. Business Case: Yulu - Hypothesis Testing

About Yulu

Yulu is India's leading micro-mobility service provider, which offers unique vehicles for the daily commute. Starting off as a mission to eliminate traffic congestion in India, Yulu provides the safest commute solution through a user-friendly mobile app to enable shared, solo and sustainable commuting.

Yulu zones are located at all the appropriate locations (including metro stations, bus stands, office spaces, residential areas, corporate offices, etc) to make those first and last miles smooth, affordable, and convenient!

Yulu has recently suffered considerable dips in its revenues. They have contracted a consulting company to understand the factors on which the demand for these shared electric cycles depends. Specifically, they want to understand the factors affecting the demand for these shared electric cycles in the Indian market.

How you can help here?

The company wants to know:

- **Which variables are significant in predicting the demand for shared electric cycles in the Indian market?**
- **How well those variables describe the electric cycle demands**

In [122…
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import chi2
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import LabelEncoder
```

In [123…
```
df=pd.read_csv('yulu_data.csv')
```

In [124…
```
df
```

Out[124]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0000	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	7	329	336
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	10	231	241
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	4	164	168
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	12	117	129
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	4	84	88

10886 rows × 12 columns

In [125…
```
df.head()
```

Out[125]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0	0	1	1

In [126…
```
df.tail()
```

Out[126]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	7	329	336
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	10	231	241
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	4	164	168
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	12	117	129
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	4	84	88

In [127…
```
df.shape
```
Out[127]:
```
(10886, 12)
```

In [128…
```
df.size
```
Out[128]:
```
130632
```

In [129…
```
df.columns
```
Out[129]:
```
Index(['datetime', 'season', 'holiday', 'workingday', 'weather', 'temp',
       'atemp', 'humidity', 'windspeed', 'casual', 'registered', 'count'],
      dtype='object')
```

In [130…
```
df.dtypes
```
Out[130]:
```
datetime       object
season          int64
holiday         int64
workingday      int64
weather         int64
temp          float64
atemp         float64
humidity        int64
windspeed     float64
casual          int64
registered      int64
count           int64
dtype: object
```

In [131…
```
df.isnull().sum()
```
Out[131]:
```
datetime      0
season        0
holiday       0
workingday    0
weather       0
temp          0
atemp         0
humidity      0
windspeed     0
casual        0
registered    0
count         0
dtype: int64
```

There is no null values in the dataset

In [132…
```
df.info()
```
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   datetime    10886 non-null  object
 1   season      10886 non-null  int64
 2   holiday     10886 non-null  int64
 3   workingday  10886 non-null  int64
 4   weather     10886 non-null  int64
 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   casual      10886 non-null  int64
 10  registered  10886 non-null  int64
 11  count       10886 non-null  int64
dtypes: float64(3), int64(8), object(1)
memory usage: 1020.7+ KB
```

Date and time shown in object type ,need to change in to date and time format

In [133…
```
df.describe()
```

Out[133]:

	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	re
count	10886.000000	10886.000000	10886.000000	10886.000000	10886.00000	10886.000000	10886.000000	10886.000000	10886.000000	1088
mean	2.506614	0.028569	0.680875	1.418427	20.23086	23.655084	61.886460	12.799395	36.021955	15
std	1.116174	0.166599	0.466159	0.633839	7.79159	8.474601	19.245033	8.164537	49.960477	15
min	1.000000	0.000000	0.000000	1.000000	0.82000	0.760000	0.000000	0.000000	0.000000	
25%	2.000000	0.000000	0.000000	1.000000	13.94000	16.665000	47.000000	7.001500	4.000000	3
50%	3.000000	0.000000	1.000000	1.000000	20.50000	24.240000	62.000000	12.998000	17.000000	11
75%	4.000000	0.000000	1.000000	2.000000	26.24000	31.060000	77.000000	16.997900	49.000000	22
max	4.000000	1.000000	1.000000	4.000000	41.00000	45.455000	100.000000	56.996900	367.000000	88

mean and median of casual and registered are very far away to one another ,it showing there is a possibility of outlier values

In [134…
```
df['datetime']=pd.to_datetime(df['datetime'])
```

In [135…
```
df
```

Out[135]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0000	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	7	329	336
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	10	231	241
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	4	164	168
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	12	117	129
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	4	84	88

10886 rows × 12 columns

In [136…
```
df.dtypes
```

Out[136]:
```
datetime      datetime64[ns]
season                 int64
holiday                int64
workingday             int64
weather                int64
temp                 float64
atemp                float64
humidity               int64
windspeed            float64
casual                 int64
registered             int64
count                  int64
dtype: object
```

Date and time changed in to date and time format

In [137…
```
for i in df.columns:
    print(i,':',df[i].nunique())
```

```
datetime : 10886
season : 4
holiday : 2
workingday : 2
weather : 4
temp : 49
atemp : 60
humidity : 89
windspeed : 28
casual : 309
registered : 731
count : 822
```

