In [62]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import t
import scipy.stats as stats
from statistics import mean, median, mode, stdev
import warnings
warnings.filterwarnings("ignore")
from scipy.stats import norm
import math
from numpy import cov
from math import sqrt
from scipy.stats import f_oneway
import calendar
import scipy.integrate as integrate
import scipy.special
from scipy.stats import levene
```

- 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 [2]:

```
ybs  = pd.read_csv(r'C:\Users\Acer\Downloads\bike_sharing.csv')
```

In [3]:

```
ybs
```

Out[3]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	c
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0000	
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	
...	...	...	...	...	...	...	...	...	...	
10881	2012-12-19 19:00:00	4	0	1	1	15.58	19.695	50	26.0027	
10882	2012-12-19 20:00:00	4	0	1	1	14.76	17.425	57	15.0013	
10883	2012-12-19 21:00:00	4	0	1	1	13.94	15.910	61	15.0013	
10884	2012-12-19 22:00:00	4	0	1	1	13.94	17.425	61	6.0032	
10885	2012-12-19 23:00:00	4	0	1	1	13.12	16.665	66	8.9981	

10886 rows × 12 columns

Basic info about the dataset

In [4]:

```
ybs.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
```

observations

- From the above its clear that we have a mix of categories that needed to be chaged to categories and some to int64

UNIQUE VALUES IN EACH COLUMN

In [5]:

```
colname = ['season','holiday','workingday','weather']
for col in colname:
    print("\nUnique values of ",col," are : ",list(ybs[col].unique()))
```

```

Unique values of  season  are :  [1, 2, 3, 4]

Unique values of  holiday  are :  [0, 1]

Unique values of  workingday  are :  [0, 1]

Unique values of  weather  are :  [1, 2, 3, 4]
```

Observations

- changing these values into terms for better understanding

In [6]:

```
ybs['season'] = ybs['season'].replace({1: 'spring', 2: 'summer' , 3: 'fall' , 4: 'winte
ybs['weather'] = ybs['weather'].replace({1: 'Clear', 2: 'Mist / cloudy' , 3: 'little ra
ybs['workingday'] = ybs['workingday'].replace({1: 'yes', 0: 'no'})
```

In [7]:

```
ybs
```

Out[7]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	c
0	2011-01-01 00:00:00	spring	0	no	Clear	9.84	14.395	81	0.0000	
1	2011-01-01 01:00:00	spring	0	no	Clear	9.02	13.635	80	0.0000	
2	2011-01-01 02:00:00	spring	0	no	Clear	9.02	13.635	80	0.0000	
3	2011-01-01 03:00:00	spring	0	no	Clear	9.84	14.395	75	0.0000	
4	2011-01-01 04:00:00	spring	0	no	Clear	9.84	14.395	75	0.0000	
...	...	...	...	...	...	...	...	...	...	
10881	2012-12-19 19:00:00	winter	0	yes	Clear	15.58	19.695	50	26.0027	
10882	2012-12-19 20:00:00	winter	0	yes	Clear	14.76	17.425	57	15.0013	
10883	2012-12-19 21:00:00	winter	0	yes	Clear	13.94	15.910	61	15.0013	
10884	2012-12-19 22:00:00	winter	0	yes	Clear	13.94	17.425	61	6.0032	
10885	2012-12-19 23:00:00	winter	0	yes	Clear	13.12	16.665	66	8.9981	

10886 rows × 12 columns

Observations

- creating new tables - Month , hour , date from 'datetime'

In [8]:

```
ybs["date"] = ybs.datetime.apply(lambda x : x.split()[0])
ybs["hour"] = ybs.datetime.apply(lambda x : x.split()[1].split(":")[0])
ybs['month'] = ybs['datetime'].apply(lambda x: x.split()[0].split('-')[1])
ybs['year'] = ybs['datetime'].apply(lambda x: x.split()[0].split('-')[0])
```

In [9]:

```
categorical_features = ['season', 'holiday', 'workingday', 'weather', 'month', 'year',

for feature in categorical_features:
    ybs[feature] = ybs[feature].astype("category")
```

In [10]:

