In [1]:
```
# Data Analytics Mini Project
# Name : Supriya Ananda Kore
# MIS : 111907055
# Batch : C
```

CLV = Expected No.of Transaction X Revenue per Transaction X Margin

Total profit generated from one client over their lifetime. CLV = Profit per year x Avg duration of relationship

Expected No.of Trasaction will be calculeted using BG/NBD model

Revenue per Transaction will be calculated using Gamma Gamma model

Beta Geometric Negative Binomial Distribution Model

In [2]:
```
import pandas as pd
import matplotlib as plt
import lifetimes as lt
```

In [3]:
```
data = pd.read_csv("CSV1.csv",encoding="cp1252")
data.head()
```

Out[3]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	United Kingdom

In [4]:
```
data.shape
```

Out[4]:
```
(541909, 8)
```

In [5]:
```
data.isnull().sum(axis=0)
```

Out[5]:
```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

In [6]:
```
#Remove time from data
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'],format="%d-%m-%Y %H:%M").dt.date
```

In [7]:
```
data.head()
```

Out[7]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01	3.39	17850.0	United Kingdom

In [8]:
```
#There are 135,080 missing values in the CustomerID column,and since our analysis is based on customer,
#Therefore i will remove missing values
data = data[pd.notnull(data['CustomerID'])]
```

In [9]:
```
#Keep records with non negative quantity
data = data[(data['Quantity']>0)]
```

In [10]:
```
#Add a new column depicting total sales
data['Total_Sales'] = data['Quantity']*data['UnitPrice']
necessary_cols = ['CustomerID','InvoiceDate','Total_Sales']
data = data[necessary_cols]
data.head()
```

Out[10]:

	CustomerID	InvoiceDate	Total_Sales
0	17850.0	2010-12-01	15.30
1	17850.0	2010-12-01	20.34
2	17850.0	2010-12-01	22.00
3	17850.0	2010-12-01	20.34
4	17850.0	2010-12-01	20.34

In [11]:
```
#Print records containing unique Customer ID
print(data['CustomerID'].nunique())
```

```
4339
```

In [12]:
```
#Check the Last order date
last_order_date = data['InvoiceDate'].max()
print(last_order_date)
```

```
2011-12-09
```

In [13]:
```
print(data[(data['CustomerID']==12346)])
```

```
       CustomerID InvoiceDate  Total_Sales
61619     12346.0  2011-01-18      77183.6
```

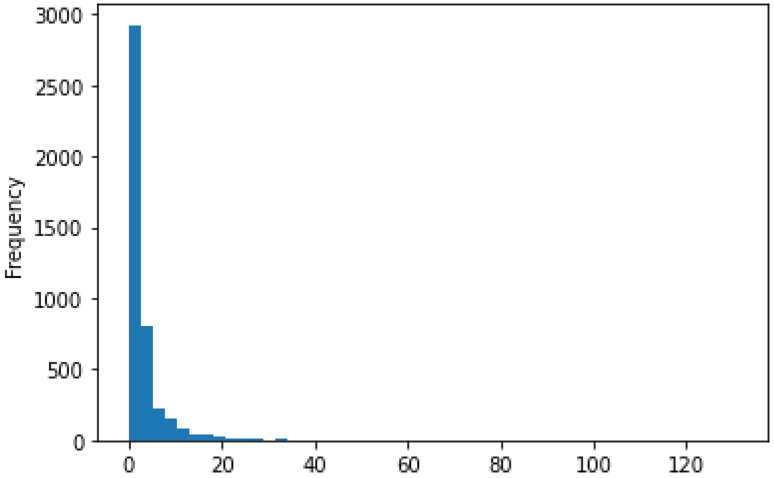
In [14]:
```
#Built-in utility function from lifetime package to transform the transactional data(one row per purchase)
#Into summary data(a frequency ,recency,and monetary)
lf_data = lt.utils.summary_data_from_transaction_data(data,'CustomerID','InvoiceDate',monetary_value_col='Total_Sales',ol
lf_data.reset_index().head()
```

Out[14]:

	CustomerID	frequency	recency	T	monetary_value
0	12346.0	0.0	0.0	325.0	0.000000
1	12347.0	6.0	365.0	367.0	599.701667
2	12348.0	3.0	283.0	358.0	301.480000
3	12349.0	0.0	0.0	18.0	0.000000
4	12350.0	0.0	0.0	310.0	0.000000

In [15]:
```
%matplotlib inline
#Create histogram to find out how many customers purchased item only once
lf_data['frequency'].plot(kind='hist',bins=50)
print(lf_data['frequency'].describe())
```

