*#import dependencies*  
**%matplotlib** inline  
**import** os

*#start python imports*  
**import** math**,** time**,** random**,** datetime

*#data manupilation*  
**import** numpy **as** np  
**import** pandas **as** pd

*#visualization*  
**import** matplotlib.pyplot **as** plt  
**import** missingno  
**import** seaborn **as** sns  
plt**.**style**.**use('seaborn-whitegrid')

*# preprocessing*  
**from** sklearn.preprocessing **import** OneHotEncoder,LabelEncoder, label\_binarize

*#Machine Learning*

**from** sklearn.model\_selection **import** train\_test\_split  
**from** sklearn **import** model\_selection, tree, preprocessing, metrics, linear\_model  
**from** sklearn.svm **import** LinearSVC  
**from** sklearn.ensemble **import** GradientBoostingClassifier  
**from** sklearn.neighbors **import** KNeighborsClassifier  
**from** sklearn.naive\_bayes **import** GaussianNB  
**from** sklearn.linear\_model **import** LinearRegression, LogisticRegression, SGDClassifier

*#ignore warnings for now*  
**import** warnings  
warnings**.**filterwarnings('ignore')

In [467]:

**from** google.colab **import** drive  
drive**.**mount('/content/drive')  
**!**ls "/content/drive/My Drive/Dataset"

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force\_remount=True).  
test.csv test\_label.csv train.csv train\_label.csv

In [468]:

data**=** os**.**listdir('../content/drive/My Drive/Dataset')  
data

Out[468]:

['train\_label.csv', 'test\_label.csv', 'test.csv', 'train.csv']

In [469]:

*# Import train dataset*  
cb\_train**=** pd**.**read\_csv('../content/drive/My Drive/Dataset/train.csv')  
cb\_train

Out[469]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** |
| **0** | 5/2/2012 19:00 | Summer | 0 | 1 | Clear + Few clouds | 22.14 | 25.760 | 77 | 16.9979 |
| **1** | 9/5/2012 4:00 | Fall | 0 | 1 | Clear + Few clouds | 28.70 | 33.335 | 79 | 19.0012 |
| **2** | 1/13/2011 9:00 | Spring | 0 | 1 | Clear + Few clouds | 5.74 | 6.060 | 50 | 22.0028 |
| **3** | 11/18/2011 16:00 | Winter | 0 | 1 | Clear + Few clouds | 13.94 | 16.665 | 29 | 8.9981 |
| **4** | 9/13/2011 13:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 33.335 | 51 | 19.0012 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **8703** | 1/16/2012 6:00 | Spring | 1 | 0 | Clear + Few clouds | 4.10 | 6.820 | 54 | 6.0032 |
| **8704** | 11/10/2011 1:00 | Winter | 0 | 1 | Mist + Cloudy | 16.40 | 20.455 | 87 | 0.0000 |
| **8705** | 4/12/2011 3:00 | Summer | 0 | 1 | Mist + Cloudy | 23.78 | 27.275 | 56 | 8.9981 |
| **8706** | 11/7/2012 1:00 | Winter | 0 | 1 | Mist + Cloudy | 11.48 | 13.635 | 61 | 16.9979 |
| **8707** | 1/10/2011 10:00 | Spring | 0 | 1 | Mist + Cloudy | 5.74 | 6.060 | 50 | 19.9995 |

8708 rows × 9 columns

In [470]:

*# Import test dataset*  
cb\_test**=** pd**.**read\_csv('../content/drive/My Drive/Dataset/test.csv')  
cb\_test

Out[470]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** |
| **0** | 5/10/2012 11:00 | Summer | 0 | 1 | Clear + Few clouds | 21.32 | 25.000 | 48 | 35.0008 |
| **1** | 6/9/2012 7:00 | Summer | 0 | 0 | Clear + Few clouds | 23.78 | 27.275 | 64 | 7.0015 |
| **2** | 3/6/2011 20:00 | Spring | 0 | 0 | Light Snow, Light Rain | 11.48 | 12.120 | 100 | 27.9993 |
| **3** | 10/13/2011 11:00 | Winter | 0 | 1 | Mist + Cloudy | 25.42 | 28.790 | 83 | 0.0000 |
| **4** | 6/2/2012 12:00 | Summer | 0 | 0 | Clear + Few clouds | 25.42 | 31.060 | 43 | 23.9994 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **2173** | 3/8/2012 3:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 63 | 26.0027 |
| **2174** | 1/12/2012 12:00 | Spring | 0 | 1 | Mist + Cloudy | 13.94 | 17.425 | 81 | 7.0015 |
| **2175** | 3/7/2012 22:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 59 | 19.9995 |
| **2176** | 5/12/2011 5:00 | Summer | 0 | 1 | Clear + Few clouds | 17.22 | 21.210 | 94 | 8.9981 |
| **2177** | 7/18/2012 16:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 34.850 | 66 | 16.9979 |

2178 rows × 9 columns

In [471]:

cb\_train\_label**=** pd**.**read\_csv('../content/drive/My Drive/Dataset/train\_label.csv')  
cb\_train\_label

Out[471]:

|  |  |
| --- | --- |
|  | **504** |
| **0** | 5 |
| **1** | 139 |
| **2** | 209 |
| **3** | 184 |
| **4** | 791 |
| **...** | ... |
| **8702** | 13 |
| **8703** | 11 |
| **8704** | 1 |
| **8705** | 92 |
| **8706** | 31 |

8707 rows × 1 columns

In [472]:

*# Import train\_label dataset and name the column*  
cb\_train\_label**=** pd**.**read\_csv('../content/drive/My Drive/Dataset/train\_label.csv',header**=None**)  
new\_col\_list**=**['Total\_booking']  
cb\_train\_label\_rename **=** cb\_train\_label**.**set\_axis(new\_col\_list, axis**=**'columns', inplace**=False**)  
cb\_train\_label\_rename

Out[472]:

|  |  |
| --- | --- |
|  | **Total\_booking** |
| **0** | 504 |
| **1** | 5 |
| **2** | 139 |
| **3** | 209 |
| **4** | 184 |
| **...** | ... |
| **8703** | 13 |
| **8704** | 11 |
| **8705** | 1 |
| **8706** | 92 |
| **8707** | 31 |

8708 rows × 1 columns

In [473]:

*#append the data*   
data\_append**=**pd**.**concat([cb\_train, cb\_train\_label\_rename], axis**=**1, ignore\_index**=False**)  
data\_append

Out[473]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** |
| **0** | 5/2/2012 19:00 | Summer | 0 | 1 | Clear + Few clouds | 22.14 | 25.760 | 77 | 16.9979 | 504 |
| **1** | 9/5/2012 4:00 | Fall | 0 | 1 | Clear + Few clouds | 28.70 | 33.335 | 79 | 19.0012 | 5 |
| **2** | 1/13/2011 9:00 | Spring | 0 | 1 | Clear + Few clouds | 5.74 | 6.060 | 50 | 22.0028 | 139 |
| **3** | 11/18/2011 16:00 | Winter | 0 | 1 | Clear + Few clouds | 13.94 | 16.665 | 29 | 8.9981 | 209 |
| **4** | 9/13/2011 13:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 33.335 | 51 | 19.0012 | 184 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **8703** | 1/16/2012 6:00 | Spring | 1 | 0 | Clear + Few clouds | 4.10 | 6.820 | 54 | 6.0032 | 13 |
| **8704** | 11/10/2011 1:00 | Winter | 0 | 1 | Mist + Cloudy | 16.40 | 20.455 | 87 | 0.0000 | 11 |
| **8705** | 4/12/2011 3:00 | Summer | 0 | 1 | Mist + Cloudy | 23.78 | 27.275 | 56 | 8.9981 | 1 |
| **8706** | 11/7/2012 1:00 | Winter | 0 | 1 | Mist + Cloudy | 11.48 | 13.635 | 61 | 16.9979 | 92 |
| **8707** | 1/10/2011 10:00 | Spring | 0 | 1 | Mist + Cloudy | 5.74 | 6.060 | 50 | 19.9995 | 31 |

8708 rows × 10 columns

In [474]:

data\_append['datetime'] **=** pd**.**to\_datetime(data\_append['datetime'])

data\_append['date'] **=** data\_append['datetime']**.**dt**.**date

data\_append['time'] **=** data\_append['datetime']**.**dt**.**time

data\_append

Out[474]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** | **date** | **time** |
| **0** | 2012-05-02 19:00:00 | Summer | 0 | 1 | Clear + Few clouds | 22.14 | 25.760 | 77 | 16.9979 | 504 | 2012-05-02 | 19:00:00 |
| **1** | 2012-09-05 04:00:00 | Fall | 0 | 1 | Clear + Few clouds | 28.70 | 33.335 | 79 | 19.0012 | 5 | 2012-09-05 | 04:00:00 |
| **2** | 2011-01-13 09:00:00 | Spring | 0 | 1 | Clear + Few clouds | 5.74 | 6.060 | 50 | 22.0028 | 139 | 2011-01-13 | 09:00:00 |
| **3** | 2011-11-18 16:00:00 | Winter | 0 | 1 | Clear + Few clouds | 13.94 | 16.665 | 29 | 8.9981 | 209 | 2011-11-18 | 16:00:00 |
| **4** | 2011-09-13 13:00:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 33.335 | 51 | 19.0012 | 184 | 2011-09-13 | 13:00:00 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **8703** | 2012-01-16 06:00:00 | Spring | 1 | 0 | Clear + Few clouds | 4.10 | 6.820 | 54 | 6.0032 | 13 | 2012-01-16 | 06:00:00 |
| **8704** | 2011-11-10 01:00:00 | Winter | 0 | 1 | Mist + Cloudy | 16.40 | 20.455 | 87 | 0.0000 | 11 | 2011-11-10 | 01:00:00 |
| **8705** | 2011-04-12 03:00:00 | Summer | 0 | 1 | Mist + Cloudy | 23.78 | 27.275 | 56 | 8.9981 | 1 | 2011-04-12 | 03:00:00 |
| **8706** | 2012-11-07 01:00:00 | Winter | 0 | 1 | Mist + Cloudy | 11.48 | 13.635 | 61 | 16.9979 | 92 | 2012-11-07 | 01:00:00 |
| **8707** | 2011-01-10 10:00:00 | Spring | 0 | 1 | Mist + Cloudy | 5.74 | 6.060 | 50 | 19.9995 | 31 | 2011-01-10 | 10:00:00 |

8708 rows × 12 columns

In [475]:

data\_append**.**info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 8708 entries, 0 to 8707  
Data columns (total 12 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 datetime 8708 non-null datetime64[ns]  
 1 season 8708 non-null object   
 2 holiday 8708 non-null int64   
 3 workingday 8708 non-null int64   
 4 weather 8708 non-null object   
 5 temp 8708 non-null float64   
 6 atemp 8708 non-null float64   
 7 humidity 8708 non-null int64   
 8 windspeed 8708 non-null float64   
 9 Total\_booking 8708 non-null int64   
 10 date 8708 non-null object   
 11 time 8708 non-null object   
dtypes: datetime64[ns](1), float64(3), int64(4), object(4)  
memory usage: 816.5+ KB

In [476]:

*#check data types*  
data\_append**.**dtypes

Out[476]:

datetime datetime64[ns]  
season object  
holiday int64  
workingday int64  
weather object  
temp float64  
atemp float64  
humidity int64  
windspeed float64  
Total\_booking int64  
date object  
time object  
dtype: object

In [477]:

*#count data types*  
data\_append**.**dtypes**.**value\_counts()

Out[477]:

object 4  
int64 4  
float64 3  
datetime64[ns] 1  
dtype: int64

In [478]:

*#get count mean, median, mode and standard deviation*  
data\_append**.**describe()

Out[478]:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **holiday** | **workingday** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** |
| **count** | 8708.000000 | 8708.000000 | 8708.000000 | 8708.000000 | 8708.000000 | 8708.000000 | 8708.000000 |
| **mean** | 0.028480 | 0.680294 | 20.243020 | 23.655552 | 61.649173 | 12.843891 | 193.007005 |
| **std** | 0.166348 | 0.466389 | 7.813494 | 8.501107 | 19.199578 | 8.190314 | 181.552211 |
| **min** | 0.000000 | 0.000000 | 0.820000 | 0.760000 | 0.000000 | 0.000000 | 1.000000 |
| **25%** | 0.000000 | 0.000000 | 13.940000 | 16.665000 | 46.000000 | 7.001500 | 43.000000 |
| **50%** | 0.000000 | 1.000000 | 20.500000 | 24.240000 | 61.000000 | 12.998000 | 148.000000 |
| **75%** | 0.000000 | 1.000000 | 26.240000 | 31.060000 | 77.000000 | 16.997900 | 286.000000 |
| **max** | 1.000000 | 1.000000 | 41.000000 | 45.455000 | 100.000000 | 56.996900 | 977.000000 |

In [479]:

*#finding any columns with missing values*   
null\_columns**=**data\_append**.**columns[data\_append**.**isnull()**.**any()]  
data\_append[null\_columns]**.**isnull()**.**sum()

Out[479]:

Series([], dtype: float64)

In [480]:

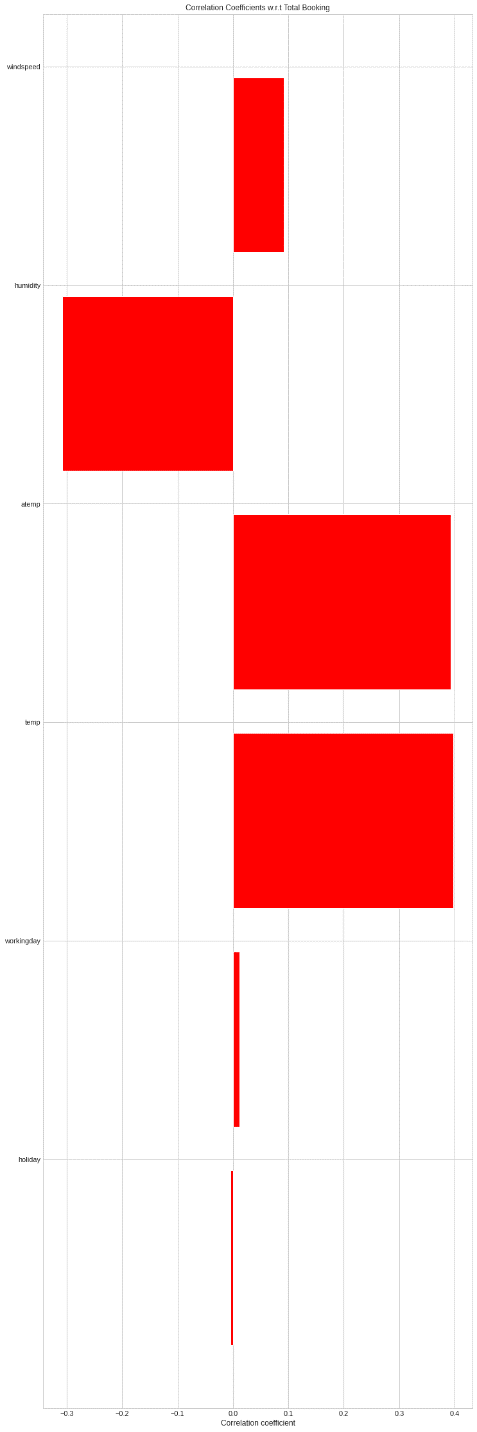
*#correlation between Total\_booking with other features*  
corr**=**data\_append**.**corr()["Total\_booking"]  
corr[np**.**argsort(corr, axis**=**0)[::**-**1]]

