Predictions using spark ml library regression algorithms

Content:

- Spark ml library presentation
- Algorithms selected
    - Decision tree regression
    - Random forest regression
    - Gradient boosted tree regression
- Implementation on the Databike data set
    - Most Used station
    - Least used station
- References

Ps: on the Implementation on the Databike data set part

Generic figures title :

Plot: Data distribution of the actual and predicted data bike usage using **{algrotithms_name**}

Heatmap: Heat map correlation plot showing interval value of the regression line

Test diagram: Test **{num}'s** features importance : **{RMSE value**}

Spark ML

Apache Spark's Machine Learning Library (MLlib) is designed for simplicity, scalability, and easy integration with other tools. With the scalability, language compatibility, and speed of Spark, data scientists can focus on their data problems and models instead of solving the complexities surrounding distributed data (such as infrastructure, configurations, and so on).
Built on top of Spark, MLlib is a scalable machine learning library consisting of common learning algorithms and utilities, including classification, regression, clustering, collaborative filtering, dimensionality reduction, and underlying optimization primitives. Spark MLLib seamlessly integrates with other Spark components such as Spark SQL, Spark Streaming, and DataFrames and is installed in the Databricks runtime. The library is usable in Java, Scala, and Python as part of Spark applications, so that you can include it in complete workflows.
MLlib allows for preprocessing, munging, training of models, and making predictions at scale on data. You can even use models trained in MLlib to make predictions in Structured Streaming. Spark provides a sophisticated machine learning API for performing a variety of machine learning tasks, from classification to regression, clustering to deep learning.[1]

MLlib is Spark's scalable machine learning library consisting of common learning algorithms and utilities that can be divided into two categories:

Supervised learning :

- Classification and regression
    - naive Bayes
    - linear models (SVMs, logistic regression, linear regression)
    - decision trees
    - ensembles of trees (Random Forests and Gradient-Boosted Trees)

Unsupervised learning :

- Collaborative filtering
    - alternating least squares (ALS)
- Clustering
    - k-means
- Dimensionality reduction
    - principal component analysis (PCA)
    - singular value decomposition (SVD)

Spark.ml: high-level APIs for ML pipelines:

Spark 1.2 includes a new package called spark.ml, which aims to provide a uniform set of high-level APIs that help users create and tune practical machine learning pipelines. It is currently an alpha component, and we would like to hear back from the community about how it fits real-world use cases and how it could be improved.

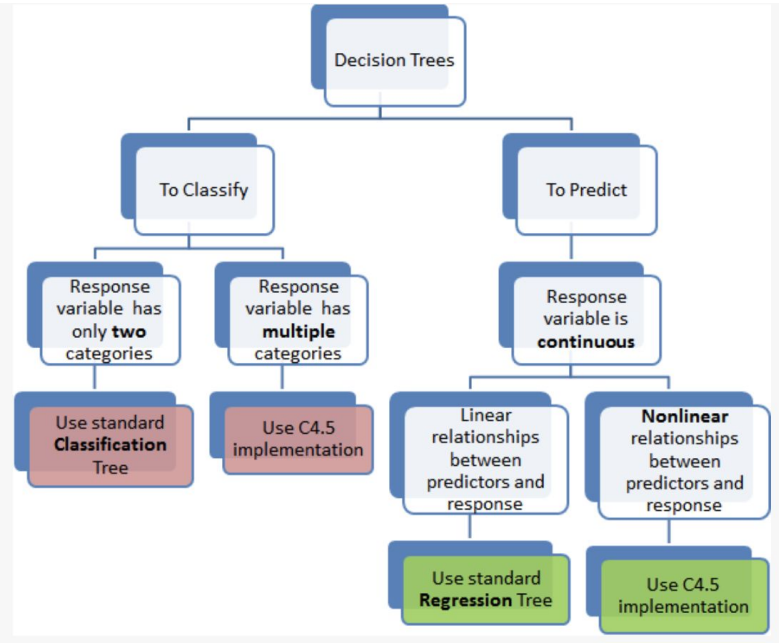
Algorithms selected

Regression analysis consists of a set of machine learning methods that allow us to predict a continuous outcome variable (y) based on the value of one or multiple predictor variables (x). Since our goal is to make predictions on a continuous feature, the next step is to implement the algorithms provided the library and compare their results.

DECISION TREE REGRESSION

How does it work?

A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). We use standard deviation to calculate the homogeneity of a numerical sample. If the numerical sample is completely homogeneous its standard deviation is zero.[3]