```
ybs.drop('datetime' , axis = 1)
```

Out[10]:

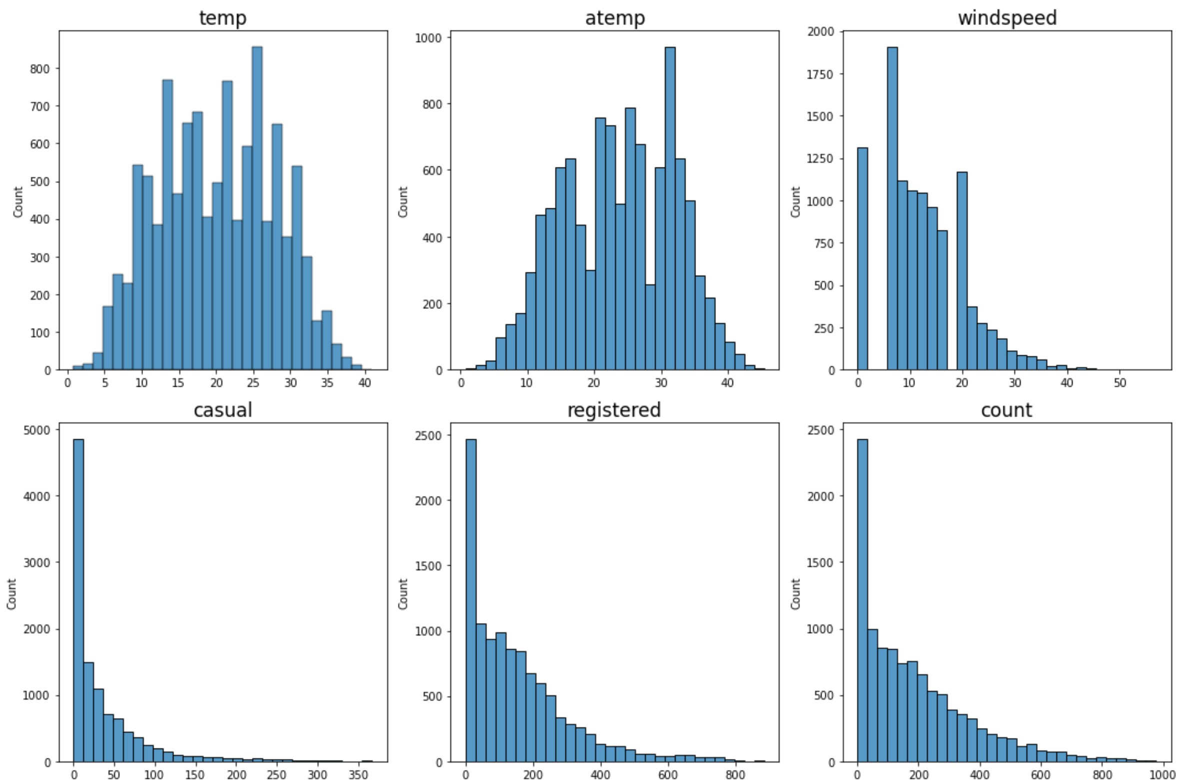
	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	reg
0	spring	0	no	Clear	9.84	14.395	81	0.0000	3	
1	spring	0	no	Clear	9.02	13.635	80	0.0000	8	
2	spring	0	no	Clear	9.02	13.635	80	0.0000	5	
3	spring	0	no	Clear	9.84	14.395	75	0.0000	3	
4	spring	0	no	Clear	9.84	14.395	75	0.0000	0	
...	...	...	...	...	...	...	...	...	...	
10881	winter	0	yes	Clear	15.58	19.695	50	26.0027	7	
10882	winter	0	yes	Clear	14.76	17.425	57	15.0013	10	
10883	winter	0	yes	Clear	13.94	15.910	61	15.0013	4	
10884	winter	0	yes	Clear	13.94	17.425	61	6.0032	12	
10885	winter	0	yes	Clear	13.12	16.665	66	8.9981	4	

10886 rows × 15 columns

EDA UNIVARIATE

In [11]:

```
cols = ['temp','atemp','windspeed','casual','registered', 'count']
fig, axes = plt.subplots(2,3,figsize = (10,5))
count = 0
for i in range(2):
    for j in range(3):
        s = cols[count+j]
        sns.histplot(ybs[s].values, ax = axes[i][j],bins = 30)
        axes[i][j].set_title(s,fontsize=17)
        fig=plt.gcf()
        fig.set_size_inches(15,10)
        plt.tight_layout()
    count = count+j+1
```

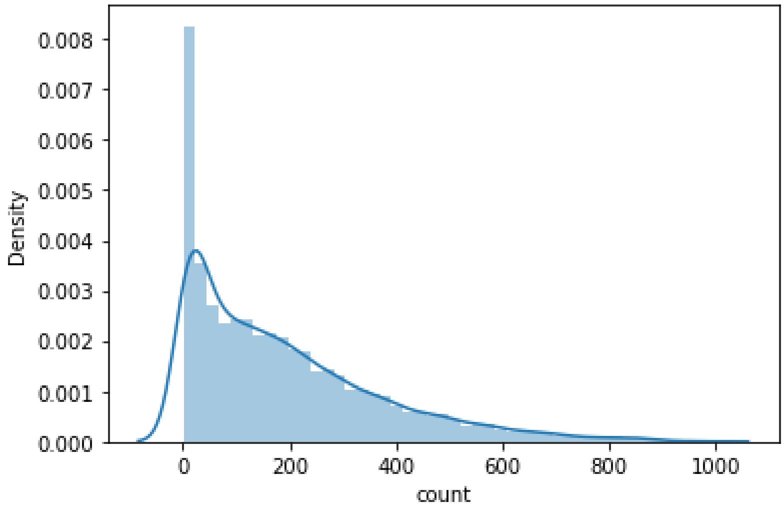


Observations

- count , Casual and Registered distribution looks right skewed
- The values in the casual is defenitely the highest which needed to be converted to registered which can be a potential company growth
- All the Casual , Registerd , Count distribution looks same and doing Box Cox / Log normal may convert it to Normal or near normal if we need to perform statistical tests
- The Windspeed , Temp and aTemp looks very irregular as its as close to a real environment data

In [12]:

```
sns.distplot(ybs['count'])
plt.show()
```

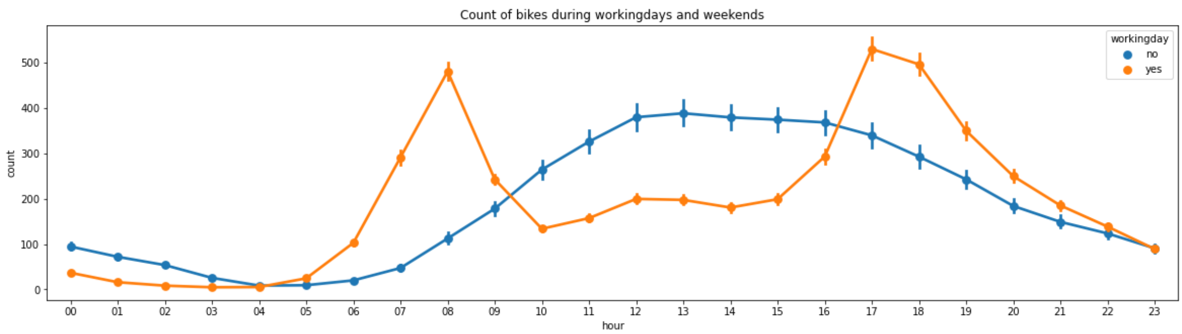


Bivariate analysis

In [13]:

```
fig, ax = plt.subplots(figsize = (20,5))
sns.pointplot(data = ybs , x ='hour' , y ='count', hue = 'workingday')
ax.set(title='Count of bikes during workingdays and weekends')
plt.show()

```



Observation

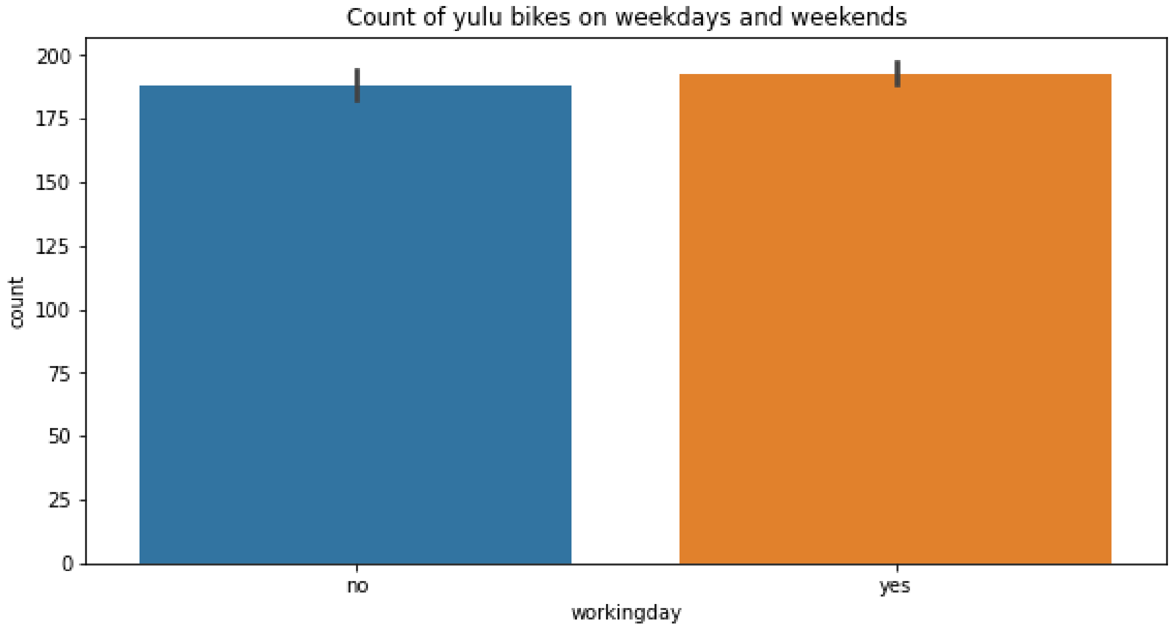
- we can clearly see that on the working days the demand increase at specific timing between 6AM - 9AM with a peak at 8AM and 4PM - 8PM with a spike at 5PM
- On OFF days the demand is usually rising during the mid day timing especially from 12 noon to 5PM with a gradual reduction till 8PM.
- These timing can really be worked on by the company so as to create a more Yulu biased transport withing the youngsters

In [14]:

```
fig, ax = plt.subplots(figsize=(10,5))
sns.barplot(data=ybs, x='workingday', y='count')
ax.set(title='Count of yulu bikes on weekdays and weekends')
```

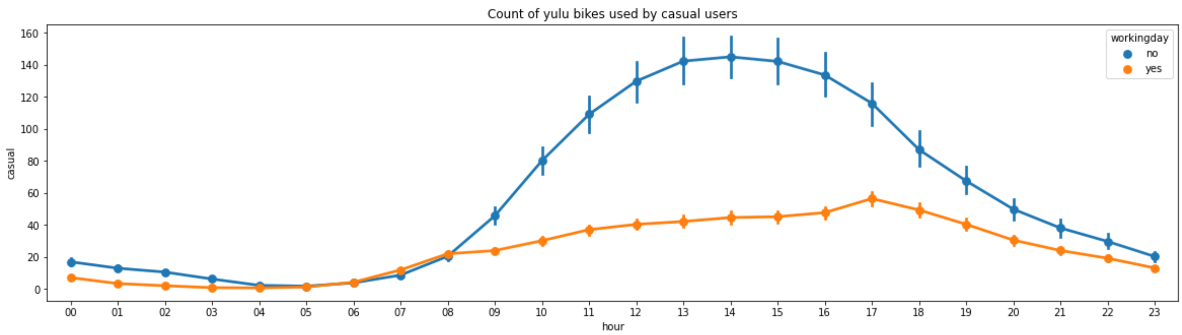
Out[14]:

```
[Text(0.5, 1.0, 'Count of yulu bikes on weekdays and weekends')]
```



In [15]:

```
fig, ax = plt.subplots(figsize = (20,5))
sns.pointplot(data = ybs , x ='hour' , y ='casual', hue = 'workingday')
ax.set(title='Count of yulu bikes used by casual users')
plt.show()
```

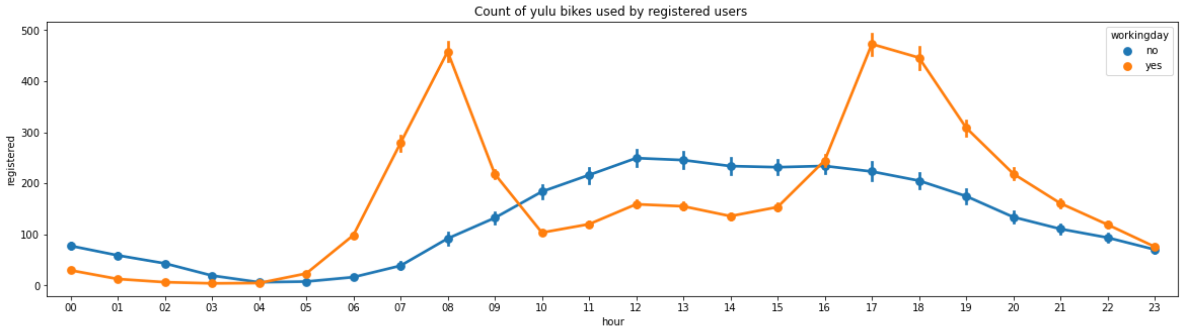


Observations

- the number of casual users are alot higher on the weekends or off days suggesting that the people who are keeping a job timings are registered more than others
- this are can be worked on by the company to make a transaction from casual to registered.
- keeping a reminder system of giving a small offer inorder to register mught help the publicity and reach of the YULU bikes
- the reach and demand of the bikes are too high for causal users on Off days suggest its popularity but the negligence in convertion

In [16]:

