IE 434 - Deep Learning and its Applications
Final Project Report

Project Title: NYC Citi Bike Rentals Prediction Model

Team Members:
Amarthya Kuchana - kuchana2
Kibae Kim - kibaek2
Nithin Balaji - ns49
Safin Akash - santon21
Surya Vasanth - vasanth4

Problem Statement: The project aims to predict the demand for Lyft Bikes on a specific day. By forecasting customer bike-riding patterns, we plan to classify the demand for Lyft Bikes into High, Medium, and Low categories.

Data about NYC Citi Bike rentals at: https://ride.citibikenyc.com/system-data
Index of bucket "tripdata".

Label(s):

- Demand

Features:

- Start latitude
- Start longitude
- Minimum Temperature
- Maximum Temperature
- Months
- Day of the week
- Precipitation
- Snow

Project Google Drive Link:

https://drive.google.com/drive/folders/1IEKS0vg8cTmFycVqdvVfgfl5xgijP8JW?usp=drive_link

Deep Dive 2(Milestone 1):

- **Google Drive Setup:** Established a Google Drive folder for the project, granting access to TAs, graders, and Professor Sowers. This folder serves as the central repository for all project-related documents and datasets.

- **Data Acquisition and Extraction:** Successfully downloaded NYC Citi Bike rentals data. The notebook 'Data Extraction.ipynb' demonstrates the extraction of this data from ZIP files, indicating initial steps in data handling and preparation.

- **README File Creation:** Created a README file in Google Drive, listing team members, detailing the problem statement, and providing an overview of the project approach.

- **Dataset Preparation:** Developed two datasets - a smaller 'debugging' dataset for rapid code testing, and a larger 'working' dataset for training models. Both datasets are converted to pandas DataFrames for ease of manipulation.

- **Data Processing and Storage:** Utilized the pickle module for efficient data storage and retrieval, ensuring quick access and correct data type preservation. This includes the conversion of date and time fields to pandas timestamps for better time manipulation.

- **Preliminary Data Analysis:** The 'Data Extraction.ipynb' notebook outlines initial data analysis steps, including data consolidation and basic exploration.

Deep Dive 3(Milestone 2):

The objectives for Milestone 2 were twofold:

Data Exploration: This involved visualizing the data, providing descriptive statistics, and identifying any biases in labels, as well as discussing issues related to missing, imbalanced, or sparse data.

Baseline Learning: We were tasked with developing a baseline model using linear or logistic regression, including thorough documentation of the process in the notebook.

Data Exploration:

- **Visualization and Statistical Analysis:** Our team effectively visualized key aspects of the dataset and computed descriptive statistics to gain insights into the data's patterns and characteristics, such as ride durations and frequency.

- **Addressing Missing Data:** We identified and meticulously handled missing data, opting to remove rows with null values. Our analysis included a discussion on the potential biases introduced by this decision, demonstrating our critical approach to data integrity.

- **Integration and Cleaning of Weather Data:** Successfully merged weather data into our main dataset, showcasing our ability to handle and clean multiple data sources to enrich our model's predictive accuracy.

- **Demand Distribution Analysis:** A significant part of our exploration involved understanding the distribution of bike rental demand. By grouping data by start station and date, we analyzed how demand varied across different locations and times. This helped us identify patterns and trends that could be crucial for predicting bike rental demand.

- **Enhanced Data Grouping and Feature Engineering**: Building on the initial exploration, we loaded additional datasets, including station coordinates, to enrich our analysis. We performed advanced grouping by key features such as start station latitude and longitude, date, and various weather conditions. This approach allowed us to explore the daily demand patterns for bike rentals at different stations.

Baseline Learning :

In the development of our baseline model, the team focused on structuring the dataset to be conducive for logistic regression analysis. Firstly, we converted categorical variables into a format suitable for logistic regression using dummy encoding. This transformation was crucial to ensure that these variables could effectively contribute to the predictive power of the model. The dataset was divided into training and testing sets. This division is to validate the model's performance on unseen data and to prevent overfitting.