Decision tree usage diagram

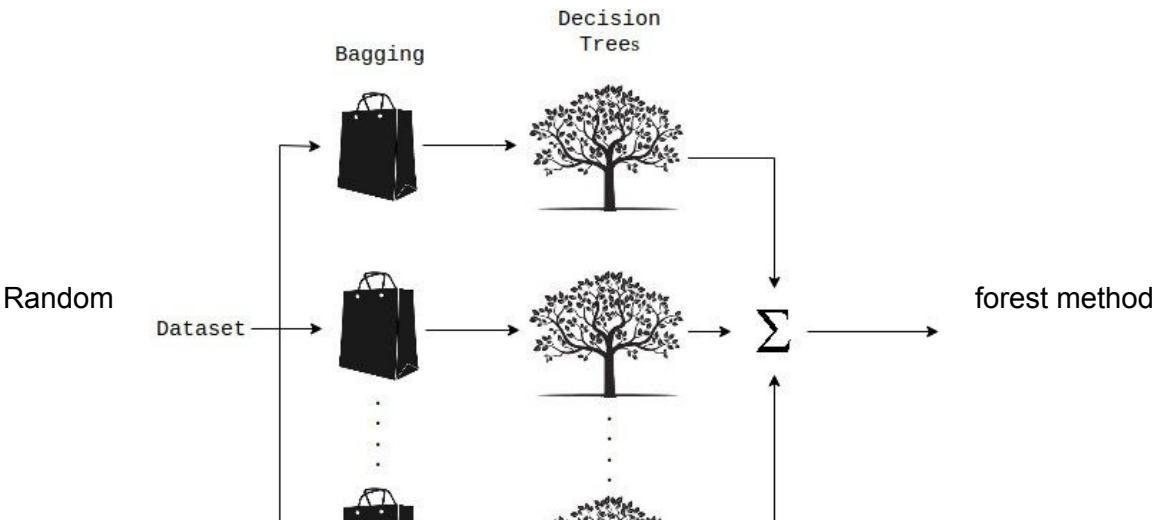
In a standard classification tree, the idea is to split the dataset based on homogeneity of data. Lets say for example we have two variables: age and weight that predict if a person is going to sign up for a gym membership or not. In our training data if it showed that 90% of the people who are older than 40 signed up, we split the data here and age becomes a top node in the tree. We can almost say that this split has made the data "90% pure". Rigorous measures of impurity, based on computing proportion of the data that belong to a class, such as entropy or Gini index are used to quantify the homogeneity in Classification trees.[2]
In a regression tree the idea is this: since the target variable does not have classes, we fit a regression model to the target variable using each of the independent variables. Then for each independent variable, the data is split at several split points. At each split point, the "error" between the predicted value and the actual values is squared to get a "Sum of Squared Errors (SSE)". The split point errors across the variables are compared and the variable/point yielding the lowest SSE is chosen as the root node/split point. This process is recursively continued.[2]

Regression trees are needed when the response variable is numeric or continuous. For example, the predicted price of a consumer good. Thus regression trees are applicable for prediction type of problems as opposed to classification.
Keep in mind that in either case, the predictors or independent variables may be categorical or numeric. It is the target variable that determines the type of decision tree needed.[2]

RANDOM FOREST REGRESSION

How does it work?

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap Aggregation, commonly known as bagging. What is bagging you may ask? Bagging, in the Random Forest method, involves training each decision tree on a different data sample where sampling is done with replacement.



Random forest method

Random Forest Regression: Process
The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees. If you want to read more on Random Forests, I have included some reference links which provide in depth explanations on this topic.

GRADIENT BOOSTED TREE REGRESSION

How does it work?
The idea of boosting came out of the idea of whether a weak learner can be modified to become better. Hypothesis boosting was the idea of filtering observations, leaving those observations that the weak learner can handle and focusing on developing new weak learns to handle the remaining difficult observations.[5]

Gradient boosting involves three elements:

- A loss function to be optimized:The loss function used depends on the type of problem being solved. It must be differentiable, but many standard loss functions are supported and you can define your own.
- A weak learner to make predictions : Decision trees are used as the weak learner in gradient boosting.Specifically regression trees are used that output real values for splits and whose output can be added together, allowing subsequent models outputs to be added and “correct” the residuals in the predictions.Trees are constructed in a greedy manner, choosing the best split points based on purity scores like Gini or to minimize the loss.Initially, such as in the case of AdaBoost, very short decision trees were used that only had a single split, called a decision stump. Larger trees can be used generally with 4-to-8 levels.It is common to constrain the weak learners in specific ways, such as a maximum number of layers, nodes, splits or leaf nodes.This is to ensure that the learners remain weak, but can still be constructed in a greedy manner.
- An additive model to add weak learners to minimize the loss function: Trees are added one at a time, and existing trees in the model are not changed.A gradient descent procedure is used to minimize the loss when adding trees.Traditionally, gradient descent is used to minimize a set of parameters, such as the coefficients in a regression equation or weights in a neural network. After calculating error or loss, the weights are updated to minimize that error.Instead of parameters, we have weak learner sub-models or more specifically decision trees. After calculating the loss, to perform the gradient descent procedure, we must add a tree to the model that reduces the loss (i.e. follow the

gradient). We do this by parameterizing the tree, then modify the parameters of the tree and move in the right direction by (reducing the residual loss.Generally this approach is called functional gradient descent or gradient descent with functions.[5]

Implementation on the Databike data set

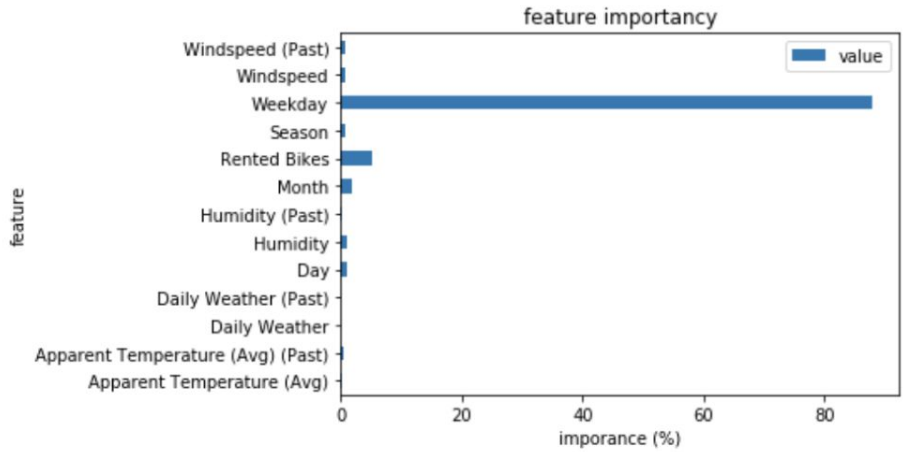
The first step is to apply the algorithm using all the features, then each time based on the features importance function, try to remove the least important columns and check for improvements in the RMSE value
The result of the testing process is described each step by a features importance plot. The results are as well listed on a table.
The figures below show the correlation between the predicted data and the real future measured data.

