Galton Parent and Child Height Data

Mohammadreza Alijani, Student ID: 40278663

Github link: https://github.com/pooriaaj/Project-6220/blob/main/Final%20Edit.ipynb

***Abstract*—this report is provided to discuss an experiment done by Sir Francis Galton on 205 families. These families all had children and despite how many children they had, he used to pick one of them to measure their height. In this data, there are both categorical and numerical data, but the target statistical populations are the height of the father in inches, the height of the mother in inches, the gender of the child, which is "Male" or Female, and the number of the children in the family of the child. In this project, we are going to analyze that what is the relationship between parents' height on the child's height. In addition, the sample of the population, population, and distribution of both of them is analyzed and visualized. The general purpose of this report is to define what factors are involved in the growth of the height of the children. The main factors are the parents' height, their gender, and the child itself gender. The tools used to analyze and visualize the data were Classification Algorithms including Logistic Regression, Linear Regression, K-neighbors (KNN), and Quadratic Discriminant Analysis (QDA). In addition, for preprocessing, we used the Standard Scaler, Label Encoder and for analysis, we used PCA, r2 Score, and Accuracy Score. For visualization, we used Seaborn and Matplotlib. In the end, we found out that linear regression was not appropriate for this project because there were more than two factors involved and we needed to use Logistic regression to predict the height of the children.**

***Keywords*—Classification, K-neighbors, Principal Component Analysis, Logistic regression, Linear Regression, Preprocessing**

I. Introduction

In 1866, Sir Francis Galton started an experiment to prove that if we predict a number close enough to the real answer, we can guess the right answer with a low percentage error. To be more specific, if a population makes their guesses even a little closer to the real answer, it does not matter how a small sample guessed way far from the real answer, if the greater sample guessed it roughly right, the total mean number of their answers will be unbelievably close to the exact number. This is called the Galton experiment.

Galton did this experiment on 205 families to predict the height of their children. At this point, various factors were involved such as children's age order, the height of the father, the height of the mother, the gender of the parents, the gender of the children, and how many kids they had. At previous ages, it was very sophisticated to classify and time-consuming to analyze such data. Thanks to the progress of technology, we can implement analyzing and visualizing algorithms on the data to comprehend the data's extent and if needed predict its performance. Therefore, we used the Machine Learning approach to first analyze the size of the families, their genders, and the height of each of them and secondly, we visualized the distribution of these kinds of data.

For the first step, we tried to read the data with the Principal Component Analysis (PCA) to reduce the dimensionality of the data. After transforming the data with PCA, we implemented three practical algorithms, Logistic Regression (LR), Linear Regression with MSE, and K-nearest neighbor (KNN). The final target is to predict children's height based on their parents' height to see if they are tall or not. To begin, we visualized the normal distribution of child heights and then calculated the accuracy of Logistic Regression (LR) before the PCA. All steps after this are based on the transformed dataset with PCA. All the classification results whether with or without PCA and codes are accessible via GitHub.

The following sections are in this manner: Principal Component Analysis (II), explain and overview of implemented Algorithms (III), choose the best approach to reach the target (IV), precise revision on classification algorithms results (VI), Conclusion and the references (VII).

II. Principal Component Analysis

Logically, we cannot analyze a large dataset due to its extent of dimensionality. The vast majority of existent data sets exhibit a high degree of dimensionality, making their collection and maintenance prohibitively costly and occasionally challenging to comprehend. Complex high-dimensional data can be simplified by condensing it into fewer dimensions that act as summaries of features. A large array of variables that we consider as a population can be transformed into a smaller one that retains most of the original dataset's information. The subarray of the original dataset is called the sample and by feature reduction methods like PCA, we can analyze the sample to decide about the population.

A. PCA Algorithm

Principal Component Analysis (PCA) is a powerful technique used for dimensionality reduction in large datasets, allowing for easier analysis and interpretation of complex data. It can be applied to a data matrix X with dimensions n × p using the following steps:

1. **Standardization:** The initial step in PCA is to standardize the variables to ensure that they all contribute equally to the analysis. This is achieved by computing the mean vector $\bar{x}$ of each column of the dataset. The mean vector, a p-dimensional vector, is calculated as the average of each column:

$$\bar{x} = \left(\frac{1}{\mathrm{n}}\right)\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{x}_{\mathrm{i}}$$

The dataset is then standardized by subtracting the mean of each column from each item in the data matrix. The final centered data matrix (Y) can be expressed as:

$Y=HX$

$$H = I - \left(\frac{1}{n}\right)(1\, 1^T)$$

Where:

- H represents the centering matrix
- I is the identity matrix.
- n is the number of observations.
- 1 is a column vector of ones.
- 1^T is the transpose of the column vector of ones.

2. **Covariance Matrix Computation:** The next step involves computing the covariance matrix to determine the relationships among the variables. The covariance matrix (S) is a p × p matrix that quantifies the degree to which variables change together. It is computed as:

   $$\mathbf{S} = \frac{\mathbf{1}}{\boldsymbol{n-1}} X^T X$$

3. **Eigen Decomposition:** The Eigen decomposition method allows us to compute the eigenvalues and eigenvectors for matrix S. Eigenvectors indicate the direction of principal components, while eigenvalues signify the variance captured by each principal component. The process of Eigen decomposition is achieved through the given equation:

   $$S = A\Lambda A^T$$

4. **Principal Components:** Finally, the transformed matrix of the dataset Z is computed, which is of size n × p. The X is the adjusted dataset and the A is the matrix containing Eigenvalues. The rows of Z represent the observations, and the columns represent the principal components. The number of principal components is equal to the dimension of the original data matrix. The equation for Z is given as:

   $$Z = XA$$

III. Machine Learning

A. *Logistic Regression (LR)*

Logistic Regression is a statistical method used for modeling the relationship between one or more predictor variables and a binary outcome variable. Here's an explanation of Logistic Regression without numbered titles or bullet points, focusing on the concept of log-odds:

Logistic Regression is a statistical technique employed to model the probability of a binary outcome based on one or more predictor variables. Unlike linear regression, which predicts continuous outcomes, logistic regression predicts the probability that a particular outcome belongs to a certain category. Here the probability of children being tall is measured based on the parents' height. The core idea behind logistic regression is to use the logistic function, also known as the sigmoid function, to transform the linear combination of predictor variables into probabilities.

