Part-B

December 16, 2019

In [40]:
```
import numpy as np
import pandas as pd
from datetime import datetime
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

%matplotlib inline
```

In [41]:
```
# Reading data from input file
df = pd.read_csv("Appointment-No-Show-Data.csv")
print(df.dtypes)
print("----------------------------------------")
df.shape
```

```
PatientId         float64
AppointmentID       int64
Gender             object
ScheduledDay       object
AppointmentDay     object
Age                 int64
Neighbourhood      object
Scholarship         int64
Hipertension        int64
Diabetes            int64
Alcoholism          int64
Handcap             int64
SMS_received        int64
No-show            object
dtype: object
----------------------------------------
```

Out[41]: (110527, 14)

In [42]:
```
# Question 1

print('Total number of unique patients = {}.'.format(len(df.groupby(['PatientId']).cou
```

```
Total number of unique patients = 62299.
```

In [43]:
```
# Question 2 - Adding categorical binning to age column

bins = [0, 17, 36, 64, 200]
bin_name = ["Kids","Young Adult","Older Adult","Elderly"]
df['binned_age'] = pd.cut(df['Age'], bins,labels=bin_name)
df.head()
```

Out[43]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show	binned_age
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	1	0	0	0	0	No	Older Adult
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	0	0	0	0	0	No	Older Adult
2	4.262962e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	0	0	0	0	0	0	No	Older Adult
3	8.679512e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	0	0	0	0	0	0	No	Kids
4	8.841186e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	1	1	0	0	0	No	Older Adult

In [44]:
```
# Question 3 - Percentage of patients with more than 1 appointment
grouped = df.groupby(['PatientId'])
print("Total number of people with multiple appointments:")
print(sum(grouped.count()["AppointmentID"].apply(lambda x: 1 if x > 1 else 0)))

print("Percentage of people with multiple appointments:")
print(int(sum(grouped.count()["AppointmentID"].apply(lambda x: 1 if x > 1 else 0))/le
```

```
Total number of people with multiple appointments:
24379
Percentage of people with multiple appointments:
39
```

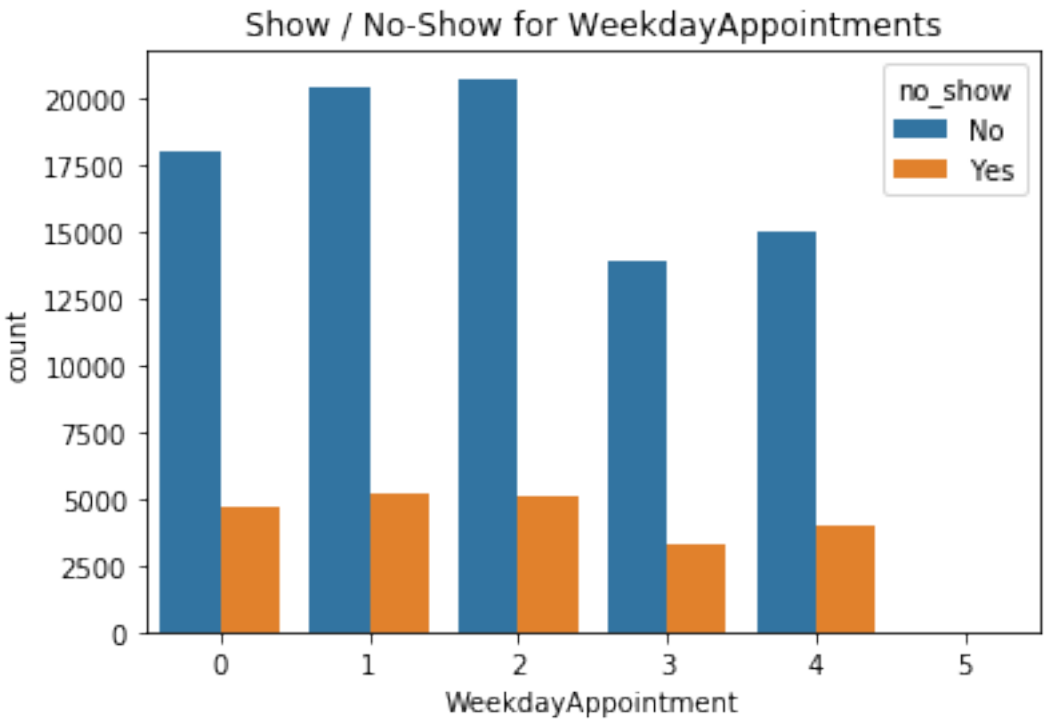
In [45]:
```
# Question 4 - Percentage of patients with more than 1 appointment and went to all ap
a = df.groupby(['PatientId','No-show'])

key_list = []
```

```
        for k, gp in a:
            key_list.append(k)

        counter = 0;
        for k,gp in a:
            if(k[1] == 'Yes' and (k[0],'No') in key_list):
                continue
            elif(k[1] == 'No' and (k[0],'Yes') in key_list):
                continue
            elif(len(gp) > 1):
                counter = counter + 1
            else:
                continue

        print("Percentage of people with multiple appointments and Never missed any:")
        print(int(counter/len(grouped.count())*100))
```

```
Percentage of people with multiple appointments and Never missed any:
23
```

```
In [46]: # Converting the data type to date time
         df.rename(columns = {'No-show':'no_show'}, inplace = True)
         df["ScheduledDay"] = pd.to_datetime(df["ScheduledDay"])
         df["AppointmentDay"] = pd.to_datetime(df["AppointmentDay"])

         # Weekday function identifies the day of the week. 0-6 corresponds to Monday-Sunday

         df["WeekdayScheduled"] = df["ScheduledDay"].dt.weekday
         df["WeekdayAppointment"] = df["AppointmentDay"].dt.weekday
```

```
In [47]: Weekday_attended = df.groupby(["WeekdayAppointment"]).describe()["Age"]["count"]
         b = df.groupby(["WeekdayAppointment",'no_show']).count()
         b

         ax = sns.countplot(x=df.WeekdayAppointment, hue=df.no_show, data=df)
         ax.set_title("Show / No-Show for WeekdayAppointments")
         x_ticks_labels=['Monday', 'T','W','Th','Fri','Sat','Sun']
         plt.show();

         print("It is clear that most people miss their treatments on Weekday 1 - Tuesday")
```



