In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import jinja2
from pycaret.clustering import *
%matplotlib inline
```

In [2]:
```
df = pd.read_csv("Data_problem 1.csv")
```

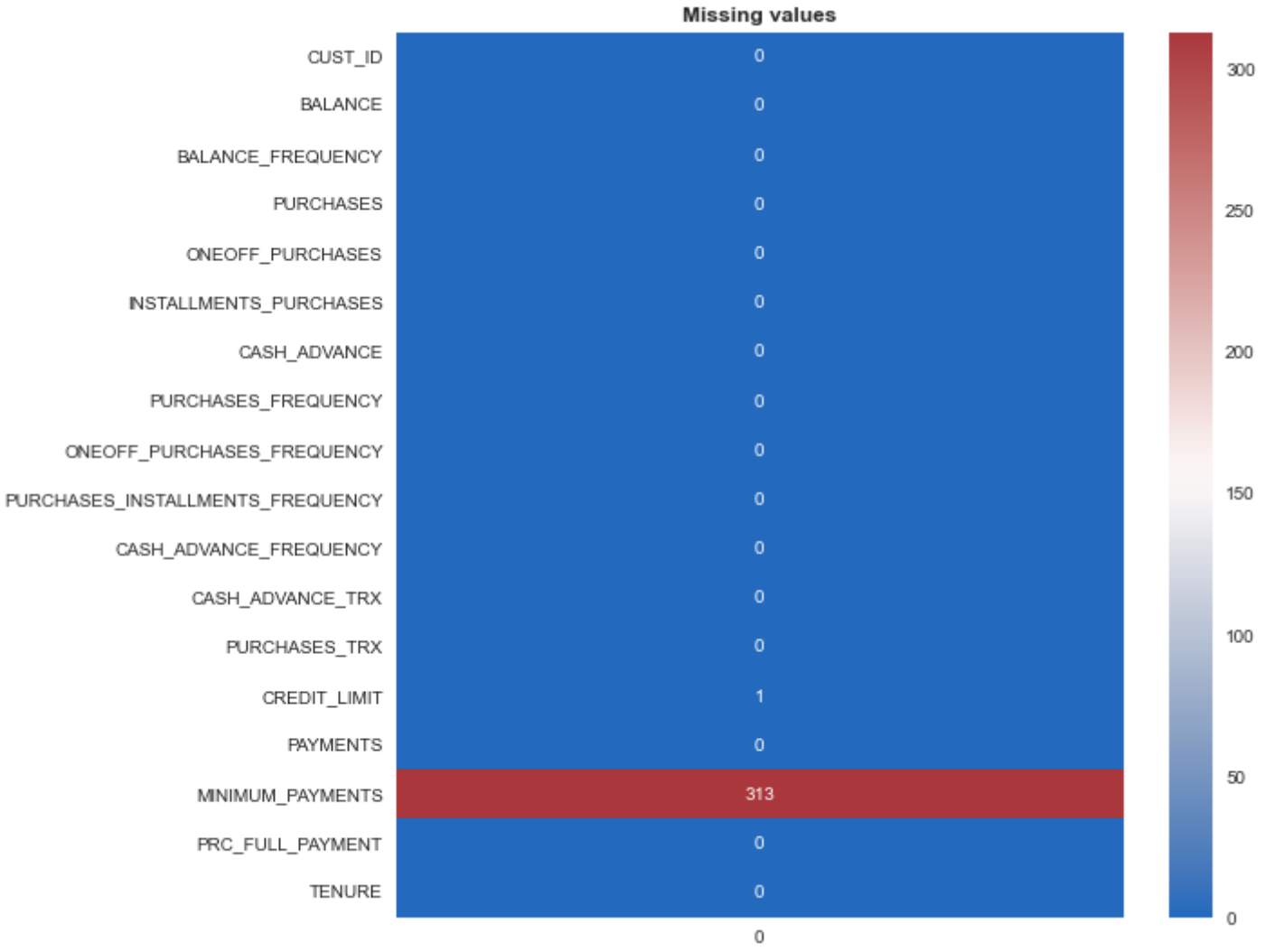
In [3]:
```
df.head(10)
```

Out[3]:

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENC
0	C10001	40.900749	0.818182	95.40	0.00	95.40	0.000000	0.1666
1	C10002	3202.467416	0.909091	0.00	0.00	0.00	6442.945483	0.0000
2	C10003	2495.148862	1.000000	773.17	773.17	0.00	0.000000	1.0000
3	C10004	1666.670542	0.636364	1499.00	1499.00	0.00	205.788017	0.0833
4	C10005	817.714335	1.000000	16.00	16.00	0.00	0.000000	0.0833
5	C10006	1809.828751	1.000000	1333.28	0.00	1333.28	0.000000	0.6666
6	C10007	627.260806	1.000000	7091.01	6402.63	688.38	0.000000	1.0000
7	C10008	1823.652743	1.000000	436.20	0.00	436.20	0.000000	1.0000
8	C10009	1014.926473	1.000000	861.49	661.49	200.00	0.000000	0.3333
9	C10010	152.225975	0.545455	1281.60	1281.60	0.00	0.000000	0.1666

In [4]:
```
plt.figure(figsize=(9,9))
plt.title("Missing values",fontweight="bold")
ax = sns.heatmap(df.isna().sum().to_frame(),annot=True,fmt="d",cmap="vlag")
ax.set_label("Amount missing")
plt.show
```

Out[4]: <function matplotlib.pyplot.show(close=None, block=None)>



In [5]:

```
categorical_col = df.CUST_ID
numerical_col = df.iloc[:,1:]
numerical_col
```

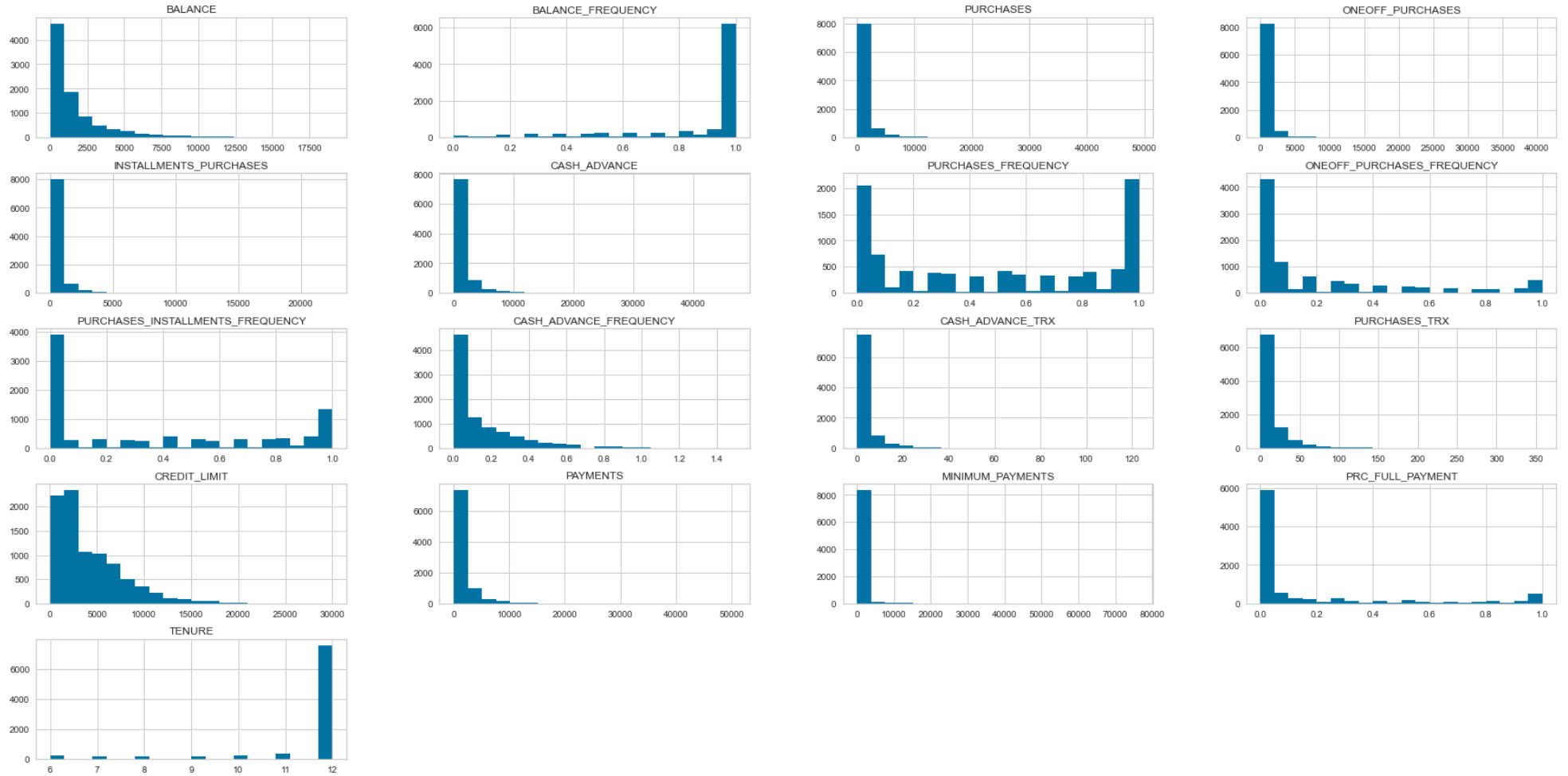
Out[5]:

	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	ON
0	40.900749	0.818182	95.40	0.00	95.40	0.000000	0.166667	
1	3202.467416	0.909091	0.00	0.00	0.00	6442.945483	0.000000	
2	2495.148862	1.000000	773.17	773.17	0.00	0.000000	1.000000	
3	1666.670542	0.636364	1499.00	1499.00	0.00	205.788017	0.083333	
4	817.714335	1.000000	16.00	16.00	0.00	0.000000	0.083333	
...	...	...	...	...	...	...	...	
