Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

Summary

- **Main Drivers:** Humidity and temperature are the most important factors influencing bike rentals.
- **Statistical Significance:** Both season and weather have a significant effect on demand (confirmed by ANOVA and Chi-square tests).
- **Predictive Modeling:** A Random Forest model explains ~31% of the variance in demand, with top features confirmed by SHAP values and partial dependence plots.
- **Actionable Insights:** Demand is highest in moderate weather and peak seasons; business strategy should focus on inventory and marketing accordingly.

Table of Contents

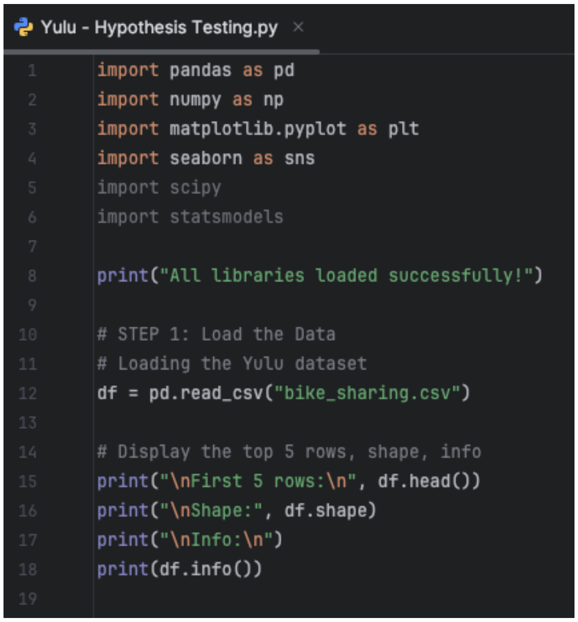
Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025

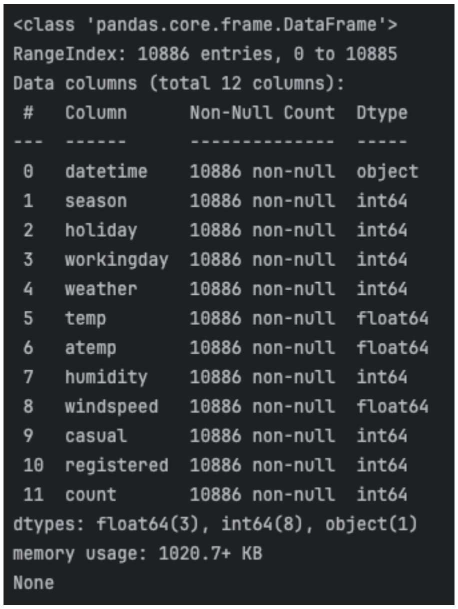
Introduction

This project explores and models the Yulu bike sharing dataset, applying a full data science workflow: from data cleaning and visualization to advanced machine learning and interpretability.

Step 1: Data Loading & Initial Exploration



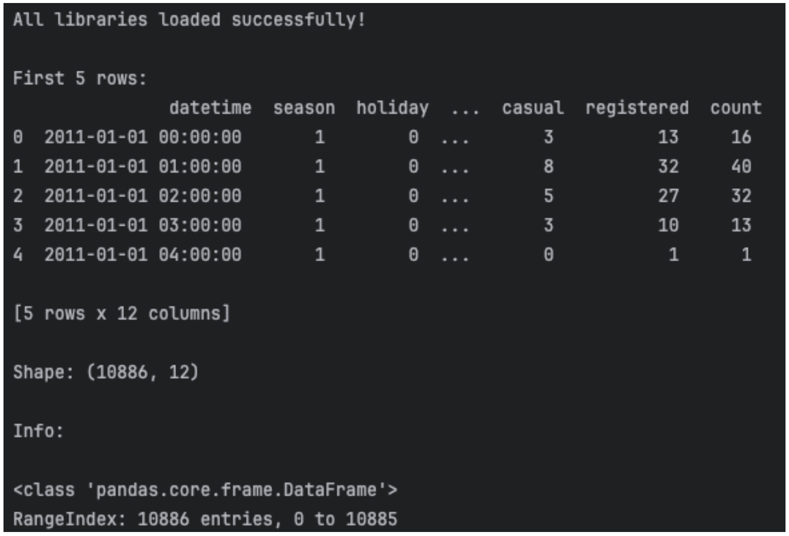
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import scipy
import statsmodels

print("All libraries loaded successfully!")

# STEP 1: Load the Data
# Loading the Yulu dataset
df = pd.read_csv("bike_sharing.csv")

# Display the top 5 rows, shape, info
print("\nFirst 5 rows:\n", df.head())
print("\nShape:", df.shape)
print("\nInfo:\n")
print(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
None
```

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025



```
All libraries loaded successfully!

First 5 rows:
              datetime  season  holiday  ...  casual  registered  count
0  2011-01-01 00:00:00       1        0  ...       3          13     16
1  2011-01-01 01:00:00       1        0  ...       8          32     40
2  2011-01-01 02:00:00       1        0  ...       5          27     32
3  2011-01-01 03:00:00       1        0  ...       3          10     13
4  2011-01-01 04:00:00       1        0  ...       0           1      1

[5 rows x 12 columns]

Shape: (10886, 12)

Info:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
```

What was done:

We loaded the Yulu bike sharing data using Pandas, and displayed the first five rows, the shape of the data, and its structure (data types and non-null counts).

Why was it done:

Initial exploration is critical to understand what features are available, their types (numeric or categorical), and if there are any obvious issues with the dataset before diving into deeper analysis.

What did we find:

- Dataset contains 10,886 records and 12 columns.
- Key columns include date/time, weather, season, workingday, temp, humidity, and the count of rentals.
- Many columns are integers or floats, but several (season, holiday, working day, weather) are categorical by nature, even if they are represented as integers.

Yulu Bike Sharing Data Analysis & Predictive Modeling

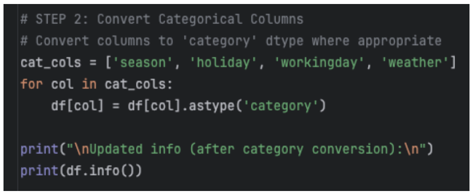
Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

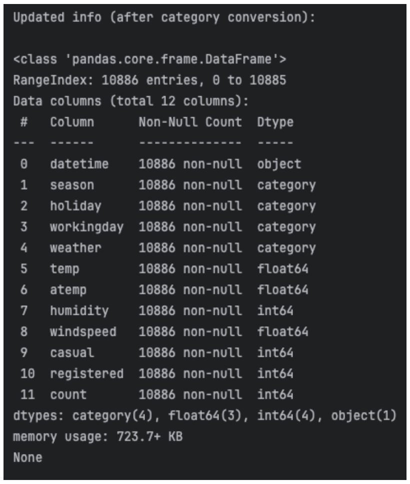
Step 2: Data Cleaning & Categorical Conversion

Summary:

Converted season, holiday, workingday, weather to categorical dtype for better analysis and memory use.



```python
# STEP 2: Convert Categorical Columns
# Convert columns to 'category' dtype where appropriate
cat_cols = ['season', 'holiday', 'workingday', 'weather']
for col in cat_cols:
    df[col] = df[col].astype('category')

print("\nUpdated info (after category conversion):\n")
print(df.info())
```



```
Updated info (after category conversion):

<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  category
 2   holiday     10886 non-null  category
 3   workingday  10886 non-null  category
 4   weather     10886 non-null  category
 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: category(4), float64(3), int64(4), object(1)
memory usage: 723.7+ KB
None
```

What was done:

Converted columns ‘season’, ‘holiday’, ‘working day’, and ‘weather’ to the ‘category’ dtype.

Why was it done:

Using the category data type:

- Reduces memory usage
- Ensures correct analysis and visualization of categorical features (such as season, weather, etc.)
- Enables proper statistical tests for categorical vs numeric variables

What did we find:

After conversion, the data types reflect that these variables are categorical, which will help with grouped analysis and plotting.

Yulu Bike Sharing Data Analysis & Predictive Modeling

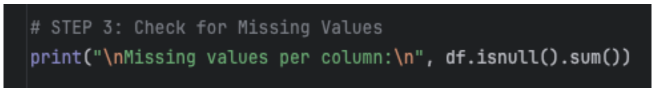
Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

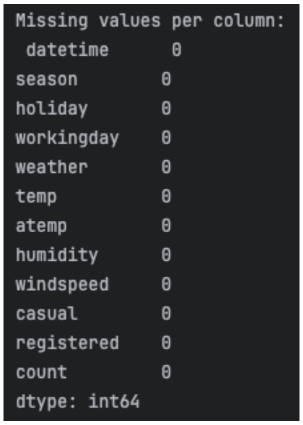
Step 3: Missing Value Check

Result:

No missing values found.



```
# STEP 3: Check for Missing Values
print("\nMissing values per column:\n", df.isnull().sum())
```



```
Missing values per column:
 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
```

What was done:

Checked each column for missing values using .isnull().sum().

Why was it done:

Missing data can cause errors in later analysis, statistical testing, or modeling. It’s best practice to confirm data completeness.

What did we find:

No missing values were detected. The dataset is complete and ready for further analysis.

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025

Step 4: Statistical Summary

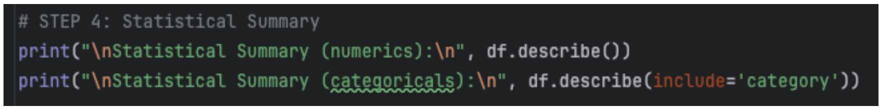
Numeric summary:

count (target): min=1, max=977, mean≈192, std≈181

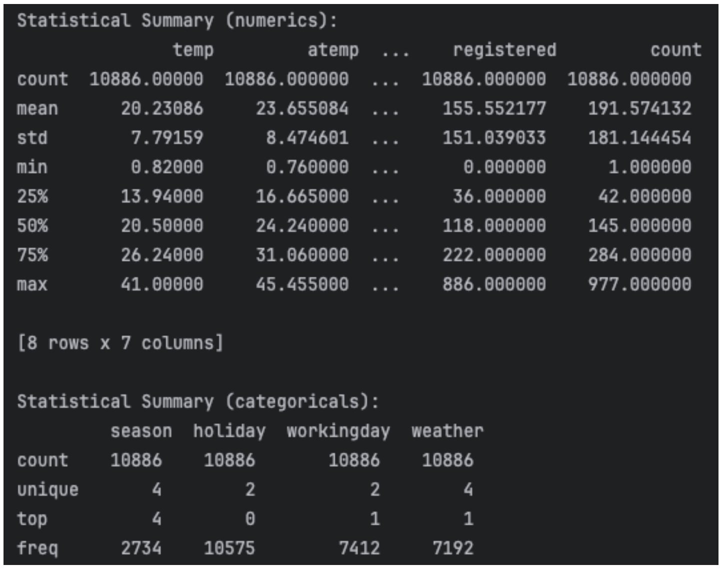
Features: temp, atemp, humidity, windspeed, casual, registered (see code for quartiles)

Categorical summary:

4 seasons, 2 holiday values, 2 workingday, 4 weather categories.



