LinearRegression_expt

June 11, 2026

[3]:
```
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error
import joblib
```

[4]:
```
df = pd.read_csv("hour.csv")
```

[5]:
```
df.head(3)
```

[5]:

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	01/01/2011	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0	3	13	16
1	2	01/01/2011	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0	8	32	40
2	3	01/01/2011	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0	5	27	32

[80]:
```
df.describe()
```

[80]:

	instant	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
count	17379.0000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000	17379.000000
mean	8690.0000	2.501640	0.502561	6.537775	11.546752	0.028770	3.003683	0.682721	1.425283	0.496987	0.475775	0.627229	0.190098	35.676218	153.786869	189.463088
std	5017.0295	1.106918	0.500008	3.438776	6.914405	0.167165	2.005771	0.465431	0.639357	0.192556	0.171850	0.192930	0.122340	49.305030	151.357286	181.387599
min	1.0000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.020000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
25%	4345.5000	2.000000	0.000000	4.000000	6.000000	0.000000	1.000000	0.000000	1.000000	0.340000	0.333300	0.480000	0.104500	4.000000	34.000000	40.000000
50%	8690.0000	3.000000	1.000000	7.000000	12.000000	0.000000	3.000000	1.000000	1.000000	0.500000	0.484800	0.630000	0.194000	17.000000	115.000000	142.000000
75%	13034.5000	3.000000	1.000000	10.000000	18.000000	0.000000	5.000000	1.000000	2.000000	0.660000	0.621200	0.780000	0.253700	48.000000	220.000000	281.000000
max	17379.0000	4.000000	1.000000	12.000000	23.000000	1.000000	6.000000	1.000000	4.000000	1.000000	1.000000	1.000000	0.850700	367.000000	886.000000	977.000000

```
min        0.000000      0.000000      0.000000      1.000000      0.020000
25%        0.000000      1.000000      0.000000      1.000000      0.340000
50%        0.000000      3.000000      1.000000      1.000000      0.500000
75%        0.000000      5.000000      1.000000      2.000000      0.660000
max        1.000000      6.000000      1.000000      4.000000      1.000000

              atemp           hum     windspeed        casual    registered  \
count  17379.000000  17379.000000  17379.000000  17379.000000  17379.000000
mean       0.475775      0.627229      0.190098     35.676218    153.786869
std        0.171850      0.192930      0.122340     49.305030    151.357286
min        0.000000      0.000000      0.000000      0.000000      0.000000
25%        0.333300      0.480000      0.104500      4.000000     34.000000
50%        0.484800      0.630000      0.194000     17.000000    115.000000
75%        0.621200      0.780000      0.253700     48.000000    220.000000
max        1.000000      1.000000      0.850700    367.000000    886.000000

                cnt
count  17379.000000
mean     189.463088
std      181.387599
min        1.000000
25%       40.000000
50%      142.000000
75%      281.000000
max      977.000000
```

[6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17379 entries, 0 to 17378
Data columns (total 17 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   instant     17379 non-null  int64
 1   dteday      17379 non-null  object
 2   season      17379 non-null  int64
 3   yr          17379 non-null  int64
 4   mnth        17379 non-null  int64
 5   hr          17379 non-null  int64
 6   holiday     17379 non-null  int64
 7   weekday     17379 non-null  int64
 8   workingday  17379 non-null  int64
 9   weathersit  17379 non-null  int64
 10  temp        17379 non-null  float64
 11  atemp       17379 non-null  float64
 12  hum         17379 non-null  float64
 13  windspeed   17379 non-null  float64
 14  casual      17379 non-null  int64
```

