**A Project report on**

**Prediction of Parkinson’s disease and severity of the type using Machine Learning**

A Dissertation submitted to JNTUH in partial fulfillment of the

academic requirements for the award of the degree.

#### Bachelor of Technology

**In**

#### Computer Science &Engineering

Submitted by

**Ranabotu Rakshith Reddy**

(19H51A0521)

**Ranga Satya Raj**

(19H51A0522)

**Gurijala Ganendhar**

(19H51A05K3)

Under the esteemed Guidance of

**Dr. Poongodai**   
 Associate Professor



#### Department of Computer Science &Engineering CMR COLLEGE OF ENGINEERING & TECHNOLOGY

**(AUTONOMOUS)**

**(NAAC Accredited with ‘A+’ Grade & NBA Accredited) (Approved by AICTE, Permanently Affiliated to JNTU Hyderabad)**

**KANDLAKOYA, MEDCHAL ROAD, HYDERABAD-501401**

#### 2019-2023

**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

KANDLAKOYA, MEDCHALROAD, HYDERABAD–501401

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



**CERTIFICATE**

This is to certify that the Major Project report entitled **"Prediction of Parkinson’s disease and severity of the type using Machine Learning"** being submitted by **Ranabotu Rakshith Reddy(19H51A0521)**, **Ranga Satya Raj (19H51A0522)** and **Gurijala Ganendhar(19H51A05K3)** in partial fulfillment for the award of Bachelor of Technology in Computer Science and Engineering is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

#### Dr. Poongodai Dr. S. Siva Skandha

#### Associate Professor Associate Professor and HOD

#### Dept. Of CSE Dept. Of CSE

**Acknowledgment**

With great pleasure we want to take this opportunity to express my heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Dr. Poongodai**,**Associate,Professor**, Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We would like to thank **Dr. Siva Skandha Sanagala,** Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete my project work successfully.

We are very grateful to **Dr. Vijaya Kumar Koppula**, Dean-Academic, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Dr. V A Narayana,** Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the Teaching & Non- teaching staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Mr. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care.

Finally, We extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

#### SIGNATURE

Ranabotu Rakshith Reddy(19H51A0521)

Ranga Satya Raj (19H51A0522)

Gurijala Ganendhar (19H51A05K3)

### TABLE OF CONTENTS



|  |  |  |
| --- | --- | --- |
| **CHAPTERS** | **DESCRIPTION** | **PAGE No.** |
|  | **List of Figures** | i |
|  | **List of Tables** | ii |
|  | **Abstract** | 1 |
| **1** | **Introduction** | 2 |
| 1.1 | Introduction | 3 |
| 1.2 | Research Challenge | 5 |
| 1.3 | Motivation | 6 |
| 1.4 | Problem Definition | 6 |
| 1.5 | Objective | 7 |
|  |  |  |
| **2** | **Literature Survey** | 8 |
| 2.1 | Domain Introduction | 9 |
| 2.2 | Background/Related Work | 15 |
|  |  |  |
| **3** | **Existing System** | 16 |
| 3.1 | Voice Data Analysis using  Decision Tree  3.1.1 Performance Analysis | 17 |
| 3.2 | Spiral Drawing Analysis using Random Forest Classifier  3.2.1 Performance Analysis | 20 |
| 3.3 | Parkinson’s Disease Prediction using KNN  3.3.1 Performance Analysis | 22 |
| 3.4 | Comparison of Existing  Models | 23 |
| 3.5 | Data Collection | 25 |
| **4** | **Proposed System** | 27 |
| **5** | **Conclusion** | 32 |
| **6** | **References** | 34 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIG. No** | **TITLE OF FIGURE** | **PAGE No.** |
| 1.1 | Parkinson’s Disease Patients | 4 |
| 2.1 | Logistic Regression | 10 |
| 2.2 | Decision Tree | 11 |
| 2.3 | Random Forest Classifier | 12 |
| 2.4 | K – Nearest Neighbors Classifier | 13 |
| 2.5 | Support Vector Machine | 14 |
| 3.1 | Correlation Matrix | 18 |
| 3.2 | Architecture Diagram of Voice Data Analysis using Decision Tree | 19 |
| 3.3 | Architecture Diagram of Spiral Drawing Analysis using Random Forest Classifier | 20 |
| 3.4 | Spiral Drawing of Healthy Person | 21 |
| 3.5 | Spiral Drawing of Parkinson’s Patient | 21 |
| 4.1 | Decision Tree Classifier | 29 |

i

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE. No** | **TITLE OF TABLE** | **PAGE No.** |
| 3.1 | Accuracy of Decision Tree | 19 |
| 3.2 | Accuracy of Random Forest Classifier | 21 |
| 3.3 | Machine Learning Models and their accuracies | 22 |
| 3.4 | Description of Dataset | 26 |

ii

**Abstract**

Parkinson disease is a neurodegenerative disorder that affects nervous system and the root cause of it is falling rates of dopamine levels in the forebrain. It is a chronic degenerative disease with progressive illness, which means it develops new symptoms over time. This happens with progressive neuronal loss in the substantia nigra of brain. People with Parkinson disease cannot do their works as a normal human. Though clinical assessments considered ample amount of data that include various features, sometimes it is hard to decide whether a person is suffering from Parkinson disease or not based on the type of data, feature selection methods help to solve this issue. Various methods are developed, proposed, and analyzed to detect the Parkinson disease, given the required data. This project predicts Parkinson disease using machine learning algorithms by applying various technologies.

# CHAPTER 1

## **INTRODUCTION**

#### INTRODUCTION

Parkinson's disease (PD), or simply Parkinson's, is a long-term degenerative disorder of the central nervous system that mainly affects the motor system. The symptoms usually emerge slowly, and as the disease worsens, non-motor symptoms become more common. The most obvious early symptoms are tremor, rigidity, slowness of movement, and difficulty with walking. Cognitive and behavioral problems may also occur with depression, anxiety, and apathy occurring in many people with PD. Parkinson's disease dementia becomes common in the advanced stages of the disease. Those with Parkinson's can also have problems with their sleep and sensory systems. The motor symptoms of the disease result from the death of cells in the substantia nigra, a region of the midbrain, leading to a dopamine deficit. The cause of this cell death is poorly understood, but involves the build-up of misfolded proteins into Lewy bodies in the neurons. Collectively, the main motor symptoms are also known as parkinsonism or a parkinsonian syndrome.

Parkinson's disease symptoms can be different for everyone. Early signs are mild that goes unnoticed. Symptoms usually begin on one side of your body and gets worsen on that side, afterwards it affects both the sides. Parkinson's symptoms may include

* Tremor
* Slowed movement
* Rigid muscles.
* Impaired posture and balance.
* Loss of automatic movements
* Speech changes
* Writing changes

The Parkinson's disease is due to a loss of neurons that produce a chemical messenger in the brain called dopamine. when there is a decrease in level of the amino acid named dopamine it leads to the abnormal brain activity, which leads to Parkinson’s disease.

The cause of Parkinson's disease is still a question mark, but several factors appear to play a role, including:

* Genes
* Environmental
* Triggers

As a result, people suffer from this disease for many years before diagnosis. The estimated results have shown that there are 7-10 million people are affected by Parkinson’s disease worldwide. People with age above 50 are the ones who has the higher possibility of getting Parkinson’s disease but still an estimated 4 percentage of people who are under the age 50 are diagnosed with Parkinson’s disease. There is no cure or prevention for PD. However, the disease can be controlled in early stage. Since this disease is progressive in nature, negligence in the diagnosis of this disease in the early stage and monitoring at different stages would create a severe negative impact on the patients in terms of healthcare costs as well as the severe health-related disorders. To prevent the major negative impact on PD patient's it is necessary to detect the PD at the early stage

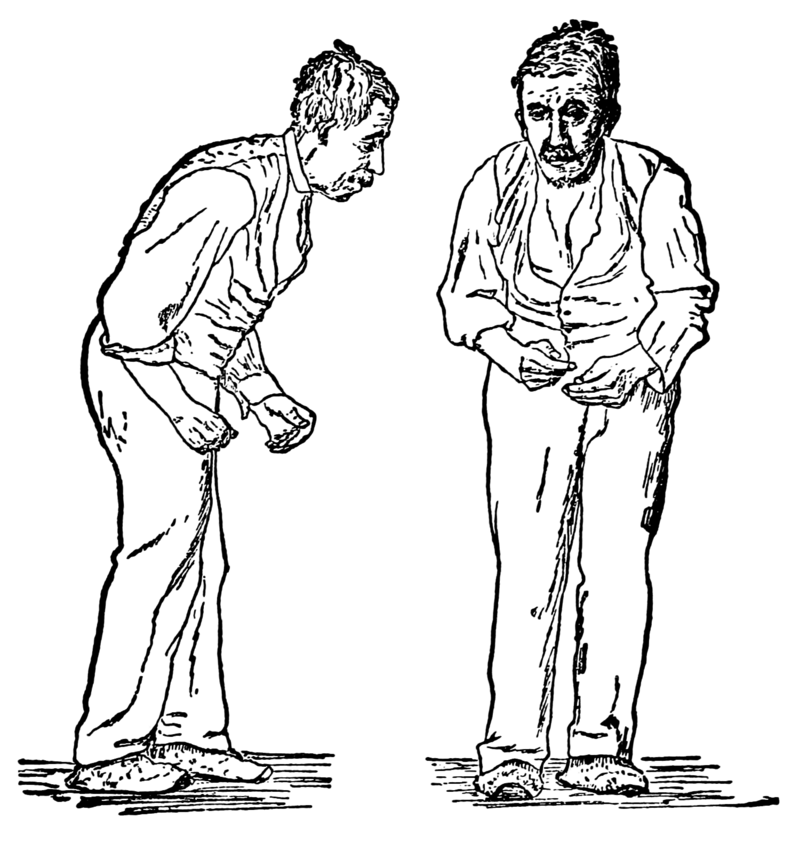


Fig. 1.1 Parkinson’s Disease Patients

* 1. **RESEARCH CHALLENGES**

The study shed some vital light on prior and ongoing research to predict Parkinson’s disease in an efficient and effective manner. Since most of the medical centers, hospitals, or diagnosis centers are not equipped with modern computer-based machines for testing and diagnosis, early detection of Parkinson’s disease is not possible. Using machine learning algorithms on the lab data, a model can be generated for a much more efficient diagnosis. Analysis based on the input and classification algorithms may give various accuracy rates

In study it is observed that until now, in the majority of cases full features have been taken into consideration. As many datasets have imbalanced class, class balancing is needed for increasing the performance of classifier model. It is important that apart from accuracy it is important to make sure that the model has low false positive rates, as it could affect the mental health of the person, if the model predicts the person of having a disease while the person does not actually have any kind of disease.

#### 

#### MOTIVATION

The brain is the main controller of our body. Therefore, any damage to this sensitive part of the human body will affect badly on the other organs. One of these negative effects is Parkinson’s disease. Parkinson’s disease (PD) is a chronic, progressive, neurodegenerative disorder which begins when a certain area of the brain has been damaged.

Parkinson’s disease (PD) is a progressive neurodegenerative condition leading to the death of the dopamine containing cells of the substantia nigra. There is no consistently reliable test that can distinguish Parkinson’s disease from other conditions with similar clinical presentations.

This disease cannot be accurately identified at early stages with MRI, CT scans. Parkinson disease is one of the most serious diseases. Hence diagnosing it at an earlier stage could help prevent or reduce the effects.

* 1. **PROBLEM DEFINITION**

To build a robust model capable of predicting the Parkinson’s disease at early stages with higher accuracy. In general, Parkinson’s disease cannot be detected at the early stages using CT/MRI scans. There is no cure for Parkinson’s disease at severe stages. This disease is commonly seen in old age people. Predicting the disease at early stage will increase the chance of curing Parkinson’s disease.

* 1. **OBJECTIVES**

The main aim of this project is to automate the Parkinson's disease diagnosis process in order to discover this disease as early as possible. If we discover this disease earlier, then the treatments are more likely to improve the quality life of the patients and their families

The main objective of this project is to understand what is Parkinson's disease and build a robust model that would predict Parkinson’s disease in early stage with minimal error rate. To improve accuracy of the model used for detecting and predicting Parkinson’s disease. To build supervised machine learning algorithms to train the models on cleaned UCI Parkinson's dataset.

# CHAPTER 2

**LITERATURE SURVEY**

## 

#### DOMAIN INTRODUCTION

#### The domain under which the study has been done is Machine Learning and the popular machine learning algorithms used for predictions. More about this domain is explained below.

**MACHINE LEARNING**

Machine learning (ML) is the study of computer algorithms that can improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks. A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers; but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory, and application domains to the field of machine learning.

There are variety of algorithms in machine learning that help us in predictions. Below are a few popular and most used ones.

**LOGISTIC REGRESSION**

Logistic regression is a process of modelling the probability of a discrete outcome given an input variable. The most common logistic regression models a binary outcome; something that can take two values such as true/false, yes/no, and so on. Multinomial logistic regression can model scenarios where there are more than two possible discrete outcomes. Logistic regression is a useful analysis method for classification problems, where you are trying to determine if a new sample fits best into a category. As aspects of cyber security are classification problems, such as attack detection, logistic regression is a useful analytic technique.

Logistic regression, despite its name, is a classification model rather than regression model. Logistic regression is a simple and more efficient method for binary and linear classification problems. It is a classification model, which is very easy to realize and achieves very good performance with linearly separable classes. It is an extensively employed algorithm for classification in industry. The logistic regression model, like the Adaline and perceptron, is a statistical method for binary classification that can be generalized to multiclass classification. Scikit-learn has a highly optimized version of logistic regression implementation, which supports multiclass classification task

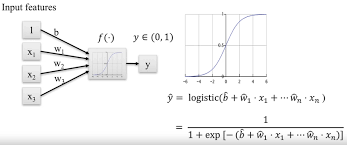


Fig. 2.1 Logistic Regression

**DECISION TREE**

A decision tree is a map of the possible outcomes of a series of related choices. It allows an individual or organization to weigh possible actions against one another based on their costs, probabilities, and benefits. They can be used either to drive informal discussion or to map out an algorithm that predicts the best choice mathematically.

A decision tree typically starts with a single node, which branches into possible outcomes. Each of those outcomes leads to additional nodes, which branch off into other possibilities. This gives it a treelike shape.

There are three different types of nodes: chance nodes, decision nodes, and end nodes. A chance node, represented by a circle, shows the probabilities of certain results. A decision node, represented by a square, shows a decision to be made, and an end node shows the final outcome of a decision path.

A decision tree can also be used to help build automated predictive models, which have applications in machine learning, data mining, and statistics. Known as decision tree learning, this method takes into account observations about an item to predict that item’s value.

In these decision trees, nodes represent data rather than decisions. This type of tree is also known as a classification tree. Each branch contains a set of attributes, or classification rules, that are associated with a particular class label, which is found at the end of the branch.

These rules, also known as decision rules, can be expressed in an if-then clause, with each decision or data value forming a clause, such that, for instance, “if conditions 1, 2 and 3 are fulfilled, then outcome x will be the result with y certainty.”

Each additional piece of data helps the model more accurately predict which of a finite set of values the subject in question belongs to. That information can then be used as an input in a larger decision-making model.

Sometimes the predicted variable will be a real number, such as a price. Decision trees with continuous, infinite possible outcomes are called regression trees.

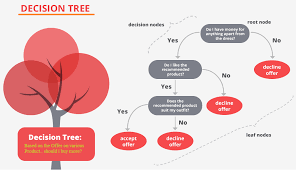


Fig. 2.2 Decision Tree

**RANDOM FOREST CLASSIFIER**

A random forest is a machine learning technique that’s used to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems.

A random forest algorithm consists of many decision trees. The ‘forest’ generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.

The (random forest) algorithm establishes the outcome based on the predictions of the decision trees. It predicts by taking the average or mean of the output from various trees. Increasing the number of trees increases the precision of the outcome.

A random forest eradicates the limitations of a decision tree algorithm. It reduces the overfitting of datasets and increases precision. It generates predictions without requiring many configurations in packages (like scikit-learn).

Classification in random forests employs an ensemble methodology to attain the outcome. The training data is fed to train various decision trees. This dataset consists of observations and features that will be selected randomly during the splitting of nodes.