```
count    4339.000000
mean        2.864024
std         5.952745
min         0.000000
25%         0.000000
50%         1.000000
75%         3.000000
max       131.000000
Name: frequency, dtype: float64
```



In [16]:
```
one_time_buyers = round(sum(lf_data['frequency']==0)/float(len(lf_data))*(100),2)
print("Percentage of customers purchase the item only once:",one_time_buyers,"%")
```

```
Percentage of customers purchase the item only once: 35.7 %
```

In [17]:
```
#Frequency/Recency Analysis using the BG/NBD Model
import lifetimes
bgf = lifetimes.BetaGeoFitter(penalizer_coef=0.0)
bgf.fit(lf_data['frequency'],lf_data['recency'],lf_data['T'])
print(bgf)
```

```
<lifetimes.BetaGeoFitter: fitted with 4339 subjects, a: 0.00, alpha: 68.89, b: 6.75, r: 0.83>
```

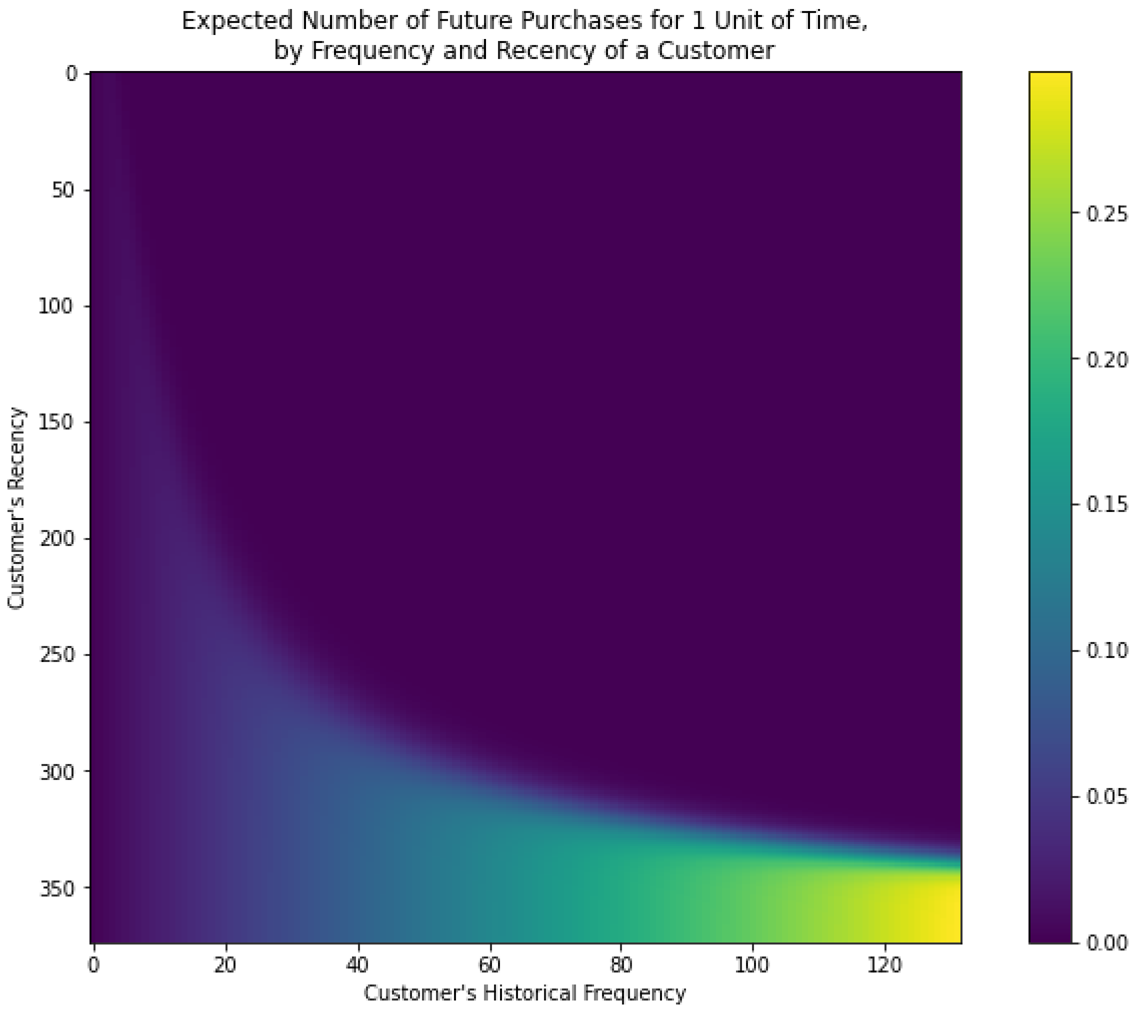
In [18]:
```
bgf.summary
```

Out[18]:

	coef	se(coef)	lower 95% bound	upper 95% bound
r	0.826433	0.026780	0.773944	0.878922
alpha	68.890678	2.611055	63.773011	74.008345
a	0.003443	0.010347	-0.016837	0.023722
b	6.749363	22.412933	-37.179985	50.678711