```
 15  registered  17379 non-null  int64
 16  cnt         17379 non-null  int64
dtypes: float64(4), int64(12), object(1)
memory usage: 2.3+ MB
```

[8]:
```
df.corr(numeric_only=True)["cnt"].abs().sort_values(ascending=False)
```

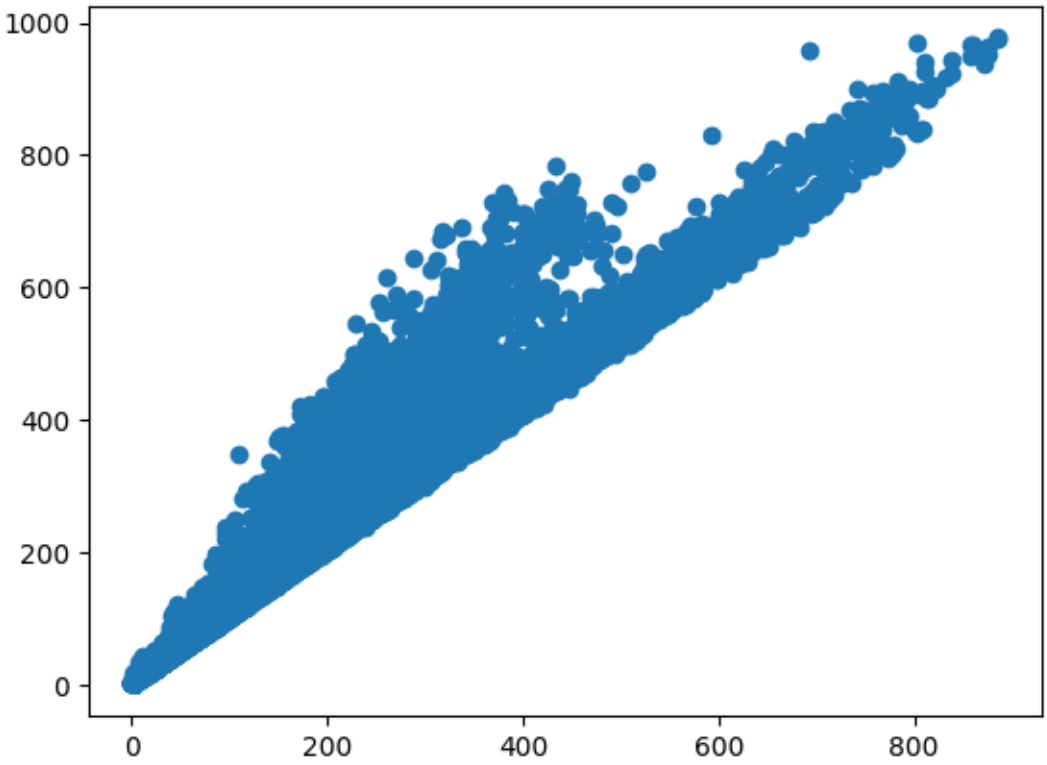
[8]:
```
cnt           1.000000
registered    0.972151
casual        0.694564
temp          0.404772
atemp         0.400929
hr            0.394071
hum           0.322911
instant       0.278379
yr            0.250495
season        0.178056
weathersit    0.142426
mnth          0.120638
windspeed     0.093234
holiday       0.030927
workingday    0.030284
weekday       0.026900
Name: cnt, dtype: float64
```

[9]:
```
df.corr(numeric_only=True)["cnt"].sort_values(ascending=False)
```

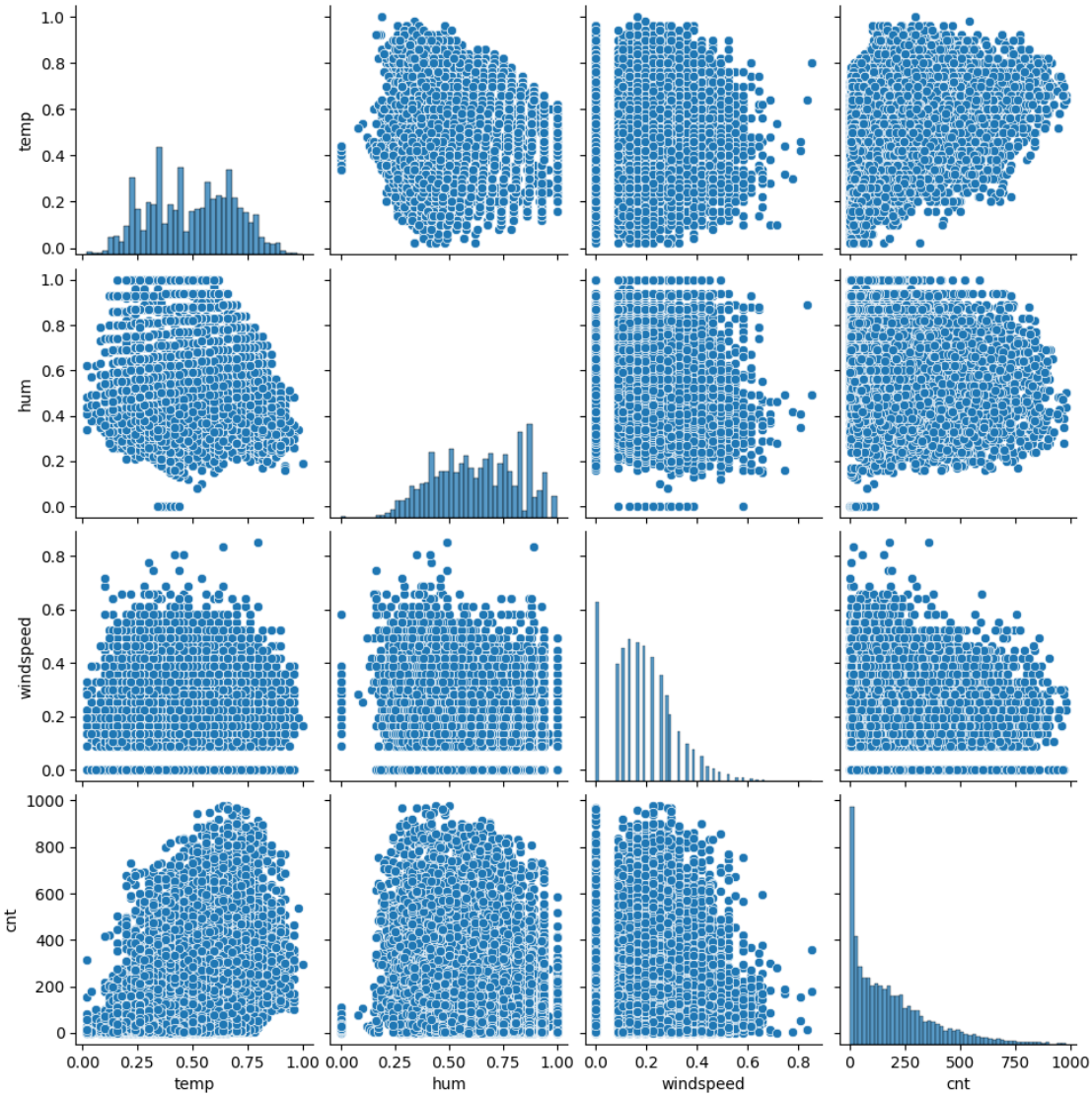
[9]:
```
cnt           1.000000
registered    0.972151
