Module 6 Final Project

Course: **ALY6020: Predictive Analytics**

CPS Fall Quarter Term A

Instructor: Nina Rajabi Nasab

Prepared By: Gunjan Paladiya
Larissa Anoh
Sachit Gopal

Assignment Completion Date: October 29sth, 2023

I. Abstract

This analysis addresses the challenge of predicting congestion at Denver International Airport (DEN). It involves two scenarios: one with a continuous target variable "diff" and the other with a categorical target variable "status." The methodology includes extensive data preprocessing and the evaluation of machine learning models. Scenario 1 suggests that Random Forest is the best model for predicting "diff," influenced by temperature and day. In Scenario 2, Random Forest leads in predicting "status" but requires improvements in capturing congestion. Comparing PyCaret and traditional modeling reveals metric discrepancies.

II. Introduction

The goal of this analysis is to create two separate scenarios in response to the airline manager's request to improve situational awareness and forecast congestion at the Denver airport. The continuous variable "diff," which stands for the hourly difference between scheduled and actual flight volumes, is the target variable in the first scenario. In the second case, "diff" is converted to a categorical variable that represents "normal" or "congested" airport conditions. The analytical method entails thorough data cleansing, which includes handling missing values, treating outliers, removing superfluous columns, and addressing multicollinearity. In addition, features are created, and transformations are applied as needed to get the dataset ready for model building.

Several machine learning models, such as SVM, Decision Tree, Random Forest, Gradient Boosting, and Linear Regression, are used in Part 2 of the analysis to forecast the continuous variable "diff." Following a comparison of these models' performances, the accuracy and salient aspects of the most appropriate model are suggested. In scenario 2, where the categorical variable "status" is formed from "diff," Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, and Neural Network models are employed, Part 3 concentrates on the situation. To choose the best model for scenario 2, the recall rate, precision, and overall accuracy of the models are evaluated. A supplemental section (Part 4) examines the PyCaret library to see if its outcomes match those of conventional machine learning techniques. With the use of this comprehensive study, the airline manager should be able to anticipate and successfully manage airport congestion.

III. Methodology

Part 1: Data preprocessing involves handling outliers, missing values, and multicollinearity.

Part 2: Scenario 1 utilizes five models to predict "diff." Random Forest is recommended, with temperature and day as influential features.

Part 3: Scenario 2 predicts "status" with five models, favoring Random Forest. Key features include temperature and day.

Part 4: A comparison with PyCaret reveals metric differences, emphasizing the need for validation.

IV. Analysis

Part 1: Data Cleansing

We have eliminated the "airport" column since this data solely applies to Denver Airport [1]. The redundant "airport" column was removed from the dataset, leaving behind other relevant columns such as date, hour, scheduled and actual flight volumes, temperature, wind speed, wind gust, visibility, ceiling, and precipitation data. This action streamlines the dataset by eliminating unnecessary information [2]. Aside from this, the IQR has been used to handle the outliers. Values falling below the lower bound have been transferred to the lower bound value, while values surpassing the upper bound have been transferred to the upper bound value [3][4]. The number of missing values in the dataset was determined in the following phase, and we discovered that there are missing values in 6 columns [5]. To handle them, we utilized the "mean" of each column. However, as the temperature column contains a small number of negative values, changing its null value using the mean would not be a wise decision, thus we eliminated the null value for that column.

Subsequently, we created a correlation map to assess multicollinearity [6]. The results showed that there was a strong link between Scheduled_volume and Actual_volume, leading us to remove the latter column. Additionally, we used "LableEncoder()" to encode the day, month, and year that we took into consideration when parsing the data from the date column [7].

Part 2: For Scenario 1

Predicting the continuous target variable "diff," which stands for the hourly difference between scheduled and actual flight volumes at the Denver airport, is the goal of Scenario 1. Several machine learning models will be used and evaluated for their prediction power to accomplish this. The models that were selected for assessment include Gradient Boosting, Decision Tree, Random Forest, Support Vector Machine (SVM), and Linear Regression. The goal of this comparative analysis is to determine which model better represents the relationships and patterns found in the data, offering important new information about the variables affecting airport traffic. Providing how well the models perform in handling the given prediction task.