In [138…
```
for i in df.columns:
    print(i,':\n',df[i].unique())
```

```
datetime :
 ['2011-01-01T00:00:00.000000000' '2011-01-01T01:00:00.000000000'
 '2011-01-01T02:00:00.000000000' ... '2012-12-19T21:00:00.000000000'
 '2012-12-19T22:00:00.000000000' '2012-12-19T23:00:00.000000000']
```

```
season :
 [1 2 3 4]
holiday :
 [0 1]
workingday :
 [0 1]
weather :
 [1 2 3 4]
temp :
 [ 9.84  9.02  8.2  13.12 15.58 14.76 17.22 18.86 18.04 16.4  13.94 12.3
 10.66  6.56  5.74  7.38  4.92 11.48  4.1   3.28  2.46 21.32 22.96 23.78
 24.6  19.68 22.14 20.5  27.06 26.24 25.42 27.88 28.7  30.34 31.16 29.52
 33.62 35.26 36.9  32.8  31.98 34.44 36.08 37.72 38.54  1.64  0.82 39.36
 41.  ]
atemp :
 [14.395 13.635 12.88  17.425 19.695 16.665 21.21  22.725 21.97  20.455
 11.365 10.605  9.85   8.335  6.82   5.305  6.06   9.09  12.12   7.575
 15.91   3.03   3.79   4.545 15.15  18.18  25.    26.515 27.275 29.545
 23.485 25.76  31.06  30.305 24.24  18.94  31.82  32.575 33.335 28.79
 34.85  35.605 37.12  40.15  41.665 40.91  39.395 34.09  28.03  36.365
 37.88  42.425 43.94  38.635  1.515  0.76   2.275 43.18  44.695 45.455]
humidity :
 [ 81  80  75  86  76  77  72  82  88  87  94 100  71  66  57  46  42  39
  44  47  50  43  40  35  30  32  64  69  55  59  63  68  74  51  56  52
  49  48  37  33  28  38  36  93  29  53  34  54  41  45  92  62  58  61
  60  65  70  27  25  26  31  73  21  24  23  22  19  15  67  10   8  12
  14  13  17  16  18  20  85   0  83  84  78  79  89  97  90  96  91]
windspeed :
 [ 0.      6.0032 16.9979 19.0012 19.9995 12.998  15.0013  8.9981 11.0014
 22.0028 30.0026 23.9994 27.9993 26.0027  7.0015 32.9975 36.9974 31.0009
 35.0008 39.0007 43.9989 40.9973 51.9987 46.0022 50.0021 43.0006 56.9969
 47.9988]
casual :
 [  3   8   5   0   2   1  12  26  29  47  35  40  41  15   9   6  11   4
   7  16  20  19  10  13  14  18  17  21  33  23  22  28  48  52  42  24
  30  27  32  58  62  51  25  31  59  45  73  55  68  34  38 102  84  39
  36  43  46  60  80  83  74  37  70  81 100  99  54  88  97 144 149 124
  98  50  72  57  71  67  95  90 126 174 168 170 175 138  92  56 111  89
  69 139 166 219 240 147 148  78  53  63  79 114  94  85 128  93 121 156
 135 103  44  49  64  91 119 167 181 179 161 143  75  66 109 123 113  65
  86  82 132 129 196 142 122 106  61 107 120 195 183 206 158 137  76 115
 150 188 193 180 127 154 108  96 110 112 169 131 176 134 162 153 210 118
 141 146 159 178 177 136 215 198 248 225 194 237 242 235 224 236 222  77
  87 101 145 182 171 160 133 105 104 187 221 201 205 234 185 164 200 130
 155 116 125 204 186 214 245 218 217 152 191 256 251 262 189 212 272 223
 208 165 229 151 117 199 140 226 286 352 357 367 291 233 190 283 295 232
 173 184 172 320 355 326 321 354 299 227 254 260 207 274 308 288 311 253
 197 163 275 298 282 266 220 241 230 157 293 257 269 255 228 276 332 361
 356 331 279 203 250 259 297 265 267 192 239 238 213 264 244 243 246 289
 287 209 263 249 247 284 327 325 312 350 258 362 310 317 268 202 294 280
 216 292 304]
registered :
 [ 13  32  27  10   1   0   2   7   6  24  30  55  47  71  70  52  26  31
  25  17  16   8   4  19  46  54  73  64  67  58  43  29  20   9   5   3
  63 153  81  33  41  48  53  66 146 148 102  49  11  36  92 177  98  37
  50  79  68 202 179 110  34  87 192 109  74  65  85 186 166 127  82  40
  18  95 216 116  42  57  78  59 163 158  51  76 190 125 178  39  14  15
  56  60  90  83  69  28  35  22  12  77  44  38  75 184 174 154  97 214
  45  72 130  94 139 135 197 137 141 156 117 155 134  89  80 108  61 124
 132 196 107 114 172 165 105 119 183 175  88  62  86 170 145 217  91 195
 152  21 126 115 223 207 123 236 128 151 100 198 157 168  84  99 173 121
 159  93  23 212 111 193 103 113 122 106  96 249 218 194 213 191 142 224
 244 143 267 256 211 161 131 246 118 164 275 204 230 243 112 238 144 185
 101 222 138 206 104 200 129 247 140 209 136 176 120 229 210 133 259 147
 227 150 282 162 265 260 189 237 245 205 308 283 248 303 291 280 208 286
 352 290 262 203 284 293 160 182 316 338 279 187 277 362 321 331 372 377
 350 220 472 450 268 435 169 225 464 485 323 388 367 266 255 415 233 467
 456 305 171 470 385 253 215 240 235 263 221 351 539 458 339 301 397 271
 532 480 365 241 421 242 234 341 394 540 463 361 429 359 180 188 261 254
 366 181 398 272 167 149 325 521 426 298 428 487 431 288 239 453 454 345
 417 434 278 285 442 484 451 252 471 488 270 258 264 281 410 516 500 343
 311 432 475 479 355 329 199 400 414 423 232 219 302 529 510 348 346 441
 473 335 445 555 527 273 364 299 269 257 342 324 226 391 466 297 517 486
 489 492 228 289 455 382 380 295 251 418 412 340 433 231 333 514 483 276
 478 287 381 334 347 320 493 491 369 201 408 378 443 460 465 313 513 292
 497 376 326 413 328 525 296 452 506 393 368 337 567 462 349 319 300 515
 373 399 507 396 512 503 386 427 312 384 530 310 536 437 505 371 375 534
 469 474 553 402 274 523 448 409 387 438 407 250 459 425 422 379 392 430
 401 306 370 449 363 389 374 436 356 317 446 294 508 315 522 494 327 495
 404 447 504 318 579 551 498 533 332 554 509 573 545 395 440 547 557 623
 571 614 638 628 642 647 602 634 648 353 322 357 314 563 615 681 601 543
 577 354 661 653 304 645 646 419 610 677 618 595 565 586 670 656 626 581
 546 604 596 383 621 564 309 360 330 549 589 461 631 673 358 651 663 538
 616 662 344 640 659 770 608 617 584 307 667 605 641 594 629 603 518 665
 769 749 499 719 734 696 688 570 675 405 411 643 733 390 680 764 679 531
 637 652 778 703 537 576 613 715 726 598 625 444 672 782 548 682 750 716
 609 698 572 669 633 725 704 658 620 542 575 511 741 790 644 740 735 560
 739 439 660 697 336 619 712 624 580 678 684 468 649 786 718 775 636 578
 746 743 481 664 711 689 751 745 424 699 552 709 591 757 768 767 723 558
```

```
 561 403 502 692 780 622 761 690 744 857 562 702 802 727 811 886 406 787
 496 708 758 812 807 791 639 781 833 756 544 789 742 655 416 806 773 737
 706 566 713 800 839 779 766 794 803 788 720 668 490 568 597 477 583 501
 556 593 420 541 694 650 559 666 700 693 582]
count :
 [ 16  40  32  13   1   2   3   8  14  36  56  84  94 106 110  93  67  35
  37  34  28  39  17   9   6  20  53  70  75  59  74  76  65  30  22  31
   5  64 154  88  44  51  61  77  72 157  52  12   4 179 100  42  57  78
  97  63  83 212 182 112  54  48  11  33 195 115  46  79  71  62  89 190
 169 132  43  19  95 219 122  45  86 172 163  69  23   7 210 134  73  50
  87 187 123  15  25  98 102  55  10  49  82  92  41  38 188  47 178 155
  24  18  27  99 217 130 136  29 128  81  68 139 137 202  60 162 144 158
 117  90 159 101 118 129  26 104  91 113 105  21  80 125 133 197 109 161
 135 116 176 168 108 103 175 147  96 220 127 205 174 121 230  66 114 216
 243 152 199  58 166 170 165 160 140 211 120 145 256 126 223  85 206 124
 255 222 285 146 274 272 185 191 232 327 224 107 119 196 171 214 242 148
 268 201 150 111 167 228 198 204 164 233 257 151 248 235 141 249 194 259
 156 153 244 213 181 221 250 304 241 271 282 225 253 237 299 142 313 310
 207 138 280 173 332 331 149 267 301 312 278 281 184 215 367 349 292 303
 339 143 189 366 386 273 325 356 314 343 333 226 203 177 263 297 288 236
 240 131 452 383 284 291 309 321 193 337 388 300 200 180 209 354 361 306
 277 428 362 286 351 192 411 421 276 264 238 266 371 269 537 518 218 265
 459 186 517 544 365 290 410 396 296 440 533 520 258 450 246 260 344 553
 470 298 347 373 436 378 342 289 340 382 390 358 385 239 374 598 524 384
 425 611 550 434 318 442 401 234 594 527 364 387 491 398 270 279 294 295
 322 456 437 392 231 394 453 308 604 480 283 565 489 487 183 302 547 513
 454 486 467 572 525 379 502 558 564 391 293 247 317 369 420 451 404 341
 251 335 417 363 357 438 579 556 407 336 334 477 539 551 424 346 353 481
 506 432 409 466 326 254 463 380 275 311 315 360 350 252 328 476 227 601
 586 423 330 569 538 370 498 638 607 416 261 355 552 208 468 449 381 377
 397 492 427 461 422 305 375 376 414 447 408 418 457 545 496 368 245 596
 563 443 562 229 316 402 287 372 514 472 511 488 419 595 578 400 348 587
 497 433 475 406 430 324 262 323 412 530 543 413 435 555 523 441 529 532
 585 399 584 559 307 582 571 426 516 465 329 483 600 570 628 531 455 389
 505 359 431 460 590 429 599 338 566 482 568 540 495 345 591 593 446 485
 393 500 473 352 320 479 444 462 405 620 499 625 395 528 319 519 445 512
 471 508 526 509 484 448 515 549 501 612 597 464 644 712 676 734 662 782
 749 623 713 746 651 686 690 679 685 648 560 503 521 554 541 721 801 561
 573 589 729 618 494 757 800 684 744 759 822 698 490 536 655 643 626 615
 567 617 632 646 692 704 624 656 610 738 671 678 660 658 635 681 616 522
 673 781 775 576 677 748 776 557 743 666 813 504 627 706 641 575 639 769
 680 546 717 710 458 622 705 630 732 770 439 779 659 602 478 733 650 873
 846 474 634 852 868 745 812 669 642 730 672 645 694 493 668 647 702 665
 834 850 790 415 724 869 700 793 723 534 831 613 653 857 719 867 823 403
 693 603 583 542 614 580 811 795 747 581 722 689 849 872 631 649 819 674
 830 814 633 825 629 835 667 755 794 661 772 657 771 777 837 891 652 739
 865 767 741 469 605 858 843 640 737 862 810 577 818 854 682 851 848 897
 832 791 654 856 839 725 863 808 792 696 701 871 968 750 970 877 925 977
 758 884 766 894 715 783 683 842 774 797 886 892 784 687 809 917 901 887
 785 900 761 806 507 948 844 798 827 670 637 619 592 943 838 817 888 890
 788 588 606 608 691 711 663 731 708 609 688 636]
```

In [139…
```
df['season'].value_counts()
```

Out[139]:
```
4    2734
2    2733
3    2733
1    2686
Name: season, dtype: int64
```

In [140…
```
for i in df.columns:
    print(i,':',df[i].value_counts())
```

```
datetime : 2011-01-01 00:00:00    1
2012-05-01 21:00:00    1
2012-05-01 13:00:00    1
2012-05-01 14:00:00    1
2012-05-01 15:00:00    1
                      ..
2011-09-02 04:00:00    1
2011-09-02 05:00:00    1
2011-09-02 06:00:00    1
2011-09-02 07:00:00    1
2012-12-19 23:00:00    1
Name: datetime, Length: 10886, dtype: int64
season : 4    2734
2    2733
3    2733
1    2686
Name: season, dtype: int64
holiday : 0    10575
1      311
Name: holiday, dtype: int64
workingday : 1    7412
0    3474
Name: workingday, dtype: int64
weather : 1    7192
2    2834
3     859
```

```
4       1
Name: weather, dtype: int64
temp : 14.76    467
26.24    453
28.70    427
13.94    413
18.86    406
22.14    403
25.42    403
16.40    400
22.96    395
27.06    394
24.60    390
12.30    385
21.32    362
17.22    356
13.12    356
29.52    353
10.66    332
18.04    328
20.50    327
30.34    299
9.84     294
15.58    255
9.02     248
31.16    242
8.20     229
27.88    224
23.78    203
32.80    202
11.48    181
19.68    170
6.56     146
33.62    130
5.74     107
7.38     106
31.98     98
34.44     80
35.26     76
4.92      60
36.90     46
4.10      44
37.72     34
36.08     23
3.28      11
0.82       7
38.54      7
39.36      6
2.46       5
1.64       2
41.00      1
Name: temp, dtype: int64
atemp : 31.060    671
25.760    423
22.725    406
20.455    400
26.515    395
16.665    381
25.000    365
33.335    364
21.210    356
30.305    350
15.150    338
21.970    328
24.240    327
17.425    314
31.820    299
34.850    283
27.275    282
32.575    272
11.365    271
14.395    269
29.545    257
19.695    255
15.910    254
12.880    247
13.635    237
34.090    224
12.120    195
28.790    175
23.485    170
10.605    166
35.605    159
9.850     127
18.180    123
36.365    123
37.120    118
9.090     107
37.880     97
```

```
28.030     80
7.575      75
38.635     74
6.060      73
39.395     67
6.820      63
8.335      63
18.940     45
40.150     45
40.910     39
5.305      25
42.425     24
41.665     23
3.790      16
4.545      11
3.030       7
43.940      7
2.275       7
43.180      7
44.695      3
0.760       2
1.515       1
45.455      1
Name: atemp, dtype: int64
humidity : 88    368
94    324
83    316
87    289
70    259
     ...
8       1
10      1
97      1
96      1
91      1
Name: humidity, Length: 89, dtype: int64
windspeed : 0.0000     1313
8.9981     1120
11.0014    1057
12.9980    1042
7.0015     1034
15.0013     961
6.0032      872
16.9979     824
19.0012     676
19.9995     492
22.0028     372
23.9994     274
26.0027     235
27.9993     187
30.0026     111
31.0009      89
32.9975      80
35.0008      58
39.0007      27
36.9974      22
43.0006      12
40.9973      11
43.9989       8
46.0022       3
56.9969       2
47.9988       2
51.9987       1
50.0021       1
Name: windspeed, dtype: int64
casual : 0      986
1      667
2      487
3      438
4      354
      ...
332      1
361      1
356      1
331      1
304      1
Name: casual, Length: 309, dtype: int64
registered : 3      195
4      190
5      177
6      155
2      150
      ...
570      1
422      1
678      1
565      1
636      1
Name: registered, Length: 731, dtype: int64
```

```
count : 5      169
4      149
3      144
6      135
2      132
      ...
801      1
629      1
825      1
589      1
636      1
Name: count, Length: 822, dtype: int64
```