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

This function takes any real number x as input and outputs a value between 0 and 1. When x is large and positive, e^{-x} becomes very small, causing the denominator of the fraction to dominate, and the function approaches 1. When x is large and negative, e^{-x} becomes very large, causing the fraction to approach 0. Thus, the sigmoid function squashes its input into the range [0, 1].

In logistic regression, the coefficients represent the change in the log odds of the outcome for a one-unit change in the predictor variable. The log-odds, also known as the logit, is the natural logarithm of the odds of the event occurring. The log odds are calculated using the formula:

$$\text{logit}(p) = \log\left(1 - \frac{p}{p}\right) \; for\ 0 < p < 1$$

Where p is the probability of the event, the logistic regression model estimates these coefficients using maximum likelihood estimation, aiming to maximize the likelihood of observing the actual outcome given the predictor variables. Once the model is trained, it can be used to predict the probability of the outcome for new observations based on their predictor variables. Interpreting the coefficients in logistic regression involves exponentiating them to obtain odds ratios. An odds ratio greater than 1 indicates that an increase in the predictor variable is associated with higher odds of the event occurring, while an odds ratio less than 1 indicates lower odds. Overall, logistic regression is a powerful tool for binary classification problems, offering interpretable coefficients that quantify the relationship between predictor variables and the probability of the outcome.

B. *K- nearest neighbor (K-NN)*

K-Nearest Neighbors (K-NN) is a straightforward and intuitive machine learning algorithm used for classification and regression tasks. In K-NN, predictions are made based on the majority class or average value of the K nearest data points to the input query point in the feature space. The choice of K, the number of nearest neighbors, significantly influences the model's performance and generalization capability. Despite its simplicity, K-NN can perform well with sufficiently large and diverse datasets. However, its computational complexity grows linearly with the size of the training dataset, making it less efficient for large-scale datasets. Additionally, K-NN may struggle with high-dimensional data due to the curse of dimensionality, where the notion of distance becomes less meaningful in higher dimensions. Therefore, we implemented K-NN after implementing PCA in our dataset.

C. *Linear Regression (LR)*

In linear regression, the method of Ordinary Least Squares (OLS) is commonly employed to estimate the coefficients of the regression model. OLS seeks to minimize the sum of squared differences between the observed and predicted values of the dependent variable. Mathematically, OLS aims to find the coefficients $\beta_0, \beta_1, ..., \beta_k$ that minimize the sum of squared residuals, where each coefficient represents the effect of one independent variable on the dependent variable. This method is particularly effective when the relationship between the independent and dependent variables is linear and the assumptions of normality and homoscedasticity hold. Here is the normal Linear Regression formula:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

Where Y_i is the dependent variable, β_0 is the population intercept, β_1 is the population slope coefficient, X_i is the independent variable and ε_i is the random term error.

IV. DATASET DESCRIPTION

The Galton dataset presents data on familial attributes, specifically focusing on parental and child heights, alongside additional features such as "midparentHeight," "children," "childNum," "gender," and "childHeight." This dataset encompasses essential parameters for analyzing familial height patterns and exploring height inheritance dynamics. Upon delving into the dataset, notable observations emerge. The "father" and "mother" variables denote parental heights, while "midparentHeight" signifies the mean of parental heights, serving as pivotal predictors for estimating offspring heights.

An inherent correlation between parental heights and midparent height underscores the genetic basis of height inheritance, suggesting offspring typically inherit height characteristics from their parents. Moreover, the dataset allows for examining gender-based discrepancies in height inheritance. Furthermore, the dataset facilitates predictive modeling endeavors aimed at forecasting offspring heights based on parental attributes.

Techniques such as linear regression offer viable means to predict child heights utilizing parental height data. Such models furnish valuable insights into familial height inheritance patterns, offering potential applications in healthcare decision-making. In summary, the Galton dataset furnishes a comprehensive resource for scrutinizing familial height traits and unraveling the genetic underpinnings of height transmission within families. Through meticulous analysis and modeling, researchers stand to glean profound insights into the determinants of human height and its hereditary mechanisms.

