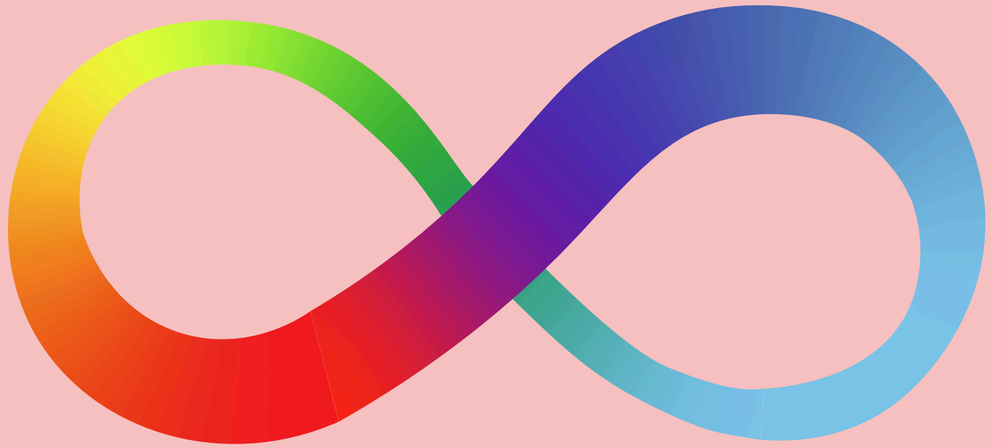
Presentation by: Group 7

Unraveling ASD Susceptibility Markers for Predictive Diagnosis



BIO221 | Winter Semester | 2024

ASD

Autism spectrum disorder (ASD) is a neurological and developmental disability caused by differences in the brain. People with ASD often have problems with social communication and interaction and restricted or repetitive behavior. Their symptoms can also affect their ability to function in school, work or other areas of life.

Problem Statement

Our objective was to contribute to a deeper comprehension of the genetic architecture of ASD. We aimed to identify genes whose mutations cause ASD, look for these mutations, and understand their effect.

Syndromic Genes

with gene score = 3

Ref: Sfari Gene

RIMS2	regulating synaptic membrane exocytosis 2
SRSF1	serine and arginine rich splicing factor 1
TCEAL1	transcription elongation factor A like 1
TRIM8	tripartite motif containing 8
YWHAG	tyrosine 3-monooxygenase/tryptophan 5-monooxyg activation protein gamma
ZFHX3	zinc finger homeobox 3
ZFX	zinc finger protein X-linked

DHX9	DExH-box helicase 9
FGF13	fibroblast growth factor 13
FRMD5	FERM domain containing 5
FRYL	FRY like transcription coactivator
MACF1	microtubule actin crosslinking factor 1
MSX2	msh homeobox 2
NAA10	N-alpha-acetyltransferase 10, NatA catalytic subunit
PABPC1	poly(A) binding protein cytoplasmic 1
PJA1	praja ring finger ubiquitin ligase 1

ADGRL1	adhesion G protein-coupled receptor L1
ATP2B1	ATPase plasma membrane Ca2+ transporting 1
CAMK2D	calcium/calmodulin dependent protein kinase II delta
CBX1	chromobox 1
CDH2	cadherin 2
CDK19	cyclin dependent kinase 19
CERT1	ceramide transporter 1
CSNK1G1	casein kinase 1 gamma 1
CTR9	CTR9homolog, Paf1/RNA polymerase II complex component

DAVID

QUERY PASSED

The genes identified as syndromic and of type 3(sfari gene scoring) are passed as the Query Gene List which is further analysed.

TOOL USED

Functional Annotation:

- Functional Annotation Clustering
- Functional Annotation Chart
- Functional Annotation Table

FURTHER ANALYSIS

The results from DAVID are analyzed to find an association between ASD and related neurodivergence.

32 record(s)

ATP2B1	ATPase plasma membrane Ca2+ transporting 1(ATP2B1)
GOTERM_BP_DIRECT	negative regulation of cytokine production, regulation of vascular smooth regulation of bone mineralization, ion transmembrane transport, regulatio cytosolic calcium ion concentration, positive regulation of calcium ion trans insulin stimulus, regulation of cardiac conduction, calcium ion export from
GOTERM_CC_DIRECT	immunological synapse, nucleoplasm, plasma membrane, membrane, bas presynaptic membrane, cell projection, intracellular membrane-bounded o glutamatergic synapse,
GOTERM_MF_DIRECT	calcium-transporting ATPase activity, protein binding, calmodulin binding, ATPase activity, PDZ domain binding, metal ion binding,
INTERPRO	P_typ_ATPase, ATPase_P-typ_cation-transptr_N, ATPase_P-typ_cation-tran ATPase_P-typ_P_site, ATP_Ca_trans_C, HAD_sf, ATPase_P-typ_TM_dom_s P_typ_ATPase_HD_dom,
KEGG_PATHWAY	Calcium signaling pathway, cGMP-PKG signaling pathway, cAMP signaling p synthesis and secretion, Endocrine and other factor-regulated calcium rea absorption,
OMIM_DISEASE	Intellectual developmental disorder, autosomal dominant 66,
SMART	Cation_ATPase_N,
UP_KW_BIOLOGICAL_PROCESS	Calcium transport, Ion transport, Transport

