# CREDIT CARD CUSTOMER SEGMENTATION

A major project report submitted in fulfillment of the requirements for the award of the Degree of Bachelor of Technology

In

**COMPUTER SCIENCE & ENGINEERING**


Submitted by

**Pooja (1501292214)**

**Vijay Kumar (1501292255)**

**Vishal Kumar (1501292258)**

Under the Guidance of

**Prof. Mohini Prasad Mishra**

**Dept. of Computer Science and Engineering**




**Dept. Of Computer Science & Engineering**
**Gandhi Engineering College, Bhubaneswar**



## Declaration

We hereby declare that the project entitled "CREDIT CARD CUSTOMER SEGMENTATION" submitted for the Bachelor in Technology Degree is my original work and the project has not formed the basis for the award of any degree, associate-ship, fellowship or any other similar titles.

Signature of the Students:

1.

2.

3.

Place:

Date

# CERTIFICATE

*This is to certify that the project work entitled "**CREDIT CARD CUSTOMER SEGMENTATION**" is the bona fide work carried out by **POOJA (1501292214)**, **VIJAY KUMAR (1501292255)** and **VISHAL KUMAR (1501292258)** students of BACHELOR OF TECHNOLOGY, GANDHI ENGINEERING COLLEGE during the academic year 2015-19 in partial fulfilment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING.*

Mr. Sudhanshu Shekhar Bisoyi — Prof. (CSE) & GUIDE

Mr. Mohini Prasad Mishra — HOD (CSE)

{EXTERNAL EXAMINER}

# ACKNOWLEDGEMENT

It is a great pleasure and privilege to express our profound sense of gratitude to our esteemed guide Mr. Sudhanshu Shekhar Bisoyi, Asst. Prof. (Dept. of CSE), who helped & coordinated us in completion of the project. We also sincerely thank to Mr. Mohini Prasad Mishra, HOD (Dept. of CSE) and all the teachers of our Department for their suggestions, motivation and support during the project work and keen personal interest throughout the progress of my project work.

I express my thanks to all my friends, my family for their timely, suggestions and encouragements.

Pooja

Vijay Kumar

Vishal Kumar

# CONTENT

# ABSTRACT

Customer segmentation is the practice of dividing a customer base into groups of individuals that are similar in specific ways relevant to marketing, such as age, gender, interests and spending habits.

It has been crucial for credit card operators to conduct targeted marketing with effective customer segmentation in recent years. The clustering technique using data mining is the most effective tool for dividing the customers into various segments.

The dataset we select for this project contains 8950 rows and 18 columns. Apart from data cleaning activities such as Missing value imputation and Outlier treatment, we will also use dimensionality reduction techniques such as Principal Component Analysis to reduce the dimensions. Also we derive certain KPIs that we find suitable for the analysis.

In this project we select K-Means clustering for segmentation purpose. The objective of the project is to demonstrate the working of this clustering technique. While we can get an indicator for the best number of segments, it is always suitable to create the segments as per the business requirement.

**Technology used:**

- Machine Learning using Python

# CHAPTER-1

# INTRODUCTION

## 1.1 Motivation and Objective

Machine Learning has become increasingly popular tool in almost every industry out there be it Smartphone Industry, Healthcare or Banks. With Machine Learning we can achieve so much. We will be using machine learning algorithms along with some data analysis techniques in our project.

Each individual is so different that ideally we would want to reach out to each one of them in a different way.

Problem: The volume is too large for customization at individual level

Solution: Identify segments where people have same characters and target each of these segments in a different way

Our project focus is to divide credit card customers into various segments using clustering techniques.

## 1.2 Problem Statement

The dataset we select for this project contains 8950 rows and 18 columns. We have to create multiple segments and divide the customers into those segments accordingly. Also in order to do that we need to clean the dataset and it also requires feature engineering.

## 1.3 Machine Learning

Machine Learning is an idea to learn from examples and experience, without being explicitly programmed.

*"A computer program is said to learn from experience E with some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." -Tom M. Mitchel*

### 1.3.1 Examples of Machine Learning

There are many examples of machine learning. Here are a few examples of classification problems where the goal is to categorize objects into a fixed set of categories.

**Face detection**: Identify faces in images (or indicate if a face is present).

**Email filtering**: Classify emails into spam and not-spam.

**Medical diagnosis**: Diagnose a patient as a sufferer or non-sufferer of some disease.

**Customer segmentation**: Create segments and divide customers into segments.

## 1.3.2 Need of Machine Learning

Machine Learning is a field which is raised out of Artificial Intelligence (AI). Applying AI, we wanted to build better and intelligent machines. But except for few mere tasks such as finding the shortest path between point A and B, we were unable to program more complex and constantly evolving challenges. There was a realization that the only way to be able to achieve this task was to let machine learn from itself. This sounds similar to a child learning from its self. So machine learning was developed as a new capability for computers. And now machine learning is present in so many segments of technology, that we don't even realize it while using it.

Finding patterns in data on planet earth is possible only for human brains. The data being very massive, the time taken to compute is increased, and this is where Machine Learning comes into action, to help people with large data in minimum time.

If big data and cloud computing are gaining importance for their contributions, machine learning as technology helps analyze those big chunks of data, easing the task of data scientists in an automated process and gaining equal importance and recognition.

The techniques we use for data mining have been around for many years, but they were not effective as they did not have the competitive power to run the algorithms. If you run deep learning with access to better data, the output we get will lead to dramatic breakthroughs which is machine learning.

## 1.3.3 Kinds of Machine Learning

There are three kinds of Machine Learning Algorithms.

