International Journal of Engineering Research and Science & Technology

IJERST

ISSN : 2319-5991

www.ijerst.com

Email: editor@ijerst.com or editor.ijerst@gmail.com

International Journal of Engineering Research and Science & Technology

https://doi.org/10.62643/ijitce.2024.v20.i3.pp171-178 ISSN 2319-5991 www.ijerst.com

Vol. 20, Issue 3, 2024

An Empirical model of Data Analysis and Techniques for Breast Cancer Detection

[1] G. Chamundeswari, [2] K.P.S.B.Sasikanth,[3] K.V.S. Rama Krishna, [4]K. Subhashini, [5]A. SaiTeja

[1]Ramachandra College of Engineering, Eluru, Andhra Pradesh, India
[2,3,4,5]SIR C R Reddy College of Engineering, Eluru, Andhra Pradesh, India

Abstract—Breast cancer is a common and deadly disease that affects millions of women worldwide. Early detection is crucial for successful treatment, and machine learning techniques can help improve the accuracy of breast cancer diagnosis. In this work, the breast cancer dataset obtained from Kaggleis analysed using multiple decomposition techniques and classification algorithms, with a particular focus on model optimization.

PCA, Sparse PCA, Fast ICA, and NMF were applied for dimensionality reduction, and Naive Bayes, Logistic Regression, Kernel SVM,SVM,KNN and Random Forest for classification. A combination of hyper parameter tuning and the model performance is optimized using cross-validation.

I INTRODUCTION

Various models of Machine learning have become increasingly popular in various fields for their ability to predict outcomes and classify data accurately. However, the performance of these models is heavily dependent on the selection of the right algorithm and parameters. This project, aims to optimize the performance of machine learning models using various decomposition techniques and classification algorithms. We explore the effectiveness of PCA, Sparse PCA, Fast ICA, and NMF are explored for dimensionality reduction, and the performance of logistic regression, KNN, SVM, kernel SVM, Naive Bayes, and random forest classification algorithms are evaluated using different performance metrics like F1-score, confusion matrix, and specificity. The focus is on identifying the most effective combination of decomposition technique and classification algorithm for model optimization, with the goal of improving the performance of machine learning models in various applications. To showcase the effectiveness of this approach, a breast cancer dataset is used as a sample dataset, but the approach can be applied to any other dataset in different fields of research.

II LITERATURE SURVEY

[1] Evaluated the effectiveness in breast cancer prediction of various machine learning algorithms and found that SVM and Random Forest achieved the best performance.[2]Proposed an ensemble of machine learning models for breast cancer detection, combining regression and classification models. The proposed method achieved high accuracy and outperformed other methods.An overview of machine learning techniques for breast cancer prediction and diagnostics was provided in [3], highlighting the strengths and limitations of various methods.[11]Proposed a data preprocessing method for exam analysis systems, including data cleaning, data transformation, and data integration.Overall, these papers highlight the importance of machine learning and data preprocessing techniques for accurate and reliable classification and prediction in various domains. They also demonstrate the potential of novel methods such as deep learning for improving the interpretability and accuracy of machine learning models.

III PROPOSED SYSTEM

Fig1: Proposed system

Dataset:

The considered dataset “breast cancer” databases were obtained from the University of Wisconsin. Our bodies are made up of cells. The human body has around 100 trillion cells. And those cells normally respond in a predictable manner. They follow specific guidelines, split when taught to divide, and remain silent. When commanded to remain dormant, they remain in a certain place inside their tissue and do not migrate. The collection includes qualities and traits such as: sample code number, lump thickness, id number, uniformity of cell shape, uniformity of cell size, marginal adhesion, normal nucleoli, mitoses, bare nuclei, single epithelial cell size, bland chromatin, and mitochondria. Classification: 2 for benign, 4 for malignant. Dataset is normalized using Z-score normalization.

Decomposition techniques:

The models performance is enhanced even more using different decomposition techniques.

PCA :principal component analysis, is a strategy for lowering a dataset's dimensionality. while retaining as much of the variation in the data as possible. Here are some of the important formulas used in PCA:

International Journal of Engineering Research and Science & Technology

https://doi.org/10.62643/ijitce.2024.v20.i3.pp171-178 ISSN 2319-5991 www.ijerst.com

Vol. 20, Issue 3, 2024

1.Covariance matrix calculation: The covariance matrix is computed by multiplying the covariance of each pair of variables in the dataset by the number of variables in the dataset. The covariance matrix C is given by the following if X is an n x m matrix with n observations and m variables.

$$C = (1/n) * (X - X\_mean) ^T * (X - X\_mean)$$

X_mean is a vector that includes the mean of every variable of length m.

