Requirements (Environment)

* OS: Microsoft Windows
* Memory (RAM): 32GB maximum
* Disk Space: 2GB
* CPU/GPU: intel CORE i7/ intel iRIS xe or equivalent
* Software: Python, Jupyter Notebook

Packages Required

* Pandas
* Numpy
* Plotly.pyplot
* Seaborn
* Statistics
* os
* matplotlib.pyplot
* Sklearn.ensemble import RandomForestClassifier
* Sklearn.svm import SVC
* Sklearn.model\_selection import cross\_validate, train\_test\_split
* Sklearn.preprocessing import StandardScaler
* Sklearn.multioutput import MultiOutputClassifier
* Sklearn.model\_selection import train\_test\_split
* Sklearn.impute import SimpleImputer
* Sklearn.metrics import classification\_report, accuracy\_score
* Sklearn.metrics import classification\_report
* sklearn.feature\_selection import SelectKBest, f\_classif

Instructions

1. Data Preprocessing
   1. First, the data files must be merged to be used in the SVM models. To do this we need the [**TRAIN\_FUNCTIONAL\_CONNECTOME\_MATRICES\_new\_36P\_Pearson.csv**](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EVyEPPcE4o5OuWtSQ3PUnuwBtQGaK7TwQC416ENec75c8g?e=USA9fz)**,** [**TRAIN\_QUANTITATIVE\_METADATA\_new.xlsx**](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EaesyzY7v-BLmYM5ZXnEWwYBcY0jt5kuavkJMOsaJkcKHA?e=98gxFw)**,** [**TRAIN\_CATEGORICAL\_METADATA\_new.xlsx**](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EROljcTNwZ9JrS0kwJQu3e8BvssJJKGSV8EmPze7fKJJVA?e=UhKqSX), and [**TRAINING\_SOLUTIONS.xlsx**](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EdVvjUSpv6pJmsHG6KGcIusBWS0gSnj2CNHYAxcoC7qPvw?e=GesWw0)datasets.
   2. Preprocessing the data includes filling in missing data, one-hot encoding categorical data, removing ID values, and splitting the data into training and testing before standardizing the features. This is performed on the [**final\_merged\_training\_data.csv**](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/ES2DqCsw3Z5Ppf5drlX-RtMB2yHafRLhK6BOEiNmXVHM9Q?e=PzPbdc) and again on the [**final\_merged\_testing\_data.csv**](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/Eb0v_vWqrxtLnFkcYWJp5UwBh5TYho-aNTjG3BsKKm0HkA?e=YDGSa7)**.**
2. EDA
   1. **Load Data:** We used pandas to load the training dataset and inspect its structure. We loaded the training dataset and examined its shape, variable types, and initial class distributions:
      1. print(train\_df.shape)  
         print(train\_df.dtypes)  
         print(train\_df['ADHD\_Outcome'].value\_counts())  
         print(train\_df['Sex\_F'].value\_counts())

This revealed class imbalance, particularly in ADHD\_Outcome and Sex\_F, which informed our use of class-weighted models.

* 1. **Handle Missing Data:** We quantified missingness and dropped columns with more than 50% missing values:
     1. missing\_percent = train\_df.isnull().mean() \* 100  
        train\_df = train\_df.drop(columns=missing\_percent[missing\_percent > 50].index)
     2. Then, we imputed remaining missing values:
     3. Categorical features were filled with "Missing"
     4. Numeric features were filled with the column median

cat\_cols = train\_df.select\_dtypes(include='object').columns  
num\_cols = train\_df.select\_dtypes(include='number').columns  
  
train\_df[cat\_cols] = train\_df[cat\_cols].fillna("Missing")  
train\_df[num\_cols] = train\_df[num\_cols].fillna(train\_df[num\_cols].median())

