STA 135 Final Project
Diabetes in Female Pima Indians

Group Members
Alex Prado (914399483)
Laura Valenzuela (919202591)
Thomas Venner (916857208)
Vanessa Vu (918156114)

Abstract

In the Pima Indians, obesity and diabetes have become more common during this century. This may be a result of fast cultural and dietary changes in a population that is genetically susceptible to diabetes[1]. The goal of this report is to explore the link between health and health history factors, and the onset of diabetes in female Pima Indians. We leverage exploratory data analysis (EDA) techniques such as factor analysis, scatter plots, and histograms, to guide our application of multivariate data analysis techniques to create accurate predictions of the onset of diabetes. We consider generative classification techniques such as Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) for our modeling purposes, and evaluate our chosen model's performance using various metrics. Finally, we will use predictor permutations to determine predictor importance to our fitted model and answer the question of the most important predictors concerning the prediction of diabetes.

1 Introduction

Diabetes and obesity are major health concerns that are influenced by a combination of genetic and environmental factors. Due to the rapid changes in their environment over the century, the Pima Indians have become more genetically susceptible to diabetes as compared to individuals of other populations, causing them to become a subject of extensive research. Many researchers collaborated to construct the Pima Indians Diabetes Database. Therefore, our report revolves around determining the feasibility of predicting the likelihood of a female developing diabetes based on factors, such as blood pressure, BMI, age, etc. After preprocessing the dataset, we employed the following classification methods: QDA and factor analysis. Essentially, we aim to evaluate the accuracy of our predictive models and to identify the most significant risk factors that influence the development of diabetes.

2 Data Description

This data set has been taken from the UCI Repository of Machine Learning Databases. Originally from the National Institute of Diabetes and Digestive and Kidney Diseases, contains information from a Pima Indians population near Phoenix, Arizona, USA[3]. It contains 768 observations on 9 variables. All the patients are female and at least 21 years old of Pima Indian heritage[2]. After cleaning the data, the number of observations end up being 392. See Table 1 for variable information below.

Table 1: Data Description

Feature	Description	Data Types
Pregnant	Number of times pregnant	Numeric
Glucose	plasma glucose concentration	Numeric
Pressure	diastolic blood pressure (mm Hg)	Numeric
Triceps	triceps skin fold thickness (mm)	Numeric
Insulin	2-hour serum insulin (mu U/ml)	Numeric
Mass	Body Mass Index	Numeric
Pedigree	diabetes pedigree function	Numeric
Age	Age (years)	Numeric
Diabetes	2 levels 'pos' and 'neg'	Factor

3 Exploratory Data Analysis

After cleaning the data, a correlation plot matrix was created to conveniently view the class relationships between each combination of predictors, the distributions of predictors by class, and the correlations between predictors as a whole and by class. Finally, the bottom right corner plot shows that the classes are moderately imbalanced, favoring the non diabetic class. This visualization largely assisted in our search for class separability, which is a necessary component of prediction. We note that the variable 'glucose levels' appears most discriminative of the classes for all combinations with other predictors. Next, we will excavate the latent factors in our data set using factor analysis in order to better interpret the variability of our data.

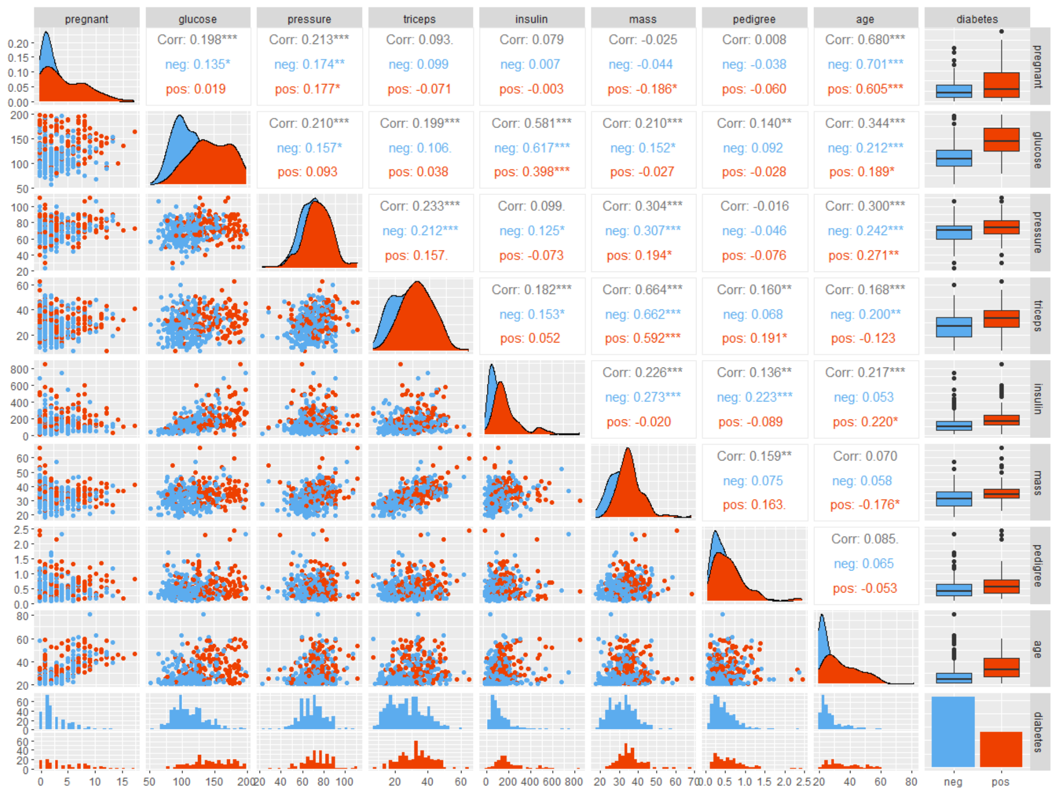
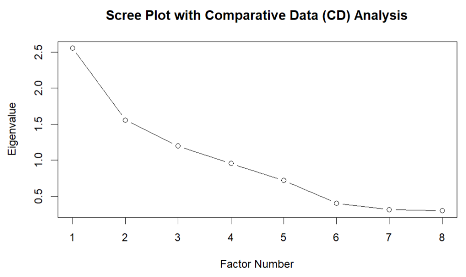