```
# STEP 4: Statistical Summary
print("\nStatistical Summary (numerics):\n", df.describe())
print("\nStatistical Summary (categoricals):\n", df.describe(include='category'))
```



```
Statistical Summary (numerics):
              temp         atemp  ...    registered         count
count  10886.00000  10886.000000  ...  10886.000000  10886.000000
mean      20.23086     23.655084  ...    155.552177    191.574132
std        7.79159      8.474601  ...    151.039033    181.144454
min        0.82000      0.760000  ...      0.000000      1.000000
25%       13.94000     16.665000  ...     36.000000     42.000000
50%       20.50000     24.240000  ...    118.000000    145.000000
75%       26.24000     31.060000  ...    222.000000    284.000000
max       41.00000     45.455000  ...    886.000000    977.000000

[8 rows x 7 columns]

Statistical Summary (categoricals):
        season  holiday  workingday  weather
count    10886    10886       10886    10886
unique       4        2           2        4
top          4        0           1        1
freq      2734    10575        7412     7192
```

What was done:

Generated summary statistics using .describe() for both numeric and categorical variables.

Why was it done:

Provides a snapshot of distributions, ranges, central tendencies, and helps spot unusual patterns (like potential outliers or skewness).

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

What did we find:

- Numeric features (e.g., ‘count’, ‘temp’, ‘humidity’) have wide ranges and standard deviations.
- Target variable ‘count’ ranges from 1 to 977, mean ≈ 192, std ≈ 181.
- The data is moderately to heavily right-skewed (more low-demand periods than high).
- Categorical features distribute as expected (e.g., 4 seasons, binary holiday/working day).

Step 5: Univariate Analysis - Continuous

Graphs:

Distribution histograms for each numeric column.

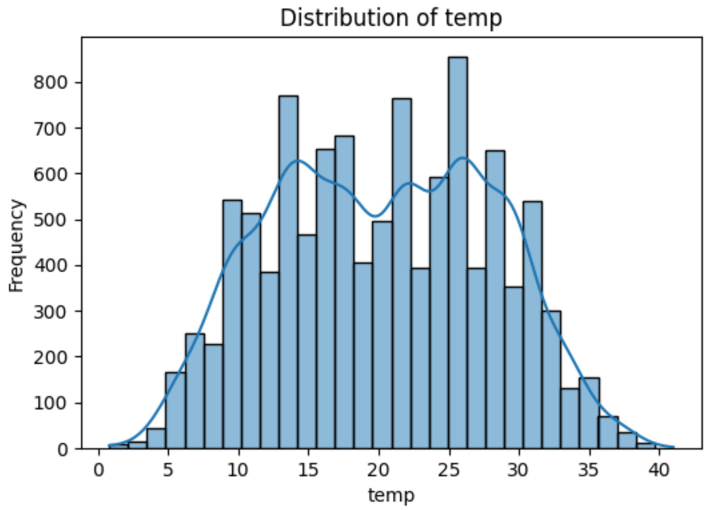
Include your plotted images, e.g.:

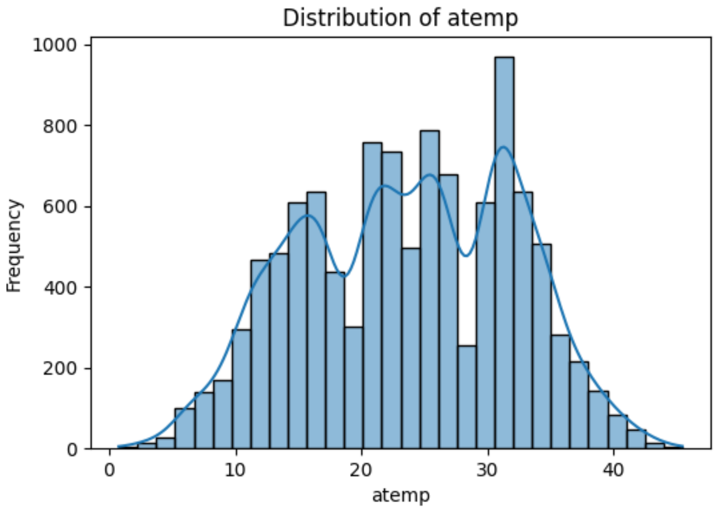
Comment:

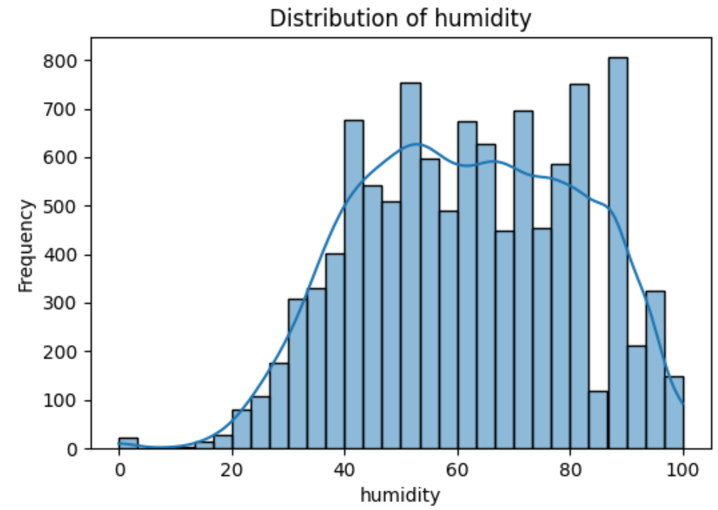
temp/atemp: roughly normal

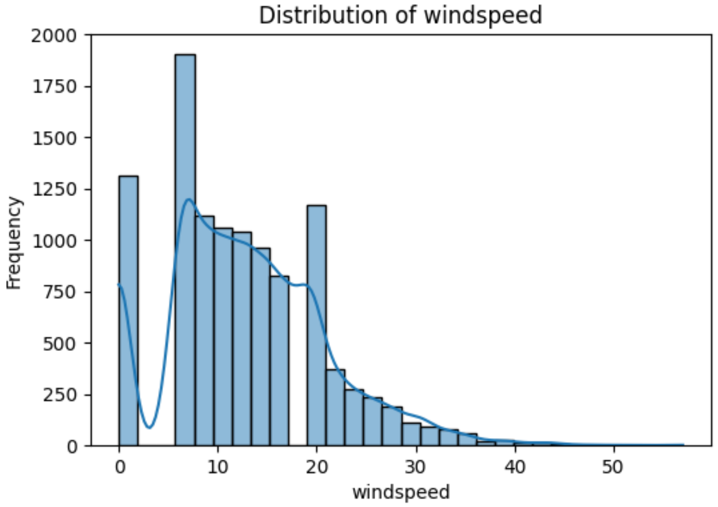
count, casual, registered: right-skewed (long tail)

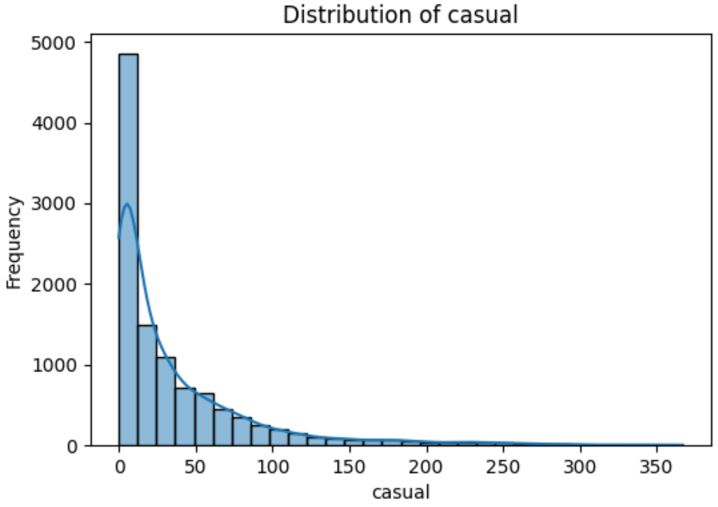
humidity/windspeed: variable distributions

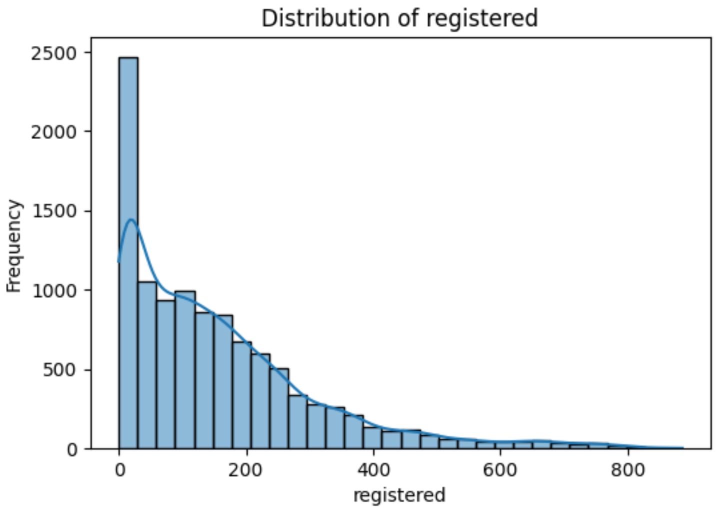


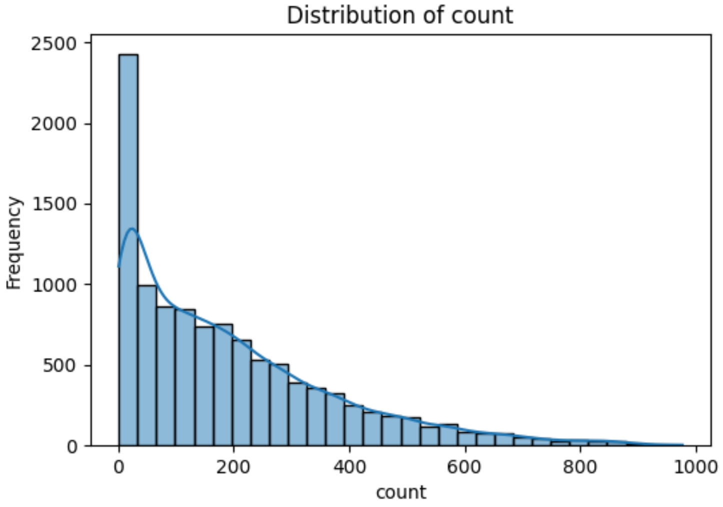



Distribution of humidity
Frequency
0
100
200
300
400
500
600
700
800
0
20
40
60
80
100
humidity


Distribution of windspeed
Frequency
0
250
500
750
1000
1250
1500
1750
2000
0
10
20
30
40
50
windspeed


Distribution of casual
Frequency
0
1000
2000
3000
4000
5000
0
50
100
150
200
250
300
350
casual





What was done:

Plotted histograms (with KDE curves) for each numeric variable: temp, atemp, humidity, windspeed, casual, registered, count.

Why was it done:

To visualize distributions, check for skewness, modality, and outliers. This informs transformations or modeling decisions.

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

What did we find:

- temp/atemp: Mostly normal distributions.
- humidity: Slight left-skew; some high values.
- windspeed: Widespread, but no severe outliers.
- count/casual/registered: Strong right-skew; many low-rental hours, a few very high-demand periods.

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

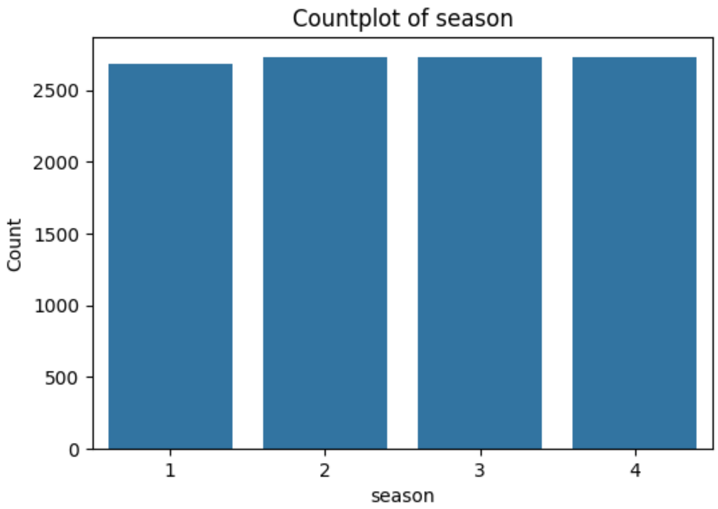
Step 6: Univariate Analysis - Categorical

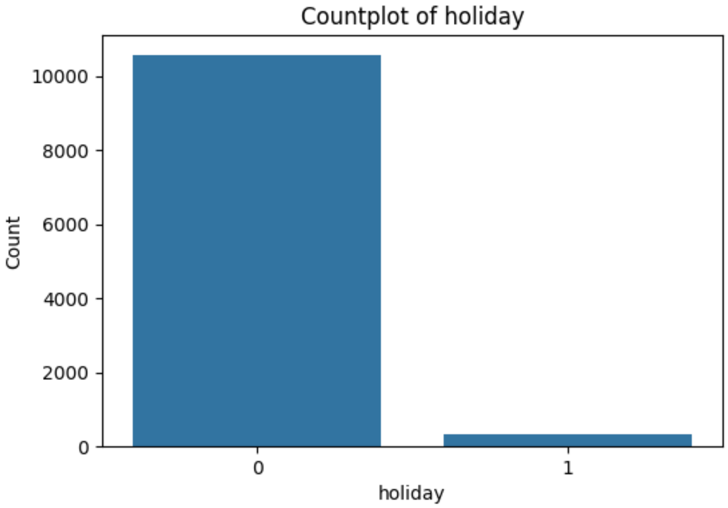
Graphs:

Bar plots for season, holiday, workingday, weather.

Comment:

Season and weather categories are balanced; holidays are rare.

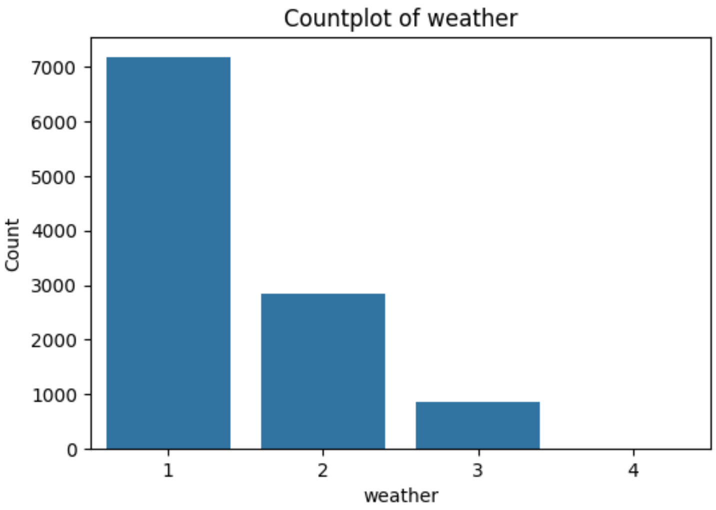


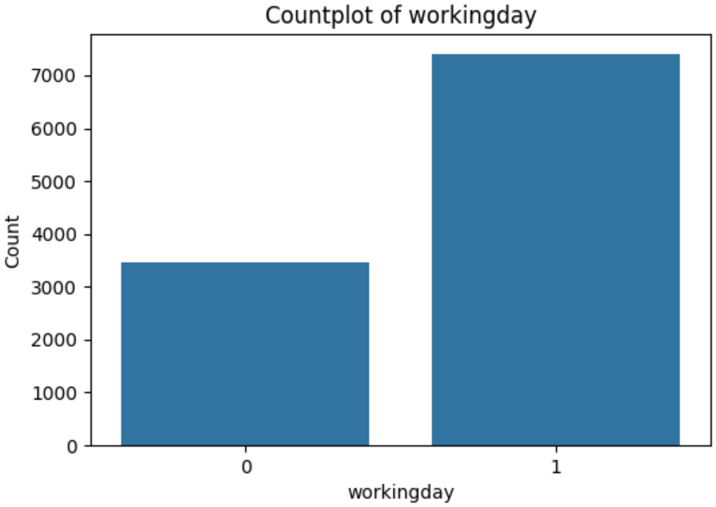


Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025





What was done:

Used countplots (bar charts) to visualize the frequency of each category in season, holiday, workingday, and weather.

Why was it done:

To ensure no category is grossly over- or under-represented, which could bias statistical analysis or model training.

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

What did we find:

- All seasons and weather categories are represented.
- The majority of days are not holidays.
- Working days are more common than non-working days.

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025

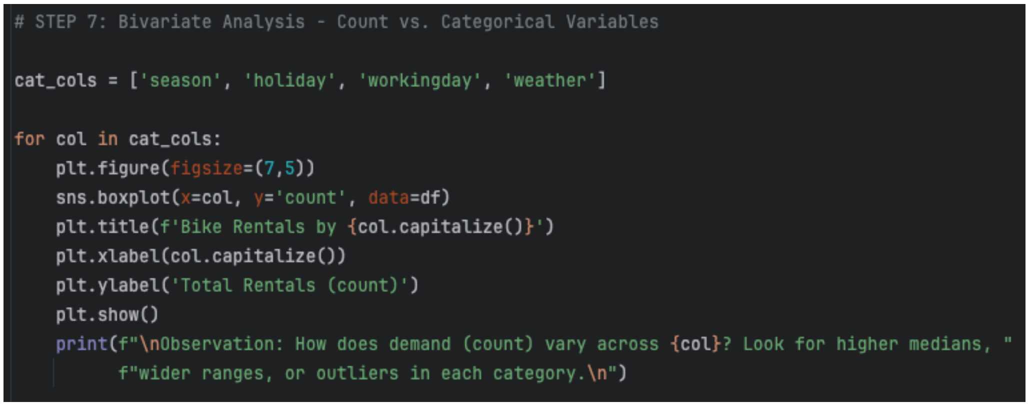
Step 7: Bivariate Analysis

Boxplots:

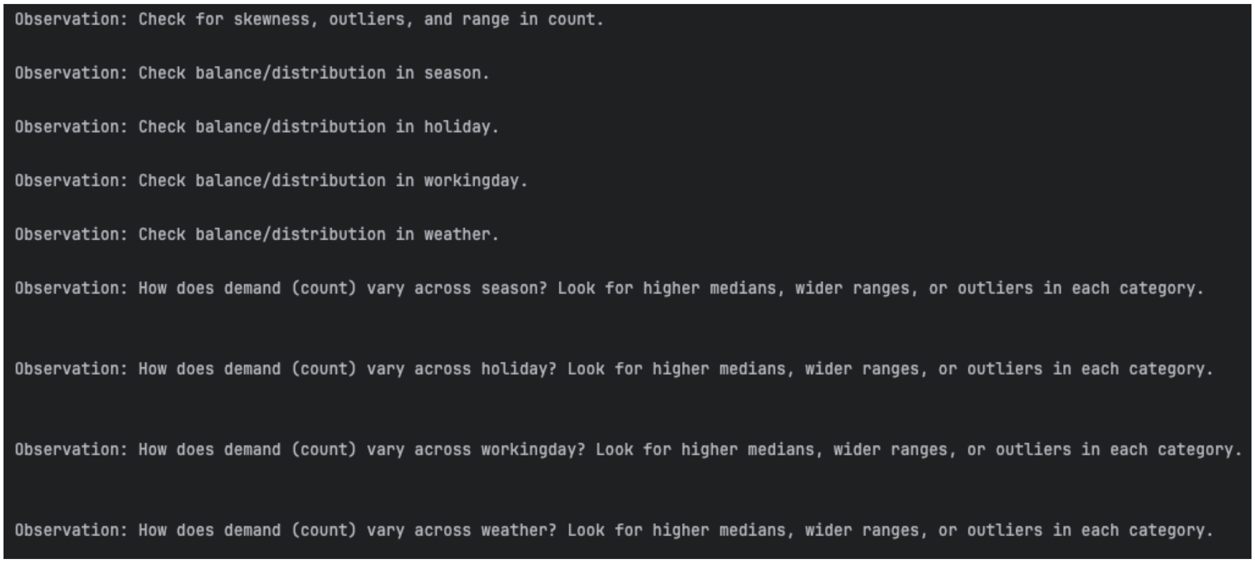
Demand (count) by season, holiday, workingday, weather.

Comment:

Higher demand in some seasons/weather, visible outliers.