The eigenvalues and eigenvectors of the covariance matrix are computed as follows: The following formula is used to compute the eigenvectors and eigenvalues of the covariance matrix :

$$C * v = \lambda * v$$

where v is an eigenvector of C and λ is the corresponding eigenvalue.

3. Principal components' calculation: The primary components are the linear combinations of the original variables that capture the greatest amount of variance in the data. The Principal component is the product of the original data matrix X and the matrix of eigenvectors V:

$$PC = X * V$$

Where PC is a matrix of n x k dimension of principal components, and V is a matrix of m x k dimension of eigenvectors representingthe k largest eigenvalues.

4.Calculation of the proportion of variance explained: The ratio of the associated eigenvalue to the total eigenvalues indicates the percentage of variation explained by each primary component:

$$p\_i = \lambda\_i / (\sum\lambda)$$

Where, p_i is the percentage of variance explained by the i-th principal component and _i denotes the associated eigenvalue..

5.Calculation of the total variance explained: The sum of the respective eigenvalues is the amount of variation that the first k main components can explain in total:

$$var(k) = \sum\_\{i=1\}^k \lambda\_i$$

Where, var(k) is the total variance explained by the first k principal components.

FAST ICA: FastICA (Fast Independent Component Analysis) is an algorithm used to separate a multivariate signal into independent, non-Gaussian components. The algorithm is based on the minimization of mutual information between the components. Here are the key formulas used in FastICA:

Whitening: The input data is first preprocessed by whitening it to remove any correlation between the components. The whitening operation can be expressed using the following formula:

$$X\_white = E * X$$

The input data is represented by X, E the whitening matrix, and X_white- whitened data.

Nonlinearity: FastICA uses a nonlinearity function, g(x), to estimate the independent components. The nonlinearity function should be a non-quadratic and non-linear function, such as the hyperbolic tangent function or the exponential function.

Contrast function: The contrast function is used to measure the non-Gaussianity of the components. TTheFastICA algorithm seeks to maximisethe components' non-Gaussianity. The contrast function is expressed as follows:

$$J(w) = E[g(w^T x)^2] - k$$

Where, w - weight vector, x - input data, g(.) the nonlinearity function, and k is a constant term.

Gradient descent: The FastICA algorithm uses gradient descent to find the weight vector that maximizes the contrast function. The following is an expression for the weight vector updating rule:

$$w\_new = E[x\ g(w^T x)] - E[g'(w^T x)] * w\_old$$

Wherew_oldindicates the older estimate of the weight vector, w_new, the updated one, g'(.) the nonlinearity function derivative, and E[] the expected value.

Orthogonality constraint: To ensure that the estimated components are independent of each other, the weight vectors are updated in an iterative process while enforcing an orthogonality constraint. This can be achieved by orthogonalizing the weight vectors after each update step.

These formulas are the key components of the FastICA algorithm and are used to estimate independent components in multivariate data.

Sparse PCA: Sparse PCA (Principal Component Analysis) is a variant of PCA that produces sparse principal components, which have only a few non-zero values. This method is useful for feature selection in high-dimensional data. Here are some of the key formulas used in Sparse PCA:

Objective function: The objective of Sparse PCA is to find sparse principal components that describes the data's maximum amount of variance in the data. This can be expressed using the following formula:

$$J(X, V) = ||X - XV||^2 + \lambda||V||\_1$$

X represents input data, V denotes the sparse principal component matrix, ||.|| denotes Frobenius norm, and λ is a parameter that controls the sparsity**.**

Singular Value Decomposition (SVD):Sparse PCA uses SVD to decompose the input data into its principal components. The SVD of the input data X is represented as:

$$X = U\Sigma V^T$$

Where Σ is a diagonal matrix holding the singular values and U and V are orthogonal matrices.