casual        0.694564
temp          0.404772
atemp         0.400929
hr            0.394071
instant       0.278379
yr            0.250495
season        0.178056
mnth          0.120638
windspeed     0.093234
workingday    0.030284
weekday       0.026900
holiday      -0.030927
weathersit   -0.142426
hum          -0.322911
Name: cnt, dtype: float64
```

[86]:
```
plt.scatter(df['registered'], df['cnt'])
```

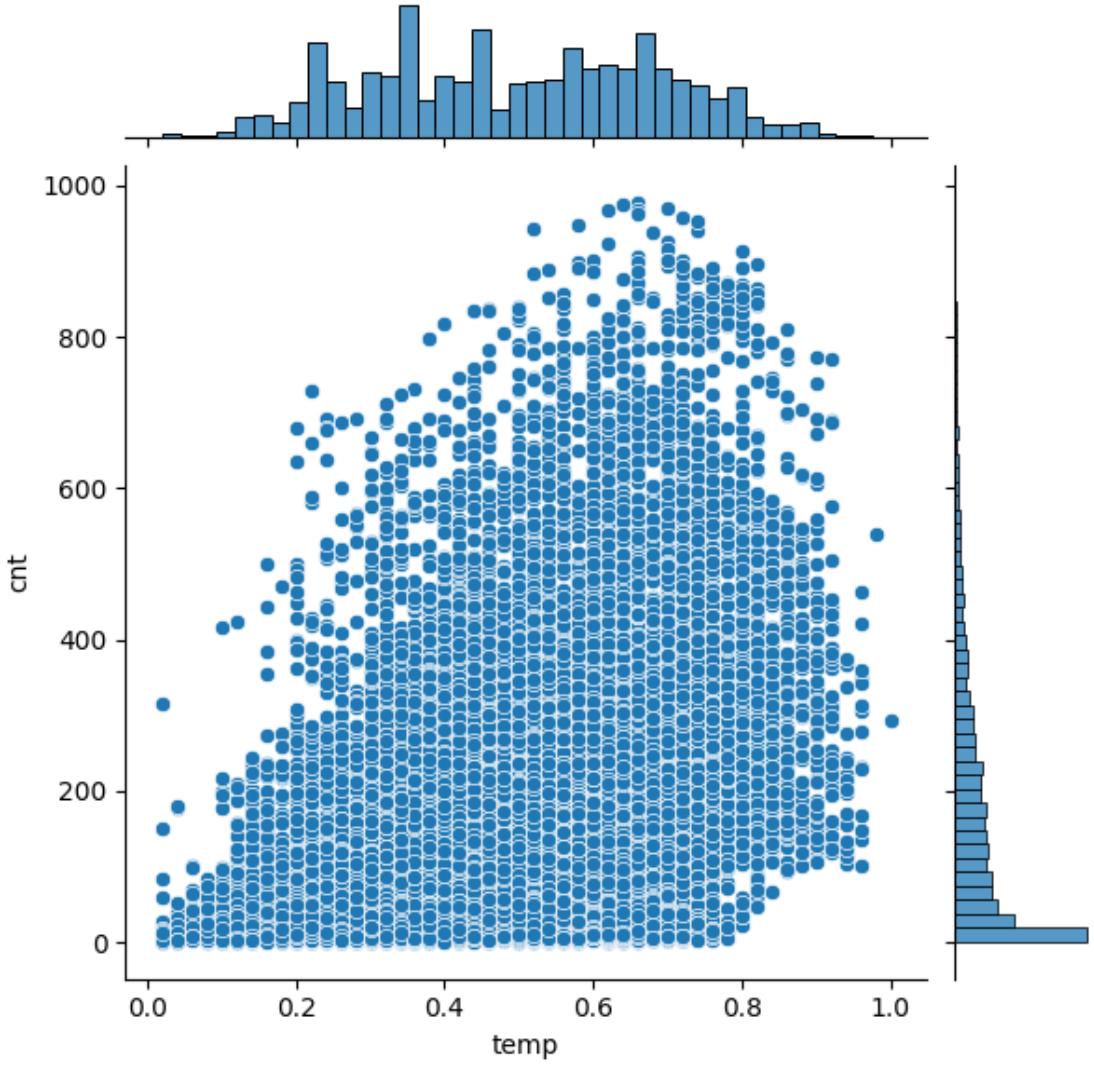
[86]:
```
<matplotlib.collections.PathCollection at 0x1c80fd681d0>
```



[89]:
```
import seaborn as sns
sns.pairplot(df[["temp", "hum", "windspeed", "cnt"]])
plt.show()
```

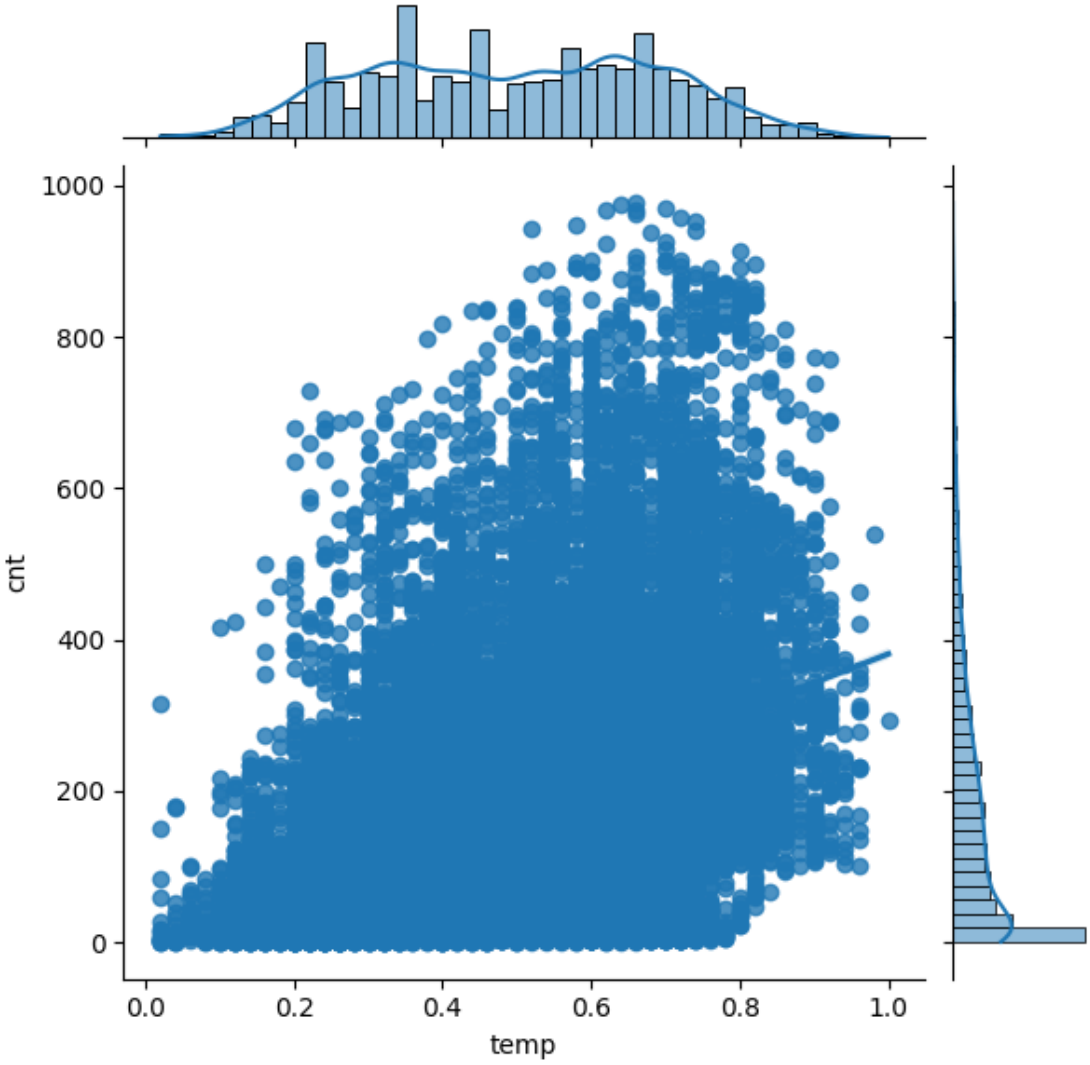


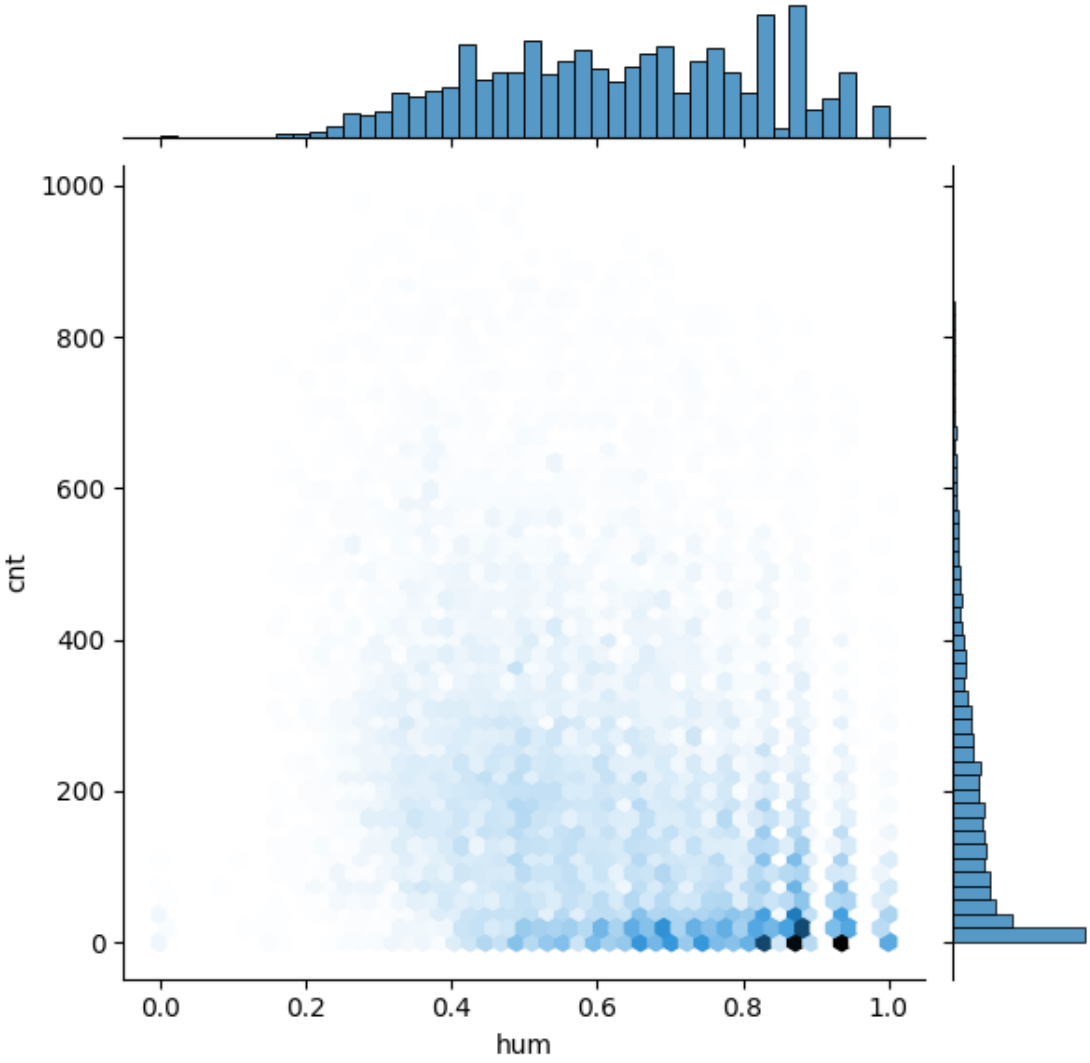
[90]:
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.jointplot(data=df, x="temp", y="cnt", kind="scatter")
plt.show()
```

