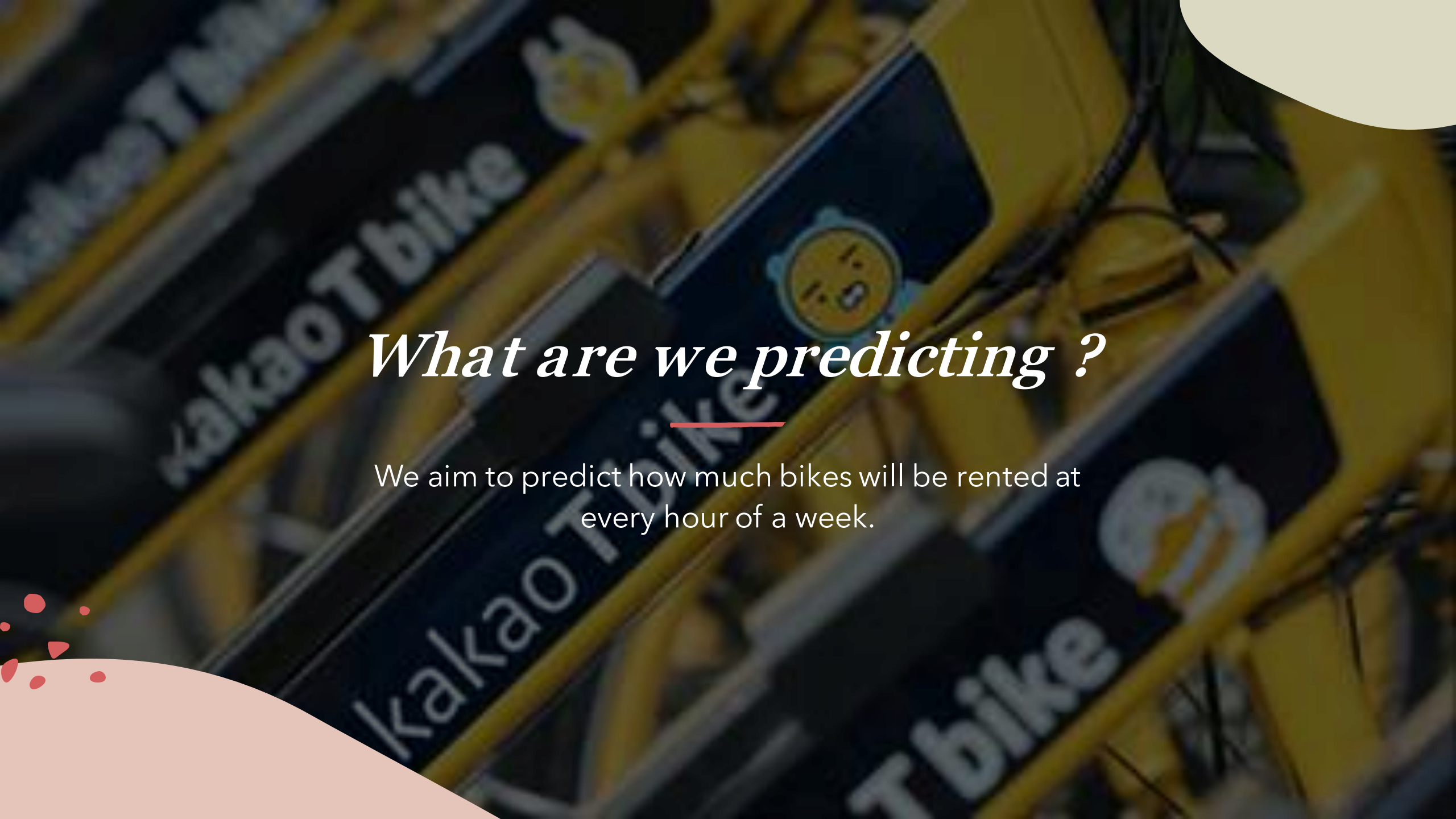
Seoul Bike Sharing Demand Analysis

Summary

1) Dataset explanation
2) What we aim to predict
3) Different variables
4) Data visualization
5) Data pre processing
6) Data modeling



What are we predicting ?

We aim to predict how much bikes will be rented at every hour of a week.

Seoul Bike sharing demand Datafile

This datafile contains for each hour in a period of one year : How many bikes have been rented, information about the weathers conditions and information about the rent time.