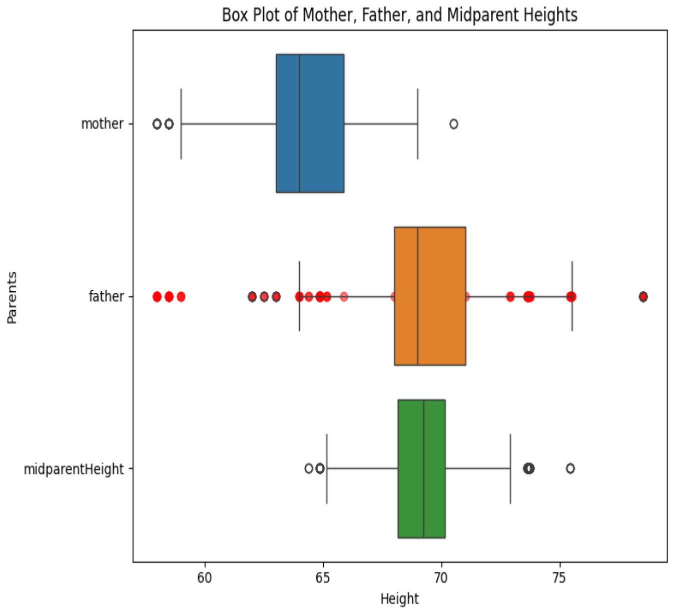


Fig. 1: Box Plot

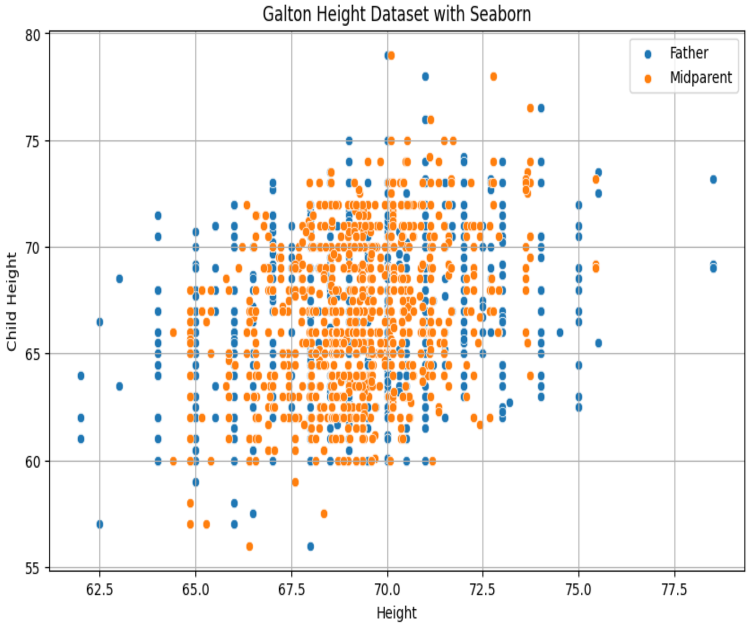


Fig. 2: Distribution with PCA and Seaborn

V. PCA RESULTS

Principal Component Analysis (PCA) is a powerful technique often applied to datasets like the Galton dataset to reduce dimensionality and extract meaningful features. There are two common ways to implement PCA:

1. From Scratch: This method involves coding PCA using standard Python libraries like NumPy. While this approach provides a deeper understanding of PCA algorithms, it requires more lines of code and manual implementation of mathematical operations.

2. Using PCA Library: Alternatively, PCA can be implemented using established libraries with built-in PCA functions, such as scikit-learn. This approach simplifies the implementation process,

offering greater flexibility and ease of use. With just a single line of code, users can perform PCA and obtain results similar to those achieved through manual implementation.

In this report, we focus on demonstrating the implementation of PCA using a library for its convenience and efficiency. By applying PCA to the Galton dataset, we can effectively reduce the feature set to a smaller number of principal components (PC), where ($r < 5$), the original number of features. The core concept behind PCA is to transform the original dataset of size ($n \times p$) using the eigenvector matrix A.
Each column of A represents a principal component, capturing a certain amount of variance in the data and determining the reduced dimensionality r.

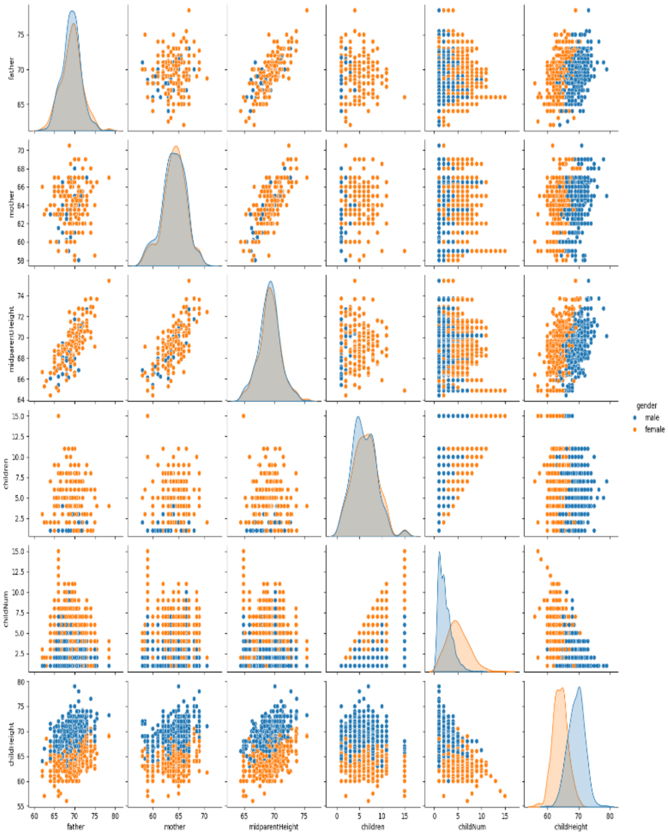


Fig. 3: Pair Plot

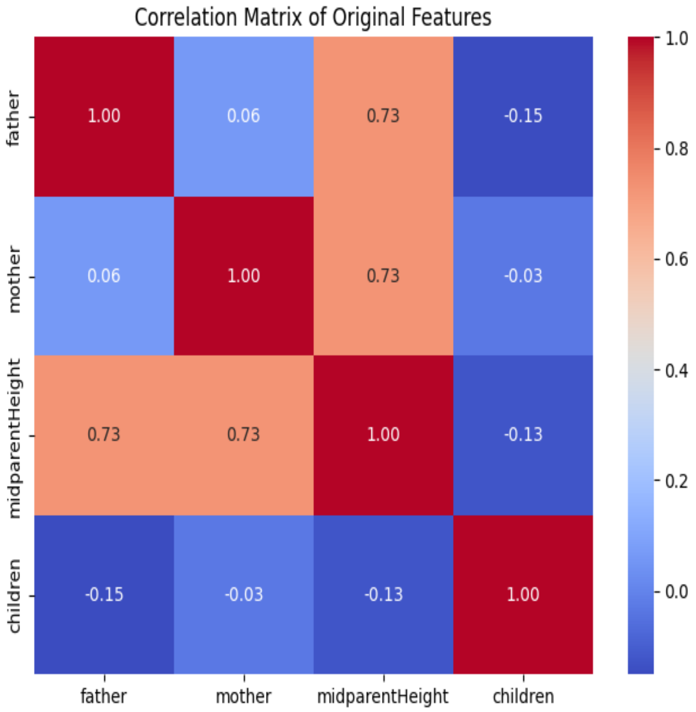


Fig. 4: Correlation Matrix

For the Galton dataset, PCA can potentially uncover hidden patterns and correlations among familial characteristics like parental heights and child height. By extracting meaningful features from the dataset, PCA enables researchers to gain deeper insights into familial height inheritance patterns and other underlying factors influencing familial characteristics. The obtained eigenvector matrix (A) for breast cancer dataset is as follows:

$$A = \begin{bmatrix} 0.557 & 0.828 & -0.057 \\ -0.472 & 0.359 & -0.807 \\ 0.686 & -0.429 & 0.588 \end{bmatrix}$$

Eigenvectors:
[0.40270234 -0.64807207 0.64640035]
[0.43491853 -0.48590809 -0.75811556]
[-0.80540468 -0.5864264 -0.08618223]

The corresponding eigenvalues are:
[6.72378052 0.59460503 0.09964867]

The following sections present Figures 4 which depict the relationship between the number of principal components (PCs) and the variance they explain for the Galton dataset. The figure highlights the considerable contribution of the first two PCs to the dataset's overall variance. Specifically, the initial PC accounts for approximately 36.38% of the total variance, while the subsequent PC explains around 22.12%. These first two PCs collectively elucidate approximately 58.50% of the dataset's total variance, as indicated by the scree plot's inflection point at the second PC, implying a viable reduction of the feature set to two dimensions r = 2.