UP_SEQ_FEATURE	DOMAIN:RNA polymerase N-terminal, DOMAIN:RNA polymerase Rpb1, REGION:Bridging
ADGRL1	adhesion G protein-coupled receptor L1(ADGRL1)
GOTERM_BP_DIRECT	heterophilic cell-cell adhesion via plasma membrane cell adhesion molecules, cell surface receptor signaling pathway, adenylate cyclase-activating G-protein coupled receptor signa intracellular calcium source, positive regulation of synapse maturation,
GOTERM_CC_DIRECT	plasma membrane, membrane, axon, growth cone, presynaptic membrane, neuron proje
GOTERM_MF_DIRECT	G-protein coupled receptor activity, protein binding, latrotoxin receptor activity, carbohydr
INTERPRO	GPS, GPCR_2_secretin-like, Lectin_gal-bd_dom, GPCR_2_extracellular_dom, Olfac-like_d GPCR_2_latrophilin, GPCR_2-like_7TM, GPCR_2_secretin-like_CS, Latrophilin-1_TM, GAIN bd_sf, GAIN_dom_sf,
OMIM_DISEASE	Developmental delay, behavioral abnormalities, and neuropsychiatric disorders,
SMART	HormR, OLF, GPS,
UP_KW_CELLULAR_COMPONENT	Membrane, Synapse, Synaptosome, Cell projection, Cell membrane,
UP_KW_DISEASE	Disease variant, Intellectual disability, Autism spectrum disorder,
UP_KW_DOMAIN	Signal, Transmembrane, Transmembrane helix,
UP_KW_LIGAND	Lectin,
UP_KW_MOLECULAR_FUNCTION	G-protein coupled receptor, Receptor, Transducer,
UP_KW_PTM	Autocatalytic cleavage, Glycoprotein, Methylation, Phosphoprotein, Disulfide bond

OMIM DISEASE

It is a database cataloging all known diseases with a genetic component, and it provides detailed information about the genetic basis, clinical features, and inheritance patterns of these diseases.

- Out of 32 results obtained from DAVID, 12 had the SiPhy_29way_logOdds value greater than 5, indicating stronger evolutionary conservation.
- These 12 identified genes are also linked to neurological disorders and neurodivergence closely associated with autism spectrum disorder (ASD).

Gene Plots

Polyphen2_HVAR_score

- It is used as a metric to predict the potential impact of a genetic variant on protein function.
- It expresses the likelihood of it being damaging or deleterious with scores ranging from 0 to 1 (0 being benign, 1 being damaging).

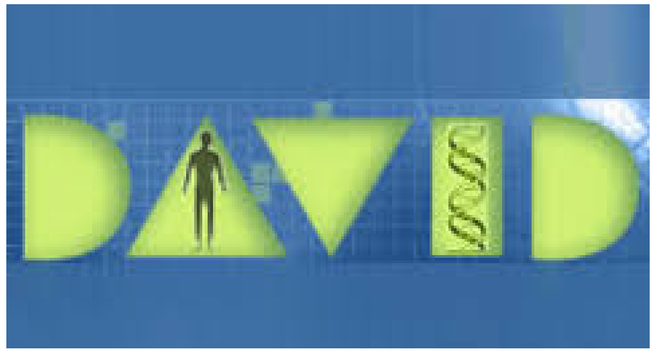
SIFT_score

- SIFT score predicts the impact of genetic variants on protein function, categorizing them as tolerated (higher score) or damaging (lower score).

Note: Both plots are scatter plots done against locus, categorized by chromosome.

Tools Used

biopython



Challenges || Further Expansions

01 There's a lack of comprehensive data on gene expressions, coupled with a significant amount of discrepancies.

02 Available tools have slow response times, which makes working with large amounts of data infeasible.

01 If gene expression data is accessible, conducting Differential Expression Analysis becomes feasible.

02 With consistent and comprehensive data, the integration of ML models can streamline the analysis of large datasets.

Code Snippets

Click here to ask Blackbox to help you code faster

```
import pandas as pd
```

Click here to ask Blackbox to help you code faster

```
data = pd.read_csv("final_final_data.tsv", sep = '\t')
```

Click here to ask Blackbox to help you code faster

```
data.info()
```

Click here to ask Blackbox to help you code faster

```
import statistics
import numpy as np
```

Click here to ask Blackbox to help you code faster

```
chromosomally_sorted_data = data.sort_values(by = 'chromosome')
chromosomally_sorted_data

dfs_by_number = {}

for number in chromosomally_sorted_data['chromosome'].unique():
    temp_df = chromosomally_sorted_data[chromosomally_sorted_data['chromosome'] == number]
    dfs_by_number[number] = temp_df

def median_with_nan(lst):
    valid_values = [x for x in lst if not np.isnan(x)]
```

```python
chromosomally_sorted_data = data.sort_values(by = 'chromosome')
chromosomally_sorted_data

dfs_by_number = {}

for number in chromosomally_sorted_data['chromosome'].unique():
    temp_df = chromosomally_sorted_data[chromosomally_sorted_data['chromosome'] == number]
    dfs_by_number[number] = temp_df

def median_with_nan(lst):
    valid_values = [x for x in lst if not np.isnan(x)]
    if valid_values:
        return statistics.median(valid_values)
    else:
        return float('nan')

dfs_by_num_poly = {}
dfs_by_num_positions = {}

for num in dfs_by_number:
    dfs_by_number[num].dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
    dff = dfs_by_number[num]['Polyphen2_HDIV_score'].tolist()
    positions_num = dfs_by_number[num]['start_hg19'].tolist()
    median_dff = median_with_nan(dff)
    for i in range(len(dff)):
        if np.isnan(dff[i]):
            dff[i] = median_dff
    dfs_by_num_poly[num] = dff
    dfs_by_num_positions[num] = positions_num


dfs_by_num_poly