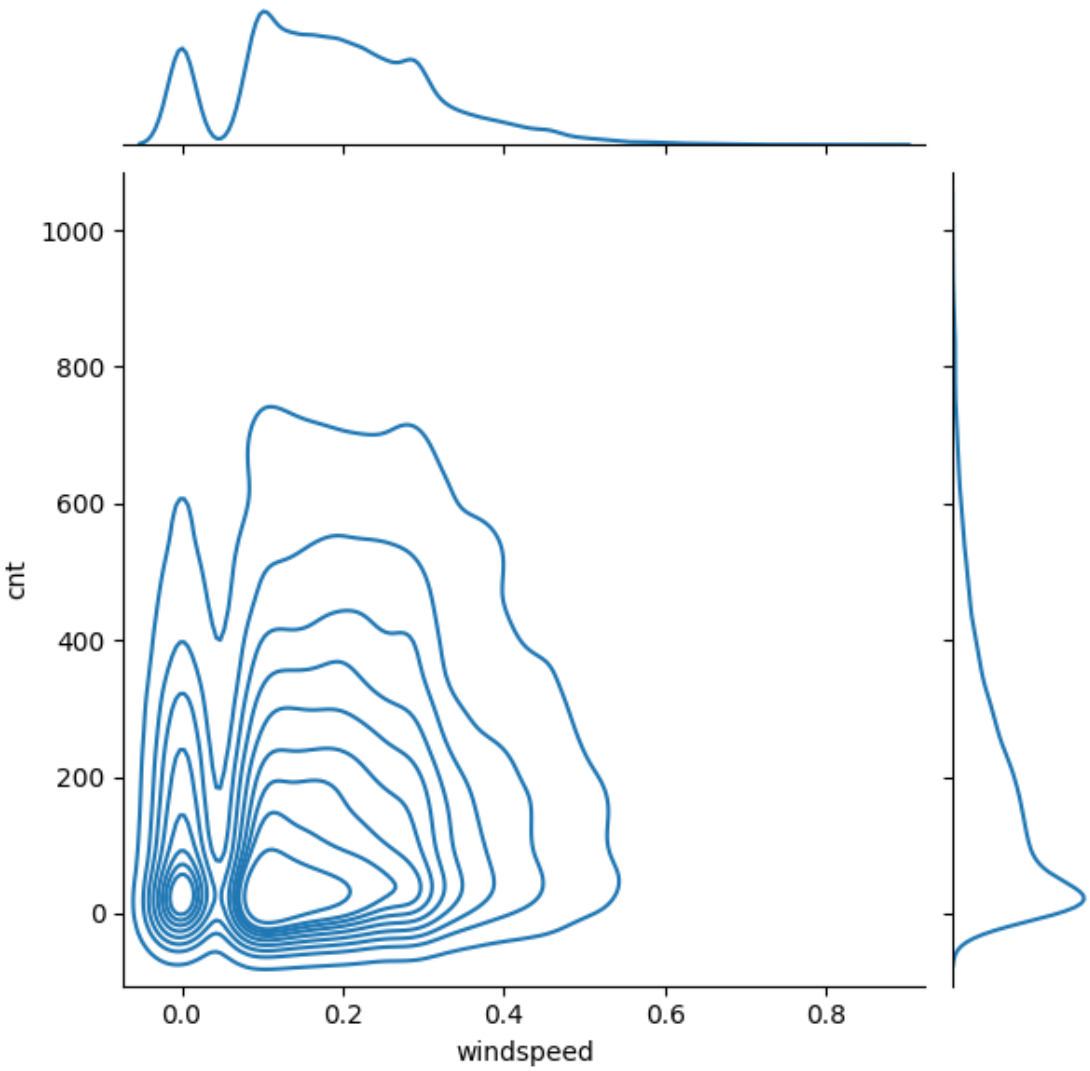


[91]:
```
sns.jointplot(data=df, x="temp", y="cnt", kind="reg")
sns.jointplot(data=df, x="hum", y="cnt", kind="hex")
sns.jointplot(data=df, x="windspeed", y="cnt", kind="kde")
```