```
fig, ax = plt.subplots(figsize = (20,5))
sns.pointplot(data = ybs , x ='hour' , y ='registered', hue = 'workingday')
ax.set(title='Count of yulu bikes used by registered users ')
plt.show()
```



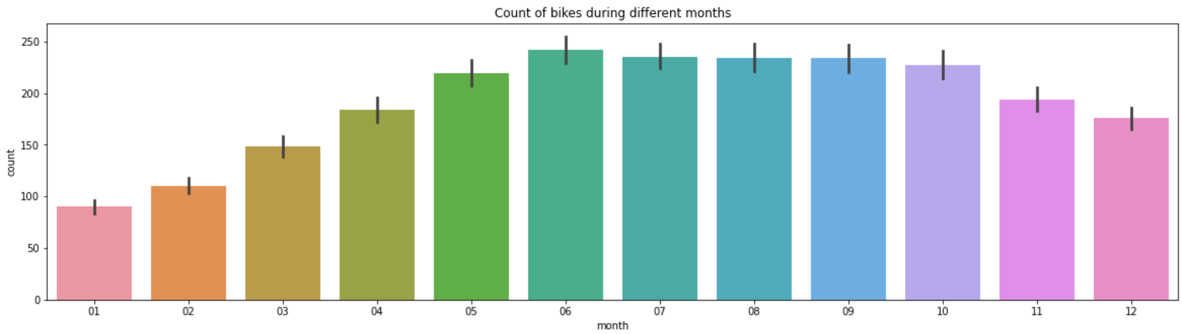
Bikes with respect to Months and seasons

In [17]:

```
fig ,ax = plt.subplots(figsize = (20,5))
sns.barplot(data = ybs , x= 'month', y = 'count')
ax.set(title='Count of bikes during different months')
```

Out[17]:

```
[Text(0.5, 1.0, 'Count of bikes during different months')]
```

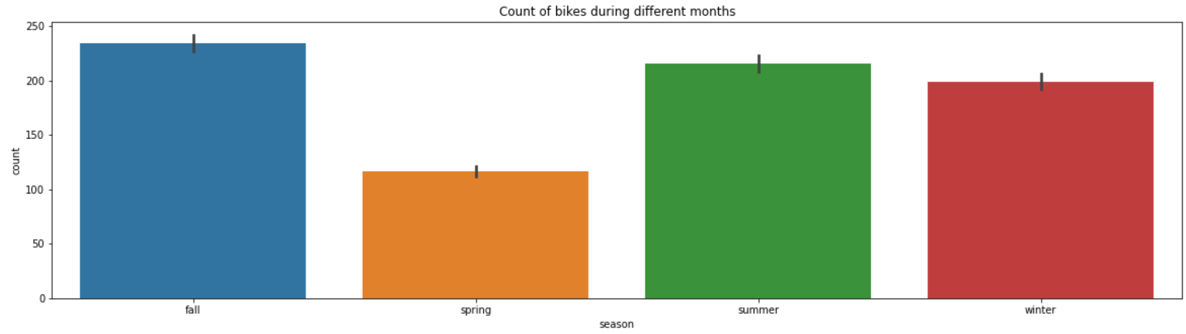


In [18]:

```
fig ,ax = plt.subplots(figsize = (20,5))
sns.barplot(data = ybs , x= 'season', y = 'count')
ax.set(title='Count of bikes during different months')
```

Out[18]:

```
[Text(0.5, 1.0, 'Count of bikes during different months')]
```



Observations

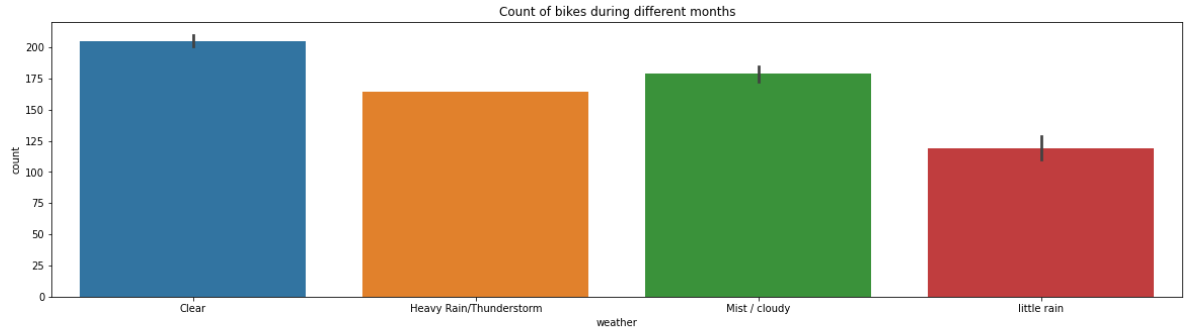
- The winter - Dec to Feb shows shows a sudden dip in the counts as most users will be home for vacation and by Feb its catching up
- The Spring time the count shows a linear growth in the demand but yet it is the time period with the lowest demand - this needs to be tackled
- The summer is set for the maximum demand and usage of the bikes from june to August
- The fall is also very similar with the summer on demands

In [19]:

```
fig ,ax = plt.subplots(figsize = (20,5))
sns.barplot(data = ybs , x= 'weather', y = 'count')
ax.set(title='Count of bikes during different months')
```

Out[19]:

```
[Text(0.5, 1.0, 'Count of bikes during different months')]
```



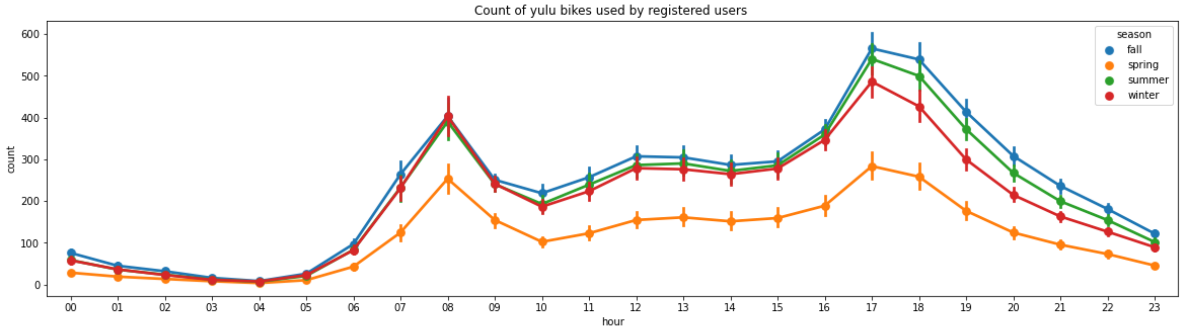
In [ ]:

```

```

In [20]:

```
fig, ax = plt.subplots(figsize = (20,5))
sns.pointplot(data = ybs , x ='hour' , y ='count', hue = 'season')
ax.set(title='Count of yulu bikes used by registered users ')
plt.show()
```



Observations

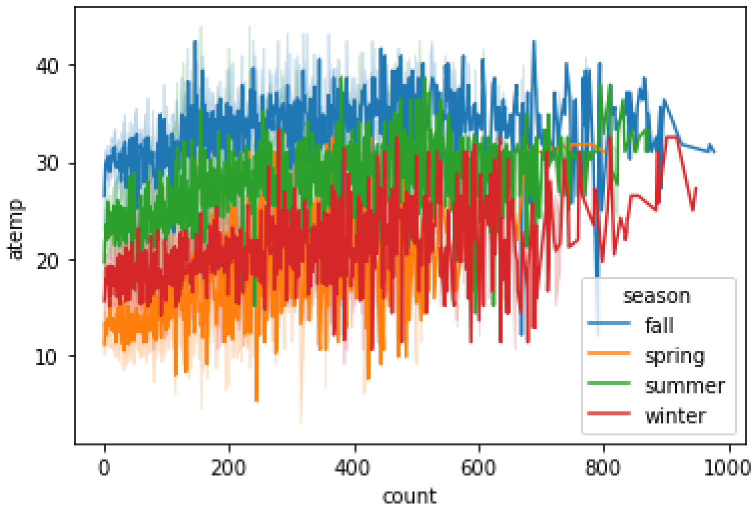
- The fall and summer demands for the bikes are almost identical and high followed closely by the Winter.
- The area of focus can be on the Spring which is clearly lagging behind

In [21]:

```
sns.lineplot(x = 'count',y = 'atemp',hue = 'season',data = ybs)
```

Out[21]:

```
<AxesSubplot:xlabel='count', ylabel='atemp'>
```

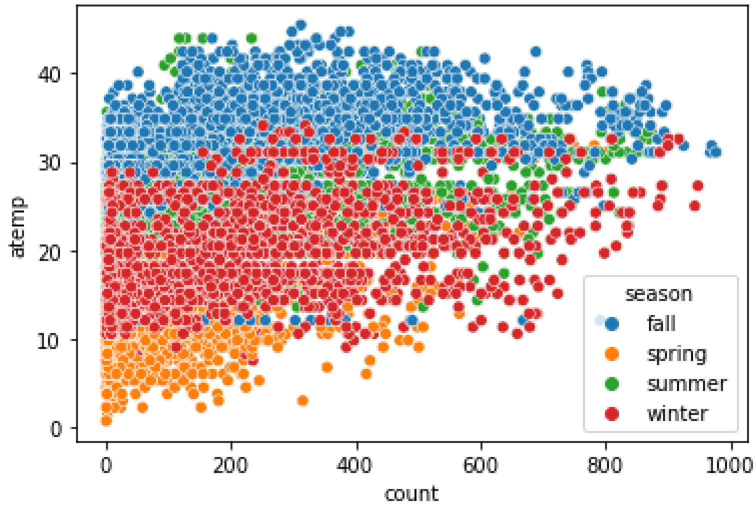


In [22]:

```
sns.scatterplot(x = 'count',y = 'atemp',hue = 'season',data = ybs)
```

Out[22]:

```
<AxesSubplot:xlabel='count', ylabel='atemp'>
```



Observations

- It was observed that the spring had the lowest demand and from this graph we can identify that during the spring the actual atemp(felt temperature) feels lower than the actual which maybe the reason behind low usagge

In [23]:

```
ybs.head()
```

Out[23]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual
0	2011-01-01 00:00:00	spring	0	no	Clear	9.84	14.395	81	0.0	3
1	2011-01-01 01:00:00	spring	0	no	Clear	9.02	13.635	80	0.0	8
2	2011-01-01 02:00:00	spring	0	no	Clear	9.02	13.635	80	0.0	5
3	2011-01-01 03:00:00	spring	0	no	Clear	9.84	14.395	75	0.0	3
4	2011-01-01 04:00:00	spring	0	no	Clear	9.84	14.395	75	0.0	0

HYPOTHESIS

Hypothesis Testing:

2- Sample T-Test to check if Working Day has an effect on the number of electric cycles rented.

In [24]:

```
working = ybs.loc[ybs['workingday'] == 'yes']['count']
offdays = ybs.loc[ybs['workingday'] == 'no']['count']
```

In [25]:

```
working
```

Out[25]:

```
47         5
48         2
49         1
50         3
51        30
        ...
10881    336
10882    241
10883    168
10884    129
10885     88
Name: count, Length: 7412, dtype: int64
```

Framework:

- 1 - Define Ho and Ha besed on what we want
- 2 - definne experiment with Test statistic
- 3 - Decide if its one tailed or two tailed
- 4 - Compute Tobs
- 5 - Fix alpha value
- 6 - Compare P(val) with alpha val

Experiment

- 1
  - Ho = Demand for Rental bikies are same on Offdays and Working days
  - Ha = Demand for Rental bikies are different on Offdays and Working days

- 2 Test stat = ((M1 - M2) / sqrt((s1/n1) + (s2/n2)))

- 3
  - This will be a two tailed test.

- 4 calculating Tobs

In [26]:

```
M1 = np.mean(working)
M2 = np.mean(offdays)

s1 = np.std(working)
s2 = np.std(offdays)

n1 = len(working)
n2 = len(offdays)

Tobs = ((M1 - M2) / sqrt((s1**2/n1) + (s2**2/n2)))
```

In [27]:

```
Tobs
```

Out[27]:

```
1.2364033017261238
```

In [73]:

```
#pval
pval = 2*(1- stats.t.cdf(x = Tobs , df = len(working) + len(offdays) - 2 ))
pval
```

Out[73]:

```
0.21633536943928133
```

In [29]:

```
stats.ttest_ind(working,offdays)
```

Out[29]:

```
Ttest_indResult(statistic=1.2096277376026694, pvalue=0.22644804226361348)
```

In [77]:

```
if pval > 0.05:
    print("Fail to Reject Null Hypothesis : \naverage of cycles rented on workingdays =
else:
    print("Reject Null Hypothesis :\naverage of cycles rented on workingdays != average
```

```
Fail to Reject Null Hypothesis :
average of cycles rented on workingdays = average  of cycles rented on offda
y
```

Observations

- Fromt the above Hypothesis testing we have failed to reject the null Hypothesis and accepted the fact that the demand for the bikes are same for working as well as Off days
- As opposed to the conclusion from the graphs thus emphasisng the usage of hypothesis testing

Chi-square test to check if Weather is dependent on the season

Framework:

- 1 - Define Ho and Ha besed on what we want
- 2 - definne experiment with Test statistic
- 3 - Decide if its one tailed or two tailed
- 4 - Compute Tobs
- 5 - Fix alpha value
- 6 - Compare P(val) with alpha val

Experiment

- 1
  - Ho = Weather is dependent on the season
  - Ha = Weather is dependent on the season

In [31]:

```
observed = pd.crosstab(index = ybs['season'] , columns = ybs['weather'], values = ybs['

observed
```

Out[31]:

weather / season	Clear	Heavy Rain/Thunderstorm	Mist / cloudy	little rain
fall	470116	0	139386	31160
spring	223009	164	76406	12919
summer	426350	0	134177	27755
winter	356588	0	157191	30255

In [32]:

```
a = pd.crosstab(index = ybs['season'] , columns = ybs['weather'], values = ybs['count']
a.columns = ['Clear', 'Heavy Rain/Thunderstorm' , 'Mist / cloudy', 'little rain' , 'row
a.index = ['fall' , 'spring' , 'summer', 'winter' , 'col_total']
a
```

Out[32]:

	Clear	Heavy Rain/Thunderstorm	Mist / cloudy	little rain	row_total
fall	470116	0	139386	31160	640662
spring	223009	164	76406	12919	312498
summer	426350	0	134177	27755	588282
winter	356588	0	157191	30255	544034
col_total	1476063	164	507160	102089	2085476

In [33]:

```
# a['row_total'][0:4]
# a.loc['col_total'][0:4]
```

In [34]:

```
expected =  np.outer(a['row_total'][0:4] ,a.loc['col_total'][0:4])/2085476
expected = pd.DataFrame(expected)
expected.columns = ['Clear', 'Heavy Rain/Thunderstorm' , 'Mist / cloudy', 'little rain'
expected.index = ['fall' , 'spring' , 'summer', 'winter']
expected
```

Out[34]:

	Clear	Heavy Rain/Thunderstorm	Mist / cloudy	little rain
fall	453449.223921	50.381097	155800.469495	31361.925488
spring	221180.553204	24.574568	75995.353425	15297.518802
summer	416375.587044	46.261980	143062.350811	28797.800166
winter	385057.635831	42.782356	132301.826269	26631.755545

In [35]:

```
stats.chi2_contingency(observed)
```

Out[35]:

```
(11769.559450959445,
 0.0,
 9,
 array([[4.53449224e+05, 5.03810967e+01, 1.55800469e+05, 3.13619255e+04],
        [2.21180553e+05, 2.45745681e+01, 7.59953534e+04, 1.52975188e+04],
        [4.16375587e+05, 4.62619795e+01, 1.43062351e+05, 2.87978002e+04],
        [3.85057636e+05, 4.27823557e+01, 1.32301826e+05, 2.66317555e+04]]))
```

In [36]:

```
chi_squared_stat = (((observed-expected)**2)/expected).sum().sum()
print(chi_squared_stat)
```

```
11769.559450959445
```

In [37]:

```
scipy.stats.chi2_contingency(observed= a)
```

Out[37]:

```
(11769.559450959445,
 0.0,
 16,
 array([[4.53449224e+05, 5.03810967e+01, 1.55800469e+05, 3.13619255e+04,
         6.40662000e+05],
        [2.21180553e+05, 2.45745681e+01, 7.59953534e+04, 1.52975188e+04,
         3.12498000e+05],
        [4.16375587e+05, 4.62619795e+01, 1.43062351e+05, 2.87978002e+04,
         5.88282000e+05],
        [3.85057636e+05, 4.27823557e+01, 1.32301826e+05, 2.66317555e+04,
         5.44034000e+05],
        [1.47606300e+06, 1.64000000e+02, 5.07160000e+05, 1.02089000e+05,
         2.08547600e+06]]))
```

In [38]:

