**INFOSYS SPRINBOARD 6.0 -BATCH 5 (OCT-13)**

**RideWise — Predicting Bike-Sharing Demand Based on Weather & Events**

**Project Goal, Outcomes, and Use Cases**

**Goal:**

The primary goal of this project is to predict bike-sharing demand using historical usage data, weather information, and temporal factors. By developing a robust regression model, the project aims to assist city planners and operators in optimizing bike availability, improving fleet management, and enhancing user satisfaction through data-driven decisions.

**Expected Outcomes:**

1. Model Development:  
   Build and evaluate multiple regression models (Linear Regression, Random Forest, Gradient Boosting, etc.) to accurately predict bike rental counts.
2. Feature Understanding:  
   Identify the most influential factors affecting bike demand, such as temperature, humidity, and time-based patterns.
3. Operational Insights:  
   Enable smarter decisions in station rebalancing, maintenance scheduling, and resource allocation by forecasting demand trends.
4. Documentation & Visualization:  
   Present findings through visualizations and reports, explaining model behavior and feature importance in clear, interpretable terms.

**Use Cases:**

1. **Predicting Bike-Sharing Demand**
   * Description:  
     Using historical usage records, weather conditions, and temporal attributes (hour, weekday, season), the model predicts the number of bikes rented for a given time period.
   * Application:  
     This supports real-world tasks such as fleet rebalancing, station stocking, and anticipating demand spikes during events or extreme weather.
2. **Assessing Feature Impact**
   * Description:  
     The trained model helps analyze how factors like temperature changes, city events, or neighborhood attributes influence demand.
   * Application:  
     Enables urban planners and policymakers to understand and optimize ridership patterns, plan infrastructure upgrades, and improve the city’s sustainable mobility strategies.

**DATASET SOURCE:** KAGGLE <<https://www.kaggle.com/datasets/lakshmi25npathi/bike-sharing-dataset>>

**ABOUT DATASET:**

- instant: record index

- dteday : date

- season : season (1:springer, 2:summer, 3:fall, 4:winter)

- yr : year (0: 2011, 1:2012)

- mnth : month ( 1 to 12)

- hr : hour (0 to 23)

-holiday : weather day is holiday or not (extracted from http://dchr.dc.gov/page/holiday-schedule)

- weekday : day of the week

- workingday : if day is neither weekend nor holiday is 1, otherwise is 0.

-weathersit :

- 1: Clear, Few clouds, Partly cloudy, Partly cloudy

- 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist

- 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds

- 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

- temp : Normalized temperature in Celsius. The values are divided to 41 (max)

- atemp: Normalized feeling temperature in Celsius. The values are divided to 50 (max)

- hum: Normalized humidity. The values are divided to 100 (max)

- windspeed: Normalized wind speed. The values are divided to 67 (max)

- casual: count of casual users

- registered: count of registered users

- cnt: count of total rental bikes including both casual and registered

**PROJECT FLOW :**

| **Step** | **Name** | **Purpose** | **Tools / Algorithms** |
| --- | --- | --- | --- |
| 1 | Data Cleaning | Handle missing, duplicate, wrong data | Pandas, Excel Power Query |
| 2 | Encoding | Convert categories to numbers | Label Encoding, One-Hot |
| 3 | Scaling | Equalize feature ranges | StandardScaler, MinMax |
| 4 | Feature Engineering | Add logical/derived columns | DateTime breakdowns, ratios |
| 5 | Feature Extraction | Create abstract new features | PCA, Autoencoders, TF-IDF |
| 6 | Feature Selection | Keep the best ones | RFE, SelectKBest, Feature Importance |
| 7 | Splitting | Train/Test | train\_test\_split() |
| 8 | Model Building | Train regression model | Linear, Random Forest, XGBoost |
| 9 | Evaluation | Check model accuracy | MAE, RMSE, R² |

**DATASET OBSERVATION:**

**PROS:**

1. No Missing Values:  
   Both day.csv and hour.csv are already well-structured — no NaNs, no blanks, no weird encoding issues. That’s *rare* and good.
2. Normalized Numeric Features:  
   Features like temp, atemp, hum, and windspeed are already normalized (scaled between 0–1). So, scaling isn’t a must, though re-scaling for model consistency is still a good idea if you add new features.
3. Balanced and Complete Time Coverage:  
   It covers entire 2011–2012, both daily and hourly data. That’s awesome for trend analysis, time-based modeling, and seasonality detection.
4. No ID Leaks:  
   The instant column is just a record index — safe to drop, not a data leak.
5. Categorical Variables are Consistent:  
   Columns like season, weathersit, holiday, workingday, weekday, mnth, and yr are all numeric but categorical. So you can cleanly one-hot encode them.

**AREA TO FOCUS:**

1. Feature Redundancy (temp vs atemp):  
   temp (actual temp) and atemp (feeling temperature) are *highly correlated (~0.99)*.  
   → Keep only one (usually atemp gives slightly better correlation with demand).
2. Normalization Confusion:  
   Since temperature, humidity, and windspeed are scaled, any feature engineering (like temp difference or wind chill) will require reversing normalization (multiply back to real-world values).
3. Categorical Variables as Numbers:  
   season = 1,2,3,4 and weathersit = 1,2,3,4 are *not ordinal* (the model might assume linearity).  
   → Must convert to One-Hot Encoding or treat as categorical.
4. Temporal Leakage:  
   Including casual and registered when predicting cnt is data leakage — they literally sum up to cnt.  
   → Drop both; use only features available *before* prediction time.
5. Time-Series Nature Ignored:  
   Currently it’s just treated as rows — but time correlation is huge here (hour of day, day of week, seasonality).  
   → Need feature engineering to capture this temporal behavior.

