Selfish Peer detection in P2P Network using ML

Mentor
Dr. Kunwar Pal
Assistant Professor, CSE

Students
Aditya Wankhede 17103094
Rahul Shah 17103067

A Peer to Peer network

Peer-to-peer (**P2P**)[1] computing or networking is a distributed application architecture that partitions tasks or workloads between peers. Peers make a portion of their resources, such as processing power, disk storage or network bandwidth, directly available to other network participants, without the need for central coordination by servers or stable hosts. Peers are both suppliers and consumers of resources.

Factors affecting the performance of a P2P network

- Chunk Scheduling: Actual data to be shared.
- Peer Scheduling: Selecting a particular peer for uploading the content.
- Device Identification.
- Selfish Peer.

Selfish Peer and their challenges

A selfish peer preserves its own resources and uses the resources of other peers and consumes there resources i.e. it does not use its upload bandwidth to upload the content.
More number of selfish peer degrade the performance of the network, so a P2P network can only a handle some extent of selfish peers.

According to case study
70% of all files is shared by 5% of peers and 98% of all files are shared by 25% peers.

These free riders are a key obstacle and its presence significantly degrade the overall performance of the network.

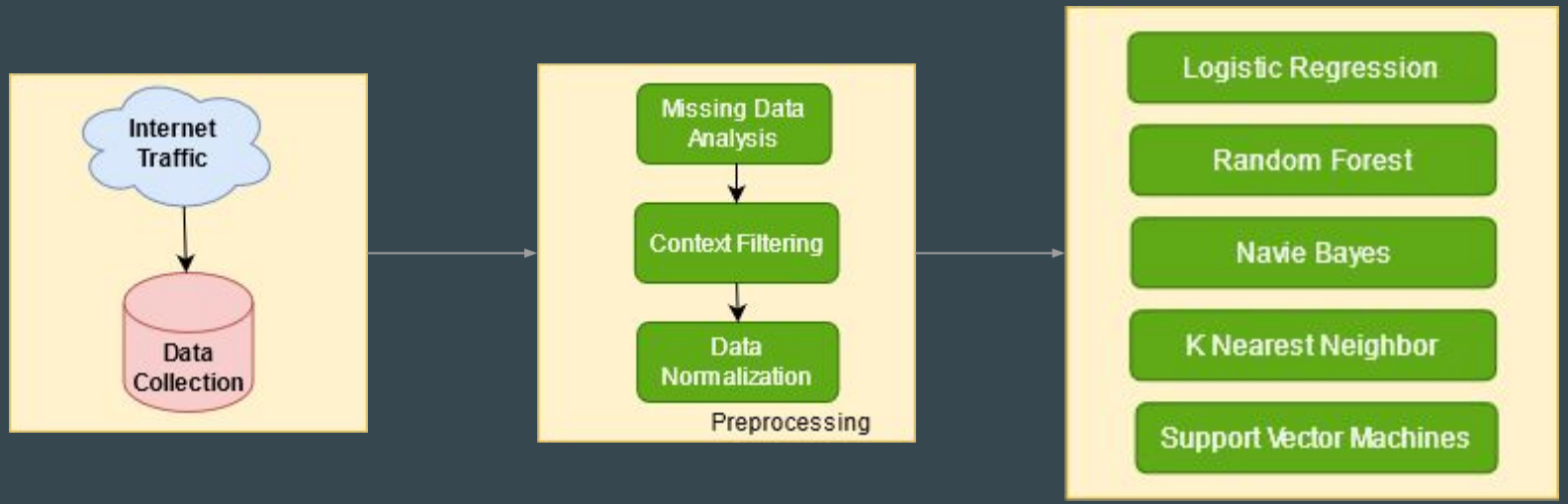
Our approach towards a solution

✅ Build a machine learning model that will help us to predict which peer is acting as a selfish peer.

✅ Training a model using different classifier algorithms and comparing their accuracies for obtaining best performing model.

Optimizing the models further using nature inspired ML algorithms.

Continued.


Internet Traffic
Data Collection
Missing Data Analysis
Context Filtering
Data Normalization
Preprocessing
Logistic Regression
Random Forest
Navie Bayes
K Nearest Neighbor
Support Vector Machines

Dataset

We obtained the dataset from NIT Sikkim.

