Big Data Analysis for Diabetes on BRFSS Dataset

1st Amirreza Sokhankhosh
Computer Science
University of Manitoba
Winnipeg, Canada
sokhanka@myumanitoba.ca

2nd Sahar Lamey
Computer Science
University of Manitoba
Winnipeg, Canada
rahimzas@myumanitoba.ca

***Abstract*—Nowadays, with the rising trend of using computer systems, substantial amounts of data are collected and stored every day. These datasets often include rewarding knowledge and information that can assist businesses, researchers, and policymakers in their decision-making process. The Behavioral Risk Factor Surveillance System (BRFSS) is an organization that collects health-related data from U.S. residents via telephone surveys. In this paper, we perform data science and data mining techniques on the BRFSS data to unravel non-trivial information about diabetes disease. First, we study the most probable indicators of diabetes using a vertical frequent pattern mining algorithm named VIPER. In addition, to provide the public with a prediction tool for this disease, a Deep Learning Model is trained on the BRFSS dataset that can classify diabetic patients. To the best of our knowledge, our classifier outperforms all the existing works available on the internet for this dataset. At last, we study the possibility of using matrix decomposition for knowledge extraction in big data from a theoretical point.**

***Index Terms*—BRFSS Data, Diabetes, DNN, VIPER**

I. Introduction

The rising trend of technology and its vast applications have allowed the production and storage of countless records of information around the globe. At the moment of writing this paper, Google is queried 99,000 times in a second, which is only a small demonstration of the abundant production of data in this era [1]. The data created at every second holds valuable information that is not known at first sight and requires data processing and data mining techniques to be disclosed. The Behavioral Risk Factor Surveillance System is an organization that collects health-related information from U.S. residents via telephone surveys [2]. Such information includes having or not having high blood pressure, high cholesterol, diabetes, heart disease, and many other health-related attributes. This data is gathered annually and is made available for public use.

In this paper, we primarily aim to extract underlying knowledge about diabetes disorder while utilizing the BRFSS dataset. our primary motivations are providing the healthcare community with some beneficial information about diabetes as well as raising public awareness about this illness. To be specific, we aim to answer questions such as:

- What factors are the most predictive of diabetes disease?
- Using the BRFSS dataset, can we build a classification model that predicts the probability of having diabetes for new patients?
- Which age groups are most susceptible to being diagnosed with diabetes?
- Is there a difference between the distribution of diabetic patients in males or females?

To answer such questions, we first apply the VIPER algorithm [3] to a preprocessed version of the dataset available on the Kaggle website [4] to extract frequent patterns, containing *diabetes=1*, in the dataset. We define frequent patterns as those with a higher percentage of diabetic patients than the overall population. This helps us understand the truly vulnerable populations in the dataset. The results of the VIPER algorithm illustrate that people who are affiliated with BMI greater than 30, are more than 60 years old, diagnosed with heart disease, high blood pressure, or high cholesterol are more prone to diabetes. As mentioned before, we reach this conclusion by considering the percentage of people suffering from diabetes in each group. In the Results Section, we demonstrate the outcome of the VIPER algorithm in further detail.

Moreover, we build a Deep Neural Network (DNN) classifier that is trained to predict diabetic patients. Due to the imbalanced nature of the illness–only 13.9% of the dataset have diabetes-we incorporate the Synthetic Minority Oversampling Technique (SMOTE) to train the DNN properly. To the best of our knowledge, the resulting model outperforms all the previous works on the Kaggle website [5].

At last, we study matrix decomposition from a theoretical point. Big data, such as our own, are usually organized in matrix forms and are generally redundant and noisy. Therefore, matrix decomposition becomes one of the fundamental tools to attack the collected data. Matrix decomposition has been extensively studied due to its effectiveness and is still an active topic today. The conceptual idea of matrix decomposition is that the primitive big and noisy data matrix can be approximated by the product of two or more compact low-rank matrices. This theoretical study is the contribution of the second author and is discussed at the end of the paper. On the other hand, the utilization of the VIPER algorithm and the DNN model is the work of the first author.

II. Related Work

Advances in the utilization of technology in healthcare settings have enabled both the collection of electronic medical records and the construction of big data analysis [6, 7]. Among all the applications of big data science and machine learning, we focus on diabetes mellitus (DM) disorder. Since there is no definitive cure for this illness, early diagnosis and

ID	Male	Female	Diabetes
0	1	0	1
1	1	0	1
2	0	1	1
3	0	1	1
4	0	1	1
5	0	1	0
6	0	1	0
7	0	1	0

TABLE I: An example dataset that illustrates the problem of using minimum support in VIPER.

blood glucose awareness are crucial for people diagnosed with DM [8]. Consequently, many studies have incorporated data science and data mining strategies to predict blood glucose levels or the probability of being affiliated with this metabolic disorder [8, 9].

As mentioned, several works have focused on predicting the value of blood glucose by utilizing distinct machine learning models such as Artificial Neural networks, both Feed Forward [10, 11] and Recurrent Neural Networks [12], Support Vector Machines [13], Random Forests [14], and Genetic Programming methods [15].