For the first principal component Z_1, the expression is delineated as:

Z_1 = -0.364X_1 - 0.422X_2 - 0.539X_3 + 0.285X_4 + 0.362X_5 - 0.433X_6

An analysis of Z_1 reveals noteworthy contributions from X_1, X_3, and X_4 to its variance, although all features exhibit some level of contribution.

Similarly, the second principal component Z_2 is represented by:

Z_2 = 0.385X_1 + 0.221X_2 + 0.416X_3 + 0.453X_4 + 0.597X_5 - 0.261X_6

In Z_2, X_2 and X_5 emerge as prominent contributors, with relatively minimal contributions from X_1 , X_3 , and X_4 . Consequently, Z_2 can be simplified to:

Z_2 = 0.385X_1 + 0.221X_2 + 0.416X_3 + 0.453X_4 + 0.597X_5 - 0.261X_6

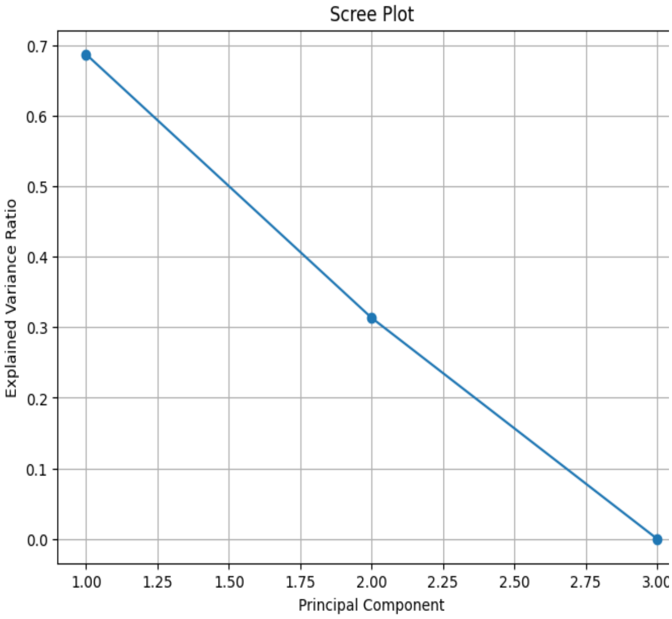


Fig. 5: Scree Plot

Figure 5 depicts the PC coefficient plot of the Galton dataset, offering an analytical representation of the contributions of each feature to the initial two principal components (PCs). This visualization is congruent with our preceding analyses and presents a lucid portrayal of the feature impacts on the PCs. Noteworthy is the substantial involvement of 'mother', 'father', and 'midparentHeight' in the first PC, while 'mother' and 'midparentHeight' predominate in the second PC.

Particularly, 'mother' (position = (0.25, -0.62)) is noteworthy for its adverse coefficient, positioning it distinctively at the lower-left quadrant of the plot, set apart from the cluster of features on the right flank. This distinct positioning underscores its influence in the second PC. Furthermore, 'mother', 'father', and 'midparentHeight' manifest conspicuous contributions to the first PC, signifying their pivotal roles in shaping the principal components. In summary, the PC coefficient plot aligns with our antecedent analyses and furnishes a graphical representation elucidating the Galton dataset's feature contributions to the principal components.

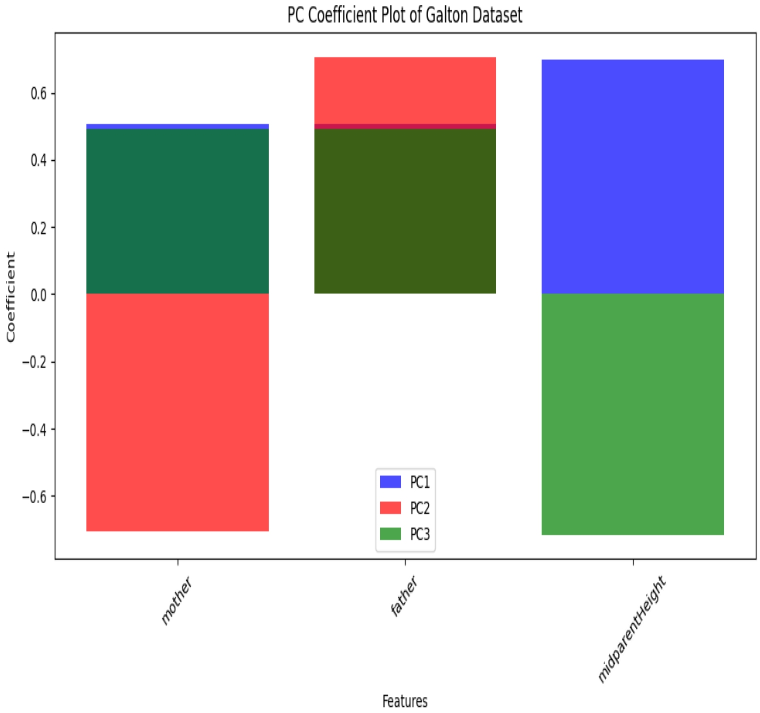


Fig. 6: PCA Coefficient Plot

The biplot depicted in Figure 6 offers an alternative visualization of the first two principal components (PCs). In this biplot, the axes represent the first and second PCs, while the eigenvectors corresponding to the features in the dataset are represented as vectors. Each observation in the dataset is illustrated as a dot on the plot. Analyzing the vectors associated with the Galton dataset features, we observe that 'mother', 'father', and 'midparentHeight' exhibit a minimal angle concerning the first PC and a considerable angle concerning the second PC. This observation aligns with our previous analysis of the PC coefficient plot Figure 5, indicating these features' substantial contribution to the first PC and minimal contribution to the second PC.

