Classification of Autism Spectrum Disorder using Machine Learning

Utkarsh Patel
Third-year Undergrad
Department of E&ECE, IIT Kharagpur

Mentor

Debasis Samanta
Associate Professor
Department of Computer Science, IIT Kharagpur

Abstract

- In this study, we implemented a data-driven approach to classify ASD patients and typically developing (TD) participants by using resting-state fMRI data.
- SVM and KNN were used for classification purpose.
- Neural Networks were also used (just for comparison)
- Each classifier was fine-tuned via grid search on hyper-parameters via cross-validation.
- Finally, a stacked model was also used comprising of fine-tuned classifier as its base models.

What is Autism Spectrum Disorder?

Autism Spectrum Disorder

- It is a complex developmental condition that involves persistent challenges in social interaction, speech and nonverbal communication, and restricted/repetitive behaviors.
- One in 59 children is estimated to have autism in United States.

Why Machine Learning?

Autism Spectrum Disorder

- There is no medical test for autism.
- Early diagnosis and treatment are important to reducing the symptoms of autism and improving the quality of life for people with autism and their families.
- This procedure usually takes a lot of time like 5-10 years.
- Hence, the objective is to use machine learning algorithms for classification.

Why Functional Connectivity Analysis important?

Role of Functional Connectivity

- Past extensive brain imaging studies have reported that ASD is associated with brain connectivity.
- It has been found that in case of autism, some brain regions have weaker functional connectivity (than normal) which may be responsible for their social behaviour, while some of them are highly correlated (than normal) which may account for their exceptional ability to concentrate.
- Studies have found that most of the gold medallists in IMO, IOI, IPhO, etc. have autism.

The Dataset

ABIDE Dataset

- For this task, we used ABIDE dataset
- It contains aggregated functional and structural brain imaging data collected from laboratories around the world to accelerate our understanding of the neural bases of autism.
- The ABIDE 1 dataset contains fMRI images of 1112 subjects, 539 from individuals with ASD, and 573 from typical controls.

Pre-processing

Pre-processing

- For this task, I used the data provided by NYU only. Reasons being:
  - Data provided from different laboratories have different configuration for their instruments, way of examining patients, etc.
  - Of all the sources, NYU had maximum number of subjects (172)
- For pre-processing, I used CPAC pipeline (as the task involves functional connectivity analysis).

nilearn

- nilearn is the library for doing ‘Statistics for Neuroimaging in Python’
- Uses sklearn as backend
- This library is used in this project for:
  - Extracting pre-processed ABIDE 1 dataset
  - Building connectivity matrices from the time-series signals obtained from fMRI images

Brain MAP / Atlas

Brain Atlas

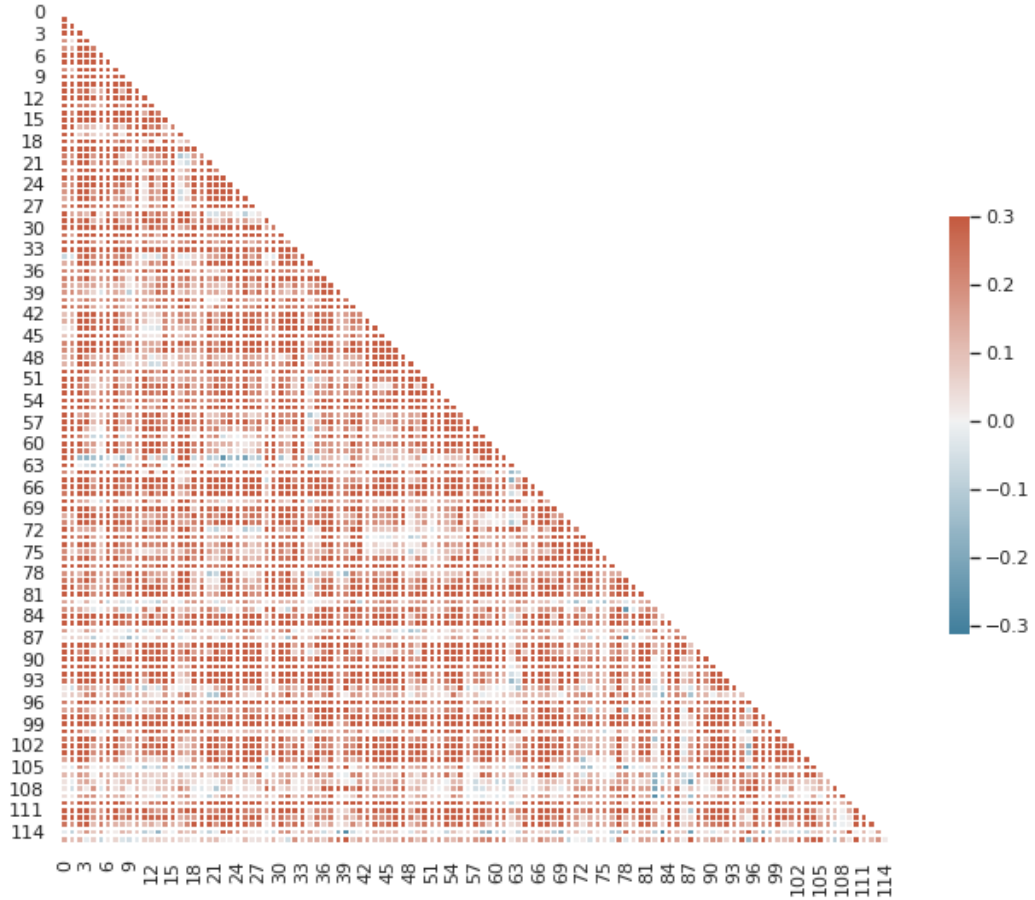
- As per Wikipedia, a brain atlas is composed of serial sections along different anatomical planes of the healthy or diseased developing or adult animal or human brain where each relevant brain structure is assigned a number of coordinates to define its outline or volume.
- Brain atlases are contiguous, comprehensive results of visual brain mapping and may include anatomical, genetical or functional features.
- In this project, I used AAL atlas, which has 116 ROIs (functional clusters).