**a. Supervised Learning**

**b. Unsupervised Learning**

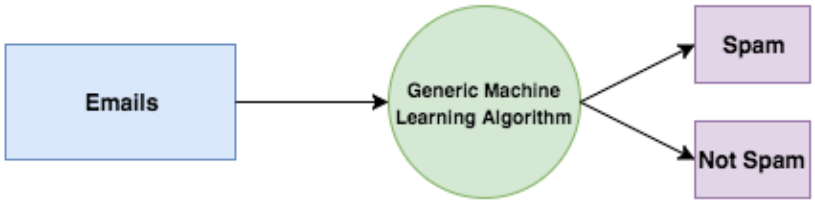
**c. Reinforcement Learning**

### 1.3.3.1 Supervised Learning

A majority of practical machine learning uses supervised learning.

In supervised learning, the system tries to learn from the previous examples that are given. (On the other hand, in unsupervised learning, the system attempts to find the patterns directly from the example given.)

**Example**:



Supervised learning problems can be further divided into two parts, namely classification, and regression.

**Classification**: A classification problem is when the output variable is a category or a group, such as “black” or “white” or “spam” and “no spam”.

**Regression**: A regression problem is when the output variable is a real value, such as “Rupees” or “height.”

## 1.3.3.2 Unsupervised Learning

In unsupervised learning, the algorithms are left to themselves to discover interesting structures in the data.

This is called unsupervised learning because unlike supervised learning above, there are no given correct answers and the machine itself finds the answers.

Unsupervised learning problems can be further divided into association and clustering problems.

**Association:** An association rule learning problem is where you want to discover rules that describe large portions of your data, such as “people that buy X also tend to buy Y”.

**Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

**Examples of Clustering Applications:**

**Marketing:** Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs

**Market Segmentation:** Grouping people (with willingness, purchasing power, and Authority to buy etc…) according to their similarity in several dimensions related to product under consideration

**Sales Segmentation:** Clustering can help you to identify what type of customers can buy what type of products

**Credit Risk:** Segmentation customers based on their credit history ν Operations: High Performer segmentation & promotion based on person’s performance

**Land use:** Identification of areas of similar land use in an earth observation database

**Insurance:** Identifying groups of motor insurance policy holders with a high average claim cost

**City-planning:** Identifying groups of houses according to their house type, value, and geographical location

**Earth-quake studies:** Observed earth quake epicenters should be clustered along continent faults

## Applications of customer segmentation

Customer segmentation can help other parts of your business. It will allow you to:

- Improve customer retention by providing products tailored for specific segments
- Increase profits by leveraging disposable incomes and willingness to spend
- Grow your business quicker by focusing marketing campaigns on segments with higher propensity to buy
- Improve customer lifetime value by identifying purchasing patterns and targeting customers when they are in the market
- Retain customers by appearing as relevant and responsive
- Identify new product opportunities and improve the products you already have
- Optimize operations by focusing on geographies, age groups etc. with the most value
- Increase sales by offering free shipping to high frequency buyers
- Offer improved customer support to VIP customers
- Gain brand evangelists by incentivizing them to comment, review or talk about your product with free gifts or discounts
- Reactivate customers who have churned and no longer interact with you

### 1.3.3.3 Reinforcement Learning

A computer program will interact with a dynamic environment in which it must perform a particular goal (such as playing a game with an opponent or driving a car). The program is provided feedback in terms of rewards and punishments as it navigates its problem space.

Using this algorithm, the machine is trained to make specific decisions. It works this way: the machine is exposed to an environment where it continuously trains itself using trial and error method.

# CHAPTER-2

# PROCESS and SPECIFICATION REQUIREMENT

## 2.1 Process:

1. Understanding the Data
2. Exploratory Data Analysis
3. Feature Engineering
4. Missing Value and Outlier Treatment
5. Standardizing the data
6. Reducing Dimensions using PCA
7. Applying K-Means
8. Selecting optimum number of clusters using Silhouette Coefficient
9. Profiling

## 2.2 Software Requirements:

i. Python
ii. Anaconda (JUPYTER Notebook)

### 2.2.1 LIBRARIES USED:

i. NumPy
ii. SciKit Learn
iii. Scipy
iv. Pandas
v. Matplotlib
vi. Seaborn

# CHAPTER-3

# METHODOLOGY

## 3.1 Loading Packages

- `import pandas as pd`
- `import numpy as np`
- `import matplotlib.pyplot as plt`
- `import seaborn as sns`
- `import scipy.stats as stats`
- `import pandas_profiling`

## 3.2 Reading the data

```
cc = pd.read_csv("CC_General.csv")
```

## 3.3 Understanding the Data

- `cc.shape`

```
Out[435]:   (8950, 18)
```

We found out that training data has 8950 rows and 18 columns.

- `cc.columns`

```
Out[436]: Index(['CUST_ID', 'BALANCE', 'BALANCE_FREQUENCY', 'PURCHASES',
       'ONEOFF_PURCHASES', 'INSTALLMENTS_PURCHASES', 'CASH_ADVANCE',
       'PURCHASES_FREQUENCY', 'ONEOFF_PURCHASES_FREQUENCY',
       'PURCHASES_INSTALLMENTS_FREQUENCY', 'CASH_ADVANCE_FREQUENY',
       'CASH_ADVANCE_TRX', 'PURCHASES_TRX', 'CREDIT_LIMIT', 'PAYMENTS',
       'MINIMUM_PAYMENTS', 'PRC_FULL_PAYMENT', 'TENURE'],
      dtype='object')
```

- `cc.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8950 entries, 0 to 8949

Data columns (total 18 columns):
CUST_ID                             8950 non-null object
BALANCE                             8950 non-null float64
BALANCE_FREQUENCY                   8950 non-null float64
PURCHASES                           8950 non-null float64
ONEOFF_PURCHASES                    8950 non-null float64
INSTALLMENTS_PURCHASES              8950 non-null float64
CASH_ADVANCE                        8950 non-null float64
PURCHASES_FREQUENCY                 8950 non-null float64
ONEOFF_PURCHASES_FREQUENCY          8950 non-null float64
PURCHASES_INSTALLMENTS_FREQUENCY    8950 non-null float64
CASH_ADVANCE_FREQUENCY              8950 non-null float64
```

```
CASH_ADVANCE_TRX                     8950 non-null int64
PURCHASES_TRX                        8950 non-null int64
CREDIT_LIMIT                         8949 non-null float64
PAYMENTS                             8950 non-null float64
MINIMUM_PAYMENTS                     8637 non-null float64
PRC_FULL_PAYMENT                     8950 non-null float64
TENURE                               8950 non-null int64
dtypes: float64(14), int64(3), object(1)
memory usage: 1.2+ MB
```

## 3.4 Missing value and Outlier Treatment

### 3.4.1 Missing value Imputation

After exploring all the variables in our data, we now imputed the missing values and treated the outliers because missing data and outliers can have adverse effect on the model performance.

- `cc.isnull().any()`

```
Out[442]:
CUST_ID                              False
BALANCE                              False
BALANCE_FREQUENCY                    False
PURCHASES                            False
ONEOFF_PURCHASES                     False
INSTALLMENTS_PURCHASES               False
CASH_ADVANCE                         False
PURCHASES_FREQUENCY                  False
ONEOFF_PURCHASES_FREQUENCY           False
PURCHASES_INSTALLMENTS_FREQUENCY     False
CASH_ADVANCE_FREQUENCY               False
CASH_ADVANCE_TRX                     False
PURCHASES_TRX                        False
CREDIT_LIMIT                          True
PAYMENTS                             False
MINIMUM_PAYMENTS                      True
PRC_FULL_PAYMENT                     False
TENURE                               False
dtype: bool
```

- `cc.CREDIT_LIMIT.fillna(cc.CREDIT_LIMIT.mean(),inplace=True)`
- `cc.MINIMUM_PAYMENTS.fillna(cc.MINIMUM_PAYMENTS.mean(),inplace=True)`

### 3.4.2 Outlier Treatment

In statistics, an **outlier** is an observation point that is distant from other observations. An **outlier** may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the **data** set. An **outlier** can cause serious problems in **statistical analyses**.

For our ease we defined a function for outlier treatment using 99 percentile rule.

```
#function for treating outliers
def out(x):
    x = x.clip_upper(x.quantile(0.99))
    x = x.clip_lower(x.quantile(0.01))
    return x
```

We applied this function to all numerical variables to treat outliers if any.

## 3.5 Feature Engineering

Based on the domain knowledge, we came up with new features that might affect the segmentation. We created the following three new features:

#### 3.5.1 Monthly average purchase

```
cc["monthly_avg_pur"] = cc["PURCHASES"]/cc["TENURE"]
```

#### 3.5.2 Monthly cash advance

```
cc["monthly_cash_adv"] = cc.CASH_ADVANCE/cc.TENURE
```

#### 3.5.3 purchase by type:

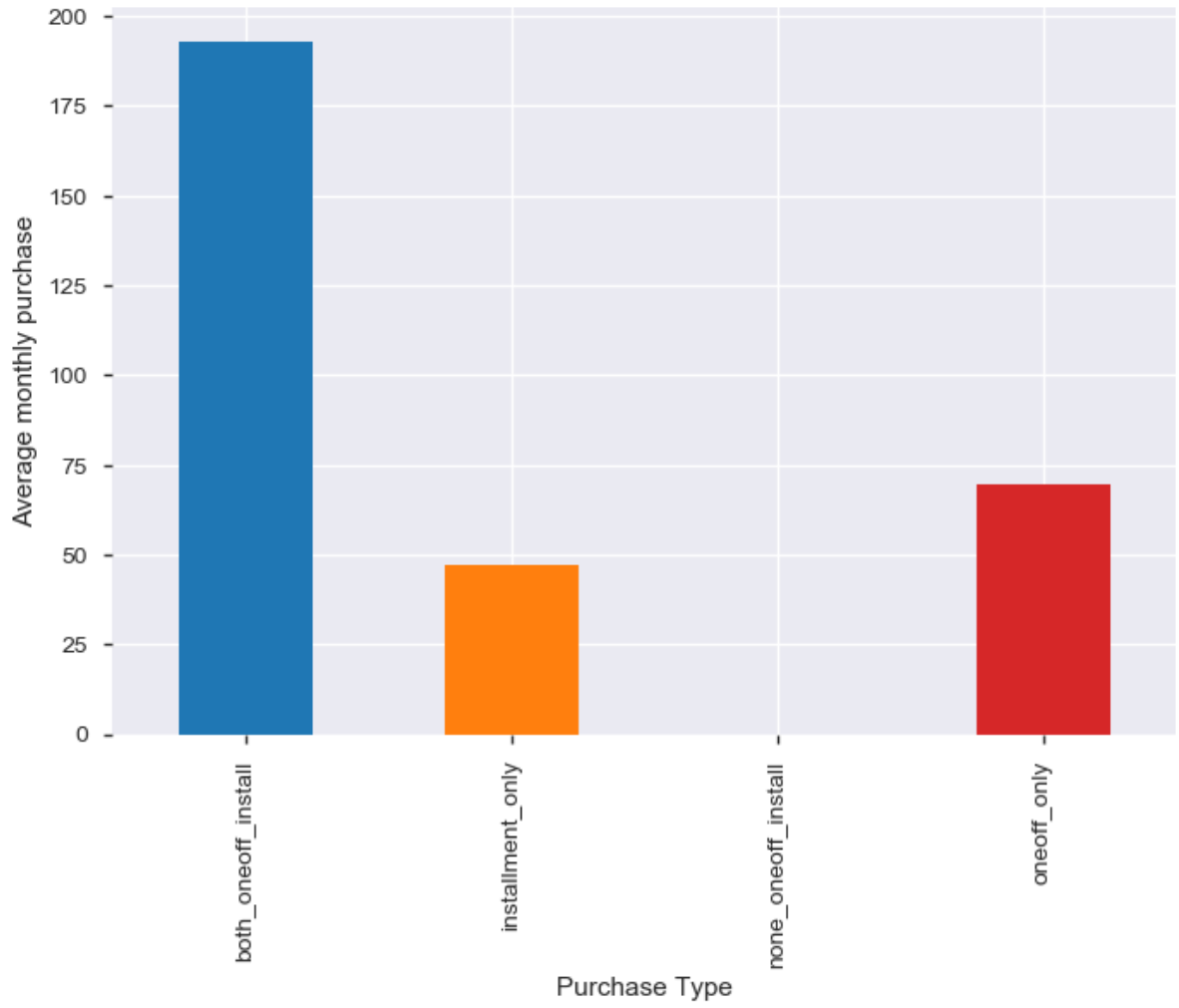
oneoff_only, installment_only, both_oneoff_install, none_oneoff_install