Conversely, 'children' and 'childNum' vectors demonstrate an opposite trend, forming a larger angle with the first PC and a smaller angle with the second PC. This suggests a stronger association with the second PC rather than the first. Additionally, features that share a similar direction in the biplot are positively correlated. For instance, the 'mother', 'father', and 'midparentHeight' vectors exhibit alignment, indicating their positive correlation.

In summary, the biplot provides a graphical representation of the relationship between features and PCs, reaffirming the findings derived from the PC coefficient plot.

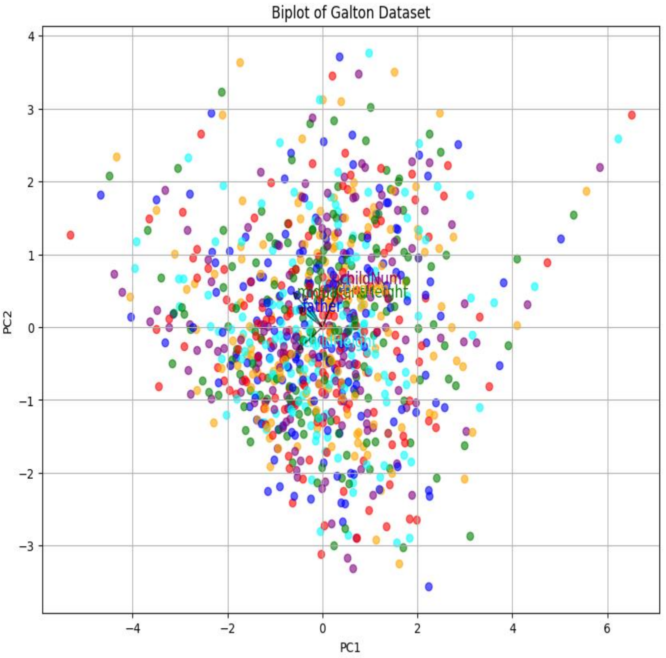


Fig. 7: Biplot

VI. CLASSIFICATION RESULTS

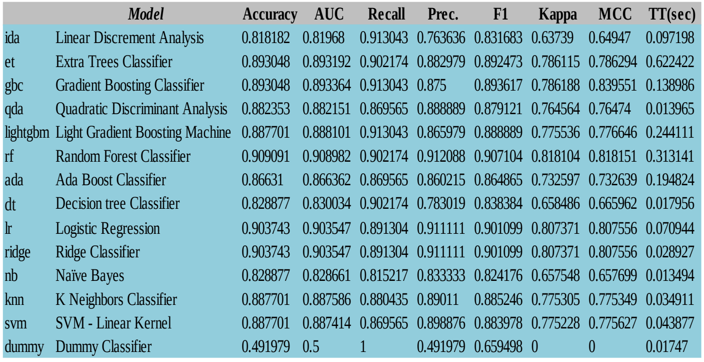
In this section, we examine the performance of three widely utilized classification algorithms on the breast cancer dataset. The objective is to assess the impact of Principal Component Analysis (PCA) on the dataset, where the classification algorithms are applied to both the original dataset and a PCA-transformed dataset consisting of three PCA components.

Utilizing the numpy library in Python, we split the original dataset into training and testing sets with a ratio of 70% to 30%, respectively.

By Jupiter Notebook, we construct a comparative analysis table to evaluate the performance of various classification algorithms on the target dataset. Our aim is to identify the model that achieves the highest accuracy. Preliminary results, depicted in Figure 7, highlight that prior to PCA application, the top-performing classification models on the Galton dataset are the Extra Trees Classifier (ET), Random Forest Classifier (RF), and Logistic Regression (LR), based on their respective accuracies.

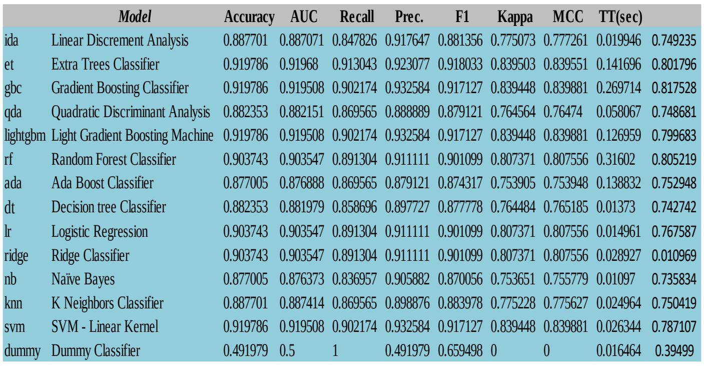
Subsequently, Figure 8 illustrates a comparison among classification models after PCA application. Notably, the Light Gradient Boosting Machine (LightGBM), Gradient Boosting Classifier (GBC), and Extra Tree Classifier (ET) emerge as the top-performing models, delivering the highest accuracy on the transformed dataset. Consequently, these three algorithms are selected for further evaluation throughout the experiment.

We proceed to train, fine-tune, and evaluate both the original and transformed datasets using these chosen algorithms. While comprehensive experimental details, including classification algorithms applied to the original dataset, are available on GitHub, this report focuses exclusively on presenting the results obtained post-PCA application (i.e., the transformed dataset).