Most Used station

Decision tree

Testing process :

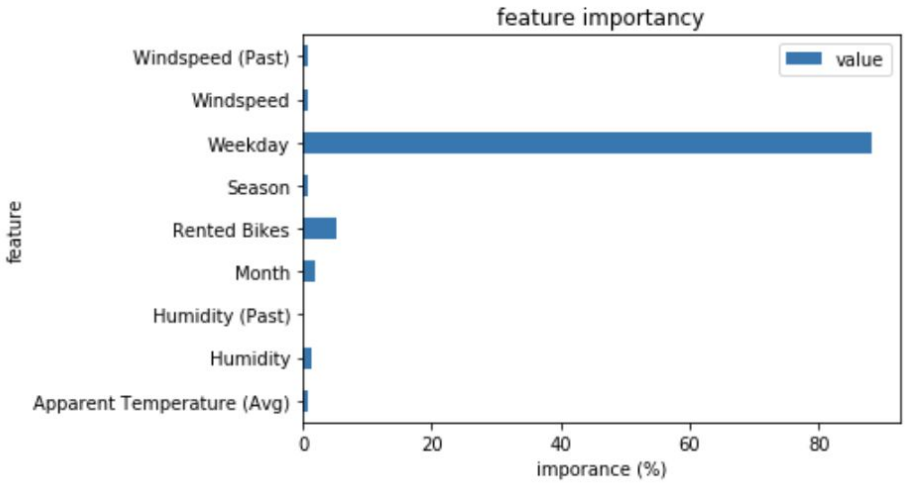
We start by generating a model considering all the features: RMSE : 71.35



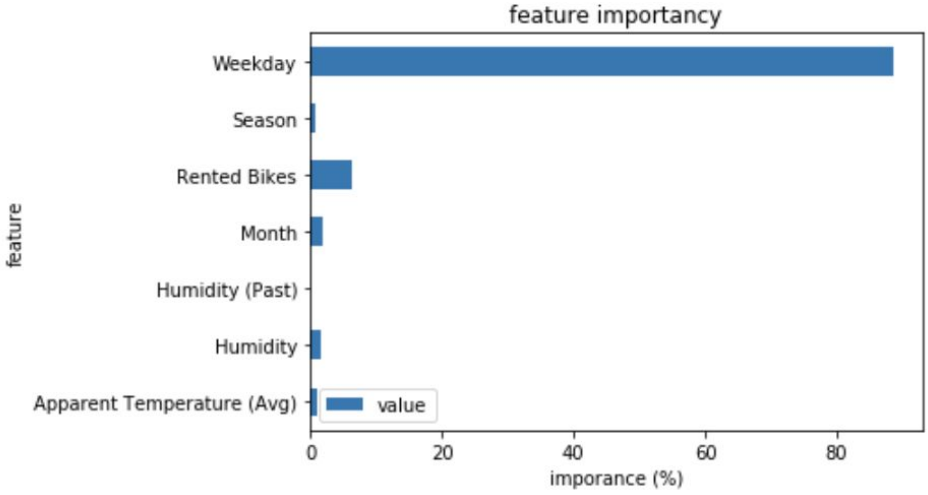
The table below shows the removed columns and RMSE results on on each test:
The choice of removing features is explained by the features importance plots.

	TEST	Day	Month	Season	Week-day	Daily Weather	Daily Weather Past	Humi-dity	Humi-dity Past	Wind-speed	Wind-speed Past	A Temp Avg	A Temp Avg Past	Rented Bikes	RMSE	RMSE (on training)
Decision tree regression	1	1	1	1	1	1	1	1	1	1	1	1	1	1	71.35	47.66
	2	0	1	1	1	0	0	1	1	1	1	1	0	1	57.55	48.06
	3	0	1	1	1	0	0	1	1	1	1	1	0	1	57.55	48.06
	4	0	1	1	1	0	0	1	1	0	0	1	0	1	59.27	48.71
	5	0	1	0	1	0	0	1	1	0	0	1	0	1	56.98	49.09
	6	0	1	0	1	0	0	1	0	0	0	1	0	1	56.98	46..09
	7	0	1	0	1	0	0	0	0	0	0	0	0	1	54.43	50.78