Figure 1: Diabetes Correlation Plot

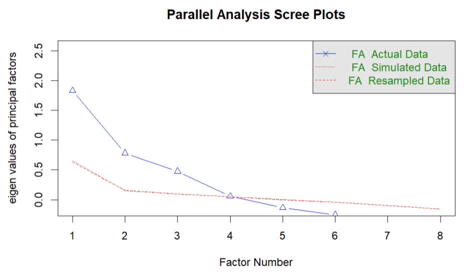
3.1 Factor Analysis

Factor analysis is a statistical technique used to explore the underlying structure of observed variables and identify latent factors that explain their correlations. It involves estimating factor loadings, which represent the strength and direction of the relationship between observed variables and latent factors. The covariance matrix of the observed variables is decomposed into two components: the factor loading matrix, denoted as $\mathbf{L}$, and the unique factor matrix, denoted as $\mathbf{\Psi}$. The factor analysis model is represented as $\mathbf{S} = \mathbf{L}\mathbf{L}' + \mathbf{\Psi}$, where $\mathbf{L}'$ is the transpose of $\mathbf{L}$.

Determining the appropriate number of factors in factor analysis can be done using Comparative Data (CD) analysis, which compares eigenvalues. Eigenvalues reflect the amount of variance explained by each factor. CD analysis involves comparing the eigenvalues from the actual dataset with those from randomly generated datasets with the same size and correlation structure. By examining the scree plot, which displays eigenvalues in descending order, we can identify the point where the eigenvalues of the actual dataset continue to surpass those of the random datasets. This point indicates the optimal number of factors to retain.



((a)) Scree Plot with CD Analysis



((b)) Parallel Analysis Scree Plot

Figure 2: Comparison of Scree Plots

The ideal number of factors based on Comparative Data (CD) Analysis was 3.

The Varimax rotation is a widely used method in factor analysis that simplifies the factor structure and enhances factor interpretability. It maximizes the variance of squared loadings within factors and minimizes cross-loadings across factors.

To perform Varimax rotation:

1. Calculate the factor loadings matrix, denoted as L, representing variable loadings on factors. 2. Center the loadings matrix by subtracting column means. 3. Initialize a rotation matrix, denoted as R, as an identity matrix. 4. Iterate until convergence: - Compute the transformed loadings matrix, denoted as L', by multiplying centered loadings L with rotation matrix R: $L' = LR$. - Apply an orthogonal transformation to L' to obtain a new rotation matrix R'. - Update R with R'.

The goal is to find an optimal R that results in loadings with high variance within factors and low variance across factors. Optimization involves minimizing an objective function, the sum of squared loadings variances within factors. Iterative algorithms like Kaiser's or singular value decomposition (SVD) are commonly used. Varimax rotation simplifies the factor structure, with variables tending to have high loadings on a single factor, aiding interpretation.

Factor loadings in factor analysis are calculated using the following:

$$\text{Loading}_{ij} = \sqrt{\frac{\text{Var}(\text{factor}_i)}{\text{Var}(\text{variable}_j)}} \times \text{corr}(\text{factor}_i, \text{variable}_j) \tag{1}$$

Hence, we obtain:

Typically, loadings above 0.3 or 0.4 are moderately significant, while loadings above 0.5 or 0.6 are considered stronger. However, these thresholds are not strict rules, but rather rough guidelines. Based on the factor loadings table in Table 2, the variables with high loadings for each factor are as follows:

Table 2: Factor Loadings

Variable	MR1	MR2	MR3
pregnant	0.756		
glucose	0.121	0.222	0.742
pressure	0.294	0.289	
triceps	0.661	0.104	0.136
insulin	0.115		0.746
mass	0.986		0.144
pedigree	0.144		0.162
age		0.885	0.201

Factor 1 (MR1): Variables with high loadings: Pregnant, Triceps, Mass Interpretation: This factor may represent a measure of overall body size or adiposity, as it includes variables related to pregnancy, triceps skinfold thickness, and body mass.