```

Click here to ask Blackbox to help you code faster

```
all_genes = data['gene_symbol'].unique().tolist()
all_genes
```

Click here to ask Blackbox to help you code faster

```
len(data['gene_symbol'].unique())
```

Click here to ask Blackbox to help you code faster

```
dtype_col1 = data['gene_symbol'].apply(type)
is_float_in_gene_symbol = any(dtype == float for dtype in dtype_col1)
is_float_in_gene_symbol
```

Click here to ask Blackbox to help you code faster

```
df_no_float = data[~data['gene_symbol'].apply(lambda x: isinstance(x, float))]
```

Click here to ask Blackbox to help you code faster

```
dtype_col1 = df_no_float['gene_symbol'].apply(type)
is_float_in_gene_symbol = any(dtype == float for dtype in dtype_col1)
is_float_in_gene_symbol
```

Click here to ask Blackbox to help you code faster

```
gene_list = ["ADGRL1", "ATP2B1", "CAMK2D", "CBX1", "CDH2", "CDK19", "CERT1", "CSNK1G1",
```

Click here to ask Blackbox to help you code faster

```
gene_list = ["ADGRL1", "ATP2B1", "CAMK2D", "CBX1", "CDH2", "CDK19", "CERT1", "CSNK1G1",
    "CTR9", "DHX9", "FGF13", "FRMD5", "FRYL", "MACF1", "MSX2", "NAA10", "PABPC1",
    "PJA1", "POLR2A", "POLR3A", "PPFIA3", "PPP3CA", "PRPF8", "RFX4", "RFX7", "RIMS2",
    "SRSF1", "TCEAL1", "TRIM8", "YWHAG", "ZFHX3", "ZFX"]

filtered_data = df_no_float[df_no_float['gene_symbol'].isin(gene_list)]
filtered_data
```

Click here to ask Blackbox to help you code faster

```
usable_data = df_no_float.copy()
```

Click here to ask Blackbox to help you code faster

```
columns_to_be_removed = ['code_change', 'protein_change', 'gene_detail', 'cytoband', 'clinvar_20150629','LRT_score', 'LRT_pred', 'MutationTaster_score'
'MutationTaster_pred', 'MutationAssessor_score', 'MutationAssessor_pred', 'FATHMM_score', 'FATHMM_pred', 'RadialSVM_score', 'RadialSVM_pred',
'LR_score', 'LR_pred', 'VEST3_score', 'CADD_raw', 'CADD_phred', 'GERP_RS', 'phyloP46way_placental', 'phyloP100way_vertebrate']
data1 = usable_data.drop(columns = columns_to_be_removed)
data5 = data1.copy()
```

Click here to ask Blackbox to help you code faster

```
column_reduction_data = data1.copy()
```

Click here to ask Blackbox to help you code faster

```
column_reduction_data.dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
```

```
Click here to ask Blackbox to help you code faster
column_reduction_data = data1.copy()
```

```
Click here to ask Blackbox to help you code faster
column_reduction_data.dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
```

```
Click here to ask Blackbox to help you code faster
column_reduction_data.dropna(subset=['SIFT_score'], inplace=True)
```

```
Click here to ask Blackbox to help you code faster
column_reduction_data.info()
```

```
Click here to ask Blackbox to help you code faster
final_data = column_reduction_data.copy()
final_data

```

```python
sfari_genes = pd.read_csv('SFARI-Gene_genes_03-28-2024release_05-12-2024export.csv')
value_list = [3.0]
filtered_df = sfari_genes[sfari_genes['gene-score'].isin(value_list)]
filtered = filtered_df.copy()
filtered
```