```python
# STEP 7: Bivariate Analysis - Count vs. Categorical Variables

cat_cols = ['season', 'holiday', 'workingday', 'weather']

for col in cat_cols:
    plt.figure(figsize=(7,5))
    sns.boxplot(x=col, y='count', data=df)
    plt.title(f'Bike Rentals by {col.capitalize()}')
    plt.xlabel(col.capitalize())
    plt.ylabel('Total Rentals (count)')
    plt.show()
    print(f"\nObservation: How does demand (count) vary across {col}? Look for higher medians, "
          f"wider ranges, or outliers in each category.\n")
```

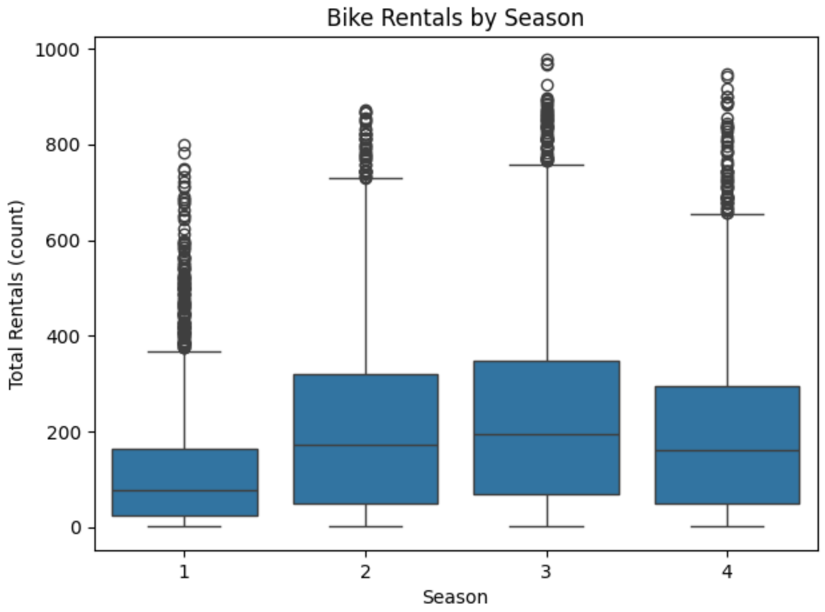


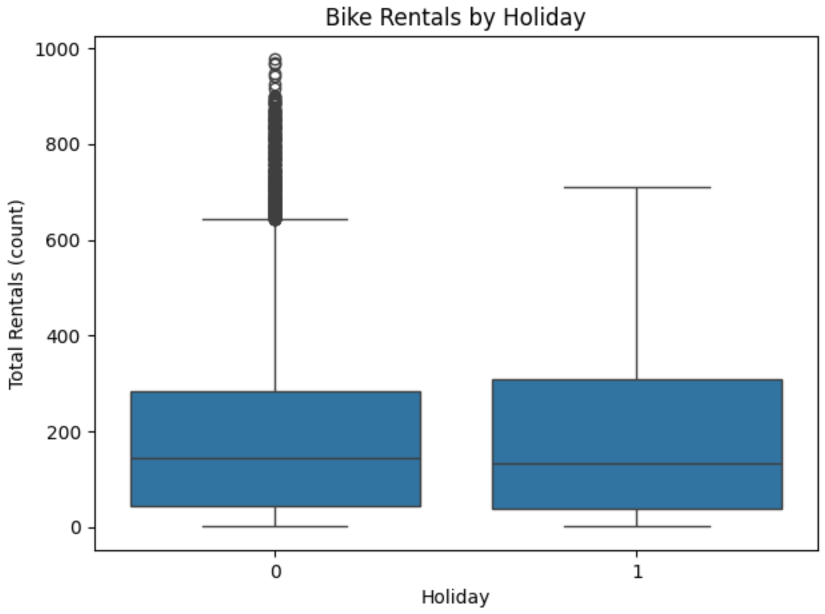
```
Observation: Check for skewness, outliers, and range in count.

Observation: Check balance/distribution in season.

Observation: Check balance/distribution in holiday.

Observation: Check balance/distribution in workingday.

Observation: Check balance/distribution in weather.

Observation: How does demand (count) vary across season? Look for higher medians, wider ranges, or outliers in each category.


Observation: How does demand (count) vary across holiday? Look for higher medians, wider ranges, or outliers in each category.


Observation: How does demand (count) vary across workingday? Look for higher medians, wider ranges, or outliers in each category.