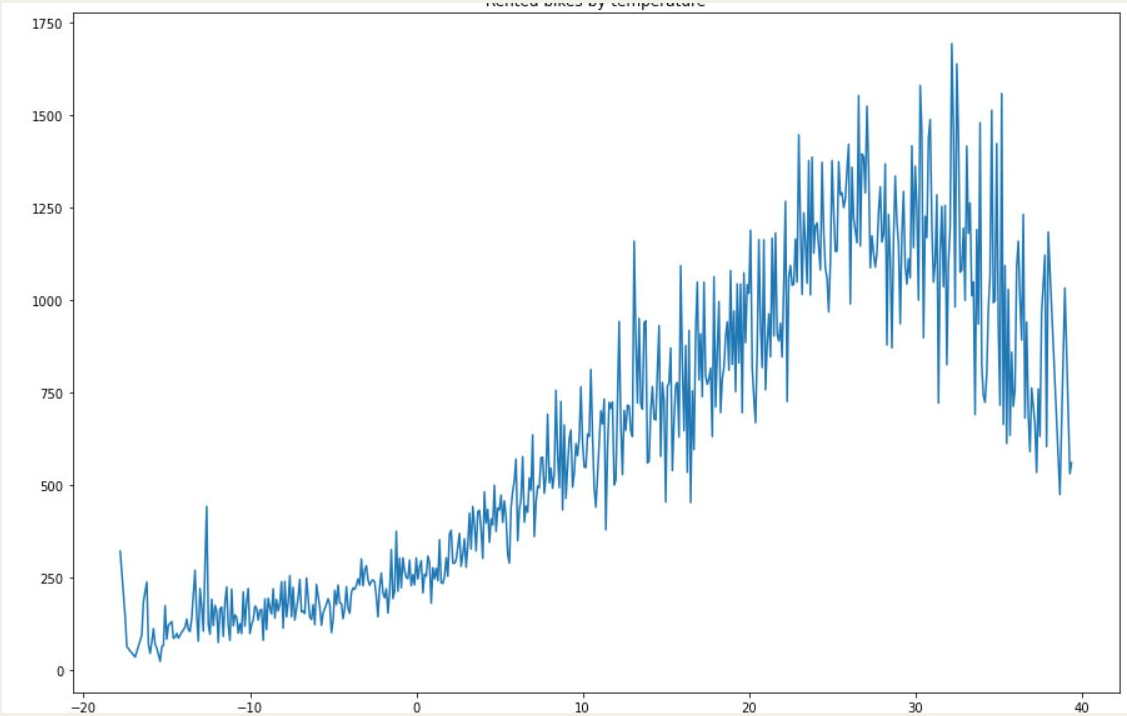
The dataset's variables

We created more variables based on time for a more precise analysis : the day, month and year out of the date variable.

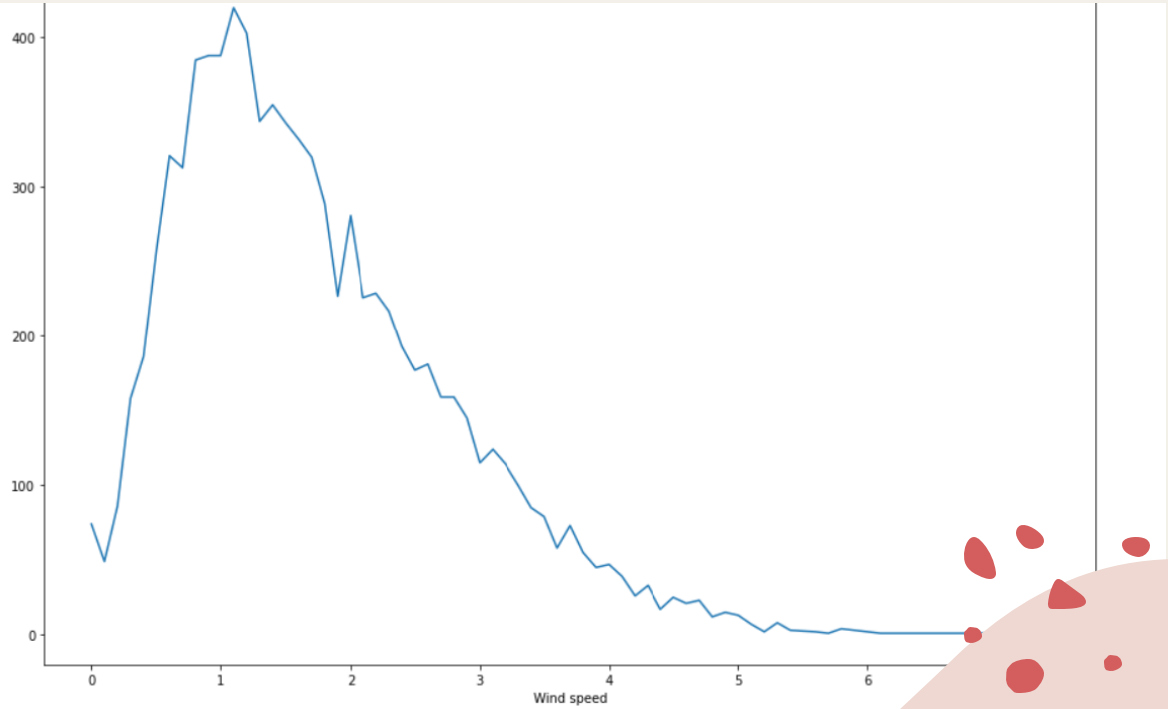
	Date	Rented Bike Count	Hour	Temperature(°C)	Humidity(%)	Wind speed (m/s)	Visibility (10m)	Dew point temperature(°C)	Solar Radiation (MJ/m2)	Rainfall(mm)	Snowfall (cm)	Seasons	Holiday	Functioning Day
0	01/12/2017	254	0	-5.2	37	2.2	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
1	01/12/2017	204	1	-5.5	38	0.8	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
2	01/12/2017	173	2	-6.0	39	1.0	2000	-17.7	0.0	0.0	0.0	Winter	No Holiday	Yes
3	01/12/2017	107	3	-6.2	40	0.9	2000	-17.6	0.0	0.0	0.0	Winter	No Holiday	Yes
4	01/12/2017	78	4	-6.0	36	2.3	2000	-18.6	0.0	0.0	0.0	Winter	No Holiday	Yes