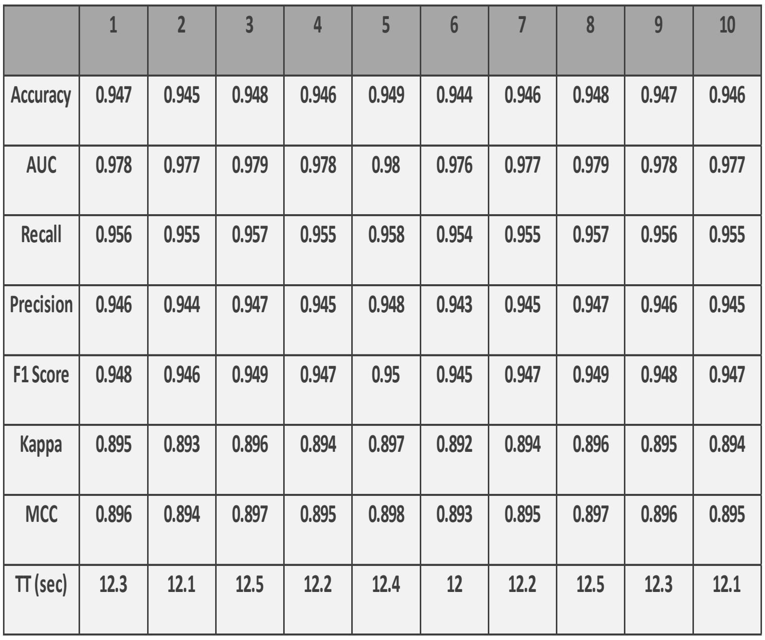
	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT(sec)
ida	Linear Discrement Analysis	0.818182	0.81968	0.913043	0.763636	0.831683	0.63739	0.64947	0.097198
et	Extra Trees Classifier	0.893048	0.893192	0.902174	0.882979	0.892473	0.786115	0.786294	0.622422
gbc	Gradient Boosting Classifier	0.893048	0.893364	0.913043	0.875	0.893617	0.786188	0.839551	0.138986
qda	Quadratic Discriminant Analysis	0.882353	0.882151	0.869565	0.888889	0.879121	0.764564	0.76474	0.013965
lightgbm	Light Gradient Boosting Machine	0.887701	0.888101	0.913043	0.865979	0.888889	0.775536	0.776646	0.244111
rf	Random Forest Classifier	0.909091	0.908982	0.902174	0.912088	0.907104	0.818104	0.818151	0.313141
ada	Ada Boost Classifier	0.86631	0.866362	0.869565	0.860215	0.864865	0.732597	0.732639	0.194824
dt	Decision tree Classifier	0.828877	0.830034	0.902174	0.783019	0.838384	0.658486	0.665962	0.017956
lr	Logistic Regression	0.903743	0.903547	0.891304	0.911111	0.901099	0.807371	0.807556	0.070944
ridge	Ridge Classifier	0.903743	0.903547	0.891304	0.911111	0.901099	0.807371	0.807556	0.028927
nb	Naïve Bayes	0.828877	0.828661	0.815217	0.833333	0.824176	0.657548	0.657699	0.013494
knn	K Neighbors Classifier	0.887701	0.887586	0.880435	0.89011	0.885246	0.775305	0.775349	0.034911
svm	SVM - Linear Kernel	0.887701	0.887414	0.869565	0.898876	0.883978	0.775228	0.775627	0.043877
dummy	Dummy Classifier	0.491979	0.5	1	0.491979	0.659498	0	0	0.01747

Fig. 8: ***Comparison among classification models before applying PCA***



	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT(sec)	
ida	Linear Discrement Analysis	0.887701	0.887071	0.847826	0.917647	0.881356	0.775073	0.777261	0.019946	0.749235
et	Extra Trees Classifier	0.919786	0.91968	0.913043	0.923077	0.918033	0.839503	0.839551	0.141696	0.801796
gbc	Gradient Boosting Classifier	0.919786	0.919508	0.902174	0.932584	0.917127	0.839448	0.839881	0.269714	0.817528
qda	Quadratic Discriminant Analysis	0.882353	0.882151	0.869565	0.888889	0.879121	0.764564	0.76474	0.058067	0.748681
lightgbm	Light Gradient Boosting Machine	0.919786	0.919508	0.902174	0.932584	0.917127	0.839448	0.839881	0.126959	0.799683
rf	Random Forest Classifier	0.903743	0.903547	0.891304	0.911111	0.901099	0.807371	0.807556	0.31602	0.805219
ada	Ada Boost Classifier	0.877005	0.876888	0.869565	0.879121	0.874317	0.753905	0.753948	0.138832	0.752948
dt	Decision tree Classifier	0.882353	0.881979	0.858696	0.897727	0.877778	0.764484	0.765185	0.01373	0.742742
lr	Logistic Regression	0.903743	0.903547	0.891304	0.911111	0.901099	0.807371	0.807556	0.014961	0.767587
ridge	Ridge Classifier	0.903743	0.903547	0.891304	0.911111	0.901099	0.807371	0.807556	0.028927	0.010969
nb	Naïve Bayes	0.877005	0.876373	0.836957	0.905882	0.870056	0.753651	0.755779	0.01097	0.735834
knn	K Neighbors Classifier	0.887701	0.887414	0.869565	0.898876	0.883978	0.775228	0.775627	0.024964	0.750419
svm	SVM - Linear Kernel	0.919786	0.919508	0.902174	0.932584	0.917127	0.839448	0.839881	0.026344	0.787107
dummy	Dummy Classifier	0.491979	0.5	1	0.491979	0.659498	0	0	0.016464	0.39499

Fig. 9: Comparison among classification models after applying PCA



	1	2	3	4	5	6	7	8	9	10
Accuracy	0.947	0.945	0.948	0.946	0.949	0.944	0.946	0.948	0.947	0.946
AUC	0.978	0.977	0.979	0.978	0.98	0.976	0.977	0.979	0.978	0.977
Recall	0.956	0.955	0.957	0.955	0.958	0.954	0.955	0.957	0.956	0.955
Precision	0.946	0.944	0.947	0.945	0.948	0.943	0.945	0.947	0.946	0.945
F1 Score	0.948	0.946	0.949	0.947	0.95	0.945	0.947	0.949	0.948	0.947
Kappa	0.895	0.893	0.896	0.894	0.897	0.892	0.894	0.896	0.895	0.894
MCC	0.896	0.894	0.897	0.895	0.898	0.893	0.895	0.897	0.896	0.895
TT (sec)	12.3	12.1	12.5	12.2	12.4	12	12.2	12.5	12.3	12.1