Observation: How does demand (count) vary across weather? Look for higher medians, wider ranges, or outliers in each category.
```

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

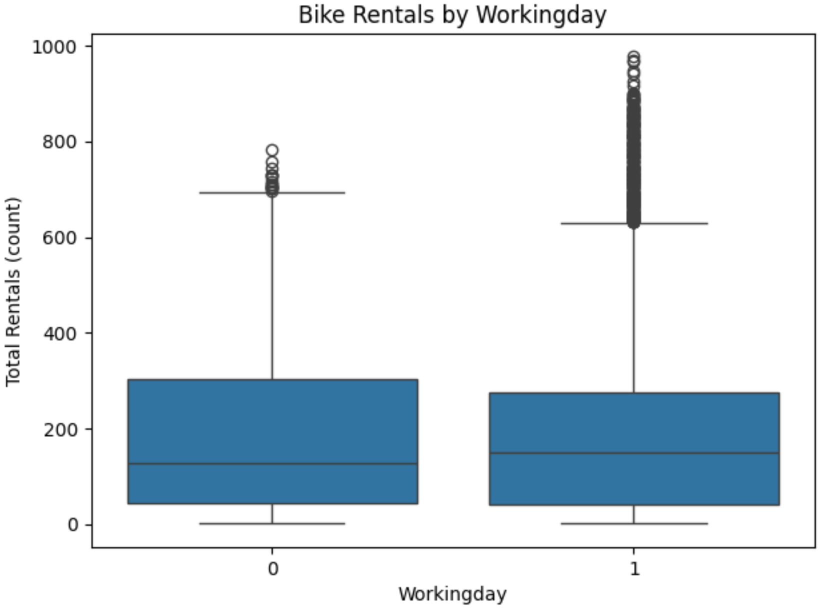
Date: July 2025

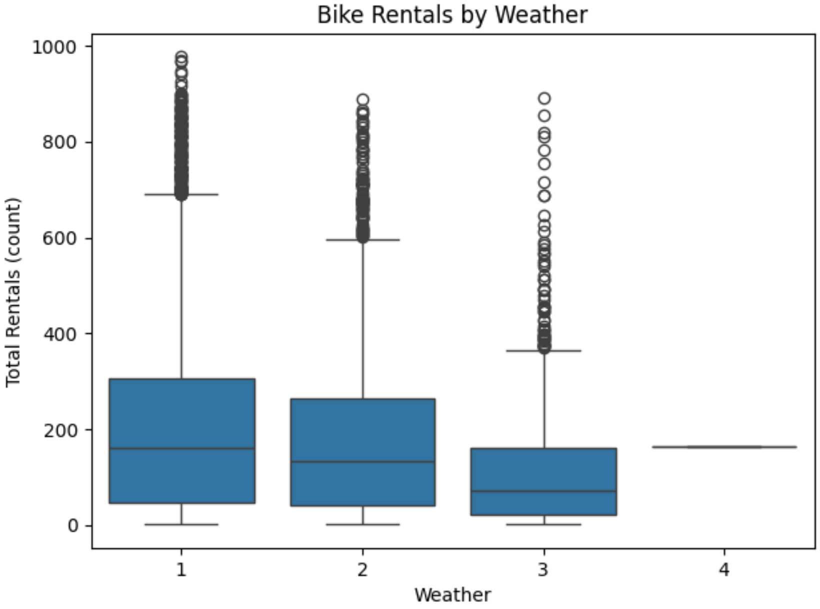




Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025





What was done:

Boxplots showing rental demand (count) across each level of season, holiday, workingday, and weather.

Why was it done:

Boxplots reveal:

- The distribution of the target variable (count) within each group
- Median, spread, and outliers for each category
- Initial hints at which factors affect demand most

What did we find:

- Season: Seasons 3 and 2 have higher medians and broader spreads; season 1 is lowest.
- Weather: Clear weather (type 1) has highest demand; poor weather sharply reduces rentals.
- Holiday/working day: Small difference, but some outliers on holidays (special events).

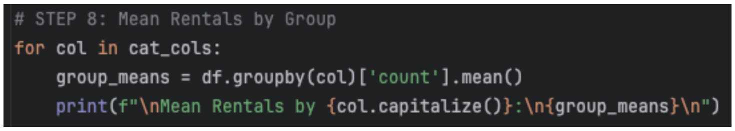
Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

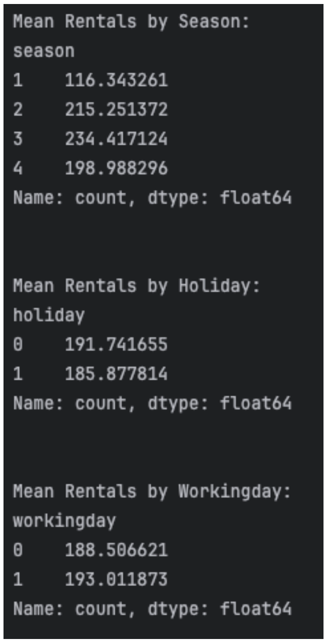
Date: July 2025

Step 8: Mean Rentals by Group

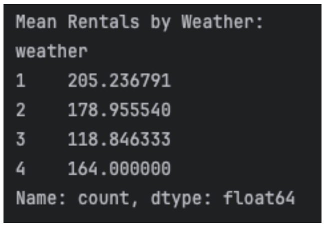
Category	Mean Rentals
Season 1	116
Season 2	215
Season 3	234
Season 4	199



```python
# STEP 8: Mean Rentals by Group
for col in cat_cols:
    group_means = df.groupby(col)['count'].mean()
    print(f"\nMean Rentals by {col.capitalize()}:\n{group_means}\n")
```



```
Mean Rentals by Season:
season
1    116.343261
2    215.251372
3    234.417124
4    198.988296
Name: count, dtype: float64


Mean Rentals by Holiday:
holiday
0    191.741655
1    185.877814
Name: count, dtype: float64


Mean Rentals by Workingday:
workingday
0    188.506621
1    193.011873
Name: count, dtype: float64
```



```
Mean Rentals by Weather:
weather
1    205.236791
2    178.955540
3    118.846333
4    164.000000
Name: count, dtype: float64
```

What was done:

Calculated and printed the average rental demand (count) for each season, holiday, working day, and weather group.

Why was it done:

Numerical summary of group differences supplements boxplots and supports hypothesis testing.

What did we find:

- Season 3 (fall) has the highest mean rentals, followed by season 2 (summer).
- Rentals are slightly higher on working days.
- Clear weather days have the highest mean demand; demand drops in bad weather.

Yulu Bike Sharing Data Analysis & Predictive Modeling

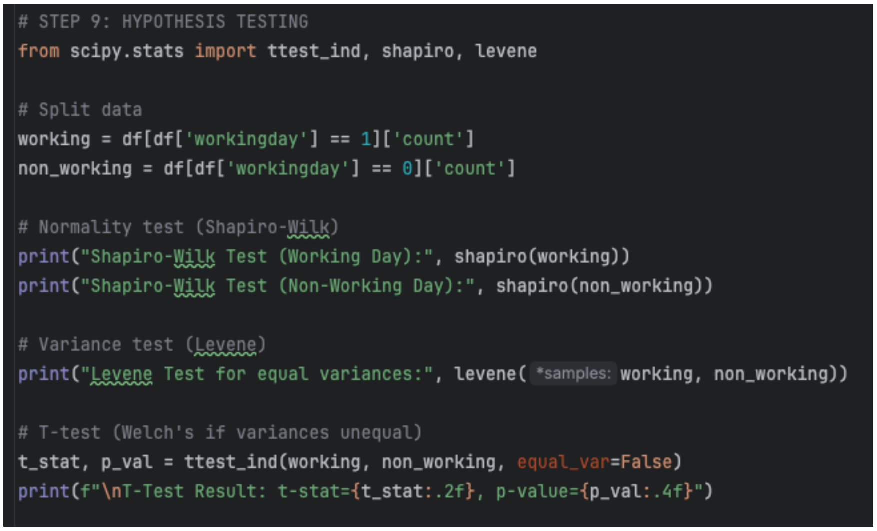
Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

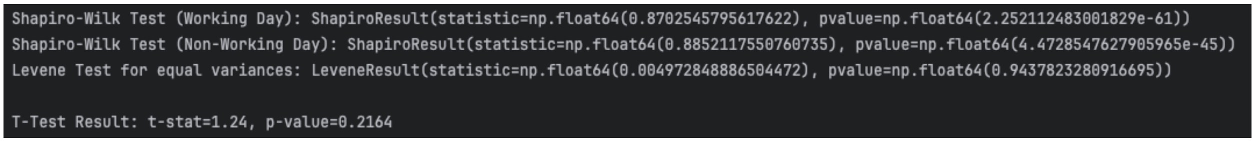
Step 9: Hypothesis Testing (T-test)

T-test: Working day vs. non-working day

- t-stat=1.24, p-value=0.2164
- Interpretation: No significant difference in mean rentals between working and non-working days.



```
# STEP 9: HYPOTHESIS TESTING
from scipy.stats import ttest_ind, shapiro, levene

# Split data
working = df[df['workingday'] == 1]['count']
non_working = df[df['workingday'] == 0]['count']

# Normality test (Shapiro-Wilk)
print("Shapiro-Wilk Test (Working Day):", shapiro(working))
print("Shapiro-Wilk Test (Non-Working Day):", shapiro(non_working))

# Variance test (Levene)
print("Levene Test for equal variances:", levene( *samples: working, non_working))

# T-test (Welch's if variances unequal)
t_stat, p_val = ttest_ind(working, non_working, equal_var=False)
print(f"\nT-Test Result: t-stat={t_stat:.2f}, p-value={p_val:.4f}")
```



```
Shapiro-Wilk Test (Working Day): ShapiroResult(statistic=np.float64(0.8702545795617622), pvalue=np.float64(2.252112483001829e-61))
Shapiro-Wilk Test (Non-Working Day): ShapiroResult(statistic=np.float64(0.8852117550760735), pvalue=np.float64(4.4728547627905965e-45))
Levene Test for equal variances: LeveneResult(statistic=np.float64(0.004972848886504472), pvalue=np.float64(0.9437823280916695))

T-Test Result: t-stat=1.24, p-value=0.2164
```

What was done:

- Checked normality (Shapiro-Wilk) and variance (Levene test)
- Used independent t-test to compare mean rentals between working days and non-working days.

Why was it done:

To formally test if the average demand is statistically different on working vs non-working days (beyond visual observation).

What did we find:

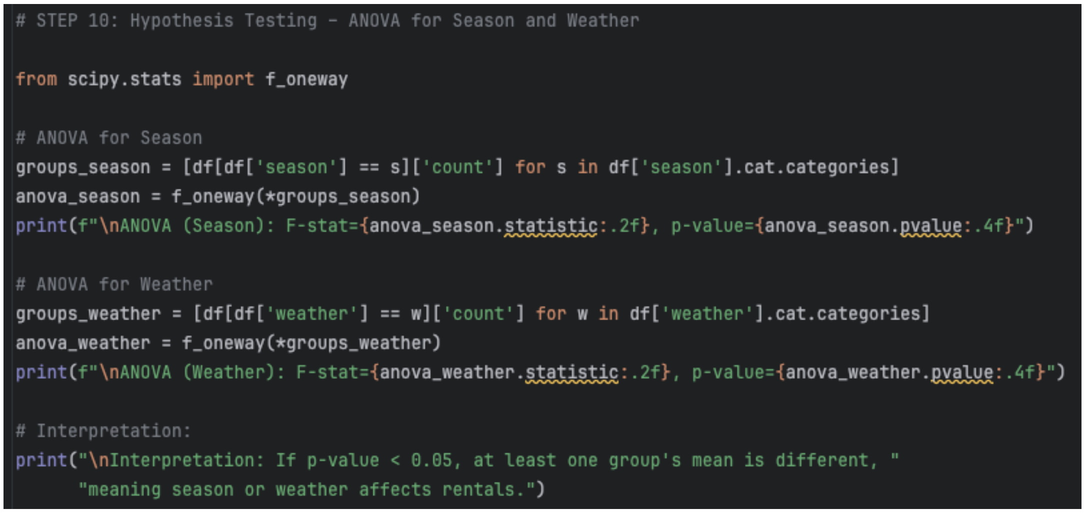
- Both distributions are not normal (p-value << 0.05, as expected for large datasets).
- Variance is equal (Levene test p=0.94).
- T-test: p-value = 0.216 → No statistically significant difference in average rentals between working and non-working days.

Step 10: Hypothesis Testing – ANOVA (Season/Weather)

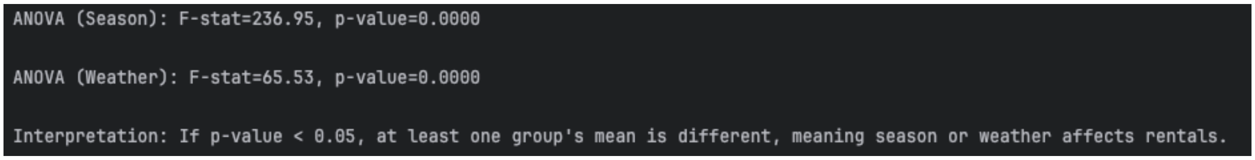
- Season: F=236.95, p<0.0001
- Weather: F=65.53, p<0.0001

Interpretation:

Season and weather significantly affect rental counts.



```python
# STEP 10: Hypothesis Testing - ANOVA for Season and Weather

from scipy.stats import f_oneway

# ANOVA for Season
groups_season = [df[df['season'] == s]['count'] for s in df['season'].cat.categories]
anova_season = f_oneway(*groups_season)
print(f"\nANOVA (Season): F-stat={anova_season.statistic:.2f}, p-value={anova_season.pvalue:.4f}")

# ANOVA for Weather
groups_weather = [df[df['weather'] == w]['count'] for w in df['weather'].cat.categories]
anova_weather = f_oneway(*groups_weather)
print(f"\nANOVA (Weather): F-stat={anova_weather.statistic:.2f}, p-value={anova_weather.pvalue:.4f}")

# Interpretation:
print("\nInterpretation: If p-value < 0.05, at least one group's mean is different, "
      "meaning season or weather affects rentals.")
```



```
ANOVA (Season): F-stat=236.95, p-value=0.0000

ANOVA (Weather): F-stat=65.53, p-value=0.0000

Interpretation: If p-value < 0.05, at least one group's mean is different, meaning season or weather affects rentals.
```

What was done:

One-way ANOVA to compare average demand across seasons and weather types.

Why was it done:

To check if there is a significant difference in mean rentals between groups with more than two categories (e.g., 4 seasons).

What did we find:

- Both p-values are <0.001.
- Season and weather both significantly affect bike demand. Some seasons and weather types see much higher usage.

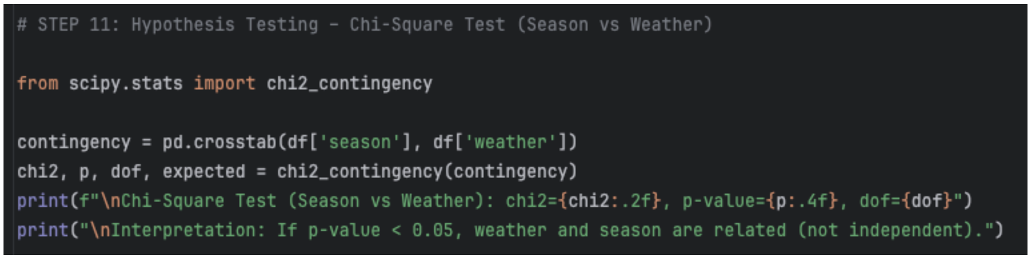
Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

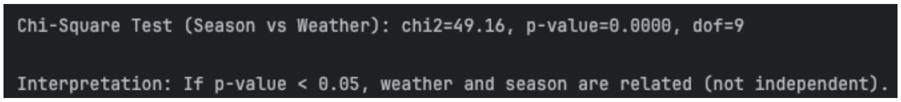
Date: July 2025

Step 11: Hypothesis Testing – Chi-Square Test

- Chi2=49.16, p<0.0001
- Interpretation: Season and weather are statistically related (not independent).