✓ 0.1s

	status	gene-symbol	gene-name	ensembl-id	chromosome	genetic-category	gene-score	syndromic	eagle	number-o repor
5	9	ABL2	ABL proto-oncogene 2, non-receptor tyrosine ki...	ENSG00000143322	1	Rare Single Gene Mutation, Functional	3.0	0	NaN	
15	9	ADGRL1	adhesion G protein-coupled receptor L1	ENSG00000072071	19	Rare Single Gene Mutation, Syndromic	3.0	1	NaN	
24	9	AGAP5	ArfGAP with GTPase domain, ankyrin repeat and ...	ENSG00000172650	10	Rare Single Gene Mutation	3.0	0	NaN	
37	9	ALDH1L1	aldehyde dehydrogenase 1 family member L1	ENSG00000144908	3	Rare Single Gene Mutation	3.0	0	NaN	
60	9	ARHGEF2	Rho/Rac guanine nucleotide exchange factor 2	ENSG00000116584	1	Rare Single Gene Mutation, Syndromic	3.0	0	NaN	
...	...	...	...	...	...	...	...	...	...	
1142	9	YWHAG	tyrosine 3-monooxygenase/tryptophan 5-monooxyg...	ENSG00000170027	7	Rare Single Gene Mutation, Syndromic	3.0	1	NaN	
1143	9	YWHAZ	tyrosine 3-monooxygenase/tryptophan 5-monooxyg...	ENSG00000164924	8	Rare Single Gene Mutation, Syndromic, Genetic ...	3.0	0	NaN	
1149	9	ZBTB47	zinc finger and BTB domain containing 47	ENSG00000114853	3	Rare Single Gene Mutation	3.0	0	NaN	
1165	9	ZFHX3	zinc finger homeobox 3	ENSG00000140836	16	Rare Single Gene Mutation, Syndromic	3.0	1	NaN	
1166	9	ZFX	zinc finger protein X-linked	ENSG00000005889	X	Rare Single Gene Mutation, Syndromic	3.0	1	NaN	

Click here to ask Blackbox to help you code faster

```python
value_list = [1]
high_gene_score_syndromic = filtered[filtered['syndromic'].isin(value_list)]
high_gene_score_syndromic_list = high_gene_score_syndromic['gene-symbol'].tolist()
print(high_gene_score_syndromic_list)
```

✓ 0.0s Python

```
['ADGRL1', 'ATP2B1', 'CAMK2D', 'CBX1', 'CDH2', 'CDK19', 'CERT1', 'CSNK1G1', 'CTR9', 'DHX9', 'FGF13', 'FRMD5', 'FRYL', 'MACF1', 'MSX2', 'NAA10', 'PABPC1', 'PJA1',
```

Click here to ask Blackbox to help you code faster

```python
high_gene_score_syndromic_df = df_no_float[df_no_float['gene_symbol'].isin(high_gene_score_syndromic_list)]
high_gene_score_syndromic_df.dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
high_gene_score_syndromic_polyphen2 = high_gene_score_syndromic_df['Polyphen2_HDIV_score'].tolist()
```

Python

Click here to ask Blackbox to help you code faster

```python
value_list1 = [0]
high_gene_score_non_syndromic = filtered[filtered['syndromic'].isin(value_list1)]
high_gene_score_non_syndromic_list = high_gene_score_non_syndromic['gene-symbol'].tolist()
high_gene_score_non_syndromic_df = df_no_float[df_no_float['gene_symbol'].isin(high_gene_score_non_syndromic_list)]
high_gene_score_non_syndromic_df.dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
high_gene_score_non_syndromic_polyphen2 = high_gene_score_non_syndromic_df['Polyphen2_HDIV_score'].tolist()
```

Python

Click here to ask Blackbox to help you code faster

```python

value_list = [1.0]
filtered_df = sfari_genes[sfari_genes['gene-score'].isin(value_list)]
filtered_2 = filtered_df.copy()
filtered_2
```

Click here to ask Blackbox to help you code faster

```python
value_list1 = [0]
high_gene_score_non_syndromic = filtered[filtered['syndromic'].isin(value_list1)]
high_gene_score_non_syndromic_list = high_gene_score_non_syndromic['gene-symbol'].tolist()
high_gene_score_non_syndromic_df = df_no_float[df_no_float['gene_symbol'].isin(high_gene_score_non_syndromic_list)]
high_gene_score_non_syndromic_df.dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
high_gene_score_non_syndromic_polyphen2 = high_gene_score_non_syndromic_df['Polyphen2_HDIV_score'].tolist()
```

Pytho

Click here to ask Blackbox to help you code faster

```python

value_list = [1.0]
filtered_df = sfari_genes[sfari_genes['gene-score'].isin(value_list)]
filtered_2 = filtered_df.copy()
filtered_2
```

Pytho

Click here to ask Blackbox to help you code faster

```python
import numpy as np
```

Pytho

Click here to ask Blackbox to help you code faster

```python

value_list1 = [0]
low_gene_score_non_syndromic = filtered_2[filtered_2['syndromic'].isin(value_list1)]
low_gene_score_non_syndromic_list = low_gene_score_non_syndromic['gene-symbol'].tolist()
low_gene_score_non_syndromic_df = df_no_float[df_no_float['gene_symbol'].isin(low_gene_score_non_syndromic_list)]
low_gene_score_non_syndromic_df.dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
low_gene_score_non_syndromic_polyphen2 = low_gene_score_non_syndromic_df['Polyphen2_HDIV_score'].tolist()

```

```
value_list = [1.0]
filtered_df = sfari_genes[sfari_genes['gene-score'].isin(value_list)]
filtered_2 = filtered_df.copy()
filtered_2
```

[ ]

Click here to ask Blackbox to help you code faster

```
import numpy as np
```

[ ]

Click here to ask Blackbox to help you code faster

```
value_list1 = [0]
low_gene_score_non_syndromic = filtered_2[filtered_2['syndromic'].isin(value_list1)]
low_gene_score_non_syndromic_list = low_gene_score_non_syndromic['gene-symbol'].tolist()
low_gene_score_non_syndromic_df = df_no_float[df_no_float['gene_symbol'].isin(low_gene_score_non_syndromic_list)]
low_gene_score_non_syndromic_df.dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
low_gene_score_non_syndromic_polyphen2 = low_gene_score_non_syndromic_df['Polyphen2_HDIV_score'].tolist()