We chose **logistic regression** for our baseline model due to its simplicity and efficiency in binary classification tasks. Logistic regression was selected for its suitability in predicting binary outcomes, aligning well with our project's requirements of classifying bike rental demand into categories. The confusion matrix provides a comprehensive overview of how well the model classifies data points into different categories. The confusion matrix allowed us to calculate the accuracy of the model by evaluating the ratio of correctly predicted instances to the total number of instances.

Confusion Matrix:

Actual \ Predicted	Low Prediction	Mid Prediction	High Prediction
Actual Low	785	35	1433
Actual Mid	55	882	1190
Actual High	445	431	3888

The accuracy obtained was 60.7 %. Even though it is better than random guessing, there is a need for accuracy improvement and therefore, we planned to work on neural networks.

```
[ ]  import numpy as np

     accuracy = np.trace(conf_matrix) / np.sum(conf_matrix)
     print("Accuracy:", accuracy)
```

```
Accuracy: 0.6075021872265967
```

Summary:

Throughout Milestone 2, our team has successfully navigated the complexities of data analysis and baseline model development. In the "Data Exploration" notebook, we delved deep into the dataset, employing a variety of visualization techniques and statistical analyses to uncover key patterns and insights. We addressed the challenges of missing data, ensuring the integrity of our analysis. Our efforts in integrating and cleansing additional weather data demonstrated our ability to enhance the dataset's richness and relevance. The baseline learning phase was a critical component of our project, establishing a benchmark for future model comparisons. The logistic regression model, with its simplicity and interpretability, served as an effective baseline, providing valuable insights into the factors influencing bike rental demand.

Deep Dive 4(Milestone 3):

Deep Learning Model Construction and Refinement:

The first step in our deep learning endeavor was to prepare our dataset for the model. This process involved loading our preprocessed data, transforming our categorical labels into a numerical format compatible with deep learning models. We opted for PyTorch as our deep learning framework, given its flexibility and efficiency in handling complex models like ours.

We chose the **Gated Recurrent Unit (GRU)** model for its proficiency in managing sequential data, a crucial aspect given the time-series nature of our bike rental data. Understanding that deep learning models often benefit from architectural enhancements, we introduced an advanced version of the GRU model. This ImprovedGRUModel incorporated additional hidden layers and implemented dropout techniques and batch normalization to combat overfitting and improve generalization.

Training the model was an intricate process. We split our dataset into training and validation sets, ensuring we had a robust means to evaluate our model's performance. Mini-batch learning was a focal point of our training strategy. By experimenting with different batch sizes, we sought to find a balance between computational efficiency and the model's learning effectiveness. This approach allowed us to optimize the learning process, making it more adaptable to the varying scales of data.

Comparison of optimizers based on the performance:

Another area of investigation was the selection of optimizers. We tested different optimizers, including **Adam and SGD**, to understand their influence on the model's learning curve and overall performance. This experimentation was crucial as the choice of optimizer can significantly affect the speed and stability of the model's training phase. Here is an analysis of the performance of these optimizers in our deep learning model:

Results:

a) For Learning rate of 0.001:

epoch	**Adam Accuracy**	**SGD Accuracy**
100	**72%**	**67.9%**
300	**75%**	**54.0%**
500	**76%**	**55%**
600	**77.3%**	**-**

b) For Learning rate of 0.01:

epoch	**Adam Accuracy**	**SGD Accuracy**
100	**66.6%**	**55.3%**
300	**65.1%**	**56.8%**
500	**65.1%**	**45.1%**

Analysis of Adam Optimizer:

Performance at Learning Rate 0.001:

At a learning rate of 0.001, Adam shows a consistent improvement in accuracy with an increase in epochs: 72% (100 epochs), 75% (300 epochs), 76% (500 epochs), and 77.3% (600 epochs). This indicates that the model benefits from longer training periods when using Adam with this learning rate, likely due to effective adaptation of the learning rate for each parameter.