Factor 2 (MR2): Variables with high loadings: Age, Systolic Blood Pressure Interpretation: This factor may represent a measure of age along with blood pressure risk, as it includes variables related to glucose levels.

Factor 3 (MR3): Variables with high loadings: Glucose, Insulin Interpretation: This factor may represent a combination of variables related to glucose metabolism and insulin levels.

4 Classifier Models

Based on our EDA, we chose to use generative models to predict the onset of diabetes based on the predictors in the data set. We will use Box's M test to determine whether to use Linear Discriminant Analysis or Quadratic Discriminant Analysis for modeling.

4.1 Box's M Test

To determine whether to use LDA or QDA, we look to test the assumption that separates them. That is, we test whether $\Sigma_1 = \Sigma_{-1}$ where we encode the presence of diabetes with 1 and the absence of diabetes with -1 (the choice of -1 and 1 instead of 0 and 1 will become clear in the next section). We estimate Σ_j with S_j, the sample covariance matrix of class j with $j \in \{-1, 1\}$. We will use $i = 1, 2, \dots N$ to index each observation vector $\mathbf{x}_i$. Note that $\mathbf{x}_i \in \mathbb{R}^p$ where p is the number of predictors, and that N is the total number of observations considered (across classes). Before we define S_j, we define $\mathbf{x}_{ij}$

$$\bar{\mathbf{x}}_j = \frac{1}{n_j} \sum_{i=1}^{n_j} \mathbf{x}_{ij} \tag{2}$$

where $\mathbf{x}_{ij}$ is observation i belonging to class j with n_j being the total number of observations in class j such that $\sum_j n_j = N$. Additionally, we estimate $P(Y = j)$ by $\frac{n_j}{N}$, the proportion of the sample of class j, and Σ_j with S_j (the covariance matrix of class j), where

$$S_j = \frac{1}{n_j - 1} \sum_{i=1}^{n_j} (\mathbf{x}_{ij} - \bar{\mathbf{x}}_j)(\mathbf{x}_{ij} - \bar{\mathbf{x}}_j)^T \tag{3}$$

Finally, we define S_{pl} as

$$S_{pl} = \frac{(n_1 - 1)S_1 + (n_{-1} - 1)S_{-1}}{n_1 + n_{-1} - 2} \tag{4}$$

To conduct Box's M test, we first compute the M statistic, defined as

$$M = \left(\frac{|S_1|}{|S_{pl}|}\right)^{v_1/2} \left(\frac{|S_{-1}|}{|S_{pl}|}\right)^{v_2/2} \tag{5}$$

where $v_j = n_j - 1$ is the degrees of freedom corresponding to the covariance matrix of class j. To conclude the test, we convert our test statistic to a χ^2 statistic using Eq. 6

$$u = -2(1 - c_1) \ln M \tag{6}$$

where

$$u \sim \chi^2 \left[\frac{1}{2}(k-1)p(p+1)\right] \tag{7}$$

where k is the number of groups. c_1 is defined as

$$c_1 = \left(\sum_j \frac{1}{v_j} - \frac{1}{\sum_j v_j}\right) \left(\frac{2p^2 + 3p - 1}{6(p+1)(k-1)}\right) \tag{8}$$

Applying the Box's M test to our data, we get $\chi^2_{stat} = 164$ which follows χ^2_{36} under the null. Comparing to the critical value, $\chi^2_{crit} = 51$, since $\chi^2_{stat} > \chi^2_{crit}$, we reject the null hypothesis that $\Sigma_1 = \Sigma_{-1}$. Hence, we will use QDA for classification modeling, which does not assume $\Sigma_1 = \Sigma_{-1}$. We note that we manually calculated the result of the Box's M test without using R Box's M functions (using the equations show above) and confirmed our results using R's Box's M functions.

4.2 Quadratic Discriminant Analysis

Given the result of the Box's M test, we will use QDA because there was significant evidence in the Box's M test to reject the null that $\Sigma_1 = \Sigma_{-1}$ and QDA does not assume $\Sigma_1 = \Sigma_{-1}$ unlike LDA. We can calculate the decision boundary of the classifier starting with Bayes optimal decision criteria for two classes in Eq. 9, which assigns an observation to the class which corresponds to the highest posterior probability given the observed vector $\mathbf{x}_i$. We encode the event that a subject has diabetes as $Y = 1$, and we encode the event that a subject does not have diabetes as $Y = -1$.

$$\hat{y}_i = \operatorname{sign} \log \left(\frac{P(Y = 1|\mathbf{x}_i)}{P(Y = -1|\mathbf{x}_i)}\right) \tag{9}$$

Due to the definition of conditioning (Eq. 10), Eq. 9 can be expressed equivalently in Eq. 11,

$$P(Y|X) = \frac{P(Y, X)}{P(X)} \tag{10}$$

$$\hat{y}_i = \operatorname{sign} \log \left(\frac{P(X = \mathbf{x}_i|Y = 1)P(Y = 1)}{P(X = \mathbf{x}_i|Y = -1)P(Y = -1)}\right) \tag{11}$$