```

[ ]

Click here to ask Blackbox to help you code faster

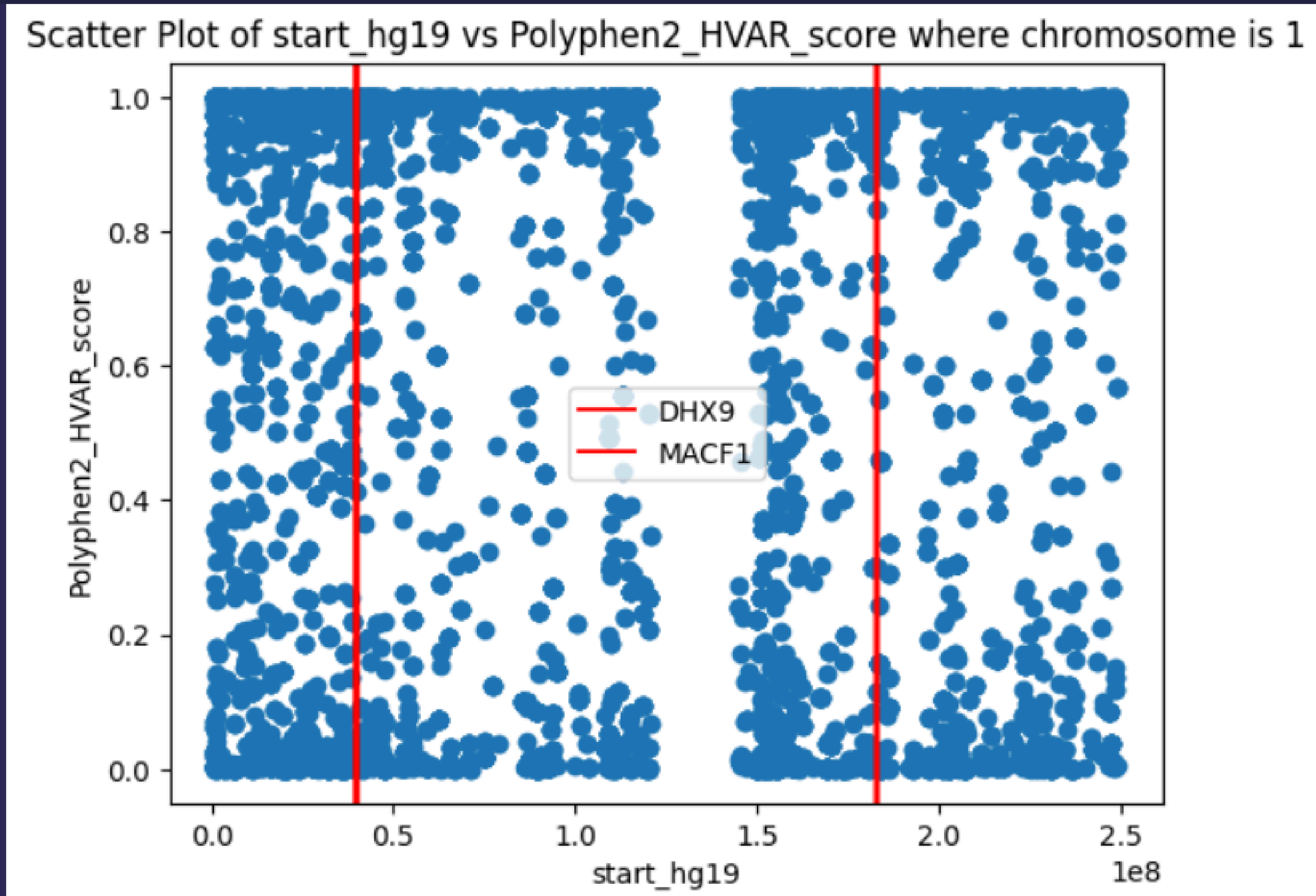
```
value_list2 = [1]
low_gene_score_syndromic = filtered_2[filtered_2['syndromic'].isin(value_list2)]
low_gene_score_syndromic_list = low_gene_score_syndromic['gene-symbol'].tolist()
low_gene_score_syndromic_df = df_no_float[df_no_float['gene_symbol'].isin(low_gene_score_syndromic_list)]
low_gene_score_syndromic_df.dropna(subset=['Polyphen2_HDIV_score'], inplace=True)
low_gene_score_syndromic_polyphen2 = low_gene_score_syndromic_df['Polyphen2_HDIV_score'].tolist()
```