8945	28.493517	1.000000	291.12	0.00	291.12	0.000000	1.000000	
8946	19.183215	1.000000	300.00	0.00	300.00	0.000000	1.000000	
8947	23.398673	0.833333	144.40	0.00	144.40	0.000000	0.833333	
8948	13.457564	0.833333	0.00	0.00	0.00	36.558778	0.000000	
8949	372.708075	0.666667	1093.25	1093.25	0.00	127.040008	0.666667	

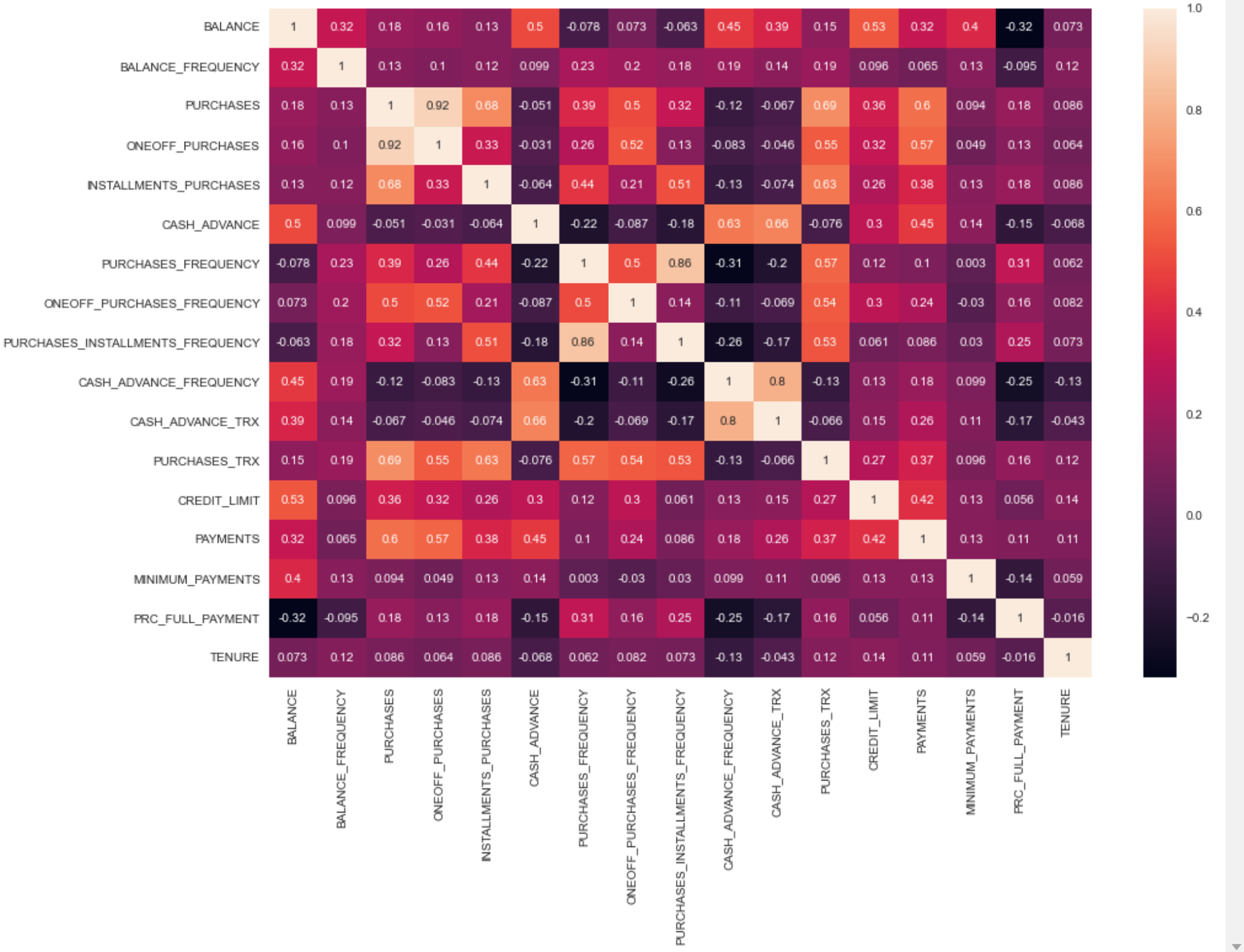
8950 rows × 17 columns

In [6]:
```
numerical_col.hist(bins = 20, figsize=(30,15))
```

Out[6]:
```
array([[<AxesSubplot:title={'center':'BALANCE'}>,
        <AxesSubplot:title={'center':'BALANCE_FREQUENCY'}>,
        <AxesSubplot:title={'center':'PURCHASES'}>,
        <AxesSubplot:title={'center':'ONEOFF_PURCHASES'}>],
       [<AxesSubplot:title={'center':'INSTALLMENTS_PURCHASES'}>,
        <AxesSubplot:title={'center':'CASH_ADVANCE'}>,
        <AxesSubplot:title={'center':'PURCHASES_FREQUENCY'}>,
        <AxesSubplot:title={'center':'ONEOFF_PURCHASES_FREQUENCY'}>],
       [<AxesSubplot:title={'center':'PURCHASES_INSTALLMENTS_FREQUENCY'}>,
        <AxesSubplot:title={'center':'CASH_ADVANCE_FREQUENCY'}>,
        <AxesSubplot:title={'center':'CASH_ADVANCE_TRX'}>,
        <AxesSubplot:title={'center':'PURCHASES_TRX'}>],
       [<AxesSubplot:title={'center':'CREDIT_LIMIT'}>,
        <AxesSubplot:title={'center':'PAYMENTS'}>,
        <AxesSubplot:title={'center':'MINIMUM_PAYMENTS'}>,
        <AxesSubplot:title={'center':'PRC_FULL_PAYMENT'}>],
       [<AxesSubplot:title={'center':'TENURE'}>, <AxesSubplot:>,
        <AxesSubplot:>, <AxesSubplot:>]], dtype=object)
```

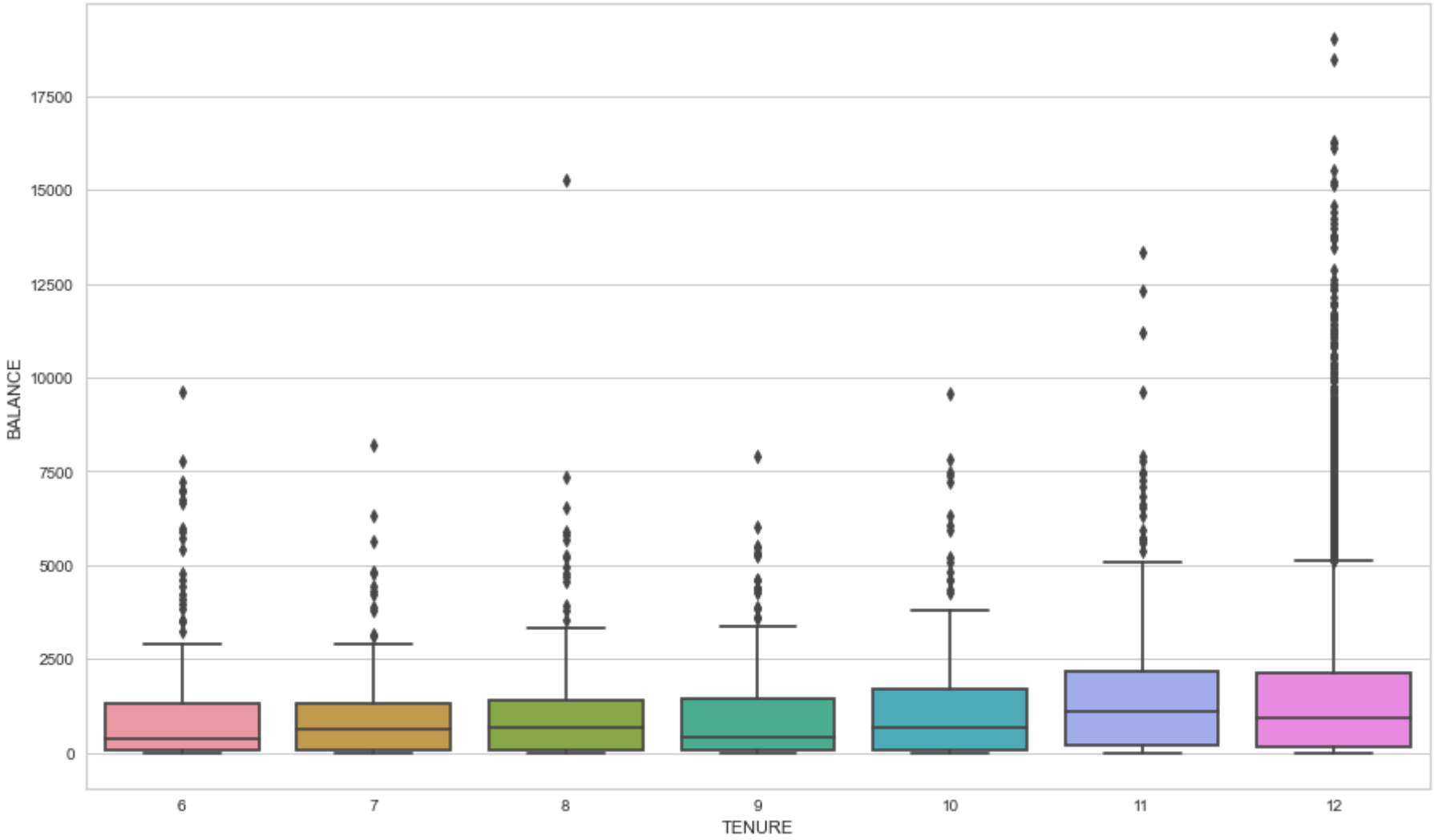

BALANCE
BALANCE_FREQUENCY
PURCHASES
ONEOFF_PURCHASES
INSTALLMENTS_PURCHASES
CASH_ADVANCE
PURCHASES_FREQUENCY
ONEOFF_PURCHASES_FREQUENCY
PURCHASES_INSTALLMENTS_FREQUENCY
CASH_ADVANCE_FREQUENCY
CASH_ADVANCE_TRX
PURCHASES_TRX
CREDIT_LIMIT
PAYMENTS
MINIMUM_PAYMENTS
PRC_FULL_PAYMENT
TENURE

In [7]:
```
plt.figure(figsize=(15,10))
sns.heatmap(data= df.corr(), annot= True)
plt.show()
```

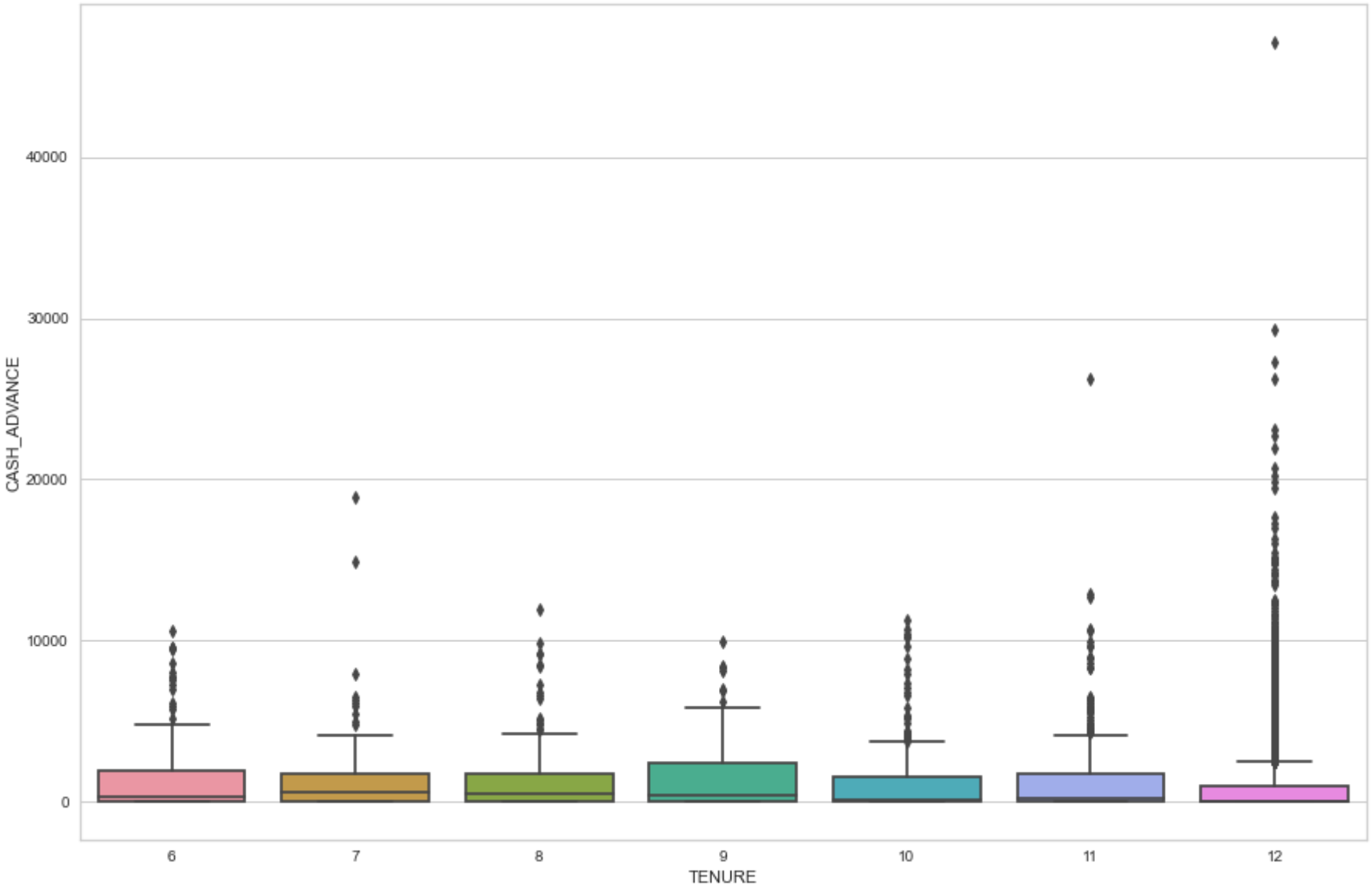

BALANCE 1 0.32 0.18 0.16 0.13 0.5 -0.078 0.073 -0.063 0.45 0.39 0.15 0.53 0.32 0.4 -0.32 0.073
BALANCE_FREQUENCY 0.32 1 0.13 0.1 0.12 0.099 0.23 0.2 0.18 0.19 0.14 0.19 0.096 0.065 0.13 -0.095 0.12
PURCHASES 0.18 0.13 1 0.92 0.68 -0.051 0.39 0.5 0.32 -0.12 -0.067 0.69 0.36 0.6 0.094 0.18 0.086
ONEOFF_PURCHASES 0.16 0.1 0.92 1 0.33 -0.031 0.26 0.52 0.13 -0.083 -0.046 0.55 0.32 0.57 0.049 0.13 0.064
INSTALLMENTS_PURCHASES 0.13 0.12 0.68 0.33 1 -0.064 0.44 0.21 0.51 -0.13 -0.074 0.63 0.26 0.38 0.13 0.18 0.086
