn non-linear models.
- Capability to learn models in real-time (on-line learning) using partial_fit.

The disadvantages of Multi-layer Perceptron (MLP) include:

- MLP with hidden layers have a non-convex loss function where there exists more than one local minimum. Therefore different random weight initializations can lead to different validation accuracy.
- MLP requires tuning a number of hyperparameters such as the number of hidden neurons, layers, and iterations.

- MLP is sensitive to feature scaling.

HYPERPARAMETER OPTIMIZATION

In machine learning, **hyperparameter optimization** or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. By contrast, the values of other parameters (typically node weights) are learned.

The same kind of machine learning model can require different constraints, weights or learning rates to generalize different data patterns. These measures are called hyperparameters, and have to be tuned so that the model can optimally solve the machine learning problem. Hyperparameter optimization finds a tuple of hyperparameters that yields an optimal model which minimizes a predefined loss function on given independent data. The objective function takes a tuple of hyperparameters and returns the associated loss. Cross-validation is often used to estimate this generalization performance.

Grid search

The traditional way of performing hyperparameter optimization has been *grid search*, or a *parameter sweep*, which is simply an exhaustive searching through a manually specified subset of the hyperparameter space of a learning algorithm. A grid search algorithm must be guided by some performance metric, typically measured by cross-validation on the training set or evaluation on a held-out validation set.

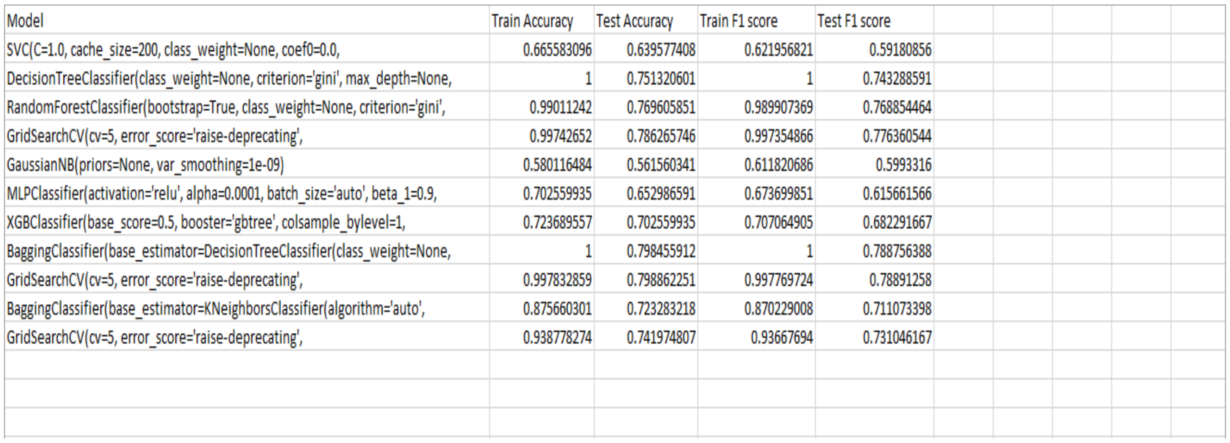
Since the parameter space of a machine learner may include real-valued or unbounded value spaces for certain parameters, manually set bounds and discretization may be necessary before applying grid search.

For example, a typical soft-
margin SVM classifier equipped with an RBF kernel has at least two hyperparameters that need to be tuned for good performance on unseen data: a regularization constant *C* and a kernel hyperparameter γ. Both parameters are continuous, so to perform grid search, one selects a finite set of "reasonable" values for each, say Grid search then trains an SVM with each pair (*C*, γ) in the Cartesian product of these two sets and evaluates their performance on a held-out validation set (or by internal cross-validation on the training set, in which case multiple SVMs are trained per pair). Finally, the grid search algorithm outputs the settings that achieved the highest score in the validation procedure.

Grid search suffers from the curse of dimensionality, but is often embarrassingly parallel because the hyperparameter settings it evaluates are typically independent of each other.

MODEL EVUALATION:

Evaluating the trained model against the test set so to see how the model performs on real-world/ unseen data. After evaluating our models against the test sets the accuracy scores obtained are given below.



Model	Train Accuracy	Test Accuracy	Train F1 score	Test F1 score
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,	0.665583096	0.639577408	0.621956821	0.59180856
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,	1	0.751320601	1	0.743288591
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',	0.99011242	0.769605851	0.989907369	0.768854464
GridSearchCV(cv=5, error_score='raise-deprecating',	0.99742652	0.786265746	0.997354866	0.776360544
GaussianNB(priors=None, var_smoothing=1e-09)	0.580116484	0.561560341	0.611820686	0.5993316
MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,	0.702559935	0.652986591	0.673699851	0.615661566
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,	0.723689557	0.702559935	0.707064905	0.682291667
BaggingClassifier(base_estimator=DecisionTreeClassifier(class_weight=None,	1	0.798455912	1	0.788756388
GridSearchCV(cv=5, error_score='raise-deprecating',	0.997832859	0.798862251	0.997769724	0.78891258
BaggingClassifier(base_estimator=KNeighborsClassifier(algorithm='auto',	0.875660301	0.723283218	0.870229008	0.711073398
GridSearchCV(cv=5, error_score='raise-deprecating',	0.938778274	0.741974807	0.93667694	0.731046167