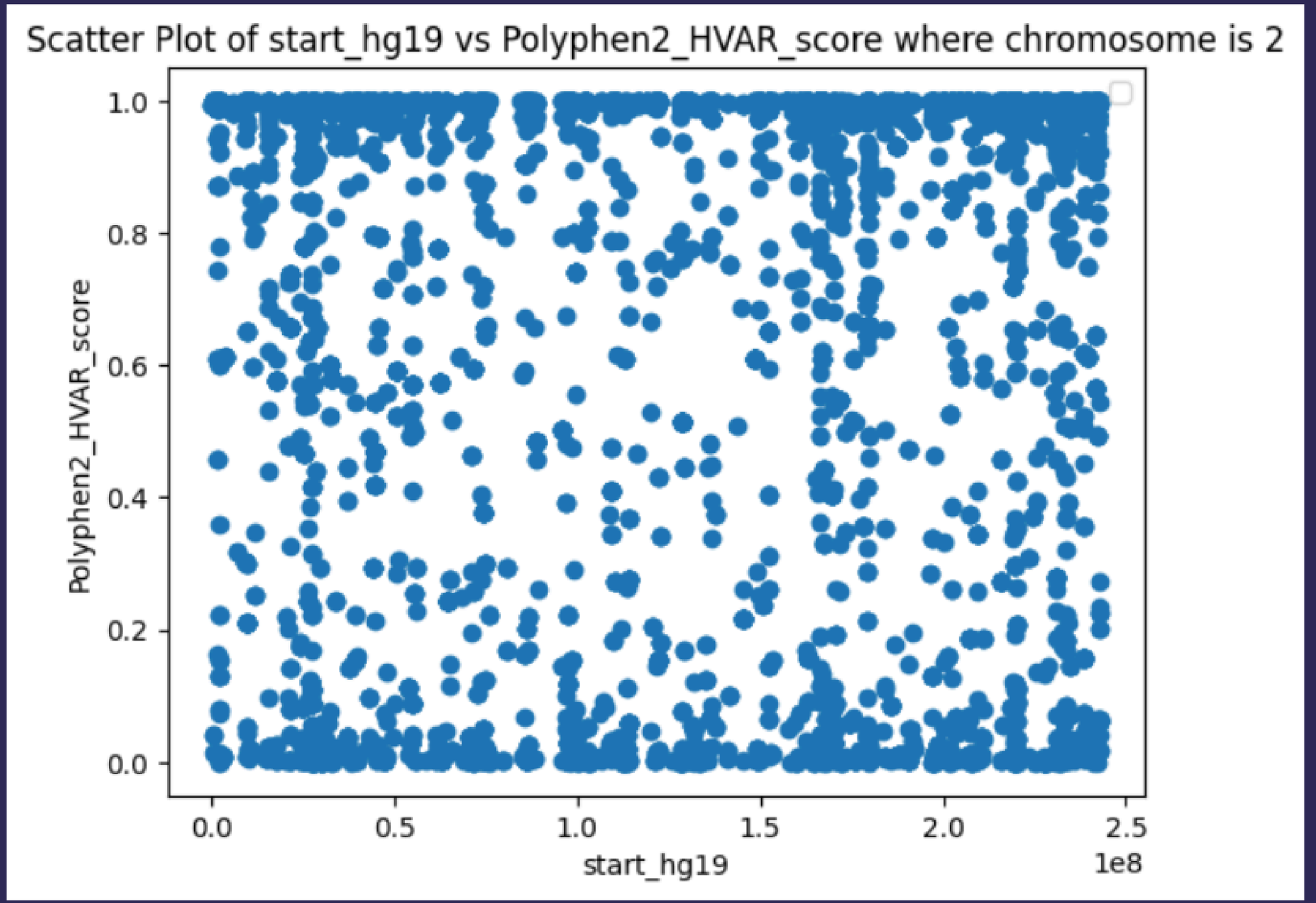
Click here to ask Blackbox to help you code faster