Dataset had the information about the flow of data between two peers in a Peer to Peer to network. The Column in the data set were:

- Source: IP address of the source peer. (String, Categorical Variable)
- Destination: IP address of the destination peer. (String, Categorical Variable)
- Protocol: Protocol used. (String, Categorical Variable)
- Src_port: Source port used for communication.
- Dest_port: Destination port used for communication.

Dataset

- Flow_count: Total number of flow during the connection.
- Flow_size: Amount of data transferred.
- Pkt_size_of_first_flow: Packet size of the first flow.
- Flow_duration: Duration of the flow.
- First_flow_inter_arrival_time: Arrival time for the first flow.

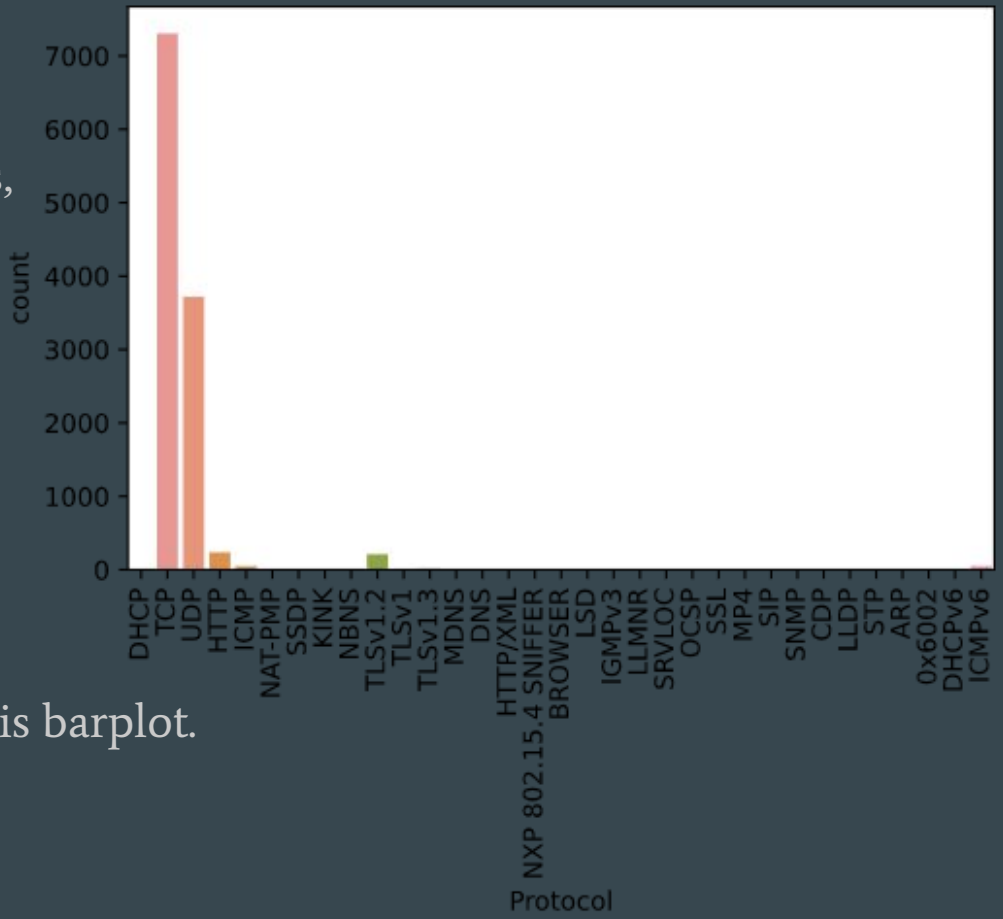
Source	Destination	Protocol	Src_port	Dest_port	Flow_count	Flow_size	pkt_size_of_first_flow	Flow_duration	first_flow_inter_arrival_time
0.0.0.0	255.255.255.255	DHCP	-1	-1	279	95418	342	3335.924975	0.005128
1.1.247.102	103.119.242.114	TCP	-1	-1	5	330	66	293.814945	0.135032
1.1.247.102	103.119.242.114	TCP	4001	4001	1	66	66	254.990643	0.747853
1.156.246.187	103.119.242.114	UDP	37186	32478	1	331	331	1821.708214	0.227403
1.164.136.162	103.119.242.114	UDP	7536	32478	1	143	143	950.648788	3.748469

Some Insight into data

Even though the dataset has 11682 rows, the unique values in the dataset are:

- IPs: 1360
- Protocols: 33
- Ports: 5987

Out of 33 protocols only 3 protocols namely TCP, UDP and HTTP have a significant contribution as seen from this barplot.