In [141…
```
cat_cols= ['season', 'holiday', 'workingday', 'weather']
#for col in cat_cols:
 #  df[col] = df[col].astype('object')
```

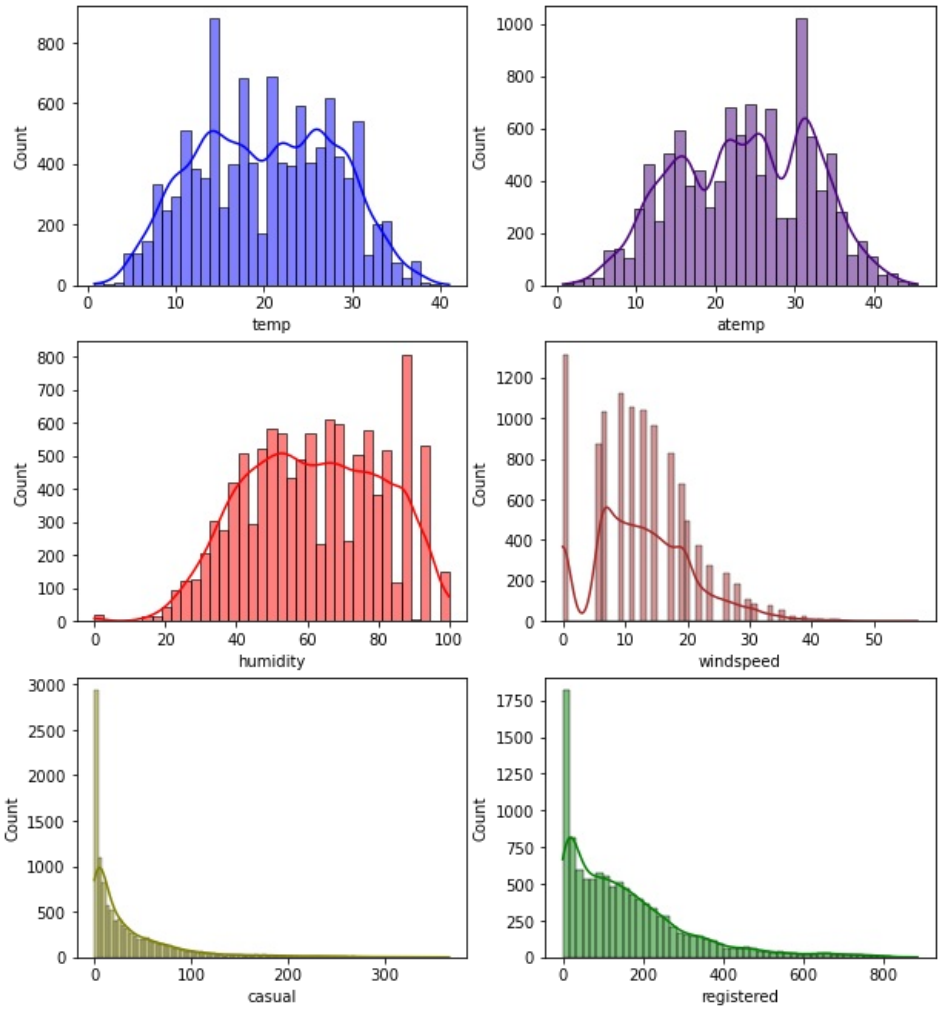
In [142…
```
df[cat_cols].melt().groupby(['variable', 'value'])[['value']].count()
```

Out[142]:

		value
variable	value	
holiday	0	10575
	1	311
season	1	2686
	2	2733
	3	2733
	4	2734
weather	1	7192
	2	2834
	3	859
	4	1
workingday	0	3474
	1	7412

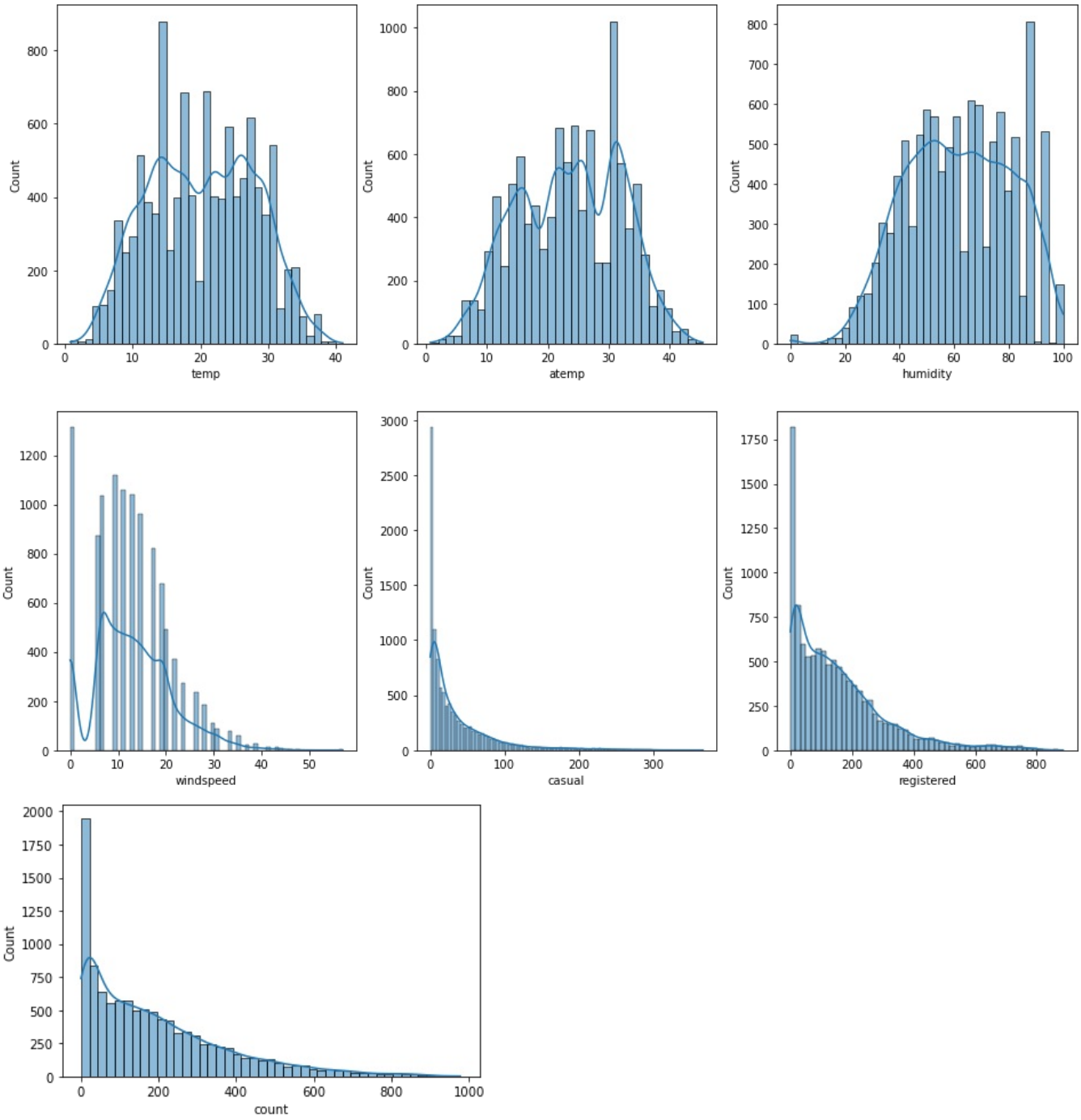
Univariate Analysis¶

In [143…
```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(10,8))
fig.subplots_adjust(top=1.2)
sns.histplot(data=df, x="temp", kde=True,color="blue", ax=axis[0,0])
sns.histplot(data=df, x="atemp", kde=True,color="indigo", ax=axis[0,1])
sns.histplot(data=df, x="humidity", kde=True,color="red", ax=axis[1,0])
sns.histplot(data=df, x="windspeed", kde=True,color="brown", ax=axis[1,1])
sns.histplot(data=df, x="casual", kde=True,color="olive", ax=axis[2,0])
sns.histplot(data=df, x="registered", kde=True,color="green", ax=axis[2,1])
plt.show()
```



In [144…

```
num_cols = ['temp', 'atemp', 'humidity', 'windspeed', 'casual', 'registered','count']

fig, axis = plt.subplots(nrows=2, ncols=3, figsize=(16, 12))

index = 0
for row in range(2):
    for col in range(3):
        sns.histplot(df[num_cols[index]], ax=axis[row, col], kde=True)
        index += 1

plt.show()
sns.histplot(df[num_cols[-1]], kde=True)
plt.show()
```

