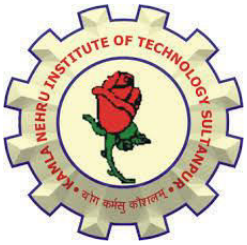
MACHINE LEARNING PROJECT



An Internship Report

On

"ADMISSION PREDICTION USING ML"

Submitted by

1. Shweta mishra
2. Akshay khandare
3. Pranay wanjari
4. Omkar joshi

Under the guidance of

Mr. Gurvansh Singh

M.Tech

Knowledge Solution India

KNOWLEDGE SOLUTION INDIA

DHOLEP ATIL ROAD,PUNE

WWW.KSINDIA.CO.IN

2 Table of content

*****

Abstract

In this project, we were asked to experiment with a real world dataset, and to explore how machine learning algorithms can be used to find the patterns in data. We were expected to gain experience using a common data-mining and machine learning library, Weka, and were expected to submit a report about the dataset and the algorithms used. After performing the required tasks on a dataset of my choice, herein lies my final report. Keywords: Machine Learning, Pattern Recognition, Classification, Supervised learning, Artificial Intelligence.

Introduction:

Machine learning is a sub-domain of computer science which evolved from the study of pattern recognition in data, and also from the computational learning theory in artificial intelligence. It is the first-class ticket to most interesting careers in data analytics today. As data sources proliferate along with the computing power to process them, going straight to the data is one of the most straightforward ways to quickly gain insights and make predictions. Machine Learning can be thought of as the study of a list of sub-problems, viz: decision making, clustering, classification, forecasting, deep-learning, inductive logic programming, support vector machines, reinforcement learning, similarity and metric learning, genetic algorithms, sparse dictionary learning, etc. Supervised learning, or classification is the machine learning task of inferring a function from a labeled data . In Supervised learning, we have a training set, and a test set. The training and test set consists of a set of examples consisting of input and output vectors, and the goal of the supervised learning algorithm is to infer a function that maps the input vector to the output vector with minimal error. In an optimal scenario, a model trained on a set of examples will classify an unseen example in a correct fashion, which requires the model to generalize from the training set in a reasonable way. In layman's terms, supervised learning can be termed as the process of concept learning, where a brain is exposed to a set of inputs and result vectors and the brain learns the concept that relates said inputs to outputs. A wide array of supervised machine learning algorithms are available to the machine learning enthusiast, for example Neural Networks, Decision Trees, Support Vector Machines, Random Forest, Naïve Bayes Classifier, Bayes Net, Majority Classifier etc., and they

each have their own merits and demerits. There is no single algorithm that works for all cases, as merited by the No free lunch theorem . In this project, we try and find patterns in a dataset , which is score record of under graduate student. there marks in GRE Score,TOEFL Score,University Rating,SOP,LOR ,CGPA,Research,Chan

Machine learning (ML):

the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop conventional algorithms to perform the needed tasks.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

Simple Definition: Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.

Types of macine learning:

Supervised learning:

A support vector machine is a supervised learning model that divides the data into regions separated by a linear boundary. Here, the linear boundary divides the black circles from the white.

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that

improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

Unsupervised learning:

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, such as finding the probability density function. Though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Semi-supervised learning:

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Some of the training examples are missing training labels, yet many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

In weakly supervised learning, the training labels are noisy, limited, or imprecise; however, these labels are often cheaper to obtain, resulting in larger effective training sets.

Reinforcement learning

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms. In machine learning, the environment is typically represented as a Markov decision process (MDP). Many reinforcement learning algorithms use dynamic programming techniques. Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible.

Reinforcement learning algorithms are used in autonomous vehicles or in learning to play a game against a human opponent.

Software Libraries in python:

Pandas:

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python data analysis" itself.

Numpy:

Numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

Matplotlib:

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.[3] SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter, since then it has an active development community, and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012, and further joined by Thomas Caswell.

Seaborn:

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

For a brief introduction to the ideas behind the library, you can read the introductory notes. Visit the installation page to see how you can download

the package and get started with it. You can browse the example gallery to see what you can do with seaborn, and then check out the tutorial and API reference to find out how.

To see the code or report a bug, please visit the GitHub repository. General support questions are most at home on stackoverflow or discourse, which have dedicated channels for seaborn.

Random module:

This module is subject to page protection. It is a highly visible module in use by a very large number of pages, or is substituted very frequently. Because vandalism or mistakes would affect many pages, and even trivial editing might cause substantial load on the servers, it is protected from editing.

This module contains a number of functions that use random numbers. It can output random numbers, select a random item from a list, and reorder lists randomly. The randomly reordered lists can be output inline, or as various types of ordered and unordered lists. The available functions are outlined in more detail below.

Problems and Issues in Supervised learning:

Before we get started, we must know about how to pick a good machine learning algorithm for the given dataset. To intelligently pick an algorithm to use for a supervised learning task, we must consider the following factors : 1. Heterogeneity of Data: Many algorithms like neural networks and support vector machines like their feature vectors to be homogeneous numeric and normalized. The algorithms that employ distance metrics are very sensitive to this, and hence if the data is heterogeneous, these methods should be the afterthought. Decision Trees can handle heterogeneous data very easily. 2. Redundancy of Data: If the data contains redundant information, i.e. contain highly correlated values, then it's useless to use

distance based methods because of numerical instability. In this case, some sort of Regularization can be employed to the data to prevent this situation. 3. Dependent Features: If there is some dependence between the feature vectors, then algorithms that monitor complex interactions like Neural Networks and Decision Trees fare better than other algorithms.

Bias-Variance Tradeoff: A learning algorithm is biased for a particular input x if, when trained on each of these data sets, it is systematically incorrect when predicting the correct output for x, whereas a learning algorithm has high variance for a particular input x if it predicts different output values when trained on different training sets. The prediction error of a learned classifier can be related to the sum of bias and variance of the learning algorithm, and neither can be high as they will make the prediction error to be high. A key feature of machine learning algorithms is that they are able to tune the balance between bias and variance automatically, or by manual tuning using bias parameters, and using such algorithms will resolve this situation. 5. Curse of Dimensionality: If the problem has an input space that has a large number of dimensions, and the problem only depends on a subspace of the input space with small dimensions, the machine learning algorithm can be confused by the huge number of dimensions and hence the variance of the algorithm can be high. In practice, if the data scientist can manually remove irrelevant features from the input data, this is likely to improve the accuracy of the learned function. In addition, there are many algorithms for feature selection that seek to identify the relevant features and discard the irrelevant ones, for instance Principle Component Analysis for unsupervised learning. This reduces the dimensionality. 6. Overfitting: The programmer should know that there is a possibility that the output values may constitute of an inherent noise which is the result of human or sensor errors.

In this case, the algorithm must not attempt to infer the function that exactly matches all the data. Being too careful in fitting the data can cause overfitting, after which the model will answer perfectly for all training examples but will have a very high error for unseen samples. A practical way of preventing this is stopping the learning process prematurely, as well as applying filters to the data in the pre-learning phase to remove noises. Only after considering all these factors can we pick a supervised learning algorithm that works for the dataset we are working on. For example, if we were working with a dataset consisting of heterogeneous data, then decision trees would fare better than other algorithms. If the input space of the dataset we were working on had 1000 dimensions, then it’s better to first perform PCA on the data before using a supervised learning algorithm on it.

Dataset

Let's start with loading all the libraries and dependencies.

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import random
```

The columns in the dataset are Serial Number, GRE score, TOEFL score, University Rating, SOP, LOR, CGPA, Research and Chance of Admit.

read_csv is a pandas function to read csv files.

head() method is used to return top n (5 by default) rows of a DataFrame or series.

I removed the Serial Number column as that does not seem relevant to the context here. The first five rows of the dataset are shown below.

```
df = pd.read_csv("Admission_Predict_Ver1.1.csv")

df.columns
```

```
Index(['Serial No.', 'GRE Score', 'TOEFL Score', 'University
Rating', 'SOP',
       'LOR ', 'CGPA', 'Research', 'Chance of Admit '],
dtype='object')
```

```
len(df.columns)
```

```
9
```

Dropping unwanted columns (Example:Art No , Correspondence Address , volume)

```
df = df.drop('Serial No.',axis = 1)
df.head()
```

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	337	118	4	4.5	4.5	9.65	1	0.92
1	324	107	4	4.0	4.5	8.87	1	0.76
2	316	104	3	3.0	3.5	8.00	1	0.72
3	322	110	3	3.5	2.5	8.67	1	0.80
4	314	103	2	2.0	3.0	8.21	0	0.65

Check if there are fields contianing null values

df.isnull().any()

```
Serial No.           False
GRE Score            False
TOEFL Score          False
University Rating    False
SOP                  False
LOR                  False
CGPA                 False
Research             False


dtype: bool
```

df.describe(include='all')

Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	
count	500.000000	500.000000	500.000000	500.000000	500.000000	500.00000	500.000000	500.000000
mean	250.500000	316.472000	107.192000	3.114000	3.374000	3.48400	8.576440	0.560000
std	144.481833	11.295148	6.081868	1.143512	0.991004	0.92545	0.604813	0.496884
min	1.000000	290.000000	92.000000	1.000000	1.000000	1.00000	6.800000	0.000000
25%	125.750000	308.000000	103.000000	2.000000	2.500000	3.00000	8.127500	0.000000
50%	250.500000	317.000000	107.000000	3.000000	3.500000	3.50000	8.560000	1.000000
75%	375.250000	325.000000	112.000000	4.000000	4.000000	4.00000	9.040000	1.000000
max	500.000000	340.000000	120.000000	5.000000	5.000000	5.00000	9.920000	1.000000

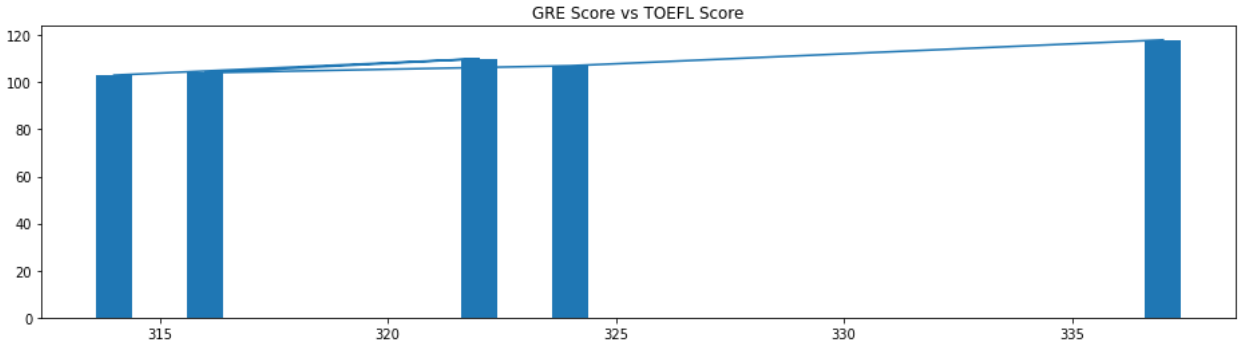
Renaming columns

```
df.rename(columns={"Serial No.":"Serial_No","GRE
Score":"GRE_Score","TOEFL Score":"TOEFL_Score","University
Rating":"University_Rating","Chance of Admit
":"Chance_of_Admit"},inplace=True)
```

```
df.columns
Index(['Serial_No', 'GRE_Score', 'TOEFL_Score',
'University_Rating', 'SOP',
       'LOR ', 'CGPA', 'Research'],


      dtype='object')
```

Comparison between GRE Score and TOEFL Score

```
x = df.GRE_Score

y = df.TOEFL_Score

plt.figure(figsize=(16,4))

plt.bar(x[:5],y[:5])

plt.plot(x[:5],y[:5])

plt.title('GRE Score vs TOEFL Score')


plt.savefig("output.jpg")
```



df.isnull().any()

```
Serial_No            False
GRE_Score            False
TOEFL_Score          False
University_Rating    False
SOP                  False
LOR                  False
CGPA                 False
Research             False

dtype: bool
```

Number of Students who have scored more than 300

df1 = df.where(df.GRE_Score>300)

df1.Serial_No.count()

```
447
```

Minimum GRE Score and TOEFL Score to get admission in 5 rating university

df1 = df.where(df.University_Rating==5)

print("Min GRE Score : ",df1.GRE_Score.min())

```
df1 = df.where(df.University_Rating==5)

print("Min TOEFL Score : ",df1.TOEFL_Score.min())
```

```
Min GRE Score :  303.0
Min TOEFL Score :  101.0
```

Minimum , Maximum , Median, Mean of the GRE and TOEFL Scores

```
print("Minimum GRE Score : ",df.GRE_Score.min())

print("Maximum GRE Score : ",df.GRE_Score.max())

print("Median GRE Score : ",df.GRE_Score.median())

print("Mean GRE Score : ",df.GRE_Score.mean())

print("**********************************")

print("Minimum TOEFL Score : ",df.TOEFL_Score.min())

print("Maximum TOEFL Score : ",df.TOEFL_Score.max())

print("Median TOEFL Score : ",df.TOEFL_Score.median())

print("Mean TOEFL Score : ",df.TOEFL_Score.mean())
```

```
Minimum GRE Score :  290
Maximum GRE Score :  340
Median GRE Score :  317.0
Mean GRE Score :  316.472
******************************
Minimum TOEFL Score :  92
Maximum TOEFL Score :  120
Median TOEFL Score :  107.0
Mean TOEFL Score :  107.192
```

df = df.drop('Serial No.',axis = 1)4

df.head()

GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit	
0	337	118	4	4.5	4.5	9.65	1	0.92
1	324	107	4	4.0	4.5	8.87	1	0.76
2	316	104	3	3.0	3.5	8.00	1	0.72
3	322	110	3	3.5	2.5	8.67	1	0.80
4	314	103	2	2.0	3.0	8.21	0	0.65

describe() method computes a summary of statistics like count, mean, standard deviation, min, max and quartile values pertaining to the DataFrame columns.

admissions.describe()

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
count	500.000000	500.000000	500.000000	500.000000	500.00000	500.000000	500.000000	500.00000
mean	316.472000	107.192000	3.114000	3.374000	3.48400	8.576440	0.560000	0.72174
std	11.295148	6.081868	1.143512	0.991004	0.92545	0.604813	0.496884	0.14114
min	290.000000	92.000000	1.000000	1.000000	1.00000	6.800000	0.000000	0.34000
25%	308.000000	103.000000	2.000000	2.500000	3.00000	8.127500	0.000000	0.63000
50%	317.000000	107.000000	3.000000	3.500000	3.50000	8.560000	1.000000	0.72000
75%	325.000000	112.000000	4.000000	4.000000	4.00000	9.040000	1.000000	0.82000
max	340.000000	120.000000	5.000000	5.000000	5.00000	9.920000	1.000000	0.97000

A **pairs plot** allows us to see both distribution of single variables and relationships between two variables. The pairs plot builds on two figures, the histogram and the scatter plot. The histogram allows us to see the distribution of a single variable while the scatter plots shows the relationship between two variables.

sns.pairplot(df)

The **heatmap** is a way of representing the data in a 2-dimensional form. The data values are represented as colors in the graph. The goal of the heatmap is to provide a colored visual summary of information.

```
corr = df.corr()
sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values)
```

I split the data into two different sets, one for the independent features — x, and one for the dependent variable — y (which is the last column). Next I split the dataset x into two separate sets — xTrain and xTest. Similarly, I splitted the dataset y into two sets as well — yTrain and yTest. The training set has 75% of data while test set has 25% of it.

random_state acts as the seed for the random number generator during the split.

```
X = df.drop('Chance of Admit ',axis = 1)
y = df['Chance of Admit ']

X_train,X_val,y_train,y_val = train_test_split(X,y,test_size = .25,random_state = 123)
```

I have used random forest algorithm for solving this regression problem. The random forest combines hundreds or thousands of decision trees, trains each one on a slightly different set of the observations, splitting nodes in each tree considering a limited number of the features. The final predictions of the random forest are made by averaging the predictions of each individual tree.

RFs train each tree independently, using a random sample of the data. This randomness helps to make the model more robust than a single decision tree, and less likely to overfit on the training data.

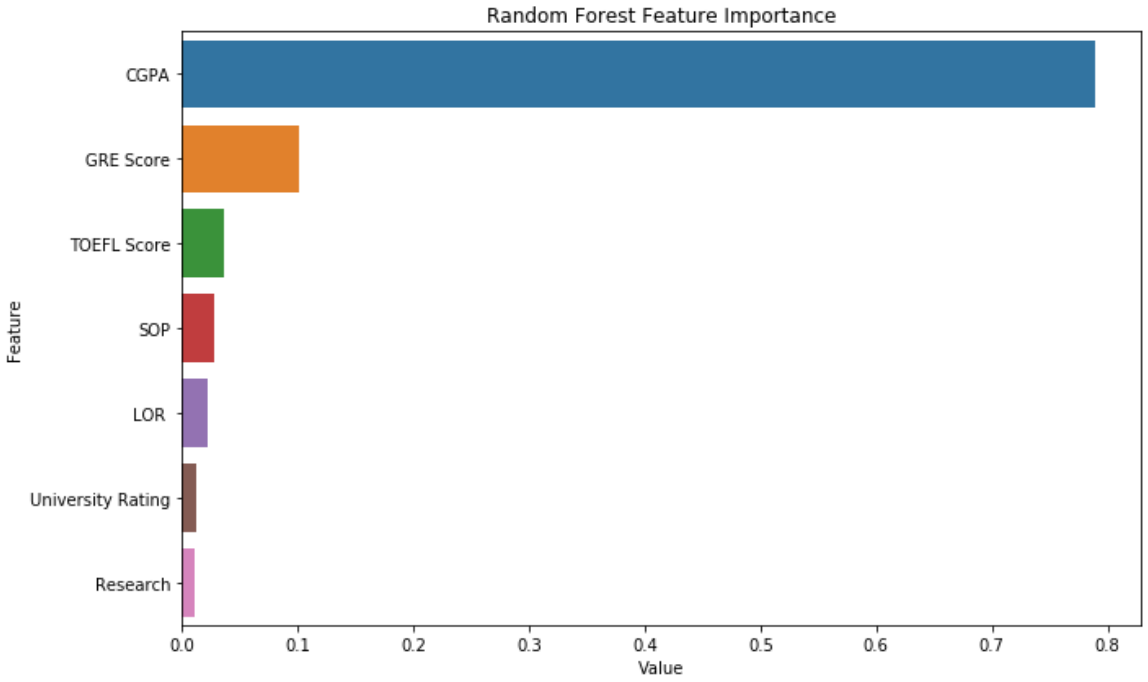
n_estimators is the number of trees to be used in the forest. Since Random Forest is an ensemble method comprising of creating multiple decision trees, this parameter is used to control the number of trees to be used in the process.

```
rf_model = RandomForestRegressor(n_estimators = 100,random_state = 42)
rf_model.fit(X_train,y_train)print('Mean absolute error for RF model: %0.4f'
%mean_absolute_error(y_val,rf_model.predict(X_val)))
```

The mean absolute error for the random forest model is 0.0421.

The final part was to plot the class wise feature importance. Clearly CGPA is the most important criteria for graduate admission followed by GRE and TOEFL score.

```
feature_importance = pd.DataFrame(sorted(zip(rf_model.feature_importances_,
X.columns)), columns=['Value','Feature']) plt.figure(figsize=(10, 6))
sns.barplot(x="Value", y="Feature",
data=feature_importance.sort_values(by="Value", ascending=False))
```



Conclusions

As a quick summary, I used random forest algorithm to visualize the importance of each features for graduate admission. This could be of great help for students preparing for their higher studies.

This article was written to show how Machine Learning could be used to calculate probabilities for admission and does not attempt to actually get the results right, since the data used is not enough for it (or maybe the event itself is simply not predictable). Please, take this just as a tutorial, just because it's a cool and up to date subject. A much deeper approach should've been used to get better results and make them meaningful.