

Machine Learning

Machine Learning Assignment

**Module Co-ordinator: Besher Alhalabi**

Name: Muhammad Nawab Anjum

ID: 21171479

Email:Muhammad.anjum2@mail.bcu.ac.uk

Table of Contents

[Machine Learning Subfields 1](#_Toc90027400)

[Domain Description 2](#_Toc90027401)

[Problems Definition 3](#_Toc90027402)

[Scenario 1 3](#_Toc90027403)

[Dataset Description 3](#_Toc90027404)

[Dataset Exploration (pre-processing and wrangling) 3](#_Toc90027405)

[Main Python libraries that I have used for this project. 4](#_Toc90027406)

[Data Cleaning 5](#_Toc90027407)

[Different Approaches to overcome the Outlier 6](#_Toc90027408)

[Data Visualisation 6](#_Toc90027409)

[Histograms 6](#_Toc90027410)

[BarPlot 7](#_Toc90027411)

[Box Plot 7](#_Toc90027412)

[Pie Chart 8](#_Toc90027413)

[Feature Selection 9](#_Toc90027414)

[Coloration b/w features 9](#_Toc90027415)

[Correlation Analysis. 9](#_Toc90027416)

[Experiment and Evaluation 10](#_Toc90027417)

[Supervised learning 10](#_Toc90027418)

[Analysis and Results 10](#_Toc90027419)

[Conclusions 10](#_Toc90027420)

[References 11](#_Toc90027421)

[Appendixes 11](#_Toc90027422)

# Machine Learning Subfields

Machine learning and artificial intelligence is most advanced technologies. Machine learning is a branch of AI artificial intelligence and computer science which focuses on the use of data and algorithms to imitate the way that human lean, gradually improving its accuracy. (IBM).Machine learning algorithm is vastly used to train data and get exact result according to our requirements. Most popular Google search engine used machine learning algorithm. Social networking sites also used machine learning algorithm, Facebook, Twitter, YouTube, Instagram, and email spam filter is also used to save time. (Das & Dey, 2015)

There are many machine learning techniques that are as follows:

1. Machine learning (Supervised & Unsupervised Learning)
2. Inductive Deductive Learning & reinforcement learning
3. Deductive Transductive Learning algorithm
4. Multi task and active learning
5. Transfer and Online learning

Mostly the supervised and unsupervised technique is used in our daily lives. Some other learning algorithm are being used for different purposes and have different perspective. Deep learning and its other part is also a machine learning artificial intelligence neural network part. Artificial algorithm is sub domain of other domain and its work together to perform task. The classification of image processing using deep learning algorithm either the person is male and female, face recognition and image labeller. Artificial neural network can train large set of images datasets and its easy method to identify images. (Machine Learning Mastery, 2021)

# Domain Description

Mobile apps are one of the fastest growing segments in the downloadable apps market. Of all the markets, we choose the Google Play Store because of its growing popularity and its recent rapid growth. One of the main reasons for this popularity is that about 81% of applications are free. With titan companies like Samsung, LG, Motorola and HTC all launching Android phones, it soon became the most popular mobile operating system, reaching more than one billion active users by 2014.

As Android grew, so did Google's control over the operating system. Initially, manufacturing partners were able to customize most of the platform to their liking; however, Google has added more mandatory services and conditions each year, ensuring benefits for its own app package.

Google Play has grown tremendously over the last decade, reaching $ 38.6 billion by 2020. By 2020, more than 2.9 million apps were available in stores, downloaded 108 billion times.

Developers and consumers play a critical role in determining how industry impacts affect future technologies. However, the lack of a clear understanding of the inner workings and functionality of the popular software industry affects developers and users alike. In this article, we will try to explain the strengths of the Google Play Store and how we can use various configuration files for prediction purposes.

In this article, I provide a longitudinal study of the metadata for Google Play applications, which provides unique information that is not available with a standard approach to take a single application. Use the feature longitudinal analysis of the application to determine whether or not an application will be successful. Our analysis is divided into four phases: data extraction, data purification, data visualization, and application of different models. I will first collect data from the Kaggle website. In the next step I will try to delete the data from the dataset to reduce the error rate. Once the dataset is complete, we try to analyse the dataset using different maps and replace the unnecessary things in the dataset. In the last step, I will use different supervise and unsupervised algorithms on the data set to find out which offers the highest percentage accuracy. Finally, I will present the results of the analysis to provide a clear picture of the relationship between interests. The relevance and future directions of research are discussed in detail in the last chapter, Conclusions and future papers.

# Problems Definition

## Scenario 1

App analysis will be used to ﬁnd whether an app will be successful or not?

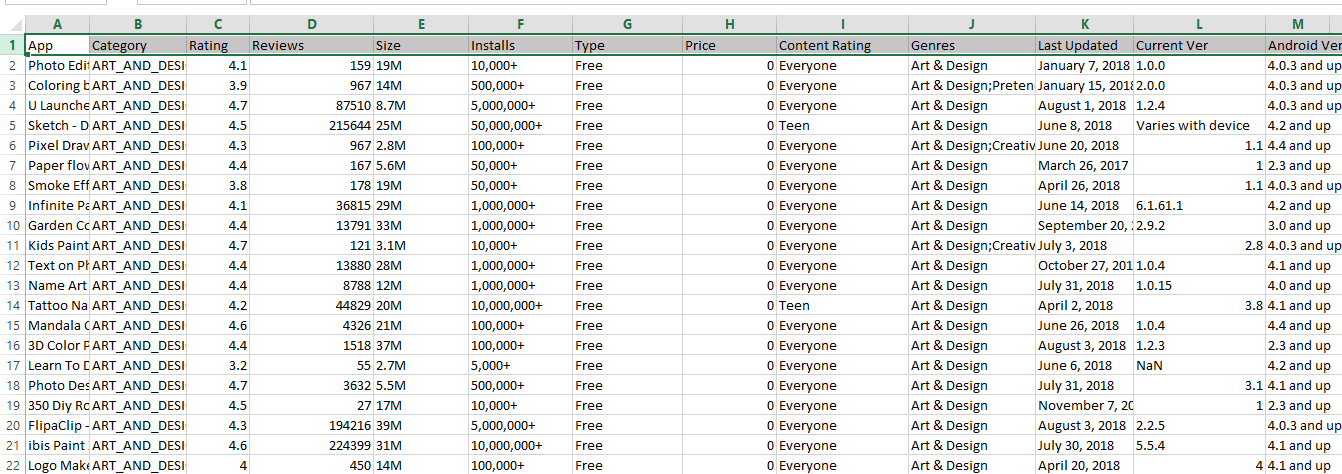
# Dataset Description

**Oxford Dictionary** defines a dataset as **“a collection of data that is treated as a single unit by a computer”.** This means that a dataset contains a lot of different data but can be used to train algorithms for the purpose of finding predictable patterns throughout the dataset.

In my case I have download the dataset from the [Kaggle](https://www.kaggle.com/lava18/google-play-store-apps/code) this is a web scrapping repository where [Kaggle](https://www.kaggle.com/lava18/google-play-store-apps/code) user contribute such kind of data. My Play Store apps dataset has total of 10841 rows and 13 columns.

The columns of the dataset are as follows:

|  |  |
| --- | --- |
| **Feature** | **Description** |
| App (Name) | The name of the App on the play store |
| Category | Category that app belongs to like Beauty, Business etc |
| Rating | Overall user rating from 1-5 |
| Genres | Shows what genre a particular app belongs to.e.g fashion, Music etc. An app can belong to more than one genre |
| Current Version | It is build version |
| Reviews | Number of user reviews |
| Size | Size of each app, how much memory they consume like 35MB etc. |
| Installs | Number of installation for the app |
| Last Updated | When this App is being updated on the store |
| Type (Free/Paid) | App is paid or free, Price of each app, 0 for apps that are free |
| Android Version | Mobile compatibility |
| Content Rating | (Everyone/Teenager/Adult) |



Sample Dataset

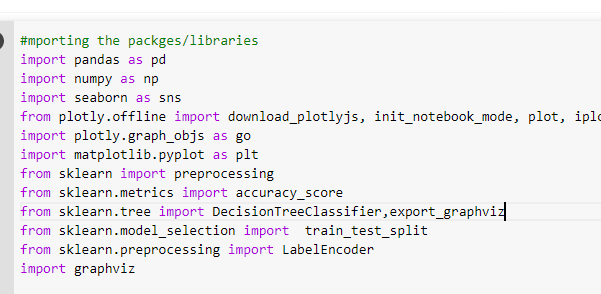
# Dataset Exploration (pre-processing and wrangling)

Most of the time, the data come with garbage/waste values, which need to be addressed before affecting performance. Trained models that predict outcome. Several steps are used to pre-process this data. Reprocessing is important into transitioning raw data into more desirable format. Undergoing the pre-processing process can help with completeness and compellability. For instance, you'll see if certain values were recorded or not. Also, you'll see how trustable the info is. It could also help with finding how consistent the values are. We need pre-processing because most real-world data are dirty. Data can be noisy i.e. the data can contain outliers or simply errors generally. Data can also be incomplete i.e. there can be some missing values.

In this Section, I will discuss more about Google Play Store Apps Based on the Given Information, using the notebook I have created to understand my dataset.

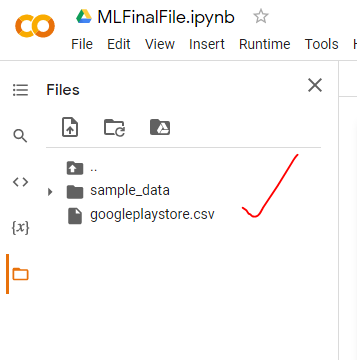
## Main Python libraries that I have used for this project.

* **Pandas**: The library is highly optimized for performance, with critical code paths written in python or C. It deal with DataFrame,time series data, reading and writing data between in-memory data structures and different file formats.
* **Scikit Learn**: This machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines. Also ,it can work with other libraries and packages like Numpy.
* **Graphviz:** Graphviz is an open-source python module that is used to create graph objects which can be completed using different nodes and edges. It is based on the DOT language of the Graphviz software and in python it allows us to download the source code of the graph in DOT language.
* **NumPy:** Is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
* **Matplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.
* **Seaborn:** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.



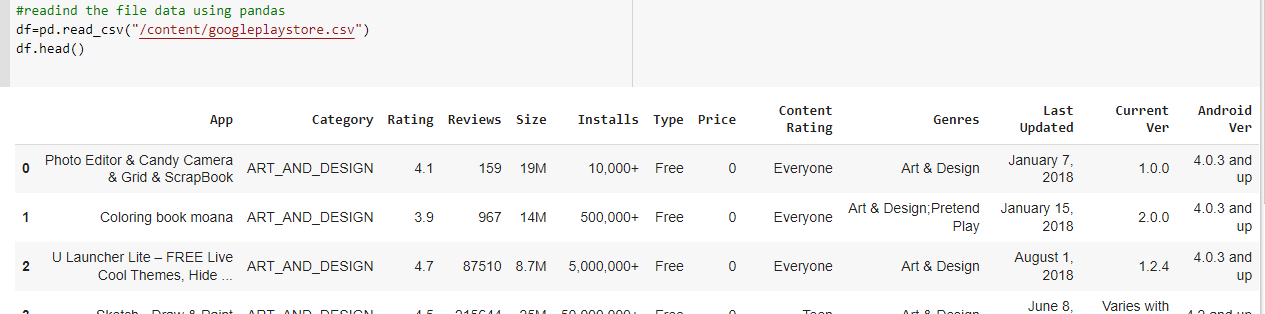
Libraries importing

The first step for any data science project is to enter your data. Often you will be dealing with data in Comma Separated Value (CSV) files. Now next main task is to read the dataset using **Pandas** Read**\_**CSV It is an easy way to read the CSV file data as a dataset. Before you can use Panda to import your data, you need to know where your data is in your file system and what your current task list is. I’m my case I have upload the CSV file into local directory of the **Colab**



df=pd.read\_csv("/content/googleplaystore.csv")

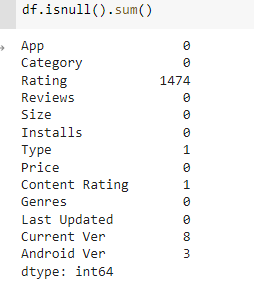
After reading the file, just checking that file is read or not by printing the head of the dataset.