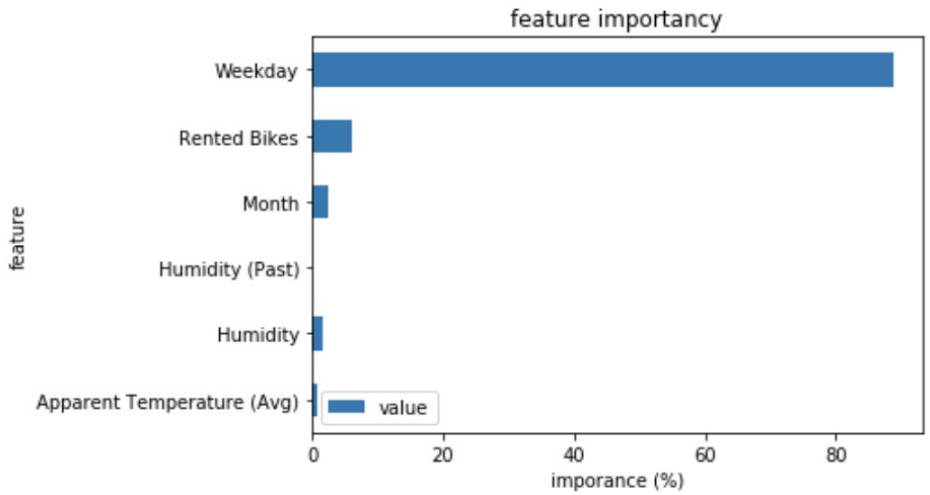
Test3: 57.55



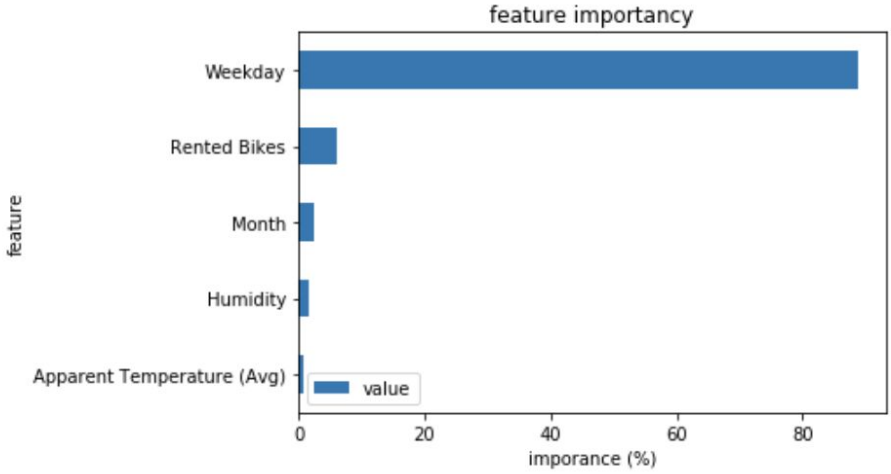
Test4: 59.27



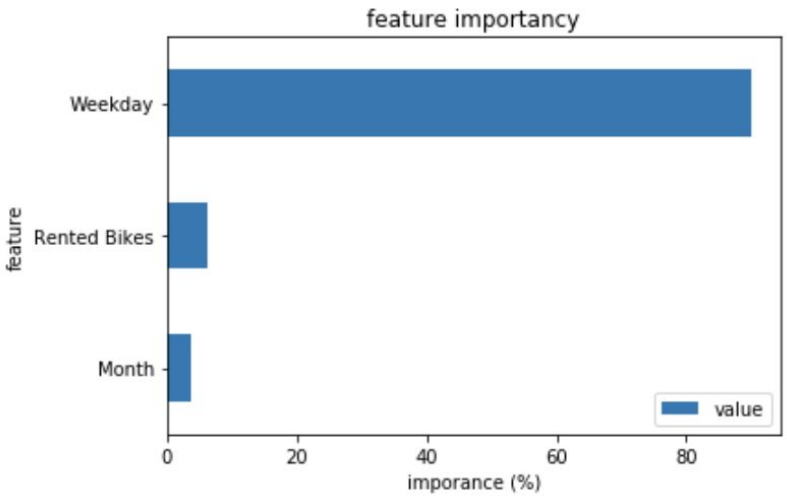
Test 5: 56.98



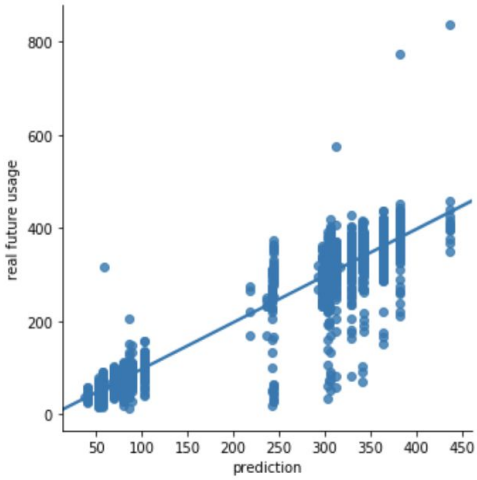
Test 6 : 56.98

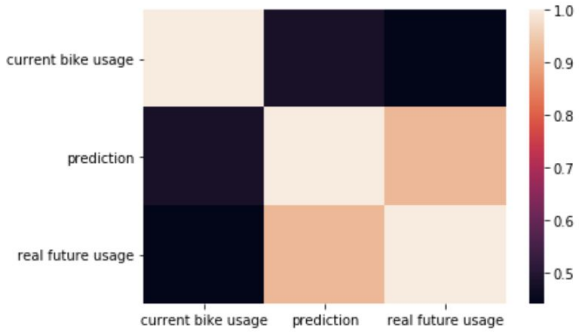


Test 7: 54.43



Final correlation results :
RMSE : 54.43

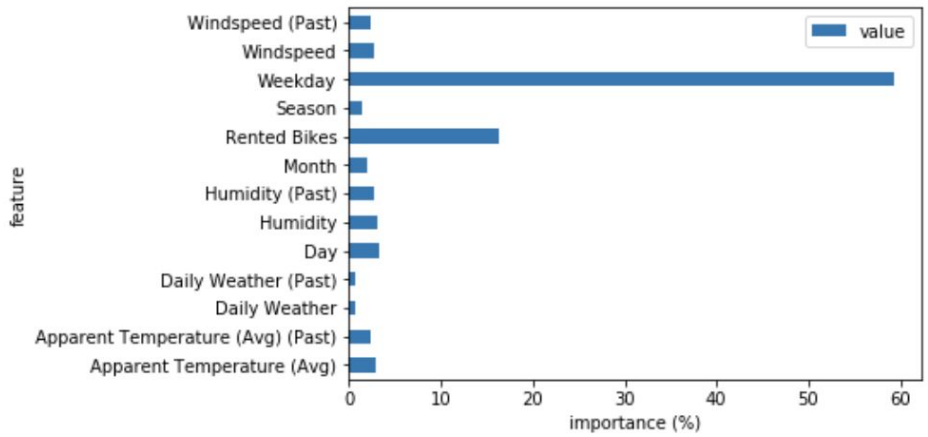




Random forest

Testing process :