A rain forest system relies on various decision trees. Every decision tree consists of decision nodes, leaf nodes, and a root node. The leaf node of each tree is the final output produced by that specific decision tree. The selection of the final output follows the majority-voting system. In this case, the output chosen by the majority of the decision trees becomes the final output of the rain forest system.

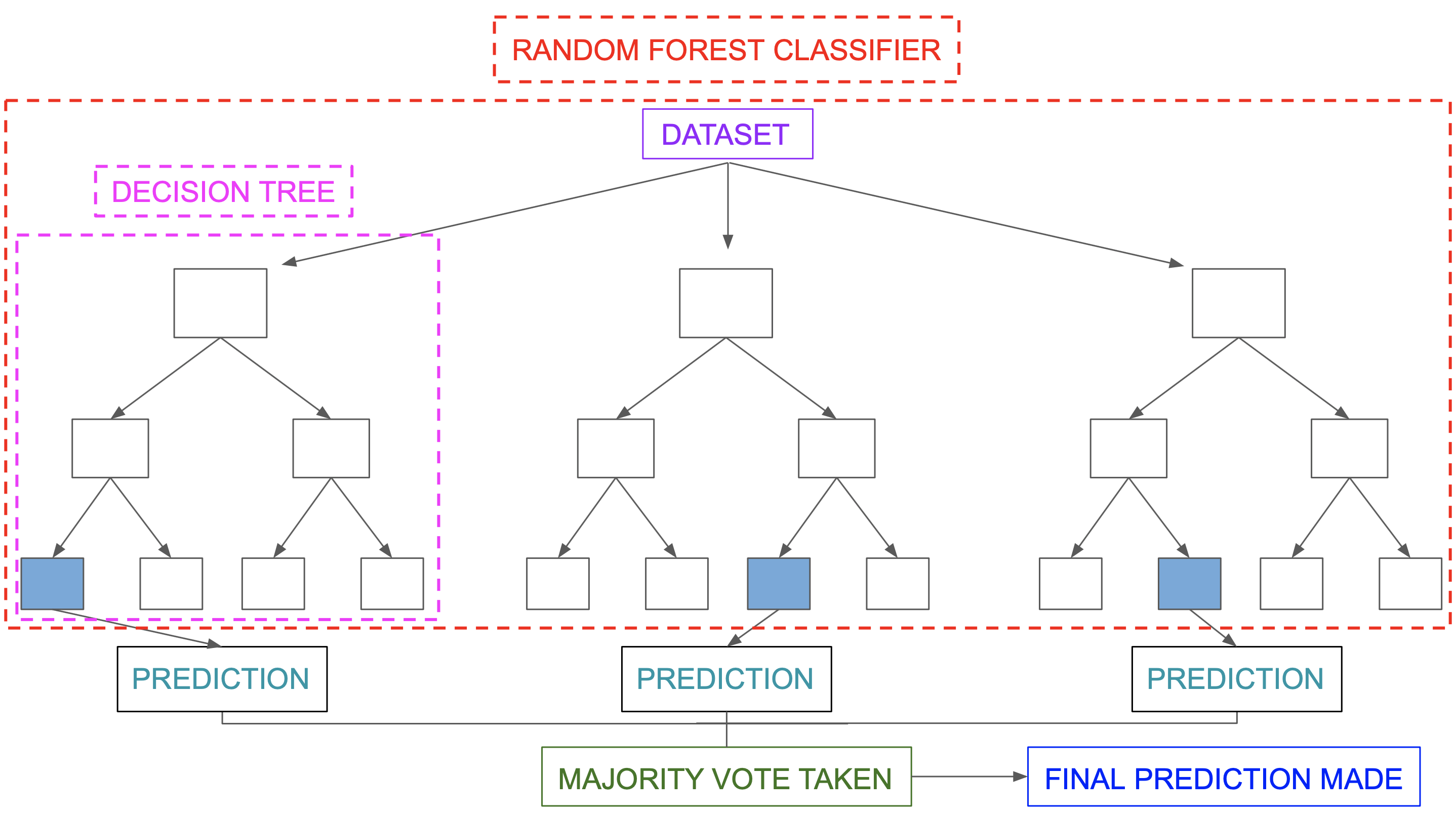


Fig. 2.3 Random Forest Classifier

**K – NEAREST NEIGHBORS ALGORITHM**

The k-Nearest Neighbours (KNN) family of classification algorithms and regression algorithms is often referred to as memory-based learning or instance-based learning.

These terms correspond to the main concept of KNN. The concept is to replace model creation by memorizing the training data set and then use this data to make predictions.

The KNN algorithm uses a majority voting mechanism. It collects data from a training data set and uses this data later to make predictions for new records.

For each new record, the k-closest records of the training data set are determined. Based on the value of the target attribute of the closest records, a prediction is made for the new record.

The basic nearest neighbour (NN) algorithm makes classification predictions or regression predictions for an arbitrary instance. To this purpose, the NN algorithm identifies a training instance that is closest to the arbitrary instance. Then, the NN algorithm returns the class label or target function value of the training instance as the predicted class label or target function value for the arbitrary instance.

The KNN algorithm expands this process by using a specified number k≥1 of the closest training instances instead of using only one instance. Typical values range from 1 to several dozens.

The output depends on whether you use the KNN algorithm for classification or regression.

In KNN classification, the predicted class label is determined by the voting for the nearest neighbors, that is, the majority class label in the set of the selected k instances is returned.

In KNN regression, the average value of the target function values of the nearest neighbors is returned as the predicted value.

By using a specified number k≥1, you can control the trade-off between overfitting prevention and resolution. Overfitting prevention might be important for noisy data. Resolution might be important to get different predictions for similar instances.

Diagram

Description automatically generated

Fig. 2.4 K – Nearest Neighbors Classifier

**SUPPORT VECTOR MACHINE**

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes

so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane.



Fig. 2.5 Support Vector Machine

**2.2 BACKGROUND/ RELATED WORK**

Machine learning calculations are exceptionally useful in giving essential measurements, continuous information, and progressed examination regarding the patient's illness, lab test results, strain, family history, clinical preliminary information, and more to specialists. Classification regarding prediction of Parkinson’s disease has been intensively researched in the past few years, Different ML algorithms may be used to predict diseases with each one having its own strength and weaknesses.

# CHAPTER 3

## **EXISTING SYSTEMS**

#### VOICE DATA ANALYSIS USING DECISION TREE

#### Performance Analysis

#### Training Accuracy describes the accuracy achieved on the training set.

# Validation Accuracy describes the accuracy achieved on the Test Set.

# Confusion matrix is the easiest way to measure the performance of a classification problem where the output can be of two or more type of classes. A confusion matrix is nothing but a table with two dimensions viz. “Actual” and “Predicted” and furthermore, both the dimensions have “True Positives (TP)”, “True Negatives (TN)”, “False Positive (FP)”, “False Negatives (FN)” as shown below FIG: 1 Confusion Matrix

# Precision is defined as the ratio of correctly predicted positive observations to the total predicted positive observations.

# The formula for Precision is Precision =TP/TP +FP

# The Sensitivity or Recall is defined as the proportion of correctly identified positives.

# The formula for Recall is Recall =TP/TP+FN

# F1-Score is the Harmonic Mean of Precision and Recall.

* + - A correlation matrix is simply a table which displays the correlation. The measure is best used in variables that demonstrate a linear relationship between each other.

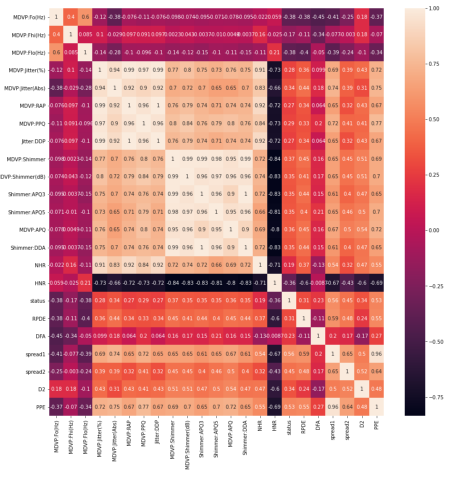


Fig. 3.1 Correlation Matrix

By using machine learning techniques, the problem can be solved with minimal error rate. The voice dataset of Parkinson's disease from the UCI Machine learning library is used as input. PD voice dataset is collected from UCI machine learning repository and these are stored into the RStudio environment as Testing and Training datasets. These are stored into the RStudio environment as Testing and Training datasets. R is a programming language and software environment for statistical analysis, graphics representation, data analysis and as well as machine learning. It involves the following steps and procedures

1. **Importing data to RStudio** - organize the data in an Excel worksheet to include column names in the first row (i.e., person’s voice collected at various time zones) and each subsequent row contains all the information (i.e. set of 22 parameter is taken into consideration and the person’s voice range for those parameters is tested and then noted), finally the status column shows two values 0 (healthy) and 1(affected). Import data into RStudio, using the "Import data..." feature.
2. **Classification** - It is also called a prediction tree. It uses a structure to specify sequences of decisions and consequences, the goal is to predict a response or output. The forecast can be accomplished by creating a decision tree with test points and branches. At each check point, a decision is made to pick a particular branch and cross the trees and can be used in a variety of disciplines, on the basis of individual characteristics