CASH_ADVANCE 0.5 0.099 -0.051 -0.031 -0.064 1 -0.22 -0.087 -0.18 0.63 0.66 -0.076 0.3 0.45 0.14 -0.15 -0.068
PURCHASES_FREQUENCY -0.078 0.23 0.39 0.26 0.44 -0.22 1 0.5 0.86 -0.31 -0.2 0.57 0.12 0.1 0.003 0.31 0.062
ONEOFF_PURCHASES_FREQUENCY 0.073 0.2 0.5 0.52 0.21 -0.087 0.5 1 0.14 -0.11 -0.069 0.54 0.3 0.24 -0.03 0.16 0.082
PURCHASES_INSTALLMENTS_FREQUENCY -0.063 0.18 0.32 0.13 0.51 -0.18 0.86 0.14 1 -0.26 -0.17 0.53 0.061 0.086 0.03 0.25 0.073
CASH_ADVANCE_FREQUENCY 0.45 0.19 -0.12 -0.083 -0.13 0.63 -0.31 -0.11 -0.26 1 0.8 -0.13 0.13 0.18 0.099 -0.25 -0.13
CASH_ADVANCE_TRX 0.39 0.14 -0.067 -0.046 -0.074 0.66 -0.2 -0.069 -0.17 0.8 1 -0.066 0.15 0.26 0.11 -0.17 -0.043
PURCHASES_TRX 0.15 0.19 0.69 0.55 0.63 -0.076 0.57 0.54 0.53 -0.13 -0.066 1 0.27 0.37 0.096 0.16 0.12
CREDIT_LIMIT 0.53 0.096 0.36 0.32 0.26 0.3 0.12 0.3 0.061 0.13 0.15 0.27 1 0.42 0.13 0.056 0.14
PAYMENTS 0.32 0.065 0.6 0.57 0.38 0.45 0.1 0.24 0.086 0.18 0.26 0.37 0.42 1 0.13 0.11 0.11
MINIMUM_PAYMENTS 0.4 0.13 0.094 0.049 0.13 0.14 0.003 -0.03 0.03 0.099 0.11 0.096 0.13 0.13 1 -0.14 0.059
PRC_FULL_PAYMENT -0.32 -0.095 0.18 0.13 0.18 -0.15 0.31 0.16 0.25 -0.25 -0.17 0.16 0.056 0.11 -0.14 1 -0.016
TENURE 0.073 0.12 0.086 0.064 0.086 -0.068 0.062 0.082 0.073 -0.13 -0.043 0.12 0.14 0.11 0.059 -0.016 1
BALANCE
BALANCE_FREQUENCY
PURCHASES
ONEOFF_PURCHASES
INSTALLMENTS_PURCHASES
CASH_ADVANCE
PURCHASES_FREQUENCY
ONEOFF_PURCHASES_FREQUENCY
PURCHASES_INSTALLMENTS_FREQUENCY
CASH_ADVANCE_FREQUENCY
CASH_ADVANCE_TRX
PURCHASES_TRX
CREDIT_LIMIT
PAYMENTS
MINIMUM_PAYMENTS
PRC_FULL_PAYMENT
TENURE
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

In [8]:

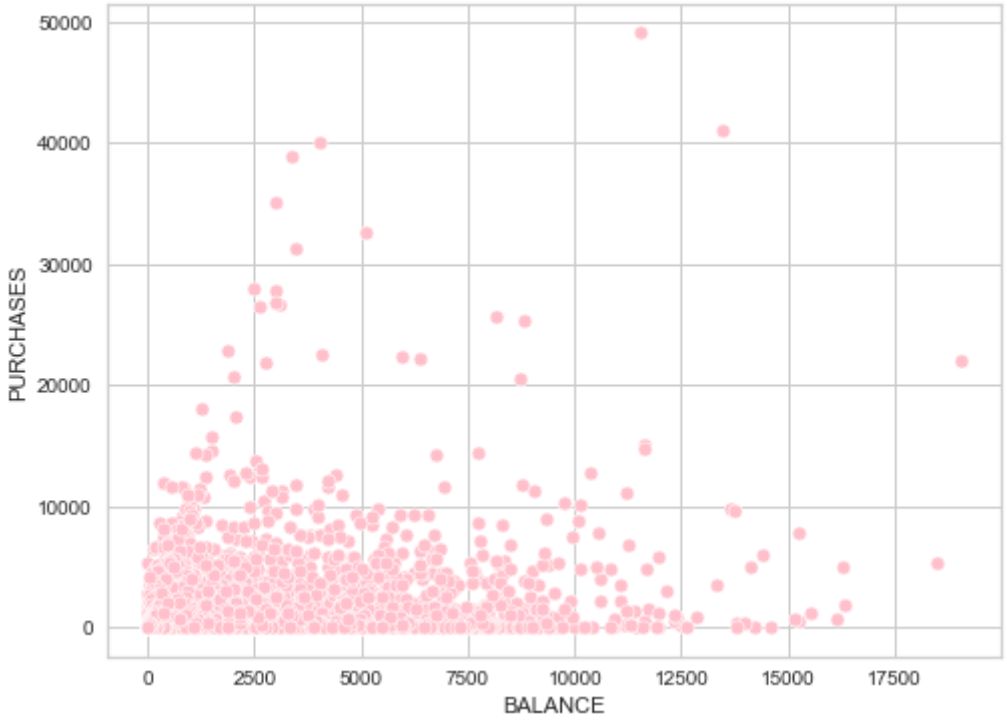
```
plt.figure(figsize=(15,9))
sns.boxplot(x = "TENURE", y ="BALANCE", data = df)
plt.show()
```



In [9]:
```
plt.figure(figsize=(15,10))
sns.boxplot(x = "TENURE", y ="CASH_ADVANCE", data = df)
plt.show()
```



In [10]:
```
plt.figure(figsize=(8,6))
sns.scatterplot(x = "BALANCE", y ="PURCHASES", data = df, color = "pink")
plt.show()
```



In [11]:
```
df['limit_usage'] = df["PURCHASES"]/df["CREDIT_LIMIT"]
df["PUR-BAL"] = df["PURCHASES"] /df["BALANCE"]
df["BAL-INS"] = df["INSTALLMENTS_PURCHASES"]/df["BALANCE"]
df
```

Out[11]:

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQU
0	C10001	40.900749	0.818182	95.40	0.00	95.40	0.000000	0.1
1	C10002	3202.467416	0.909091	0.00	0.00	0.00	6442.945483	0.0
2	C10003	2495.148862	1.000000	773.17	773.17	0.00	0.000000	1.0
3	C10004	1666.670542	0.636364	1499.00	1499.00	0.00	205.788017	0.0
4	C10005	817.714335	1.000000	16.00	16.00	0.00	0.000000	0.0
...	...	...	...	...	...	...	...	