Rules

Rules used to determine the selfish peer were:

- Flow_size < T1
- Flow_duration < T2
- Src_port is one of T3
- Size of First flow > T4

Thresholds

Let n be the number of rows in the dataset then thresholds are as follow:

- T1

$$\frac{\sum_{i=1}^{n} FlowSize[i]}{n}$$

- T2

$$\frac{\sum_{i=1}^{n} FlowDuration[i]}{n}$$

- T3 [1074, 1392, 49160]

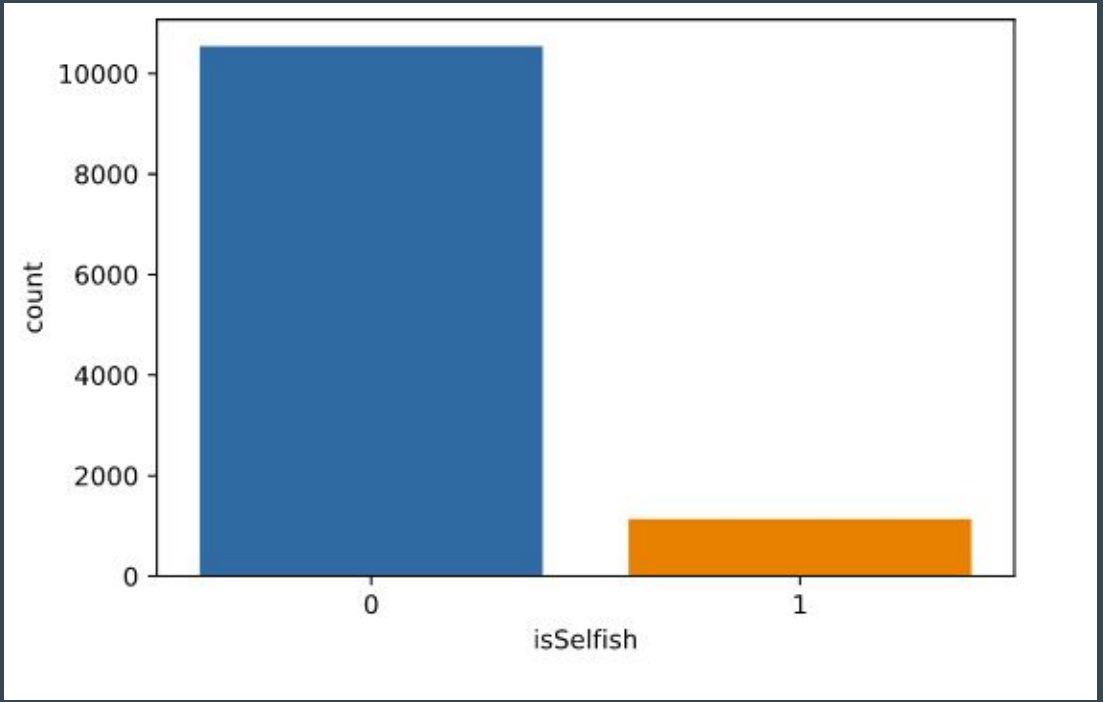
T4

$$\frac{\sum_{i=1}^{n} \frac{PktSizeOfFirstFlow[i]}{FlowSize[i]}}{n}$$

Selfish Peers

After applying the rules.

There were 1133 Selfish flows and 10549 non Selfish flows



Binary Classification Problem

Since the problem is to predict whether a peer a selfish or not, it can be classified as a binary classification problem. We can use Binary Classification Algorithms to build and train ML models.