![Diagram, table

Description automatically generated](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAAAAAAAD/4RCyRXhpZgAATU0AKgAAAAgAAodpAAQAAAABAAAIMuocAAcAAAgMAAAAJgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFkAMAAgAAABQAABCAkAQAAgAAABQAABCUkpEAAgAAAAM4NwAAkpIAAgAAAAM4NwAA6hwABwAACAwAAAh0AAAAABzqAAAACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMjAyMToxMjoxMiAyMzozODo1OAAyMDIxOjEyOjEyIDIzOjM4OjU4AAAA/+EJoGh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8APD94cGFja2V0IGJlZ2luPSfvu78nIGlkPSdXNU0wTXBDZWhpSHpyZVN6TlRjemtjOWQnPz4NCjx4OnhtcG1ldGEgeG1sbnM6eD0iYWRvYmU6bnM6bWV0YS8iPjxyZGY6UkRGIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczp4bXA9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8iPjx4bXA6Q3JlYXRlRGF0ZT4yMDIxLTEyLTEyVDIzOjM4OjU4Ljg2ODwveG1wOkNyZWF0ZURhdGU+PC9yZGY6RGVzY3JpcHRpb24+PC9yZGY6UkRGPjwveDp4bXBtZXRhPg0KICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICA8P3hwYWNrZXQgZW5kPSd3Jz8+/9sAQwACAQEBAQECAQEBAgICAgIEAwICAgIFBAQDBAYFBgYGBQYGBgcJCAYHCQcGBggLCAkKCgoKCgYICwwLCgwJCgoK/9sAQwECAgICAgIFAwMFCgcGBwoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoK/8AAEQgBPAI8AwEiAAIRAQMRAf/EAB8AAAEFAQEBAQEBAAAAAAAAAAABAgMEBQYHCAkKC//EALUQAAIBAwMCBAMFBQQEAAABfQECAwAEEQUSITFBBhNRYQcicRQygZGhCCNCscEVUtHwJDNicoIJChYXGBkaJSYnKCkqNDU2Nzg5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6g4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2drh4uPk5ebn6Onq8fLz9PX29/j5+v/EAB8BAAMBAQEBAQEBAQEAAAAAAAABAgMEBQYHCAkKC//EALURAAIBAgQEAwQHBQQEAAECdwABAgMRBAUhMQYSQVEHYXETIjKBCBRCkaGxwQkjM1LwFWJy0QoWJDThJfEXGBkaJicoKSo1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoKDhIWGh4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uLj5OXm5+jp6vLz9PX29/j5+v/aAAwDAQACEQMRAD8A/fyiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKM0AFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVyMnjDWh4i1jTVePy7K+SKH93ztNtDIc8/3nb8K66vPZf+Rx8Sf9hWL/0itqqIup6FRRRUjCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACmyPtFOrB8Xaze6aIUtmA3ZJJotzaBfl1ZtedR51cT/AMJVrP8Az2X8qP8AhKtZ/wCey/lVeykT7aJ2pmOOtNWZj3rjD4p1jHMy/lTW8WaxGm8Opx/s1XspGcqnNJWO2Ezk4al8056mvMfFPxr0Twj4k0HwbrviC3g1XxG0q6TasDm4MYywH0GK4S//AG/vgJperWGh3XxLtTNqGoXFlbtHA7R+dBN5MgLAYUCTC5PGTS9nKSsaucY6M+ifNPqaQzOOK8F8a/tq/Cf4feEI/HXinx3DBp0sZZZI4WkY4kMbDaOeHDL06ijxh+2n8LPBGlaJrGveNbdYfEV1Fb6R5MTSNNJIFZRheR8rqefWlClKXu9jLmuz3vzTik85w3Jrx2f9pTwt/aljomm+KLK+urzVhp3lWMokaGby2cq4B+XGxutWdZ/aM8J6L4o1LwVf+JraHVNJ0T+1tQtZOGgtME+Yfb5T+VHs5KKl3GpPncex6351HnV836b/AMFBvgXq3iKDwvZfEOE3Vzp630GbaQI1u0fmqd2MbigyF6mtfxv+2p8Jfh7qWm6J4l8d28N1rEyw6eixO4lkZVYDKg7RtYHJ4GaPZyL9rE9586kaXIxmvAPHn7b3wp+G/ie38GeJ/GkC6leaXNqFrawxs++GMNliw4H+rk6ntVy2/a28I6kNHn0fVU1C21ryBY3Vi6yRu0rFQuQeoxzxxTjRlKSj3FKrGMebse6CX5lSpKp6azSQRyv1bmrlQ/ddiovmin3CiiigYUUUUAFFFFABRRRQAUUUUAFFFFADLiTyoGk/urmlRiUU+tOopPUAooopgFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAV57L/wAjj4k/7CsX/pFbV6FXnsv/ACOPiT/sKxf+kVtVRF1PQqKKKkYUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABSMflpaR/u0AIhJ61y/j7O+3x6N/OumZykbMB6V87ftxftD+OvglP4Rt/BXg+31Q6xcXB1CS6l2La20SB3fpyxHAHrWlH+NFGdX+Gz0HDLxmlVc9a+MLX/gp1478SXfibRPA3w3tdTvtD8TNYx/M8cclulo9y8m5lGWATbxkEsOa0Lb/gpf4tsviP4J+GGv/CuRdQ8QX06ap9jjeVIIRJGsRVlXAJEmTuIxiui7ag+6Zz8q9pNdrH2AUHamsuF5pFuGK5K0jzZXBT8q0CMdTxn9rv8AZy8W/HC58NeI/BXiRdL1fw7aXb6XfFfmiuJMKGU9ht3fpXkOofsJ/E/RZF8MeDJ9HXQ9We/ttXe6hHmwW1zeR3G+L/poCDz13c16n+2P+0p8RvgNqWg2PgD4dXHiIX2j31xdfZZYwbXyym2U72HA3EkDJ9q574UftOfE74i/EO1stYsNOt9Ni8XT6PDJY3QkW8UWgkLN/dKyblPuDSjo7mkve1ZmeJP2Dtf8TW/iPQZfHMljY3niK3utEmtgDLY2pczXKDdkbmmZ2HHRq63xX+x/4e8V+JPA9rf6akWj+B7tprOS0uJIbi5kEUMcTyNGwLbWQkqeCvbFe7LHEoxGpCbdoX6cZ/MUmxSGB5DNnHp/kU4+7JyXUjY+SPg3+yN8Sfgr8ZdH8eaja2lxbr4kmhuW0qIg3MTRT7by4A4aXMgDNXbftG/sh6/8WPjT/wALF0HxKljBqOn2mmaqm35rnTtv72InqdzBB9GevoJ1j7DAB+UdMcY/nivnf9ob9qz4ofC/40yfDvw98N5LzSlbS45NfeaMQ2JmSQsJAW3Yyq8hTipn/DUexa+Ny6s5Lwt+xH8UrLxZZ+DtcuNIPhG3m06+nkjU/a/OtbYRCNDnO0nqe4yO9aGlfsA67qyeEl8d+Ppll0fV76fVGsdu69t2SNIIDkHCbY1Bxg11/wCzN8fvH/xa8V2M/i+xtbWHVvDL3dna2Fx5kSGK6kjDg+jptPIr3ZDG6KpVtu38TVR1p3MX/EsfPXjf9i/SfiP8R77x9r+nQ2s//CE/2HpNvY3DxwxZmuzJI6KwV9yXMedwOO1cz4H/AGQ9e+Bfizwvq3hqyhfSdP8AEUJ/srTY9qwRN/HtHGAcn8c19V7Ru3ZP0/L+gpYsCUZz6/jwM/lU6x95bodlL3Xsz0PS0KWcP+6KuVV0wk2MJP8AcFWq4d22dqioxSQUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFeey/wDI4+JP+wrF/wCkVtXoVeey/wDI4+JP+wrF/wCkVtVRF1PQqKKKkZ4J+3l/wUI+En/BPzwdofjD4raFrGqf8JHri6Vo+naHbedcXFwYmkwF/wB1G/KvnBf+Dif9ncjLfs0/Fcf9y5/9etT/AILL26y/HX9k4SIrK3xyAZSOv/Eqva9SGmWAH/HlH9BGKAPGv+Iib9nX/o2r4r/+E5/9ej/iIm/Z1/6Nq+K//hOf/Xr0vTvGHgXU/GOoeAbW7t31bS7KG7vrFVBkihlLBGI9Dsb8q5fW/wBpf9nvw/4cj8Val4z0/wCxy6pcadFJGVYyXMDlJkUDrsYEH6UX96wHOf8AERP+zr/0bV8V/wDwnP8A69J/xET/ALOu7B/Zq+K//hOf/XrtPD/xr+C/izS9B1fQPEljcweJ7p7bRWXb/pEyozMg9wFPFUIP2kf2f7rxxN8PLXxbYvqlu0omjVRtVo1LOu7puAByPUEU+V6+W4P3Wl3OZ/4iJv2d+v8AwzX8V/8Awnf/AK9Kf+Dij9ncHA/Zq+K3/hOf/XrpvhL+0V8Cfjjfyab8MvEVrqTRw+czwx/Js3YLbsYqlo37Vn7NuveN5Ph3p3jnTW1RdSksDAxUA3KH5os/3h6daQS8jH/4iJv2df8Ao2r4r/8AhOf/AF6P+Iib9nX/AKNq+K//AITn/wBeu91n4q/Cjw/8RtP+Emra3ZweItU0+a9sNMk2iSSCL7749BkVyHhT9sf9mPxt44/4Vx4a8Y28+s+akf2HyPmVnfameOAcMc/7NAFH/iIm/Z1/6Nq+K/8A4Tn/ANekP/BxR+zuDx+zV8Vv/Cc/+vXdeEPjB8HfHuma9rPhPxPp95a+Gb6az1ueNl22k0S5kVz/AAke/UVz0H7V/wCzZdeCdV+IsXjWxGj6LJFHqF5IoCxNI2I8/wC929aHJRXM9gclFXZij/g4o/Z3Iz/wzV8Vv/Cc/wDr0g/4OKf2eAOf2avit/4Tn/16m1f9t/8AZR0Dw+vijWfG1tbWMkjKtxJblVyMcE46/MPzr0P4c+N/AHxX8Mx+L/As8N5YTSNGJhHjLL1HNOMXztP7O/z2JlLliu72OM+Hn/BwN+zN48+J3hn4X3vwc+IGg3HirXLfSdNv9a0XybcXMzbY1Zs9zX3lbzecm7HevzF/4KXWcFva/Az7LborN+0V4WBKrjj7SeK/TiyiaGPy2P3eKlX6lE1I/wB2lpsn3DTAhlc+UcVwfxb8HeEPFj2CeKNLjufs6yrb+d0CkAMPxHFd1ISICR1r5D/4Kba/8R7DVfh7oHw+j12a41LULsLDoMbF2lWHdH5m3pFu+8egFFOoo4mnC2/X5E1FzYeU10O28E/s+fs3aLLe634Q8J6eq6heNJeSQyFhLKVaMnk8HaSpAxWhffAT4F6rqen+IrjwZbvqGk3fnWNxC7K0Um0DJ2kZzgcHIyK+Kz8Lv2qvhj4Kk0l38WXNnqmrfa7qHSFZ5ra8lSXZtCjKwh1G7t8wJr2X9m2X4l/CDxvrHiv9o3xpc29jrNq+P7WuAsUFyt04WOPPT90AfeuyEb+z+Zj1v3PpmXxR4OtvNW81SONrfH2hWuFBi6ct6c1Pa6noF6PLt5t0iqrMiyA4Ung/Trz7Gvz7+LPwu+Ms3xH8afEtvDeta14M8Q6558cOgSSvNe2Cv0CqfUqVKjpmvXf2SV8c+Hfjx/wj3inwl4qj+0fDvTYbq/1C3l+xx3ULSOYlZht3BHXPctkdqcG3G/mydj6g8TeCPBPiBLW71nSo7iaO1eBZH6mJvvKfY4rjPD/7N3wf8IeI7PxN4R0T+z5INQub64ihclJriZCpcg5wfmPTHWvQbpN1rbn+8tV/K/2asXMy1u0jcxw/LZxnpRu0j/pp/wB9VV8r/Zo8r/ZoEWiNIkG0h/xasfxb8Nfhz4kMx8Q+H4rpr5IWui//AC12qQufoGOKvCL/AGam1Bd0sa/9MV/lUtXK5mcX8NvgL8LfhTrX9o+C9Oazt49MWxtbNGJWCMSmQ4LZPJY967gvpJOTv/Oqvlf7NHlf7NUtFYm2ty1u0j/pp/31QraV5g2B93G35qq+V/s0scWJVbb3FTL4QXxI9F08H7NCQP8AlmKtVX044s4f+uYqSebyscda4TsI7i/8i6jtvKZvMYDjt71YpiIHbznX5v4fan0AFFFFABRRRQAUUEgcE0UAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFeey/wDI4+JP+wrF/wCkVtXoVeey/wDI4+JP+wrF/wCkVtVRF1PQqKKKkZ8H/wDBZL/kuf7JX/Zdf/cVeV6sc+UceleU/wDBZP8A5Ll+yX/2XX/3FXlerRuA3zj5cjHvQB8j/trePpP2WPj7Z/tLoMWeveBdS8P3SJ/y1v4V8+yX3LEsg92rB/Z8+EumfCn9pX4afDb4gi3kSw+D0tzbNeMPLuNWnuBLezfNwXZ3Y564NfTnxl+B3w2+PGg6f4a+KHh1NQsdO1a01G0RmKmO4hYFH9/mHTuKj+Lf7P8A8PfjaunjxhpbC60nclnfWcxhli/hYBlOcFQMiplHVSjuEbc3vbHxvoUkVv8AtFx2ngxV/sKH9oHVBoLQ/wCpEn9jMXWP/Z87d04zXq37F+n+AJP+Ceuh3ur/AGGXXpPDt/Nq0lxt+1DVWSc3Bb+LzBJvr27w3+zn8KfC1h4b0fSPDsUUXhXUHv8AS35LC5dCjSsf4mZSQSaydP8A2SvgvpXi+78caRo9xb3Fw1w81mt032dpZlKvII84DNkn8a2hLSo5ddvuE1zOLfTY8z/4JY/8LL/4Zi8KSeMLHwzHp0mhRjSJtJm3XckYkYnzwVGDj0JrxjVbXwvN+xX8VLyKO1XxFD8YtUbw7LHj7QNQ+3r5OzHzZzx9K+u/gr+yn8MPgHdR3fw/W8t447VraO1kvHeJI2OSFUnA6ms3QP2LPgZonj+b4gW3h2RrqbVpdV+zTXDNALx23GbYTt3bvaspe7FWHCXs7uWtz44+NPx6ST9oe+/aAPhvVbm88C+PtI0yLUobNmt105IjBfgydAu+UMfdK9x8SfEDwd8JPj78f/jCNJtTDpHw38P3tv5MSg7vKvNhXjqSU6dSRXvVr8BPhhb/AA21b4T/APCOR/2L4ga4l1S2Y582SZt0hJ65J59qq67+zH8JPEMOrWOpeH/MTXtN0+z1DdIf31vaNiBT6gBmpSdok025Ssz4t/ZHu9C1LxD4o/Zy8R2+reH7L4rfC2G9vNQ1C1a3xq8e9boqWxklJVYkf3Kq/G7xd4+8HfAHxZ+yt4/0Lw/ceJvCsfhW70zxJprD7LrGnPqIhg8/A+R1eJ9wwflORmvu34g/A34cfEq/0vUvFWirJcaLDPDpskbFTEk0fluvGOCo6e1cd/ww/wDAN/B+teENQ8PzXkWvNbfb7i5unabEO1ohvJyAp5AHck0VI81NRezev5k1IucZRXY4H9qLRfEl5+w7r118UdC8MLqq3Vtu/wCEf/eQqhuYCvJVSGweeO1fSHhXTrTT9Ds47G2jhj+zxnZDGFXJUE9K4vQ/2avhrongbUfhhPBeXmk6ndLLcQ3t20jMyMCOT0xtHFehQIltGltEuFjQbV/ugKAB+VbUtfbye8mvuWi/CxnPmcqT/lTT+8+a/wDgphjyPgTj