8945	C19186	28.493517	1.000000	291.12	0.00	291.12	0.000000	1.0
8946	C19187	19.183215	1.000000	300.00	0.00	300.00	0.000000	1.0
8947	C19188	23.398673	0.833333	144.40	0.00	144.40	0.000000	0.8
8948	C19189	13.457564	0.833333	0.00	0.00	0.00	36.558778	0.0
8949	C19190	372.708075	0.666667	1093.25	1093.25	0.00	127.040008	0.6

8950 rows × 21 columns

In [ ]:
```
new_df = df
new_df.fillna(0, inplace = True)
new_df = new_df.drop(["TENURE","CUST_ID","CASH_ADVANCE_TRX","PURCHASES_INSTALLMENTS_FREQUENCY","BALANCE_FREQUENCY","ONEOFF_PURCHAS
```

In [16]:
```
clusters = setup(new_df,normalize = True,
            pca = True,silent =True,
            combine_rare_levels= True,
            remove_multicollinearity= True,session_id=12)
```

	Description	Value
0	session_id	12
1	Original Data	(8950, 13)
2	Missing Values	False
3	Numeric Features	13
4	Categorical Features	0
5	Ordinal Features	False
6	High Cardinality Features	False
7	High Cardinality Method	None
8	Transformed Data	(8950, 12)
9	CPU Jobs	-1
10	Use GPU	False
11	Log Experiment	False
12	Experiment Name	cluster-default-name
13	USI	5246
14	Imputation Type	simple
15	Iterative Imputation Iteration	None
16	Numeric Imputer	mean
17	Iterative Imputation Numeric Model	None
18	Categorical Imputer	mode
19	Iterative Imputation Categorical Model	None
20	Unknown Categoricals Handling	least_frequent
21	Normalize	True
22	Normalize Method	zscore
23	Transformation	False
24	Transformation Method	None

	Description	Value
25	PCA	True
26	PCA Method	linear
27	PCA Components	0.990000
28	Ignore Low Variance	False
29	Combine Rare Levels	True
30	Rare Level Threshold	0.100000
31	Numeric Binning	False
32	Remove Outliers	False
33	Outliers Threshold	None
34	Remove Multicollinearity	True
35	Multicollinearity Threshold	0.900000
36	Remove Perfect Collinearity	False
37	Clustering	False
38	Clustering Iteration	None
39	Polynomial Features	False
40	Polynomial Degree	None
41	Trignometry Features	False
42	Polynomial Threshold	None
43	Group Features	False
44	Feature Selection	False
45	Feature Selection Method	classic
46	Features Selection Threshold	None
47	Feature Interaction	False
48	Feature Ratio	False
49	Interaction Threshold	None

In [17]:
```
models()
```

Out[17]:

ID	Name	Reference
kmeans	K-Means Clustering	sklearn.cluster._kmeans.KMeans
ap	Affinity Propagation	sklearn.cluster._affinity_propagation.Affinity...
meanshift	Mean Shift Clustering	sklearn.cluster._mean_shift.MeanShift
sc	Spectral Clustering	sklearn.cluster._spectral.SpectralClustering
hclust	Agglomerative Clustering	sklearn.cluster._agglomerative.AgglomerativeCl...
