RAJSHAHI UNIVERSITY OF ENGINEERING & TECHNOLOGY
DEPARTMENT OF ELECTRONICS & TELECOMMUNICATION ENGINEERING,
RAJSHAHI-6204, BANGLAGLADESH.

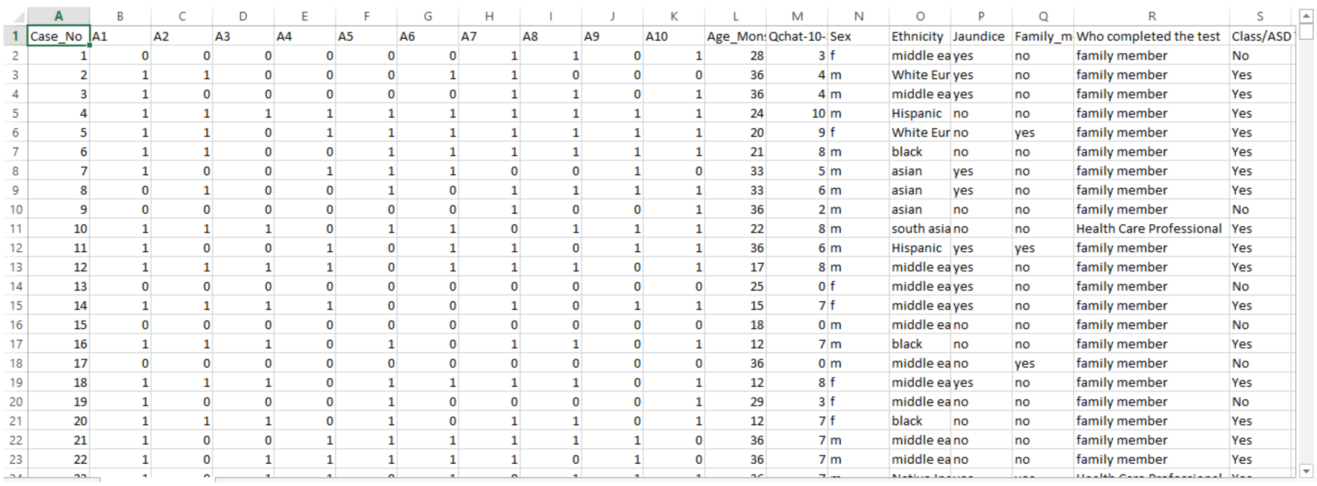
REPORT BASED ON AUTISM DETECTION MODEL

SUBMITTED TO:
MD. Rakib Hossain
Assistant Professor
Dept. of ETE, RUET

SUBMITTED BY:
Name: Minhazul Islam
Roll: 1804038
Year: 3rd Year Even Semester

Autism For Toddlers Detection Model

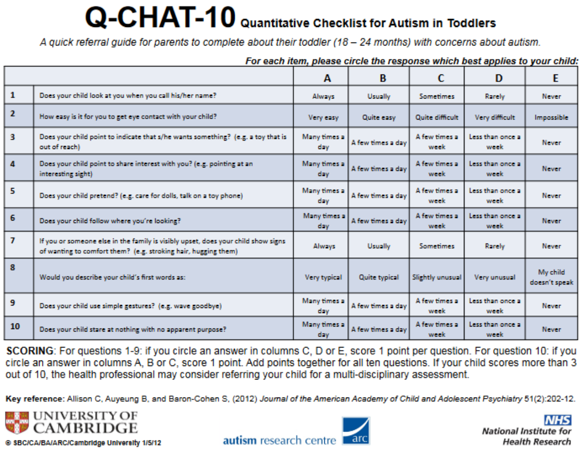
The Dataset:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	Case_No	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	Age_Mon:	Qchat-10-	Sex	Ethnicity	Jaundice	Family_m	Who completed the test	Class/ASD
2	1	0	0	0	0	0	0	1	1	0	1	28	3	f	middle ea	yes	no	family member	No
3	2	1	1	0	0	0	1	1	0	0	0	36	4	m	White Eur	yes	no	family member	Yes
4	3	1	0	0	0	0	0	1	1	0	1	36	4	m	middle ea	yes	no	family member	Yes
5	4	1	1	1	1	1	1	1	1	1	1	24	10	m	Hispanic	no	no	family member	Yes
6	5	1	1	0	1	1	1	1	1	1	1	20	9	f	White Eur	no	yes	family member	Yes
7	6	1	1	0	0	1	1	1	1	1	1	21	8	m	black	no	no	family member	Yes
8	7	1	0	0	1	1	1	0	0	1	0	33	5	m	asian	yes	no	family member	Yes
9	8	0	1	0	0	1	0	1	1	1	1	33	6	m	asian	yes	no	family member	Yes
10	9	0	0	0	0	0	0	1	0	0	1	36	2	m	asian	no	no	family member	No
11	10	1	1	1	0	1	1	0	1	1	1	22	8	m	south asia	no	no	Health Care Professional	Yes
12	11	1	0	0	1	0	1	1	0	1	1	36	6	m	Hispanic	yes	yes	family member	Yes
13	12	1	1	1	1	0	1	1	1	0	1	17	8	m	middle ea	yes	no	family member	Yes
14	13	0	0	0	0	0	0	0	0	0	0	25	0	f	middle ea	yes	no	family member	No
15	14	1	1	1	1	0	0	1	0	1	1	15	7	f	middle ea	yes	no	family member	Yes
16	15	0	0	0	0	0	0	0	0	0	0	18	0	m	middle ea	no	no	family member	No
17	16	1	1	1	0	1	0	1	1	0	1	12	7	m	black	no	no	family member	Yes
18	17	0	0	0	0	0	0	0	0	0	0	36	0	m	middle ea	no	yes	family member	No
19	18	1	1	1	0	1	1	1	1	0	1	12	8	f	middle ea	yes	no	family member	Yes
20	19	1	0	0	0	1	0	0	0	0	1	29	3	f	middle ea	no	no	family member	No
21	20	1	1	1	0	1	0	1	1	0	1	12	7	f	black	no	no	family member	Yes
22	21	1	0	0	1	1	1	1	1	1	0	36	7	m	middle ea	no	no	family member	Yes
23	22	1	0	1	1	1	1	1	0	1	0	36	7	m	middle ea	no	no	family member	Yes

Here data is given for 1055 cases.

The Questions:



Q-CHAT-10 **Quantitative Checklist for Autism in Toddlers**

