AI BASED DIABETES PREDICTION SYSTEM

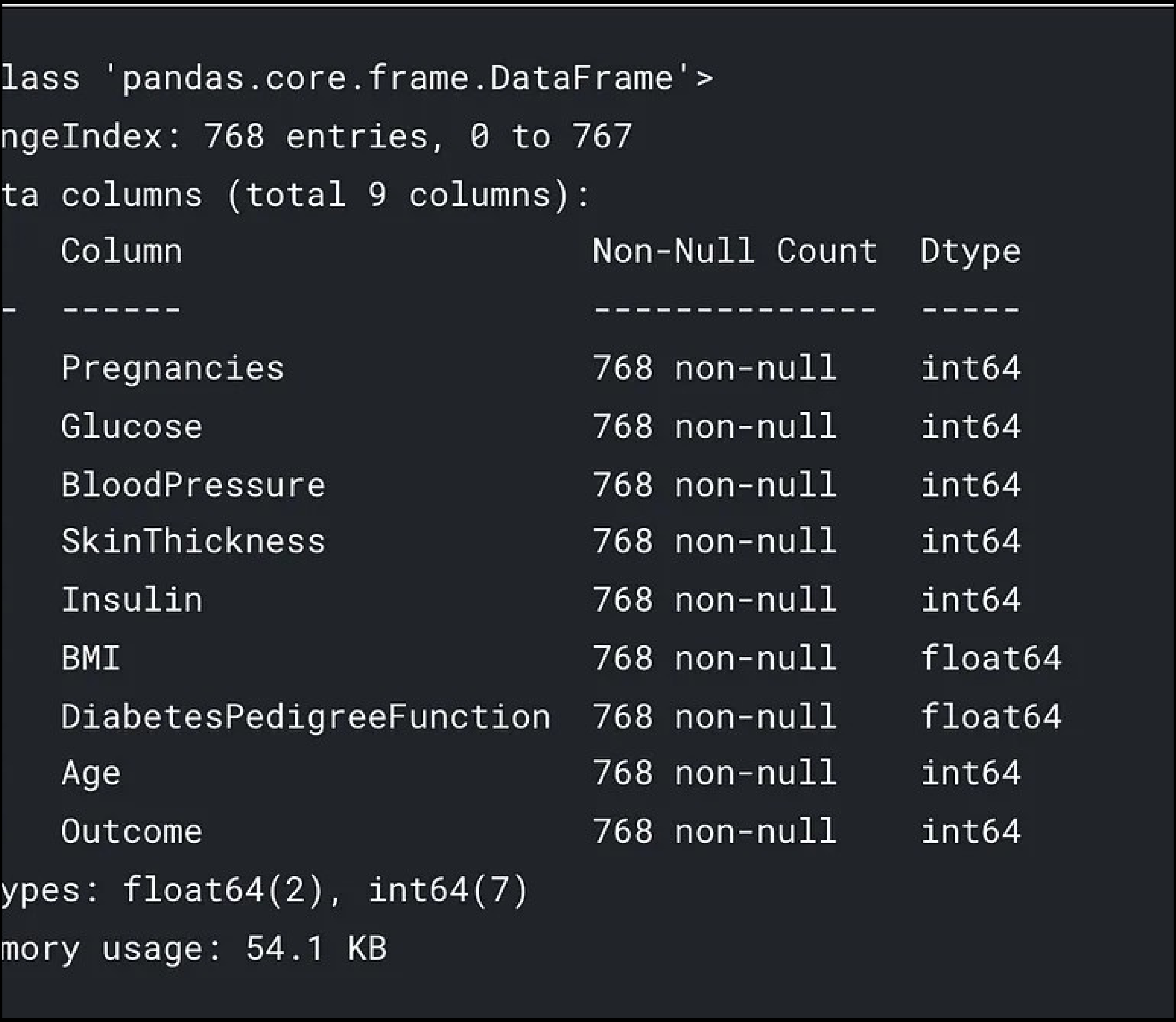
J.SEBASTIN SELVIN ERALD

III RD YEAR CSE

RVS COLLEGE OF ENGINEERING

TABLE OF CONTENTS

```
IMPORT DATASET
DF = PD.READ_CSV("../INPUT/PIMA-INDIANS-DIABETES-DATABASE/DIABETES.CSV")
# GET FAMILIER WITH DATASET STRUCTURE
DF.INFO()
```



```
lass 'pandas.core.frame.DataFrame'>
ngeIndex: 768 entries, 0 to 767
ta columns (total 9 columns):
   Column                    Non-Null Count  Dtype
-  ------                    --------------  -----
   Pregnancies               768 non-null    int64
   Glucose                   768 non-null    int64
   BloodPressure             768 non-null    int64
   SkinThickness             768 non-null    int64
   Insulin                   768 non-null    int64
   BMI                       768 non-null    float64
   DiabetesPedigreeFunction  768 non-null    float64
   Age                       768 non-null    int64
   Outcome                   768 non-null    int64
ypes: float64(2), int64(7)
mory usage: 54.1 KB
```

```
DF['GLUCOSE'] = DF['GLUCOSE'].REPLACE(0, DF['GLUCOSE'].MEAN())
# CORRECTING MISSING VALUES IN BLOOD PRESSURE
DF['BLOODPRESSURE'] = DF['BLOODPRESSURE'].REPLACE(0, DF['BLOODPRESSURE'].MEAN()) # THERE ARE 35 RECORDS WITH 0 BLOODPRESSURE IN DATASET
# CORRECTING MISSING VALUES IN BMI
DF['BMI'] = DF['BMI'].REPLACE(0, DF['BMI'].MEDIAN())
```

```
# Data Transformation
q  = QuantileTransformer()
X = q.fit_transform
transformedDF = q.transform(X)
transformedDF = pd.DataFrame(X)
transformedDF.columns =
['Pregnancies', 'Glucose',
'BloodPressure', 'SkinThickness',
'Insulin', 'BMI',
'DiabetesPedigreeFunction', 'Age',
'Outcome']
# Show top 5 rows
transformedDF.head
```

```
import pandas as pd
from sklearn.model_selection
import train_test_split
from sklearn.preprocessing
import StandardScaler,
LabelEncoder

# Load the dataset
data =
pd.read_csv('your_dataset.csv')

# Data Exploration (Optional)
# You can explore

Data Cleaning
# Handle missing values and remove
duplicates if necessary
```

```
# Data Preprocessing
# For example, encoding
categorical variables
label_encoder =
LabelEncoder()
data['categorical_column'] =
label_encoder

X = data.drop('target_column',
axis=1)  # Features
y = data['target_column']  #
Target variable
```