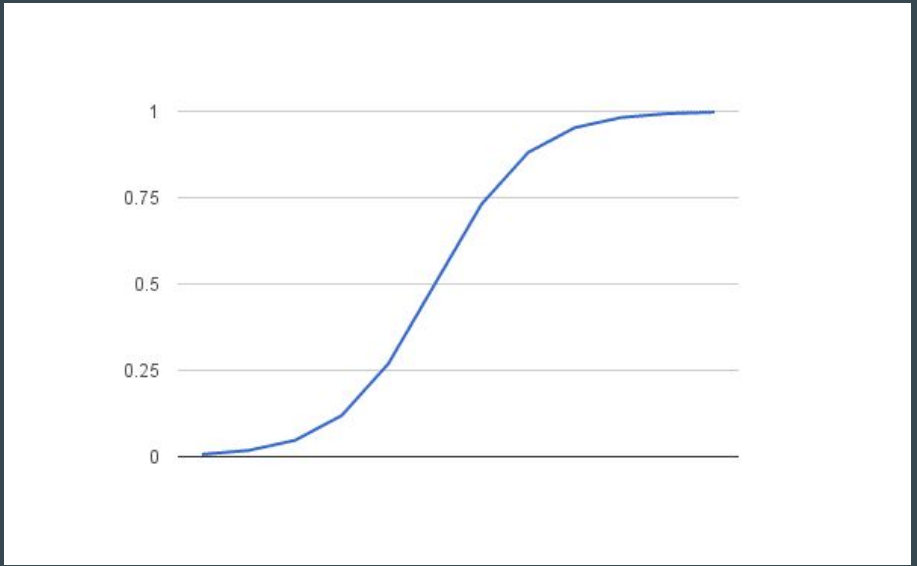
- Logistic Regression
- Random Forest
- Naive Bayes
- K nearest neighbor
- Support Vector Machines.

Implementation Details

- Programming Language: Python
- Libraries:
  - Sci-kit learn (sklearn) for Classification algorithms implementation.
  - Seaborn and Matplotlib for graphs and visualization
- Jupyter Notebook to run code.

Logistic Regression

The logistic function, also called the sigmoid function is an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.



Logistic Regression Classifier accuracies

Features	Accuracy
SrcPort, DestPort, FlowCount, FlowSize, PktSize, FlowDuration, FirstFlowArrival, PktToFlow	0.9024
SrcPort, FlowCount, FlowSize, FlowDuration, FirstFlowArrival, PktToFlow	0.9024
SrcPort, FlowCount, FirstFlowArrival, PktToFlow	0.9024

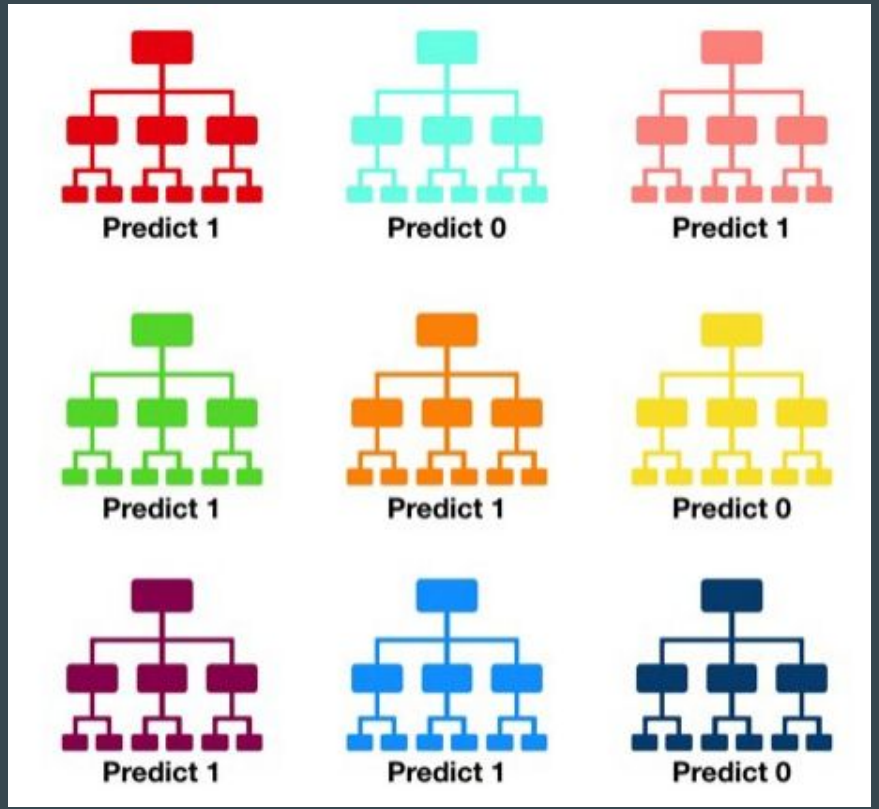
Random Forest

Decision Trees

In decision trees the data is continuously split according to a certain parameter.

