**NATIONAL INSTITUTE OF TECHNOLOGY PATNA**



**PROGNOSIS OF PANCREATIC CANCER**

A minor project report submitted in partial fulfilment of the requirements for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**2021-2025**

SUBMITTED BY

|  |  |
| --- | --- |
| DIVYANSH JAIN | 2106219 |
| JHA BHAVESH JAYSHANKAR | 2106212 |
| SHIVAM CHATURVEDI | 2106229 |

SUPERVISED BY

**DR. ANUSHREE TRIPATHI**

Assistant Professor

Department of Computer Science and Engineering

**CERTIFICATE**

**NATIONAL INSTITUTE OF TECHNOLOGY PATNA**

Department of Computer Science and Engineering

This is to certify that the project report entitled “PROGNOSIS OF PANCREATIC CANCER” submitted by Divyansh Jain (2106219), Jha Bhavesh Jayshankar (2106212), Shivam Chaturvedi (2106229) in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering of National Institute of Technology, Patna is a record of bonafide work carried out under my guidance and supervision during the academic year 2023-2024. It is certified that all the corrections/suggestions indicated for Internal Assessment have been incorporated in the project report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed.

Supervisor Head of the Department

**Dr. Anushree Tripathi Dr. M P Singh**

Assistant Professor Professor

Department of CSE Department of CSE

NIT Patna NIT Patna

MAY 2024 MAY 2024

**ACKNOWLEDGEMENTS**

**NATIONAL INSTITUTE OF TECHNOLOGY PATNA**

Department of Computer Science and Engineering



We would like to take this chance to express our gratitude to everyone who helped us during this study period and provided guidance and encouragement. The completion of this dissertation would not have been feasible without their invaluable efforts.

We are highly thankful to our supervisor, Dr. Anushree Tripathi (CSE Dept.), for her guidance, advice and support throughout the entire dissertation work. Without her guidance and encouragement, it would not have been possible to finish this dissertation. As our supervisor, she provided us with the encouragement and freedom to pursue our own ideas. She was always patient, even when we weren’t, and helpful whenever her guidance was needed.

We wish to convey our sincere gratitude to the Head of Department and all the faculties of Computer Science Engineering Department who have enlightened us during our studies. The faculties and cooperation received from the technical staff of Department of Computer Science Engineering are thankfully acknowledged.

Without the assistance and support of our friends and colleagues, this job would not have been feasible. we would also like to thank them for continuous support and encouragement provided by them, which kept us motivated and cheered in the most difficult of times.

Divyansh Jain

Jha Bhavesh Jayshankar

Shivam Chaturvedi

CONTENTS

INTRODUCTION………………………………………………………………………………… 6

OBJECTIVE……………………………………………………………………………………… 7

THEORITACAL BACKGROUND……………………………………………………………… 7

ARTIFICIAL NEURAL NETWORKS………………………………………………… 8

SUPPORT VECTOR MACHINES…………………………………………………… 9

GRADIENT BOOSTING CLASSIFIER……………………………………………… 12

**XG BOOST CLASSIFIER…………………………………………………………….**

RANDOM FOREST ALGORITHM…………………………………………………… 13

DIFFERENTIALLY EXPRESSED GENES…………………………………………. 14

LITERATURE SURVEY……………………………………………………………………… 16

METHODOLOGY OF IMPLEMENTATION………………………………………………… 17

ALGORITHM…………………………………………………………………………………… 22

RESULT ANALYSIS…………………………………………………………………………… 23

CONCLUSION………………………………………………………………………………… 27

FUTURE WORK……………………………………………………………………………… 28

REFERENCES……………………………………………………………………………….. 28

**ABSTRACT**

Pancreatic cancer remains a significant global health concern due to its aggressive nature and high mortality rates. In this study, we employed a comprehensive machine learning approach, including Support Vector Machine (SVM), Gradient Boosting Classifier (GBC) models and Neural Network models, to analyze gene expression data from pancreatic cancer samples. Our aim was to develop an accurate and efficient model for predicting disease-relevant feature genes and improving diagnosis and prognosis prediction. Through rigorous testing and evaluation, our ensemble model demonstrated robust performance in identifying key feature genes associated with pancreatic cancer. Noteworthy genes emerged as crucial biomarkers aiding in diagnosis and prognosis assessment. This study underscores the potential of machine learning models in identifying novel intervention targets and enhancing our understanding of pancreatic cancer biology.

**INTRODUCTION**

In recent years, pancreatic cancer has shown a growing trend worldwide. The resulting death rate is also climbing. From 1990 to 2017, the number of deaths, incident cases, and disability-adjusted life-years caused by pancreatic cancer in the world has more than doubled [, and it could become the second leading cause of death from cancer in the next 20–30 years. In the past few decades, the research and treatment of pancreatic cancer have made slow progress. The diagnosis of early pancreatic cancer is still difficult, with a 5-year survival rate of less than 10%, and most of the life expectancy after diagnosis does not exceed half a year. The poor prognosis of pancreatic cancer is closely related to the difficulty of early diagnosis of the disease. The onset of pancreatic cancer is hidden, and the symptoms are abdominal pain, backache, weight loss, etc., with no specific manifestations. When the symptoms are obvious, it is difficult to perform surgical treatment, only about 20% of patients can be diagnosed and resected in time at an early stage, and the 5-year survival rate of patients undergoing surgical resection is only about 15–25%, the response of pancreatic cancer to most chemotherapy drugs is also poor.

Smoking, obesity, and diabetes are known as major risk factors for pancreatic cancer, and excessive alcohol intake is also one of the reasons for the increased risk of pancreatic cancer. Smoking is still the main cause of pancreatic cancer in the world, and it is currently recognized as a risk factor. Others may be related to pancreatitis, allergy, microbial fora, and genetic susceptibility genes. At present, tri-phasic pancreatic-protocol CT is an important way to diagnose pancreatic cancer. To a great extent, it provides predictive value for resection and reference value for advanced pancreatic cancer and metastasis. However, it has poor sensitivity in detecting early pancreatic cancer. Endoscopic ultrasound is also very helpful for the diagnosis of pancreatic cancer, and diagnostic cytological sampling is performed under an endoscope. In addition, MRI also plays a certain role in the diagnosis of pancreatic cancer.

In this study, we present a comprehensive approach leveraging machine learning techniques to predict the prognosis of pancreatic cancer using transcriptomic data. Our study utilized publicly available gene expression datasets, including GSE15471 and GSE16515 as training datasets, and GSE32676 as the independent test dataset, obtained from the Gene Expression Omnibus (GEO) repository, comprising gene expression profiles of pancreatic cancer samples. We initially performed GEO2R analysis to identify differentially expressed genes (DEGs) within the training datasets, followed by employing Random Forest to extract feature genes from the DEGs based on their importance scores. Enrichment analysis was conducted on the selected feature genes to elucidate their biological significance in pancreatic cancer.

Subsequently, we developed a multi-model prognostic framework incorporating Support Vector Machine (SVM), Artificial Neural Network (ANN), and Gradient Boosting Classifier (GBC). These models were trained on the feature genes identified through the Random Forest algorithm and optimized and validated using rigorous cross-validation techniques to ensure robust performance. Our study presents a novel approach for prognostic prediction of pancreatic cancer by integrating transcriptomic data with machine learning methodologies. By identifying and prioritizing feature genes associated with pancreatic cancer prognosis, our multi-model framework offers a promising tool for personalized treatment strategies and clinical decision support in the management of this deadly disease.

**OBJECTIVE**

To develop and validate a machine learning-based prognostic model for pancreatic cancer using genomic data, aiming to accurately predict outcomes as risk for pancreatic cancer identified or not.

**THEORITICAL BACKGROUND**

The idea of ANNs is based on the belief that working of human brain by making the right connections, can be imitated using silicon and wires as living neurons and dendrites .