Data Visualization

Rented bikes by Temperature

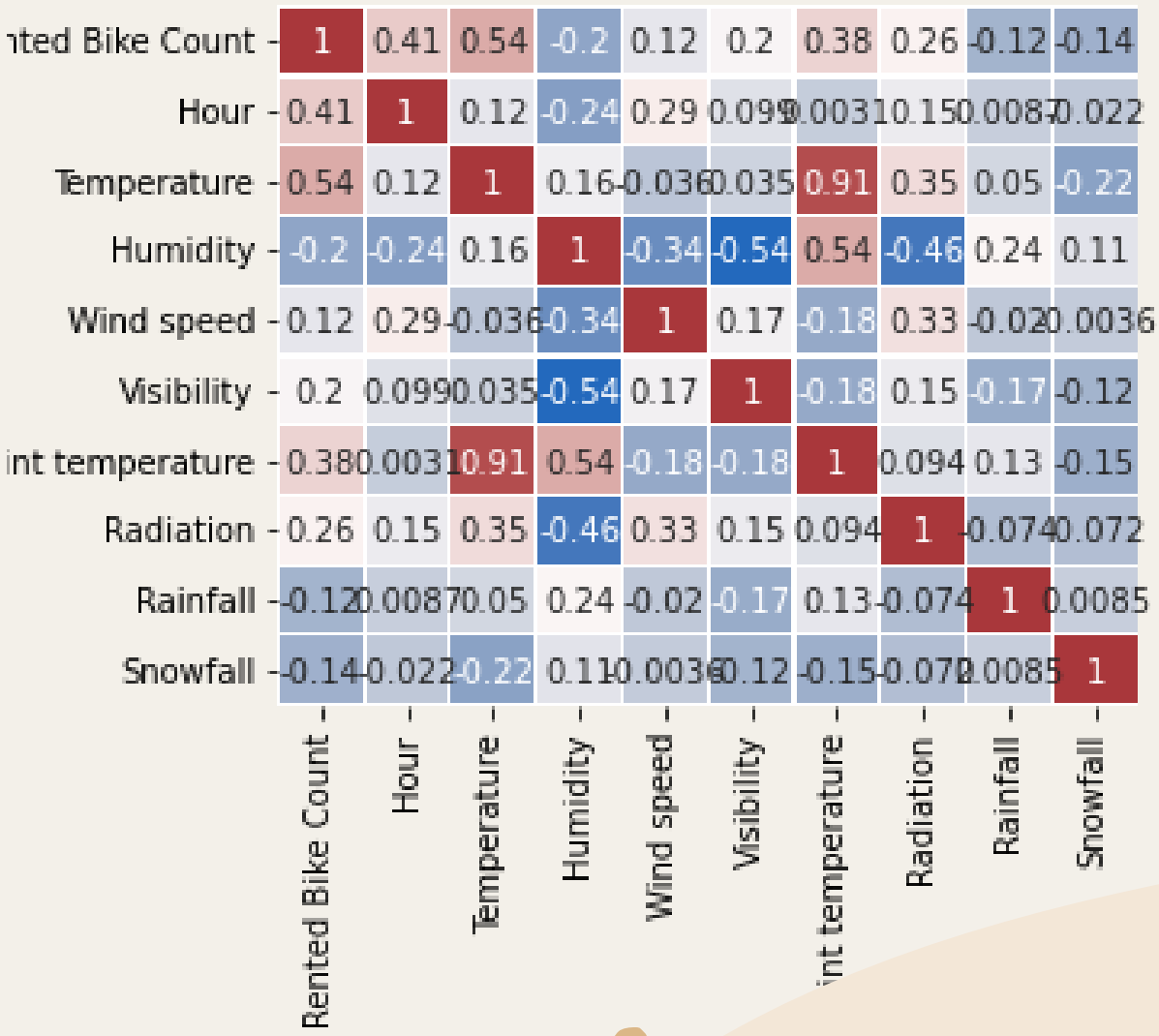


Rented bikes by wind speed

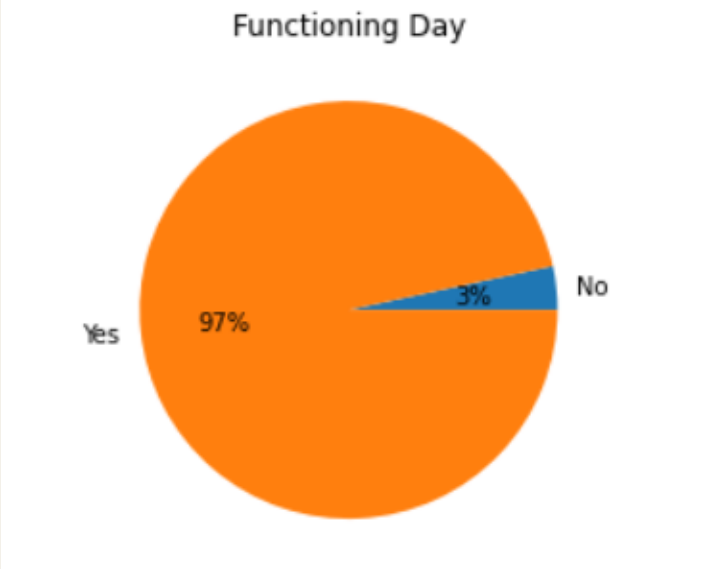


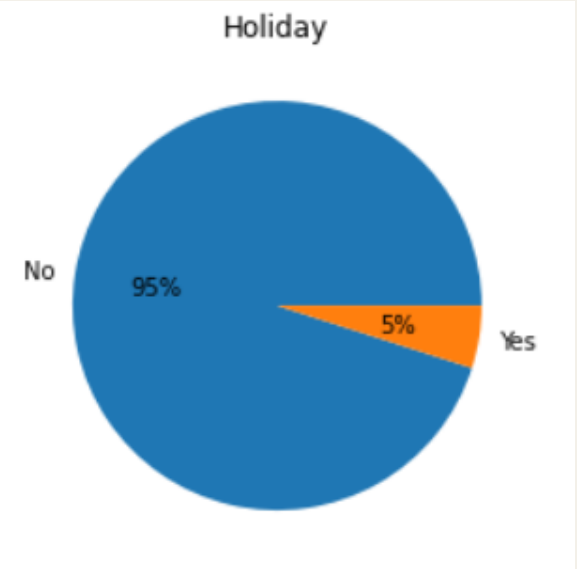
Correlation table

The temperature and hour seem correlated with our target : the rented bike count.