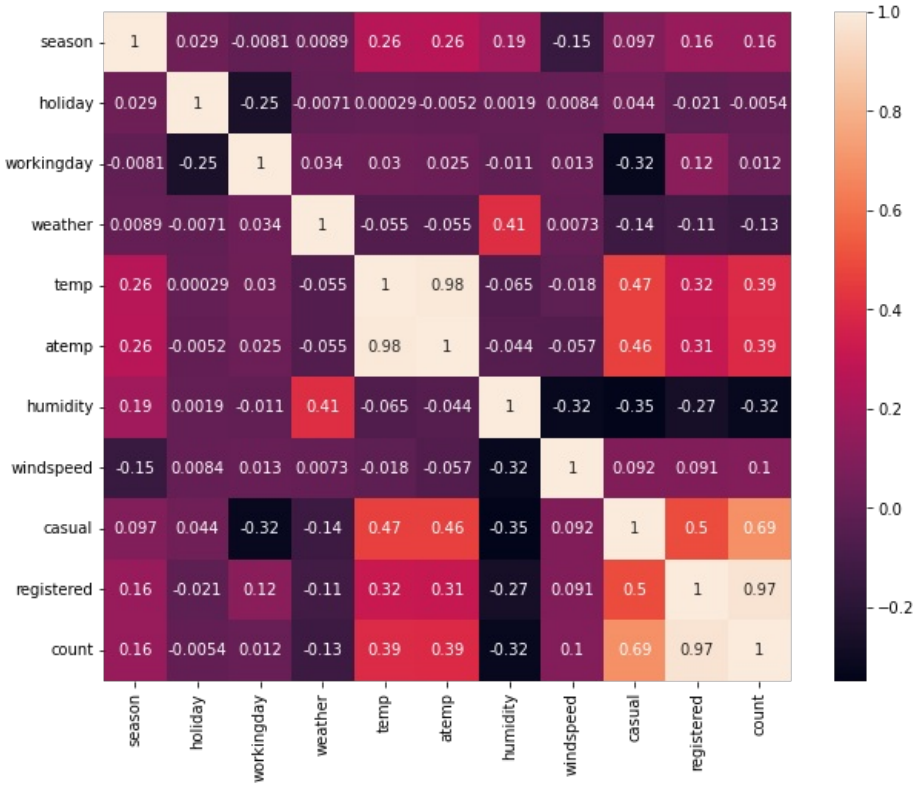


- **casual, registered and count somewhat looks like Log Normal Distrinution**
- **temp, atemp and humidity looks like they follows the Normal Distribution**
- **windspeed follows the binomial distribution**

Correlation of the quantitative data

In [145…

```
plt.figure(figsize=(10,8))
sns.heatmap(df.corr(), annot=True)
plt.show()
```



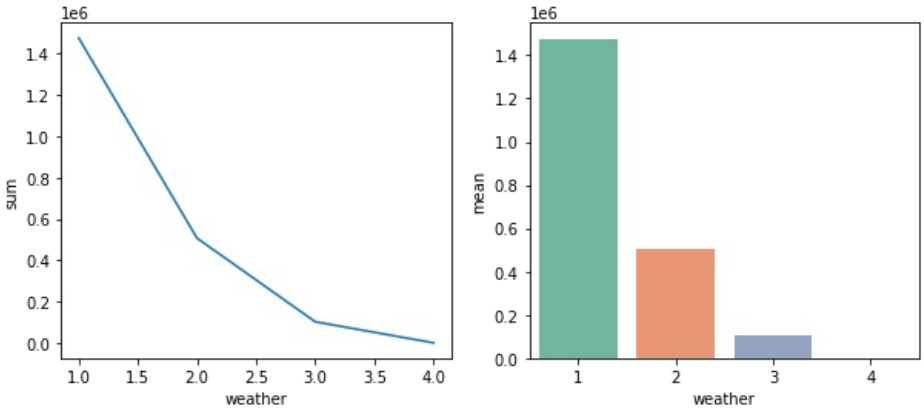
Dependence of weather on total user count

In [146…
```
x=pd.DataFrame({'sum':df.groupby("weather")['count'].sum().sort_values(ascending=False),
                'mean':df.groupby("weather")['count'].sum().sort_values(ascending=False)})
weather_count = x.reset_index()
weather_count
```

Out[146]:

	weather	sum	mean
0	1	1476063	1476063
1	2	507160	507160
2	3	102089	102089
3	4	164	164

In [147…
```
plt.figure(figsize=(10,4))
plt.subplot(121)
sns.lineplot(data=weather_count, x='weather', y='sum')
plt.subplot(122)
sns.barplot(data=weather_count, x='weather', y='mean', palette='Set2')
plt.show()
```



In clear weather (weather 1) more bikes are used

Dependence of season on total user count

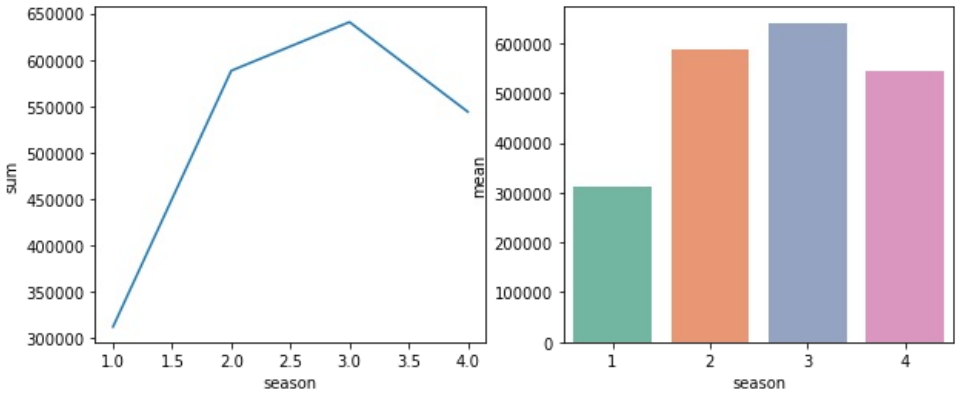
In [148…
```
x=pd.DataFrame({'sum':df.groupby("season")['count'].sum().sort_values(ascending=False),
                'mean':df.groupby("season")['count'].sum().sort_values(ascending=False)})
season_count = x.reset_index()
season_count
```

Out[148]:

	season	sum	mean
0	3	640662	640662
1	2	588282	588282
2	4	544034	544034
3	1	312498	312498

In [149…

```
plt.figure(figsize=(10,4))
plt.subplot(121)
sns.lineplot(data=season_count, x='season', y='sum')
plt.subplot(122)
sns.barplot(data=season_count, x='season', y='mean', palette='Set2')
plt.show()
```



In fall season(season-3) more bikes are used

Dependence of temperature on total user count

In [150…

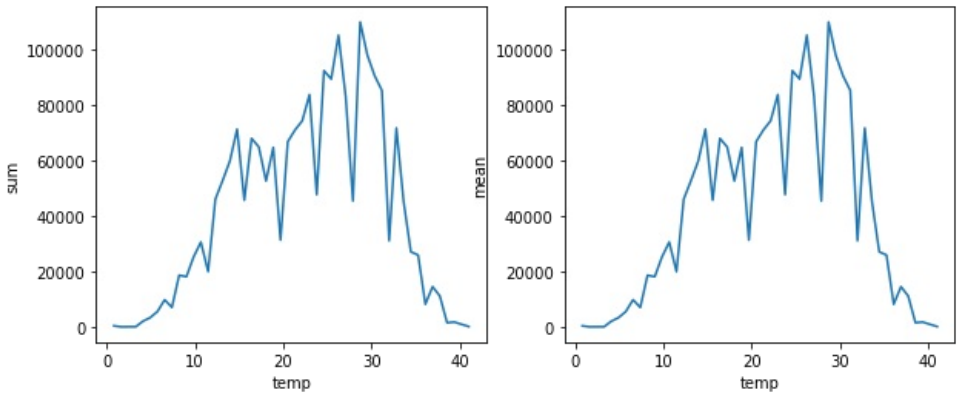
```
x=pd.DataFrame({'sum':df.groupby("temp")['count'].sum().sort_values(ascending=False),
                'mean':df.groupby("temp")['count'].sum().sort_values(ascending=False)})
temp_count = x.reset_index()
temp_count
```

Out[150]:

	temp	sum	mean
0	28.70	110029	110029
1	26.24	105279	105279
2	29.52	98025	98025
3	24.60	92501	92501
4	30.34	90655	90655
5	25.42	89491	89491
6	31.16	85378	85378
7	22.96	83895	83895
8	27.06	83144	83144
9	22.14	74441	74441
10	32.80	71836	71836
11	14.76	71431	71431
12	21.32	71126	71126
13	16.40	68087	68087
14	20.50	66928	66928
15	17.22	65009	65009
16	18.86	64835	64835
17	13.94	59907	59907
18	13.12	52883	52883
19	18.04	52768	52768
20	23.78	47837	47837
21	12.30	46201	46201
22	15.58	45819	45819
23	27.88	45569	45569
24	33.62	45282	45282
25	19.68	31460	31460
26	31.98	31231	31231
27	10.66	30730	30730
28	34.44	27218	27218
29	35.26	26063	26063
30	9.84	25414	25414
31	11.48	20103	20103
32	8.20	18777	18777
33	9.02	18257	18257
34	36.90	14661	14661
35	37.72	11294	11294
36	6.56	9944	9944
37	36.08	8346	8346
38	7.38	7182	7182
39	5.74	5696	5696
40	4.92	3505	3505
41	4.10	2212	2212
42	39.36	1907	1907
43	38.54	1672	1672
44	0.82	544	544
45	41.00	294	294
46	2.46	215	215
47	3.28	212	212
48	1.64	183	183

In [151…

```
plt.figure(figsize=(10,4))
plt.subplot(121)
sns.lineplot(data=temp_count, x='temp', y='sum')
plt.subplot(122)
sns.lineplot(data=temp_count, x='temp', y='mean')
plt.show()
```