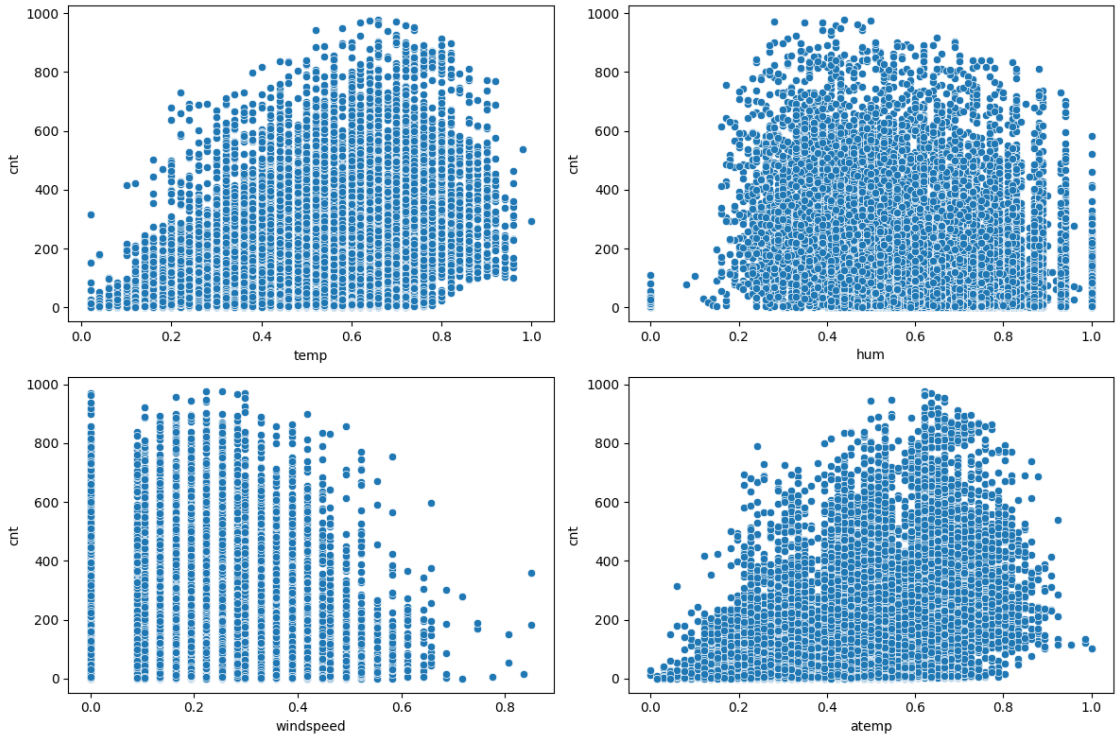
[91]: <seaborn.axisgrid.JointGrid at 0x1c8296b9190>


