Data Analytics Project

Cycling traffic in Paris

by
Loris Durbano
Matea Mutz
Sohith Varma Bhupathiraju

Instructor: Tarik Anouar

Table of Contents

1. Data exploration

1.1 Context

The City of Paris has been deploying permanent bicycle counters for several years to assess the development of cycling. The aim of this project is:

1. To carry out an analysis of the collected data by from bike counters
2. To visualize the timetables and the areas of affluence
3. To provide tools to the town hall of Paris so that it can judge the improvements to be made to the various cycling areas of the city.

The original dataset:

```
RangeIndex: 981724 entries, 0 to 981723
Data columns (total 16 columns):
 #   Column                                 Non-Null Count   Dtype
---  ------                                 --------------   -----
 0   Identifiant du compteur                965174 non-null  object
 1   Nom du compteur                        981724 non-null  object
 2   Identifiant du site de comptage        965174 non-null  float64
 3   Nom du site de comptage                965174 non-null  object
 4   Comptage horaire                       981724 non-null  int64
 5   Date et heure de comptage              981724 non-null  object
 6   Date d'installation du site de comptage    965174 non-null  object
 7   Lien vers photo du site de comptage    955220 non-null  object
 8   Coordonnées géographiques              965174 non-null  object
 9   Identifiant technique compteur         953158 non-null  object
 10  ID Photos                              955220 non-null  object
 11  test_lien_vers_photos_du_site_de_comptage_  955220 non-null  object
 12  id_photo_1                             955220 non-null  object
 13  url_sites                              965174 non-null  object
 14  type_dimage                            955220 non-null  object
 15  mois_annee_comptage                    981724 non-null  object
dtypes: float64(1), int64(1), object(14)
memory usage: 119.8+ MB
```

Overview of the descriptive statistics:

	Identifiant du site de comptage	Comptage horaire
count	9.651740e+05	981724.000000
mean	1.334743e+08	75.888616
std	7.461401e+07	104.685238
min	1.000031e+08	0.000000
25%	1.000475e+08	12.000000
50%	1.000560e+08	41.000000
75%	1.000563e+08	94.000000
max	3.000303e+08	8190.000000

- ➔ **981,724 entries**
- ➔ **16 columns**
- ➔ **Between 0 and 28,566 (2.9%) missing values per column**
- ➔ **Mix of temporal & geographical information**

We observe some duplicates & useless data: URL columns, counters technical ID for instance. **The target variable is Comptage horaire / Hourly counting.**

The mean is almost 76 counting/hour. The maximum (8190) seems to be an outlier (to be confirmed). The minimum is 0, we do not have negative outliers. The standard deviation is relatively high compared to the mean, indicating substantial variability in the hourly counts. There are many hours with counts far from the average.

The median (41) is much lower than the mean (75.8886), which suggests that the distribution of hourly counts is skewed. There are more hours with low counts, but a few hours with very high counts pull the mean up.

Based on the quartile information, we could figure out the outliers' thresholds. To process it, we will need to make some changes to the original dataset:

- Translate the column names in English
- Split the counting date (initial format: 2023-04-01T09:00:00+02:00) into counting_year/month/day/hour
- Split the geographical coordinates into 2 columns (latitude & longitude)

Column’s labels after the initial changes:

- **Counter Information:** Counter_ID, Counter_name, Counter_site_ID, Counter_site_name, Counter_technical_ID
- **Target variable**: Hourly_counting
- **Location Information**: Latitude, Longitude
- **Date Information**: Installation_date_of_the_counting_site, Month_year_counting_date, Counting_year, Counting_month, Counting_day, Counting_hour
- **Picture Information**: Picture_URL, Picture_ID, Test_URL_to_counting_site_picture, Picture_1_ID, Counter_picture_URL, Image_format

1.2 Statistical Tests

We conducted three statistical tests:

1. ANOVA test (Hourly counting / Geographical coordinates)
2. ANOVA test (Hourly counting / Counter names)
3. Spearman test (Hourly counting / Counting hour)

1.2.1 ANOVA test (Hourly counting / Geographical coordinates)

- H0: The geographical coordinates do not have any significant influence on hourly counting
- H1: The geographical coordinates have a significant influence on hourly counting

	df	sum_sq	mean_sq	F	PR(>F)
Geographic_coordinates	73.0	2.698854e+09	3.697061e+07	4471.467399	0.0
Residual	965100.0	7.979558e+09	8.268115e+03	NaN	NaN

Since F is very high, we can conclude that changes in geographic coordinates must have an impact on hourly counting. PR(>F) is 0.0. Therefore, we can reject H0.

1.2.2 ANOVA test (Hourly counting / Counter names)

- H0: Counter_name does not have a significant influence on hourly counting
- H1: Counter_name has a significant influence on hourly counting

	df	sum_sq	mean_sq	F	PR(>F)
Counter_name	103.0	2.880170e+09	2.796282e+07	3484.007258	0.0
Residual	981620.0	7.878531e+09	8.026050e+03	NaN	NaN

Since F is very high, we can conclude that changing counters must have an impact on hourly counting. PR(>F) is 0.0. Therefore, we can reject H0.

1.2.3 Spearman test (Hourly counting / Counting hour)

- H0: There is no significant correlation between hourly counting and counting hours
- H1: There is a significant relationship between hourly counting and counting hours

```
Spearman correlation test
Correlation coef: 0.32377936397290663
p-value 0.0
```

The positive Spearman correlation coefficient suggests a (weak) positive relationship between the two variables. The very low p-value (<0.05) indicates that this correlation is statistically significant, implying that it is unlikely to have occurred by chance. Therefore, we can reject H0.

2. Data visualization

There are several important reasons why we need to visualize our data before performing a model. First of all, we need to visualize the data in order to get a graphical representation of the dataset and the most relevant information we can extract from the current data. It helps detect patterns and trends. Then, we need to plot the grapes in order to identify the outliers and anomalies. Data visualization also helps us understand the relationship between different variables and assess the quality of the data. Understanding the data's structure helps us select the best model. At the end, with data visualization, we can better communicate the results to different stakeholders.

2.1 Target variable distribution

Firstly, we will check the distribution of the target variable. The graph used is boxplot. The median number of hourly counting is 41. Since there are a lot of outliers, it's difficult to interpret the boxplot. We would need to manage these outliers to avoid skewing the data.

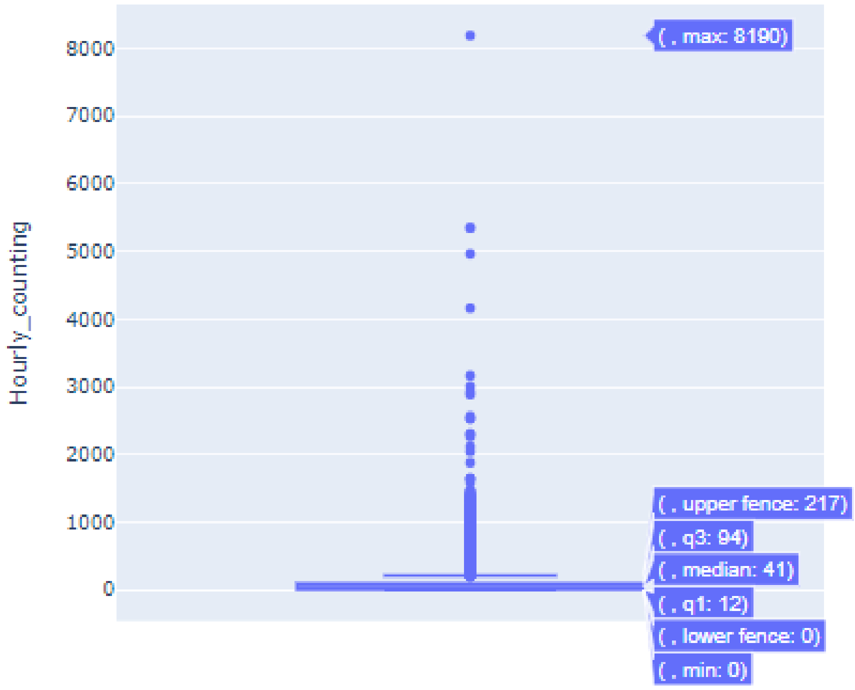
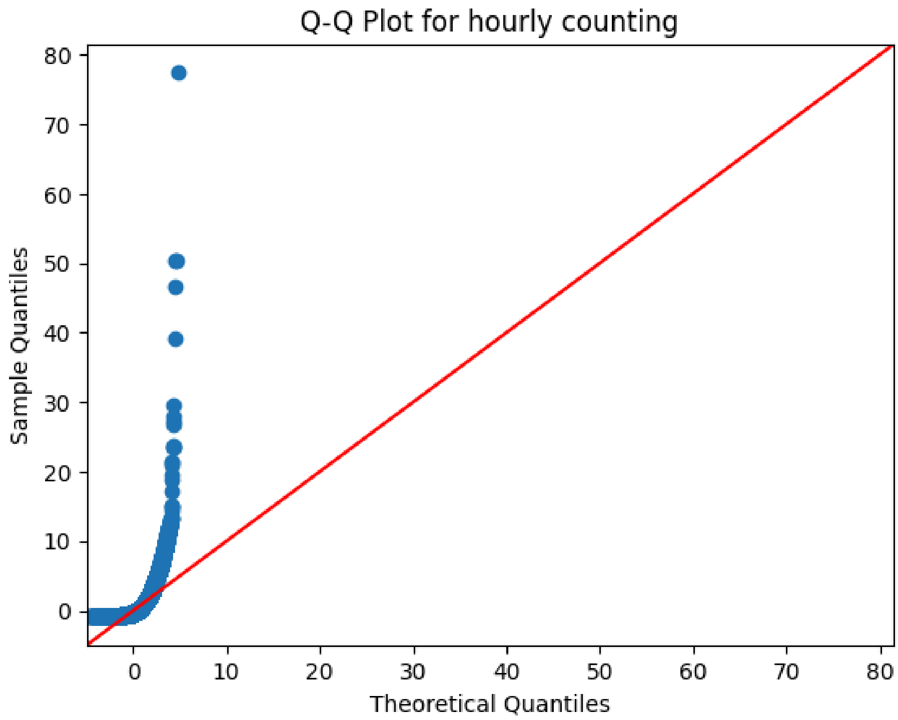


Figure 1: Target variable distribution

To further understand the target variable distribution, we have also created a Q-Q Plot.



The points rise from the lower to higher quantiles, indicating that the data is skewed, as we have seen before (there are many small values and a few large values).

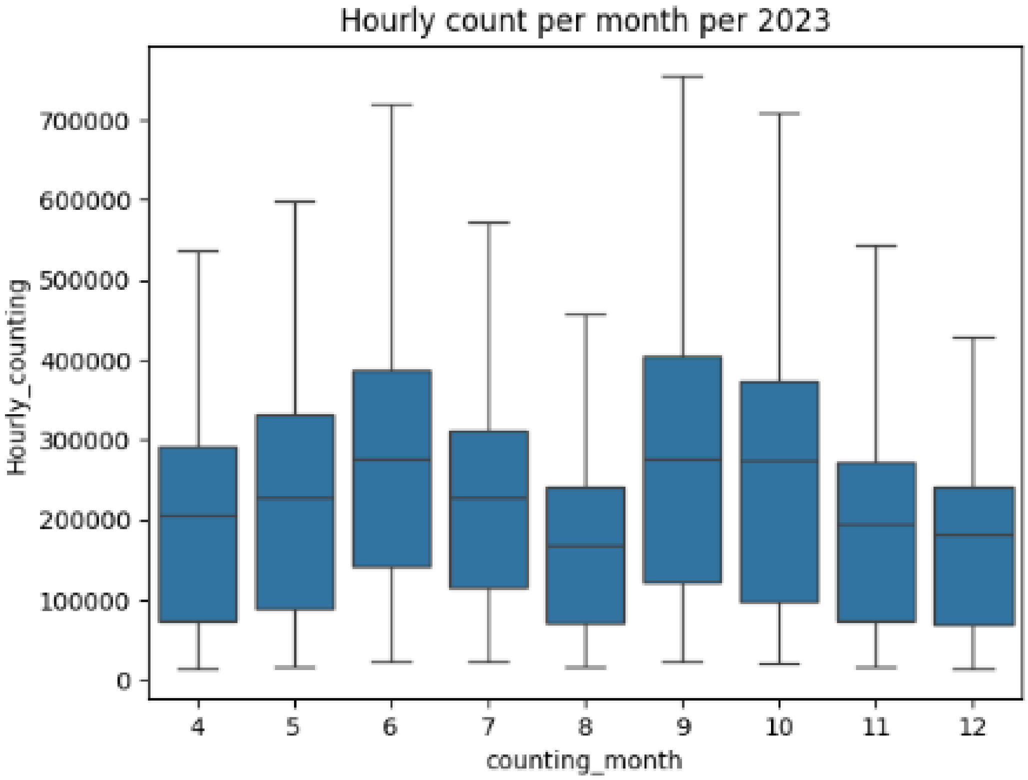
The significant deviation at the upper end shows the outliers we saw in the initial boxplot.

2.1 Target variable evolution

In order to better understand our target variable, we will check different visualizations and their interpretations.

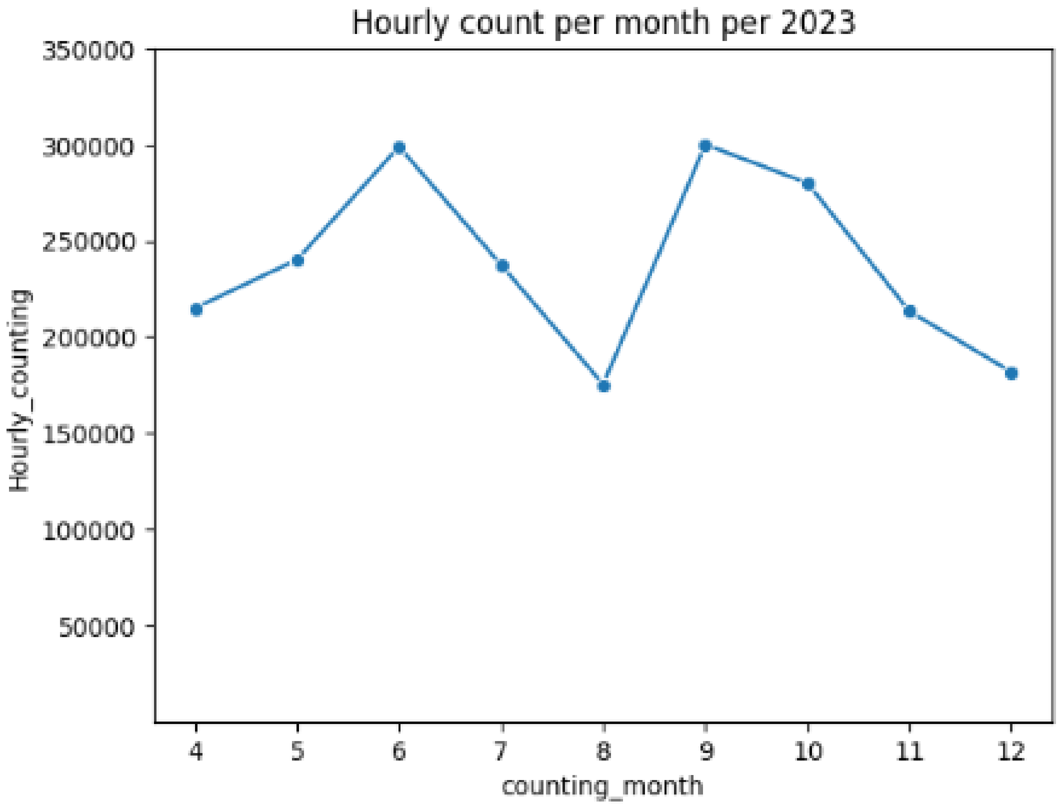
2.2.1 Target variable monthly evolution

In order to get a deeper insight into the target variable monthly distribution, we have grouped the data by "counting_month" and "counting_hour" and put focus only on the year 2023. The reason we chose 2023 is because in that year we have the most complete data.



Every month from April to December, we have between 80 000 and 400 000 bicycle counts. We have a lot of outliers that might indicate that there are some traffic points where we have much more counts than expected. We can see here that the most counts are in June and September. We can observe that weather conditions have an impact on bicycle traffic.

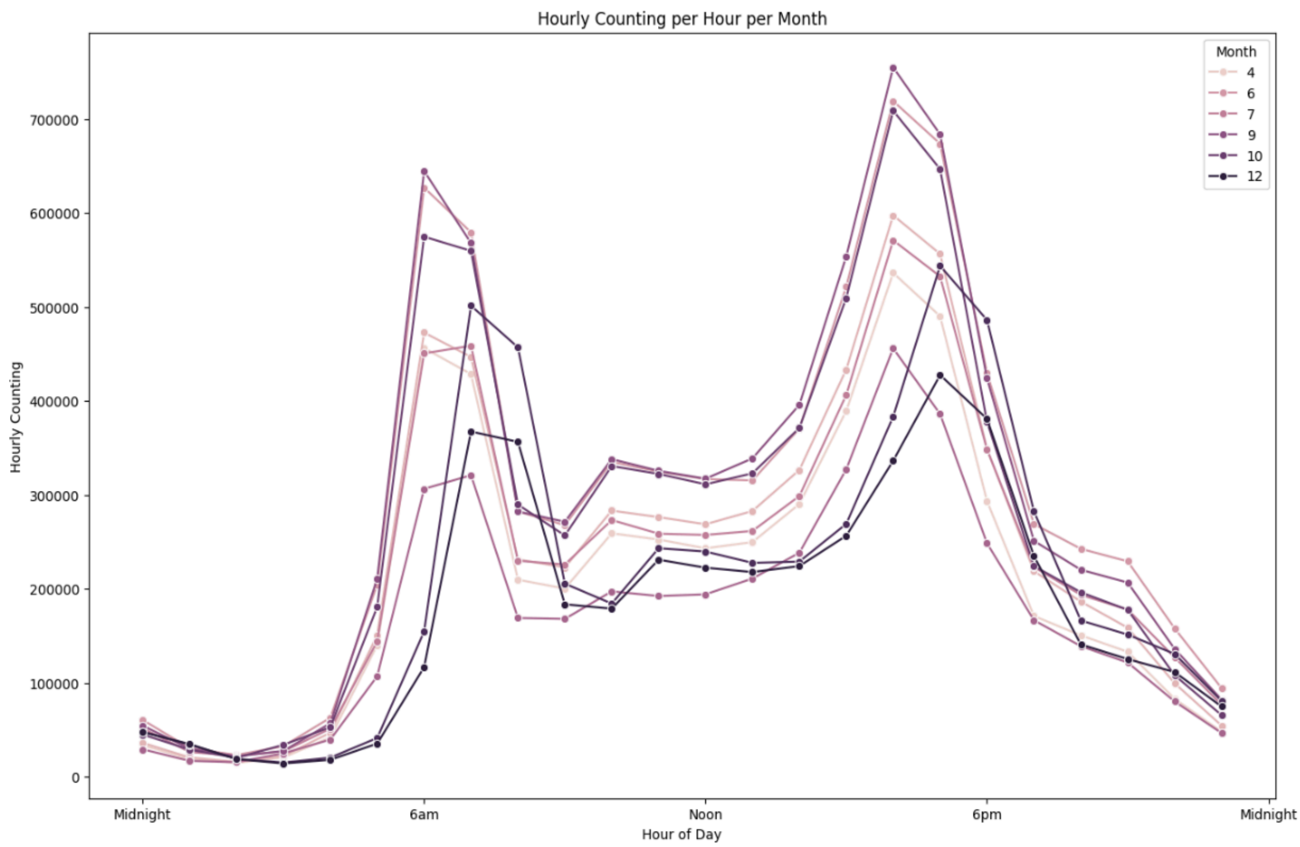
To further see the total nnumberof hourly counts for 2023, we have created a line plot.