STORING THE MODELS:

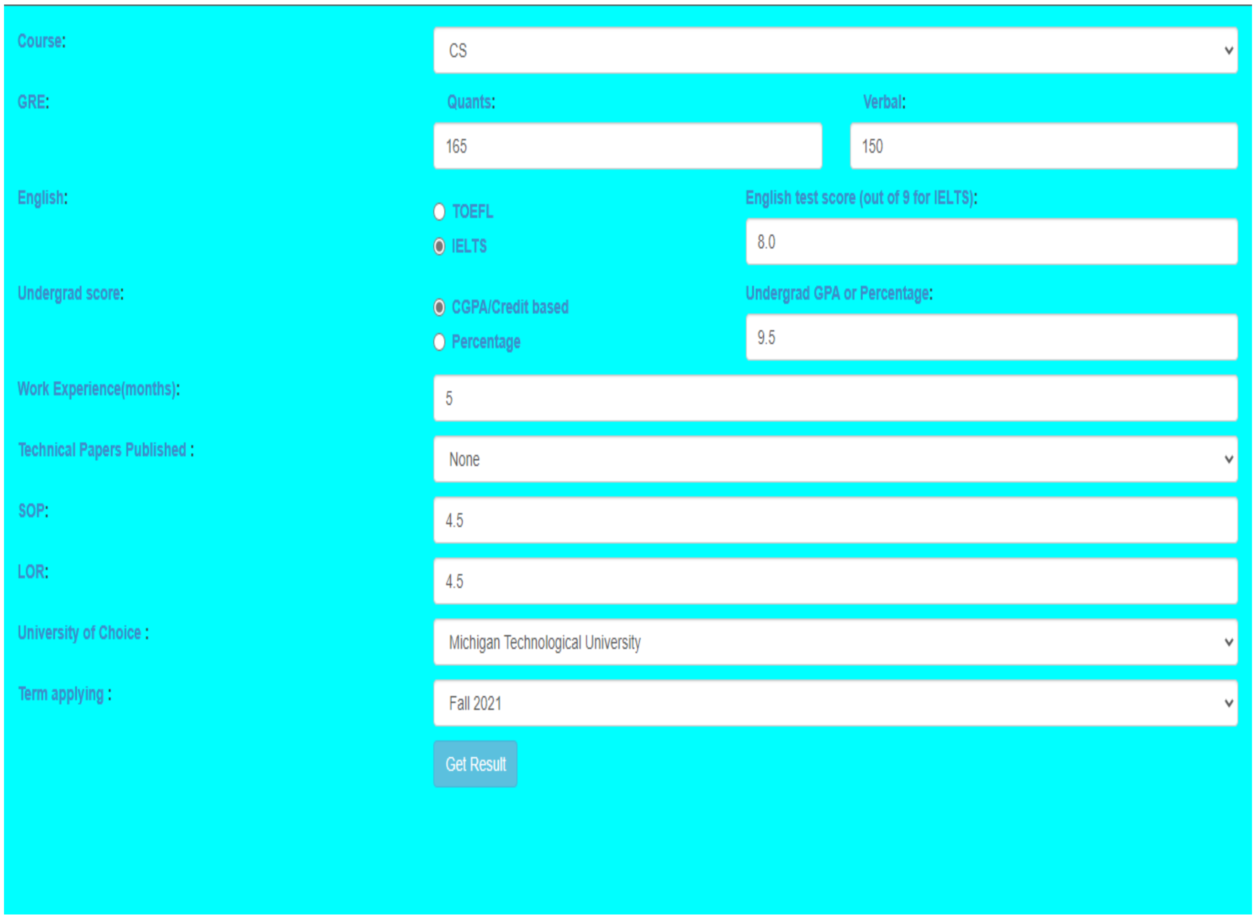
The above models are stored in the form of data stream. This, is done by the techniques of pickling of data. *"Pickling"* is the process whereby a Python object hierarchy is converted into a byte stream, and *"unpickling"* is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy. Pickling (and unpickling) is alternatively known as "serialization", "marshalling," or "flattening"; however, to avoid confusion, the terms used here are "pickling" and "unpickling".

It is also done to compress the model size.

