Distribution Analysis after Clustering

December 16, 2022

0.0.1 Distribution Analysis at a Cluster X MSA level

Sarthak Arora

[2]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import os
import math as mt
from unidip import UniDip
import unidip.dip as dip
from scipy import stats
import statsmodels.api as sm
import pylab
from scipy.stats import norm
from Common_Functions import add_unique_identifier, data_split
from statsmodels.stats.multitest import fdrcorrection


import warnings
warnings.filterwarnings('ignore')
```

[3]:
```
os.getcwd()
```

[3]:
```
'/home/lennon_mccartney/Downloads/Prediction-of-commercial-insurance-payments-
for-surgical-procedure-using-DataRobot-main'
```

0.0.2 getting cluster groups we get using only the surgical cost

[11]:
```
df_clusters = pd.read_csv("Kmeans_clusters/clusters_only_using_NormCost.csv")
cluster_0 = df_clusters[df_clusters["cluster"] == 0].group.values
cluster_1 = df_clusters[df_clusters["cluster"] == 1].group.values
cluster_2 = df_clusters[df_clusters["cluster"] == 2].group.values
```

[12]:
```
# cluster_0 = ["ant_tls_fusion",
# "colorect",
# "femoral shaft fixation",
# "hepat",
# "hip_fracture_fixation",
# "intracranial_thromb",
# "post_cerv_fusion",
# "post_tls_fusion",
# "prox_tibia_fixation",
# "revision_tha",
# "revision_tka",
# "tha",
# "thoracic",
# "tka",
# "tpa"]

# cluster_1 = ["breast reconstruction",
# "bsp",
# "bunionectomy",
# "cardiac ablation_additional_discrete",
# "cardiac ablation_linear_focal",
# "cardiac_ablaton_anesthesia",
# "cardiac_ablaton_ice",
# "clavicle fixation",
# "fess",
# "hysterect",
# "kidney ablation",
# "lap appendectomy",
# "mastectomy",
# "navigation",
# "orthovisc_monovisc",
# "pnn",
# "robotic_assisted_surgery",
# "rtc_slap_bank",
# "septoplasty",
# "tavr"]

# cluster_2 = [
#     "ankle_fix",
# "ant_cerv_fusion",
# "bariatric",
# "cardiac ablation",
# "hernia",
# "laac",
# "liver ablation",
# "lung ablation",
# "lung ablation",
# "partial shoulder arthroplasty",
# "pka",
# "prostatectomy",
# "proximal humerus",
# "radius/ulna internal fixation",
# "tsa"
# ]
```

0.0.3 reading the main data and applying preprocessing functions

```
[10]: data = pd.read_csv('Data_Files/JnJ_Files/priv_mcare_f_pay_2022Oct18.csv')
      data = add_unique_identifier(data)
      model_data, future_data = data_split(data, count_col_name = 'priv_count')
      data = model_data
```

```
[13]: ## number of unique procedures in the training data
      data["group"].nunique()
```

```
[13]: 42
```

```
[14]: ## total number of missing mcare_median values in the training data
      total_mcare_missing = data["mcare_pay_median"].isna().sum()
      total_mcare_missing
```

```
[14]: 470
```

0.0.4 adding a new column to the training data for cluster labels

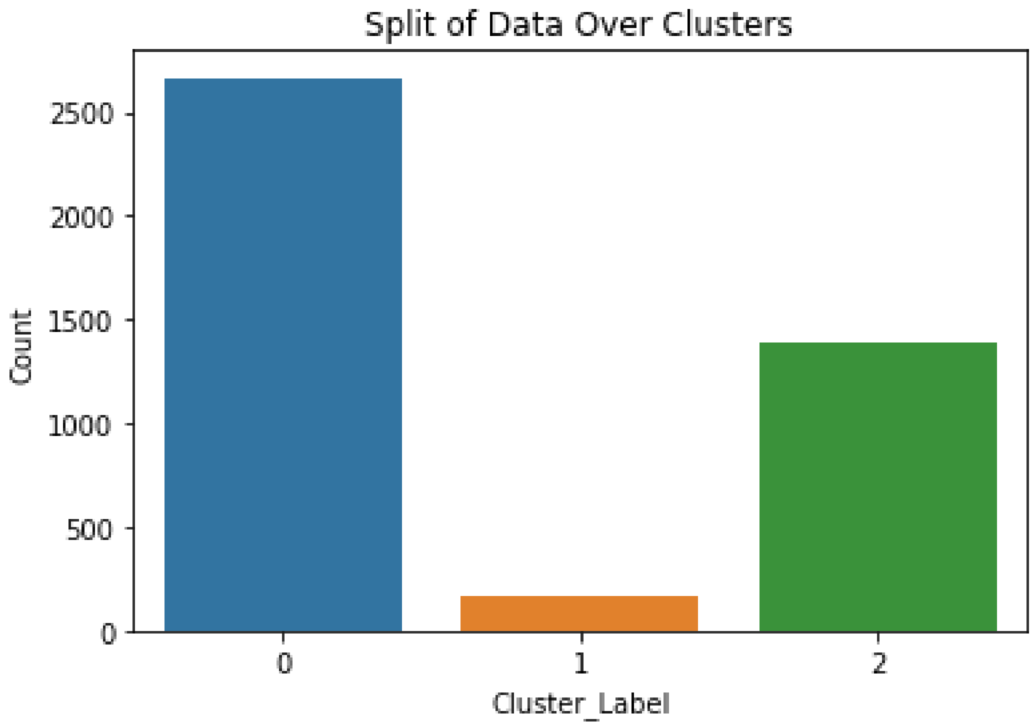
```
[15]: cluster_labels = []
      for i in range(len(data)):
          if data["group"].iloc[i] in cluster_0:
              cluster_labels.append(0)
          elif  data["group"].iloc[i] in cluster_1:
              cluster_labels.append(1)
          elif  data["group"].iloc[i] in cluster_2:
              cluster_labels.append(2)
          else:
              cluster_labels.append("NaN")
```

```
[16]: data["clusters"] = cluster_labels
```

```
[17]: cluster_means = data.groupby('clusters', as_index=False)['mcare_pay_median'].
       ↪mean().rename(columns={'mcare_pay_median':'mcare_cluster_mean'})
```

```
[18]: fig = sns.barplot(data=data.groupby(by = ["clusters"]).size().
       ↪reset_index(),x="clusters",y = 0)
      fig.set(xlabel ="Cluster_Label", ylabel = "Count", title ='Split of Data Over␣
       ↪Clusters')
```

[18]:
```
[Text(0.5, 0, 'Cluster_Label'),
 Text(0, 0.5, 'Count'),
 Text(0.5, 1.0, 'Split of Data Over Clusters')]
```



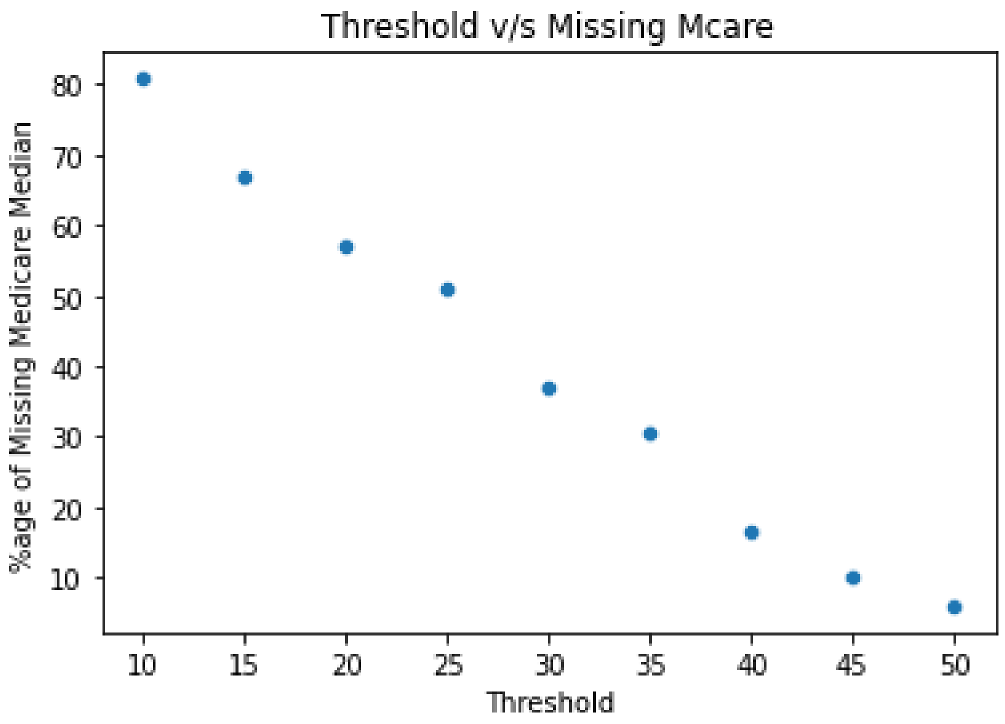
0.0.5 creating a dataframe having number of training examples at an MSA X Cluster Label level.

[19]:
```
da_points = data.groupby(by = ['clusters','msa']).size().reset_index()
```

0.0.6 Looking at how the number of missing Mcare median values change as we change the threshold (i.e. the minimum number of points we require in a MSA X Cluster distribution)

[20]:
```
thresholds = [10,15,20,25,30,35,40,45,50]
missing_count = []
for t in thresholds:
    temp1 = da_points[da_points[0] >= t]
    temp2 = data.merge(temp1, on = ['clusters','msa'],how = 'inner')
    missing_count.append(temp2["mcare_pay_median"].isna().sum())
fig = sns.scatterplot(x=thresholds,y = 100*(missing_count/total_mcare_missing))
fig.set(xlabel ="Threshold", ylabel = "%age of Missing Medicare Median", title␣
 ↪='Threshold v/s Missing Mcare')
```

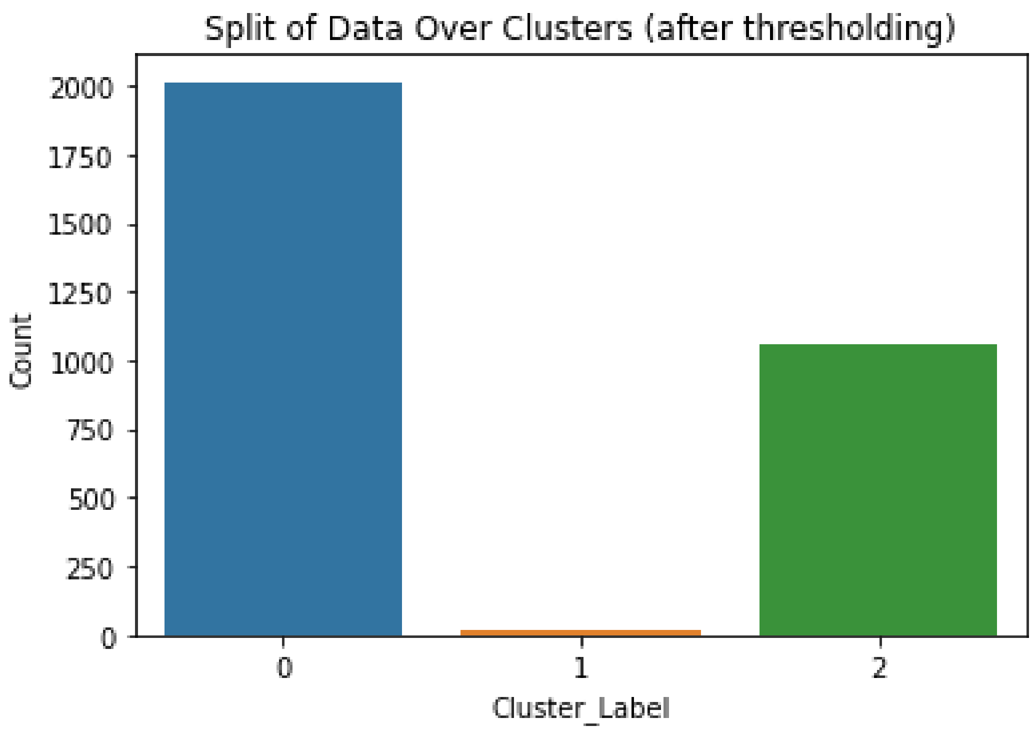
```
[20]: [Text(0.5, 0, 'Threshold'),
       Text(0, 0.5, '%age of Missing Medicare Median'),
       Text(0.5, 1.0, 'Threshold v/s Missing Mcare')]
```



```
[21]: da_points = da_points[da_points[0] >= 15]
```

```
[22]: fig = sns.barplot(data=da_points.groupby(by = ["clusters"]).sum().
       ↪reset_index(),x="clusters",y = 0)
      fig.set(xlabel ="Cluster_Label", ylabel = "Count", title ='Split of Data Over␣
       ↪Clusters (after thresholding)')
```