Extracting data and building connectivity matrices

Snippet

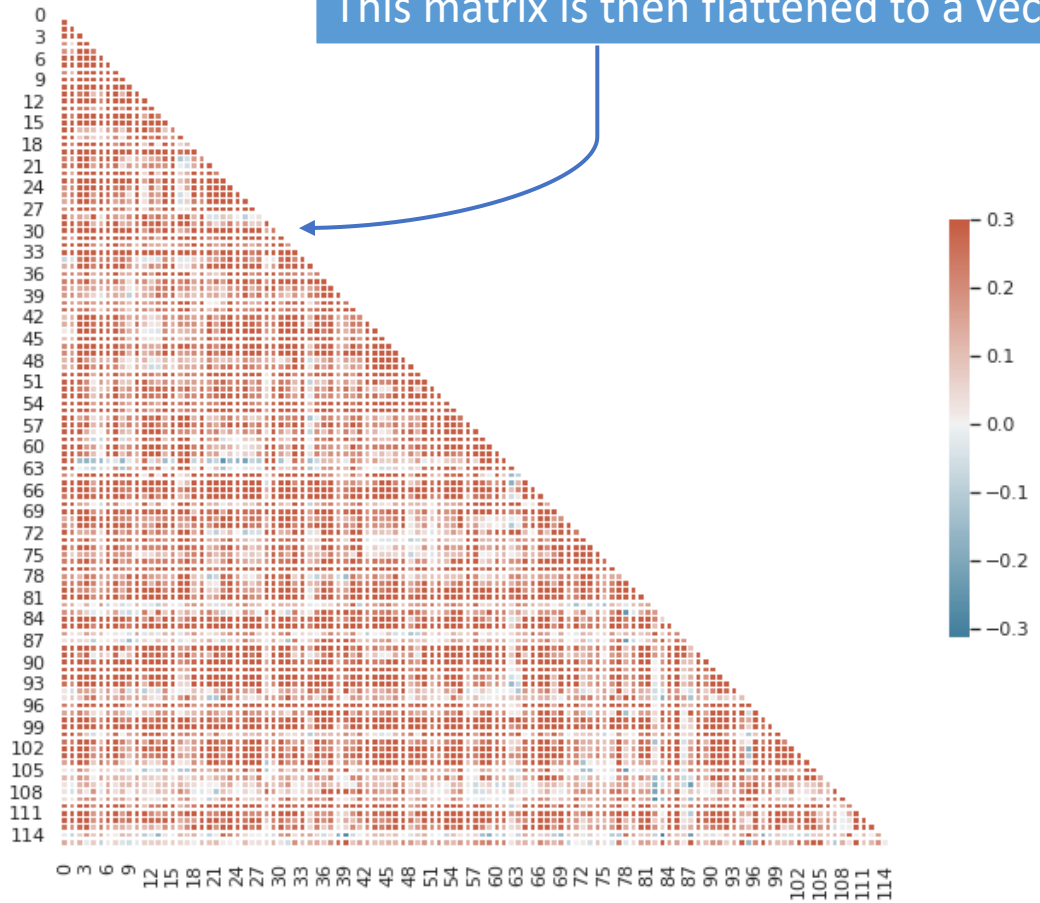
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import nilearn

from nilearn.datasets import fetch_abide_pcp
from nilearn.connectome import ConnectivityMeasure, sym_to_vec