Now I’m successful to reading the file, next step is star to clean the imported data.

## Data Cleaning

Checking the total null values, I’m using the isnull.sum() method this method will return the sum of missing values



In the above picture we can see that Rating field has 1474 Null values type has 1 and content rating 1 as well as android ver 3. Category has 1.9 that is considered as a mislabelled ,so we have to remove this using the below code

mislabel = df.loc[df["Category"] == "1.9"]

df = df.drop(int(mislabel.index.values),axis=0)

print(df["Category"].unique())

Also Rating has some null values so removing the null values from the dataset.

#dropping null values

df = df.drop(df[df['Rating'].isnull()].index, axis=0)

## Different Approaches to overcome the Outlier

* **Mean:** If we have a sample of numeric values, then its mean or the average is the total sum of the values (or observations) divided by the number of values
* **Median:** The median of a sample of numeric data is the value that lies in the middle when we sort the data. The data may be sorted in ascending or descending order, the median remains the same.
* **Mode:** The mode is the most frequent observation (or observations) in a sample. If we have the sample [4, 1, 2, 2, 3, 5], then its mode is 2 because 2 appears two times in the sample whereas the other elements only appear once.

## Data Visualisation

Data visualization in python is perhaps one of the most utilized features for data science with python in today’s day and age.

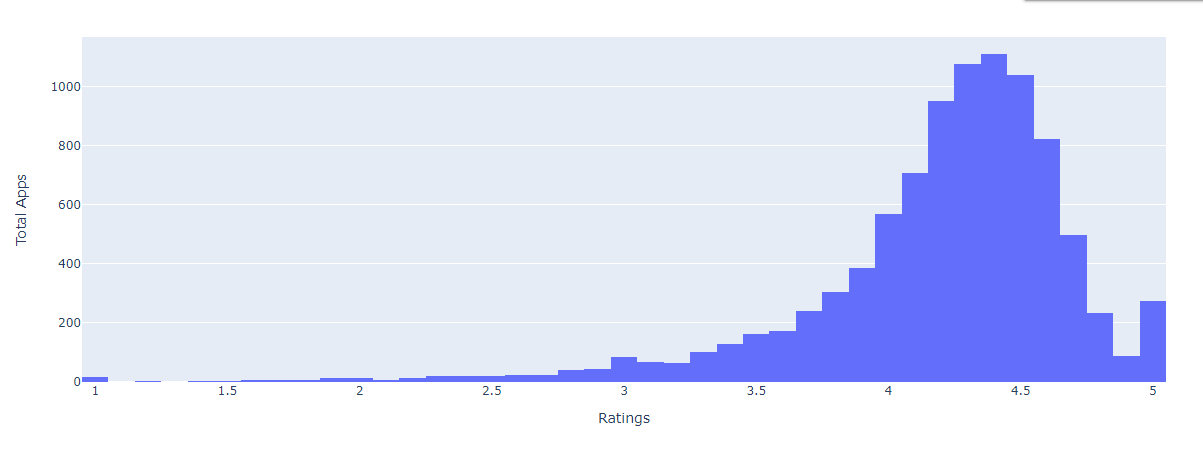
A good visualization must follow the **Gestalt Laws.**

* **Law of Proximity**: We view objects close to each other as belonging to a group.
* **Law of Similarity**: We look for likeness and differences in objects and link similar objects as belonging to a group.
* **Law of Closure**: Our minds tend to see complete figures or forms even if a picture is incomplete.
* **Law of Enclosure**: We view objects as belonging to a group when they are enclosed in a way that creates a boundary or border around them.
* **Law of Connectedness**: We view objects connected to each other as a single group as opposed to objects that are not linked in the same way.
* **Law of Continuity**: Our propensity is to see shapes as continuous to the greatest degree possible. The human eye follows lines, curves or a sequence of shapes to create pathways.

The libraries in python come with lots of different features that enable users to make highly customized, elegant, and interactive plots

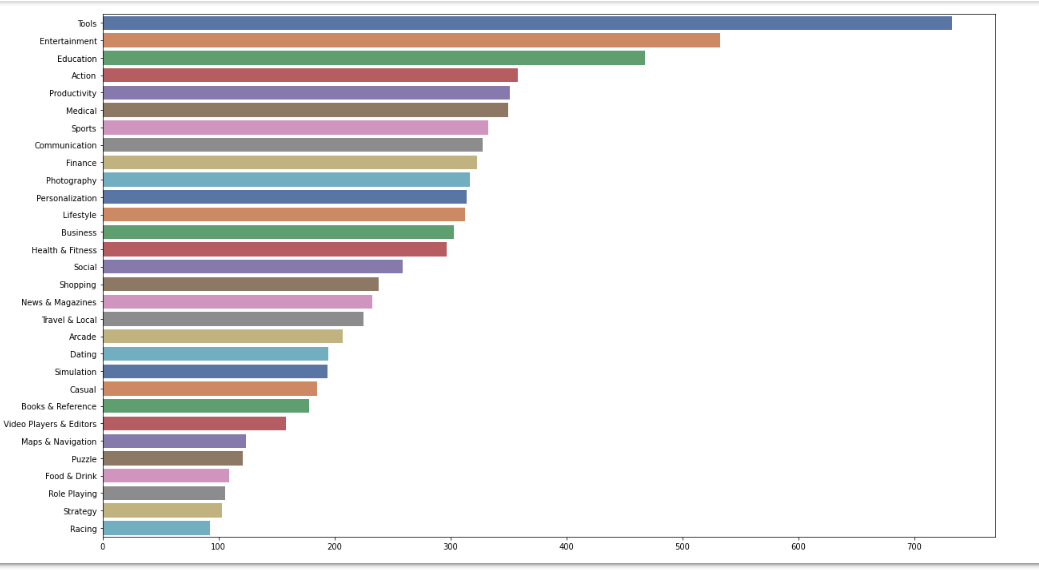
### Histograms

A histogram is a graph showing frequency distributions. It is a graph showing the number of observations within each given interval.



In this chartTotal Apps on yaxis and Rating on xaxis. There are total almost 1099 application that are coming under the 4.5 rating and 275 is rated as 5 , whereas 27 applications is in 1 rating that’s are fail/flops or bad in the play store.

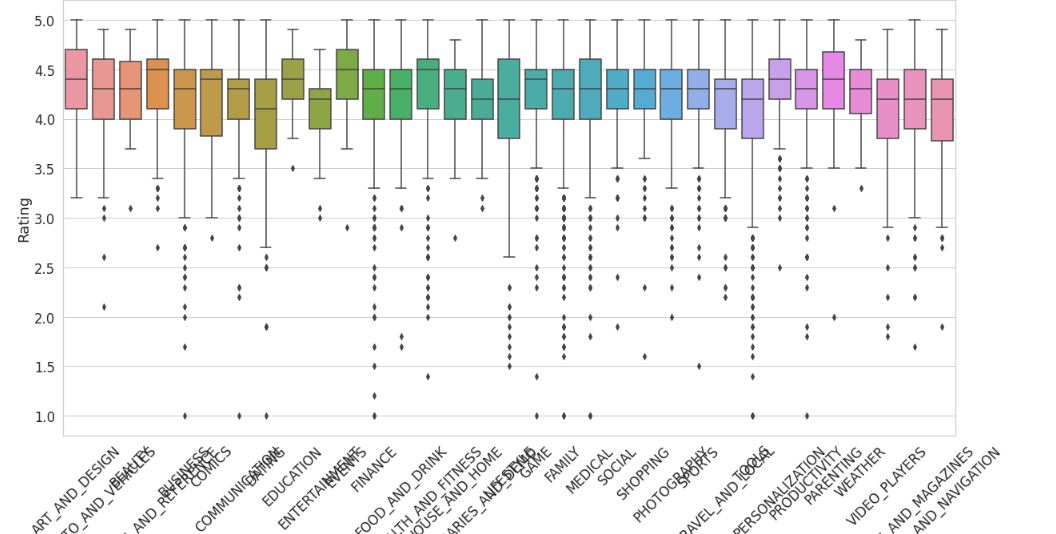
### BarPlot

A barplot shows the relationship between a numeric and a categorical variable. Each entity of the categorical variable is represented as a bar. The size of the bar represents its numeric value

The chart showing the top 30 app genres with different colour Tools, Entertainment and education category is the main category that are the most top category in the playstore.

### Box Plot

A Box Plot is also known as Whisker plot is created to display the summary of the set of data values having properties like minimum, first quartile, median, third quartile and maximum. In the box plot, a box is created from the first quartile to the third quartile, a vertical line is also there which goes through the box at the median. Here x-axis denotes the data to be plotted while the y-axis shows the frequency distribution.

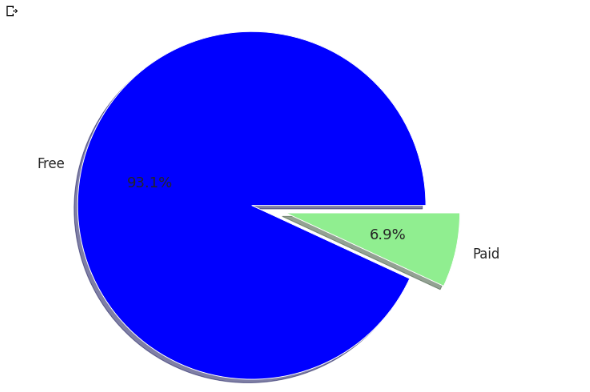


As showing the chart the Rating and Category in BoxPlot, xaxis showing the Category whereas yaxis showing the rating. As we can see that the graph showing **4.5** rating the maximum rating of an application that is considered as a good or successful application in the playstore.

### Pie Chart

A pie chart show a part-to-whole relationship in our data. Each slice represents one component and all slices added together equal the whole. Useful to show up data that is classified into nominal or ordinal categories.

As we can see in our Pie chart, we can see that 93 %( Approx.) Of apps in the google play store are free and 7 %(Approx.) Are paid



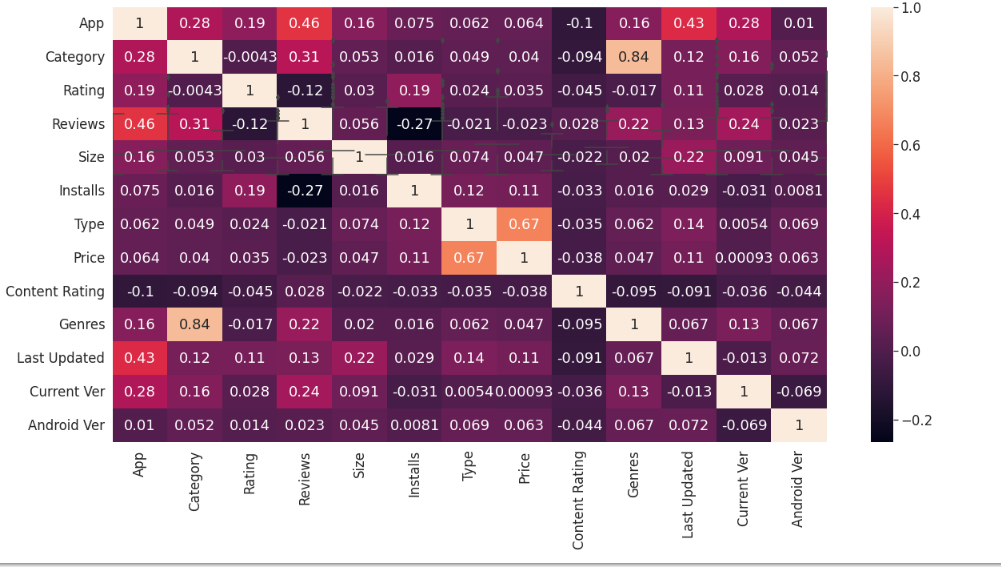
## Feature Selection

In machine learning and statistics, feature selection, also known as variable selection, attribute selection or variable subset selection, is the process of selecting a subset of relevant features for use in model

## Coloration b/w features

Correlation is a statistical term which in common usage refers to how close two variables are to having a linear relationship with each other.

Features with high correlation are more linearly dependent and hence have almost the same effect on the dependent variable. So, when two features have high correlation, we can drop one of the two features.



Correlation Matrix

Correlation is mostly used in Machine leering to figure out the least correlating variable that tends to be the best for classification.

### Correlation Analysis.

We used correlation for two purposed Feature engineering, regression

* **Feature engineering:** To optimize feature , like we have too much feature in our dataset but we are focusing on related to our business
* **Regression:** To know the relation b/w variable

At the end of correlation we will found the 1(positive) or -1(negative) values or 0 these are some term that are mostly using **Positive correlation,** **Negative Correlation, Zero Correlation**

* **Positive correlation :** When value of A increase the value of B must be increased
* **Negative Correlation:** When value of A increase the value of B decreased
* **Zero Correlation:** not relationship

# Experiment and Evaluation

Now this is our main concern and topic, for that I have created a complete description about **ML**,**Dataset**,**Domain**, **Visualisation** etc. Our main concern here to implement at least 3 algorithm which must be from the both **supervised** and **unsupervised** .

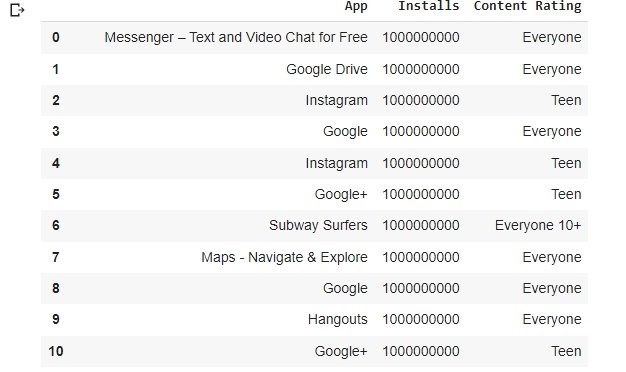
## Supervised learning

Let’s talk about some basics about supervised machine learning. **Supervised learning** machine learning is a type of machine learning, The main goal in supervised learning is to learn a model from labled training data that allows us to make predictions about unseen or future data, also the desired output labels are already known. Examples of Supervised Learning are ,Regression, Decision Tree, Random Forest, KNN, Logistic Regression etc.

There are two main terms are being used Independent, dependent Variables,

* **Independent** **Variables:** Independent variables that do not have any dependence to another variables.Another name for independent variables is Predictor variables. In data, independent variables will also be known as regressors, controlled variables, manipulated variables, explanatory variable, exposure variable, and/or input variable.
* **Dependent Variables:** These variables are opposite to independent variable the outcome of these variables is totally depends on the Independent variables. Another name for dependent variables is Predicted variables. In data, dependent variables will also be known as response variable, regressand, measured variable, observed variable, responding variable, explained variable, outcome variable, experimental variable, and/or output variables

After Feature engineering and data wrangling below is the data we will be working on for the exercise looks like this



Here is information about Google Store Rating, App, and Install. The goal is whether application is successful or not.

If the "Install “column has a value of "10000” or greater it means that the application, and if the value is less than 100000, it means that the application is not successful as this is our **threshold (**10000) Value

## Naive Bayes Classifiers

Naive Bayes classifiers based on supervised machine learning has a collection of classification algorithms based on **Bayes’** **Theorem**. These are collection of algorithms that share a common principle, i.e. every pair of features being classified is independent of each other.

Assumption: The fundamental Naive Bayes

### Gaussian Naive Bayes

A Gaussian Naive Bayes algorithm is a special type of NB algorithm. This is especially used when the Column/feature/characteristics have continuous values. It is also assumed that all feature follow a Gaussian distribution, ie a normal distribution.When plotted, it gives a bell shaped curve which is symmetric about the mean of the feature values as shown below:

![Diagram

Description automatically generated](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAeAB4AAD/4RDgRXhpZgAATU0AKgAAAAgABAE7AAIAAAAHAAAISodpAAQAAAABAAAIUpydAAEAAAAOAAAQyuocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEJlc2hlcgAAAAWQAwACAAAAFAAAEKCQBAACAAAAFAAAELSSkQACAAAAAzgxAACSkgACAAAAAzgxAADqHAAHAAAIDAAACJQAAAAAHOoAAAAIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAyMDIwOjExOjIzIDIxOjM5OjIzADIwMjA6MTE6MjMgMjE6Mzk6MjMAAABCAGUAcwBoAGUAcgAAAP/hCxlodHRwOi8vbnMuYWRvYmUuY29tL3hhcC8xLjAvADw/eHBhY2tldCBiZWdpbj0n77u/JyBpZD0nVzVNME1wQ2VoaUh6cmVTek5UY3prYzlkJz8+DQo8eDp4bXBtZXRhIHhtbG5zOng9ImFkb2JlOm5zOm1ldGEvIj48cmRmOlJERiB4bWxuczpyZGY9Imh0dHA6Ly93d3cudzMub3JnLzE5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMiPjxyZGY6RGVzY3JpcHRpb24gcmRmOmFib3V0PSJ1dWlkOmZhZjViZGQ1LWJhM2QtMTFkYS1hZDMxLWQzM2Q3NTE4MmYxYiIgeG1sbnM6ZGM9Imh0dHA6Ly9wdXJsLm9yZy9kYy9lbGVtZW50cy8xLjEvIi8+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczp4bXA9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8iPjx4bXA6Q3JlYXRlRGF0ZT4yMDIwLTExLTIzVDIxOjM5OjIzLjgxNDwveG1wOkNyZWF0ZURhdGU+PC9yZGY6RGVzY3JpcHRpb24+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iPjxkYzpjcmVhdG9yPjxyZGY6U2VxIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpsaT5CZXNoZXI8L3JkZjpsaT48L3JkZjpTZXE+DQoJCQk8L2RjOmNyZWF0b3I+PC9yZGY6RGVzY3JpcHRpb24+PC9yZGY6UkRGPjwveDp4bXBtZXRhPg0KICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICA8P3hwYWNrZXQgZW5kPSd3Jz8+/9sAQwAHBQUGBQQHBgUGCAcHCAoRCwoJCQoVDxAMERgVGhkYFRgXGx4nIRsdJR0XGCIuIiUoKSssKxogLzMvKjInKisq/9sAQwEHCAgKCQoUCwsUKhwYHCoqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioq/8AAEQgBjwIQAwEiAAIRAQMRAf/EAB8AAAEFAQEBAQEBAAAAAAAAAAABAgMEBQYHCAkKC//EALUQAAIBAwMCBAMFBQQEAAABfQECAwAEEQUSITFBBhNRYQcicRQygZGhCCNCscEVUtHwJDNicoIJChYXGBkaJSYnKCkqNDU2Nzg5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6g4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2drh4uPk5ebn6Onq8fLz9PX29/j5+v/EAB8BAAMBAQEBAQEBAQEAAAAAAAABAgMEBQYHCAkKC//EALURAAIBAgQEAwQHBQQEAAECdwABAgMRBAUhMQYSQVEHYXETIjKBCBRCkaGxwQkjM1LwFWJy0QoWJDThJfEXGBkaJicoKSo1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoKDhIWGh4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uLj5OXm5+jp6vLz9PX29/j5+v/aAAwDAQACEQMRAD8A+kMk9BSZb0oj/wBWPxp1ADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADct6UZb0p1FADYv9WPx/nTqbF/qx+P86dQAUUVUttW0+8vLm1tL63nuLUgXEUcoZoif7wHT8aB2b1LdFQ2t5bX0HnWU8dxFuK74nDDIOCMj3oub22s/K+13EcPnSCOPzHC73PRRnqeOlAWd7E1FFFAgooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiobu8trC3M99cRW8K8GSVwqj8TSbS1YbE1FZP/CV+Hv+g5p3/gUn+NXbfUrK7k8u1u4Jn8tZdscgY7D0bjsfWkpRezIU4vZlmiiiqLCioLy+tNPg86/uYbaLIXfM4RcnoMmnTXUFvbNcXE0cUCjc0jsAoHrmldCutiWikR1kRXjYMrDIYHIIpaYwooooAKKKZPPFbQvNcSJFEgyzu2Ao9STRsA+imxyJLGskTq6MMqynII9azH8UaDHI0b61p6upKspuUBBHUdaTkluyXKK1bNWioHvrSKz+1yXMKW2M+czgJj1z0qZWDqGUgqRkEd6Y7oWikJAIBPJ6UMyopZyFVRkkngCgYtFVYtUsJmhWG9t5GuAWhCygmQDqV9fwq1Re4k09gooqKe5gtoJJriaOKKMZd3YBVHqSelF7DJaKajrLGrxsHRhlWU5BHrTqACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbF/qx+P86dTYv9WPx/nTqAIrkTG1lFqUWcofLLjKhscZ9s14boc2r+H/AIa+KNUlez+0X+oPbrNbxkTea0uxyWPVRnKjtzXu9eZap4M1KD4U6xpzhXuxfS38KwnO4ed5ij6kDFYVI638n+a/S56GDqRiuSWzlH7tf+AW9b1S9+Gul6I1tbwyeHLdFgvdsZ82M4++MHGCc54qBJtX+IHw91q91CKO1tZlaXSUVSsqbPmSRmz1JA6Y4qbWdJu/iVZaIFmjj8Nyxie7CSESSvj7mMcAH3qCKDV/h/4B1uy1CVLq0hVotJcMWlcv8qRlcdiR+FE+a8r7a2/r8jaHLyRtb2t1f7/zvv5fM7HwfqsmueDdK1KcYlubZHkx/exz+ua2axfB2lS6H4M0rTbj/XW9siSY/vY5/XNbVdD3PMny8z5dgooopEBRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVBeWVtqFsbe+gjuIWOTHIoYH8KnopNJqzA8707w5oz/E7VrV9LtWt47WJkiMQ2qT1IFRz2+rw/E+9tPC4tLYLpsKlpkJSNAeAAK6ey0S6t/Heo6vIY/s1zbxxoA3zZXrkU+10a5h8eX+sOU+zT2kcKAH5tynJyK46dHSK21l/wC3W/Q4PYaNJW97p2OYb4g6ha+GQbyO2XVftrWW9iViBH8Z9qt+HvF2oSeKotG1S70+/FxCZIp7HI2kdVINVrzwJqM9ncSRm2N3Hqb3luknzI6n+FuK0tA0TWP+EgXUdStLDTreKIqlvaKCXY/xFsVNL2zknO/T8tfx+ZmlXU0ruyt+et/l/wAAZ8UrdbvwnFbuSqy3kKEjtlsVgvqdxa+C9c8Maw2b3T4SIXb/AJbQ5+U/hXaeL9Gudc0uC3sygeO6ilO84G1WyayvH/gubxLaxz6W6w6hENm5m2h0PVSaK1Ofvygt9PVW/R/qXiKc3OVSC1UUvXe6/JkX9s61eahaaH4c+zwm3s45bm5uFLAZHCgCqVx461qx0LVI7q3tjq+l3UUL7QfLlVzwQM5GRV+fQte0fWotV0Jbe5MtqkF1bTOVGVGAwNUZfBGsXmialNeSW51bU7yGeVVYiONEYYUHvxmq/eNta9fz0t8v+CZT9uovkvfX0ty6fO9vxLGoa34vsJNJsmGntf6jK4I2tsiXGRzntVa31rxxeX+paZC+lifTRukuDG22TIyFAzweK6bV9Fub7xLo19CU8mydjLuPPIxxTdM0O6tPEGu3kpj8q/2+VhueFxz6U5U5tvV9fyVvxN/ZVOdLmdrr7ra/iYjeMNWv/CujX1j9isnvt/2i4um/dw7CQcDOTkjis/8A4Si71rwz4lsL64tLtrSDKXNoCEcH2Pemr4G1qy0/w8y29lfSaasyy2sznyyXckMDjkgEVNb+EPEPm62biLT411S22KsDFViYdFxjp71jWVWXMtdU/wAtPxOeEsQ3DmT2V/u1/H5/I7fQP+RcsP8Ar3T+VcV4A0DSdV03WJdR062uZP7VnTfJGCwHy8Z/Gu70q2ez0i1tpseZFEqNg8ZArh9I0fxr4eS/t9Mg0uSG5u5LhXllbcN2O2PQCuqpZTTlG6s+l+qOipGypuUb230v0Moyf8I7Z+LNJt0S5sbPZLBBcZdU3EZXr0rR1Dxjqv8AbjaXY3enaWsNvG0X2xG/fsVzhecAVbPge/8A+ET1eGa5judY1Rg8sp+VBg8KPYU3VPD+vm5mj+wafrFnLCixpdHa0DBcEA45Fc/LUjTS1/q+mnbQ5+SrH4U0v+C9NPltsUNYuPEWoa/4Suc2VtNL5m2NsuFcA7iSOoK4x6VpWmu+I9W1fVoUjsk03TpHSVnU7pRtztHP61BB4O1zS9J8Nixe2uLvS5ZHlSVyEw+eAfbOK3dG0G8sV17z/LzqEzPFtbOAVxzTjCpzSvdbv8I/rf7jWlCo5Ju6va/3P9dziRdaldXvhKXRILSC7lt5gqlSIox3OM5rbl8W6/o/9tafrCWsl9ZWRvLaeJCI5FzjBUnPWov+ES8R2EegT6YbRrnTIpFkSRztfd26VZ/4RLWta/tm/wBde3gvb2z+x28UJLJEnXJPuamCqJaXvr6bfnf+rGSjWTfLe7t6fCrmpqXiG9tdP8PTQiLfqUkazblJA3Jk4545rkIJNYtD40nvWsbmGPcZYjG2HfAx3+7jrWrHoHi2/l0SPU0sILbSpUOInJaQKMZ6elT3vhfW5LnxLBAtq1pq0ZaN2ch1fAABGOnWlVVScnJJ9bf+Ar9TR+0motp6W/J/qKuu61qWoWmi+G0tLUwWMVxczzISiblG1FAqtP421u00W/imt7Y6vYXcdu2AfLlDnggZ4q03h/X9D1eHVNCW3uWlsora7tpnKjcigBlNQf8ACGavc6XdT3skB1K+vormVVYhI1Q/dB78VpP2rk7X3f3X0t8v1uS/bW0vf8LW/O/z+Rcsda8SWHiyw07xD9ilh1KN2jNsrAxsozg568V2lc/q2i3V74r0PUYTH5Fj5vm7mwfmXAwO9dBXVTTimmdlFSi5KW19L+i/W4UUUVodAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQA2L/AFY/H+dOpsX+rH4/zp1ABQRkYNFec/Gv4hN4A8EGaxkC6pev5NqO692fHsP1IoA9AtbO2sYfJs4I4ItxbZGoUZJyTgeppLmytr3yvtdvHP5Mglj8xA2xx0YZ6HnrWL4F8VW/jTwZp+tWxGZ4wJUH8Eg4YfnXQ0Du73CiiigQUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFADYv9WPx/nTqbF/qx+P86dQAV4T8c/h3ca7bap4r1rUWFvptuI9PsYRwBkbncnuSe3oK92rh/jJ/wAkl1v/AK5D/wBCFAGf8KPAF14CtpI7PUTd6PfxJcJDKMPBIQM4PQg16RVHRP8AkAaf/wBe0f8A6CKvUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVk+FtTm1nwxZahdBRLPHuYKMDOSK1q5zwB/yIWlf9cj/wChGgDo6KKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbF/qx+P86dTYv9WPx/nTqACuH+Mn/JJdb/65D/0IV3FcP8ZP+SS63/1yH/oQoA6rRP8AkAaf/wBe0f8A6CKvVR0T/kAaf/17R/8AoIq9QAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABXOeAP8AkQtK/wCuR/8AQjXR1zngD/kQtK/65H/0I0AdHRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFADYv9WPx/nTqbF/qx+P8AOnUAFcP8ZP8Akkut/wDXIf8AoQruK4f4yf8AJJdb/wCuQ/8AQhQB1Wif8gDT/wDr2j/9BFXqo6J/yANP/wCvaP8A9BFXqACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAK5zwB/yIWlf9cj/AOhGujrnPAH/ACIWlf8AXI/+hGgDo6KKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbF/qx+P86dTYv8AVj8f506gArh/jJ/ySXW/+uQ/9CFdxXD/ABk/5JLrf/XIf+hCgDqtE/5AGn/9e0f/AKCKvVR0T/kAaf8A9e0f/oIq9QAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFZuueINL8N6a9/rd7FaW6fxSNjJ9AOpPsKANKuc8Af8iFpX/XI/wDoRrl0+I/ijxCxbwR4JuLmzz8t9qcwtUceqqRkj8qx/Cfir4gaP4P0+R/BMOp6csZw9jfAzYyefLI5PsKAPY6K5Xwl8RdA8YM9vYTvbahEP32n3aeXPF65U9fqMiuqoAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigBsX+rH4/zp1Ni/1Y/H+dOoAK4f4yf8kl1v8A65D/ANCFdxXD/GT/AJJLrf8A1yH/AKEKAOq0T/kAaf8A9e0f/oIq9VHRP+QBp/8A17R/+gir1ABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRXP+M/F9n4O0M3lyrT3MrCK0tI+XuJT0VR/nFAEPjTxraeEbGJfKa91S7by7HT4eZJ3+nYeprC8O/D+51LUU8SfEWRNS1c/NBZdbaxHZVXoW9WNWfBHg67ivpPFXi9lufEV6v3eqWUZ6RJ6Y7nvXdUAAAAAAwB0ArnPAH/IhaV/1yP8A6Ea6Ouc8Af8AIhaV/wBcj/6EaAK3jL4fab4sWO6Rm07WLc7rXUrb5ZY29yPvD2NZngvxnfx6u/hHxuEg16AZhnAxHfx/309/UV39cp4/8Fx+L9FX7NJ9l1azbzrC7X70Ug5Az6HoRQB1dFcl8PvF7+J9Hlg1OL7Lremv9n1G2PBSQfxD/ZPUGutoAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAGxf6sfj/OnU2L/AFY/H+dOoAK4f4yf8kl1v/rkP/QhXcVw/wAZP+SS63/1yH/oQoA6rRP+QBp//XtH/wCgir1UdE/5AGn/APXtH/6CKvUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUjMqIWchVUZJJwAKAKOua1Y+HdFudU1WZYbW2Qu7H+Q9SfSuG8F6He+KteHjzxXA0cjKV0ewk6WkJ/jI/vt19qp2ob4t+L/ALZIG/4Q7RZ8W6EfLqVyvVz6op6epr1QAAAAYA6CgAooooAK5zwB/wAiFpX/AFyP/oRro65zwB/yIWlf9cj/AOhGgDo6KKKAPNviBZXPhLxBb/ELRYmdYFEGtW0Y/wBfbZ/1mO7J1+n416FY3tvqWnwXtlKs1vcRrJFIhyGUjINSTQx3EDwzoskcilXRhkMCMEGvNfBssngDxnP4Ev3Y6XeFrrQZnPRScvb59VPI9jQB6bRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQA2L/Vj8f506mxf6sfj/OnUAFcP8ZP+SS63/wBch/6EK7iuH+Mn/JJdb/65D/0IUAdVon/IA0//AK9o/wD0EVeqjon/ACANP/69o/8A0EVeoAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigArzbx1qd54t8QJ8P/DkzxeYgl1q9jP8Ax7W5/wCWYP8Affp9PxroPHvi7/hFtEUWUf2nV75/s+n2o6ySnofoOpo8A+ER4U0NhdyfadWvnNxqF03LSyt159B0FAG9pWl2ei6VbabpsCwWttGI4o0HCgVboooAKKKKACuc8Af8iFpX/XI/+hGujrnPAH/IhaV/1yP/AKEaAOjooooAK5X4heFG8U+Gyti/karYyC60+4HWOZeRz6HofrXVUUAc14D8Vr4v8LxXkieRfQsbe+tzwYZl4YY+vIrpa8x15W+HnxIh8RwjboevOttqaj7sM3RJfx6GvTVYMoZTkEZBHegBaKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbF/qx+P86dTYv9WPx/nTqACuH+Mn/JJdb/65D/0IV3FcP8ZP+SS63/1yH/oQoA6rRP8AkAaf/wBe0f8A6CKvVR0T/kAaf/17R/8AoIq9QAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFV9Qv7bS9Onvr6VYbe3jMkkjHAVQMk1YrzDxPM/xH8cjwbYO39h6SyT67Oh4lfqlsD+GW/wARigCbwLYXPjDxDL4+1yJkjcGLRraQf6mHP+sx/eb+Vek0yGKOCFIYUWOONQqqowFA6AU+gAooooAKKKKACuc8Af8AIhaV/wBcj/6Ea6Ouc8Af8iFpX/XI/wDoRoA6OiiigAooooAztf0S08R6Dd6TqKB4LqMo3t6Ee4rk/hjrd2ILzwnrzk6voTiIs3WeE/6uQeuRxXe15z8TLG50G+sfiBo0TSXGk/JqMMY5uLMn5/qV+8P/AK1AHo1FV9PvrbVNNt7+xlWa2uYllikU8MrDIP5GrFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUANi/wBWPx/nTqbF/qx+P86dQAVw/wAZP+SS63/1yH/oQruK4f4yf8kl1v8A65D/ANCFAHVaJ/yANP8A+vaP/wBBFXqo6J/yANP/AOvaP/0EVeoAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKr399b6Zp897eyrFb26GSR2OAoAoA5f4i+LJ/DmjRWejIJ9e1WT7Np0HX5z1c/7Kjk1e8DeEoPBvhiHTo3M9y7Ge9um5a4nbl3J9z+gFcx4AsLjxX4gufiBrUTL56mDSLdx/qLfP38dmbr9K9JoAKKKKACiiigAooooAK5zwB/yIWlf9cj/wChGujrnPAH/IhaV/1yP/oRoA6OiiigAooooAKZLFHPC8UyLJHIpV0YZDA8EEU+igDzPwDLJ4J8XX3w+v3Y2R3XuhSuc7oGOXhz6o2fwP0r0yuK+J3hy61bQ4NW0T5db0SX7ZYuOrEfej+jDj8q3fCfiO18WeGLLWLLhbiPLoesbjhlPuDQBsUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFADYv9WPx/nTqbF/qx+P8AOnUAFcP8ZP8Akkut/wDXIf8AoQruK4f4yf8AJJdb/wCuQ/8AQhQB1Wif8gDT/wDr2j/9BFXqo6J/yANP/wCvaP8A9BFXqACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACvMvF88nj7xpD4I09z/ZdkVudbmQ8EdUgz6nqR6V0nxA8W/wDCJ+HfMtI/tGqXri20+2HJlmbgceg6n6UvgHwl/wAIl4cENzJ9o1O7c3GoXJ6yzNyefQdBQB0sEMdtbxwwII4o1CoqjAUDoKfRRQAUUUUAFFFFABRRRQAVzngD/kQtK/65H/0I10dc54A/5ELSv+uR/wDQjQB0dFFFABRRRQAUUUUABGRg15lpv/FvfinLpT/JofiZjPaE/dguh99Pbd1+tem1zHxB8Lf8JZ4TntIG8u/gIuLKYdY5k5U/0oA6eiuX+Hvir/hLPCcN1cL5d/bk297Cescy8MP6/jXUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFADYv8AVj8f506mxf6sfj/OnUAFcP8AGT/kkut/9ch/6EK7iuH+Mn/JJdb/AOuQ/wDQhQB1Wif8gDT/APr2j/8AQRV6qOif8gDT/wDr2j/9BFXqACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKjuLiK1tpLi5kWKGJC7uxwFUDJJNSV5n47u5/G3iiD4faRKyWuFuNduIzjy4M5WEH+85/T8RQAnguCbx94vl8e6nGy6bb7rbQLeQf8s84e4I9WPT2/CvTahtLWCxs4bW0iWKCFBHHGgwFUDAAFTUAFFFFABRRRQAUUUUAFFFFABXOeAP+RC0r/rkf/QjXR1zngD/AJELSv8Arkf/AEI0AdHRRRQAUUUUAFFFFABRRRQB5fq3/Fu/irBrCfu9C8TOLa9H8MN1/A/tu6GvUOtY3izw3aeLvC19ot+MR3UZVXA5jfqrj3Bwa5/4WeJrvV9Bn0bX/l1/QZfsV+pPMm37ko9Q6jOe5BoA7miiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigBsX+rH4/zp1Ni/1Y/H+dOoAK4f4yf8kl1v/rkP/QhXcVw/xk/5JLrf/XIf+hCgDqtE/wCQBp//AF7R/wDoIq9VHRP+QBp//XtH/wCgir1ABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUVHPPFbW8k9w6xxRqWd2OAoHU0Ac9478Wp4R8OPcxJ5+oXDCCxth1mmbhRj07moPh74SfwxoLSai/2jWNQkNzqFweryN2+g6Cuc8IwS/EHxlJ421BGGk2TNBokLjhsHDT49+gr0+gAooooAKKKKACiiigAooooAKKKKACuc8Af8iFpX/XI/8AoRro65zwB/yIWlf9cj/6EaAOjooooAKKKKACiiigAooooAK8x+IEEngzxhYfELT0Y22FstbjQffgJ+WXHqhxz6Y7CvTqrajp9vqum3FhfRrLb3MbRyIwyCpGDQBLBPFc28c9u6yRSqHR1OQwIyDUlebfDTULnQNUvvh/rUjNPpv73TpXPM9qT8v1K9K9JoAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbF/qx+P8AOnU2L/Vj8f506gArh/jJ/wAkl1v/AK5D/wBCFdxXD/GT/kkut/8AXIf+hCgDqtE/5AGn/wDXtH/6CKvVR0T/AJAGn/8AXtH/AOgir1ABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAV5n47vbjxp4oh+HuiyskG1bjXrqM/6mDtCD/ef09PY10fj7xePCWgB7WL7Vqt44t9PtByZpm6ceg6mm/D7wgfCWgN9tl+1axqEhutSuz1lmbkj/AHR0A/xoA6SysrfTrGGzsolht4EEccajAVQMAVPRRQAUUUUAFFFFABRRRQAUUUUAFFFFABXOeAP+RC0r/rkf/QjXR1zngD/kQtK/65H/ANCNAHR0UUUAFFFFABRRRQAUUUUAFFFFAHA/FDQLyS0tPFXh9M61oL+fGq9Z4f44j9RzXVeG9fs/E/h2z1fTn3QXMYYDup7qfcHitQjIweRXlumn/hWnxLfSJDs8O+JJDLZMfu211/FH7Buo96APUqKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbF/qx+P86dTYv8AVj8f506gArh/jJ/ySXW/+uQ/9CFdxXD/ABk/5JLrf/XIf+hCgDqtE/5AGn/9e0f/AKCKvVR0T/kAaf8A9e0f/oIq9QAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVXv7620zT572+lWG3t0MkkjHAUCrFeXeI55PiX40/4RKwdv7A0t1l1mdOkzjlYAf1NAE3gixuPGvieTx9rUTJbAGHRLWQf6qHvKR/eb+Vel0yGGO3gSGBFjjjUKiqMBQOgp9ABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABXOeAP8AkQtK/wCuR/8AQjXR1zngD/kQtK/65H/0I0AdHRRRQAUUUUAFFFFABRRRQAUUUUAFYHjXwpa+MvC9zpV0fLdhvt51+9BKOVcfQ1v0UAcV8NPFd1rukXGla+vk+IdFk+y6hEf4iPuyj/ZYc5+tdrXm3xF0q78O61a/EPw7CXurBfK1W2T/AJfLTPP/AAJOoP8AgBXe6Tqtprek22pabMs1rdRiSN17g/1oAuUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFADYv9WPx/nTqbF/qx+P8AOnUAFcP8ZP8Akkut/wDXIf8AoQruK4f4yf8AJJdb/wCuQ/8AQhQB1Wif8gDT/wDr2j/9BFXqo6J/yANP/wCvaP8A9BFXqACiiigAooooAKKKKACiiigAooooAKKKwPGXi6x8GeH5NRvt0srHy7a1j5kuJT91FHck0AY3xD8U3litr4a8MgS+IdXykAHIt4/4pm9AB09TW54P8K2fg/w7BpdlmRly887/AHp5DyzsfUmsL4eeFL2x+1eJ/FREviTV8PP3FrH/AAwJ6ADr6mu5oAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACuc8Af8iFpX/XI/8AoRro65zwB/yIWlf9cj/6EaAOjooooAKKKKACiiigAooooAKKKKACiiigBrossbJIoZWGCD3FeW6VI/wr8bf2JcsR4X1qUvp8jfds5z1iz2U9q9UrI8UeG7HxZ4fuNJ1JMxzL8rj70bjowPYg0Aa/WivPfh94nvba+m8FeLn265p6/uJn4F9APuyKe59RXoVABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQA2L/Vj8f506mxf6sfj/ADp1ABXD/GT/AJJLrf8A1yH/AKEK7iuH+Mn/ACSXW/8ArkP/AEIUAdVon/IA0/8A69o//QRV6qOif8gDT/8Ar2j/APQRV6gAooooAKKKKACiiigAooooAKKKwvFnjDS/B2lfbNUly7nbBbRjMk79lVepNAFjxJ4k03wpok2qaxOIoIhwBy0jdlUdyfSuL8JeH9T8V+Ik8ceNIDCyKRo+ltyLOM/8tGH/AD0Yfl/Jvh7wnq3i7XIfFfxATyxEd+m6NnKWw7O/q/8AKvS6ACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAK5zwB/yIWlf9cj/6Ea6Ouc8Af8iFpX/XI/8AoRoA6OiiigAooooAKKKKACiiigAooooAKKKKACiiigDlPHfgmPxdp8MtncHT9bsH87TtQQfNDIOx9VPQiqfgnx5Lq15L4d8VWw0vxPZD99bE/JcL/wA9Yj/Ep647frXb1zHjTwPZeMLSFzK9jqlo2+y1CDiSBvr3HqKAOnorzjQPHmoaDqcfhz4kolpek7bXVFGLe8Hbn+FvY16MrBlDKQQRkEd6AFooooAKKKKACiiigAooooAKKKKACiiigBsX+rH4/wA6dTYv9WPx/nTqACuH+Mn/ACSXW/8ArkP/AEIV3FcP8ZP+SS63/wBch/6EKAOq0T/kAaf/ANe0f/oIq9VHRP8AkAaf/wBe0f8A6CKvUAFFFFABRRRQAUUUUAFFZ2t+INK8Oac99rd9DZ26DlpGxn2A7n2FefNrPi74mkw+GFm8NeHGOH1WdMXNyv8A0yQ/dB/vHmgDb8VfEaLTNS/sDwvaNrniOQfLaQn5Lf8A2pX6KB6daj8KfD6W31T/AISTxpdjWPEUg4cj9zaD+5EvbHr1Nb3hTwbo3g3TPsei22wud01xId0s7d2djyTW7QAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAVzngD/kQtK/65H/0I10dc54A/5ELSv+uR/wDQjQB0dFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQBn61oWm+IdNksNZtI7q3kGCjjOPcHsfevPhY+LfhjIW0hJ/E/hkcmzLZu7Rf8AYJ++B6da9RooAwvC/jLQ/GNibnQr5JinE0DfLLC3o6HkH9K3a4zxT8M9L1+/GsabPNoWvx/6vU7A7Hb2kXpIPrzjjNYieNPFngdhB8Q9M/tHT1OF1zS4yRj1liHKn3HHtQB6dRWboniLSPEdit3omoQXkLfxRODj2I6g1pUAFFFFABRRRQAUUUUAFFFFADYv9WPx/nTqbF/qx+P86dQAVw/xl/5JLrn/AFxH/oQruKrajp1pq2nTWOowLPbTqUkjcZDCgChouq6eNBsAb62BFtHkGZf7o96u/wBrad/z/wBr/wB/l/xrjx8GPAwGBowA/wCur/40v/CmfA3/AEBh/wB/X/xoA6/+1tO/5/7X/v8AL/jR/a2nf8/9r/3+X/GuQ/4Uz4G/6Aw/7+v/AI0f8KZ8Df8AQGH/AH9f/GgDr/7W07/n/tf+/wAv+NH9rad/z/2v/f5f8a5D/hTPgb/oDD/v6/8AjR/wpnwN/wBAYf8Af1/8aAOquvEGj2Vs9xdapZxRRjLO064A/OvPbz4uS+ILqTTvh1Zx3sgO19SvXEVtF7jPL/hWrJ8FfAkqbZNEVl9DK3+NKnwW8CRrtj0RVUdllYD+dAFDRPAekNqKa14412DxHq4O5DPKot7c+kcecfia79dU01VAW+tQAMACZeP1rkf+FM+Bv+gMP+/r/wCNH/CmfA3/AEBh/wB/X/xoA6/+1tO/5/7X/v8AL/jR/a2nf8/9r/3+X/GuQ/4Uz4G/6Aw/7+v/AI0f8KZ8Df8AQGH/AH9f/GgDr/7W07/n/tf+/wAv+NH9rad/z/2v/f5f8a5D/hTPgb/oDD/v6/8AjR/wpnwN/wBAYf8Af1/8aAOv/tbTv+f+1/7/AC/40f2tp3/P/a/9/l/xrkP+FM+Bv+gMP+/r/wCNH/CmfA3/AEBh/wB/X/xoA6/+1tO/5/7X/v8AL/jR/a2nf8/9r/3+X/GuQ/4Uz4G/6Aw/7+v/AI0f8KZ8Df8AQGH/AH9f/GgDr/7W07/n/tf+/wAv+NH9rad/z/2v/f5f8a5D/hTPgb/oDD/v6/8AjR/wpnwN/wBAYf8Af1/8aAOv/tbTv+f+1/7/AC/40f2tp3/P/a/9/l/xrkP+FM+Bv+gMP+/r/wCNH/CmfA3/AEBh/wB/X/xoA6/+1tO/5/7X/v8AL/jR/a2nf8/9r/3+X/GuQ/4Uz4G/6Aw/7+v/AI0f8KZ8Df8AQGH/AH9f/GgDr/7W07/n/tf+/wAv+NH9rad/z/2v/f5f8a5D/hTPgb/oDD/v6/8AjR/wpnwN/wBAYf8Af1/8aAOv/tbTv+f+1/7/AC/40f2tp3/P/a/9/l/xrkP+FM+Bv+gMP+/r/wCNH/CmfA3/AEBh/wB/X/xoA6/+1tO/5/7X/v8AL/jR/a2nf8/9r/3+X/GuQ/4Uz4G/6Aw/7+v/AI0f8KZ8Df8AQGH/AH9f/GgDr/7W07/n/tf+/wAv+Nc74C1OwTwLpave26sIjkGVQR8x96pf8KZ8Df8AQGH/AH9f/GkX4L+BVXC6KoA7CV/8aAOw/tbTv+f+1/7/AC/40f2tp3/P/a/9/l/xrkP+FM+Bv+gMP+/r/wCNH/CmfA3/AEBh/wB/X/xoA6/+1tO/5/7X/v8AL/jR/a2nf8/9r/3+X/GuQ/4Uz4G/6Aw/7+v/AI0f8KZ8Df8AQGH/AH9f/GgDr/7W07/n/tf+/wAv+NH9rad/z/2v/f5f8a5D/hTPgb/oDD/v6/8AjR/wpnwN/wBAYf8Af1/8aAOv/tbTv+f+1/7/AC/40f2tp3/P/a/9/l/xrkP+FM+Bv+gMP+/r/wCNH/CmfA3/AEBh/wB/X/xoA6/+1tO/5/7X/v8AL/jR/a2nf8/9r/3+X/GuQ/4Uz4G/6Aw/7+v/AI0f8KZ8Df8AQGH/AH9f/GgDr/7W07/n/tf+/wAv+NH9rad/z/2v/f5f8a5D/hTPgb/oDD/v6/8AjR/wpnwN/wBAYf8Af1/8aAOv/tbTv+f+1/7/AC/40f2tp3/P/a/9/l/xrkP+FM+Bv+gMP+/r/wCNH/CmfA3/AEBh/wB/X/xoA6/+1tO/5/7X/v8AL/jR/a2nf8/9r/3+X/GuQ/4Uz4G/6Aw/7+v/AI0f8KZ8Df8AQGH/AH9f/GgDr/7W07/n/tf+/wAv+NH9rad/z/2v/f5f8a5D/hTPgb/oDD/v6/8AjR/wpnwN/wBAYf8Af1/8aAOv/tbTv+f+1/7/AC/40japprqVa+tGUjBBmXn9a5H/AIUz4G/6Aw/7+v8A40f8KZ8Df9AYf9/X/wAaAKet/D7wvd3zan4d1ZPDmrHn7Tp86qrn/aTO1qoDxz4r8HLs8UW1n4gsE/5iGlzKJQPVoif5Vt/8KZ8Df9AYf9/X/wAaP+FM+Bv+gMP+/r/40AaHhz4leFPFEWdM1eBZR96C4bypF9irVv8A9rad/wA/9r/3+X/GuL/4Uh4AEokGgxhwc7hI2f51L/wpnwN/0Bh/39f/ABoA6/8AtbTv+f8Atf8Av8v+NeeeK/jJZeB/F8Wn69Csul3aB4L60cPsPQq6itT/AIUz4G/6Aw/7+v8A415r40+Aj+J/F0Fp4cs4tE0W1T9/eSsXedz2Rc9AO5wOaAPcdC8SaR4lsFvND1CC9hYZzE4JH1HUVp1xPw/+FXh/4dwE6Sks13Iu2W6mbLN+HQfhXbUAFFFFADYv9WPx/nTqbF/qx+P86dQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUANaREZVZ1VmOFBPWjzEMhj3rvAyVzziuI8bypD418HGRwgN84yTj+A06zmjk+NWoojhmj0uIMAfuncTU82qXnb8LnT7D3Oa/S/42O3oooqjmCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbF/qx+P8AOnU2L/Vj8f506gAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooA4rx54SfxVq+gJLamewt7h3uisuwopXAPUHr6Vb0f4b+HNB1eHVNNtp0vIlKiRrl2Lg/3sn5uveuqoqVFJ3Oj6zV5FTTstgoooqjnCiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbF/qx+P8AOnU2PiMfjTqACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooA//9k=)

The conditional probability is calculated by

![Text, whiteboard

Description automatically generated](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAeAB4AAD/4RDgRXhpZgAATU0AKgAAAAgABAE7AAIAAAAHAAAISodpAAQAAAABAAAIUpydAAEAAAAOAAAQyuocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEJlc2hlcgAAAAWQAwACAAAAFAAAEKCQBAACAAAAFAAAELSSkQACAAAAAzA1AACSkgACAAAAAzA1AADqHAAHAAAIDAAACJQAAAAAHOoAAAAIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAyMDIwOjExOjIzIDIxOjQwOjI3ADIwMjA6MTE6MjMgMjE6NDA6MjcAAABCAGUAcwBoAGUAcgAAAP/hCxlodHRwOi8vbnMuYWRvYmUuY29tL3hhcC8xLjAvADw/eHBhY2tldCBiZWdpbj0n77u/JyBpZD0nVzVNME1wQ2VoaUh6cmVTek5UY3prYzlkJz8+DQo8eDp4bXBtZXRhIHhtbG5zOng9ImFkb2JlOm5zOm1ldGEvIj48cmRmOlJERiB4bWxuczpyZGY9Imh0dHA6Ly93d3cudzMub3JnLzE5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMiPjxyZGY6RGVzY3JpcHRpb24gcmRmOmFib3V0PSJ1dWlkOmZhZjViZGQ1LWJhM2QtMTFkYS1hZDMxLWQzM2Q3NTE4MmYxYiIgeG1sbnM6ZGM9Imh0dHA6Ly9wdXJsLm9yZy9kYy9lbGVtZW50cy8xLjEvIi8+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczp4bXA9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8iPjx4bXA6Q3JlYXRlRGF0ZT4yMDIwLTExLTIzVDIxOjQwOjI3LjA1MDwveG1wOkNyZWF0ZURhdGU+PC9yZGY6RGVzY3JpcHRpb24+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iPjxkYzpjcmVhdG9yPjxyZGY6U2VxIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpsaT5CZXNoZXI8L3JkZjpsaT48L3JkZjpTZXE+DQoJCQk8L2RjOmNyZWF0b3I+PC9yZGY6RGVzY3JpcHRpb24+PC9yZGY6UkRGPjwveDp4bXBtZXRhPg0KICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICA8P3hwYWNrZXQgZW5kPSd3Jz8+/9sAQwAHBQUGBQQHBgUGCAcHCAoRCwoJCQoVDxAMERgVGhkYFRgXGx4nIRsdJR0XGCIuIiUoKSssKxogLzMvKjInKisq/9sAQwEHCAgKCQoUCwsUKhwYHCoqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioq/8AAEQgAVQIVAwEiAAIRAQMRAf/EAB8AAAEFAQEBAQEBAAAAAAAAAAABAgMEBQYHCAkKC//EALUQAAIBAwMCBAMFBQQEAAABfQECAwAEEQUSITFBBhNRYQcicRQygZGhCCNCscEVUtHwJDNicoIJChYXGBkaJSYnKCkqNDU2Nzg5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6g4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2drh4uPk5ebn6Onq8fLz9PX29/j5+v/EAB8BAAMBAQEBAQEBAQEAAAAAAAABAgMEBQYHCAkKC//EALURAAIBAgQEAwQHBQQEAAECdwABAgMRBAUhMQYSQVEHYXETIjKBCBRCkaGxwQkjM1LwFWJy0QoWJDThJfEXGBkaJicoKSo1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoKDhIWGh4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uLj5OXm5+jp6vLz9PX29/j5+v/aAAwDAQACEQMRAD8A+kNvuaTb7mnUUAN2+5o2+5p1FADdvuaNvuadRQA3b7mjb7mnUUAN2+5o2+5p1FADdvuaNvuadRQA3b7mjb7mnUUAN2+5o2+5p1FADdvuaNvuadRQA3b7mjb7mnUUAN2+5o2+5p1FADdvuaNvuadWB4q8Z6V4P024vNWF2628BnZba1eU7c4+8BtXnj5iB09aAN3b7mjb7mkglWe3jmQELIoYA9cEZp9ADdvuaNvuadRQA3b7mjb7mnUUAN2+5o2+5p1FADdvuaNvuadRQA3b7mjb7mnUUAN2+5o2+5p1cd8S/Et3oHhyGz0PnXNauF0/TR/ckfrIfZFy2fpQBpWPiWLVfFN3pOlwSXMGngpfXwcCKGbjEI/vvg5bHC8A8nFbu33NZfhfw7aeFPDdpo9huaO3T55X+/NIeXkY92ZiSfrWtQA3b7mjb7mnUUAN2+5poaNpGjWQF0ALKGGVB6ZH4H8qzPFmvJ4X8JalrUieZ9jgZ0j/AOej9EX8WIH40eGtIfRtEjjvZRPqE37+/ucY86dgN7fQYwB2VVHQUAau33NG33NZS+LNAdXKaxZsEUNxMDuBbaCv94FvlBGeeOtaFlfWupWaXVhcR3ED5CyRtuUkEgjPqCCD6EGgCXb7mjb7mnUUAN2+5o2+5p1VrnUbKzurW3u7qGGa8cx28cjgNKwUsQo7nAJoAn2+5o2+5qFtQs01JNPa6hF7JEZktzIPMZAQCwXrgEgZ96sUARyNHEu6WQIuQuWYAZJwB+JIFOKgDJY1Hd2lvf2ctpeQpPbzIUkikGVdSMEEVzfgq/nb+1/D2pStcXOh3f2cSync01u6CSFmPc7G2k9yhPegBlx8QNMtdMg1e4tL5NDndVTVwImgAdtquQHMiqSQMlBjPOK6srxwT+deTeKvEPgnW9Jm8K2+uaJp3h+zcfbYLe6hSW5Ebb/s0EQYEZYDLY5+6oJJK+t0AYXh3xJFrrXlpNBJYapp8givLGVwzRkjKupH3o2HKtxn0BBA3Nvua8/+J0E/h8Wnj/R42a80T5b6JP8Al7sWP7xD7r99SemD613dndwahYwXlnIstvcRrLFIvR0YZBH1BoAkB+Yj0p1MH+tan0AFFFFABRRRQAUUUUAFFFFAGH4wv9U0nw1d6npEthE1jDJcTG+jdkMaIWONrAg8dea5LwH4l8eeN/Bdr4hxoGnpd7zDA9tO5KqxXJPmDGSp7HjFM/aC1s6N8HNSjiJE+pSR2MWO+9ssPxRXFaPhXxJoHh3wfpOjQDUpGsbOOArFo92zOyqASAIuSTk0AMtPiDf6Z8RLTwb4ysLa2utRhMun31jKzw3GM5QqwBRvlPcjp6iug8WeIJdEs7S306JJ9W1O4FpYQyZ2mQgku+OdiKGdsdlx1IridO8Iar4x+L1v498RWUml6dpdv5Ok6fOR58h+b97KoJCcuxC5zwucY51DN/av7QiwSENDoegmSNSfuzTygFv++EA/GgDt9OtHsdPht5rqW7lQfvJ5j80jE5Leg5JwBwBwMAVZqlq+qRaPpcl5MjSbSqRxJjdLI7BERc8ZZmVRnjnmuR8SeNdZ8IajoceqWdpfDW7g2cNrZblkhnIzGN7HDoTwzbVxnOD0oA7uiuT0PX/EM/jnUdC1jSGFnaWscyarFbPFBJI2CYkLM2/Ab7wxgq2QOK6F9VsI9Wj0t7yFb+WIzJbFx5jRg4LBeuM8ZoAxNZvrqD4heGLOGd0trmK9M0QPyyFVjK5Htk/nXTVyOv8A/JUPCH/XG/8A/QI6d4s8V3OkWery6abVV0axa8u5rpWdc7WKQqFIO5tuSc8Arwd3AB1lYvieHVRpv27w/Ixv7LMyWpP7u8UD5oW64LAcMOVOO2Qc6bxwlt8Jf+E1ubQxf8Spb/7K7Y+ZkDKmfdiBnHfpWx4b1K41nwtpep31strcXlpFPJArbhGzoGK59s0AP0DW7PxJ4fstY0xy9reRCWPcMEZ6qR2IOQR6g1oV598MJTZ6z418Pb90em65JNAueI47hRKFHsGL/nXoNABRXO6p4xi0rUpbNtD167MeMzWmmvLG2QDww4PXH1zVT/hYMP8A0LPij/wUSUAdbRXLW/juG4uoof8AhHfEkXmOE3y6VIqrk4yT2A9a6mgAooooAKKKKACuI+Mv/JG/E3/Xkf5iu3rlPiFoGs+LPCt94f0s2MEF/B5cl1cyOWj+boI1XB4HXcPpQB0Wm/8AIKtP+uKf+girNZ2iJqcOnpBrEVokkKqiPazM4kAHUhlG36ZP1rRoAKKKKACiiq+oX9rpWnXF/qM6W9rbRtLNK5wEUDJJoAsUVx2geJNd8YaV/bWkW9ppOkyqWs21GJpZrlP+ehVXURKe3LEjnjpV2y8W/bPBumaytkftWphFtrMSf6yRs4AbH3MAtux9wE47UAdJRXCav4z1nw94u0bw/eWtnqN3r0Uv2P7KGiEEke0t5m5mJQKxO4AH5D8vPE+l+MNXin8Uf8JJo80FloShob8WzQLfAKS/lozNnleCGOdwHBoA7SivPofHGvj4gP4Pk02zutQbTo7/AM6B2SKzDOVZJSSS2PlwVA3ZHCjJGlYeLrq3+Iz+DdeFs91LYDULO6tUMaypuKMjIzMVYEEg5II9MUAdfXlrufEv7TCQOS1p4T0gyKp5AubjAz/37Yf9816lXk3wsYXnxc+J99JkyDULe2GeyxrIv9B+VAHrNFFFABRRRQBxHxgUn4Y3zg4WK5s5ZD6It1EzH8ACfwrf8V3mkWHhW+n8S3X2XStgS5lyR8rMF2/LzyWC8etXNX0u11zRbzS9Rj8y1vIXglX1Vhg49Dz1rK0O1uL3w82h+K7MXUtoqwTSTRBob1B9yUZyMkAEr1Vs9tpIBy7G11r9oOAJ5Zt/DOhl8jA2TXD4A+gjX9a27XU/D/hHwPBJoWnSRaUIZrmGC3AT90N0jy5dgApHzZJ/iHritd/Cfh+RIVk0WwcQAiMNbqdoJyR05yeTnqeetV/Gvh+bxP4Tn0i2lSNZ5IvNVyVEsSyKzx7gDt3KCucHGehoAs6Qgm8H2Y0q6ul86zVre4viZpl3JlWk3HLMM5IJ9qzv7E8Xf9DfB/4KE/8Ai63dNhuoLBEv5lmuMszFF2quWJCr7KCFBPJA5q1QBg6dpfiO31COXUfEkN7bLnfAunLEX4OPmDnGDg9O1cvfzxa3+0TpOnl1K+HtFnvtv/TWd1ix+Cc/8C969GrgPDngVLnxV4p1/wAXaPay3d/qe2yMypKVtIo1SMgjO3dgkr9MigChoOqWupfGrxNrTSedHYx2ugWRi+be7ZmmH/ATjPsh9K6zTvFZv/Fs2iSadLblLM3glduQnmmNd64+QttZlGScA5weK04NC0m11AX1tplpDdiMRiaOFVYKBgDIHQDj6cV574gW/kv/ABbJ4Va4i1nVVj04Wtxa7mJRCqzo4fEcOJSdzjqrYySAQDuvC3iCPxT4atNat4Hghu97RI5BJQOVVuP7wAb8a5/w4ol+Lvjltu6L7PpsTZHBcRysR/3y6fmK6KytI/C/hK0sbSCW5TTrSO3iigTLybVCqAOnOByTgdSQMmq/hPQptGsLqbUGR9T1O6e9vmQkqJGAARSedqIqoPXbnjNAD08G+GIpFkj8OaSjqQystjECCO4O2tqiigCG9tIdQsLizu0EkFxE0UqH+JWGCPyNed/Ai/nk+HsuiXz77rw9qE+lyE9cRtlf0bH/AAGvSq8o+FebX4sfE/TyNqrqMFyqlhyZVkYn+X50AeqD/WtT6YP9a1PoAKKKKACiiigAooooAKKKKAPG/isf+El+Mnw+8IL80cdw2qXcfYonK5+ojkH/AAKvZK5gfDXwYLgXA8N6f546TeSPM+u7rn3zXSQxJBCkMK7Y41Cqo7ADAFAD68702L7J+0brQkPN94ft54+OoSVkbn2OPzHtXolcZ44spdO1jRvGVnE0h0dpIr+ONctJZygCQgDqUIWTHorYoA2PFPivQ/CVhb3fiO6FvDcXKW8P7ppC0p5AAUE9ic9sVw2uzR+I/wBpHw5pIZXh8O6dcajMM8ebJtRV+qho3/4EK9Ev9N0vxHp0AvI47u33LcW8qOQVbHyyRupBU4PDKQcH3qtJ4R0OSOAfYRHJbs7RTwyPHMpf7581SH+bA3c/N3zQBKNftfN1UkN9m0pAZ7hRld+0s6ADklV2k/72OoNY2geJBrHjfUdMvdEhtNR06ygle4SUTNGkxYiF22ja3yhioJBz14yems7K206zS1sYI4II87Y41wBk5P4kkknuTXC+ANK13TtV1WfVLJoLvUdUubvULiTDK0Y+S3iiIOSAgVs9sFTyeADT1/8A5Kh4Q/643/8A6BHWF8cUkufB+l6JETFFr+vWenXUi8bY3Ykkn/gCit3X/wDkqHhD/rjf/wDoEdUvF+st/wAJlY6Br1hpS+FLi0ae8v8AU5xHmUE7I4skDcCqMT1AOeMcgGP8bpIrnw3oXgmzZY5fEOqW1n5KHBW3Vgztj0GE/OvUURYo1SNQqKAFUDgAdq8s0PwpB4j+Ktr4rtbWWLRNEgeKxuLpnebUJ3GGl3SEuY1X5VJOCRleK9E17WoNA0We/uFaUoAsUEf355DwkajuzMQB9aAON+HyNP8AEb4i6iuDDJqdvaqyjA3w26hh9fmGa9DrnPAnh6fw34Vjg1Blk1O7mkvdQkTo1xKxd/qATtB9FFdHQBzuqXHjRNSlXRdM0GayGPLku9RmikbgZyqwMBzn+I8flVT7X8Rf+gN4X/8ABvcf/I1dbRQBy1vdePzdRC70jw2sBcea0WqzswXPJANuATj1IrqaKKAKOsaPZ67pzWOpLK0DMGPk3EkLAg8YdGDD8DXPf8Ku8L/889V/8Hl7/wDHq6+igDkP+FXeF/8Annqv/g8vf/j1V7/4eeD9L025v77+1Ira1iaaaQ65fYVFGSf9d6Cu3rivGhbxF4g0rwZDkwXBGoasR0FpGw2xn/rpIAv+6r0AcfZaD9vsILyD4c+JBFcRrKm/xbIrbWGRlTc5Bweh5FT/APCLSf8AROvEX/hXv/8AJNdk/iS41fxrqnhjRr23sbnS7eKWd57V5mcyDIK/MqhQMZySSSeBjJi8DeML3XdW8QaBrcEEereH7hIriW1BEM6SKWjkUMSVJCnKknHrzQByX/CLSf8AROvEX/hXv/8AJNH/AAi0n/ROvEX/AIV7/wDyTXr1FAHkP/CLSf8AROvEX/hXv/8AJNH/AAi0n/ROvEX/AIV7/wDyTXr1ZHiLRZ9ctbS2ivRbQx3kU11G0PmLdQqctCwyMBuOeenQjigDzj/hFpP+ideIv/Cvf/5Jq38c4b4fAG/isbd4Nkdt9ogEhdo4w6bl3fxYwMnuATWhoWnpH8Xrufwpax2egQaebfUDboEt57zzPlCqOC6KGDMBxkKTkYHoE0MdxC8M8ayxSKUdHUFWUjBBB6igDm7fwb4bstDghka4n0y3gUBLvUp5bfy1UYyjuUK4Hpin3njbw3p39hJeXHknWLn7PpiG3bMrZ2gqMfKp3DDHAww7GrFv4N0G1ESx2JeGEgw2808ksMOOmyJmKJjttAxVzVdC07WvIOo2+97Zy8EqSNHJExGCVdCGXI4ODyOtAHnljNH4j/aZvpQ6vB4a0ZbeMA/8t5XyzD1wuVPuMV6JYajFq7XyxwkwW1wbcSNgrMVA3FfYMSh/2kaqsvhHQ5fIIsFhaCIwo9vI8LGMncUZkILKTyQ2QSSTyar+J9csPBXhGeeJEhMNu62NrDH/AKyRUJWNFUe1AHGfCu4h8QfEb4heJBIsjNqMemwgHlIoF25HsxwfwqPwnbS+L/jvrnjWP5tG0u1/sfT5u07g5kZT3VWLjPQ7uOhpfg/4W0G7+F+mw3SGfUBG39o7ZJIpC0jtJ5UwBBYAP91sjv3zXqVra29lax21lBHb28ShI4okCIijoABwBQBLXkvwiBj+JfxQjfAcawj7c84bzCD+NetV5X4cjOhftK+K7Fvkj13S7fUYfRjGfLb8clzQB6pXN6h4A0HVNQmvbsan50zbn8nWLuJc+yJKFH4AV0lFAHJf8Ky8Nemsf+D6+/8Aj1aGi+DdI0C+a703+0PNZDGftGqXNwuCQfuySMueBzjNbtFAGJrng/R/EdzHcaol4ZI02KbfULi3GM55ETqD16msv/hV3hf/AJ56r/4PL3/49XX0UAch/wAKu8L/APPPVf8AweXv/wAeo/4Vd4X/AOeeq/8Ag8vf/j1dfRQBwuo+AfBmlW6zX39rIruI0VNZv5HkY8hVRZSzHAJwATgH0rO03w74A1PWn0dP7bttTSPzfsd5qmowSPHnG9A8o3rnuucd67HW/DUOuaro1/Je3lrLpFybiMW0uwS5UqUfjlT3Hpkd64zV7aTxd8cvD8+jgm08JpcHUb1fumWZAq24P8TADLD+ENzg8UAb3/CrvC//ADz1X/weXv8A8eo/4Vd4X/556r/4PL3/AOPVXvPFniDW9a1DTPAFhp0yaXL5F5qOqTOsHngAmGNYwWZlyNxOAOnJqn8O/E/jTxVJPca3b6Ha2lleXFjeQW/nGdJoiVIBJKkZwc+lAHI6xoMWl+BJ/GMmn3lvaWztJNpN5qGpRXIiEmwjzDc4DkfMPkwcgd91d/D8M/Cs8EcqRasFkUMA2t3oOCM/89q5TxD4vuta1KaHUvAPiy+06zl32dommgQ3Uq8rJMWcEruwQm3AxuO44C+tx7/KTztvmbRv29M98e1AHJf8Ku8L/wDPPVf/AAeXv/x6j/hV3hf/AJ56r/4PL3/49XX0UAch/wAKu8L/APPPVf8AweXv/wAerc0Pw/p/h20kttKW4WOR/Mb7Rdy3DZwB96RmIHHQHH51p0UAFeTfDRftHxt+J92VZdtxaQLzkHCOD+Pyj869ZJwMmvKvgXH9vsvFfijHya7r1xNAwH3oVbCnP1LD8KAPUh/rWp9MH+tan0AFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAQWVja6baLa2EEdvAhYrFGuFXJLHA7ck8VPRRQAUUUUAV5bC1nvre8mgR7m1DiGUj5ow+A2PrgflViiigAqCaytri6t7ieFJJbYs0LsMmMkYJHocZGfQkdzU9FABRRRQAUUUUAFFFFABXIWnxI0e8+IaeEY47lLiWza6t7mWPZFchWIIjJ5bgMdwGCFOCa6+vK/jno93Do2meONDX/AImvha5F1wceZbkgSKfboT7bvWgD0+6uobKzmuruRYoII2klkboqqMkn6AVynw9tri8sr3xVqcbR3viCUXCROPmgtVGLeI++z5j/ALUjVma5rdv49tPDuhaM3mWevxLqF+39ywTaWRsdC7lY8em/0r0MAKAAMAcADtQBk67rA0e3X7JaNfalc/Ja2cbBWnYerH7qLnLMeAPUkA814Z8HHS7bXLefWT/wlGsmO81W9tkwY1csqLFn7qqEkVCckEbsdBU3iX4WaH4r15tX1S81dLkxiJRbX7xIiD+FQOgJ5Pua0vCPgbSfBUd2ukPeStdsrSyXlw0znaCAMnnAyePegDy7WfBvg9bPxHqUGnytbWWoWekpM99MxMrTIs8u4vyczhTnPMR967TxHa2Xwu8B6hP4J05YL68mht4FeR5F86RxGjNuJ4G7OO+Md62X+HugyeBZvCTxTNpk7GSTMx81nMvmly/Xdv5zWzquj2Ot6RLpmpw+faygBlLEHIIKkMDkMCAQQcgjNAHj/wAS/hto2meFbG587UNS8TXWpWtvFdXN7KzXsjyKHRl3bQpQOcKBtxx3z6HeeBktvBs/h/wZfv4bSeQs9xEhncAjDbd7ZBOAM547etadt4Zs4tUh1K7mudRvbdClvNeSBvIBAB2KAFBIHLY3Huawviuk6+A724t5pEdUFvGocqqSTSJEszEEf6sOzc8d+oBAA/wr4N1zQ9Shm1nxa+sWltbtDa2S6bDaRwklfmHlYzgKQAeBuNdlXD2eoHU/ivBplhqUlzZ6Bo5N3sm3CS4mcKgkxwWCRO3P9/NdxQAUUUUAFFFFABRRRQAV5x8VLWTRdS8P+P7ONmfQLkpfqi5LWUvyynHfZncPxPavR6juLeK7tpbe5jWWGZCkkbjKupGCCO4IoAWGWO4hSaB1kikUMjqchgRkEH0p9ch4P0jVPCV1L4cMb3WgRgyaVd7wWtkzzbSZOTtJ+Rufl4JBAz19ABRRRQAUUUUAFFFFAHM+OpPFY0NYPBFnbz3kz7ZZZ7gReTH3K8H5j0B7deelc74Vt/H0OpaVp1/4f0fQtBtHeSb+z7xpHf5HwuD1y7BiTknFekUUAcRoOi694S03VNM0qxt7t7rUbi7tryWfbGomcvmUfeLKTjCg7gByueNfwz4ej8FeDxY2xm1GeISXE8mAJLudiXdsdAWY8DPHAzXQUUAcN/wsCe18O+LtU1CxhZPD0jQo1vKdlzKIlYxgkZyJH8vdjk9h0rtbZ5ZLWJ7iMRTMgMiA5CtjkZ74NeWR+G9S1+Cz06C3v9M0681f+19Xs7uJQIj5xmaLzNo83fIBgLwozliNoPq9ABRRRQAUUUUAcb8U9fn0TwPcW+ljzNY1dhpunRA/M00vy5H+6CWz04rY8H+HIPCPg3S9BtiGSxt1jZwPvv1dvxYk/jWNoug3+t+Lj4u8U25tntkaDR9Mdgxs4zw8rkEjzXwOhwq4GSc12dADB/rWp9MH+tan0AFFFFABRRRQAUUUUAFFFFAEfnx+v6Gjz4/736GneWvpR5a0AN8+P+9+ho8+P+9+hp3lrR5a0AN8+P8AvfoaPPj/AL36GneWtHlrQA3z4/736Gjz4/736GneWtHlrQA3z4/736Gjz4/736GneWtHlrQA3z4/736Gjz4/736GneWtHlrQA3z4/wC9+ho8+P8Avfoad5a0eWtADfPj/vfoaPPj/vfoad5a0eWtADfPj/vfoaPPj/vfoad5a0eWtADfPj/vfoaiuVtb20mtbpBLBOjRyRspw6kYIP1Bqfy1o8taAOB+Fvw8g+HWnX0El81/LNOywSsGzFahmaOLnpy7sccbnNd558f979DTvLWjy1oAb58f979DR58f979DTvLWjy1oAb58f979DR58f979DTvLWjy1oAb58f8Ae/Q1m63Zy6pBbJaai9m0M4lYeWXSdQrAxyLkEod2cZHIHbitTy1o8taAMLw5oFl4eF3KkglvL6QSXMyxeWpwMKiIOERRwF56kkkkk7fnx/3v0NO8taPLWgBvnx/3v0NHnx/3v0NO8taPLWgBvnx/3v0NHnx/3v0NO8taPLWgBvnx/wB79DR58f8Ae/Q07y1o8taAG+fH/e/Q0efH/e/Q07y1o8taAG+fH/e/Q0efH/e/Q07y1o8taAG+fH/e/Q0efH/e/Q07y1o8taAG+fH/AHv0NHnx/wB79DTvLWjy1oAb58f979DR58f979DTvLWjy1oAb58f979DR58f979DTvLWjy1oAb58f979DR58f979DTvLWjy1oAb58f8Ae/Q0efH/AHv0NO8taPLWgBvnx/3v0NHnx/3v0NO8taPLWgBvnx/3v0NHnx/3v0NO8taPLWgBvnx/3v0NHnx+v6GneWtHlr6UAIvLkjoafSAY6UtABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAf/9k=)

We have created a GaussianB classification. Classifiers are trained using training data. We can use the fit () method to train it. After creating the classification, our model is ready to predict. We can use the predict () method as its parameters along with the test set properties.

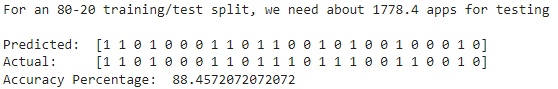
model = GaussianNB()

model.fit(X\_train,y\_train)

predicted = model.predict(X\_test)

**Accuracy of our Gaussian Naive Bayes model**

It’s time to test the quality of our model. We have made some predictions. Let’s compare the model’s prediction with actual target values for the test set.



Our model is giving an accuracy of **88.45%.** We can create random test datasets and test the model to get know how overall well the trained Gaussian Naive Bayes model is performing.

## A confusion matrix

To describe the performance of a classification models or a Classifier confusion matrix help for that. Confusion matrix also known as an error matrix. It helps us to find out, how many times our model has given correct or wrong output and of what type. Hence, it is a very important tool for evaluating classification models.

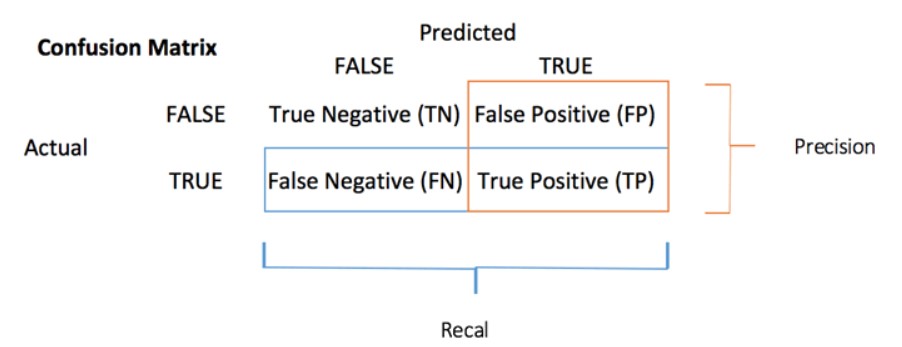
In confusion matrix 4 types of outcomes possible.

**TP: True Positive:** Predicted values correctly predicted as actual positive

**FP: False Positive:** Predicted values incorrectly predicted an actual positive. i.e., Negative values predicted as positive

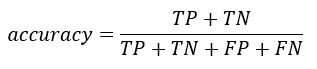
**FN: False Negative:** Positive values predicted as negative

**TN: True Negative**: Predicted values correctly predicted as an actual negative



Very clear explanation we can found at [Confusion matrix](https://en.wikipedia.org/wiki/Confusion_matrix) .

To compute the accuracy test from the confusion matrix below formula is being in used:



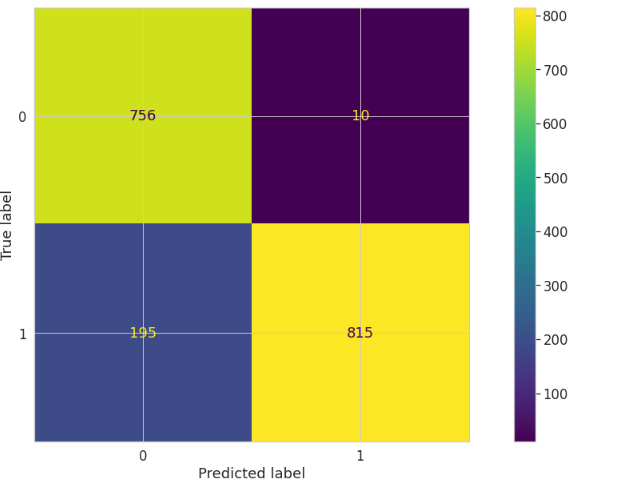
Our model predict 756 True Negatives and 815 True Positives!

#TN = 756

#FP = 10

#FN = 195

#TP = 815



|  |  |  |  |
| --- | --- | --- | --- |
| **N=1776** | **Predict No** | **Predict Yes** |  |
| Actual No | TN = 756 | FP = 10 | **766** |
| Actual Yes | FN = 195 | TP = 815 | **1010** |
|  | **951** | **825** |  |
|  |  |  |  |

# This is a list of rates that are often computed from a confusion matrix for a binary classifier:

**Accuracy**: how often is the classifier correct?

**(TP+TN)/total = (815+756)/1776 = 0.8845**

**Misclassification Rate:** How often is wrong?

**(FP+FN)/total = (10+195)/1776 = 0.115.**

Equivalent to 1 minus accuracy also known as Error Rate.

**True Positive Rate**: When it's actually yes, how often does it predict yes?

**TP/actual yes = 815/825 = 0.98**

Also known as "Sensitivity" or "Recall"

**False Positive Rate:** When it's actually no, how often does it predict yes?

**FP/actual no = 10/951 = 0.0105**

**True Negative Rate:** When it's actually no, how often does it predict no?

TN/actual no = 756/951= 0.794

Equivalent to 1 minus False Positive Rate also known as "Specificity"

After calculating Confusing matrix the accuracy is **0.884,** so its means that **Accuracy of our Gaussian Naive Bayes model** is good

# Extreme Gradient Boosting (XGBoost) Ensemble

XGBoost is a **ensemble** Machine Learning algorithm that overcome the Bias + Variance +Irreducible Error.

Ensemble has three method **Bagging**, **Booting** and **Stacking** .

XGBoost applies a better regularization technique to reduce overfitting, and it is one of the differences from the gradient boosting.

XGBClassifier\_eval\_set = [(X\_test, y\_test)]

model\_xgb = xgb.XGBClassifier(n\_estimators=500,

                          learning\_rate=0.05,

                          random\_state=42,

                          eval\_set=XGBClassifier\_eval\_set,

                          max\_depth=3,

                          eval\_metric='merror',

                          early\_stopping\_rounds=10,

                          verbose=True,

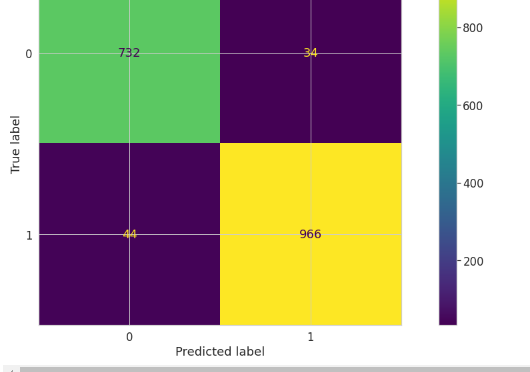
                          n\_jobs=-1)

model\_xgb.fit(X\_train, y\_train);

print("XGBoost Classifier Test Accuracy:", model\_xgb.score(X\_test, y\_test))

**Accuracy**: XGBoost Classifier Test Accuracy: **0.956081081081081**

Below are the confusion matrix





## K-mean Clustering

“Clustering” is the process of grouping similar entities together. Unsupervised machine learning technique is to find similarities in the data point and group similar data points together.

The Kmeans algorithm is a repetitive algorithm that tries to divide the database into predetermined special subgroups (clusters), which do not overlap because each data point belongs to a single group. Try to make the data points in the cluster as similar as possible, while making the clusters as different (by far) as possible.

# Analysis and Results

# Conclusions

# References

<https://www.python-graph-gallery.com/barplot/>

https://towardsdatascience.com/feature-selection-correlation-and-p-value-da8921bfb3cf

<https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/>

https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/

# Appendixes