Other than predicting the blood glucose level of diabetic individuals, which is a regression task, some machine learning applications focus on classifying patients to facilitate the early diagnosis of diabetes. El-Sappagh et al [9] utilize distinct methods such as logistic regressions, k-nearest neighbors, naive Bayes, and more to properly classify patients suffering from DM.

As previously mentioned, we study a preprocessed version of the BRFSS dataset available on the Kaggle website [4]. At the time of writing this paper, 102 codes are available on the Kaggle website for various data science studies on this dataset on Kaggle. To get the best models overall, we sort the studies based on their upvotes and select the top ones for comparison. Sobuj [16] used the majority of the favored classifiers in machine learning including SVM, Random Forests, and more to predict diabetic patients, however, due to the imbalanced nature of the data, the models did not achieve satisfactory precisions. In addition, Elgendy [17] uses ML models such as XGBoost and Catboost along with an over-sampling procedure that greatly facilitates the process of learning the diabetic class. Nevertheless, deep learning models are not considered in this study. Teboul [18] employs a simple neural network and undersamples the dataset to balance learning. This immensely assists the training procedure and outperforms simple machine learning models which do not use over or undersampling. In this work, we propose using a Deep Neural Network and controlling the imbalanced data using SMOTE, an over-sampling method for managing imbalanced data.

III. Preprocessing

The Behavioral Risk Factor Surveillance System has been gathering data annually via telephone surveys since 1988. It is noteworthy that this information is not preprocessed for machine learning models before publication. In this paper, for the sake of simplicity, we use a slightly cleaner version of this data from 2015 available on the Kaggle website [4]. This dataset includes 22 categorical attributes, 19 of which might have a possible correlation with diabetes disease. Since the VIPER algorithm takes bit vectors as input, we encode all the attributes using a *One Hot Encoder*. To illustrate the procedure of one hot encoding consider the following example. Suppose that the BMI column only takes four categorical values of $\{$*Below 18.5*, *18.5 - 24.9*, *25 - 29.9*, *Above 30*$\}$. By *One Hot Encoding*, we convert the BMI column into 4 columns: *BMI-0*, *BMI-1*, *BMI-2*, and *BMI-3* each representing BMIs *Below 18.5*, *18.5 - 24.9*, *25 - 29.9*, and *Above 30*, respectively. In this manner, if the BMI value of a record is 27, then the value of all the mentioned columns will be 0, except *BMI-2* which will be 1. After one hot encoding, our dataset is ready to be fed into the VIPER algorithm as well as our Deep Neural Network (DNN).

Algorithm 1 VIPER algorithm - computing frequent patterns

```
Require: data, $\alpha x, \alpha y$
    $minconf \leftarrow data[''diabetes''].sum()/len(data)$
    $N_d \leftarrow data[\alpha\mathrm{x}] \cdot data[\alpha\mathrm{y}]$
    $N_a \leftarrow len(data)$
    $conf \leftarrow \frac{N_d}{N_a}$
    if $conf > minconf$ then
        return ”Frequent”
    end if
    return ”Not Frequent”
```

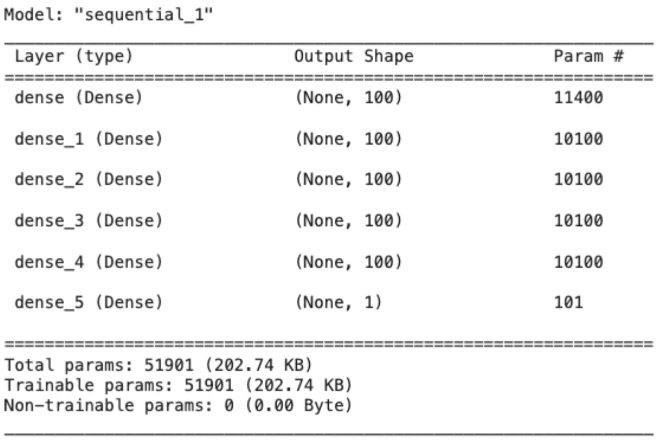
IV. Finding Frequent Patterns

We find frequent patterns by employing the VIPER algorithm. Since we are extracting information about diabetic patients, we are only interested in frequent patterns that contain *diabetes=1*. To discover patterns in the diagnosed population, we must first compute the logical *AND* operation of the diabetes column on the rest of the columns. Then, we simply carry on by computing the VIPER algorithm. As a consequence, the resulting patterns are only those of diabetic patients. However, before proceeding, we must define our minimum support for implementing the VIPER algorithm.

In this paper, we define an endangered population, or frequent pattern, as a community where its individuals are more likely to suffer from diabetes than the general population. In other words, we aim to unravel frequent patterns in populations where the percentage of people who have diabetes is greater than that of the overall population. In practical terms, we run the VIPER algorithm using *Minimum Confidence* rather than *Minimum Support*.

	Data Size	Diabetic	Non-diabetic
Original	253,680	13.93%	86.07%
after SMOTE	436,668	50%	50%

TABLE II: Distribution of DM patients in original and over-sampled datasets.



