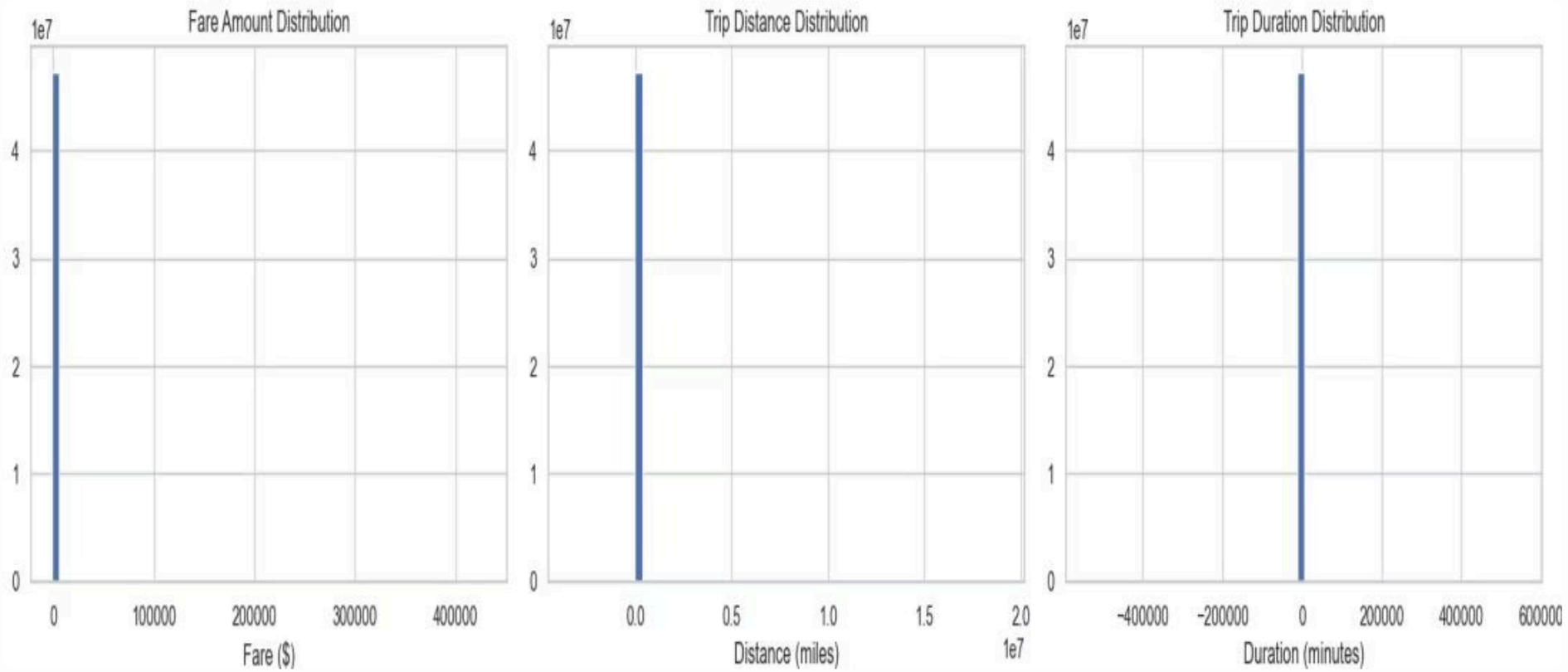
NYC Taxi Fare Prediction – KNN Model Report

by Daniel Miuta



Executive Summary

A K-Nearest Neighbors (KNN) regression model was developed to predict NYC taxi fares. The analysis includes preprocessing, feature engineering, scaling, outlier handling, hyperparameter optimization, and visual inspection of model performance. KNN demonstrates solid baseline accuracy and interpretability.

Data Overview & Processing

Dataset Characteristics

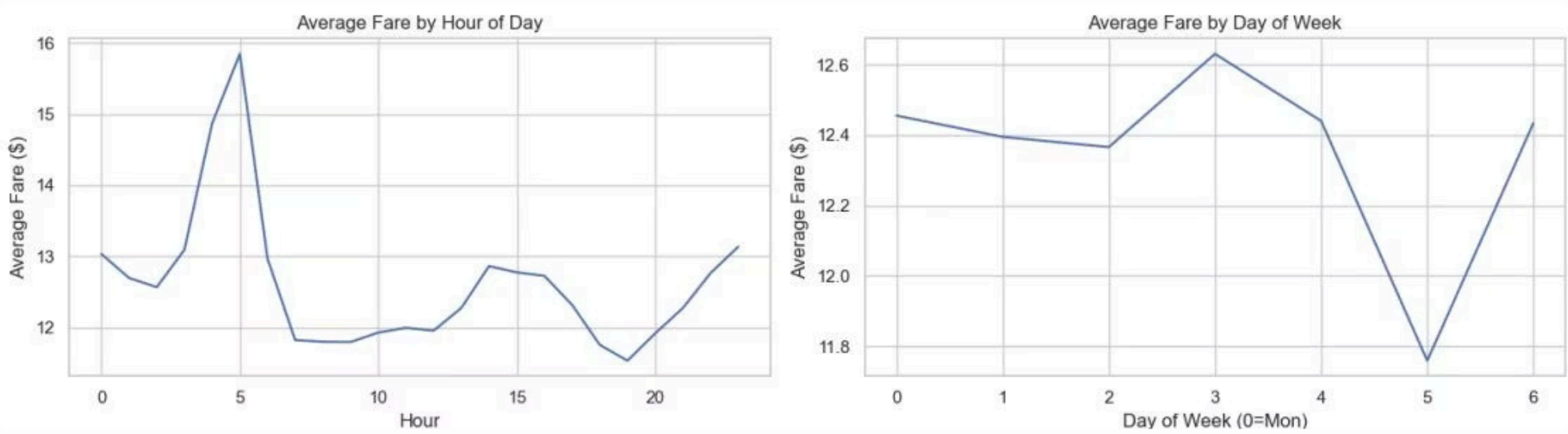
- NYC Yellow Taxi trip records
- Raw CSV inputs containing timestamps, coordinates, trip metrics, and fare values
- Engineered features: trip_duration, hour, day_of_week, speed_mph

Key Processing Steps

1. Timestamp conversion for pickup and dropoff
2. Feature engineering: duration, temporal features, speed
3. Geographic filtering: NYC bounding box
4. Removing invalid or extreme distances, durations, and fares

Three histograms illustrating the distribution of key features:

- **Fare Amount Distribution**: Shows a highly concentrated distribution centered around $0, with a maximum value of approximately $4.7 million (1e7).
- **Trip Distance Distribution**: Shows a highly concentrated distribution centered around 0.0 miles, with a maximum value of approximately 2.0 miles (1e7).
- **Trip Duration Distribution**: Shows a highly concentrated distribution centered around 0 minutes, with a maximum value of approximately 200,000 minutes (600,000).



Two line graphs illustrating fare patterns:

- **Average Fare by Hour of Day:** Shows the average fare ($) over the hour of day (0 to 24). The fare is generally stable between 12.0 and 13.0, with a sharp peak around 5 hours (reaching approximately 16.0) and a sharp dip around 20 hours (reaching approximately 11.8).
- **Average Fare by Day of Week:** Shows the average fare ($) over the day of week (0=Mon to 6=Sat). The fare is stable between 12.2 and 12.6, with a peak around day 3 (approximately 12.6) and a deep trough around day 5 (approximately 11.8).

Model Development & Training

Algorithm Selection: KNN

KNN is a distance-based algorithm that requires standardized features for reliable performance. It serves as a transparent baseline model, showing how fares correlate with nearby similar trips.

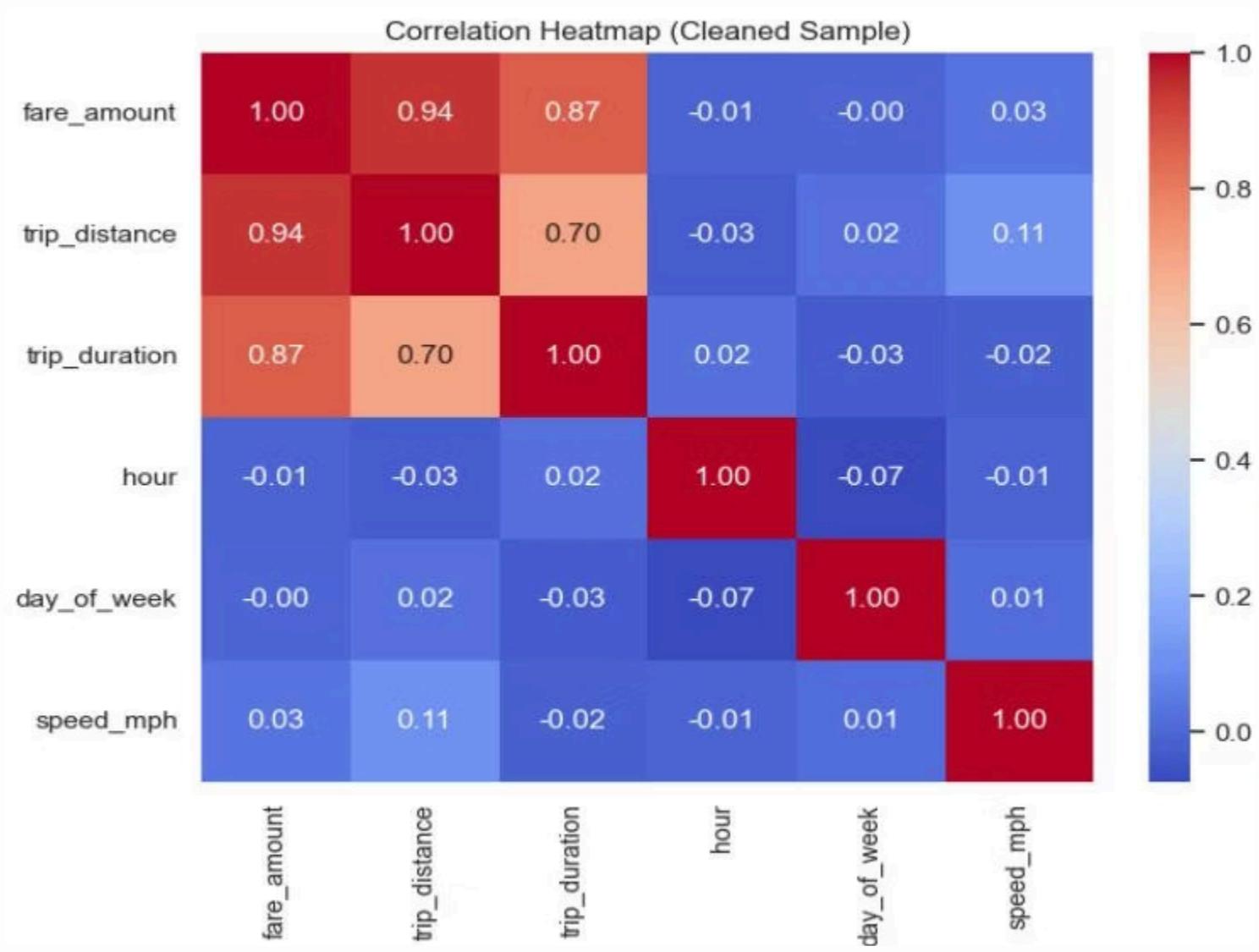
Feature Preparation

- Standardized key variables using a scaling pipeline
- Core features: distance, duration, speed, temporal variables, coordinates

Hyperparameter Optimization

GridSearchCV tuned:

- n_neighbors (K)
- weights (uniform/distance)
- metric (euclidean/manhattan)



Correlation Heatmap (Cleaned Sample)

	fare_amount	trip_distance	trip_duration	hour	day_of_week	speed_mph
fare_amount	1.00	0.94	0.87	-0.01	-0.00	0.03
trip_distance	0.94	1.00	0.70	-0.03	0.02	0.11
trip_duration	0.87	0.70	1.00	0.02	-0.03	-0.02
hour	-0.01	-0.03	0.02	1.00	-0.07	-0.01
day_of_week	-0.00	0.02	-0.03	-0.07	1.00	0.01
speed_mph	0.03	0.11	-0.02	-0.01	0.01	1.00

Color scale (Correlation values): 1.0 (Red), 0.8, 0.6, 0.4, 0.2, 0.0 (Blue).

Performance Results & Production Readiness

Primary Metrics

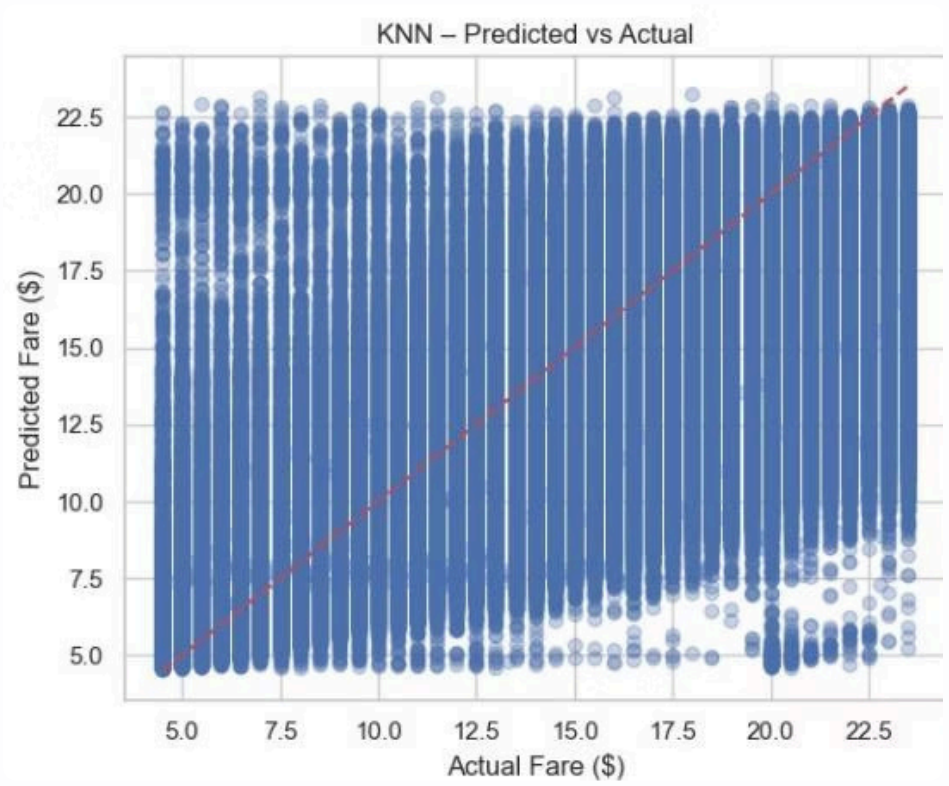
The KNN model achieved the following results:

- RMSE $\approx$ 0.90
- $R^2 \approx$ 0.98
- MAE $\approx$ 0.30

These values show consistent prediction ability over cleaned data.

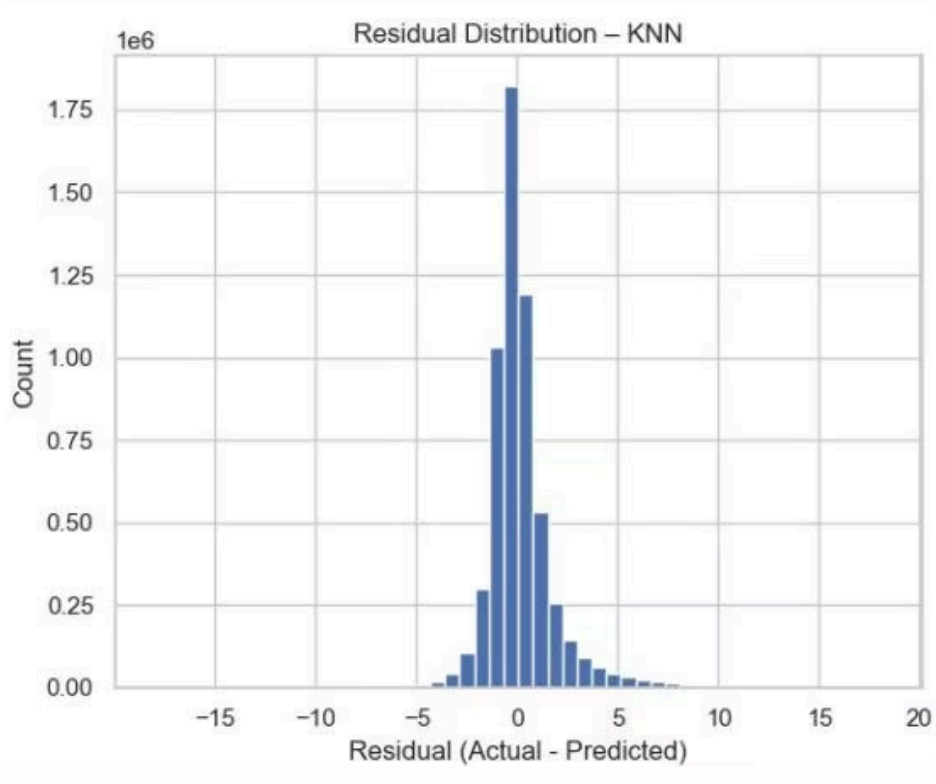
Residual & Prediction Analysis

- Predicted vs actual fares align linearly
- Residual distribution is centered around zero
- Larger errors appear in long or extreme trips



KNN – Predicted vs Actual

Scatter plot showing Predicted Fare ($) versus Actual Fare ($). The x-axis (Actual Fare) ranges from 5.0 to 22.5. The y-axis (Predicted Fare) ranges from 5.0 to 22.5. A red line indicates a linear fit. The data points are clustered, suggesting the model predicts fares accurately.



Residual Distribution – KNN

Histogram showing the Count of residuals (Actual - Predicted) versus the residual value. The x-axis ranges from -15 to 20. The y-axis (Count) is logarithmic, ranging from 0.00 to 1.75. The distribution is centered around 0, indicating a good fit with minimal residuals.

Production Readiness & Business Impact

Performance Benchmarks

- Good baseline accuracy
- Requires standardized inputs
- Slower inference for large datasets versus tree-based models

Scalability Assessment

- Training cost: minimal
- Inference cost: high due to neighbor distance scanning
- Suitable for small to medium datasets

Critical Success Factors

1. Removing outliers
2. Accurate geographic filtering
3. Feature standardization
4. Hyperparameter tuning