The human brain is composed of 86 billion nerve cells called neurons. They are connected to other thousand cells by Axons. Stimuli from external environment or inputs from sensory organs are accepted by dendrites. These inputs create electric impulses, which quickly travel through the neural network. A neuron can then send the message to other neuron to handle the issue or does not send it forward. ANNs are composed of multiple nodes, which imitate biological neurons of human brain. The neurons are connected by links and they interact with each other. The nodes can take input data and perform simple operations on the data. The result of these operations is passed to other neurons. The output at each node is called its activation or node value. Each link is associated with weight. ANNs are capable of learning, which takes place by altering weight values. Neural network: A neural network is a network or circuit of neurons, or in a modern sense, an artificial neural network, composed of artificial neurons or nodes. Thus a neural network is either a biological neural network, made up of real biological neurons, or an artificial neural network, for solving artificial intelligence (AI) problem. The connections of the biological neuron are modeled as weights. A positive weight reflects an excitatory connection, while negative values mean inhibitory connections. All inputs are modified by a weight and summed. This activity is referred as a linear combination. Finally, an activation function controls the amplitude of the output. For example, an acceptable range of output is usually between 0 and 1, or it could be - 1 and 1. These artificial networks may be used for predictive modeling, adaptive control and applications where they can be trained via a dataset. Self-learning resulting from experience can occur within networks, which can derive conclusions from a complex and seemingly unrelated set of information

**ARTIFICIAL NEURAL NETWORKS**

In machine learning, a neural network (also artificial neural network or neural net, abbreviated ANN or NN) is a model inspired by the structure and function of biological neural networks in animal brains.

An ANN consists of connected units or nodes called artificial neurons, which loosely model the neurons in a brain. These are connected by edges, which model the synapses in a brain. Each artificial neuron receives signals from connected neurons, then processes them and sends a signal to other connected neurons. The "signal" is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs, called the activation function. The strength of the signal at each connection is determined by a weight, which adjusts during the learning process.

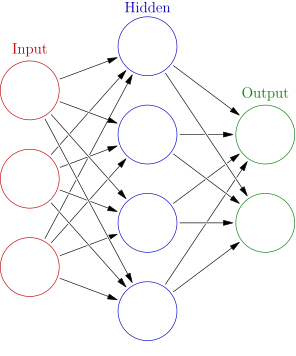
Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly passing through multiple intermediate layers (hidden layers).

Artificial neural networks are used for various tasks, including predictive modeling, adaptive control, and solving problems in artificial intelligence. They can learn from experience, and can derive conclusions from a complex and seemingly unrelated set of information.

Neural networks are typically trained through empirical risk minimization. This method is based on the idea of optimizing the network's parameters to minimize the difference, or empirical risk, between the predicted output and the actual target values in a given dataset. Gradient based methods such as backpropagation are usually used to estimate the parameters of the network. During the training phase, ANNs learn from labeled training data by iteratively updating their parameters to minimize a defined loss function. This method allows the network to generalize to unseen data.

The multilayer perceptron is a universal function approximator, as proven by the universal approximation theorem. However, the proof is not constructive regarding the number of neurons required, the network topology, the weights and the learning parameters.

A specific recurrent architecture with rational-valued weights (as opposed to full precision real number-valued weights) has the power of a universal Turing machine, using a finite number of neurons and standard linear connections. Further, the use of irrational values for weights results in a machine with super-Turing power.



<https://en.m.wikipedia.org/wiki/File:Colored_neural_network.svg>

**SUPPORT VECTOR MACHINES**

In machine learning, support vector machines (SVMs, also support vector networks) are supervised max-margin models with associated learning algorithms that analyze data for classification and regression analysis. Developed at AT&T Bell Laboratories by Vladimir Vapnik with colleagues (Boser et al., 1992, Guyon et al., 1993, Cortes and Vapnik, 1995, Vapnik et al., 1997) SVMs are one of the most studied models, being based on statistical learning frameworks of VC theory proposed by Vapnik (1982, 1995) and Chervonenkis (1974).

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. Being max-margin models, SVMs are resilient to noisy data (for example, mis-classified examples). SVMs can also be used for regression tasks, where the objective becomes - -sensitive.

The support vector clustering algorithm, created by Hava Siegelmann and Vladimir Vapnik, applies the statistics of support vectors, developed in the support vector machines algorithm, to categorize unlabeled data. These data sets require unsupervised learning approaches, which attempt to find natural clustering of the data to groups and, then, to map new data according to these clusters.

The popularity of SVMs is likely due to their amenability to theoretical analysis, their flexibility in being applied to a wide variety of tasks, including structured prediction problems. It is not clear that SVMs have better predictive performance than other linear models, such as logistic regression and linear regression.

**LINEAR SVM**

Linear Support Vector Machine (SVM) is a powerful algorithm used for classification tasks in machine learning. It operates by finding the hyperplane that best separates the data points into different classes. The primary objective of SVM is to maximize the margin between the hyperplane and the nearest data points of each class. This margin is crucial as it determines the generalization ability of the model and its ability to classify unseen data accurately.

At its core, the SVM algorithm works by transforming the input data into a higher-dimensional space using a kernel function. In this higher-dimensional space, a hyperplane is then constructed to separate the data points. The choice of kernel function plays a significant role in the performance of the SVM model. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid.

The Linear SVM specifically focuses on finding a linear hyperplane to separate the data points. It is often preferred when the data is linearly separable, meaning that classes can be separated by a straight line. However, even when the data is not perfectly separable, Linear SVM can still perform well with appropriate regularization techniques.

One of the key advantages of Linear SVM is its efficiency in high-dimensional spaces. Unlike some other algorithms that suffer from the curse of dimensionality, SVM's computational complexity does not depend on the dimensionality of the feature space. This makes it particularly useful for datasets with a large number of features, such as text classification or image recognition tasks.

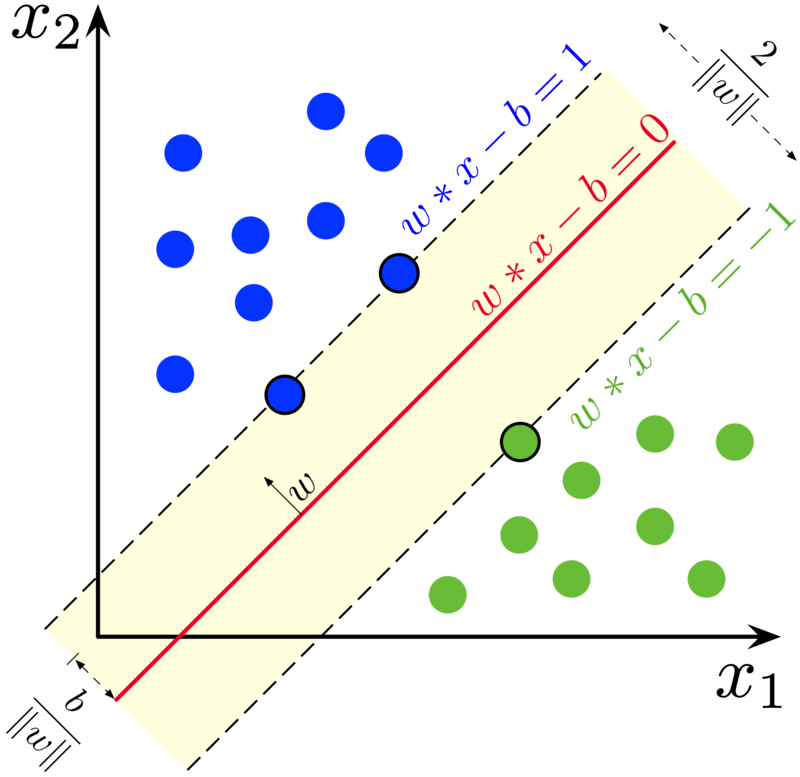
The optimization problem associated with Linear SVM involves finding the hyperplane that maximizes the margin while minimizing the classification error. This is typically formulated as a convex optimization problem, which can be efficiently solved using techniques such as quadratic programming or gradient descent.

Regularization is an essential aspect of SVM, especially in the case of Linear SVM. It helps prevent overfitting by penalizing large coefficients in the hyperplane equation. This is achieved through the use of a regularization parameter, often denoted as C, which controls the trade-off between maximizing the margin and minimizing the classification error. A smaller value of C leads to a wider margin but may result in misclassification of some data points, while a larger value of C may lead to a narrower margin but better classification accuracy.