- 
```
def pur_type(x):
    if((x.ONEOFF_PURCHASES>0) & (x.INSTALLMENTS_PURCHASES==0)):
          return "oneoff_only"
    if((x.ONEOFF_PURCHASES==0) & (x.INSTALLMENTS_PURCHASES>0)):
          return "installment_only"
    if((x.ONEOFF_PURCHASES>0) & (x.INSTALLMENTS_PURCHASES>0)):
          return "both_oneoff_install"
    if((x.ONEOFF_PURCHASES==0) & (x.INSTALLMENTS_PURCHASES==0)):
          return "none_oneoff_install"
```
- 
```
cc["purchase_type"] = cc.apply(pur_type,axis=1)
```

## 3.6 Getting Insights from above features

### 3.6.1 Average monthly purchase by purchase type

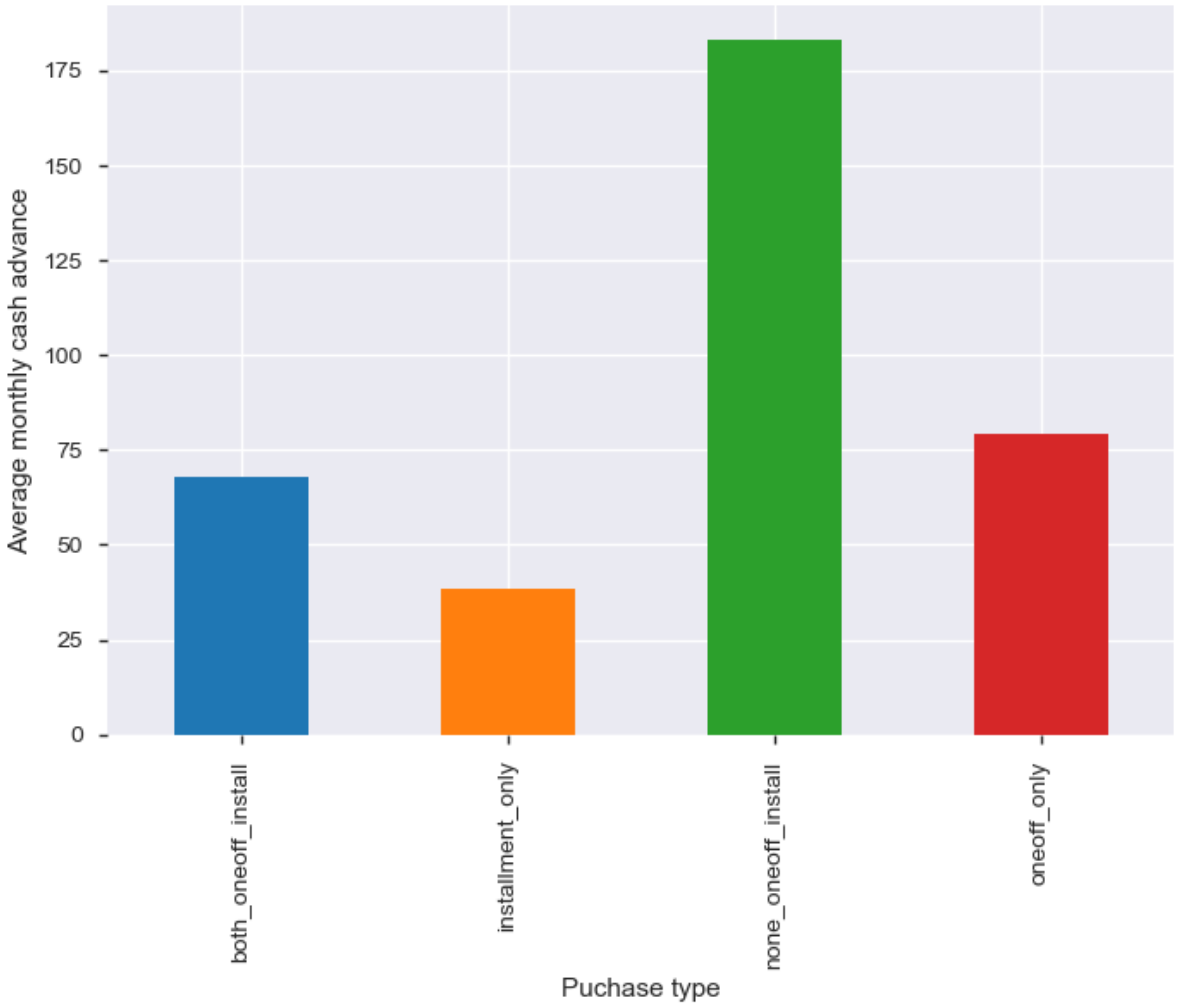
```
pt=cc.groupby(by="purchase_type")
pt["monthly_avg_pur"].mean().plot(kind="bar")
plt.xlabel("Purchase Type")
plt.ylabel("Average monthly purchase")
plt.show()
```




Average monthly purchase using both oneoff and installments are very high almost 193 whereas average monthly purchase using installments only is very low i.e around 47.

### 3.6.2 Average Monthly Cash Advance