```
# STEP 11: Hypothesis Testing - Chi-Square Test (Season vs Weather)

from scipy.stats import chi2_contingency

contingency = pd.crosstab(df['season'], df['weather'])
chi2, p, dof, expected = chi2_contingency(contingency)
print(f"\nChi-Square Test (Season vs Weather): chi2={chi2:.2f}, p-value={p:.4f}, dof={dof}")
print("\nInterpretation: If p-value < 0.05, weather and season are related (not independent).")
```



```
Chi-Square Test (Season vs Weather): chi2=49.16, p-value=0.0000, dof=9

Interpretation: If p-value < 0.05, weather and season are related (not independent).
```

What was done:

Chi-square test of independence between season and weather.

Why was it done:

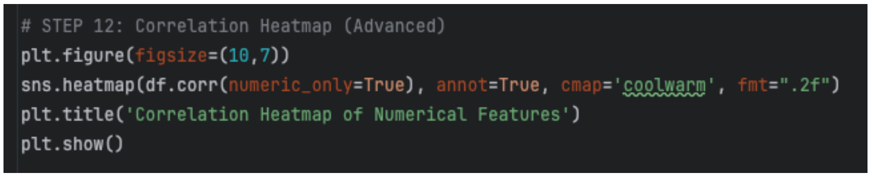
To test if these two predictors are independent or if some weather types are more likely in certain seasons.

What did we find:

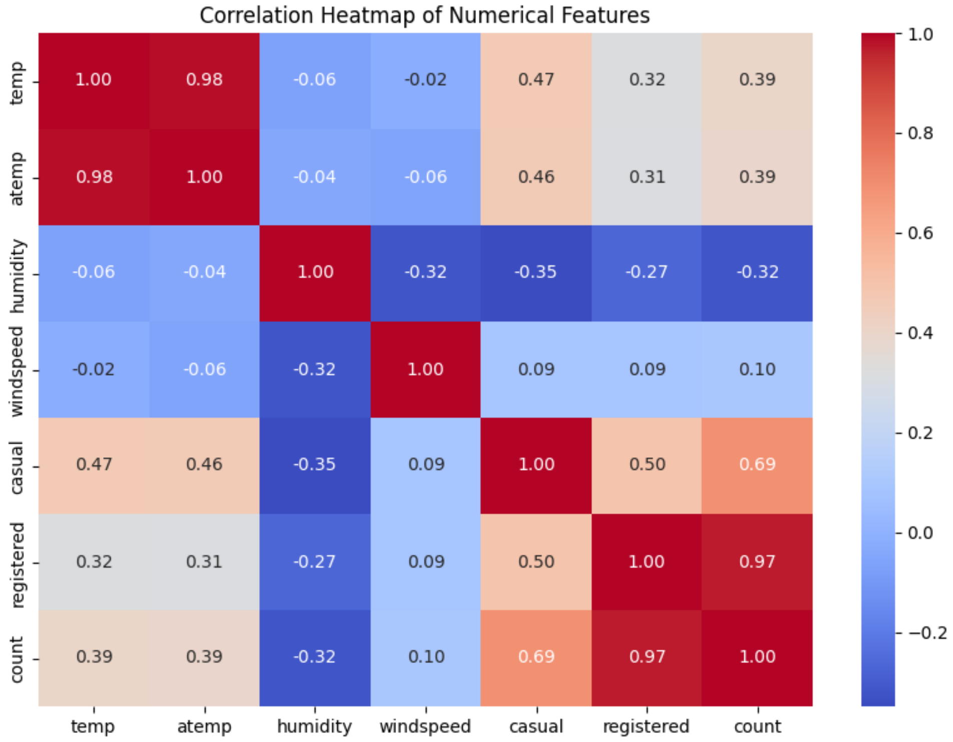
P-value < 0.001: Season and weather are not independent (e.g., certain weather types are much more likely in some seasons).

Step 12: Correlation Heatmap

Count highly correlated with registered, moderately with casual, and weakly with weather variables.



```
# STEP 12: Correlation Heatmap (Advanced)
plt.figure(figsize=(10,7))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap of Numerical Features')
plt.show()
```



	temp	atemp	humidity	windspeed	casual	registered	count
temp	1.00	0.98	-0.06	-0.02	0.47	0.32	0.39
atemp	0.98	1.00	-0.04	-0.06	0.46	0.31	0.39
humidity	-0.06	-0.04	1.00	-0.32	-0.35	-0.27	-0.32
windspeed	-0.02	-0.06	-0.32	1.00	0.09	0.09	0.10
casual	0.47	0.46	-0.35	0.09	1.00	0.50	0.69
registered	0.32	0.31	-0.27	0.09	0.50	1.00	0.97
count	0.39	0.39	-0.32	0.10	0.69	0.97	1.00

What was done:

Heatmap of pairwise Pearson correlations among all numeric variables.

Why was it done:

To see which variables move together (positively or negatively), which helps in feature selection and understanding data structure.

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

What did we find:

- Registered and count are very highly correlated (as expected).
- atemp and temp are highly correlated (they measure similar things).
- Humidity and count are negatively correlated: higher humidity, fewer rentals.

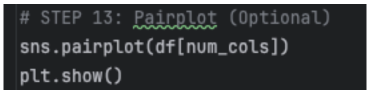
Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

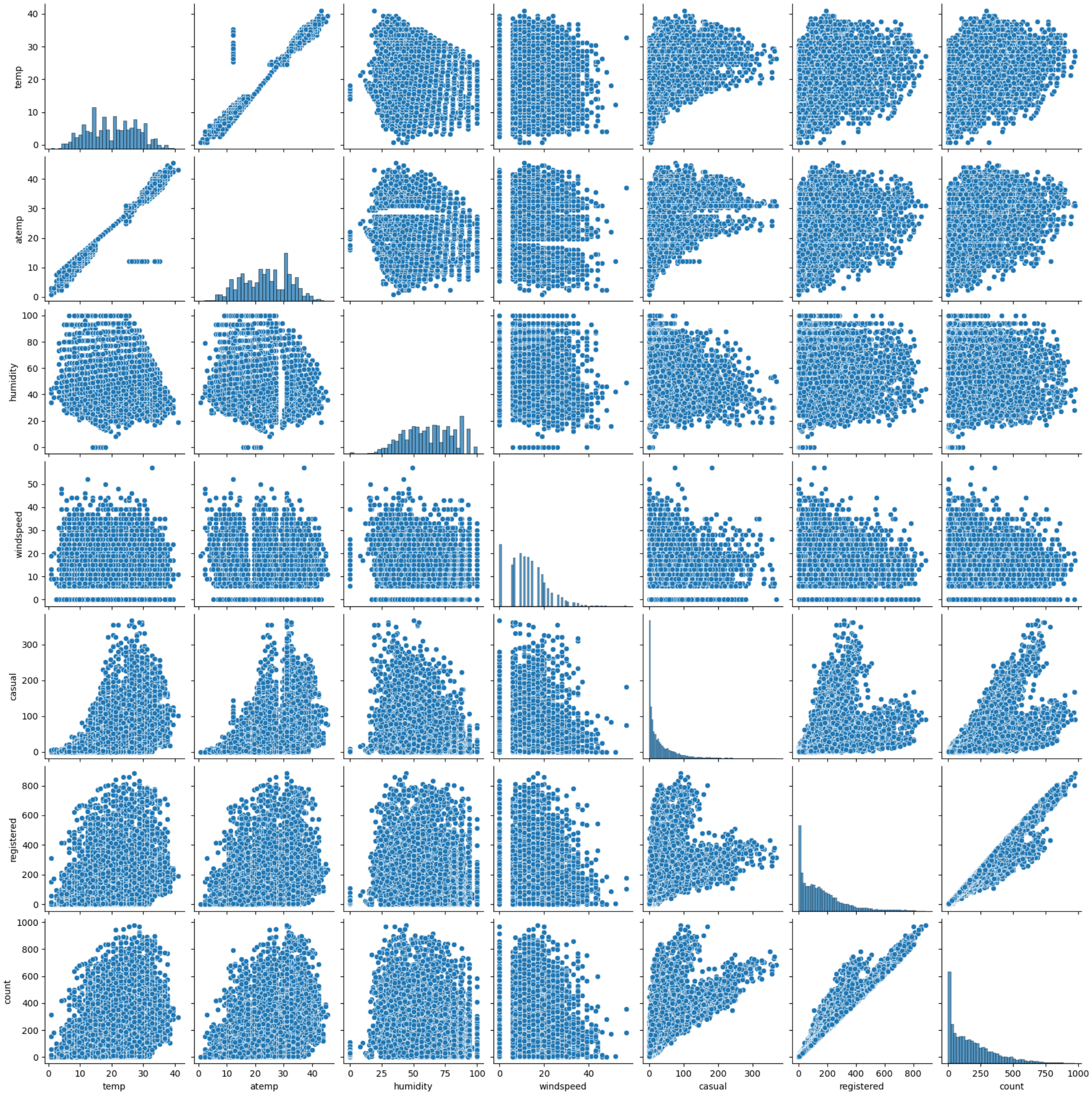
Date: July 2025

Step 13: Pairplot Analysis

Scatterplot matrix confirms trends/correlations (e.g., temp/atemp vs. count).



```
# STEP 13: Pairplot (Optional)
sns.pairplot(df[num_cols])
plt.show()
```



Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025

What was done:

Pairwise scatterplots (pair plot) of all numeric variables.

Why was it done:

To spot non-linear relationships, multi-variable patterns, or clusters.

What did we find:

- Rentals tend to increase with temp/atemp.
- No obvious clusters, but some non-linear trends between variables.

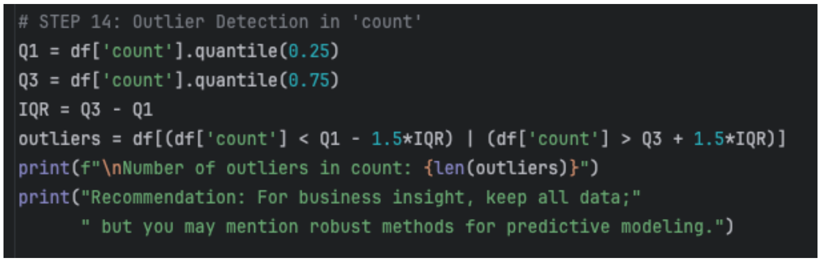
Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

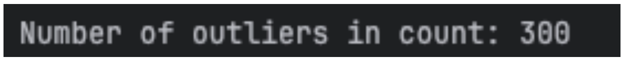
Date: July 2025

Step 14: Outlier Detection

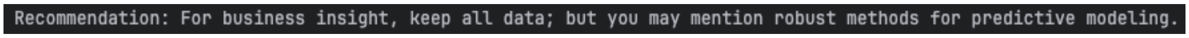
- Outliers in count: 300
- Comment: For business, keep all data. Robust models can handle outliers.