Thresholding:To obtain a sparse principal component matrix, the elements in the matrix V are thresholded using a

International Journal of Engineering Research and Science & Technology

https://doi.org/10.62643/ijitce.2024.v20.i3.pp171-178 ISSN 2319-5991 www.ijerst.com

Vol. 20, Issue 3, 2024

soft-thresholdingoperator. The soft-thresholding operator is represented as follows :

$$S(x, \lambda) = sign(x) \max(|x| - \lambda, 0)$$

X, the input value, λ is the threshold parameter, and S(.) is the soft-thresholding operator.

Eigenvalue calculation:To decide how many sparse main components to keep, the eigenvalues of the input data are computed. The eigenvalues show how much variation is accounted for by each major component..

Explained variance:The eigenvalues are used to compute the variance explained by each sparse main component. The i-th sparse principal component's explanation of variance is given by the following formula:

Variance explained by PC_i = (λ_i / Σλ) * 100%

whereλ_i- the i-th eigenvalue, and Σλ is the sum of all eigenvalues.

These formulas are used to compute the sparse principal components in Sparse PCA. The resulting sparse principal components can be used for feature selection in high-dimensional data**.**

Factory Analysis:Finding underlying factors or dimensions that explain the pattern of correlations within a set of reported variables is done statistically using factor analysis. Here are some key formulas used in factor analysis:

Correlation matrix:The correlations between all pairs of observed variables are shown by the correlation matrix. It is typically denoted as R and has dimensions p x p. The number of observed variables is represented by p. The formula for the correlation between variables i and j is:

$$r_{ij} = cov(x_i, x_j) / (s_i * s_j)$$

where, cov(x_i, x_j) - covariance between variables i and j, and s_i and s_j- the standard deviations of variables i and j, respectively.

Eigenvalues:The amount of variation in the data explained by each component is expressed in terms of eigenvalues. They are obtained by solving the characteristic equation of the correlation matrix:

$$|R - \lambda I| = 0$$

where λ is the eigenvalue and I is the identity matrix.

Eigen vectors:Eigen vectors are the corresponding factor loadings for each eigenvalue. They indicate the degree to which each observed variable is associated with each factor. Eigenvectors can be obtained by solving the equation:

$$(R - \lambda I)\, v = 0$$

where v is the eigenvector and λ is the eigenvalue.

Factor scores: Factor scores are the values of each observation on each factor. They are evaluated as the sum of the product of each observed variable and its relevant factor loading:

$$F_i = b_{1i} * x_{1i} + b_{2i} * x_{2i} + ... + b_{pi} * x_{pi}$$

whereF_i is the factor score for observation i, b_ji is the factor loading for variable j in factor i, and x_ji is the value of variable j for observation i.

Factor rotation:Factor rotation is a technique used to simplify and interpret the factor structure. There are various methods of rotation, but one common approach is orthogonal rotation, which maintains the independence of the factors. The most widely used orthogonal rotation method is the Varimax rotation, which aims to maximize the variance of the squared factor loadings within each factor.

These are some of the key formulas used in factor analysis. There are other more advanced techniques such as confirmatory factor analysis, which involves specifying a priori the structure of the factors and their relationships, and structural equation modeling, whichallows for testing of complex models with multiple latent variables.

Non-negative matrix factorization: Non-negative matrix factorization (NMF) is a dimensionality reduction technique that factorizes on-negative data matrix into two non-negative matrices, typically with lower rank, such that the product approximates the original matrix. Here are some key formulas used in NMF:

Data matrix:The data matrix X has dimensions n x m, n represents the number of observations and m, the number of variables. Each element x_ij in X represents the value of variable j for observation i.

Factor matrices:The goal of NMF is to factorize X into two non-negative matrices A and B, with dimensions n x k and k x m, respectively, where k is the number of factors. The elements a_ik and b_kj in A and B represent the contribution of factor k to the value of variable j for observation i, respectively.

Objective function:The objective of NMF is to find the factor matrices A and B that minimize the reconstruction error between X and the product WH. One commonly used objective function is the Frobenius norm:

$$\min ||X - AB||^2_F$$

where ||.||^2_F is the squared Frobenius norm, defined as the sum of the squared elements of a matrix.

International Journal of Engineering Research and Science & Technology

https://doi.org/10.62643/ijitce.2024.v20.i3.pp171-178 ISSN 2319-5991 www.ijerst.com

Vol. 20, Issue 3, 2024

Updating rules: NMF is typically solved iteratively using updating rules that minimize the objective function. One popular algorithm for NMF is the multiplicative updating rule, which updates A and B alternatively using the following equations:

b_kj<- b_kj * (a_ik * x_ij / (a_ik * (AB)_ij + epsilon))

a_ik<- a_ik * (x_ij * b_kj / ((AB)_ij * b_kj + epsilon))

To prevent division by zero, there is a tiny positive constant called epsilon.

Sparsity constraint: NMF can be extended to impose sparsity on the factor matrices, which can help identify interpretable features in the data. To achieve sparsity is to add a penalty term to the objective function, such as the L1 norm:

min ||X - AB||^2_F + lambda ||A||_1 + mu ||B||_1

where lambda and mu are regularization parameters that control the sparsity of A and B, respectively, and ||.||_1 is the L1 norm, defined as the sum of the absolute values of the elements of a matrix.

These are some of the key formulas used in NMF. There are other variations and extensions of NMF, such as sparse NMF, non-negative tensor factorization, and non-negative sparse PCA, which employ different regularization constraints and optimization algorithms.

In this system, we have used different classification techniques to analyses the dataset that is assigned. The system will return the algorithm with best accuracy among the algorithms used.

Classification techniques used,

a) Logistic Regression
b) Support Vector Machine
c) K-Nearest Neighbor
d) Kernel Support Vector Machine
e) Naive Bayes
f) Random Forest
g) Decision Tree

Among all these techniques one is selected based on their performances.

Training the system:

1.**Logistic Regression**

Logistic regression is a statistical technique used to predict a binary outcome variable (such as true or false, yes, or no) based on one or more predictor variables. It models the probability of the outcome variable using the logistic function, which maps any real-valued input to a probability score between 0 and 1. The logistic regression model estimates the values of the regression coefficients that maximize the likelihood of the observed data using the maximum likelihood estimation method. It is commonly used in various fields, including medicine, economics, and machine learning for binary classification tasks. The goal is to predict the probability of occurring an event.

The Logistic regression formula is given by.

p(y=1 | x) = 1 / (1 + exp(-z))

Where,

(y=1 | x) is the probability of the event occurring given the predictor variables

x denotes the predictor variables

z = b0 + b1x1 + b2x2 + ... + bpxp is the linear combination of the predictor variables and their respective coefficients, where b0 is the intercept and b1, b2, ..., bp are the coefficients for the predictor variables.

2.**Support Vector Machine**

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

The Support Vector Machine is given by

w*x + b = 0

where w is the weight vector, x is the input vector, and b is the bias.

3. K-Nearest Neighbor

KNN, or K-Nearest Neighbors, is a type of machine learning algorithm that is used for classification and regression tasks. It is a non-parametric method that works by comparing an input example to the k nearest training examples in a feature space, and then assigning a label or value to the input example based on the majority label or average value of its k nearest neighbors.

Formula used to calculate distance between points:

$$\text{Euclidean Distance between } A_1 \text{ and } B_2 = \sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$$

4. **Kernel Support Vector Machine**

Kernel SVM (Support Vector Machine) is a type of machine learning algorithm used for classification and regression tasks. It is a supervised learning method that works by finding a hyperplane that separates different classes in a high-dimensional space.

f(x) = sum(alpha_i * y_i * K(x, x_i)) + b

International Journal of Engineering Research and Science & Technology

https://doi.org/10.62643/ijitce.2024.v20.i3.pp171-178 ISSN 2319-5991 www.ijerst.com

Vol. 20, Issue 3, 2024

where f(x) is the decision function, alpha_i is the Lagrange multiplier, Y_i is the label of the ith training example, x_i is the ith training example, K(x, x_i) is the kernel function, and b is the bias.

5.**Naive Bayes**

Naive Bayes is a probabilistic machine learning algorithm used for classification tasks. It is based on Bayes' theorem, which states that the probability of a hypothesis (in this case, a class label) given some observed evidence (in this case, input features) is proportional to the probability of the evidence given the hypothesis times the prior probability of the hypothesis.

$$P(c|x) = P(x|c) * P(c) / P(x)$$

where P(c|x) is the probability of class c given the input features x, P(x|c) is the probability of observing the input features x given the class c, P(c) is the prior probability of class c, and P(x) is the probability of observing the input features x.

6.**Random Forest**

Random Forest is a machine learning algorithm that is used for classification and regression. It is a collection of multiple decision trees, where each tree is trained on a random subset of the data and outputs a prediction. The final prediction of a Random Forest is determined by taking the average (for regression) or majority vote (for classification) of the predictions from individual trees. This combination of multiple trees makes the algorithm more robust to overfitting compared to a single decision tree.

$$f(x) = mode\{p1(x), p2(x), ..., pm(x)\}$$

7.**Decision Tree**

Decision tree is a popular machine learning algorithm that is widely used for both classification and regression tasks. It is a tree-structured model that makes of loadings (regression coefficients), and E is the residual matrix.

Y = UQ' + F, where Y is the matrix of dependent variables, U is the matrix of scores (weights), Q is the matrix of loadings (regression coefficients), and F is the residual matrix.

The scores and loadings are estimated by iteratively minimizing the residuals in both X and Y. The final regression equation can then be expressed as Y = XB + E, where B is the matrix of regression coefficients.

Evaluation Metrics:

At this stage the metric for the different algorithms is evaluated. Based on those metrics the best algorithm is evaluated for the data set. For the calculation of the metric the used methods are

1.Accuracy

2.Specificity

3.F1 score

a. K -fold cross validation

Accuracy: Accuracy is a common evaluation metric used in classification tasks to measure the proportion of correctly classified instances out of the total number of instances in a dataset. It is calculated as follows:

Accuracy = (Number of Correct Predictions) / (Total Number of Predictions)

where the number of correct predictions is the number of instances that are correctly classified by the model, and the total number of predictions is the total number of instances in the dataset.

Specificity: Specificity is a commonly used evaluation metric in binary classification tasks to measure the proportion of true negative predictions out of the total number of negative instances in a dataset. It is calculated as follows:

Specificity = True Negatives / (True Negatives + False Positives)

where True Negatives (TN) are the number of instances that are truly negative and are correctly classified as negative by the model, and False Positives (FP) are the number of instances that are truly negative but are incorrectly classified as positive by the model.

F1-score: F1-score is a common evaluation metric used in classification tasks that combines precision and recall into a single value. It is calculated as the harmonic mean of precision and recall, as follows:

F1-score = 2 * (Precision * Recall) / (Precision + Recall)

Where Precision is the proportion of true positive predictions out of the total number of positive predictions, and Recall (also known as sensitivity) is the **proportion of true** positive predictions out of the total number of positive instances in the dataset.

Precision = True Positives / (True Positives + False Positives)

Recall = True Positives / (True Positives + False Negatives)

Where True Positives (TP) are the number of instances that are truly positive and are correctly classified as positive by the model, False Positives (FP) are the number of instances that are truly negative but are incorrectly classified as positive by the model, and False Negatives (FN) are the number of instances that are truly positive but are incorrectly classified as negative by the model.

K-fold Cross Validation: K-fold cross-validation is a commonly used method for evaluating the performance of a machine learning model by partitioning the dataset into k equally sized folds, using k-1 folds for training the model, and evaluating it on the remaining fold. This process is

International Journal of Engineering Research and Science & Technology

https://doi.org/10.62643/ijitce.2024.v20.i3.pp171-178

ISSN 2319-5991 www.ijerst.com

Vol. 20, Issue 3, 2024

repeated k times, with each fold serving as the validation set once.

The average performance of the model across the k-folds is then computed as the evaluation metric, such as accuracy, precision, recall, or F1-score. The formula for k-fold cross-validation is as follows:

Partition the dataset into k equally sized folds.

For each fold i in k:

a. Train the model on the k-1 folds except for fold i.

b. Evaluate the model on fold i and record the evaluation metric.

Compute the average evaluation metric across the k-folds.