We start by generating a model considering all the features: RMSE : 60.04

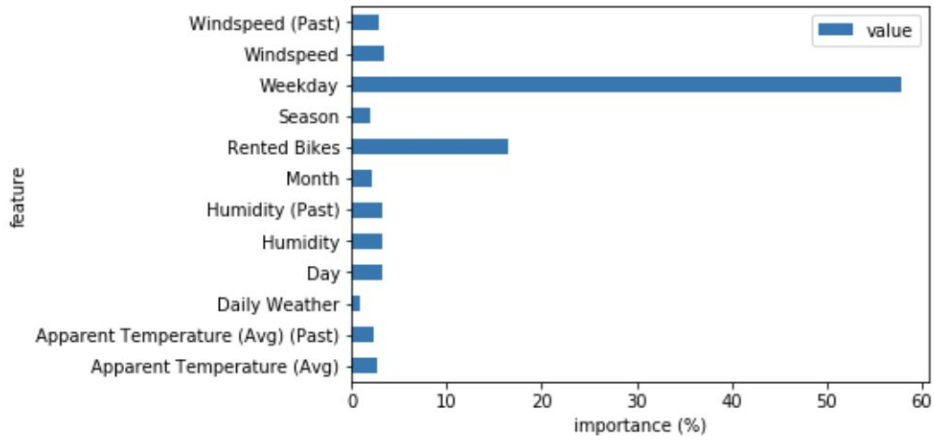


The table below shows the removed columns and RMSE results on each test:
The choice of removing features is explained by the features importance plots.

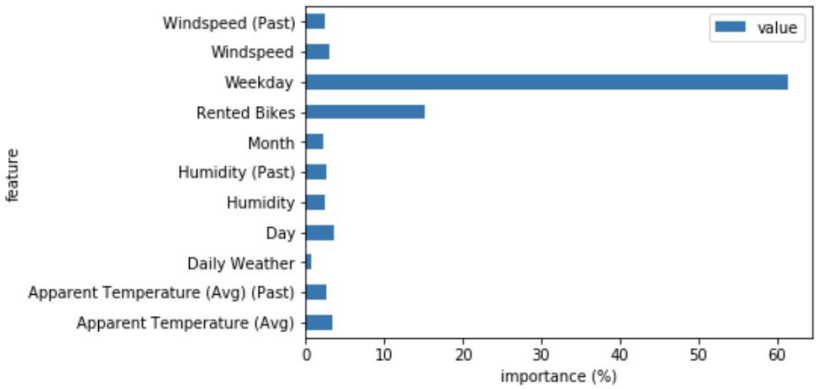
	TEST	Day	Month	Season	Week-day	Daily Weather	Daily Weather Past	Humi-dity	Humi-dity Past	Wind-speed	Wind-speed Past	A Temp Avg	A Temp Avg Past	Rented Bikes	RMSE	RMSE(on training)
Random forest regression	1	1	1	1	1	1	1	1	1	1	1	1	1	1	60.04	22.30
	2	1	1	1	1	1	0	1	1	1	1	1	1	1	62.41	22.76
	3	1	1	0	1	1	0	1	1	1	1	1	1	1	60.98	22.42
	4	1	0	1	1	1	0	0	1	0	1	1	1	1	64.47	22.61
	5	0	0	0	1	0	0	0	1	0	0	0	0	1	63.08	44.46

The process consists of removing after each test the features with the least importance.