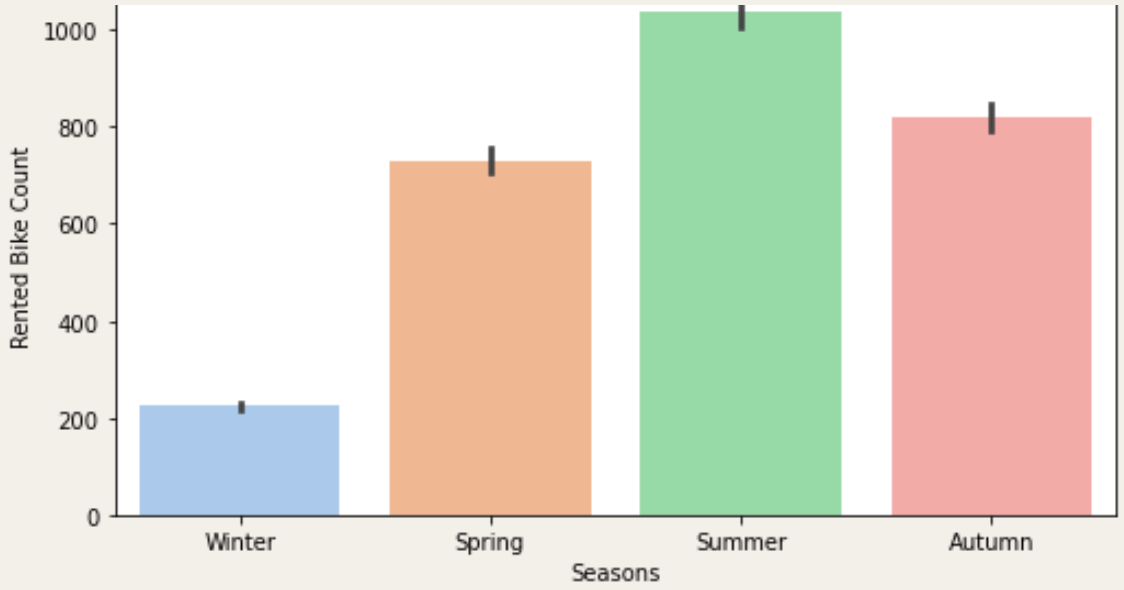


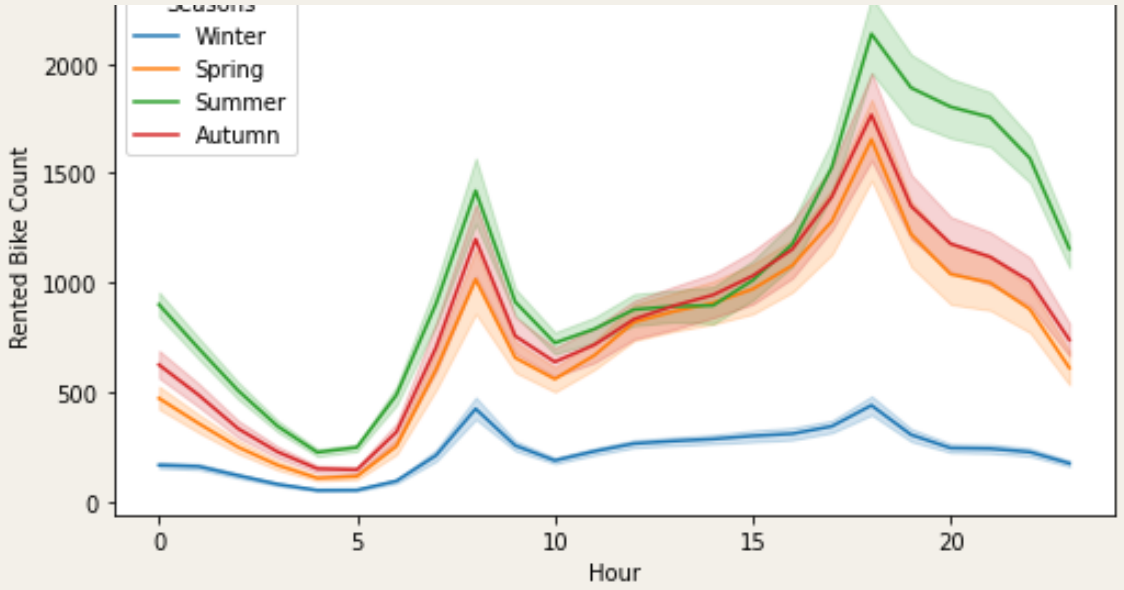
Holiday	Seasons / Functioning Day	Autumn	Spring	Summer	Winter	TOTAL
Holiday	No	24.0	NaN	NaN	NaN	24
	Yes	96.0	72.0	48.0	192.0	408
No Holiday	No	223.0	48.0	NaN	NaN	271
	Yes	1841.0	2088.0	2160.0	1968.0	8057
TOTAL		2184.0	2208.0	2208.0	2160.0	8760