Out[480]:

Total\_booking 1.000000  
temp 0.397456  
atemp 0.392754  
windspeed 0.092090  
workingday 0.012285  
holiday -0.004391  
humidity -0.307982  
Name: Total\_booking, dtype: float64

In [481]:

*#plotting correlations*  
num\_feat**=**data\_append**.**columns[data\_append**.**dtypes**!=**object]  
num\_feat**=**num\_feat[1:**-**1]   
labels **=** []  
values **=** []  
**for** col **in** num\_feat:  
 labels**.**append(col)  
 values**.**append(np**.**corrcoef(data\_append[col]**.**values, data\_append**.**Total\_booking**.**values)[0,1])  
   
ind **=** np**.**arange(len(labels))  
width **=** 0.9  
fig, ax **=** plt**.**subplots(figsize**=**(12,40))  
rects **=** ax**.**barh(ind, np**.**array(values), color**=**'red')  
ax**.**set\_yticks(ind**+**((width)**/**2.))  
ax**.**set\_yticklabels(labels, rotation**=**'horizontal')  
ax**.**set\_xlabel("Correlation coefficient")  
ax**.**set\_title("Correlation Coefficients w.r.t Total Booking");



In [482]:

*#multicollinearity in regression*

correlations**=**data\_append**.**corr()  
attrs **=** correlations**.**iloc[:**-**1,:**-**1] *# all except target*

threshold **=** 0.5  
important\_corrs **=** (attrs[abs(attrs) **>** threshold][attrs **!=** 1.0]) \  
 **.**unstack()**.**dropna()**.**to\_dict()

unique\_important\_corrs **=** pd**.**DataFrame(  
 list(set([(tuple(sorted(  
 key)), important\_corrs[key]) \  
 **for** key **in** important\_corrs])),   
 columns**=**['Attribute Pair', 'Correlation'])

unique\_important\_corrs **=** unique\_important\_corrs**.**iloc[  
 abs(unique\_important\_corrs['Correlation'])**.**argsort()[::**-**1]]

unique\_important\_corrs

Out[482]:

|  |  |  |
| --- | --- | --- |
|  | **Attribute Pair** | **Correlation** |
| **0** | (atemp, temp) | 0.984035 |

In [483]:

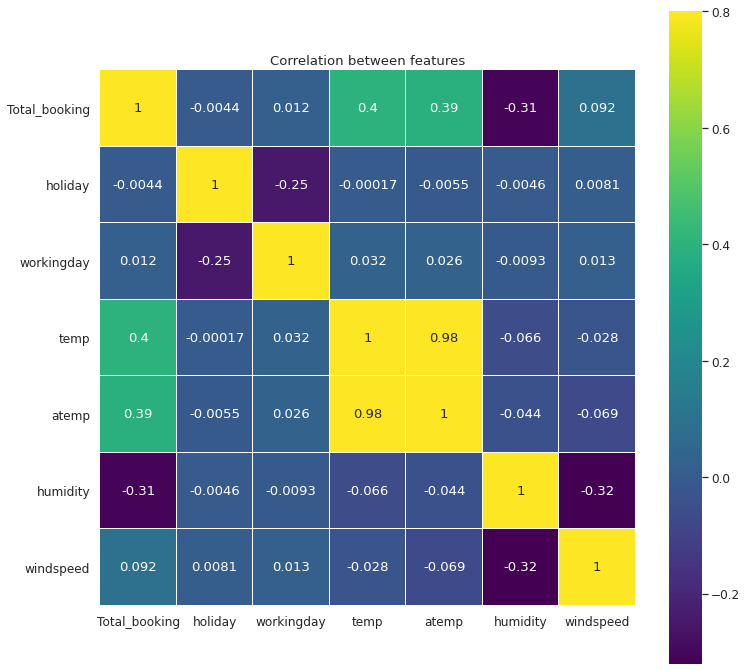
*#heatmap showing multicollinearity*

'"Need to create single features befor it can be used in model predictors, yellow blocks shows correlation"'

corrMatrix**=**data\_append[["Total\_booking", "holiday","workingday","temp",  
 "atemp","humidity","windspeed"]]**.**corr()

sns**.**set(font\_scale**=**1.10)  
plt**.**figure(figsize**=**(12, 12))

sns**.**heatmap(corrMatrix, vmax**=**.8, linewidths**=**0.01,  
 square**=True**,annot**=True**,cmap**=**'viridis',linecolor**=**"white")  
plt**.**title('Correlation between features');



In [484]:

*#pivotal features*  
data\_append[['date','Total\_booking']]**.**groupby(['date'],  
as\_index**=False**)**.**mean()**.**sort\_values(by**=**'date', ascending**=False**)

Out[484]:

|  |  |  |
| --- | --- | --- |
|  | **date** | **Total\_booking** |
| **455** | 2012-12-19 | 232.454545 |
| **454** | 2012-12-18 | 247.238095 |
| **453** | 2012-12-17 | 224.941176 |
| **452** | 2012-12-16 | 155.761905 |
| **451** | 2012-12-15 | 201.388889 |
| **...** | ... | ... |
| **4** | 2011-01-05 | 68.300000 |
| **3** | 2011-01-04 | 70.894737 |
| **2** | 2011-01-03 | 59.800000 |
| **1** | 2011-01-02 | 40.764706 |
| **0** | 2011-01-01 | 45.500000 |

456 rows × 2 columns

In [485]:

data\_append[['time','Total\_booking']]**.**groupby(['time'],  
as\_index**=False**)**.**mean()**.**sort\_values(by**=**'time', ascending**=False**)

Out[485]:

|  |  |  |
| --- | --- | --- |
|  | **time** | **Total\_booking** |
| **23** | 23:00:00 | 89.292225 |
| **22** | 22:00:00 | 137.354571 |
| **21** | 21:00:00 | 174.534626 |
| **20** | 20:00:00 | 232.181564 |
| **19** | 19:00:00 | 314.095109 |
| **18** | 18:00:00 | 428.382514 |
| **17** | 17:00:00 | 474.742466 |
| **16** | 16:00:00 | 315.207547 |
| **15** | 15:00:00 | 257.245283 |
| **14** | 14:00:00 | 243.546742 |
| **13** | 13:00:00 | 259.721485 |
| **12** | 12:00:00 | 257.929730 |
| **11** | 11:00:00 | 215.986188 |
| **10** | 10:00:00 | 171.083333 |
| **9** | 09:00:00 | 220.566929 |
| **8** | 08:00:00 | 369.526012 |
| **7** | 07:00:00 | 216.612813 |
| **6** | 06:00:00 | 74.980822 |
| **5** | 05:00:00 | 19.588889 |
| **4** | 04:00:00 | 6.475073 |
| **3** | 03:00:00 | 11.764873 |
| **2** | 02:00:00 | 23.225000 |
| **1** | 01:00:00 | 33.923513 |
| **0** | 00:00:00 | 56.208571 |

In [486]:

data\_append[['holiday','Total\_booking']]**.**groupby(['holiday'],  
as\_index**=False**)**.**mean()**.**sort\_values(by**=**'holiday', ascending**=False**)

Out[486]:

|  |  |  |
| --- | --- | --- |
|  | **holiday** | **Total\_booking** |
| **1** | 1 | 188.350806 |
| **0** | 0 | 193.143499 |

In [487]:

data\_append[['workingday','Total\_booking']]**.**groupby(['workingday'],  
as\_index**=False**)**.**mean()**.**sort\_values(by**=**'workingday', ascending**=False**)

Out[487]:

|  |  |  |
| --- | --- | --- |
|  | **workingday** | **Total\_booking** |
| **1** | 1 | 194.535955 |
| **0** | 0 | 189.753592 |

In [488]:

data\_append[['temp','Total\_booking']]**.**groupby(['temp'],  
as\_index**=False**)**.**mean()**.**sort\_values(by**=**'temp', ascending**=False**)

Out[488]:

|  |  |  |
| --- | --- | --- |
|  | **temp** | **Total\_booking** |
| **48** | 41.00 | 294.000000 |
| **47** | 39.36 | 317.833333 |
| **46** | 38.54 | 217.500000 |
| **45** | 37.72 | 352.038462 |
| **44** | 36.90 | 327.769231 |
| **43** | 36.08 | 332.500000 |
| **42** | 35.26 | 326.400000 |
| **41** | 34.44 | 351.047619 |
| **40** | 33.62 | 353.786408 |
| **39** | 32.80 | 364.994083 |
| **38** | 31.98 | 317.481481 |
| **37** | 31.16 | 362.900524 |
| **36** | 30.34 | 303.125000 |
| **35** | 29.52 | 280.378472 |
| **34** | 28.70 | 255.133136 |
| **33** | 27.88 | 204.502793 |
| **32** | 27.06 | 215.770393 |
| **31** | 26.24 | 229.684507 |
| **30** | 25.42 | 222.584098 |
| **29** | 24.60 | 241.589577 |
| **28** | 23.78 | 237.790123 |
| **27** | 22.96 | 213.090909 |
| **26** | 22.14 | 186.212698 |
| **25** | 21.32 | 203.265306 |
| **24** | 20.50 | 202.501873 |
| **23** | 19.68 | 182.945736 |
| **22** | 18.86 | 159.798107 |
| **21** | 18.04 | 164.061776 |
| **20** | 17.22 | 187.105455 |
| **19** | 16.40 | 165.701587 |
| **18** | 15.58 | 185.451923 |
| **17** | 14.76 | 157.044199 |
| **16** | 13.94 | 139.089506 |
| **15** | 13.12 | 150.034364 |
| **14** | 12.30 | 116.561728 |
| **13** | 11.48 | 110.760563 |
| **12** | 10.66 | 97.483516 |
| **11** | 9.84 | 90.768240 |
| **10** | 9.02 | 77.129353 |
| **9** | 8.20 | 81.254054 |
| **8** | 7.38 | 70.597701 |
| **7** | 6.56 | 67.633929 |
| **6** | 5.74 | 55.148148 |
| **5** | 4.92 | 50.104167 |
| **4** | 4.10 | 50.583333 |
| **3** | 3.28 | 18.333333 |
| **2** | 2.46 | 43.000000 |
| **1** | 1.64 | 91.500000 |
| **0** | 0.82 | 77.714286 |

In [489]:

data\_append[['atemp','Total\_booking']]**.**groupby(['atemp'],  
as\_index**=False**)**.**mean()**.**sort\_values(by**=**'atemp', ascending**=False**)

Out[489]:

|  |  |  |
| --- | --- | --- |
|  | **atemp** | **Total\_booking** |
| **59** | 45.455 | 312.000000 |
| **58** | 44.695 | 376.000000 |
| **57** | 43.940 | 214.500000 |
| **56** | 43.180 | 307.142857 |
| **55** | 42.425 | 308.739130 |
| **54** | 41.665 | 291.500000 |
| **53** | 40.910 | 328.741935 |
| **52** | 40.150 | 407.633333 |
| **51** | 39.395 | 317.608696 |
| **50** | 38.635 | 341.158730 |
| **49** | 37.880 | 355.400000 |
| **48** | 37.120 | 334.079208 |
| **47** | 36.365 | 356.356436 |
| **46** | 35.605 | 322.443548 |
| **45** | 34.850 | 279.865546 |
| **44** | 34.090 | 297.811429 |
| **43** | 33.335 | 244.557823 |
| **42** | 32.575 | 329.598131 |
| **41** | 31.820 | 257.427386 |
| **40** | 31.060 | 306.877477 |
| **39** | 30.305 | 225.996390 |
| **38** | 29.545 | 155.351220 |
| **37** | 28.790 | 140.636986 |
| **36** | 28.030 | 119.241935 |
| **35** | 27.275 | 211.598131 |
| **34** | 26.515 | 213.090909 |
| **33** | 25.760 | 180.426426 |
| **32** | 25.000 | 202.613559 |
| **31** | 24.240 | 202.501873 |
| **30** | 23.485 | 182.945736 |
| **29** | 22.725 | 159.798107 |
| **28** | 21.970 | 164.061776 |
| **27** | 21.210 | 187.105455 |
| **26** | 20.455 | 165.701587 |
| **25** | 19.695 | 185.451923 |
| **24** | 18.940 | 168.606061 |
| **23** | 18.180 | 128.130435 |
| **22** | 17.425 | 145.502024 |
| **21** | 16.665 | 155.807692 |
| **20** | 15.910 | 130.940000 |
| **19** | 15.150 | 128.985130 |
| **18** | 14.395 | 112.156250 |
| **17** | 13.635 | 95.828125 |
| **16** | 12.880 | 90.067961 |
| **15** | 12.120 | 105.437500 |
| **14** | 11.365 | 98.526570 |
| **13** | 10.605 | 102.477612 |
| **12** | 9.850 | 80.188679 |
| **11** | 9.090 | 79.247059 |
| **10** | 8.335 | 58.020000 |
| **9** | 7.575 | 47.966102 |
| **8** | 6.820 | 62.612245 |
| **7** | 6.060 | 69.719298 |
| **6** | 5.305 | 56.150000 |
| **5** | 4.545 | 80.555556 |
| **4** | 3.790 | 38.466667 |
| **3** | 3.030 | 82.285714 |
| **2** | 2.275 | 38.000000 |
| **1** | 1.515 | 3.000000 |
| **0** | 0.760 | 1.000000 |

In [490]:

data\_append[['humidity','Total\_booking']]**.**groupby(['humidity'],  
as\_index**=False**)**.**mean()**.**sort\_values(by**=**'humidity', ascending**=False**)

Out[490]:

|  |  |  |
| --- | --- | --- |
|  | **humidity** | **Total\_booking** |
| **87** | 100 | 66.710280 |
| **86** | 96 | 71.000000 |
| **85** | 94 | 94.657258 |
| **84** | 93 | 78.145570 |
| **83** | 92 | 76.000000 |
| **...** | ... | ... |
| **4** | 13 | 17.000000 |
| **3** | 12 | 29.000000 |
| **2** | 10 | 107.000000 |
| **1** | 8 | 77.000000 |
| **0** | 0 | 30.833333 |

88 rows × 2 columns

In [491]:

data\_append[['windspeed','Total\_booking']]**.**groupby(['windspeed'],  
as\_index**=False**)**.**mean()**.**sort\_values(by**=**'windspeed', ascending**=False**)

Out[491]:

|  |  |  |
| --- | --- | --- |
|  | **windspeed** | **Total\_booking** |
| **27** | 56.9969 | 358.000000 |
| **26** | 51.9987 | 5.000000 |
| **25** | 50.0021 | 171.000000 |
| **24** | 47.9988 | 140.500000 |
| **23** | 46.0022 | 67.333333 |
| **22** | 43.9989 | 209.428571 |
| **21** | 43.0006 | 92.833333 |
| **20** | 40.9973 | 150.600000 |
| **19** | 39.0007 | 168.409091 |
| **18** | 36.9974 | 211.578947 |
| **17** | 35.0008 | 216.688889 |
| **16** | 32.9975 | 195.614286 |
| **15** | 31.0009 | 195.746667 |
| **14** | 30.0026 | 217.814433 |
| **13** | 27.9993 | 223.737179 |
| **12** | 26.0027 | 221.242105 |
| **11** | 23.9994 | 225.727273 |
| **10** | 22.0028 | 186.530201 |
| **9** | 19.9995 | 221.027990 |
| **8** | 19.0012 | 216.801457 |
| **7** | 16.9979 | 217.408537 |
| **6** | 15.0013 | 207.717277 |
| **5** | 12.9980 | 207.928398 |
| **4** | 11.0014 | 201.247362 |
| **3** | 8.9981 | 175.006772 |
| **2** | 7.0015 | 172.091898 |
| **1** | 6.0032 | 155.417889 |
| **0** | 0.0000 | 164.940341 |