dbscan	Density-Based Spatial Clustering	sklearn.cluster._dbscan.DBSCAN
optics	OPTICS Clustering	sklearn.cluster._optics.OPTICS
birch	Birch Clustering	sklearn.cluster._birch.Birch
kmodes	K-Modes Clustering	kmodes.kmodes.KModes

In [18]:
```
KMEANS = create_model("kmeans",num_clusters=4)
```

	Silhouette	Calinski-Harabasz	Davies-Bouldin	Homogeneity	Rand Index	Completeness
0	0.2532	1651.2791	1.3649	0	0	0

In [19]:
```
hierr_clust = create_model("hclust",num_clusters=6)
```

	Silhouette	Calinski-Harabasz	Davies-Bouldin	Homogeneity	Rand Index	Completeness
0	0.227	1431.7102	1.2324	0	0	0

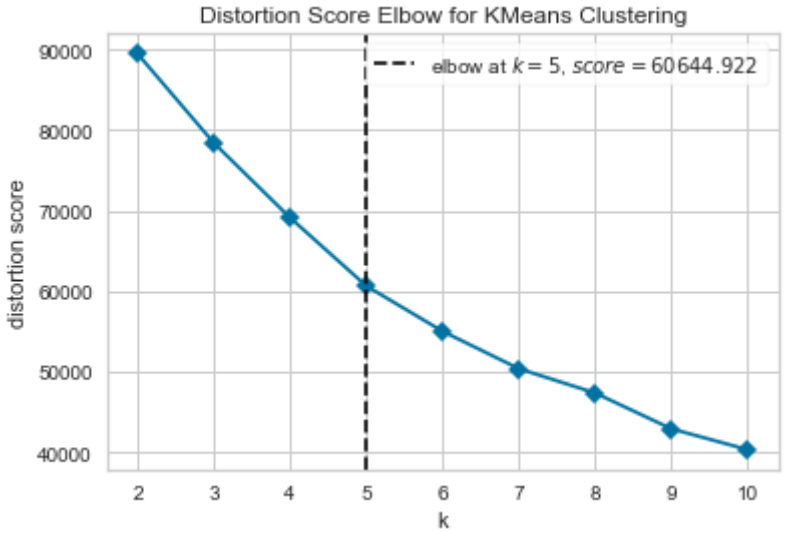
In [20]:
```
KMEANS_result = assign_model(KMEANS )
KMEANS_result
```

Out[20]:

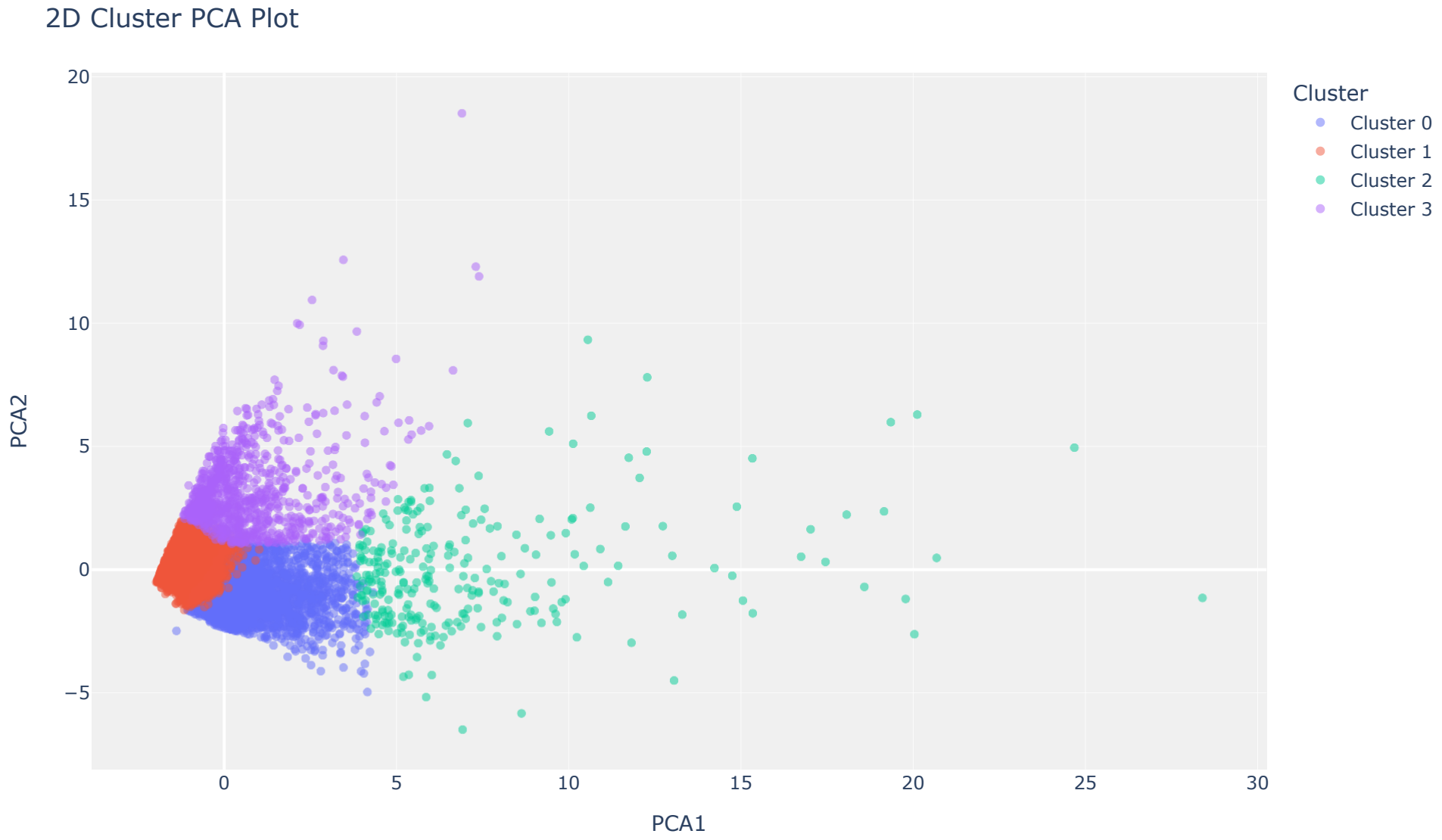
	BALANCE	PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	PURCHASES_TRX	CREDIT_LIMIT	PAYMENTS	MIN
0	40.900749	95.40	95.40	0.000000	0.166667	2	1000.0	201.802084	
1	3202.467416	0.00	0.00	6442.945483	0.000000	0	7000.0	4103.032597	
2	2495.148862	773.17	0.00	0.000000	1.000000	12	7500.0	622.066742	
3	1666.670542	1499.00	0.00	205.788017	0.083333	1	7500.0	0.000000	
4	817.714335	16.00	0.00	0.000000	0.083333	1	1200.0	678.334763	
...	...	...	...	...	...	...	...	...	