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 100)               11400

 dense_1 (Dense)             (None, 100)               10100

 dense_2 (Dense)             (None, 100)               10100

 dense_3 (Dense)             (None, 100)               10100

 dense_4 (Dense)             (None, 100)               10100

 dense_5 (Dense)             (None, 1)                 101

=================================================================
Total params: 51901 (202.74 KB)
Trainable params: 51901 (202.74 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Fig. 1: Selected model for classification.

Running the classic VIPER algorithm using minimum support brings about essential problems in our knowledge discovery. To illustrate, consider Table I, which consists of 8 individuals (2 Males and 6 Females) and their binary diabetes diagnosis value. If we run the VIPER algorithm using *minsup* = *3*, the bit vector ”$Male\&Diabetes$” will be overlooked since it only contains two diabetic individuals. However, what is left unnoticed is the fact that 100% of the Male population, ID = $\{0, 1\}$, are diabetic and are in extreme danger. To account for this problem, we consider the minimum confidence of each population as the threshold for accepting that population as frequent, or endangered. For each bit vector, we compute the percentage of individuals who are diabetic and consider them frequent if their confidence value is greater than the percentage of diabetic patients in the whole dataset. Algorithm 1 illustrates this procedure. In this code, $data$ is a pandas DataFrame containing all the attributes of the BRFSS diabetic dataset. In addition, $\alpha x, \alpha y$ are two population names, such as ”$age\&stroke$” and ”$age\&highBP$”, which contain the same prefixes, ”age”.

Other than using Minimum Confidence as the threshold for computing frequent patterns, we make no changes to the classic VIPER algorithm. Our implementation as well as all the results of this algorithm are available at our GitHub repository[1].

V. Prediction Model

We employ a Deep Neural Network (DNN) to build a prediction model that can be used by the public as well as healthcare communities to predict the probability of an individual being affiliated with diabetes. Since the BRFSS data is available at the UCI Machine Learning repository as well as the Kaggle website, several works have focused on building a prediction model on this dataset using distinct machine learning models such as Logistic Regression, Decision Tree, Xgboost, RandomForest, Simple Neural Networks, and more. In this work, we propose using Deep Neural Networks along with the Synthetic Minority Oversampling Technique (SMOTE) to account for the lower number of diabetic individuals and better learning.

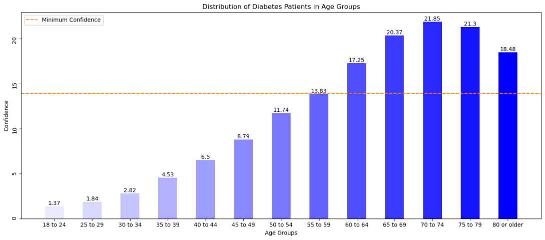


Fig. 2: Distribution of diabetic patients in age groups.

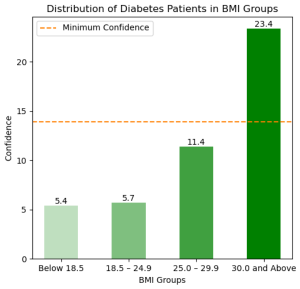


Fig. 3: Distribution of diabetic patients in BMI groups.

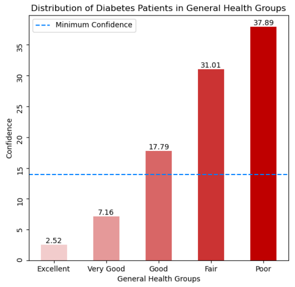


Fig. 4: Distribution of diabetic patients in general health groups.

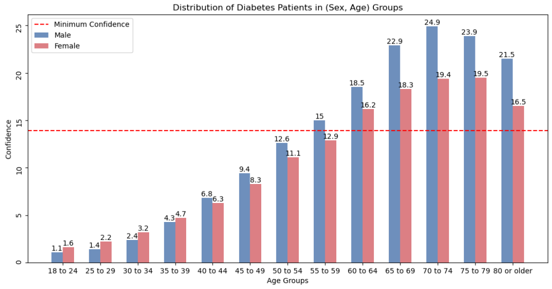


Fig. 5: Distribution of diabetic patients in (sex, age) groups.

The architecture of our model is showcased in Figure 1. It is noteworthy that each hidden *Dense* layer is initialized using the LeCun normal initializer and SELU activation function. In the results section, we overview the accuracy of our model as well as the best previous models.

As mentioned, we use SMOTE to bring balance to the BRFSS dataset. The original data contains only 13.9% diabetic individuals. We use SMOTE to create synthetic data for underrepresented classes which is in our case diabetes = 1. Table II illustrates the number of records before and after using SMOTE. Using SMOTE facilitates better learning in the less frequent class. Other approaches to solving imbalanced

[1] https://github.com/amirrezasokhankhosh/Big-data-analysis-for-diabetes

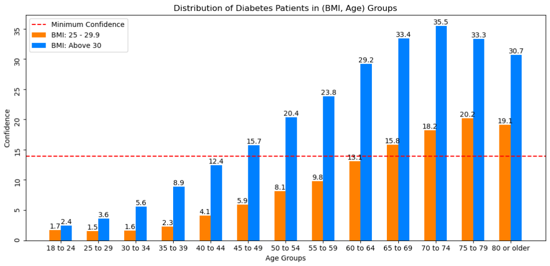


Fig. 6: Distribution of diabetic patients in (BMI, age) groups.