Performance at Learning Rate 0.01:

With a higher learning rate of 0.01, the accuracy peaks at 66.6% for 100 epochs but then declines slightly to 65.1% for both 300 and 500 epochs. This suggests that a higher learning rate might lead to overshooting the optimal points during training, which prevents further improvement in accuracy beyond a certain point.

Analysis of SGD Optimizer:

Performance at Learning Rate 0.001:

For SGD with a learning rate of 0.001, there is an initial increase in accuracy to 67.9% at 100 epochs but a subsequent decline to 54.0% and 55.0% for 300 and 500 epochs, respectively. This could indicate that the learning rate is too low for SGD, causing the optimizer to get stuck in local minima or converge too slowly.

Performance at Learning Rate 0.01:

At a learning rate of 0.01, the accuracy initially is 55.3% at 100 epochs and slightly improves to 56.8% at 300 epochs but then significantly drops to 45.1% at 500 epochs. This pattern might suggest that while the learning rate is more suitable for faster convergence, the model may be overfitting with increased training, leading to poorer performance on the test set.

Conclusion:

The Adam optimizer outperforms SGD in this context, achieving higher accuracy across different epochs and learning rates. Its adaptive learning rate mechanism appears to be more effective for the dataset and model architecture we are using.

These results highlight the importance of carefully selecting and tuning the optimizer and its parameters based on the specific characteristics of the data and the model.
Based on our model's performance, continuing with **Adam at a learning rate of 0.001 and experimenting with epochs around 300-600 seems** to be the best approach.

Hyperparameter Tuning:

We adjusted parameters like hidden size, number of layers, dropout rate, and learning rate. This tuning process was pivotal in enhancing the model's predictive capability.

- We have set the number of hidden units to 50. Increasing the number of hidden units to 50 allows your neural network to capture more complex patterns in the data. A larger hidden size can enable the model to learn a richer representation of the input features, which can be particularly beneficial in capturing the nuances of bike rental demand patterns.

- Using two layers in your neural network makes it deeper, which can enhance its ability to model complex relationships in the data. Deep networks can capture hierarchical representations, which might be crucial given the time-series nature of the bike rental data.

- Setting a dropout rate of 0.2 helps prevent overfitting by randomly dropping 20% of the neurons in the layers during training. This ensures that the network does not become overly reliant on any specific neuron and can generalize better to unseen data.

- The output size of 3 suggests that your model is likely performing a classification task with three possible classes. This aligns with the nature of the prediction problem, presumably categorizing bike rental demand into three distinct categories(High, Medium and Low Demand).

- A batch size of 64 is a common choice in training neural networks. This size is generally a good balance between the model's training speed and its ability to converge to a good solution.

Summary:

Our team has successfully navigated the challenges of building and fine-tuning a GRU-based deep learning model. The thorough investigation into mini-batch learning, optimizer effects, and

hyperparameter tuning has not only augmented our model's accuracy but also enriched our understanding of deep learning as a powerful tool in predictive analytics.

Deep Dive 5(Milestone 4):

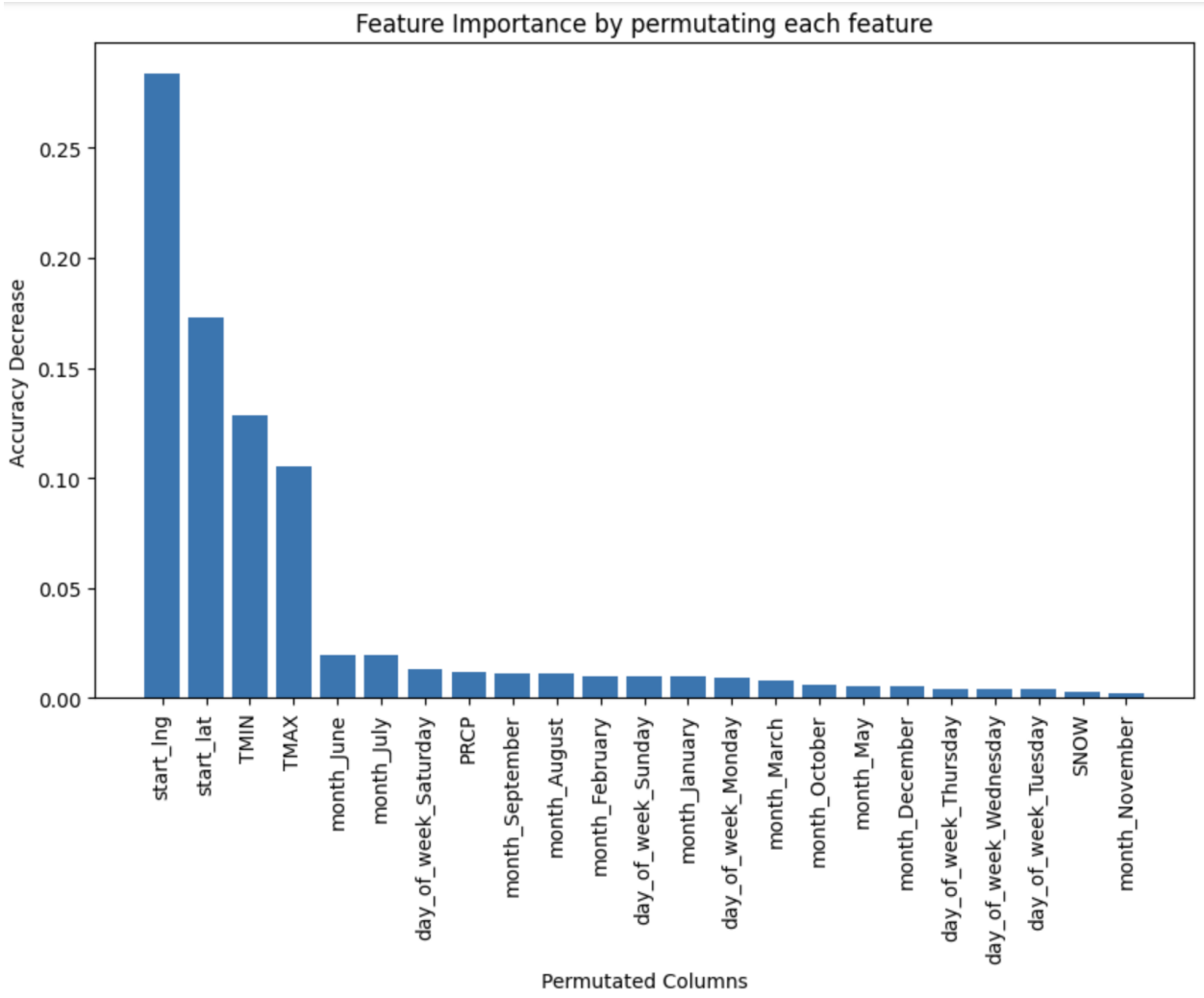
Milestone 4 of our project focused on the critical task of assessing the feature importance for our deep learning model. This phase involved a detailed investigation into how each feature in the dataset influences the model's predictive accuracy. Our methodology centered around permutation feature importance, a powerful technique for understanding feature relevance in a model-agnostic way.

Feature Importance:

The process commenced with the preparation of our test dataset and the deployment of our ImprovedGRUModel, a GRU-based neural network meticulously developed for this study. The test data was transformed into PyTorch tensors, aligning with our deep learning framework's requirements. We then embarked on a systematic evaluation where each feature of the dataset was individually permuted. This permutation involved randomly shuffling the values within a single feature column across the dataset, thereby disrupting the structure and information that the feature provides to the model.