/o4rwr/6UGv00j/i/wB6vzL/AOCmBAT4Ep6ftFeFt3/gSa/TSP8Ai/3qzOgdSOMrjFLRQBWuT5UJbH8QH61x3xBstNuLvT9TuraNprUObeRl+ZMgZx9RXbXhAtnZh0XNeJ/tOX/7QWlnSIvgl4I0XWI2kcaj/a2oGDyVBG0pj73Ga0pe7UT7E1Peg0eF3f8AwUStre21IWvwW1i4ubPxDcadb2PnRLNdQ28Zknutpf5UQBeDy2eBXeeCPGfwv/atGu+GPEXw/e4svD97E0barH8twZbeOVXTnph8ZOMkVxd//wAE3tA8T3d94m174m6xHruq3y3lxdW08amFTGY5LdMJjYwfBJGTgc1syfs8/Fj9n3WNa8ffs/Tp4qvPEE1vFNoniDUhDbWcEEWxDEVUHIAVeSeK6klG1uhidr8NfiR4e1Xxz4m+C2heFbjTY/Bq2scMkkYWGaOSMNiIeg6c/hmu+a124kX+HHFeY/Cv4BeIdA+KmrfHnxb4ruf7V8Uadaxal4fW4WS0s3iXgRHAOB6knNerBAykeYvfHP4ULlWhMh0v/Hta/wC6a5/4lfETw58KPA2qfEPxdcNFpuk2bXF3JHGWYKoycAdTXTT22y3twZl+VTXF/Hj4ca18VfhJr3w98OeILXT7zWNPa1hvLq286OLdwWKZGeCceh5qgSOb8W/tbfCbwbJfQatqFx5mn+HYdamjjtyc20pATHqxLAYrP8QftsfBvw98EtM+PV1c3jaPqtwYLWGO1Jn8wK7OpTqCoQ59K4Pxf+xB8a/GSfZ9Z+NOiyRyeA49BuNuilWkuEcMLjIfOBtGB2qDxF/wTv13Vf2bdN/Z1s/ifBHb6ddR3C6nJbMZjIyyfaCDuBG/eQPT3rP95ytdehp+79pFdLanXn/goP8AAWd9ch0qbUr5tDtlnuPsdi0nmxkKS6Y6hdwz6V7JoXiPTvF2i6X4o0dy1rqOnQ3FuzDB2ugYZHrg18ry/wDBMzxh4dufEuofC34z2uj3Gq2dvY6a0li0iwWa+WZY3Xd8zSbMFhjg19UeEPCl94V8KaRoWqX9vNcWelwwzS28XlxsyrglV5wKoz0uZPxN+K/hL4TWOm6h4uunij1TV7fTbQpGWzPM4VQcdB6mvMtS/wCCgXwQsbDxTqFpDq14vg7V/wCztbW0053aOTaSWAHVRjr0rq/2ovgv44+N3gax8K+A/G1hodxa65aX8t5eWP2jcsEqyhFGRtJKjJ9MjvXCeF/2LPE1hr/jDWfEHj7Tm/4S66tzdppunmFTCkm9lYFjl2wQW44NaRtfUGrHZ+Ov2sfhj8OpfBqeJ3vIV8bsq6Q/2VtqsQrASH+Dhh1rn9I/b5+CutW0d/Yxas0La7Hpckh05wI3k/1ch9I27N0NM+PP7J3jf43poOi3nxE0+z0fR9Uklkjj08mdrbcmyON93yMFTaTg5z2rhtG/YA+NuitpkWnfH7TTDba5bT30EmjfJd2VshS3gI3feXcxLdzjiso80nK5UoxXKfc2k3Sz2sOz7pQVceMSFS38JzVDRbaeG2hWRg22FQzL34HNaNcbOgKRSSORSLIrlgp+7wadSAKPwoooAKKKKACiiigAooooAKKKKACiiigBsrSLGWjj3Nj5V3YzWa194t8jI8PWYk9P7SYqPf8A1Yz+lalFADYjKR+9UfhTqKry38cUhibrQBYoqNJwyhgp5pfO/wBg0APopnnf7Bo87/YNAD6KZ53+waPO/wBg0APoqN59oyVqNb0M23B6ZoAsUVHBN5u7j7rYqSgAooooAKKKKACvPZf+Rx8Sf9hWL/0itq9Crz2X/kcfEn/YVi/9IraqiLqehUUUVIz4P/4LJ8/HL9ksf9V1/wDcVeVpftM/GtfgF8Jr74gRaT9vvPtUNppljux591PIsUK57DJyazf+CyX/ACXP9kr/ALLr/wC4q8ql+3D8LvFfxT+Al1YeCrfz9V0fVrLWLG33Y+0NazeYY/qV3Ae+KmT1QHCav+0h+0D8GvETeE/jxaaLcy654J1HV/DV1o8e1EurOEytbyZAzkENnpgGuf8A2d/2qP2ovjPPp3gmTUfCLa14l+H1j4o0u9s4y0Fis+0vBOAM7x5mBx2qH4mad8Q/2tfGEPi22+GuoaRY+DfAutoI775ZLzUb2zMKQxj+6oHX1Nc/+wD8G9c8KfE34d694b+HGpeGYdF+HNvpPjqXUPu3t0scSoiD2ZWP41tRtHEa7WFV/g38zpvh1+1Z+0+P2ZvEXx1+IR8M3Vx9vbRvDGmafbsvnaib42cO/KjKFiDj0Brr/Cf7W/ivXPgx8JvHF3o1qmseLPGlr4b8U2i/dtrtJZYbgL/uyRH8K8m0/wDZ/wDj34v+FPwz+DOkm58OrZePta8Ua1qTQiQRLb3EslorLkZy0m7H+yKk8Ffs9fHL4ZePLb4Yay1xr2maZ8dNP8T2uuNCI1aC4jeS5O3JwFkJOM9WrnjzSi1Lo9PTmX6Nl1N4tdtfub/QteGf+CmPxD1H4UfGLXfEPg+xs9e+H+tgeH4CP3d/p7XsdsJj6kFmB9xXfeBvj7+0F8Uv2hfF3gbQvEPhTT9A8H69a2MljeL/AKbexyWyzuV46/NxXzr+0x+x98brj9mePxX8NPDcz+Im8aXmma7pa/evdHuNVS5WQ/8AXN4lYezGvUvhn4Qsfh1+2N8QPF3jj4Q6xeX2seKLGXQdYtlJgSL7FHDuPPZwa3qRjJQS3sr+tjGnzxhK292vlcv+Bv2xv2i4PD2n/HX4h6boE3gO/wDHk3h66jskK3Vgpufs8NwxIAI3YB5P3hXqv7U/xo+Jvw68Z/DvwH8L9W0KxuPGusXtvcX2v/6mCOGzecY4PJKivnv4beCPip46+D1p+yBN8NtQsnf4lyaprusXi4t4rCLUftI2erNtUfjXu/7WHwMj+L3xi+DcGreHZL7RdJ8QanJqjKxAhU6e6RkkerAD86i3M7Fxjyu6PPvjl+1j+0T8MfiNc+BdK17wi6+G/honijWJrj/mJyfaTEYbbjuOn1rqpP2k/jp8avG03gD9nnSNLsbjRPCmn6v4hutejLL9pvIPNhswoBIYRhCTj+KvP/2mP2K734ofHrxNPonhyZbPTvgtDa+Fb5WO1L6G780Re+QB1rQ+Geu/E/4B/E7Wvi1rnwp1LULX4i+EdIuXgsVHmWeqWlmLSSFh2UrGhBoj7zcXspfoNe7K5neK/wDgpT4+0e2+EPiOy8HWcen+KNVv9O8fQzcvpzWty1rMVPtKOvpXpPxv/a/8TeBP2vvh3+zx4G0e2vNL1yZv+EovZOWtlkRjbqnuVjY/TFeP6L+xd471jw54T8G+LdIYSaxo/i+/1iVeV0+/1KY3UMefVGdRn1Wj4A/Ar4+6laeAfjD8YPDsn/CWX3xKaTVk6m1021tHtLfJ9CF3fV6qT5aza2Wi+5L9Bcq5Wu56Z/wUq3xW3wJQ/Mf+GiPCYY+p+0MM1+lkmv6PZTNBfanbwvnO2SZVOPxNfmn/AMFLXLQ/AiRR979ovwqB+F02a+5vin+yp4C+MPib/hKvFN9qiXAjESrZ3hjUAe2KnYZ6J/wlXhv/AKD1n/4EL/jQ/ivw2qFhrtnwP+fhf8a8W/4d+/Bz/oJa9/4Mz/hSS/8ABPz4NtGynUte5X/oKH/CgD1K4+LPw/8A+EitfBUniqxXVL+3aezs2uV8yVFIDFRnnFeLftx/tO337P8A/wAI2umeGI9SbV5rgSeddLF5MUab3cFiMsB0XqT2rEu/+CXvwxuP2hfBfx6j8X65HJ4NiuGtdP8AtxKzTycZc91C5+XHXmu4/ac/Z48G/GrxD4T1bxxaLcweHp7i4is2HyySOm0E/TrTpS/2mC6dfuJq6YeTW/Q+V/B3/BRrx/pvhKRfE3w0W51R9Qd9La4vkiW6sFVpGkJLDawC4C9TjpXsH7Mv7W+q/tBeNdc8P3ngJtJtba3e40ed5gxu4UmMLMVz8uGFJ448A/sc+JdLe18QX2g/ZtN1WG3lZptqwXAEm2IkdCQxrS8V/CNPEdu1l+zh8SbTwjq+nzeVqF5Y2i3DCOT955TDcNuSd1dlGXPGLfVmE7xrOPax4l46/br+Nek/Gbxp8LNBj8PxfYdaGnaFdX8ZSO0IYjzJ2IwykDtnB9K9Q/Zq/aV8Y/Fr4it4Z8U2ml29s/gvTtXsVt7gGaR5mkSUleoXdGNvsue4rUX4JfAr4l2F98OPGE2l+INetIw3iG4CbJ5ZnZS0rgdNzLxycc11nhP9nT4T+CfHn/Cx/C/hSOz1ZdLi05bhCf8Aj2jX5Y8eg7URfNH5iPQrp3a1tzjqtV8v6VYl/wCPa1/3TTasVyu28tjFKQ4HSo9T1Ww0m1m1DUrlYbe3jaSaaRsKigZJJ9MVhT/F74ZWt4thc+NtPjmbTzfLG03JtgMmX/dxzmr548yb6C5JnRfvNmMfw1NfyvujTH/LFf5VjWXj3whqXhZvGdh4itJNJEbOdRWT90FXgtn0BrTe7tr9La+sbhZYZrWN4ZUOVdSvBHsahfEhPbQNzkZC00s5OQtLI+0qp/iOPp71kXXjrwfZW95dXPiazSPT3C30jTDbbseiv6HkUTlTj8vyKlfmi+l9TW3ODsKUsRkVlO3+KsfS/H/g3XNXutB0fxNa3V9YruvLWGXLwj3qn/wuj4WpPHaN4709ZJL82Ua+dybgdYhxy1OPwu/r8gUpS372PatKJ+xRgn+AVPNu8shDzWNNavfwWf2e6mjkt5FkVoZCu7AwQfUHPQ8Vo2YvPPY3b5+UbfT3rgl8R1ktqjRrtapaKKkAooooAKKKKACiiigAooooAKKKKACiiigAooooAK+G/wBtL49/tdfFj9sFv2Fv2MvFGm+Gb3SfCMeveL/FmoQ+YbNJnKwRIuDywVj07V9yV8S/BSN5v+C4nx+jKfd+EvhAhvq13/hSv73yf5Eydo3OMh/Y5/4LLsNkH/BR7S9o4XGhD/43Syfsd/8ABZqIFm/4KOaX8vX/AIkY/wDjdehf8FTPiJ/wglj8KdN1Dxzq2gaHrXxIW08Q3WisVna1FrK20Yz/ABAVq/sMfEj4reOv2PtW8UtrUmrXUc2qJ4Nvr7/W3VsryC1aT3IVaUW3GT7X/C3+ZUtKjj6Hkzfsf/8ABZcIH/4eO6Zg+uhD/wCN0R/sf/8ABZaSPzF/4KP6Xjdj/kBj/wCN1yvwu/aO8d/DP9kf4jfFPQfHetaj8VtD8O+f4m8G+IGx9kvFnKG5hB/5ZD5gAM5BFfSnwp/a+8Q6jZeF/C3xI8DSWGvat8MZ/Fd4iygqqwsEK/VjkilzONNyfl+Kf+QP4orve/yt/meKt+yD/wAFk0jaaX/gpFpKoikszaIAB/5Doj/Y/wD+Cyk0Szxf8FItKaNhlWXQxg/T93XRftAf8FDLzxR8DdQ0b4TfDPUdUv8AWPhO/iTVWt5AraXaT744n923K54/umoPgx+1WPhVZXGveKNW1bVJrf4a+ExZ6KxHlyXlzHsQIc53u3LZ9KUpSSi/OzJjLmS8zIk/Yx/4LNupB/4KPaWcYP8AyAxz/wCOVe/Zx+NP7cf7Mn7Yvhf9kf8AbL8faZ42074gaPeXfhnxNY2/lSCa2AaSJlwMDBr6H/Zt/ajv/jR4z8RfDDxt4Fm8M+KPDKWs95pM0gfzILlC8UqsO3ysv1Brxv8AbSj/AONqf7KQ2f8ALt4ox/4DR1oUfadqcr07VNUNqpUtmpqACiiigAooooAK89l/5HHxJ/2FYv8A0itq9Crz2X/kcfEn/YVi/wDSK2qoi6noVFFFSM+D/wDgsl/yXP8AZK/7Lr/7iryvVst5Z+XPHA9a8p/4LJf8lz/ZK/7Lr/7iryvV1+7SauBXSJirEJtHQjp+P4UqxuDmI/LwMY/WrGBjGKAoHAFN6yuEvejysrrF5JwV43ZwPTuPxqMrKxWQxLu4ZeOhxzVwqDyRSbV9Kb1dyub3bf1tYpT32nWzw297eQxyXUhjt1dsGR8ZIA78VC2q6O13c2v9o26yQIr3EZcbol/hZvQcH8q8Z/a1up7P4vfAryLloUk+J0qzKrECVf7KvPkPr8wB+or5/wD2uvF/iHQvjP8AGXRdI1i4s4daXwHot3cxSFTbW93fNBOy4+6SknWolJ06qcuun3pMrSKcv63sfbeiePvAWs2zXuh+MNNuo45xBI9vdKyiQ/dUn+8fStmXE7hMfdb8jkf0zXy9+1H8FPh78Fvh74WvPhtoi6Y178SvDltdLbkqssaXBGTjrkcE96+n7ZCsh5+Xkfqf/rVpK1Oo4PdGFOUqseZbBGJo265GBSSIZfuxAAnBJXpj09jVnA9KAoAxipj7vN5u5ZWWF412AbgzOzD0Jb/AUqw8+Xt43fL7CrGPajA64oA+X/8AgpZlYvgSrR7f+MivC23HQD7Sa/TaP+L/AHq/M3/gpl/qfgT/ANnF+Fv/AEoNfplH/F/vUAOoY4Wikb7tAEMj7QGxXk37RXx0+G3wnl0yLxvrbWrX282oWEvvx246ZzivWLrd9nYr97/69eHftT/FX4Y/CZtCvPiZYmc6hcSRWcaWvm4IALMeOFUck+lFG8pxT3loEuWNNt9D4t1j9mn436nNqt/4b+Es114auPE1xcWml3jfvrhru3I+0lurLGwBX+7ur1v4Na/N+yp4x8aT/Gnw5daLourXFgbHWGLy/b7qK2SKVgOwJUt+NdF4R/4KMfBjVPCd1rupNqFvJZ6udPNrDasxkwGPmpgfNGEXJYcCu3+HH7QPwg/aK1/WPBnh4tqUmgur3H2y1/dvlmTMeR8wDKQSO9d1OKjKEVstDGXLJuT3epxXwBsvE/ij9prxd8Vbr4cNp+iX+lwweH9XhTauoQf6zzHHdyScegFfQnlzEbPJbO2vnnxD+3x4S8F+K/E3w5sfBN1dXnh28Fjp9tZuM3s+eUUA/JgA4zjIrtvg5+0zoHxm8Xy+GPD+nXkH2fQLPVvtE4wrJcmXCj1wYyD7kipp/D82ZyPVpbecW9uDC3CntUfkTf8APJvyp15cXP2a3Zbhvu1X+03v/PVq0JOP/aB8Max4j+DnijRtM0GXUprzRbiCGwjkKtM7RlVAI5HJr5ksfBfx5a+8MXup/syiZtB+G9xaKsj/APHxdtDt+zyHuhCgY/2q+y/tN7/z1aj7Te/89WrN003c0VSSVj5GvvhD+0n4i/Y2/wCFOeC/h7HpeqXXm3GrWt/KVilEjyvJCuMbRuK4A7V9K/Bbw14t8PfB7wjoXjGzWPVLHw5Z299HGdwSVIlUgH8K6T7TeHgzN0qxqFzcAxBJ2z5K/wAq0XuyRBXns7h0yIW6EHjqO9fFPxn+AX7Unji0+Jng/wCF/wAOYrG01rxIt4ZNSuW238Yi25Q5GPnG7A9q+1vtN7/z1aj7Te/89WrKVGMua/VNfePm/d8nmn9x8Z/s3/AH9o34d/tBeNvG/iHwdcRweINLitfME2Y4pzKR5kXsEJLZJ5UdK4LwT+xN8e9LsdLtdV0fXpL7T/Hy6hpl5NOreQ23Essvy/PG/GAeRjrX6E/ab3/nq1EVxdGZd8rYzyK0k9n2SX3ApPSPnc7nQbaeGythdNukWFQ7L0Jxz+tadVbHmCE4/gFWq4L3OoKKKKACiiigAorD+IvxL8DfCbwrP43+IviW20nSbVlW4vrp9qIWOFH1J4FZvwx+Onwq+Mukya38MvGNrq1vDMYpmt2O6NwM4ZThl/EVPNfYDrqbv9qh/tGA7dpzubArLt/H/hC78U3Xgm1123k1axto7i8sVk/eRRSZ2OR6HBxVJPqHkbW/jOKaZiDjFYup/ELwhpPiDT/Ceo65BDqWrJM+m2TP+8uBEoaQqO+0EZ+taQvIZQrg/Kw61Wl7BH4tS0sgJxRvGcVXW5jznd36fhmoYda0ye9lsYLyN54QPOhWQFkz0yOoz71HclczNAHIyKKqpqlptz5nSnnULcdW/hyKavYonoqGK7SRtg606S4jjGWpXtowJK+J/gvexWX/AAXF+PUV0dhuPhH4RMO7jeFa83EeuOM/WvrrXPij8N/DF5/Z3iXx3o+n3G3d5F9qUUT49cMwNfHP7bv7MXh744fG3Tv2mP2aP2yNG+Hvju30dtI1HVINXheO9sckiN1D8lScgnpRa7uD1PbP2pf2Z9V+Pnij4Y6tE1u1r4N8df2xqdtcKGW4g+zyRbMHrywqn+zZ+zd8Qf2fvhz4s+HWmaratbN4g1G78HDbkWlvMxeKNh6IWYY9MV8pj9nL/goGBhf+CxWigdv9Ih/+Kpf+Gc/+Cgn/AEmM0T/wIh/+Koto13/UHrJyPU9d/YY+Onxu8XeMviN8XU0TSdW1L4dXHhfTY9HUBLtpZUl+0T4PJDIAB23GpvG37JP7WmrWHgvx14X1DQ7bxVpvw8vvB+swTHdAtvK6NHcJzywKnI968m/4Zz/4KCf9JjNE/wDAiH/4qj/hnP8A4KCf9JjNE/8AAiH/AOKpSjzRUfT8L/5k8vvX/rW3+R3kn/BPv9pTwJ4ctNL+F/ibSTc638HbbwT4puLxM+UbdpWjnj57+fICPpWrrv8AwTk+J+oafd3On+LbWDUrPwz4Xi0iTblGvtK3btw/uvkivLv+Gc/+Cgn/AEmM0T/wIh/+Ko/4Zz/4KCf9JjNE/wDAiH/4qnJc0OV97hCHJKLXRWPqv9l/4BfFHwx8WfFn7RHxwmsV8ReJtPsNNWz03mK2tbUSbecnJZ5Xb24FeWftl3kNx/wVY/ZTWNxu+zeJ2255ANtFg15M/wCzn/wUE24/4fE6K3/bzD/8VXZfsn/soXPgP9oWz/ab/ax/bh0X4jeJtD0uSw8MyTarDHHYRyn96wG/qQAM0yj74tJ0nTclTV5i/wC1j+z1pHxF0b4Vz/FLRW1jxDG7aTbw6lE4mZeqAhj83t3r0uGdJ03xnigB9FFFABRRRQAV57L/AMjj4k/7CsX/AKRW1ehV57L/AMjj4k/7CsX/AKRW1VEXU9CoooqRnwb/AMFlZUT46fslA/8ARdM/+Uq9r1lDlc1gf8FVv2KfjJ+19pnw31n4GeM7bQ/EXw78bHX7G7u4RIjH7HLBtwf+uleARfsLf8FkFjAH7Unhz/wVx/4UAfT9FfMX/DDH/BZH/o6Tw5/4LI/8KP8Ahhj/AILI/wDR0nhz/wAFkf8AhQB9O0V8xf8ADDH/AAWR/wCjpPDn/gsj/wAKP+GGP+CyP/R0nhz/AMFkf+FAHr/x6+CGl/G7wvY6Vd381jf6TrEWqaLqVv8Afs7uMMFcDuCpKkd81xmn/sU+FNS8CeLvDHxL1y417U/GjRSaxrUyhZS8Kr5BTH3fLIBHuM1yf/DDH/BZH/o6Tw5/4LI/8KP+GGP+CyP/AEdJ4c/8Fkf+FTy63f8AWlinK6t/W9zpbX9jzWtX0C10f4k/FvVNfOn65puoWL3ChRF9jYlFwO5J5Ne6QiUYLn3+lfM3/DDH/BZH/o6Tw5/4LI/8KP8Ahhj/AILI/wDR0nhz/wAFkf8AhWnNdtvqZxjy6I+naK+Yv+GGP+CyP/R0nhz/AMFkf+FH/DDH/BZH/o6Tw5/4LI/8Kko+naK+Yv8Ahhj/AILI/wDR0nhz/wAFkf8AhR/wwx/wWR/6Ok8Of+CyP/CgBf8Agpnxb/ApvT9orwrn/wACTX6YW8qyBiv96vy/m/4Je/8ABSL4p/Ej4f6l+0B+0Loup+H/AAf4+03xHNZ2tiiPI1rKHABA71+nWnI6NNvX70mRQBapH+7S0UAQzLviZB34r5t/bh/Zw8Q/H/xJ4JttP8SPpljpM18dUkjjDNNDJDs8vn+90z2r6XYqi7iK4v4mahp8QtzPeJHgOcswHTk1VNfvoy7E1Vz05R7nyt4g/wCCffg3VLKzh8M+L7jS5rPCLJDGpDQsrJJHg/3g4GeoxWvD8DfEf7NlnB4n+APhuPXL6eJ7OW1vJNiiFpnnLZA5IY459a+YPE/xi+JNlqPiTQP+Fi+KptLi8XXLTeILG1l/dzG3ZrS0jGPub12tjg8V9H/sffGC98QfEPx94V8a+Lb64vrOfT51t9QtWiitSbSMSRRFgA2JMg4rsiuTXzuc8t0+ysVPEH7COkeOJbz4jxaxceG/FuvXX2/VJ7M+YIbo8jg/3RuUHuDXYfB39lW9+EfxQj8dad8QLiexi8LWuirpLwAL5cBchy2fvGSSRv8AgVct8FPE/iG8/ba+I2hReLdV1TRpNLt5IbSaNxb6dMrnKqTwSQe3YV9H7kEeN38NEVZW+YmyaX/j1tf901G7pHjcPvHFSSuotrYFv4TXmf7XXiK98L/s6eLtf0vX7rTbi10SaSC8soTJJG4U7doAJ3E4A+tUNI9GaeFRuPTrn2xnNH2mAAEkDP8An+tfDnxW/aR8ZTLew/Dfx1rEl5q3wjtLizt5NLkCW0m9BLMTj/WbNxx14qj45+M/xkm/YS8MjRvFOrWmtLfKms6y1rI07WoSV45OmR5hRR7UiuVc6R94G5t/mUOudvqKfdSK1xFt5/dKP0r80G+PPxItPF3xLufiX4s8YWNjJo9kNPtbG3m36bdssKgZA6M5Zjj+HNfoX8Kb6S/+Gvhu/n1GS8km0G28y8kjKtOwQAuQe5OaXUzOkldIl3OKja8tV6uOmeK8Y/bl+Ilz8Nvh1oPiK08TX2nY8Zaakv8AZ9q8r3EZuF8yM7Rwvl7yT6CvnXxD8avFPh64+KWt2/xU8RSaXbeKtHl0+4XT5WEIeYb7WMY6FMkkcYFMrl0PvRpIlfy24PH60RyxtJtB5XqPSvnT46/H7SbTxr8KdW0XxpqlvY33iqW0vra302RluF+zSZMnHCq5j68ZI9K8z0f9o67svD+iSXXxO16SS1+M09heNJpcu64ti7YQ8cRqMc9KUvhYkrs/RexkRbSEt/dFWY5o5DhGrzr42+KPHHhz4Vt4q+GujDU7yz8m4ey5DTQghnVf9ornFXvgd8WfCvxb8D2fjfw3ebobmLEkT8PBIOHjcHkMrAjB9K4TpjK6O5opsU0cw3RtmnUFBSNkrVa51rS7OTy7u9jiJ+75jAZpP7e0T/oL23/f5f8AGh6gfNP/AAV2t7e5/Y8a1vEDRSeOfDqSK3QqdTgBH418/ftifFm6/Ys/aD8deLPgHpKRnUPhzoqapYWeBHBqV3q4sYpsDhW8uTP4ZNfa37Tnwg+HP7THwnufhP4y1oJZXV5b3PmWt0EkjkhkEiMpB4IZQfwrzyw/Ye+Aq/DnxN4I8VavNrsvivYNY1fU77zLqTyuYSHJyPLYBlx0IzUwjySuB89eFf2if2wbk+G/gF4s1K40PUPGnxEk07RfFWoKpmjso7E3UmVHG4lSg9a1vid4u/aC+HvxK+JXh/4f3lrrXjHSfhroLrqkUYE1zuu5VlYc4LCIZUete23/AOxF8JdX8AWfgvWPHupXV7p2sLqmm+IptUzeWtysZiDI+ePkO3HcU/8A4Y4+FMS6pJB4uvo9Q1LRLPTJNYTVD9oC28rSI4bOdxZjmtJy9xsI/wAaLPnWL9qu7tE+FvxMg1z/AISO+sdJ8ZG/uNQtfLuoZra0VjCwBO1gwVW65Ap3in9pf9pb4I6Noev6r8TY9ej+Inwr1fxBYW4gAGj3lrHHMuMHmPEm3nuK+gfAf7EPwC8I2Oj+ZJ9tm0eXVpXmvLsM15JqKbLl5OfmLLVbwt+wR8BdAt9Qt7zX7vUFuvD82hWK32oeYNNsJjl4YMn5QeM98AVnT/iKT7EVtpW7/gcr+yJ8UPj5bfHyx+F3xf8AHq+IIPE/wos/FkEn2cRm0uGungkjGDypUp+Veb+J7v4z/DL9rX9qr4q6B8U7ryvDPgTS9Q0zR5o8xLKbSYpjngAr2619b+G/gl8MvCvxG0v4l6ZqEX2/S/CKeHbZmuQQLNZfMAxnrurn/H/7J3wh+IPxJ8RfEzVvEc0dx4s8NpoviGzt77bDeW6BghZc8soY4PvWdLm5Yc3d3+8fPKPtEuyt9x8sfGH9oL9rjwXovw5+E+g+Kr7VvEvjjwpd+Jrq+023VWt9qRBYFB6rukye+KyvGX7a/wC2/wCKPGum+DfCXgm6tfEXhnwf4d1LxBpdqyGGa6u/9fHIx/gAVuRX1p8Tv2Tvhd8VLPwyH8TXGj6h4RtWtdH1bTb/AMudLdlCNHuB5BCAkVQ8UfsWfB/WvFGn+MbDxZe6feWek2um3zWWplP7Rtrc/ukmIPzY9TzXQ/iCHwL0PeNAubi4gtnu4ds0lurygdAxXkfgavXkcrMu1eP4qzdG1LR7SNIW1O2UKMKvnjgDgd/StB/EOgpw2sW3P/TYVnKPNNSKPP8A4h/st/A74r6+fE3xE+H9tqV95YQXEsjg7R2+VhWD/wAMG/sof9Eesf8Av9L/APFV7NbXVpexefaTJIh/iQ5FSYHpVAeK/wDDBv7KH/RHrH/v9L/8VR/wwb+yh/0R6x/7/S//ABVe1YHpRgelAHiv/DBv7KH/AER6x/7/AEv/AMVR/wAMG/sof9Eesf8Av9L/APFV7VgelGB6UAeK/wDDBv7KH/RHrH/v9L/8VR/wwb+yh/0R6x/7/S//ABVe1YHpRgelAHiv/DBv7KH/AER6x/7/AEv/AMVR/wAMG/sof9Eesf8Av9L/APFV7VgelGB6UAfDf7dX7M37Fv7IngxP2/fFvw7MNn8Ibf8AtKOzsZXzJM80cSuSWOcFxxX1l+z18U9H+OPwQ8K/GXw7byRaf4q0O11Wyjl+8sU8SyKD74avlz/g4gA/4c1/HTj/AJlm2/8AS+3r2P8A4Jk/8o7vgj/2S3Q//SKKgD3KiiigAooooAK89l/5HHxJ/wBhWL/0itq9Crz2X/kcfEn/AGFYv/SK2qoi6noVFFFSMQorHJWk8tP7tOooAb5af3aPLT+7TqKAG+Wn92jy0/u06igBvlp/do8tP7tOooAb5af3aPLT+7TqKAG+Wn92jy0/u06igBvlp/do8tP7tOooAb5af3acFC9BRRQAUE4GTRTXzsNAEd45a0fyT82Plrw39qD4TfCP4h3Gjz/FfVWtWgkZbH/iYCBWfIOOvzHPavbpQwt2A9K+S/8AgpF8KviX8VPEHw40T4f6VHcf6Zf/ANoSXLMI7YGDCzfKR86tyueM0QlL21Nd3r5aEVG1h3Pqdx4L+KfwL1fwwuo6PrGi/Yre+Fk/2iaJdkyHaFfJ4cgHHc1D8Q/h18E/j/by+F9Xmtbh9NuFnuI9JvFWaNjkKX2cjODwa+V9Z/4Jz/FHRNCS08IXOn3n2q8EmpWd1NIqyyvHKhueGH7xDsI7cniu6+GXgfTP2ENUvPiZ8R7zUtQh8RI9pePpNm91LLcCd5ldl3cARnbn1rti4y5bdTDX2ji+lj2Ky+O/7M3wqtdT8IjxzptnJ4Ojjh1hZFcyQjoA7Bfnb6ZI712Xhf4v/Dvxrqb6L4V8RWN9dR6fBfSQW7EstvMWEUh44DbGx34r4/8AFP7H/wAT/HPi7Vv2gPh1PY30XiDWm1fT9H1q1Kq0ZbIWVQc8g8gk4KjFelfsvfAf4vfCL4wDU9b0XSl0VvA2n6XdXlvu8+W6g3OWOTjb+92j2X604SvG/mxNH05d3ca20DLboflqleSWt5btFeafHJHj5o2XcrexFTy/8etr/umo5GjjXc/SqHzFb+zNC+5N4fs933d32ccp0xnHTHalmtNFZFt20O1Mar/qfJG3Hbj6Z/OpHljLYz977tBdR94/5xVcsO5etufqiKPTNAbzJpfDtm/mjEhe3X956E1dkngtRHBDZoqiFNigYCrjoBUIljKYU/w/0p906tcxgH/liv8AKs7GXUiu10/UlWHUNKgmjVs7ZIwwHHXnvjP51FLpWhzwsp8O2ZV23tGYRhmHQnj0q0Sq9e9NM0StjNVYuWxWksNHlRI30O1YRsWjDQj5c9x6HOPypV07QmID+H7M5YN/x7r9/OC3Trx1qyXjUbu23NNWaLzBz6frUyXukc2tjspYXfTo0ji3BkUbR/Ovn/4leEvEf7K/xJuvjj8PtOkufCussp8ZaHCv+rk6fbIlHdRywHWvpHTxmzhyP4BVbxJZQahZfYbiyWeObKyRuoKlSOQR3FcJ1xjyoq+CfE+heJNBt/EGi6nFcWd7Cs1tPG2VdCM5rbR1kXehyK+ctKS5/ZQ8bjQr3zJPAniC8I0+RidukXLn/Vn0iY5x6V7/AKPqFo+nQlZ1YOvyMD94etAz84/+C0PhXxJ8Z/2wf2eP2crf4i614f0XxPDr9xqjaLeNDJK1vHGUyQR/erz1v+CPHhJRlv2mfiL/AOD2T/4qvY/+CmSK3/BUH9lkltq/2X4q+b0/cwVuftW/G3W/gp8PrOfwlp0d3r/ibW7TRdAt5fu/abkkb29kRXfH+zUTk6cVfduxdOHtJNLorngI/wCCO/hJhkftMfEXrj/kPSf/ABVJ/wAOdvCXH/GS/wAReTj/AJDsn/xVej+N/ir+0L8ENV8E6B4/1ew1RfFHj610tbyGDY3kPC7NwD1yox7VY/bp/af8bfs8adoMHw6sobi+kuJNV12OVd3l6NaSQ/a2HocSrg+xpykqfuy3IPMf+HO/hI8/8NMfEX/wfSf/ABVJ/wAOd/CJbb/w0v8AEX/wfSf/ABVegfGf9pH4gv8AtEaD8Evhz4h03RLbVPCZ1u11DVFzHqbGUYt0bIAITJPfBpPDn7QPxf8AHv7UviD4UWGsaTpNv4T1GyhuNDu1xLqMMsP7yeNieQGJIx/drSKvKK7/AOVw+y3/AFtc8/P/AAR38JA7R+0t8Remf+Q7J64/vUH/AII7+Egf+Tl/iL/4PJP/AIqtq9/ax+Mni5PBOg+HfFGm6FceJfGniDS7vUryMFIobASGMcnq23mtr4k/tf8AjX4Ma74q8Ma+bXWJtB8E6TdaW8K7VvdQvLhoVHH8JOD9KzjeXLLv/lf/ADDq1/W9jjP+HO3hIjI/aX+InHb+3ZP/AIqg/wDBHfwngf8AGS/xF64/5Dsn/wAVXs2h+Mvjt8JvAXiT4k/HbX9OurS00H+0LQww7WhmC/6k88g8c+9eWaB+0x+0t41/ZB8TfEyHWdP0fxl4Dub5PEVk1vuWVVRZ4QAT8oMcic+uaqrs5Lul94SXLG5m/wDDnbwkDj/hpf4i/wDg+k/+Kpv/AA548IZx/wANMfET/wAHsn/xVddY/tL/ABe+EkfhHUfjNrtrqNlrnhXVfEN9NbwBGjhtrWOXYvvkvUGh/tFftHeErXwD8afihHYnwv4+1O1tpNHhjxJpi3ZPkHd325AOe9CWrXb/ADa/Qnm963lf8E/1Oa/4c6+FM4H7S/xF/wDB9J/8VR/w548ID5z+0x8RPlUn/kPSemf71ZOjf8FIPihD8OvjZP4n0e2g1jwHqU0vhFlXCXtkt2tvnHcgk5r7a0mdrzSrW9lC7p4w7Y9T/SiPvRuHN71v63scj/wQG8UeLPFH/BPazPjHxTeaxdab4217TYr6/lLyvDb30kcYYnqQqivtavhz/g3u/wCUfUv/AGU/xT/6c5q+46CgooooAKKKKACiiigAooooA+K/+DiH/lDX8dP+xZtv/S+3r2L/AIJk/wDKO74I/wDZLdD/APSKKvHf+DiH/lDX8dP+xZtv/S+3r2L/AIJk/wDKO74I/wDZLdD/APSKKgD3KiiigAooooAK89l/5HHxJ/2FYv8A0itq9Crz2X/kcfEn/YVi/wDSK2qoi6noVFFFSMKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigApHOEzS02T7hoBEbOoTc/TvXK/EB0/0d0+8qkr7V0s6mW1ZD7fzrxr9oz4c/FLxrfaVJ4E+LN74ditVkF0lpGG+07iOCM9MVULc