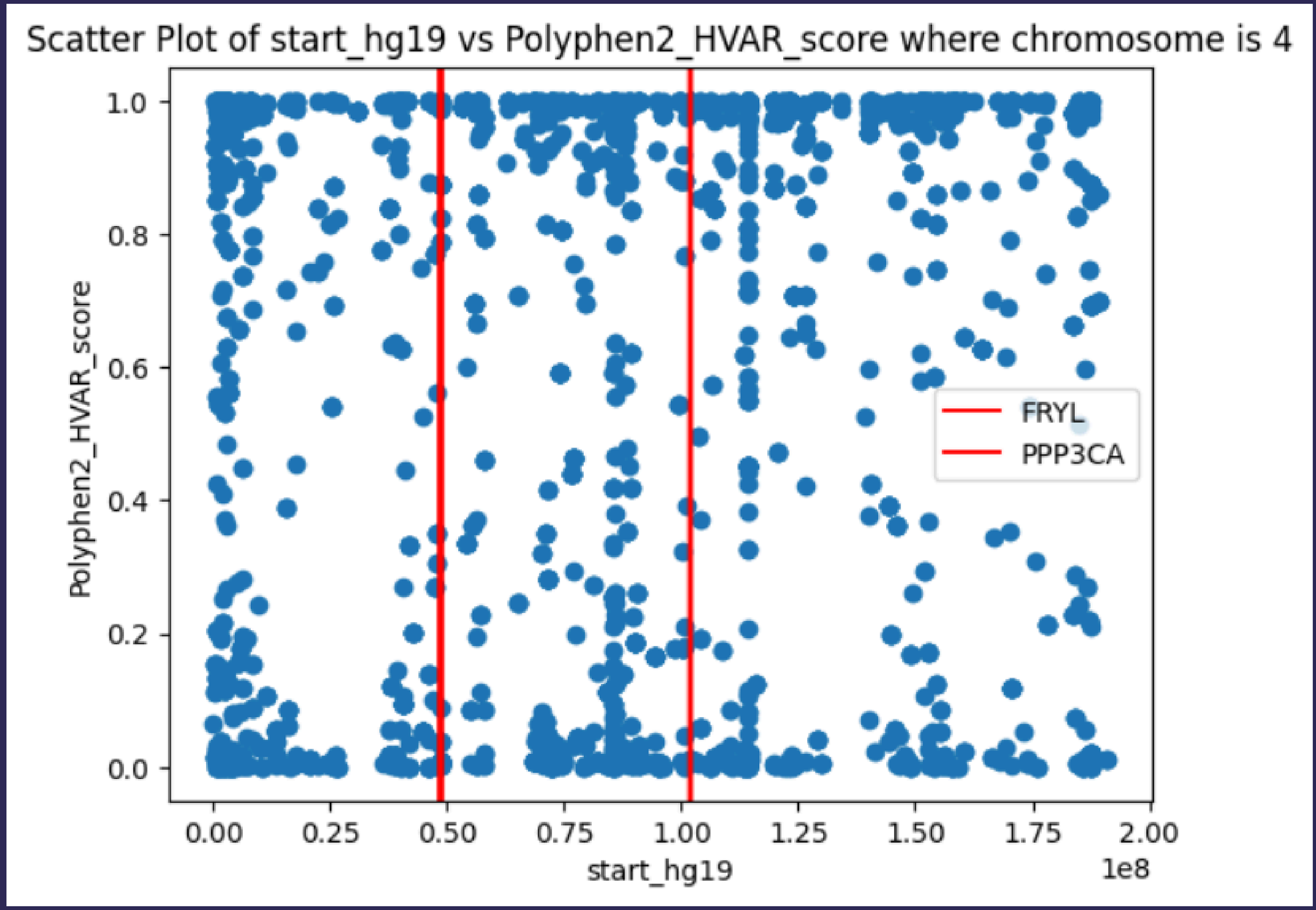
```python
import matplotlib.pyplot as plt
```

Click here to ask Blackbox to help you code faster

```python
for i in range(1,25):
    legend_handles = {}
    chr = i
    filtered_df = data[data['chromosome'] == chr]
    plt.scatter(filtered_df['start_hg19'], filtered_df['Polyphen2_HVAR_score'])
    plt.xlabel('start_hg19')
    plt.ylabel('Polyphen2_HVAR_score')
    if (i == 23):
        chr = "X"
    elif (i == 24):
        chr = "Y"
    filtered_syndromic_chromosome = high_gene_score_syndromic_df[high_gene_score_syndromic_df['chromosome'] == i]
    for i, row in filtered_syndromic_chromosome.iterrows():
        line = plt.axvline(row['start_hg19'], color = 'red', label = row['gene_symbol'])
        if row['gene_symbol'] not in legend_handles:
            legend_handles[row['gene_symbol']] = line
    plt.title(f'Scatter Plot of start_hg19 vs Polyphen2_HVAR_score where chromosome is {chr}')
    plt.legend(legend_handles.values(), legend_handles.keys())
    plt.show()
```