- **Linear Regression [8]:**
  - The results show the performance metrics of the Linear Regression model that was used to forecast the continuous target variable "diff," which stands for the hourly difference between the number of flights that occur at the Denver airport and the number that is scheduled. The average absolute difference between the expected and actual values is indicated by the Mean Absolute Error (MAE) of 3.26. The average squared difference is quantified by the Mean Squared Error (MSE) of 23.41, which highlights bigger errors. A reasonable fit is indicated by the R-squared value of 0.36, which indicates that the model explains around 36% of the variance in the target variable. When taken as a whole, these measures offer information on how well the Linear Regression model forecasts airport traffic and captures underlying patterns in the data.

- **SVM [9]:**
  - The following are the performance metrics for the Support Vector Machine (SVM) model for the congestion prediction task at the Denver airport: 3.69 as the Mean Absolute Error (MAE), 39.52 as the Mean Squared Error (MSE), 0.09 as the R-squared (R2) value, and 6.29 as the Root Mean Squared Error (RMSE). The model's inability to adequately explain the variability in the target variable is suggested by the negative R-squared value, which shows that the model does not fit the data well. Higher values indicate a wider gap between the expected and actual values. The MAE and RMSE values offer insights into the typical size of mistakes in the forecasts. Together, these indicators highlight how crucial it is to continue refining the model or looking into different strategies to increase the predictive accuracy for congestion at the Denver airport.
- **Decision Tree [10]:**
  - The effectiveness of the Decision Tree model is assessed using several measures. The average absolute variances between the expected and actual values are measured by the Mean Absolute Error (MAE), which has a value of 2.86. The average squared differences are quantified by the Mean Squared Error (MSE), which stands at 23.40 and sheds light on the model's capacity to account for data variability. The target variable's share of variance explained by the model is indicated by the R-squared value of 0.36. When the R-squared is lower, it is possible that the model is underestimating the complexity of the underlying patterns. Overall, the evaluation metrics point out areas where the Decision Tree model's accuracy and robustness in predicting the hourly difference between scheduled and actual flight volumes might be improved, even though it offers some degree of predictive capability.
- **Random Forest [11]:**
  - The performance indicators of the Random Forest model offer important information about how well it can predict the continuous target variable "diff," which stands for the hourly difference between the actual and scheduled flight volumes at the Denver airport. A reasonably good match between the model and the data is indicated by the Mean Squared Error (MSE) of 12.29 and the R-squared value of 0.66. The average amount of mistakes is further quantified by the Root Mean Squared Error (RMSE) of 3.51, which indicates that the model's predictions typically differ by 3.51 flights from the actual values. All these measures show how well Random Forest was able to identify trends in the dataset and forecast airport congestion, providing useful information for the efficient handling of air traffic delays.
- **Gradient Boosting [12]:**
  - The average squared difference between the expected and actual values was 13.83, which is the result of the gradient-boosting model's performance. With an R-squared of 0.62, the model appears to have caught roughly 62% of the variability in the target variable. The average size of the errors is indicated by the Root Mean Squared Error (RMSE) of 3.72, which provides a gauge of the accuracy of the model. Together, these measures show how well the model can forecast the continuous goal variable "diff" in the scenario [13]. Even though the R-squared value suggests a respectable degree of explanatory power, more analysis and comparison with alternative models will be necessary to ascertain the best method for forecasting traffic at Denver Airport.

Part 3: For Scenario 2

Predicting the categorical variable "status," which indicates whether the airport is "congested" or "normal" based on the converted "diff" column, becomes the main emphasis of Scenario 2. The aim is to utilize diverse categorization methods and assess their efficacy concerning total accuracy, recall rate, and precision. The models that are being compared are Neural Network, Gradient Boosting, Random Forest, Decision Tree, and Logistic Regression. This investigation aims to determine which model is most suited for forecasting the congestion level at the Denver airport and to identify the key elements that influence the model's forecasting ability. We can give well-informed suggestions for the final model selection in Scenario 2 by evaluating the precision and accuracy of each model.