35BO3K09l/mfKOvftwfGTTNS1jwhdalpNhrtv4gvRFb3kEfl2+n20PmgbsYZphjb1Iwa9y/Z4+NWv/GHxL4s0Txba6bGumy2R0+2tmDukM9qku5+4bLfhUkH7GP7N+v6HJHd+Gv7Qa7vo727vjJuaW4RGQtn0I3Aim+K/2U7LSb2+8U/A7xZdeDtd1y7RtW1G1Xd58arhUwTxgAYrbC+7CkvUxqWliJv0M34W/tTWfxF/aw8U/AnRrqxbT9B0GO4t442Hni4Sd4pgw/u8AivcC6iPOO1cP8P/AIS/C3wRrUnjR/sD+JRpq2uua5vRZZwW3s0nPBJ55ruYn0OY4huGb5Q3yuD8p6Hr0961p/D82ZyJJJF+zWw/2TXDftD+NvEHw9+DniPxt4VhtZL/AEvSZLm2S9kCx7lGck+voO9egXkemG3hJaT7v8Nc38Q/APg74leEbzwR4ntZLjTdQjEd5Czbd6Z6ZqiT59+KX7ZXjnwFDf6naaRp95GPh9a6tpNvZyo0txdyukZ6nhVLfQ4rD8Z/tu+PvDn7HHhv4u6NpUl94o1jUjaXkMdvFutmTe0m5CNnATHTB7V7Q37Hn7Pvnx3k/hli0OirpCszk4tByEx6e9WL/wDZX+Bl74KX4dSeFD/ZKtEfs/mY2lAQGHuQWzUcmpp9pPsfO3xX/bH/AGivhvb6l4j0ZbPXNI1DS9Ok0WOzsI1ls5LvawMm4YJCb2wfpX1T8IvGb/ET4aeG/HBfe2paPDPLIybdxZMk47c5qn4g+A/wm8VeHb3wnrHhlWsdQaE3Ea8EeUMR4/3en0rq9J8N+HPCelWPhrSLUw2djZpDaxJ/AijAFV1JPPv2oPil4m+E3hLR/EfhxbILceJtPsb6S+kC7IZ7hImK56thj+ANeJ6/+0n+0c1/8StE0PXNJtbnwlfW0mi+fZxyfbLaTgIMA5LPtAJ5r6M+LnwW+G3xt0CDw5450uS8tbS8jvIYWk24lQ5VvwrL8Mfs3fB7wpLqF1pfh7dJql5Fc3zytku0R/dj6A4NMbPHP2mP2rvin8FdZ+FnhrTjYm48TFz4jdoUZbdlktIvmyPkG+dumDnHpS6D+0N8fvD2vRw+O9Q0K+06b4iQ6IuoJGkO6Ex5fYONxDkLnr8pr2Xxp+zd8G/HesWOveKPDgvLzTbyS4tDI/3ZHdXb8MoDj2pPFP7PXwd8c2el6f4g8PM8Wl6t/aGnovy+XcE58w++TUS+EhfEj3zTJUe1iCNn92tWiAeorL0iAWkdvbxDKrGoX2AGBWpXGdhzfxE8I6J430mXwt4h0tLqyu4ik0brkcjA+h968X+Gfi7Xvgl44X4D/E7UXayuAH8Ga7cMcTR5P+iux/5aKMYzyR619FlFY5K15f8AtPfCZPjL8N9U8H2Voq6l5JfSLw8Nb3IUmNwe2GAoA+Qv+ClziT/gp7+ysin7+leKlH18mCrn7aXwz8YeP/Anh3xX8OrNbzWvBPjCz8RWmnSNj7WsO5ZYgfUxyPj3xXzv8Xbv4wfBf9p79lG3/bS8Z6bDr+lz+NI7jWLi62wzQuImhYs2MEgnj2r6wH7Vf7MmMf8AC9fC3/g3j/xqXHmSv0dylJxvy9VY8I/aK8bfFz4v6f4B+Iujfs6+ILe38H/Emz1C60uRk+1SwCCRXZRnGAWAFZHjv4DfFr9tj40+JfiDq8OveBfD6+Bx4c0yxvdoluVn+a4JA/vNtH/ARX0d/wANWfsyYx/wvXwt/wCDaP8AxoH7Vn7Mo6fHbwt/4No/8aKkfaT5nuEHy0+U+U/Fnw/+KXiH9lbw/wDBL4m/s+anq3ivSdFbTvDfii0kHm2VzDIYUmkfOVUxhHyOuTW98W/hh8S/HHifwP4dg+E17bePvD+raQ0nxFtZAtvLZw4a4EjA7mLKZF2njJFfR3/DVf7Mn/RdfCv/AINo/wDGj/hqr9mPOf8AhevhX/wbR/41opWlF9v8rCvpb+tj5p8G/sleKPE2v/DfR/iX4Ikl03S/H3izUNU3NxHHceYbdz7NW1+0X+yr478b/EPxoPh/oaxWsXgnQW8OtI3yTXljeNP5Ge2QAPxr3wftV/syA5Hx18Lf+DaP/Gj/AIaq/Zjzn/he3hf/AMG8f+NTH3YxXb/Kwu/9dbngnx51z4+/tXfBpPgx4e+D2teFBqmpadZ61qWoSKDFAX3zlMHlAq/+PVkXf7LHxo+GHiD4weHNIv77xNpvxD+HqTRXM0n3NUgTyBH/AMCi2n8K+kv+Gqv2Y8Y/4Xt4W/8ABtH/AI0f8NV/sydP+F7eFv8AwbR/40S96NvNP7gl70bHjPxl/Z18Z/FRPh74Ul0loLNPhrrGi6xdbv8Aj1lubKOJAf8AgQrn9I0b48fGvwT8MP2b/FXwjvtEbwfrdjP4s8QXLL9mnjsG2qbfud7KDz2avob/AIaq/Zjzn/he3hf/AMG8f+NH/DVf7Mmc/wDC9vC3/g2j/wAad9X5/wCbf6k21v5W/BL9D4h/ag/Yt+PXiH9mHUte+G3hGT/hLh451SGbS1kAa90a51AS8nPUBFcemTX6GeH4pLfRrO3nBWSO1iWRSPuttGR+Brkf+Gqf2YyMf8L28L/+DaP/ABpkv7Vf7M4Klfjt4X+9k/8AE3j/AMaUfdjYOX3r/wBb3If+De7/AJR9S/8AZT/FP/pzmr7jr4X/AODeG9tNR/4J3/2hYXKTQTfEvxRJDNG2VdTqcpBB7givuigoKKKKACiiigAooooAKKKRnVBl2xQB8Wf8HEP/AChr+On/AGLNt/6X29exf8Eyf+Ud3wR/7Jbof/pFFXjf/Bw9LHJ/wRs+OgRs/wDFM2//AKX29ew/8Eypol/4J4fBFWfn/hVuif8ApFFQB7rRQCCMg0UAFFFFABXnsv8AyOPiT/sKxf8ApFbV6FXnsv8AyOPiT/sKxf8ApFbVURdT0KiiipGFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUj8rS0EZGDQBWmIWAkj+IfzFfIv/BTPxx4+8MX/gHQfA13q8cmpX91uj0cHfJJFFviV8D/AFe773tX17dootnz6VxHxK0LSdQ1LT9SvLSN7izWYWs+35owy4Yj6iinF+2pyv8AD+OhnKP7mUe5+c82lftU/C7wWukXer+KGtdW1UXlw9gGaS2umjl8qFePlh8z7w9xXtf7MOvfE74eeNda8R/tE+OJrex121MsS6xPthtrqO5kUQx54AMQBq1L+394KjtNRubf4fapdS2niKbTIbFVQTT+Qu+aYLnIRV55x14rtvDnij4TftcjWfC2q+EJbyx8N30OwahHhJWlgV1kXuRtcV3U/wB3TUezMZa1nPufLHxZ8NfGaH4peOPGl/perXfgvWtfLRzaDNI8l7p4bGAMnBBK7SuMj1r139kPWPGWlfHJvDHiux8Q+Zc/DnSlkmvo2+zw3EfmOVDHjcY5Iwe+4GvSPg1+0D8P/GPxe8Ufs2eGvC9xYzeA4Yo91wgWOVc7QYx1IHr0r177Pag+asahtuNwxmiK5Y28w3LEv/Hra8fwmo2WM/eFSSsPsttz/Ca574j+PfDnw18Iah448WagbbTdLs2uLuZVJ2qv0qg5Td/dNkUhMOOccV5vrP7THwn0CW8Or+Jo7f8As3wzHrV75y7RHayY2tuPHU9M1U1f9rH4T6L8H9P+NV1qN02k6qB/Z/lwmSWcc5wqgk/KGPHap5vdbKt7yieqEIy5x/CaW8K/aYv+uK/yrzc/tQfCQaZ4j1yLxMJLPwzYQ3WpzCM4Ecib128fN8vpXc6NrVj4k0nT/EemTeZa3+nw3NtJjG5HQMv6EUavYjqXsoD2o3Re1cz8Sfid4T+F9jYah4t1H7PHqeqQabafuy264mcKg46dxntmuZ1v9p/4SeGr7WtLv/EDR3Gg31laalGYS22W5OIl6c5PX0qYy5p8nUs9MxETv4zmhfLEowB/k15z8Xv2ivAfwWudJHjRb1Y9YkEdvc29m0kYJcIoYgELy1Yvgf8AbN+A/wAQZLeTwx4kkkWfxENFj3wtk3WCdvTgcdfaqupXS6EcvvI+oNP2/ZYWxz5Yq1VXTMfYoSp/gFWq4bWOzoFNMcZbcU5p1FAjw39qr9hD9nH9smXR0+PngGPWG0JpW0uR2KtB5n3sEdM4ryf/AIcXf8E7P+iJR/8AgXJ/8VX2QY1LbiKXYvpQB8bf8OLv+Cdn/REo/wDwLk/+Ko/4cXf8E7P+iJR/+Bcn/wAVX2TsX0o2L6UAfG3/AA4u/wCCdn/REo//AALk/wDiqP8Ahxd/wTs/6IlH/wCBcn/xVfZOxfSjYvpQB8bf8OLv+Cdn/REo/wDwLk/+Ko/4cXf8E7P+iJR/+Bcn/wAVX2TsX0o2L6UAfG3/AA4u/wCCdn/REo//AALk/wDiqP8Ahxd/wTs/6IlH/wCBcn/xVfZOxfSjYvpQB8bf8OLv+Cdn/REo/wDwLk/+Ko/4cXf8E7P+iJR/+Bcn/wAVX2TsX0o2L6UAfG3/AA4u/wCCdn/REo//AALk/wDiqbL/AMEMv+Cd8aZX4Kxj1/0qTn/x6vsvYvpWb4o8Q6J4W0qTWtf1a2sbOFS011dzLHGg92Y4FAHHfswfs7/C39lj4U23wb+DnhmPSNBsLiSS2sYvuq0jbnP4mvRNw9a8Uf8Ab2/Y0tppLef9pTworxsQwOrR9fzpP+G/P2L/APo5fwn/AODaP/GgD2zcPWjcPWvE/wDhvz9i/wD6OX8J/wDg2j/xo/4b8/Yv/wCjl/Cf/g2j/wAaAPbNw9aNw9a8T/4b8/Yv/wCjl/Cf/g2j/wAaP+G/P2L/APo5fwn/AODaP/GgD2zcPWjcPWvE/wDhvz9i/wD6OX8J/wDg2j/xo/4b8/Yv/wCjl/Cf/g2j/wAaAPbNw9aq6qC0KlHx83OO4xXjv/Dfn7F//Ry/hP8A8G0f+NB/b7/YuPX9pfwn/wCDaP8AxoA/KP8A4OBP+Cgvxj+E93+0B+wx8abCOLwb44+Gtne/DG/WHbunjurczwlv4icOfwr2D/ghb/wUG+Lv7V3xD0n4H+BbGOT4U/C34N6Hpt5qXk/Ncax9mi3KG/2RkY9q5H/g5q0H9jz9tr9iaT4hfDX45+Fb7x18P5lvNLgt9SjMt3auwWaEc89Q2P8AZNer/wDBC1f2Iv8Agn7+wP4X8Da18f8AwjH4q8QWqat4qk/tSLes8qgiM8/wLgfXNAH6d2rZt1LHtUm4eteJJ+35+xfsH/GS3hP/AMG0f+NL/wAN+fsX/wDRy/hP/wAG0f8AjQB7ZuHrSF1BwWrxT/hvz9i//o5fwn/4No/8asaP+2/+yR4l1aDRdA/aI8LXF1O22OKPV4gW/wDHqAPY1ZW+6a8+l/5HHxJ/2FYv/SK2rudOu7W4gE1vcxyRuA0bqwIZfUHuK4aU/wDFY+I/+wrF/wCkVtVR6i6noVFFFSMKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAozRSOcJmgCK8ci1fy+Wx8o968V/aY8WfG3w5PpUXws+G9prscvmG8a41EweRyNpHyNnv6V7PI2YySO+a8E/bO/aRb4A/8ACOx2nhZtUm1h7gSKswTyYYkDyPz1wueKqlKPtoxb1ZNT+G5LU8f1L/gnLeeK9VvPGesfHDWrfXtSv/Pa7h8tTZ28kPlzWqYXBVgcFsZOK17L4K/GP9m7Wdc8bfCvULnxl/wkDW0H/CO6leLbw6dFBEscbRsqEk7VA5rgvBv/AAUc8T3HhO4bxD8MZrjVhfs+nKJxGLvTVVpGn9BsTHHU4r1L9m/9rF/2hPGeseHIfBdzpdtYwrPpV5NJu+1wiaSJ2I7fMveuyPvcvmYy0qOHY2PhH8Ada0j4lal8e/GOvzS6zrWnpF/ZRZTHpylgzRKwUFxnua9W+wyf89F/76r4+8Xft7/FW0+MnjT4TeFfDmkyXem60mmaDLcLIqrJkjfN82GXaD93GD1r1b9nj9pHxl8XvGR0XxF4YtbG0fwdpuq2U0MhLTSzmWOY4zwoMe1foaIyvG/mTse53VhILaEeYn3fWuH+Pvw78S/Ef4O+IvAvhO60+PUdW0uW1t5tRhMkMe9SCxXIzge9drKSba25/hNRsA3BNUHMfK3jX9i/4+/EErZ+J/F3heS0k8B22j3qxaa6tc3UciuGb5/9X8qjb6Z5qSX9i/412fwU8P8AgLwx4z0CHVPDd3NPprT2DNbq1wkkc67d33FDLsGeMd6+pNnPBpvlZGN1Ty+613K5veT7HyP4m/4J+/F3XrPXPA8Hxkt9D0G70nTbPTptJs8XTtaQGLEzMSHBJOeBzivpr4V+BdS+H/wz8OeAtU1ptQuNF0K2spL6bhp2jQLuOPXFbzwRlcnsCf61Jd/8fMeP+eK/yo95bGZ5f+1X8I/iR8WfAmm+GvhprOkWV9D4isr17jVLUzKEgkEuFAIwxKjn0rxfVv8Agnx8bdDbxveeGfjrFrF54wu9PuYV8QWIljsmt7jzm27SCT/COeBX1s6KxDN/D0/KjyxnO6pjFxqc/UrmPLvE/wAHPGnxG0vRNO+Ier6bINJ1Rr2b7HbFROFjbyF5J4WRgx9Sq14h8Of+Cfvxt+GGmoPDfjDwykz+PrfW7hf7NfaYYlK5X5/llct8x6cdK+v/ACkwOen+NKE/eA575/WnyqN2uoub3kdvopmWC3jlILCNQ+Omcc4/GtOqmnIBbQsP+eYq3XEzsCiiigQUUUUAFFFFABRRRQAUUUUAFFFFABRRRQBDfZ8j5T/EP51+XHxo1jxr/wAFSv2n/F3gO68aahpvwX+GOsNpElrplwYm1/U4/wDXbmH8CNhfev1G1LP2N9px8vX04r8yP+CPdot1+xFovi+5T/TvEfiDWtT1Jz1ed9RnDE/98igDsNA/4J2/sn6JYLp8fwos5FUD55ssx49avf8ADAn7KX/RItN/75NezFMnOaCoH8VAHjP/AAwJ+yl/0SLTf++TR/wwJ+yl/wBEi03/AL5NeyYHY0cAZzR5AeN/8MCfspf9Ei03/vk0f8MCfspf9Ei03/vk17JketHGM5o06EuXLoeN/wDDAv7KX/RItN/75NH/AAwJ+yl/0SLTf++TXsYI3HmlcgHrR9m4+bS543/wwJ+yl/0SLTf++TR/wwL+yiOvwj03/vk17MFz/FTWUDqaFqNani8//BPz9ku6iaG4+D2mSRsuGVo8gj0py/sBfsoKu1PhFpoAXbgL2r2bgjrQigjrVSjbqK/vWPG/+GBP2Uv+iRab/wB8mj/hgT9lL/okWm/98mvZiuBndSYB6NUra4zxr/hgT9lL/okWm/8AfJrD8df8E0f2TPG2kNpb/DpLFmBVLnTpmikjJ7gjuOtfQIXLdaUJzkmpjLmA8F/4Jy/Gj4ufs3/tOX3/AATt+O/i+417S7jRm1f4X+IL5szS2aNtltJD/EycEe1fZ8ZJ8VeICf8AoJxf+kVtXw9+0dbppX/BRf8AZR8Q2Q8u6uPF2tafPIOrwNp5fafbcK+4v+Zs8Q/9hSH/ANIratI9Qe56JRRRUgFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAU2T7hp1BGRigCtIdsLAivDP2rf2ZvAP7QHibwff/EOwW9s/Dt5cXEVm33ZJGj2qT9OvvXut5DvtnXOM968i/aC+Onwm+DzaYnxO8f2Ojm8Mgs/tiyfvuQPl2Kf1xVU4w9tCct0TLSi4I8n8Z/AX9jTxppP9meJF8KtbaVqUMTrJq0Ci3nG7ZA/zfITk4U4J9KveKfgxeabCv/DMnifSfDWtQv5N5dG0S5zAzGXyiv8AASzF+cZBzXyvrXwH+Mk9zrUvhv4KXWpeF7jxJdXENvdXWLi7mubc+XdM4QZSFuVBGRu617B8AvGtn+zB4z8bzfHa0k8K6PqF5p7WWvavdSTRajdJapHLsAQlckEjtiuyLUeXyMZ/xHPud/e/su/s+/ErTbnwR8UbDQtY8SSwm48Q/ZZIo7iSZ2BMzRg7kDMuQSAK6PwR+yz8E/ht8Rx8U/CXg+O11tdHh0yO8T+GzjXakOOwXr9TXjf7L/gL4n2/7Ynjr4oT3U2peGPEdq1xZatNhorlCwMIh+UMoVeoJPbGK+rGt5iNxhbP+6amPuxtfqzNsml/49rX/dNU9Vu7ewt5L68uI4YYY9800zhVRQclix4AHU/Srk0EwhhAhb7v9015n+114b1TxL+zn4u0PSPDN5qt7daLNFa6fazGJ5pCMIAwHHzYJ9QKq6DXsdfN4/8AA1rdT6bN410mO6t7Vbq6hbVIlkityP8AWuu7Kof7xwMd6Zd+PfBWnaFF4v1Dxro9rpNwF8nVJtTiS3kz0Cylgh3dsHmvjH4l+CfjN8RtetLiz/ZzurG4sfBOnC6uo7/D6kVMfn2ch2cjapUfWtTx18EPib4v/Y5s/hpP8KdWsry11xtQ0Wws7xSLSOQzGOFt6EMkYdcjHOOMVPMjRU7q59ZXHxN+GumyXkGp/ETw/bnTsLeRT61An2ct9wOC/wApPbOM9q6A3MV60F5DNHIslujLJGwKuuODkcEe9fBvib4KfGjw5qvxM8QwfB248Qrr1pZx3ljeTL5c15iONpoDsyqoMsAS3PpX2l8J9HudK+Gnh3T5tPuoWt9DtoWiuG3PGyoAQx9fWquiEm3Y2tY1Cx0+KOfUL6O3jaZE8yaYIpLMFVeerFiAB3zWXaePfB90bn7H400eb7DP5GoeTqETC1lJACPhvkYn+E4JzXmf7ceh+K9S+HWgzeEPAmoa7qFj4w068htrG6MXliKYO7twdyhA/Hrivn3xX8J/iyg+JfiTR/gjq039r+J9HvtPtFvhG0k8UvzP8qAbAoPBByT14ojKLepbptH27c6jZ2NxDDc30MLXkmy3hklCiaXAYBf7xCqxwOymsw/FD4cNerZj4jaGbiS7+zJD/bMG4z94Au/Jf/Y6j0rwz9omT4m+Kdf+Gfibw98ItUe80vxDcQ3TrqG1bKF7UxiQjbhiXcEemD614Z8L/gn8fPBWiWPhHxZ8GbzXFufiAl3/AGtcXC/a7FUGZplKxjcGbYBnkDPJrOnU9tzK1rfiEqbhZn6k6eP9Gh/65CrNZehBzaW7SBlPlj5W6jjpWpXKbRlzIKKKKCgooooAKKKKACiiigAooooAKKKKACiiigCK9UPaSA/3D/KvzO/4I8jP/BP3wS3/AE9av/6crmv0yu/+PWT/AHDX5m/8Eef+Uffgn/r61f8A9OVzQB9PV4b+1t49+MWi+NPhx8OPgv4jtdJvvGHiC8srq9urUTBI4bOS44B/3K9yyPWvn/8AbC/Z7uvj98SPhXZ3FpdNo+jeJr+81qWwvGhljjbTpo42DLzy7BTyM5oX8SPqHRsw/gp+3fpM3hLStG+LYkuvFN5r2oaRaxaJb7xqDWkmx51xwByCfSvRLP8Aa5+EWpfEIfDdbu8tpJPtC22rTQEWc8kALzRLJ0MijJx6I2OlcqP2YtL8H/tEfC698AeFIYvC/hfQNZiuJuvlzz+XiQk8lmIOTXhPwt/Yh1DRPGeseF/FXw31GbVF1nXbrS/F02tyfZYluIpVR1hxgPtnA69CaHL/AGyS6WNKcfdVRneeMv26Lbxx8btE0b4Oa5qFnpY8F67qM7X1qy2915KAwToejoDuPr7V6lov7XngOytPDfhzXri+1jWdQ0e1vdSbR7QvHAs4+Uy4+4HPIH0r528K/BH49+JIvCHhm9+EMejw+DfhZrPh2e888Fry6lhCRMvyj5Gx6kgmut+BPw2+NP7OfxRvp5fhiNctfGWh6DCLoXA26TJb2y28yOCpyBs3DBHJrno8ywOu93+v+RNRc2O02sv0NbwL+3zrPxh+HV14o0HQ59AutM+Iw8P3E2pWu6O6hW8eDCdfnYLz6E16hb/tf/CG7+KEfww+13jSTatJpsOsfZ/9DlugPmt1k/vDBHpkGvCfD/wM+MNj4E174TJ8PfLaz+OEniKx1ETfJd2dxfm43KNv8Kn1603Rfgl8ZRY6P+ztfeAFhh0P4oNr8vjLzv8AXWouJJ1+Xbnc2/afm6CtY/xoR8jn5XKE436nuXgD9tj4L/Evx3a+AfD99qH2i/uri2sbyazZbee5gLCSJG7spVvbA4Jq38Y/2tPhd8FvEn/CO+K3vp5Le1W61SSxtTIun27HCvMf4ASefavHvhP+zl8RvDXhz4S2t74dhhuPDvxG1nUNWZW5itp5bp42zjncJFpv7R/w4+M1h8QPixp3hH4bL4itPir4Rh03S7xpgo02TynhbeCpyo3buCOla/ClLua6uTiuh7rrf7THwk0Gw8Salc63ILfwjDaS6tPAuQqThTEV9chxXnnhb9vnwrffFz4nfD/xT4R1TT9N+HNvFPda00P7t4GQOxPPXBGB3rxn4u/AD9ofwRoPjr4PeCPh8Nej8aaFoUFvrH2naLV7SJBKrrg5YmPg571r/tBeCfHfw+sv2i7698FM2j+KvA1vdWerxyDCSwwrG0TLjucc5/Cs43jUdR7JbCkrxse4aJ+3N8EPEWh674gS51Kzh0G2t726W+tTG01rO4jSZAeWRiwH51nftV/tteFP2cvh34o8U6Tot5rGp+GYdPmutLgX7sd2+2Ns/wA68Bvvgl8d/wBovwT4k8ZJ8O4vD5j+Eel6JpFi1xuW/uoZhdE5CjAI+VeDy1anjj4R/H/47+FfjF4i1X4Uf2RLr3hvRrfw/ps11vaaWzfc4J2jk44471rVd5K3a/3kwjKO59kfDrxdD4/8IaZ4xg0+6s1vrNZltrqPY6bh0K9iK3q574ZavqereB9In1rw3JpN01jGJrGRwxhYKAVyAP5V0NKVuhofPf7SsSy/t7fsm7v+igax/wCmxq+2WAXxf4iA/wCgpF/6RW1fFH7SP/J+37Jv/ZQNX/8ATYa+2JP+Rw8Rf9hSL/0itqIbMHuj0KgHNIeTil6VIBRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFACMu5dprI8Q+BvCPigRnxH4csr7yf9V9qtVk2fTI4rYoYbhigGUo9JsIYlgito1RV2qqoMAelVdY8EeFPEtqLLxD4fs76FW3LHdW6uob1wRWr5fvSqu2nzSJUSpaaBpFhbR2dlp8MMMS7Y444wFUegFTf2fbYx5S/wDfIqaileRRD9gtv+ea/wDfIpDp1qesK/8AfIqeii8gKbaXZBgRbp8v3fkHFOFhbAYES/8AfIqwUyc5o8v3ouwIDpdi4y1un/fIpW061VQqxL7fKKsAYGKRl3d6LyCyKj6daPjfCpwcj5R1oaxs+EaBDzn7oq15fvTWiOc0b7jVupXlsLNjkwJ/3yKVNLssj/Rk46fKOKsmIHqaco2jFF5bIjqQpbxRspTsc1NTQmDnNOoKCiiigAooooAKKKKACiiigAooooAKKKKACiiigCO7/wCPWT/cNfmb/wAEef8AlH34J/6+tX/9OVzX6ZXf/HrJ/uGvzN/4I8/8o+/BP/X1q/8A6crmh6AfQ/jXxf4b+H/hnUPGvi/VY7HS9NtzLfXU33UUDP4njAA5JIrm/hD8evhn8ddGm134Z63JdQ2dz9muo7q1aC4ifbuVXjcBl/gYZHINec/8FMtatbL9krXvDCWEl5qvia8sdJ8N2kb7PN1KS4R4AW6BdyZPsCPavKP+CbXxgguPG2r/AAx8b6LND8TNcjutT+INp5e1NJkt/JtoI1H9x1O5W7hT0oi+ZNIly5Wj68j8Z+DpvFLeBY/Femya1DbefLo63ifaUgzxIY87tue+KdF4v8J3niSbwgPEFi2sQW6TTaStwv2iGNs7XaPO4KR0OMV8P/G7WV+C/wDwUh1H9qS8uvJ0zQ5NJ0LxUzMdsWn3sLgSnthJVU/8CrlfDPxh1P4c/Hrx9+2zNb3F5c+IPhjfanptsx62sWpfZ7XaCcfdAI6ferJS95Lr/T/Q0nGTi+V7tfml+p+jkyySD5CM7j97uf8AIrndP+Lfw7vtc1Dw3b+KbVbzSdWTTryOaQLi7ZPNWIZ6sVINfH3h79r39qu18E32jeINB1eyudc8RaLYeGfEXiCwii2G8LecoVJHyI9vBJGd3tWR4TvPGvgb4geKovibeWuuapD+0RpySXwi271/sxTvAHRwvX3repy0+ZdF+tv+CZynbll30/r8D77iSMKAuPl9KxvFfjjwv4Lk09fE+qx2v9r6kLLTg/8Ay8TMpIT68E182+Bfj58ZPEPwC1H9qrxP8RLfT9B1q0uDpOmJahmtG88pCIxn5pDgDBxkt14rzHw5+0Z48+N0uj+HPH9lqP2zwf8AHCxsoZ9UtUhupImtXlTeiuwBG7HBPFTCPNiIwW4Vqclh5Ndz7u8Ra5pPhfQLzxNrl0tvZ6fatc3lwVz5caLlm/AA1T8I+PPCHjaGKfw3r1rdefYQ3qRxyDzBBKuY3ZPvKGHrXzF+2r8SfjR4w8T/ABC+B/wz8TWuj6f4b+GD6vqUk0XmfbfP81FjyOihY259TXKeBfjFP8LvDupajYXUem36/CvwqlpfW9v5txc3E8IUQxqSNz9doyB71FOXtJTfRW/Hm/8AkfxNJR5eSPV/5L/M+42iQnJ/i/wxVLxN4a0PxboN14X8R6bFeWN9bmG7tpkDLLGcZUjuOK+B7X/goV8W73wBZ+HPFHim68M3j+PNY0W48RahpbSXEcVpAJYt8MO8hnLhTgnGOa6Lwz8c/wBsvxR4V8L6Df8AjiLStauvhndeJ9WmmsfnaRHJjTbn5QVABHvTqfwn52X32ZHNr9/4Ox9uWmn2VhbwWtpCscdvGqQxquAqqu0D8BxUl0EhhaYnAVefpnNfIdh+0v8AH7466Wbj4aeJ7Xw/P4f+Hdpr2piSHf8AbLmaJ5AmOyfu8HPqareCP2yPip45+KHhHVPGF7J4W8LeJtBsZ9J8+0D2mq3E8X763Mik+XKGb5QwAIHBzxVN81RrskOL5qal3Prvw/rmi+ILNdR0HVbe8t2yiTW8gZSVJBHHoeK0K+C/DX7R3xN+H/grwTLpkj+G/Bl1fakdS1+z08zQW90Lx1SK4A+ZI2RfvAMMnnFfdGjXUV7p9tdw3iXCzQCTzo87WzyCM+2D+NJO4HhP7SP/ACft+yb/ANlA1f8A9Nhr7Yk/5HDxF/2FIv8A0itq+J/2kf8Ak/b9k3/soGr/APpsNfbEn/I4eIv+wpF/6RW1XDZg9z0IUUCipAKKKKACiiigAooooAKKKKACiiigAooooAM0U1l3c5p1ABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFAEd3/wAesn+4a/M3/gjz/wAo+/BP/X1q/wD6crmv0v1It9jZVON3FfmB/wAEltRXw5+zpq3wE1ndDrXw58b61pGq2b8SRf6dLLGxHoySAg96T1QHu/x++DPhn49+AJPh94saWOH7XDe2d1BJsktbqF98UqN2YNXnvwd/YZ8AfCP4rr8e9M8SaldeKLq3kt9b1S7ugzajGduyOQAAHbjjGMV7psEpZWHIx+FKtuq8gDIxt9qVNcrdyZK55P8AFL9j34WfF+y8eaf4wtpp4fH2n21nrEfm4/dQZ2Ohx8rA4/KoL79i34OaraQ6Rdaexsofh+3hRbXzMAWZkEi/8CyCc+teviEjgtxjB96SS3DtvY/NQo7v+tmv1Lv7y/rqn+h4gP2I/C198Mrz4b+KPG2tan513aT2N/cXW6Wwa25h8ogAAqSecHPejwf+w58NPCUMpk1bUb66n8ZQ+J7y8vLnc897HCIQSfQjqOK9wMTMu0t70eT2B9/1B/pSlGUozT62E7O3kfPd3+wF8MZNC1bwfH4j1pdBvjJPY6Mtz+506VpjLuiGOzkkA5xVj4c/sI/DfwLcNq02u6lqWo3Xi6HxHeX19dbnuLuKHylOcDC47e1e9tbBvvfN0xntSG1UgjHBPT8aun7subqFR80bf1seN/HP9jbwF8c/FL+KtQ1/U9NmvNCbR9WTS7ryxqNmWZvLfjnBY4xjrWf4q/YQ+FniXQ30uS+vLWaPS9Ns7O8gn+e0awBEDj/aA4z3r3Rbcxn92eP4fal+zhn8xuuKLJPTuvwv/mPXkXdX/Jf5HxF8Z/2A5PAl14Z1LwT4K1Hx5pcevanq2v2sutG1upru6ijTzvN2n5cxklcfxV6t+y7+yvrOieArDXfjPdyN4jbQdQ0d4FujL9l0+4mdo4NxGXZEKru9ulfRLRseh+8xLe+TmkWHD+YOGo3lLtpb5JIVvdjfz/FtngOtfsF/Dq/tdOs/DviTWNHW18PR6FqTWNxtN9ZoGCpIf4sB2GRg81oa9+w58OdW8SaBqU2sX8ei+G5rSWz8PCcfZVltk2xPtxnI46HBPWvbFtIwxcLg9vanLCVHDcnqfXp/hVKyu+4baI8E8SfsG+B/EHg6z8BxeLNVs9IjMwvtNjuh5d7HJO0xRgQe7HkYOK9z0bTLLSLGHT9O+WGGJY4U/uqqgAfpVgxEtvzzuyvtSNiEqSOTwzeg9azirAeBftI/8n7fsm/9lA1f/wBNhr7Yk/5HDxF/2FIv/SK2r4X+L1//AMLC/wCCpf7Nvwv8Ns1xfeFptZ8T63HH1trU232dC47bmPHrX3MGLeLPELH/AKCkX/pFbVpDqD3PRDntQDkUUHPapAKKAc0UAFFGaM0AFFN3jrTtw6mgAooz2ooAKKKKACiiigAooooAKKBRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFAEV3gxfMP4hX57ftrfsOftNfCX9oi+/ba/YPWwvtR1qKNfHngPUGMcOrhFx50ZAIE2PUc+tfoZLH5q7c981EbJS27d3zQB+YGgftqft2SWH/E7/4JaeNY7pW2yeVrFuVOO45q9/w2j+2t/wBIvPHP/g3t/wDGv0yFpt6OaPsx/v0Afmb/AMNo/trf9IvPHP8A4N7f/Gj/AIbR/bW/6ReeOf8Awb2/+Nfpl9mP9+j7Mf79AH5m/wDDaP7a3/SLzxz/AODe3/xo/wCG0f21v+kXnjn/AMG9v/jX6ZfZj/fo+zH+/QB+Zv8Aw2j+2t/0i88c/wDg3t/8aP8AhtH9tb/pF545/wDBvb/41+mX2Y/36Psx/v0Afmb/AMNo/trf9IvPHP8A4N7f/Gj/AIbR/bW/6ReeOf8Awb2/+Nfpl9mP9+j7Mf79AH5m/wDDaP7a3/SLzxz/AODe3/xqOX9tP9tVMs3/AAS+8dcL21e3/wAa/Tb7Mf79R3ulxX9nNYzu2yeJo5NrEHaQQcEdDzQB+ZK/tt/tnTR+fB/wTJ8cSR8H91rFscg8DBzg81Y/4bR/bW/6ReeOf/Bvb/41+iXw1+Ffhr4UeDrXwN4Ua7+w2alYftl5JPJgtu5eQljye5roPsx/v0Afmb/w2j+2t/0i88c/+De3/wAaz9f/AGuP+CiPiSJfD/w4/wCCZXiCy1a7BjttQ17XoVtbcnozhQxYZ/lX6hfZj/foa03feegD42/4JsfsF/Eb4E+JfEf7Tn7UPiiDXvix40jSPUri2U+RpdqvK2cOedoz14zX0VGNvirxAM5/4mcXP/blbV332JdxbPU5NUP+EO0j7ZdX/lN5l5Msk53HlhGkYP8A3yi/lVRdgNaiiipACtN2n+9TqKADGetGKKKADA9KCAeooooANvO6kwwH3qWigBAWBwaTfxkinUdeooAM9qKCoNAGO9ABQTgU1SSDzSgd6AFHAxRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQB//9k=)

Fig. 3.2 Architecture Diagram of Voice Data Analysis using Decision Tree

The predicted output for voice data analysis based on classification is with an accuracy of 88%.

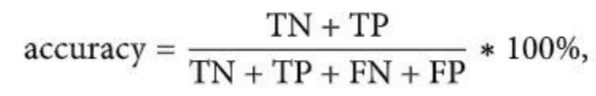
|  |  |
| --- | --- |
| **Model Name** | **Accuracy** |
| Decision Tree | 88% |

Table 3.1 Accuracy of Decision Tree

#### SPIRAL DRAWING ANALYSIS using RANDOM FOREST CLASSIFIER

#### Performance Analysis

Evaluation metrics were used to evaluate the performance of the random forest classifier. One of these measures is through the confusion matrix, from which the accuracy is extracted by computing the correctly classified samples (TP and TN) and the incorrectly classified samples (FP and FN), as shown in the following equation.



In this method, input is spiral drawings from the machine learning repository. These drawings are subjected to resizing and histogram equalization. The images from the spiral drawings were resized to 256px width and 256px height. The spiral drawing images that are collected were lack in contrast and brightness. Therefore, contrast enhancement and adjustment performed over all the images using histogram equalization.

Random Forest Classifier is applied to these enhanced images and Parkinson’s disease prediction is made.

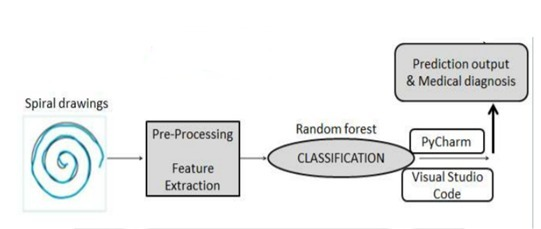


Fig. 3.3 Architecture Diagram of Spiral Drawing Analysis using Random Forest Classifier



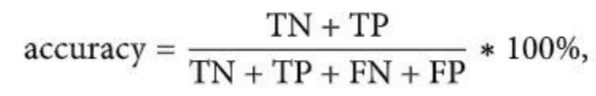
Fig. 3.4 Spiral Drawing of Healthy Person Fig. 3.5 Spiral Drawing of Parkinson’s Patient

|  |  |
| --- | --- |
| **Model Name** | **Accuracy** |
| Random Forest using spiral drawings | 83% |

Table 3.2 Accuracy of Random Forest Classifier

#### 3.3 Parkinson’s Disease Prediction using KNN

#### Performance Analysis

Evaluation metrics were used to evaluate the performance of four classifiers. One of these measures is through the confusion matrix, from which the accuracy is extracted by computing the correctly classified samples (TP and TN) and the incorrectly classified samples (FP and FN), as shown in the following equation.

The dataset used consists of a range of biomedical voice measurement with 195 samples of features from 31 people, 23 with Parkinson’s disease (PD) and 8 of them are the control group. The data set has about 75% of cases suffering from Parkinson disease and 25% of cases which are healthy.

Feature importance analysis is done which can provide insight into the model. Most important scores are calculated by a predictive model that has been fit on the dataset.

This paper implemented four different algorithms out of which the one with higher accuracy is determined. These algorithms are:

* Logistic Regression
* Decision Tree
* SVM (Support Vector Machine)
* KNN (K-Nearest Neighbor)

|  |  |  |
| --- | --- | --- |
| **CLASSIFICATION TECHNIQUES** | **TRAINING ACCURACY RATE** | **TEST ACCURACY RATE** |
| Logistic Regression | 0.88 | 0.79 |
| Decision Tree | 1.0 | 0.9 |
| SVM | 0.89 | 0.92 |
| KNN | 0.94 | 0.95 |

#### Table 3.3 Machine Learning Models and their accuracies

#### 3.4 Comparison of Existing Models

**Decision Trees**

**Advantages**

Simple to understand and to interpret. Trees can be visualized.

Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.

**Disadvantages**

Decision-tree learners can create over-complex trees that do not generalize the data well. This is called overfitting. Mechanisms such as pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.

Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.

**Random Forest Classifier**

**Advantages**

It reduces overfitting in decision trees and helps to improve the accuracy

It is flexible to both classification and regression problems

**Disadvantages**

It requires much computational power as well as resources as it builds numerous trees to combine their outputs.

It also requires much time for training as it combines a lot of decision trees to determine the class.

Due to the ensemble of decision trees, it also suffers interpret ability and fails to determine the significance of each variable.

**Logistic Regression**

**Advantages**

Logistic regression is easier to implement, interpret, and very efficient to train.

It can easily extend to multiple classes (multinomial regression) and a natural probabilistic view of class predictions.

**Disadvantages**

Logistic Regression requires average or no multicollinearity between independent variables.

The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables.

**KNN**

**Advantages**

No Training Period: KNN is called Lazy Learner (Instance based learning). It does not learn anything in the training period. It does not derive any discriminative function from the training data. In other words, there is no training period for it. It stores the training dataset and learns from it only at the time of making real time predictions. This makes the KNN algorithm much faster than other algorithms that require training e.g., SVM, etc.

Since the KNN algorithm requires no training before making predictions, new data can be added seamlessly which will not impact the accuracy of the algorithm.

**Disadvantages**

Does not work well with large dataset: In large datasets, the cost of calculating the distance between the new point and each existing points is huge which degrades the performance of the algorithm.

Does not work well with high dimensions: The KNN algorithm doesn't work well with high dimensional data because with large number of dimensions, it becomes difficult for the algorithm to calculate the distance in each dimension.

**SVM**

**Advantages**

SVM works relatively well when there is a clear margin of separation between classes.

SVM is effective in cases where the number of dimensions is greater than the number of samples.

**Disadvantages**

SVM does not perform very well when the data set has more noise i.e. target classes are overlapping.

In cases where the number of features for each data point exceeds the number of training data samples, the SVM will under perform

#### 3.4 Data Collection

The datasets are taken from Kaggle (a website that provides various datasets) for Parkinson's disease.

Parkinson's Disease Dataset:

1. Number of samples: 195

2. Number of attributes: 23

3. Number of Parkinson's patients: 23

4. Number of non-Parkinson’s patients: 8

#### Description of Dataset:

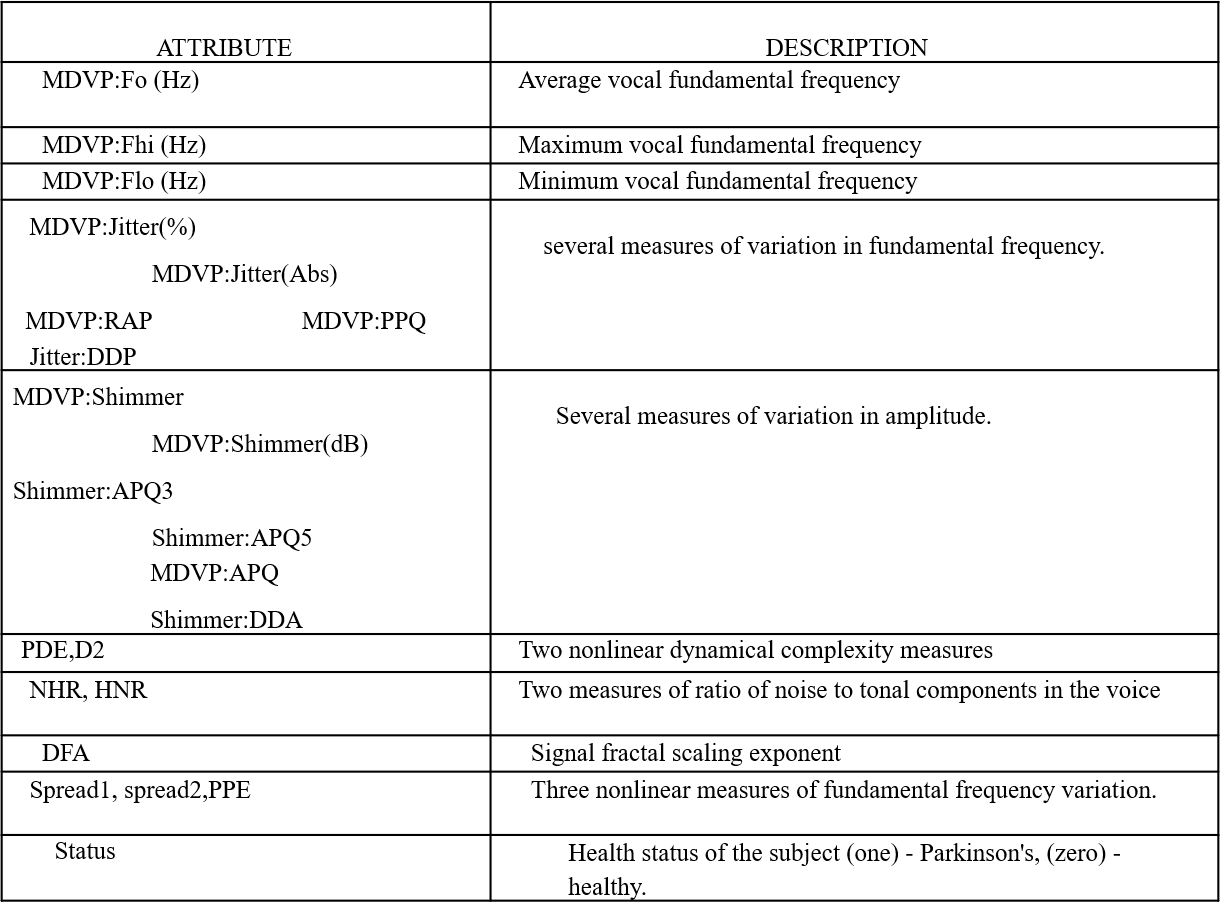


Table 3.4 Description of Dataset

# 

# CHAPTER 4

## **PROPOSED SYSTEM**

#### XGBOOST ALGORITHM

# XGBoost stands for eXtreme Gradient Boosting and it’s an open-source implementation of the gradient boosted trees algorithm designed for speed and performance. It has been one of the most popular machine learning techniques in Kaggle competitions, due to its prediction power and ease of use. It is a supervised learning algorithm that can be used for regression or classification tasks.

# XGBoost classifier is a Machine learning algorithm that is applied for structured and tabular data. XGBoost is an extreme gradient boost algorithm. And that means it’s a big Machine learning algorithm with lots of parts. XGBoost works with large, complicated datasets. XGBoost is an ensemble modelling technique.

# How XGBoost works?

# To understand XGBoost first, a clear understanding of decision trees and ensemble learning algorithms is needed.

# Gradient boosting

# **Boosting**is an **ensemble method**, meaning it’s a way of combining predictions from several models into one. It does that by taking each predictor sequentially and modelling it based on its predecessor’s error (giving more weight to predictors that perform better):

# Fit a first model using the original data

# Fit a second model using the residuals of the first model

# Create a third model using the sum of models 1 and 2

# **Gradient boosting** is a specific type of boosting, called like that because it minimizes the loss function using a **gradient descent algorithm.**

# Decision trees

# Decision trees are arguably the most easily interpretable ML algorithms you can find and, if used in combination with the right techniques, can be quite powerful.

# A decision tree has this name because of its visual shape, which looks like a tree, with a root and many nodes and leaves. Imagine you take a list of titanic survivors with some information such as their age and gender, and a binary variable telling who survived the disaster and who didn’t. You now want to create a classification model, to predict who will survive, based on this data. A very simple one would look like this:

# As you can see, decision trees are just a sequence of simple decision rules that, combined, produce a prediction of the desired variable.

# .

# Fig. 4.1 Decision Tree Classifier

# Advantages of XGBoost Algorithm in Machine Learning

# XGBoost is an efficient and easy to use algorithm which delivers high performance and accuracy as compared to other algorithms. XGBoost is also known as regularized version of GBM. Let see some of the advantages of XGBoost algorithm:

# Regularization: XGBoost has in-built L1 (Lasso Regression) and L2 (Ridge Regression) regularization which prevents the model from overfitting. That is why, XGBoost is also called regularized form of GBM (Gradient Boosting Machine). While using Scikit Learn library, we pass two hyper-parameters (alpha and lambda) to XGBoost related to regularization. alpha is used for L1 regularization and lambda is used for L2 regularization.

# Parallel Processing: XGBoost utilizes the power of parallel processing and that is why it is much faster than GBM. It uses multiple CPU cores to execute the model. While using Scikit Learn library, thread hyper-parameter is used for parallel processing. Thread represents number of CPU cores to be used. If you want to use all the available cores, don't mention any value for thread and the algorithm will detect automatically.

# Handling Missing Values: XGBoost has an in-built capability to handle missing values. When XGBoost encounters a missing value at a node, it tries both the left- and right-hand split and learns the way leading to higher loss for each node. It then does the same when working on the testing data.

# Cross Validation: XGBoost allows user to run a cross-validation at each iteration of the boosting process and thus it is easy to get the exact optimum number of boosting iterations in a single run. This is unlike GBM where we have to run a grid-search and only a limited values can be tested.

# Effective Tree Pruning: A GBM would stop splitting a node when it encounters a negative loss in the split. Thus, it is more of a greedy algorithm. XGBoost on the other hand make splits up to the max depth specified and then start pruning the tree backwards and remove splits beyond which there is no positive gain.

# Implementation Steps:

1. Load all the libraries
2. Load the dataset
3. Data Cleaning & Feature Engineering
4. Tune and run the model

# CHAPTER 5

## **CONCLUSION**

1. **CONCLUSION**

**Voice Data Analysis using Decision Tree**

In this method, the voice data available at UCI machine learning repository is used. The algorithm used is Decision Tree Classifier. The accuracy of this model is 88%.

**Spiral Drawing Analysis using Random Forest Classifier**

In this method, the spiral drawing data available at machine learning repository is used. This system predicts by integrating spiral drawing inputs of normal and Parkinson’s affected patients. The accuracy of this model is 83%.

**Parkinson's Disease Prediction using several Machine Learning Algorithms**

In this solution, the algorithms used are Logistic Regression, Decision Tree, KNN and SVM. The KNN algorithm has the most accuracy compared to all other existing solutions. The accuracy of KNN model is 95%

# CHAPTER 6

## **REFERENCES**

1. **REFERENCES**
2. National Institute for Health and Care Excellence (NICE). Parkinson’s disease: diagnosis and management in primary and secondary care. NICE clinical guidelines 35. June2006.Available http://www.nice.org.uk/guidance/cg35/resources/guidanceparkinsons-disease-pdf.Accessed April 28, 2015.
3. Parkinson J. An Essay on the Shaking Palsy. London: Sherwood, Neely, and Jones;1817: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4517533/pdf/ptj4008504.pdf
4. <https://easychair.org/publications/preprint_open/NQlD>
5. Early diagnosis of Parkinson’s disease using machine learning algorithms Zehra Karapinar Senturk Duzce University, Engineering Faculty, Department of Computer Engineering, 81620 Duzce, Turkey
6. A. Tsanas, et al., "Novel speech signal processing algorithms for high-accuracy classification of Parkinson's disease," IEEE Transactions on biomedical engineering, vol. 59, pp. 1264-1271, January 2012.
7. <http://ijariie.com/AdminUploadPdf/EARLY_DETECTION_OF_PARKINSON_S_DISEASE_USING_MACHINE_LEARNING_ijariie11591.pdf>