```
[22]: [Text(0.5, 0, 'Cluster_Label'),
       Text(0, 0.5, 'Count'),
       Text(0.5, 1.0, 'Split of Data Over Clusters (after thresholding)')]
```



0.0.7 filtering out the MSA X Cluster sets which have more than or equal to 15 training examples

```
[23]: filter_df = data.merge(da_points, on = ['clusters','msa'],how = 'inner')
      filter_df = filter_df.merge(cluster_means, on = ['clusters'], how = 'left')
      filter_df["msa_cluster"] = filter_df["clusters"].astype(str) + "_" +␣
       ↪filter_df["msa"].astype(str)
      filter_df
```

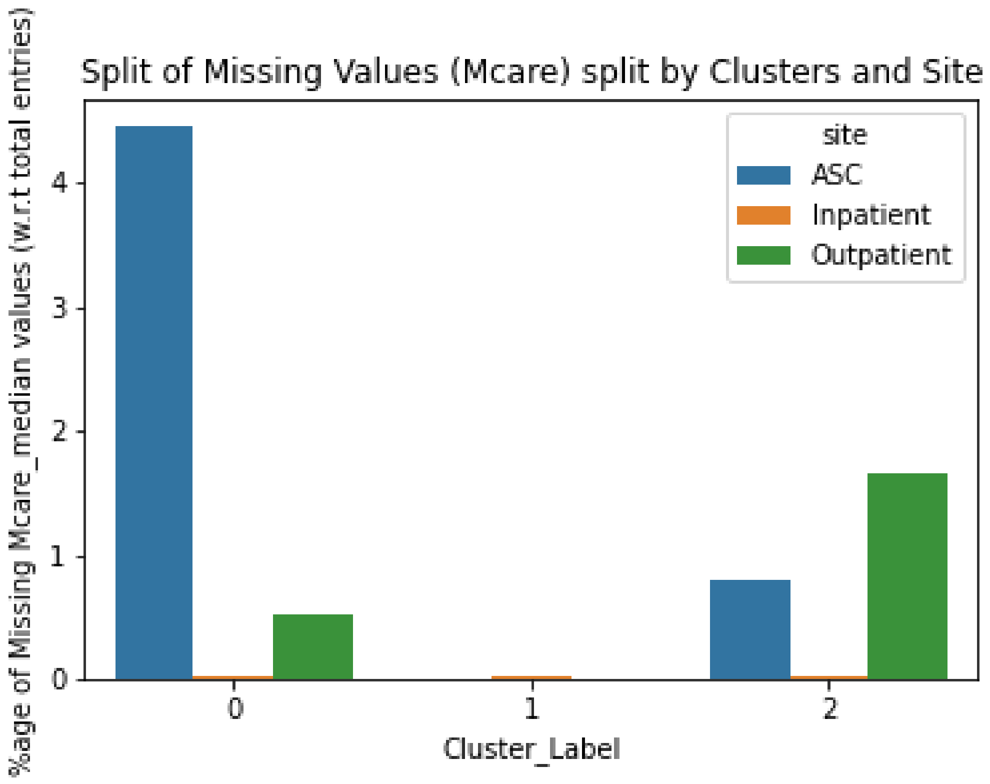
```
[23]:                                          id      msa  year        site  \
      0     26420.0_2018_Inpatient_breast reconstruction  26420.0  2018   Inpatient
      1               26420.0_2020_Outpatient_navigation  26420.0  2020  Outpatient
      2                      26420.0_2019_Outpatient_bsp  26420.0  2019  Outpatient
      3                26420.0_2018_Outpatient_bariatric  26420.0  2018  Outpatient
      4                26420.0_2019_Outpatient_bariatric  26420.0  2019  Outpatient
      …                                              …        …     …           …
      3080                  23104.0_2018_Outpatient_fess  23104.0  2018  Outpatient
      3081             23104.0_2019_Outpatient_hysterect  23104.0  2019  Outpatient
      3082             23104.0_2018_Outpatient_hysterect  23104.0  2018  Outpatient
      3083         23104.0_2019_Outpatient_rtc_slap_bank  23104.0  2019  Outpatient
      3084         23104.0_2018_Outpatient_rtc_slap_bank  23104.0  2018  Outpatient

                      group  priv_count  priv_pay_mean  priv_pay_median  \
0     breast reconstruction       51.0   47327.576470        21408.000
1                navigation       53.0   24278.586230        22610.750
2                       bsp       62.0   12194.493390        12292.785
3                 bariatric       63.0    8510.499365         8440.050
4                 bariatric       65.0   12158.861540         9200.830
…                         …          …              …                …
3080                   fess      559.0   11841.383290        10782.000
3081              hysterect      610.0   14291.381590        13581.885
3082              hysterect      726.0   12070.976400        10439.340
3083           rtc_slap_bank     741.0   17471.221130        17544.300
3084           rtc_slap_bank    1034.0   16664.924480        16274.790

      priv_pay_iqr  mcare_count  …  mcare_pay_median  mcare_pay_sd  \
0        23351.715         81.0  …          10395.16  12242.654150
1         3899.050        427.0  …          10575.60   1806.259876
2        10401.335         63.0  …           3976.12   1578.030820
3        11238.265         49.0  …              0.00   1977.703154
4        13049.110         59.0  …           2181.87   1873.258734
…              …             …  …              …            …
3080      7816.055        453.0  …           3730.22    990.914736
3081      7350.695        212.0  …           6129.90   1782.534108
3082      6875.600        234.0  …           5967.75   1722.511997
3083      9170.440       1368.0  …           4397.70   1625.875048
3084      8958.390       1383.0  …           4299.44   1098.442750

                                 CBSA_NAME  State        lon        lat  \
0     Houston-The Woodlands-Sugar Land, TX  Texas -95.622552  29.598443
1     Houston-The Woodlands-Sugar Land, TX  Texas -95.622552  29.598443
2     Houston-The Woodlands-Sugar Land, TX  Texas -95.622552  29.598443
3     Houston-The Woodlands-Sugar Land, TX  Texas -95.622552  29.598443
4     Houston-The Woodlands-Sugar Land, TX  Texas -95.622552  29.598443
…                                      …    …          …        …
3080         Dallas-Fort Worth-Arlington, TX  Texas -96.920913  32.707875
3081         Dallas-Fort Worth-Arlington, TX  Texas -96.920913  32.707875
3082         Dallas-Fort Worth-Arlington, TX  Texas -96.920913  32.707875
3083         Dallas-Fort Worth-Arlington, TX  Texas -96.920913  32.707875
3084         Dallas-Fort Worth-Arlington, TX  Texas -96.920913  32.707875

      clusters   0  mcare_cluster_mean  msa_cluster
0            0  48          5221.74077    0_26420.0
1            0  48          5221.74077    0_26420.0
2            0  48          5221.74077    0_26420.0
3            0  48          5221.74077    0_26420.0
4            0  48          5221.74077    0_26420.0
…          …   ..                 …            …
3080         0  43          5221.74077    0_23104.0
3081         0  43          5221.74077    0_23104.0
```

```
3082          0  43          5221.74077    0_23104.0
3083          0  43          5221.74077    0_23104.0
3084          0  43          5221.74077    0_23104.0

[3085 rows x 22 columns]
```

0.0.8 How the missing and non missing values are distributed based on the cluster as well as the type of site (Inpatient, Outpatient or ASC)

```
[24]: missing_mcare = filter_df[filter_df["mcare_pay_median"].
       ↪isna()][["msa","clusters","site"]]
      temp = pd.DataFrame({"msa":[0, 0, 0],
                           "clusters":[1, 2, 0],
                          "site":["Inpatient","Inpatient","Inpatient"]})
      x = missing_mcare.append(temp)[["clusters","site"]].groupby(by =␣
       ↪["clusters","site"]).size().reset_index()
      x[0] = (x[0]/data.shape[0])*100
      fig = sns.barplot(data=x,x="clusters",y = 0,hue="site")
      fig.set(xlabel ="Cluster_Label", ylabel = "%age of Missing Mcare_median values␣
       ↪(w.r.t total entries)", title ='Split of Missing Values (Mcare) split by␣
       ↪Clusters and Site')
```

```
[24]: [Text(0.5, 0, 'Cluster_Label'),
       Text(0, 0.5, '%age of Missing Mcare_median values (w.r.t total entries)'),
       Text(0.5, 1.0, 'Split of Missing Values (Mcare) split by Clusters and Site')]
```



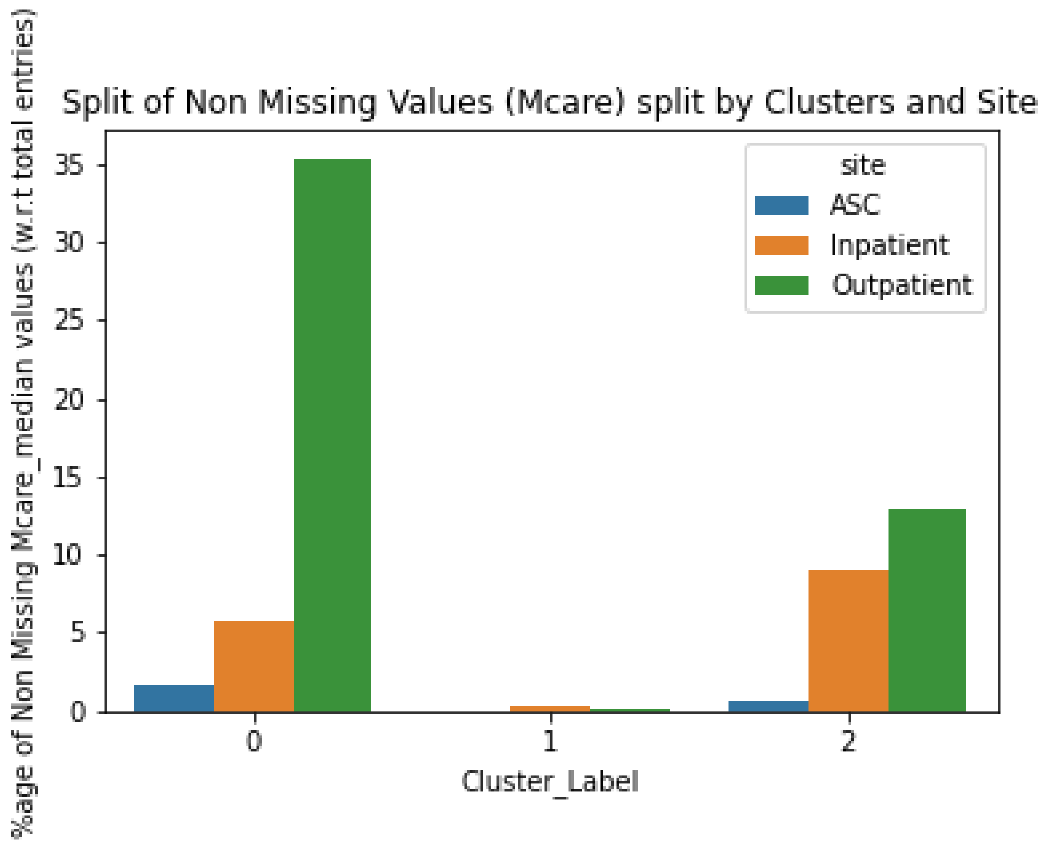
```
[25]: missing_mcare.append(temp)[["clusters","site"]].groupby(by =␣
       ↪["clusters","site"]).size().reset_index()
      x[0] = (x[0]/total_mcare_missing)*100
      x
```

```
[25]:    clusters        site         0
      0         0         ASC  0.948092
      1         0   Inpatient  0.005043
      2         0  Outpatient  0.110947
      3         1   Inpatient  0.005043
      4         2         ASC  0.171463
      5         2   Inpatient  0.005043
      6         2  Outpatient  0.353013
```

```
[26]: non_missing_mcare = filter_df[~filter_df["mcare_pay_median"].
       ↪isna()][["msa","clusters","site"]]
      x =non_missing_mcare[["clusters","site"]].groupby(by = ["clusters","site"]).
       ↪size().reset_index()
      x[0] = (x[0]/data.shape[0])*100
      fig = sns.barplot(data=x,x="clusters",y = 0,hue="site")
```

```
fig.set(xlabel ="Cluster_Label", ylabel = "%age of Non Missing Mcare_median␣
↪values (w.r.t total entries)", title ='Split of Non Missing Values (Mcare)␣
↪split by Clusters and Site')
```

[26]:
```
[Text(0.5, 0, 'Cluster_Label'),
 Text(0, 0.5, '%age of Non Missing Mcare_median values (w.r.t total entries)'),
 Text(0.5, 1.0, 'Split of Non Missing Values (Mcare) split by Clusters and
Site')]
```



[27]:
```
non_missing_mcare[["clusters","site"]].groupby(by = ["clusters","site"]).size().
↪reset_index()
```

[27]:
```
   clusters        site     0
0         0         ASC    67
1         0   Inpatient   245
2         0  Outpatient  1491
3         1   Inpatient    13
4         1  Outpatient     3
5         2         ASC    30
6         2   Inpatient   379
```

