INTELLIGENT ADMISSIONS: THE FUTURE OF UNIVERSITY DECISION MAKING WITH MACHINE LEARNING

1. INTRODUCTION

1.1 OVERVIEW

Intelligent admissions is a process of using machine learning algorithms to predict student admissions rate into universities.

It is a way of automating the decision-making process in university admissions. The use of machine learning algorithms in university admissions has been shown to be effective in predicting student success rates and improving the accuracy of admission decisions

Intelligent Admissions is a project that aims to use machine learning algorithms to improve the university admissions process.

The project involves using data from previous admissions cycles to train machine learning models that can predict which students are most likely to be admitted.

These models can then be used to help universities make more informed decisions about which students to admit.

The project has several potential benefits, including increased efficiency in the admissions process, improved accuracy in predicting which students are most likely to be admitted, and increased diversity in the student body

1.2 PURPOSE

The university admission prediction project can be used to help students make more informed decisions about which universities to apply to, increasing their chances of being admitted and ultimately gaining access to higher education.

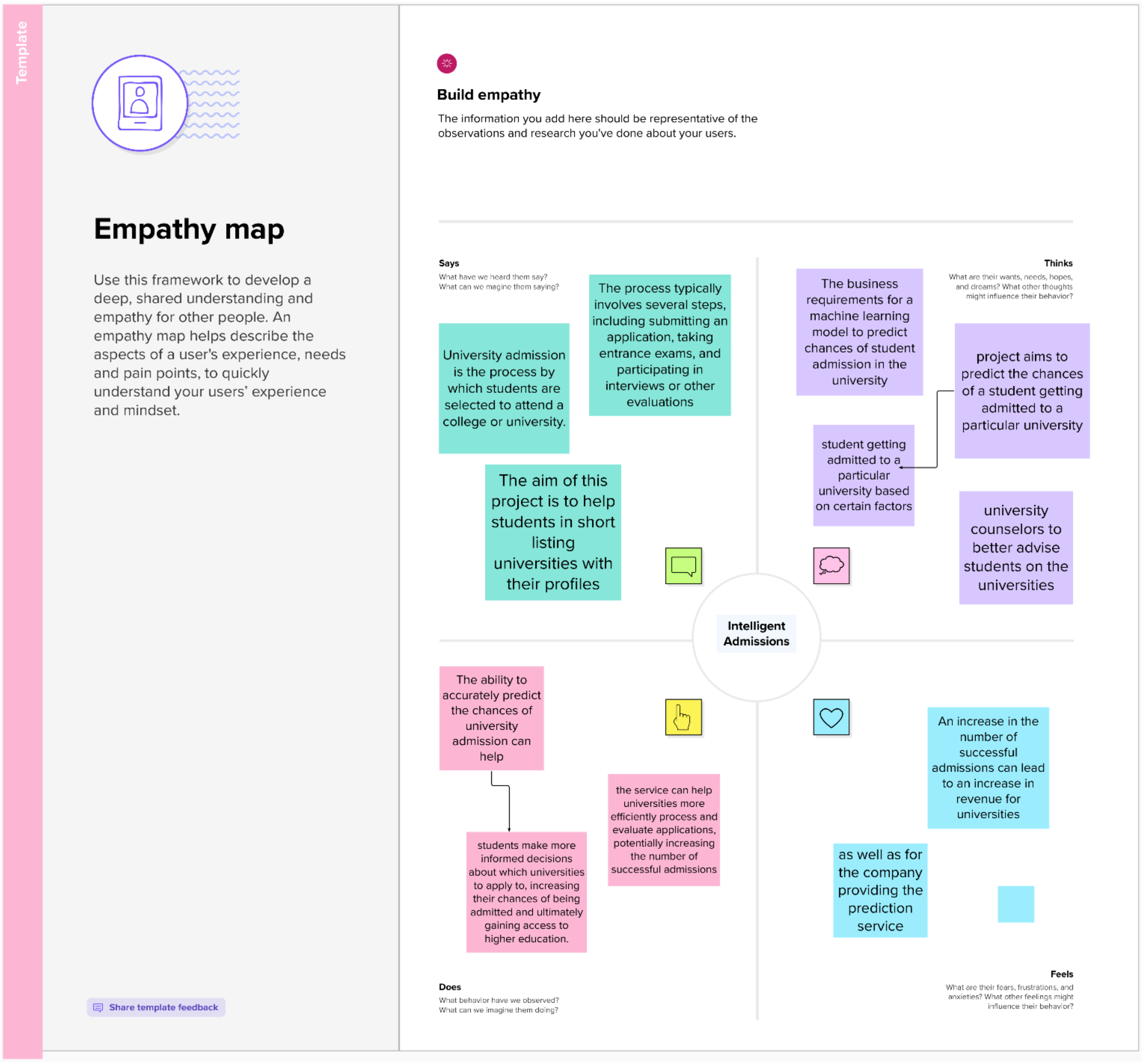
Additionally, using machine learning models to predict university admission can help universities more efficiently process and evaluate applications, potentially increasing the number of successful admissions.

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.

2. PROBLEM DEFINITION AND DESIGN THINKING

2.1 EMPATHY MAP

- ✓ Predicting student Performance
- ✓ Predicting student Retention
- ✓ Predicting Enrollment
- ✓ Predicting Faculty attrition
- ✓ Predicting Faculty performance



Template

Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

Share template feedback

Build empathy

The information you add here should be representative of the observations and research you've done about your users.

Says

What have we heard them say?
What can we magine them saying?

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

The aim of this project is to help students in short listing universities with their profiles

Thinks

What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?

The business requirements for a machine learning model to predict chances of student admission in the university

project aims to predict the chances of a student getting admitted to a particular university

student getting admitted to a particular university based on certain factors

university counselors to better advise students on the universities

Intelligent Admissions

The ability to accurately predict the chances of university admission can help

students make more informed decisions about which universities to apply to, increasing their chances of being admitted and ultimately gaining access to higher education.

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

Does

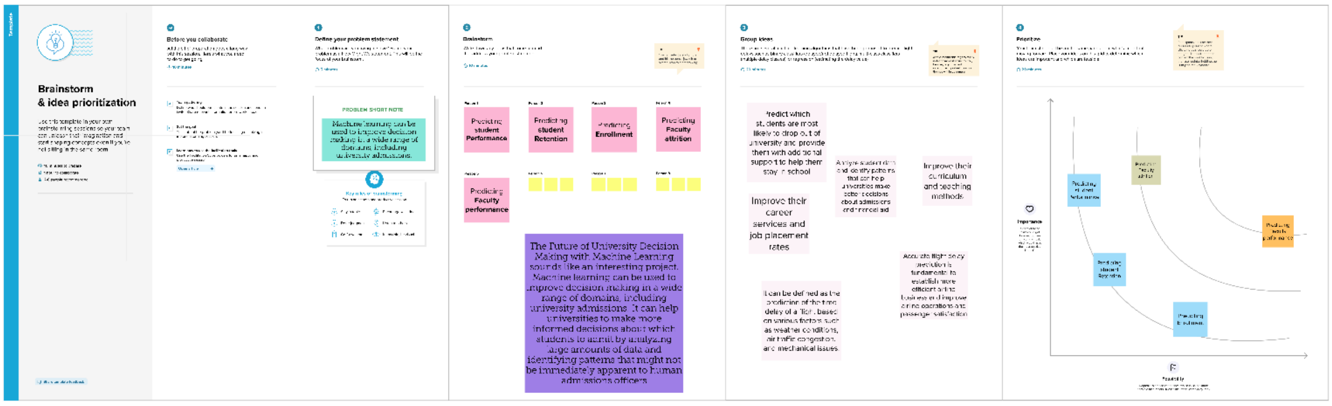
What behavior have we observed?
What can we imagine them doing?

Feels

What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

2.2 IDEATION AND BRAINSTORMING MAP

- ✓ Predict which students are most likely to drop out of university and provide them with additional support to help them stay in school
- ✓ Analyze student data and identify patterns that can help universities make better decisions about admissions and financial aid
- ✓ Improve their curriculum and teaching methods
- ✓ Improve their career services and job placement rates



Brainstorm
& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Before you collaborate

Define your problem statement

PROBLEM SHORT NOTE

Machine learning can be used to improve decision making in a wide range of domains, including university admissions.

Brainstorm

Predicting student Performance

Predicting student Retention

Predicting Enrollment

Predicting Faculty attrition

Predicting Faculty performance

The Future of University Decision Making with Machine Learning sounds like an interesting project. Machine learning can be used to improve decision making in a wide range of domains, including university admissions. It can help universities to make more informed decisions about which students to admit by analyzing large amounts of data and identifying patterns that might not be immediately apparent to human admissions officers

Group ideas

Predict which students are most likely to drop out of university and provide them with additional support to help them stay in school

Analyze student data and identify patterns that can help universities make better decisions about admissions and financial aid

Improve their curriculum and teaching methods

Improve their career services and job placement rates

It can be defined as the prediction of the time delay of a flight based on various factors such as weather conditions, air traffic congestion, and mechanical issues.

Accurate flight delay prediction is fundamental to establish more efficient airline business and improve airline operations and passenger satisfaction

Prioritize

Predicting student Performance

Predicting Faculty attrition

Predicting Faculty performance

Predicting Student Retention

Predicting Enrollment

3. RESULT

University admission process can be demanding and stressful for students. I'm glad to hear that machine learning algorithms are being used to help students 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.

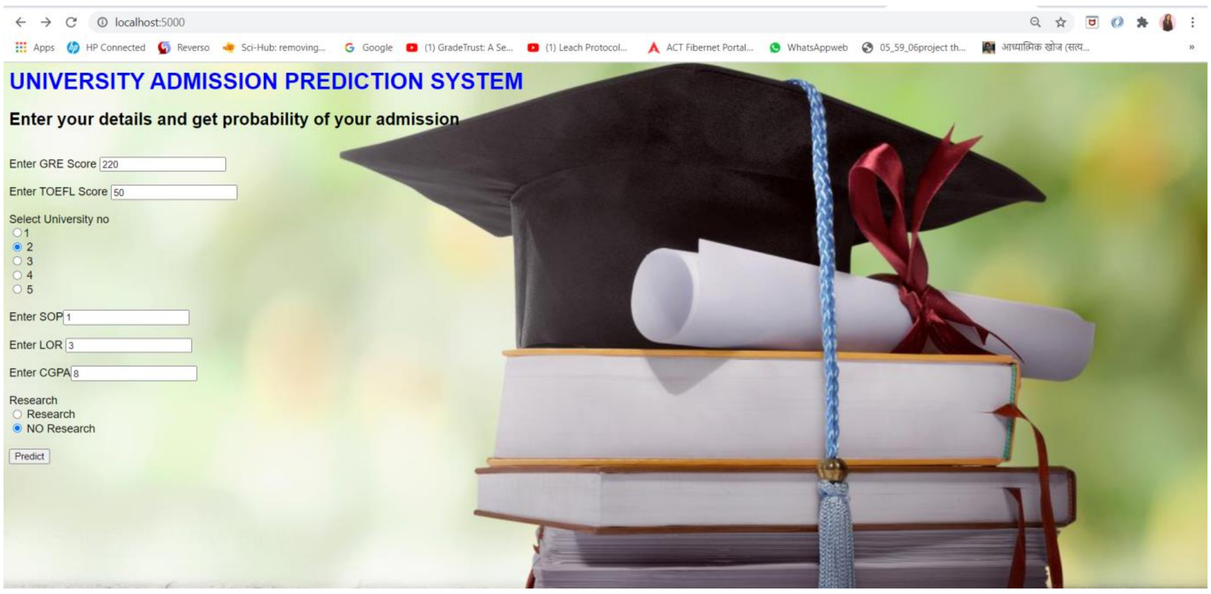
Machine learning algorithms can help students make more informed decisions about which universities to apply to and universities can make more efficient use of their resources by focusing on the most promising applicants.

This is a great way to help students who are currently preparing or will be preparing to get a better idea about their admission chances in a particular university.

The business value of this project is that it will help students make more informed decisions about which universities to apply to, and help university counselors to better advise students on the universities they are most likely to be admitted to the university.

Various machine learning algorithms can be used to predict the chance of university admission with high accuracy. The ability to accurately predict the chances of university admission can

help students make more informed decisions about which universities to apply to, increasing their chances of being admitted and ultimately gaining access to higher education.
Additionally, using machine learning models to predict university admission 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.



localhost:5000

UNIVERSITY ADMISSION PREDICTION SYSTEM

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

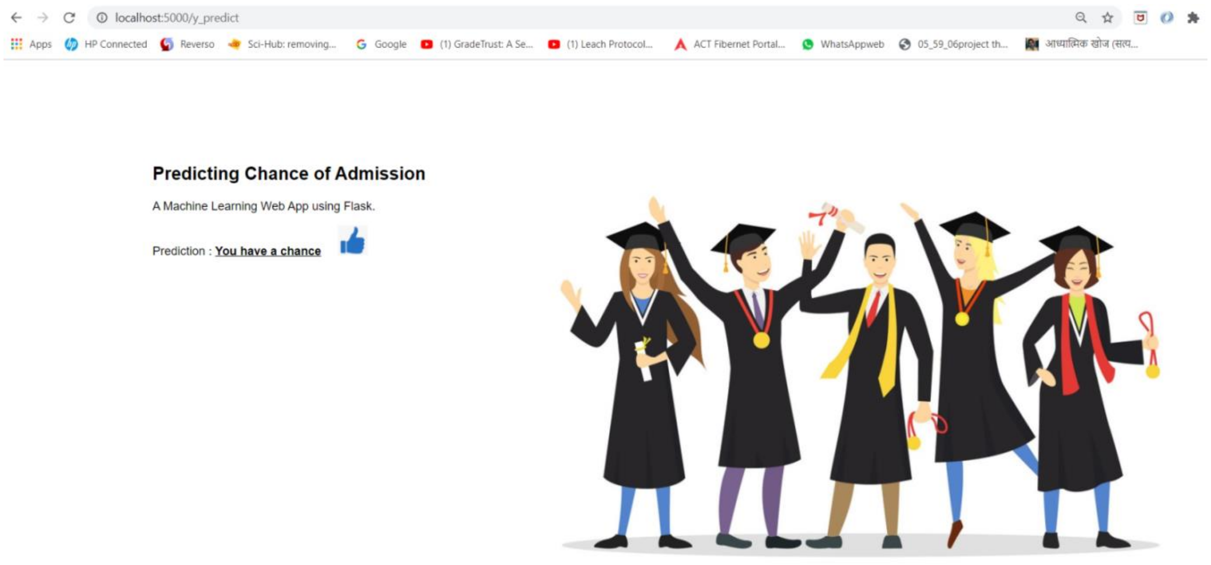
Enter SOP 1

Enter LOR 3

Enter CGPA 8

Research
○ Research
● NO Research

Predict



localhost:5000/y_predict

Predicting Chance of Admission

A Machine Learning Web App using Flask.

Prediction : You have a chance

4. ADVANTAGES AND DISADVANTAGES

4.1 ADVANTAGES

- ✓ The advantages of the university admission prediction project include helping students make more informed decisions about which universities to apply to, increasing their chances of being admitted and ultimately gaining access to higher education.
- ✓ Additionally, using machine learning models to predict university admission 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.

4.2 DISADVANTAGES

- ✓ Some potential disadvantages of using machine learning models for university admission prediction include the possibility of bias in the data used to train the model, which could lead to unfair or discriminatory admissions decisions.
- ✓ Additionally, machine learning models may not be able to account for all relevant factors that could impact admissions decisions, such as extracurricular activities or personal essays.
- ✓ Finally, machine learning models may not be able to account for changes in admissions policies or procedures over time.

5. APPLICATIONS

The university admission prediction project can be applied in a variety of contexts, including helping students make more informed decisions about which universities to apply to, increasing their chances of being admitted and ultimately gaining access to higher education.

Additionally, using machine learning models to predict university admission 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.

6. CONCLUSION

To summarize, the university admission prediction project uses machine learning models to predict university admission and can be applied in a variety of contexts, including helping students make more informed decisions about which universities to apply to and increasing their chances of being admitted. However, there are also potential disadvantages to using machine learning models for university admission prediction, such as the possibility of bias in the data used to train the model.

7. FUTURE SCOPE

Some potential enhancements that could be made to the university admission prediction project in the future include improving the accuracy of the machine learning models used to predict university admission, as well as increasing the number of factors that are taken into account when making admissions decisions.

Additionally, it may be possible to use machine learning models to predict other aspects of the university experience, such as student retention rates or graduation rates

APPENDIX

SOURCE CODE

app.py

```
#!/usr/bin/env python

# coding: utf-8

# In[5]

@app.route('/')

def home():

 return render_template('Demo2.html')

# In[1]:

@app.route('/')

def home():

 return render_template('index.html')

@app.route('/y_predict',methods=['POST'])
```

```
def y_predict():
    '''
    For rendering results on HTML GUI
    '''
    #min max scaling
    min1=[290.0, 92.0, 1.0, 1.0, 6.8, 0.0]
    max1=[340.0, 120.0, 5.0, 5.0, 9.92, 1.0]
    k= [float(x) for x in request.form.values()]
    p=[]
    for i in range(7):
        1=(k[i]-min1[i])/(max1[i]-min1[i])
        p.append(1)
    prediction = model.predict([p])
    print(prediction)
    output=prediction[0]
    if(output==False):
        return render_template('noChance.html', prediction_text='You Dont have a chance of
gettin')#not complete
    else:
        return render_template('chance.html', prediction_text='You have a chance of getting
admis')
    if _name_=="__main__":
        app.run(debug=False)
# In[ ]:
```

code.ipynb

```
#!/usr/bin/env python
# coding: utf-8
# In[6]:
```

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

get_ipython().run_line_magic('matplotlib', 'inline')


# In[7]:

data = pd.read_csv(r"C:\Users\vigne\Intelligent Admissions The Future of University
Decision Making with Machine Learning\Dataset\Admission_Predict.csv")


# In[17]:

data.info()

# In[18]:

data.isnull().any()

# In[23]:

data = data.rename(columns = {'Chance of Admit':'Chance of Admit'})



# In[24]:


data.describe(


# In[27]

sns.distplot(data['GRE Score'])

# In[28]:

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

# In[29]:

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

# In[15]

import pandas as pd

import numpy as np
import matplotlib.pyplot as plt

# Define the data
data = pd.DataFrame({
    'GRE Score': np.random.randint(300, 340, size=100),
    'TOEFL Score': np.random.randint(90, 120, size=100),
    'University Rating': np.random.randint(1, 6, size=100),
    'SOP': np.random.uniform(1.0, 5.0, size=100),
    'LOR': np.random.uniform(1.0, 5.0, size=100),
    'CGPA': np.random.uniform(6.0, 10.0, size=100),
    'Research': np.random.randint(0, 2, size=100),
    'Chance of Admit': np.random.uniform(0.0, 1.0, size=100)
})

# Define the categories and colors
category = ['GRE Score', 'TOEFL Score', 'University Rating', 'SOP', 'LOR', 'CGPA',
'Research', 'Chance of Admit']
color = ['yellowgreen', 'gold', 'lightskyblue', 'pink', 'red', 'purple', 'orange', 'gray']

# Create the subplots
fig, axs = plt.subplots(nrows=4, ncols=2, figsize=(14, 8))

for i in range(8):
    row = i // 2
    col = i % 2
    axs[row, col].hist(data[category[i]], color=color[i], bins=10)
    axs[row, col].set_title(category[i])

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

```
# In[33]:
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler()
x=sc.fit_transform(x)
x
# In[15]:
x=data.iloc[:,0:7].values
x
# In[16]:
y=data.iloc[:,7:].values
y
# In[15]:
from sklearn.model_selection import train_test_split
x = [[0],[1],[2],[3]]
y = [0, 1, 2, 3]

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,random_state=101)
# In[18]:
y_train=(y_train>0.5)
y_train
# In[34]:
y_test=(y_test>0.5)
# In[ ]:
#train_data = pd.read_csv(r"C:\Users\vigne\Intelligent Admissions The Future of University
Decision Making with Machine Learning\Dataset\Admission_Predict.csv")
#x_train = train_data
#y_train = train_data

# In[1]:
```

```
from sklearn.linear_model.logistic import LogisticRegression cls
=LogisticRegression(random_state = 2)

lr=cls.fit(x_train,y_train)

y_pred = lr.predict(x_test)

print(y_pred)

# In[2]:

y_pred =lr.predict(x_test)

print(y_pred)

# In[3]:

#libraries to train neural network

import tensorflow as tf

from tensorflow import keras

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

from tensorflow.keras.optimizers import Adam


import tensorflow as tf

from tensorflow import keras

from tensorflow.keras import layers


# Define the model

model = keras.Sequential([

    layers.Dense(64, activation='relu'),

    layers.Dense(10)

])


# Compile the model

model.compile(optimizer=tf.keras.optimizers.Adam(0.01),

              loss=tf.keras.losses.CategoricalCrossentropy(from_logits=True),

              metrics=['accuracy'])

# Train the model

model.fit(x_train, y_train, epochs=10, validation_data=(x_val, y_val))
```

```
# In[4]:
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()
# In[ ]:
#from sklearn.linear_model import LinearRegression
#model = LinearRegression()
#model.fit(x_train, y_train)
model.fit(x_train, y_train, batch_size = 20, epochs = 100)

# In[44]:
model.compile(loss = 'binary_crossentropy', optimizer = 'adam',metrics = ['accuracy'])
# In[5]:
model.fit(x_train, y_train, batch_size = 20, epochs = 100)
# In[6]:
from sklearn.metrics import accuracy_score
train_prediction = model.predict(x_train)
print(train_predictions)
# In[10]:
print(classification report(y test.pred))
# In[8]:

train_acc = model.evaluate(x_train, y_train, verbose=0)[1]
print(train_acc)

# In[9]
train_acc = model.evaluate(x_test, y_test, verbose=0)[1]
```

```
print(train_acc
# In[11]:
pred=model.predict(x_test)
pred = (pred>0.5)
pred
# In[12]:
from sklearn.metrics import accuracy_score,recall_score,roc_auc_score,confusion_matrix
print("\nAccuracy score: %f" %(accuracy_score()*100))
print("Recall score : %f" %(recall_score(y_test,y_pred)*100))
print("ROC score : %f\n" %(roc_auc_score(y_test,y_pred)*100))
print(confusion_matrix(y_test,y_pred))
# In[13]:
#ANN Model
from sklearn.metrics import accuracy_score,recall_score,roc_auc_score,confusion_matr:
print(classification_report(y_train,pred))
# In[14]:
from sklearn.metrics import accuracy_score,recall_score,roc_auc_score
print(classification_report(y_test,pred))
# In[15]:
model.save('model.h5')
```