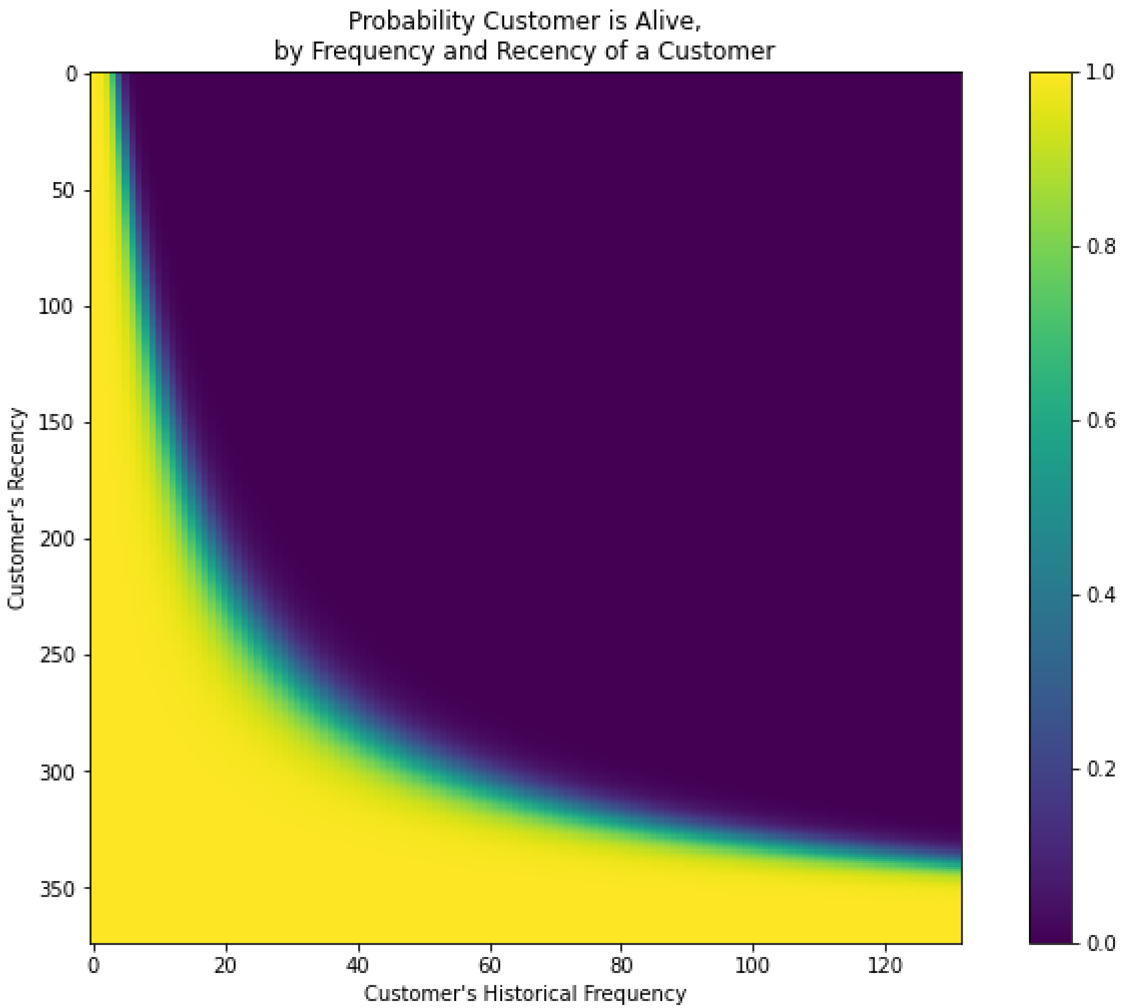
In [19]:
```
#Visualizing our frequency/recency matrix
%matplotlib inline
import matplotlib.pyplot as plt
from lifetimes.plotting import plot_frequency_recency_matrix
fig = plt.figure(figsize=(12,8))
plot_frequency_recency_matrix(bgf)
```

Out[19]:
```
<AxesSubplot:title={'center':'Expected Number of Future Purchases for 1 Unit of Time,\nby Frequency and Recency of a Cu
stomer'}, xlabel="Customer's Historical Frequency", ylabel="Customer's Recency">
```



In [20]:
```
#predict if the customers are surely alive
from lifetimes.plotting import plot_probability_alive_matrix
fig = plt.figure(figsize=(12,8))
plot_probability_alive_matrix(bgf)
```

Out[20]:
```
<AxesSubplot:title={'center':'Probability Customer is Alive,\nby Frequency and Recency of a Customer'}, xlabel="Customer's Historical Frequency", ylabel="Customer's Recency">
```



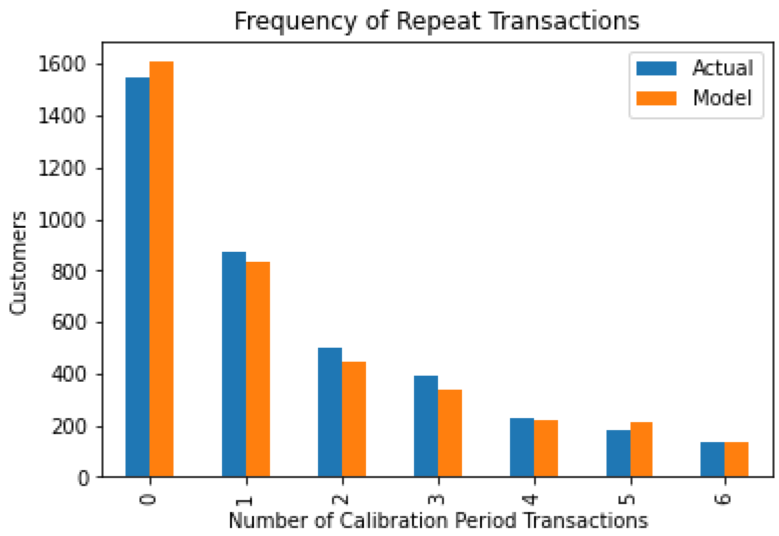
In [21]:
```
#Predict future transaction in next 10 days .Top 10 customers that the model expects them to make purchase in next 10 day
t = 10