```
7          2  Outpatient   543
```

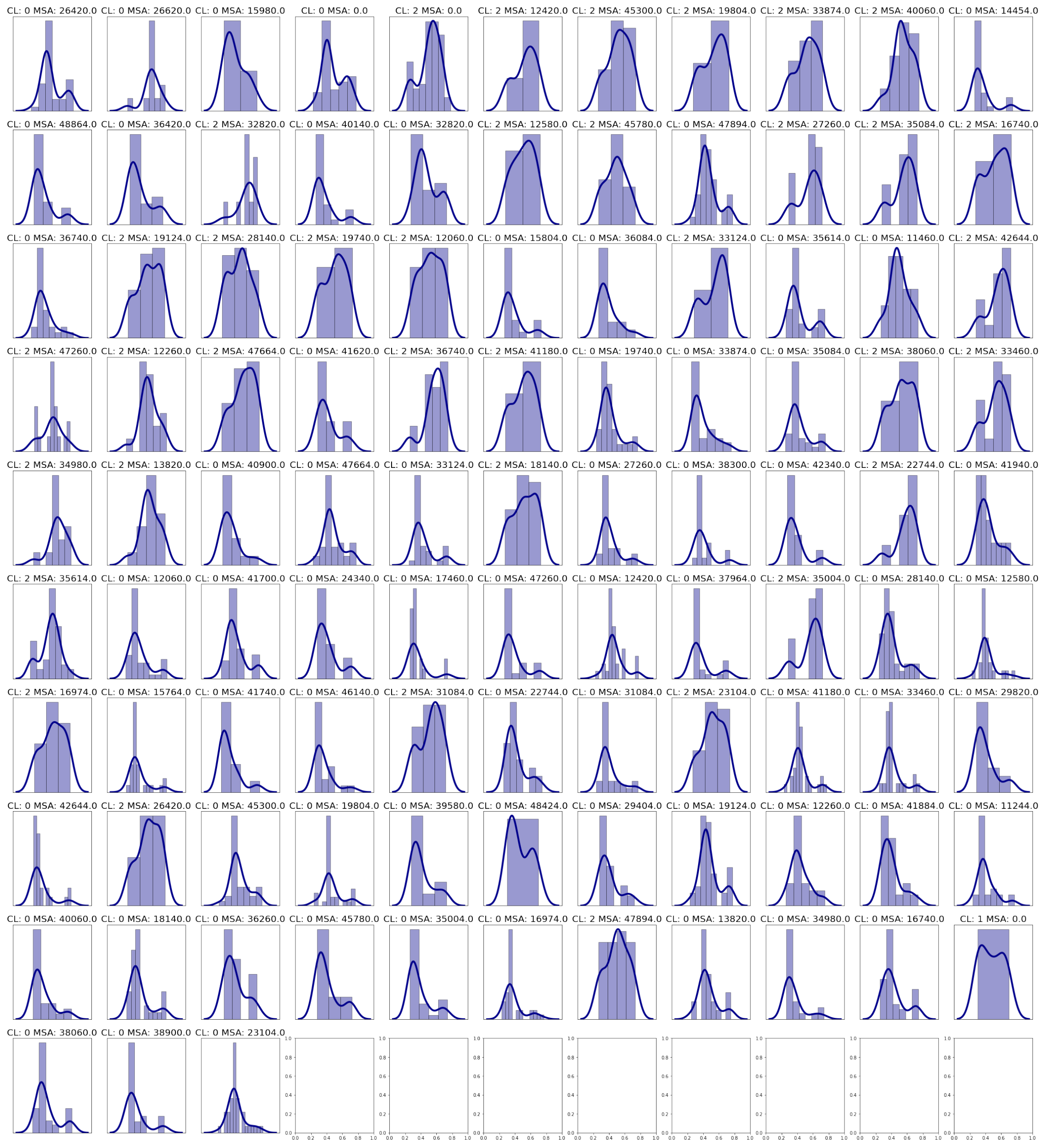
[28]:
```
filter_df[(filter_df["msa_cluster"] == "0_26420.0")]["mcare_cluster_mean"].
↪unique()[0]
```

[28]:
```
5221.740770207848
```

0.0.9 plotting these distributions using histogram plots and KDE plots (to look for multimodality), we also quantify this multimodality using hartigan's dip test and perform normality tests, t - tests (to check if mean of the distribution is same as the mean of the cluster it belongs to).

[29]:
```
uniq_msa_clust = filter_df["msa_cluster"].unique()
grid = mt.ceil(np.sqrt(len(uniq_msa_clust)))
```

[30]:
```
fig, axs = plt.subplots(grid-1, grid, figsize=(40,45))
normality_test = []
t_test_p = []
t_test_st = []
clusters = []
msas = []
interval_list = []
n_pts = []
missing_pts = []
means = []
sd_list = []
for i,x in enumerate(uniq_msa_clust):

    dt_pts = filter_df[(filter_df["msa_cluster"] == x)]["mcare_pay_median"]
    clusters.append(int(x.split("_")[0]))
    msas.append(float(x.split("_")[1]))
    k ,p = stats.normaltest(dt_pts,nan_policy="omit")
#     label.append(x)
    normality_test.append(p)
    n_pts.append(len(dt_pts[~np.isnan(dt_pts)]))
    missing_pts.append(len(dt_pts[np.isnan(dt_pts)]))
    gp_stats = dt_pts.describe()
    mu = gp_stats[1]
    std = gp_stats[2]
    means.append(mu)
    sd_list.append(std)
    cl_mean = filter_df[(filter_df["msa_cluster"] == x)]["mcare_cluster_mean"].
↪unique()[0]
    t_test = stats.ttest_1samp(dt_pts, popmean=cl_mean,nan_policy='omit')
    t_test_p.append(t_test[1])
    t_test_st.append(t_test[0])
    Q1 = np.percentile(dt_pts[~np.isnan(dt_pts)], 25, interpolation =␣
↪'midpoint')
    Q3 = np.percentile(dt_pts[~np.isnan(dt_pts)], 75, interpolation =␣
↪'midpoint')
    IQR = Q3 - Q1
    if len(dt_pts) == 0:
        opt_bins = 10
    else:
        bin_width = 2*IQR/np.cbrt(len(dt_pts))
        rng = max(dt_pts[~np.isnan(dt_pts)]) - min(dt_pts[~np.isnan(dt_pts)])
        opt_bins = int(rng/bin_width)
    sns.distplot(dt_pts, hist=True, kde=True,
             bins=opt_bins, color = 'darkblue',
             hist_kws={'edgecolor':'black'},
             kde_kws={'linewidth': 4},
                ax=axs[int(i/grid),i%grid])
    dt_pts = np.msort(dt_pts)
    intervals = UniDip(dt_pts[~np.isnan(dt_pts)]).run()
    interval_list.append(intervals)
    axs[int(i/grid),i%grid].get_yaxis().set_visible(False)
    axs[int(i/grid),i%grid].get_xaxis().set_visible(False)
    axs[int(i/grid),i%grid].set_title("CL: " + str(int(x.split("_")[0])) + "␣
↪MSA: " + str(float(x.split("_")[1])),fontsize = 20)
# plt.savefig("density_plots.png")
#       axs[int(i/grid),i%grid].set_title(intervals[0][1])
```



0.0.10 saving useful metrics from the above distributions and tests in a dataframe

```
[31]: output_df = pd.DataFrame(list(zip(clusters,msas,␣
       ↪fdrcorrection(normality_test,alpha=0.1)[1],fdrcorrection(t_test_p,alpha=0.
       ↪1)[1],t_test_st,interval_list,n_pts,missing_pts,means,sd_list)),columns=["Cluster_Label","M
      output_df["msa_cluster"] = output_df["Cluster_Label"].astype(str) + "_" +␣
       ↪output_df["MSA"].astype(str)
```

```
# print("%age of MSAxCluster groups where we cannot reject the NULL hypothesis␣
↪is (90% CI) :",100*len(output_df[output_df["Normality_p_value"] >= 0.1])/
↪len(output_df))
# output_df.sort_values(by=["Normality_p_value"],ascending=False)
output_df
```

```
[31]:      Cluster_Label      MSA  Normality_p_value  T_test_p_value  T-Statistic  \
     0                 0  26420.0           0.127812        0.788805     0.620920
     1                 0  26620.0           0.013838        0.123862    -2.943756
     2                 0  15980.0           0.422595        0.316869    -1.843431
     3                 0      0.0           0.031388        0.303299     1.756479
     4                 2      0.0           0.080576        0.924204     0.177721
     ..              …        …                …             …           …
     97                0  16740.0           0.046467        0.648921    -0.947087
     98                1      0.0           0.062180        0.181929    -2.349800
     99                0  38060.0           0.013838        0.924204     0.135872
     100               0  38900.0           0.004533        0.631821     1.061741
     101               0  23104.0           0.016144        0.241226    -2.010111

         dip_test_interval  no_pts  no_missing_mcare          mean            SD  \
     0            [(0, 44)]      45                 3   5486.180995   2856.920569
     1            [(0, 11)]      12                 3   3945.866667   1501.400623
     2            [(0, 12)]      13                 3   4703.731538   1013.170067
     3            [(0, 72)]      73                 9   5829.697603   2957.272034
     4            [(0, 81)]      82                13  11668.956890   5658.711046
     ..                 …     …                …           …           …
     97           [(0, 36)]      37                 6   4831.688772   2505.147573
     98           [(0, 15)]      16                 0  20374.278437  10163.846384
     99           [(0, 38)]      39                 3   5275.723344   2481.169257
     100          [(0, 23)]      24                 3   5798.572292   2661.559017
     101          [(0, 38)]      39                 4   4502.874443   2233.368042

         msa_cluster
     0     0_26420.0
     1     0_26620.0
     2     0_15980.0
     3         0_0.0
     4         2_0.0
     ..          …
     97    0_16740.0
     98        1_0.0
     99    0_38060.0
     100   0_38900.0
     101   0_23104.0

     [102 rows x 11 columns]
```

```
[32]: output_df[(output_df["T_test_p_value"] > 0.5) & (output_df["no_pts"] > 25)]
```

```
[32]:     Cluster_Label      MSA  Normality_p_value  T_test_p_value  T-Statistic  \
      0               0  26420.0           0.127812        0.788805     0.620920
      4               2      0.0           0.080576        0.924204     0.177721
      6               2  45300.0           0.389993        0.790155    -0.568563
      7               2  19804.0           0.116350        0.853197    -0.457147
      18              0  47894.0           0.047050        0.990200    -0.012364
      21              2  16740.0           0.025541        0.788805    -0.618736
      22              0  36740.0           0.012674        0.886566    -0.377989
      23              2  19124.0           0.087106        0.924204    -0.244749
      35              2  47664.0           0.221029        0.942695     0.084182
      39              0  19740.0           0.008259        0.788805    -0.692952
      40              0  33874.0           0.017239        0.853197     0.451613
      41              0  35084.0           0.031388        0.548577     1.234094
      42              2  38060.0           0.062180        0.892486    -0.351189
      47              0  47664.0           0.167777        0.788805    -0.615943
      48              0  33124.0           0.011629        0.788805     0.707992
      49              2  18140.0           0.046467        0.790155    -0.573838
      50              0  27260.0           0.004533        0.537178    -1.283444
      56              0  12060.0           0.012674        0.788805    -0.617792
      57              0  41700.0           0.039443        0.531739    -1.305039
      66              2  16974.0           0.168960        0.924204    -0.130666
      67              0  15764.0           0.003276        0.546821     1.256212
      71              0  22744.0           0.039328        0.857795     0.427354
      73              2  23104.0           0.346953        0.898509    -0.303102
      75              0  33460.0           0.004629        0.931201     0.110398
      77              0  42644.0           0.003270        0.656180     0.931989
      78              2  26420.0           0.256317        0.892486    -0.341462
      79              0  45300.0           0.221029        0.590400    -1.152877
      80              0  19804.0           0.061373        0.898509    -0.299509
      84              0  19124.0           0.092272        0.788805    -0.639812
      87              0  11244.0           0.007252        0.631821     1.009715
      88              0  40060.0           0.009285        0.631821    -1.023471
      89              0  18140.0           0.013838        0.905487    -0.279896
      93              0  16974.0           0.003276        0.853197     0.442169
      94              2  47894.0           0.345591        0.788805    -0.725179
      97              0  16740.0           0.046467        0.648921    -0.947087
      99              0  38060.0           0.013838        0.924204     0.135872