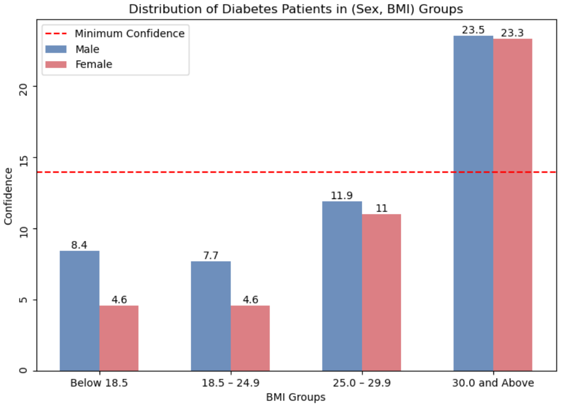


Fig. 7: Distribution of diabetic patients in (sex, BMI) groups.

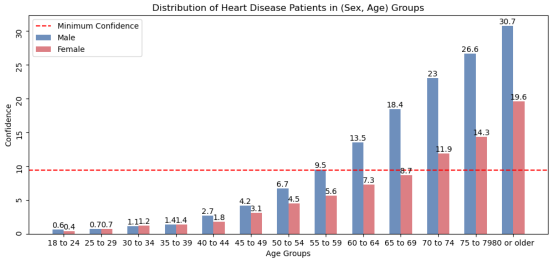


Fig. 8: Distribution of heart disease patients in (sex, age) groups.

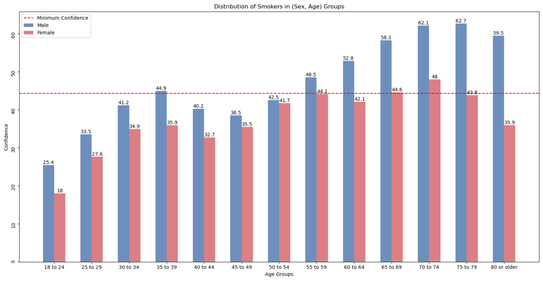


Fig. 9: Distribution of smokers in (sex, age) groups.

data problems include using under-sampling and class weights. After careful examination, we find that SMOTE outperforms the other two approaches by a large margin.

VI. Experiments & Results

First, we discuss the results of the VIPER algorithm and the knowledge we deduce using this approach. Then, we discuss the result of our prediction model as well as its improvement compared to existing works.

A. Result of VIPER

By incorporating the VIPER algorithm, we determine that people with ages greater than 60 are more susceptible to having diabetes. As shown in Figure 2 the risk of this illness escalates as age increases. Furthermore, Figure 3 illustrates that greater BMI values increase the chance of an individual being diagnosed with diabetes. In addition, as shown in Figure 4, ”General Health” shows an inverse relationship with diabetes, i.e. individuals with better general health descriptions are less affiliated with this illness. Between males and females, males prove to be more likely to have diabetes by 2.2%. This difference is best shown in Figure 5, where we illustrate the percentage of diabetic patients in (age, gender) groups. Other than the relationships above, we find that smoking, high blood pressure, heart disease or attack, having difficulty walking, and high cholesterol are correlated with being diabetic.

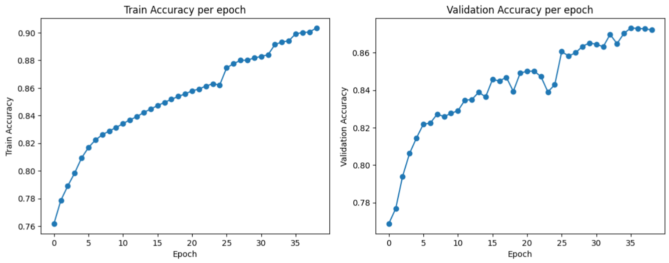
Furthermore, Figure 6 demonstrates the distribution of diabetic individuals in different (BMI, age) groups. As is evident from this figure, people with a BMI greater than 30, at ages between 25 and 69, are twice as susceptible to being diabetic than individuals with a BMI value between 25 to 30. From Figure 7, we can deduce that there is not much difference in the risk of this illness between each gender in higher BMI values.

Lastly, we find that this dataset can be used for a variety of knowledge extraction tasks other than diabetes disease. To illustrate, Figure 8 showcases the distribution of heart disease or attacks in different (age-gender) groups. From this figure, we can deduce that males of all ages are more susceptible to heart problems. This might be the consequence of the existence of more smokers among men in every age group, as shown in Figure 9.

B. DNN Results

The DNN model is trained using Nadam optimizer and Early stopping to prevent overfitting. Figure 10 represents the training and validation accuracy of the model throughout training, which shows that the model is trained properly without overfitting. The accuracy of our model on the test data is 86.33%, however, this number does not account for the brilliancy of our model since we have a classification task at hand. To appropriately examine the efficiency of our model, we compute the confusion matrix as well as metrics such as precision, recall, and f1 score. Figure 11 and Table III show the computed confusion matrix along with the precision, recall, and f1 score of our model and other models on the

Kaggle website [16, 18]. As is evident from this table, our model outperforms the existing works in the majority of the metrics. Keep in mind that some of these models are trained on imbalanced datasets, resulting in poor precision values and f1 scores.



(a) Training accuracy

(b) Validation accuracy

Fig. 10: Train and validation accuracy of the model during the training procedure.

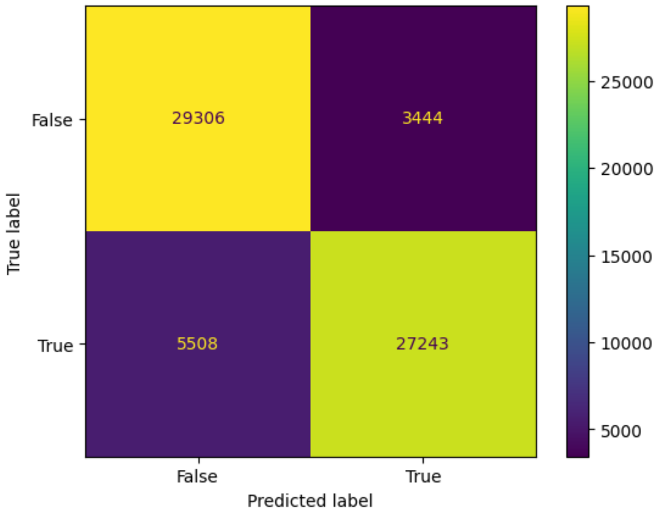


Fig. 11: Confusion Matrix computed on the test dataset.

	Test Acc	Precision	Recall	F1 Score
Our model	86.33%	88.77%	83.18%	85.88%
XGBoost (O.S.[2])	86.00%	94.00%	77.00%	84.70%
XGBoost	86.35%	51.00%	29.00%	37.00%
Simple NN (U.S.[3])	74.00%	73.00%	78.00%	75.41%
Logistic Regression	72.42%	30.00%	72.00%	42.00%
Decision Tree	79.98%	30.00%	33.00%	31.00%
Random Forest	85.18%	44.00%	29.00%	35.00%
AdaBoost	83.84%	42.00%	47.00%	45.00%
Gradient Boost	86.76%	56.00%	19.00%	28.00%

TABLE III: Results of ML models

VII. Matrix Factorization

In this section, we go over a few algorithms for Matrix Factorization. The following is a clarification of the way the problem of matrix factorization is connected to the problem of clusters we consider in our project. In fact, the well-known problem of k mean cluster can be considered as matrix factorization by considering the following description. We assume A to be our dataset. Then, the objective function of hard k-means clustering is:

$$\sum_{i=1}^{k}\sum_{j=1}^{n} z_{ij}\|x_j-\mu_j\|^2 = \|A-WZ\|^2$$

where

- $A \in R^{m\times n}$ is a matrix of data vectors $a_j \in R^m$,
- $W \in R^{m\times k}$ is a matrix of cluster centroids $\mu_i \in R$,
- $Z \in R^{k\times n}$ is a matrix of binary indicator variables such that $z_{ij} = 1$ if x_j falls in cluster C_j and $z_{ij} = 0$ otherwise.

We also know that

$$\|A-WZ\|^2 = tr[A^TA]-2tr[A^TWZ]+tr[Z^TW^TWZ]. \quad (1)$$

Putting together, the problem of k mean clusters with this interpretation switches to the problem of optimization. Our goal changes to finding the matrix factorization W and Z of the data matrix X such that the norm of the loss gets minimized. Since the objective function is convex with respect to a given W and Z and vise versa we can minimize 1 to matrix Z with a given w and vise versa. Differentiating 1 with respect to z given w and equating with zero we get

$$-2trW^TA + 2W^TWZ = 0,$$

then $Z = (W^TW)^{-1}W^TA$. To make sure our Z minimizes the objective function, we should also make sure W^TW is positive definite. Similarly, we have $W^T = (ZZ^T)^{-1}ZA^T$. **We consider algorithms for solving the following problem**: The matrix A is approximately factorized into an $m\times k$ matrix W and a $k \times n$ matrix Z. Usually, k is chosen to be smaller than m or n, so that W and Z are smaller than the original matrix A. This results in a compressed version of the original data matrix. One of the algorithms we can apply is Alternating Least Squares. The algorithm is as follows:

1) Input : Matrix $A \in m \times n$ with $m \leq n$;
2) Initialize $W = W_0$ and $Z = Z_0$; (These matrices need to be full rank, most of the time identity matrices are the easier option or we can use the random matrix with full rank.)
3) Choose a stop criterion on the approximation error δ , Choose maximal number of iterations C and $i = 0$.
4) While $|A - WZ| > \delta$ and $i \leq C$ do $i = i + 1$, $Z = (W^TW)^{-1}W^TA$ and $W^T = (ZZ^T)^{-1}ZA^T$.
5) Output W and Z.