In [492]:

*#visualizing target variables(univariate analysis)*  
sns**.**distplot(data\_append['Total\_booking'], color**=**"r", kde**=False**)  
plt**.**title("Distribution of Total booking")  
plt**.**ylabel("Number of Occurences")  
plt**.**xlabel("Total booking");



In [493]:

*#skewness*

data\_append['Total\_booking']**.**skew()

Out[493]:

1.2379281606952324

In [494]:

*#kurtosis*

data\_append['Total\_booking']**.**kurt()

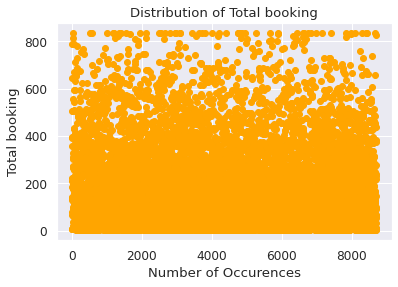
Out[494]:

1.3117222889080962

In [495]:

*#need to remove some outliers.*  
upperlimit **=** np**.**percentile(data\_append**.**Total\_booking**.**values, 99.5)  
data\_append['Total\_booking']**.**iloc[data\_append['Total\_booking']**>**upperlimit] **=** upperlimit

plt**.**scatter(range(data\_append**.**shape[0]), data\_append["Total\_booking"]**.**values,color**=**'orange')  
plt**.**title("Distribution of Total booking")  
plt**.**xlabel("Number of Occurences")  
plt**.**ylabel("Total booking");



In [496]:

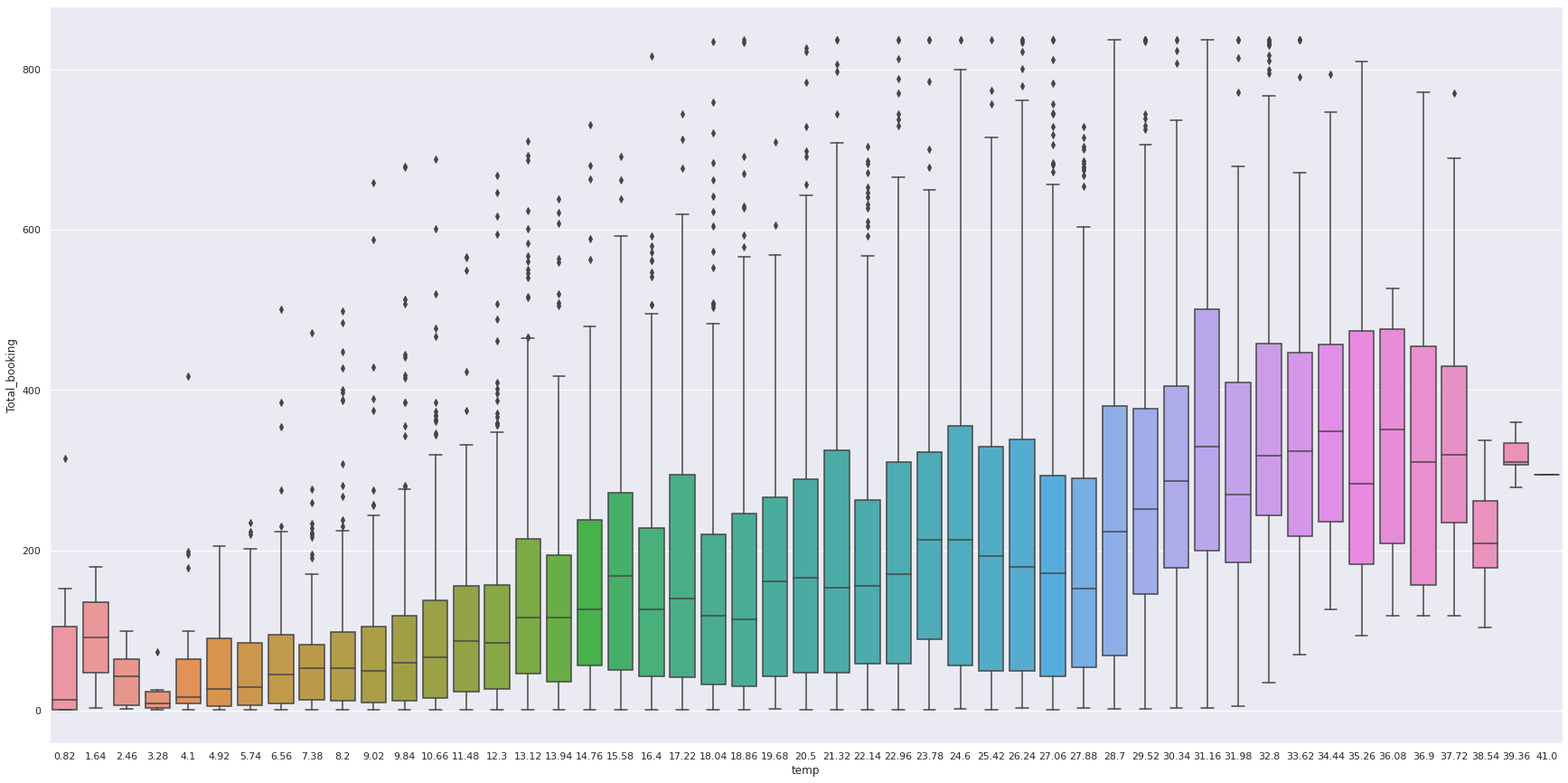
*#lets see if there are any columns with missing values*   
null\_columns**=**data\_append**.**columns[data\_append**.**isnull()**.**any()]  
data\_append[null\_columns]**.**isnull()**.**sum()

Out[496]:

Series([], dtype: float64)

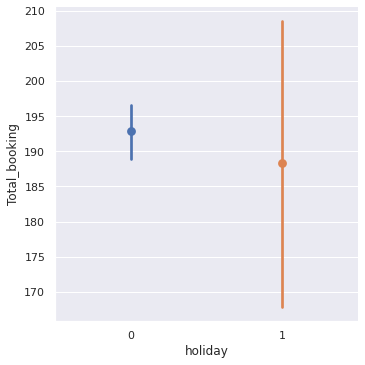
In [497]:

**import** seaborn **as** sns  
sns**.**set(rc**=**{'figure.figsize':(30,15)})  
sns**.**boxplot("temp","Total\_booking",data**=**data\_append);



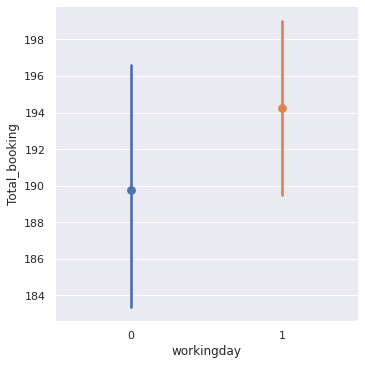
In [498]:

sns**.**factorplot("holiday","Total\_booking",data**=**data\_append,hue**=**"holiday");



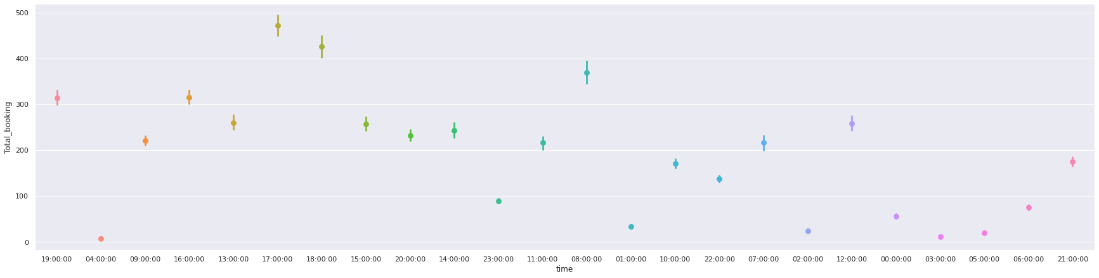
In [499]:

sns**.**factorplot("workingday","Total\_booking",data**=**data\_append,hue**=**"workingday");



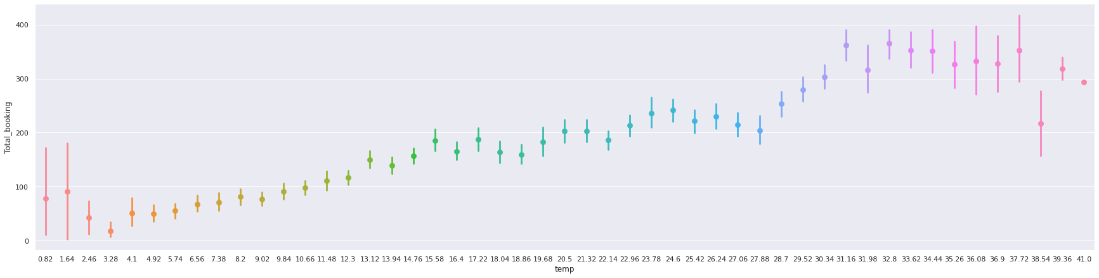
In [500]:

sns**.**factorplot("time","Total\_booking",data**=**data\_append,hue**=**"time", size**=**6, aspect**=**4);



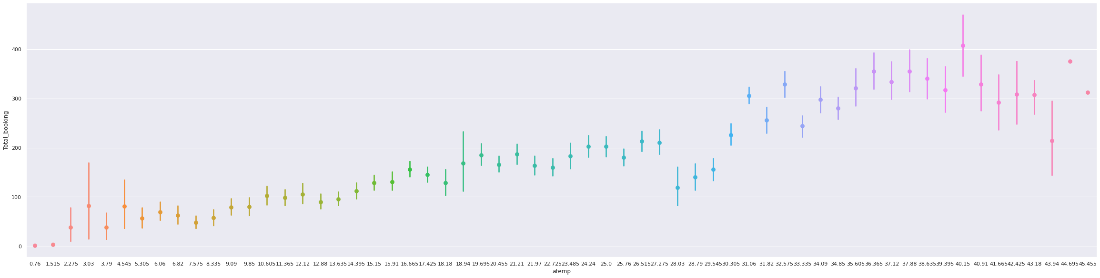
In [501]:

sns**.**factorplot("temp","Total\_booking",data**=**data\_append,hue**=**"temp", size**=**6, aspect**=**4);



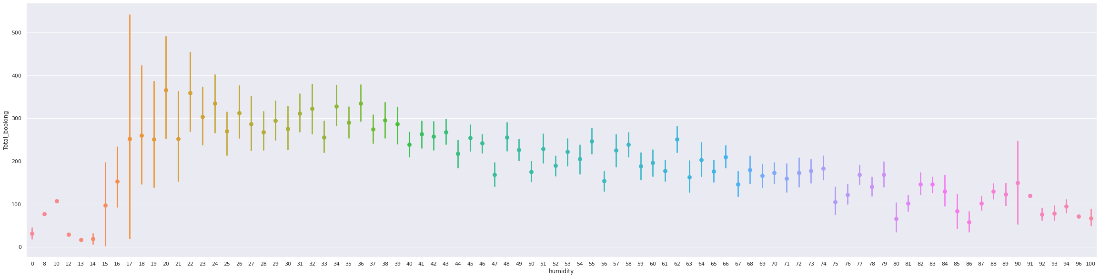
In [502]:

sns**.**factorplot("atemp","Total\_booking",data**=**data\_append,hue**=**"atemp", size**=**8, aspect**=**4);



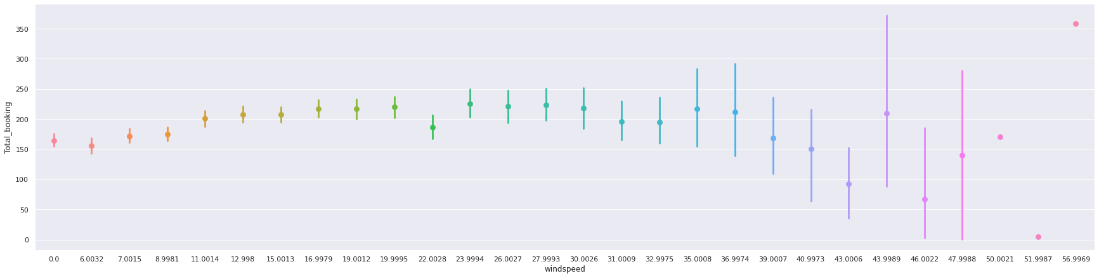
In [503]:

sns**.**factorplot("humidity","Total\_booking",data**=**data\_append,hue**=**"humidity", size**=**8, aspect**=**4);



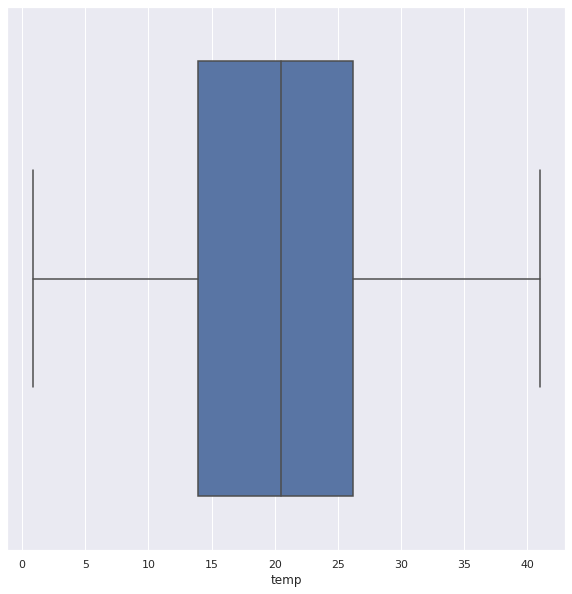
In [504]:

sns**.**factorplot("windspeed","Total\_booking",data**=**data\_append,hue**=**"windspeed", size**=**6, aspect**=**4);



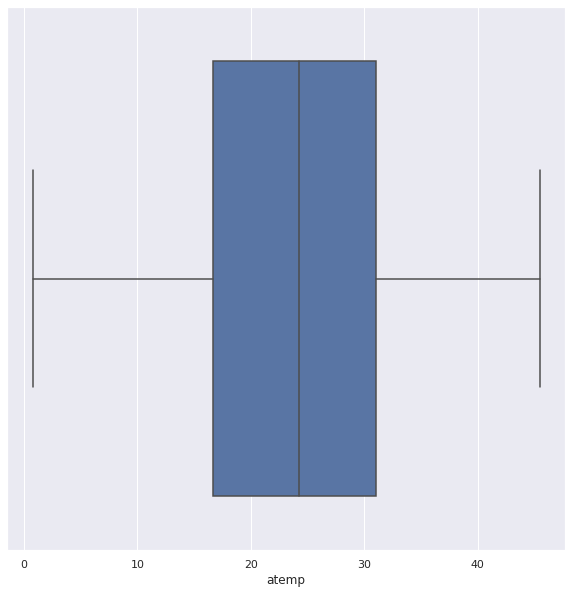
In [505]:

*#boxplot for temp*  
sns**.**set(rc**=**{'figure.figsize':(10,10)})  
sns**.**boxplot("temp",data**=**data\_append);



