saipranay190198 / My-personal-files Private

Code Issues Pull requests Actions Projects Security Insights

main

My-personal-files / TimeSeriesForecasting.ipynb

saipranay190198 Add files via upload History

1 contributor

707 lines (707 sloc) | 159 KB

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from datetime import datetime
from pandas import Series
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
train = pd.read_csv("Train_timeseries.csv")
test = pd.read_csv("Test_timeseries.csv")
```

In [3]:
```
train
```

Out[3]:

	ID	Datetime	Count
0	0	25-08-2012 00:00	8
1	1	25-08-2012 01:00	2
2	2	25-08-2012 02:00	6
3	3	25-08-2012 03:00	2
4	4	25-08-2012 04:00	2
...	...	...	...
18283	18283	25-09-2014 19:00	868
18284	18284	25-09-2014 20:00	732
18285	18285	25-09-2014 21:00	702
18286	18286	25-09-2014 22:00	580
18287	18287	25-09-2014 23:00	534

18288 rows × 3 columns

In [4]:
```
test
```

Out[4]:

	ID	Datetime
0	18288	26-09-2014 00:00
1	18289	26-09-2014 01:00
2	18290	26-09-2014 02:00
3	18291	26-09-2014 03:00
4	18292	26-09-2014 04:00
...	...	...
5107	23395	26-04-2015 19:00
5108	23396	26-04-2015 20:00

5109	23397	26-04-2015 21:00
5110	23398	26-04-2015 22:00
5111	23399	26-04-2015 23:00

5112 rows × 2 columns

In [5]:
```
train_original = train.copy()
test_original = test.copy()
```

In [6]:
```
train.columns
```

Out[6]:
```
Index(['ID', 'Datetime', 'Count'], dtype='object')
```

In [7]:
```
test.columns
```

Out[7]:
```
Index(['ID', 'Datetime'], dtype='object')
```

In [8]:
```
train.dtypes, test.dtypes
```

Out[8]:
```
(ID           int64
 Datetime    object
 Count        int64
 dtype: object,
 ID           int64
 Datetime    object
 dtype: object)
```

In [18]:
```
train['Datetime'] = pd.to_datetime(train.Datetime, format = None )
test['Datetime'] = pd.to_datetime(test.Datetime, format = None)
train_original['Datetime'] = pd.to_datetime(train_original.Datetime, form
test_original['Datetime'] = pd.to_datetime(test_original.Datetime, format
```

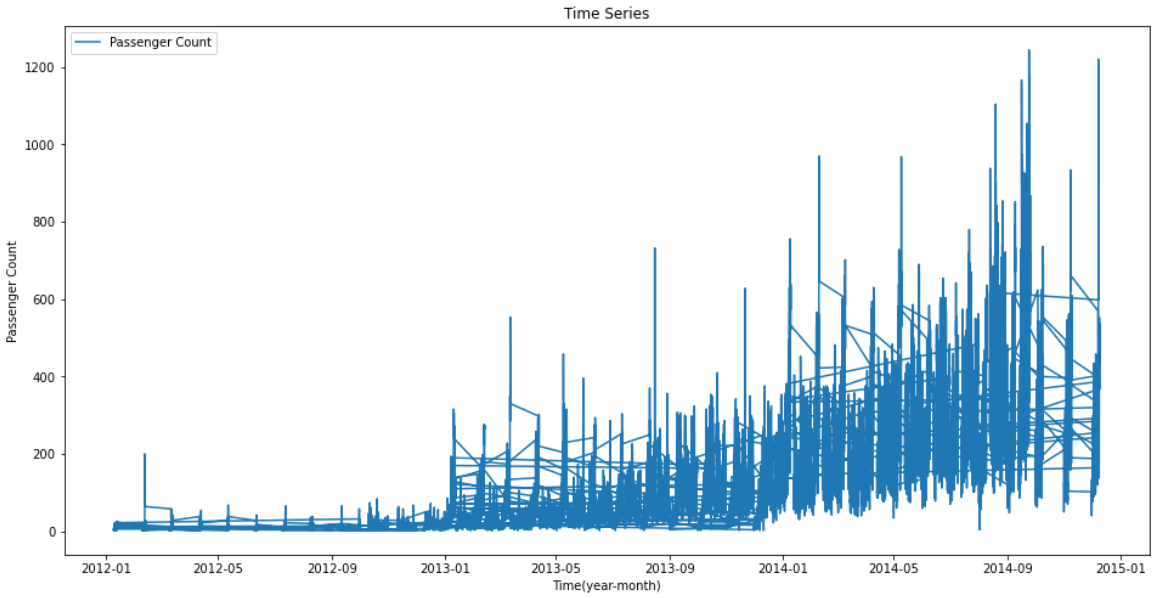
In [19]:
```
for i in (train, test, test_original,train_original):
    i['Year'] = i.Datetime.dt.year
    i['Month'] = i.Datetime.dt.month
    i['day'] = i.Datetime.dt.day
    i['Hour'] = i.Datetime.dt.hour
```

In [21]:
```
train['day of week'] = train['Datetime'].dt.dayofweek
temp = train['Datetime']
```

In [35]:
```
def applyer(row):
    if row.dayofweek == 5 or row.dayofweek == 6:
        return 1
    else:
        return 0
    temp2 = train['Datetime'].apply(applyer)
    train['weekend'] = temp2
```

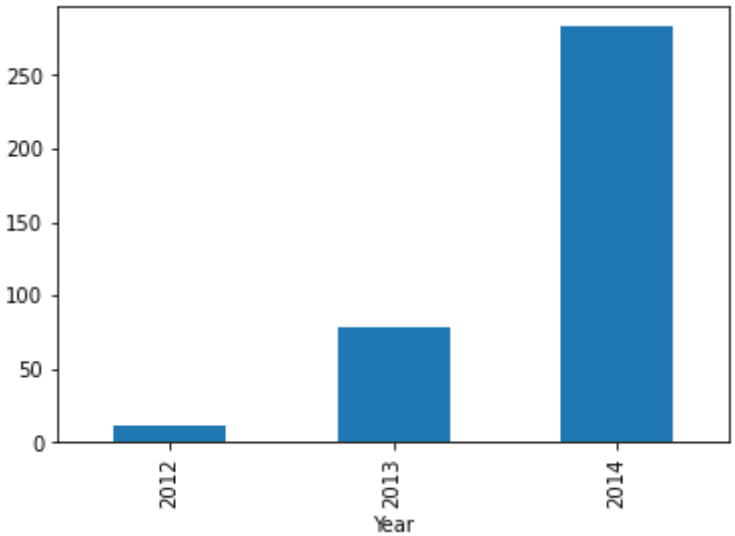
In [23]:
```
train.index = train['Datetime']