By assessing the model’s accuracy with each feature permutation and comparing it to the original accuracy, we could gauge the impact of each feature on the model’s predictive capability. A significant drop in accuracy upon permuting a feature indicated its importance to the model's predictions. This process was repeated multiple times for each feature to ensure statistical robustness, mitigating the randomness inherent in the permutation process. The results from these iterations were averaged to derive a more reliable and consistent estimate of each feature's importance.

Our analysis revealed that certain features in the dataset had a profound impact on the model’s accuracy. These features, which caused the most substantial decrease in accuracy upon permutation, were identified as the most crucial in determining bike rental demand.



From the chart, it's clear that the feature 'start_lng' - the longitude where a bike rental starts has the highest impact on the model's performance. When the values of 'start_lng' are shuffled, there is a significant decrease in accuracy, indicating that the model relies heavily on this feature to make predictions.

The 'start_lat', or the latitude of the rental start, also shows a substantial impact, though less than 'start_lng', reinforcing the importance of location in the model's predictions. Other features such as 'TMIN', 'TMAX', and the month of June also contribute to model accuracy but to a lesser extent.

The insights gained from this analysis are instrumental in understanding the dynamics influencing bike rental demand and will be pivotal in advancing the predictive accuracy and applicability of your model.

Conclusion:

As we conclude our project, our team has successfully developed a predictive model for NYC Citi Bike Rentals. This model, crafted through the application of deep learning techniques, stands as a predictive tool capable of classifying the expected demand for bikes at any given station and time. The demand is classified as Low, Medium and High.

Our journey began with the integration of diverse data sets, capturing variables such as geographical coordinates, temporal elements, and weather conditions. The model's ability to interpret the significance of starting longitude and latitude demonstrates the critical role of location in demand prediction. The day of the week and month of the year further refine the model's predictions, accounting for the natural ebb and flow of urban life. Meanwhile, environmental factors like temperature, precipitation, and snow introduce a level of nuance, acknowledging the impact of weather on urban transportation choices.

For instance, by inputting specific data points such as a December Sunday at Astor Place, with no precipitation or snow, our model can forecast whether the demand for bikes will be low, medium, or high. Such predictions are not mere estimations; they are the result of a finely tuned algorithm that has learned from extensive data to recognize and anticipate patterns.

By providing a means to anticipate demand, our model offers valuable insights for operational planning and resource allocation within the urban bike-sharing ecosystem. It is a step toward more intelligent and responsive urban transportation systems, where efficiency and user satisfaction are continually enhanced.

1. **Optimized Fleet Management**: By accurately predicting bike demand, companies can efficiently manage their fleet. For instance, during periods of high demand, additional bikes can be allocated to the station, and during low demand, the excess bikes can be redistributed to other locations. This optimization reduces operational costs and increases customer satisfaction by ensuring availability when needed.

2. **Dynamic Pricing Strategy**: Companies could use demand predictions to implement dynamic pricing strategies. During peak demand times, prices could be slightly increased to manage demand and optimize revenue. Conversely, lower prices during off-peak times could attract more users, balancing overall usage and maintaining consistent revenue streams.

3. **Maintenance and Staffing Schedules**: Predicting demand allows for better planning of maintenance and staffing. High-demand periods might require more staff for customer service

and bike maintenance, while lower-demand periods could see reduced staffing, optimizing labor costs. Regular maintenance can also be scheduled during predicted low-demand periods, minimizing service disruption.

4. **Targeted Marketing and Promotions**: Knowledge of demand patterns allows for more effective marketing strategies. Promotions and advertisements can be targeted for periods of lower demand to boost usage, or in areas where the data shows potential growth in user base.

5. **Urban Planning and Infrastructure Development**: This model could be valuable to city planners and local governments for informed decision-making regarding urban infrastructure. Knowing where and when bike demand is highest can guide the development of new bike lanes, parking spaces, and even influence broader transportation policy decisions.