This graph shows the hourly count per month in 2023 to see in which months we have the most counts. We can see here that the most counts are in June and September.

2.2.2 Target variable hourly evolution per month

In order to get even more insights into the difference per hourly count per time of the day (hour of day) and the month, we wanted to display the hourly count per month along the day.

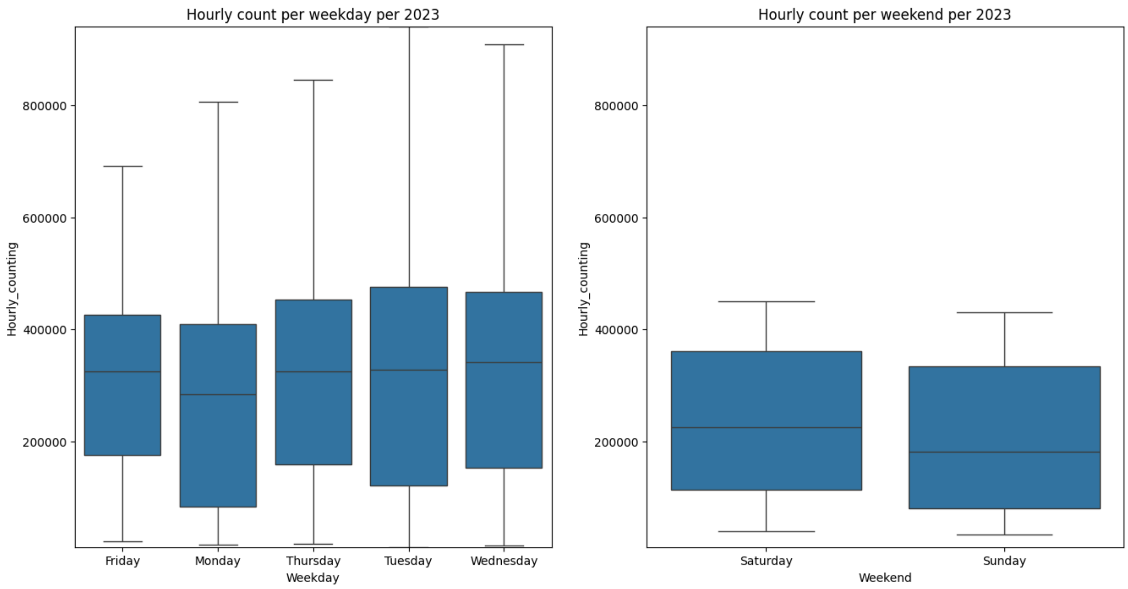


In order to create this graph, we have taken into account only the data for the year 2023 and grouped the data based on "counting_month" and "counting_hour" variables.

We can observe a peak in hourly counting during rush hours (in the morning from 6 to 7 and in the afternoon from 16 to 17). This indicates that a lot of workers / students are using bicycles on a daily basis during working hours. On this graph, we can see that even though the hourly counts are different regarding the counting month, the trend is the same. We have the same peaks and lows for all months, showing the importance of commuters and students in the cycling traffic accross the city.

2.2.3 Target variable weekday evolution

As a next step, we wanted to check if there was any difference in the target variable in case we had a weekday or weekend.

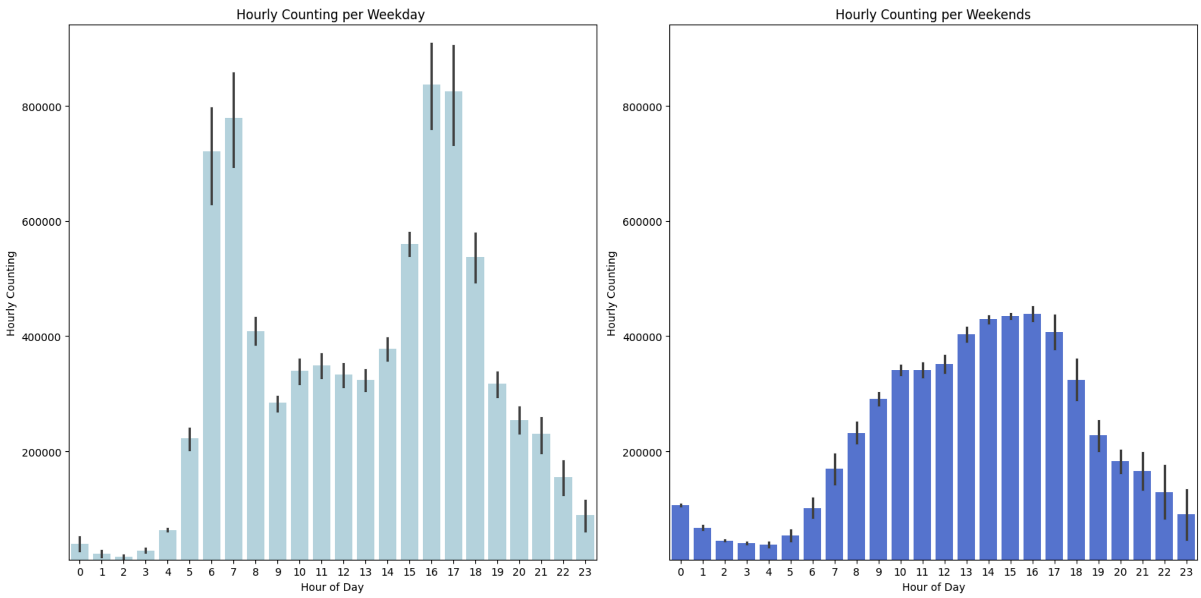


We can observe higher counting during the weekdays; therefore, we could conclude that bicycle movements are mainly done by workers / students. Also, we can identify many more outliers in the case of a working day.

In the following section, we will look deeper into hourly counting per hour per weekday and the weekend to see exactly the difference in hourly counting.

2.2.4 Target variable hourly evolution

As mentioned in the previous section, we also wanted to analyze if there is a difference in hourly count per different time of day regarding if it is a weekend or a weekday.



On this graph, we can clearly see that the peaks are different by hour of day for the weekday (6-7, 16-17) compared to the weekend. While the total hourly count during the peak hours on the weekdays comes close to (or even exceeds) 800,000, on the weekend we don’t have this spike, and the hourly counts don't exceed 450000 counts. The curve is more flatter.

3. Data preprocessing

The three steps were carried out for data preprocessing :

Delete null values: There are approximately 16,550 null values found in the dataset. Since the temporal and geographical values are missing for these entries, we decided to remove them (they represent less than 2% of the total entries).

Through the boxplot, there are many outliers. After careful verification of outlier values, it is possible that, in some event, the frequency of cycles roaming around the city is realistic. So the only outlier with a frequency greater than 8000 is deleted since it is isolated and exceptional for the targeted counter.

Changing the categorical to numerical using encoding. There are many categorical variables which have ordinal data. The factorize method was used for the encoding part. The unnecessary column variables were deleted to keep the data set simple to use for the further machine learning models to implement

Before Preprocessing

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 981724 entries, 0 to 981723
Data columns (total 16 columns):
 #   Column                                   Non-Null Count   Dtype
---  ------                                   --------------   -----
 0   Identifiant du compteur                  965174 non-null  object
 1   Nom du compteur                          981724 non-null  object
 2   Identifiant du site de comptage          965174 non-null  float64
 3   Nom du site de comptage                  965174 non-null  object
 4   Comptage horaire                         981724 non-null  int64
 5   Date et heure de comptage                981724 non-null  object
 6   Date d'installation du site de comptage  965174 non-null  object
 7   Lien vers photo du site de comptage      955220 non-null  object
 8   Coordonnées géographiques                965174 non-null  object
 9   Identifiant technique compteur           953158 non-null  object
 10  ID Photos                                955220 non-null  object
 11  test_lien_vers_photos_du_site_de_comptage_  955220 non-null  object
 12  id_photo_1                               955220 non-null  object
 13  url_sites                                965174 non-null  object
 14  type_dimage                              955220 non-null  object
 15  mois_annee_comptage                      981724 non-null  object
dtypes: float64(1), int64(1), object(14)
memory usage: 119.8+ MB
```

After Preprocessing

```
<class 'pandas.core.frame.DataFrame'>
Index: 965173 entries, 0 to 981723
Data columns (total 16 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   Counter_ID1         965173 non-null  int64
 1   Counter_ID2         965173 non-null  int64
 2   Counter_site_name   965173 non-null  int64
 3   Hourly_counting     965173 non-null  int64
 4   Latitude            965173 non-null  float64
 5   Longitude           965173 non-null  float64
 6   counting_year       965173 non-null  int64
 7   counting_month      965173 non-null  int64
 8   counting_day        965173 non-null  int64
 9   counting_hour       965173 non-null  int64
 10  counting_day_name   965173 non-null  int64
 11  weekday             965173 non-null  int64
 12  weekend             965173 non-null  int64
 13  installation_year   965173 non-null  float64
 14  installation_month  965173 non-null  float64
 15  installation_day    965173 non-null  float64
dtypes: float64(5), int64(11)
memory usage: 125.2 MB
```

4. Modeling

For our modeling, we are going to test 6 models: Linear regression, Decision Tree Regressor, Random Forest Regressor, Lasso, LassoCV and Ridge.

At the end of this chapter, we will give an overview of the performance, determine which model performed the best, and conclude which model we are going to use for the prediction.

4.1 Linear Regression

4.1.1 Introduction and advantages of a model

Our first model that we want to test is Linear Regression. This type of model is used when there is a linear relationship between our target and explanatory variables.

There are several advantages to this model:

- **Good Starting Point:** Linear Regression models are a good place to start and then see, if more complex models are needed
- **Ease of implementation :** Linear Regression models are easy to implement and interpret
- **Ease of use:** They are not very complicated to use
- **Applicable:** They are applicable in many cases
- **Efficiency:** Very fast to train.

Linear regression works best when the relationship between variables is straight and clear-cut, but it can struggle with messy data, like outliers or when variables interact in complicated ways. It does not handle non-linear relationships well and may overfit when the number of predictors is large relative to the number of observations.

After training our model, we calculated the intercept and the coefficients of each variable estimated by the model. The estimated coefficient β1 quantifies the relationship between the target variable and the explanatory variable. The constant β0 (intercept) captures all the information not explained by x. Therefore, the intercept is an important parameter in a linear regression model, providing insight into the expected outcome when all predictors are zero.

	Estimated value
Intercept	64.961192
weekday	32.079506
installation_year	28.417564
counting_hour	17.534177
Latitude	12.308821
Longitude	3.626612
counting_day	-0.013624
Counter_site_name	-0.266047
counting_day_name	-0.411895
counting_month	-1.421986
counting_year	-5.486686
installation_day	-5.902878
Counter_ID2	-11.185420
installation_month	-20.515514
Counter_ID1	-29.869004

In our case, if all other parameters were 0, we would have 64 hours counted. When explanatory variables increase by one unit, then the target variable decreases on average by 29.87 units.

4.1.2 Performance metrics

Mean Absolute Error (MAE):

Train set: 63.782944743970766
Test set: 63.85670363268088

Here we see that the model has a similar performance (error) on the train and test data.

Mean Squared Error (MSE):

Train set: 9760.854473172947
Test set: 9841.313150249058

Here we have very high values, which indicates that there are a lot of errors, even though the model predicts well on both train and test data.

Root Mean Squared Error (RMSE):

Train set: 98.79703676311829
Test set: 99.20339283637963

The model performs similarly on the train and test data sets.

Residual Standard Error (RSE):

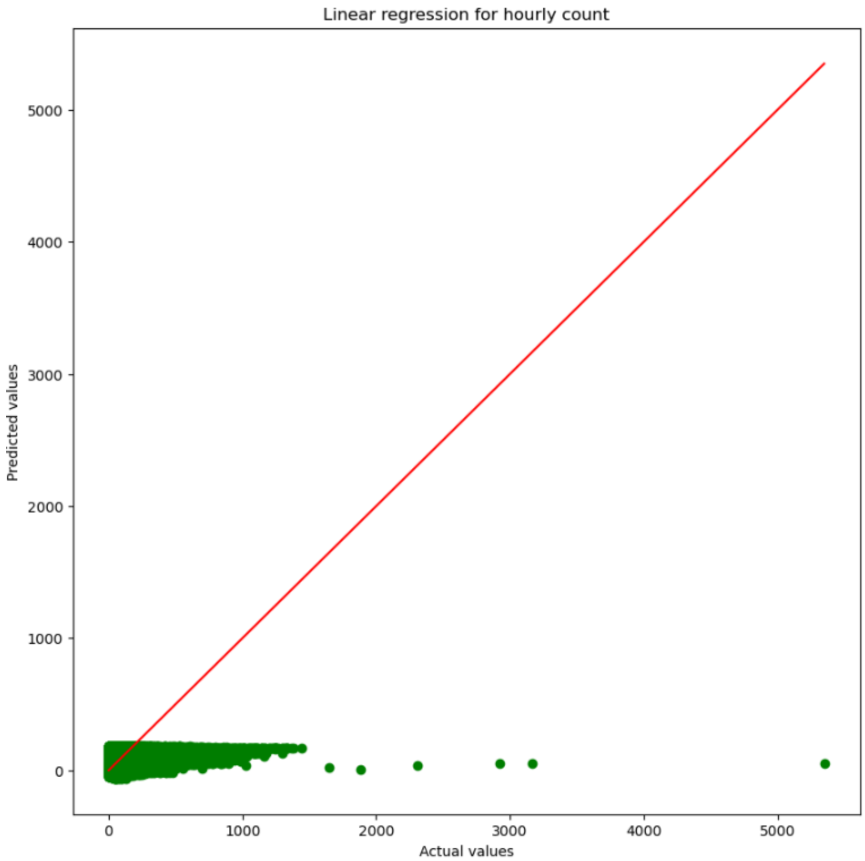
Train set: 98.79717324620661
Test set: 99.20380396972541

The close values of the train and test values indicate that the model has similar performance.

Coefficient of Determination:

Train set: 0.1104310014621509
Test set: 0.11054458897058861

Here we see how poorly this model explains only 11% of the variability of explanatory variables.



We observe on the scatter plot that our model predicts very poorly on the test set.
Based on our performance metrics and visualization of the model, we see that Linear Regression is not the right model to predict our data.

4.2 Decision Tree Regressor

4.2.1 Introduction and advantages of a model

We developed a decision tree regressor that can offer us some advantages :

- **Interpretability**: Decision trees are easy to understand and interpret.
- **Non-linear relationships**: Decision trees can capture nonlinear relationships between features and the target variable.
- **Feature importance**: They provide a way to measure the importance of each feature in predicting the target variable, which can be useful for feature selection and understanding the underlying patterns in the data.
- **Robustness to outliers**: decision trees are relatively robust to outliers in the data, as the splitting criteria can naturally handle extreme values.

Regarding the disadvantages of this model, we should be aware that it is prone to overfitting, it can show bias towards dominant features and the interpretability can be reduced due to complex trees.

4.2.2 Performance metrics

We first implement a decision tree regressor model without modifying any hyperparameter.

We obtained these results :

Mean Absolute Error (MAE):
Train Set: 1.3438737689586242
Test Set: 18.976431241555943

The MAE on the training set is much lower than on the test set. This indicates that there is a significant increase in error when predicting on the test set.

Mean Squared Error (MSE):
Train Set: 84.8827773702511
Test Set: 2023.6478113836233

The low train MSE indicates that the model fits the training data quite well, with relatively low errors, however, the high test MSE indicates that the model performs poorly on the test data, suggesting a significant discrepancy between the training and test performance.

Root Mean Squared Error (RMSE):
Train Set: 9.213184974277414
Test Set: 44.98497317308996

The model has large prediction errors on the test set, emphasizing poor generalization.

Residual Standard Error (RSE):
Train set: 9.213197701824512
Test set: 44.98515960645682

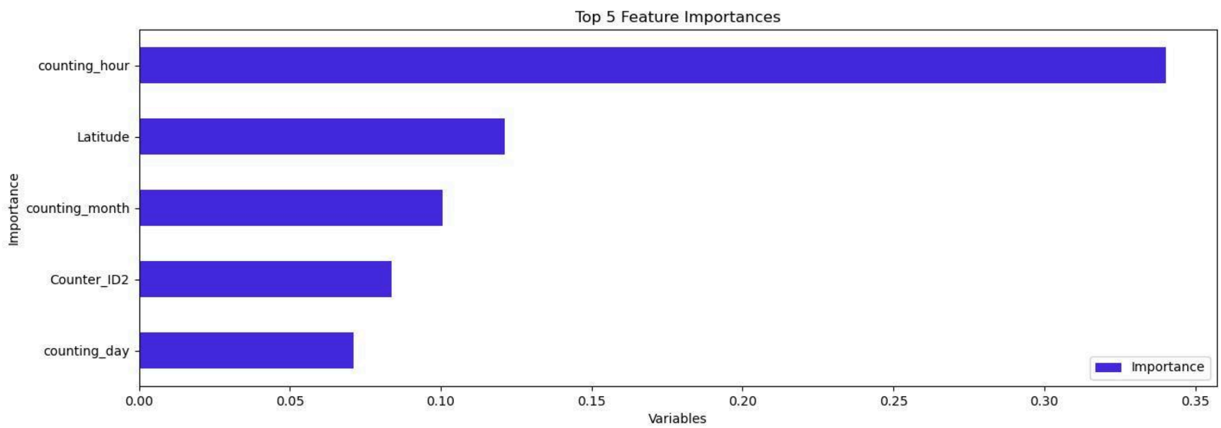
Like the RMSE metric, the RSE shows low prediction errors on the training set and confirms the poor performance on test data.

R^2 (Coefficient of Determination):
Train Set: 0.9922853838628101
Test Set: 0.8155874678616393

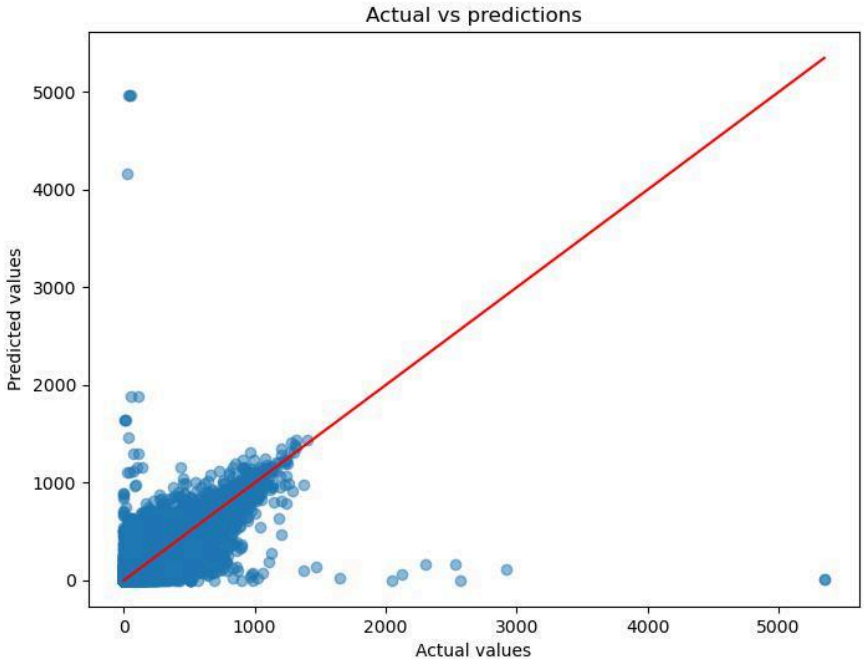
The R^2 value is very high on the training set, indicating that the model explains 99.23% of the variance in the training data.

However, the R^2 on the test set is 0.818, which is lower but still reasonably good, explaining about 82 % of the variance in the test data.

The significant difference between the training and test metrics suggests that the model is overfitting.

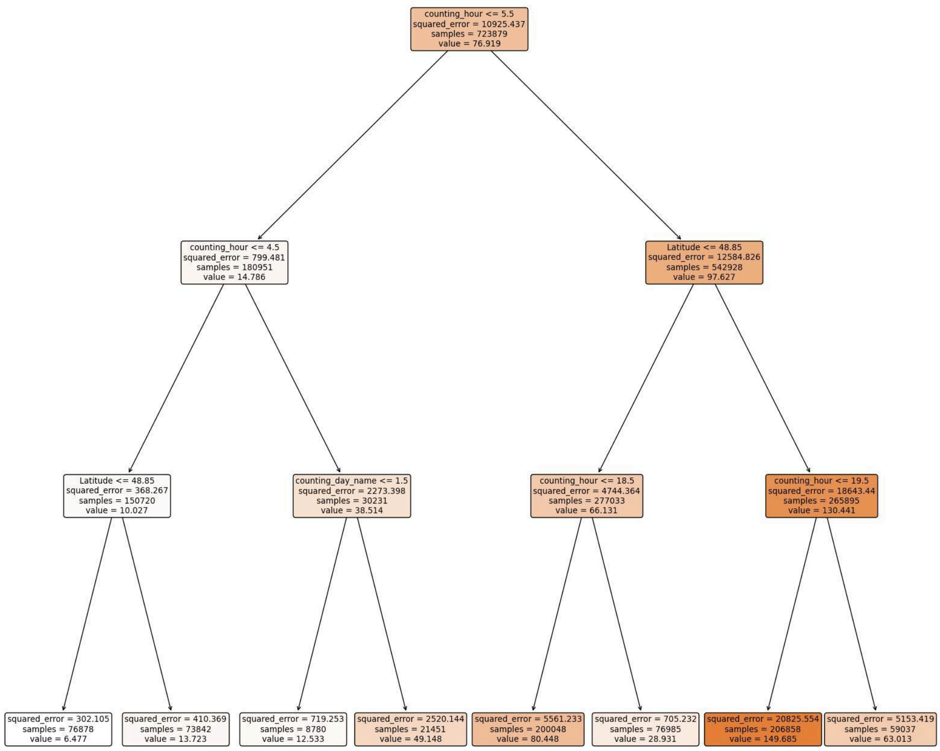


The model relies heavily on Counting_hour for predictions, followed by other features like Latitude and counting_month.
This insight can help in understanding that the hour of counting seems to have a lot of influence on the number of counted bicycles.



The scatter plot indicates that the model has a good fit overall but exhibits some errors, especially at higher values of the target variable. This suggests that while the model performs well, it may benefit from further tuning or from more complex modeling techniques to improve accuracy

The plots together indicate that the model has captured the key features influencing the target variable. However, it may still be overfitting or underfitting certain aspects of the data. Further analysis and potential model adjustments could improve its performance.



We modified the hyperparameters (max_depth=10, min_samples_split=2, min_samples_leaf=1, random_state=42) and scaled the data in order to get a better model's performance. Unfortunately, we didn't succeed in optimizing the model's accuracy and we obtained the following metrics :

Mean Absolute Error (MAE):
Train Set: 31.856207101622108
Test Set: 31.948153782343347

Mean Squared Error (MSE):
Train Set: 3605.202534791112
Test Set: 3967.1722741198205

Root Mean Squared Error (RMSE):

Train Set: 60.04333880449281
Test Set: 62.98549256868458

Residual Standard Error (RSE):
Train set: 60.04342175131573
Test set: 62.9857536024613

R^2 (Coefficient of Determination):
Train Set: 0.6688289362457583
Test Set: 0.6496497611244065

Cross-validation scores: [-0.29200329 0.33016705 0.393109 0.18065878 0.03006565]
Average cross-validation score: 0.12839943686080144

The new model has reduced overfitting, as indicated by the similar error metrics between the training and test sets.
Despite reducing overfitting, the overall performance of the model has decreased.
The higher MAE, MSE, RMSE, and lower R^2 values indicate that the model's predictions are less accurate.
The low and variable cross-validation scores further suggest that the model's generalization ability is limited.

4.3 Random Forest regressor

4.3.1 Introduction and advantages of a model

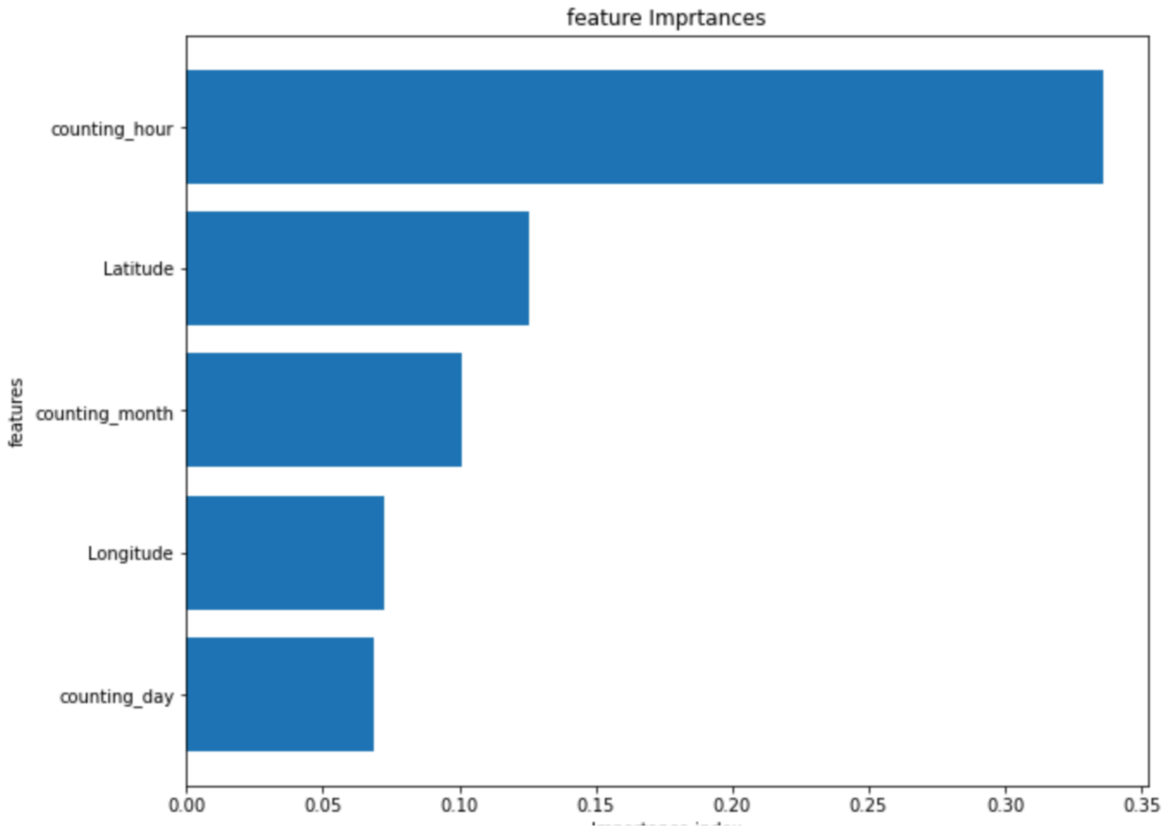
This model has advantages from the demerits from Decision Tree Classifier. This model is unbiased to the variables and builds multiple trees around 100 from bootstrap data, averages the result of each tree, and delivers the output, which makes the output more robust.

The data set prepared for the train and test sets was 80% and 20% without scaling. The Random Forest Regressor is not sensible to the scaled data set.

Random Forest Regressor	**(R^2) Train set**	**(R^2) Test set**
Without specification	0.9794	0.90507
With Depth = 3	0.2490	0.25972
With Depth =5	0.41417	0.40429

By the observation of the scores from the table with and without depth of trees specification. The score of the train test set tends to increase proportionally to the depth specified but still gives a low score as the lowest leaf nodes are not as pure. But without depth specification, the model seems to produce more pure leaf nodes for all the trees.
The score between the train and test sets seems to have a small difference, which leads to overfitting but is still ignorable as the difference is small.

The feature importance for the first tree:



4.3.2 Performance metrics

The Random forest Regression metrics are

Mean Absolute Error (MAE):
Train Set: 5.99
Test Set: 14.623

The MAE on the training set is a bit lower than on the test set. This indicates that there is a significant increase in error when predicting on the test set.

Mean Squared Error (MSE):
Train Set: 222.18
Test Set: 1099.28

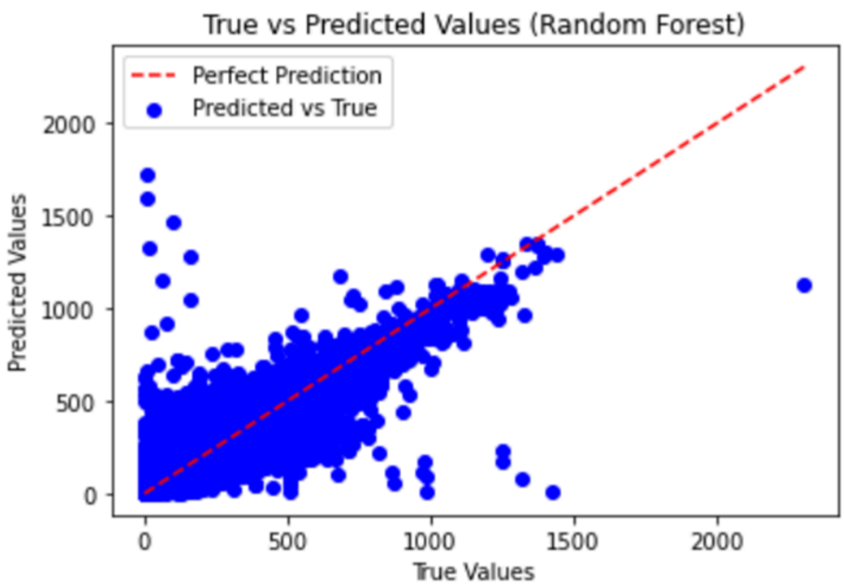
The MSE parameter is for highest penalizing parameter as we see the the train set is lower than the test set, Which means the model is more overfitting towards train rather than the test set.

Root Mean Squared Error (RMSE):
Train Set: 14.5
Test Set: 35.42

Residual Standard Error (RSE):
Train set: 0.0165
Test set: 0.0806

The RSE shows the lowest score in the train set compared to the test set. the performance of the model for the test set is still considered good, as the difference is not much.

R^2 (Coefficient of determination):
Train Set: 0.97
Test Set: 0.90



Even though the performance of the model is better, it still takes more computation time as the generation of n trees is unbiased to the feature variable.

4.4 Lasso

4.4.1 Introduction and advantages of a model

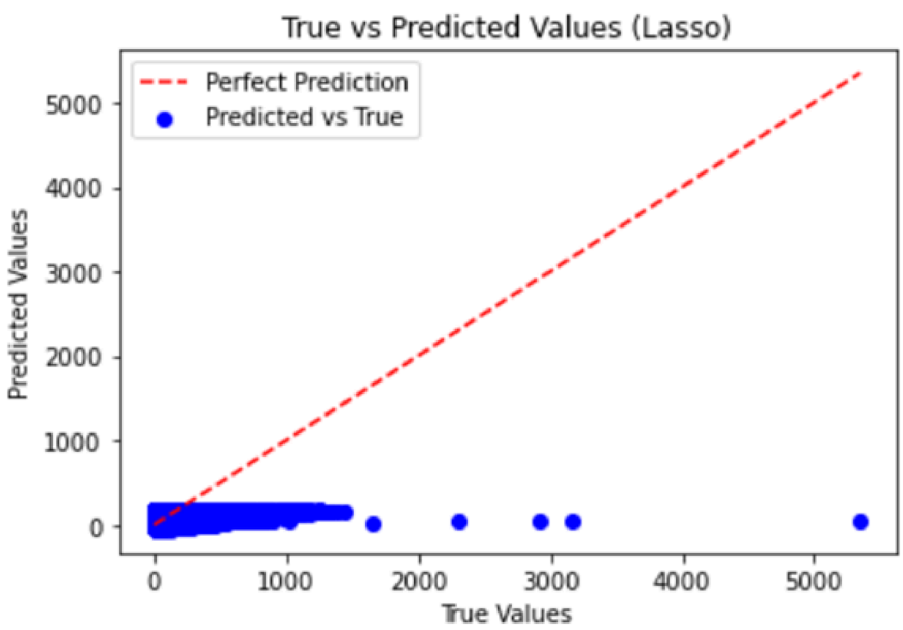
This model falls under the family of linear regression but was developed due to the demerits of linear regression. This model adds an extra parameter to the mean squared value to penalize the steepness of the regression line and is also biased to certain parameters where their value tends to be zero.

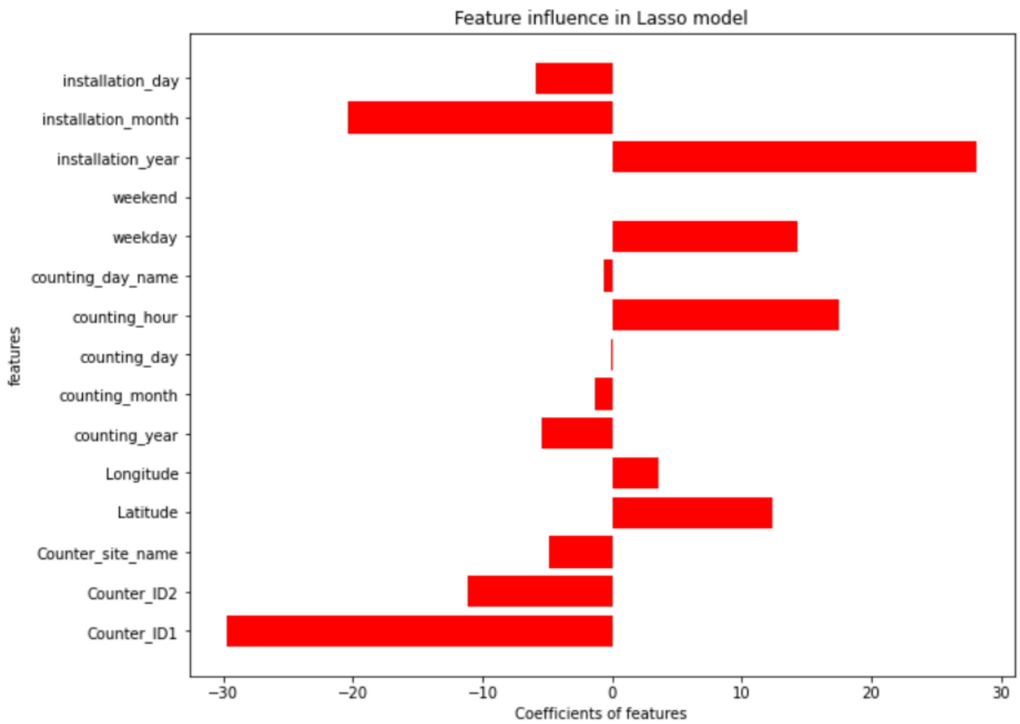
Lasso Regression is sensible to the scaling values :

Lasso Regression	Train	Test
Without Scaling	0.09685	0.09525
With scaling	0.10755	0.10733

As per the score, the model is very underfit to predict the true values. So, the next step gave a bit of freedom to the model to choose the best value for the training.

The total metrics compared between the models are the Relative absolute error, the Root Mean squared Error and the Residual standard Error).





4.4.2 Performance metrics

Mean Absolute Error (MAE):
Train Set: 63.7
Test Set: 63.82

The MAE on the training set is considerably high for train and test sets . This indicates that there is a model that is not fitted between the train and test set.

Mean Squared Error (MSE):
Train Set: 9758.9
Test Set: 9869.3

As per MAE the high penalizing parameter shows the higher values of train and test set and indicates the poor performance of train and test set.

Root Mean Squared Error (RMSE):
Train Set: 99.78
Test Set: 99.34

Residual Standard Error (RSE):
Train set: 0.112
Test set: 0.226

The RSE shows the highest score in the train set than the test set compared to the other models.

R^2 (Coefficient of Determination):
Train Set: 0.11
Test Set: 0.11

Even after hyper parameter tuning and use of scaling the score of Lasso still remains low which makes the model unfit for this kind of data set.

Lasso Regression	**Train**	**Test**
Without Alpha Tuning	0.10755	0.10733
With Alpha Tuning	0.11044	0.11051

The Lasso model is also part of the linear regression family. The issue with all linear regression models regarding some particular datasets is that they are unfit to predict as they are unable to capture non linearity. In the case of Lasso the model can shrink some coefficients to zero, still maintaining linearity in the relationships.

4.5 LassoCV

4.5.1 Introduction and advantages of a model

After checking the Lasso, we also wanted to test the model CV. The main difference between Lasso and Lasso CV lies in how they handle the regularization parameter α (alpha) in the Lasso regression model. In Lasso regression, the regularization parameter α\alpha is a fixed value that must be chosen by the user, while LassoCV automates the selection of the optimal alpha by performing cross-validation.
It checks how well the model works with different alpha values and picks the one that has the lowest error during cross-validation.

There are several advantages to this model:

- **Optimal alpha Selection:** LassoCV automates the selection of the optimal alpha
- **Easy to use:** Since the alpha is automated, the model is less complicated than Lasso
- **Reduced Overfitting:** This way of selecting alpha that balances bias and variance, reduces the risk of overfitting
- **Customizable Cross-Validation**: You can choose how to split your data into different parts to test your model.

LassoCV helps in selecting important features and preventing overfitting by shrinking coefficients towards zero, but it can be sensitive to how data is scaled and may not work well with highly correlated variables.

4.5.2 Performance metrics

There are several metrics to check the performance of Lasso CV. We will now check: R^2, MAE, MSE, RSE and RMSE on train and test sets to compare with other models.

Mean Absolute Error (MAE):
Train set: 63.744273399476775
Test set: 63.818414335810154

Here we see that the model has similar performance (error) on the train and test data.

Mean Squared Error (MSE):

Train set: 9761.463974543169
Test set: 9842.163481325553

Here we have very high values, which indicates that there are a lot of errors. We want the MSE to be as close to 0 as possible.

Root Mean Squared Error (RMSE):
Train set: 98.80012132858526
Test set: 99.20767854014906

The model performs similarly on the train and test data sets.

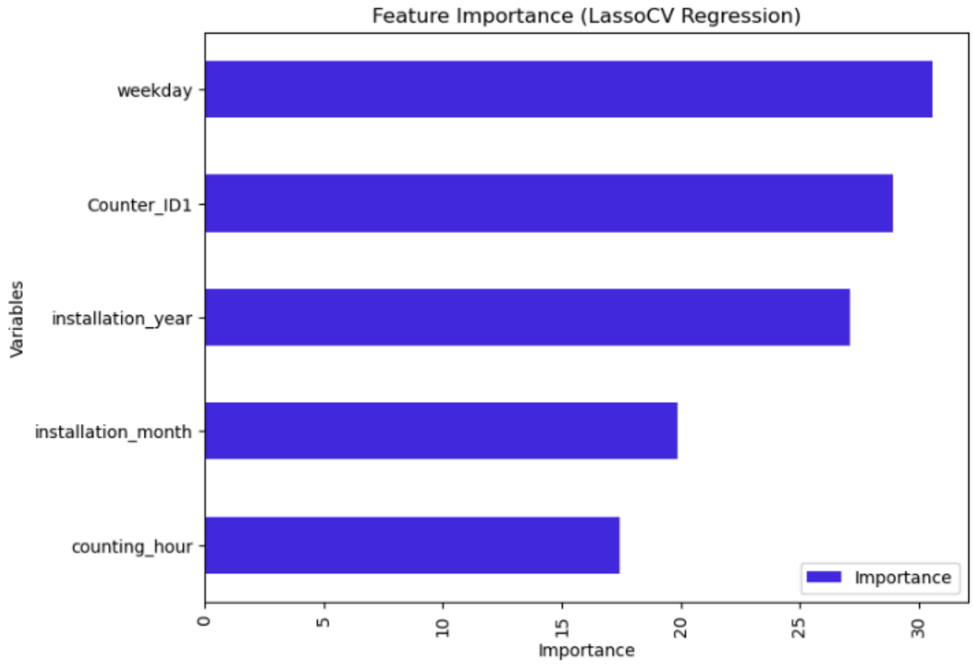
Residual Standard Error (RSE):
Train set: 98.80025781593474
Test set: 99.20808969125629

The close values from train and test values indicate that the model has similar performance on train and test data.

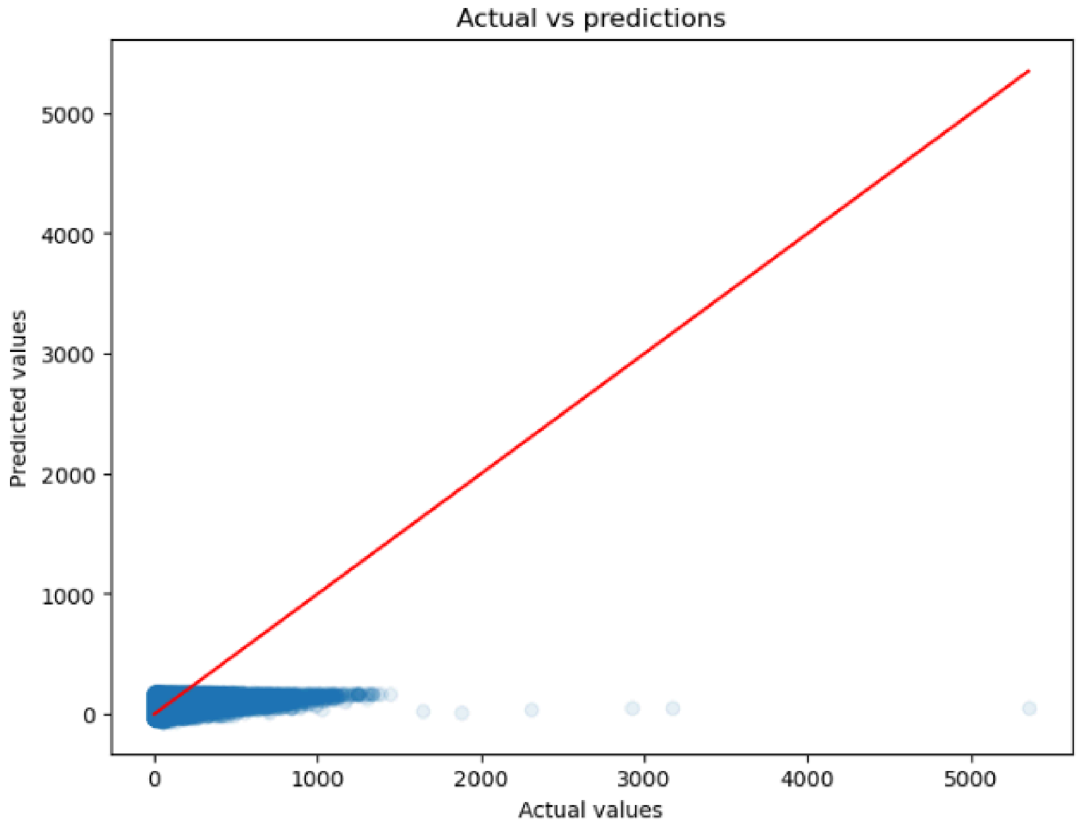
R^2 (Coefficient of Determination):
Train set: 0.110375453710158
Test set: 0.11046773626144202

Here we see how poorly this model explains only 11% of the variability of explanatory variables.

After training our model, we have calculated the feature importance.



We see that the most important variable is "weekday."



We see that LassoCV poorly predicts hourly counting.

Based on our performance metrics and visualization of the model, we see that Lasso CV is not a right model to predict our data.

4.6 Ridge

4.6.1 Introduction and advantages of a model

Ridge regression is a powerful tool for improving the robustness and generalization of predictive models, particularly useful in contexts where data is complex and where there are risks of overfitting and multicollinearity. The Ridge regression model offers several advantages :

- **Reduction of overfitting**: Ridge regression adds a penalty term (regularization term) to the cost function, which prevents the coefficients from becoming too large. This helps reduce the risk of overfitting, especially when the number of explanatory variables is high relative to the number of observations.
- **Handling multicollinearity**: when there is multicollinearity, Ridge regression stabilizes the coefficients by imposing a penalty, thus better handling multicollinearity.
- **Ease of implementation**: Ridge regression is relatively simple to implement and understand.
- **Robustness**: In situations where data contains noise, Ridge regression can be more robust compared to ordinary least squares regression because it limits the complexity of the model.

On the other side, Ridge regression assumes linearity, is sensitive to outliers, and requires feature scaling. That could be a barrier to the model's performance, depending on the data structure we want to use to make predictions.

4.6.2 Performance metrics

Mean Absolute Error (MAE):
Train Set: 63.73434575780083
Test Set: 63.956104361998456

Mean Squared Error (MSE):
Train Set: 9673.637539312582
Test Set: 10103.068128652458

Root Mean Squared Error (RMSE):

Train Set: 98.35465184378714
Test Set: 100.51401956270806

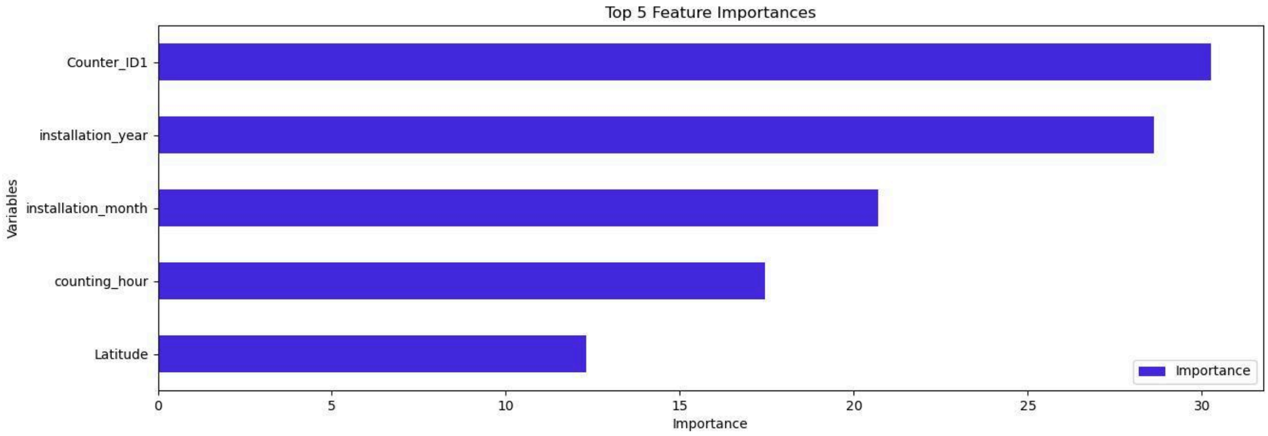
Residual Standard Error (RSE):
Train set: 98.35478771574317
Test set: 100.5144361277466

The trained model has similar performance on both the training and test sets, as indicated by the comparable values for MAE, MSE, RMSE, and RSE.

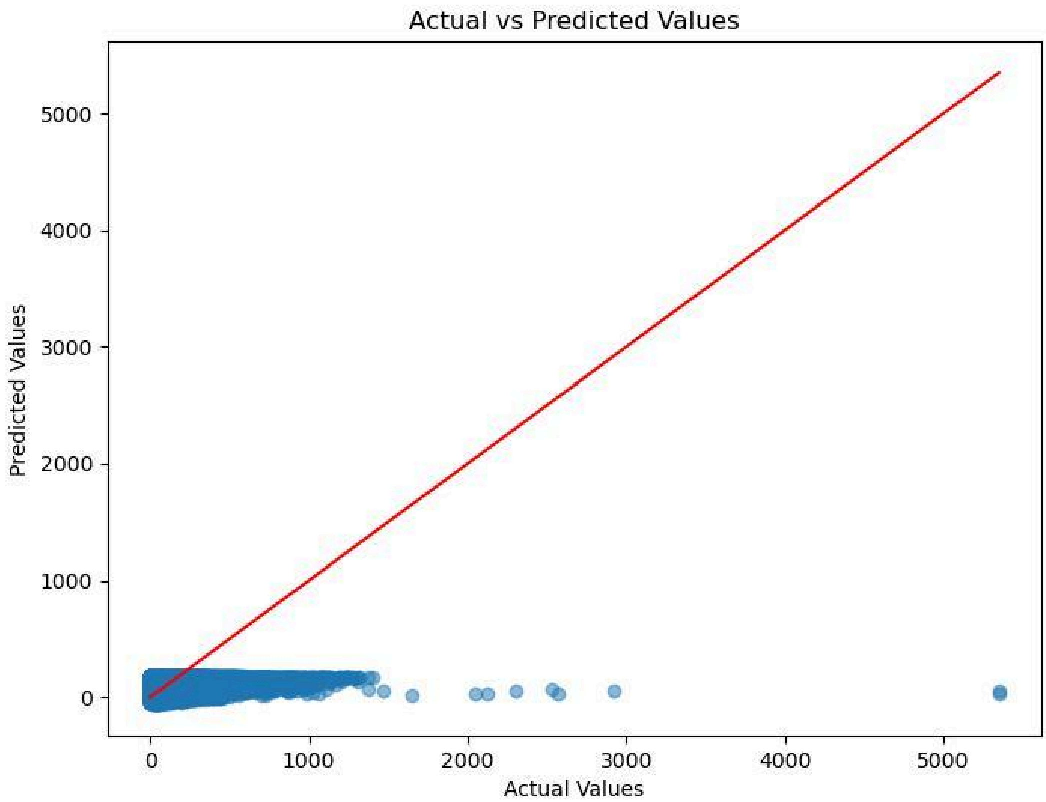
The error values (MAE, MSE, RMSE, RSE) are relatively high, suggesting that the model's predictions are not very accurate.

R^2 (Coefficient of Determination):
Train Set: 0.1113872789805046
Test Set: 0.10777448326589167

The low R^2 values indicate that the model explains only a small fraction of the variance in the target variable.



The high importance of Counter_ID1 may imply that specific counters have unique patterns that are critical for predictions.
The importance of counting_hour, installation_year, and installation_month suggests that temporal factors are crucial in predicting the target variable.



The plot shows a concentration of points at lower values with some dispersion, indicating decent performance but critical issues at higher values.

We tried without success to improve our model's performance by tuning the hyperparameters using the GridSearch function to choose the best alpha among these values : [0.01, 0.1, 1.0, 10.0, 100.0, 1000.0]. The model found that our initial alpha (10) was the best possible hyperparameter.

4.7 Model's assessments

Metric	MAE		MSE		RMSE		R^2	
Model	Train set	Test set	Train set	Test set	Train set	Test set	Train set	Test set
Random Forest	**5.99**	**14.623**	**222.18**	**1099.28**	**14.5**	**35.42**	**0.98**	**0.90**
Lasso	63.7	63.82	9758.9	9869.3	99.78	99.34	0.11	0.11
Decision Tree	**31.9**	**31.9**	**3664.2**	**3786.3**	**60.5**	**61.5**	**0.67**	**0.65**
Ridge	63.9	63.5	9787.4	9761.7	98.9	98.8	0.11	0.11
Linear	63.7	63.8	9760.8	9841.3	98.7	99.2	0.11	0.11
Lasso CV	63.7	63.8	9761.4	9842.1	98.8	99.2	0.11	0.11

Lasso, Ridge, Linear, and Lasso CV models have very similar performance metrics with significantly higher error rates (MAE, MSE, RMSE) and much lower R^2 values around 0.11 for both train and test sets.

Decision Tree shows better performance than Lasso and Ridge models but still has significantly higher error rates than Random Forest.

The Random Forest model has the lowest MAE, MSE, and RMSE on the test set compared to all other models. This indicates that the predictions made by the Random Forest are, on average, closer to the actual values than those made by the other models.
The R^2 value for the Random Forest is 0.90 on the test set, which is substantially higher than those of the other models . It indicates a better fit of the model to the data.

Although the Random Forest model shows some increase in error from the training set to the test set, its performance degradation is much less severe than that of the Decision Tree. This suggests that the Random Forest model generalizes better to unseen data.
Given these points, the Random Forest model demonstrates a strong ability to predict the target variable accurately and with less error compared to the other models.
This makes it a relevant choice for your predictions.

5. Conclusion

Throughout the project, we made several changes to our dataset in order to train several machine learning models dedicated to providing useful information and actionable advice to Paris' city authorities regarding bicycle traffic across the city.

The dataset needed to be transformed in order to train our model, therefore, we got rid of the missing values and variables without much importance. Once we trained our 6 models, we decided to keep the one that gave us the best results, the **random forest model**. Even if this model seems slightly overfitted compared to others, this could not be considered significant.

Actually, the performance of this model was the highest among all the trained models, and its strengths convinced us to choose it to predict the future values of our target variable. This model, well suited for our regression problem, can capture complex relationships between the target variable and the features, offers very good accuracy, and can ignore the noise we may have in the data. It allows us to identify the relative importance of every feature, helping us to strengthen our knowledge about which ones have influence over the final result of hourly counting for every counter. Finally, this model is a good alternative for large datasets like the one we worked with and is the one that makes the most reliable predictions on unseen data.
To conclude, we can affirm that the Random Forest will be a robust and reliable tool for decision makers to manage the affluence areas and timetables of cycling traffic across the French capital.