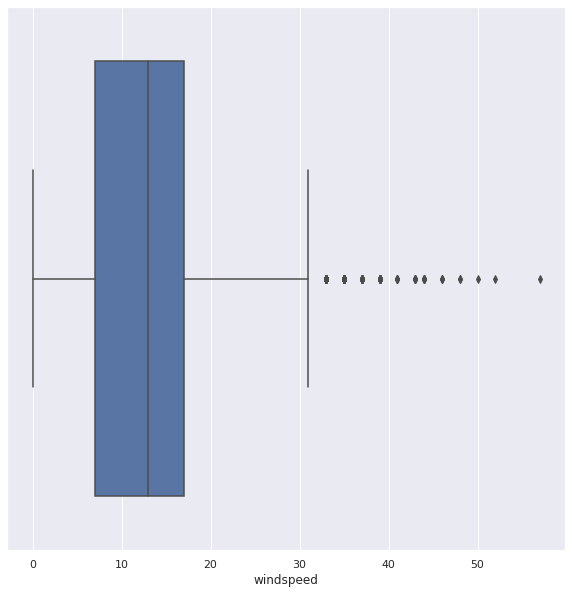
In [506]:

*#boxplot for atemp*  
sns**.**set(rc**=**{'figure.figsize':(10,10)})  
sns**.**boxplot("atemp",data**=**data\_append);



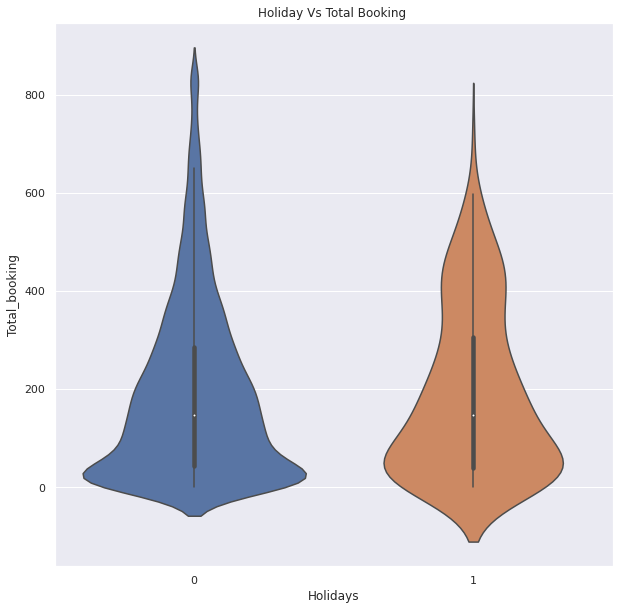
In [507]:

*#boxplot for windspeed*  
sns**.**set(rc**=**{'figure.figsize':(10,10)})  
sns**.**boxplot("windspeed",data**=**data\_append);



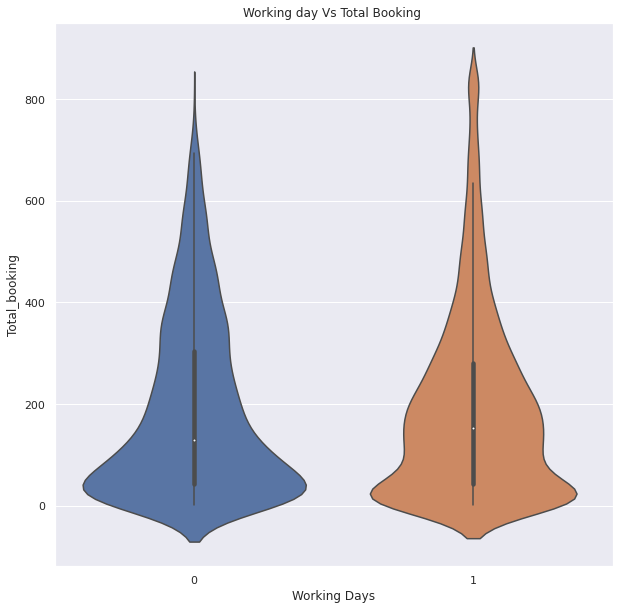
In [508]:

sns**.**violinplot(data\_append["holiday"],data\_append["Total\_booking"])  
plt**.**title("Holiday Vs Total Booking ")  
plt**.**ylabel("Total\_booking")  
plt**.**xlabel("Holidays");



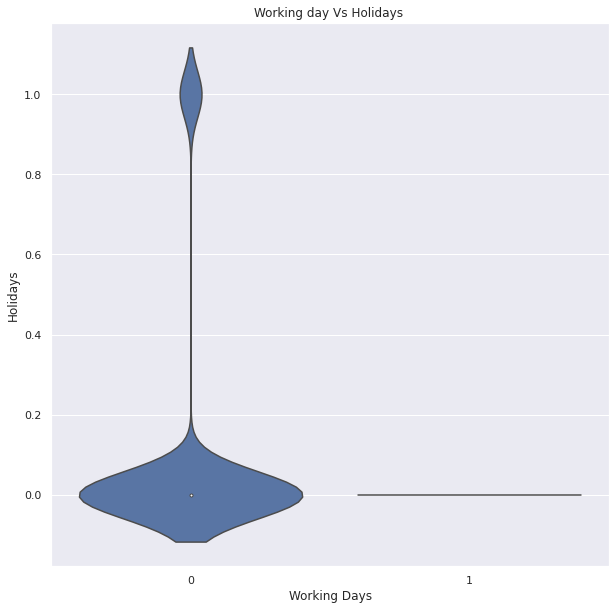
In [509]:

sns**.**violinplot(data\_append["workingday"],data\_append["Total\_booking"])  
plt**.**title("Working day Vs Total Booking ")  
plt**.**ylabel("Total\_booking")  
plt**.**xlabel("Working Days");



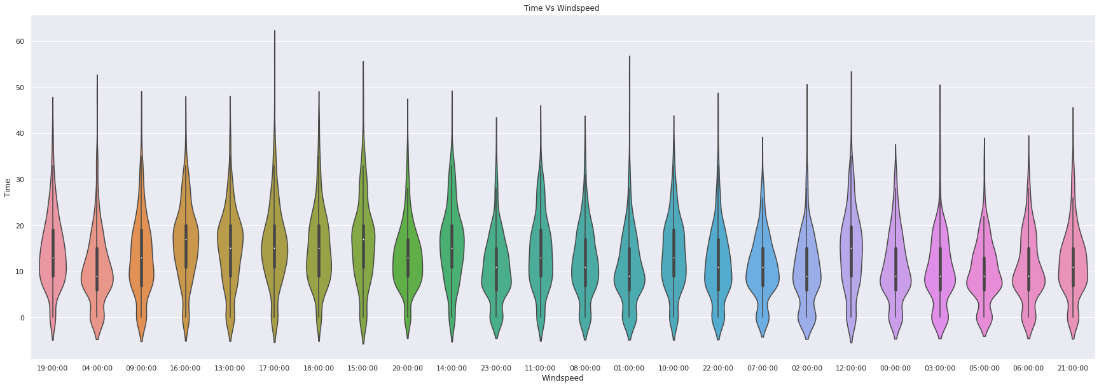
In [510]:

sns**.**violinplot(data\_append["workingday"],data\_append["holiday"])  
plt**.**title("Working day Vs Holidays ")  
plt**.**ylabel("Holidays")  
plt**.**xlabel("Working Days");



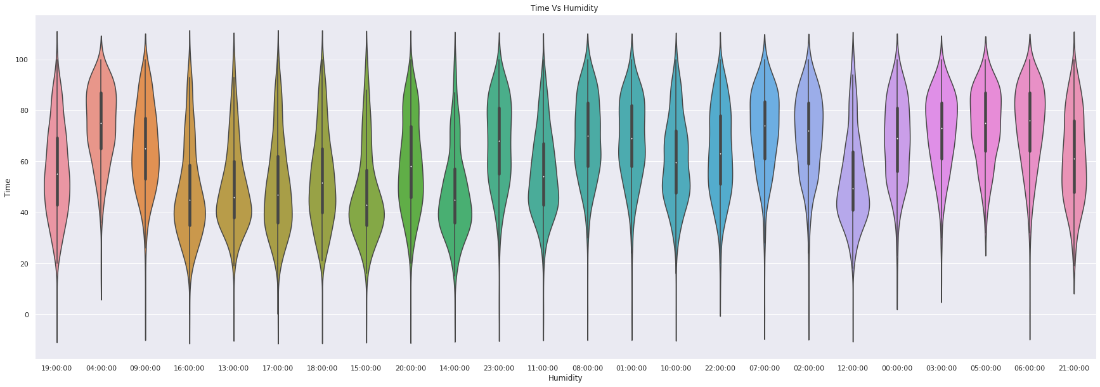
In [511]:

sns**.**set(rc**=**{'figure.figsize':(30,10)})  
sns**.**violinplot(data\_append["time"],data\_append["windspeed"])  
plt**.**title("Time Vs Windspeed ")  
plt**.**ylabel("Time")  
plt**.**xlabel("Windspeed");



In [512]:

sns**.**set(rc**=**{'figure.figsize':(30,10)})  
sns**.**violinplot(data\_append["time"],data\_append["humidity"])  
plt**.**title("Time Vs Humidity ")  
plt**.**ylabel("Time")  
plt**.**xlabel("Humidity");



In [513]:

*#looking at test data*

cb\_test

Out[513]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** |
| **0** | 5/10/2012 11:00 | Summer | 0 | 1 | Clear + Few clouds | 21.32 | 25.000 | 48 | 35.0008 |
| **1** | 6/9/2012 7:00 | Summer | 0 | 0 | Clear + Few clouds | 23.78 | 27.275 | 64 | 7.0015 |
| **2** | 3/6/2011 20:00 | Spring | 0 | 0 | Light Snow, Light Rain | 11.48 | 12.120 | 100 | 27.9993 |
| **3** | 10/13/2011 11:00 | Winter | 0 | 1 | Mist + Cloudy | 25.42 | 28.790 | 83 | 0.0000 |
| **4** | 6/2/2012 12:00 | Summer | 0 | 0 | Clear + Few clouds | 25.42 | 31.060 | 43 | 23.9994 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **2173** | 3/8/2012 3:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 63 | 26.0027 |
| **2174** | 1/12/2012 12:00 | Spring | 0 | 1 | Mist + Cloudy | 13.94 | 17.425 | 81 | 7.0015 |
| **2175** | 3/7/2012 22:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 59 | 19.9995 |
| **2176** | 5/12/2011 5:00 | Summer | 0 | 1 | Clear + Few clouds | 17.22 | 21.210 | 94 | 8.9981 |
| **2177** | 7/18/2012 16:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 34.850 | 66 | 16.9979 |

2178 rows × 9 columns

In [514]:

*#looking at test\_label and naming Total\_booking*

cb\_test\_label**=** pd**.**read\_csv('../content/drive/My Drive/Dataset/test\_label.csv')  
new\_col\_list\_test**=**['Total\_booking']  
cb\_test\_label\_rename **=** cb\_test\_label**.**set\_axis(new\_col\_list\_test, axis**=**'columns', inplace**=False**)  
cb\_test\_label\_rename

Out[514]:

|  |  |
| --- | --- |
|  | **Total\_booking** |
| **0** | 87 |
| **1** | 11 |
| **2** | 84 |
| **3** | 668 |
| **4** | 53 |
| **...** | ... |
| **2172** | 3 |
| **2173** | 144 |
| **2174** | 159 |
| **2175** | 29 |
| **2176** | 224 |

2177 rows × 1 columns

In [515]:

*#append the data for test*  
data\_append\_test**=**pd**.**concat([cb\_test, cb\_test\_label\_rename], axis**=**1, ignore\_index**=False**)  
data\_append\_test

Out[515]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** |
| **0** | 5/10/2012 11:00 | Summer | 0 | 1 | Clear + Few clouds | 21.32 | 25.000 | 48 | 35.0008 | 87.0 |
| **1** | 6/9/2012 7:00 | Summer | 0 | 0 | Clear + Few clouds | 23.78 | 27.275 | 64 | 7.0015 | 11.0 |
| **2** | 3/6/2011 20:00 | Spring | 0 | 0 | Light Snow, Light Rain | 11.48 | 12.120 | 100 | 27.9993 | 84.0 |
| **3** | 10/13/2011 11:00 | Winter | 0 | 1 | Mist + Cloudy | 25.42 | 28.790 | 83 | 0.0000 | 668.0 |
| **4** | 6/2/2012 12:00 | Summer | 0 | 0 | Clear + Few clouds | 25.42 | 31.060 | 43 | 23.9994 | 53.0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **2173** | 3/8/2012 3:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 63 | 26.0027 | 144.0 |
| **2174** | 1/12/2012 12:00 | Spring | 0 | 1 | Mist + Cloudy | 13.94 | 17.425 | 81 | 7.0015 | 159.0 |
| **2175** | 3/7/2012 22:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 59 | 19.9995 | 29.0 |
| **2176** | 5/12/2011 5:00 | Summer | 0 | 1 | Clear + Few clouds | 17.22 | 21.210 | 94 | 8.9981 | 224.0 |
| **2177** | 7/18/2012 16:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 34.850 | 66 | 16.9979 | NaN |

2178 rows × 10 columns

In [516]:

*#seperate date time column*

data\_append\_test['datetime'] **=** pd**.**to\_datetime(data\_append\_test['datetime'])

data\_append\_test['date'] **=** data\_append\_test['datetime']**.**dt**.**date

data\_append\_test['time'] **=** data\_append\_test['datetime']**.**dt**.**time

data\_append\_test

Out[516]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** | **date** | **time** |
| **0** | 2012-05-10 11:00:00 | Summer | 0 | 1 | Clear + Few clouds | 21.32 | 25.000 | 48 | 35.0008 | 87.0 | 2012-05-10 | 11:00:00 |
| **1** | 2012-06-09 07:00:00 | Summer | 0 | 0 | Clear + Few clouds | 23.78 | 27.275 | 64 | 7.0015 | 11.0 | 2012-06-09 | 07:00:00 |
| **2** | 2011-03-06 20:00:00 | Spring | 0 | 0 | Light Snow, Light Rain | 11.48 | 12.120 | 100 | 27.9993 | 84.0 | 2011-03-06 | 20:00:00 |
| **3** | 2011-10-13 11:00:00 | Winter | 0 | 1 | Mist + Cloudy | 25.42 | 28.790 | 83 | 0.0000 | 668.0 | 2011-10-13 | 11:00:00 |
| **4** | 2012-06-02 12:00:00 | Summer | 0 | 0 | Clear + Few clouds | 25.42 | 31.060 | 43 | 23.9994 | 53.0 | 2012-06-02 | 12:00:00 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **2173** | 2012-03-08 03:00:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 63 | 26.0027 | 144.0 | 2012-03-08 | 03:00:00 |
| **2174** | 2012-01-12 12:00:00 | Spring | 0 | 1 | Mist + Cloudy | 13.94 | 17.425 | 81 | 7.0015 | 159.0 | 2012-01-12 | 12:00:00 |
| **2175** | 2012-03-07 22:00:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 59 | 19.9995 | 29.0 | 2012-03-07 | 22:00:00 |
| **2176** | 2011-05-12 05:00:00 | Summer | 0 | 1 | Clear + Few clouds | 17.22 | 21.210 | 94 | 8.9981 | 224.0 | 2011-05-12 | 05:00:00 |
| **2177** | 2012-07-18 16:00:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 34.850 | 66 | 16.9979 | NaN | 2012-07-18 | 16:00:00 |

2178 rows × 12 columns

In [517]:

*#checking null values*

data\_append\_test**.**isnull()**.**sum()

Out[517]:

datetime 0  
season 0  
holiday 0  
workingday 0  
weather 0  
temp 0  
atemp 0  
humidity 0  
windspeed 0  
Total\_booking 1  
date 0  
time 0  
dtype: int64

In [518]:

*#null values found in total bookings replacing with random values*  
'"replacing na values with random"'

data\_append\_test["Total\_booking"]**.**fillna( method **=**'ffill', inplace **= True**)   
data\_append\_test**.**isnull()**.**sum()

Out[518]:

datetime 0  
season 0  
holiday 0  
workingday 0  
weather 0  
temp 0  
atemp 0  
humidity 0  
windspeed 0  
Total\_booking 0  
date 0  
time 0  
dtype: int64

In [519]:

*#for train data*

data\_append**.**isnull()**.**sum()

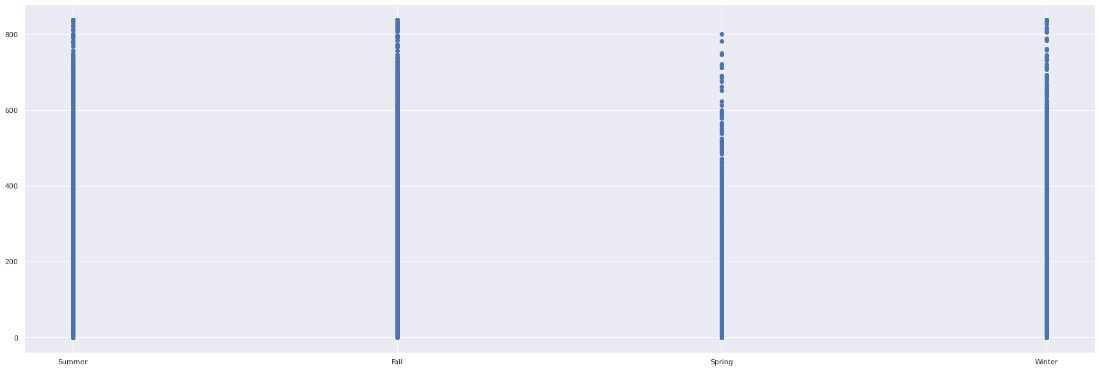
Out[519]:

datetime 0  
season 0  
holiday 0  
workingday 0  
weather 0  
temp 0  
atemp 0  
humidity 0  
windspeed 0  
Total\_booking 0  
date 0  
time 0  
dtype: int64

In [520]:

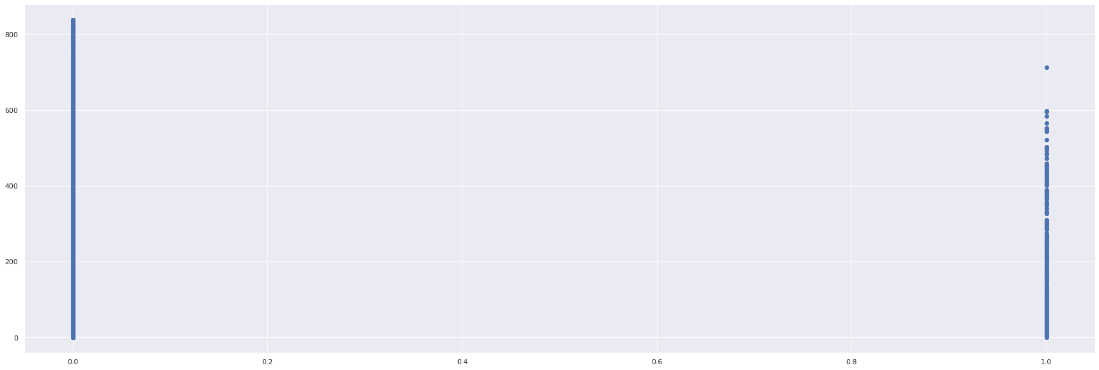
*#plotting columns of season*

plt**.**scatter(data\_append['season'],data\_append['Total\_booking'])  
plt**.**show()



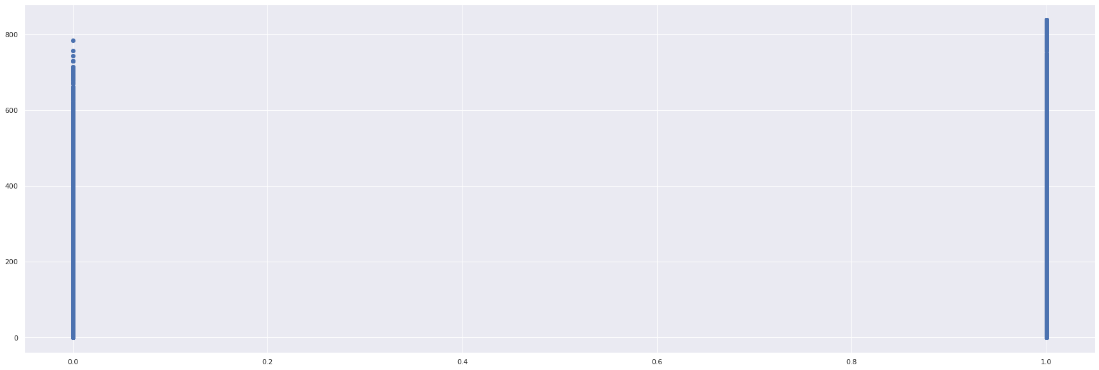
In [521]:

plt**.**scatter(data\_append['holiday'],data\_append['Total\_booking'])  
plt**.**show()



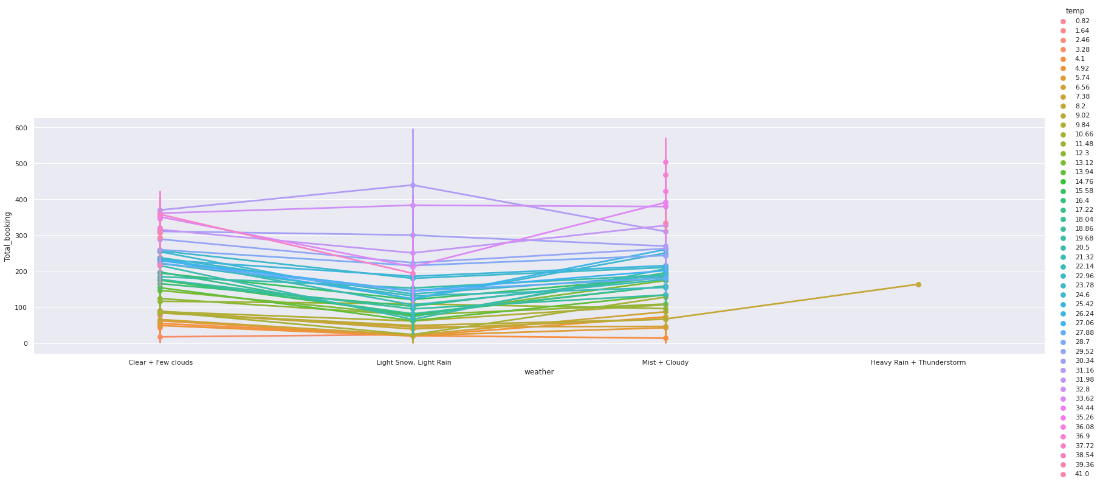
In [522]:

plt**.**scatter(data\_append['workingday'],data\_append['Total\_booking'])  
plt**.**show()



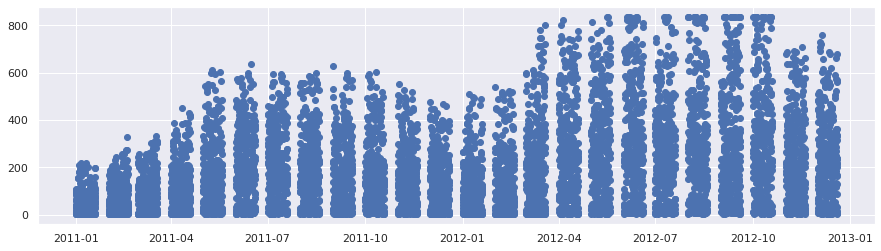
In [523]:

sns**.**factorplot("weather","Total\_booking",data**=**data\_append,hue**=**"temp", size**=**6, aspect**=**4);



In [524]:

plt**.**figure(figsize**=**(15,4))  
plt**.**scatter(data\_append['date'],data\_append['Total\_booking'])  
plt**.**show()



In [525]:

data\_append\_test

Out[525]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** | **date** | **time** |
| **0** | 2012-05-10 11:00:00 | Summer | 0 | 1 | Clear + Few clouds | 21.32 | 25.000 | 48 | 35.0008 | 87.0 | 2012-05-10 | 11:00:00 |
| **1** | 2012-06-09 07:00:00 | Summer | 0 | 0 | Clear + Few clouds | 23.78 | 27.275 | 64 | 7.0015 | 11.0 | 2012-06-09 | 07:00:00 |
| **2** | 2011-03-06 20:00:00 | Spring | 0 | 0 | Light Snow, Light Rain | 11.48 | 12.120 | 100 | 27.9993 | 84.0 | 2011-03-06 | 20:00:00 |
| **3** | 2011-10-13 11:00:00 | Winter | 0 | 1 | Mist + Cloudy | 25.42 | 28.790 | 83 | 0.0000 | 668.0 | 2011-10-13 | 11:00:00 |
| **4** | 2012-06-02 12:00:00 | Summer | 0 | 0 | Clear + Few clouds | 25.42 | 31.060 | 43 | 23.9994 | 53.0 | 2012-06-02 | 12:00:00 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **2173** | 2012-03-08 03:00:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 63 | 26.0027 | 144.0 | 2012-03-08 | 03:00:00 |
| **2174** | 2012-01-12 12:00:00 | Spring | 0 | 1 | Mist + Cloudy | 13.94 | 17.425 | 81 | 7.0015 | 159.0 | 2012-01-12 | 12:00:00 |
| **2175** | 2012-03-07 22:00:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 59 | 19.9995 | 29.0 | 2012-03-07 | 22:00:00 |
| **2176** | 2011-05-12 05:00:00 | Summer | 0 | 1 | Clear + Few clouds | 17.22 | 21.210 | 94 | 8.9981 | 224.0 | 2011-05-12 | 05:00:00 |
| **2177** | 2012-07-18 16:00:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 34.850 | 66 | 16.9979 | 224.0 | 2012-07-18 | 16:00:00 |

2178 rows × 12 columns

In [564]:

train**=** data\_append**.**append(data\_append\_test)

In [565]:

train

Out[565]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **datetime** | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** | **date** | **time** |
| **0** | 2012-05-02 19:00:00 | Summer | 0 | 1 | Clear + Few clouds | 22.14 | 25.760 | 77 | 16.9979 | 504.0 | 2012-05-02 | 19:00:00 |
| **1** | 2012-09-05 04:00:00 | Fall | 0 | 1 | Clear + Few clouds | 28.70 | 33.335 | 79 | 19.0012 | 5.0 | 2012-09-05 | 04:00:00 |
| **2** | 2011-01-13 09:00:00 | Spring | 0 | 1 | Clear + Few clouds | 5.74 | 6.060 | 50 | 22.0028 | 139.0 | 2011-01-13 | 09:00:00 |
| **3** | 2011-11-18 16:00:00 | Winter | 0 | 1 | Clear + Few clouds | 13.94 | 16.665 | 29 | 8.9981 | 209.0 | 2011-11-18 | 16:00:00 |
| **4** | 2011-09-13 13:00:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 33.335 | 51 | 19.0012 | 184.0 | 2011-09-13 | 13:00:00 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **2173** | 2012-03-08 03:00:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 63 | 26.0027 | 144.0 | 2012-03-08 | 03:00:00 |
| **2174** | 2012-01-12 12:00:00 | Spring | 0 | 1 | Mist + Cloudy | 13.94 | 17.425 | 81 | 7.0015 | 159.0 | 2012-01-12 | 12:00:00 |
| **2175** | 2012-03-07 22:00:00 | Spring | 0 | 1 | Clear + Few clouds | 18.86 | 22.725 | 59 | 19.9995 | 29.0 | 2012-03-07 | 22:00:00 |
| **2176** | 2011-05-12 05:00:00 | Summer | 0 | 1 | Clear + Few clouds | 17.22 | 21.210 | 94 | 8.9981 | 224.0 | 2011-05-12 | 05:00:00 |
| **2177** | 2012-07-18 16:00:00 | Fall | 0 | 1 | Clear + Few clouds | 30.34 | 34.850 | 66 | 16.9979 | 224.0 | 2012-07-18 | 16:00:00 |

10886 rows × 12 columns

In [566]:

train**.**info()

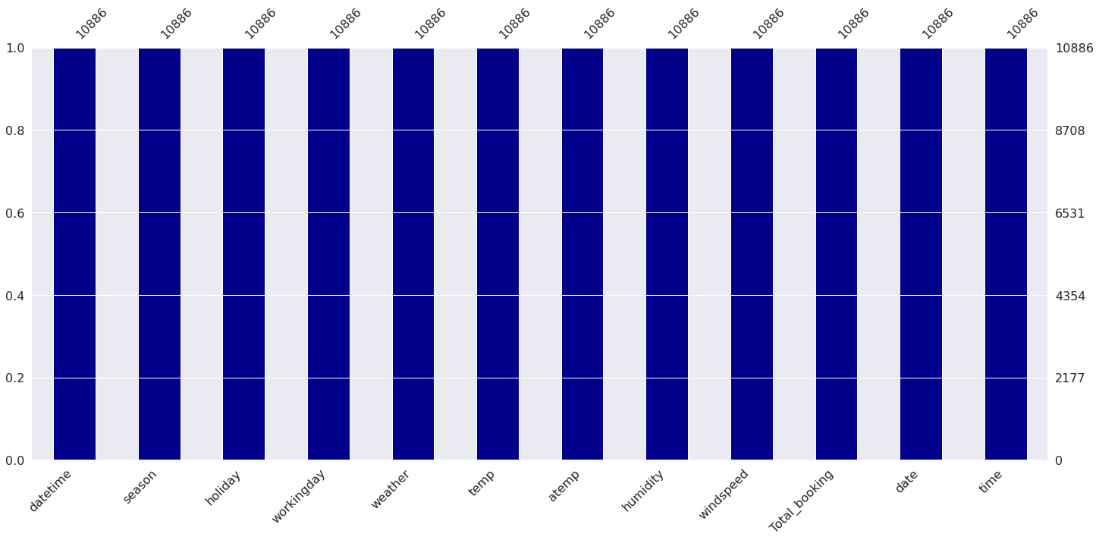
<class 'pandas.core.frame.DataFrame'>  
Int64Index: 10886 entries, 0 to 2177  
Data columns (total 12 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 datetime 10886 non-null datetime64[ns]  
 1 season 10886 non-null object   
 2 holiday 10886 non-null int64   
 3 workingday 10886 non-null int64   
 4 weather 10886 non-null object   
 5 temp 10886 non-null float64   
 6 atemp 10886 non-null float64   
 7 humidity 10886 non-null int64   
 8 windspeed 10886 non-null float64   
 9 Total\_booking 10886 non-null float64   
 10 date 10886 non-null object   
 11 time 10886 non-null object   
dtypes: datetime64[ns](1), float64(4), int64(3), object(4)  
memory usage: 1.1+ MB

In [567]:

**import** missingno **as** msno  
msno**.**bar(train, color**=**'darkblue')

Out[567]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7effcde99668>



In [568]:

train['Total\_booking']**.**value\_counts()

Out[568]:

5.0 169  
4.0 149  
3.0 144  
6.0 135  
2.0 132  
 ...   
755.0 1  
624.0 1  
747.0 1  
636.0 1  
606.0 1  
Name: Total\_booking, Length: 787, dtype: int64

In [569]:

train['Total\_booking']**.**fillna(0)**.**value\_counts()

Out[569]:

5.0 169  
4.0 149  
3.0 144  
6.0 135  
2.0 132  
 ...   
755.0 1  
624.0 1  
747.0 1  
636.0 1  
606.0 1  
Name: Total\_booking, Length: 787, dtype: int64

In [570]:

train**.**drop('datetime', axis**=**1, inplace**=True**)

In [571]:

**from** sklearn.preprocessing **import** LabelEncoder

df\_lb **=** train**.**copy()  
df\_lb['season']**.**value\_counts()

Out[571]:

Winter 2734  
Summer 2733  
Fall 2733  
Spring 2686  
Name: season, dtype: int64

In [572]:

lbl\_enc **=** LabelEncoder()

*# fit label encoder and transform values on ord\_2 column*  
df\_lb**.**loc[:, "season"] **=** lbl\_enc**.**fit\_transform(df\_lb['season']**.**values)

df\_lb['season']**.**value\_counts()

Out[572]:

3 2734  
2 2733  
0 2733  
1 2686  
Name: season, dtype: int64

In [573]:

df\_lb1 **=** train**.**copy()  
df\_lb1['weather']**.**value\_counts()

Out[573]:

Clear + Few clouds 7192  
 Mist + Cloudy 2834  
 Light Snow, Light Rain 859  
 Heavy Rain + Thunderstorm 1  
Name: weather, dtype: int64

In [574]:

lbl\_enc **=** LabelEncoder()

*# fit label encoder and transform values on ord\_2 column*  
df\_lb1**.**loc[:, "weather"] **=** lbl\_enc**.**fit\_transform(df\_lb1['weather']**.**values)

df\_lb1['weather']**.**value\_counts()

Out[574]:

0 7192  
3 2834  
2 859  
1 1  
Name: weather, dtype: int64

In [575]:

df\_lb2 **=** train**.**copy()  
df\_lb2['date']**.**value\_counts()

Out[575]:

2012-05-05 24  
2012-11-12 24  
2012-09-11 24  
2012-11-18 24  
2011-06-08 24  
 ..  
2011-01-12 22  
2011-03-10 22  
2011-02-11 22  
2011-01-03 22  
2011-01-18 12  
Name: date, Length: 456, dtype: int64

In [576]:

lbl\_enc **=** LabelEncoder()

*# fit label encoder and transform values on ord\_2 column*  
df\_lb2**.**loc[:, "date"] **=** lbl\_enc**.**fit\_transform(df\_lb2['date']**.**values)

df\_lb2['date']**.**value\_counts()

Out[576]:

455 24  
354 24  
266 24  
274 24  
282 24  
 ..  
2 22  
11 22  
47 22  
29 22  
17 12  
Name: date, Length: 456, dtype: int64

In [577]:

df\_lb3 **=** train**.**copy()  
df\_lb3['time']**.**value\_counts()

Out[577]:

20:00:00 456  
12:00:00 456  
19:00:00 456  
23:00:00 456  
16:00:00 456  
15:00:00 456  
18:00:00 456  
14:00:00 456  
22:00:00 456  
13:00:00 456  
21:00:00 456  
17:00:00 456  
10:00:00 455  
07:00:00 455  
11:00:00 455  
00:00:00 455  
09:00:00 455  
08:00:00 455  
06:00:00 455  
01:00:00 454  
05:00:00 452  
02:00:00 448  
04:00:00 442  
03:00:00 433  
Name: time, dtype: int64

In [578]:

lbl\_enc **=** LabelEncoder()

*# fit label encoder and transform values on ord\_2 column*  
df\_lb3**.**loc[:, "time"] **=** lbl\_enc**.**fit\_transform(df\_lb3['time']**.**values)

df\_lb3['time']**.**value\_counts()

Out[578]:

23 456  
20 456  
16 456  
17 456  
18 456  
15 456  
12 456  
19 456  
13 456  
21 456  
14 456  
22 456  
6 455  
11 455  
10 455  
9 455  
7 455  
8 455  
0 455  
1 454  
5 452  
2 448  
4 442  
3 433  
Name: time, dtype: int64

In [579]:

*# Transforming Categorical features into numarical features*  
lb\_en **=** LabelEncoder()  
lb\_en1 **=** LabelEncoder()  
lb\_en2 **=** LabelEncoder()  
lb\_en3 **=** LabelEncoder()  
lb\_en4 **=** LabelEncoder()  
lb\_en5 **=** LabelEncoder()  
lb\_en6 **=** LabelEncoder()  
lb\_en7 **=** LabelEncoder()  
lb\_en8 **=** LabelEncoder()  
train**.**loc[:,'season'] **=** lb\_en**.**fit\_transform(train**.**loc[:,'season'])   
train**.**loc[:,'weather'] **=** lb\_en1**.**fit\_transform(train**.**loc[:,'weather'])   
train**.**loc[:,'temp'] **=** lb\_en2**.**fit\_transform(train**.**loc[:,'temp'])   
train**.**loc[:,'atemp'] **=** lb\_en3**.**fit\_transform(train**.**loc[:,'atemp'])   
train**.**loc[:,'humidity'] **=** lb\_en4**.**fit\_transform(train**.**loc[:,'humidity'])  
train**.**loc[:,'windspeed'] **=** lb\_en5**.**fit\_transform(train**.**loc[:,'windspeed'])   
train**.**loc[:,'Total\_booking'] **=** lb\_en6**.**fit\_transform(train**.**loc[:,'Total\_booking'])   
train**.**loc[:,'date'] **=** lb\_en7**.**fit\_transform(train**.**loc[:,'date'])   
train**.**loc[:,'time'] **=** lb\_en8**.**fit\_transform(train**.**loc[:,'time'])

In [580]:

train**.**head()

Out[580]:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **season** | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** | **date** | **time** |
| **0** | 2 | 0 | 1 | 0 | 26 | 33 | 68 | 7 | 503 | 305 | 19 |
| **1** | 0 | 0 | 1 | 0 | 34 | 43 | 70 | 8 | 4 | 384 | 4 |
| **2** | 1 | 0 | 1 | 0 | 6 | 7 | 41 | 10 | 138 | 12 | 9 |
| **3** | 3 | 0 | 1 | 0 | 16 | 21 | 20 | 3 | 208 | 207 | 16 |
| **4** | 0 | 0 | 1 | 0 | 36 | 43 | 42 | 8 | 183 | 164 | 13 |

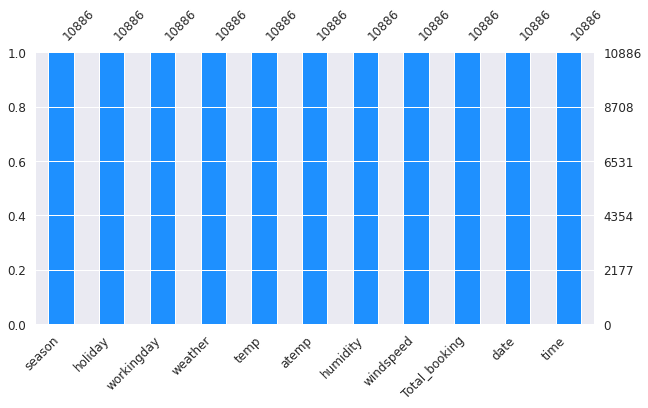
In [581]:

train**.**info()

<class 'pandas.core.frame.DataFrame'>  
Int64Index: 10886 entries, 0 to 2177  
Data columns (total 11 columns):  
 # Column Non-Null Count Dtype  
--- ------ -------------- -----  
 0 season 10886 non-null int64  
 1 holiday 10886 non-null int64  
 2 workingday 10886 non-null int64  
 3 weather 10886 non-null int64  
 4 temp 10886 non-null int64  
 5 atemp 10886 non-null int64  
 6 humidity 10886 non-null int64  
 7 windspeed 10886 non-null int64  
 8 Total\_booking 10886 non-null int64  
 9 date 10886 non-null int64  
 10 time 10886 non-null int64  
dtypes: int64(11)  
memory usage: 1020.6 KB

In [582]:

missingno**.**bar(train,color**=**"dodgerblue", sort**=**"ascending", figsize**=**(10,5), fontsize**=**12);



In [583]:

test **=** train[10886:]**.**drop('Total\_booking',axis**=**1)  
train **=** train[:10866]

In [584]:

test**.**info()

<class 'pandas.core.frame.DataFrame'>  
Int64Index: 0 entries  
Data columns (total 10 columns):  
 # Column Non-Null Count Dtype  
--- ------ -------------- -----  
 0 season 0 non-null int64  
 1 holiday 0 non-null int64  
 2 workingday 0 non-null int64  
 3 weather 0 non-null int64  
 4 temp 0 non-null int64  
 5 atemp 0 non-null int64  
 6 humidity 0 non-null int64  
 7 windspeed 0 non-null int64  
 8 date 0 non-null int64  
 9 time 0 non-null int64  
dtypes: int64(10)  
memory usage: 0.0 bytes

In [585]:

train**.**info()

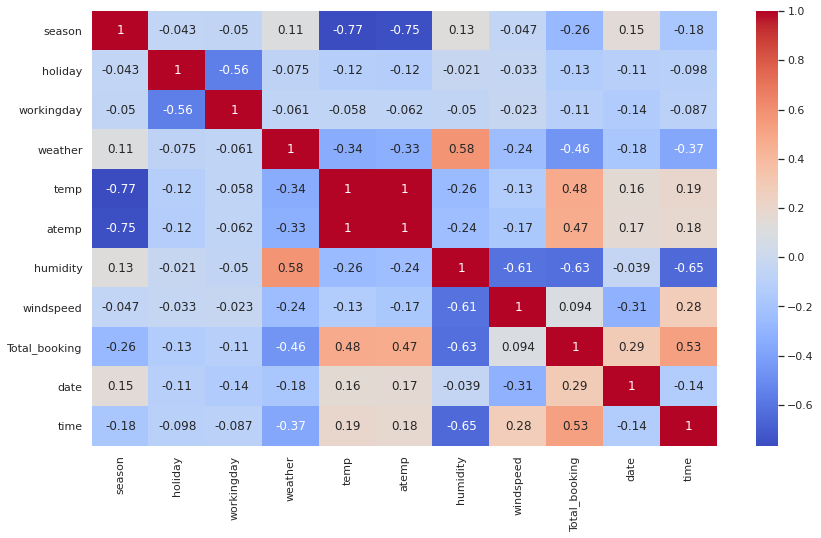
<class 'pandas.core.frame.DataFrame'>  
Int64Index: 10866 entries, 0 to 2157  
Data columns (total 11 columns):  
 # Column Non-Null Count Dtype  
--- ------ -------------- -----  
 0 season 10866 non-null int64  
 1 holiday 10866 non-null int64  
 2 workingday 10866 non-null int64  
 3 weather 10866 non-null int64  
 4 temp 10866 non-null int64  
 5 atemp 10866 non-null int64  
 6 humidity 10866 non-null int64  
 7 windspeed 10866 non-null int64  
 8 Total\_booking 10866 non-null int64  
 9 date 10866 non-null int64  
 10 time 10866 non-null int64  
dtypes: int64(11)  
memory usage: 1018.7 KB

In [586]:

a**=** train**.**corr()  
plt**.**figure(figsize**=**(14,8))  
sns**.**heatmap(a**.**corr(), annot**=True**, cmap**=**'coolwarm')

Out[586]:

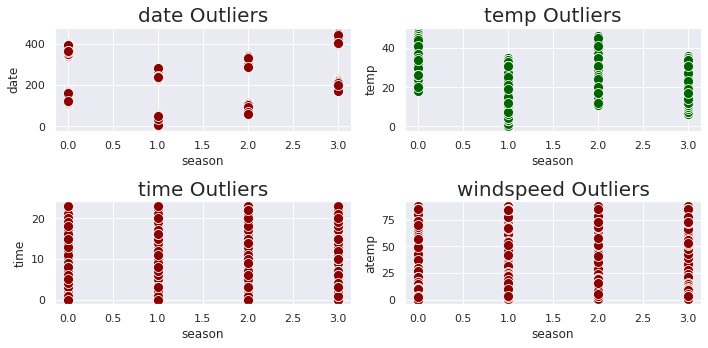
<matplotlib.axes.\_subplots.AxesSubplot at 0x7effcfde6198>



In [587]:

fig, ax **=** plt**.**subplots(nrows**=**2,ncols**=**2, figsize**=**(10,5))  
a **=** sns**.**scatterplot(x**=**'season', y**=**'date', data**=**train,ax**=**ax[0][0], color**=**'darkred', s**=**100)  
b **=** sns**.**scatterplot(x**=**'season', y**=**'temp', data**=**train,ax**=**ax[0][1], color**=**'darkgreen', s**=**100)  
c **=** sns**.**scatterplot(x**=**'season', y**=**'time', data**=**train,ax**=**ax[1][0], color**=**'darkred', s**=**100)  
d **=** sns**.**scatterplot(x**=**'season', y**=**'atemp', data**=**train,ax**=**ax[1][1], color**=**'darkred', s**=**100)  
e **=** sns**.**scatterplot(x**=**'season', y**=**'holiday', data**=**train,ax**=**ax[1][1], color**=**'darkred', s**=**100)  
f **=** sns**.**scatterplot(x**=**'season', y**=**'weather', data**=**train,ax**=**ax[1][1], color**=**'darkred', s**=**100)  
g **=** sns**.**scatterplot(x**=**'season', y**=**'humidity', data**=**train,ax**=**ax[1][1], color**=**'darkred', s**=**100)  
h **=** sns**.**scatterplot(x**=**'season', y**=**'windspeed', data**=**train,ax**=**ax[1][1], color**=**'darkred', s**=**100)  
a**.**set\_title('date Outliers', fontsize**=**20)  
b**.**set\_title('temp Outliers', fontsize**=**20)  
c**.**set\_title('time Outliers', fontsize**=**20)  
d**.**set\_title('atemp Outliers', fontsize**=**20)  
e**.**set\_title('holiday Outliers', fontsize**=**20)  
f**.**set\_title('weather Outliers', fontsize**=**20)  
g**.**set\_title('humidity Outliers', fontsize**=**20)  
h**.**set\_title('windspeed Outliers', fontsize**=**20)

plt**.**tight\_layout()



In [588]:

train['Total\_booking']**.**value\_counts()

Out[588]:

4 169  
3 149  
2 143  
5 135  
1 132  
 ...   
786 1  
754 1  
746 1  
738 1  
771 1  
Name: Total\_booking, Length: 787, dtype: int64

In [589]:

train**.**columns

Out[589]:

Index(['season', 'holiday', 'workingday', 'weather', 'temp', 'atemp',  
 'humidity', 'windspeed', 'Total\_booking', 'date', 'time'],  
 dtype='object')

In [640]:

X **=** train**.**iloc[:,1:**-**1] *# X value contains all the variables except labels*  
y **=** train**.**iloc[:,**-**1] *# these are the labels*

In [641]:

X

Out[641]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **holiday** | **workingday** | **weather** | **temp** | **atemp** | **humidity** | **windspeed** | **Total\_booking** | **date** |
| **0** | 0 | 1 | 0 | 26 | 33 | 68 | 7 | 503 | 305 |
| **1** | 0 | 1 | 0 | 34 | 43 | 70 | 8 | 4 | 384 |
| **2** | 0 | 1 | 0 | 6 | 7 | 41 | 10 | 138 | 12 |
| **3** | 0 | 1 | 0 | 16 | 21 | 20 | 3 | 208 | 207 |
| **4** | 0 | 1 | 0 | 36 | 43 | 42 | 8 | 183 | 164 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **2153** | 0 | 0 | 0 | 19 | 26 | 73 | 3 | 33 | 405 |
| **2154** | 0 | 0 | 0 | 12 | 17 | 51 | 3 | 151 | 49 |
| **2155** | 0 | 1 | 0 | 35 | 44 | 57 | 3 | 27 | 360 |
| **2156** | 0 | 0 | 3 | 17 | 22 | 53 | 3 | 35 | 202 |
| **2157** | 0 | 1 | 0 | 34 | 43 | 70 | 2 | 280 | 367 |

10866 rows × 9 columns

In [642]:

y

Out[642]:

0 19  
1 4  
2 9  
3 16  
4 13  
 ..  
2153 20  
2154 0  
2155 6  
2156 3  
2157 5  
Name: time, Length: 10866, dtype: int64

In [643]:

**from** sklearn.model\_selection **import** train\_test\_split  
X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X,y, test\_size**=**0.3)

In [644]:

**from** sklearn.preprocessing **import** MinMaxScaler  
mms **=** MinMaxScaler()  
X\_scaled **=** pd**.**DataFrame(mms**.**fit\_transform(X\_train), columns**=**X\_train**.**columns)  
X\_test\_scaled **=** pd**.**DataFrame(mms**.**transform(X\_test), columns**=**X\_test**.**columns)  
*# we have now fit and transform the data into a scaler for accurate reading and results.*

In [645]:

**from** imblearn.over\_sampling **import** SMOTE  
oversample **=** SMOTE()  
X\_balanced, y\_balanced **=** oversample**.**fit\_resample(X\_scaled, y\_train)  
X\_test\_balanced, y\_test\_balanced **=** oversample**.**fit\_resample(X\_test\_scaled, y\_test)  
*# we have addressed the issue of oversampling here*

In [646]:

y\_train**.**value\_counts()

Out[646]:

22 329  
6 328  
0 327  
11 326  
19 325  
18 325  
23 324  
20 322  
14 321  
5 321  
10 320  
2 319  
15 316  
17 315  
8 314  
16 314  
9 313  
1 313  
13 312  
12 309  
21 308  
4 307  
7 304  
3 294  
Name: time, dtype: int64

In [647]:

**from** sklearn.linear\_model **import** LogisticRegression  
**from** sklearn.neighbors **import** KNeighborsClassifier  
**from** sklearn.svm **import** SVC  
**from** sklearn.tree **import** DecisionTreeClassifier  
**from** sklearn.ensemble **import** RandomForestClassifier  
**from** xgboost **import** XGBClassifier  
**from** sklearn.neural\_network **import** MLPClassifier

In [648]:

classifiers **=** {  
 "LogisticRegression" : LogisticRegression(),  
 "KNeighbors" : KNeighborsClassifier(),  
 "SVC" : SVC(),  
 "DecisionTree" : DecisionTreeClassifier(),  
 "RandomForest" : RandomForestClassifier(),  
 "XGBoost" : XGBClassifier(),  
 "MLPClassifier" : MLPClassifier(solver**=**'lbfgs', alpha**=**1e-5,hidden\_layer\_sizes**=**(5, 2), random\_state**=**1, max\_iter**=**10000)  
}

In [654]:

**%%time**  
train\_scores **=** []  
test\_scores **=** []

**for** key, classifier **in** classifiers**.**items():  
 classifier**.**fit(X\_balanced, y\_balanced)  
 train\_score **=** classifier**.**score(X\_balanced, y\_balanced)  
 train\_scores**.**append(train\_score)  
 test\_score **=** classifier**.**score(X\_test\_balanced, y\_test\_balanced)  
 test\_scores**.**append(test\_score)

print(train\_scores)  
print(test\_scores)

[0.15260891590678824, 0.3699341438703141, 0.20263424518743667, 0.999113475177305, 0.999113475177305, 0.3290273556231003, 0.041666666666666664]  
[0.1361111111111111, 0.11277777777777778, 0.15138888888888888, 0.16194444444444445, 0.1686111111111111, 0.18472222222222223, 0.041666666666666664]  
CPU times: user 25.1 s, sys: 1.76 s, total: 26.9 s  
Wall time: 25 s

In [655]:

print(classification\_report(y\_test\_balanced, prediction))

precision recall f1-score support

0 0.16 0.15 0.15 150  
 1 0.21 0.25 0.22 150  
 2 0.17 0.13 0.15 150  
 3 0.24 0.31 0.27 150  
 4 0.29 0.43 0.35 150  
 5 0.25 0.29 0.27 150  
 6 0.16 0.19 0.17 150  
 7 0.19 0.19 0.19 150  
 8 0.43 0.39 0.41 150  
 9 0.18 0.36 0.24 150  
 10 0.14 0.13 0.14 150  
 11 0.09 0.05 0.07 150  
 12 0.14 0.08 0.10 150  
 13 0.07 0.05 0.05 150  
 14 0.13 0.11 0.12 150  
 15 0.11 0.09 0.10 150  
 16 0.16 0.24 0.19 150  
 17 0.23 0.20 0.21 150  
 18 0.21 0.20 0.21 150  
 19 0.08 0.06 0.07 150  
 20 0.14 0.07 0.09 150  
 21 0.10 0.07 0.08 150  
 22 0.13 0.13 0.13 150  
 23 0.18 0.25 0.21 150

accuracy 0.18 3600  
 macro avg 0.18 0.18 0.18 3600  
weighted avg 0.18 0.18 0.18 3600

In [656]:

**%%time**  
xgb **=** XGBClassifier()  
model **=** xgb**.**fit(X\_balanced, y\_balanced)  
prediction **=** xgb**.**predict(X\_test\_balanced)

CPU times: user 8.08 s, sys: 20 ms, total: 8.1 s  
Wall time: 8.12 s

In [657]:

**from** sklearn.metrics **import** classification\_report  
print(classification\_report(y\_test\_balanced, prediction))

precision recall f1-score support

0 0.16 0.15 0.15 150  
 1 0.21 0.25 0.22 150  
 2 0.17 0.13 0.15 150  
 3 0.24 0.31 0.27 150  
 4 0.29 0.43 0.35 150  
 5 0.25 0.29 0.27 150  
 6 0.16 0.19 0.17 150  
 7 0.19 0.19 0.19 150  
 8 0.43 0.39 0.41 150  
 9 0.18 0.36 0.24 150  
 10 0.14 0.13 0.14 150  
 11 0.09 0.05 0.07 150  
 12 0.14 0.08 0.10 150  
 13 0.07 0.05 0.05 150  
 14 0.13 0.11 0.12 150  
 15 0.11 0.09 0.10 150  
 16 0.16 0.24 0.19 150  
 17 0.23 0.20 0.21 150  
 18 0.21 0.20 0.21 150  
 19 0.08 0.06 0.07 150  
 20 0.14 0.07 0.09 150  
 21 0.10 0.07 0.08 150  
 22 0.13 0.13 0.13 150  
 23 0.18 0.25 0.21 150

accuracy 0.18 3600  
 macro avg 0.18 0.18 0.18 3600  
weighted avg 0.18 0.18 0.18 3600

In [658]:

**%%time**  
lr**=**LogisticRegression()  
model **=** lr**.**fit(X\_balanced, y\_balanced)  
prediction **=** lr**.**predict(X\_test\_balanced)

CPU times: user 2.06 s, sys: 1.63 s, total: 3.69 s  
Wall time: 1.89 s

In [659]:

**%%time**  
**from** sklearn.metrics **import** classification\_report, confusion\_matrix  
print(confusion\_matrix(y\_test\_balanced, prediction))  
print(classification\_report(y\_test\_balanced, prediction))

[[ 4 13 18 28 33 5 9 5 3 0 5 1 0 0 7 2 0 1 0 0 1 2 0 13]  
 [ 2 12 13 41 29 1 10 3 9 1 3 1 2 1 6 2 0 3 1 1 0 1 2 6]  
 [ 1 5 12 59 37 2 4 5 6 0 5 0 0 0 8 0 1 0 0 0 1 1 2 1]  
 [ 1 10 12 57 51 0 3 4 10 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0]  
 [ 0 2 14 66 48 0 5 3 3 0 0 4 0 0 3 0 0 0 0 0 0 1 1 0]  
 [ 1 10 17 41 60 1 3 4 7 0 1 0 0 0 1 1 0 1 1 0 0 0 1 0]  
 [ 3 2 6 36 16 9 31 16 4 4 2 4 0 0 5 4 0 0 0 0 1 1 3 3]  
 [ 0 1 1 27 5 1 9 20 41 10 1 1 3 2 3 0 3 6 0 7 4 2 2 1]  
 [ 0 1 1 19 5 1 8 9 58 1 6 1 0 0 3 1 0 22 4 1 0 3 1 5]  
 [ 4 1 2 3 0 3 2 24 13 16 10 3 7 6 5 9 5 4 5 10 12 3 2 1]  
 [ 5 0 3 6 2 2 15 16 6 5 6 13 3 7 10 14 1 3 0 6 7 3 16 1]  
 [ 7 1 4 0 3 2 5 13 5 4 1 11 4 10 11 16 4 18 6 4 8 7 4 2]  
 [ 1 1 1 3 4 1 1 6 6 5 2 12 3 11 21 27 7 16 4 6 9 2 1 0]  
 [ 2 1 2 3 1 1 1 4 2 4 3 12 2 6 20 26 9 19 8 5 13 4 1 1]  
 [ 3 1 0 2 1 0 4 3 4 4 6 9 2 6 23 45 8 15 1 3 3 2 4 1]  
 [ 0 0 2 7 4 3 1 6 6 6 3 9 3 3 22 36 10 15 3 2 6 3 0 0]  
 [ 1 1 1 2 1 0 6 5 4 7 0 2 4 9 18 22 29 16 5 8 5 2 2 0]  
 [ 0 1 1 1 2 0 1 11 16 1 0 1 0 0 7 11 8 66 12 4 0 7 0 0]  
 [ 2 1 0 1 3 0 4 4 28 1 2 1 1 4 9 6 8 54 9 7 1 2 2 0]  
 [ 1 2 0 7 3 0 2 10 19 5 1 1 4 6 8 12 10 21 11 11 4 6 2 4]  
 [ 1 2 1 6 7 1 4 17 8 7 4 5 3 8 6 10 18 7 2 17 8 5 2 1]  
 [ 0 4 3 5 4 1 20 13 4 4 8 3 4 1 10 12 7 2 0 3 20 6 9 7]  
 [ 2 3 1 8 6 1 20 6 6 8 8 5 1 4 17 10 0 0 3 4 6 9 11 11]  
 [12 8 9 16 23 8 24 9 3 0 4 0 1 0 7 4 0 2 1 0 1 4 8 6]]  
 precision recall f1-score support

0 0.08 0.03 0.04 150  
 1 0.14 0.08 0.10 150  
 2 0.10 0.08 0.09 150  
 3 0.13 0.38 0.19 150  
 4 0.14 0.32 0.19 150  
 5 0.02 0.01 0.01 150  
 6 0.16 0.21 0.18 150  
 7 0.09 0.13 0.11 150  
 8 0.21 0.39 0.28 150  
 9 0.17 0.11 0.13 150  
 10 0.07 0.04 0.05 150  
 11 0.11 0.07 0.09 150  
 12 0.06 0.02 0.03 150  
 13 0.07 0.04 0.05 150  
 14 0.10 0.15 0.12 150  
 15 0.13 0.24 0.17 150  
 16 0.23 0.19 0.21 150  
 17 0.23 0.44 0.30 150  
 18 0.12 0.06 0.08 150  
 19 0.11 0.07 0.09 150  
 20 0.07 0.05 0.06 150  
 21 0.08 0.04 0.05 150  
 22 0.14 0.07 0.10 150  
 23 0.09 0.04 0.06 150

accuracy 0.14 3600  
 macro avg 0.12 0.14 0.12 3600  
weighted avg 0.12 0.14 0.12 3600

CPU times: user 21.8 ms, sys: 0 ns, total: 21.8 ms  
Wall time: 22 ms

In [660]:

**%%time**  
su**=**SVC()  
model **=** su**.**fit(X\_balanced, y\_balanced)  
prediction **=** su**.**predict(X\_test\_balanced)

CPU times: user 5.93 s, sys: 2.98 ms, total: 5.94 s  
Wall time: 5.95 s

In [661]:

print(confusion\_matrix(y\_test\_balanced, prediction))  
print(classification\_report(y\_test\_balanced, prediction))

[[19 14 10 21 17 26 1 5 1 2 2 0 0 2 1 2 0 1 2 0 0 7 4 13]  
 [15 15 11 29 16 20 6 6 4 2 2 2 2 1 1 1 0 4 0 2 0 2 3 6]  
 [10 25 4 38 24 18 2 4 1 0 4 2 0 0 2 1 0 0 0 3 1 2 3 6]  
 [ 7 19 10 38 38 19 4 5 5 2 2 0 0 0 1 0 0 0 0 0 0 0 0 0]  
 [ 4 21 9 48 28 21 4 3 2 0 1 3 1 0 1 0 0 0 0 0 0 1 1 2]  
 [ 7 17 13 40 35 20 1 1 6 2 1 2 0 0 1 0 0 0 0 0 0 0 1 3]  
 [ 4 2 6 39 6 19 29 6 5 5 9 4 0 1 2 4 0 0 0 0 0 4 3 2]  
 [ 0 3 2 32 3 1 10 36 5 15 1 2 4 2 1 0 4 0 0 11 10 4 3 1]  
 [ 0 1 2 24 0 3 2 13 56 4 2 3 2 0 1 1 0 12 2 4 0 7 3 8]  
 [ 1 4 0 7 0 4 7 17 0 26 13 6 3 3 2 5 4 3 1 5 16 12 7 4]  
 [ 2 1 1 7 0 6 16 4 0 8 21 21 4 8 7 12 0 1 0 2 1 4 21 3]  
 [ 3 1 0 3 1 3 10 3 4 10 14 7 14 19 7 12 2 4 5 3 7 9 8 1]  
 [ 1 1 1 4 1 1 3 3 3 5 11 12 5 26 11 21 6 7 3 3 11 4 6 1]  
 [ 3 0 1 2 0 3 2 0 0 2 13 15 6 21 16 25 9 3 6 3 9 8 2 1]  
 [ 1 0 0 5 1 4 4 1 1 4 10 10 4 21 5 41 11 6 1 3 7 4 6 0]  
 [ 3 1 2 5 0 6 3 2 2 3 11 11 15 13 8 32 10 5 5 2 5 4 1 1]  
 [ 1 2 1 5 0 0 4 5 0 6 5 5 4 13 14 12 40 10 4 2 11 3 2 1]  
 [ 4 2 0 0 0 0 1 9 13 10 6 2 3 10 4 9 6 52 12 2 1 4 0 0]  
 [ 1 0 0 1 1 0 3 12 16 2 11 2 3 11 3 10 8 39 10 7 2 6 1 1]  
 [ 4 0 0 8 2 0 0 18 10 12 4 3 3 4 8 5 19 6 3 21 8 9 1 2]  
 [ 3 0 0 7 4 2 4 9 1 11 12 4 2 3 3 5 19 3 4 7 19 11 10 7]  
 [ 6 6 2 7 3 5 14 6 0 9 8 5 2 3 0 8 8 0 2 1 18 11 14 12]  
 [ 3 2 0 14 2 4 15 5 2 4 14 4 2 8 3 8 0 0 0 5 10 19 15 11]  
 [ 9 4 5 22 13 16 17 2 4 2 6 2 1 1 3 3 0 0 0 0 0 10 15 15]]  
 precision recall f1-score support

0 0.17 0.13 0.15 150  
 1 0.11 0.10 0.10 150  
 2 0.05 0.03 0.03 150  
 3 0.09 0.25 0.14 150  
 4 0.14 0.19 0.16 150  
 5 0.10 0.13 0.11 150  
 6 0.18 0.19 0.19 150  
 7 0.21 0.24 0.22 150  
 8 0.40 0.37 0.38 150  
 9 0.18 0.17 0.18 150  
 10 0.11 0.14 0.13 150  
 11 0.06 0.05 0.05 150  
 12 0.06 0.03 0.04 150  
 13 0.12 0.14 0.13 150  
 14 0.05 0.03 0.04 150  
 15 0.15 0.21 0.17 150  
 16 0.27 0.27 0.27 150  
 17 0.33 0.35 0.34 150  
 18 0.17 0.07 0.10 150  
 19 0.24 0.14 0.18 150  
 20 0.14 0.13 0.13 150  
 21 0.08 0.07 0.07 150  
 22 0.12 0.10 0.11 150  
 23 0.15 0.10 0.12 150

accuracy 0.15 3600  
 macro avg 0.15 0.15 0.15 3600  
weighted avg 0.15 0.15 0.15 3600

In [662]:

rf**=**RandomForestClassifier()  
model **=** rf**.**fit(X\_balanced, y\_balanced)  
prediction **=** rf**.**predict(X\_test\_balanced)

In [663]:

print(confusion\_matrix(y\_test\_balanced, prediction))  
print(classification\_report(y\_test\_balanced, prediction))

[[29 24 5 4 3 28 3 2 5 2 2 2 0 1 0 0 0 0 2 2 3 2 9 22]  
 [26 29 28 2 7 15 4 5 11 1 2 0 1 0 1 0 0 2 3 0 1 0 5 7]  
 [10 50 12 24 18 5 2 6 6 2 1 0 0 0 0 0 0 0 1 2 1 1 4 5]  
 [ 2 15 13 35 46 3 12 6 4 3 1 0 0 0 0 0 0 1 0 3 1 1 0 4]  
 [ 3 3 21 39 34 26 7 5 2 0 3 0 0 0 0 0 0 0 0 1 0 0 2 4]  
 [19 14 6 3 21 61 10 2 6 0 2 1 0 0 0 0 1 1 0 1 0 1 1 0]  
 [ 1 2 1 14 11 13 37 9 5 10 10 3 1 1 2 3 0 0 0 1 0 4 12 10]  
 [ 3 5 5 13 2 2 9 32 7 20 2 1 4 3 1 0 3 1 1 16 11 1 4 4]  
 [11 6 5 4 1 3 5 12 61 6 1 1 1 1 0 0 0 2 10 5 3 4 3 5]  
 [ 1 3 2 0 4 0 4 13 10 30 14 2 7 7 3 1 4 3 3 6 9 15 6 3]  
 [ 1 2 0 0 2 0 11 6 2 5 23 36 11 2 8 2 1 1 3 5 3 7 17 2]  
 [ 5 0 1 1 0 1 7 1 2 3 17 9 28 14 11 6 7 7 5 6 5 6 6 2]  
 [ 1 4 4 2 0 1 1 4 1 9 5 19 12 27 14 11 6 3 8 4 4 7 2 1]  
 [ 3 1 0 0 2 2 3 0 0 3 5 10 27 9 29 21 9 5 5 2 4 6 3 1]  
 [ 2 4 0 0 1 2 1 1 1 3 8 10 13 36 6 25 10 3 7 2 5 4 1 5]  
 [ 0 3 1 3 1 0 0 0 2 3 5 6 17 27 27 6 18 12 4 3 0 6 3 3]  
 [ 0 2 0 0 1 0 2 3 0 2 4 6 6 14 17 19 34 9 11 9 4 5 0 2]  
 [ 2 0 1 0 0 1 1 3 7 6 0 8 3 2 7 5 16 16 58 10 1 1 0 2]  
 [ 0 1 0 1 0 0 0 5 17 1 8 5 3 4 5 2 8 32 28 19 7 2 1 1]  
 [ 4 2 1 2 3 0 0 10 7 9 7 1 4 1 1 5 17 5 22 26 14 1 5 3]  
 [ 2 2 1 1 0 0 1 8 3 17 3 7 5 5 1 2 14 3 8 17 18 24 6 2]  
 [ 6 1 2 2 2 4 5 3 1 18 6 8 4 4 3 5 3 1 0 2 22 12 25 11]  
 [ 4 4 1 2 1 0 11 3 7 5 19 2 6 0 4 3 0 3 1 0 4 35 19 16]  
 [24 5 4 3 1 3 15 5 2 3 6 6 1 1 0 1 0 0 1 1 7 5 30 26]]  
 precision recall f1-score support

0 0.18 0.19 0.19 150  
 1 0.16 0.19 0.17 150  
 2 0.11 0.08 0.09 150  
 3 0.23 0.23 0.23 150  
 4 0.21 0.23 0.22 150  
 5 0.36 0.41 0.38 150  
 6 0.25 0.25 0.25 150  
 7 0.22 0.21 0.22 150  
 8 0.36 0.41 0.38 150  
 9 0.19 0.20 0.19 150  
 10 0.15 0.15 0.15 150  
 11 0.06 0.06 0.06 150  
 12 0.08 0.08 0.08 150  
 13 0.06 0.06 0.06 150  
 14 0.04 0.04 0.04 150  
 15 0.05 0.04 0.04 150  
 16 0.23 0.23 0.23 150  
 17 0.15 0.11 0.12 150  
 18 0.15 0.19 0.17 150  
 19 0.18 0.17 0.18 150  
 20 0.14 0.12 0.13 150  
 21 0.08 0.08 0.08 150  
 22 0.12 0.13 0.12 150  
 23 0.18 0.17 0.18 150

accuracy 0.17 3600  
 macro avg 0.16 0.17 0.17 3600  
weighted avg 0.16 0.17 0.17 3600

In [622]:

**%%time**  
dt **=** DecisionTreeClassifier(random\_state**=**0)  
model **=** dt**.**fit(X\_balanced, y\_balanced)  
prediction **=** dt**.**predict(X\_test\_balanced)

CPU times: user 72.3 ms, sys: 11 ms, total: 83.3 ms  
Wall time: 85.9 ms

In [610]:

**from** sklearn.preprocessing **import** StandardScaler  
feature\_scaler **=** StandardScaler()  
X\_train **=** feature\_scaler**.**fit\_transform(X\_train)  
X\_test **=** feature\_scaler**.**transform(X\_test)

In [611]:

**from** sklearn.ensemble **import** RandomForestClassifier  
classifier **=** RandomForestClassifier(n\_estimators**=**300, random\_state**=**0)

In [612]:

**from** sklearn.model\_selection **import** cross\_val\_score  
all\_accuracies **=** cross\_val\_score(estimator**=**classifier, X**=**X\_train, y**=**y\_train, cv**=**5)

In [613]:

grid\_param **=** {  
 'n\_estimators': [100, 300, 500, 800, 1000],  
 'criterion': ['gini', 'entropy'],  
 'bootstrap': [**True**, **False**]  
}

In [616]:

**from** sklearn.model\_selection **import** GridSearchCV  
gd\_sr **=** GridSearchCV(estimator**=**classifier,  
 param\_grid**=**grid\_param,  
 scoring**=**'accuracy',  
 cv**=**5,  
 n\_jobs**=-**1)

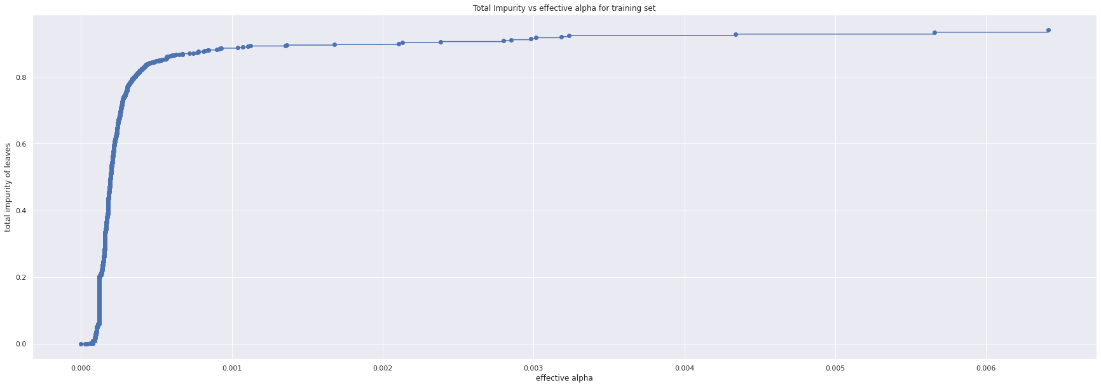
In [629]:

dt2 **=** DecisionTreeClassifier(random\_state**=**0)  
path **=** dt2**.**cost\_complexity\_pruning\_path(X\_balanced, y\_balanced)  
model **=** dt2**.**fit(X\_balanced, y\_balanced)  
ccp\_alphas, impurities **=** path**.**ccp\_alphas, path**.**impurities

In [630]:

**%%time**  
fig, ax **=** plt**.**subplots()  
ax**.**plot(ccp\_alphas[:**-**1], impurities[:**-**1], marker**=**'o', drawstyle**=**"steps-post")  
ax**.**set\_xlabel("effective alpha")  
ax**.**set\_ylabel("total impurity of leaves")  
ax**.**set\_title("Total Impurity vs effective alpha for training set")

CPU times: user 18 ms, sys: 1.99 ms, total: 20 ms  
Wall time: 21.8 ms



In [666]:

model **=** linear\_model**.**LinearRegression()  
parameters **=** {'fit\_intercept':[**True**,**False**], 'normalize':[**True**,**False**], 'copy\_X':[**True**, **False**]}  
grid **=** GridSearchCV(model,parameters, cv**=None**)  
grid**.**fit(X\_train, y\_train)  
print("r2 / variance : ", grid**.**best\_score\_)  
print("Residual sum of squares: %.2f"  
 **%** np**.**mean((grid**.**predict(X\_test) **-** y\_test) **\*\*** 2))

r2 / variance : 0.15648035213197822  
Residual sum of squares: 39.49

In [668]:

rfc**=**RandomForestClassifier(random\_state**=**42)

In [669]:

param\_grid **=** {   
 'n\_estimators': [200, 500],  
 'max\_features': ['auto', 'sqrt', 'log2'],  
 'max\_depth' : [4,5,6,7,8],  
 'criterion' :['gini', 'entropy']  
}

In [670]:

CV\_rfc **=** GridSearchCV(estimator**=**rfc, param\_grid**=**param\_grid, cv**=** 5)  
CV\_rfc**.**fit(X\_train, y\_train)

Out[670]:

GridSearchCV(cv=5, error\_score=nan,  
 estimator=RandomForestClassifier(bootstrap=True, ccp\_alpha=0.0,  
 class\_weight=None,  
 criterion='gini', max\_depth=None,  
 max\_features='auto',  
 max\_leaf\_nodes=None,  
 max\_samples=None,  
 min\_impurity\_decrease=0.0,  
 min\_impurity\_split=None,  
 min\_samples\_leaf=1,  
 min\_samples\_split=2,  
 min\_weight\_fraction\_leaf=0.0,  
 n\_estimators=100, n\_jobs=None,  
 oob\_score=False, random\_state=42,  
 verbose=0, warm\_start=False),  
 iid='deprecated', n\_jobs=None,  
 param\_grid={'criterion': ['gini', 'entropy'],  
 'max\_depth': [4, 5, 6, 7, 8],  
 'max\_features': ['auto', 'sqrt', 'log2'],  
 'n\_estimators': [200, 500]},  
 pre\_dispatch='2\*n\_jobs', refit=True, return\_train\_score=False,  
 scoring=None, verbose=0)

In [671]:

CV\_rfc**.**best\_params\_

Out[671]:

{'criterion': 'entropy',  
 'max\_depth': 8,  
 'max\_features': 'auto',  
 'n\_estimators': 500}

In [672]:

rfc1**=**RandomForestClassifier(random\_state**=**42, max\_features**=**'auto', n\_estimators**=** 500, max\_depth**=**8, criterion**=**'gini')

In [674]:

rfc1**.**fit(X\_train, y\_train)

Out[674]:

RandomForestClassifier(bootstrap=True, ccp\_alpha=0.0, class\_weight=None,  
 criterion='gini', max\_depth=8, max\_features='auto',  
 max\_leaf\_nodes=None, max\_samples=None,  
 min\_impurity\_decrease=0.0, min\_impurity\_split=None,  
 min\_samples\_leaf=1, min\_samples\_split=2,  
 min\_weight\_fraction\_leaf=0.0, n\_estimators=500,  
 n\_jobs=None, oob\_score=False, random\_state=42, verbose=0,  
 warm\_start=False)

In [675]:

pred**=**rfc1**.**predict(X\_test)

In [676]:

print("Accuracy for Random Forest on CV data: ",accuracy\_score(y\_test,pred))

Accuracy for Random Forest on CV data: 0.19355828220858895

In [678]:

**import** xgboost **as** xgb  
model **=** xgb**.**XGBRegressor()

model, pred **=** algorithm\_pipeline(X\_train, X\_test, y\_train, y\_test, model,   
 param\_grid, cv**=**5)

Fitting 5 folds for each of 60 candidates, totalling 300 fits

[Parallel(n\_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.  
[Parallel(n\_jobs=-1)]: Done 37 tasks | elapsed: 39.3s  
[Parallel(n\_jobs=-1)]: Done 158 tasks | elapsed: 4.2min  
[Parallel(n\_jobs=-1)]: Done 300 out of 300 | elapsed: 8.1min finished

[00:59:37] WARNING: /workspace/src/objective/regression\_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.

In [680]:

print(np**.**sqrt(**-**model**.**best\_score\_))  
print(model**.**best\_params\_)

5.504082406620877  
{'criterion': 'gini', 'max\_depth': 6, 'max\_features': 'auto', 'n\_estimators': 200}

*From <*[*https://notebooks.githubusercontent.com/view/ipynb?browser=chrome&color\_mode=auto&commit=cbc228f3b0f42bff122ff0d34c3c7cd06560cb2a&device=unknown\_device&enc\_url=68747470733a2f2f7261772e67697468756275736572636f6e74656e742e636f6d2f736169736831352f4d616368696e652d4c6561726e696e672f636263323238663362306634326266663132326666306433346333633763643036353630636232612f4361625f426f6f6b696e672e6970796e62&logged\_in=true&nwo=saish15%2FMachine-Learning&path=Cab\_Booking.ipynb&platform=windows&repository\_id=231844062&repository\_type=Repository&version=99#6fe1b8c9-9947-4a96-8e8a-b68ed9d02f17*](https://notebooks.githubusercontent.com/view/ipynb?browser=chrome&color_mode=auto&commit=cbc228f3b0f42bff122ff0d34c3c7cd06560cb2a&device=unknown_device&enc_url=68747470733a2f2f7261772e67697468756275736572636f6e74656e742e636f6d2f736169736831352f4d616368696e652d4c6561726e696e672f636263323238663362306634326266663132326666306433346333633763643036353630636232612f4361625f426f6f6b696e672e6970796e62&logged_in=true&nwo=saish15%2FMachine-Learning&path=Cab_Booking.ipynb&platform=windows&repository_id=231844062&repository_type=Repository&version=99#6fe1b8c9-9947-4a96-8e8a-b68ed9d02f17)*>*

s