Dependence of feeling temperature on total user count

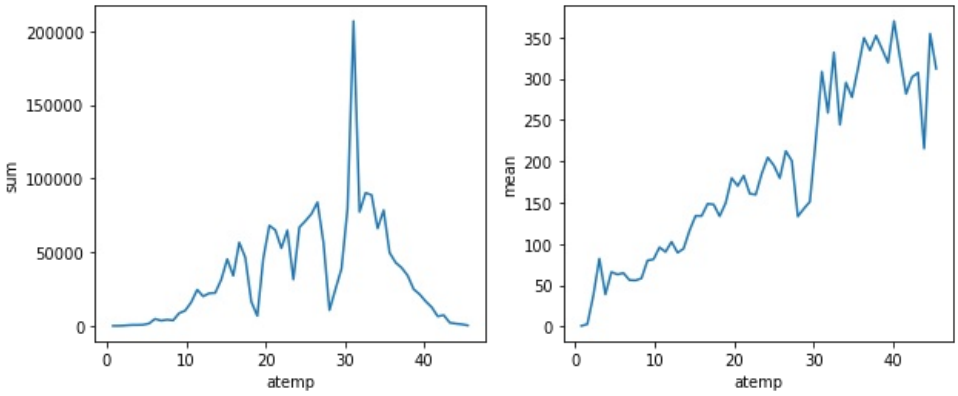
In [152…
```
x = pd.DataFrame({'sum' : df.groupby('atemp')['count'].sum().sort_values(ascending=False),
                  'mean': df.groupby('atemp')['count'].mean().sort_values(ascending=False)})
atemp_count = x.reset_index()
atemp_count
```

Out[152]:

	atemp	sum	mean
0	0.760	2	1.000000
1	1.515	3	3.000000
2	2.275	266	38.000000
3	3.030	576	82.285714
4	3.790	625	39.062500
5	4.545	727	66.090909
6	5.305	1580	63.200000
7	6.060	4736	64.876712
8	6.820	3552	56.380952
9	7.575	4195	55.933333
10	8.335	3682	58.444444
11	9.090	8560	80.000000
12	9.850	10345	81.456693
13	10.605	15928	95.951807
14	11.365	24510	90.442804
15	12.120	20018	102.656410
16	12.880	22111	89.518219
17	13.635	22351	94.308017
18	14.395	31334	116.483271
19	15.150	45281	133.967456
20	15.910	34010	133.897638
21	16.665	56582	148.509186
22	17.425	46409	147.799363
23	18.180	16431	133.585366
24	18.940	6730	149.555556
25	19.695	45819	179.682353
26	20.455	68087	170.217500
27	21.210	65009	182.609551
28	21.970	52768	160.878049
29	22.725	64835	159.692118
30	23.485	31460	185.058824
31	24.240	66928	204.672783
32	25.000	71215	195.109589
33	25.760	75982	179.626478
34	26.515	83895	212.392405
35	27.275	56542	200.503546
36	28.030	10665	133.312500
37	28.790	24985	142.771429

38	29.545	38819	151.046693
39	30.305	79552	227.291429
40	31.060	206885	308.323398
41	31.820	77338	258.655518
42	32.575	90235	331.746324
43	33.335	88855	244.107143
44	34.090	66121	295.183036
45	34.850	78518	277.448763
46	35.605	49631	312.144654
47	36.365	42957	349.243902
48	37.120	39429	334.144068
49	37.880	34128	351.835052
50	38.635	24848	335.783784
51	39.395	21386	319.194030
52	40.150	16631	369.577778
53	40.910	12656	324.512821
54	41.665	6473	281.434783
55	42.425	7247	301.958333
56	43.180	2150	307.142857
57	43.940	1508	215.428571
58	44.695	1063	354.333333
59	45.455	312	312.000000

In [153…
```
plt.figure(figsize=(10,4))
plt.subplot(121)
sns.lineplot(data=atemp_count, x='atemp', y='sum')
plt.subplot(122)
sns.lineplot(data=atemp_count, x='atemp', y='mean')
plt.show()
```



Dependence of humidity on total user count

In [154…
```
x = pd.DataFrame({'sum' : df.groupby('humidity')['count'].sum().sort_values(ascending=False),
                  'mean': df.groupby('humidity')['count'].mean().sort_values(ascending=False)})
humidity_count = x.reset_index()
humidity_count
```

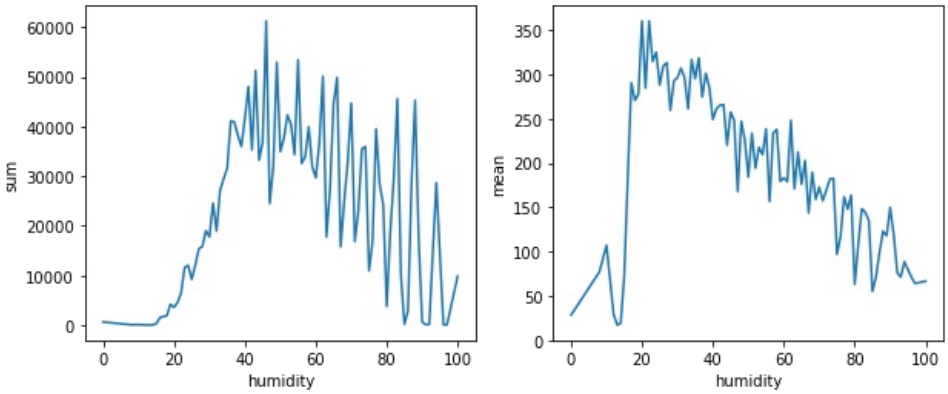
Out[154]:

	humidity	sum	mean
0	0	623	28.318182
1	8	77	77.000000
2	10	107	107.000000
3	12	29	29.000000
4	13	17	17.000000
...	...	...	...
84	93	14586	71.151220
85	94	28666	88.475309
86	96	71	71.000000
87	97	64	64.000000
88	100	9841	66.493243

89 rows × 3 columns

In [155…

```
plt.figure(figsize=(10,4))
plt.subplot(121)
sns.lineplot(data=humidity_count, x='humidity', y='sum')
plt.subplot(122)
sns.lineplot(data=humidity_count, x='humidity', y='mean')
plt.show()
```



Dependence of wind speed on total user count

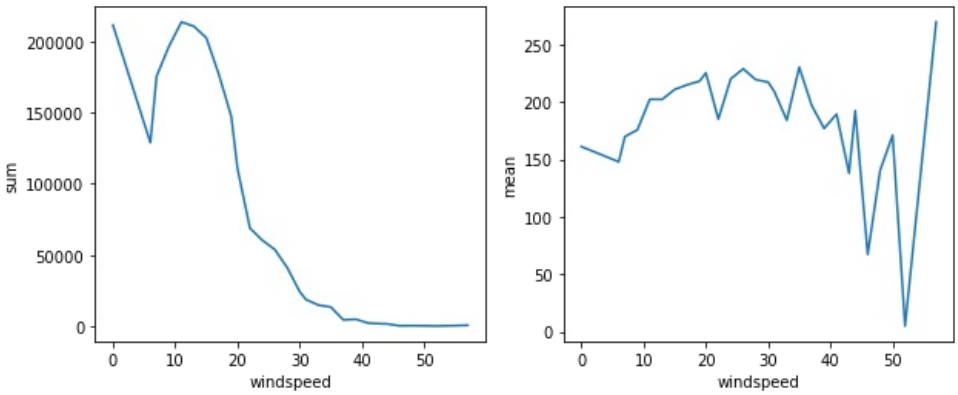
In [156…

```
x = pd.DataFrame({'sum' : df.groupby('windspeed')['count'].sum().sort_values(ascending=False),
                  'mean': df.groupby('windspeed')['count'].mean().sort_values(ascending=False)})
windspeed_count = x.reset_index()
windspeed_count
```

Out[156]:

	windspeed	sum	mean
0	0.0000	211526	161.101295
1	6.0032	128938	147.864679
2	7.0015	175627	169.852031
3	8.9981	196723	175.645536
4	11.0014	213791	202.262062
5	12.9980	210744	202.249520
6	15.0013	202611	210.833507
7	16.9979	177034	214.847087
8	19.0012	147403	218.051775
9	19.9995	110816	225.235772
10	22.0028	68840	185.053763
11	23.9994	60283	220.010949
12	26.0027	53755	228.744681
13	27.9993	41021	219.363636
14	30.0026	24106	217.171171
15	31.0009	18597	208.955056
16	32.9975	14726	184.075000
17	35.0008	13349	230.155172
18	36.9974	4335	197.045455
19	39.0007	4776	176.888889
20	40.9973	2083	189.363636
21	43.0006	1655	137.916667
22	43.9989	1539	192.375000
23	46.0022	202	67.333333
24	47.9988	281	140.500000
25	50.0021	171	171.000000
26	51.9987	5	5.000000
27	56.9969	539	269.500000

In [157…
```
plt.figure(figsize=(10,4))
plt.subplot(121)
sns.lineplot(data=windspeed_count, x='windspeed', y='sum')
plt.subplot(122)
sns.lineplot(data=windspeed_count, x='windspeed', y='mean')
plt.show()
```



In [158…
```
df1=df.copy()
```

Average no. of users per week and per month

In [159…
```
df1['weekday'] = df1['datetime'].dt.day_name()
df1['year'] = df1['datetime'].dt.year
df1['month'] = df1['datetime'].dt.month_name()
df1['day'] = df1['datetime'].dt.day
df1
```

Out[159]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	weekday	year
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0000	3	13	16	Saturday	2011
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40	Saturday	2011
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32	Saturday	2011
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13	Saturday	2011
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1	Saturday	2011
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	7	329	336	Wednesday	2012
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	10	231	241	Wednesday	2012
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	4	164	168	Wednesday	2012
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	12	117	129	Wednesday	2012
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	4	84	88	Wednesday	2012

10886 rows × 16 columns

In [160…

```
weekday_count_df = df1.groupby('weekday')['count'].mean().sort_values(ascending=False).reset_index()
month_count_df = df1.groupby('month')['count'].mean().sort_values(ascending=False).reset_index()
weekday_count_df
#month_count_df
```

Out[160]:

	weekday	count
0	Friday	197.844343
1	Thursday	197.296201
2	Saturday	196.665404
3	Monday	190.390716
4	Tuesday	189.723847
5	Wednesday	188.411348
6	Sunday	180.839772