Regularization is a technique used to prevent overfitting. Overfitting occurs when a model is overly complex and fits the training data too closely, resulting in poor performance on new, unseen data. Regularization adds a constraint or penalty term to the loss function used in model optimization, discouraging overly complex models. This results in a trade-off between

[2] Over Sampling

[3] Under Sampling

having a simple, generalizable model and fitting the training data well. L1 regularization and L2 regularization are all common types of regularization. By adding regularizers (L_2) the objective function for $0 \leq \lambda_z, \lambda_w$ would be

$$\|A - WZ\|^2 + \lambda_w\|W\|^2 + \lambda_z\|Z\|^2 \tag{2}$$

By repeating the same process we get

$$Z = (W^TW + \lambda_z I)^{-1}W^TA$$

and

$$W^T = (ZZ^T + \lambda_w I)^{-1}ZA^T.$$

The second algorithm would have a few steps different from the Alternative Least Squares and that is to give values to λ_w and λ_z plus the equations for Z and W need to be updated to the recent ones.

1) Input : Matrix $A \in m \times n$ with $m \leq n$;
2) Initialize $W = W_0$ and $Z = Z_0$; (These matrices need to be full rank, most of the time identity matrices are the easier option.
3) Choose regularization parameters λ_w, λ_z;
4) Choose a stop criterion on the approximation error δ , Choose maximal number of iterations C and $i = 0$.
5) While $||A - WZ|| > \delta$ and $i \leq C$ do $i = i + 1$,$Z = (W^TW + \lambda_z I)^{-1}W^TA$, $W^T = (ZZ^T + \lambda_w I)^{-1}ZA^T$.
6) Output W and Z.

In the first and second algorithms, we can reduce the loss via the inverse of matrices. The reality is frequently not this straightforward, particularly in the big data analysis. As data volumes explode, the size of the inversion matrix will grow at a fast pace (e.g., the matrix inversion algorithm by LU decomposition in [19]), which poses a great challenge to the storage and computational resources. This leads to the creation of an ongoing development of the gradient-based optimization technique. The gradient descent (GD) method and the stochastic gradient descent (SGD) method are among the simplest, fastest, and most efficient gradient-based optimization approaches.

Now let us consider the minimization of the objective function 2 with respect to z_n, we can decompose the objective loss function into

$$L(z_n) = \|Wz_n - a_n\|^2 + \lambda_z\|z_n\|^2 + C_{z_n}, \tag{3}$$

Where

$$C_{z_n} = \sum_{i \neq n} \|Wz_i - a_i\lambda_z \sum_{i \neq n} \|z_i\|^2 + \lambda_w\|W\|^2,$$

z_i and a_i are columns of Z and A, respectively for $i \in 1, ..., n$. Since C_{z_n} is constant with respect to z_n so that by differentiating we get

$$\frac{\partial Lz_n}{\partial z_n} = 2W^TWz_n - 2W^Ta_n + 2\lambda_n z_n = 0,$$

therefore, we have

$$z_n = (W^TW\lambda_z I)^{-1}W^Ta_n,$$

for $i \in 1, ...n$. Similarly, we can get

$$w_n = (ZZ^T + \lambda_w I)^{-1}Zb_m,$$

for $i \in 1, ..., m$ where w_i and b_i are columns of W^T and A^T, respectively. We can come up with a similar algorithm with better performance than the first two algorithms. We are hoping this algorithm speeds up the convergence rate of our problem. Because in previous algorithms the computation time of the inverse of the matrix and the storage of it can be the challenging part of the algorithm for big data. However, in the following algorithm, we can eliminate the step-seeking inversion of a matrix.

1) Input : Matrix $A \in m \times n$ with $m \leq n$;
2) Initialize $W = W_0 \in R^{m \times k}$ and $Z = Z_0 \in R^{k \times n}$; (These matrices do not need to be full rank.)
3) Choose regularization parameters λ_w, λ_z and choose step size ζ_w, ζ_z;
4) Choose a stop criterion on the approximation error δ , Choose maximal number of iterations C and $i = 0$.
5) While $||A - WZ|| > \delta$ and $i \leq C$ do $i = i + 1$
   For $N = 1, ..., n$ do $z_N^{k+1} = z_N^k - \zeta_z \frac{\nabla L_{z_N^k}}{|\nabla L_{z_N^k}|}$
   For $M = 1, ..., m$ do $w_M^{k+1} = w_M^k - \zeta_w \frac{\nabla L_{w_M^k}}{|\nabla L_{w_M^k}|}$
6) Return W^T and Z.