```
df=(len(observed)-1)*(len(observed.columns)-1)
```

In [39]:

```
T_critical=stats.chi2.ppf(0.95,df)
T_critical
```

Out[39]:

```
16.918977604620448
```

In [40]:

```
if chi_squared_stat > T_critical:
    print("Reject Null Hypothesis : \nWeather and Season are dependent variables")
else:
    print("Failed to Reject Null Hypothesis :\nWeather and Season are independent Varia
```

```
Reject Null Hypothesis :
Weather and Season are dependent variables
```

Anova

ANNOVA to check if No. of cycles rented is similar or different in different 1. weather 2. season

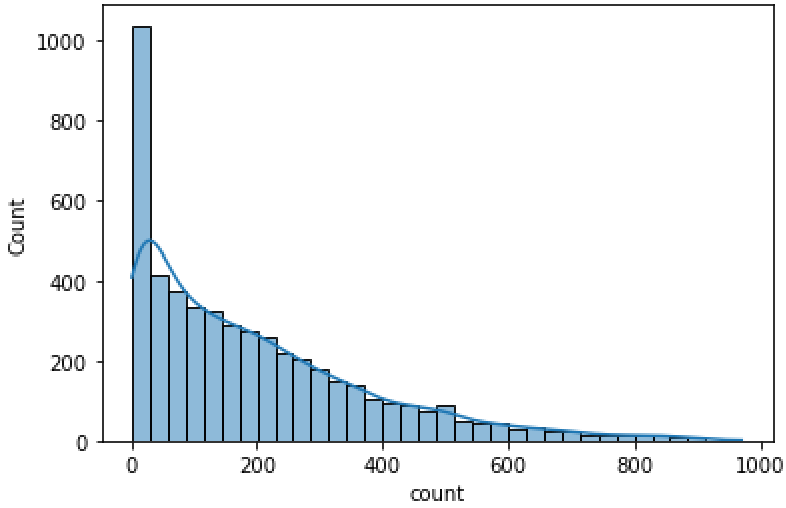
Checking is the data is normally distributed

In [59]:

```
sns.histplot((ybs['count'].sample(5000)), kde = True)
```

Out[59]:

```
<AxesSubplot:xlabel='count', ylabel='Count'>
```

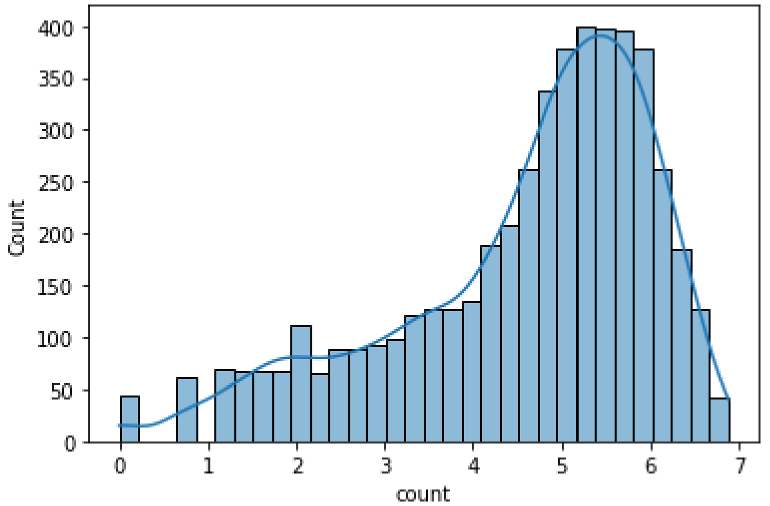


In [57]:

```
#Taking the log of the above distribution sample as it's not normal.
sns.histplot(np.log(ybs['count'].sample(5000)), kde = True)
```

Out[57]:

```
<AxesSubplot:xlabel='count', ylabel='Count'>
```



In [58]:

```
stats.shapiro(ybs['count'].sample(5000))
```

Out[58]:

```
ShapiroResult(statistic=0.8756548762321472, pvalue=0.0)
```

- Sapiro test shows that is not normally distributed
- now we will try the test for variance to see if it holds true

In [69]:

```
anova_weather1 = (ybs.loc[ybs['weather'] == 'Clear']['count'])
anova_weather2 =  (ybs.loc[ybs['weather'] == 'Mist / cloudy']['count'])
anova_weather3 =  (ybs.loc[ybs['weather'] == 'little rain']['count'])
anova_weather4 =  (ybs.loc[ybs['weather'] == 'Heavy Rain/Thunderstorm']['count'])
```

In [70]:

```
levene(anova_weather1, anova_weather2, anova_weather3,anova_weather4)
```

Out[70]:

```
LeveneResult(statistic=54.85106195954556, pvalue=3.504937946833238e-35)
```

since the pval is < than the the alpha we reject the hypothesis of equal varience

Even after taking log, the distribution is not exactly normal. So our assumption doesn't holds true. Also, we have confirmed with the statistical test -Shapiro wik test that the series is not normal.Still we will be going ahead with the test just to check the results.

In [41]:

```
ybs.head()
```

Out[41]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual
0	2011-01-01 00:00:00	spring	0	no	Clear	9.84	14.395	81	0.0	3
1	2011-01-01 01:00:00	spring	0	no	Clear	9.02	13.635	80	0.0	8
2	2011-01-01 02:00:00	spring	0	no	Clear	9.02	13.635	80	0.0	5
3	2011-01-01 03:00:00	spring	0	no	Clear	9.84	14.395	75	0.0	3
4	2011-01-01 04:00:00	spring	0	no	Clear	9.84	14.395	75	0.0	0

In [42]:

```
anova_weather1 = np.log(ybs.loc[ybs['weather'] == 'Clear']['count'])
anova_weather2 =  np.log(ybs.loc[ybs['weather'] == 'Mist / cloudy']['count'])
anova_weather3 =  np.log(ybs.loc[ybs['weather'] == 'little rain']['count'])
anova_weather4 =  np.log(ybs.loc[ybs['weather'] == 'Heavy Rain/Thunderstorm']['count']]
```

In [46]:

```
stats.f_oneway(anova_weather1,anova_weather2,anova_weather3)
pvalue=5.716684801108396e-33
T_critical = 0.05
```

In [56]:

```
if pvalue > T_critical:
    print("Reject Null Hypothesis : \ncount of bikes rented is samet in different types
else:
    print("Failed to Reject Null Hypothesis :\ncount of bikes rented is different in di
```

```
Failed to Reject Null Hypothesis :
count of bikes rented is different in different types of weather
```

Anova Season

In [ ]:

```
anova_summer = (ybs.loc[ybs['season'] == 'summer']['count'])
anova_fall =  (ybs.loc[ybs['season'] == 'fall']['count'])
anova_spring =  (ybs.loc[ybs['season'] == 'spring']['count'])
anova_winter =  (ybs.loc[ybs['season'] == 'winter']['count'])
```

In [71]:

```
levene(anova_summer, anova_fall, anova_spring,anova_winter)
```

Out[71]:

```
LeveneResult(statistic=9.640605587638786, pvalue=2.3678125658230693e-06)
```

since the pval is < than the the alpha we reject the hypothesis of equal varience

Here also the data is ot normally distributed so it fails the assumptions of ANOVA but still carrying forward in doing the analysis

In [50]:

```
anova_summer = np.log(ybs.loc[ybs['season'] == 'summer']['count'])
anova_fall =  np.log(ybs.loc[ybs['season'] == 'fall']['count'])
anova_spring =  np.log(ybs.loc[ybs['season'] == 'spring']['count'])
anova_winter =  np.log(ybs.loc[ybs['season'] == 'winter']['count'])
```

In [51]:

```
stats.f_oneway(anova_summer,anova_fall,anova_spring,anova_winter)
```

Out[51]:

```
F_onewayResult(statistic=192.44768979509692, pvalue=1.3071364586230693e-121)
```

In [52]:

```
pvalue=1.3071364586230693e-121
T_critical = 0.05
```

In [55]:

```
if pvalue > T_critical:
    print("Reject Null Hypothesis : \ncount of bikes rented is same in different types
else:
    print("Failed to Reject Null Hypothesis :\ncount of bikes rented is different in di
```

```
Failed to Reject Null Hypothesis :
count of bikes rented is different in different types of seasons
```

In [ ]:

```

```

Recommendations and Insights

- Missing data or Null values are not present.
- count , Casual and Registered distribution looks right skewed
- The values in the casual is defenitely the highest which needed to be converted to registered which can be a potential company growth

- All the Casual , Registerd , Count distribution looks same and doing Box Cox / Log normal may convert it to Normal or near normal if we need to perform statistical tests
- The Windspeed , Temp and aTemp looks very irregular as its as close to a real environment data
- we can clearly see that on the working days the demand increase at specific timing between 6AM - 9AM with a peak at 8AM and 4PM - 8PM with a spike at 5PM
- On OFF days the demand is usually rising during the mid day timing especially from 12 noon to 5PM with a gradual reduction till 8PM.
- These timing can really be worked on by the company so as to create a more Yulu biased transport withing the youngsters
- the number of casual users are alot higher on the weekends or off days suggesting that the people who are keeping a job timings are registered more than others
- this are can be worked on by the company to make a transaction from casual to registered.
- keeping a reminder system of giving a small offer inorder to register mught help the publicity and reach of the YULU bikes
- the reach and demand of the bikes are too high for causal users on Off days suggest its popularity but the negligence in convertion
- The winter - Dec to Feb shows shows a sudden dip in the counts as most users will be home for vacation and by Feb its catching up
- The Spring time the count shows a linear growth in the demand but yet it is the time period with the lowest demand - this needs to be tackled
- The summer is set for the maximum demand and usage of the bikes from june to August
- The fall is also very similar with the summer on demands

- 2 sample t-test:
  - The distribution of the samples is right-skwed and it's not normal which violates is our assumption for conducting 2 sample t test with unequal variance .Hence log-transformation done
  - We got a p-value of 0.22 which is greater than 0.05 and hence we can say that we can accept the null hypothesis.
  - Conclusion : As the p value > alpha(0.05) , we accept H0 and thus we can say that the count of renting of bikes in both working and non-working days is equal.
  - Fromt the above Hypothesis testing we have failed to reject the null Hypothesis and accepted the fact that the demand for the bikes are same for working as well as Off days as opposed to the conclusion from the graphs thus emphasisng the usage of hypothesis testing

```
- Chi-Square test:
    - p- value (6.734426550686341e-08) < alpha(0.05) --> so we can reject H0 Which means weather and seasons have a significant dependency.
    - We can conclude that our tstat > tcritical , we can reject the H0 , so weather and Seasons are dependent on each other.

```

- One-way Anova:
  - Even after taking log, the distribution is not exactly normal. So our assumption doesn't holds true. Also, we have confirmed with the statistical test -Shapiro wilk test that the series is not normal. Still we will be going ahead with the test just to check the results.
  - As the p value < alpha(0.05) , we reject H0 and thus we can conclude that count of bikes differs with a change in weather_code.
  - As the p value < alpha(0.05) , we reject H0 and thus we can conclude that count of bikes differs with a change in season.

Conclusion

- In order to conclude, we can say that the major factors affecting the count of bikes rented are season and weather_code. The working and non working days can't be considered as a significant factor in predicting the future of rental business. At the same time, the business team must focus on the months other than winter months for increasing the bike parking zones as during the winter months of (Nov, Dec, Jan, Feb), theres's a considerable dip in the cnt. So the team can utilize these months for serving some other purpose such as renting electric cars, etc which can be a comfortable means for commute in cold