The formula for computing the average evaluation metric is as follows:

Average Evaluation Metric = (Evaluation Metric Fold 1 + Evaluation Metric Fold 2 + ... + Evaluation Metric Fold k) / k

IV EXPERIMENTAL RESULTS

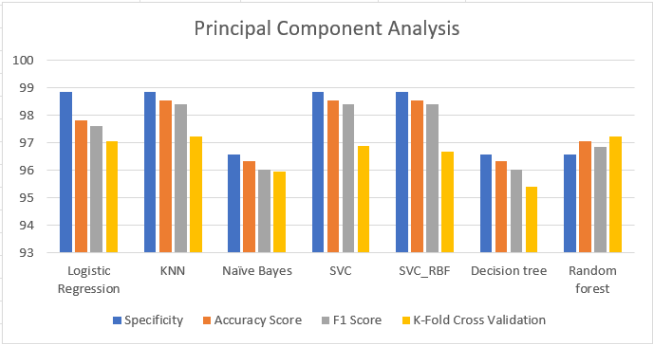

Fig 2: Performance Evaluation using PCA

The results of each classification technique under different metric evaluation are as shown above. The above graph depicts the results that are achieved by using Principle Component Analysis.

Table 1: Performance Evaluation using PCA

PCA	Specificity	Accuracy Score	F1 Score	K-Fold Cross Validation
Logistic Regression	98.86	97.81	97.6	97.06
KNN	98.86	98.54	98.41	97.25
Naïve Bayes	96.59	96.35	96.04	95.95
SVC	98.86	98.54	98.41	96.88
SVC_RBF	98.86	98.54	98.41	96.69
Decision tree	96.59	96.35	96.04	95.41
Random forest	96.59	97.08	96.85	97.24

From the above results, the algorithms that produced the best results were Logistic regression, KNN, Support Vector Classifier and Support Vector Classifier RBF with **98.86%** under Specificity.

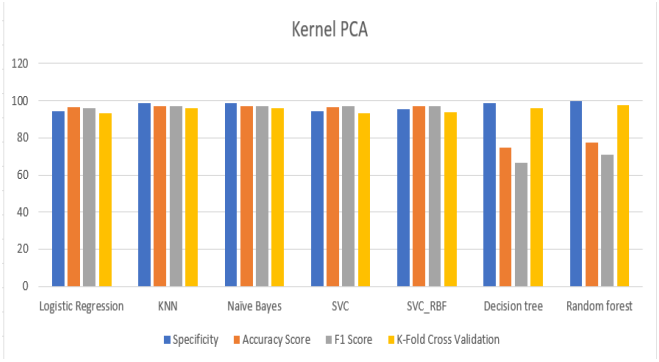

Fig 3: Performance Evaluation using Kernal PCA

The above graph depicts the results that are achieved by using Kernal PCA.

Table 2: Performance Evaluation using Kernal PCA

KERNEL PCA	Specificity	Accuracy Score	F1 Score	K-Fold Cross Validation
Logistic Regression	94.11	96.35	96.19	93.41
KNN	98.82	97.08	96.87	96.16
Naïve Bayes	98.82	97.08	96.87	96.16
SVC	94.11	96.35	96.87	93.23
SVC_RBF	95.29	97.08	96.94	93.97
Decision tree	98.82	74.45	66.73	95.98
Random forest	100	77.37	71.05	97.43

From the above results, the algorithms that produced the best results were Random forest with 100%, KNN, Naïve Baye's and Decision tree with **98.82%** under Specificity.

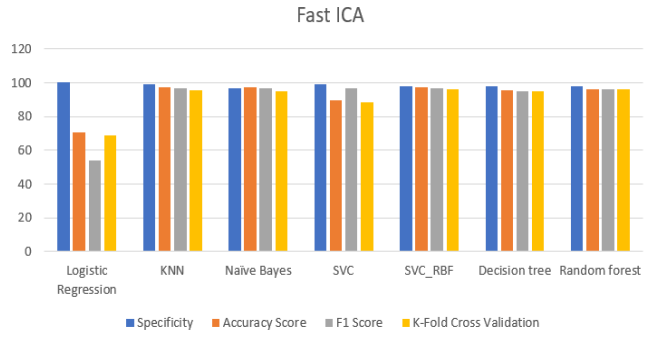

Fig 4: Performance Evaluation using Fast ICA

The results of each classification technique under different metric evaluation are as shown above. The above graph depicts the results that are achieved by using Fast ICA.

Table 3: Performance Evaluation using Fast ICA

International Journal of Engineering Research and Science & Technology

https://doi.org/10.62643/ijitce.2024.v20.i3.pp171-178

ISSN 2319-5991 www.ijerst.com

Vol. 20, Issue 3, 2024

Fast ICA	Specificity	Accuracy Score	F1 Score	K-Fold Cross Validation
Logistic Regression	100	70.8	53.87	68.67
KNN	98.88	97.08	96.72	95.43
Naïve Bayes	96.66	97.08	96.79	94.9
SVC	98.88	89.78	96.72	88.27
SVC_RBF	97.77	97.08	96.76	95.99
Decision tree	97.77	95.62	95.09	94.88
Random forest	97.77	96.35	95.93	96.35

From the above results, the algorithms that produced the best results were Logistic regression with**100%,**KNN and SVC with **98.88% under** Specificity.

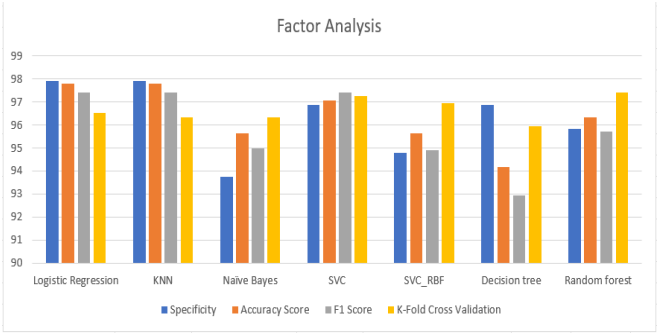

Fig 5: Performance Evaluation using Factor Analysis

The results of each classification technique under different metric evaluation are as shown above. The above graphs depicts the results that are achieved by using Factor Analysis.

Table 4: Performance Evaluation using Factor Analysis

FACTOR ANALYSIS	Specificity	Accuracy Score	F1 Score	K-Fold Cross Validation
Logistic Regression	97.91	97.81	97.4	96.52
KNN	97.91	97.81	97.4	96.33
Naïve Bayes	93.75	95.62	94.97	96.33
SVC	96.87	97.08	97.4	97.25
SVC_RBF	94.79	95.62	94.91	96.96
Decision tree	96.87	94.16	92.93	95.96
Random forest	95.83	96.35	95.73	97.43

From the above results, the algorithms that produced the best results were Logistic Regression and KNN with **97.91%** under Specificity.

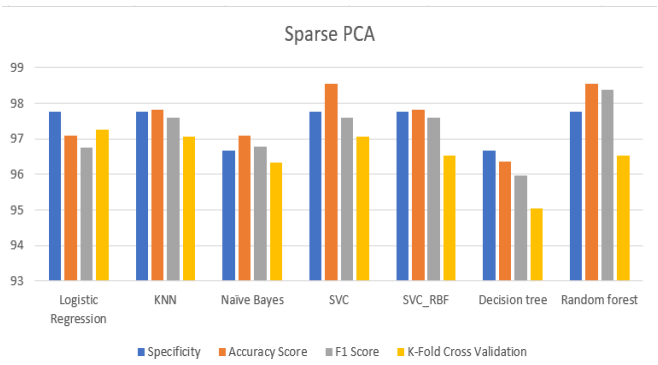

Fig 6: Performance Evaluation using Sparce PCA

The results of each classification technique under different metric evaluation are as shown above. The above graphs depict the results that are achieved by using Sparse PCA.

Table 5:Performance Evaluation using Sparce PCA

Sparse PCA	Specificity	Accuracy Score	F1 Score	K-Fold Cross Validation
Logistic Regression	97.77	97.08	96.76	97.25
KNN	97.77	97.81	97.58	97.07
Naïve Bayes	96.66	97.08	96.79	96.33
SVC	97.77	98.54	97.58	97.06
SVC_RBF	97.77	97.81	97.58	96.52
Decision tree	96.66	96.35	95.97	95.05
Random forest	97.77	98.54	98.39	96.52

From the above results, the algorithms that produced the best results were Support Vector Classifier and Random Forest with **98.54% under** Accuracy Score.

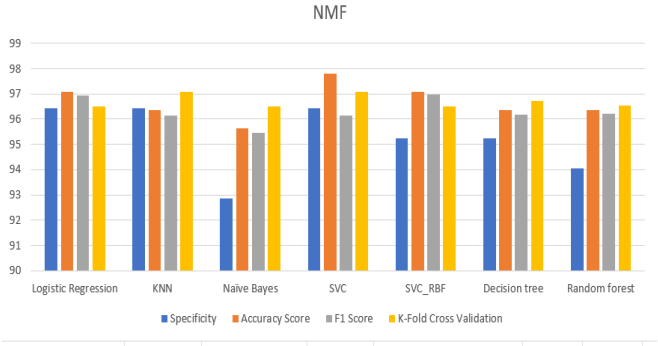

Fig 7: Performance Evaluation using NMF

The results of each classification technique under different metric evaluation are as shown above. The above graph depicts the results that are achieved by using Non Negative Matrix Factorization.