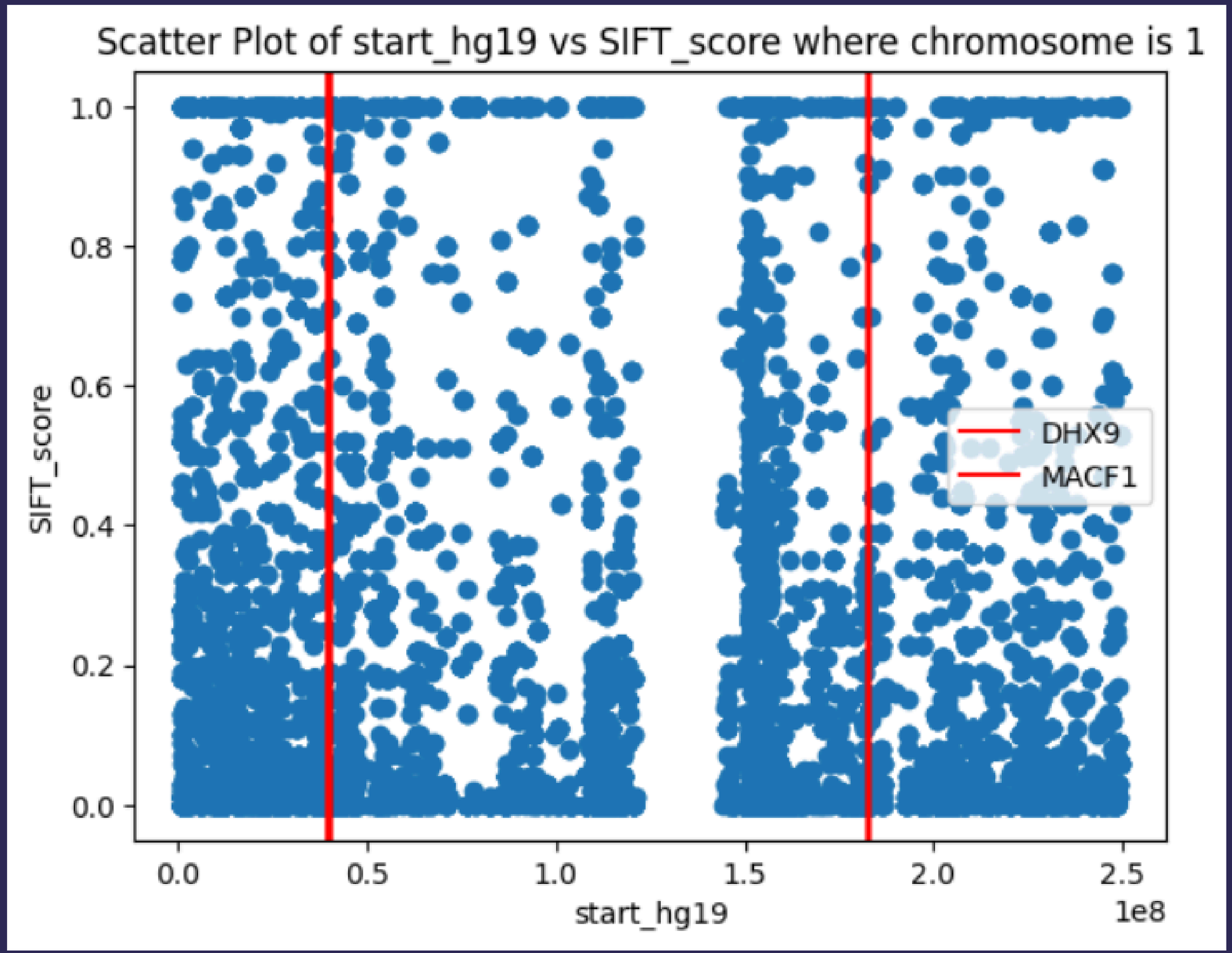

Scatter Plot of start_hg19 vs Polyphen2_HVAR_score where chromosome is 1
Polyphen2_HVAR_score
1.0
0.8
0.6
0.4
0.2
0.0
DHX9
MACF1
0.0
0.5
1.0
1.5
2.0
2.5
start_hg19
1e8


Scatter Plot of start_hg19 vs Polyphen2_HVAR_score where chromosome is 2
Polyphen2_HVAR_score
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.5
1.0
1.5
2.0
2.5
start_hg19
1e8


Scatter Plot of start_hg19 vs Polyphen2_HVAR_score where chromosome is 4
Polyphen2_HVAR_score
1.0
0.8
0.6
0.4
0.2
0.0
FRYL
PPP3CA
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
start_hg19
1e8

Click here to ask Blackbox to help you code faster

```python
for i in range(1,25):
    legend_handles = {}
    chr = i
    filtered_df = data[data['chromosome'] == chr]
    plt.scatter(filtered_df['start_hg19'], filtered_df['SIFT_score'])
    plt.xlabel('start_hg19')
    plt.ylabel('SIFT_score')
    if (i == 23):
        chr = "X"
    elif (i == 24):
        chr = "Y"
    filtered_syndromic_chromosome = high_gene_score_syndromic_df[high_gene_score_syndromic_df['chromosome'] == i]
    for i, row in filtered_syndromic_chromosome.iterrows():
        line = plt.axvline(row['start_hg19'], color = 'red', label = row['gene_symbol'])
        if row['gene_symbol'] not in legend_handles:  # Add only unique handles and labels
            legend_handles[row['gene_symbol']] = line
    plt.legend(legend_handles.values(), legend_handles.keys())
    plt.title(f'Scatter Plot of start_hg19 vs SIFT_score where chromosome is {chr}')
    plt.show()
```


Scatter Plot of start_hg19 vs SIFT_score where chromosome is 1
SIFT_score
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.5
1.0
1.5
2.0
2.5
start_hg19
1e8
DHX9
MACF1

References

- https://gene.sfari.org/database/human-gene/
- https://varicarta.msl.ubc.ca/downloads
- https://david.ncifcrf.gov/tools.jsp
- http://genetics.bwh.harvard.edu/pph2/
- Kong, Sek Won, Christin D. Collins, Yuko Shimizu-Motohashi, Ingrid A. Holm, Malcolm G. Campbell, In-Hee Lee, Stephanie J. Brewster et al. "Characteristics and predictive value of blood transcriptome signature in males with autism spectrum disorders." *PloS one* 7, no. 12 (2012): e49475.
- Rastegari, M., Salehi, N. & Zare-Mirakabad, F. Biomarker prediction in autism spectrum disorder using a network-based approach. BMC Med Genomics 16, 12 (2023). https://doi.org/10.1186/s12920-023-01439-5.

Member Contributions

Aditi Sharma (2022025): Data analyzing, pre-processing, research

Ananya Garg (2022068): DAVID , Polyphen2, Data processing

Gurupriya (2022191): OMIM data check, SIFT

Mann Nariya (2022278): SIFT plot, data mining

Medha Kashyap (2022292): SFARI and DAVID analysis, research

Nischaya Roy (2022333): Data Identification, polyphen2 and sift plot, Research

Thank You

Presentation by: Group 7

BIO221 | Winter Semester | 2024