df = train.drop('ID', 1)
ts = df['Count']
plt.figure(figsize = (16,8))
plt.plot(ts, label = 'Passenger Count')
plt.title('Time Series')
plt.xlabel("Time(year-month)")
plt.ylabel("Passenger Count")
plt.legend(loc = 'best')
```

Out[23]:
```
<matplotlib.legend.Legend at 0x25b6cebaeb0>
```



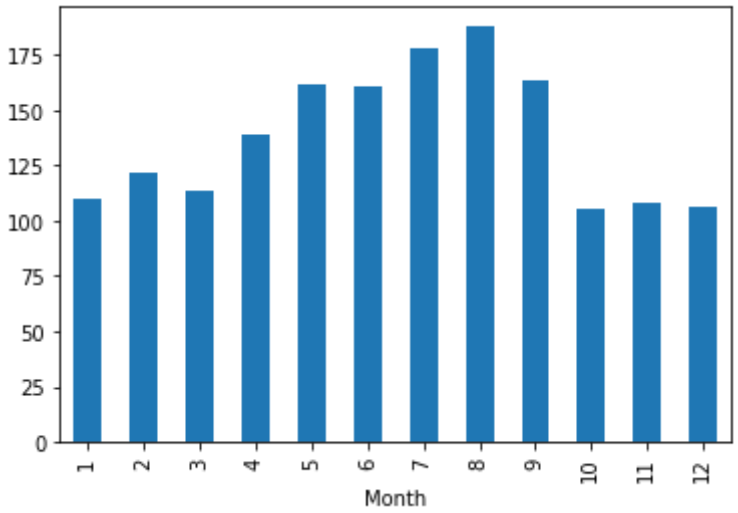
In [24]:
```
train.groupby('Year')['Count'].mean().plot.bar()
```

Out[24]:
```
<AxesSubplot:xlabel='Year'>
```



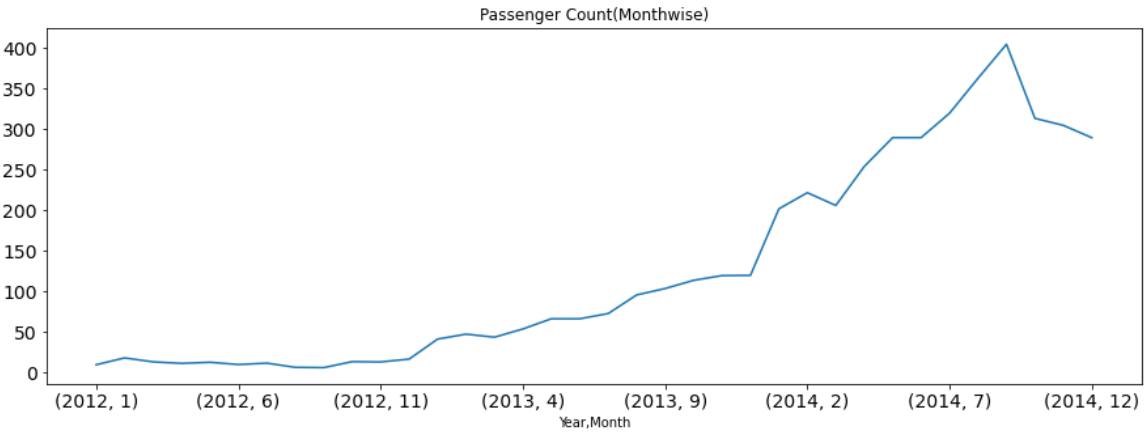
In [26]:
```
train.groupby('Month')['Count'].mean().plot.bar()
```

Out[26]:
```
<AxesSubplot:xlabel='Month'>
```



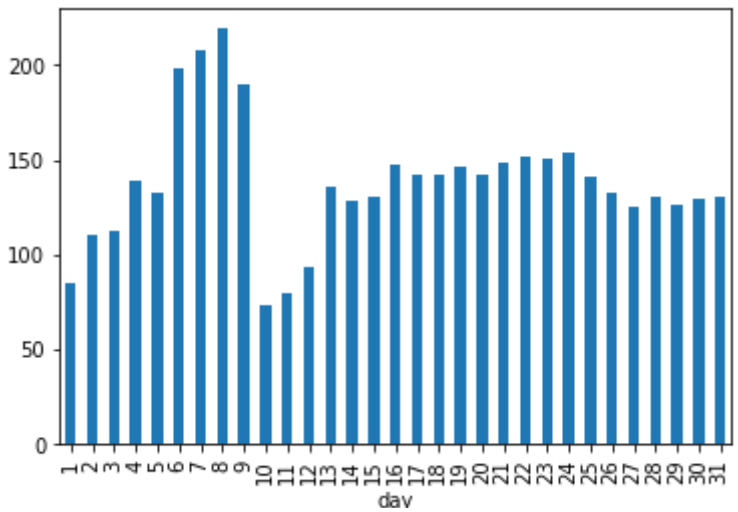
In [28]:
```
temp = train.groupby(['Year','Month'])['Count'].mean()
temp.plot(figsize = (15,5), title = 'Passenger Count(Monthwise)', fontsiz