Random Forest

It consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction



Random Forest Classifier accuracies

Features	Accuracies
SrcPort, DestPort, FlowCount, FlowSize, PktSize, FlowDuration, FirstFlowArrival, PktToFlow	0.9563
SrcPort, DestPort, FlowSize, PktSize, FlowDuration, FirstFlowArrival	0.9580
SrcPort, DestPort, FlowDuration, FirstFlowArrival	0.9717
SrcPort, DestPort, FlowDuration	0.9897

Naive Bayes

Naive Bayes classifiers are a collection of classification algorithm based on Bayes' Theorem.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

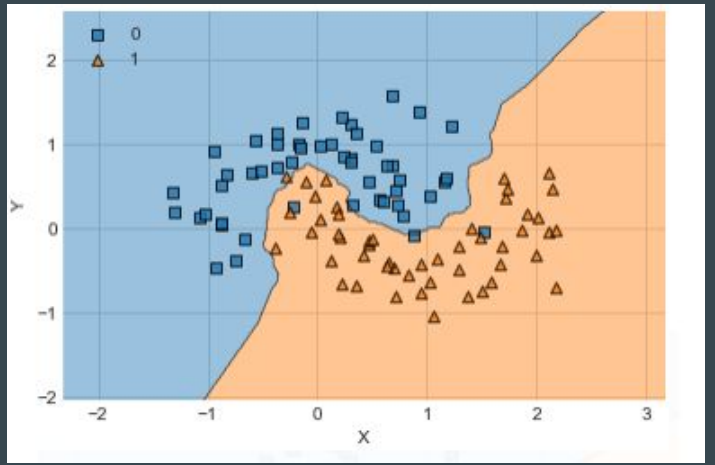
Naive Bayes Classifier accuracies

Features	Accuracies
SrcPort, DestPort, FlowCount, FlowSize, PktSize, FlowDuration, FirstFlowArrival, PktToFlow	0.4009
SrcPort, DestPort, PktSize, FlowDuration, FirstFlowArrival, PktToFlow	0.8292
SrcPort, DestPort, PktSize, FirstFlowArrival	0.9024

KNN

K-NN algorithm stores all the available data and classifies a new data point based on the similarity.

- Calculate the Euclidean distance of K number of neighbors
- Take the K nearest neighbors and count the number of the data points in each category.
- Assign the new data points to that category for which the number of the neighbor is maximum.



KNN Classifier accuracies

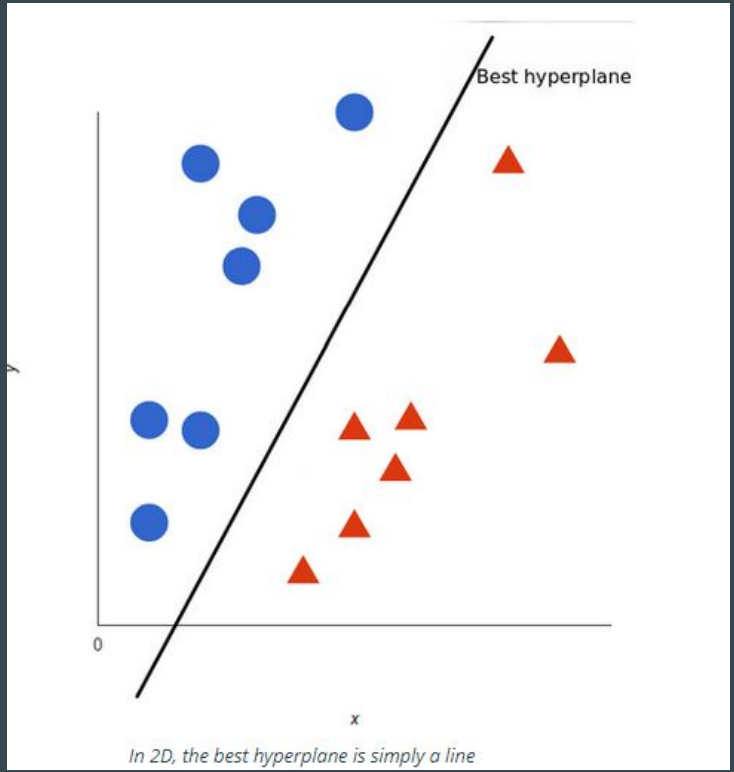
Features	Accuracies
SrcPort, DestPort, FlowCount, FlowSize, PktSize, FlowDuration, FirstFlowArrival, PktToFlow	0.9345
SrcPort, DestPort, FlowCount, FlowSize, FlowDuration, PktToFlow	0.9362
SrcPort, DestPort, FlowSize, PktToFlow	0.9020

SVM

A Support Vector Machine(SVM) is a supervised learning algorithm that sorts data into two categories.

It is trained with a series of data already classified into two categories,building the model as it is initially trained.

The task is to determine which category a new data point belongs in.



In 2D, the best hyperplane is simply a line

SVM

Features	Accuracies
SrcPort, DestPort, FlowCount, FlowSize, PktSize, FlowDuration, FirstFlowArrival, PktToFlow	0.9024
SrcPort, DestPort, FlowCount, FlowSize, FirstFlowArrival, PktToFlow	0.9024
DestPort, FlowCount, FlowSize, PktToFlow	0.9024

Classifier Comparisons

	Logistic Regression	Random Forest	Naive Bayes	KNN	SVM
All Features	0.9024	0.9563	0.4009	0.9345	0.9024
Selective Features	0.9024	0.9897	0.9024	0.9362	0.9024

Improving Accuracies

- Adding more data.
- Hyperparameter optimization using Nature inspired algorithm.

Final Evaluation

Insights when more data was added.

	Before	After
Total rows	11682	137181
Unique IPs	1360	20668
Protocol [TCP, UDP, HTTP]	[7360, 3720, 237]	[55082, 75540, 1208]
isSelfish [Yes, No]	[1133, 10549]	[9559, 127622]

Effect of adding new data

Logistic Regression

	Previous	**Now**
Accuracy	0.9024	**0.9277**
Selected Features	SrcPort, FlowCount, FirstFlowArrival, PktToFlow	'Src_port', 'Dest_port', 'Flow_count', 'Flow_size'

Naive Bayes

	Previous	**Now**
Accuracy	0.9024	**0.9277**
Selected Features	SrcPort, DestPort, PktSize, FirstFlowArrival	'Src_port', 'Dest_port', 'Flow_duration', 'pkt_size_to_flow_size'

Random Forest

	Previous	**Now**
Accuracy	0.9877	**0.9424**
Selected Features	SrcPort, DestPort, FlowDuration	'Src_port', 'Dest_port', 'Flow_duration', 'first_flow_inter_arrival_time

KNN

	Previous	**Now**
Accuracy	0.9362	**0.9470**
Selected Features	SrcPort, DestPort, FlowCount, FlowSize, FlowDuration, PktToFlow	'Src_port', 'Dest_port', 'Flow_count', 'Flow_size', 'Flow_duration', 'pkt_size_to_flow_size'

SVM

	Previous	**Now**
Accuracy	0.9024	**0.9277**
Selected Features	DestPort, FlowCount, FlowSize, PktToFlow	'Src_port', 'Flow_count', 'Flow_size', 'Flow_duration'

Comparing accuracies so far

	Logistic Regression	Random Forest	Naive Bayes	KNN	SVM
All Features	0.9024	0.9563	0.4009	0.9345	0.9024
Selective Features	0.9024	0.9897	0.9024	0.9362	0.9024
All Features (After)	0.92765	0.9324	0.9051	0.9397	0.9277
Selective Features (After)	**0.92766**	**0.9424**	**0.9277**	**0.9470**	**0.9277**

Further Optimization

For each of the supervised binary classification algorithms above, we optimised the parameters to obtain the best possible accuracies.

The classification can be further improved if we can optimize the hyperparameters.

Hyperparameters: Hyperparameters are those parameters that do not depend on the data, but still play a significant role in training the model.

To optimize the hyperparameters we use Particle Swarm Optimization (PSO).

Particle Swarm Optimization

The particle swarm optimization (PSO) algorithm, proposed by Kennedy and Eberhart, is a metaheuristic algorithm based on the concept of swarm intelligence capable of solving complex mathematics problems existing in engineering.

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values

1. The best solution (fitness) the particle has achieved so far. This value is called pbest.
2. Another "best" value is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population.

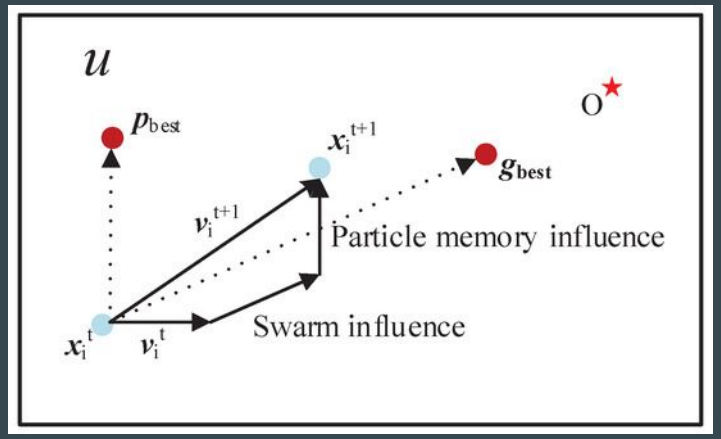
PSO

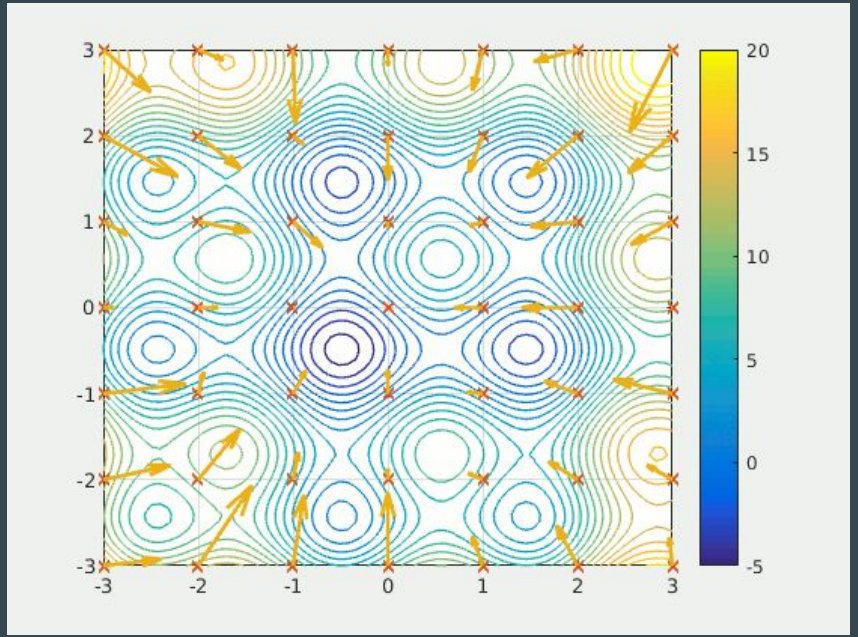
Formula for PSO implementation

1. v[] = v[] + c1 * rand() * (pbest[] - present[]) + c2 * rand() * (gbest[] - present[])
2. present[] = persent[] + v[]

v[] is the particle velocity, persent[] is the current particle (solution). pbest[] and gbest[] are defined as stated before. rand () is a random number between (0,1). c1, c2 are learning factors. usually c1 = c2 = 2.

PSO Visualisation





Accuracies after optimisation with PSO

	Logistic Regression	Naive Bayes	KNN
Selective Features (Before)	0.92766	0.9277	0.9470
Selective Features	0.6317	0.9277	0.9585

Result

In case of Logistic Regression, accuracy actually decreased by a significant amount, it is because more iterations are need to get the best accuracy. But number of iterations required is undefined and the process is costly to perform hit and trail.

In case of Naive Bayes accuracy did not improve. Maybe because there is no more optimisation available for Naive Bayes or maybe because there is a need of more iterations in PSO.

In case of KNN the accuracy increased from 0.9470 to .9585 which is maximum. So the generated KNN classifier is the best model for detection of selfish peer in P2P network.

Deliverables

- KNN model with an accuracy of 95.85%
- A comparative analysis of supervised classification algorithms.
- Impact of performing PSO on the above mentioned classification algorithms.

Project Extensions

- The project can be further extended by building an Application Programming Interface (API). The API can implement an endpoint that will predict whether a peer is selfish or not based on the data provided. The API can be further extended to allow the user to use any of the above mentioned algorithms.
- Like Particle Swarm Optimization any other nature inspired algorithms can be used to optimize the models and its impact on optimizing any classification algorithm can be studied.

References

1. Peer to Peer Network
   https://en.wikipedia.org/wiki/Peer-to-peer
2. Logistic Regression
   https://machinelearningmastery.com/logistic-regression-for-machine-learning/
3. KNN visualisation
   https://towardsdatascience.com/knn-visualization-in-just-13-lines-of-code-32820d72c6b6
4. Random Forest
   https://towardsdatascience.com/understanding-random-forest-58381e0602d2
5. SVM
   https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/

Thank you.