```python
# STEP 14: Outlier Detection in 'count'
Q1 = df['count'].quantile(0.25)
Q3 = df['count'].quantile(0.75)
IQR = Q3 - Q1
outliers = df[(df['count'] < Q1 - 1.5*IQR) | (df['count'] > Q3 + 1.5*IQR)]
print(f"\nNumber of outliers in count: {len(outliers)}")
print("Recommendation: For business insight, keep all data;"
      " but you may mention robust methods for predictive modeling.")
```



```
Number of outliers in count: 300
```



```
Recommendation: For business insight, keep all data; but you may mention robust methods for predictive modeling.
```

What was done:

Used the Interquartile Range (IQR) method to detect outliers in the ‘count’ variable.

Why was it done:

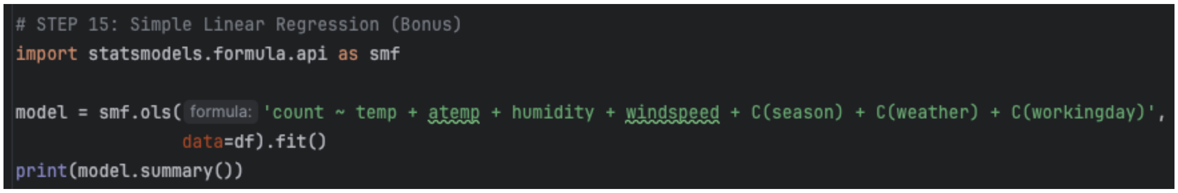
Outliers can skew averages, affect hypothesis tests, and degrade model performance. Identifying them helps decide on transformations or robust modeling.

What did we find:

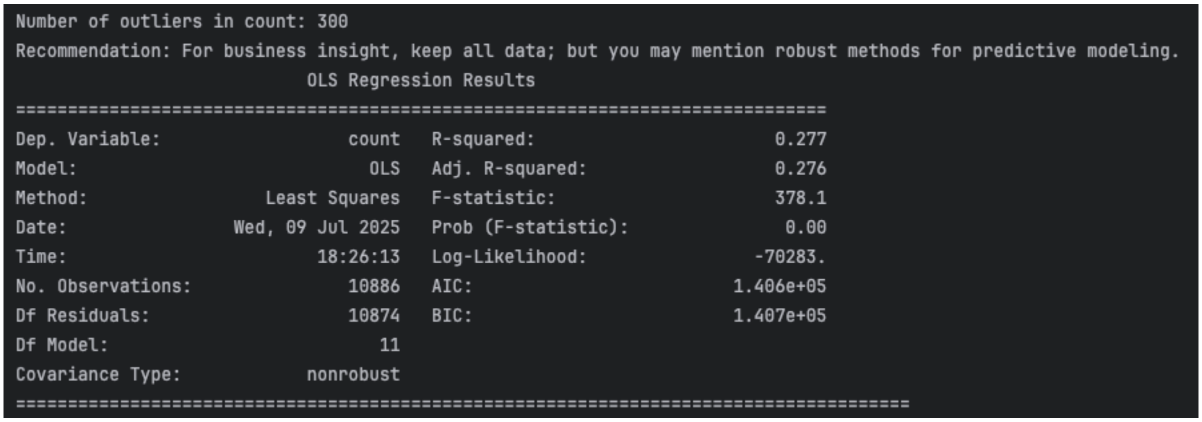
- 300 outliers in the count variable.
- For business analysis, all data kept; robust modeling methods recommended if these are not data errors.

Step 15: Linear Regression Model

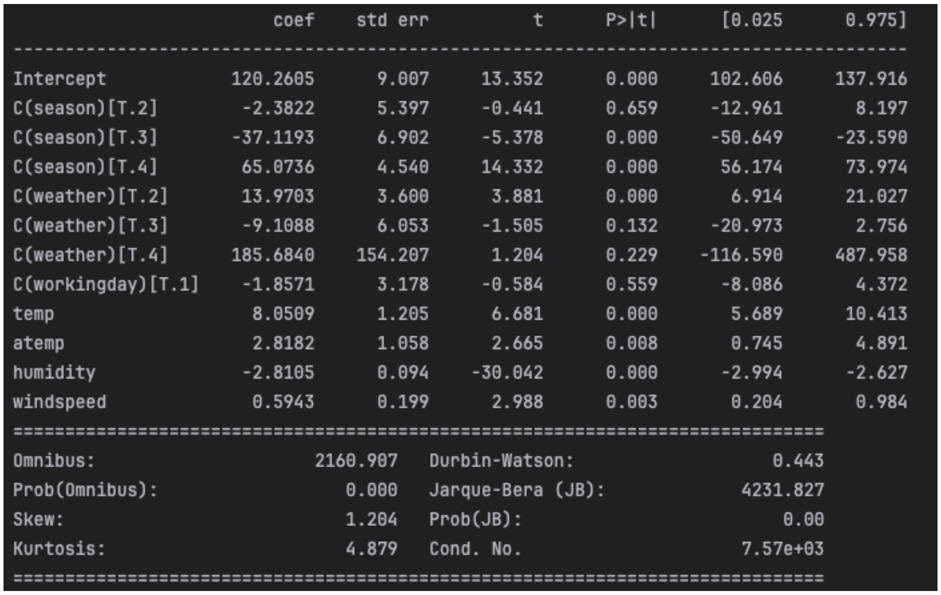
- R^2 = 0.28
- Key coefficients: temp, humidity, season_4, weather_2 significant
- Insert OLS regression summary



```python
# STEP 15: Simple Linear Regression (Bonus)
import statsmodels.formula.api as smf

model = smf.ols( formula: 'count ~ temp + atemp + humidity + windspeed + C(season) + C(weather) + C(workingday)',
                data=df).fit()
print(model.summary())
```



```
Number of outliers in count: 300
Recommendation: For business insight, keep all data; but you may mention robust methods for predictive modeling.
                            OLS Regression Results
==============================================================================
Dep. Variable:                  count   R-squared:                       0.277
Model:                            OLS   Adj. R-squared:                  0.276
Method:                 Least Squares   F-statistic:                     378.1
Date:                Wed, 09 Jul 2025   Prob (F-statistic):               0.00
Time:                        18:26:13   Log-Likelihood:                -70283.
No. Observations:               10886   AIC:                         1.406e+05
Df Residuals:                   10874   BIC:                         1.407e+05
Df Model:                          11
Covariance Type:            nonrobust
======================================================================================
```



```
                         coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
Intercept            120.2605      9.007     13.352      0.000     102.606     137.916
C(season)[T.2]        -2.3822      5.397     -0.441      0.659     -12.961       8.197
C(season)[T.3]       -37.1193      6.902     -5.378      0.000     -50.649     -23.590
C(season)[T.4]        65.0736      4.540     14.332      0.000      56.174      73.974
C(weather)[T.2]       13.9703      3.600      3.881      0.000       6.914      21.027
C(weather)[T.3]       -9.1088      6.053     -1.505      0.132     -20.973       2.756
C(weather)[T.4]      185.6840    154.207      1.204      0.229    -116.590     487.958
C(workingday)[T.1]    -1.8571      3.178     -0.584      0.559      -8.086       4.372
temp                   8.0509      1.205      6.681      0.000       5.689      10.413
atemp                  2.8182      1.058      2.665      0.008       0.745       4.891
humidity              -2.8105      0.094    -30.042      0.000      -2.994      -2.627
windspeed              0.5943      0.199      2.988      0.003       0.204       0.984
==============================================================================
Omnibus:                     2160.907   Durbin-Watson:                   0.443
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             4231.827
Skew:                           1.204   Prob(JB):                         0.00
Kurtosis:                       4.879   Cond. No.                     7.57e+03
==============================================================================
```

What was done:

Fitted a linear regression model using main numeric and categorical features.

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

Why was it done:

To estimate the effect of each feature on demand, and create a baseline predictive model.

What did we find:

- R^2 = 0.28 (model explains 28% of the variance).
- Humidity, temp, atemp, and season_4 (winter) are significant predictors.
- Signs of multicollinearity (e.g., temp/atemp, or categorical dummies).

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

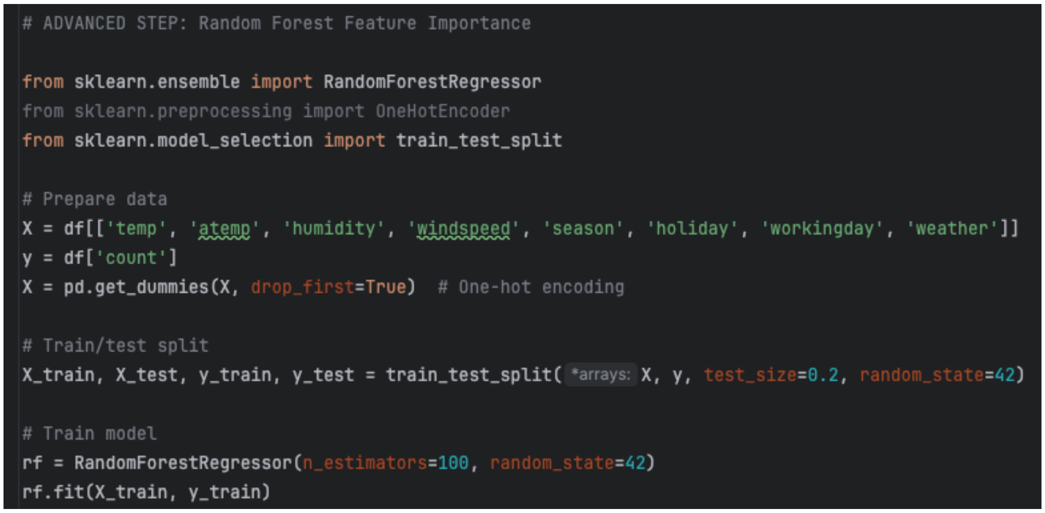
Step 16: Random Forest & Feature Importance

Random Forests are ensemble machine learning models that can capture complex, non-linear patterns. They also provide feature importance metrics, telling us which variables matter most for prediction

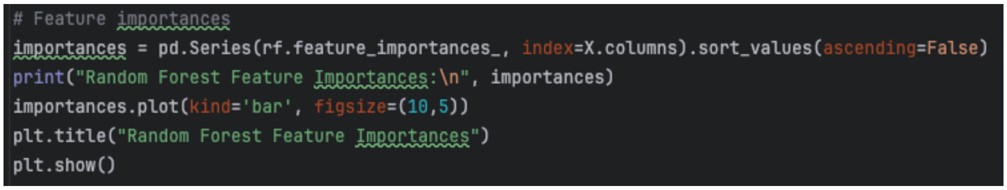
Top features: humidity, atemp, windspeed, temp

Model performance:

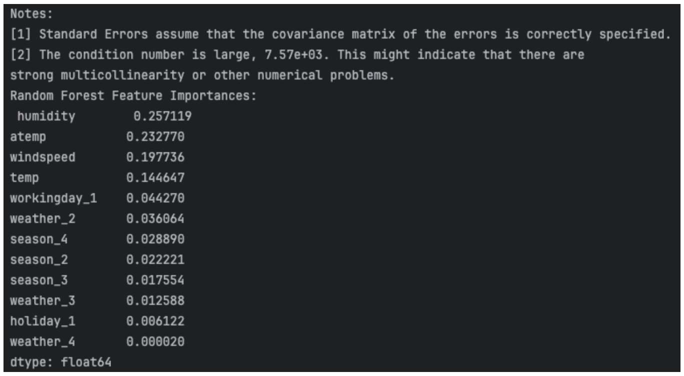
- MAE: 107.6
- RMSE: 151
- R^2: 0.31



```python
# ADVANCED STEP: Random Forest Feature Importance

from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split

# Prepare data
X = df[['temp', 'atemp', 'humidity', 'windspeed', 'season', 'holiday', 'workingday', 'weather']]
y = df['count']
X = pd.get_dummies(X, drop_first=True)  # One-hot encoding

# Train/test split
X_train, X_test, y_train, y_test = train_test_split( *arrays: X, y, test_size=0.2, random_state=42)

# Train model
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
```



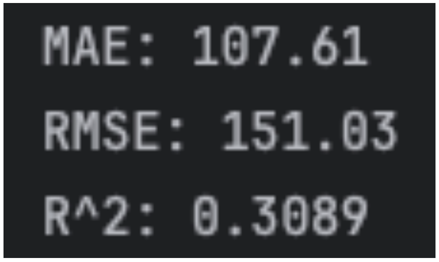
```python
# Feature importances
importances = pd.Series(rf.feature_importances_, index=X.columns).sort_values(ascending=False)
print("Random Forest Feature Importances:\n", importances)
importances.plot(kind='bar', figsize=(10,5))
plt.title("Random Forest Feature Importances")
plt.show()
```



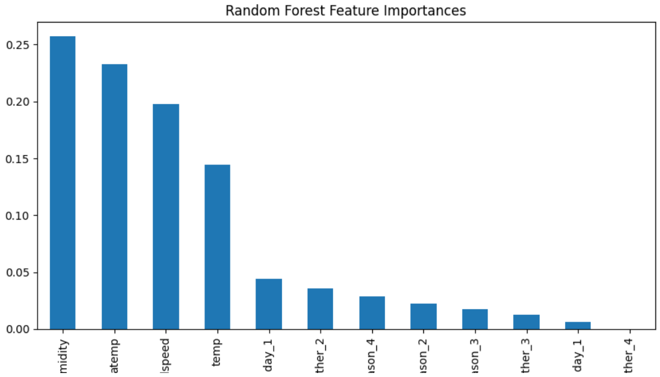
```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 7.57e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
Random Forest Feature Importances:
 humidity       0.257119
atemp          0.232770
windspeed      0.197736
temp           0.144647
workingday_1   0.044270
weather_2      0.036064
season_4       0.028890
season_2       0.022221
season_3       0.017554
weather_3      0.012588
holiday_1      0.006122
weather_4      0.000020
dtype: float64
```

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025