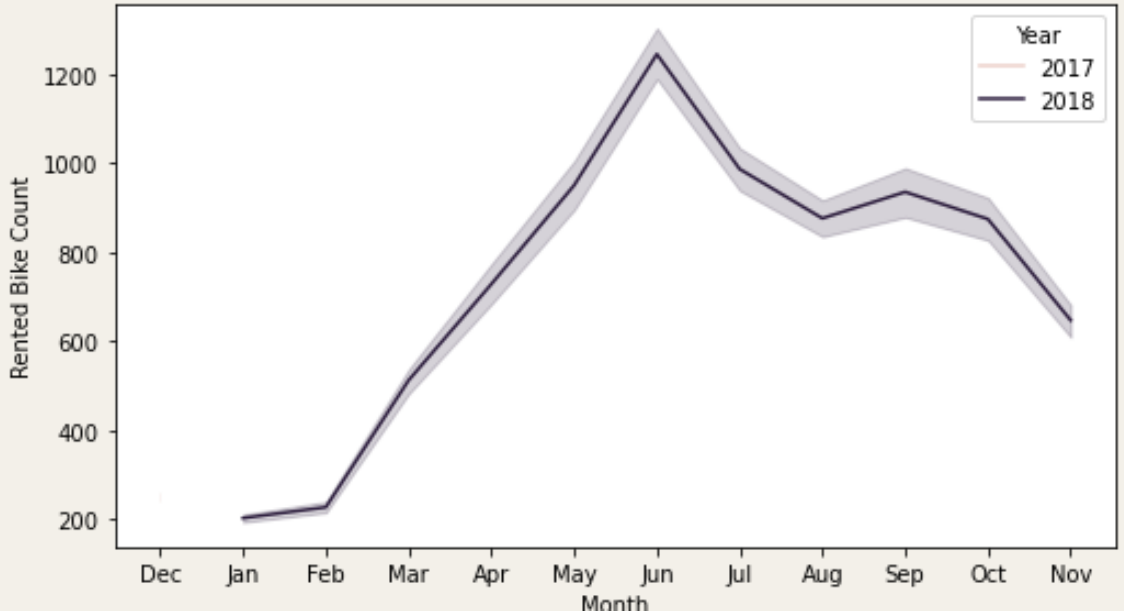


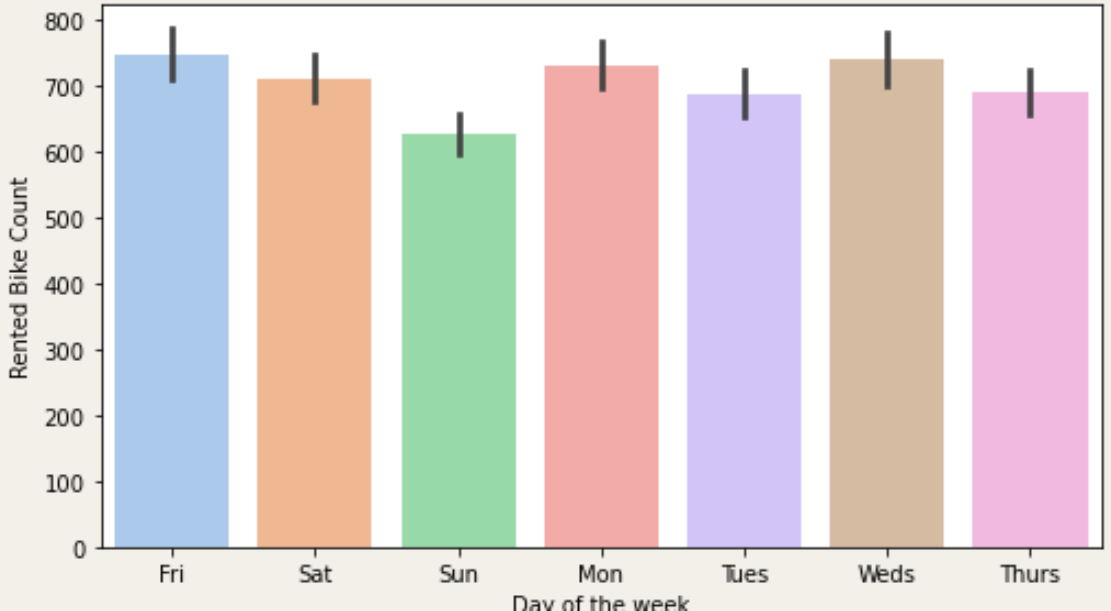


Time impact on bike rental

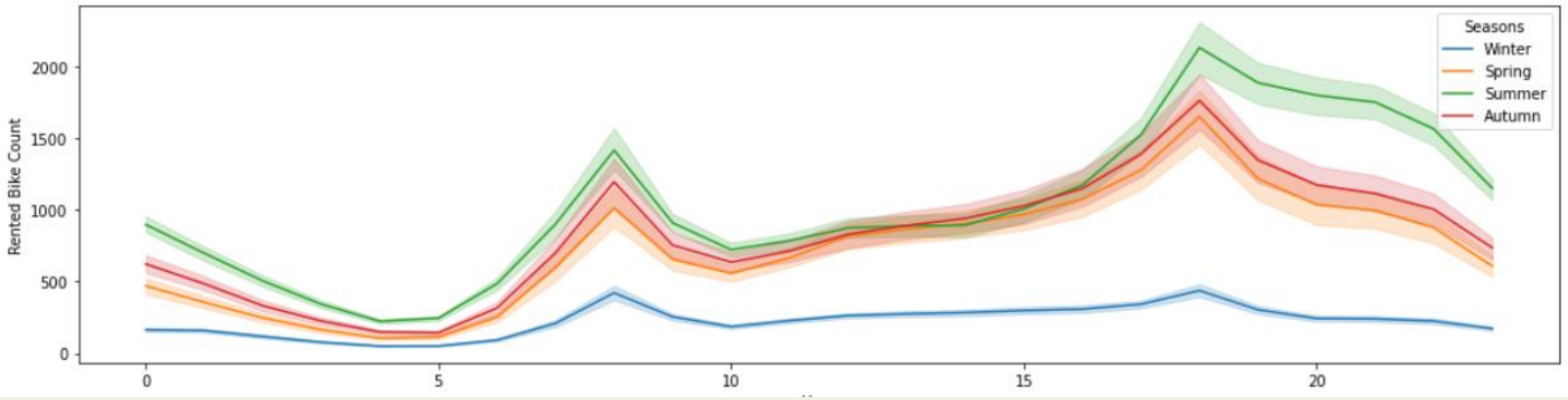








Variables we created

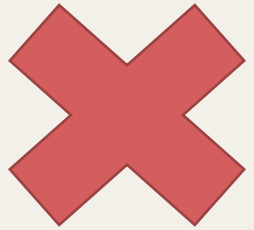


Day of the week	rentedbikes_1dayago
Sun	-0.528675
Sun	-0.675066
Sun	-0.777930
Sun	-1.045763
Sun	-1.198660
...	...
Fri	0.721069
Fri	0.474575
Fri	0.440997
Fri	0.352408
Fri	0.101931

Pre-processing Steps

- Power Transformer
- Encoding (get_dummies)
- Scaling (MaxAbsScaler)

- Dealing with outliers
- Feature Selection (KBest)

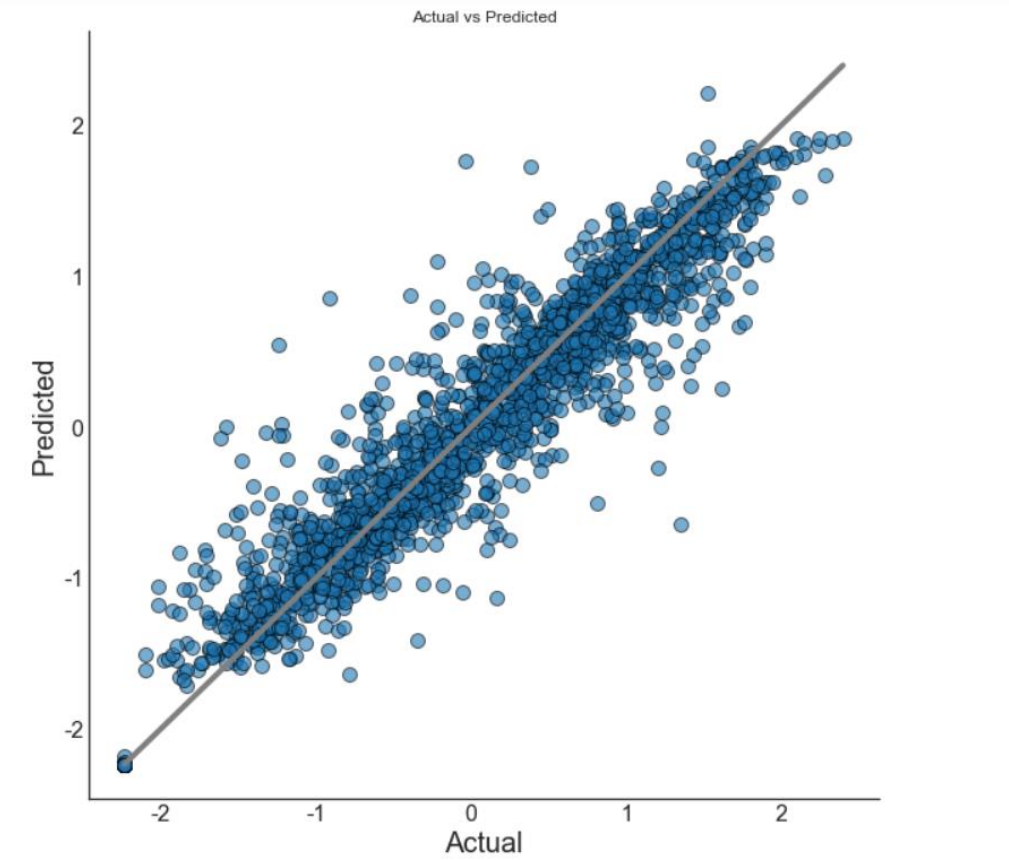


Data modeling

```
Returned hyperparameter: {'LR__fit_intercept': True}
Best regression accuracy in train is: 0.8184012859054697
Regression accuracy on test is: 0.8020385145521969

Returned hyperparameter: {'PR__linearregression__fit_intercept': True,
'PR__polynomialfeatures__degree': 2}
Best regression accuracy in train is: 0.8995815910011565
Regression accuracy on test is: 0.9042007463785586

Returned hyperparameter: {'svr__C': 5}
Best regression accuracy in train is: 0.8097030734274192
Regression accuracy on test is: 0.7949854501762174

Returned hyperparameter: {'DT__max_depth': 8}
Best regression accuracy in train is: 0.8203569497789799
Regression accuracy on test is: 0.8201856908865777

Returned hyperparameter: {'knn__n_neighbors': 7}
Best regression accuracy in train is: 0.8151145001863019
Regression accuracy on test is: 0.8177631405290825

Returned hyperparameter: {'RF__max_depth': 19, 'RF__n_estimators': 240}
Best regression accuracy in train is: 0.8932391381888863
Regression accuracy on test is: 0.8971102960669909
```