data = fetch_abide_pcp(derivatives=['rois_aal'], SITE_ID=['NYU’])
conn_est = ConnectivityMeasure(kind='correlation')
conn_matrices = conn_est.fit_transform(data['rois_aal'])
```

Connectivity Matrix b/w ROIs given by correlation of its time-series signal



Connectivity Matrix b/w ROIs given by correlation of its time-series signal

This matrix is then flattened to a vector and is used in training the model



Model Selection

Model Selection

- After flattening the connectivity matrix, we have feature vectors $x_i, i = 1, 2, \ldots, 172$ and each x_i is a 6786 dimensional vector.
- We cannot use dimensionality reduction because every correlation is important for classification. (Tried it, however doesn't increase accuracy)
- Tried various algorithm, however only Support Vector Machines and K Nearest Neighbours classifier gave satisfactory results.
- Later tuned the SVM and KNN classifier.

Classification using Support Vector Machines

Support Vector Machines

- Support Vector Machines are a class of supervised learning, in which a hyperplane which produces maximum margin between the class is used for classification.
- It has various hyperparameter:
  - Kernel – linear, radial, sigmoid, polynomial
  - Regularization factor C
  - Fitting parameter γ
- A careful tuning is required to get best results

Tuning SVM

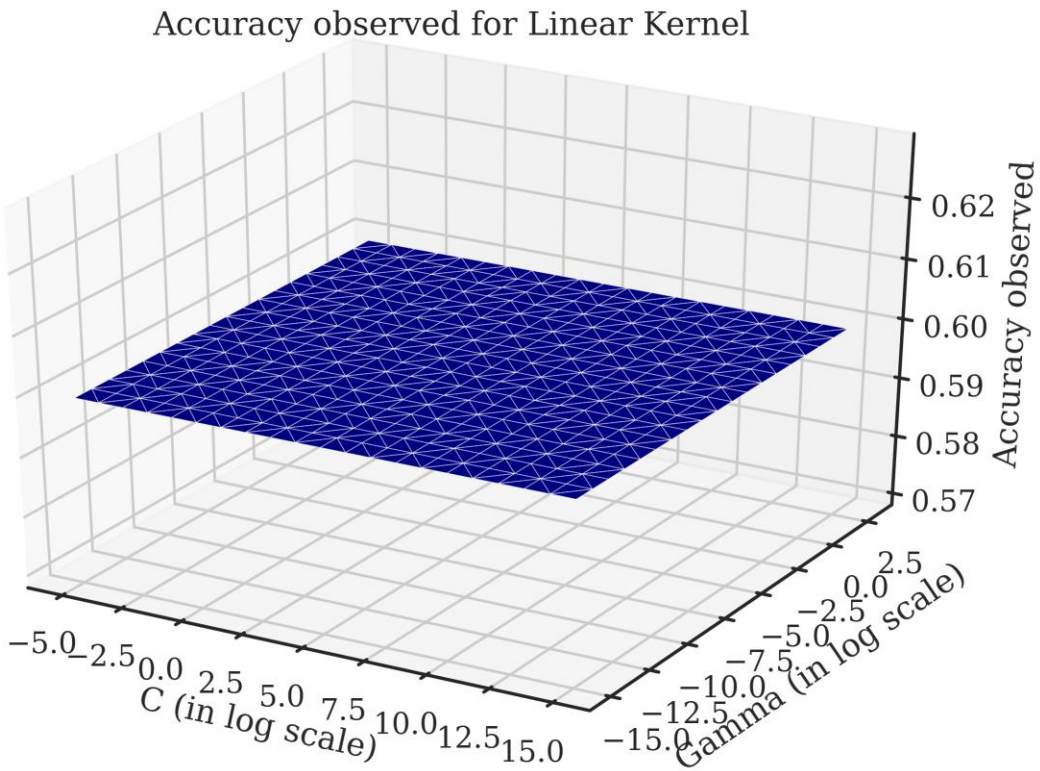
- Tuning regularization constant C
  - For this, we varied C from 2^{-5} to 2^{15}
- Tuning fitting parameter
  - For this, we varied γ from 2^{-15} to 2^{3}

- Basically, a grid search is performed that give us the best possible accuracy

Linear kernel

Linear Kernel

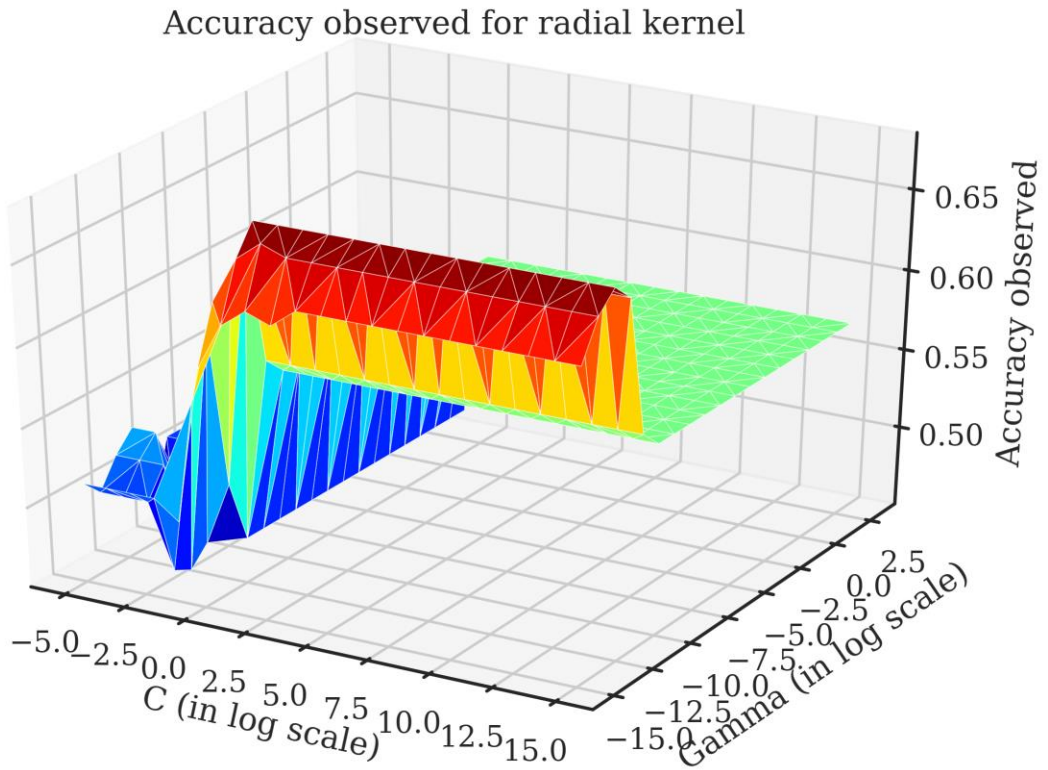
- For linear kernel, the 5-fold cross-validation accuracy was found to be 59.92% for all values of C and γ



Radial kernel

Radial Kernel

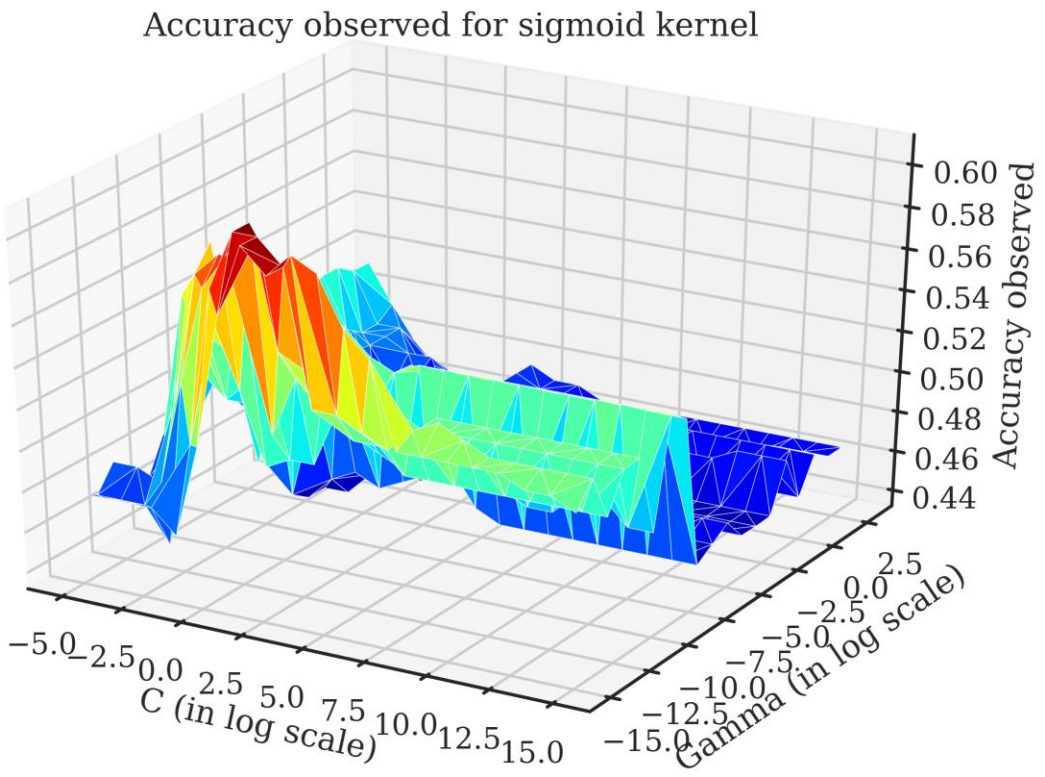