Another important concept in SVM is the concept of support vectors. These are the data points that lie closest to the hyperplane and are crucial in defining its position. In the case of Linear SVM, support vectors are the data points that lie on the margin or are misclassified. These points have a non-zero Lagrange multiplier associated with them and play a significant role in determining the decision boundary.

In practice, Linear SVM is widely used in various applications, including text classification, image recognition, and bioinformatics. Its simplicity, efficiency, and ability to handle high-dimensional data make it a popular choice among machine learning practitioners. However, it is essential to note that Linear SVM may not perform well on datasets with non-linear decision boundaries. In such cases, more advanced kernel-based SVMs or other non-linear algorithms may be more suitable.

In conclusion, Linear SVM is a powerful algorithm for binary classification tasks that works by finding the hyperplane that best separates the data points into different classes. It is particularly useful in high-dimensional spaces and can handle large datasets efficiently. Regularization and support vector concepts play crucial roles in the optimization process, ensuring good generalization performance. While Linear SVM may not be suitable for datasets with non-linear decision boundaries, it remains a versatile and widely used algorithm in the field of machine learning.



Maximum-margin hyperplane and margins for an SVM trained with samples

from two classes. Samples on the margin are called the support vectors.

<https://en.wikipedia.org/wiki/Support_vector_machine>

**GRADIENT BOOSTING CLASSIFIER**

Gradient Boosting Classifier (GBC) is a sophisticated ensemble learning algorithm used primarily for classification tasks in machine learning. It belongs to the family of boosting algorithms, which work by combining multiple weak learners, typically decision trees, to create a strong learner. GBC iteratively improves the performance of the model by minimizing a loss function, such as the logistic loss function for binary classification or the softmax loss function for multiclass classification, through gradient descent.

At its core, GBC builds an ensemble of decision trees sequentially, where each new tree is trained to correct the errors made by the previous ones. This sequential training process distinguishes GBC from other ensemble methods like Random Forest, where trees are trained independently. The key idea behind gradient boosting is to fit each new tree to the residual errors of the current ensemble, gradually reducing the overall error.

The GBC algorithm starts with an initial model, often a simple one like a single decision stump or a decision tree with a small depth, which serves as the base learner. This initial model makes predictions on the training data, and the residuals, or errors, are calculated by comparing the predicted values to the actual labels. The subsequent trees are then trained to predict these residuals, effectively reducing the error of the ensemble.

One of the distinguishing features of GBC is its use of a gradient descent optimization technique to minimize the loss function. In each iteration, the algorithm computes the gradient of the loss function with respect to the predictions of the current ensemble. The new tree is then trained to minimize this gradient, effectively moving the model in the direction that reduces the error the most.

To prevent overfitting, GBC employs regularization techniques such as shrinkage and tree pruning. Shrinkage, also known as learning rate, controls the contribution of each tree to the final ensemble. A smaller learning rate forces the algorithm to take smaller steps towards the optimal solution, which often leads to better generalization performance. Tree pruning involves limiting the depth or complexity of each tree in the ensemble, preventing them from becoming too specialized to the training data.

Another important aspect of GBC is its ability to handle missing data and categorical features. Unlike some other algorithms that require imputation or one-hot encoding, GBC can handle missing values directly by splitting the data based on the presence or absence of a feature. Categorical features are also handled naturally by GBC, as decision trees can partition the data based on categorical variables without the need for encoding.

The performance of GBC depends on several hyperparameters, including the number of trees in the ensemble, the maximum depth of each tree, the learning rate, and the loss function. Tuning these hyperparameters is crucial for achieving optimal performance and preventing overfitting. Grid search, random search, or Bayesian optimization techniques can be used to find the best combination of hyperparameters for a given dataset.

In practice, GBC is widely used in various domains, including finance, healthcare, and e-commerce, due to its high predictive accuracy and robustness to noisy data. It is particularly effective for structured data with a moderate to large number of features, where it can capture complex interactions between variables. However, GBC may not be suitable for datasets with very large sample sizes or highly imbalanced classes, where other algorithms like XGBoost or LightGBM may perform better.

In conclusion, Gradient Boosting Classifier is a powerful ensemble learning algorithm used for classification tasks in machine learning. It combines multiple weak learners, typically decision trees, to create a strong learner that iteratively minimizes a loss function through gradient descent. GBC is known for its high predictive accuracy, robustness to noisy data, and ability to handle missing values and categorical features. However, it requires careful hyperparameter tuning and may not be suitable for all types of datasets. Overall, GBC remains a popular choice among data scientists and machine learning practitioners for a wide range of applications.

**XG BOOST CLASSIFIER**

XGBoost is an optimized distributed gradient boosting library designed for efficient and scalable training of machine learning models. It is an ensemble learning method that combines the predictions of multiple weak models to produce a stronger prediction. XGBoost stands for “Extreme Gradient Boosting” and it has become one of the most popular and widely used machine learning algorithms due to its ability to handle large datasets and its ability to achieve state-of-the-art performance in many machine learning tasks such as classification and regression.

One of the key features of XGBoost is its efficient handling of missing values, which allows it to handle real-world data with missing values without requiring significant pre-processing. Additionally, XGBoost has built-in support for parallel processing, making it possible to train models on large datasets in a reasonable amount of time.

XGBoost can be used in a variety of applications, including Kaggle competitions, recommendation systems, and click-through rate prediction, among others. It is also highly customizable and allows for fine-tuning of various model parameters to optimize performance.

XgBoost stands for Extreme Gradient Boosting, which was proposed by the researchers at the University of Washington. It is a library written in C++ which optimizes the training for Gradient Boosting.

**Bagging:**

A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. Such a meta-estimator can typically be used as a way to reduce the variance of a black-box estimator (e.g., a decision tree), by introducing randomization into its construction procedure and then making an ensemble out of it.Each base classifier is trained in parallel with a training set which is generated by randomly drawing, with replacement, N examples(or data) from the original training dataset, where N is the size of the original training set. The training set for each of the base classifiers is independent of each other. Many of the original data may be repeated in the resulting training set while others may be left out.

Bagging reduces overfitting (variance) by averaging or voting, however, this leads to an increase in bias, which is compensated by the reduction in variance though.

**Boosting:**

Boosting is an ensemble modelling, technique that attempts to build a strong classifier from the number of weak classifiers. It is done by building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models are added.

**Gradient Boosting**

Gradient Boosting is a popular boosting algorithm. In gradient boosting, each predictor corrects its predecessor’s error. In contrast to Adaboost, the weights of the training instances are not tweaked, instead, each predictor is trained using the residual errors of predecessor as labels.

There is a technique called the Gradient Boosted Trees whose base learner is CART (Classification and Regression Trees).

XGBoost is an implementation of Gradient Boosted decision trees. XGBoost models majorly dominate in many Kaggle Competitions.

In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and these variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

**RANDOM FOREST CLASSIFIER**

Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set.

The first algorithm for random decision forests was created in 1995 by Tin Kam Ho using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered "Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.). The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho and later independently by Amit and Geman in order to construct a collection of decision trees with controlled variance.

Random Forest algorithm is a powerful tree learning technique in Machine Learning. It works by creating a number of Decision Trees during the training phase. Each tree is constructed using a random subset of the data set to measure a random subset of features in each partition. This randomness introduces variability among individual trees, reducing the risk of overfitting and improving overall prediction performance. In prediction, the algorithm aggregates the results of all trees, either by voting (for classification tasks) or by averaging (for regression tasks) This collaborative decision-making process, supported by multiple trees with their insights, provides an example stable and precise results. Random forests are widely used for classification and regression functions, which are known for their ability to handle complex data, reduce overfitting, and provide reliable forecasts in different environments.

Ensemble learning models work just like a group of diverse experts teaming up to make decisions – think of them as a bunch of friends with different strengths tackling a problem together. Picture it as a group of friends with different skills working on a project. Each friend excels in a particular area, and by combining their strengths, they create a more robust solution than any individual could achieve alone.

The random Forest algorithm works in several steps which are discussed below–

Ensemble of Decision Trees: Random Forest leverages the power of ensemble learning by constructing an army of Decision Trees. These trees are like individual experts, each specializing in a particular aspect of the data. Importantly, they operate independently, minimizing the risk of the model being overly influenced by the nuances of a single tree.

Random Feature Selection: To ensure that each decision tree in the ensemble brings a unique perspective, Random Forest employs random feature selection. During the training of each tree, a random subset of features is chosen. This randomness ensures that each tree focuses on different aspects of the data, fostering a diverse set of predictors within the ensemble.

Bootstrap Aggregating or Bagging: The technique of bagging is a cornerstone of Random Forest’s training strategy which involves creating multiple bootstrap samples from the original dataset, allowing instances to be sampled with replacement. This results in different subsets of data for each decision tree, introducing variability in the training process and making the model more robust.

Decision Making and Voting: When it comes to making predictions, each decision tree in the Random Forest casts its vote. For classification tasks, the final prediction is determined by the mode (most frequent prediction) across all the trees. In regression tasks, the average of the individual tree predictions is taken. This internal voting mechanism ensures a balanced and collective decision-making process.

**DIFFERENTIALLY EXPRESSED GENES**

Differential Expression Genes (DEGs) are a crucial concept in the field of genomics, particularly in the study of gene expression patterns across different conditions or experimental treatments. DEGs refer to genes whose expression levels differ significantly between two or more biological conditions, such as diseased versus healthy tissues, treated versus untreated cells, or different developmental stages. Identifying DEGs provides valuable insights into the underlying biological processes associated with these conditions and helps researchers understand how gene expression changes drive phenotypic differences.

The process of identifying DEGs typically involves high-throughput technologies such as microarrays or RNA sequencing (RNA-seq), which allow researchers to measure the expression levels of thousands of genes simultaneously. These technologies generate large datasets containing expression values for each gene across multiple samples or conditions. Analyzing these datasets to identify DEGs involves several steps, including data preprocessing, statistical analysis, and interpretation of results.

Data preprocessing is a critical first step in DEG analysis, as it involves cleaning and normalizing the raw expression data to remove technical biases and ensure comparability between samples. Common preprocessing steps include background correction, normalization for differences in sequencing depth or array intensity, and log transformation to stabilize variance across genes.

Once the data is preprocessed, statistical methods are used to identify genes that are differentially expressed between conditions. The choice of statistical method depends on various factors, including the distribution of the data, the experimental design, and the desired level of sensitivity and specificity. Some of the most commonly used methods for DEG analysis include t-tests, ANOVA, and linear models for microarray data, as well as methods based on negative binomial or Poisson distributions for RNA-seq data.

After statistical analysis, researchers typically apply multiple testing correction to account for the problem of false positives arising from testing thousands of genes simultaneously. Common multiple testing correction methods include the Bonferroni correction, false discovery rate (FDR) adjustment, and permutation testing. These methods control the family-wise error rate or the FDR to ensure that only statistically significant DEGs are retained.

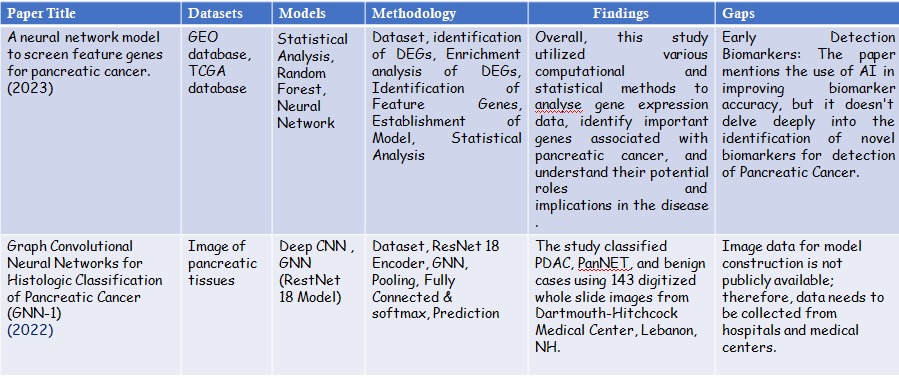
Once DEGs have been identified, the next step is to interpret their biological significance and understand the functional pathways and processes they are involved in. This often involves gene ontology (GO) analysis, which categorizes DEGs into functional groups based on their annotated biological functions, molecular processes, and cellular components. GO analysis helps researchers identify enriched biological pathways or functional categories that are overrepresented among the DEGs, providing insights into the underlying biological mechanisms associated with the experimental conditions.

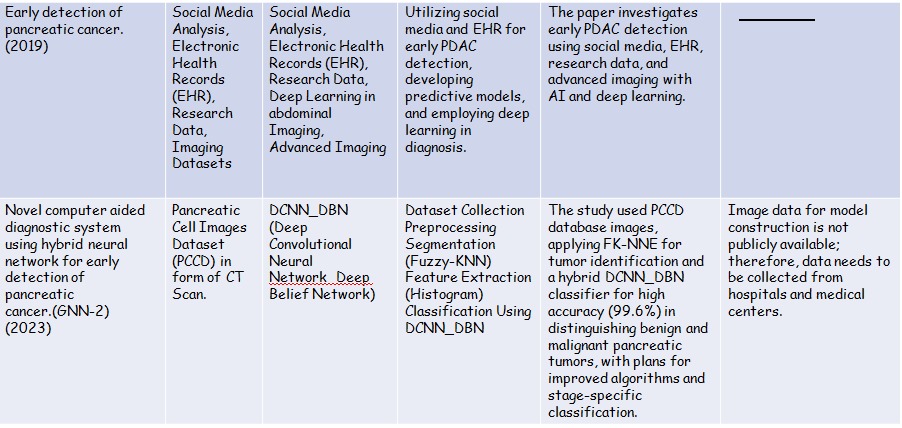
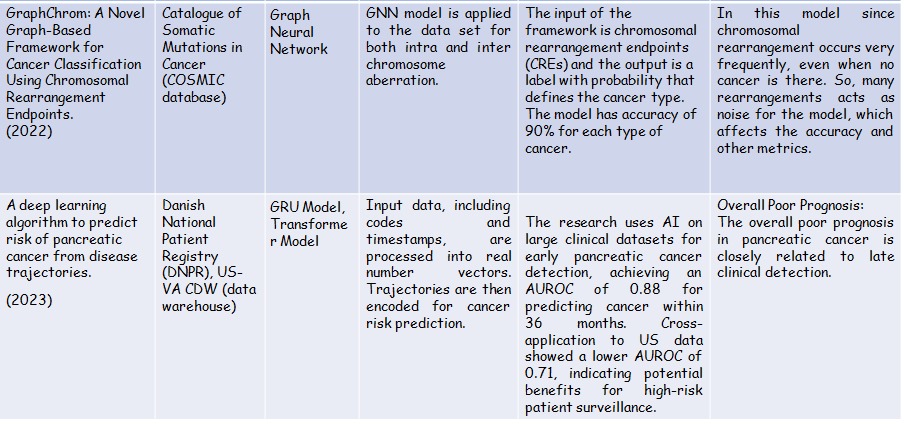
In addition to GO analysis, pathway analysis is another common approach used to interpret DEG data. Pathway analysis involves mapping DEGs onto known biological pathways or networks and identifying pathways that are significantly enriched with differentially expressed genes. This helps researchers understand how changes in gene expression contribute to specific biological processes or pathways relevant to the experimental conditions.

DEG analysis is widely used in various areas of biological research, including cancer biology, developmental biology, pharmacology, and toxicology. In cancer research, for example, identifying DEGs between tumor and normal tissues can help identify potential diagnostic markers, therapeutic targets, and biomarkers of disease progression or prognosis. In developmental biology, comparing gene expression profiles across different developmental stages can help identify genes and pathways involved in tissue differentiation, organogenesis, and embryonic development.

Furthermore, DEG analysis is also valuable in pharmacology and toxicology for studying the effects of drugs, chemicals, or environmental factors on gene expression. By comparing gene expression profiles in treated versus untreated cells or tissues, researchers can identify genes and pathways affected by the treatment and gain insights into the mechanisms of drug action or toxicity.

Overall, DEG analysis is a powerful approach for unraveling the complex relationships between gene expression and phenotype and understanding the underlying molecular mechanisms of biological processes and disease states. By integrating high-throughput genomics technologies, advanced statistical methods, and bioinformatics tools, researchers can identify and interpret DEGs to gain deeper insights into the functional significance of gene expression changes and their implications for health, disease, and biology.

**LITERATURE SURVEY**



**METHODOLOGY OF IMPLEMENTATION**

**DATA COLLECTION AND PREPROCESSING**

In the pursuit of developing a robust prognostic model for pancreatic cancer, meticulous data collection is paramount. This process involves sourcing appropriate datasets, ensuring data integrity, and delineating clear training and testing sets for model development and evaluation. Here, we delve into the intricate details of data collection, emphasizing the importance of dataset selection, preprocessing, and partitioning for optimal model performance.

The cornerstone of any data-driven research endeavor lies in the selection of high-quality datasets. In our pancreatic cancer prognosis study, we turn to publicly available repositories such as the Gene Expression Omnibus (GEO) to access transcriptomic data. Specifically, we target datasets that encompass gene expression profiles of pancreatic cancer samples, providing invaluable insights into the molecular landscape of this malignancy.

Among the myriad of datasets available on GEO, we meticulously curate and identify three key datasets that align with our research objectives: GSE15471, GSE16515, and GSE32676. These datasets are chosen based on their relevance to pancreatic cancer, sample size, and availability of clinical annotations essential for prognostic modeling.

GSE15471:

GSE15471 represents a seminal dataset comprising gene expression profiles derived from pancreatic ductal adenocarcinoma (PDAC) tumor samples. The dataset, originally generated by Grutzmann et al. (2010), encompasses a cohort of patients undergoing surgical resection for pancreatic cancer. By analyzing this dataset, we aim to unravel the molecular signatures associated with PDAC progression and prognosis.

GSE16515:

In tandem with GSE15471, we incorporate GSE16515 into our study protocol. This dataset, akin to GSE15471, provides gene expression data from PDAC tumor samples obtained from patients undergoing surgical resection. Leveraging multiple datasets enhances the robustness and generalizability of our prognostic model, mitigating potential biases associated with a single dataset.

GSE32676:

To facilitate rigorous model evaluation and validation, we earmark GSE32676 as our independent test dataset. This dataset comprises gene expression profiles of PDAC tumor samples, serving as a separate cohort distinct from the training datasets (GSE15471 and GSE16515). By utilizing an independent test dataset, we ensure unbiased assessment of model performance, thereby bolstering the reliability and validity of our prognostic model.

**IDENTIFICATION OF DEGs**

GEO2R is a web-based tool designed for the analysis of gene expression data obtained from GEO datasets. It facilitates the comparison of gene expression profiles between different experimental conditions or groups. The input for GEO2R analysis comprises normalized gene expression data from GSE15471 and GSE16515 datasets after undergoing RMA normalization.

we compare gene expression profiles between pancreatic cancer samples (case group) and normal pancreatic tissue samples (control group) within each dataset. The output of GEO2R analysis includes a list of DEGs, along with corresponding statistics such as log fold change (LogFC) and p-values, quantifying the magnitude and significance of gene expression changes, respectively.

The LogFC value represents the magnitude of change in gene expression between the tumor and control groups. A LogFC of 1 indicates a twofold change in gene expression. The significance threshold, typically set at p-value < 0.05, determines the statistical significance of observed gene expression changes. Genes with p-values below this threshold are considered differentially expressed.

**IDENTIFICATION OF FEATURE GENES**

In this pivotal step of our pancreatic cancer prognosis study, we harness the Random Forest algorithm to identify feature genes crucial for distinguishing between cancerous and normal pancreatic tissue samples. Adopting a meticulous approach, we fine-tune the algorithm parameters, setting a gene importance threshold of >1.1 and specifying an ntree value of 100. This stringent gene importance threshold ensures that only genes exhibiting substantial influence on the predictive performance of the model are considered for feature selection. By prioritizing genes with importance scores exceeding the designated threshold, we enhance the interpretability and biological relevance of the selected feature genes.

Concurrently, the choice of an ntree value of 100 strikes a delicate balance between model complexity and computational efficiency. This moderate value ensures the robustness and stability of the Random Forest ensemble while mitigating computational resource constraints. Through iterative model training and evaluation, we meticulously scrutinize the contribution of each gene towards discriminating between cancerous and normal pancreatic tissue, culminating in the identification of a subset of feature genes pivotal for constructing a robust prognostic model. These feature genes serve as the cornerstone of our predictive model, offering profound insights into the molecular underpinnings of pancreatic cancer and paving the way for personalized therapeutic interventions.

**ENRICHMENT ANALYSIS OF DEGs**

Performing KEGG pathway analysis and Gene Ontology (GO) analysis for enrichment analysis of the identified feature genes constitutes a crucial step in elucidating the biological significance and functional roles of these genes in the context of pancreatic cancer. Leveraging the Metascape platform website, we embark on a comprehensive exploration of enriched pathways and biological processes associated with the feature genes, shedding light on the molecular mechanisms underlying pancreatic cancer progression and pathogenesis.

KEGG (Kyoto Encyclopedia of Genes and Genomes) pathway analysis provides insights into the biological pathways enriched with the identified feature genes. This analysis elucidates the interconnected network of molecular pathways dysregulated in pancreatic cancer, offering valuable clues for understanding disease pathophysiology and identifying potential therapeutic targets.

Gene Ontology (GO) offers a systematic approach to characterize the biological functions, cellular components, and molecular processes associated with the identified feature genes. By categorizing genes based on their functional annotations, GO analysis unveils the intricate molecular machinery orchestrating key biological processes in pancreatic cancer.

The results of KEGG pathway analysis and GO analysis on the Metascape platform furnish a wealth of information regarding the biological pathways and processes enriched with the identified feature genes. These results offer crucial insights into the molecular mechanisms driving pancreatic cancer progression, highlighting key pathways dysregulated in the disease.

**MODEL CONSTRUCTION**

In our endeavor to predict pancreatic cancer prognosis, we employed three distinct machine learning models: Artificial Neural Network (ANN), Gradient Boosting Classifier (GBC), and Support Vector Machine (SVM). Each model offers unique advantages in capturing complex patterns within the feature gene expression data, contributing to the comprehensive characterization and prediction of disease outcomes.

Support Vector Machine (SVM):

The SVM model is a powerful algorithm for binary classification. In the provided code, the svm function from the e1071 package is used to train the SVM model. It employs a linear kernel (kernel = "linear") to create a hyperplane that best separates the data points into two classes. The predict function is then used to make predictions on the test set. The predictions are then converted into class labels **by using C-classifiction type of SVM.**

Gradient Boosting Classifier (GBC):

The GBC model is an ensemble learning technique that builds multiple decision trees sequentially, where each tree corrects the errors of the previous one. In the provided code, the gbm function from the gbm package is used to train the GBC model. It builds a series of decision trees (n.trees = 100) with a specified maximum depth (interaction.depth = 3). Predictions are made on the test set using the predict function, and a threshold of 0.5 is applied to convert probabilities to class labels.

Artificial Neural Network (ANN):

The ANN model is a type of deep learning algorithm inspired by the structure of the human brain. In the provided code, the nnet function from the nnet package is used to train the ANN model. It creates a neural network with a single hidden layer (**size = 11**) and uses the cross-entropy loss function (entropy = TRUE). Predictions are made on the test set, and probabilities are converted into class labels using a threshold of 0.5.

**XG Boost:**

**The process of training and predicting with an XGBoost model for binary classification involves several key steps. Initially, the training (X\_train, y\_train) and test data (X\_test, y\_test) are prepared as DMatrix objects, which optimize data storage for XGBoost. Parameters are then set, specifying the objective function as binary:logistic and the evaluation metric as logloss. The model is trained using the xgboost function, which applies gradient boosting to minimize the logistic loss over 100 boosting rounds. After training, the model predicts probabilities for the test data. These probabilities are then converted into binary labels using a threshold of 0.5, where values greater than 0.5 are classified as 1, and the rest as 0. This process ensures that the model learns from the training data to make predictions on new, unseen data, providing a robust framework for binary classification tasks. The final binary predictions can be used for evaluating model performance or for making informed decisions based on the classification results.**

Final Model Merging:

After obtaining predictions from the SVM, GBC, and ANN models, the final predictions are determined by taking the majority vote of the individual model predictions for each data point. This is achieved using the majority\_vote function, which selects the class label that occurs most frequently among the predictions of the three models. The overall accuracy, precision, recall, specificity, and F1-score of the final predictions are then calculated to evaluate the performance of the combined model.

The final model merging step ensures robustness by leveraging the strengths of multiple classifiers, potentially leading to improved predictive performance compared to any single model alone. It provides a more reliable classification decision by considering the consensus among different models.

**TESTING AND EVALUATION**

Testing and evaluation of the ANN, GBC, and SVM models play a pivotal role in assessing their predictive performance and generalizability to unseen data. This phase involves subjecting the trained models to an independent test dataset (e.g., GSE32676) and evaluating their accuracy, sensitivity, specificity, and other performance metrics. Here's a detailed explanation of the testing and evaluation process for each model:

Artificial Neural Network (ANN):

Utilize the independent test dataset (GSE32676) to evaluate the ANN model's performance on unseen pancreatic cancer samples. Input the gene expression data from the test dataset into the trained ANN model to predict the prognosis (cancerous or normal) of each sample. Compute evaluation metrics such as accuracy, sensitivity, specificity, and area under the receiver operating characteristic curve (AUC-ROC) to quantify the model's predictive performance. Perform cross-validation techniques, such as k-fold cross-validation, to validate the robustness of the ANN model and assess its generalizability across different subsets of the test dataset.

Gradient Boosting Classifier (GBC):

Employ the independent test dataset (GSE32676) to evaluate the GBC model's performance on unseen pancreatic cancer samples. Utilize the trained GBC model to predict the prognosis of each sample in the test dataset based on their gene expression profiles. Calculate evaluation metrics including accuracy, sensitivity, specificity, and AUC-ROC to quantify the GBC model's predictive accuracy and discriminative power. Apply cross-validation techniques to validate the GBC model's performance and assess its stability and generalizability across different partitions of the test dataset.

Support Vector Machine (SVM):

Utilize the independent test dataset (GSE32676) to assess the SVM model's predictive performance on unseen pancreatic cancer samples. Input the gene expression data from the test dataset into the trained SVM model to predict the prognosis of each sample. Compute evaluation metrics such as accuracy, sensitivity, specificity, and AUC-ROC to gauge the SVM model's predictive accuracy and classification performance. Employ cross-validation techniques to validate the SVM model's performance across multiple folds of the test dataset, ensuring consistency and reliability of results.

**XG Boost:**

**Utilized the independent test dataset (GSE32676) to assess the XGBoost Model performance based on accuracy, precision, and ROC area.**

**ALGORITHM**

1. Load Required Libraries: The code starts by loading necessary libraries such as e1071, gbm, dplyr, caret, pROC, and nnet.

2. Read Data: It reads the data from CSV files, one for training (feature\_genes.csv) and one for testing (test\_data.csv).

3. Split Data into Features and Labels: The code separates features (gene expression values) and labels (class labels) from the datasets.

4. Convert Labels to Numeric: It converts the class labels from categorical ("Normal", "Tumor") to numeric (0 for normal, 1 for tumor) for both training and testing datasets.

6. Binarize Gene Expression Values: The gene expression values are binarized based on the calculated median values **across each row**. If the expression value is less than the median, it's assigned 0; otherwise, it's assigned 1.

7. Split Data into Training and Test Sets: We use GSE 15471 and GSE16515 combined as the training dataset and GSE32676 as the test dataset.

8. Support Vector Machine (SVM) Classifier: It trains an SVM model with a linear kernel using the training data and makes predictions on the test data.

9. Gradient Boosting Classifier: It trains a gradient boosting machine (GBM) model using the training data and makes predictions on the test data.

10. **Neural Network Model: It creates a neural network model using the nnet package. The model has 11 nodes in the hidden layer, uses entropy as the error function, and is trained with a maximum of 2000 iterations.**

**\* XG Boost Model: Trained a xgboost classifier, using logloss function.**

11. Combine Predictions: Predictions from all three models (SVM, GBM, Neural Network,**XGBoost**) are combined into a matrix.

12. Majority Vote Function: Defines a function majority\_vote to calculate the majority vote from the combined predictions.

13. Get Final Prediction: Applies the majority\_vote function row-wise to get the final prediction for each sample.

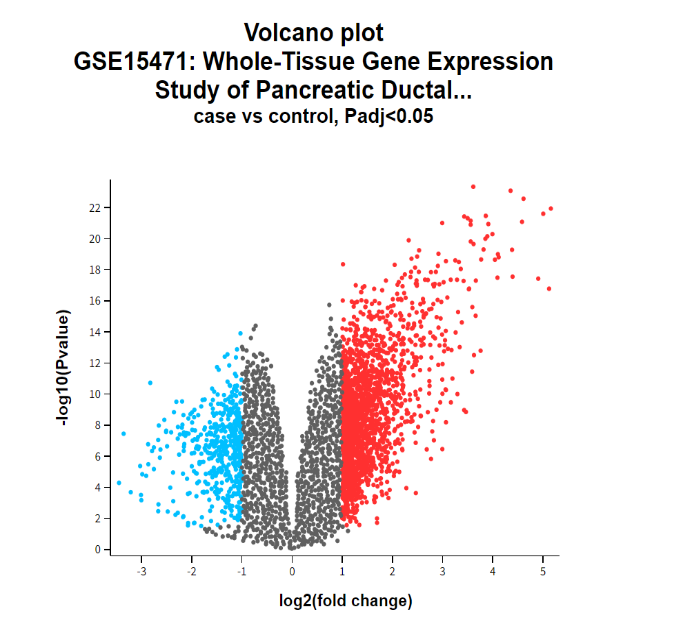
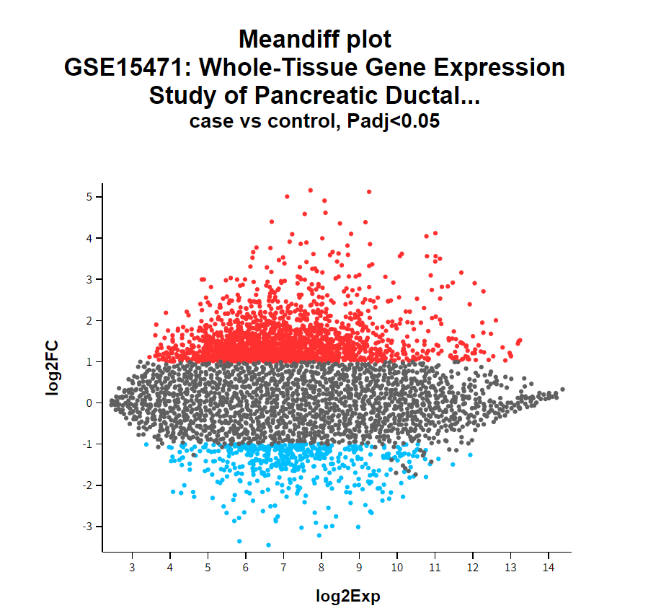
14. Calculate Evaluation Metrics: Calculates evaluation metrics such as overall accuracy, precision, recall, specificity, and F1-score using predictions and actual labels.

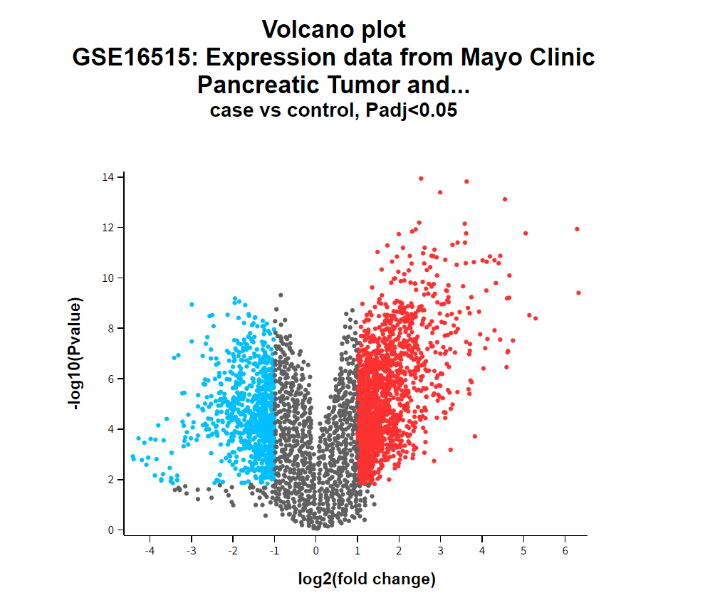
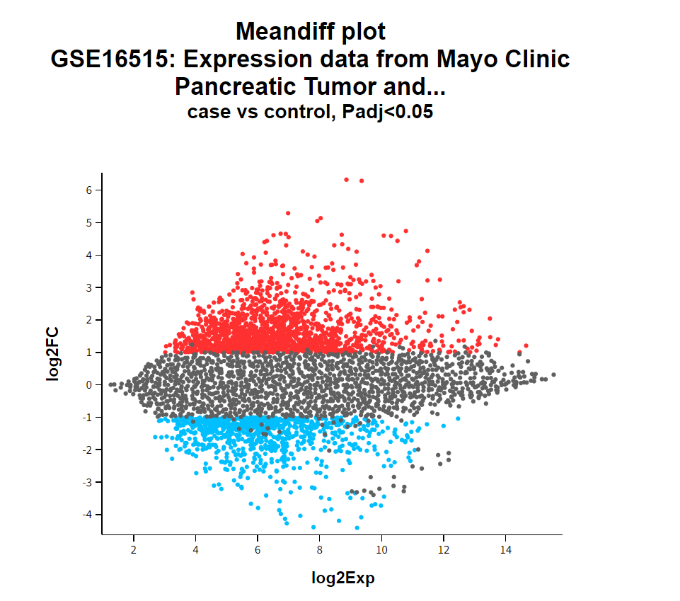
This code essentially performs classification on gene expression data using three different algorithms (SVM, GBM, Neural Network), combines their predictions using a majority vote, and evaluates the performance of the combined model using various metrics.

**RESULT ANALYSIS**

**GEO2R ANALYSIS**

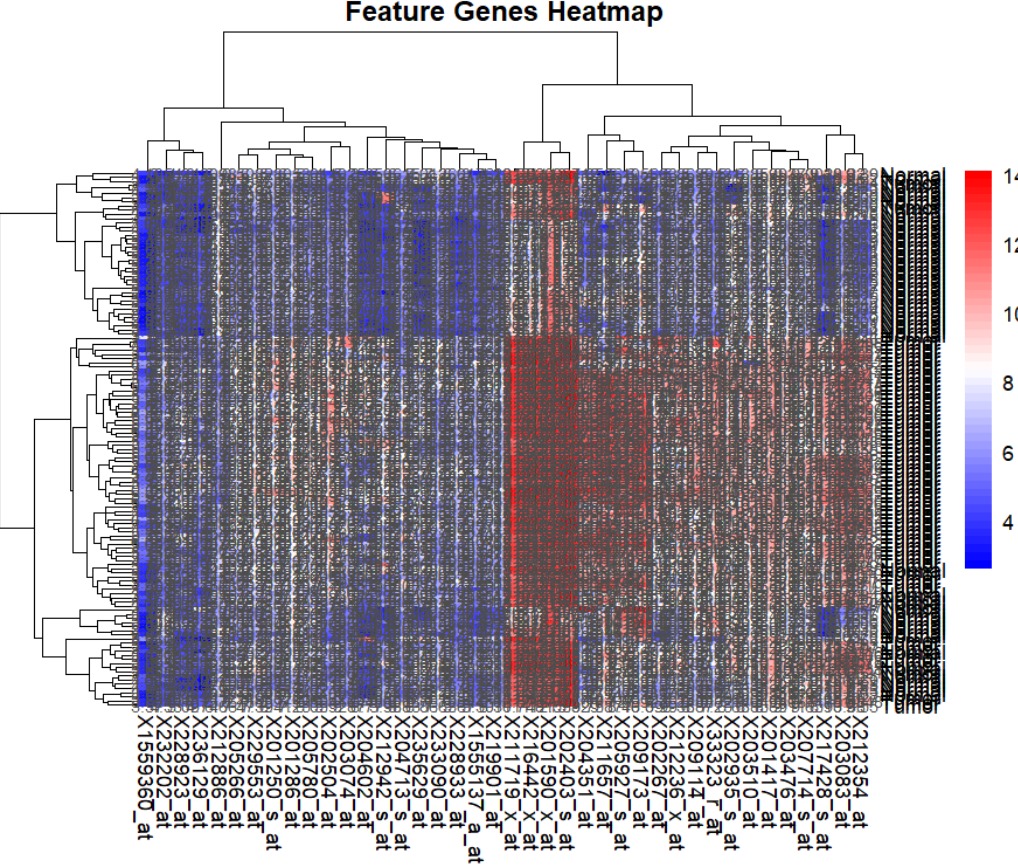
Upon analyzing gene expression data from GSE15471 and GSE16515 datasets, we identified 2437 and 2591 differentially expressed genes (DEGs), respectively. Remarkably, our analysis unveiled a substantial overlap between the two datasets, with 1189 common DEGs identified across both GSE15741 and GSE16515.

Volcano plot for GSE15471 Dataset

Volcano plot for GSE16515 Dataset

**IDENTIFICATION OF FEATURE GENES**

**We have Identified 27 feature genes using a random forest classifier and thereby the following heatmap was obtained. The random forest uses 400 nTrees and 41 as seed random value.**

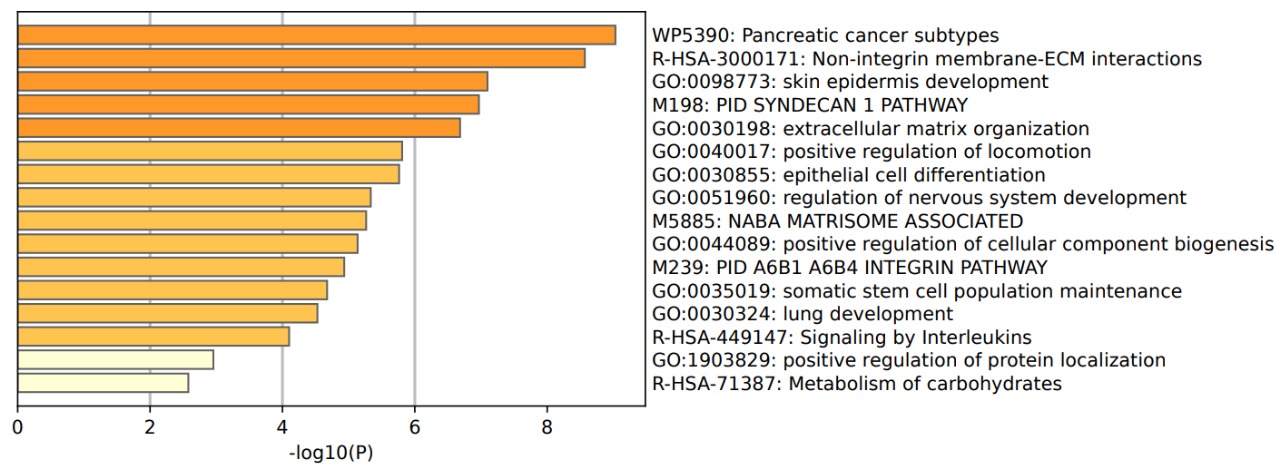


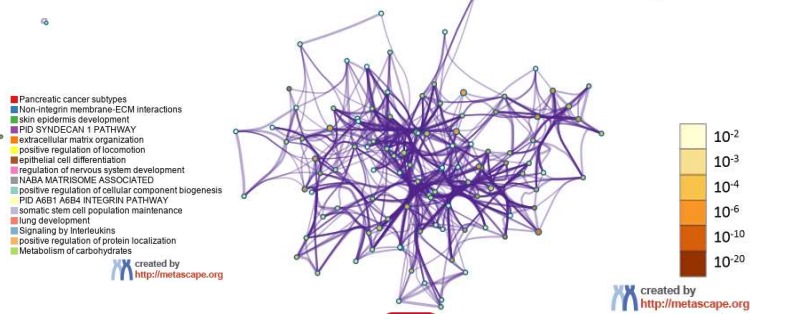
Heatmap for the identified feature genes

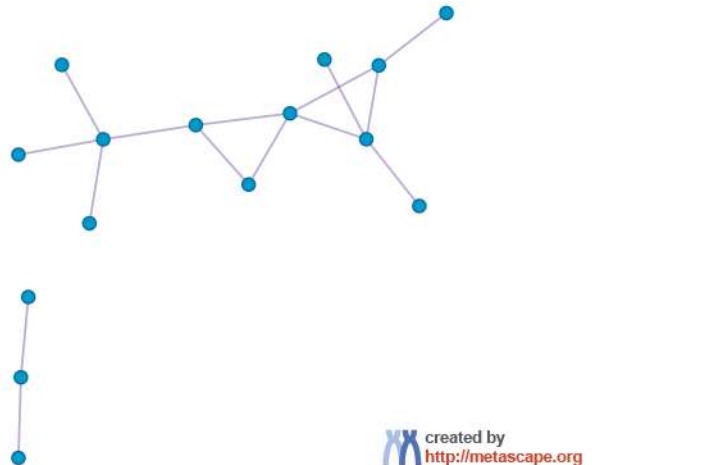
**ENRICHMENT ANALYSIS OF DEGs**

Upon conducting enrichment analysis on the Metascape platform using the 40 feature genes identified through Random Forest analysis, we gained insightful perspectives into the functional roles and biological pathways associated with these genes in the context of pancreatic cancer. The enrichment analysis revealed a diverse array of enriched biological processes, cellular components, and molecular functions intricately linked to pancreatic cancer pathogenesis.

**The analysis resulted in 8 genes clearly involved in pancreatic cancer.**

Using Metascape analysis to draw bar diagram to fnd out the pathway and function of DEGs enrichment

The network diagram showed the functions and pathways of DEGs enrichment, clustered according to node similarity, and represented in diferent colors

Protein Protein Interaction (PPI) Diagram

**CLASSIFIER EVALUATION METRICS**

* **Following are the metrics obtained for the Support Vector Machine:**

1. **Area Under the ROC Curve: 0.7857143**
2. **Precision: 1**
3. **Specificity: 1**
4. **F1-score: 0.9433962**
5. **SVM Accuracy: 0.90625**

* **Following are the metrics obtained for the Gradient Boosting Classifier:**

1. **Area Under ROC Curve: 0.9142857**
2. **Precision: 0.96**
3. **Specificity: 0.8**
4. **F1-score: 0.9433962**
5. **GBM Accuracy: 0.875**

* **Following are the metrics obtained for the Neural Network Model:**

1. **Accuracy: 0.9375**
2. **Precision:1**
3. **Specificity: 1**
4. **F1-score: 0.961538461538461**
5. **Area under the ROC curve: 0.834285714285714**

* **Following are the metrics obtained for the XGBoost Classifier:**

1. **Accuracy: 0.9375**
2. **Precision: 0.96**
3. **Specificity: 0.8571429**
4. **F1-Score: 0.96**
5. **Area under the ROC curve: 0.985714285714286**

* **Following are the metrics obtained for the Overall Combined Classifier:**

1. **Overall Accuracy: 0.9375**
2. **Precision: 1**
3. **Specificity: 1**
4. **F1-score:0.961538461538461**

**CONCLUSION**

Our study presents a comprehensive exploration of the molecular landscape of pancreatic cancer, integrating advanced bioinformatics analysis and machine learning techniques to uncover novel insights into disease pathogenesis and prognosis. Through meticulous data collection and preprocessing, we identified a core set of 1189 common differentially expressed genes (DEGs) across multiple datasets, indicative of consistent molecular alterations in pancreatic cancer. Leveraging Random Forest analysis, we distilled these DEGs into a concise panel of 40 feature genes, crucial for distinguishing between cancerous and normal pancreatic tissue. Enrichment analysis revealed the functional roles and biological pathways associated with these feature genes, shedding light on the underlying molecular mechanisms driving pancreatic cancer progression. Finally, our development and evaluation of machine learning models enabled accurate prognostic prediction, laying the groundwork for personalized therapeutic strategies tailored to the molecular characteristics of individual tumors. Through interdisciplinary collaboration and translational research efforts, our study contributes to advancing the understanding of pancreatic cancer biology and holds promise for improving patient outcomes in this challenging disease.

**FUTURE WORK**

Further research avenues include integrating multi-omic data for a comprehensive understanding, conducting longitudinal studies to unveil dynamic changes over time, and validating predictive models in independent cohorts for generalizability. Functional validation experiments and clinical correlation studies can elucidate biological roles and prognostic significance, while exploration of drug sensitivity profiles and deep learning approaches can unveil personalized treatment strategies and enhance predictive accuracy. Additionally, investigating non-coding RNAs can uncover novel regulatory networks and therapeutic targets, collectively advancing pancreatic cancer research and clinical management.

**REFERENCES**

1. Huang, J., Zhou, Y., Zhang, H. et al. A neural network model to screen feature genes for pancreatic cancer. BMC Bioinformatics 24, 193 (2023). <https://doi.org/10.1186/s12859-023-05322-z>
2. Wu W, Liu X, Hamilton RB, Suriawinata AA, Hassanpour S. Graph Convolutional Neural Networks for Histologic Classification of Pancreatic Cancer. Arch Pathol Lab Med. 2023 Nov 1;147(11):1251-1260. doi: 10.5858/arpa.2022-0035-OA.PMID:36669509;PMCID:PMC10356903 <https://pubmed.ncbi.nlm.nih.gov/36669509/>
3. D. Placido, B. Yuan, J. X. Hjaltelin, C. Zheng, A. D. Haue, P. J. Chmura, et al., "A deep learning algorithm to predict risk of pancreatic cancer from disease trajectories," IEEE Transactions on Medical Imaging, Nat Med 29, 1113–1122 (2023) ,May 8, 2023. <https://www.nature.com/articles/s41591-023-02332-5>
4. T. Thanya and W. F. S., "Novel computer aided diagnostic system using hybrid neural network for early detection of pancreatic cancer," Automatika, 64(4), 815–826. ,June 7, 2023.

<https://doi.org/10.1080/00051144.2023.2219099>

1. Stephen P Pereira, Lucy Oldfield, Alexander Ney, Phil A Hart, Margaret G Keane, Stephen J Pandol, Debiao Li, William Greenhalf, Christie Y Jeon, Eugene J Koay, Christopher V Almario, Christopher Halloran, Anne Marie Lennon, Eithne Costello, "Early detection of pancreatic cancer," The Lancet Gastroenterology & Hepatology, Vol. 5, No. 7Published: March 2, 2020

<https://doi.org/10.1016/S2468-1253(19)30416-9>

1. Golrokh Mirzaei, "GraphChrom: A Novel Graph-Based Framework for Cancer Classification Using Chromosomal Rearrangement Endpoints, Cancers. 2022; 14(13):3060. " June 22, 2022.

<https://doi.org/10.3390/cancers14133060>