lf_data['pred_num_txn'] = round(bgf.conditional_expected_number_of_purchases_up_to_time(t,lf_data['frequency'],lf_data['
lf_data.sort_values(by='pred_num_txn',ascending=False).head(10).reset_index()
```

Out[21]:

	CustomerID	frequency	recency	T	monetary_value	pred_num_txn
0	14911.0	131.0	372.0	373.0	1093.661679	2.98
1	12748.0	113.0	373.0	373.0	298.360885	2.58
2	17841.0	111.0	372.0	373.0	364.452162	2.53
3	15311.0	89.0	373.0	373.0	677.729438	2.03
4	14606.0	88.0	372.0	373.0	135.890114	2.01
5	12971.0	70.0	369.0	372.0	159.211286	1.61
6	13089.0	65.0	367.0	369.0	893.714308	1.50
7	14527.0	53.0	367.0	369.0	155.016415	1.23
8	13798.0	52.0	371.0	372.0	706.650962	1.20
9	16422.0	47.0	352.0	369.0	702.472340	1.09

In [22]:
```
#Assessing model fit
from lifetimes.plotting import plot_period_transactions
plot_period_transactions(bgf)
```

Out[22]:
```
<AxesSubplot:title={'center':'Frequency of Repeat Transactions'}, xlabel='Number of Calibration Period Transactions', y
label='Customers'>
```



In [23]:
```
#Customer's future transaction prediction for next 10 days

t = 10
individual = lf_data.loc[14911]
bgf.predict(t,individual['frequency'],individual['recency'],individual['T'])
```

Out[23]: 2.9830238639043056

In [24]:
```
#Check if there is correlation between monetary value and frequency in order to use gamma gamma model for CLV calculatio
lf_data[['monetary_value','frequency']].corr()
```

Out[24]:

	monetary_value	frequency
monetary_value	1.000000	0.046161
frequency	0.046161	1.000000

In [25]:
```
#Shortlist customers who had at least one repeate purchase with the company
shortlisted_customers = lf_data[lf_data['frequency']>0]
print(shortlisted_customers.head().reset_index())
```

```
   CustomerID  frequency  recency      T  monetary_value  pred_num_txn
0     12347.0        6.0    365.0  367.0      599.701667          0.16
1     12348.0        3.0    283.0  358.0      301.480000          0.09
2     12352.0        6.0    260.0  296.0      368.256667          0.19
3     12356.0        2.0    303.0  325.0      269.905000          0.07
4     12358.0        1.0    149.0  150.0      683.200000          0.08
```

In [26]:
```
print("The number of Returning Customers are:",len(shortlisted_customers))
```

```
The number of Returning Customers are: 2790
```

In [27]:
```
#Train gamma gamma model by taking into account the monetary_value
import lifetimes
ggf = lifetimes.GammaGammaFitter(penalizer_coef=0)
ggf.fit(shortlisted_customers['frequency'],shortlisted_customers['monetary_value'])
print(ggf)
```

```
<lifetimes.GammaGammaFitter: fitted with 2790 subjects, p: 2.10, q: 3.45, v: 485.57>
```

In [28]:
```
#After applying Gamma-Gamma model,now we can estimate average transaction value for each customer.
print(ggf.conditional_expected_average_profit(lf_data['frequency'],lf_data['monetary_value']).head(10))
```

```
CustomerID
12346.0    416.917667
12347.0    569.988807
12348.0    333.762672
12349.0    416.917667
12350.0    416.917667
12352.0    376.166864
12353.0    416.917667
12354.0    416.917667
12355.0    416.917667
12356.0    324.008941
dtype: float64
```

In [29]:
```
lf_data['pred_txn_value'] = round(ggf.conditional_expected_average_profit(lf_data['frequency'],lf_data['monetary_value']
lf_data.reset_index().head()
```

Out[29]:

	CustomerID	frequency	recency	T	monetary_value	pred_num_txn	pred_txn_value
0	12346.0	0.0	0.0	325.0	0.000000	0.02	416.92
1	12347.0	6.0	365.0	367.0	599.701667	0.16	569.99
2	12348.0	3.0	283.0	358.0	301.480000	0.09	333.76
3	12349.0	0.0	0.0	18.0	0.000000	0.10	416.92
4	12350.0	0.0	0.0	310.0	0.000000	0.02	416.92

In [30]:
```
lf_data['CLV'] = round(ggf.customer_lifetime_value(bgf,lf_data['frequency'],lf_data['recency'],lf_data['T'],lf_data['mon
lf_data.drop(lf_data.iloc[:,0:6],inplace=True,axis=1)
lf_data.sort_values(by='CLV',ascending=False).head(10).reset_index()
```

Out[30]:

	CustomerID	CLV
0	14646.0	222128.93
1	18102.0	178895.33
2	16446.0	175531.47
3	17450.0	147476.62
4	14096.0	127589.20
5	14911.0	109442.13
6	12415.0	96290.23
7	14156.0	89410.33
8	17511.0	67660.41
9	16029.0	58729.62