- For radial kernel, the maximum 5-fold cross-validation was found to be 67.97% for $\gamma = 0.000122$ and $C \geq 4$.



Sigmoid kernel

Sigmoid kernel

- The maximum 5-fold cross-validation accuracy was found to be 61.04% for $\gamma = 0.000122$ and $C = 2$.

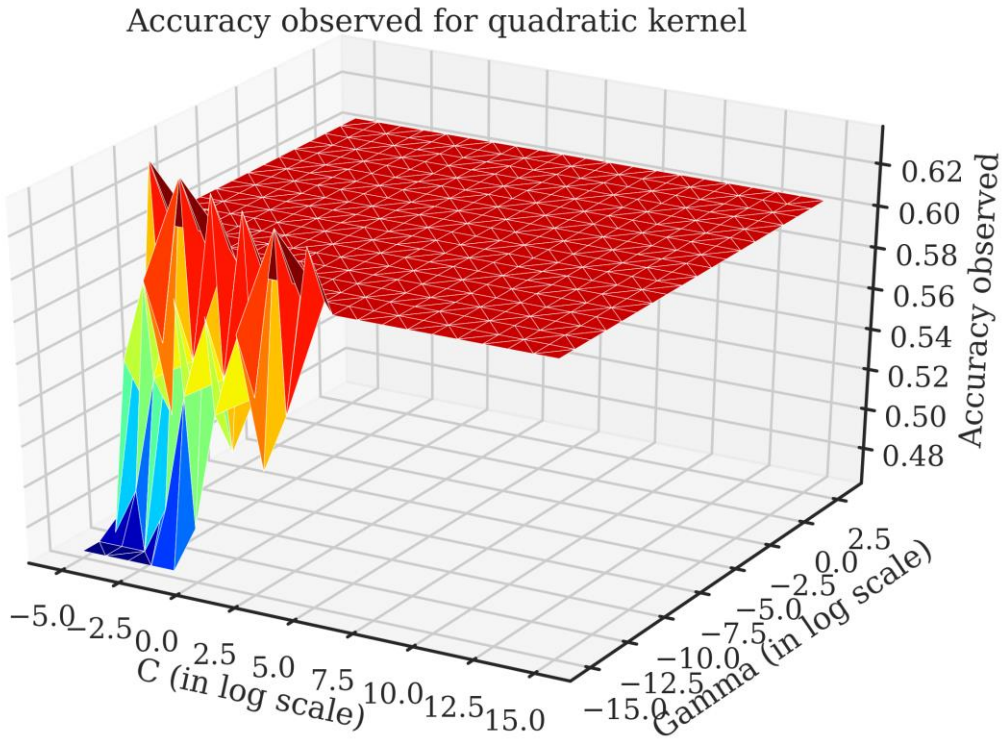


Quadratic kernel

Quadratic kernel

- The 5-fold cross-validation accuracy of 63.41% was observed for values of γ and C satisfying

$$2log_2\gamma + log_2C + 25 = 0 \;\; \rightarrow C\gamma^2 = 2^{-25}$$



Summary for SVM

Summary for SVM

Kernel	Best Accuracy	C	γ
Linear	59.92	$\mathbb{R}$	$\mathbb{R}$
Radial	**67.97**	$\geq 2^1$	2^{-13}
Sigmoid	61.04	2^1	2^{-13}
Quadratic	63.41	2^3	2^{-14}

Classification using K Nearest Neighbours

K Nearest Neighbours

- In KNN, we have two hyper-parameters
  - Value of k
  - Distance function
- sklearn uses *Minkowski* metric for distance function, which has a hyper-parameter p

$$d_p(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_i |x_i - y_i|^p \right)^{\frac{1}{p}}$$

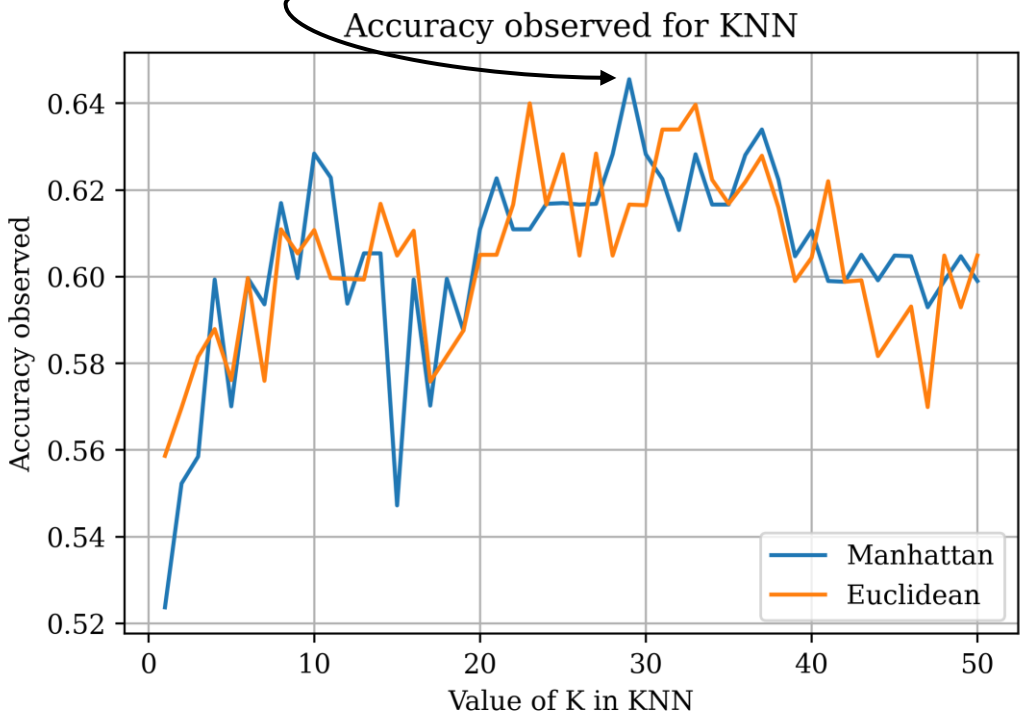
- $p = 1$ corresponds to *Manhattan* distance, while $p = 2$ corresponds to *Euclidean* distance.

Tuning KNN classifier

- KNN classifier is tuned by varying:
  - Value of k from 1 to 50
  - Value of p in $\{1, 2\}$

Accuracy observed for KNN

Optimal Hyper-parameters
$k = 29, p = 1$



p	Best Accuracy	k
1	**64.55**	29
2	63.99	23

SVM vs KNN

SVM vs KNN

- The best accuracy observed for SVM is 67.97 for radial kernel.
- The best accuracy observed for KNN is 64.55 for $k = 29$, $p = 1$.

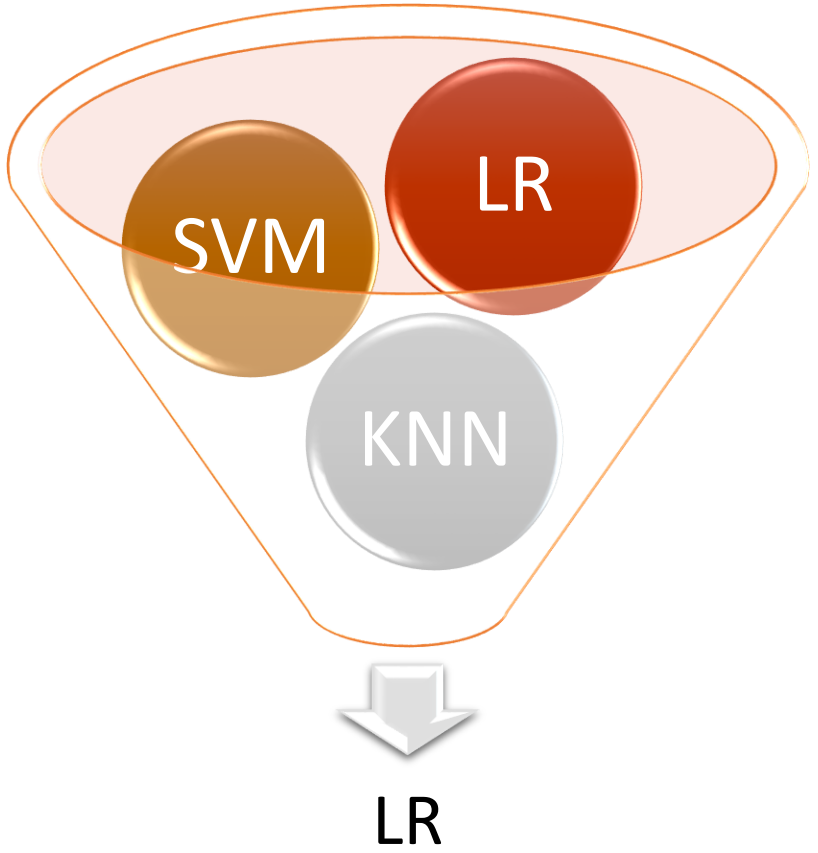
- Hence, SVM outperforms KNN over NYU data set.

SVM + KNN
Stacked Model

Stacked Model

- In stacked ensemble learning, we have
  - Base models (level 0)
  - Meta models (level 1)
- I used the fine-tuned SVM, KNN and logistic regression model as base models
- Logistic regression was also used in meta model