```
MAE: 107.61
RMSE: 151.03
R^2: 0.3089
```



What was done:

Trained a Random Forest Regressor; extracted and plotted feature importances.

Why was it done:

Random Forests capture non-linearities and feature interactions; feature importances show which predictors most influence demand.

What did we find:

- Most important features: humidity, atemp, windspeed, temp.
- Model R^2 = 0.31 (better than linear regression).
- Weather and season features have some, but less, importance than core numeric predictors.

Yulu Bike Sharing Data Analysis & Predictive Modeling

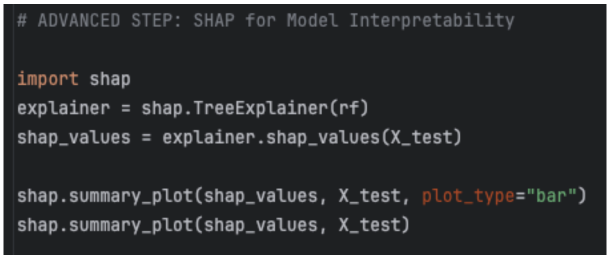
Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

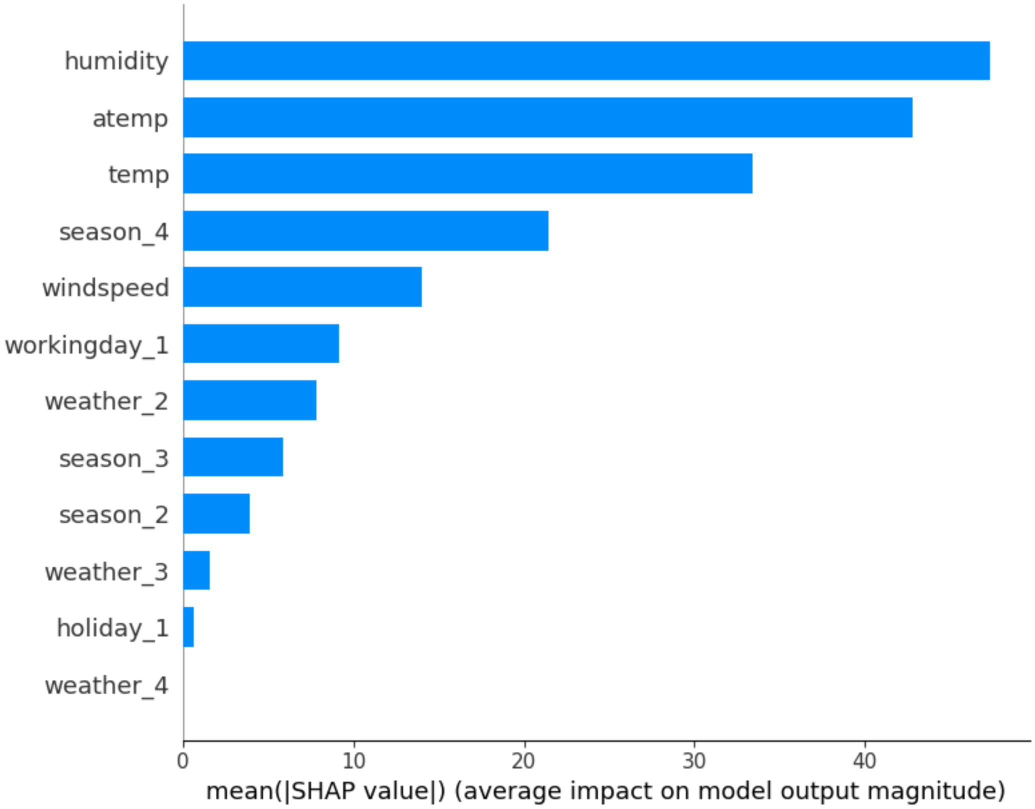
Step 17: SHAP Analysis

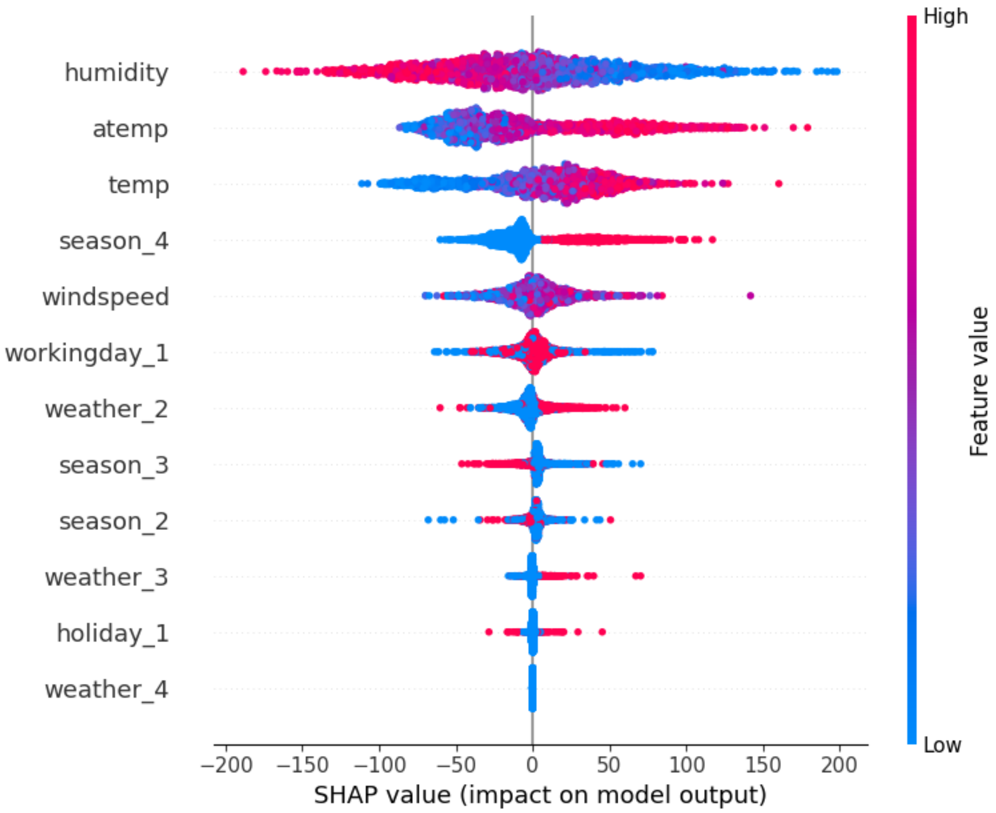
Top impactful features:

- Humidity
- Atemp
- Temp
- season_4
- windspeed



```
# ADVANCED STEP: SHAP for Model Interpretability

import shap
explainer = shap.TreeExplainer(rf)
shap_values = explainer.shap_values(X_test)

shap.summary_plot(shap_values, X_test, plot_type="bar")
shap.summary_plot(shap_values, X_test)
```





What was done:

Used SHAP (SHapley Additive exPlanations) values to interpret the Random Forest model, both globally and per-prediction.

Why was it done:

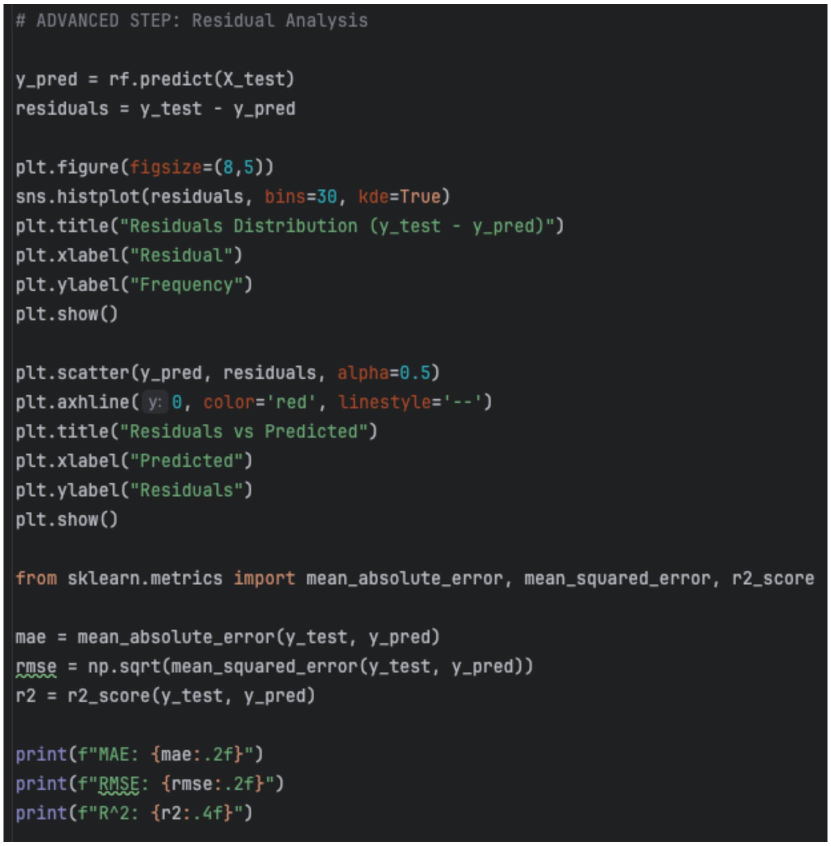
SHAP values provide transparent, quantitative explanation for each prediction, crucial for business applications.

What did we find:

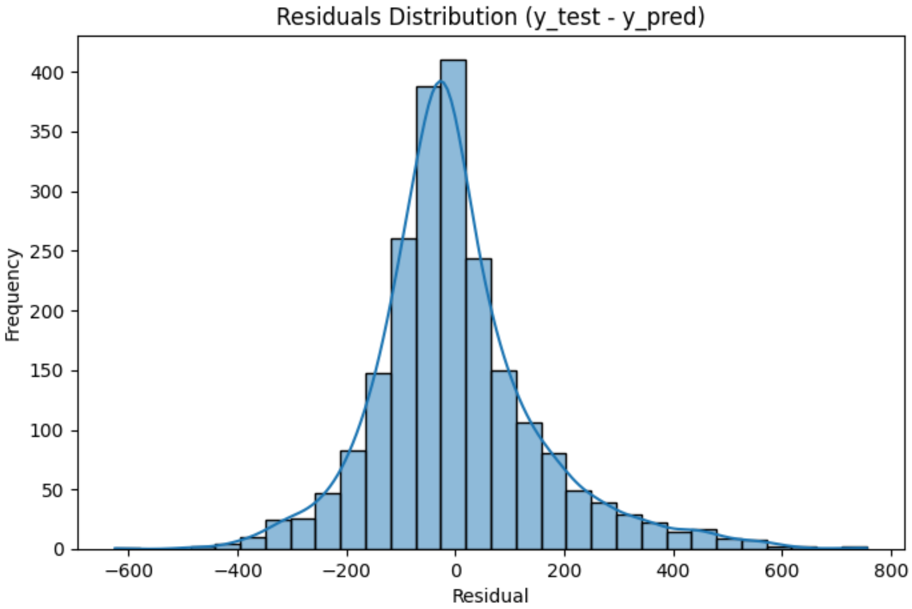
- Confirmed importance of humidity, atemp, temp, windspeed.
- Visualized how each feature (and value) drives predictions up or down.

Step 18: Residual Analysis

Residuals roughly symmetric around zero; some pattern at higher predicted values.

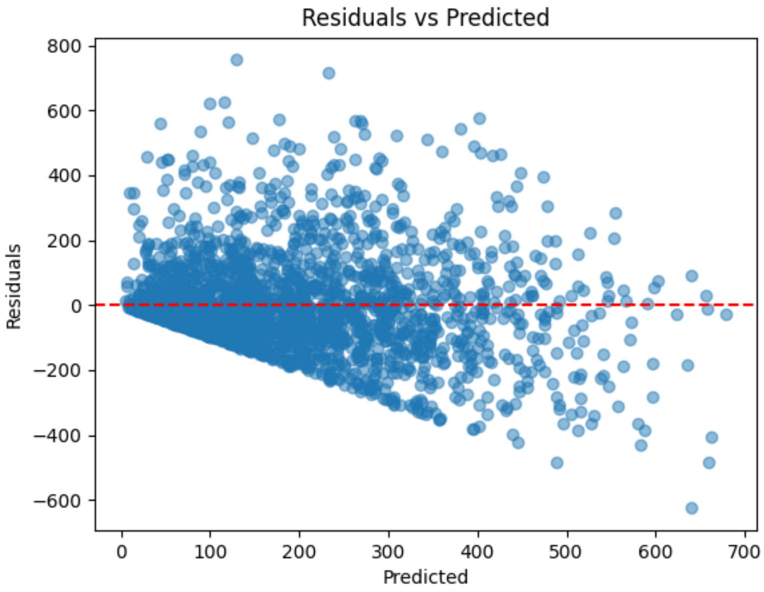