* 1. **Distribution Analysis:** We visualized key behavioral variables using histograms and boxplots to assess skewness and outliers:
     1. import seaborn as sns  
        import matplotlib.pyplot as plt  
          
        sns.histplot(train\_df['SDQ\_SDQ\_Hyperactivity'], kde=True)  
        plt.title('Distribution: SDQ Hyperactivity')  
        plt.show()  
          
        sns.boxplot(data=train\_df, y='SDQ\_SDQ\_Difficulties\_Total')  
        plt.title('Boxplot: SDQ Difficulties Total')  
        plt.show()
  2. **Correlation and Redundancy:** A heatmap was used to assess correlation among numeric features:
     1. corr = train\_df[num\_cols].corr()  
        sns.heatmap(corr, cmap='coolwarm', center=0)  
        plt.title('Correlation Heatmap')  
        plt.show()
  3. **Group Comparisons:**  
      We explored ADHD and sex group differences using violin plots:
     1. sns.violinplot(data=train\_df, x='ADHD\_Outcome', y='SDQ\_SDQ\_Hyperactivity')  
        plt.title('Hyperactivity by ADHD Diagnosis')  
        plt.show()  
          
        sns.violinplot(data=train\_df, x='Sex\_F', y='SDQ\_SDQ\_Difficulties\_Total')  
        plt.title('Difficulties Total by Sex')  
        plt.show()
     2. These plots suggested interaction effects between ADHD and sex, justifying the use of sex-specific or multi-output modeling.