A. Principal Component Analysis

Before starting the section, we clarify the data matrix X can be our $m \times n$ data. Suppose we have m points denoted by a matrix $X = [x_1, ..., x_m]$, where x_i are in R^n. We want to store the points in a way that requires less memory but may lose some precision. We would like the lost precision to be as little as possible. One way to encode these points is to reduce the dimension. We use an encoding function $f(x) = c$ to take an n-dimensional point x and return the lower dimension one $c \in R^l$. We also need a decoding function $g(f(x)) \approx x$ such that it approximates x with higher precision. Assume the decoding function is the matrix $D \in R^{n \times l}$. So that $g(f(x)) = g(c) = Dc$. In the PCA concept, we suppose D to be a 2 orthonormal matrix. That is, D has orthogonal columns and $d^Td = 1$ for all columns of D.So the problem of finding a better decoding matrix changes to an optimization problem with the following objective function:

$$\begin{aligned} min_c \quad 2\|x - Dc\|_2^2 &= min_c(x - Dc)^T(x - Dc) \\ &= min_c \quad x^Tx - 2x^TDc + c^TD^TDc. \end{aligned}$$

With the constraints on the encoding matrix D, we have $D^TD = I_l$. Thus, we should minimize $-2x^TDc + c^Tc$ with respect to c. x^Tx is eliminated because it is a constant with respect to c. By differentiating the objective function with respect to c, we obtain $-2D^Tx + 2c = 0$ then $D^Tx = c$. Therefore, $x \approx g(f(x)) = Dc = DD^Tx$. Now it is clear that $\|X - DD^TX\|_F^2 = tr[(X - DD^TX)^T(X - DD^TX)]$ should be minimized with respect to D, where D is an orthonormal matrix. Simplifying we get

$$min_D \quad tr(X^TX) + tr(X^TXDD^T)$$

$$-tr(DD^TX^TX) + DD^TX^TXDD^T$$

. Which equals to

$$Max_Dtr(X^TXDD^T)$$

. Here, Eigen decomposition plays its part. X^TX is a symmetric matrix so its singular value decomposition will be its eigenvalue decomposition. In other words, we have $X^TX = U^T\Lambda U$ where Λ is a diagonal matrix square root of eigenvalues of X^TX, or eigenvalues of X on its main diagonal in descending order. If we consider $l = 1$ then PCA will take the largest eigenvalue and the corresponding eigenvector will show the direction of stretch. In fact, the optimal d is given by the eigenvector of X^TX corresponding to the largest eigenvalue. In the general case for $l > 1$, matrix D is given by l eigenvectors corresponding to the largest eigenvalues. So in our data matrix, we can calculate X^TX and by applying PCA we can reduce the dimensionality to three by picking the three largest eigenvalues. To visualize and analyze it in three dimensions.

B. Cholesky Decomposition

The Cholesky decomposition of a symmetric positive definite matrix X^TX is its decomposition into the product of a lower triangular matrix L and its transpose:

$$LL^T \approx X^TX,$$

where L is called the Cholesky factor of X^TX1. An alternative form of the Cholesky decomposition is using its upper triangular $X^TX \approx U^TU$. But for our bid data matrix X with very large n dimensionality, the complexity of Cholesky decomposition is $O(n^3)$. In specific, it requires approximately $\frac{1}{3}n^3$ floating points operations (flops) to compute a Cholesky decomposition of a $n \times n$ positive definite matrix X^TX.

C. LU Decomposition

The LU decomposition of a symmetric positive definite matrix X^TX is its decomposition into the product of a lower triangular matrix L and an upper triangular matrix U: the matrix inversion algorithm by LU decomposition poses a great challenge to the storage and computational resources. So can not be applied for big data analysis.

D. QR Factorization

QR decomposition can be applied to our square matrix X^TX and also to rectangular one X, even if a matrix does not have a full rank. QR decomposition is Gram–Schmidt orthogonalization of columns of X, starting from the first column. QR algorithms have an enormously high computational complexity, negatively impacting the process of large-sized matrices and big-data factorization. So that these algorithms need to be optimized for use in data analysis. We think we can apply Gram-Schmidt Orthogonalization to our big data. This sounds time-consuming because if we consider our data matrix with m columns of attributes then at every step we have vectors with very large n elements. We can apply the divide and conquer method for QR factorization of the data matrix.

VIII. Future Works

The BRFSS data contains plenty of valuable information that can be utilized for various applications. In this paper, we have employed data mining and data science approaches to extract information regarding diabetes disease using this dataset. Similar works can be done for heart disease or attack as well as smoking.

Another avenue of research might be doing the same analysis on the BRFSS data of different years, and computing the difference between them. In this way, we can answer questions such as: What attributes are most predictive of diabetes regardless of time? or What age groups are becoming more endangered of this illness?

At last, one can look into approaches in Causal Inference such as matching or Inverse Probability of Treatment Weighting (IPTW) to deduce causal relationships between these attributes instead of correlations.

IX. Conclusion

In this paper, we utilize the BRFSS data to unravel knowledge about diabetes disease. Our motivation is to provide the healthcare community with valuable tools and raise public awareness about this illness. We use the VIPER algorithm, with minimum confidence instead of minimum support, to detect endangered populations. We find that greater values of BMI, ages above 60, heart diseases, smoking, high cholesterol, and high blood pressure are highly correlated with diabetes. In addition, we build a Deep Neural Network and train it using the Synthetic Minority Oversampling Technique (SMOTE) to solve the imbalance issue of the data. Our model is available to the public and can be used for future predictions.

References

[1] T. Flensted. (2023) How many people use google? statistics facts. [Online]. Available: https://seo.ai/blog/how-many-people-use-google

[2] C. of Disease Control and Prevention. (2023) Behavioral risk factor surveillance system. [Online]. Available: https://www.cdc.gov/brfss/index.html

[3] P. Shenoy, J. R. Haritsa, S. Sudarshan, G. Bhalotia, M. Bawa, and D. Shah, "Viper: A vertical approach to mining association rules," 2000. [Online]. Available: https://api.semanticscholar.org/CorpusID:60478193

[4] A. Teboul. (2021) Diabetes health indicators dataset. [Online]. Available: https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset

[5] Kaggle. (2023) Kaggle. [Online]. Available: https://www.kaggle.com/

[6] C. K. Leung, Y. Chen, S. Shang, and D. Deng, "Big data science on COVID-19 data," in *2020 IEEE 14th International Conference on Big Data Science and Engineering (BigDataSE)*. IEEE, pp. 14–21. [Online]. Available: https://ieeexplore.ieee.org/document/9343361/

[7] M. C. Riddle, L. Blonde, H. C. Gerstein, E. W. Gregg, R. R. Holman, J. M. Lachin, G. A. Nichols, A. Turchin, and W. T. Cefalu, "*Diabetes Care* editors' expert forum

2018: Managing big data for diabetes research and care," vol. 42, no. 6, pp. 1136–1146. [Online]. Available: https://diabetesjournals.org/care/article/42/6/1136/36005/Diabetes-Care-Editors-Expert-Forum-2018-Managing

[8] A. Z. Woldaregay, E. Årsand, S. Walderhaug, D. Albers, L. Mamykina, T. Botsis, and G. Hartvigsen, "Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes," vol. 98, pp. 109–134. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0933365717306218

[9] S. El-Sappagh, M. Elmogy, F. Ali, T. Abuhmed, S. M. R. Islam, and K.-S. Kwak, "A comprehensive medical decision–support framework based on a heterogeneous ensemble classifier for diabetes prediction," vol. 8, no. 6, p. 635. [Online]. Available: https://www.mdpi.com/2079-9292/8/6/635

[10] S. M. Pappada, B. D. Cameron, P. M. Rosman, R. E. Bourey, T. J. Papadimos, W. Olorunto, and M. J. Borst, "Neural network-based real-time prediction of glucose in patients with insulin-dependent diabetes," vol. 13, no. 2, pp. 135–141. [Online]. Available: http://www.liebertpub.com/doi/10.1089/dia.2010.0104

[11] C. Pérez-Gandía, A. Facchinetti, G. Sparacino, C. Cobelli, E. Gómez, M. Rigla, A. De Leiva, and M. Hernando, "Artificial neural network algorithm for online glucose prediction from continuous glucose monitoring," vol. 12, no. 1, pp. 81–88. [Online]. Available: http://www.liebertpub.com/doi/10.1089/dia.2009.0076

[12] W. Sandham, E. Lehmann, D. Hamilton, and M. Sandilands, "Simulating and predicting blood glucose levels for improved diabetes healthcare," in *4th IET International Conference on Advances in Medical, Signal and Information Processing (MEDSIP 2008)*. IEE, pp. 121–121. [Online]. Available: https://digital-library.theiet.org/content/conferences/10.1049/cp_20080433

[13] J. Li and C. Fernando, "Smartphone-based personalized blood glucose prediction," vol. 2, no. 4, pp. 150–154. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2405959516301126

[14] W. Xao, F. Shao, J. Ji, R. Sun, and C. Xing, "Fasting blood glucose change prediction model based on medical examination data and data mining techniques," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*. IEEE, pp. 742–747. [Online]. Available: http://ieeexplore.ieee.org/document/7463811/

[15] J. I. Hidalgo, J. M. Colmenar, J. L. Risco-Martin, A. Cuesta-Infante, E. Maqueda, M. Botella, and J. A. Rubio, "Modeling glycemia in humans by means of grammatical evolution," vol. 20, pp. 40–53. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S156849461300402X

[16] M. S. I. Sobuj. (2021) Diabetes health indicators (86.76% acc). [Online]. Available: https://www.kaggle.com/code/shohanursobuj/diabetes-health-indicators-86-76-acc

[17] A. S. Elgendy. (2021) Example model - simple neural network. [Online]. Available: https://www.kaggle.com/code/alexteboul/example-model-simple-neural-network

[18] A. Teboul. (2021) Example model - simple neural network. [Online]. Available: https://www.kaggle.com/code/alexteboul/example-model-simple-neural-network

[19] J. Lu, "A rigorous introduction to linear models," publisher: arXiv Version Number: 4. [Online]. Available: https://arxiv.org/abs/2105.04240