It is clear that most people miss their treatments on Weekday 1 - Tuesday

```
In [48]: df = df[df["Age"] <100]
         df = df[df["Age"] > -1]

         df.head()
```

Out[48]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	\
0	2.987250e+13	5642903	F	2016-04-29 18:38:08	2016-04-29	62	
1	5.589978e+14	5642503	M	2016-04-29 16:08:27	2016-04-29	56	
2	4.262962e+12	5642549	F	2016-04-29 16:19:04	2016-04-29	62	
3	8.679512e+11	5642828	F	2016-04-29 17:29:31	2016-04-29	8	
4	8.841186e+12	5642494	F	2016-04-29 16:07:23	2016-04-29	56	

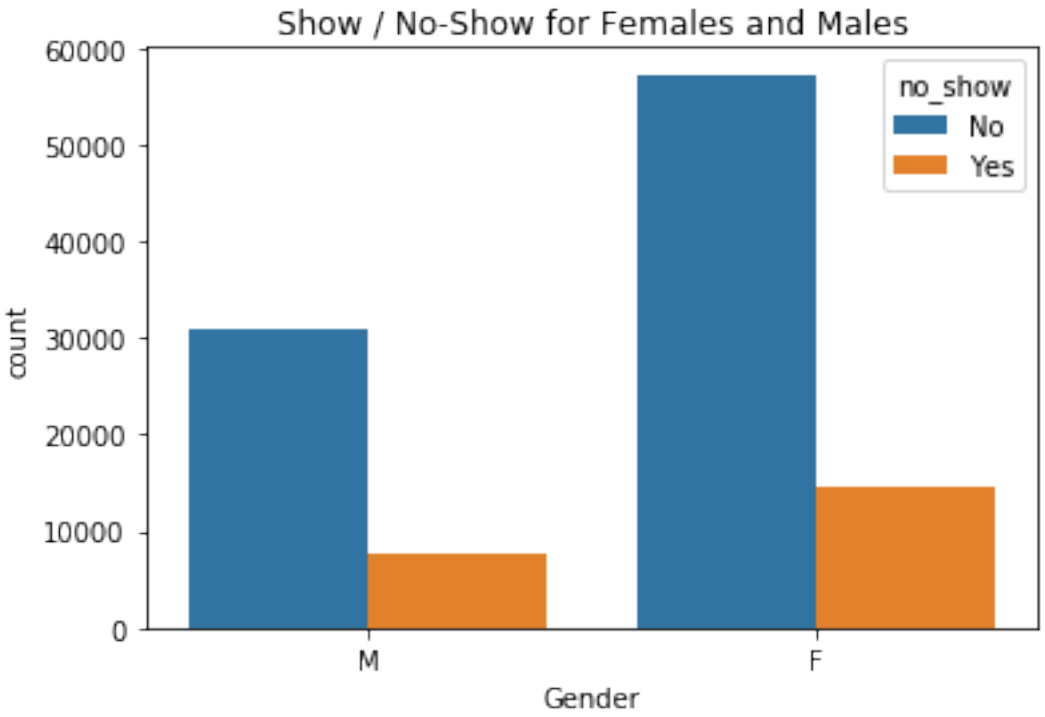
	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	\
0	JARDIM DA PENHA	0	1	0	0	
1	JARDIM DA PENHA	0	0	0	0	
2	MATA DA PRAIA	0	0	0	0	
3	PONTAL DE CAMBURI	0	0	0	0	
4	JARDIM DA PENHA	0	1	1	0	