1. Feature Extraction - PCA
   1. Feature extraction is done on the [**TRAIN\_FUNCTIONAL\_CONNECTOME\_MATRICES\_new\_36P\_Pearson.csv**](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/ETccyRbTsQVJhqJ1sNw_ETwBR84rrHMiLWIi-WvtTH4EHw?e=d5rQut)data, in order to reduce the dimensions of the 19900-column dataset to less than 1200 (the number of rows, or participants in the data).
   2. Remove the ID columns (first column) and use **StandardScaler.fit\_transform** to standardize the data.
   3. To determine the appropriate number of components to reduce the dimensions of the data from 19900 to less than 1200, create a bar plot with a line indicating the cumulative variance with each additional principal component
      1. Do this by using the function **PCA** from **sklearn.decomposition**, applying the standardized data. Get the explained variance ratio by calling **pca.explained\_variance\_ratio** on the PCA data.
      2. Plot the first 1000 components using the function **bar** from **matplotlib.pyplot**
   4. After determining 1000 components as sufficiently reducing the initial data, now we need to run PCA and select the number of components (n\_components) to 1000.
   5. After running the PCA, the resulting data must be converted to a data frame and merged with the original participant\_ids. Save this data as [Reduced\_Dimension\_Data.csv](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EQH1oxMHarxLgs1baffeCr4BHEBQD-SEcqNiE1bpucOHiQ?e=icWnwJ) so that in the future it can be merged with the categorical and numeric datasets.
2. Feature Selection – Top 25 Features
   1. The top 25 features are selected using a Random Forest classifier on the [**final\_train\_df.csv**](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/Ea_L8yNPtEFOrHN5sJIBoSQBnEHU6nwlPXL4VakQm9_ZHQ?e=NQIgYL) file. The columns utilized for this are exclusively the numeric columns, therefore, drop the categorical columns and ID column, replacing any missing values using median imputation.
   2. Split the dataset by sex creating x and y datasets with the y containing the target variables (sex, ADHD) and the x containing the remaining columns.
   3. Using a **RandomForestClassifier** imported from **sklearn.ensemble,** train two separate classifiers, one for each subset of data (female, male).
   4. Extract the feature importance using the **RandomForestClassifier.feature\_importances\_** function, ensuring to convert to a series forplotting purposes.
   5. Plot the top features for each subset of data using **plotly.pyplot.**
3. Feature Engineering
4. **Dataset Merging and Preprocessing:**  
   Our process began by combining three major datasets: [TRAIN\_CATEGORICAL\_METADATA\_new.xlsx](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/ET3oJ5tqqs5FjKpLFkPMotMBSyZ5pn7AzLvHo0hpdvOsCw?e=VADJFg), [TRAIN\_QUANTITATIVE\_METADATA\_new.xlsx](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EUtoaBcNSplDm3JdVwhmyLUBJGiiD-2GeW7umSaAsZ5uTA?e=THQOKG), and [Reduced\_Dimension\_Data.csv](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EQH1oxMHarxLgs1baffeCr4BHEBQD-SEcqNiE1bpucOHiQ?e=uRCj4b), which contained PCA-reduced fMRI features. The PCA reduction, previously completed, reduced the 19,900-dimensional fMRI data into 1,000 principal components using sklearn.decomposition.PCA. All datasets were merged using a standardized participant\_id column. We also removed redundant fields like Unnamed: 0 and extra ID columns. The resulting merged dataset ([final\_model\_data.csv](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/Ecu_ksrKWa1GhPaOGuaVcycBGS8nUSGJB1kPvxeuwLUvhQ?e=TbCbgi)) combined neuroimaging and behavioral or demographic features for all training participants.
5. **Data Cleaning and Scaling:**  
   To ensure clean input to our models, missing values in categorical features were filled with the string "Missing", and numerical columns had missing values imputed using the median. We then applied StandardScaler from sklearn.preprocessing to standardize the numerical features. This preprocessing ensured that all inputs were on the same scale and ready for downstream modeling and feature selection.
6. **Feature Selection: Sex-Specific ADHD and Sex Prediction:**  
   We performed targeted feature selection using RandomForestClassifier from sklearn.ensemble. For ADHD prediction, the training dataset was split by sex. We trained one model using only male participants and another using only female participants to predict the ADHD\_Outcome column. From each model, we extracted the top 20 most important features based on feature importance scores and saved them as [top\_features\_adhd\_male.csv](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/Ee2aeROOpQdEsaFXcJXAKy8BN3r0FAl3mVAA42ad5Bam3g?e=6DVPMo) and [top\_features\_adhd\_female.csv](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EaiUcWJJTshOnIVYy6ueh7kB0qAgalzeHRI6lww54hNG-g?e=hm4em8). In parallel, a third Random Forest model was trained on the full dataset to predict Sex\_F, from which we extracted and saved the top 20 features into [top\_features\_sex.csv](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EegEXEpAts5Fq_E-AVvvk0ABAT-BeiRJ95IkUF9L9lXniQ?e=eWihre).
7. **Shared Feature Consolidation for Multi-Output Modeling:**  
   To prepare a feature set for multi-output classification, we merged the top features from the three previous models (ADHD male, ADHD female, and sex prediction). After removing duplicates and filtering out non-informative metadata like Enroll\_Year and Scan\_Location, we constructed a final consolidated set of 54 unique features. This list was saved in [shared\_top\_features.csv](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/EQXR6IX_LB9AlMkyzhBMG6UBA7N2NC1ZcTCabhC301Iveg?e=qCeo8V) and used as the input for our multi-output SVM classifier to jointly predict ADHD and sex.
8. **Application to Test Data:**  
   The same feature engineering and preprocessing steps applied to the training data were also replicated for the test dataset to maintain consistency. This involved merging the test metadata with the PCA-reduced fMRI data, handling missing values through mean imputation for numerical fields and filling "Missing" for categorical fields, followed by one-hot encoding of categorical variables. The test data was then standardized using the same scaler that was fit on the training set. Finally, we selected the same 54-feature set that was used during training for both ADHD and sex prediction tasks. These steps ensure that the test dataset is processed identically to the training data, supporting valid model evaluation and reproducibility of predictions.
9. **Feature Importance Visualization:**  
   To better understand which features contributed most to predictions, we visualized the top 20 features for each task using bar plots. For ADHD prediction in males, the most important features included SDQ\_Hyperactivity, APQ\_P\_OPD, and several PCA-derived neuroimaging components. These findings supported the role of both behavioral assessments and functional brain features in ADHD classification for males. In contrast, the female subset revealed different top features, such as SDQ\_Conduct and other distinct PCA components, reinforcing the need for sex-specific models. For sex prediction, the feature importance plot highlighted a mix of behavioral, demographic, and fMRI-based features, including parental education and PCA components. These visualizations helped validate the multi-modal modeling approach and informed how different features contribute to different prediction tasks.
10. Model Training & Prediction Generation: MultiOutput SVM
    1. **Model Setup:** We implemented a multi-label classification pipeline using MultiOutputClassifier from sklearn.multioutput with SVC from sklearn.svm as the base model. The SVC was configured with:
       1. RBF kernel: kernel='rbf'
       2. Probability estimates enabled: probability=True
       3. Balanced class weights: class\_weight='balanced'