- **Logistic Regression [14]:**
  - The aim variable was classified as "normal" or "congested," and the effectiveness of the Logistic Regression model was assessed in predicting the level of congestion in an airport. The classification report offers a thorough rundown of the model's functionality. Just 67% of the cases that were projected to be congested were,s according to the comparatively low prediction accuracy of 67%. Nevertheless, the recall for congestion is significantly lower at 24%, indicating that a significant number of real-world crowded situations were missed by the model. Positively, the model showed good recall (97%) and precision (82%) in predicting typical scenarios. The macro-average F1-score is 0.62, indicating a trade-off between recall and precision, despite the overall accuracy of 81%. With 928 false negatives compared to 298 genuine positives, the confusion matrix further demonstrates how frequently the model misclassifies crowded cases. This investigation highlights the need for enhancements in congestion prediction, possibly via feature engineering or alternative model exploration.
- **Decision Tree [15]:**
  - The confusion matrix and classification report shed light on how well the Decision Tree model is working. The model obtained balanced performance for both congested and normal classes in terms of precision, recall, and F1-score. For the crowded class, the precision is 0.48, meaning that 48% of the time the model is right when it predicts congestion. Recall for the congested class is likewise 0.48, indicating that 48% of real congested occurrences were captured by the model. With a weighted average F1-score of 0.77, the model's overall accuracy is 77%. The values (591, 635, 653, 3795) in the confusion matrix stand for true positive, false positive, false negative, and true negative, in that order. Together, these measures show that the model can successfully categorize both normal and congested settings, with the potential for further optimization.
- **Random Forest [16]:**
  - The confusion matrix and classification report offer insightful information on how well the Random Forest model performs. According to the classification report, the model's overall accuracy is 84%; it can detect congested cases with a precision of 71% and typical instances with an accuracy of 86%. Recall, which measures the capacity to recognize instances with accuracy, is 42% in crowded environments and 95% in regular circumstances. The precision-recall ratio, or F1-score, is 53% in congested cases and 90% in typical cases. Looking at the confusion matrix, we can see that the

model accurately recognized 520 (true positives) out of 1226 congested occurrences, but incorrectly classified 706 (false negatives) as normal. The model misclassified 212 as congested (false positives) and properly forecasted 4236 (true negatives) for the usual cases (4448). Together, these indicators show how well the algorithm can differentiate between typical airport conditions and busy ones.

- **Gradient Boosting [17]:**
  - The Gradient Boosting model's output offers insightful information about how well it performs. An extensive assessment of the model's accuracy, recall, and F1-score for every class is provided in the Classification Report. The precision for the "congested" class is 0.72, meaning that 72% of the time the model is right when it forecasts congestion. Nevertheless, the recall of 0.33 for this class indicates that the model has difficulty capturing all cases of congestion. Conversely, the "normal" class shows good recall (0.96) and precision (0.84), demonstrating the model's ability to accurately identify non-congested scenarios. The overall accuracy of 0.83 highlights how well the model predicts the future. With 405 true positive predictions for congestion, 4289 true negatives for typical scenarios, 821 erroneous positives, and 159 false negatives, the Confusion Matrix provides additional insight into the model's performance. Together, these measurements show the trade-offs and advantages of the gradient-boosting approach in differentiating between crowded and typical airport situations.
- **Neural Network [18]:**
  - The Neural Network model's ability to anticipate airport congestion is thoroughly evaluated by the classification report and confusion matrix. According to the classification report, the model accurately predicts congestion when it comes to the "congested" class, as seen by the exceptionally high precision of 1.00 for this class. The recall for the "congested" class, however, is noticeably low at 0.00, indicating that the model has difficulty detecting actual cases of congestion. The confusion matrix illustrates this disparity, with only 2 out of 1226 real congested instances correctly recognized and 4448 out of 4448 correctly identified typical instances. With an overall accuracy of 78%, the model must become more sensitive to instances of congestion. These results demonstrate how well the model recognizes normal conditions. Still, they also emphasize how crucial it is to correct the imbalance and improve the model's capacity to identify congestion to produce forecasts that are more reliable.

Part 4 (Bonus)

Based on our results between PyCaret and the traditional modeling approach for both Scenario 1 (Regression) and Scenario 2 (Classification) we realized there are differences in the specific metrics reported by PyCaret and the traditional modeling approach for both Scenario 1 and Scenario 2. These differences could be attributed to variations in how the two approaches handle data preprocessing, feature engineering, and model hyperparameter tuning. [13] [14] [15] [16] [17] [18] [19] [20]

For Scenario 1 (Regression):

- In the traditional approach, the key metric R-squared (R2) for Linear Regression was approximately 0.3561. [13]

- In PyCaret, the R2 value for the Linear Regression model was reported as 0.4950.[19]

These differences suggest that there might have been variations in data preprocessing, feature scaling, or model settings, which led to divergent results.

For Scenario 2 (Classification):

- In PyCaret, the light gradient boosting machine model also reported high accuracy, AUC and F1 but with some differences in precision and recall. [20]

Again, the variations in precision and recall could be due to differences in preprocessing or default settings in PyCaret.

The key takeaway here is that while PyCaret aims to streamline the modeling process and provide an easy way to compare multiple models, it may not always produce identical results to a traditional, manual modeling approach. Differences can arise due to various factors, including parameter settings, preprocessing, and specific metrics used for evaluation.

It's important to note that PyCaret is a valuable tool for quickly exploring and comparing models, especially for those who are new to machine learning. However, for critical applications, it's essential to thoroughly understand the settings and parameters used by PyCaret and validate results against a more traditional modeling approach to ensure the chosen model is reliable and suitable for the specific task.

V. Conclusion

This analysis addresses the critical challenge of predicting congestion at Denver International Airport. Two distinct scenarios were explored: one involving a continuous target variable, "diff," and the other with a categorical target variable, "status." The methodology encompassed extensive data preprocessing, feature engineering, and model evaluation.

In Scenario 1, Random Forest emerged as the best-performing model for predicting the continuous variable "diff." Temperature and day of the week were identified as influential factors affecting congestion. Scenario 2, where "diff" was transformed into the categorical variable "status," also favored the Random Forest model. However, there is room for improvement in capturing congestion cases.

A comparison with PyCaret demonstrated some differences in the reported metrics, emphasizing the need for thorough validation when using automated machine learning tools.

***Recommendations for Future Studies:

1. Feature Engineering: Explore advanced feature engineering techniques to extract more relevant information from the dataset. Consider interactions between variables and the creation of new features that might improve model performance.

2. Imbalanced Data: Address the class imbalance issue in Scenario 2. Explore techniques like oversampling, undersampling, or the use of different classification algorithms to improve the model's ability to detect congestion cases.

3. Hyperparameter Tuning: Fine-tune hyperparameters for the selected models to achieve better predictive accuracy. Grid search and randomized search can be effective tools for this purpose.

4. Alternative Models: Investigate the performance of advanced machine learning models or ensemble methods that could provide even better predictions for both scenarios.

5. External Data Sources: Consider incorporating external data sources, such as real-time weather data or historical flight data, to enhance predictive capabilities.

6. Interpretability: Focus on the interpretability of models to understand the factors contributing to congestion better. This can provide valuable insights into airport management.

7. Validation: Continue to validate model performance with real-world data to ensure their reliability in practical applications.

By addressing these recommendations in future studies, we can further enhance the accuracy and applicability of congestion prediction models, ultimately aiding airport managers in optimizing air traffic operations and reducing passenger disruptions.

VI. References

- Python, R. (2023, July 24). *Python AI: How to build a neural network & make predictions*. https://realpython.com/python-ai-neural-network/

- Brownlee, J. (2022, August 23). How to Calculate Precision, Recall, F1, and More for Deep Learning Models. Machine Learning Mastery. https://machinelearningmastery.com/how-to-calculate-precision-recall-f1-and-more-for-deep-learning-models/

- Analytics Vidhya. (2022, September). Dealing with Outliers Using the IQR Method. https://www.analyticsvidhya.com/blog/2022/09/dealing-with-outliers-using-the-iqr-method/
- GeeksforGeeks. (2021, October 29). Handwritten Digit Recognition using Neural Network. GeeksforGeeks.

  https://www.geeksforgeeks.org/handwritten-digit-recognition-using-neural-network/

- DataTechNotes. (2019, October). Accuracy Check in Python - MAE, MSE, RMSE, R-squared. DataTechNotes.

  https://www.datatechnotes.com/2019/10/accuracy-check-in-python-mae-mse-rmse-r.html

- Lewinson, E. (2020, February 8). Creating Benchmark Models the Scikit-Learn Way. Towards Data Science.

  https://towardsdatascience.com/creating-benchmark-models-the-scikit-learn-way-af227f6ea977

VII. Appendix

[1]

	airport	date	hour	Scheduled_volume	Actual_volume	diff	diff1	diff2	diff3	diff4	diff5	temperature	wind_speed	wind_gust	visibility	ceiling	in_1_hour_precip
0	DEN	1/1/2013	8	94	95	-1	9	0	-1	0	-1	-11.7	4.0	NaN	9.00	40000.0	0.00
1	DEN	1/1/2013	9	108	104	4	-1	9	0	-1	0	-13.9	0.0	NaN	9.00	40000.0	0.00
2	DEN	1/1/2013	10	146	145	1	4	-1	9	0	-1	-11.1	4.0	NaN	7.00	6500.0	0.00
3	DEN	1/1/2013	11	131	128	3	1	4	-1	9	0	-12.2	6.0	NaN	7.00	5000.0	0.00
4	DEN	1/1/2013	12	86	88	-2	3	1	4	-1	9	-11.7	9.0	NaN	8.00	14000.0	0.00
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
28396	DEN	4/30/2016	19	62	61	1	1	-3	-5	1	1	1.1	18.0	NaN	0.25	600.0	0.02
28397	DEN	4/30/2016	20	39	40	-1	1	1	-3	-5	1	1.1	16.0	26.0	0.75	400.0	0.03
28398	DEN	4/30/2016	21	20	21	-1	-1	1	1	-3	-5	1.1	20.0	NaN	0.75	600.0	0.02
28399	DEN	4/30/2016	22	19	19	0	-1	-1	1	1	-3	0.6	13.0	NaN	0.50	300.0	0.01
28400	DEN	4/30/2016	23	16	16	0	0	-1	-1	1	1	1.1	17.0	NaN	1.25	600.0	0.01

28401 rows × 17 columns

[2]

	date	hour	Scheduled_volume	Actual_volume	diff	diff1	diff2	diff3	diff4	diff5	temperature	wind_speed	wind_gust	visibility	ceiling	in_1_hour_precip
0	1/1/2013	8	94	95	-1	9	0	-1	0	-1	-11.7	4.0	NaN	9.0	40000.0	0.0
1	1/1/2013	9	108	104	4	-1	9	0	-1	0	-13.9	0.0	NaN	9.0	40000.0	0.0
2	1/1/2013	10	146	145	1	4	-1	9	0	-1	-11.1	4.0	NaN	7.0	6500.0	0.0
3	1/1/2013	11	131	128	3	1	4	-1	9	0	-12.2	6.0	NaN	7.0	5000.0	0.0
4	1/1/2013	12	86	88	-2	3	1	4	-1	9	-11.7	9.0	NaN	8.0	14000.0	0.0

[3]

	date	hour	Scheduled_volume	Actual_volume	diff	diff1	diff2	diff3	diff4	diff5	temperature	wind_speed	wind_gust	visibility	ceiling	in_1_hour_precip
247	1/11/2013	18	129	127	2	3	8	9	9	2	-3.9	10.0	21.0	0.25	400.0	0.02
248	1/11/2013	19	99	89	10	2	3	8	9	9	-3.9	14.0	NaN	5.00	600.0	0.04
252	1/11/2013	23	8	9	-1	-3	-5	-2	10	2	-8.3	13.0	NaN	0.50	1100.0	0.01
253	1/12/2013	0	4	7	-3	-1	-3	-5	-2	10	-10.0	8.0	NaN	6.00	9000.0	0.01
315	1/14/2013	16	85	83	2	-3	2	3	-4	2	-14.4	4.0	NaN	1.25	11000.0	0.01
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
28396	4/30/2016	19	62	61	1	1	-3	-5	1	1	1.1	18.0	NaN	0.25	600.0	0.02
28397	4/30/2016	20	39	40	-1	1	1	-3	-5	1	1.1	16.0	26.0	0.75	400.0	0.03
28398	4/30/2016	21	20	21	-1	-1	1	1	-3	-5	1.1	20.0	NaN	0.75	600.0	0.02
28399	4/30/2016	22	19	19	0	-1	-1	1	1	-3	0.6	13.0	NaN	0.50	300.0	0.01
28400	4/30/2016	23	16	16	0	0	-1	-1	1	1	1.1	17.0	NaN	1.25	600.0	0.01

1259 rows × 16 columns

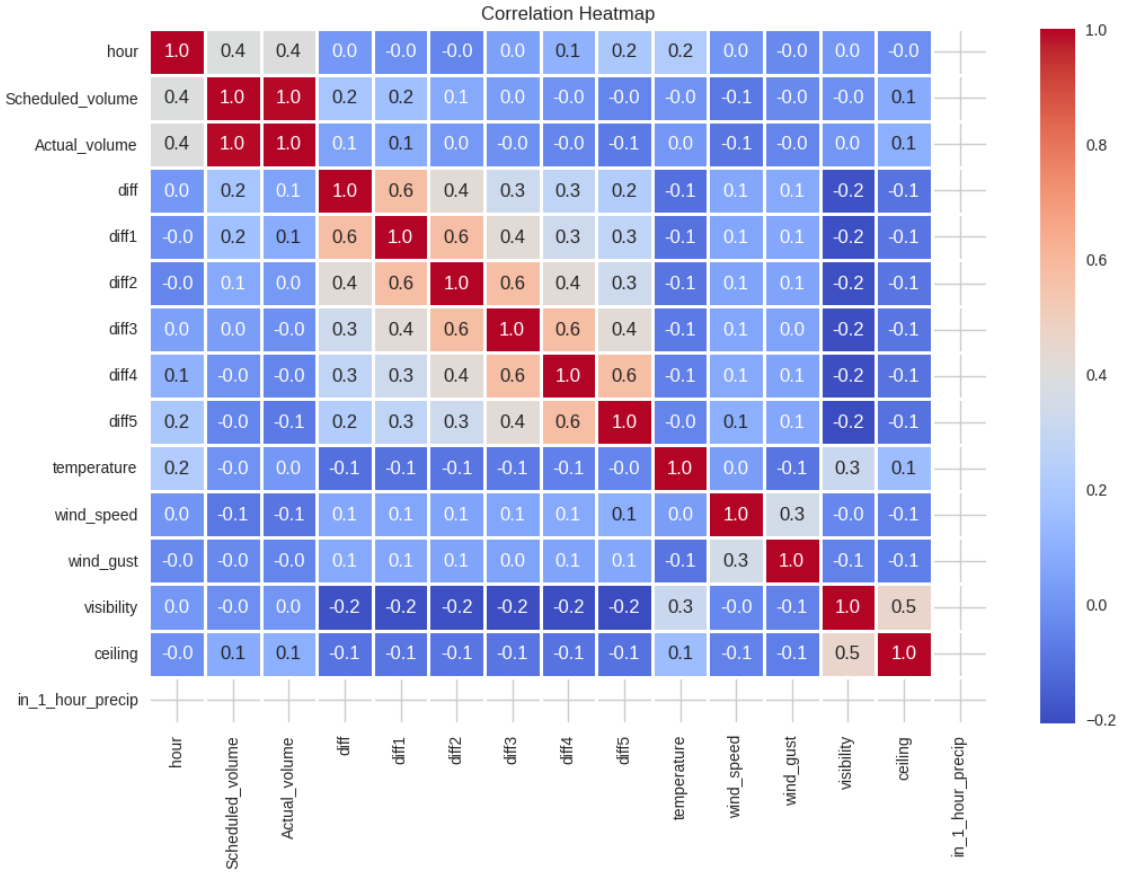
[4]

```
Column: in_1_hour_precip
Number of outliers: 1259
Updated column statistics:
count    28368.0
mean         0.0
std          0.0
min          0.0
25%          0.0
50%          0.0
75%          0.0
max          0.0
Name: in_1_hour_precip, dtype: float64
```

[5]

```
date                  0
hour                  0
Scheduled_volume      0
Actual_volume         0
diff                  0
diff1                 0
diff2                 0
diff3                 0
diff4                 0
diff5                 0
temperature          33
wind_speed           56
wind_gust         25745
visibility           34
ceiling              33
in_1_hour_precip     33
dtype: int64
```

[6]



[7]

	hour	Scheduled_volume	diff	diff1	diff2	diff3	diff4	diff5	temperature	wind_speed	wind_gust	visibility	ceiling	in_1_hour_precip	day	month	year
0	8	94	-1	9	0	-1	0	-1	-11.7	4.0	23.774473	9.00	40000.0	0.0	0	0	0
1	9	108	4	-1	9	0	-1	0	-13.9	0.0	23.774473	9.00	40000.0	0.0	0	0	0
2	10	146	1	4	-1	9	0	-1	-11.1	4.0	23.774473	7.00	6500.0	0.0	0	0	0
3	11	131	3	1	4	-1	9	0	-12.2	6.0	23.774473	7.00	5000.0	0.0	0	0	0
4	12	86	-2	3	1	4	-1	9	-11.7	9.0	23.774473	8.00	14000.0	0.0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
28396	19	62	1	1	-3	-5	1	1	1.1	18.0	23.774473	0.25	600.0	0.0	29	3	3
28397	20	39	-1	1	1	-3	-5	1	1.1	16.0	26.000000	0.75	400.0	0.0	29	3	3
28398	21	20	-1	-1	1	1	-3	-5	1.1	20.0	23.774473	0.75	600.0	0.0	29	3	3
28399	22	19	0	-1	-1	1	1	-3	0.6	13.0	23.774473	0.50	300.0	0.0	29	3	3
28400	23	16	0	0	-1	-1	1	1	1.1	17.0	23.774473	1.25	600.0	0.0	29	3	3

28368 rows × 17 columns

[8]

```
Linear Regression MAE: 3.2652213800893204
Linear Regression MSE: 23.41082406536133
Linear Regression R-squared: 0.35609390748825065
Linear Regression mean squared error: 4.8384733196909675
```

[9]

```
Mean Absolute Error (MAE): 3.6899894254494185
Mean Squared Error (MSE): 39.52044413112443
R-squared (R2): -0.08699525842207745
Root Mean Squared Error (RMSE): 6.286528782334845
```

[10]

```
Decision Tree MAE: 2.87909763835037
Decision Tree MSE: 23.867113147691224
Decision Tree R-squared: 0.343543844353415
Decision Tree mean squared error: 4.885397951824521
```

[11]

```
Random Forest MSE: 12.568842950299612
Random Forest R-squared: 0.6542986044008632
Random Forest RMSE: 3.5452564012070567
Random Forest mean squared error: 3.5452564012070567
```

[12]

```
Gradient Boost MSE: 13.871488376299844
Gradient Boost R-squared: 0.6184698217897814
Gradient Boost RMSE: 3.724444707107335
Gradient Boost mean squared error: 3.724444707107335
```

[13]

	Model	MAE	MSE	RMSE	R-squared (R2)
0	Linear Regression	3.265221	23.410824	4.838473	0.356094
1	SVM	3.689989	39.520444	6.286529	-0.086995
2	Decision Tree	2.879098	23.867113	4.885398	0.343544
3	Random Forest	2.089727	12.568843	3.545256	0.654299
4	Gradient Boost	2.387350	13.871488	3.724445	0.618470

[14]

```
logistic regression Classification Report:
              precision    recall  f1-score   support

   congested       0.67      0.25      0.36      1226
      normal       0.82      0.97      0.89      4448

    accuracy                           0.81      5674
   macro avg       0.75      0.61      0.63      5674
weighted avg       0.79      0.81      0.78      5674


Confusion Matrix:
 [[ 303  923]
 [ 149 4299]]
```

[15]

```
Decision Tree Classification Report:
              precision    recall  f1-score   support

   congested       0.48      0.49      0.49      1226
      normal       0.86      0.85      0.85      4448

    accuracy                           0.77      5674
   macro avg       0.67      0.67      0.67      5674
weighted avg       0.78      0.77      0.78      5674


Confusion Matrix:
 [[ 605  621]
 [ 663 3785]]
```

[16]

```
  Random Forest Classification Report:
              precision    recall  f1-score   support

   congested       0.70      0.42      0.53      1226
      normal       0.86      0.95      0.90      4448

    accuracy                           0.84      5674
   macro avg       0.78      0.69      0.71      5674
weighted avg       0.82      0.84      0.82      5674


Confusion Matrix:
 [[ 518  708]
 [ 219 4229]]
```

[17]

```
Gradient Boosting Classification Report:
              precision    recall  f1-score   support

   congested       0.72      0.33      0.45      1226
      normal       0.84      0.96      0.90      4448

    accuracy                           0.83      5674
   macro avg       0.78      0.65      0.67      5674
weighted avg       0.81      0.83      0.80      5674


Confusion Matrix:
 [[ 405  821]
 [ 159 4289]]
```

[18]

```
NN (MLP) Classification Report:
              precision    recall  f1-score   support

   congested       1.00      0.00      0.00      1226
      normal       0.78      1.00      0.88      4448

    accuracy                           0.78      5674
   macro avg       0.89      0.50      0.44      5674
weighted avg       0.83      0.78      0.69      5674


Confusion Matrix:
 [[   2 1224]
 [   0 4448]]
```

[19]

11	Imputation type	simple
12	Numeric imputation	mean
13	Categorical imputation	mode
14	Maximum one-hot encoding	25
15	Encoding method	None
16	Fold Generator	KFold
17	Fold Number	10
18	CPU Jobs	1
19	Use GPU	False
20	Log Experiment	False
21	Experiment Name	reg-default-name
22	USI	ebdc

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
et	Extra Trees Regressor	1.6700	7.7606	2.7758	0.8003	0.5591	0.6128	7.7840
rf	Random Forest Regressor	1.7117	8.1852	2.8511	0.7902	0.5689	0.6291	15.0080
lightgbm	Light Gradient Boosting Machine	1.7730	8.6779	2.9280	0.7801	0.5809	0.6277	0.5810
xgboost	Extreme Gradient Boosting	1.7756	8.6517	2.9285	0.7773	0.5846	0.6387	0.4100
gbr	Gradient Boosting Regressor	1.9795	9.6125	3.0916	0.7534	0.6260	0.7062	3.0830
dt	Decision Tree Regressor	2.2793	17.7983	4.1955	0.5317	0.7509	0.8532	0.3220
ada	AdaBoost Regressor	3.3150	19.6494	4.3926	0.4951	0.9309	1.3986	1.0070
ridge	Ridge Regression	2.8903	19.8063	4.4408	0.4950	0.8282	1.0001	0.2640
br	Bayesian Ridge	2.8890	19.8063	4.4408	0.4950	0.8279	0.9986	0.4850
lr	Linear Regression	2.8905	19.8063	4.4408	0.4950	0.8283	1.0004	0.2570
lar	Least Angle Regression	3.2867	22.4307	4.6982	0.4236	0.8978	1.3530	0.1450
knn	K Neighbors Regressor	3.0245	23.5421	4.8385	0.4002	0.8194	1.0221	0.7850
en	Elastic Net	3.1776	24.0910	4.9005	0.3841	0.8385	1.0123	0.2420
lasso	Lasso Regression	3.2716	25.1333	5.0056	0.3573	0.8396	1.0626	0.2280
llar	Lasso Least Angle Regression	3.2716	25.1333	5.0056	0.3573	0.8396	1.0626	0.1560
huber	Huber Regressor	3.1584	26.3492	5.1223	0.3293	0.8329	0.9427	1.3280
omp	Orthogonal Matching Pursuit	3.6231	39.4923	6.2566	0.0070	0.8969	0.9722	0.3960
dummy	Dummy Regressor	3.6145	39.8256	6.2820	-0.0007	0.8534	0.9471	0.0950
par	Passive Aggressive Regressor	12.1596	305.4475	14.4807	-8.2316	1.4606	5.5588	0.4680

[20]

	Description	Value
0	Session id	8409
1	Target	status
2	Target type	Binary
3	Target mapping	congested: 0, normal: 1
4	Original data shape	(28368, 17)
5	Transformed data shape	(28368, 17)
6	Transformed train set shape	(19857, 17)
7	Transformed test set shape	(8511, 17)
8	Numeric features	16
9	Preprocess	True
10	Imputation type	simple
11	Numeric imputation	mean
12	Categorical imputation	mode
13	Fold Generator	StratifiedKFold
14	Fold Number	10
15	CPU Jobs	1
16	Use GPU	False
17	Log Experiment	False
18	Experiment Name	clf-default-name
19	USI	fc22

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lightgbm	Light Gradient Boosting Machine	0.8306	0.8622	0.9466	0.8526	0.8971	0.4255	0.4450	0.2730
et	Extra Trees Classifier	0.8286	0.8582	0.9474	0.8501	0.8961	0.4152	0.4361	1.3200
rf	Random Forest Classifier	0.8261	0.8551	0.9466	0.8481	0.8946	0.4049	0.4262	2.1600
gbc	Gradient Boosting Classifier	0.8195	0.8481	0.9589	0.8344	0.8923	0.3503	0.3861	2.4970
ada	Ada Boost Classifier	0.8087	0.8266	0.9541	0.8272	0.8861	0.3070	0.3410	0.7490
lr	Logistic Regression	0.8026	0.7788	0.9637	0.8164	0.8839	0.2527	0.2990	0.2270
ridge	Ridge Classifier	0.8014	0.0000	0.9772	0.8083	0.8848	0.2110	0.2771	0.0270
lda	Linear Discriminant Analysis	0.8010	0.7881	0.9546	0.8199	0.8821	0.2667	0.3028	0.0390
nb	Naive Bayes	0.7816	0.7367	0.9296	0.8160	0.8691	0.2260	0.2450	0.0250
knn	K Neighbors Classifier	0.7809	0.7219	0.9168	0.8227	0.8672	0.2526	0.2654	1.0000
dummy	Dummy Classifier	0.7801	0.5000	1.0000	0.7801	0.8765	0.0000	0.0000	0.0300
dt	Decision Tree Classifier	0.7685	0.6663	0.8488	0.8537	0.8512	0.3300	0.3301	0.1010
svm	SVM - Linear Kernel	0.6782	0.0000	0.7737	0.8300	0.7205	0.1394	0.1792	0.1690
qda	Quadratic Discriminant Analysis	0.4841	0.4734	0.4588	0.7183	0.4728	0.0258	0.0345	0.0440