```
pt["monthly_cash_adv"].mean().plot(kind="bar")
plt.ylabel("Average monthly cash advance")
plt.xlabel("Puchase type")
plt.show()
```

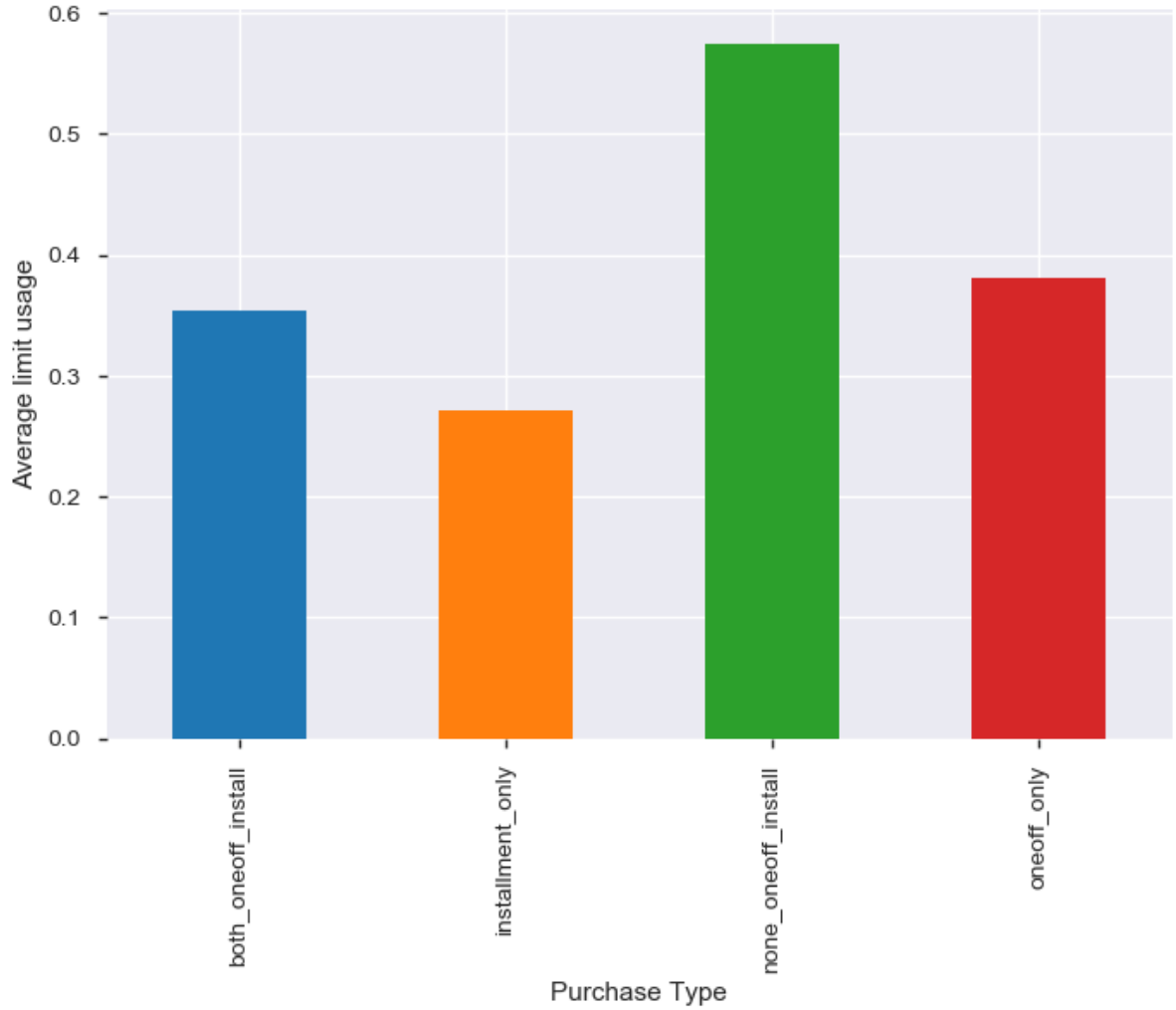



Customers who didn't do either of oneoff or installment purchases take highest monthly cash advance.

### 3.6.3 Average limit usage by purchase type

```
pt["limit_usage"].mean().plot(kind="bar")
```

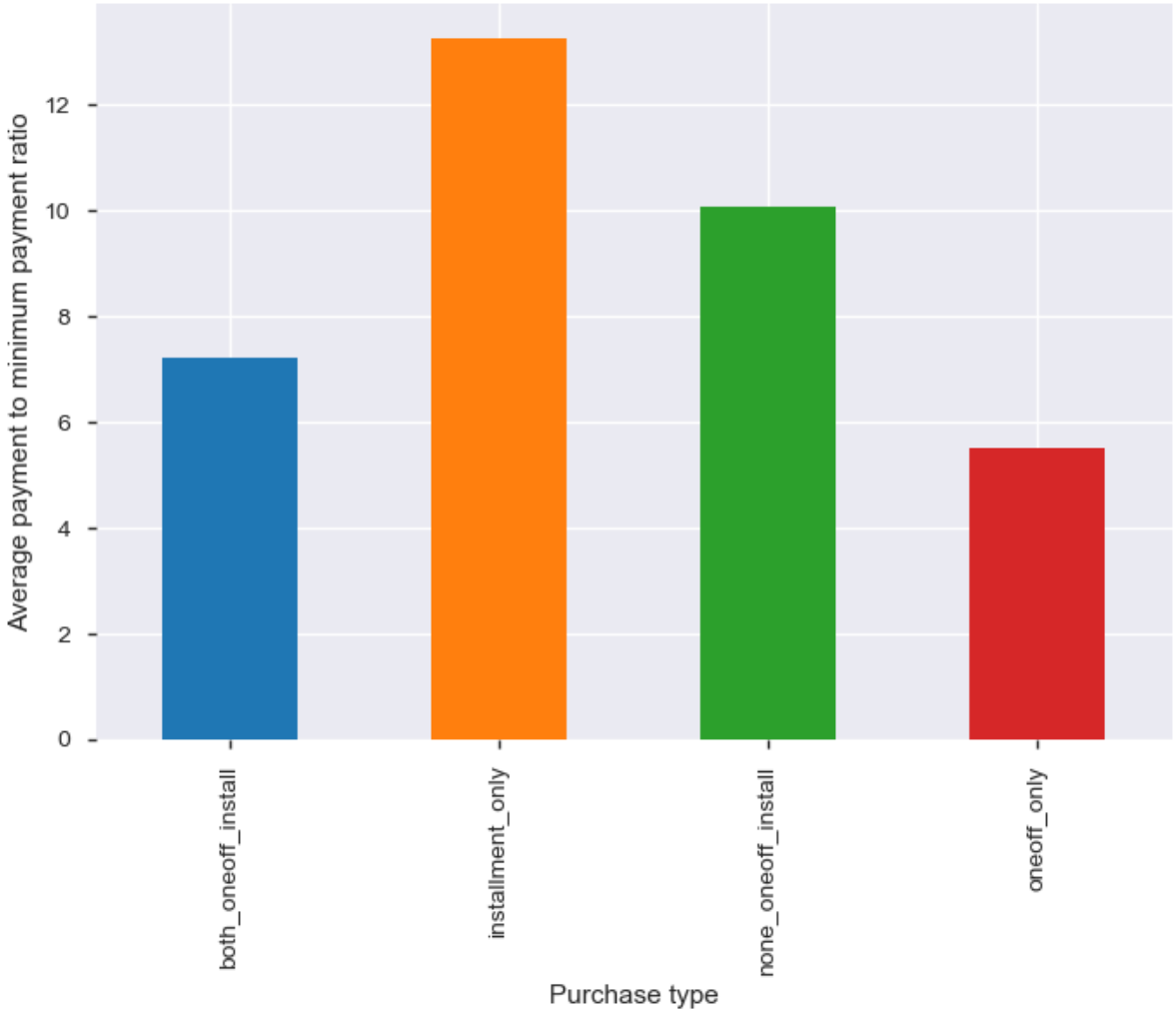
```
plt.xlabel("Purchase Type")
plt.ylabel("Average limit usage")
plt.show()
```




Customers with installments only purchases have good balance to credit limit ratio.

### 3.6.4 Average payments to minimum payment ratio

```
pt["pay_minpay"].mean().plot(kind="bar")
plt.xlabel("Purchase type")
plt.ylabel("Average payment to minimum payment ratio")
plt.show()
```




Customers with installments only purchases are paying their dues fast.

## 3.7 Standardize the Data

Our data may be at different levels or may contain different units. It will definitely not be suitable to move ahead and use this data without solving this problem. This can be done by standardizing the data. Calculate the mean absolute deviation:

- Calculate the standardized measurement (z-score)
- Using mean absolute deviation is more robust than using standard deviation. Since the deviations are not squared the effect of outliers is somewhat reduced but their z- scores do not become too small; therefore, the outliers remain detectable.

```
#importing necesarry packages

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

cc_scaled = sc.fit_transform(cc_dum)
```

## 3.8 Principal Component Analysis

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components. If there are n observations with p variables, then the number of distinct principal components is. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors (each being a linear combination of the variables and containing n observations) are an uncorrelated orthogonal basis set. PCA is sensitive to the relative scaling of the original variables.

PCA is mostly used as a tool in exploratory data analysis and for making predictive models. It is often used to visualize genetic distance and relatedness between populations. PCA can be done by eigenvalue decomposition of a data covariance (or correlation) matrix or singular value decomposition of a data matrix, usually after a normalization step of the initial data. The normalization of each attribute consists of mean centering – subtracting each data value from its variable's measured mean so that its empirical mean (average) is zero – and, possibly, normalizing each variable's variance to make it equal to 1; see Z-scores.[4] The results of a PCA are usually discussed in terms of component scores, sometimes called factor scores (the transformed variable values corresponding to a particular data point), and loadings (the weight by which each standardized original variable should be multiplied to get the component score).[5] If component scores are standardized to unit variance, loadings must contain the data variance in them (and that is the magnitude of eigenvalues). If component scores are not standardized (therefore they contain the data variance) then loadings must be unit-scaled, ("normalized") and these weights are called eigenvectors; they are the cosines of orthogonal rotation of variables into principal components or back.

```
#importing necessary libraries

from sklearn.decomposition import PCA

#initialising PCA with final 8 components

pc_final = PCA(n_components=8)

#fitting the model and transform on cc_scaled

reduced_com = pc_final.fit_transform(cc_scaled)
```

### 3.8.1 Dimensionality Reduction

Such dimensionality reduction can be a very useful step for visualizing and processing high-dimensional datasets, while still retaining as much of the variance in the dataset as possible. For example, selecting L = 2 and keeping only the first two principal components finds the two-dimensional plane through the high-dimensional dataset in which the data is most spread out, so if the data contains clusters these too may be most spread out, and therefore most

visible to be plotted out in a two-dimensional diagram; whereas if two directions through the data (or two of the original variables) are chosen at random, the clusters may be much less spread apart from each other, and may in fact be much more likely to substantially overlay each other, making them indistinguishable.

### 3.8.2 Limitations

As noted above, the results of PCA depend on the scaling of the variables. This can be cured by scaling each feature by its standard deviation, so that one ends up with dimensionless features with unital variance [14]

The applicability of PCA as described above is limited by certain (tacit) assumptions [15] made in its derivation. In particular, PCA can capture linear correlations between the features but fails when this assumption is violated (see Figure 6a in the reference). In some cases, coordinate transformations can restore the linearity assumption and PCA can then be applied (see kernel PCA).

Another limitation is the mean-removal process before constructing the covariance matrix for PCA. In fields such as astronomy, all the signals are non-negative, and the mean-removal process will force the mean of some astrophysical exposures to be zero, which consequently creates unphysical negative fluxes,[16] and forward modeling has to be performed to recover the true magnitude of the signals.[17] As an alternative method, non-negative matrix factorization focusing only on the non-negative elements in the matrices, which is well-suited for astrophysical observations.

### 3.8.3 Factor analysis

Principal component analysis creates variables that are linear combinations of the original variables. The new variables have the property that the variables are all orthogonal. The PCA transformation can be helpful as a pre-processing step before clustering. PCA is a variance-focused approach seeking to reproduce the total variable variance, in which components reflect both common and unique variance of the variable. PCA is generally preferred for purposes of data reduction (i.e., translating variable space into optimal factor space) but not when the goal is to detect the latent construct or factors.

Factor analysis is similar to principal component analysis, in that factor analysis also involves linear combinations of variables. Different from PCA, factor analysis is a correlation-focused approach seeking to reproduce the inter-correlations among variables, in which the factors "represent the common variance of variables, excluding unique variance".[45] In terms of the correlation matrix, this corresponds with focusing on explaining the off-diagonal terms (i.e. shared co-variance), while PCA focuses on explaining the terms that sit on the diagonal. However, as a side result, when trying to reproduce the on-diagonal terms, PCA also tends to fit relatively well the off-diagonal correlations.[46] Results given by PCA and factor analysis are very similar in most situations, but this is not always the case, and there are some problems where the results are significantly different. Factor analysis is generally used when the research purpose is detecting data structure (i.e., latent constructs or factors) or causal modeling.

## 3.8.4 Loadings

**Definition:**

Component loadings are the ordinary product moment correlation between each original variable and each component score.

**Interpretation:**

By looking at of component loadings one can ascertain which of the original variables tend to "load" on a given new variable. This may facilitate interpretations, creation of subscales, etc.

Loadings=Eigenvectors * sqrt(Eigenvalues)

Loadings are the covariance/correlations between the original variables and the unit-scaled components.

```
pd.DataFrame(pc_final.components_.T*np.sqrt(pc_final.explained_variance_))
```

## 3.9 Clustering

Clustering of data is a method by which large sets of data are grouped into clusters of smaller sets of similar data.

Cluster: a collection of data objects

- Similar to one another within the same cluster
- Dissimilar to the objects in other clusters

Clustering is unsupervised classification: no predefined classes

## Definition:

Given a set of data points, each having a set of attributes, and a similarity measure among them, find clusters such that:

- data points in one cluster are more similar to one another (high intra-class similarity)
- data points in separate clusters are less similar to one another (low inter-class similarity)

Similarity measures: e.g. Euclidean distance if attributes are continuous.

**Examples of Clustering Applications:**

- **Marketing:** Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- **Market Segmentation:** Grouping people (with willingness, purchasing power, and Authority to buy etc…) according to their similarity in several dimensions related to product under consideration

- **Sales Segmentation:** Clustering can help you to identify what type of customers can buy what type of products
- **Credit Risk:** Segmentation customers based on their credit history ν Operations: High Performer segmentation & promotion based on person's performance
- **Land use:** Identification of areas of similar land use in an earth observation database
- **Insurance:** Identifying groups of motor insurance policy holders with a high average claim cost
- **City-planning:** Identifying groups of houses according to their house type, value, and geographical location
- **Earth-quake studies:** Observed earth quake epicenters should be clustered along continent faults

## 3.10 K-Means Clustering

K-means clustering is one of the simplest and popular unsupervised machine learning algorithms.

Typically, unsupervised algorithms make inferences from datasets using only input vectors without referring to known, or labelled, outcomes.

We define a target number *k*, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster.

Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares.

In other words, the K-means algorithm identifies *k* number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible.

The *'means'* in the K-means refers to averaging of the data; that is, finding the centroid.

### 3.10.1 Working

To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids

It halts creating and optimizing clusters when either:

- The centroids have stabilized—there is no change in their values because the clustering has been successful.
- The defined number of iterations has been achieved.

**Input:** *k* (the number of clusters),
*D* (a set of lift ratios)
**Output:** a set of k clusters
**Method:**
Arbitrarily choose *k* objects from *D* as the initial cluster centers;
**Repeat:**
1. (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
2. Update the cluster means, i.e., calculate the mean value of the objects for each cluster
**Until** no change;

```
#importing kmeans for clustering
from sklearn.cluster import KMeans


#let's observe kmeans with k=3
k3_means = KMeans(n_clusters=3,random_state=121)
k3_means.fit(cc_scaled2)
```

### 3.10.2 Silhouette Coefficient

Silhouette refers to a method of interpretation and validation of consistency within clusters of data. The technique provides a succinct graphical representation of how well each object has been classified.

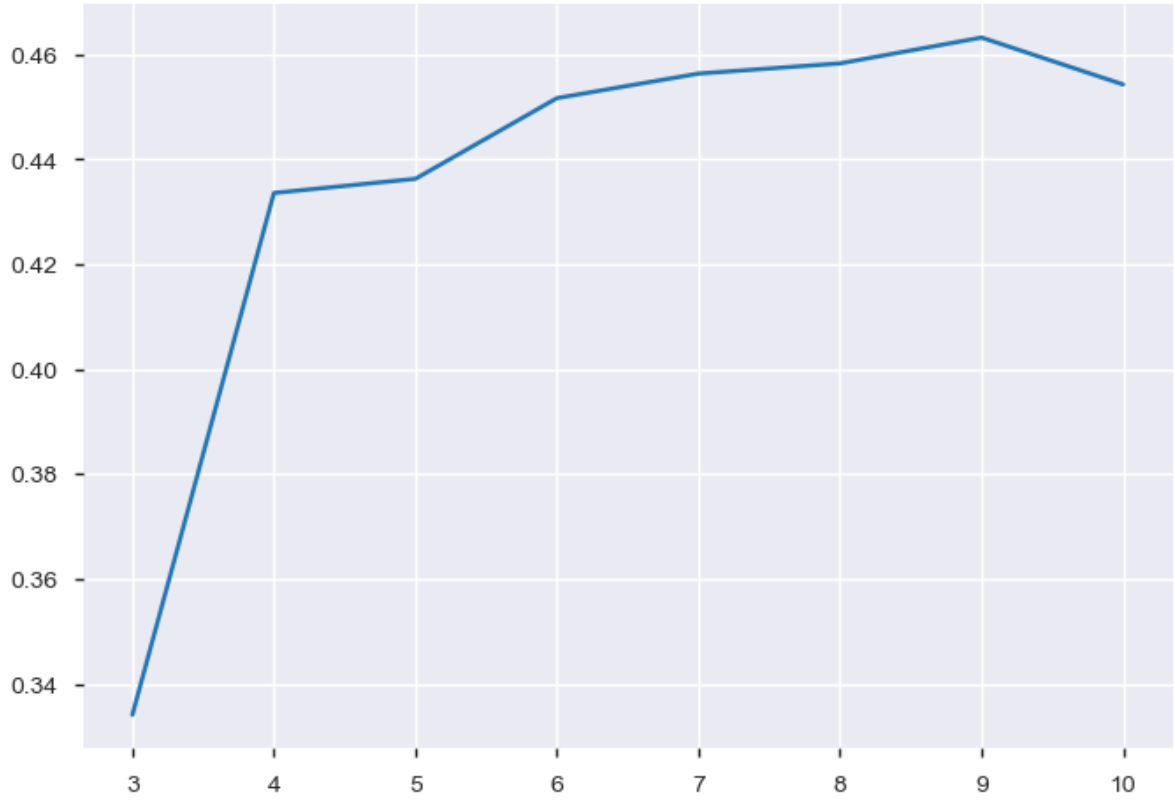
The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

The silhouette can be calculated with any distance metric, such as the Euclidean distance or the Manhattan distance.

```
#importing mterics for silhoutte coefficient calculation
from sklearn import metrics


#calculating sc for k=3 to 10
k_range = range(3,11)
sc = []
for k in k_range:
    km = KMeans(n_clusters=k,random_state=121)
    km.fit(cc_scaled2)
    sc.append(metrics.silhouette_score(cc_scaled2,km.labels_))


plt.plot(k_range,sc)
plt.show()
```




Based on Silhouette coefficient there can be 9 or 8 clusters depending on the business requirement. However the best score is for 9 clusters.

### 3.10.3 Profiling

Profiling refers to segment customers based on the clusters created by observing and analyzing the customers in segments. Mean gives a very good indication of distribution of data so we are finding mean for each variable grouped by each clusters.

The profiling is done using an Excel sheet.

# CHAPTER-4

# CONCLUSION

In the whole project we performed the following steps:

1. Understanding the Data
2. Exploratory Data Analysis
3. Feature Engineering
4. Missing Value and Outlier Treatment
5. Standardizing the data
6. Reducing Dimensions using PCA
7. Applying K-Means
8. Selecting optimum number of clusters using Silhouette Coefficient
9. Profiling
10. Getting Insights from these clusters

We found 9 clusters using Silhouette Coefficient and did profiling using these 9 segments.

## Insights with 9 clusters:

**Segment 1:** Customers of this segment do 100% purchases one off purchases.

**Segment 2:** This is the largest segment consisting of 27.8% of total customers. All the purchases made by these customers are done using both one off and installments.

**Segment 3:** Less transacting customers and prefer only purchasing through installments. Second highest segment consisting of 18.4% customers.

**Segment 4:** Customers with highest balance, highest monthly cash advance and highest minimum payment amount. Consists of 2.2% of customers.

**Segment 5:** Customers with lowest balance and highest payment to minimum payment ratio and almost zero limit usage. Lowest segment consisting of only 0.1% customers.

**Segment 6:** Customers with lowest credit limit and lowest payments. But these are newest customers. Overall low transacting segment. Consist of 6.3% of customers.

**Segment 7:** These are the customers with lowest transactions with 9% of total customers.

**Segment 8:** Third highest segment. These customers have taken cash advance only and not done any purchase at all.

**Segment 9:** These are the customers with highest transactions but consists only 1.3% of total customers.

# BIBLIOGRAPHY


## 1. Machine Learning Courses

**1.1** Coursera(Andrew NG)

**1.2** YouTube(SimpliLearn, Edureka)

## 2. Machine Learning Repositories

**2.1** UCI

**2.2** Kaggle

## 3. Websites

**3.1** www.analytixlabs.com

**3.2** www.kdnuggets.com

**3.3** www.analyticsvidhya.com

**3.4** www.machinelearningmastery.com