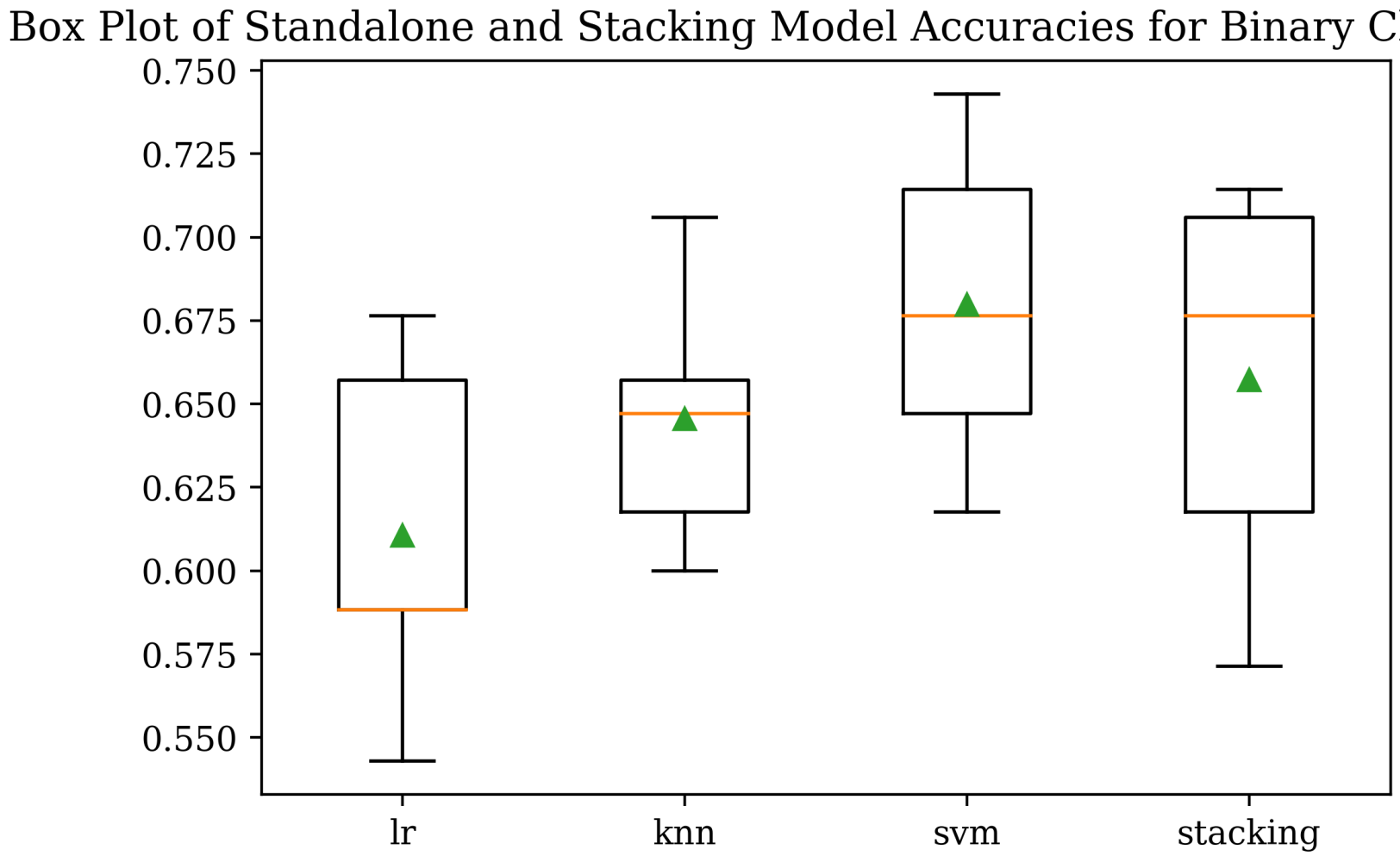
Stacked Model Architecture



Base Models

Meta Model

Performance of Stacked model



Standalone fine-tuned SVM classifier outperforms the stacked model

Classification using Neural Networks

Neural Networks

- The best ML classifier of the task is SVM giving accuracy as high as 67.97.
- Let’s see whether Neural Networks can outperforms the SVM or not.

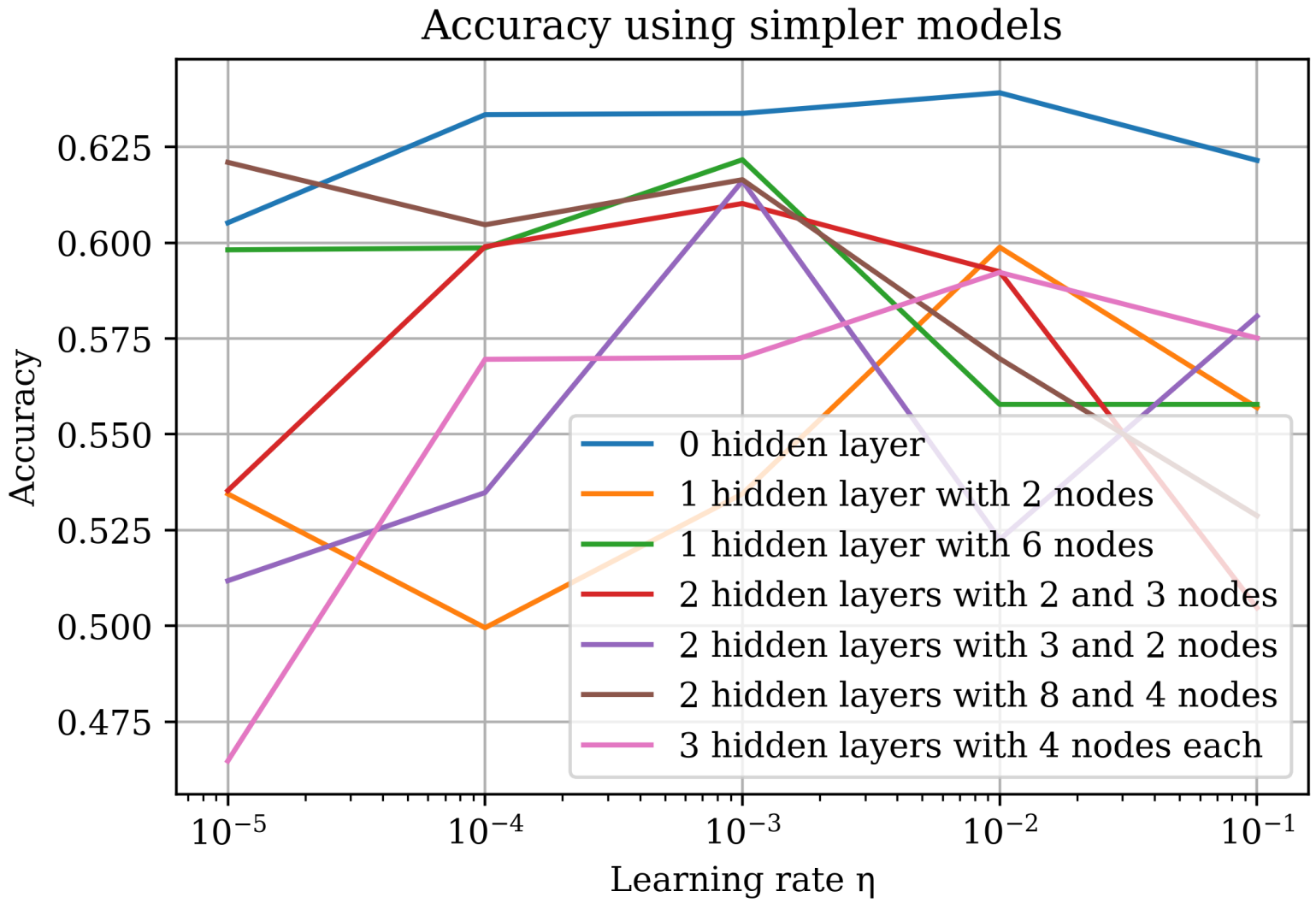
Tuning Neural Networks

- Neural Networks have a large number of hyper-parameters
  - # of hidden layers
  - # of nodes in each hidden layer
  - Non-linear activation used in hidden layers
  - Learning rate
  - Optimization algorithms (Adam, Momentum, RMSProp, Adagrad)
  - Regularization algorithms (Dropout, L2-regularization)
  - .
  - .
  - .

Tuning Neural Networks

- For this task, I am using sklearn's MLP API.
- Some background information:
  - It uses 'Adam' optimizer by default.
  - It uses SGD for optimization.
  - We can vary the (initial) learning rate and the architecture
  - I trained the model for 3000 epochs.

Performance of NN



We can observe that model with zero hidden layer (a logistic regression) is performing far-better as compared to complex models over a wide range of learning rates. However, the peak accuracy is 63.75 which is less than that observed for SVM (67.97)

Conclusion

Conclusion

- It is clear than a standalone fine-tuned SVM classifier is outperforming *stacked* model as well as the neural networks for this task.
- However, we may be able to further push the performance of the classifier if we can figure out a new way to extract functional connectivity instead of relying on correlation-based approach.

Future Work

- Apply new ways to extract functional connectivity.
- Check the proposed algorithm with different people according to their gender and age.

Thank You.