A quick referral guide for parents to complete about their toddler (18 – 24 months) with concerns about autism.

For each item, please circle the response which best applies to your child:

		A	B	C	D	E
1	Does your child look at you when you call his/her name?	Always	Usually	Sometimes	Rarely	Never
2	How easy is it for you to get eye contact with your child?	Very easy	Quite easy	Quite difficult	Very difficult	Impossible
3	Does your child point to indicate that s/he wants something? (e.g. a toy that is out of reach)	Many times a day	A few times a day	A few times a week	Less than once a week	Never
4	Does your child point to share interest with you? (e.g. pointing at an interesting sight)	Many times a day	A few times a day	A few times a week	Less than once a week	Never
5	Does your child pretend? (e.g. care for dolls, talk on a toy phone)	Many times a day	A few times a day	A few times a week	Less than once a week	Never
6	Does your child follow where you're looking?	Many times a day	A few times a day	A few times a week	Less than once a week	Never
7	If you or someone else in the family is visibly upset, does your child show signs of wanting to comfort them? (e.g. stroking hair, hugging them)	Always	Usually	Sometimes	Rarely	Never
8	Would you describe your child's first words as:	Very typical	Quite typical	Slightly unusual	Very unusual	My child doesn't speak
9	Does your child use simple gestures? (e.g. wave goodbye)	Many times a day	A few times a day	A few times a week	Less than once a week	Never
10	Does your child stare at nothing with no apparent purpose?	Many times a day	A few times a day	A few times a week	Less than once a week	Never

SCORING: For questions 1-9: if you circle an answer in columns C, D or E, score 1 point per question. For question 10: if you circle an answer in columns A, B or C, score 1 point. Add points together for all ten questions. If your child scores more than 3 out of 10, the health professional may consider referring your child for a multi-disciplinary assessment.

Key reference: Allison C, Auyeung B, and Baron-Cohen S, (2012) *Journal of the American Academy of Child and Adolescent Psychiatry* 51(2):202-12.

UNIVERSITY OF CAMBRIDGE

© SBC/CA/BA/ARC/Cambridge University 1/5/12

autism research centre arc

NHS National Institute for Health Research

These are the questions that are present in the dataset**.**

The Code:

```
#Importing Libraries

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt
import pickle
import numpy as np
# Importing the DATASET and dropping unneccesary columns and some plots
asd = pd.read_csv('Toddler_Autism.csv')
print(asd['Ethnicity'].value_counts())
sns.countplot(x='Class/ASD Traits ' , data=asd)
asd.drop(['Case_No','Qchat-10-Score'], axis = 1, inplace = True)
asd.columns

# Encoding categorical data
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
columns = ['Ethnicity', 'Who completed the test' , 'Family_mem_with_ASD', 'Class/ASD Traits ', 'Sex', 'Jaundice']
for col in columns:
    asd[col] = le.fit_transform(asd[col])
#print(asd.dtypes)

# X and Y matrix formation
X = asd.drop(['Class/ASD Traits '], axis = 1)
Y = asd['Class/ASD Traits ']

#for differentiate the ethnicities
#from sklearn.compose import ColumnTransformer
#from sklearn.preprocessing import OneHotEncoder
#ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [12])], remainder='passthrough')
#X = np.array(ct.fit_transform(X))
# Train test split
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size = 0.30, random_state = 40)

#for better accuracy
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
```

```
x_test = sc.transform(x_test)


#******************************LOGISTIC REGRESSION ***********************************#
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(solver='lbfgs', max_iter=1000)
#the default solver in LogisticRegression is 'lbfgs'
classifier.fit(x_train, y_train)
y_pred = classifier.predict(x_test)


# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print('for logistic regression= \n',cm)
from sklearn.metrics import accuracy_score
print('for Logistic Regression the accuracy is= ',accuracy_score(y_test, y_pred))


#*****************************DECISION TREE CLASSIFIER ***********************************#
from sklearn.tree import DecisionTreeClassifier
classifier1 = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier1.fit(x_train, y_train)
# Predicting the Test set results
y_pred1 = classifier1.predict(x_test)


# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred1)
print('for random tree classifier= \n',cm)
print('for Random Tree Classifier the accuracy is= ',accuracy_score(y_test, y_pred1))


#*************************************SVM ***********************************#
from sklearn.svm import SVC
classifier2 = SVC()
classifier2.fit(x_train, y_train)
y_pred2 = classifier2.predict(x_test)
```

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred2)
print('for SVM= \n' ,cm)
print('for SVM the accuracy is= ',accuracy_score(y_test, y_pred2))



#**************************RANDOM FOREST CLASSIFIER *************************************#
from sklearn.ensemble import RandomForestClassifier
classifier3 = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
classifier3.fit(x_train, y_train)

# Predicting the Test set results
y_pred3 = classifier3.predict(x_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred3)
print('for random forest classification= \n' ,cm)
print('for Random forest Classification the accuracy is= ',accuracy_score(y_test, y_pred3))

#*******************************NAIVE BAYES *************************************#
from sklearn.naive_bayes import GaussianNB
classifier4 = GaussianNB()
classifier4.fit(x_train, y_train)
y_pred4 = classifier4.predict(x_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred4)
print('for naive bayes= \n',cm)
print('for Naive bayes the accuracy is= ',accuracy_score(y_test, y_pred4))
```

```
#****************************** KNEIGHBOURS CLASSIFIER***********************************#
from sklearn.neighbors import KNeighborsClassifier
classifier5 = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier5.fit(x_train, y_train)

# Predicting the Test set results
y_pred5 = classifier5.predict(x_test)

# Making the Confusion Matrix
cm = confusion_matrix(y_test, y_pred5)
print('for KNeighbour Classifier= \n',cm)
print('for KNeighbour Classifier the accuracy is= ',accuracy_score(y_test, y_pred5))

# Saving model to disk
pickle_out=open("classifier.pkl","wb")
pickle.dump(classifier, pickle_out)
pickle_out.close()

pickle_out=open("classifier2.pkl","wb")
pickle.dump(classifier2, pickle_out)
pickle_out.close()

pickle_out=open("classifier3.pkl","wb")
pickle.dump(classifier3, pickle_out)
pickle_out.close()

pickle_out=open("classifier4.pkl","wb")
pickle.dump(classifier4, pickle_out)
pickle_out.close()

pickle_out=open("classifier5.pkl","wb")
pickle.dump(classifier5, pickle_out)
pickle_out.close()
```

```
#pickle.dump(classifier, open('model.pkl','wb'))


#for other purposes
# Loading model to compare the results
#model = pickle.load(open('model.pkl','rb'))
#print(model.predict([[2, 9, 6]]))
```

For this code the predicted results are given below:

```
for logistic regression=
 [[ 97   0]
 [  0 220]]
for Logistic Regression the accuracy is=  1.0
for random tree classifier=
 [[ 85  12]
 [ 14 206]]
for Random Tree Classifier the accuracy is=  0.917981072555205
for SVM=
 [[ 89   8]
 [  2 218]]
for SVM the accuracy is=  0.9684542586750788
for random forest classification=
 [[ 86  11]
 [ 12 208]]
for Random forest Classification the accuracy is=  0.9274447949526814
for naive bayes=
 [[ 87  10]
 [  9 211]]
for Naive bayes the accuracy is=  0.9400630914826499
for KNeighbour Classifier=
 [[ 89   8]
 [ 15 205]]

for KNeighbour Classifier the accuracy is=  0.9274447949526814
```

- Now if the standard scalar part is omitted, the result varies for some of the classifiers. Standard Scalar can be useful because many machine learning algorithms assume that the features of the input data are normally distributed, and standardizing the features can help the algorithm to converge faster and perform better.

Then the results are given below:

```
for logistic regression=

 [[ 95   2]

 [  2 218]]

for Logistic Regression the accuracy is=  0.9873817034700315

for random tree classifier=

 [[ 85  12]

 [ 14 206]]

for Random Tree Classifier the accuracy is=  0.917981072555205

for SVM=

 [[ 44  53]

 [  1 219]]

for SVM the accuracy is=  0.8296529968454258

for random forest classification=

 [[ 86  11]

 [ 11 209]]

for Random forest Classification the accuracy is=  0.9305993690851735

for naive bayes=

 [[ 87  10]

 [  9 211]]

for Naive bayes the accuracy is=  0.9400630914826499

for KNeighbour Classifier=

 [[ 85  12]
 [ 18 202]]

for KNeighbour Classifier the accuracy is=  0.9053627760252366
```

- One-hot encoding is a method used to represent categorical variables as numerical data. It involves converting each categorical value into a new column and assigning a 1 or 0 (True/False) value to the column. For example, if a categorical variable has three categories: "red", "green", and "blue", then one-hot encoding will create three new columns, one for each category. If the original categorical value is "green", then the "green" column will have a value of 1 and the other two columns will have a value of 0.

One-hot encoding is often used in machine learning models as a way to represent categorical variables in a format that can be used as input. If it is used in “ethnicity" column in particular to separate or convert each value into new column , the result also varies for classifiers.

The result for this case:

```
for logistic regression=

 [[ 96   1]

 [  1 219]]

for Logistic Regression the accuracy is=  0.9936908517350158

for random tree classifier=

 [[ 85  12]

 [ 17 203]]

for Random Tree Classifier the accuracy is=  0.9085173501577287

for SVM=

 [[ 37  60]

 [  0 220]]

for SVM the accuracy is=  0.8107255520504731

for random forest classification=

 [[ 90   7]

 [  8 212]]

for Random forest Classification the accuracy is=  0.9526813880126183

for naive bayes=

 [[ 92   5]
 [ 38 182]]
for Naive bayes the accuracy is=  0.8643533123028391
for KNeighbour Classifier=
 [[ 87  10]
 [ 18 202]]
for KNeighbour Classifier the accuracy is=  0.9116719242902208
```

Deployment Using FastAPI

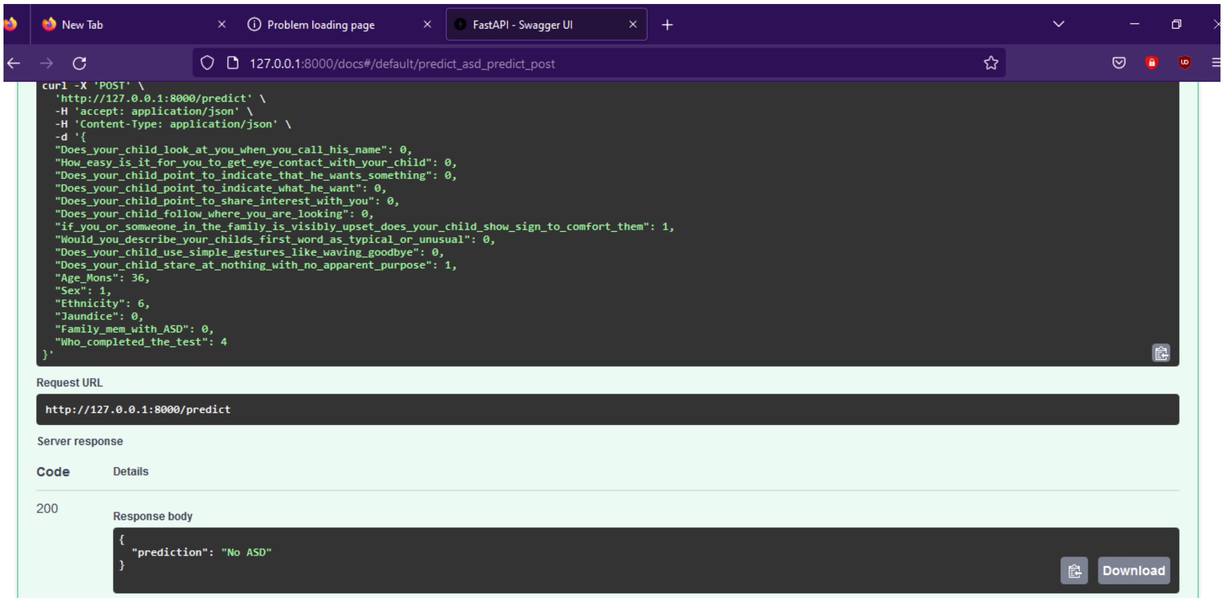
- Now for the deployment part Pickle file is created. It is used to store Python objects, such as lists, dictionaries, and class instances, in a binary format that can be saved to disk and later retrieved. Pickle files are often used to save data that needs to be persisted across sessions, such as trained machine learning models or large data sets that take a long time to load. It is deployed in FastAPI.

The code for this case is given below:

```
import uvicorn
from fastapi import FastAPI
from ASDs import ASD
import pickle
#create app object
app= FastAPI()
pickle_in=open("classifier.pkl","rb")
classifier=pickle.load(pickle_in)


#index routes, index page
@app.get('/')
def index():
    return{'message:' 'hi, my friend'}



#route with a parameter
@app.get('/{name}')
def get_name(name: str):
    return{'welcome here':f'{name}'}


#expose the prediction functionality; make a prediction from the json data and return the predicted asd
@app.post('/predict')
def predict_asd(data:ASD):
    data=data.dict()
    A1=data['Does_your_child_look_at_you_when_you_call_his_name']
    A2=data['How_easy_is_it_for_you_to_get_eye_contact_with_your_child']
    A3=data['Does_your_child_point_to_indicate_that_he_wants_something']
    A4=data['Does_your_child_point_to_indicate_what_he_want']
    A5=data['Does_your_child_point_to_share_interest_with_you']
    A6=data['Does_your_child_follow_where_you_are_looking']
    A7=data['if_you_or_somweone_in_the_family_is_visibly_upset_does_your_child_show_sign_to_comfort_them']
    A8=data['Would_you_describe_your_childs_first_word_as_typical_or_unusual']
    A9=data['Does_your_child_use_simple_gestures_like_waving_goodbye']
    A10=data['Does_your_child_stare_at_nothing_with_no_apparent_purpose']
    Age_Mons=data['Age_Mons']
    Sex=data['Sex']
    Ethnicity=data['Ethnicity']
    #Ethnicity- White European/asian/middle eastern/south asian/black/Hispanic/Others/Latino/mixed/Pacific/Native Indian
    Jaundice=data['Jaundice']
    Family_mem_with_ASD=data['Family_mem_with_ASD']
    Who_completed_the_test=data['Who_completed_the_test']

classifier.predict([[A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,Age_Mons,Sex,Ethnicity,Jaundice,Family_mem_with_ASD, Who_completed_the_test]])

prediction=classifier.predict([[A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,Age_Mons,Sex,Ethnicity,Jaundice,Family_mem_with_ASD, Who_completed_the_test]])
    if (prediction[0]>0.7):
        prediction="ASD"
    elif (prediction[0]<0.7 or prediction[0]>0.5):
        prediction="Should be under observation"
    else:
        prediction="No ASD"
    return{
        'prediction':prediction
        }

#run the api in unicorn
if __name__=='__main__':
    uvicorn.run(app,host='127.0.0.1',port=8000)
#python -m uvicorn app:app --reload
```

- Another file is also necessary to create this pickle file. Which is used to create the basemodel. The code for that is given below:

```
from pydantic import BaseModel
class ASD(BaseModel):
    Does_your_child_look_at_you_when_you_call_his_name: int
    How_easy_is_it_for_you_to_get_eye_contact_with_your_child: int
    Does_your_child_point_to_indicate_that_he_wants_something: int
    Does_your_child_point_to_indicate_what_he_want: int
    Does_your_child_point_to_share_interest_with_you: int
    Does_your_child_follow_where_you_are_looking: int
    if_you_or_somweone_in_the_family_is_visibly_upset_does_your_child_show_sign_to_comfort_them: int
    Would_you_describe_your_childs_first_word_as_typical_or_unusual: int
    Does_your_child_use_simple_gestures_like_waving_goodbye: int
    Does_your_child_stare_at_nothing_with_no_apparent_purpose: int
    Age_Mons: int
    Sex: int
    Ethnicity: int
    Jaundice: int
    Family_mem_with_ASD: int
    Who_completed_the_test: int
```



```
curl -X 'POST' \
  'http://127.0.0.1:8000/predict' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "Does_your_child_look_at_you_when_you_call_his_name": 0,
  "How_easy_is_it_for_you_to_get_eye_contact_with_your_child": 0,
  "Does_your_child_point_to_indicate_that_he_wants_something": 0,
  "Does_your_child_point_to_indicate_what_he_want": 0,
  "Does_your_child_point_to_share_interest_with_you": 0,
  "Does_your_child_follow_where_you_are_looking": 0,
  "if_you_or_somweone_in_the_family_is_visibly_upset_does_your_child_show_sign_to_comfort_them": 1,
  "Would_you_describe_your_childs_first_word_as_typical_or_unusual": 0,
  "Does_your_child_use_simple_gestures_like_waving_goodbye": 0,
  "Does_your_child_stare_at_nothing_with_no_apparent_purpose": 1,
  "Age_Mons": 36,
  "Sex": 1,
  "Ethnicity": 6,
  "Jaundice": 0,
  "Family_mem_with_ASD": 0,
  "Who_completed_the_test": 4
}'
```

Request URL

```
http://127.0.0.1:8000/predict
```

Server response

Code	Details
200	Response body { "prediction": "No ASD" }

```
{
  "prediction": "No ASD"
}
```

Fig: Deployment using FastAPI

Deployment using STREAMLIT WEBPAGE:

Streamlit Webpage has better interface than FastAPI. So it is better to deploy there the ML model.

Now to deploy it in streamlit webpage necessary codes are given below:

- **For Making a predictive Sytem**

input_data=(

0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 36, 1, 6,
0, 0, 4

)

#input data as numpy array

input_data_as_np_array=np.asarray(input_data)

#reshape the array for one instnce

input_data_reshaped=input_data_as_np_array.reshape(1,-1)

#scaling the data for one instance

std_data=sc.transform(input_data_reshaped)

prediction=classifier3.predict(std_data)

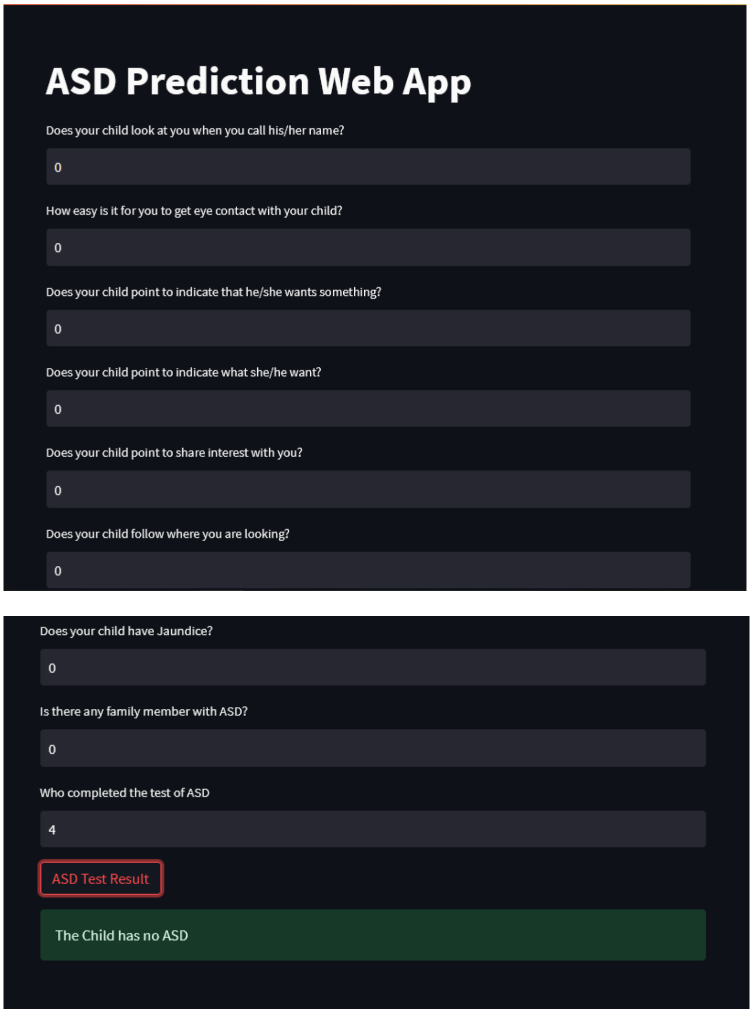
```
print('The prediction is: ',prediction)
#print('Here--->\n [1] Denotes ASD \n [0] Denotes No ASD')


if (prediction[0] == 1):
    print('The person has ASD')
else:
    print('The person has no ASD')
```

- **Now for the webpage part:**

```
import numpy as np
import pickle
import streamlit as st
#LOADING THE SAVED MODEL
loaded_model=pickle.load(open('J:/project 2.0/trained_model.sav','rb'))


# creating a function for Prediction
def ASD_prediction(input_data):
    #input data as numpy array
    input_data_as_np_array=np.asarray(input_data)
    #reshape the array for one instnce
    input_data_reshaped=input_data_as_np_array.reshape(1,-1)



    prediction=loaded_model.predict(input_data_reshaped)
    print('The prediction is: ',prediction)

    if (prediction[0] == 1):
        return 'The Child has ASD'
    else:
        return 'The Child has no ASD'

def main()
```

```
    # giving a title
    st.title('ASD Prediction Web App')


    # getting the input data from the user
    A1=st.text_input('Does your child look at you when you call his/her name?')
    A2=st.text_input(' How easy is it for you to get eye contact with your child?')
    A3=st.text_input('Does your child point to indicate that he/she wants something?')
    A4=st.text_input('Does your child point to indicate what she/he want?')
    A5=st.text_input('Does your child point to share interest with you?')
    A6=st.text_input('Does your child follow where you are looking?')
    A7=st.text_input('if you or someone in the family is visibly upset does your child show sign to comfort them?')
    A8=st.text_input('Would you describe your childs first word as typical or unusual?')
    A9=st.text_input('Does your child use simple gestures like waving goodbye?')
    A10=st.text_input('Does your child stare at nothing with no apparent purpose?')
    Age_Mons=st.text_input('Age Month of your child')
    Sex=st.text_input('Sex of the child')
    Ethnicity=st.text_input('Ethnicity of the child')
    Jaundice=st.text_input('Does your child have Jaundice?')
    Family_mem_with_ASD=st.text_input('Is there any family member with ASD?')
    Who_completed_the_test=st.text_input('Who completed the test of ASD')


    # code for Prediction
    diagnosis = ''


    # creating a button for Prediction
    if st.button('ASD Test Result'):
        diagnosis = ASD_prediction([A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,Age_Mons,Sex,Ethnicity,Jaundice,Family_mem_with_ASD, Who_completed_the_test])
    st.success(diagnosis)


if __name__ == '__main__':
```

```
main()

#streamlit run "J:\project 2.0\web_app.py"
```



ASD Prediction Web App

Does your child look at you when you call his/her name?

0

How easy is it for you to get eye contact with your child?

0

Does your child point to indicate that he/she wants something?

0

Does your child point to indicate what she/he want?

0

Does your child point to share interest with you?

0

Does your child follow where you are looking?

0

Does your child have Jaundice?

0

Is there any family member with ASD?

0

Who completed the test of ASD

4

The Child has no ASD

Fig: Deployment Using Streamlit Webpage