1000
800
600
400
200
0
cnt
0.0
0.2
0.4
0.6
0.8
1.0
temp

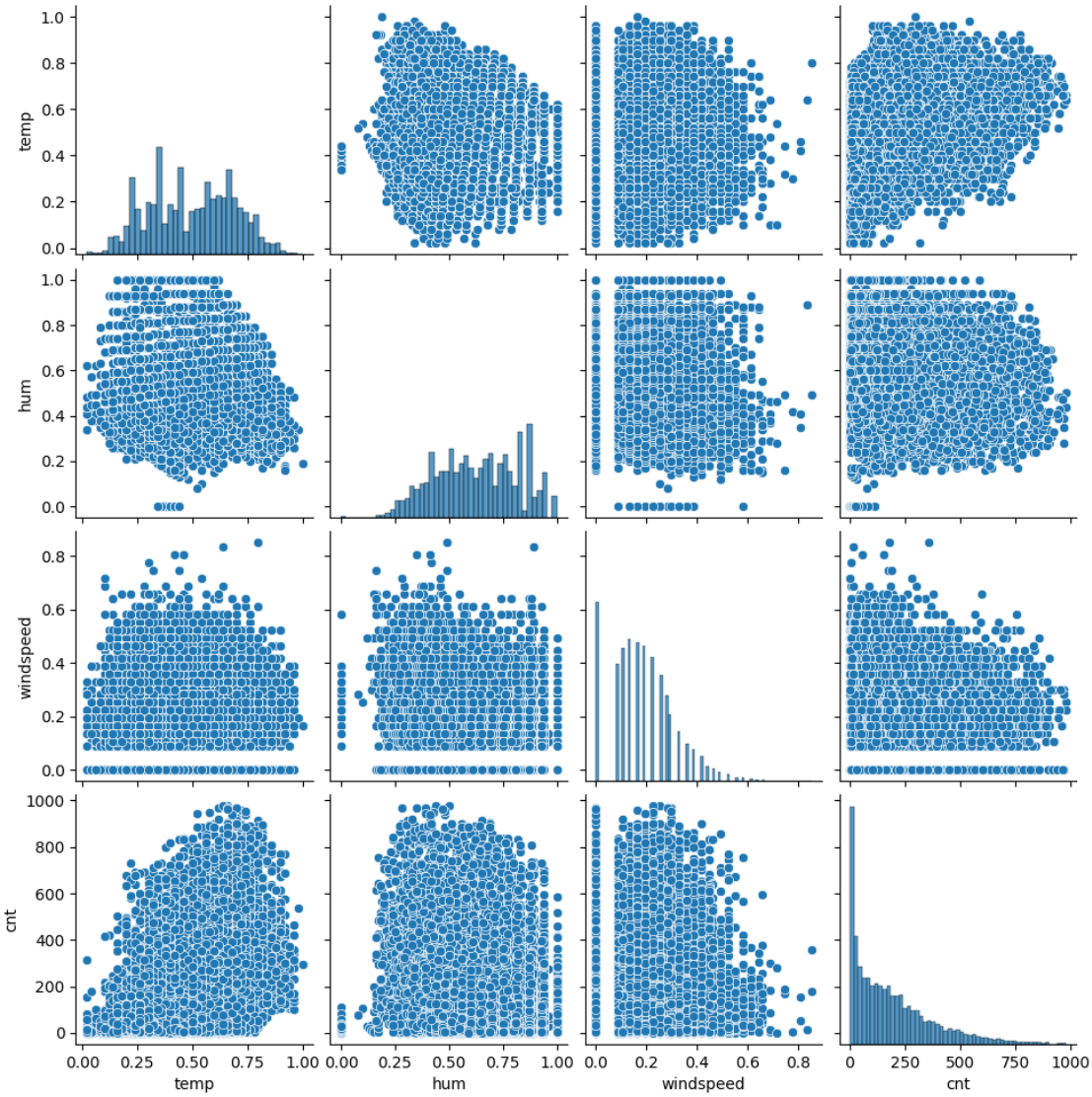

1000
800
600
400
200
0
cnt
0.0
0.2
0.4
0.6
0.8
1.0
hum



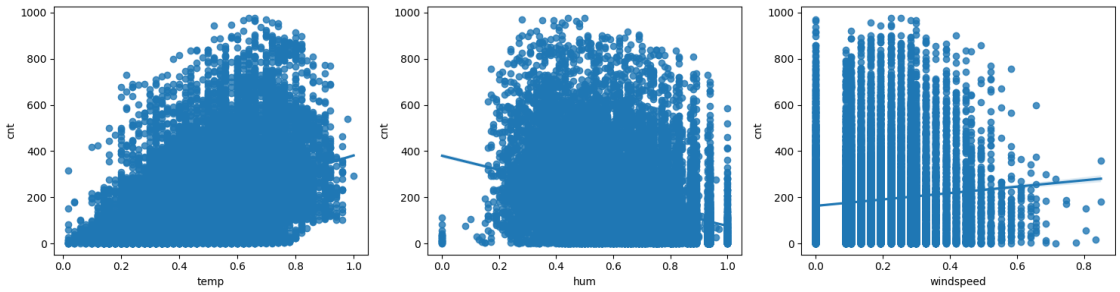
[92]:
```
import matplotlib.pyplot as plt
import seaborn as sns

fig, axes = plt.subplots(2, 2, figsize=(12, 8))

sns.scatterplot(data=df, x="temp", y="cnt", ax=axes[0, 0])
sns.scatterplot(data=df, x="hum", y="cnt", ax=axes[0, 1])
sns.scatterplot(data=df, x="windspeed", y="cnt", ax=axes[1, 0])
sns.scatterplot(data=df, x="atemp", y="cnt", ax=axes[1, 1])

plt.tight_layout()
plt.show()
```



[93]:
```
sns.pairplot(df[["temp", "hum", "windspeed", "cnt"]])
plt.show()
```



[94]:
```
fig, axes = plt.subplots(1, 3, figsize=(15, 4))

sns.regplot(data=df, x="temp", y="cnt", ax=axes[0])
sns.regplot(data=df, x="hum", y="cnt", ax=axes[1])
sns.regplot(data=df, x="windspeed", y="cnt", ax=axes[2])

plt.tight_layout()
plt.show()
```



[39]:
```
cols_to_drop = ["instant", "dteday", "casual", "registered"]
df = df.dropcolums=cols_to_drop)]
```

[40]:
```
df.head(3)
```

[40]:

	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	cnt
0	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0	16
1	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0	40
2	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0	32

[58]:
```
y = df["cnt"]
X = df.iloc[:, :-1]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,␣
↪random_state = 120)
```

[59]:
```
model = LinearRegression()
model.fit(X_train, y_train)
```

[59]:
```
LinearRegression()
```

[60]:
```
y_pred = model.predict(X_test)
```

[74]:
```
print("R2 score : ", round(r2_score(y_test, y_pred), 4))
print("RMSE : ", mean_squared_error(y_test, y_pred) ** 0.5)
```

```
R2 score :  0.3839
RMSE :  142.8827144770966
```

[34]:
```
X
```

[34]:

	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed
0	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0000
1	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0000
2	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0000
3	1	0	1	3	0	6	0	1	0.24	0.2879	0.75	0.0000
4	1	0	1	4	0	6	0	1	0.24	0.2879	0.75	0.0000
…	…	..	…	..	…	…	…	…	…	…	…	…
17374	1	1	12	19	0	1	1	2	0.26	0.2576	0.60	0.1642
17375	1	1	12	20	0	1	1	2	0.26	0.2576	0.60	0.1642
17376	1	1	12	21	0	1	1	1	0.26	0.2576	0.60	0.1642
17377	1	1	12	22	0	1	1	1	0.26	0.2727	0.56	0.1343
17378	1	1	12	23	0	1	1	1	0.26	0.2727	0.65	0.1343

[17379 rows x 12 columns]

[75]:
```
split_idx = int(len(df) * 0.7)
X_train, X_test = X.iloc[:split_idx], X.iloc[split_idx:]
y_train, y_test = y.iloc[:split_idx], y.iloc[split_idx:]
```

[76]:
```
df.corr(numeric_only=True)["cnt"].sort_values(ascending=False)
```

[76]:
```
cnt           1.000000
temp          0.404772
atemp         0.400929
hr            0.394071
yr            0.250495
season        0.178056
mnth          0.120638
windspeed     0.093234
workingday    0.030284
weekday       0.026900
holiday      -0.030927
weathersit   -0.142426
hum          -0.322911
```

```
Name: cnt, dtype: float64
```

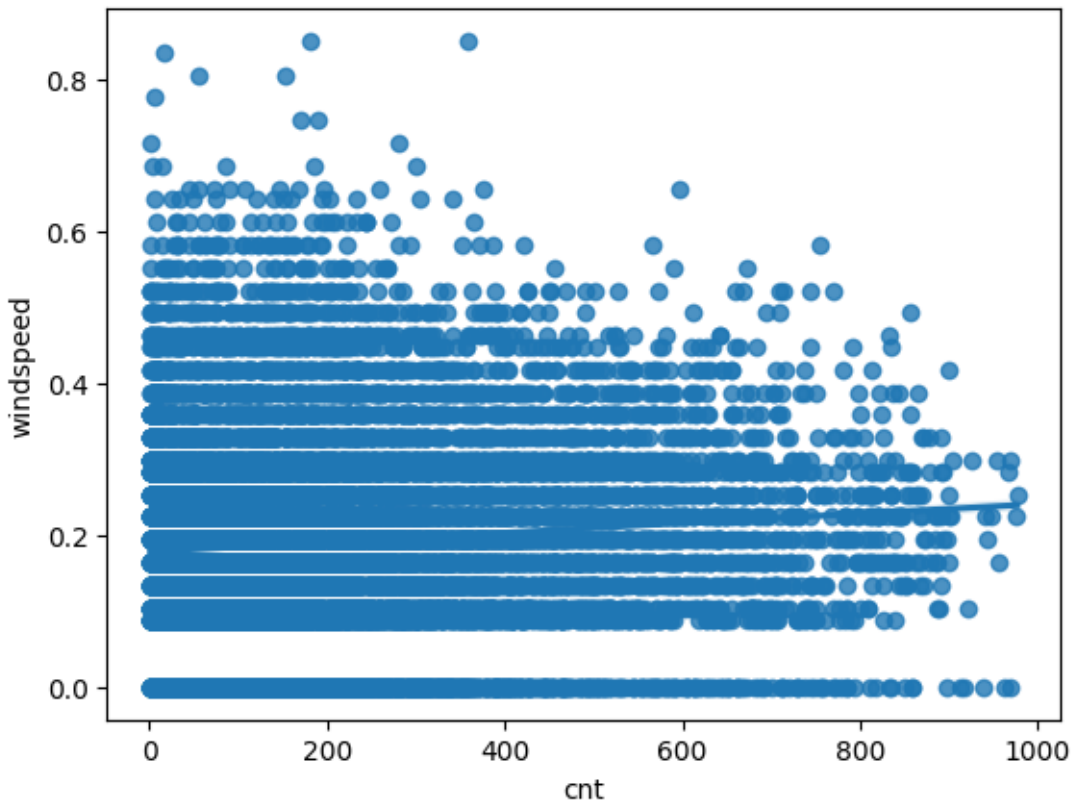
[108]:
```
df.corr(numeric_only=True)["cnt"].drop("cnt").abs().sort_values(ascending=False)
```

[108]:
```
registered    0.972151
casual        0.694564
temp          0.404772
atemp         0.400929
hr            0.394071
hum           0.322911
instant       0.278379
yr            0.250495
season        0.178056
weathersit    0.142426
mnth          0.120638
windspeed     0.093234
holiday       0.030927
workingday    0.030284
weekday       0.026900
Name: cnt, dtype: float64
```

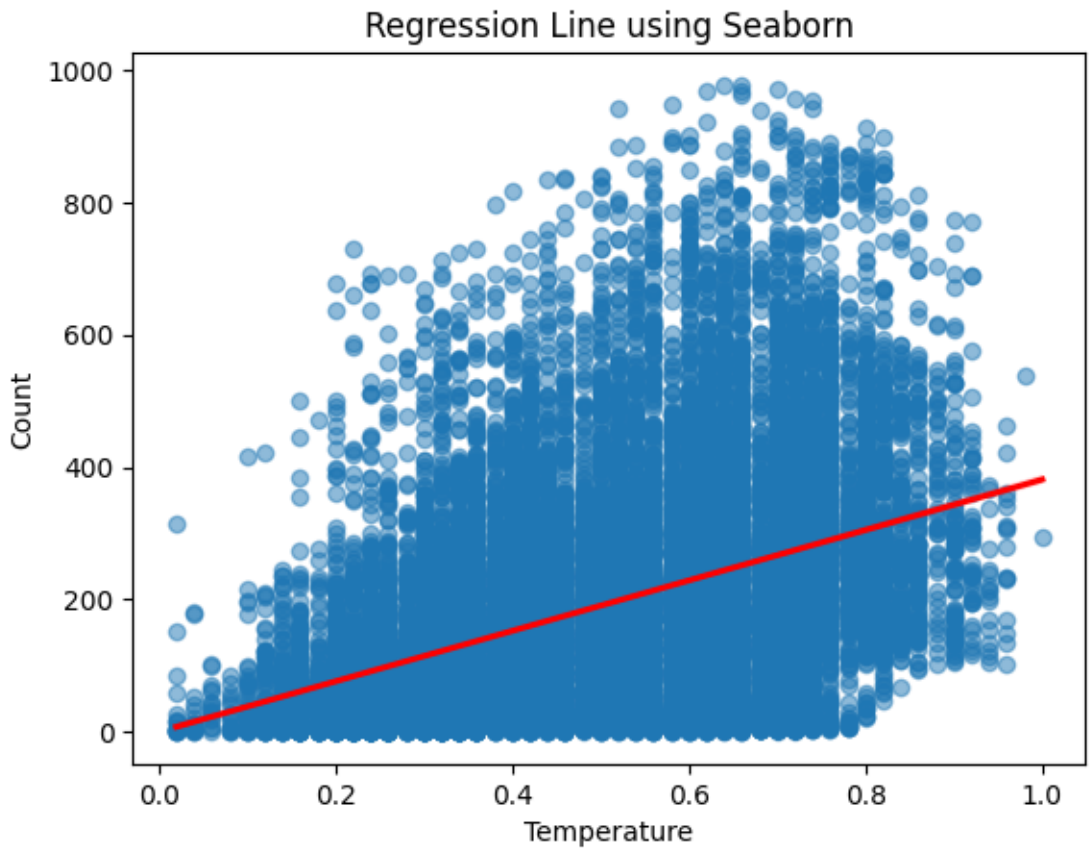
[ ]:
```

```

[104]:
```
#sns.regplot(x=df['temp'], y=df['cnt'])
sns.regplot(x=df['cnt'], y=df['windspeed'])
```

[104]:
```
<Axes: xlabel='cnt', ylabel='windspeed'>
```

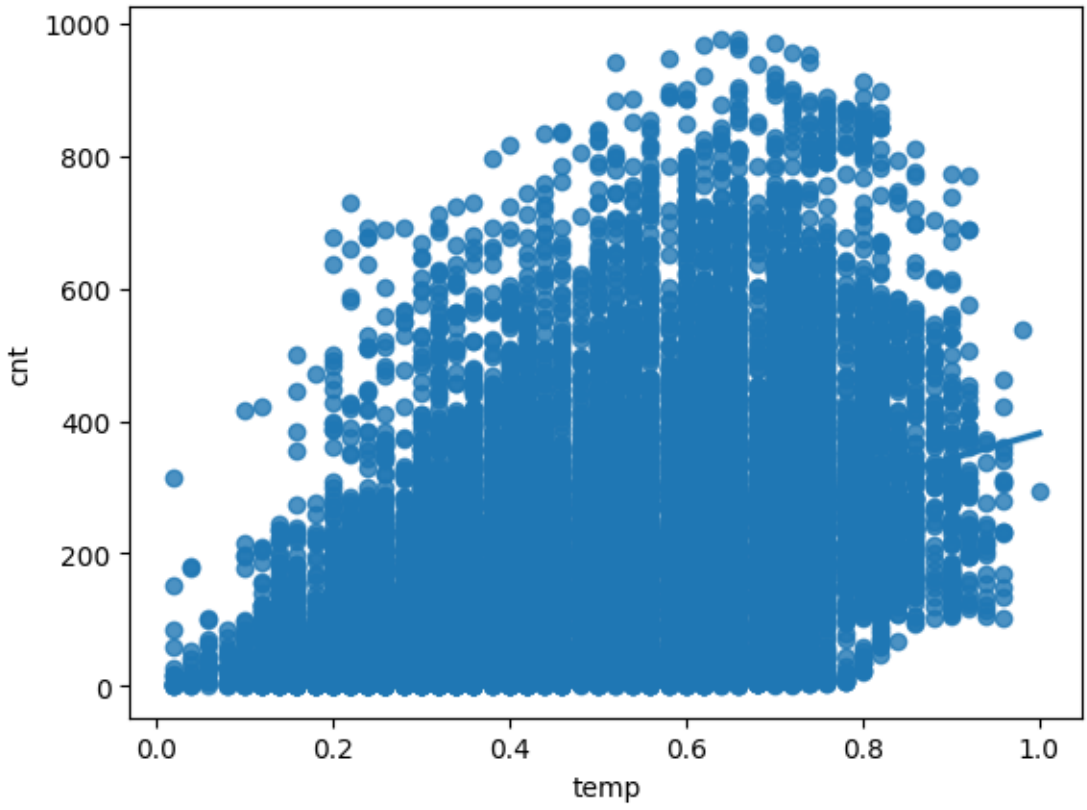


[96]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# This plots the scatter points AND the regression line automatically
sns.regplot(x=df['temp'], y=df['cnt'], scatter_kws={'alpha':0.5},␣
 ↪line_kws={'color':'red'})

plt.xlabel('Temperature')
plt.ylabel('Count')
plt.title('Regression Line using Seaborn')
plt.show()
```



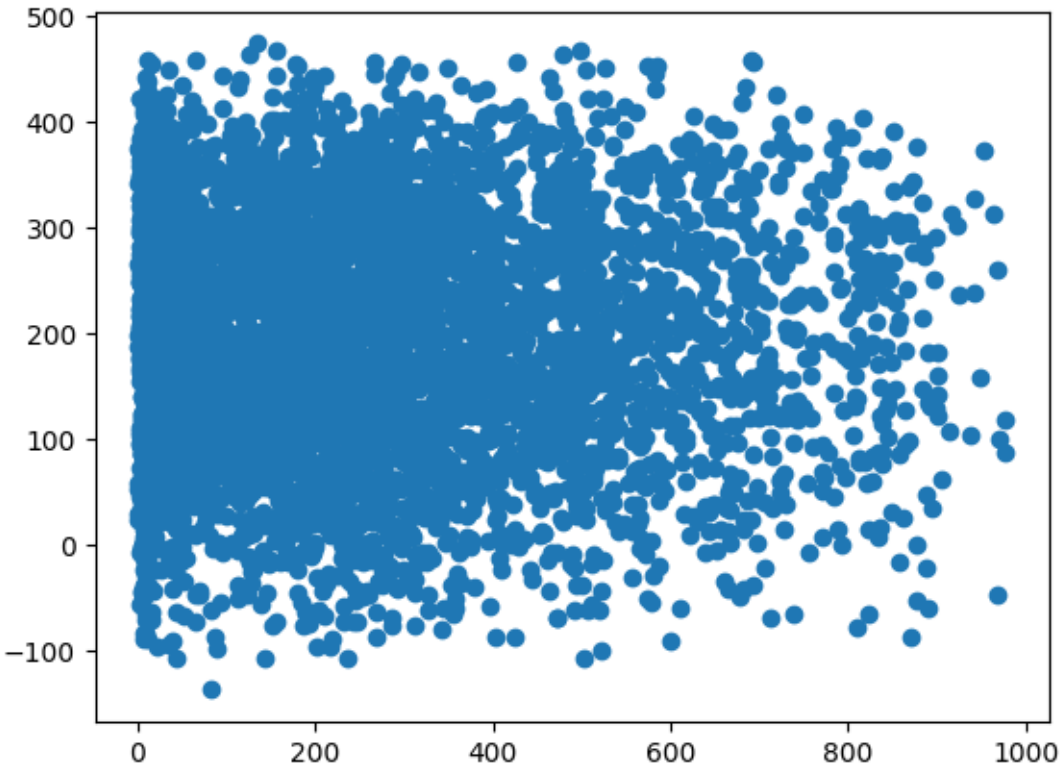
[97]:
```
sns.regplot(x=df['temp'], y=df['cnt'])
```

[97]:
```
<Axes: xlabel='temp', ylabel='cnt'>
```



[105]:
```
plt.scatter(y_test, y_pred)
```

[105]:
```
<matplotlib.collections.PathCollection at 0x1c82b94bb90>
```



[109]:
```
corr_with_target = df.corr(numeric_only=True)["cnt"].drop("cnt")
print(corr_with_target.abs().sort_values(ascending=False))
```

```
registered    0.972151
casual        0.694564
temp          0.404772
atemp         0.400929
hr            0.394071
hum           0.322911
instant       0.278379
yr            0.250495
season        0.178056
weathersit    0.142426
mnth          0.120638
windspeed     0.093234
holiday       0.030927
workingday    0.030284
weekday       0.026900
Name: cnt, dtype: float64
```

[110]:
```
from sklearn.linear_model import LinearRegression
import pandas as pd
```

```
X = df.drop(columns=["cnt", "casual", "registered", "instant", "dteday"])
y = df["cnt"]

model = LinearRegression()
model.fit(X, y)

coef_df = pd.DataFrame({
    "feature": X.columns,
    "coefficient": model.coef_
})

print(coef_df.sort_values("coefficient", key=lambda s: s.abs(),␣
 ↪ascending=False))
```

```
       feature  coefficient
9        atemp   233.157087
10         hum  -198.184681
1           yr    81.087156
8         temp    78.149780
11   windspeed    41.565215
4      holiday   -21.879216
0       season    19.899338
3           hr     7.670597
6   workingday     3.939225
7   weathersit    -3.432098
5      weekday     1.878354
2         mnth    -0.008648
```

[111]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
import pandas as pd

X = df.drop(columns=["cnt", "casual", "registered", "instant", "dteday"])
y = df["cnt"]

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

model = LinearRegression()
model.fit(X_scaled, y)

coef_df = pd.DataFrame({
    "feature": X.columns,
    "coefficient": model.coef_
})
```

```
print(coef_df.reindex(coef_df["coefficient"].abs().sort_values(ascending=False).
  ↪index))
```

```
       feature  coefficient
3           hr    53.036086
1           yr    40.543046
9        atemp    40.066943
10         hum   -38.234638
0       season    22.026304
8         temp    15.047786
11   windspeed     5.084952
5      weekday     3.767441
4      holiday    -3.657340
7   weathersit    -2.194272
6   workingday     1.833383
2         mnth    -0.029738
```

[ ]: