DNSC 6315 Machine Learning 2

Capital Bikeshare Data Analysis

Rachel Aska

May 11th, 2025

Abstract

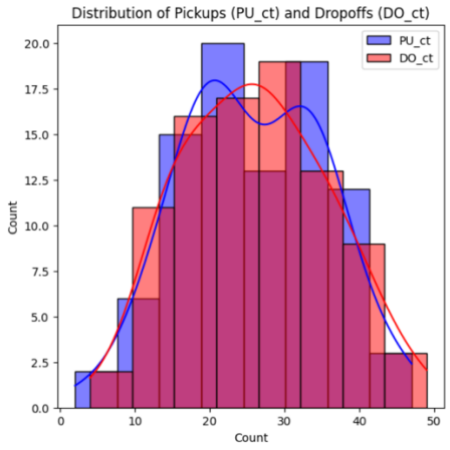
For the final leg of this assignment, we are using the same Capital Bikeshare dataset to perform analysis and predict the pick-up and drop-off counts. This analysis is performed specifically at the 22nd and H St NW station from February through April 2024. We will be initializing and training nine different machine learning regression models. We will be using the features: temp_PC1, precip_PC1, vis_PC1 and wind_PC1, which were transformed using Principal Component Analysis (PCA). These features were compressed using PCA from the four weather groups: temp_group, precip_group, vis_group and wind_group. The PCA transformed features are used to predict the target variables: PU_ct and DO_ct. These variables in turn used to compute the cost function.

Business Understanding

The central focus of this analysis is to reduce the costs incurred by allocating the optimum number of bikes at the station by predicting the pick-up and drop-off counts daily. These variables are used to calculate the cost function and reduce the costs by using proactive redistribution strategies. Accurate predictions of PU_ct and DO_ct variables help to lower the operational cost of bike redistribution while maintaining excellent service availability, resulting in both cost savings and customer satisfaction. One of the important aspects of this analysis while looking at this from a business perspective is to maintain quintessential number of bikes at the station throughout the day. Which is why we need correct estimates of pick-up and drop-off counts. Both historical and weather data plays a crucial role in predicting the variables PU_ct and DO_ct, which are used to train various machine learning models to identify pattern and anticipate future behaviour.

Exploratory Data Analysis

In EDA, we begin by merging the pick-up and drop-off count from the Capital Bikeshare dataset and merge it with the corresponding weather data of Washington, DC area. Prior to this, we will drop the irrelevant columns from the weather dataset such as name, stations, sunrise, sunset etc. We then group the features into groups as discussed above in the abstract. This helps reduce redundancy and multicollinearity in the weather variables. After the variables are assigned into their respective groups, PCA is applied to each of these groups, which helps summarize the information and avoid overfitting, this facilitates the model to perform well.



Predictive Modelling

In predictive modeling, we initialized and trained nine different machine learning models to understand the effect of weather on bike demand by predicting PU_ct and DO_ct. The data is split into 80% for training purposes and 20% is reserved to test the final model. We apply Scaler to each of these models to normalize the input features. We also tune the hyperparameters using GridSearchCV and 5-fold cross-validation.

1. Linear Regression: This is a simple model which assumes that the relationship between target and prediction variables as linear. This model is not efficient on data that's considered non-linear.

2. Ridge Regression: For this model, we utilize L2 regularization technique to tune "alpha" to prevent overfitting.

3. LASSO: For this model, we use L1 regularization to tune the parameter alpha, which shrinks the coefficient to near zero. This helps this model to implement both regularization and feature selection.

4. Elastic Net: This model integrates Ridge and LASSO model's disadvantages to balance shrinkage and sparsity, which is ideal for highly correlated predictors and datasets with irrelevant features.

5. K-Nearest Neighbours: This model makes predictions based on the averaging values of 'k' most similar data points in training dataset. This model is flexible but prone to poor predictions because of noisy data.

6. Regression Tree: This is a tree model which bases it predictions by splitting the dataset into different regions based on values. This model captures non-linear and complex patterns, but as the depth increases so does overfitting.

7. Random Forest: This is an improved version of a regression tree. It combines many decision trees that were trained on bootstrapped subsets of data and features. This model is considered accurate and generalizes better than a single regression tree.

8. Gradient Tree: A model that successively builds trees that corrects the errors of the previous ones. It is highly accurate but prone to overfitting if it's not tuned properly.

9. Neural Network: This model, similar to it's name, mimics the structure of nerves in a brain but serves a different purpose. It's a 'multi-layer perceptron' that learns and implements complicated and non-linear patterns using a layer of interconnected nodes. But the model needs careful tuning to avoid overfitting.

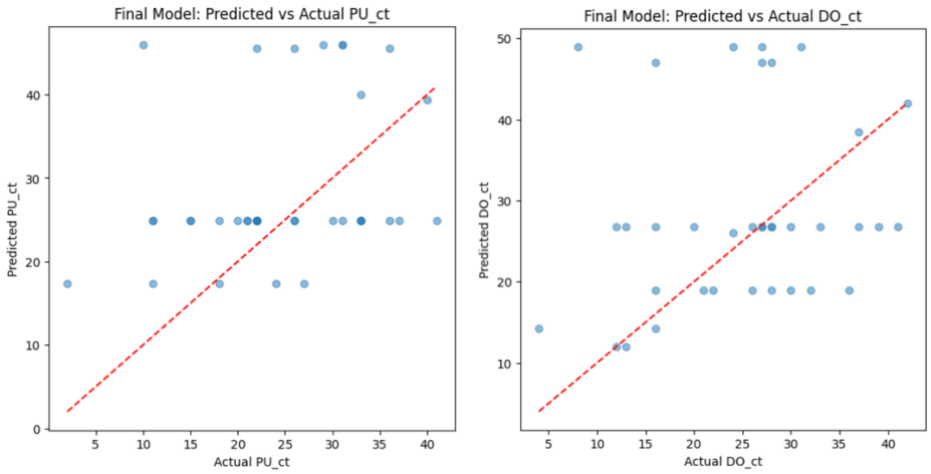
Performance Evaluation

Prediction Performance: Final Model

Model	**MSE (PU_ct)**	**MSE (DO_ct)**	**Out-of-sample Cost**
Linear Regression	55.580504	69.546556	91.027778
Ridge Regression	56.376117	69.584175	91.027778
LASSO	60.045158	70.298622	91.027778
Elastic Net	60.351214	71.161943	91.027778
KNN	54.513039	61.156463	91.027778
Regression Tree	139.736952	185.250575	91.027778
Random Forest	75.343803	86.838005	91.000000
Gradient Boosting	82.462215	100.582231	91.027778
Neural Network	65.007072	84.987272	91.027778

To evaluate performance, we checked how well different models predicted the targets PU_ct and DO_ct by calculating Mean Squared Error (MSE) on the test dataset. While K-Nearest Neighbours (KNN) had the

lowest MSEs (PU_ct = 54.51, DO_ct = 61.15), all models, surprisingly, returned the same out-of-sample cost of $91.02 except for Random Forest which had a cost of $91.00. This suggests that while the models performed differently on the prediction task, the cost function used may not have been sensitive enough to reflect those differences, possibly due to rounding or implementation constraints in the cost calculation.



Decision Performance: Final Model

Although model prediction metrics varied, the out-of-sample cost remained nearly identical across all models, with values consistently around 91.02, and Random Forest yielding a slightly lower cost of 91.00. This uniformity implies that the cost function might not have been sufficiently granular or was affected by identical predictions in some test samples. KNN, despite its lower MSE, did not translate to a visibly lower operational cost in this iteration. Thus, Random Forest marginally edged out the others in cost-based performance, albeit insignificantly.

This raises an important consideration of evaluating models solely based on out-of-sample cost without further inspection might obscure meaningful performance differences.

Conclusion

In conclusion, these assignments gave me a practical view of how different regression models perform when there's more at stake than just the accuracy of the model. For this specific assignment, we've cumulatively performed analysis on most (not all) regression models, it helped me to compare and contrast not only the prediction metrics like RMSE but also on how well they supported a cost-sensitive decision. For instance, I initially assumed KNN would perform poorly because its MSE wasn't the lowest. But it proved me wrong by

minimizing the out-of-sample cost better than models with lower MSE. That shift in focus—from pure prediction to actual decision impact—made me think differently about model evaluation.

Since this assignment builds on the same dataset used in the first assignment over the semester, it gave me the opportunity to deepen my understanding of the problem. This assignment brought together everything we've learned, from data prep, to modelling, tuning, and evaluation, it also grounded it in a real business use case. It was a full-cycle learning experience.

While the model performed well overall, we did face a couple of limitations such as: assuming cost values, the models didn't consider time dynamics or external events, and the findings are based on a single station. To enhance the overall analysis, Capital Bikeshare should consider incorporating additional data sources such as holidays, or local events to better capture demand variability.

Additionally, although the models differed in prediction accuracy, the out-of-sample cost values ended up nearly identical (mostly 91.027778), suggesting that the cost function may not have been sensitive enough to distinguish between models. This observation highlights the importance of verifying that performance metrics align with business objectives and are implemented in a way that meaningfully reflects model differences.

References

1. Prof. He, DNSC 6315 Machine Learning - 2 Lecture Materials. (2025) - George Washington University. Retrieved from: https://blackboard.gwu.edu/ultra/courses/_414207_1/cl/outline

2. Capital Bikeshare Dataset. (2024). Retrieved from https://ride.capitalbikeshare.com/system-data

3. National Oceanic and Atmospheric Administration. (2024). Climate Data Online: Washington, D.C. Weather Data (February–April 2024). Retrieved from https://www.ncdc.noaa.gov/cdo-web/

4. Pandas Development Team. (2024). pandas: Python Data Analysis Library. Available at https://pandas.pydata.org/

5. Scikit-learn Developers. (2024). scikit-learn: Machine Learning in Python. Retrieved from https://scikit-learn.org/

6. Matplotlib Development Team. (2024). Matplotlib: Visualization with Python. https://matplotlib.org/

7. OpenAI (ChatGPT), to streamline and edit the document. Retrieved from: https://www.chatgpt.com