from sklearn.svm import SVC  
from sklearn.multioutput import MultiOutputClassifier  
  
svc = SVC(kernel='rbf', probability=True, class\_weight='balanced', random\_state=42)  
multi\_clf = MultiOutputClassifier(svc)

* 1. **Data Preparation:** We used standardized feature data (X\_scaled) and selected ADHD\_Outcome and Sex\_F as our outcome variables. We split the data into training and validation sets while preserving class distributions using stratified sampling:
     1. from sklearn.model\_selection import train\_test\_split  
        X\_train\_split, X\_val\_split, y\_train\_split, y\_val\_split = train\_test\_split(  
         X\_scaled, y[['ADHD\_Outcome', 'Sex\_F']], test\_size=0.3, stratify=y[['ADHD\_Outcome', 'Sex\_F']], random\_state=42  
        )
  2. **Model Training:** The multi-output SVM was trained using the training features and corresponding labels:
     1. multi\_clf.fit(X\_train\_split, y\_train\_split)
  3. **Prediction:** After training, predictions were generated on the validation feature set:
     1. y\_pred = multi\_clf.predict(X\_val\_split)
  4. **Evaluation:** We evaluated performance using classification\_report for each target separately:
     1. from sklearn.metrics import classification\_report  
          
        print("📊 ADHD Diagnosis:")  
        print(classification\_report(y\_val\_split['ADHD\_Outcome'], y\_pred[:, 0]))  
          
        print("📊 Sex Prediction:")  
        print(classification\_report(y\_val\_split['Sex\_F'], y\_pred[:, 1]))

This produced precision, recall, F1-score, and accuracy for each class within both targets. These results were later compared to models trained on selected top features and engineered datasets to assess the impact of feature scope and preprocessing.

1. Model Training & Prediction Generation: SingleOutput SVM
   1. Set up a SVM classifier using **SVC** from the **sklearn.svm** package with the parameters for kernel set to ‘rbf’, probability set to True, and class\_weight set to ‘balanced’.
   2. Then use the **svm.fit** function on the training data prepared from the MultiOutput SVM portion for the features (x\_train\_split) and the training data for the target variable ADHD (y\_train\_split[‘ADHD\_Outcome’]). In order to get the predictions for the model, use svm.predict on the testing data (X\_val\_split) to get the ADHD predicted values for each row.
   3. Use **sklearn.metrics classification\_report** in order to assess the performance of the SVM classifier model, feeding the actual results (y\_val\_split[‘ADHD\_Outcome’] and the predicted results output from the **svm.predict**.
   4. The **classification\_report** will output the precision, recall, and F1 scores for both predictions of presence of ADHD (1) and absence of ADHD (0).
   5. Repeat this process by selecting for Female only and Male only subsets of the training data, on the top 25 features extracted from the Multioutput SVM, and on the feature engineered dataset ([final\_model\_data.csv).](https://depauledu.sharepoint.com/:x:/s/DSC672-Capstone/Ecu_ksrKWa1GhPaOGuaVcycBGS8nUSGJB1kPvxeuwLUvhQ?e=TbCbgi)