        dip_test_interval  no_pts  no_missing_mcare          mean           SD  \
      0          [(0, 44)]      45                 3   5486.180995  2856.920569
      4          [(0, 81)]      82                13  11668.956890  5658.711046
      6          [(0, 30)]      31                 1  11054.320806  4931.385112
      7          [(0, 25)]      26                 3  11047.795192  5689.699707
      18         [(0, 38)]      39                 3   5216.321102  2737.355990
      21         [(0, 33)]      34                 2  10975.952136  5484.254439
```

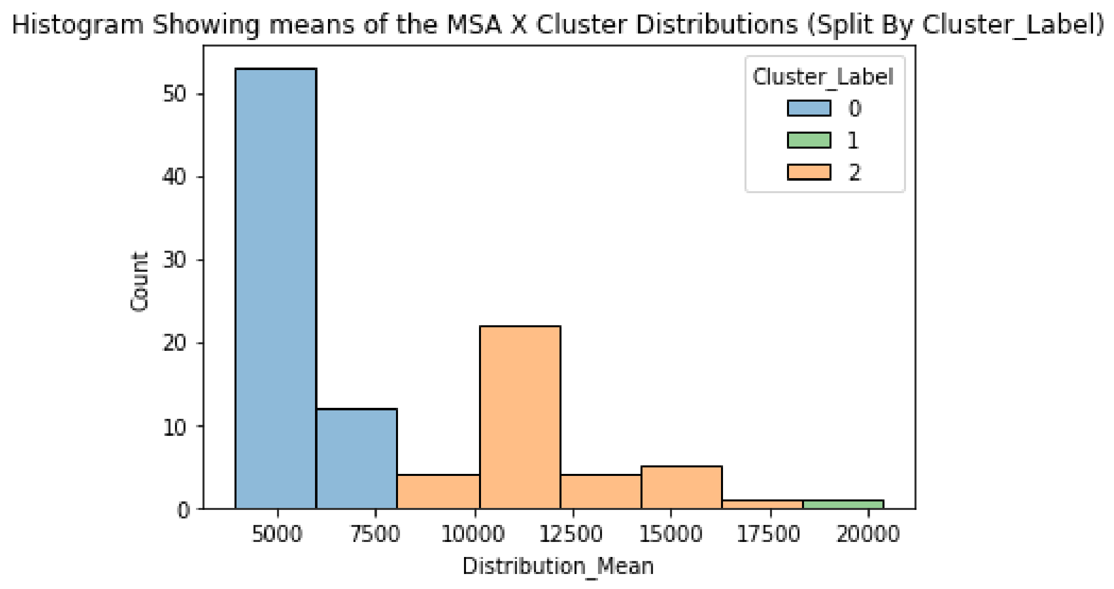
```
22         [(0, 28)]      29                   3   5060.612045  2295.584796
23         [(0, 36)]      37                   2  11345.310382  5283.491647
35         [(0, 31)]      32                   7  11641.294203  5603.976773
39         [(0, 31)]      32                   6   4957.776087  2154.852929
40         [(0, 32)]      33                   3   5380.861515  2024.031903
41         [(0, 33)]      34                   4   5818.803129  2821.050857
42         [(0, 35)]      36                   3  11226.210943  5666.830722
47         [(0, 41)]      42                   3   4979.627948  2547.426171
48         [(0, 31)]      32                   4   5558.630786  2691.752250
49         [(0, 32)]      33                   5  11015.573819  5429.094858
50         [(0, 26)]      27                   3   4744.015741  1934.117228
56         [(0, 43)]      44                   3   4991.558520  2471.475014
57         [(0, 26)]      27                   4   4729.288408  1960.751466
66         [(0, 29)]      30                   6  11425.910033  5532.690456
67         [(0, 27)]      28                   1   5795.347857  2416.187603
71         [(0, 25)]      26                   3   5456.377433  2799.593927
73         [(0, 30)]      31                   3  11284.967097  5013.558166
75         [(0, 26)]      27                   3   5268.885514  2218.991225
77         [(0, 25)]      26                   3   5665.219231  2426.321132
78         [(0, 32)]      33                   3  11241.610606  5321.063152
79         [(0, 35)]      36                   3   4800.723448  2191.130885
80         [(0, 34)]      35                   3   5075.306000  2892.465696
84         [(0, 44)]      45                   3   4972.795809  2610.098853
87         [(0, 26)]      27                   3   5681.337734  2365.157247
88         [(0, 25)]      26                   4   4857.313846  1815.606184
89         [(0, 35)]      36                   4   5100.702750  2594.632371
93         [(0, 39)]      40                   3   5426.624379  2930.545850
94         [(0, 28)]      29                   3  10848.461704  5268.270167
97         [(0, 36)]      37                   6   4831.688772  2505.147573
99         [(0, 38)]      39                   3   5275.723344  2481.169257

   msa_cluster
0    0_26420.0
4        2_0.0
6    2_45300.0
7    2_19804.0
18   0_47894.0
21   2_16740.0
22   0_36740.0
23   2_19124.0
35   2_47664.0
39   0_19740.0
40   0_33874.0
41   0_35084.0
42   2_38060.0
47   0_47664.0
48   0_33124.0
```

```
49    2_18140.0
50    0_27260.0
56    0_12060.0
57    0_41700.0
66    2_16974.0
67    0_15764.0
71    0_22744.0
73    2_23104.0
75    0_33460.0
77    0_42644.0
78    2_26420.0
79    0_45300.0
80    0_19804.0
84    0_19124.0
87    0_11244.0
88    0_40060.0
89    0_18140.0
93    0_16974.0
94    2_47894.0
97    0_16740.0
99    0_38060.0
```

0.0.11

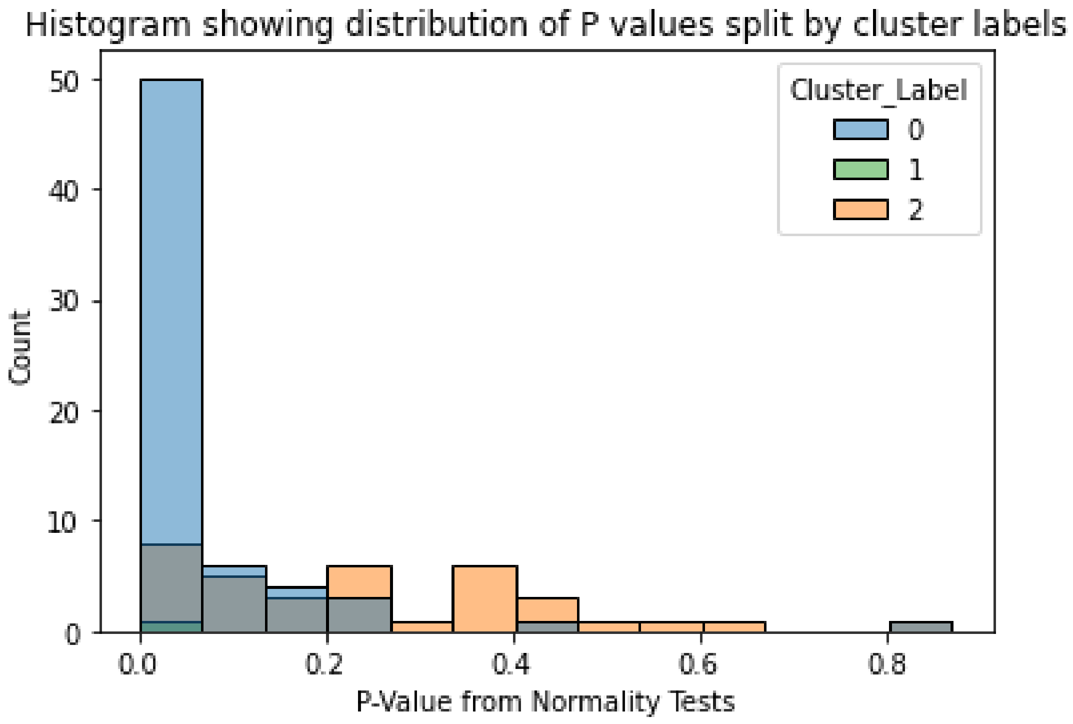
```
[33]: fig = sns.histplot(data=output_df, x="mean", hue="Cluster_Label",palette =␣
       ↪['tab:blue', 'tab:green', 'tab:orange'])
      fig.set(xlabel ="Distribution_Mean", ylabel = "Count", title ='Histogram␣
       ↪Showing means of the MSA X Cluster Distributions (Split By Cluster_Label)')
```

```
[33]: [Text(0.5, 0, 'Distribution_Mean'),
       Text(0, 0.5, 'Count'),
       Text(0.5, 1.0, 'Histogram Showing means of the MSA X Cluster Distributions
      (Split By Cluster_Label)')]
```



```
[34]: fig = sns.histplot(data=output_df, x="Normality_p_value",␣
       ↪hue="Cluster_Label",palette = ['tab:blue', 'tab:green', 'tab:orange'])
      fig.set(xlabel ="P-Value from Normality Tests", ylabel = "Count", title␣
       ↪='Histogram showing distribution of P values split by cluster labels')
```