Since the value of Eq. 11 is determined by the sign of the log ratio, the decision boundary is found by equating the log ratio to 0 (see Eq. 12).

$$0 \overset{!}{=} \log \left(\frac{P(X = \mathbf{x}_i|Y = 1)P(Y = 1)}{P(X = \mathbf{x}_i|Y = -1)P(Y = -1)}\right) \tag{12}$$

Using the generative assumptions of QDA, we have

$$P(X = \mathbf{x}_i|Y = 1) \sim N_p(\boldsymbol{\mu}_1, \Sigma_1) \tag{13}$$

$$P(X = \mathbf{x}_i | Y = -1) \sim N_p(\boldsymbol{\mu}_{-1}, \Sigma_{-1}) \tag{14}$$

where $\Sigma_1 \neq \Sigma_{-1}$. Substituting the pdfs of the multivariate normal distributions of the respective classes into the decision boundary equation we have

$$0 = \log\left(\frac{P(Y=1)}{P(Y=-1)}\right) + \log\left(\frac{|\Sigma_1|}{|\Sigma_{-1}|}\right) - \frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_1)^T \Sigma_1^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_1) + \frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_{-1})^T \Sigma_{-1}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_{-1}) \tag{15}$$

Using the fact that

$$-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_j)^T \Sigma^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_j) = \mathbf{x}_i^T \Sigma^{-1} \boldsymbol{\mu}_j - \frac{1}{2}\boldsymbol{\mu}_j^T \Sigma^{-1} \boldsymbol{\mu}_j - \frac{1}{2}\mathbf{x}_i^T \Sigma^{-1} \mathbf{x}_i \tag{16}$$

we have

$$\begin{aligned} 0 = \log\left(\frac{P(Y=1)}{P(Y=-1)}\right) + \log\left(\frac{|\Sigma_1|}{|\Sigma_{-1}|}\right) + \mathbf{x}_i^T \Sigma_1^{-1} \boldsymbol{\mu}_1 - \frac{1}{2}\boldsymbol{\mu}_1^T \Sigma_1^{-1} \boldsymbol{\mu}_1 - \frac{1}{2}\mathbf{x}_i^T \Sigma_1^{-1} \mathbf{x}_i \\ - \mathbf{x}_i^T \Sigma_{-1}^{-1} \boldsymbol{\mu}_{-1} + \frac{1}{2}\boldsymbol{\mu}_{-1}^T \Sigma_{-1}^{-1} \boldsymbol{\mu}_{-1} + \frac{1}{2}\mathbf{x}_i^T \Sigma_{-1}^{-1} \mathbf{x}_i \end{aligned} \tag{17}$$

Rewriting Eq. 17 in the characteristic quadratic form, we have

$$\begin{aligned} 0 = \log\left(\frac{P(Y=1)}{P(Y=-1)}\right) + \log\left(\frac{|\Sigma_1|}{|\Sigma_{-1}|}\right) - \frac{1}{2}\boldsymbol{\mu}_1^T \Sigma_1^{-1} \boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_{-1}^T \Sigma_{-1}^{-1} \boldsymbol{\mu}_{-1} + \mathbf{x}_i^T (\Sigma_1^{-1} \boldsymbol{\mu}_1 - \Sigma_{-1}^{-1} \boldsymbol{\mu}_{-1}) \\ + \mathbf{x}_i^T (-\frac{1}{2}\Sigma_1^{-1} + \frac{1}{2}\Sigma_{-1}^{-1}) \mathbf{x}_i \end{aligned} \tag{18}$$

$$0 = \beta_0 + \mathbf{x}_i^T \boldsymbol{\beta}_1 + \mathbf{x}_i^T A \mathbf{x}_i \tag{19}$$

where $\beta_0 = \log\left(\frac{P(Y=1)}{P(Y=-1)}\right) + \log\left(\frac{|\Sigma_1|}{|\Sigma_{-1}|}\right) - \frac{1}{2}\boldsymbol{\mu}_1^T \Sigma_1^{-1} \boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_{-1}^T \Sigma_{-1}^{-1} \boldsymbol{\mu}_{-1}$, $\boldsymbol{\beta}_1 = \Sigma_1^{-1} \boldsymbol{\mu}_1 - \Sigma_{-1}^{-1} \boldsymbol{\mu}_{-1}$, and $A = -\frac{1}{2}\Sigma_1^{-1} + \frac{1}{2}\Sigma_{-1}^{-1}$. In other words, Eq. 18 is the QDA decision boundary. Alternatively, one could express the QDA decision as

$$\arg\max_j \delta_j(\mathbf{x}_i) \tag{20}$$

where $\delta_j(\mathbf{x}_i)$ is the quadratic score function for class j given by

$$\delta_j(\mathbf{x}_i) = -\frac{1}{2}\log(|\Sigma_j|) - \frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_j) + \log(P(Y = j)) \tag{21}$$

For statistical purposes, we estimate $\boldsymbol{\mu}_j$ with $\bar{\mathbf{x}}_j$. Additionally, we estimate $P(Y = j)$ by $\frac{n_j}{N}$, the proportion of the sample of class j, and Σ_j with S_j. As an aside, we manually calculated the QDA decision boundary in R using the equations above, which is omitted from the report for brevity (see coding files).

5 Results and Interpretation

5.1 QDA Performance

After fitting QDA and calculating the decision boundary as shown above (see code for manual decision boundary calculation), we estimate the generalization error of the QDA model using 10 fold

Table 3: QDA Confusion Matrix CV Rates

	TP	TN
PP	20.2%	9.60%
PN	12.9%	57.2%

cross validation (CV), repeated 1000 times.The confusion matrix shown below in Table 3 reflects the results of this procedure.

From the confusion table diagonal, we see that the model did relatively well in classifying subjects as diabetic or non diabetic, with the difference in true positive rates and true negative rates mostly coming from the fact that the data had more cases of non diabetic than diabetic cases. Additionally, we see that the false negative rate is slightly higher (about 1.25 times) the false positive rate, revealing that the model misclassifies diabetic subjects as non diabetic subjects marginally more than it misclassifies non diabetic subjects as diabetic subjects. Additionally, we review other model CV based performance metrics in Table 4.

Table 4: CV Performance Metrics

Metric	Value
Accuracy	0.77
95% CI for Accuracy	(0.77, 0.78)
Sensitivity	0.61
Specificity	0.86
Pos Pred Value	0.68
Neg Pred Value	0.82
Prevalence	0.33
Detection Rate	0.20
Detection Prevalence	0.30
Balanced Accuracy	0.73

We note that the generalization error is just below 80% indicating a respectable model performance. Since the classes are not excessively imbalanced, we regard accuracy as valid measure of the generalization error of our fitted QDA model. We also note that the 95% confidence interval for the accuracy is very small, which indicates that our CV accuracy estimate is a good one. We see that our model specificity is modestly higher than our model sensitivity, indicating that our model is better at correctly classifying non diabetic subjects than correctly classifying diabetic subjects. Additionally, we see that our negative predictive value is modestly larger than our positive predictive value, indicating that a negative prediction (no diabetes) by our model is more likely to be correct than a positive prediction (diabetes present). Finally, noting the moderate class imbalance in our dataset, we place higher importance on the balanced accuracy rate of our model at 73%.

5.2 QDA Predictor Importances using Predictor Permutations

Predictor importance to the QDA model can be approximated using predictor permutations. For each predictor, the predictor column is permuted, essentially removing the information from the predictor on the response. Then the model performance is again calculated, and the difference between the model performance with and without the predictor being permuted is regarded as the predictor importance.

Table 5: QDA Predictor Importances

Predictor	Importance
Glucose	0.12
Pregnancies	0.083
Age	0.079
Insulin	0.039
Skin Thickness	0.016
Diabetes Pedigree	0.016
Blood Pressure	0.005
BMI	0.002

We note that glucose has the largest predictor importance, followed by pregnancies and age. We recall that glucose was the most discriminative factor upon visual inspection in the scatter plot EDA, so the fact that it is the most important predictor is no surprise. We also note that the predictor 'age' dominated the second factor in factor analysis, which corroborates the result here that 'age' is among the most important predictors of diabetes. Additionally, we recall that pregnancies was given a high loading in factor one, signifying it's importance, and thus this fact is also corroborated by pregnancies being high in permutation importance.

6 Conclusion

The Pima Indians started to show a rapid increase in diabetes in the last decades, and our report set out to try to predict this trend. From our exploratory data analysis, we saw that glucose seemed to be the most discriminative predictor for all combinations of itself with other predictors. After determining the optimal number of factor for factor analysis, we uncovered 3 factors which best approximated the variance in our predictors. These factors are glucose metabolism, age, and pregnancy and adiposity. Next, we conducted the Box's M test, and due to rejection of the null that $\Sigma_{-1} = \Sigma_1$, we chose to use Quadratic Discriminant Analysis for predictive modeling. Our QDA fit achieved a balanced accuracy level of 73% based on generalization error estimates. Using permutation predictor importances, we note that glucose was rated the most important predictor, followed by pregnancies and age in predicting the onset of diabetes.

7 References

1. Knowler WC, Pettitt DJ, Savage PJ, Bennett PH. Diabetes incidence in Pima Indians: contributions of obesity and parental diabetes. Am J Epidemiol. 1981 Feb;113(2):144-56. doi: 10.1093/oxfordjournals.aje.a113079. PMID: 7468572.
2. UCI Machine Learning. "Pima Indians Diabetes Database." Kaggle, 2016, `www.kaggle.com/datasets/uciml/pima-indians-diabetes-database`.
3. npradaschnor. "Pima Indians Diabetes Dataset." GitHub, 20 Feb. 2021, `www.github.com/npradaschnor/Pima-Indians-Diabetes-Dataset`.
4. Tabachnick, B. G., & Fidell, L. S. (2007). Using Multivariate Statistics (5th ed.). Boston, MA: Pearson Education.
5. Hayton, J. C., Allen, D. G., & Scarpello, V. (2004). Factor Retention Decisions in Exploratory Factor Analysis: A Tutorial on Parallel Analysis. Organizational Research Methods, 7(2), 191-205. doi:10.1177/1094428104263675
6. Rencher, Alvin C. Methods of Multivariate Analysis (1st ed.). Wiley, 1995.

8 Contribution

Alex Prado: Section 3.1, 4
Laura Valenzuela: Section 2, 3, 4.1
Thomas Venner: Section 4, 5.1, 6
Vanessa Vu: Section 1, 4.1, 4.2

9 Appendix

9.1 Calculating Permutation Importance

```
[
frame=lines,
framesep=2mm,
baselinestretch=1.2,
bgcolor=LightGray,
fontsize=\footnotesize,
linenos
]{python}
####################################
# pythong code
import pandas as pd

# Load the dataset
df = pd.read_csv('diabetes.csv')

# Print the column names
column_names = df.columns
print("Column Names:", column_names)

import numpy as np
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.inspection import permutation_importance

# Load the dataset
df = pd.read_csv('diabetes.csv')

# Drop rows with NaN values and zeroes (except for diabetes column)
df = df.dropna()  # Drop rows with NaN values
df = df[(df.iloc[:, :-1] != 0).all(axis=1)]
# Drop rows with zeroes in all columns except for the outcome

# Separate the predictor variables and the target variable
X = df.drop('Outcome', axis=1)
y = df['Outcome']

# Perform Quadratic Discriminant Analysis
qda_model = QuadraticDiscriminantAnalysis()
qda_model.fit(X, y)

# Calculate permutation importance
importance = permutation_importance(qda_model, X, y, n_repeats=100, random_state=42)

# Get the importance scores and feature names
importance_scores = importance.importances_mean
feature_names = X.columns

# Create a DataFrame of feature importance
importance_df = pd.DataFrame({'Feature': feature_names, 'Importance': importance_scores})

# Sort the DataFrame by importance scores in descending order
importance_df = importance_df.sort_values('Importance', ascending=False)

# Print the ranked predictors and their importance scores
for i, row in importance_df.iterrows():
    print(f"Predictor: {row['Feature']} | Importance: {row['Importance']}")
```

```
# Print the total number of rows before and after cleaning
total_rows_before = len(df)
total_rows_after = len(X)
print(f"Total rows before cleaning: {total_rows_before}")
print(f"Total rows after cleaning: {total_rows_after}")
```

9.2 Data Cleaning, Outlier Removal, And Factor Analysis

```
#####################################

# install.packages("HSAUR2")  # Install HSAUR2 package if not installed
# install.packages("dplyr")   # Install dplyr package if not installed
# install.packages("ggplot2") # Install ggplot2 package if not installed
# install.packages("rlang")
# install.packages("psych")

library(HSAUR2)
library(dplyr)
# library(ggplot2)
library(mlbench)
library(dplyr)
data(PimaIndiansDiabetes)

total_rows_before <- nrow(PimaIndiansDiabetes)  # Total rows before cleaning
total_rows_before # 768
head(PimaIndiansDiabetes)  # Show the first few rows of the dataset
summary(PimaIndiansDiabetes)  # Provide summary statistics of the dataset
str(PimaIndiansDiabetes)  # Display the structure of the dataset


# Handling missing values
Pima_clean <- PimaIndiansDiabetes


# Remove rows with NaN or zero values
Pima_clean <- PimaIndiansDiabetes %>%
  filter(!is.nan(glucose) & glucose != 0 &
         !is.nan(pressure) & pressure != 0 &
         !is.nan(triceps) & triceps != 0 &
         !is.nan(insulin) & insulin != 0 &
         !is.nan(mass) & mass != 0)



# Print the total rows before and after cleaning
total_rows_before <- nrow(PimaIndiansDiabetes)  # Total rows before cleaning
total_rows_after <- nrow(Pima_clean)  # Total rows after cleaning
cat("Total rows before cleaning:", total_rows_before, "\n")
cat("Total rows after cleaning:", total_rows_after, "\n")


library(psych)
library(nFactors)

# Display the number of rows before removing missing or non-finite values
cat("Number of rows before removing values:", nrow(Pima_clean), "\n")

# Remove rows with missing or non-finite values
```

```
Pima_clean <- na.omit(Pima_clean)
Pima_clean <- Pima_clean[is.finite(rowSums(Pima_clean)), ]

# Display the number of rows after removing values
cat("Number of rows after removing values:", nrow(Pima_clean), "\n")

# Perform factor analysis on the cleaned dataset
factor_analysis <- fa(Pima_clean, nfactors = ncol(Pima_clean), rotate = "varimax")

# Perform CD analysis to determine the ideal number of factors
cd_factors <- fa.parallel(Pima_clean, fm = "ml", fa = "fa", n.iter = 100)

# Plot scree plot with CD analysis results
plot(cd_factors$eigen$qev, type = "b", xlab = "Factor Number", ylab = "Eigenvalue",
     main = "Scree Plot with Comparative Data (CD) Analysis")

# Identify the ideal number of factors based on CD analysis
ideal_factors <- cd_factors$nfact

# Print the ideal number of factors
cat("Ideal number of factors based on Comparative Data (CD) Analysis:", ideal_factors, "\n")

# Perform CD analysis to determine the ideal number of factors
cd_factors <- fa.parallel(Pima_clean, fm = "ml", fa = "fa", n.iter = 100)

# Calculate eigenvalues manually
eigenvals <- eigen(cor(Pima_clean))$values

# Plot scree plot with eigenvalues
plot(1:length(eigenvals), eigenvals, type = "b", xlab = "Factor Number", ylab = "Eigenvalue",
     main = "Scree Plot with Comparative Data (CD) Analysis")

# Identify the ideal number of factors based on CD analysis
ideal_factors <- cd_factors$nfact

# Print the ideal number of factors
cat("Ideal number of factors based on Comparative Data (CD) Analysis:", ideal_factors, "\n")

# Perform factor analysis on the cleaned dataset
factor_analysis <- fa(Pima_clean, nfactors = 3, rotate = "varimax")

# Display factor analysis results
print(factor_analysis)

# Extract factor loadings
factor_loadings <- factor_analysis$loadings
print(factor_loadings)

# Extract communalities
communalities <- factor_analysis$communality
print(communalities)



####################################

install.packages('mlbench')
install.packages("mvoutlier")
install.packages("heplots")
```

~~~
```{r}
library(mlbench)
library(ggplot2)
library(psych)
library(mvoutlier)
library(heplots)
library(MASS)
library(caret)
library(klaR)
```
\begin{verbatim}
## Data Cleaning
```{r}
data(PimaIndiansDiabetes)
raw_data <- PimaIndiansDiabetes
summary(raw_data)
str(raw_data)
```

```{r}
# pos=1 and neg=0
raw_data$diabetes <- ifelse(raw_data$diabetes == 'pos', 1, 0)
# remove 0 from glucose, pressure, triceps, insulin, and mass
cols <- c('glucose', 'pressure', 'triceps', 'insulin', 'mass')
# iterate
for (col in cols){
  raw_data = raw_data[raw_data[[col]] != 0,]
}
# check for nas
data <- na.omit(raw_data)
head(data,5)

plot = ggpairs(data, aes(color = diabetes))
plot + scale_fill_manual(values =
c("Steelblue2", "orangered2")) + scale_color_manual(values = c("Steelblue2", "orangered2"))

```
~~~

9.3 Box's M Test

````
#Box's M test by hand

#turn 'pos' and 'neg' into 1 and -1
data$diabetes <- ifelse(data$diabetes == 'pos', 1, -1)

data1 = data

diabetes_neg <- subset(data1, diabetes == "-1")
diabetes_pos <- subset(data1, diabetes == "1")

k = 2

n1 <- dim(diabetes_neg)[1]
n2 <- dim(diabetes_pos)[1]

p = 8

v1 = n1 - 1
````

~~~
v2 = n2 - 1

S1 <- cov(diabetes_neg[, -ncol(diabetes_neg)])
S2 <- cov(diabetes_pos[, -ncol(diabetes_pos)])


Spl <- ((v1*S1) + (v2*S2))/(n1 + n2 - 2)
Spl

ln_M = 1/2*(v1*log(det(S1)) + v2*log(det(S2))) - (1/2*(v1+v2) * log(det(Spl)))
ln_M

c1 = ((1/v1 + 1/v2 - 1/(v1+v2)) *(2*p^2 + 3*p - 1)) / (6*(p+1)*(k -1))
c1

u = -2*(1 - c1)*ln_M
u

qchisq(0.95, 1/2*(k -1)*p*(p+1))


#We reject the null

## Box M's

```{r}
# https://www.rdocumentation.org/packages/heplots/versions/1.4-2/topics/boxM
predictors <- data[, 1:8]  # subset predictors
response <- data$diabetes # subset response variable
res <- boxM(predictors, response)
res
summary(res) # qda needed
```
Box's M-test for Homogeneity of Covariance Matrices

data:  predictors
Chi-Sq (approx.) = 164.12, df = 36, p-value < 2.2e-16

Summary for Box's M-test of Equality of Covariance Matrices

Chi-Sq:  164.115
df:  36
p-value: < 2.2e-16

log of Covariance determinants:
       0        1   pooled
30.87231 34.22850 32.41413

Eigenvalues:
             0            1       pooled
1 1.078036e+04 1.776328e+04 1.307555e+04
2 3.762490e+02 7.489258e+02 5.097229e+02
3 1.718975e+02 1.877078e+02 1.750828e+02
4 1.019106e+02 1.315215e+02 1.089349e+02
5 7.117669e+01 7.266615e+01 8.046340e+01
6 1.863463e+01 2.328887e+01 2.040720e+01
7 3.284752e+00 8.601793e+00 5.164432e+00
8 8.258758e-02 1.533693e-01 1.108329e-01
~~~

```
Statistics based on eigenvalues:
                     0            1       pooled
product   2.556663e+13 7.332446e+14 1.194751e+14
sum       1.152359e+04 1.893615e+04 1.397544e+04
precision 8.000711e-02 1.490871e-01 1.075895e-01
max       1.078036e+04 1.776328e+04 1.307555e+04
```

9.4 QDA

~~~
## QDA

```{r}
# fit QDA model
# https://www.statology.org/quadratic-discriminant-analysis-in-r/
set.seed(37)
# defining test and training data set
sample <- sample(c(TRUE, FALSE), nrow(data), replace = TRUE, prob = c(0.7, 0.3))
train <- data[sample, ]
test <- data[!sample, ]
# subset train and test predictor and response variables
train_predictors <- train[, 1:8]
train_response <- train$diabetes
test_predictors <- test[, 1:8]
test_response <- test$diabetes

# fit QDA model
qda_model <- qda(train_predictors, train_response)
qda_model
```
Call:
qda(train_predictors, train_response)

Prior probabilities of groups:
        0         1
0.6689895 0.3310105

Group means:
  pregnant  glucose pressure  triceps  insulin     mass  pedigree      age
0 2.703125 112.3177 69.43750 27.45833 128.6094 31.43802 0.4527396 28.06250
1 4.568421 142.7474 74.01053 33.88421 203.2105 36.24421 0.6409158 35.61053

## QDA Decision Boundary

```{r}
# beta0
# subset data based on class
diabetes_neg <- subset(data, diabetes == 0)
diabetes_pos <- subset(data, diabetes == 1)

# compute class probabilities
p_neg <- nrow(diabetes_neg) / nrow(data)
p_pos <- nrow(diabetes_pos) / nrow(data)

# compute covariance matrices for each class subset
cov_neg <- cov(diabetes_neg[, -ncol(diabetes_neg)])
cov_pos <- cov(diabetes_pos[, -ncol(diabetes_pos)])
~~~

~~~
# compute means for each class subset
mean_neg <- colMeans(diabetes_neg[, -ncol(diabetes_neg)])
mean_pos <- colMeans(diabetes_pos[, -ncol(diabetes_pos)])

# compute beta0
inv_cov_neg <- solve(cov_neg)
inv_cov_pos <- solve(cov_pos)

beta0 <- log(p_pos/p_neg) + log(det(cov_pos)/det(cov_neg)) - 0.5*t(mean_pos)%*%inv_cov_pos%*%mean
0.5*t(mean_neg) %*% inv_cov_neg %*% mean_neg
beta0
```
[1,] -2.419751

```{r}
# beta1
beta1 <- solve(cov_pos)%*%mean_pos - solve(cov_neg)%*%mean_neg
beta1
```
                 [,1]
pregnant  0.535368416
glucose  -0.076703398
pressure -0.071036495
triceps   0.118485374
insulin   0.037707267
mass      0.117657366
pedigree -2.723463014
age       0.005885856

```{r}
# A
A <- -0.5*solve(cov_pos) + 0.5*solve(cov_neg)
A
```

             pregnant       glucose      pressure       triceps       insulin          mass
pregnant  0.0929807448 -3.532717e-04 -7.192773e-04  1.095946e-03 -2.504351e-04  1.978042e-03  0.0
glucose  -0.0003532717  7.282534e-04  3.396059e-05  1.270795e-04 -1.538872e-04 -2.496136e-05  0.0
pressure -0.0007192773  3.396059e-05  5.900994e-04  7.212266e-04 -7.718575e-05 -8.691131e-04 -0.0
triceps   0.0010959463  1.270795e-04  7.212266e-04 -4.689142e-05  8.770951e-05 -1.981767e-03  0.0
insulin  -0.0002504351 -1.538872e-04 -7.718575e-05  8.770951e-05  4.853780e-05 -2.543007e-04 -0.0
mass      0.0019780415 -2.496136e-05 -8.691131e-04 -1.981767e-03 -2.543007e-04  3.859875e-03  0.0
pedigree  0.0818751359  7.148955e-03 -3.122927e-04  1.874153e-02 -6.133926e-03  1.606863e-02  2.7
age      -0.0174409614 -4.619574e-04  3.994559e-04 -2.491952e-03  2.427152e-04 -5.135866e-05 -0.0
~~~

9.5 QDA CV Metrics

```
library(mlbench)
data(PimaIndiansDiabetes2)

#perform some cleaning
data <- PimaIndiansDiabetes2
data <- na.omit(data)

#convert diabetes to 1 and 0 for confusion matrix purposes
data$diabetes <- as.factor(ifelse(data$diabetes == 'pos', 1, 0))

#number of CV reps is reps * 10, so do this for 1000 CV reps
reps = 100
```

```
#preallocate datalist vector for later confusion matrix aggregate computation
datalist = vector("list", length = reps)

#compute CV metrics for specified number of reps
for (i in 1:reps) {
folds <- sample(rep(1:10, length.out = nrow(data)), size = nrow(data), replace = F)

CV_qda <- lapply(1:10, function(x){
  model <- qda(diabetes ~ ., data[folds != x, ])
  preds <- predict(model,  data[folds == x,], type="response")$class
  return(data.frame(preds, real = data$diabetes[folds == x]))
})
datalist[[i]] <- do.call(rbind, CV_qda)
}

#aggregate results in one large matrix
library(caret)
big_data <- do.call(rbind, datalist)

#output results
confusionMatrix(big_data$preds, big_data$real, positive = '1')
```