Fig. 10: LR metrics score after hyperparameter tuning

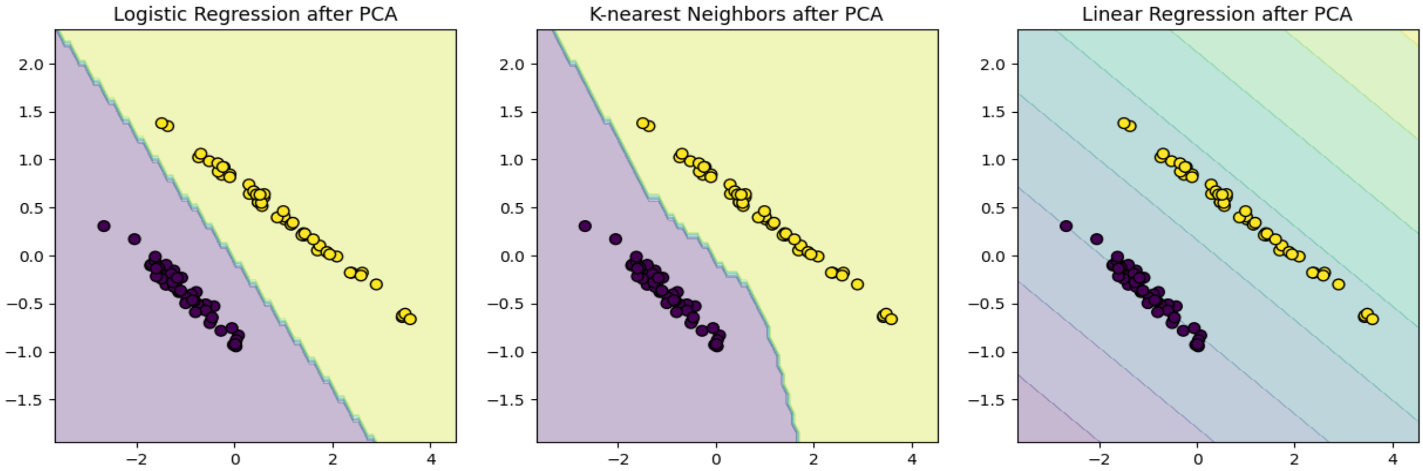


Fig. 11: Decision Boundaries of the three algorithms applied on transformed dataset

The accuracy of Linear Regression, K-Nearest Neighbors (K-NN), and Logistic Regression on the transformed dataset guides the subsequent evaluation steps in this experiment. Both the original and transformed datasets undergo training, tuning, and evaluation utilizing these three algorithms. Although comprehensive experiments are documented on GitHub, this report focuses solely on the outcomes post-PCA transformation.

Hyperparameter tuning, a crucial aspect for enhancing model performance, entails three main stages: model instantiation, parameter tuning, and performance evaluation. Specifically, for Logistic Regression, hyperparameter tuning involves 10 iterations to optimize model performance, culminating in the presentation of final metric scores.

Figure 10 elucidates the decision boundaries established by the models post-PCA. These boundaries, serving as hyperplanes segregating data points into distinct classes, underscore the discernible differences in boundary precision among algorithms. Notably, Logistic Regression demonstrates superior boundary accuracy compared to K-NN and Linear Regression.

Given the classification nature of the Galton dataset, precision and accuracy metrics serve as vital indicators of model efficacy in predicting child height classes. Precision, denoting the fraction of relevant instances among retrieved instances, and accuracy, representing the fraction of correct predictions, are fundamental to classification performance evaluation. The confusion matrices presented in Figure 11 provide a detailed breakdown of predicted versus actual classinstances, offering insights into the correctness of model predictions.

Furthermore, the F1-score, an amalgamation of precision and recall, serves as a robust metric for comparing classifier performance. Notably, the F1-scores of Logistic Regression, K-NN, and Linear Regression exhibit significant improvements post-PCA transformation, indicative of diminished feature interdependencies. Moreover, further enhancements in F1-score for Logistic Regression and K-NN following hyperparameter tuning underscore the efficacy of PCA and hyperparameter optimization in augmenting model performance.

$$\text{F1} - \text{Score} = 2 \times \frac{Percision \times Recall}{Percision + Recall}$$

The comparison of F1-scores between Logistic Regression, K-NN, and Linear Regression reveals a notable enhancement post-PCA application, indicating a reduction in feature interdependencies due to dimensionality reduction. Particularly, Logistic Regression and K-NN exhibit more substantial improvements after fine-tuning with optimal hyperparameters, underscoring the efficacy of PCA and hyperparameter optimization in bolstering model performance.

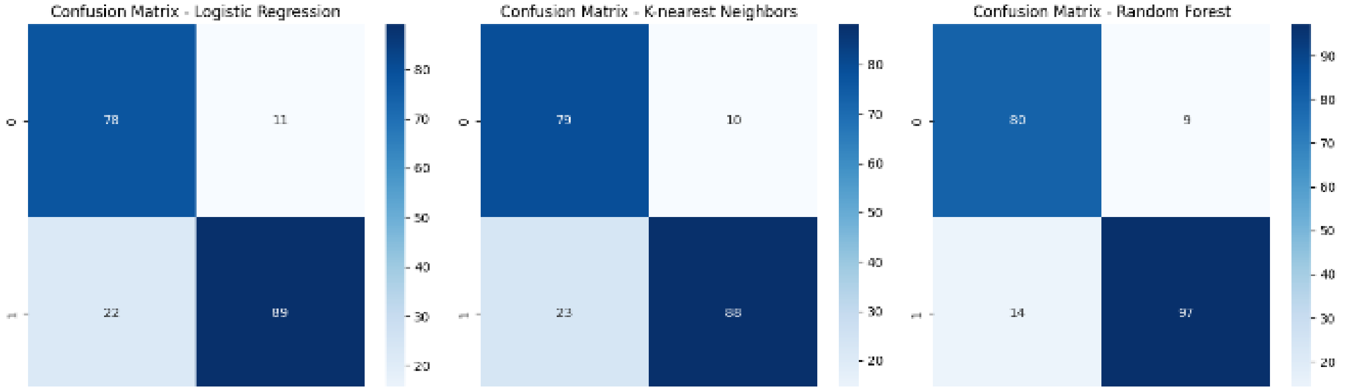


Fig. 12: Confusion matrices of the three classification algorithms applied on transformed dataset