8945	28.493517	291.12	291.12	0.000000	1.000000	6	1000.0	325.594462	
8946	19.183215	300.00	300.00	0.000000	1.000000	6	1000.0	275.861322	
8947	23.398673	144.40	144.40	0.000000	0.833333	5	1000.0	81.270775	
8948	13.457564	0.00	0.00	36.558778	0.000000	0	500.0	52.549959	
8949	372.708075	1093.25	0.00	127.040008	0.666667	23	1200.0	63.165404	

8950 rows × 14 columns

In [21]:
```
plot_model(KMEANS,plot='elbow' )
```

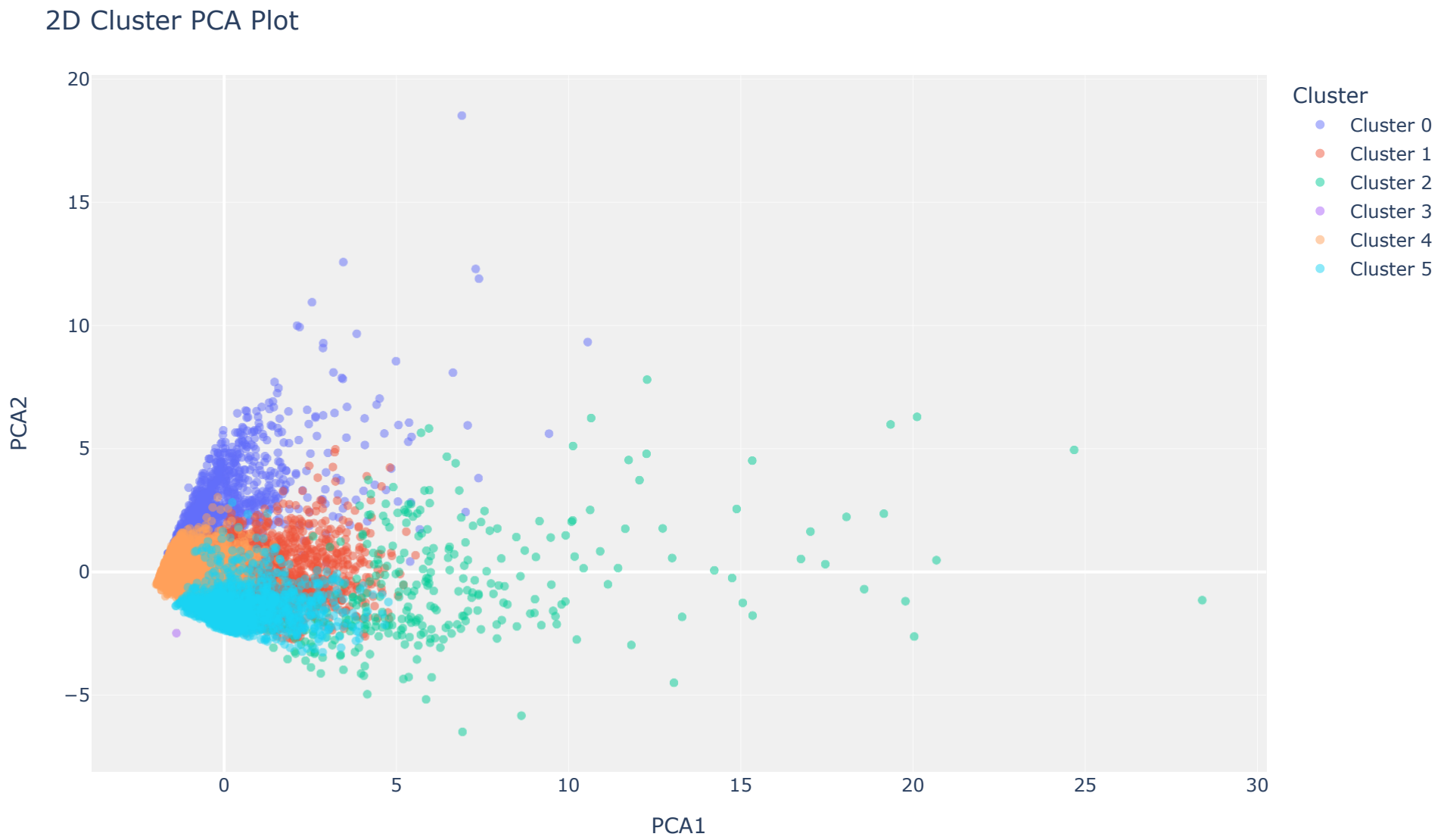


In [22]:
```
plot_model(KMEANS)
```



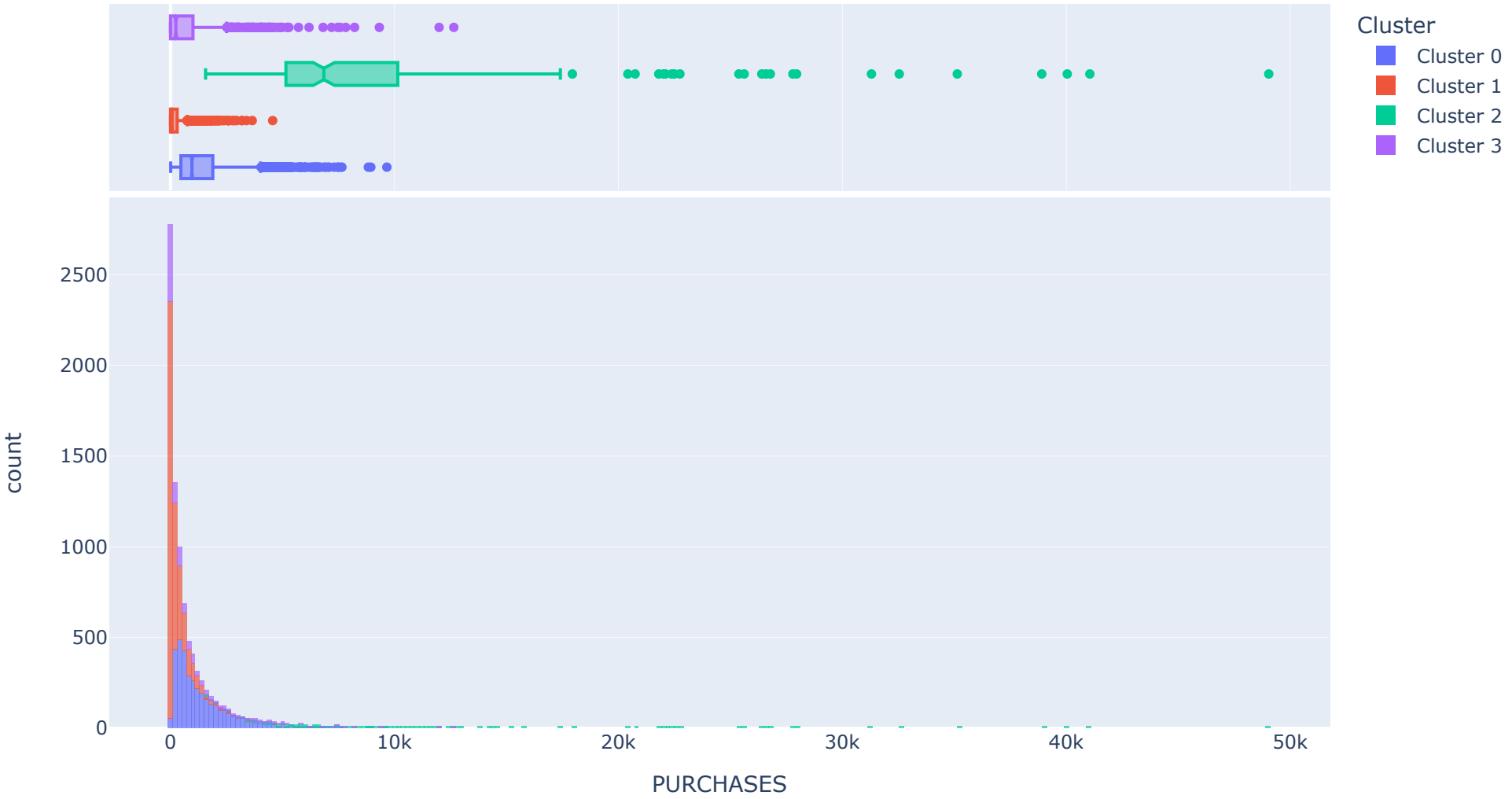
KMeans RESULT -The customers are divided into 4 clusters based on the above final 14 features. Some points of cluster1 are overlapping with the space of cluster0. Besides that we can seperate the clusters from one another fairly

In [23]:
```
plot_model(hierr_clust)
```



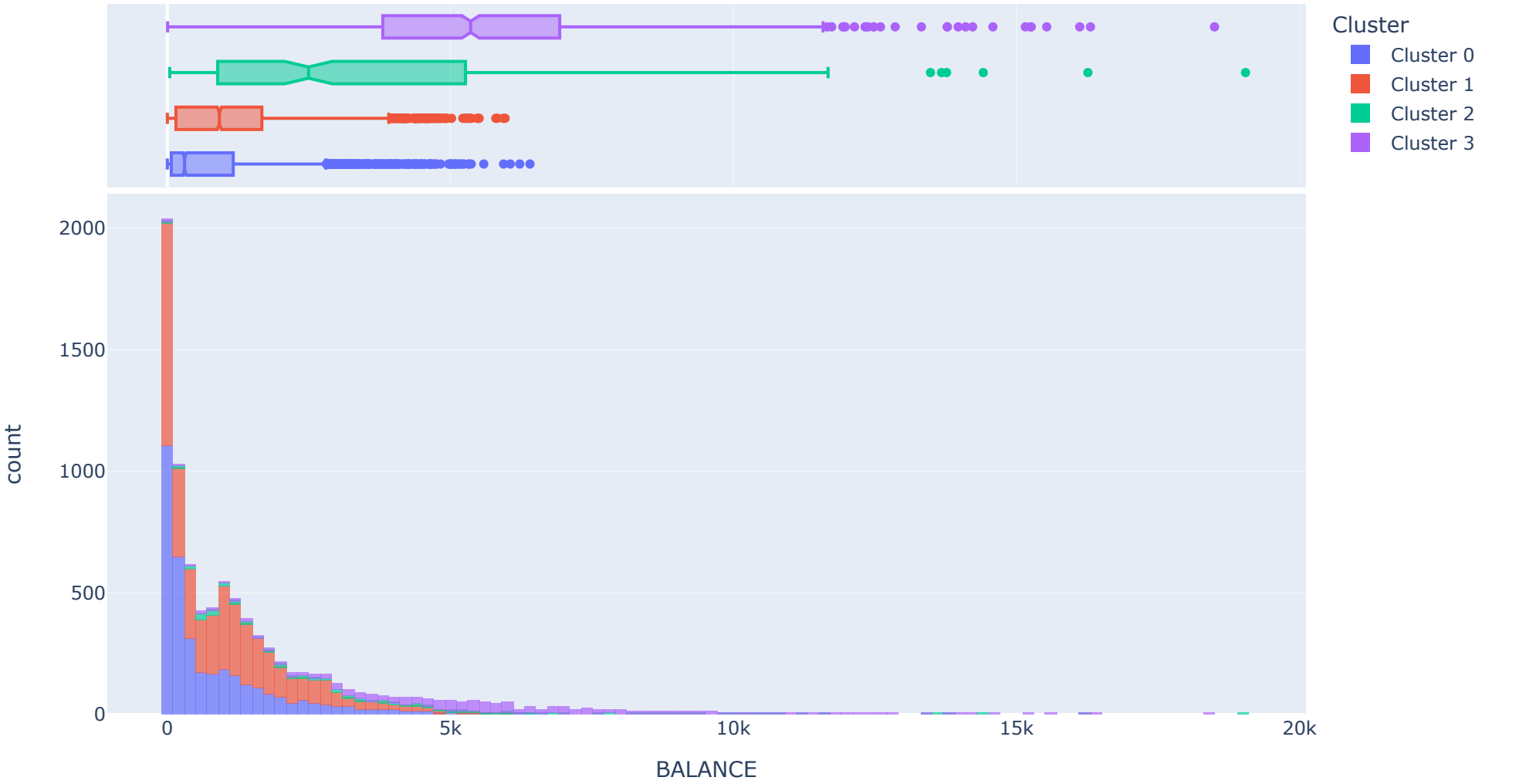
Hierarchial Result:- The results are much worse using this technique as the datapoints in all the clusters are overlapping each other. There is no rough decision boundary> Due to this I have used the KMeans as our final model

In [24]:
```
plot_model(KMEANS, plot = "distribution", feature = "PURCHASES")
```