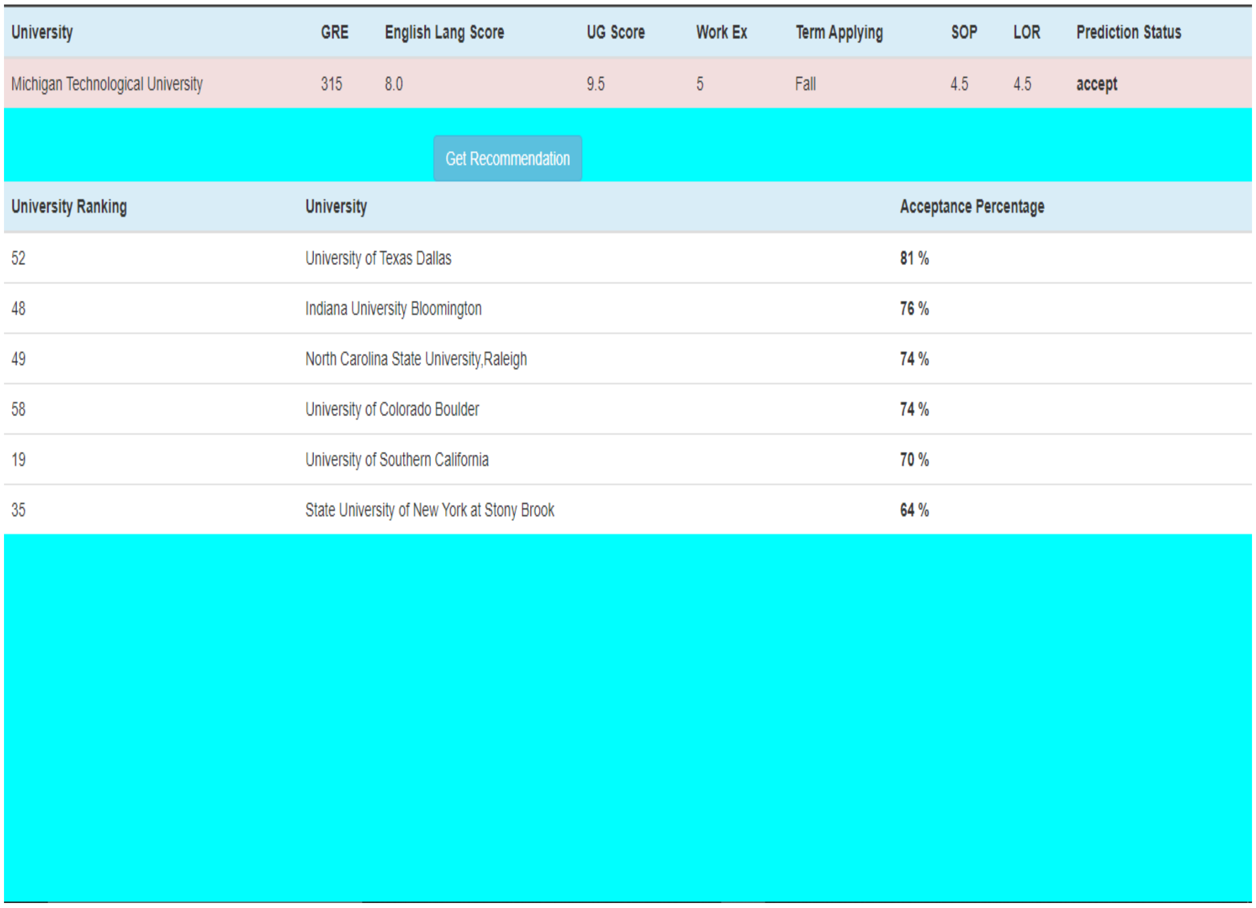
FLASK-APP

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

In this project a web app for prediction the change of admit of a student in a particular university for MS and then recommend any 5 universities in which they may get admitted to based on the information given by the user which includes GRE score (quants and verbal), English proficiency test score, U.G score, work experience, number of research papers published, SOP score, LOR score and the student choice of university.



Course:	CS	
GRE:	Quants: 165	Verbal: 150
English:	TOEFL [✓ IELTS]	English test score (out of 9 for IELTS): 8.0
Undergrad score:	[✓ CGPA/Credit based] Percentage	Undergrad GPA or Percentage: 9.5
Work Experience(months):	5	
Technical Papers Published :	None	
SOP:	4.5	
LOR:	4.5	
University of Choice :	Michigan Technological University	
Term applying :	Fall 2021	



University	GRE	English Lang Score	UG Score	Work Ex	Term Applying	SOP	LOR	Prediction Status
Michigan Technological University	315	8.0	9.5	5	Fall	4.5	4.5	**accept**

University Ranking	University	Acceptance Percentage
52	University of Texas Dallas	**81 %**
48	Indiana University Bloomington	**76 %**
49	North Carolina State University,Raleigh	**74 %**
58	University of Colorado Boulder	**74 %**
19	University of Southern California	**70 %**
35	State University of New York at Stony Brook	**64 %**

Conclusion and future Scope: Thus, a web app using flask to predict the admission prediction of a student in a particular university and also recommend up to more 5 universities was built .

Future scope:

- Include more universities
- Expand the filed od study

- Include the feature to predict scholarship details
- Suggest areas to improve for the student to get probably admitted in the particular university.
- Auto-ml