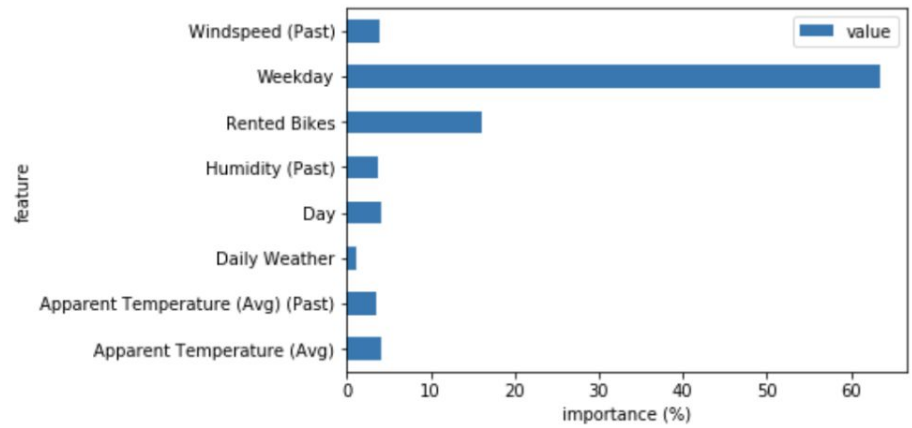
Test2 : 62.41



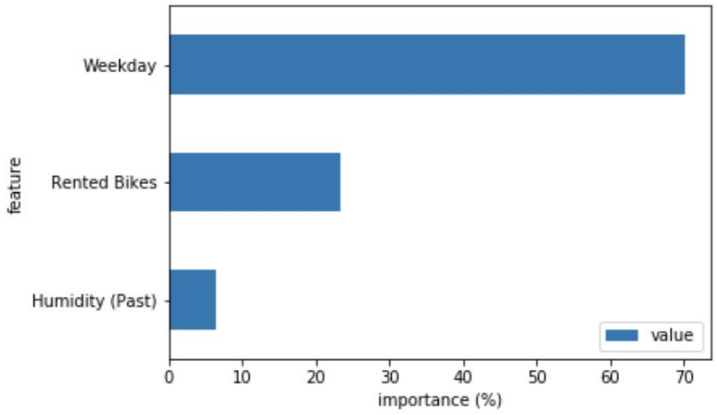
Test3 : 60.98



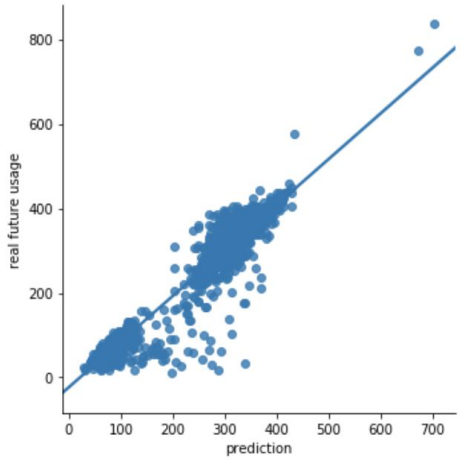
Test4 : 64,64

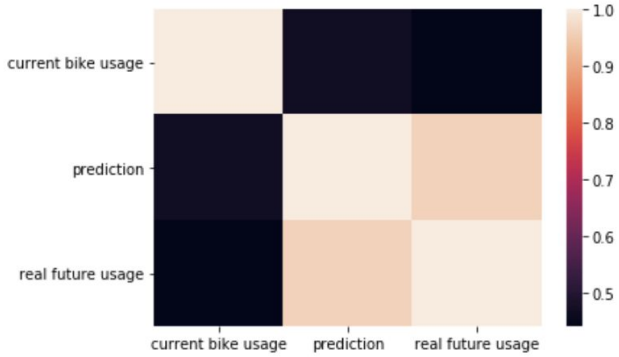


Test5 : 63.08



Final correlation results :
RMSE: 60.04

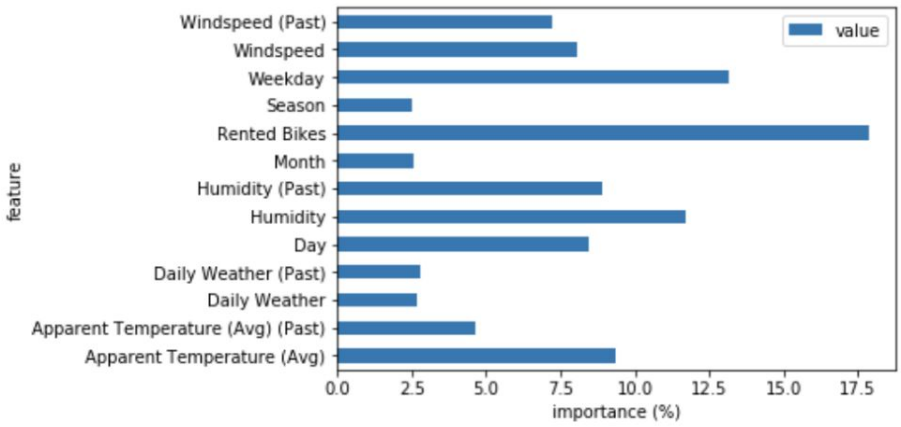




Gradient boosted tree regression

Testing process :

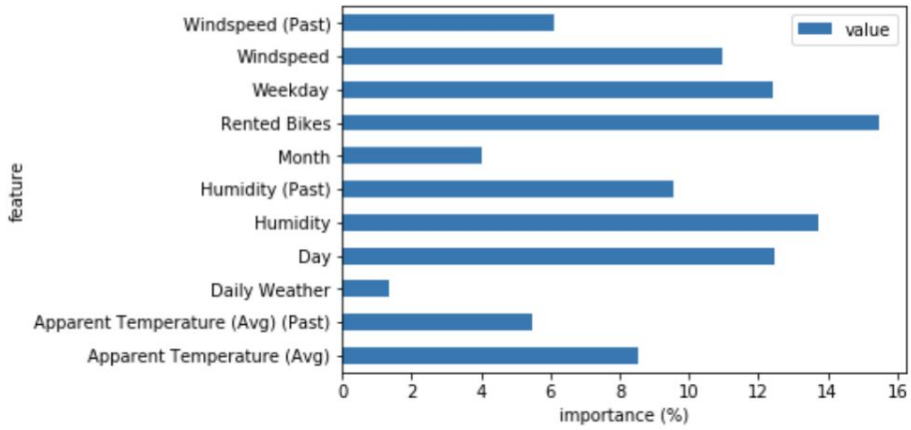
We start by generating a model considering all the features: RMSE : 81.30



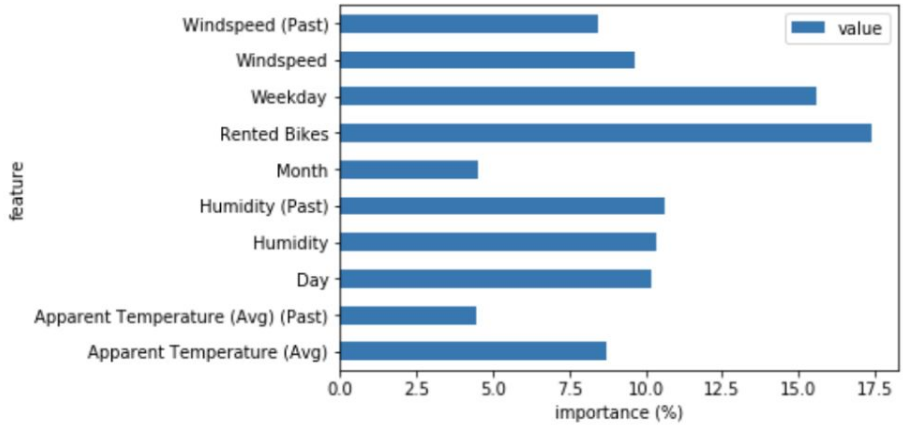
The table below shows the removed columns and RMSE results on each test:
The choice of removing features is explained by the features importance plots.