The majority of customers from cluster 0 and cluster 1 made purchases around 0 and 7000. However the customers in Cluster 2 and cluster3 have much higher amount purchases. This can be seen clearly in the box plots's whiskers suggesting the outliers of the data

In [25]:
```
plot_model(KMEANS, plot = "distribution", feature = "BALANCE")
```



The balance of customers of cluster1 and cluster0 for most cases was in range0 to 2k. Then it went downhill as the highest around 6500k. Cluster3 and cluster2 had customers having much higher balance for most cases

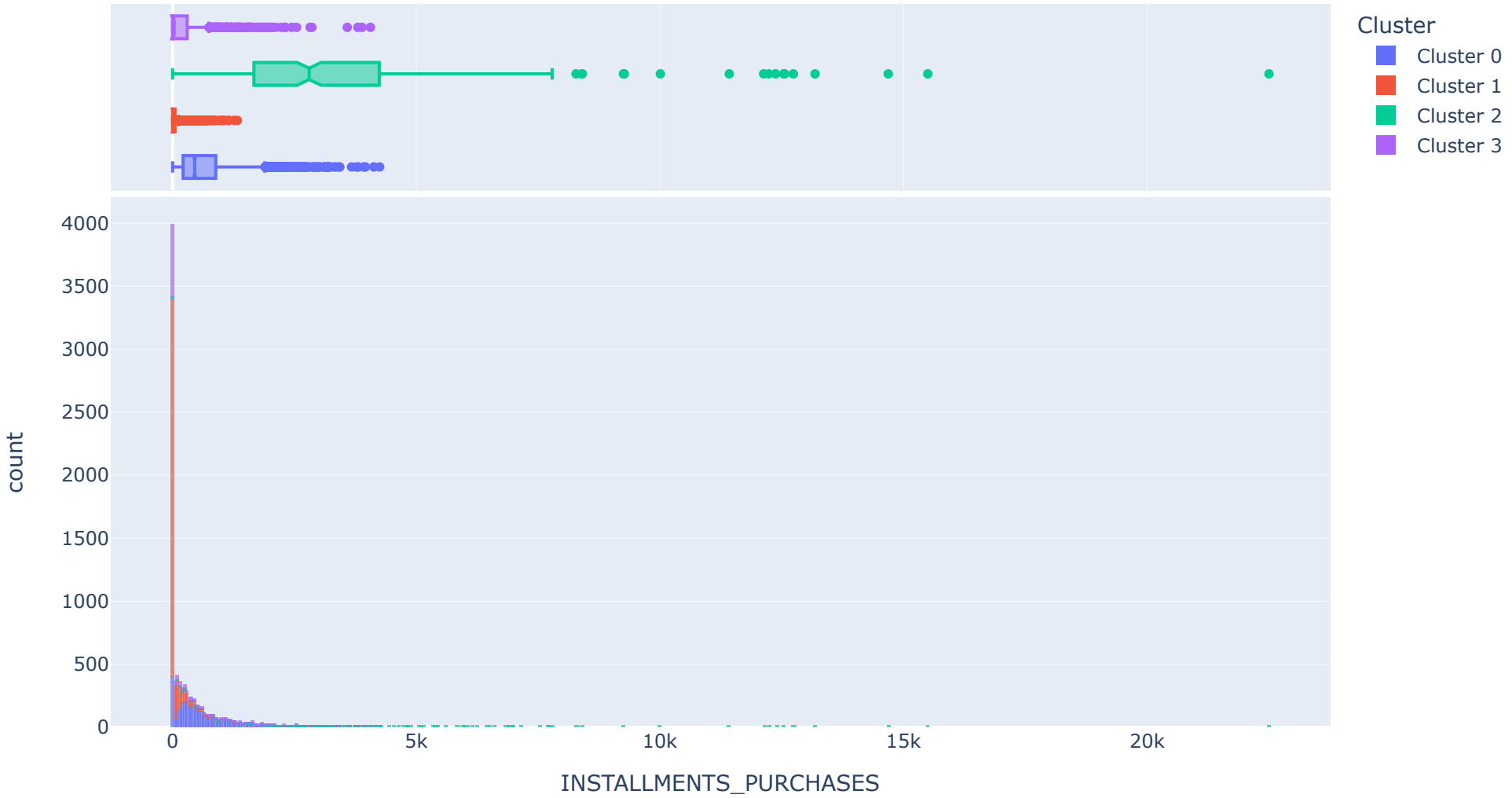
In [26]:
```
plot_model(KMEANS, plot = "distribution", feature = "CREDIT_LIMIT")
```

Once again the same trend is followed cluster 1 and cluster 0 had smaller credit limits than cluster2 and cluster3 barring some exceptions. Majority of people from clusters 0 and 1 had a credit limit upto 10k

In [27]:
```
plot_model(KMEANS, plot = "distribution", feature = "PRC_FULL_PAYMENT")
```

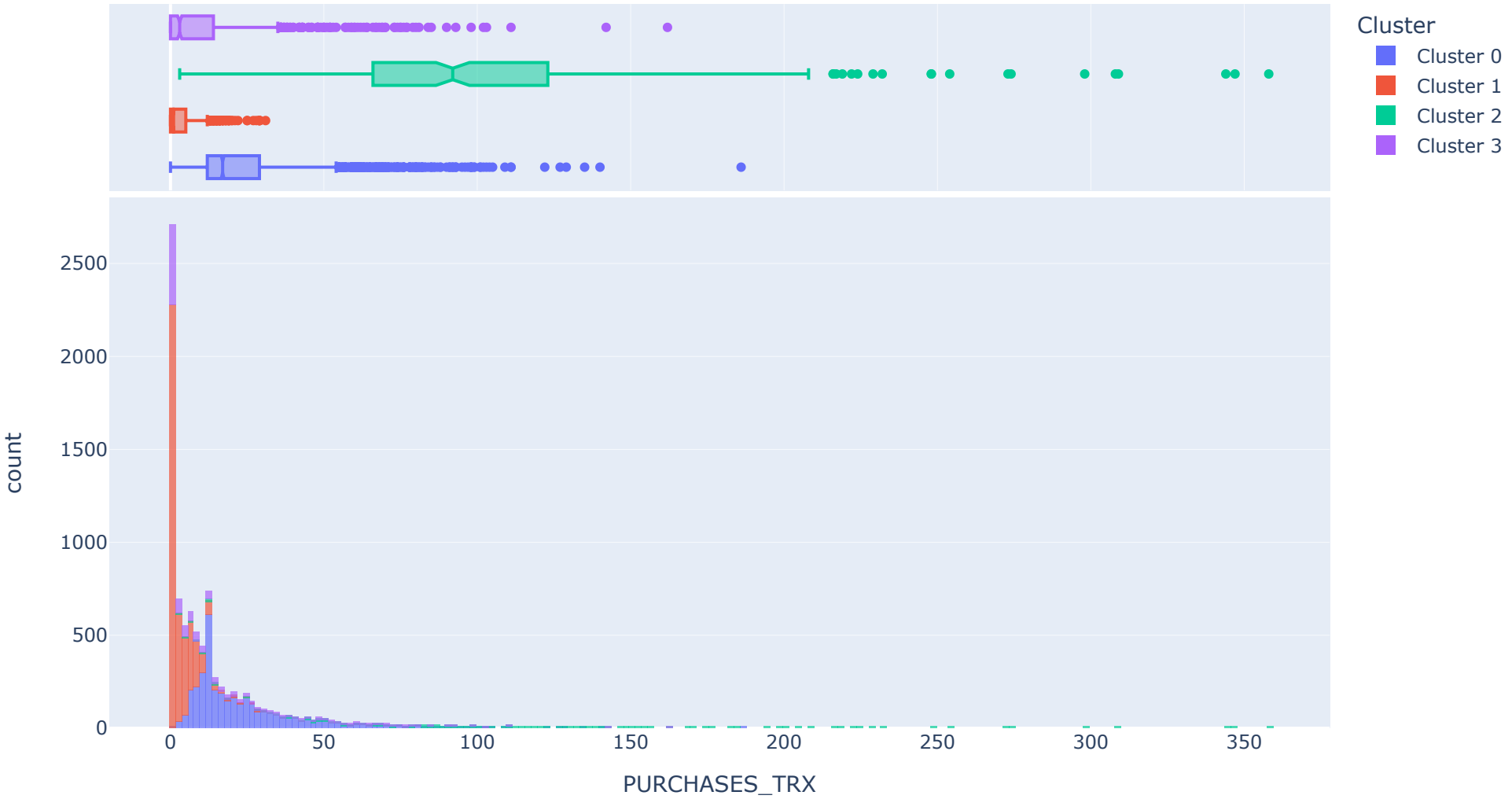
Here the trend is different from the previous variables. Cluster0 and cluster 2 have the customers whose payment were paid in full of the statement. Eventhough customers from cluster1 had lower credit, purchases from cluster 2 the payment of due statement is more for cluster 2.