```
[34]: [Text(0.5, 0, 'P-Value from Normality Tests'),
       Text(0, 0.5, 'Count'),
       Text(0.5, 1.0, 'Histogram showing distribution of P values split by cluster
      labels')]
```



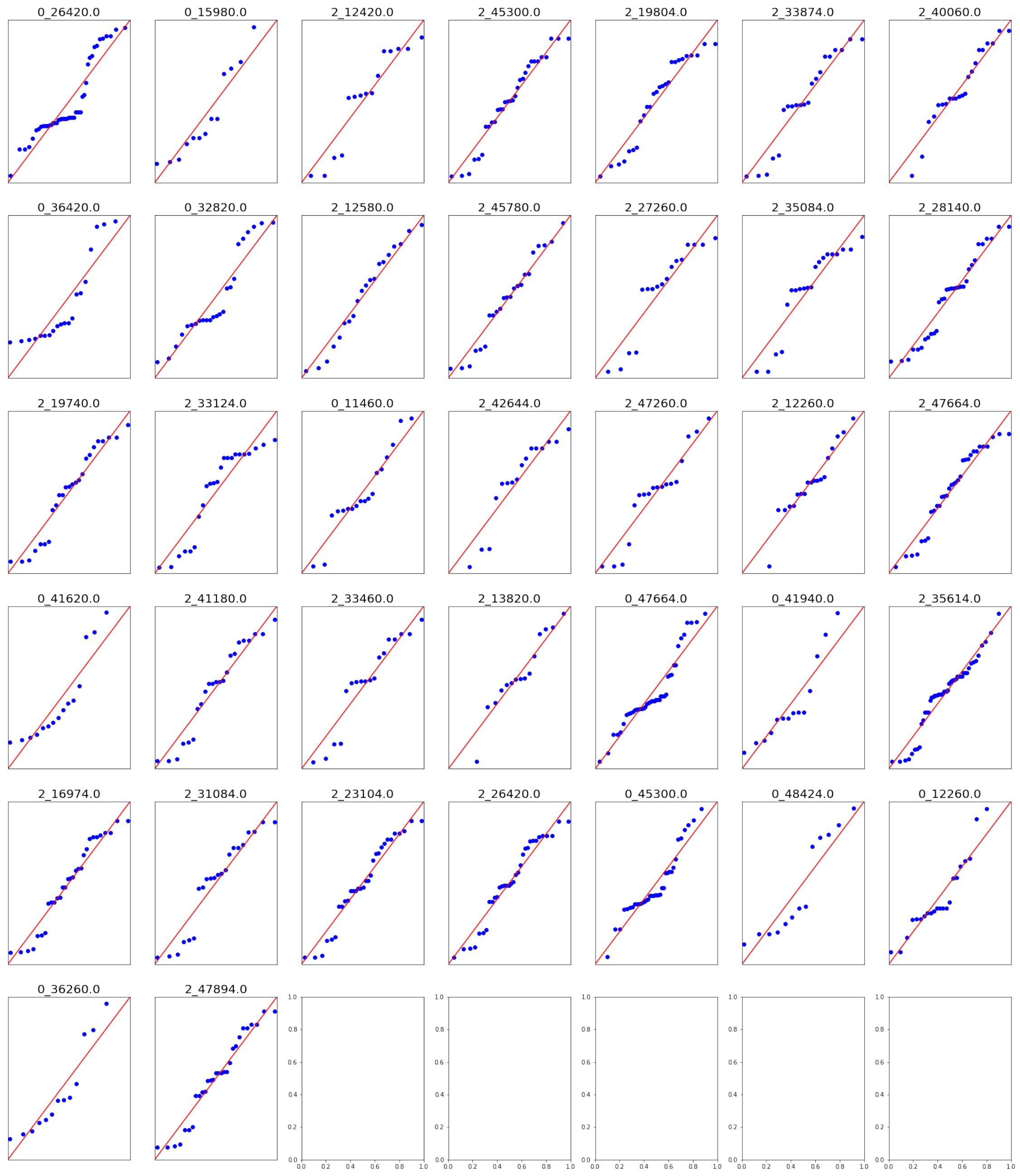
0.0.12 Checking how many missing values do we have for Mcare median where we cannot reject the NULL hyp. in the normality test (i.e. the distributions are normal)

[35]:
```
output_df[output_df["Normality_p_value"] > 0.1]["no_missing_mcare"].sum()
```

[35]:
```
94
```

0.0.13 To confirm if these distrbutions are actually normal, we see their QQ plots

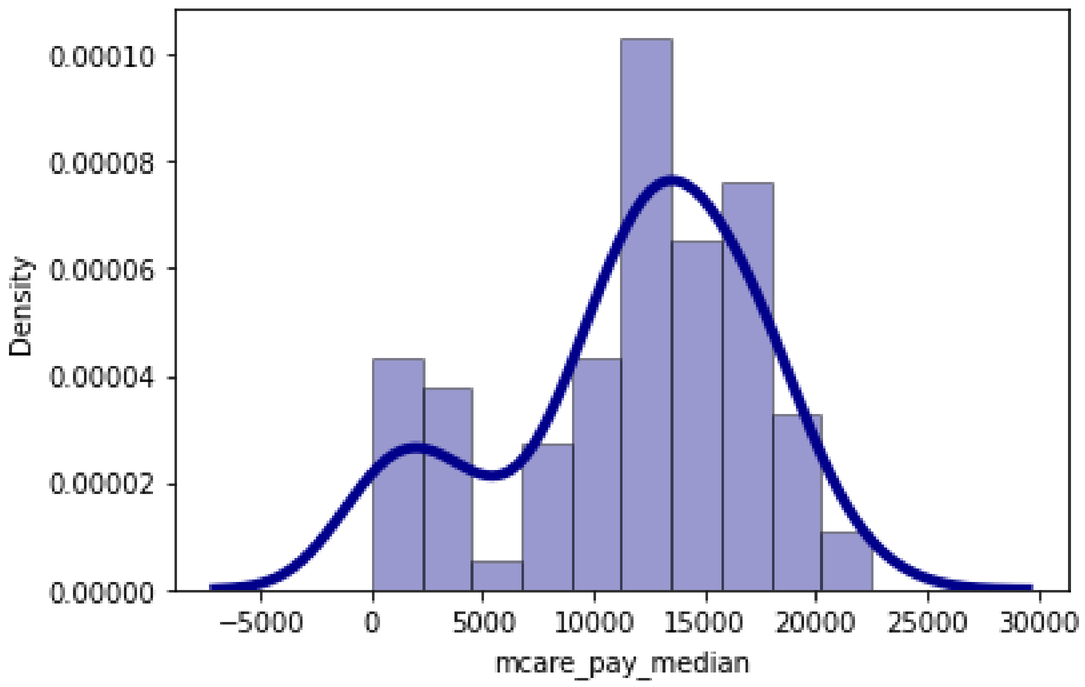
[36]:
```
normality_check = filter_df.merge(output_df[output_df["Normality_p_value"] > 0.
 ↪1], on = ["msa_cluster"],how = 'inner')
uniq_msa_clust_norm = normality_check["msa_cluster"].unique()
grid = mt.ceil(np.sqrt(len(uniq_msa_clust_norm)))
fig, axs = plt.subplots(grid-1, grid, figsize=(30,35))
for i,x in enumerate(uniq_msa_clust_norm):
    dt_pts = normality_check[(normality_check["msa_cluster"] ==␣
 ↪x)]["mcare_pay_median"]
    gp_stats = dt_pts.describe()
    mu = gp_stats[1]
    std = gp_stats[2]
    rmv_outliers = dt_pts[(dt_pts > mu - 3*std) & (dt_pts < mu + 3*std)].values
    sm.qqplot(rmv_outliers,dist=norm(mu,std), line ='45',ax=axs[int(i/
↪grid),i%grid])
    axs[int(i/grid),i%grid].get_yaxis().set_visible(False)
    axs[int(i/grid),i%grid].get_xaxis().set_visible(False)
    axs[int(i/grid),i%grid].set_title(str(x),fontsize = 20)
```



0.0.14 trying to remove multimodality by splitting into ASC, inpatient and outpatient (testing this out for one of the examples)

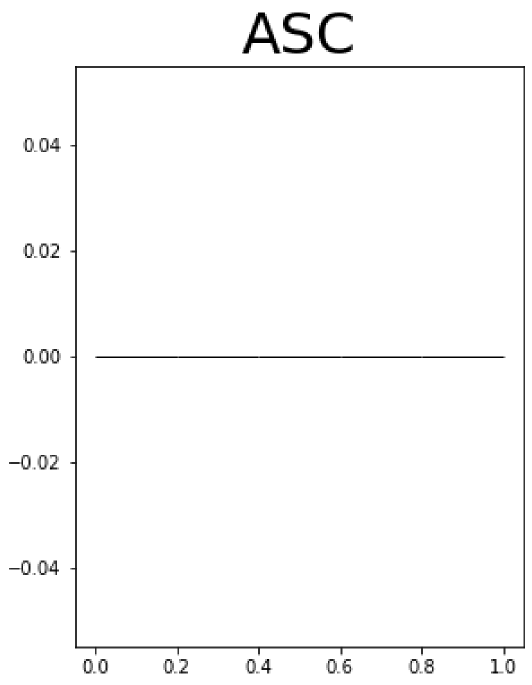
[37]:
```
multi_mod_test = filter_df[filter_df["msa_cluster"] == uniq_msa_clust[4]]
sns.distplot(multi_mod_test["mcare_pay_median"], hist=True, kde=True,
             bins=10, color = 'darkblue',
             hist_kws={'edgecolor':'black'},
             kde_kws={'linewidth': 4})
```

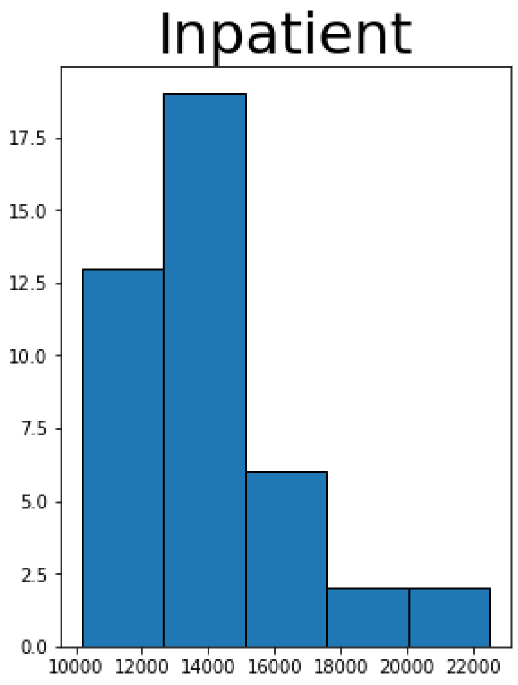
[37]:
```
<AxesSubplot:xlabel='mcare_pay_median', ylabel='Density'>
```

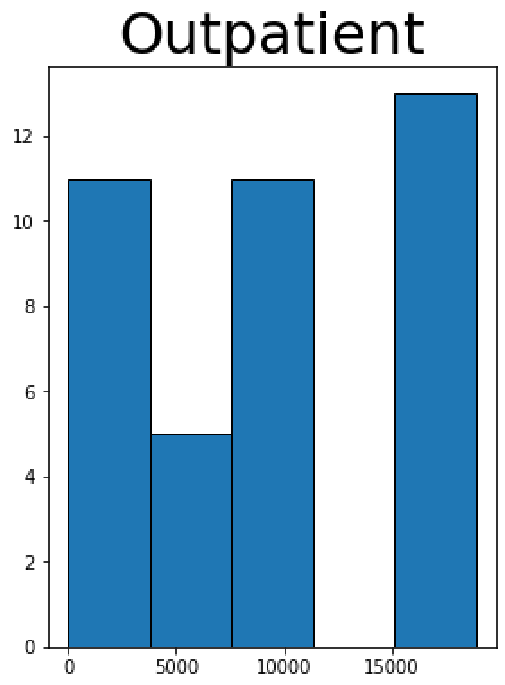


[38]:
```
fig = plt.figure(figsize = (10,15))
ax = fig.gca()
plt.tight_layout()
fig = (multi_mod_test["mcare_pay_median"]).hist(by = multi_mod_test["site"]␣
 ↪,bins = 5, ax = ax,ec="k",rot = 0)
[x.title.set_size(32) for x in fig.ravel()]
# plt.savefig("diff_gp_hist.png",dpi = 100)
```

[38]:
```
[None, None, None, None]
```

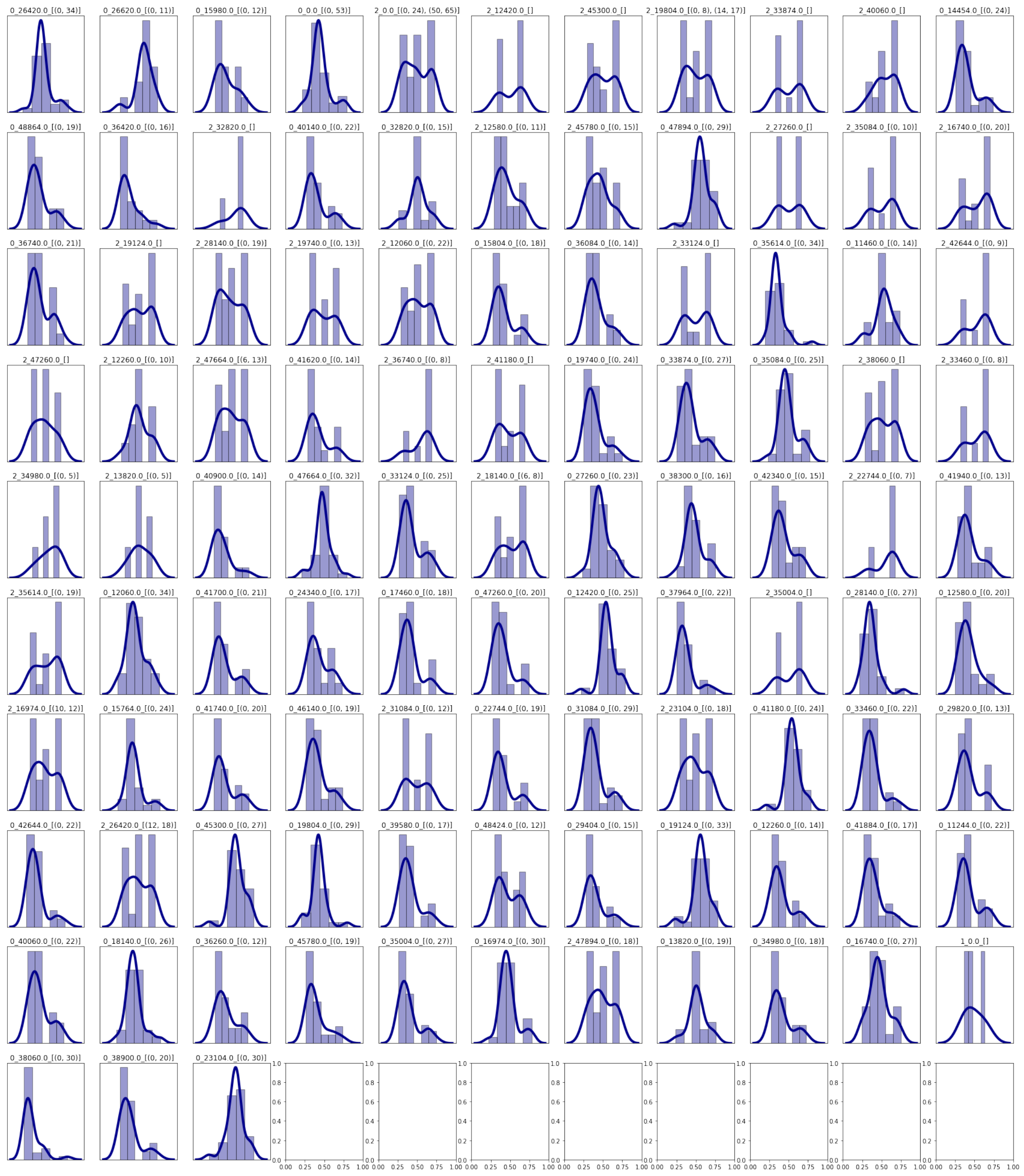

ASC
0.04
0.02
0.00
−0.02
−0.04
0.0
0.2
0.4
0.6
0.8
1.0


Inpatient
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
10000
12000
14000
16000
18000
20000
22000


Outpatient
12
10
8
6
4
2
0
0
5000
10000
15000

0.0.15 looking at distributions for mcare_pay_median for outpatient site only

[39]:
```
grid = mt.ceil(np.sqrt(len(uniq_msa_clust)))
fig, axs = plt.subplots(grid-1, grid, figsize=(30,35))
t_test_p = []
t_test_st = []
means = []
sd_list = []
clusters = []
msas = []
n_pts = []
missing_pts = []
for i,x in enumerate(uniq_msa_clust):
    dt_pts = filter_df[(filter_df["msa_cluster"] == x) & (filter_df["site"] ==␣
 ↪"Outpatient")]["mcare_pay_median"]
#     Q1 = np.percentile(dt_pts[~np.isnan(dt_pts)], 25, interpolation =␣
 ↪'midpoint')
#     Q3 = np.percentile(dt_pts[~np.isnan(dt_pts)], 75, interpolation =␣
 ↪'midpoint')
#     IQR = Q3 - Q1

#     if (opt_bins == 0) or IQR == 0:
#         opt_bins = 10
#     else:
#         bin_width = 2*IQR/np.cbrt(len(dt_pts))
#         rng = max(dt_pts[~np.isnan(dt_pts)]) - min(dt_pts[~np.isnan(dt_pts)])
#         opt_bins = int(rng/bin_width)

#     if opt_bins == 0:
#         opt_bins = 5
    sns.distplot(dt_pts, hist=True, kde=True,
                 bins=5, color = 'darkblue',
                 hist_kws={'edgecolor':'black'},
                 kde_kws={'linewidth': 4},
                    ax=axs[int(i/grid),i%grid])
    t_test = stats.ttest_1samp(dt_pts, popmean=np.
 ↪mean(dt_pts),nan_policy='omit')
    t_test_p.append(t_test[1])
    t_test_st.append(t_test[0])
    clusters.append(int(x.split("_")[0]))
    msas.append(float(x.split("_")[1]))
    n_pts.append(len(dt_pts[~np.isnan(dt_pts)]))
    missing_pts.append(len(dt_pts[np.isnan(dt_pts)]))
    gp_stats = dt_pts.describe()
    mu = gp_stats[1]
    std = gp_stats[2]
    means.append(mu)
    sd_list.append(std)
    dt_pts = np.msort(dt_pts)
    intervals = UniDip(dt_pts[~np.isnan(dt_pts)]).run()
    axs[int(i/grid),i%grid].get_yaxis().set_visible(False)
    axs[int(i/grid),i%grid].get_xaxis().set_visible(False)
    axs[int(i/grid),i%grid].set_title(str(x) + "_" +str(intervals))
# plt.savefig("density_plots.png")
#     axs[int(i/grid),i%grid].set_title(intervals[0][1])
```



```
[40]: output_df = pd.
       ↪DataFrame(list(zip(clusters,msas,t_test_p,t_test_st,n_pts,missing_pts,means,sd_list)),colum
      output_df["msa_cluster"] = output_df["Cluster_Label"].astype(str) + "_" +␣
       ↪output_df["MSA"].astype(str)
      output_df
```

```
[40]:      Cluster_Label      MSA  T_test_p_value   T-Statistic  no_pts  \
      0                0  26420.0             1.0  0.000000e+00      35
      1                0  26620.0             1.0  0.000000e+00      12
      2                0  15980.0             1.0 -6.473207e-15      13
      3                0      0.0             1.0  3.070660e-15      54
      4                2      0.0             1.0  0.000000e+00      40
      ..             …        …             …           …     …
      97               0  16740.0             1.0  3.995134e-15      28
      98               1      0.0             1.0  0.000000e+00       3
      99               0  38060.0             1.0 -3.481811e-15      31
      100              0  38900.0             1.0  3.743855e-15      21
      101              0  23104.0             1.0  1.984632e-15      31

           no_missing_mcare         mean           SD msa_cluster
      0                   0  4766.000571  1985.854460   0_26420.0
      1                   0  3945.866667  1501.400623   0_26620.0
      2                   0  4703.731538  1013.170067   0_15980.0
      3                   0  4541.429444  2176.533544       0_0.0
      4                   1  9116.143500  6812.260432       2_0.0
      ..                …          …          …         …
      97                  3  4054.546607  1204.613751   0_16740.0
      98                  0  8983.376667  1047.786119       1_0.0
      99                  0  4722.290000  1454.372916   0_38060.0
      100                 0  4892.516429  1113.245254   0_38900.0
      101                 1  3965.257258  1275.765891   0_23104.0

      [102 rows x 9 columns]
```

```
[41]: output_df[(output_df["no_missing_mcare"] != 0)]
```

```
[41]:      Cluster_Label      MSA  T_test_p_value   T-Statistic  no_pts  \
      4                2      0.0             1.0  0.000000e+00      40
      5                2  12420.0             1.0  0.000000e+00       9
      6                2  45300.0             1.0  0.000000e+00      20
      7                2  19804.0             1.0  0.000000e+00      17
      8                2  33874.0             1.0 -8.215408e-16      12
      9                2  40060.0             1.0 -1.244729e-15      13
      13               2  32820.0             1.0  0.000000e+00       4
      16               2  12580.0             1.0  0.000000e+00      12
      20               2  35084.0             1.0  0.000000e+00      11
      21               2  16740.0             1.0 -1.348174e-15      21
```

```
23                2  19124.0              1.0 -2.877737e-15      22
24                2  28140.0              1.0  0.000000e+00      20
25                2  19740.0              1.0  0.000000e+00      14
26                2  12060.0              1.0  2.760690e-15      23
27                0  15804.0              1.0  0.000000e+00      19
32                2  42644.0              1.0  1.591858e-15      10
34                2  12260.0              1.0  0.000000e+00      11
35                2  47664.0              1.0  1.239041e-15      19
37                2  36740.0              1.0  8.800847e-16       9
38                2  41180.0              1.0 -1.107192e-15      14
39                0  19740.0              1.0  0.000000e+00      25
41                0  35084.0              1.0  0.000000e+00      26
42                2  38060.0              1.0  0.000000e+00      22
43                2  33460.0              1.0 -7.509264e-16       9
44                2  34980.0              1.0  0.000000e+00       6
45                2  13820.0              1.0  0.000000e+00       6
48                0  33124.0              1.0  0.000000e+00      26
51                0  38300.0              1.0  3.645283e-15      17
55                2  35614.0              1.0  0.000000e+00      20
57                0  41700.0              1.0  4.835056e-15      22
60                0  47260.0              1.0  0.000000e+00      21
63                2  35004.0              1.0  0.000000e+00      10
64                0  28140.0              1.0  0.000000e+00      28
66                2  16974.0              1.0 -1.225100e-15      18
70                2  31084.0              1.0  0.000000e+00      13
73                2  23104.0              1.0 -2.717326e-15      19
74                0  41180.0              1.0 -3.948769e-15      25
78                2  26420.0              1.0  0.000000e+00      21
81                0  39580.0              1.0  0.000000e+00      18
88                0  40060.0              1.0  0.000000e+00      23
89                0  18140.0              1.0  0.000000e+00      27
91                0  45780.0              1.0  0.000000e+00      20
94                2  47894.0              1.0  0.000000e+00      19
95                0  13820.0              1.0  0.000000e+00      20
97                0  16740.0              1.0  3.995134e-15      28
101               0  23104.0              1.0  1.984632e-15      31

     no_missing_mcare          mean           SD msa_cluster
4                   1   9116.143500  6812.260432       2_0.0
5                   3  10928.281111  7253.122933   2_12420.0
6                   1  10380.367500  5963.467812   2_45300.0
7                   3   9505.459706  6424.323965   2_19804.0
8                   1  11622.231667  7669.934125   2_33874.0
9                   3  11742.670769  5268.985845   2_40060.0
13                  3  12758.817500  7142.368857   2_32820.0
16                  3  12655.144583  6949.849871   2_12580.0
20                  2  13783.368182  8153.903097   2_35084.0
```

```
21                    1  11587.624524  6182.923986   2_16740.0
23                    2  10993.558182  5929.531110   2_19124.0
24                    3   8534.549500  6413.780630   2_28140.0
25                    1   9955.000714  6559.401177   2_19740.0
26                    3   9555.180000  6319.845081   2_12060.0
27                    2   4833.425000  1145.035282   0_15804.0
32                    2  13885.312000  7226.961748   2_42644.0
34                    3  10248.246364  4459.616477   2_12260.0
35                    3   9912.993158  6399.136517   2_47664.0
37                    1  12431.855556  6200.503734   2_36740.0
38                    3   9202.150714  6147.116359   2_41180.0
39                    3   4510.542400   937.357195   0_19740.0
41                    1   5083.902885  1316.294008   0_35084.0
42                    3  10645.060909  6336.270634   2_38060.0
43                    1  13018.604444  7266.981078   2_33460.0
44                    2  11156.853333  5615.788611   2_34980.0
45                    1   9447.340000  4973.365388   2_13820.0
48                    1   4608.896731   994.266088   0_33124.0
51                    1   3928.630000  1028.710978   0_38300.0
55                    3  13571.593000  7452.955265   2_35614.0
57                    1   4115.696591   882.287180   0_41700.0
60                    2   4120.448095   850.395665   0_47260.0
63                    3  14407.149000  9502.978089   2_35004.0
64                    1   4150.861786  1521.238527   0_28140.0
66                    3  10531.104444  6299.335089   2_16974.0
70                    3  12463.995385  8641.369631   2_31084.0
73                    3  10033.610526  5835.730521   2_23104.0
74                    1   3933.971000  1151.618145   0_41180.0
78                    3  10050.917619  6160.295402   2_26420.0
81                    1   4384.711389   914.148235   0_39580.0
88                    1   4305.197391   970.036545   0_40060.0
89                    1   4275.334444   990.572715   0_18140.0
91                    1   4353.415750  1172.377285   0_45780.0
94                    3  10312.939474  5982.012846   2_47894.0
95                    1   3657.548250  1105.616764   0_13820.0
97                    3   4054.546607  1204.613751   0_16740.0
101                   1   3965.257258  1275.765891   0_23104.0
```

[42]:
```
output_df[(output_df["no_missing_mcare"] != 0)]["no_missing_mcare"].sum()
```

[42]:
```
92
```

0.0.16 looking at distributions of mcare_pay_median at a cluster X site level

[43]:
```
cluster_site_df = data
cluster_site_df["cluster_site"] = [str(i) + j + z for i,j,z in␣
 ↪zip(list(data["clusters"].values),["_"]*data.shape[0],data["site"].values)]
```

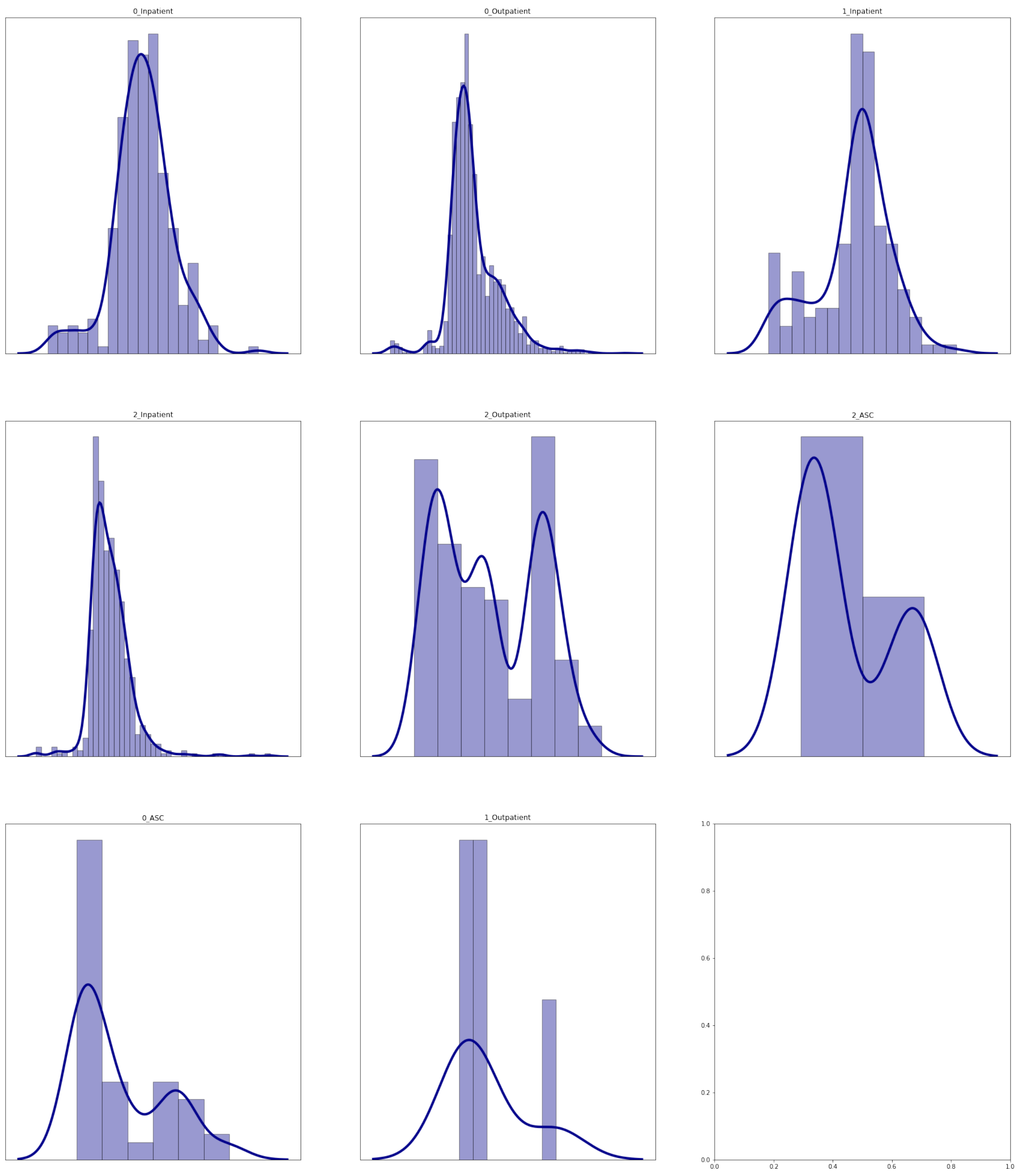
[44]:
```
cluster_site_df
```

[44]:
```
                                            id      msa  year  \
62502   26420.0_2018_Inpatient_breast reconstruction  26420.0  2018
62503  28660.0_2018_Outpatient_breast reconstruction  28660.0  2018
62504  37340.0_2018_Outpatient_breast reconstruction  37340.0  2018
62505  26620.0_2019_Outpatient_breast reconstruction  26620.0  2019
62506  15980.0_2020_Outpatient_breast reconstruction  15980.0  2020
…                                              …        …     …
66716                    0.0_2018_Outpatient_hysterect      0.0  2018
66717                nan_2018_Outpatient_rtc_slap_bank      NaN  2018
66718                0.0_2020_Outpatient_rtc_slap_bank      0.0  2020
66719                0.0_2019_Outpatient_rtc_slap_bank      0.0  2019
66720                0.0_2018_Outpatient_rtc_slap_bank      0.0  2018

             site                 group  priv_count  priv_pay_mean  \
62502   Inpatient  breast reconstruction        51.0   47327.576470
62503  Outpatient  breast reconstruction        51.0    8048.568431
62504  Outpatient  breast reconstruction        51.0   34211.443140
62505  Outpatient  breast reconstruction        51.0    6233.098627
62506  Outpatient  breast reconstruction        51.0   21964.297060
…             …                     …          …            …
66716  Outpatient             hysterect      6335.0   14677.391800
66717  Outpatient          rtc_slap_bank      6910.0   14537.588820
66718  Outpatient          rtc_slap_bank      7071.0   14769.380650
66719  Outpatient          rtc_slap_bank      8154.0   14388.208800
66720  Outpatient          rtc_slap_bank      9895.0   13873.629140

       priv_pay_median  priv_pay_iqr  mcare_count  mcare_los  mcare_pay_mean  \
62502         21408.00    23351.7150         81.0    3.54321    14027.076170
62503          9495.08     6468.6300         72.0    0.00000     4325.565833
62504         17213.58    24034.8250         68.0    0.00000     5490.479412
62505          5643.14     3529.6100         70.0    0.00000     5081.383000
62506         11824.70    17573.6400         71.0    0.00000     6320.088451
…                  …            …          …          …              …
66716         12040.63    11607.0600       2732.0    0.00000     5067.601204
66717         12213.92    10731.0675      10765.0    0.00000     5460.250379
66718         12657.14    12031.9000      17493.0    0.00000     4579.359115
66719         11857.93    11813.4150      20552.0    0.00000     4243.786995
66720         11897.50    11273.3400      21120.0    0.00000     4124.830504
```

```
       mcare_pay_median  mcare_pay_sd                            CBSA_NAME  \
62502          10395.16  12242.654150  Houston-The Woodlands-Sugar Land, TX
62503           3609.09   2791.209179                   Killeen-Temple, TX
62504           4130.65   3880.597914     Palm Bay-Melbourne-Titusville, FL
62505           3947.27   2657.779225                      Huntsville, AL
62506           4640.27   4468.429407             Cape Coral-Fort Myers, FL
…                   …           …                                  …
66716           5491.66   1787.807187                                 NaN
66717           4843.63   4561.683011                                 NaN
66718           4554.80   1716.745885                                 NaN
66719           4259.62   1626.198810                                 NaN
66720           4166.52   1441.987219                                 NaN

         State        lon        lat  clusters  cluster_site
62502    Texas -95.622552  29.598443         0   0_Inpatient
62503    Texas -97.698227  31.124151         0  0_Outpatient
62504  Florida -80.721442  28.263933         0  0_Outpatient
62505  Alabama -86.586104  34.730369         0  0_Outpatient
62506  Florida -81.872308  26.640628         0  0_Outpatient
…          …         …         …       …           …
66716      NaN        NaN        NaN         0  0_Outpatient
66717      NaN        NaN        NaN         0  0_Outpatient
66718      NaN        NaN        NaN         0  0_Outpatient
66719      NaN        NaN        NaN         0  0_Outpatient
66720      NaN        NaN        NaN         0  0_Outpatient

[4219 rows x 20 columns]
```

[45]:
```
uniq_cluster_site = cluster_site_df["cluster_site"].unique()
grid = mt.ceil(np.sqrt(len(uniq_cluster_site)))
```

[46]:
```
fig, axs = plt.subplots(grid, grid, figsize=(30,35))
n_pts = []
missing_pts = []
means = []
sd_list = []
labels = []
for i,x in enumerate(uniq_cluster_site):

    dt_pts = cluster_site_df[(cluster_site_df["cluster_site"] ==␣
 ↪x)]["mcare_pay_median"]
    Q1 = np.percentile(dt_pts[~np.isnan(dt_pts)], 25, interpolation =␣
 ↪'midpoint')
    Q3 = np.percentile(dt_pts[~np.isnan(dt_pts)], 75, interpolation =␣
 ↪'midpoint')
    IQR = Q3 - Q1
    if len(dt_pts) == 0:
        opt_bins = 10
    else:
        bin_width = 2*IQR/np.cbrt(len(dt_pts))
        rng = max(dt_pts[~np.isnan(dt_pts)]) - min(dt_pts[~np.isnan(dt_pts)])
        opt_bins = int(rng/bin_width)

    n_pts.append(len(dt_pts[~np.isnan(dt_pts)]))
    missing_pts.append(len(dt_pts[np.isnan(dt_pts)]))
    gp_stats = dt_pts.describe()
    mu = gp_stats[1]
    std = gp_stats[2]
    means.append(mu)
    sd_list.append(std)
    labels.append(x)
    sns.distplot(dt_pts, hist=True, kde=True,
             bins=opt_bins, color = 'darkblue',
             hist_kws={'edgecolor':'black'},
             kde_kws={'linewidth': 4},
                ax=axs[int(i/grid),i%grid])
#     dt_pts = np.msort(dt_pts)
#     intervals = UniDip(dt_pts[~np.isnan(dt_pts)]).run()
#     interval_list.append(intervals)
    axs[int(i/grid),i%grid].get_yaxis().set_visible(False)
    axs[int(i/grid),i%grid].get_xaxis().set_visible(False)
    axs[int(i/grid),i%grid].set_title(str(x))
# plt.savefig("density_plots.png")
#     axs[int(i/grid),i%grid].set_title(intervals[0][1])
```



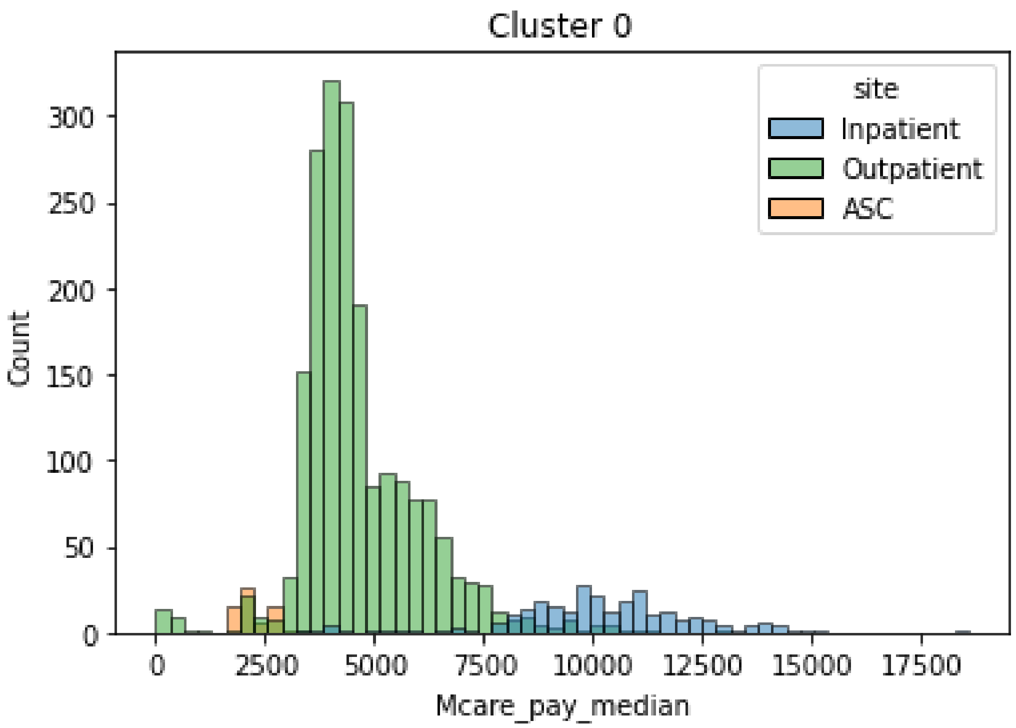
```
[47]: output_df = pd.
       ↪DataFrame(list(zip(labels,n_pts,missing_pts,means,sd_list)),columns=["index","no_pts","no_m
      # output_df["msa_cluster"] = output_df["Cluster_Label"].astype(str) + "_" +␣
       ↪output_df["MSA"].astype(str)
      # print("%age of MSAxCluster groups where we cannot reject the NULL hypothesis␣
       ↪is (90% CI) :",100*len(output_df[output_df["Normality_p_value"] >= 0.1])/
       ↪len(output_df))
      # output_df.sort_values(by=["Normality_p_value"],ascending=False)
```

```
output_df
```

[47]:
```
         index  no_pts  no_missing_mcare          mean           SD
0  0_Inpatient     277                 0  10171.440957  2169.787843
1 0_Outpatient    1975                38   4629.089180  1410.055609
2  1_Inpatient     157                 0  26819.030096  8108.833445
3  2_Inpatient     588                 1  13401.810723  3762.698828
4 2_Outpatient     650                77  10197.057185  6684.118193
5        2_ASC      30                47   4902.140878  2391.621123
6        0_ASC      67               307   2228.008524   401.101239
7 1_Outpatient       5                 0  11461.455000  5021.924998
```

[48]:
```
fig = sns.histplot(data=cluster_site_df[cluster_site_df["clusters"] == 0],␣
 ↪x="mcare_pay_median", hue="site",palette = ['tab:blue', 'tab:green', 'tab:
 ↪orange'])
fig.set(xlabel ="Mcare_pay_median", ylabel = "Count", title ='Cluster 0')
```

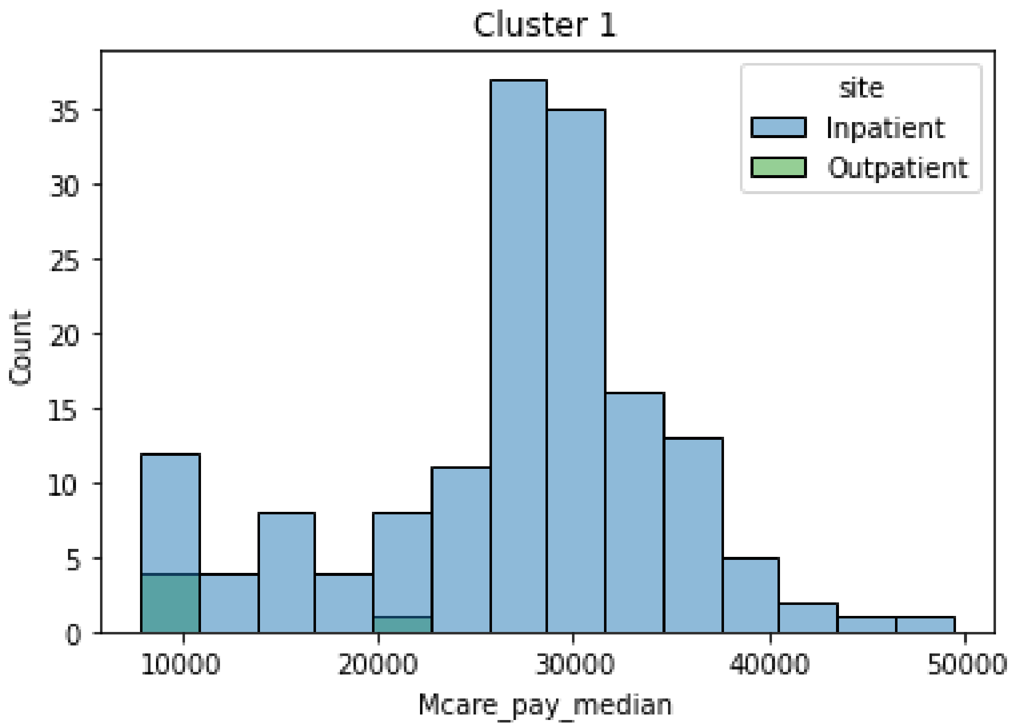
[48]:
```
[Text(0.5, 0, 'Mcare_pay_median'),
 Text(0, 0.5, 'Count'),
 Text(0.5, 1.0, 'Cluster 0')]
```



[49]:
```
fig = sns.histplot(data=cluster_site_df[cluster_site_df["clusters"] == 1],␣
 ↪x="mcare_pay_median", hue="site",palette = ['tab:blue', 'tab:green'])
```

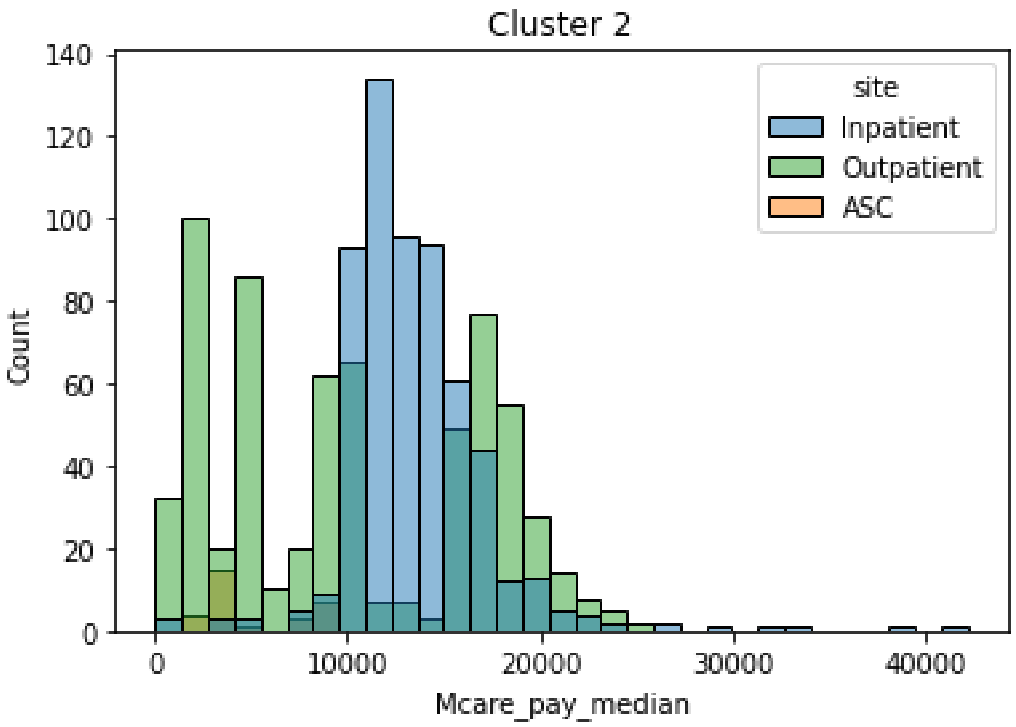
```
fig.set(xlabel ="Mcare_pay_median", ylabel = "Count", title ='Cluster 1')
```

```
[49]: [Text(0.5, 0, 'Mcare_pay_median'),
       Text(0, 0.5, 'Count'),
       Text(0.5, 1.0, 'Cluster 1')]
```



```
[50]: fig = sns.histplot(data=cluster_site_df[cluster_site_df["clusters"] == 2],␣
       ↪x="mcare_pay_median", hue="site",palette = ['tab:blue', 'tab:green', 'tab:
       ↪orange'])
      fig.set(xlabel ="Mcare_pay_median", ylabel = "Count", title ='Cluster 2')
```

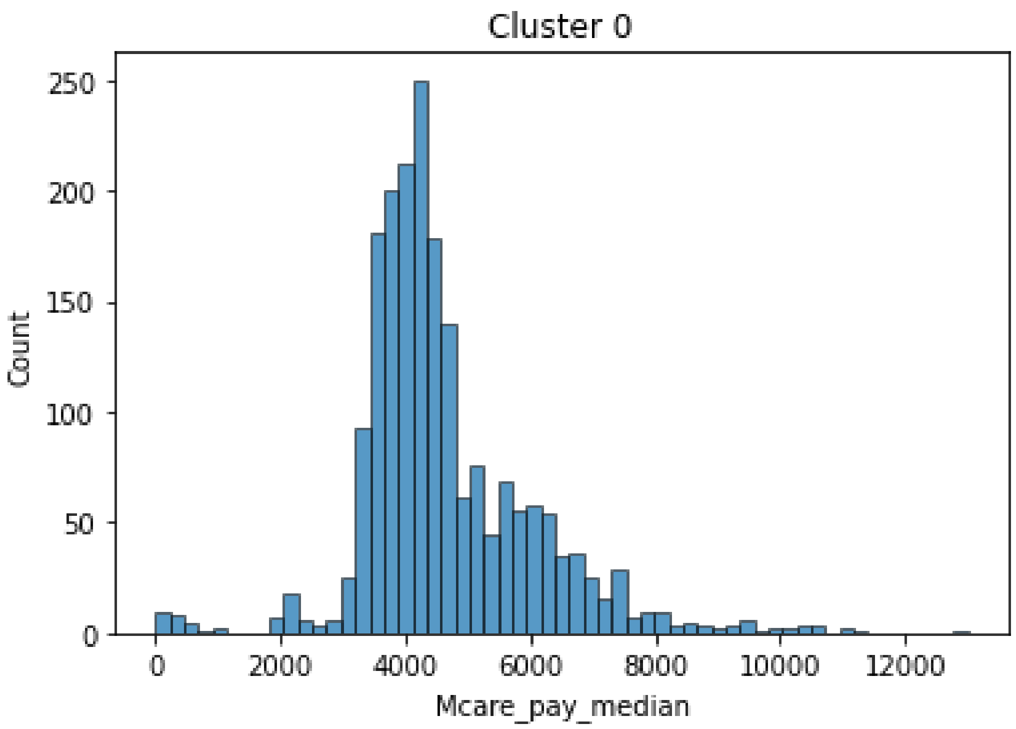
```
[50]: [Text(0.5, 0, 'Mcare_pay_median'),
       Text(0, 0.5, 'Count'),
       Text(0.5, 1.0, 'Cluster 2')]
```



0.0.17 checking for normality in the outpatient category in cluster 0

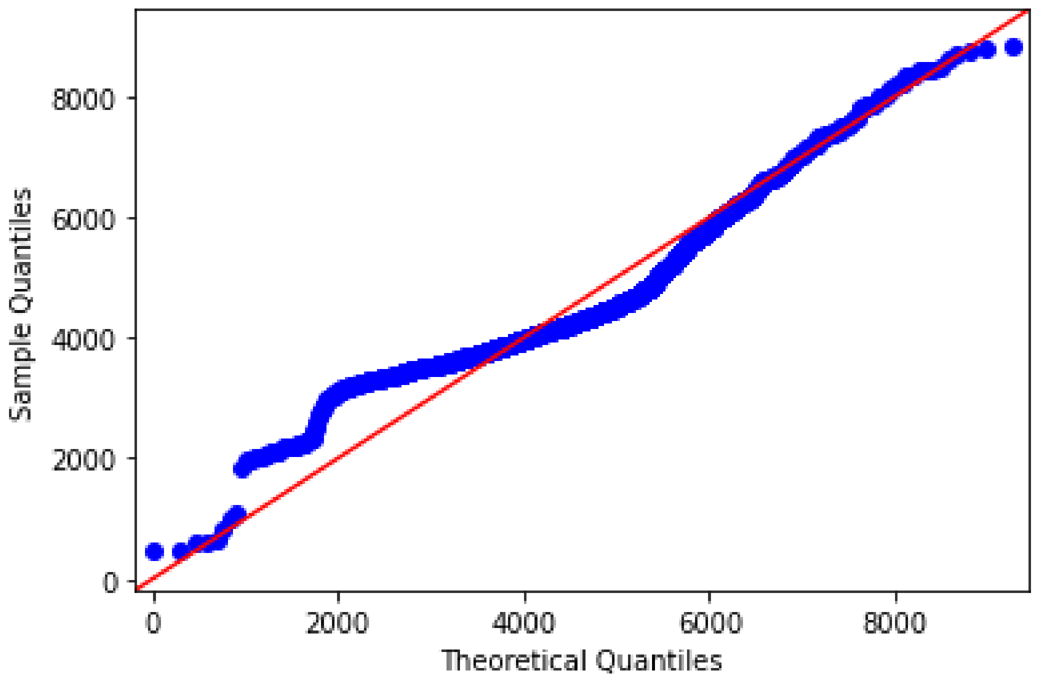
```
[51]: fig = sns.histplot(data=cluster_site_df[(cluster_site_df["clusters"] == 0) &␣
       ↪(cluster_site_df["site"] == "Outpatient")], x="mcare_pay_median",palette =␣
       ↪['tab:blue', 'tab:green', 'tab:orange'])
      fig.set(xlabel ="Mcare_pay_median", ylabel = "Count", title ='Cluster 0')
```

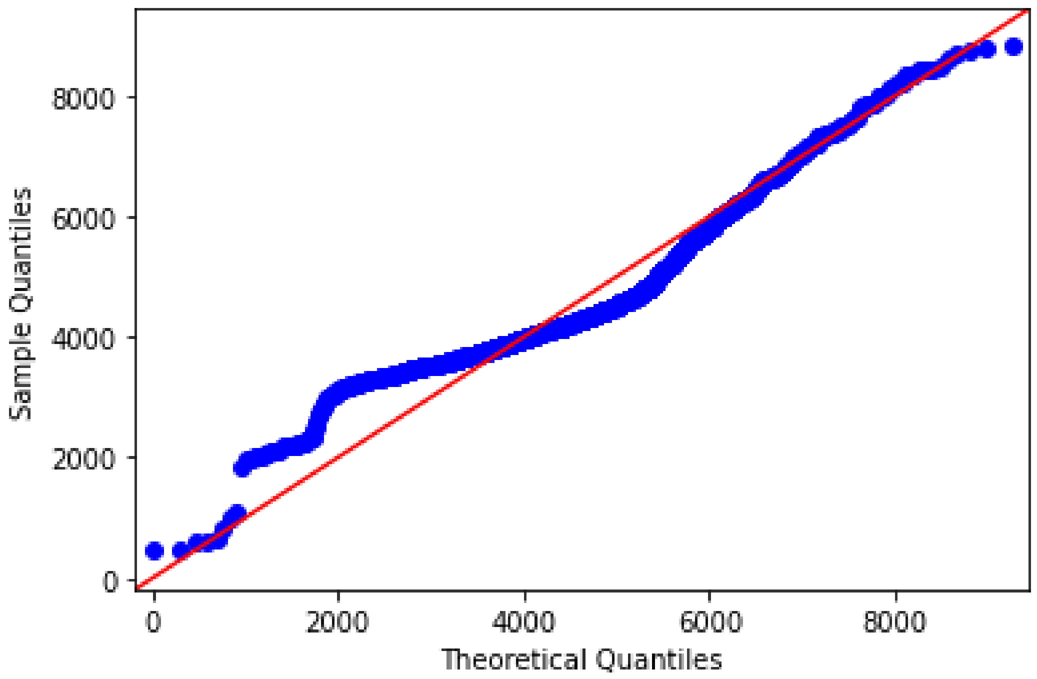
```
[51]: [Text(0.5, 0, 'Mcare_pay_median'),
       Text(0, 0.5, 'Count'),
       Text(0.5, 1.0, 'Cluster 0')]
```



[52]:
```
out_0 = cluster_site_df[(cluster_site_df["clusters"] == 0) &␣
 ↪(cluster_site_df["site"] == "Outpatient")]["mcare_pay_median"]
gp_stats = out_0.describe()
mu = gp_stats[1]
std = gp_stats[2]
rmv_outliers = out_0[(out_0 > mu - 3*std) & (out_0 < mu + 3*std)].values
sm.qqplot(rmv_outliers,dist=norm(mu,std), line ='45')
```

[52]:


Sample Quantiles
Theoretical Quantiles
0
2000
4000
6000
8000


Sample Quantiles
Theoretical Quantiles
0
2000
4000
6000
8000