	TEST	Day	Month	Season	Week-day	Daily Weather	Daily Weather Past	Humi-dity	Humi-dity Past	Wind-speed	Wind-speed Past	A Temp Avg	A Temp Avg Past	Rented Bikes	RMSE	RMSE (on training)
Gradient boosted tree regression	1	1	1	1	1	1	1	1	1	1	1	1	1	1	81.30	29.30
	2	1	1	0	1	1	0	1	1	1	1	1	1	1	88.48	30.79
	3	0	1	0	1	0	0	1	1	1	1	1	1	1	84.92	30.53
	4	1	0	0	1	0	0	1	1	1	1	1	1	1	76.71	30.35
	5	0	0	0	1	0	0	0	1	0	0	0	0	1	63.26	42.34

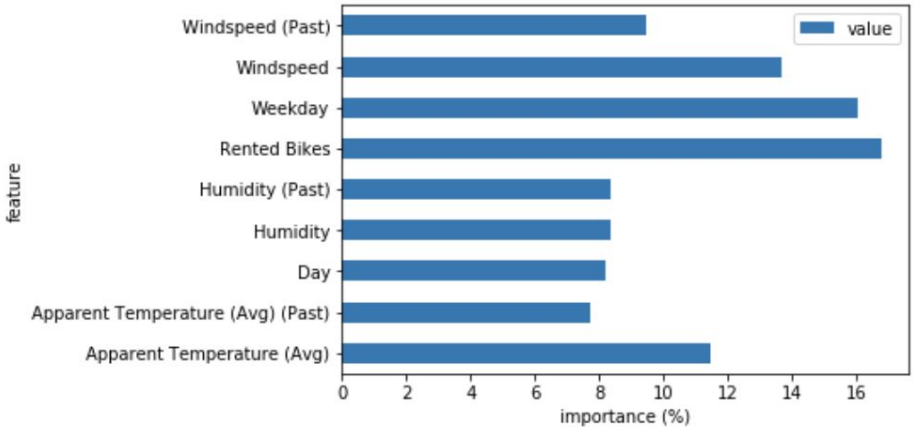
Test2 : 88.48



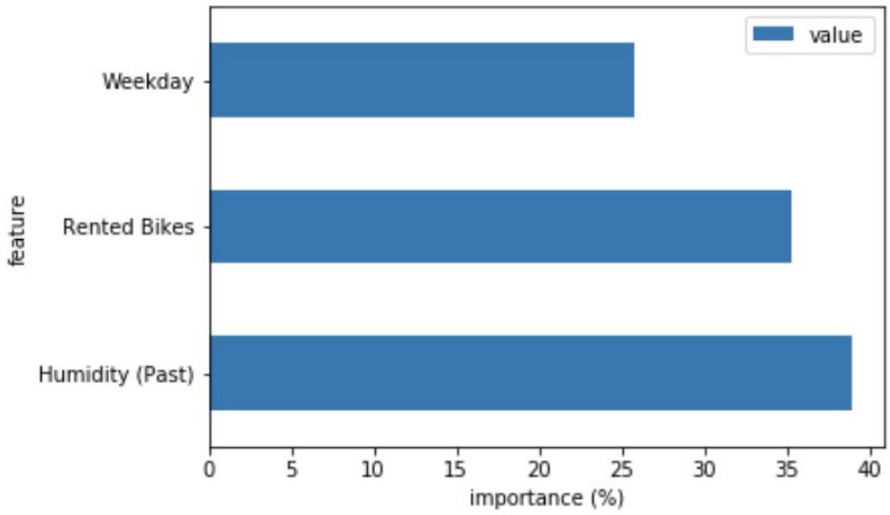
Test3: 84.92



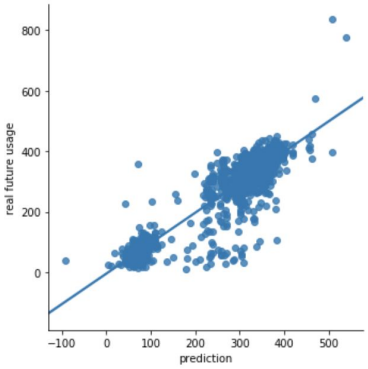
Test4: 76.71

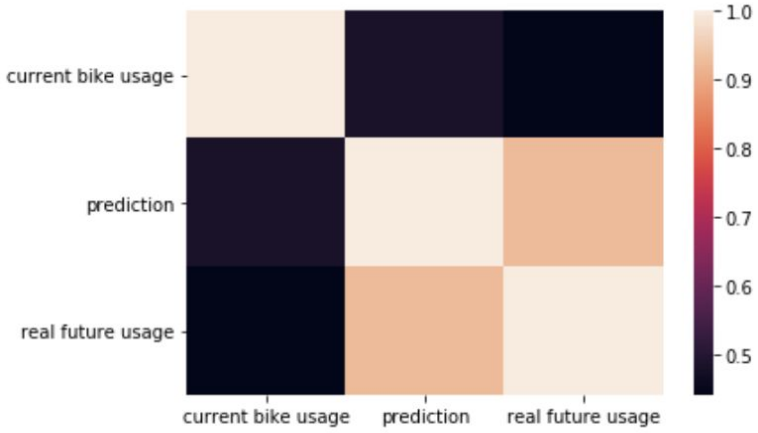


Test5: 63.26



Final correlation results :
RMSE: 63.26



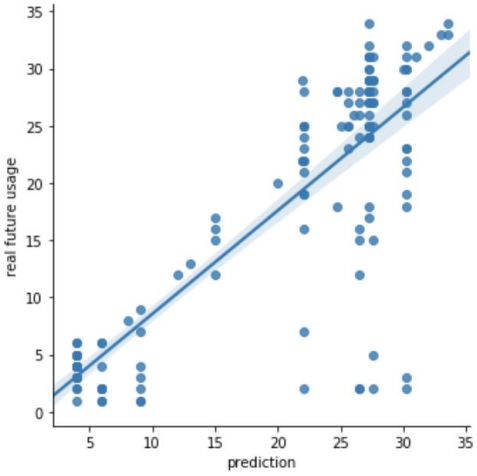


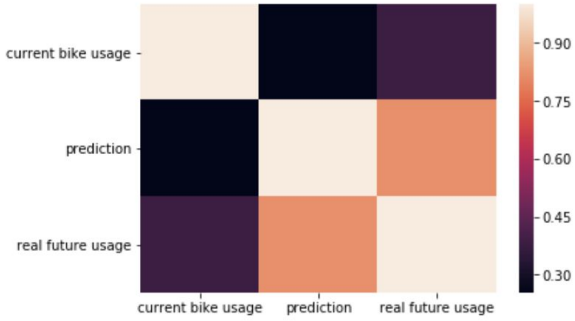
LEAST USED STATION:

Data on 145 day

Decision tree regression

Final correlation results :
RMSE: 11.72



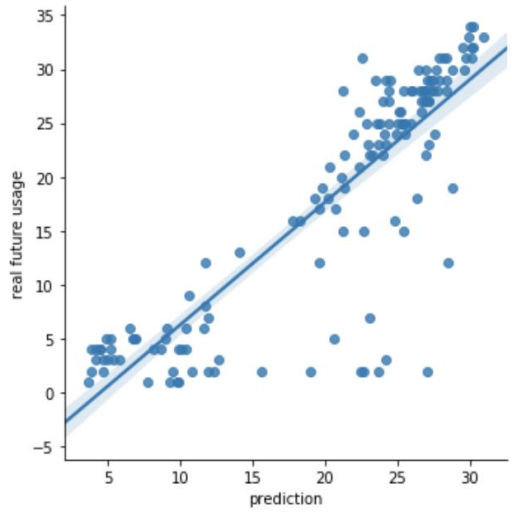


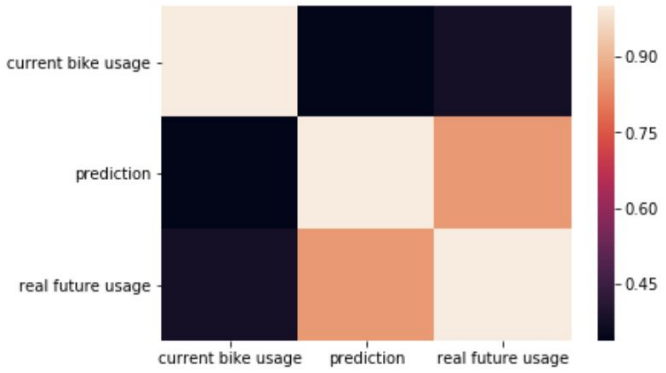
The table below shows the removed columns and RMSE results:

	TEST	Day	Month	Season	Week-day	Daily Weather	Daily Weather Past	Humi-dity	Humi-dity Past	Wind-speed	Wind-speed Past	A Temp Avg	A Temp Avg Past	Rented Bikes	RMSE	RMSE (on training)
Decision tree regression	1	1	1	1	1	1	1	1	1	1	1	1	1	1	11.72	2.46

Random forest

Final correlation results :
RMSE: 10.69



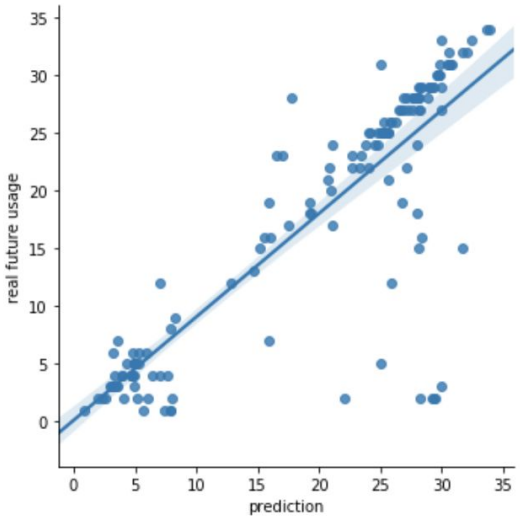


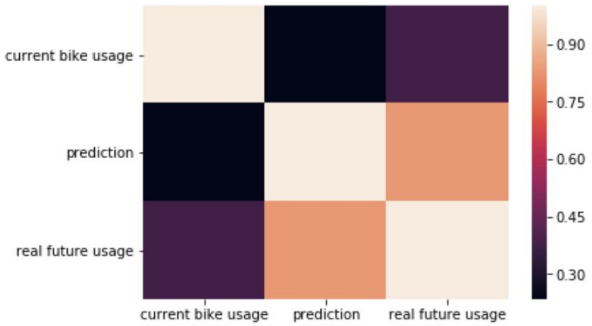
The table below shows the removed columns and RMSE results:

	TEST	Day	Month	Season	Week-day	Daily Weather	Daily Weather Past	Humi-dity	Humi-dity Past	Wind-speed	Wind-speed Past	A Temp Avg	A Temp Avg Past	Rented Bikes	RMSE	RMSE(on training)
Random forest regression	1	1	1	1	1	1	1	1	1	1	1	1	1	1	10.69	2.27

Gradient boosted tree regression

Final correlation results :
RMSE: 11.60





The table below shows the removed columns and RMSE results:

	TEST	Day	Month	Season	Week-day	Daily Weather	Daily Weather Past	Humi-dity	Humi-dity Past	Wind-speed	Wind-speed Past	A Temp Avg	A Temp Avg Past	Rented Bikes	RMSE	RMSE (on training)
Gradient boosted tree regression	1	0	0	0	1	0	0	0	1	0	0	0	0	1	11.60	1.04

References

[1]https://databricks.com/glossary/what-is-machine-learning-library
[2]http://www.simafore.com/blog/bid/62482/2-main-differences-between-classification-and-regression-trees
[3] https://www.saedsayad.com/decision_tree_reg.htm
[4] http://spark.apache.org/docs/1.2.1/mllib-guide.html
[5]https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/