Materials and methods

This section is divided into the following subgroups:

- Evaluation of factors that could determine the essentiality of a gene or protein and acquiring of such data (parameter selection).
- Consolidation of the data into a single dataset and pre-processing of the data
- Testing the predictability of individual parameters and then testing the predictability of the entire dataset
- Evaluation of performance

Parameter selection

The following parameters were considered and their individual features selected

1. **Network topological data**

Proteins play major roles in all cellular processes: they regulate almost every process in the cell , are responsible for creation of signalling cascades, may act as selective transporters on the cell membrane, help to accelerate chemical reactions, and so on. In most of these tasks, proteins never work alone, rather they work in concert, by creating complexes, modifying one another while processing, and transporting other proteins to desired sub-cellular locations.

These protein complexes act as small machineries that perform a variety of important tasks in the cell. They execute important functions like cell metabolism, signal transduction, DNA transcription and duplication, DNA damage repair, etc. Identifying and characterizing the full repertoire of these cellular machineries and the interplay between them are of crucial importance for understanding the functionality of a living cell.

A deeper insight in the protein-protein interactions reveal that often proteins are multifunctional and thus a single protein may be member of more than one complex. This leads to the notion that the cellular mechanisms are also highly connected, than individual metabolisms working alone.

The essential proteins are generally multifunctional and thus their interactivity with other proteins are supposed to be high compared to the non-essential proteins. Thus a study of pattern and extent of interact ability of a protein may indicate its extent dispensability in the cell.

Large-scale identification of protein-protein interactions

Works by Mewes et al.(1998),Xenarios et al. (2000) were the first ones attempting to create a proteome level view of protein- protein interactions (PPI). They gathered data from different literature describing small interactomes and aggregating the knowledgebase. Uetz et al., 2000 had the first attempt of elucidation of PPI in a systematic manner. He created yeast two-hybrid system. This is a ‘bait’ and ‘prey’ approach where a DNA binding domain is fused to a ’bait’ protein and a matching transcription activation domain is fused to a library of ’prey’ proteins. In each case of detection of interactos, one pair of specific bait and prey proteins are introduced into a yeast cell using standard genetics techniques.The system is designed in such a manner that, if the bait and prey proteins physically interact, they a reporter gene is transcribed. Affinity purification approach were developed later to assess the PPI in a large scale where each prey protein is fused to Tandem Affinity Purification (TAP) tag and then introduced to the yeast cell. This tag purifies the protein with all its interaction partners. Then mass spectrometry is utilised to identify the interaction partners that were captured with the bait protein. This method is very much useful for identification of stable complexes.in 2002, the initial works by Gavin et al., Ho et al. had covered small datasets and later success of the system had paved the way towards large scale identification of PPIs , almost covering the entire yeast proteome(Gavin et al., 2006; Krogan et al., 2006).

Protein Interaction Networks

The elucidation of connectivity of proteins in terms of their function had paved the way towards Protein interaction networks (PIN) as deluge of data from high throughput systems were available. Implementation of mathematical graph theory in PPI had created a new field of study called systems biology. in graph theory , *graphs* are mathematical structures used to model pairwise relations between objects. A graph here consists of *vertices*, *nodes*, or *points* which are connected by *edges*, *arcs*, or *lines*. When two proteins are connected they may form an undirected graph, as there is no distinction between the two vertices associated with each edge. Erdos and Renyi's (1960) work on random graph model had paved the way to create large scale PINs in which each two nodes are connected with an edge with probability p.Later Barabasi and Albert (1999) had demonstrated that biological networks fit a power law distribution, where most of the proteins interact with a small number of partners, and a few proteins (termed hubs) interact with a large number of partners. The analysis of these biological networks reveal many useful insights.

Essential genes in PPI networks

In 2009, Gabriel Del Rio et al. investigated the detectability of essential genes through their properties in the cellular PIN which is different from non-essential genes.They opined that centrality measures of the networks may reveal the complex nature of functioning of essential genes. Hahn MW et al. (2005) also had shown the effect of centrality in here eukaryotic PINs.

In our study, few parameters of yeast PPI were analysed through centrality measures to create a machine learning based predictive model.

Parameters considered for machine learning framework

The consideration of parameters revolve around the ideation that essential proteins, hold a central position in PPI and thus their centrality values will considerably differ from the non-essential proteins.

The following measurements of centrality were considered:

Degree Centrality:

It is defined as the number of links incident upon a node (i.e., the number of ties that a node has).

The degree centrality of a vertex v, for a given graph G:=(V.E) with |V| vertices and |E| edges, is defined as

$$C_D(v) = \deg(v)$$

Betweenness centrality

Betweenness centrality is an indicator of a node's centrality in a network. It is equal to the number of shortest paths from all vertices to all others that pass through that node. A node with high betweenness centrality has a large influence on the transfer of items through the network, under the assumption that item transfer follows the shortest paths.

Closeness

In a connected graph, the closeness centrality (or closeness) of a node is a measure of centrality in a network, calculated as the sum of the length of the shortest paths between the node and all other nodes in the graph. Thus the more central a node is, the closer it is to all other nodes.

The expression of closeness centrality is:

$$C(x) = \frac{1}{\sum_y d(y,x)}.$$

Where, d(y,x) is the distance between vertices x and y

Eigenvector Centrality (also called eigencentrality)

It is a measure of the influence of a node in a network. It assigns relative scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes.

For a given graph G:=(V,E)} G:=(V,E) with |V| vertices let A = (a_{v,t}) be the adjacency matrix, i.e. a_{v,t} = 1 if vertex v is linked to vertex t, and a_{v,t} = 0 otherwise. The relative centrality score of vertex v can be defined as:

$$x_v = \frac{1}{\lambda} \sum_{t \in M(v)} x_t = \frac{1}{\lambda} \sum_{t \in G} a_{v,t} x_t$$

where M(v) is a set of the neighbors of v and lambda is a constant. With a small rearrangement this can be rewritten in vector notation as the eigenvector equation

$Ax=\lambda x$.

Method of obtaining network topological data

Obtaining PPI Data

- Yeast protein –protein interaction data were downloaded from Biogrid database (https://thebiogrid.org/). It is an interaction repository with data compiled through comprehensive curation efforts (Tyers M et al.,2006).
- The data downloaded was of version BIOGRID-ALL-3.2.102. The data were downloaded as a mitab file and proteins related to *S. cerevisiae* were selected from that. Biogrid separately supplies an organism list for that purpose.
- The data were formatted in columns containing Interactor A and interactor B
- Pajek a program for large network analysis was downloaded from http://vlado.fmf.uni-lj.si/pub%20/networks/pajek/default.htm .
- The resulting tab delimited file was a read through Pajek. The data were filtered to match the other genes under study and from this subset a network file was created having an extension .net .
- The data were further channelized through igraph package (Csardi S.et al.,2006). It is a collection of network analysis tools with the emphasis on efficiency, portability and ease of use.The R programming interface of igraph was used in this study.

- First the .net file was read into R through igraph package.
- Then the graph object was created . In the graph, the number of vertices and number of edges were 5126 24840 respectively.
- The different measures of centrality : Degree, Beweenness, Closeness and Eigenvector were computed through R programming environment.
- The following box gives the R programs and commands used for the purpose:

```
>library (igraph)

> scere_graph<- read_graph("scere_final_0305.net", format="pajek")

># this command creates the graph reading it as a Paject network file)

> summary(scere_graph)
IGRAPH U-W- 5126 24840 --
+ attr: id (v/c), weight (e/n)
> closeness(scere_graph, vids = V(scere_graph), mode = c("all") ->closeness_scere

> betweenness(scere_graph, v = V(scere_graph), directed = FALSE, weights = NULL, normali
zed = FALSE)
 ->betweenness_scere

> degree(scere_graph, v = V(scere_graph), mode = "all", loops = TRUE, normalized = FALSE)
-> degree_scere
>eigen_centrality(scere_graph, directed = FALSE, scale = TRUE, weights = NULL,options = ar
pack_defaults)-> eigen
```

- Neighbourhood analysis were done for the same data through Pajek. The parameters calculated here are
- Total number of neighbors
- First degree neighbourhood size and
- Second degree neighbourhood size

The resulting data were consolidated along with the data for information about each candidate being essential protein or not and were formatted into a csv file.

The first 20 rows of the data are furnished here for illustrative purpose:

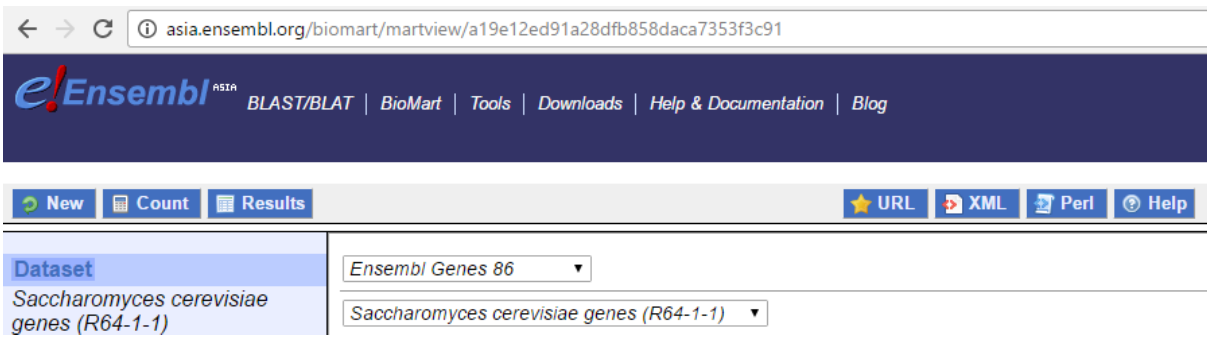
ID	Betweenness	Closeness_all	Degree all	Eigen	k_nbor all	N-size1	N -size2	Essentiality
YAL002W	3565.715	0.284801	8	0.05658	3	9	311	0
YAL007C	16185.61	0.294473	25	0.028054	3	26	354	0
YAL010C	1413.266	0.269198	4	0.018303	4	5	127	0
YAL012W	6882.895	0.281516	5	0.02165	4	6	398	0
YAL017W	2085.567	0.296826	12	0.068302	3	13	447	0
YAL019W	1933.118	0.281408	6	0.01707	3	7	343	0
YAL021C	17310.38	0.31669	40	0.303465	3	41	723	0
YAL022C	2159.115	0.246264	5	0.002842	4	6	149	0
YAL023C	0	0.220639	1	0.000272	4	2	39	0
YAL024C	795.1337	0.285261	5	0.028983	3	6	369	0
YAL026C	215.0488	0.240137	3	0.003408	4	4	34	0
YAL029C	4244.607	0.30329	16	0.061599	3	17	490	0
YAL031C	388.2568	0.284028	3	0.025554	3	4	394	0
YAL032C	13208.07	0.296671	22	0.04311	3	23	494	1
YAL033W	497.7513	0.227011	2	0.000998	4	3	16	1
YAL035W	7931.164	0.291144	18	0.047392	2	19	420	0
YAL038W	571.6137	0.239676	2	0.002441	3	3	50	1
YAL041W	27161.11	0.308308	29	0.066916	3	30	575	1
YAL049C	16.94193	0.260218	2	0.008821	3	3	66	0
YAL053W	6892.674	0.258356	7	0.004791	4	8	92	0

2. Peptide Length

The length of a protein may have a profound effect on is functionality. In 2002, Lipman et al. demonstrated a clear correlation between protein sequence conservation and it length. He showed that the conserved proteins generally have larger lengths compared to their non-conserved counterparts. It is proven that functionally important proteins are more evolutionarily conserved than protein less vital for life (Hirsh AE, et al.,2001) and this holds true for essential proteins (Koonin et al,2002).Thus it was felt that the length of the open reading frame (ORF) may serve as an important classifier for distinction between essential and non- essential genes.

Method

- The ORF length information was obtained from Ensembl (Biomart).
- In the ensembl website (http://asia.ensembl.org/index.html), first Biomart database was selected (http://asia.ensembl.org/biomart/martview/a19e12ed91a28dfb858daca7353f3c91).
- In the Biomart database, Ensembl genes 86 and under that *Saccharomyces cerevisiae* genes (R64-1-1) was selected.

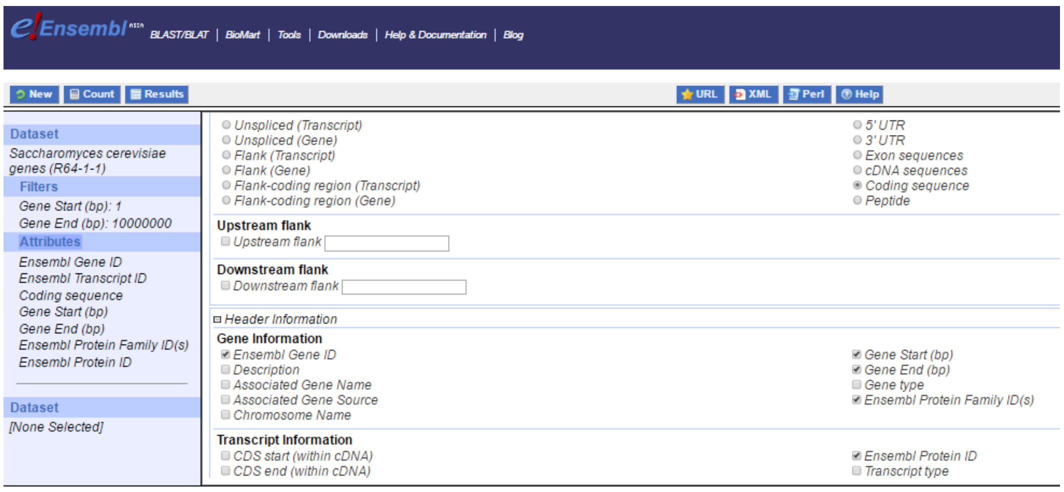

asia.ensembl.org/biomart/martview/a19e12ed91a28dfb858daca7353f3c91

Dataset
Saccharomyces cerevisiae genes (R64-1-1)

Ensembl Genes 86
Saccharomyces cerevisiae genes (R64-1-1)

Ensembl ASIA
BLAST/BLAT | BioMart | Tools | Downloads | Help & Documentation | Blog
New
Count
Results
URL
XML
Perl
Help

- Under the subfield attributes, “sequences” were selected. Also the attributes were further filtered to “coding sequence “ only.
- The subfield “Header Information” was selected where the subfields “Gene Start (bp)”,”Gene End (bp)” were seleted. Also the subfields, “Ensembl gene id” and “Ensembl protein id ” were selected as identifiers.


Ensembl
BLAST/BLAT | BioMart | Tools | Downloads | Help & Documentation | Blog
New
Count
Results
URL
XML
Perl
Help

Dataset
Saccharomyces cerevisiae genes (R64-1-1)
Filters
Gene Start (bp): 1
Gene End (bp): 10000000
Attributes
Ensembl Gene ID
Ensembl Transcript ID
Coding sequence
Gene Start (bp)
Gene End (bp)
Ensembl Protein Family ID(s)
Ensembl Protein ID

Dataset
[None Selected]

○ Unspliced (Transcript)
○ Unspliced (Gene)
○ Flank (Transcript)
○ Flank (Gene)
○ Flank-coding region (Transcript)
○ Flank-coding region (Gene)
○ 5' UTR
○ 3' UTR
○ Exon sequences
○ cDNA sequences
● Coding sequence
○ Peptide

Upstream flank
☐ Upstream flank

Downstream flank
☐ Downstream flank

⊟ Header Information

Gene Information
✓ Ensembl Gene ID
☐ Description
☐ Associated Gene Name
☐ Associated Gene Source
☐ Chromosome Name
✓ Gene Start (bp)
✓ Gene End (bp)
☐ Gene type
✓ Ensembl Protein Family ID(s)

Transcript Information
☐ CDS start (within cDNA)
☐ CDS end (within cDNA)
✓ Ensembl Protein ID
☐ Transcript type

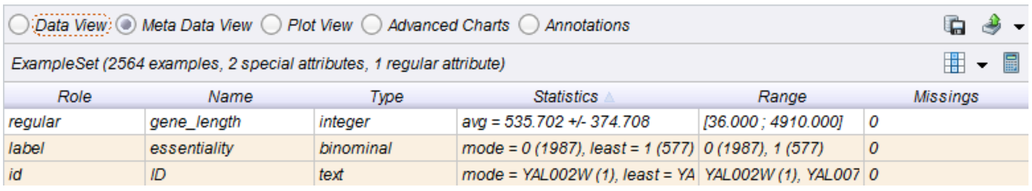
- The resulting dataset was downloaded as a csv file. It was further consolidated with essential and non-essential gene information and examples having missing values were eliminated.
- The following table is provided as an illustrative view of the first 20 cases of the entire dataset.

ID	gene_length	essentiality
YAL002W	1274	0
YAL007C	215	0
YAL010C	493	0
YAL012W	394	0
YAL017W	1356	0
YAL019W	1131	0
YAL021C	837	0
YAL022C	517	0
YAL023C	759	0
YAL024C	1435	0

YAL026C	1355	0
YAL029C	1471	0
YAL031C	760	0
YAL032C	379	1
YAL033W	173	1
YAL035W	1002	0
YAL038W	500	1
YAL041W	854	1
YAL049C	246	0
YAL051W	1047	0

- The consolidated data were fed into Rapidminer and neural network was applied as a classifier with default parametric values.

The following is the metadata view of the dataset



ExampleSet (2564 examples, 2 special attributes, 1 regular attribute)

Role	Name	Type	Statistics	Range	Missings
regular	gene_length	integer	avg = 535.702 +/- 374.708	[36.000 ; 4910.000]	0
label	essentiality	binominal	mode = 0 (1987), least = 1 (577)	0 (1987), 1 (577)	0
id	ID	text	mode = YAL002W (1), least = YA	YAL002W (1), YAL007	0

3. Disorderness of proteins

Protein structure is conserved to serve its function, but it is also seen that certain proteins lack a definite folded structure under certain physiological conditions like pH 7 and 25^{0}C (Uversky *et al*, Tompa *et al*). It does not contradict with the conservation of the overall structure of the proteins, because only a certain part of the protein is found to be in disordered state. The disorder serves to play important roles in interactions with other proteins or DNA. These proteins are also found to be 'adaptive' (Gunasekaran *et al*). RNA and protein chaperones, transcriptional regulators, signal transduction proteins, ion channels, and motor proteins are commonly found to have disordered regions (Dunker *et al*, Wright *et al*, Dyson *et al*, Demchenko *et al*). Thus disordered proteins could play crucial roles to keep the cell alive and in controlling gene expression. Studies on disordered proteins have revealed that certain amino acids occur in disordered regions, which have been termed disorder promoting (Campen *et al*, Wang *et al*) and various predictors have been designed on this theory (Campen *et al*, Ferron *et al*).Estimations of disordered fractions of many proteins from the proteomes of many species has been done with validations of the level of predictability (Dunker *et al*[2000],Romero *et al*, Campen *et al*, Uversky *et al*).

In few interesting studies, the functional distributions of predicted disordered proteins were performed. At least 20 different proteins were studied using 710 Swiss Protein Functional keywords. It was found that out of those 302 of the indicated functions were carried out by structured proteins while disordered proteins or disordered regions carried out around 238 of the functions (Xie *et al*). These functions by the latter group included signalling, regulation and control which is crucial for the life of the organism and also for its fertility.

The essential genes are crucial for survival of the organism and the exhibit multi-functionality. As the disorderness promotes multi-functionality, it was hypothesised that extent of disorderness of proteins could be utilised as a basis of segregation between essential and non – essential proteins.

Materials and methods

Sequences of the genes of *S. cerevisiae* were downloaded from Ensembl (www.asia.ensmbl.org) using R Programming environment. BiomaRt package of R (Smedley *et al*.,Durnick *et al*.) was used to extract the data from the Ensembl server (Biomart).

The disorderness of the proteins can be characterised by two parameters, viz. percentage of disorder and length of disorder. The length of disorder (dis_lengh) was calculated from Fold index, a web based tool based on Uversky et al. which can be accessed at http://bip.weizmann.ac.il/fldbin/findex.

In 2010, Uversky et al. proposed a method to predict if a given protein would assume a defined fold or is intrinsically unfolded. This algorithm was modified by Prilusky J et al. (2004) to create the tool fold index. There, they transformed the equation of the boundary line separating ‘folded’ from ‘disordered’ proteins into a simple index, called Foldindex that differentiates between folded and intrinsically unfolded proteins.

Uversky et al. (2000) proposed that the mean net charge, |<R>|,as the absolute value of the difference between the numbers of positively and negatively charged residues at neutral pH , divided by the total residue number, and the mean hydrophobicity, <H>, as the sum of all residue hydrophobicities, divided by the total number of residues, using the Kyte/Doolittle scale (Kyte and Doolittle, 1982), which was here rescaled to a range of 0–1.They rearranged their fold boundary equation $|<R>| = 2.785<H> - 1.151$ (Uversky et al., 2000) to yield the Fold Index as:

$$I_F^{KD} = 2.785<H> - |<R>| - 1.151$$

Here, All positive values here represent proteins (or domains) that are likely to be folded, and negative values represent those ones likely to be intrinsically unfolded.

The software has a simple interface where the protein sequence can be uploaded and the output includes the values of

Number of disordered residues: This numeric value was taken as length of disorderness

Percentage of disorderness was calculated using the formula:

(Number of dirodered residues/Total number of residues (sequence length)*100

This value was regarded as a parameter in the machine learning framework.

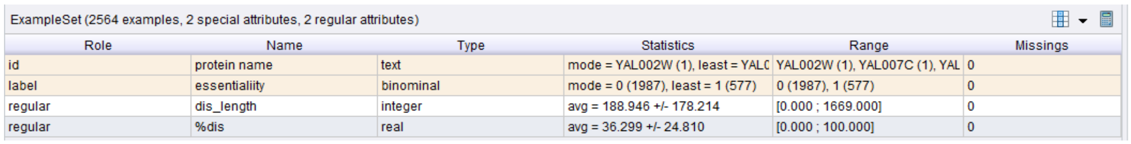
The data were consolidated with essential gene information and a dataset were obtained where essential and non-essential *S cerevisiae* proteins with respective length of disorder and percentage of disorder was created in a csv formatted file.

The first 20 cases are furnished here as illustrative purpose.

id	dis_length	%disorder	Essentiality
YAL002W	89	6.9858713	0
YAL007C	38	17.674419	0
YAL010C	136	27.586207	0
YAL012W	8	2.0304569	0
YAL017W	566	41.740413	0
YAL019W	657	58.090186	0
YAL021C	364	43.48865	0

YAL022C	112	21.663443	0
YAL023C	216	28.458498	0
YAL024C	678	47.247387	0
YAL026C	415	30.627306	0
YAL029C	405	27.532291	0
YAL031C	301	39.605263	0
YAL032C	310	81.794195	1
YAL033W	35	20.231214	1
YAL035W	444	44.311377	0
YAL038W	46	9.2	1
YAL041W	320	37.470726	1
YAL049C	0	0	0
YAL051W	232	22.158548	0

The metadata view of the dataset is presented below



ExampleSet (2564 examples, 2 special attributes, 2 regular attributes)

Role	Name	Type	Statistics	Range	Missings
id	protein name	text	mode = YAL002W (1), least = YALC	YAL002W (1), YAL007C (1), YAL	0
label	essentialiity	binominal	mode = 0 (1987), least = 1 (577)	0 (1987), 1 (577)	0
regular	dis_length	integer	avg = 188.946 +/- 178.214	[0.000 ; 1669.000]	0
regular	%dis	real	avg = 36.299 +/- 24.810	[0.000 ; 100.000]	0

. 4. Strand bias

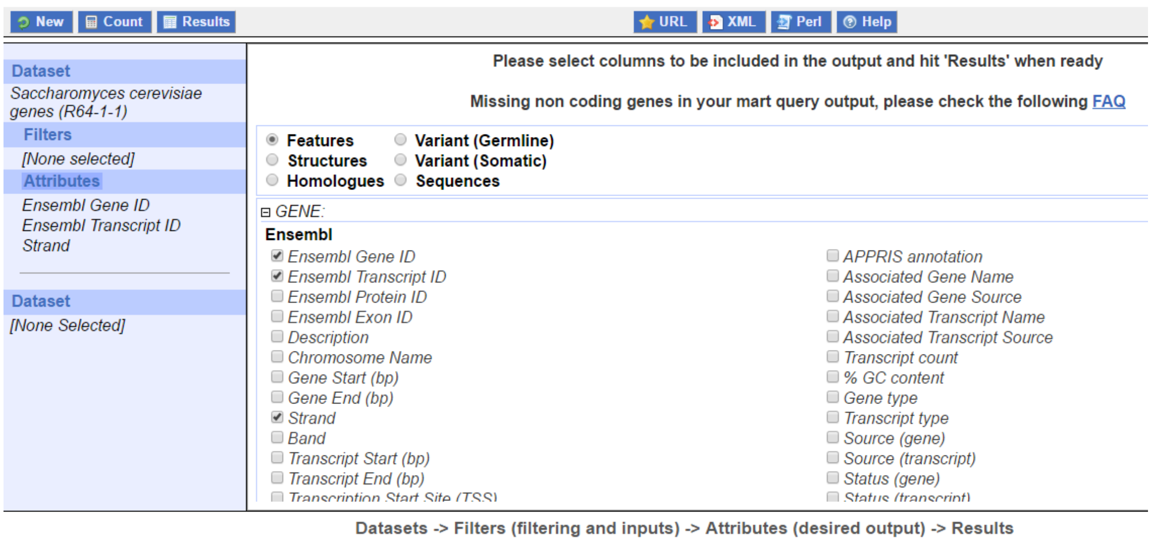
It is seen that replication and transcription simultaneously occur in a DNA molecule. DNA Polymerase in *E.coli* proceeds 10–20 times faster than RNA polymerase (Hiirose S et al.,1983). Thus both head-on and co-oriented collisions will have high chance of occurrence in replicating bacteria. But, the outcome of the collision will depend on whether the polymerases are co-oriented or not. This results in occassional transcription abortion and more severe replication slow-down when genes are in the lagging strand. Thus it is felt, that during the course of evolution, nature has preferentially positioned the translation-related highly expressed genes (rDNA and ribosomal proteins) in the leading strand to prevent lower transcription abortion and maintain higher replication rates.

Lin et al. (2010) found that strand bias was a major factor in determining essentiality in bacterial genomes as essential genes were majorly located in leading strands.It is felt that the same theory might be applicable in case of eukaryotic genomes like S. cerevisiae. Thus position of the gene in the leading or lagging strand was considered as a parameter.

Method

The strand position data of the genes under consideration was obtained from ensemble (Biomart) server

- In the ensembl website (http://asia.ensembl.org/index.html), first Biomart database was selected (http://asia.ensembl.org/biomart/martview/a19e12ed91a28dfb858daca7353f3c91).
- In the Biomart database, Esemble genes 86 and under that *Saccaromyces cerevisiae* genes (R64-1-1) was selected.
- Under the attributes menu “strand” was selected as a feature apart from the other ientifiers of the gene
- The following figure shows the selection process in Biomart server



Dataset
Saccharomyces cerevisiae genes (R64-1-1)
Filters
[None selected]
Attributes
Ensembl Gene ID
Ensembl Transcript ID
Strand

Dataset
[None Selected]

Please select columns to be included in the output and hit 'Results' when ready

Missing non coding genes in your mart query output, please check the following FAQ

[● Features] ○ Variant (Germline)
○ Structures ○ Variant (Somatic)
○ Homologues ○ Sequences

⊟ *GENE:*

Ensembl

✓ *Ensembl Gene ID*	☐ *APPRIS annotation*
✓ *Ensembl Transcript ID*	☐ *Associated Gene Name*
☐ *Ensembl Protein ID*	☐ *Associated Gene Source*
☐ *Ensembl Exon ID*	☐ *Associated Transcript Name*
☐ *Description*	☐ *Associated Transcript Source*
☐ *Chromosome Name*	☐ *Transcript count*
☐ *Gene Start (bp)*	☐ *% GC content*
☐ *Gene End (bp)*	☐ *Gene type*
✓ *Strand*	☐ *Transcript type*
☐ *Band*	☐ *Source (gene)*
☐ *Transcript Start (bp)*	☐ *Source (transcript)*
☐ *Transcript End (bp)*	☐ *Status (gene)*
☐ *Transcription Start Site (TSS)*	☐ *Status (transcript)*

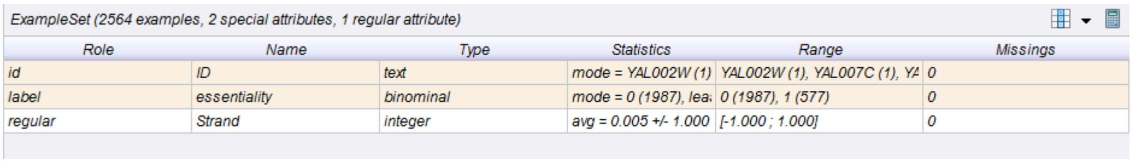
Datasets -> Filters (filtering and inputs) -> Attributes (desired output) -> Results

- The data were consolidated with essential gene related information and were formatted in a csv file.
- The following table shows the first 20 rows of data for illustrative purpose.It is to be noted that for the purpose of coding in machine learning framework, the position of the gene under question I leading strand was coded as “1” and in the lagging strand as “-1”.

ID	Strand	essentiality
YAL002W	1	0
YAL007C	-1	0
YAL010C	-1	0
YAL012W	1	0
YAL017W	1	0
YAL019W	1	0
YAL021C	-1	0

YAL022C	-1	0
YAL023C	-1	0
YAL024C	-1	0
YAL026C	-1	0
YAL029C	-1	0
YAL031C	-1	0
YAL032C	-1	1
YAL033W	1	1
YAL035W	1	0
YAL038W	1	1
YAL041W	1	1
YAL049C	-1	0
YAL051W	1	0

The meta data view of the dataset is as follows:



ExampleSet (2564 examples, 2 special attributes, 1 regular attribute)

Role	Name	Type	Statistics	Range	Missings
id	ID	text	mode = YAL002W (1)	YAL002W (1), YAL007C (1), YA	0
label	essentiality	binominal	mode = 0 (1987), lea	0 (1987), 1 (577)	0
regular	Strand	integer	avg = 0.005 +/- 1.000	[-1.000 ; 1.000]	0

5. Protein abundance

Proteins play a crucial role in our lives as they are required for almost all cellular processes. The deluge of sequence information found in biological databases gives an idea about the Different mRNA coded by the genome but it does not reflect the post-translational modifications and also the expression level during the normal phase of cellular life. The rapid development of high throughput methods in proteome studies has made possible to enlist the complete plethora of proteins found in cells in different phases of the life cycle (Schena et al., Eisen et al.). The techniques include DNA Microarrays tagging the ORFs with high-affinity epitopes and expression from its natural chromosomal location (Ghaemmaghami et al.) , use of high throughput flow cytometry and GFP tagged yeast strains (Newman et al.) to calculate the protein abundance level in a single cell , etc. Saccharomyces cerevisiae remains one of the most well studied organisms with a detailed profiling of protein concentrations in the nearly proteome wide level.

Ghaemmaghami et al. reported that nearly 80% of the proteome is expressed during the normal growth conditions. Thus the abundance of proteins crucial for life may show a significant variance with the proteins expressed only during special conditions.

It has been reported that PPI networks constructed using affinity purification methods for yeast and *Eschericia coli* demonstrate a correlation between protein degree, or number of interactions, and cellular abundance (Ivanic et al.). Ning et al also reported that there is a strong correlation between hub proteins and essential proteins. Essential proteins being crucial for cellular functioning occur in larger complexes and core proteins are involved in more number of biological processes than attachment proteins(Chakroborty et al.)

The protein expressions are regulated by on various factors in a cell and their spatial distribution also varies. But it is often observed that the different proteins maintain an innate and specific range of abundance levels in a cell (Wang et al). In a growing yeast cell, the absolute abundance levels may range from 32 to 500000 copies per cell, with the rarest proteins being low abundant (Ghaemmaghami, S. et al. *(2003)).*Ivanic et al., did an analysis of correlation of protein abundance with high throughput protein-protein interaction studies and came to a conclusion that essential proteins also show high abundance compared to their non-essential counterparts.

Thus, we felt it interesting to assess the extent of relationship of protein abundance to its essentiality for the organism. A direct correlation based analysis of abundance with essentiality is not possible as Greenbaum et al., opined that the abundance data are often quite complex and noisy to ascertain their expression features. Machine learning approaches have effectively demonstrated that it can significantly classify and segregate between noisy data (Zhu). So in this work an attempt has been made to predict essentiality of proteins (and their corresponding genes) using abundance data as classifier in a machine learning framework and test if this could be used as a sole parameter to predict essentiality of a protein. The various methods mentioned above including topological properties and codon usage bias include many parameters. Calculation of topological features of proteins in biological network is also a cumbersome task. Thus the prediction of essentiality just using a single parameter, i.e. protein abundance may be an easy and simple process to implement.

MATERIAL AND METHODS

Gene sequence and related information of *S. cerevisiae* were downloaded from Ensembl (www.ensmbl.org) using R Programming environment. BiomaRt package of R (Smedley *et al.*,Durnick *et al.*) was used to extract the data from the Ensembl server (Biomart).

The information about essential genes was downloaded from the Database of Essential Genes (DEG version 5, Zhang *et al*).This information was used to segregate the yeast proteins among essential and non-essential types. There is no database of non-essential genes. Thus from the entire genepool of *S. cerevisiae*, the genes not included under essential gene database were regarded as non-essential. The protein abundance data were obtained from PaxDb: Protein Abundance Across Organisms (http://pax-db.org/#!home, Wang et al.).The paxDb database contains many datasets related to S. cerevisiae. The dataset used here is S.cerevisiae - Whole organism, SC (Serikawa,MCP,2003) and was retrieved from the location :

http://pax-db.org/data/abundances/4932-
Saccharomyces_cerevisiae_SC_biomart_18097_O__1reps.txt

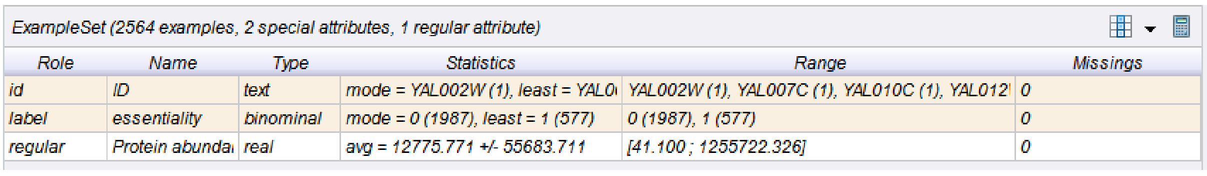
The abundance data is presented in a numerical framework, i.e. expressing average steady-state protein abundances in molecular counts, normalized to parts per million (ppm).

Both the datasets obtained from Biomart (Ensembl) and PaxDb were consolidated and uncommon entries were removed. The final dataset, which was used is presented below with first 20 rows as illustrative purpose. All the data were further consolidated with essential gene data to create the final dataset for machine learning based classification.

ID	Protein abundance	essentiality
YAL002W	736.4646	0
YAL007C	26271.85	0
YAL010C	768.1028	0
YAL012W	38300.99	0
YAL017W	966.5969	0
YAL019W	6796.713	0
YAL021C	2779.05	0

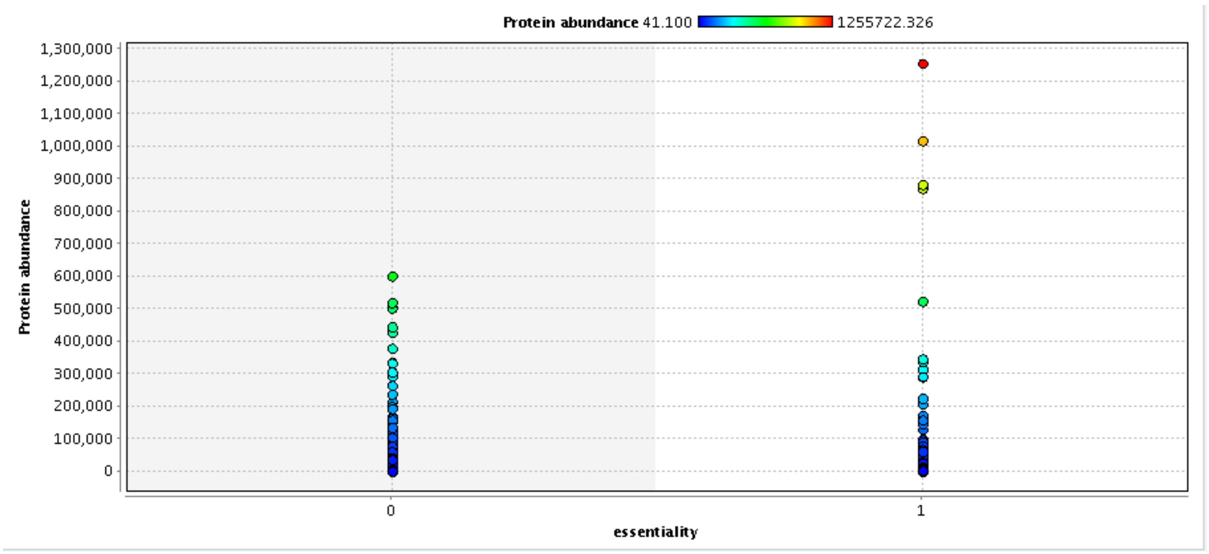
YAL022C	216.6345	0
YAL023C	6512.437	0
YAL024C	304.4945	0
YAL026C	606.1983	0
YAL029C	2205.346	0
YAL031C	227.3864	0
YAL032C	1674.509	1
YAL033W	2229.45	1
YAL035W	13406.22	0
YAL038W	290834.3	1
YAL041W	1010.901	1
YAL049C	1797.084	0
YAL051W	92.29656	0

The metadata view of the dataset is presented below



ExampleSet (2564 examples, 2 special attributes, 1 regular attribute)

Role	Name	Type	Statistics	Range	Missings
id	ID	text	mode = YAL002W (1), least = YAL0	YAL002W (1), YAL007C (1), YAL010C (1), YAL012	0
label	essentiality	binominal	mode = 0 (1987), least = 1 (577)	0 (1987), 1 (577)	0
regular	Protein abunda	real	avg = 12775.771 +/- 55683.711	[41.100 ; 1255722.326]	0



6. Expression level

In the post genomic era, a major thrust have been given towards linking genotypes to phenotypes (Abecasis et al.,2010).Implementation of large-scale loss-of-function studies have elucidated genome-wide phenotype information for several eukaryotic models, including *Drosophila melanogaster* (Boutros et al., 2004), *Saccharomyces cerevisiae* (Winzeler et al., 1999), *Caenorhabditis elegans* (Kamath et al., 2003). Essential genes play crucial function in the cell and thus they may have high expression levels. This theory is also corroborated by the facts that the highly connected proteins , which are generally essential in nature are having high expression level. He highly connected proteins are also part of many protein complexes and thus they are supposed to have a higher expression level.

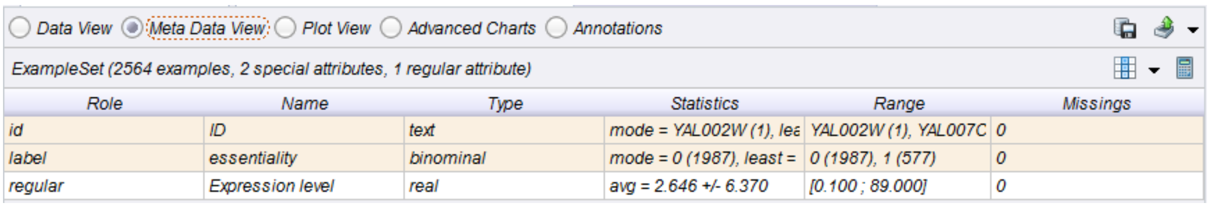
Earlier, the expression level of a gene used to be calculated by measuring the transcribed mRNA (northern blot), the expressed protein (Western Blot), or by directly staining the protein or mRNA when it is still in the cell. Recent emergence of high throughput techniques has dynamically changed way of gene expression gene expression studies. The recent methods include DNA microarrays, serial analysis of gene expression (SAGE), and high-throughput sequencing allow larger screens of multiple molecules simultaneously.

Method

The protein expression data were collected from Holstege et al. (http://web.wi.mit.edu/young/pub/data/orf_transcriptome.txt) the data were formatted and consolidated with essential gene expression data. The dataset were formatted into a csv file for analysis with rapidminer. The first 20 rows of data are presented here for illustrative purpose

ID	Expression level	essentiality
YAL002W	0.2	0
YAL007C	2.2	0
YAL010C	0.3	0
YAL012W	8.9	0
YAL017W	0.4	0
YAL019W	0.9	0
YAL021C	1.3	0
YAL022C	0.9	0
YAL023C	2.7	0
YAL024C	0.2	0
YAL026C	0.7	0
YAL029C	1	0
YAL031C	0.4	0
YAL032C	0.1	1
YAL033W	1.2	1
YAL035W	8.8	0
YAL038W	50.4	1
YAL041W	0.4	1
YAL049C	2.4	0
YAL051W	0.6	0

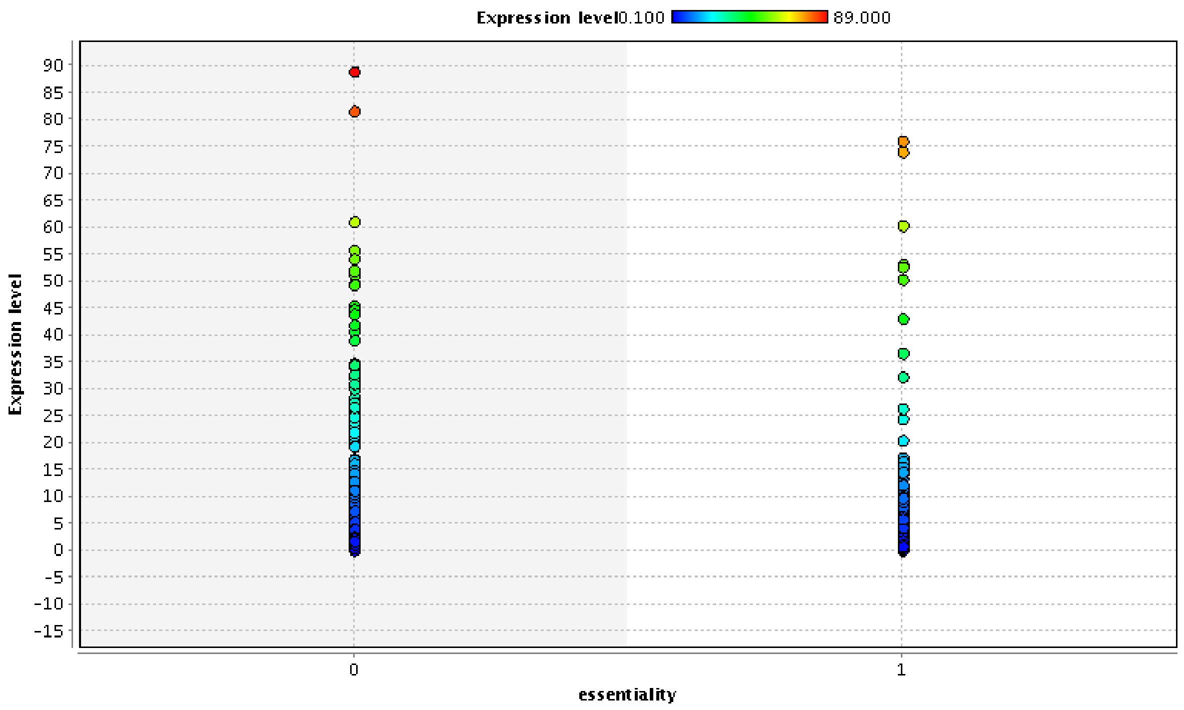
The following figure shows meta data view:



ExampleSet (2564 examples, 2 special attributes, 1 regular attribute)

Role	Name	Type	Statistics	Range	Missings
id	ID	text	mode = YAL002W (1), lea	YAL002W (1), YAL007C	0
label	essentiality	binominal	mode = 0 (1987), least =	0 (1987), 1 (577)	0
regular	Expression level	real	avg = 2.646 +/- 6.370	[0.100 ; 89.000]	0

The following figure shows the distribution of the data



7. Codon Usage Bias

For translation, triplet codons of amino acid sequences are required . As the nucleic acids are made up of four types of nucleotides, in the genetic code, there 43 = 64 codons (tri-nucleotide sequences) are possible. For translation into a peptide , each of these codons requires a tRNA molecule with a complementary anticodon sequence. If each of the tRNA molecule had paired with its complementary mRNA codon using canonical Watson-Crick base pairing, then 64 types (species) of tRNA molecule would be required. Among these three stop codons are encountered (UAA, UAG and UGA). But the organisms have much lesser number of tRNA species in the cell. In 1966, Francis Crick observed this phenomenon and proposed the Wobble Hypothesis to account for this. He opined that the 5' base on the anticodon, which binds to the 3' base on the mRNA, was relaxed for the third position of the codon, unlike the first two positions where a canonical Watson-Crick base pairing is followed . This has led to the Wobble hypothesis.

It could be expected that all triplets coding for the same amino acid should be equally frequent in the organisms, but it has been established for a long time that this is not the case, both among organisms and among genes from a single species (Grantham et al. 1981). While few amino acids are encoded by a single codon, most amino acids are encoded by two to six different codons. Different codons that may encode the same amino acid are termed as synonymous codons. However, although synonymous codons encode the same amino acids, it has been shown for a wide variety of organisms that different synonymous codons are used with different frequencies. This phenomenon has been termed codon bias. Several factors govern the unequal usage of bases at third codon positions within synonymous codons which include directional mutational bias ((Levine and Whitemore, 2000),translational selection (Ikemura, 1985, Gupta and Ghosh, 2001), secondary protein structure (Oresisc and Shalloway, 1998; Gupta et al., 2002), replicational and transcriptional selection ((McInerney, 1998; Romero et al., 2000), and environmental factors (Lynn et al., 2002; Basak et al., 2004).

The strength of codon bias varies across genes within each genome. Few genes are found to be using a highly biased set of codons and others may use the different synonymous codons with similar frequencies(Gouy M et al. ,1982). The codon bias is found to have a strong correlation with the level of gene expression (Duret et al.,1999; Ikemura et al.,1985). Ghaemmaghami S et al. (2003) had shown that this rule follows well in case of *S. cerevisiae*.Sharp and Li (1987) had proven that the codon bias is most distinct in highly expressed genes and it exhibits a strong preference for a subset of synonymous codons, which are often called as optimized or preferred codons (Paul P et al.,2014). As essential genes are often highly expressed and they show high phyletic retention, they are expected to show significant difference in codon usage bias compared to non-essential genes. The work on *S. aureus* bacteriophage genes by Paul Pet al. has shown that codon usage bias may be a tool to distinguish between essential and non-essential phage genes.{ bibilo].

Materials and methods

Sequence retrieval

Coding ORF Sequences of Saccharomyces cerevisiae were downloaded from Biomart of Ensembl using R Programing environment. The package BiomaRt (Durinck, S et al.,2010) was used for the purpose.

The R codes of the sequence retrieval process is given below:

```
library(biomaRt)

listMarts()

ensembl=useMart("ensembl")

listDatasets(ensembl)

mart <- useMart(biomart = "ensembl", dataset = "scerevisiae_gene_ensembl")

View(mart)

summary(mart)

ensembl=useDataset("scerevisiae_gene_ensembl",mart=ensembl)

listAttributes(ensembl)

genes.with.id=getBM(attributes=c("ensembl_gene_id", "coding"),values="*", mart=
ensembl)

seq= getSequence(type="entrezgene",seqType =
"coding",mart=ensembl,id=genes.with.id)

yeast_sequences=data.frame(seq)

write.csv(yeast_sequences,file="yeast_sequences.csv",col.names = T,row.names = T)
```

The resulting file was consolidated with essential gene data file to match the gene ids with description of essentiality .

Calculation of codon usage bias

GCUA: General Codon Usage Analysis software was used for performing various calculations of codon usage bias of the sequences under study. GCUA, developed by McInerney J.O (1998) is a widely used software. The program for Mac version was downloaded from http://mcinerneylab.com/software/gcua/# .

Various methods and measures have been suggested for calculation of codon usage bias. The following measures as calculated by GCUA software were incorporated in the current study:

Nc:

It refers to "Effective Number of codons". This measure is useful for analysis of how biased a gene is in terms of its codon usage.For the gene that tends to use all codons with equal frequency will have Nc value of 61, whereas Nc value of 20 will be found for a gene that is effectively using only a single codon for each amino acid. A relationship has been observed between Nc and the base composition of a gene. Genes that having a higher bias in base compositions being expected to have lower Nc values. This might be pointing to the fact that there is some kind of selective pressure (like translational selection) on the gene to use a smaller subset of codons or optimal codons. Optimal codons are the ones that correspond to the major abundance tRNA for that amino acid. This pattern may be well correlated with phyletic retention of the genes.

GC3:

The GC composition at the third site of codons. T is found to be affected by mutational bias. Thus essential genes may show a similar GC3 composition, which may be highly varying between non-essential genes.

Multivariate Analyses:

Each gene is represented by a set of co-ordinates. These are the codon usage statistics for each codon.For the universal genetic code, the gene is represented by 59 co-ordinates (each of the 59 codons for which there is a synonymous alternative), but this figure varies, depending on the genetic code that is being used.If an amino acid coded by three codons, a gene is encoded only by three codons, the position of the gene can be plotted in a high-dimensional space by reference to the RSCU values of each of these three codons. The position of the gene on this space, is given by its codon count values.   If all of the genes for a particular organism are taken and are plotted on this high-dimensional space according to their codon usage statistics, A cloud like appearance will be there. In the absence of any codon usage bias, this 'cloud' would look spherical. The points should be dotted around the axes in a random way, with no general direction. However, if there is some factor influencing codon selection on some genes, then the 'cloud' will no longer be spherical, but will have some kind of skewness. The values of the plot in four axes are represented by Ax1, Ax2, Ax3 and Ax4.

Consolidation of the data:

The calculated codon usage related values were appended with list of genes containing information being essential or not and a suitable file was created in csv format for analysis in Rapidminer.

8. Rare Codons:

Codon usage bias has been found to be a major factor in identifying the recently acquired genes and differentiate those with the phyletically retained genes.Since the 1970s, the uneven use of codon usage of synonymous codons have been reported. Researchers have reported various influence factors for the phenomenon of codon usage bias. These include abundance of isoacceptor tRNA[Kanya S et. al], amino acid composition[D'Onofrio G et. al], mRNA secondary structure [Zama M et. al], the efficiency of translation initiation[Stenström CM et. al] and GC content [Knight RD et. al], gene length[Marín A] and few other properties.The scientists are yet to reach a consensus on the formation mechanism of codon usage bias, but it has been well utilised in estimation and comparison of the expression level of endogenous genes (Yu X et. Al) and detection of change in the efficiency of expression of exogenous genes (Lee MH et. Al).The codon usage bias throws ample illumination to find the course of evolution of genes (Thakur S et. Al).

From early investigations in Escherichia coli[Kanaya S et. al], it was found that usage of preferred codons in genes was positively correlated with their respective major isoacceptor tRNA levels, and this was explained as an adaptation of highly expressed genes to translational efficiency. Since the same rules of codon bias pattern have been found to be consistent in various taxa like in S. cerevisiae [Ikemura T et. al], Drosophila[Moriyama EN] and C. elegans[Duret L et. al]. The initial findings of these and similar studies had the consensus that highly expressed gene must show high codon usage bias but with the development of high throughput technology for gene sequencing and detection of expression level, doubts over this theory have increased over the time [Dos Reis M et. Al, Lavner Y et.al, Singh ND et. Al).

Over the last three decades,various indices have been developed to describe and measure the degree of codon bias, and the same have been applied to codon bias analysis. It has been established that in prokaryotes, many indices exhibit a positive correlation with the gene expression level, such as CAI (Codon Adaptation Index) [Sharp PM et. al], CBI (Codon Bias Index)[Bennetzen JL et. al], and Fop (Frequency of optimal Codons)[Kanaya S et. al].

Wu et. al developed a new measure , called usage of rare codons to measure and explain the usage of codon bias in an objective manner. They found that the pattern of usage of rare codons were markedly different between essential and non-essential genes of *E. coli.* It is felt that the same pattern may be used as a classifier to detect essential genes of *S. cerevisiae*. They calculated the rare codons in case of *E. coli* and in this Ph.D work, the objective was to use the pattern of rare codons for detection of essentiality in yeast. The rare codons are not phyletically retained and often are found in acquired genes through HGT or other means . The rare codons of *S. cerevisiae* was calculated in the same manner as Wu. et. al.

1. Identification of rare codons of based on codon pairs preference

   We calculated the occurrence frequency of each kind of different six-nucleotide strings in 2564 sequences of *S. cerevisiae* in two ways:

   By using the criterion derived from statistical analysis, the “rare codon pairs” and “normal codon pairs” were defined.

Thirteen rare codons (GGA,CTC,TAG,CTA,ACA, GAC, AGG, AGA, CCC, GGG, GAG, ACT, and ATA) were identified by the statistical test method for hyper- geometric distribution, which was used to evaluate the contribution of the sixty-four codons to the rare codon pairs

2. F_{rare} (the frequency of rare codons) was used as an index of codon bias

 The F_{rare} value of genes was calculated based on the rare codons identified by the method mentioned above.

Methodology

The programs were written in PERL language (ActiveState Perl, v5.8.4) ad run in DOS.

Codon usage patterns were calculated using GCUA - General Codon Usage Analysis (McInerney et. al).[http://mcinerneylab.com/software/gcua/]

BioEdit version 7 was used to load the gene sequences and translated into their respective protein sequences as per their general codon usage table. [http://www.mbio.ncsu.edu/bioedit/bioedit.html]

The output files generated by GCUA and Perl programs were loaded into Excel (Microsoft) for further analysis.

Identification of rare codons based on codon pairs preference

Codon pairs searching and statistical test

A total of 4096 (64 × 64 = 4096) different six-nucleotide strings made up of 2 codons are there. An array containing these strings was made for search in gene sequences of *S. cerevisiae*.

Perl programs were made to search the 4096 strings in 2564 sequences of *S. cerevisiae* and then, the “rare strings” and “normal strings” were defined by statistical analysis.

Searching the strings as per the open reading frame:

According to synonymous codon's encoding rule , one string made up of two amino acids would correspond to several six-nucleotide strings. So at first, the frequencies of all the six-nucleotide strings and the corresponding two amino acid strings were calculated by open reading frame within all gene sequences and protein sequences respectively in *S. cerevisiae* . Then the statistical test method for binomial cumulative distribution was implemented to get a P1 value of each codon pair, by which the probability of the real frequency can be displayed if it is assumed that codon usage was random (P1 values of 4050 strings were obtained after excluding the strings which are not adapted to analysis because they only contain "ATG" or "TGG" or their corresponding two amino acids strings don't exist in protein sequences).

$$P_1 = \sum_{m=0}^{k} C_n^m \cdot p^m \cdot q^{n-m}$$

p: the probability of a codon pair's occurrence corresponding to any given two amino acids string based on encoding rule.

$$p = \frac{1}{syn(I) \cdot syn(J)}; \quad q = 1\text{-}p$$

k: the frequency of a codon pair according to open reading frame; m: 0 ≦ m ≧ k;

n: the frequency of the corresponding 2 amino acid string encoded by the codon pair in protein sequence; syn(i): the degeneracy of the amino acid coded by i.

3. Searching the strings in spite of open reading frame The actual occurrence frequency of all the six-nucleotide strings in gene sequences was obtained by general search in spite of open reading frame, and also P2 was calculated by the same method for binomial cumulative distribution above.

$$P_2 = \sum_{m=0}^{k} C_n^m \cdot p^m \cdot q^{n-m}$$

p: the probability of a six-nucleotide string's occurrence because of possible composition of four types of nucleotides.

$p = \frac{1}{4^6};\ q = 1\text{-}p$

k: the frequency of six-nucleotide strings by general searching in 2564 sequences;

m: $0 \leqq m \geqq k$;

n: $n = \sum_i^m (Ni\text{-}5)$

where Ni is the number of nucleotides in gene i

3) The criterion for dividing all codon pairs into "rare" and "normal" groups

Through the analysis above, we got two P values (P1, P2) for each codon pair, then a cutoff value P_0 ($P_0 = 0.01/2564/4289 = 9.09\times 10^{-10}$) was made to be the criterion. For a codon pair, If P1 and P2 are both less than P_0 ($P_1 < P_0$ and $P_2 < P_0$), it will be defined as "rare codon pair". Otherwise, it will be put into the "normal" group.

Calculation of "F_{rare}" (frequency of rare codon) index

The definition of F_{rare} of gene g is :

$$F_{rare}(g) = \frac{1}{N}\sum_i syn(i) n_i(g)$$

$n_i(g)$: the count of the codon i in the gene g; N: the total number of codons in gene g; syn(i): the degeneracy of the amino acid encoded by i

The data were consolidated in to a file fec_data.csv which consisted of three columns.

The first column contains the gene id

The second column contained Fec value and

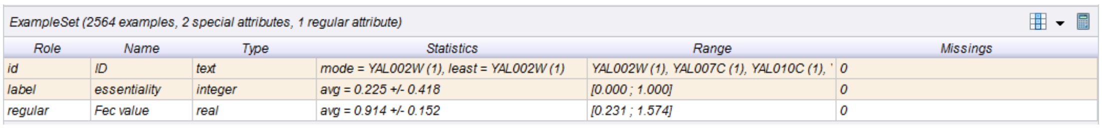
The third column contained the information about essentiality. Here , the value 0 corresponds to non-essential genes and the value 1 corresponds to essential genes

A truncated view of the csv file containing only 20 rows of data is presented here.

ID	Fec value	essentiality
YAL002W	0.981961	0
YAL007C	0.944444	0
YAL010C	1.145749	0
YAL012W	0.41519	0
YAL017W	1.048637	0
YAL019W	0.936396	0
YAL021C	0.655131	0
YAL022C	0.621622	0
YAL023C	0.6	0
YAL024C	1.04039	0
YAL026C	1.056047	0
YAL029C	0.961957	0
YAL031C	0.86728	0
YAL032C	0.831579	1
YAL033W	1.074713	1
YAL035W	0.907278	0
YAL038W	0.491018	1
YAL041W	0.881871	1

YAL049C	1.048583	0
YAL051W	0.971374	0

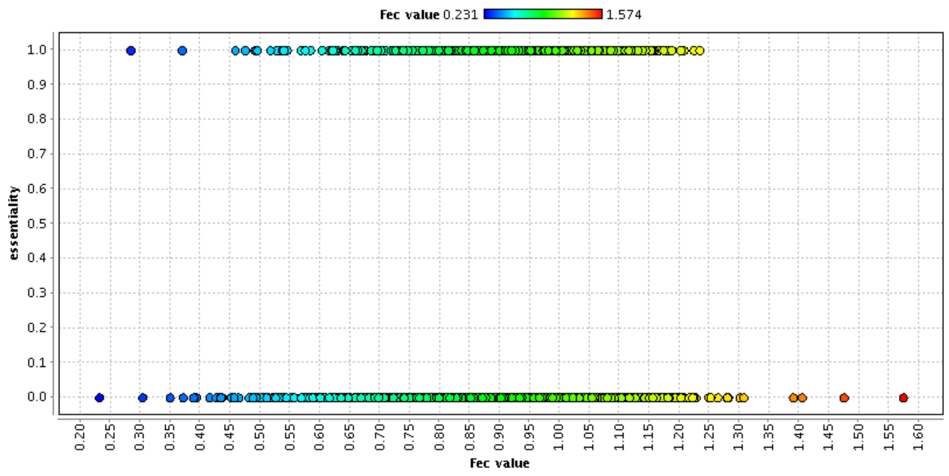
The metadata view of the dataset is as follows:



ExampleSet (2564 examples, 2 special attributes, 1 regular attribute)

Role	Name	Type	Statistics	Range	Missings
id	ID	text	mode = YAL002W (1), least = YAL002W (1)	YAL002W (1), YAL007C (1), YAL010C (1), '	0
label	essentiality	integer	avg = 0.225 +/- 0.418	[0.000 ; 1.000]	0
regular	Fec value	real	avg = 0.914 +/- 0.152	[0.231 ; 1.574]	0

The distribution of fec value with respect to essential and non-essential genes was found to be as follows:



9. Complex number of proteins

The functioning of the proteins is not a solo event, rather they often form macromolecular complexes to execute their functions. When more than two proteins associate themselves with each other, they are known to be in a complex or such formation is known as multiprotein complex. Proteins in the complex are linked by non-covalent protein–protein interactions. The different protein complexes may have different degrees of stability .Thus they could be transient or permanent, the later having a longer half life. The elucidation of internal subunit arrangement of proteins in complexes play an important role in understanding of their functions. Few interesting findings have come up regarding the proteins that are found in complexes. Proteins having more number of interactions evolve at a slower rate. They do possess important functionalities, and thus nature conserves them at a higher degree. The evolutionary rate of the protein is also correlated with their local centrality (like degree) .Thus it may have a direct impact of the protein being essential as the essential proteins have high expressiveness, and slower evolution rate. The differentially functional proteins are also found to be generally essential and such proteins are generally found to be participating in many complexes(Chakraborty et al.). The determination of complex number, thus may be important to find essentiality of the proteins as essential proteins are expected to have higher complex numbers than non-essential proteins.

Wang et al. (2009) noted that larger protein complexes are more likely to be essential, explaining why essential genes are more likely to have high cocomplex interaction degree. Ryan et al. (2013) referred to the observation that entire complexes appear essential as "modular essentiality"

Method

The protein complex related data were collected from Gavin et al. When the data were formatted in spreadsheet, search was made for a particular protein and number of complexes of which it was a member. These integer values were treated as complex numbers.

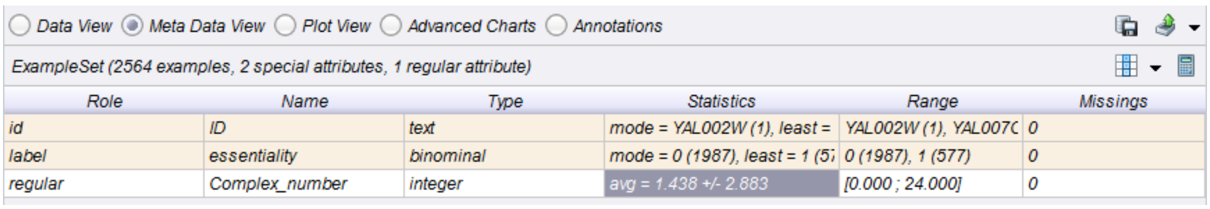
The data were consolidated with the information of the each of the cadidates being essential or not and formatted in a csv file.

The following table is an illustration of the first 20 rows of the dataset

ID	Complex_number	essentiality
YAL002W	1	0
YAL007C	1	0
YAL010C	0	0
YAL012W	1	0
YAL017W	5	0
YAL019W	0	0
YAL021C	1	0
YAL022C	0	0
YAL023C	0	0
YAL024C	2	0
YAL026C	2	0
YAL029C	2	0
YAL031C	0	0
YAL032C	4	1
YAL033W	1	1
YAL035W	9	0
YAL038W	7	1
YAL041W	1	1
YAL049C	0	0

YAL051W	0	0

The metadata view of the complex number dataset is as follows:

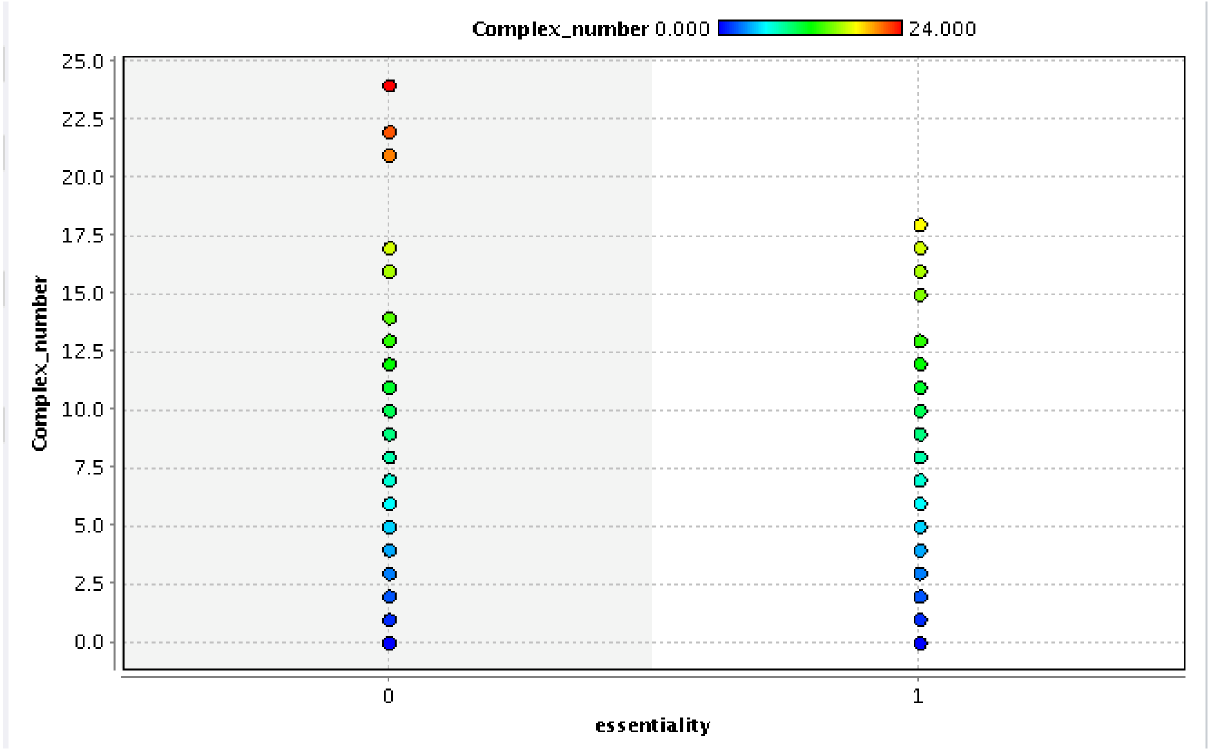


Data View Meta Data View Plot View Advanced Charts Annotations

ExampleSet (2564 examples, 2 special attributes, 1 regular attribute)

Role	Name	Type	Statistics	Range	Missings
id	ID	text	mode = YAL002W (1), least =	YAL002W (1), YAL007C	0
label	essentiality	binominal	mode = 0 (1987), least = 1 (57	0 (1987), 1 (577)	0
regular	Complex_number	integer	avg = 1.438 +/- 2.883	[0.000 ; 24.000]	0

The distribution of complex number data with respect to essentiality is given in the figure



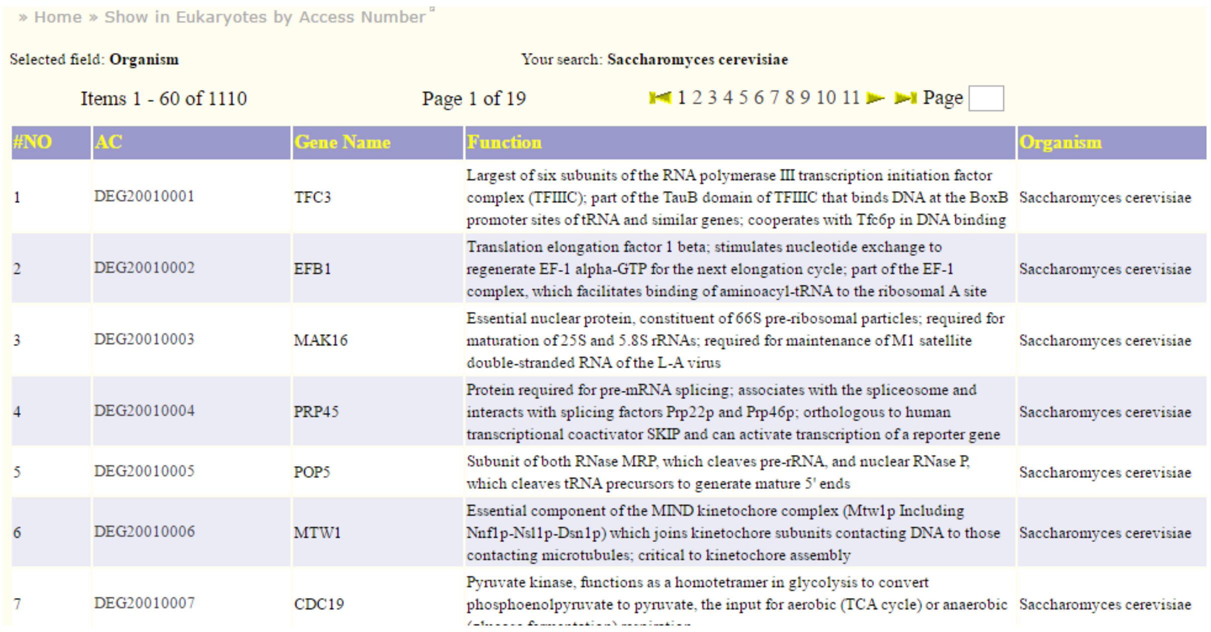
Essential Gene Data

The essential gene data was downloaded from DEG (Database of Essential Genes) (Zhang R et al, 2004,2009) version 5 which can be accessed at http://tubic.tju.edu.cn/deg/ .

The database hosts information about essential genes of Bacteria, Archea and Eukaryotes. Recently, they have also added a sub-database on on-essential genes (Luo et al.,2014), which was not available when this work had taken place.The database has 1110 genes of Saccharomyces cerevisiae which is sourced from Giaever G, et al (2002).

The data were collected and formatted into Microsoft Excel. The data consisted of DEG accession number, Gene name , functional annotation of the gene and name of the organism. From this the name of the gene was matched with the GO ID , Ensembl ID and GenBank ID for matching the related data from other databases.

The following figure shows the typical arrangement of DEG essential gene data of *S. cerevisiae*



» Home » Show in Eukaryotes by Access Number

Selected field: **Organism** Your search: **Saccharomyces cerevisiae**

Items 1 - 60 of 1110 Page 1 of 19 1 2 3 4 5 6 7 8 9 10 11 Page

#NO	AC	Gene Name	Function	Organism
1	DEG20010001	TFC3	Largest of six subunits of the RNA polymerase III transcription initiation factor complex (TFIIIC); part of the TauB domain of TFIIIC that binds DNA at the BoxB promoter sites of tRNA and similar genes; cooperates with Tfc6p in DNA binding	Saccharomyces cerevisiae
2	DEG20010002	EFB1	Translation elongation factor 1 beta; stimulates nucleotide exchange to regenerate EF-1 alpha-GTP for the next elongation cycle; part of the EF-1 complex, which facilitates binding of aminoacyl-tRNA to the ribosomal A site	Saccharomyces cerevisiae
3	DEG20010003	MAK16	Essential nuclear protein, constituent of 66S pre-ribosomal particles; required for maturation of 25S and 5.8S rRNAs; required for maintenance of M1 satellite double-stranded RNA of the L-A virus	Saccharomyces cerevisiae
4	DEG20010004	PRP45	Protein required for pre-mRNA splicing; associates with the spliceosome and interacts with splicing factors Prp22p and Prp46p; orthologous to human transcriptional coactivator SKIP and can activate transcription of a reporter gene	Saccharomyces cerevisiae
5	DEG20010005	POP5	Subunit of both RNase MRP, which cleaves pre-rRNA, and nuclear RNase P, which cleaves tRNA precursors to generate mature 5' ends	Saccharomyces cerevisiae
6	DEG20010006	MTW1	Essential component of the MIND kinetochore complex (Mtw1p Including Nnf1p-Nsl1p-Dsn1p) which joins kinetochore subunits contacting DNA to those contacting microtubules; critical to kinetochore assembly	Saccharomyces cerevisiae
7	DEG20010007	CDC19	Pyruvate kinase, functions as a homotetramer in glycolysis to convert phosphoenolpyruvate to pyruvate, the input for aerobic (TCA cycle) or anaerobic	Saccharomyces cerevisiae

Saccharomyces Genome Database (SGD)

Sequence files of all *Saccharomyces cerevisae* genes available were downloaded from the Saccharomyces Genome Database (SGD) (Cherry J.M. et al.,2012). SGD provides comprehensive integrated encyclopaedic biological information for *Saccharomyces* *cerevisiae. They also host various* search and analysis tools to explore these data. Other related data like gene expression data from high throughput experiments, Biological pathway related data, phenotypic data, GFP tags also are available here. The resource can be accessed at http://www.yeastgenome.org/.

Preparation of data for machine learning framework

The data downloaded from SGD contained data for all the S. cerevisiae genes which included information about essential and non-essential genes. At the time of the work , The SGD had information about 6803 genes of *S . cerevisiae*.

The rest of the genes, other than the 1110 genes marked as essential by DEG, were considered as non-essential. But it should be noted that since we cannot consider the dataset by DEG as all-inclusive set of essential genes, the genes considered as non-essential may still contain some essential genes. Attempt was made to see, inspite of this possible discrepancy if I was possible to predict essentiality of genes by an *in-silico*, machine learning approach.

Pre-processing of Data

As the data of the various features were appended, it was seen that all the genes were not having related information of all the features. Thus the spreadsheet were having missing values in various cases. Machine learning classifiers are often highly perturbed with the missing values in the dataset. Although there are measures to replace missing value by an approximation (http://docs.rapidminer.com/studio/operators/cleansing/missing/impute_missing_values.html), it was decided to omit the datasets with missing values to reduce any approximation.

A final dataset was obtained by elimination and it contained 2564 genes out of which 1987 were on-essential genes and 577 essential genes were there. This dataset was fed to the machine learning framework.

The entire dataset along with subsets of data of individual parameters have been enclosed here in electronic format

Machine Learning through Rapidminer

The dataset resulting from incorporation of all the parameters was formatted into a csv file. The machine learning classifier often is perturbed by the missing values. So examples were eliminated where any value of the feature was missing.

For machine learning framework, Rapidminer Studio, version 5 (community edition) was used. Rapidminer is a widely used machine learning and predictive modelling software and can be accessed at https://rapidminer.com/products/studio/ .

Rapidminer offers a Graphical user based interface where a workflow can be designed for the entire operations and the same can also be done by xml programming. Individual units of tools in Rapidminer are called operators.

Importing file into Rapidminer

The first step of the machine learning exercise is importing the file by the Rapidminer environment. This can be done by ”Read CSV“ operator.In the rapidminer window, The file is uploaded and then data import wizard helps to import the data and set the roles of the individual attributes.

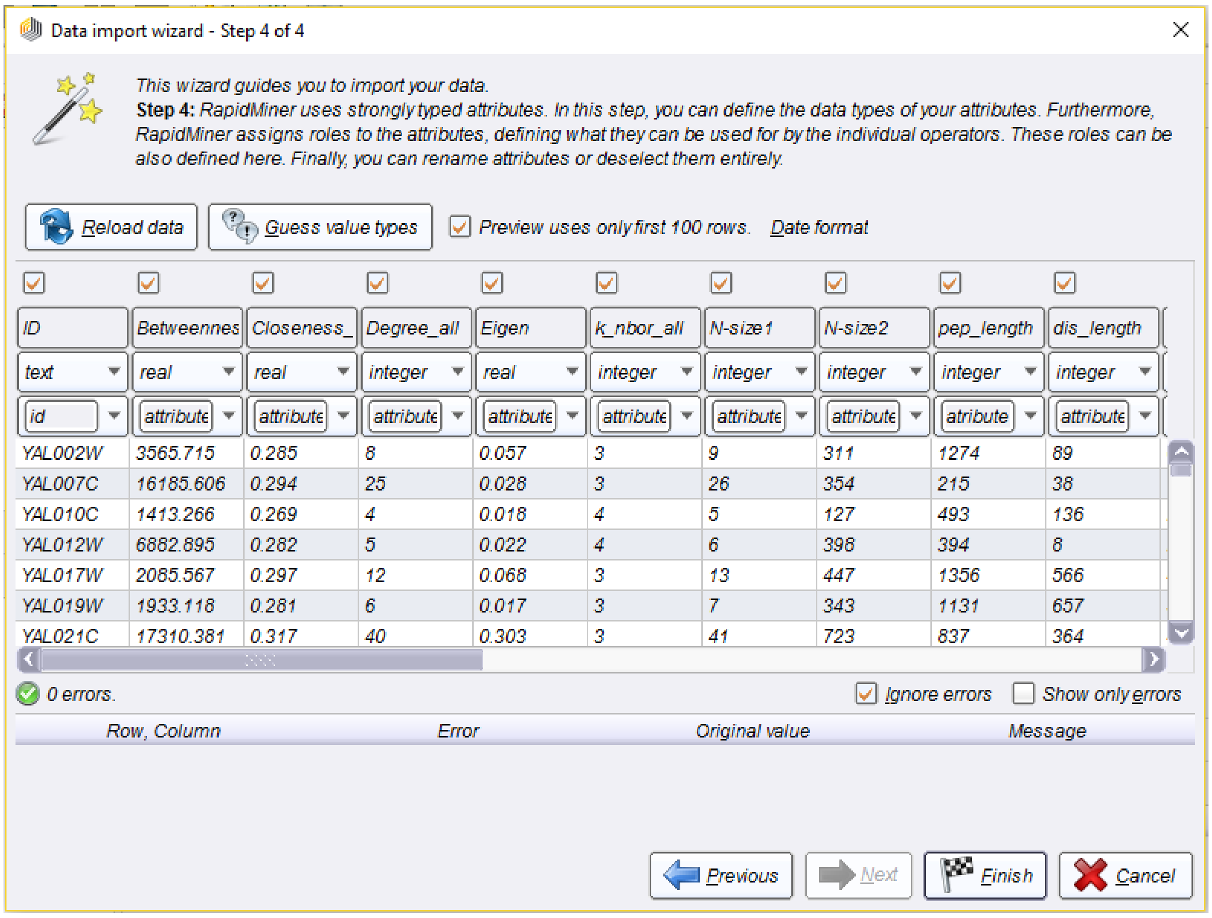
Setting up of Roles in Rapidminer

ID: In Rapidminer, the role of the attribute which is used to uniquely identify the example is known as ID.The protein ID in the first column of the dataset is used as “ID” and the type is set as “text”.

Label: This is a special attribute. It acts as a target attribute for learning operators. Labels identify the examples in any way and they must be predicted for new examples that are not yet characterized in such a manner. The label is also called 'goal variable'. In this case, the last column of the dataset is having the attribute “essentiality”, which contains information whether the given gene/ protein is essential or not. It has either a value of 1 or 0. The value 1 denotes that it is essential and value 0 denotes that it is non-essential. Thus the role of this attribute is set as “label” and the type is set as “binomial”.

Regular attributes: The values of other attributes in the example set contain the information about different features. These values act as input variables for the learning task. For the attributes where the values are only integers, the value type is set as “integer” and for the other cases the value type is “real”.

The figure below shows the setting of the role of the ID and few other regular attributes


Data import wizard - Step 4 of 4

This wizard guides you to import your data.
Step 4: *RapidMiner uses strongly typed attributes. In this step, you can define the data types of your attributes. Furthermore, RapidMiner assigns roles to the attributes, defining what they can be used for by the individual operators. These roles can be also defined here. Finally, you can rename attributes or deselect them entirely.*

Reload data | Guess value types | ✓ Preview uses only first 100 rows. | Date format

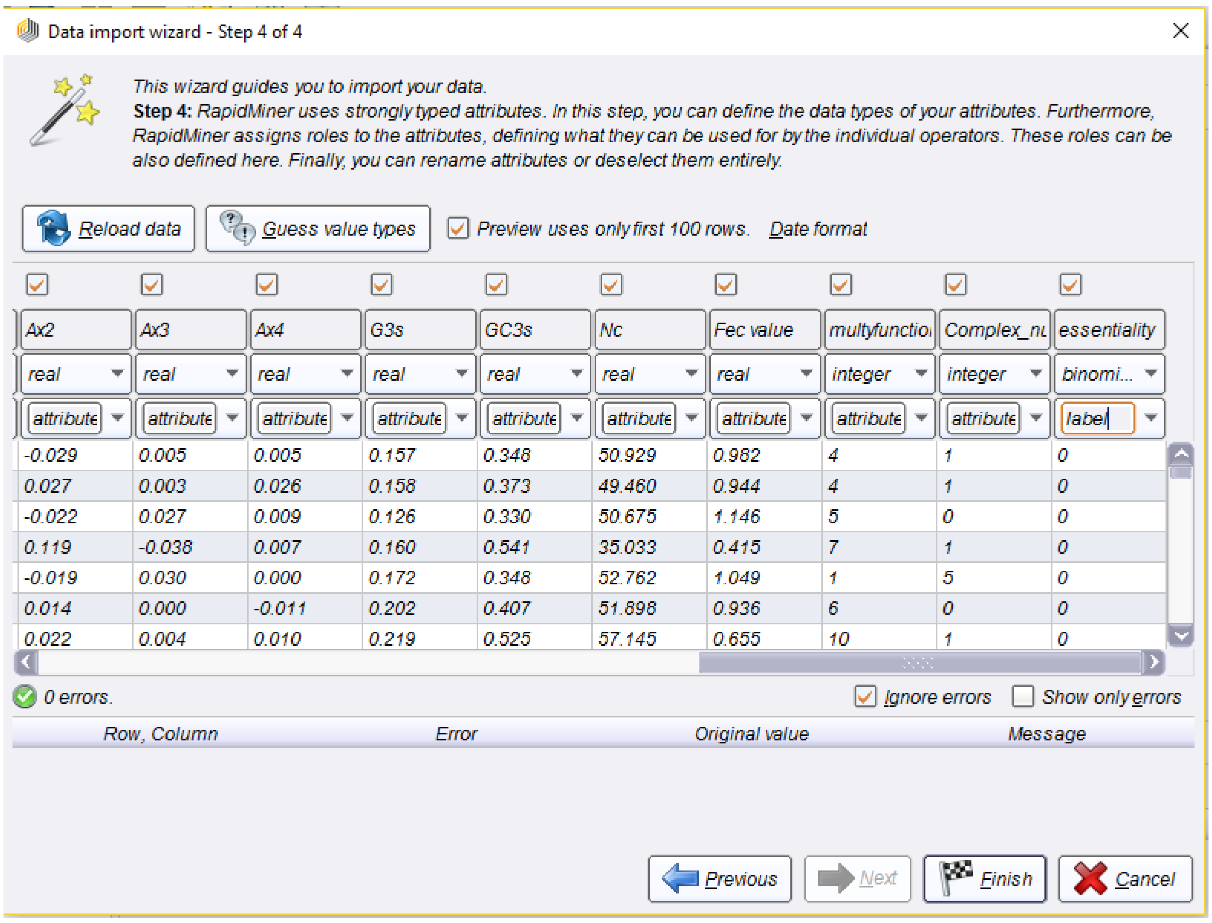
✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ID	Betweennes	Closeness_	Degree_all	Eigen	k_nbor_all	N-size1	N-size2	pep_length	dis_length
text	real	real	integer	real	integer	integer	integer	integer	integer
id	attribute	attribute	attribute	attribute	attribute	attribute	attribute	atribute	attribute
YAL002W	3565.715	0.285	8	0.057	3	9	311	1274	89
YAL007C	16185.606	0.294	25	0.028	3	26	354	215	38
YAL010C	1413.266	0.269	4	0.018	4	5	127	493	136
YAL012W	6882.895	0.282	5	0.022	4	6	398	394	8
YAL017W	2085.567	0.297	12	0.068	3	13	447	1356	566
YAL019W	1933.118	0.281	6	0.017	3	7	343	1131	657
YAL021C	17310.381	0.317	40	0.303	3	41	723	837	364

0 errors. ✓ Ignore errors ☐ Show only errors

Row, Column	Error	Original value	Message

Previous | Next | Finish | Cancel

The figure below shows the setting of roles as label and few other regular attributes



This wizard guides you to import your data.
Step 4: *RapidMiner uses strongly typed attributes. In this step, you can define the data types of your attributes. Furthermore, RapidMiner assigns roles to the attributes, defining what they can be used for by the individual operators. These roles can be also defined here. Finally, you can rename attributes or deselect them entirely.*

☑ Preview uses only first 100 rows.

☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
Ax2	Ax3	Ax4	G3s	GC3s	Nc	Fec value	multyfunctio	Complex_nu	essentiality
real	real	real	real	real	real	real	integer	integer	binomi...
attribute	attribute	attribute	attribute	attribute	attribute	attribute	attribute	attribute	label
-0.029	0.005	0.005	0.157	0.348	50.929	0.982	4	1	0
0.027	0.003	0.026	0.158	0.373	49.460	0.944	4	1	0
-0.022	0.027	0.009	0.126	0.330	50.675	1.146	5	0	0
0.119	-0.038	0.007	0.160	0.541	35.033	0.415	7	1	0
-0.019	0.030	0.000	0.172	0.348	52.762	1.049	1	5	0
0.014	0.000	-0.011	0.202	0.407	51.898	0.936	6	0	0
0.022	0.004	0.010	0.219	0.525	57.145	0.655	10	1	0

0 errors. ☑ Ignore errors ☐ Show only errors

Row, Column	Error	Original value	Message

The output of the read csv is connected to the result port of Rapidminer and the processing is started. The result is displayed and metadata view presents a consolidated view of the dataset.

The xml code of the entire operation is given below:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<process version="5.3.015">

 <context>

  <input/>

  <output/>
```

```
    <macros/>
  </context>
  <operator activated="true" class="process" compatibility="5.3.015" expanded="true" name="Process">
    <process expanded="true">
      <operator activated="true" class="read_csv" compatibility="5.3.015" expanded="true" height="60" name="Read CSV" width="90" x="112" y="75">
        <parameter key="csv_file" value="F:\thesis\data\scere_truncated.csv"/>
        <list key="annotations"/>
        <list key="data_set_meta_data_information"/>
      </operator>
      <connect from_op="Read CSV" from_port="output" to_port="result 1"/>
      <portSpacing port="source_input 1" spacing="0"/>
      <portSpacing port="sink_result 1" spacing="0"/>
      <portSpacing port="sink_result 2" spacing="0"/>
    </process>
  </operator>
</process>
```

Classification:

Neural network had been used as a classifier algorithm in this work.The Neural net used here had three major parameters:

Training cycle: 500

Learning rate : 0.3

Momentum: 0.2

Error epsilon value : 10^{-5}

The data were normalised and the system used shuffled sampling.This parameter builds random subsets of the ExampleSet. Examples are chosen randomly for making subsets.

The neural network algorithm employed here used a feed-forward neural networktrained by a back propagation algorithm.

Cross validation:

Cross validation is used to estimate the statistical performance of a learning operator.It is mainly used to estimate how accurately a model (learnt by a particular learning operator) will perform in practice.

In Rapidminer, cross validation (X-validation) is a nested operator. It has two subprocesses: a training subprocess and a testing subprocess. The training subprocess is used for training a model. The trained model is then applied in the testing subprocess. The performance of the model is also measured during the testing phase.

The input ExampleSet is partitioned into *k* subsets of equal size. Of the *k* subsets, a single subset is retained as the testing data set (i.e. input of the testing subprocess), and the remaining *k − 1* subsets are used as training data set (i.e. input of the training subprocess). The cross-validation process is then repeated *k* times, with each of the *k* subsets used exactly once as the testing data. The *k* results from the *k* iterations then can be averaged (or otherwise combined) to produce a single estimation. The value *k* can be adjusted using the *number of validations* parameter.

In this case the number of validations parameter was set to 10. This parameter specifies the number of subsets the ExampleSet should be divided into (each subset has equal number of examples). Also the same number of iterations will take place. Each iteration involves training a model and testing that model.

The input port accepted labelled example set

The output gives three types of data:

Model: The training subprocess returns a model, which is trained on the input Example Set. There is an option to save the model for further use.

Training data: The ExampleSet that was given as input at the training input port is passed without changing to the output through this port. This is usually used to reuse the same ExampleSet in further operators or to view the ExampleSet in the Results Workspace.

Averagable (Performance Vector) : The testing subprocess must return a Performance Vector. This is usually generated by applying the model and measuring its performance.

A parameter “Average performances only” had been used here which means that only performance vectors should be averaged or all types of averagable result vectors .

Inside the nested operator, the sampled segmented data is fed to the neural network and the model is retained for performance testing. The output is given to “apply model” operator. Here due to 10 fold cross validation, the model is trained on 90% of the data and tested on rest of the 10% data. The performance is noted by the “performance operator”. Here the data are cycled ten times (due to 10 fold cross validation) and the performances of ten iterations are averaged.

Here binomial performance operator was used. This operator is used for statistical performance evaluation of binominal classification tasks i.e. classification tasks where the *label* attribute has a binominal type. This operator delivers a list of performance criteria values of the binominal classification task.

The following criteria were considered for testing of the performance:

Accuracy

Relative number of correctly classified examples or in other words percentage of correct predictions. It has a Boolean range.

Classification error

Relative number of misclassified examples or in other words percentage of incorrect predictions.

Precision

Relative number of correctly as positive classified examples among all examples classified as positive i.e. precision=(Positives Correctly Classified)/(Total Predicted Positives). Note that the Total Predicted Positives is the sum of True Positives and False Positives. This is the same as the positive predictive value.

Recall

This parameter specifies the relative number of correctly as positive classified examples among all positive examples i.e. recall=(Positives Correctly Classified)/(Total Positives). It is also called hit rate or true positive rate. This is the same as sensitivity.

False positive

This parameter specifies the absolute number of negative examples that were incorrectly classified as positive examples. In other words, if the example is negative and it is classified as positive, it is counted as a false positive. Range: boolean

False negative

This parameter specifies the absolute number of positive examples that were incorrectly classified as negative examples. In other words, if the example is positive and it is classified as negative, it is counted as a false negative. Range: boolean

True positive

This parameter specifies the absolute number of positive examples that were correctly classified as positive examples. In other words, if the example is positive and it is classified as positive, it is counted as a true positive. Range: boolean

True negative

This parameter specifies the absolute number of negative examples that were correctly classified as negative examples. In other words, if the example is negative and it is classified as negative, it is counted as a true negative. Range: Boolean

Specificity

The relative number of correctly as negative classified examples among all negative examples i.e. specificity=(Negatives Correctly Classified)/(Total Negatives). Note that Total Negatives is equal to sum of True Negatives and False Positives.

Evaluation of performance of individual parameters and their relevance to model building exercise

Weight by Correlation

This operator calculates the relevance of the attributes by computing the value of correlation for each attribute of the input ExampleSet with respect to the label attribute. This weighting scheme is based upon correlation and it returns the absolute or squared value of correlation as attribute weight.

The Weight by Correlation operator calculates the weight of attributes with respect to the label attribute by using correlation. The higher the weight of an attribute, the more relevant it is considered.

The Rapidminer operator Weight by Correlation takes example set as input and provides attribute weights as the output.

Parameters used:

Normalize weights

This parameter indicates if the calculated weights should be normalized or not. Here , it was set to true, and hence all weights are normalized in range from 0 to 1.

The xml code of the process is given below

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<process version="5.3.015">

  <context>

    <input/>

    <output/>

    <macros/>

  </context>

  <operator activated="true" class="process" compatibility="5.3.015" expanded="true"
name="Process">

    <process expanded="true">

      <operator activated="true" class="read_csv" compatibility="5.3.015" expanded="true"
height="60" name="Read CSV" width="90" x="45" y="30">

        <parameter key="csv_file" value="F:\thesis\data\scere_truncated.csv"/>
        <parameter key="column_separators" value=","/>

        <parameter key="first_row_as_names" value="false"/>

        <list key="annotations">

          <parameter key="0" value="Name"/>

        </list>

        <parameter key="encoding" value="windows-1252"/>

        <list key="data_set_meta_data_information">

          <parameter key="0" value="Sq.true.text.id"/>

          <parameter key="1" value="Status.true.integer.label"/>

          <parameter key="2" value="Time in position.true.real.attribute"/>

          <parameter key="3" value="Years of Service.true.real.attribute"/>

          <parameter key="4" value="PerformanceIndex.true.integer.attribute"/>

          <parameter key="5" value="Number of Direct Reports.true.integer.attribute"/>

          <parameter key="6" value="Number of Colleagues.true.integer.attribute"/>

          <parameter key="7" value="Expat(1/0).true.integer.attribute"/>

        </list>

      </operator>

      <operator activated="true" class="weight_by_correlation" compatibility="5.3.015"
expanded="true" height="76" name="Weight by Correlation" width="90" x="246" y="75"/>

      <connect from_op="Read CSV" from_port="output" to_op="Weight by Correlation"
to_port="example set"/>

      <connect from_op="Weight by Correlation" from_port="weights" to_port="result 1"/>

      <connect from_op="Weight by Correlation" from_port="example set" to_port="result
2"/>

      <portSpacing port="source_input 1" spacing="0"/>

      <portSpacing port="sink_result 1" spacing="0"/>

      <portSpacing port="sink_result 2" spacing="0"/>
```

```
      <portSpacing port="sink_result 3" spacing="0"/>
    </process>
  </operator>
</process>.
```

Evaluation of the predictability

Lift Chart

This operator generates a lift chart for the given model and Example- Set based on the discretized confidences and a Pareto chart.

Description

The Create Lift Chart operator creates a lift chart based on a Pareto plot for the discretized confidence values of the given ExampleSet and model. The model is applied on the ExampleSet and a lift chart is produced afterwards.

The lift chart measures the effectiveness of models by calculating the ratio between the result obtained with a model and the result obtained without a model. The result obtained without a model is based on randomly selected records.

The “Create lift chat” operator of Rapidminer has the following ports:

Input ports:

Example set: The ExampleSet that was given as input is passed with- out changing to the output through this port. This is usually used to reuse the same ExampleSet in further operators or to view the ExampleSet in the Results Workspace.

Model: The model that was given as input is passed without changing to the output through this port. This is usually used to reuse the same model in further operators or to view the model in the Results Workspace.

Lift pareto chart: For the given model and ExampleSet a lift chart is gen-erated based on the discretized confidences and a Pareto chart. This lift chart is delivered through this port.

Parameters:

Target class (string): This parameter indicates the target class for which the lift chart should be produced.Here the target class was 1 of the label Essentiality. Thus the chart was used to see the predictability of the essential gene candidates.

Binning type (selection) : This parameter indicates the binning type of the confidences.