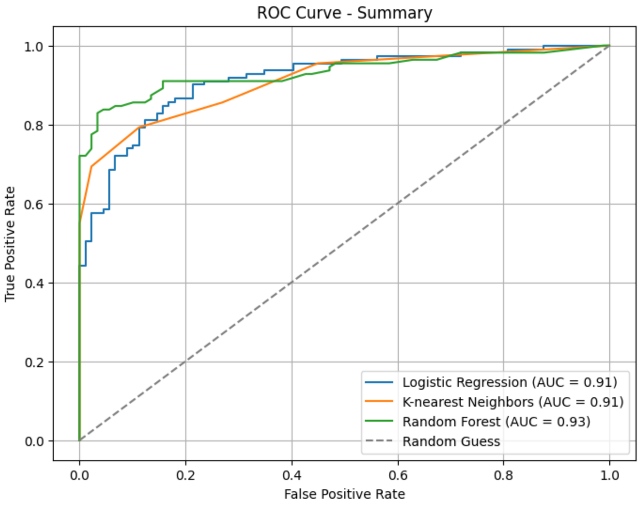


Fig. 13: ROC Curve LR

Furthermore, the ROC curve analysis, illustrated in Fig. 13 for the LR algorithm, provides insights into the model's classification performance across various thresholds. This curve, depicting the True Positive Rate against the False Positive Rate, serves as a complementary visualization to the confusion matrix. The ROC curve's depiction of Logistic Regression, K-Nearest Neighbor, and Random Forest's ability to predict three classes with 91%, 91%, and 93% accuracy aligns with the findings from the confusion matrix, affirming RF's superior classification capabilities.

Overall, these observations highlight the efficacy of LR, K-NN, and RF in successfully distinguishing between benign and classes, thereby underlining their potential for accurate classification tasks.

In the final phase of our analysis, we endeavored to elucidate the impact of the independent variables, namely the parental heights of the father and mother, on the outcomes predicted by our linear regression model. To achieve this objective, we incorporated the SHAP (Shapley Additive explanations) framework into our computational workflow. SHAP, a state-of-the-art method for interpreting complex machine learning models, facilitates the decomposition of model predictions to attribute them to individual features. Leveraging the SHAP. Explainer functionality, we conducted an in-depth examination of the linear regression model utilized in our study.

The resultant summary plot, driven by SHAP values, offered a comprehensive overview of the relative significance of the father and mother heights in influencing the model predictions. Through this visual representation, we gained valuable insights into the contribution of each feature towards the variability observed in the predicted outcomes. Moreover, by focusing on a specific sample indexed as 1 within the dataset, we enhanced our understanding of the feature importance dynamics in a more granular manner.

We harnessed the power of SHAP to construct a force plot, a graphical visualization tool designed to depict the individual impact of each component on the predictive model. By elucidating the marginal contributions of the father and mother heights to the model's predictions, the force plot provided a nuanced perspective on the interplay between the independent variables and the outcome variable.

In summary, the integration of SHAP into our analytical framework facilitated a comprehensive interpretation of the linear regression model, shedding light on the relative importance of parental heights in predicting offspring heights. Through visualizations such as the summary plot and force plot, we were able to unravel the intricate relationships between the input variables and the model predictions, thus enhancing the interpretability and transparency of our analysis.

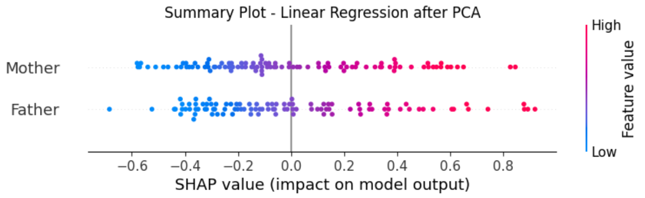


Fig. 14: Summary Plot

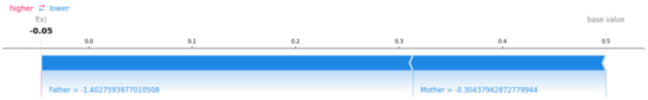


Fig. 15: Force Plot

VII. Conclusion

In conclusion, this project embarked on an insightful journey to explore the intricate relationship between parental heights and their influence on offspring heights, as observed in the Galton dataset. Through the lens of various statistical and machine learning techniques, we endeavored to unravel the underlying factors contributing to the growth of children's heights. Our exploration commenced with a thorough examination of familial attributes, including parental heights, child heights, and additional features such as gender and number of children. This comprehensive exploratory data analysis (EDA) laid the groundwork for subsequent analyses, providing valuable insights into the dataset's characteristics.

Leveraging Principal Component Analysis (PCA), we sought to reduce the dimensionality of the dataset while retaining essential information. PCA proved instrumental in unearthing hidden patterns and correlations among familial characteristics, particularly highlighting the pivotal role of parental heights in shaping offspring heights. With a focus on predictive modeling, we implemented various machine-learning algorithms, including Logistic Regression, K-Nearest Neighbors, and Linear Regression. These algorithms underwent a rigorous evaluation to ascertain their efficacy in capturing height inheritance dynamics and predicting child heights based on parental attributes.

Through hyper parameter tuning and meticulous model evaluation, we fine-tuned the classification models to optimize performance. Metrics such as accuracy, precision, recall, and F1-score served as benchmarks for assessing the effectiveness of the models. Additionally, ROC curve analysis provided insights into the models' classification capabilities across different thresholds, further enhancing our understanding of their performance.

In pursuit of interpretability, we integrated the SHAP (Shapley Additive Explanations) framework into our analytical workflow. This facilitated a deeper understanding of the linear regression model's predictions, elucidating the relative significance of parental heights in influencing offspring heights. Visualizations such as summary plots and force plots offered nuanced insights into feature importance dynamics, enhancing the transparency and interpretability of our analysis.

In summary, this project contributes to the broader discourse on height inheritance, shedding light on the multifaceted nature of familial attributes' influence on offspring heights. By leveraging advanced analytical techniques and machine learning algorithms, we gained valuable insights into height inheritance dynamics, paving the way for informed decision-making and future research endeavors in this domain.

REFERENCES

[1] https://colab.research.google.com/github/myconcordia/INSE6220/blob/main/Tutorial1.ipynb#scrollTo=JZRxI2pQP1DY
[2] https://www.analyticsvidhya.com/blog/2021/12/12-data-plot-types-for-visualization/
[3] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2986552/#:~:text=More%20than%20a%20100%20years,'1%20(Figure%201)
[4] https://datarepository.wolframcloud.com/resources/Galton-Parent-and-Child-Height-Data/
[5] https://matplotlib.org/stable/users/installing/index.html
[6] https://chat.openai.com/
[7] https://stackoverflow.com/