```

Out[28]:
```
<AxesSubplot:title={'center':'Passenger Count(Monthwise)'}, xlabel='Year,
Month'>
```



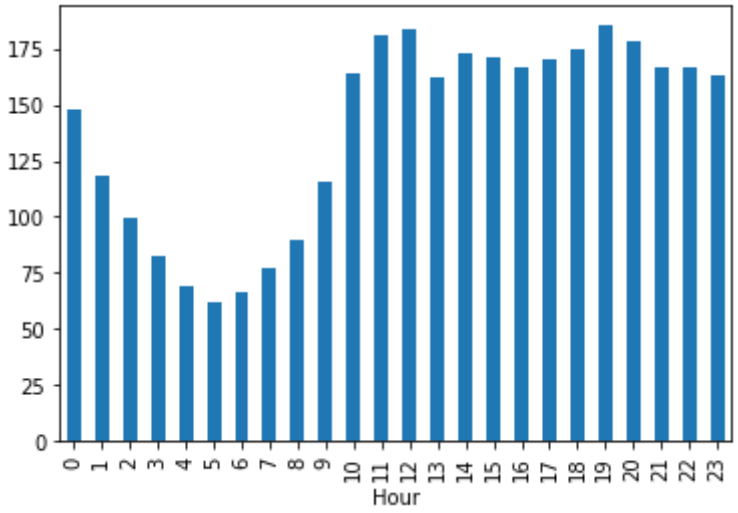
In [29]:
```
train.groupby('day')['Count'].mean().plot.bar()
```

Out[29]:
```
<AxesSubplot:xlabel='day'>
```



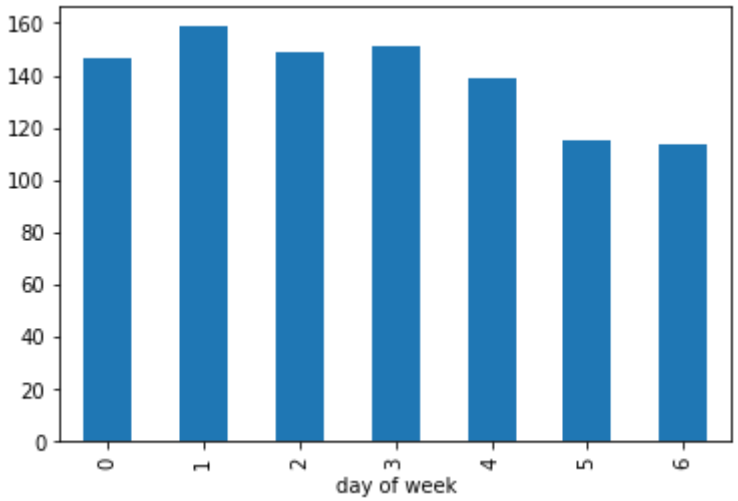
In [31]:
```
train.groupby('Hour')['Count'].mean().plot.bar()
```

Out[31]:
```
<AxesSubplot:xlabel='Hour'>
```



In [41]:
```
train.groupby('day of week')['Count'].mean().plot.bar()
```

Out[41]:
```
<AxesSubplot:xlabel='day of week'>
```



In [38]:
```
train.columns
```

Out[38]:
```
Index(['ID', 'Datetime', 'Count', 'Year', 'Month', 'day', 'Hour',
       'day of week'],
      dtype='object')
```

In [ ]:
```
```