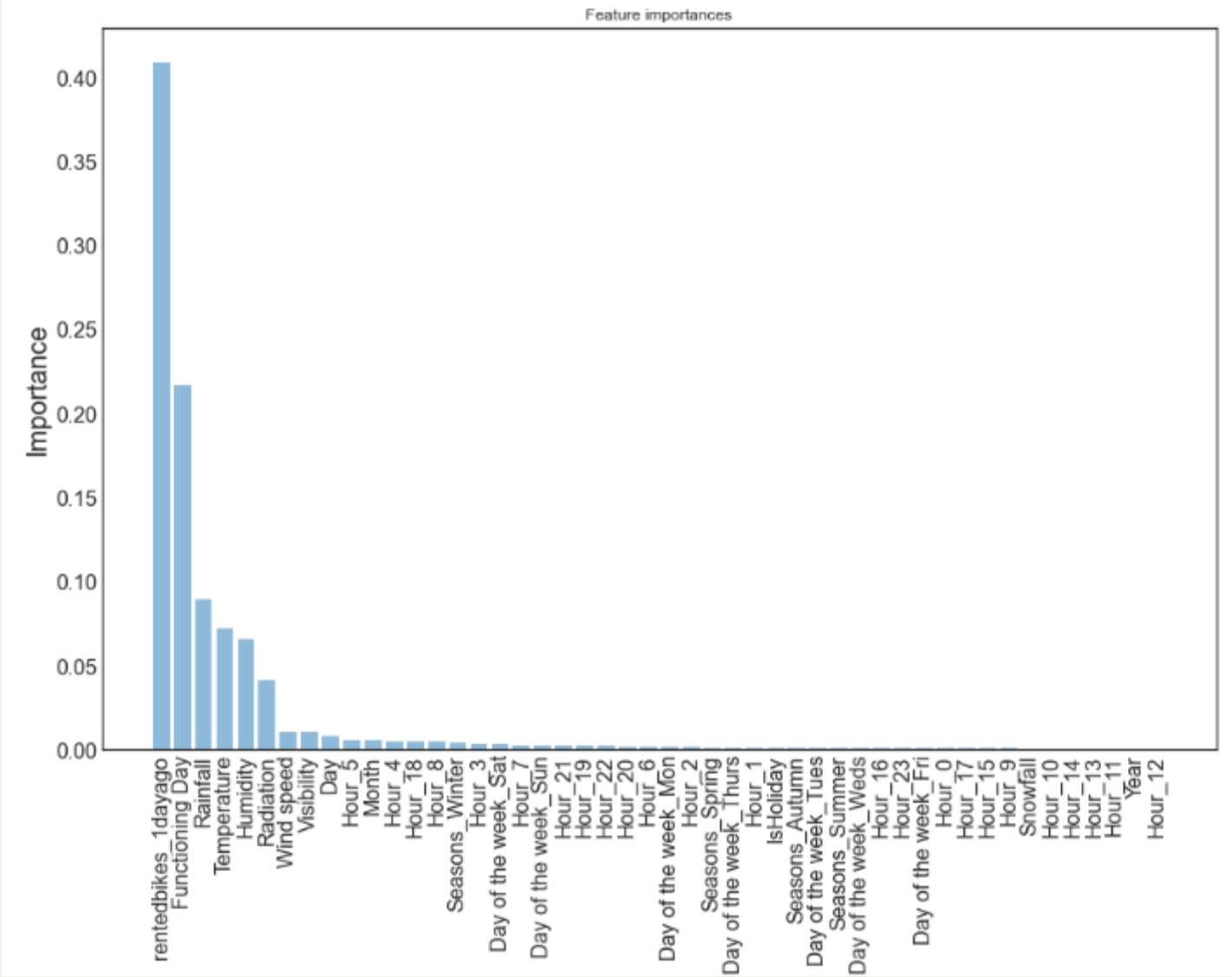
	R-Square (%)	MSE (%)	CV R-Square (%)	CV std (%)	Time to train (secondes)	Time to predict (secondes)
Polynomial Regression	90.421203	9.222543	81.845652	0.750530	0.506417	0.032913
RF	89.753698	9.865222	89.710139	0.924676	8.587755	0.090759
Decision Tree Regression	81.940411	17.387917	82.690024	1.291261	0.028930	0.002985
KNN	81.837306	17.487188	79.615288	0.513011	0.002575	0.264268
Linear Regression	80.203851	19.059891	81.845273	0.750338	0.012965	0.000000
SVR	51.849174	46.360002	37.057300	23.787940	0.391312	0.001975

Random Forest




```
Model R-Square : 89.75%
Model MSE : 9.87%
Cross Val R-Square: 89.71 %
Cross Val Standard Deviation: 0.92 %
```

Most important features



The Flask API