Table 6:Performance Evaluation using NMF

NMF	Specificity	Accuracy Score	F1 Score	K-Fold Cross Validation
Logistic Regression	96.42	97.08	96.94	96.52
KNN	96.42	96.35	96.16	97.07
Naïve Bayes	92.85	95.62	95.46	96.52
SVC	96.42	97.81	96.16	97.07
SVC_RBF	95.23	97.08	96.96	96.52
Decision tree	95.23	96.35	96.19	96.72
Random forest	94.04	96.35	96.21	96.53

From the above results, the algorithms that produced the best results were Support Vector Classifier with **97.81%,** Logistic Regression and Support Vector Machine RBF with **97.08%** under accuracy score.

International Journal of Engineering Research and Science & Technology

https://doi.org/10.62643/ijitce.2024.v20.i3.pp171-178

ISSN 2319-5991 www.ijerst.com

Vol. 20, Issue 3, 2024

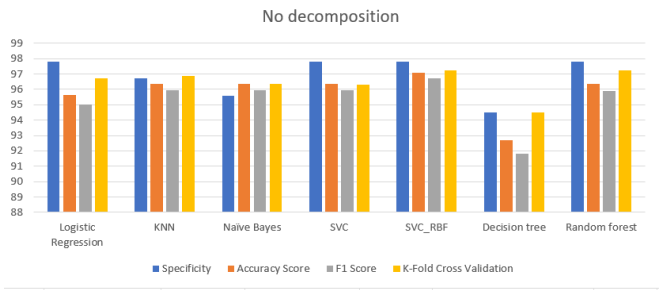

Fig 8: Performance Evaluation under no decomposition

The results of each classification technique under different metric evaluation are as shown above. The above graphs depicts the results are achieved without any decomposition techniques.

Table 7:Performance Evaluation under no decomposition

No decompositi	Specificity	Accuracy Score	F1 Score	K-Fold Cross Validation
Logistic Regression	97.8	95.62	95.03	96.7
KNN	96.7	96.35	95.93	96.89
Naïve Bayes	95.6	96.35	95.97	96.34
SVC	97.8	96.35	95.93	96.33
SVC_RBF	97.8	97.08	96.72	97.26
Decision tree	94.5	92.7	91.81	94.51
Random forest	97.8	96.35	95.88	97.25

From the above results, the algorithms that produced the best results were Logistic regression , Random forest, Support Vector Classifier and Support Vector Classifier RBF with **97.8%** under Specificity

VI.CONCLUSION

This research paper reflects the metric evaluation of different regression techniques on the same dataset. Among all those different regression techniques Random Forest came as the best technique. By this metric evaluation the conclusion is that the Random Forest is best suitable for the prediction of the price. The same experimentation can be done with other datasets.

REFERENCES

[1] Effectiveness evaluation of machine learning algorithms for breast cancer prediction

[2] Ensemble of Machine Learning Fusion Models for Breast Cancer Detection Based on the Regression Model

[3] Machine Learning Algorithms for Breast Cancer Prediction and Diagnosis

[4] Research Paper Classification using Supervised Machine Learning Techniques

[5] Classification Algorithms on Datamining: A Study

[6] XAI Framework for Cardiovascular Disease Prediction Using Classification techniques.

[7] Hierarchical Medical Classification Based on DLCF

[8]A Python Tool for Machine Learning with Feature Ranking and Reduction

[9] DataPre-processing for Supervised Leaning

[10] Multi-label Dysfluency Classification

[11] Research on Data Pre-processing in Exam Analysis System

[12] Multivariate comparison of classification performance measures

[13] Land Use Land Cover Classification Using Different ML Algorithms onSentinel-2 Imagery.

[14] Satellite image classification and quality parameters using ML classifier.

[15]A Review on Data Pre-processing Techniques Toward Efficient and Reliable Knowledge Discovery From Building Operational Data