Holly Cohan,Yesh Onipede, Yu Xia

Week 5 Report: Feature Engineering

The objective of this week's assignment is to engineer new features using existing fields in our dataset to improve the prediction of rideshare prices. By enriching the dataset with contextually meaningful variables related to weather, time, and events, we aim to better capture fluctuations in rideshare prices based on external influences, such as traffic, weather conditions, and major events in Boston.

Tradeoff of Reducing Dimensionality vs. Including Relevant Engineered Features

Before we proceed discussing the features we have engineered, it is important to address the tradeoff of Reducing Dimensionality vs. Including Relevant Engineered Features. In the context of feature engineering, there is always a tradeoff between reducing dimensionality and adding additional relevant features. On the one hand, reducing dimensionality can simplify models, decrease computational costs, and prevent overfitting, particularly when many features are redundant or irrelevant. This can lead to more efficient training and better generalization to new data. On the other hand, adding engineered features can capture important patterns and variability in the data, potentially improving the model's performance by explaining more of the variance in prices. For example, factors such as rush hour, extreme weather, or a local sports game may have significant effects on rideshare prices that would be missed without these additional variables.

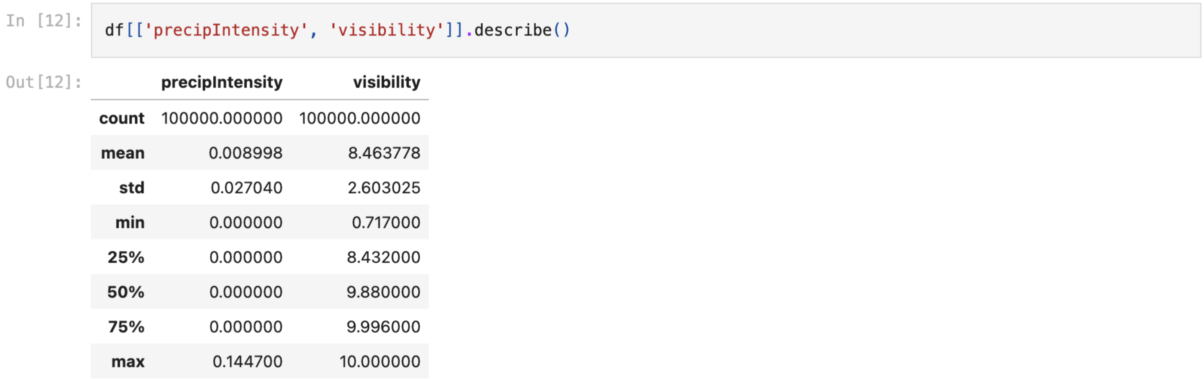
While these new features add complexity by increasing the dimensionality of the dataset, they also make the models more interpretable, especially in cases where each feature directly corresponds to a real-world phenomenon that influences price variability. Balancing the two approaches is critical: the goal is to introduce meaningful features that enhance prediction without unnecessarily increasing noise or complexity.

New Features Engineered:

Weather related Features:

- There are several variables, icon, short_summary, and long_summary, that are very similar in its context. We are thinking about combining them into one variable to reduce redundancy and dimensionality.
- We used a custom weighting system which assigns scores to each type of weather event based on severity or relevance. The weighting system will assess the weather condition by considering all three variables above combined with visibility.
- To calculate the weights, we need to understand how these variables actually reflect the weather on that date and how severe the conditions are for driving. We first took a look at the precipitation intensity for these days, and as shown below, precipitation is all under 0.1447, which is considered moderate. However, there are days where visibility is very low (less than 1 mile). To factor these in, we determined a weighting system that will penalize with low visibility or reward

with high visibility to differentiate between various weather conditions indicated in the icon, short_summary, and long_summary variables.



In [12]:
```
df[['precipIntensity', 'visibility']].describe()
```

Out[12]:

	precipIntensity	visibility
count	100000.000000	100000.000000
mean	0.008998	8.463778
std	0.027040	2.603025
min	0.000000	0.717000
25%	0.000000	8.432000
50%	0.000000	9.880000
75%	0.000000	9.996000
max	0.144700	10.000000

- We wrote a function to assess weather severity based on these ideas. Replacing these variables with the new variable severity effectively reduces the dimensionality of the dataframe and improves interpretability.

Time related Features:

- Rush Hour: Identifies whether the ride occurred during typical rush hours (6-9 AM or 4-6 PM), which are known to experience higher demand.

- Weekend: Identifies whether the ride took place on a weekend (Saturday or Sunday).

- Bruins or Celtics Game Day: Identifies if the ride took place on a day when the Boston Bruins or Celtics had a game in the city.

While none of these variables reduce the dimensionality of the dataset, we feel that all three could have a critical impact on the prices of rideshare, and therefore are imperative to include in the model. It is commonly known that rideshare prices tend to be higher on weekends and during rush hour on weekdays, and we would like to better understand these relationships. There is also an enormous demand for rideshares during and after sporting events in Boston, so we plan to explore these relationships as well.

Data Preprocessing:

After all the feature engineering and encoding, we were left with 84 columns in our features dataset. Since we conducted feature engineering on the original data before preprocessing, we ran the pipeline again to prepare the data for modeling.

Imputing Missing Values:

- We imputed missing values of the Uber Taxi rides using the Taxi fare breakdowns in the Uber app, as explained in the previous report.

Encoding Categorical Variables:

- We encoded the categorical variable using dummy coding, one-hot encoding, and ordinal encoding based on the type and context of the variable.
- After encoding, we were left with 84 features, which are significantly less than 110 variables without feature engineering.

Standardization:

- We also standardized the features using StandardScaler to ensure that each feature contributes equally to the distance calculations involved in PCA. By combining PCA with standardization, we can effectively reduce dimensionality while preserving the most significant variance in our dataset, making it more suitable for predictive modeling techniques.

Principal Component Analysis:

- We conducted PCA again to address high dimensionality and correlation identified. We set a threshold of 0.95 for the cumulative explained variance ratio, which resulted in retaining **N** principal components. This threshold was chosen to ensure that we capture at least 95% of the variance in the dataset while reducing the dimensionality of our feature space. By selecting these components, we aim to minimize noise and redundancy in the data, thus enhancing the model's performance and interpretability. As the result, the PCA reduce the features to 45 components, which is about half of the original training dataset after feature engineering and categorical variable encoding.