	Handcap	SMS_received	no_show	binned_age	WeekdayScheduled	\
0	0	0	No	Older Adult	4	

```
1          0              0      No  Older Adult                 4
2          0              0      No  Older Adult                 4
3          0              0      No         Kids                 4
4          0              0      No  Older Adult                 4

   WeekdayAppointment
0                   4
1                   4
2                   4
3                   4
4                   4
```

In [49]:
```
# Question 6 - Creation of NoShowHistory column
df = df.sort_values(by = ['AppointmentDay', 'ScheduledDay'], axis = 0)
df['NoShowHistory'] = (df.groupby('PatientId')['no_show'].apply(lambda x : x.shift(1)
```

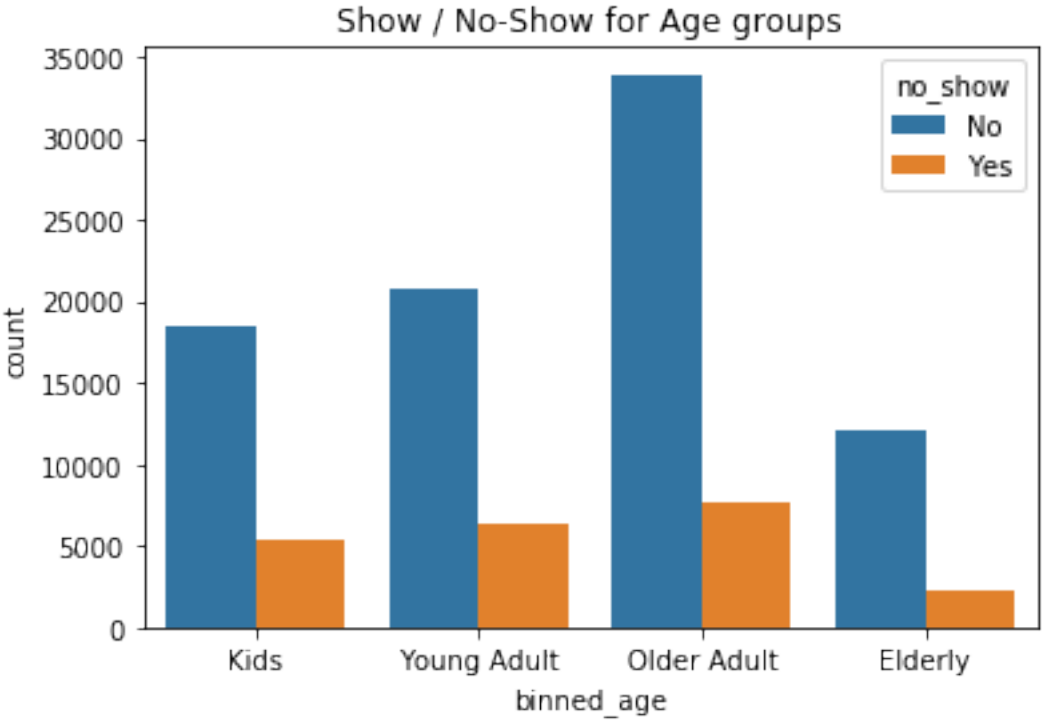
In [50]:
```
# Question 7 - Gender

all_appointments_by_f = len(df.loc[df['Gender'] == "F"])
all_appointments_by_m = len(df.loc[df['Gender'] == "M"])

missed_appointments_by_f = len(df.query('no_show == "Yes" and Gender == "F"'))
missed_appointments_by_m = len(df.loc[(df['Gender'] == "M") & (df['no_show'] == "Yes"

missed_ratio_f = int(round(missed_appointments_by_f/all_appointments_by_f*100))
missed_ratio_m = int(round(missed_appointments_by_m/all_appointments_by_m*100))

ax = sns.countplot(x=df.Gender, hue=df.no_show, data=df)
ax.set_title("Show / No-Show for Females and Males")
x_ticks_labels=['Female', 'Male']
plt.show();

print('Out of {} appointments made by females, {} were missed with the ratio of {}%.'
print('Out of {} appointments made by males, {} were missed with the ratio of {}%.'.fo
```




```
Out of 71830 appointments made by females, 14591 were missed with the ratio of 20%.
Out of 38685 appointments made by males, 7725 were missed with the ratio of 20%.
```

```
In [51]: # Question 7 - Age group
         all_appointments_by_kids = len(df.loc[df['binned_age'] == "Kids"])
         all_appointments_by_young_adults = len(df.loc[df['binned_age'] == "Young Adult"])
         all_appointments_by_older_adults = len(df.loc[df['binned_age'] == "Older Adult"])
         all_appointments_by_elderly = len(df.loc[df['binned_age'] == "Elderly"])

         missed_appointments_by_kids = len(df.query('no_show == "Yes" and binned_age == "Kids"
         missed_appointments_by_young_adults = len(df.query('no_show == "Yes" and binned_age ==
         missed_appointments_by_older_adults = len(df.query('no_show == "Yes" and binned_age ==
         missed_appointments_by_elderly = len(df.query('no_show == "Yes" and binned_age == "El

         missed_ratio_kids = int(round(missed_appointments_by_kids/all_appointments_by_kids*100
         missed_ratio_young_adults = int(round(missed_appointments_by_young_adults/all_appointn
         missed_ratio_old_adults = int(round(missed_appointments_by_older_adults/all_appointmen
         missed_ratio_elderly = int(round(missed_appointments_by_elderly/all_appointments_by_el

         ax = sns.countplot(x=df.binned_age, hue=df.no_show, data=df)
         ax.set_title("Show / No-Show for Age groups")
         x_ticks_labels=['Kids', 'Young Adults','Older Adults','Elderly']
         plt.show();
```

```
print('Out of {} appointments made by kids, {} were missed with the ratio of {}%.'.fo
print('Out of {} appointments made by young adults, {} were missed with the ratio of
print('Out of {} appointments made by older adults, {} were missed with the ratio of
print('Out of {} appointments made by elderly, {} were missed with the ratio of {}%.'
```




```
Out of 23840 appointments made by kids, 5358 were missed with the ratio of 22%.
Out of 27204 appointments made by young adults, 6447 were missed with the ratio of 24%.
Out of 41541 appointments made by older adults, 7642 were missed with the ratio of 18%.
Out of 14391 appointments made by elderly, 2230 were missed with the ratio of 15%.
```

In [52]:
```
# Question 8 - AwaitingTime
df['ScheduledDay'] = pd.to_datetime(df['ScheduledDay']).dt.date.astype('datetime64[ns]
df['awaiting_time_days'] = (df.AppointmentDay - df.ScheduledDay).dt.days
df = df[(df.awaiting_time_days >= 0)]
```

In [53]:
```
bins = [-1,0, 4, 15, 200]
bin_name = ["SameDay","OneDay","TenDays","Forever"]
df['binned_awaitingdays'] = pd.cut(df['awaiting_time_days'], bins,labels=bin_name)
df.head(5)
```

Out[53]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	\
954	1.423329e+12	5217179	M	2016-01-05	2016-04-29	84	

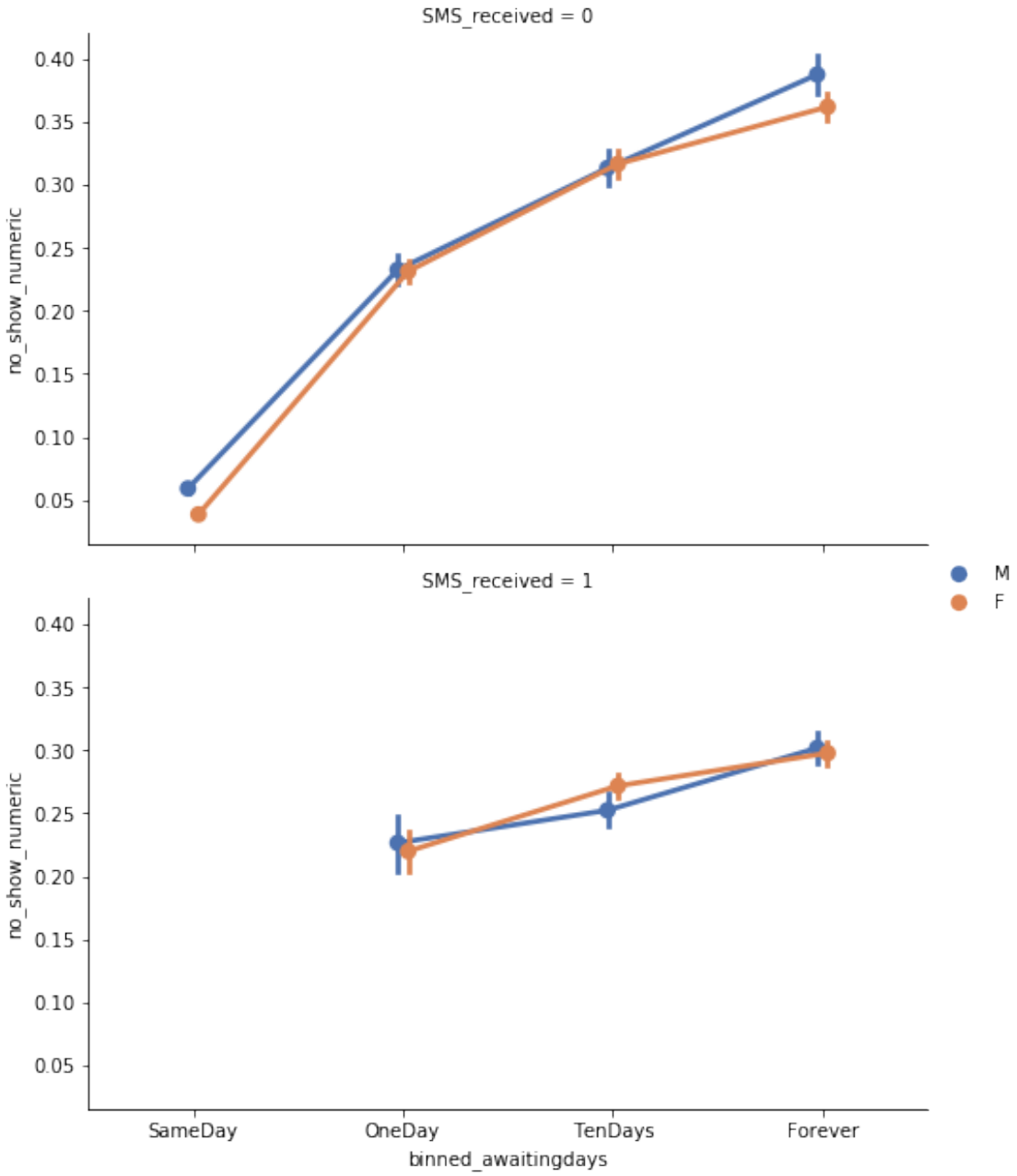
```
953  4.616858e+12        5218520      F   2016-01-05     2016-04-29   83
959  5.558963e+13        5235449      F   2016-01-11     2016-04-29   74
957  9.189694e+13        5235643      F   2016-01-11     2016-04-29   70
958  1.534482e+12        5235655      F   2016-01-11     2016-04-29   87

    Neighbourhood  Scholarship  Hipertension  Diabetes  Alcoholism  Handcap  \
954   SANTO ANDRÉ            0             1         1           0        1
953      REDENÇÃO            0             1         0           0        0
959    MONTE BELO            0             0         0           0        0
957      GURIGICA            0             1         1           0        0
958   JUCUTUQUARA            0             0         0           0        0

     SMS_received no_show binned_age  WeekdayScheduled  WeekdayAppointment  \
954             1      No    Elderly                 1                   4
953             1      No    Elderly                 1                   4
959             1      No    Elderly                 0                   4
957             1      No    Elderly                 0                   4
958             1      No    Elderly                 0                   4

    NoShowHistory  awaiting_time_days binned_awaitingdays
954           NaN                 115             Forever
953           NaN                 115             Forever
959           NaN                 109             Forever
957           NaN                 109             Forever
958           NaN                 109             Forever
```

In [54]:
```
# Question 9 - Same day appointments and show-up
awaiting = len(df[(df.awaiting_time_days == 0)])

awaiting_not_showed_up = len(df.query('awaiting_time_days  == 0 and no_show == "Yes"'))
awaiting_not_showed_up_ratio = int(round(awaiting_not_showed_up/awaiting*100))

print('Out of all patients scheduling an appointment for the same day (in total {}), 
```

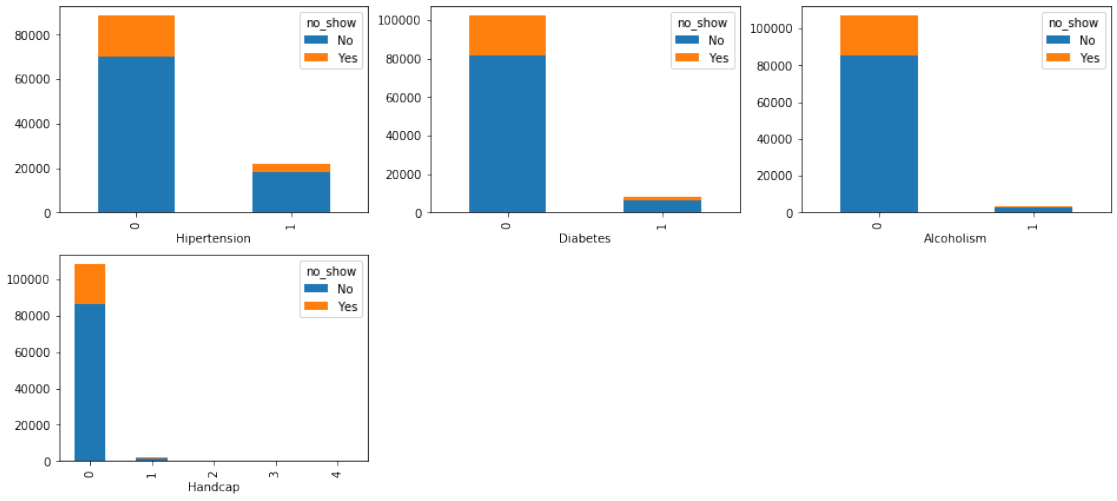
```
Out of all patients scheduling an appointment for the same day (in total 38560), 1792 of patie
```

In [55]:
```
# Question 10
df['no_show_numeric'] = np.where(df['no_show']=='Yes', 1, 0)
grid = sns.FacetGrid(df, row='SMS_received', height=4.4, aspect=1.6)
grid.map(sns.pointplot, 'binned_awaitingdays', 'no_show_numeric', 'Gender', palette='
grid.add_legend();

print("The plot shows that if sms is recieved, people tend to show up compared to tho
```

```
The plot shows that if sms is recieved, people tend to show up compared to those who didn't re
```



In [56]:
```
# Question 11
# Correlation plots for various factors

categorical_vars = ['Hipertension', 'Diabetes', 'Alcoholism', 'Handcap']
fig = plt.figure(figsize=(16, 11))
for i, var in enumerate(categorical_vars):
    ax = fig.add_subplot(3, 3, i+1)
    df.groupby([var, 'no_show'])[var].count().unstack('no_show').plot(ax=ax, kind='ba
```




```
In [59]: def compare_by_column(df, column_name):

             comparison_df = pd.DataFrame()
             comparison_df['no_show'] = (
                 df[df.no_show == "Yes"][column_name].value_counts()
                 )
             comparison_df['show_up'] = (
                 df[df.no_show == "No"][column_name].value_counts()
                 )
             comparison_df['no_scholorship'] = (
                 df[df.Scholarship == 0][column_name].value_counts()
                 )
             comparison_df['scholarship'] = (
                 df[df.Scholarship == 1][column_name].value_counts()
                 )
             # In case some for column_name values, there isn't a single True or
             # a single False in no_show, they'll come up as NaN in the
             # comparison_df. We fill those with zeros.
             comparison_df = comparison_df.fillna(0)
             comparison_df['sample_size'] = comparison_df.no_show + comparison_df.show_up
             comparison_df['no_show_rate'] = (
                 comparison_df.no_show / (comparison_df.no_show + comparison_df.show_up)
                 )
             comparison_df['no_sc_rate'] = (
                 comparison_df.no_scholorship / (comparison_df.no_scholorship + comparison_df.
                 )

             comparison_df.sort_index(inplace = True)
             comparison_df.name = column_name
             overall_no_show_rate = (
```

```
            comparison_df.no_show.sum() / comparison_df.sample_size.sum()
            )
        return (comparison_df, overall_no_show_rate)

    # Create a comparison_df with the neighbourhood column
    neighbourhood_df = compare_by_column(df, 'Neighbourhood')[0]

    neighbourhood_df = (
        neighbourhood_df[neighbourhood_df.sample_size >= 100]
        .sort_values('no_show_rate',ascending=False)
        )

    print("SANTOS DUMONT is the neighbourhood with maximum no-show rate")

    neighbourhood_df.head(20)
```

```
SANTOS DUMONT is the neighbourhood with maximum no-show rate
```

Out[59]:

	no_show	show_up	no_scholorship	scholarship	\
SANTOS DUMONT	369	907.0	1041	235.0	
SANTA CECÍLIA	123	325.0	423	25.0	
SANTA CLARA	134	372.0	476	30.0	
ITARARÉ	923	2591.0	3203	311.0	
JESUS DE NAZARETH	696	2157.0	2583	270.0	
HORTO	42	133.0	169	6.0	
ILHA DO PRÍNCIPE	532	1734.0	1687	579.0	
CARATOÍRA	591	1974.0	2109	456.0	
ANDORINHAS	518	1740.0	1935	323.0	
PRAIA DO SUÁ	294	994.0	1137	151.0	
GURIGICA	456	1562.0	1596	422.0	
BENTO FERREIRA	193	665.0	835	23.0	
PARQUE MOSCOSO	179	623.0	792	10.0	
MARUÍPE	424	1477.0	1780	121.0	
DO MOSCOSO	92	321.0	302	111.0	
ENSEADA DO SUÁ	52	183.0	229	6.0	
ARIOVALDO FAVALESSA	62	220.0	230	52.0	
ILHA DAS CAIEIRAS	235	836.0	868	203.0	
FONTE GRANDE	149	533.0	596	86.0	
CRUZAMENTO	304	1094.0	1228	170.0	

	sample_size	no_show_rate	no_sc_rate
SANTOS DUMONT	1276.0	0.289185	0.815831
SANTA CECÍLIA	448.0	0.274554	0.944196
SANTA CLARA	506.0	0.264822	0.940711
ITARARÉ	3514.0	0.262664	0.911497
JESUS DE NAZARETH	2853.0	0.243954	0.905363
HORTO	175.0	0.240000	0.965714

ILHA DO PRÍNCIPE	2266.0	0.234775	0.744484
CARATOÍRA	2565.0	0.230409	0.822222
ANDORINHAS	2258.0	0.229407	0.856953
PRAIA DO SUÁ	1288.0	0.228261	0.882764
GURIGICA	2018.0	0.225966	0.790882
BENTO FERREIRA	858.0	0.224942	0.973193
PARQUE MOSCOSO	802.0	0.223192	0.987531
MARUÍPE	1901.0	0.223041	0.936349
DO MOSCOSO	413.0	0.222760	0.731235
ENSEADA DO SUÁ	235.0	0.221277	0.974468
ARIOVALDO FAVALESSA	282.0	0.219858	0.815603
ILHA DAS CAIEIRAS	1071.0	0.219421	0.810458
FONTE GRANDE	682.0	0.218475	0.873900
CRUZAMENTO	1398.0	0.217454	0.878398