```
# ADVANCED STEP: Residual Analysis

y_pred = rf.predict(X_test)
residuals = y_test - y_pred

plt.figure(figsize=(8,5))
sns.histplot(residuals, bins=30, kde=True)
plt.title("Residuals Distribution (y_test - y_pred)")
plt.xlabel("Residual")
plt.ylabel("Frequency")
plt.show()

plt.scatter(y_pred, residuals, alpha=0.5)
plt.axhline( y: 0, color='red', linestyle='--')
plt.title("Residuals vs Predicted")
plt.xlabel("Predicted")
plt.ylabel("Residuals")
plt.show()

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"MAE: {mae:.2f}")
print(f"RMSE: {rmse:.2f}")
print(f"R^2: {r2:.4f}")
```



Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025



What was done:

Plotted distribution and scatter of residuals (actual - predicted) for the Random Forest model.

Why was it done:

To assess model fit: ideally, residuals should be centered around zero and randomly dispersed.

What did we find:

- Most residuals are close to zero, but some pattern remains (potential further improvement).
- Model could be further refined for high-count hours.

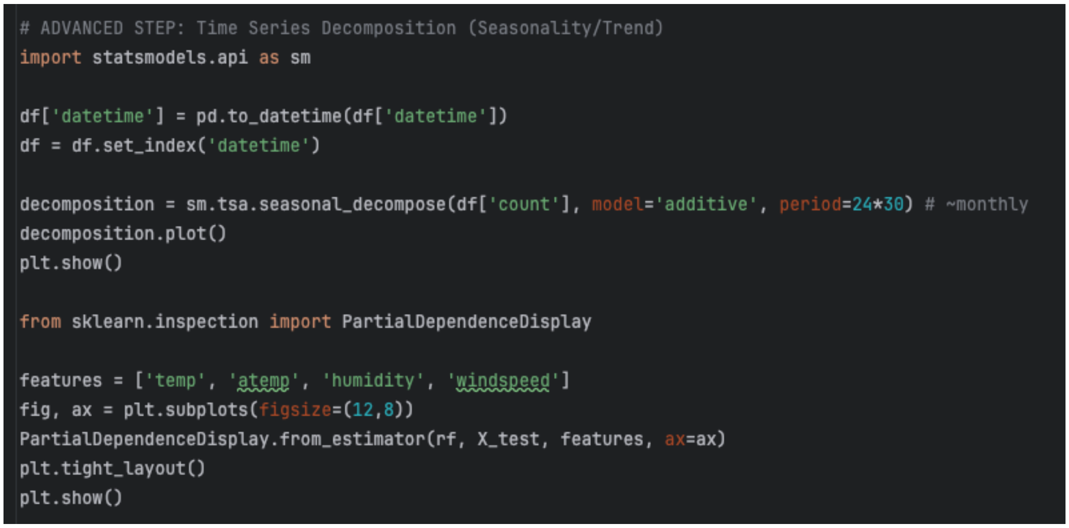
Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

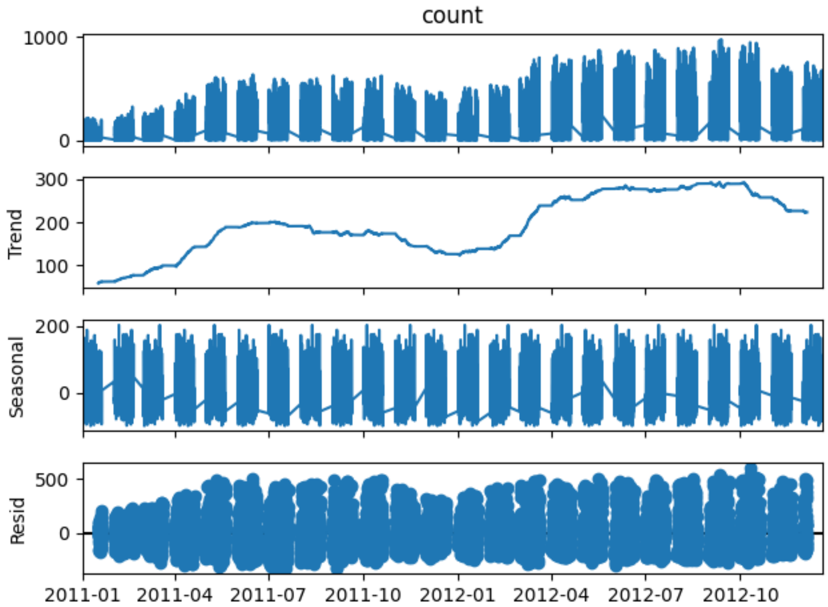
Date: July 2025

Step 19: Time Series Decomposition

- Trend: Increasing usage over time
- Seasonality: Monthly/weekly cycles detected



```
# ADVANCED STEP: Time Series Decomposition (Seasonality/Trend)
import statsmodels.api as sm

df['datetime'] = pd.to_datetime(df['datetime'])
df = df.set_index('datetime')

decomposition = sm.tsa.seasonal_decompose(df['count'], model='additive', period=24*30) # ~monthly
decomposition.plot()
plt.show()

from sklearn.inspection import PartialDependenceDisplay

features = ['temp', 'atemp', 'humidity', 'windspeed']
fig, ax = plt.subplots(figsize=(12,8))
PartialDependenceDisplay.from_estimator(rf, X_test, features, ax=ax)
plt.tight_layout()
plt.show()
```



What was done:

Applied seasonal decomposition to the time-indexed demand data.

Why was it done:

To extract trend and seasonal patterns from the rental counts over time—vital for demand forecasting and resource planning.

What did we find:

- Clear long-term upward trend in rentals.
- Strong periodic (likely monthly and weekly) seasonality.

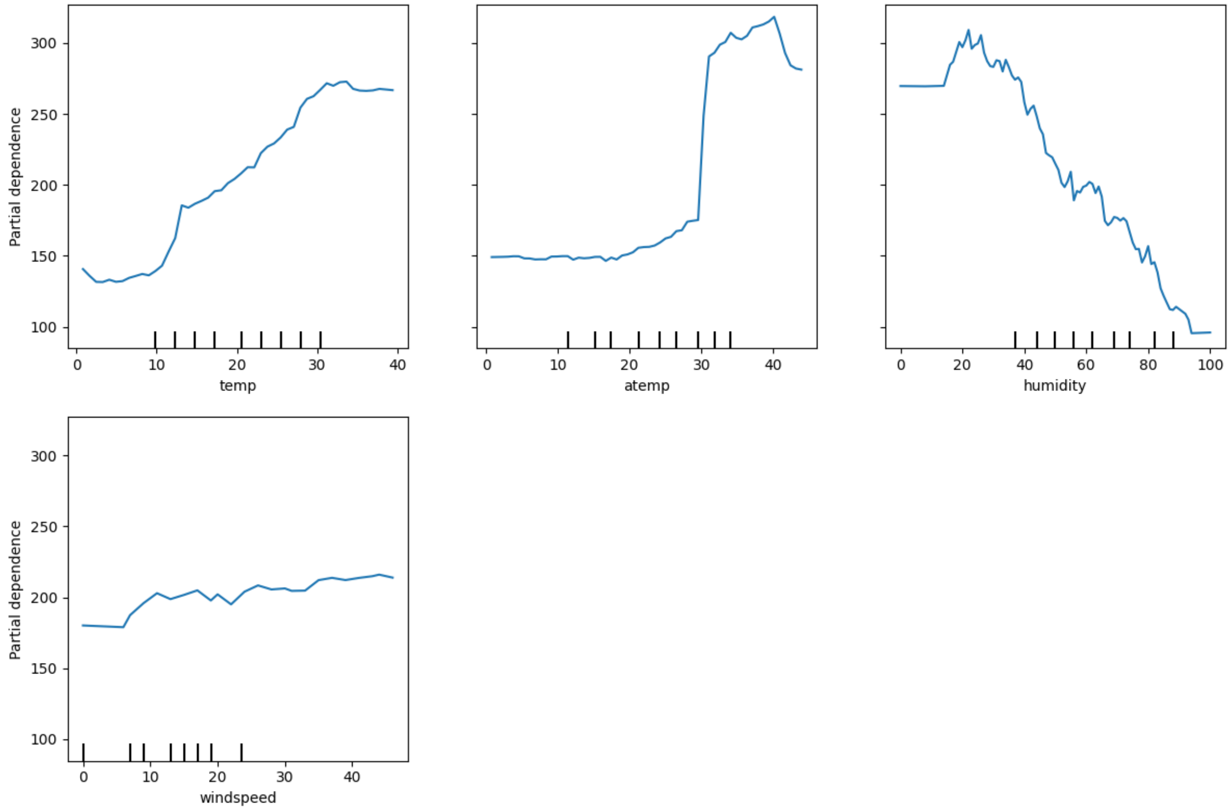
Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

Step 20: Partial Dependence Plots

Shows the marginal effect of each variable on predicted rental counts



What was done:

Plotted partial dependence plots for top predictors using Random Forest.

Why was it done:

To visualize the isolated effect of each feature on predicted demand (controlling for others).

What did we find:

Rentals increase with temp/atemp to a point, then flatten or decline at extremes (e.g., very hot or humid weather).

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

Table of Results

Test	Statistic	p-value	Conclusion
T-Test (Working Day)	**1.24**	**0.2164**	**No significant difference**
ANOVA (Season)	**236.95**	**<0.0001**	**Significant Difference**
ANOVA (Weather)	**65.53**	**<0.0001**	**Significant Difference**
Chi-square (Season/Wea)	**49.16**	**<0.0001**	**Not Independent**

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)

Date: July 2025

Business Recommendations

- Prioritize bike availability and marketing during peak seasons and favorable weather conditions.
- Monitor humidity and temperature closely—these are the best predictors of rental demand.
- Use advanced models (Random Forest) for real-time demand prediction and resource allocation.
- Continue to collect and analyze data for feature engineering and model improvements.
- While our model explains 31% of the variance in rentals, there may be additional factors not captured in this dataset (e.g., local events, pricing).
- Future work could incorporate time-of-day, holidays, or use deep learning methods.

Yulu Bike Sharing Data Analysis & Predictive Modeling

Author: Shivesh Raj Sahu (Ethan)
Date: July 2025

Limitations & Future Work

Limitations:

- Explained Variance:
  The Random Forest regression model explains approximately 31% of the variance in bike rental demand. This means a significant portion of demand remains unexplained by the current set of features.
- Feature Scope:
  The analysis relies only on the available dataset features—weather, season, temperature, humidity, and similar variables. Other external factors like pricing, city events, holidays, local transportation disruptions, promotions, and macroeconomic conditions may strongly impact demand but were not available for modeling.
- Temporal Patterns:
  Although time series decomposition was performed, advanced temporal models (such as ARIMA, SARIMA, Prophet, or LSTM) could provide a deeper understanding of seasonality, long-term trends, and demand anomalies.
- Model Limitations:
  Both linear regression and Random Forest have inherent limitations. Linear regression is sensitive to multicollinearity and assumes linearity; Random Forests, while powerful, can overfit or miss subtle patterns in time-series or rare-event data. Neither model extrapolates well outside the observed data range.
- Interpretability:
  Feature importances and SHAP values improve interpretability, but complex models (like Random Forest) still retain some “black box” characteristics, making it challenging to fully understand all predictions—especially for business stakeholders.

Future Work:

- Expand Feature Set:
  Integrate additional data sources—such as real-time events, pricing changes, marketing campaigns, public holidays, or even traffic and weather forecasts—to capture more drivers of demand.
- Advanced Modeling:

Explore more sophisticated machine learning and deep learning models (e.g., Gradient Boosting Machines, XGBoost, LSTM/Prophet for time-series) to improve predictive accuracy and capture non-linear and temporal effects.

- User Segmentation:
  Segment users by registration status (casual vs. registered), geography, and trip frequency. Personalized models may reveal distinct demand patterns among different groups.
- Geospatial Analysis:
  If spatial (location) data becomes available, analyze demand patterns across neighborhoods. This could enable zone optimization, targeted expansion, or rebalancing of the bike fleet.
- Experimental Design:
  Recommend A/B testing or pilot interventions (e.g., targeted promotions, variable pricing) to measure the causal effect on demand.
- Model Deployment:
  Develop and deploy a real-time dashboard for demand prediction and resource allocation, integrating machine learning model outputs into daily business operations.