Spring 2023

Intro. to Machine Learning - CSC461
Dr. Abbas Rammal
Machine Learning Project

Cynthia Daou 202104972

Table of Contents

1. Introduction: 3
2. Describing the Dataset: 3
3. Analyzing and Visualizing the Dataset: 4
   I. Age_Mons 4
   II. Sex: 5
   III. Ethnicity: 5
   IV. Jaundice: 6
   V. A1-A10 Attributes: 7
4. Data Pre-processing: 8
   I. Dropping columns: 8
   II. Age Binning: 9
   III. Ethnicity binning: 9
   IV. Data Encoding: 9
   V. Final Dataset: 10
5. Splitting Data: 10
6. Training the Model: 11
   I. SVM: 11
   II. Logistic Regression: 12
   III. Decision Tree: 13
      • Decision tree with default parameters: 13
      • Decision tree with customized parameters: 13
   IV. Comparing models: 15
7. Parameter Tuning: 15
8. Making Predictions: 16
9. Final Decision Tree: 18
10. Conclusion: 18
11. References: 19

1. Introduction:

Diagnosing Autism Spectrum Disorder (ASD), a neurodevelopmental condition that results in significant healthcare costs, has been challenging. Early diagnosis of ASD can help reduce costs, but long waiting times and expensive procedures make it difficult to diagnose. There is a need for an easily accessible and effective screening method for ASD to inform health professionals and individuals about whether they should pursue formal clinical diagnosis. However, datasets related to ASD are rare, making it difficult to improve the efficiency and accuracy of the screening process. For this Machine Learning project, a dataset related to autism screening of toddlers was used which was proposed by Dr. Fayez Thabtah (Department of Digital Technology Manukau Institute of Technology, Auckland, New Zealand), which includes ten behavioral features (Q-Chat-10) and other individual characteristics to improve the classification of ASD cases. The purpose of the model is to accurately classify whether a child is prone to autism which suggests that a child should undergo a clinical diagnosis.

2. Describing the Dataset:

The dataset contains 19 attributes, categorical, continuous, and binary, and 1054 instances. Attributes A1, A2, A3, A4, A5, A6, A7, A8, A9, A10 are Yes/No behavioral questions. For these attributes if the response was Sometimes / Rarely / Never "1" is assigned to the question (A1-A9). However, for question 10 (A10), if the response was Always / Usually / Sometimes then "1" is assigned to that question. The Q-chat-10- score is the sum of scores for A1-A10 for each instance. Find below the description of each attribute.

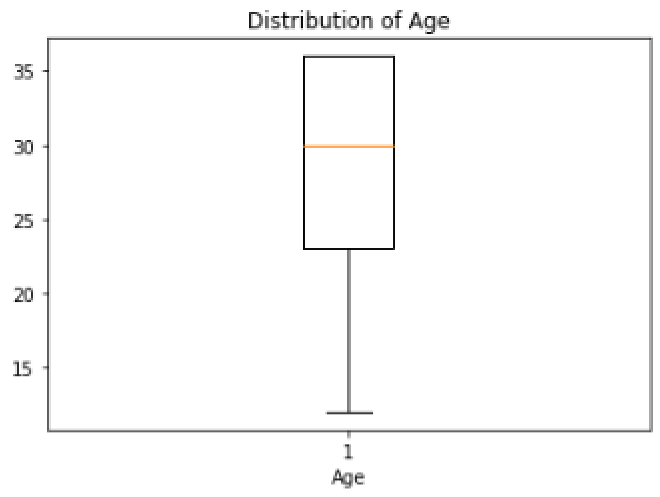
Variable in Dataset	**Corresponding Q-chat-10-Toddler Features**
A1	Does your child look at you when you call his/her name?
A2	How easy is it for you to get eye contact with your child?
A3	Does your child point to indicate that s/he wants something? (e.g. a toy that is out of reach)
A4	Does your child point to share interest with you? (e.g. poin9ng at an interes9ng sight)
A5	Does your child pretend? (e.g. care for dolls, talk on a toy phone)
A6	Does your child follow where you're looking?
A7	If you or someone else in the family is visibly upset, does your child show signs of wan9ng to comfort them? (e.g. stroking hair, hugging them)
A8	Would you describe your child's first words as:
A9	Does your child use simple gestures? (e.g. wave goodbye)
A10	Does your child stare at nothing with no apparent purpose?

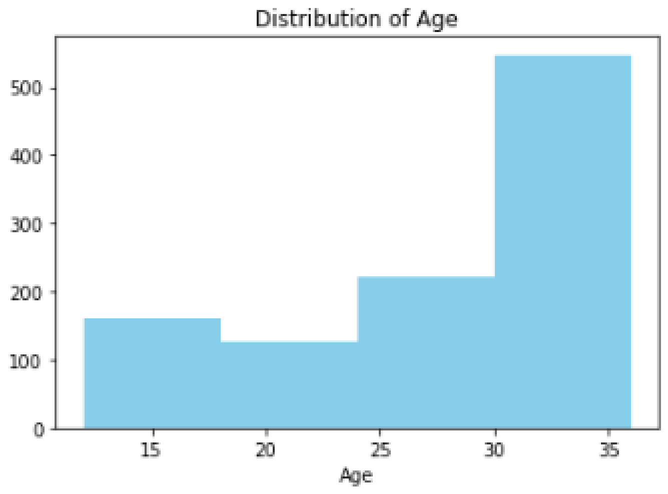
In addition to the behavioral questions, the remaining attributes are Age (continuous), Ethnicity (categorical), Sex (binary), Jaundice (binary), Family members with ASD (binary), who completed the test (categorical), and ASD_Traits (binary).

3. Analyzing and Visualizing the Dataset:

To visualize the dataset, many methods were used depending on the nature of the attribute.

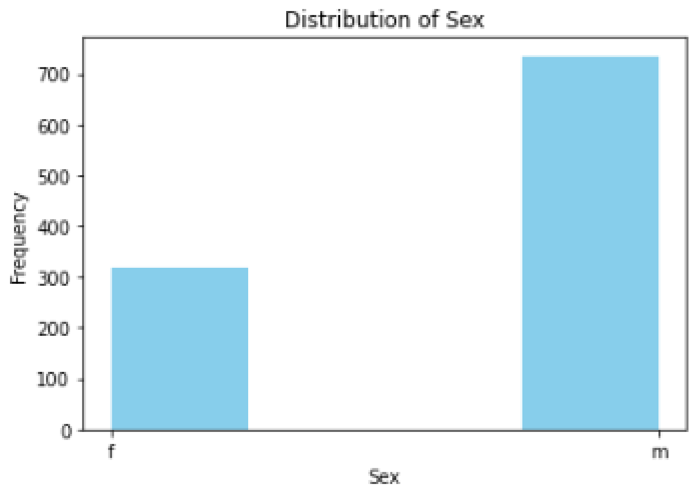
I. Age_Mons (Age in months):



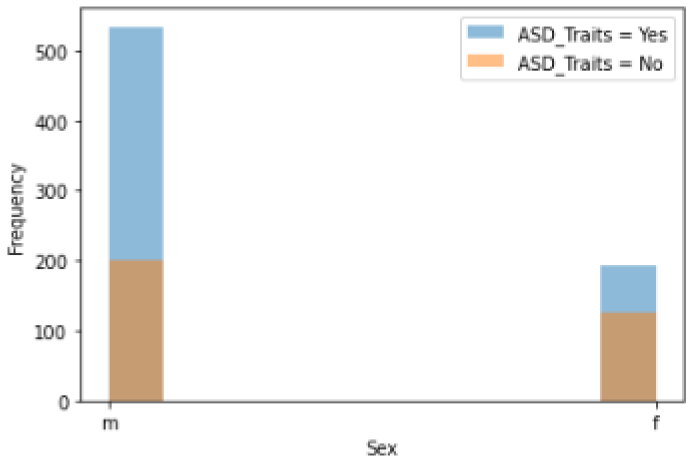


The above graphs show that the age ranges from 12 months until 36 months. The data is more centered above 30 months. The BoxPlot shows that 50% of data falls below 30 months and the remaining are between 30 and 36 months.

II. Sex:



The Histogram shows that around 70% of the data are males.

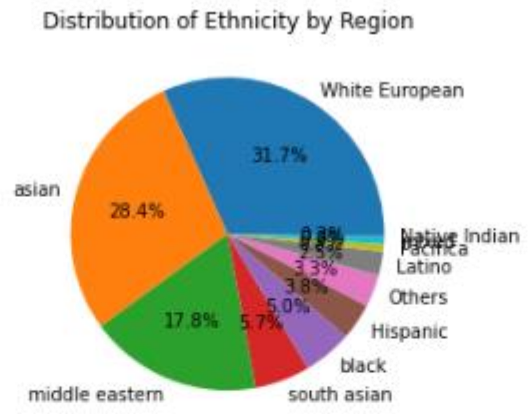


The above histogram shows the distribution of males and females with respect to the ASD_Traits attribute. Out of the males, around 28% showed no ASD traits whereas around 2/3rd of females in the data did not show ASD_Traits. According to these results, we cannot derive any conclusions as the portion of males is significantly greater than the portion of females.

III. Ethnicity:

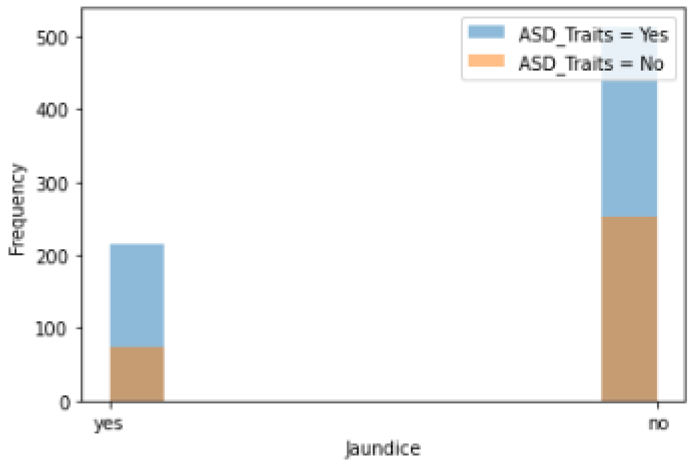
White European	334
asian	299
middle eastern	188
south asian	60
black	53
Hispanic	40
Others	35
Latino	26
Pacifica	8
mixed	8

```
Native Indian      3
Name: Ethnicity, dtype: int64
```



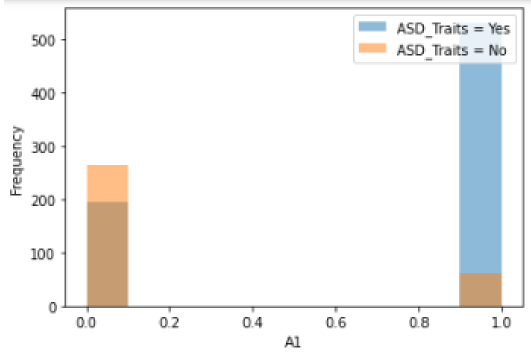
The dataset includes a variety of ethnicities, we can see that we have 3 major groups which are White European, Asian, and Middle eastern. This attribute will be later grouped into fewer bins to simplify analyzing the data.

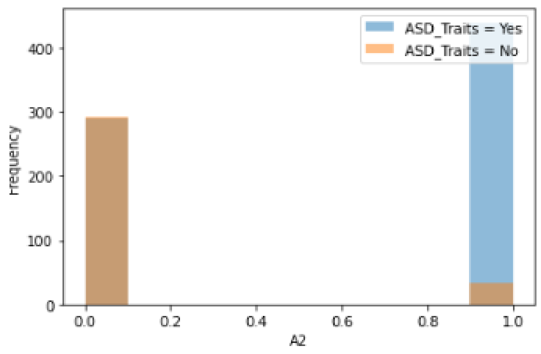
IV. Jaundice:

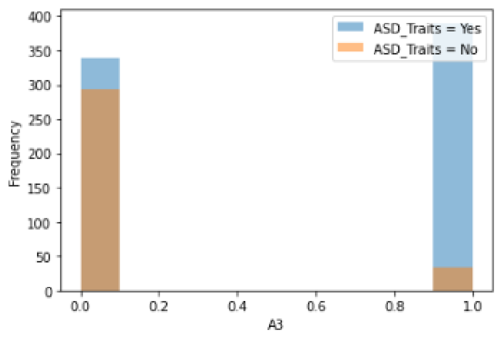


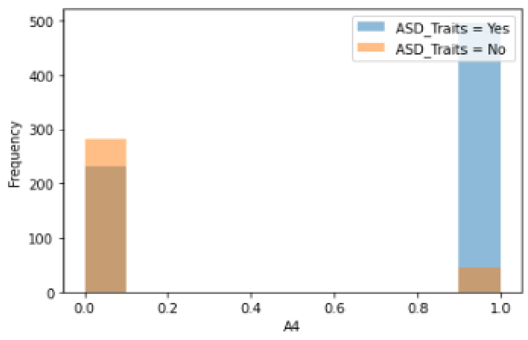
The correlation between ASD Traits and Jaundice is not clear form the graph as we can see that among the toddlers that do not have Jaundice, around half of them show ASD Traits while a bit more than half of toddlers that have Jaundice show ASD Traits.

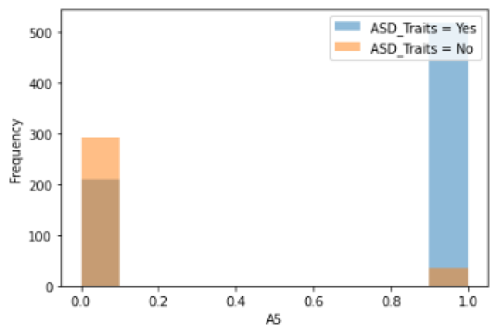
V. A1-A10 Attributes:

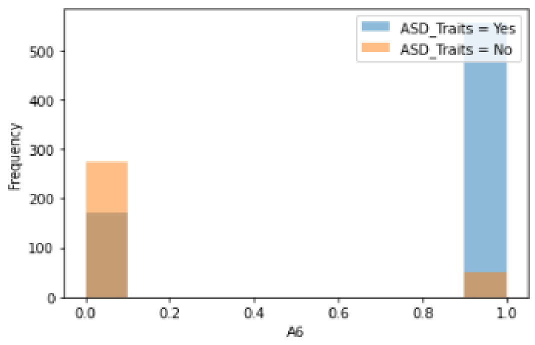


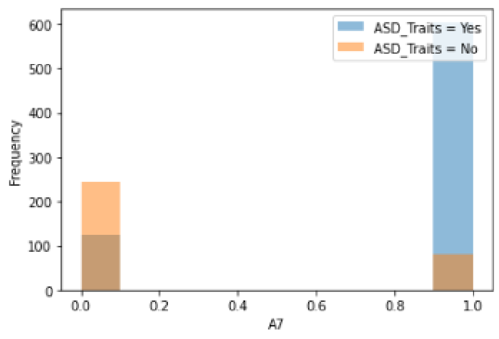


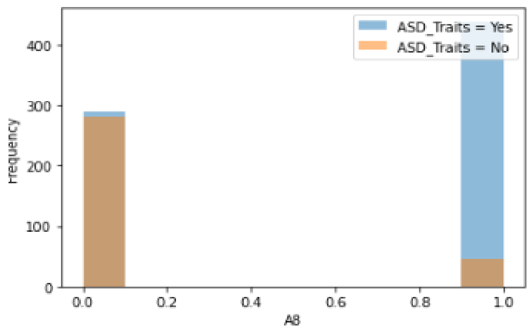


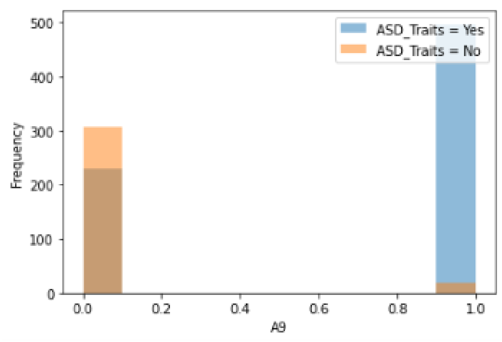


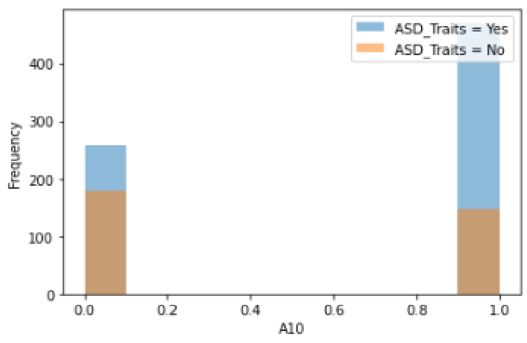












We can see from the above graph that most (if not all) of these attributes are a major determinant of whether the child has ASD_Traits. Mainly because if the answer is positive to any of these questions, it is usually that the child has a minor sign of ASD. Toddlers that score more than 3 on these questions are likely to have ASD_Traits according to the Qchat-10-score attribute.

4. Data Pre-processing:

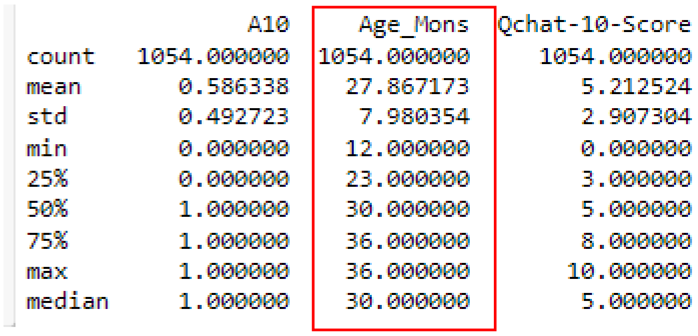
I. Dropping columns:

The dataset did not have any missing or noisy data which can be seen from the above graphs. The columns Case_No and Qchat-10-score were dropped so that they don't affect the prediction of the model. Then, the chi-squared test was applied to see whether there is a significant association between the behavioral features recorded in the dataset and the presence of Autism Spectrum Disorder (ASD). A contingency table was created for each feature and the target variable and the chi-squared values and p-values for each contingency table was calculated. The attribute "Family_mem_with_ASD" has the lowest chi-squared results with highest p-value (greater than 0.05) which shows that the results (ASD_Traits) has a very low correlation with this attribute so it was dropped from the data -frame. Below are the chi-squared test results.

```
                    Feature  Chi-Squared        p-value
18               ASD_Traits  1049.324266  3.413525e-230
12           Qchat-10-Score  1054.000000  4.336525e-220
9                        A9   348.821326   7.652682e-78
6                        A6   339.263970   9.229228e-76
5                        A5   332.002152   3.521437e-74
7                        A7   331.745509   4.005126e-74
4                        A4   266.830911   5.566898e-60
1                        A1   265.337325   1.178016e-59
2                        A2   224.389727   9.975027e-51
8                        A8   190.464319   2.517566e-43
3                        A3   175.115060   5.650417e-40
10                      A10    33.300959   7.894371e-09
14                Ethnicity    43.571293   3.929551e-06
13                      Sex    14.043553   1.786252e-04
11                 Age_Mons    56.625261   1.882686e-04
15                 Jaundice     5.427049   1.982715e-02
17  Who completed the test     3.788061   4.354478e-01
0                   Case_No  1054.000000   4.855174e-01
16      Family_mem_with_ASD     0.120943   7.280135e-01
```

The Qchat-10-score attribute is 100% correlated with the ASD_Traits because it will Directly affect the results of prediction. Case_No is irrelevant to the analysis, so it was also dropped.

II. Age Binning:



```
                 A10     Age_Mons  Qchat-10-Score
count    1054.000000  1054.000000     1054.000000
mean        0.586338    27.867173        5.212524
std         0.492723     7.980354        2.907304
min         0.000000    12.000000        0.000000
25%         0.000000    23.000000        3.000000
50%         1.000000    30.000000        5.000000
75%         1.000000    36.000000        8.000000
max         1.000000    36.000000       10.000000
median      1.000000    30.000000        5.000000
```

The minimum value of Age is 12 months, and the maximum value is 36 months. Hence, a logical binning would be to divide age into 4 bins with each 6 months grouped together. This was the split used for the Age_Mons attribute '11-18', '18-24', '24-30', '30-36'. The first bin starts from 11 however, since the minimum value of age is 12 and because we need 12 to be included in the bin, the lower bound was set to 11.

III. Ethnicity binning:

The data is distributed among 11 ethnicities. However, some categories are a minority and have very few observations. So according to the data visualized for ethnicities, I chose to split "Ethnicity" into 6 bins to simplify model training and increase the observations for each category. The new categories for Ethnicity are "Asian", "Latino/Hispanic", "Middle Eastern", "Black", "Others", and "white European". "South Asian" was grouped with "Asian", "Latino" and "Hispanic" were grouped together under "Latino/Hispanic" due features and cultural similarities. "Pacifica", "mixed", "Native Indian" were all grouped with "others" due to the very few observations foe each category.

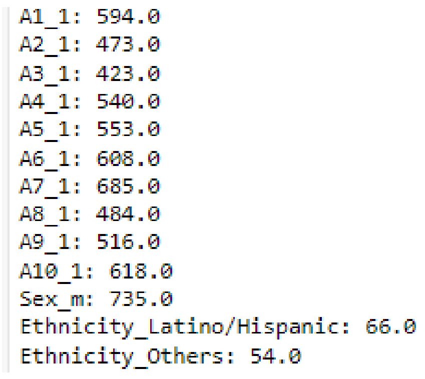
IV. Data Encoding:

To encode the data, all attributes were transformed to categorical attributes. Then encoded using OneHotEncoder from Sklearn to binary. The original columns were replaced with the new columns that include each attribute with its categories as columns. Many columns that were redundant were dropped after the encoding since for each attribute, two columns for "aatributе_0" and "attribute_1" were added. All columns with "Attribute-name_0" were dropped. In addition, "sex_female" column was also dropped since it can be deduced from the column "sex_male".

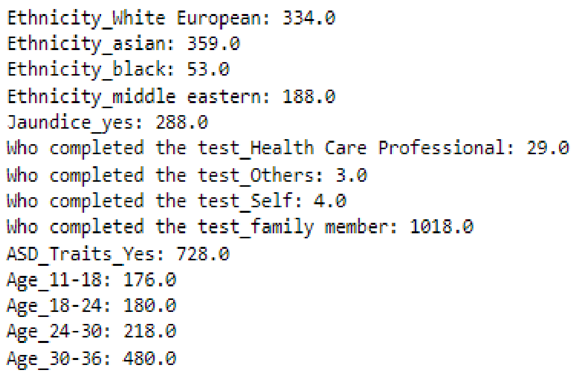
V. Final Dataset:

The final dataset has the following columns: ***'A1_1', 'A2_1', 'A3_1', 'A4_1', 'A5_1', 'A6_1', 'A7_1', 'A8_1', 'A9_1', 'A10_1', 'Sex_m', 'Ethnicity_Latino/Hispanic', 'Ethnicity_Other', 'Ethnicity_White European', 'Ethnicity_asian', 'Ethnicity_black', 'Ethnicity_middle eastern', 'Jaundice_yes', 'Who completed the test_Health Care Professional', 'Who completed the test_Others', 'Who completed the test_Self', 'Who completed the test_family member', 'ASD_Traits_Yes', 'Age_11-18', 'Age_18-24', 'Age_24-30', 'Age_30 -36'***

Below is the number of positive observations for each column:



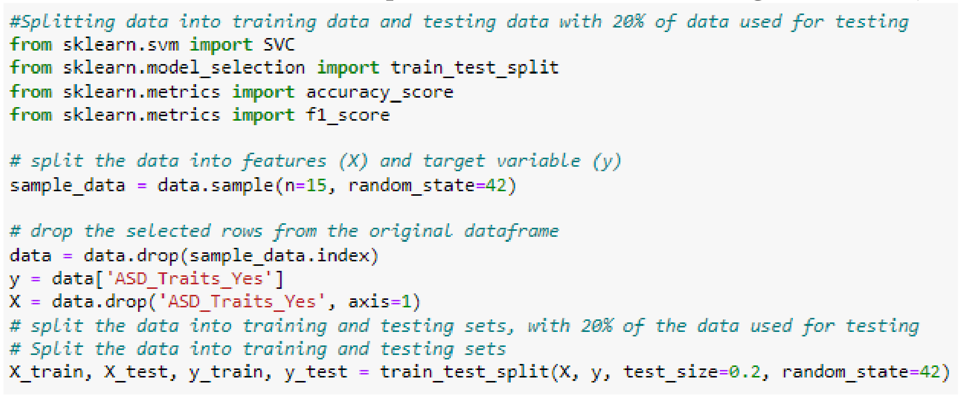
```
A1_1: 594.0
A2_1: 473.0
A3_1: 423.0
A4_1: 540.0
A5_1: 553.0
A6_1: 608.0
A7_1: 685.0
A8_1: 484.0
A9_1: 516.0
A10_1: 618.0
Sex_m: 735.0
Ethnicity_Latino/Hispanic: 66.0
Ethnicity_Others: 54.0
```



```
Ethnicity_White European: 334.0
Ethnicity_asian: 359.0
Ethnicity_black: 53.0
Ethnicity_middle eastern: 188.0
Jaundice_yes: 288.0
Who completed the test_Health Care Professional: 29.0
Who completed the test_Others: 3.0
Who completed the test_Self: 4.0
Who completed the test_family member: 1018.0
ASD_Traits_Yes: 728.0
Age_11-18: 176.0
Age_18-24: 180.0
Age_24-30: 218.0
Age_30-36: 480.0
```

5. Splitting Data:

The data was split into 20% testing data and 80% training data. Before splitting 15 random observations were removed from the dataset and moved to a new sample dataset to be used later for prediction. The remaining data was split between testing and training. The below code was used to split the data and label the target variable (ASD_Traits)

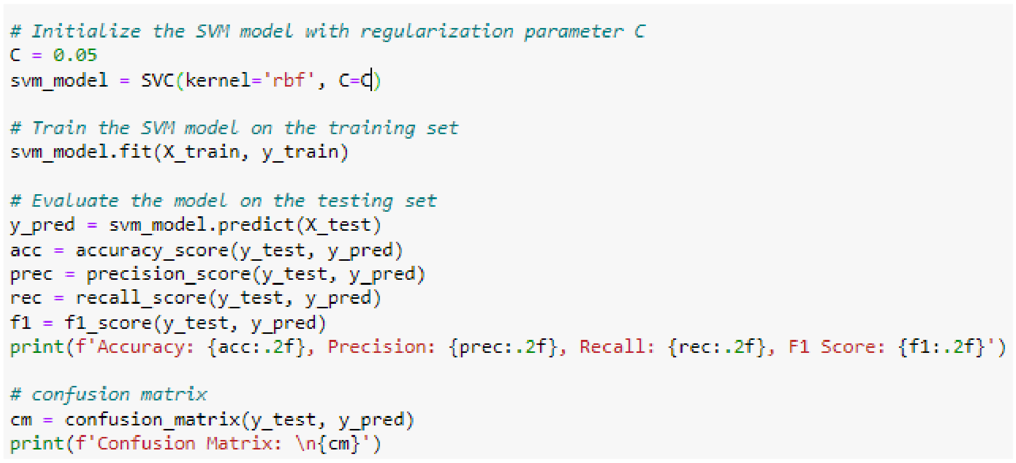


```
#Splitting data into training data and testing data with 20% of data used for testing
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score

# split the data into features (X) and target variable (y)
sample_data = data.sample(n=15, random_state=42)

# drop the selected rows from the original dataframe
data = data.drop(sample_data.index)
y = data['ASD_Traits_Yes']
X = data.drop('ASD_Traits_Yes', axis=1)
# split the data into training and testing sets, with 20% of the data used for testing
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

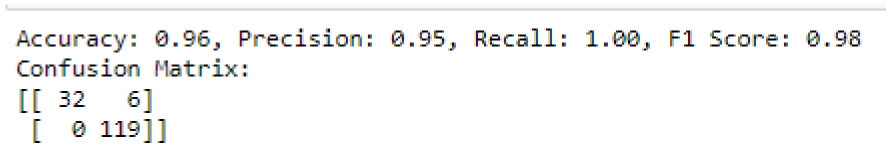
6. Training the Model:

To train the model to accurately predict the results according to the given attribute, I used three models: SVM, Logistic Regression, and Decision tree. The results of each model will be discussed and further analyzed to deduce the optimal model to be used.

I. SVM:



```python
# Initialize the SVM model with regularization parameter C
C = 0.05
svm_model = SVC(kernel='rbf', C=C)

# Train the SVM model on the training set
svm_model.fit(X_train, y_train)

# Evaluate the model on the testing set
y_pred = svm_model.predict(X_test)
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
print(f'Accuracy: {acc:.2f}, Precision: {prec:.2f}, Recall: {rec:.2f}, F1 Score: {f1:.2f}')

# confusion matrix
cm = confusion_matrix(y_test, y_pred)
print(f'Confusion Matrix: \n{cm}')
```



```
Accuracy: 0.96, Precision: 0.95, Recall: 1.00, F1 Score: 0.98
Confusion Matrix:
[[ 32   6]
 [  0 119]]
```

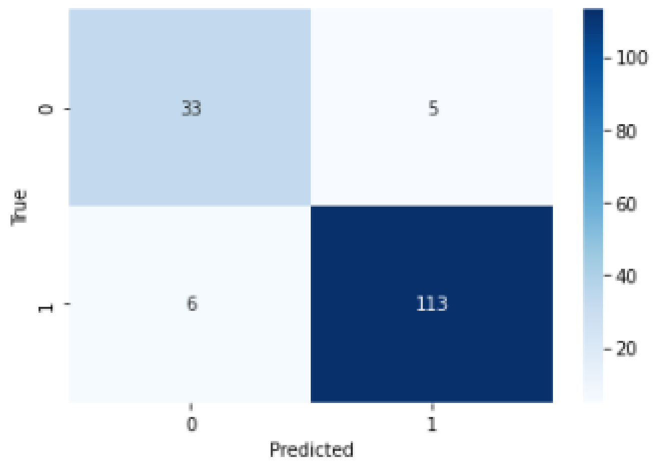
The above results of the SVM model suggest that it is performing well on new data. The accuracy of the SVM model is 96%, meaning that it correctly classified 96% of the test data instances. This indicates that the SVM model performs well in separating the two classes in the test data. The precision of the SVM model is 95%, meaning that when it predicts a positive instance, it is correct 95% of the time. This is a high precision score, indicating that the SVM model has a low rate of false positives. The recall of the SVM model is 100%, meaning that it correctly all the positive instances in the test data. This high recall score indicates that the SVM model is not predicting any false negatives. The F1 score of the SVM model is 98%, which is the harmonic mean of precision and recall. This score combines both precision and recall giving a single measure of the overall performance of the SVM model. A high F1 score indicates that the model has a good balance between precision and recall.

The confusion matrix reflects the results of the above discussed metrics. We can see that there are not any false negatives as the recall is 1. The SVM model correctly predicted 32 positive instances and 119 negative instances, while misclassifying 6 negative instances as positive. The high number of true positives and true negatives, and low number of false positives and false negatives in the confusion matrix indicates that the SVM model performs well in separating the two classes.

II. Logistic Regression:

For this model, the following hyperparameters were used to find accurate results: lr_model = LogisticRegression(penalty='l1', solver='saga', max_iter=150, C=0.05).
After trying with several parameters, the value of C was very critical for the model to avoid overfitting. Values over 0.2 for C were causing the model to overfit data with an accuracy score of 1. The value of penalty parameter was set to ‘l1’ in order to try to limit overfitting since it tends to make the mode sparser and less complex. These two parameters were the most important after experimenting with the different possibilities.
After using the above parameters, below are the results of the model.

```
Accuracy: 0.9299363057324841
Precision: 0.9576271186440678
recall: 0.9495798319327731
F1-Score: 0.9535864978902954
```

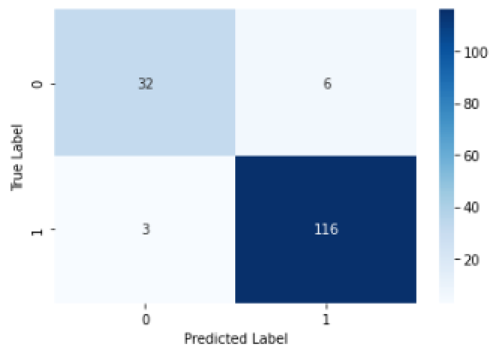


According to these results, the logistic regression model appears to perform well with an accuracy rate 93%, precision around 96%, recall 95% and f1-score 95%.
Even though the results of SVM are slightly higher, this might be a better performing model since the numbers of false predictions are more logically distributed. The SVM model’s high percentage of the above metrics’ scores may suggest that it is overfitting data especially with the recall score which shows that no False Negatives were predicted.
The confusion matrix shows results similar to the SVM model only with more false negatives (0 vs. 6) and fewer true negatives (119 vs. 113).
Overall, the logistic regression model seems to perform similar to the SVM model but with slightly different trade-offs in terms of false negative and true negativ

III. Decision Tree:

- Decision tree with default parameters:

```
Confusion Matrix:
[[ 32   6]
 [  3 116]]
```



```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.91      0.84      0.88        38
         1.0       0.95      0.97      0.96       119

    accuracy                           0.94       157
   macro avg       0.93      0.91      0.92       157
weighted avg       0.94      0.94      0.94       157

Training Accuracy: 1.0
Test Accuracy: 0.9426751592356688
```

First, the decision tree was trained without specifying any parameters that would optimize the performance of the model. The confusion matrix is similar to that of the logistic regression model. In the classification report, we can see that the recall score and f1-score are also very close to the scores of SVM and logistic regression models. Moreover, the training accuracy of the model is 1 which suggests that it is overfitting the data and will not perform well on new unseen data. We can see that the accuracy percentage of the model dropped to 94% on the testing dataset.

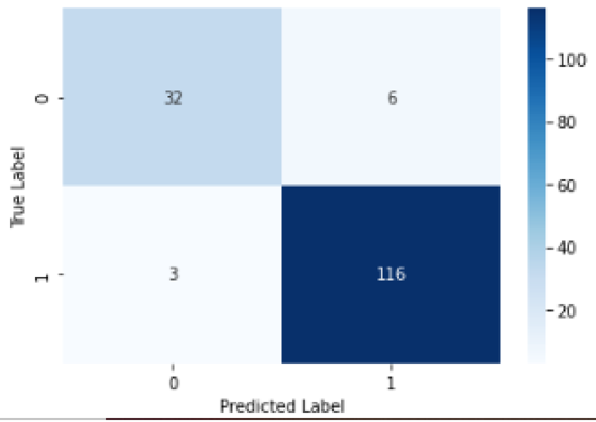
- Decision tree with customized parameters:

To make the decision tree model more accurate, we will find the maximum depth of the decision tree by using GridSearchCV to search over the parameter grid using 5-fold cross-validation. The given max_depth are [1, 3, 5, 7, 9]. The function will evaluate the performance of the model using each combination of hyperparameters (in this case depths of the tree) by performing 5-fold cross-validation. 5-fold cross-validation involves splitting the data into 5 equal parts or "folds". The model is trained on 4 of the folds and tested on the remaining fold. This process is repeated 5 times, with each fold used as the test set once. The average performance of the model across all 5 folds is then calculated. The combination of hyperparameters that result in the best performance, as measured by a scoring metric (e.g. accuracy, F1 score, etc.), is selected as the optimal combination. The results of the GridSearchCV function are:

Best parameters: {'max_depth': 7}
Best cross-validation score: 0.90

Hence, a new decision tree was created using the new max_depth of 7 and the results of confusion matrix and classification report are generated again.

```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.90      0.92      0.91        38
         1.0       0.97      0.97      0.97       119

    accuracy                           0.96       157
   macro avg       0.94      0.94      0.94       157
weighted avg       0.96      0.96      0.96       157

Confusion Matrix:
```



The values of classification report are slightly better than the previous decision tree. We can see that the precision, recall, and f1-score has slightly increased for both positive and negative predictions. Recall score was increased from 0.84 in the tree with default parameters to 0.92 in the new tree. Hence, f1-score also increased from 0.88 to 0.92. There was no change in the confusion matrix as all true positives, true negatives, false positives, and false negative remain the same number of predictions.
The training and testing accuracy shown below, along with the above metrics suggest that the model is well trained and is performing well on the testing data.

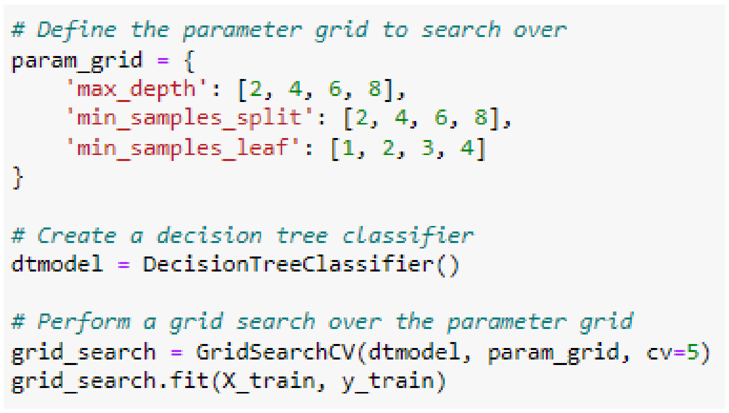
Training Accuracy: 0.9824561403508771
Test Accuracy: 0.9554140127388535

IV. Comparing models:

The performance of all three models was very similar. The dataset at hand is not very complicated and can be well classified and interpreted, hence, all models did a good job at predicting the ASD_Traits based on the provided attributes. Since one model should be chosen to rely on for predicting the ASD_traits in children, we will start by eliminating the SVM model. The very good performance of the SVM especially with a recall of 1 may not be realistic and may be a drawback for the model since it might be overfitting and memorizing the training data. Even if the data is simple, 100% accurate prediction of all positive instances from the testing dataset may be a bit misleading and questionable. The logistic regression model will not be considered as well due to similar overfitting issues. As discussed above, the value of the penalty was extremely critical for the model. A slight reduction in the strength of the penalty applied was causing the model to overfit the data which suggest that the model is vulnerable to overfitting and might not perform well on new data. Hence, the decision tree model will be considered since it as a simple method that can be well visualized and interpreted. The performance of the decision tree model was accurate with a good distribution of all the metrics scores discussed earlier. The decision tree structure will explicitly show how the input variables are combined to make predictions. Hence, further optimization for the decision tree model will be performed in the next steps.

7. Parameter Tuning:

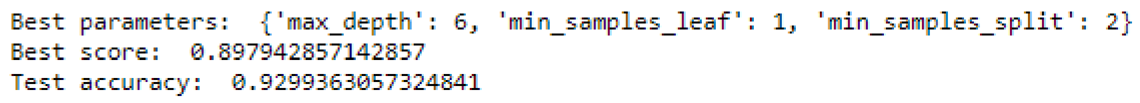
After choosing the decision tree mode, we will apply parameter tuning to try to find the parameters that will optimize the performance of the model. Parameter tuning is the process of adjusting the values of parameters in the machine learning model to optimize its performance on a given task. The goal is to find the optimal combination of parameter values that produces the best results. The process typically involves selecting a set of values for each parameter and then using a search algorithm to find the combination of values that maximizes the performance metric, such as accuracy or F1-score. Hence, GridSearchCV will be used in order to find the optimal maximum depth of the decision tree, minimum samples split, and minimum samples leaf. The below code was used to search among the given set of parameters for the ones with the best results.



```python
# Define the parameter grid to search over
param_grid = {
    'max_depth': [2, 4, 6, 8],
    'min_samples_split': [2, 4, 6, 8],
    'min_samples_leaf': [1, 2, 3, 4]
}

# Create a decision tree classifier
dtmodel = DecisionTreeClassifier()

# Perform a grid search over the parameter grid
grid_search = GridSearchCV(dtmodel, param_grid, cv=5)
grid_search.fit(X_train, y_train)
```

The results of the best parameters are:
{'max_depth': 6, 'min_samples_leaf': 1, 'min_samples_split': 2}
the maximum depth of the decision tree is 8 levels, which limits the complexity of the model and helps prevent overfitting. A min_samples_leaf of 1 means that each leaf node in the tree must have at least one sample, which may result in a more complex model, but can capture more fine-grained distinctions in the data. A min_samples_split of 2 means that a node can only be split if it has at least 2 samples, which also helps prevent overfitting by ensuring that each split is based on a sufficient number of samples.

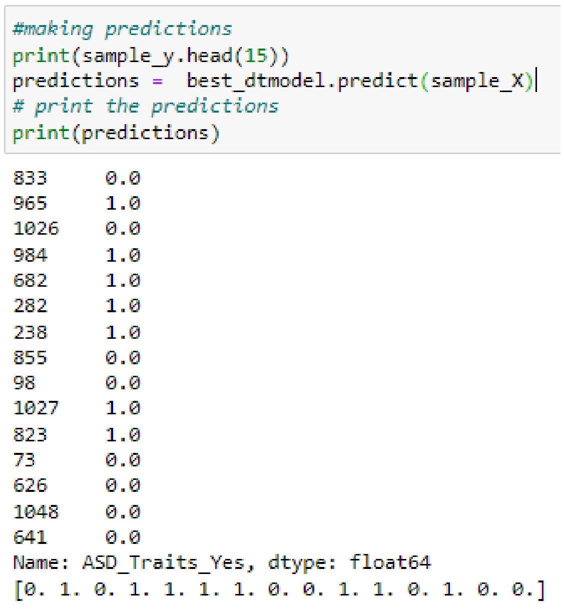
Hence, a new decision tree model was trained according to these new parameters. The best score is calculated by keeping track of the best scores while doing the GridSearch among all parameters provided.



```
Best parameters:  {'max_depth': 6, 'min_samples_leaf': 1, 'min_samples_split': 2}
Best score:  0.897942857142857
Test accuracy:  0.9299363057324841
```

8. Making Predictions:

After choosing the final model and trying to optimize the model with the best parameters. We can now use the model for making prediction on unseen data. We will use the sample data that was taken from the original data frame in the preprocessing step before training the models. The sample data has 15 observations, it was split into sample_X and sample_y. ‘sample_X’ contains all independent features that the model should consider when predicting the “ASD_Traits” and the ‘sample_y’ set contains the dependent variable that will be predicted. Predict() method will be used on the latest tuned decision tree model and the results will be compared with the ‘sample_y’ set.

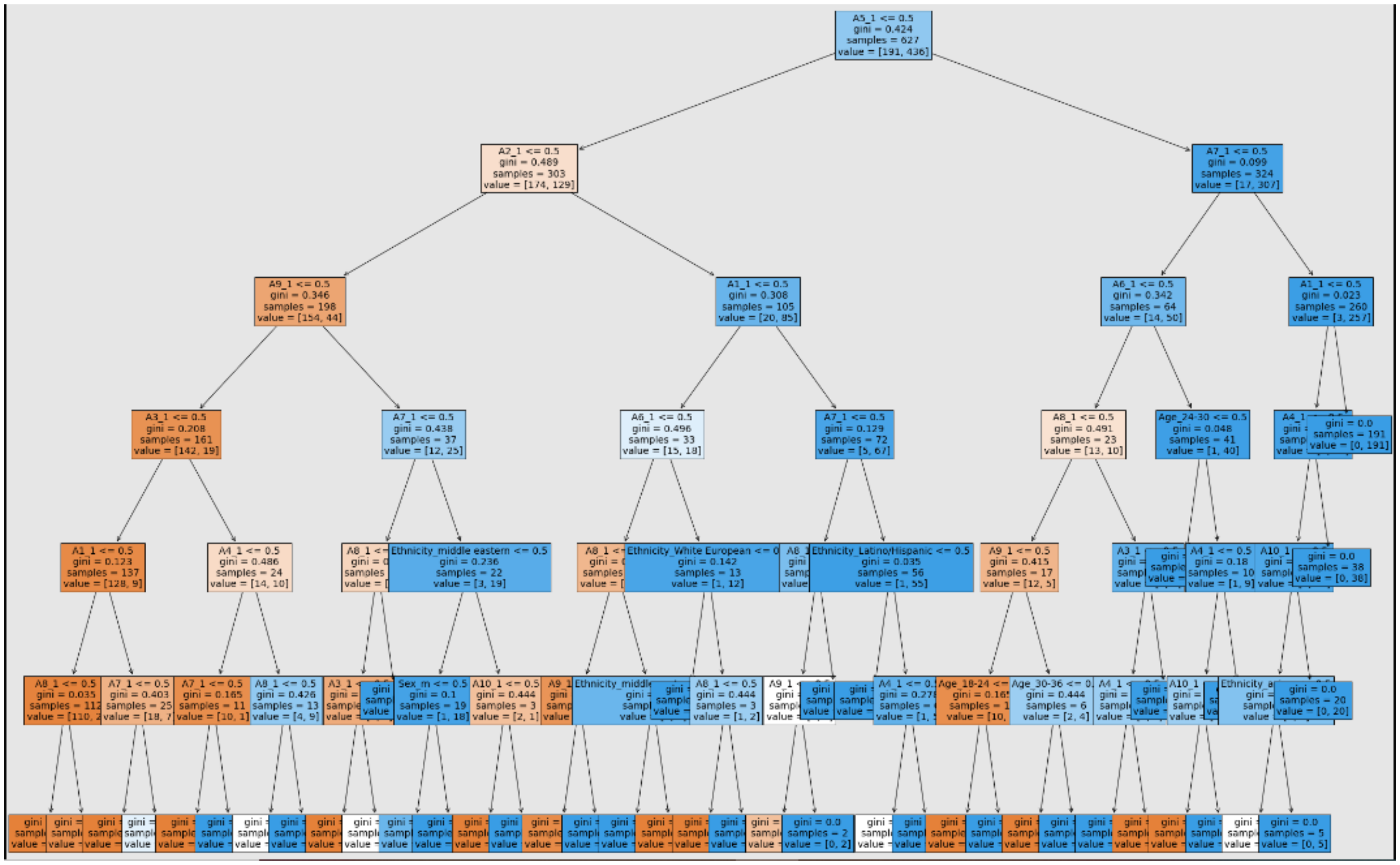


```
#making predictions
print(sample_y.head(15))
predictions =  best_dtmodel.predict(sample_X)
# print the predictions
print(predictions)
```

```
833     0.0
965     1.0
1026    0.0
984     1.0
682     1.0
282     1.0
238     1.0
855     0.0
98      0.0
1027    1.0
823     1.0
73      0.0
626     0.0
1048    0.0
641     0.0
Name: ASD_Traits_Yes, dtype: float64
[0. 1. 0. 1. 1. 1. 1. 0. 0. 1. 1. 0. 1. 0. 0.]
```

The results for the sample set were first printed then the model predicted the target variable for the provided sample set. Comparing the actual observation to the model's prediction, the model accurately predicted 14 out of 15 test cases and only failed to predict 1 test case, which is around 6.67% error.

9. Final Decision Tree:



10. Conclusion:

In conclusion, to develop a well-trained model that can accurately predict any data, many steps were followed. After finding a suitable data set to be used, data was visualized to get a better understanding of how attributes are distributed and their relationship with the target variable. Then, preprocessing the data was a major step to make sure the dataset is ready to be used for training the models properly without making it too complex or too simple for the model. After preprocessing, the dataset can be used to train different models. The models that were tested are SVM, Logistic regression, and decision tree. Although they all performed similar due to the simplicity of the dataset, there were brief variations and drawbacks for each model. Finally, Decision tree model was chosen as it serves the purpose of the model and handles the characteristics of the data well. So we tried to further optimize the decision tree model by choosing the best parameters and tried to use the model to make predictions on unseen data.

11. References:

- Fadi. “Autism Screening Data for Toddlers.” *Kaggle*, 23 July 2018, https://www.kaggle.com/datasets/fabdelja/autism-screening-for-toddlers.

- Nitheeshbopparaju. “Autism.” *Kaggle*, Oct. 2022, www.kaggle.com/code/nitheeshbopparaju/autism.