**Data Preprocessing and Feature Selection**

**1. Data Inspection & Cleaning**

* **Goal:** Understand dataset structure, detect issues, and ensure clean inputs for modeling.
* **Steps & Observations:**
  1. **Data overview:** Dataset contains 17,379 rows and 17 columns, with mixed numeric and categorical features.
  2. **Data types check:** dteday converted from string to datetime to enable temporal feature engineering.
  3. **Missing values:** No missing values found—data is clean and well-structured.
  4. **Duplicate records:** None detected.
  5. **Outlier handling:** Features like cnt and windspeed may have extreme spikes; handled using the **IQR method** and **Z-score** for extreme cases.
  6. **Dropping unnecessary columns:** instant (record index), casual, and registered removed to prevent **temporal leakage** (they directly reveal the target cnt).

**note:** Temporal leakage occurs when features include information from the future or directly contain the target, which can make the model overfit.

**2. Encoding Categorical Variables**

* **Categorical Columns:** season, weathersit, mnth, weekday, holiday, workingday, yr, hr.
* **Approach & Observations:**
  1. **Ordinal encoding:** season and weathersit treated as ordinal for certain models where order matters.
  2. **One-Hot Encoding:** Applied to nominal features (weekday, mnth, holiday) to avoid false assumptions of linearity.

**note:** One-Hot Encoding converts each category into a separate binary column (0 or 1) so models don’t assume an order where none exists.

**Insight:** One-hot avoids misleading coefficients in linear models and ensures categorical meaning is preserved for tree-based models.

**3. Feature Scaling**

* **Numeric Features:** temp, atemp, hum, windspeed.
* **Approach & Observations:**
  1. Already normalized (scaled 0–1) but slight rescaling applied using **StandardScaler** to maintain consistency across engineered features.
  2. Scaling ensures that models like Linear Regression treat each feature **equally**, avoiding domination by larger values.

**note:** StandardScaler subtracts the mean and divides by standard deviation to center data around zero with unit variance.

**4. Feature Engineering**

* **Goal:** Capture temporal, behavioral, and weather-driven patterns to improve prediction accuracy.
* **Derived Features & Observations:**
  1. hour\_category → morning, afternoon, night: captures diurnal demand patterns.
  2. is\_weekend → 1 for Saturday/Sunday, else 0: reflects behavioral differences.
  3. day\_type → combines holiday and working day logic.
  4. temp\_feel\_diff → difference between atemp and temp: captures perceived temperature effects.
  5. Lag and trend features: cnt\_lag\_1 (previous hour demand), cnt\_roll\_mean\_3 (3-hour rolling average) to capture time-series dependencies.
  6. Sin/Cos transformations (sin\_hour, cos\_hour) to capture cyclical nature of hours.

**note:** Lag features use previous time steps as predictors; sin/cos transforms help the model learn cyclical patterns like hourly peaks.

I**nsight:** These engineered features address the **time-series nature** of the dataset, allowing even non-sequential models (like RandomForest) to learn temporal dependencies.

**5. Feature Selection**

* **Goal:** Retain only the most informative features to improve model interpretability and performance.

| **Step** | **Purpose** | **Tools / Technique** | **Observations & Decisions** |
| --- | --- | --- | --- |
| Correlation Analysis | Detect redundant or irrelevant features | df.corr(), heatmap | Dropped one of temp/atemp (correlation >0.98) to reduce multicollinearity |
| Remove Leakage Variables | Prevent overfitting | Domain knowledge | Dropped casual, registered, instant |
| Variance Threshold | Remove near-zero variance features | VarianceThreshold(threshold=0.01) | One-hot columns with very low counts removed |
| Univariate Feature Selection | Rank single-variable strength | SelectKBest(f\_regression) | Top predictors identified: hour, temp, hum, weathersit, season |
| Multicollinearity Check | Detect confusing feature correlations | **VIF** (Variance Inflation Factor) | Dropped features with VIF > 10 to stabilize linear models |
| Tree-Based Feature Importance | Model-driven feature ranking | RandomForest, XGBoost | RandomForest chosen for non-linear importance; confirmed key features include hour, temp, hum, weathersit, workingday, is\_weekend |
| Recursive Feature Elimination (RFE) | Systematically remove weak features | RFE() with LinearRegression or RandomForest | Optimal 10–12 features retained for final model |
| Cross-Validation | Validate feature subsets | cross\_val\_score() | RMSE/R² plateaued after selected features → final feature set chosen |

**note:**

* **VIF:** Measures multicollinearity; high VIF → redundant predictors.
* **RFE:** Recursively removes least important features to find best subset.

**Insight:**

* RandomForest is robust to non-linear relationships and provides feature importance directly.
* RFE allows systematic pruning of features, particularly useful when mixing linear and tree-based methods.
* Combining correlation analysis, VIF, and tree-based importance ensures **redundancy is minimized, leakage is removed, and temporal + weather features are prioritized**.

**6. Final Feature Set for Modeling**

['hour', 'season', 'weathersit', 'workingday', 'is\_weekend',

'temp', 'hum', 'windspeed', 'sin\_hour', 'cos\_hour',

'cnt\_lag\_1', 'cnt\_roll\_mean\_3']

**Rationale:**

* **Temporal context:** hour, sin\_hour, cos\_hour, is\_weekend, workingday, cnt\_lag\_1, cnt\_roll\_mean\_3
* **Weather conditions:** temp, hum, windspeed, weathersit
* **Behavioral patterns:** workingday, is\_weekend
* **Trend analysis:** cnt\_lag\_1, cnt\_roll\_mean\_3

This ensures a **balanced, informative feature set** that captures **time, weather, behavior, and recent trends**, allowing regression models to make accurate bike demand predictions.