In [161…

```
month_count_df
```

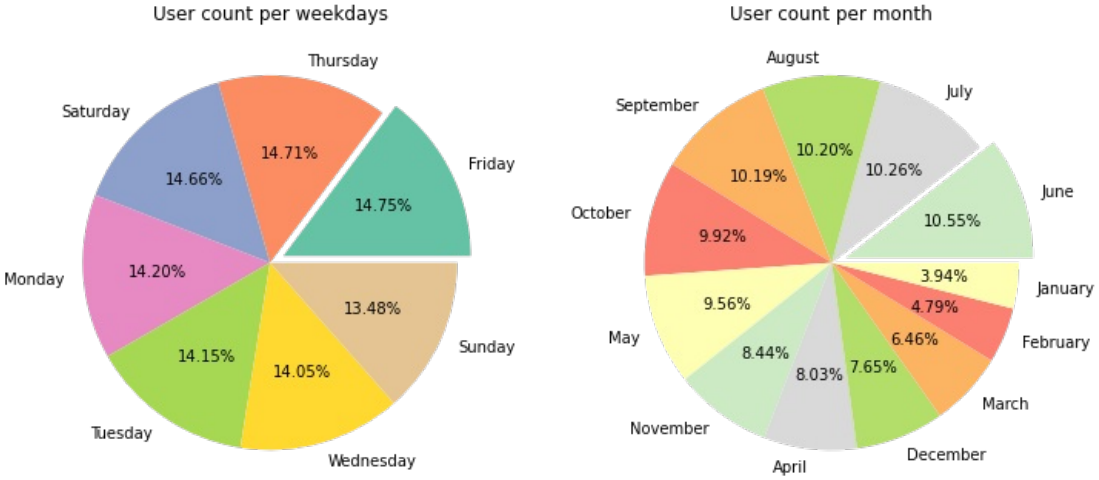
Out[161]:

	month	count
0	June	242.031798
1	July	235.325658
2	August	234.118421
3	September	233.805281
4	October	227.699232
5	May	219.459430
6	November	193.677278
7	April	184.160616
8	December	175.614035
9	March	148.169811
10	February	110.003330
11	January	90.366516

In [162…

```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(12, 6))
```

```
palette_color = sns.color_palette('Set2')
axs[0].pie(data=weekday_count_df, x=weekday_count_df['count'], colors=palette_color, labels=['Friday','Thursday
axs[0].set_title("User count per weekdays")

palette_color = sns.color_palette('Set3_r')
axs[1].pie(data=month_count_df, x=month_count_df['count'], colors=palette_color, labels=['June', 'July','August
axs[1].set_title("User count per month")
plt.show()
```

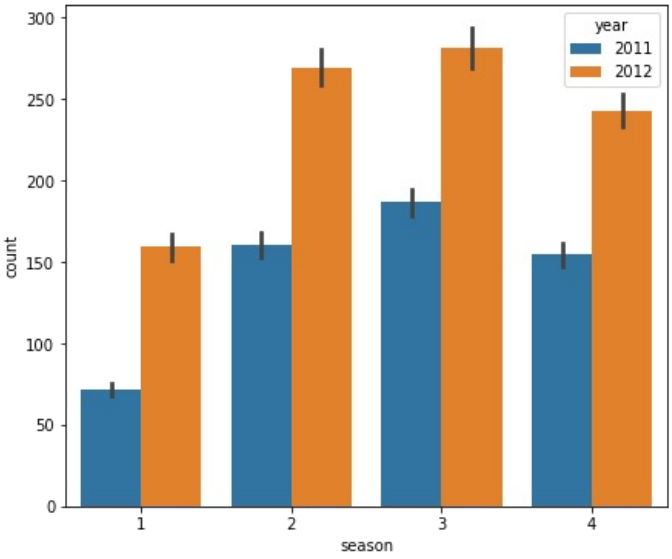


on the basis of month,in june they are more used the bikes.

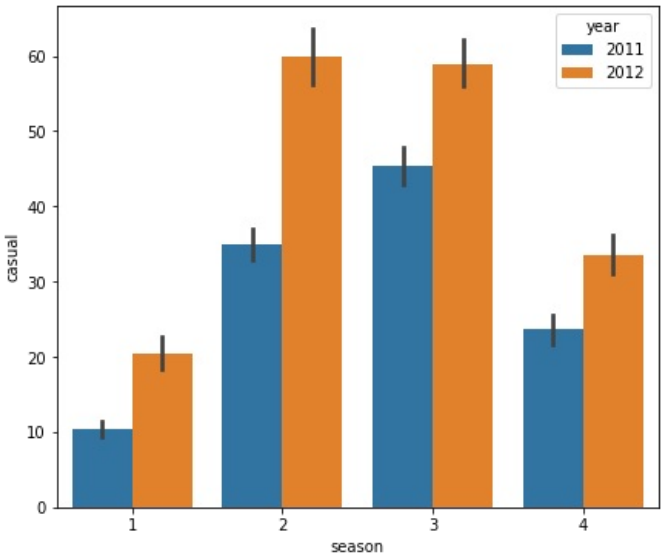
on the basis days,in friday they are more used bikes .

In [163…
```
plt.figure(figsize=(7,6))
sns.barplot(data=df1,x='season',y='count',hue='year')
plt.show
```

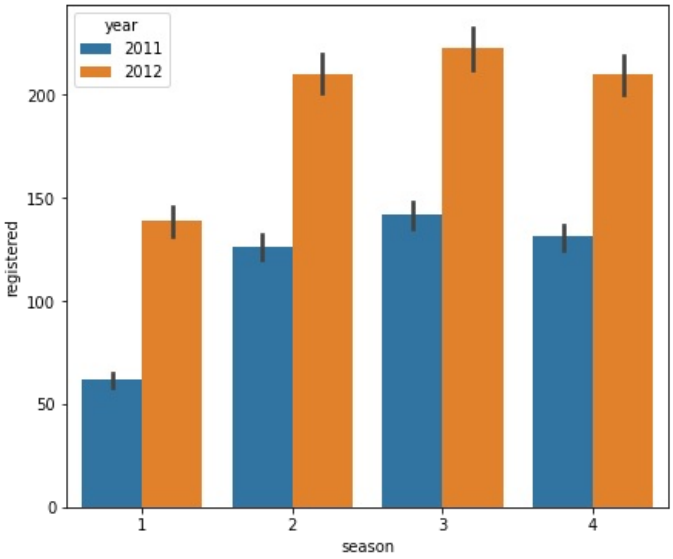
Out[163]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



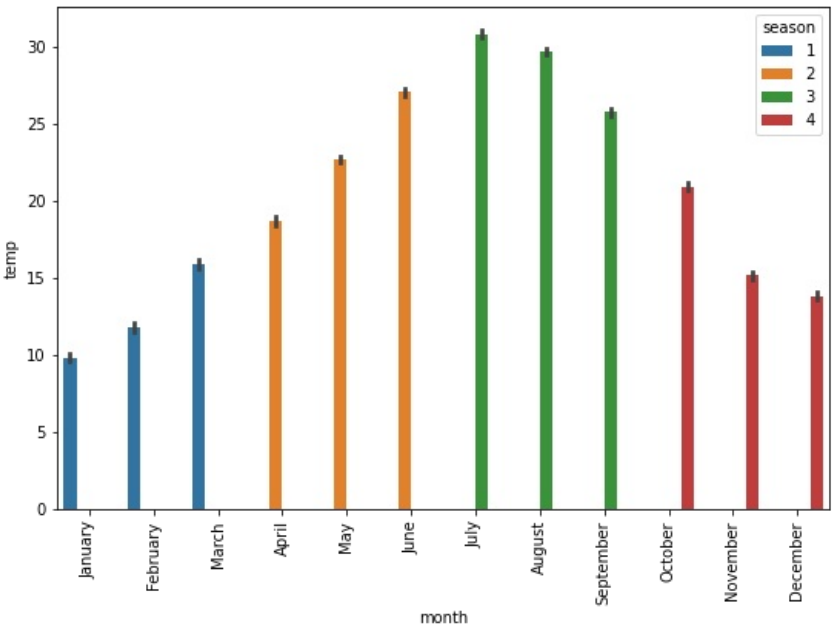
In [164…
```
plt.figure(figsize=(7,6))
sns.barplot(data=df1,x='season',y='casual',hue='year')
plt.show()
```



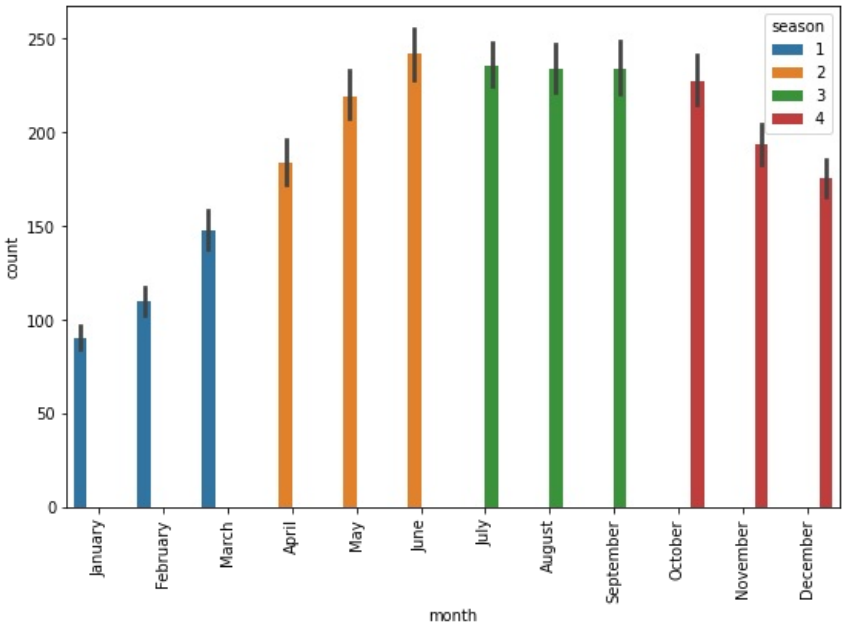
In [165…
```
plt.figure(figsize=(7,6))
sns.barplot(data=df1,x='season',y='registered',hue='year')
plt.show()
```



In [166…
```
plt.figure(figsize=(9,6))
sns.barplot(data=df1,x='month',y='temp',hue='season')
plt.xticks(rotation=90)
plt.show()
```



In [167…
```
plt.figure(figsize=(9,6))
sns.barplot(data=df1,x='month',y='count',hue='season')
plt.xticks(rotation=90)
plt.show()
```



In [168…
```
df1['season']=df1['season'].replace({1:'spring',2:'summer',3:'flat',4:'winter'})
df1['weather']=df1['weather'].replace({1:'clear',2:'mist',3:'lightsnow',4:'heavyrain'})
df1['temp'] = pd.cut(df1['temp'],bins=[0,10,20,30,45],labels=['low_temp','medium_temp','normal_temp','high_temp
df1['atemp'] = pd.cut(df1['atemp'],bins=[0,15,25,35,50],labels=['low_temp','medium_temp','normal_temp','high_te
df1['humidity'] = pd.cut(df1['humidity'],bins=[0,25,50,75,102],labels=['< 25%','< 50%','< 75%','< 100%'])
df1['windspeed'] = pd.cut(df1['windspeed'],bins=[-1,15,30,45,60],labels=['low speed','medium speed','high speed
```

In [169…
```
df1
```

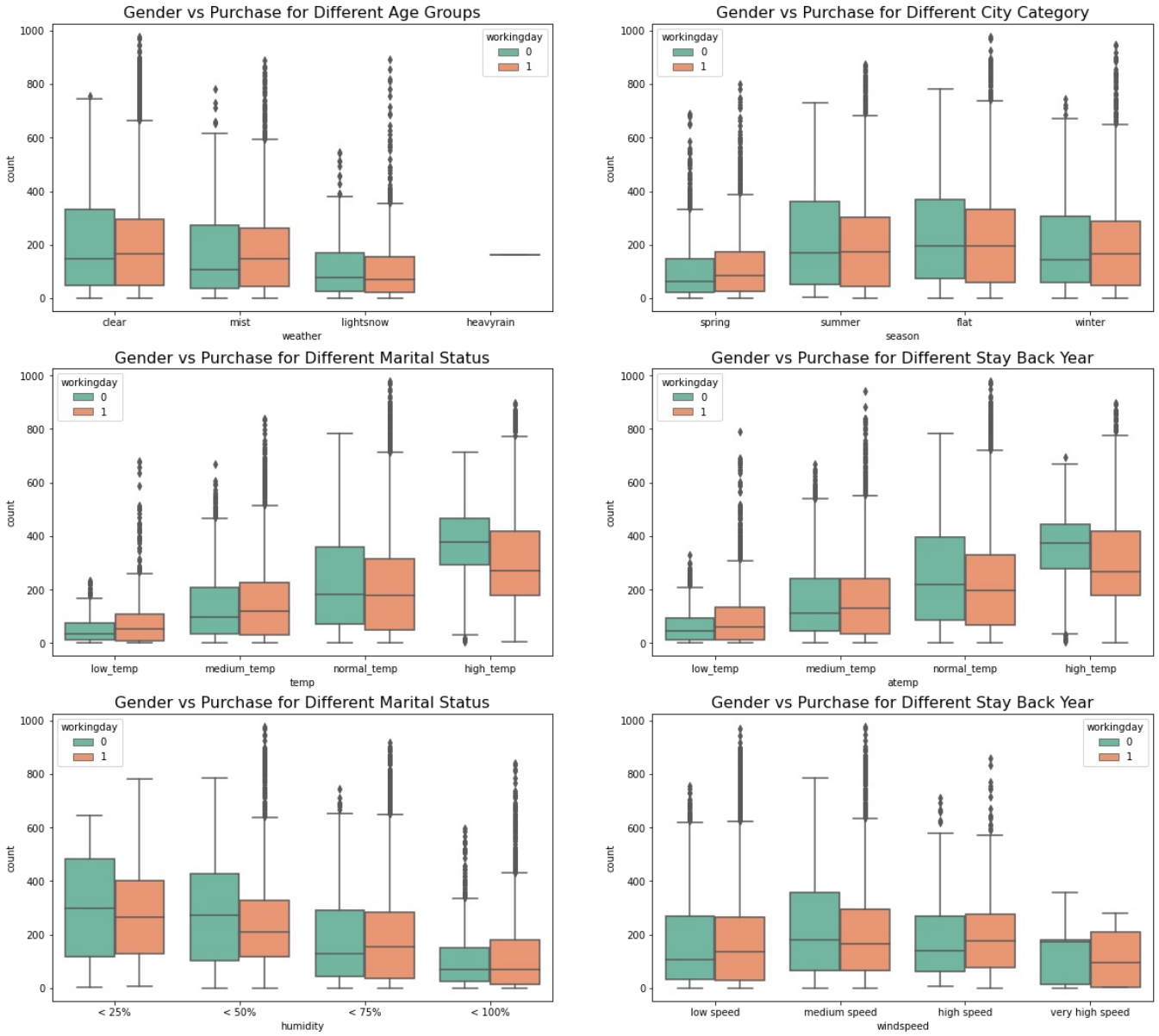
Out[169]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	w
0	2011-01-01 00:00:00	spring	0	0	clear	low_temp	low_temp	< 100%	low speed	3	13	16	Sa
1	2011-01-01 01:00:00	spring	0	0	clear	low_temp	low_temp	< 100%	low speed	8	32	40	Sa
2	2011-01-01 02:00:00	spring	0	0	clear	low_temp	low_temp	< 100%	low speed	5	27	32	Sa
3	2011-01-01 03:00:00	spring	0	0	clear	low_temp	low_temp	< 75%	low speed	3	10	13	Sa
4	2011-01-01 04:00:00	spring	0	0	clear	low_temp	low_temp	< 75%	low speed	0	1	1	Sa
...	...	...	...	...	...	...	...	...	...	...	...	...	
10881	2012-12-19 19:00:00	winter	0	1	clear	medium_temp	medium_temp	< 50%	medium speed	7	329	336	Wed
10882	2012-12-19 20:00:00	winter	0	1	clear	medium_temp	medium_temp	< 75%	medium speed	10	231	241	Wed
10883	2012-12-19 21:00:00	winter	0	1	clear	medium_temp	medium_temp	< 75%	medium speed	4	164	168	Wed
10884	2012-12-19 22:00:00	winter	0	1	clear	medium_temp	medium_temp	< 75%	low speed	12	117	129	Wed
10885	2012-12-19 23:00:00	winter	0	1	clear	medium_temp	medium_temp	< 75%	low speed	4	84	88	Wed

10886 rows × 16 columns

In [170…
```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(20, 10))
fig.subplots_adjust(top=1.5)
sns.boxplot(data=df1, y='count', x='weather', hue='workingday', palette='Set2', ax=axis[0,0])
sns.boxplot(data=df1, y='count', x='season', hue='workingday', palette='Set2', ax=axis[0,1])
sns.boxplot(data=df1, y='count', x='temp', hue='workingday', palette='Set2', ax=axis[1,0])
sns.boxplot(data=df1, y='count', x='atemp', hue='workingday', palette='Set2', ax=axis[1,1])
sns.boxplot(data=df1, y='count', x='humidity', hue='workingday', palette='Set2', ax=axis[2,0])
sns.boxplot(data=df1, y='count', x='windspeed', hue='workingday', palette='Set2', ax=axis[2,1])

axis[0,0].set_title("Gender vs Purchase for Different Age Groups",fontsize=16)
```

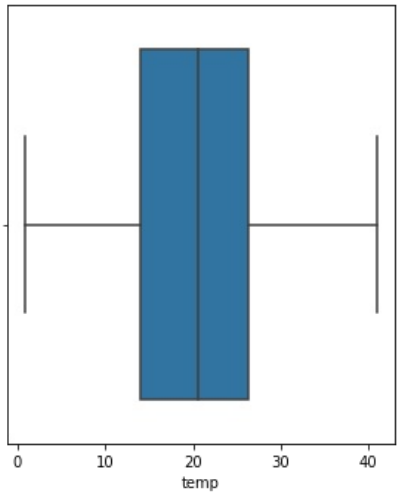
```
axis[0,1].set_title("Gender vs Purchase for Different City Category", fontsize=16)
axis[1,0].set_title("Gender vs Purchase for Different Marital Status", fontsize=16)
axis[1,1].set_title("Gender vs Purchase for Different Stay Back Year", fontsize=16)
axis[2,0].set_title("Gender vs Purchase for Different Marital Status", fontsize=16)
axis[2,1].set_title("Gender vs Purchase for Different Stay Back Year", fontsize=16)
plt.show()
```

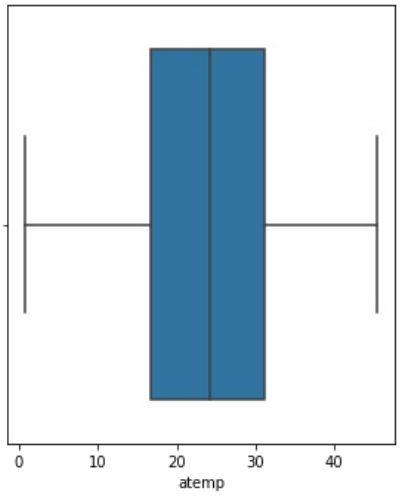


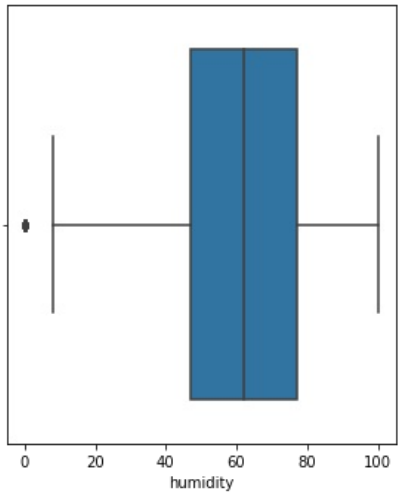
Detecting outliers

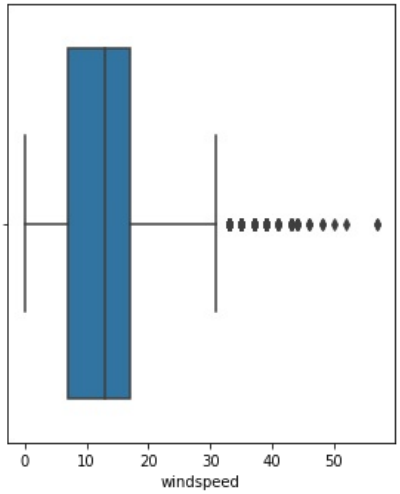
In [171…

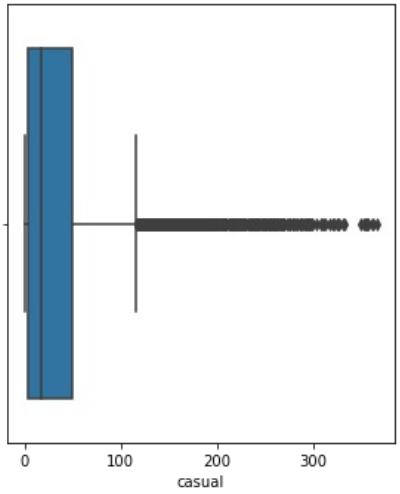
```
fig, axis = plt.subplots(nrows=2, ncols=3, figsize=(16, 12))

index = 0
for row in range(2):
    for col in range(3):
        sns.boxplot(x=df[num_cols[index]], ax=axis[row, col])
        index += 1

plt.show()
sns.boxplot(x=df[num_cols[-1]])
plt.show()
```

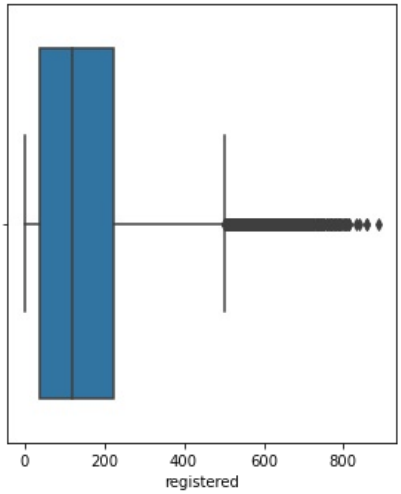


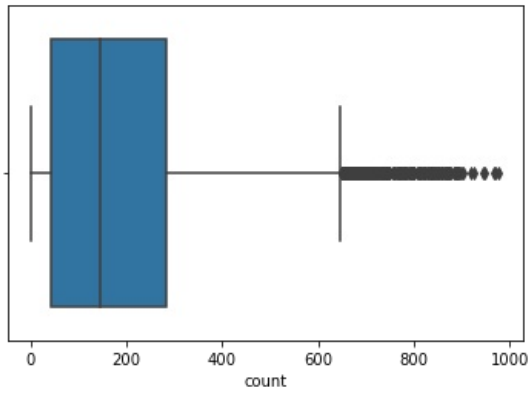












Hypothesis Testing

Chi Square testing

Null Hypothesis (H0): Weather is independent of the season

Alternate Hypothesis (H1): Weather is not independent of the season

Significance level (alpha): 0.05

In [175…
```
from scipy import stats
```

In [209…
```
#H0:Weather is Independent on season
#H1:Weather is dependent on season
df_ws = pd.crosstab(df.weather,df.season,margins=True,margins_name='Total')
df_ws
```

Out[209]:

season	1	2	3	4	Total
weather					
1	1759	1801	1930	1702	7192
2	715	708	604	807	2834
3	211	224	199	225	859
4	1	0	0	0	1
Total	2686	2733	2733	2734	10886

In [210…
```
# Above weather 4 has less expected count so not include 4
df_w=df[-(df['weather']==4)]
df_ws = pd.crosstab(df_w.weather, df_w.season, margins=True, margins_name='Total')
df_ws
```

Out[210]:

season	1	2	3	4	Total
weather					
1	1759	1801	1930	1702	7192
2	715	708	604	807	2834
3	211	224	199	225	859
Total	2685	2733	2733	2734	10885

In [186…
```
from scipy import stats
```

In [187…
```
stats, p_value, dof, expected = stats.chi2_contingency(df_ws)

print("p_value : ", p_value)
print("dof : ", dof)
print("expected : ", expected)
```

```
p_value :  6.664576536706683e-06
dof :  12
expected :  [[ 1774.04869086  1805.76352779  1805.76352779  1806.42425356
   7192.        ]
 [  699.06201194   711.55920992   711.55920992   711.81956821
   2834.        ]
 [  211.8892972    215.67726229   215.67726229   215.75617823
    859.        ]
 [ 2685.          2733.          2733.          2734.
  10885.        ]]
```

In [188…
```
alpha = 0.05
if p_value >= alpha:
    print('We Accept the Null Hypothesis : Weather is Independent on season  ')
else:
    print('We reject the Null Hypothesis : Weather is dependent on season ')
```

```
We reject the Null Hypothesis : Weather is dependent on season
```

We reject the Null Hypothesis : Weather is dependent on season

In [189…
```
#H0 = workingday is Independent on season
#H1 = workingday is dependent on season
df_count_season = pd.crosstab(df.workingday,df.season,margins=True,margins_name='Total')
df_count_season
```

Out[189]:

season	1	2	3	4	Total
workingday					
0	858	840	888	888	3474
1	1828	1893	1845	1846	7412
Total	2686	2733	2733	2734	10886

In [191…
```
from scipy import stats
```

In [192…
```
stat, p_value, dof, expected = stats.chi2_contingency(df_count_season)

print("p_value : ", p_value)
print("dof : ", dof)
print("expected : ", expected)
```

```
p_value :  0.9583429307736173
dof :  8
expected :  [[  857.17104538   872.16994305   872.16994305   872.48906853
   3474.        ]
 [ 1828.82895462  1860.83005695  1860.83005695  1861.51093147
   7412.        ]
 [ 2686.          2733.          2733.          2734.
  10886.        ]]
```

In [197…
```
alpha = 0.05
if p_value >= alpha:
    print('We Accept the Null Hypothesis : Workingday is Independent on season  ')
else:
    print('We reject the Null Hypothesis : Workingday is dependent on season ')
```

```
We Accept the Null Hypothesis : Workingday is Independent on season
```

Anova Testing

In [198…
```
gp1 = df[df['weather']==1]['count'].values
gp2 = df[df['weather']==2]['count'].values
gp3 = df[df['weather']==3]['count'].values
gp4 = df[df['weather']==4]['count'].values

gp5 = df[df['season']==1]['count'].values
gp6 = df[df['season']==2]['count'].values
gp7 = df[df['season']==3]['count'].values
gp8 = df[df['season']==4]['count'].values
```

In [199…
```
stats.f_oneway(gp1, gp2, gp3, gp4, gp5, gp6, gp7, gp8)
```

Out[199]:
```
F_onewayResult(statistic=127.96661249562491, pvalue=2.8074771742434642e-185)
```

In [200…
```
#H0 : count of bikes  is similar across various season
#Ha : count of bikes is different across various season
season_1 = df[df['season']==1]['count']
season_2 = df[df['season']==2]['count']
season_3 = df[df['season']==3]['count']
season_4 = df[df['season']==4]['count']
```

In [201…
```
statistic, p_value  = stats.f_oneway(season_1,season_2,season_3,season_4)
print("statistic : ", statistic)
print("p value : ", p_value)
```

```
statistic :  236.94671081032106
p value :  6.164843386499654e-149
```

In [202…
```
alpha = 0.05
if p_value >= alpha:
    print('We Accept the Null Hypothesis : count of bikes is similar across various season ')
else:
    print('We reject the Null Hypothesis : count of bikes is different across various season ')
```

```
We reject the Null Hypothesis : count of bikes is different across various season
```

In [203…
```
#H0 : count of bikes  is similar across various weather
#Ha : count of bikes is different across various weather
weather_1 = df[df['weather']==1]['count']
weather_2 = df[df['weather']==2]['count']
weather_3 = df[df['weather']==3]['count']
weather_4 = df[df['weather']==4]['count']
```

In [204…
```
statistic, p_value  = stats.f_oneway(weather_1,weather_2,weather_3,weather_4)
print("statistic : ", statistic)
print("p value : ", p_value)
```

```
statistic :  65.53024112793271
p value :  5.482069475935669e-42
```

In [205…
```
alpha = 0.05
if p_value >= alpha:
    print('We Accept the Null Hypothesis : count of bikes  is similar across various weather ')
else:
    print('We reject the Null Hypothesis : count of bikes is different across various weather ')
```

```
We reject the Null Hypothesis : count of bikes is different across various weather
```

Insights and Recommendations

- **In summer and fall seasons more bikes are rented as compared to other seasons.**
- **On the basis of month,in june they are more used the bikes.**
- **On the basis days,in friday they are more used bikes .**
- **In clear weather (weather 1) more bikes are used**
- **Casual, registered and count somewhat looks like Log Normal Distrinution**

- **Temp, atemp and humidity looks like they follows the Normal Distribution**
- **windspeed follows the binomial distribution.**
- **Whenever the humidity is less than 20, number of bikes rented is very very low.**
- **Whenever the temperature is less than 10, number of bikes rented is less.**
- **Whenever the windspeed is greater than 35, number of bikes rented is less.**
- **In summer and fall seasons the company should have more bikes in stock to be rented. Because the demand in these seasons is higher as compared to other seasons.**
- **In very low humid days, company should have less bikes in the stock to be rented.**
- **Whenever temprature is less than 10 or in very cold days, company should have less bikes.**
- **Whenever the windspeed is greater than 35 or in thunderstorms, company should have less bikes in stock to be rented.**

In [ ]: