CHIKKANNA GOVERNMENT ARTS COLLEGE

DEPARTMENT OF BACHELOR OF COMPUTER APPLICATION

TIRUPUR-641602

(AFFILIATED TO BHARATHIAR UNIVERSITY)

TEAM MEMBERS NAME :

YOGESH G (2022J0060)

GOKUL S (2022J0028)

BOOPATHY K (2022J0023)

ASHOK KUMAR M (2022J0018)

Intelligent Admissions: The Future of University Decision Making with Machine Learning

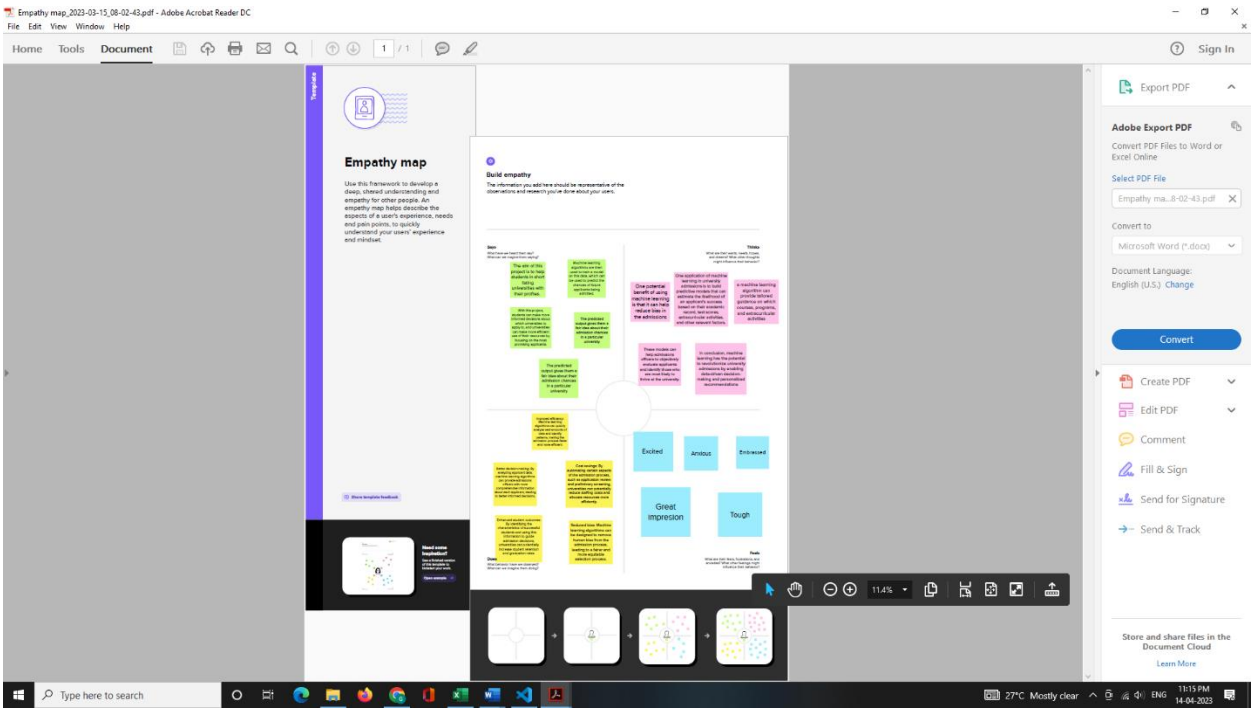
1.INTRODUCTION:

University admission is the process by which students are selected to attend a college or university. The process typically involves several steps, including submitting an application, taking entrance exams, and participating in interviews or other evaluations.
Students are often worried about their chances of admission in University. the university admission process for students can be demanding, but by being well-informed, prepared, and organized, students
can increase their chances of being admitted to the university of their choice.

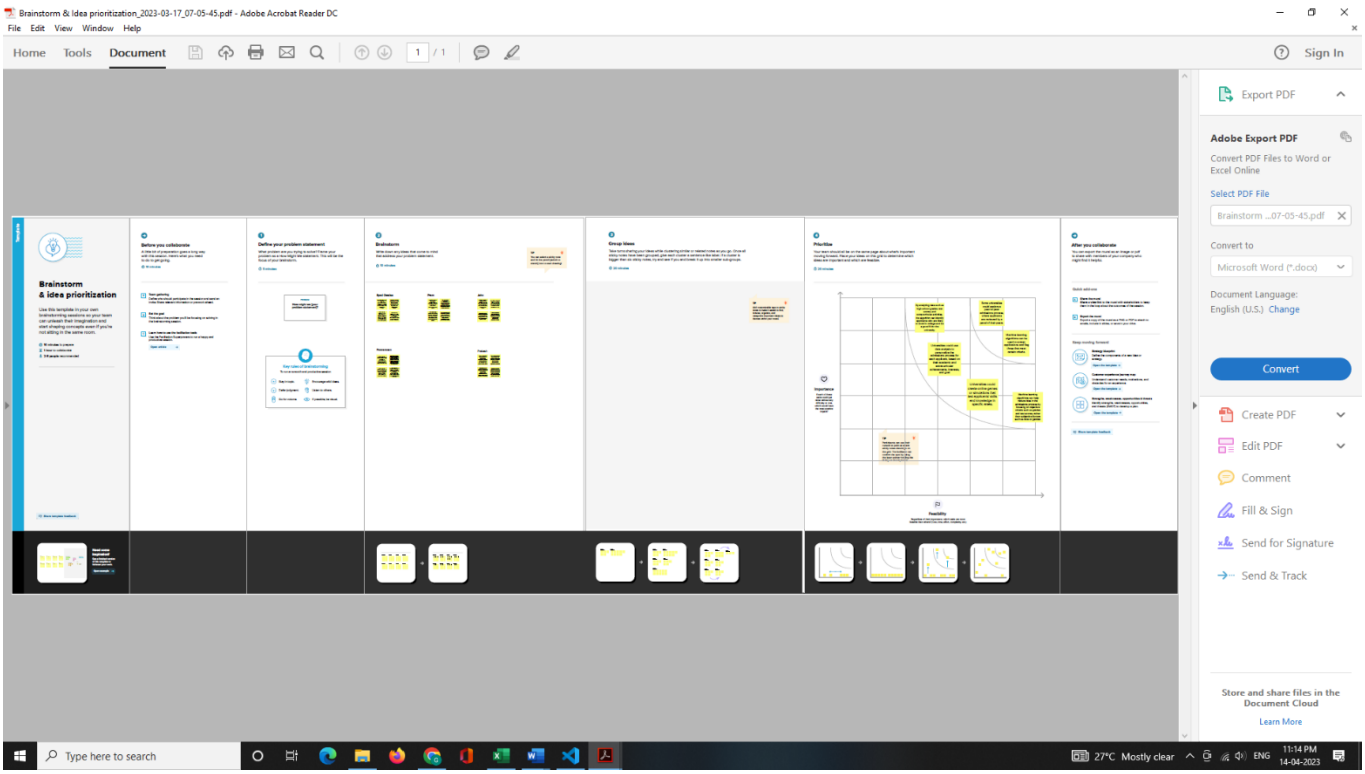
The aim of this project is to help students in short listing universities with their profiles. Machine learning algorithms are then used to train a model on this data, which can be used to predict the chances of future applicants being admitted. With this project, students can make more informed decisions about which universities to apply to, and universities can make more efficient use of their resources by focusing on the most promising applicants.The predicted output gives them a fair idea about their admission chances in a particular university. This analysis should also help students who are currently preparing or will be preparing to get a better idea.

2. PROBLEM DEFINITION & DESIGN THINKING

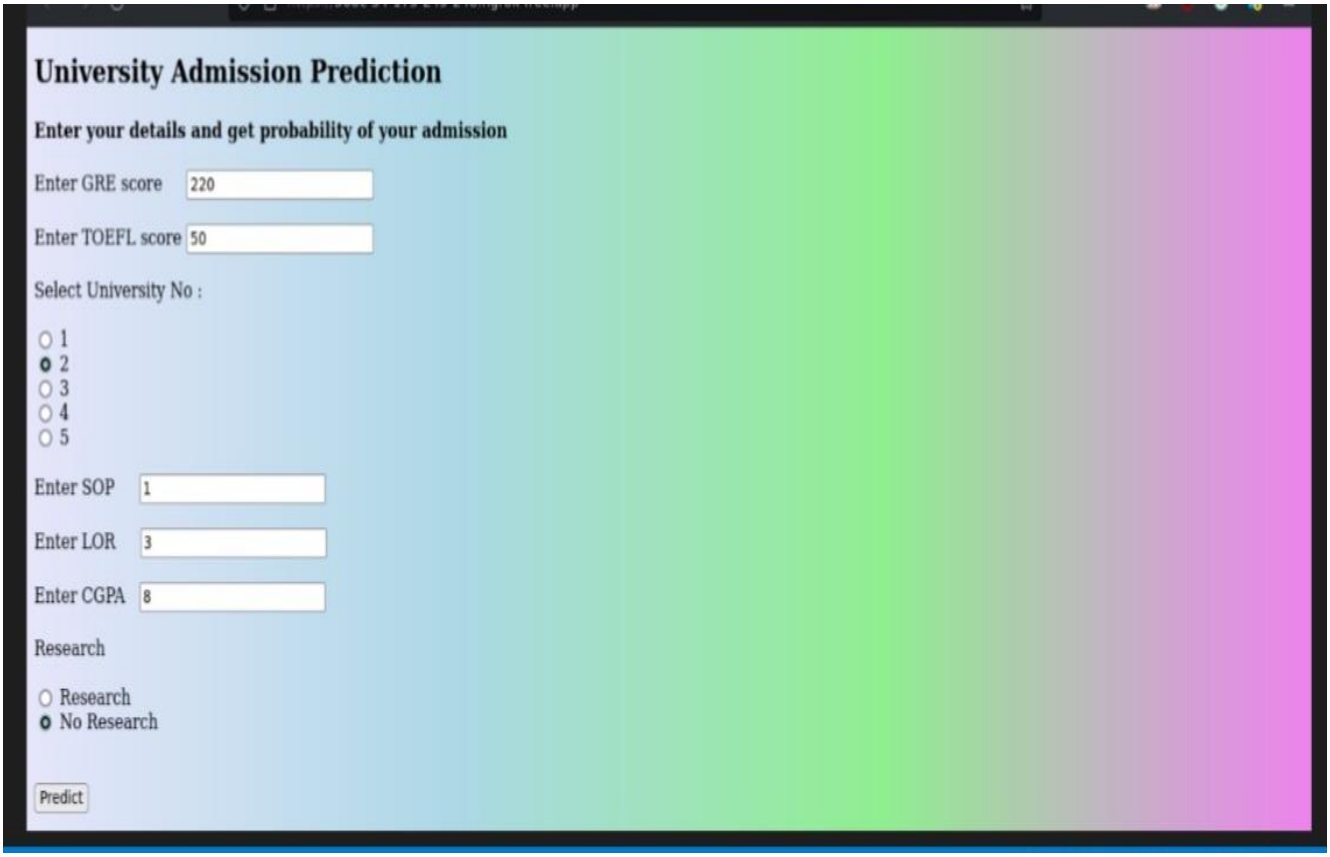
2.1. EMPATHY MAP



2.2 IDEATION & BRAINSTORMINGS MAP



3. RESULT


University Admission Prediction
Enter your details and get probability of your admission
Enter GRE score 220
Enter TOEFL score 50
Select University No :
1
2
3
4
5
Enter SOP 1
Enter LOR 3
Enter CGPA 8
Research
Research
No Research
Predict

4.ADVANTAGES & DISADVANTAGES

ADVANTAGES :

- Quick Identification of Trends and Patterns
- It is automatic and does not require human interference
- Continual Development
- Adoption by Multi-industries

DISADVANTAGES :

Despite its many benefits and growing popularity, machine learning isn't flawless. There are a few disadvantages of machine learning that constrain its functioning. Let us analyze them below in detail.

- Data Acquisition
- Time and Resources Intensive
- Chances of faulty Interpretation of data
- Requirement of more Space
- It can cause blood clots.

5. APPLICATION

Machine Learning has an additional benefit of **processing large chunks of data that is sometimes tiresome for men to do and eventually lead to a failure in making the right decision**. It is easily adaptable to new and complex data. After processing the data, it is capable of analyzing any flaws or errors

6. CONCLUSION

1. Identify the admission criteria: Determine the factors that will be considered for admission to the university, such as academic qualifications, extracurricular activities, personal statement, recommendation

2. Gather data: Collect data on previous admission cycles, including applications, admissions, and demographics of admitted students. This data will be used to train the AI model

3. Select an AI model: Choose the appropriate AI model for the admission system, such as a decision tree, a neural network, or a random forest model

4. Prepare the data: Clean and preprocess the data to ensure that it is accurate, complete, and consistent. This may involve removing duplicates, filling in missing values, and transforming data into a suitable format for the AI model.

5. Train the model: Use the prepared data to train the AI model using a suitable algorithm. The model should be optimized for accuracy, precision, and recall.

6. Integrate the model into the admission process: Once the model has been tested and validated, integrate it into the admission process. This may involve creating a user interface for admissions officers to interact with the model, or automating the decision-making process entirely

7. FUTURE SCOPE

The success of any business is built on sound decision-making. But reaching reliable conclusions is now tougher than ever. In addition to having to process vast amounts of data, today's decision-makers need to consider potential enterprise-wide impacts. And time pressure or lack of information often result in decisions based on habits, bias, and heuristics. Now there's a powerful toolkit that supports more effective and accurate decision-making. Its name? Decision intelligence.

By drawing together these diverse domains, this approach helps companies enhance their ability to process and understand vast volumes of data – allowing them to gain greater overall insight, see their business decisions in a much broader context, and review potential impacts across and beyond the organization.

In a nutshell, decision intelligence enables companies to harness AI and data to rapidly make accurate, consistent decisions and zoom in on specific needs and issues in their business. It allows them to gather data and use machine learning to model it in order to accurately predict outcomes for optimal commercial decision-making.

8. APPENDIX

A.SOURCE CODE :

```
#import necessary libraries
import pandas as pd
import numpy as np
import pickle
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import sklearn
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import  RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix, f1_score
```

```
#read_csv is a pandas function to read csv files
data =
pd.read_csv(r"C:\Users\Shivani_SB\OneDrive\Desktop\Projects\10.University_Admissi
on_Prediction-main\10.University_Admission_Prediction-
main\Dataset\Admission_Predict.csv")
```

```
#head() method is used to return top n (5 by default) rows of a DataFrame or
series.
data.head(8)
```

```
#let us drop Serial No. Column as it is not required for prediction
data.drop(["Serial No."],axis=1,inplace=True)
data.head()
```

```
data.describe()
```

```
data.info()
```

```
#Let us rename the column Chance of Admit because it has trainling space
data=data.rename(columns = {'Chance of Admit ':'Chance of Admit'})
```

```
data.isnull().any()
```

```
data.corr()
```

```
plt.figure(figsize=(10,7))

sns.heatmap(data.corr(),annot=True,cmap="RdYlGn")
```

```
sns.pairplot(data=data,hue='Research',markers=["^", "v"],palette='inferno')
```

```
sns.scatterplot(x='University Rating',y='CGPA',data=data,color='Red', s=100)
```

```
category = ['GRE Score','TOEFL Score','University Rating','SOP','LOR
','CGPA','Research','Chance of Admit']
color =
['yellowgreen','gold','lightskyblue','pink','red','purple','orange','gray']
start = True
for i in np.arange(4):
    fig = plt.figure(figsize=(14,8))
    plt.subplot2grid((4,2),(i,0))
    data[category[2*i]].hist(color=color[2*i],bins=10)
    plt.title(category[2*i])
    plt.subplot2grid((4,2),(i,1))
    data[category[2*i+1]].hist(color=color[2*i+1],bins=10)
    plt.title(category[2*i+1])

plt.subplots_adjust(hspace = 0.7, wspace = 0.2)
plt.show()
```

```
print('Mean CGPA Score is :',int(data['CGPA'].mean()))
print('Mean GRE Score is :',int(data['GRE Score'].mean()))
print('Mean TOEFL Score is :',int(data['TOEFL Score'].mean()))
#print('Mean University rating is :',int(data[data['University
Rating']<=500].University Rating.mean()))
```

```
data.head()
```

```
x=data.iloc[:,0:-1].values
x
```

```
y=data['Chance of Admit'].values
y
```

```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler()
x=sc.fit_transform(x)
x
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,
test_size=0.20,random_state=42)
#random_state acts as the seed for the random number generator during the split
```

```
y_train.shape
```

```
x_train
```

```
### Let us convert it into classification problem
chance of admit>0.5 as true
chance of admit<0.5 as false
```

```
y_train=(y_train>0.5)
y_train
```

```
y_test=(y_test>0.5)
```

```
y_test
```

```
#Model building - Logistic Regression
def logreg(x_train,x_test,y_train,y_test):
    lr = LogisticRegression(random_state=0)
    lr.fit(x_train,y_train)
    y_lr_tr = lr.predict(x_train)
    print(accuracy_score(y_lr_tr,y_train))
    yPred_lr = lr.predict(x_test)
    print(accuracy_score(yPred_lr,y_test))
    print("***Logistic Regression***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_lr))
    print("Classification Report")
    print(classification_report(y_test,yPred_lr))
```

```
#printing the train accuracy and test accuracy respectively
logreg(x_train,x_test,y_train,y_test)
```

```
#testing on test & random input values
lr = LogisticRegression(random_state=0)
lr.fit(x_train,y_train)
print("Predicting on test values")
lr_pred =lr.predict(x_test)
print("output is: ",lr_pred)
print("Predicting on random input")
lr_pred_own = lr.predict(sc.transform([[337,118,4,4.5,4.5,9.65,1]]))
print("output is: ",lr_pred_own)
```

```
#Model building - Decision Tree Classifier
def decisionTree(x_train,x_test,y_train,y_test):
    dtc = DecisionTreeClassifier(criterion="entropy",random_state=0)
    dtc.fit(x_train,y_train)
    y_dt_tr = dtc.predict(x_train)
    print(accuracy_score(y_dt_tr,y_train))
    yPred_dt = dtc.predict(x_test)
    print(accuracy_score(yPred_dt,y_test))
    print("***Decision Tree***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_dt))
    print("Classification Report")
    print(classification_report(y_test,yPred_dt))
```

```
#printing the train accuracy and test accuracy respectively
decisionTree(x_train,x_test,y_train,y_test)
```

```
#testing on test & random input values
dtc = DecisionTreeClassifier(criterion="entropy",random_state=0)
dtc.fit(x_train,y_train)
print("Predicting on test values")
dtc_pred =dtc.predict(x_test)
print("output is: ",dtc_pred)
print("Predicting on random input")
dtc_pred_own = dtc.predict(sc.transform([[337,118,4,4.5,4.5,9.65,1]]))
print("output is: ",dtc_pred_own)
```

```
#Model building - Random Forest Classifier
def RandomForest(x_tarin,x_test,y_train,y_test):
    rf =
RandomForestClassifier(criterion="entropy",n_estimators=10,random_state=0)
    rf.fit(x_train,y_train)
    y_rf_tr = rf.predict(x_train)
    print(accuracy_score(y_rf_tr,y_train))
    yPred_rf = rf.predict(x_test)
    print(accuracy_score(yPred_rf,y_test))
    print("***Random Forest***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_rf))
    print("Classification Report")
    print(classification_report(y_test,yPred_rf))
```

```
#printing the train accuracy and test accuracy respectively
RandomForest(x_train,x_test,y_train,y_test)
```

```
#testing on test & random input values
rf = RandomForestClassifier(criterion="entropy",n_estimators=10,random_state=0)
rf.fit(x_train,y_train)
print("Predicting on test values")
rf_pred =rf.predict(x_test)
print("output is: ",rf_pred)
print("Predicting on random input")
rf_pred_own = rf.predict(sc.transform([[337,118,4,4.5,4.5,9.65,1]]))
print("output is: ",rf_pred_own)
```

```
# Importing the Keras libraries and packages
import keras
from keras.models import Sequential
from keras.layers import Dense
```

```
# Initialising the ANN
classifier = Sequential()
```

```
# Adding the input layer and the first hidden layer
classifier.add(Dense(units=7, activation='relu', input_dim=7))
```

```
# Adding the second hidden layer
classifier.add(Dense(units=7, activation='relu'))
```

```
# Adding the output layer
classifier.add(Dense(units=1, activation='linear'))
```

```
# Compiling the ANN
classifier.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

```
# Fitting the ANN to the Training set
model = classifier.fit(x_train, y_train, batch_size=10, validation_split=0.33,
epochs=20)
```

```
ann_pred = classifier.predict(x_test)
ann_pred = (ann_pred>0.5)
print(accuracy_score(ann_pred,y_test))
print("***ANN Model***")
print("Confusion_Matrix")
print(confusion_matrix(y_test,ann_pred))
print("Classification Report")
print(classification_report(y_test,ann_pred))
```

```
#testing on test & random input values
print("Predicting on test input")
ann_pred = classifier.predict(x_test)
ann_pred = (ann_pred>0.5)
print("output is: ",ann_pred)
print("Predicting on random input")
ann_pred_own = classifier.predict(sc.transform([[337,118,4,4.5,4.5,9.65,1]]))
ann_pred_own = (ann_pred_own>0.5)
print("output is: ",ann_pred_own)
```

```
ann_pred_train = classifier.predict(x_train)
ann_pred_train = (ann_pred_train>0.5)
print(accuracy_score(ann_pred_train,y_train))
print("***ANN Model***")
print("Confusion_Matrix")
print(confusion_matrix(ann_pred_train,y_train))
print("Classification Report")
print(classification_report(ann_pred_train,y_train))
```

```
pickle.dump(lr,open('university.pkl','wb'))
```

OUTPUT:


Admission Prediction
You have a chance


University Admission Prediction
Enter your details and get probability of your admission
Enter GRE score 220
Enter TOEFL score 50
Select University No :
1
2
3
4
5
Enter SOP 1
Enter LOR 3
Enter CGPA 8
Research
Research
No Research
Predict