In [28]:
```
plot_model(KMEANS, plot = "distribution", feature = "INSTALLMENTS_PURCHASES")
```



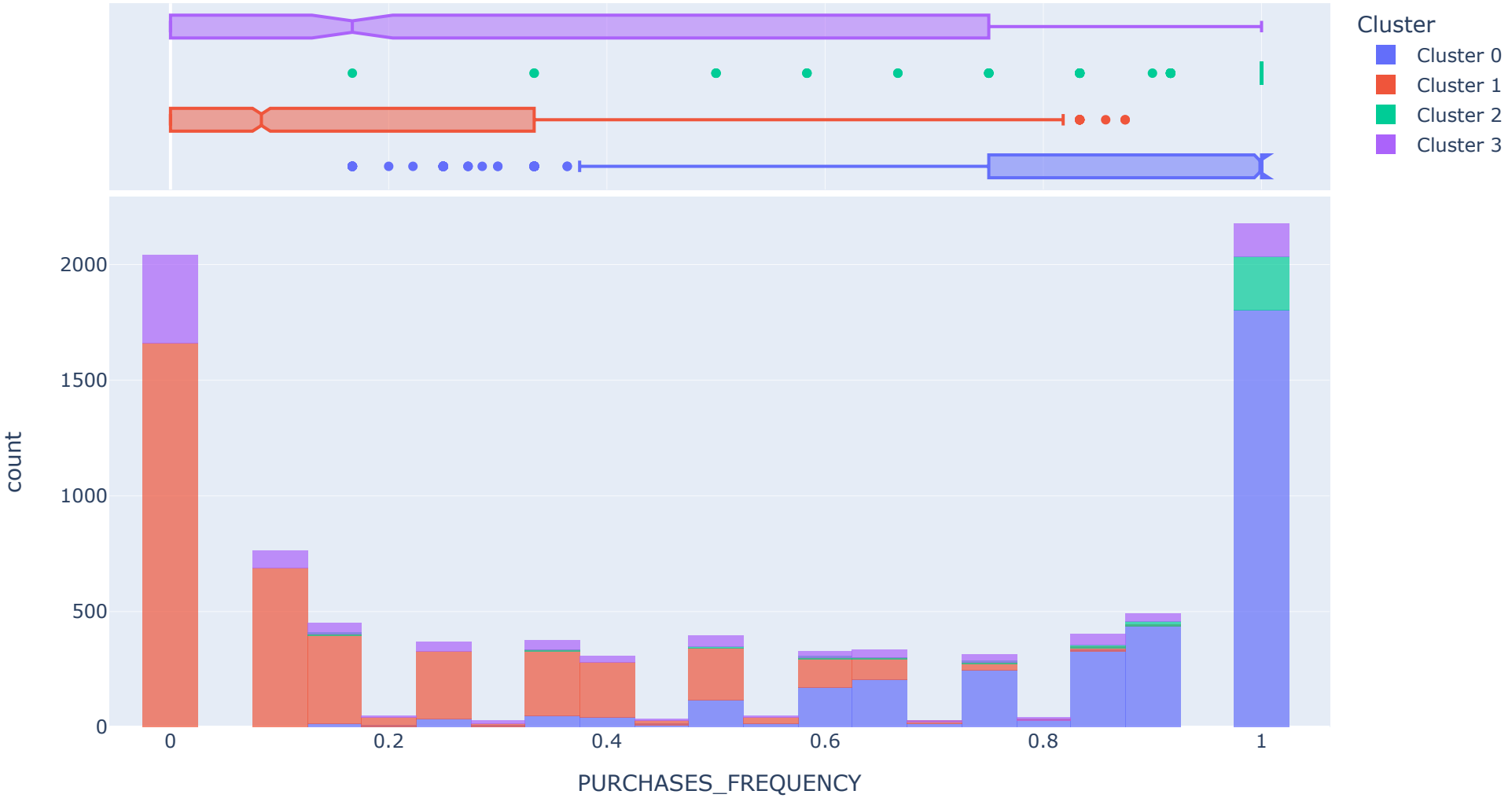
Customers from cluster 1 had very few purchases in installments. The most highest installment purchases were of customers from cluster2 eventhough their credit limit was somewhat lower from the customers of cluster3.

In [29]:
```
plot_model(KMEANS, plot = "distribution", feature = "PURCHASES_TRX")
```



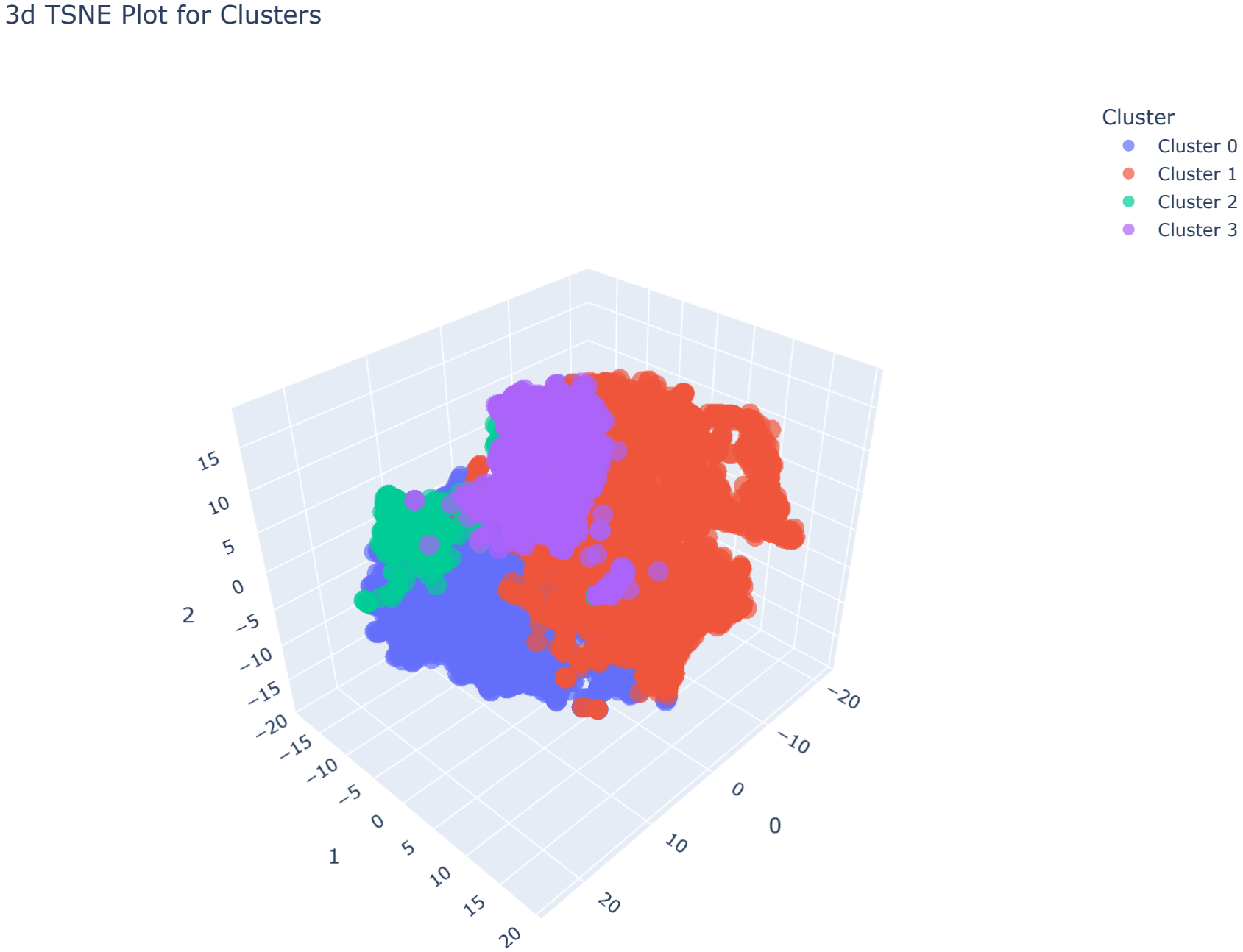
The Average amount paid from the customers in cluster1 was around 0 to 30. Customers from cluster0 had a much larger range than cluster0. However cluster0 had more outliers. The customers from cluster2 had the highest range and amount average payment per transaction

In [30]:
```
plot_model(KMEANS, plot = "distribution", feature = "PURCHASES_FREQUENCY")
```



Interestingly customers from cluster0 had the numbers in purchase frequency as their credit limit and balance were the lowest in general. Customers from cluster1 and cluster3 had varying purchase frequency throughout

In [31]:
```
plot_model(KMEANS, plot ='tsne')
```



Overall analysis:- Cluster1 and Cluster0 customers had low credit limit, low balance, low purchases Cluster2 and Cluster3 had customers with high credit limit but low purchase frequency on average compared to other clusters